INTERNATIONAL STANDARD

ISO/IEC 13818-1

First edition
1996-04-15

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

Technologies de l'information — Codage des images animées et du son associé: Systèmes

Reference number
ISO/IEC 13818-1:1996(E)

CONTENTS

		Page
Introduction		vi
Intro. 1	Transport Stream	vii
Intro. 2	Program Stream	ix
Intro. 3	Conversion between Transport Stream and Program Stream	x
Intro. 4	Packetized Elementary Stream	x
Intro. 5	Timing model	x
Intro. 6	Conditional access	xi
Intro. 7	Multiplex-wide operations	xi
Intro. 8	Individual stream operations (PES Packet Layer)	xi
Intro. 8.1	Demultiplexing	xi
Intro. 8.2	Synchronization	xii
Intro. 8.3	Relation to compression layer	xii
Intro. 9	System reference decoder	xii
Intro. 10	Applications	xii
SECTION 1 – GENERAL		1
1.1	Scope	1
1.2	Normative references	1
1.3	Identical Recommendations | International Standards	1
1.4	Additional references	2
SECTION 2 – TECHNICAL ELEMENTS		2
2.1	Definitions	2
2.2	Symbols and abbreviations	5
2.2.1	Arithmetic operators	5
2.2.2	Logical operators	6
2.2.3	Relational operators	6
2.2.4	Bitwise operators	6
2.2.5	Assignment	6
2.2.6	Mnemonics	6
2.2.7	Constants	7
2.3	Method of describing bit stream syntax	7
2.4	Transport Stream bitstream requirements	8
2.4.1	Transport Stream coding structure and parameters	8
2.4.2	Transport Stream system target decoder	8
2.4.3	Specification of the Transport Stream syntax and semantics	17
2.4.3.2	Transport Stream packet layer	18
2.4.3.3	Semantic definition of fields in Transport Stream packet layer	18
2.4.3.4	Adaptation field	20
2.4.3.5	Semantic definition of fields in adaptation field	20
2.4.3.6	PES packet	29

2.4.3.7 Semantic definition of fields in PES packet 29
2.4.3.8 Carriage of Program Streams and ISO/IEC 11172-1 Systems streams in the Transport Stream 38
2.4.4 Program specific information 39
2.4.4.1 Pointer 41
2.4.4.2 Semantics definition of fields in pointer syntax 41
2.4.4.3 Program association Table 41
2.4.4.4 Table_id assignments 42
2.4.4.5 Semantic definition of fields in program association section 42
2.4.4.6 Conditional access Table 43
2.4.4.7 Semantic definition of fields in conditional access section 43
2.4.4.8 Program Map Table 44
2.4.4.9 Semantic definition of fields in Transport Stream program map section 44
2.4.4.10 Syntax of the Private section 46
2.4.4.11 Semantic definition of fields in private section 46
2.5 Program Stream bitstream requirements 47
2.5.1 Program Stream coding structure and parameters 47
2.5.2 Program Stream system target decoder 47
2.5.2.1 System clock frequency 48
2.5.2.2 Input to the Program Stream system target decoder 49
2.5.2.3 Buffering 49
2.5.2.4 PES streams 50
2.5.2.5 Decoding and presentation 51
2.5.3 Specification of the Program Stream syntax and semantics 51
2.5.3.1 Program Stream 51
2.5.3.2 Semantic definition of fields in Program Stream 51
2.5.3.3 Pack layer of Program Stream 51
2.5.3.4 Semantic definition of fields in program stream pack 52
2.5.3.5 System header 53
2.5.3.6 Semantic definition of fields in system header 53
2.5.3.7 Packet layer of Program Stream 55
2.5.4 Program Stream map 55
2.5.4.1 Syntax of Program Stream map 55
2.5.4.2 Semantic definition of fields in Program Stream map 56
2.5.5 Program Stream directory 57
2.5.5.1 Syntax of Program Stream directory packet 57
2.5.5.2 Semantic definition of fields in Program Stream directory 57
2.6 Program and program element descriptors 59
2.6.1 Semantic definition of fields in program and program element descriptors 59
2.6.2 Video stream descriptor 60
2.6.3 Semantic definitions of fields in video stream descriptor 60
2.6.4 Audio stream descriptor 61
2.6.5 Semantic definition of fields in audio stream descriptor 61
2.6.6 Hierarchy descriptor 61
2.6.7 Semantic definition of fields in hierarchy descriptor 62
2.6.8 Registration descriptor 62
2.6.9 Semantic definition of fields in registration descriptor 63
2.6.10 Data stream alignment descriptor 63
2.6.11 Semantics of fields in data stream alignment descriptor 63
2.6.12 Target background grid descriptor 64
2.6.13 Semantics of fields in target background grid descriptor 64
2.6.14 Video window descriptor 65
2.6.15 Semantic definition of fields in video window descriptor 65
2.6.16 Conditional access descriptor 65
2.6.17 Semantic definition of fields in conditional access descriptor 66
2.6.18 ISO 639 language descriptor 66
2.6.19 Semantic definition of fields in ISO 639 language descriptor 66

2.6.20 System clock descriptor 66
2.6.21 Semantic definition of fields in system clock descriptor 67
2.6.22 Multiplex buffer utilization descriptor 67
2.6.23 Semantic definition of fields in multiplex buffer utilization descriptor 67
2.6.24 Copyright descriptor 68
2.6.25 Semantic definition of fields in copyright descriptor 68
2.6.26 Maximum bitrate descriptor 68
2.6.27 Semantic definition of fields in maximum bitrate descriptor 68
2.6.28 Private data indicator descriptor 69
2.6.29 Semantic definition of fields in Private data indicator descriptor 69
2.6.30 Smoothing buffer descriptor 69
2.6.31 Semantic definition of fields in smoothing buffer descriptor 70
2.6.32 STD descriptor 70
2.6.33 Semantic definition of fields in STD descriptor 70
2.6.34 IBP descriptor 70
2.6.35 Semantic definition of fields in IBP descriptor 70
2.7 Restrictions on the multiplexed stream semantics 71
2.7.1 Frequency of coding the system clock reference 71
2.7.2 Frequency of coding the program clock reference 71
2.7.3 Frequency of coding the elementary stream clock reference 71
2.7.4 Frequency of presentation timestamp coding 71
2.7.5 Conditional coding of timestamps 71
2.7.6 Timing constraints for scalable coding 72
2.7.7 Frequency of coding P-STD_buffer_size in PES packet headers 72
2.7.8 Coding of system header in the Program Stream 72
2.7.9 Constrained system parameter Program Stream 73
2.7.10 Transport Stream 73
2.8 Compatibility with ISO/IEC 11172 74

Annex A – CRC Decoder Model 75
A.0 CRC decoder model 75

Annex B – Digital Storage Medium Command and Control (DSM-CC) 76
B.0 Introduction 76
B.0.1 Purpose 76
B.0.2 Future applications 76
B.0.3 Benefits 76
B.0.4 Basic functions 77
B.0.4.1 Stream selection 77
B.0.4.2 Retrieval 77
B.0.4.3 Storage 77
B.1 General elements 77
B.1.1 Scope 77
B.1.2 Overview of the DSM-CC application 77
B.1.3 The transmission of DSM-CC commands and acknowledgments 78
B.2 Technical elements 79
B.2.1 Definitions 79
B.2.2 Specification of DSM-CC syntax 80
B.2.3 Semantics of fields in specification of DSM-CC syntax 80
B.2.4 Control layer 81
B.2.5 Semantics of fields in control layer 81
B.2.6 Acknowledgment layer 82
B.2.7 Semantics of fields in acknowledgment layer 83
B.2.8 Time code 83
B.2.9 Semantics of fields in time code 84

Annex C – Program Specific Information 85
C.0 Explanation of Program Specific Information in Transport Streams 85
C.1 Introduction 85

C.3 The Mapping of Sections into Transport Stream Packets 86
C.4 Repetition Rates and Random Access 86
C.5 What is a Program? 86
C.6 Allocation of program_number 87
C.7 Usage of PSI in a Typical System 87
C.8 The Relationships of PSI Structures 88
C.8.1 Program Association Table 88
C.8.2 Program Map Table 88
C.8.3 Conditional Access Table 88
C.8.4 Network Information Table 90
C.8.5 Private_section() 90
C.8.6 Descriptors 90
C.9 Bandwidth Utilization and Signal Acquisition Time 90

Annex D – Systems Timing Model and Application Implications of this Recommendation | International Standard 93
D.0 Introduction 93
D.0.1 Timing Model 93
D.0.2 Audio and Video Presentation Synchronization 94
D.0.3 System Time Clock recovery in the decoder 96
D.0.4 SCR and PCR Jitter 98
D.0.5 Clock Recovery in the Presence of Network Jitter 99
D.0.6 System clock used for chroma sub-carrier generation 100
D.0.7 Component video and audio reconstruction 101
D.0.8 Frame Slipping 101
D.0.9 Smoothing of network jitter 101

Annex E – Data Transmission Applications 103
E.0 General considerations 103
E.1 Suggestion 103

Annex F – Graphics of Syntax for this Recommendation | International Standard 104
F.0 Introduction 104
F.0.1 Transport Stream syntax 104
F.0.2 PES packet 105
F.0.3 Program Association Section 106
F.0.4 CA section 106
F.0.5 TS program map section 107
F.0.6 Private section 107
F.0.7 Program Stream 108
F.0.8 Program Stream map 109

Annex G – General Information 110
G.0 General Information 110
G.0.1 Sync Byte Emulation 110
G.0.2 Skipped picture status and decoding process 110
G.0.3 Selection of PID Values 110
G.0.4 PES start_code emulation 110

Annex H – Private Data 111
H.0 Private Data 111

Annex I – Systems conformance and real-time interface 112
I.0 Systems conformance and real-time interface 112

Annex J – Interfacing Jitter-Inducing Networks to MPEG-2 Decoders 113
J.0 Introduction 113
J.1 Network compliance models 113
J.2 Network specification for jitter smoothing 114
J.3 Example decoder implementations 115
J.3.1 Network adapter followed by an MPEG-2 decoder 115
J.3.2 Integrated decoder 115

Annex K – Splicing Transport Streams 117
K.0 Introduction 117
K.1 The different types of splicing point 117
K.1.1 Ordinary splicing points 117
K.1.2 Seamless splicing points 117
K.2 Decoder behaviour on splices 118
K.2.1 On non-seamless splices 118
K.2.2 On seamless splices 118
K.2.3 Buffer Overflow 118

Foreword

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

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO/IEC 13818-1 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, in collaboration with ITU-T. The identical text is published as ITU-T Recommendation H.222.0.

ISO/IEC 13818 consists of the following parts, under the general title *Information technology — Generic coding of moving pictures and associated audio information*:

— *Part 1: Systems*

— *Part 2: Video*

— *Part 3: Audio*

— *Part 4: Compliance testing*

— *Part 6: Extensions for DSM-CC*

— *Part 9: Extension for real time interface for systems decoders*

Annex A forms an integral part of this part of ISO/IEC 13818. Annexes B to K are for information only.

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 ITU-T Rec. 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.

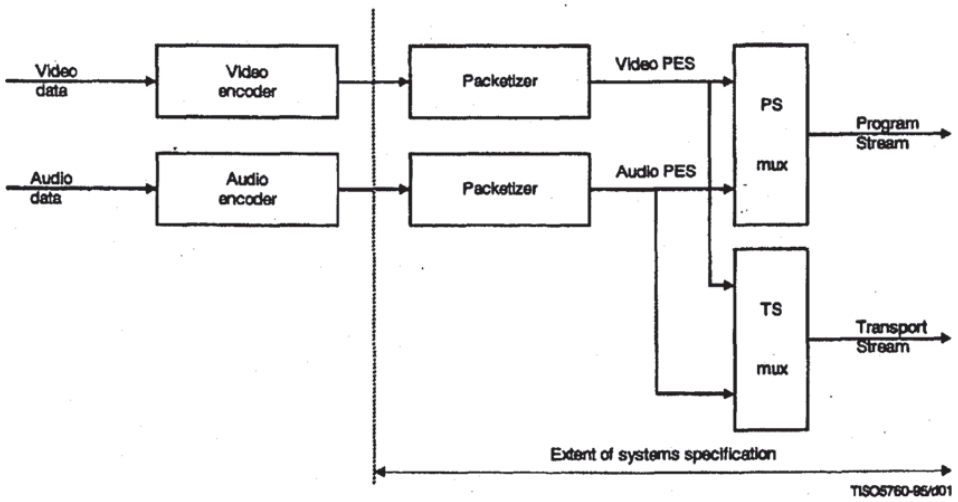


Figure Intro. 1 – Simplified overview the scope of this Recommendation | International Standard

The **Program Stream** is analogous and similar to 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 which comprise a single program to be in separate streams which 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 differ: 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 ITU-T Rec. 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 2.4.2.2.

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.

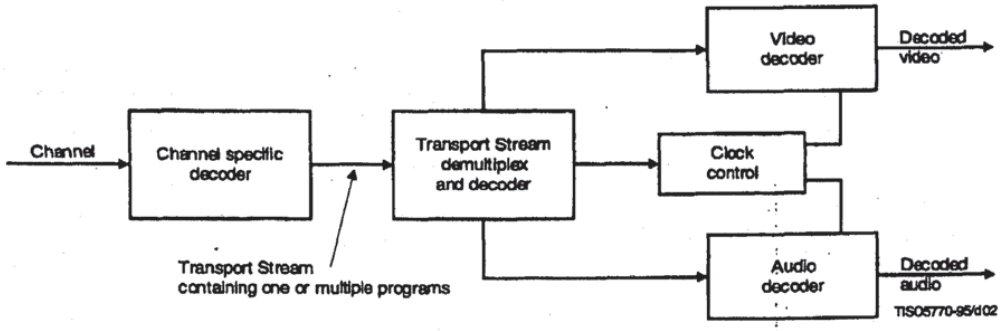


Figure Intro. 2 – Prototypical transport demultiplexing and decoding example

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.

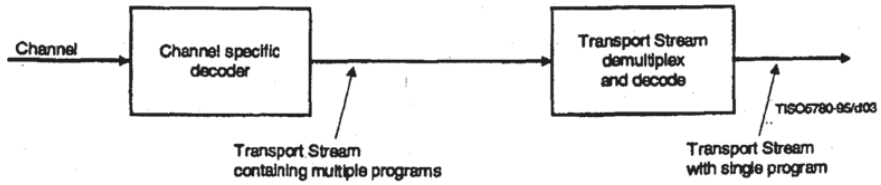


Figure Intro. 3 – Prototypical transport multiplexing example

Figure Intro. 4 illustrates a case in which an 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.

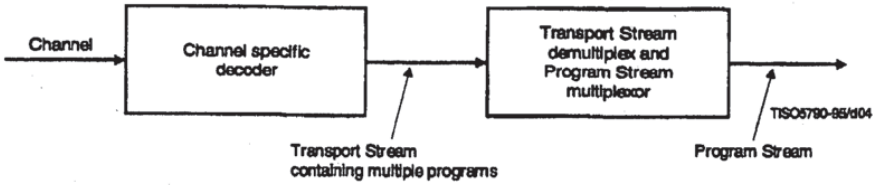


Figure Intro. 4 – Prototypical Transport Stream to Program Stream conversion

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 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.

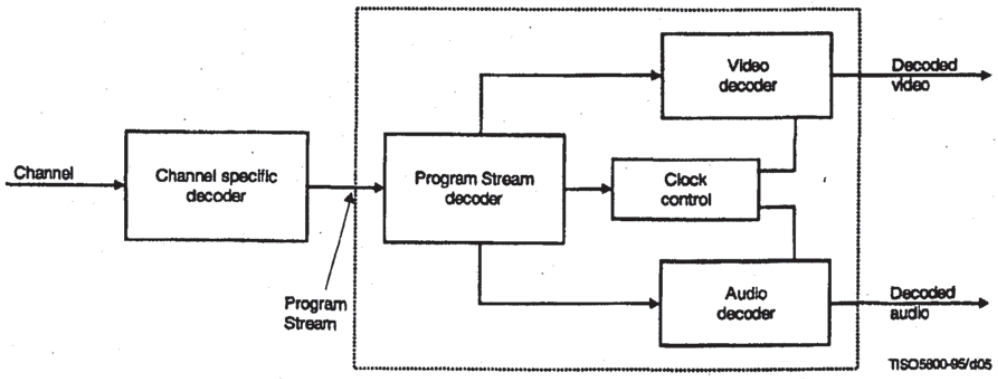


Figure Intro. 5 – Prototypical decoder for Program Streams

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. 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

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 (PTS) 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 ITU-T Rec. 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 ITU-T Rec. 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

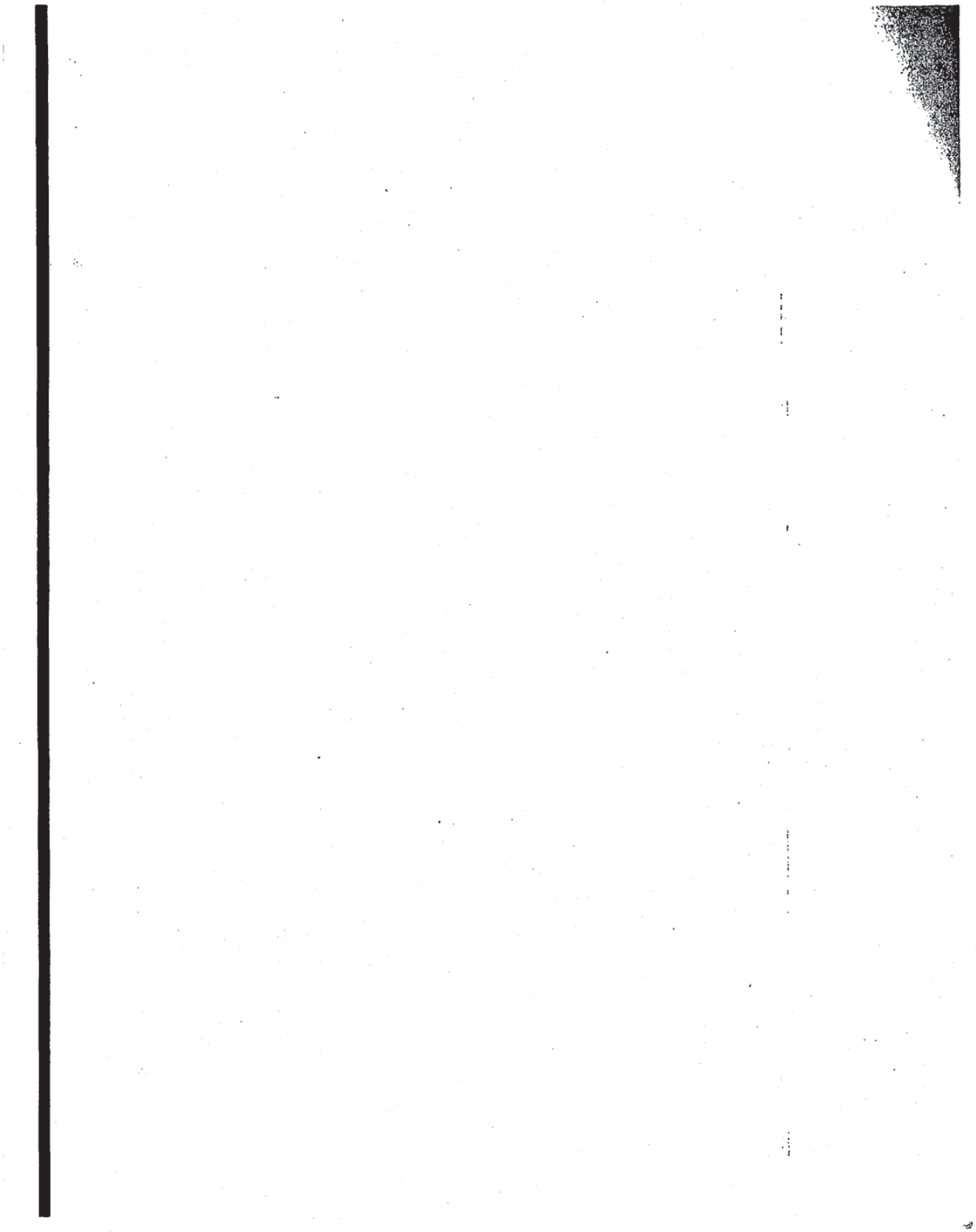
The streams defined in this Recommendation | International Standard are intended to be as useful as possible to a wide variety of applications. Application developers should select the most appropriate stream.

Modern data communications networks may be capable of supporting ITU-T Rec. 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 ITU-T Rec. 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.



PRIOR-ART_0001251

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 five 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; and
5) time identification.

An ITU-T Rec. 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.

An ITU-T Rec. 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 indicated 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.

1.3 Identical Recommendations | International Standards

- ITU-T Recommendation H.262 (1995) | ISO/IEC 13818-2: ...[1], *Information technology – Generic coding of moving pictures and associated audio information: Video.*

[1] To be published.

1.4 Additional references

- ISO 639-2: ...[2], *Codes for the representation of names of languages – Part 2: Alpha-3 code.*
- ISO 8859-1:1987, *Information processing – 8-bit single-byte coded graphic character sets – Part 1: Latin alphabet No. 1.*
- 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 13522-1: ...[2], *Information technology – Coding of Multimedia and Hypermedia information – Part 1: MHEG object representation – Base notation (ASN.1).*
- ISO/IEC 13818-3:1995, *Information technology – Generic coding of moving pictures and associated audio information – Part 3: Audio.*
- Recommendation ITU-R BT.601.3, *Encoding parameters of digital television for studios.*
- Recommendation ITU-R BT.470-2, *Television systems.*
- Recommendation ITU-R BR.648, *Digital recording of audio signals.*
- Report ITU-R BO.955.2, *Satellite sound broadcasting of vehicular, portable, and fixed receivers in the range 500-3000 MHz.*
- CCITT Recommendation J.17 (1988), *Pre-emphasis used on sound-programme circuits.*
- IEEE Standard 1180:1990, *Standard Specification for the Implementations of 8 by 8 Inverse Discrete Cosine Transform.*
- IEC Publication 908:1987, *Compact disc digital audio system.*

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.

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.

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

2.1.3 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.

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

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

[2] To be published.

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

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

2.1.8 constrained system parameter stream; CSPS (system): A Program Stream for which the constraints defined in 2.7.9 of ITU-T Rec. H.222.0 | ISO/IEC 13818-1 apply.

2.1.9 CRC: The Cyclic Redundancy Check to verify the correctness of data.

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

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

2.1.12 decoder: An embodiment of a decoding process.

2.1.13 decoding (process): The process defined in ITU-T Rec. H.222.0 | ISO/IEC 13818-1 that reads an input coded bit stream and outputs decoded pictures or audio samples.

2.1.14 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.

2.1.15 digital storage media; DSM: A digital storage or transmission device or system.

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

2.1.17 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.

2.1.18 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.

2.1.19 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.

2.1.20 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.

2.1.21 Elementary Stream Clock Reference; ESCR (system): A time stamp in the PES Stream from which decoders of PES streams may derive timing.

2.1.22 encoder: An embodiment of an encoding process.

2.1.23 encoding (process): A process, not specified in ITU-T Rec. H.222.0 | ISO/IEC 13818-1, that reads a stream of input pictures or audio samples and produces a coded bit stream conforming to ITU-T Rec. H.222.0 | ISO/IEC 13818-1.

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

2.1.25 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.

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

2.1.27 forbidden: The term "forbidden", when used in the clauses of ITU-T Rec. H.222.0 | ISO/IEC 13818-1 defining the coded bit stream, indicates that the value specified shall never be used.

2.1.28 (multiplexed) stream (system): A bit stream composed of 0 or more elementary streams combined in a manner that conforms to ITU-T Rec. H.222.0 | ISO/IEC 13818-1.

2.1.29 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 ITU-T Rec. H.222.0 | ISO/IEC 13818-1.

2.1.30 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 of ITU-T Rec. H.222.0 | ISO/IEC 13818-1.

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

2.1.32 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 of ITU-T Rec. H.222.0 | ISO/IEC 13818-1.

2.1.33 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.

2.1.34 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.

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

2.1.36 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 of ITU-T Rec. H.222.0 | ISO/IEC 13818-1.

2.1.37 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.

2.1.38 PES Stream (system): A PES Stream 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 of ITU-T Rec. H.222.0 | ISO/IEC 13818-1.

2.1.39 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.

2.1.40 presentation unit; PU (system): A decoded Audio Access Unit or a decoded picture.

2.1.41 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.

2.1.42 Program Clock Reference; PCR (system): A time stamp in the Transport Stream from which decoder timing is derived.

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

2.1.44 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 of ITU-T Rec. H.222.0 | ISO/IEC 13818-1. An example of privately defined PSI data is the non-mandatory network information table.

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

2.1.46 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 ITU-T Rec. H.222.0 | ISO/IEC 13818-1, all reserved bits shall be set to '1'.

2.1.47 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.

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

2.1.49 splicing (system): The concatenation, performed on the system level, of two different elementary streams. The resulting system stream conforms totally to ITU-T Rec. H.222.0 | ISO/IEC 13818-1. The splice may result in discontinuities in timebase, continuity counter, PSI, and decoding.

2.1.50 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 2-18 of ITU-T Rec. H.222.0 | ISO/IEC 13818-1.

2.1.51 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.

2.1.52 still picture: A coded still picture consists of a video sequence containing exactly one coded picture which is intra-coded. This 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.

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

2.1.54 System Clock Reference; SCR (system): A time stamp in the Program Stream from which decoder timing is derived.

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

2.1.56 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.

2.1.57 Transport Stream packet header (system): The leading fields in a Transport Stream packet, up to and including the continuity_counter field.

2.1.58 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.

2.2 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 $-\infty$
%	Modulus operator. Defined only for positive numbers.
Sign( )	Sign(x) = 1 if x > 0; 0 if x == 0; −1 if 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

$\sqrt{}$	Square root
$\log_{10}$	Logarithm to base ten
$\log_e$	Logarithm to base e

2.2.2 Logical operators

||	Logical OR
&&	Logical AND
!	Logical NOT

2.2.3 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

2.2.4 Bitwise operators

&	AND
|	OR
>>	Shift right with sign extension
<<	Shift left with 0 fill

2.2.5 Assignment

=	Assignment operator

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.
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 Constants

π	3.14159265359
e	2.71828182845

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 ITU-T Rec. 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 four PSI tables:

- Program Association Table;
- Program Map Table;
- Conditional Access Table;
- Network Information Table.

These tables contain the necessary and sufficient information to demultiplex and present programs. The Program Map Table, in Table 2-28, 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.

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 2-27 of the PSI).

2.4.2 Transport Stream system target decoder

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.

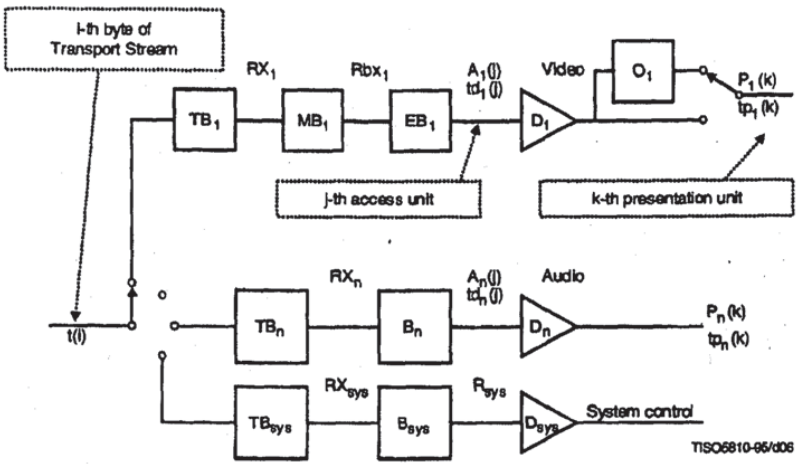


Figure 2-1 – Transport Stream system target decoder notation

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} The rate at which data are removed from TB_{sys}.

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

2.4.2.1 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:

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

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

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 $(300 \times 2^{33}/\text{system\_clock\_frequency})$ 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.

2.4.2.2 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:

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

where:

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

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

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.

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

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 is given by

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

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.

NOTE – $i'' < i \leq i'$

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.

2.4.2.3 Buffering

Complete Transport Stream packets containing data from elementary stream n, as indicated by its PID, are passed to the transport buffer for stream n, TB_n. This includes duplicate Transport Stream packets and packets with no payload. Transfer of the i-th byte from the system target decoder input to TB_n is instantaneous, so that the i-th byte enters the buffer for stream n, of size TBS_n, at time t(i).

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 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:

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

where

R_{max}[profile, level] is specified according to the profile and level which can be found in Table 8-13 of ITU-T Rec. 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 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 or 1, and all Transport Stream packets identified via the Program Association Table (see Table 2-25) 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}.

12

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 Summary of Constrained Parameters in ISO/IEC 11172-2 and Table 8-14 of ITU-T Rec. H.262 | ISO/IEC 13818-2.

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:

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

and where $VBV_{max}[profile, level]$ is defined in Table 8-14 in ITU-T Rec. H.262 | ISO/IEC 13812-2 and $R_{max}[profile, level]$ is defined in Table 8-13 in ITU-T Rec. H.262 | ISO/IEC 13818-2, and vbv buffer size is carried in the sequence header described in 6.2.2 of ITU-T Rec. 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:

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

and where $R_{max}[profile, level]$ is defined in Table 8-13 in ITU-T Rec. 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:

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

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 1 – 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

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

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

$$BS_{dec} + BS_{oh} = 2848 \text{ bytes}$$

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 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 2.4.3.6).

For Low and Main level:

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

For High-1440 and High level:

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

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 R_{bx}. 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 ITU-T Rec. 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:

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

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 2 – 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 2-29), where R_{max}[profile, level] is defined in ITU-T Rec. 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 ITU-T Rec. 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}.

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

NOTE 3 – 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 ITU-T Rec. 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.

2.4.2.4 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.

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

2.4.2.5 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.

2.4.2.6 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 ITU-T Rec. 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 Decoders buffers shall be less than or equal to one second except for still picture video data. Specifically: $td_n(j) - t(i) <= 1$ second for all j, and all bytes i in access unit $A_n(j)$.

For still picture video data, the delay is constrained by $td_n(j) - t(i) <= 60$ 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.

2.4.3 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
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 PSI data (refer to 2.4.4).

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 2-29).

When the payload of the Transport Stream packet contains PSI data, the payload_unit_start_indicator has the following significance: if the Transport Stream packet carries the first byte of a PSI 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 PSI 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 2-29).

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. | 8

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 2-25). PID value 0x0001 is reserved for the Conditional Access Table (see Table 2-27). PID values 0x0002 - 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 - 0x000F	Reserved
0x00010 ... 0x1FFE	May be assigned as network_PID, Program_map_PID, elementary_PID, or for other purposes
0x1FFF	Null packet
NOTE – The transport packets with PID values 0x0000, 0x0001, and 0x0010 - 0x1FFE are allowed to carry a PCR.	

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

ITU-T Rec. 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), PSI sections (refer to 2.4.4), packet stuffing bytes after PSI 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 below.

2.4.3.4 Adaptation field

See Table 2-6.

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 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 PSI, 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.

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
reserved	5	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
}		
for (i = 0; i < N; i++) {		
reserved	8	bslbf
}		
}		
for (i = 0; i < N; i++) {		
stuffing_byte	8	bslbf
}		
}		
}		

21

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:

- Video – The first byte of a video sequence header.
- Audio – The first byte of an audio frame.

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 or in the case of video, the first byte of an elementary stream access point or a sequence_end_code followed by an 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 PSI 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 the first byte of a video sequence header if the PES stream type (refer to Table 2-29) is 1 or 2, or shall contain the first byte of an audio frame if the PES stream type is 3 or 4. In addition, in the case of video, a presentation timestamp shall be present in the PES packet containing the first picture following the sequence header. 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 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 video, this field may be set to '1' only if the payload contains one or more bytes from an intra-coded slice. 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', it indicates that a splice_countdown field shall be present in the associated 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:

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

where

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

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

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 purposes of this subclause, an access point is defined as follows:

- Video – The first byte of a video_sequence_header.
- Audio – The first byte of an audio frame.

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 not be such that private data extends 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 not be set to '1' in Transport Stream packets in which the splicing_point_flag is not set to '1'. 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, if the elementary stream carried in this PID is an audio stream, the splice_type field shall be set to '0000'. If the elementary stream carried in this PID is a video stream, it shall fulfill the constraints indicated by the splice_type value.

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 fulfills:

$$\text{offset} = t_1(i) - t(i)$$

$$\text{ltw\_offset} = \text{offset}//1$$

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:

$$t_1(A_{i+j}) = t_1(A_i) + j * 188 * 8\text{-bits/byte} / R$$

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:

$$\text{offset} = t_1(A_i) - t(A_i)$$

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 an audio stream, this field shall have the value '0000'. If the elementary stream carried in that PID is a video stream, 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 2-16.

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 ITU-T Rec. H.262 | ISO/IEC 13818-2.
   - $(t_{n+1} - t_n)$ is defined in C.9 through C.12 of ITU-T Rec. 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)

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
1100-1111	User-defined

Table 2-8 – Splice parameters Table 2

Main Profile Low Level, SNR Profile Low Level (both layers), High Profile MainLevel (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
1100-1111	User-defined

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) 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 = 45.0×10^6 bit/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
1100-1111	User-defined

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) 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
1100-1111	User-defined

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
1100-1111	User-defined

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
1100-1111	User-defined

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
1100-1111	User-defined

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
1100-1111	User-defined

Table 2-15 – Splice parameters Table 9

High Profile High Level (base layer) 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
1100-1111	User-defined

Table 2-16 – Splice parameters Table 10

High Profile High Level (all 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
1100-1111	User-defined

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).

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 2-17.

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 2-18. In Transport Streams, the stream_id may be set to any valid value which correctly describes the elementary stream type as defined in Table 2-18. In Transport Streams, the elementary stream type is specified in the Program Specific Information as specified in 2.4.4.

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 2-19).

Table 2-17 – 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
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
}		

Table 2-17 – PES packet *(continued.)*

Syntax	No. of bits	Mnemonic
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
}		
if (PES_extension_flag_2 == '1') {		
marker_bit	1	bslbf
PES_extension_field_length	7	uimsbf
for (i = 0; i < PES_extension_field_length; 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
}		
}		

Table 2-17 – PES packet *(concluded)*

Syntax	No. of bits	Mnemonic
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
}		
}		
}		

Table 2-18 – Stream_id assignments

stream_id	Note	Stream coding
1011 1100	1	program_stream_map
1011 1101	2	private_stream_1
1011 1110		padding_stream
1011 1111	3	private_stream_2
110x xxxx		ISO/IEC 13818-3 or ISO/IEC 11172-3 audio stream number x xxxx
1110 xxxx		ITU-T Rec. H.262 | ISO/IEC 13818-2 or ISO/IEC 11172-2 video stream number xxxx
1111 0000	3	ECM_stream
1111 0001	3	EMM_stream
1111 0010	5	ITU-T Rec. H.222.0 | ISO/IEC 13818-1 Annex B or ISO/IEC 13818-6_DSMCC_stream
1111 0011	2	ISO/IEC_13522_stream
1111 0100	6	ITU-T Rec. H.222.1 type A
1111 0101	6	ITU-T Rec. H.222.1 type B
1111 0110	6	ITU-T Rec. H.222.1 type C
1111 0111	6	ITU-T Rec. H.222.1 type D
1111 1000	6	ITU-T Rec. H.222.1 type E
1111 1001	7	ancillary_stream
1111 1010 ... 1111 1110		Reserved data stream
1111 1111	4	program_stream_directory

The notation x means that the value '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.

NOTES

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

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

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

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

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

6 This stream_id is associated with stream_type 0x09 in Table 2-29.

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 2.4.3.7).

Table 2-19 – 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 start code or audio syncword 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 2-47 and Table 2-48 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 is 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 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 PES packet if the first byte of the picture start code is present in the PES packet.

For audio presentation units (PUs), video PUs in low_delay sequences, and B-pictures, the presentation time $tp_n(k)$ shall be equal to the decoding time $td_n(k)$.

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 $tp_n(k)$ shall be equal to the decoding time $td_n(k')$ 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 $tp_n(k)$ and $td_n(k)$ shall be the same as if the original stream had continued without a discontinuity and without ending.

NOTE 1 – A low_delay sequence is a video sequence in which the low_delay flag is set (refer to 6.2.2.3 of ITU-T Rec. H.262 | ISO/IEC 13818-2).

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.

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:

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

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

In the case of 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 PES packet if the first byte of the picture start code is present in the PES packet.

In the case of scalable coding refer to 2.7.6.

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:

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

where

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

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

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. 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 2.4.2.3.

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 ITU-T Rec. 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 ITU-T Rec. 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 2-20 – 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 2-21, 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 2-22.

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 ITU-T Rec. H.262 | ISO/IEC 13818-2 Video progressive sequences, the picture should be displayed for N * rep_cntrl picture duration.

In the case of ITU-T Rec. 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 ITU-T Rec. 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 2-21, 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 2-21, 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 2-22, "Coefficient selection values".

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 ITU-T Rec. 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 ITU-T Rec. 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 ITU-T Rec. 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 2-21.

Table 2-21 – 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 ITU-T Rec. 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 2-22.

Table 2-22 – 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 2 – 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 ITU-T Rec. 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:

$$\text{if } (P\text{-}STD\_buffer\_scale == 0)$$
$$BS_n = P - STD\_buffer\_size \times 128 \qquad (2\text{-}16)$$

else

$$BS_n = P - STD\_buffer\_size \times 1024 \qquad (2\text{-}17)$$

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

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.

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 ITU-T Rec. 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.

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, ITU-T Rec. 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 2-18). 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 ITU-T Rec. H.262 | ISO/IEC 13818-1 PES packet headers. ISO/IEC 11172-1 packet header field values are copied to the equivalent ITU-T Rec. H.262 | ISO/IEC 13818-1 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 ITU-T Rec. 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 labeled 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 four table structures as shown in Table 2-23. 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 2-23 – Program specific information

Structure Name	Stream Type	PID number	Description
Program Association Table	ITU-T Rec. H.222.0 | ISO/IEC 13818-1	0x00	Associates Program Number and Program Map Table PID
Program Map Table	ITU-T Rec. H.222.0 | ISO/IEC 13818-1	Assignment indicated in the PAT	Specifies PID values for components of one or more programs
Network Information Table	Private	Assignment indicated in the PAT	Physical network parameters such as FDM frequencies, Transponder Numbers, etc.
Conditional Access Table	ITU-T Rec. H.222.0 | ISO/IEC 13818-1	0x01	Associates one or more (private) EMM streams each with a unique PID value

ITU-T Rec. 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 ITU-T Rec. H.222.0 | ISO/IEC 13818-1 defined PSI table into Transport Stream packets.

Along with ITU-T Rec. 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 ITU-T Rec. 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 ITU-T Rec. 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 2-24.

Adaptation fields may occur in Transport Stream packets carrying PSI sections.

Within a Transport Stream, packet stuffing bytes of value 0xFF may be found after the last byte of a section, in which case all following 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). 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 labeled 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 ITU-T Rec. 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.

There are no restrictions on the occurrence of start codes, sync bytes or other bit patterns in PSI data, whether this Recommendation | International Standard or private.

2.4.4.1 Pointer

The pointer_field syntax is defined in Table 2-24.

Table 2-24 – 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 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 2.25).

Table 2-25 – 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 PSI section as shown in Table 2-26.

Table 2-26 – table_id assignment values

Value	Description
0x00	program_association_section
0x01	conditional_access_section(CA_section)
0x02	TS_program_map_section
0x03 - 0x3F	ITU-T Rec. H.222.0 | ISO/IEC 13818-1 reserved
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 2-26.

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 (CA) Table provides the association between one or more 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 2-27.

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

Table 2-27 – 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 2-26.

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 2-28).

Table 2-28 – 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 2-26.

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 2-29.

Table 2-29 – Stream type assignments

Value	Description
0x00	ITU-T | ISO/IEC reserved
0x01	ISO/IEC 11172-2 Video
0x02	ITU-T Rec. H.262 | ISO/IEC 13818-2 Video or ISO/IEC 11172-2 constrained parameter video stream
0x03	ISO/IEC 11172-3 Audio
0x04	ISO/IEC 13818-3 Audio
0x05	ITU-T Rec. H.222.0 | ISO/IEC 13818-1 private_sections
0x06	ITU-T Rec. H.222.0 | ISO/IEC 13818-1 PES packets containing private data
0x07	ISO/IEC 13522 MHEG
0x08	Annex A - DSM CC
0x09	ITU-T Rec. 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	ISO/IEC 13818-1 auxiliary
0x0F - 0x7F	ITU-T Rec. H.222.0 | ISO/IEC 13818-1 reserved
0x80 - 0xFF	User private

NOTE – An ITU-T Rec. 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 Program Stream Directory and Program Stream Map.

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 Table 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 2-30).

Table 2-30 – 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 2-26 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.

11/13/206

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.

2.5 Program Stream bitstream requirements

2.5.1 Program Stream coding structure and parameters

The ITU-T Rec. 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 2-18) 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 play-back of Program Streams from a variety of decoders with different architectures or timing schedules.

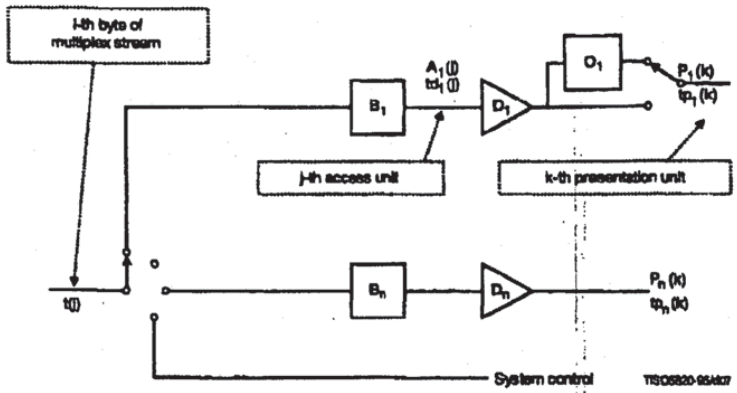


Figure 2-2 – Program Stream system target decoder notation

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:

$$SCR(i) = SCR\_base(i) \times 300 + SCR\_ext(i) \tag{2-18}$$

where

$$SCR\_base(i) = ((system\_clock\_frequency \times t(i))\ DIV\ 300)\ \%\ 2^{33} \tag{2-19}$$

$$SCR\_ext(i) = ((system\_clock\_frequency \times t(i))\ DIV\ 1)\ \%\ 300 \tag{2-20}$$

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.

$$t(i) = \frac{SCR(i')}{system\_clock\_frequency} + \frac{i - i'}{program\_mux\_rate \times 50} \tag{2-21}$$

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 equation 2-16 and equation 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 ITU-T Rec. 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:

$$0 \leq F_n(t) \leq BS_n$$

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 2.4.2.6) 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 1 second except for still picture video data. 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 then the delay is constrained by:

$$td_n(j) - t(i) \leq 1 \text{ sec}$$

else in the case of still picture video data the delay is constrained by:

$$td_n(j) - t(i) \leq 60 \text{ sec}$$

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 ITU-T Rec. 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.

Buffer sizes BS_n in the PES-STD model are defined as follows:

- *For ITU-T Rec. H.262 | ISO/IEC 13818-2 video:*

$$BS_n = VBV_{max}[profile, level] + BS_{oh}$$

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 ITU-T Rec. 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:*

$$BS_n = 2848 \text{ bytes}$$

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 2.4.2.4, and 2.4.2.5 respectively.

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 2-31.

Table 2-31 – 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 Table 2-32.

Table 2-32 – Program Stream pack

Syntax	No. of bits	Mnemonic

pack() {
  pack_header()
  while (nextbits() == packet_start_code_prefix) {
  PES_packet()
  }
}		

Table 2-33 – Program Stream pack header

Syntax	No. of bits	Mnemonic

pack_header() {
  pack_start_code
  '01'
  system_clock_reference_base [32..30]
  marker_bit
  system_clock_reference_base [29..15]
  marker_bit
  system_clock_reference_base [14..0]
  marker_bit
  system_clock_reference_extension
  marker_bit
  program_mux_rate
  marker_bit
  marker_bit
  reserved
  pack_stuffing_length
  for (i = 0; i < pack_stuffing_length; i++) {
  stuffing_byte
  }
  if (nextbits() == system_header_start_code) {
  system_header ()
  }
}
	32
	2
	3
	1
	15
	1
	15
	1
	9
	1
	22
	1
	1
	5
	3
	
	8
		bslbf
		bslbf
		bslbf
		bslbf
		bslbf
		bslbf
		bslbf
		bslbf
		uimsbf
		bslbf
		uimsbf
		bslbf
		bslbf
		bslbf
		uimsbf
		
		bslbf

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 an ITU-T Rec. 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 2-34.

Table 2-34 – 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
'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 ITU-T Rec. H.222.0 | ISO/IEC 13818-1 stream shall adhere to the following linear equation:

$$SCR\_base(i) = ((c1 * i + c2)\ DIV\ 300)\ \%\ 2^{33} \qquad (2\text{-}22)$$

$$SCR\_ext(i) = ((c1 * i + c2)\ DIV\ 300)\ \%\ 300 \qquad (2\text{-}23)$$