# TITLE PAGE PROVIDED BY ISO

## CD 11172-1

## CODING OF MOVING PICTURES AND ASSOCIATED AUDIO -- FOR DIGITAL STORAGE MEDIA AT UP TO ABOUT 1.5 Mbit/s --

## Part 1: Systems

## CONTENTS

## FOREWORD

This standard is a committee draft that was submitted for approval to ISO-IEC/JTC1 SC29 on 22 November 1991. It was prepared by SC29/WG11, also known as MPEG (Moving Pictures Expert Group). MPEG was formed in 1988 to establish a standard for the coded representation of moving pictures and associated audio stored on digital storage media.

This standard is published in four parts. Part 1 - systems - specifies the system coding layer of the standard. It defines a multiplexed structure for combining audio and video data and means of representing the timing information needed to replay synchronized sequences in real-time. Part 2 - video - specifies the coded representation of video data and the decoding process required to reconstruct pictures. Part 3 - audio - specifies the coded representation of audio data. Part 4 - conformance testing - is still in preparation. It will specify the procedures for determining the characteristics of coded bit streams and for testing compliance with the requirements stated in Parts 1, 2 and 3.

In Part 1 of this standard all annexes are informative and contain no normative requirements.

In Part 2 of this standard 2-Annex A, 2-Annex B and 2-Annex C contain normative requirements and are an integral part of this standard. 2-Annex D and 2-Annex E are informative and contain no normative requirements.

In Part 3 of this standard 3-Annex A and 3-Annex B contain normative requirements and are an integral part of this standard. All other annexes are informative and contain no normative requirements.

## INTRODUCTION - PART 1: SYSTEMS

The systems specification addresses the problem of combining one or more data streams from the video and audio parts of this standard with timing information to form a single stream. Once combined into a single stream, the data are in a form well suited to digital storage or transmission. The syntactical and semantic rules imposed by this systems specification enable synchronized playback without overflow or underflow of decoder buffers under a wide range of stream retrieval or receipt conditions. 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 encoder or decoders. However, bitstream properties do impose functional and performance requirements on encoders and decoders. 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 and decoders.

A prototypical audio/video decoder system is depicted in figure I.1 to illustrate the function of an ISO 11172 decoder. The architecture is not unique -- System Decoder functions including decoder timing control might equally well be distributed among elementary stream decoders and the Medium Specific Decoder -- but this figure is useful for discussion. The prototypical decoder design does not imply any normative requirement for the design of an ISO 11172 decoder. Indeed non-audio/video data is also allowed, but not shown.

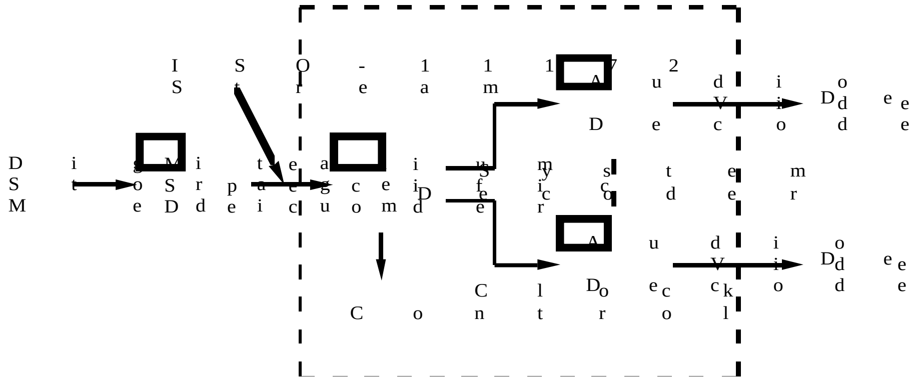


**Figure I.1 -- Prototypical ISO 11172 System Decoder**

The prototypical ISO 11172 decoder shown in figure I.1 is composed of System, Video, and Audio decoders conforming to Parts 1, 2, and 3, respectively, of this Standard. In this decoder the multiplexed coded representation of one or more audio and/or video streams is assumed to be stored on a DSM (or network) in some medium-specific format. The medium specific format is not governed by this International Standard, nor is the medium-specific decoding part of the prototypical ISO 11172 decoder.

The prototypical decoder accepts as input an ISO 11172 stream and relies on a System Decoder to extract timing information from the stream. The System 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 System Decoder, the Video and Audio Decoders, and the Medium Specific Decoder. The Video and Audio Decoders are synchronized with each other and with the DSM using this timing information.

ISO 11172 streams are constructed in two layers: a system layer and a compression layer. The input stream to the System 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 System Decoder either apply to the entire ISO 11172 stream ("multiplex-wide operations"), or to individual elementary streams ("stream-specific operations"). The ISO 11172 system layer is divided into two sub-layers, one for multiplex-wide operations (the pack layer), and one for stream-specific operations (the packet layer).

## I.1 Multiplex-wide Operations (Pack layer)

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

ISO 11172 streams are composed of packs whose headers facilitate the above tasks. Pack headers specify intended times at which each byte is to enter the system decoder from the DSM, 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.

An additional multiplex-wide operation is a decoder's ability to establish what resources are required to decode an ISO 11172 stream. The first pack of each ISO 11172 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.

## I.2 Individual Stream Operations (Packet Layer)

The principal stream-specific operations are 1) demultiplexing, and 2) synchronizing playback of multiple elementary streams. These topics are discussed next.

### I.2.1 Demultiplexing

On encoding, ISO 11172 streams are formed by multiplexing elementary streams. Elementary streams may include private, reserved, and padding streams in addition to ISO 11172 audio and video streams. The streams are temporally subdivided into packets, and the packets are serialized. A packet contains coded bytes from one and only one elementary stream.

On decoding, demultiplexing is required to reconstitute elementary streams from the multiplexed ISO 11172 stream. stream_id codes in packet headers make this possible.

### I.2.2 Synchronization

Synchronization among multiple streams is effected with presentation time stamps in the ISO 11172 bitstream. The time stamps are in units of 90kHz. Playback of N streams is synchronized by adjusting the playback of all streams to a master time base rather than by adjusting the playback of one stream to match that of another. The master time base may be one of the N decoders' clocks, the DSM or channel clock, or it may be some external clock.

Because presentation time-stamps apply to the decoding of individual elementary streams, they reside in the packet layer. 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 is also possible with DSM timing time stamps in the multiplexed data stream.

### I.2.3 Relation to Compression Layer

The packet layer is independent of the compression layer in some senses, but not in all. It is independent in the sense that packets need not start at compression layer start codes, as defined in parts 2 and 3. For example, a video packet may start at any point in the video stream. However, time stamps encoded in packet headers apply to presentation times of compression layer constructs (namely, presentation units).

## I.3 System Reference Decoder

Part 1 of ISO 11172 employs a "System Target Decoder," (STD) to provide a formalism for timing and buffering relationships. Because the STD is parameterized in terms of ISO 11172 fields (for example, buffer sizes) each ISO 11172 stream leads to its own parameterization of the STD. It is up to encoders to ensure that bitstreams they produce play in normal speed, forward play on corresponding STDs. 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.

# 1 GENERAL NORMATIVE ELEMENTS

## 1.1 Scope

This part of ISO 11172 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 this International Standard. The system layer supports five basic functions: 1) the synchronization of multiple compressed streams on playback, 2) the interleaving of multiple compressed streams into a single stream, 3) the initialization of buffering for playback start up, 4) continuous buffer management, and 5) time identification.

An ISO 11172 multiplexed bit stream is constructed in two layers: the outer most layer is the system layer, and the inner most 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 are not defined by the specification, but are supported by the system layer providing that the other types of data adhere to the constraints defined in clause 2.4 of this part.

## 1.2 References

The following standards contain provisions which, through reference in this text, constitute provisions of this International Standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

Recommendations and reports of the CCIR, 1990
XVIIth Plenary Assembly, Dusseldorf, 1990
Volume XI - Part 1
Broadcasting Service (Television)
Rec 601-2 "Encoding parameters of digital television for studios"

IEEE Draft Standard "Specification for the Implementations of 8 by 8 Inverse Discrete Cosine Transform", P1180/D2, July 18, 1990.

Recommendations and reports of the CCIR
Volume X
Rec 953 "Encoding parameters of digital audio"

# 2 TECHNICAL NORMATIVE ELEMENTS

## 2.1 Definitions

For the purposes of this International Standard, the following definitions apply.

**AC coefficient**: Any DCT coefficient for which the frequency in one or both dimensions is non-zero.

**access unit**: in the case of compressed audio an access unit is an audio access unit. In the case of compressed video an access unit is the coded representation of a picture.

**Adaptive segmentation**: A subdivision of the digital representation of an audio signal in variable segments of time.

**adaptive bit allocation**: The assignment of bits to subbands in a time and frequency varying fashion according to a psychoacoustic model.

**adaptive noise allocation**: The assignment of coding noise to frequency bands in a time and frequency varying fashion according to a psychoacoustic model.

**Alias:** Mirrored signal component resulting from sub-Nyquist sampling.

**Analysis filterbank**: Filterbank in the encoder that transforms a broadband PCM audio signal into a set of subsampled subband samples.

**Audio Access Unit:** An Audio Access Unit is defined as the smallest part of the encoded bitstream which can be decoded by itself, where decoded means "fully reconstructed sound".

**audio buffer**: A buffer in the system target decoder for storage of compressed audio data.

**backward motion vector**: A motion vector that is used for motion compensation from a reference picture at a later time in display order.

**Bark**: Unit of critical band rate.

**bidirectionally predictive-coded picture; B-picture**: A picture that is coded using motion compensated prediction from a past and/or future reference picture.

**bitrate:** The rate at which the compressed bitstream is delivered from the storage medium to the input of a decoder.

**Block companding**: Normalizing of the digital representation of an audio signal within a certain time period.

**block:** An 8-row by 8-column orthogonal block of pels.

**Bound:** The lowest subband in which intensity stereo coding is used.

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

**channel**: A digital medium that stores or transports an ISO 11172 stream.

**chrominance (component)**: A matrix, block or sample of pels representing one of the two colour difference signals related to the primary colours in the manner defined in CCIR Rec 601. The symbols used for the colour difference signals are Cr and Cb.

**coded audio bitstream**: A coded representation of an audio signal as specified in this International Standard.

**coded video bitstream**: A coded representation of a series of one or more pictures as specified in this International Standard.

**coded order**: The order in which the pictures are stored and decoded. This order is not necessarily the same as the display order.

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

**coding parameters**: The set of user-definable parameters that characterise a coded video bitstream. Bit-streams are characterised by coding parameters. Decoders are characterised by the bitstreams that they are capable of decoding.

**component**: A matrix, block or sample of pel data from one of the three matrices (luminance and two chrominance) that make up a picture.

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

**constant bitrate coded video**: A compressed video bitstream with a constant average bitrate.

**constant bitrate:** Operation where the bitrate is constant from start to finish of the compressed bitstream.

**Constrained Parameters**: In the case of the video specification, the values of the set of coding parameters defined in Part 2 Clause 2.4.4.4.

**constrained system parameter stream (CSPS)**: An ISO 11172 multiplexed stream for which the constraints defined in Part 1 Clause 2.4.6 apply.

**CRC:** Cyclic redundancy code.

**Critical Band Rate**: Psychoacoustic measure in the spectral domain which corresponds to the frequency selectivity of the human ear.

**Critical Band:** Part of the spectral domain which corresponds to a width of one Bark.

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

**DC-coefficient**: The DCT coefficient for which the frequency is zero in both dimensions.

**DC-coded picture; D-picture**: A picture that is coded using only information from itself. Of the DCT coefficients in the coded representation, only the DC-coefficients are present.

**DCT coefficient**: The amplitude of a specific cosine basis function.

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

**decoder input buffer**: The first-in first-out (FIFO) buffer specified in the video buffering verifier.

**decoder input rate**: The data rate specified in the video buffering verifier and encoded in the coded video bitstream.

**decoder**: An embodiment of a decoding process.

**decoding process**: The process defined in this International Standard that reads an input coded bitstream and outputs decoded pictures or audio samples.

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

**Dequantization [Audio]**: Decoding of coded subband samples in order to recover the original quantized values.

**dequantization**: The process of rescaling the quantized DCT coefficients after their representation in the bitstream has been decoded and before they are presented to the inverse DCT.

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

**discrete cosine transform; DCT**: Either the forward discrete cosine transform or the inverse discrete cosine transform. The DCT is an invertible, discrete orthogonal transformation. The inverse DCT is defined in 2-Annex A of Part 2.

**display order**: The order in which the decoded pictures should be displayed. Normally this is the same order in which they were presented at the input of the encoder.

**editing**: The process by which one or more compressed bitstreams are manipulated to produce a new compressed bitstream. Conforming editted bitstreams must meet the requirements defined in this International Standard.

**elementary stream**: A generic term for one of the coded video, coded audio or other coded bit streams.

**encoder**: An embodiment of an encoding process.

**encoding process:** A process, not specified in this International Standard, that reads a stream of input pictures or audio samples and produces a valid coded bitstream as defined in this International Standard.

**Entropy coding**: Variable length noiseless coding of the digital representation of a signal to reduce redundancy.

**fast forward**: The process of displaying a sequence, or parts of a sequence, of pictures in display-order faster than real-time.

**FFT**: Fast Fourier Transformation. A fast algorithm for performing a discrete Fourier transform (an orthogonal transform).

**Filterbank [audio]:** A set of band-pass filters covering the entire audio frequency range.

**Fixed segmentation**: A subdivision of the digital representation of an audio signal in to fixed segments of time.

**forbidden**: The term 'forbidden" when used in the clauses defining the coded bitstream indicates that the value shall never be used. This is usually to avoid emulation of start codes.

**forced updating**: The process by which macroblocks are intra-coded from time-to-time to ensure that mismatch errors between the inverse DCT processes in encoders and decoders cannot build up excessively.

**forward motion vector**: A motion vector that is used for motion compensation from a reference picture at an earlier time in display order.

**Frame [audio]:** A part of the audio signal that corresponds to a fixed number of audio PCM samples.

**future reference picture**: The future reference picture is the reference picture that occurs at a later time than the current picture in display order.

**Granules [Layer II]**: 96 subband samples, 3 consecutive subband samples for all 32 subbands that are considered together before quantisation

**Granules [Layer III]**: 576 frequency lines that carry their own side information.

**group of pictures**: A series of one or more pictures intended to assist random access. The group of pictures is one of the layers in the coding syntax defined in Part 2 of this International Standard.

**Hann window:** A time function applied sample-by-sample to a block of audio samples before Fourier transformation.

**Huffman coding:** A specific method for entropy coding.

**Hybrid filterbank [audio]:** A **s**erial combination of subband filterbank and MDCT.

**IMDCT**: Inverse Modified Discrete Cosine Transform.

**Intensity stereo**: A method of exploiting stereo irrelevance or redundancy in stereophonic audio programmes based on retaining at high frequencies only the energy envelope of the right and left channels.

**interlace:** The property of conventional television pictures where alternating lines of the picture represent different instances in time.

**intra coding**: Compression coding of a block or picture that uses information only from that block or picture.

**intra-coded picture; I-picture**: A picture coded using information only from itself.

**ISO 11172 (multiplexed) stream**: A bitstream composed of zero or more elementary streams combined in the manner defined in Part 1 of this International Standard.

**Joint stereo coding:** Any method that exploits stereophonic irrelevance or stereophonic redundancy.

**Joint stereo mode:** A mode of the audio coding algorithm using joint stereo coding.

**layer [audio]**: One of the levels in the coding hierarchy of the audio system defined in this International Standard.

**layer [video and systems]**: One of the levels in the data hierarchy of the video and system specifications defined in Parts 1 and 2 of this International Standard.

**luminance (component)**: A matrix, block or sample of pels representing a monochrome representation of the signal and related to the primary colours in the manner defined in CCIR Rec 601. The symbol used for luminance is Y.

**macroblock**: The four 8 by 8 blocks of luminance data and the two corresponding 8 by 8 blocks of chrominance data coming from a 16 by 16 section of the luminance component of the picture. Macroblock is sometimes used to refer to the pel data and sometimes to the coded representation of the pel and other data elements defined in the macroblock layer of the syntax defined in Part 2 of this International Standard. The usage is clear from the context.

**Mapping [audio]**: Conversion of an audio signal from time to frequency domain by subband filtering and/or by MDCT.

**Masking threshold [audio]:** A function in frequency and time below which an audio signal cannot be perceived by the human auditory system.

**Masking**: property of the human auditory system by which an audio signal cannot be perceived in the presence of another audio signal .

**MDCT**: Modified Discrete Cosine Transform.

**motion compensation**: The use of motion vectors to improve the efficiency of the prediction of pel values. The prediction uses motion vectors to provide offsets into the past and/or future reference frames containing previously decoded pels that are used to form the prediction.

**motion vector estimation**: The process of estimating motion vectors during the encoding process.

**motion vector**: A two-dimensional vector used for motion compensation that provides an offset from the coordinate position in the current picture to the coordinates in a reference picture.

**MS stereo**: A method of exploiting stereo irrelevance or redundancy in stereophonic audio programmes based on coding the sum and difference signal instead of the left and right channels.

**non-intra coding**: Coding of a block or picture that uses information both from itself and from blocks and pictures occurring at other times.

**Non-tonal component**: A noise-like component of an audio signal.

**Nyquist sampling:** Sampling at or above twice the maximum bandwidth of a signal.

**pack**: A pack consists of a pack header followed by one or more packets. It is a layer in the system coding syntax described in Part 1 of this standard.

**packet data**: Contiguous bytes of data from an elementary stream present in a packet.

**packet header**: The data structure used to convey information about the elementary stream data contained in the packet data.

**packet**: A packet consists of a header followed by a number of contiguous bytes from an elementary data stream. It is a layer in the system coding syntax described in Part 1 of this International Standard.

**Padding**: 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.

**past reference picture**: The past reference picture is the reference picture that occurs at an earlier time than the current picture in display order.

**pel aspect ratio**: The ratio of the nominal vertical height of pel on the display to its nominal horizontal width.

**pel**: An 8-bit sample of luminance or chrominance data.

**picture period**: The reciprocal of the picture rate.

**picture rate**: The nominal rate at which pictures should be output from the decoding process.

**picture**: Source or reconstructed image data. A picture consists of three rectangular matrices of 8-bit numbers representing the luminance and two chrominance signals. The Picture layer is one of the layers in the coding syntax defined in Part 2 of this International Standard. NOTE: the term "picture" is always used in this standard in preference to the terms field or frame.

**Polyphase filterbank**: A set of equal bandwidth filters with special phase interrelationships, allowing for an efficient implementation of the filterbank.

**prediction**: The use of predictor to provide an estimate of the pel or data element currently being decoded.

**predictive-coded picture; P-picture**: A picture that is coded using motion compensated prediction from the past reference picture.

**predictor**: A linear combination of previously decoded pels or data elements.

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

**presentation unit**: A decoded audio access unit or a decoded picture.

**Psychoacoustic model**: A mathematical model of the masking behaviour of the human auditory system.

**quantization matrix**: A set of sixty-four 8-bit scaling values used by the dequantizer.

**quantized DCT coefficients**: DCT coefficients before dequantization. A variable length coded representation of quantized DCT coefficients is stored as part of the compressed video bitstream.

**quantizer scale factor**: A data element represented in the bitstream and used by the decoding process to scale the dequantization.

**random access**: The process of beginning to read and decode the coded bitstream at an arbitrary point.

**reference picture**: Reference pictures are the nearest adjacent I- or P-pictures to the current picture in display order.

**reorder buffer:** A buffer in the system target decoder for storage of a reconstructed I-picture or a reconstructed P-picture.

**reserved**: The term "reserved" when used in the clauses defining the coded bitstream indicates that the value may be used in the future for ISO defined extensions.

**reverse play**: The process of displaying the picture sequence in the reverse of display order.

**Scalefactor band**: A set of frequency lines in Layer III which are scaled by one scalefactor.

**Scalefactor index:** A numerical code for a scalefactor.

**Scalefactor**: Factor by which a set of values is scaled before quantization.

**sequence header**: A block of data in the coded bitstream containing the coded representation of a number of data elements. It is one of the layers of the coding syntax defined in Part 2 of this International Standard.

**Side information:** Information in the bitstream necessary for controlling the decoder.

**skipped macroblock**: A macroblock for which no data is stored.

**slice**: A series of macroblocks. It is one of the layers of the coding syntax defined in Part 2 of this International Standard.

**Slot [audio]:** A slot is an elementary part in the bitstream. In Layer I a slot equals four bytes, in Layers II and III one byte.

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

**Spreading function:** A function that describes the frequency spread of masking.

**start codes**: 32-bit codes embedded in that coded bitstream that are unique. They are used for several purposes including identifying some of the layers in the coding syntax.

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

**stuffing (bits); stuffing (bytes)**: Code-words that may be inserted into the compressed bitstream that are discarded in the decoding process. Their purpose is to increase the bitrate of the stream.

**Subband [audio]**: Subdivision of the audio frequency band.

**Subband filterbank:** A set of band filters covering the entire audio frequency range. In Part 3 of this International Standard the subband filterbank is a polyphase filterbank.

**Syncword:** A 12-bit code embedded in the audio bitstream that identifies the start of a frame.

**Synthesis filterbank**: Filterbank in the decoder that reconstructs a PCM audio signal from subband samples.

**system header**: The system header is a data structure defined in Part 1 of this International Standard that carries information summarising the system characteristics of the ISO 11172 multiplexed stream.

**system target decoder; STD**: A hypothetical reference model of a decoding process used to describe the semantics of an ISO 11172 multiplexed bitstream.

**time-stamp**: A term that indicates the time of an event.

**Tonal component**: A sinusoid-like component of an audio signal.

**variable bitrate:** Operation where the bitrate varies with time during the decoding of a compressed bitstream.

**variable length coding; VLC:** A reversible procedure for coding that assigns shorter code-words to frequent events and longer code-words to less frequent events.

**video buffering verifier; VBV**: A hypothetical decoder that is conceptually connected to the output of the encoder. Its purpose is to provide a constraint on the variability of the data rate that an encoder or editing process may produce.

**video sequence**: A series of one or more groups of pictures.

**zig-zag scanning order**: A specific sequential ordering of the DCT coefficients from (approximately) the lowest spatial frequency to the highest.

## 2.2 Symbols and Abbreviations

The mathematical operators used to describe this 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 zero.

### 2.2.1 Arithmetic Operators

| | |
|---|---|
| + | Addition. |
| - | Subtraction (as a binary operator) or negation (as a unary operator). |
| ++ | Increment. |
| - - | Decrement. |
| * | Multiplication. |
| ^ | Power |
| / | Integer division with truncation of the result toward zero. 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 zero 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 toward -∞. |
| % | Modulus operator. Defined only for positive numbers. |
| Sign( ) | $\mathrm{Sign}(x) \begin{array}{rl} = 1 & x > 0 \\ 0 & x == 0 \\ -1 & x < 0 \end{array}$ |
| NINT ( ) | Nearest integer operator. Returns the nearest integer value to the real-valued argument. Half-integer values are rounded away from zero. |
| sin | Sine |
| cos | Cosine |
| exp | Exponential |
| √ | 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 zero 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 the 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.

uimsbf Unsigned integer, most significant bit first.

vlclbf Variable length code, left bit first, where "left" refers to the order in which the VLC codes are written in Annex B.

ch channel.

gr granule of three sub-band samples in audio layer 2, twelve sub-band samples in audio layer 3.

rpchof remainder polynomial coefficients, highest order first.

sb sub-band.

scfsi scale-factor selector information.

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

### 2.2.7 Constants

| | |
|---|---|
| pi | 3.14159265359... |
| e | 2.71828182846... |

## 2.3 Method of Describing Bit Stream Syntax

The bit stream retrieved by the decoder is described in clause 2.4.3. 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 clause 2.4.4. The following constructs are used to express the conditions when data elements are present, and are in normal type:

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

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

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

for (  i = 0; i < n; i++)  {    The group of data elements occurs n times.  Conditional constructs
   data_element  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 one 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 [n]**data_element [n] is the n+1th element of an 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 clause 2.4.2 implements a satisfactory decoding procedure. In particular, it defines a correct and error-free input bitstream. Actual decoders must include a means to look for start codes 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.

### Definition of bytealigned function

The function bytealigned () returns 1 if the current position is on a byte boundary, that is the next bit in the bit stream is the first bit in a byte.

### Definition of nextbits function

The function nextbits () permits comparison of a bit string with the next bits to be decoded in the bit stream.

### Definition of next_start_code function

The next_start_code function removes any zero bit and zero byte stuffing and locates the next start code.

```
next_start_code() {
  while ( !bytealigned() )
    zero_bit                                          1           "0"
  while ( nextbits() != '0000 0000 0000 0000 0000 0001' )
    zero_byte                                         8           "00000000"
  }
```

## 2.4 Requirements

### 2.4.1 Coding Structure and Parameters

The system coding layer allows one or more elementary streams to be combined into a single stream. Data from each elementary stream are multiplexed and encoded together with information that allows elementary streams to be replayed in synchronism.

#### ISO 11172 stream

An ISO 11172 stream consists of one or more elementary streams multiplexed together. Each elementary stream consists of access units, which are the coded representation of presentation units. The presentation unit for a video elementary stream is a picture. The corresponding access unit includes all the coded data for the picture as well as any group of picture or sequence headers that immediately precede it (see sub-clause 2.4.1 in Part 2 of this standard for definitions of group of picture and sequence headers). The presentation unit for an audio elementary stream is an audio frame (see sub-clause 2.4.3.1 in Part 3 of this standard for a definition of an audio frame).

Data from elementary streams is stored in packets. A packet consists of a packet header followed by packet data. The packet header begins with a 32-bit start-code and also identifies the stream to which the packet data belongs. The packet header may contain decoding and/or presentation time-stamps (DTS and PTS) that refer to the first access unit in the packet. The packet data contains a variable number of contiguous bytes from one elementary stream.

Packets are organised in packs. A pack commences with a pack header and is followed by zero or more packets. The pack header begins with a 32-bit start-code. The pack header is used to store timing and bitrate information.

The stream begins with a system header that may optionally be repeated. The system header carries a summary of the system parameters defined in the stream.

### 2.4.2 System Target Decoder

The semantics of the multiplexed stream specified in sub-clause 2.4.4 and the constraints on these semantics specified in sub-clause 2.4.5 require exact definitions of decoding events and the times at which these events occur. The definitions needed are set out in this standard using a hypothetical decoder known as the system target decoder (STD).

The STD is a conceptual model used to define these terms precisely and to model the decoding process during the construction of ISO 11172 streams. The STD is defined only for this purpose. Neither the architecture of the STD nor the timing described precludes uninterrupted, synchronized play-back of ISO 11172 multiplexed streams from a variety of decoders with different architectures or timing schedules.

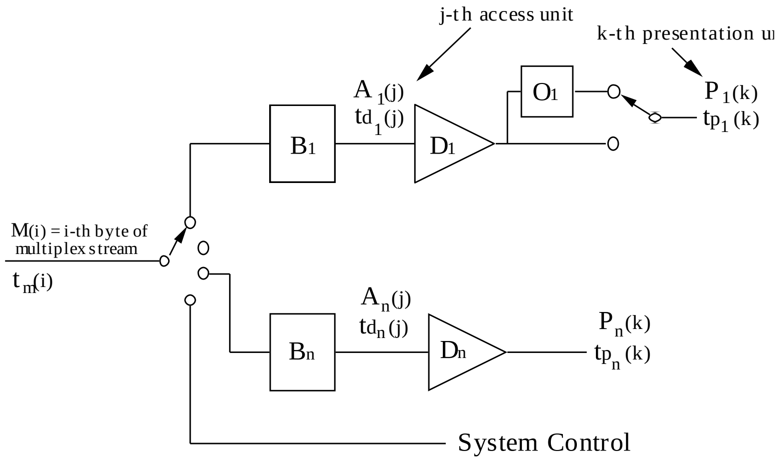


**Figure I.2 Diagram of System Target Decoder**

### Notation

The following notation is used to describe the system target decoder and is illustrated in the figure above.

i, i' are indices to bytes in the ISO 11172 stream

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.

M(i) is the $i^{th}$ byte in the ISO 11172 multiplexed stream.

tm(i) indicates the time in seconds at which the $i^{th}$ byte of the ISO 11172 multiplexed stream enters the system target decoder.

SCR(i) is the time encoded in the SCR field measured in units of the 90kHz system clock where i is the byte index of the final byte of the SCR field.

$A_n(j)$ is the $j^{th}$ access unit in elementary stream n. Note that access units are 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.

$tp_n(k)$ 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 elementary stream n.

### System Clock Frequency

Timing information is carried by several data fields defined in this standard in sub-clauses 2.4.3 and 2.4.4. 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:

$$90\ 000 - 4{,}5 <= \text{system\_clock\_frequency} <= 90\ 000 + 4{,}5$$

$$\text{rate of change of system\_clock\_frequency with time} <= 250 * 10^{-6}\ \text{Hz/s}$$

The notation "system_clock_frequency" is used in several places in this standard to refer to the frequency of a clock meeting these requirements.

### Input to the System Target Decoder

Data from the ISO 11172 multiplexed stream enters the system target decoder. The $i^{th}$ byte, M(i), enters at time tm(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 encoded in the pack header. The value encoded in the SCR(i') field indicates time tm(i'), where i' refers to the last byte of the SCR field, M(i').

Specifically:

$$SCR(i') = NINT\ (\ \text{system\_clock\_frequency} * tm(i')\ )\ \%\ 2^{33}$$

The input arrival time, tm(i), for all other bytes shall be constructed from SCR(i') and the rate at which data arrive, where the arrival rate within each pack is the value represented in the mux_rate field in that pack's header (see clauses 2.4.3.2 and 2.4.4.2) .

$$tm(i) = SCR(i') + \frac{i}{(\text{mux\_rate} * 50)}$$

Where:

| | |
|---|---|
| i' | is the index in the pack of the final byte of the system_clock_reference field in the pack header. |
| i | is the relative index of any byte in the pack; i =1 for the first byte after the system_clock_reference field. |
| SCR(i') | is the time encoded in the system_clock_reference field in units of the system clock. |
| mux_rate | is a field defined in clauses 2.4.3.2 and 2.4.4.2. |

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 system target decoder.

### Buffering

The packet data from elementary stream n is copied to the input buffer for stream n, $B_n$. Transfer of byte M(i) from the system target decoder input to $B_n$ is instantaneous, so that byte M(i) enters the buffer for stream n, of size $BS_n$, at time tm(i).

Bytes present in the pack, system or packet headers of ISO 11172 stream but not part of the packet data (for example the SCR, DTS, PTS, packet_length fields, etc.- see sub-clause 2.4.3) 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 parameters in the syntax (see sub-clauses 2.4.3 and 2.4.4).

At the decoding time, $td_n(j)$, all the data for the access unit that has been in the input buffer longest ($A_n(j)$ ) is removed instantaneously. In the case of a video elementary stream, group of picture and sequence header data that precedes the picture is removed at the same time. As the access unit is removed from the buffer it is instantaneously decoded into a presentation unit.

### Decoding

Elementary streams buffered in $B_1$ through $B_n$ are decoded instantaneously by decoders $D_1$ through $D_n$ and may be delayed in reorder buffers $O_1$ through $O_n$ before being presented to the viewer at the output of the system target decoder. Reorder buffers are used only in video decoding to store I-pictures and P-pictures while the sequence of presentation units is reordered before presentation.

In the case of a video elementary stream, some access units may not be stored in presentation order. These access units will need to be reordered before presentation. In particular, an I-picture or a P-picture stored before one or more B-pictures must be delayed in the reorder buffer, $O_n$, of the system target decoder before being presented. It should be delayed until the next I-picture or P-picture is decoded. While it is stored in the reorder buffer, the subsequent B-pictures are decoded and presented.

If $P_n(k)$ is an I-picture or a P-picture that needs to be reordered before presentation, it is stored in $O_n$ after being decoded and the picture previously stored in $O_n$ is presented. Subsequent B-pictures are decoded and presented without reordering.

The time at which a presentation unit $P_n(k)$ is presented to the viewer is $tp_n(k)$. For presentation units that are not reordered $tp_n(k)$ is equal to $td_n(j)$ since the access units are decoded instantaneously. For presentation units that are reordered $tp_n(k)$ and $td_n(j)$ differ by the time that $P_n(k)$ is delayed in the reorder buffer, which is nominally an integral number of picture periods.

Part 2 clause 2.4.1 of this standard explains reodering of video pictures in greater detail.

### 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 system target decoder the display of a video presentation unit (a picture) occurs instantaneously at its presentation time, $tp_n(k)$.

In the system target decoder 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.

## 2.4.3 Specification of the System Stream Syntax

The following syntax describes a stream of bytes.

### 2.4.3.1 ISO 11172 Layer

```
iso11172_stream() {
        do {
                pack()
        } while (nextbits() == pack_start_code)
        iso_11172_end_code                                          32 bslbf
}
```

### 2.4.3.2 Pack Layer

#### Pack

```
pack() {
        pack_start_code                                     32 bslbf
        '0010'                                                      4 bslbf
        system_clock_reference [32..30]                     3 bslbf
        marker_bit                                                  1 bslbf
        system_clock_reference [29..15]                     15 bslbf
        marker_bit                                                  1 bslbf
        system_clock_reference [14..0]                      15 bslbf
        marker_bit                                                  1 bslbf
        marker_bit                                                  1 bslbf
        mux_rate                                                    22 uimsbf
        marker_bit                                                  1 bslbf

        if (nextbits() == system_header_start_code)
                system_header ()

        while (nextbits() == packet_start_code)
                packet()
}
```

### System header

```
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
        reserved_byte                                               8 bslbf
        while (nextbits () == '1') {
                stream_id                                           8 uimsbf
                '11'                                                2 bslbf
                STD_buffer_bound_scale                              1 bslbf
                STD_buffer_size_bound                       13 uimsbf
        }
}
```

### 2.4.3.3 Packet Layer

```
packet() {
        packet_start_code_prefix                                      24 bslbf
        stream_id                                                      8 uimsbf
        packet_length                                          16 uimsbf

        if (packet_start_code != private_stream_2) {
                while (nextbits() == '1')
                        stuffing_byte                                  8 bslbf

                if (nextbits () == '01') {
                        '01'                                           2 bslbf
                        STD_buffer_scale                               1 bslbf
                        STD_buffer_size                        13 uimsbf
                }

                if (nextbits() == '0010') {
                        '0010'                                         4 bslbf
                        presentation_time_stamp[32..30]        3 bslbf
                        marker_bit                                     1 bslbf
                        presentation_time_stamp[29..15]        15 bslbf
                        marker_bit                                     1 bslbf
                        presentation_time_stamp[14..0]         15 bslbf
                        marker_bit                                     1 bslbf
                }

                else if (nextbits() == '0011') {
                        '0011'                                         4 bslbf
                        presentation_time_stamp[32..30]        3 bslbf
                        marker_bit                                     1 bslbf
                        presentation_time_stamp[29..15]        15 bslbf
                        marker_bit                                     1 bslbf
                        presentation_time_stamp[14..0]         15 bslbf
                        marker_bit                                     1 bslbf
                        '0001'                                         4 bslbf
                        decoding_time_stamp[32..30]            3 bslbf
                        marker_bit                                     1 bslbf
                        decoding_time_stamp[29..15]            15 bslbf
                        marker_bit                                     1 bslbf
                        decoding_time_stamp[14..0]                     15 bslbf
                        marker_bit                                     1 bslbf
                }
                else
                        '0000 1111'                                    8 bslbf

        }
        for (i = 0; i < N; i++) {
                packet_data_byte                                       8 bslbf
        }
}
```

## 2.4.4 Semantic Definition of Fields in Syntax

### 2.4.4.1 ISO 11172 Layer

**iso_11172_end_code** -- The iso_11172_end_code is the bit string "0000 0000 0000 0000 0000 0001 1011 1001" (000001B9 in hexadecimal). It terminates the ISO 11172 multiplexed stream.

### 2.4.4.2 Pack Layer

#### Pack

**pack_start_code** -- The pack_start_code is the bit string "0000 0000 0000 0000 0000 0001 1011 1010" (000001BA in hexadecimal). It identifies the beginning of a pack.

**system_clock_reference** -- The system_clock_reference (SCR) is a 33-bit number coded in three separate fields. It indicates the intended time of arrival of the last byte of the system_clock_reference field at the input of the system target decoder. The value of the SCR is measured in the number of periods of a 90kHz system clock with a tolerance specified in clause 2.4.2. Using the notation of clause 2.4.2 the value encoded in the system_clock_reference is:

$$\text{NINT (system\_clock\_frequency * (tm(i)) ) \% } 2^{33}$$

where M(i) is the last byte of the coded system_clock_reference field.

**marker_bit** -- A marker_bit is a one bit field that has the value "1".

**mux_rate --** This is a positive integer specifying the rate at which the system target decoder receives the ISO 11172 multiplexed stream during the pack in which it is included. The value of mux_rate is measured in units of 50 bytes/second rounded upwards. The value zero is forbidden. The value represented in mux_rate is used to define the time of arrival of bytes at the input to the system target decoder in clause 2.4.2 of Part 1 of this standard. The value encoded in the mux_rate field may vary from pack to pack in an ISO 11172 multiplexed stream.

#### System Header

**system_header_start_code --** The system_header_start_code is the bit string "0000 0000 0000 0000 0000 0001 1011 1011" (000001BB in hexadecimal). It identifies the beginning of a system header.

**header_length --** The header_length specifies the number of bytes remaining in the header after the header_length field. The header_length shall be equal to the number of bytes in the system header following the header_length field. Note that future extensions of this standard may extend the system header.

**rate_bound --** The rate_bound is an integer value greater than or equal to the maximum value of the mux_rate field coded in any pack of the ISO 11172 multiplexed stream. It may be used by a decoder to assess whether it is capable of decoding the entire stream.

**audio_bound --** The audio_bound is an integer in the inclusive range from 0 to 32 greater than or equal to the maximum number of audio streams simultaneously multiplexed in the ISO 11172 multiplexed stream.

**fixed_flag --** The fixed_flag is a one-bit flag. If its value is set to "1" fixed bitrate operation is indicated. If its value is set to "0" variable bitrate operation is indicated. During fixed bitrate operation, the value encoded in all system_clock_reference fields in the multiplexed ISO 11172 stream shall adhere to the following linear equation:

$$\text{SCR(i) = NINT (c1 * i + c2) \% } 2^{33}$$

where c1 is a real-valued constant valid for all i

c2 is a real-valued constant valid for all i
i is the index in the multiplexed ISO 11172 stream of the final byte of any system_clock_reference field in the stream.

**CSPS_flag --** The CSPS_flag is a one-bit flag. If its value is set to "1" the ISO 11172 multiplexed stream meets the constraints defined in Part 1 clause 2.4.6 of this standard.

**system_audio_lock_flag --** The system_audio_lock_flag is a one-bit flag indicating that there is a specified, constant harmonic relationship between the audio sampling rate and the system clock frequency in the system target decoder. Clause 2.4.2 defines system_clock_frequency and the audio sampling rate is specified in Part 3 of this standard. The system_audio_lock_flag may only be set to "1" if, for all presentation units in all audio elementary streams in the ISO 11172 multiplexed stream, the ratio of system_clock_frequency and the actual audio sampling rate is constant and equal to the value indicated in the following table at the nominal sampling rate of the elementary stream, where "/" indicates exact real division.

| Nominal audio sampling frequency (kHz) | 32 | 44,1 | 48 |
|---|---|---|---|
| system_clock_frequency / audio sampling rate | 90/32 | 900 / 441 | 90 / 48 |

**system_video_lock_flag --** The system_video_lock_flag is a one-bit flag indicating that there is a specified, constant harmonic relationship between the video picture rate and the system clock frequency in the system target decoder. Clause 2.4.2 defines system_clock_frequency and the video picture rate is specified in Part 2 of this standard. The system_video_lock_flag may only be set to "1" if, for all presentation units in all video elementary streams in the ISO 11172 multiplexed stream, the ratio of system_clock_frequency to the actual video picture rate, SCPR, is constant and equal to the value indicated in the following table at the nominal picture rate of the elementary stream.

$$\text{SCPR} = \frac{\text{system\_clock\_frequency}}{\text{picture rate in the STD}}$$

| Nominal picture rate (Hz) | 23,976 | 24 | 25 | 29,97 | 30 | 50 | 59,94 | 60 |
|---|---|---|---|---|---|---|---|---|
| Ratio SCPR | 3 753,75 | 3 750 | 3 600 | 3 003 | 3 000 | 1 800 | 1 501,5 | 1 500 |

**video_bound --** The video_bound is an integer in the inclusive range from 0 to 16 greater than or equal to the maximum number of video streams simultaneously multiplexed in the ISO 11172 multiplexed stream.

**reserved_byte --** This byte is reserved for future use by ISO. Until otherwise specified by ISO it shall have the value "1111 1111".

**stream_id --** The stream_id indicates the type and number of the stream in the multiplex to which the following STD_buffer_scale and STD_buffer_size fields refer.

If stream_id equals "1011 1000" the STD_buffer_scale and STD_buffer_size fields following the stream_id refer to all audio streams in the ISO 11172 multiplexed stream.

If stream_id equals "1011 1001" the STD_buffer_scale and STD_buffer_size fields following the stream_id refer to all video streams in the ISO 11172 multiplexed stream.

If the stream_id takes on any other value it shall be a byte value greater than or equal to "1011 1100" and shall be interpreted as referring to the stream type and number according to the following table. This table is also used to identify the stream type and number indicated by the stream_id field that forms a part of the packet_start_code defined in clause 2.4.4.3.

**stream_id table**

| stream_id | stream type |
|---|---|
| 1011 1100 | reserved stream |
| 1011 1101 | private_stream_1 |
| 1011 1110 | padding stream |
| 1011 1111 | private_stream_2 |
| 110x xxxx | audio stream - number xxxxx |
| 1110 xxxx | video stream - number xxxx |
| 1111 xxxx | reserved data stream - number xxxx |
| The notation x means that the values 0 and 1 are both permitted and result in the same stream type. The stream number is given by the values taken by the x's. | |

Each elementary stream present in the ISO 11172 multiplexed stream shall have its STD_buffer_scale and STD_buffer_size specified exactly once by this mechanism in each system header.

**STD_buffer_bound_scale --** The STD_buffer_bound_scale is a one-bit field that indicates the scaling factor used to interpret the subsequent STD_buffer_size_bound field. If the preceding stream_id indicates an audio stream, STD_buffer_bound_scale shall have the value "0". If the preceding stream_id indicates a video stream, STD_buffer_bound_scale shall have the value "1". For all other stream types, the value of the STD_buffer_bound_scale may be either "1" or "0".

**STD_buffer_size_bound --** The STD_buffer_size_bound is a 13-bit unsigned integer defining a value greater than or equal to the size of the input buffer, $BS_n$, in the system target decoder. If STD_buffer_bound_scale has the value "0" then STD_buffer_size_bound measures the buffer size bound in units of 128 bytes. If STD_buffer_bound_scale has the value "1" then STD_buffer_size_bound measures the buffer size bound in units of 1024 bytes. Thus:

```
if (STD_buffer_bound_scale == 0)
        BSn <= STD_buffer_size_bound * 128;
else
        BSn <= STD_buffer_size_bound * 1024;
```

### 2.4.4.3 Packet Layer

**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" (000001 in hexadecimal).

**stream_id --** The stream_id specifies the type and number of the elementary stream as defined by the stream_id table in clause 2.4.4.2.

**packet_length** -- The packet_length specifies the number of bytes remaining in the packet after the packet_length field.

**stuffing_byte** -- This is a fixed 8-bit value equal to "1111 1111" that can be inserted to increase the bitrate to that required by the digital storage medium. It is discarded by the decoder. No more than sixteen stuffing bytes shall be present in one packet header.

**STD_buffer_scale** -- The STD_buffer_scale is a one-bit field that indicates the scaling factor used to interpret the subsequent STD_buffer_size field. If the preceding stream_id that forms part of the packet_start_code indicates an audio stream, STD_buffer_scale shall have the value "0". If the preceding stream_id that forms part of the packet_start_code indicates a video stream, STD_buffer_scale shall have the value "1". For all other stream types, the value may be either "1" or "0".

**STD_buffer_size** -- The STD_buffer_size is a 13-bit unsigned integer defining the size of the input buffer, BSn, in the system target decoder. If STD_buffer_scale has the value "0" then the STD_buffer_size measures the buffer size in units of 128 bytes. If STD_buffer_scale has the value "1" then the STD_buffer_size measures the buffer size in units of 1024 bytes. Thus:

```
if (STD_buffer_scale == 0)
        $BS_n$ = STD_buffer_size * 128;
else
        $BS_n$ = STD_buffer_size * 1024;
```

**presentation_time_stamp --** The presentation_time_stamp (PTS) is a 33-bit number coded in three separate fields. It indicates the intended time of presentation in the system target decoder of the presentation unit that corresponds to the first access unit that commences in the packet. The value of PTS is measured in the number of periods of a 90kHz system clock with a tolerance specified in clause 2.4.2. Using the notation of clause 2.4.2 the value encoded in the presentation_time_stamp is:

$$\text{NINT (system\_clock\_frequency} * (tp_n(k)\,)\,)\ \%\ 2^{33}$$

where

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

$P_n(k)$ is the presentation unit corresponding to the first access unit that commences in the packet data. An access unit commences in the packet if the first byte of a video picture start code or the first byte of the synchronization word of an audio frame (see Parts 2 and 3 of this standard) is present in the packet data.

**decoding_time_stamp --** The decoding_time_stamp (DTS) is a 33-bit number coded in three separate fields. It indicates the intended time of decoding in the system target decoder of the first access unit that commences in the packet. The value of DTS is measured in the number of periods of a 90kHz system clock with a tolerance specified in clause 2.4.2. Using the notation of clause 2.4.2 the value encoded in the decoding_time_stamp is:

$$\text{NINT (system\_clock\_frequency} * (td_n(j)\,)\,)\ \%\ 2^{33}$$

where

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

$A_n(j)$ is the first access unit that commences in the packet data. An access unit commences in the packet if the first byte of a video picture start code or the first byte of the synchronization word of an audio frame (see Parts 2 and 3 of this standard) is present in the packet data.

**packet_data_byte --** packet_data_bytes shall be contiguous bytes of data from the elementary stream indicated by the stream_id of the packet_start_code. The byte-order of the elementary stream shall be preserved. The number of packet_data_bytes, N, may be calculated from the packet_length field. N is equal to the value indicated in the packet_length minus the number of bytes between the last byte of the packet_length field and the first packet_data_byte.

In the case of a video stream, packet_data_bytes are coded video data as defined in Part 2 of this standard. In the case of an audio stream, packet_data_bytes are coded audio data as defined in Part 3 of this standard. In the case of a padding stream, packet_data_bytes consist of padding bytes. Each padding byte is a fixed bit-string with the value "1111 1111". In the case of a private stream (type 1 or type 2), packet_data_bytes are user definable and will not be defined by ISO in the future. The contents of packet_data_bytes in reserved streams may be specified in the future by ISO.

## 2.4.5 Restrictions on the Multiplexed Stream Semantics

### 2.4.5.1 Buffer Management

The ISO 11172 multiplexed stream, M(i) in the notation described in clause 2.4.2, shall be constructed and tm(i) shall be chosen so that the input buffers of size $BS_1$ through BSn neither overflow nor underflow in the system target decoder. That is:

$0 <= F_n (t) <= BS_n$ for all t and n

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

$F_n(t)$ is the instantaneous fullness of STD buffer $B_n$.

For all ISO 11172 multiplexed streams the delay caused by system target decoder input buffering shall be less than or equal to one second. The input buffering delay is the difference in time between a byte entering the input buffer and when it is decoded.

Specifically:

$td_n(j) - tm(i) <= 1$

For all bytes M(i) contained in access unit j.

### 2.4.5.2 Frequency of Coding the system_clock_reference

The ISO 11172 multiplexed stream, M(i), shall be constructed so that the time interval between the final bytes of system_clock_reference fields in successive packs shall be less than or equal to 0,7 seconds. Thus:

$|t_m(i) - t_m(i')| <= 0,7$ seconds

for all i and i' where M(i) and M(i') are the last bytes of consecutive system_clock_reference fields.

### 2.4.5.3 Frequency of presentation_time_stamp coding

The ISO 11172 multiplexed stream M(i) shall be constructed so that the maximum difference between coded presentation_time_stamps is 0,7 seconds. Thus:

$|tp_n(k) - tp_n(k'')| <= 0,7$ seconds

for all k, k'' and n, where:

$P_n(k)$ and $P_n(k'')$ are presentation units for which presentation_time_stamps are coded in the packet header

and

k and k'' are chosen so that there is no presentation unit, $P_n(k')$, with a presentation_time_stamp coded in its packet header, with k < k' < k''.

### 2.4.5.4 Conditional Coding of Time Stamps

The presentation_time_stamp shall be encoded in the packet containing the first access unit of each elementary stream in an ISO 11172 multiplexed stream.

If a discontinuity exists in any elementary audio or video stream in the ISO11172 multiplex then a presentation_time_stamp shall be encoded referring to the first access unit after each discontinuity.

A discontinuity exists at the start of presentation unit $P_n(k)$ in an elementary stream n if the presentation time $tp_n(k)$ is greater than the largest value permissible given the specified tolerance on the system_clock_frequency.

Thereafter, presentation_time_stamps may be present in any packet header with the following exception. If no access unit commences in the packet data, the presentation_time_stamp shall not be present in the packet header. If a presentation_time_stamp is present in a packet header it shall refer to the presentation unit corresponding to the first access unit that commences in the packet data.

A decoding_time_stamp shall appear in a packet header if and only if the following two conditions are met:

a) A presentation_time_stamp is present in the packet header
b) The decoding time differs from the presentation time.

For the purpose of this sub-clause a video access unit commences in a packet if the first byte of the picture_start_code is present in the packet data (see Part 2 of this standard). An audio access unit commences in a packet if the first byte of the synchronization word of the audio frame is present in the packet data (see Part 3 of this standard).

#### 2.4.5.5 Frequency of Coding STD_buffer_size in Packet Headers

The STD_buffer_scale and STD_buffer_size fields shall occur in the first packet of each elementary stream and again whenever the value changes. They may also occur in any other packet.

#### 2.4.5.6 Coding of System Header

The first pack of an ISO 11172 multiplexed stream shall commence with a system header. The system header may also be present in any other pack. The values encoded in all the system headers in the ISO 11172 multiplexed stream shall be identical.

## 2.4.6 Constrained System Parameter Stream

An ISO 11172 multiplexed stream is a "constrained system parameters stream" (CSPS) if it conforms to the bounds specified in this section. ISO 11172 multiplexed streams are not limited to the bounds specified by the CSPS. A CSPS may be identified by means of the CSPS_flag defined in the stream header (clause 2.4.3.2). The CSPS is a subset of all possible ISO 11172 multiplexed streams.

### Packet Rate

In the CSPS, the maximum rate at which packets shall arrive at the input to the system target decoder is 300 packets per second if the value encoded in the mux_rate field is less than or equal to 5 000 000 bits/second. For higher bit-rates the CSPS packet rate is bounded by a linear relation to the value encoded in the mux_rate field.

Specifically, for all packs p in the ISO 11172 multiplexed stream,

$$NP <= (tm(i') - tm(i)) * 300 * \max [1, \frac{R_{max}}{5*10^6}]$$

where

$$R_{max} = 8 * \max ( \frac{i' - i}{tm(i') - tm(i)} ) \text{ bits / second}$$

NP is the number of packet_start_codes and system_header_start_codes between adjacent pack_start_codes or between the last pack_start_code and the iso_11172_end_code.

tm(i) is the time, measured in seconds, encoded in the system_clock_reference of pack p.

tm(i') is the time, measured in seconds, encoded in the system_clock_reference for pack p+1, immediately following pack p, or in the case of the final pack in the ISO 11172 stream, the time of arrival of the last byte of the iso_11172_end_code.

### System Target Decoder Buffer Size

In the case of a CSPS the maximum size of each input buffer in the system target decoder is bounded. Different bounds apply for video elementary streams and audio elementary streams.

In the case of a video elementary stream in a CSPS the following applies:

In clause 2.4.3.2 of Part 2 of this standard the horizontal picture size, horizontal_size, and the vertical picture size, vertical_size, are defined. If the values encoded in horizontal_size and vertical_size meet the constraints on the picture size specified for the constrained_parameters_flag in clause 2.4.3.2 of Part 2, then

$$BS_n <= 46 * 1024 \text{ bytes.}$$

For all other video elementary streams in a CSPS,

$$BS_n <= \max [46 * 1024, R_{vmax} * 46 * 1024 \ // \ (1,856*10^6)] \text{ bytes}$$

where $R_{vmax}$ is the greatest value of video bit_rate specified or used in the elementary video stream; reference Part 2 clause 2.4.3.2.

In the case of an audio elementary stream in a CSPS the following applies:

$BS_n$ <= 4096 bytes.

# 1-ANNEX A (informative)

# DESCRIPTION OF THE SYSTEM CODING LAYER

## 1-A.1 Overview

ISO 11172 Part 1 specifies the syntax and semantics of the system layer coding of combined coded audio, video, and private data. The coding specification provides data fields and semantic constraints on the data stream to support the necessary system functions. These include the synchronized presentation of decoded information, the construction of a multiplexed stream, the management of buffers for coded data, start-up and random access, and absolute time identification.

While the specification applies to the coded bitstream, there are implications on the functions that encoders and decoders must perform and the degrees of freedom given to them.

The encoding system performs coding of audio and video data, coding of system layer information, and multiplexing. Coding the system layer information includes creating time-stamps: presentation time-stamps (PTS) and decoding time-stamp (DTS) fields are used for synchronization of audio and video. System clock reference (SCR) fields are used in conjunction with PTS and DTS for synchronization and buffer management. The use of a common time base, the system time-clock (STC), to unify the measurement of the timing of coded data (SCR) and the timing of the presentation of data (the PTS and DTS fields), ensures correct synchronization and buffer management.

The decoding system performs parsing and demultiplexing of the ISO 11172 data stream, the system functions listed above, and the decoding and presentation of elementary streams. The specification of the system is written in terms of an idealized reference decoder known as the system target decoder (STD). The purpose of the STD is to provide a clear, simple model of the decoding system so that the various terms can be specified unambiguously. In the case of video data the term "presentation unit" (PU) refers to decoded pictures, and in the case of audio data it refers to decoded audio frames. The term "access unit" (AU) refers to the coded representation of presentation units.

This Annex explains how the system functions are provided by encoders and decoders and the degrees of freedom that are available. Clause 1-A.2 outlines the operation of an encoding system, Clause 1-A.3 describes the characteristics of the data stream and Clause 1-A.4 outlines the functions of a decoder. Clause 1-A.5 describes a set of parameters suitable for use on CD-ROM devices, and Clause 1-A.6 contains a worked example of a system bitstream.

## 1-A.2 Encoder Operations

### 1-A.2.1 Degrees of freedom

Flexibility in the system layer syntax allows a wide latitude in creating the multiplexed bitstream. Multiple elementary streams of audio, video, private and padding data can be combined in a practical way into a single stream. Two private data streams of different types are provided. One type is completely private and the other includes the syntax defined in this Standard to support synchronization and buffer management. If more than two private data streams are needed, an unlimited number of sub-streams may be defined. Up to 32 audio and 16 video streams may be multiplexed simultaneously. Packs and packets (the elementary units of the multiplexed stream defined in Clauses 2.4.3.2 and 2.4.3.3 of Part 1 of this Standard) may vary in length in order to allocate different bitrates to different streams, or for other reasons. Elementary streams and multiplexed bitstreams may vary their rates from time to time, or operate at a fixed bitrate. The amount of buffering needed in the STD model decoder may be specified individually for each elementary stream. To facilitate random access, the encoding system may include frequent occurrences of the information needed to start decoding, such as SCR, PTS and system headers. Applications will be further assisted if this information is located in the data stream close to the information needed in elementary streams to start decoding (for example video sequence headers and Intra-Pictures - see Part 2 of this Standard).

The encoder has the option of following a set of specific constraints and setting the Constrained System Parameter Stream (CSPS) flag, the system_audio_lock flag, or the system_video_lock flag. These optional flags may be set independently of one another. The Constrained System Parameters are a defined sub-set of all possible system layer parameters and encoding options. Their purpose is to define a restricted set of parameters and options that can be decoded by economical decoders while being broad enough in application to gain widespread use. The system_audio_lock_flag indicates that all the audio streams have an exact relationship to the system clock frequency. The system_video_lock_flag indicates that all the video streams have an exact relationship to the system clock frequency.

### 1-A.2.2 Synchronization

ISO 11172-1 provides for end-to-end synchronization of the complete encoding and decoding process. This function is provided through the use of time-stamps, particularly Presentation Time-stamps (PTS). This end-to-end synchronization is illustrated in Figure 1-A.1 which includes a prototypical encoder and a prototypical decoder. While these prototypical encoding and decoding systems are not normative, they illustrate the functions expected of real systems.

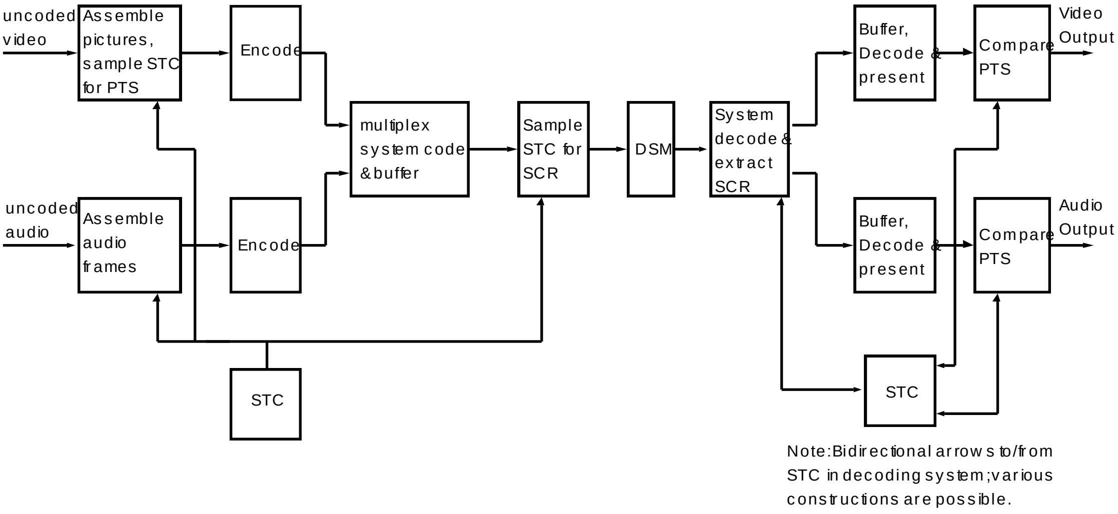

uncoded video
Assemble pictures, sample STC for PTS
Encode
uncoded audio
Assemble audio frames
Encode
multiplex system code & buffer
Sample STC for SCR
DSM
System decode & extract SCR
Buffer, Decode & present
Compare PTS
Video Output
Buffer, Decode & present
Compare PTS
Audio Output
STC
STC
Note:Bidirectional arrows to/from STC in decoding system;various constructions are possible.

**Figure 1-A.1**

In the prototypical encoding system, there is a single system time-clock (STC) which is available to the audio and video encoders. Audio samples entering the audio encoder are organized into audio presentation units (PU). Some, but not necessarily all, of the audio PUs have PTS values associated with them, which are samples of the STC at the time the first sample of the PU is input to the encoder. Likewise, video pictures enter the video encoder, and the STC values at the times that this occurs are used to create video PTS fields. SCR values represent the time when the last byte of the SCR field leaves the encoder.

The Standard specifies the encoder and decoder functions in terms of a reference decoder model known as the System Target Decoder (STD). In this model video pictures and audio presentation units are presented to the user instantaneously. Actual *decoders* will generally introduce post-processing and presentation delays. These decoding delays should not be compensated by real *encoders*. Real encoders must generate bitstreams that play correctly on the idealised STD. Doing this may involve, for instance, choosing the value of the PTS at the time corresponding to the middle of a raster-scanned picture. Such an offset is acceptable providing that it is constant, does not introduce jitter into the sequence of PTS values, and the constraints on the bitstream buffering are respected. The delays that occur in any specific, real decoder must be compensated in that decoder, not the encoder.

SCR and PTS fields, and DTS where required, must be inserted by the encoder at intervals not exceeding 0,7 seconds as measured by the values contained in the fields. The time interval refers to coded data time for SCR fields, and presentation time for PTS and DTS fields. These fields need not be periodic, and they may be encoded more frequently than the minimum time specified.

Because clock frequencies generally deviate from their nominal values, the use of independent clocks for the generation of PTS, DTS and SCR fields results in synchronization or buffer management problems. Therefore all the PTS, DTS and SCR fields in the multiplexed stream must be samples of the same STC or have values that are equivalent to those which would have been obtained from a single clock. It is not permissible, for example, to use independent clocks to produce the PTS and SCR fields in the various streams. This requirement is specified via the definition of the "system_clock_frequency" in the definitions of PTS, DTS, and SCR.

While there is a specification on the frequency tolerance of the "system_clock_frequency" function used to create the time-stamps, there are no explicit specifications in Parts 2 and 3 of this standard on the accuracy of the picture rate, audio sample rate, or bitrate, nor on the jitter of these parameters. This issue will be addressed in the compliance testing specification.

In practice the picture rate and audio sample rate will not exactly match the nominal rate unless they are specifically locked to the STC. There is no requirement that these rates should be locked to the system_clock_frequency. However, if either or both rates are locked and have an exact relationship to the system_clock_frequency the encoder may record this by setting the system_audio_lock_flag and/or the system_video_lock_flag. In the case where the audio sample rates and video picture rates are not locked to the STC, the values implied by the PTS fields should closely match the nominal rates indicated in the elementary data streams in order to avoid problems in decoders.

### 1-A.2.3 Multiplexing

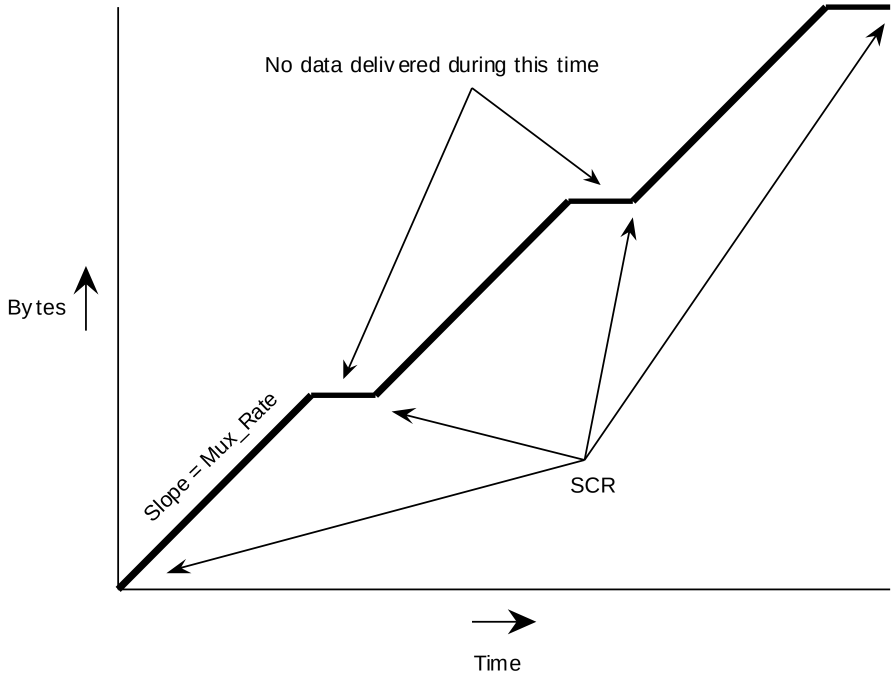
Data from elementary streams are kept distinct by the use of packets, each having a packet start code that identifies the stream. A data packet never contains data from more than one elementary stream and byte ordering is preserved. Thus, after removing the packet headers, packet data from all packets with a common stream identifier are concatenated to recover a single elementary stream.

There is wide latitude in how the multiplex is constructed (the size of packets and the relative placement of packets from different streams). The multiplex is constrained primarily by the STD model including the specified buffer sizes. The multiplex must be constructed by the encoder in such a way as to ensure that the STD buffers do not overflow or underflow.

In general, short packets require less STD buffering but more system coding overhead than large packets. Other considerations, such as the need to align packets with the sectors on specific storage media, may influence the choice of packet length. A discussion of these factors is given for the special case of CD-ROM in Clause 1-A.5 of this Annex.

Multiplexing may either occur in conjunction with the encoding of elementary streams or the operations may be independent. If multiplexing is combined with coding then the system is free to use the full range of the STD buffer. If multiplexing is independent from the coding, the elementary encoders must allow sufficient space in the STD buffers to allow for multiplexing. In the case of CD-ROM sector-based multiplexing, a headroom of 6 * 1024 bytes is generally sufficient. This is why the buffering limit is 40 kBytes in the video constrained parameters (see Part 2 of this Standard) and 46 kBytes in the constrained system parameters stream (see Part 1 of this Standard).

Coding for use with a bursty DSM or channel in general requires additional buffering in the STD model beyond that required with a constant-latency DSM or channel. The additional buffering required may be reduced through the careful use of multiplexing and the mux_rate field. The STD uses a byte arrival schedule specified by the SCR and mux_rate fields. In some cases the STD byte arrival schedule can be made to duplicate the actual delivery schedule of the bursty DSM or channel, permitting optimization of STD buffer usage.



STD data input arrival schedule
with mux_rate > average rate between SCR's

**Figure 1-A.2**

### 1-A.2.4 Encoder Constraints caused by Decoder Buffering

It is the encoding system's responsibility to ensure that the STD model never overflows or underflows its buffers. Encoders must specify the sizes of the buffers used in the STD model for each stream, and limits on the sizes of the STD buffers valid for the entire ISO 11172 stream must be placed in the system header packet. The STD model specifies the exact times when each byte of coded data enters and leaves each buffer in terms of the common system_clock_frequency. This timing is specified by a byte arrival schedule, which is specified in the data stream via the SCR and mux_rate fields, and by a byte removal schedule which is specified by the PTS and DTS fields, the audio sample rate, audio AU and PU sizes, the video picture rate, and the STD model with instantaneous elementary stream decoders.

The STD is a model of a decoder. Encoders must apply the STD model in the creation of multiplexed bitstreams, but real decoders need not be implemented with the same architecture as the STD decoder. Variations in access unit size cause the elementary stream encoders to contribute to part of the STD buffer occupancy, and the action of multiplexing contributes to the rest. In the STD model a single buffer is used for both demultiplexing and elementary stream decoding.

Encoders setting the CSPS_flag must not specify STD buffer sizes larger than those permitted in the constrained system parameter limits. For audio, the limit is 4096 bytes. For video the limit depends on both the picture size and the video bitrate, $R_V$. If either the picture size is less than or equal to the maximum allowed by the constrained parameter video stream (see Part 2 of this Standard) or $R_V$ is less than or equal to 1 856 000 bits/second then the limit is 46 * 1024 bytes. Otherwise the limit is:

$$46 * 1024 * R_V / 1\ 856\ 000 \text{ bytes.}$$

If the video bitrate is variable, the peak video bitrate throughout the video stream is used for $R_V$ in this formula.

## 1-A.3 Data Stream Characteristics

### System Header

The System Header is a special packet that contains no elementary stream data. Instead it indicates decoding requirements for each of the elementary streams. It indicates a number of limits that apply to the entire ISO 11172 stream, such as data rate, the number of audio and video streams, and the STD buffer size limits for the individual elementary streams. A decoding system may use these limits to establish its ability to play the stream.

The system header contains a flag indicating whether or not the data stream is encoded for constant rate delivery to the STD. If the data rate averaged over the time intervals between SCRs is constant throughout the stream and equal to the value in the mux_rate field, the constant rate flag may be set. If the mux_rate fields indicate a rate higher than this, data is delivered to the STD in bursts at the rates indicated by the mux_rate fields. The mux_rate field will never be lower than that implied by the SCR fields.

The system header must be the first pack of the ISO 11172 stream. It may be repeated within the stream as often as necessary. In broadcast applications this may be desirable.

Real-time encoding systems must calculate suitable limits for the values in the header before starting to encode. Non-real-time encoders may make two passes over the data to find suitable values.

### Padding stream

A padding stream is provided. It may be used to maintain a constant total data rate, to achieve sector alignment, or to prevent buffer underflow. As the padding stream is not associated with decoding and presentation, it has neither a buffer in the STD model nor PTS or DTS fields.

Stuffing of up to 16 bytes is allowed within each data packet. This can be used for similar purposes to the padding stream and is well suited to providing word (16-bit) or long word (32-bit) alignment in applications where 8-bit alignment is not sufficient.

### Private Streams

Two private stream types, private_stream_1 and private_stream_2, are provided for applications not defined in ISO 11172. Private_stream_1 follows the same syntax as audio and video streams. It may contain stuffing bytes, a buffer size field, and PTS and DTS fields. The use of these fields is not specified in ISO 11172. Private_stream_2 is similar except that no syntax is specified for stuffing bytes, buffer sizes, PTS or DTS fields.

Although only two private stream identifiers are provided, private streams may be designed to include branching fields to support an unlimited number of private sub-streams. This mechanism is not defined in ISO 11172.

## 1-A.4 Decoder Operations

Figures 1-A.3 and 1-A.4 show two different models of an implementation of a decoding system that are used in the following sub-clauses to illustrate the operation of the system. Both models represent possible implementations.

### 1-A.4.1 Decoder synchronization

#### Time-Stamps

The PTS, DTS and SCR fields are the basis for synchronization in decoders. Decoders parse the data stream and extract the PTS or DTS fields contained in the Packet coding layer and the relevant coded data. The PTS and DTS fields are associated with the first access unit (AU) that commences in a packet containing a PTS and/or DTS field. Picture start codes and audio syncwords are not necessarily located at the start of packets, and there may be more than one AU commencing in a packet.

PTS and DTS fields are not necessarily encoded for each picture or audio PU. They are required to occur with intervals not exceeding 0,7 seconds. This bound allows the construction of a control loop using the PTS values which has guaranteed stability with a known bandwidth. For those PUs for which PTS is not encoded, the decoder can approximate the correct value as the sum of the most recent PTS and an increment. The increment is the nominal number of system_clock_frequency cycles per PU times the number of PUs since the last PTS.

DTSs specify the time at which all the bytes of an access unit are removed from the buffer of an elementary stream decoder in the STD model. The STD model assumes instantaneous decoding of access units. In audio streams, and for B-pictures in video streams, the decoding time is the same as the presentation time and so only the PTSs are encoded; DTS values are implied. In video streams, for I-pictures and P-pictures the DTS values are nominally equal to the PTS value minus the number of picture periods of video reordering delay times the picture period in units of the 90kHz STC. The DTS and PTS need not be encoded for every access unit. Intervening values may be calculated from known DTS and PTS values and the rate of PUs for each stream.

Similarly SCR values, which measure the time of events in the coded data stream, are required to occur with intervals not exceeding 0,7 seconds. Again, this allows construction of a controller using SCR values with a guaranteed stability.

### Clock relationships

A decoding system, including all of the synchronized decoders and the source of the coded data, must have exactly one independent time-master. This fact is a natural result of the requirement to avoid overflow and underflow in finite size buffers, while maintaining synchronization of the presentation of data. All other synchronized entities must slave the timing of their operation to the time-master. If a decoder attempts to have more than one simultaneous time-master it may experience problems with buffer management or synchronization.

A decoder system has complete freedom in choosing which entity is the time-master. Typically these entities include the video decoder, the audio decoder, a separate STC, or the data source. Whichever entity is the time-master must communicate to the others the correct value of the STC. A time slave will typically maintain a local STC which is incremented nominally at 90kHz between updates or corrections. In this way each entity has a continuously updated value of the STC which is nominally correct and which it uses to compare with the time-stamps.

Two examples are presented to illustrate different approaches to designing a decoder. One uses the audio decoder's clock as the time-master, and the other relies on the DSM clock as the time-master.

### Example: Audio as time-master

In this first example, the audio decoder is the time-master in a decoding system. Its operation is described here and illustrated in Figure 1-A.3.

The system time clock (STC) is typically initialized to be equal to the value encoded in the first SCR field when that field enter the decoder's buffer. Thereafter the audio decoder controls the STC. As the audio decoder decodes audio AUs and presents audio PUs, it finds PTS fields associated with some of the audio PUs. As the beginning of each PU is output to the user, the associated PTS field contains the correct value of the decoder's STC in an idealized decoder following the STD model. The audio decoder may use this value to update the STC immediately, or to control the STC values via a control loop.

The other decoders then use this STC to determine the correct time to present their decoded data, at the times when their PTS fields are equal to the current value of the STC.

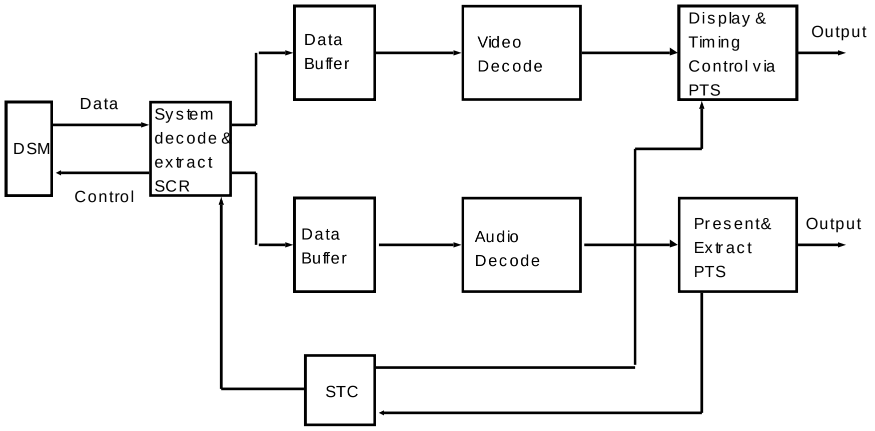


**Figure 1-A.3**

Note that the data source (DSM or channel) must provide data to the decoder on a schedule determined by the SCR and mux_rate field values and the decoder's STC. This time relationship is necessary in order to manage the decoder buffers. Buffer management is described further Clause 1-A.4.3.

The DSM control mechanism obtains data from the DSM at a rate at least equal to that specified in the mux_rate field for each pack until the next SCR field is received. It is not necessary to obtain more data from the DSM until the STC value equals the most recently received SCR value. If more data is read, more buffering will be required.

### Example: DSM as time-master

In this second example, illustrated in Figure 1-A.4, the DSM is the time-master, and the audio and video decoders are implemented as separate decoder subsystems, each receiving the complete multiplexed data stream and extracting and using only that portion of the stream needed.

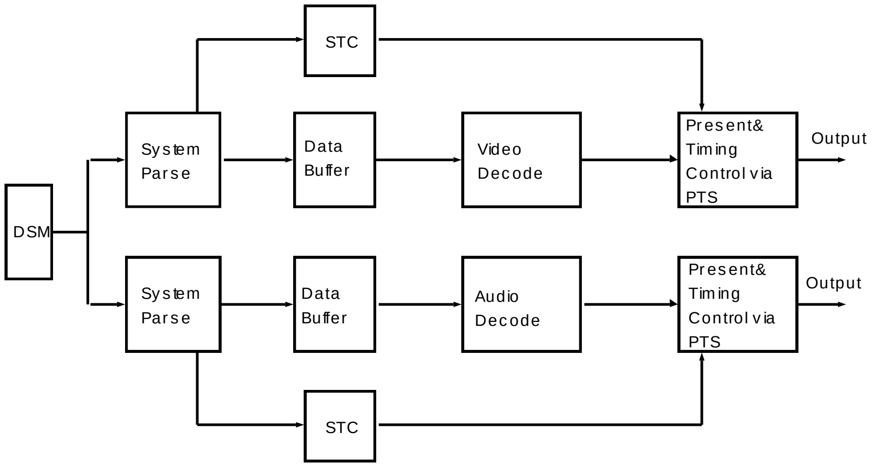


**Figure 1-A.4**

Each decoder receives and parses the complete multiplexed data stream and extracts the system layer information and the coded data needed by that decoder. Synchronization is implemented by the individual decoders slaving their timing to the DSM. The DSM timing is indicated by the SCR fields, which contain the expected value of the decoder's STC at the time that the last byte of the SCR is received by the decoder. Each decoder has a separate STC that is initialized to the first value of SCR received and increments at a nominal 90 kHz rate. The correct timing of the STC is maintained by ensuring that the STC is equal to the SCR values at the time that the SCRs are received. The STC may be maintained either by updating the STC with the value of the SCRs or via a control loop, using the SCR values as reference inputs.

## 1-A.4.2 Decoder Start-up Synchronization

### Finding start codes at random access

The syntax specified in Part 1 of this Standard fully specifies the location of the system start codes. If decoding of the stream begins with the first byte, all start codes can be found without uncertainty. After random access to the stream, or in broadcast applications, where decoding may begin at an arbitrary byte, the problem of finding start codes must be solved.

The thirty-two bit pack and packet start codes are constructed so that they cannot occur in video data, which is expected to be the largest portion of the data in most applications. Therefore parsing can start after a random access with a low probability of incorrectly identifying packet data as a start code. Nonetheless, the probability is not zero because start code emulation may occur in audio or private streams.

If additional protection is desired, for example where a large number of audio streams are multiplexed or there is a large amount of private data, a parser could detect a 32-bit pack_start_code followed 8 bytes later by a 24-bit packet_start_code_prefix. Once a probable system start code is found by a parser, the packet_length field may be used to predict the position of the next start code. In this way the probability of incorrectly identifying coded data as a start code decreases geometrically as each successive start code is found. A decoder performing this function has the options of discarding, or saving, data until parsing is operating with a sufficient level of confidence. Decoding can then begin with the earliest available data for which system start codes are known.

If the application includes a means of directly addressing known system start codes, then the probability of incorrect parsing of start codes can be made zero.

### System Layer Startup Considerations

Once the decoding system has locked on to the data stream it can begin decoding data.

Decoding systems determine the correct time to start decoding by comparing the DTS (or PTS) fields extracted from the stream or computed as described above, with the current value of the STC. The delay from the time the decoder begins to process data until it can begin to present decoded data is bounded from below by the start-up delay implied by the SCR and PTS fields. A decoder following the STD model may produce decoded output as soon as the following conditions are met:

a) At least one SCR field has been extracted and the STC is synchronized with the DSM via the SCRs and mux_rate fields.

b) At least one PTS has been extracted.

c) A complete coded AU is available and the correct associated PTS value (coded or computed) is known.

d) The PTS for an AU which is available is equal to the current STC value.

DTS fields may be used by a decoder to control the input and reorder buffering.

In addition there may be other constraints imposed by the elementary decoders (for example the need for sequence layer information and an I-picture in video coded according to Part 2 of this Standard).

This procedure guarantees that the streams will be synchronized, but it does not necessarily ensure that the start up will be simultaneous. It may be necessary to discard some decoded audio or video and to wait until all elementary decoders are ready. In general there will not be audio and video PUs that start at the same time or with the same PTS values. Thus exactly simultaneous start up of audio and video may require muting some audio samples.

### Coding Layer Startup Considerations

Apart from the system layer decoding considerations, start-up may not be possible immediately, particularly in the case of video. On startup a video decoder requires video sequence layer information and must begin decoding at an I-picture at the start of a group of pictures. In some applications it may be possible to retain the video sequence layer information from a previous access. For more information see Part 2 of this Standard.

#### Compensation of Actual Decoding Delays

The system target decoder (STD) is a model of a decoder. Encoders must apply the STD model in the creation of a multiplexed stream, but real decoders need not be implemented with the STD architecture. In practice real decoders will not be instantaneous. If a real decoder cannot remove and decode an entire access unit instantaneously it will need to delay completion of the processing and presentation compared with the values specified in the STD, and a buffer larger than is specified in the STD will be needed. The effective values of PTS and DTS used for timing may be modified in the decoder to accommodate the decoding delay.

For example, if a video decoder requires one picture period to decode a picture, it may delay completion of decoding all the pictures by one picture period with respect to the values indicated by DTS. This in turn requires additional video decoder buffering (the size of the average coded picture). Proper synchronization can be maintained at the output of the entire system by adding one picture period to the effective values of PTS and DTS for all the elementary streams to compensate for this delay.

#### Channel Smoothing

ISO 11172 streams are, in their most general form, channel independent. They do not assume a specific set of channel characteristics. Decoding from a bursty DSM or channel in general requires additional smoothing buffers not present in the STD model. It is up to decoders to compensate for deviations of a real channel from the STD byte arrival schedule derived from the SCR and mux_rate fields.

In some cases the STD byte arrival schedule can be made to duplicate the actual performance of a bursty DSM or channel. In these cases no extra channel smoothing is required and the performance of the system will be optimized.

### 1-A.4.3 Buffer Management in the Decoder

Buffer management uses the System Target Decoder (STD) model. Elementary stream decoder buffers are assured not to overflow or underflow during decoding as long as the data stream conforms to the specification and the complete decoding system is synchronized in terms of SCR and DTS. The STD model precisely specifies the times at which each data byte enters and leaves the buffer of each elementary stream decoder in terms of a common system time-clock.

Part 2 of this Standard specifies buffer management and start-up delay for video, using the vbv_delay parameter. When video is combined with system layer coding these specifications may conflict. In this case the information in the system layer prevails.

### 1-A.4.4 Time Identification

The absolute time of presentation of the material contained in the coded data stream is indicated in the PTS fields. These fields are defined as modulo $2^{33}$ values of the 90kHz STC, and therefore can indicate programmes of greater than 24 hours in length. If required, PTS fields can be transcoded into other formats such as SMPTE. There is no requirement that the PTS values are initialized to any particular value at the start of the stream.

An application can find the coded data associated with a particular value of presentation time by searching the system layer coding for values of PTS equal to or within an appropriate range of the desired presentation time. Note that SMPTE-like time-codes are also defined in the video coding layer defined in Part 2 of this Standard.

## 1-A.5 Parameters for CD-ROM multiplexing

In this clause an example of a multiplex method for CD-ROM is presented. The example is developed for one video and one audio elementary stream. The multiplexed ISO 11172 stream is stored on a CD-ROM without additional error correction - a mode with 2 324 bytes in each sector. Packs are constructed to be this length so that they may be stored one in each sector. The duration of one sector equals 1/75 second, resulting in a total available bitrate of 8 * 2 324 * 75 = 1 394 400 bits / second.

The audio stream is coded in stereo with the ISO 11172 audio layer II coding method at a bitrate of 192000 bits / second = 24 000 bytes / second. The sample rate used is 44 100 samples / second. Audio presentation units are 1 152 samples each, and so the size of an audio access unit equals:

$$\frac{1\ 152 * 24\ 000}{44\ 100} \text{ bytes}$$

As this result is not an integer, most audio access units are 627 bytes but some are only 626 bytes.

The video stream is coded with a bitrate of 1 158 000 bits / second = 144 750 bytes / second. The value of $B_{vbv}$ used is 36 kBytes, leaving sufficient headroom in the 46 kByte STD buffer of the Constrained System Parameters for the multiplexing.

The packs are designed to coincide with the sectors. Each pack contains a pack header, one packet of coded audio or coded video, and one packet of a padding stream. The padding stream ensures that all audio and video packets contain exactly 2 250 data bytes and that packs are 2 324 bytes long.

The coded audio bitrate of 24 000 bits/ second, with each sector containing 2 250 bytes, requires an average 24 000 / 2 250 = $10^{2}/_{3}$ audio sectors/second. Similarly, the coded video bitrate of 144 750 bytes / second, with 2 250 bytes per sector, requires an average of 144 750 / 2 250 = $64^{1}/_{3}$ video sectors per second. In total, therefore, exactly 75 sectors of audio and video data are required each second for the combined bitstream, exactly filling the total bandwidth of the CD-ROM.

Interleaving the audio and video sectors must not cause the STD buffers to overflow or underflow. Many interleaving schemes are possible that will lead to a multiplexed stream following the Constrained System Parameters. In this example a simple interleaving scheme is used that repeats every 3-seconds (225 sectors). The scheme starts with 6 video sectors followed by one audio sector. This pattern is repeated 31 times, resulting in an interleave of 217 sectors. The last pattern in the interleave scheme consists of 7 video sectors followed by 1 audio sector. The three second period of 225 sectors contains 32 audio sectors and 193 video sectors. On average there are 193/3 = 64 1/3 video sectors/second and 32/3 = 10 2/3 audio sectors/second, as required.

## 1-A.6 Example of an ISO 11172 stream

A sample ISO 11172 stream is presented here to illustrate the syntax and semantic rules governing generation of such streams. This example does not use the same parameters defined in the previous clause. The sample stream is a constrained system parameter stream combining two elementary streams: one video and one audio. The elementary streams are assumed to have been generated with the following specifications:

### 1-A.6.1 Audio

Layer II encoding
48 kHz sample rate
24 000 bytes/sec rate for a pair of stereo channels
1 152 samples per presentation unit
576 bytes per access unit

The stream so generated, with place holders for coded audio and video data, is listed in clause 1-A.6.9 "Sample data stream".

### 1-A.6.2 Video

Constrained parameter video encoding at 150 000 bytes/second.
25 Hz picture rate source.
40 * 1024 Byte video buffer verifier

The order of pictures at the decoder input is:

1I 4P 2B 3B 7P 5B 6B 10P 8B 9B 13I 11B 12B 16P 14B 15B 19P 17B 18B 22P 20B 21B 25I

I pictures coded at 19 000 bytes each
P pictures coded at 10 000 bytes each
B pictures coded at 2 800 or 2 900 (2 875 byte average) each

### 1-A.6.3 Multiplexing strategy

The example employs packets of length 2 048 bytes for both audio and video. The multiplex starts with thirteen video packets to limit audio buffering requirements. Thereafter, one audio packet is interleaved with every 6 to 7 video packets to match the 6.25 ratio of video bit rate to audio bit rate.

For simplicity, packets are constructed with a common number of packet_data_byte entries. Stuffing bytes are used to ensure that all packets have 20 header bytes and 2 028 data bytes.

A pack is generated every third packet. This structure is somewhat arbitrary, but leads to a pack rate of roughly 29 Hz, comfortably over the 1 to 2 Hz requirement of clause 2.4.5.2 (Coding of the system_clock_reference). The cost of such frequent pack formation is not great: all pack headers except the first are 12 bytes long, so pack headers account for some 0.2% of the total bitrate.

The sample bitstream is long word aligned. That is, all packets and all packet data start at 32-bit boundaries. Because the first pack header is 29 bytes long (it contains 17 bytes of system header information), a special padding stream packet appears in the first pack. This 7-byte packet guarantees long word alignment for subsequent packets.

To summarize, the stream is composed of packs and packets as follows:

Pack 1

| | |
|---|---|
| header (includes system_header) | 29 bytes |

| | |
|---|---|
| Padding stream packet | 7 bytes |
| Video packet #1 | 2048 bytes |
| Video packet #2 | 2048 bytes |
| Video packet #3 | 2048 bytes |

Pack 2

| | |
|---|---|
| header | 12 bytes |
| Video packet #4 | 2048 bytes |
| Video packet #5 | 2048 bytes |
| Video packet #6 | 2048 bytes |

Pack 3

| | |
|---|---|
| header | 12 bytes |
| Video packet #7 | 2048 bytes |
| Video packet #8 | 2048 bytes |
| Video packet #9 | 2048 bytes |

Pack 4

| | |
|---|---|
| header | 12 bytes |
| Video packet #10 | 2048 bytes |
| Video packet #11 | 2048 bytes |
| Video packet #12 | 2048 bytes |

Pack 5

| | |
|---|---|
| header | 12 bytes |
| Video packet #13 | 2048 bytes |
| Audio packet #1 | 2048 bytes |
| Video packet #14 | 2048 bytes |

.
.
.

### 1-A.6.4 System Clock Reference (SCR)

Bytes 5 to 9 of every pack header contain encoded system_clock_reference fields. The multiplexed stream's data rate is computed from the data in clauses 1-A.6.1 and 1-A.6.3, and the following formula:

Rmux = (video data rate +audio data rate)(1+packet header +pack header * packs/packet)/data

$$
\begin{aligned}
\text{Rmux} &= (150\ 000 + 24\ 000)\left(1 + \frac{20 + 12/3}{2\ 028}\right) \\
&= 176\ 059 \text{ bytes/sec}
\end{aligned}
$$

$R_{mux}$ and the 90 kHz clock frequency are used by the encoder to convert SCR field byte indexes to system_clock_reference values. The first SCR field, equal to 3 904, simply reflects a non-zero starting value for the encoder's clock. Subsequent SCR fields evaluate to:

| Pack | system_clock_reference |
|---|---|
| 1 | 3 904 |
| 2 | 7 063 |

| | |
|---|---|
| 3 | 10 210 |
| 4 | 13 357 |
| 5 | 16 504 |

To understand the source of these numbers, consider the second pack's SCR value, SCR2 . The SCR2 field occurs 6 180 bytes after the first pack's. SCR2 is related to SCR1 in terms of the elapsed time. For this example's constant rate byte delivery, SCR2 is:

SCR2 = SCR1 + 6 180 * 90 000/176 059
= 7 063

## 1-A.6.5 Presentation Time-stamps (PTS)

The video coding model used for this example leads to coded pictures of the type:

1I 4P 2B 3B 7P 5B 6B 10P 8B 9B 13I 11B 12B 16P 14B 15B 19P 17B 18B 22P 20B 21B 25I

Recalling that coded I, P, and B pictures are assumed to be 19 000, 10 000 and 2 800 to 2 900 bytes, respectively, and that packets contain 2 028 bytes of data each, it follows that picture start codes occur in video packet#1 (I picture), video packet#10 (P picture), video packet#15 (B-picture), etc. This is reflected by the presence of PTS fields in video packets 1, 10, 15, etc., in the sample stream listing.

In this example, N, the number of coded pictures between I pictures equals 12. The number of consecutive B pictures (M-1) between I or P pictures equals two, and thus M=3.

The audio coding model used for this example employs 576 byte access units, hence every 2 048-byte audio packet contains an access unit start code. All audio packets contain PTS fields.

The value of an elementary stream's first Decoding Time-stamp (DTS) field (or PTS if the two are equal) when compared with the initial SCR field, determines the decoder start-up delay for that stream. In the example, the first video DTS field has the value 22 804. The difference between the first pack's SCR value and the first video packet's DTS value is:

start-up delay = (22 804 - 3 904)/90 000 seconds
= 210 msecs.

The encoder has therefore indicated a start-up delay of:

This delay is required to prevent overflow or underflow in the system target decoder. It tells the decoder that the first I picture should be decoded 210 msec and presented 250 msec after reading the last byte of the first SCR field in the multiplexed stream.

Note that the first PTS field in the audio stream equals 26 395, a number slightly lower than the video's. This inequality arises if the video and audio encoders are not turned on at exactly the same instant, and does not imply synchronization error.

The system_audio_lock_flag is set in the system header packet of the the sample bitstream, but the system_video_lock_flag is reset. Therefore, decoders may assume a harmonic relationship between the audio clock and the system time clock, but may not assume such a relationship between the video clock and the system time clock. PTS and DTS value present in the stream are consistent with exact clocks for both video and audio; in practice, however, because the video clock is not locked some draft would appear in video time-stamps. Over one second, or 90 000 clock cycles, errors of 50 parts per million would lead to PTS values differing from the nominal values by 4 or 5. The discrepancy accumulates over time.

## 1-A.6.6 Decoding Time-stamp (DTS)

For I and P pictures, it is generally true that System Target Decoder operations for decoding and presentation occur at different times. Steady state operation with this example's GOP structure (M=3, N=12) leads to I- and P-pictures being decoded three picture periods before their presentation. Thus, video packet #10 has DTS equal to 26 404 but PTS is equal to 37 204. This 10 800 clock cycle, 120 msec difference requires the P-picture to be stored in the system target decoder's reorder buffer for 3 picture periods.

Analysis of DTS and PTS values for the first I-picture (video packet #1) reveals a relationship needed to initialize the reorder buffer. The I-picture is decoded when the decoder's clock reaches 22 804, but nothing is displayed. The initialization is complete 40 msec later when the P-picture discussed in the previous paragraph is decoded, and the I-picture is displayed.

The second audio PTS field (value of 35 035) lags the first by 8 640 clock ticks, or 96 msec. Audio presentation units are 1 152 samples long, which at a 48 kHz sampling rate, corresponds to 24 msec. The second audio PTS field, therefore, appears in the stream after the start code for the fifth audio access unit.

### 1-A.6.7 Buffer Sizes

The example documents a constrained system parameter stream with images conforming to video’s constrained parameters. The maximum allowable buffer sizes in the STD for such streams are used. These are:

Video streams: 46 * 1 024 bytes
Audio streams: 4 * 1 024 bytes

### 1-A.6.8 Adherence to System Target Decoder (STD)

For a multiplexed stream to be a valid ISO 11172 stream, it must play on the system target decoder without overflow or underflow of any STD buffer. Tables 1-A.1 and 1-A.2 track buffer occupancy for the STD video and audio buffers, respectively. The tables demonstrate that the one-second long sample bitstream complies with the STD buffering requirements.

Table 1-A.1: System Target Decoder video buffer occupancy

| Input Picture Index & Type (in coded order) | End-of-picture delivery time (msec) | Decoding/ Presentation time (msec) | Buffer Occupancy (bytes) |
|---|---|---|---|
| -- | 0 | ---- | ---- |
| 1I | 109 | 210/250 | 34 568 |
| 4P | 178 | 250/370 | 21 560 |
| 2B | 194 | 290 | 17 468 |
| 3B | 211 | 330 | 20 928 |
| 7P | 280 | 370/490 | 23 584 |
| 5B | 297 | 410 | 20 196 |
| 6B | 313 | 450 | 22 676 |
| 10P | 382 | 490/610 | 26 664 |
| 8B | 399 | 530 | 21 692 |
| 9B | 427 | 570 | 25 756 |
| 13I | 548 | 610/730 | 27 884 |
| 11B | 564 | 650 | 15 848 |
| 12B | 580 | 690 | 17 900 |
| 16P | 650 | 730/850 | 21 964 |
| 14B | 678 | 770 | 16 816 |
| 15B | 694 | 810 | 20 880 |
| 19P | 763 | 850/970 | 23 016 |

| | | | |
|---|---|---|---|
| 17B | 780 | 890 | 20 072 |
| 18B | 796 | 930 | 22 460 |
| 22P | 866 | 970/1 090 | 26 088 |
| 20B | 882 | 1 010 | 23 052 |
| 21B | 898 | 1 050 | 27 308 |
| 25I | 1 019 | 1 090/1 210 | 31 372 |

The end-of-picture delivery time is the time of arrival of the final byte of the picture at the input of the video buffer in the STD.

In preparing these tables, coded B-pictures were assumed to alternate between 2,800 and 2,900 bytes in a manner leading to an overall video rate of 150,000 bytes/second.

Table 1-A.2: System Target Decoder audio buffer occupancy

| AAU# | end-of-AAU | PTS | Buffer |
|---|---|---|---|
| | delivery time | | Occupancy |
| | (msec) | | (bytes) |
| -- | 152 | ----- | ----- |
| 1 | 155 | 250 | 3 000 |
| 2 | 158 | 274 | 3 480 |
| 3 | 161 | 298 | 2 904 |
| 4 | 246 | 322 | 2 328 |
| 5 | 249 | 346 | 3 780 |
| 6 | 253 | 370 | 3 204 |
| 7 | 256 | 394 | 2 628 |
| 8 | 329 | 418 | 3 904 |
| 9 | 333 | 442 | 3 504 |
| 10 | 336 | 466 | 2 928 |
| 11 | 409 | 490 | 2 444 |
| 12 | 412 | 514 | 3 804 |
| 13 | 416 | 538 | 3 228 |
| 14 | 419 | 562 | 2 652 |
| 15 | 492 | 586 | 2 696 |
| 16 | 496 | 610 | 3 528 |
| 17 | 499 | 634 | 2 952 |
| 18 | 584 | 658 | 2 376 |
| 19 | 587 | 682 | 3 828 |
| 20 | 590 | 706 | 3 252 |
| 21 | 594 | 730 | 2 676 |

Each row in tables 1-A.1 and 1-A.2 indicates timing and buffer occupancy for a single video or audio access unit. The columns in the table are, from left to right:

1) Identification of the access unit.
2) The time of arrival of the final byte of the access unit.
3) The access unit's decoding and presentation time-stamp.
4) The number of bytes in the STD buffer immediately before extraction of the access unit.

Consider, for example, the row in table 1-A.1 for picture 4P. This picture's final byte occurs at byte number 31 444 in the multiplex stream. The stream is delivered at a constant rate of 176 059 bytes per second. Therefore, the delivery of picture 4P is complete 1 000 * 31 444 / 176 059 = 178 msec into the stream. The picture's DTS and PTS values are encoded in the stream. They are 250 msec and 370 msec greater than the SCR of the first pack. At time 250 msec, when the picture is decoded the 22nd packet - an audio packet - is being delivered. At that time the video buffer is not being filled. The buffer contains the contents of exactly 20 video packets, less one I-picture that was extracted 40 msec earlier. The buffer fullness is therefore 20*2 028 - 19 000 = 21 560 bytes.

By comparing decoding times with delivery times it is possible to see that underflow is avoided. So long as anaccess unit has been completely delivered before it is required for decoding, underflow does not occur.

If the maximum buffer fullness immediately before decoding each access unit is compared with the STD buffer size for the stream, it is possible to determine that buffer overflow is avoided. In this example the video stream buffer never exceeds 46 kBytes and the audio buffer never exceeds 4 kBytes. Note that the late placement of the first audio packet is necessary to avoid audio buffer overflow.

### 1-A.6.9 Sample data stream

| # of bytes | field description | coded values |
|---|---|---|
| 4 | pack_start_code (#1) | 000001BA |
| 1 | marker_bit, SCR-32 thru 30 & marker_bit | 21 |
| 2 | SCR-29 thru 15 & marker_bit | 0001 |
| 2 | SCR-14 thru 0 & marker_bit | 1E81(3904) |
| 3 | marker_bit, mux_rate & marker_bit | 801B83 |
| | | |
| 4 | system_header_start_code | 000001BB |
| 1 | header_length | 0C |
| 3 | marker_bit, rate_bound & marker_bit | 801B83 |
| 1 | audio_bound, fixed_flag , CSPS_flag | 07 |
| 1 | system_audio_lock_flag, system_video_lock_flag & marker_bit & video_bound | A1 |
| 1 | reserved_byte | FF |
| 1 | stream_id (audio) | C0 |
| 2 | marker_bit, STD_buffer_scale & STD_buffer_size | C020 |
| 1 | stream_id (video) | E3 |
| 2 | marker_bits, STD_buffer_scale & STD_buffer_size | E02E |
| | | |
| 3 | packet_start_coded_prefix | 000001 |
| 1 | stream_id (padding) | BE |
| 2 | packet_length | 0001 |
| 1 | marker_bit (1_byte) | 0F |
| | | |
| 3 | packet_start_code_prefix (#1V) | 000001 |
| 1 | stream_id (video) | E3 |
| 2 | packet_length | 07FA |
| 2 | stuffing_bytes | FFFF |
| 1 | marker_bit, PTS-32 thru 30 & marker_bit | 31 |
| 2 | PTS-29 thru 15 & marker_bit | 0001 |
| 2 | PTS-14 thru 0 & marker_bit | CE49 |
| 1 | marker_bit, DTS-32 thru 30 & marker_bit | 11 |
| 2 | DTS-29 thru 15 & marker_bit | 0001 |
| 2 | DTS-14 thru 0 & marker_bit | B229 |
| 2 028 | packet_data_byte | XXX...X |
| | | |
| 3 | packet_start_code_prefix (#2V) | 000001 |
| 1 | stream_id | E3 |
| 2 | packet_length | 07FA |
| 14 | stuffing_byte | FF....FF |
| 2 028 | packet_data_byte | XXX...X |
| | | |
| 3 | packet_start_code_prefix (#3V) | 000001 |
| 1 | stream_id | E3 |
| 2 | packet_length | 07FA |
| 14 | stuffing_byte | FF....FF |

|  |  |  |
|---|---|---|
| 2 028 | packet_data_byte | XXX...X |

| | | |
|---|---|---|
| 4 | pack_start_code (#2) | 000001BA |
| 1 | marker_bit, SCR-32 thru 30 & marker_bit | 21 |
| 2 | SCR-29 thru 15 & marker_bit | 0001 |
| 2 | SCR-14 thru 0 & marker_bit | 372F |
| 3 | marker_bit, mux_rate & marker_bit | 801B83 |

| | | |
|---|---|---|
| 3 | packet_start_code_prefix (#4V) | 000001 |
| 1 | stream_id | E3 |
| 2 | packet_length | 07FA |
| 14 | stuffing_bytes | FF....FF |
| 2 028 | packet_data_byte | XXX...X |

| | | |
|---|---|---|
| 3 | packet_start_code_prefix (#5V) | 000001 |
| 1 | stream_id | E3 |
| 2 | packet_length | 07FA |
| 14 | stuffing_byte | FF....FF |
| 2 028 | packet_data_byte | XXX...X |

| | | |
|---|---|---|
| 3 | packet_start_code_prefix (#6V) | 000001 |
| 1 | stream_id | E3 |
| 2 | packet_length | 07FA |
| 14 | stuffing_byte | FF....FF |
| 2 028 | packet_data_byte | XXX...X |

| | | |
|---|---|---|
| 4 | pack_start_code (#3) | 000001BA |
| 1 | marker_bit, SCR-32 thru 30 & marker_bit | 21 |
| 2 | SCR-29 thru 15 & marker_bit | 0001 |
| 2 | SCR-14 thru 0 & marker_bit | 4FC5 |
| 3 | marker_bit, mux_rate & marker_bit | 801B83 |

| | | |
|---|---|---|
| 3 | packet_start_code_prefix (#7V) | 000001 |
| 1 | stream_id | E3 |
| 2 | packet_length | 07FA |
| 14 | stuffing_byte | FF....FF |
| 2 028 | packet_data_byte | XXX...X |

| | | |
|---|---|---|
| 3 | packet_start_code_prefix (#8V) | 000001 |
| 1 | stream_id | E3 |
| 2 | packet_length | 07FA |
| 14 | stuffing_byte | FF....FF |
| 2 028 | packet_data_byte | XXX...X |

| | | |
|---|---|---|
| 3 | packet_start_code_prefix (#9V) | 000001 |
| 1 | stream_id | E3 |
| 2 | packet_length | 07FA |
| 14 | stuffing_byte | FF....FF |
| 2 028 | packet_data_byte | XXX...X 4 |

| | | | |
|---|---|---|---|
| | pack_start_code (#4) | | 000001BA |
| 1 | marker_bit, SCR-32 thru 30 & marker_bit | | 21 |
| 2 | SCR-29 thru 15 & marker_bit | 0001 | |
| 2 | SCR-14 thru 0 & marker_bit | 685B | |
| 3 | marker_bit, mux_rate & marker_bit | 801B83 | |

| | | | |
|---|---|---|---|
| 3 | packet_start_code_prefix (#10V) | | 000001 |
| 1 | stream_id | | E3 |
| 2 | packet_length | | 07FA |
| 2 | stuffing_byte | | FFFF |
| 2 | marker_bit, STD_buffer_scale & STD_buffer_size | | 602E |
| 1 | marker_bit, PTS-32 thru 30 & marker_bit | | 31 |
| 2 | PTS-29 thru 15 & marker_bit | | 0003 |
| 2 | PTS-14 thru 0 & marker_bit | | 22A9 |
| 1 | marker_bit, DTS-32 thru 30 & marker_bit | | 11 |
| 2 | DTS-29 thru 15 & marker_bit | | 0001 |
| 2 | DTS-14 thru 0 & marker_bit | CE49 | |
| 2 028 | packet_data_byte | | XXX...X |

| | | | |
|---|---|---|---|
| 3 | packet_start_code_prefix (#11V) | | 000001 |
| 1 | stream_id | | E3 |
| 2 | packet_length | | 07FA |
| 14 | stuffing_byte | | FF....FF |
| 2 028 | packet_data_byte | | XXX...X |

| | | | |
|---|---|---|---|
| 3 | packet_start_code_prefix (#12V) | | 000001 |
| 1 | stream_id | | E3 |
| 2 | packet_length | | 07FA |
| 14 | stuffing_byte | | FF....FF |
| 2 028 | packet_data_byte | | XXX...X |

| | | | |
|---|---|---|---|
| 4 | pack_start_code (#5) | | 000001BA |
| 1 | marker_bit, SCR-32 thru 30 & marker_bit | | 21 |
| 2 | SCR-29 thru 15 & marker_bit | 0001 | |
| 2 | SCR-14 thru 0 & marker_bit | 80F1 | |
| 3 | marker_bit, mux_rate & marker_bit | 801B83 | |

| | | | |
|---|---|---|---|
| 3 | packet_start_code_prefix (#13V) | | 000001 |
| 1 | stream_id | | E3 |
| 2 | packet_length | | 07FA |
| 14 | stuffing_byte | | FF....FF |
| 2 028 | packet_data_byte | | XXX...X |

| | | | |
|---|---|---|---|
| 3 | packet_start_code_prefix (#1A) | | 000001 |
| 1 | stream_id (audio) | C0 | |
| 2 | packet_length | | 07FA |
| 7 | stuffing_bytes | | FF....FF |
| 2 | marker_bit, STD_buffer_scale & STD_buffer_size | | 4020 |
| 1 | marker_bit, PTS-32 thru 30 & marker_bit | | 31 |
| 2 | PTS-29 thru 15 & marker_bit | | 0001 |
| 2 | PTS-14 thru 0 & marker_bit | | CE37 |
| 2 028 | packet_data_byte | | XXX...X |

| | | | |
|---|---|---|---|
| 3 | packet_start_code_prefix (#14V) | | 000001 |
| 1 | stream_id | | E3 |
| 2 | packet_length | | 07FA |

| | | |
|---|---|---|
| 14 | stuffing_byte | FF....FF |
| 2 028 | packet_data_byte | XXX...X |

| | | | |
|---|---|---|---|
| 4 | pack_start_code (#6) | | 000001BA |
| 1 | marker_bit, SCR-32 thru 30 & marker_bit | | 21 |
| 2 | SCR-29 thru 15 & marker_bit | 0001 | |
| 2 | SCR-14 thru 0 & marker_bit | 9987 | |
| 3 | marker_bit, mux_rate & marker_bit | 801B83 | |

| | | | |
|---|---|---|---|
| 3 | packet_start_code_prefix (#15V) | | 000001 |
| 1 | stream_id | | E3 |
| 2 | packet_length | | 07FA |
| 9 | stuffing_byte | | FF....FF |
| 1 | marker_bit, PTS-32 thru 30 & marker_bit | | 31 |
| 2 | PTS-29 thru 15 & marker_bit | | 0001 |
| 2 | PTS-14 thru 0 & marker_bit | | EA69 |
| 2 028 | packet_data_byte | | XXX...X |

| | | | |
|---|---|---|---|
| 3 | packet_start_code_prefix (#16V) | | 000001 |
| 1 | stream_id | | E3 |
| 2 | packet_length | | 07FA |
| 9 | stuffing_byte | | FF....FF |
| 1 | marker_bit, PTS-32 thru 30 & marker_bit | | 31 |
| 2 | PTS-29 thru 15 & marker_bit | | 0003 |
| 2 | PTS-14 thru 0 & marker_bit | | 0689 |
| 2 028 | packet_data_byte | | XXX...X |

| | | | |
|---|---|---|---|
| 3 | packet_start_code_prefix (#17V) | | 000001 |
| 1 | stream_id | | E3 |
| 2 | packet_length | | 07FA |
| 14 | stuffing_byte | | FF....FF |
| 2 028 | packet_data_byte | | XXX...X |

| | | | |
|---|---|---|---|
| 4 | pack_start_code (#7) | | 000001BA |
| 1 | marker_bit, SCR-32 thru 30 & marker_bit | | 21 |
| 2 | SCR-29 thru 15 & marker_bit | 0001 | |
| 2 | SCR-14 thru 0 & marker_bit | B21D | |
| 3 | marker_bit, mux_rate & marker_bit | 801B83 | |

| | | | |
|---|---|---|---|
| 3 | packet_start_code_prefix (#18V) | | 000001 |
| 1 | stream_id | | E3 |
| 2 | packet_length | | 07FA |
| 2 | stuffing_byte | | FFFF |
| 2 | marker_bit, STD_buffer_scale & STD_buffer_size | | 602E |
| 1 | marker_bit, PTS-32 thru 30 & marker_bit | | 31 |
| 2 | PTS-29 thru 15 & marker_bit | | 0003 |
| 2 | PTS-14 thru 0 & marker_bit | | 7709 |
| 1 | marker_bit, DTS-32 thru 30 & marker_bit | | 11 |
| 2 | DTS-29 thru 15 & marker_bit | | 0003 |
| 2 | DTS-14 thru 0 & marker_bit | 22A9 | |
| 2 028 | packet_data_byte | | XXX...X |

| | | | |
|---|---|---|---|
| 3 | packet_start_code_prefix (#19V) | | 000001 |
| 1 | stream_id | | E3 |
| 2 | packet_length | | 07FA |
| 14 | stuffing_byte | | FF....FF |
| 2 028 | packet_data_byte | | XXX...X |

| | | | |
|---|---|---|---|
| 3 | packet_start_code_prefix (#20V) | | 000001 |

| | | |
|---|---|---|
| 1 | stream_id | E3 |
| 2 | packet_length | 07FA |
| 14 | stuffing_byte | FF....FF |
| 2 028 | packet_data_byte | XXX...X |

| | | | |
|---|---|---|---|
| 4 | pack_start_code (#8) | | 000001BA |
| 1 | marker_bit, SCR-32 thru 30 & marker_bit | | 21 |
| 2 | SCR-29 thru 15 & marker_bit | 0001 | |
| 2 | SCR-14 thru 0 & marker_bit | CAB3 | |
| 3 | marker_bit, mux_rate & marker_bit | 801B83 | |
| | | | |
| 3 | packet_start_code_prefix (#2A) | | 000001 |
| 1 | stream_id (audio) | C0 | |
| 2 | packet_length | | 07FA |
| 7 | stuffing_bytes | | FF....FF |
| 2 | marker_bit, STD_buffer_scale & STD_buffer_size | | 4020 |
| 1 | marker_bit, PTS-32 thru 30 & marker_bit | | 31 |
| 2 | PTS-29 thru 15 & marker_bit | | 0003 |
| 2 | PTS-14 thru 0 & marker_bit | | 11B7 |
| 2 028 | packet_data_byte | | XXX...X |
| | | | |
| 3 | packet_start_code_prefix (#21V) | | 000001 |
| 1 | stream_id | | E3 |
| 2 | packet_length | | 07FA |
| 14 | stuffing_byte | | FF....FF |
| 2 028 | packet_data_byte | | XXX...X |
| | | | |
| 3 | packet_start_code_prefix (#22V) | | 000001 |
| 1 | stream_id | | E3 |
| 2 | packet_length | | 07FA |
| 14 | stuffing_byte | | FF....FF |
| 2 028 | packet_data_byte | | XXX...X |
| : | | | |
| : | | | |
| 4 | iso_11172_end_code | | 000001B9 |