
**Information technology — Coding of
audio-visual objects —**

**Part 3:
Audio**

*Technologies de l'information — Codage des objets audiovisuels —
Partie 3: Codage audio*



Reference number
ISO/IEC 14496-3:2005(E)

AVA Documents
Document Sharing Hub
© ISO/IEC 2005

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

© ISO/IEC 2005

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Foreword

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

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

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

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

ISO/IEC 14496-3 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

This third edition cancels and replaces the second edition (ISO/IEC 14496-3:2001), which has been technically revised. It also incorporates the Amendments ISO/IEC 14496-3:2001/Amd.1:2003, ISO/IEC 14496-3:2001/Amd.2:2004, ISO/IEC 14496-3:2001/Amd.3:2005 and ISO/IEC 14496-3:2001/Amd.6:2005, and the Technical Corrigenda ISO/IEC 14496-3:2001/Cor.1:2002 and ISO/IEC 14496-3:2001/Cor.2:2004.

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

- *Part 1: Systems*
- *Part 2: Visual*
- *Part 3: Audio*
- *Part 4: Conformance testing*
- *Part 5: Reference software*
- *Part 6: Delivery Multimedia Integration Framework (DMIF)*
- *Part 7: Optimized reference software for coding of audio-visual objects* [Technical Report]
- *Part 8: Carriage of ISO/IEC 14496 contents over IP networks*
- *Part 9: Reference hardware description* [Technical Report]
- *Part 10: Advanced Video Coding*

- *Part 11: Scene description and application engine*
- *Part 12: ISO base media file format*
- *Part 13: Intellectual Property Management and Protection (IPMP) extensions*
- *Part 14: MP4 file format*
- *Part 15: Advanced Video Coding (AVC) file format*
- *Part 16: Animation Framework eXtension (AFX)*
- *Part 17: Streaming text format*
- *Part 18: Font compression and streaming*
- *Part 19: Synthesized texture stream*
- *Part 20: Lightweight Application Scene Representation (LSeR) and Simple Aggregation Format (SAF)*

0 Introduction

0.1 Overview

ISO/IEC 14496-3 (MPEG-4 Audio) is a new kind of audio standard that integrates many different types of audio coding: natural sound with synthetic sound, low bitrate delivery with high-quality delivery, speech with music, complex soundtracks with simple ones, and traditional content with interactive and virtual-reality content. By standardizing individually sophisticated coding tools as well as a novel, flexible framework for audio synchronization, mixing, and downloaded post-production, the developers of the MPEG-4 Audio standard have created new technology for a new, interactive world of digital audio.

MPEG-4, unlike previous audio standards created by ISO/IEC and other groups, does not target a single application such as real-time telephony or high-quality audio compression. Rather, MPEG-4 Audio is a standard that applies to every application requiring the use of advanced sound compression, synthesis, manipulation, or playback. The subparts that follow specify the state-of-the-art coding tools in several domains; however, MPEG-4 Audio is more than just the sum of its parts. As the tools described here are integrated with the rest of the MPEG-4 standard, exciting new possibilities for object-based audio coding, interactive presentation, dynamic soundtracks, and other sorts of new media, are enabled.

Since a single set of tools is used to cover the needs of a broad range of applications, *interoperability* is a natural feature of systems that depend on the MPEG-4 Audio standard. A system that uses a particular coder — for example a real-time voice communication system making use of the MPEG-4 speech coding toolset — can easily share data and development tools with other systems, even in different domains, that use the same tool — for example a voicemail indexing and retrieval system making use of MPEG-4 speech coding.

The remainder of this Introduction gives a more detailed overview of the capabilities and functioning of MPEG-4 Audio. First a discussion of concepts, that have changed since the MPEG-2 audio standards, is presented. Then the MPEG-4 Audio toolset is outlined.

0.2 Concepts of MPEG-4 Audio

As with previous MPEG standards, MPEG-4 does not standardize methods for encoding sound. Thus, content authors are left to their own decisions as to the best method of creating bitstream payloads. At the present time, methods to automatically convert natural sound into synthetic or multi-object descriptions are not mature; therefore, most immediate solutions will involve interactively-authoring the content stream in some way. This process is similar to current schemes for MIDI-based and multi-channel mixdown authoring of soundtracks.

Many concepts in MPEG-4 Audio are different from those in previous MPEG Audio standards. For the benefit of readers who are familiar with MPEG-1 and MPEG-2, we provide a brief overview here.

0.2.1 Audio storage and transport facilities

In all of the MPEG-4 tools for audio coding, the coding standard ends at the point of constructing access units that contain the compressed data. The MPEG-4 Systems (ISO/IEC 14496-1) specification describes how to convert these individually coded access units into elementary streams.

There is no standard transport mechanism of these elementary streams over a channel. This is because the broad range of applications that can make use of MPEG-4 technology have delivery requirements that are too wide to easily characterize with a single solution. Rather, what is standardized is an interface (the Delivery Multimedia Interface Format, or DMIF, specified in ISO/IEC 14496-6) that describes the capabilities of a transport layer and the communication between transport, multiplex, and demultiplex functions in encoders and decoders. The use of DMIF and the MPEG-4 Systems specification allows transmission functions that are much more sophisticated than are possible with previous MPEG standards.

However, LATM and LOAS were defined to provide a low overhead audio multiplex and transport mechanism for natural audio applications, which do not require sophisticated object-based coding or other functions provided by MPEG-4 Systems.

The following table gives an overview about the multiplex, storage and transmission formats currently available for MPEG-4 Audio within the MPEG-4 framework:

	Format	Functionality defined in MPEG-4:	Functionality originally defined in:	Description
Multiplex	M4Mux	ISO/IEC 14496-1 (normative)	-	MPEG-4 Multiplex scheme
	LATM	ISO/IEC 14496-3 (normative)	-	Low Overhead Audio Transport Multiplex
Storage	ADIF	ISO/IEC 14496-3 (informative)	ISO/IEC 13818-7 (normative)	Audio Data Interchange Format, (AAC only)
	MP4FF	ISO/IEC 14496-12 (normative)	-	MPEG-4 File Format
Transmission	ADTS	ISO/IEC 14496-3 (informative)	ISO/IEC 13818-7 (normative, exemplarily)	Audio Data Transport Stream, (AAC only)
	LOAS	ISO/IEC 14496-3 (normative, exemplarily)	-	Low Overhead Audio Stream, based on LATM, three versions are available: AudioSyncStream() EPAudioSyncStream() AudioPointerStream()

To allow for a user on the remote side of a channel to dynamically control a server streaming MPEG-4 content, MPEG-4 defines backchannel streams that can carry user interaction information.

0.2.2 MPEG-4 Audio supports low-bitrate coding

Previous MPEG Audio standards have focused primarily on transparent (undetectable) or nearly transparent coding of high-quality audio at whatever bitrate was required to provide it. MPEG-4 provides new and improved tools for this purpose, but also standardizes (and has tested) tools that can be used for transmitting audio at the low bitrates suitable for Internet, digital radio, or other bandwidth-limited delivery. The new tools specified in MPEG-4 are the state-of-the-art tools that support low-bitrate coding of speech and other audio.

0.2.3 MPEG-4 Audio is an object-based coding standard with multiple tools

Previous MPEG Audio standards provided a single toolset, with different configurations of that toolset specified for use in various applications. MPEG-4 provides several toolsets that have no particular relationship to each other, each with a different target function. The profiles of MPEG-4 Audio specify which of these tools are used together for various applications.

Further, in previous MPEG standards, a single (perhaps multi-channel or multi-language) piece of content was transmitted. In contrast, MPEG-4 supports a much more flexible concept of a *soundtrack*. Multiple tools may be used to transmit several *audio objects*, and when using multiple tools together an *audio composition* system is provided to create a single soundtrack from the several audio substreams. User interaction, terminal capability, and speaker configuration may be used when determining how to produce a single soundtrack from the component objects. This capability gives MPEG-4 significant advantages in quality and flexibility when compared to previous audio standards.

0.2.4 MPEG-4 Audio provides capabilities for synthetic sound

In natural sound coding, an existing sound is compressed by a server, transmitted and decompressed at the receiver. This type of coding is the subject of many existing standards for sound compression. In contrast, MPEG-4 standardizes a novel paradigm in which synthetic sound descriptions, including synthetic speech and synthetic

music, are transmitted and then *synthesized* into sound at the receiver. Such capabilities open up new areas of very-low-bitrate but still very-high-quality coding.

0.2.5 MPEG-4 Audio provides capabilities for error robustness

Improved error robustness capabilities for all coding tools are provided through the error-resilient bitstream payload syntax. This tool supports advanced channel coding techniques, which can be adapted to the special needs of given coding tools and a given communications channel. This error-resilient bitstream payload syntax is mandatory for all error resilient object types.

The error protection tool (EP tool) provides unequal error protection (UEP) for MPEG-4 Audio in conjunction with the error-resilient bitstream payload. UEP is an efficient method to improve the error robustness of source coding schemes. It is used by various speech and audio coding systems operating over error-prone channels such as mobile telephone networks or Digital Audio Broadcasting (DAB). The bits of the coded signal representation are first grouped into different classes according to their error sensitivity. Then error protection is individually applied to the different classes, giving better protection to more sensitive bits.

Improved error robustness for AAC is provided by a set of error resilience tools. These tools reduce the perceived degradation of the decoded audio signal that is caused by corrupted bits in the bitstream payload.

0.2.6 MPEG-4 Audio provides capabilities for scalability

Previous MPEG Audio standards provided a single bitrate, single bandwidth toolset, with different configurations of that toolset specified for use in various applications. MPEG-4 provides several bitrate and bandwidth options within a single stream, providing a scalability functionality that permits a given stream to scale to the requirement of different channels and applications or to be responsive to a given channel that has dynamic throughput characteristics. The tools specified in MPEG-4 are the state-of-the-art tools providing scalable compression of speech and audio signals.

0.3 The MPEG-4 Audio tool set

0.3.1 Speech coding tools

0.3.1.1 Overview

Speech coding tools are designed for the transmission and decoding of synthetic and natural speech.

Two types of speech coding tools are provided in MPEG-4. The *natural* speech tools allow the compression, transmission, and decoding of human speech, for use in telephony, personal communication, and surveillance applications. The *synthetic* speech tool provides an interface to text-to-speech synthesis systems; using synthetic speech provides very-low-bitrate operation and built-in connection with facial animation for use in low-bitrate video conferencing applications.

0.3.1.2 Natural speech coding

The MPEG-4 speech coding toolset covers the compression and decoding of natural speech sound at bitrates ranging between 2 and 24 kbit/s. When variable bitrate coding is allowed, coding at even less than 2 kbit/s, for example an average bitrate of 1.2 kbit/s, is also supported. Two basic speech coding techniques are used: One is a parametric speech coding algorithm, HVXC (Harmonic Vector eXcitation Coding), for very low bit rates; and the other is a CELP (Code Excited Linear Prediction) coding technique. The MPEG-4 speech coders target applications range from mobile and satellite communications, to Internet telephony, to packaged media and speech databases. It meets a wide range of requirements encompassing bitrate, functionality and sound quality.

MPEG-4 HVXC operates at fixed bitrates between 2.0 kbit/s and 4.0 kbit/s using a bitrate scalability technique. It also operates at lower bitrates, typically 1.2 - 1.7 kbit/s, using a variable bitrate technique. HVXC provides communications-quality to near-toll-quality speech in the 100 Hz – 3800 Hz band at 8 kHz sampling rate. HVXC also allows independent change of speed and pitch during decoding, which is a powerful functionality for fast access to speech databases. HVXC functionalities including 2.0 - 4.0 kbit/s fixed bitrate modes and a 2.0 kbit/s maximum variable bitrate mode.

Error Resilient (ER) HVXC extends operation of the variable bitrate mode to 4.0 kbit/s to allow higher quality variable rate coding. The ER HVXC therefore provides fixed bitrate modes of 2.0 - 4.0 kbit/s and a variable bitrate of either less than 2.0 kbit/s or less than 4.0 kbit/s, both in scalable and non-scalable modes. In the variable bitrate modes, non-speech parts are detected in unvoiced signals, and a smaller number of bits are used for these non-speech parts to reduce the average bitrate. ER HVXC provides communications-quality to near-toll-quality speech in the 100 Hz - 3800 Hz band at 8 kHz sampling rate. When the variable bitrate mode is allowed, operation at lower average bitrate is possible. Coded speech using variable bitrate mode at typical bitrates of 1.5 kbit/s average, and at typical bitrate of 3.0 kbit/s average has essentially the same quality as 2.0 kbit/s fixed rate and 4.0 kbit/s fixed rate respectively. The functionality of pitch and speed change during decoding is supported for all modes. ER HVXC has a bitstream payload syntax with the error sensitivity classes to be used with the EP-Tool, and some error concealment functionality is supported for use in error-prone channels such as mobile communication channels. The ER HVXC speech coder target applications range from mobile and satellite communications, to Internet telephony, to packaged media and speech databases.

MPEG-4 CELP is a well-known coding algorithm with new functionality. Conventional CELP coders offer compression at a single bit rate and are optimized for specific applications. Compression is one of the functionalities provided by MPEG-4 CELP, but MPEG-4 also enables the use of one basic coder in multiple applications. It provides scalability in bitrate and bandwidth, as well as the ability to generate bitstream payloads at arbitrary bitrates. The MPEG-4 CELP coder supports two sampling rates, namely, 8 kHz and 16 kHz. The associated bandwidths are 100 Hz – 3800 Hz for 8 kHz sampling and 50 Hz – 7000 Hz for 16 kHz sampling. The silence compression tool comprises a voice activity detector (VAD), a discontinuous transmission (DTX) unit and a comfort noise generator (CNG) module. The tool encodes/decodes the input signal at a lower bitrate during the non-active-voice (silent) frames. During the active-voice (speech) frames, MPEG-4 CELP encoding and decoding are used.

The silence compression tool reduces the average bitrate thanks to compression at a lower-bitrate for silence. In the encoder, a voice activity detector is used to distinguish between regions with normal speech activity and those with silence or background noise. During normal speech activity, the CELP coding is used. Otherwise a silence insertion descriptor (SID) is transmitted at a lower bitrate. This SID enables a comfort noise generator (CNG) in the decoder. The amplitude and the spectral shape of this comfort noise are specified by energy and LPC parameters in methods similar to those used in a normal CELP frame. These parameters are optionally re-transmitted in the SID and thus can be updated as required.

MPEG has conducted extensive verification testing in realistic listening conditions in order to prove the efficacy of the speech coding toolset.

0.3.1.3 Text-to-speech interface

Text-to-speech (TTS) capability is becoming a rather common media type and plays an important role in various multi-media application areas. For instance, by using TTS functionality, multimedia content with narration can be easily created without recording natural speech. Before MPEG-4, however, there was no way for a multimedia content provider to easily give instructions to an unknown TTS system. With **MPEG-4 TTS Interface**, a single common interface for TTS systems is standardized. This interface allows speech information to be transmitted in the international phonetic alphabet (IPA), or in a textual (written) form of any language.

The **MPEG-4 Hybrid/Multi-Level Scalable TTS Interface** is a superset of the conventional TTS framework. This extended TTS Interface can utilize prosodic information taken from natural speech in addition to input text and can thus generate much higher-quality synthetic speech. The interface and its bitstream payload format is scalable in terms of this added information; for example, if some parameters of prosodic information are not available, a decoder can generate the missing parameters by rule. Normative algorithms for speech synthesis and text-to-phoneme translation are not specified in MPEG-4, but to meet the goal that underlies the MPEG-4 TTS Interface, a decoder should fully utilize all the provided information according to the user's requirements level.

As well as an interface to text-to-speech synthesis systems, MPEG-4 specifies a joint coding method for phonemic information and facial animation (FA) parameters and other animation parameters (AP). Using this technique, a single bitstream payload may be used to control both the text-to-speech interface and the facial animation visual object decoder (see ISO/IEC 14496-2, Annex C). The functionality of this extended TTS thus ranges from conventional TTS to natural speech coding and its application areas, from simple TTS to audio presentation with TTS and motion picture dubbing with TTS.

0.3.2 Audio coding tools

0.3.2.1 Overview

Audio coding tools are designed for the transmission and decoding of recorded music and other audio soundtracks.

0.3.2.2 General audio coding tools

MPEG-4 standardizes the coding of natural audio at bitrates ranging from 6 kbit/s up to several hundred kbit/s per audio channel for mono, two-channel-, and multi-channel-stereo signals. General high-quality compression is provided by incorporating the MPEG-2 AAC standard (ISO/IEC 13818-7), with certain improvements, as MPEG-4 AAC. At 64 kbit/s/channel and higher ranges, this coder has been found in verification testing under rigorous conditions to meet the criterion of “indistinguishable quality” as defined by the European Broadcasting Union.

General audio (GA) coding tools comprise the AAC tool set expanded by alternative quantization and coding schemes (Twin-VQ and BSAC). The general audio coder uses a perceptual filterbank, a sophisticated masking model, noise-shaping techniques, channel coupling, and noiseless coding and bit-allocation to provide the maximum compression within the constraints of providing the highest possible quality. Psychoacoustic coding standards developed by MPEG have represented the state-of-the-art in this technology since MPEG-1 Audio; MPEG-4 General Audio coding continues this tradition.

For bitrates ranging from 6 kbit/s to 64 kbit/s per channel, the MPEG-4 standard provides extensions to the GA coding tools, that allow the content author to achieve the highest quality coding at the desired bitrate. Furthermore, various bit rate scalability options are available within the GA coder. The low-bitrate techniques and scalability modes provided within this tool set have also been verified in formal tests by MPEG.

The **MPEG-4 low delay** coding functionality provides the ability to extend the usage of generic low bitrate audio coding to applications requiring a very low delay in the encoding / decoding chain (e.g. full-duplex real-time communications). In contrast to traditional low delay coders based on speech coding technology, the concept of this low delay coder is based on general perceptual audio coding and is thus suitable for a wide range of audio signals. Specifically, it is derived from the proven architecture of MPEG-2/4 Advanced Audio Coding (AAC) and all capabilities for coding of 2 (stereo) or more sound channels (multi-channel) are available within the low delay coder. It operates at up to 48 kHz sampling rate and uses a frame length of 512 or 480 samples, compared to the 1024 or 960 samples used in standard MPEG-2/4 AAC to enable coding of general audio signals with an algorithmic delay not exceeding 20 ms. Also the size of the window used in the analysis and synthesis filterbank is reduced by a factor of 2. No block switching is used to avoid the “look-ahead” delay due to the block switching decision. To reduce pre-echo artefacts in the case of transient signals, window shape switching is provided instead. For non-transient portions of the signal a sine window is used, while a so-called low overlap window is used for transient portions. Use of the bit reservoir is minimized in the encoder in order to reach the desired target delay. As one extreme case, no bit reservoir is used at all.

The **MPEG-4 BSAC** is used in combination with the AAC coding tools and replaces the noiseless coding of the quantized spectral data and the scalefactors. The MPEG-4 BSAC provides fine grain scalability in steps of 1 kbit/s per audio channel, i.e. 2 kbit/s steps for a stereo signal. One base layer stream and many small enhancement layer streams are used. To obtain fine step scalability, a bit-slicing scheme is applied to the quantized spectral data. First the quantized spectral values are grouped into frequency bands. Each of these groups contains the quantized spectral values in their binary representation. Then the bits of a group are processed in slices according to their significance. Thus all most significant bits (MSB) of the quantized values in a group are processed first. These bit-slices are then encoded using an arithmetic coding scheme to obtain entropy coding with minimal redundancy. In order to implement fine grain scalability efficiently using MPEG-4 Systems tools, the fine grain audio data can be grouped into large-step layers and these large-step layers can be further grouped by concatenating large-step layers from several sub-frames. Furthermore, the configuration of the payload transmitted over an Elementary Stream (ES) can be changed dynamically (by means of the MPEG-4 backchannel capability) depending on the environment, such as network traffic or user interaction. This means that BSAC can allow for real-time adjustments to the quality of service. In addition to fine grain scalability, it can improve the quality of an audio signal that is decoded from a stream transmitted over an error-prone channel, such as a mobile communication networks or Digital Audio Broadcasting (DAB) channel.

MPEG-4 SBR (Spectral Band Replication) is a bandwidth extension tool used in combination with the AAC general audio codec. When integrated into the MPEG AAC codec, a significant improvement of the performance is available, which can be used to lower the bitrate or improve the audio quality. This is achieved by replicating the

highband, i.e. the high frequency part of the spectrum. A small amount of data representing a parametric description of the highband is encoded and used in the decoding process. The data rate is by far below the data rate required when using conventional AAC coding of the highband.

0.3.2.3 Parametric audio coding tools

The parametric audio coding tool **MPEG-4 HILN** (Harmonic and Individual Lines plus Noise) codes non-speech signals like music at bitrates of 4 kbit/s and higher using a parametric representation of the audio signal. The basic idea of this technique is to decompose the input signal into audio objects which are described by appropriate source models and represented by model parameters. Object models for sinusoids, harmonic tones, and noise are utilized in the HILN coder. HILN allows independent change of speed and pitch during decoding.

The Parametric Audio Coding tools combine very low bitrate coding of general audio signals with the possibility of modifying the playback speed or pitch during decoding without the need for an effects processing unit. In combination with the speech and audio coding tools in MPEG-4, improved overall coding efficiency is expected for applications of object based coding allowing selection and/or switching between different coding techniques.

This approach allows to introduce a more advanced source model than just assuming a stationary signal for the duration of a frame, which motivates the spectral decomposition used in e.g. the MPEG-4 General Audio Coder. As known from speech coding, where specialized source models based on the speech generation process in the human vocal tract are applied, advanced source models can be advantageous, especially for very low bitrate coding schemes.

Due to the very low target bitrates, only the parameters for a small number of objects can be transmitted. Therefore a perception model is employed to select those objects that are most important for the perceptual quality of the signal.

In HILN, the frequency and amplitude parameters are quantized according to the “just noticeable differences” known from psychoacoustics. The spectral envelope of the noise and the harmonic tones are described using LPC modeling as known from speech coding. Correlation between the parameters of one frame and those of consecutive frames is exploited by parameter prediction. Finally, the quantized parameters are entropy coded and multiplexed to form a bitstream payload.

A very interesting property of this parametric coding scheme arises from the fact that the signal is described in terms of frequency and amplitude parameters. This signal representation permits speed and pitch change functionality by simple parameter modification in the decoder. The HILN Parametric Audio Coder can be combined with MPEG-4 Parametric Speech Coder (HVXC) to form an integrated parametric coder covering a wider range of signals and bitrates. This integrated coder supports speed and pitch change. Using a speech/music classification tool in the encoder, it is possible to automatically select the HVXC for speech signals and the HILN for music signals. Such automatic HVXC/HILN switching was successfully demonstrated and the classification tool is described in the informative Annex of the MPEG-4 standard.

MPEG-4 SSC, (Sinusoidal Coding) is a parametric coding tool that is capable of full bandwidth high quality audio coding. The coding tool dissects a monaural or stereo audio signal into a number of different objects that each can be parameterized efficiently and encoded at a low bit-rate. These objects are, transients: representing dynamic changes in the temporal domain, sinusoids: representing deterministic components, and noise: representing components that do not have a clear temporal or spectral localisation. The fourth object, that is only relevant for stereo input signals, captures the stereo image. As the signal is represented in a parametric domain, independent, high quality pitch and tempo scaling are possible at low computational cost.

0.3.3 Lossless audio coding tools

MPEG-4 DST (Direct Stream Transfer) provides lossless coding of oversampled audio signals.

0.3.4 Synthesis tools

Synthesis tools are designed for very low bitrate description and transmission, and terminal-side synthesis, of synthetic music and other sounds.

The MPEG-4 toolset providing general audio synthesis capability is called **MPEG-4 Structured Audio**, and it is described in subpart 5 of ISO/IEC 14496-3. MPEG-4 Structured Audio (the SA coder) provides very general capabilities for the description of synthetic sound, and the normative creation of synthetic sound in the decoding terminal. High-quality stereo sound can be transmitted at bitrates from 0 kbit/s (no continuous cost) to 2-3 kbit/s for extremely expressive sound using these tools.

Rather than specify a particular method of synthesis, SA specifies a flexible language for describing methods of synthesis. This technique allows content authors two advantages. First, the set of synthesis techniques available is not limited to those that were envisioned as useful by the creators of the standard; any current or future method of synthesis may be used in MPEG-4 Structured Audio. Second, the creation of synthetic sound from structured descriptions is normative in MPEG-4, so sound created with the SA coder will sound the same on any terminal.

Synthetic audio is transmitted via a set of *instrument* modules that can create audio signals under the control of a *score*. An instrument is a small network of signal-processing primitives that control the parametric generation of sound according to some algorithm. Several different instruments may be transmitted and used in a single Structured Audio bitstream payload. A score is a time-sequenced set of commands that invokes various instruments at specific times to contribute their output to an overall music performance. The format for the description of instruments is SAOL, the Structured Audio Orchestra Language. The format for the description of scores is SASL, the Structured Audio Score Language.

Efficient transmission of sound samples, also called *wavetables*, for use in sampling synthesis is accomplished by providing interoperability with the MIDI Manufacturers Association Downloaded Sounds Level 2 (DLS-2) standard, which is normatively referenced by the Structured Audio standard. By using the DLS-2 format, the simple and popular technique of wavetable synthesis can be used in MPEG-4 Structured Audio soundtracks, either by itself or in conjunction with other kinds of synthesis using the more general-purpose tools. To further enable interoperability with existing content and authoring tools, the popular MIDI (Musical Instrument Digital Interface) control format can be used instead of, or in addition to, scores in SASL for controlling synthesis.

Through the inclusion of compatibility with MIDI standards, MPEG-4 Structured Audio thus represents a unification of the current technique for synthetic sound description (MIDI-based wavetable synthesis) with that of the future (general-purpose algorithmic synthesis). The resulting standard solves problems not only in very-low-bitrate coding, but also in virtual environments, video games, interactive music, karaoke systems, and many other applications.

0.3.5 Composition tools

Composition tools are designed for object-based coding, interactive functionality, and audiovisual synchronization.

The tools for audio composition, like those for visual composition, are specified in the MPEG-4 Systems standard (ISO/IEC 14496-1). However, since readers interested in audio functionality are likely to look here first, a brief overview is provided.

Audio composition is the use of multiple individual “audio objects” and mixing techniques to create a single soundtrack. It is analogous to the process of recording a soundtrack in a multichannel mix, with each musical instrument, voice actor, and sound effect on its own channel, and then “mixing down” the multiple channels to a single channel or single stereo pair. In MPEG-4, the multichannel mix itself may be transmitted, with each audio source using a different coding tool, and a set of instructions for mixdown also transmitted in the bitstream payload. As the multiple audio objects are received, they are decoded separately, but not played back to the listener; rather, the instructions for mixdown are used to prepare a single soundtrack from the “raw material” given in the objects. This final soundtrack is then played for the listener.

An example serves to illustrate the efficacy of this approach. Suppose, for a certain application, we wish to transmit the sound of a person speaking in a reverberant environment over stereo background music, at very high quality. A traditional approach to coding would demand the use of a general audio coding at 32 kbit/s/channel or above; the sound source is too complex to be well-modeled by a simple model-based coder. However, in MPEG-4 we can represent the soundtrack as the conjunction of several objects: a speaking person passed through a reverberator added to a synthetic music track. We transmit the speaker’s voice using the CELP tool at 16 kbit/s, the synthetic music using the SA tool at 2 kbit/s, and allow a small amount of overhead (only a few hundreds of bytes as a fixed cost) to describe the stereo mixdown and the reverberation. Using MPEG-4 and an object-based approach thus allows us to describe in less than 20 kbit/s total a stream that might require 64 kbit/s to transmit with traditional coding, at equivalent quality.

Additionally, having such structured soundtrack information present in the decoding terminal allows more sophisticated client-side interaction to be included. For example, the listener can be allowed (if the content author desires) to request that the background music be muted. This functionality would not be possible if the music and speech were coded into the same audio track.

With the **MPEG-4 Binary Format for Scenes (BIFS)**, specified in MPEG-4 Systems, a subset tool called AudioBIFS allows content authors to describe sound scenes using this object-based framework. Multiple sources may be mixed and combined, and interactive control provided for their combination. Sample-resolution control over mixing is provided in this method. Dynamic download of custom signal-processing routines allows the content author to exactly request a particular, normative, digital filter, reverberator, or other effects-processing routine. Finally, an interface to terminal-dependent methods of 3-D audio spatialisation is provided for the description of virtual-reality and other 3-D sound material.

As AudioBIFS is part of the general BIFS specification, the same framework is used to synchronize audio and video, audio and computer graphics, or audio with other material. Please refer to ISO/IEC 14496-1 (MPEG-4 Systems) for more information on AudioBIFS and other topics in audiovisual synchronization.

0.3.6 Scalability tools

Scalability tools are designed for the creation of bitstream payloads that can be transmitted, without recoding, at several different bitrates.

Many of the stream types in MPEG-4 are *scalable* in one manner or another. Several types of scalability in the standard are discussed below.

Bitrate scalability allows a bitstream payload to be parsed into a bitstream payload of lower bitrate such that the combination can still be decoded into a meaningful signal. The bitstream payload parsing can occur either during transmission or in the decoder. Scalability is available within each of the natural audio coding schemes, or by a combination of different natural audio coding schemes.

Bandwidth scalability is a particular case of bitrate scalability, whereby part of a bitstream payload representing a part of the frequency spectrum can be discarded during transmission or decoding. This is available for the CELP speech coder, where an extension layer converts the narrow band base layer speech coder into a wide band speech coder. Also the general audio coding tools which all operate in the frequency domain offer a very flexible bandwidth control for the different coding layers.

Encoder complexity scalability allows encoders of different complexity to generate valid and meaningful bitstream payloads. An example for this is the availability of a high quality and a low complexity excitation module for the wideband CELP coder allowing to choose between significant lower encoder complexity or optimized coding quality.

Decoder complexity scalability allows a given bitstream payload to be decoded by decoders of different levels of complexity. A subtype of decoder complexity scalability is *graceful degradation*, in which a decoder dynamically monitors the resources available, and scales down the decoding complexity (and thus the audio quality) when resources are limited. The Structured Audio decoder allows this type of scalability; a content author may provide (for example) several different algorithms for the synthesis of piano sounds, and the content itself decides, depending on available resources, which one to use.

0.3.7 Upstream

Upstream tools are designed for the dynamic control the streaming of the server for bitrate control and quality feedback control.

The **MPEG-4 upstream** or backchannel allows a user on a remote side to dynamically control the streaming of MPEG-4 content from a server. Backchannel streams carrying the user interaction information.

0.3.8 Error robustness facilities

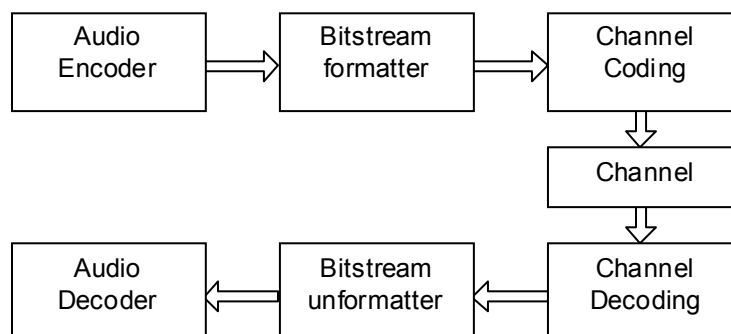
0.3.8.1 Overview

Error robustness facilities include tools for error resilience as well as for error protection.

The error robustness facilities provide improved performance on error-prone transmission channels. They are comprised of error resilient bitstream payload reordering, a common error protection tool and codec specific error resilience tools.

0.3.8.2 Error resilient bitstream payload reordering

Error resilient bitstream payload reordering allows the effective use of advanced channel coding techniques like unequal error protection (UEP), which can be perfectly adapted to the needs of the different coding tools. The basic idea is to rearrange the audio frame content depending on its error sensitivity in one or more instances belonging to different error sensitivity categories (ESC). This rearrangement can be either data element-wise or even bit-wise. An error resilient bitstream payload frame is build by concatenating these instances.



The basic principle is depicted in the figure above. A bitstream payload is reordered according to the error sensitivity of single bitstream payload elements or even single bits. This new arranged bitstream payload is channel coded, transmitted and channel decoded. Prior to audio decoding, the bitstream payload is rearranged to its original order.

0.3.8.3 Error protection

The EP tool provides unequal error protection. It receives several classes of bits from the audio coding tools, and then applies forward error correction codes (FEC) and/or cyclic redundancy codes (CRC) for each class, according to its error sensitivity.

The error protection tool (EP tool) provides the unequal error protection (UEP) capability to the set of ISO/IEC 14496-3 codecs. Main features of this tool are:

- providing a set of error correcting/detecting codes with wide and small-step scalability, both in performance and in redundancy
- providing a generic and bandwidth-efficient error protection framework, which covers both fixed-length frame streams and variable-length frame streams
- providing a UEP configuration control with low overhead.

0.3.8.4 Error resilience tools for AAC

Several tools are provided to increase the error resilience for AAC. These tools improve the perceived audio quality of the decoded audio signal in case of corrupted bitstream payloads, which may occur e. g. in the presence of noisy transmission channels.

- The *Virtual CodeBooks tool (VCB11)* extends the sectioning information of an AAC bitstream payload. This permits the detection of serious errors within the spectral data of an MPEG-4 AAC bitstream payload.

Virtual codebooks are used to limit the largest absolute value possible within any scalefactor band that uses escape values. While all virtual codebooks use the codebook 11, the sixteen virtual codebooks introduced by VCB11 provide sixteen different limitations of the spectral values belonging to the corresponding subclause. Therefore, errors in the transmission of spectral data that result in spectral values exceeding the indicated limit can be located and appropriately concealed.

- The *Reversible Variable Length Coding tool (RVLC)* replaces the Huffman and DPCM coding of the scalefactors in an AAC bitstream payload. The RVLC uses symmetric codewords to enable both forward and backward decoding of the scalefactor data. In order to have a starting point for backward decoding, the total number of bits of the RVLC part of the bitstream payload is transmitted. Because of the DPCM coding of the scalefactors, also the value of the last scalefactor is transmitted to enable backward DPCM decoding. Since not all nodes of the RVLC code tree are used as codewords, some error detection is also possible.
- The *Huffman codeword reordering (HCR)* algorithm for AAC spectral data is based on the fact that some of the codewords can be placed at known positions so that these codewords can be decoded independent of any error within other codewords. Therefore, this algorithm avoids error propagation to those codewords, the so-called priority codewords (PCW). To achieve this, segments of known length are defined and those codewords are placed at the beginning of these segments. The remaining codewords (non-priority codewords, non-PCW) are filled into the gaps left by the PCWs using a special algorithm that minimizes error propagation to the non-PCWs codewords. This reordering algorithm does not increase the size of spectral data. Before applying the reordering algorithm, the PCWs are determined by sorting the codewords according to their importance.

Information technology — Coding of audio-visual objects —

Part 3: Audio

ISO/IEC 14496-3 contains ten subparts:

Subpart 1: Main

Subpart 2: Speech coding — HVXC

Subpart 3: Speech coding — CELP

Subpart 4: General Audio coding (GA) — AAC, TwinVQ, BSAC

Subpart 5: Structured Audio (SA)

Subpart 6: Text To Speech Interface (TTSI)

Subpart 7: Parametric Audio Coding — HILN

Subpart 8: Parametric coding for high quality audio — SSC

Subpart 9: MPEG-1/2 Audio in MPEG-4

Subpart 10: Lossless coding of oversampled audio — DST

Content for Subpart 1

1.1	Scope	3
1.2	Normative references	3
1.3	Terms and definitions	4
1.4	Symbols and abbreviations	16
1.4.1	Arithmetic operators	16
1.4.2	Logical operators	17
1.4.3	Relational operators	17
1.4.4	Bitwise operators	17
1.4.5	Assignment	18
1.4.6	Mnemonics	18
1.4.7	Constants	18
1.4.8	Method of describing bitstream payload syntax	18
1.5	Technical overview	20
1.5.1	MPEG-4 audio object types	20
1.5.2	Audio profiles and levels	26
1.6	Interface to ISO/IEC 14496-1 (MPEG-4 Systems)	36
1.6.1	Introduction	36
1.6.2	Syntax	36
1.6.3	Semantics	41
1.6.4	Upstream	44
1.6.5	Signaling of SBR	46
1.7	MPEG-4 Audio transport stream	49
1.7.1	Overview	49
1.7.2	Synchronization Layer	50
1.7.3	Multiplex Layer	53
1.8	Error protection	64
1.8.1	Overview of the tools	64
1.8.2	Syntax	67
1.8.3	General information	70
1.8.4	Tool description	73
Annex 1.A	(informative) Audio Interchange Formats	90
1.A.1	Introduction	90
1.A.2	AAC Interchange formats	90
1.A.3	Syntax	90
1.A.4	Semantic	93
Annex 1.B	(informative) Error protection tool	95
1.B.1	Example of out-of-band information	95
1.B.2	Example of error concealment	98
1.B.3	Example of EP tool setting and error concealment for HVXC	103
Annex 1.C	(informative) Patent statements	119

Subpart 1: Main

1.1 Scope

ISO/IEC 14496-3 (MPEG-4 Audio) is a new kind of audio International Standard that integrates many different types of audio coding: natural sound with synthetic sound, low bitrate delivery with high-quality delivery, speech with music, complex soundtracks with simple ones, and traditional content with interactive and virtual-reality content. By standardizing individually sophisticated coding tools as well as a novel, flexible framework for audio synchronization, mixing, and downloaded post-production, the developers of the MPEG-4 Audio standard have created new technology for a new, interactive world of digital audio.

MPEG-4, unlike previous audio standards created by ISO/IEC and other groups, does not target a single application such as real-time telephony or high-quality audio compression. Rather, MPEG-4 Audio is a standard that applies to every application requiring the use of advanced sound compression, synthesis, manipulation, or playback. The subparts that follow specify the state-of-the-art coding tools in several domains; however, MPEG-4 Audio is more than just the sum of its parts. As the tools described here are integrated with the rest of the MPEG-4 International Standard, exciting new possibilities for object-based audio coding, interactive presentation, dynamic soundtracks, and other sorts of new media, are enabled.

Since a single set of tools is used to cover the needs of a broad range of applications, *interoperability* is a natural feature of systems that depend on the MPEG-4 Audio International Standard. A system that uses a particular coder — for example, a real-time voice communication system making use of the MPEG-4 speech coding toolset — can easily share data and development tools with other systems, even in different domains, that use the same tool — for example, a voicemail indexing and retrieval system making use of MPEG-4 speech coding. A multimedia terminal that can decode the Natural Audio Profile of MPEG-4 Audio has audio capabilities that cover the entire spectrum of audio functionality available today and into the future.

1.2 Normative references

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

ISO/IEC 11172-3:1993, *Information technology — Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s — Part 3: Audio*

ITU-T Rec.H.222.0(1995) | ISO/IEC 13818-1:2000, *Information technology — Generic coding of moving pictures and associated audio information — Part 1: Systems*

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

ISO/IEC 13818-7:2004, *Information technology — Generic coding of moving pictures and associated audio information — Part 7: Advanced Audio Coding (AAC)*

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

ISO/IEC 14496-11, *Information technology — Coding of audio-visual objects — Part 11: Scene description and application engine*

ITU-T Recommendation H.223/Annex C: MULTIPLEXING PROTOCOL FOR LOW BITRATE MULTIMEDIA COMMUNICATION OVER HIGHLY ERROR_PRONE CHANNELS, April 1998.

(c) 1996 MIDI Manufacturers Association, *The Complete MIDI 1.0 Detailed Specification* v. 96.2

(c) 1998 MIDI Manufacturers Association, *The MIDI Downloadable Sounds Specification* v. 98.2

1.3 Terms and definitions

- 1.3.1. **AAC:** Advanced Audio Coding.
- 1.3.2. **AAC program:** A set of main audio channels, coupling channel, lfe channel and associated data streams intended to be decoded and played back simultaneously. A program may be defined by default, or specifically by a `program_config_element()`. A given `single_channel_element()`, `channel_pair_element()`, `coupling_channel_element()`, `lfe_channel_element()` or `data_stream_element()` may accompany one or more programs in any given stream.
- 1.3.3. **Audio access unit:** An individually accessible portion of audio data within an elementary stream.
- 1.3.4. **Audio composition unit:** An individually accessible portion of the output that an audio decoder produces from audio access units.
- 1.3.5. **Absolute time:** The time at which sound corresponding to a particular event is really created; time in the real-world. Contrast score time.
- 1.3.6. **Actual parameter:** The expression which, upon evaluation, is passed to an opcode as a parameter value.
- 1.3.7. **A-cycle:** See audio cycle.
- 1.3.8. **Adaptive codebook:** An approach to encode the long-term periodicity of the signal. The entries of the codebook consists of overlapping segments of past excitations.
- 1.3.9. **Alias:** Mirrored spectral component resulting from sampling.
- 1.3.10. **Analysis filterbank:** Filterbank in the encoder that transforms a broadband PCM audio signal into a set of spectral coefficients.
- 1.3.11. **Ancillary data:** Part of the bitstream payload that might be used for transmission of ancillary data.
- 1.3.12. **API:** Application Programming Interface.
- 1.3.13. **A-rate:** See audio rate.
- 1.3.14. **Asig:** The lexical tag indicating an a-rate variable.
- 1.3.15. **Audio buffer:** A buffer in the system target decoder (see ISO/IEC 13818-1) for storage of compressed audio data.
- 1.3.16. **Audio cycle:** The sequence of processing which computes new values for all a-rate expressions in a particular code block.
- 1.3.17. **Audio rate:** The rate type associated with a variable, expression or statement which may generate new values as often as the sampling rate.
- 1.3.18. **Audio sample:** A short snippet or clip of digitally represented sound. Typically used in wavetable synthesis.
- 1.3.19. **AudioBIFS:** The set of tools specified in ISO/IEC 14496-1 (MPEG-4 Systems) for the composition of audio data in interactive scenes.
- 1.3.20. **Authoring:** In Structured Audio, the combined processes of creatively composing music and sound control scripts, creating instruments which generate and alter sound, and encoding the instruments, control scripts, and audio samples in MPEG-4 Structured Audio format.

- 1.3.21. **Backus-Naur Format:** (BNF) A format for describing the syntax of programming languages, used here to specify the SAOL and SASL syntax.
- 1.3.22. **Backward compatibility:** A newer coding standard is backward compatible with an older coding standard if decoders designed to operate with the older coding standard are able to continue to operate by decoding all or part of a bitstream payload produced according to the newer coding standard.
- 1.3.23. **Bandwidth scalability:** The possibility to change the bandwidth of the signal during transmission.
- 1.3.24. **Bank:** A set of samples used together to define a particular sound or class of sounds with wavetable synthesis.
- 1.3.25. **Bark:** The Bark is the standard unit corresponding to one critical band width of human hearing.
- 1.3.26. **Beat:** The unit in which score time is measured.
- 1.3.27. **Bitrate:** The rate at which the compressed stream is delivered to the input of a decoder.
- 1.3.28. **Bitrate scalability:** The possibility to transmit a subset of the bitstream payload and still decode the bitstream payload with the same decoder.
- 1.3.29. **Bitstream payload verifier:** A process by which it is possible to test and verify that all the requirements specified in ISO/IEC 14496-3 are met by the bitstream payload.
- 1.3.30. **Bitstream; stream:** An ordered series of bits that forms the coded representation of the data.
- 1.3.31. **Block companding:** Normalizing of the digital representation of an audio signal within a certain time period.
- 1.3.32. **BNF:** See Backus-Naur Format.
- 1.3.33. **BSAC:** Bit Sliced Arithmetic Coding
- 1.3.34. **Bus:** An area in memory which is used to pass the output of one instrument into the input of another.
- 1.3.35. **Byte:** Sequence of 8-bits.
- 1.3.36. **Byte aligned:** A bit in a data function is byte-aligned if its position is a multiple of 8-bits from the first bit of this data function.
- 1.3.37. **CELP:** Code Excited Linear Prediction.
- 1.3.38. **Center channel:** An audio presentation channel used to stabilize the central component of the frontal stereo image.
- 1.3.39. **Channel:** A sequence of data representing an audio signal intended to be reproduced at one listening position.
- 1.3.40. **Coded audio bitstream:** A coded representation of an audio signal.
- 1.3.41. **Coded representation:** A data element as represented in its encoded form.
- 1.3.42. **Composition (compositing):** Using a scene description to mix and combine several separate audio tracks into a single presentation.
- 1.3.43. **Compression:** Reduction in the number of bits used to represent an item of data.
- 1.3.44. **Constant bitrate:** Operation where the bitrate is constant from start to finish of the coded bitstream.

- 1.3.45. **Context:** See state space.
- 1.3.46. **Control:** An instruction used to describe how to use a particular synthesis method to produce sound.
- EXAMPLES:
"Using the piano instrument, play middle C at medium volume for 2 seconds."
"Glissando the violin instrument up to middle C."
"Turn off the reverberation for 8 seconds."
- 1.3.47. **Control cycle:** The sequence of processing which computes new values for all control-rate expressions in a particular code block.
- 1.3.48. **Control period:** The length of time (typically measured in audio samples) corresponding to the control rate.
- 1.3.49. **Control rate:** (1) The rate at which instantiation and termination of instruments, parametric control of running instrument instances, sharing of global variables, and other non-sample-by-sample computation occurs in a particular orchestra. (2) The rate type of variables, expressions, and statements that can generate new values as often as the control rate.
- 1.3.50. **Core coder:** The term core coder is used to denote a base layer coder in certain scalability configurations. A core coder does not code the spectral samples of the MDCT filterbank of the subsequent AAC coding layers, but operates on a time domain signal. The output of the core decoder has to be up-sampled and transformed into the spectral domain, before it can be combined with the output of the AAC coding layers. Within the MPEG-4 Audio standard only the MPEG-4 CELP coder is a valid core coder. However, in principal, also another AAC coding layer, operating at a lower sampling rate, could be used on the time domain signal, and then combined with the other coding layer in exactly the same way as described for the CELP coder, and would therefore be called a core coder.
- 1.3.51. **CRC:** The Cyclic Redundancy Check to verify the correctness of data.
- 1.3.52. **Critical band:** This unit of bandwidth represents the standard unit of bandwidth expressed in human auditory terms, corresponding to a fixed length on the human cochlea. It is approximately equal to 100 Hz at low frequencies and 1/3 octave at higher frequencies, above approximately 700 Hz.
- 1.3.53. **Data element:** An item of data as represented after encoding and before decoding.
- 1.3.54. **Data function:** An encapsulation of data elements forming a logic unit.
- 1.3.55. **Decoded stream:** The decoded reconstruction of a compressed bitstream.
- 1.3.56. **Decoder:** An embodiment of a decoding process.
- 1.3.57. **Decoding (process):** The process that reads an input coded bitstream payload and outputs decoded audio samples.
- 1.3.58. **Demultiplexing:** Splitting one bitstream into several.
- 1.3.59. **DFT:** Discrete Fourier Transform.
- 1.3.60. **Digital storage media; DSM:** A digital storage or transmission device or system.
- 1.3.61. **Dimension conversion:** A method to convert a dimension of a vector by a combination of low pass filtering and linear interpolation.
- 1.3.62. **Discrete cosine transform; DCT:** Either the forward discrete cosine transform or the inverse discrete cosine transform. The DCT is an invertible, discrete orthogonal transformation.
- 1.3.63. **Downmix:** A matrixing of n channels to obtain less than n channels.

- 1.3.64. **DST:** Direct Stream Transfer
- 1.3.65. **Duration:** The amount of time between instantiation and termination of an instrument instance.
- 1.3.66. **Editing:** The process by which one or more coded bitstreams are manipulated to produce a new coded bitstream. Conforming edited bitstreams must meet the requirements defined in part 3 of ISO/IEC 14496.
- 1.3.67. **Elementary stream (ES):** A sequence of data that originates from a single producer in the transmitting MPEG-4 Terminal and terminates at a single recipient, e.g. an AVObject or a Control Entity in the receiving MPEG-4 Terminal. It flows through one FlexMux Channel.
- 1.3.68. **Encoder:** An embodiment of an encoding process.
- 1.3.69. **Encoding (process):** A process, not specified in ISO/IEC 14496, that reads a stream of input audio samples and produces a valid coded bitstream payload as defined in part 3 of ISO/IEC 14496.
- 1.3.70. **Enhancement layer(s):** The part(s) of the bitstream payload that is possible to drop in a transmission and still decode the bitstream payload.
- 1.3.71. **Entropy coding:** Variable length lossless coding of the digital representation of a signal to reduce statistical redundancy.
- 1.3.72. **Envelope:** A loudness-shaping function applied to a sound, or more generally, any function controlling a parametric aspect of a sound
- 1.3.73. **EP:** Error Protection
- 1.3.74. **ER:** Error resilience or Error Resilient (as appropriate)
- 1.3.75. **Event:** One control instruction.
- 1.3.76. **Excitation:** The excitation signal represents the input to the LPC module. The signal consists of contributions that cannot be covered by the LPC model.
- 1.3.77. **Expression:** A mathematical or functional combination of variable values, symbolic constants, and opcode calls.
- 1.3.78. **FFT:** Fast Fourier Transformation. A fast algorithm for performing a discrete Fourier transform (an orthogonal transform).
- 1.3.79. **Filterbank:** A set of band-pass filters covering the entire audio frequency range.
- 1.3.80. **Fine rate control:** The possibility to change the bitrate by, under some circumstances, skipping transmission of the LPC indices.
- 1.3.81. **Fixed codebook:** The fixed codebook contains excitation vectors for the speech synthesis filter. The contents of the codebook are non-adaptive (i.e. fixed).
- 1.3.82. **Flag:** A variable which can take one of only the two values defined in this specification.
- 1.3.83. **Formal parameter:** The syntactic element that gives a name to one of the parameters of an opcode.
- 1.3.84. **Forward compatibility:** A newer coding standard is forward compatible with an older coding standard if decoders designed to operate with the newer coding standard are able to decode bitstream payloads of the older coding standard.
- 1.3.85. **Frame:** A part of the audio signal that corresponds to a certain number of audio PCM samples.
- 1.3.86. **F_s:** Sampling frequency.

- 1.3.87. **FSS:** Frequency Selective Switch. Module which selects one of two input signals independently in each scalefactor band.
- 1.3.88. **Fundamental frequency:** A parameter which represents signal periodicity in frequency domain.
- 1.3.89. **Future wavetable:** A wavetable that is declared but not defined in the SAOL orchestra; its definition must arrive in the bitstream payload before it is used.
- 1.3.90. **Global block:** The section of the orchestra that describes global variables, route and send statements, sequence rules, and global parameters.
- 1.3.91. **Global context:** The state space used to hold values of global variables and wavetables.
- 1.3.92. **Global parameters:** The sampling rate, control rate, and number of input and output channels of audio associated with a particular orchestra.
- 1.3.93. **Global variable:** A variable that can be accessed and/or changed by several different instruments.
- 1.3.94. **Grammar:** A set of rules that describes the set of allowable sequences of lexical elements comprising a particular language.
- 1.3.95. **Guard expression:** The expression standing at the front of an if, while, or else statement that determines whether or how many times a particular block of code is executed.
- 1.3.96. **Hann window:** A time function applied sample-by-sample to a block of audio samples before Fourier transformation.
- 1.3.97. **Harmonic lines:** A set of spectral components having a common fundamental frequency.
- 1.3.98. **Harmonic magnitude:** Magnitude of each harmonic.
- 1.3.99. **Harmonic synthesis:** A method to obtain a periodic excitation from harmonic magnitudes.
- 1.3.100. **Harmonics:** Samples of frequency spectrum at multiples of the fundamental frequency.
- 1.3.101. **HCR:** Huffman codebook reordering
- 1.3.102. **HILN:** Harmonic and Individual Lines plus Noise (parametric audio coding).
- 1.3.103. **Huffman coding:** A specific method for entropy coding.
- 1.3.104. **HVXC:** Harmonic Vector eXcitation Coding (parametric speech coding).
- 1.3.105. **Hybrid filterbank:** A serial combination of subband filterbank and MDCT. Used in MPEG-1 and MPEG-2 Audio.
- 1.3.106. **I-cycle:** See initialisation cycle.
- 1.3.107. **IDCT:** Inverse Discrete Cosine Transform.
- 1.3.108. **Identifier:** A sequence of characters in a textual SAOL program that denotes a symbol.
- 1.3.109. **IFFT:** Inverse Fast Fourier Transform.
- 1.3.110. **IMDCT:** Inverse Modified Discrete Cosine Transform.
- 1.3.111. **Index:** Number indicating the quantized value(s).
- 1.3.112. **Individual line:** A spectral component described by frequency, amplitude and phase.

- 1.3.113. **Informative:** Aspects of a standards document that are provided to assist implementers, but are not required to be implemented in order for a particular system to be compliant to the standard.
- 1.3.114. **Initial phase:** A phase value at the onset of voiced signal in harmonic synthesis.
- 1.3.115. **Initialisation cycle:** See initialisation pass.
- 1.3.116. **Initialisation pass:** The sequence of processing that computes new values for each i-rate expression in a particular code block.
- 1.3.117. **Initialisation rate:** The rate type of variables, expressions, and statements that are set once at instrument instantiation and then do not change.
- 1.3.118. **Instance:** See instrument instantiation.
- 1.3.119. **Instantiation:** The process of creating a new instrument instantiation based on an event in the score or statement in the orchestra.
- 1.3.120. **Instrument:** An algorithm for parametric sound synthesis, described using SAOL. An instrument encapsulates all of the algorithms needed for one sound-generation element to be controlled with a score.
- NOTE - An MPEG-4 Structured Audio instrument does not necessarily correspond to a real-world instrument. A single instrument might be used to represent an entire violin section, or an ambient sound such as the wind. On the other hand, a single real-world instrument that produces many different timbres over its performance range might be represented using several SAOL instruments.
- 1.3.121. **Instrument instantiation:** The state space created as the result of executing a note-creation event with respect to a SAOL orchestra.
- 1.3.122. **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.
- 1.3.123. **Interframe prediction:** A method to predict a value in the current frame from values in the previous frames. Interframe prediction is used in VQ of LSP.
- 1.3.124. **International Phonetic Alphabet; IPA :** The worldwide agreed symbol set to represent various phonemes appearing in human speech.
- 1.3.125. **I-pass:** See initialisation pass.
- 1.3.126. **IPQF:** inverse polyphase quadrature filter
- 1.3.127. **I-rate:** See initialisation rate.
- 1.3.128. **Ivar:** The lexical tag indicating an i-rate variable.
- 1.3.129. **Joint stereo coding:** Any method that exploits stereophonic irrelevance or stereophonic redundancy.
- 1.3.130. **Joint stereo mode:** A mode of the audio coding algorithm using joint stereo coding.
- 1.3.131. **K-cycle:** See control cycle.
- 1.3.132. **K-rate:** See control rate.
- 1.3.133. **Ksig:** The lexical tag indicating a k-rate variable.
- 1.3.134. **Lexical element:** See token.

- 1.3.135. **Lip shape pattern** : A number that specifies a particular pattern of the preclassified lip shape.
- 1.3.136. **Lip synchronization** : A functionality that synchronizes speech with corresponding lip shapes.
- 1.3.137. **Looping**: A typical method of wavetable synthesis. Loop points in an audio sample are located and the sound between those endpoints is played repeatedly while being simultaneously modified by envelopes, modulators, etc.
- 1.3.138. **Low frequency enhancement (LFE) channel**: A limited bandwidth channel for low frequency audio effects in a multichannel system.
- 1.3.139. **LPC**: Linear Predictive Coding.
- 1.3.140. **LPC residual signal**: A signal filtered by the LPC inverse filter, which has a flattened frequency spectrum.
- 1.3.141. **LPC synthesis filter**: An IIR filter whose coefficients are LPC coefficients. This filter models the time varying vocal tract.
- 1.3.142. **LSP**: Line Spectral Pairs.
- 1.3.143. **LTP**: Long Term Prediction.
- 1.3.144. **M/S stereo**: A method of removing imaging artifacts as well as exploiting stereo irrelevance or redundancy in stereophonic audio programs based on coding the sum and difference signal instead of the left and right channels.
- 1.3.145. **Main audio channels**: All single_channel_elements or channel_pair_elements in one program.
- 1.3.146. **Mapping**: Conversion of an audio signal from time to frequency domain by subband filtering and/or by MDCT.
- 1.3.147. **Masking**: A property of the human auditory system by which an audio signal cannot be perceived in the presence of another audio signal.
- 1.3.148. **Masking threshold**: A function in frequency and time below which an audio signal cannot be perceived by the human auditory system.
- 1.3.149. **MIDI**: The Musical Instrument Digital Interface standards. Certain aspects of the MPEG-4 Structured Audio tools provide interoperability with MIDI standards.
- 1.3.150. **Mixed voiced frame**: A speech segment which has both voiced and unvoiced components.
- 1.3.151. **Modified discrete cosine transform (MDCT)**: A transform which has the property of time domain aliasing cancellation.
- 1.3.152. **Moving picture dubbing** : A functionality that assigns synthetic speech to the corresponding moving picture while utilizing lip shape pattern information for synchronization.
- 1.3.153. **MPE**: Multi Pulse Excitation.
- 1.3.154. **MPEG-4 Audio Text-to-Speech Decoder** : A device that produces synthesized speech by utilizing the M-TTS bitstream payload while supporting all the M-TTS functionalities such as speech synthesis for FA and MP dubbing.
- 1.3.155. **M-TTS sentence** : This defines the information such as prosody, gender, and age for only the corresponding sentence to be synthesized.

- 1.3.156. **M-TTS sequence** : This defines the control information which affects all M-TTS sentences that follow this M-TTS sequence.
- 1.3.157. **Multichannel**: A combination of audio channels used to create a spatial sound field.
- 1.3.158. **Multilingual**: A presentation of dialogue in more than one language.
- 1.3.159. **Multiplexing**: Combining several bitstream payloads into one.
- 1.3.160. **Natural Sound**: A sound created through recording from a real acoustic space. Contrasted with synthetic sound.
- 1.3.161. **NCC**: Number of Considered Channels. In case of AAC, it is the number of channels represented by the elements SCE, independently switched CCE and CPE, i.e. once the number of SCEs plus once the number of independently switched CCEs plus twice the number of CPEs. With respect to the naming conventions of the MPEG-AAC decoders and payloads, $NCC=A+I$. This number is used to derive the required decoder input buffer size (see subpart 4, subclause 4.5.3.1). In case of other codecs, it is the total number of channels.
- 1.3.162. **Noise component**: A signal component modeled as noise.
- 1.3.163. **Non-tonal component**: A noise-like component of an audio signal.
- 1.3.164. **Normative**: Those aspects of a standard that must be implemented in order for a particular system to be compliant to the standard.
- 1.3.165. **Nyquist sampling**: Sampling at or above twice the maximum bandwidth of a signal.
- 1.3.166. **OD**: Object Descriptor.
- 1.3.167. **Opcode**: A parametric signal-processing function that encapsulates a certain functionality so that it may be used by several instruments.
- 1.3.168. **Orchestra**: The set of sound-generation and sound-processing algorithms included in an MPEG-4 bitstream payload. Includes instruments, opcodes, routing, and global parameters.
- 1.3.169. **Orchestra cycle**: A complete pass through the orchestra, during which new instrument instantiations are created, expired ones are terminated, each instance receives one k-cycle and one control period worth of a-cycles, and output is produced.
- 1.3.170. **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.
- 1.3.171. **Parameter**: A variable within the syntax of this specification which may take one of a range of values. A variable which can take one of only two values is a flag or indicator and not a parameter.
- 1.3.172. **Parameter fields**: The names given to the parameters to an instrument.
- 1.3.173. **Parser**: Functional stage of a decoder which extracts from a coded bitstream payload a series of bits representing coded elements.
- 1.3.174. **P-fields**: See parameter fields.
- 1.3.175. **Phoneme/bookmark-to-FAP converter** : A device that converts phoneme and bookmark information to FAPs.
- 1.3.176. **Pi**: The constant $\pi = 3.14159...$

- 1.3.177. **Pitch:** A parameter which represents signal periodicity in the time domain. It is expressed in terms of the number of samples.
- 1.3.178. **Pitch control:** A functionality to control the pitch of the synthesized speech signal without changing its speed.
- 1.3.179. **PNS:** Perceptual Noise Substitution.
- 1.3.180. **Polyphase filterbank:** A set of equal bandwidth filters with special phase interrelationships, allowing an efficient implementation of the filterbank.
- 1.3.181. **Postfilter:** A filter to enhance the perceptual quality of the synthesized speech signal.
- 1.3.182. **PQF:** polyphase quadrature filter
- 1.3.183. **Prediction:** The use of a predictor to provide an estimate of the sample value or data element currently being decoded.
- 1.3.184. **Prediction error:** The difference between the actual value of a sample or data element and its predictor.
- 1.3.185. **Predictor:** A linear combination of previously decoded sample values or data elements.
- 1.3.186. **Presentation channel:** An audio channel at the output of the decoder.
- 1.3.187. **Production rule:** In Backus-Naur Form grammars, a rule that describes how one syntactic element may be expressed in terms of other lexical and syntactic elements.
- 1.3.188. **PSNR:** Peak Signal to Noise Ratio.
- 1.3.189. **Psychoacoustic model:** A mathematical model of the masking behaviour of the human auditory system.
- 1.3.190. **Random access:** The process of beginning to read and decode the coded stream at an arbitrary point.
- 1.3.191. **Rate semantics:** The set of rules describing how rate types are assigned to variables, expressions, statements, and opcodes, and the normative restrictions that apply to a bitstream regarding combining these elements based on their rate types.
- 1.3.192. **Rate type:** The “speed of execution” associated with a particular variable, expression, statement, or opcode.
- 1.3.193. **Rate-mismatch error:** The condition that results when the rate semantics rules are violated in a particular SAOL construction. A type of syntax error.
- 1.3.194. **Reserved:** The term "reserved" when used in the subclauses defining the coded bitstream payload indicates that the value may be used in the future for ISO/IEC defined extensions.
- 1.3.195. **Route statement:** A statement in the global block that describes how to place the output of a certain set of instruments onto a bus.
- 1.3.196. **RPE:** Regular Pulse Excitation.
- 1.3.197. **Run-time error:** The condition that results from improper calculations or memory accesses during execution of a SAOL orchestra.
- 1.3.198. **RVLC:** Reversible Variable Length Coding
- 1.3.199. **Sample:** See Audio sample.

- 1.3.200. **Sample Bank Format:** A component format of MPEG-4 Structured Audio that allows the description of a set of samples for use in wavetable synthesis and processing methods to apply to them.
- 1.3.201. **Sampling Frequency (Fs):** Defines the rate in Hertz which is used to digitize an audio signal during the sampling process.
- 1.3.202. **SAOL:** The Structured Audio Orchestra Language, pronounced like the English word "sail". SAOL is a digital-signal processing language that allows for the description of arbitrary synthesis and control algorithms as part of the content bitstream payload.
- 1.3.203. **SAOL orchestra:** See orchestra.
- 1.3.204. **SASBF:** The MPEG-4 Structured Audio Sample Bank Format, an efficient format for the transmission of blocks of wavetable (sample data) compatible with the MIDI method for the same.
- 1.3.205. **SASL:** The Structured Audio Score Language. SASL is a simple format that allows for powerful and flexible control of music and sound synthesis.
- 1.3.206. **SBA:** Segmented Binary Arithmetic Coding which is the error resilient tool for BSAC.
- 1.3.207. **SBR:** Spectral Band Replication.
- 1.3.208. **Scalefactor:** Factor by which a set of values is scaled before quantization.
- 1.3.209. **Scalefactor band:** A set of spectral coefficients which are scaled by one scalefactor.
- 1.3.210. **Scalefactor index:** A numerical code for a scalefactor.
- 1.3.211. **Scheduler:** The component of MPEG-4 Structured Audio that describes the mapping from control instructions to sound synthesis using the specified synthesis techniques. The scheduler description provides normative bounds on event-dispatch times and responses.
- 1.3.212. **Scope :** The code within which access to a particular variable name is allowed.
- 1.3.213. **Score:** A description in some format of the sequence of control parameters needed to generate a desired music composition or sound scene. In MPEG-4 Structured Audio, scores are described in SASL and/or MIDI.
- 1.3.214. **Score time:** The time at which an event happens in the score, measured in beats. Score time is mapped to absolute time by the current tempo.
- 1.3.215. **Semantics:** The rules describing what a particular instruction or bitstream payload element should do. Most aspects of bitstream payload and SAOL semantics are normative in MPEG-4.
- 1.3.216. **Send statement:** A statement in the global block that describes how to pass a bus on to an effect instrument for post-processing.
- 1.3.217. **Sequence rules:** The set of rules, both default and explicit, given in the global block that define in what order to execute instrument instantiations during an orchestra cycle.
- 1.3.218. **SIAQ:** Scalable Inverse AAC Quantization Module.
- 1.3.219. **Side information:** Information in the bitstream payload necessary for controlling the decoder.
- 1.3.220. **Signal variable:** A unit of memory, labelled with a name, that holds intermediate processing results. Each signal variable in MPEG-4 Structured Audio is instantaneously representable by a 32-bit floating point value.

- 1.3.221. **Sinusoidal synthesis:** A method to obtain a time domain waveform by a sum of amplitude modulated sinusoidal waveforms.
- 1.3.222. **Spatialisation:** The process of creating special sounds that a listener perceives as emanating from a particular direction.
- 1.3.223. **Spectral coefficients:** Discrete frequency domain data output from the analysis filterbank.
- 1.3.224. **Spectral envelope:** A set of harmonic magnitudes.
- 1.3.225. **Speed control:** A functionality to control the speed of the synthesized speech signal without changing its pitch or phonemes.
- 1.3.226. **Spreading function:** A function that describes the frequency spread of masking effects.
- 1.3.227. **SQ:** Scalar Quantization.
- 1.3.228. **SSC:** SinuSoidal Coding, parametric coder for high quality audio.
- 1.3.229. **State space:** A set of variable-value associations that define the current computational state of an instrument instantiation or opcode call. All the “current values” of the variables in an instrument or opcode call.
- 1.3.230. **Statement:** “One line” of a SAOL orchestra.
- 1.3.231. **Stereo-irrelevant:** A portion of a stereophonic audio signal which does not contribute to spatial perception.
- 1.3.232. **Structured audio:** Sound-description methods that make use of high-level models of sound generation and control. Typically involving synthesis description, structured audio techniques allow for ultra-low bitrate description of complex, high-quality sounds.
- 1.3.233. **Stuffing (bits); stuffing (bytes):** Code-words that may be inserted at particular locations in the coded bitstream payload that are discarded in the decoding process. Their purpose is to increase the bitrate of the stream which would otherwise be lower than the desired bitrate.
- 1.3.234. **Surround channel:** An audio presentation channel added to the front channels (L and R or L, R, and C) to enhance the spatial perception.
- 1.3.235. **Symbol:** A sequence of characters in a SAOL program, or a symbol token in a MPEG-4 Structured Audio bitstream payload, that represents a variable name, instrument name, opcode name, table name, bus name, etc.
- 1.3.236. **Symbol table:** In an MPEG-4 Structured Audio bitstream payload, a sequence of data that allows the tokenised representation of SAOL and SASL code to be converted back to a readable textual representation. The symbol table is an optional component.
- 1.3.237. **Symbolic constant:** A floating-point value explicitly represented as a sequence of characters in a textual SAOL orchestra, or as a token in a bitstream payload.
- 1.3.238. **Syncword:** A code embedded in audio transport streams that identifies the start of a transport frame.
- 1.3.239. **Syntax:** The rules describing what a particular instruction or bitstream payload element should look like. All aspects of bitstream payload and SAOL syntax are normative in MPEG-4.
- 1.3.240. **Syntax error:** The condition that results when a bitstream payload element does not comply with its governing rules of syntax.
- 1.3.241. **Synthesis:** The process of creating sound based on algorithmic descriptions.

- 1.3.242. **Synthesis filterbank:** Filterbank in the decoder that reconstructs a PCM audio signal from subband samples.
- 1.3.243. **Synthetic Sound:** Sound created through synthesis.
- 1.3.244. **Tempo:** The scaling parameter that specifies the relationship between score time and absolute time. A tempo of 60 beats per minute means that the score time measured in beats is equivalent to the absolute time measured in seconds; higher numbers correspond to faster tempi, so that 120 beats per minute is twice as fast.
- 1.3.245. **Terminal:** The “client side” of an MPEG transaction; whatever hardware and software are necessary in a particular implementation to allow the capabilities described in this document.
- 1.3.246. **Termination:** The process of destroying an instrument instantiation when it is no longer needed.
- 1.3.247. **Text-to-speech synthesizer :** A device producing synthesized speech according to the input sentence character strings.
- 1.3.248. **Timbre:** The combined features of a sound that allow a listener to recognise such aspects as the type of instrument, manner of performance, manner of sound generation, etc. Those aspects of sound that distinguish sounds equivalent in pitch and loudness.
- 1.3.249. **TNS:** Temporal Noise Shaping
- 1.3.250. **Token:** A lexical element of a SAOL orchestra a keyword, punctuation mark, symbol name, or symbolic constant.
- 1.3.251. **Tokenisation:** The process of converting a orchestra in textual SAOL format into a bitstream payload representation consisting of a stream of tokens.
- 1.3.252. **Tonal component:** A sinusoid-like component of an audio signal.
- 1.3.253. **Trick mode :** A set of functions that enables stop, play, forward, and backward operations for users.
- 1.3.254. **TTSI:** Text to Speech Interface.
- 1.3.255. **TwinVQ:** Transform domain Weighted Interleave Vector Quantization.
- 1.3.256. **Unvoiced frame:** Frame containing unvoiced speech which looks like random noise with no periodicity.
- 1.3.257. **V/UV decision:** Decision whether the current frame is voiced or unvoiced or mixed voiced.
- 1.3.258. **Variable:** See signal variable.
- 1.3.259. **Variable bitrate:** Operation where the bitrate varies with time during the decoding of a coded stream.
- 1.3.260. **Variable length code (VLC):** A code word assigned by variable length encoder (See variable length coding).
- 1.3.261. **Variable length coding:** A reversible procedure for coding that assigns shorter code-words to frequent symbols and longer code-words to less frequent symbols.
- 1.3.262. **Variable length decoder:** A procedure to obtain the symbols encoded with a variable length coding technique.
- 1.3.263. **Variable length encoder:** A procedure to assign variable length codewords to symbols.
- 1.3.264. **VCB11:** Virtual Codebooks for codebook 11.

- 1.3.265. **Vector quantizer:** Tool that quantizes several values to one index.
- 1.3.266. **Virtual codebook:** If several codebook values refer to one and the same physical codebook, these values are called virtual codebooks.
- 1.3.267. **Voiced frame:** A voiced speech segment is known by its relatively high energy content, but more importantly it contains periodicity which is called the pitch of the voiced speech.
- 1.3.268. **VQ:** Vector Quantization.
- 1.3.269. **VXC:** Vector eXcitation Coding. It is also called CELP (Coded Excitation Linear Prediction). In HVXC, no adaptive codebook is used.
- 1.3.270. **Wavetable synthesis:** A synthesis method in which sound is created by simple manipulation of audio samples, such as looping, pitch-shifting, enveloping, etc.
- 1.3.271. **White Gaussian noise:** A noise sequence which has a Gaussian distribution.
- 1.3.272. **Width:** The number of channels of data that an expression represents.

1.4 Symbols and abbreviations

The mathematical operators used in this part of ISO/IEC 14496 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.

1.4.1 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 towards $-\infty$.
	Absolute value. $ x = x$ when $x > 0$ $ x = 0$ when $x == 0$ $ x = -x$ when $x < 0$
%	Modulus operator. Defined only for positive numbers.

Sign() Sign.
 $\text{Sign}(x) = 1$ when $x > 0$
 $\text{Sign}(x) = 0$ when $x == 0$
 $\text{Sign}(x) = -1$ when $x < 0$

INT () Truncation to integer operator. Returns the integer part of the real-valued argument.

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.

$\sqrt{\quad}$ Square root.

log₁₀ Logarithm to base ten.

log_e Logarithm to base e.

log₂ Logarithm to base 2.

ceil() Ceiling operator. Returns the smallest integer that is greater than or equal to the real-valued argument.

1.4.2 Logical operators

|| Logical OR.

&& Logical AND.

! Logical NOT

1.4.3 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.

1.4.4 Bitwise operators

A two's complement number representation is assumed where the bitwise operators are used.

& AND

| OR

>> Shift right with sign extension.

<< Shift left with zero fill.

1.4.5 Assignment

= Assignment operator.

1.4.6 Mnemonics

The following mnemonics are defined to describe the different data types used in the coded bitstream payload.

bslbf	Bit string, left bit first, where "left" is the order in which bit strings are written in ISO/IEC 14496. 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.
L, C, R, LS, RS	Left, center, right, left surround and right surround audio signals
rpchof	Remainder polynomial coefficients, highest order first. (Audio)
uimbsf	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.
window	Number of the actual time slot in case of block_type == 2, 0 <= window <= 2. (Audio)

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

1.4.7 Constants

π	3,14159265358...
e	2,71828182845...

1.4.8 Method of describing bitstream payload syntax

The bitstream payload retrieved by the decoder is described in the syntax section of each subpart. Each data item in the bitstream payload is in bold type.

It is described by

- its name
- its length in bits, where "X..Y" indicates that the number of bits is one of the values between X and Y including X and Y. "{X;Y}" means the number of bits is X or Y, depending on the value of other data elements in the bitstream payload.),
- a mnemonic for its type and order of transmission.

Data elements forming a logic unit are encapsulated in data functions.

data_function (); Data function call.

```
data_function ( ) {           Data function entity.
    ...
}
```

The action caused by a decoded data element in a bitstream payload depends on the value of that data element and on data elements previously decoded. The decoding of the data elements and the definition of the state variables used in their decoding are described in the subclauses following the syntax section of each subpart. The following constructs are used to express the conditions when data elements are present, and are in normal type:

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

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

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

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

```
switch (expression) {        If the condition formed by the comparison of expression and const-expr is true, then
    case const-expr:          the data stream continues with the subsequent data elements. An optionally break
        data_element;         statement can be used to immediately leave the switch, data elements beyond a break
        break;                do not occur in the data stream.
    case const-expr:
        data_element;
}
```

```
for (expr1; expr2; expr3)    Expr1 is an expression specifying the initialisation of the loop. Normally it specifies the
{                             initial state of the counter. Expr2 is a condition specifying a test made before each
    data_element              iteration of the loop. The loop terminates when the condition is not true. Expr3 is an
    ...                       expression that is performed at the end of each iteration of the loop, normally it
}                             increments a counter.
```

Note that the most common usage of this construct is as follows:

```

for ( i = 0; i < n; i++) {
    data_element
    ...
}

```

The group of data elements occurs n times. Conditional constructs within the group of data elements may depend on the value of the loop control variable i , which is set to zero for the first occurrence, incremented to one for the second occurrence, and so forth.

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

data_element [] data_element [] is an array of data. The number of data elements is indicated by the context.

data_element [n] data_element [n] is the $n+1$ th element of an array of data.

data_element [m][n] data_element [m][n] is the $m+1, n+1$ th element of a two-dimensional array of data.

data_element [l][m][n] data_element [l][m][n] is the $l+1, m+1, n+1$ th element of a three-dimensional array of data.

data_element [m..n] 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 this implements a satisfactory decoding procedure. In particular, it defines a correct and error-free input bitstream payload. Actual decoders must include a means to deal with incorrect bitstream payloads and to find the start of the described elements.

Definition of nextbits function

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

1.5 Technical overview

1.5.1 MPEG-4 audio object types

1.5.1.1 Audio object type definition

Table 1.1 — Audio Object Type definition based on Tools/Modules

Object Type ID	Audio Object Type																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
0	Null																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					

Object Type ID	Audio Object Type	gain control	block switching	window shapes - standard	window shapes - AAC LD	filterbank - standard	filterbank - SSR	TNS	LTP	intensity	coupling	frequency deomain prediction	PNS	MS	SIAQ	FSS	upsampling filter tool	quantisation&coding - AAC	quantisation&coding – TwinVQ	quantisation&coding - BSAC	AAC ER Tools	ER payload syntax	EP Tool 1)	CELP	Silence Compression	HVXC	HVXC 4kbit/s VR	SA tools	SASBF	MIDI	HILN	TTSI	SBR	Layer-1	Layer-2	Layer-3	SSC (Transient, Sinusoid, Noise)	Parametric stereo	DST	Remark		
7	TwinVQ		X	X		X		X	X																																	
8	CELP																							X																		
9	HVXC																									X																
10	(reserved)																																									
11	(reserved)																																									
12	TTSI																															X										
13	Main synthetic																										X	X	X												3)	
14	Wavetable synthesis																											X	X													4)
15	General MIDI																												X													
16	Algorithmic Synthesis and Audio FX																										X															
17	ER AAC LC		X	X		X		X		X			X	X				X			X	X	X																			
18	(reserved)																																									
19	ER AAC LTP		X	X		X		X	X	X			X	X				X			X	X	X																			5)
20	ER AAC scalable		X	X		X		X		X			X	X	X	X	X				X	X	X																			6)
21	ER TwinVQ		X	X		X		X						X					X			X	X																			
22	ER BSAC		X	X		X		X		X			X	X					X				X																			
23	ER AAC LD				X	X		X	X	X			X	X			X				X	X	X																			
24	ER CELP																					X	X	X	X																	
25	ER HVXC																					X	X			X	X															
26	ER HILN																					X	X								X											
27	ER Parametric																					X	X			X	X				X											
28	SSC																									X	X											X	X			
29	(reserved)																																									
30	(reserved)																																									
31	(escape)																																									
32	Layer-1																																	X								
33	Layer-2																																		X							
34	Layer-3																																			X						
35	DST																																									
36 - 95	(reserved)																																									X

Notes:

- 1) The bit parsing function is mandatory on decoder site. However, the error detection and error correction functions are optional.
- 2) Contains AAC LC.
- 3) Contains Wavetable synthesis and Algorithmic Synthesis and Audio FX.
- 4) Contains General MIDI.
- 5) Contains ER AAC LC.
- 6) The upsampling filter tool is only required in combination with a core coder.

1.5.1.2 Description

1.5.1.2.1 NULL object type

The NULL object provides the possibility to feed raw PCM data directly to the audio compositor. No decoding is involved. However, an audio object descriptor is used to specify the sampling rate and the audio channel configuration.

1.5.1.2.2 AAC - Main object type

The AAC Main object is very similar to the AAC Main Profile that is defined in ISO/IEC 13818-7. However, additionally the PNS tool is available. The restrictions of the AAC Main profile with respect to multiple programs and mixdown elements also apply to the AAC Main object type. The AAC Main object type bitstream payload syntax is compatible with the syntax defined in ISO/IEC 13818-7. All the MPEG-2 AAC multi-channel capabilities are available. A decoder capable to decode a MPEG-4 main object stream can also parse and decode a MPEG-2 AAC raw data stream. On the other hand, although a MPEG-2 AAC coder can parse an MPEG-4 AAC Main bitstream payload, decoding may fail, since PNS might have been used.

1.5.1.2.3 AAC - Low Complexity (LC) object type

The MPEG-4 AAC Low Complexity object type is the counterpart to the MPEG-2 AAC Low Complexity Profile, with exactly the same restrictions as mentioned above for the AAC Main object type.

1.5.1.2.4 AAC - Scalable Sampling Rate (SSR) object type

The MPEG-4 AAC Scalable Sampling Rate object type is the counterpart to the MPEG-2 AAC Scalable Sampling Rate Profile, with exactly the same restrictions as mentioned above for the AAC Main object type.

1.5.1.2.5 AAC - Long Term Predictor (LTP) object type

The MPEG-4 AAC LTP object type is similar to the AAC Main object type. However, a Long Term Predictor replaces the MPEG-2 AAC predictor. The LTP achieves a similar coding gain, but requires significantly lower implementation complexity. The bitstream payload syntax for this object type is very similar to the syntax defined in ISO/IEC 13818-7. An MPEG-2 AAC LC profile bitstream payload can be decoded without restrictions using an MPEG-4 AAC LTP object decoder.

1.5.1.2.6 SBR object type

The SBR Object contains the SBR-Tool and can be combined with the audio object types indicated in Table 1.2.

Table 1.2 — Audio object types that can be combined with the SBR Tool

Audio Object Type	Combination with SBR Tool permitted	Object Type ID
Null		0
AAC main	X	1
AAC LC	X	2
AAC SSR	X	3
AAC LTP	X	4
SBR		5
AAC Scalable	X	6
TwinVQ		7
CELP		8
HVXC		9
(Reserved)		10
(Reserved)		11
TTSI		12
Main synthetic		13
Wavetable synthesis		14
General MIDI		15
Algorithmic Synthesis and Audio FX		16
ER AAC LC	X	17
(Reserved)		18
ER AAC LTP	X	19
ER AAC scalable	X	20
ER TwinVQ		21
ER BSAC		22
ER AAC LD		23
ER CELP		24
ER HVXC		25
ER HILN		26
ER Parametric		27
SSC		28
(Reserved)		29
(Reserved)		30
(Reserved)		31

1.5.1.2.7 AAC Scalable object type

The scalable AAC object uses a different bitstream payload syntax to support bitrate- and bandwidth- scalability. A large number of scalable combinations are available, including combinations with TwinVQ and CELP coder tools. However, only mono, or 2-channel stereo objects are supported.

1.5.1.2.8 TwinVQ object type

The TwinVQ object belongs to the GA coding scheme that quantizes the MDCT coefficients. This coding scheme is based on fixed rate vector quantization instead of Huffman coding in AAC.

Low bit rate mono and stereo audio coding is available. Scalable audio coding schemes are also available in the Scalable Audio Profile combined with AAC scalable object type.

1.5.1.2.9 CELP object type

The CELP object is supported by the CELP speech coding tools, which provide coding at 8 kHz and 16 kHz sampling rate at bit rates in the range 4-24 kbit/s. Additionally, bit rate scalability and bandwidth scalability are

available in order to provide scalable decoding of CELP streams. CELP Object always contains exactly one mono audio signal.

1.5.1.2.10 HVXC object type

The HVXC object is supported by the parametric speech coding (HVXC) tools, which provide fixed bitrate modes (2.0 - 4.0 kbit/s) in a scalable and a non-scalable scheme, a variable bitrate mode (< 2.0 kbit/s) and the functionality of pitch and speed change. Only 8 kHz sampling rate and mono audio channel are supported.

1.5.1.2.11 TTSI object type

The TTSI object is supported by the TTSI tools described in subpart 6. It allows very-low-bitrate phonemic descriptions of speech to be transmitted in the bitstream payload and then synthesized into sound. No particular speech synthesis method is specified in MPEG-4; rather, the TTSI tools specify an *interface* to non-normative synthesis methods. This method has a bit rate ranging 200 ~ 1200 bit/s. The TTSI object also supports synchronization of the synthesized speech with the facial animation defined in ISO/IEC 14496-2.

1.5.1.2.12 Main Synthetic object type

The main synthetic object allows the use of all MPEG-4 Structured Audio tools (described in subpart 5 of this standard). It supports flexible, high-quality algorithmic synthesis using the SAOL music-synthesis language; efficient wavetable synthesis with the SASBF sample-bank format; and enables the use of high-quality mixing and postproduction in the Systems AudioBIFS toolset. Sound can be described at 0 kbit/s (no continuous cost) to 3-4 kbit/s for extremely expressive sounds in the MPEG-4 Structured Audio format.

1.5.1.2.13 Wavetable Synthesis object type

The wavetable synthesis object is supported only by the SASBF format and MIDI tools. It allows the use of simple „sampling synthesis“ in presentations where the quality and flexibility of the full synthesis toolset is not required.

1.5.1.2.14 General Midi object type

The General MIDI object is included only to provide interoperability with existing content. Normative sound quality and decoder behavior are not provided with the General MIDI object.

1.5.1.2.15 Algorithmic Synthesis and Audio FX object type

The Algorithmic Synthesis object provides SAOL-based synthesis capabilities for very low-bitrate terminals. It is also used to support the AudioBIFS **AudioFX** node in content where sound synthesis capability is not needed.

1.5.1.2.16 Error Resilient (ER) AAC Low Complexity (LC) object type

The Error Resilient (ER) MPEG-4 AAC Low Complexity object type is the counterpart to the MPEG-4 AAC Low Complexity object, with additional error resilient functionality.

1.5.1.2.17 Error Resilient (ER) AAC Long Term Predictor (LTP) object type

The Error Resilient (ER) MPEG-4 AAC LTP object type is the counterpart to the MPEG-4 AAC LTP object, with additional error resilient functionality.

1.5.1.2.18 Error Resilient (ER) AAC scalable object type

The Error Resilient (ER) MPEG-4 AAC scalable object type is the counterpart to the MPEG-4 AAC scalable object, with additional error resilient functionality.

1.5.1.2.19 Error Resilient (ER) TwinVQ object type

The Error Resilient (ER) TwinVQ object type is the counterpart to the MPEG-4 TwinVQ object, with additional error resilient functionality.

1.5.1.2.20 Error Resilient (ER) BSAC object type

The ER BSAC object is supported by the fine grain scalability tool (BSAC: Bit-Sliced Arithmetic Coding). It provides error resilience as well as fine step scalability in the MPEG-4 General Audio (GA) coder. It is used in combination with the AAC coding tools and replaces the noiseless coding and the bitstream payload formatting of MPEG-4 AAC coder. A large number of scalable layers are available, providing 1 kbit/s/ch enhancement layer, i.e. 2 kbit/s steps for a stereo signal.

1.5.1.2.21 Error Resilient (ER) AAC LD object type

The AAC LD object is supported by the low delay AAC coding tool. It also permits combinations with the PNS tool and the LTP tool. AAC LD object provides the ability to extend the usage of generic low bitrate audio coding to applications requiring a very low delay of the encoding / decoding chain (e.g. full-duplex real-time communications).

1.5.1.2.22 Error Resilient (ER) CELP object type

The ER CELP object is supported by silence compression and ER tools. It provides the ability to reduce the average bitrate thanks to a lower-bitrate compression for silence, with additional error resilient functionality.

1.5.1.2.23 Error Resilient (ER) HVXC object type

The ER HVXC object is supported by the parametric speech coding (HVXC) tools, which provide fixed bitrate modes (2.0-4.0 kbit/s) and variable bitrate modes (< 2.0 kbit/s and < 4.0 kbit/s) both in a scalable and a non-scalable scheme, and the functionality of pitch and speed change. The syntax to be used with the EP-Tool, and the error concealment functionality are supported for the use for error-prone channels. Only 8 kHz sampling rate and mono audio channel are supported.

1.5.1.2.24 Error Resilient (ER) HILN object type

The ER HILN object is supported by the parametric audio coding tools (HILN: Harmonic and Individual Lines plus Noise) which provide coding of general audio signals at very low bitrates ranging from below 4 kbit/s to above 16 kbit/s. Bitrate scalability and the functionality of speed and pitch change are available. The ER HILN object supports mono audio objects at a wide range of sampling rates.

1.5.1.2.25 Error Resilient (ER) Parametric object type

The ER Parametric object is supported by the parametric audio coding and speech coding tools HILN and HVXC. This integrated parametric coder combines the functionalities of the ER HILN and the ER HVXC objects. Only 8 kHz sampling rate and mono audio channel are supported.

1.5.1.2.26 SSC Audio object type

The SSC (SinuSoidal Coding) object combines the SSC parametric coding tools: transients, sinusoids, noise and parametric stereo. Only 44.1kHz and coding of mono, dual mono and (parametric) stereo are supported.

1.5.1.2.27 Layer-1 Audio object type

The Layer-1 object is the counterpart of the audio coding scheme Layer-1 specified in ISO/IEC 11172-3 and 13818-3.

1.5.1.2.28 Layer-2 Audio object type

The Layer-2 object is the counterpart of the audio coding scheme Layer-2 specified in ISO/IEC 11172-3 and ISO/IEC 13818-3.

1.5.1.2.29 Layer-3 Audio object type

The Layer-3 object is very similar to the audio coding scheme Layer-3 specified in ISO/IEC 11172-3 and ISO/IEC 13818-3. However, the use of Layer 3 encoded data as defined in ISO/IEC 13818-3 is limited to the "Lower

Sampling Frequencies" case, i.e. the Layer 3 multi-channel syntax defined in ISO/IEC 13818-3 is not permitted in this scope. Furthermore, additional sampling rates have been specified.

1.5.2 Audio profiles and levels

1.5.2.1 Profiles

The following Audio Profiles have been defined:

1. The **Speech Audio Profile** provides a parametric speech coder, a CELP speech coder and a Text-To-Speech interface.
2. The **Synthetic Audio Profile** provides the capability to generate sound and speech at very low bitrates.
3. The **Scalable Audio Profile**, a superset of the Speech Audio Profile, is suitable for scalable coding of speech and music, for transmission methods such as Internet and Digital Broadcasting.
4. The **Main Audio Profile** is a superset of the scalable profile, the speech profile and the synthesis profile, containing tools for natural and synthetic audio.
5. The **High Quality Audio Profile** contains the CELP speech coder and the Low Complexity AAC coder including Long Term Prediction. Scalable coding can be performed by the AAC Scalable object type. Optionally, the new error resilient (ER) bitstream payload syntax may be used.
6. The **Low Delay Audio Profile** contains the HVXC and CELP speech coders (optionally using the ER bitstream payload syntax), the low-delay AAC coder and the Text-to-Speech interface TTSI.
7. The **Natural Audio Profile** contains all natural audio coding tools available in MPEG-4.
8. The **Mobile Audio Internetworking Profile** contains the low-delay and scalable AAC object types including TwinVQ and BSAC. This profile is intended to extend communication applications using non-MPEG speech coding algorithms with high quality audio coding capabilities.
9. The **AAC Profile** contains the audio object type 2 (AAC-LC).
10. The **High Efficiency AAC Profile** contains the audio object types 5 (SBR) and 2 (AAC LC). The High Efficiency AAC Profile is a superset of the AAC Profile.

Table 1.3 — Audio Profiles definition

Object Type ID	Audio Object Type	Speech Audio Profile	Synthetic Audio Profile	Scalable Audio Profile	Main Audio Profile	High Quality Audio Profile	Low Delay Audio Profile	Natural Audio Profile	Mobile Audio Internetworking Profile	AAC Profile	High Efficiency AAC Profile
0	Null										
1	AAC main				X			X			
2	AAC LC			X	X	X		X		X	X
3	AAC SSR				X			X			
4	AAC LTP			X	X	X		X			
5	SBR										X
6	AAC Scalable			X	X	X		X			
7	TwinVQ			X	X			X			
8	CELP	X		X	X	X	X	X			
9	HVXC	X		X	X		X	X			
10	(reserved)										
11	(reserved)										
12	TTSI	X	X	X	X		X	X			
13	Main synthetic		X		X						
14	WaveTable synthesis		(Subset of Main Synthetic)		(Subset of Main Synthetic)						

15	General MIDI		(Subset of Main Synthetic)		(Subset of Main Synthetic)						
16	Algorithmic Synthesis and Audio FX		(Subset of Main Synthetic)		(Subset of Main Synthetic)						
17	ER AAC LC					X		X	X		
18	(reserved)										
19	ER AAC LTP					X		X			
20	ER AAC Scalable					X		X	X		
21	ER TwinVQ							X	X		
22	ER BSAC							X	X		
23	ER AAC LD						X	X	X		
24	ER CELP					X	X	X			
25	ER HVXC						X	X			
26	ER HILN							X			
27	ER Parametric							X			
28	SSC										
29	(reserved)										
30	(reserved)										
31	(escape)										
32	Layer-1										
33	Layer-2										
34	Layer-3										
35	DST										

In addition to the profile descriptions given above it is stated that AAC Scalable objects using wide-band CELP core layer (with or without ER bitstream payload syntax) are **not** part of any Audio Profile.

1.5.2.2 Complexity units

Complexity units are defined to give an approximation of the decoder complexity in terms of processing power and RAM usage required for processing MPEG-4 Audio bitstream payloads in dependence of specific parameters.

The approximated processing power is given in "Processor Complexity Units" (PCU), specified in integer numbers of MOPS. The approximated RAM usage is given in "RAM Complexity Units" (RCU), specified in mostly integer numbers of kWords (1000 words). The RCU numbers do not include working buffers that can be shared between different objects and/or channels.

If a profile level is specified by the maximum number of complexity units, then a flexible configuration of the decoder handling different types of objects is allowed under the constraint that both values for the total complexity for decoding and sampling rate conversion (if needed) do not exceed this limit.

The following table gives complexity estimates for the different object types. PCU values are given in MOPS per channel, RCU values in kWords per channel (with respect to AAC, channel refers to Main channel, e. g. the channel of a SCE, one channel of a CPE, or the channel of an independently switched CCE).

Table 1.4 — Complexity of Audio Object Types and SR conversion

Object Type	Parameters	PCU (MOPS)	RCU	Remark
AAC Main	fs = 48 kHz	5	5	1)
AAC LC	fs = 48 kHz	3	3	1)
AAC SSR	fs = 48 kHz	4	3	1)
AAC LTP	fs = 48 kHz	4	4	1)
SBR	fs = 24/48 kHz (in/out) (SBR tool)	3	2.5	1)
	fs = 24/48 kHz (in/out) (Low Power SBR tool)	2	1.5	1)
	fs = 48/48 kHz (in/out) (Down Sampled SBR tool)	4.5	2.5	1)
	fs = 48/48 kHz (in/out) (Low Power Down Sampled SBR tool)	3	1.5	1)
AAC Scalable	fs = 48 kHz	5	4	1), 2)
TwinVQ	fs = 24 kHz	2	3	1)
CELP	fs = 8 kHz	1	1	
CELP	fs = 16 kHz	2	1	
CELP	fs = 8/16 kHz (bandwidth scalable)	3	1	
HVXC	fs = 8 kHz	2	1	
TTSI		-	-	4)
General MIDI		4	1	
Wavetable Synthesis	fs = 22.05 kHz	depends on bitstream payloads (3)	depends on bitstream payloads (3)	
Main Synthetic		depends on bitstream	depends on bitstream	
Algorithmic Synthesis and AudioFX		depends on bitstream	depends on bitstream	
Sampling Rate Conversion	rf = 2, 3, 4, 6, 8, 12	2	0.5	7)
ER AAC LC	fs = 48 kHz	3	3	1)
ER AAC LTP	fs = 48 kHz	4	4	1)
ER AAC Scalable	fs = 48 kHz	5	4	1), 2)
ER TwinVQ	fs = 24 kHz	2	3	1)
ER BSAC	fs = 48 kHz (input buffer size=26000bits)	4	4	1)
	fs = 48 kHz (input buffer size=106000bits)	4	8	
ER AAC LD	fs = 48 kHz	3	2	1)
ER CELP	fs = 8 kHz	2	1	
	fs = 16 kHz	3	1	
ER HVXC	fs = 8 kHz	2	1	
ER HILN	fs = 16 kHz, ns=93	15	2	6)
	fs = 16 kHz, ns=47	8	2	
ER Parametric	fs = 8 kHz, ns=47	4	2	5), 6)

Definitions:

- fs = sampling frequency
- rf = ratio of sampling rates

Notes:

- 1) PCU proportional to sampling frequency.
- 2) Includes core decoder.

- 3) See ISO/IEC 14496-4.
- 4) The complexity for speech synthesis is not taken into account.
- 5) Parametric coder in HILN mode, for HVXC mode see ER HVXC.
- 6) PCU depends on f_s and n_s , see below.
- 7) Sampling Rate Conversion is needed, if objects of different sampling rates are combined in a scene (see ISO/IEC 14496-1:2000, subclause 9.2.2.13.2.1). The specified values have to be added for each required conversion.

PCU for HILN:

The computational complexity of HILN depends on the sampling frequency f_s and the maximum number of sinusoids n_s to be synthesized simultaneously. The value of n_s for a frame is the total number of harmonic and individual lines synthesized in that frame, i.e. the number of starting plus continued plus ending lines. For f_s in kHz, the PCU in MOPS is calculated as follows:

$$PCU = (1 + 0.15 \cdot n_s) \cdot f_s / 16$$

The typical maximum values of n_s are 47 for 6 kbit/s HILN and 93 for 16 kbit/s HILN streams.

PCU and RCU for AAC:

For AAC object types, PCU and RCU depend on sampling frequency and channel configuration in the following way:

PCUs

$$PCU = (f_s / f_{s_ref}) \cdot PCU_ref \cdot (2 \cdot \#CPE + \#SCE + \#LFE + \#IndepCouplingCh + 0.3 \cdot \#DepCouplingCh)$$

f_s : actual sampling frequency

f_{s_ref} : reference sampling frequency (sampling frequency for the given PCU_ref)

PCU_ref : reference PCU given in Table 1.4

$\#SCE$: number of SCEs

$\#CPE$: ...

RCUs

$\#CPE < 2$:

$$RCU = RCU_ref \cdot [\#SCE + 0.5 \cdot \#LFE + 0.5 \cdot \#IndepCouplingCh + 0.4 \cdot \#DepCouplingCh] + [RCU_ref + (RCU_ref - 1)] \cdot \#CPE$$

$\#CPE \geq 2$:

$$RCU = RCU_ref \cdot [\#SCE + 0.5 \cdot \#LFE + 0.5 \cdot \#IndepCouplingCh + 0.4 \cdot \#DepCouplingCh] + [RCU_ref + (RCU_ref - 1) \cdot (2 \cdot \#CPE - 1)]$$

RCU_ref : reference RCU given in Table 1.4

$\#SCE$: number of SCEs

$\#CPE$: ...

1.5.2.3 Levels within the profiles

The notation used to specify the number of audio channels indicates the number of main audio channels. Based on the number of main audio channels (A), Table 1.5 indicates for object types derived from the AAC multichannel syntax the number of LFE channels (L), the number of independently switched coupling channels (I), and the number of dependently switched coupling channels (D) in the form A.L.I.D.

Table 1.5 — Maximum number of the individual AAC channel types depending on the specified number of main audio channels

AOT	Number of Main Audio Channels		
	1	2	5
1 (AAC main)	1.0.0.0	2.0.0.0	5.1.1.1
2 (AAC LC)	1.0.0.0	2.0.0.0	5.1.0.1
3 (AAC SSR)	1.0.0.0	2.0.0.0	5.1.0.0
4 (AAC LTP)	1.0.0.0	2.0.0.0	5.1.0.1
17 (ER AAC LC)	1.0.0.0	2.0.0.0	5.1.0.0
19 (ER AAC LTP)	1.0.0.0	2.0.0.0	5.1.0.0
23 (ER AAC LD)	1.0.0.0	2.0.0.0	5.1.0.0

Note: In the case of scalable coding schemes, only the first instantiation of each object type will be counted to determine the number of objects relevant to the level definition and complexity metric. For example, in a scalable coder consisting of a CELP core coder and two enhancement layers implemented by means of AAC scalable objects, one CELP object and one AAC scalable object and their associated complexity metrics are counted since there is almost no overhead associated with the second (and any further) GA enhancement layer.

- **Levels for Speech Audio Profile**

Two levels are defined by number of objects:

1. One speech object.
2. Up to 20 speech objects.

- **Levels for Synthetic Audio Profile**

Three levels are defined :

1. Synthetic Audio 1: All bitstream payload elements may be used with:
 - "Low processing" (exact numbers in ISO/IEC 14496-4:2000)
 - Only core sample rates may be used
 - No more than one TTSI object
2. Synthetic Audio 2: All bitstream payload elements may be used with:
 - "Medium processing" (exact numbers in ISO/IEC 14496-4:2000).
 - Only core sample rates may be used.
 - no more than four TTSI objects.
3. Synthetic Audio 3: All bitstream payload elements may be used with:
 - "High processing" (exact numbers in ISO/IEC 14496-4:2000).
 - no more than twelve TTSI objects.

- **Levels for Scalable Audio Profile**

Four levels are defined by configuration; complexity units define the fourth level:

1. Maximum 24 kHz of sampling rate, one mono object (all object Types).
 2. Maximum 24 kHz of sampling rate, one stereo object or two mono objects (all object Types).
 3. Maximum 48 kHz of sampling rate, one stereo object or two mono objects (all object Types).
 4. Maximum 48 kHz of sampling rate, one object with 5 main channels or multiple objects with at maximum one integer factor sampling rate conversion for a maximum of two channels.
- Flexible configuration is allowed with $PCU < 30$ and $RCU < 19$.

For the audio object types 2 (AAC LC), and 4 (AAC LTP), only a frame length of 1024 samples is permitted with respect to level 1, 2, 3 and 4. For the audio object types 2 (AAC LC) and 4 (AAC LTP), mono or stereo mixdown elements are not permitted with respect to level 1, 2, 3 and 4. For the audio object type 6 (AAC Scalable) the following restrictions apply. The number of AAC layers shall not exceed 8 in any scalable configuration. If audio object type 8 (CELP) is used as core layer coder, the number of CELP layers shall not exceed 2. If audio object type 7 (TwinVQ) is used as base layer coder, only one mono TwinVQ layer is permitted.

• Levels for Main Audio Profile

Main Audio Profile contains all natural and synthetic object types. Levels are then defined as a combination of the two different types of levels from the two different metrics defined for natural tools (computation-based metrics) and synthetic tools (macro-oriented metrics).

For Object Types not belonging to the Synthetic Profile four levels are defined:

1. Natural Audio 1: $PCU < 40$, $RCU < 20$
2. Natural Audio 2: $PCU < 80$, $RCU < 64$
3. Natural Audio 3: $PCU < 160$, $RCU < 128$
4. Natural Audio 4: $PCU < 320$, $RCU < 256$

For Object Types belonging to the Synthetic Profile the same three Levels are defined as above, i.e. Synthetic Audio 1, Synthetic Audio 2 and Synthetic Audio 3.

Four Levels are then defined for Main Profile:

1. Natural Audio 1 + Synthetic Audio 1
2. Natural Audio 2 + Synthetic Audio 1
3. Natural Audio 3 + Synthetic Audio 2
4. Natural Audio 4 + Synthetic Audio 3

For the audio object types 1 (AAC main), 2 (AAC LC), 3 (AAC SSR), and 4 (AAC LTP), only a frame length of 1024 samples is permitted with respect to level 1, 2, 3 and 4. For the audio object types 1 (AAC Main), 2 (AAC LC), 3 (AAC SSR) and 4 (AAC LTP), mono or stereo mixdown elements are not permitted with respect to level 1, 2, 3 and 4. For the audio object type 6 (AAC Scalable) the following restrictions apply. The number of AAC layers shall not exceed 8 in any scalable configuration. If audio object type 8 (CELP) is used as core layer coder, the number of CELP layers shall not exceed 2. If audio object type 7 (TwinVQ) is used as base layer coder, only one mono TwinVQ layer is permitted.

• Levels for the High Quality Audio Profile

Table 1.6 — Levels for the High Quality Audio Profile

Level	Max. channels/object	Max. sampling rate [kHz]	Max PCU ^{*2}	Max RCU ^{*2}	EP-Tool: Max. redundancy by class FEC ^{*1}	EP-Tool: Max. # stages of interleaving per object
1	2	22.05	5	8	0 %	0
2	2	48	10	8	0 %	0
3	5	48	25	12 ^{*3}	0 %	0
4	5	48	100	42 ^{*3}	0 %	0
5	2	22.05	5	8	20%	9
6	2	48	10	8	20%	9
7	5	48	25	12 ^{*3}	20%	22
8	5	48	100	42 ^{*3}	20%	22

- *1: The value specifies the maximum redundancy based on the available audio object with the longest maximum frame length. The redundancy might be larger in the case of smaller frame lengths. However, the application of any class FEC is not permitted if 0 % is specified. The limit is valid independently for each audio object. Since this value does neither cover the EP header and its protection nor any CRC, another 5 % must always be added to this value in order to derive the necessary increase of the minimum decoder input buffer. This does imply, that not more than those 5 % might be spent for the EP header and its protection or any CRC.
- *2: Level 5 to 8 do not include RAM and computational complexity for the EP tool.
- *3: Sharing of work buffers between multiple objects or channel pair elements is assumed.

For the audio object types 2 (AAC LC), 4 (AAC LTP), 17 (ER AAC LC), and 19 (ER AAC LTP) only a frame length of 1024 samples is permitted with respect to level 1, 2, 3, 4, 5, 6, 7 and 8. For the audio object types 2 (AAC LC) and 4 (AAC LTP), mono or stereo mixdown elements are not permitted with respect to level 1, 2, 3, 4, 5, 6, 7 and 8. For the audio object type 6 and 20 ((ER) AAC Scalable) the following restrictions apply. The number of AAC layers shall not exceed 8 in any scalable configuration. If audio object type 8 or 24 ((ER) CELP) is used as core layer coder, the number of CELP layers shall not exceed 2.

• Levels for the Low Delay Audio Profile

Table 1.7 — Levels for the Low Delay Audio Profile

Level	Max. channels/object	Max. sampling rate [kHz]	Max PCU ^{*2}	Max RCU ^{*2}	EP-Tool: Max. redundancy by class FEC ^{*1}	EP-Tool: Max. # stages of interleaving per object
1	1	8	2	1	0 %	0
2	1	16	3	1	0 %	0
3	1	48	3	2	0 %	0
4	2	48	24	12 ^{*3}	0 %	0
5	1	8	2	1	100%	5
6	1	16	3	1	100%	5
7	1	48	3	2	20%	5
8	2	48	24	12 ^{*3}	20%	9

- *1: The value specifies the maximum redundancy based on the available audio object with the longest maximum frame length. The redundancy might be larger in the case of smaller frame lengths. However, the application of any class FEC is not permitted if 0 % is specified. The limit is valid independently for each audio object. Since this value does neither cover the EP header and its protection nor any CRC, another 5 % must always be added to this value in order to derive the necessary increase of the minimum decoder input buffer. This does imply, that not more than those 5 % might be spent for the EP header and its protection or any CRC.
- *2: Level 5 to 8 do not include RAM and computational complexity for the EP tool.
- *3: Sharing of work buffers between multiple objects or channel pair elements is assumed.

• Levels for the Natural Audio Profile

Table 1.8 — Levels for the Natural Audio Profile

Level	Max. sampling rate [kHz]	Max PCU ^{*2}	EP-Tool: Max. redundancy by class FEC ^{*1}	EP-Tool: Max. # stages of interleaving per object
1	48	20	0 %	0
2	96	100	0 %	0
3	48	20	20%	9
4	96	100	20%	22

- *1: The value specifies the maximum redundancy based on the available audio object with the longest maximum frame length. The redundancy might be larger in the case of smaller frame lengths. However, the application of any class FEC is not permitted if 0 % is specified. The limit is valid independently for each audio object.

Since this value does neither cover the EP header and its protection nor any CRC, another 5 % must always be added to this value in order to derive the necessary increase of the minimum decoder input buffer. This does imply, that not more than those 5 % might be spent for the EP header and its protection or any CRC.

*2: Level 3 and 4 do not include computational complexity for the EP tool.

No RCU limitations are specified for this profile.

For the audio object types 1 (AAC main), 2 (AAC LC), 3 (AAC SSR), 4 (AAC LTP), 17 (ER AAC LC), and 19 (ER AAC LTP) only a frame length of 1024 samples is permitted with respect to level 1, 2, 3 and 4. For the audio object types 1 (AAC Main), 2 (AAC LC), 3 (AAC SSR) and 4 (AAC LTP), mono or stereo mixdown elements are not permitted with respect to level 1, 2, 3 and 4. For the audio object type 6 and 20 ((ER)AAC Scalable) the following restrictions apply. The number of AAC layers shall not exceed 8 in any scalable configuration. If audio object type 8 or 24 ((ER) CELP) is used as core layer coder, the number of CELP layers shall not exceed 2. If audio object type 7 or 21 ((ER) TwinVQ) is used as base layer coder, only one mono TwinVQ layer is permitted.

- Levels for the Mobile Audio Internetworking Profile**

Table 1.9 — Levels for the Mobile Audio Internetworking Profile

Level	Max. channels/object	Max. sampling rate [kHz]	Max PCU ^{*3}	Max RCU ^{*2 *3}	Max. # audio objects	EP-Tool: Max. redundancy by class FEC ^{*1}	EP-Tool: Max. # stages of interleaving per object
1	1	24	2.5	4	1	0 %	0
2	2	48	10	8	2	0 %	0
3	5	48	25	12 ^{*4}	-	0 %	0
4	1	24	2.5	4	1	20%	5
5	2	48	10	8	2	20%	9
6	5	48	25	12 ^{*4}	-	20%	22

*1: The value specifies the maximum redundancy based on the available audio object with the longest maximum frame length. The redundancy might be larger in the case of smaller frame lengths. However, the application of any class FEC is not permitted if 0 % is specified. The limit is valid independently for each audio object. Since this value does neither cover the EP header and its protection nor any CRC, another 5 % must always be added to this value in order to derive the necessary increase of the minimum decoder input buffer. This does imply, that not more than those 5 % might be spent for the EP header and its protection or any CRC.

*2: The maximum RCU for one channel in any object in this profile is 4. For the ER BSAC, this limits the input buffer size. The maximum possible input buffer size in bits for this case is given in PCU/RCU (Table 1.4).

*3: Level 4 to 6 do not include RAM and computational complexity for the EP tool.

*4: Sharing of work buffers between multiple objects or channel pair elements are assumed.

For the audio object type 17 (ER AAC LC) only a frame length of 1024 samples is permitted with respect to level 1,2,3,4,5 and 6. For the audio object type 20 (ER AAC Scalable) the following restrictions apply. The number of AAC layers shall not exceed 8 in any scalable configuration. If audio object type 21 (ER TwinVQ) is used as base layer coder, only one mono TwinVQ layer is permitted.

- Levels for the AAC Profile**

Table 1.10 — Levels for the AAC Profile

Level	Max. channels/ object	Max. sampling rate [kHz]	Max. PCU	Max. RCU
1	2	24	3	5
2	2	48	6	5
3	NA	NA	NA	NA
4	5	48	19	15
5	5	96	38	15

For the audio object type 2 (AAC LC), mono or stereo mixdown elements are not permitted.

The NA (Not Applicable) levels are introduced to emphasize the hierarchical structure of the AAC Profile and the High Efficiency AAC Profile. Hence, a decoder supporting the High Efficiency AAC Profile at a given level can decode an AAC Profile stream of the same or a lower level. The NA levels are not indicated in the audioProfileLevelIndication table (Table 1.12).

- Levels for the High Efficiency AAC Profile**

Table 1.11 — Levels for the High Efficiency AAC Profile

Level	Max. channels/ object	Max. AAC sampling rate, SBR not present [kHz]	Max. AAC sampling rate, SBR present [kHz]	Max. SBR sampling rate [kHz] (in/out)	Max. PCU	Max. RCU	Max. PCU Low power SBR	Max. RCU Low power SBR
1	NA	NA	NA	NA	NA	NA	NA	NA
2	2	48	24	24/48	9	10	7	8
3	2	48	48	48/48 (Note 1)	15	10	12	8
4	5	48	24/48 (Note 2)	48/48 (Note 1)	25	28	20	23
5	5	96	48	48/96	49	28	39	23

Note 1: For level 3 and level 4 decoders, it is mandatory to operate the SBR tool in downsampled mode if the sampling rate of the AAC core is higher than 24kHz. Hence, if the SBR tool operates on a 48kHz AAC signal, the internal sampling rate of the SBR tool will be 96kHz, however, the output signal will be downsampled by the SBR tool to 48kHz.

Note 2: For one or two channels the maximum AAC sampling rate, with SBR present, is 48kHz. For more than two channels the maximum AAC sampling rate, with SBR present, is 24kHz.

For the audio object type 2 (AAC LC), mono or stereo mixdown elements are not permitted.

1.5.2.4 audioProfileLevelIndication

audioProfileLevelIndication is a data element of the InitialObjectDescriptor defined in ISO/IEC 14496-1. It indicates the profile and level required to process the content associated with this InitialObjectDescriptor as defined in Table 1.12.

Table 1.12 — audioProfileLevelIndication values

Value	Profile	Level
0x00	Reserved for ISO use	-
0x01	Main Audio Profile	L1
0x02	Main Audio Profile	L2
0x03	Main Audio Profile	L3
0x04	Main Audio Profile	L4
0x05	Scalable Audio Profile	L1
0x06	Scalable Audio Profile	L2
0x07	Scalable Audio Profile	L3
0x08	Scalable Audio Profile	L4
0x09	Speech Audio Profile	L1
0x0A	Speech Audio Profile	L2
0x0B	Synthetic Audio Profile	L1
0x0C	Synthetic Audio Profile	L2
0x0D	Synthetic Audio Profile	L3
0x0E	High Quality Audio Profile	L1
0x0F	High Quality Audio Profile	L2
0x10	High Quality Audio Profile	L3
0x11	High Quality Audio Profile	L4
0x12	High Quality Audio Profile	L5
0x13	High Quality Audio Profile	L6
0x14	High Quality Audio Profile	L7
0x15	High Quality Audio Profile	L8
0x16	Low Delay Audio Profile	L1
0x17	Low Delay Audio Profile	L2
0x18	Low Delay Audio Profile	L3
0x19	Low Delay Audio Profile	L4
0x1A	Low Delay Audio Profile	L5
0x1B	Low Delay Audio Profile	L6
0x1C	Low Delay Audio Profile	L7
0x1D	Low Delay Audio Profile	L8
0x1E	Natural Audio Profile	L1
0x1F	Natural Audio Profile	L2
0x20	Natural Audio Profile	L3
0x21	Natural Audio Profile	L4
0x22	Mobile Audio Internetworking Profile	L1
0x23	Mobile Audio Internetworking Profile	L2
0x24	Mobile Audio Internetworking Profile	L3
0x25	Mobile Audio Internetworking Profile	L4
0x26	Mobile Audio Internetworking Profile	L5
0x27	Mobile Audio Internetworking Profile	L6
0x28	AAC Profile	L1
0x29	AAC Profile	L2
0x2A	AAC Profile	L4
0x2B	AAC Profile	L5
0x2C	High Efficiency AAC Profile	L2
0x2D	High Efficiency AAC Profile	L3
0x2E	High Efficiency AAC Profile	L4
0x2F	High Efficiency AAC Profile	L5
0x30 - 0x7F	reserved for ISO use	-
0x80 - 0xFD	user private	-
0xFE	no audio profile specified	-
0xFF	no audio capability required	-
NOTE — Usage of the value 0xFE indicates that the content described by this InitialObjectDescriptor does not comply to any audio profile specified in ISO/IEC 14496-3.		

Usage of the value 0xFF indicates that none of the audio profile capabilities are required for this content.

1.6 Interface to ISO/IEC 14496-1 (MPEG-4 Systems)

1.6.1 Introduction

The header streams are transported via MPEG-4 systems. These streams contain configuration information, which is necessary for the decoding process and parsing of the raw data streams. However, an update is only necessary if there are changes in the configuration.

The payloads contain all information varying on a frame to frame basis and therefore carry the actual audio information.

1.6.2 Syntax

1.6.2.1 AudioSpecificConfig

AudioSpecificConfig() extends the abstract class DecoderSpecificInfo, as defined in ISO/IEC 14496-1, when DecoderConfigDescriptor.objectTypeIndication refers to streams complying with ISO/IEC 14496-3 in this case the existence of AudioSpecificConfig() is mandatory.

Table 1.13 — Syntax of AudioSpecificConfig()

Syntax	No. of bits	Mnemonic
AudioSpecificConfig ()		
{		
audioObjectType = GetAudioObjectType();		
samplingFrequencyIndex;	4	bslbf
if (samplingFrequencyIndex == 0xf) {		
samplingFrequency;	24	uimsbf
}		
channelConfiguration;	4	bslbf
sbrPresentFlag = -1;		
if (audioObjectType == 5) {		
extensionAudioObjectType = audioObjectType;		
sbrPresentFlag = 1;		
extensionSamplingFrequencyIndex;	4	uimsbf
if (extensionSamplingFrequencyIndex == 0xf)		
extensionSamplingFrequency;	24	uimsbf
audioObjectType = GetAudioObjectType();		
}		
else {		
extensionAudioObjectType = 0;		
}		
switch (audioObjectType) {		