INTERNATIONAL STANDARD

ISO/IEC 14496-3

First edition
1999-12-15

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 14496-3:1999(E)

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 copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 734 10 79
E-mail copyright@iso.ch
Web www.iso.ch

Printed 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. 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 part of ISO/IEC 14496 may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

International Standard 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*.

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 testing*

— *Part 6: Delivery Multimedia Integration Framework (DMIF)*

Annexes 2.A to 2.C, 3.C, 4.A and 5.A form a normative part of this part of ISO/IEC 14496. Annexes 1.A, 1.B, 2.D, 3.A, 3.B, 3.D to 3.F, 4.B and 5.B to 5.G are for information only.

Due to its technical nature, this part of ISO/IEC 14496 requires a special format as several standalone electronic files and, consequently, does not conform to some of the requirements of the ISO/IEC Directives, Part 3.

Information technology — Coding of audio-visual objects —

Part 3: Audio

Subpart 1: Main

Structure of this part of ISO/IEC 14496:

This part of ISO/IEC 14496 comprises six subparts:

Subpart 1: Main
Subpart 2: Speech coding - HVXC
Subpart 3: Speech coding - CELP
Subpart 4: General Audio coding (GA)
Subpart 5: Structured audio
Subpart 6: Text to speech interface

For reasons of manageability of large documents, this part of ISO/IEC 14496 is divided into six files, corresponding to the six subparts of the standard:

1. a025035e.pdf contains Subpart 1.
2. b025035e.pdf contains Subpart 2.
3. c025035e.pdf contains Subpart 3.
4. d025035e.pdf contains Subpart 4.
5. e025035e.pdf contains Subpart 5.
6. f025035e.pdf contains Subpart 6.

Contents for Subpart 1

1.1 SCOPE 4
1.1.1 Overview of MPEG-4 Audio 4
1.1.2 New concepts in MPEG-4 Audio 4
1.1.3 MPEG-4 Audio capabilities 5
1.1.3.1 Overview of capabilities 5
1.1.3.2 MPEG-4 speech coding tools 5
1.1.3.3 MPEG-4 general audio coding tools 6
1.1.3.4 MPEG-4 Audio synthesis tools 6
1.1.3.5 MPEG-4 Audio composition tools 7
1.1.3.6 MPEG-4 Audio scalability tools 8
1.2 NORMATIVE REFERENCES 8
1.3 TERMS AND DEFINITIONS 8
1.4 SYMBOLS AND ABBREVIATIONS 10
1.4.1 Arithmetic operators 11
1.4.2 Logical operators 11
1.4.3 Relational operators 11
1.4.4 Bitwise operators 11
1.4.5 Assignment 11
1.4.6 Mnemonics 12
1.4.7 Constants 12
1.4.8 Method of describing bitstream syntax 12
1.5 TECHNICAL OVERVIEW 13
1.5.1 MPEG-4 Audio Object Types 13
1.5.1.1 Audio Object Type Definition 13
1.5.1.2 Description 14
1.5.2 Audio Profiles and Levels 15
1.5.2.1 Profiles 15
1.5.2.2 Complexity Units 16
1.5.2.3 Levels within the Profiles 17

1.6 INTERFACE TO MPEG-4 SYSTEMS 18
1.6.1 Introduction 18
1.6.2 Syntax 18
1.6.2.1 Audio DecoderSpecificInfo 18
1.6.2.2 HvxcSpecificConfig 19
1.6.2.3 CelpSpecificConfig 19
1.6.2.4 GASpecificConfig 19
1.6.2.5 StructuredAudioSpecificConfig 19
1.6.2.6 TTSSpecificConfig 19
1.6.2.7 Payloads 19
1.6.3 Semantics 19
1.6.3.1 AudioObjectType 19
1.6.3.2 samplingFrequency 19
1.6.3.3 samplingFrequencyIndex 19
1.6.3.4 channelConfiguration 20
ANNEX 1.A (INFORMATIVE) AUDIO INTERCHANGE FORMATS 21
1.A.1 Introduction 21
1.A.2 Interchange format streams 21
1.A.2.1 MPEG-2 AAC Audio_Data_Interchange_Format, ADIF 21
1.A.2.2 Audio_Data_Transport_Stream frame, ADTS 22
1.A.2.2.4 MPEG-4 Audio Transport Stream (MATS) 23
1.A.2 Decoding of interface formats 26
1.A.2.1 Audio_Data_Interchange_Format (ADIF) 26
1.A.2.2 Audio_Data_Transport_Stream (ADTS) 26
1.A.2.3 MPEG-4 Audio Transport Stream (MATS) 28
ANNEX 1.B (INFORMATIVE) LIST OF PATENT HOLDERS 31

Subpart 1: Main

1.1 Scope

1.1.1 Overview of MPEG-4 Audio

This part of ISO/IEC 14496 (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. A multimedia terminal that can decode the Main Profile of MPEG-4 Audio has audio capabilities that cover the entire spectrum of audio functionality available today and in the future.

The remainder of this clause 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.

1.1.2 New concepts in MPEG-4 Audio

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

- **MPEG-4 has no standard for transport.** In all of the MPEG-4 tools for audio and visual coding, the coding standard ends at the point of constructing a sequence of *access units* that contain the compressed data. The MPEG-4 Systems (ISO/IEC 14496-1) specification describes how to convert the individually coded objects into a bitstream that contains a number of multiplexed sub-streams.

  There is no standard mechanism for transport of this stream 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 bitstream specification allows transmission functions that are much more sophisticated than are possible with previous MPEG standards.

  For applications which do not require sophisticated transport functionality, object-based coding, synchronization with other media, or other functions provided by MPEG-4 Systems, a private, not normative transport may be used to deliver a single MPEG-4 Audio stream. An example private transport for this purpose is given in Informative Annex A of subpart 1.

- **MPEG-4 Audio encourages 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 tools specified in MPEG-4 are the state-of-the-art tools available for low-bitrate coding of speech and other audio.

- **MPEG-4 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 (subclause 5.1) 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 all that was transmitted. In MPEG-4, by contrast, the concept of a *soundtrack* is much more flexible. Multiple tools may be used to transmit several *audio objects*; when using multiple tools together, an *audio composition* system is used to create a single soundtrack from the 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 allows significant advantages in quality and flexibility in MPEG-4 over previous audio standards.

- **MPEG-4 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. MPEG-4 also 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.

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

1.1.3 MPEG-4 Audio capabilities

1.1.3.1 Overview of capabilities

The MPEG-4 Audio tools can be broadly organized into several categories:

1. *Speech* tools for the transmission and decoding of synthetic and natural speech
2. *Audio* tools for the transmission and decoding of recorded music and other audio soundtracks
3. *Synthesis* tools for very low bitrate description and transmission, and terminal-side synthesis, of synthetic music and other sounds
4. *Composition* tools for object-based coding, interactive functionality, and audiovisual synchronization
5. *Scalability* tools for the creation of bitstreams that can be transmitted, without recoding, at several different bitrates

Each of these types of tools will be described in more detail in the following subclauses.

1.1.3.2 MPEG-4 speech coding tools

1.1.3.2.1 Introduction

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 videoteleconferencing applications. Each of these tools will be discussed.

1.1.3.2.2 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 the variable bitrate coding is allowed, coding at even less than 2 kbit/s, such as 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 coder targets applications from mobile and satellite communications, to Internet telephony, to packaged media and speech databases. It meets a wide range of requirements covering bitrates, functionality and sound quality and is specified in subparts 2 and 3.

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, in variable bitrate mode. HVXC provides communications-quality to near-toll-quality speech in the 100-3800 Hz band at 8kHz sampling rate. HVXC also allows independent change of speed and pitch during decoding, which is a powerful functionality for fast access to 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 bitstreams at arbitrary bitrates. The MPEG-4 CELP coder supports two sampling rates, namely, 8 and 16 kHz. The associated bandwidths are 100 – 3800 Hz for 8 kHz sampling and 50 – 7000 Hz for 16 kHz sampling.

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

1.1.3.2.3 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 sound. Before MPEG-4, however, there was no way for a multimedia content provider to easily give instructions to an unknown TTS system. In MPEG-4, 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. It is specified in subpart 6.

The MPEG-4 TTS package, Hybrid/Multi-Level Scalable TTS Interface, can be considered as 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 format is strongly 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 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.

1.1.3.3 MPEG-4 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 the use of the MPEG-2 AAC standard (ISO/IEC 13818-7), with certain improvements, within the MPEG-4 tool set. 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.

Subpart 4 of MPEG-4 specifies the AAC tool set, in the General Audio coder. This coding technique 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 for nearly 10 years; MPEG-4 General Audio coding continues this tradition.

For bitrates from 6 kbit/s up to 64 kbit/s per channel, the MPEG-4 standard provides extensions to AAC and the TwinVQ tools that allow the content author to achieve highest quality by altering the tool used depending on the bit rate. Furthermore, various bit rate scalability options are available within the GA coder (see subclause 1.1.3.6.). The low-bitrate techniques and scalability modes provided with this tool set have also been verified in formal tests by MPEG.

1.1.3.4 MPEG-4 Audio synthesis tools

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. (There is also a tool for the transmission of synthetic speech; it is

described above in subclause 1.2.2 and in subpart 6). 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. 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—SAOL, the Structured Audio Orchestra Language—and that for the description of scores—SASL, the Structured Audio Score Language—are specified in subpart 6.

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.

1.1.3.5 MPEG-4 Audio composition tools

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. 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 bitstream 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.

1.1.3.6 MPEG-4 Audio scalability tools

Many of the bitstream 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 to be parsed into a bitstream of lower bitrate such that the combination can still be decoded into a meaningful signal. The bitstream 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 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 encoder 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 bitstreams. 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 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.

1.2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 14496. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of ISO 14496 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

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 13818-1:1996, *Information technology - Generic coding of moving pictures and associated audio information: Systems.*
ISO/IEC 13818-3:1998, *Information technology - Generic coding of moving pictures and associated audio information - Part 3: Audio*.
ISO/IEC 13818-7:1997, *Information technology - Generic coding of moving pictures and associated audio information - Part 7: Advanced Audio Coding (AAC).*
(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*.

Contents for Subpart 2

2.1 Scope 4
2.2 Definitions 4
2.3 Bitstream syntax 5
2.3.1 Decoder configuration (HvxcSpecificConfig) 5
2.3.2 Bitstream frame (alPduPayload) 6
2.3.2.1 HVXC bitstream frame 6
2.4 Bitstream semantics 9
2.4.1 Decoder configuration (HvxcSpecificConfig) 9
2.4.2 Bitstream frame (alPduPayload) 9
2.5 HVXC decoder tools 10
2.5.1 Overview 10
2.5.1.1 Framing structure and block diagram of the decoder 10
2.5.1.2 Delay mode 11
2.5.2 LSP decoder 13
2.5.2.1 Tool description 13
2.5.2.2 Definitions 13
2.5.2.3 Decoding process 14
2.5.2.4 Tables 17
2.5.3 Harmonic VQ decoder 17
2.5.3.1 Tool description 17
2.5.3.2 Definitions 17
2.5.3.3 Decoding process 18
2.5.3.4 Tables 20
2.5.4 Time domain decoder 20
2.5.4.1 Tool description 20
2.5.4.2 Definitions 21
2.5.4.3 Decoding process 21
2.5.4.4 Tables 22
2.5.5 Parameter interpolation for speed control 22
2.5.5.1 Tool description 22
2.5.5.2 Definitions 22

2.5.5.3 Speed control process **22**

2.5.6 Voiced component synthesizer **24**

2.5.6.1 Tool description **24**

2.5.6.2 Definitions **25**

2.5.6.3 Synthesis process **26**

2.5.7 Unvoiced component synthesizer **34**

2.5.7.1 Tool description **34**

2.5.7.2 Definitions **34**

2.5.7.3 Synthesis process **34**

2.5.8 Variable rate decoder **36**

2.5.8.1 Tool description **36**

2.5.8.2 Definitions **36**

2.5.8.3 Decoding process **37**

Annex 2.A (informative) **HVXC Encoder tools** **39**

2.A.1 Overview of encoder tools **39**

2.A.2 Normalization **39**

2.A.2.1 Tool description **39**

2.A.2.2 Normalization process **39**

2.A.3 Pitch estimation **43**

2.A.3.1 Tool description **43**

2.A.3.2 Pitch estimation process **43**

2.A.3.3 Pitch tracking **43**

2.A.4 Harmonic magnitudes extraction **45**

2.A.4.1 Tool description **45**

2.A.4.2 Harmonic magnitudes extraction process **45**

2.A.5 Perceptual weighting **46**

2.A.5.1 Tool description **46**

2.A.6 Harmonic VQ encoder **46**

2.A.6.1 Tool description **46**

2.A.6.2 Encoding process **46**

2.A.7 V/UV decision **47**

2.A.7.1 Tool description **47**

2.A.7.2 Encoding process **47**

2.A.8 Time domain encoder **48**

2.A.8.1 Tool description **48**

2.A.8.2 Encoding process **48**

2.A.9 Variable rate encoder **49**

Annex 2.B (informative) **HVXC Decoder tools** **53**

2.B.1 Postfilter **53**

2.B.1.1 Tool description **53**

2.B.1.2 Definitions **53**

2.B.1.3 Processing **53**

2.B.1.3.1 Voiced speech **53**

2.B.1.3.2 Unvoiced speech **54**

2.B.2 Post processing **54**

2.B.2.1 Tool description **54**

2.B.2.2 Definitions **55**

Annex 2.C (informative) **System layer definitions** **57**

2.C.1 Random access point **57**

2.C.2 MPEG-4 Audio Transport Stream (MATS) **57**

Annex 2.D (normative) **VQ codebooks for HVXC** **58**

2.D.1 List of the VQ codebooks **58**

2.D.2 CbAm **58**

2.D.3 CbAm4k **65**

2.D.4 CbCelp **95**

2.D.5 CbCelp4k **103**

2.D.6 CbLsp **106**

2.D.7 CbLsp4k **110**

Subpart 2: Speech coding - HVXC

2.1 Scope

MPEG-4 parametric speech coding uses Harmonic Vector eXcitation Coding (HVXC) algorithm, where harmonic coding of LPC residual signals for voiced segments and Vector eXcitation Coding (VXC) for unvoiced segments are employed. HVXC allows coding of speech signals at 2.0 kbps and 4.0 kbps with a scalable scheme, where 2.0 kbps decoding is possible not only using the 2.0 kbps bit-stream but also using a 4.0 kbps bit-stream. HVXC also provides variable bit rate coding where a typical average bit-rate is around 1.2-1.7 kbit/s. Independent change of speed and pitch during decoding is possible, which is a powerful functionality for fast data base search. The frame length is 20 ms, and one of four different algorithmic delays, 33.5 ms, 36ms, 53.5 ms, 56 ms can be selected.

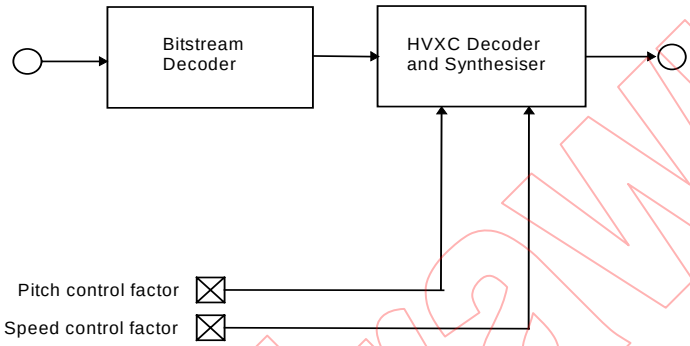

Figure 2.1.1 - Block diagram of the parametric speech decoder

Contents for Subpart 3

3.1 Scope 3

3.1.1 General description of the CELP decoder 3

3.1.2 Functionality of MPEG-4 CELP 3

3.2 Definitions 5

3.3 Bitstream syntax 6

3.3.1 Header syntax 7

3.3.2 Frame syntax 7

3.3.3 LPC syntax 8

3.3.4 Excitation syntax 9

3.4 Semantics 10

3.5 MPEG-4 CELP Decoder tools 15

3.5.1 General Introduction to the MPEG-4 CELP decoder tool-set 15

3.5.2 AAC/CELP scalable configuration 16

3.5.3 Helping variables 16

3.5.4 Bitstream elements for the MPEG-4 CELP decoder tool-set 17

3.5.5 CELP bitstream demultiplexer 17

3.5.6 CELP LPC decoder and interpolator 18

3.5.7 CELP excitation generator 35

3.5.8 CELP LPC synthesis filter 54

Annex 3.A (informative) **MPEG-4 CELP decoder tools** 55

3.A.1 CELP post-processor 55

Annex 3.B (informative) **MPEG-4 CELP encoder tools** 58

3.B.1 General Introduction to the MPEG-4 CELP encoder tool-set 58

3.B.2 Helping variables 58

3.B.3 Bistream elements for the MPEG-4 CELP encoder tool-set 60

3.B.4 CELP preprocessing 60

3.B.5 CELP LPC analysis 61

3.B.6 CELP LPC quantizer and interpolator 62

3.B.7 CELP LPC analysis filter 70

3.B.8 CELP weighting module 70

3.B.9 CELP excitation analysis 71

3.B.10 CELP bitstream multiplexer **84**

Annex 3.C (normative) **Tables** **85**

3.C.1 LSP VQ tables and gain VQ tables for 8 kHz sampling rate **85**

3.C.2 LSP VQ tables and gain VQ tables for the 16 kHz sampling rate **91**

3.C.3 Gain tables for the bitrate scalable tool **103**

3.C.4 LSP VQ tables and gain VQ tables for the bandwidth scalable tool **104**

Annex 3.D (informative) **Tables** **112**

3.D.1 Bandwidth expansion tables in LPC analysis of the mode II coder **112**

3.D.2 Downsampling filter coefficients for the bandwidth scalable tool **112**

Annex 3.E (informative) **Example of a simple CELP transport stream** **113**

Annex 3.F (informative) **Random access points** **115**

Subpart 3: Speech Coding - CELP

3.1 Scope

3.1.1 General description of the CELP decoder

This subclause provides a brief overview of the CELP (Code Excited Linear Prediction) decoder. A basic block diagram of the CELP decoder is given in Figure 3.1.

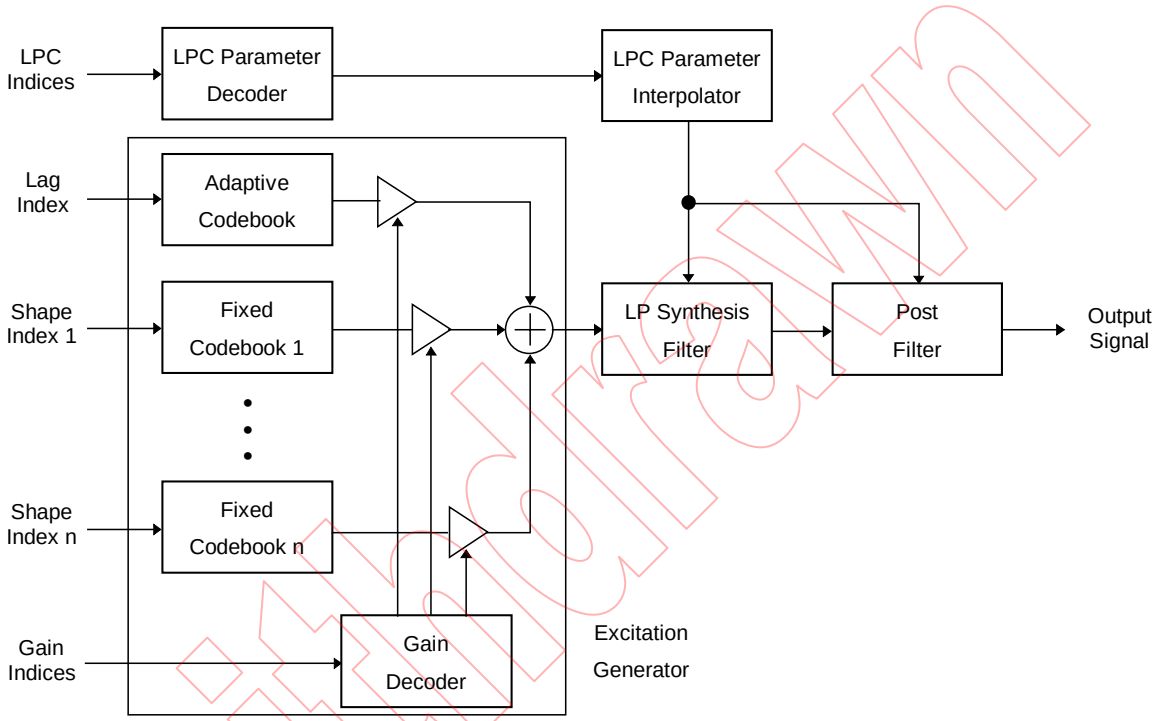

Figure 3.1 – Block diagram of a CELP decoder

The CELP decoder primarily consists of an excitation generator and a synthesis filter. Additionally, CELP decoders often include a post-filter. The excitation generator has an adaptive codebook to model periodic components, fixed codebooks to model random components and a gain decoder to represent a speech signal level. Indices for the codebooks and gains are provided by the encoder. The codebook indices (pitch-lag index for the adaptive codebook and shape index for the fixed codebook) and gain indices (adaptive and fixed codebook gains) are used to generate the excitation signal. It is then filtered by the linear predictive synthesis filter (LP synthesis filter). Filter coefficients are reconstructed using the LPC indices, then are interpolated with the filter coefficients of successive analysis frames. Finally, a post-filter can optionally be applied in order to enhance the speech quality.

3.1.2 Functionality of MPEG-4 CELP

MPEG-4 CELP is a generic coding algorithm with new functionalities. Conventional CELP coders offer compression at a single bitrate and are optimized for specific applications. Compression is one of the functions provided by MPEG-4 CELP, enabling the use of one basic coder for various applications. It provides scalability in bitrate and bandwidth, as well as the ability to generate bitstreams at arbitrary bitrates. The MPEG-4 CELP coder supports two sampling rates, namely, 8 and 16 kHz. The associated bandwidths are 100 – 3400 Hz for 8 kHz sampling rate and 50 – 7000 Hz for 16 kHz sampling rate.

3.1.2.1 Configuration of the MPEG-4 CELP coder

Two different tools can be used to generate the excitation signal. These are the Multi-Pulse Excitation (MPE) tool or

the Regular-Pulse Excitation (RPE) tool. MPE is used for speech sampled at 8 kHz or 16 kHz. RPE is only used for speech sampled at 16 kHz. The two possible coding modes are summarized in Table 3.1.

Table 3.1 – Coding modes in the MPEG-4 CELP coder

Coding Mode	Excitation tool	Sampling rate
I	RPE	16 kHz
II	MPE	8, 16 kHz

3.1.2.2 Features of the MPEG-4 CELP coder

The MPEG-4 CELP coder offers the following functionality, depending on the coding mode.

Table 3.2 – Functionality of the MPEG-4 CELP coder

Coding Mode	Functionality
I	Multiple bitrates, FineRate Control
II	Multiple bitrates, Bitrate Scalability, Bandwidth Scalability, FineRate Control

Multiple bitrates: The available bitrates depend on the coding mode and the sampling rate. The following fixed bitrates are supported:

Table 3.3 – Fixed bitrates for the mode I coder

Bitrates for the 16 kHz sampling rate (bit/s)
14400, 16000, 18667, 22533

Table 3.4 – Fixed bitrates for the mode II coder

Bitrates for the 8 kHz sampling rate (bit/s)	Bitrates for the 16 kHz sampling rate (bit/s)
3850, 4250, 4650, 4900, 5200, 5500, 5700, 6000, 6200, 6300, 6600, 6900, 7100, 7300, 7700, 8300, 8700, 9100, 9500, 9900, 10300, 10500, 10700, 11000, 11400, 11800, 12000, 12200	10900, 11500, 12100, 12700, 13300, 13900, 14300, 14700, 15900, 17100, 17900, 18700, 19500, 20300, 21100, 13600, 14200, 14800, 15400, 16000, 16600, 17000, 17400, 18600, 19800, 20600, 21400, 22200, 23000, 23800

Fine Rate Control: Enables fine step bitrate control (permitting variable bitrate operation). This is achieved purely by controlling the transmission rate of the LPC parameters using a combinations of the two bitstream elements **interpolation_flag** and **LPC_present** flag. Using FineRate Control it is possible to vary the ratio of LPC-frames to total frames between 50% and 100%. This enables the bitrate to be decreased with respect to the anchor bitrate, as defined in the Semantics.

Bitrate Scalability: Bitrate scalability is provided by adding enhancement layers. Enhancement layers can be added with a step of 2000 bit/s for signals sampled at 8 kHz or 4000 bit/s for signals sampled at 16 kHz. A maximum of three enhancement layers may be combined with any bitrate chosen from Table 3.4.

Bandwidth Scalability: Bandwidth scalability to cover both sampling rates is achieved by incorporating a bandwidth extension tool in the CELP coder. This is an enhancement tool, supported in Mode II, which may be added if scalability from the 8 kHz sampling rate to the 16 kHz sampling rate is required. A complete coder with bandwidth scalability consists of a core CELP coder for the 8 kHz sampling rate and the bandwidth extension tool to provide a single layer of scalability. The core CELP coder for the 8 kHz sampling rate can comprise several layers. It should be noted that an 8 kHz sampling rate coder with this tool is not the same as a 16 kHz sampling rate coder. Both configurations (8 kHz sampling rate coder with bandwidth scalability and 16 kHz sampling rate coder) offer greater intelligibility and naturalness of decoded speech than does the 8 kHz coder alone because they expand the bandwidth to 7 kHz. The additional bitrate required for the bandwidth scalability tool can be selected from 4 discrete steps for each core layer bitrate as shown in Table 3.5.

Table 3.5 – Bitrates for the bandwidth scalable mode

Bitrate of the core layer (bit/s)	Additional bitrate (bit/s)
3850 - 4650	+9200, +10400, +11600, +12400
4900 - 5500	+9467, +10667, +11867, +12667
5700 - 10700	+10000, +11200, +12400, +13200
11000 - 12200	+11600, +12800, +14000, +14800

3.1.2.3 Algorithmic delay of MPEG-4 CELP modes

The algorithmic delay of the CELP coder comes from the frame length and an additional look ahead length. The frame length depends on the coding mode and the bitrate. The look ahead length, which is an informative parameter, also depends on the coding mode. The delays presented below are applicable to the modes where FineRate Control is off. When FineRate Control is on, additional one-frame delay is introduced. Bandwidth scalability in the mode II coder requires an additional look ahead of 5 ms due to down-sampling.

Table 3.6 – Delay and frame length for the mode I coder of the 16 kHz sampling rate

Bitrate for Mode I (bit/s)	Delay (ms)	Frame Length (ms)
14400	26.25	15
16000	18.75	10
18667	26.56	15
22533	26.75	15

Table 3.7 – Delay and frame length for the mode II coder of the 8 kHz sampling rate

Bitrate for Mode II (bit/s)	Delay (ms)	Frame Length (ms)
3850, 4250, 4650	45	40
4900, 5200, 5500, 6200	35	30
5700, 6000, 6300, 6600, 6900, 7100, 7300, 7700, 8300, 8700, 9100, 9500, 9900, 10300, 10500, 10700	25	20
11000, 11400, 11800, 12000, 12200	15	10

Table 3.8 – Delay and frame length for the mode II coder of the 16 kHz sampling rate

Bitrate for Mode II (bit/s)	Delay (ms)	Frame Length (ms)
10900, 11500, 12100, 12700, 13300, 13900, 14300, 14700, 15900, 17100, 17900, 18700, 19500, 20300, 21100	25	20
13600, 14200, 14800, 15400, 16000, 16600, 17000, 17400, 18600, 19800, 20600, 21400, 22200, 23000, 23800	15	10

Contents for Subpart 4

4.1 Scope 5

4.1.1 Technical Overview 5

4.1.1.1 Encoder and Decoder Block Diagrams 5

4.1.1.2 Overview of the Encoder and Decoder Tools 8

4.2 Normative References 11

4.3 GA-specific definitions 11

4.4 Syntax 13

4.4.1 GA Specific Configuration 13

4.4.1.1 Program config element 14

4.4.2 GA Bitstream Payloads 15

4.4.2.1 Payloads for the audio object types AAC_main, AAC_SSR, AAC_LC and AAC_LTP 15

4.4.2.2 Payloads for the audio object type AAC_scalable 19

4.4.2.3 Payloads for the audio object type Twin_VQ 22

4.4.2.4 Subsidiary payloads 24

4.5 General information 29

4.5.1 Decoding of the GA specific configuration 29

4.5.1.1 GA_SpecificConfig 29

4.5.1.2 Program Config Element (PCE) 29

4.5.2 Decoding of the GA bitstream payloads 31

4.5.2.1 Top Level Payloads for the audio object types AAC_main, AAC_SSR, AAC_LC and AAC_LTP 31

4.5.2.2 Payloads for the audio object type AAC_scalable 36

4.5.2.3 Decoding of an individual_channel_stream (ICS) and ics_info 53

4.5.2.4 Payloads for the audio object type TwinVQ 58

4.5.2.5 Dynamic Range Control (DRC) 61

4.5.3 Buffer requirements 64

4.5.3.1 Minimum decoder input buffer 64

4.5.3.2 Bit reservoir 64

4.5.3.3 Maximum bit rate 65

4.5.4 Tables 65

4.5.5 Figures 71

4.6 GA-Tool Descriptions 72

4.6.1 Quantization 72
4.6.1.1 Tool description 72
4.6.1.2 Definitions 72
4.6.1.3 Decoding process 72
4.6.2 Scalefactors 72
4.6.2.1 Tool description 72
4.6.2.2 Definitions 72
4.6.2.3 Decoding process 73
4.6.3 Noiseless coding 74
4.6.3.1 Tool description 74
4.6.3.2 Definitions 74
4.6.3.3 Decoding process 76
4.6.3.4 Tables 78
4.6.4 Interleaved vector quantization 79
4.6.4.1 Tool description 79
4.6.4.2 Definitions 79
4.6.4.3 Parameter settings 79
4.6.4.4 Decoding process 79
4.6.4.5 Diagrams 82
4.6.5 Frequency domain prediction 83
4.6.5.1 Tool description 83
4.6.5.2 Definitions 83
4.6.5.3 Decoding process 84
4.6.5.4 Diagrams 89
4.6.6 Long Term Prediction (LTP) 90
4.6.6.1 Tool description 90
4.6.6.2 Definitions 90
4.6.6.3 Decoding process 90
4.6.6.4 Integration of LTP with other GA tools 91
4.6.6.5 LTP in a scalable GA decoder 92
4.6.7 Joint Coding 92
4.6.7.1 M/S stereo 92
2 Subpart 4

4.6.7.2 Intensity Stereo (IS) 93
4.6.7.3 Coupling channel 95
4.6.8 Temporal Noise Shaping (TNS) 98
4.6.8.1 Tool description 98
4.6.8.2 Definitions 98
4.6.8.3 Decoding process 98
4.6.8.4 Maximum TNS order and bandwidth 100
4.6.8.5 TNS in the scalable coder 100
4.6.9 Spectrum normalization 102
4.6.9.1 Tool description 102
4.6.9.2 Definitions 102
4.6.9.3 Decoding process 103
4.6.9.4 Diagrams 109
4.6.9.5 Tables 110
4.6.10 Filterbank and block switching 111
4.6.10.1 Tool description 111
4.6.10.2 Definitions 111
4.6.10.3 Decoding process 112
4.6.11 Gain Control 116
4.6.11.1 Tool description 116
4.6.11.2 Definitions 117
4.6.11.3 Decoding process 117
4.6.11.4 Diagrams 121
4.6.11.5 Tables 122
4.6.12 Perceptual Noise Substitution (PNS) 123
4.6.12.1 Tool description 123
4.6.12.2 Definitions 123
4.6.12.3 Decoding process 123
4.6.12.4 Integration with the intra channel prediction tools 124
4.6.12.5 Integration with other AAC tools 125
4.6.12.6 Integration into a scalable AAC-based coder (AudioObjectType AAC_scalable) 125
4.6.13 Frequency Selective Switch (FSS) Module 125

4.6.13.1 FSS in combined TwinVQ /CELP- AAC systems:....125
4.6.13.2 FSS in combined mono / stereo scalable configurations....127
4.6.14 Upsampling filter tool....127
4.6.14.1 Tool description....127
4.6.14.2 Definitions....128
4.6.14.3 Decoding process....128
Annex 4.A (normative) Normative Tables....130
4.A.1 Huffman codebook tables for AAC-type noisless coding....130
4.A.2 Window tables....143
4.A.3 Differential scalefactor to index tables....145
4.A.4 Tables for TwinVQ....146
Annex 4.B (informative) Encoder tools....163
4.B.1 Weighted interleave vector quantization....163
4.B.2 Spectrum normalization....165
4.B.3 Psychoacoustic model....169
4.B.4 Gain control....199
4.B.5 Filterbank and block switching....200
4.B.6 Frequency domain prediction....206
4.B.7 Long Term Prediction....209
4.B.8 Temporal Noise Shaping (TNS)....211
4.B.9 Joint coding....213
4.B.10 Quantization....214
4.B.11 Noiseless coding....220
4.B.12 Perceptual Noise Substitution (PNS)....222
4.B.13 Random access points for GA coded bit streams (ObjectTypes 0x1 to 0x7)....223
4.B.14 Scalable AAC with core coder....223
4.B.15 Scalable controller....225
4.B.16 Features of AAC dynamic range control....225

Subpart 4: General Audio (GA) Coding: AAC/TwinVQ

4.1 Scope

The General Audio (GA) coding subpart of MPEG-4 Audio is mainly intended to be used for generic audio coding at all but the lowest bitrates. Typically, GA encoding is used for complex music material in mono from 6 kbit/s per channel and for stereo signals from 12 kbit/s per stereo signal up to broadcast quality audio at 64 kbit/s or more per channel. MPEG-4 coded material can be represented either by a single set of data, like in MPEG-1 and MPEG-2 Audio, or by several subsets which allow the decoding at different quality levels, depending on the number of subsets being available at the decoder side (bitrate scalability).

MPEG-2 Advanced Audio Coding (AAC) syntax (including support for multi-channel audio) is fully supported by MPEG-4 Audio GA coding. All the features and possibilities of the MPEG-2 AAC standard also apply to MPEG-4. AAC has been tested to allow for ITU-R 'indistinguishable' quality according to [4] at data rates of 320 kb/s for five full-bandwidth channel audio signals. In MPEG-4 the tools derived from MPEG-2 AAC are available together with other MPEG-4 GA coding tools which provide additional functionalities, like bit rate scalability and improved coding efficiency at very low bit rates. Bit rate scalability is either achieved with only GA coding tools, or by using a combination with an external (non-GA, e.g. CELP) core coder.

MPEG-4 GA coding is not restricted to some fixed bitrates but supports a wide range of bitrates and variable rate coding. While efficient mono, stereo and multi-channel coding is possible using extended, MPEG-2 AAC derived tools, the document also provides extensions to this tool set which allow mono/stereo scalability, where a mono signal can be extracted by decoding only subsets of the encoded stereo stream.

4.1.1 Technical Overview

4.1.1.1 Encoder and Decoder Block Diagrams

The block diagrams of the GA encoder and decoder reflect the structure of MPEG-4 GA coding. In general, there are the MPEG-2 AAC related tools with MPEG-4 add-ons for some of them and the tools related to the Twin-VQ quantization and coding. The Twin-VQ is an alternative module for the AAC-type quantization and it is based on an interleaved vector quantization and LPC (Linear Predictive Coding) spectral estimation. It operates from 6 kbit/s/ch and is recommended to be used below 16 kbit/s/ch with constant bitrate.

The basic structure of the MPEG-4 GA system is shown in Figures 4.1.1 and 4.1.2. The data flow in this diagram is from left to right, top to bottom. The functions of the decoder are to find the description of the quantized audio spectra in the bitstream, decode the quantized values and other reconstruction information, reconstruct the quantized spectra, process the reconstructed spectra through whatever tools are active in the bitstream in order to arrive at the actual signal spectra as described by the input bitstream, and finally convert the frequency domain spectra to the time domain, with or without an optional gain control tool. Following the initial reconstruction and scaling of the spectrum reconstruction, there are many optional tools that modify one or more of the spectra in order to provide more efficient coding. For each of the optional tools that operate in the spectral domain, the option to "pass through" is retained, and in all cases where a spectral operation is omitted, the spectra at its input are passed directly through the tool without modification.

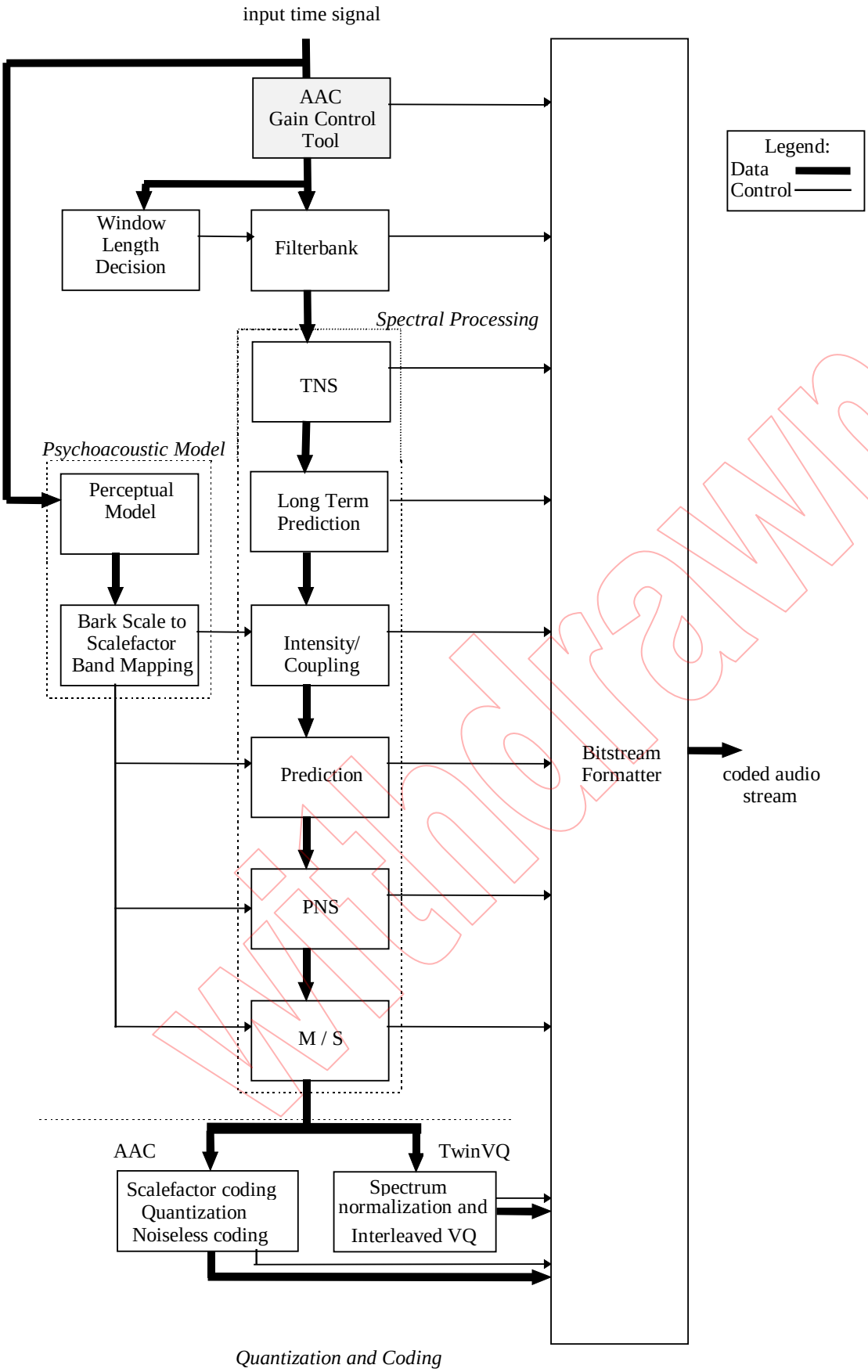

Figure 4.1.1 - Block diagram GA non scalable encoder

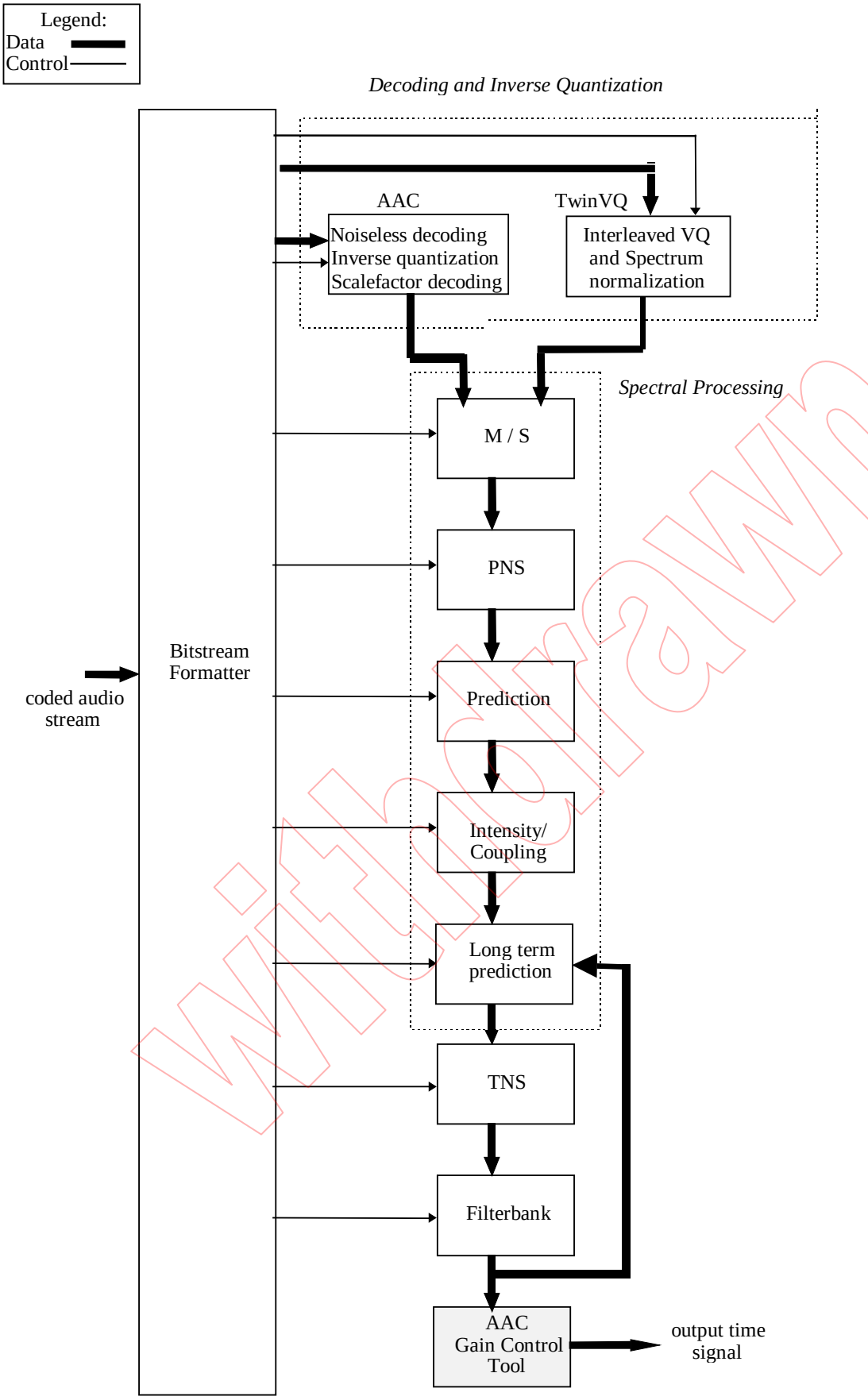

Figure 4.1.2 - Block diagram of the GA non scalable decoder

4.1.1.2 Overview of the Encoder and Decoder Tools

The input to the bitstream demultiplexer tool is the MPEG-4 GA bitstream. The demultiplexer separates the bitstream into the parts for each tool, and provides each of the tools with the bitstream information related to that tool.
The outputs from the bitstream demultiplexer tool are:

- The quantized (and optionally noiselessly coded) spectra represented by either
  - the sectioning information and the noiselessly coded spectra (AAC) or
  - a set of indices of code vectors (TwinVQ)
- The M/S decision information (optional)
- The predictor side information (optional)
- The perceptual noise substitution (PNS) information (optional)
- The intensity stereo control information and coupling channel control information (both optional)
- The temporal noise shaping (TNS) information (optional)
- The filterbank control information
- The gain control information (optional)
- Bitrate scalability related side information (optional)

The AAC noiseless decoding tool takes information from the bitstream demultiplexer, parses that information, decodes the Huffman coded data, and reconstructs the quantized spectra and the Huffman and DPCM coded scalefactors.
The inputs to the noiseless decoding tool are:

- The sectioning information for the noiselessly coded spectra
- The noiselessly coded spectra

The outputs of the noiseless decoding tool are:

- The decoded integer representation of the scalefactors:
- The quantized values for the spectra

The inverse quantizer tool takes the quantized values for the spectra, and converts the integer values to the non-scaled, reconstructed spectra. This quantizer is a non-uniform quantizer.
The input to the Inverse Quantizer tool is:

- The quantized values for the spectra

The output of the inverse quantizer tool is:

- The un-scaled, inversely quantized spectra

The scalefactor tool converts the integer representation of the scalefactors to the actual values, and multiplies the un-scaled inversely quantized spectra by the relevant scalefactors.
The inputs to the scalefactors tool are:

- The decoded integer representation of the scalefactors
- The un-scaled, inversely quantized spectra

The output from the scalefactors tool is:

- The scaled, inversely quantized spectra

The M/S tool converts spectra pairs from Mid/Side to Left/Right under control of the M/S decision information, improving stereo imaging quality and sometimes providing coding efficiency.
The inputs to the M/S tool are:

- The M/S decision information
- The scaled, inversely quantized spectra related to pairs of channels

The output from the M/S tool is:

- The scaled, inversely quantized spectra related to pairs of channels, after M/S decoding

Note: The scaled, inversely quantized spectra of individually coded channels are not processed by the M/S block, rather they are passed directly through the block without modification. If the M/S block is not active, all spectra are passed through this block unmodified.

The prediction tool reverses the prediction process carried out at the encoder. This prediction process re-inserts the redundancy that was extracted by the prediction tool at the encoder, under the control of the predictor state

information. This tool is implemented as a second order backward adaptive predictor. The inputs to the prediction tool are:

- The predictor state information
- The predictor side information
- The scaled, inversely quantized spectra

The output from the prediction tool is:

- The scaled, inversely quantized spectra, after prediction is applied.

Note: If the prediction is disabled, the scaled, inversely quantized spectra are passed directly through the block without modification.

Alternatively, there is a forward adaptive long term prediction tool provided. The inputs to the long term prediction tool are:

- The reconstructed time domain output of the decoder
- The scaled, inversely quantized spectra

The output from the long term prediction tool is:

- The scaled, inversely quantized spectra, after prediction is applied.

Note: If the prediction is disabled, the scaled, inversely quantized spectra are passed directly through the block without modification.

The perceptual noise substitution (PNS) tool implements noise substitution decoding on channel spectra by providing an efficient representation for noise-like signal components.
The inputs to the perceptual noise substitution tool are:

- The inversely quantized spectra
- The perceptual noise substitution control information

The output from the perceptual noise substitution tool is:

- The inversely quantized spectra

Note: If either part of this block is disabled, the scaled, inversely quantized spectra are passed directly through this part without modification. If the perceptual noise substitution block is not active, all spectra are passed through this block unmodified.

The intensity stereo / coupling tool implements intensity stereo decoding on pairs of spectra. In addition, it adds the relevant data from a dependently switched coupling channel to the spectra at this point, as directed by the coupling control information.
The inputs to the intensity stereo / coupling tool are:

- The inversely quantized spectra
- The intensity stereo control information and coupling control information

The output from the intensity stereo / coupling tool is:

- The inversely quantized spectra after intensity and coupling channel decoding.

Note: If either part of this block is disabled, the scaled, inversely quantized spectra are passed directly through this part without modification. The intensity stereo tool and M/S tools are arranged so that the operation of M/S and Intensity stereo are mutually exclusive on any given scalefactor band and group of one pair of spectra.

The temporal noise shaping (TNS) tool implements a control of the fine time structure of the coding noise. In the encoder, the TNS process has flattened the temporal envelope of the signal to which it has been applied. In the decoder, the inverse process is used to restore the actual temporal envelope(s), under control of the TNS information. This is done by applying a filtering process to parts of the spectral data.
The inputs to the TNS tool are:

- The inversely quantized spectra
- The TNS information

The output from the TNS block is:

- The inversely quantized spectra

Note: If this block is disabled, the inversely quantized spectra are passed through without modification.

The filterbank tool applies the inverse of the frequency mapping that was carried out in the encoder, as indicated by the filterbank control information and the presence or absence of gain control information. An inverse modified discrete cosine transform (IMDCT) is used for the filterbank tool. If the gain control tool is not used, the IMDCT input consists of either 1024 or 128 (depending on **window_sequence**) spectral coefficients (if **frameLengthFlag** is set to '0'), or of 960 or 120 spectral coefficients (if **frameLengthFlag** is set to '1'), respectively. If the gain control tool is used, the filterbank tool is configured to use four sets of either 256 or 32 coefficients, depending of the value of window_sequence.
The inputs to the filterbank tool are:

- The inversely quantized spectra

- The filterbank control information

The output(s) from the filterbank tool is (are):

- The time domain reconstructed audio signal(s).

Two alternative, but very similar versions of this tool are available. The version with a frame length of 960 samples, which is not available in ISO/IEC 13818-7, allows for an integer frame length. For example at 48 kHz sampling rate, the frame length is exactly 20 ms with this version. This is especially useful for the CELP/AAC bitrate scalability combinations, where this allows the construction of combined CELP layer frames, which have a length of a multiple of 10 ms, and AAC enhancement layer frames. However, this feature can be used for configurations with only AAC or TwinVQ coding as well.

When present, the gain control tool applies a separate time domain gain control to each of 4 frequency bands that have been created by the gain control PQF filterbank in the encoder. Then, it assembles the 4 frequency bands and reconstructs the time waveform through the gain control tool's filterbank.
The inputs to the gain control tool are:

- The time domain reconstructed audio signal(s)
- The gain control information

The output(s) from the gain control tool is (are):

- The time domain reconstructed audio signal(s)

If the gain control tool is not active, the time domain reconstructed audio signal(s) are passed directly from the filterbank tool to the output of the decoder. This tool is used for the scalable sampling rate (SSR) object type only.

The spectrum normalization tool converts the reconstructed flat spectra to the actual values at the decoder. The spectral envelope is specified by LPC coefficients, a Bark scale envelope, periodic peak components, and gain.
The input to the spectral normalization tool are

- The reconstructed flat spectra
- The information of LPC coefficients, a Bark scale envelope, periodic peak components and gain

The output from the spectral normalization tool is

- The reconstructed actual spectra

The interleaved VQ tool converts the vector index to the flattened spectra at the TwinVQ decoder
by means of table look-up of the codebook and inverse interleaving of the spectra. Quantization noise is minimized by a weighted distortion measure at the encoder instead of an adaptive bit allocation. This is an alternative to the AAC quantization tool.
The input to the interleaved VQ tool is:

- A set of indices of the code vector.

The output from the TwinVQ tool is:

- The reconstructed flattened spectra

The Frequency Selective Switch (FSS) tool is used to control the combination of the AAC coding layer with both, TwinVQ, and CELP coding layer, if these are used as base layer coder in scalable configurations. In a second function this tool is applied to control the combination of mono and stereo coding layer in scalable configurations where both mono, and stereo coding layer are used to code a stereo input signal.

The Up-sampling Filter tool adapts the sampling rate of a CELP core coder, which can be used as base layer coder in scalable configurations, to the sampling rate of the AAC extension layer.
The input to the Upsampling Filter tool is:

- The output of a CELP core coder running at a lower sampling rate than the AAC extension layer

The output from the Up-sampling Filter tool is:

- The up-sampled CELP core coder output, matching the sampling rate of the AAC extension layer, transformed into the frequency domain with exactly the same frequency and time resolution as the AAC extension layer.

4.2 Normative References

[1] 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*.

[2] ITU-T Rec.H.222.0(1995) | ISO/IEC 13818-1:1996, *Information technology - Generic coding of moving pictures and associated audio information: – Part 1: Systems.*

[3] ISO/IEC 13818-3:1997, *Information technology - Generic coding of moving pictures and associated audio information: - Part 3: Audio*.

[4] ISO/IEC 13818-7:1997, *Information technology - Generic coding of moving pictures and associated audio information: - Part 7: Advanced Audio Coding (AAC).*

Contents for Subpart 5

5.1 Scope 8
5.1.1 Overview of section 8
5.1.1.1 Purpose 8
5.1.1.2 Introduction to major elements 8
5.2 Normative references 8
5.3 Definitions 8
5.4 Symbols and abbreviations 13
5.4.1 Mathematical operations 13
5.4.2 Description methods 13
5.4.2.1 Bitstream syntax 13
5.4.2.2 SAOL syntax 14
5.4.2.3 SASL Syntax 14
5.5 Bitstream syntax and semantics 14
5.5.1 Introduction to bitstream syntax 14
5.5.2 Bitstream syntax 14
5.6 Object types 19
5.7 Decoding process 20
5.7.1 Introduction 20
5.7.2 Decoder configuration header 20
5.7.3 Bitstream data and sound creation 20
5.7.3.1 Relationship with systems layer 20
5.7.3.2 Bitstream data elements 20
5.7.3.3 Scheduler semantics 21
5.7.4 Conformance 25
5.8 SAOL syntax and semantics 26
5.8.1 Relationship with bitstream syntax 26
5.8.2 Lexical elements 26
5.8.2.1 Concepts 26
5.8.2.2 Identifiers 27
5.8.2.3 Numbers 27
5.8.2.4 String constants 27
5.8.2.5 Comments 27
5.8.2.6 Whitespace 28
5.8.3 Variables and values 28
5.8.4 Orchestra 28
5.8.5 Global block 29
5.8.5.1 Syntactic form 29
5.8.5.2 Global parameter 29

5.8.5.3 Global variable declaration 31
5.8.5.4 Route statement 32
5.8.5.5 Send statement 33
5.8.5.6 Sequence specification 34

5.8.6 Instrument definition 36
5.8.6.1 Syntactic form 36
5.8.6.2 Instrument name 36
5.8.6.3 Parameter fields 36
5.8.6.4 Preset tag 36
5.8.6.5 Instrument variable declarations 37
5.8.6.6 Block of code statements 39
5.8.6.7 Expressions 46
5.8.6.8 Standard names 54

5.8.7 Opcode definition 58
5.8.7.1 Syntactic Form 58
5.8.7.2 Rate tag 58
5.8.7.3 Opcode name 58
5.8.7.4 Formal parameter list 59
5.8.7.5 Opcode variable declarations 59
5.8.7.6 Opcode statement block 60
5.8.7.7 Opcode rate 60

5.8.8 Template declaration 62
5.8.8.1 Syntactic form 62
5.8.8.2 Semantics 62
5.8.8.3 Template instrument definitions 62

5.8.9 Reserved words 63

5.9 SAOL core opcode definitions and semantics 64

5.9.1 Introduction 64

5.9.2 Specialop type 64

5.9.3 List of core opcodes 65

5.9.4 Math functions 66
5.9.4.1 Introduction 66
5.9.4.2 int 66
5.9.4.3 frac 66
5.9.4.4 dbamp 66
5.9.4.5 ampdb 66
5.9.4.6 abs 66
5.9.4.7 sgn 66
5.9.4.8 exp 66
5.9.4.9 log 67
5.9.4.10 sqrt 67
5.9.4.11 sin 67
5.9.4.12 cos 67
5.9.4.13 atan 67
5.9.4.14 pow 67
5.9.4.15 log10 68
5.9.4.16 asin 68
5.9.4.17 acos 68
5.9.4.18 ceil 68
5.9.4.19 floor 68
5.9.4.20 min 68
5.9.4.21 max 68

5.9.5 Pitch converters 69
5.9.5.1 Introduction to pitch representations 69
5.9.5.2 gettune 69
5.9.5.3 settune 69

5.9.5.4 octpch 70
5.9.5.5 pchoct 70
5.9.5.6 cpspch 70
5.9.5.7 pchcps 70
5.9.5.8 cpsoct 71
5.9.5.9 octcps 71
5.9.5.10 midipch 71
5.9.5.11 pchmidi 71
5.9.5.12 midioct 71
5.9.5.13 octmidi 72
5.9.5.14 midicps 72
5.9.5.15 cpsmidi 72

5.9.6 Table operations 72
5.9.6.1 ftlen 72
5.9.6.2 ftloop 72
5.9.6.3 ftloopend 73
5.9.6.4 ftsr 73
5.9.6.5 ftbasecps 73
5.9.6.6 ftsetloop 73
5.9.6.7 ftsetend 73
5.9.6.8 ftsetbase 73
5.9.6.9 ftsetsr 74
5.9.6.10 tableread 74
5.9.6.11 tablewrite 74
5.9.6.12 oscil 74
5.9.6.13 loscil 75
5.9.6.14 doscil 75
5.9.6.15 koscil 76

5.9.7 Signal generators 76
5.9.7.1 kline 76
5.9.7.2 aline 77
5.9.7.3 kexpon 77
5.9.7.4 aexpon 78
5.9.7.5 kphasor 78
5.9.7.6 aphasor 79
5.9.7.7 pluck 79
5.9.7.8 buzz 80
5.9.7.9 grain 80

5.9.8 Noise generators 81
5.9.8.1 Note on noise generators and pseudo-random sequences 81
5.9.8.2 irand 82
5.9.8.3 krand 82
5.9.8.4 arand 82
5.9.8.5 ilinrand 82
5.9.8.6 klinrand 82
5.9.8.7 alinrand 83
5.9.8.8 iexprand 83
5.9.8.9 kexprand 83
5.9.8.10 aexprand 83
5.9.8.11 kpoissonrand 83
5.9.8.12 apoissonrand 84
5.9.8.13 igaussrand 84
5.9.8.14 kgaussrand 85
5.9.8.15 agaussrand 85

5.9.9 Filters 85
5.9.9.1 port 85
5.9.9.2 hipass 85
5.9.9.3 lopass 86
5.9.9.4 bandpass 86
5.9.9.5 bandstop 86
5.9.9.6 biquad 87

5.9.9.7 allpass 87
5.9.9.8 comb 87
5.9.9.9 fir 88
5.9.9.10 iir 88
5.9.9.11 firt 88
5.9.9.12 iirt 89

5.9.10 Spectral analysis 89
5.9.10.1 fft 89
5.9.10.2 ifft 90

5.9.11 Gain control 91
5.9.11.1 rms 91
5.9.11.2 gain 92
5.9.11.3 balance 92
5.9.11.4 compressor 93

5.9.12 Sample conversion 95
5.9.12.1 decimate 95
5.9.12.2 upsamp 95
5.9.12.3 downsamp 96
5.9.12.4 samphold 96
5.9.12.5 sblock 96

5.9.13 Delays 97
5.9.13.1 delay 97
5.9.13.2 delay1 97
5.9.13.3 fracdelay 97

5.9.14 Effects 98
5.9.14.1 reverb 98
5.9.14.2 chorus 99
5.9.14.3 flange 99
5.9.14.4 fx_speedc 99
5.9.14.5 speedt 99

5.9.15 Tempo functions 100
5.9.15.1 gettempo 100
5.9.15.2 settempo 100

5.10 SAOL core wavetable generators 100
5.10.1 Introduction 100
5.10.2 Sample 100
5.10.3 Data 101
5.10.4 Random 101
5.10.5 Step 102
5.10.6 Lineseg 103
5.10.7 Expseg 103
5.10.8 Cubicseg 104
5.10.9 Spline 104
5.10.10 Polynomial 105
5.10.11 Window 105
5.10.12 Harm 106
5.10.13 Harm_phase 106
5.10.14 Periodic 106
5.10.15 Buzz 107
5.10.16 Concat 107

5.10.17 Empty 107

5.11 SASL syntax and semantics 108
5.11.1 Introduction 108
5.11.2 Syntactic form 108
5.11.3 Instr line 109
5.11.4 Control line 109
5.11.5 Tempo line 109
5.11.6 Table line 110
5.11.7 End line 110

5.12 SAOL/SASL tokenisation 110
5.12.1 Introduction 110
5.12.2 SAOL tokenisation 111
5.12.3 SASL tokenisation 111

5.13 Sample Bank syntax and semantics 112
5.13.1 Introduction 112
5.13.2 Elements of bitstream 112
5.13.3 Decoding process 112
5.13.3.1 Object type 2 112
5.13.3.2 Object type 4 113

5.14 MIDI semantics 113
5.14.1 Introduction 113
5.14.2 Object type 1 decoding process 114
5.14.3 Mapping MIDI events into orchestra control 114
5.14.3.1 Introduction 114
5.14.3.2 MIDI events 114
5.14.3.3 Standard MIDI Files 116
5.14.3.4 Default controller values 117

5.15 Input sounds and relationship with AudioBIFS 117
5.15.1 Introduction 117
5.15.2 Input sources and phaseGroup 118
5.15.3 The AudioFX node 118
5.15.3.1 Introduction 118
5.15.3.2 AudioFX orchestra parameters 118
5.15.3.3 AudioFX orchestra instantiation 119
5.15.3.4 AudioFX orchestra execution 119
5.15.3.5 Speed change functionality in the AudioFX node 119
5.15.4 Interactive 3-D spatial audio scenes 119

Annex 5.A (normative) Coding tables 120

Annex 5.B (informative) Encoding 123
5.B.1. Introduction 123

5.B.2. Basic encoding **123**
5.B.2.1. Introduction **123**
5.B.2.2. Tokenisation of SAOL data **123**
5.B.2.3. Tokenisation of SASL data **123**
5.B.2.4. Disassembly of sound samples **123**
5.B.2.5 Assembly of decoder configuration information **124**
5.B.2.6 Assembly of streaming bitstream **124**

Annex 5.C (informative) **lex/yacc grammars for SAOL** **125**
5.C.1 Introduction **125**
5.C.2 Lexical grammar for SAOL in lex **125**
5.C.3 Syntactic grammar for SAOL in yacc **127**

Annex 5.D (informative) **PICOLA Speed change algorithm** **131**
5.D.1 Tool description **131**
5.D.2 Speed control process **131**
5.D.3 Time scale compression (High speed replay) **131**
5.D.4 Time scale expansion (Low speed replay) **132**

Annex 5.E (informative) **Random access to Structured audio bitstreams** **134**
5.E.1 Introduction **134**
5.E.2 Difficulties in general-purpose random access **134**
5.E.3 Making Structured Audio bitstreams randomly-accessible **135**
5.E.3.1 Introduction **135**
5.E.3.2 Constructs to avoid **135**
5.E.3.3 Altering bitstreams to make them randomly accessible **135**

Annex 5.F (informative) **Directly-connected MIDI and microphone control of the orchestra** **139**
5.F.1 Introduction **139**
5.F.2 MIDI controller recommended practices **139**
5.F.3 Live microphone recommended practices **140**

Annex 5.G (informative) **Bibliography** **141**

Alphabetical Index to Subpart 5 of ISO/IEC 14496-3 ..... **142**

Figures

Figure 5.1 - Example of ordering instruments with ‘sequence’ 35
Figure 5.2 - Example of ordering instruments with ‘sequence’ 35
Figure 5.3 - Compressor characteristic function 94
Figure 5.4 - Block diagram for ‘fracdelay’ example 98
Figure 5.D.1 - Block Diagram of the Speed Controller 131
Figure 5.D.2 - Principle of Time Scale Compression 132
Figure 5.D.3 - Principle of Time Scale Expansion 133

Tables

Table 5.1 - Example of calculating bus routing values 33
Table 5.2 - Binary operators 51
Table 5.3 - Order of operations 52
Table 5.4 - Default MIDI Controller Values 117

Subpart 5: Structured audio

5.1 Scope

5.1.1 Overview of subpart

5.1.1.1 Purpose

The Structured Audio toolset enables the transmission and decoding of synthetic sound effects and music by standardising several different components. Using Structured Audio, high-quality sound can be created at extremely low bandwidth. Typical synthetic music may be coded in this format at bitrates ranging from 0 kbps (no continuous cost) to 2 or 3 kbps for extremely subtle coding of expressive performance using multiple instruments.

MPEG-4 does not standardise a particular set of synthesis methods, but a method for describing synthesis methods. Any current or future sound-synthesis method may be described in the MPEG-4 Structured Audio format.

5.1.1.2 Introduction to major elements

There are five major elements to the Structured Audio toolset:

1. The Structured Audio Orchestra Language, or SAOL. SAOL is a digital-signal processing language which allows for the description of arbitrary synthesis and control algorithms as part of the content bitstream. The syntax and semantics of SAOL are standardised here in a normative fashion.

2. The Structured Audio Score Language, or SASL. SASL is a simple score and control language which is used in certain object types (see subclause 5.6) to describe the manner in which sound-generation algorithms described in SAOL are used to produce sound.

3. The Structured Audio Sample Bank Format, or SASBF. The Sample Bank format allows for the transmission of banks of audio samples to be used in wavetable synthesis and the description of simple processing algorithms to use with them.

4. A normative scheduler description. The scheduler is the supervisory run-time element of the Structured Audio decoding process. It maps structural sound control, specified in SASL or MIDI, to real-time events dispatched using the normative sound-generation algorithms.

5. Normative reference to the MIDI standards, standardised externally by the MIDI Manufacturers Association. MIDI is an alternate means of structural control which can be used in conjunction with or instead of SASL. Although less powerful and flexible than SASL, MIDI support in this standard provides important backward-compatibility with existing content and authoring tools. MIDI support in this standard consists of a list of recognised MIDI messages and normative semantics for each.

5.2 Normative references

[DLS] (c) 1997 MIDI Manufacturers Association, *The MIDI Downloadable Sounds Specification, v. 97.1.*

[DLS2] (c) 1998 MIDI Manufacturers Association, *The MIDI Downloadable Sounds Specification, v. 98.2*.

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

Contents for Subpart 6

6.1 Scope **2**

6.2 Definitions **2**

6.3 Symbols and abbreviations **3**

6.4 MPEG-4 audio text-to-speech bitstream syntax **3**

6.4.1 MPEG-4 audio TTSSpecificConfig **3**
6.4.2 MPEG-4 audio text-to-speech payload **3**

6.5 MPEG-4 audio text-to-speech bitstream semantics **5**

6.5.1 MPEG-4 audio TTSSpecificConfig **5**
6.5.2 MPEG-4 audio text-to-speech payload **6**

6.6 MPEG-4 audio text-to-speech decoding process **7**

6.6.1 Interface between DEMUX and syntactic decoder **8**
6.6.2 Interface between syntactic decoder and speech synthesizer **8**
6.6.3 Interface from speech synthesizer to compositor **8**
6.6.4 Interface from compositor to speech synthesizer **8**
6.6.5 Interface between speech synthesizer and phoneme/bookmark-to-FAP converter **9**

Annex 6.A (informative) **Applications of MPEG-4 audio text-to-speech decoder** **10**

Subpart 6 : TTSI

6.1 Scope

This subpart of ISO/IEC 14496-3 specifies the coded representation of MPEG-4 Audio Text-to-Speech (M-TTS) and its decoder for high quality synthesized speech and for enabling various applications. The exact synthesis method is not a standardization issue partly because there are already various speech synthesis techniques.

This subpart of ISO/IEC 14496-3 is intended for application to M-TTS functionalities such as those for facial animation (FA) and moving picture (MP) interoperability with a coded bitstream. The M-TTS functionalities include a capability of utilizing prosodic information extracted from natural speech. They also include the applications to the speaking device for FA tools and a dubbing device for moving pictures by utilizing lip shape and input text information.

The text-to-speech (TTS) synthesis technology is recently becoming a rather common interface tool and begins to play an important role in various multimedia application areas. For instance, by using TTS synthesis functionality, multimedia contents with narration can be easily composed without recording natural speech sound. Moreover, TTS synthesis with facial animation (FA) / moving picture (MP) functionalities would possibly make the contents much richer. In other words, TTS technology can be used as a speech output device for FA tools and can also be used for MP dubbing with lip shape information. In MPEG-4, common interfaces only for the TTS synthesizer and for FA/MP interoperability are defined. The M-TTS functionalities can be considered as a superset of the conventional TTS framework. This TTS synthesizer can also utilize prosodic information of natural speech in addition to input text and can generate much higher quality synthetic speech. The interface bitstream format is strongly user-friendly: if some parameters of the prosodic information are not available, the missed parameters are generated by utilizing preestablished rules. The functionalities of the M-TTS thus range from conventional TTS synthesis function to natural speech coding and its application areas, i.e., from a simple TTS synthesis function to those for FA and MP.