FINAL DRAFT AMENDMENT© ISO/IEC 2007 — All rights reservedISO/IEC 14496-3:2005/FDAM 7:2007(E) 63Part 3: Audio, AMENDMENT 7: Audio and systems interactionInformation technology — Coding of audio-visual objectsÉlément introductif — Élément central — Partie 3: Élément complémentaireInformation technology — Coding of audio-visual objects — Part 3: Audio, AMENDMENT 7: Audio and systems interactionE2007-01-19(50) ApprovalISO/IECISO/IEC J20057AmendmentInternational Standard 2007ISO/IEC 14496‑ISO/IEC 14496‑3ISO/IEC 14496-3/FDAM 7:2007  Coding of Audio, Picture, Multimedia and Hypermedia InformationInformation Technology11291 2Heading 2;h2;H2;H21;Œ©o‚µ 2;?co??E 2;뙥2;?c1;?co?ƒÊ 2;?2Heading 1;h1;H1;H11;Heading U;Heading;Œ©o‚µ 1;?co??E 1;뙥;?c;?co?ƒÊ 1;?;?co?ƒ  1;Œ;Titre 1       STD Version 2.150   4D:\ct\text\work-marrakesh\wXXXX (14496-3\_FDAM7 TR on Audio Systems Interaction)-hp1.doc            ISO/IEC JTC 1/SC 29

Date:   2024-07-19

ISO/IEC 14496-24

ISO/IEC JTC 1/SC 29/WG 6

Secretariat:  JISC

Information technology — Coding of audio-visual objects — Part 24: Audio and systems interaction

*Technologies de l'information — Codage d'objets audiovisuels — Partie 24: Codage audio et interaction de systèmes*

**Copyright notice**

This ISO document is a Draft International Standard and is copyright-protected by ISO. Except as permitted under the applicable laws of the user's country, neither this ISO draft nor any extract from it may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, photocopying, recording or otherwise, without prior written permission being secured.

Requests for permission to reproduce should be addressed to 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

Reproduction may be subject to royalty payments or a licensing agreement.

Violators may be prosecuted.

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 14496‑24 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information Technology*, Subcommittee SC 29, *Coding of Audio, Picture, Multimedia and Hypermedia Information*.

This second edition cancels and replaces the first edition (ISO/IEC 14496-24:2008), which has been technically revised.

The main changes are as follows:

— addition of details about complex audio and system interaction scenarios and HE-AAC content signalling

— refactored description of timestamp and delay handling

— extension of the HE-AAC example

A list of all parts in the ISO 14496 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Information technology — Coding of audio-visual objects — Part 24: Audio and systems interaction

# Scope

This document describes the desired joint behavior of MPEG-4 Systems (MPEG-4 File Format) and MPEG-4 Audio codecs. It is desired that MPEG-4 Audio encoders and decoders permit finite length signals to be encoded to a file (particularly MPEG-4 files) and decoded again to obtain the identical signal, subject to codec distortions. This will allow the use of audio in systems implementations (particularly MPEG-4 Systems), perhaps with other media such as video, in a deterministic fashion. Most importantly, the decoded signal will have nothing “extra” at the beginning or “missing” at the end.

This permits:

1. an exact ‘round trip’ from raw audio to encoded file back to raw audio (excepting encoding artifacts);
2. predictable synchronization between audio and other media such as video;
3. correct behavior when performing random access as well as when starting at the beginning of a stream;
4. identical behavior when edits are applied in the raw domain and the encoded domain (again, excepting encoding artifacts).

It is also expected that there be predictable interoperability between encoders (as represented by files) and decoders. There are two kinds of audio ‘offsets’ (or ‘delay’ in the context of transmission): those that are result from the encoding process, and those that are result from the decoding process. This document is primarily concerned with the latter.

These issues are resolved by the following:

* The handling of composition time stamps for audio composition units is specified. Special care is taken in the case of compressed data, like HE-AAC coded audio, that can be decoded in a backward compatible fashion as well as in an enhanced fashion.
* Examples are given that show how a finite length signals can be encoded to an MPEG-4 file and decoded again to obtain the identical signal, excepting codec distortions. Most importantly, the decoded signal has nothing “extra” at the beginning or “missing” at the end.

# Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements 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 14496‑3:2019, *Information technology* — *Coding of audio-visual objects* — *Part 3: Audio*

ISO/IEC 14496‑12:2022, *Information technology* — *Coding of audio-visual objects* — *Part 12: ISO base media file format*

# Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 14496‑3 and in ISO/IEC 14496‑12 apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at <https://www.iso.org/obp>

— IEC Electropedia: available at <https://www.electropedia.org/>

# Motivating audio composition time stamp handling

Following ISO/IEC 14496-3 subclause 1.6.5.2, there are 3 ways to signal HE-AAC content:

* Implicit signalling (backward compatible): the audioObjectType in the AudioSpecificConfig structure is set to 2, the AudioSpecificInfo does not contain any extension
* Explicit signalling
  + Backward compatible: the audioObjectType in the AudioSpecificConfig structure is also set to 2, but the AudioSpecificInfo contains SBR extension
  + Hierarchical (non-backward compatible): the audioObjectType in the AudioSpecificConfig structure is set to 5.

For compressed data, like HE-AAC coded audio, which can be decoded by different decoder configurations, decoding can be done in a backward-compatible fashion (AAC only) as well as in an enhanced fashion (AAC+SBR). In order to ensure that timestamps are correct (so that audio remains synchronized with other media), the following is taken into consideration concerning MPEG-4 Systems and Audio:

* If compressed data permits both backward-compatible and enhanced decoding, and if the decoder is operating in a backwards-compatible fashion, then the decoder is not expected to take any action. However, if the decoder is operating in enhanced fashion such that it is using a post-processor that inserts some additional delay (e.g., the SBR post-processor in HE-AAC), then it is expected to notify Systems about the additional time delay incurred relative to the backwards-compatible mode. The exact notification mechanism between the audio decoder (post-processor) and the systems layer is implementation-specific and is not reflected in Systems syntax, i.e. the additional delay is not included in the edit list. With the delay thus notified, Systems can compensate for the additional delay.
* Specifically for HE-AAC (using any of the available signalling mechanisms, i.e., implicit signalling, backward compatible explicit signalling, or hierarchical explicit signalling) the original access unit timestamps (i.e. its decoding time stamp, which in this case is the same as its composition time stamp; but also its presentation time, i.e. after application of any edit list instruction) apply to backward-compatible AAC decoding and playback adjustment for delay compensation is expected in case of AAC+SBR decoding.

Figure 1 shows the composition unit that is generated by an AAC decoder (upper half) and by an HE-AAC decoder operating SBR in dual-rate mode (lower half) when being fed with an access unit of an HE-AAC bitstream. Note that the composition time stamp associated to said access unit applies to the n-th sample of the composition unit. For the AAC decoder case, n has the value 1. For the HE-AAC decoder case, n has the value 962+1 to reflect the additional algorithmic delay of 962 samples of the SBR tool at the HE-AAC output sampling rate (which is twice the sampling rate of the backward compatible AAC output).

Note also that in this document, the term ‘sample’ refers to an audio sample and not an ISOBMFF sample, for which the term ‘access unit’ is preferred in this document.”

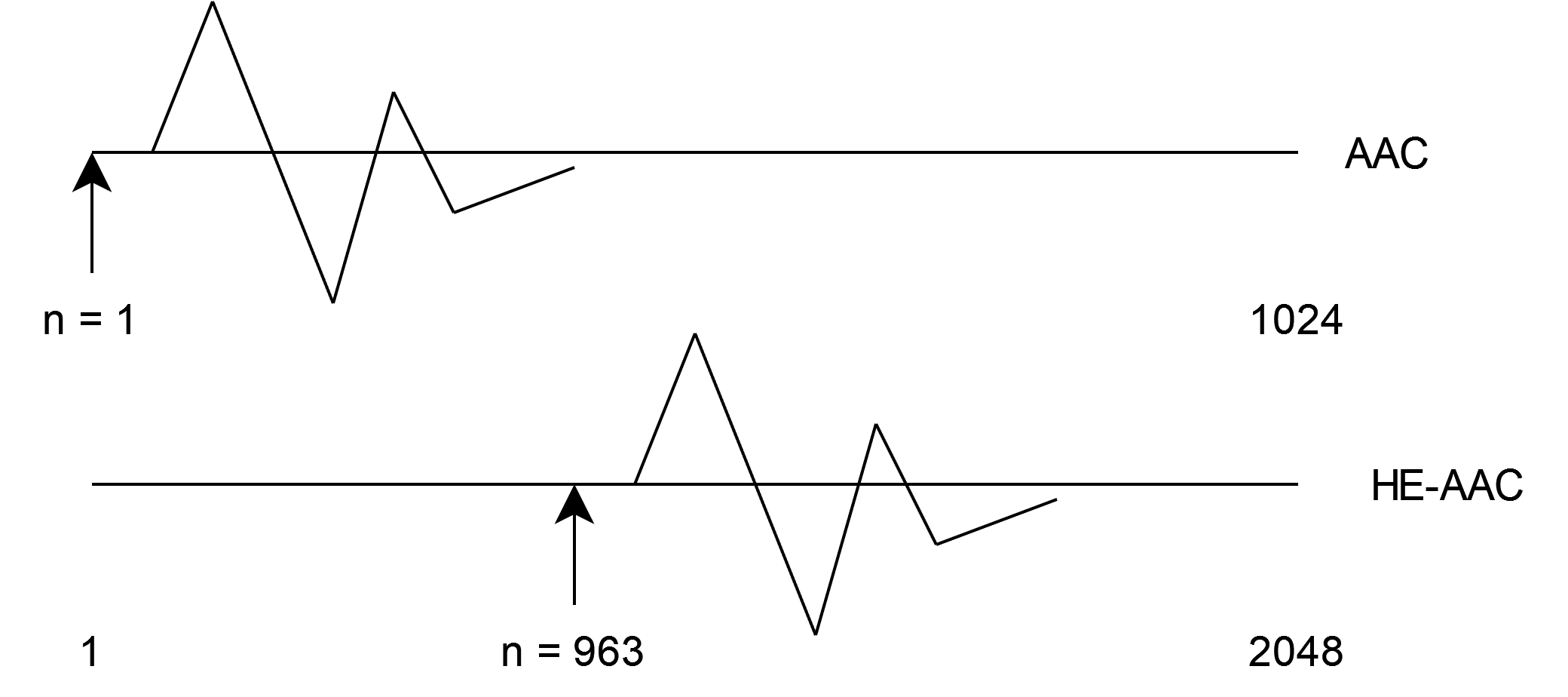


Figure 1 – Composition unit (audio waveform segment) generated by AAC decoder and HE-AAC decoder fed with the same access unit (bitstream frame).

# AAC Encoder/Decoder Behavior

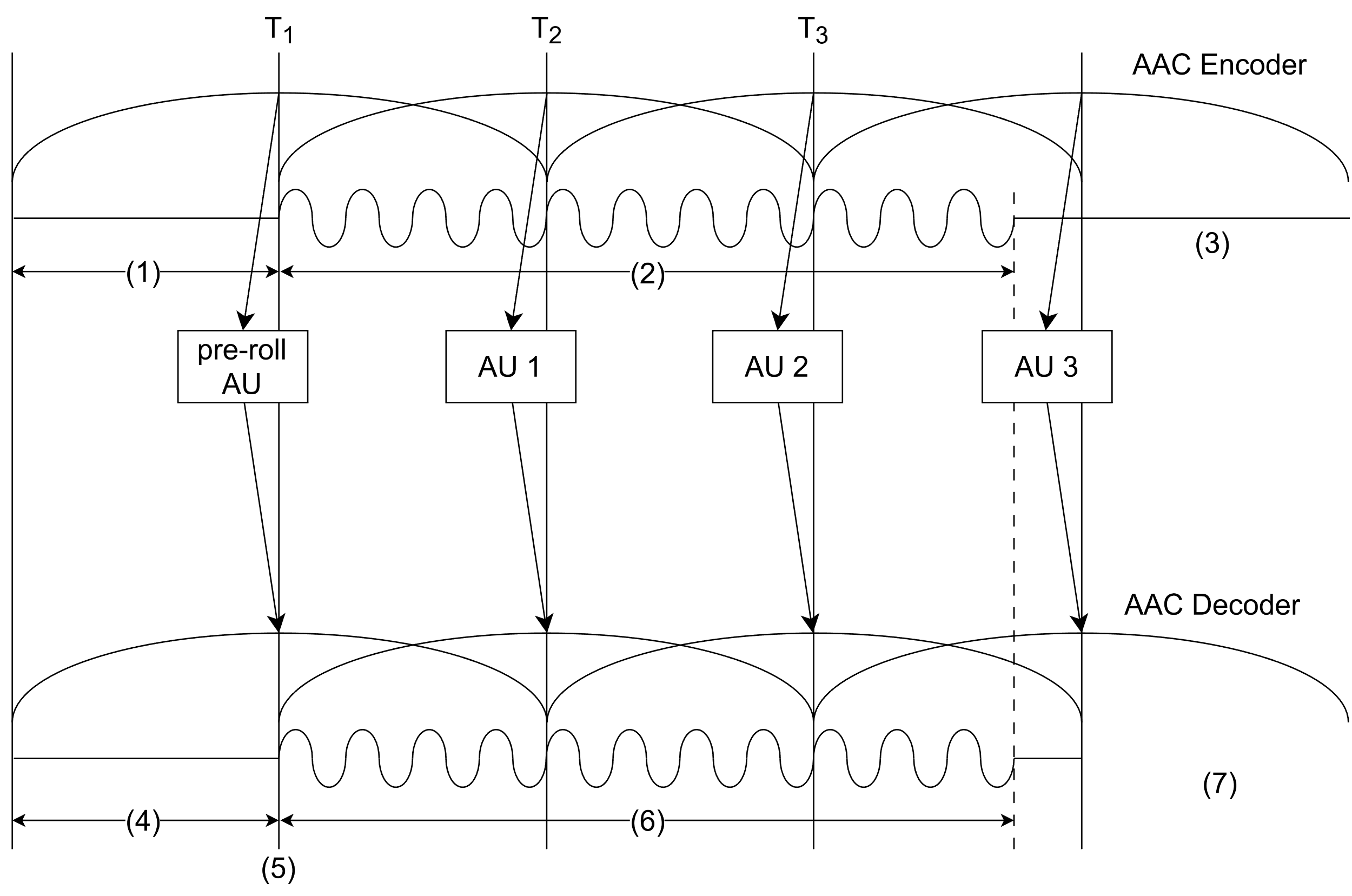
## Example 1: AAC

### Overview

Figure 2 shows the AAC encoder and decoder behavior with respect to the association of encoder input blocks, access units (AU), timestamps and decoder output blocks or composition units (CU). Note that the input signal is only two and a fraction blocks long (as indicated by the oscillating waveform). The encoder essentially extends the waveform at both ends to facilitate encoding of the entire waveform. The ISO Base Media File Format “helper” information “pre-roll” and “edit-list” facilitate exact reconstruction of the encoded waveform segment in the case that the compressed data is stored in an MPEG-4 Format file.

The specifics of encoder behavior are:

* The encoder is expected to produce normative access units
* The timestamp associated with those access units is expected to be the time of the first sample of the waveform in the corresponding composition unit.



Key

(1) encoder state at start-up

(2) input signal

(3) trailing data required to encode last AU

(4) samples discarded by Systems

(5) edit list start

(6) edit list duration

(7) last block remains in decoder as state variable

Figure 2 – AAC encoder/decoder behavior.

In this example the AAC encoder has a start-up state that represents a virtual 1024 samples that precede the first block of 1024 input samples. This virtual 1024 are concatenated with the first 1024 and then are windowed by the length 2048 window and encoded into one access unit. The window shifts over by 1024, such that the next 1024 samples are shifted in and the oldest 1024 samples are shifted out. This defines the 50% overlap processing that is inherent in the MDCT and is the reason that the figure associates an access unit with a window rather than an input block. Note that some AAC encoders potentially have a start-up state (or “look-head”) that is considerably more than 1024 samples. It is the responsibility of the system that uses the encoder, to transfer to the file the correct information (various encoders add 1024 samples, 2048, or even 2048+64 samples).

On shut-down, the encoder in this example is expected to create an additional one and a fraction blocks of samples (typically filling the remainder of the block with zero data) in order to form the last windowed segment of 2048 for the MDCT. Without creating the trailing portion of the last MDCT window, the encoder would not encode the leading portion of the MDCT window, which is valid data.

The decoder produces a composition unit as output for every access unit it receives as input. The edit-list indicates the desired audio output (that is, the valid samples) from amongst the set of samples in the output composition units. In this example, the edit list specifies that Systems discard the first 1024 audio samples (exactly the result from decoding the pre-roll access unit), and also discard the last 256 samples of the decoded waveform (so that the length of the retained audio segment is 2816 samples). In this way four access units are decoded to obtain an exact representation, within the constraints of lossy coding distortion, of the input waveform. Syntax in the ISO base media file format can instruct Systems to perform exactly these operations, such that the desired audio, and only the desired audio is obtained. If there are multiple post-processors that insert additional delay, such as if the Parametric Stereo tool is used in combination with the SBR tool for HE-AACv2, the sum of the additional decoder delay resulting from the use of multiple post-processors is expected to be handled in the same way.

The ISO File Format syntax can specify the need for “pre-roll”, and in this example the roll-distance value of −1 indicates to Systems that it is expected to start the sequence of access units presented to the decoder with the access unit immediately prior to the access unit whose corresponding compositionBuffer contains the start of the desired audio. This includes the cases of starting at the beginning of the audio (the start of the edit list), random access, or where the user has performed further editing in the encoded domain. The pre-roll syntax is shown in the next section.

### Pre-roll

The detailed pre-roll syntax it shown in ANNEX A. For this example, the pre-roll box would contain:

Grouping-type = ‘roll’

Entry-count = 1

Sample-count = <number\_of\_AUs\_in\_track>

Group-description-index = 1

Roll-distance = −1

This indicates that there is one “pre-roll” group, that one ‘extra’ AU is supplied to the decoder, and that this applies no matter where the audio starts playback. Note that Sample-count is equal to the number of samples (or access units) in the track.

### Edit-list

The detailed edit list syntax it shown in ‎Annex A. For this example, the edit list box would contain:

Entry-count = 1

edit-duration = 35 (movie timescale is typically 1/600 seconds)

Media-Time = 1024 (media timescale is 48000 kHz)

Media-Rate = 1

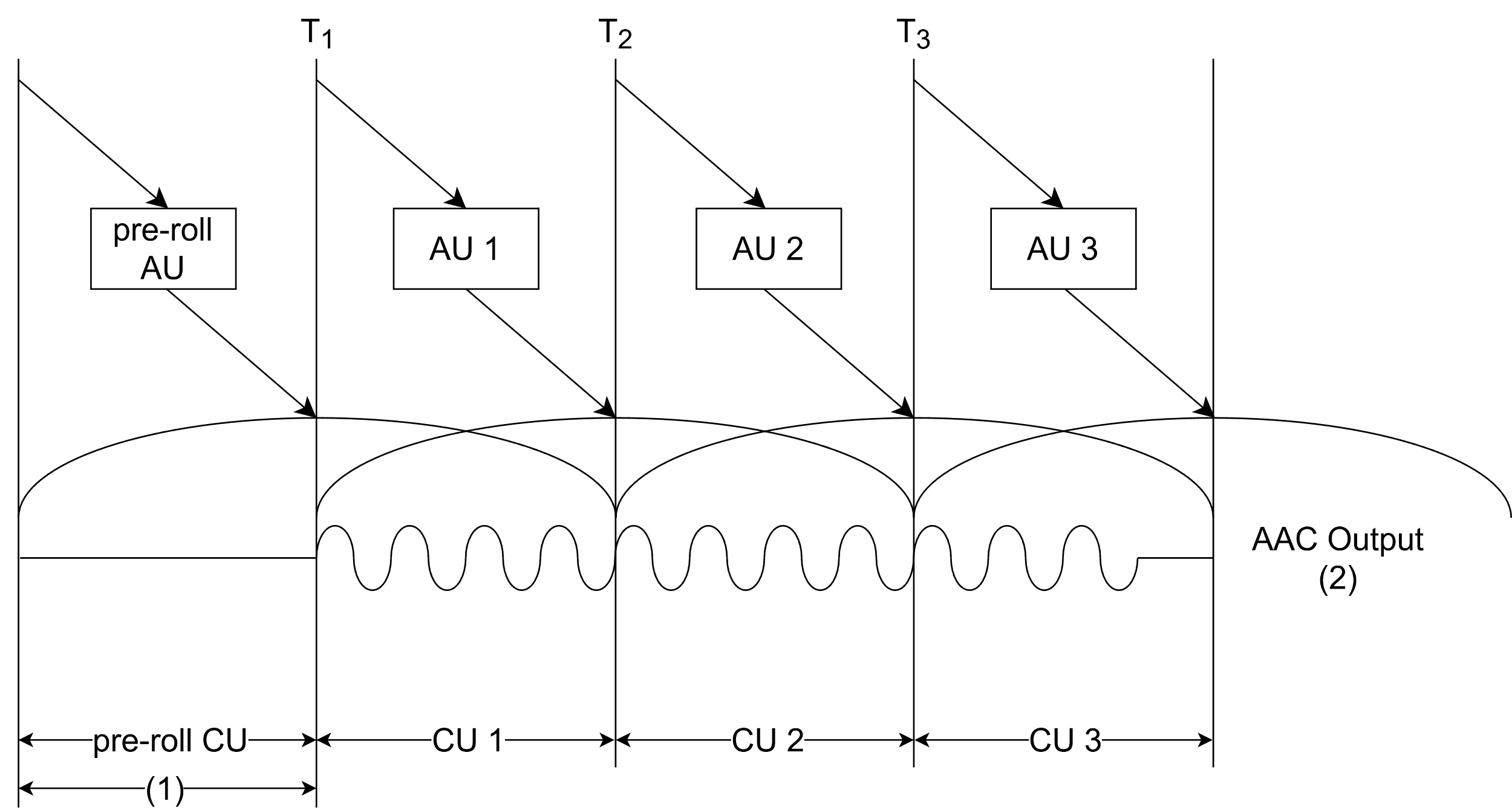
Note that the edit duration is normally expressed in movie timescale units and that the edit start is expressed in media timescale units. The example above indicates that there is one edit, its duration is that of the entire input waveform rounded to the nearest movie timescale value (note that 2816\*48000/600 = 35.2), and that the edit begins after the first 1024 samples, or 1024 in media timescale (indicated as “samples discarded by Systems” in Figure 2). Further note that the movie timescale could be changed to be equal to the media timescale (e.g. for audio-only movies), thereby removing the rounding problem when specifying edit duration.

Track duration is an integer that indicates the duration of this track (in the timescale indicated in the Movie Header Box). The value of this field is equal to the sum of the durations of all of the track's edits. If there is no edit list, then the duration is the sum of the sample durations, converted into the timescale in the Movie Header Box. If the duration of this track cannot be determined then duration is set to all 1s (32-bit maxint).

### Compressed Information and Decoder behavior

In Figure 3, the encoder processing is not indicated, but instead it emphasizes that an access unit has a time stamp, the access unit is decoded into a composition unit, and the timestamp is the time of the first audio sample in that composition unit. Given that process, the encoder behaves such that the timestamps on the access units are correct.

The pre-roll and edit-list information carried in the MPEG-4 File then permit the Systems layer to recover the desired decoded audio segment.



Key

(1) samples discarded by Systems

(2) last block remains in decoder as state variable

Figure 3 – AAC decoder behaviour.

## Example 2: HE-AAC

### Overview

Figure 4 shows the HE-AAC decoder behavior with respect to the access units and associated composition units. An HE-AAC decoder is essentially an AAC decoder followed by an SBR “post-processing” stage. The additional delay imposed by the SBR tool is due to the QMF bank and the data buffers within the SBR tool. It can be derived by the following.



where

,  and .

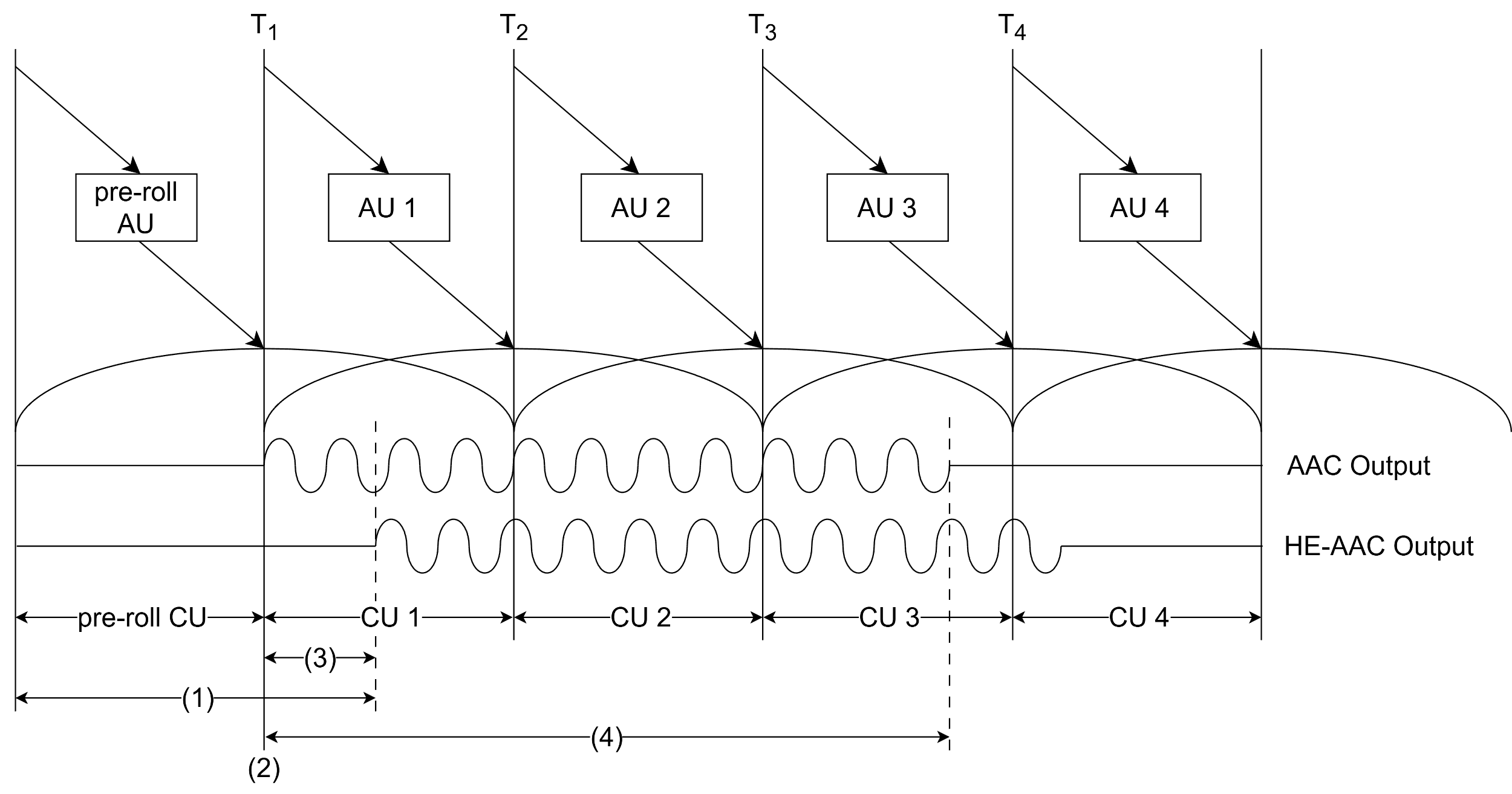
This means that the delay imposed by the SBR tool (at the input sampling rate, i.e., the output sampling rate of the AAC) is



samples. Typically, the SBR tool runs in the “upsampling” (or “dual rate”) mode, in which case the 481 sample delay at the AAC sampling rate translates to a 962 sample delay at the SBR output rate. It could also operate at the same sampling rate as the AAC output (denoted as “downsampled SBR mode”), in which case the additional delay is only 481 samples at the SBR output rate. If the bitstream contains HE-AAC content but is signaled in a “backwards compatible” mode (either explicit or implicit) and if the decoder does not have SBR support, the SBR tool is not applied and the AAC output is the decoder output. In this case there is no additional delay.

Figure 4 shows the decoder behavior for the most common case in which the SBR tool runs in upsampling mode and the additional delay is 962 output samples. This delay corresponds to approx. 47% of the length of the upsampled AAC frame (after SBR processing). T1 is the timestamp associated with CU 1, that is, the timestamp for the first valid sample of HE-AAC output. Decoders are expected to either remove the additional 962 samples introduced by the SBR tool or signal them to the Systems layer in an implementation-specific way, so that the Systems layer can align the first valid sample of HE-AAC output with T1. Further note that if HE-AAC is running in “downsampled SBR mode” or “single-rate” mode, the delay would be 481 samples but the timestamp would be identical since in single-rate mode the CU’s are half the number of samples so that the delay is still 47% of the CU duration.

For all of the available signaling mechanisms (i.e., implicit signaling, backward compatible explicit signaling, or hierarchical explicit signaling) if the decoder is HE-AAC then it is expected to convey to Systems any additional delay incurred by SBR processing, otherwise the lack of an indication from the decoder indicates that the decoder is AAC. Hence, Systems can compensate for the additional SBR delay.



Key

(1) samples discarded by Systems

(2) edit list start

(3) SBR “dual-rate” processing delay of 962 samples

(4) edit list duration

Figure 4 – HE-AAC decoder behavior: dual-rate mode.

### Pre-roll

For the above example, the pre-roll information would be the same as for the AAC example given in subclause 5.1.2.

In general, the pre-roll information is not modified if a bitstream is re-encoded using the SBR tool, independent of the signalling mode (backwards-compatible implicit, backwards-compatible explicit, or non-backwards-compatible hierarchical).

### Edit-list

For the above example, the edit-list information would be the same as for the AAC example given in subclause 5.1.3.

In general, the edit-list information is not modified if a bitstream is re-encoded using the SBR tool, independent of the signalling mode (backwards-compatible implicit, backwards-compatible explicit, or non-backwards-compatible hierarchical).

# Streaming Considerations

Not strictly within the scope of MPEG, it is nonetheless worth discussing how these issues can be handled in RTP streaming. In that context, there is no ‘edit list’ and there is also no ability to warn of the need for pre-roll. The best that can be done is to send all the access units, but with timestamps that respect the edit list.

Consider the following example:

* The time-scale of the audio stream, and the sampling rate, are 48000 Hz.
* The file indicates that the ‘desired’ audio starts at sample 65 in the second access unit.
* The desired time of the start of the audio is R on the RTP time-line.

The system sends the first access unit with a timestamp R-64-1024, and the second with a timestamp of R-64. The client will play out the (undesired) pre-roll data, but the subsequent audio will be in sync with other media, such as video, as is desired.

1. - Relevant ISO Base Media File Format Syntax
   1. Pre-roll syntax

ISO/IEC 14496-12:2005, ISO Base Media File Format, specifies the syntax of the “pre-roll” (“AudioRollRecoveryEntry”) information in Section 8.40.4.

aligned(8) class SampleToGroupBox   
 extends FullBox(‘sbgp’, version = 0, 0)  
{  
 unsigned int(32) grouping\_type;  
 unsigned int(32) entry\_count;  
 for (i=1; i <= entry\_count; i++)  
 {  
 unsigned int(32) sample\_count;  
 unsigned int(32) group\_description\_index;  
 }  
}

// Sequence Entry   
abstract class SampleGroupDescriptionEntry (unsigned int(32) handler\_type)   
{  
}

// Visual Sequence   
abstract class VisualSampleGroupEntry (type) extends  
 SampleGroupDescriptionEntry (type)  
{  
}

// Audio Sequences  
abstract class AudioSampleGroupEntry (type) extends SampleGroupDescriptionEntry (type)  
{  
}

aligned(8) class SampleGroupDescriptionBox (unsigned int(32) handler\_type)  
 extends FullBox('sgpd', version, 0){  
 unsigned int(32) grouping\_type;  
if (version==1) { unsigned int(32) default\_length; }  
 unsigned int(32) entry\_count;  
 int i;  
 for (i = 1 ; i <= entry\_count ; i++){  
 if (version==1) {  
 if (default\_length==0) {  
 unsigned int(32) description\_length;  
 }  
 }  
 switch (handler\_type){  
 case 'vide': // for video tracks  
 VisualSampleGroupEntry (grouping\_type);  
 break;  
 case 'soun': // for audio tracks  
 AudioSampleGroupEntry(grouping\_type);  
 break;  
 case 'hint': // for hint tracks  
 HintSampleGroupEntry(grouping\_type);  
 break;  
 }  
 }  
}

class VisualRollRecoveryEntry() extends VisualSampleGroupEntry (’roll’)  
{  
 signed int(16) roll-distance;  
}

roll\_distance is a signed integer that gives the number of samples to decode in order for a sample to be decoded correctly. A positive value indicates the number of samples after the group member sample to decode such that at the last of these recovery is complete, i.e. the last sample is correct. A negative value indicates the number of samples before the group member sample to decode in order for recovery to be complete at the marked sample. The value zero is not used; the sync sample table documents random access points for which no recovery roll is needed.

* 1. Edit-list syntax

Section 8.26 of ISO/IEC 14496-12:2003, ISO Base Media File Format, specifies the edit list box, and Section 8.26.2 specifies its syntax, which is repeated here:

aligned(8) class EditListBox extends FullBox(‘elst’, version, 0) {  
 unsigned int(32) entry\_count;  
 for (i=1; i ≤ entry\_count; i++) {  
 if (version==1) {  
 unsigned int(64) segment\_duration;  
 int(64) media\_time;  
 } else { // version==0  
 unsigned int(32) segment\_duration;  
 int(32) media\_time;  
 }  
 int(16) media\_rate\_integer;  
 int(16) media\_rate\_fraction;  
 }  
}