Universal Serial Bus
Device Class Definition
for
Audio Data Formats

Release 1.0

March 18, 1998

Scope of This Release

This document is the 1.0 release of this device class definition.

Contributors

Gal Ashour	IBM Corporation
Billy Brackenridge	Microsoft Corporation
Oren Tirosh	Altec Lansing
Craig Todd	Dolby Laboratories
Remy Zimmermann	Logitech
Geert Knapen	Philips ITCL
	Interleuvenlaan 74-76
	B-3001 Leuven-Heverlee BELGIUM
	Phone: +32 16 390 734
	Fax: +32 16 390 600
	E-mail: Geert.Knapen@innet.be

Revision History

Revision	Date	Filename	Author	Description
0.1	Dec. 1, 96	Frmts01.doc	Geert Knapen	Initial version
0.2	Jan. 1, 97	Frmts02.doc	Geert Knapen	Corrected typos.
0.3	Mar. 1, 97	Frmts03.doc	Geert Knapen	Adapted template and contents to correspond with core document.
0.9rc	Apr. 1, 97	Frmts09rc.doc	Geert Knapen	Brought in line with core document. Added Type II descriptors and requests.
0.9	May 1, 97	Frmts09.doc	Geert Knapen	Added details for MPEG and AC-3. Added format-specific requests.
0.9CE	Sep 1, 97	Frmts09CE.doc	Geert Knapen	Copy-edited for publication on the web.
0.9a	Oct 1, 97	Frmts09a.doc	Geert Knapen	Incorporated RRs
1.0RC	Mar 1, 98	Frmts10RC.doc	Geert Knapen	Added the Transfer Delimiter concept. Cleaned up the formatting.
1.0	Mar 18, 98	Frmts10.doc	Geert Knapen	Changed all references to 1.0

INTELLECTUAL PROPERTY DISCLAIMER

THIS SPECIFICATION IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER INCLUDING ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION, OR SAMPLE.

A LICENSE IS HEREBY GRANTED TO REPRODUCE AND DISTRIBUTE THIS SPECIFICATION FOR INTERNAL USE ONLY. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY OTHER INTELLECTUAL PROPERTY RIGHTS IS GRANTED OR INTENDED HEREBY.

AUTHORS OF THIS SPECIFICATION DISCLAIM ALL LIABILITY, INCLUDING LIABILITY FOR INFRINGEMENT OF PROPRIETARY RIGHTS, RELATING TO IMPLEMENTATION OF INFORMATION IN THIS SPECIFICATION. AUTHORS OF THIS SPECIFICATION ALSO DO NOT WARRANT OR REPRESENT THAT SUCH IMPLEMENTATION(S) WILL NOT INFRINGE SUCH RIGHTS.

Please send comments via electronic mail to techsup@usb.org

Table of Contents

List of Tables

1 Introduction

The intention of this document is to describe in detail all the Audio Data Formats that are supported by the Audio Device Class. This document is considered an integral part of *the Audio Device Class Specification*, although subsequent revisions of this document are independent of the revision evolution of the main *USB Audio Specification*. This is to easily accommodate the addition of new Audio Data Formats without impeding the core *USB Audio Specification*.

1.1 Related Documents

- *Universal Serial Bus Specification*, 1.0 final draft revision (also referred to as the *USB Specification*). In particular, see Chapter 9, “USB Device Framework.”
- *Universal Serial Bus Device Class Definition for Audio Data Formats* (referred to in this document as *USB Audio Data Formats*).
- *Universal Serial Bus Device Class Definition for Terminal Types* (referred to in this document as *USB Audio Terminal Types*).
- ANSI S1.11-1986 standard.
- MPEG-1 standard ISO/IEC 111172-3 1993.
- MPEG-2 standard ISO/IEC 13818-3 Feb. 20, 1997.
- Digital Audio Compression Standard (AC-3), ATSC A/52 Dec. 20, 1995. (available from http://www.atsc.org)
- ANSI/IEEE-754 floating-point standard.
- ISO/IEC 958 International Standard: *Digital Audio Interface and Annexes.*
- ISO/IEC 1937 standard.
- ITU G.711 standard.

1.2 Terms and Abbreviations

This section defines terms used throughout this document. For additional terms that pertain to the Universal Serial Bus, see Chapter 2, “Terms and Abbreviations,” in the *USB Specification*.

Audio Frame	A collection of audio subframes, each containing a PCM audio sample of a different physical audio channel, taken at the same moment in time.
Audio Stream	A concatenation of a potentially very large number of audio frames ordered according to ascending time.
Audio Subframe	Holds a single PCM audio sample.
DVD	Acronym for Digital Versatile Disc.
Encoded Audio Bitstream	A concatenation of a potentially very large number of encoded audio frames, ordered according to ascending time.
Encoded Audio Frame	A sequence of bits that contains an encoded representation of one or more physical audio channels.
MPEG	Acronym for Moving Pictures Expert Group.
PCM	Acronym for Pulse Coded Modulation.
Transfer Delimiter	A unique token that indicates an interruption in an isochronous data packet stream. Can be either a zero-length data packet or the absence of an isochronous transfer in a certain USB frame.

2 Audio Data Formats

Audio Data Formats can be divided in three main groups according to type.

The first group, Type I, deals with audio data streams that are constructed on a sample-by-sample basis. Each audio sample is represented by a single independent symbol and the data stream is built up by concatenating those symbols. Different compression schemes may be used to transform the audio samples into symbols. If multiple physical audio channels are formatted into a single audio channel cluster, then samples at time *x* of subsequent channels are transmitted interleaved, according to the cluster channel ordering as described in the main *USB Audio Specification*, followed by samples at time *x*+1, interleaved in the same fashion and so on. The notion of physical channels is explicitly preserved during transmission. A typical example of Type I formats is the standard PCM audio data.

The second group, Type II, deals with those formats that do not preserve the notion of physical channels during the transmission. Typically, all non-PCM encoded audio data streams belong to this group. A number of audio samples, often originating from multiple physical channels, are encoded into a number of bits in such a way that, after transmission, the original audio samples can be reconstructed to a certain degree of accuracy. The number of bits used for transmission is typically one or more orders of magnitude smaller than the number of bits needed to represent the original PCM audio samples, effectively realizing a considerable bandwidth reduction during transmission.

The third group, Type III, contains special formats that do not fit in both previous groups. In fact, they mix characteristics of Type I and Type II groups to transmit audio data streams. One or more non-PCM encoded audio data streams are packed into “pseudo-stereo samples” and transmitted as if they were real stereo PCM audio samples. The sampling frequency of these pseudo samples matches the sampling frequency of the original PCM audio data streams. Therefore, clock recovery at the receiving end is easier than it is in the case of Type II formats. The drawback is that unless multiple non-PCM encoded streams are packed into one pseudo stereo stream, more bandwidth than necessary is consumed.

Section A.1, “Audio Data Format Codes” summarizes the Audio Data Formats that are currently supported in the Audio Device Class. The following sections explain those formats in more detail.

2.1 Transfer Delimiter

Isochronous data streams are continuous in nature, although the actual number of bytes sent per packet may vary throughout the lifetime of the stream (for rate adaptation purposes for instance). To indicate a temporary stop in the isochronous data stream without closing the pipe (and thus relinquishing the USB bandwidth), an in-band Transfer Delimiter needs to be defined. This specification considers two situations to be a Transfer Delimiter. The first is a zero-length data packet and the second is the absence of an isochronous transfer in a particular USB frame. Both situations are considered equivalent and the audio function is expected to behave the same. However, the second type consumes less isochronous USB bandwidth (i.e. zero bandwidth). In both cases, this specification considers a Transfer Delimiter to be an entity that can be sent over the USB.

2.2 Type I Formats

The following sections describe the Audio Data Formats that belong to Type I. A number of terms and their definition are presented.

2.2.1 USB Packets

Audio data streams that are inherently continuous must be packetized when sent over the USB. The quality of the packetizing algorithm directly influences the amount of effort needed to reconstruct a reliable sample clock at the receiving side. The goal must be to keep the instantaneous number of samples

per frame (n_i) as close as possible to the average number of samples per frame, (n_{av}). The average n_{av} should be calculated as a sliding average over a period of 256 frames.

If the sampling rate is a constant, the allowable variation on n_i is limited to one sample, that is, $\Delta n_i = 1$. This implies that all packets must either contain INT (n_{av}) (small packet) or INT (n_{av}) + 1 (large packet) samples. For all *i*:

$$n_i = \text{INT}(n_{av}) \mid \text{INT}(n_{av}) + 1$$

Note: In the case where n_{av} = INT (n_{av}), n_i may vary between INT (n_{av}) - 1 (small packet), INT (n_{av}) (medium packet) and INT (n_{av}) + 1 (large packet).

To limit the needed buffer depths to acceptable limits, this specification limits the cumulative difference between n_{av} and n_i to ±1.5 samples.

If the sampling rate can be varied (to implement pitch control), the allowable pitch shift is 1kHz/ms. That is, the allowable variation on n_i is limited to one sample per frame. For all *i*:

$$n_{i+1} = n_i \pm 1$$

Pitch control is restricted to adaptive endpoints only. AudioStreaming interfaces that support pitch control on their isochronous endpoint are required to report this in the class-specific endpoint descriptor. In addition, a Set/Get Pitch Control request is required to enable or disable the pitch control functionality.

2.2.2 Audio Subframe

The basic structure used to represent audio data is the audio subframe. An audio subframe holds a single audio sample. An audio subframe always contains an integer number of bytes.

This specification limits the possible audio subframe sizes (**bSubframeSize**) to 1, 2, 3 or 4 bytes per audio subframe. An audio sample is represented using a number of bits (**bBitResolution**) less than or equal to the total number of bits available in the audio subframe, i.e. **bBitResolution** ≤ **bSubframeSize***8.

AudioStreaming endpoints must be constructed in such a way that a valid transfer can take place as long as the reported audio subframe size (**bSubframeSize**) is respected during transmission. If the reported bits per sample (**bBitResolution**) do not correspond with the number of significant bits actually used during transfer, the device will either discard trailing significant bits ([actual_bits_per_sample] > **bBitResolution**) or interpret trailing zeros as significant bits ([actual_bits_per_sample] < **bBitResolution**).

2.2.3 Audio Frame

An audio frame consists of a collection of audio subframes, each containing an audio sample of a different physical audio channel, taken at the same moment in time. The number of audio subframes in an audio frame equals the number of logical audio channels in the audio channel cluster. The ordering of the audio subframes in the audio frame obeys the rules set forth in the *USB Audio Specification*. All audio subframes must have the same audio subframe size.

2.2.4 Audio Streams

An audio stream is a concatenation of a potentially very large number of audio frames, ordered according to ascending time. Streams are packetized when transported over USB whereby USB packets can only contain an integer number of audio frames. Each packet always starts with the same channel, and the channel order is respected throughout the entire transmission. If, for any reason, there are no audio frames available to construct a USB packet, a Transfer Delimiter must be sent instead.

2.2.5 Type I Format Type Descriptor

The Type I format type descriptor starts with the usual three fields: **bLength**, **bDescriptorType**, and **bDescriptorSubtype**.

The **bFormatType** field indicates this is a Type I descriptor. The **bNrChannels** field contains the number of physical channels in the audio data stream. The **bSubframeSize** field indicates how many bytes are used to transport an audio subframe. The **bBitResolution** field indicates how many bits of the total number of available bits in the audio subframe are truly used by the audio function to convey audio information.

The sampling frequency capabilities of the isochronous data endpoint of the AudioStreaming Interface are reported as well. Depending on the **bSamFreqType** field, the length of the descriptor varies and the interpretation of the trailing fields differs. Sampling frequencies occupy three bytes and are expressed in Hz to support over-sampled, reduced bit-resolution systems (the range is from 0 to 16,777,215 Hz).

Table 2-1: Type I Format Type Descriptor

Offset	Field	Size	Value	Description
0	bLength	1	Number	Size of this descriptor, in bytes: 8+(ns*3)
1	bDescriptorType	1	Constant	CS_INTERFACE descriptor type.
2	bDescriptorSubtype	1	Constant	FORMAT_TYPE descriptor subtype.
3	bFormatType	1	Constant	FORMAT_TYPE_I. Constant identifying the Format Type the AudioStreaming interface is using.
4	bNrChannels	1	Number	Indicates the number of physical channels in the audio data stream.
5	bSubframeSize	1	Number	The number of bytes occupied by one audio subframe. Can be 1, 2, 3 or 4.
6	bBitResolution	1	Number	The number of effectively used bits from the available bits in an audio subframe.
7	bSamFreqType	1	Number	Indicates how the sampling frequency can be programmed: 0: Continuous sampling frequency 1..255: The number of discrete sampling frequencies supported by the isochronous data endpoint of the AudioStreaming interface (ns)
8...				See sampling frequency tables, below.

Depending on the value in the **bSamFreqType** field, the layout of the next part of the descriptor is as shown in the following tables.

Table 2-2: Continuous Sampling Frequency

Offset	Field	Size	Value	Description

Offset	Field	Size	Value	Description
8	tLowerSamFreq	3	Number	Lower bound in Hz of the sampling frequency range for this isochronous data endpoint.
11	tUpperSamFreq	3	Number	Upper bound in Hz of the sampling frequency range for this isochronous data endpoint.

Table 2-3: Discrete Number of Sampling Frequencies

Offset	Field	Size	Value	Description
8	tSamFreq [1]	3	Number	Sampling frequency 1 in Hz for this isochronous data endpoint.
…	…	…	…	…
8+(ns-1)*3	tSamFreq [ns]	3	Number	Sampling frequency ns in Hz for this isochronous data endpoint.

Note: In the case of adaptive isochronous data endpoints that support only a discrete number of sampling frequencies, the endpoint must at least tolerate ±1000 PPM inaccuracy on the reported sampling frequencies.

2.2.6 Supported Formats

The following paragraphs list all currently supported Type I Audio Data Formats.

2.2.6.1 PCM Format

The PCM (Pulse Coded Modulation) format is the most commonly used audio format to represent audio data streams. The audio data is not compressed and uses a signed two's-complement fixed point format. It is left-justified (the sign bit is the Msb) and data is padded with trailing zeros to fill the remaining unused bits of the subframe. The binary point is located to the right of the sign bit so that all values lie within the range [-1,+1).

2.2.6.2 PCM8 Format

The PCM8 format is introduced to be compatible with the legacy 8-bit wave format. Audio data is uncompressed and uses 8 bits per sample (**bBitResolution** = 8). In this case, data is unsigned fixed-point, left-justified in the audio subframe, Msb first. The range is [0,255].

2.2.6.3 IEEE_FLOAT Format

The IEEE_FLOAT format is based on the ANSI/IEEE-754 floating-point standard. Audio data is represented using the basic single-precision format. The basic single-precision number is 32 bits wide and has an 8-bit exponent and a 24-bit mantissa. Both mantissa and exponent are signed numbers, but neither is represented in two's-complement format. The mantissa is stored in sign magnitude format and the exponent in biased form (also called excess-n form). In biased form, there is a positive integer (called the bias) which is subtracted from the stored number to get the actual number. For example, in an eight-bit exponent, the bias is 127. To represent 0, the number 127 is stored. To represent -100, 27 is stored. An

exponent of all zeroes and an exponent of all ones are both reserved for special cases, so in an eight-bit field, exponents of -126 to +127 are possible. In the basic floating-point format, the mantissa is assumed to be normalized so that the most significant bit is always one, and therefore is not stored. Only the fractional part is stored.

The 32-bit IEEE-754 floating-point word is broken into three fields. The most significant bit stores the sign of the mantissa, the next group of 8 bits stores the exponent in biased form, and the remaining 23 bits store the magnitude of the fractional portion of the mantissa. For further information, refer to the ANSI/IEEE-754 standard.

The data is conveyed over USB using 32 bits per sample (**bBitResolution** = 32; **bSubframeSize** = 4).

2.2.6.4 ALaw Format and μLaw Format

Starting from 12- or 16-bits linear PCM samples, simple compression down to 8-bits per sample (one byte per sample) can be achieved by using logarithmic companding. The compressed audio data uses 8 bits per sample (**bBitsPerSample** = 8). Data is signed fixed point, left-justified in the subframe, Msb first. The compressed range is [-128,128]. The difference between Alaw and μLaw compression lies in the formulae used to achieve the compression. Refer to the ITU G.711 standard for further details.

2.3 Type II Formats

Type II formats are used to transmit non-PCM encoded audio data into bitstreams that consist of a sequence of encoded audio frames.

2.3.1 Encoded Audio Frames

An encoded audio frame is a sequence of bits that contains an encoded representation of one or more physical audio channels. The encoding takes place over a fixed number of audio samples. Each encoded audio frame contains enough information to entirely reconstruct the audio samples (albeit not lossless), encoded in the encoded audio frame. No information from adjacent encoded audio frames is needed during decoding. The number of samples used to construct one encoded audio frame depends on the encoding scheme. (For MPEG, the number of samples per encoded audio frame (n_f) is 384 for Layer I or 1152 for Layer II. For AC-3, the number of samples is 1536.)

In most cases, the encoded audio frame represents multiple physical audio channels. The number of bits per encoded audio frame may be variable. The content of the encoded audio frame is defined according to the implemented encoding scheme. Where applicable, the bit ordering shall be MSB first, relative to existing standards of serial transmission or storage of that encoding scheme. An encoded audio frame represents an interval longer than the USB frame time of 1 ms. This is typical of audio compression algorithms that use psycho-acoustic or vocal tract parametric models.

Note: It is important to make a clear distinction between an audio frame (see Section 2.2.3, “Audio Frame”) and an encoded audio frame. The overloaded use of the term audio frame could cause confusion. Therefore, this specification will always use the qualifier ‘encoded’ to refer to MPEG or AC-3 encoded audio frames.

2.3.2 Audio Bitstreams

An encoded audio bitstream is a concatenation of a potentially very large number of encoded audio frames, ordered according to ascending time. Subsequent encoded audio frames are independent and can be decoded separately.

2.3.3 USB Packets

Encoded audio bitstreams are packetized when transported over an isochronous pipe. Each USB packet contains only part of a single encoded audio frame. Packet sizes are determined according to the short-packet protocol. The encoded audio frame is broken down into a number of packets, each containing **wMaxPacketSize** bytes except for the last packet, which may be smaller and contains the remainder of the encoded audio frame. If the **MaxPacketsOnly** bit D7 in the **bmAttributes** field of the class-specific endpoint descriptor is set, the last (short) packet must be padded with zero bytes to **wMaxPacketSize** length. No USB packet may contain bits belonging to different encoded audio frames. If the encoded audio frame length is not a multiple of 8 bits, the last byte in the last packet is padded with zero bits. The decoder must ignore all padded extra bits and bytes. Consecutive encoded audio frames are separated by at least one Transfer Delimiter. A Transfer Delimiter must be sent in all consecutive USB frames until the next encoded audio frame is due. The above rules guarantee that a new encoded audio frame always starts on a USB packet boundary.

2.3.4 Bandwidth Allocation

The encoded audio frame time t_f equals the number of audio samples per encoded audio frame n_f divided by the sampling rate f_s of the original audio samples.

$$t_f = \frac{n_f}{f_s}$$

The allocated bandwidth for the pipe must accommodate for the largest possible encoded audio frame to be transmitted within an encoded audio frame time. This should take into account the Transfer Delimiter requirement and any differences between the time base of the stream and the USB frame timer. The device may choose to consume more bandwidth than necessary (by increasing the reported **wMaxPacketSize**) to minimize the time needed to transmit an entire encoded audio frame. This can be used to enable early decoding and therefore minimize system latency.

2.3.5 Timing

The timing reference point is the beginning of an encoded audio frame. Therefore, the USB packet that contains the first bits (usually the encoded audio frame sync word) of the encoded audio frame is used as a timing reference in USB space. This USB packet is called the reference packet. The transmission of the reference packet of an encoded audio frame should begin at the target playback time of that frame (minus the endpoint's reported delay) rounded to the nearest USB frame time. This guarantees that, at the receiving end, the arrival of subsequent reference packets matches the encoded audio frame time t_f as closely as possible.

2.3.6 Type II Format Type Descriptor

The Type II Format Type descriptor starts with the usual three fields **bLength**, **bDescriptorType** and **bDescriptorSubtype**.

The **bFormatType** field indicates this is a Type II descriptor. The **wMaxBitRate** field contains the maximum number of bits per second this interface can handle. It is a measure for the buffer size available in the interface. The **wSamplesPerFrame** field contains the number of non-PCM encoded audio samples contained within a single encoded audio frame

The sampling frequency capabilities of the endpoint are reported using the **bSamFreqType** field and following fields.

Table 2-4: Type II Format Type Descriptor

Offset	Field	Size	Value	Description

Offset	Field	Size	Value	Description
0	bLength	1	Number	Size of this descriptor, in bytes: 9+(ns*3)
1	bDescriptorType	1	Constant	CS_INTERFACE descriptor type.
2	bDescriptorSubtype	1	Constant	FORMAT_TYPE descriptor subtype.
3	bFormatType	1	Constant	FORMAT_TYPE_II. Constant identifying the Format Type the AudioStreaming interface is using.
4	wMaxBitRate	2	Number	Indicates the maximum number of bits per second this interface can handle. Expressed in kbits/s.
6	wSamplesPerFrame	2	Number	Indicates the number of PCM audio samples contained in one encoded audio frame.
8	bSamFreqType	1	Number	Indicates how the sampling frequency can be programmed: 0: Continuous sampling frequency 1..255: The number of discrete sampling frequencies supported by the isochronous data endpoint of the AudioStreaming interface (ns)
9...				See sampling frequency tables, below.

Depending on the value in the **bSamFreqType** field, the layout of the next part of the descriptor is as shown in the following tables.

Table 2-5: Continuous Sampling Frequency

Offset	Field	Size	Value	Description
9	tLowerSamFreq	3	Number	Lower bound in Hz of the sampling frequency range for this isochronous data endpoint.
12	tUpperSamFreq	3	Number	Upper bound in Hz of the sampling frequency range for this isochronous data endpoint.

Table 2-6: Discrete Number of Sampling Frequencies

Offset	Field	Size	Value	Description
9	tSamFreq [1]	3	Number	Sampling frequency 1 in Hz for this isochronous data endpoint.
…	…	…	…	…

Offset	Field	Size	Value	Description
9+(ns-1)*3	tSamFreq [ns]	3	Number	Sampling frequency ns in Hz for this isochronous data endpoint.

Note: In the case of adaptive isochronous data endpoints that support only a discrete number of sampling frequencies, the endpoint must at least tolerate ±1000 PPM inaccuracy on the reported sampling frequencies.

2.3.7 Rate feedback

If the isochronous data endpoint needs explicit rate feedback (adaptive source, asynchronous sink), the feedback pipe shall report the number of equivalent PCM audio samples. The host will accumulate this data and start transmission of an encoded audio frame whenever the current number of samples exceeds the number of samples per encoded audio frame. The remainder is kept in the accumulator.

2.3.8 Supported Formats

The following sections list all currently supported Type II Audio Data Formats. Format-specific descriptors and format-specific requests are explained in more detail.

2.3.8.1 MPEG Format

In the current specification, only MPEG decoding aspects are considered. Real-time MPEG encoding peripherals are not (yet) available and consequently are not covered by this specification.

2.3.8.1.1 MPEG Format-Specific Descriptor

The **wFormatTag** field is a duplicate of the **wFormatTag** field in the class-specific AudioStreaming interface descriptor. The same field is used here to identify the format-specific descriptor.

The **bmMPEGCapabilities** bitmap field describes the capabilities of the MPEG decoder built into the AudioStreaming interface.

Some general information must be retrieved from the Format Type-specific descriptor. For instance, the sampling frequencies supported by the decoder are reported through the Format Type-specific descriptor. This includes the ability of the decoder to handle low sampling frequencies (16 kHz, 22.05 kHz and 24 kHz) besides the standard 32 kHz, 44.1 kHz and 48 kHz sampling frequencies.

Bits D2..0 of the **bmMPEGCapabilities** field are used to indicate which layers this decoder is capable of processing. The different layers relate to the different algorithms that are used during encoding and decoding.

Bit D3 indicates that the decoder can only process the MPEG-1 base stream. Therefore, only Left and Right channels will be output.

Bit D4 indicates that the decoder can handle MPEG-2 streams that contain two independent stereo pairs instead of the normal 3/2 encoding scheme. This bit is only applicable for MPEG-2 decoders.

Bit D5 indicates that the decoder supports the MPEG dual channel mode. In this case, the MPEG-1 base stream does not contain Left and Right channels of a stereo pair but instead contains two independent mono channels. One of these channels can be selected through the proper request (Dual Channel Control) and reproduced over the Left and Right output channels simultaneously.

Bit D6 indicates that the decoder supports the DVD MPEG-2 augmentation to 7.1 channels instead of the standard 5.1 channels.

Bit D6 indicates that the multilingual information that is encoded at normal sampling rates (32 kHz, 44.1 kHz or 48 kHz). This bit is only applicable for MPEG-2 decoders.

Bit D7 indicates that the decoder is capable of processing streams that are encoded using adaptive multi-channel prediction.

Bits D9..8 indicate if the decoder can process embedded multilingual information. Multilingual capabilities can consist of being able to process multilingual information encoded at the same sampling frequency as the main audio channels (D9..8 = ‘01’). Some decoders may provide the additional capability to process multilingual information encoded at half the sampling frequency of the main audio channels (D9..8 = ‘11’).

Bits D15..10 are reserved for future extensions.

The **bmMPEGFeatures** field indicates compression-related features.

Bits D5..4 report which type of Dynamic Range Control the MPEG decoder supports. Some decoders do not implement DRC (D5..4 = ‘00’). If implemented, the DRC can either use the stream embedded gain parameters as is (D5..4 = ‘01’) or can provide for additional DRC scaling factors, either a single scaling factor that influences both the boost and cut value simultaneously (D5..4 = ‘10’) or a separate scaling factor for the boost and the cut value (D5..4 = ‘11’)

All other bits are reserved.

Table 2-7: MPEG Format-Specific Descriptor

Offset	Field	Size	Value	Description
0	bLength	1	Number	Size of this descriptor, in bytes: 9
1	bDescriptorType	1	Constant	CS_INTERFACE descriptor type.
2	bDescriptorSubtype	1	Constant	FORMAT_SPECIFIC descriptor subtype.
3	wFormatTag	2	Constant	MPEG. Constant identifying the precise format the AudioStreaming interface is using.

Offset	Field	Size	Value	Description
5	bmMPEGCapabilities	2	Bitmap	Bitmap identifying the MPEG capabilities of the decoder. A bit set indicates that the capability is supported: D2..0: Layer support: D0 = Layer I D1 = Layer II D2 = Layer III D3: MPEG-1 only. D4: MPEG-1 dual-channel. D5: MPEG-2 second stereo. D6: MPEG-2 7.1 channel augmentation. D7: Adaptive multi-channel prediction. D9..8: MPEG-2 multilingual support: 00 = Not supported 01 = Supported at Fs 10 = Reserved 11 = Supported at F_s and ½F_s. D15..10: Reserved.
7	bmMPEGFeatures	1	Bitmap	Bitmap identifying the features the decoder supports. A bit set indicates that the feature is supported: D3..0: Reserved. D5..4: Internal Dynamic Range Control: 00 = not supported. 01 = supported but not scalable. 10 = scalable, common boost and cut scaling value. 11 = scalable, separate boost and cut scaling value. D7..6: Reserved.

2.3.8.1.2 MPEG Format-Specific Requests

The following paragraphs describe the Set and Get MPEG Control requests. Some of the requests control parameters that are also dependent on the content of the incoming MPEG data stream. In general, the behavior of the MPEG decoder is primarily controlled by the incoming bitstream. Parameters set using MPEG Control requests retain their setting, even if that setting is not applicable to the current incoming bitstream. As an example, consider a decoder that is receiving a stream containing two independent stereo channel pairs. In this case, the Select Second Stereo Control can be enabled so that the second stereo channel is reproduced over the Left and Right channel. If the incoming stream is now switched to a full 5.1 encoded stream, the Select Second Stereo Control has no more influence and the decoder overrides its setting and produces full 5.1 sound. However, if the incoming stream switches back to the previous format, the Select Second Stereo Control becomes active again and resumes its previous setting so that the second stereo channel is reproduced again over the Left and Right channel.

2.3.8.1.2.1 Set MPEG Control Request

This request is used to set an attribute of an MPEG Control inside an AudioStreaming interface of the audio function.

Table 2-8: Set MPEG Control Request Values

bmRequest Type	bRequest	wValue	wIndex	wLength	Data
00100001B	SET_CUR SET_MIN SET_MAX SET_RES	CS	Zero and Interface	Length of parameter block	Parameter plock

The **bRequest** field indicates which attribute the request is manipulating. The MIN, MAX and RES attributes are usually not supported for the Set request. Further details on which attributes are supported for which Controls can be found in Section 2.3.8.1.2.3, “MPEG Controls.”

The **wValue** field specifies the Control Selector (CS) in the high byte and zero in the low byte. The Control Selector indicates which type of control this request is manipulating. If the request specifies an unknown or unsupported CS to that interface, the control pipe must indicate a stall.

For a description of the parameter blocks for the different Controls that can be addressed through the Set AC-3 Control request, see Section 2.3.8.1.2.3, “MPEG Controls.”

2.3.8.1.2.2 Get MPEG Control Request

This request returns the attribute setting of a specific MPEG Control inside an AudioStreaming interface of the audio function.

Table 2-9: Get MPEG Control Request Values

bmRequest Type	bRequest	wValue	wIndex	wLength	Data
10100001B	GET_CUR GET_MIN GET_MAX GET_RES	CS	Zero and Interface	Length of parameter block	Parameter block

The **bRequest** field indicates which attribute the request is reading.

The **wValue** field specifies the Control Selector (CS) in the high byte and zero in the low byte. The Control Selector indicates which type of control this request is addressing. If the request specifies an unknown or unsupported CS to that interface, the control pipe must indicate a stall.

For a description of the parameter blocks for the different Controls that can be addressed through the Get AC-3 Control request, see Section 2.3.8.1.2.3, “MPEG Controls.”

2.3.8.1.2.3 MPEG Controls

The following paragraphs present a detailed description of all possible AC-3 Controls an AudioStreaming interface can incorporate. For each Control, the layout of the parameter block together with the appropriate Control Selector is listed. The Control Selector codes are defined in Section A.3.1, “MPEG Control Selectors.”

2.3.8.1.2.3.1 Dual Channel Control

The Dual Channel Control is used to select which of the two available channels in the MPEG-1 base stream is actually retrieved and reproduced over the Left and Right output channels. If this Control is addressed on a decoder that does not implement Dual Channel Control (D4 = '0' in the **bmMPEGCapabilities** field of the MPEG format-specific descriptor), the control pipe must indicate a stall.

The Dual Channel Control can have only the current setting attribute (CUR). The position of the Channel2Enable switch can be either TRUE or FALSE. When FALSE, Channel I is selected, and when TRUE, Channel II is selected. The current setting of the Control can be queried using a Get MPEG Control request.

Table 2-10: Dual Channel Control Parameter Block

Control Selector		MP_DUAL_CHANNEL_CONTROL		
wLength		1		
Offset	**Field**	**Size**	**Value**	**Description**
0	BChannel2Enable	1	Number	The setting for the attribute of the Dual Channel Control. Channel I selected when FALSE, Channel II selected when TRUE.

2.3.8.1.2.3.2 Second Stereo Control

The Second Stereo Control is used to select the second stereo channel pair that can be encoded in an MPEG-2 stream instead of the multi-channel stereophonic information (3/2). If this Control is addressed on a decoder that does not implement Second Stereo support (D5 = '0' in the **bmMPEGCapabilities** field of the MPEG format-specific descriptor), the control pipe must indicate a stall.

The Second Stereo Control can have only the current setting attribute (CUR). The position of the 2ndStereoEnable switch can be either TRUE or FALSE. When FALSE, the main stereo channel pair is selected; when TRUE, the second stereo channel pair is selected. The current setting of the Control can be queried using a Get MPEG Control request.

Table 2-11: Second Stereo Control Parameter Block

Control Selector		MP_SECOND_STEREO_CONTROL		
wLength		1		
Offset	**Field**	**Size**	**Value**	**Description**
0	B2ndStereoEnable	1	Number	The setting for the attribute of the Second Stereo Control. Main stereo channel pair selected when FALSE, second stereo channel pair selected when TRUE.

2.3.8.1.2.3.3 Multilingual Control

The Multilingual Control is used to select the multilingual channel actually retrieved from the MPEG stream. If this Control is addressed on a decoder that does not implement multilingual support (D9..8 =

‘00’ in the **bmMPEGCapabilities** field of the MPEG format-specific descriptor), the control pipe must indicate a stall.

The Multilingual Control supports only the CUR Control attribute. The valid range is from zero (0x00) to seven (0x07). The actual range depends on the incoming MPEG stream. It may contain only a limited number of multilingual channels (less than seven). The Multilingual Control honors the request to the best of its abilities. It may truncate the attribute values to its closest available settings. It will report these settings when queried during a Get MPEG Control request.

Table 2-12: Multilingual Control Parameter Block

Control Selector		MP_MULTILINGUAL_CONTROL		
wLength		1		
Offset	**Field**	**Size**	**Value**	**Description**
0	bMultiLingual	1	Number	The setting for the attribute of the multilingual channel selection: 0 = decode no channel 1..7 = decode channel 1..7 8..255 = reserved

2.3.8.1.2.3.4 Dynamic Range Control

The Dynamic Range Control (DRC) is used to enable or disable the Dynamic Range Control functionality of the decoder. If the decoder does not support Dynamic Range control (D5..4 = ‘00’ in the **bmMPEGFeatures** field of the MPEG format-specific descriptor), the control pipe must indicate a stall when receiving this request.

The Dynamic Range Control can have only the current setting attribute (CUR). The position of the DRC switch can be either TRUE or FALSE. TRUE means that the MPEG decoder is using the Dynamic Range control words (possibly with additional scaling) contained in the MPEG bit stream to control the audio dynamic range. FALSE means the control words are being ignored, and the original signal dynamic range is being reproduced. The current setting of the Control can be queried using a Get MPEG Control request.

Table 2-13: Dynamic Range Control Parameter Block

Control Selector		MP_DYN_RANGE_CONTROL		
wLength		1		
Offset	**Field**	**Size**	**Value**	**Description**
0	bEnable	1	Bool	The setting for the Dynamic Range Control CUR attribute. Enabled when TRUE, disabled when FALSE.

2.3.8.1.2.3.5 Scaling Control

The Scaling Control is used to manipulate the single scaling coefficient used by MPEG decoders that implement a common boost/cut scaling value for Dynamic Range Control (D5..4 = ‘10’ in the **bmMPEGFeatures** field of the MPEG format-specific descriptor). If this Control is addressed on a non-‘10’ decoder, the control pipe must indicate a stall.

The Scaling Control can support all possible Control attributes (CUR, MIN, MAX, and RES). The valid range for the CUR, MIN, MAX, and RES attributes is from zero (0x00) to 255/256 (0xFF). The Scaling Control honors the request to the best of its abilities. It may round the **bScale** attribute value to its closest available setting. It will report this rounded setting when queried during a Get MPEG Control request.

Table 2-14: Scaling Control Parameter Block

Control Selector		MP_SCALING_CONTROL		
wLength		1		
Offset	**Field**	**Size**	**Value**	**Description**
0	bScale	1	Number	The setting for the attribute of the Scaling Control.

2.3.8.1.2.3.6 High/Low Scaling Control

The High/Low Scaling Control is used to manipulate the two scaling coefficients used by MPEG decoders that implement an independent boost and cut scaling value for Dynamic Range Control (D5..4 = ‘11’ in the **bmMPEGFeatures** field of the MPEG format-specific descriptor). If this Control is addressed on a non-‘11’ decoder, the control pipe must indicate a stall.

The High/Low Scaling Control can support all possible Control attributes (CUR, MIN, MAX, and RES). The valid range for the CUR, MIN, MAX, and RES attributes is from zero (0x00) to 255/256 (0xFF). The High/Low Scaling Control honors the request to the best of its abilities. It may round the **bLowScale** and **bHighScale** attribute values to their closest available settings. It will report these rounded settings when queried during a Get MPEG Control request. The **bLowScale** value is used by the MPEG decoder to scale the Dynamic Range control words that apply a gain increase (for low sound levels). The **bHighScale** value is used by the MPEG decoder to scale the Dynamic Range control words that apply a gain reduction (for high level sounds).

Table 2-15: High/Low Scaling Control Parameter Block

Control Selector		MP_HILO_SCALING_CONTROL		
wLength		2		
Offset	**Field**	**Size**	**Value**	**Description**
0	bLowScale	1	Number	The setting for the attribute of the low level Scaling Control.
1	bHighScale	1	Number	The setting for the attribute of the high level Scaling Control.

2.3.8.2 AC-3 Format

In the current specification, only AC-3 decoding aspects are considered. Real-time AC-3 encoding peripherals are not (yet) available and consequently are not covered by this specification.

2.3.8.2.1 AC-3 Format-Specific Descriptor

The **wFormatTag** field is a duplicate of the **wFormatTag** field in the class-specific AudioStreaming interface descriptor. The same field is used here to identify the format-specific descriptor.

The **bmBSID** bitmap field describes which bit stream ID modes this decoder is capable of processing. BSID modes can range from 0 to 31. A bit set indicates that BSID mode [bit_position] is supported. Standard AC-3 decoders must be capable of processing at least BSID modes 0 to 8. Therefore, the lower 9 bits of the **bmBSID** field must be set.

The **bmAC3Features** bitmap field indicates compression-related features.

Bits D3..0 indicate which mode the decoder supports. To ease the design of decoder products, Dolby Digital ICs offer standard operating modes called “Line Mode” and “RF Mode.” These modes are included within the Dolby Digital decoder IC itself, thus greatly simplifying the implementation of dialog normalization, dynamic range control and downmixing functions, all of which are necessary in Dolby Digital products. Two “Custom Modes” offer additional design flexibility aimed at more esoteric audio products for which additional implementation cost and complexity are not of primary concern.

Bits D5..4. indicate which type of Dynamic Range Control the AC-3 decoder supports. Some decoders do not implement DRC (D5..4 = ‘00’). If implemented, the DRC can either use the stream embedded gain parameters as is (D5..4 = ‘01’) or can provide for additional DRC scaling factors: either a single scaling factor that influences both the boost and cut value simultaneously (D5..4 = ‘10’), or a separate scaling factor for the boost and the cut value (D5..4 = ‘11’)

All other bits are reserved.

Table 2-16: AC-3 Format-Specific Descriptor

Offset	Field	Size	Value	Description
0	bLength	1	Number	Size of this descriptor, in bytes: 10
1	bDescriptorType	1	Constant	CS_INTERFACE descriptor type.
2	bDescriptorSubtype	1	Constant	FORMAT_SPECIFIC descriptor subtype.
3	wFormatTag	2	Constant	AC-3. Constant identifying the precise format the AudioStreaming interface is using.
5	bmBSID	4	Bitmap	A bit set to 1 indicates that the corresponding BSID mode is supported.
9	bmAC3Features	1	Bitmap	A bit set to 1 indicates that the mentioned feature is supported: D0: RF mode D1: Line mode D2: Custom0 mode D3: Custom1 mode D5..4: Internal Dynamic Range Control: 00 = not supported. 01 = supported but not scalable. 10 = scalable, common boost and cut scaling value. 11 = scalable, separate boost and cut scaling value. D7..6: Reserved

2.3.8.2.2 AC-3 Format-Specific Requests

The following paragraphs describe the Set and Get AC-3 Control requests.

2.3.8.2.2.1 Set AC-3 Control Request

This request is used to set an attribute of an AC-3 Control inside an AudioStreaming interface of the audio function.

Table 2-17: Set AC-3 Control Request Values

bmRequest Type	bRequest	wValue	wIndex	wLength	Data
00100001B	SET_CUR SET_MIN SET_MAX SET_RES	CS	Zero and Interface	Length of Parameter block	Parameter Block

The **bRequest** field indicates which attribute the request is manipulating. The MIN, MAX, and RES attributes are usually not supported for the Set request. Further details on which attributes are supported for which Controls can be found in Section 2.3.8.2.2.3, “AC-3 Controls.”

The **wValue** field specifies the Control Selector (CS) in the high byte and zero in the low byte. The Control Selector indicates which type of control this request is manipulating. If the request specifies an unknown or unsupported CS to that interface, the control pipe must indicate a stall.

For a description of the parameter blocks for the different Controls that can be addressed through the Set AC-3 Control request, see Section 2.3.8.2.2.3, “AC-3 Controls.”

2.3.8.2.2.2 Get AC-3 Control Request

This request returns the attribute setting of a specific AC-3 Control inside an AudioStreaming interface of the audio function.

Table 2-18: Get AC-3 Control Request Values

bmRequest Type	bRequest	wValue	wIndex	wLength	Data
10100001B	GET_CUR GET_MIN GET_MAX GET_RES	CS	Zero and Interface	Length of parameter block	Parameter block

The **bRequest** field indicates which attribute the request is reading.

The **wValue** field specifies the Control Selector (CS) in the high byte and zero in the low byte. The Control Selector indicates which type of control this request is addressing. If the request specifies an unknown or unsupported CS to that interface, the control pipe must indicate a stall.

For a description of the parameter blocks for the different Controls that can be addressed through the Get AC-3 Control request, see Section 2.3.8.2.2.3, “AC-3 Controls.”

2.3.8.2.2.3 AC-3 Controls

The following paragraphs present a detailed description of all possible AC-3 Controls an AudioStreaming interface can incorporate. For each Control, the layout of the parameter block together with the

appropriate Control Selector are listed. The Control Selector codes are defined in Section A.3.2, “AC-3 Control Selectors.”

2.3.8.2.2.3.1 Mode Control

The Mode Control is used to change the compression mode of the AC-3 decoder in the AudioStreaming interface. A Mode Control can only support the CUR attribute. The valid range for the CUR attribute is described through the **bmComprFeatures** field of the AC-3 format-specific descriptor. Bits D3..0 describe which compression modes the AC-3 decoder supports. Valid values are:

- 0: RF mode
- 1: Line mode
- 2: Custom0 mode
- 3: Custom1 mode

If the Mode Control request specifies an unsupported mode, the control pipe must indicate a stall. The current setting can be queried during a Get AC-3 Control request.

Table 2-19: Mode Control Parameter Block

Control Selector		AC_MODE_CONTROL		
wLength		1		
Offset	**Field**	**Size**	**Value**	**Description**
0	bMode	1	Number	The setting for the attribute of the Compression Mode Control: 0: RF mode 1: Line mode 2: Custom0 mode 3: Custom1 mode All other values are reserved.

2.3.8.2.2.3.2 Dynamic Range Control

The Dynamic Range Control (DRC) is used to enable or disable the Dynamic Range Control functionality of the decoder. The Dynamic Range Control can have only the current setting attribute (CUR). The position of the Dynamic Range Control switch can be either TRUE or FALSE. TRUE means that the AC-3 decoder is using the Dynamic Range control words (possibly with additional scaling) contained in the AC-3 bit stream to control the audio dynamic range. FALSE means the control words are being ignored and the original signal dynamic range is being reproduced. The current setting of the Control can be queried using a Get AC-3 Control request.

Table 2-20: Dynamic Range Control Parameter Block

Control Selector	MP_DYN_RANGE_CONTROL
wLength	1

Offset	Field	Size	Value	Description
0	bEnable	1	Bool	The setting for the Dynamic Range Control CUR attribute. Enabled when TRUE, disabled when FALSE.

2.3.8.2.2.3.3 Scaling Control

The Scaling Control is used to manipulate the single scaling coefficient used by AC-3 decoders that implement a common boost/cut scaling value for Dynamic Range Control. (D5..4 = '10' in the **bmAC3Features** field of the AC-3 format-specific descriptor.) If this Control is addressed on a non-'10' decoder, the control pipe must indicate a stall.

The Scaling Control can support all possible Control attributes (CUR, MIN, MAX, and RES). The valid range for the CUR, MIN, MAX, and RES attributes is from zero (0x00) to 255/256 (0xFF). The Scaling Control honors the request to the best of its abilities. It may round the **bScale** attribute value to its closest available setting. It will report this rounded setting when queried during a Get AC-3 Control request.

Table 2-21: Scaling Control Parameter Block

Control Selector		AC_SCALING_CONTROL		
wLength		1		
Offset	**Field**	**Size**	**Value**	**Description**
0	bScale	1	Number	The setting for the attribute of the Scaling Control.

2.3.8.2.2.3.4 High/Low Scaling Control

The High/Low Scaling Control is used to manipulate the two scaling coefficients used by AC-3 decoders that implement an independent boost and cut scaling value for Dynamic Range Control. (D5..4 = '11' in the **bmAC3Features** field of the AC-3 format-specific descriptor.) If this Control is addressed on a non-'11' decoder, the control pipe must indicate a stall.

The High/Low Scaling Control can support all possible Control attributes (CUR, MIN, MAX, and RES). The valid range for the CUR, MIN, MAX, and RES attributes is from zero (0x00) to 255/256 (0xFF). The High/Low Scaling Control honors the request to the best of its abilities. It may round the **bLowScale** and **bHighScale** attribute values to their closest available settings. It will report these rounded settings when queried during a Get AC-3 Control request. The **bLowScale** value is used by the AC-3 decoder to scale the Dynamic Range control words which apply a gain increase (for low sound levels). The **bHighScale** value is used by the AC-3 decoder to scale the Dynamic Range control words which apply a gain reduction (for high level sounds).

Table 2-22: High/Low Scaling Control Parameter Block

Control Selector	AC_HILO_SCALING_CONTROL
wLength	2

Offset	Field	Size	Value	Description
0	bLowScale	1	Number	The setting for the attribute of the low-level Scaling Control.
1	bHighScale	1	Number	The setting for the attribute of the high-level Scaling Control.

2.4 Type III Formats

These formats are based upon the IEC1937 standard. The IEC1937 standard describes a method to transfer non-PCM encoded audio bitstreams over an IEC958 digital audio interface, together with the transfer of the accompanying “Channel Status” and “User Data.”

The IEC958 standard specifies a widely used method of interconnecting digital audio equipment with two-channel linear PCM audio. The IEC1937 standard describes a way in which the IEC958 interface shall be used to convey non-PCM encoded audio bit streams for consumer applications.

The same basic techniques used in IEC1937 are reused here to convey non-PCM encoded audio bit streams over a Type III formatted audio stream.

2.4.1 Type III Format Type Descriptor

The Type III Format Type is identical to the Type I PCM Format Type, set up for two-channel 16-bit PCM data. It therefore uses two audio subframes per audio frame. The subframe size is two bytes and the bit resolution is 16 bits. The Type III Format Type descriptor is identical to the Type I Format Type descriptor but with the **bNrChannels** field set to two, the **bSubframeSize** field set to two and the **bBitResolution** field set to 16. All the techniques used to correctly transport Type I PCM formatted streams over USB equally apply to Type III formatted streams.

The non-PCM encoded audio bitstreams that are transferred within the basic 16-bit data area of the IEC1937 subframes (time-slots 12 [LSB] to 27 [MSB]) are placed unaltered in the two available 16-bit audio subframes per audio frame of the Type III formatted USB stream. The additional information in the IEC1937 subframes (channel status, user bits etc.) is discarded. Refer to the IEC1937 standard for a detailed description of the exact contents of the subframes.

The layout of the Type III Format Type descriptor is given here for clarity. All preassigned fields have been filled in.

Table 2-23: Type III Format Type Descriptor

Offset	Field	Size	Value	Description
0	bLength	1	Number	Size of this descriptor, in bytes: 8+(ns*3)
1	bDescriptorType	1	Constant	CS_INTERFACE descriptor type.
2	bDescriptorSubtype	1	Constant	FORMAT_TYPE descriptor subtype.
3	bFormatType	1	Constant	FORMAT_TYPE_III. Constant identifying the Format Type the AudioStreaming interface is using.
4	bNrChannels	1	Number	Indicates the number of ‘virtual’ physical channels in the audio data stream. Must be set to two.

Offset	Field	Size	Value	Description
5	bSubframeSize	1	Number	The number of bytes occupied by one audio subframe. Must be set to 2.
6	bBitResolution	1	Number	The number of effectively used bits from the available bits in an audio subframe.
7	bSamFreqType	1	Number	Indicates how the sampling frequency can be programmed: 0: Continuous sampling frequency 1..255: The number of discrete sampling frequencies supported by the isochronous data endpoint of the AudioStreaming interface (ns)
8...				See sampling frequency tables, below.

Depending on the value in the **bSamFreqType** field, the layout of the next part of the descriptor is as shown in the following tables.

Table 2-24: Continuous Sampling Frequency

Offset	Field	Size	Value	Description
8	tLowerSamFreq	3	Number	Lower bound in Hz of the sampling frequency range for this isochronous data endpoint.
11	tUpperSamFreq	3	Number	Upper bound in Hz of the sampling frequency range for this isochronous data endpoint.

Table 2-25: Discrete Number of Sampling Frequencies

Offset	Field	Size	Value	Description
8	tSamFreq [1]	3	Number	Sampling frequency 1 in Hz for this isochronous data endpoint.
…	…	…	…	…
8+(ns-1)*3	tSamFreq [ns]	3	Number	Sampling frequency ns in Hz for this isochronous data endpoint.

Note: In the case of adaptive isochronous data endpoints that support only a discrete number of sampling frequencies, the endpoint must at least tolerate ±1000 PPM inaccuracy on the reported sampling frequencies.

3 Adding New Audio Data Formats

Adding new Audio Data Formats to this specification is achieved by proposing a fully documented Audio Data Format to the Audio Device Class Working Group. Upon acceptance, they will register the new Audio Data Format (attribute a unique wFormatTag) and update this document accordingly. This process will also guarantee that new releases of generic USB audio drivers will support the newly registered Audio Data Formats.

It is always possible to use vendor-specific definitions if the above procedure is considered unsatisfactory.

Appendix A. Additional Audio Device Class Codes

A.1 Audio Data Format Codes

A.1.1 Audio Data Format Type I Codes

Table A-1: Audio Data Format Type I Codes

Name	wFormatTag
TYPE_I_UNDEFINED	0x0000
PCM	0x0001
PCM8	0x0002
IEEE_FLOAT	0x0003
ALAW	0x0004
MULAW	0x0005

A.1.2 Audio Data Format Type II Codes

Table A-2: Audio Data Format Type II Codes

Name	wFormatTag
TYPE_II_UNDEFINED	0x1000
MPEG	0x1001
AC-3	0x1002

A.1.3 Audio Data Format Type III Codes

Table A-3: Audio Data Format Type III Codes

Name	wFormatTag
TYPE_III_UNDEFINED	0x2000
IEC1937_AC-3	0x2001
IEC1937_MPEG-1_Layer1	0x2002
IEC1937_MPEG-1_Layer2/3 or IEC1937_MPEG-2_NOEXT	0x2003
IEC1937_MPEG-2_EXT	0x2004
IEC1937_MPEG-2_Layer1_LS	0x2005

Name	wFormatTag
IEC1937_MPEG-2_Layer2/3_LS	0x2006

A.2 Format Type Codes

Table A-4: Format Type Codes

Format Type Code	Value
FORMAT_TYPE_UNDEFINED	0x00
FORMAT_TYPE_I	0x01
FORMAT_TYPE_II	0x02
FORMAT_TYPE_II	0x03

A.3 Format-Specific Control Selectors

A.3.1 MPEG Control Selectors

Table A-5: MPEG Control Selectors

Control Selector	Value
MPEG_CONTROL_UNDEFINED	0x00
MP_DUAL_CHANNEL_CONTROL	0x01
MP_SECOND_STEREO_CONTROL	0x02
MP_MULTILINGUAL_CONTROL	0x03
MP_DYN_RANGE_CONTROL	0x04
MP_SCALING_CONTROL	0x05
MP_HILO_SCALING_CONTROL	0x06

A.3.2 AC-3 Control Selectors

Table A-6: AC-3 Control Selectors

Control Selector	Value
AC_CONTROL_UNDEFINED	0x00
AC_MODE_CONTROL	0x01
AC_DYN_RANGE_CONTROL	0x02
AC_SCALING_CONTROL	0x03

Control Selector	Value
AC_HILO_SCALING_CONTROL	0x04