IEC 62379-2

Edition 1.0 2008-09

INTERNATIONAL STANDARD

Common control interface for networked digital audio and video products – Part 2: Audio

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland
Email: inmail@iec.ch
Web: www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

- Catalogue of IEC publications: www.iec.ch/searchpub

The IEC on-line Catalogue enables you to search by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, withdrawn and replaced publications.

- IEC Just Published: www.iec.ch/online_news/justpub

Stay up to date on all new IEC publications. Just Published details twice a month all new publications released. Available on-line and also by email.

- Electropedia: www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 20 000 terms and definitions in English and French, with equivalent terms in additional languages. Also known as the International Electrotechnical Vocabulary online.

- Customer Service Centre: www.iec.ch/webstore/custserv

If you wish to give us your feedback on this publication or need further assistance, please visit the Customer Service Centre FAQ or contact us:

Email: csc@iec.ch
Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00

IEC 62379-2

Edition 1.0 2008-09

INTERNATIONAL STANDARD

Common control interface for networked digital audio and video products – Part 2: Audio

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

PRICE CODE XC

ICS 33.160.01

ISBN 2-8318-9987-7

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

COMMON CONTROL INTERFACE FOR NETWORKED DIGITAL AUDIO AND VIDEO PRODUCTS –

Part 2: Audio

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with an IEC Publication.
6) All users should ensure that they have the latest edition of this publication.
7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 62379-2 has been prepared technical area 4: Digital system interfaces and protocols, of IEC technical committee 100: Audio, video and multimedia systems and equipment.

The text of this standard is based on the following documents:

FDIS	Report on voting
100/1405/FDIS	100/1445/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

The committee has decided that the contents of this publication will remain unchanged until the maintenance result date indicated on the IEC web site under "http://webstore.iec.ch" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

A bilingual version of this publication may be issued at a later date.

INTRODUCTION

IEC 62379 specifies the common control interface, a protocol for managing equipment which conveys audio and/or video across digital networks.

This part of IEC 62379 specifies those aspects that are specific to audio equipment.

An introduction to the common control interface is given in IEC 62739-1.

COMMON CONTROL INTERFACE FOR NETWORKED DIGITAL AUDIO AND VIDEO PRODUCTS –

Part 2: Audio

1 Scope

This part of IEC 62379 specifies aspects of the common control interface of IEC 62379-1 that are specific to audio.

2 Normative references

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

AES3-2003, *AES standard for digital audio — Digital input-output interfacing — Serial transmission format for two-channel linearly represented digital audio data*

AES10-2003*, AES recommended practice for digital audio engineering – Serial multichannel audio digital interface (MADI)*

AES50-2005, *AES standard for digital audio engineering – High-resolution multi-channel audio interconnection (HRMAI)*

IEC 62379-1:2007, *Common control interface for networked audio and video products – Part 1: General*

ITU-T Recommendation G.711, *Pulse code modulation (PCM) of voice frequencies*

ITU-T Recommendation G.722, *7kHz audio-coding within 64 kbit/s*

ITU-T Recommendation J.41, *Characteristics of equipment for the coding of analogue high quality sound programme signals for transmission on 384 kbit/s channels*

ITU-T Recommendation J.57, *Transmission of digital studio quality sound signals over H1 channels*

3 Terms, definitions and abbreviations

For the purposes of this document, the terms and definitions given in IEC 62379-1 apply along with the following abbreviations.

3.1 Abbreviations

3.1.1
pulse code modulation
PCM

3.1.2
motion pictures expert group
MPEG

3.1.3
advanced audio coding
AAC

4 Audio format definitions

4.1 Audio signal format definitions

At any point in the audio signal chain, the audio data will be in a particular format. For management purposes, the format shall be identified by an object identifier, either a “common control interface standard” object identifier defined in this standard or an object identifier defined elsewhere.

NOTE Permitting audio format identifiers to be defined outside this standard allows use of proprietary formats within the standard protocol and also allows industry standard formats to emerge that may eventually be incorporated into future revisions of this standard.

4.1.1 Audio parameters

4.1.1.1 General

The definitions in 4.1.2 make reference to parameters which provide additional information about the format. These parameters shall be mapped to sub-identifier values as specified in the other subclauses of 4.1.1. Any parameter may be unspecified.

The sub-identifier values shall be appended to the object identifiers as additional arcs, in the order in which the parameters are listed in the relevant subclause of 4.1.2; except that if a parameter is unspecified and either it is the last parameter or all subsequent parameters are also unspecified, then it shall be omitted.

NOTE For all parameters, "unspecified" is coded as zero, so this rule ensures that the OID does not end with a zero arc.

EXAMPLE: If the last two parameters are bit depth and sampling frequency, then 16-bit 48kHz would be coded as .16.48000, 16-bit (with sampling frequency unspecified) as .16, and 48kHz (with bit depth unspecified) as .0.48000.

4.1.1.2 Channel arrangement

The sub-identifier for channel arrangement shall be a value of the following type:

```
ChannelArrangement ::= INTEGER {
  unspecified          (0),
  discreteMono         (1),
  stereo               (2),
  jointStereo          (3),
  surround             (4),
  surroundWithDownmix (5)
} (unspecified.. surroundWithDownmix)
```

`discreteMono` shall indicate that each channel (if there is more than one) is a separate audio signal.

`stereo` should only be used with an even number of channels, and shall indicate that each pair of channels (if there is more than one pair) is a separate stereo audio signal, with the first channel of the pair being the left channel.

`jointStereo` should only be used with an even number of channels, and shall indicate that each pair of channels (if there is more than one pair) is a separate stereo audio signal, with the first channel of the pair being the M channel and the second the S channel.

`surround` and `surroundWithDownmix` should only be used with 3 to 8 channels (inclusive), Each shall indicate that the first six channels contain left, right, centre, low frequency effects, left surround, and right surround respectively. `surround` shall indicate that the next two channels contain left and right channels (respectively) of a stereo signal suitable for matrix decoding. `surroundWithDownmix` shall indicate that the next two channels contain left and right channels (respectively) of a stereo downmix.

Where more than one arrangement can be used to describe a format, the smallest applicable value should be used.

EXAMPLE 1 A single stereo pair (2 channels) could be described by values `stereo (2)`, `surround (4)`, or `surroundWithDownmix (5)`. The smallest of these values, i.e. 2, should be used.

EXAMPLE 2 Surround sound with no accompanying stereo signal (6 channels) could be described by values `surround (4)`, or `surroundWithDownmix (5)`. The smaller of these values, i.e. 4, should be used.

4.1.1.3 Number of channels

The sub-identifier for number of channels shall be a value of the following type:

```
NumberChannels ::= INTEGER
-- An integer representing the number of audio channels
-- A value of zero shall indicate unspecified.
```

4.1.1.4 Bit depth

The sub-identifier for bit depth shall be a value of the following type:

```
BitDepth ::= INTEGER
-- An integer representing the audio bit depth in bits per sample.
-- A value of zero shall indicate unspecified.
```

4.1.1.5 Sampling frequency

The sub-identifier for sampling frequency shall be a value of the following type:

```
SamplingFrequency ::= INTEGER
-- An integer representing the audio sampling frequency in Hz
-- A value of zero shall indicate unspecified.
```

4.1.1.6 Bit rate

The sub-identifier for bit rate shall be a value of the following type:

```
BitRate ::= INTEGER
-- An integer representing the bit rate of the encoded signal in bits per
--      second.
-- A value of zero shall indicate unspecified.
```

4.1.2 Audio signal formats

Audio signal formats shall be rooted at the following location in the MIB tree:

```
iec62379             OBJECT IDENTIFIER ::= { iso(1) standard(0) 62379 }

audioFormat          OBJECT IDENTIFIER ::= { iec62379 audio(2) format(2) }

audioSignalFormat    OBJECT IDENTIFIER ::= { audioFormat signal(1) }
```

The following definitions shall be used to identify the specified formats.

NOTE Annex A contains an exemplar set of formats defined by this standard.

4.1.2.1 Unspecified audio

```
unspecifiedAudio     OBJECT IDENTIFIER ::=
                                        { audioSignalFormat unspecified(0) }
-- wildcard – any supported format allowed
```

4.1.2.2 No audio

```
noAudio              OBJECT IDENTIFIER ::= { audioSignalFormat none(1) }
-- indicates the output is muted
```

4.1.2.3 Analogue audio

```
analogueAudio         OBJECT IDENTIFIER ::= { audioSignalFormat analogue(2)
                }
-- analogue audio
```

The analogue audio identifier shall have two parameters. The first shall be the channel arrangement and the second shall be the number of channels.

4.1.2.4 PCM audio

```
pcmAudio             OBJECT IDENTIFIER ::= { audioSignalFormat pcm(3) }
-- linear PCM audio
```

The linear PCM format identifier shall have four parameters. The first shall be the channel arrangement, the second shall be the number of channels, the third shall be the bit depth and the fourth shall be the sampling frequency.

4.1.2.5 MPEG-1 layer 2 audio

```
mp2Audio             OBJECT IDENTIFIER ::= { audioSignalFormat mp2(4) }
-- MPEG-1 layer 2 audio
```

The MP2 format identifier shall have four parameters. The first shall be the channel arrangement, the second shall be the number of channels, the third shall be the sampling frequency, and the fourth shall be the bit rate.

4.1.2.6 MPEG-1 layer 3 audio

```
mp3Audio             OBJECT IDENTIFIER ::= { audioSignalFormat mp3(5) }
-- MPEG-1 layer 3 audio
```

The MP3 format identifier shall have four parameters. The first shall be the channel arrangement, the second shall be the number of channels, the third shall be the sampling frequency and the fourth shall be the bit rate.

4.1.2.7 AAC audio

```
aacAudio             OBJECT IDENTIFIER ::= { audioSignalFormat aac(6) }
-- AAC audio

aacLC                OBJECT IDENTIFIER ::= { aacAudio aacLC(1) }
-- AAC audio with the low complexity profile

aacMain              OBJECT IDENTIFIER ::= { aacAudio aacMain(2) }
-- AAC audio with the main profile

aacSRS               OBJECT IDENTIFIER ::= { aacAudio aacSRS(3) }
-- AAC audio with the sample-rate-scalable profile
```

```
aacLTP                OBJECT IDENTIFIER ::= { aacAudio aacLTP(4) }
-- AAC audio with the long term prediction profile

aacLD                 OBJECT IDENTIFIER ::= { aacAudio aacLD(5) }
-- AAC audio with the low delay profile
```

The AAC format identifier shall have four parameters. The first shall be the channel arrangement, the second shall be the number of channels, the third shall be the sampling frequency and the fourth shall be the bit rate.

4.1.2.8 Audio conforming to ITU-T Recommendation G.711

```
g711Audio             OBJECT IDENTIFIER ::= { audioSignalFormat g711(7) }
-- G711 audio

g711ALaw              OBJECT IDENTIFIER ::= { g711 aLaw(1) }
-- G711 audio, A-Law encoded

g711MuLaw             OBJECT IDENTIFIER ::= { g711 muLaw(2) }
-- G711 audio, mu-Law encoded
```

4.1.2.9 Audio conforming to ITU-T Recommendation G.722

```
g722Audio             OBJECT IDENTIFIER ::= { audioSignalFormat g722(8) }
-- G722 audio
```

The G722 identifier shall have one parameter. This shall be the bit rate.

4.1.2.10 APT-X audio

```
aptXAudio             OBJECT IDENTIFIER ::= { audioSignalFormat aptX(9) }
-- apt-X audio
```

The APT-X format identifier shall have five parameters. The first shall be the channel arrangement, the second shall be the number of channels, the third shall be the bit depth, the fourth shall be the sampling frequency, and the fifth shall be the bit rate.

4.1.2.11 Enhanced APT-X audio

```
enhancedAptXAudio   OBJECT IDENTIFIER ::= { audioSignalFormat enhAptX(10) }
-- enhanced apt-X audio
```

The enhanced APT-X format identifier shall have five parameters. The first shall be the channel arrangement, the second shall be the number of channels, the third shall be the bit depth, the fourth shall be the sampling frequency and the fifth shall be the bit rate.

4.1.2.12 Audio conforming to ITU-T Recommendation J.41

```
j41Audio              OBJECT IDENTIFIER ::= { audioSignalFormat j41(11) }
-- J41 audio

j41ALawA              OBJECT IDENTIFIER ::= { j41 aLawA(1) }
-- J41 audio using A-law companding, variant A

j41ALawB              OBJECT IDENTIFIER ::= { j41 aLawB(2) }
-- J41 audio using A-law companding, variant B

j41Nic                OBJECT IDENTIFIER ::= { j41 nic(3) }
-- J41 audio using near instantaneous companding
```

4.1.2.13 Audio conforming to ITU-T Recommendation J.57

```
j57Audio              OBJECT IDENTIFIER ::= { audioSignalFormat j57(12) }
-- J57 audio

j57H11                OBJECT IDENTIFIER ::= { j57 h11(1) }
-- J57 audio using an H11 channel

j57H12                OBJECT IDENTIFIER ::= { j57 h12(2) }
-- J57 audio using an H12 channel
```

4.1.2.14 Invalid audio

```
invalidAudio           OBJECT IDENTIFIER ::= { audioSignalFormat invalid(13)
                }
-- indicates an error, such as inability to decode a signal earlier in
-- the chain
```

4.2 Audio transport format definitions

Audio transport formats shall be rooted at the following location in the MIB tree:

```
audioTransportFormat OBJECT IDENTIFIER ::= { audioFormat transport(2) }
```

The following definitions shall be used to identify the specified transport formats.

```
unspecifiedTransport OBJECT IDENTIFIER ::=
                                    { audioTransportFormat unspecified(0) }

analogue    OBJECT IDENTIFIER ::= { audioTransportFormat analogue(1) }

AES3        OBJECT IDENTIFIER ::= { audioTransportFormat aes3(2) }

AES10       OBJECT IDENTIFIER ::= { audioTransportFormat aes10(3) }

AES50       OBJECT IDENTIFIER ::= { audioTransportFormat aes50(4) }
```

4.3 Audio metadata format definitions

Audio metadata formats shall be rooted at the following location in the MIB tree:

```
audioMetadataFormat  OBJECT IDENTIFIER ::= { audioFormat metadata(3) }
```

The following definitions shall be used to identify the specified metadata formats.

```
unspecifiedMetadata  OBJECT IDENTIFIER ::=
                                   { audioMetadataFormat unspecified(0) }
```

5 MIB definitions for audio blocks

5.1 General

This clause defines a set of managed object types for representing control functions in network controlled audio equipment. The format of the definitions is as specified in IEC 62379-1.

For management purposes, a piece of audio equipment shall be modelled as a number of discrete audio blocks and audio connectors, as specified in IEC 62379-1. Each audio block may have zero or more inputs and zero or more outputs, and each input or output may carry one or more channels. Each audio connector shall connect one audio block output to one audio block input with a one-to-one mapping of channels between the blocks.

NOTE 1 A piece of equipment may be fixed-function, in which case the number of audio blocks present and the connections between them will be immutable, or it may be programmable, in which case the number of audio blocks present and/or the connections between them may be changed by the user.

Each audio block shall be modelled either by one of the standard audio block types defined in this standard or by an audio block type defined elsewhere. Associated with each defined block type shall be a (possibly empty) group of managed object types that represent the control functions for that block. A block type shall be identified by the node in the object identifier tree that is the root node for the group of managed object types associated with that block type.

NOTE 2 Permitting audio block types to be defined outside this standard allows control of proprietary functions using the standard protocol and also allows industry standard block types to emerge that may eventually be incorporated into future revisions of this standard.

NOTE 3 An empty group of managed object types is permitted to allow for blocks that have no associated control functions.

NOTE 4 Annex E contains worked examples of the block structure.

5.2 Type definitions

In addition to the types defined in IEC 62379-1, the following types are used to specify the syntax of the abstract data structures representing managed object values.

5.2.1 Textual conventions

```
AudioTransportType ::= OBJECT IDENTIFIER
-- A reference to the transport used for an audio connection.
-- The value may be defined in 4.2, or in a sub-part of IEC 62379-5, or
-- elsewhere.

AudioLevel ::= INTEGER {
  mInfinity (-20000),
  fullScale (0),
  pInfinity (20000)
} (mInfinity..pInfinity)
-- An absolute or relative audio level in units of 0.01dB.

AudioPhase ::= INTEGER (-18000..18000)
-- An absolute or relative phase value in units of 0.01 degree.

AudioQuality ::= INTEGER {
  worst (1),
  low   (32),
  high  (96),
  best  (127)
} (worst..best)
-- An enumeration identifying an audio signal processing quality level.
```

```
-- Semantics are equipment specific. Equipment that supports selectable
-- quality levels shall as a minimum support the values low and high.


AudioChannel ::= INTEGER {
  left  (1),
  right (2)
} (1..240)
-- An enumeration identifying an audio channel. Stereo equipment shall
-- accept the values left and right. Multi-channel equipment should
-- accept a contiguous range of values starting from 1 and may disregard
-- the association of the values 1 and 2 with left and right.

Aes3ChannelData ::= OCTET STRING (SIZE(24))
-- The channel status data associated with an AES3 channel.

Aes3UserData ::= OCTET STRING (SIZE(24))
-- User data associated with an AES3 channel.

AudioClipSelector ::= INTEGER (1..2147483647)
-- An enumeration identifying an audio clip. Semantics are equipment
-- specific.

AudioClipActivation ::= INTEGER {
  play      (1),
  stop      (2),
  stopAtEnd (3)
} (play..stopAtEnd)
-- An enumeration for audio clip playback control.

AudioClipTrackControl ::= INTEGER {
  previous (1),
  next     (2)
} (previous..next)
-- An enumeration for audio clip track control.

AudioClipRepeat ::= INTEGER {
  none  (1),
  track (2),
  all   (3)
} (none..all)
-- An enumeration for audio clip repeat control.

AudioRecoveryMode ::= INTEGER {
  auto (1),
  slow (2),
  fast (3)
} (auto..fast)
-- An enumeration identifying an audio limiter recovery mode. Semantics
-- are equipment specific.

AudioAlarmStatus ::= INTEGER {
  ok      (1),
  warning (2),
  failure (3)
} (ok..failure)
-- An enumeration representing the status of an audio alarm

AudioLevelAlarmType ::= INTEGER {
  lower  (1),
  higher (2)
} (lower..higher)
-- An enumeration representing the trigger condition for an audio level
```

```
-- alarm. If lower, the alarm is triggered by the audio level being less
-- than the threshold; if higher, the alarm is triggered by the audio
-- level being greater than the threshold.
```

5.2.2 Sequences

```
APortEntry ::= SEQUENCE {
  aPortBlockId    BlockId,
  aPortDirection  PortDirection,
  aPortFormat     MediaFormat,
  aPortTransport  AudioTransportType,
  aPortName       Utf8String
}

Aes3DataEntry ::= SEQUENCE {
  aes3BlockId        BlockId,
  aes3ChannelId      AudioChannel,
  aes3ChannelData    Aes3ChannelData,
  aes3UserData       Aes3UserData,
  aes3ValidityError  TruthValue
}

APhantomEntry ::= SEQUENCE {
  aPhantomBlockId  BlockId,
  aPhantomEnabled  TruthValue,
  aPhantomLevel    CardinalNumber
}

ALockedEntry ::= SEQUENCE {
  aLockedBlockId          BlockId,
  aLockedTime             CardinalNumber,
  aLockedSamplesInserted  CardinalNumber,
  aLockedSamplesDropped   CardinalNumber
}

AMixerBlockEntry ::= SEQUENCE {
  aMixerBlockId       BlockId,
  aMixerFadeDuration  CardinalNumber,
  aMixerFadeNow       TruthValue
}

AMixerInputEntry ::= SEQUENCE {
  aMixerInputBlockId      BlockId,
  aMixerInputNumber       IndexNumber,
  aMixerInputLevel        AudioLevel,
  aMixerInputFadeToLevel  AudioLevel,
  aMixerInputDelay        CardinalNumber
}

ACrosspointBlockEntry ::= SEQUENCE {
  aCrosspointBlockId    BlockId,
  aCrosspointConfigure  TruthValue,
  aCrosspointCopy       BlockId
}

ACrosspointPathEntry ::= SEQUENCE {
  aCrosspointPathBlockId   BlockId,
  aCrosspointPathSrc       AudioChannel,
  aCrosspointPathDst       AudioChannel,
  aCrosspointPathGain      AudioLevel,
  aCrosspointPathNewGain   AudioLevel,
  aCrosspointPathPhase     AudioPhase,
  aCrosspointPathNewPhase  AudioPhase
```

```
}

AClipPlayerBlockEntry ::= SEQUENCE {
  acpBlockId              BlockId,
  acpSelection            AudioClipSelector,
  acpMainFaderLevelNow    AudioLevel,
  acpMainFaderLevelStart  AudioLevel,
  acpMainFaderLevelStop   AudioLevel,
  acpMainFaderDuration    CardinalNumber,
  acpClipFaderLevelNow    AudioLevel,
  acpClipFaderLevelStart  AudioLevel,
  acpClipFaderLevelStop   AudioLevel,
  acpClipFaderDuration    CardinalNumber,
  acpStartDelay           IntegerNumber,
  acpActivation           AudioClipActivation,
  acpTrackControl         AudioClipTrackControl,
  acpTrackNumber          CardinalNumber,
  acpRepeat               AudioClipRepeat
}

AClipSequenceEntry ::= SEQUENCE {
  aClipSequenceBlockId  BlockId,
  aClipSequenceId       AudioClipSelector,
  aClipSequenceName     Utf8String
}

ALimiterBlockEntry ::= SEQUENCE {
  aLimiterBlockId       BlockId,
  aLimiterThreshold     AudioLevel,
  aLimiterAttackTime    CardinalNumber,
  aLimiterGainMakeup    AudioLevel,
  aLimiterRecoveryTime  CardinalNumber,
  aLimiterRecoveryMode  AudioRecoveryMode
}

AConverterBlockEntry ::= SEQUENCE {
  aConverterBlockId       BlockId,
  aConverterQuality       AudioQuality,
  aConverterEnabled       TruthValue,
  aConverterDithering     TruthValue,
  aConverterOutputFormat  MediaFormat,
  aConverterError         TruthValue
}

ALevelAlarmBlockEntry ::= SEQUENCE {
  alaBlockId       BlockId,
  alaType          AudioLevelAlarmType,
  alaThreshold     AudioLevel,
  alaWarningTime   CardinalNumber,
  alaFailureTime   CardinalNumber,
  alaCounter       CardinalNumber,
  alaEnabled       TruthValue,
  alaStatus        AudioAlarmStatus
}
```

5.3 Audio port and associated managed object type definitions

5.3.1 Generic port functionality

All audio inputs to and outputs from the unit shall be represented using an audio port block. A base table of managed objects provides control common to all ports; extension tables provide functionality specific to certain types of port.

An audio port block shall have the following structure, where c is the number of channels on the input or output:

IEC 1492/08

Figure 1 – Audio port blocks

The group of objects in Table 1 shall be implemented by all compliant audio equipment that contains one or more audio ports. The root node for these objects shall be

{ iso(1) standard(0) iec62379 audio(2) audioMIB(1) audioPort(1) }

This node shall be used as the audio block type identifier for audio port blocks.

Table 1 – Managed objects for audio ports

Identifier	Syntax	Index	Readable	Writable	Volatile	Status
aPortTable(1)	SEQUENCE OF APortEntry		none	none	no	m
└aPortEntry(1)	APortEntry		none	none	no	m
├aPortBlockId(1)	BlockId	yes	none	none	no	m
├aPortDirection(2)	PortDirection		listener	none	no	m
├aPortFormat(3)	MediaFormat		listener	none	yes	m
├aPortTransport(4)	AudioTransportType		listener	none	no	o
└aPortName(5)	Utf8String		listener	supervisor	no	o

5.3.1.1 aPortTable

A table of audio port descriptors for this unit. Each audio port on the unit has a corresponding entry in this table.

5.3.1.2 aPortEntry

An entry in the audio port table.

5.3.1.3 aPortBlockId

The block identifier for this port. Used as an index when accessing the audio port table.

5.3.1.4 aPortDirection

The direction (input or output) of this port.

5.3.1.5 aPortFormat

The format of the audio data currently being received or transmitted via this port. If the port is not active, the value `noAudio` shall be returned.

5.3.1.6 aPortTransport

The type of transport used by the port.

5.3.1.7 aPortName

The name assigned to this port. This is an arbitrary text string assigned by the system manager.

5.3.2 AES3 ancillary data

The group of objects in Table 2 shall be implemented by all compliant audio equipment that allows access to AES3 ancillary data. The root node for these objects shall be

{ iso(1) standard(0) iec62379 audio(2) audioMIB(1) audioPort(1) }

Table 2 – Managed objects for AES3 ancillary data

Identifier	Syntax	Index	Readable	Writable	Volatile	Status
aes3DataTable(2)	SEQUENCE OF Aes3DataEntry		none	none	no	m
└aes3DataEntry(1)	Aes3DataEntry		none	none	no	m
├aes3BlockId(1)	BlockId	yes	none	none	no	m
├aes3ChannelId(2)	AudioChannel	yes	none	none	no	m
├aes3ChannelData(3)	Aes3ChannelData		listener	none	yes	o
├aes3UserData(4)	Aes3UserData		listener	none	yes	o
└aes3ValidityError(5)	TruthValue		listener	supervisor	yes	o

5.3.2.1 aes3DataTable

A table of AES3 ancillary data descriptors for this unit. Each channel of each AES3 port on the unit has a corresponding entry in this table.

5.3.2.2 aes3DataEntry

An entry in the AES3 ancillary data table.

5.3.2.3 aes3BlockId

The block identifier of the associated audio port. Used as an index when accessing the AES3 ancillary data table.

5.3.2.4 aes3ChannelId

The channel number. Used as an index when accessing the AES3 ancillary data table.

5.3.2.5 aes3ChannelData

The current channel status data (24 bytes) for this channel.

5.3.2.6 aes3UserData

The current user data for this channel.

5.3.2.7 aes3ValidityError

An indication that shows whether this channel's validity bit has been set in any subframe. It shall be automatically set to `true` when a subframe with V=1 is detected. It shall only be set to `false` as a result of a `SET` operation on this object.

NOTE V=1 indicates an error in the case of linear PCM; for other formats V=1 should be set in every subframe.

5.3.3 Phantom power

The group of objects in Table 3 shall be implemented by all compliant audio equipment that provides phantom power on an audio port. The root node for these objects shall be

{ iso(1) standard(0) iec62379 audio(2) audioMIB(1) audioPort(1) }

Table 3 – Managed objects for phantom power

Identifier	Syntax	Index	Readable	Writable	Volatile	Status
aPhantomTable(3)	SEQUENCE OF APhantomEntry		none	none	no	m
└aPhantomEntry(1)	APhantomEntry		none	none	no	m
├aPhantomBlockId(1)	BlockId	yes	none	none	no	m
├aPhantomEnabled(2)	TruthValue		listener	operator	no	m
└aPhantomLevel(3)	CardinalNumber		listener	supervisor	no	o

5.3.3.1 aPhantomTable

A table of audio port phantom power descriptors for this unit. Each audio port on the unit that supports phantom power has a corresponding entry in this table.

5.3.3.2 aPhantomEntry

An entry in the phantom power table.

5.3.3.3 aPhantomBlockId

The block identifier of the associated audio port. Used as an index when accessing the phantom power table.

5.3.3.4 aPhantomEnabled

When set to `true`, phantom power is provided by this port. When set to `false`, phantom power is not provided by this port.

5.3.3.5 aPhantomLevel

The level of the phantom power provided in millivolts.

5.3.4 Audio locked to reference

The group of objects in Table 4 shall be implemented by all compliant audio equipment that provides statistics for audio signals being locked to a reference signal. The root node for these objects shall be

{ iso(1) standard(0) iec62379 audio(2) audioMIB(1) audioPort(1) }

Table 4 – Managed objects for audio locked

Identifier	Syntax	Index	Readable	Writable	Volatile	Status
aLockedTable(3) |	SEQUENCE OF ALockedEntry		none	none	no	m
└aLockedEntry(1)	ALockedEntry		none	none	no	m
├aLockedBlockId(1)	BlockId	yes	none	none	no	m
├aLockedTime(2)	CardinalNumber		listener	none	yes	m
├aLockedSamplesInserted(3)	CardinalNumber		listener	supervisor	yes	m
└aLockedSamplesDropped(4)	CardinalNumber		listener	supervisor	yes	m

5.3.4.1 aLockedTable

A table of audio port locked descriptors for this unit. Each audio port on the unit that supports measurement of lock against the reference has a corresponding entry in this table.

5.3.4.2 aLockedEntry

An entry in the locked table.

5.3.4.3 aLockedBlockId

The block identifier of the associated audio port. Used as an index when accessing the locked table.

5.3.4.4 aLockedTime

The time (in seconds) that the audio signal on this port has been locked to the unit's reference clock.

5.3.4.5 aLockedSamplesInserted

The number of audio samples that have been inserted due to buffer underrun or overrun on this port. The value shall be updated every 10 s by adding the number of samples inserted in the last 10 s to half the previous value.

5.3.4.6 aLockedSamplesDropped

The number of audio samples that have been dropped due to buffer underrun or overrun on this port. The value shall be updated every 10 s by adding the number of samples inserted in the last 10 s to half the previous value.

5.4 Other audio block and associated managed object type definitions

5.4.1 Audio mixer blocks

An audio mixer block shall have the following structure:

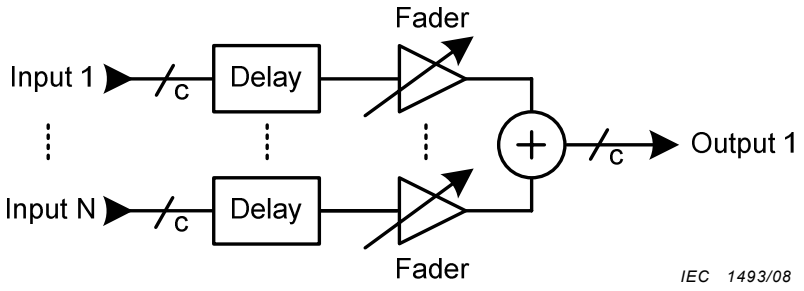


Figure 2 – Audio mixer block

where c is the number of channels on a connection.

NOTE 1 An audio mixer block may be used to represent a simple switched selector or combiner, by limiting the permitted values for the fader level controls to `mInfinity` or `fullScale`.

NOTE 2 The delay function permits audio streams that have passed through various processing or transport paths to be brought back into time alignment, either with other audio streams or with associated video streams. Equipment that does not support this functionality is represented as having a fixed zero delay.

The group of objects in Table 5 shall be implemented by all compliant audio equipment that has a management model that incorporates one or more audio mixer blocks. The root node for these objects shall be

{ iso(1) standard(0) iec62379 audio(2) audioMIB(1) audioMixer(2) }

This node shall be used as the block type identifier for audio mixer blocks.

Table 5 – Managed objects for audio mixer blocks

Identifier	Syntax	Index	Readable	Writable	Volatile	Status
aMixerBlockTable(1)	SEQUENCE OF AMixerBlockEntry		none	none	no	m
└aMixerBlockEntry(1)	AMixerBlockEntry		none	none	no	m
├aMixerBlockId(1)	BlockId	yes	none	none	no	m
├aMixerFadeDuration(2)	CardinalNumber		listener	operator	no	o
└aMixerFadeNow(3)	TruthValue		listener	operator	yes	o
aMixerInputTable(2)	SEQUENCE OF AMixerInputEntry		none	none	no	m
└aMixerInputEntry(1)	AMixerInputEntry		none	none	no	m
├aMixerInputBlockId(1)	BlockId	yes	none	none	no	m
├aMixerInputNumber(2)	IndexNumber	yes	none	none	no	m
├aMixerInputLevel(3)	AudioLevel		listener	operator	no	m
├aMixerInputFadeToLevel(4)	AudioLevel		listener	operator	no	o
└aMixerInputDelay(5)	CardinalNumber		listener	operator	no	o

5.4.1.1 aMixerBlockTable

A table of audio mixer block descriptors for this unit. Each audio mixer block in the unit has a corresponding entry in this table.

5.4.1.2 aMixerBlockEntry

An entry in the audio mixer block table.

5.4.1.3 aMixerBlockId

The block identifier for this block. Used as an index when accessing the audio mixer block table.

5.4.1.4 aMixerFadeDuration

The transition time (in milliseconds) for a smooth fade or cross-fade operation performed by this block. A value of zero indicates a hard switchover.

5.4.1.5 aMixerFadeNow

When set to `true`, causes the block to perform a smooth fade or cross-fade operation. Using values taken from this block's entries in the mixer input table, the unit shall simultaneously ramp the fader level for each input in `aMixerInputEntry` from its current value to the value specified by `aMixerInputFadeToLevel`. Automatically reset to `false` when the operation is complete.

5.4.1.6 aMixerInputTable

A table of audio mixer input descriptors for this unit. Each input of each audio mixer block in the unit has a corresponding entry in this table.

5.4.1.7 aMixerInputEntry

An entry in the audio mixer input table.

5.4.1.8 aMixerInputBlockId

The block identifier of the associated block. Used as an index when accessing the audio mixer input table.

5.4.1.9 aMixerInputNumber

The block input number for this input. Used as an index when accessing the audio mixer input table.

5.4.1.10 aMixerInputLevel

The fader level for this input. When a level is set, the fader immediately changes to the level specified, over the duration in `aMixerFadeDuration`. For blocks that only support switching between inputs, the only permitted values are `mInfinity` and `fullScale`. Blocks that automatically switch between inputs may reject `SET` operations on this object.

5.4.1.11 aMixerInputFadeToLevel

The fader level for this input that will be applied when `aMixerFadeNow` is set to `true`. For blocks that only support switching between inputs, the only permitted values are `mInfinity` and `fullScale`. Blocks that automatically switch between inputs may reject `SET` operations on this object.

5.4.1.12 aMixerInputDelay

The delay (in microseconds) applied to samples arriving at this input.

5.4.2 Audio crosspoint blocks

An audio crosspoint block shall have the following structure:

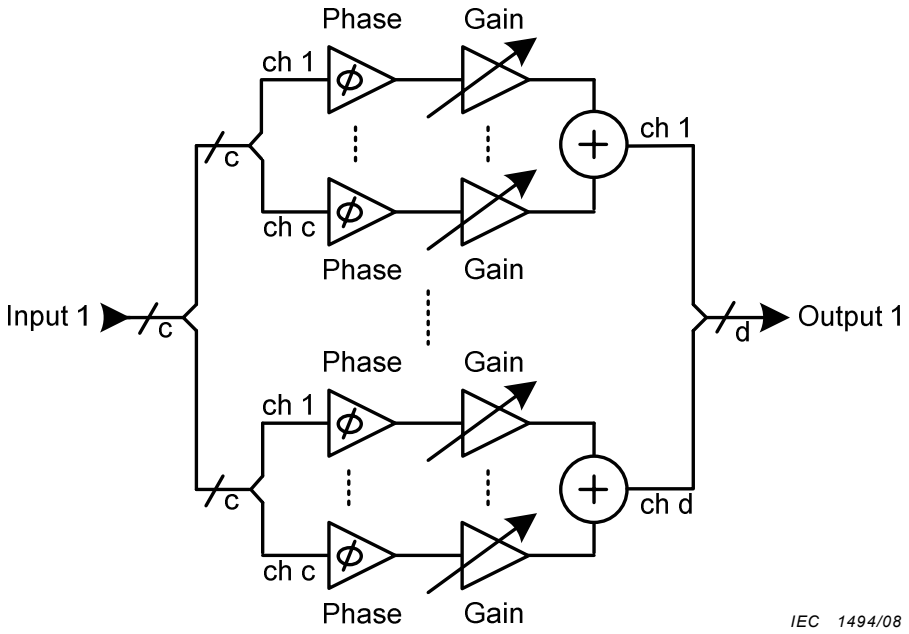


Figure 3 – Audio crosspoint block

where c is the number of input channels and d is the number of output channels.

The group of objects in Table 6 shall be implemented by all compliant audio equipment that has a management model that incorporates one or more audio crosspoint blocks. The root node for these objects shall be

{ iso(1) standard(0) iec62379 audio(2) audioMIB(1) audioCrosspoint(3) }

This node shall be used as the block type identifier for audio crosspoint blocks.

Table 6 – Managed objects for audio crosspoint blocks

Identifier	Syntax	Index	Readable	Writable	Volatile	Status
aCrosspointBlockTable(1)	SEQUENCE OF ACrosspointBlockEntry		none	none	no	m
└aCrosspointBlockEntry(1)	ACrosspointBlockEntry		none	none	no	m
├aCrosspointBlockId(1)	BlockId	yes	none	none	no	m
├aCrosspointConfigure(2)	TruthValue		listener	operator	yes	m
└aCrosspointCopy(3)	BlockId		none	operator	yes	o
aCrosspointPathTable(2)	SEQUENCE OF ACrosspointPathEntry		none	none	no	m
└aCrosspointPathEntry(1)	ACrosspointPathEntry		none	none	no	m
├aCrosspointPathBlockId(1)	BlockId	yes	none	none	no	m
├aCrosspointPathSrc(2)	AudioChannel	yes	none	none	no	m
├aCrosspointPathDst(3)	AudioChannel	yes	none	none	no	m
├aCrosspointPathGain(4)	AudioLevel		listener	operator	no	m
├aCrosspointPathNewGain(5)	AudioLevel		listener	operator	no	o
├aCrosspointPathPhase(6)	AudioPhase		listener	operator	no	m
└aCrosspointPathNewPhase (7)	AudioPhase		listener	operator	no	o

5.4.2.1 aCrosspointBlockTable

A table of audio crosspoint block descriptors for this unit. Each audio crosspoint block in the unit has a corresponding entry in this table.

5.4.2.2 aCrosspointBlockEntry

An entry in the audio crosspoint block table.

5.4.2.3 aCrosspointBlockId

The block identifier for this block. Used as an index when accessing the audio crosspoint block table.

5.4.2.4 aCrosspointConfigure

When set to `true`, causes this block to be configured as specified by the corresponding `aCrosspointPathNewGain` and `aCrosspointPathNewPhase` entries in the crosspoint path table. Automatically reset to `false` if the block supports delayed configuration and any of the corresponding entries in the crosspoint path table are modified. Permanently `true` if the block does not support delayed configuration.

5.4.2.5 aCrosspointCopy

When set to a block identifier that identifies another audio crosspoint block in the unit with an identical structure to this block, copies the values of `aCrosspointPathGain` and `aCrosspointPathPhase` for each path in the crosspoint from the identified block to this block. If set to a block identifier that does not identify an audio crosspoint block in the unit with an identical structure to this block, the `SET` operation shall be rejected.

NOTE A possible application is the ability to have some common configurations available as presets by creating 'dummy' crosspoint blocks with the required settings which are referenced in the block table but which are not actually part of the audio path.

5.4.2.6 aCrosspointPathTable

A table of audio crosspoint path descriptors for this unit. Each path in each audio crosspoint block in the unit has a corresponding entry in this table.

5.4.2.7 aCrosspointPathEntry

An entry in the audio crosspoint path table.

5.4.2.8 aCrosspointPathBlockId

The block identifier of the associated block. Used as an index when accessing the audio crosspoint path table.

5.4.2.9 aCrosspointPathSrc

The source audio channel associated with this path. Used as an index when accessing the audio crosspoint path table.

5.4.2.10 aCrosspointPathDst

The destination audio channel associated with this path. Used as an index when accessing the audio crosspoint path table.

5.4.2.11 aCrosspointPathGain

The signal gain from source to destination of this path. If this value is set, the change occurs immediately.

5.4.2.12 aCrosspointPathNewGain

The signal gain from source to destination of this path that will be applied when `aCrosspointConfigure` is set to `true`.

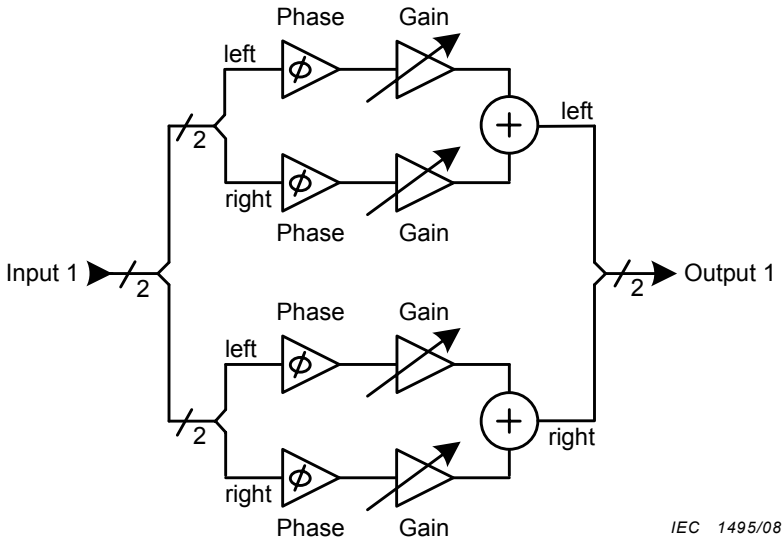
5.4.2.13 aCrosspointPathPhase

The phase shift from source to destination of this path. If this value is set, the change occurs immediately.

5.4.2.14 aCrosspointPathNewPhase

The phase shift from source to destination of this path that will be applied when `aCrosspointConfigure` is set to `true`.

EXAMPLE: The crosspoint block permits a variety of audio functions to be modelled. As an example, consider the following stereo crosspoint:



There are four routes through the crosspoint representing what effect an input channel (the source, represented by `aCrosspointPathSrc`) has on an output channel (the destination, represented by `aCrosspointPathDst`). The source / destination pairs represent:

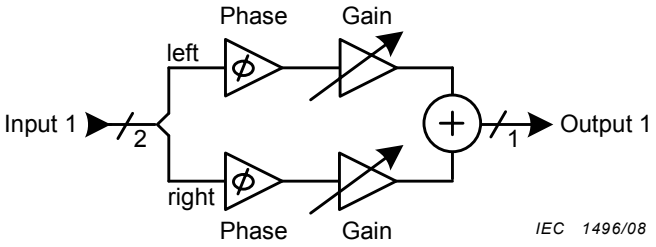
- the effect the left channel of the input has on the left channel of the output;
- the effect the right channel of the input has on the left channel of the output,
- the effect the left channel of the input has on the right channel of the output;
- the effect the right channel of the input has on the right channel of the output.

By setting different values for phase and gain for each source/destination pair the audio may be manipulated. For example, different gain values may be used to mono or leg-swap the incoming audio.

The following shows the fader settings (via `aCrosspointPathGain`) for different effects; *stereo* passes the audio unaltered, *swap* swaps the incoming left and right channels and *mono* places (M6) mono audio on both channels of the output.

Src	Dst	Gain		
		stereo	swap	mono
left	left	fullScale	mInfinity	-600
right	left	mInfinity	fullScale	-600
left	right	mInfinty	fullScale	-600
right	right	fullScale	mInfinity	-600

The number of input and output channels does not have to be the same, so the following models a block that takes a two channel stereo input to give a single channel mono output:



Src	Dest	Gain
left	left	-600
right	left	-600

5.4.3 Audio clip player blocks

An audio clip player block shall have the following structure:

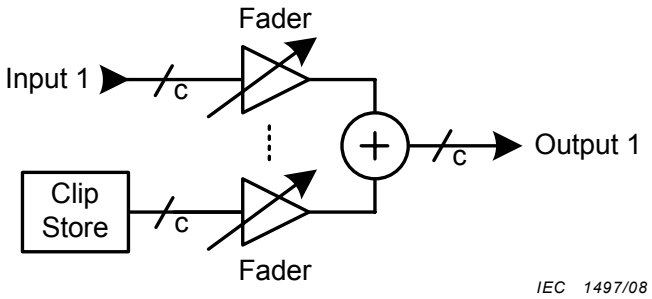


Figure 4 – Audio clip player block

where c is the number of channels on a connection. This represents a block that can either switch or mix a selected pre-stored audio sequence into an audio stream.

The group of objects in Table 7 shall be implemented by all compliant audio equipment that has a management model that incorporates one or more audio clip player blocks. The root node for these objects shall be

{ iso(1) standard(0) iec62379 audio(2) audioMIB(1) audioClipPlayer(4) }

This node shall be used as the block type identifier for audio clip player blocks.

Table 7 – Managed objects for audio clip player blocks

Identifier	Syntax	Index	Readable	Writable	Volatile	Status
aClipPlayerBlockTable(1)	SEQUENCE OF AClipPlayerBlockEntry		none	none	no	m
└aClipPlayerBlockEntry(1)	AClipPlayerBlockEntry		none	none	no	m
├acpBlockId(1)	BlockId	yes	none	none	no	m
├acpSelection(2)	AudioClipSelector		listener	operator	no	o
├acpMainFaderLevelNow(3)	AudioLevel		listener	operator	no	o
├acpMainFaderLevelStart(4)	AudioLevel		listener	operator	no	o
├acpMainFaderLevelStop(5)	AudioLevel		listener	operator	no	o
├acpMainFaderDuration(6)	CardinalNumber		listener	operator	no	o
├acpClipFaderLevelNow(7)	AudioLevel		listener	operator	no	o
├acpClipFaderLevelStart(8)	AudioLevel		listener	operator	no	o
├acpClipFaderLevelStop(9)	AudioLevel		listener	operator	no	o
├acpClipFaderDuration(10)	CardinalNumber		listener	operator	no	o
├acpStartDelay(11)	IntegerNumber		listener	operator	no	o
├acpActivation(12)	AudioClipActivation		listener	operator	yes	m
├acpTrackControl(13)	AudioClipTrackControl		none	operator	yes	o
├acpTrackNumber(14)	CardinalNumber		listener	operator	yes	o
└acpRepeat(15)	AudioClipRepeat		listener	operator	yes	o
aClipSequenceTable(2)	SEQUENCE OF AClipSequenceEntry		none	none	no	o
└aClipSequenceEntry(1)	AClipSequenceEntry		none	none	no	m
├aClipSequenceBlockId(1)	BlockId	yes	none	none	no	m
├aClipSequenceId(2)	AudioClipSelector	yes	none	none	no	m
└aClipSequenceName(3)	Utf8String		listener	operator	no	m

5.4.3.1 aClipPlayerBlockTable

A table of audio clip player block descriptors for this unit. Each audio clip player block in the unit has a corresponding entry in this table.

5.4.3.2 aClipPlayerBlockEntry

An entry in the audio clip player block table.

5.4.3.3 acpBlockId

The block identifier for this block. Used as an index when accessing the audio clip player block table.

5.4.3.4 acpSelection

The pre-stored sequence to be played by this block.

5.4.3.5 acpMainFaderLevelNow

The fader level for the main signal path. When a level is set, the fader immediately changes to the level specified, over the duration `acpMainFaderDuration`.

5.4.3.6 acpMainFaderLevelStart

The fader level for the main signal path when the clip player starts. The fader adopts this value automatically over the duration `acpMainFaderDuration`.

5.4.3.7 acpMainFaderLevelStop

The fader level for the main signal path when the clip player stops. The fader adopts this value automatically over the duration `acpMainFaderDuration`.

5.4.3.8 acpMainFaderDuration

The transition time (in milliseconds) for a smooth fade operation performed by the main fader. A value of zero indicates a hard switchover.

5.4.3.9 acpClipFaderLevelNow

The fader level for the pre-stored sequence. When a level is set, the fader immediately changes to the level specified, over the duration `acpClipFaderDuration`.

5.4.3.10 acpClipFaderLevelStart

The fader level for the pre-stored sequence when the clip player starts. The fader adopts this value automatically over the duration `acpClipFaderDuration`.

5.4.3.11 acpClipFaderLevelStop

The fader level for the pre-stored sequence when the clip player stops. The fader adopts this value automatically over the duration `acpClipFaderDuration`.

5.4.3.12 acpClipFaderDuration

The transition time (in milliseconds) for a smooth fade operation performed by the clip fader. A value of zero indicates a hard switchover.

5.4.3.13 acpStartDelay

The time offset (in milliseconds) between the clip player being activated and the pre-stored sequence being played. Equipment that supports scheduled operations may permit negative values.

5.4.3.14 acpActivation

When set to `play`, causes the clip player to start playing the selected pre-recorded sequence. Automatically reset to `stop` when playback is complete. Equipment may permit manual setting to `stop` to abort playback immediately, or to `stopAtEnd` to abort playback when the current track finishes. If the current state is `stopAtEnd`, setting to `stop` should cancel the stop at end of track request and stops playback immediately.

5.4.3.15 acpTrackControl

When set to `next`, causes the clip player to jump to the start of the next track in the selected sequence. If the clip player is playing the last track in the selected sequence, it may either jump to the first track in the sequence or reject the `SET` operation.

When set to `previous`, causes the clip player to jump to the start of the previous track in the selected sequence. If the clip player is playing the first tack in the selected sequence, it may either jump to the last track in the sequence or reject the `SET` operation.

5.4.3.16 acpTrackNumber

The number of the track currently being played. If the selected clip is not divided into tracks, the value 1 shall be returned. If the clip player is inactive, the value 0 shall be returned.

When set to a valid track number for the selected sequence, causes the clip player to jump to the start of that track. If set to an invalid track number, the `SET` operation shall be rejected.

5.4.3.17 acpTrackRepeat

When set to `none`, the clip player shall automatically stop playback when the end of the selected sequence is reached. When set to `all`, the clip player shall automatically jump back to the start of the sequence whenever the end of the selected sequence is reached. When set to `track`, the clip player shall automatically jump back to the start of the track whenever the end of a tack in the selected sequence is reached.

5.4.3.18 aClipSequenceTable

A table of audio clip sequence descriptors for this unit. Each audio clip sequence available for each audio clip player block in the unit has a corresponding entry in this table.

5.4.3.19 aClipSequenceEntry

An entry in the audio clip sequence table.

5.4.3.20 aClipSequenceBlockId

The block identifier for the associated audio clip player block. Used as an index when accessing the audio clip sequence table.

5.4.3.21 aClipSequenceId

The clip sequence identifier. Used as an index when accessing the audio clip sequence table.

5.4.3.22 aClipSequenceName

The name assigned to this clip sequence. This is an arbitrary text string.

5.4.4 Audio limiter blocks

An audio limiter block shall have the following structure:

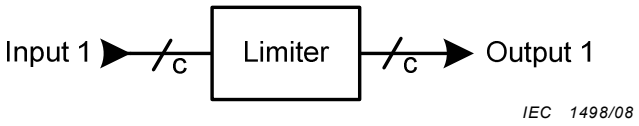


Figure 5 – Audio limiter block

where c is the number of channels on a connection. This represents a block that can limit the incoming audio signal to a preset maximum level.

The group of objects in Table 8 shall be implemented by all compliant audio equipment that has a management model that incorporates one or more audio limiter blocks. The root node for these objects shall be

{ iso(1) standard(0) iec62379 audio(2) audioMIB(1) audioLimiter(5) }

This node shall be used as the block type identifier for audio limiter blocks.

Table 8 – Managed objects for audio limiter blocks

Identifier	Syntax	Index	Readable	Writable	Volatile	Status
aLimiterBlockTable(1)	SEQUENCE OF ALimiterBlockEntry		none	none	no	m
└aLimiterBlockEntry(1)	ALimiterBlockEntry		none	none	no	m
├aLimiterBlockId(1)	BlockId	yes	none	none	no	m
├aLimiterThreshold(2)	AudioLevel		listener	supervisor	no	m
├aLimiterAttackTime(3)	CardinalNumber		listener	supervisor	no	o
├aLimiterGainMakeup(4)	AudioLevel		listener	supervisor	no	o
├aLimiterRecoveryTime(5)	CardinalNumber		listener	supervisor	no	o
└aLimiterRecoveryMode(6)	AudioRecoveryMode		listener	supervisor	no	o

5.4.4.1 aLimiterBlockTable

A table of audio limiter block descriptors for this unit. Each audio limiter block in the unit has a corresponding entry in this table.

5.4.4.2 aLimiterBlockEntry

An entry in the audio limiter block table.

5.4.4.3 aLimiterBlockId

The block identifier for this block. Used as an index when accessing the audio limiter block table.

5.4.4.4 aLimiterThreshold

The limiter threshold for this block. This describes the level at which the limiter starts to attenuate the input signal.

5.4.4.5 aLimiterAttackTime

The attack time (in milliseconds) for this block. This describes how long it takes for the limiter to start attenuating when the input signal exceeds the limiter threshold.

5.4.4.6 aLimiterGainMakeup

The gain makeup for this block. This describes the amount of gain applied to the signal after limiting has been applied.

5.4.4.7 aLimiterRecoveryTime

The recovery time (in milliseconds) for this block. This describes how long it takes for the limiter to stop attenuating when the input signal drops below the limiter threshold.

5.4.4.8 aLimiterRecoveryMode

The recovery mode for this block. This describes the way in which the limiter will recover after being activated. Three descriptions are currently used: `auto`, `slow` and `fast`, but how these are used depends upon the implementation within the equipment.

5.4.5 Audio converter blocks

An audio converter block shall have the following structure:

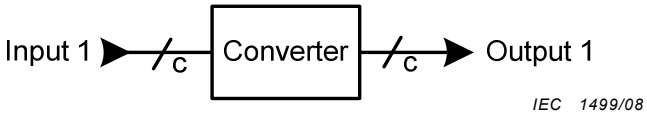


Figure 6 – Audio converter block

where c is the number of channels on a connection. This represents a block that converts an incoming audio signal in one audio format to an outgoing audio signal in a different audio format.

NOTE This block may be used for any kind of conversion including the encoding and decoding of compressed formats.

The block's mode table shall be used to determine what format the converter should output; if only one mode is enabled then the converter block is forced to perform that conversion, if it is able. If more than one mode is enabled, the block should pick the output format according to its own implementation rules. If the block does not support any of the output formats that are enabled, it shall set `aConverterError` to `true`.

The group of objects in Table 9 shall be implemented by all compliant audio equipment that has a management model that incorporates one or more audio converter blocks. The root node for these objects shall be

{ iso(1) standard(0) iec62379 audio(2) audioMIB(1) audioConverter(6) }

This node shall be used as the block type identifier for audio converter blocks.

Table 9 – Managed objects for audio converter blocks

Identifier	Syntax	Index	Readable	Writable	Volatile	Status
aConverterBlockTable(1)	SEQUENCE OF AConverterBlockEntry		none	none	no	m
└aConverterBlockEntry(1)	AConverterBlockEntry		none	none	no	m
├aConverterBlockId(1)	BlockId	yes	none	none	no	m
├aConverterQuality(2)	AudioQuality		listener	supervisor	no	o
├aConverterEnabled(3)	TruthValue		listener	supervisor	no	o
├aConverterDithering(4)	TruthValue		listener	supervisor	no	o
├aConverterOutputFormat(5)	MediaFormat		listener	none	no	m
└aConverterError(6)	TruthValue		listener	none	no	o

5.4.5.1 aConverterBlockTable

A table of audio converter block descriptors for this unit. Each audio converter block in the unit has a corresponding entry in this table.

5.4.5.2 aConverterBlockEntry

An entry in the audio converter block table.

5.4.5.3 aConverterBlockId

The block identifier for this block. Used as an index when accessing the audio converter block table.

5.4.5.4 aConverterQuality

The quality of the conversion performed by this block.

5.4.5.5 aConverterEnabled

If `true`, indicates the incoming audio signal may be converted to one of the enabled audio formats for this block's output in the mode table. If `false`, indicates the incoming audio signal must be output in the same format as it arrives.

5.4.5.6 aConverterDithering

If `true`, indicates dithering is applied to the converted audio signal. If `false`, indicates no dithering is applied to the converted audio signal.

5.4.5.7 aConverterOutputFormat

A description of the current audio signal format being output by the converter block.

5.4.5.8 aConverterError

If `true`, indicates the audio cannot be converted to any of the output formats enabled in the mode table.

5.4.6 Audio level alarm blocks

An audio level alarm block shall have the following structure:

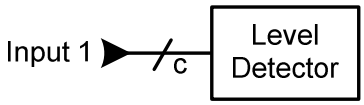


Figure 7 – Audio level alarm block

where c is the number of channels on a connection. This represents a block that detects audio level fault conditions in an audio stream.

NOTE An audio level alarm block may be used to represent, for example, an audio loss detector or an overload indicator.

The group of objects in Table 10 shall be implemented by all compliant audio equipment that has a management model that incorporates one or more audio level alarm blocks. The root node for these objects shall be

{ iso(1) standard(0) iec62379 audio(2) audioMIB(1) audioLevelAlarm(7) }

This node shall be used as the block type identifier for audio level alarm blocks.

Table 10 – Managed objects for audio level alarm blocks

Identifier	Syntax	Index	Readable	Writable	Volatile	Status
aLevelAlarmBlockTable(1)	SEQUENCE OF ALevelAlarmBlockEntry		none	none	no	m
└aLevelAlarmBlockEntry(1)	ALevelAlarmBlockEntry		none	none	no	m
├alaBlockId(1)	BlockId	yes	none	none	no	m
├alaType(2)	AudioLevelAlarmType		listener	supervisor	no	m
├alaThreshold(3)	AudioLevel		listener	supervisor	no	m
├alaWarningTime(4)	CardinalNumber		listener	supervisor	no	o
├alaFailureTime(5)	CardinalNumber		listener	supervisor	no	m
├alaCounter(6)	CardinalNumber		listener	supervisor	no	m
├alaEnabled(7)	TruthValue		listener	supervisor	no	m
└alaStatus(8)	AudioAlarmStatus		listener	none	yes	m

5.4.6.1 aLevelAlarmBlockTable

A table of audio level alarm block descriptors for this unit. Each audio level alarm block in the unit has a corresponding entry in this table.

5.4.6.2 aLevelAlarmBlockEntry

An entry in the audio level alarm block table.

5.4.6.3 alaBlockId

The block identifier for this block. Used as an index when accessing the audio level alarm block table.

5.4.6.4 alaType

The type of audio level alarm. If `lower`, the alarm is triggered by the audio level being less than the threshold; if `higher`, the alarm is triggered by the audio level being greater than the threshold.

5.4.6.5 alaThreshold

The detection threshold for an audio level alarm to be raised by this block.

5.4.6.6 alaWarningTime

The detection time (in seconds) for an audio level warning alarm to be raised by this block; see 5.4.6.10.

5.4.6.7 alaFailureTime

The detection time (in seconds) for an audio level failure alarm to be raised by this block; see 5.4.6.10.

5.4.6.8 alaCounter

The length of time (in seconds) for which the audio level has been in breach of (i.e. above or below, as determined by `alaType`) the detection threshold.

The counter shall be held at zero whenever the audio level is not in breach of the detection threshold.

The counter may be set by the management entity; if at the time of the `SET` request the audio is in breach of the detection threshold, the counter shall continue from the value set.

5.4.6.9 alaEnabled

Controls whether the alarm is primed for operation. If `true` warning and failure alarms will be raised when the audio level alarm counter breaches the warning and failure times respectively. If `false`, the alarm is disabled and no alarms will be raised.

5.4.6.10 alaStatus

Indicates the status of the alarm block. The value shall be `failure` if the audio level is in breach of the detection threshold and `alaCounter` is greater than or equal to `alaFailureTime`, otherwise `warning` if the audio level is in breach of the detection threshold and `alaCounter` is greater than or equal to `alaWarningTime`, otherwise `ok`.

6 Status broadcasts

6.1 General

NOTE Status broadcasts are specified in IEC 62379-1. This standard specifies audio-related status pages and groups.

6.2 Type definitions

In addition to the types defined in IEC 62379-1, the following types are used to specify the syntax of the abstract data structures used in status pages:

6.2.1 Textual conventions

```
AudioConverterStatus ::= OCTET STRING (SIZE(1))
-- a set of bits representing the status of an audio converter
-- bit 0 (lsb) = converter enabled (1=enabled, 0=not enabled)
-- bit 1       = dithering applied (1=dithering, 0=no dithering)
-- bit 2       = conversion ok (1=input converted, 0=converter error)
-- bit 3       = reserved
-- bit 4       = reserved
-- bit 5       = reserved
-- bit 6       = reserved
-- bit 7 (msb) = reserved
```

6.2.2 Sequences

```
AudioFormatsMapEntry ::= SEQUENCE {
  afmNumber  IndexNumber,
  afmFormat  MediaFormat
}
```

6.3 Audio formats mapping

NOTE 1 Due to the varying length of object identifiers, providing an indication of a particular audio format in a status page is problematic. Consequently, a table assigning an arbitrary id to an audio format is defined to allow a shorter, fixed length code to be associated with a format. The mapping chosen is equipment specific and management software should query the table with SNMP GET-NEXT requests on start-up to learn a device's map.

The group of objects in Table 11 shall be implemented by all compliant audio equipment that reports audio format information in status pages. The root node for these objects shall be

{ iso(1) standard(0) iec62379 audio(2) audioStatusMIB(4) }

Table 11 – Managed objects for audio format mappings

Identifier	Syntax	Index	Readable	Writable	Volatile	Status
audioFormatsMapTable(1)	SEQUENCE OF AudioFormatsMapEntry		none	none	no	m
└audioFormatsMapEntry(1)	AudioFormatsMapEntry		none	none	no	m
├afmNumber(1)	IndexNumber	yes	none	none	no	m
└afmFormat(2)	MediaFormat		listener	supervisor	yes	m

6.3.1 audioFormatsMapTable

A table of audio format to index number mappings for this unit. Each mapping has a corresponding entry in this table.

6.3.2 audioFormatsMapEntry

An entry in audio formats map table.

6.3.3 afmNumber

The audio format number for this mapping.

6.3.4 afmFormat

The format of the audio data associated with this format number.

6.4 Page formats

6.4.1 Audio port page

This page shall be produced by all audio ports and shall contain the entries in Table 12:

Table 12 – Status entries for audio port page

Octet(s)	Description	Value	Note(s)
1..2	Page number		
3..4	Block identifier	BlockId	1
5..8	Audio format	IndexNumber	2
9..10	Channel 1 peak level	AudioLevel	3
…	repeat previous entry for each remaining channel	…	4

NOTE 1 Coded with the block identifier for the block.

NOTE 2 Coded with the mapping from audioFormatsMapTable for the audio format of the port as given by aPortFormat.

NOTE 3 Coded with the audio peak level detected on the specified channel of the specified port since the last time this page was broadcast, in units of 0,01dB. A value of 0 represents full scale and values of 1 or greater indicate overload.

NOTE 4 The number of audio channels shall be inferred from the page length.

6.4.2 AES3 ancillary data page

This page shall be produced by all audio ports allowing access to AES3 ancillary data and shall contain the entries in Table 13:

Table 13 – Status entries for AES3 ancillary data page

Octet(s)	Description	Value	Note(s)
1..2	Page number		
3..4	Block identifier	`BlockId`	1
5..28	Channel 1 channel data	`Aes3ChannelData`	2
29..52	Channel 1 user data	`Aes3UserData`	3
53	Channel 1 validity error	`TruthValue`	4
…	repeat previous 3 entries for each remaining channel	…	5
NOTE 1 Coded with the block identifier for the block.			
NOTE 2 Coded with the current value of `aes3ChannelData` in the table entry associated with this channel.			
NOTE 3 Coded with the current value of `aes3UserData` in the table entry associated with this channel.			
NOTE 4 Coded with the current value of `aes3ValidityError` in the table entry associated with this channel.			
NOTE 5 The number of audio channels shall be inferred from the page length.			

6.4.3 Audio mixer page

This page shall be produced by all audio mixer blocks and shall contain the entries in Table 14:

Table 14 – Status entries for audio mixer page

Octet(s)	Description	Value	Note(s)
1..2	Page number		
3..4	Block identifier	`BlockId`	1
5..8	Input number	`IndexNumber`	2
9..12	Delay	`CardinalNumber`	3
13..14	Level	`AudioLevel`	4
…	repeat previous three entries for each input	…	5
NOTE 1 Coded with the block identifier for the block.			
NOTE 2 Coded with the number of the current input as given by `aMixerInputNumber` in the table entry associated with this mixer input.			
NOTE 3 Coded with the current value of `aMixerInputDelay` in the table entry associated with this mixer input.			
NOTE 4 Coded with the current value of `aMixerInputLevel` in the table entry associated with this mixer input.			
NOTE 5 The number of inputs to the mixer shall be inferred from the page length.			

6.4.4 Audio crosspoint page

This page shall be produced by all audio crosspoint blocks and shall contain the entries in Table 15:

Table 15 – Status entries for audio crosspoint page

Octet(s)	Description	Value	Note(s)
1..2	Page number		
3..4	Block identifier	BlockId	1
5..6	Crosspoint path source	AudioChannel	2
7..8	Crosspoint path destination	AudioChannel	3
9..10	Crosspoint path phase	AudioPhase	4
11..12	Crosspoint path gain	AudioLevel	5
…	repeat previous four entries for each path	…	6

NOTE 1 Coded with the block identifier for the block.

NOTE 2 Coded with the source of the current path as given by aCrosspointPathSrc in the table entry associated with this crosspoint path.

NOTE 3 Coded with the destination of the current path as given by aCrosspointPathDst in the table entry associated with this crosspoint path.

NOTE 4 Coded with the current value of aCrosspointPathPhase in the table entry associated with this crosspoint path.

NOTE 5 Coded with the current value of aCrosspointPathGain in the table entry associated with this crosspoint path.

NOTE 6 The number of paths through the crosspoint shall be inferred from the page length.

6.4.5 Audio clip player page

This page shall be produced by all audio clip player blocks and shall contain the entries in Table 16:

Table 16 – Status entries for audio clip player page

Octet(s)	Description	Value	Note(s)
1..2	Page number		
3..4	Block identifier	BlockId	1
5	Clip player activation	AudioClipActivation	2
6..9	Clip player selection	AudioClipSelector	3
10..13	Clip player track number	CardinalNumber	4
14..15	Main fader level	AudioLevel	5
16..17	Clip player fader level	AudioLevel	6

NOTE 1 Coded with the block identifier for the block.

NOTE 2 Coded with the current value of acpActivation in the table entry associated with this clip player.

NOTE 3 Coded with the current value of acpSelection in the table entry associated with this clip player.

NOTE 4 Coded with the current value of acpTrackNumber in the table entry associated with this clip player.

NOTE 5 Coded with the current value of acpMainFaderLevelNow in the table entry associated with this clip player.

NOTE 6 Coded with the current value of acpClipFaderLevelNow in the table entry associated with this clip player.

6.4.6 Audio limiter page

This page shall be produced by all audio limiter blocks and shall contain the entries in Table 17:

Table 17 – Status entries for audio limiter page

Octet(s)	Description	Value	Note(s)
1..2	Page number		
3..4	Block identifier	`BlockId`	1
5..6	Limiter threshold	`AudioLevel`	2
7..10	Limiter attack time	`CardinalNumber`	3
11..12	Limiter gain makeup	`AudioLevel`	4
13..16	Limiter recovery time	`CardinalNumber`	5
17	Limiter recovery mode	`AudioRecoveryMode`	6

NOTE 1 Coded with the block identifier for the block.

NOTE 2 Coded with the current value of `aLimiterThreshold` in the table entry associated with this limiter.

NOTE 3 Coded with the current value of `aLimiterAttackTime` in the table entry associated with this limiter.

NOTE 4 Coded with the current value of `aLimiterGainMakeup` in the table entry associated with this limiter.

NOTE 5 Coded with the current value of `aLimiterRecoveryTime` in the table entry associated with this limiter.

NOTE 6 Coded with the current value of `aLimiterRecoveryMode` in the table entry associated with this limiter.

6.4.7 Audio converter page

This page shall be produced by all audio converter blocks and shall contain the entries in Table 18:

Table 18 – Status entries for audio converter page

Octet(s)	Description	Value	Note(s)
1..2	Page number		
3..4	Block identifier	`BlockId`	1
5	Converter status	`AudioConverterStatus`	2
6..9	Output format	`IndexNumber`	3

NOTE 1 Coded with the block identifier for the block.

NOTE 2 Coded with the current value of the converter status.

NOTE 3 Coded with the mapping from `audioFormatsMapTable` for the output audio format of the converter as given by `aConverterOutputFormat`.

6.4.8 Audio level alarm page

This page shall be produced by all audio level alarm blocks and shall contain the entries in Table 19:

Table 19 – Status entries for audio level alarm page

Octet(s)	Description	Value	Note(s)
1..2	Page number		
3..4	Block identifier	`BlockId`	1
5	Alarm enabled	`TruthValue`	2
6	Alarm status	`AudioAlarmStatus`	3
7..10	Counter	`CardinalNumber`	4
11..12	Audio threshold	`AudioLevel`	5
13..16	Warning time	`CardinalNumber`	6
17..20	Failure time	`CardinalNumber`	7

NOTE 1 Coded with the block identifier for the block.

NOTE 2 Coded with the current value of `alaEnabled` in the table entry associated with the audio level alarm.

NOTE 3 Coded with the current value of `alaStatus` in the table entry associated with the audio level alarm.

NOTE 4 Coded with the current value of `alaCounter` in the table entry associated with the audio level alarm.

NOTE 5 Coded with the current value of `alaThreshold` in the table entry associated with the audio level alarm.

NOTE 6 Coded with the current value of `alaWarningTime` in the table entry associated with the audio level alarm.

NOTE 7 Coded with the current value of `alaFailureTime` in the table entry associated with the audio level alarm.

6.5 Page groups

6.5.1 `audioPorts`

This group shall be supported by all audio units with ports. The root node for this group shall be

{ iso(1) standard(0) iec62379 audio(2) audioStatusGroup(3) audioPorts(1) }

A broadcast of this group shall consist of the following:

- page 1: Audio port, broadcast at the base page rate;
- page 2: AES3 ancillary data, broadcast at half the base page rate.

The default base page rate for this group shall be 60 pages per minute (1 page every second).

6.5.2 `standardAudioBlocks`

This group shall be supported by all audio units with some blocks defined in this standard. The root node for this group shall be

{ iso(1) standard(0) iec62379 audio(2) audioStatusGroup(3) standardAudioBlocks(2) }

A broadcast of this group shall consist of the following:

- page 1: Audio mixer, broadcast at the base page rate;
- page 2: Audio crosspoint, broadcast at the base page rate;
- page 3: Audio clip player, broadcast at the base page rate;
- page 4: Audio limiter, broadcast at the base page rate;
- page 5: Audio converter, broadcast at the base page rate.

The default base page rate for this group shall be 60 pages per minute (1 page every second).

6.5.3 audioAlarms

This group shall be supported by all audio units with alarms. The root node for this group shall be

{ iso(1) standard(0) iec62379 audio(2) audioStatusGroup(3) audioAlarms (3) }

A broadcast of this group shall consist of the following:

- page 1: Audio Level Alarm, broadcast at the base page rate.

The default base page rate for this group shall be 60 pages per minute (1 page every second).

Annex A
(informative)

Machine-readable audio format definitions

This annex provides a machine-readable version of the audio data format definitions which is intended to be interpretable by standard MIB browsing software tools. If there is any inconsistency between this annex and Clause 4, Clause 4 takes precedence.

The format used to describe the data format identifiers conforms to IETF STD 58 (SMIv2).

NOTE 1 This annex is not intended to cover every format permitted by the definitions in Clause 4.

```
IEC62379-2-FORMATS DEFINITIONS ::= BEGIN

IMPORTS

  MODULE-IDENTITY, OBJECT-IDENTITY
    FROM SNMPv2-SMI

  iec62379
    FROM IEC62379-1-MIB;

audioFormat MODULE-IDENTITY
  LAST-UPDATED  "200804301500Z"
  ORGANIZATION  "IEC PT62379"
  CONTACT-INFO  "Not specified"
  DESCRIPTION   "The audio format identifiers defined in clause 4 of
                 IEC 62379-2."
  REVISION      "200804301500Z"
  DESCRIPTION   "Edition 1.0 final draft."
  ::= { audio 2 }

audio OBJECT IDENTIFIER ::= { iec62379 2 }


-- 4.1 Audio signal format definitions

-- 4.1.1 Audio parameters

ChannelArrangement ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "The audio channel arrangement in the audio format."
  SYNTAX        INTEGER (0..5)
                -- {
                --   unspecified         (0),
                --   discreteMono        (1),
                --   stereo              (2),
                --   jointStereo         (3),
                --   surround            (4),
                --   surroundWithDownmix (5)
                -- } (unspecified..surroundWithDownmix)

NumberChannels ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "An integer representing the number of audio channels."
  SYNTAX        INTEGER (0..2147483647)

BitDepth ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "An integer representing the audio bit depth in bits per
                 sample."
  SYNTAX        INTEGER (0..2147483647)
```

```
SamplingFrequency ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "An integer representing the audio sample frequency in Hz."
  SYNTAX        INTEGER (0..2147483647)

BitRate ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "An integer representing the audio bit rate in bits per second."
  SYNTAX        INTEGER (0..2147483647)


-- 4.1.2 Audio signal formats

audioSignalFormat OBJECT IDENTIFIER ::= { audioFormat 1 }


-- 4.1.2.1 Unspecified audio

unspecifiedAudio OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "AudioFormat wildcard value - any supported format allowed."
  ::= { audioSignalFormat 0 }


-- 4.1.2.2 No audio

noAudio OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Indicates the output is muted."
  ::= { audioSignalFormat 1 }


-- 4.1.2.3 Analogue audio

analogueAudio OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Analogue audio."
  ::= { audioSignalFormat 2 }


-- analogue discrete mono

analogueMono OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Discrete mono analogue audio."
  ::= { analogueAudio 1 }

analogueMono1Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Discrete mono analogue audio with 1 channel."
  ::= { analogueMono 1 }

analogueMono2Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Discrete mono analogue audio with 2 channels."
  ::= { analogueMono 2}

analogueMono3Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Discrete mono analogue audio with 3 channels."
  ::= { analogueMono 3}

analogueMono4Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Discrete mono analogue audio with 4 channels."
  ::= { analogueMono 4}


-- analogue stereo
```

```
analogueStereo OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Stereo analogue audio."
  ::= { analogueAudio 2 }

analogueStereo2Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Stereo analogue audio with 2 channels (1 L-R pair)."
  ::= { analogueStereo 2 }

analogueStereo4Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Stereo analogue audio with 4 channels (2 L-R pairs)."
  ::= { analogueStereo 4}


-- analogue jointstereo

analogueJointStereo OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Joint stereo analogue audio."
    ::= { analogueAudio 3 }

analogueJointStereo2Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Joint stereo analogue audio with 2 channels (1 M-S pair)."
  ::= { analogueJointStereo 2 }

analogueJointStereo4Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Joint stereo analogue audio with 4 channels (2 M-S pairs)."
  ::= { analogueJointStereo 4}


-- 4.1.2.4 PCM audio

pcmAudio OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Linear PCM audio."
  ::= { audioSignalFormat 3 }


-- PCM discrete mono

pcmMono OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Discrete mono linear PCM audio."
  ::= { pcmAudio 1 }

pcmMono1Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Discrete mono linear PCM audio with 1 channel."
  ::= { pcmMono 1 }

pcmMono1Chan16 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "16-bit discrete mono linear PCM audio with 1 channel."
  ::= { pcmMono1Chan 16 }

pcmMono1Chan16at44100 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k 16-bit discrete mono linear PCM audio with 1 channel."
  ::= { pcmMono1Chan16 44100 }

pcmMono1Chan16at48000 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k 16-bit discrete mono linear PCM audio with 1 channel."
  ::= { pcmMono1Chan16 48000 }

pcmMono1Chan16at96000 OBJECT-IDENTITY
```

```
  STATUS        current
  DESCRIPTION   "96k 16-bit discrete mono linear PCM audio with 1 channel."
  ::= { pcmMono1Chan16 96000 }

pcmMono1Chan20 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "20-bit discrete mono linear PCM audio with 1 channel."
  ::= { pcmMono1Chan 20 }

pcmMono1Chan20at44100 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k 20-bit discrete mono linear PCM audio with 1 channel."
  ::= { pcmMono1Chan20 44100 }

pcmMono1Chan20at48000 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k 20-bit discrete mono linear PCM audio with 1 channel."
  ::= { pcmMono1Chan20 48000 }

pcmMono1Chan20at96000 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "96k 20-bit discrete mono linear PCM audio with 1 channel."
  ::= { pcmMono1Chan20 96000 }

pcmMono1Chan24 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "24-bit discrete mono linear PCM audio with 1 channel."
  ::= { pcmMono1Chan 24 }

pcmMono1Chan24at44100 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k 24-bit discrete mono linear PCM audio with 1 channel."
  ::= { pcmMono1Chan24 44100 }

pcmMono1Chan24at48000 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k 24-bit discrete mono linear PCM audio with 1 channel."
  ::= { pcmMono1Chan24 48000 }

pcmMono1Chan24at96000 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "96k 24-bit discrete mono linear PCM audio with 1 channel."
  ::= { pcmMono1Chan24 96000 }

pcmMono2Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Discrete mono linear PCM audio with 2 channels."
  ::= { pcmMono 2 }

pcmMono2Chan16 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "16-bit discrete mono linear PCM audio with 2 channels."
  ::= { pcmMono2Chan 16 }

pcmMono2Chan16at44100 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k 16-bit discrete mono linear PCM audio with 2 channels."
  ::= { pcmMono2Chan16 44100 }

pcmMono2Chan16at48000 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k 16-bit discrete mono linear PCM audio with 2 channels."
  ::= { pcmMono2Chan16 48000 }

pcmMono2Chan16at96000 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "96k 16-bit discrete mono linear PCM audio with 2 channels."
  ::= { pcmMono2Chan16 96000 }

pcmMono2Chan20 OBJECT-IDENTITY
```

```
    STATUS        current
    DESCRIPTION   "20-bit discrete mono linear PCM audio with 2 channels."
    ::= { pcmMono2Chan 20 }

  pcmMono2Chan20at44100 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "44.1k 20-bit discrete mono linear PCM audio with 2 channels."
    ::= { pcmMono2Chan20 44100 }

  pcmMono2Chan20at48000 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "48k 20-bit discrete mono linear PCM audio with 2 channels."
    ::= { pcmMono2Chan20 48000 }

  pcmMono2Chan20at96000 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "96k 20-bit discrete mono linear PCM audio with 2 channels."
    ::= { pcmMono2Chan20 96000 }

  pcmMono2Chan24 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "24-bit discrete mono linear PCM audio with 2 channels."
    ::= { pcmMono2Chan 24 }

  pcmMono2Chan24at44100 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "44.1k 24-bit discrete mono linear PCM audio with 2 channels."
    ::= { pcmMono2Chan24 44100 }

  pcmMono2Chan24at48000 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "48k 24-bit discrete mono linear PCM audio with 2 channels."
    ::= { pcmMono2Chan24 48000 }

  pcmMono2Chan24at96000 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "96k 24-bit discrete mono linear PCM audio with 2 channels."
    ::= { pcmMono2Chan24 96000 }


  -- PCM stereo

  pcmStereo OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "Stereo linear PCM audio."
    ::= { pcmAudio 2 }

  pcmStereo2Chan OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "Stereo linear PCM audio with 2 channels (1 L-R pair)."
    ::= { pcmStereo 2 }

  pcmStereo2Chan16 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "16-bit stereo linear PCM audio with 2 channels."
    ::= { pcmStereo2Chan 16 }

  pcmStereo2Chan16at44100 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "44.1k 16-bit stereo linear PCM audio with 2 channels."
    ::= { pcmStereo2Chan16 44100 }

  pcmStereo2Chan16at48000 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "48k 16-bit stereo linear PCM audio with 2 channels."
    ::= { pcmStereo2Chan16 48000 }

  pcmStereo2Chan16at96000 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "96k 16-bit stereo linear PCM audio with 2 channels."
```

```
  ::= { pcmStereo2Chan16 96000 }

pcmStereo2Chan20 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "20-bit stereo linear PCM audio with 2 channels."
  ::= { pcmStereo2Chan 20 }

pcmStereo2Chan20at44100 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k 20-bit stereo linear PCM audio with 2 channels."
  ::= { pcmStereo2Chan20 44100 }

pcmStereo2Chan20at48000 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k 20-bit stereo linear PCM audio with 2 channels."
  ::= { pcmStereo2Chan20 48000 }

pcmStereo2Chan20at96000 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "96k 20-bit stereo linear PCM audio with 2 channels."
  ::= { pcmStereo2Chan20 96000 }

pcmStereo2Chan24 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "24-bit stereo linear PCM audio with 2 channels."
  ::= { pcmStereo2Chan 24 }

pcmStereo2Chan24at44100 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k 24-bit stereo linear PCM audio with 2 channels."
  ::= { pcmStereo2Chan24 44100 }

pcmStereo2Chan24at48000 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k 24-bit stereo linear PCM audio with 2 channels."
  ::= { pcmStereo2Chan24 48000 }

pcmStereo2Chan24at96000 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "96k 24-bit stereo linear PCM audio with 2 channels."
  ::= { pcmStereo2Chan24 96000 }


-- PCM joint stereo

pcmJointStereo OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Joint stereo linear PCM audio."
  ::= { pcmAudio 3 }

pcmJointStereo2Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Joint stereo linear PCM audio with 2 channels (1 M-S pair)."
  ::= { pcmJointStereo 2 }

pcmJointStereo2Chan16 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "16-bit joint stereo linear PCM audio with 2 channels."
  ::= { pcmJointStereo2Chan 16 }

pcmJointStereo2Chan20 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "20-bit joint stereo linear PCM audio with 2 channels."
  ::= { pcmJointStereo2Chan 20 }

pcmJointStereo2Chan24 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "24-bit joint stereo linear PCM audio with 2 channels."
  ::= { pcmJointStereo2Chan 24 }
```

```
-- 4.1.2.5 MPEG-1 layer 2 audio

mp2Audio OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION    "MPEG1 layer 2 audio."
  ::= { audioSignalFormat 4 }


-- MP2 discrete mono

mp2Mono OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Discrete mono MP2 audio."
  ::= { mp2Audio 1 }

mp2Mono1Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Discrete mono MP2 audio with 1 channel."
  ::= { mp2Mono 1 }


-- MP2 discrete mono with 1 channel at 44100

mp2Mono1Chan44100 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio with 1 channel."
  ::= { mp2Mono1Chan 44100 }

mp2Mono1Chan44100at32 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio at 32kb/s with 1 channel."
  ::= { mp2Mono1Chan44100 32000 }

mp2Mono1Chan44100at40 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio at 40kb/s with 1 channel."
  ::= { mp2Mono1Chan44100 40000 }

mp2Mono1Chan44100at48 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio at 48kb/s with 1 channel."
  ::= { mp2Mono1Chan44100 48000 }

mp2Mono1Chan44100at56 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio at 56kb/s with 1 channel."
  ::= { mp2Mono1Chan44100 56000 }

mp2Mono1Chan44100at64 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio at 64kb/s with 1 channel."
  ::= { mp2Mono1Chan44100 64000 }

mp2Mono1Chan44100at80 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio at 80kb/s with 1 channel."
  ::= { mp2Mono1Chan44100 80000 }

mp2Mono1Chan44100at96 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio at 96kb/s with 1 channel."
  ::= { mp2Mono1Chan44100 96000 }

mp2Mono1Chan44100at112 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio at 112kb/s with 1 channel."
  ::= { mp2Mono1Chan44100 112000 }

mp2Mono1Chan44100at128 OBJECT-IDENTITY
```

```
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio at 128kb/s with 1 channel."
  ::= { mp2Mono1Chan44100 128000 }

mp2Mono1Chan44100at160 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio at 160kb/s with 1 channel."
  ::= { mp2Mono1Chan44100 160000 }

mp2Mono1Chan44100at192 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio at 192kb/s with 1 channel."
  ::= { mp2Mono1Chan44100 192000 }

mp2Mono1Chan44100at224 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio at 224kb/s with 1 channel."
  ::= { mp2Mono1Chan44100 224000 }

mp2Mono1Chan44100at256 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio at 256kb/s with 1 channel."
  ::= { mp2Mono1Chan44100 256000 }

mp2Mono1Chan44100at320 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k discrete mono MP2 audio at 320kb/s with 1 channel."
  ::= { mp2Mono1Chan44100 320000 }


-- MP2 discrete mono with 1 channel at 48000

mp2Mono1Chan48000 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k mono MP2 audio with 1 channel."
  ::= { mp2Mono1Chan 48000 }

mp2Mono1Chan48000at32 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k discrete mono MP2 audio at 32kb/s with 1 channel."
  ::= { mp2Mono1Chan48000 32000 }

mp2Mono1Chan48000at40 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "8k discrete mono MP2 audio at 40kb/s with 1 channel."
  ::= { mp2Mono1Chan48000 40000 }

mp2Mono1Chan48000at48 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k discrete mono MP2 audio at 48kb/s with 1 channel."
  ::= { mp2Mono1Chan48000 48000 }

mp2Mono1Chan48000at56 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k discrete mono MP2 audio at 56kb/s with 1 channel."
  ::= { mp2Mono1Chan48000 56000 }

mp2Mono1Chan48000at64 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k discrete mono MP2 audio at 64kb/s with 1 channel."
  ::= { mp2Mono1Chan48000 64000 }

mp2Mono1Chan48000at80 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k discrete mono MP2 audio at 80kb/s with 1 channel."
  ::= { mp2Mono1Chan48000 80000 }

mp2Mono1Chan48000at96 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k discrete mono MP2 audio at 96kb/s with 1 channel."
```

```
    ::= { mp2Mono1Chan48000 96000 }

  mp2Mono1Chan48000at112 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "48k discrete mono MP2 audio at 112kb/s with 1 channel."
    ::= { mp2Mono1Chan48000 112000 }

  mp2Mono1Chan48000at128 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "48k discrete mono MP2 audio at 128kb/s with 1 channel."
    ::= { mp2Mono1Chan48000 128000 }

  mp2Mono1Chan48000at160 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "48k discrete mono MP2 audio at 160kb/s with 1 channel."
    ::= { mp2Mono1Chan48000 160000 }

  mp2Mono1Chan48000at192 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "48k discrete mono MP2 audio at 192kb/s with 1 channel."
    ::= { mp2Mono1Chan48000 192000 }

  mp2Mono1Chan48000at224 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "48k discrete mono MP2 audio at 224kb/s with 1 channel."
    ::= { mp2Mono1Chan48000 224000 }

  mp2Mono1Chan48000at256 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "48k discrete mono MP2 audio at 256kb/s with 1 channel."
    ::= { mp2Mono1Chan48000 256000 }

  mp2Mono1Chan48000at320 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "8k discrete mono MP2 audio at 320kb/s with 1 channel."
    ::= { mp2Mono1Chan48000 320000 }


  -- MP2 stereo

  mp2Stereo OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "Stereo MP2 audio."
    ::= { mp2Audio 2 }

  mp2Stereo2Chan OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "Stereo MP2 audio with 2 channels (1 L-R pair)."
    ::= { mp2Stereo 2 }


  -- MP2 stereo with 2 channels at 44100

  mp2Stereo2Chan44100 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "44.1k stereo MP2 audio."
    ::= { mp2Stereo2Chan 44100 }

  mp2Stereo2Chan44100at32 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "44.1k stereo MP2 audio at 32kb/s."
    ::= { mp2Stereo2Chan44100 32000 }

  mp2Stereo2Chan44100at40 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "44.1k stereo MP2 audio at 40kb/s."
    ::= { mp2Stereo2Chan44100 40000 }

  mp2Stereo2Chan44100at48 OBJECT-IDENTITY
    STATUS        current
```

```
  DESCRIPTION   "44.1k stereo MP2 audio at 48kb/s."
  ::= { mp2Stereo2Chan44100 48000 }

mp2Stereo2Chan44100at56 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k stereo MP2 audio at 56kb/s."
  ::= { mp2Stereo2Chan44100 56000 }

mp2Stereo2Chan44100at64 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k stereo MP2 audio at 64kb/s."
  ::= { mp2Stereo2Chan44100 64000 }

mp2Stereo2Chan44100at80 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k stereo MP2 audio at 80kb/s."
  ::= { mp2Stereo2Chan44100 80000 }

mp2Stereo2Chan44100at96 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k stereo MP2 audio at 96kb/s."
  ::= { mp2Stereo2Chan44100 96000 }

mp2Stereo2Chan44100at112 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k stereo MP2 audio at 112kb/s."
  ::= { mp2Stereo2Chan44100 112000 }

mp2Stereo2Chan44100at128 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k stereo MP2 audio at 128kb/s."
  ::= { mp2Stereo2Chan44100 128000 }

mp2Stereo2Chan44100at160 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k stereo MP2 audio at 160kb/s."
  ::= { mp2Stereo2Chan44100 160000 }

mp2Stereo2Chan44100at192 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k stereo MP2 audio at 192kb/s."
  ::= { mp2Stereo2Chan44100 192000 }

mp2Stereo2Chan44100at224 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k stereo MP2 audio at 224kb/s."
  ::= { mp2Stereo2Chan44100 224000 }

mp2Stereo2Chan44100at256 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k stereo MP2 audio at 256kb/s."
  ::= { mp2Stereo2Chan44100 256000 }

mp2Stereo2Chan44100at320 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k stereo MP2 audio at 320kb/s."
  ::= { mp2Stereo2Chan44100 320000 }


-- MP2 stereo with 2 channels at 48000

mp2Stereo2Chan48000 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio."
  ::= { mp2Stereo2Chan 48000 }

mp2Stereo2Chan48000at32 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio at 32kb/s."
  ::= { mp2Stereo2Chan48000 32000 }
```

```
mp2Stereo2Chan48000at40 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio at 40kb/s."
  ::= { mp2Stereo2Chan48000 40000 }

mp2Stereo2Chan48000at48 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio at 48kb/s."
  ::= { mp2Stereo2Chan48000 48000 }

mp2Stereo2Chan48000at56 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio at 56kb/s."
  ::= { mp2Stereo2Chan48000 56000 }

mp2Stereo2Chan48000at64 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio at 64kb/s."
  ::= { mp2Stereo2Chan48000 64000 }

mp2Stereo2Chan48000at80 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio at 80kb/s."
  ::= { mp2Stereo2Chan48000 80000 }

mp2Stereo2Chan48000at96 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio at 96kb/s."
  ::= { mp2Stereo2Chan48000 96000 }

mp2Stereo2Chan48000at112 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio at 112kb/s."
  ::= { mp2Stereo2Chan48000 112000 }

mp2Stereo2Chan48000at128 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio at 128kb/s."
  ::= { mp2Stereo2Chan48000 128000 }

mp2Stereo2Chan48000at160 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio at 160kb/s."
  ::= { mp2Stereo2Chan48000 160000 }

mp2Stereo2Chan48000at192 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio at 192kb/s."
  ::= { mp2Stereo2Chan48000 192000 }

mp2Stereo2Chan48000at224 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio at 224kb/s."
  ::= { mp2Stereo2Chan48000 224000 }

mp2Stereo2Chan48000at256 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio at 256kb/s."
  ::= { mp2Stereo2Chan48000 256000 }

mp2Stereo2Chan48000at320 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k stereo MP2 audio at 320kb/s."
  ::= { mp2Stereo2Chan48000 320000 }


-- MP2 joint stereo

mp2JointStereo OBJECT-IDENTITY
```

```
  STATUS        current
  DESCRIPTION   "Joint stereo MP2 audio."
  ::= { mp2Audio 3 }

mp2JointStereo2Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Joint stereo MP2 audio with 2 channels (1 M-S pair)."
  ::= { mp2JointStereo 2 }

mp2JointStereo2Chan44100 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "44.1k joint stereo MP2 audio."
  ::= { mp2JointStereo2Chan 44100 }

mp2JointStereo2Chan48000 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "48k joint stereo MP2 audio."
  ::= { mp2JointStereo2Chan 48000 }


-- 4.1.2.6 MPEG-1 layer 3 audio

mp3Audio OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "MPEG1 layer 3 audio."
  ::= { audioSignalFormat 5 }

mp3Mono OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Discrete mono MP3 audio."
  ::= { mp3Audio 1 }

mp3Mono1Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Discrete mono MP3 audio with 1 channel."
  ::= { mp3Mono 1 }

mp3Stereo OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Stereo MP3 audio."
  ::= { mp3Audio 2 }

mp3Stereo2Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Stereo MP3 audio with 2 channels (1 L-R pair)."
  ::= { mp3Stereo 2 }

mp3JointStereo OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Joint stereo MP3 audio."
  ::= { mp3Audio 3 }

mp3JointStereo2Chan OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Joint stereo MP3 audio with 2 channels (1 M-S pair)."
  ::= { mp3JointStereo 2 }


-- 4.1.2.7 AAC audio

aacAudio OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "aac audio."
  ::= { audioSignalFormat 6 }

aacLC OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "aac audio with the low complexity profile."
  ::= { aacAudio 1 }
```

```
aacMain OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "aac audio with the main profile."
  ::= { aacAudio 2 }

aacSRS OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "aac audio with the sample-rate-scalable profile."
  ::= { aacAudio 3 }

aacLTP OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "aac audio with the long term prediction profile."
  ::= { aacAudio 4 }

aacLD OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "aac audio with the low delay profile."
  ::= { aacAudio 5 }


-- 4.1.2.8 G711 audio

g711Audio OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "G711 audio."
  ::= { audioSignalFormat 7 }

g711ALaw OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "G711 audio denoting A-Law encoding."
  ::= { g711Audio 1 }

g711MuLaw OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "G711 audio denoting mu-Law encoding."
  ::= { g711Audio 2 }


-- 4.1.2.9 G722 audio

g722Audio OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "G722 audio."
  ::= { audioSignalFormat 8 }

g722at48 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "G722 audio at 48kbits/s."
  ::= { g722Audio 48 }

g722at56 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "G722 audio at 56kbits/s."
  ::= { g722Audio 56 }

g722at64 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "G722 audio at 64kbits/s."
  ::= { g722Audio 64 }


-- 4.1.2.10 apt-X audio

aptXAudio OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "APT-X audio."
  ::= { audioSignalFormat 9 }
```

```
-- 4.1.2.11 enhanced apt-X audio

enhAptXAudio OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Enhanced APT-X audio."
  ::= { audioSignalFormat 10 }


-- 4.1.2.12 J41 audio

j41Audio OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "J41 audio."
  ::= { audioSignalFormat 11 }

j41ALawA OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "J41 audio using A-law companding, variant A."
  ::= { j41Audio 1 }

j41ALawB OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "J41 audio using A-law companding, variant B."
  ::= { j41Audio 2 }

j41Nic OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "J41 audio using near instantaneous companding."
  ::= { j41Audio 3 }


-- 4.1.2.13 J57 audio

j57Audio OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "J57 audio."
  ::= { audioSignalFormat 12 }

j57H11 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "J57 using an H11 channel."
  ::= { j57Audio 1 }

j57H12 OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "J57 using an H12 channel."
  ::= { j57Audio 2 }


-- 4.1.2.14 Invalid audio

invalidAudio OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Indicates an error, such as an inability to decode a signal
                 earlier in the chain."
  ::= { audioSignalFormat 13 }


-- 4.2 Audio transport format definitions

audioTransportFormat OBJECT IDENTIFIER ::= { audioFormat transport(2) }

unspecifiedTransport OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Unspecified transport."
  ::= { audioTransportFormat 0 }

analogue OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "Analogue transport."
```

```
    ::= { audioTransportFormat 1 }

  aes3 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "AES3 transport."
    ::= { audioTransportFormat 2 }

  aes10 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "AES10 transport."
    ::= { audioTransportFormat 3 }

  aes50 OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "AES50 transport."
    ::= { audioTransportFormat 4 }


  -- Metadata

  audioMetadataFormat OBJECT IDENTIFIER ::= { audioFormat 3 }

  unspecifiedMetadata OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION   "Unspecified metadata."
    ::= { audioMetadataFormat 0 }


  END
```

Annex B
(informative)

Machine-readable audio block definitions

This annex provides a machine-readable version of the audio block definitions which is intended to be interpretable by standard MIB browsing software tools. It does not express all the requirements of the standard, for instance where access to an object is restricted at certain privilege levels. If there is any inconsistency between this annex and Clause 5, Clause 5 takes precedence.

The format used to describe the MIB objects conforms to IETF STD 58 (SMIv2).

```
IEC62379-2-MIB DEFINITIONS ::= BEGIN

IMPORTS

  MODULE-IDENTITY, OBJECT-TYPE
    FROM SNMPv2-SMI

  OBJECT-GROUP, MODULE-COMPLIANCE
    FROM SNMPv2-CONF

  TEXTUAL-CONVENTION, TruthValue
    FROM SNMPv2-TC

  IntegerNumber, CardinalNumber, IndexNumber, Utf8String, BlockId,
  MediaFormat, PortDirection, iec62379, OperationId, ObjectName,
  ObjectValue, PrivilegeLevel, OperationState
    FROM IEC62379-1-MIB;

audioMIB MODULE-IDENTITY
  LAST-UPDATED  "200804301500Z"
  ORGANIZATION  "IEC PT62379"
  CONTACT-INFO  "Not specified"
  DESCRIPTION   "The MIB module for managing audio functions in IEC 62379
                 compliant equipment."
  REVISION      "200804301500Z"
  DESCRIPTION   "Edition 1.0 final draft."
  ::= { audio 1 }

audio OBJECT IDENTIFIER ::= { iec62379 2 }

-- Object identifier values for module compliance statements

audioMIBCompliance OBJECT IDENTIFIER ::= { audioMIB 0 }


-- Object identifier values for MIB object groups
-- the following definitions are also used as BlockType values

audioPort         OBJECT IDENTIFIER ::= { audioMIB 1 }

audioMixer        OBJECT IDENTIFIER ::= { audioMIB 2 }

audioCrosspoint   OBJECT IDENTIFIER ::= { audioMIB 3 }

audioClipPlayer   OBJECT IDENTIFIER ::= { audioMIB 4 }

audioLimiter      OBJECT IDENTIFIER ::= { audioMIB 5 }

audioConverter    OBJECT IDENTIFIER ::= { audioMIB 6 }

audioLevelAlarm   OBJECT IDENTIFIER ::= { audioMIB 7 }
```

```
-- 5.2 Type definitions

-- 5.2.1 Textual conventions

AudioTransportType ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "A reference to the transport used for an audio connection."
  SYNTAX        OBJECT IDENTIFIER

AudioLevel ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "An absolute or relative audio level in units of 0.01dB."
  SYNTAX        INTEGER (-20000..20000)
                -- {
                --   mInfinity (-20000),
                --   fullScale (0),
                --   pInfinity (20000)
                -- } (mInfinity..pInfinity)

AudioPhase ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "An absolute or relative phase value in units of 0.01 degree."
  SYNTAX        INTEGER (-18000..18000)

AudioQuality ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "An enumeration identifying an audio signal processing quality
                 level. Semantics are equipment specific. Equipment that
                 supports selectable quality levels shall as a minimum support
                 the values low and high."
  SYNTAX        INTEGER (1..127)
                -- {
                --   worst (1),
                --   low   (32),
                --   high  (96),
                --   best  (127)
                -- } (worst..best)

AudioChannel ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "An enumeration identifying an audio channel. Stereo equipment
                 shall accept the values left and right. Multi-channel equipment
                 should accept a contiguous range of values starting from 1 and
                 may disregard the association of the values 1 and 2 with left
                 and right."
  SYNTAX        INTEGER (1..240)
                -- {
                --   left  (1),
                --   right (2)
                -- } (1..240)

Aes3ChannelData ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "The channel status data associated with an AES3 channel."
  SYNTAX        OCTET STRING (SIZE(24))

Aes3UserData ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "The user status data associated with an AES3 channel."
  SYNTAX        OCTET STRING (SIZE(24))

AudioClipSelector ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "An enumeration identifying an audio clip. Semantics are
                 equipment specific."
  SYNTAX        INTEGER (1..2147483647)

AudioClipActivation ::= TEXTUAL-CONVENTION
  STATUS        current
```

```
  DESCRIPTION   "An enumeration for audio clip playback control."
  SYNTAX        INTEGER {
                  play      (1),
                  stop      (2),
                  stopAtEnd (3)
                } -- (play..stopAtEnd)

AudioClipTrackControl ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "An enumeration for audio clip track control."
  SYNTAX        INTEGER {
                  previous(1),
                  next    (2)
                } -- (previous..next)

AudioClipRepeat ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "An enumeration for audio clip repeat control."
  SYNTAX        INTEGER {
                  none  (1),
                  track (2),
                  all   (3)
                } -- (none..all)

AudioRecoveryMode ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "An enumeration identifying an audio limiter recovery mode.
                 Semantics are equipment specific."
  SYNTAX        INTEGER {
                  auto (1),
                  slow (2),
                  fast (3)
                } -- (auto..fast)

AudioAlarmStatus ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "An enumeration representing an alarm's status"
  SYNTAX        INTEGER {
                  ok      (1),
                  warning (2),
                  failure (3)
                } -- (ok..failure)

AudioLevelAlarmType ::= TEXTUAL-CONVENTION
  STATUS        current
  DESCRIPTION   "An enumeration representing the trigger condition for an audio
                 level alarm. If lower, the alarm is triggered by the audio
                 level being less than the threshold; if higher, the alarm is
                 triggered by the audio level being greater than the threshold"
  SYNTAX        INTEGER {
                  lower  (1),
                  higher (2)
                } -- (lower..higher)


-- 5.2.3 Sequences

APortEntry ::= SEQUENCE {
  aPortBlockId    BlockId,
  aPortDirection  PortDirection,
  aPortFormat     MediaFormat,
  aPortTransport  AudioTransportType,
  aPortName       Utf8String
}

Aes3DataEntry ::= SEQUENCE {
  aes3BlockId        BlockId,
  aes3ChannelId      AudioChannel,
  aes3ChannelData    Aes3ChannelData,
  aes3UserData       Aes3UserData,
```

```
  aes3ValidityError  TruthValue
}

APhantomEntry ::= SEQUENCE {
  aPhantomBlockId  BlockId,
  aPhantomEnabled  TruthValue,
  aPhantomLevel    CardinalNumber
}

ALockedEntry ::= SEQUENCE {
  aLockedBlockId          BlockId,
  aLockedTime             CardinalNumber,
  aLockedSamplesInserted  CardinalNumber,
  aLockedSamplesDropped   CardinalNumber
}

AMixerBlockEntry ::= SEQUENCE {
  aMixerBlockId       BlockId,
  aMixerFadeDuration  CardinalNumber,
  aMixerFadeNow       TruthValue
}

AMixerInputEntry ::= SEQUENCE {
  aMixerInputBlockId      BlockId,
  aMixerInputNumber       IndexNumber,
  aMixerInputLevel        AudioLevel,
  aMixerInputFadeToLevel  AudioLevel,
  aMixerInputDelay        CardinalNumber
}

ACrosspointBlockEntry ::= SEQUENCE {
  aCrosspointBlockId    BlockId,
  aCrosspointConfigure  TruthValue,
  aCrosspointCopy       BlockId
}

ACrosspointPathEntry ::= SEQUENCE {
  aCrosspointPathBlockId   BlockId,
  aCrosspointPathSrc       AudioChannel,
  aCrosspointPathDst       AudioChannel,
  aCrosspointPathGain      AudioLevel,
  aCrosspointPathNewGain   AudioLevel,
  aCrosspointPathPhase     AudioPhase,
  aCrosspointPathNewPhase  AudioPhase
}

AClipPlayerBlockEntry ::= SEQUENCE {
  acpBlockId              BlockId,
  acpSelection            AudioClipSelector,
  acpMainFaderLevelNow    AudioLevel,
  acpMainFaderLevelStart  AudioLevel,
  acpMainFaderLevelStop   AudioLevel,
  acpMainFaderDuration    CardinalNumber,
  acpClipFaderLevelNow    AudioLevel,
  acpClipFaderLevelStart  AudioLevel,
  acpClipFaderLevelStop   AudioLevel,
  acpClipFaderDuration    CardinalNumber,
  acpStartDelay           IntegerNumber,
  acpActivation           AudioClipActivation,
  acpTrackControl         AudioClipTrackControl,
  acpTrackNumber          CardinalNumber,
  acpRepeat               AudioClipRepeat
}

AClipSequenceEntry ::= SEQUENCE {
  aClipSequenceBlockId  BlockId,
  aClipSequenceId       AudioClipSelector,
  aClipSequenceName     Utf8String
}
```

```
ALimiterBlockEntry ::= SEQUENCE {
  aLimiterBlockId       BlockId,
  aLimiterThreshold     AudioLevel,
  aLimiterAttackTime    CardinalNumber,
  aLimiterGainMakeup    AudioLevel,
  aLimiterRecoveryTime  CardinalNumber,
  aLimiterRecoveryMode  AudioRecoveryMode
}

AConverterBlockEntry ::= SEQUENCE {
  aConverterBlockId       BlockId,
  aConverterQuality       AudioQuality,
  aConverterEnabled       TruthValue,
  aConverterDithering     TruthValue,
  aConverterOutputFormat  MediaFormat,
  aConverterError         TruthValue
}

ALevelAlarmBlockEntry ::= SEQUENCE {
  alaBlockId       BlockId,
  alaType          AudioLevelAlarmType,
  alaThreshold     AudioLevel,
  alaWarningTime   CardinalNumber,
  alaFailureTime   CardinalNumber,
  alaCounter       CardinalNumber,
  alaEnabled       TruthValue,
  alaStatus        AudioAlarmStatus
}


-- 5.3 Audio port and associated managed object type definitions

-- 5.3.1 Generic port functionality

aPortTable OBJECT-TYPE
  SYNTAX        SEQUENCE OF APortEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.1."
  ::= { audioPort 1 }

aPortEntry OBJECT-TYPE
  SYNTAX        APortEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.1."
  INDEX         { aPortBlockId }
  ::= { aPortTable 1 }

aPortBlockId OBJECT-TYPE
  SYNTAX        BlockId
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.1."
  ::= { aPortEntry 1 }

aPortDirection OBJECT-TYPE
  SYNTAX        PortDirection
  MAX-ACCESS    read-only
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.1."
  ::= { aPortEntry 2 }

aPortFormat OBJECT-TYPE
  SYNTAX        MediaFormat
  MAX-ACCESS    read-only
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.1."
  ::= { aPortEntry 3 }
```

```
aPortTransport OBJECT-TYPE
  SYNTAX        AudioTransportType
  MAX-ACCESS    read-only
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.1."
  ::= { aPortEntry 4 }

aPortName OBJECT-TYPE
  SYNTAX        Utf8String
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.1."
  ::= { aPortEntry 5 }


-- 5.3.2 AES3 ancillary data

aes3DataTable OBJECT-TYPE
  SYNTAX        SEQUENCE OF Aes3DataEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.2."
  ::= { audioPort 2 }

aes3DataEntry OBJECT-TYPE
  SYNTAX        Aes3DataEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.2."
  INDEX         { aes3BlockId, aes3ChannelId }
  ::= { aes3DataTable 1 }

aes3BlockId OBJECT-TYPE
  SYNTAX        BlockId
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.2."
  ::= { aes3DataEntry 1 }

aes3ChannelId OBJECT-TYPE
  SYNTAX        AudioChannel
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.2."
  ::= { aes3DataEntry 2 }

aes3ChannelData OBJECT-TYPE
  SYNTAX        Aes3ChannelData
  MAX-ACCESS    read-only
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.2."
  ::= { aes3DataEntry 3 }

aes3UserData OBJECT-TYPE
  SYNTAX        Aes3UserData
  MAX-ACCESS    read-only
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.2."
  ::= { aes3DataEntry 4 }

aes3ValidityError OBJECT-TYPE
  SYNTAX        TruthValue
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.2."
  ::= { aes3DataEntry 5 }


-- 5.3.3 Phantom power
```

```
aPhantomTable OBJECT-TYPE
  SYNTAX        SEQUENCE OF APhantomEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.3."
  ::= { audioPort 3 }

aPhantomEntry OBJECT-TYPE
  SYNTAX        APhantomEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.3."
  INDEX         { aPhantomBlockId }
  ::= { aPhantomTable 1 }

aPhantomBlockId OBJECT-TYPE
  SYNTAX        BlockId
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.3."
  ::= { aPhantomEntry 1 }

aPhantomEnabled OBJECT-TYPE
  SYNTAX        TruthValue
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.3."
  ::= { aPhantomEntry 2 }

aPhantomLevel OBJECT-TYPE
  SYNTAX        CardinalNumber
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.3."
  ::= { aPhantomEntry 3 }


-- 5.3.4 Audio locked to Reference

aLockedTable OBJECT-TYPE
  SYNTAX        SEQUENCE OF ALockedEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.4."
  ::= { audioPort 4 }

aLockedEntry OBJECT-TYPE
  SYNTAX        ALockedEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.4."
  INDEX         { aLockedBlockId }
  ::= { aLockedTable 1 }

aLockedBlockId OBJECT-TYPE
  SYNTAX        BlockId
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.4."
  ::= { aLockedEntry 1 }

aLockedTime OBJECT-TYPE
  SYNTAX        CardinalNumber
  MAX-ACCESS    read-only
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.3.4."
  ::= { aLockedEntry 2 }

aLockedSamplesInserted OBJECT-TYPE
  SYNTAX        CardinalNumber
```

```
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.3.4."
    ::= { aLockedEntry 3 }

  aLockedSamplesDropped OBJECT-TYPE
    SYNTAX        CardinalNumber
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.3.4."
    ::= { aLockedEntry 4 }


  aPortGroup OBJECT-GROUP
    OBJECTS       { aPortDirection,
                    aPortFormat,
                    aPortTransport,
                    aPortName,
                    aes3ChannelData,
                    aes3UserData,
                    aes3ValidityError,
                    aPhantomEnabled,
                    aPhantomLevel,
                    aLockedTime,
                    aLockedSamplesInserted,
                    aLockedSamplesDropped }
    STATUS        current
    DESCRIPTION   "The group of objects used to control an audio port."
    ::= { audioPort 99 }


  -- 5.4 Other audio block and associated managed object type definitions

  -- 5.4.1 Audio mixer blocks

  aMixerBlockTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF AMixerBlockEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.1."
    ::= { audioMixer 1 }

  aMixerBlockEntry OBJECT-TYPE
    SYNTAX        AMixerBlockEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.1."
    INDEX         { aMixerBlockId }
    ::= { aMixerBlockTable 1 }

  aMixerBlockId OBJECT-TYPE
    SYNTAX        BlockId
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.1."
    ::= { aMixerBlockEntry 1 }

  aMixerFadeDuration OBJECT-TYPE
    SYNTAX        CardinalNumber
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.1."
    ::= { aMixerBlockEntry 2 }

  aMixerFadeNow OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.1."
    ::= { aMixerBlockEntry 3 }
```

```
aMixerInputTable OBJECT-TYPE
  SYNTAX        SEQUENCE OF AMixerInputEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.1."
  ::= { audioMixer 2 }

aMixerInputEntry OBJECT-TYPE
  SYNTAX        AMixerInputEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.1."
  INDEX         { aMixerInputBlockId, aMixerInputNumber }
  ::= { aMixerInputTable 1 }

aMixerInputBlockId OBJECT-TYPE
  SYNTAX        BlockId
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.1."
  ::= { aMixerInputEntry 1 }

aMixerInputNumber OBJECT-TYPE
  SYNTAX        IndexNumber
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.1."
  ::= { aMixerInputEntry 2 }

aMixerInputLevel OBJECT-TYPE
  SYNTAX        AudioLevel
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.1."
  ::= { aMixerInputEntry 3 }

aMixerInputFadeToLevel OBJECT-TYPE
  SYNTAX        AudioLevel
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.1."
  ::= { aMixerInputEntry 4 }

aMixerInputDelay OBJECT-TYPE
  SYNTAX        CardinalNumber
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.1."
  ::= { aMixerInputEntry 5 }

audioMixerGroup OBJECT-GROUP
  OBJECTS       { aMixerFadeDuration,
                  aMixerFadeNow,
                  aMixerInputLevel,
                  aMixerInputFadeToLevel,
                  aMixerInputDelay }
  STATUS        current
  DESCRIPTION   "The group of objects used to control an audio mixer block."
  ::= { audioMixer 99 }


-- 5.4.2 Audio crosspoint blocks

aCrosspointBlockTable OBJECT-TYPE
  SYNTAX        SEQUENCE OF ACrosspointBlockEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.2."
  ::= { audioCrosspoint 1 }
```

```
aCrosspointBlockEntry OBJECT-TYPE
  SYNTAX        ACrosspointBlockEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.2."
  INDEX         { aCrosspointBlockId }
  ::= { aCrosspointBlockTable 1 }

aCrosspointBlockId OBJECT-TYPE
  SYNTAX        BlockId
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.2."
  ::= { aCrosspointBlockEntry 1 }

aCrosspointConfigure OBJECT-TYPE
  SYNTAX        TruthValue
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.2."
  ::= { aCrosspointBlockEntry 2 }

aCrosspointCopy OBJECT-TYPE
  SYNTAX        BlockId
  MAX-ACCESS    write-only
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.2."
  ::= { aCrosspointBlockEntry 3 }

aCrosspointPathTable OBJECT-TYPE
  SYNTAX        SEQUENCE OF ACrosspointPathEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.2."
  ::= { audioCrosspoint 2 }

aCrosspointPathEntry OBJECT-TYPE
  SYNTAX        ACrosspointPathEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.2."
  INDEX         { aCrosspointPathBlockId,
                  aCrosspointPathSrc,
                  aCrosspointPathDst }
  ::= { aCrosspointPathTable 1 }

aCrosspointPathBlockId OBJECT-TYPE
  SYNTAX        BlockId
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.2."
  ::= { aCrosspointPathEntry 1 }

aCrosspointPathSrc OBJECT-TYPE
  SYNTAX        AudioChannel
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.2."
  ::= { aCrosspointPathEntry 2 }

aCrosspointPathDst OBJECT-TYPE
  SYNTAX        AudioChannel
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.2."
  ::= { aCrosspointPathEntry 3 }

aCrosspointPathGain OBJECT-TYPE
  SYNTAX        AudioLevel
```

```
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.2."
  ::= { aCrosspointPathEntry 4 }

aCrosspointPathNewGain OBJECT-TYPE
  SYNTAX        AudioLevel
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.2."
  ::= { aCrosspointPathEntry 5 }

aCrosspointPathPhase OBJECT-TYPE
  SYNTAX        AudioPhase
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.2."
  ::= { aCrosspointPathEntry 6 }

aCrosspointPathNewPhase OBJECT-TYPE
  SYNTAX        AudioPhase
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.2."
  ::= { aCrosspointPathEntry 7 }

audioCrosspointGroup OBJECT-GROUP
  OBJECTS       { aCrosspointConfigure,
                  aCrosspointCopy,
                  aCrosspointPathGain,
                  aCrosspointPathNewGain,
                  aCrosspointPathPhase,
                  aCrosspointPathNewPhase }
  STATUS        current
  DESCRIPTION   "The group of objects used to control an audio crosspoint
                 block."
  ::= { audioCrosspoint 99 }


-- 5.4.3 Audio clip player blocks

aClipPlayerBlockTable OBJECT-TYPE
  SYNTAX        SEQUENCE OF AClipPlayerBlockEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
  ::= { audioClipPlayer 1 }

aClipPlayerBlockEntry OBJECT-TYPE
  SYNTAX        AClipPlayerBlockEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
  INDEX         { acpBlockId }
  ::= { aClipPlayerBlockTable 1 }

acpBlockId OBJECT-TYPE
  SYNTAX        BlockId
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
  ::= { aClipPlayerBlockEntry 1 }

acpSelection OBJECT-TYPE
  SYNTAX        AudioClipSelector
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
  ::= { aClipPlayerBlockEntry 2 }
```

```
  acpMainFaderLevelNow OBJECT-TYPE
    SYNTAX        AudioLevel
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
    ::= { aClipPlayerBlockEntry 3 }

  acpMainFaderLevelStart OBJECT-TYPE
    SYNTAX        AudioLevel
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
    ::= { aClipPlayerBlockEntry 4 }

  acpMainFaderLevelStop OBJECT-TYPE
    SYNTAX        AudioLevel
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
    ::= { aClipPlayerBlockEntry 5 }

  acpMainFaderDuration OBJECT-TYPE
    SYNTAX        CardinalNumber
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
    ::= { aClipPlayerBlockEntry 6 }

  acpClipFaderLevelNow OBJECT-TYPE
    SYNTAX        AudioLevel
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
    ::= { aClipPlayerBlockEntry 7 }

  acpClipFaderLevelStart OBJECT-TYPE
    SYNTAX        AudioLevel
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
    ::= { aClipPlayerBlockEntry 8 }

  acpClipFaderLevelStop OBJECT-TYPE
    SYNTAX        AudioLevel
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
    ::= { aClipPlayerBlockEntry 9 }

  acpClipFaderDuration OBJECT-TYPE
    SYNTAX        CardinalNumber
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
    ::= { aClipPlayerBlockEntry 10 }

  acpStartDelay OBJECT-TYPE
    SYNTAX        IntegerNumber
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
    ::= { aClipPlayerBlockEntry 11 }

  acpActivation OBJECT-TYPE
    SYNTAX        AudioClipActivation
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
    ::= { aClipPlayerBlockEntry 12 }
```

```
acpTrackControl OBJECT-TYPE
  SYNTAX        AudioClipTrackControl
  MAX-ACCESS    write-only
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
  ::= { aClipPlayerBlockEntry 13 }

acpTrackNumber OBJECT-TYPE
  SYNTAX        CardinalNumber
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
  ::= { aClipPlayerBlockEntry 14 }

acpRepeat OBJECT-TYPE
  SYNTAX        AudioClipRepeat
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
  ::= { aClipPlayerBlockEntry 15 }

aClipSequenceTable OBJECT-TYPE
  SYNTAX        SEQUENCE OF AClipSequenceEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
  ::= { audioClipPlayer 2 }

aClipSequenceEntry OBJECT-TYPE
  SYNTAX        AClipSequenceEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
  INDEX         { aClipSequenceBlockId, aClipSequenceId }
  ::= { aClipSequenceTable 1 }

aClipSequenceBlockId OBJECT-TYPE
  SYNTAX        BlockId
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
  ::= { aClipSequenceEntry 1 }

aClipSequenceId OBJECT-TYPE
  SYNTAX        AudioClipSelector
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
  ::= { aClipSequenceEntry 2 }

aClipSequenceName OBJECT-TYPE
  SYNTAX        Utf8String
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.3."
  ::= { aClipSequenceEntry 3 }

audioClipPlayerGroup OBJECT-GROUP
  OBJECTS       { acpSelection,
                  acpMainFaderLevelNow,
                  acpMainFaderLevelStart,
                  acpMainFaderLevelStop,
                  acpMainFaderDuration,
                  acpClipFaderLevelNow,
                  acpClipFaderLevelStart,
                  acpClipFaderLevelStop,
                  acpClipFaderDuration,
                  acpStartDelay,
                  acpActivation,
                  acpTrackControl,
```

```
                  acpTrackNumber,
                  acpRepeat,
                  aClipSequenceName }
    STATUS        current
    DESCRIPTION   "The group of objects used to control an audio clip player
                   block."
    ::= { audioClipPlayer 99 }


  -- 5.4.4 Audio limiter blocks

  aLimiterBlockTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF ALimiterBlockEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.4."
    ::= { audioLimiter 1 }

  aLimiterBlockEntry OBJECT-TYPE
    SYNTAX        ALimiterBlockEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.4."
    INDEX         { aLimiterBlockId }
    ::= { aLimiterBlockTable 1 }

  aLimiterBlockId OBJECT-TYPE
    SYNTAX        BlockId
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.4."
    ::= { aLimiterBlockEntry 1 }

  aLimiterThreshold OBJECT-TYPE
    SYNTAX        AudioLevel
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.4."
    ::= { aLimiterBlockEntry 2 }

  aLimiterAttackTime OBJECT-TYPE
    SYNTAX        CardinalNumber
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.4."
    ::= { aLimiterBlockEntry 3 }

  aLimiterGainMakeup OBJECT-TYPE
    SYNTAX        AudioLevel
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.4."
    ::= { aLimiterBlockEntry 4 }

  aLimiterRecoveryTime OBJECT-TYPE
    SYNTAX        CardinalNumber
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.4."
    ::= { aLimiterBlockEntry 5 }

  aLimiterRecoveryMode OBJECT-TYPE
    SYNTAX        AudioRecoveryMode
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION   "See IEC 62379-2 clause 5.4.4."
    ::= { aLimiterBlockEntry 6 }

  audioLimiterGroup OBJECT-GROUP
    OBJECTS       { aLimiterThreshold,
```

```
                  aLimiterAttackTime,
                  aLimiterGainMakeup,
                  aLimiterRecoveryTime,
                  aLimiterRecoveryMode }
  STATUS        current
  DESCRIPTION   "The group of objects used to control an audio limiter block."
  ::= { audioLimiter 99 }


-- 5.4.5 Audio converter blocks

aConverterBlockTable OBJECT-TYPE
  SYNTAX        SEQUENCE OF AConverterBlockEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.5."
  ::= { audioConverter 1 }

aConverterBlockEntry OBJECT-TYPE
  SYNTAX        AConverterBlockEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.5."
  INDEX         { aConverterBlockId }
  ::= { aConverterBlockTable 1 }

aConverterBlockId OBJECT-TYPE
  SYNTAX        BlockId
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.5."
  ::= { aConverterBlockEntry 1 }

aConverterQuality OBJECT-TYPE
  SYNTAX        AudioQuality
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.5."
  ::= { aConverterBlockEntry 2 }

aConverterEnabled OBJECT-TYPE
  SYNTAX        TruthValue
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.5."
  ::= { aConverterBlockEntry 3 }

aConverterDithering OBJECT-TYPE
  SYNTAX        TruthValue
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.5."
  ::= { aConverterBlockEntry 4 }

aConverterOutputFormat OBJECT-TYPE
  SYNTAX        MediaFormat
  MAX-ACCESS    read-only
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.5."
  ::= { aConverterBlockEntry 5 }

aConverterError OBJECT-TYPE
  SYNTAX        TruthValue
  MAX-ACCESS    read-only
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.5."
  ::= { aConverterBlockEntry 6 }

audioConverterGroup OBJECT-GROUP
  OBJECTS       { aConverterQuality,
```

```
                    aConverterEnabled,
                    aConverterDithering,
                    aConverterOutputFormat,
                    aConverterError}
    STATUS          current
    DESCRIPTION     "The group of objects used to control an audio converter block."
    ::= { audioConverter 99 }


  -- 5.4.6 Audio level alarm block

  aLevelAlarmBlockTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF ALevelAlarmBlockEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION     "See IEC 62379-2 clause 5.4.6."
    ::= { audioLevelAlarm 1 }

  aLevelAlarmBlockEntry OBJECT-TYPE
    SYNTAX          ALevelAlarmBlockEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION     "See IEC 62379-2 clause 5.4.6."
    INDEX           { alaBlockId }
    ::= { aLevelAlarmBlockTable 1 }

  alaBlockId OBJECT-TYPE
    SYNTAX          BlockId
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION     "See IEC 62379-2 clause 5.4.6."
    ::= { aLevelAlarmBlockEntry 1 }

  alaType OBJECT-TYPE
    SYNTAX          AudioLevelAlarmType
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION     "See IEC 62379-2 clause 5.4.6."
    ::= { aLevelAlarmBlockEntry 2 }

  alaThreshold OBJECT-TYPE
    SYNTAX          AudioLevel
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION     "See IEC 62379-2 clause 5.4.6."
    ::= { aLevelAlarmBlockEntry 3 }

  alaWarningTime OBJECT-TYPE
    SYNTAX          CardinalNumber
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION     "See IEC 62379-2 clause 5.4.6."
    ::= { aLevelAlarmBlockEntry 4 }

  alaFailureTime OBJECT-TYPE
    SYNTAX          CardinalNumber
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION     "See IEC 62379-2 clause 5.4.6."
    ::= { aLevelAlarmBlockEntry 5 }

  alaCounter OBJECT-TYPE
    SYNTAX          CardinalNumber
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION     "See IEC 62379-2 clause 5.4.6."
    ::= { aLevelAlarmBlockEntry 6 }

  alaEnabled OBJECT-TYPE
    SYNTAX          TruthValue
```

```
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.6."
  ::= { aLevelAlarmBlockEntry 7 }

alaStatus OBJECT-TYPE
  SYNTAX        AudioAlarmStatus
  MAX-ACCESS    read-only
  STATUS        current
  DESCRIPTION   "See IEC 62379-2 clause 5.4.6."
  ::= { aLevelAlarmBlockEntry 8 }

audioLevelAlarmGroup OBJECT-GROUP
  OBJECTS       { alaType,
                  alaThreshold,
                  alaWarningTime,
                  alaFailureTime,
                  alaCounter,
                  alaEnabled,
                  alaStatus }
  STATUS        current
  DESCRIPTION    "The group of objects used to control an audio level alarm
block."
  ::= { audioLevelAlarm 99 }


-- Compliance statements

  audioMIBComplianceV1 MODULE-COMPLIANCE
  STATUS        current
  DESCRIPTION   "The compliance statement for entities that conform to
                 IEC 62379-2 (2008)."

  MODULE  -- this module

    MANDATORY-GROUPS {}

    GROUP       aPortGroup
    DESCRIPTION "Mandatory for equipment that contains one or more analogue
                 ports."

    GROUP       audioMixerGroup
    DESCRIPTION "Mandatory for equipment that contains one or more audio
                 mixer blocks."

    GROUP       audioCrosspointGroup
    DESCRIPTION "Mandatory for equipment that contains one or more audio
                 crosspoint blocks."

    GROUP       audioClipPlayerGroup
    DESCRIPTION "Mandatory for equipment that contains one or more audio
                 clip player blocks."

    GROUP       audioLimiterGroup
    DESCRIPTION "Mandatory for equipment that contains one ormore audio
                 limiter blocks."

    GROUP       audioConverterGroup
    DESCRIPTION "Mandatory for equipment that contains one or more audio
                 converter blocks."

    GROUP       audioLevelAlarmGroup
    DESCRIPTION "Mandatory for equipment that contains one or more audio
                 alarm blocks."

  MODULE IEC62379-2-STATUS-MIB

    GROUP       audioFormatsMapGroup
    DESCRIPTION "Mandatory for equipment that reports audio format
                 information in status pages."
```

```
    ::= {audioMIBCompliance 1 }

END
```

Annex C
(informative)

Machine-readable status page group definitions

This annex provides a machine-readable version of the status page group definitions which is intended to be interpretable by standard MIB browsing software tools. If there is any inconsistency between this annex and Clause 6, Clause 6 takes precedence.

The format used to describe the status page group identifiers conforms to IETF STD 58 (SMIv2).

```
IEC62379-2-STATUS-GROUPS DEFINITIONS ::= BEGIN

IMPORTS

  MODULE-IDENTITY, OBJECT-IDENTITY
    FROM SNMPv2-SMI

  iec62379
    FROM IEC62379-1-MIB;

audioStatusGroup MODULE-IDENTITY
  LAST-UPDATED  "200804301500Z"
  ORGANIZATION  "IEC PT62379"
  CONTACT-INFO  "Not specified"
  DESCRIPTION   "The status page group identifiers defined in clause 6.5 of
                 IEC 62379-2."
  REVISION      "200804301500Z"
  DESCRIPTION   "Edition 1.0 final draft."
  ::= { audio 3 }

audio OBJECT IDENTIFIER ::= { iec62379 2 }


-- 6.5 Page groups

audioPorts OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "See clause 6.5.1 of IEC 62379-2."
  ::= { audioStatusGroup 1 }

standardAudioBlocks OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "See clause 6.5.2 of IEC 62379-2."
  ::= { audioStatusGroup 2 }

audioAlarms OBJECT-IDENTITY
  STATUS        current
  DESCRIPTION   "See clause 6.5.3 of IEC 62379-2."
  ::= { audioStatusGroup 3 }

END
```

Annex D
(informative)

Machine-readable status page MIB definitions

This annex provides a machine-readable version of the status page MIB definitions which is intended to be interpretable by standard MIB browsing software tools. If there is any inconsistency between this annex and Clause 6, Clause 6 takes precedence.

The format used to describe the status page group identifiers conforms to IETF STD 58 (SMIv2).

```
IEC62379-2-STATUS-MIB DEFINITIONS ::= BEGIN

IMPORTS

  MODULE-IDENTITY, OBJECT-IDENTITY
    FROM SNMPv2-SMI

  iec62379, IndexNumber, MediaFormat
    FROM IEC62379-1-MIB;

audioStatusMIB MODULE-IDENTITY
  LAST-UPDATED  "200804301500Z"
  ORGANIZATION  "IEC PT62379"
  CONTACT-INFO  "Not specified"
  DESCRIPTION   "The MIB module for managing audio status pages in
                 IEC 62379-2."
  REVISION      "200804301500Z"
  DESCRIPTION   "Edition 1.0 final draft."
  ::= { audio 4 }


-- Node definitions

audio OBJECT IDENTIFIER ::= { iec62379 2 }

audioStatusMIBCompliance OBJECT IDENTIFIER ::= { audioStatusMIB 1 }

audioFormatsMap OBJECT IDENTIFIER ::= { audioStatusMIB 2 }


-- 6.2 Application wide type definitions

-- 6.2.1 Textual conventions

AudioConverterStatus ::= TEXTUAL-CONVENTION
  STATUS current
  DESCRIPTION "A set of bits representing the status of an audio converter:
               bit 0 (lsb) = converter enabled (1=enabled, 0=not enabled)
               bit 1       = dithering applied (1=dithering, 0=no dithering)
               bit 2       = conversion ok (1=input converted, 0=converter error)
               bit 3       = reserved
               bit 4       = reserved
               bit 5       = reserved
               bit 6       = reserved
               bit 7 (msb) = reserved"
  SYNTAX OCTET STRING (SIZE(1))


-- 6.2.2 Sequences

AudioFormatsMapEntry ::= SEQUENCE {
  afmNumber  IndexNumber,
  afmFormat  MediaFormat
```

```
}


-- 6.3 Audio formats mapping

audioFormatsMapTable OBJECT-TYPE
  SYNTAX        SEQUENCE OF AudioFormatsMapEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See clause 6.3 of IEC 62379-2."
  ::= { audioFormatsMap 1 }

audioFormatsMapEntry OBJECT-TYPE
  SYNTAX        AudioFormatsMapEntry
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See clause 6.3 of IEC 62379-2."
  INDEX         { afmNumber }
  ::= { audioFormatsMapTable 1 }

afmNumber OBJECT-TYPE
  SYNTAX        IndexNumber
  MAX-ACCESS    not-accessible
  STATUS        current
  DESCRIPTION   "See clause 6.3 of IEC 62379-2."
  ::= { audioFormatsMapEntry 1 }

afmFormat OBJECT-TYPE
  SYNTAX        MediaFormat
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "See clause 6.3 of IEC 62379-2."
  ::= { audioFormatsMapEntry 2 }

audioFormatsMapGroup OBJECT-GROUP
  OBJECTS       { afmFormat }
  STATUS        current
  DESCRIPTION   "The group of objects used to represent audio formats in
                 a status page "
  ::= { audioFormatsMap 99 }


-- Compliance statements

audioStatusMIBComplianceV1 MODULE-COMPLIANCE
  STATUS        current
  DESCRIPTION   "The compliance statement for entities that conform to
                 IEC 62379-2 (2008)."

  MODULE -- this module
  GROUP         audioFormatsMapGroup
  DESCRIPTION   "Mandatory for equipment that reports audio format
                 information in status pages."
  ::= { audioStatusMIBCompliance 1 }

END
```

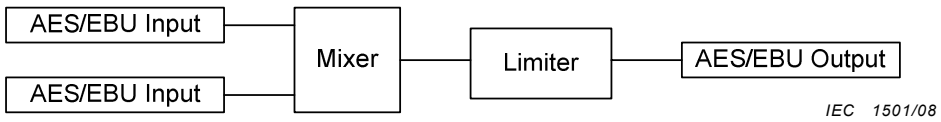
Annex E
(informative)

Worked examples

This annex provides two worked examples to show how Parts 1 and 2 of this standard may be used to model audio devices. If there is any inconsistency between this annex and the main clauses of Parts 1 and 2, those clauses take precedence.

E.1 Example 1

To show the relationships between the various tables defined in Parts 1 and 2 of IEC 62379 consider a simple audio device that mixes two inputs and provides a limiter, as represented by the following diagram:



Assume the unit is capable of carrying two channel stereo 24 bit audio at either 44,1k or 48k but that it is currently operating at 48k.

E.1.1 Block table

The first thing that is required is a block table showing all the functional blocks that make up the unit. The mapping of a block to a blockId is arbitrary; what matters is that a particular block is always associated with the same blockId. The connector table will show the structure of the unit by showing the connections between the blocks.

The table is located at *iso.standard.iec67329.general.generalMIB.block.blockTable (1.0.62379.1.1.2.1)*

As there are five blocks in the example, this table will have five rows:

.blockTable.blockEntry (.1.1)	
blockId (1)	*blockType (2)*
①	1.0.62379.2.1.1 (Audio port)
②	1.0.62379.2.1.1 (Audio port)
③	1.0.62379.2.1.2 (Mixer block)
④	1.0.62379.2.1.5 (Limiter block)
⑤	1.0.62379.2.1.1 (Audio port)

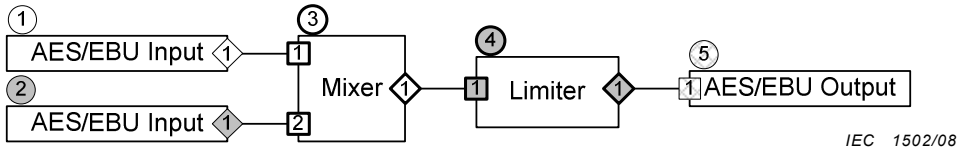
E.1.2 Connector table

To show the structure of the unit, a connector table is required to express all the links that exist between the blocks defined in the block table.

This table is located at *iso.standard.iec67329.general.generalMIB.block.connectorTable (1.0.62379.1.1.2.2)*

.connectorTable.connectorEntry (.2.1)			
connRxBlockId (1)	*connRxBlockInput (2)*	*connTxBlockId (3)*	*connTxBlockOutput (4)*
③	□1	①	◇1
③	□2	②	◇1
④	□1	③	◇1
⑤	□1	④	◇1

Below is the diagram of the unit annotated with each block's blockId (in circles), input numbers (squares) and output numbers (diamonds):



E.1.3 Mode table

The mode table shows all the formats that are valid for each internal output.

This table is located at *iso.standard.iec67329.general.generalMIB.block.modeTable (1.0.62379.1.1.2.3)*

In this example, there are four internal outputs, each capable of running at 44,1k or 48k. There are therefore eight entries in this table:

.modeTable.modeEntry (.3.1)			
mBlockId (1)	*mBlockOutput (2)*	*mMediaFormat (3)*	*mEnabled (4)*
①	◇1	1.0.62379.2.2.1.3.2.2.24.44100	true (1)
①	◇1	1.0.62379.2.2.1.3.2.2.24.48000	true (1)
②	◇1	1.0.62379.2.2.1.3.2.2.24.44100	true (1)
②	◇1	1.0.62379.2.2.1.3.2.2.24.48000	true (1)
③	◇1	1.0.62379.2.2.1.3.2.2.24.44100	true (1)
③	◇1	1.0.62379.2.2.1.3.2.2.24.48000	true (1)
④	◇1	1.0.62379.2.2.1.3.2.2.24.44100	true (1)
④	◇1	1.0.62379.2.2.1.3.2.2.24.48000	true (1)

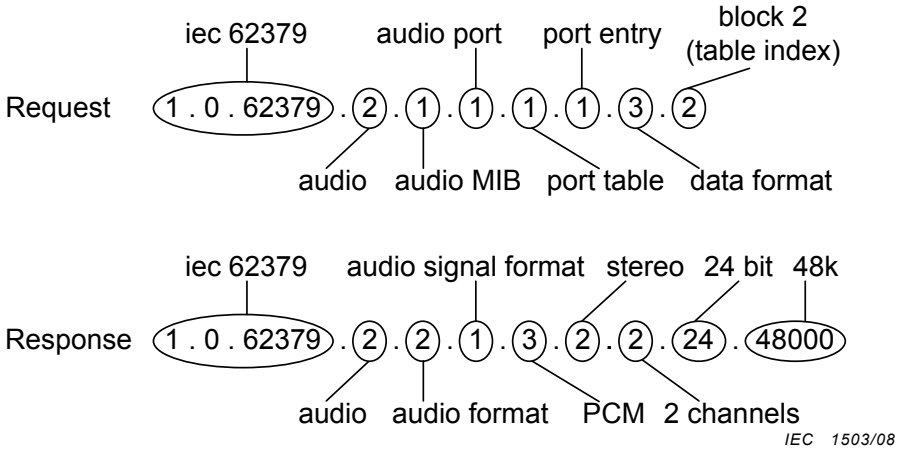
E.1.4 Audio ports

Audio ports are rooted at *iso.standard.iec62379.audio.audioMIB.audioPort (1.0.62379.2.1.1)*

The block consists of a table, *aPortTable*, with entries for all ports in the unit, located at *1.0.62379.2.1.1.1*. In this example, there are three entries in this table, one for each port. *aPortDataFormat* shows that all ports are carrying two channel stereo PCM audio, at 24 bits, 48k; *aPortTransport* shows that this audio is carried in an AES3 transport:

.aPortTable.aPortEntry (.1.1)				
aPortBlockId (1)	*aPortDirection (2)*	*aPortDataFormat (3)*	*aPortTransport (4)*	*aPortName (5)*
①	input (1)	1.0.62379.2.2.1.3.2.2.24.48000	1.0.62379.2.2.2.2	
②	input (1)	1.0.62379.2.2.1.3.2.2.24.48000	1.0.62379.2.2.2.2	
⑤	output (2)	1.0.62379.2.2.1.3.2.2.24.48000	1.0.62379.2.2.2.2	

So, to find the data format of block 2, issue a GET on *1.0.62379.2.1.1.1.1.3.2* to get a GET RESPONSE value of *1.0.62379.2.2.1.3.2.2.24.48000*:



The port block also consists of other tables to represent functionality specific to a particular port. As the ports in this example represent AES/EBU audio, AES3 data may be obtained from *aes3DataTable*, located at *1.0.62379.2.1.1.2*. There is an entry here for each channel of each port that supports AES3 data: in this example, this table has six rows:

.aAes3DataTable.aAes3DataEntry (.2.1)				
aes3 BlockId (1)	*aes3 ChannelId (2)*	*aes3 ChannelData (3)*	*aes3 UserData (4)*	*aes3 ValidityError (5)*
①	1			
①	2			
②	1			
②	2			
⑤	1			
⑤	2			

E.1.5 Mixer block

The mixer block is rooted at *iso.standard.iec62379.audio.audioMIB.audioMixer (1.0.62379.2.1.2)*

This block consists of two tables, *aMixerBlockTable*, located at *1.0.62379.2.1.2.1* and *aMixerInputTable*, located at *1.0.62379.2.1.2.2*. There is one entry in *aMixerBlockTable* for each mixer block in a unit and an entry in *aMixerInputTable* for each input of each mixer.

As the example has one mixer block with two inputs, *aMixerBlockTable* has one entry (of type *AMixerBlockEntry*) and *aMixerInputTable* has two entries (of type *AMixerInputEntry*):

.aMixerBlockTable.aMixerBlockEntry (.1.1)		
aMixerBlockId (1)	*aMixerFadeDuration (2)*	*aMixerFadeNow (3)*
③		

.aMixerInputTable.aMixerInputEntry (.2.1)				
aMixerInput BlockId (1)	*aMixerInput Number (2)*	*aMixerInput Level (3)*	*aMixerInput FadeToLevel (4)*	*aMixerInput Delay (5)*
③	1			
③	2			

E.1.6 Limiter block

The limiter block is located at *iso.standard.iec62379.audio.audioMIB.audioLimiter (1.0.62379.2.1.5)*.

This block consists of a single table, *aLimiterBlockTable*, located at *1.0.62379.2.1.5.1*. As there is only one limiter in the example, the limiter table has one entry (of type *ALimiterBlockEntry*):

.aLimiterTable.aLimiterEntry (.1.1)					
aLimiter BlockId (1)	*aLimiter Threshold (2)*	*aLimiter AttackTime (3)*	*aLimiter GainMakeup (4)*	*aLimiter RecoveryTime (5)*	*aLimiter RecoveryMode (6)*
④					

To set the limiter threshold to –60dB set *1.0.62379.2.1.5.1.1.2.4* to *–6000*.

E.1.7 Summary of tables

iso.standard.iec62379 (1.0.62379)

.general.generalMIB (.1.1)

.block (.2)

.blockTable.blockEntry (.1.1)

blockId (1)	blockType (2)
①	1.0.62379.2.1.1 (Audio port)
②	1.0.62379.2.1.1 (Audio port)
③	1.0.62379.2.1.2 (Mixer block)
④	1.0.62379.2.1.5 (Limiter block)
⑤	1.0.62379.2.1.1 (Audio port)

.connectorTable.connectorEntry (.2.1)

connRxBlockId (1)	connRxBlockInput (2)	connTxBlockId (3)	connTxBlockOutput (4)
③	1	①	◇1
③	2	②	◇2
④	1	③	◇1
⑤	1	④	◇2

.modeTable.modeEntry (.3.1)

mBlockId (1)	mBlockOutput (2)	mMediaFormat (3)	mEnabled (4)
①	◇1	1.0.62379.2.2.1.3.2.2.24.44100	true (1)
①	◇1	1.0.62379.2.2.1.3.2.2.24.48000	true (1)
②	◇2	1.0.62379.2.2.1.3.2.2.24.44100	true (1)
②	◇2	1.0.62379.2.2.1.3.2.2.24.48000	true (1)
③	◇1	1.0.62379.2.2.1.3.2.2.24.44100	true (1)
③	◇1	1.0.62379.2.2.1.3.2.2.24.48000	true (1)
④	◇2	1.0.62379.2.2.1.3.2.2.24.44100	true (1)
④	◇2	1.0.62379.2.2.1.3.2.2.24.48000	true (1)

.audio.audioMIB (.2.1)

.audioPort (.1)

.aPortTable.aPortEntry (.1.1)

aPortBlockId (1)	aPortDirection (2)	aPortDataFormat (3)	aPortTransport (4)	aPortName (5)
①	input (1)	1.0.62379.2.2.1.3.2.2.24.48000	1.0.62379.2.2.2.2	
②	input (1)	1.0.62379.2.2.1.3.2.2.24.48000	1.0.62379.2.2.2.2	
⑤	output (2)	1.0.62379.2.2.1.3.2.2.24.48000	1.0.62379.2.2.2.2	

.aAes3DataTable.aAes3DataEntry (.2.1)

aes3 BlockId (1)	aes3 ChannelId (2)	aes3 ChannelData (3)	aes3 UserData (4)	aes3 ValidityError (5)
①	1			
①	2			
②	1			
②	2			
⑤	1			
⑤	2			

.audioMixer (.2)

.aMixerBlockTable.aMixerBlockEntry (.1.1)

aMixerBlockId (1)	aMixerFadeDuration (2)	aMixerFadeNow (3)
③		

.aMixerInputTable.aMixerInputEntry (.2.1)

aMixerInput BlockId (1)	aMixerInput Number (2)	aMixerInput Level (3)	aMixerInput FadeToLevel (4)	aMixerInput Delay (5)
③	1			
③	2			

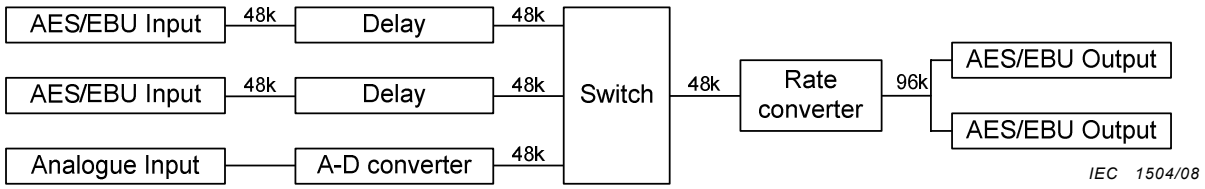
.audioLimiter (.5)

.aLimiterTable.aLimiterEntry (.1.1)

aLimiter BlockId (1)	aLimiter Threshold (2)	aLimiter AttackTime (3)	aLimiter GainMakeup (4)	aLimiter RecoveryTime (5)	aLimiter RecoveryMode (6)
④					

E.2 Example 2

Below, another device is shown.



This unit works at 24 bit, 48k until the rate converter, which can output 16 or 24 bit audio at 44,1k, 48k, 88,2k or 96k sampling. It is currently configured to output 24 bit, 96k. The entire unit works with two channel stereo.

E.2.1 Block and connector tables

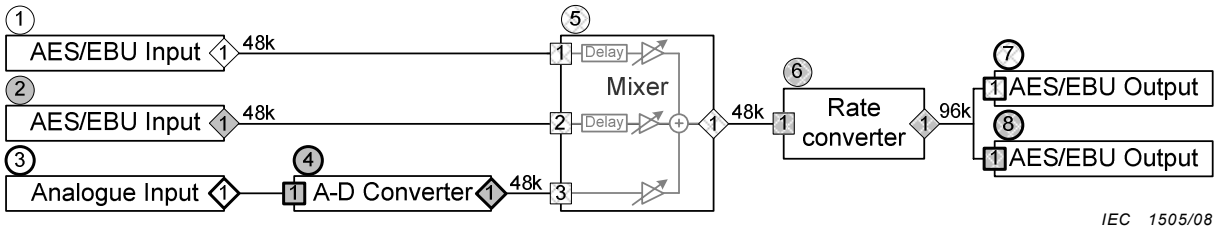
The two delays and the switch can be implemented using a mixer block with three inputs, which leads to the following block table:

.blockTable.blockEntry (.1.1)	
blockId (1)	*blockType (2)*
①	1.0.62379.2.1.1 (Audio port)
②	1.0.62379.2.1.1 (Audio port)
③	1.0.62379.2.1.1 (Audio port)
④	1.0.62379.2.1.6 (Converter block)
⑤	1.0.62379.2.1.2 (Mixer block)
⑥	1.0.62379.2.1.6 (Converter block)
⑦	1.0.62379.2.1.1 (Audio port)
⑧	1.0.62379.2.1.1 (Audio port)

The two AES/EBU outputs are derived from the same source, the rate converter, as shown in the connector table:

.connectorTable.connectorEntry (.2.1)			
connRxBlockId (1)	*connRxBlockInput (2)*	*connTxBlockId (3)*	*connTxBlockOutput (4)*
④	1	③	◇1
⑤	1	①	◇1
⑤	2	②	◇1
⑤	3	④	◇1
⑥	1	⑤	◇1
⑦	1	⑥	◇1
⑧	1	⑥	◇1

This diagram shows the unit annotated with block, input and output numbers. It also shows the delays and switch redrawn as a mixer block:



E.2.2 Mode table

The mode table shows blocks 1, 2, 4 and 5 operating at 24 bit, 48k, along with analogue audio in block 3. There are eight entries for the rate converter: one for each combination of 16 or 24 bit and 44,1, 48, 88,2 or 96k. The *mEnabled* column is used by a converter block to specify which of the supported modes should be output: in this case, 24 bit, 96k is selected.

.modeTable.modeEntry (.3.1)			
mBlockId (1)	*mBlockOutput (2)*	*mMediaFormat (3)*	*mEnabled (4)*
1	1	1.0.62379.2.2.1.3.2.2.24.48000	true (1)
2	1	1.0.62379.2.2.1.3.2.2.24.48000	true (1)
3	1	1.0.62379.2.2.1.2.2.2	true (1)
4	1	1.0.62379.2.2.1.3.2.2.24.48000	true (1)
5	1	1.0.62379.2.2.1.3.2.2.24.48000	true (1)
6	1	1.0.62379.2.2.1.3.2.2.16.44100	false (2)
6	1	1.0.62379.2.2.1.3.2.2.16.48000	false (2)
6	1	1.0.62379.2.2.1.3.2.2.16.88200	false (2)
6	1	1.0.62379.2.2.1.3.2.2.16.96000	false (2)
6	1	1.0.62379.2.2.1.3.2.2.24.44100	false (2)
6	1	1.0.62379.2.2.1.3.2.2.24.48000	false (2)
6	1	1.0.62379.2.2.1.3.2.2.24.88200	false (2)
6	1	1.0.62379.2.2.1.3.2.2.24.96000	true (1)

E.2.3 Audio ports

There are four AES/EBU ports: two inputs working at 24 bit, 48k and two output ports, working at 24 bit, 96k. In addition, there is a two channel stereo analogue port too.

.aPortTable.aPortEntry (.1.1)				
aPortBlockId (1)	*aPortDirection (2)*	*aPortDataFormat (3)*	*aPortTransport (4)*	*aPortName (5)*
1	input (1)	1.0.62379.2.2.1.3.2.2.24.48000	1.0.62379.2.2.2.2	
2	input (1)	1.0.62379.2.2.1.3.2.2.24.48000	1.0.62379.2.2.2.2	
3	input (1)	1.0.62379.2.2.1.2.2.2	1.0.62379.2.2.2.1	
7	output (2)	1.0.62379.2.2.1.3.2.2.24.96000	1.0.62379.2.2.2.2	
8	output (2)	1.0.62379.2.2.1.3.2.2.24.96000	1.0.62379.2.2.2.2	

There is also an *aes3DataTable* with two entries for each of the four AES/EBU ports.

E.2.4 Mixer block

The mixer block as defined ‘comes with’ a delay on each input. In this example, there is no delay on the analogue input so the delay entry for input 3 will be fixed at zero.

.aMixerBlockTable.aMixerBlockEntry (.1.1)		
aMixerBlockId (1)	*aMixerFadeDuration (2)*	*aMixerFadeNow (3)*
⑤		

.aMixerInputTable.aMixerInputEntry (.2.1)				
aMixerInput BlockId (1)	*aMixerInput Number (2)*	*aMixerInput Level (3)*	*aMixerInput FadeToLevel (4)*	*aMixerInput Delay (5)*
⑤	1	As this block represents a switch, one of these will be at fullscale, the other two at minus infinity		Inputs 1 and 2 can accept a delay value; input 3 has a fixed delay of zero
⑤	2			
⑤	3			

E.2.5 Converter blocks

Both the analogue-digital converter and the sample rate converter can be implemented by a converter block, rooted at *iso.standard.iec62379.audio.audioMIB.converter (1.0.62379.2.1.6)*.

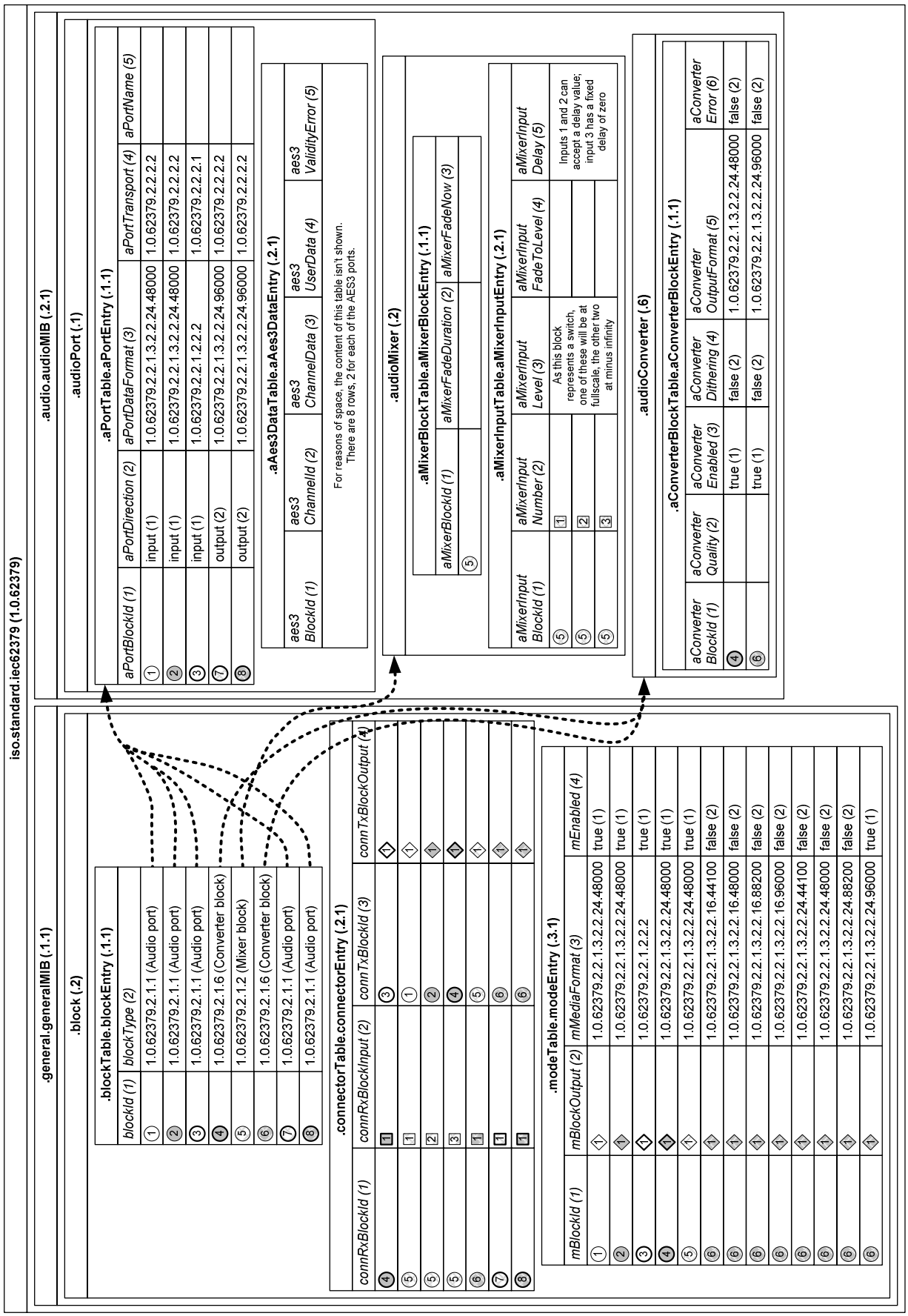
This block consists of a single table, *aConverterBlockTable*, located at *1.0.62379.2.1.6.1*, which contains an entry for each converter.

As there are two converters, *aConverterBlockTable* contains two entries. Note that although the sample rate converter output goes to two AES3 ports they both come from the same output of the rate converter.

.aConverterBlockTable.aConverterBlockEntry (.1.1)					
aConverter BlockId (1)	*aConverter Quality (2)*	*aConverter Enabled (3)*	*aConverter Dithering (4)*	*aConverter OutputFormat (5)*	*aConverter Error (6)*
④		true (1)	false (2)	1.0.62379.2.2.1.3.2.2.24.48000	false (2)
⑥		true (1)	false (2)	1.0.62379.2.2.1.3.2.2.24.96000	false (2)

It is the mode table that controls what formats the converter block should output and not *aConverterBlockTable*: *aConverterOutputFormat* is a read only field which shows what format is being output, chosen by the unit's own implementation rules based on which entries in the mode table are enabled. In this example, the analogue to digital converter can only convert from analogue to 24 bit, 48k audio, so there is only one possibility in the mode table. In the case of the sample rate converter, there are eight possible output formats but only one is set to enabled (24 bit, 96k) and so the unit is forced to output that format.

E.2.6 Summary of tables

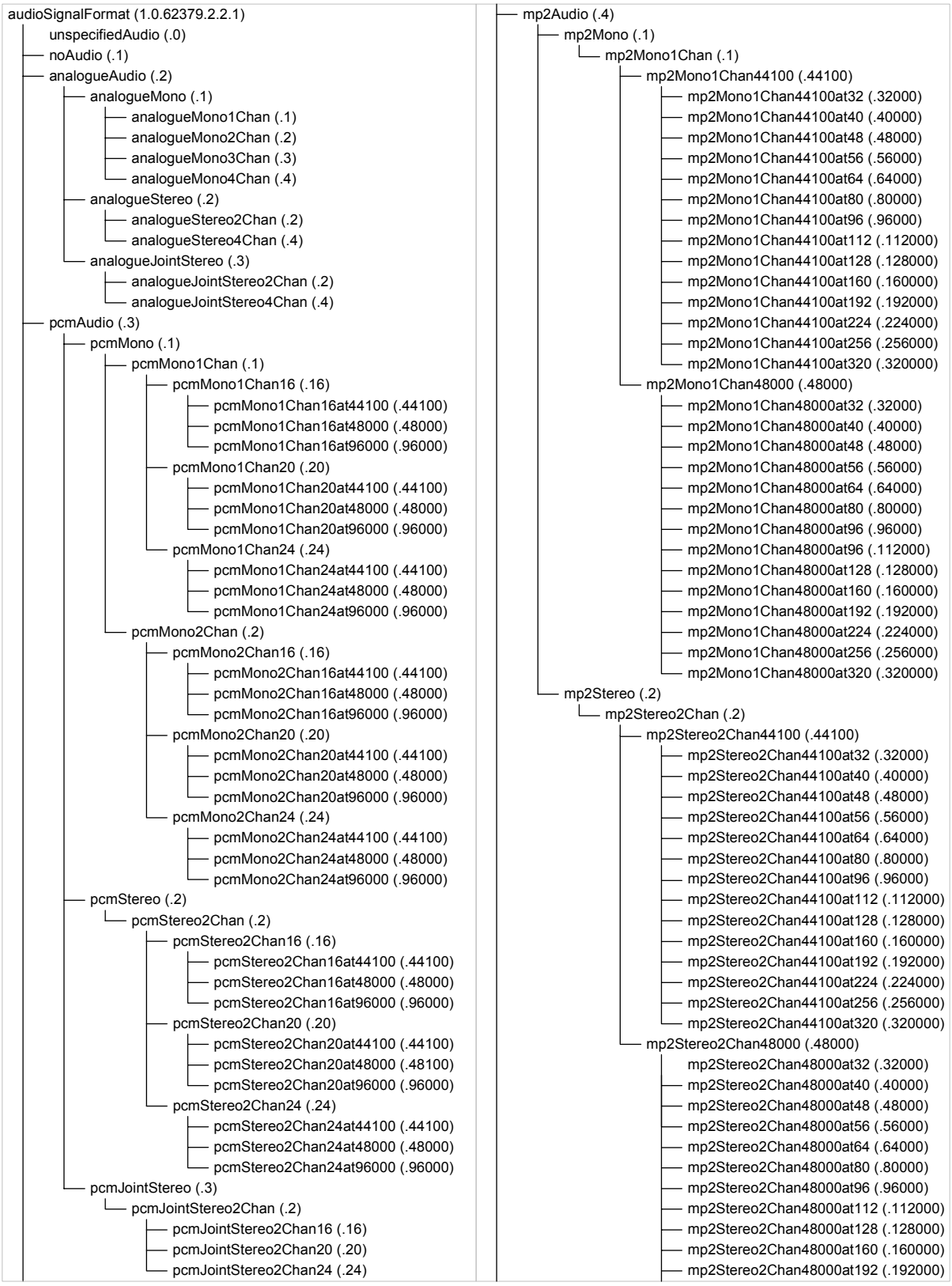


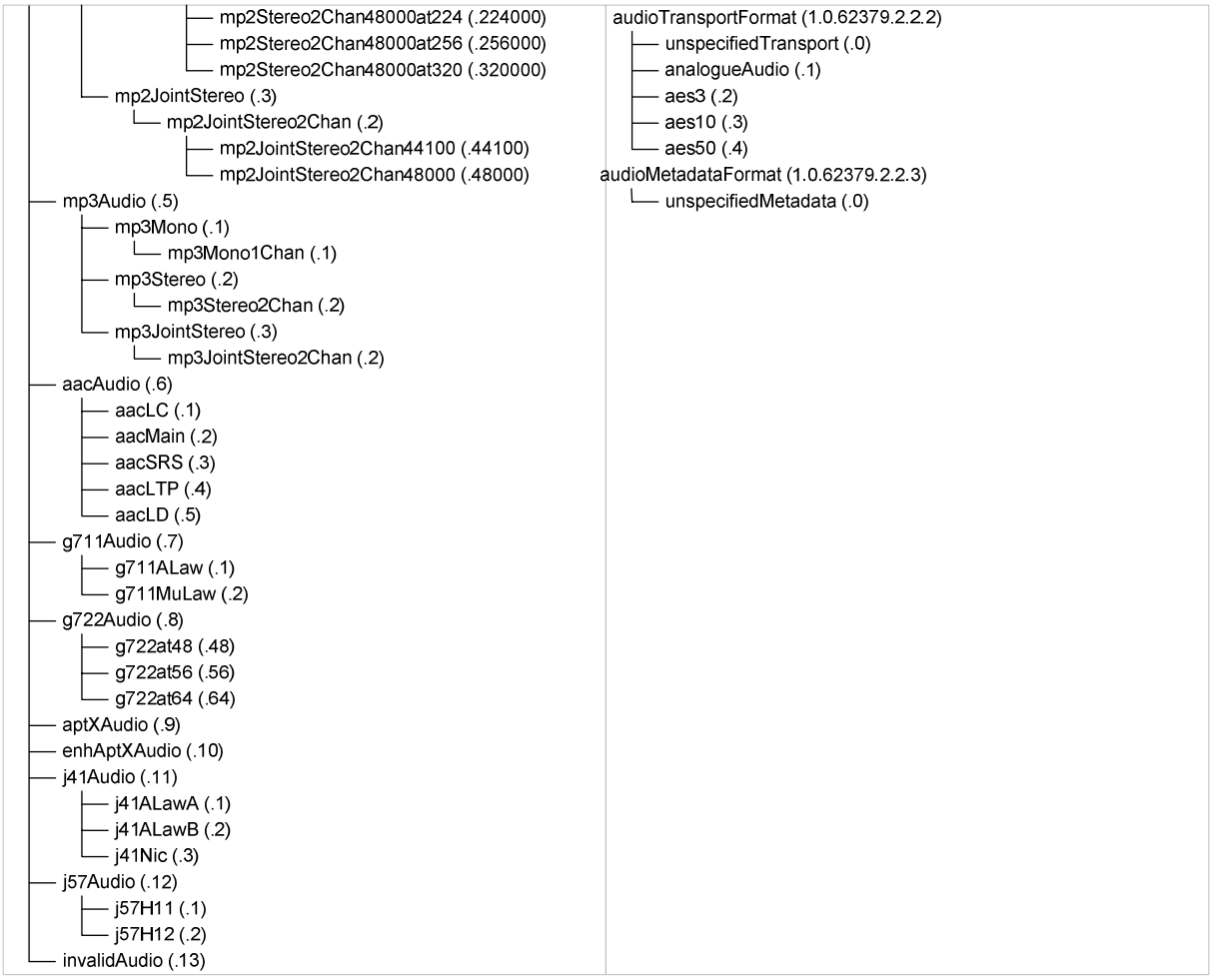
Annex F
(informative)

Tree of example audio formats

This annex shows a graphical tree of the example audio formats listed in Annex A. If there is any inconsistency between this annex and Clause 4, Clause 4 takes precedence.

NOTE 1 This annex is not intended to cover every format permitted by the definitions in Clause 4.





INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

3, rue de Varembé
PO Box 131
CH-1211 Geneva 20
Switzerland

Tel: + 41 22 919 02 11
Fax: + 41 22 919 03 00
info@iec.ch
www.iec.ch