INTERNATIONAL STANDARD

ISO/IEC 23008-3

First edition
2015-10-15

AMENDMENT 3
2017-01

Information technology — High efficiency coding and media delivery in heterogeneous environments —

Part 3: **3D audio**

AMENDMENT 3: MPEG-H 3D Audio Phase 2

Technologies de l'information — Codage à haute efficacité et livraison des medias dans des environnements hétérogènes —

Partie 3: Audio 3D

AMENDEMENT 3: Phase 2 de l'audio 3D MPEG-H

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Reference number
ISO/IEC 23008-3:2015/Amd.3:2017(E)

© ISO/IEC 2017

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2017, Published in Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Contents

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Foreword

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

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

Amendment 3 to ISO/IEC 23008-3:2015 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Introduction

The following text describes the Amendment 3 to the specification ISO/IEC 23008-3:2015 MPEG-H 3D Audio in an "Amendment"-style, i.e. in "Replace A with B"-style. It includes additions and changes that serve a number of purposes:

- improving the coding efficiency especially for low bitrate coding modes (for scene based as well as for object based and for multichannel based content)
- adding descriptive metadata
- updating the MHAS description
- some improvements for usage of MPEG-H 3D Audio in broadcasting applications
- a tool for Advanced Loudness Control
- a layered coding mode for coding of scene based content

It is envisioned that this amendment will be merged with the current version of the MPEG-H 3D Audio specification resulting in a second edition of the standard. Text with yellow highlight shall be adjusted by the editor of a new edition of ISO/IEC 23008-3.

For ease of review the document is structured by clauses, each of which reflect an approved set of changes.

New Clauses, Tables and Figures are typically labelled "AMDX.Y", where X is the number of the clause it appears in in this document and Y is an increasing integer counter.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Information technology — High efficiency coding and media delivery in heterogeneous environments — Part 3: 3D audio, AMENDMENT 3: MPEG-H 3D Audio Phase 2

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

1 Profiles and Levels

Add the following definition of profiles and levels to clause 4 Technical Overview:

4.X MPEG-H 3D Audio profiles and levels

4.X.1 Introduction

This subclause defines profiles and their levels for MPEG-H 3D Audio.

Complexity units are defined to give an approximation of the decoder complexity in terms of processing power required for the decoding process. The approximated processing power is given in "Processor Complexity Units" (PCU), specified in Millions Operations Per Second (MOPS).

4.X.2 Profiles

The following Audio Profiles are defined:

1. The Main Profile of MPEG-H 3D Audio provides a complete set of features for low-bitrate and high-quality coding, and rendering for all playback scenarios, exclusively based on the first edition of the MPEG-H 3D Audio specification ISO/IEC 23008-3:2015 3D Audio.

2. The High Profile of MPEG-H 3D Audio provides a complete set of features for low-bitrate and high-quality coding, and rendering for all playback scenarios.
   The High Profile is a superset of the Low-complexity Profile.

3. The Low Complexity Profile provides features for broadcasting and streaming with a reduced complexity of the decoder;

Table P1 — Summary of the Location of and Normative Reference to the Definitions of MPEG-H 3D Audio profiles. USAC and MPEG-H 3DA Main Profile are provided for information only

Tool / Module		defined in ISO/IEC	sub-clause	USAC 23003-3	MPEG-H 3DA Main Profile	MPEG-H 3DA High Profile	MPEG-H 3DA Low-Complexity Profile
block switching		14496-3	4.6.11	X	X	X	X
window shapes	AAC based	14496-3	4.6.11	X	X	X	X
	additional windows	23003-3	6.2.9.3	X	X	X	X
filter bank	AAC based	14496-3	4.6.11	X	X	X	X
	additional USAC	23003-3	7.9	X	X	X	X

TNS		14496-3	4.6.9	X	X	X	X
intensity		14496-3	4.6.8.2				
coupling		14496-3	4.6.8.3				
perceptual noise synthesis	PNS	14496-3	4.6.13				
	noise filling	23003-3	7.2	X	X	X	X
MS	basic mid/side coding	14496-3	4.6.8.1	X	X	X	X
	MDCT based complex prediction	23003-3	7.7.2	X	X	X	X
quantization	non-uniform	14496-3	4.6.1	X	X	X	X
	uniform	23003-3	7.1	X	X	X	X
entropy coding	Huffman	14496-3	4.6.3				
	context adaptive arithmetic coding	23003-3	7.4	X	X	X	X
SBR	base	14496-3	4.6.18	X	X	X	
	enhanced	23003-3	7.5	X	X	X	
parametric stereo extension	Parametric Stereo	14496-3	8.6.4 / 8.A				
	MPEG Surround 2-1-2 (incl. residual coding)	23003-3	6.2.13	X	X	X	
	Quad Channel Element	23008-3	5.5		X	X	
ACELP		23003-3	7.14	X	X	X	X
frequency domain noise shaping	scale factor based	14496-3	4.6.2	X	X	X	X
	LPC based	23003-3		X	X	X	X
Intelligent Gap Filling	IGF for FD	23008-3			X	X	X
Improved LPD coding	IGF for TCX and TBE in ACELP	23008-3 Amd3				X	X
	LPD stereo	23008-3 Amd3				X	X
Predictors for FD and TCX	frequency-domain prediction and time-domain post-filtering	23008-3 Amd3				X	X
Discrete Multi-channel coding	MCT	23008-3 Amd3				X	X
Format Converter	Generic downmix	23008-3	10, Amd3.1		X	X	X (Note 4)
Immersive Rendering	Immersive rendering within format converter	23008-3	11, Amd3.2		X	X	X (Note 4)
Static metadata	Metadata Audio Elements (MAE) and Audio Scene Information (ASI) Decoder and Renderer	23008-3	15		X	X	X
Dynamic object metadata	Object Audio Metadata (OAM) Decoder and Renderer	23008-3	7, 8		X	X	X
	MPEG Surround Extension	23003-1 Amd 3	9			X	
SAOC-3D	Decoder and Renderer	23008-3	9		X	X	
HOA	Decoder and Renderer	23008-3 and Amd3	12		X	X	X (Note 5)

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

	Near Field Compensation	23008-3			X	X	X (Note1)
	Subband Directional Prediction	23008-3 Amd3				X	
	Parametric Ambiance Replication (PAR)	23008-3 Amd3				X	
	Phase-based decorrelation	23008-3 Amd3				X	
Binaural	FD-binaural, TD-binaural	23008-3	13		X	X	X (Note2)
	HOA2Binaural H2B	23008-3			X	X	X (Note2)
DRC	DRC-1	23003-4			X	X	X (Note3)
	DRC-2 (single band)	23003-4			X	X	X
	DRC-2 (multi band)	23003-4					
	DRC-3 (single band)	23003-4			X	X	X
Sample Rate Converter		23008-3 Amd3	Amd3. 3			X	X
Peak Limiter	Unguided clipping prevention	23008-3 23003-4	D			X	X
Loudness	Loudness metadata and handling	23003-4	6		X	X	X
	Loudness compensation	23008-3 Amd3				X	X
MHAS	MPEG-H 3D audio stream	23008-3	14		X	X	X
	Truncation message and CRC packet type, ASI packet type	23008-3 Amd3				X	X
File Format	Carriage of MPEG-H 3D Audio in ISO base media file format	23008-3 Amd2			(Note 6)		
Interfaces and processing	Interfaces and processing for Interaction data and local setup info	23008-3	17,18		X	X	X
Carriage of system data	Carriage of System Data for the interaction with System Engine	23008-3 Amd4				X	X
TCC	Tonal Component Coding	23008-3 Amd3				X	
IC	Internal Channel	23008-3 Amd3				X	
HREP	High Resolution Envelope Processing	23008-3 Amd3				X	

Note 1: Restrictions apply dependent on the levels
Note 2: Implementation of binaural rendering is only mandated if headphone reproduction is supported.
Note 3: Multi-band DRC-1 shall be applied in the STFT domain of the TD format converter.
Note 4: The TD format converter downmix shall be applied for downmixing.
Note 5: In order to achieve target complexity for the LC profile at a given level implementers should study Annex G.
Note 6: File Format encapsulation is independent of the profile that is used for the bitstream. A profile level indicator is part of the file format specification (see XXX).

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd3:2017

4.X.2.1 Levels of the Low Complexity Profile

Table P2 — Levels and their corresponding restrictions for the Low Complexity Profile

Level	Max. Sampling rate	Max. no. of core ch. in compressed data stream	Max. no. of decoder processed core ch.	Max. no. of loudspeaker output ch.	Example of max. loudspeaker configuration	Max. no. of decoded objects	Example of a max. Config C+O	Max. HOA order	Example of max. HOA order + O
1	48000	10	5	2	2.0	5	2 ch. + 3 static obj. NOTE 1	2	2nd order + 3 static obj. NOTE 1
2	48000	18	9	8	7.1	9	6 ch. + 3 static obj. NOTE 1	4	4th order + 3 static obj. NOTE 1
3	48000	32	16	12	11.1	16	12 ch. + 4 obj.	6	6th order + 4 obj.
4	48000	56	28	24	22.2	28	24 ch. + 4 obj.	6	6th order + 4 obj.
5	96000	56	28	24	22.2	28	24 ch. + 4 obj.	6	6th order + 4 obj.
NOTE 1 – In this context "static objects" are understood as channel-based signals without accompanying OAM data which are not also associated to a channel bed									

— The use of switch groups determines the subset of core channels out of the core channels in the bitstream that shall be decoded.

— If the mae_AudioSceneInfo() contains switch groups (mae_numSwitchGroups>0), then the elementLengthPresent flag shall be 1

— The number of channels of the signaled referenceLayout shall not exceed the maximum number of loudspeaker output channels as defined in the levels Table P2

Table P3 — Approximated worst case processing power (PCU) of decoder modules and the whole decoder for the different Levels of the Low Complexity Profile given in MOPS

Level	Core LC	Format Converter	Object Renderer	HOA[2]	Objects only Renderer	DRC	Limiter	Binaural[1]	Worst case PCU
1	33	3	0	3	9	6	4	7	58
2	59	10	0	17	16	18	5	19	118
3	106	36	7	36	29	24	6	27	206
4	186	113	7	93	50	30	9	46	392
5	373	226	14	186	50	34	19	92	758
[1] NOTE: The complexity numbers for binaural processing are calculated on the basis of BRIR filters of 1 second length measured in a BS.1116 compliant room. [2] NOTE: The complexity numbers for the HOA spatial decoding and rendering are based on the Low									

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Complexity Combined HOA Spatial Decoding and Rendering described in Annex G.

4.X.2.2 Restrictions for the Low Complexity Profile and Levels

In the low complexity profile the core decoder, format converter, object renderer, HOA renderer and DRC and peak limiter operate in time domain, MDCT-domain or STFT-domain.

The following restrictions apply for HOA renderer and decoder:

Table P4 — Restrictions for the HOA spatial Decoding and Rendering according to the Level of the Low Complexity Profile

Restriction applies to	**Maximum allowed value depending on Mpegh3daProfileLevelIndication**				
	Lvl 1	**Lvl 2**	**Lvl 3**	**Lvl 4**	**Lvl 5**
HOA order (max)	2	4	6	6	6
Number of Predominant Sounds (max)	3	5	7	8	8
Number of directional signals used in prediction (max)	2	3	4	5	5
The Near Field Compensation (NFC) processing may be applied to HOA content of an order which is smaller or equal to:	N/A (NFC not allowed)	1	2	3	3

NFC may be employed in not more than one signal group of type SignalGroupTypeHOA.

The following restrictions apply to MPEG-D DRC (ISO/IEC 23003-4) when employed as part of MPEG-H 3D audio:

— drcFrameSizePresent and timeDeltaMinPresent shall be set to 0.

— gainInterpolationType shall be set to 1.

— dependsOnDrcSetPresent shall be set to 0 for drcInstructionsUniDrc() with downmixId == 0.

— HOA signal groups shall be restricted to one drcChannelGroup and DRC gains shall be applied to the HOA core channels (HOATransportChannels).

— The values of bsSequenceIndex within drcInstructionsUniDrc() shall be unique in simultaneously applied DRC sets except for bsSequenceIndex == 0.

— Multiband DRC shall be restricted to drcInstructionsUniDrc() with downmixId == 0. If the bitstream should contain multiband DRC, the number of multiband DRC core channels shall be restricted as follows:

(numAudioChannels +
numAudioObjects + numAudioObjectsMB +
numHOATransportChannels + numHOATransportChannelsMB)
≤ (numCoreChannelsMax(Lvl) – dependsOnDrcSetPresentFlag – 1)

, where:

— numAudioChannels, numAudioObjects and numHOATransportChannels are the number of C, O and HOA core channels as specified in Table 4.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

  - numAudioObjectsMB and numHOATransportChannelsMB are the number of O and HOA core channels out of numAudioObjects and numHOATransportChannels that contain multiband DRC
  - numCoreChannelsMax is the maximum number of core channels ("No of Core ch") depending on the Mpegh3daProfileLevelIndication field as defined in Table P2
  - dependsOnDrcSetPresentFlag is set to one if the bitstream contains any configuration with dependsOnDrcSetPresent==1 (otherwise zero).

— nNodes shall be restricted to a maximum value of 32, where nNodes is the number of encoded gain values in the current DRC frame.

Table P5 — Restrictions applying to DRC processing according to the Levels of the Low Complexity Profile

Restriction applies to	Maximum allowed value depending on Mpegh3daProfileLevelIndication				
	Lvl 1	Lvl 2	Lvl 3	Lvl 4	Lvl 5
nDrcChannelGroupsTotal (Note 1)	5	9	16	28	28
drcCoefficientsUniDrcCount	4	4	4	4	4
bandCount (Note 2)	2	4	4	4	4
sequenceCountTotal (Note 3)	24	28	32	48	63
drcInstructionsUniDrcCount	16	16	32	32	32
Note 1: Maximum allowed number of simultaneously active DRC channel groups in all applied DRC sets. Note 2: Maximum allowed number of DRC bands for multiband DRC. Note 3: Sum of all nDrcBands in drcGainSequence() structures plus number of sequences with gainCodingProfile=3.					

The following tool specific restrictions apply:

— If the independent noise filling (INF) of the intelligent gap filling (IGF) is activated (i.e. if igfUseEnf==1), then the Complex Prediction tool shall be restricted to real-only prediction, i.e. complex_coef shall be 0.

— If Stereo Filling is activated (i.e. if stereo_filling==1), then the Complex Prediction tool shall be restricted to real-only prediction, i.e. complex_coef shall be 0.

— The independent noise filling of the intelligent gap filling shall not be employed in cases where igfBgn corresponds to an audio frequency higher than 8 kHz.

— The LPD mode shall only be employed at 3DA core coder sampling rates (as defined in Table 2 — Syntax of mpegh3daConfig()) ≤ 32000 Hz

EXAMPLE For a 48 kHz input signal, the encoder resamples the signal to a 32 kHz core coder sampling rate and the LPD decoder operates at this lower sampling rate. After the core decoding the signal is resampled to 48 kHz.

— The multi-channel coding tool (MCT) shall not employ more stereo boxes than specified in Table P9

Table P9 — Restrictions applying to MCT processing according to the Levels of the Low Complexity Profile

Restriction applies to	Maximum allowed value depending on Mpegh3daProfileLevelIndication				
	Lvl 1	Lvl 2	Lvl 3	Lvl 4	Lvl 5
Number of stereo boxes in MCT	5	9	16	28	28

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

The following restrictions apply to coding of audio objects and the associated OAM data:

Table P10 — Restrictions applying to Object Processing According to the Levels of the Low Complexity Profile

Restriction applies to	Maximum allowed value n depending on Mpegh3daProfileLevelIndication				
	Lvl 1	Lvl 2	Lvl 3	Lvl 4	Lvl 5
(number of objects without divergence) + 3·(number of objects with divergence > 0) ≤ n	5	9	16	28	28

— Efficient Object Metadata Decoding is not permitted, i.e. lowDelayMetadataCoding shall be 1.

— Furthermore the OAM frame length shall comply to:

OAMFrameLength = outputFrameLength / n,
with n being a positive integer in the range of $\{1,\ldots,4\}$

— Objects shall not employ divergence and spread at the same time.

— If an object is defined with a spatial extent (spread $\alpha > 0.0°$ for uniform spread, spread_width $\alpha_{width} > 0.0°$ for non-uniform spread) it shall have a divergence value equal to zero.

— If an object is defined with a divergence value > 0, it shall not have a spatial extent (spread α shall be equal to 0.0° for uniform spread, spread_width α_{width} shall be equal to 0.0° for non-uniform spread)

The following restrictions apply to binaural rendering:

The value of bsBinauralDataFormatID in BinauralRendering() should be set to 1 (if the FD Binaural renderer is implemented) or to 2 (if the TD Binaural renderer is implemented). The value of bsBinauralDataFormatID can be set to 0 if the Parameterization of Binaural Room Impulse Responses according to 13.2.3 or 13.3.3 is implemented.

The number of BRIR sets is restricted to a maximum number of 3.

In case of H2B filters, the number of BRIR filter pairs to be provided shall correspond to 'Maximum H2B filter order' column in Table P8. In the other cases, the following applies:

The number of BRIR pairs in each BRIR set shall correspond to the number indicated in the relevant level-dependent row of Table P8. The measured BRIR positions shall correspond to all nominal geometric positions corresponding to the list of LoudspeakerGeometry indices in Table P8. The correspondence between LoudspeakerGeometry index and nominal geometric position is defined in ISO/IEC 23001-8. Thereby, it is ensured that one BRIR pair is available for each possible regular input channel configuration that can be used within the indicated level.

An input channel configuration is regular if it is defined by means of an ISO/IEC 23001-8 ChannelConfiguration or a list of ISO/IEC 23001-8 LoudspeakerGeometry (CICPspeakerIdx).

If binaural rendering is activated, the measured BRIR positions shall be passed to the mpegh3daLocalSetupInformation(). Thus, all renderer stages are set to the target layout that is equal to the transmitted channel configuration. As one BRIR is available per regular input channel, the Format Converter can be passed through in case regular input channel positions are used.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table P8 — The binaural restrictions for the LC profile

Level	Number of BRIR pairs	Maximum H2B filter order	BRIR positions by means of Loudspeaker Position Abbreviation	BRIR positions by means of LoudspeakerGeometry according to ISO/IEC 23001-8
1	3	1	L, R, C	0, 1, 2
2	10	2	L, R, C, LS, Rs, Lc, Rc, Lsr, Rsr, Cs	0, 1, 2, 4, 5, 6, 7, 8, 9, 10,
3	21	3	L, R, C, Ls, Rs, Lc, Rc, Lsr, Rsr, Cs, Lss, Rss, Lv, Rc, Cv, Lvr, Rvr, Cvr, Rs, Lvs, Rvs	0, 1, 2, 4, 5, 6, 7, 8, 9, 10, 13, 14, 17, 18, 19, 20, 21, 22, 25, 30, 31
4	28	5	L, R, C, Ls, Rs, Lc, Rc, Lsr, Rsr, Cs, Lss, Rss, Lv, Rc, Cv, Lvr, Rvr, Cvr, Lvss, Rvss, Ts, Lb, Rb, Cb, Lvs, Rvs, Lbs, Rbs,	0, 1, 2, 4, 5, 6, 7, 8, 9, 10, 13, 14, 17, 18, 19, 20, 21, 22, 23, 24, 25, 27, 28, 29, 30, 31, 37, 38,
5	28	6	L, R, C, Ls, Rs, Lc, Rc, Lsr, Rsr, Cs, Lss, Rss, Lv, Rc, Cv, Lvr, Rvr, Cvr, Lvss, Rvss, Ts, Lb, Rb, Cb, Lvs, Rvs, Lbs, Rbs,	0, 1, 2, 4, 5, 6, 7, 8, 9, 10, 13, 14, 17, 18, 19, 20, 21, 22, 23, 24, 25, 27, 28, 29, 30, 31, 37, 38,

The following additional parameter values restrictions apply:

— The value of kMax in FdBinauralRendererParam() shall be equal to or less than 48 (bands).

— The value of kConv in FdBinauralRendererParam() shall be equal to 32.

— The values of rt60[k] in SfrBrirParam() shall be less than or equal to 1.0 (sec).

— The average of the values of nFilter[k] shall be less than or equal to 64.

— The values of nFilter[k] in VoffBrirParam() should be less than or equal to 256.

The following coding tools, modules, or features shall not be employed

— Time warped filterbank

— 768 sample outputFrameLength, i.e. coreSbrFrameLengthIndex shall not be 0

The following text describes restrictions dependent on the length of the arithmetic coder codeword, arith_data(). For this text the following definitions apply:

F_{sOut}	core coder sampling rate as indicated by means of usacSamplingFrequencyIndex or usacSamplingFrequency in mpegh3daConfig()
F_{sMax}	maximum allowed sampling rate of a given level in this profile
N_{chMax}	maximum number of decoder processed core channels of a given level in this profile according to Table P2.
N_{chLtpf}	number of core coder channels in which the long term post filter (LTPF) is applied
N_{chInf}	number of core coder channels in which the independent noise filling (INF) is applied

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

$Nbits_{arith\_data()}(ch)$ number of bits used for arithmetic coding of spectral data, arith_data(), for core coder channel ch for a given frame

$Nbits_{arith\_all} = \sum_{\text{all channels}} Nbits_{\text{arith\_data()}}(ch)$, i.e. the sum of all bits used for the arithmetic coding of spectral data of all core coder channel

— In any given audio frame $Nbits_{arith\_all}$ shall comply with the following restriction:

$$Nbits_{arith\_all} < \frac{(3072 \cdot N_{chMax} - 2048 \cdot N_{chLtpf} - 2048 \cdot N_{chInf}) \cdot F_{\text{sMax}}}{F_{\text{sOut}}}$$

The following restrictions apply to the AudioPreRoll() extension:

— Decoders conforming to this profile shall support the full decoding and correct handling of the AudioPreRoll() extension .

— The number of pre-roll frames, numPreRollFrames, in an AudioPreRoll() extension payload shall not exceed 1 (one).

— In access units that are embedded as pre-roll in an AudioPreRoll() extension the usacExtElementPresent field for extensions of type ID_EXT_ELE_AUDIOPREROLL shall be 0.

The following restrictions apply to the employed sampling rate and the resampler block:

— The sampling rate that is signaled by means of usacSamplingFrequencyIndex or usacSamplingFrequency shall be one of the values in the first column of Table P6.

— Depending on the above mentioned sampling rate and the profile level the resampler may employ one of the resampling ratios indicated in Table P6.

Table P6 — Allowed Sampling Rates and Resampling Ratios

Allowed sampling rate	Allowed resampling ratio depending on Mpegh3daProfileLevelIndication				
	Lvl 1	Lvl 2	Lvl 3	Lvl 4	Lvl 5
96000	N/A	N/A	N/A	N/A	1
88200	N/A	N/A	N/A	N/A	1
64000	N/A	N/A	N/A	N/A	1.5
58800	N/A	N/A	N/A	N/A	1.5
48000	1	1	1	1	1 or 2
44100	1	1	1	1	1 or 2
32000	1.5	1.5	1.5	1.5	1.5 or 3
29400	1.5	1.5	1.5	1.5	1.5 or 3
24000	2	2	2	2	2
22050	2	2	2	2	2
16000	3	3	3	3	3
14700	3	3	3	3	3

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

The following restrictions apply to the coding of the audio scene information structure:

Table P7 — ProfileLevel dependent restrictions to selected fields of the mae_AudioSceneInfo()

Restriction applies to	Allowed maximum value depending on Mpegh3daProfileLevelIndication				
	Lvl 1	Lvl 2	Lvl 3	Lvl 4	Lvl 5
mae_numGroups	5	9	16	28	28
mae_numSwitchGroups	2	4	8	14	14
mae_numGroupPresets	4	4	8	16	31
mae_bsGroupPresetNumConditions	5	9	16	16	16
mae_numDownmixIdGroupPresetExtensions per mae_groupPresetID	4	4	8	16	31
mae_bsNumDescLanguages	4	4	4	8	16
mae_bsDescriptionDataLength	256	256	256	256	256

— If mae_numSwitchGroups > 0, then elementLengthPresent shall be set to 1.

4.X.2.3 Levels of the Main Profile

Currently blank — Placeholder for Main Profile

4.X.2.4 Levels of the High Profile

Currently blank — Placeholder for High Profile

In 5.3.2 replace Table 39 — Value of mpegh3daProfileLevelIndication

Table 39 — Value of mpegh3daProfileLevelIndication

value	Indication of Profile	Indication of Level
0x00	reserved for ISO use	-
0x01-0xff	reserved for future profile definition	

with:

Table 39 — Value of mpegh3daProfileLevelIndication

value	Indication of Profile	Indication of Level
0x00	reserved for ISO use	-
0x01	Main Profile	L1
0x02	Main Profile	L2
0x03	Main Profile	L3
0x04	Main Profile	L4
0x05	Main Profile	L5
0x06	High Profile	L1

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

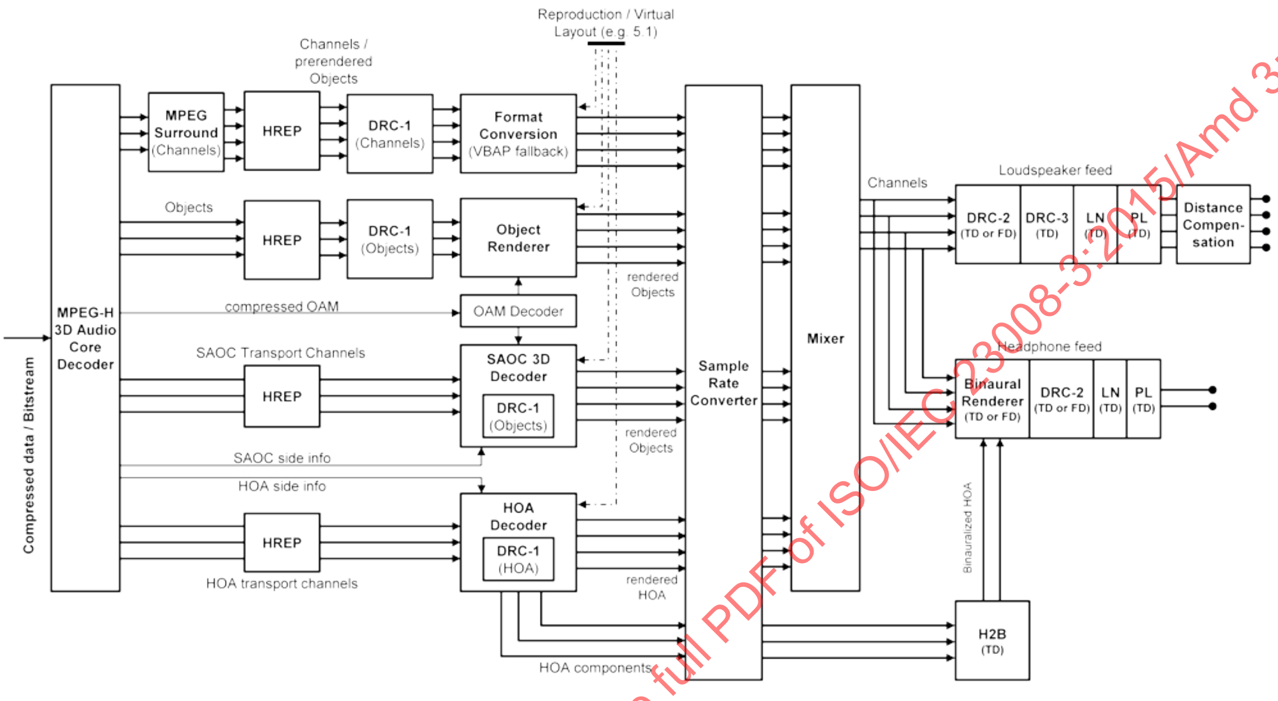
0x07	High Profile	L2
0x08	High Profile	L3
0x09	High Profile	L4
0x0A	High Profile	L5
0x0B	Low Complexity Profile	L1
0x0C	Low Complexity Profile	L2
0x0D	Low Complexity Profile	L3
0x0E	Low Complexity Profile	L4
0x0F	Low Complexity Profile	L5
0x10-0xFF	reserved for future profile definition	

Please also study Clause 31 of this Amendment document

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

2 Technical Overview - Update

Replace Figure 1 — Block diagram of the 3D-Audio decoder. (DRC: Dynamic Range Control, SAOC: Spatial Audio Object Coding, HOA: Higher Order Ambisonics, LN: Loudness Normalization, PL: Peak Limiter) with:



Add the following descriptive text after the description of the "HOA Decoder and Renderer" in clause 4.2:

The MPEG Surround Decoder takes the downmix signals coming from the MPEG-H 3D Audio decoder and performs the guided MPEG Surround upmix using the MPEG Surround side information to reproduce the multichannel signal for the transmitted loudspeaker layout.

In Table 1 — MPEG-H 3DA functional blocks and internal processing domain replace:

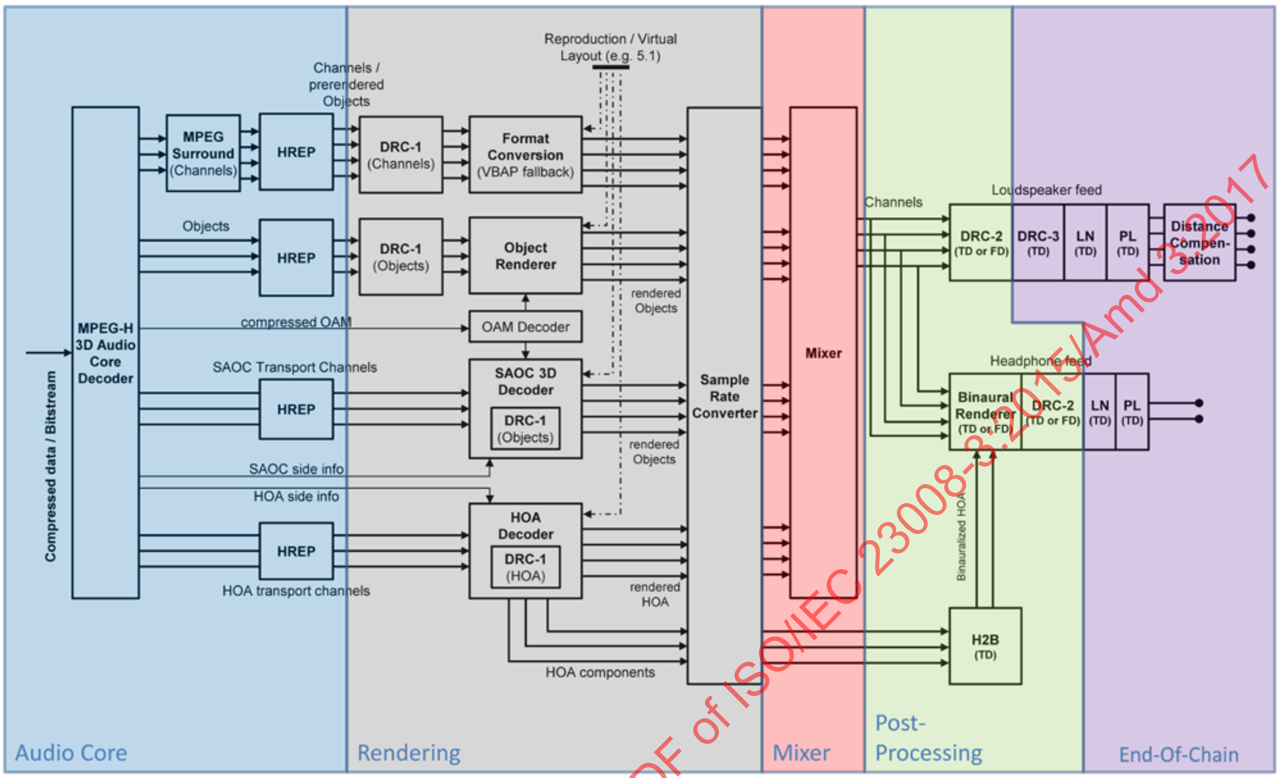
Audio Core	MPEG-H 3D Audio Core Coder	FD or TD[1)]

with:

Audio Core	MPEG-H 3D Audio Core Coder	FD or TD[1)]
	MPEG Surround	FD

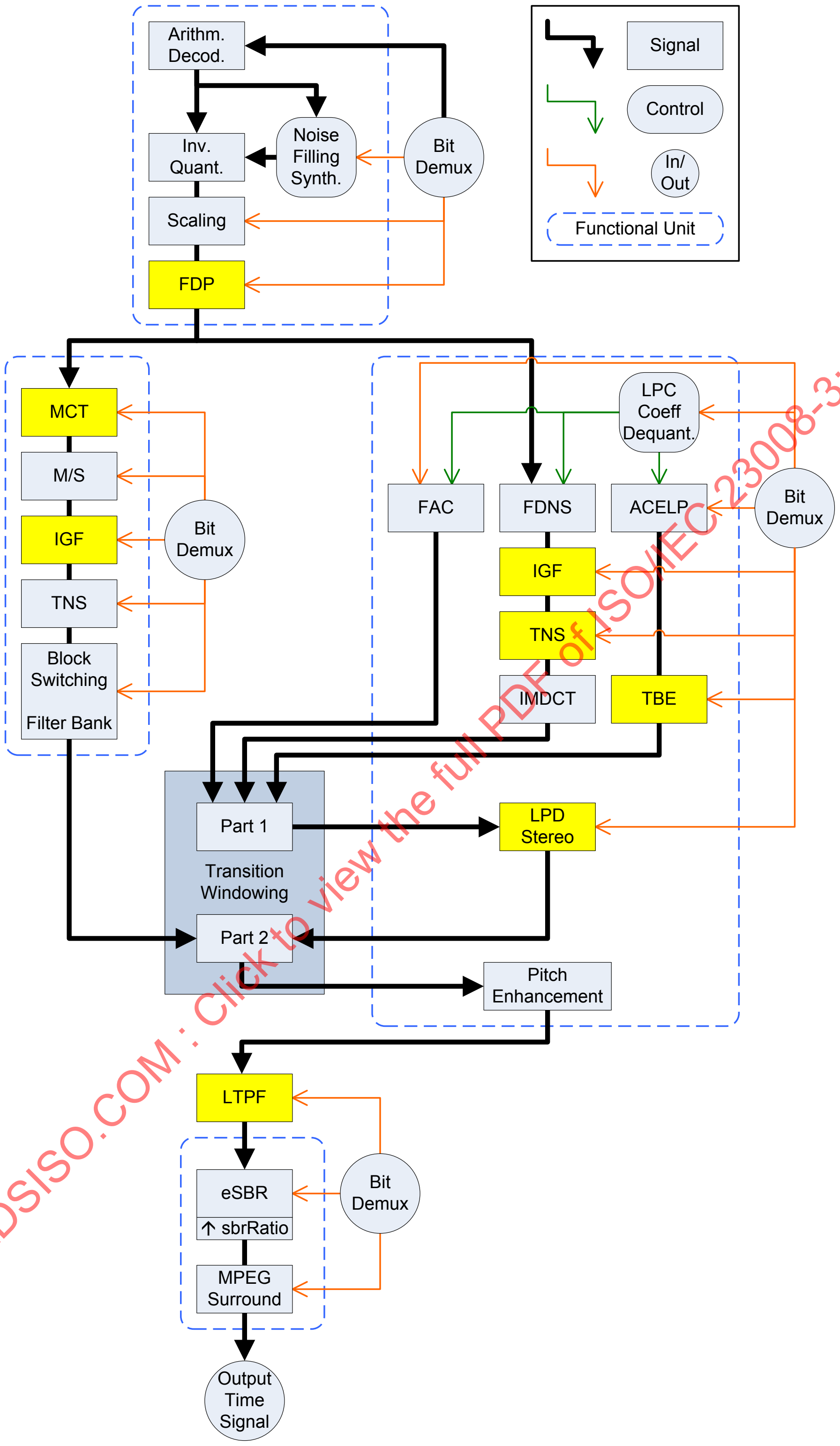
STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Replace Figure 2 — MPEG-H 3D audio decoder overview with signal processing context with:



STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In clause "4.2 Overview over the codec building blocks" after the first paragraph add following text and figure:



STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Figure AMD3.XX — Simplified block diagram of the typical MPEG-H core decoder configuration

Figure AMD3.XX shows a simplified block diagram of the typical MPEG-H core decoder building blocks. The major differences compared to MPEG-D USAC technology are highlighted in yellow. The highlighted tools are described in detail in section "MPEG-H 3D Audio Core decoder".

© ISO/IEC 2017 – All rights reserved

3 MPEG Surround

Add the following new clause:

x MPEG Surround

x.1 Technical Overview

The output of the 3D Audio core decoder of the "channels / prerendered objects" path may be further processed by MPEG Surround (MPS). Figure AMD1.1 shows a schematic of a combined 3D Audio core decoder and a MPS decoder.

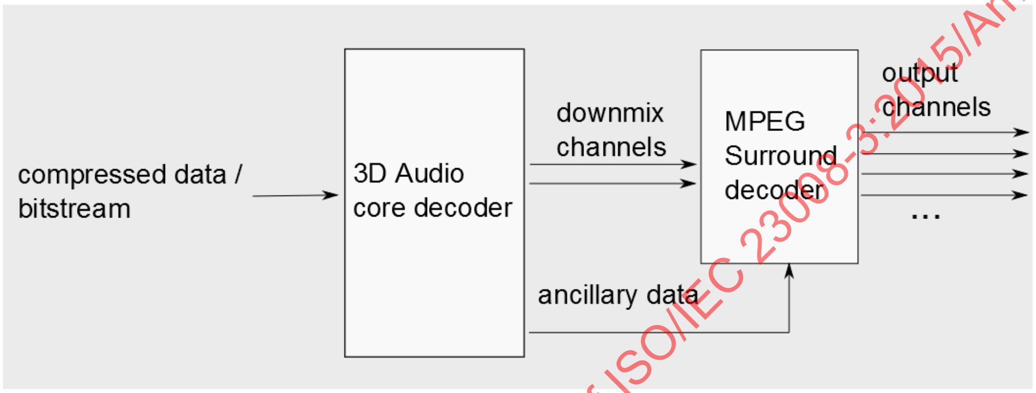


Figure AMD1.1: Block Diagram of a 3D Audio Core Decoder with an MPEG Surround decoder

If the SBR tool is active, a 3D Audio core decoder can typically be efficiently combined with a subsequent MPS decoder by connecting them in the QMF domain in the same way as it is described for USAC in clause 4.3 of ISO/IEC 23003-1:2012, *MPEG-D (MPEG audio technologies), Part 3: Unified Speech and Audio Coding, 2012.* . If a connection in the QMF domain is not possible, the tools need to be connected in the time domain.

If MPS side information is embedded into a 3D Audio bitstream by means of the usacExtElement mechanism (with usacExtElementType being ID_EXT_ELE_MPEGS), the time-alignment between the 3D Audio data and the MPS data assumes the most efficient connection between the 3D Audio core decoder and the MPS decoder. If the SBR tool in the 3D Audio core decoder is active and if MPS employs a 64-band QMF domain representation (see 6.6.3 in ISO/IEC 23003-1:2007, *Information technology — MPEG audio technologies — Part 1: MPEG Surround*), the most efficient connection is in the QMF domain. Otherwise, the most efficient connection is in the time domain. This corresponds to the time-alignment for the combination of HE-AAC and MPS as defined in 4.4, 4.5, and 7.2.1 of ISO/IEC 23003-1:2007, *Information technology — MPEG audio technologies — Part 1: MPEG Surround*.

The additional delay introduced by adding MPS decoding after 3D Audio decoding is given by clause 4.5 of ISO/IEC 23003-1:2007, *Information technology — MPEG audio technologies — Part 1: MPEG Surround* and depends on whether HQ MPS or LP MPS is used, and whether MPS is connected to the 3D Audio core decoder in the QMF domain or in the time domain.

If multiple signal groups of type SignalGroupTypeChannels are present in the bitstream, one extension element conveying MPEG Surround data shall only refer to exactly only signal group. As per 5.3.1, the corresponding channel elements shall directly follow that extension element.

If the MPEG Surround tool shall be used for one signal group of type SignalGroupTypeChannels, all core coder channels belonging to that signal group shall be fed to the MPEG Surround tool.

x.2 Syntax and Data Structure

The bitstream syntax and data structure are identical to the definitions in ISO/IEC 23003-1:2007, *Information technology — MPEG audio technologies — Part 1: MPEG Surround* and ISO/IEC 23003-1:2007/Amd.3:201x, *MPEG Surround extension for 3D Audio*.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

x.3 Tool Description

The processing of the MPEG Surround tools are fully specified in ISO/IEC 23003-1:2007, *Information technology — MPEG audio technologies — Part 1: MPEG Surround* and ISO/IEC 23003-1:2007/Amd.3:201x, *MPEG Surround extension for 3D Audio*

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

4 3D Audio Phase II – HOA (Subband Directional Prediction, Parametric Ambiance Replication, Phase-based Decorrelation, HOA Layered Coding)

The following changes are related to HOA Matrix Encoder / Decoder

In Table 23 — Syntax of HOARenderingMatrix() replace:

precisionLevel	**2**	**uimsbf**
if (**gainLimitPerHoaOrder**) {	**1**	**uimsbf**

with:

maxHoaOrder = sqrt(NumOfHoaCoeffs)-1;		
precisionLevel	**2**	**uimsbf**
isNormalized	**1**	**uimsbf**
if (**gainLimitPerHoaOrder**) {	**1**	**uimsbf**

In the same Table, replace

if (**isFullMatrix**) {	**1**	**uimsbf**
firstSparseOrder	**1**	**uimsbf**

with:

if (**isFullMatrix**==0) {	**1**	**uimsbf**
nbitsHoaOrder = ceil(log2(maxHoaOrder+1));		
firstSparseOrder	**nbitsHoaOrder**	**uimsbf**

In the same Table, replace

} else { /* isAnyValueSymmetric==0 */		
if { **isAnySignSymmetric**) {	**1**	**uimsbf**
for (i=0; i<numPairs; ++i)		
signSymmetricPairs[i] = **boolVal**	**1**	**uimsbf**

with:

} else {		
if (**isAnySignSymmetric**) {	**1**	**uimsbf**
for (i=0; i<numPairs; ++i)		
signSymmetricPairs[i] = **boolVal;**	**1**	**uimsbf**

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In Table 25 — Syntax of DecodeHoaMatrixData() replace:

```
    for (i = 0; i < inputCount; ++i) {
        for (j = 0; j < outputCount; ++j) {
            hoaMatrix[i *outputCount+j] *= signMatrix[i*outputCount+j];
            hoaMatrix[i *outputCount+j] /= sqrt(2 × ceil(sqrt(i+1)-1) + 1);
        }
    }
```

with:

```
    for (i = 0; i < inputCount; ++i) {
        for (j = 0; j < outputCount; ++j) {
            hoaMatrix[i *outputCount+j] *= signMatrix[i*outputCount+j];
            hoaMatrix[i *outputCount+j] /= sqrt(2 × ceil(sqrt(i+1)-1) + 1);
        }
    }
if(isNormalized) {
    currentScalar = 0.0;
    for (i = 0; i < inputCount; ++i) {
        for (j = 0; j < outputCount; ++j) {
            if (!outputConfig[j].isLFE)
                currentScalar += hoaMatrix[i * outputCount + j] *
                                hoaMatrix[i * outputCount + j];
        }
    }
    currentScalar = 1.0/sqrt(currentScalar);
    for (i = 0; i < inputCount; ++i) {
        for (j = 0; j < outputCount; ++j) {
            if(!outputConfig[j].isLFE)
                hoaMatrix[i * outputCount + j] *= currentScalar;
        }
    }
```

In subclause 5.3.6 HOA Rendering Matrix Data Elements add before precisionLevel:

maxHoaOrder	HOA Order of the transmitted matrix.
isNormalized	Indicates if the HOA rendering matrix $\boldsymbol{D}$ is energy normalized, so that $\|\| \boldsymbol{D}\|\|_f = \sum_{l=1}^{L} \sum_{n=1}^{(N+1)^2} D_{l,n}^2 = 1$ with l being the non-LFE loudspeakers in the outputConfig.

In the same subclause before firstSparseOrder:

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

nbitsHoaOrder	Number of bits reading firstSparseOrder.

In Table 24 5.4.3.3 Decoding of HOA Rendering Matrix Coefficients after:

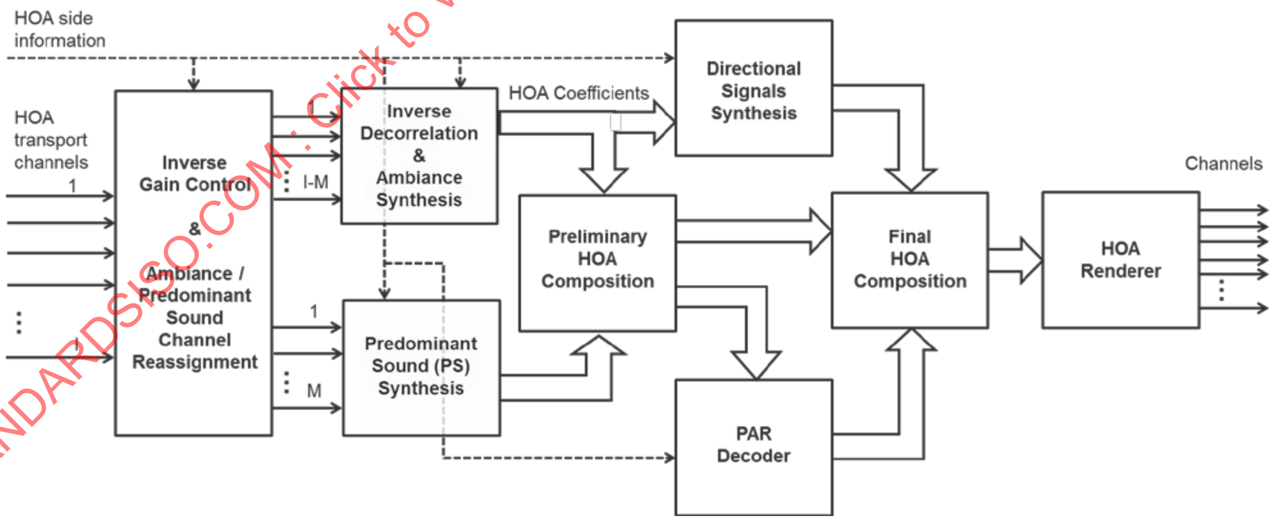
In this case the code words to decode the individual matrix elements for the left loudspeaker are reduced or completely omitted accordingly.

Add:

If the bitfield *isNormalized* was set to 1 the final HOA rendering matrix $\boldsymbol{D}$ is created by dividing each weighting value in the L rows of the HOA rendering matrix that are associated with non-LFE loudspeakers by the matrix's Frobenius Norm $\sum_{l=1}^{L}\sum_{n=1}^{(N+1)^2} D_{l,n}^2$ computed from its L rows associated with non-LFE loudspeakers.

The following changes are related to the chapter 12 Higher Order Ambisonics (HOA).

Replace Figure 33 — Simplified block diagram of the decoder with:



HOA side information
HOA transport channels
1
Inverse Gain Control & Ambiance / Predominant Sound Channel Reassignment
I-M
1
M
Inverse Decorrelation & Ambiance Synthesis
Predominant Sound (PS) Synthesis
HOA Coefficients
Directional Signals Synthesis
Preliminary HOA Composition
PAR Decoder
Final HOA Composition
HOA Renderer
Channels

In subclause 12.1.1 Block Diagram replace:

The fixed subset of the $(I - M)$ HOA coefficient signals is re-correlated, this means the decorrelation at the HOA encoding stage is reversed. Next, all of the $(I - M)$ HOA coefficient signals are used to create the ambient HOA components. The predominant HOA components are synthesized from the M predominant

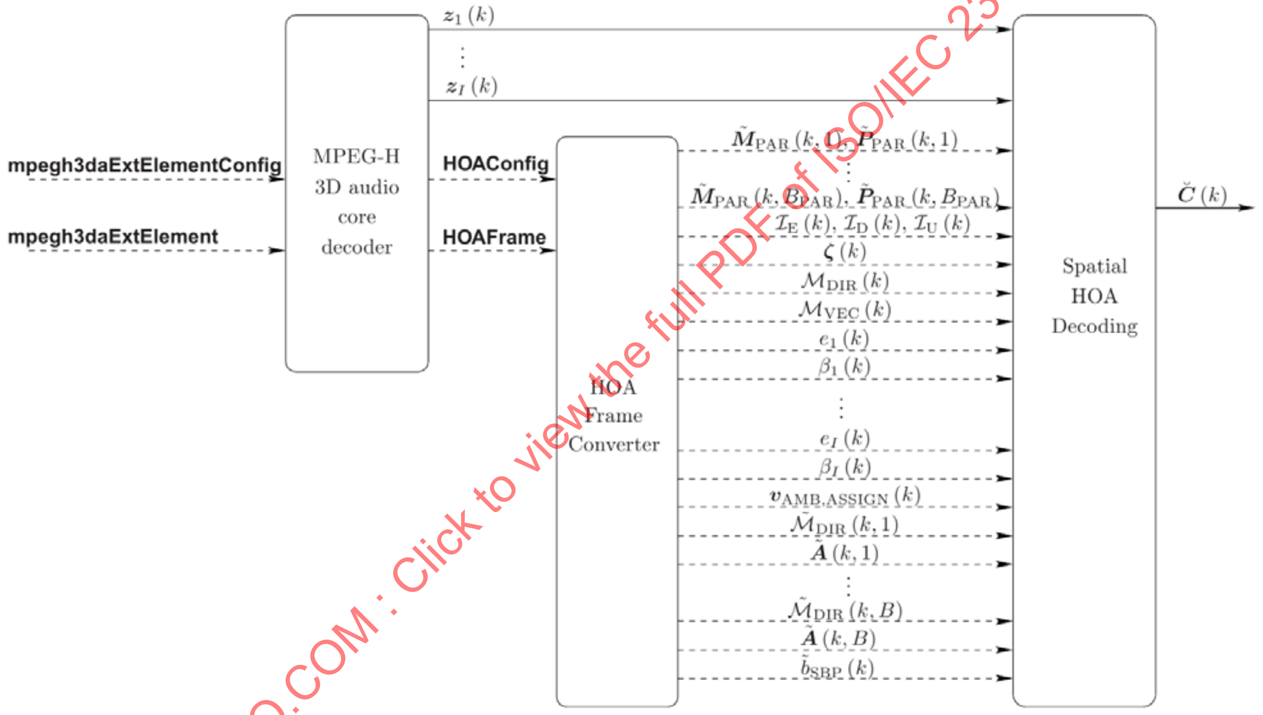
STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

signals and the corresponding parameters. Finally, the predominant and the ambient HOA components are composed into the desired full HOA representation, which is then rendered to a given loudspeaker setup.

with:

The fixed subset of the $(I - M)$ HOA coefficient signals is re-correlated, this means the decorrelation at the HOA encoding stage is reversed. Next, all of the $(I - M)$ HOA coefficient signals are used to create the ambient HOA component. The ambient HOA component is input to the Directional Signals Synthesis and the Preliminary HOA Composition. The Directional Signals Synthesis creates a new HOA representation from the ambient HOA component by prediction. The predominant HOA component is synthesized from the M predominant sound signals and the corresponding parameters. The predominant and the ambient HOA component are composed into the preliminary HOA representation, which is then input to the Parametric Ambience Replication (PAR). The PAR adds missing ambience components to the preliminary HOA representation, which is parametrically created from its input signals. Finally the output of the PAR, the preliminary HOA representation and the output of the Directional Signals Synthesis are composed to the decoded HOA representation that is rendered to the loudspeaker setup by the HOA Renderer.

Replace Figure 34 — The architecture of the HOA decoder tools with:



mpegh3daExtElementConfig
mpegh3daExtElement
MPEG-H 3D audio core decoder
$z_1(k)$
$z_I(k)$
HOAConfig
HOAFrame
HOA Frame Converter
$\tilde{M}_{\mathrm{PAR}}(k,1), \tilde{P}_{\mathrm{PAR}}(k,1)$
$\tilde{M}_{\mathrm{PAR}}(k,B_{\mathrm{PAR}}), \tilde{P}_{\mathrm{PAR}}(k,B_{\mathrm{PAR}})$
$\mathcal{I}_{\mathrm{E}}(k), \mathcal{I}_{\mathrm{D}}(k), \mathcal{I}_{\mathrm{U}}(k)$
$\zeta(k)$
$\mathcal{M}_{\mathrm{DIR}}(k)$
$\mathcal{M}_{\mathrm{VEC}}(k)$
$e_1(k)$
$\beta_1(k)$
$e_I(k)$
$\beta_I(k)$
$v_{\mathrm{AMB,ASSIGN}}(k)$
$\tilde{\mathcal{M}}_{\mathrm{DIR}}(k,1)$
$\tilde{A}(k,1)$
$\tilde{\mathcal{M}}_{\mathrm{DIR}}(k,B)$
$\tilde{A}(k,B)$
$\tilde{b}_{\mathrm{SBP}}(k)$
Spatial HOA Decoding
$\check{C}(k)$

In subclause 12.1.2.1 HOA Decoding Tools replace:

— Ambient Synthesis

— HOA Compositions

with:

— Ambience Synthesis

— Preliminary HOA Composition

— Sub-band Directional Signals Synthesis

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

— Parametric Ambience Replication Decoder

— HOA Composition

Add new subclause 12.1.2.X "Layered Coding for HOA" after subclause 12.1.2.2 "HOA Renderer":

12.1.2.X Layered Coding for HOA

For the streaming of the compressed HOA sound field representation over a transmission channel with time-varying conditions layered coding is a means to adapt the quality of the received sound representation to the transmission conditions, and in particular to avoid undesired signal dropouts.

For layered coding the compressed HOA sound field representation is subdivided into a high priority base layer of a relatively small size and additional enhancement layers with decremental priorities and arbitrary sizes. Each enhancement layer is assumed to contain incremental information to complement that of all lower priority layers in order to improve the quality of the compressed HOA sound field representation. The idea is to control the amount of error protection for the transmission of the individual layers according to their priority. In particular, the base layer is provided with a high error protection, which is reasonable and affordable due to its low size.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Replace Table 119 — Syntax of HOAConfig() with:

Table 119 — Syntax of HOAConfig()

Syntax	No. of bits	Mnemonic
HOAConfig()		
{		
HoaOrder = escapedValue(3,5,0);	**3,8**	**uimsbf**
NumOfHoaCoeffs = (HoaOrder + 1)^2;		
IsScreenRelative;	**1**	**uimsbf**
UsesNfc;	**1**	**bslbf**
if (UsesNfc) {		
NfcReferenceDistance;	**32**	**bslbf**
}		
HOADecoderConfig();		
}		
NOTE: HoaOrder = 30 … 38 are reserved.		

Replace Table 120 — Syntax of HOADecoderConfig() with:

Table 120 — Syntax of HOADecoderConfig()

Syntax	No. of bits	Mnemonic
HOADecoderConfig(numHOATransportChannels)		
{		
MinAmbHoaOrder = escapedValue(3,5,0) – 1;	**3,8**	**uimsbf**
MinNumOfCoeffsForAmbHOA = (MinAmbHoaOrder + 1)^2;		
NumOfAdditionalCoders = numHOATransportChannels – MinNumOfCoeffsForAmbHOA;		
NumLayers = 1;		
NumHOAChannelsLayer[0] = numHOATransportChannels;		
if(**SingleLayer** == 0){	**1**	**Bslbf**

```
        HOALayerChBits = ceil(log2(NumOfAdditionalCoders));
        NumHOAChannelsLayer[0] = codedLayerCh +                        HOALayerChBits   uimsbf
                                MinNumOfCoeffsForAmbHOA;
        remainingCh = numHOATransportChannels –
                                NumHOAChannelsLayer[0];
        while (remainingCh>1) {
            HOALayerChBits = ceil(log2(remainingCh));
            NumHOAChannelsLayer[NumLayers] =                           HOALayerChBits   uimsbf
                NumHOAChannelsLayer[NumLayers-1] +
                    codedLayerCh + 1;
            remainingCh = remainingCh –
                        NumHOAChannelsLayer[NumLayers];
            NumLayers++;
        }
        if (remainingCh) {
            NumHOAChannelsLayer[NumLayers] =
                NumOfAdditionalCoders;
            NumLayers++;
        }
    }

    CodedSpatialInterpolationTime;                                     3                uimsbf
    SpatialInterpolationMethod;                                        1                bslbf
    CodedVVecLength;                                                   2                uimsbf
    MaxGainCorrAmpExp;                                                 3                uimsbf
    HOAFrameLengthIndicator;                                           2                uimsbf

    if( MinAmbHoaOrder < HoaOrder ) {
        DiffOrderBits = ceil( log2(HoaOrder- MinAmbHoaOrder+1))
        MaxHOAOrderToBeTransmitted = DiffOrder +                       DiffOrderBits    uimsbf
                                MinAmbHoaOrder;
    }
    else {
        MaxHOAOrderToBeTransmitted = HoaOrder;
    }
    MaxNumOfCoeffsToBeTransmitted =
                (MaxHOAOrderToBeTransmitted + 1)^2;
    MaxNumAddActiveAmbCoeffs =
                MaxNumOfCoeffsToBeTransmitted
                - MinNumOfCoeffsForAmbHOA;
    VqConfBits = ceil( log2( ceil( log2( NumOfHoaCoeffs+1 ))));
    NumVVecVqElementsBits;                                             VqConfBits       uimsbf
    if( MinAmbHoaOrder == 1) {
        UsePhaseShiftDecorr;                                           1                bslbf
    }

    if(SingleLayer==1) {
        HOADecoderEnhConfig();
    }

    AmbAsignmBits = ceil( log2( MaxNumAddActiveAmbCoeffs ) );
    ActivePredIdsBits = ceil( log2( NumOfHoaCoeffs ) );
    i = 1;
    while( i * ActivePredIdsBits
          + ceil( log2( i ) ) < NumOfHoaCoeffs ){
        i++;
    }
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
        NumActivePredIdsBits = ceil( log2( max( 1, i – 1 ) ) );
        GainCorrPrevAmpExpBits = ceil( log2( ceil( log2(
                                           1.5 * NumOfHoaCoeffs ) )
                                         + MaxGainCorrAmpExp + 1 ) );
        for (i=0; i<NumOfAdditionalCoders; ++i){
            AmbCoeffTransitionState[i] = 3;
        }
}
```

NOTE: MinAmbHoaOrder = 30 … 37 are reserved. HOAFrameLengthIndicator = 3 is reserved. CodedVVecLength = 3 is reserved.

Add the following tables before Figure 35 — Example for HOAConfig() in subclause 12.2.1 Configuration of HOA elements:

Table AMD3.HLC.1 — Syntax of HOAEnhConfig()

Syntax	No. of bits	Mnemonic
HOAEnhConfig()		
{		
HOALayerIdxBits = ceil(log2(NumOfAdditionalCoders+1));		
LayerIdx	HOALayerIdxBits	**uimsbf**
HOADecoderEnhConfig();		
}		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table AMD3.HLC.2 — Syntax of HOADecoderEnhConfig()

Syntax	No. of bits	Mnemonic
HOADecoderEnhConfig()		
{		
MaxNoOfDirSigsForPrediction = **MaxNoOfDirSigsForPrediction** + 1;	**2**	**uimsbf**
NoOfBitsPerScalefactor = **NoOfBitsPerScalefactor** + 1;	**4**	**uimsbf**
if(**PredSubbandsIdx** < 3) {	**2**	**uimsbf**
NumOfPredSubbands =		
NumOfPredSubbandsTable[PredSubbandsIdx];		
PredSubbandWidths =		
PredSubbandWidthTable[PredSubbandsIdx];		
}		
else {		
CodedNumberOfSubbands	**5**	**uimsbf**
NumOfPredSubbands = CodedNumberOfSubbands+1;		
PredSubbandWidths =		
getSubbandWidths(NumOfPredSubbands);		
}		
if (NumOfPredSubbands > 0) {		
FirstSBRSubbandIdxBits = ceil(log2 (NumOfPredSubbands+1));		
FirstSBRSubbandIdx;	**FirstSBR Subband IdxBits**	**uimsbf**
MaxNumOfPredDirs **=** 2^(**MaxNumOfPredDirsLog2**);	**3**	**uimsbf**
MaxNumOfPredDirsPerBand **=** escapedValue(3,2,5) + 1;		
NumOfBitsPerDirIdx =		
NumOfBitsPerDirIdxTable[**DirGridTableIdx**];	**2**	**uimsbf**
}		
if(**ParSubbandTableIdx** < 3) {	**2**	**uimsbf**
NumOfParSubbands =		
NumOfParSubbandsTable[ParSubbandTableIdx];		
ParSubbandWidths =		
ParSubbandWidthTable[ParSubbandTableIdx];		
}		
else {		
CodedNumberOfSubbands	**5**	**uimsbf**
NumOfParSubbands = CodedNumberOfSubbands+1;		
ParSubbandWidths =		
getSubbandWidths(NumOfParSubbands);		
}		
if(NumOfParSubbands > 0) {		
LastFirstOrderSubbandIdxBits =		
ceil(log2(NumOfParSubbands + 1));		
LastFirstOrderSubbandIdx;	**LastFirst OrderSu bbandIdx Bits**	**uimsbf**
for (idx = 0; idx < NumOfParSubbands; idx++) {		
UseRealCoeffsPerParSubband[idx];	**1**	**bslbf**
}		
for (idx = 0; idx < LastFirstOrderSubBandIdx; idx++) {		
UpmixHoaOrderPerParSubband[idx] = 1;		
MaxNumOfDecoSigs[idx]=		
(UpmixHoaOrderPerParSubband[idx] + 1)^2;		
}		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
        for ( idx = LastFirstOrderSubBandIdx;
            idx < NumOfParSubbands; idx++) {
            UpmixHoaOrderPerParSubband[idx] = 2;
            MaxNumOfDecoSigs[idx] =
                (UpmixHoaOrderPerParSubband[idx] + 1)^2;
        }
    }
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table AMD1.1 — Syntax of getSubbandWidths()

Syntax	No. of bits	Mnemonic
getSubbandWidths(NumberOfSubbands)		
{		
totalBwSum = 0;		
if(NumberOfSubbands > 1) {		
CodedBwFirstBand	**1..**	**uclbf**
bw[0] = CodedBwFirstBand+1;		
totalBwSum = totalBwSum + bw[0];		
if(NumberOfSubbands > 2) {		
for (nb = 1; nb < NumberOfSubbands-2; nb++) {		
bw[nb] = bw[nb-1] + **bw_diff**;	**1..**	**uclbf**
totalBwSum = totalBwSum + bw[nb];		
}		
bw[nb] = bw[nb-1] + **bw_diff**;	**5**	**uimsbf**
totalBwSum = totalBwSum + bw[nb];		
}		
}		
bw[NumberOfSubbands-1] = 64 – totalBwSum;		
return(bw);		
}		

Remove Figure 35 — Example for HOAConfig()

Replace Table 121 — Syntax of HOAFrame with:

Table 121 — Syntax of HOAFrame

Syntax	No. of bits	Mnemonic
HOAFrame()		
{		
NumOfDirSigs = 0;		
NumOfVecSigs = 0;		
NumOfAddHoaChans = 0;		
for(lay=0; (lay< NumLayers); ++lay){		
NumOfDirSigsPerLayer[lay] = 0;		
NumOfAddHoaChansPerLayer[lay] = 0;		
NumOfContAddHoaChans[lay] = 0;		
NumOfNewAddHoaChans[lay] = 0;		
}		
hoaIndependencyFlag;	**1**	**bslbf**

```
for(i=0; i< NumOfAdditionalCoders; ++i){
    ChannelSideInfoData(i);
    HOAGainCorrectionData(i);
    switch ChannelType[i] {
    case 0:
        DirSigChannelIds[NumOfDirSigs] = i + 1;
        NumOfDirSigs++;
        for(lay=0; (lay< NumLayers); ++lay){
            if( (MinNumOfCoeffsForAmbHOA + i ) <
                    NumHOAChannelsLayer[lay]){
                NumOfDirSigsPerLayer[lay]++;
            }
        }
        break;
    case 1:
        VecSigChannelIds[NumOfVecSigs] = i + 1;
        lay = 0;
        while( (MinNumOfCoeffsForAmbHOA + i )
                ≥ NumHOAChannelsLayer[lay]) {
            lay ++;
        }
        VecSigLayerIdx[NumOfVecSigs] = lay;
        NumOfVecSigs++;
        break;
    case 2:
        for(lay=0; (lay< NumLayers); ++lay){
            if( (MinNumOfCoeffsForAmbHOA + i ) <
                    NumHOAChannelsLayer[lay]){
                if (AmbCoeffTransitionState[i] == 0) {
                    ContAddHoaCoeff[lay]
                        [NumOfContAddHoaChans[lay]]
                            = AmbCoeffIdx[i];
                    NumOfContAddHoaChans[lay]++;
                }
            }else{
                if(AmbCoeffTransitionState[i] == 1) {
                    NewAddHoaCoeff[lay]
                        [NumOfNewAddHoaChans]
                            = AmbCoeffIdx[i];
                    NumOfNewAddHoaChans[lay]++;
                }


                    AddHoaCoeffPerLayer[lay]
                        [NumOfAddHoaChans]
                            = AmbCoeffIdx[i];
                    NumOfAddHoaChansPerLayer[lay]++;
                }
            }
        AddHoaCoeff[NumOfAddHoaChans] = AmbCoeffIdx[i];
        NumOfAddHoaChans++;
        break;
    }
}

for ( i= NumOfAdditionalCoders;
        i< NumHOATransportChannels; ++i){
    HOAGainCorrectionData(i);
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
    for(i=0; i< NumOfVecSigs; ++i){
        VVectorData ( VecSigChannelIds(i) );
    }

    if(SingleLayer==1) {
        HOAEnhFrame();
    }
}
```

NOTE: the encoder shall set hoaIndependencyFlag to 1 if usacIndependencyFlag (see mpegh3daFrame() in Table 26) is set to 1.
NOTE: If SingleLayer == 1 set NumLayers = 1.

Add the following table into Section 12.2.2 after Table 121

Table AMD3.HLC.3 — Syntax of HOAEnhFrame

Syntax	No. of bits	Mnemonic
HOAEnhFrame()		
{		
if(((SingleLayer==1) & (NumOfDirSigs > 0)) |		
((SingleLayer==0) & (NumOfDirSigsPerLayer[lay]) > 0)){		
HOAPredictionInfo()		
}		
if(NumOfPredSubbands > 0) {		
HOADirectionalPredictionInfo();		
}		
if(NumOfParSubbands > 0) {		
HOAParInfo();		
}		
}		
Note: lay is the index of the currently active HOA enhancement layer If SingleLayer==0 then use NumOfPredSubbands and NumOfParSubbands from the **HOADecoderEnhConfig** of the corresponding layer with index lay		

Replace Table 122 — Syntax of ChannelSideInfoData(i) with:

Table 122 — Syntax of ChannelSideInfoData(i)

Syntax	No. of bits	Mnemonic
ChannelSideInfoData(i)		
{		
ChannelType[i]	**2**	**uimsbf**
switch ChannelType[i]		
{		
case 0:		
ActiveDirsIds[i];	**10**	**uimsbf**
break;		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
    case 1:
        if(hoaIndependencyFlag){
            if(CodedVVecLength==1){
                NewChannelTypeOne(k)[i];                                1             bslbf
            }
            NbitsQ(k)[i]                                                4             uimsbf
            if (NbitsQ(k)[i] == 4) {
                CodebkIdx(k)[i];                                        3             uimsbf
                NumVvecIndices(k)[i]++;                                 NumVVecV      uimsbf
                                                                        qElements
                                                                        Bits
            }
            elseif (NbitsQ(k)[i] >= 6) {
                PFlag(k)[i] = 0;
                CbFlag(k)[i];                                           1             bslbf
            }
        }
        else{
            if(CodedVVecLength==1){
                NewChannelTypeOne(k)[i] = (1!=ChannelType(k-1)[i]));
            }
            bA;                                                         1             bslbf
            bB;                                                         1             bslbf
            if ((bA + bB) == 0) {
                NbitsQ(k)[i] = NbitsQ(k-1)[i];
                PFlag(k)[i] = PFlag(k-1)[i];
                CbFlag(k)[i] = CbFlag(k-1)[i];
                CodebkIdx(k)[i] = CodebkIdx(k-1)[i];
                NumVvecIndices(k)[i] =
                            NumVvecIndices(k-1)[i];
            }
            else{
                NbitsQ(k)[i]  = (8*bA)+(4*bB)+uintC;                    2             uimsbf
                if (NbitsQ(k)[i] == 4) {
                    CodebkIdx(k)[i];                                    3             uimsbf
                NumVvecIndices(k)[i]++;                                 NumVVecV      uimsbf
                                                                        qElements
                                                                        Bits
                }
                elseif (NbitsQ(k)[i] >= 6) {
                PFlag(k)[i];                                            1             bslbf
                CbFlag(k)[i];                                           1             bslbf
                }
            }
        }
        break;
    case 2:
        AddAmbHoaInfoChannel(i);
        break;
    default:
  }
}
```

NOTE   CodebkIdx = 4 … 6 are reserved.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Replace Table 126 — Syntax of VVectorData() with:

Syntax	No. of bits	Mnemonic
VVectorData(i)		
{		
if (CodedVVecLength == 1) {		
VVecLengthUsed = VVecLength[i];		
VVecCoeffIdUsed = VVecCoeffId[i];		
} else {		
VVecLengthUsed = VVecLength;		
VVecCoeffIdUsed = VVecCoeffId;		
}		
if (NbitsQ(k)[i] == 4) {		
If (NumVvecIndices(k)[i] == 1) {		
VvecIdx[0] = **VvecIdx** + 1;	**nbitsIdx**	**uimsbf**
WeightVal[0] = ((**SgnVal***2)-1);	**1**	**uimsbf**
} else {		
WeightIdx;	**8**	**uimsbf**
for (j=0; j< NumVvecIndices(k)[i]; ++j) {		
VvecIdx[j] = **VvecIdx** + 1;	**nbitsIdx**	**uimsbf**
if (j<8) {		
WeightVal[j] = ((**SgnVal***2)-1) * WeightValCdbk[WeightIdx][j];	**1**	**uimsbf**
} else {		
WeightVal[j] = ((**SgnVal***2)-1) * WeightValCdbk[WeightIdx][6+j%2];	**1**	**uimsbf**
}		
}		
}		
}		
else if (NbitsQ(k)[i] == 5) {		
for (m=0; m< VVecLengthUsed; ++m){		
aVal[i][m] = (**VecVal** / 128.0) – 1.0;	**8**	**uimsbf**
}		
else if(NbitsQ(k)[i] >= 6) {		
for (m=0; m< VVecLengthUsed; ++m){		
huffIdx = *huffSelect*(VVecCoeffIdUsed[m], PFlag[i], CbFlag[i]);		
cid = *huffDecode*(NbitsQ[i], huffIdx, **huffVal**);	**dynamic**	**huffDecode**
aVal[i][m] = 0.0;		
if (cid > 0) {		
aVal[i][m] = sgn = (**sgnVal** * 2) - 1;	**1**	**bslbf**
if (cid > 1) {		
aVal[i][m] = sgn * (2.0^(cid -1) + **intAddVal**);	**cid-1**	**uimsbf**
}		
}		
}		
}		
}		
NOTE: See 12.4.1.10.X for computation of VVecLength and nbitsIdx		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Include following changes in Table 127 Syntax of HOAPredictionInfo(), which is defined in Section 12.2.2 (changes highlighted in grey)

Table 127 — Syntax of HOAPredictionInfo()

Syntax	No. of bits	Mnemonic
HOAPredictionInfo()		
{		
PredIdsBits = ceil(log2(NumOfDirSigsPerLayer[lay] + 1));		
if(**PSPredictionActive**){	**1**	**bslbf**
NumActivePred = 0;		
if(**KindOfCodedPredIds**){	**1**	**bslbf**
NumActivePred = **NumActivePredIds** + 1;	**NumActivePredIdsBits**	**uimsbf**
i=0;		
while(i < NumActivePred){		
PredIds[i] = **PredIds**[i] + 1;	**ActivePredIdsBits**	**uimsbf**
i++;		
}		
}		
else{		
for (i=0; i<(HoaOrder +1)^2; i++) {		
if(**ActivePred[i]**) {	**1**	**bslbf**
NumActivePred ++;		
}		
}		
}		
NumOfGains=0;		
for (i=0; i<NumActivePred * MaxNoOfDirSigsForPrediction; i++) {		
if (**PredDirSigIds[i]** > 0) {	**PredIdsBits**	**uimsbf**
PredDirSigIds[i] =		
DirSigChannelIds[PredDirSigIds[i] - 1];		
NumOfGains++;		
}		
}		
for (i=0; i< NumOfGains; i++) {		
bsPredGains	**NoOfBitsPerScalefactor**	**uimsbf**
PredGains[i] = bsPredGains – $2^{(\text{NoOfBitsPerScalefactor}-1)}$		
}		
}		
}		
Note: lay is the index of the currently active HOA enhancement layer. In case of SingleLayer==1 set lay to zero.		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Add the following tables before Figure 36 — Two examples for HOAFrame()

Table AMD1.2 — Syntax of HOADirectionalPredictionInfo()

Syntax	No. of bits	Mnemonic
HOADirectionalPredictionInfo()		
{		
if(**UseDirectionalPrediction**) {	**1**	**bslbf**
if (!hoaIndependencyFlag**)** {		
KeepPreviousGlobalPredDirsFlag;	**1**	**bslbf**
}		
else{		
KeepPreviousGlobalPredDirsFlag = 0;		
}		
if(!KeepPreviousGlobalPredDirsFlag) {		
NumOfGlobalPredDirs = **NumOfGlobalPredDirs** + 1;	MaxNumOfPredDirsLog2	**bslbf**
NumBitsForRelDirGridIdx = ceil(		
log2(NumOfGlobalPredDirs));		
for (idx=0; idx < NumOfGlobalPredDirs; idx++) {		
GlobalPredDirsIds[idx];	NumOfBitsPerDirIdx	**uimsbf**
}		
}		
else{		
/* Keep values from previous HOADirectionalPredictionInfo		
payload for NumOfGlobalPredDirs and		
GlobalPredDirsIds. */		
}		
SortedAddHoaCoeff = sort(AddHoaCoeffPerLayer[lay],		
'ascend');		
for (band = 0; band < NumOfPredSubbands; band++) {		
for (dir = 0; dir < MaxNumOfPredDirsPerBand; dir++) {		
for (hoaIdx = 0;		
hoaIdx < MinNumOfCoeffsForAmbHOA;		
hoaIdx++) {		
DecodedMagDiff[band][dir][hoaIdx] = 0;		
DecodedAngleDiff[band][dir][hoaIdx] = 0;		
}		
}		
}		
for (band = 0; band < NumOfPredSubbands; band++) {		
if (!hoaIndependencyFlag**)** {		
KeepPreviousDirPredMatrixFlag[band];	**1**	**bslbf**
}		
else{		
KeepPreviousDirPredMatrixFlag[band] = 0;		
}		
if (!KeepPreviousDirPredMatrixFlag[band]) {		
UseHuffmanCodingDiffMag;	**1**	**bslbf**
if(band < FirstSBRSubbandIdx) {		
UseHuffmanCodingDiffAngle;	**1**	**bslbf**
for (dir = 0; dir < MaxNumOfPredDirsPerBand; dir++) {		
if (**DirIsActive[band][dir]**) {	**1**	**bslbf**
RelDirGridIdx;	NumBitsForRelDirGridIdx	**uimsbf**

STANDARDSISO.COM Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
                        PredDirGridIdx[band][dir] =
                                GlobalPredDirsIds[RelDirGridIdx];
                        for ( hoaIdx = 0;
                            hoaIdx < MinNumOfCoeffsForAmbHOA;
                            hoaIdx++ ) {
                            readDirPredDiffValues (band, dir, hoaIdx,
                                UseHuffmanCodingDiffAbs,
                                UseHuffmanCodingDiffAngle,
                                FirstSBRSubbandIdx);
                        }
                        for ( idx = 0;
                            idx < NumOfAddHoaChansPerLayer[lay];
                            idx++ ) {
                            readDirPredDiffValues (band, dir,
                                SortedAddHoaCoeff[idx] -1,
                                UseHuffmanCodingDiffAbs,
                                UseHuffmanCodingDiffAngle,
                                FirstSBRSubbandIdx);
                        }
                    }
                }
            }
        }
    }
}
```

Note: lay is the index of the currently active HOA enhancement layer.
In case of SingleLayer==1 set lay to zero.

Table AMD1.3 — Syntax for readDirPredDiffValues()

```
Syntax                                                          No. of bits   Mnemonic
readDirPredDiffValues(band, dir, hoaIdx,
                UseHuffmanCodingDiffAbs,
                UseHuffmanCodingDiffAngle,
                FirstSBRSubbandIdx)
{
    if(UseHuffmanCodingDiffAbs) {
        if( band < FirstSBRSubbandIdx ) {
            DecodedMagDiff[band][dir][hoaIdx] =                 1..10         vlclbf
                HuffmanMagDiffNoSbr[HuffmanCodedMagDiff];
        else {
            DecodedMagDiff[band][dir][hoaIdx] =                 1..12         vlclbf
                HuffmanMagDiffSbr[HuffmanCodedRealMagDiff];
        }
        if(DecodedMagDiff[band][dir][hoaIdx] ≤ -8){
            DecodedMagDiff[band][dir][hoaIdx] = -8 -            1..           uclbf
                runLengthCodedVal;
            {
            else if (DecodedMagDiff[band][dir][hoaIdx] ≥ 9){
                DecodedMagDiff[band][dir][hoaIdx] = 9 +         1..           uclbf
                    runLengthCodedVal;
            }
        }
    }
    else {
        DecodedMagDiff[band][dir][hoaIdx] = CodedMagDiff -7;    4             uimsbf
        if(DecodedMagDiff[band][dir][hoaIdx] ≤ -7){
            DecodedMagDiff[band][dir][hoaIdx] = -7 -            1..           uclbf
                runLengthCodedVal;
        {
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

else if (DecodedMagDiff[band][dir][hoaIdx] ≥ 8)**{**		
DecodedMagDiff[band][dir][hoaIdx] = 8 +	**1..**	**uclbf**
runLengthCodedVal;		
}		
}		
if(band < FirstSBRSubbandIdx) {		
if(UseHuffmanCodingDiffAngle) {		
DecodedAngleDiff[band][dir][hoaIdx] =	**1..7**	**vlclbf**
DecTableAngleDiff[**HuffCodedAngleDiff**];		
}		
else {		
DecodedAngleDiff[band][dir][hoaIdx] =	**4**	**uimsbf**
DecTableAngleDiff[**CodedAngleDiff**];		
}		
}		
}		

Table AMD1.4 — Syntax of HOAParInfo()

Syntax	No. of bits	Mnemonic
HOAParInfo()		
{		
if (**UsePar**) {	**1**	**bslbf**
for (band = 0; band < NumOfParSubbands; band++) {		
for(n = 0;		
n < MaxNumOfDecoSigs[band];		
n++) {		
for(m = 0;		
m < MaxNumOfDecoSigs[band];		
m++) {		
DecodedParMagDiff [band][n][m] = 0;		
DecodedParAngleDiff [band][n][m] = 0;		
}		
}		
if (!hoaIndependencyFlag) {		
KeepPreviousParMatrixFlag[band];	**1**	**bslbf**
}		
else{		
KeepPreviousParMatrixFlag[band] = 0;		
}		
if (!KeepPreviousParMatrixFlag[band]) {		
ParDecorrSigsSelectionTableIdx[band]	**2**	**uimsbf**
if (UpmixHoaOrderPerParSubband[band] == 2) {		
NumOfDecorrSigsPerParSubband =		
NumOfDecorrSigsPerParSubbandTable[		
ParDecorrSigsSelectionTableIdx[band]]		
ParSelectedDecorrSigsIdxMatrix =		
ParSelectedDecorrSigsIdxMatrixTable[		
ParDecorrSigsSelectionTableIdx[band]]		
}		
else{		
NumOfDecorrSigsPerParSubband =		
NumOfDecorrSigsPerFirstOrderPar		
SubbandTable[		
ParDecorrSigsSelectionTableIdx[band]]		
ParSelectedDecorrSigsIdxMatrix =		
ParFirstOrderSelectedDecorr		
SigsIdxMatrixTable [		
ParDecorrSigsSelectionTableIdx[band]]		
}		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

if (**UseReducedNoOfUpmixSigs**){	**1**	**bslbf**
for(n = 0;		
n < MaxNumOfDecoSigs[band]; n++) {		
UseParUpmixSig[band][n];	**1**	**bslbf**
}		
}		
else {		
for(n = 0;		
n < MaxNumOfDecoSigs[band]; n++) {		
UseParUpmixSig[band][n] = 1;		
}		
}		
UseParHuffmanCodingDiffAbs;	**1**	**bslbf**
if(!UseRealCoeffsPerParSubband[band]) {		
UseParHuffmanCodingDiffAngle;	**1**	**bslbf**
}		
for(n = 0;		
n < MaxNumOfDecoSigs[band];		
n++) {		
if(UseParUpmixSig[band][n]) {		
for(m = 0;		
m <		
NumOfDecorrSigsPerParSubband ;		
m++) {		
c = ParSelectedDecorrSigsIdxMatrix[n][m];		
readParDiffValues(band, n, c,		
UseParHuffmanCodingDiffAbs,		
UseParHuffmanCodingDiffAngle);		
}		
}		
}		
}		
}		
}		
}		

Table AMD1.5 — Syntax for readParDiffValues()

Syntax	No. of bits	Mnemonic
readParDiffValues (band, idx, decoIdx, UseParHuffmanCodingDiffAbs,		
UseParHuffmanCodingDiffAngle)		
{		
if(UseParHuffmanCodingDiffAbs) {		
if(UseRealCoeffsPerParSubband[band]) {		
DecodedParMagDiff[band][idx][decoIdx] =	**1..10**	**vlclbf**
HuffmanMagDiffSbr[**HuffmanCodedParMagDiff**];		
else {		
DecodedParMagDiff[band][idx][decoIdx] =	**1..12**	**vlclbf**
HuffmanMagDiffNoSbr[**HuffmanCodedRealParMagDiff**];		
}		
if (DecodedParMagDiff [band][dir][hoaIdx] ≤ -8**)**		
DecodedParMagDiff[band][idx][decoIdx] = -8 -	**1..**	**uclbf**
runLengthCodedVal;		
}		
else if (DecodedParMagDiff [band][dir][hoaIdx] ≥ 9)		
DecodedParMagDiff[band][idx][decoIdx] = 9 +	**1..**	**uclbf**
runLengthCodedVal;		
}		
else {		
DecodedParMagDiff[band][idx][decoIdx] =	**4**	**uimsbf**
CodedParMagDiff - 7;		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

if (DecodedParMagDiff [band][dir][hoaIdx] ≤ -7)		
DecodedParMagDiff[band][idx][decoIdx] = -7 - **runLengthCodedVal**;	**1..**	**uclbf**
}		
else if (DecodedParMagDiff [band][dir][hoaIdx] ≥ 8)		
DecodedParMagDiff[band][idx][decoIdx] = 8 + **runLengthCodedVal**;	**1..**	**uclbf**
}		
}		
if(!UseRealCoeffsPerParSubband[band]){		
if(UseParHuffmanCodingDiffAngle) {		
DecodedParAngleDiff[band][idx][decoIdx] = DecTableAngleDiff[**HuffCodedParAngleDiff**];	**1..7**	**vlclbf**
}		
else {		
DecodedParAngleDiff[band][idx][decoIdx] = DecTableAngleDiff[**CodedParAngleDiff**];	**4**	**uimsbf**
}		
}		
}		

Replace "Table 128 – SingleLayer definition" with the following table and add the additional definitions of elements following the new Table 128 :

Table 128 — SingleLayer definition

Value	Meaning
0	/* reserved */
1	HOA signal is provided in a single layer

With:

Table 128 — SingleLayer definition

Value	Meaning
0	HOA signal is provided in multiple layers; enables the signaling of the distribution of the HOA transport channels into the different layers
1	HOA signal is provided in a single layer

codedLayerCh	This element indicates for the first (i.e. base) layer the number of included transport signals, which is given by **codedLayerCh** + MinNumOfCoeffsForAmbHOA. For the higher (i.e. enhancement) layers, this element indicates the number of additional signals included into an enhancement layer compared to the next lower layer, which is given by **codedLayerCh** + 1.
HOALayerChBits	This element indicates the number of bits for reading **codedLayerCh.**
NumLayers	This element indicates (after the reading of the HOADecoderConfig()) the total number of layers within the bit stream.
NumHOACannelsLayer	This element is an array consisting of NumLayers elements, of which the i-th element indicates the number of transport signals included in all layers up to the i-th layer.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

HOAEnhConfig()	This payload contains the HOA enhancement configuration data of an HOA enhancement layer in the HOA Layered Coding Mode.
HOALayerIdxBits	This element indicates the number of bits used for signaling the element LayerIdx.
LayerIdx	This element indicates the index of the HOA enhancement layer payload, where the index is defined in the range from zero to NumLayers-1.
HOADecoderEnhConfig()	This payload contains all configuration elements for the HOA enhancement layer payloads.

In subclause 12.3.1 Definitions of HOA Config replace:

NumOfHoaCoeffs	This element determines the number of HOA coefficients of the coded HOA representation, which is equal to the number of HOA coefficients to be reconstructed.

With:

NumOfHoaCoeffs	This element determines the number of HOA coefficients of the coded HOA representation, which is equal to the number of HOA coefficients to be reconstructed.
IsScreenRelative	This element indicates if the HOA representation shall be rendered with respect to the reproduction screen size as described in clause 18.X.

In subclause 12.3.1 Definitions of HOA Config replace:

MaxNumAddActiveAmbCoeffs	This element determines the maximum number of additional HOA channels that can be used additionally for the coding of the ambient HOA representation.

With:

MaxHOAOrderToBeTransmitted	This element indicates the maximum HOA order of the additional ambient HOA coefficients to be transmitted.
MaxNumOfCoeffsToBeTransmitted	This element indicates the maximum number of HOA coefficients to be transmitted, computed depending on MaxHOAOrderToBeTransmitted.
MaxNumAddActiveAmbCoeffs	This element signals the maximum index for the signaling of additional ambient HOA coefficients.
VqConfBits	This element indicates the number of bits necessary to signal the element **NumVVecVqElementsBits**
NumVVecVqElementsBits	This element indicates the number of bits used to signal the element **NumVvecIndices** in ChannelSideInfoData()
UsePhaseShiftDecorr	This element signals the usage of the phase shift de-correlation in the ambience synthesis, where true means it is used.
PredSubbandsIdx	This element signals the table index for the sub-band configuration of the Sub-band Directional Signals Synthesis

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

NumOfPredSubbands	This element contains the number of sub-band groups used for the Sub-band Directional Signals Synthesis
NumOfPredSubbandsTable	This table contains the number of sub-band groups for each PredSubbandsIdx for the Sub-band Directional Signals Synthesis
PredSubbandWidths[idx]	This array of NumOfPredSubbands elements contains the number of QMF sub-bands per sub-band group of the Sub-band Directional Signals Synthesis.
PredSubbandWidthTable	This table contains the bandwidths of the sub-band groups for each table index PredSubbandsIdx for the Sub-band Directional Signals Synthesis

Table AMD1.6 — Directional Prediction Subbands Table

PredSubbandsIdx	NumOfPredSubbandsTable	PredSubbandWidthTable
0	0	[]
1	10	[1, 1, 1, 2, 2, 2, 3, 6, 11, 35]
2	20	[1,1,1,1,1,1,1,1,1,1,1,2,2,2,2,4,5,7,11, 18]

getSubbandWidths()	This function reads a flexible sub-band group configuration.
FirstSBRSubbandIdxBits	This element determines the number of bits for reading **FirstSBRSubbandIdx.**
FirstSBRSubbandIdx	Indicates the first sub-band group of the Directional Signals Synthesis where the samples are reconstructed by the SBR tool. To deactivate the special SBR processing in the Sub-band Directional Signals Synthesis set the value of this element to NumOfPredSubbands + 1.
MaxNumOfPredDirsLog2	This element signals the logarithm to the base of two from the maximum number of signals that are predicted in the Sub-band Directional Signals Synthesis.
MaxNumOfPredDirs	This element signals the maximum number of signals that are predicted in the Sub-band Directional Signals Synthesis.
MaxNumOfPredDirsPerBand	This element contains the maximum number of predicted signals per sub-band group of the Sub-band Directional Signals Synthesis.
DirGridTableIdx	This index determines the grid of potential directions of directional sub-band signals created at the Sub-band Directional Signals Synthesis.
NumOfBitsPerDirIdx	The element determines the number of bits for signaling the index of a virtual loudspeaker position index.
NumOfGridPointsTable	This table determines the number of potential directions of directional sub-band signals created at the Sub-band Directional Signals Synthesis.
NumOfBitsPerDirIdxTable	This table determines the number of bits used to code each single potential direction of a directional sub-band signal created at the Sub-band Directional Signals Synthesis.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table AMD1.7 — Quantized Direction Index Table

DirGridTableIdx	NumOfGridPointsTable	NumOfBitsPerDirIdxTable
0	256	8
1	484	9
2	900	10
3	reserved	reserved

ParSubbandTableIdx	This element signals the table index for the sub-band configuration of the Parametric Ambience Replication decoder
NumOfParSubbands	This element signals the number of sub-band groups used by the Parametric Ambience Replication decoder
NumOfParSubbandsTable	This table contains the number of sub-band groups for each PAR sub-band group table index **ParSubbandTableIdx**
ParSubbandWidths	This array of NumOfParSubbands elements contains the number of QMF sub-bands per sub-band group of the Parametric Ambience Replication decoder.
ParSubbandWidthTable	This table contains the bandwidths of the sub-band groups for each table index ParSubbandTableIdx for the PAR decoder

Table AMD1.8 — PAR Subbands Table

ParSubbandTableIdx	NumOfParSubbandsTable	ParSubbandWidthTable
0	0	[]
1	4	[1, 1, 22, 40]
2	8	[1, 1, 1, 2, 2, 5, 10, 42]

LastFirstOrderSubbandIdxBits	This element determines the number of bits required for reading LastFirstOrderSubbandIdx
LastFirstOrderSubbandIdx	This element indicates the index of the last PAR sub-band group that uses an HOA order of one.
UseRealCoeffsPerParSubband[idx]	This Boolean array indicates for each PAR sub-band group if the mixing matrix consists of real-valued non-negative (true) or complex valued elements matrix (false).
UpmixHoaOrderPerParSubband[idx]	This array contains the HOA order used in each PAR sub-band group
MaxNumOfDecoSigs[idx]	This array contains the maximum number of de-correlated signals per PAR sub-band group.

Add the following subclause after subclause 12.3.1 Definitions of HOA Config

12.3.x Syntax of getSubbandBandwidths()

This function reads the bandwidth for NumberOfSubbands sub-band groups.

CodedBwFirstBand	This element signals the bandwidth of the first sub-band group by the number of QMF sub-bands
NumberOfSubbands	This element contains the total number of sub-band groups

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

bw[idx]	This array holds the bandwidth of each sub-band group in the number of QMF sub-bands
bw_diff	This element contains the differentially coded bandwidth of a sub-band group

Replace the subclause 12.3.2.1 HOAFrame() with the following subclause

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

12.3.2.1 HOAFrame()

The HOAFrame() holds the information that is required to decode the L samples of a HOA frame of order HoaOrder.

HOAFrame()	This block of data contains the data required to decode one frame of L samples of the coded HOA representation.
NumOfDirSigs	This elements determines the number of active directional signals in the current HOAFrame().
NumOfVecSigs	This element determines the number of active vector-based signals in the current HOAFrame().
NumOfAddHoaChans	This element determines the total number of additional ambient HOA channels in the current HOAFrame().
NumOfDirSigsPerLayer[lay]	This elements determines the number of active directional signals in the current HOAFrame() that are actually used in the HOA enhancement layer *lay*.
NumOfAddHoaChansPerLayer[lay]	This element signals the total number of additional ambient HOA coefficients actually used in the HOA enhancement layer *lay*.
NumOfContAddHoaChans[lay]	This elements determines the number of continuous additional ambient HOA coefficients in the current HOAFrame() that are actually used in the HOA enhancement layer *lay*.
NumOfNewAddHoaChans[lay]	This element determines the number of newly introduced additional ambient HOA channels in the current HOAFrame() that are actually used in the HOA enhancement layer *lay*.
hoaIndependencyFlag	This flag signals that the current frame is an independent frame that can be decoded without having knowledge about the previous frame. Otherwise this flag is equal to the usacIndependencyFlag**.**
ChannelSideInfoData(i)	This payload holds the side information for the i-th of the NumOfAdditionalCoders channels of flexible ChannelType.
HOAGainCorrectionData(i)	This payload contains data for the inverse gain correction of channel i.
DirSigChannelIds[NumOfDirSigs]	This element stores the channel index of each active directional signal of the current frame.
VecSigChannelIds[NumOfVecSigs]	This element stores the channel index of each active vector-based signal of the current frame.

VecSigLayerIdx[i]	This element indicates for the i-th vector based channel element of the current frame the index of the enhancement layer that transmits the corresponding vector-based signal.
ContAddHoaCoeff [lay][NumOfContAddHoaChans]	
	This 2D array contains the HOA coefficient indices for each continuous additional ambient HOA coefficient actually used in the HOA enhancement layer *lay*. For these HOA coefficients the AmbCoeffTransitionState word is of value 0 in the current frame.
NewAddHoaCoeff [lay][NumOfContAddHoaChans]	
	This 2D array contains the HOA coefficient indices for each additional ambient HOA coefficient, which was not present in the preceding frame used in the HOA enhancement layer *lay*.
AddHoaCoeffPerLayer[lay]	This array contains the HOA coefficient indices for each additional ambient HOA coefficient actually used in the HOA enhancement layer lay.
AddHoaCoeff[lay]	This element stores the HOA coefficient number of each additional ambient HOA channel in the current frame.
HOAEnhFrame()	This payload contains the frame data for the HOA enhancement payloads HOAPredictionInfo(), HOADirectionalPredictionInfo() and HOAParInfo().

Add the subclause 12.3.2.X after 12.3.2.1 HOAFrame()

12.3.2.X HOAEnhFrame()

HOAPredictionInfo()	
	This payload contains data for the prediction of dominant sound sources from the active directional signals of the current frame.
HOADirectionalPredictionInfo	This payload contains data for the Sub-band Directional Signals Synthesis.
HOAParInfo	This payload contains data for the Parametric Ambience Replication.

In subclause 12.3.2.2 ChannelSideInfoData(i) add after CodebkIdx[i]

NumVvecIndices(k)[i]	The number of vectors used to dequantize a vector-quantized V-vector.

In subclause 12.3.2.2 ChannelSideInfoData(i) add before PFlag[i]

NewChannelTypeOne[i]	This flag indicates if in the previous frame (k-1) the transport channel was not initialized as a Vector-based Signal.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In subclause 12.3.2.5 VVectorData(VecSigChannelIds(i)) remove:

NumVvecIndices	The number of vectors used to dequantize a vector-quantized V-vector.

Add the following subclauses after 12.3.2.6 HOAPredictionInfo():

12.3.2.x HOADirectionalPredictionInfo

This Payload contains data for the Directional Signals Synthesis.

UseDirectionalPrediction	This flag indicates if the Directional Signals Synthesis is performed in the current frame.
KeepPreviousGlobalPredDirsFlag	This flag indicates that the elements NumOfGlobalPredDirs, NumBitsForRelDirGridIdx and GlobalPredDirsIds read from the previous HOADirectionalPredictionInfo payload are used.
NumOfGlobalPredDirs	This element determines the available number of global, broadband directions.
NumBitsForRelDirGridIdx	This element indicates the number of bit required for reading **GlobalPredDirsIds.**
GlobalPredDirsIds	This array contains the indices for the NumOfGlobalPredDirs directions, where each index belongs to a direction from the grid of positions selected by **DirGridTableIdx.**
SortedAddHoaCoeff	This array contains the HOA coefficients indices from AddHoaCoeff sorted ascendingly.
KeepPreviousDirPredMatrixFlag[band]	This Boolean is true if the signal prediction matrix is kept from the previous frame and false if the mixing matrix is update by new magnitude and angle differences.
UseHuffmanCodingDiffMag	This Boolean element is true if the magnitude differences are encoded by Huffman coding.
UseHuffmanCodingDiffAngle	This Boolean element is true if the angle differences are encoded by Huffman coding.
DirIsActive	This Boolean element is true if the current direction of the current sub-band group has been transmitted.
RelDirGridIdx	This element holds the relative index of the current direction index stored in GlobalPredDirsIds
PredDirGridIdx[band][dir]	This 2D array contains the direction indices belonging to the direction grid selected by **DirGridTableIdx** for all active directions of each sub-band group
readDirPredDiffValues()	This function reads the coded magnitude and angle differences.

12.3.2.x.1 Syntax for readDirPredDiffValues()

This function reads the magnitude and phase differences for the Directional Signal Synthesis.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

HuffCodedMagDiff This element signals the Huffman coded magnitude difference using Table **F.x Huffman Table for Decoding HuffmanMagDiffNoSbr**

DecodedMagDiff[band][dir][idx] This 3D array holds the decoded magnitude difference from Table **F.x Huffman Table for Decoding HuffmanMagDiffSbr** for the prediction of each directional signal with the index dir of a sub-band group with the index band from the HOA coefficient with the index idx

runLengthCodedVal This run length code determines magnitude difference if a corresponding escape word has been signaled.

HuffCodedAngleDiff This element signals the Huffman coded angle difference using Table **F.x Huffman Table for DecTableAngleDiff**

DecodedAngleDiff[band][dir][hoaIdx] This 3D array holds the decoded angle difference from Table **F.x Huffman Table for DecTableAngleDiff** for the prediction of each directional signal with the index dir of a sub-band group with the index band from the HOA coefficient with the index idx

HuffCodedSbrMagDiff This element signals the Huffman coded magnitude difference using Table **F.x Huffman Table for Decoding HuffmanMagDiffSbr**.

CodedMagDiff This element holds the magnitude difference index quantized with 4 plain bits.

CodedAngleDiff This element holds the angle difference index quantized with 4 plain bits.

12.3.2.x HOAParInfo

This Payload contains data for the PAR decoder.

UsePar This Boolean is true if the PAR decoding is performed in the current frame.

KeepPreviousParMatrixFlag[idx] This Boolean is true if the mixing matrix is kept from the previous frame and false if the mixing matrix is update by new magnitude and angle differences.

ParDecorrSigsSelectionTableIdx[idx] This table index determines the number and the corresponding indices of the de-correlated signals that are used to create one up-mix signal.

NumOfDecorrSigsPerParSubband This element indicates the number of de-correlated signals that are used to create one up-mix signal, which is obtained for UpmixHoaOrderPerParSubband[idx] of two from Table **F.x Table for ParDecorrSigsSelectionTableIdx referring to NumOfDecorrSigsPerParSubbandTable and ParSelectedDecorrSigsIdxMatrixTable** and for UpmixHoaOrderPerParSubband[idx] of one from Table **F.x Table for ParDecorrSigsSelectionTableIdx referring to NumOfDecorrSigsPerFirstOrderParSubbandTable and ParFirstOrderSelectedDecorrSigsIdxMatrixTable** exploiting the index ParDecorrSigsSelectionTableIdx[idx]

ParSelectedDecorrSigsIdxMatrix[sig][idx] This matrix contains the NumOfDecorrSigsPerParSubband indices of the de-correlated signals that are used to create the up-mix signal with the index sig, where the matrix is obtained for UpmixHoaOrderPerParSubband[idx] of two from Table **F.x Table for ParDecorrSigsSelectionTableIdx referring to NumOfDecorrSigsPerParSubbandTable and ParSelectedDecorrSigsIdxMatrixTable** and for

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

UpmixHoaOrderPerParSubband[idx] of one from Table **F.x Table for ParDecorrSigsSelectionTableIdx referring to ParFirstOrderPermIdxVectorTable** exploiting the index ParDecorrSigsSelectionTableIdx[idx].

UseReducedNoOfUpmixSigs	This elements signals that not all up-mix signals are created.
UseParUpmixSig[idx][n]	For each PAR sub-band group of index idx the elements of the Boolean matrix are true if the up-mix signal with an index n is created or false if it is not created.
UseParHuffmanCodingDiffAbs	This Boolean indicates by true that the magnitude differences are Huffman coded.
UseParHuffmanCodingDiffAngle	This Boolean indicates by true that the angle differences are Huffman coded.

The index is equal to the index of the de-correlated signal used to create the current up-mix signal. It is obtained from Table **F.x Table for ParDecorrSigsSelectionTableIdx referring to NumOfDecorrSigsPerParSubbandTable and ParSelectedDecorrSigsIdxMatrixTable** for the current value of **ParDecorrSigsSelectionTableIdx.**

12.3.2.x.1 readParDiffValues()

This function reads magnitude and phase differences of the mixing matrices that are used by the PAR decoder.

HuffCodedParMagDiff	This element signals the Huffman coded magnitude difference using Table **F.x Huffman Table for Decoding ParHuffmanMagDiffNoSbr**
runLengthCodedVal	This run length code determines magnitude difference if a corresponding escape word has been signaled.
DecodedParMagDiff[band][idx][decoIdx]	This 3D array holds the decoded magnitude differences for the prediction of each virtual loudspeaker signal with the index idx of a sub-band group with the index band from the de-correlated signal with the index decoIdx
HuffCodedParAngleDiff	This element signals the Huffman coded angle difference using Table **F.x Huffman Table for ParDecTableAngleDiff**
DecodedParAngleDiff[band][idx][decoIdx]	This 3D array holds the decoded angle differences for the prediction of each virtual loudspeaker signal with the index idx of a sub-band group with the index band from the de-correlated signal with the index idx
HuffmanCodedRealParMagDiff	This element signals the Huffman coded magnitude difference using Table **F.x Huffman Table for Decoding ParHuffmanMagDiffSbr**
CodedParMagDiff	This element holds the coded magnitude difference index quantized with 4 plain bits.
CodedParAngleDiff	This element holds the coded angle difference index quantized with 4 plain bits.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In subclause 12.4.1.2 Global Parameter replace:

D_{PRED} = MaxNoOfDirSigsForPrediction	Maximum number of directional signals used for the prediction of dominant sound sources

With:

D_{PRED} = MaxNoOfDirSigsForPrediction	Maximum number of directional signals used for the prediction of dominant sound sources
$D_{\mathrm{SB}} = \mathrm{MaxNumOfPredDirsPerBand}$	Maximum number of directions per sub-band for Sub-band Directional Signals Synthesis
$\mathrm{D_{MAX}} = \mathrm{MaxNumOfPredDirs};$	Maximum number of different directions over all sub-bands for Sub-band Directional Signals Synthesis

In subclause 12.4.1.2 Global Parameter replace:

r_{max}	Determines the maximum speaker distance of the listening setup.

With:

r_{max}	Determines the maximum speaker distance of the listening setup. In case the actual loudspeaker distances are unknown (hasLoudspeakerDistance == 0) r_{max} is set to infinity (r_{max}=∞).

Add the following subclauses after 12.4.1.2 Global Parameter

12.4.1.2.1 Upper and lower bounds of sub-band groups for Sub-band Directional Signals Synthesis

$\mathrm{B} = \mathrm{NumOfPredSubbands};$

$\mathcal{L}(1) = 1;$

```
for (b=0; b < $\mathrm{NumOfPredSubbands} - 1;$ b++)
{
        $\mathcal{U}(\mathrm{b}+1) = \mathcal{L}(\mathrm{b}) + \mathrm{PredSubbandWidths}[\mathrm{b}] - 1;$
        $\mathcal{L}(\mathrm{b}+1) = \mathcal{U}(\mathrm{b}+1) + 1;$
}
$\mathcal{U}(\mathrm{B}) = \mathcal{L}(\mathrm{B}-1) + \mathrm{PredSubbandWidths}[\mathrm{B}-1] - 1;$
```

12.4.1.2.2 Upper and lower bounds of sub-band groups for PAR

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
$G$ = NumOfParSubbands ;

$\mathcal{L}_{\mathrm{PAR}}(1) = 1$;

for (g=0; g < NumOfParSubbands − 1; g++)
{
        $\mathcal{U}_{\mathrm{PAR}}(g+1) = \mathcal{L}_{\mathrm{PAR}}(g) +$ ParSubbandWidths [g] − 1;
        $\mathcal{L}_{\mathrm{PAR}}(g+1) = \mathcal{U}_{\mathrm{PAR}}(g+1) + 1$;
}
$\mathcal{U}_{\mathrm{PAR}}(G) = \mathcal{L}_{\mathrm{PAR}}(G-1) +$ ParSubbandWidths [$G-1$] − 1;
```

12.4.1.2.3 Orders $\mathbf{N_{PAR}(g)}$ for each sub-band group $\mathbf{g = 1, \dots, G}$ for PAR

```
for (g=0; g < NumOfParSubbands; g++)
{
    $N_{\mathrm{PAR}}(g+1) =$ UpmixHoaOrderPerParSubband[g];
}
```

Add the following subclause after subclause "12.4.1.2 Global Parameter"

12.4.1.x Frame and user dependent parameters

$M_{\mathrm{LAY}}(k)$	Number of all actually used layers for the k-th frame (to be specified) at the decoder side. Note that in the case of layered coding (indicated by SingleLayer==0) this number must be less or equal to the total number of layers present in the bit stream, i.e. $M_{\mathrm{LAY}} \leq$ NumLayers. In the case of single-layered coding (indicated by SingleLayer==1) M_{LAY} is set to one.

Dependent on the choice of $M_{\mathrm{LAY}}(k)$ the number $I_{\mathrm{ADD,LAY}}(k)$ of additional transport channels actually used for spatial HOA decoding (i.e. additional to the O_{MIN} channels that are implicitly always used) is computed as follows:

```
if(SingleLayer | (!SingleLayer & $M_{\mathrm{LAY}}(k)$ == NumLayers))
{
    $I_{\mathrm{ADD,LAY}}(k)$ = NumOfAdditionalCoders;
}
else
{
    $I_{\mathrm{ADD,LAY}}(k)$ = NumHOAChannelsLayer[0] - MinNumOfCoeffsForAmbHOA;
    for (m=1; m < $M_{\mathrm{LAY}}$ ; m++){
        $I_{\mathrm{ADD,LAY}}(k)$ = NumHOACannelsLayer[$M_{\mathrm{LAY}}(k) - 1$] - MinNumOfCoeffsForAmbHOA;
    }
}
```

The number is required to extract from the total side information the part that is relevant for the actually used transport signals. For this reason, in the following, it is used for the conversion of the bit stream parameters to the parameters used in the description of the actual spatial HOA decoding in subclause 12.4.2.

In subclause "12.4.1.5 Assignment vector $\boldsymbol{v}_{AMB,ASSIGN}(\boldsymbol{k})$" replace NumOfAdditionalCoders by $I_{ADD,LAY}(k)$

12.4.1.5 Assignment vector $\boldsymbol{v}_{\mathrm{AMB,ASSIGN}}(\boldsymbol{k})$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
for (i=0; i <$I_{\mathrm{ADD,LAY}}(k)$; i++){
    if(ChannelType[i]==2){
        $v_{\mathrm{AMB,ASSIGN}}(k)$[i+1] = AmbCoeffIdx[i];
    }
    else{
        $v_{\mathrm{AMB,ASSIGN}}(k)$[i+1]=0;
    }
}
nIdx = 1;
for (i= $I_{\mathrm{ADD,LAY}}(k)$; i < NumHOATransportChannels; i++){
    $v_{\mathrm{AMB,ASSIGN}}(k)$[i+1]=nIdx;
    nIdx++;
}
```

In subclause "12.4.1.6 Tuple set $\mathcal{M}_{DIR}(k)$" replace NumOfAdditionalCoders by $I_{ADD,LAY}(k)$

12.4.1.6 Tuple set $\mathcal{M}_{\mathrm{DIR}}(k)$

```
$\mathcal{M}_{\mathrm{DIR}}(k)=\emptyset$;
for (i=0; i < $I_{\mathrm{ADD,LAY}}(k)$; i++){
    if(ChannelType[i]==0){
        $\mathcal{M}_{\mathrm{DIR}}(k)=\mathcal{M}_{\mathrm{DIR}}(k) \cup (i+1, \mathbf{ActiveDirsIds}[i])$;
    }
}
```

In subclause "12.4.1.7 The sets $\mathcal{J}_E(k)$, $\mathcal{J}_D(k)$ and $\mathcal{J}_U(k)$" replace NumOfAdditionalCoders by $I_{ADD,LAY}(k)$

12.4.1.7 The sets $\mathcal{J}_{\mathrm{E}}(k)$, $\mathcal{J}_{\mathrm{D}}(k)$ and $\mathcal{J}_{\mathrm{U}}(k)$

```
$\mathcal{J}_{\mathrm{E}}(k) = \emptyset$;
$\mathcal{J}_{\mathrm{D}}(k) = \emptyset$;
$\mathcal{J}_{\mathrm{U}}(k) = \{1, \ldots O_{MIN}\}$;
for (i=0; i < $I_{\mathrm{ADD,LAY}}(k)$; i++){
    switch(AmbCoeffTransitionState[i]){
        case 0:
        {
$\mathcal{J}_{\mathrm{U}}(k)=\mathcal{J}_{\mathrm{U}}(k) \cup \{\mathrm{AmbCoeffIdx}[i]\}$;
break;
        }
        case 1:
        {
$\mathcal{J}_{\mathrm{E}}(k)=\mathcal{J}_{\mathrm{E}}(k) \cup \{\mathrm{AmbCoeffIdx}[i]\}$;
break;
        }
        case 2:
        {
$\mathcal{J}_{\mathrm{D}}(k)=\mathcal{J}_{\mathrm{D}}(k) \cup \{\mathrm{AmbCoeffIdx}[i]\}$;
        }
    }
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

In subclause 12.4.1.10.2 VVecLength and VVecCoeffId replace:

The size of the Vector codebook depends on the value NumVvecIndices and on the HOA order. If the variable NumVvecIndices is set to 1, the vector codebook containing HOA expansion coefficients derived from Annex F is used. If NumVvecIndices is larger than 1, the Vector codebook with O vector is used in combination with 256x8 weighting values (Table in Annex F.12). For the HOA order 4, the Vector codebook with 32 entries as derived from the Table in Annex F.6 is used.

With:

The size of the Vector codebook depends on the value CodebkIdx(k)[i], on the value NumVvecIndices(k)[i] and on the HOA order. If NumVvecIndices is larger than 1, the 256x8 weighting values (Table in Annex F.12) are used. If NumVvecIndices is larger than 8, the last 2 columns of the 256x8 weighting values (Table in Annex F.12) are used repeatedly with a modular operator.

If the CodebkIdx(k)[i] is set to 0, a codebook containing the HOA expansion coefficients derived from Annex F.9 is used.

If the CodebkIdx(k)[i] is set to 1 the V-vector codebook is generated based on the loudspeaker directions in Annex F.LP and used with scaling. If the CodebkIdx(k)[i] is set to 2, the V-vector codebook based on the loudspeaker directions in Annex F.LP is generated and used without further scaling. If the CodebkIdx(k)[i] is set to 3, the V-vector codebook with 64 entries as shown in the Table in Annex F.2D is used. If the CodebkIdx(k)[i] is set to 7, a vector with O vectors is used. For the HOA order 4, the Vector codebook with 32 entries as derived from the Table in Annex F.6 is used.

Add the following changes concerning Section 12.4.1.10.2 (VVecLength and VVecCoeffId)

The codedVVecLength word indicates:

0) Complete vector length (NumOfHoaCoeffs elements). Indicates that all of the coefficients for the predominant vectors (NumOfHoaCoeffs) are specified.

1) Vector elements 1 to MinNumOfCoeffsForAmbHOA and all elements defined in ContAddHoaCoeff[lay] are not transmitted, where lay is the index of layer containing the vector based signal corresponding to the vector. Indicates that only those coefficients of the predominant vector corresponding to the number greater than a MinNumOfCoeffsForAmbHOA are specified. Further those NumOfContAddAmbHoaChan[lay] coefficients identified in ContAddAmbHoaChan[lay] are subtracted. The list ContAddAmbHoaChan[lay] specifies additional channels corresponding to an order that exceeds the order MinAmbHoaOrder.

2) Vector elements 1 to MinNumOfCoeffsForAmbHOA are not transmitted. Indicates that those coefficients of the predominant vectors corresponding to the number greater than a MinNumOfCoeffsForAmbHOA are specified.

In case of codedVVecLength==1 both the VVecLength[i] array as well as the VVecCoeffId[i][m] 2D array are valid for the VVector of index i, in the other cases both the VVecLength element as well as the VVecCoeffId[m] array are valid for all VVector within the HOAFrame. For the assignment algorithm below a helper function is defined as follows.

```
switch CodedVVecLength{
    case 0:
        VVecLength = NumOfHoaCoeffs;
        for (m=0; m<VVecLength; ++m) {
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
            VVecCoeffId[m] = m;
        }
        break;
    case 1:
        for (i=0; i < NumOfVecSigs; ++i) {
            lay = VecSigLayerIdx[i];
        VVecLength[i] = NumOfHoaCoeffs – MinNumOfCoeffsForAmbHOA –
NumOfContAddHoaChans[lay] - (NewChannelTypeOne[i] * NumOfNewAddHoaChans[lay]);
            CoeffIdx = MinNumOfCoeffsForAmbHOA;
            for (m=0; m<VVecLength[i]; ++m) {
                CoeffIdx++;
                if(NewChannelTypeOne[i]) {
                    bIsInArray = ( isMemberOf(CoeffIdx, ContAddHoaCoeff[lay],
NumOfContAddHoaChans[lay]) || isMemberOf(CoeffIdx, NewAddHoaChans[lay],
NumOfNewAddHoaChans[lay]) );
                    while (bIsInArray) {
                        CoeffIdx++;
                        bIsInArray = ( (isMemberOf(CoeffIdx, ContAddHoaCoeff[lay],
NumOfContAddHoaChans[lay]) || isMemberOf(CoeffIdx, NewAddHoaChans[lay],
NumOfNewAddHoaChans[lay]) );
                    }
                }
                else{
                bIsInArray = isMemberOf(CoeffIdx, ContAddHoaCoeff[lay],
                    NumOfContAddHoaChans[lay]);
                while (bIsInArray) {
                    CoeffIdx++;
                    bIsInArray = isMemberOf(CoeffIdx, ContAddHoaCoeff[lay],
                        NumOfContAddHoaChans[lay]);
                }
            }
            VVecCoeffId[i][m] = CoeffIdx-1;
            }
        }
        break;
    case 2:
        VVecLength = NumOfHoaCoeffs – MinNumOfCoeffsForAmbHOA;
        for (m=0; m< VVecLength; ++m) {
            VVecCoeffId[m] = m + MinNumOfCoeffsForAmbHOA;
        }
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Add the grey marked changes for the pseudo code given in Section 12.4.1.10.5 (Conversion of VVec element)

```
if (CodedVVecLength == 1) {
    VVecLengthUsed = VVecLength[i];
    VVecCoeffIdUsed = VVecCoeffId[i];
} else {
    VVecLengthUsed = VVecLength;
    VVecCoeffIdUsed = VVecCoeffId;
}
for (m=0; m<O; ++m) {
    $v^{(i)}{}_{m}(k)$ = 0;
}

if (NbitsQ(k)[i] == 4) {
        FNorm = 1.0;
for (m=0; m< O; ++m) {
        TmpVVec[m] = 0;
            for (j=0; j< NumVvecIndices(k)[i]; ++j) {
            TmpVVec[m]+=WeightVal[j]*VecDict[CodebkIdx].[VvecIdx[j]][m];
```

© ISO/IEC 2017 – All rights reserved

```
            }
        }
    if (doScaling) {
        FNorm = 0.0;
            for (m=0; m<O; ++m) {
                FNorm += TmpVVec[m] * TmpVVec[m];
            }
        }
        for (m=0; m< VVecLengthUsed; ++m) {
            idx = VVecCoeffIDUsed[m];
            $v^{(i)}{}_{\mathrm{idx}}(k)$ = TmpVVec[idx] * (N+1)/sqrt(FNorm);
        }
    }
    elseif (NbitsQ(k)[i] == 5) {
        for (m=0; m< VVecLengthUsed; ++m) {
           $v^{(i)}{}_{\mathrm{VVecCoeffIdUsed[m]}}(k)$ = (N+1)*aVal[i][m];
        }
    }
    elseif (NbitsQ(k)[i] >= 6) {
        for (m=0; m< VVecLengthUsed; ++m) {
           $v^{(i)}{}_{\mathrm{VVecCoeffIdUsed[m]}}(k)$ = (N+1) * (2^(16 – NbitsQ(k)[i])*aVal[i][m])/2^15;
            if (PFlag(k)[i] == 1) {
                $v^{(i)}{}_{\mathrm{VVecCoeffIdUsed[m]}}(k)$ += $floor\left(0.5 + v^{(i)}{}_{\mathrm{VVecCoeffId[m]}}(k-1) * 2^{14}\right) * 2^{-14}$;
            }
        }
    }
    if ( (CodedVVecLength == 1) & (($M_{\mathrm{LAY}}(k) - 1$) > VecSigLayerIdx[i]) ) {
        for (m=0; m< VVecLengthUsed; ++m) {
            CoeffIdx = VVecCoeffIdUsed[m];
            if (isMemberOf(CoeffIdx, ContAddHoaCoeff[$M_{\mathrm{LAY}}(k) - 1$],
                        NumOfContAddHoaChans[$\mathrm{M_{LAY}(k) - 1}$]) ) {
                $v^{(i)}{}_{\mathrm{CoeffIdx}}(k)$= 0;
            }
        }
    }
```

Add before subclause 12.4.1.10.6 Tuple set $\boldsymbol{\mathcal{M}_{VEC}(k)}$:

selectCodebk

```
switch CodebkIdx {
    case 0:
        cdbLen = 900;
        nbitsIdx = 10;
        doScaling = 1;
        break;
    case 1:
        cdbLen = 34;
        nbitsIdx = 6;
        doScaling = 1;
        break;
    case 2:
        cdbLen = 34;
        nbitsIdx = 6;
        doScaling = 0;
        break;
    case 3:
        cdbLen = 64;
        nbitsIdx = 6;
        doScaling = 1;
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
        break;
    case 7:
        cdbLen = (N+1)*(N+1);
        if (N==4){
            cdbLen = 32;
        }
        nbitsIdx = ceil(log2(NumOfHoaCoeffs));
        doScaling = 1;
}
```

In subclause "12.4.1.10.6 Tuple set $\boldsymbol{\mathcal{M}_{VEC}(k)}$" replace NumOfAdditionalCoders by $I_{ADD,LAY}$ (k)

12.4.1.10.6 Tuple set $\mathcal{M}_{\mathrm{VEC}}(\boldsymbol{k})$

```
$\mathcal{M}_{\mathrm{VEC}}(k)=\emptyset$;
for (i=0; i < $I_{\mathrm{ADD,LAY}}(k)$; i++){
    if(ChannelType[i]==1){
        $\mathcal{M}_{\mathrm{VEC}}(k)=\mathcal{M}_{\mathrm{VEC}}(k) \cup (i+1,\ \boldsymbol{v}^{(i)}(k))$;
    }
}
```

Add the following subclauses after 12.4.1.10.6 Tuple set

12.4.1.x Tuple sets $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(\boldsymbol{k},\boldsymbol{b})$ for Sub-band Directional Signals Synthesis

```
$\mathrm{Q_{VAR}} = \mathrm{NumOfGridPointsTable}$;

for (b=0; b < NumOfPredSubbands; b++)
{
    $\tilde{\mathcal{J}}_{\mathrm{DIR}}(\mathrm{k,b}) = \emptyset$;
    if(KeepPreviousDirPredMatrixFlag[b])
    {
        $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(\mathrm{k,b}) = \widetilde{\mathcal{M}}_{\mathrm{DIR}}(\mathrm{k-1,b})$;
    }
    else
    {
        $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(\mathrm{k,b}) = \emptyset$;
        for (d=0; d < MaxNumOfPredDirsPerBand; d++)
        {
            if (DirIsActive[b][d])
            {
                $\boldsymbol{\Omega}_{\mathrm{SB,d}}(\mathrm{k-1,b}) = \boldsymbol{\Omega}^{(\sqrt{\mathrm{Q_{VAR}}}-1)}_{\mathrm{PredDirGridIdx[b][d]}}$ ; // according to tables F.9 – F.11
                $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(\mathrm{k,b}) = \widetilde{\mathcal{M}}_{\mathrm{DIR}}(\mathrm{k,b}) \cup \{ (d+1,\ \boldsymbol{\Omega}_{\mathrm{SB,d}}(k-1,b))\}$;
            }
        }
    }
}
```

12.4.1.x Sub-band prediction indicator $\tilde{b}_{\mathrm{SBP}}(k)$ for Sub-band Directional Signals Synthesis

$\tilde{b}_{\mathrm{SBP}}(k) = \mathrm{UseDirectionalPrediction}$ **;**

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

12.4.1.x Prediction coefficient matrices $\widetilde{A}(k,b)$ for Sub-band Directional Signals Synthesis

Since the prediction coefficient matrix elements are coded differentially, before starting to decode/convert the elements for a k-th independency frame it is necessary to initialize the quantized values to zero for the previous frame as follows:

```
if (hoaIndependencyFlag(k) )
{
    for (b=0; b < NumOfPredSubbands; b++)
    {
        for (d=0; d < MaxNumOfPredDirsPerBand; d++)
        {
            for (n=0; n < MaxNumOfCoeffsToBeTransmitted; n++)
            {
                $\mathrm{IntQuantMag(k-1)[b][d][n]} = 0$;
                $\mathrm{IntQuantAngle(k-1)[b][d][n]} = 0$;
            }
        }
    }
}
```

The actual decoding/conversion is assumed to be performed as follows:

```
for (b=0; b < NumOfPredSubbands; b++)
{
    if((KeepPreviousDirPredMatrixFlag[$b$] != 0) && !hoaIndependencyFlag)
    {
        $\widetilde{\boldsymbol{A}}(\mathrm{k,b}) = \widetilde{\boldsymbol{A}}(\mathrm{k}-1,\mathrm{b})$;
        for (d=0; d < MaxNumOfPredDirsPerBand ; d++){
            for (n=0; n < MaxNumOfCoeffsToBeTransmitted; n++){
                $\mathrm{IntQuantMag}(k)\mathrm{[b][d][n]} = \mathrm{IntQuantMag}(k-1)\mathrm{[b][d][n]}$;
                $\mathrm{IntQuantAngle}(k)\mathrm{[b][d][n]} = \mathrm{IntQuantAngle}(k-1)\mathrm{[b][d][n]}$;
            }
        }
    }
    else
    {
        for (d=0; d < MaxNumOfPredDirsPerBand; d++)
        {
            if (DirIsActive[b][d])
            {
                for (n=0; n < MaxNumOfCoeffsToBeTransmitted; n++)
                {
                    if( $(n+1) \in \mathcal{I}_{\mathrm{E}}(k) \cup \mathcal{I}_{\mathrm{D}}(k) \cup \mathcal{I}_{\mathrm{U}}(k)$ ){
                        $\mathrm{IntQuantMag(k)[b][d][n]} = \mathrm{IntQuantMag(k-1)[b][d][n]}$
                                                  $+ \mathrm{DecodedMagDiff[b][d][n]}$;
                        if(IntQuantMag(k)[b][d][n] == 0){
                            IntQuantAngle(k)[b][d][n] = 0;
                        }
                        else{
                            $\mathrm{IntQuantAngle(k)[b][d][n]} = \mathrm{IntQuantAngle(k-1)[b][d][n]}$
                                                  $+ \mathrm{DecodedAngleDiff[b][d][n]}$;
                            // constrain the quantized angle to lie in interval [-7,...,8]
                            $\mathrm{IntQuantAngle(k)[b][d][n]} = \left((\mathrm{IntQuantAngle(k)[b][d][n]} + 7)\mathrm{mod}16\right) - 7$;
                        }
                        if (IntQuantMag(k)[b][d][n] > 8){
                            $\mathrm{FloatMag} = \left(\frac{8}{7}\right)^{(\mathrm{IntQuantMag(k)[b][d][n]}-8)}$;
                        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
                            else{
                                FloatMag = $\mathrm{IntQuantMag}(k)[b][d][n] \cdot 1/8$
                            }
                            FloatAngle = $\mathrm{IntQuantAngle}(k)[b][d][n] \cdot \pi/8$;
                            $\tilde{\boldsymbol{A}}(k,b)[d+1][n+1]$ = FloatMag $\cdot$ exp( i $\cdot$ FloatAngle);
                        }
                        else{
                            $\tilde{\boldsymbol{A}}(k,b)[d+1][n+1] = 0$;
                            $\mathrm{IntQuantMag}(k)[b][d][n] = 0$;
                            $\mathrm{IntQuantAngle}(k)[b][d][n] = 0$;
                        }
                    }
                }
                else
                {
                    for (n=0; n < MaxNumOfCoeffsToBeTransmitted; n++)
                    {
                        $\mathrm{IntQuantMag}(k)[b][d][n] = 0$;
                        $\mathrm{IntQuantAngle}(k)[b][d][n] = 0$;
                        $\tilde{\boldsymbol{A}}(k,b)[d+1][n+1] = 0$;
                    }
                }
            }
        }
    }
```

12.4.1.x Mixing matrices $\mathbf{M}_{\mathrm{PAR}}(k,g)$ for Parametric Ambience Replication

Since the mixing matrix elements are coded differentially, before starting to decode/convert the elements for a k-th independency frame it is necessary to initialize the quantized values to zero for the previous frame as follows:

```
if (hoaIndependencyFlag(k) )
{
    for (g=0; g < NumOfParSubbands; g++)
    {
        for (d=0; d < MaxNumOfDecoSigs(g); d++)
        {
            for (n=0; n < MaxNumOfDecoSigs(g); n++)
            {
                $\mathrm{IntQuantMagPAR}(k-1)[g][d][n] = 0$;
                $\mathrm{IntQuantAnglePAR}(k-1)[g][d][n] = 0$;
            }
        }
    }
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

The actual decoding/conversion is assumed to be performed as follows:

```
for ($g$=0; $g <$ NumOfParSubbands; $g$++)
{
    if((KeepPreviousParMatrixFlag[$g$] != 0)  && !hoaIndependencyFlag)
    {
        $\widetilde{\boldsymbol{M}}_{\mathrm{PAR}}(k,g) = \widetilde{\boldsymbol{M}}_{\mathrm{PAR}}(k-1,g)$;
        for (d=0; d < MaxNumOfDecoSigs(g); d++){
            for (n=0; n < MaxNumOfDecoSigs(g); n++){
                IntQuantParMag($k$)[g][d][n ] = IntQuantParMag($k-1$)[g][d][n ];
                IntQuantParAngle($k$)[g][d][n ]  = IntQuantParAngle($k-1$)[g][d][n ];
            }
        }
    }
    else
    {
        $\mathrm{N}_{\mathrm{SIG}}(k,g)$ =  NumOfDecorrSigsPerParSubbandTable
                                [ParDecorrSigsSelectionTableIdx[$g$]];
      CurrParSelectedDecorrSigsIdxMatrix = ParSelectedDecorrSigsIdxMatrixTable
                                                        [ParDecorrSigsSelectionTableIdx[$g$]];
        for (d=0; d < MaxNumOfDecoSigs($g$); d++)
        {
            for (n=0; n < MaxNumOfDecoSigs($g$); n++){
                    IntQuantParMag(k)[g][d][n ] = 0;
                    IntQuantParAngle(k)[g][d][n ]  = 0;
                    $\mathrm{M}_{\mathrm{PAR}}(\boldsymbol{k},\boldsymbol{g})$[d + 1][n + 1] = 0;
            }
            if(UseParUpmixSig[$g$][$d$] != 0)
            {
                for (n=0; n < $\mathrm{N}_{\mathrm{SIG}}$(k, g); n++)
                {
                    CurrColIdx = CurrParSelectedDecorrSigsIdxMatrix [d][n];
                    IntQuantParMag(k)[g][d][CurrColIdx ] =  IntQuantParMag(k − 1)[g][d][CurrColIdx ]
                                + DecodedParMagDiff[g][d][CurrColIdx ];
                    if(IntQuantParMag(k)[g][d][CurrColIdx ]  == 0)
                    {
                        IntQuantParAngle(k)[g][d][CurrColIdx ]  = 0;
                    }
                    else
                    {
                        IntQuantParAngle(k)[g][d][CurrColIdx ] =
                            IntQuantParAngle(k − 1)[b][g][d][CurrColIdx ]
                            + DecodedParAngleDiff[b][d][CurrColIdx ];
                        // constrain the quantized angle to lie in interval [-7,...,8]
                        IntQuantParAngle(k)[g][d][CurrColIdx ] =
                                ((IntQuantParAngle(k)[g][d][CurrColIdx ] + 7)mod16 ) − 7;
                    }
                    if (IntQuantParMag(k)[b][d][n] > 8)
                    {
                        FloatParMag = $\left(\frac{8}{7}\right)^{(\mathrm{IntQuantParMag(k)[b][d][n]}-8)}$;
                    }
                    else
                    {
                        FloatParMag =  IntQuantParMag(k)[b][d][n] · 1/8
                    }
                    FloatParAngle  =  IntQuantParAngle(k)[b][d][CurrColIdx ] · π/8 ;
                    $\boldsymbol{M}_{\mathrm{PAR}}(k,g)$[d + 1][CurrColIdx  + 1] =  FloatParMag · exp( i ·  FloatParAngle);
            }
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
        }
    }
}
```

12.4.1.x Permutation matrices $\boldsymbol{P}_{\mathrm{PAR}}(\boldsymbol{k},\boldsymbol{g})$ for PAR

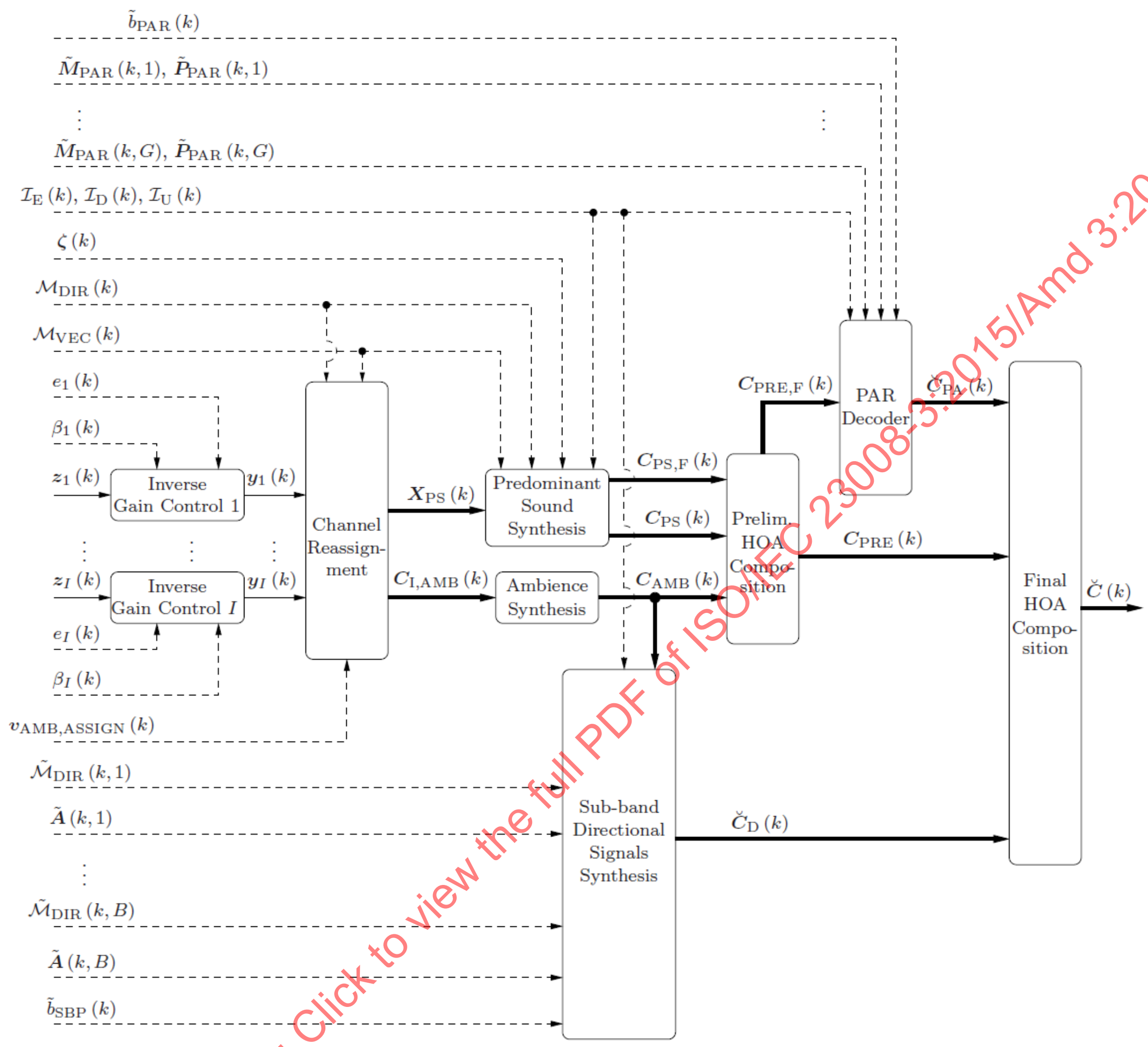
```
for (g=0; g < LastFirstOrderSubBandIdx; g++)
{
        $\mathrm{N_{SIG}(k,g)}$ = NumOfDecorrSigsPerFirstOrderParSubbandTable
                                [ParDecorrSigsSelectionTableIdx[$g$]];
        CurrParPermIdxVector =
                            ParFirstOrderPermIdxVectorTable[ParDecorrSigsSelectionTableIdx[g]];
        for (d=0; d < MaxNumOfDecoSigs(g); d++)
        {
            for (n=0; n < MaxNumOfDecoSigs(g); n++){
                $\boldsymbol{P}_{\mathrm{PAR}}(k,g)[\mathrm{n}+1][\mathrm{d}+1] = 0$;
            }
            $\boldsymbol{P}_{\mathrm{PAR}}(k,g)$[CurrParPermIdxVector[d]+1][d + 1]= 1;
        }
}
for (g= LastFirstOrderSubBandIdx; g < NumOfParSubbands; g++)
{
        $\mathrm{N_{SIG}(k,g)}$ = NumOfDecorrSigsPerParSubbandTable
                                [ParDecorrSigsSelectionTableIdx[$g$]];
        CurrParPermIdxVector =
                            ParPermIdxVectorTable[ParDecorrSigsSelectionTableIdx[g]];
        for (d=0; d < MaxNumOfDecoSigs(g); d++)
        {
            for (n=0; n < MaxNumOfDecoSigs(g); n++){
                $\boldsymbol{P}_{\mathrm{PAR}}(k,g)[\mathrm{n}+1][\mathrm{d}+1] = 0$;
            }
            $\boldsymbol{P}_{\mathrm{PAR}}(k,g)$[CurrParPermIdxVector[d]+1][d + 1]= 1;
        }
}
```

12.4.1.x Indicator $\tilde{b}_{\mathrm{PAR}}(k)$ for use of PAR

$\tilde{b}_{\mathrm{PAR}}(k) = \mathrm{UsePAR}$ **;**

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Replace Figure 40 — Architecture of spatial HOA decoder with:



$\tilde{b}_{\mathrm{PAR}}(k)$
$\tilde{M}_{\mathrm{PAR}}(k,1)$, $\tilde{P}_{\mathrm{PAR}}(k,1)$
$\tilde{M}_{\mathrm{PAR}}(k,G)$, $\tilde{P}_{\mathrm{PAR}}(k,G)$
$\mathcal{I}_{\mathrm{E}}(k)$, $\mathcal{I}_{\mathrm{D}}(k)$, $\mathcal{I}_{\mathrm{U}}(k)$
$\zeta(k)$
$\mathcal{M}_{\mathrm{DIR}}(k)$
$\mathcal{M}_{\mathrm{VEC}}(k)$
$e_1(k)$
$\beta_1(k)$
$z_1(k)$
Inverse Gain Control 1
$y_1(k)$
$z_I(k)$
Inverse Gain Control I
$y_I(k)$
$e_I(k)$
$\beta_I(k)$
$v_{\mathrm{AMB,ASSIGN}}(k)$
Channel Reassign-ment
$X_{\mathrm{PS}}(k)$
$C_{\mathrm{I,AMB}}(k)$
Predominant Sound Synthesis
Ambience Synthesis
$C_{\mathrm{PS,F}}(k)$
$C_{\mathrm{PS}}(k)$
$C_{\mathrm{AMB}}(k)$
Prelim. HOA Compo-sition
$C_{\mathrm{PRE,F}}(k)$
$C_{\mathrm{PRE}}(k)$
PAR Decoder
$\check{C}_{\mathrm{PA}}(k)$
Final HOA Compo-sition
$\check{C}(k)$
$\tilde{\mathcal{M}}_{\mathrm{DIR}}(k,1)$
$\tilde{A}(k,1)$
$\tilde{\mathcal{M}}_{\mathrm{DIR}}(k,B)$
$\tilde{A}(k,B)$
$\tilde{b}_{\mathrm{SBP}}(k)$
Sub-band Directional Signals Synthesis
$\check{C}_{\mathrm{D}}(k)$

In subclause 12.4.2.1 General Architecture replace:

In the Predominant Sound Synthesis processing block the HOA representation of the predominant sound component $\boldsymbol{C}_{\mathrm{PS}}(k)$ is computed from the frame $\boldsymbol{X}_{\mathrm{PS}}(k)$ of all predominant sound signals. It uses the tuple sets $\mathcal{M}_{\mathrm{DIR}}(k)$ and $\mathcal{M}_{\mathrm{VEC}}(k)$, the set $\boldsymbol{\zeta}(k)$ of prediction parameters and the sets $\mathcal{I}_{\mathrm{E}}(k)$, $\mathcal{I}_{\mathrm{D}}(k)$, $\mathcal{I}_{\mathrm{U}}(k)$ of coefficient indices of the ambient HOA component, which have to be enabled, disabled and to remain active in the $(k)^{th}$ frame.

In the Ambience Synthesis processing block, the ambient HOA component frame $\boldsymbol{C}_{\mathrm{AMB}}(k)$ is created from the frame $\boldsymbol{C}_{\mathrm{I,AMB}}(k)$ of the intermediate representation of the ambient HOA component. Note that this processing also comprises an inverse spatial transform to invert the spatial transform applied in the encoder (see Annex 0) for decorrelating the first O_{MIN} coefficients of the ambient HOA component.

Finally, in the HOA Composition processing block the ambient HOA component frame $\boldsymbol{C}_{\mathrm{AMB}}(k)$ and the frame $\boldsymbol{C}_{\mathrm{PS}}(k)$ of the predominant sound HOA component are superimposed to provide the decoded HOA frame $\boldsymbol{C}(k)$.

In the following, the individual processing blocks are described in more detail.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

With:

In the Predominant Sound Synthesis processing block the HOA representation of the predominant sound component $\boldsymbol{C}_{\mathrm{PS}}(k)$ is computed from the frame $\boldsymbol{X}_{\mathrm{PS}}(k)$ of all predominant sound signals. It uses the tuple sets $\mathcal{M}_{\mathrm{DIR}}(k)$ and $\mathcal{M}_{\mathrm{VEC}}(k)$, the set $\boldsymbol{\zeta}(k)$ of prediction parameters and the sets $\mathcal{I}_{\mathrm{E}}(k)$, $\mathcal{I}_{\mathrm{D}}(k)$, $\mathcal{I}_{\mathrm{U}}(k)$, which contain indices of coefficient sequences of the ambient HOA component, which are to be enabled, to be disabled and active but not be enabled or disabled, respectively. Additionally, a modified version $\boldsymbol{C}_{\mathrm{PS,F}}(k)$ of $\boldsymbol{C}_{\mathrm{PS}}(k)$ is computed by fading in coefficient sequences with indices contained in the index set $\mathcal{I}_{\mathrm{E}}(k)$ and fading out coefficient sequences with indices contained in the index set $\mathcal{I}_{\mathrm{D}}(k)$. The modified version is only needed for the later computation of the modified version $\boldsymbol{C}_{\mathrm{PRE,F}}(k)$ of the preliminary decoded HOA representation (see Sec. 1.5) to be input to the PAR decoder.

In the Ambience Synthesis processing block, the ambient HOA component frame $\boldsymbol{C}_{\mathrm{AMB}}(k)$ is created from the frame $\boldsymbol{C}_{\mathrm{I,AMB}}(k)$ of the intermediate representation of the ambient HOA component. Note that this processing also comprises an inverse spatial transform to invert the spatial transform applied in the encoder (see Annex 0) for decorrelating the first O_{MIN} coefficients of the ambient HOA component.

The ambient HOA component frame $\boldsymbol{C}_{\mathrm{AMB}}(k)$ and the frame $\boldsymbol{C}_{\mathrm{PS}}(k)$ of the predominant sound HOA component are superposed in the Preliminary HOA Composition processing block to provide the frame $\boldsymbol{C}_{\mathrm{PRE}}(k)$ of the preliminary decoded HOA representation. Additionally, the frame $\boldsymbol{C}_{\mathrm{PRE,F}}(k)$ of a modified version of the preliminary decoded HOA representation is computed by replacing the frame $\boldsymbol{C}_{\mathrm{PS}}(k)$ by its modified version $\boldsymbol{C}_{\mathrm{PS,F}}(k)$ for the superposition. The resulting modified HOA representation $\boldsymbol{C}_{\mathrm{PRE,F}}(k)$ is then successively input into the PAR decoder instead of the original version $\boldsymbol{C}_{\mathrm{PRE}}(k)$ to avoid signal discontinuities after performing on $\boldsymbol{C}_{\mathrm{PRE,F}}(k)$ the truncation and coefficient selection (see Sec. 0).

Finally, in the HOA Composition processing block the ambient HOA component frame $\boldsymbol{C}_{\mathrm{AMB}}(k)$ and the frame $\boldsymbol{C}_{\mathrm{PS}}(k)$ of the predominant sound HOA component are superimposed to provide the decoded HOA frame $\boldsymbol{C}(k)$.

In the Sub-band Directional Signals Synthesis processing block the frame $\breve{\boldsymbol{C}}_{\mathrm{D}}(k)$ of the HOA representation of the composition of all predicted sub-band directional signals is computed. Each directional sub-band signal is assumed to be predicted by a complex valued weighted sum of the transmitted coefficient sequences of the ambient HOA component $\boldsymbol{C}_{\mathrm{AMB}}(k)$, where the indices of the transmitted coefficient sequences are assumed to be among the first O_{MAX}. The prediction of each directional signal related to the j-th sub-band, $j = 1, \ldots, F$, belonging to the b-th sub-band group is carried out using the prediction coefficients matrix $\mathbf{A}(k,b) \in \mathbb{C}^{D_{\mathrm{SB}} \times O_{\mathrm{MAX}}}$ and the tuple set $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k,b)$. The F assumed sub-bands are uniquely assigned to B sub-band groups, which are determined by the sub-band group configuration specified in the HOAConfig(). It defines for each b-th sub-band group a lower index bound $\mathcal{L}(b)$ and an upper index bound $\mathcal{U}(b)$ such that sub-bands with indices between these bounds, i.e. with $\mathcal{L}(b) \le j \le \mathcal{U}(b)$, are assumed to belong to this sub-band group.

Per sub-band group there are at most D_{SB} potential active direction trajectories, where the indices identifying the active direction trajectories for the b-th sub-band group are assumed to be contained in the set $\tilde{\mathcal{J}}_{\mathrm{DIR}}(k,b) \subseteq \{1, \ldots, D_{\mathrm{SB}}\}$. For each index $d \in \tilde{\mathcal{J}}_{\mathrm{DIR}}(k,b)$ of an active direction trajectory the respective direction is denoted by $\boldsymbol{\Omega}_{\mathrm{SB},d}(k,b)$, both of which are assumed to be contained as tuples in the set $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k,b)$, i.e.

$$\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k,b) = \left\{ \left(d, \boldsymbol{\Omega}_{\mathrm{SB},d}(k,b) \right) \middle| \, d \in \tilde{\mathcal{J}}_{\mathrm{DIR}}(k,b) \right\}.$$

Note that the set $\tilde{\mathcal{J}}_{\mathrm{DIR}}(k,b)$ can be computed from $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k,b)$, since it contains the first elements of all tuples of $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k,b)$.

In order to avoid artifacts in the predicted directional sub-band signals due to changes of the estimated directions and prediction coefficients between successive frames, the prediction is performed on concatenated long frames consisting of two temporally successive frames. In particular, that means that each quantity $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k,b)$ and $\mathbf{A}(k,b)$ is related to the $(k+1)$-th and k-th frame. The quantity $\tilde{b}_{\mathrm{SBP}}(k)$ (which can be equal to zero or one) indicates if a prediction of sub-band directional signals is to be performed related to the frames k and $k+1$ at all.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

The frame $\breve{\boldsymbol{C}}_{\mathrm{D}}(k)$ is assumed to have only non-zero contributions for those coefficient sequences of the ambient HOA component that are not already transmitted within the transport channels. Further, if coefficient sequences of the ambient HOA component are faded in (or faded out respectively), the corresponding coefficient sequences of the HOA representation $\breve{\boldsymbol{C}}_{\mathrm{D}}(k)$ are faded out (or faded in respectively).

Note further that the time domain coefficient sequences of the HOA representation $\breve{\boldsymbol{C}}_{\mathrm{D}}(k)$ of the composition of all predicted sub-band directional signals receive a delay of $D_{\mathrm{QMF}} = 577$ samples due to the successive application of the QMF based analysis and synthesis filter banks, which is expressed by the breve symbol above the variables.

Within the Parametric Ambience Replication (PAR) Decoder processing block ambient components, which are potentially still missing within the preliminary decoded HOA representation $\boldsymbol{C}_{\mathrm{PRE}}(k)$, are parametrically replicated from the modified version $\boldsymbol{C}_{\mathrm{PRE,F}}(k)$ of it. The replication is carried out in the frequency domain using Quadrature Mirror Filters (QMF) with $F = 64$ sub-bands (see ISO/IEC 23003-1:2007, *Information technology — MPEG audio technologies — Part 1: MPEG Surround*). Each individual sub-band j, $j = 1, \dots, F$, is processed using the corresponding parameters of the g-th sub-band group, $g = 1, \dots, G$, to which it is uniquely assigned. The assignment is determined by the PAR related sub-band group configuration specified in the HOAConfig(). It defines for each g-th sub-band group a lower index bound $\mathcal{L}_{\mathrm{PAR}}(g)$ and an upper index bound $\mathcal{U}_{\mathrm{PAR}}(g)$ such that sub-bands with indices between these bounds, i.e. with $\mathcal{L}_{\mathrm{PAR}}(b) \leq j \leq \mathcal{U}_{\mathrm{PAR}}(b)$, are assumed to belong to this sub-band group. The PAR related side information for the k-th frame consists of the mixing matrices $\boldsymbol{M}_{\mathrm{PAR}}(k,g)$ and the permutation matrices $\boldsymbol{P}_{\mathrm{PAR}}(k,g)$ for the individual G sub-band groups $g = 1, \dots, G$, as well as the quantity $\tilde{b}_{\mathrm{PAR}}(k)$, which indicates (by a zero or one) whether PAR is to be performed for the frames k and $k+1$ at all.

In a last step, in the Final HOA Composition processing block the frame $\boldsymbol{C}_{\mathrm{PRE}}(k)$ of the preliminary decoded HOA representation, the frame $\breve{\boldsymbol{C}}_{\mathrm{D}}(k)$ of the HOA representation of the composition of all predicted sub-band directional signals and the frame $\breve{\boldsymbol{C}}_{\mathrm{PA}}(k)$ of the replicated ambient HOA component are superposed to provide the frame $\breve{\boldsymbol{C}}(k)$ of the decoded HOA representation, considering the delay between the individual HOA representations to be superposed.

In the following, the individual processing blocks are described in more detail.

Add at the end of subclause 12.4.2.2 Channel Reassignment:

Note that for the case that the flag UsePhaseShiftDecorr has a value of 1, it is assumed that the frame $\boldsymbol{C}_{\mathrm{I,AMB}}(k+1)$ instead of $\boldsymbol{C}_{\mathrm{I,AMB}}(k)$ is obtained from the transport channels, which has to be ensured by a respective modification at the spatial HOA encoding stage.

In subclause 12.4.2.4.1 General replace:

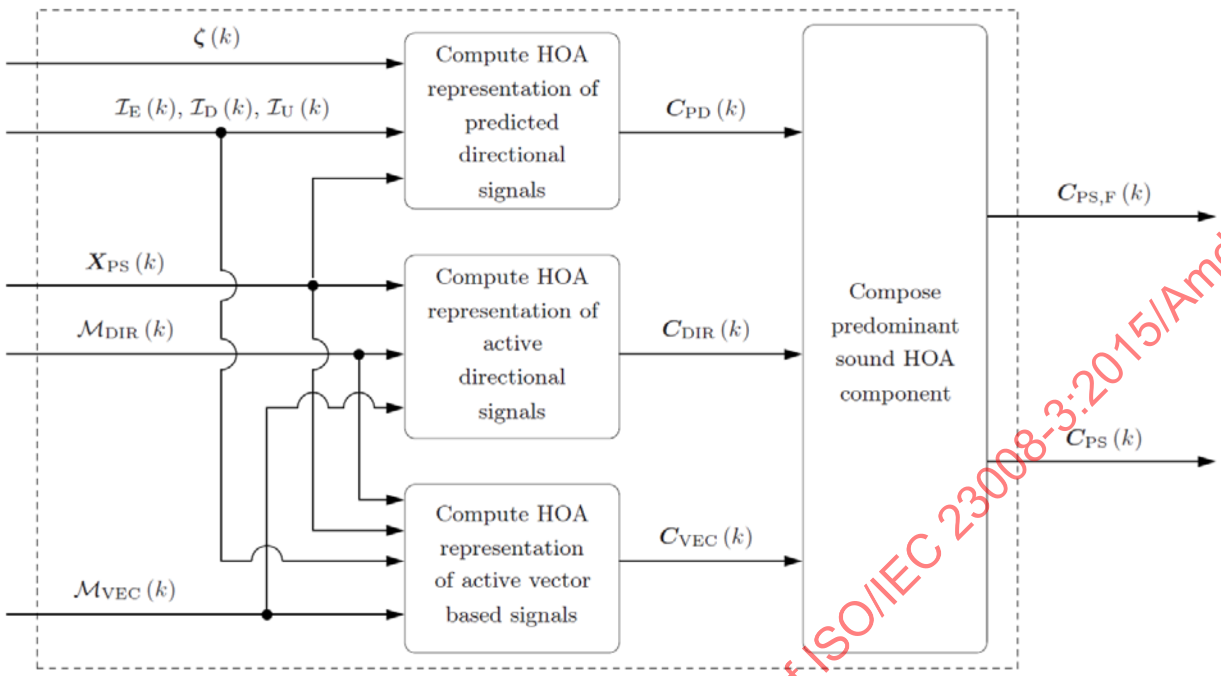
As illustrated in Figure 41, the processing can be subdivided into four processing steps, which are described in the following.

With:

Additionally, the frame $\boldsymbol{C}_{\mathrm{PS,F}}(k)$ of a modified version of $\boldsymbol{C}_{\mathrm{PS}}(k)$ is computed, where the modification only consist of fading in coefficient sequences with indices contained in the index set $\mathcal{I}_{\mathrm{E}}(k)$ and fading out coefficient sequences with indices contained in the index set $\mathcal{I}_{\mathrm{D}}(\mathrm{k})$. The modified version is only needed for the later computation of the modified version $\mathrm{C}_{\mathrm{PRE,F}}(\mathrm{k})$ of the preliminary decoded HOA representation (see Sec. 0) to be input to the PAR decoder. As illustrated in Figure 41, the processing can be subdivided into four processing steps, which are described in the following.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Replace Figure 41 — Predominant sound synthesis with:



$\zeta(k)$
$\mathcal{I}_{\mathrm{E}}(k)$, $\mathcal{I}_{\mathrm{D}}(k)$, $\mathcal{I}_{\mathrm{U}}(k)$
$\boldsymbol{X}_{\mathrm{PS}}(k)$
$\mathcal{M}_{\mathrm{DIR}}(k)$
$\mathcal{M}_{\mathrm{VEC}}(k)$
Compute HOA representation of predicted directional signals
Compute HOA representation of active directional signals
Compute HOA representation of active vector based signals
$\boldsymbol{C}_{\mathrm{PD}}(k)$
$\boldsymbol{C}_{\mathrm{DIR}}(k)$
$\boldsymbol{C}_{\mathrm{VEC}}(k)$
Compose predominant sound HOA component
$\boldsymbol{C}_{\mathrm{PS,F}}(k)$
$\boldsymbol{C}_{\mathrm{PS}}(k)$

In subclause 12.4.2.4.2 Compute HOA representation of active directional signals replace:

The sample values of the faded out and faded in directional HOA components are then determined by

$$c_{\mathrm{DIR,OUT},i}(k,l) = \sum_{d\in\mathcal{J}_{\mathrm{DIR,NZ}}(k-1)} c_{\mathrm{DIR,I},i}^{(d)}(k-1;k,l)\cdot\begin{cases} w_{\mathrm{DIR}}(L+l) & \text{if } d\in\mathcal{J}_{\mathrm{DIR}}(k) \\ w_{\mathrm{VEC}}(L+l) & \text{if } d\in\mathcal{J}_{\mathrm{VEC}}(k) \\ 1 & \text{else}\end{cases}$$

$$c_{\mathrm{DIR,IN},i}(k,l) = \sum_{d\in\mathcal{J}_{\mathrm{DIR,NZ}}(k)} c_{\mathrm{DIR,I},i}^{(d)}(k;k,l)\cdot\begin{cases} w_{\mathrm{DIR}}(l) & \text{if } d\in\mathcal{J}_{\mathrm{DIR}}(k-1)\cup\mathcal{J}_{\mathrm{VEC}}(k-1) \\ 1 & \text{else}\end{cases},$$

where $\mathcal{J}_{\mathrm{DIR,NZ}}(k)$ denotes the set of those first elements of $\mathcal{M}_{\mathrm{DIR}}(k)$ where the corresponding second element is non-zero.

With:

In the case of single-layered coding (indicated by SingleLayer==1 (see Table 120)), the sample values of the faded out and faded in directional HOA components are then determined by

$$c_{\mathrm{DIR,OUT},i}(k,l) = \sum_{d\in\mathcal{J}_{\mathrm{DIR,NZ}}(k-1)} c_{\mathrm{DIR,I},i}^{(d)}(k-1;k,l)\cdot\begin{cases} w_{\mathrm{DIR}}(L+l) & \text{if } d\in\mathcal{J}_{\mathrm{DIR,NZ}}(k) \\ w_{\mathrm{VEC}}(L+l) & \text{if } d\in\mathcal{J}_{\mathrm{VEC}}(k) \\ 1 & \text{else}\end{cases}$$

$$c_{\mathrm{DIR,IN},i}(k,l) = \sum_{d\in\mathcal{J}_{\mathrm{DIR,NZ}}(k)} c_{\mathrm{DIR,I},i}^{(d)}(k;k,l)\cdot\begin{cases} w_{\mathrm{DIR}}(l) & \text{if } d\in\mathcal{J}_{\mathrm{DIR}}(k-1)\cup\mathcal{J}_{\mathrm{VEC}}(k-1) \\ 1 & \text{else}\end{cases},$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2016 – All rights reserved

where $\mathcal{J}_{\mathrm{DIR,NZ}}(k)$ denotes the set of those first elements of $\mathcal{M}_{\mathrm{DIR}}(k)$ where the corresponding second element is non-zero.

In the case of multiple layered coding (indicated by SingleLayer==0 (see Table 120)), the sample values of the faded out and faded in directional HOA components are then determined by

$$c_{\mathrm{DIR,OUT},i}(k,l) = \sum_{d\in\mathcal{J}_{\mathrm{DIR,NZ}}(k-1)} c_{\mathrm{DIR,I},i}^{(d)}(k-1;k,l) \cdot \begin{cases} w_{\mathrm{DIR}}(L+l)\cdot 1 & \text{if } d\in\mathcal{J}_{\mathrm{DIR,NZ}}(k) \wedge (i\notin \mathcal{J}_{\mathrm{E}}(k)\cup\mathcal{J}_{\mathrm{D}}(k)) \\ w_{\mathrm{DIR}}(L+l)\cdot w_{\mathrm{DIR}}(L+l) & \text{if } d\in\mathcal{J}_{\mathrm{DIR,NZ}}(k)\wedge i\in \mathcal{J}_{\mathrm{E}}(k) \\ w_{\mathrm{DIR}}(L+l)\cdot w_{\mathrm{DIR}}(l) & \text{if } d\in\mathcal{J}_{\mathrm{DIR,NZ}}(k)\wedge i\in \mathcal{J}_{\mathrm{D}}(k) \\ w_{\mathrm{VEC}}(L+l) & \text{if } d\in\mathcal{J}_{\mathrm{VEC}}(k) \\ 1\cdot w_{\mathrm{DIR}}(l) & \text{if } d\in\mathcal{J}_{\mathrm{DIR,Z}}(k)\wedge i\in \mathcal{J}_{\mathrm{D}}(k) \\ 1 & \text{else} \end{cases},$$

$$c_{\mathrm{DIR,IN},i}(k,l) = \sum_{d\in\mathcal{J}_{\mathrm{DIR,NZ}}(k)} c_{\mathrm{DIR,I},i}^{(d)}(k;k,l) \cdot \begin{cases} w_{\mathrm{DIR}}(l) & \text{if } \left(d\in\mathcal{J}_{\mathrm{DIR}}(k-1)\cup\mathcal{J}_{\mathrm{VEC}}(k-1)\right)\wedge\left(i\notin \mathcal{J}_{\mathrm{E}}(k)\cup\mathcal{J}_{\mathrm{D}}(k)\right) \\ w_{\mathrm{DIR}}(l)\cdot w_{\mathrm{DIR}}(l) & \text{if } \left(d\in\mathcal{J}_{\mathrm{DIR}}(k-1)\cup\mathcal{J}_{\mathrm{VEC}}(k-1)\right)\wedge i\in \mathcal{J}_{\mathrm{D}}(k) \\ w_{\mathrm{DIR}}(l)\cdot w_{\mathrm{DIR}}(L+l) & \text{if } \left(d\in\mathcal{J}_{\mathrm{DIR}}(k-1)\cup\mathcal{J}_{\mathrm{VEC}}(k-1)\right)\wedge i\in \mathcal{J}_{\mathrm{E}}(k) \\ 1\cdot w_{\mathrm{DIR}}(L+l) & \text{if } \left(d\notin\mathcal{J}_{\mathrm{DIR}}(k-1)\cup\mathcal{J}_{\mathrm{VEC}}(k-1)\right)\wedge i\in \mathcal{J}_{\mathrm{E}}(k) \\ 1 & \text{else} \end{cases},$$

where $\mathcal{J}_{\mathrm{DIR,Z}}(k)$ denotes the set of those first elements of $\mathcal{M}_{\mathrm{DIR}}(k)$ where the corresponding second element is zero.

In subclause 12.4.2.4.4.2 Spatio-temporal interpolation of V-vectors replace:

— If there are coefficient sequences of the ambient HOA component that are explicitly additionally transmitted and faded in during the k-th frame (of which the indices are contained in the set $\mathcal{J}_{\mathrm{E}}(k)$), the respective coefficient sequences of the HOA representation $\tilde{\boldsymbol{C}}_{\mathrm{VEC}}(k)$ created from transport channel [i] have to be faded out using the fade-out part of the window w_{DIR}.

With:

— If there are coefficient sequences of the ambient HOA component that are explicitly additionally transmitted and faded in during the k-th frame (of which the indices are contained in the set $\mathcal{J}_{\mathrm{E}}(k)$), and the parameter NewChannelTypeOne[i] equals zero, the respective coefficient sequences of the HOA representation $\tilde{\boldsymbol{C}}_{\mathrm{VEC}}(k)$ have to be faded out using the fade-out part of the window w_{DIR}. The respective V-vector elements in $\boldsymbol{v}_{\mathrm{I}}^{(i)}(k)$ are discarded from the spatio-temporal interpolation in the following frame $k+1$ by setting them to zero.

Add at the end of subclause 12.4.2.4.5 Compose complete predominant sound component :

The modified predominant sound HOA representation $\boldsymbol{C}_{\mathrm{PS,F}}(k)$ is computed from $\boldsymbol{C}_{\mathrm{PS}}(k)$ by fading in coefficient sequences with indices contained in the index set $\mathcal{J}_{\mathrm{E}}(k)$ and fading out coefficient sequences with indices contained in the index set $\mathcal{J}_{\mathrm{D}}(k)$. In particular, the individual samples $c_{\mathrm{PS,F},n}(k,l)$ of the modified predominant sound HOA representation $\boldsymbol{C}_{\mathrm{PS,F}}(k)$ are computed according to

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

$$c_{\mathrm{PS,F},n}(k,l) = \begin{cases} c_{\mathrm{PS},n}(k,l)\cdot w_{\mathrm{DIR}}(l) & \text{if } n \in \mathcal{J}_{\mathrm{E}}(k) \\ c_{\mathrm{PS},n}(k,l)\cdot w_{\mathrm{DIR}}(L+l) & \text{if } n \in \mathcal{J}_{\mathrm{D}}(k) \\ c_{\mathrm{PS},n}(k,l) & \text{else} \end{cases} \quad \text{for } n = 1,\dots,O, l = 1,\dots,L.$$

In subclause 12.4.2.5 Ambience Synthesis replace:

The first O_{MIN} coefficients of the ambient HOA component are obtained by

$$\begin{bmatrix} c_{\mathrm{AMB},1}(k) \\ c_{\mathrm{AMB},2}(k) \\ \vdots \\ c_{\mathrm{AMB},O_{\mathrm{MIN}}}(k) \end{bmatrix} = \boldsymbol{\Psi}^{(N_{\mathrm{MIN}},N_{\mathrm{MIN}})} \cdot \begin{bmatrix} c_{\mathrm{I,AMB},1}(k) \\ c_{\mathrm{I,AMB},2}(k) \\ \vdots \\ c_{\mathrm{I,AMB},O_{\mathrm{MIN}}}(k) \end{bmatrix},$$

where $\boldsymbol{\Psi}^{(N_{\mathrm{MIN}},N_{\mathrm{MIN}})}$ denotes the mode matrix of order N_{MIN} with respect to the predefined directions $\boldsymbol{\Omega}_n^{(N_{\mathrm{MIN}})}, n = 1,\dots,O_{\mathrm{MIN}}$, defined in Annex F.1.5. Note that the multiplication by the mode matrix $\boldsymbol{\Psi}^{(N_{\mathrm{MIN}},N_{\mathrm{MIN}})}$ represents the inverse spatial transform intended to invert the spatial transform applied in the encoder (see Annex C.5.3.3.2) for decorrelating the first O_{MIN} coefficients of the ambient HOA component. The sample values of the remaining coefficients of the ambient HOA component are set according to

$$c_{\mathrm{AMB},n}(k,l) = c_{\mathrm{I,AMB},n}(k,l) \ \text{ for } \ O_{\mathrm{MIN}} < n \le O \ .$$

With:

The first O_{MIN} coefficient sequences of the ambient HOA component are computed as outlined in the following two subclauses. The sample values of the remaining higher-order coefficient sequences of the ambient HOA component are set according to

$$c_{\mathrm{AMB},n}(k,l) = c_{\mathrm{I,AMB},n}(k,l) \quad \text{for } \ O_{\mathrm{MIN}} < n \le O.$$

By default the first O_{MIN} HOA coefficient sequences are reconstructed with the method outlined in subclause 12.4.2.5.1. If N_{MIN} is of value 1, an alternative synthesis method described in subclause 12.4.2.5.2 can be used. In this case the flag UsePhaseShiftDecorr signals which of the two processing methods shall be applied.

Add the following two subclauses at the end of subclause 12.4.2.5 Ambience Synthesis:

12.4.2.5.1 Spatial transform

The first O_{MIN} coefficient sequences of the ambient HOA component are obtained by

$$\begin{bmatrix} \mathbf{c}_{\mathrm{AMB},1}(k) \\ \mathbf{c}_{\mathrm{AMB},2}(k) \\ \vdots \\ \mathbf{c}_{\mathrm{AMB},O_{\mathrm{MIN}}}(k) \end{bmatrix} = \boldsymbol{\Psi}_{\mathrm{MIN}} \cdot \begin{bmatrix} \mathbf{c}_{\mathrm{I,AMB},1}(k) \\ \mathbf{c}_{\mathrm{I,AMB},2}(k) \\ \vdots \\ \mathbf{c}_{\mathrm{I,AMB},O_{\mathrm{MIN}}}(k) \end{bmatrix},$$

where $\boldsymbol{\Psi}^{(N_{\mathrm{MIN}},N_{\mathrm{MIN}})}$ denotes the mode matrix of order N_{MIN} defined in Annex F.1.5 with respect to the predefined directions $\boldsymbol{\Omega}_n^{(N_{\mathrm{MIN}})}$, $n = 1,\dots,O_{\mathrm{MIN}}$ defined in Annex F.2-F.11. Note that the multiplication by the mode matrix represents the inverse spatial transform intended to invert the spatial transform applied in the encoder (see subclause C.5.3.3.2) for de-correlating the first O_{MIN} coefficient sequences of the ambient HOA component.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

12.4.2.5.2 Phase based transform

If the flag UsePhaseShiftDecorr is == 1, the following processing is applied to reconstruct the first four coefficient sequences of the ambient HOA component by

$$\begin{bmatrix} \mathbf{c}_{\mathrm{AMB},1}(k) \\ \mathbf{c}_{\mathrm{AMB},2}(k) \\ \mathbf{c}_{\mathrm{AMB},3}(k) \\ \mathbf{c}_{\mathrm{AMB},4}(k) \end{bmatrix} = \begin{bmatrix} c(3) \cdot A_{+90}(k) + c(2) \cdot \left[\mathbf{c}_{\mathrm{I,AMB},1}(k) + \mathbf{c}_{\mathrm{I,AMB},2}(k)\right] \\ B_{+90}(k) + c(5) \cdot \left[\mathbf{c}_{\mathrm{I,AMB},1}(k) - \mathbf{c}_{\mathrm{I,AMB},2}(k)\right] + c(6) \cdot \mathbf{c}_{\mathrm{I,AMB},3}(k) \\ \mathbf{c}_{\mathrm{I,AMB},4}(k) \\ c(4) \cdot \left[\mathbf{c}_{\mathrm{I,AMB},1}(k) + \mathbf{c}_{\mathrm{I,AMB},2}(k)\right] - A_{+90}(k) \end{bmatrix}$$

with the coefficients c as defined in Table AMD1.9 and $A_{+90}(k)$ and $B_{+90}(k)$ are the frames of +90 degree phase shifted signals A and B defined by

$$A(k) = c(0) \cdot \left[\mathbf{c}_{\mathrm{I,AMB},1}(k) - \mathbf{c}_{\mathrm{I,AMB},2}(k)\right],$$

$$B(k) = c(1) \cdot \left[\mathbf{c}_{\mathrm{I,AMB},1}(k) + \mathbf{c}_{\mathrm{I,AMB},2}(k)\right].$$

Note that the phase shift operation introduces a delay of one frame. In order to avoid this delay in the decoder for the case that the flag UsePhaseShiftDecorr has a value of 1, it is assumed that the Channel Reassignment processing block (see subclause 12.4.2.3) provides the frame $\boldsymbol{C}_{\mathrm{I,AMB}}(k+1)$ instead of $\boldsymbol{C}_{\mathrm{I,AMB}}(k)$, which can be maintained by a respective modification at the spatial HOA encoding stage.

Table AMD1.9 — Coefficients for phase-based transform

n	c(n)
0	1.0140887535122356
1	0.22902729095022714
2	0.98199999999999998
3	0.16084982644276205
4	0.51316810111307576
5	0.97489691762770481
6	-0.88020833333333337

Add the following subclauses after 12.4.2.5 Ambience Synthesis:

12.4.2.x Preliminary HOA composition

In case of single layered coding:

In the case of single-layered coding (indicated by SingleLayer==1 (see Table 120)), the frame $\boldsymbol{C}_{\mathrm{PRE}}(k)$ of the preliminary decoded HOA representation is computed by

$$\boldsymbol{C}_{\mathrm{PRE}}(k) = \boldsymbol{C}_{\mathrm{AMB}}(k) + \boldsymbol{C}_{\mathrm{PS}}(k) = \boldsymbol{C}_{\mathrm{AMB}}(k) + \boldsymbol{C}_{\mathrm{DIR}}(k) + \boldsymbol{C}_{\mathrm{PD}}(k) + \boldsymbol{C}_{\mathrm{VEC}}(k).$$

Additionally, the frame $\boldsymbol{C}_{\mathrm{PRE,F}}(k)$ of a modified version of the preliminary decoded HOA representation is computed by

$$\boldsymbol{C}_{\mathrm{PRE,F}}(k) = \boldsymbol{C}_{\mathrm{AMB}}(k) + \boldsymbol{C}_{\mathrm{PS,F}}(k).$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

This modified HOA representation is assumed to be successively input to the PAR decoder instead of the original version $\boldsymbol{C}_{\mathrm{PRE}}(k)$ to av oid si gnal di scontinuities af ter pe rforming on $\boldsymbol{C}_{\mathrm{PRE,F}}(k)$ the trun cation and coefficient selection (see subclause **12.4.2.x.2 Truncation and Coefficient Selection**).

In case of multiple layered coding:

In the case of multiple layered coding (indicated by SingleLayer==0 (see Table 120)), the coefficient sequences $\boldsymbol{c}_{\mathrm{PRE},i}(k)$, $i = 1, \dots, O$, of the preliminary decoded HOA representation $\boldsymbol{C}_{\mathrm{PRE}}(k)$ are computed by

$$\boldsymbol{c}_{\mathrm{PRE},i}(k) = \begin{cases} \boldsymbol{c}_{\mathrm{AMB},i}(k) & \text{if } i \in \mathcal{J}_{\mathrm{U}}(k) \\ \boldsymbol{c}_{\mathrm{AMB},i}(k) + \boldsymbol{c}_{\mathrm{PS},i}(k) & \text{if } i \in \mathcal{J}_{\mathrm{E}}(k) \cup \mathcal{J}_{\mathrm{D}}(k). \\ \boldsymbol{c}_{\mathrm{PS},i}(k) & \text{else} \end{cases}$$

This means that the transmitted coefficient sequences with indices $i \in \mathcal{J}_{\mathrm{E}}(k) \cup \mathcal{J}_{\mathrm{D}}(k) \cup \mathcal{J}_{\mathrm{U}}(k)$ actually represent the original HOA representation instead of its ambient component. Hence, for the transmitted coefficient sequences which are neither faded in nor faded out within the current frame, nothing has to be added to them. For the transmitted coefficient sequences that are faded in (or faded out) within the current frame, i.e. those with indices $i \in \mathcal{J}_{\mathrm{E}}(k) \cup \mathcal{J}_{\mathrm{D}}(k)$, the corresponding coefficient sequences of the predominant sound HOA representation $\boldsymbol{C}_{\mathrm{PS}}(k)$ are added, which are supposed to have been appropriately faded out (or faded in) at the Predominant Sound Synthesis.

Similarly, the coefficient sequences $\boldsymbol{c}_{\mathrm{PRE,F},i}(k)$ of the modified version of the preliminary decoded HOA frame $\boldsymbol{C}_{\mathrm{PRE,F}}(k)$ are computed by

$$\boldsymbol{c}_{\mathrm{PRE,F},i}(k) = \begin{cases} \boldsymbol{c}_{\mathrm{AMB},i}(k) & \text{if } i \in \mathcal{J}_{\mathrm{U}}(k) \\ \boldsymbol{c}_{\mathrm{AMB},i}(k) + \boldsymbol{c}_{\mathrm{PS,F},i}(k) & \text{if } i \in \mathcal{J}_{\mathrm{E}}(k) \cup \mathcal{J}_{\mathrm{D}}(k). \\ \boldsymbol{c}_{\mathrm{PS,F},i}(k) & \text{else} \end{cases}$$

Note that if the number of actually used layers changes between two successive frames (i.e. if $M_{\mathrm{LAY}}(k) \neq M_{\mathrm{LAY}}(k-1)$), with the above computation there occurs in general a discontinuity in all coefficient sequences of the preliminary decoded HOA representation and its modified version between the $(k-1)$-th and k-th frame. One possible solution for this problem is to introduce an additional delay of one frame within the preliminary HOA composition and to fade out and fade in the coefficient sequences at the discontinuity.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

12.4.2.x Sub-band Directional Signals Synthesis

The purpose of the Sub-band Directional Signals Synthesis is to approximate the non-transmitted coefficient sequences of the residual (i.e. ambient) HOA component by a composition of directional sub-band signals, which are predicted by a weighted/scaled sum of the transmitted coefficient sequences of the residual (i.e. ambient) HOA component, where the scaling is complex valued in general. In particular, each directional sub-band signal related to the j-th sub-band, $j \in \{1, \dots, F\}$, is represented parametrically by complex valued prediction scaling factors matrices $\mathbf{A}(k, b)$ and tuple sets $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k, b)$ related to the b-th sub-band group $(b \in 1, \dots, B)$ which includes the j-th sub-band. Per sub-band group there are at most D_{SB} potential active direction trajectories, where the indices identifying the active direction trajectories for the b-th sub-band group are assumed to be contained in the set $\tilde{\mathcal{J}}_{\mathrm{DIR}}(k, b) \subseteq \{1, \dots, D_{\mathrm{SB}}\}$. For each index $d \in \tilde{\mathcal{J}}_{\mathrm{DIR}}(k, b)$ of an active direction trajectory the respective direction is denoted by $\boldsymbol{\Omega}_{\mathrm{SB},d}(k, b)$, both of which are assumed to be contained as tuples in the set $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k, b)$, i.e.

$$\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k, b) = \left\{ \left(d, \boldsymbol{\Omega}_{\mathrm{SB},d}(k, b) \right) \middle| \, d \in \tilde{\mathcal{J}}_{\mathrm{DIR}}(k, b) \right\}.$$

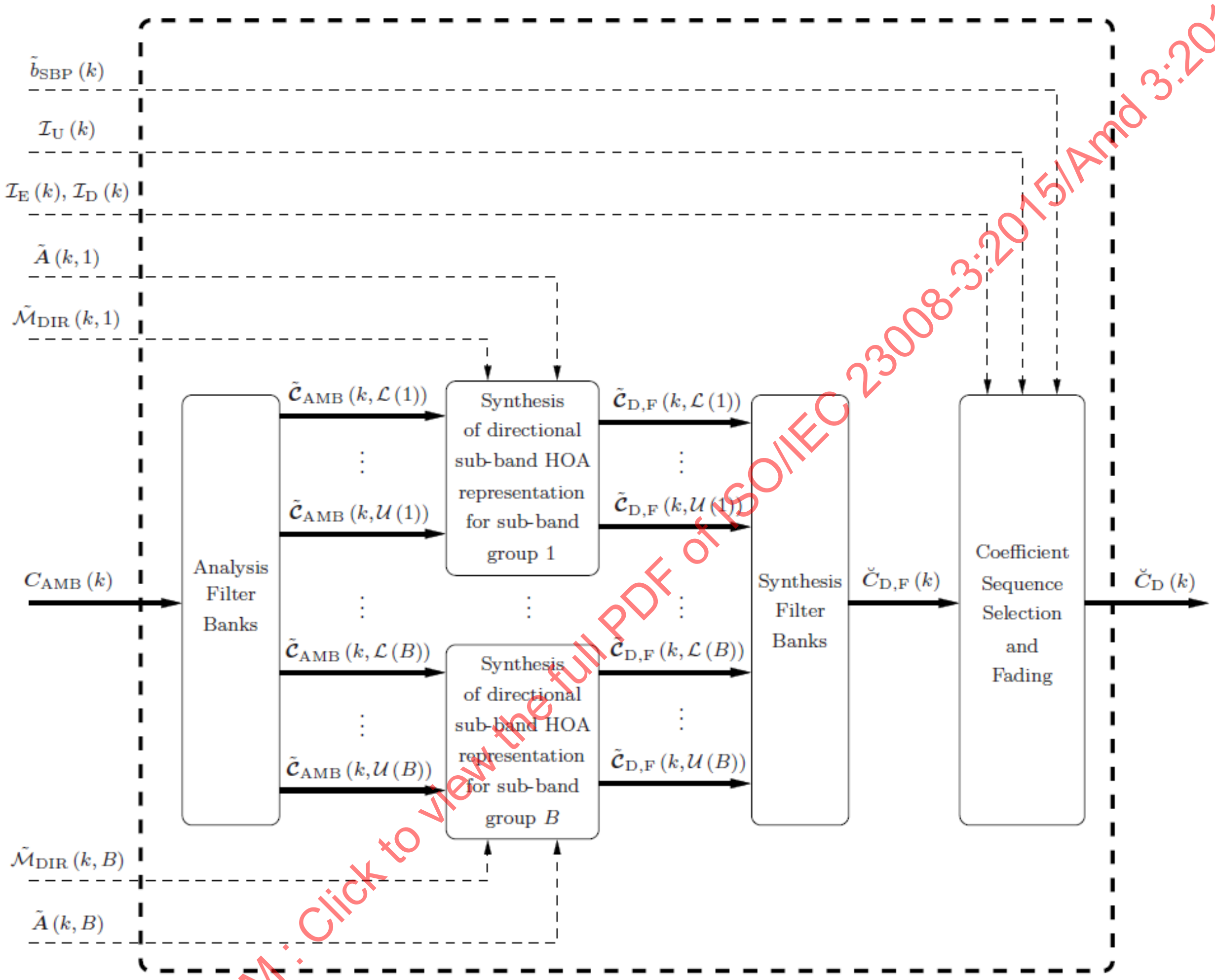
Note that the set $\tilde{\mathcal{J}}_{\mathrm{DIR}}(k, b)$ is assumed to consist of the first elements of the tuples of $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k, b)$, and can hence be computed from $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k, b)$.

Further note, that in the absence of predominant sound signals the ambient component corresponds to a "truncated" version of the original HOA representation. Truncation in this context means that the original HOA

© ISO/IEC 2017 – All rights reserved

representation is approximated by only I of its total O coefficient sequences, i.e. by those that were transmitted within the I transport channels.

The detailed architecture of the Sub-band Directional Signals Synthesis is illustrated in Figure **AMD1.1**. The individual processing units to compute the frame $\breve{\boldsymbol{C}}_{\mathrm{D}}(k)$ of the HOA representation of the composition of all predicted sub-band directional signals will be described in the following.



$\tilde{b}_{\mathrm{SBP}}(k)$
$\mathcal{I}_{\mathrm{U}}(k)$
$\mathcal{I}_{\mathrm{E}}(k), \mathcal{I}_{\mathrm{D}}(k)$
$\tilde{A}(k,1)$
$\tilde{\mathcal{M}}_{\mathrm{DIR}}(k,1)$
$C_{\mathrm{AMB}}(k)$
Analysis Filter Banks
$\tilde{\boldsymbol{c}}_{\mathrm{AMB}}(k,\mathcal{L}(1))$
$\tilde{\boldsymbol{c}}_{\mathrm{AMB}}(k,\mathcal{U}(1))$
$\tilde{\boldsymbol{c}}_{\mathrm{AMB}}(k,\mathcal{L}(B))$
$\tilde{\boldsymbol{c}}_{\mathrm{AMB}}(k,\mathcal{U}(B))$
Synthesis of directional sub-band HOA representation for sub-band group 1
Synthesis of directional sub-band HOA representation for sub-band group B
$\tilde{\boldsymbol{c}}_{\mathrm{D,F}}(k,\mathcal{L}(1))$
$\tilde{\boldsymbol{c}}_{\mathrm{D,F}}(k,\mathcal{U}(1))$
$\tilde{\boldsymbol{c}}_{\mathrm{D,F}}(k,\mathcal{L}(B))$
$\tilde{\boldsymbol{c}}_{\mathrm{D,F}}(k,\mathcal{U}(B))$
Synthesis Filter Banks
$\breve{C}_{\mathrm{D,F}}(k)$
Coefficient Sequence Selection and Fading
$\breve{C}_{\mathrm{D}}(k)$
$\tilde{\mathcal{M}}_{\mathrm{DIR}}(k,B)$
$\tilde{A}(k,B)$

Figure AMD1.1 — Sub-band directional signals synthesis.

12.4.2.x.1 Analysis Filter Banks

Each frame $\boldsymbol{c}_{\mathrm{AMB},n}(k)$, $n = 1, \dots, O$, of an individual coefficient sequence of the ambient HOA representation $\boldsymbol{C}_{\mathrm{AMB}}(k)$ is first decomposed into frames of individual sub-band signals $\tilde{c}_{\mathrm{AMB},n}(k,j)$, $j = 1, \dots, F$. For each sub-band j, $j = 1, \dots, F$, the frames of the sub-band signals of the individual HOA coefficient sequences are collected into the sub-band HOA representation $\tilde{\boldsymbol{C}}_{\mathrm{AMB}}(k,j)$ as

$$\tilde{\boldsymbol{C}}_{\mathrm{AMB}}(k,j) = \begin{bmatrix} \tilde{c}_{\mathrm{AMB},1}(k,j) \\ \tilde{c}_{\mathrm{AMB},2}(k,j) \\ \vdots \\ \tilde{c}_{\mathrm{AMB},O}(k,j) \end{bmatrix} \quad \text{for} \quad j = 1, \dots, F.$$

The filter bank is assumed to be based on Quadrature Mirror Filters (QMF) with a total of $F = 64$ sub-bands (ISO/IEC 23003-1:2007, *Information technology — MPEG audio technologies — Part 1: MPEG Surround*).

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Note that, in contrast to the HOA coefficient sequences $\boldsymbol{c}_{\mathrm{AMB},n}(k)$ their sub-band representations $\tilde{\boldsymbol{c}}_{\mathrm{AMB},n}(k,j)$ are complex valued in general. Further, the sub-band signals are decimated in time compared to the original time-domain signals by a factor of F. As a consequence, the number of samples in the frames $\tilde{\boldsymbol{c}}_{\mathrm{AMB},n}(k,j)$ is $L_{\mathrm{SB}} := L/F$. It is assumed that L is an integral multiple of F to assure that L_{SB} has a positive integer value.

A further important implementation issue is that the coefficient sequences of the ambient HOA representation with indices greater than O_{MAX} are assumed to be zero. Hence, the application of the analysis filters can be restricted to the HOA coefficient sequences $\boldsymbol{c}_{\mathrm{AMB},n}(k)$ with indices $n = 1, \dots, O_{\mathrm{MAX}}$ only. The sub-band signal frames $\tilde{\boldsymbol{c}}_{\mathrm{AMB},n}(k,j)$ with indices $n = O_{\mathrm{MAX}} + 1, \dots, O$ can be set to zero.

12.4.2.x.2 Synthesis of directional sub-band HOA representation for individual sub-band groups

In order to avoid artifacts due to changes of the directions and prediction coefficients between successive frames, the computation of the directional sub-band HOA representation is based on the concept of overlap add in the sub-band domain. Hence, the HOA representation $\tilde{\boldsymbol{C}}_{\mathrm{D,F}}(k,j)$ of active directional sub-band signals related to the j-th sub-band, $j = 1, \dots, F$, is computed as the sum of a faded out component and a faded in component:

$$\tilde{\boldsymbol{C}}_{\mathrm{D,F}}(k,j) = \tilde{\boldsymbol{C}}_{\mathrm{D,F,OUT}}(k,j) + \tilde{\boldsymbol{C}}_{\mathrm{D,F,IN}}(k,j).$$

To compute the two individual components, in a first step the instantaneous frame of all directional sub-band signals $\tilde{\boldsymbol{X}}_{\mathrm{I}}(k_1;k;j)$ for the j-th sub-band is computed by

$$\tilde{\boldsymbol{X}}_{\mathrm{I}}(k_1;k;j) = \mathbf{A}(k_1,b)\tilde{\boldsymbol{C}}_{\mathrm{AMB}}(k,j) \quad \text{for} \quad k_1 \in \{k, k+1\} \text{ and } \mathcal{L}(b) \le j \le \mathcal{U}(b).$$

using the ambient sub-band HOA representation $\tilde{\boldsymbol{C}}_{\mathrm{AMB}}(k,j)$ for the k-th frame and the prediction coefficients matrix $\mathbf{A}(k_1,b)$ for the b-th sub-band group including the j-th sub-band and for the (k_1)-th frame, where $k_1 \in \{k, k+1\}$. The (matrix) frame $\tilde{\boldsymbol{X}}_{\mathrm{I}}(k_1;k;j)$ is assumed to be composed of the (row) frames of the individual directional sub-band signals $\tilde{\boldsymbol{x}}_{\mathrm{I},d}(k_1;k;j)$, $d = 1, \dots, D_{\mathrm{SB}}$, according to

$$\tilde{\boldsymbol{X}}_{\mathrm{I}}(k_1;k;j) = \begin{bmatrix} \tilde{\boldsymbol{x}}_{\mathrm{I},1}(k_1;k;j) \\ \vdots \\ \tilde{\boldsymbol{x}}_{\mathrm{I},D_{\mathrm{SB}}}(k_1;k;j) \end{bmatrix}.$$

Note that all elements of the frame $\boldsymbol{x}_{\mathrm{I},d}(k_1;k;j)$ of a directional signal are zero if the corresponding direction is not active, i.e., if the corresponding directional trajectory index d is not contained in the set $\mathcal{J}_{\mathrm{DIR}}(k,b)$.

In a second step, the instantaneous sub-band HOA representation $\tilde{\boldsymbol{C}}^{(d)}_{\mathrm{D,F,I}}(k_1;k;j)$ of the active directional sub-band signal $\tilde{\boldsymbol{x}}_{\mathrm{I},d}(k_1;k;j)$ with direction index $d \in \mathcal{J}_{\mathrm{DIR}}(k,b)$ with respect to the direction $\boldsymbol{\Omega}_{\mathrm{SB},d}(k,b)$ is obtained as

$$\tilde{\boldsymbol{C}}^{(d)}_{\mathrm{D,F,I}}(k_1;k;j) = \boldsymbol{\psi}\left(\boldsymbol{\Omega}_{\mathrm{SB},d}(k,b)\right)\tilde{\boldsymbol{x}}_{\mathrm{I},d}(k_1;k;j),$$

where $\boldsymbol{\psi}\left(\boldsymbol{\Omega}_{\mathrm{SB},d}(k,b)\right) \in \mathbb{R}^{O}$ denotes the mode vector with respect to the direction $\boldsymbol{\Omega}_{\mathrm{SB},d}(k,b)$ which is computed as described in Annex F.1.5.

Assuming the matrices $\tilde{\boldsymbol{C}}_{\mathrm{D,F,OUT}}(k,j)$, $\tilde{\boldsymbol{C}}_{\mathrm{D,F,IN}}(k,j)$, and $\tilde{\boldsymbol{C}}^{(d)}_{\mathrm{D,F,I}}(k_1;k;j)$ to be composed of their samples by

$$\tilde{\boldsymbol{C}}_{\mathrm{D,F,OUT}}(k,j) = \begin{bmatrix} \tilde{c}_{\mathrm{D,F,OUT},1}(k,j;1) & \dots & \tilde{c}_{\mathrm{D,F,OUT},1}(k,j;L_{\mathrm{SB}}) \\ \vdots & \ddots & \vdots \\ \tilde{c}_{\mathrm{D,F,OUT},O}(k,j;1) & \dots & \tilde{c}_{\mathrm{D,F,OUT},O}(k,j;L_{\mathrm{SB}}) \end{bmatrix} \in \mathbb{R}^{O \times L_{\mathrm{SB}}},$$

$$\tilde{\boldsymbol{C}}_{\mathrm{D,F,IN}}(k,j) = \begin{bmatrix} \tilde{c}_{\mathrm{D,F,IN},1}(k,j;1) & \dots & \tilde{c}_{\mathrm{D,F,IN},1}(k,j;L_{\mathrm{SB}}) \\ \vdots & \ddots & \vdots \\ \tilde{c}_{\mathrm{D,F,IN},O}(k,j;1) & \dots & \tilde{c}_{\mathrm{D,F,IN},O}(k,j;L_{\mathrm{SB}}) \end{bmatrix} \in \mathbb{R}^{O \times L_{\mathrm{SB}}},$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

$$\tilde{\boldsymbol{C}}_{\mathrm{D,F,I}}^{(d)}(k_1;k;j) = \begin{bmatrix} \tilde{c}_{\mathrm{D,F,I},1}^{(d)}(k_1;k;j;1) & \dots & \tilde{c}_{\mathrm{D,F,I},1}^{(d)}(k_1;k;j;L_{\mathrm{SB}}) \\ \vdots & \ddots & \vdots \\ \tilde{c}_{\mathrm{D,F,I},O}^{(d)}(k_1;k;j;1) & \dots & \tilde{c}_{\mathrm{D,F,I},O}^{(d)}(k_1;k;j;L_{\mathrm{SB}}) \end{bmatrix} \in \mathbb{R}^{O\times L_{\mathrm{SB}}},$$

the sample values of the faded out and faded in components of the HOA representation of active directional sub-band signals are finally determined for $1 \le l \le L_{\mathrm{SB}}$, $1 \le j \le F$ and $\mathcal{L}(b) \le j \le \mathcal{U}(b)$ by

$$\tilde{c}_{\mathrm{D,F,OUT},n}(k,j;l) = \sum_{d\in\tilde{\mathfrak{I}}_{\mathrm{DIR}}(k,b)} \tilde{c}_{\mathrm{D,F,I},n}^{(d)}(k;k;j;l)\cdot w_{\mathrm{SB}}(L_{\mathrm{SB}}+l)$$

$$\tilde{c}_{\mathrm{D,F,IN},n}(k,j;l) = \sum_{d\in\tilde{\mathfrak{I}}_{\mathrm{DIR}}(k+1,b)} \tilde{c}_{\mathrm{D,F,I},n}^{(d)}(k+1;k;j;l)\cdot w_{\mathrm{SB}}(l)$$

where the vector

$$\mathbf{w}_{\mathrm{SB}} = [w_{\mathrm{SB}}(1) \quad w_{\mathrm{SB}}(2) \quad \dots \quad w_{\mathrm{SB}}(2L_{\mathrm{SB}})]^T \in \mathbb{R}^{2L_{\mathrm{SB}}}$$

represents an overlap add (periodic Hann) window function to be applied on the sub-band signals, of which the elements are defined by

$$w_{\mathrm{SB}}(l) = \frac{1}{2}\left[1-\cos\left(2\pi\frac{l-1}{2L_{\mathrm{SB}}}\right)\right].$$

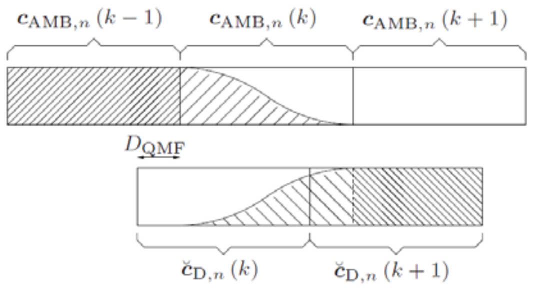
12.4.2.x.3 Synthesis Filter Banks

The individual time domain coefficient sequences $\breve{c}_{\mathrm{D,F},n}(k)$, $n = 1,\dots,O$, of the HOA representation $\breve{\boldsymbol{C}}_{\mathrm{D,F}}(k)$ of the composition of all predicted sub-band directional signals are synthesized from the corresponding sub-band coefficient sequences $\tilde{c}_{\mathrm{D,F},n}(k,j)$, $j = 1,\dots,F$ by the Synthesis Filter Banks. Note that the synthesized time domain coefficient sequences have a delay of $D_{\mathrm{QMF}} = 577$ samples due to the successive application of the QMF based analysis and synthesis filter banks, which is expressed by the breve symbol above the variables.

12.4.2.x.4 Coefficient Sequence Selection and Fading

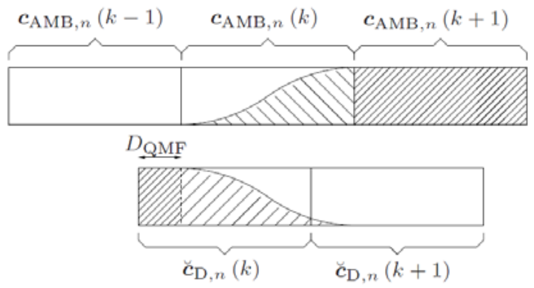
In a final step, the preliminary computed HOA representation $\breve{\boldsymbol{C}}_{\mathrm{D,F}}(\breve{k})$ of the composition of all predicted sub-band directional signals is modified to have only contributions for those coefficient sequences, which have not been explicitly transmitted for the ambient HOA component. Further, for those coefficient sequences of the ambient HOA component that are explicitly additionally transmitted and faded in (or faded out), the respective coefficient sequences of the preliminary HOA representation $\breve{\boldsymbol{C}}_{\mathrm{D,F}}(k)$ of the composition of all predicted sub-band directional signals have to be modified by fading them out (or fading them in), respectively. Due to the delay between $\boldsymbol{C}_{\mathrm{AMB}}(k)$ and $\breve{\boldsymbol{C}}_{\mathrm{D,F}}(k)$ of D_{QMF} samples the fading of the coefficient sequences of $\breve{\boldsymbol{C}}_{\mathrm{D,F}}(k)$ is performed across frame boundaries, as illustrated in Figure **AMD1.2**. The modified HOA representation of the composition of all predicted sub-band directional signals is denoted by $\breve{\boldsymbol{C}}_{\mathrm{D}}(k)$ with its coefficient sequences $\breve{c}_{\mathrm{D},n}(k)$, $n = 1,\dots,O$.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017



$c_{\mathrm{AMB},n}(k-1)$
$c_{\mathrm{AMB},n}(k)$
$c_{\mathrm{AMB},n}(k+1)$
D_{QMF}
$\breve{c}_{\mathrm{D},n}(k)$
$\breve{c}_{\mathrm{D},n}(k+1)$

(a) Example of a faded coefficient sequence $\breve{c}_{\mathrm{D},n}$ of the HOA representation of the composition of all predicted sub-band directional signals in the case the respective coefficient sequence $c_{\mathrm{AMB},n}$ of the ambient HOA component is faded out, i.e. if $n \in \mathcal{I}_{\mathrm{U}}(k-1) \wedge n \in \mathcal{I}_{\mathrm{D}}(k) \wedge n \notin (\mathcal{I}_{\mathrm{U}}(k+1) \cup \mathcal{I}_{\mathrm{E}}(k+1) \cup \mathcal{I}_{\mathrm{D}|}(k+1))$



$c_{\mathrm{AMB},n}(k-1)$
$c_{\mathrm{AMB},n}(k)$
$c_{\mathrm{AMB},n}(k+1)$
D_{QMF}
$\breve{c}_{\mathrm{D},n}(k)$
$\breve{c}_{\mathrm{D},n}(k+1)$

(b) Example of a faded coefficient sequence $\breve{c}_{\mathrm{D},n}$ of the HOA representation of the composition of all predicted sub-band directional signals in the case the respective coefficient sequence $c_{\mathrm{AMB},n}$ of the ambient HOA component is faded in, i.e. if $n \notin (\mathcal{I}_{\mathrm{U}}(k-1) \cup \mathcal{I}_{\mathrm{E}}(k-1) \cup \mathcal{I}_{\mathrm{D}}(k-1)) \wedge n \in \mathcal{I}_{\mathrm{E}}(k) \wedge n \in \mathcal{I}_{\mathrm{U}}(k+1)$

Figure AMD1.2 — Illustration of faded coefficient sequences of $\breve{C}_{\mathrm{D}}(k)$.

For the case that a coefficient sequence $\boldsymbol{c}_{\mathrm{AMB},n}$ of the ambient HOA component is faded out in the k-th frame (i.e. $n \in \mathcal{J}_{\mathrm{D}}(k)$) as illustrated in Figure AMD1.2a, the fade in of the coefficient sequence $\breve{\boldsymbol{c}}_{\mathrm{D},n}$ in the k-th frame begins D_{QMF} samples later, where in particular the fading in is finished only at the D_{QMF}-th sample of the $(k+1)$-th frame.

Similarly, for the case that a coefficient sequence $\boldsymbol{c}_{\mathrm{AMB},n}$ of the ambient HOA component is faded in in the k-th frame (i.e. $n \in \mathcal{J}_{\mathrm{E}}(k)$) as illustrated in Figure AMD1.2b, the fade out of the coefficient sequence $\breve{\boldsymbol{c}}_{\mathrm{D},n}$ in the k-th frame begins D_{QMF} samples later, where in particular the fading out is finished only at the D_{QMF}-th sample of the $(k+1)$-th frame.

Finally, it has to be considered that a fade in or fade out of the coefficient sequences $\breve{\boldsymbol{c}}_{\mathrm{D,F},n}(k)$ of the HOA representation of the composition of all predicted sub-band directional signals is only required if it is not already present, resulting from overlap-add processing.

In the case that $\tilde{b}_{\mathrm{SBP}}(k-1) = 0$ and $\tilde{b}_{\mathrm{SBP}}(k) = 1$, there is already a fade in within each of the k-th frames $\breve{\boldsymbol{c}}_{\mathrm{D,F},n}(k)$, $n = 1, \dots, O$, such that is not necessary to apply an additional fade in. Similarly, in the case that $\tilde{b}_{\mathrm{SBP}}(k-1) = 1$ and $\tilde{b}_{\mathrm{SBP}}(k) = 0$, there is already a fade out within each of the k-th frames $\breve{\boldsymbol{c}}_{\mathrm{D,F},n}(k)$, $n = 1, \dots, O$, such that is not necessary to apply an additional fade out. Altogether, the computation of the sample values $\breve{c}_{\mathrm{D},n}(k,l)$, $n = 1, \dots, O$, $l = 1, \dots, L$, of the coefficient sequences of the HOA representation of the composition of all predicted sub-band directional signals is hence formally expressed by

$$\breve{c}_{\mathrm{D},n}(k,l) = \breve{c}_{\mathrm{D,F},n}(k,l) \cdot \begin{cases} w_{\mathrm{DIR}}(l + L - D_{\mathrm{QMF}}) & \text{if } 1 \le l \le D_{\mathrm{QMF}} \wedge n \in \mathcal{J}_{\mathrm{D}}(k-1) \wedge \tilde{b}_{\mathrm{SBP}}(k-2) = 1 \\ w_{\mathrm{DIR}}(l + 2L - D_{\mathrm{QMF}}) & \text{if } 1 \le l \le D_{\mathrm{QMF}} \wedge n \in \mathcal{J}_{\mathrm{E}}(k-1) \wedge \tilde{b}_{\mathrm{SBP}}(k-1) = 1 \\ 1 & \text{if } 1 \le l \le D_{\mathrm{QMF}} \wedge \\ & \{n \notin \mathcal{J}_{\mathrm{U}}(k-1) \cup \mathcal{J}_{\mathrm{E}}(k-1) \cup \mathcal{J}_{\mathrm{D}}(k-1) \\ & \quad \vee \left(n \in \mathcal{J}_{\mathrm{D}}(k-1) \wedge \tilde{b}_{\mathrm{SBP}}(k-2) = 0\right) \\ & \quad \vee \left(n \in \mathcal{J}_{\mathrm{E}}(k-1) \wedge \tilde{b}_{\mathrm{SBP}}(k-1) = 0\right)\} \\ w_{\mathrm{DIR}}(l - D_{\mathrm{QMF}}) & \text{if } D_{\mathrm{QMF}} < l \le L \wedge n \in \mathcal{J}_{\mathrm{D}}(k) \wedge \tilde{b}_{\mathrm{SBP}}(k-1) = 1 \\ w_{\mathrm{DIR}}(l + L - D_{\mathrm{QMF}}) & \text{if } D_{\mathrm{QMF}} < l \le L \wedge n \in \mathcal{J}_{\mathrm{E}}(k) \wedge \tilde{b}_{\mathrm{SBP}}(k) = 1 \\ 1 & \text{if } D_{\mathrm{QMF}} < l \le L \wedge \\ & \{n \notin \mathcal{J}_{\mathrm{U}}(k) \cup \mathcal{J}_{\mathrm{E}}(k) \cup \mathcal{J}_{\mathrm{D}}(k) \\ & \quad \vee \left(n \in \mathcal{J}_{\mathrm{D}}(k) \wedge \tilde{b}_{\mathrm{SBP}}(k-1) = 0\right) \\ & \quad \vee \left(n \in \mathcal{J}_{\mathrm{E}}(k) \wedge \tilde{b}_{\mathrm{SBP}}(k) = 0\right)\} \\ 0 & \text{else} \end{cases} .$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

12.4.2.x Parametric Ambience Replication (PAR) Decoder

The Parametric Ambience Replication (PAR) Decoder, as illustrated in Figure **AMD1.3**, replicates an ambient HOA component $\boldsymbol{C}_{\text{PA}}(k)$ to complement the missing ambience in the preliminary reconstructed HOA component $\boldsymbol{C}_{\text{PRE}}(k)$. The replicated ambient component is created in the sub-band domain, where its sub-band representation $\boldsymbol{\mathcal{C}}_{\text{PA}}(k,j)$ related to the j-th sub-band is assumed to be of order $N_{\text{PAR}}(g)$ depending on the corresponding g-th sub-band group $g = 1,\ldots,G$. The orders $N_{\text{PAR}}(g)$ for each sub-band group $g = 1,\ldots,G$ are specified in subclause **12.4.1.2.3 Orders $\mathbf{N_{PAR}(g)}$ for each sub-band group $\mathbf{g = 1,\ldots,G}$ for PAR**. The sub-band representation $\boldsymbol{\mathcal{C}}_{\text{PA}}(k,j)$ is represented and created by means of $O_{\text{PAR}}(g) = (N_{\text{PAR}}(g)+1)^2$ virtual loudspeaker sub-band signals $\boldsymbol{\mathcal{X}}_{\text{PA}}(k,j)$ at directions $\boldsymbol{\Omega}_d^{(N_{\text{PAR}}(g))}$, $d = 1,\ldots,O_{\text{PAR}}(g)$, defined in the tables in the Annexes F.2 – F.11. These up-mix sub-band signals are computed as a mixture of the sub-band signals $\boldsymbol{\mathcal{X}}_{\text{DEC}}(k,j)$, which are themselves created by de-correlation filters from the virtual loudspeaker sub-band signals representing a so-called truncated and sparse sub-band HOA representation $\boldsymbol{\mathcal{C}}_{\text{SP}}(k,j)$. The latter is obtained from the sub-band HOA representation $\boldsymbol{\mathcal{C}}_{\text{PRE}}(k,j)$ by reducing its order to $N_{\text{PAR}}(g)$ and setting all coefficient sequences to zero which are zero in the intermediate representation of the ambient HOA component $\text{C}_{\text{I,AMB}}(\text{k})$. The number of de-correlated sub-band signals to be mixed for the creation of each up-mix sub-band signal is allowed to vary over time according to the values of NumOfDecorrSigs PerParSubbandTable in order to adapt to the diffuseness of the ambient HOA component to be replicated. This number, denoted by $N_{\text{SIG}}(k,g)$ specified in subclause **12.4.1.x Permutation matrices $\boldsymbol{P}_{\mathbf{PAR}}(\boldsymbol{k},\boldsymbol{g})$ for PAR** , offers the possibility to control the amount of side information required to code the mixing matrices $\boldsymbol{M}_{\text{PAR}}(k,g)$ for the individual sub-band groups $g = 1,\ldots,G$. Further, for $N_{\text{SIG}}(k,g) < O_{\text{PAR}}(g)$ the mixing uses de-correlated sub-band signals obtained from virtual loudspeaker signals $\boldsymbol{\mathcal{X}}_{\text{SP}}(k,j)$ at directions in the neighborhood of the direction of the up-mix signal. This operation prevents that directional components of the truncated and sparse sub-band HOA representation $\boldsymbol{\mathcal{C}}_{\text{SP}}(k,j)$ are undesirably spatially distributed over all directions for the replication of the ambient HOA component. An additional aspect is that for each number $N_{\text{SIG}}(k,g)$ and each individual up-mix sub-band signal it is specified in Table **F.x Table for ParDecorrSigsSelectionTableIdx referring to NumOfDecorrSigsPerParSubbandTable and ParSelectedDecorrSigsIdxMatrixTable**, which de-correlated sub-band signals have to be mixed. In order to decrease the mutual correlation between each group of de-correlated sub-band signals to be mixed, the assignment of the virtual loudspeaker signals to the de-correlation filters is adapted to the choice of de-correlated sub-band signals. This assignment is expressed through the permutation matrices $\boldsymbol{P}_{\text{PAR}}(k,g)$ for the individual sub-band groups $g = 1,\ldots,G$. The individual processing units of the PAR Decoder to compute the frame $\boldsymbol{C}_{\text{PA}}(k)$ of the replicated ambient HOA component will be described in the following.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

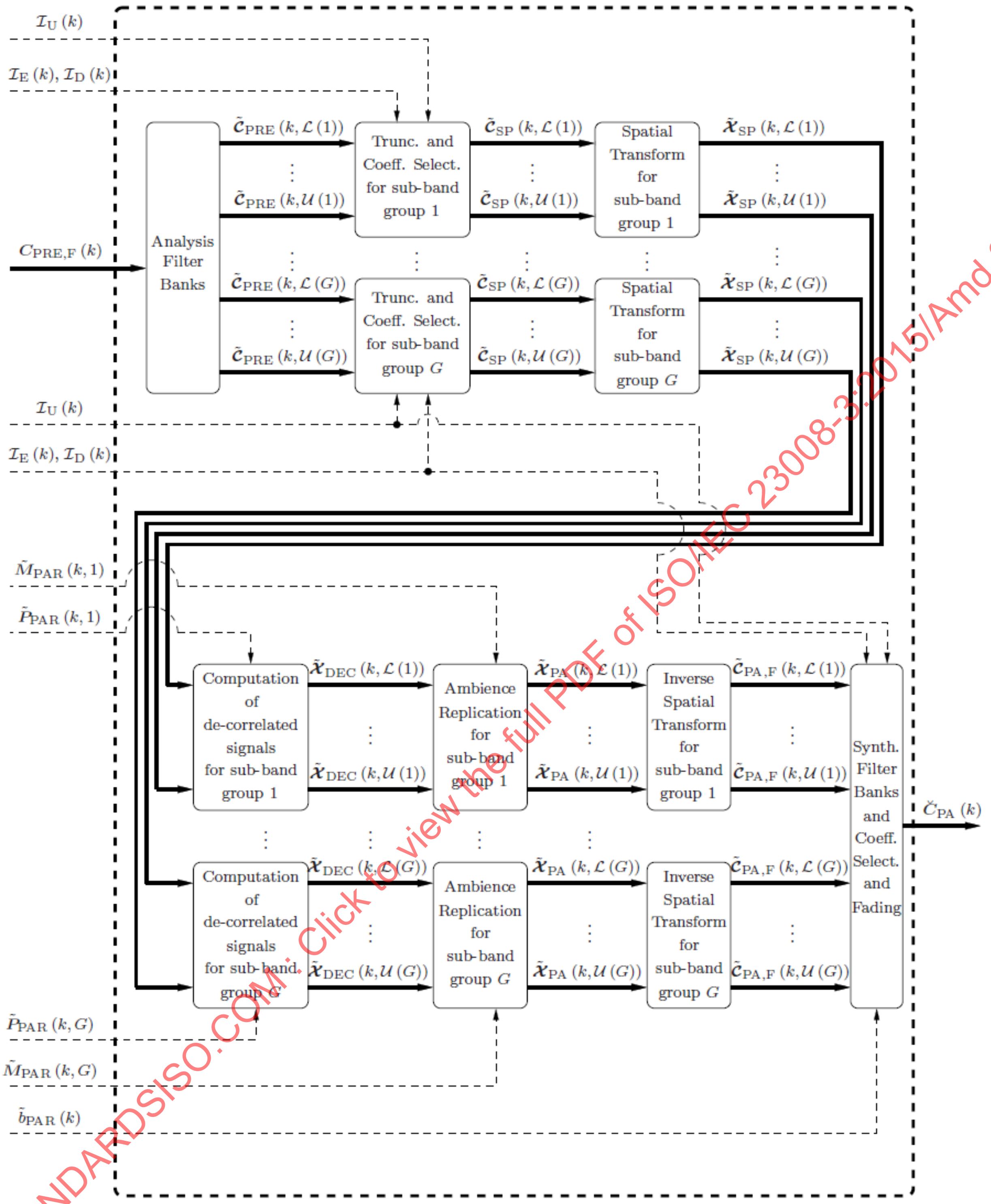

$\mathcal{I}_U(k)$
$\mathcal{I}_E(k), \mathcal{I}_D(k)$
$C_{PRE,F}(k)$
Analysis Filter Banks
$\tilde{c}_{PRE}(k,\mathcal{L}(1))$
$\tilde{c}_{PRE}(k,\mathcal{U}(1))$
$\tilde{c}_{PRE}(k,\mathcal{L}(G))$
$\tilde{c}_{PRE}(k,\mathcal{U}(G))$
Trunc. and Coeff. Select. for sub-band group 1
Trunc. and Coeff. Select. for sub-band group G
$\tilde{c}_{SP}(k,\mathcal{L}(1))$
$\tilde{c}_{SP}(k,\mathcal{U}(1))$
$\tilde{c}_{SP}(k,\mathcal{L}(G))$
$\tilde{c}_{SP}(k,\mathcal{U}(G))$
Spatial Transform for sub-band group 1
Spatial Transform for sub-band group G
$\tilde{x}_{SP}(k,\mathcal{L}(1))$
$\tilde{x}_{SP}(k,\mathcal{U}(1))$
$\tilde{x}_{SP}(k,\mathcal{L}(G))$
$\tilde{x}_{SP}(k,\mathcal{U}(G))$
$\tilde{M}_{PAR}(k,1)$
$\tilde{P}_{PAR}(k,1)$
Computation of de-correlated signals for sub-band group 1
Computation of de-correlated signals for sub-band group G
$\tilde{x}_{DEC}(k,\mathcal{L}(1))$
$\tilde{x}_{DEC}(k,\mathcal{U}(1))$
$\tilde{x}_{DEC}(k,\mathcal{L}(G))$
$\tilde{x}_{DEC}(k,\mathcal{U}(G))$
Ambience Replication for sub-band group 1
Ambience Replication for sub-band group G
$\tilde{x}_{PA}(k,\mathcal{L}(1))$
$\tilde{x}_{PA}(k,\mathcal{U}(1))$
$\tilde{x}_{PA}(k,\mathcal{L}(G))$
$\tilde{x}_{PA}(k,\mathcal{U}(G))$
Inverse Spatial Transform for sub-band group 1
Inverse Spatial Transform for sub-band group G
$\tilde{c}_{PA,F}(k,\mathcal{L}(1))$
$\tilde{c}_{PA,F}(k,\mathcal{U}(1))$
$\tilde{c}_{PA,F}(k,\mathcal{L}(G))$
$\tilde{c}_{PA,F}(k,\mathcal{U}(G))$
Synth. Filter Banks and Coeff. Select. and Fading
$\check{C}_{PA}(k)$
$\tilde{P}_{PAR}(k,G)$
$\tilde{M}_{PAR}(k,G)$
$\tilde{b}_{PAR}(k)$

Figure AMD1.3 — PAR Decoder.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

12.4.2.x.1 Analysis Filter Banks

Each frame $\boldsymbol{c}_{\text{PRE,F},n}(k)$, $n = 1, \dots, O_{\text{PAR,MAX}}$, of the first

$$O_{\text{PAR,MAX}} = \max_{g} O_{\text{PAR}}(g)$$

© ISO/IEC 2017 – All rights reserved

coefficient sequences of the modified version $\boldsymbol{C}_{\mathrm{PRE,F}}(k)$ of the preliminary decoded HOA representation is first decomposed into frames of individual sub-band signals $\tilde{\boldsymbol{c}}_{\mathrm{PRE},n}(k,j)$, $j = 1,\dots,F$. For each sub-band $j = 1,\dots,F$, the frames of the sub-band signals of the individual HOA coefficient sequences are collected into the sub-band HOA representation $\boldsymbol{C}_{\mathrm{PRE}}(k,j)$ as

$$\tilde{\boldsymbol{C}}_{\mathrm{PRE}}(k,j) = \begin{bmatrix} \tilde{\boldsymbol{c}}_{\mathrm{PRE},1}(k,j) \\ \tilde{\boldsymbol{c}}_{\mathrm{PRE},2}(k,j) \\ \vdots \\ \tilde{\boldsymbol{c}}_{\mathrm{PRE},O_{\mathrm{PAR,MAX}}}(k,j) \end{bmatrix} \quad \text{for} \quad j = 1,\dots,F.$$

The filter bank is assumed to be based on Quadrature Mirror Filters (QMF) with a total of $F = 64$ sub-bands (ISO/IEC 23003-1:2007, *Information technology — MPEG audio technologies — Part 1: MPEG Surround*).

12.4.2.x.2 Truncation and Coefficient Selection

For each j-th sub-band belonging to the g-th sub-band group, $g = 1,\dots,G$, a truncated version $\tilde{\boldsymbol{C}}_{\mathrm{SP}}(k,j)$ of $\tilde{\boldsymbol{C}}_{\mathrm{PRE}}(k,j)$ of order $N_{\mathrm{PAR}}(g)$ is computed, which is composed of the individual coefficient sequences according to

$$\tilde{\boldsymbol{C}}_{\mathrm{SP}}(k,j) = \begin{bmatrix} \tilde{\boldsymbol{c}}_{\mathrm{SP},1}(k,j) \\ \tilde{\boldsymbol{c}}_{\mathrm{SP},2}(k,j) \\ \vdots \\ \tilde{\boldsymbol{c}}_{\mathrm{SP},O_{\mathrm{PAR}}(g)}(k,j) \end{bmatrix} \quad \text{for} \quad \mathcal{L}_{\mathrm{PAR}}(g) \le j \le \mathcal{U}_{\mathrm{PAR}}(g) \quad g = 1,\dots,G.$$

The coefficient sequences $\tilde{\boldsymbol{c}}_{\mathrm{SP},n}(k,j)$ with $n = 1,\dots,O_{\mathrm{PAR}}(g)$ of the truncated HOA representation $\tilde{\boldsymbol{C}}_{\mathrm{SP}}(k,j)$ are either taken from $\tilde{\boldsymbol{C}}_{\mathrm{PRE}}(k,j)$ if their index is contained in the index set of the transmitted coefficient sequences of the ambient HOA component, defined by

$$\mathcal{J}_{\mathrm{AMB,ACT}}(k) := \mathcal{J}_{\mathrm{E}}(k) \cup \mathcal{J}_{\mathrm{D}}(k) \cup \mathcal{J}_{\mathrm{U}}(k),$$

or set to zero else, i.e.:

$$\tilde{\boldsymbol{c}}_{\mathrm{SP},n}(k,j) = \begin{cases} \tilde{\boldsymbol{c}}_{\mathrm{PRE},n}(k,j) & \text{if} \quad n \in \mathcal{J}_{\mathrm{AMB,ACT}}(k) \\ \boldsymbol{0} & \text{else} \end{cases}.$$

12.4.2.x.3 Spatial Transform

Each truncated HOA sub-band representation $\tilde{\boldsymbol{C}}_{\mathrm{SP}}(k,j)$ of the j-th sub-band belonging to the g-th sub-band group, $g = 1,\dots,G$, is subjected to a spatial transform, which means the rendering to $O_{\mathrm{PAR}}(g)$ virtual loud-speaker signals $\mathbf{x}_{\mathrm{SP},d}(k,j)$, at the directions $\boldsymbol{\Omega}_d^{(N_{\mathrm{PAR}}(g))}$, $d = 1,\dots,O_{\mathrm{PAR}}(g)$, defined in the tables in the Annexes F.2 – F.11. Arranging the individual virtual loudspeaker signals in the matrix $\tilde{\boldsymbol{X}}_{\mathrm{SP}}(k,j)$ according to

$$\tilde{\boldsymbol{X}}_{\mathrm{SP}}(k,j) = \begin{bmatrix} \boldsymbol{x}_{\mathrm{SP},1}(k,j) \\ \boldsymbol{x}_{\mathrm{SP},2}(k,j) \\ \vdots \\ \boldsymbol{x}_{\mathrm{SP},O_{\mathrm{PAR}}(g)}(k,j) \end{bmatrix} \quad \text{for} \quad \mathcal{L}_{\mathrm{PAR}}(g) \le j \le \mathcal{U}_{\mathrm{PAR}}(g), \quad g = 1,\dots,G,$$

the spatial transform is expressed by means of multiplying the truncated HOA representation $\tilde{\boldsymbol{C}}_{\mathrm{SP}}(k,j)$ with the inverse of the mode matrix $\boldsymbol{\Psi}^{(N_{\mathrm{PAR}}(g),N_{\mathrm{PAR}}(g))}$ (defined in Annex F.1.5) with respect to these directions by

$$\tilde{\boldsymbol{X}}_{\mathrm{SP}}(k,j) = \left(\boldsymbol{\Psi}^{(N_{\mathrm{PAR}}(g),N_{\mathrm{PAR}}(g))}\right)^{-1} \cdot \tilde{\boldsymbol{C}}_{\mathrm{SP}}(k,j) \quad \text{for} \quad \mathcal{L}_{\mathrm{PAR}}(g) \le j \le \mathcal{U}_{\mathrm{PAR}}(g).$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

12.4.2.x.4 Computation of de-correlated sub-band signals

For the computation of the de-correlated signals $\tilde{\mathcal{X}}_{\mathrm{DEC}}(k,j)$ for the j-th sub-band belonging to the g-th sub-band group, the virtual loudspeaker sub-band signals $\tilde{x}_{\mathrm{SP},d}(k,j)$, $d = 1,\dots,O_{\mathrm{PAR}}(g)$ are first assigned to the $O_{\mathrm{PAR}}(g)$ de-correlation filters. To obtain continuous input signals for the de-correlation filters over-lap add processing is employed. In particular, the input signals $\tilde{\mathcal{X}}_{\mathrm{INDEC}}(k,j)$ to the de-correlation filters for the j-th sub-band are computed as the sum of a faded out component and a faded in component:

$$\tilde{\mathcal{X}}_{\mathrm{INDEC}}(k,j) = \tilde{\mathcal{X}}_{\mathrm{INDEC,OUT}}(k,j) + \tilde{\mathcal{X}}_{\mathrm{INDEC,IN}}(k,j).$$

To compute the two individual components, in a first step the instantaneous frame $\tilde{\mathcal{X}}_{\mathrm{INDEC,I}}(k_1;k;j)$ of all permuted virtual loudspeaker sub-band signals of the sparse and truncated HOA sub-band representation $\tilde{\boldsymbol{C}}_{\mathrm{SP}}(k,j)$ for the k-th frame is computed by

$$\tilde{\mathcal{X}}_{\mathrm{INDEC,I}}(k_1;k;j) = \tilde{\boldsymbol{P}}_{\mathrm{PAR}}(k_1,g)\cdot\tilde{\mathcal{X}}_{\mathrm{SP}}(k,j) \quad \text{for} \quad k_1 \in \{k-1,k\} \quad \text{and} \quad \mathcal{L}_{\mathrm{PAR}}(g) \le j \le \mathcal{U}_{\mathrm{PAR}}(g),$$

where $\tilde{\boldsymbol{P}}_{\mathrm{PAR}}(k_1,g)$ is the permutation matrix for the g-th sub-band group including the j-th sub-band and for the (k_1)-th frame, where $k_1 \in \{k-1,k\}$. The sample values of the faded out and faded in components of the input signal frames to the de-correlation filters are determined for $1 \le l \le L_{\mathrm{SB}}$, $\mathcal{L}_{\mathrm{PAR}}(g) \le j \le \mathcal{U}_{\mathrm{PAR}}(g)$, $g = 1,\dots,G$, and $d = 1,\dots,O_{\mathrm{PAR}}(g)$ by

$$\tilde{x}_{\mathrm{INDEC,OUT},d}(k,j;l) = \tilde{x}_{\mathrm{INDEC,I},d}(k-1;k;j;l)\cdot w_{\mathrm{SB}}(L_{\mathrm{SB}}+l)$$

$$\tilde{x}_{\mathrm{INDEC,IN},d}(k,j;l) = \tilde{x}_{\mathrm{INDEC,I},d}(k;k;j;l)\cdot w_{\mathrm{SB}}(l),$$

where $w_{\mathrm{SB}}(l)$ denote the elements of the window function vector $\mathbf{w}_{\mathrm{SB}}$ defined in subclause **12.4.2.x.2 Synthesis of directional sub-band HOA representation for individual sub-band groups**. In a next step, each d-th output signal $\tilde{x}_{\mathrm{DEC},d}(k,j)$, $d = 1,\dots,O_{\mathrm{PAR}}(g)$, is computed by applying to the d-th input sub-band signal $\tilde{x}_{\mathrm{INDEC},d}(k,j)$ one of the ten different 3rd order IIR de-correlation filters (indexed from 0 to 9) as specified in ISO/IEC 23003-1:2007, *Information technology — MPEG audio technologies — Part 1: MPEG Surround* Table A.29, where the index of the used all-pass filter is selected in dependence on the signal index d according to Table **AMD1.11**.

Table AMD1.11 — Assignment of signal index to filter index

Signal index d	Filter set X
1	0
2	1
3	2
4	3
5	4
6	5
7	7
8	8
9	9

12.4.2.x.5 Ambience Replication

The replicated ambient HOA component for the j-th sub-band belonging to the g-th sub-band group, $g = 1,\dots,G$, is represented by means of up-mix signals being virtual loudspeaker sub-band signals $\tilde{x}_{\mathrm{PA},d}(k,j)$ at the directions $\boldsymbol{\Omega}_d^{(N_{\mathrm{PAR}}(g))}$, $d = 1,\dots,O_{\mathrm{PAR}}(g)$, defined in the tables in the Annexes F.2 – F.11. The up-mix sub-band signals are computed by re-assigning the de-correlated signals back to the virtual loudspeaker directions and successively mixing them. For the purpose of signal continuity overlap-add processing on the up-mix

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

signals is carried out. In particular, the up-mix signals $\widetilde{\boldsymbol{\mathcal{X}}}_{\mathrm{PA}}(k,j)$ for the j-th sub-band are computed as the sum of a faded out component and a faded in component:

$$\widetilde{\boldsymbol{\mathcal{X}}}_{\mathrm{PA}}(k,j) = \widetilde{\boldsymbol{\mathcal{X}}}_{\mathrm{PA,OUT}}(k,j) + \widetilde{\boldsymbol{\mathcal{X}}}_{\mathrm{PA,IN}}(k,j).$$

To compute the two individual components, in a first step the instantaneous frame $\widetilde{\boldsymbol{\mathcal{X}}}_{\mathrm{PA,I}}(k_1;k;j)$ of all up-mix sub-band signals for the k-th frame is computed by

$$\widetilde{\boldsymbol{\mathcal{X}}}_{\mathrm{PA,I}}(k_1;k;j) = \widetilde{\boldsymbol{M}}_{\mathrm{PAR}}(k_1,g)\left(\widetilde{\boldsymbol{P}}_{\mathrm{PAR}}(k_1,g)\right)^{-1} \cdot \widetilde{\boldsymbol{\mathcal{X}}}_{\mathrm{DEC}}(k,j)$$

$$\text{for} \quad k_1 \in \{k-1,k\}, \mathcal{L}_{\mathrm{PAR}}(g) \le j \le \mathcal{U}_{\mathrm{PAR}}(g) \quad \text{and} \quad g = 1,\dots,G.$$

Here, $\boldsymbol{P}_{\mathrm{PAR}}(k_1,g)$ denotes the inverse of the permutation matrix describing the re-assignment for the g-th sub-band group and the (k_1)-th frame, where $k_1 \in \{k-1,k\}$. Further, $\boldsymbol{M}_{\mathrm{PAR}}(k_1,g)$ is the corresponding mixing matrix. Assuming the matrices $\boldsymbol{X}_{\mathrm{PA,OUT}}(k,j)$, $\widetilde{\boldsymbol{\mathcal{X}}}_{\mathrm{PA,IN}}(k,j)$, and $\widetilde{\boldsymbol{\mathcal{X}}}_{\mathrm{PA,I}}(k_1;k;d)j$ to be composed of their samples by

$$\widetilde{\boldsymbol{\mathcal{X}}}_{\mathrm{PA,OUT}}(k,j) = \begin{bmatrix} \tilde{x}_{\mathrm{PA,OUT},1}(k,j;1) & \dots & \tilde{x}_{\mathrm{PA,OUT},1}(k,j;L_{\mathrm{SB}}) \\ \vdots & \ddots & \vdots \\ \tilde{x}_{\mathrm{PA,OUT},O_{\mathrm{PAR}}(g)}(k,j;1) & \dots & \tilde{x}_{\mathrm{PA,OUT},O_{\mathrm{PAR}}(g)}(k,j;L_{\mathrm{SB}}) \end{bmatrix} \in \mathbb{C}^{O_{\mathrm{PAR}}(g)\times L_{\mathrm{SB}}},$$

$$\widetilde{\boldsymbol{\mathcal{X}}}_{\mathrm{PA,IN}}(k,j) = \begin{bmatrix} \tilde{x}_{\mathrm{PA,IN},1}(k,j;1) & \dots. & \tilde{x}_{\mathrm{PA,IN},1}(k,j;L_{\mathrm{SB}}) \\ \vdots & \ddots & \vdots \\ \tilde{x}_{\mathrm{PA,IN},O_{\mathrm{PAR}}(g)}(k,j;1) & \dots & \tilde{x}_{\mathrm{PA,IN},O_{\mathrm{PAR}}(g)}(k,j;L_{\mathrm{SB}}) \end{bmatrix} \in \mathbb{C}^{O_{\mathrm{PAR}}(g)\times L_{\mathrm{SB}}},$$

$$\widetilde{\boldsymbol{\mathcal{X}}}_{\mathrm{PA,I}}(k_1;k;j) = \begin{bmatrix} \tilde{x}_{\mathrm{PA,I},1}(k_1;k;j;1) & \dots & \tilde{x}_{\mathrm{PA,I},1}(k_1;k;j;L_{\mathrm{SB}}) \\ \vdots & \ddots & \vdots \\ \tilde{x}_{\mathrm{PA,I},O_{\mathrm{PAR}}(g)}(k_1;k;j;1) & \dots & \tilde{x}_{\mathrm{PA,I},O_{\mathrm{PAR}}(g)}(k_1;k;j;L_{\mathrm{SB}}) \end{bmatrix} \in \mathbb{C}^{O_{\mathrm{PAR}}(g)\times L_{\mathrm{SB}}},$$

the sample values of the faded out and faded in components of the up-mix sub-band signals are determined for $1 \le l \le L_{\mathrm{SB}}$, $\mathcal{L}_{\mathrm{PAR}}(g) \le j \le \mathcal{U}_{\mathrm{PAR}}(g)$, $g = 1,\dots,G$, and $d = 1,\dots,O_{\mathrm{PAR}}(g)$ by

$$\tilde{x}_{\mathrm{PA,OUT},d}(k,j;l) = \tilde{x}_{\mathrm{PA,I},d}(k-1;k;j;l) \cdot w_{\mathrm{SB}}(L_{\mathrm{SB}}+l)$$

$$\tilde{x}_{\mathrm{PA,IN},d}(k,j;l) = \tilde{x}_{\mathrm{PA,I},d}(k;k;j;l) \cdot w_{\mathrm{SB}}(l),$$

where $w_{\mathrm{SB}}(l)$ denote the elements of the window function vector $\mathbf{w}_{\mathrm{SB}}$ defined in subclause **12.4.2.x.2 Synthesis of directional sub-band HOA representation for individual sub-band groups**.

12.4.2.x.6 Inverse Spatial Transform

The virtual loudspeaker signals $\widetilde{\boldsymbol{\mathcal{X}}}_{\mathrm{PA}}(k,j)$ representing the replicated ambient HOA component for the j-th sub-band belonging to the g-th sub-band group, $g = 1,\dots,G$, are subjected to an inverse spatial transform to provide their HOA representation $\widetilde{\boldsymbol{\mathcal{C}}}_{\mathrm{PA,I}}(k,j)$, which is expressed by means of multiplication with the mode matrix $\boldsymbol{\Psi}^{(N_{\mathrm{PAR}}(g),N_{\mathrm{PAR}}(g))}$ (defined in Annex F.1.5) as

$$\widetilde{\boldsymbol{\mathcal{C}}}_{\mathrm{PA,I}}(k,j) = \boldsymbol{\Psi}^{(N_{\mathrm{PAR}}(g),N_{\mathrm{PAR}}(g))} \cdot \widetilde{\boldsymbol{\mathcal{X}}}_{\mathrm{PA}}(k,j) \quad \text{for} \quad \mathcal{L}_{\mathrm{PAR}}(g) \le j \le \mathcal{U}_{\mathrm{PAR}}(g) \quad \text{and} \quad g = 1,\dots,G.$$

The final output HOA representation $\widetilde{\boldsymbol{\mathcal{C}}}_{\mathrm{PA}}(k,j)$ is obtained from $\widetilde{\boldsymbol{\mathcal{C}}}_{\mathrm{PA,I}}(k,j)$ by filling it up with zeros to have an order $N_{\mathrm{PAR,MAX}}$, i.e.

$$\widetilde{\boldsymbol{\mathcal{C}}}_{\mathrm{PA,F}}(k,j) = \begin{bmatrix} \widetilde{\boldsymbol{\mathcal{C}}}_{\mathrm{PA,I}}(k,j) \\ \mathbf{0} \end{bmatrix} \in \mathbb{C}^{N_{\mathrm{PAR,MAX}}\times L_{\mathrm{SB}}} \quad \text{for} \quad j = 1,\dots,F.$$

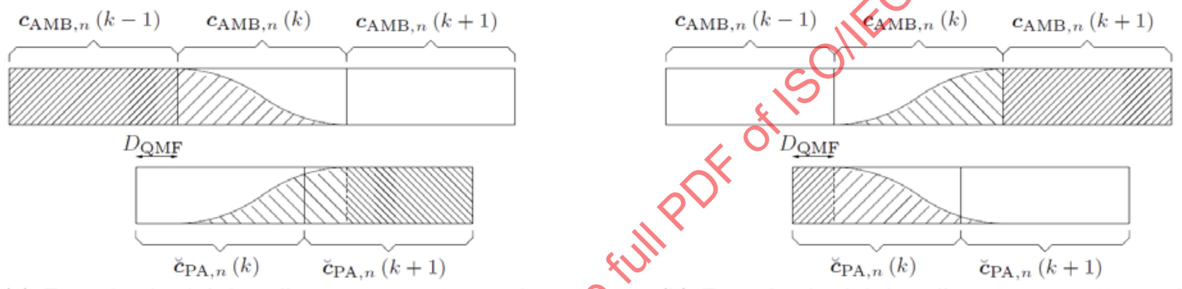
12.4.2.x.7 Synthesis Filter Banks and Coefficient Selection and Fading

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In a first step, the frame $\breve{\boldsymbol{C}}_{\mathrm{PA,F}}(k) \in \mathbb{R}^{O \times L}$ of the preliminary parametrically replicated ambient HOA component is computed as follows:

Its first $O_{\mathrm{PAR,MAX}}$ time domain coefficient sequences $\boldsymbol{c}\breve{}_{\mathrm{PA},n}(k)$, $n = 1, \dots, O_{\mathrm{PAR,MAX}}$, are synthesized from the corresponding sub-band coefficient sequences $\boldsymbol{c}\breve{}_{\mathrm{PA},n}(k,j)$, $j = 1, \dots, F$, by the Synthesis Filter Banks. The remaining time-domain coefficient sequences $\boldsymbol{c}\breve{}_{\mathrm{PA},n}(k)$ with indices $n = O_{\mathrm{PAR,MAX}} + 1, \dots, O$ are set to zero. Note that the synthesized time domain coefficient sequences have a delay of $D_{\mathrm{QMF}} = 577$ samples due to the successive application of the QMF based analysis and synthesis filter banks, which is expressed by the breve symbol above the variables.

In a second step, the frame $\breve{\boldsymbol{C}}_{\mathrm{PA,F}}(k)$ of the preliminary parametrically replicated ambient HOA component is modified to have only contributions for those coefficient sequences, which have not been explicitly transmitted for the ambient HOA component. Further, for those coefficient sequences of the ambient HOA component that are explicitly additionally transmitted and faded in (or faded out), the respective coefficient sequences of the HOA representation $\breve{\boldsymbol{C}}_{\mathrm{PA,F}}(k)$ of the parametrically replicated ambient HOA component have to be modified by fading them out (or fading them in), respectively. Due to the delay between $\boldsymbol{C}_{\mathrm{AMB}}(k)$ and $\breve{\boldsymbol{C}}_{\mathrm{PA,F}}(k)$ of D_{QMF} samples the fading of the coefficient sequences of $\breve{\boldsymbol{C}}_{\mathrm{PA,F}}(k)$ is performed across frame boundaries as illustrated in Figure AMD1.100. The resulting HOA representation of the parametrically replicated ambient HOA component is denoted by $\boldsymbol{C}\breve{}_{\mathrm{PA}}(k)$ with its coefficient sequences $\boldsymbol{c}\breve{}_{\mathrm{PA},n}(k)$, $n = 1, \dots, O$.


$c_{\mathrm{AMB},n}(k-1)$
$c_{\mathrm{AMB},n}(k)$
$c_{\mathrm{AMB},n}(k+1)$
D_{QMF}
$\breve{c}_{\mathrm{PA},n}(k)$
$\breve{c}_{\mathrm{PA},n}(k+1)$
$c_{\mathrm{AMB},n}(k-1)$
$c_{\mathrm{AMB},n}(k)$
$c_{\mathrm{AMB},n}(k+1)$
D_{QMF}
$\breve{c}_{\mathrm{PA},n}(k)$
$\breve{c}_{\mathrm{PA},n}(k+1)$

(a) Example of a faded coefficient sequence $\breve{c}_{\mathrm{PA},n}$ of the HOA representation of the parametrically replicated ambient HOA component in the case the respective coefficient sequence $c_{\mathrm{AMB},n}$ of the ambient HOA component is faded out, i.e. if $n \in \mathcal{I}_{\mathrm{U}}(k-1) \wedge n \in \mathcal{I}_{\mathrm{D}}(k) \wedge n \notin (\mathcal{I}_{\mathrm{U}}(k+1) \cup \mathcal{I}_{\mathrm{E}}(k+1) \cup \mathcal{I}_{\mathrm{D}}(k+1))$

(b) Example of a faded coefficient sequence $\breve{c}_{\mathrm{PA},n}$ of the HOA representation of the parametrically replicated ambient HOA component in the case the respective coefficient sequence $c_{\mathrm{AMB},n}$ of the ambient HOA component is faded in, i.e. if $n \notin (\mathcal{I}_{\mathrm{U}}(k-1) \cup \mathcal{I}_{\mathrm{E}}(k-1) \cup \mathcal{I}_{\mathrm{D}}(k-1)) \wedge n \in \mathcal{I}_{\mathrm{E}}(k) \wedge n \in \mathcal{I}_{\mathrm{U}}(k+1)$

Figure AMD1.100— Illustration of faded coefficient sequences of $\breve{\boldsymbol{C}}_{\mathrm{PA}}(k)$.

For the case that a coefficient sequence $\boldsymbol{c}_{\mathrm{AMB},n}$ of the ambient HOA component is faded out in the k-th frame (i.e. $n \in \mathcal{I}_{\mathrm{D}}(k)$) as illustrated in Figure AMD1.100a, the fade in of the coefficient sequence $\breve{\boldsymbol{c}}_{\mathrm{PA},n}$ in the k-th frame begins D_{QMF} samples later, where in particular the fading in is finished only at the D_{QMF}-th sample of the $(k+1)$-th frame.

Similarly, for the case that a coefficient sequence $\boldsymbol{c}_{\mathrm{AMB},n}$ of the ambient HOA component is faded in in the k-th frame (i.e. $n \in \mathcal{I}_{\mathrm{E}}(k)$) as illustrated in Figure AMD1.100b, the fade out of the coefficient sequence $\breve{\boldsymbol{c}}_{\mathrm{PA},n}$ in the k-th frame begins D_{QMF} samples later, where in particular the fading out is finished only at the D_{QMF}-th sample of the $(k+1)$-th frame.

Finally, it has to be considered that a fade in or fade out of the coefficient sequences $\breve{\boldsymbol{c}}_{\mathrm{PA,F},n}(k)$ of the HOA representation of the composition of all predicted sub-band directional signals is only required if it is not already present, resulting from overlap-add processing.

In the case that $\tilde{b}_{\mathrm{PAR}}(k-1) = 0$ and $\tilde{b}_{\mathrm{PAR}}(k) = 1$, there is already a fade in within each of the k-th frames $\breve{\boldsymbol{c}}_{\mathrm{PA,F},n}(k)$, $n = 1, \dots, O$, such that is not necessary to apply an additional fade in. Similarly, in the case that $\tilde{b}_{\mathrm{PAR}}(k-1) = 1$ and $\tilde{b}_{\mathrm{PAR}}(k) = 0$, there is already a fade out within each of the k-th frames $\breve{\boldsymbol{c}}_{\mathrm{PA,F},n}(k)$, $n = 1, \dots, O$, such that is not necessary to apply an additional fade out. Altogether, the computation of the sample

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

values $\breve{c}_{\mathrm{PA},n}(k,l)$, $n = 1,\dots,O$, $l = 1,\dots,L$, of the coefficient sequences of the HOA representation of the parametrically replicated ambient HOA component is hence formally expressed by

$$\breve{c}_{\mathrm{PA},n}(k,l) = \breve{c}_{\mathrm{PA,F},n}(k,l) \cdot \begin{cases} w_{\mathrm{DIR}}\left(l + L - D_{\mathrm{QMF}}\right) & \text{if } 1 \le l \le D_{\mathrm{QMF}} \wedge n \in \mathcal{J}_{\mathrm{D}}(k-1) \wedge \tilde{b}_{\mathrm{PAR}}(k-2) = 1 \\ w_{\mathrm{DIR}}\left(l + 2L - D_{\mathrm{QMF}}\right) & \text{if } 1 \le l \le D_{\mathrm{QMF}} \wedge n \in \mathcal{J}_{\mathrm{E}}(k-1) \wedge \tilde{b}_{\mathrm{PAR}}(k-1) = 1 \\ 1 & \text{if } 1 \le l \le D_{\mathrm{QMF}} \wedge \\ & \{n \notin \mathcal{J}_{\mathrm{U}}(k-1) \cup \mathcal{J}_{\mathrm{E}}(k-1) \cup \mathcal{J}_{\mathrm{D}}(k-1) \\ & \quad \vee \left(n \in \mathcal{J}_{\mathrm{D}}(k-1) \wedge \tilde{b}_{\mathrm{PAR}}(k-2) = 0\right) \\ & \quad \vee \left(n \in \mathcal{J}_{\mathrm{E}}(k-1) \wedge \tilde{b}_{\mathrm{PAR}}(k-1) = 0\right)\} \\ w_{\mathrm{DIR}}\left(l - D_{\mathrm{QMF}}\right) & \text{if } D_{\mathrm{QMF}} < l \le L \wedge n \in \mathcal{J}_{\mathrm{D}}(k) \wedge \tilde{b}_{\mathrm{PAR}}(k-1) = 1 \\ w_{\mathrm{DIR}}\left(l + L - D_{\mathrm{QMF}}\right) & \text{if } D_{\mathrm{QMF}} < l \le L \wedge n \in \mathcal{J}_{\mathrm{E}}(k) \wedge \tilde{b}_{\mathrm{PAR}}(k) = 1 \\ 1 & \text{if } D_{\mathrm{QMF}} < l \le L \wedge \\ & \{n \notin \mathcal{J}_{\mathrm{U}}(k) \cup \mathcal{J}_{\mathrm{E}}(k) \cup \mathcal{J}_{\mathrm{D}}(k) \\ & \quad \vee \left(n \in \mathcal{J}_{\mathrm{D}}(k) \wedge \tilde{b}_{\mathrm{PAR}}(k-1) = 0\right) \\ & \quad \vee \left(n \in \mathcal{J}_{\mathrm{E}}(k) \wedge \tilde{b}_{\mathrm{PAR}}(k) = 0\right)\} \\ 0 & \text{else} \end{cases} .$$

12.4.2.6 HOA composition

Finally the decoded HOA frame $\boldsymbol{C}(k)$ is computed by

$$\boldsymbol{C}(k) = \boldsymbol{C}_{\mathrm{AMB}}(k) + \boldsymbol{C}_{\mathrm{DIR}}(k) + \boldsymbol{C}_{\mathrm{PD}}(k) + \boldsymbol{C}_{\mathrm{VEC}}(k) = \boldsymbol{C}_{\mathrm{AMB}}(k) + \boldsymbol{C}_{\mathrm{PS}}(k).$$

With:

12.4.2.x Final HOA composition

The frame $\breve{\boldsymbol{C}}(k)$ of the finally reconstructed HOA representation is computed by a superposition of the frame $\boldsymbol{C}_{\mathrm{PRE}}(k)$ of the preliminary decoded HOA representation, the frame $\breve{\boldsymbol{C}}_{\mathrm{D}}(k)$ of the HOA representation of the composition of all predicted sub-band directional signals and the frame $\breve{\boldsymbol{C}}_{\mathrm{PA}}(k)$ of the replicated ambient HOA component. For the superposition the delay between the individual HOA representations to be superposed has to be considered. Hence, the computation of the sample values $\breve{c}_n(k,l)$, $n = 1,\dots,O$, $l = 1,\dots,L$, of the individual coefficient sequences of $\breve{\boldsymbol{C}}(k)$ is given by

$$\breve{c}_n(k,l) = \breve{c}_{\mathrm{D},n}(k,l) + \breve{c}_{\mathrm{PA},n}(k,l) + \begin{cases} c_{\mathrm{PRE},n}\left(k-1, L + l - D_{\mathrm{QMF}}\right) & \text{if } 1 \le l \le D_{\mathrm{QMF}} \\ c_{\mathrm{PRE},n}\left(k, l - D_{\mathrm{QMF}}\right) & \text{if } D_{\mathrm{QMF}} < l \le L \end{cases}.$$

Add new subclause 12.4.X "Layered Coding for HOA" at the end of section 12:

12.4.x Layered Coding for HOA

12.4.x.1 General

This section describes the layered coding for HOA sound field representations. First the structure of the compressed HOA sound field representation is described. Based on this description it is shown how the side information is divided in case of HOA layered coding mode. In the following sub clauses the initialization of the decoder and the behavior of the decoder in case of drop outs of enhancement layers are described.

12.4.x.1 Structure of compressed HOA sound field representation

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In the following the compressed HOA representation is described and structured from the perspective of layered coding. It can be generally decomposed into the following components:

- A basic compressed HOA sound field representation consisting of a number of complementary components (ID_USAC_SCE, ID_USAC_CPE type of usacElementType in Table 26), being the monaural transport signals representing either predominant sound signals or coefficient sequences of the original HOA representation
- Basic side information (ID_EXT_ELE_HOA type of usacExtElementType in Table 8) which describes for each of these monaural signals how it spatially contributes to the sound field. This can be separated into the following two different components:
  - Side information related to individual (monaural) transport signals, which is independent of the existence of other transport signals. Such side information may for instance specify a monaural signal to represent a directional signal (meaning a general plane wave) with a certain direction of incidence. Alternatively, a monaural signal may be specified as a coefficient sequence of the original HOA representation having certain index.
  - Side information related to vector based signals in the mode CodedVVecLength = 1, of which the directional distribution is specified by means of a vector. This side information is dependent on the transmitted coefficient sequences of the original HOA component. In the mentioned mode particular components of this vector are implicitly set to zero and are not part of the compressed vector representation. These components are those with indices equal to those of coefficient sequence of the original HOA representation, which are part of the basic compressed sound field representation. That means that if individual components of the vector are coded, their total number depends on the basic compressed sound field representation, in particular on which coefficient sequences of the original HOA representation it contains.
    If no coefficient sequences of the original HOA representation are contained in the basic compressed sound field representation, the dependent basic side information for each vector-based signal consists of all the vector components and has its greatest size. In the case that coefficient sequences of the original HOA representation with certain indices are added to the basic compressed sound field representation, the vector components with those indices are removed from the side information for each vector-based signal, thereby reducing the size of the dependent basic side information for the vector-based signals.
- Enhancement side information (ID_EXT_ELE_HOA_ENH_LAYER type of usacExtElementType in Table 8) to parametrically improve the basic compressed HOA sound field representation consisting of the following components:
  - Parameters related to the (broadband) spatial prediction to (linearly) predict missing portions of the sound field from the directional signals.
  - Parameters related to the Sub-band Directional Signals Synthesis and the Parametric Ambience Replication, which allow a frequency dependent, parametric prediction of additional monaural signals to be spatially distributed in order to complement a so far spatially incomplete or deficient compressed HOA representation. The prediction is based on coefficient sequences of the basic compressed sound field representation. An important aspect is that the mentioned complementary contribution to the sound field is represented within the compressed HOA representation not by means of additional quantized signals, but rather by means of extra side information of a comparably much smaller size. Hence, the two mentioned coding tools are especially suited for the compression of HOA representations at low data rates.

The layered coding mode is indicated by SingleLayer==0 in the HOADecoderConfig(). Referring to the above described structure of the compressed HOA sound field representation, the base layer for one single frame is composed of the basic side information contained in the HOAFrame(), the payloads containing the transport signals included in the base layer (SCEs and CPEs) and one payload HOAEnhFrame() with enhancement side information.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

The individual $(\mathrm{NumLayers} - 1)$ enhancement layers each contain payloads with additional transport signals (SCEs and CPEs) and one payload HOAEnhFrame() with corresponding enhancement side information adapted to the increased number of transport signals

This structuring of the information contained in the bitstream corresponds to the different types of HOA extension elements provided in the mpegh3daExtElement()s. The same structuring of information can be found in corresponding configuration information, which consists of the configuration information for the HOA transport channels (ID_USAC_SCE and ID_USAC_CPE in Table 8 — Syntax of mpegh3daDecoderConfig()) and the HOA configuration information separated into base (ID_EXT_ELE_HOA) and enhancement (ID_EXT_ELE_HOA_ENH_LAYER) configuration information. Analogously to the payload in the mpegh3daFrame() each layer has its own enhancement configuration information. Whereas only the base layer (**LayerIdx** == 0) additionally contains the basis HOA configuration information. Note: in case of single layer HOA coding the enhancement information in the configuration, as well as in the frame payload is embedded into the basis information and not sent as separate extension payload packet.

Furthermore, it shall be noted that in case of layered coding the information of the sound field is still part of one and the same SignalGroup of SignalGroupTypeHOA.

For illustration of the concept the following Figure X1 shows exemplarily for a two layer HOA coding the ordering of the MHAS packets in a stream.

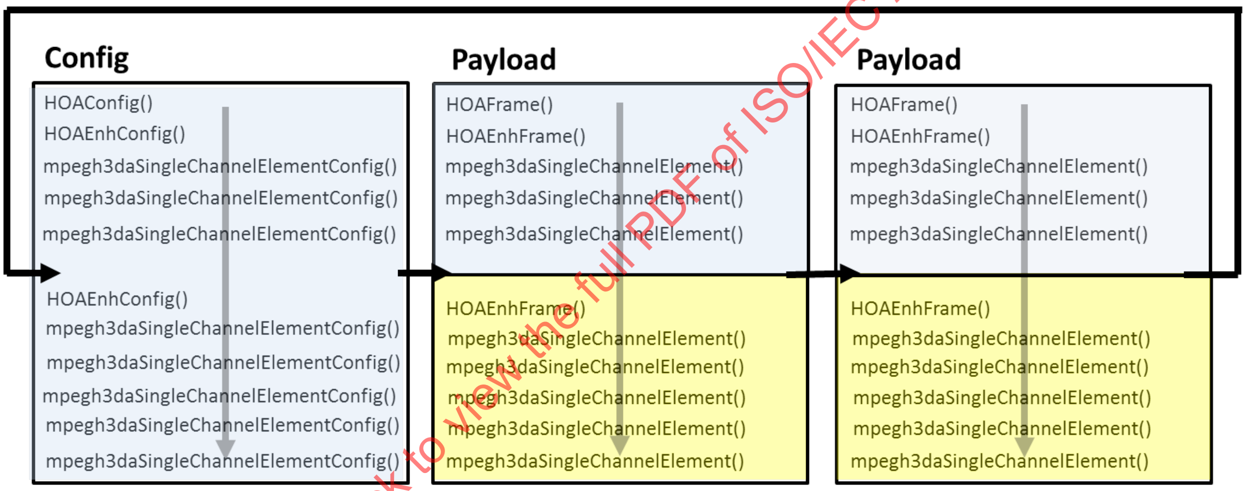

Config
HOAConfig()
HOAEnhConfig()
mpegh3daSingleChannelElementConfig()
mpegh3daSingleChannelElementConfig()
mpegh3daSingleChannelElementConfig()
HOAEnhConfig()
mpegh3daSingleChannelElementConfig()
mpegh3daSingleChannelElementConfig()
mpegh3daSingleChannelElementConfig()
mpegh3daSingleChannelElementConfig()
mpegh3daSingleChannelElementConfig()
Payload
HOAFrame()
HOAEnhFrame()
mpegh3daSingleChannelElement()
mpegh3daSingleChannelElement()
mpegh3daSingleChannelElement()
HOAEnhFrame()
mpegh3daSingleChannelElement()
mpegh3daSingleChannelElement()
mpegh3daSingleChannelElement()
mpegh3daSingleChannelElement()
mpegh3daSingleChannelElement()
Payload
HOAFrame()
HOAEnhFrame()
mpegh3daSingleChannelElement()
mpegh3daSingleChannelElement()
mpegh3daSingleChannelElement()
HOAEnhFrame()
mpegh3daSingleChannelElement()
mpegh3daSingleChannelElement()
mpegh3daSingleChannelElement()
mpegh3daSingleChannelElement()
mpegh3daSingleChannelElement()

Figure X1: Ordering of the MHAS packets for an exemplary HOA layered coding with 2 layers (base and enhancement layer). The packets are sent starting from the left box to the right box and in the boxes from top to bottom. The light blue blocks compile the base layer and the yellow blocks compile the enhancement layer.

The figure clarifies an exception for the general ordering of the payloads in 3daFrame(). Generally, the information of one SignalGroup is ordered in a way, that first all extension payloads are sent followed by the SCE and CPE packets belonging to this group. In case of layered coding this convention is modified to allow for an easy extraction of the packets belonging to different layers. As shown in figure X1 the HOAEnhFrame() packet shall directly precede the SCE or CPE packets belonging to the same layer. The partitioning of the payload information and the ordering shall be the same as signaled in the HOAConfig().

Note: To enable a reasonable division of the HOA transport channels into the base and enhancement layer, described by their assignment to the mpegh3daChannelElments(), the parameter MinAmbHoaOrder in HOADecoderConfig() has to be set to -1 by the encoder. Otherwise the transport channels corresponding to the always transmitted ambiance, described by the minimum ambiance order, would end up in the highest layers, respectively in the last mpegh3daChannelElments(). Typically this information should be sent in the base layer. In case MinAmbHoaOrder=-1 the assignment of the transport channels to the mpegh3daChannelElments() is completely flexible and can be defined by the encoder.

12.4.x.2 Initialization of the Decoder for HOA layered coding

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

As already shown in Figure X1, the complete configuration information shall be sent in the base layer. This ensures that for the initialization of the decoder, e.g. in case of a configuration change, all configuration information is available to initialize the complete decoder including all necessary core decoder instances for all layers. This means the decoder is initialized in the same way as in the single layer mode.

12.4.x.3 Decoder behavior in HOA layered coding mode

The information contained in the different layers is incremental. This means, the information of an enhancement layer can only be meaningfully decoded if all lower layers are present. For example, in case of three layers, the third layer can only be decoded if the base layer and the first enhancement layer are both available. For the HOA spatial decoding process only the HOAEnhFrame() information of the highest available layer, for which all lower layers are also present, may be used. All other HOAEnhFrame() packets should not be used.

In case an enhancement layer is not available, in the loop over all elements of the frame in mpegh3daFrame() the corresponding elements to the missing enhancement layer are not present. To allow for a most similar processing of all decoder instances the missing information starting with the first missing enhancement layer could be replaced by dummy payloads and the core decoder instances could be set to concealment mode and their output signals could be faded out appropriately. The parsing of the HOAEnhFrame() blocks are skipped for non-available layers and the HOAEnhFrame() of the next lower layer shall be used for decoding.

After a dropout of an enhancement layer the decoding can only be started again at a random access point. If the random access point is an IPF, the pre-roll information could be used to seamlessly start the decoding of the newly available HOA transport channels. An appropriate fade in could to be performed on the output signals of the transport channels.

In Annex C.5.3 Spatial HOA Encoding replace:

A possible architecture of the spatial HOA encoder is depicted in the following Figure C.6.

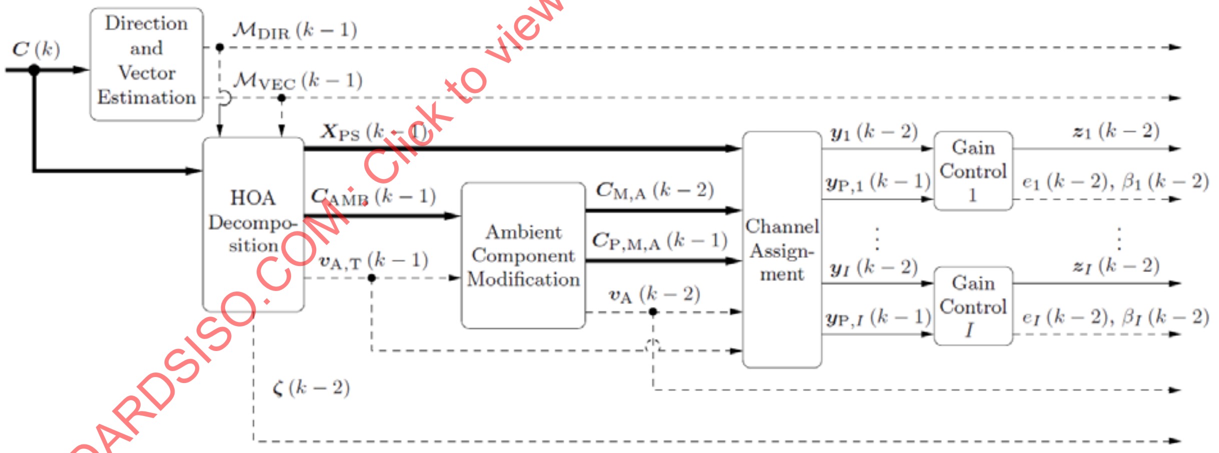


Figure C.6 — Architecture of spatial HOA encoder

First, the k-th frame $\boldsymbol{C}(k)$ of the HOA representation is input to a Direction and Vector Estimation processing block, which is assumed to provide the tuple sets $\mathcal{M}_{\mathrm{DIR}}(k-1)$ and $\mathcal{M}_{\mathrm{VEC}}(k-1)$.

Using both tuple sets $\mathcal{M}_{\mathrm{DIR}}(k-1)$ and $\mathcal{M}_{\mathrm{VEC}}(k-1)$, the initial HOA frame $\boldsymbol{C}(k-1)$ is decomposed in the HOA Decomposition into the frame $\boldsymbol{X}_{\mathrm{PS}}(k-1)$ of all predominant sound (i.e. directional and vector-based) signals and the frame $\boldsymbol{C}_{\mathrm{AMB}}(k-1)$ of the ambient HOA component. Note the delay of one frame, respectively, which is due to overlap add processing in order to avoid blocking artifacts. Furthermore, the HOA Decomposition is assumed to output some prediction parameters $\boldsymbol{\zeta}(k-2)$ related to the predominant sound HOA component computed from the directional signals. Additionally, a target assignment vector $\boldsymbol{v}_{\mathrm{A,T}}(k-1)$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

containing information about the assignment of predominant sound signals, which were determined in the HOA Decomposition processing block, to the I available channels is assumed to be provided. The affected channels can be assumed to be occupied, meaning they are not available to transport any coefficient sequences of the ambient HOA component in the respective time frame.

In the Ambient Component Modification processing block, the frame $\boldsymbol{C}_{\mathrm{AMB}}(k-1)$ of the ambient HOA component is modified according to the information provided by the target assignment vector $\boldsymbol{v}_{\mathrm{A,T}}(k-1)$. In particular, it is determined which coefficient sequences of the ambient HOA component are to be transmitted in the given I channels, depending, amongst other aspects, on the information (contained in the target assignment vector $\boldsymbol{v}_{\mathrm{A,T}}(k-1)$) about which channels are available and not already occupied by predominant sound signals. Additionally, a fade in and out of coefficient sequences is performed if the indices of the chosen coefficient sequences vary between successive frames. The case that only the first O_{MIN} coefficient sequences of the ambient HOA component $\boldsymbol{C}_{\mathrm{AMB}}(k-2)$ are chosen to be perceptually coded corresponds to a reduction of the order of the ambient component from O to O_{MIN}.

Furthermore, it is assumed that the first O_{MIN} coefficient sequences of the ambient HOA component $\boldsymbol{C}_{\mathrm{AMB}}(k-2)$ are always chosen to be perceptually coded to be and to be transmitted. In order to de-correlate these HOA coefficient sequences, it is proposed to transform them to directional signals (i.e. general plane wave functions) impinging from some predefined directions $\boldsymbol{\Omega}_d^{(N_{\mathrm{MIN}})}$, $d = 1, \dots, O_{\mathrm{MIN}}$ related to order N_{MIN}.

Along with from the modified ambient HOA component $\boldsymbol{C}_{\mathrm{M,A}}(k-1)$ a temporally predicted modified ambient HOA component $\boldsymbol{C}_{\mathrm{P,M,A}}(k-1)$ is computed to be later used in the Gain Control processing block in order to allow a reasonable look ahead.

The information about the modification of the ambient HOA component is directly related to the assignment of all possible types of signals to the available channels. The final information about the assignment is assumed to be contained in the final assignment vector $\boldsymbol{v}_{\mathrm{A}}(k-2)$. In order to compute this vector, information contained in the target assignment vector $\boldsymbol{v}_{\mathrm{A,T}}(k-1)$ is exploited.

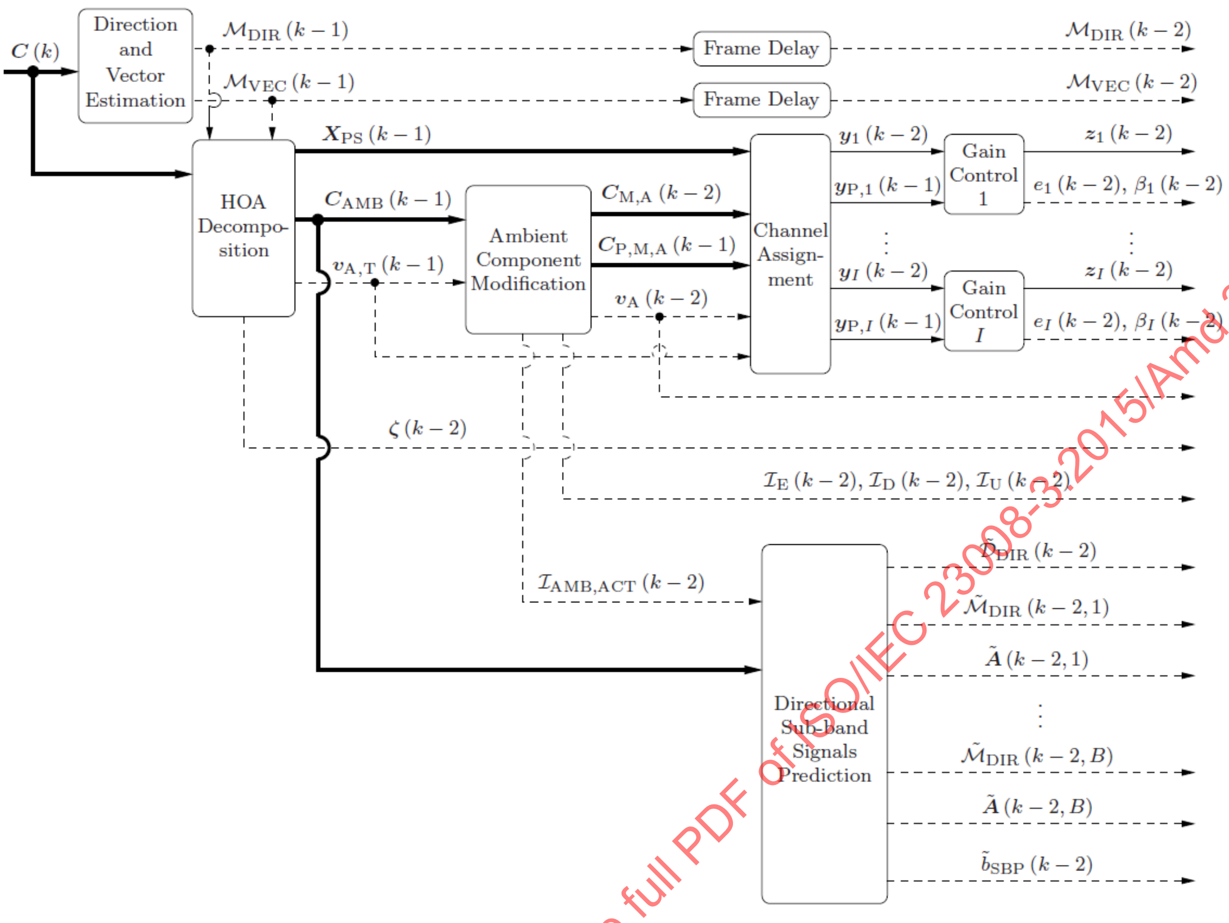
The Channel Assignment assigns with the information provided by the assignment vector $\boldsymbol{v}_{\mathrm{A}}(k-2)$ the appropriate signals contained in $\boldsymbol{X}_{\mathrm{PS}}(k-2)$ and that contained in $\boldsymbol{C}_{\mathrm{M,A}}(k-2)$ to the I available channels, yielding the signals $\boldsymbol{y}_i(k-2)$, $i = 1, \dots, I$. Further, appropriate signals contained in $\boldsymbol{X}_{\mathrm{PS}}(k-1)$ and that in $\boldsymbol{C}_{\mathrm{P,AMB}}(k-1)$ are also assigned to the I available channels, yielding the predicted signals $\boldsymbol{y}_{\mathrm{P},i}(k-2)$, $i = 1, \dots, I$.

Each of the signals $\boldsymbol{y}_i(k-2)$, $i = 1, \dots, I$, is finally processed by a Gain Control, where the signal gain is smoothly modified to achieve a value range that is suitable for the perceptual encoders. The predicted signal frames $\boldsymbol{y}_{\mathrm{P},i}(k-2)$, $i = 1, \dots, I$, allow a kind of look ahead in order to avoid severe gain changes between successive blocks. The gain modifications are assumed to be reverted in the spatial decoder with the gain control side information, consisting of the exponents $e_i(k-2)$ and the exception flags $\beta_i(k-2)$, $i = 1, \dots, I$.

With:

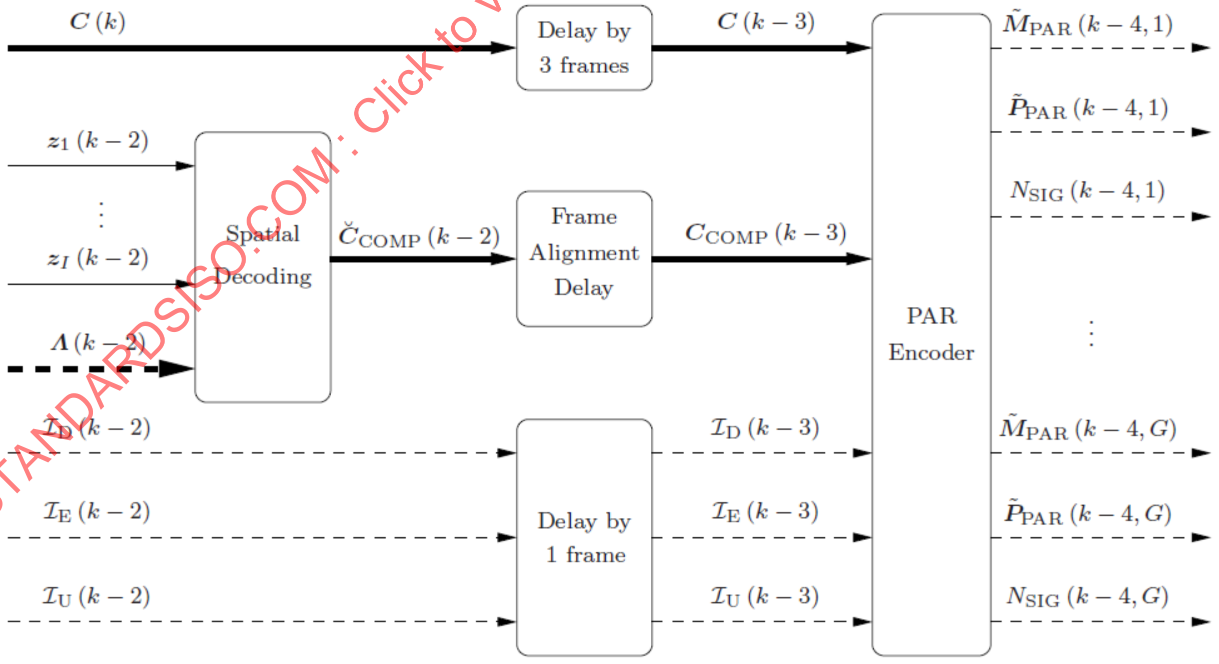
The architecture of the spatial HOA encoder can be separated into two successive parts, which are illustrated in Figure **AMD1.4** and Figure **AMD1.5**, respectively.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017



$C(k)$
Direction and Vector Estimation
$\mathcal{M}_{\mathrm{DIR}}(k-1)$
$\mathcal{M}_{\mathrm{VEC}}(k-1)$
Frame Delay
Frame Delay
$\mathcal{M}_{\mathrm{DIR}}(k-2)$
$\mathcal{M}_{\mathrm{VEC}}(k-2)$
HOA Decomposition
$X_{\mathrm{PS}}(k-1)$
$C_{\mathrm{AMB}}(k-1)$
$v_{\mathrm{A,T}}(k-1)$
Ambient Component Modification
$C_{\mathrm{M,A}}(k-2)$
$C_{\mathrm{P,M,A}}(k-1)$
$v_{\mathrm{A}}(k-2)$
Channel Assignment
$y_1(k-2)$
$y_{\mathrm{P},1}(k-1)$
$y_I(k-2)$
$y_{\mathrm{P},I}(k-1)$
Gain Control 1
Gain Control I
$z_1(k-2)$
$e_1(k-2),\ \beta_1(k-2)$
$z_I(k-2)$
$e_I(k-2),\ \beta_I(k-2)$
$\varsigma(k-2)$
$\mathcal{I}_{\mathrm{E}}(k-2),\ \mathcal{I}_{\mathrm{D}}(k-2),\ \mathcal{I}_{\mathrm{U}}(k-2)$
$\mathcal{I}_{\mathrm{AMB,ACT}}(k-2)$
Directional Sub-band Signals Prediction
$\tilde{D}_{\mathrm{DIR}}(k-2)$
$\tilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,1)$
$\tilde{A}(k-2,1)$
$\tilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,B)$
$\tilde{A}(k-2,B)$
$\tilde{b}_{\mathrm{SBP}}(k-2)$

Figure AMD1.4 — Architecture of spatial HOA encoder (part 1)



$C(k)$
Delay by 3 frames
$C(k-3)$
$z_1(k-2)$
$z_I(k-2)$
$\Lambda(k-2)$
Spatial Decoding
$\check{C}_{\mathrm{COMP}}(k-2)$
Frame Alignment Delay
$C_{\mathrm{COMP}}(k-3)$
$\mathcal{I}_{\mathrm{D}}(k-2)$
$\mathcal{I}_{\mathrm{E}}(k-2)$
$\mathcal{I}_{\mathrm{U}}(k-2)$
Delay by 1 frame
$\mathcal{I}_{\mathrm{D}}(k-3)$
$\mathcal{I}_{\mathrm{E}}(k-3)$
$\mathcal{I}_{\mathrm{U}}(k-3)$
PAR Encoder
$\tilde{M}_{\mathrm{PAR}}(k-4,1)$
$\tilde{P}_{\mathrm{PAR}}(k-4,1)$
$N_{\mathrm{SIG}}(k-4,1)$
$\tilde{M}_{\mathrm{PAR}}(k-4,G)$
$\tilde{P}_{\mathrm{PAR}}(k-4,G)$
$N_{\mathrm{SIG}}(k-4,G)$

Figure AMD1.5 — Architecture of spatial HOA encoder (part 2)

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In the first part, the k-th frame $\boldsymbol{C}(k)$ of the HOA representation is input to a Direction and Vector Estimation processing block, which is assumed to provide the tuple sets $\mathcal{M}_{\mathrm{DIR}}(k-1)$ and $\mathcal{M}_{\mathrm{VEC}}(k-1)$.

Using both tuple sets $\mathcal{M}_{\mathrm{DIR}}(k-1)$ and $\mathcal{M}_{\mathrm{VEC}}(k-1)$, the initial HOA frame $\boldsymbol{C}(k-1)$ is decomposed in the HOA Decomposition into the frame $\boldsymbol{X}_{\mathrm{PS}}(k-1)$ of all predominant sound (i.e. directional and vector-based) signals and the frame $\boldsymbol{C}_{\mathrm{AMB}}(k-1)$ of the ambient HOA component. Note the delay of one frame, respectively, which is due to overlap add processing in order to avoid blocking artifacts. Furthermore, the HOA Decomposition is assumed to output some prediction parameters $\boldsymbol{\zeta}(k-2)$ related to the predominant sound HOA component computed from the directional signals. Additionally, a target assignment vector $\boldsymbol{v}_{\mathrm{A,T}}(k-1)$ containing information about the assignment of predominant sound signals, which were determined in the HOA Decomposition processing block, to the I available channels is assumed to be provided. The affected channels can be assumed to be occupied, meaning they are not available to transport any coefficient sequences of the ambient HOA component in the respective time frame.

In the Ambient Component Modification processing block, the frame $\boldsymbol{C}_{\mathrm{AMB}}(k-1)$ of the ambient HOA component is modified according to the information provided by the target assignment vector $\boldsymbol{v}_{\mathrm{A,T}}(k-1)$. In particular, it is determined which coefficient sequences of the ambient HOA component are to be transmitted in the given I channels, depending, amongst other aspects, on the information (contained in the target assignment vector $\boldsymbol{v}_{\mathrm{A,T}}(k-1)$) about which channels are available and not already occupied by predominant sound signals.

It is assumed that the indices of the selected coefficient sequences to be transmitted are contained in the set $\mathcal{J}_{\mathrm{AMB,ACT}}(k-2)$. They are constrained to be not greater than $O_{\mathrm{MAX}}=(N_{\mathrm{MAX}}+1)^2$, where $N_{\mathrm{MAX}} \le N$ is a predefined order that is specified in the HOAConfig(). Reducing the value of this maximum order can be used to decrease the computational complexity as well as to increase the coding efficiency.

Additionally, a fade in and out of coefficient sequences is performed if the indices of the chosen coefficient sequences vary between successive frames. The indices of ambient HOA coefficient sequences that are selected to be transmitted and supposed to be faded out, faded in or not faded at all in the $(\mathrm{k}-2)$-th frame are contained in the sets $\mathcal{J}_{\mathrm{D}}(k-2)$, $\mathcal{J}_{\mathrm{E}}(k-2)$, and $\mathcal{J}_{\mathrm{U}}(k-2)$, respectively.

Furthermore, it is assumed that the first O_{MIN} coefficient sequences of the ambient HOA component $\boldsymbol{C}_{\mathrm{AMB}}(k-2)$ are always chosen to be perceptually coded to be transmitted. In order to de-correlate these HOA coefficient sequences, it is proposed to transform them to directional signals (i.e. general plane wave functions) impinging from some predefined directions $\boldsymbol{\Omega}_d^{(N_{\mathrm{MIN}})}$, $d=1,\ldots,O_{\mathrm{MIN}}$ related to order N_{MIN}.

Along with the modified ambient HOA component $\boldsymbol{C}_{\mathrm{M,A}}(k-1)$ a temporally predicted modified ambient HOA component $\boldsymbol{C}_{\mathrm{P,M,A}}(k-1)$ is computed to be later used in the Gain Control processing block in order to allow a reasonable look ahead.

The information about the modification of the ambient HOA component is directly related to the assignment of all possible types of signals to the available channels. The final information about the assignment is assumed to be contained in the final assignment vector $\boldsymbol{v}_{\mathrm{A}}(k-2)$. In order to compute this vector, information contained in the target assignment vector $\boldsymbol{v}_{\mathrm{A,T}}(k-1)$ is exploited.

The goal of the Directional Sub-band Signals Prediction, which is carried out in the frequency domain using Quadrature Mirror Filters (QMF) with $F=64$ sub-bands is to approximate the long frame of the ambient HOA component $[\,\boldsymbol{C}_{\mathrm{AMB}}(k-2)\ \ \boldsymbol{C}_{\mathrm{AMB}}(k-1)]$ by a composition of predicted directional sub-band signals. Each directional sub-band signal is assumed to be predicted by a weighted sum of active coefficient sequences of the ambient HOA component, i.e. those coefficient sequences whose indices are contained in the set $\mathcal{J}_{\mathrm{AMB,ACT}}(k-2)$. The idea is that these coefficient sequences will be transmitted within the I transport channels and, hence, will be available at the decompression stage to approximate the non-transmitted coefficient sequences of the ambient HOA component by their predicted versions.

The prediction of each individual directional sub-band signal to be performed at the decompression stage is based on parameters of the corresponding sub-band group including the sub-band of interest. It is assumed

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

that there are F sub-bands that are assigned to B sub-band groups, which are determined by the sub-band group configuration to be specified in the HOAConfig() (see subclause **12.4.1.2.1 Upper and lower bounds of sub-band groups for Sub-band Directional Signals Synthesis**) It defines for each b-th sub-band group a lower index bound $\mathcal{L}(b)$ and an upper index bound $\mathcal{U}(b)$ such that sub-bands with indices between these bounds, i.e. with $\mathcal{L}(b) \leq j \leq \mathcal{U}(b)$, are assumed to belong to this sub-band group. The parameters for each b-th sub-band group, $b = 1, \dots, B$, comprise on the one hand the prediction coefficients matrix $\mathbf{A}(k-2,b)$, which is used to compute the directional sub-band signals from the active coefficient sequences of the ambient HOA component. On the other hand, the parameters include the tuple set $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,b)$ containing direction information to compute the HOA representation of the directional sub-band signals. The first element d of each tuple of the set

$\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,b)$ denotes the index of an active direction trajectory, of which there are at most D_{SB}. The second element $\boldsymbol{\Omega}_{\mathrm{SB},d}(k-2,b)$ of each tuple indicates the corresponding direction. Note that the indexing of the direction trajectories is important to provide continuous directional sub-band signals on the one hand, and to exploit temporal dependencies between successive prediction coefficient matrices $\mathbf{A}(k-2,b)$ for an efficient coding on the other hand.

Further, for an efficient coding of the directions for the individual sub-band groups it is assumed that all of them are contained in the ordered direction set $\tilde{\mathcal{D}}_{\mathrm{DIR}}(k-2)$, of which the number of elements is constrained to be not greater than a predefined number of D_{MAX}, of which a typical value is 8 or 16. Hence, the coding of the directions for the individual sub-band groups may be done by their index of the corresponding direction in the set $\tilde{\mathcal{D}}_{\mathrm{DIR}}(k-2)$.

The Directional Sub-band Signals Prediction also outputs the binary quantity $\tilde{\mathrm{b}}_{\mathrm{SBP}}(\mathrm{k}-2)$ indicating if a prediction of sub-band directional signals is to be performed related to the frames $\mathrm{k}-2$ and $\mathrm{k}-1$ at all.

The Channel Assignment assigns with the information provided by the assignment vector $\boldsymbol{v}_{\mathrm{A}}(k-2)$ the appropriate signals contained in $\boldsymbol{X}_{\mathrm{PS}}(k-2)$ and that contained in $\boldsymbol{C}_{\mathrm{M,A}}(k-2)$ to the I available channels, yielding the signals $\boldsymbol{y}_i(k-2)$, $i = 1, \dots, I$. Further, appropriate signals contained in $\boldsymbol{X}_{\mathrm{PS}}(k-1)$ and that in $\boldsymbol{C}_{\mathrm{P,AMB}}(k-1)$ are also assigned to the I available channels, yielding the predicted signals $\boldsymbol{y}_{\mathrm{P},i}(k-2)$, $i = 1, \dots, I$.

Each of the signals $\boldsymbol{y}_i(k-2)$, $i = 1, \dots, I$, is finally processed by a Gain Control, where the signal gain is smoothly modified to achieve a value range that is suitable for the perceptual encoders. The predicted signal frames $\boldsymbol{y}_{\mathrm{P},i}(k-2)$, $i = 1, \dots, I$, allow a kind of look ahead in order to avoid severe gain changes between successive blocks. The gain modifications are assumed to be reverted in the spatial decoder with the gain control side information, consisting of the exponents $e_i(k-2)$ and the exception flags $\beta_i(k-2)$, $i = 1, \dots, I$.

The second part of the spatial HOA encoder consists of the computation of side information related to Parametric Ambience Replication (PAR). The main idea of PAR is to complement the preliminary encoded HOA representation by potentially missing ambient components, which are parametrically replicated from itself. For that purpose, in a first step the HOA representation $\boldsymbol{C}_{\mathrm{COMP}}(k-2)$ is reconstructed by a spatial decoder using the signals $\boldsymbol{z}_i(k-2)$, $i = 1, \dots, I$, and the side information

$$\boldsymbol{\Lambda}(k-2) := \{\mathcal{M}_{\mathrm{DIR}}(k-2), \mathcal{M}_{\mathrm{VEC}}(k-2), e_1(k-2), \dots, e_I(k-2), \beta_1(k-2), \dots, \beta_{\mathrm{I}}(k-2),$$

$$\boldsymbol{v}_{\mathrm{A}}(k-2), \zeta(k-2), \mathcal{J}_{\mathrm{E}}(k-2), \mathcal{J}_{\mathrm{D}}(k-2), \mathcal{J}_{\mathrm{U}}(k-2),$$

$$\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,1), \dots, \widetilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,B), \boldsymbol{A}(k-2,1), \dots, \boldsymbol{A}(k-2,\mathrm{B}), \tilde{b}_{\mathrm{SBP}}(k-2)\}$$

obtained in the first part of the spatial HOA encoder. This side information does obviously not contain any PAR related components. Hence, the PAR decoder has to be omitted for this special spatial HOA decoding. Due to the successive application of the QMF based analysis and synthesis filter banks in the Sub-band Directional Signals Synthesis of the spatial HOA decoder, the reconstructed HOA representation $\breve{\boldsymbol{C}}_{\mathrm{COMP}}(k-2)$ has a delay of $D_{\mathrm{QMF}} = 577$ samples, which is expressed by the breve symbol above the variable. Hence, in a next step, the reconstructed HOA representation $\breve{\boldsymbol{C}}_{\mathrm{COMP}}(k-2)$ is delayed by $L - D_{\mathrm{QMF}}$ samples to align it with the frames of the original HOA representation. Finally, the original HOA representation (delayed by 3 frames) $\boldsymbol{C}(k-3)$ and the delayed reconstructed HOA representation $\boldsymbol{C}_{\mathrm{COMP}}(k-3)$ together with the index sets

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

$\mathcal{I}_{\mathrm{E}}(k-3)$, $\mathcal{I}_{\mathrm{D}}(k-3)$, and $\mathcal{I}_{\mathrm{U}}(k-3)$ are input to the PAR Encoder processing block, which provides PAR related side information. The PAR is assumed to be carried out in the frequency domain using Quadrature Mirror Filters (QMF) with $F = 64$ sub-bands. Each individual sub-band j, $j = 1, \ldots, F$, is processed using the corresponding parameters of the g-th sub-band group, $g = 1, \ldots, G$, to which it is uniquely assigned. The assignment is determined by the PAR related sub-band group configuration specified in the HOAConfig() (see also subclause **12.4.1.2.2 Upper and lower bounds of sub-band groups for PAR**). It defines for each g-th sub-band group a lower index bound $\mathcal{L}_{\mathrm{PAR}}(g)$ and an upper index bound $\mathcal{U}_{\mathrm{PAR}}(g)$ such that sub-bands with indices between these bounds, i.e. with $\mathcal{L}_{\mathrm{PAR}}(g) \leq j \leq \mathcal{U}_{\mathrm{PAR}}(g)$, are assumed to belong to this sub-band group. The PAR related side information consists of the mixing matrices $\boldsymbol{M}_{\mathrm{PAR}}(k-4, g)$, the permutation matrices $\boldsymbol{P}_{\mathrm{PAR}}(k-4, g)$ and the numbers $N_{\mathrm{SIG}}(k-4, g)$ for the individual G sub-band groups $g = 1, \ldots, G$.

In Annex C.5.3.3 Ambient Component Modification replace:

The Ambient Component Modification processing block has the purpose to appropriately modify the coefficient sequences of the ambient HOA component. In particular, it makes the decision which of the coefficient sequences of the ambient HOA component $\boldsymbol{C}_{\mathrm{AMB}}(k-2)$ are to be chosen to be perceptually coded. Further, a fade in and out of coefficient sequences is performed if the indices of the chosen coefficient sequences vary between successive frames. It is assumed that the first O_{MIN} coefficient sequences of the ambient HOA component $\boldsymbol{C}_{\mathrm{AMB}}(k-2)$ are always chosen to be perceptually coded. In order to de-correlate these HOA coefficient sequences, it is proposed to transform them to directional signals (i.e. general plane wave functions) impinging from some predefined directions $\boldsymbol{\Omega}_{\mathrm{MIN},d}$, $d = 1, \ldots, O_{\mathrm{MIN}}$. The resulting modified frame of the ambient HOA component is denoted by $\boldsymbol{C}_{\mathrm{M,A}}(k-2)$. The information about the choice of the ambient HOA coefficient sequences to be transmitted, about their assignment and about the assignment of the predominant sound signals to the given I channels is assumed to be contained in the assignment vector $\boldsymbol{v}_{\mathrm{A}}(k-2)$.

With:

The Ambient Component Modification processing block has the purpose to appropriately modify the coefficient sequences of the ambient HOA component. In particular, it makes the decision which of the coefficient sequences of the ambient HOA component $\boldsymbol{C}_{\mathrm{AMB}}(k-2)$ are to be chosen to be perceptually coded. Further, a fade in and out of coefficient sequences is performed if the indices of the chosen coefficient sequences vary between successive frames. It is assumed that the first O_{MIN} coefficient sequences of the ambient HOA component $\boldsymbol{C}_{\mathrm{AMB}}(k-2)$ are always chosen to be perceptually coded. In order to de-correlate these HOA coefficient sequences, it is proposed to transform them to directional signals (i.e. general plane wave functions) impinging from some predefined directions $\boldsymbol{\Omega}_{\mathrm{MIN},d}$, $d = 1, \ldots, O_{\mathrm{MIN}}$. The resulting modified frame of the ambient HOA component is denoted by $\boldsymbol{C}_{\mathrm{M,A}}(k-2)$. The information about the choice of the ambient HOA coefficient sequences to be transmitted, about their assignment and about the assignment of the predominant sound signals to the given I channels is assumed to be contained in the assignment vector $\boldsymbol{v}_{\mathrm{A}}(k-2)$. The case that only the first O_{MIN}coefficient sequences of the ambient HOA component $\boldsymbol{C}_{\mathrm{AMB}}(k-2)$are chosen to be perceptually coded corresponds to a reduction of the order of the ambient component from O to O_{MIN}. The indices of ambient HOA coefficient sequences to be transmitted are assumed to be output in the set $\mathcal{I}_{\mathrm{AMB,ACT}}(k-2)$.

In Annex C.5.3.3.1 Computation of the assignment vector replace:

The actual computation of the assignment vector $\boldsymbol{v}_{\mathrm{A}}(k-2)$ is summarized in Table C.1. Along with its computation auxiliary quantities consisting of the sets $\mathcal{I}_{\mathrm{AMB,ACT}}(k-2)$, $\mathcal{I}_{\mathrm{D}}(k-2)$ and $\mathcal{I}_{\mathrm{E}}(k-2)$ are computed, which contain the indices of ambient HOA coefficient sequences that are supposed to be active, disabled and enabled in the $(k-2)$-th frame, respectively. The set $\mathcal{I}_{\mathrm{AMB,ACT}}(-2)$ is assumed to be initialized to an empty set.

Further, during the computation of the assignment vector, the set $\mathcal{I}_{\mathrm{AMB,P}}(k-2)$ of indices of all non-active ambient HOA coefficient sequences which might potentially be activated in the $(k-2)$-th frame is assumed to be determined. This set is assumed to have $\mathcal{N}_{\mathrm{AMB,P}}(k-2)$ elements denoted by $i_{\mathrm{AMB,P,1}}(k-2), \ldots, i_{\mathrm{AMB,P},\mathcal{N}_{\mathrm{AMB,P}}(k-2)}(k-2)$.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

With:

The actual computation of the assignment vector $\boldsymbol{v}_{\mathrm{A}}(k-2)$ is summarized in Table C.1. Along with its computation the set $\mathcal{I}_{\mathrm{AMB,ACT}}(k-2)$ of indices of active ambient HOA coefficient sequences to be transmitted for the $(k-2)$-th frame is computed together with additional auxiliary quantities consisting of the sets $\mathcal{I}_{\mathrm{D}}(k-2)$ and $\mathcal{I}_{\mathrm{E}}(k-2)$, which contain the indices of ambient HOA coefficient sequences that are supposed to be disabled and enabled in the $(k-2)$-th frame, respectively. The set $\mathcal{I}_{\mathrm{AMB,ACT}}(-2)$ is assumed to be initialized to an empty set.

Further, during the computation of the assignment vector, the set $\mathcal{I}_{\mathrm{AMB,P}}(k-2)$ of indices of all non-active ambient HOA coefficient sequences which might potentially be activated in the $(k-2)$-th frame is assumed to be determined. This set is assumed to have $\mathcal{N}_{\mathrm{AMB,P}}(k-2)$ elements denoted by $i_{\mathrm{AMB,P},1}(k-2), \dots, i_{\mathrm{AMB,P},\mathcal{N}_{\mathrm{AMB,P}}(k-2)}(k-2)$. It is in particular assumed to be a subset of $\mathcal{I}_{\mathrm{AMB,ALL}} = \{1, \dots, O_{\mathrm{MAX}}\}$, which denotes the set of indices of ambient HOA coefficient sequences that might be chosen to be transmitted within the I transport channels.

Replace Annex C.5.3.3.2 Computation of the modified ambient HOA component with:

C.5.3.3.2 Computation of the modified ambient HOA component

As already mentioned, it is assumed that the first O_{MIN} coefficient sequences of the ambient HOA component $\boldsymbol{C}_{\mathrm{AMB}}(k-2)$ are always chosen to be perceptually coded. In order to de-correlate these HOA coefficient sequences, it is proposed to apply a transform to them as outlined in subclause C.5.3.3.2.1.

The same is done for the temporally predicted ambient HOA component, where it is assumed that $\boldsymbol{C}_{\mathrm{P,AMB,LOW}}(k-1)$ is obtained from $\boldsymbol{C}_{\mathrm{P,AMB}}(k-1)$ by taking only the first O_{MIN} rows:

$$\boldsymbol{X}_{\mathrm{P,AMB,LOW}}(k-1) = \left(\boldsymbol{\Psi}^{(N_{\mathrm{MIN}},N_{\mathrm{MIN}})}\right)^{-1}\boldsymbol{C}_{\mathrm{P,AMB,LOW}}(k-1).$$

By further assuming the frame $\boldsymbol{C}_{\mathrm{M,A}}(k-2)$ of the modified ambient HOA component and the temporally predicted frame $\boldsymbol{C}_{\mathrm{P,M,A}}(k-1)$ of the modified ambient HOA component to be composed by means of its samples as

$$\boldsymbol{C}_{\mathrm{M,A}}(k-2) = \begin{bmatrix} c_{\mathrm{M,A},1}(k-2,1) & \dots & c_{\mathrm{M,A},1}(k-2,L) \\ \vdots & \ddots & \vdots \\ c_{\mathrm{M,A},O}(k-2,1) & \dots & c_{\mathrm{M,A},O}(k-2,L) \end{bmatrix}$$

and

$$\boldsymbol{C}_{\mathrm{P,M,A}}(k-1) = \begin{bmatrix} c_{\mathrm{P,M,A},1}(k-1,1) & \dots & c_{\mathrm{P,M,A},1}(k-1,L) \\ \vdots & \ddots & \vdots \\ c_{\mathrm{P,M,A},O}(k-1,1) & \dots & c_{\mathrm{P,M,A},O}(k-1,L) \end{bmatrix},$$

the individual samples are computed by

$$c_{\mathrm{M,A},n}(k-2,l) = \begin{cases} x_{\mathrm{AMB,LOW},n}(k-2,l) & \text{if } 1 \le n \le O_{\mathrm{MIN}} \\ c_{\mathrm{AMB},n}(k-2,l)w(l) & \text{if } n \in \mathcal{I}_{\mathrm{E}}(k-2) \\ c_{\mathrm{AMB},n}(k-2,l)w(L+l) & \text{if } n \in \mathcal{I}_{\mathrm{D}}(k-2) \\ c_{\mathrm{AMB},n}(k-2,l) & \text{if } n \in \mathcal{I}_{\mathrm{AMB,ACT}}(k-2)\backslash\left(\mathcal{I}_{\mathrm{E}}(k-2) \cup \mathcal{I}_{\mathrm{D}}(k-2)\right) \\ 0 & \text{else} \end{cases}$$

and

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

$$c_{\mathrm{P,M,A},n}(k-1,l) = \begin{cases} x_{\mathrm{AMB,LOW},n}(k-1,l) & \text{if } 1 \le n \le O_{\mathrm{MIN}} \\ c_{\mathrm{AMB},n}(k-1,l) & \text{if } O_{\mathrm{MIN}} < n \le O. \\ 0 & \text{else} \end{cases}$$

C.5.3.3.2.1 Transform for the O_{MIN} low order coefficient sequences of the ambient HOA component

By default the first O_{MIN} HOA coefficient sequences of $\boldsymbol{C}_{\mathrm{AMB}}(k-1)$ and $\boldsymbol{C}_{\mathrm{P,AMB}}(k-1)$ are transformed to provide the signal frames $\boldsymbol{X}_{\mathrm{AMB,LOW}}(k-2)$ and $\boldsymbol{X}_{\mathrm{P,AMB,LOW}}(k-1)$, respectively, where the transform is described in subclause C.5.3.3.2.2 If N_{MIN} is of value 1, an alternative synthesis method described in . C.5.3.3.2.3 can be used. In the latter case the flag $\mathrm{UsePhaseShiftDecorr}$ has to be set to 1.

C.5.3.3.2.2 Spatial Transform

The first O_{MIN} coefficient sequences of the ambient HOA component are subjected to a spatial transform, where they are transformed to directional signals (i.e. general plane wave functions) impinging from some predefined directions $\boldsymbol{\Omega}_d^{(N_{\mathrm{MIN}})}$, $d = 1,\dots,O_{\mathrm{MIN}}$.

Assuming $\boldsymbol{C}_{\mathrm{AMB,LOW}}(k-2)$ to be the matrix created from $\boldsymbol{C}_{\mathrm{AMB}}(k-2)$ by taking only the first O_{MIN} rows, the transform is given by

$$\boldsymbol{X}_{\mathrm{AMB,LOW}}(k-2) = \left(\boldsymbol{\Psi}^{(N_{\mathrm{MIN}},N_{\mathrm{MIN}})}\right)^{-1} \cdot \boldsymbol{C}_{\mathrm{AMB,LOW}}(k-2),$$

where $\boldsymbol{\Psi}^{(N_{\mathrm{MIN}},N_{\mathrm{MIN}})}$ denotes the mode matrix of order N_{MIN} with respect to the predefined directions $\boldsymbol{\Omega}_n^{(N_{\mathrm{MIN}})}, n = 1,\dots,O_{\mathrm{MIN}}$, defined in Annex F.1.5.

The same is done for the temporally predicted ambient HOA component, where it is assumed that $\boldsymbol{C}_{\mathrm{P,AMB,LOW}}(k-1)$ is obtained from $\boldsymbol{C}_{\mathrm{P,AMB}}(k-1)$ by taking only the first O_{MIN} rows:

$$\boldsymbol{X}_{\mathrm{P,AMB,LOW}}(k-1) = \left(\boldsymbol{\Psi}^{(N_{\mathrm{MIN}},N_{\mathrm{MIN}})}\right)^{-1} \boldsymbol{C}_{\mathrm{P,AMB,LOW}}(k-1).$$

C.5.3.3.2.3 Phase-based transform

The phase-based transform for the first $\mathrm{O_{MIN}}$ HOA coefficient sequences of $\mathrm{C_{AMB}}(\mathrm{k}-1)$is defined by

$$\begin{bmatrix} \mathrm{x_{AMB,LOW,1}}(\mathrm{k}-2) \\ \mathrm{x_{AMB,LOW,2}}(\mathrm{k}-2) \\ \mathrm{x_{AMB,LOW,3}}(\mathrm{k}-2) \\ \mathrm{x_{AMB,LOW,4}}(\mathrm{k}-2) \end{bmatrix} = \begin{bmatrix} \mathrm{d}(9) \cdot (\mathrm{S}(\mathrm{k}-2) + \mathrm{M}(\mathrm{k}-2)) \\ \mathrm{d}(9) \cdot (\mathrm{M}(\mathrm{k}-2) - \mathrm{S}(\mathrm{k}-2)) \\ \mathrm{d}(8) \cdot (\mathrm{B}_{+90}(\mathrm{k}-2) + \mathrm{d}(5) \cdot \boldsymbol{c}_{\mathrm{AMB},2}(\mathrm{k}-2)) \\ \boldsymbol{c}_{\mathrm{AMB},3}(\mathrm{k}-2) \end{bmatrix},$$

with the coefficients d as defined in Table 36, the signal frames $S(k-2)$ and $M(k-2)$ being defined by

$$\begin{aligned} \mathrm{S}(\mathrm{k}-2) &= \mathrm{A}_{+90}(\mathrm{k}-2) + \mathrm{d}(6) \cdot \mathrm{c_{AMB,2}}(\mathrm{k}-2) \\ \mathrm{M}(\mathrm{k}-2) &= \mathrm{d}(4) \cdot \mathrm{c_{AMB,1}}(\mathrm{k}-2) + \mathrm{d}(5) \cdot \mathrm{c_{AMB,4}}(\mathrm{k}-2) \end{aligned}$$

and $A_{+90}(k-2)$ and $B_{+90}(k-2)$ are the frames of +90 degree phase shifted signals A and B defined by

$$\begin{aligned} \mathrm{A}(\mathrm{k}-2) &= \mathrm{d}(0) \cdot \mathrm{c_{AMB,LOW,1}}(\mathrm{k}-2) + \mathrm{d}(1) \cdot \mathrm{c_{AMB,4}}(\mathrm{k}-2) \\ \mathrm{B}(\mathrm{k}-2) &= \mathrm{d}(2) \cdot \mathrm{c_{AMB,LOW,1}}(\mathrm{k}-2) + \mathrm{d}(3) \cdot \mathrm{c_{AMB,4}}(\mathrm{k}-2)^{\cdot} \end{aligned}$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

The phase-based transform for the first O_{MIN} HOA coefficient sequences of $\boldsymbol{C}_{\mathrm{P,AMB}}(k-1)$ is defined accordingly. Note that this kind of transform introduces a delay of one frame.

Table AMD1.12 — Coefficients for phase-based transform

n	d(n)
0	0.34202009999999999
1	0.41629927335044281
2	0.14319999999999999
3	0.53170257350013528
4	0.93969259999999999
5	0.15152053650908184
6	0.53517399036360758
7	0.57735026918962584
8	0.94060406122874030
9	0.500000000000000

Add following subclauses after C.5.3.3 Ambient Component Modification:

C.5.3.x Directional Sub-bands Signals Prediction

The purpose of the Directional Sub-band Signals Prediction is to approximate the ambient HOA component by a composition of directional sub-band signals, which are predicted by a weighted sum of those coefficient sequences of the ambient HOA component that are supposed to be transmitted within the I given transport channels, i.e. the coefficient sequences with indices contained in the set $\mathcal{I}_{\mathrm{AMB,ACT}}(k-2)$. The idea is that these coefficient sequences will be available at the decompression stage to approximate the non-transmitted coefficient sequences of the ambient HOA component by their predicted versions.

The prediction of each individual directional sub-band signal to be performed at the decompression stage is based on parameters of the corresponding sub-band group including the sub-band of interest. It is assumed that there are F sub-bands that are assigned to B sub-band groups, which are determined by the sub-band group configuration to be specified in the HOAConfig() (see subclause **12.4.1.2.1 Upper and lower bounds of sub-band groups for Sub-band Directional Signals Synthesis**). It defines for each b-th sub-band group a lower index bound $\mathcal{L}(b)$ and an upper index bound $\mathcal{U}(b)$ such that sub-bands with indices between these bounds, i.e. with $\mathcal{L}(b) \leq j \leq \mathcal{U}(b)$, are assumed to belong to this sub-band group.

The parameters for each b-th sub-band group, $b = 1, \ldots, B$, comprise on the one hand the prediction coefficients matrix $\mathbf{A}(k-2,b)$, which is used to compute the directional sub-band signals from the active coefficient sequences of the ambient HOA component. On the other hand, the parameters include the tuple set $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,b)$ containing direction information to compute the HOA representation of the directional sub-band signals. The first element d of each tuple of the set $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,b)$ denotes the index of an active direction trajectory, of which there are at most D_{SB}. The second element $\boldsymbol{\Omega}_{\mathrm{SB},d}(k-2,b)$ of each tuple indicates the corresponding direction. Note that the indexing of the direction trajectories is important to provide continuous directional sub-band signals on the one hand, and to exploit temporal dependencies between successive prediction coefficient matrices $\mathbf{A}(k-2,b)$ for an efficient coding on the other hand.

Further, for an efficient coding of the individual sub-band directions it is assumed that all of them are contained in the ordered direction set $\widetilde{\mathcal{D}}_{\mathrm{DIR}}(k-2)$, of which the number of elements is constrained to be not greater than a predefined number of D_{PRED}, of which a typical value is 8 or 16. Hence, the coding of the individual sub-band directions may be done by their index of the corresponding direction in the set $\widetilde{\mathcal{D}}_{\mathrm{DIR}}(k-2)$.

In order to avoid artifacts in the predicted directional sub-band signals due to changes of the estimated directions and prediction coefficients between successive frames, the prediction is performed on concatenated

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

long frames consisting of two temporally successive frames. In particular, that means that each quantity $\widetilde{\mathcal{D}}_{\mathrm{DIR}}(k-2)$, $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,b)$ and $\mathbf{A}(k-2,b)$, $b=1,\dots,B$, is related to the $(k-1)$-th and $(k-2)$-th frame. At decompression, these parameters are then assumed to be used to perform overlap add processing with the predicted directional sub-band signals.

Note that in the absence of predominant sound signals the ambient component corresponds to a "truncated" version of the original HOA representation. Truncation in this context means that the original HOA representation is approximated by only I of its total O coefficient sequences, i.e. by those that are chosen to be transmitted within the I transport channels.

A possible architecture for the Directional Sub-band Signals Prediction is illustrated in Figure **AMD1.6**. The individual processing units to compute the prediction parameters will be described in the following.

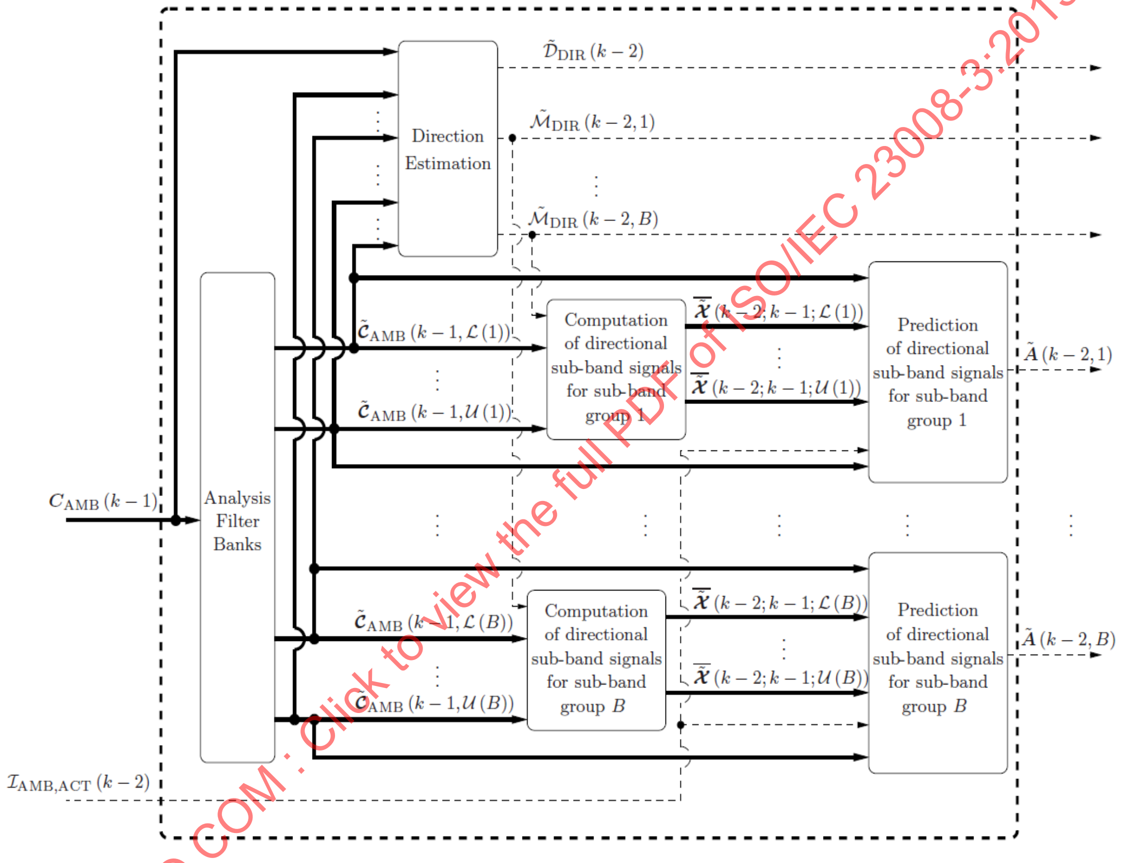

$\widetilde{\mathcal{D}}_{\mathrm{DIR}}(k-2)$
$\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,1)$
$\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,B)$
Direction Estimation
$C_{\mathrm{AMB}}(k-1)$
Analysis Filter Banks
$\tilde{C}_{\mathrm{AMB}}(k-1,\mathcal{L}(1))$
$\tilde{C}_{\mathrm{AMB}}(k-1,\mathcal{U}(1))$
Computation of directional sub-band signals for sub-band group 1
$\widetilde{\mathcal{X}}(k-2;k-1;\mathcal{L}(1))$
$\widetilde{\mathcal{X}}(k-2;k-1;\mathcal{U}(1))$
Prediction of directional sub-band signals for sub-band group 1
$\tilde{A}(k-2,1)$
$\tilde{C}_{\mathrm{AMB}}(k-1,\mathcal{L}(B))$
$\tilde{C}_{\mathrm{AMB}}(k-1,\mathcal{U}(B))$
Computation of directional sub-band signals for sub-band group B
$\widetilde{\mathcal{X}}(k-2;k-1;\mathcal{L}(B))$
$\widetilde{\mathcal{X}}(k-2;k-1;\mathcal{U}(B))$
Prediction of directional sub-band signals for sub-band group B
$\tilde{A}(k-2,B)$
$\mathcal{I}_{\mathrm{AMB,ACT}}(k-2)$

Figure AMD1.6 — Directional sub-band signals prediction

C.5.3.x.1 Analysis Filter Banks

Each frame $\boldsymbol{c}_{\mathrm{AMB},n}(k-1)$, $n=1,\dots,O$, of an individual coefficient sequence of the ambient HOA representation $\boldsymbol{C}_{\mathrm{AMB}}(k-1)$ is first decomposed into frames of individual sub-band signals $\tilde{\boldsymbol{c}}_{\mathrm{AMB},n}(k-1,j)$, $j=1,\dots,F$. For each sub-band j, $j=1,\dots,F$, the frames of the sub-band signals of the individual HOA coefficient sequences are collected into the sub-band HOA representation $\tilde{\boldsymbol{C}}_{\mathrm{AMB}}(k-1,j)$ as

$$\tilde{\boldsymbol{C}}_{\mathrm{AMB}}(k-1,j)=\begin{bmatrix}\tilde{\boldsymbol{c}}_{\mathrm{AMB},1}(k-1,j)\\ \tilde{\boldsymbol{c}}_{\mathrm{AMB},2}(k-1,j)\\ \vdots\\ \tilde{\boldsymbol{c}}_{\mathrm{AMB},O}(k-1,j)\end{bmatrix}\quad \text{for}\quad j=1,\dots,F.$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

The filter bank is assumed to be based on Quadrature Mirror Filters (QMF) with a total of $F = 64$ sub-bands, which are also employed for perceptual coding. Note that, in contrast to the HOA coefficient sequences $\boldsymbol{c}_{\mathrm{AMB},n}(k-1)$ their sub-band representations $\tilde{\boldsymbol{c}}_{\mathrm{AMB},n}(k-1,j)$ are complex valued in general. Further, the sub-band signals are decimated in time compared to the original time-domain signals by a factor of F. As a consequence, the number of samples in the frames $\tilde{\boldsymbol{c}}_{\mathrm{AMB},n}(k-1,j)$ is $L_{\mathrm{SB}} := L/F$. It is assumed that L is an integral multiple of F to assure that L_{SB} has a positive integer value.

C.5.3.x.2 Direction Estimation

The direction estimation processing block has the purpose to analyze the input HOA representation and compute for each b-th sub-band group, $j = 1, \dots, B$, a set $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,b)$ of tuples specifying directions of sub-band general plane wave functions with a major contribution to the sound field for sub-bands belonging to the sub-band group. In this context the term "major contribution" may for instance refer to the most power compared to sub-band general plane waves impinging from other directions. However, it may also refer to a high relevance in terms of the human perception.

In order to avoid artifacts in the predicted directional sub-band signals at decompression due to changes of the estimated directions and prediction coefficients between successive frames, the direction estimation and the prediction of directional sub-band signals are supposed to be performed on concatenated long frames, which consist of the $(k-1)$-th and $(k-2)$-th input frame. At decompression, the quantities estimated on these long frames are then assumed to be used to perform overlap add processing with the predicted directional sub-band signals.

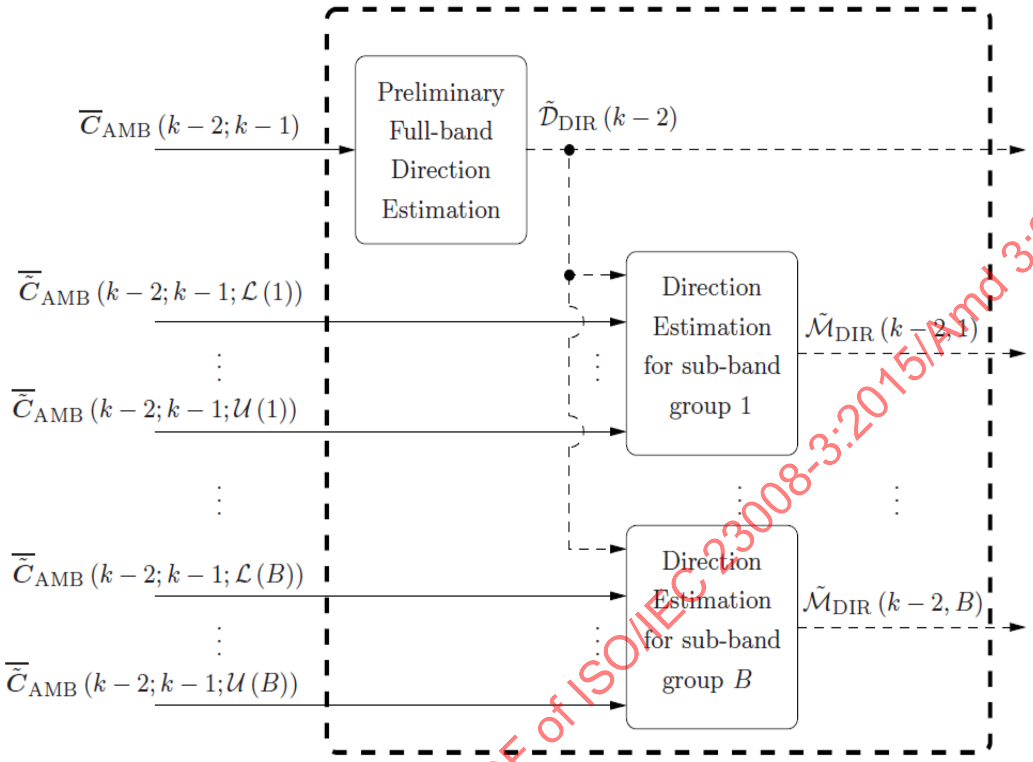
The direction estimation for sub-bands j related to the b-th sub-band group is assumed to provide the set $\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,b)$ of tuples consisting on the one hand of the indices $d \in \tilde{\mathcal{J}}_{\mathrm{DIR}}(k-2,b)$ identifying the individual (active) direction trajectories as well as on the other hand the respective estimated directions $\boldsymbol{\Omega}_{\mathrm{SB},d}(k-2,b)$, i.e.

$$\widetilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,b) = \left\{ \left(d, \boldsymbol{\Omega}_{\mathrm{SB},d}(k-2,b) \right) \middle| \, d \in \tilde{\mathcal{J}}_{\mathrm{DIR}}(k-2,b) \right\}.$$

The index set $\tilde{\mathcal{J}}_{\mathrm{DIR}}(k-2,b)$ is assumed to be a subset of $\{1, \dots, D_{\mathrm{SB}}\}$, where D_{SB} is the maximum number of possible directions per sub-band group, and hence per sub-band. The value of D_{SB} can be specified in the HOAConfig() and is typically small for the reason of coding efficiency. In this context, the indexing of the direction trajectories is important to provide continuous directional sub-band signals on the one hand, and to exploit temporal dependencies between successive prediction coefficient matrices $\mathbf{A}(k-2,b)$ for an efficient coding on the other hand. To further increase the coding efficiency for the side information, the individual directions $\boldsymbol{\Omega}_{\mathrm{SB},d}(k-2,b)$ of all sub-band groups are constrained to be contained in the direction set $\tilde{\mathcal{D}}_{\mathrm{DIR}}(k-2)$, which itself is constrained to contain not more than D_{PRED} directions. The value of D_{PRED} can be specified in the HOAConfig() as a power to the base 2, of which a typical value is 16 or 8. Hence, the coding of the directions for the individual sub-band groups may be done by their index of the corresponding direction in the set $\tilde{\mathcal{D}}_{\mathrm{DIR}}(k-2)$. One possible idea for the direction estimation is illustrated in Figure **AMD1.7** .

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved



$\overline{C}_{\mathrm{AMB}}(k-2;k-1)$
Preliminary Full-band Direction Estimation
$\tilde{\mathcal{D}}_{\mathrm{DIR}}(k-2)$
$\tilde{\overline{C}}_{\mathrm{AMB}}(k-2;k-1;\mathcal{L}(1))$
$\tilde{\overline{C}}_{\mathrm{AMB}}(k-2;k-1;\mathcal{U}(1))$
Direction Estimation for sub-band group 1
$\tilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,1)$
$\tilde{\overline{C}}_{\mathrm{AMB}}(k-2;k-1;\mathcal{L}(B))$
$\tilde{\overline{C}}_{\mathrm{AMB}}(k-2;k-1;\mathcal{U}(B))$
Direction Estimation for sub-band group B
$\tilde{\mathcal{M}}_{\mathrm{DIR}}(k-2,B)$

Figure AMD1.7 — Proposed architecture of direction estimation

In a first step a full-band direction estimation is performed on a direction grid consisting of Q test directions $\boldsymbol{\Omega}_q^{(\sqrt{Q}-1)}$, $q = 1,\dots,Q$ (with Q set in the HOAConfig() and the directions defined in Annexes F.2 – F.11) using the concatenated long frame

$$\overline{\boldsymbol{C}}_{\mathrm{AMB}}(k-2;k-1) = [\boldsymbol{C}_{\mathrm{AMB}}(k-2) \quad \boldsymbol{C}_{\mathrm{AMB}}(k-1)]$$

consisting of the previous and current input frames, $\boldsymbol{C}_{\mathrm{AMB}}(k-2)$ and $\boldsymbol{C}_{\mathrm{AMB}}(k-1)$, of the full-band HOA representation of the ambient component. The number Q of test directions for the direction estimation is to be specified in the HOAConfig().

This direction search is assumed to provide a number of $D_{\mathrm{FB}}(k-2) \le D_{\mathrm{PRED}}$ direction candidates $\boldsymbol{\Omega}_{\mathrm{CA},d}(k-2)$, $d = 1,\dots,D_{\mathrm{FB}}(k-2)$, which are supposed to be contained in the set $\tilde{\mathcal{D}}_{\mathrm{CA}}(k-2)$, i.e.

$$\tilde{\mathcal{D}}_{\mathrm{CA}}(k-2) = \{\boldsymbol{\Omega}_{\mathrm{CA},1}(k-2),\dots,\boldsymbol{\Omega}_{\mathrm{CA},D_{\mathrm{FB}}(k-2)}(k-2)\}.$$

A typical value for the maximum number of direction candidates per frame is $D_{\mathrm{PRED}} = 16$. The direction estimation can be accomplished e.g. by combining the information obtained from a directional power distribution of the input HOA representation together with a simple source movement model for the Bayesian inference of the directions.

In a second step, the direction search is carried out for each individual sub-band group, however not on the initial direction grid consisting of Q test directions, but rather on the candidate set $\tilde{\mathcal{D}}_{\mathrm{CA}}(k-2)$. The number of directions for the b-th sub-band, $b = 1,\dots,B$, denoted by $D_{\mathrm{SB}}(k-2,b)$, is assumed to be not greater than D_{SB}, which is typically distinctly smaller than D_{PRED}. As the full-band version, the direction search for each individual b-th sub-band group, $b = 1,\dots,B$, is also supposed to be performed on long concatenated frames of sub-band signals belonging to this sub-band group

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

$$\bar{\tilde{\boldsymbol{c}}}_{\mathrm{AMB}}(k-2;k-1;j) = [\tilde{\boldsymbol{c}}_{\mathrm{AMB}}(k-2,j) \quad \tilde{\boldsymbol{c}}_{\mathrm{AMB}}(k-1,j)] \quad j = \mathcal{L}(b),\dots,\mathcal{U}(b)$$

consisting of the previous and current frame. In a last step, the desired set $\tilde{\mathcal{D}}_{\mathrm{DIR}}(k-2)$ of all full-band direction candidates, which do actually occur as sub-band directions, is determined as

$$\tilde{\mathcal{D}}_{\mathrm{DIR}}(k-2) := \left\{ \boldsymbol{\Omega}_{\mathrm{CA},d}(k-2) \middle| \, \exists b \in \{1,\dots,B\} \text{ and } d \in \tilde{\mathcal{J}}_{\mathrm{DIR}}(k-2,b) \; s.\,t. \; \boldsymbol{\Omega}_{\mathrm{CA},d}(k-2) = \boldsymbol{\Omega}_{\mathrm{SB},d}(k-2,b) \right\}.$$

To be able to refer to them, the directions of the set $\tilde{\mathcal{D}}_{\mathrm{DIR}}(k-2)$ are finally denoted by $\boldsymbol{\Omega}_{\mathrm{JOINED},d}(k-2)$, $d = 1,\dots,D_{\mathrm{JOINED}}(k-2)$ with $D_{\mathrm{JOINED}}(k-2)$ denoting their number.

C.5.3.x.3 Computation of directional sub-band signals for individual sub-band groups

This processing block is assumed to compute for each b-th sub-band group, $b = 1,\dots,B$, long frames of directional sub-band signals $\bar{\tilde{x}}_d(k-2;k-1;j)$, $d = 1,\dots,D_{\mathrm{SB}}$, related to the $(k-2)$-th and $(k-1)$-th frame for each sub-band j contained in this sub-band group, i.e. for $j = \mathcal{L}(b),\dots,\mathcal{U}(b)$.

For a clearer presentation of the computation, all potential D_{SB} directional signals for each j-th sub-band of a sub-band group b are arranged in the matrix $\bar{\tilde{\boldsymbol{X}}}(k-2;k-1;j)$ as

$$\bar{\tilde{\boldsymbol{X}}}(k-2;k-1;j) = \begin{bmatrix} \bar{\tilde{x}}_1(k-2;k-1;j) \\ \bar{\tilde{x}}_2(k-2;k-1;j) \\ \vdots \\ \bar{\tilde{x}}_{D_{\mathrm{SB}}}(k-2;k-1;j) \end{bmatrix} \in \mathbb{C}^{D_{\mathrm{SB}} \times 2L_{\mathrm{SB}}}.$$

It is assumed that the frames of the inactive directional sub-band signals, i.e. those long signal frames $\bar{\tilde{x}}_d(k-2;k-1;j)$ whose index d is not contained within the set $\tilde{\mathcal{J}}_{\mathrm{DIR}}(k-2,b)$ for the corresponding sub-band group b, are set to zero.

The remaining long signal frames $\bar{\tilde{x}}_d(k-2;k-1;j)$, i.e. those with index $d \in \tilde{\mathcal{J}}_{\mathrm{DIR}}(k-1,b)$, are assumed to be collected within the matrix $\bar{\tilde{\boldsymbol{X}}}_{\mathrm{ACT}}(k-2;k-1;j) \in \mathbb{C}^{D_{\mathrm{SB}}(k-1,b) \times 2L_{\mathrm{SB}}}$. One possibility to compute the active directional sub-band signals contained therein is to minimize the error between their HOA representation and the sub-band HOA representation of the input ambient component. The solution is given by

$$\bar{\tilde{\boldsymbol{X}}}_{\mathrm{ACT}}(k-2;k-1;j) = \left(\boldsymbol{\Psi}_{\mathrm{SB}}(k-2,b)\right)^{+} \bar{\tilde{\boldsymbol{C}}}_{\mathrm{AMB}}(k-2;k-1;j),$$

where $(\cdot)^{+}$ denotes the Moore-Penrose pseudo-inverse and $\boldsymbol{\Psi}_{\mathrm{SB}}(k-2,b) \in \mathbb{R}^{O \times D_{\mathrm{SB}}(k-1,b)}$ denotes the mode matrix with respect to the direction estimates in the set $\left\{\boldsymbol{\Omega}_{\mathrm{SB},d}(k-2,b) \middle| d \in \tilde{\mathcal{J}}_{\mathrm{DIR}}(k-2,b)\right\}$.

C.5.3.x.4 Prediction of directional sub-band signals for individual sub-band groups

The prediction of all directional sub-band signals related to each j-th sub-band of the b-th sub-band group, $b = 1,\dots,B$, which are contained in the matrix $\bar{\tilde{\boldsymbol{X}}}(k-2;k-1;j)$, is assumed to performed by a matrix multiplication of $\bar{\tilde{\boldsymbol{C}}}_{\mathrm{AMB}}(k-2;k-1;j)$ with the matrix $\mathbf{A}(k-2,b)$ with all potential prediction coefficients. In particular, the predicted version of $\bar{\tilde{\boldsymbol{X}}}(k-2;k-1;j)$, which is denoted by $\bar{\tilde{\boldsymbol{X}}}_{\mathrm{P}}(k-2;k-1;j)$, is assumed to be computed by

$$\bar{\tilde{\boldsymbol{X}}}_{\mathrm{P}}(k-2;k-1;j) = \mathbf{A}(k-2,b)\bar{\tilde{\boldsymbol{C}}}_{\mathrm{AMB}}(k-2;k-1;j) \quad \text{for} \quad j = \mathcal{L}(b),\dots,\mathcal{U}(b) \quad \text{and} \quad b = 1,\dots,B.$$

Note that per construction all rows of $\mathbf{A}(k-2,b)$ except for those with index $d \in \tilde{\mathcal{J}}_{\mathrm{DIR}}(k-2,b)$ are zero, meaning that obviously only the active directional sub-band signals are predicted. Further, all columns of $\mathbf{A}(k-2,b)$ except for those with index $n \in \mathcal{J}_{\mathrm{AMB,ACT}}(k-2)$ are also zero, meaning that for prediction only those HOA coefficient sequences are considered which are supposed to be transmitted and to be available for prediction at HOA decompression.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

At HOA decompression, the original sub-band HOA representation $\tilde{\boldsymbol{C}}_{\mathrm{AMB}}(k-1,j)$ is in general not available, but instead only a perceptually decoded version of it, which is used for the prediction of the directional sub-band signals. In the case the core coder to encode the individual transport signals employs spectral band replication (SBR), it does not make sense to exploit any phase relationships for the prediction from "replicated high frequency content" by using complex valued prediction coefficients, since the SBR cannot be assumed to preserve any phase relationships. Instead, it is more reasonable to use only real valued prediction coefficients for the frequency region affected by SBR.

For that reason, the prediction coefficients are assumed to be complex valued for sub-band groups b smaller than the sub-band group index j_{SBR} (specified in the HOAConfig()), and real valued for the remaining sub-band groups, i.e.

$$\mathbf{A}(k-2,b) \in \begin{cases} \mathbb{C}^{D_{\mathrm{SB}} \times O} & \text{for} \quad 1 \le j < j_{\mathrm{SBR}} \\ \mathbb{R}^{D_{\mathrm{SB}} \times O} & \text{for} \quad j_{\mathrm{SBR}} \le j \le B \end{cases}.$$

It is reasonable to set the sub-band group index j_{SBR} to be the highest one such that the frequency corresponding to the lowest sub-band in this group is below the SBR frequency.

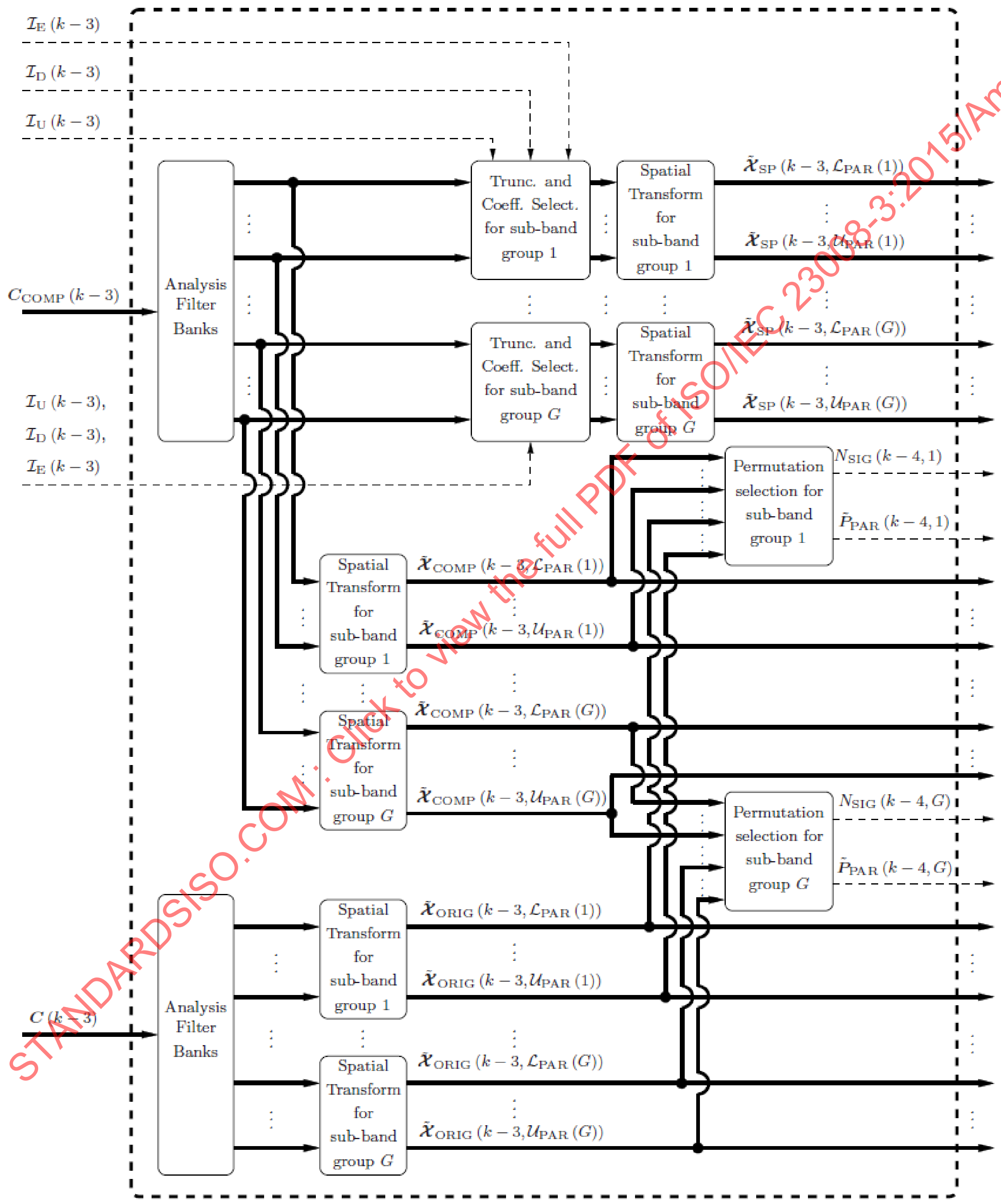
C.5.3.x Parametric Ambience Replication (PAR) Encoder

The main idea of Parametric Ambience Replication (PAR) is to complement the preliminary encoded HOA representation by potentially missing ambient components, which are parametrically replicated from itself. The replication is performed in the sub-band domain assuming F sub-bands that are assigned to G sub-band groups. The assignment is determined by the PAR related sub-band group configuration, which is specified in the HOAConfig() (see subclause **12.4.1.2.1 Upper and lower bounds of sub-band groups for Sub-band Directional Signals Synthesis**). It defines for each g-th sub-band group, $\mathrm{g} = 1, \ldots, \mathrm{G}$, a lower index bound $\mathcal{L}_{\mathrm{PAR}}(\mathrm{g})$ and an upper index bound $\mathcal{U}_{\mathrm{PAR}}(\mathrm{g})$ such that sub-bands with indices j between these bounds, i.e. with $\mathcal{L}_{\mathrm{PAR}}(\mathrm{g}) \le \mathrm{j} \le \mathcal{U}_{\mathrm{PAR}}(\mathrm{g})$, are assumed to belong to this sub-band group. The sub-band representation of the replicated ambient component for the j-th sub-band is assumed to be of order $\mathrm{N}_{\mathrm{PAR}}(\mathrm{g})$ depending on the corresponding g-th sub-band group. The orders $\mathrm{N}_{\mathrm{PAR}}(\mathrm{g})$ for each sub-band group $\mathrm{g} = 1, \ldots, \mathrm{G}$ are specified in the HOAConfig(). The mentioned sub-band representation of the replicated ambient component is hence represented and created by means of $\mathrm{O}_{\mathrm{PAR}}(\mathrm{g}) = (\mathrm{N}_{\mathrm{PAR}}(\mathrm{g}) + 1)^2$ virtual loudspeaker sub-band signals at directions $\boldsymbol{\Omega}_{\mathrm{d}}^{(\mathrm{N}_{\mathrm{PAR}}(\mathrm{g}))}$, $\mathrm{d} = 1, \ldots, \mathrm{O}_{\mathrm{PAR}}(\mathrm{g})$, defined in the tables in annexes F.2 and F.3. These up-mix sub-band signals are computed as a mixture of the sub-band signals created by de-correlation filters from the virtual loudspeaker sub-band signals representing the truncated sub-band HOA representation of $\boldsymbol{C}_{\mathrm{COMP}}(\mathrm{k}-3)$. Truncation in this context means for sub-bands j belonging to the g-th sub-band group the reduction of the order to $\mathrm{N}_{\mathrm{PAR}}(\mathrm{g})$ and the setting of all coefficient sequences to zero, whose indices are not contained in either of the sets $\mathcal{I}_{\mathrm{E}}(\mathrm{k}-3)$, $\mathcal{I}_{\mathrm{D}}(\mathrm{k}-3)$, or $\mathcal{I}_{\mathrm{U}}(\mathrm{k}-3)$. The number of de-correlated sub-band signals to be mixed for the creation of each up-mix sub-band signal is allowed to vary over time according to the values in **Table F.x Table for ParDecorrSigsSelectionTableIdx referring to NumOfDecorrSigsPerParSubbandTable and ParSelectedDecorrSigsIdxMatrixTable** in order to adapt to the diffuseness of the ambient HOA component to be replicated. This number, denoted by $\mathrm{N}_{\mathrm{SIG}}(\mathrm{k}-4,\mathrm{g})$, offers the possibility to control the amount of side information required to code the mixing matrices $\mathrm{M}_{\mathrm{PAR}}(\mathrm{k}-4,\mathrm{g})$ for the individual sub-band groups $\mathrm{g} = 1, \ldots, \mathrm{G}$. Further, for $\mathrm{N}_{\mathrm{SIG}}(\mathrm{k},\mathrm{g}) < \mathrm{O}_{\mathrm{PAR}}(\mathrm{g})$ the mixing uses de-correlated sub-band signals obtained from virtual loudspeaker signals $\boldsymbol{X}_{\mathrm{COMP}}(\mathrm{k},\mathrm{j})$ at directions in the neighborhood of the direction of the up-mix signal. This operation prevents that directional components of the truncated sub-band HOA representations of $\boldsymbol{C}_{\mathrm{COMP}}(\mathrm{k}-3)$ are undesirably spatially distributed over all directions for the replication of the ambient HOA component. An additional aspect is that for each number $\mathrm{N}_{\mathrm{SIG}}(\mathrm{k}-4,\mathrm{g})$ and each individual up-mix sub-band signal it is specified in **Table F.x Table for ParDecorrSigsSelectionTableIdx referring to NumOfDecorrSigsPerParSubbandTable and ParSelectedDecorrSigsIdxMatrixTable** , which de-correlated sub-band signals have to be mixed. In order to decrease the mutual correlation between each group of de-correlated sub-band signals to be mixed, the assignment of the virtual loudspeaker signals to the de-correlation filters is adapted to the choice of de-correlated sub-band signals. This assignment is expressed through the permutation matrices $\boldsymbol{P}_{\mathrm{PAR}}(\mathrm{k}-4,\mathrm{g})$ for the individual sub-band groups $\mathrm{g} = 1, \ldots, \mathrm{G}$. Note that for PAR overlap add processing is used to handle time varying parameters. Hence, each parameter indexed by the index $\mathrm{k}-4$ is assumed to be valid jointly for the frames $\mathrm{k}-4$ and $\mathrm{k}-3$. A possible realization of the PAR encoder may be decomposed into two parts, which are illustrated in Figure **AMD1.8** and Figure 27. The first part is concerned with the task how to determine the numbers $\mathrm{N}_{\mathrm{SIG}}(\mathrm{k}-4,\mathrm{g})$ of de-correlated sub-band signals

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

to be mixed for the creation of each up-mix sub-band signal related to the g-th sub-band group, $g = 1, \ldots, G$. These numbers unambiguously specify the permutation matrices $\boldsymbol{P}_{\mathrm{PAR}}(k-4, g)$, $g = 1, \ldots, G$, through Table F.x **Table for ParDecorrSigsSelectionTableIdx referring to ParPermIdxVectorTable**. The second part of the PAR encoder deals with problem of computing the mixing matrices $\boldsymbol{M}_{\mathrm{PAR}}(k-4, g)$, $g = 1, \ldots, G$. Both parts will be described in more detail in the following.

C.5.3.x.1 Part 1 of PAR Encoder



$\mathcal{I}_{\mathrm{E}}(k-3)$
$\mathcal{I}_{\mathrm{D}}(k-3)$
$\mathcal{I}_{\mathrm{U}}(k-3)$
$C_{\mathrm{COMP}}(k-3)$
Analysis Filter Banks
Trunc. and Coeff. Select. for sub-band group 1
Trunc. and Coeff. Select. for sub-band group G
Spatial Transform for sub-band group 1
Spatial Transform for sub-band group G
$\tilde{\mathcal{X}}_{\mathrm{SP}}(k-3, \mathcal{L}_{\mathrm{PAR}}(1))$
$\tilde{\mathcal{X}}_{\mathrm{SP}}(k-3, \mathcal{U}_{\mathrm{PAR}}(1))$
$\tilde{\mathcal{X}}_{\mathrm{SP}}(k-3, \mathcal{L}_{\mathrm{PAR}}(G))$
$\tilde{\mathcal{X}}_{\mathrm{SP}}(k-3, \mathcal{U}_{\mathrm{PAR}}(G))$
$\mathcal{I}_{\mathrm{U}}(k-3)$,
$\mathcal{I}_{\mathrm{D}}(k-3)$,
$\mathcal{I}_{\mathrm{E}}(k-3)$
Permutation selection for sub-band group 1
$N_{\mathrm{SIG}}(k-4, 1)$
$\bar{P}_{\mathrm{PAR}}(k-4, 1)$
$\tilde{\mathcal{X}}_{\mathrm{COMP}}(k-3, \mathcal{L}_{\mathrm{PAR}}(1))$
$\tilde{\mathcal{X}}_{\mathrm{COMP}}(k-3, \mathcal{U}_{\mathrm{PAR}}(1))$
$\tilde{\mathcal{X}}_{\mathrm{COMP}}(k-3, \mathcal{L}_{\mathrm{PAR}}(G))$
$\tilde{\mathcal{X}}_{\mathrm{COMP}}(k-3, \mathcal{U}_{\mathrm{PAR}}(G))$
Permutation selection for sub-band group G
$N_{\mathrm{SIG}}(k-4, G)$
$\bar{P}_{\mathrm{PAR}}(k-4, G)$
$\tilde{\mathcal{X}}_{\mathrm{ORIG}}(k-3, \mathcal{L}_{\mathrm{PAR}}(1))$
$\tilde{\mathcal{X}}_{\mathrm{ORIG}}(k-3, \mathcal{U}_{\mathrm{PAR}}(1))$
$C(k-3)$
$\tilde{\mathcal{X}}_{\mathrm{ORIG}}(k-3, \mathcal{L}_{\mathrm{PAR}}(G))$
$\tilde{\mathcal{X}}_{\mathrm{ORIG}}(k-3, \mathcal{U}_{\mathrm{PAR}}(G))$

Figure AMD1.8 — Part 1 of PAR Encoder

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

In the first part of the PAR encoder, as illustrated Figure **AMD1.8**, both, the frame of the reconstructed HOA representation $\boldsymbol{C}_{\mathrm{COMP}}(\mathrm{k}-3)$ and the frame of the delayed original HOA representation $\boldsymbol{C}(\mathrm{k}-3)$, are first decomposed into frames of individual sub-band HOA representations for each of the F assumed sub-bands by the application of the Analysis Filter Banks, as described in subclause **12.4.2.x.1 Analysis Filter Banks**. The resulting individual sub-band HOA representations are then subjected to a Spatial Transform dependent on the corresponding sub-band group (see subclause **12.4.2.x.3 Spatial Transform** for a detailed description of the processing). The frames of all resulting virtual loud-speaker sub-band signals for the j-th sub-band $\mathrm{j}=1,\ldots,\mathrm{F}$, are denoted by $\boldsymbol{\mathcal{X}}_{\mathrm{COMP}}(\mathrm{k}-3,\mathrm{j})$ and $\boldsymbol{\mathcal{X}}_{\mathrm{ORIG}}(\mathrm{k}-3,\mathrm{j})$, respectively. In a next step, for each g-th sub-band group, $\mathrm{g}=1,\ldots,\mathrm{G}$, the number $\mathrm{N}_{\mathrm{SIG}}(\mathrm{k}-4,\mathrm{g})$ of de-correlated sub-band signals to be mixed for the creation of an up-mix sub-band signal is determined from a comparison between the extended frames

$$[\mathcal{X}_{\mathrm{COMP}}(\mathrm{k}-4,\mathrm{j}) \quad \mathcal{X}_{\mathrm{COMP}}(\mathrm{k}-3,\mathrm{j})],\quad \mathcal{L}_{\mathrm{PAR}}(\mathrm{g})\leq \mathrm{j}\leq \mathcal{U}_{\mathrm{PAR}}(\mathrm{g})$$

of the virtual loud-speaker sub-band signals of the preliminary reconstructed HOA representation and the extended frames

$$[\mathcal{X}_{\mathrm{ORIG}}(\mathrm{k}-4,\mathrm{j}) \quad \mathcal{X}_{\mathrm{ORIG}}(\mathrm{k}-3,\mathrm{j})],\quad \mathcal{L}_{\mathrm{PAR}}(\mathrm{g})\leq \mathrm{j}\leq \mathcal{U}_{\mathrm{PAR}}(\mathrm{g})$$

of the virtual loud-speaker sub-band signals of the original HOA representation. The idea is that the number $\mathrm{N}_{\mathrm{SIG}}(\mathrm{k}-4,\mathrm{g})$ may be chosen, on the one hand, in dependence on the diffuseness of the ambient HOA component to be replicated, and on the other hand, in dependence on the available data rate for the PAR side information. As a rule of thumb, the value of $\mathrm{N}_{\mathrm{SIG}}(\mathrm{k}-4,\mathrm{g})$ should be increased the more diffuseness is missing in the preliminary reconstructed HOA representation compared to the original HOA representation. On the contrary, if a low PAR side information data rate is desired this value should be kept low. An important constraint is that $\mathrm{N}_{\mathrm{SIG}}(\mathrm{k}-4,\mathrm{g})$ is allowed only to have values as that of NumOfDecorrSigsPerParSubbandTable given in Table **F.x Table for ParDecorrSigsSelectionTableIdx referring to NumOfDecorrSigsPerParSubbandTable and ParSelectedDecorrSigsIdxMatrixTable**. The permutation matrices $\boldsymbol{P}_{\mathrm{PAR}}(\mathrm{k}-4,\mathrm{g})$, $\mathrm{g}=1,\ldots,\mathrm{G}$, which define the assignment of the virtual loud-speaker sub-band signals to the de-correlation filters, are selected according to Table **F.x Table for ParDecorrSigsSelectionTableIdx referring to ParPermIdxVectorTable** dependent on the numbers $\mathrm{N}_{\mathrm{SIG}}(\mathrm{k}-4,\mathrm{g})$. The dependence expressed by Table **F.x Table for ParDecorrSigsSelectionTableIdx referring to ParPermIdxVectorTable** is chosen to minimize the mutual correlation between the signals to be mixed. As a pre-processing step for the subsequent computation of the mixing matrices in the second part of the PAR Encoder, the sub-band HOA representation of the reconstructed HOA component $\boldsymbol{C}_{\mathrm{COMP}}(\mathrm{k}-3)$ is subjected to a Truncation, a Coefficient Selection and a sub-band group dependent Spatial Transform (see subclauses **12.4.2.x.2 Truncation and Coefficient Selection** and **12.4.2.x.3 Spatial Transform** for a detailed description of the processing) to provide the frames of all virtual loud-speaker sub-band signals $\boldsymbol{\mathcal{X}}_{\mathrm{SP}}(\mathrm{k}-3,\mathrm{j})$ for each j-th sub-band, $\mathrm{j}=1,\ldots,\mathrm{F}$.

C.5.3.x.2 Part 2 of PAR Encoder

In the second part of the PAR Encoder, depicted in Figure **AMD1.9**, the virtual loud-speaker sub-band signals $\boldsymbol{\mathcal{X}}_{\mathrm{SP}}(\mathrm{k}-3,\mathrm{j})$, representing the reconstructed HOA component $\boldsymbol{C}_{\mathrm{COMP}}(\mathrm{k}-3)$ are subject to a de-correlation. In particular, the de-correlated sub-band signals $\boldsymbol{\mathcal{X}}_{\mathrm{DEC}}(\mathrm{k}-3,\mathrm{j})$ for the j-th sub-band belonging to the g-th sub-band group, i.e. for $\mathcal{L}_{\mathrm{PAR}}(\mathrm{g})\leq \mathrm{j}\leq \mathcal{U}_{\mathrm{PAR}}(\mathrm{g})$, are computed according to the description in subclause **12.4.2.x.4 Computation of de-correlated sub-band signals** using the permutation matrix $\boldsymbol{P}_{\mathrm{PAR}}(\mathrm{k}-4,\mathrm{g})$ for the assignment of the virtual loud-speaker sub-band signals $\boldsymbol{\mathcal{X}}_{\mathrm{SP}}(\mathrm{k}-3,\mathrm{j})$ to the de-correlation filters.

In a final step, the permutation matrix $\boldsymbol{M}_{\mathrm{PAR}}(\mathrm{k}-4,\mathrm{g})$ for each g-th sub-band group, $\mathrm{g}=1,\ldots,\mathrm{G}$, is computed such as to approximate the extended frames

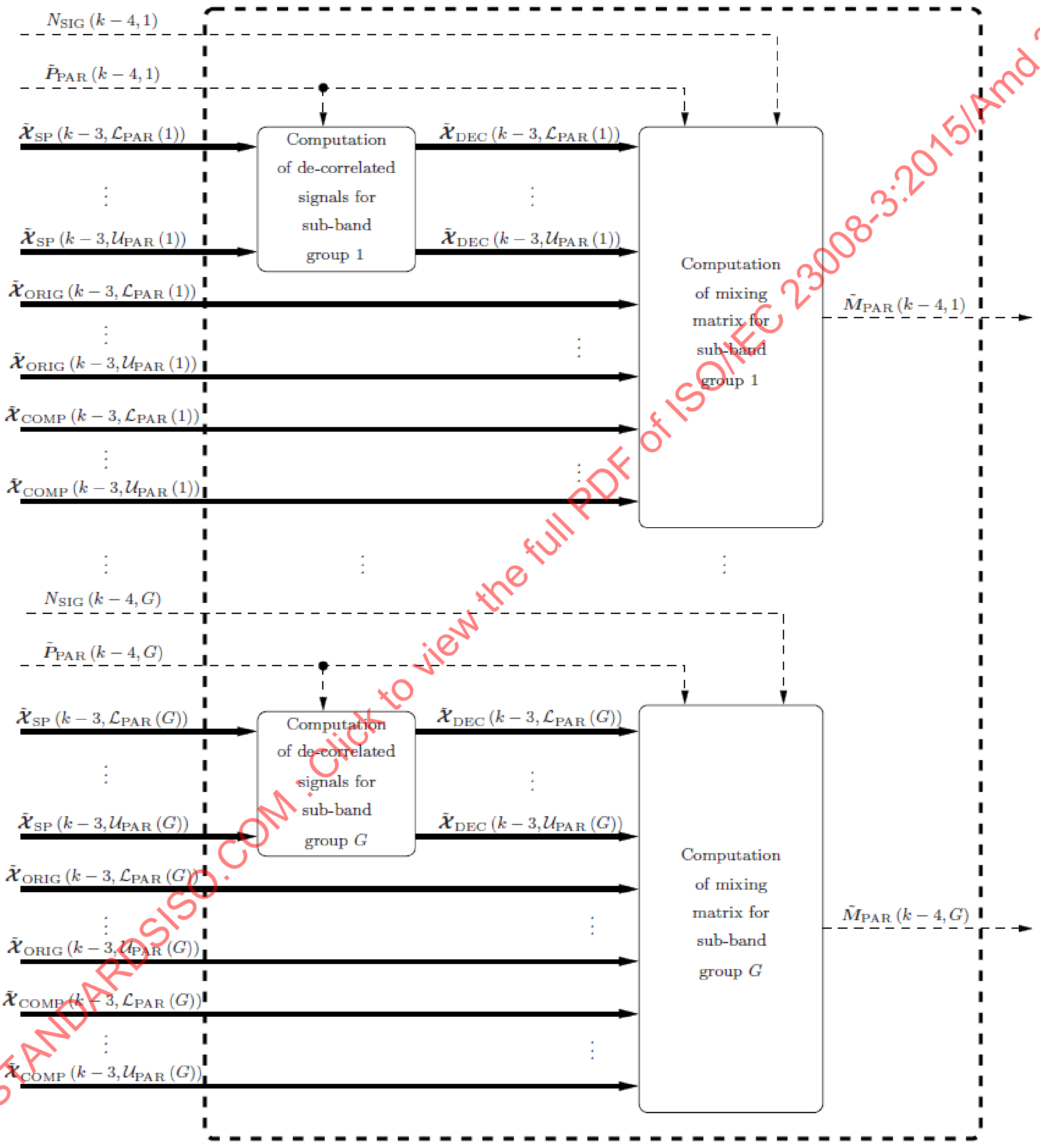
$$[\boldsymbol{\mathcal{X}}_{\mathrm{ORIG}}(\mathrm{k}-4,\mathrm{j}) \quad \boldsymbol{\mathcal{X}}_{\mathrm{ORIG}}(\mathrm{k}-3,\mathrm{j})]-[\boldsymbol{\mathcal{X}}_{\mathrm{COMP}}(\mathrm{k}-4,\mathrm{j}) \quad \boldsymbol{\mathcal{X}}_{\mathrm{COMP}}(\mathrm{k}-3,\mathrm{j})],\quad \mathcal{L}_{\mathrm{PAR}}(\mathrm{g})\leq \mathrm{j}\leq \mathcal{U}_{\mathrm{PAR}}(\mathrm{g})$$

of the virtual loud-speaker sub-band signals of the residual between the original and preliminary reconstructed HOA representation for all sub-bands j belonging to the g-th sub-band group by the following mixture

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

$$\boldsymbol{M}_{\mathrm{PAR}}(\mathrm{k}-4,\mathrm{g})\left(\boldsymbol{P}_{\mathrm{PAR}}(\mathrm{k}-4,\mathrm{g})\right)^{-1}[\boldsymbol{\mathcal{X}}_{\mathrm{DEC}}(\mathrm{k}-4,\mathrm{j})\quad \boldsymbol{\mathcal{X}}_{\mathrm{DEC}}(\mathrm{k}-3,\mathrm{j})],\quad \mathcal{L}_{\mathrm{PAR}}(\mathrm{g})\leq \mathrm{j}\leq \mathcal{U}_{\mathrm{PAR}}(\mathrm{g})$$

of the de-correlated signals created from the virtual loud-speaker sub-band signals of the preliminary reconstructed HOA representation.



$N_{\mathrm{SIG}}(k-4,1)$
$\tilde{P}_{\mathrm{PAR}}(k-4,1)$
$\tilde{\mathcal{X}}_{\mathrm{SP}}(k-3,\mathcal{L}_{\mathrm{PAR}}(1))$
$\tilde{\mathcal{X}}_{\mathrm{SP}}(k-3,\mathcal{U}_{\mathrm{PAR}}(1))$
Computation of de-correlated signals for sub-band group 1
$\tilde{\mathcal{X}}_{\mathrm{DEC}}(k-3,\mathcal{L}_{\mathrm{PAR}}(1))$
$\tilde{\mathcal{X}}_{\mathrm{DEC}}(k-3,\mathcal{U}_{\mathrm{PAR}}(1))$
$\tilde{\mathcal{X}}_{\mathrm{ORIG}}(k-3,\mathcal{L}_{\mathrm{PAR}}(1))$
$\tilde{\mathcal{X}}_{\mathrm{ORIG}}(k-3,\mathcal{U}_{\mathrm{PAR}}(1))$
$\tilde{\mathcal{X}}_{\mathrm{COMP}}(k-3,\mathcal{L}_{\mathrm{PAR}}(1))$
$\tilde{\mathcal{X}}_{\mathrm{COMP}}(k-3,\mathcal{U}_{\mathrm{PAR}}(1))$
Computation of mixing matrix for sub-band group 1
$\tilde{M}_{\mathrm{PAR}}(k-4,1)$
$N_{\mathrm{SIG}}(k-4,G)$
$\tilde{P}_{\mathrm{PAR}}(k-4,G)$
$\tilde{\mathcal{X}}_{\mathrm{SP}}(k-3,\mathcal{L}_{\mathrm{PAR}}(G))$
$\tilde{\mathcal{X}}_{\mathrm{SP}}(k-3,\mathcal{U}_{\mathrm{PAR}}(G))$
Computation of de-correlated signals for sub-band group G
$\tilde{\mathcal{X}}_{\mathrm{DEC}}(k-3,\mathcal{L}_{\mathrm{PAR}}(G))$
$\tilde{\mathcal{X}}_{\mathrm{DEC}}(k-3,\mathcal{U}_{\mathrm{PAR}}(G))$
$\tilde{\mathcal{X}}_{\mathrm{ORIG}}(k-3,\mathcal{L}_{\mathrm{PAR}}(G))$
$\tilde{\mathcal{X}}_{\mathrm{ORIG}}(k-3,\mathcal{U}_{\mathrm{PAR}}(G))$
$\tilde{\mathcal{X}}_{\mathrm{COMP}}(k-3,\mathcal{L}_{\mathrm{PAR}}(G))$
$\tilde{\mathcal{X}}_{\mathrm{COMP}}(k-3,\mathcal{U}_{\mathrm{PAR}}(G))$
Computation of mixing matrix for sub-band group G
$\tilde{M}_{\mathrm{PAR}}(k-4,G)$

Figure AMD1.9 — Part 2 of PAR Encoder

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In subclause C.5.4.1 Conversion to ActiveDirSigs[i] replace:

```
    if (DirIdx+1 == $\mathcal{I}_{\mathrm{DIR}}(k-2)$[NoOfActDirs]){
        **ActiveDirSigs**[DirIdx] = 1;
        NoOfActDirs++;
    }
```

With:

```
    if (DirIdx+1 == $\mathcal{I}_{\mathrm{DIR}}(k-4)$[NoOfActDirs]){
        **ActiveDirSigs**[DirIdx] = 1;
        NoOfActDirs++;
    }
```

In subclause C.5.4.2 Conversion to ActiveDirsIds[idx] replace:

```
for ($i = 0$; $i < I$; $i++$){
    if ($\mathrm{TYPE}_i(k-2) == DIR$){
        **ActiveDirIds**[NoOfActDirs] = $\boldsymbol{\Omega}^{(i)}_{\mathrm{QUANT}}(k-2)$;
        NoOfActDirs++;
    }
}
```

With:

```
for ($i = 0$; $i < I$; $i++$){
    if ($\mathrm{TYPE}_i(k-4) == DIR$){
        **ActiveDirIds**[NoOfActDirs] = $\boldsymbol{\Omega}^{(i)}_{\mathrm{QUANT}}(k-4)$;
        NoOfActDirs++;
    }
}
```

In subclause C.5.4.3 Conversion of prediction parameters $\boldsymbol{\zeta}(k-2)$ (replacement of all (k-2) with (k-4)) replace completec subclause with:

C.5.4.3 Conversion of prediction parameters $\boldsymbol{\zeta}(\mathbf{k}-4)$

The prediction parameters $\zeta(\mathrm{k}-4) = \{\mathrm{p}_{\mathrm{TYPE}}(\mathrm{k}-4), \mathrm{P}_{\mathrm{IND}}(\mathrm{k}-4), \mathrm{P}_{\mathrm{Q,F}}(\mathrm{k}-4)\}$ are converted to an intermediate coded representation, which corresponds to that used in the description of the HOAPredictionInfo payload in Table 127 — Syntax of HOAPredictionInfo, as follows:

First the flag $\mathrm{PSPredictionActive}$ is set which indicates whether a spatial prediction is performed at all:

```
$\mathrm{PSPredictionActive} = 0$;
if ($\mathcal{I}_{\mathrm{DIR}}(\mathrm{k}-4) \neq \emptyset$){
    for ($\mathrm{n} = 0$; $\mathrm{n} < 0$; $\mathrm{n}++$){
        for ($\mathrm{d} = 0$; $\mathrm{d} < \mathrm{D}_{\mathrm{PRED}}$; $\mathrm{d}++$){
       if ($\mathrm{p}_{\mathrm{IND,d+1,n+1}}(\mathrm{k}-4) \neq 0$){
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
                PSPredictionActive = 1;
            }
        }
    }
}
```

In the case that $\mathrm{PSPredictionActive} == 1$, the number NumActivePred of directions for which directional signals are predicted is computed as follows:

```
NumActivePred = 0;
for ($n = 0;\ n < O;\ n++$){
    if($p_{\mathrm{TYPE},n+1}(k-4) == 1$){
        NumActivePred++;
    }
}
```

Depending on the value of NumActivePred it is decided whether the indices n of the directions $\Omega_n^{(N)}$, for which directional signals are predicted, are either

— coded by a bit array **ActivePred** consisting of O elements, of which the n-th element indicates if the prediction for the direction $\boldsymbol{\Omega}_n^{(N)}$ is predicted or not. The bit array is computed according to

```
        for($n = 0;\ n < O;\ n++$){
            $\mathrm{ActivePred}[n] = p_{\mathrm{TYPE},n+1}(k-4)$;
        }
```

— coded by the coded number $\mathbf{NumActivePredIds} = \mathrm{NumActivePred} - 1$ and the array $\mathbf{PredIds}$ consisting of indices n of the directions $\boldsymbol{\Omega}_n^{(N)}$, for which directional signals are predicted. The array $\mathrm{PredIds}$ is computed according to

```
        $\mathrm{actIdx} = 0$;
        for($n = 0;\ n < O;\ n++$){
            if ($p_{\mathrm{TYPE},n+1}(k-4) == 1$){
                $\mathrm{PredIds}[\mathrm{actIdx}] = n$;
                $\mathrm{actIdx}++$;
            }
        }
```

The decision about which kind of coding is used is indicated by the value of $\mathrm{KindOfCodedPredIds}$, which is set to zero in the first case or to one in the second case. The decision is taken dependent on the value of $\mathrm{NumActivePred}$ according to

$$\mathrm{KindOfCodedPredIds} = \begin{cases} 1 & \text{if } \ \mathrm{NumActivePred} \le M_M \\ 0 & \text{else} \end{cases},$$

where M_M is the greatest integer satisfying

$$\lceil \log_2(M_M) \rceil + M_M \cdot \lceil \log_2(O) \rceil < O\,.$$

The elements $p_{\mathrm{IND},d,n}(k-4)$ of the matrix $\mathrm{P}_{\mathrm{IND}}(k-4)$, which are indices of predominant sound signals to be used for the prediction of signals at the directions $\Omega_n^{(N)}$, are coded according to

```
$\mathrm{TotalIdx} = 0$;
for($n = 0;\ n < O;\ n++$){
    if ($p_{\mathrm{TYPE},n+1}(k-4) == 1$){
        for ($d = 0;\ d < D_{\mathrm{PRED}};\ d++$){
            $\mathrm{PredDirSigIds}[\mathrm{TotalIdx}] = p_{\mathrm{IND},d+1,n+1}(k-4)$;
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
                $\mathrm{TotalIdx}++;$
            }
        }
}
```

The corresponding quantized prediction factors $p_{Q,F,d,n}(k-4)$, which are elements of the matrix $P_{Q,F}(k-4)$, are coded according to

```
$\mathrm{TotalIdx} = 0;$
for($n = 0;\ n < O;\ n++$){
    if ($p_{TYPE,n+1}(k-4) == 1$){
        for ($d = 0;\ d < D_{PRED};\ d++$){
            if ($p_{IND,d+1,n+1}(k-4) \neq 0$){
                $\mathrm{PredGains[TotalIdx]} = p_{Q,F,d+1,n+1}(k-4);$
                $\mathrm{TotalIdx}++;$
            }
        }
    }
}
```

In subclause C.5.4.4 Coding of ambient HOA coefficients side information replace:

```
for ($i = 0;\ i < I;\ i++$){
    if ($\mathrm{TYPE}_i(k-2) == AMB$ ){
        if ($\mathrm{COEFFIDX}_i(k-2) ==\ \mathrm{COEFFIDX}_i(k-3)$ )
        {
            AmbCoeffIdxChanged[$i$] = false;
        }
        else
        {
            AmbCoeffIdxChanged[$i$]  = true;
            CodedAmbCoeffIdx[NoOfCoeffs++]  = $\mathrm{COEFFIDX}_i(k-2)$;
        }
    }
}
```

With:

```
for ($i = 0;\ i < I;\ i++$){
    if ($\mathrm{TYPE}_i(k-4) == AMB$ ){
        if ($\mathrm{COEFFIDX}_i(k-4) ==\ \mathrm{COEFFIDX}_i(k-5)$ )
        {
            AmbCoeffIdxChanged[$i$] = false;
        }
        else
        {
            AmbCoeffIdxChanged[$i$]  = true;
            CodedAmbCoeffIdx[NoOfCoeffs++]  = $\mathrm{COEFFIDX}_i(k-4)$;
        }
    }
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In subclause C.5.4.5 Coding of channel type replace:

```
for ($i = 0$; $i < I$; $i++$){
    ChannelType[i] = $\mathrm{TYPE}_i(k-2)$;
}
```

With:

```
for ($i = 0$; $i < I$; $i++$){
    ChannelType[i] = $\mathrm{TYPE}_i(k-4)$;
}
```

In subclause C.5.4.6 Conversion to CodedGainCorrectionExp[n] replace:

```
for ($i = 0$; $i < I$; $i++$){
    CodeLength = 0;
    switch($e_i(k-2)$){
        case 0:
        {
            CodeLength = 1;
            break;
        }
        case -1:
        {
            CodeLength = 2;
            break;
        }
        default:
        {
            CodeLength = $e_i(k-2)$ + 2;
        }
    }

    if(IndependencyFlag){
        GainCorrPrevAmpExp[i] = ceil( log2( $g_{\mathrm{IGC},i}(k-3)$ ) );
    }
    for(l=0; l < (CodeLength - 1); l++){
        CodedGainCorrectionExp[l] = 0;
    }
    CodedGainCorrectionExp[l] = 1;
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

With:

```
for ($i = 0; i < I; i++$){
    CodeLength = 0;
    switch($e_i(k-4)$){
        case 0:
        {
            CodeLength = 1;
            break;
        }
        case -1:
        {
            CodeLength = 2;
            break;
        }
        default:
        {
            CodeLength = $e_i(k-4)$ + 2;
        }
    }

    if(IndependencyFlag){
        GainCorrPrevAmpExp[i] = ceil( log2( $g_{\mathrm{IGC},i}(k-5)$ ) );
    }
    for(l=0; l < (CodeLength – 1); l++){
        CodedGainCorrectionExp[l] = 0;
    }
    CodedGainCorrectionExp[l] = 1;
}
```

In subclause C.5.4.7 Conversion to GainCorrectionException[i] replace:

```
for($i$=0; $i$ < $I$; $i$++){
    GainCorrectionException[$i$] = $\beta_{i+1}(k-2)$;
}
```

With:

```
for($i$=0; $i$ < $I$; $i$++){
    GainCorrectionException[$i$] = $\beta_{i+1}(k-4)$;
}
```

In subclause C.5.4.8 Coding of VVector replace:

As previously described an assignment vector $\boldsymbol{v}_{\mathrm{A}}(k-2)$ containing the side information for each transport channel is provided. The quantized vector data for the transport channels containing vector-based predominant sound signals is assigned to the bit stream as follows:

```
for (i=0; i < J; i++){
    if ($\mathrm{TYPE}_i(k-2) == VEC$)
    {
        l=0;
        for (q = 0; q < VVecLength; q++){
                VecVal[i][q] = $\boldsymbol{v}_{\mathrm{QUANT}}^{(i)}(k-2)[q+O_{MIN}]$;
                l++;
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
        }
    }
}
```

With:

As previously described an assignment vector $\boldsymbol{v}_{\mathrm{A}}(k-4)$ containing the side information for each transport channel is provided. The quantized vector data for the transport channels containing vector-based predominant sound signals is assigned to the bit stream as follows:

```
for (i=0; i < J; i++){
    if $(\mathrm{TYPE}_i(k-4) == VEC)$
    {
        l=0;
        for (q = 0; q < VVecLength; q++){
                VecVal[i][q] = $\boldsymbol{v}^{(i)}_{\mathrm{QUANT}}(k-4)[q+O_{MIN}]$;
                l++;
        }
    }
}
```

Add the following subclauses after C.5.4.8 Coding of VVector:

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

C.5.4.x Coding of parameters for directional sub-band signals prediction

1. $\mathrm{NumOfGlobalPredDirs} = D_{\mathrm{JOINED}}(\mathrm{k}-4)$;
2. $\mathrm{UseDirectionalPrediction} = \begin{cases} 1 & \text{if } \exists \mathrm{b} \in \{1,\dots,\mathrm{B}\} \text{ such that } \boldsymbol{A}(\mathrm{k}-4,\mathrm{b}) \neq 0 \\ 0 & \text{else} \end{cases}$,
   where 0 denotes a zero matrix with the same dimensions as $\boldsymbol{A}(\mathrm{k}-4,\mathrm{b})$.
3. for (d =0; d < $\mathrm{NumOfGlobalPredDirs}$; d++)
```
{
    $\mathrm{GlobalPredDirsIds[d]} = \mathrm{q}$ such that $\boldsymbol{\Omega}_{\mathrm{JOINED,d+1}}(\mathrm{k}-4) = \boldsymbol{\Omega}_{\mathrm{q}}^{(\sqrt{\mathrm{Q}}-1)}$
}
```
4. for (b=0; b < NumOfPredSubbands; b++)
```
{
    $\mathrm{UseHuffmanCodingDiffMag[b]}$ is set to $1$ or $0$ depending on whether it is more efficient to use
    Huffman code or not for the differential coding for the magnitudes of all elements of the
prediction
    coefficient matrix $\mathbf{A}(k-4,b)$.

    $\mathrm{UseHuffmanCodingDiffAngle[b]}$ is set to $1$ or $0$ depending on whether it is more efficient to use
    Huffman code or not for the differential coding for the angles of all elements of the prediction
    coefficient matrix $\mathbf{A}(k-4,b)$.


    for ( d = 0; d < MaxNumOfPredDirsPerBand; d++)
    {
                $\mathrm{DirIsActive[b][d]} = \begin{cases} 1 & \text{if } \mathrm{d} \in \tilde{\mathcal{J}}_{\mathrm{DIR}}(\mathrm{k}-4,\mathrm{b}) \\ 0 & \text{else} \end{cases}$

        if $(\mathrm{DirIsActive[b][d]} == 1)$
        {
            $\mathrm{RelDirGridIdx[b][d]} = \mathrm{q}$ such that $\boldsymbol{\Omega}_{JOINED,q+1}(\mathrm{k}-4) = \boldsymbol{\Omega}_{\mathrm{SB,d+1}}(\mathrm{k}-4,\mathrm{b})$
        }
    }
```

© ISO/IEC 2017 – All rights reserved

}

5. Since the prediction coefficient matrix elements are coded differentially, before starting to encode the elements for a $(k-4)$-th independency frame it is necessary to initialize the quantized values to zero for the previous frame as follows:

```
if ($\mathrm{hoaIndependencyFlag}(k-4)$ )
{
    for (b=0; b < NumOfPredSubbands; b++){
        for ( d = 0; d < MaxNumOfPredDirsPerBand; d++) {
            for (n = 0; n < MaxNumOfCoeffsToBeTransmitted; n++)
            {
                $\mathrm{IntQuantMag}(k-5)[\mathrm{b}][\mathrm{d}][\mathrm{n}] = 0;$
                $\mathrm{IntQuantAngle}(k-5)[\mathrm{b}][\mathrm{d}][\mathrm{n}] = 0;$
            }
        }
    }
}
```

The actual encoding is assumed to be performed as follows:

```
for (b=0; b < NumOfPredSubbands; b++){
    for ( d = 0; d < MaxNumOfPredDirsPerBand; d++) {
        for (n = 0; n < MaxNumOfCoeffsToBeTransmitted; n++){
            if( $(n+1) \in \mathcal{J}_E(k) \cup \mathcal{J}_D(k) \cup \mathcal{J}_U(k)$ ){
                $\mathrm{FloatMag} = |\tilde{\boldsymbol{A}}(k-4,b)[d+1][n+1]|;$
                $\mathrm{IntQuantMag}(k-4)[\mathrm{b}][\mathrm{d}][\mathrm{n}] = 0;$
                $\mathrm{IntQuantAngle}(k-4)[\mathrm{b}][\mathrm{d}][\mathrm{n}] = 0;$
                if($\mathrm{FloatMag} > 1$) {
                    $\mathrm{IntQuantMag}(k-4)[\mathrm{b}][\mathrm{d}][\mathrm{n}] = \left\lfloor \frac{\ln(\mathrm{FloatMag})}{\ln(8/7)} + 8.5 \right\rfloor$
                }
                else{
                    $\mathrm{IntQuantMag}(k-4)[\mathrm{b}][\mathrm{d}][\mathrm{n}] = \lfloor \mathrm{FloatMag} \cdot 8 + 0.5 \rfloor;$
                }

                $\mathrm{FloatAngle} = \arg(\tilde{\boldsymbol{A}}(k-4,b)[d+1][n+1]);$
                if($\mathrm{FloatAngle} < -7 \cdot \frac{\pi}{8} - \frac{\pi}{16}$){
                    $\mathrm{IntQuantAngle}(k-4)[\mathrm{b}][\mathrm{d}][\mathrm{n}] = 8;$
                }
                else{
                    $\mathrm{IntQuantAngle}(k-4)[\mathrm{b}][\mathrm{d}][\mathrm{n}] = \left\lfloor \mathrm{FloatAngle} \cdot \frac{8}{\pi} + 0.5 \right\rfloor;$
                }
                $\mathrm{DecodedMagDiff}[\mathrm{b}][\mathrm{d}][\mathrm{n}] = \mathrm{IntQuantMag}(k-4)[\mathrm{b}][\mathrm{d}][\mathrm{n}]$
                                        $-\mathrm{IntQuantMag}(k-5)[\mathrm{b}][\mathrm{d}][\mathrm{n}];$
                $\mathrm{DecodedAngleDiff}[\mathrm{b}][\mathrm{d}][\mathrm{n}] = 0;$
                // If the magnitude after quantization is equal to zero,
                // the transmitted difference in quantized angle differences is ignored.
                // Hence, in that case it is reasonable to code the difference
                //as efficient as possible by setting it to zero.
                if($\mathrm{IntQuantMag}[b][d][n] != 0$){
                    $\mathrm{DecodedAngleDiff}[b][d][n] = \mathrm{IntQuantAngle}(k-4)[\mathrm{b}][\mathrm{d}][\mathrm{n}]$
                                        $-\mathrm{IntQuantAngle}(k-5)[\mathrm{b}][\mathrm{d}][\mathrm{n}];$
                    // Constrain the angle difference to lie in interval ]-pi,pi].
                    if ($\mathrm{DecodedAngleDiff}[\mathrm{b}][\mathrm{d}][\mathrm{n}] < -7$){
                        $\mathrm{DecodedAngleDiff}[\mathrm{b}][\mathrm{d}][\mathrm{n}] += 16;$
                    }
                    else{
                        if ($\mathrm{DecodedAngleDiff}[b][d][n] > 8$){
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
                        $\text{DecodedAngleDiff}[b][d][n] -= 16;$
                    }
                }
            }
            else{
                $\text{IntQuantAngle}(k-4)[b][d][n] = 0;$
            }
        }
        else{
            $\text{DecodedMagDiff}[b][d][n] = 0;$
            $\text{DecodedAngleDiff}[b][d][n] = 0;$
            $\text{IntQuantMag}(k-4)[b][d][n];$
            $\text{IntQuantAngle}(k-4)[b][d][n] = 0;$
        }
    }
  }
}
```

C.5.4.x Coding of parameters for Parametric Ambience Replication (PAR)

1. $\underline{\text{UsePar}} = \begin{cases} 1 & \text{if } \exists\ g \in \{1,\dots,G\} \text{ such that } \boldsymbol{M}_{\text{PAR}}(k-4,g) \neq 0 \\ 0 & \text{else} \end{cases}$

2. for (g =0; g < NumOfParSubbands; g++){

```
    $\text{KeepPreviousParMatrixFlag}[g] = \begin{cases} 1 & \text{if } \boldsymbol{M}_{\text{PAR}}(k-4,g) = \boldsymbol{M}_{\text{PAR}}(k-5,g) \\ 0 & \text{else} \end{cases};$

    Set $\text{ParDecorrSigsSelectionTableIdx}[g]$ according to Table 26 depending on value of
    $\text{N}_{\text{SIG}}(k-4,g) = \text{NumOfDecorrSigsPerParSubbandTable}$

    $\text{UseReducedNoOfUpmixSigs}[g] = \begin{cases} 1 & \exists \text{ any zero row in } \boldsymbol{M}_{\text{PAR}}(k-4,g) \\ 0 & \text{else} \end{cases};$

    if ($\text{UseReducedNoOfUpmixSigs}[g]{=}{=}1$){
        for ($n{=}0; n < \text{MaxNumOfDecoSigs}[g]; n{+}{+}$){
            $\text{UseParUpmixSig}[g][n] = \begin{cases} 0 & (n+1)-\text{th row in } \boldsymbol{M}_{\text{PAR}}(k-4,g) \text{ is zero} \\ 1 & \text{else} \end{cases};$
        }
    }

    $\text{UseParHuffmanCodingDiffAbs}[g] =$
        $\begin{cases} 1 & \text{for Huffman coding the differences of magnitudes of } \boldsymbol{M}_{\text{PAR}}(k-4,g) \text{ and } \boldsymbol{M}_{\text{PAR}}(k-5,g) \\ 0 & \text{for conventional coding} \end{cases};$

    if ($\text{UseRealCoeffsPerParSubband}[g]{=}{=}0$){
        $\text{UseParHuffmanCodingDiffAngle}[g] =$
            $\begin{cases} 1 & \text{for Huffman coding the differences of angles of } \boldsymbol{M}_{\text{PAR}}(k-4,g) \text{ and } \boldsymbol{M}_{\text{PAR}}(k-5,g) \\ 0 & \text{for conventional coding} \end{cases};$
    }
}
```

3. Since the elements of the mixing matrices $\boldsymbol{M}_{\text{PAR}}(k-4,g)$, $g = 1,\dots,G$, are coded differentially, before starting to encode the elements for a $(k-4)$-th independency frame, which means that the hoaIndependencyFlag in the HOAFrame() payload is set to one, it is necessary to initialize the quantized values to zero for the previous frame as follows:

```
for ($g$ =0; $g$ < NumOfParSubbands; $g$ ++){
    for (d=0; d < MaxNumOfDecoSigs($g$); d++){
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
            for (n=0; n < MaxNumOfDecoSigs($g$); n++){
                $\mathrm{IntQuantMagPAR}(k-5)[g][\mathrm{d}][\mathrm{n}] = 0$;
                $\mathrm{IntQuantAnglePAR}(\mathrm{k}-5)[g][\mathrm{d}][\mathrm{n}] = 0$;
            }
        }
}
```

The actual encoding is assumed to be performed as follows:

```
for ($g$ =0; $g$ < NumOfParSubbands; $g$ ++){
    for (d=0; d < MaxNumOfDecoSigs($g$); d++){
        for (n=0; n < MaxNumOfDecoSigs($g$); n++){
            $\mathrm{FloatMagPAR} = |\boldsymbol{M}_{\mathrm{PAR}}(\mathrm{k}-4,\mathrm{g})[\mathrm{d}][\mathrm{n}]|$;
            $\mathrm{IntQuantMagPAR}(k-4)[g][\mathrm{d}][\mathrm{n}] = 0$;
            $\mathrm{IntQuantAnglePAR}(k-4)[g][\mathrm{d}][\mathrm{n}] = 0$;
            if(FloatMagPAR > 1)
            {
                $\mathrm{IntQuantMagPAR}(k-4)[g][\mathrm{d}][\mathrm{n}] = \left\lfloor \frac{\ln(\mathrm{FloatMagPAR})}{\ln(8/7)} + 8.5 \right\rfloor$
            }
            else
            {
                $\mathrm{IntQuantMagPAR}(k-4)[g][\mathrm{d}][\mathrm{n}] = \lfloor \mathrm{FloatMagPAR} \cdot 8 + 0.5 \rfloor$;
            }

            $\mathrm{FloatAnglePAR} = \arg(\boldsymbol{M}_{\mathrm{PAR}}(\mathrm{k}-4,\mathrm{g})[\mathrm{d}][\mathrm{n}])$;
            if(FloatAnglePAR < $-7 \cdot \frac{\pi}{8} - \frac{\pi}{16}$){
                $\mathrm{IntQuantAnglePAR}(k-4)[g][\mathrm{d}][\mathrm{n}] = 8$;
            }
            else{
                $\mathrm{IntQuantAnglePAR}(k-4)[g][\mathrm{d}][\mathrm{n}] = \left\lfloor \mathrm{FloatAnglePAR} \cdot \frac{8}{\pi} + 0.5 \right\rfloor$;
            }
            $\mathrm{DecodedParMagDiff}[g][\mathrm{d}][\mathrm{n}] = \mathrm{IntQuantMagPAR}(\mathrm{k}-4)[g][\mathrm{d}][\mathrm{n}]$
                                        $-\mathrm{IntQuantMagPAR}(\mathrm{k}-5)[g][\mathrm{d}][\mathrm{n}]$;
            $\mathrm{DecodedParAngleDiff}[g][\mathrm{d}][\mathrm{n}] = 0$;
            // If the magnitude after quantization is equal to zero,
            // the transmitted difference in quantized angle differences is ignored.
            // Hence, in that case it is reasonable to code the difference
            //as efficient as possible by setting it to zero.
            if($\mathrm{IntQuantMagPAR}[b][d][n] \,!= 0$){
                $\mathrm{DecodedParAngleDiff}[b][d][n] = \mathrm{IntQuantAnglePAR}(k-4)[g][\mathrm{d}][\mathrm{n}]$
                                        $-\mathrm{IntQuantAnglePAR}(k-5)[g][\mathrm{d}][\mathrm{n}]$;
                // Constrain the angle difference to lie in interval ]-pi,pi].
                if ($\mathrm{DecodedParAngleDiff}[g][\mathrm{d}][\mathrm{n}] < -7$){
                    $\mathrm{DecodedParAngleDiff}[g][\mathrm{d}][\mathrm{n}] \mathrel{+}= 16$;
                }
                else{
                    if ($\mathrm{DecodedParAngleDiff}[g][d][n] > 8$){
                        $\mathrm{DecodedParAngleDiff}[g][d][n] \mathrel{-}= 16$;
                }
            }
            else
            {
                $\mathrm{IntQuantAnglePAR}(\mathrm{k}-4)[g][\mathrm{d}][\mathrm{n}] = 0$;
            }
        }
    }
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Add the following section after C.5 HOA Encoder:

C.x MPEG Surround Encoder Tool

The following building blocks are specified for very low bitrate coding of channel content (see Figure **AMD1.10**, Figure **AMD1.11** and Figure **AMD1.12**):

- Pre-rendering/mixing: A pre-rendering/mixing stage is used on the encoder side to convert a channel and object input scene into a channel scene before encoding.
- Format conversion: Format conversion can be applied on the encoder side to lower the number of channels to achieve good rate / distortion results for a given bit rate.
- MPEG Surround encoding: MPEG Surround can be applied with the following tree configurations: 5-2-5, 7-2-7 or 9-2-9 based on 7-2-7 with an arbitrary tree extension.
- 3D Audio core encoder: Either the format converted channels or the MPS stereo downmix is coded with a 3D Audio core encoder. The MPS side information is multiplexed into an mpegh3daExtElement.
- Phase 1 SAOC 3D encoder can be applied for encoding object signals.

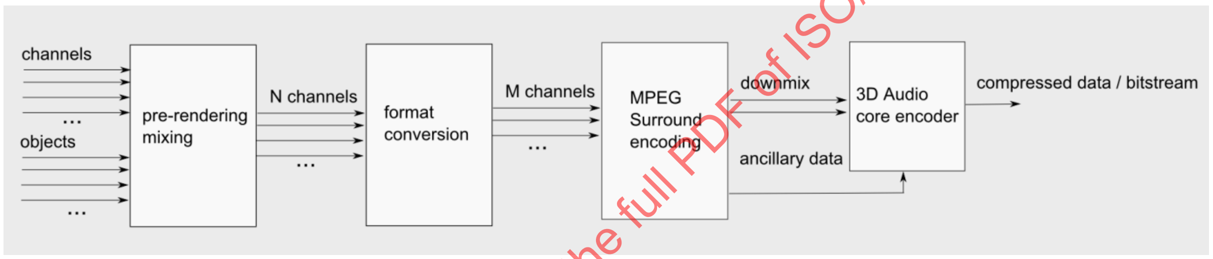


Figure AMD1.10 — Block Diagram of a 3DA Phase 2 encoder with an MPEG Surround encoder

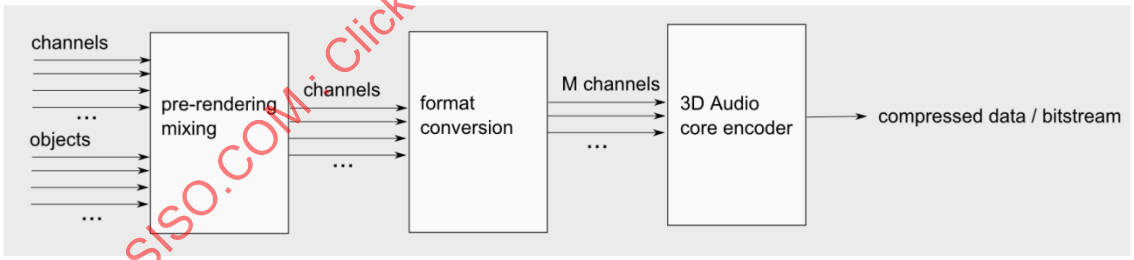


Figure AMD1.11 — Block Diagram of a 3DA Phase 2 encoder with reduced number of channels

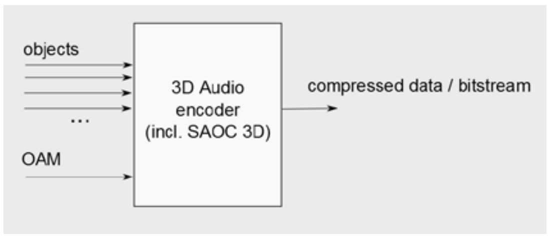


Figure AMD1.12 — Block Diagram of a 3DA Phase 2 encoder with reduced number of channels

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

In Annex F.1.5 Definition of the mode matrix replace:

$$\mathbf{S}_q^{(N_1)} := \left[S_0^0\left(\boldsymbol{\Omega}_q^{(N_2)}\right) \quad S_{-1}^{-1}\left(\boldsymbol{\Omega}_q^{(N_2)}\right) \quad S_{-1}^{0}\left(\boldsymbol{\Omega}_q^{(N_2)}\right) \quad S_{-1}^{1}\left(\boldsymbol{\Omega}_q^{(N_2)}\right) \quad S_{-2}^{-2}\left(\boldsymbol{\Omega}_q^{(N_2)}\right) \quad S_{-1}^{-2}\left(\boldsymbol{\Omega}_q^{(N_2)}\right) \quad \ldots \quad S_{N_1}^{N_1}\left(\boldsymbol{\Omega}_q^{(N_2)}\right) \right]^T \in \mathbb{R}^{O_1},$$

where $\mathrm{O}_1 = (\mathrm{N}_1 + 1)^2$.

With:

$$\mathbf{S}_q^{(N_1)} := \left[S_0^0\left(\boldsymbol{\Omega}_q^{(N_2)}\right) \quad S_{-1}^{-1}\left(\boldsymbol{\Omega}_q^{(N_2)}\right) \quad S_{-1}^{0}\left(\boldsymbol{\Omega}_q^{(N_2)}\right) \quad S_{-1}^{1}\left(\boldsymbol{\Omega}_q^{(N_2)}\right) \quad S_{-2}^{-2}\left(\boldsymbol{\Omega}_q^{(N_2)}\right) \quad S_{-1}^{-2}\left(\boldsymbol{\Omega}_q^{(N_2)}\right) \quad \ldots \quad S_{N_1}^{N_1}\left(\boldsymbol{\Omega}_q^{(N_2)}\right) \right]^T \in \mathbb{R}^{O_1}$$

denoting the mode vector of order N_1 with respect to the directions $\boldsymbol{\Omega}_q^{(N_2)}$,
where $\mathrm{O}_1 = (\mathrm{N}_1 + 1)^2$.
Correct table numbering mismatch for annex F.23 (correct references to the tables in the standard text accordingly!)

Replace Annex F.23 HOA Tables for Dynamic Range Control with the following clauses:

F.23 Spherical grid for DRC DSHT for order N=1

Inclination θ in rad,	Azimuth ϕ in rad,	q
0.33983655	3.14159265	3.14159271
1.57079667	0.00000000	3.14159267
2.06167886	1.95839324	3.14159262
2.06167892	-1.95839316	3.14159262

F.x Spherical grid for DRC DSHT for order N=2

Inclination θ in rad,	Azimuth ϕ in rad,	q
1.57079633	0.00000000	1.41002219
2.35131567	3.14159265	1.36874571
1.21127801	-1.18149779	1.36874584
1.21127606	1.18149755	1.36874598
1.31812905	-2.45289512	1.41002213
0.00975782	-0.00009218	1.41002214
1.31812792	2.45289621	1.41002230
2.41880319	1.19514740	1.41002223
2.41880555	-1.19514441	1.41002209

F.x Spherical grid for DRC DSHT for order N=3

Inclination θ in rad,	Azimuth ϕ in rad,	q
0.49220083	0.00000000	0.75567412
1.12054210	-0.87303924	0.75567398
2.52370429	-0.05517088	0.75567401
2.49233024	-2.15479457	0.87457076
1.57082248	0.00000000	0.87457075
2.02713647	1.01643753	0.75567388
1.61486095	-2.60674413	0.75567396
2.02713675	-1.01643766	0.75567398
1.08936018	2.89490077	0.75567412
1.18114721	0.89523032	0.75567399
0.65554353	1.89029902	0.75567382
1.60934762	1.91089719	0.87457082
2.68498672	2.02012831	0.75567392
1.46575084	-1.76455426	0.75567402
0.58248614	-2.22170415	0.87457060
2.00306837	2.81329239	0.75567389

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

F.x Spherical grid for DRC DSHT for order N=4

Inclination θ in rad,	Azimuth ϕ in rad,	q_i
1.57079633	0.00000000	0.52689274
2.39401407	0.00000000	0.48518011
1.14059283	-1.75618245	0.52688432
1.33721851	0.69215601	0.47027816
1.72512898	-1.33340585	0.48037442
1.17406779	-0.79850952	0.51130478
0.69042674	1.07623171	0.50662254
1.47478735	1.43953896	0.52158458
1.67073876	2.25235428	0.52835300
2.52745842	-1.33179653	0.52388165
1.81037110	3.05783641	0.49800736
1.91827560	-2.03351312	0.48516540
0.27992161	2.55302196	0.50663531
0.47981675	-1.18580204	0.50824199
2.37644317	2.52383590	0.45807408
0.98508365	2.03459671	0.47260252
2.18924206	1.58232601	0.49801422
1.49441825	-2.58932194	0.51745117
2.04428895	0.76615262	0.51744164
2.43923726	-2.63989327	0.52146074
1.10308418	2.88498471	0.52158484
0.78489181	-2.54224201	0.47027748
2.96802845	1.25258904	0.52145388
1.91816652	-0.63874484	0.48036020
0.80829458	-0.00991977	0.50824345

F.x Spherical grid for DRC DSHT for order N=5

Inclination θ in rad,	Azimuth ϕ in rad,	q_i
1.57079633	0.00000000	0.34493574
2.68749293	3.14159265	0.35131373
1.92461621	-1.22481468	0.35358151
1.95917092	3.06534485	0.36442231
2.18883411	0.08893301	0.36437350
0.35664531	-2.15475973	0.33953855
1.32915731	-1.05408340	0.35358417
2.21829206	2.45308518	0.33534647
1.00903070	2.31872053	0.34739607
0.99455136	-2.29370294	0.36437101
1.13601102	-0.46303195	0.33534542
0.41863640	0.63541391	0.35131934
1.78596913	-0.56826765	0.34739591
0.56658255	-0.66284593	0.36441956
2.25292410	0.89044754	0.36437098
2.67263757	-1.71236120	0.36442208
0.86753981	-1.50749854	0.34068122
1.38158330	1.72190554	0.35358401
0.98578154	0.23428465	0.35131950
1.45079827	-1.69748851	0.34739437
2.09223697	-1.85025366	0.33534659
2.62854417	1.70110685	0.34494256
1.44817433	-2.83400771	0.33953463
2.37827410	-0.72817212	0.34068529
0.82285875	1.51124182	0.33534531
0.40679748	2.38217051	0.34493552
0.84332549	-3.07860398	0.36437337
1.38947809	2.83246237	0.34068522
1.61795773	-2.27837285	0.34494274
2.17389505	-2.58540735	0.35131361
1.65172710	2.28105193	0.35358166
1.67862104	0.57097606	0.33953819
2.02514031	1.70739195	0.34739443
1.12965858	0.89802542	0.36442004
2.82979093	0.17840931	0.33953488
1.67550339	1.18664952	0.34068114

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

F.x Spherical grid for DRC DSHT for order N=6

Inclination θ in rad,	Azimuth ϕ in rad,	q
1.57079633	0.00000000	0.23821170
2.42144792	0.00000000	0.23821175
0.32919895	2.78993083	0.26169552
1.06225899	1.49243160	0.25534085
1.01526896	-2.16495206	0.25092628
1.10570423	-1.59180661	0.25099550
1.47319543	1.14258135	0.26160776
2.15414541	1.88359269	0.24442720
0.20805372	-0.52863458	0.25487678
0.50141101	-2.11057110	0.25619096
1.98041218	0.28912378	0.26288225
0.83752075	-2.81667891	0.25837996
2.44130228	0.81495962	0.26772416
1.21539727	-1.00788022	0.25534092
2.62944184	-1.58354086	0.26437874
1.86884674	-2.40686906	0.25619091
0.68705554	-1.20612227	0.25576026
1.52325470	-1.98940871	0.26169551
2.39097364	-2.37336381	0.25576025
0.98667678	0.86446728	0.26014219
2.27078506	-3.06771779	0.25099551
2.33605400	2.51674567	0.26455002
1.29371004	2.03656562	0.25576032
0.86334494	2.77720222	0.25092620
1.94118355	-0.37820559	0.26772409
2.10323413	-1.28283816	0.24442725
1.87416330	0.80785741	0.23821179
1.63423157	1.65277986	0.26437876
2.06477636	1.31341296	0.25595469
0.82305807	-0.47771423	0.26437883
2.04154780	-1.85106655	0.25487677
0.61285067	0.33640173	0.24442716
1.08029340	0.10986230	0.25595472
1.60164764	-1.43535015	0.26455000
2.66513701	1.69643796	0.26014228
1.35887781	-2.58083733	0.25838000
1.78658555	2.25563014	0.25487674
1.83333508	2.80487382	0.26169549
0.78406009	2.08860099	0.25099560
2.94031615	-0.07888534	0.26160780
1.34658213	2.57400947	0.25619094
1.73906669	-0.87744928	0.26014223
0.50210739	1.33550547	0.26455007
2.38040297	-0.75104092	0.25595462
1.41826790	0.54845193	0.26772418
1.77904107	-2.93136138	0.25092628
1.35746628	-0.47759398	0.26160765
1.31545731	3.12752832	0.25838016
2.81487011	-3.12843671	0.25534100

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

F.x Spherical grid for DRC DSHT for order N=7

Inclination θ in rad,	Azimuth ϕ in rad,	q_i
1.57079633	0.00000000	0.19495795
2.45610519	0.00000000	0.19495809
0.39336242	1.03016214	0.19791987
0.89422674	-2.33320867	0.19872783
0.43545329	-1.90611766	0.20164788
2.82600944	2.32040743	0.18728551
1.59930590	0.43907779	0.18583001
0.64745165	2.11280421	0.20273761
1.90012440	2.19672239	0.19118821
0.77544211	1.42837415	0.18728574
0.69899330	-0.36084163	0.18728569
2.04670638	-3.01527456	0.19927210
2.12677074	1.22510187	0.18728884
0.23447523	2.63866702	0.19927208
2.40003196	-2.63346362	0.19791989
1.45925921	-0.93421891	0.18728903
2.73580260	-0.95164110	0.19927196
1.48655587	-1.60786838	0.20474450
1.68102326	-2.09640999	0.19679660
2.36367468	2.19127430	0.19694872
2.32176930	-0.62030401	0.18583004
2.04546892	1.72866718	0.20474448
0.77250696	2.81267760	0.19495808
1.69576568	2.96849129	0.18583019
0.58175363	-2.82533899	0.18583001
2.87203994	0.51065147	0.20273763
1.39561603	0.84819515	0.20164795
1.88874012	-0.76208433	0.19872768
1.51281601	-2.91380498	0.19495791
1.08945861	2.38896622	0.19812700
1.83804298	-2.57622643	0.20273767
1.47075901	2.08646502	0.18728889
1.36153209	-2.47361065	0.19812703
1.15458107	0.35914488	0.19927210
2.23247953	-1.21160054	0.20164792
2.46336120	2.94460384	0.19682398
1.22791750	2.91409534	0.20129566
1.05015851	-2.89422460	0.20129569
2.12820204	-2.18619520	0.18728575
2.53612755	1.47084632	0.19679675
2.46546154	-1.76957871	0.19682388
1.55027992	2.54237851	0.19872776
2.84242076	-2.37565709	0.19791995
1.23439281	1.32812183	0.19694863
0.69294302	0.32555256	0.19791995
2.05614763	2.67444874	0.20164784
1.09765326	1.83278284	0.19679663
1.12592284	-0.14469268	0.20273758
1.96529200	-0.24906723	0.20129567
1.96552627	-1.69235565	0.19694871
1.76960407	-1.25179553	0.19118810
1.91136466	0.76951720	0.19872771
1.07406395	-0.69991724	0.19679662
0.69689253	-1.09648035	0.19694872
1.68400415	1.21349142	0.19118831
2.39207241	0.70375526	0.19812696
0.94043078	0.86239912	0.19682389
0.88424480	-1.70121947	0.19118805
1.24865844	-1.99210255	0.18728907
0.24866075	-0.52239150	0.19682390
1.16191600	-1.27749516	0.20474460
2.03676720	0.25756109	0.20129581
1.59547480	1.65945485	0.20474451
1.50944693	-0.45747372	0.19812716

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

F.x Spherical grid for DRC DSHT for order N=8

Inclination θ in rad,	Azimuth ϕ in rad,	q
1.57079633	0.00000000	0.16035506
2.37045281	3.14159265	0.15319651
1.16118114	-2.76809755	0.14493850
1.96701676	-2.93113550	0.15659032
0.44283230	2.79053078	0.16051177
2.27466442	-1.58633222	0.15823741
1.90257136	-1.34326395	0.15410190
1.22671270	-0.30245108	0.15555055
1.18223758	3.09331226	0.16269226
0.77757990	-3.00089450	0.15370974
1.71069627	1.52817508	0.15330871
2.10766509	-2.12432828	0.15659023
1.91267082	1.11619994	0.15721852
1.50641800	-1.38243964	0.15314283
1.90689598	0.26039435	0.15542290
1.87573690	-1.75821192	0.15263979
0.66839369	0.92054291	0.15198113
2.99900287	1.37574537	0.15918574
1.25503817	1.48780173	0.15554994
2.12588745	1.55120363	0.15777865
1.46867749	-1.78074999	0.15251277
1.42058775	0.80011364	0.14438414
1.39768944	-2.42566469	0.15882669
1.68448339	2.64994016	0.15410149
2.68318874	2.44896055	0.15624640
1.18069550	-2.06498432	0.15624611
0.95913015	-0.72316181	0.15330920
1.46298411	1.86944950	0.15860486
2.29917993	2.02407291	0.15772368
2.19757565	0.59478929	0.14875234
1.38002876	-0.71570078	0.15860447
1.66966479	-2.13711058	0.15319630
1.10433150	-1.56966543	0.15772351
1.51539913	1.17237827	0.15450396
1.67195210	-0.98489422	0.15912839
0.42176814	-1.52602006	0.15232202
1.06037388	1.88421174	0.14355622
0.80107740	1.50114356	0.16035524
2.52167695	-2.05535482	0.15604687
2.28748597	-2.58806117	0.15620875
2.58630438	1.56214454	0.15317153
2.72391006	-0.27193316	0.15370972
1.62712008	-0.39597375	0.14355623
2.34174815	0.09134393	0.16051150
1.06986925	0.59872741	0.16355869
1.92177200	-0.65082006	0.16077195
0.34213508	0.27645779	0.16434890
1.94986982	2.34291415	0.15314342
2.30898057	1.08877369	0.15232257
1.98986693	2.91045974	0.15264014
1.59231399	3.06757473	0.15823739
1.29864641	2.69395181	0.14576771
2.29317210	2.58349572	0.15251182
0.05793299	-2.59778561	0.14875351
1.99058472	-0.17696672	0.15658029
2.73125795	-2.83579628	0.15882635
2.15451150	-1.01207152	0.14576802
1.07129033	1.08091974	0.15708292
1.52711913	2.26168806	0.15912867
1.16236502	0.13916825	0.15708284
1.81256106	0.70838029	0.16434891
1.85530369	-2.50737902	0.15620805
0.74434685	0.30249140	0.14438364
1.49824071	0.40090739	0.15198070
1.56474110	-2.81849429	0.15604718
2.32808998	-0.50109067	0.15441122
1.87070391	1.92956677	0.15353048
2.88720571	-1.59178147	0.14493905
0.79687026	-1.22448828	0.15777882
0.49073383	-2.49121159	0.15636169
0.86597756	2.72407864	0.15441117
0.90380271	-2.41570722	0.15918617

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
0.69379152        2.15722882      0.15658036
0.77950388       -1.87939660      0.15317146
0.38197320        1.59711308      0.15542262
2.54311968       -1.11209930      0.16269195
1.21275328       -1.10763361      0.15353067
1.12072521        2.31432004      0.16077173
0.51475687       -0.64396568      0.15721788
2.62938152        0.64839067      0.15636234
0.82751400       -0.22838738      0.15450447
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

F.x Spherical grid for DRC DSHT for order N=9

Inclination θ in rad,	Azimuth ϕ in rad,	q
1.57079633	0.00000000	0.12828036
2.16373203	3.14159265	0.12214120
2.55778055	1.07800687	0.12411909
0.36041732	1.91355038	0.12214053
1.86313281	0.70422900	0.12980341
0.68734931	0.62766121	0.12595879
2.53087449	-3.07674104	0.12686568
1.93858183	-0.00399326	0.12591541
1.99735356	-1.78523667	0.12686523
2.54599177	-1.14448038	0.12980331
1.41727037	2.10029510	0.12827997
1.42975298	-2.03832959	0.12040441
0.49591158	-2.80876714	0.12828003
1.94504826	2.64716384	0.12980323
1.46050203	0.65963288	0.12411898
0.81994922	1.60055042	0.12980342
2.90039010	-2.83032233	0.12595875
1.17750420	-2.52395452	0.12980367
2.23201026	-1.41758015	0.12980324
1.83260522	3.00221101	0.12040296
0.27280823	0.57309173	0.12827984
1.82697936	-1.42107305	0.12411958
0.64885136	-0.63755903	0.12214100
2.92668990	-0.74715817	0.12411960
0.83608830	3.07605331	0.12595899
2.13647090	-0.37015260	0.12591690
1.17248048	-2.96343400	0.12411921
2.27278176	-0.78829766	0.12214053
2.33382027	0.02692246	0.12827987
1.44091653	-1.38509544	0.13113000
1.49774230	-2.75550338	0.12686541
1.64796000	2.38538463	0.12686567
0.86423080	-2.72191030	0.12040420
1.88340220	-0.70858136	0.12827970
1.84991189	-2.51360023	0.12591545
1.55433945	2.75646776	0.12411975
1.05621052	2.19619507	0.12040388
1.62064026	-1.72752451	0.12595883
1.13565641	-0.61914701	0.12980332
1.13683348	1.80774348	0.12214042
0.73983145	-2.25726397	0.12214044
0.29726235	-0.84075556	0.12591620
0.52655839	2.71448092	0.12686515
0.42942711	-1.90719617	0.12591540
0.91398906	1.08067861	0.12411953
2.68849385	2.50148936	0.12411959
2.36098369	-1.86914671	0.12214058
1.83657803	-2.89541550	0.12828043
1.27526530	1.01421481	0.12595902
2.35129951	2.23269302	0.12040288
1.30431251	0.32225872	0.12595868
0.90231487	2.60267985	0.12411967
2.30337398	2.71898541	0.12980326
1.39373392	-0.35543690	0.12214089
1.49751408	-3.13733860	0.12595843
2.01751030	-1.08299197	0.12040386
1.17568045	1.40534497	0.12686527
2.19236789	-2.70093341	0.12591627
2.56023704	-0.43012045	0.12686513
1.51097956	-2.38499767	0.12214025
1.28573634	2.50420996	0.12595853
0.84660824	-1.75673313	0.12686549
0.70805638	2.10420981	0.12980326
1.48574685	1.70727610	0.12591670
1.99776782	2.26037793	0.12214140
1.68420229	0.34882309	0.12686573
1.84850208	1.58929972	0.12591555
2.05336705	0.37361712	0.12214035
0.13006441	-2.96000913	0.12591661
1.76189826	1.97111953	0.12591598
2.71326544	-1.80095205	0.12040383
1.75573602	-0.35227870	0.12591593

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
 1.19314055       2.90104754     0.13112995
 1.63811123       0.98412082     0.12040459
 1.07280135       0.65657785     0.13113003
 0.91115869      -0.28257823     0.12980346
 1.29046828      -1.01963661     0.12411946
 2.37826460       0.55252527     0.12040438
 2.13871758       1.83860212     0.12828036
 2.24254365       1.40568224     0.12686556
 1.65398515      -1.06244312     0.12595861
 0.64259427      -1.30247588     0.12828045
 2.51222922      -2.44030027     0.12827971
 1.79529243      -2.10829994     0.12828002
 1.51732931      -0.70668218     0.12686586
 0.92550987      -0.95463482     0.12040314
 1.08652419      -2.13466658     0.12980337
 1.20141169      -0.04579864     0.12040323
 2.14184722      -2.24301998     0.12591668
 2.54887172       1.74602244     0.12595856
 1.05541584      -1.36282902     0.12595851
 0.91981071       0.23029246     0.12411927
 2.71843378       0.31100130     0.12595908
 1.53660179       1.34230146     0.12827978
 2.16591389       0.92821735     0.12980362
 0.56817609       0.00264963     0.12686571
 2.92170819       1.46142465     0.13112985
 1.91166616       1.21479777     0.12214024
 0.55577503       1.21928553     0.12040379
 1.22767373      -1.71900109     0.12411923
```

Add the following tables after Annex F.23 HOA Tables for Dynamic Range Control:

F.x Huffman Table for Decoding HuffmanMagDiffNoSbr

HuffmanWord	Codeword Length	HuffmanMagDiffNoSbr[HuffmanWord]
0101010	7	-8 (escape for run-length code)
0101110111	10	-7
010111001	9	-6
01011000	8	-5
0101000	7	-4
0101101	7	-3
01000	5	-2
00	2	-1
1	1	0
011	3	1
01001	5	2
0101111	7	3
0101011	7	4
01011001	8	5
010111010	9	6
010111000	9	7
0101110110	10	8
0101001	7	9 (escape for run-length code)

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

F.x Huffman Table for Decoding HuffmanMagDiffSbr

HuffmanWord	Codeword Length	HuffmanMagDiffSbr[HuffmanWord]
0100111100	10	-8 (escape for run-length code)
010011111111	12	-7
0100111101	10	-6
01001110	8	-5
0100001	7	-4
010001	6	-3
01010	5	-2
00	2	-1
1	1	0
011	3	1
01011	5	2
010010	6	3
0100110	7	4
0100000	7	5
0100111110	10	6
010011111101	12	7
010011111100	12	8
010011111110	12	9 (escape for run-length code)

F.x Huffman Table for Decoding DecTableAngleDiff

HuffmanWord	Codeword Length	DecodedAngleDiff[HuffmanWord]
0111111	7	-7
0110111	7	-6
000111	6	-5
011000	6	-4
011110	6	-3
0000	4	-2
010	3	-1
1	1	0
001	3	1
01110	5	2
00010	5	3
011001	6	4
000110	6	5
0110110	7	6
0111110	7	7
011010	6	8

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

F.x Huffman Table for Decoding ParHuffmanMagDiffNoSbr

HuffmanWord	Codeword Length	ParHuffmanMagDiffNoSbr[HuffmanWord]
001110000	9	-8 (escape for run-length code)
00111001	8	-7
001001	6	-6
00101	5	-5
10100	5	-4
0100	4	-3
1011	4	-2
100	3	-1
11	2	0
011	3	1
000	3	2
0101	4	3
10101	5	4
00110	5	5
001111	6	6
001000	6	7
0011101	7	8
001110001	9	9 (escape for run-length code)

F.x Huffman Table for Decoding ParHuffmanMagDiffSbr

HuffmanWord	Codeword Length	ParHuffmanMagDiffSbr[HuffmanWord]
1000011000	10	-8 (escape for run-length code)
11111101	8	-7
100001101	9	-6
11111111	8	-5
100000	6	-4
11110	5	-3
1001	4	-2
110	3	-1
0	1	0
101	3	1
1110	4	2
10001	5	3
111110	6	4
1000010	7	5
10000111	8	6
11111100	8	7
11111110	8	8
1000011001	10	9 (escape for run-length code)

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

F.x Huffman Table for Decoding ParDecTableAngleDiff

HuffmanWord	Codeword Length	ParDecodedAngleDiff[HuffmanWord]
100111	6	-7
111001	6	-6
10000	5	-5
10010	5	-4
0011	4	-3
000	3	-2
110	3	-1
01	2	0
101	3	1
1111	4	2
0010	4	3
10001	5	4
111011	6	5
111010	6	6
100110	6	7
111000	6	8

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

F.x Table for ParDecorrSigsSelectionTableIdx referring to NumOfDecorrSigsPerParSubbandTable and ParSelectedDecorrSigsIdxMatrixTable

ParDecorrSigs SelectionTableIdx	NumOfDecorrSigs PerParSubbandTable	ParSelectedDecorr SigsIdxMatrixTable
0	1	{ {0}, {1}, {2}, {3}, {4}, {5}, {6}, {7}, {8} }
1	2	{ {0, 2}, {1, 4}, {2, 4}, {3, 5}, {1, 4}, {3, 5}, {1, 6}, {3, 7}, {1, 8} }
2	4	{ {0, 2, 3, 8}, {1, 4, 7, 8}, {0, 2, 4, 5}, {0, 3, 5, 7}, {1, 2, 4, 5}, {2, 3, 4, 5}, {1, 3, 4, 6}, {1, 3, 7, 8}, {0, 1, 2, 8} }
3	9	{ {0, 1, 2, 3, 4, 5, 6, 7, 8}, {0, 1, 2, 3, 4, 5, 6, 7, 8}, {0, 1, 2, 3, 4, 5, 6, 7, 8}, {0, 1, 2, 3, 4, 5, 6, 7, 8}, {0, 1, 2, 3, 4, 5, 6, 7, 8}, {0, 1, 2, 3, 4, 5, 6, 7, 8}, {0, 1, 2, 3, 4, 5, 6, 7, 8}, {0, 1, 2, 3, 4, 5, 6, 7, 8}, {0, 1, 2, 3, 4, 5, 6, 7, 8} }

F.x Table for ParDecorrSigsSelectionTableIdx referring to ParPermIdxVectorTable

ParDecorrSigs SelectionTableIdx	**ParPermIdxVectorTable**
0	{0, 1, 2, 3, 4, 5, 6, 7, 8}
1	{0, 1, 3, 7, 2, 5, 6, 8, 4}
2	{0, 1, 3, 7, 5, 4, 6, 2, 8}
3	{0, 1, 2, 3, 4, 5, 6, 7, 8}

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

F.x Table for ParDecorrSigsSelectionTableIdx referring to NumOfDecorrSigsPerFirstOrderParSubbandTable and ParFirstOrderSelectedDecorrSigsIdxMatrixTable

ParDecorrSigs SelectionTableIdx	NumOfDecorrSigs PerFirstOrderParSubbandTable	ParFirstOrderSelectedDecorr SigsIdxMatrixTable
0	1	{ {0}, {1}, {2}, {3}, {4}, }
1	2	{ {0, 2}, {1, 3}, {2, 3}, {2, 3} }
2	3	{ {0, 1, 2}, {0, 1, 3}, {0, 2, 3}, {1, 2, 3} }
3	4	{ {0, 1, 2, 3, 4}, {0, 1, 2, 3, 4}, {0, 1, 2, 3, 4}, {0, 1, 2, 3, 4} }

F.x Table for ParDecorrSigsSelectionTableIdx referring to ParFirstOrderPermIdxVectorTable

ParDecorrSigs SelectionTableIdx	**ParFirstOrderPermIdxVectorTable**
0	{0, 1, 2, 3}
1	{0, 1, 3, 2}
2	{0, 1, 2, 3}
3	{0, 1, 2, 3}

Add before F.12 Table of 256x8 weighting values, WeightValCdbk:

F.LP Table of loudspeaker directions

Index q	θ_q in deg	ϕ_q in deg
0	-	
1	90	30
2	90	-30
3	90	0
4	90	110
5	90	-110
6	90	22
7	90	-22
8	90	135
9	90	-135
10	90	180

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Index q	θ_q in deg	ϕ_q in deg
11	90	90
12	90	-90
13	90	60
14	90	-60
15	55	30
16	55	-30
17	55	0
18	55	135
19	55	-135
20	55	180
21	55	90
22	55	-90
23	0	0
24	105	45
25	105	-45
26	105	0
27	55	110
28	55	-110
29	55	45
30	55	-45
31	90	45
32	90	-45
33	90	150
34	90	-150

F.2D Table of 64 horizontal-only directions

Index q	θ_q in rad	ϕ_q in rad
0	-	
1	1.570796327	0.073140204
2	1.570796327	0.171314974
3	1.570796327	0.269489745
4	1.570796327	0.367664515
5	1.570796327	0.465839286
6	1.570796327	0.564014056
7	1.570796327	0.662188827
8	1.570796327	0.760363597
9	1.570796327	0.858538367
10	1.570796327	0.956713138
11	1.570796327	1.054887908
12	1.570796327	1.153062679
13	1.570796327	1.251237449
14	1.570796327	1.349412219
15	1.570796327	1.44758699
16	1.570796327	1.54576176
17	1.570796327	1.643936531
18	1.570796327	1.742111301
19	1.570796327	1.840286072
20	1.570796327	1.938460842
21	1.570796327	2.036635612
22	1.570796327	2.134810383
23	1.570796327	2.232985153
24	1.570796327	2.331159924
25	1.570796327	2.429334694
26	1.570796327	2.527509465
27	1.570796327	2.625684235
28	1.570796327	2.723859005
29	1.570796327	2.822033776
30	1.570796327	2.920208546
31	1.570796327	3.018383317
32	1.570796327	3.116558087
33	1.570796327	3.214732858
34	1.570796327	3.312907628
35	1.570796327	3.411082398
36	1.570796327	3.509257169
37	1.570796327	3.607431939
38	1.570796327	3.70560671
39	1.570796327	3.80378148

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Index q	θ_q in rad	ϕ_q in rad
40	1.570796327	3.901956251
41	1.570796327	4.000131021
42	1.570796327	4.098305791
43	1.570796327	4.196480562
44	1.570796327	4.294655332
45	1.570796327	4.392830103
46	1.570796327	4.491004873
47	1.570796327	4.589179644
48	1.570796327	4.687354414
49	1.570796327	4.785529184
50	1.570796327	4.883703955
51	1.570796327	4.981878725
52	1.570796327	5.080053496
53	1.570796327	5.178228266
54	1.570796327	5.276403036
55	1.570796327	5.374577807
56	1.570796327	5.472752577
57	1.570796327	5.570927348
58	1.570796327	5.669102118
59	1.570796327	5.767276889
60	1.570796327	5.865451659
61	1.570796327	5.963626429
62	1.570796327	6.0618012
63	1.570796327	6.15997597
64	1.570796327	6.258150741

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

5 Optimizations and Improvements for Low Bitrate Coding

The following bit stream syntax is based on ISO/IEC 23008-3:2015, clause 5.

In Table 11 – Syntax of mpegh3daCoreConfig() replace

tw_mdct;	**1**	**bslbf**
noiseFilling;	**1**	**bslbf**

with:

tw_mdct;	**1**	**bslbf**
fullbandLpd;	**1**	**bslbf**
noiseFilling;	**1**	**bslbf**

In Table 25 – Syntax of fd_channel_stream() replace:

Syntax	No. of bits	Mnemonic

fd_channel_stream(common_window, common_tw, tns_data_present, noiseFilling, indepFlag)
{		

with:

Table 1 — Syntax of fd_channel_stream()

Syntax	No. of bits	Mnemonic

fd_channel_stream(common_window, common_tw, tns_data_present, noiseFilling, fullbandLpd, indepFlag)
{		

and in the same table replace:

if (**fac_data_present**) {	**1**	**uimsbf**
fac_length = (window_sequence==EIGHT_SHORT_SEQUENCE) ? ccfl/16 : ccfl/8;		
fac_data(1, fac_length);		
}		

with:

if (**fac_data_present**) {	**1**	**uimsbf**
if (fullbandLpd) {		
fac_length = (window_sequence==EIGHT_SHORT_SEQUENCE) ? ccfl/32 : ccfl/16;		
} else {		
fac_length = (window_sequence==EIGHT_SHORT_SEQUENCE) ? ccfl/16 : ccfl/8;		
}		
fac_data(1, fac_length);		
}		

Also replace in “Table 32 – Syntax of mpegh3daCoreCoderData()“:

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
            lpd_channel_stream(indepFlag);
        }
        else {
            if ( (nrChannels == 1) || (core_mode[0] != core_mode[1]) ) {
                tns_data_present[ch];                                   1       uimsbf
            }
            fd_channel_stream(common_window, common_tw,
                tns_data_present[ch], noiseFilling, indepFlag);
```

with:

```
            lpd_channel_stream(noiseFilling, fullbandLpd, indepFlag);
        }
        else {
            if ((nrChannels == 1) || (core_mode[0] != core_mode[1])) {
                tns_data_present[ch];                                   1       uimsbf
            }
            fd_channel_stream(common_window, common_tw,
                tns_data_present[ch], noiseFilling, fullbandLpd, indepFlag);
```

Add Table AMD2.1 – Syntax of tbe_data() at the end of 5.2.3.2

Table AMD2.1 — Syntax of tbe_data()

Syntax	No. of bits	Mnemonic
tbe_data()		
{		
tbe_heMode;	**1**	**uimsbf**
idxFrameGain;	**5**	**uimsbf**
idxSubGains;	**5**	**uimsbf**
lsf_idx[0];	**7**	**uimsbf**
lsf_idx[1];	**7**	**uimsbf**
if (tbe_heMode == 0) {		
tbe_hrConfig;	**1**	**uimsbf**
tbe_nlConfig;	**1**	**uimsbf**
idxMixConfig;	**2**	**uimsbf**
if (tbe_hrConfig == 1) {		
idxShbFrGain;	**6**	**uimsbf**
idxResSubGains;	**5**	**uimsbf**
} else {		
idxShbExcResp[0];	**7**	**uimsbf**
idxShbExcResp[1];	**4**	**uimsbf**
}		
} else {		
tbe_nlConfig = 1;		
}		
}		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

The following bit stream syntax is based on ISO/IEC 23003-3:2012, clause 5. These syntax elements have to be added to ISO/IEC 23008-3. The added syntax elements have to be removed from Table 37 in ISO/IEC 23008-3 if existing.

In ISO/IEC 23008-3:2015 Table 37 — References to USAC syntactic elements replace:

Syntax of	**Defined in**
ics_info()	ISO/IEC 23003-3:2012, 5.3, Table 22
tw_data()	ISO/IEC 23003-3:2012, 5.3, Table 27
scale_factor_data()	ISO/IEC 23003-3:2012, 5.3, Table 28
tns_data()	ISO/IEC 23003-3:2012, 5.3, Table 29
ac_spectral_data()	ISO/IEC 23003-3:2012, 5.3, Table 30
lpd_channel_stream()	ISO/IEC 23003-3:2012, 5.3, Table 31
fac_data()	ISO/IEC 23003-3:2012, 5.3, Table 39
UsacSbrData()	ISO/IEC 23003-3:2012, 5.3, Table 40
Mps212Data()	ISO/IEC 23003-3:2012, 5.3, Table 52

with:

Syntax of	**Defined in**
ics_info()	ISO/IEC 23003-3:2012, 5.3, Table 22
tw_data()	ISO/IEC 23003-3:2012, 5.3, Table 27
scale_factor_data()	ISO/IEC 23003-3:2012, 5.3, Table 28
tns_data()	ISO/IEC 23003-3:2012, 5.3, Table 29
ac_spectral_data()	ISO/IEC 23003-3:2012, 5.3, Table 30
fac_data()	ISO/IEC 23003-3:2012, 5.3, Table 39
UsacSbrData()	ISO/IEC 23003-3:2012, 5.3, Table 40
Mps212Data()	ISO/IEC 23003-3:2012, 5.3, Table 52

Add Table AMD2.2 – Syntax of lpd_channel_stream() (based on Table 31 in ISO/IEC 23003-3:2012) to subclause 5.2.3.2:

Table AMD2.2 — Syntax of lpd_channel_stream()

Syntax	No. of bits	Mnemonic
lpd_channel_stream(noiseFilling, fullbandLpd, indepFlag)		
{		
if (fullbandLpd) {		
tns_data_present;	**1**	**uimsbf**
if (noiseFilling || tns_data_present) {		
window_shape;	**1**	**uimsbf**
max_sfb;	**4**	**uimsbf**
} else {		
window_shape = max_sfb = 0;		
}		
} else {		
tns_data_present = 0;		
}		
acelp_core_mode;	**3**	**uimsbf**
if (fullbandLpd) {		NOTE 1
lpd_mode;	**3**	**uimsbf**
bpf_control_info = 1;		
} else {		
lpd_mode;	**5**	**uimsbf**
bpf_control_info;	**1**	**uimsbf**

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
    }
    core_mode_last;                                                  1        uimsbf
    fac_data_present;                                                1        uimsbf

    first_lpd_flag = !core_mode_last;
    first_tcx_flag = TRUE;
    k = 0;
    if (first_lpd_flag) { last_lpd_mode = -1; }                               NOTE 2
    nbDiv = (fullbandLpd == 1) ? 2 : 4;
    while (k < nbDiv) {
        if (k == 0) {
            if (  (core_mode_last == 1) && (fac_data_present == 1) ) {
                fac_data(0, ccfl/8);
            }
        } else {
            if (  (last_lpd_mode == 0 && mod[k] > 0) ||
                  (last_lpd_mode > 0 && mod[k] == 0) ) {
                fac_data(0, ccfl/8);
            }
        }
        if (mod[k] == 0) {
            acelp_coding(acelp_core_mode, fullbandLpd);
            last_lpd_mode = 0;
            k += 1;
        } else {
            window_sequence = EIGHT_SHORT_SEQUENCE;                           NOTE 3
            tcx_coding(lg(mod[k]), first_tcx_flag, tns_data_present, noiseFilling,
                            enhancedNoiseFilling, indepFlag);
            last_lpd_mode = mod[k];
            k += (1 << (mod[k]-1));
            first_tcx_flag = FALSE;
        }
    }

    lpc_data(first_lpd_flag);

    if ( (core_mode_last == 0) && (fac_data_present == 1) ) {
        short_fac_flag;                                              1        uimsbf
        if (fullbandLpd) {
            fac_length = short_fac_flag ? ccfl/32 : ccfl/16;
        } else {
            fac_length = short_fac_flag ? ccfl/16 : ccfl/8;
        }
        fac_data(1, fac_length);
    }
}
```

NOTE 1: **lpd_mode** defines the contents of the array mod[] as described in ISO/IEC 23003-3, Table 89 or, if fullbandLpd is equal to 1, in ISO/IEC 23008-3, Table AMD2.5

NOTE 2: first_lpd_flag is defined in ISO/IEC 23003-3, subclause 6.2.10.2.

NOTE 3: The number of spectral coefficients, lg, depends on mod[k] according to ISO/IEC 23003-3, Table 148 or, if fullbandLpd is equal to 1, in ISO/IEC 23008-3, Table AMD2.7

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Add Table AMD2.3 – Syntax of acelp_coding() (based on Table 36 in ISO/IEC 23003-3:2012:FDAM1) to subclause 5.2.3.2:

Table AMD2.3 — Syntax of acelp_coding()

Syntax	No. of bits	Mnemonic
acelp_coding(acelp_core_mode, fullbandLpd)		
{		
mean_energy;	**2**	**uimsbf**
nb_subfr = coreCoderFrameLength / 256;		NOTE
for (sfr = 0; sfr < nb_subfr; sfr++) {		
if ((sfr == 0) || ((nb_subfr == 4) && (sfr == 2))) {		
acb_index[sfr];	**9**	**uimsbf**
} else {		
acb_index[sfr];	**6**	**uimsbf**
}		
ltp_filtering_flag[sfr];	**1**	**bmsbf**
switch (acelp_core_mode) {		
case 0		
icb_index[sfr];	**20**	**uimsbf**
break;		
case 1		
icb_index[sfr];	**28**	**uimsbf**
break;		
case 2		
icb_index[sfr];	**36**	**uimsbf**
break;		
case 3		
icb_index[sfr];	**44**	**uimsbf**
break;		
case 4		
icb_index[sfr];	**52**	**uimsbf**
break;		
case 5		
icb_index[sfr];	**64**	**uimsbf**
break;		
case 6		
icb_index[sfr];	**12**	**uimsbf**
break;		
case 7		
icb_index[sfr];	**16**	**uimsbf**
break;		
}		
gains[sfr];	**7**	**uimsbf**
}		
if (fullbandLpd) {		
tbe_data();		
}		
}		
NOTE: coreCoderFrameLength designates the core frame length in samples and is equal to either 1024 or 768.		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In subclause 5.5.4.4.1.2 replace

Table 59 — Value of bl

window_sequence	bl
EIGHT_SHORT_SEQUENCE	ccfl / 8
All other sequences	ccfl

with

Table 59 — Value of bl

window_sequence	bl
EIGHT_SHORT_SEQUENCE	ccfl / 8
medium TCX	ccfl / 2
All other sequences	ccfl

In subclause 5.5.4.4.3.2 replace

The result, quantized energy information of scalefactor bands in the IGF region, is stored in igf_curr[ ][ ][ ]. For requantization use the formula:

```
igf_curr[ch][g][sfb] = 2^(igf_curr[ch][g][sfb]*0.25)
```

for each channel ch, windowgroup g and scalefactor band sfb in scope.
with:

The result, quantized energy information of scalefactor bands in the IGF region, is stored in igf_curr[ ][ ][ ]. For requantization use the formula:

```
igf_curr[ch][g][sfb] = 2^((igf_curr[ch][g][sfb]-igf_emphasis)*0.25)
```

for each channel ch, window group g and scalefactor band sfb in scope and where igf_emphasis is defined as:

```
igf_emphasis = 0;  if IGF is running in FD  mode
igf_emphasis = 40; if IGF is running in TCX mode
```

In subclause 5.5.4.4.7 replace

```
igf_apply_whitening(pMDCT[],pMDCT_flat[])
{
  stop = swb_offset[m_igfStartSfb];
  for (i = igfMin - 3; i < stop; i++) {
    power_spec[i] = pMDCT[i] * pMDCT[i];
  }

  for (i = igfMin; i < stop-3; i++) {
    env[i] = 1e-3;
    for (j = -3; j <= 3; j++) {
      env[i] += power_spec[i+j];
    }
  }

  for (i = stop-3; i < stop; i++) {
    env[i] = env[stop-4];
  }
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
  for (i = igfMin; i < stop; i++) {
    n = INT(log(env[i])/log(2));
    pMDCT_flat[i] = pMDCT[i] * pow(2, 21 – 0.5 * n);
  }
}
```

with:

The IGF whitening is performed by dividing the spectrum in pMDCT[] by the scaled square root of its estimated spectral envelope in env[]:

$$pMDCT_{flat}[k] = pMDCT[k] \cdot \frac{2^{21}}{\sqrt{env[k]}}, k = igfMin, igfMin + 1, \ldots, igfBgn - 1$$

where the spectral envelope is estimated by filtering the squared spectrum using a moving average filter of length 7 where each filter coefficient equals 1.

```
igf_apply_whitening(pMDCT[], pMDCT_flat[])
{
  stop = swb_offset[m_igfStartSfb];
  for (i = igfMin; i < stop - 3; i++) {
    env = 1e-3;
    for (j = i - 3; j <= i + 3; j++) {
      env += pMDCT[j] * pMDCT[j];
    }

    n = ~~MAX(0,~~ INT(log(env)/log(2))~~)~~;
    fac = pow(2, 21 – 0.5 * n);
    pMDCT_flat[i] = pMDCT[i] * fac;
  }

  for (; i < stop; i++) {
    pMDCT_flat[i] = pMDCT[i] * fac;
  }
}
```

Add subclause 5.5.5.4.x IGF core rescaling after subclause 5.5.5.4.11 IGF-ENF Processing:

5.5.5.4.x IGF core rescaling

If IGF is used in TCX mode, i.e. **fullbandLpd** is equal to 1 and the current frame is LPD/TCX coded**,** the decoded IGF levels are used for rescaling the MDCT coefficients in the IGF range. This process mimics the quantization noise adjustment described in ISO/IEC 14496-3:2009 subclause 4.6.2 for the TCX core coder.

After decoding the IGF levels as described in subclause 5.5.5.4.4, for each IGF scalefactor band the corresponding IGF level stored in igf_curr[ ] is multiplied on the MDCT coefficients of the TCX spectrum of the current frame:

```
for (tile = 0; tile < nT; tile++) {
  for (sfb = m_igfStartSfb; sfb < m_igfStopSfb; sfb++)
    width = (swb_offset[MIN(sfb + 1, m_igfStopSfb)] - swb_offset[sfb]);
    for (bin = 0; bin < width; bin++) {
      tb = swb_offset[sfb]+bin;
      pMDCT[tb] *= igf_curr[sfb];
  }
}
```

As TCX does not support grouping of windows, i.e. num_windows = 1, the window index w is always 0.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Add subclauses 5.5.x Fullband LPD and 5.5.x Time-domain Bandwidth Extension after 5.5.6 Audio Pre-Roll :

5.5.x Fullband LPD

5.5.x.1 Tool Description

In the fullband LPD mode, the ACELP core mode (see subclause 7.14 in ISO/IEC 23003-3:2012) is operated at half the sampling frequency as the MDCT based TCX core mode (see subclause 7.15 in ISO/IEC 23003-3:2012), i.e.

$$f_{s,TCX} = 2 \cdot f_{s,ACELP}$$

To adjust the output signals at different sampling frequencies, the ACELP output is upsampled applying a time-domain resampler. The lacking effective bandwidth ranging from $f_{s,ACELP}/2$ to $f_{s,TCX}/2$ in case of ACELP processing in comparison to TCX in combination with Enhanced Noise Filling (see subclause 5.5.x.8.5 Enhanced Noisefilling in TCX) is covered by the Time-Domain Bandwidth Extension (see subclause 5.5.x Time-domain Bandwidth Extension). This section describes the decoding of fullband LPD and related changes in decoding of existing tools.

5.5.x.2 Data Elements

fullbandLpd	this flag signals the usage of the fullband LPD tool.
window_shape	window shape of the current subframe.

5.5.x.3 Helper Elements

$f_{s,TCX}$	sampling rate of the fullband TCX.
$f_{s,ACELP}$	sampling rate of the ACELP.
lg	number of spectral coefficients of the current TCX subframe.
rl[ ]	TCX noise generation flag array, contains only the values 1 and 0.
rr[ ]	reconstructed spectrum.
x[ ]	output of the IMDCT.
nfBgn	subband index; this index indicates the start of the TCX noise generation.
nfEnd	subband index; this index indicates the stop of the TCX noise generation.
g	re-scaling gain factor.
$z_{TCX,FB}$	time-domain output of TCX sampled at $f_{s,TCX}$.
$z_{TCX,LB}$	time-domain output of TCX sampled at $f_{s,ACELP}$.
z_{ACELP}	time-domain output of ACELP sampled at $f_{s,ACELP}$.
z_{OUT}	mixed time-domain output sampled at $f_{s,TCX}$.
z_fb[ ]	decoded windowed time domain signal of the fullband TCX @ $f_{s,TCX}$.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

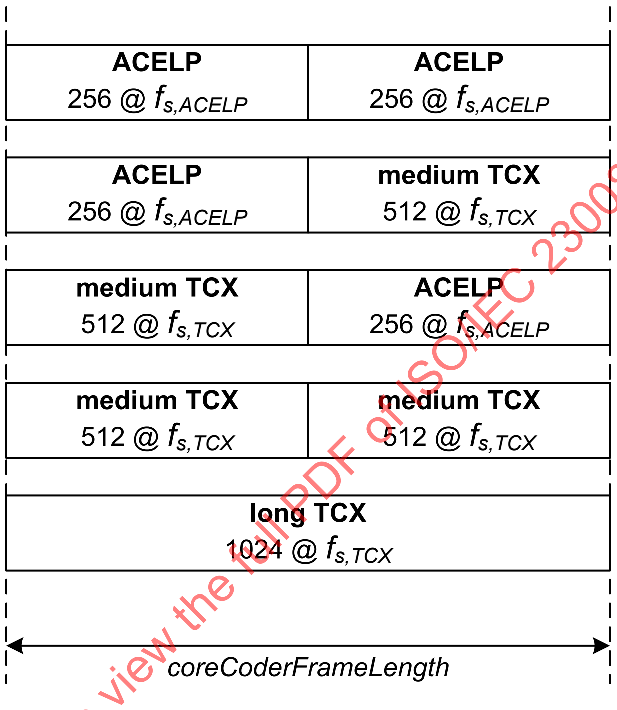
© ISO/IEC 2017 – All rights reserved

z_lb[ ]	decoded windowed time domain signal of the fullband TCX @ $f_{s,ACELP}$.

5.5.x.4 Framing

In case of fullband LPD coding, i.e. **fullbandLpd** is equal to 1, long TCX frames correspond to coreCoderFrameLength samples @ $f_{s,TCX}$ and medium TCX frames correspond to coreCoderFrameLength/2 samples @ $f_{s,TCX}$. For ACELP, the standard frame size of coreCoderFrameLength/4 samples @ $f_{s,ACELP}$ as described in subclause 7.14 in ISO/IEC 23003-3:2012 is used, while coreCoderFrameLength=1024 is fixed. Possible frame combinations within one superframe are shown in Figure **AMD2.1**.

Figure AMD2.1 — possible frame combinations in fullband LPD within one superframe



5.5.x.4.1 Mode coding

In case of fullband LPD, i.e. **fullbandLpd** is equal to 1, the lpd mode coding is the same as described in subclause 6.2.10 in ISO/IEC 23003-3:2012, while one "superframe" consists of only two frames instead of four. Due to this, Table 89 in ISO/IEC 23003-3:2012 changes to Table **AMD2.5**.

Table AMD2.5 — Mapping of coding modes for lpd_channel_stream() in case of fullband LPD

	meaning of bits in bit-field lpd_mode			**remaining mod[] entries**
lpd_mode	**bit 2**	**bit 1**	**bit 0**	
0..3	0	mod[1]	mod[0]	

4	1	0	0	mod[1]=2
				
				mod[0]=2

In case of fullband LPD, as there are only two frames, coding mode 3 is not applicable, thus changing Table 92 in ISO/IEC 23003-3:2012 to Table **AMD2.6**.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table AMD2.6 — Coding modes indicated by mod[ ] in case of fullband LPD

value of mod[x]	coding mode in frame	bitstream element
0	ACELP	acelp_coding()
1	medium TCX (ccfl/2)	tcx_coding()
2	long TCX (ccfl)	tcx_coding()

In case of fullband LPD, as there are only two frames, coding mode 3 is not applicable, thus changing Table 148 in ISO/IEC 23003-3:2012 to Table AMD2.7.

Table AMD2.7 — number of spectral coefficients as a function of mod[] and coreCoderFrameLength (ccfl)

value of mod[x]	number lg of spectral coefficients	ZL	L	M	R	ZR
1	ccfl/2	0	ccfl/2	0	ccfl/2	0
2	ccfl	ccfl/4	ccfl/2	ccfl/2	ccfl/2	ccfl/4

5.5.x.5 TD resampler

The time-domain (TD) resampler is used to upsample the time-domain output of ACELP (see subclause 7.14 in ISO/IEC 23003-3:2012) and the Forward Aliasing Cancellation tool (see subclause 7.16 in ISO/IEC 23003-3:2012) by the factor of 2, in order to bring the output to the same sampling rate as the fullband MDCT based TCX output.

Every 2 samples of the time-domain input signal are separated by a zero-valued sample to form a sequence with a sampling frequency ($f_{s,out}$) which is increased by the factor of 2 compared to the sampling frequency of the input signal ($f_{s,in} = f_{s,out}/2$). Subsequently, the upsampled sequence s_{up} is filtered using an FIR interpolation filter of length N given by the symmetric filter coefficients b_k in Table AMD2.8 to form the time-domain output signal s_{out} for each sample of index i by:

$$s_{out}(i) = \sum_{k=0}^{N-1} b_k \cdot s_{up}\,(i-k)$$

The resampler introduces a delay of 30 samples at $f_{s,out}$ to the upsampled signal s_{out}.

Table AMD2.8 — interpolation filter coefficients bk

k	b_k
30	1
29, 31	0.63487604
28, 32	0
27, 33	-0.20701353
26, 34	0
25, 35	0.11879487
24, 36	0
23, 37	-0.07926575
22, 38	0

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

21, 39	0.05615642
20, 40	0
19, 41	-0.04070711
18, 42	0
17, 43	0.02957617
16, 44	0
15, 45	-0.02122066
14, 46	0
13, 47	0.01483115
12, 48	0
11, 49	-0.00993903
10, 50	0
9, 51	0.00624819
8, 52	0
7, 53	-0.00355476
6, 54	0
5, 55	0.00170582
4, 56	0
3, 57	-0.00057701
2, 58	0
1, 59	0.00006013
0, 60	0

5.5.x.6 LPC filter

Yet, as there are only 2 ACELP frames per superframe in case of fullbandLPD, the maximum number of LPC-filters to be present within 1 superframe changes from 5 to 3. The LPC-filters LPC2 and LPC3 are not used, while the remaining LPC-filters correspond to LPC0, LPC1 and LPC4. Thus, in case of fullbandLPD, the conditions of presence of LPC-filters are defined according to Table **AMD2.9**, and possible quantization modes are defined according to Table **AMD2.10X**

Table AMD2.9 — Conditions for the presence of a given LPC filter in the bitstream in fullbandLPD

LPC filter	**Present if**
LPC0	**first_lpd_flag**=1
LPC1	mod[0]<2
LPC4	always

Table AMD2.10X — Allowed absolute and relative quantization modes, corresponding bitstream signaling of mode_lpc and coding modes for codebook numbers nk

Pos. in Bitstr.	**Present if**	**Filter**	**Quantization mode**	**mode_lpc**	**Binary Code**	***n_k* mode**
1.	always	LPC4	Absolute	0	(none)	0
2.	first_lpd_flag=1	LPC0	Absolute	0	0	0
			Relative to LPC4	1	1	3
3.	mod[0]<2	LPC1	Absolute	0	10	0
			Relative to (LPC0+LPC4)/2 (NOTE 1)	1	11	1
			Relative to LPC4	2	0	2
NOTE 1: in this mode, there is no second-stage AVQ quantizer						

5.5.x.7 ACELP decoding

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In case of fullband LPD, i.e. **fullbandLpd** is equal to 1, ACELP is decoded and processed as described in subclause 7.14 in ISO/IEC 23003-3:2012. Subsequent to the ACELP synthesis and prior to writing it to the output buffer, the ACELP output is upsampled by the factor of 2 using the time-domain resampler described in subclause **5.5.x.5. TD resampler**. The TD resampler is initialized by filtering previous samples to enable delayless upsamling.

5.5.x.8 TXC decoding

5.5.x.8.1 TCX frequency band offset tables

For the noise filling, Intelligent Gap Filling, and Temporal Noise Shaping algorithms applied in the MDCT based TCX, a vector swb_offset_tcx[] of frequency band offsets (i. e. indices of spectral bins representing frequency band borders) is required. Identically to 8-short-window FD channels and frames coded using window_sequence == EIGHT_SHORT_SEQUENCE, these TCX band offsets are based on the sampling-rate dependent swb_offset_short_window[] values defined in Tables 4.130—4.141 in ISO/IEC 14496-3:2009, subclause 4.5.4 and may be abbreviated swb_offset in functions or algorithm descriptions available to both long-window and 8-short-window FD coding and/or to both FD and LPD (i. e. MDCT based TCX) coding.

The swb_offset_tcx[] vector for a given TCX window is determined using value lg from Table **AMD2.7**:

swb_offset_tcx[i] = swb_offset_short_window[i] · f(lg),

where f(lg) = lg / 128, with lg being based on vector mod[ ] defined according to Table **AMD2.2** — Syntax of lpd_channel_stream() in subclause 5.2.3.2.

5.5.x.8.2 TCX noise generation

The noise filling in MDCT based TCX is applied as described in ISO/IEC 23003-3:2012 subclause 7.15.3. However, the noise filling start index, lg/6 and stop index, lg are modified as follows if **fullbandLpd** == 1.

A run of 8 non-zeros is detected according to the following pseudo code, where nfBgn and nfEnd depend on **fullbandLpd** and **enhancedNoiseFilling** and, in case of the latter, the swb_offset_tcx bin index array:

```
if (fullbandLpd) {
  nfBgn = (lg/6) & 2040;
  nfEnd = enhancedNoiseFilling ? igfBgn : lg;
  nfEnd = min(nfEnd, min(lg, swb_offset_tcx[max_sfb]));
} else {
  nfBgn = (lg/6);
  nfEnd = lg;
}

for (i = 0; i < nfBgn; i++) {
  rl[i] = 1;
}

for (i = nfBgn; i < nfEnd; i += 8) {
  int k, maxK = min(nfEnd, i+8);
  tmp = 0;
  for (k = i; k < maxK; k++) {
    tmp += x_tcx_invquant[k] * x_tcx_invquant[k];
  }
  if (tmp != 0) {
    for (k = i; k < maxK; k++) {
      rl[k] = 1;
    }
  } else {
    for (k = i; k < maxK; k++) {
      rl[k] = 0;
    }
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
  }
}
```

5.5.x.8.3 Adaptive Low-Frequency De-emphasis

The purpose of the adaptive low-frequency emphasis and de-emphasis (ALFE) processes is to improve the subjective performance of the frequency-domain TCX codec at low frequencies. To this end, the TCX low-frequency MDCT spectral lines are amplified prior to quantization in the encoder, thereby increasing their quantization SNR, and this boosting is undone prior to the inverse MDCT operation in the decoder.

The ALFE operates on the spectral lines in vector x[] directly after the above-noted inverse quantization, noise filling and if enabled, frequency-domain prediction (i. e. x[] represents either the x_tcx_invquant[] or outputSpecCurr[]). If both **fullbandLpd** and **enhancedNoiseFilling** are zero the conventional ALFE algorithm, described as a four-step "de-shaping" procedure in ISO/IEC 23003-3:2012 subcl. 7.15, is applied.

Otherwise, the ALFE operates based on the LPC frequency-band gains, lpcGains[], which are derived from the gains g1 and g2 used for FD noise shaping, as defined in ISO/IEC 23003-3:2012, subcl. 7.15, by

lpcGains[k] = sqrt(g1[k] * g2[k]), i. e. the geometric mean of g1 and g2 at each index k.

The ALFE decoding is done as follows. First, the minimum and maximum of the first nine gains – the low-frequency gains – are found using comparison operations executed within a loop over the gain indices 0 to 8., i. e. over lpcGains[*i*] with *i* = 0, 1,.., 8. Then, if the ratio between the minimum and maximum gain values exceeds a threshold of 1/32, a gradual lowering of the lowest lines in x is performed such that the line at index 0 is attenuated by $(\max/(32 \cdot \min))^{0.25}$ and the line at index alfeLength is not attenuated:

```
alfeLength = lg / 8;
tmp = 32 * min;
if ((max < tmp) && (tmp > 0)) {
  fac = tmp = pow(max / tmp, 1/(4 * alfeLength));
  /* gradual lowering of lowest alfeLength lines: */
  for (i = alfeLength-1; i >= 0; i--) {
    x[i] *= fac;
    fac  *= tmp;
  }
}
```

5.5.x.8.4 Adaptive Low-Frequency De-emphasis

In MDCT based TCX coding, the reconstructed spectrum rr[] is fed into an inverse MDCT. The non-windowed output signal, x[], is re-scaled by the gain, g, obtained by an inverse quantization of the decoded global_gain index as described in ISO/IEC 23003-3:2012 subclause 7.15.3. If **fullbandLpd** is equal to 1, the rescaling of the spectrum is performed in the MDCT domain. The stop index of the core coder rescaling depends on **enhancedNoiseFilling**:

```
rsEnd = enhancedNoiseFilling ? min(lg, igfBgn) : lg;
rr[i] = rr[i]·g; i = 0 .. rsEnd - 1
```

5.5.x.8.5 Enhanced Noise Filling in TCX

In MDCT based TCX coding, Enhanced Noise Filling is carried out by the Intelligent Gap Filling (IGF) tool as described in subclause 5.5.5 if **enhancedNoiseFilling** is equal to 1.

The IGF decoding process for each TCX spectrum is performed after the arithmetic decoding noise filling and rescaling, but before the de-shaping is applied to the spectrum.

5.5.x.8.6 De-shaping

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

The spectrum de-shaping is applied to the reconstructed spectrum according to ISO/IEC 23003-3:2012 subclause 7.15.3. If **fullbandLpd** is equal to 1, the inverse FDNS operation consists in filtering the reconstructed spectrum r[i] using the recursive filter:

```
rr[i] = a[i]·r[i]+b[i]·rr[i-1], i = 0 ... lg/2-1,
```

where a[i] and b[i] are derived from the left and right gains g1[k], g2[k] using the formulas:

```
a[i] = 2·g1[k]·g2[k]/(g1[k]+g2[k]),
b[i] = (g2[k]-g1[k])/(g1[k]+g2[k])
```

In the above, the variable k is equal to i/(lg/M) to take into consideration the fact that the LPC spectrums are decimated, where M=coreCoderFrameLenght/16.

The spectral coefficients between lg/2 and lg are filtered by holding the last calculated filter coefficients:

```
rr[i] = a[i]·r[i]+b[i]·rr[i-1], i = lg/2 ... lg-1,
```

where a[i] and b[i] are derived from the left and right gains g1[M-1], g2[M-1] using the formulas:

```
a[i] = 2·g1[M-1]·g2[M-1]/(g1[M-1]+g2[M-1]),
b[i] = (g2[M-1]-g1[M-1])/(g1[M-1]+g2[M-1])
```

5.5.x.8.7 Temporal Noise Shaping in TCX

In MDCT based TCX coding, Temporal Noise Shaping (TNS) is applied as in short-block FD channels (i.e. channels and frames with **window_sequence**==EIGHT_SHORT_SEQUENCE) and follows the bitstream syntax and description specified in ISO/IEC 23003-3:2012, subcl. 5.3.2 and 7.8. If **tns_data_present** ≠ 0 for a given lpd_channel_stream(), a single-window instance of tns_data() is read for each TCX spectrum.

The TNS decoding (i.e. synthesis filtering) process for each TCX spectrum is performed after the arithmetic decoding, noise filling, frequency-domain prediction and enhanced noise filling steps, as referenced in ISO/IEC 23003-3:2012, subclause 7.8 and, in case of Intelligent Gap Filling with TTS, subclause 5.5.5.4.5. The only difference is that the swb_offset_short_window[] values employed for frequency band restriction of the TNS filtering process are multiplied with a factor *f*(mod[k]), as described in subclause **5.5.x.8.1 TCX frequency band offset tables**.

5.5.x.8.8 Inverse MDCT in TCX

The reconstructed spectrum rr[] is fed into an inverse Modified Discrete Cosine Transform (IMDCT) to obtain the non-windowed time domain output signal x[] as described in ISO/IEC 23003-3:2012. If **fullbandLpd** is equal to 1, it is necessary to perform two independent IMDCTs, one for the fullband TCX output z_fb[] and one for the down-sampled lowband TCX output z_lb[] which will be later used for initializing the ACELP core. The down-sampling by the constant factor of 2 is achieved by applying an IMDCT of half the length of the regular IMDCT, i.e. by applying an IMDCT of length lg/2.

5.5.x.8.9 TCX windowing

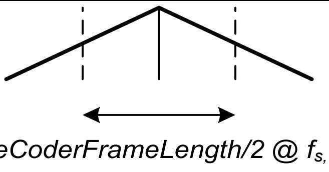
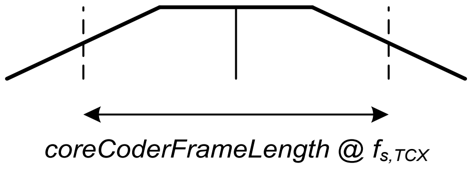
In case of fullband LPD, i.e. **fullbandLpd** is equal to 1, the windows applied to the TCX frames prior to the transform and after inverse transform are depicted in Table **AMD2.10**.

Table AMD2.10 — window shapes for TCX frames in fullband LPD

Frame	**Window Shape (schematic)**
medium TCX	

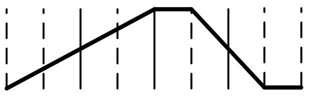
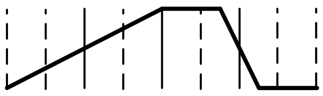
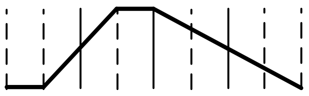
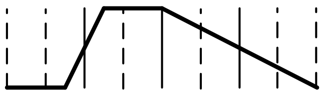
STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

	
long TCX	

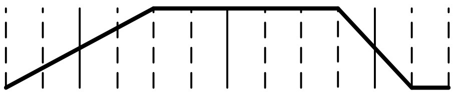
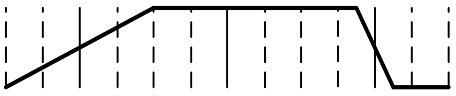
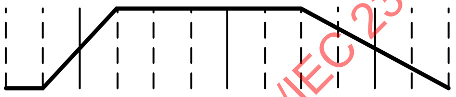
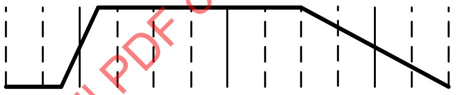
However, when switching from or to FD mode, the TCX windows are adapted in fullband LPD mode, i.e. **fullbandLpd** is equal to 1, while the windows for the FD core remain the same. The TCX transitions window shapes are schematically depicted in Table **AMD2.11** and Table **AMD2.12**.

Table AMD2.11 — transition window shapes of length N=1024 samples for medium TCX frames in fullband LPD

Transition	**Window Shape (schematic)**
medium TCX → LONG_STOP_SEQUENCE / STOP_START_SEQUENCE	
medium TCX → EIGHT_SHORT_SEQUENCE	
LONG_START_SEQUENCE / STOP_START_SEQUENCE → medium TCX	
EIGHT_SHORT_SEQUENCE → medium TCX	

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table AMD2.12 — transition window shapes of length N=1536 samples for long TCX frames in fullband LPD

Transition	Window Shape (schematic)
long TCX → LONG_STOP_SEQUENCE / STOP_START_SEQUENCE	
long TCX → EIGHT_SHORT_SEQUENCE	
LONG_START_SEQUENCE / STOP_START_SEQUENCE → long TCX	
EIGHT_SHORT_SEQUENCE → long TCX	

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

The calculation formulae of the window shapes are described in subclause 7.9.3.2 of ISO/IEC 23003-3:2012. In accordance to the previous frame, the left slope of the window shape and according to the presence of **window_shape**, the right slope of the window shape can either be a sine or a KBD window. For **window_shape** == 1, the window coefficients used for the IMDCT are given by the Kaiser – Bessel derived (KBD) window, otherwise the sine window is employed (see subclause 7.9.3.2 of ISO/IEC 23003-3:2012).

5.5.x.9 Forward Aliasing Cancellation (FAC) tool

In case of fullband LPD, i.e. **fullbandLpd** is equal to 1, the FAC tool is decoded as described in subclause 7.16 in ISO/IEC 23003-3:2012. Subsequent to the decoding and prior to writing it to the output buffer, the FAC signal is upsampled by the factor of 2, the same way as described in subclause **5.5.x.7 ACELP decoding**, using the time-domain resampler described in subclause **5.5.x.5 TD resampler**.

The different core coder sampling frequencies $f_{s,ACELP}$ & $f_{s,TCX}$ in case of fullband LPD need to be considered when determining the length of the FAC transform (fac_length) for decoding ($f_{s,ACELP}$) and writing to the output buffer ($f_{s,TCX}$):

fac_length = *coreCoderFrameLength* / 32 @ $f_{s,ACELP}$ if transitioning between ACELP and EIGHT_SHORT_SEQUENCES;

fac_length = *coreCoderFrameLength* / 16 @ $f_{s,ACELP}$ if transitioning from ACELP to LONG_STOP_SEQUENCE or from LONG_START_SQUENCE to ACELP;

fac_length = *coreCoderFrameLength* / 8 @ $f_{s,ACELP}$ if transitioning between ACELP and TCX;

© ISO/IEC 2017 – All rights reserved

5.5.x.10 Post-processing of the synthesis signal

In case of fullband LPD, i.e. **fullbandLpd** is equal to 1, the post-processing of the synthesis signal is performed as described in subclause 7.17 in ISO/IEC 23003-3:2012. While the control coefficient for the inter-harmonic attenuation α and the post-filter gain g_{PF} are determined based on the ACELP synthesis signal before resampling (see subclause **5.5.x.7 ACELP decoding**), the filtering is applied to the upsampled ACELP synthesis output. In case of post-processing of transitions between FD mode to and from ACELP, the adapted FAC area lengths must be considered (see subclause **5.5.x.9 Forward Aliasing Cancellation (FAC) tool**). The bass-post filter is always enabled for ACELP frames and FAC areas in case of fullband LPD, constantly changing the value of bpf_control_info in Table 90 in ISO/IEC 23003-3:2012 to bpf_control_info=1.

5.5.x.11 Coding mode switching

As described in subclause **5.5.x.8.8. Inverse MDCT in TCX**, the MDCT based TCX decoder generates 2 time-domain output signals at sampling frequencies $f_{s,ACELP}$ and $f_{s,TCX}$, respectively. The signal sampled at $f_{s,ACELP}$ is used to update the ACELP memories to enable seamless transitions when switching between MDCT based TCX and ACELP.

To avoid discontinuities, crossfading is applied as given by

$$z_{OUT}\left[N_{frame}-N_{xfade}+k\right] = z_{TCX,FB}\left[N_{frame}-N_{xfade}+k\right]\cdot\left(\frac{N_{xfade}-k}{N_{xfade}}\right)+z_{TCX,LB}\left[N_{frame}-N_{xfade}+k\right]\cdot\left(\frac{k}{N_{xfade}}\right)$$

$$for\ k=0,\ldots,N_{xfade}-1$$

and depicted schematically in Table **AMD2.13** for switching from MDCT based TCX to ACELP, and by

$$z_{OUT}[k] = z_{TCX,FB}[k]\cdot\left(\frac{k}{N_{xfade}}\right)+z_{TCX,LB}[k]\cdot\left(\frac{N_{xfade}-k}{N_{xfade}}\right)$$

$$for\ k=0,\ldots,N_{xfade}-1$$

and depicted schematically in Table **AMD2.14** for switching from ACELP to MDCT based TCX, where N_{frame} is the frame length of the TCX frame in samples at $f_{s,TCX}$ and N_{xfade} is the length of the crossfade range in samples at $f_{s,TCX}$, which corresponds to the delay of the TD resampler (30 samples at $f_{s,TCX}$) described in subclause **5.5.x.5 TD resampler**.

Table AMD2.13 — schematic depiction of switching from TCX to ACELP

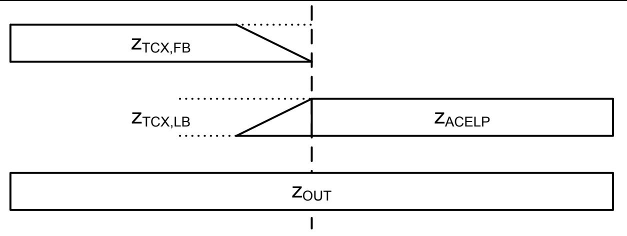
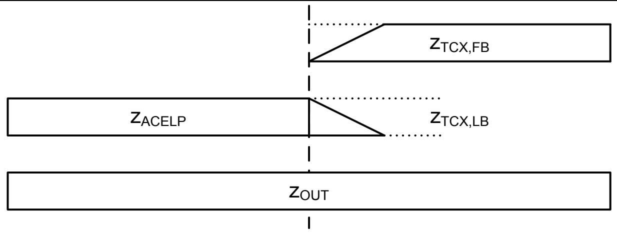
coding mode	TCX	ACELP
frame index	n-1	n
fullband TCX output		
upsampled lowband TCX and ACELP output		
mixed fullband output		

Table AMD2.14 — schematic depiction of switching from ACELP to TCX

coding mode	ACELP	TCX

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

frame index	n-1	n
fullband TCX output	 $z_{TCX,FB}$ z_{ACELP} $z_{TCX,LB}$ z_{OUT}	
upsampled lowband TCX and ACELP output		
mixed fullband output		

5.5.x Time-domain Bandwidth Extension

5.5.x.1 Introduction

This clause describes the decoding process of time-domain bandwidth extension (TBE). The TBE decoder tool is used to enable low bit rate coding of speech via the MPEG-H 3D audio codec's LPD path for ACELP mode.

5.5.x.2 General Overview

Figure AMD2.2 shows a high level framework of the TBE decoder. The input bit stream is de-multiplexed and decoded by the MPEG-H 3D Audio core decoder to produce the ACELP low band (LB) excitation and low band synthesis. The TBE bit stream is parsed and the parameters are passed to the TBE decoding tool. The high band synthesis is performed using the TBE parameters and the harmonically-extended high band excitation signal.

The synthesized high band is then up-sampled and spectrally flipped in the time-domain to generate a high band component associated with the final decoded audio. The low band is also up-sampled to the same sampling rate as the high band, and then mixed with the "delay-adjusted" high band component to generate the output. In particular, the low-band core decoder may exhibit more delay than the high band processing, which would require that the high band is delayed accordingly before mixing with low band such that the low band and high band are time-aligned to avoid any artifacts.

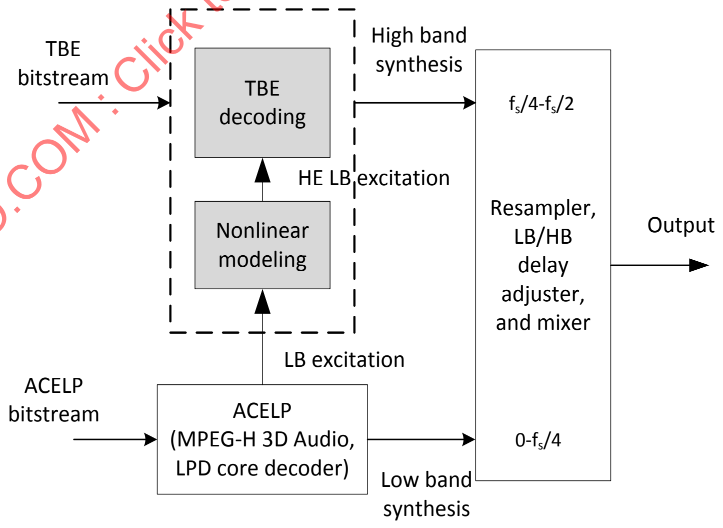


Figure AMD2.2 — Simplified block diagram of the MPEG-H 3D audio core decoder with TBE tools.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

5.5.x.3 Overview of the TBE Decoding Tools

Figure **AMD2.3** shows an overview of the TBE decoder tools. The TBE frame converter parses the TBE bit stream configuration data, tbe_data(), as described in Table AMD2.1, and passes the parameters to the TBE decoding module. The parameters associated with the TBE configuration data are described in subclause **5.5.x.4 Definitions of TBE payloads**. The acelp_coding() configuration data as described in Table AMD2.3 is used by the MPEG-H 3D audio LPD core decoder to generate the low band excitation, E_{LB} (E_LB). The nonlinear modeling module is used to generate the harmonically-extended high band excitation signal, E_{HE} (E_HE).

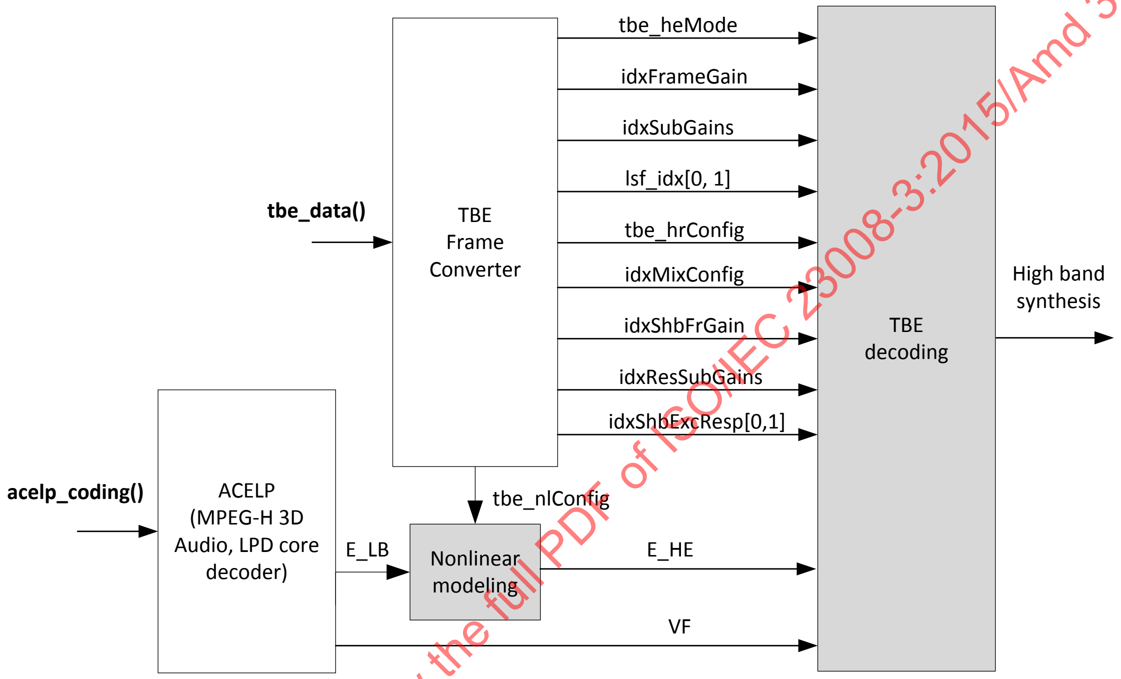


Figure AMD2.3 — Overview of the TBE decoder tools

The TBE decoding process includes the following steps:

- TBE decoding
  - Nonlinear modeling
    - Resampling of LB excitation
    - Harmonically extending the LB excitation based on tbe_nlConfig
  - Dequantization of high band parameters
    - Line spectral frequencies, temporal gains (gain shapes and gain frame), mixing configuration, high band reference gain, residual gain shapes, high band excitation inverse function.
  - High band LP estimation
  - High band excitation generation
    - **A**: Spectral flip in time domain and decimation of E_HE
    - **B**: Adaptive whitening of **A**

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

- **C**: Temporal envelope-modulated noise generation based on **B**
- HB excitation estimation based on **B** and **C**

— High band LP synthesis

— Temporal envelope adjustment of HB synthesis

— Spectral flip and upsampling.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

5.5.x.4 Definitions of TBE payloads

tbe_data()	This element contains information about the high band audio content.
tbe_heMode	This element determines whether the TBE decoding of current frame uses low bit rate high efficiency mode. If the flag is set to zero, then the high resolution configuration (tbe_hrConfig) is enabled.
idxFrameGain	This payload contains data for the overall frame gain adjustment.
idxSubGains	This payload contains data for the temporal sub-frame gain shape adjustment.
lsf_idx[0]	This payload contains LSF data associated with the first stage LSFs used to estimate the LSP and then subsequently the interpolated sub-frame LP parameters.
lsf_idx[1]	This payload contains LSF data associated with the second stage LSFs used to estimate the LSP and then subsequently the interpolated sub-frame LP parameters.
tbe_hrConfig	This flag signals whether the current frame uses high resolution configuration. The flag is only read from the bit stream if the tbe_heMode is set to zero.
tbe_nlConfig	This flag signals the NL configuration that is to be used to generate the HE LB excitation. The flag is only read from the bit stream if the tbe_heMode is set to zero. The default value of tbe_nlConfig is set to 1, if tbe_heMode is set to 1.
idxMixConfig	This payload contains data to control HB excitation generation based on **B** and **C** in subclause **5.5.x.5 TBE Decoder Processing**. The flag is only read from the bit stream if the tbe_heMode is set to zero.
idxShbFrGain	This payload contains data for the overall high band target gain. The flag is only read from the bit stream if the tbe_heMode is set to zero and tbe_hrConfig is set to 1.
idxResSubGains	This payload contains data for temporal sub-frame residual gain shape adjustment. The flag is only read from the bit stream if the tbe_heMode is set to zero and tbe_hrConfig is set to 1.
idxShbExcResp[0]	This payload contains data to filter the HE excitation **B** in subclause **5.5.x.5 TBE Decoder Processing**. The flag is only read from the bit stream if the tbe_heMode is set to zero and tbe_hrConfig is not set to 1.
idxShbExcResp[1]	This payload contains data to filter the HE LB excitation **B** in subclause **5.5.x.5 TBE Decoder Processing**. The flag is only read from the bit stream if the tbe_heMode is set to zero and tbe_hrConfig is not set to 1.

5.5.x.5 TBE Decoder Processing

© ISO/IEC 2017 – All rights reserved

5.5.x.5.1 General Overview

An overview of TBE decoder processing steps is shown in Figure AMD2.4.

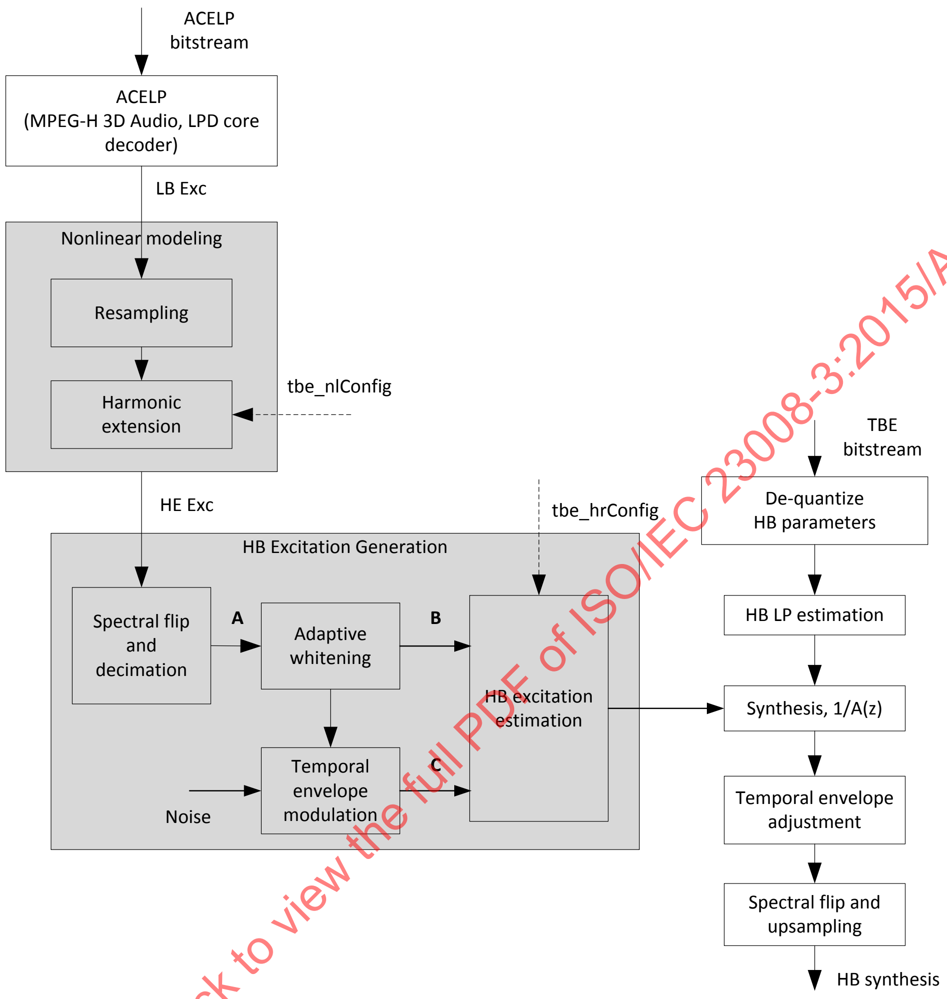


Figure AMD2.4 — Overview of the TBE decoder processing

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

5.5.x.5.2 De-quantization of HB Parameters

The codebooks used to de-quantize some of the high band TBE parameters are summarized in Table AMD2.15. Pseudo-code that describes the de-quantization process is given below.

Table AMD2.15 — List of codebook tables used to de-quantize some of the high band TBE parameters

Codebook Table	Parameter
SHBCB_GainFrame5bit	Gain frame, Table X.6, 5 bits
SHBCB_SubGain5bit	Subframe gains, Table X.5, 5 bits
tbeLSFCB1_7b	LSF first stage, Table X.1, 7 bits
tbeLSFCB2_7b	LSF second stage, Table X.2, 7 bits
tbeExcFilterCB1_7b	tbeExcFilter1, Table X.3, 7 bits
tbeExcFilterCB2_4b	tbeExcFilter2, Table X.4, 4 bits

```
frameGain = SHBCB_GainFrame5bit[idxFrameGain];

j = 4*idxSubGain;
for (i = 0; i < 4; i++) {
  subGain[i] = SHBCB_SubGain5bit[j++];
}

copyVector(tbeLSFCB1_7b + 10 * lsf_idx[0], qLsf,  10);
copyVector(tbeLSFCB2_7b + 10 * lsf_idx[1], qtemp, 10);
for (i = 0; i < 10; i++) {
  qLsf[i] = qLsf[i] + qtemp[i];
}

if (tbe_heMode==0) {
  mixFac = (idxMixConfig + 1) / 4;
  if (tbe_hrConfig == 1) {
    hbEnerTarget = 10^(0.0625 * idxShbFrGain);
    j = 4 * idxResSubGains;
    for (i = 0; i < 4; i++) {
      resSubGain[i] = SHBCB_SubGain5bit[j++];
    }
  } else {
    copyVector(tbeExcFilterCB1_7b + 10 * idxShbExcResp[0], tbeExcFilter1, 10);
    copyVector(tbeExcFilterCB2_4b +  6 * idxShbExcResp[1], tbeExcFilter2,  6);
  }
}
```

5.5.x.5.3 Nonlinear modeling

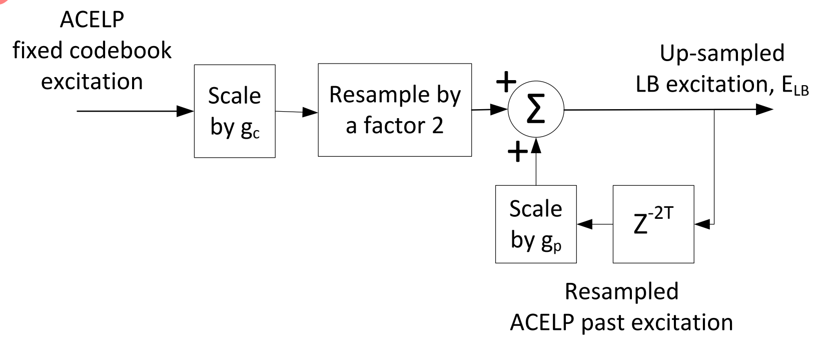
5.5.x.5.3.1 introduction

This clause describes the steps to generate the high band excitation from the low band ACELP core. To generate a high-band excitation signal that preserves the harmonic structure of the low band excitation signal, a nonlinear function is used. The time-domain harmonic extension of the low band excitation is performed after sufficient over-sampling in order to minimize aliasing.

5.5.x.5.3.2 Resampling of LB excitation

As shown in Figure **AMD2.5**, an up-sampled low band excitation signal is derived from both the periodic (adaptive codebook, ACB) and aperiodic (fixed codebook, FCB) excitation components of the low band ACELP core coder. The ACELP innovation codebook excitation is first scaled by the FCB gain, g_c, and then up-sampled by 2. A simple linear interpolator is used to perform the resampling of the scaled fixed codebook excitation by a factor of 2 as show in Figure **AMD2.5**. The ACB component in the resampled domain is obtained by shifting the past resampled LB excitation, E_{LB}, based on the fractional closed-loop pitch estimate, T, from the current frame. The up-sampled past excitation samples are scaled by the pitch gain, g_p, and combined with the up-sampled FCB excitation to generate the resampled low band excitation, E_{LB} as shown in Figure **AMD2.5**.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017


ACELP fixed codebook excitation
Scale by g_c
Resample by a factor 2
+
Σ
+
Up-sampled LB excitation, E_{LB}
Scale by g_p
Z^{-2T}
Resampled ACELP past excitation

© ISO/IEC 2017 – All rights reserved

Figure AMD2.5 — Overview of low band excitation resampling process

5.5.x.5.3.3 Harmonic Extension

The resampled low band excitation signal, E_{LB}, is processed to extend the low band pitch harmonics into the high band. The harmonically extended excitation, E_{HE}, is generated using a nonlinear function, based on the tbe_nlConfig flag from tbe_data().

$$E_{HE} = \begin{cases} |E_{LB}|, & if\ tbe\_nlConfig = 1 \\ \varepsilon_N\ sign(E_{LB})\ E_{LB}^2, & if\ tbe\_nlConfig\ = 0 \\ H_{LP}(z)\ \varepsilon_N\ sign(E_{LB})\ E_{LB}^2 + H_{HP}(z)|E_{LB}|, & if\ tbe\_nlConfig = 0\ \&\&\ idxMixConfig \leq 1 \end{cases}$$

where ε_N is the energy normalization factor between E_{LB} and E_{LB}^2. The above NL generation is for tbe_heMode=0 case. For the case, when tbe_heMode flag=1, the NL generation uses the tbe_nlConfig=1 configuration if idxSubGains is an odd value, and the tbe_nlConfig=0 configuration otherwise. In particular, the energy normalization factor is the ratio of frame energies of E_{LB} and E_{LB}^2. The transfer functions $H_{LP}(z)$ and $H_{HP}(z)$ correspond to low pass and high pass filters with a cut-off frequency 3f$_s$/4. The transfer functions are given as follows:

$$H_{LP}(z) = \frac{0.57(1 + 2z^{-1} + z^{-2})}{1 + 0.94z^{-1} + 0.33z^{-2}}, H_{HP}(z) = \frac{0.098\ (1 - 2z^{-1} + z^{-2})}{1 + 0.94z^{-1} + 0.33z^{-2}}$$

5.5.x.5.4 High band LP estimation

5.5.x.5.4.1 General

The de-quantized LSF parameters are first converted to LSP.

```
for (i = 0; i < 10; i++) {
  hbLsp_curr[i] = (float) cos(qLsf[i] * 2PI);
}
```

For smoother evolution of LP polynomial, the LSPs from current frame, hbLsp_curr, are interpolated with LSPs from previous frame over four sub-frames as shown in the pseudocode below. If the previous frame is not a TBE frame (i.e., indicated using first_frame==1), then interpolation is not performed.

```
lspInterCoeff[8] = { 0.7, 0.3, 0.5, 0.5, 0.3, 0.7, 0.1, 0.9 };

if (!first_frame) {
  copyVector(memory->hbLsp_prevmem, hbLsp_prev, 10);
} else {
  copyVector(hbLsp_curr, hbLsp_prev, 10);
}

ptrLspInterpCoef = lspInterCoeff;
for( j = 0; j < 4; j++ ) {
  for( i = 0; i < 10; i++ ) {
    lsp_temp[i] = hbLsp_prev[i] * (*ptrLspInterpCoef)
                + hbLsp_curr[i] * (*(ptrLspInterpCoef+1));
  }

  ptrLspInterpCoef += 2;

  libLsp2A( lsp_temp, lpcShb+j*(11), 10 );
  lpcShb[j*11] = 1.0;
}

copyVector(hbLsp_curr, memory->hbLsp_prevmem, 10);
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Subsequent to interpolation, the LSPs are converted to LP coefficients as described in ISO/IEC 23003-3:2012 subclause 7.13.11 considering an order of 10.

5.5.x.5.4.2 High band excitation generation

The harmonically-extended excitation, E_{HE}, from subclause **5.5.x.5.3.3 Harmonic Extension** is used as input to the high band excitation generation.

5.5.x.5.4.3 Spectral flip

The harmonic excitation, E_{HE}, is spectrally flipped so that the high band portion of the excitation is modulated down to the low frequency region. This spectral flip is accomplished in time domain as follows:

$$E_{HE}^{f}(n) = (-1)^n \, E_{HE}(n), \qquad n = 0,1,2 \ldots N-1$$

where $N = 512$ is the number of samples per frame.

```
for(i = 0; i < 512; i++) {
  Ef_HE[i] = ((i%2) == 0) ? (-E_HE[i]) : (E_HE[i]);
}
```

5.5.x.5.4.4 Decimation

The spectrally-flipped harmonic excitation, E_{HE}^{f}, is then decimated using a pair of all-pass filters to obtain an downsampled excitation signal, E_{DS}. This is done by filtering the even samples of, $E_{HE}^{f}(n)$, by an all-pass filter whose transfer function is given by:

$$H_{AP1}(z) = \left(\frac{a_{0,1} + z^{-1}}{1 + a_{0,1}z^{-1}}\right)\left(\frac{a_{1,1} + z^{-1}}{1 + a_{1,1}z^{-1}}\right)\left(\frac{a_{2,1} + z^{-1}}{1 + a_{2,1}z^{-1}}\right)$$

And the odd samples of $E_{HE}^{f}(n)$ are filtered using an all-pass filter whose transfer function is given by

$$H_{AP2}(z) = \left(\frac{a_{0,2} + z^{-1}}{1 + a_{0,2}z^{-1}}\right)\left(\frac{a_{1,2} + z^{-1}}{1 + a_{1,2}z^{-1}}\right)\left(\frac{a_{2,2} + z^{-1}}{1 + a_{2,2}z^{-1}}\right)$$

The excitation signal, E_{DS}, is estimated by averaging the outputs of the above two filters, $H_{AP1}(z)$ and $H_{AP2}(z)$. The filter coefficients are specified in Table **AMD2.16**.

Table AMD2.16 — All-pass filter coefficients for decimation

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

	All pass filter coefficients
$a_{0,1}$	0.06056541924291
$a_{1,1}$	0.42943401549235
$a_{2,1}$	0.80873048306552
$a_{0,2}$	0.22063024829630
$a_{1,2}$	0.63593943961708
$a_{2,2}$	0.94151583095682

5.5.x.5.4.5 Adaptive spectral whitening

Due to the nonlinear processing applied to obtain the excitation signal, E_{DS}, the spectrum of this excitation is no longer flat. In order to flatten the spectrum of the excitation signal, a fourth-order LP whitening is applied to E_{DS}. The excitation signal is windowed using the Hanning-based window $win_{flatten}$ as given in Annex X.7 prior to calculation of the autocorrelation coefficients. The autocorrelation of the windowed excitation signal is estimated as follows.

$$E_{DS}(n) = E_{DS}(n) \cdot win_{flatten}(n), \qquad n = 0, 1, \ldots, \left(\frac{N}{2} - 1\right), N = 256$$

$$E_{DS}\left(n + \frac{N}{2}\right) = E_{DS}\left(\mathrm{n} + \frac{N}{2}\right) \cdot win_{flatten}\left(\frac{N}{2} - 1 - n\right), n = 0, 1, \ldots, \left(\frac{N}{2} - 1\right), N = 256$$

$$r_{exc}(k) = \sum_{n=0}^{N-1-k} E_{DS}(n) \cdot E_{DS}(n + k), k = 0, 1, \ldots, 4; N = 256$$

A bandwidth expansion is applied to the autocorrelation coefficients by multiplying the coefficients by an expansion function. The expansion function is as given below:

$$r_{exc}(k) = r_{exc}(k) * BWexpCoef(k), \;\; k = 0,1,2,3,4$$

Table AMD2.17 —Bandwidth expansion coefficients

	BW expansion coefficients
BWexpCoef[5]	`[1.000030000, 0.999876638, 0.999506642, 0.998890286, 0.998028026]`

The bandwidth expanded autocorrelation coefficients are used to obtain the LPC using the Levinson-Durbin algorithm. Inverse LP filtering is performed to obtain the whitened excitation, E_{DS_w}. In the tbe_hrConfig==1 mode, the whitened excitation is further modulated by the normalized residual energy (based on idxResSubGains):

$$E_{DS\_w}(n) = E_{DS\_w}(n) * resSubGain\left(\frac{n}{64}\right), \;\; k = \frac{n}{64} = 0,1,2,3, \;\; n = 0,1,..255$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In the tbe_hrConfig != 1 mode, the whitened excitation is filtered using an FIR filter that is derived from the idxShbExcResp payload. A FIR filter, $H_{resp}(z)$ is constructed by concatenating the coefficients of tbeExcFilter1 and tbeExcFilter2. The whitened excitation, E_{DS_w}, is filtered using the FIR filter $H_{resp}(z)$.

5.5.x.5.4.6 Envelope modulated noise mixing

The whitened harmonic excitation is further modified by adding a random noise whose amplitude is modulated according to the envelope of the whitened excitation, E_{DS_w}. Th e pseu do-random nois e, E_N, is f ur the r perceptually shaped using the high band perceptually-weighted filter (PWF = A(z/γ_1)/A(z/γ_2) using $\gamma_1 = 0.55$ *and* $\gamma_2 = 0.7$. The pseu do-random nois e can be g ene rated usin g th e rand om nois e gene rator util ity function TBE_genRandVec( )as described with the following pseudo code:

```
TBE_genRandVec(){
  k1 = (TBE_randomSign(seed1, idx1) < 0) ? -563.154 : 563.154;
  k2 = (TBE_randomSign(seed2, idx2) < 0) ? -225.261 : 225.261;
    for (i = 0; i < length; i++, idx1++, idx2++) {
      idx1 &= 0x00ff;
      idx2 &= 0x00ff;
      output[i] = k1 * RVEC[idx1] + k2 * RVEC[idx2];
  }
}

TBE_randomSign(seed, idx) {
  seed = (short)(seed * 31821L + 13849L);  /* random number generator */
  idx = ABS((short)((float)seed * 0.0078f));  /* ABS is the absolute operator */
  return (seed < 0? -1: 1);
}
```

where seed1 and seed2 are initialized to 23 and 59 respectively, if the previous frame was not a TBE frame.

The ratio at which the whitened excitation and the envelope-modulated noise, E_{EM_N}, are mixed is dependent on how strongly-voiced the speech segment is. For example, the envelope-modulated noise and the spectrally whitened excitation are mixed as follows to estimate the high band excitation:

$$E_{HB}(n) = \sqrt{VF_i}\left(E_{DS_W}(n)\right) + \sqrt{P1/P2(1 - VF_i)}\ \left(E_{EM_N}(n)\right),\quad n = 0,1,2\ldots,127$$

where VF_i are the voice factors derived from LB as explained below, P1 and P2 are the power estimated from spectrally whitened excitation E_{DS_W} and the envelope modulated noise, E_{EM_N}. In particular, given that the fine signal structure in the higher bands is closely related to that in the lower band, the mixing ratio may be estimated from the low band core ACELP parameters. For example, the voicing from low band is normalized to a smaller scale, i.e., $\alpha_i = 0.95 * \alpha_i - 0.05$ and then mapped to a voice factor as follows. For each subframe, i, the normalized correlation, α_i, from the low band is mapped to a voice factor parameter, VF_i

$$VF_i = \frac{1}{1 + e^{-4\alpha_i}},\quad i = 1,2,3,4$$

Next the voice factors, VF_i, are limited to the range of [0, 1]. The voice factors undergo further smoothing to compensate for any sudden variations in the low band voicing within a frame. For the HR configuration, the voicing factors are modified based on the idxMixConfig to compensate for any mismatch between the LB and HB voicing. Next the envelope-modulated noise is power normalized such that it is at the same level as that of the harmonic excitation (as shown in the equation to estimate E_{HB} above). At each sub-frame, i, the harmonic excitation that is scaled by the factor, VF_i, and the normalized modulated noise that is scaled by the factor $(1 - VF_i)$ are mixed to generate the high band excitation.

5.5.x.5.4.7 High band synthesis

The high band excitation is then passed through the high band LP sub-frame synthesis filters to obtain the spectrally shaped excitation. In the tbe_hrConfig==1 mode, first a memory-less synthesis is performed (with

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

past LP filter memories set to zero) and the energy of the synthesized high band is matched to that of the target signal energy (based on idxShbFrGain). In the subsequent step, the scaled or energy compensated excitation signal is used to perform synthesis to obtain the spectrally shaped excitation, S_{HB}. The spectrally shaped high band signal is then scaled using the decoded gain shapes. The gain shape scaled highband signal is finally multiplied by the decoded gain frame to obtain the gain adjusted high band synthesized signal. The gain shapes and gain frame are applied on the synthesized high band, S_{HB}, as follows:

$$S'_{HB}(n) = GF\sum_{j=0}^{3} w(n-j64)gs(j)S_{HB}(n), n = 0,1,2 \ldots 271$$

where $gs(j)$ j=0,1,2,3 are the gain shapes, GF is the gain frame, and the window w(n) is defined as follows,

Table AMD2.17a — SHB gain shape synthesis window, w(n)

w(n), n = 0,1,2..15	0.007312931, 0.033416940, 0.077119580, 0.136556101, 0.209438491, 0.293364150, 0.385224703, 0.481317588, 0.577845599, 0.671497772, 0.758927143, 0.836651580, 0.901448444, 0.951083203, 0.984171147, 1.000000000
w(n), n=16, 17,..63.	1.0
w(n), n=64,65, 79	0.984171147, 0.951083203, 0.901448444, 0.836651580, 0.758927143, 0.671497772, 0.577845599, 0.481317588, 0.385224703, 0.293364150, 0.209438491, 0.136556101, 0.077119580, 0.033416940, 0.007312931, 0
w(n), for other n	0

5.5.x.5.4.8 Resampling of HB synthesis

The gain adjusted HB synthesis is up-sampled by 2 and flipped (i.e., flip from low to high band, as described in Section **5.5.x.5.4.3**) to generate a high band component associated with the final decoded speech as shown in Figure AMD2.6. The upsampling is based on an all-pass filter as shown below:

$$H_{APInterp,1}(z) = \left(\frac{b_{0,1}+z^{-1}}{1+b_{0,1}z^{-1}}\right)\left(\frac{b_{1,1}+z^{-1}}{1+b_{1,1}z^{-1}}\right)\left(\frac{b_{2,1}+z^{-1}}{1+b_{2,1}z^{-1}}\right)$$

and

$$H_{APInterp,2}(z) = \left(\frac{b_{0,2}+z^{-1}}{1+b_{0,2}z^{-1}}\right)\left(\frac{b_{1,2}+z^{-1}}{1+b_{1,2}z^{-1}}\right)\left(\frac{b_{2,2}+z^{-1}}{1+b_{2,2}z^{-1}}\right)$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table AMD2.18: All-pass filter coefficients for interpolation by a factor of 2

	All pass coefficients
$b_{0,1}$	0.06056541924291
$b_{1,1}$	0.42943401549235
$b_{2,1}$	0.80873048306552
$b_{0,2}$	0.22063024829630
$b_{1,2}$	0.63593943961708
$b_{2,2}$	0.94151583095682

The low band is up-sampled to the same sampling rate as the high band and mixed with the high band component.

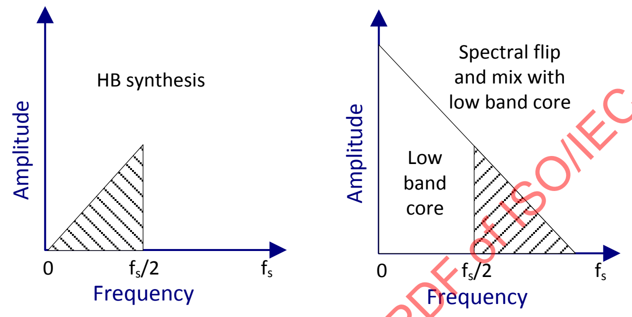


Figure AMD2.6 — Spectral flip of HB synthesis

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

5.5.x.5.4.9 Switching transition signal generation

When switching from ACELP to TCX core, and thus from TBE to IGF, or vice versa, the transition of the high-band signals is performed implicitly by the cross-fade transition mechanism of the core signals as described in subclause **5.5.x.11. Coding mode switching**. To fill the gap caused by the TBE delay (see subclause **5.5.x.5.4.10 Temporal alignment and mixing of HB and ACELP synthesis**) and provide overlapping signals for cross-fading, a transition signal syn_{trans} is generated as follows.

To obtain a continuous high band signal to the previous frame, the overlap portion of the high band synthesized signal syn_{scaled} is used, as described in subclause **5.5.x.5.4.7 High band synthesis**. This overlap portion $syn_{overlap}$ is upsampled using the same filter $H_{I,1}$ and $H_{I,2}$ and flipped subsequently as described in **5.5.x.5.4.8 Resampling of HB synthesis**, resulting in the upsampled high band signal $syn_{overlap,up}$.

The targeted length of syn_{trans} is given by

$$N_{syn,trans} = FrameLength/2 + del_{TDres} - N_{align} = 74\ samples$$

where $N_{align} = 212\ samples$ and $FrameLength/2$ are explained in subclause **5.5.x.5.4.10 Temporal alignment and mixing of HB and ACELP synthesis** and $del_{TDres} = 30\ samples$ is the delay of the TD resampler, which is used for cross-fading, see subclauses **5.5.x.5 TD resampler** and **5.5.x.11 Coding mode switching**.

© ISO/IEC 2017 – All rights reserved

To obtain the 74 samples, the 32 samples of $syn_{overlap,up}$ are extrapolated using the temporally mirrored end $syn_{prev,mirror}$ of the high band synthesized signal of the previous frame syn_{prev}, where

$$syn_{prev,mirror}(n) = syn_{prev}(512 + 1 - n); \quad for\ n = 1, \ldots, 74$$

The signals $syn_{overlap,up}$ and $syn_{prev,mirror}$ are merged to generate syn_{trans} by overlap and add using the window win_{trans} as given in Table AMDX. XX, as

$$syn_{trans}(n) = \begin{cases} win_{trans}(33-n) \cdot syn_{overlap,up}(n) + win_{trans}(n) \cdot syn_{prev,mirror}(n); & for\ n = 1, \ldots, 32 \\ syn_{prev,mirror}(n); & for\ n = 33, \ldots, 74 \end{cases}$$

Table AMDX.XX: Window for generation of transition signal

n	*win*$_{trans}$(n)	n	*win*$_{trans}$(n)	n	*win*$_{trans}$(n)
1	0.000000000	12	0.250082925	23	0.758927143
2	0.002057320	13	0.293364150	24	0.799044170
3	0.007312931	14	0.338401147	25	0.836651580
4	0.017743483	15	0.385224703	26	0.870644917
5	0.033416940	16	0.432854509	27	0.901448444
6	0.053210367	17	0.481317588	28	0.927953529
7	0.077119580	18	0.529597392	29	0.951083203
8	0.104915089	19	0.577845599	30	0.969773527
9	0.136556101	20	0.625078725	31	0.984171147
10	0.171354547	21	0.671497772	32	0.993264992
11	0.209438491	22	0.716039678	33	1.000000000

5.5.x.5.4.10 Temporal alignment and mixing of HB and ACELP synthesis

Due to the overlap of half a TCX frame length (256 samples at $f_{s,TCX}$), the ACELP synthesis ($synth_{ACELP,orig}$) is delayed by half a frame length, to be synchronized to TCX frames in case of mode switching, before being played out by the LPD module. To compensate for this delay and the intrinsic delay of the TBE, the TBE HB synthesis needs to be temporally aligned, before mixing both synthesis signals.

The intrinsic TBE delay is composed of the internal gain shape look-ahead which corresponds to $del_{GainShape} = 16\ samples$ at internal sampling frequency (i.e. downsampled domain) as described in subclause **5.5.x.5.4.7 High band synthesis** and the look-ahead of the harmonic extension which corresponds to $del_{NL} = 12\ samples$ at fullband sampling frequency as described in subclause **5.5.x.5.3.3 Harmonic Extension**. Thus, the entire TBE delay is given by

$$del_{TBE} = 2 \cdot del_{GainShape} + del_{NL} = 44\ samples$$

at fullband sampling frequency. The TBE HB synthesis ($synth_{TBE,orig}$) must be delayed by

$$N_{align} = FrameLength/2 - del_{TBE} = 212\ samples$$

at fullband sampling frequency to obtain a TBE HB synthesis signal ($synth_{TBE,align}$) aligned with the synchronized ACELP synthesis ($synth_{ACELP,sync}$).

By adding the TBE HB synthesis signal and the synchronized ACELP synthesis, the fullband output signal is obtain, given by

$$sig_{FB}(n) = synth_{ACELP,sync} + synth_{TBE,align}(n); \quad for\ n = 1, \ldots, FrameLength$$

Add new normative Annex containing the Codebook tables used to de-quantize high band TBE parameters

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Annex X

(normative)

Codebook tables used to de-quantize high band Time Domain Bandwidth Extension parameters

X.1 Codebook Table for tbeLSFCB1_7b

```
tbeLSFCB1_7b[128x10] =
```

0.04864	0.097476	0.14699	0.19811	0.25074	0.32976	0.36053	0.38823	0.42603	0.46205
0.078258	0.11627	0.15307	0.2182	0.26242	0.29803	0.34483	0.379	0.41075	0.43858
0.048724	0.099256	0.15056	0.20264	0.26402	0.34633	0.37125	0.40096	0.43365	0.46697
0.055234	0.09483	0.13869	0.18	0.22943	0.27291	0.31784	0.36125	0.41255	0.45768
0.066955	0.11182	0.15389	0.19033	0.22682	0.26791	0.34946	0.38512	0.42342	0.46221
0.14499	0.1748	0.21495	0.24694	0.28636	0.31766	0.35518	0.39053	0.4296	0.46383
0.053712	0.093501	0.14021	0.18292	0.2327	0.27472	0.32114	0.36421	0.4071	0.45041
0.066898	0.11273	0.16787	0.20781	0.24622	0.28312	0.32727	0.37084	0.40917	0.45845
0.082957	0.12982	0.17424	0.21286	0.24847	0.27936	0.31235	0.34285	0.37285	0.40878
0.050453	0.087648	0.14344	0.18596	0.23016	0.27458	0.32064	0.36275	0.41255	0.4558
0.051327	0.098449	0.15099	0.19838	0.24737	0.2934	0.33846	0.38268	0.42641	0.46512
0.05992	0.098888	0.14664	0.19564	0.24687	0.2953	0.34024	0.37436	0.4127	0.46074
0.064031	0.10319	0.14764	0.19739	0.28187	0.31927	0.3543	0.39228	0.4299	0.46405
0.05498	0.10968	0.16474	0.21945	0.27381	0.32726	0.37935	0.4227	0.44562	0.46469
0.062601	0.099835	0.1538	0.20029	0.2468	0.31243	0.34783	0.37527	0.4072	0.44755
0.074256	0.12069	0.16331	0.20117	0.24158	0.2817	0.32371	0.3614	0.3963	0.43522
0.062513	0.093121	0.14336	0.21052	0.249	0.28973	0.32993	0.3629	0.39997	0.43306
0.025216	0.043257	0.13114	0.17096	0.22462	0.26778	0.31784	0.36167	0.41032	0.45643
0.056889	0.093219	0.1666	0.21076	0.25352	0.29067	0.32129	0.34992	0.37958	0.41368
0.056254	0.095794	0.14664	0.19128	0.2395	0.2838	0.32869	0.36962	0.41141	0.45676
0.058042	0.097978	0.15892	0.19464	0.23194	0.2935	0.32952	0.35889	0.40646	0.46064
0.072914	0.1154	0.15946	0.20074	0.2373	0.26943	0.30625	0.33507	0.36892	0.45515
0.056726	0.097439	0.14751	0.19242	0.24069	0.28525	0.33185	0.37655	0.42171	0.46229
0.057682	0.099927	0.15032	0.19308	0.27116	0.30672	0.33711	0.36787	0.39748	0.42756
0.059562	0.091735	0.1341	0.17118	0.21142	0.24863	0.29472	0.35438	0.41296	0.4628
0.076592	0.11714	0.16126	0.19883	0.26108	0.2992	0.3282	0.36076	0.40563	0.46238
0.081288	0.13113	0.18743	0.23964	0.28014	0.311	0.34508	0.38208	0.41473	0.44881
0.047304	0.091933	0.14034	0.18718	0.23665	0.28615	0.34071	0.37923	0.41879	0.45996
0.073207	0.11332	0.15612	0.19594	0.23554	0.27743	0.32333	0.36004	0.40608	0.4578
0.044135	0.090759	0.13707	0.17933	0.23073	0.27435	0.32164	0.36583	0.41077	0.45869
0.061503	0.099996	0.1528	0.20125	0.23896	0.27336	0.33323	0.37198	0.4063	0.46365
0.065593	0.10711	0.16066	0.20394	0.24858	0.29493	0.33799	0.37688	0.41802	0.46047
0.1059	0.15367	0.18936	0.22616	0.26279	0.28926	0.3175	0.35417	0.40672	0.45639
0.057113	0.089617	0.12791	0.16286	0.21899	0.29084	0.35171	0.38775	0.42141	0.45997
0.053929	0.092518	0.13694	0.1872	0.23374	0.27702	0.32099	0.36716	0.41372	0.45724
0.077925	0.13348	0.17623	0.21205	0.24834	0.28115	0.32721	0.38949	0.43806	0.4682
0.060914	0.096059	0.14495	0.19642	0.26599	0.30276	0.33096	0.36561	0.428	0.46364
0.05613	0.085034	0.13865	0.22254	0.26806	0.30525	0.34751	0.38395	0.41763	0.45029
0.064602	0.10449	0.14634	0.18852	0.23804	0.2827	0.32807	0.3707	0.41459	0.45941
0.058853	0.13554	0.23094	0.26372	0.28659	0.31606	0.34768	0.38237	0.41999	0.459

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

0.05125	0.089764	0.17961	0.22403	0.2602	0.29763	0.34402	0.37597	0.40581	0.42963
0.050346	0.092015	0.14275	0.18846	0.23657	0.28188	0.33198	0.37714	0.43739	0.47361
0.090389	0.13874	0.18458	0.22703	0.29216	0.32995	0.3617	0.3935	0.43168	0.46435
0.062889	0.10177	0.15493	0.19945	0.25233	0.29641	0.32735	0.35234	0.38123	0.45544
0.033993	0.10547	0.16782	0.19104	0.21646	0.2844	0.2997	0.3822	0.42213	0.4468
0.048248	0.091496	0.13816	0.18174	0.22617	0.27066	0.31573	0.36917	0.41195	0.45331
0.074263	0.11464	0.15112	0.19072	0.24866	0.3139	0.3564	0.39229	0.42873	0.45959
0.082441	0.12775	0.17752	0.21919	0.26654	0.31239	0.35872	0.39247	0.42209	0.45135
0.06041	0.099001	0.1455	0.18432	0.22311	0.25934	0.30034	0.339	0.38092	0.42745
0.098797	0.15514	0.19949	0.23451	0.27097	0.30414	0.34261	0.38031	0.41939	0.45784
0.050761	0.10016	0.15511	0.20496	0.28976	0.35413	0.36998	0.40295	0.43271	0.46698
0.06635	0.10749	0.15417	0.20038	0.24819	0.29304	0.33287	0.36759	0.40764	0.45054
0.069487	0.12266	0.1735	0.22063	0.26539	0.3029	0.34334	0.38586	0.42719	0.4649
0.042813	0.091028	0.13746	0.18144	0.22793	0.27297	0.31582	0.36022	0.40902	0.45234
0.052359	0.093423	0.14274	0.18587	0.23586	0.28088	0.32701	0.37263	0.41803	0.46095
0.070718	0.11038	0.14953	0.18253	0.21624	0.26049	0.32113	0.3672	0.41231	0.45706
0.097556	0.12551	0.1721	0.2097	0.25496	0.29517	0.33522	0.37297	0.41498	0.45778
0.064392	0.10609	0.15365	0.19776	0.24462	0.28831	0.33764	0.3811	0.4242	0.46355
0.073186	0.12058	0.16666	0.21015	0.25381	0.29592	0.3378	0.3776	0.4159	0.4579
0.091626	0.11604	0.15985	0.19528	0.24234	0.28384	0.32961	0.37044	0.41356	0.45836
0.057081	0.095632	0.14173	0.18279	0.22838	0.28121	0.32571	0.36764	0.41234	0.45558
0.054602	0.093851	0.14172	0.18519	0.23076	0.27183	0.32759	0.36943	0.41137	0.45868
0.092356	0.14423	0.18907	0.23602	0.2752	0.30097	0.32923	0.3666	0.43299	0.47154
0.020284	0.067152	0.1739	0.19172	0.22009	0.2837	0.29809	0.38033	0.42259	0.44582
0.11677	0.17269	0.20934	0.24439	0.27788	0.31092	0.34639	0.38046	0.42032	0.45872
0.043648	0.096081	0.15832	0.21033	0.26148	0.30301	0.35086	0.39177	0.4316	0.4681
0.086671	0.12893	0.16953	0.21426	0.28242	0.30587	0.34046	0.36881	0.40498	0.4608
0.05893	0.10354	0.1605	0.21739	0.26171	0.29623	0.33495	0.37639	0.42006	0.46177
0.058451	0.10759	0.16289	0.21735	0.27477	0.31757	0.35144	0.37944	0.40977	0.45046
0.13913	0.16306	0.20118	0.22876	0.26581	0.2942	0.33234	0.36722	0.40766	0.44982
0.064312	0.10151	0.14697	0.18944	0.23353	0.27556	0.32105	0.35401	0.39019	0.43029
0.066387	0.1139	0.16282	0.1981	0.23374	0.26839	0.31431	0.37408	0.42376	0.46234
0.0619	0.10224	0.15369	0.1959	0.23929	0.28086	0.32217	0.36659	0.40559	0.44546
0.057058	0.088951	0.12949	0.17176	0.26222	0.30829	0.34607	0.38467	0.42381	0.45859
0.049937	0.094674	0.14351	0.19269	0.24155	0.30233	0.35801	0.38773	0.42523	0.4617
0.096405	0.12953	0.17659	0.21641	0.26497	0.30357	0.34612	0.38552	0.42775	0.46475
0.069274	0.12394	0.18428	0.22001	0.26097	0.29437	0.32684	0.35971	0.392	0.426
0.07173	0.11545	0.16149	0.2073	0.25003	0.29488	0.34333	0.38752	0.43048	0.46536
0.025897	0.054595	0.13747	0.18048	0.23646	0.27848	0.33024	0.37767	0.4258	0.46747
0.062263	0.10497	0.17325	0.2122	0.24606	0.27739	0.31206	0.35079	0.39653	0.44881
0.080856	0.1174	0.15965	0.20324	0.24789	0.29084	0.33559	0.3797	0.42412	0.46362
0.076273	0.1152	0.15864	0.19365	0.23081	0.26163	0.29772	0.34594	0.40989	0.45981
0.05974	0.097397	0.14663	0.18948	0.23014	0.26658	0.31188	0.35323	0.39598	0.45568
0.089729	0.12058	0.1664	0.20761	0.25415	0.29671	0.34251	0.38499	0.42675	0.46417
0.10149	0.15478	0.20253	0.24714	0.28616	0.32199	0.35857	0.39359	0.43249	0.46541
0.068134	0.10965	0.16946	0.21258	0.25223	0.29539	0.32954	0.36661	0.40406	0.44434
0.059349	0.099026	0.14925	0.20131	0.2487	0.28452	0.3177	0.35592	0.41204	0.45923
0.065178	0.10281	0.14506	0.19004	0.23703	0.28249	0.34337	0.37726	0.40877	0.44441
0.058248	0.1002	0.14662	0.19279	0.24515	0.29091	0.34616	0.40158	0.4391	0.46714
0.055435	0.10999	0.16531	0.22018	0.27454	0.32777	0.37764	0.40848	0.42345	0.4524
0.094702	0.11922	0.16528	0.20083	0.24884	0.28901	0.33374	0.37321	0.4154	0.45969
0.065859	0.11153	0.16175	0.19725	0.2667	0.2972	0.3296	0.39005	0.41908	0.44839
0.068448	0.12152	0.17471	0.22659	0.27381	0.31568	0.35792	0.39513	0.43529	0.46713
0.03068	0.069976	0.14897	0.19635	0.2515	0.29365	0.34592	0.39015	0.43145	0.46917
0.0625	0.10732	0.16161	0.22149	0.27042	0.30207	0.32751	0.35529	0.3954	0.44926
0.073237	0.11547	0.15621	0.20002	0.24426	0.28741	0.33136	0.37513	0.41916	0.46057
0.072788	0.11042	0.14969	0.172	0.28589	0.29948	0.3576	0.37192	0.43892	0.46426
0.060735	0.1061	0.15458	0.20272	0.25444	0.3012	0.34733	0.38491	0.42641	0.46586
0.061916	0.10486	0.16011	0.20586	0.24633	0.28864	0.32991	0.36585	0.43974	0.46804
0.04931	0.07895	0.14	0.19322	0.2453	0.29023	0.32821	0.35684	0.39085	0.45545

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

0.071672	0.1113	0.17002	0.20712	0.28395	0.3044	0.35077	0.37205	0.43674	0.46667
0.05397	0.08745	0.13262	0.1732	0.21461	0.25479	0.33363	0.37275	0.4197	0.45941
0.047444	0.0917	0.13765	0.18167	0.22432	0.27282	0.31944	0.36299	0.40694	0.45558
0.056411	0.10184	0.17531	0.20875	0.24522	0.27875	0.33244	0.38745	0.42799	0.46691
0.061397	0.1026	0.15555	0.19894	0.24395	0.28636	0.32936	0.37276	0.41652	0.45946
0.067086	0.10131	0.1415	0.17384	0.26202	0.29564	0.32718	0.35501	0.4041	0.45262
0.06752	0.10817	0.14602	0.18037	0.23122	0.29473	0.33742	0.37734	0.42715	0.46179
0.056339	0.097771	0.15026	0.19271	0.23301	0.2775	0.32642	0.37017	0.41986	0.46119
0.078862	0.1313	0.17098	0.21256	0.25355	0.31264	0.37304	0.40151	0.44291	0.47174
0.057228	0.087987	0.13248	0.17724	0.22664	0.29933	0.33536	0.36461	0.3966	0.4468
0.071459	0.1196	0.17572	0.21851	0.25563	0.28666	0.31849	0.35939	0.42121	0.46212
0.084105	0.11778	0.1555	0.18993	0.23727	0.29898	0.33703	0.36576	0.39955	0.44998
0.059197	0.11086	0.16253	0.21338	0.26115	0.30845	0.35535	0.39323	0.43047	0.46691
0.081157	0.14149	0.19367	0.23526	0.27272	0.31008	0.35214	0.39291	0.43546	0.46809
0.059251	0.10234	0.18619	0.23621	0.27276	0.30402	0.33806	0.37437	0.42053	0.46013
0.057636	0.11668	0.17332	0.21085	0.25074	0.28787	0.35466	0.38723	0.41556	0.45072
0.077062	0.11305	0.16977	0.21412	0.25478	0.31383	0.34803	0.37685	0.41875	0.45695
0.079259	0.12789	0.17309	0.21644	0.25272	0.28551	0.31874	0.34827	0.38158	0.45512
0.057262	0.09683	0.14579	0.18805	0.23294	0.27595	0.31965	0.366	0.4105	0.45759
0.050687	0.083651	0.16807	0.20719	0.26005	0.29982	0.33047	0.36335	0.40416	0.45711
0.074028	0.12257	0.16312	0.20115	0.24125	0.28368	0.36765	0.39595	0.43793	0.4697
0.077292	0.13171	0.18313	0.223	0.26128	0.29794	0.33419	0.37304	0.41163	0.45407
0.094223	0.12981	0.17293	0.20371	0.23887	0.27876	0.32559	0.36707	0.41258	0.45899
0.050304	0.095738	0.15217	0.19473	0.24594	0.28693	0.33484	0.38731	0.41925	0.44441
0.045328	0.086714	0.13202	0.1828	0.22775	0.27262	0.31975	0.36471	0.40814	0.45356
0.066427	0.10038	0.1371	0.17293	0.22457	0.27187	0.31852	0.38646	0.4289	0.46181
0.037658	0.091131	0.15119	0.19697	0.25015	0.28876	0.31699	0.374	0.41923	0.45872
0.05842	0.088421	0.12705	0.16466	0.2148	0.26252	0.3135	0.35408	0.39721	0.4461

X.2 Codebook Table for tbeLSFCB2_7b

```
tbeLSFCB2_7b[128x10] =
```

-0.00337	-0.00510	-0.00245	0.00060	-0.00334	-0.00731	0.00799	0.00068	-0.00497	0.01280
0.01295	0.00199	-0.01603	0.01614	0.00589	-0.00575	-0.00171	0.00060	-0.00092	-0.00469
-0.00043	0.00719	0.01071	0.00272	-0.00114	-0.01095	0.01105	0.00183	-0.00772	0.00764
-0.00320	-0.00066	0.00754	0.00244	0.01423	0.00461	0.00266	0.00977	0.00397	0.00409
-0.00900	-0.02064	0.01180	0.00291	-0.00054	-0.00549	-0.00008	-0.00686	-0.00678	-0.00002
-0.00646	0.00064	-0.00090	0.00330	-0.00139	0.00387	-0.00365	-0.00194	-0.00167	0.00174
-0.00310	0.00348	-0.00210	-0.01048	0.00088	0.01234	0.00902	-0.00303	-0.00937	0.01209
-0.00744	0.00015	-0.00688	0.00403	-0.00814	0.00404	-0.00731	0.00788	-0.00504	0.01323
0.00135	-0.01165	-0.00957	-0.00186	0.00873	0.00330	-0.00063	0.00013	0.00098	0.00650
0.00628	-0.00531	0.00132	0.01057	0.00084	-0.00405	0.00899	0.00717	-0.00353	-0.00357
-0.00445	-0.00915	-0.01136	-0.01307	-0.00679	0.00019	0.00610	0.00643	0.00774	0.00522
-0.00245	-0.01237	0.00349	-0.01072	0.00668	-0.00514	0.00465	-0.00753	0.01282	0.00145
0.00020	0.00098	-0.00067	-0.00022	-0.00024	0.00072	0.00156	0.00702	0.00783	0.00300
0.00009	-0.00134	-0.00210	0.00217	-0.00712	-0.00067	-0.00289	0.00313	0.00023	-0.00204
0.00376	-0.00171	-0.00462	-0.00358	-0.00237	-0.00759	-0.00798	-0.00179	-0.00928	-0.01537
-0.00045	-0.00215	0.00164	-0.00024	0.00582	0.00815	-0.00066	-0.00713	0.00088	-0.00162
-0.00161	-0.00855	0.00341	-0.00380	-0.00987	0.00499	-0.00495	-0.00986	-0.01351	0.00954
-0.00035	0.00520	-0.00068	-0.00133	0.01598	0.00197	-0.00185	0.00165	-0.00270	-0.01326
-0.00053	0.00002	-0.00364	-0.00144	-0.00916	-0.01694	0.00265	0.00646	0.00567	0.00078
0.00631	0.01285	-0.00311	-0.01465	0.00403	0.00264	-0.00369	0.00078	0.00634	0.00285
0.00665	-0.00361	-0.00854	0.00962	-0.00654	0.01019	0.00158	0.00435	0.00877	-0.00357
0.00338	-0.00083	-0.00835	-0.01444	-0.00405	-0.00840	-0.00385	-0.00190	-0.00825	0.00515
-0.00335	-0.00132	-0.00304	-0.00521	-0.00570	-0.00721	-0.00872	-0.00590	0.00970	0.01296
0.00213	-0.00811	-0.01250	0.01025	-0.00119	-0.00865	0.00550	-0.00397	0.00942	0.00233

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

0.00781	0.00977	0.00334	-0.00673	-0.01296	0.00435	0.00111	-0.00431	-0.00840	0.00809
0.00281	-0.00295	-0.00111	-0.00497	-0.00256	0.00023	-0.00136	-0.00411	-0.00139	-0.00247
-0.00973	-0.00705	0.00334	0.00307	-0.00091	-0.00496	-0.00195	0.00378	0.00523	0.00387
-0.00329	0.00277	0.00016	-0.00236	0.00125	-0.00644	0.00166	-0.00244	0.00023	-0.00216
-0.00146	0.00552	0.00388	-0.00039	0.00409	0.00209	-0.00137	0.00087	-0.00120	0.00082
0.00462	0.00036	0.00365	-0.00233	0.00624	0.00022	-0.00780	0.00998	-0.00181	0.01111
0.00166	0.00000	-0.00128	0.00356	0.00562	-0.00056	0.00400	-0.00137	-0.00074	0.00172
0.00059	-0.00876	-0.01031	0.01259	0.00462	-0.00022	-0.00520	-0.01173	-0.00854	0.00461
-0.00623	0.00871	0.00137	-0.01140	0.00488	-0.00649	0.00912	0.00697	0.00018	0.00161
-0.01810	-0.00014	0.00146	-0.00191	0.00653	0.00256	0.00293	-0.00032	-0.00211	-0.00089
-0.00041	-0.00486	-0.00147	-0.00212	-0.00127	-0.00410	0.00450	0.00219	0.00191	0.00042
-0.00626	-0.00094	0.00891	0.00596	0.00246	-0.00645	-0.01149	0.00095	-0.00966	0.00508
0.00439	-0.00186	-0.01025	-0.00088	-0.00483	0.00934	-0.00057	-0.01341	0.00599	0.00660
-0.00401	-0.01532	-0.00893	-0.00157	-0.00542	-0.00426	-0.00606	-0.00341	-0.00204	-0.00048
-0.00810	-0.01429	0.00390	0.00399	-0.00012	0.01031	0.01085	0.00579	0.00179	0.00282
-0.00261	-0.00601	0.00025	-0.00432	-0.00828	0.01165	0.00455	0.00070	-0.00620	-0.01734
-0.00077	-0.00906	0.00763	-0.00175	-0.01536	0.00454	-0.00274	0.01174	0.00388	-0.00443
-0.00699	0.01617	0.00363	-0.00455	-0.01085	-0.00888	-0.00345	-0.00409	-0.00176	-0.00098
0.00237	0.00694	0.00619	-0.00269	-0.00619	-0.00513	-0.00662	0.01945	0.00982	0.00334
-0.00945	0.00895	0.00229	-0.01320	0.00256	0.00168	-0.01106	0.00550	-0.00322	-0.00463
-0.00490	-0.00743	0.00205	-0.00081	-0.00575	-0.01266	0.01381	0.00730	-0.00233	-0.01681
-0.00141	0.00044	-0.00047	-0.00114	0.00259	0.00345	0.00363	0.00523	0.01856	0.01402
-0.00473	0.00005	0.01646	0.00678	0.00100	0.00647	0.00089	-0.00544	-0.00524	0.01047
-0.00077	0.00165	-0.00063	-0.00045	-0.00052	-0.00255	-0.00528	-0.00695	-0.00521	-0.00145
0.00213	0.00388	0.01048	-0.00187	-0.00761	0.00328	-0.00815	-0.00023	0.00088	-0.00450
0.01530	0.00935	0.00755	0.00303	0.00692	0.00512	0.00375	0.00463	0.00297	-0.00083
0.00130	-0.00101	0.00009	0.00100	0.00256	0.00324	0.00385	0.00622	0.00144	-0.00456
-0.00063	-0.00036	-0.00035	-0.00429	-0.00255	-0.00276	-0.00544	0.00083	0.00099	0.00295
-0.00998	0.01045	-0.00535	0.00361	0.00112	-0.00385	-0.00531	0.00469	0.00449	-0.00224
0.00004	0.00077	0.00790	-0.00007	-0.00948	-0.00132	0.00307	0.00034	0.00531	0.00734
0.00297	-0.00669	0.01277	0.00930	0.00400	0.00377	-0.00322	-0.00198	-0.00435	-0.01245
-0.00091	0.00325	0.00161	0.01818	0.01222	0.00835	0.00426	-0.00130	-0.00496	-0.00217
-0.00556	-0.00922	0.00384	0.01075	0.00487	-0.00245	-0.01137	-0.01353	0.01339	0.00754
0.00071	0.00125	0.00059	0.00104	0.00027	0.00204	0.00071	-0.00041	-0.00574	-0.00585
0.00217	0.00300	-0.00770	-0.00164	0.00010	0.00118	-0.00011	-0.00002	-0.00053	-0.00012
-0.00568	0.00131	-0.00318	0.01111	0.00041	-0.00721	0.00253	-0.00190	-0.00423	-0.00706
-0.00211	-0.00271	0.00269	-0.00181	-0.01218	-0.00419	0.00225	-0.00293	-0.00585	-0.00265
0.00179	0.00409	0.00189	0.00557	0.00192	-0.00087	-0.00680	0.01192	-0.00009	-0.01131
-0.00260	0.00019	0.00065	-0.00016	0.00166	0.00369	0.00057	-0.00867	-0.01672	-0.00577
0.00074	-0.00313	-0.00577	-0.01117	0.01268	0.00983	0.01024	0.00849	0.00353	-0.00299
-0.00414	-0.00339	-0.00184	-0.00462	0.00169	0.00135	0.00080	-0.00307	-0.00328	0.00437
-0.00414	0.00332	0.00388	0.00865	0.00266	0.00152	0.00417	-0.00481	0.01127	-0.00190
0.00014	0.00054	-0.00267	-0.00129	0.00974	-0.00129	-0.01645	-0.00023	0.00216	-0.00094
0.00344	-0.00024	0.00063	0.00090	0.00010	0.00140	-0.00278	-0.00298	-0.00558	0.00500
0.00680	-0.00135	0.00742	0.00697	-0.00743	0.00847	0.00614	-0.00265	-0.00976	0.00082
-0.00268	-0.00037	-0.00132	0.00002	0.00274	0.00038	-0.00121	0.00105	0.00223	-0.00086
-0.00155	-0.00403	-0.00214	-0.00630	0.00164	-0.00012	0.00727	0.00100	-0.00553	-0.00712
-0.00109	-0.00513	0.00426	-0.00198	-0.00064	0.00316	-0.00233	0.00407	-0.00220	-0.00041
-0.00094	-0.00156	0.00007	-0.00060	0.00234	0.00101	-0.00263	-0.00229	0.00595	0.00684
0.00249	-0.00715	0.00663	-0.00735	0.00972	-0.00423	0.00415	0.00210	-0.01681	0.00579
-0.00452	-0.00344	-0.00482	-0.00148	-0.00085	0.00147	-0.00314	-0.00339	0.00554	-0.00666
-0.00783	0.01143	0.00487	-0.00730	0.00410	0.00412	-0.00284	-0.01383	0.00927	0.00531
-0.00070	0.00038	-0.00354	0.00283	-0.00263	-0.00456	-0.00060	-0.00363	0.00131	0.00302
0.00484	0.00422	0.00167	-0.00090	-0.00187	-0.00403	0.00003	0.00316	-0.00169	-0.00139
-0.01608	0.00351	0.00713	-0.00532	-0.01154	0.00641	0.00285	-0.00096	0.00387	0.00155
-0.00274	-0.00600	0.00146	0.00442	0.00154	-0.00044	-0.00156	-0.00220	-0.00295	-0.00348
0.00451	0.00129	-0.00598	-0.00924	-0.00267	-0.00035	-0.00035	0.01025	0.00224	-0.00936
-0.00333	0.01928	0.01314	0.00532	0.00299	-0.00150	-0.00236	0.00373	0.00259	-0.00183
-0.00353	-0.00932	0.00003	0.01754	0.00853	-0.00008	-0.00624	0.00566	0.00827	0.00118
0.00217	-0.00935	0.01083	0.00067	-0.01143	0.01088	0.00003	-0.01172	0.01042	0.00396

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

-0.01660	0.00488	0.01388	0.00339	-0.00212	-0.00667	-0.00086	-0.00172	-0.00523	-0.01345
0.00673	0.00086	-0.00182	0.00715	0.00340	0.00729	0.00690	-0.00234	0.00200	0.01147
0.00663	0.01129	0.00470	-0.00276	-0.00958	-0.00489	0.00915	0.00326	-0.00505	-0.01288
0.00587	-0.00019	-0.00082	0.00326	-0.00003	-0.00385	0.00133	-0.00406	0.00256	-0.00591
0.01532	0.00977	-0.00174	0.00304	-0.00344	-0.00652	-0.01245	0.00256	0.00588	0.00005
0.01009	-0.00216	-0.00350	-0.00358	-0.00498	-0.00098	0.00291	0.00284	0.00285	0.00460
-0.00508	0.01327	-0.00038	0.00304	0.00459	-0.00144	0.00127	-0.00718	-0.00963	0.00960
0.00808	0.00285	-0.00735	0.00592	-0.00489	-0.00934	0.00689	-0.00499	-0.01021	0.00396
0.00517	0.00012	0.00051	0.00232	0.00084	0.00387	-0.00405	0.00114	0.00068	-0.00005
0.00588	0.00418	-0.01025	0.00294	0.01323	0.00363	-0.00389	-0.00421	0.01118	0.00219
-0.00067	-0.00674	0.00169	0.00331	-0.00314	-0.01072	-0.01872	0.01094	0.00269	-0.00552
0.00903	0.00325	0.00558	0.00257	0.00204	-0.00219	-0.00570	-0.01377	0.00330	0.00650
0.00284	0.01149	-0.00301	-0.00822	0.00067	0.00254	-0.00309	-0.01205	-0.00010	-0.01318
0.00884	0.00161	-0.00642	-0.00693	-0.01250	-0.00702	-0.00683	-0.01022	0.00435	-0.00224
0.00513	-0.00615	-0.01373	-0.00169	0.00414	-0.00529	-0.00926	0.01457	0.00922	0.00224
-0.00062	0.00049	0.00232	0.00591	-0.00044	0.00009	0.00059	0.00328	-0.00021	0.00418
0.00189	-0.00047	0.00472	0.00164	0.00287	-0.00387	-0.00465	0.00070	0.00565	-0.00168
0.00454	0.00316	-0.00233	0.01149	0.00352	-0.00822	-0.00614	0.00157	0.00051	0.00736
-0.00788	0.00067	-0.00855	0.00507	0.00335	0.00380	0.00738	0.00339	0.00149	0.00360
0.01501	0.00202	-0.00822	-0.00462	0.00285	0.00705	-0.00018	-0.00441	-0.00545	-0.00465
0.00489	0.01066	-0.00362	-0.00124	-0.00135	-0.00669	0.00770	-0.00296	0.01064	0.00596
0.00093	-0.00225	0.00609	0.00067	0.00030	-0.00290	0.00251	-0.00254	-0.00151	0.00109
-0.00207	-0.00532	-0.01473	-0.01715	0.00824	0.00273	-0.00514	-0.01033	-0.00059	-0.00534
0.00232	0.00251	0.00162	-0.00421	0.00149	0.00162	0.00399	-0.00223	0.00552	-0.00128
0.00526	-0.00233	0.01093	0.00588	-0.00702	-0.01361	-0.00334	-0.00177	0.00054	-0.00199
0.01063	0.00542	-0.01294	-0.00590	0.01045	-0.00500	0.01123	0.00424	-0.00410	0.00156
0.00132	0.00630	0.00032	0.00503	-0.00245	0.00182	0.00076	-0.00323	0.00108	-0.00113
-0.00675	-0.00372	0.00315	-0.00018	-0.00370	-0.00204	-0.01295	-0.01879	-0.00088	-0.00739
-0.00443	0.00812	0.00509	0.00141	0.00209	0.01110	0.01058	0.00327	-0.00349	-0.00476
0.00236	0.00203	0.00071	-0.00213	-0.00041	0.00399	0.00620	0.00326	-0.00395	0.00331
0.01170	0.01308	0.00544	0.00387	0.00234	-0.00131	-0.00686	-0.00899	-0.01449	-0.00552
-0.00455	0.00381	-0.00150	-0.00425	-0.00407	0.00142	0.00145	0.00011	0.00065	-0.00036
0.00142	-0.00377	-0.01061	0.00145	-0.00680	0.00356	0.00786	-0.00003	-0.00785	-0.00015
0.00031	-0.00046	-0.00130	0.00012	-0.00511	-0.00415	0.01774	0.01511	0.00634	0.00103
0.00216	0.01149	0.00207	-0.00597	-0.01697	0.01133	0.00409	0.00307	0.00625	-0.00341
-0.00185	-0.00120	-0.00286	-0.00065	0.00175	-0.00332	-0.00206	0.00458	-0.00494	0.00032
0.00419	-0.00080	0.00042	-0.00553	0.00667	-0.00079	-0.00148	-0.00047	-0.00114	-0.00043
-0.00415	-0.01096	-0.01411	0.00782	0.00911	0.00301	0.00108	0.00301	-0.00459	-0.01355
-0.00702	-0.01663	0.00665	-0.00800	0.00952	0.00021	-0.00605	0.00804	0.00040	-0.00767
-0.00070	-0.00255	0.00008	0.00094	-0.00347	0.00505	0.00399	-0.00214	0.00319	-0.00064
-0.00519	0.00129	0.00444	0.00199	-0.00295	-0.00070	0.00338	0.00526	0.00066	-0.00398
-0.00127	0.00067	-0.00361	-0.00203	0.01636	0.00656	-0.00244	-0.00530	-0.00926	0.00465
-0.00224	-0.00755	0.01921	0.00929	0.00078	-0.00545	0.00948	0.00684	0.00558	-0.00054
0.00061	-0.00183	-0.00318	-0.00807	0.00005	0.01402	-0.00302	0.00284	0.00492	0.00030

X.3 Codebook Table for tbeExcFilterCB1_7b

```
tbeExcFilterCB1_7b[128x10] =
```

0.09278	-0.04927	-0.1285	0.04224	0.19906	-0.11443	-0.17523	0.22302	0.13816	-0.28557
0.00011	-0.11068	-0.30536	-0.28212	-0.08996	-0.23144	-0.02353	0.01956	0.08163	0.15772
-0.0627	-0.05202	-0.09499	0.09656	0.09493	-0.07314	0.26034	0.12844	0.08331	-0.13683
0.07668	-0.02219	0.29456	0.17219	0.1365	-0.02633	0.02025	-0.0703	-0.0775	-0.36374
0.07061	-0.08168	-0.02593	-0.1945	-0.1438	0.01913	0.1581	0.13042	0.18332	0.03253
-0.04266	0.05446	0.04855	0.21232	0.34886	-0.04787	-0.14493	0.01559	0.20742	0.02468
0.29098	0.06383	0.1662	0.15719	0.14352	0.19807	0.09548	-0.05796	0.10414	-0.13189

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

-0.11786	-0.08787	0.21436	0.22915	-0.07895	-0.16376	-0.24403	-0.12381	0.00463	0.18062
0.19711	-0.20348	0.03503	-0.24203	0.1465	0.01923	-0.25212	0.07296	-0.13447	0.06531
-0.20131	0.09759	-0.08098	-0.04882	-0.1088	-0.08122	0.14914	-0.10339	0.24513	-0.19795
-0.02871	0.24279	0.05725	0.07613	0.41804	0.19693	-0.01821	0.02656	-0.18997	-0.10614
-0.08732	-0.00701	0.29408	-0.07048	-0.21163	0.21397	0.09863	-0.24706	0.16062	-0.04952
-0.10538	-0.086	0.174	0.18545	-0.10238	0.14518	0.24527	-0.11161	-0.0553	-0.04299
-0.25925	-0.21043	0.1905	0.21659	-0.08269	-0.15606	0.20488	0.1939	-0.09604	-0.07044
0.0885	-0.01086	-0.01367	-0.06393	0.25061	-0.31779	0.27489	-0.04085	-0.00347	0.00237
0.13072	0.15224	0.11707	0.17535	0.1665	0.14625	0.1913	0.15279	0.17186	0.1516
0.13914	0.13684	-0.11358	0.00496	0.19564	-0.11009	-0.05058	0.10552	-0.0155	-0.16635
-0.17303	0.30944	-0.16285	-0.12656	0.08875	0.01211	-0.22898	0.21767	-0.01427	0.01698
0.07809	-0.09757	0.16756	-0.11494	0.23395	-0.19899	0.18347	-0.1323	0.04415	0.00773
0.04403	-0.20143	-0.07105	-0.07592	-0.06155	-0.08754	0.09564	0.16468	0.30223	0.18032
-0.05453	-0.09441	0.07789	0.00663	-0.14208	0.21035	-0.11962	0.26739	-0.19667	0.23282
0.08878	0.06016	-0.03346	-0.10331	0.03731	-0.18329	0.12412	-0.15104	-0.10267	0.17931
0.13982	0.16001	0.10645	-0.09327	-0.15898	-0.17531	-0.24699	-0.12977	-0.03232	0.10249
0.17369	-0.22144	0.05599	0.04504	-0.19557	0.20916	-0.18563	0.05508	0.11578	-0.19359
0.11344	-0.14198	0.17441	-0.24905	0.07004	-0.21457	-0.06794	0.12604	-0.01816	-0.11586
-0.28752	0.01574	-0.05318	-0.16036	-0.0289	0.33425	-0.2483	-0.03269	0.23887	-0.0867
-0.20272	0.11303	0.01556	-0.09528	0.12476	-0.06551	0.18028	-0.15544	0.13792	-0.09292
0.32678	-0.34258	0.28105	-0.16207	0.15352	-0.04954	0.10858	-0.16012	0.13098	-0.18598
0.28576	-0.13167	-0.05888	0.41716	0.15788	-0.0209	0.05537	0.05957	-0.11597	-0.14661
-0.17519	-0.20084	-0.34181	-0.14418	0.04831	0.10175	0.14236	0.29364	0.19556	0.13918
-0.01579	-0.17416	0.11269	-0.07107	0.06136	0.12149	0.38782	0.13322	0.17157	0.14254
0.60481	-0.16148	-0.15186	0.01878	-0.06836	-0.1074	-0.00005	-0.05971	0.05391	-0.02553
-0.10493	-0.14194	0.38099	-0.2901	0.22174	-0.09632	0.06112	0.02876	-0.02286	0.07233
0.24364	-0.21663	-0.01911	0.0096	-0.11763	-0.24622	0.06394	0.28675	-0.21145	-0.06453
-0.01379	0.17444	-0.27516	0.24924	-0.32361	0.20715	-0.16383	0.12124	0.00005	-0.04997
-0.11352	0.00839	0.2271	-0.09127	-0.06364	0.32364	-0.00851	-0.10246	0.07668	0.1547
0.22745	0.42787	0.08653	0.20847	-0.02817	-0.01357	-0.23705	-0.11736	-0.08144	-0.11707
-0.06226	0.07085	-0.28589	-0.14047	-0.13638	0.0236	0.10832	0.14287	-0.098	-0.16612
0.2042	0.14631	-0.08956	0.14235	-0.096	0.04706	-0.11978	-0.18188	0.05349	-0.11259
-0.09417	0.15147	-0.09651	-0.14957	-0.00021	0.05522	-0.12538	0.01834	0.25829	0.12862
0.16162	-0.06564	0.0067	0.18043	-0.06129	0.03649	0.03095	0.02585	-0.02912	0.06451
-0.28975	-0.345	0.01568	0.24021	0.18663	0.21627	0.01571	0.03578	0.11493	0.08829
-0.12705	0.04453	-0.13477	0.091	-0.01383	0.00159	-0.06496	-0.05362	-0.0394	-0.02151
-0.12755	0.14558	0.15106	-0.02064	-0.29887	0.16319	0.06857	0.0212	-0.2026	0.0259
-0.17117	-0.06328	-0.03136	-0.29652	-0.1893	-0.09838	0.0278	0.09643	0.10744	0.02357
0.053	0.18895	0.16667	-0.00416	0.1834	0.07217	0.19257	-0.04169	0.12714	-0.01619
-0.0752	-0.14532	-0.07175	0.0769	0.24361	0.11323	-0.12563	-0.23458	-0.08666	0.05998
-0.05859	0.04079	0.18985	0.09698	-0.19533	-0.24503	-0.00954	0.12981	0.21652	0.07806
-0.12002	0.28904	0.01626	-0.24498	-0.0272	0.2128	-0.02151	-0.16347	-0.0967	0.20795
0.50418	0.08574	0.22471	-0.04306	0.13474	-0.08671	-0.05666	-0.03676	-0.08459	-0.06769
0.02286	0.12102	0.29062	0.12411	0.08985	-0.16048	-0.12923	-0.07103	0.1419	0.0479
-0.10845	-0.02732	0.03072	0.0004	-0.12867	0.03644	-0.08572	-0.11518	-0.06929	-0.19148
-0.02748	-0.17042	0.00163	0.22631	0.0142	-0.11956	-0.05735	0.21573	0.14477	-0.12457
-0.01568	0.11073	-0.03985	0.04661	0.05849	-0.15079	-0.10682	0.05741	-0.18892	-0.14861
0.19846	0.08849	-0.18109	0.07191	0.11123	0.11135	-0.22693	-0.13385	0.14867	0.2448
-0.06086	0.07105	0.25679	0.21256	0.15673	0.00614	-0.14847	-0.27092	-0.16918	-0.08032
0.12908	-0.03723	0.03004	-0.20515	-0.02586	-0.09931	0.064	-0.34729	-0.24735	-0.01626
0.03513	-0.15752	-0.21218	-0.01513	0.10926	0.26235	0.13124	-0.05397	-0.12931	-0.164
0.32126	-0.43877	0.29727	-0.06329	-0.01772	-0.02293	0.0103	0.01445	-0.11286	0.15418
-0.01192	-0.07969	-0.12374	-0.17855	-0.23807	0.00419	-0.02184	0.06684	0.17852	0.29893
0.10326	0.19906	0.19395	0.15637	0.1055	0.11203	0.17639	-0.15675	-0.11169	-0.08789
-0.00384	-0.20679	0.14551	-0.33019	-0.05396	-0.1172	-0.03656	-0.19987	0.31038	-0.07446
-0.06342	-0.0733	0.17837	-0.02925	-0.3374	-0.07286	0.06349	0.08309	-0.04134	-0.18
0.18227	0.01978	-0.13399	0.13816	-0.07183	-0.09984	0.19717	-0.38523	0.24609	-0.14978
-0.15414	0.11149	-0.03078	0.18451	-0.00297	0.07744	-0.19722	0.13131	-0.32475	-0.17452
0.1621	0.19069	-0.01805	-0.19237	-0.31834	-0.15308	-0.24636	-0.04185	-0.09914	0.15125
0.03948	-0.06088	0.02126	0.10788	0.09653	0.02821	0.15783	0.31781	-0.08155	-0.0866

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

-0.0929	-0.12616	0.15252	-0.0054	-0.13055	-0.06022	0.32783	-0.01737	-0.14561	0.16336
0.14679	-0.56493	-0.10632	0.04228	-0.05807	0.04268	0.09777	-0.03118	0.08026	-0.02373
0.11382	-0.10601	0.11797	-0.15816	0.18875	-0.24262	0.22059	-0.2419	0.16014	-0.16597
-0.24762	-0.1385	-0.0306	0.01652	0.2478	0.00461	0.04798	-0.13205	-0.17872	-0.27586
0.17452	-0.07422	-0.18426	-0.22858	0.17663	0.17637	0.01567	0.04839	0.17465	0.04414
0.21046	0.20266	-0.06857	0.13902	0.09125	0.15126	0.00066	0.25658	-0.11583	0.09393
0.09252	-0.15663	0.25566	-0.29825	0.20961	0.02225	-0.07936	0.18718	-0.0963	0.05259
-0.00489	0.30066	0.2113	-0.02953	-0.13884	-0.01595	0.07737	0.24748	0.00024	-0.24767
0.0832	0.132	-0.32829	0.23837	-0.14868	0.02548	0.02908	-0.06391	0.17978	-0.17895
-0.28483	0.09445	0.03269	-0.28991	0.28999	-0.15826	-0.07588	0.15063	0.02107	-0.09693
0.00589	-0.23991	-0.25147	0.0797	0.00099	-0.33922	-0.00358	0.05907	-0.00584	-0.06714
-0.3141	-0.18801	-0.19762	-0.01455	-0.15881	-0.02694	-0.11556	0.15593	0.03082	0.07944
-0.17182	-0.12576	-0.207	0.01487	-0.3132	0.07076	0.14413	-0.04814	0.07926	0.29518
-0.12083	0.22381	-0.09037	-0.00754	-0.10236	-0.30394	-0.04741	-0.19619	0.0041	-0.02265
-0.135	-0.13959	-0.07177	0.04913	0.1705	0.29018	0.21969	0.12795	-0.08954	-0.1192
-0.02099	-0.01349	0.18666	-0.11731	-0.0694	-0.01206	0.04575	-0.28522	0.06272	0.21929
0.05285	-0.24949	-0.07622	0.00994	0.12521	-0.20135	-0.23649	-0.0102	0.28374	-0.08653
0.02941	-0.11032	0.10028	-0.09118	0.02639	0.03435	-0.09569	0.1835	-0.23319	0.21404
0.24015	0.2126	-0.30559	-0.05484	0.21301	-0.21983	-0.07878	0.03676	-0.08183	-0.06813
-0.19167	0.12845	-0.02812	-0.08852	-0.00023	0.05442	-0.23315	0.25718	-0.17648	0.04466
-0.27951	-0.40209	0.03651	-0.23924	-0.0378	-0.03813	0.146	0.03969	0.01548	0.10721
0.05164	0.3033	0.16359	-0.3031	0.18994	0.09423	-0.17393	-0.10583	0.15986	-0.17391
0.28887	0.212	0.1482	0.07474	-0.0562	0.07817	0.09121	-0.0125	0.0249	0.10721
-0.10861	-0.17472	-0.16074	0.01766	0.13904	-0.01032	-0.18168	-0.07163	0.0866	0.25212
0.06583	0.1732	0.18612	0.02844	-0.00684	0.05076	0.06633	-0.01868	-0.36337	-0.00145
-0.20373	0.17958	0.01276	-0.19623	0.25943	-0.20367	0.04363	0.1202	-0.23678	0.23433
-0.31519	0.45586	-0.19958	0.15573	-0.01105	0.05693	-0.00513	-0.04786	0.04476	-0.07665
0.03913	0.01571	0.12931	-0.04987	-0.03622	0.12222	-0.12841	-0.06675	0.12739	-0.27043
0.19196	0.01555	-0.18623	-0.10338	0.10205	0.24427	-0.05114	-0.19983	-0.06205	0.23172
-0.04826	0.06662	-0.05736	0.11138	0.14769	0.16176	0.18896	0.17452	0.16037	0.16983
-0.58447	0.14411	0.15819	-0.02144	-0.05763	0.03507	0.07015	0.04073	0.00249	0.06873
-0.0266	-0.16384	0.28281	-0.21409	0.01775	0.19542	-0.24404	0.14437	0.01253	-0.14766
-0.15907	0.18067	-0.22955	0.29121	-0.1669	0.04491	0.15167	-0.15487	0.01232	0.11974
-0.10455	-0.17684	0.00579	-0.06413	-0.27576	-0.13528	-0.10478	-0.08135	-0.18969	0.04317
-0.05531	0.07263	0.32632	0.25261	0.13853	0.22197	0.05991	0.10931	-0.02463	-0.03771
-0.20395	-0.07239	-0.09158	0.08176	0.07971	0.00478	0.08317	-0.05277	-0.01755	0.30809
0.08477	0.14641	-0.2379	-0.10917	0.17376	0.00504	-0.17479	-0.04727	0.13627	-0.02987
-0.02589	-0.05581	-0.02698	0.18784	-0.07125	-0.21985	0.14818	-0.04534	-0.01314	-0.27353
0.10683	0.10494	0.03292	0.07779	0.04558	0.19903	-0.1806	-0.25146	-0.218	-0.08735
-0.01506	0.02393	-0.0587	0.18272	-0.04665	0.25365	0.03352	0.07832	0.31578	-0.18382
-0.13469	0.00802	-0.06824	-0.14264	0.09786	0.14645	0.03437	0.25509	0.12065	-0.0047
-0.17372	0.16471	-0.1458	0.20169	-0.19102	0.23891	-0.15672	0.17025	-0.14625	0.17294
-0.15096	-0.13882	-0.24174	-0.21843	-0.16274	-0.04953	-0.15773	-0.11578	0.05515	0.00516
0.17293	-0.16682	-0.18541	0.17506	0.00449	-0.19161	0.13069	-0.01691	-0.16014	0.14568
0.25916	-0.21056	0.2534	0.02887	-0.1987	0.19963	-0.21138	0.09373	-0.1072	-0.05951
0.04409	0.16961	0.14393	0.18203	-0.00318	-0.15294	0.14754	-0.08258	-0.39253	0.05379
-0.29088	0.21829	-0.18293	-0.05882	0.23213	-0.21198	0.2196	-0.17126	0.10365	0.07923
0.17379	0.04236	-0.20095	0.24801	-0.04809	-0.17663	0.2164	-0.1322	-0.05155	0.19464
0.305	-0.09919	-0.29554	0.13626	-0.13423	0.12329	-0.27827	0.21269	-0.14658	0.06335
0.1476	0.19249	0.04762	-0.09385	-0.16885	-0.22736	-0.04003	0.12119	0.21066	0.20016
0.16428	-0.23129	0.09663	0.16263	-0.18679	0.0795	0.0932	-0.21933	0.16027	0.06644
0.33437	0.09924	-0.0745	-0.3394	-0.21713	0.03763	0.16516	0.02988	0.01462	-0.08174
0.05556	0.1579	-0.10683	-0.23347	-0.02877	0.17658	0.2735	-0.04794	-0.19388	-0.06425
0.19943	0.08925	0.13219	-0.00961	-0.05766	-0.26293	-0.24305	-0.23162	-0.16636	-0.25138
0.27138	-0.13496	-0.01964	0.30839	-0.36682	0.11144	0.11409	-0.07192	-0.13701	0.09785
-0.23936	-0.16826	0.22517	-0.12862	0.14949	0.15623	0.08161	-0.08912	0.05373	-0.14484
0.01871	0.10768	0.03986	0.20721	0.07845	-0.16446	-0.10902	0.16222	-0.18383	0.00369
0.21345	-0.22268	0.07646	0.1023	-0.22599	0.20325	-0.10195	-0.10515	0.20697	-0.13575

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

-0.04869	-0.07865	-0.12637	-0.00702	-0.06357	-0.11804	-0.19936	-0.23968	-0.09776	0.0579
-0.0791	0.0253	-0.0595	0.08575	0.01864	0.07607	-0.15311	0.29688	-0.13386	0.16496
-0.0944	0.25161	-0.35154	0.16061	0.06074	-0.1722	0.1042	0.15929	-0.18229	0.19977

X.4 Codebook Table for tbeExcFilterCB2_4b

```
tbeExcFilterCB2_4b[16x6] =
```

-0.20232	0.19847	-0.096441	0.20558	0.1712	0.14251
0.18595	-0.059879	0.22704	0.098489	-0.21216	0.1781
0.14201	0.13805	0.18921	0.10948	0.21574	-0.19181
-0.15192	-0.20135	-0.16245	0.15589	-0.20001	0.15736
0.11894	-0.22244	0.10315	0.20449	0.19699	0.16757
-0.18169	-0.1969	0.21968	-0.17568	0.08214	-0.073996
-0.17077	0.094474	-0.016721	-0.21172	-0.22251	0.20939
0.17044	0.1361	-0.21435	0.22042	-0.11153	-0.1802
0.10768	-0.21295	-0.17889	-0.2272	0.16925	0.12479
0.19545	0.18475	0.06574	-0.22264	-0.15453	-0.17849
0.15459	-0.24222	-0.0084513	-0.081542	-0.17888	-0.22024
-0.19043	0.14675	-0.20002	-0.19699	0.063592	-0.17998
0.019109	0.1948	0.21405	-0.15812	0.19539	0.19374
-0.18862	0.13162	0.18746	0.17667	-0.21274	-0.12737
0.2333	0.18153	-0.21589	-0.0058148	0.043154	0.1992
-0.11754	-0.19539	-0.15312	0.1694	0.18711	-0.20856

X.5 Codebook Table for SHBCB_SubGain5bit

```
SHBCB_SubGain5bit[32x4] =
```

0.21132, 0.8919, 0.25073, 0.24781, 0.49109, 0.36892, 0.36424, 0.68769, 0.31642, 0.39282, 0.50357, 0.69143, 0.14538, 0.20545, 0.61508, 0.72523, 0.55597, 0.40693, 0.5515, 0.4636, 0.32782, 0.35346, 0.71621, 0.48607, 0.47823, 0.75732, 0.28454, 0.30591, 0.93541, 0.20099, 0.17145, 0.15627, 0.46125, 0.40196, 0.53871, 0.57318, 0.81196, 0.33227, 0.32905, 0.32359, 0.27682, 0.63987, 0.274, 0.64008, 0.55515, 0.50701, 0.48201, 0.44507, 0.47378, 0.47841, 0.60912, 0.41031, 0.66645, 0.45733, 0.47489, 0.33369, 0.28321, 0.30804, 0.31331, 0.83754, 0.72424, 0.24507, 0.26089, 0.56661, 0.50327, 0.28639, 0.75435, 0.26554, 0.55363, 0.61005, 0.4572, 0.31771, 0.11637, 0.11819, 0.15457, 0.96163, 0.49218, 0.58968, 0.40155, 0.49174, 0.33193, 0.70435, 0.4566, 0.40953, 0.41897, 0.52221, 0.45016, 0.58571, 0.30084, 0.55935, 0.69881, 0.28529, 0.35628, 0.51005, 0.57418, 0.52403, 0.19211, 0.20635, 0.90755, 0.24599, 0.7371, 0.23927, 0.56338, 0.24211, 0.64261, 0.38355, 0.435, 0.49054, 0.7583, 0.53387, 0.27663, 0.20498, 0.53563, 0.46946, 0.44187, 0.54153, 0.47413, 0.49108, 0.51251, 0.50223, 0.6421, 0.52116, 0.34328, 0.43333, 0.4512, 0.57399, 0.51208, 0.44725

X.6 Codebook Table for SHBCB_GainFrame5bit

```
SHBCB_GainFrame5bit[32] =
```

0.001011, 0.003713, 0.015517, 0.060132, 0.120586, 0.195489, 0.281113, 0.379025, 0.492374, 0.620740, 0.764047, 0.924080, 1.098318, 1.290460, 1.498303, 1.724956, 1.972520, 2.246851, 2.565094, 2.935901, 3.376749, 3.924642, 4.601801, 5.425421, 6.469817, 7.949088, 10.28801, 13.60010, 15.98213, 26.88470, 52.42506, 104.3400

X.7 Codebook Table for win_flatten

```
win_flatten[128] =
```

0.000088034, 0.000476116, 0.001169363, 0.002150485, 0.003429115, 0.005010492, 0.006898746, 0.009062795, 0.011521152, 0.014280317, 0.017346705, 0.020672203, 0.024285616, 0.028196360, 0.032413006, 0.036865973, 0.041598847, 0.046624415, 0.051953217, 0.057488784, 0.063295004, 0.069388114, 0.075780325, 0.082343259, 0.089166444, 0.096269593, 0.103666288, 0.111191524, 0.118965557, 0.127011578, 0.135344218, 0.143757482, 0.152407144, 0.161319848, 0.170510936, 0.179729455, 0.189171151, 0.198866047, 0.208829871, 0.218763164, 0.228905716, 0.239290834, 0.249934286, 0.260485032, 0.271230554, 0.282207315, 0.293430784, 0.304495752, 0.315740588, 0.327204751, 0.338903073, 0.350374111, 0.362009827, 0.373852485, 0.385915952, 0.397681021, 0.409595440, 0.421704067, 0.434019475, 0.445963724, 0.458042003, 0.470301524, 0.482753259, 0.494760120,

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

0.506885848, 0.519179745, 0.531650889, 0.543603194, 0.555659506, 0.567870932, 0.580244380, 0.592025487,
0.603896179, 0.615909077, 0.628068660, 0.639563560, 0.651134210, 0.662834422, 0.674666017, 0.685762445,
0.696921498, 0.708197860, 0.719590484, 0.730179986, 0.740819828, 0.751565232, 0.762412100, 0.772391076,
0.782409064, 0.792521481, 0.802721036, 0.811991726, 0.821291165, 0.830674628, 0.840131507, 0.848602932,
0.857094007, 0.865659523, 0.874285468, 0.881874298, 0.889474928, 0.897141336, 0.904856044, 0.911487394,
0.918124024, 0.924818762, 0.931550652, 0.937158805, 0.942767102, 0.948426913, 0.954113807, 0.958642835,
0.963168312, 0.967739839, 0.972329565, 0.975733870, 0.979132398, 0.982572706, 0.986023587, 0.988268182,
0.990506116, 0.992782769, 0.995064351, 0.996124740, 0.997178979, 0.998271140, 0.999365251, 0.999935699

X.8 Codebook Table for RVEC

```
RVEC[256] =
```

0.02164473, 0.35885197, -0.16274954, -0.08241354, 0.07313631, -0.00054929, -0.13080014, 0.07226136,
-0.13965981, -0.04834007, -0.02745908, -0.02867859, 0.11216793, 0.16604294, -0.00134274, 0.06818508,
-0.17387933, 0.09406016, -0.08150196, 0.05083200, -0.01952806, -0.10203217, -0.03067050, -0.05153965,
0.06250680, 0.00859049, -0.12008808, -0.11361376, 0.17176038, 0.01174004, -0.02275130, -0.09895785,
-0.10167463, -0.22059087, -0.05334539, -0.00629700, -0.16706355, 0.07795000, 0.08731710, 0.09669208,
0.15378080, 0.01794813, -0.01549965, -0.24923822, 0.19985947, -0.10477958, 0.06674605, -0.11186616,
-0.17927034, 0.08443811, 0.25542912, 0.03167623, 0.19633667, 0.19163096, 0.01907267, 0.12298489,
-0.03147158, 0.05562247, 0.30200079, -0.04257871, 0.08275045, -0.03386311, -0.02265750, 0.18742503,
-0.13598505, -0.32004824, -0.00438390, -0.15576170, 0.06006401, -0.00952147, 0.18848655, 0.06630960,
0.07121546, -0.00733249, 0.08277771, 0.22764891, 0.06772452, -0.09509693, -0.00172236, 0.08452052,
0.17020901, -0.03737585, 0.02349647, 0.10855560, 0.06854416, 0.07084806, 0.09390105, 0.00124924,
0.03026483, -0.15169589, 0.01347072, -0.15377805, 0.14992996, 0.11630810, 0.03483583, -0.03914850,
-0.20075595, 0.12728901, -0.04495851, -0.11576717, -0.15281813, 0.06055827, -0.03471978, -0.03617816,
0.17230885, 0.03094525, -0.15618153, 0.21792564, 0.08106838, -0.22098514, -0.10796417, 0.07131225,
0.22092983, -0.01539366, -0.02876964, -0.30910203, 0.02143815, -0.11630868, -0.00922897, 0.07431208,
0.15533504, 0.11425125, 0.07125455, -0.11914105, -0.04275274, -0.05072749, -0.22143129, 0.19787727,
-0.20946717, -0.16564523, 0.05962536, -0.22325630, -0.04333350, -0.04707248, 0.16608582, 0.00948954,
0.11283893, -0.04097161, -0.09076904, 0.26722300, 0.00987607, -0.05807892, 0.07872546, 0.08040629,
0.12927419, -0.05647410, 0.09603068, -0.02356448, -0.02160797, -0.11687102, 0.07936122, -0.05764586,
-0.10510305, -0.02326054, 0.12021790, 0.09782617, -0.22600858, -0.02555378, -0.03561033, -0.01337216,
0.11311363, -0.03096960, -0.22801498, 0.05643769, 0.13053033, 0.04452197, -0.09299882, -0.11475921,
0.02257649, -0.21770498, -0.11454470, -0.09435777, 0.00638951, -0.36990553, 0.04266735, 0.06915011,
0.07644624, -0.24336053, -0.03421960, -0.10622191, -0.17223521, 0.04054553, 0.13831380, 0.02925055,
0.16207848, -0.12994884, -0.09751288, -0.05397306, -0.09323815, 0.13425350, -0.00046960, 0.31072289,
0.13740718, 0.05835414, -0.04803475, 0.15423043, -0.09652353, 0.14896898, -0.16368309, 0.05875925,
-0.03678078, -0.19627908, 0.07034992, -0.27213186, -0.04338680, 0.01567988, -0.09158870, 0.11987700,
0.07083926, 0.01099900, -0.01084446, 0.04508050, -0.10655984, -0.13945042, 0.05837287, 0.08458713,
-0.04212087, -0.15749574, 0.11632511, 0.07976698, 0.06725866, -0.09567240, 0.03796997, -0.09355708,
-0.13569611, -0.19498724, 0.14951572, -0.16023041, 0.04185898, 0.06099325, 0.03425207, 0.16211477,
0.03998571, -0.03629408, -0.10099959, 0.19540504, 0.11653102, 0.23601755, 0.04943547, -0.26040605,
0.02153429, 0.22880882, -0.13646534, 0.03881640, -0.02896636, 0.09774253, -0.13509314, -0.08713179,
0.13485038, 0.06968338, 0.19561967, 0.07884958, -0.10365590, -0.10321335, -0.09081125, -0.00147976

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

6 Joint Channels for Low Bitrate Coding

In Table 10 — Syntax of mpegh3daChannelPairElementConfig() replace:

```
    if(shiftIndex1 > 0) {
        shiftChannel1;                                   nBits1)
    }
}
```

with:

```
    if (shiftIndex1 > 0) {
        shiftChannel1;                                   nBits1)
    }
    if (sbrRatioIndex == 0 && qceIndex == 0) {
        lpdStereoIndex;                                  1          bslbf
    } else {
        lpdStereoIndex = 0
    }
}
```

In Table 32 — Syntax of mpegh3daCoreCoderData () replace:

```
    for (ch=0; ch<nrChannels; ch++) {
        if (core_mode[ch] == 1) {
            lpd_channel_stream(indepFlag);
        }
```

with:

```
    for (ch = 0; ch < nrChannels; ch++) {
        if (core_mode[ch] == 1) {
            if (lpdStereoIndex == 1 && ch == 1 && core_mode[0] == 1) {
                lpd_stereo_stream(indepFlag);
            } else {
                lpd_channel_stream(noiseFilling, fullbandLpd, indepFlag);
            }
        }
```

Add following new table in 5.2.3.2:

Table AMD3.1 — Syntax of lpd_stereo_stream()

Syntax	No. of bits	Mnemonic
lpd_stereo_stream(indepFlag)		
{		
for (l = 0, n = 0; l < ccfl; l += M, n++) {		
res_mode;	**1**	**uimsbf**
q_mode;	**1**	**uimsbf**

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
        ipd_mode;                                               2           uimsbf
        pred_mode;                                              1           uimsbf
        cod_mode;                                               2           uimsbf

        nbands = band_config(N, res_mode);
        ipd_band_max = max_band[res_mode][ipd_mode];
        cod_band_max = max_band[res_mode][cod_mode];
        cod_L = 2*(band_limits[cod_band_max]-1);

        for (k = 1; k >= 0; k--) {
            if (q_mode == 0 || k == 1) {
                for (b = 0; b < nbands; b++) {
                    ild_idx[2n+k][b];                           5           uimsbf
                }
                for (b = 0; b < ipd_band_max; b++) {
                    ipd_idx[2n+k][b];                           3           uimsbf
                }
                if (pred_mode == 1) {
                    for (b = cod_band_max; b < nbands;b++) {
                        pred_gain_idx[2n+k][b];                 3           uimsbf
                    }
                }
            }
        }

        if (cod_mode == 1) {
            cod_gain_idx[2n+k];                                 7           uimsbf
            for (i = 0; i < cod_L/8; i++) {
                code_book_indices(i, 1, 1);
            }
        }
    }
}
```

Add the following new subclause at the end of 5.5:

5.5.x LPD Stereo Coding

5.5.x.1 Tool description

LPD stereo is a discrete M/S stereo coding, where the Mid-channel is coded by the mono LPD core coder and the Side signal coded in the DFT domain. The decoded Mid signal is output from the LPD mono decoder and then processed by the LPD stereo module. The stereo decoding is done in the DFT domain where the L and R channels are decoded. The two decoded channels are transformed back in the Time Domain and can be then combined in this domain with the decoded channels from the FD mode. FD mode uses its own stereo tools, i.e. discrete stereo with or without complex prediction described in ISO/IEC 23003-3:2012, subclause 7.12.

5.5.x.2 Data Elements

lpd_stereo_stream()	Data element to decode the stereo data for the LPD mode
lpdStereoIndex	Flag which indicates if LPD stereo is activated.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

res_mode	Flag which indicates the frequency resolution of the parameter bands.
q_mode	Flag which indicates the time resolution of the parameter bands.
ipd_mode	Bit field which defines the maximum of parameter bands for the IPD parameter.
pred_mode	Flag which indicates if prediction is used.
cod_mode	Bit field which defines the maximum of parameter bands for which the side signal is quantized.
lld_idx[k][b]	ILD parameter index for the frame k and band b.
lpd_idx[k][b]	IPD parameter index for the frame k and band b.
pred_gain_idx[k][b]	Prediction gain index for the frame k and band b.
cod_gain_idx	Global gain index for the quantized side signal.
fullBandLpd	Flag which indicates if LPD mode is in full band mode.

5.5.x.3 Help Elements

ccfl	Core code frame length.
M	Stereo LPD frame length as defined in Table **AMD3.2**.
band_config()	Function that returns the number of coded parameter bands.
band_limits()	Function that returns the number of coded parameter bands
max_band()	Function that returns the number of coded parameter bands
cod_L	Number of DFT lines for the decoded side signal.

5.5.x.4 Decoding Process

5.5.x.4.1 General

The stereo decoding is performed in the frequency domain. It acts as a post-processing of the LPD decoder. It receives from the LPD decoder the synthesis of the mono Mid signal. The Side signal is then predicted and decoded in frequency domain. Left (L) and right (R) channel spectrums are then reconstructed in frequency domain before being resynthesized in time domain. LPD stereo works with a fixed frame size equal to the size of the ACELP frame independently of the coding mode used in LPD mode.

5.5.x.4.2 Frequency analysis

The DFT spectrum of the frame index *i* is computed from the decoded frame *x* of the Mid signal of length *M* as follows:

$$\mathrm{X}_i[\mathrm{k}] = \sum_{n=0}^{N-1} w[n] \cdot x[i \cdot \mathrm{M} + n - L] \cdot e^{-2\pi jkn/N}$$

where *N* is the size of the signal analysis, *w* is the analysis window and *x* the decoded time signal from the LPD decoder at frame index *i* delayed by the overlap size *L* of the DFT. *M* is equal to the size of the ACELP frame at the output sampling rate. The DFT analysis window size *N* is equal to the LPD stereo frame size plus

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

the overlap size of the DFT. The sizes are depending whether the LPD mode is running in full-band mode as defined in Table AMD3.2.

Table AMD3.2 — DFT and frame sizes of the stereo LPD

fullBandLpd	ccfl	DFT size *N*	Frame size *M*	Overlap size *L*
0	1024	336	256	80
1	1024	672	512	160
0	768	256	192	64
1	768	512	384	128

The window w is a sine window defined as:

$$w[n] = \begin{cases} \sin\left(\frac{\pi}{2L}\left(n + \frac{1}{2}\right)\right) & \text{for } 0 \le n < L \\ 1 & \text{for } L \le n < M \\ \sin\left(\frac{\pi}{2L}\left(L - M + n + \frac{1}{2}\right)\right) & \text{for } M \le n < M + L \end{cases}$$

5.5.x.4.3 Configuration of the parameter bands

The DFT spectrum is divided into non-overlapping frequency bands called parameter bands. The partitioning of the spectrum is non-uniform and mimics the auditory frequency decomposition. Two different divisions of the spectrum are possible with bandwidths following roughly either two or four times the Equivalent Rectangular Bandwidths (ERB).

The spectrum partitioning is selected by the data element **res_mode** and defined by the following pseudo-code:

```
function nbands = band_config(N, res_mod)
band_limits[0] = 1;
nbands = 0;
while(band_limits[nbands++] < (N/2)) {
  if (stereo_lpd_res == 0) {
    band_limits[nbands] = band_limits_erb2[nbands];
  } else {
    band_limits[nbands] = band_limits_erb4[nbands];
  }
}
nbands--;
band_limits[nbands] = N/2;
return nbands
```

where *nbands* is the total number of parameter bands and *N* the DFT analysis window size. The tables *band_limits_erb2* and *band_limits_erb4* are defined in Table AMD3.3. The decoder can adaptively change the resolution of the parameter bands at every two LPD stereo frames.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Table AMD3.3 — Parameter band limits in term of DFT index k

Parameter band index *b*	*band_limits_erb2*	*band_limits_erb4*
0	1	1
1	3	3
2	5	7
3	7	13
4	9	21
5	13	33
6	17	49
7	21	73
8	25	105
9	33	177
10	41	241
11	49	337
12	57	
13	73	
14	89	
15	105	
16	137	
17	177	
18	241	
19	337	

The maximum number of parameter bands for IPD is sent within the 2 bits field **ipd_mode** data element:

$$ipd\_max\_band = max\_band[res\_mode][ipd\_mode]$$

The maximum number of parameter bands for the coding of the Side signal is sent within the 2 bits field **cod_mode** data element:

$$cod\_max\_band = max\_band[res\_mode][cod\_mode]$$

The table max_band[ ][ ] is defined in Table **AMD3.4**.

The number of decoded lines to expect for the side signal is then computed as:

$$cod\_L = 2 \cdot (band\_limits[cod\_max\_band] - 1)$$

Table AMD3.4 — Maximum number of bands for different code modes

Mode index	*max_band[0]*	*max_band[1]*
0	0	0
1	7	4
2	9	5
3	11	6

5.5.x.4.4 Inverse quantization of stereo parameters

The stereo parameters Interchannel Level Differences (ILD), Interchannel Phase Differences (IPD) and prediction gains are sent either every frame or every two frames depending of flag **q_mode**. If **q_mode** equal 0, the parameters are updated every frame. Otherwise, the parameters values are only updated for odd indices i of the LPD stereo frame within the USAC frame. The index i of the LPD stereo frame within a USAC frame can be either between 0 and 3 in LPD version 0 and between 0 and 1 in LPD version 1.

The ILD are decoded as follows:

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

$$\mathrm{ILD}_i[\mathrm{b}] = ild\_q[ild\_idx[i][b]\ ]\text{, for } 0 \le b < nbands$$

The IPD are decoded for the ipd_max_band first bands:

$$\mathrm{IPD}_i[\mathrm{b}] = \frac{\pi}{4} \cdot ipd\_idx[i][b]\ - \pi,\ \text{for } 0 \le b < ipd\_max\_band$$

The prediction gains are only decoded if **pred_mode** flag is set to one. The decoded gains are then:

$$pred\_gain_i[\mathrm{b}] = \begin{cases} 0 & \text{, for } 0 \le b < cod\_\max\_band \\ res\_pred\_gain\_q[pred\_gain\_idx[i][b]\ ] & \text{, for } cod\_\max\_band \le b < nbands \end{cases}$$

If the pred_mode is equal to zero, all gains are set to zero.

Independently of the value of **q_mode**, the decoding of the side signal is performed every frame if **code_mode** is a non-zero value. It first decodes a global gain:

$$cod\_gain_i = 10^{cod\_gain\_idx[i]/(20 \cdot \frac{127}{90})}$$

The decoded shape of the Side signal is the output of the AVQ described in ISO/IEC 23003-3:2012, subclause 7.12.

$$S_i[1 + 8k + n] = kv[k][0][n]\text{, for } 0 \le n < 8 \text{ and } 0 \le k < \frac{cod\_L}{8}$$

Table AMD3.5 — Inverse quantization table ild_q[]

index	output	index	Output
0	-50	16	2
1	-45	17	4
2	-40	18	6
3	-35	19	8
4	-30	20	10
5	-25	21	13
6	-22	22	16
7	-19	23	19
8	-16	24	22
9	-13	25	25
10	-10	26	30
11	-8	27	35
12	-6	28	40
13	-4	29	45
14	-2	30	50
15	0	31	reserved

Table AMD3.6 — Inverse quantization table res_pres_gain_q[]

index	output
0	0
1	0.1170
2	0.2270
3	0.3407
4	0.4645
5	0.6051
6	0.7763
7	1

5.5.x.4.5 Inverse channel mapping

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

The Mid signal X and Side signal S are first converted to the left and right channels L and R as follows:

$$L_i[\mathrm{k}] = X_i[k] + gX_i[k], \text{ for } band\_limits[b] \leq k < band\_limits[b+1],$$
$$R_i[\mathrm{k}] = X_i[k] - gX_i[k], \text{ for } band\_limits[b] \leq k < band\_limits[b+1],$$

where the gain g per parameter band is derived from the ILD parameter:

$$\mathrm{g} = \frac{\mathrm{c}-1}{c+1}, \text{ where } \mathrm{c} = 10^{ILD_i[b]/20}.$$

For parameter bands below cod_max_band, the two channels are updated with the decoded Side signal:

$$L_i[k] = L_i[k] + cod\_gain_i \cdot S_i[k], for\, 1 \leq k < band\_limits[cod\_max\_band],$$
$$R_i[k] = R_i[k] - cod\_gain_i \cdot S_i[k], for\, 1 \leq k < band\_limits[cod\_max\_band],$$

For higher parameter bands, the side signal is predicted and the channels updated as:

$$L_i[\mathrm{k}] = L_i[k] + pred\_gain_i[b] \cdot X_{i-1}[k], \text{for } band\_limits[b] \leq k < band\_limits[b+1],$$
$$R_i[\mathrm{k}] = R_i[k] - pred\_gain_i[b] \cdot X_{i-1}[k], \text{for } band\_limits[b] \leq k < band\_limits[b+1],$$

Finally, the channels are multiplied by a complex value aiming to restore the original energy and the inter-channel phase of the original stereo signal. First, absolute values a[k] are computed for each frequency:

$$a[k] = \sqrt{2 \cdot \frac{X_i^2[k]}{{L_i}^2[k]+{R_i}^2[k]}},$$

before being employed on the left and right spectra in combination with a rotation angle β:

$$L_i[k] = a[k] \cdot e^{j2\pi\beta} \cdot L_i[k], for\, band\_limits[b] \leq k < band\_limits[b+1],$$
$$R_i[k] = a[k] \cdot e^{j2\pi\beta} \cdot R_i[k], for\, band\_limits[b] \leq k < band\_limits[b+1],$$

where a[k] are bound to be between -12 and 12dB, and where $\beta = \mathrm{atan2}(\sin(\mathrm{IPD_i}[b]), \cos(\mathrm{IPD_i}[b]) + c)$, and where atan2(x,y) is the four-quadrant inverse tangent of x over y.

5.5.x.4.6 Time domain synthesis

From the two decoded spectrums L and R, two time domain signals, l and r, are synthesized by an inverse DFT:

$$l_i[n] = \frac{1}{N}\left(\sum_{k=0}^{N/2} L_i[k] \cdot e^{\frac{2\pi jkn}{N}} + \sum_{k=\frac{N}{2}+1}^{N-1} {L_i}^*[N-k] \cdot e^{\frac{2\pi jkn}{N}}\right), \text{ for } 0 \leq n < N$$

$$r_i[n] = \frac{1}{N}\left(\sum_{k=0}^{N/2} R_i[k] \cdot e^{\frac{2\pi jkn}{N}} + \sum_{k=\frac{N}{2}+1}^{N-1} {R_i}^*[N-k] \cdot e^{\frac{2\pi jkn}{N}}\right), \text{ for } 0 \leq n < N$$

where * denotes the complex conjugation.

Finally an overlap-add operation allows reconstructing a frame of M samples:

$$l[i \cdot \mathrm{M} + n - L] = \begin{cases} l_{i-1}[M+n] \cdot w[L-1-n] + l_i[n] \cdot w[n], \text{for } 0 \leq n < L \\ l_i[n] \qquad , \text{for } L \leq n < M \end{cases}$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

$$r[i \cdot \mathrm{M} + n - L] = \begin{cases} r_{i-1}[M+n] \cdot w[L-1-n] + r_i[n] \cdot w[n], \text{for } 0 \le n < L \\ r_i[n] \qquad\qquad , \text{for } L \le n < M \end{cases}$$

5.5.x.4.7 Post-processing

The bass post-processing is applied on two channels separately. The processing is for both channels the same as described in ISO/IEC 23003-3:2012, subclause 7.17.

5.5.x.4.8 Transition from FD mode

The transitions from FD to LPD mode are done first on the decoded Mid signal as in mono case. It is achieved by artificially creating a Mid-signal from the time domain signal decoded in FD mode.

$$x[n - ccfl/2] = 0.5 \cdot l_{i-1}[n] + 0.5 \cdot r_{i-1}[n], \text{for } ccfl \le n < \frac{ccfl}{2} + L\_fac$$

This signal is then conveyed to the LPD decoder for updating the memories and applying the FAC decoding as it is done in the mono case for transitions from FD mode to ACELP. The processing is described in ISO/IEC 23003-3:2012, subclause 7.16. In case of FD mode to TCX, a conventional overlap-add is performed. The LPD stereo decoder receives as input signal a decoded Mid signal where the transition is already done. The stereo decoder outputs then a left and right channel signals which overlap the previous frame decoded in FD mode. The signals are then cross-faded on each channel for smoothing the transition in the left and right channels:

$$l\left[n - \frac{ccfl}{2} + L\_fac\right] = \begin{cases} l_{i-1}[ccfl+n] & , \text{for } 0 \le n < \frac{ccfl}{2} - L\_fac - L \\ l_{i-1}\left[ccfl + \frac{ccfl}{2} - L\_fac - L + n\right] \cdot w[L-1-n] + l_i[n] \cdot w[n], & \text{for } 0 \le n < L \\ l_i[n] & , \text{for } L \le n < M \end{cases}$$

$$r\left[n - \frac{ccfl}{2} + L\_fac\right] = \begin{cases} r_{i-1}[ccfl+n] & , \text{for } 0 \le n < \frac{ccfl}{2} - L\_fac - L \\ r_{i-1}\left[ccfl + \frac{ccfl}{2} - L\_fac - L + n\right] \cdot w[L-1-n] + r_i[n] \cdot w[n], & \text{for } 0 \le n < L \\ r_i[n] & , \text{for } L \le n < M \end{cases}$$

A schematic illustration of the transitions is depicted in Figure AMD3.1 in case LPD is in full-band mode where M=ccfl/2.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

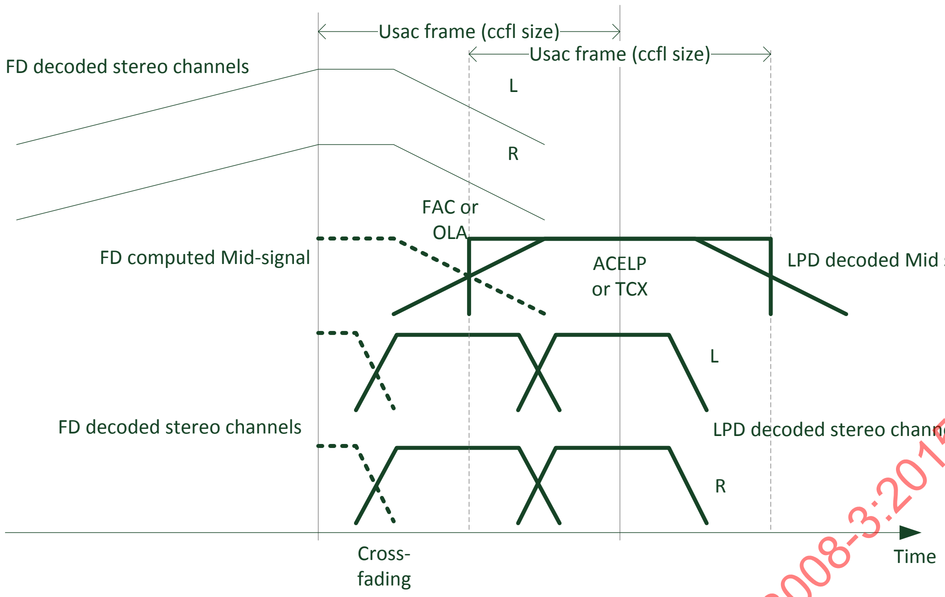


Figure AMD3.1 — Schematic representation of the transition from FD to LPD mode with LPD stereo.

5.5.x.4.9 Transition to FD mode

For transitions from LPD mode to FD mode, an extra frame is decoded by the LPD stereo decoder. The Mid signal coming from the LPD decoder is extended with zero for the frame index i=ccfl/M.

$$x[i \cdot \mathrm{M} + n - L] = \begin{cases} x[i \cdot \mathrm{M} + n - L], \text{for } 0 \le n < L + 2 \cdot L\_fac \\ 0 \qquad\qquad\qquad\quad, \text{for } L + 2 \cdot L\_fac \le n < M \end{cases}$$

The stereo decoding as described in the previous sections is performed by holding the last stereo parameters, and by switching off the Side signal inverse quantization, i.e. **code_mode** is set to 0. Moreover the right side windowing after the inverse DFT is not applied.

The resulting left and right channels are then combined to the FD mode decoded channels of the next frame by using an overlap-add processing in case of TCX to FD mode or by using a FAC for each channel in case of ACELP to FD mode.

A schematic illustration of the transitions is depicted in Figure **AMD3.2** in case LPD in full-band mode where M=ccfl/2.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

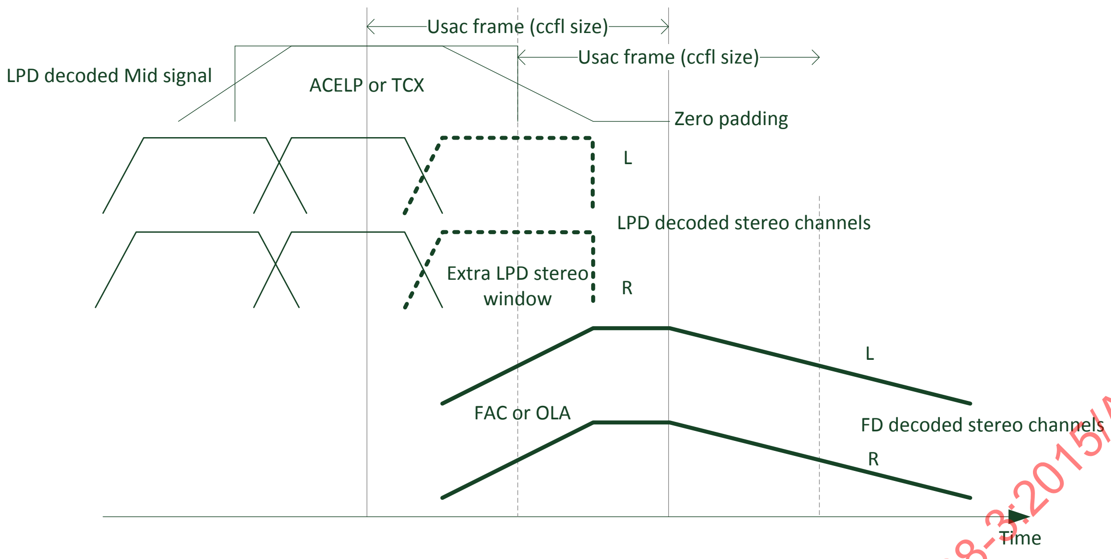


Figure AMD3.2 — Schematic representation of the transition from LPD to FD mode with LPD stereo.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

7 Discrete Multi-Channel Coding Tool

Add new subclause at the end of 5.2.2 and add the following Tables:

5.2.2.X Extension Element Configurations

Table AMD4.1 — Syntax of MCTConfig(),

Syntax	No. of bits	Mnemonic
MCTConfig()		
{		
nMCTChannels = 0;		
for(chan=0;chan < bsNumberOfSignals[grp]+1; chan++) {		
mctChanMask[chan];	**1**	uimsbf
if(mctChanMask[chan] > 0) {		
mctChannelMap[nMCTChannels]=chan;		
nMCTChannels++;		
}		
}		
}		
NOTE: The corresponding ID_USAC_EXT element shall be prior to any audio element of the corresponding signal group.		

At the end of 5.2.3.3 add the following Tables:

Table AMD4.2 — Syntax of MultichannelCodingBoxRotation()

Syntax	No. of Bits	Mnemonic
MultichannelCodingBoxRotation()		
{		
if (keepTree == 0) {	NOTE 1)	
channelPairIndex;	**nBits**	
}		
else {		
channelPairIndex=lastChannelPairIndex;		
}		
hasMctMask;	**1**	uimsbf
hasBandwiseAngles;	**1**	uimsbf
windowsPerFrame =1;		
if (hasMctMask || hasBandwiseAngles) {		
isMCTShort;	**1**	uimsbf
numMaskBands;	**5**	uimsbf
if (isMCTShort) {		
numMaskBands = numMaskBands*8;		
windowsPerFrame =8;		
}		
} else {		
numMaskBands = MAX_NUM_MC_BANDS;		
}		
if (hasMctMask) {		
for(j=0;j<numMaskBands;j++) {		
mctMask[j];	**1**	uimsbf

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

}		
} else {		
for(j=0;j<numMaskBands;j++) {		
mctMask[j] = 1;		
}		
}		
If(indepFlag > 0) {		
mct_delta_time = 0;		
} else {		
mct_delta_time;	**1**	uimsbf
}		
if (hasBandwiseAngles == 0) {		
hcod_angle[dpcm_beta[0]];	**1..10**	vlclbf
}		
else {		
for(j=0;j<numMaskBands;j++) {		
if (mctMask[j] ==1) {		
hcod_angle[dpcm_beta[j]];	**1..10**	vlclbf
}		
}		
}		
}		
NOTE 1) nBits = max(1, floor(log2(nMCTChannels · (nMCTChannels-1)/2 – 1))+1)		

Table AMD4.3 — Syntax of MultichannelCodingBoxPrediction()

Syntax	No. of Bits	Mnemonic
MultichannelCodingBoxPrediction()		
{		
if (keepTree == 0) {	NOTE 1)	
channelPairIndex;	**nBits**	uimsbf
}		
else {		
channelPairIndex= lastChannelPairIndex;		
}		
hasMctMask;	**1**	uimsbf
hasBandwiseCoeff;	**1**	uimsbf
windowsPerFrame =1;		
if (hasMctMask || hasBandwiseCoeff) {		
isMCTShort;	**1**	uimsbf
numMaskBands;	**5**	uimsbf
if (isMCTShort) {		
numMaskBands = numMaskBands*8;		
windowsPerFrame =8;		
}		
} else {		
numMaskBands = MAX_NUM_MC_BANDS;		
}		
if (hasMctMask) {		
for(j=0;j<numMaskBands;j++) {		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

mctMask[j];	**1**	uimsbf
}		
} else {		
for(j=0;j<numMaskBands;j++) {		
mctMask[j] = 1;		
}		
}		
pred_dir;	**1**	
If(indepFlag > 0) {		
mct_delta_time = 0;		
} else {		
mct_delta_time;	**1**	uimsbf
}		
if (hasBandwiseCoeff == 0) {		
hcod_sf[dpcm_alpha_q_re[0]];	**1..19**	vlclbf
}		
else {		
for(j=0;j<numMaskBands;j++) {		
if (mctMask[j] ==1) {		
hcod_sf[dpcm_alpha_q_re[j]];	**1..19**	vlclbf
}		
}		
}		
}		
NOTE 1) nBits = max(1, floor(log2(nMCTChannels · (nMCTChannels-1)/2 – 1))+1)		

Table AMD4.4 — Syntax of MultichannelCodingFrame()

Syntax	No. of bits	Mnemonic
MultichannelCodingFrame()		
{		
MCTSignalingType;	**2**	uimsbf
if (indepFlag == 1) {		
keepTree = 0;		
} else {		
keepTree;	**1**	uimsbf
}		
if (keepTree==0) {		
numPairs = escapedValue(5,8,16);		
}		
else {		
numPairs = lastNumPairs;		
}		
MCTStereoFilling = 0;		
if (MCTSignalingType > 1) {		
MCTSignalingType = MCTSignalingType – 2;		
MCTStereoFilling = 1;		
}		
for(pair=0; pair<numPairs;pair++) {		
hasStereoFilling[pair] = 0;		
if (MCTStereoFilling == 1) {		
hasStereoFilling[pair]**;**	**1**	uimsbf
}		
if (MCTSignalingType == 0) { /* tree of stereo prediction boxes */		
MultichannelCodingBoxPrediction();		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
        }
        if (MCTSignalingType == 1) { /* tree of rotation boxes */
            MultichannelCodingBoxRotation();
        }
    }
}
```

In Table 34 replace

```
    if (!common_window) {
        ics_info();
    }
    if (tw_mdct && !common_tw) {
        tw_data();
    }
    scale_factor_data();

    if (enhancedNoiseFilling) {
        igf_AllZero                                          1        uimsbf
        igf_level(igf_AllZero, indepFlag);                   0…       NOTE
        if (!igf_AllZero) {
            igf_data(indepFlag);
        } else {
            igfPrevTileIdx = {3};
            igf_PrevWhiteningLevel = {0};
        }
    }
```

with:

```
    if (!common_window) {
        ics_info();
    }
    if (tw_mdct && !common_tw) {
        tw_data();
    }

    if (indepFlag) {
        prev_aliasing_symmetry;                              1        uimsbf
    } else {
        prev_aliasing_symmetry = curr_aliasing_symmetry;
    }
    curr_aliasing_symmetry;                                  1        uimsbf

    scale_factor_data();

    if (enhancedNoiseFilling) {
        igf_AllZero;
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
        igf_level(igf_AllZero, indepFlag);              1          uimsbf
        if (!igf_AllZero) {                             0...       NOTE
            igf_data(indepFlag);
        } else {
            igfPrevTileIdx = {3};
            igf_PrevWhiteningLevel = {0};
            igf_WhiteningLevel = {0};
        }
    }
```

Add the following new subclauses "5.5.X Multichannel Coding Tool" at the end of subclause 5.5:

5.5.X Multichannel Coding Tool

5.5.X.1 Tool description

The Multichannel Coding Tool (MCT) is a method for joint coding of multiple channels for more efficient coding of time-variant horizontally and vertically distributed channels.

5.5.X.2 Terms and Definitions

Help elements:

mctChanMask[chan]	Indicates the use of the tool for a certain channel according to Table AMD4.5. The value of mctChanMask[chan] shall be 0 for any LFE channel.

Table AMD4.5 — mctChanMask

mctChanMask	Meaning
0	Multichannel coding tool not applied
1	Multichannel coding tool applied

channelPairIndex	A list of channel pair indices for each pair of channels processed by the MCT. The channelPairIndex is decoded to two channel indices with `decode_channel_pair_index()`.
hasMctMask	Indicates the transmission of a mask that indicates the use of the tool for certain scale factor bands.

Table AMD4.6 — hasMctMask

hasMctMask	Meaning
0	MCT is applied to all bands
1	A mask indicating the usage of MCT per band is transmitted

hasBandwiseAngles	Indicates whether a single angle or multiple angles are transmitted.
hasBandwiseCoeff	Indicates whether a single prediction coefficient or multiple prediction coefficients are transmitted.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

isMCTShort	Indicates whether the current processing is applied to a frame containing eight sub-windows.

Table AMD4.7 — isMCTShort

isMCTShort	Meaning
0	Stereo parameters applied to one MDCT frame
1	Stereo parameters applied to eight MDCT sub-windows

numMaskBands	The number of processing bands that are processed by MCT.
pair	The index of the currently processed stereo processing box.
band	A stereo processing band containing two scalefactor bands.
mctMask [band]	Indicates the activity of the MCT for a certain parameter band within a certain parameter pairing.
mct_delta_time	Indicates the coding scheme used for the MCT parameters:

Table AMD4.8 — mct_delta_time

mct_delta_time	Meaning
0	frequency differential coding of MCT parameters
1	time differential coding of MCT parameters

hcod_angle[ ]	The Huffman code book for angles.
dpcm_beta[band]	The differentially encoded angle to be applied.
dpcm_alpha_q_re[band]	The differentially encoded prediction coefficient to be applied.
pred_dir	Indicates the direction of prediction according to ISO/IEC 23003-3:2012, Table 120.
DEFAULT_ALPHA	Initialization value for stereo prediction, equal to 0.
DEFAULT_BETA	Initialization value for rotation angle, equal to 48.
MAX_NUM_MC_BANDS	Maximum number of MCT bands, equal to 64.
MCTSignalingType	The type of signaling MCT data.
keepTree	Indicates whether to use the same tree of channel pairs as in the previous frame.
numPairs	The number of MCT channel pairs. The maximum number of MCT channel pairs per signal group shall not exceed (nMCTChannels · (nMCTChannels -1) / 2) of that signal group.
nMCTChannels	The number of active MCT channels, where mctChanMask[chan] ==1.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

5.5.X.3 Decoding process

5.5.X.3.1 General

In case an element with `usacExtElementType ID_EXT_ELE_MCT` belongs to the currently processed signal group, the affected channels according to `mctChanMask[]` shall be decoded by the MCT. Here, the extension element with `usacExtElementType ID_EXT_ELE_MCT` shall be written before any audio element of a certain signal group.

The decoding of the Multichannel Coding Tool (MCT) is performed in multiple steps as follows:

5.5.X.3.2 Decoding of channel pair index

Channel pairs are efficiently signaled using a unique index `channelPairIndex` for each pair, dependent on the sum `nMCTChannels` of active channels in the vector `mctChanMask[]`. The decoding process is described in the function `decode_channel_pair_index()` as follows:

```
decode_channel_pair_index(channelPairIndex, channelPair[2])
{
  maxNumPairIdx = nMCTChannels*(nMCTChannels-1)/2 - 1;
  numBits = floor(log2(maxNumPairIdx))+1;
  pairCounter = 0;

  for (chan1=1; chan1 < nMCTChannels; chan1++) {
      for (chan0=0; chan0 < chan1; chan0++) {
        if (pairCounter == channelPairIndex) {
          channelPair[0] = chan0;
          channelPair[1] = chan1;
          return;
        }
        else
          pairCounter++;
      }
    }
  }
}
```

For instance, all possible channel pairs when using 6 channels can be indexed according to the following Table.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table AMD4.9 – Coding of channelPairIndex for a setup with 6 MCT channels

ch nr	0	1	2	3	4	5
0	▨	0	1	2	3	4
1	▨	▨	5	6	7	8
2	▨	▨	▨	9	10	11
3	▨	▨	▨	▨	12	13
4	▨	▨	▨	▨	▨	14
5	▨	▨	▨	▨	▨	▨

5.5.X.3.3 Decoding process for rotation angles

In case `MCTSignalingType = 1`, the `MultichannelCodingBoxRotation()` bit stream element is used. For all rotation angles the difference to a preceding (in time or frequency) value is coded using the Huffman code book specified in 5.5.X.3.6. See ISO/IEC 14496-3:2009, 4.6.3, for a detailed description of the Huffman decoding process. Rotation angles are not transmitted for `mctMask[band] = 0`. The following pseudo code describes how to decode the rotation angles `pairBeta[band]`.

```
decode_rotation()
{
 for(pair=0; pair<numPairs; pair++) {
  mctBandsPerWindow = numMaskBands[pair]/windowsPerFrame;
  for(band=0; band<numMaskBands[pair]; band++) {
   if(mct_delta_time[pair] > 0) {
    lastVal = beta_prev_frame[pair][band%mctBandsPerWindow];
   }
   else {
    if ((band % mctBandsPerWindow) == 0) {
      lastVal = DEFAULT_BETA;
    }
   }
   if (mctMask[pair][band] > 0 ) {

    newBeta = lastVal + dpcm_beta[pair][band];
    if(newBeta >= 65) {
     newBeta -= 65;
    }
    pairBeta[pair][band] = newBeta;
    beta_prev_frame[pair][band%mctBandsPerWindow] = newBeta;
    lastVal = newBeta;
   }
   else {
    beta_prev_frame[pair][band%mctBandsPerWindow] = DEFAULT_BETA; /* -45° */
   }

   /* reset fullband angle */
   beta_prev_fullband[pair] = DEFAULT_BETA;
  }
  for(band=bandsPerWindow; band<MAX_NUM_MC_BANDS; band++) {
   beta_prev_frame[pair][band] = DEFAULT_BETA;
  }
 }
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

beta_prev_frame[pair][sfb] contains the decoded rotation angles of the corresponding stereo channel pair of the last sub-window of the previous frame. If no differential coding was used for the previous frame or for the respective scale factor band in the previous frame, beta_prev_frame[sfb] is set to DEFAULT_BETA.

All rotation angles shall be reset to DEFAULT_BETA upon a transform length change and for all cases the memory is not used for the current frame.

5.5.X.3.4 Decoding process for real-valued stereo prediction

In case MCTSignalingType = 0, the MultichannelCodingBoxPrediction() bit stream element is used. Decoding is performed similar to the decoding of prediction coefficients as defined in ISO/IEC 23003-3:2012, 7.7.2.3.2. The following pseudo code describes how to decode the prediction coefficients pairAlpha[band].

```
decode_prediction()
{
  for(pair=0; pair<numPairs; pair++) {
    mctBandsPerWindow = numMaskBands[pair]/windowsPerFrame;
    for(band=0; band<numMaskBands[pair]; band++) {
      if(mct_delta_time[pair] > 0) {
        lastVal = alpha_prev_frame[pair][band%mctBandsPerWindow];
      }
      else {
        if ((band % mctBandsPerWindow) == 0) {
           lastVal = DEFAULT_ALPHA;
        }
      }
      if (mctMask[pair][band] > 0 ) {

        dpcm_alpha = -decode_huffman() + 60; /* function returns dpcm_alpha_[sfb]*/
        newAlpha = lastVal + dpcm_alpha;

        pairAlpha[pair][band] = newAlpha;
        alpha_prev_frame[pair][band%mctBandsPerWindow] = newAlpha;
        lastVal = newAlpha;
      }
      else {
        alpha_prev_frame[pair][band%mctBandsPerWindow] = DEFAULT_ALPHA;
      }

      /* reset fullband angle */
      alpha_prev_fullband[pair] = DEFAULT_ALPHA;
    }
    for(band=bandsPerWindow; band<MAX_NUM_MC_BANDS; band++) {
      alpha_prev_frame[pair][band] = DEFAULT_ALPHA;
    }
  }
}
```

All prediction coefficients shall be reset to DEFAULT_ALPHA upon a transform length change and for all cases the memory is not used for the current frame.

5.5.X.3.5 Decoding of quantized rotation angles

To avoid floating point differences of trigonometric functions on different platforms, the following lookup-tables for converting rotation angle indices directly to sin/cos shall be used:

```
tabIndexToSinAlpha[65] = {
  -1.000000f,-0.998795f,-0.995185f,-0.989177f,-0.980785f,
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
  -0.970031f,-0.956940f,-0.941544f,-0.923880f,-0.903989f,
  -0.881921f,-0.857729f,-0.831470f,-0.803208f,-0.773010f,
  -0.740951f,-0.707107f,-0.671559f,-0.634393f,-0.595699f,
  -0.555570f,-0.514103f,-0.471397f,-0.427555f,-0.382683f,
  -0.336890f,-0.290285f,-0.242980f,-0.195090f,-0.146730f,
  -0.098017f,-0.049068f, 0.000000f, 0.049068f, 0.098017f,
   0.146730f, 0.195090f, 0.242980f, 0.290285f, 0.336890f,
   0.382683f, 0.427555f, 0.471397f, 0.514103f, 0.555570f,
   0.595699f, 0.634393f, 0.671559f, 0.707107f, 0.740951f,
   0.773010f, 0.803208f, 0.831470f, 0.857729f, 0.881921f,
   0.903989f, 0.923880f, 0.941544f, 0.956940f, 0.970031f,
   0.980785f, 0.989177f, 0.995185f, 0.998795f, 1.000000f
};


tabIndexToCosAlpha[65] = {
  0.000000f, 0.049068f, 0.098017f, 0.146730f, 0.195090f,
  0.242980f, 0.290285f, 0.336890f, 0.382683f, 0.427555f,
  0.471397f, 0.514103f, 0.555570f, 0.595699f, 0.634393f,
  0.671559f, 0.707107f, 0.740951f, 0.773010f, 0.803208f,
  0.831470f, 0.857729f, 0.881921f, 0.903989f, 0.923880f,
  0.941544f, 0.956940f, 0.970031f, 0.980785f, 0.989177f,
  0.995185f, 0.998795f, 1.000000f, 0.998795f, 0.995185f,
  0.989177f, 0.980785f, 0.970031f, 0.956940f, 0.941544f,
  0.923880f, 0.903989f, 0.881921f, 0.857729f, 0.831470f,
  0.803208f, 0.773010f, 0.740951f, 0.707107f, 0.671559f,
  0.634393f, 0.595699f, 0.555570f, 0.514103f, 0.471397f,
  0.427555f, 0.382683f, 0.336890f, 0.290285f, 0.242980f,
  0.195090f, 0.146730f, 0.098017f, 0.049068f, 0.000000f
};
```

5.5.X.3.6 Huffman Tables for Differential Rotation Angles

The following Huffman tables `huff_ctabAngle[]` and `huff_ltabAngle[]` for the code words and the code word lengths, respectively, shall be used for decoding the rotation angle differences:

```
huff_ctabAngle[65] = {
  0x00000000, 0x0000000B, 0x00000012, 0x0000001B, 0x0000001F,
  0x00000031, 0x0000003A, 0x00000043, 0x00000065, 0x00000073,
  0x00000082, 0x0000009A, 0x000000CE, 0x000000EE, 0x00000106,
  0x0000013A, 0x000001D9, 0x000001DE, 0x00000202, 0x00000261,
  0x0000020F, 0x0000020E, 0x00000263, 0x00000266, 0x00000272,
  0x00000271, 0x00000277, 0x00000276, 0x00000334, 0x00000325,
  0x00000326, 0x00000327, 0x00000324, 0x00000323, 0x00000335,
  0x00000322, 0x00000320, 0x00000321, 0x00000273, 0x00000270,
  0x00000267, 0x00000260, 0x000004C4, 0x000004C5, 0x00000203,
  0x000001DF, 0x000001DA, 0x000001D8, 0x0000019B, 0x000001DB,
  0x00000132, 0x00000100, 0x000000CF, 0x000000CC, 0x0000009B,
  0x00000081, 0x00000072, 0x0000004F, 0x00000042, 0x00000038,
  0x00000030, 0x0000001E, 0x0000001A, 0x00000011, 0x0000000A
};


huff_ltabAngle[65] = {
  0x00000001, 0x00000004, 0x00000005, 0x00000005, 0x00000005,
  0x00000006, 0x00000006, 0x00000007, 0x00000007, 0x00000007,
  0x00000008, 0x00000008, 0x00000008, 0x00000008, 0x00000009,
  0x00000009, 0x00000009, 0x00000009, 0x0000000A, 0x0000000A,
  0x0000000A, 0x0000000A, 0x0000000A, 0x0000000A, 0x0000000A,
  0x0000000A, 0x0000000A, 0x0000000A, 0x0000000A, 0x0000000A,
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
  0x0000000A, 0x0000000A, 0x0000000A, 0x0000000A, 0x0000000A,
  0x0000000A, 0x0000000A, 0x0000000A, 0x0000000A, 0x0000000A,
  0x0000000A, 0x0000000A, 0x0000000B, 0x0000000B, 0x0000000A,
  0x00000009, 0x00000009, 0x00000009, 0x00000009, 0x00000009,
  0x00000009, 0x00000009, 0x00000008, 0x00000008, 0x00000008,
  0x00000008, 0x00000007, 0x00000007, 0x00000007, 0x00000006,
  0x00000006, 0x00000005, 0x00000005, 0x00000005, 0x00000004
};
```

5.5.X.3.7 Application of Multichannel Coding Tool

5.5.X.3.7.1 General

Reconstruct the spectral coefficients of all channels by iteratively looping over all transmitted stereo boxes and frequency bands as follows:

```
decode_mct()
{
  for (pair=0; pair < numPairs; pair++) {

    mctBandOffset = 0;
    alphaSfb = pairAlpha[pair];
    betaSfb = pairBeta[pair];

    /* inverse MCT application */
    for (win = 0, group = 0; group <num_window_groups; group++) {

      for (groupwin = 0; groupwin < window_group_length[group]; groupwin++, win++) {
        *dmx = spectral_data[ch1][win];
        *res = spectral_data[ch2][win];
        apply_mct_wrapper(self,dmx,res,
                          &alphaSfb[mctBandOffset], &betaSfb[mctBandOffset],
                          &mctMask[mctBandOffset],mctBandsPerWindow, alpha,
                          pair,nSamples);
      }
      mctBandOffset += mctBandsPerWindow;
    }
  }
}
```

Thereby `spectral_data[ch1]` and `spectral_data[ch2]` represent the two input and output channels of the channel pair that is currently processed in the MCT stereo processing box.

Further processing of every MCT stereo processing box is done as follows:

```
apply_mct_wrapper(self, *dmx, *res,
                  *alphaSfb, *betaSfb,
                  *mctMask, mctBandsPerWindow, alpha,
                  pair, nSamples)
{
  sfb = 0;

  if (MCTSignalingType == 0) {
    if (!bHasBandwiseCoeff[pair] && !bHasMctMask[pair]) {
      apply_mct_prediction(dmx, res, alphaSfb[0], nSamples);
    }
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
    else {
      /* apply bandwise processing */
      for (i = 0; i< mctBandsPerWindow; i++) {
        if (mctMask[i] == 1) {
          startLine = swb_offset [sfb];
          stopLine  = (sfb+2<num_swb)? swb_offset [sfb+2] : swb_offset [sfb+1];
          nSamples  = stopLine-startLine;

          apply_mct_prediction(&dmx[startLine], &res[startLine],
                               alphaSfb[i], nSamples, pred_dir);
        }
        sfb += 2;

        /* break condition */
        if (sfb >= num_swb) {
          break;
        }
      }
    }
  }
  else if (MCTSignalingType == 1) {

    /* apply fullband box */
    if (!bHasBandwiseAngles[pair] && !bHasMctMask[pair]) {
      apply_mct_rotation(dmx, res, betaSfb[0], nSamples);
    }
    else {
      /* apply bandwise processing */
      for (i = 0; i< mctBandsPerWindow; i++) {
        if (mctMask[i] == 1) {
          startLine = swb_offset [sfb];
          stopLine  = (sfb+2<num_swb)? swb_offset [sfb+2] : swb_offset [sfb+1];
          nSamples  = stopLine-startLine;

          apply_mct_rotation(&dmx[startLine], &res[startLine],
                             betaSfb[i], nSamples);
        }
        sfb += 2;

        /* break condition */
        if (sfb >= num_swb) {
          break;
        }
      }
    }
  }
  else if (MCTSignalingType == 2) {
    /* reserved */
  }
  else if (MCTSignalingType == 3) {
    /* reserved */
  }
}
```

5.5.X.3.7.2 Application of rotation angles

```
apply_mct_rotation(*dmx, *res, aIdx, nSamples)
{
  for (n=0;n<nSamples;n++) {
    L =  dmx[n] * tabIndexToCosAlpha [aIdx] - res[n] * tabIndexToSinAlpha [aIdx];
    R =  dmx[n] * tabIndexToSinAlpha [aIdx] + res[n] * tabIndexToCosAlpha [aIdx];
    dmx[n] = L;
    res[n] = R;
  }
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
}
```

5.5.X.3.7.3 Application of real-valued stereo prediction coefficients

Real-valued Stereo Prediction is performed like the upmixing process described in ISO/IEC 23003-3:2012 subclause 7.7.2.3.4 under the assumption that `ms_mask_present = 3; num_window_groups = windowsPerFrame; window_group_length = 1; cplx_pred_used[g][sfb] = mctMask[sfb]; alpha_im = 0;`

Thus, in the context of the MCT, the prediction upmixing process can be calculated using the following pseudo code:

```
apply_mct_prediction(*dmx, *res, alpha_q, nSamples, pred_dir)
{
  alpha_re = alpha_q * 0.1;

  for (n=0;n<nSamples;n++) {
    if (pred_dir == 0) {
      L = dmx[n] + alpha * dmx[n] + res[n];
      R = dmx[n] - alpha * dmx[n] - res[n];

    }
    else {
      L =   dmx[n] + alpha * dmx[n] + res[n];
      R = - dmx[n] + alpha * dmx[n] + res[n];
    }
    dmx[n] = L;
    res[n] = R;
  }
}
```

5.5.X.4 Stereo Filling in the MCT

Like Stereo Filling for IGF in a channel pair element, described in subclause 5.5.5.4.9, Stereo Filling in the Multichannel Coding Tool (MCT) fills "empty" scale factor bands (which are fully quantized to zero) at and above the noise filling start frequency using a downmix of the previous frame's output spectra.

5.5.X.4.1 Tool Description

When Stereo Filling is active in a MCT joint-channel pair (**hasStereoFilling**[pair] ≠ 0 in Table AMD4.4), all "empty" scale factor bands in the noise filling region (i. e. starting at or above noiseFillingStartOffset) of the pair's second channel are filled to a specific target energy using a downmix of the corresponding output spectra (after MCT application) of the previous frame. This is done after the FD noise filling (see 7.2 in ISO/IEC 23003-3:2012) and prior to scale factor and MCT joint-stereo application. All output spectra after completed MCT processing are saved for potential Stereo Filling in the next frame.

5.5.X.4.2 Operational Constraints

Cascaded execution of Stereo Filling algorithm (**hasStereoFilling**[pair] ≠ 0) in empty bands of the second channel is not supported for any following MCT stereo pair with **hasStereoFilling**[pair] ≠ 0 if the second channel is the same. In a channel pair element, active IGF Stereo Filling in the second (residual) channel according to subclause 5.5.5.4.9 takes precedence over – and, thus, disables – any subsequent application of MCT Stereo Filling in the same channel of the same frame.

5.5.X.4.3 Terms and Definitions

hasStereoFilling[pair]	indicates usage of Stereo Filling in currently processed MCT channel pair

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

ch1, ch2	indices of channels in currently processed MCT channel pair
spectral_data[ ][ ]	spectral coefficients of channels in currently processed MCT channel pair
spectral_data_prev[ ][ ]	output spectra after completed MCT processing in previous frame
downmix_prev[ ][ ]	estimated downmix of previous frame’s output channels with indices given by currently processed MCT channel pair
num_swb	total number of scale factor bands, see ISO/IEC 23003-3, subclause 6.2.9.4
ccfl	coreCoderFrameLength, transform length, see ISO/IEC 23003-3, subclause 6.1.
noiseFillingStartOffset	Noise Filling start line, defined depending on ccfl in ISO/IEC 23003-3, Table 109.
igf_WhiteningLevel	Spectral whitening in IGF, see 5.5.5.4.7
seed[ ]	Noise Filling seed used by randomSign(), see ISO/IEC 23003-3, subclause 7.2.

5.5.X.4.4 Decoding Process

MCT Stereo Filling is performed using four consecutive operations, which are described in the following.

Step 1: Preparation of second channel’s spectrum for Stereo Filling algorithm

If the Stereo Filling indicator for the given MCT channel pair, **hasStereoFilling**[pair], equals zero, Stereo Filling is not used and the following steps are not executed. Otherwise, scale factor application is undone if it was previously applied to the pair’s second channel spectrum, spectral_data[ch2].

Step 2: Generation of previous downmix spectrum for given MCT channel pair

The previous downmix is estimated from the previous frame’s output signals spectral_data_prev[ ][ ] that was stored after application of MCT processing. If a previous output channel signal is not available, e.g. due to an independent frame (indepFlag>0), a transform length change or core_mode == 1 , the previous channel buffer of the corresponding channel shall be set to zero.

For prediction stereo pairs, i.e. MCTSignalingType == 0, the previous downmix is calculated from the previous output channels as downmix_prev[ ][ ] defined in step 2 of subclause 5.5.5.4.9.4, whereby spectrum[window][ ] is represented by spectral_data[ ][window].

For rotation stereo pairs, i.e. MCTSignalingType == 1, the previous downmix is calculated from the previous output channels by inverting the rotation operation defined in subclause 5.5.X.3.7.1.

```
apply_mct_rotation_inverse(*R, *L, *dmx, aIdx, nSamples)
{
  for (n=0; n<nSamples; n++) {
    dmx =  L[n] * tabIndexToCosAlpha[aIdx] + R[n] * tabIndexToSinAlpha[aIdx];
  }
}
```

using L = spectral_data_prev[ch1][ ], R = spectral_data_prev[ch2][ ], dmx = downmix_prev[ ] of the previous frame and using aIdx, nSamples of current frame and MCT pair.

Step 3: Execution of Stereo Filling algorithm in empty bands of second channel

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Stereo Filling is applied in the MCT pair's second channel as in step 3 of subclause 5.5.5.4.9.4, whereby spectrum[window] is represented by spectral_data[ch2][window] and max_sfb_ste is given by num_swb.

Step 4: Scale factor application and adaptive synchronization of Noise Filling seeds

As after step 3 of subclause 5.5.5.4.9.4, the scale factors are applied on the resulting spectrum as in 7.3 of ISO/IEC 23003-3, with the scale factors of empty bands being processed like regular scale factors. In case a scale factor is not defined, e.g. because it is located above max_sfb, its value shall equal zero. If IGF is used, igf_WhiteningLevel equals 2 in any of the second channel's tiles, and both channels do not employ eight-short transformation, the spectral energies of both channels in the MCT pair are computed in the range from index noiseFillingStartOffset to index ccfl/2 – 1 before executing decode_mct( ). If the computed energy of the first channel is more than eight times greater than the energy of the second channel, the second channel's seed[ch2] is set equal to the first channel's seed[ch1].

In 5.5.3 (Transform Splitting), replace all occurrences of the words:

- *"MDCTs" with the words "lapped transforms",*
- *"MDCT" with the word "transform",*
- *"IMDCTs" with the words "inverse lapped transforms", and*
- *"IMDCT" with the words "inverse lapped transform".*

Replace subclause 5.5.3.5.1 (IMDCT) with the following text:

5.5.3.5.1 Inverse Lapped Transform

The processing for TS follows the description given in ISO/IEC 23003-3:2012 subclause "7.9, Filterbank and block switching". The following modifications or additions shall be taken into account. See also 5.5.Z.

The TS coefficients in spec[] are de-interleaved using a helper buffer[] with *N*, the window length based on the **window_sequence** value (*N* = ccfl·2 since, by definition, the sequence isn't an EIGHT_SHORT_SEQUENCE):

```
for (i = 0, i2 = 0; i < N/2; i += 1, i2 += 2) {
  spec[0][i] = spec[i2];   /* isolate 1st window */
  buffer[i]  = spec[i2+1]; /* isolate 2nd window */
}
for (i = 0; i < N/2; i += 1) {
  spec[1][i] = buffer[i];  /* copy 2nd window */
}
```

The inverse transform for each half-length TS spectrum spec[0, 1] is then defined as follows, with cs() and k_0 as specified, via the prev_aliasing_symmetry and curr_aliasing_symmetry values, by Table AMD4.9 Joint Channels for Low Bitrate Coding

$$x_{j,n} = \frac{2}{N}\sum_{k=0}^{\frac{N}{4}-1} spec[j][k]\cdot \mathrm{cs}\left(\frac{4\pi}{N}(n+n_0)(k+k_0)\right) \quad \text{for } 0 \le n < N/2\,,$$

where j = window index, must be 0 or 1,

n = time-domain sample index,

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

k = spectral coefficient index,

$n_0 = (N / 4 + 1) / 2$.

Note that for the second inverse transform $x_{1,n}$, prev_aliasing_symmetry is set to curr_aliasing_symmetry. Subsequent windowing and block switching steps for the transform outputs $x_{(0,1)}$ are defined in the next subclauses.

In 5.5.3.5.3 (Transform Splitting with LONG_START_SEQUENCE) replace:

It comprises three windows defined as follows, where *N_l/* is set to 1024 (768), N_s is set to 256 (192) respectively.

with:

It comprises three windows defined as follows, where *N_l/2* is set to 1024 (768), *N_s* is set to 256 (192), respectively.

Replace the first line of the second-to-last equation in this subclause,

$$-1 \cdot x_{0,N\_s-n-1} \cdot W_0(N\_s-n-1), \text{ for } 0 \le n < N\_s$$

with

$$sign \cdot (x_{0,3N\_s-1-n} \cdot W_0(3N\_s-1-n) + x_{1,N\_s-1-n} \cdot W_1(N\_s-1-n)), \text{ for } 0 \le n < N\_s$$

Finally, replace the text

The final windowed time domain values $Z_{i,n}$ are obtained by applying W_2:

with

where *sign* = –1 if prev_aliasing_symmetry == 0, or *sign* = 1 otherwise. The final windowed time-domain values $Z_{i,n}$ are obtained by applying W_2:

At the end of subclause 5.5 add a new subclause with the following text:

5.5.Z Filterbank and Block Switching

The frequency-to-time transformation, windowing, block switching, and overlap-and-add operations are carried out as specified in ISO/IEC 23003-3:2012, subclause 7.9. The only exception is the analytical expression for the inverse lapped transform $x_{i,n}$ of the spectral coefficients *spec*[*i*] for the window index *i*, which is now given by

$$x_{i,n} = \frac{2}{N} \sum_{k=0}^{\frac{N}{2}-1} spec[i][k] \cdot \mathrm{cs}\left(\frac{2\pi}{N}(n+n_0)(k+k_0) \right) \qquad \text{for } 0 \le n < N\,,$$

with *i*, *k*, *n*, *N*, and n_0 defined as in ISO/IEC 23003-3:2012, 7.9.3.1, and with cs() and k_0 as tabulated, using the prev_aliasing_symmetry and curr_aliasing_symmetry values, in Table AMD4.10 below. Note that for the 7 last transforms (at $i > 0$) of an EIGHT_SHORT_SEQUENCE, prev_aliasing_symmetry is set to curr_aliasing_symmetry.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table AMD4.10 – Mapping of aliasing symmetry values to parameters of the inverse lapped transform $x_{i,n}$

last frame *i*-1	**current frame *i***	
	right-side symmetry even (*symm*$_i$=0)	**right-side symmetry odd (*symm*$_i$=1)**
right-side symmetry even (*symm*$_{i-1}$=0)	$\mathrm{cs}(\ldots) = \cos(\ldots)$ $k_0 = 0.5$	$\mathrm{cs}(\ldots) = \sin(\ldots)$ $k_0 = 1.0$
right-side symmetry odd (*symm*$_{i-1}$=1)	$\mathrm{cs}(\ldots) = \cos(\ldots)$ $k_0 = 0.0$	$\mathrm{cs}(\ldots) = \sin(\ldots)$ $k_0 = 0.5$
NOTE *symm*$_{i-1}$ = value of prev_aliasing_symmetry, *symm*$_i$ = value of curr_aliasing_symmetry		

.

In channels and frames with LPD coding (core_mode[ch] ≠ 0), prev_aliasing_symmetry and curr_aliasing_symmetry shall be zero.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

8 Updates to MHAS

In Table 139 replace:

```
        case PACTYP_MPEGH3DAFRAME:
            mpegh3daFrame();
            break;
        case PACTYP_FILLDATA:
            for (i=0; i< MHASPacketLength; i++) {
                mhas_fill_data_byte(i);                     8           bslbf
            }
            break;
        case PACTYP_SYNCGAP:
            syncSpacingLength = escapedValue(16,24,24);     16,40,64    uimsbf
            break;
        case PACTYP_MARKER:
            for (i=0; i< MHASPacketLength; i++) {
                marker_byte(i);                             8           bslbf
            }
            break;
        case PACTYP_CRC16:
            mhasParity16Data;                               16          bslbf
            break;
        case PACTYP_CRC32:
            mhasParity32Data;                               32          bslbf
            break;
        case PACTYP_DESCRIPTOR:
            for (i=0; i< MHASPacketLength; i++) {
                mhas_descriptor_data_byte(i);               8           bslbf
            }
            break;
        case PACTYP_USERINTERACTION:
            mpegh3daElementInteraction();
            break;
        case PACTYP_LOUDNESS_DRC:
            mpegh3daLoudnessDrcInterface();
            break;
        case PACTYP_BUFFERINFO:
            mhas_buffer_fullness_present                    1           uimsbf
            if (mhas_buffer_fullness_present)
                mhas_buffer_fullness = escapedValue(15,24,32);  15,39,71    uimsbf
            }
            break;
    }
```

with:

```
        case PACTYP_MPEGH3DAFRAME:
            mpegh3daFrame();
            break;
        case PACTYP_AUDIOSCENEINFO:
            mae_AudioSceneInfo();
            break;
        case PACTYP_FILLDATA:
            for (i=0; i< MHASPacketLength; i++) {
                mhas_fill_data_byte(i);                     8           bslbf
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

}		
break;		
case PACTYP_SYNCGAP:		
syncSpacingLength = escapedValue(16,24,24);	**16,40,64**	**uimsbf**
break;		
case PACTYP_MARKER:		
for (i=0; i< MHASPacketLength; i++) {		
marker_byte(i);	**8**	**bslbf**
}		
break;		
case PACTYP_CRC16:		
mhasParity16Data;	**16**	**bslbf**
break;		
case PACTYP_CRC32:		
mhasParity32Data;	**32**	**bslbf**
break;		
case PACTYP_GLOBAL_CRC16:		
global_CRC_type;	**2**	**bslbf**
numProtectedPackets;	**6**	**bslbf**
mhasParity16Data;	**16**	**bslbf**
break;		
case PACTYP_ GLOBAL_CRC32:		
global_CRC_type;	**2**	**bslbf**
numProtectedPackets;	**6**	**bslbf**
mhasParity32Data;	**32**	**bslbf**
break;		
case PACTYP_DESCRIPTOR:		
for (i=0; i< MHASPacketLength; i++) {		
mhas_descriptor_data_byte(i);	**8**	**bslbf**
}		
break;		
case PACTYP_USERINTERACTION:		
mpegh3daElementInteraction();		
break;		
case PACTYP_LOUDNESS_DRC:		
mpegh3daLoudnessDrcInterface();		
break;		
case PACTYP_BUFFERINFO:		
mhas_buffer_fullness_present	**1**	**uimsbf**
if (mhas_buffer_fullness_present)		
mhas_buffer_fullness = escapedValue(15,24,32);	**15,39,71**	**uimsbf**
}		
break;		
case PACTYP_AUDIOTRUNCATION:		
audioTruncationInfo();		
break;		
}		

Add new Table in 14.2.2:

Table AMD5.1 – Syntax of audioTruncationInfo()

Syntax	No. of bits	Mnemonic
audioTruncationInfo()		
{		
isActive;	**1**	**bool**

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
        reserved;                                          1          bool
        truncFromBegin;                                    1          bool
        nTruncSamples;                                     13         uimsbf
}
```

Add new subclause in 14.3. after 14.3.2:

14.3.X Subsidiary MHAS packets

isActive	If 1 the truncation message is active, if 0 the decoder should ignore the message.
reserved	reserved bit shall be zero
truncFromBegin	if 0 truncate samples from the end, if 1 truncate samples from the beginning.
nTruncSamples	number of samples to truncate

Add in "Table 141 — Value of MHASPacketType()":

Table 141 — Value of MHASPacketType

MHASPacketType	Value
PACTYP_FILLDATA	0
PACTYP_MPEGH3DACFG	1
PACTYP_MPEGH3DAFRAME	2
PACTYP_AUDIOSCENEINFO	3
/ reserved for ISO use */*	4-5
PACTYP_SYNC	6
PACTYP_SYNCGAP	7
PACTYP_MARKER	8
PACTYP_CRC16	9
PACTYP_CRC32	10
PACTYP_DESCRIPTOR	11
PACTYP_USERINTERACTION	12
PACTYP_LOUDNESS_DRC	13
PACTYP_BUFFERINFO	14
PACTYP_GLOBAL_CRC16	15
PACTYP_GLOBAL_CRC32	16
PACTYP_AUDIOTRUNCATION	17
/ reserved for ISO use */*	18-127

Add in "14.3.2 MHASPacketPayload()"

mpegh3daFrame()	An MPEG-H 3D audio payload as defined in 5.2.3.1.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

mae_AudioSceneInfo()	An MPEG-H 3D audio scene information structure as defined in 15.2.

Add in "14.3.2 MHASPacketPayload()"

global_CRC_type	This element provides an indication on the packet types allowed within the numProtectedPackets protected packets with the global CRC specified in MHASPacketType PACTYP_GLOBAL_CRC32 or PACTYP_GLOBAL_CRC16, according to Table **AMD5.2**.

Table AMD5.2 — Value of global_CRC_type

value	Indication on the packet types allowed within the numProtectedPackets
0	multiple packets of any packet type.
1	multiple packets of any packet type, of which only one packet of type PACTYP_MPEGH3DAFRAME.
2-3	reserved for ISO use

numProtectedPackets	a 6-bit field that indicates the number of MHAS packets protected by the CRC check defined in the MHASPacketType PACTYP_GLOBAL_CRC16 and PACTYP_GLOBAL_CRC32.

Replace in "14.4.9 ***PACTYP_USERINTERACTION****":*

The MHASPacketType PACTYP_USERINTERACTION may be used to feed element interaction data to the decoder.

For this packet type, MHASPacketLabel shall have the same value as the packet of MHASPacketType PACTYP_MPEGH3DACFG, which the user interaction data refers to.

with:

The MHASPacketType PACTYP_USERINTERACTION may be used to feed element interaction data in the form of the mpegh3daElementInteraction() structure to the decoder.

For this packet type, MHASPacketLabel shall have the same value as the packets of MHASPacketType PACTYP_MPEGH3DACFG and PACTYP_AUDIOSCENEINFO (if present), which the user interaction data refers to. If its value is different, the user interaction specified in mpegh3daElementInteraction() structure shall not be applied**.**

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

At the end of "14.4 Description of MHASPacketTypes" add new subclause as follows:

14.4.x PACTYP_GLOBAL_CRC16 and PACTYP_ GLOBAL_CRC32

The MHASPacketType PACTYP_GLOBAL_CRC16 and PACTYP_ GLOBAL_CRC32 may be used for detection of errors in the subsequent numProtectedPackets MHAS packets. For this packet type, MHASPacketLabel has no meaning and shall be set to 0. This may be beneficial when an MHAS stream is conveyed over an error prone channel.

The error detection method uses one of the generator polynomial and associated shift register states as defined for mhasParity16Data or mhasParity32Data respectively.

The CRC-check includes:

— first all bits positioned before the mhasParity32Data/mhasParity16Data in the MHAS packet of type PACTYP_GLOBAL_CRC32 or PACTYP_GLOBAL_CRC16, corresponding to the fields: MHASPacketType, MHASPacketLabel, MHASPacketLength, global_CRC_type and numProtectedPackets,

— and afterwards all bits of the MHASAudioStreamPacket() of the subsequent numProtectedPackets MHAS packets.

In the case where there are no errors, each of the outputs of the shift register shall be zero. At the CRC encoder the **mhasParity16Data** / **mhasParity32Data** field is encoded with a value such that this is ensured.

14.4.x PACTYP_AUDIOTRUNCATION

The MHAS package of type PACTYP_AUDIOTRUNCATION indicates a potential truncation. Truncation in this context means the removal of audio samples from the decoded PCM samples. Audio samples are removed either before or after a truncation point as signaled in the truncation packet.

The packet contains a flag, **isActive**, which indicates whether the truncation shall actually be applied. If this flag is 0 the truncation package shall be ignored.

If the process of truncation is applied after the mixing stage, i.e. after the signal has passed all core decoder and rendering stages, but before the post-processing (DRC-2, binauralization etc.) and end-of-chain (DRC-3 etc.) the truncation point occurs delayed by the core decoding and rendering delay.

If the process of truncation occurs at a later stage, after the mixing stage (e.g., after end-of-chain), the truncation point shall be delayed by the corresponding delay of the additional processing blocks (e.g., DRC-2, binauralization) and the truncation shall be carried out at this later truncation stage (e.g. after end-of-chain).The decoded samples are either truncated from the beginning (if **truncFromBegin**==1) or from the end (if **truncFromBegin**==0). In the case of a truncation from the end, the decoder shall discard all samples at the truncation stage after the truncation point. In the case of a truncation from begin, the decoder shall discard all samples at the truncation stage prior to the truncation point.

Metadata shall be applied to the truncated audio samples such that the resulting audio samples are identical to those that would have resulted from applying the metadata to the non-truncated audio sample frame.

For correct application of the truncation the following additional rules apply:

- If **truncFromBegin** == 1
  - — The MHAS stream shall contain an additional MHAS packet of type PACTYP_MPEGH3DACFG,
  - — The MHAS truncation packet shall occur after the PACTYP_MPEGH3DACFG and before the corresponding PACTYP_MPEGH3DAFRAME that contains the AU to truncate.
- If **truncFromBegin** == 0

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

— The MHAS truncation packet shall occur before the corresponding PACTYP_MPEGH3DAFRAME that contains the AU to truncate,

— If **isActive**==1 the decoder shall perform a decoder re-initialization as if a change of decoder configuration had occurred.

Truncation messages shall be processed jointly with the AU of the following PACTYP_MPEGH3DAFRAME packet with identical MHASPacketLabel,

14.4.x PACTYP_AUDIOSCENEINFO

MHAS Packets of the MHASPacketType PACTYP_AUDIOSCENEINFO embed an MPEG-H 3D audio scene information structure, mae_AudioSceneInfo(), in the MHASPacketPayload().

MHASPacketLabel indicates the ID used for this audio scene information. This audio scene information shall be used for packets of type PACTYP_MPEGH3DAFRAME that have the same ID and packets of type PACTYP_MPEGH3DACFG, so that a unique link between audio scene information, configuration data and payloads is possible.

If present, the MHASPacketType PACTYP_AUDIOSCENEINFO shall occur immediately after each MHAS packet of type PACTYP_MPEGH3DACFG. In this case the mpegh3daConfig() within the MHAS packet of type PACTYP_MPEGH3DACFG shall not contain a mae_AudioSceneInfo() structure.

If MHAS packets of types PACTYP_MPEGH3DACFG and PACTYP_AUDIOSCENEINFO are both not present, the mae_AudioSceneInfo() can be conveyed through out-of-band means, such as session announcement/description/control protocols, either within the mpegh3daConfig() or separately.

At the end of 14.5 add new subclause as follows:

14.5.X CRC error detection

Certain applications require additional error protection, for example if MHAS is used on common serial interfaces (e.g. AES/EBU, S/PDIF), while a minimum overhead in terms of bitrate is added. Figure AMD5.1 and Figure AMD5.2 illustrate two examples on how the CRC and global CRC packets can be used for error detection.

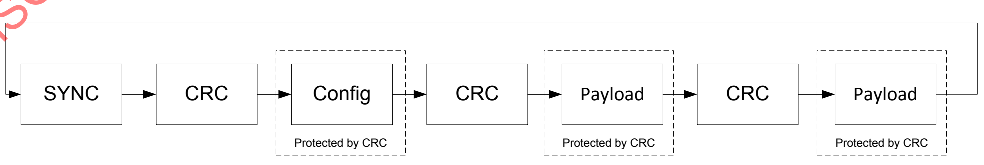


Figure AMD5.1 — Example 5

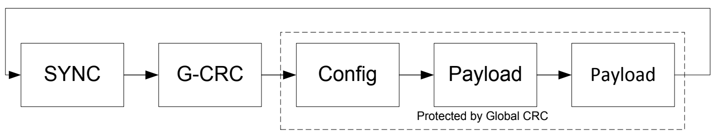


Figure AMD5.2 — Example 6

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

14.5.X Audio sample truncation

Some applications may require that at least some audio frames need to be truncated to a number of audio samples lower than the audio frame size. One example of such an application is sample accurate alignment of an audio stream to a video stream that has a different frame rate. Furthermore, this truncation may have to be inserted on the fly, e.g., for stream splicing. Such an example of stream splicing using the MHAS packets of type PACTYP_AUDIOTRUNCATION for switching between two MHAS streams, coming from different sources, is illustrated in Figure AMD5.3.

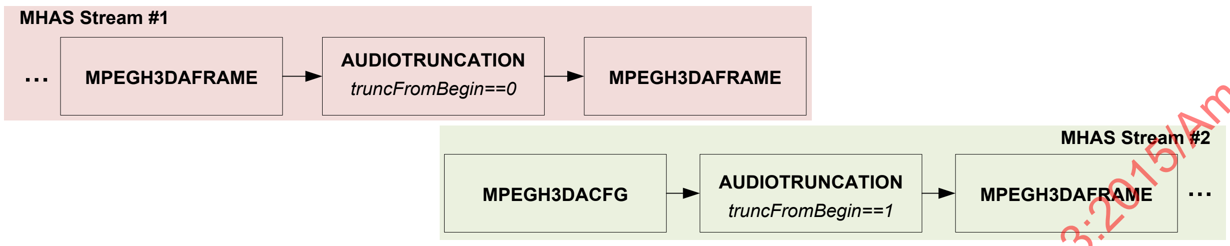


Figure AMD5.3 — Example 7

In the first stream, the MHAS packet containing the truncation message with the number of samples to be truncated at the end of the AU of the following PACTYP_MPEGH3DAFRAME indicates also the end of the stream.

Add the following paragraph to "clause 14.6" before "EXAMPLE":

To enable identification of several configurations inside one stream in the case of multi-stream delivery, the MHAS packet label shall be used as follows: The label values "1" to "16" (0x01-0x10) shall be used for the main stream, the label values "17" to "32" (0x11-0x20) shall be used for the first sub-stream, the next 16 values (0x21-0x30) for the second sub-stream, and so on.

Table AMD5.3 — Meaning of MHASPacketLabel in multi-stream environments

Value of MHASPacketLabel	Meaning
0x01-0x10	Main Stream
0x11-0x20	First Sub-Stream
0x21-0x30	Second Sub-Stream
...	…

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

9 Metadata Updates

9.1 Update of mae_Data() syntax and semantics

*In **15.1** replace*

Group presets define a combination of groups in an audio scene. A group preset contains a list of groups and an associated on/off-status for each of these groups (presets' conditions) which are not modifiable by the user. With group presets a content creator or an application can provide a restricted number of meaningful rendering options to the user.

with

Group presets define a combination of groups in an audio scene. A group preset contains a list of groups or switch groups, each referenced by their unique ID and an associated on/off-status for each of the referenced structures (presets' conditions) which are not modifiable by the user. With group presets a content creator or an application can provide a restricted number of meaningful rendering options to the user.

*Replace **Table 150** in **15.2** with*

Table 150 — Syntax of mae_GroupPresetDefinition()

Syntax	No. of bits	Mnemonic
mae_GroupPresetDefinition(numGroupPresets)		
{		
for (gp = 0; gp < numGroupPresets; gp++) {		
mae_groupPresetID[gp];	**5**	**uimsbf**
mae_groupPresetKind[gp];	**5**	**uimsbf**
mae_bsGroupPresetNumConditions[gp];	**4**	**uimsbf**
for (cnd = 0; cnd < mae_bsGroupPresetNumConditions[gp] + 1; cnd++) {		
mae_groupPresetReferenceID[gp][cnd];	**7**	**uimsbf**
mae_groupPresetConditionOnOff[gp][cnd];	**1**	**bslbf**
if (mae_groupPresetConditionOnOff[gp][cnd]) {		
mae_groupPresetDisableGainInteractivity[gp][cnd];	**1**	**bslbf**
mae_groupPresetGainFlag[gp][cnd];	**1**	**bslbf**
if (mae_groupPresetGainFlag[gp][cnd]) {		
mae_groupPresetGain[gp][cnd];	**8**	**uimsbf**
}		
mae_groupPresetDisablePositionInteractivity[gp][cnd];	**1**	**bslbf**
mae_groupPresetPositionFlag[gp][cnd];	**1**	**bslbf**
if (mae_groupPresetPositionFlag [gp][cnd]) {		
mae_groupPresetAzOffset[gp][cnd];	**8**	**uimsbf**
mae_groupPresetElOffset[gp][cnd];	**6**	**uimsbf**
mae_groupPresetDistFactor[gp][cnd];	**4**	**uimsbf**
}		
}		
}		
}		
}		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Add the following tables after ***Table 151*** *in* ***15.2****:*

Table AMD7.1 — Syntax of mae_ProductionScreenSizeDataExtension()

Syntax	No. of bits	Mnemonic
mae_ProductionScreenSizeDataExtension()		
{		
mae_overwriteProductionScreenSizeData;	1	**bslbf**
if (mae_overwriteProductionScreenSizeData) {		
/* NON-CENTERED DEFAULT PRODUCTION SCREEN */		
bsScreenSizeLeftAz;	10	**uimsbf**
bsScreenSizeRightAz;	10	**uimsbf**
}		
mae_NumPresetProductionScreens;	5	**uimsbf**
for (n = 0; n < mae_NumPresetProductionScreens; n++) {		
mae_productionScreenGroupPresetID[n];	5	**uimsbf**
mae_hasNonStandardScreenSize[n];	1	**bslbf**
if (mae_hasNonStandardScreenSize[n]) {		
isCenteredInAzimuth[n];	1	**bslbf**
if (isCenteredInAzimuth[n]) {		
bsScreenSizeAz[n];	9	**uimsbf**
} else {		
bsScreenSizeLeftAz[n];	10	**uimsbf**
bsScreenSizeRightAz[n];	10	**uimsbf**
}		
bsScreenSizeTopEl[n];	9	**uimsbf**
bsScreenSizeBottomEl[n];	9	**uimsbf**
}		
}		
}		

Table AMD7.3 — Syntax of mae_GroupPresetDefinitionExtension()

Syntax	No. of bits	Mnemonic
mae_GroupPresetDefinitionExtension()		
{		
for (gp = 0; gp < mae_numGroupPresets; gp++) {		
mae_hasSwitchGroupConditions[gp];	1	**bslbf**
if (mae_hasSwitchGroupConditions[gp]) {		
temp = mae_bsGroupPresetNumConditions[gp] +1;		
for (cnd = 0; cnd < temp; cnd++) {		
mae_isSwitchGroupCondition[gp][cnd];	1	**bslbf**
}		
}		
mae_hasDownmixIdGroupPresetExtensions[gp];	1	**bslbf**
if (mae_hasDownmixIdGroupPresetExtensions) {		
mae_numDownmixIdGroupPresetExtensions[gp];	5	**uimsbf**
for (egp = 0; egp < mae_numDownmixIdGroupPresetExtensions[gp]; egp++) {		
mae_groupPresetDownmixId[gp][egp];	7	**uimsbf**
mae_bsGroupPresetNumConditions[gp][egp]**;**	4	**uimsbf**
for (cnd = 0; cnd < mae_bsGroupPresetNumConditions[gp][egp] + 1; cnd++) {		
mae_isSwitchGroupCondition[gp][egp][cnd];	1	**bslbf**
if (mae_isSwitchGroupCondition[gp][epg][cnd]) {		
mae_groupPresetSwitchGroupID[gp][egp][cnd];	5	**uimsbf**
} else {		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
                        mae_groupPresetGroupID[gp][egp][cnd];                7      uimsbf
                    }
                    mae_groupPresetConditionOnOff[gp][egp][cnd];             1      bslbf
                    if (mae_groupPresetConditionOnOff[gp][egp][cnd] {
                                                                             1      bslbf
    mae_groupPresetDisableGainInteractivity[gp][egp][cnd];
                        mae_groupPresetGainFlag[gp][egp][cnd];               1      bslbf
                        if ( mae_groupPresetGainFlag[gp][egp][cnd]) {
                            mae_groupPresetGain[gp][egp][cnd];               8      uimsbf
                        }
                                                                             1      bslbf
    mae_groupPresetDisablePositionInteractivity[gp][egp][cnd];
                        mae_groupPresetPositionFlag[gp][egp][cnd];           1      bslbf
                        if ( mae_groupPresetPositionFlag [gp][egp][cnd] ) {
                            mae_groupPresetAzOffset[gp][egp][cnd];           8      uimsbf
                            mae_groupPresetElOffset[gp][egp][cnd];           6      uimsbf
                            mae_groupPresetDistFactor[gp][egp][cnd];         4      uimsbf
                        }
                    }
                }
            }
        }
    }
}
```

In ***15.3*** *replace*

mae_numGroupPresets	Number of defined group presets. This field can take values between 0 and 31, resulting in a maximum number of 31 group presets. A group preset is a subset of all groups of an overall scene where the on/off status of each of these groups is bound to a condition. With group presets it is possible to define a controlled behavior in dependence of the on/off status of some of the groups in an overall audio scene. A group preset is valid if all its associated conditions are true (logical AND of the conditions yields 1), i.e. if the on/off status of all associated groups is conform to the defined conditions.

with

mae_numGroupPresets	Number of defined group presets. This field can take values between 0 and 31, resulting in a maximum number of 31 group presets. A group preset is a subset of all groups of an overall scene where the on/off status of each of these groups is bound to a condition. With group presets it is possible to define a controlled behavior in dependence of the on/off status of some of the groups and/or switch groups in an overall audio scene. A group preset is valid if all its associated conditions are true (logical AND of the conditions yields 1), i.e. if the on/off status of all associated groups and the on/off status of each associated switch group is conform to the defined conditions. A condition associated with a switch group with value 1 is true if one member of the switch group is switched on. A condition associated with a switch group with value 0 is true if all members of the switch group are switched off. Switch group conditions with value 0 are only applicable for switch groups whose mae_switchGroupAllowOnOff flag is equal to 1.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

*In **15.3** replace*

mae_numGroupPresets	Number of defined group presets. This field can take values between 0 and 31, resulting in a maximum number of 31 group presets. A group preset is a subset of all groups of an overall scene where the on/off status of each of these groups is bound to a condition. With group presets it is possible to define a controlled behavior in dependence of the on/off status of some of the groups in an overall audio scene. A group preset is valid if all its associated conditions are true (logical AND of the conditions yields 1), i.e. if the on/off status of all associated groups is conform to the defined conditions. An example is a group preset with the associated groups mae_groupID$_1$ and mae_groupID$_3$. This group preset gets valid if the group with mae_groupID$_1$ is switched on and the group with mae_groupID$_3$ is switched off: validity(group preset) = ((status(mae_groupID$_1$)==on) && !(status(mae_groupID$_3$)==on)) The validity status of a group preset can be connected to a special treatment, e.g. it can be connected to a defined DRC sequence which is only applied if the group preset is valid.

with

mae_numGroupPresets	Number of defined group presets. This field can take values between 0 and 31, resulting in a maximum number of 31 group presets. A group preset is a subset of all groups of an overall scene where the on/off status of each of these groups is bound to a condition. With group presets it is possible to define a controlled behavior in dependence of the on/off status of some of the groups in an overall audio scene. A group preset is valid if all its associated conditions are true (logical AND of the conditions yields 1), i.e. if the on/off status of all associated groups is conform to the defined conditions. An example is a group preset with the associated groups mae_groupID$_1$ and mae_groupID$_3$. This group preset gets valid if the group with mae_groupID$_1$ is switched on and the group with mae_groupID$_3$ is switched off: validity(group preset) = ((status(mae_groupID$_1$)==on) && !(status(mae_groupID$_3$)==on)) The validity status of a group preset can be connected to a special treatment, e.g. it can be connected to a defined DRC sequence which is only applied if the group preset is valid. Any evaluation of the valid/selected preset (and therefore the application of preset-dependent values) shall only happen in the defined 'basic interaction mode' (see 17.7.3). The chosen/selected preset is indicated by the presetID that is received via the mae_elementInteraction() interface.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Further replace the semantics of mae_bsGroupPresetNumConditions , mae_groupPresetReferenceID, mae_groupPresetConditionOnOff, mae_groupPresetDisableGainInteractivity, mae_groupPresetGainFlag, mae_groupPresetGain, mae_groupPresetDisablePositionInteractivity, mae_groupPresetPositionFlag, mae_groupPresetAzOffset and mae_groupPresetDistFactor

with

mae_bsGroupPresetNumConditions

This field defines the number of group conditions that are associated with a group preset. The field takes values between 0 and 15; a minimum of 1 condition and a maximum of 16 conditions are assumed. A condition is a combination of a mae_groupID or a mae_switchGroupID and an on/off status.

mae_groupPresetReferenceID

This field specifies the groups or switch groups associated with a group preset. By default, this reference is interpreted as a groupID. The reference can be defined to be interpreted as a switchGroupID by extension metadata.

mae_groupPresetConditionOnOff

This flag describes the required on/off status of a group associated with a group preset. If the flag is 1, the associated group has to be switched on to validate the group preset. If the flag is 0, the associated group has to be switched off.

If the referenced ID is defined to be interpreted as a switchGroupID, a groupPresetConditionOnOff with value 1 means that one member of the switch group has to be switched on to validate the group preset.

mae_groupPresetDisableGainInteractivity

This field defines if the gain interactivity of the currently referenced group or the members of the referenced switch group shall be disabled (flag is equal to 1) or shall stay enabled (flag is equal to 0) if the preset is chosen/valid.

mae_groupPresetGainFlag This field defines whether the corresponding preset specifies an initial gain of the members of a metadata element group or of the element members of the group members of the referenced switch group. It shall only be 1 if the flag mae_allowGainInteractivity of the corresponding group or of all the members of the referenced switch group is set to 1.

mae_groupPresetGain The field defines the initial gain of the members of a metadata element group or of the element members of the group members of the referenced switch group when the corresponding preset is selected.

groupPresetGain in dB = 0.5 · (**mae_groupPresetGain** – 255) + 32;

mae_groupPresetDisablePositionInteractivity

This field defines if the position interactivity of the currently referenced group or of the members of the referenced switch group shall be disabled (flag is equal to 1) or shall stay enabled (flag is equal to 0) if the preset is chosen/valid.

mae_groupPresetPositionFlag

This field defines whether initial position interactivity data (azimuth offset, elevation offset and distance factor) is present that shall be applied to the

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

	members of a metadata element group or to the element members of the group members of the referenced switch group. It shall only be 1 if the flag mae_allowPositionInteractivity of the corresponding group or of all the members of the referenced switch group is set to 1.
mae_groupPresetAzOffset	This field defines the additional azimuth offset that shall be applied to the currently referenced group or the members of the referenced switch group if the preset is chosen/valid. This field can take values between PresetAdditionalAzOffset = -180° and AzOffset = +180°: PresetAdditionalAzOffset =1.5 · (**mae_groupPresetAzOffset** - 127)
mae_groupPresetElOffset	This field defines the additional elevation offset that shall be applied to the currently referenced group or the members of the referenced switch group if the preset is chosen/valid. This field can take values between PresetAdditionalElOffset = -90° and ElOffset = +90°: PresetAdditionalElOffset = 3 · (**mae_groupPresetElOffset** - 31)
mae_groupPresetDistFactor	This field defines the additional distance change factor that shall be applied to the currently referenced group or the members of the referenced switch group if the preset is chosen/valid. This field can take values between 0 and 15 resulting in PresetAdditionalDistFactor between 0.00025 and 8: PresetAdditionalDistFactor = $2^{(\text{mae\_groupPresetDistFactor}-12)}$;

Add the following description of semantics at the end of 15.3

mae_overwriteProductionScreenSizeData
This field defines if the bitstream contains azimuth values for a non-centered default production screen. If this flag is set to 1, the following azimuth values shall be used in the processing instead of the values from mae_ProductionScreenSizeData().

bsScreenSizeLeftAz This field defines the azimuth corresponding to the left screen edge:

$$\varphi_{\text{left}}^{\text{nominal}} = 0.5 \cdot (\mathbf{bsScreenSizeLeftAz} - 511);$$

$$\varphi_{\text{left}}^{\text{nominal}} = \min(\max(\varphi_{\text{left}}^{\text{nominal}}, -180), 180);$$

bsScreenSizeRightAz This field defines the azimuth corresponding to the right screen edge:

$$\varphi_{\text{right}}^{\text{nominal}} = 0.5 \cdot (\mathbf{bsScreenSizeRightAz} - 511);$$

$$\varphi_{\text{right}}^{\text{nominal}} = \min(\max(\varphi_{\text{right}}^{\text{nominal}}, -180), 180);$$

mae_bsNumPresetProductionScreens
This field defines the number of preset-associated production screens.

mae_productionScreenGroupPresetID
This field defines the presetID the current production screen is associated with.

mae_hasNonStandardScreenSize
This field defines if the bitstream contains a non-standard preset-associated production screen size. If the flag is one, the non-standard production screen size information follows in the bitstream.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

isCenteredInAzimuth	This flag defines if the production screen is centered in azimuth (absolute values of the azimuth angles of the left and right screen edge are identical) or not.
mae_hasSwitchGroupCondition	This flag defines if a group preset has switch group conditions (flag is equal to 1).
mae_isSwitchGroupCondition	This field defines if the condition from original preset definition references a groupID (mae_isSwitchGroupCondition is equal to 0) or if the referenced ID shall be interpreted as a switchGroupID (mae_isSwitchGroupCondition is equal to 1).
mae_hasDownmixIdGroupPresetExtensions	This flag defines if a group preset has layout-dependent extensions (flag is equal to 1). Group presets can be extended by group presets extensions, which are applicable for a specific downmixId. These extensions can overwrite the preset conditions and other preset characteristics, e.g. the group preset gain. If a preset is selected and there is a current downmixId, the conditions and characteristics of the appropriate group preset extension shall replace the corresponding values from the chosen/valid group preset in the processing and rendering.
mae_groupPresetDownmixId	This field references a downmixId, for which the current group preset extension is applicable.
mae_groupPresetSwitchGroupID	This field contains a reference to a switchGroupID that is used in a switch group condition.

9.2 Update of OAM data transmission and processing

9.2.1 OAM syntax and semantics

*In **7.2.** replace*

Metadata is conveyed for every audio object as given spatial positions (azimuth, elevation, and radius) and a linear gain at defined timestamps.

with

Metadata is conveyed for every audio object as given spatial positions (azimuth, elevation, and radius) and a linear gain at defined timestamps. In addition to that, either a uniform spread value of three distinct spread values (spread in width, height and depth dimension), as well as a dynamic object priority value can be defined.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

*and update **Table 68** as follows:*

Table 68 — Units of the decoded object metadata components

component	unit	value range
azimuth	° (degree)	-180; 180
elevation	° (degree)	-90; 90
radius	m (meter)	0.5; 16
gain	none (linear)	0.004; 5.957
spread (uniform)	° (degree)	0; 180
spread width	° (degree)	0; 180
spread height	° (degree)	0; 90
spread depth	m (meter)	0; 15.5
dynamic object priority	none	0; 7

*In **7.3.1** update **Table 69** as follows:*

Table 69 — Syntax of ObjectMetadataConfig()

```
Syntax                                                      No. of bits   Mnemonic
ObjectMetadataConfig()
{
    lowDelayMetadataCoding;                                 1             bslbf
    hasCoreLength;                                          1             bslbf
    if (!hasCoreLength) {
        frameLength;                                        6             uimsbf
        OAMFrameLength = (frameLength+1)<<6;
    } else {
        OAMFrameLength = outputFrameLength;
    }
    hasScreenRelativeObjects;                               1             bslbf
    if( hasScreenRelativeObjects ) {
        for ( o = 0; o < num_objects; o++ ) {   /* NOTE 1 */
            isScreenRelativeObject[o];                      1             bslbf
        }
    }
    hasDynamicObjectPriority;                               1             bslbf
    hasUniformSpread;                                       1             bslbf
}
```

NOTE 1: num_objects is equal to the number of objects in the associated signal group.

*Add the following table in front of former **Table 70** in **7.3.1**.*

Table AMD7.6– Syntax of objectMetadataFrame()

```
Syntax                                                      No. of bits   Mnemonic
objectMetadataFrame() {
    if (OAMFrameLength* < coreCoderFrameLength**) {
        for ( i = 0; i < coreCoderFrameLength / OAMFrameLength; ++i ) {
            object_metadata_present;                        1             uimsbf
            if (object_metadata_present)
                object_metadata();
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
    }
    else {
        object_metadata();
    }
}
```
* given by ObjectMetadataConfig()
** given by mpegh3daConfig()

*Further, add the following text to the end of **7.3.1**:*

The coreCoderFrameLength (given by mpegh3daConfig()) shall be an integer multiple of the OAMFrameLength. The OAMFrameLength shall not be greater than the coreCoderFrameLength.
If the coreCoderFrameLength is an integer multiple of the OAMFrameLength (coreCoderFrameLength $= n \cdot$ OAMFrameLength), the structure object_metadata() is sent n times in the objctMetadataFrame() as shown in Table 70. In this case, the i^{th} object_metadata() ($i = 0,1,2,..,n\text{-}1$) is available at the sample index $s[i]$, with $s[i]$ given by the following equation

$$s[i] = (i + 1) \cdot \text{OAMFramelength-1}$$

wheres is the sample index of an audio frame (0,1,2,.., $n \cdot \text{OAMFramelength-1}$).

*Update **Table 72** in **7.3.2** as follows:*

Table 72 — Syntax of intracoded_object_metadata_efficient()

Syntax	No. of bits	Mnemonic
intracoded_object_metadata_efficient()		
{		
ifperiod;	6	uimsbf
if (num_objects>1) {		
common_azimuth;	1	bslbf
if (common_azimuth) {		
default_azimuth;	8	tcimsbf
}		
else {		
for (o = 0; o < num_objects; o++) {		
position_azimuth[o];	8	tcimsbf
}		
}		
common_elevation;	1	bslbf
if (common_elevation) {		
default_elevation;	6	tcimsbf
}		
else {		
for (o = 0; o < num_objects; o++) {		
position_elevation[o];	6	tcimsbf
}		
}		
common_radius;	1	bslbf
if (common_radius) {		
default_radius;	4	uimsbf
}		
else {		
for (o = 0; o < num_objects; o++) {		
position_radius[o];	4	uimsbf
}		
}		
common_gain;	1	bslbf
if (common_gain) {		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
            default_gain;                                   7       tcimsbf
        }
        else {
            for (o = 0; o < num_objects; o++) {
                gain_factor[o];                             7       tcimsbf
            }
        }
        common_spread;                                      1       bslbf
        if (common_spread) {
            if (hasUniformSpread) {
                default_spread;                             7       uimsbf
            }
            else {
                default_spread_width;                       7       uimsbf
                default_spread_height;                      5       uimsbf
                default_spread_depth;                       4       uimsbf
            }
        }
        else {
            for (o = 0; o < num_objects; o++) {
                if (hasUniformSpread) {
                    spread[o];                              7       uimsbf
                }
                else {
                    spread_width[o];                        7       uimsbf
                    spread_height[o];                       5       uimsbf
                    spread_depth[o];                        4       uimsbf
                }
            }
        }
        if (hasDynamicObjectPriority) {
            common_dynamic_object_priority;                 1       bslbf
            if (common_dynamic_object_priority) {
                default_dynamic_object_priority;            3       uimsbf
            }
            else {
                for (o = 0; o < num_objects; o++) {
                    dynamic_object_priority[o];             3       uimsbf
                }
            }
        }
    }
    else {
        position_azimuth;                                   8       tcimsbf
        position_elevation;                                 6       tcimsbf
        position_radius;                                    4       uimsbf
        gain_factor;                                        7       tcimsbf
        if (hasUniformSpread) {
            spread;                                         7       uimsbf
        }
        else {
            spread_width;                                   7       uimsbf
            spread_height;                                  5       uimsbf
            spread_depth;                                   4       uimsbf
        }
        if (hasDynamicObjectPriority) {
            dynamic_object_priority;                        3       uimsbf
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
    }
}
```

*Update **Table 73** in **7.3.3** as follows:*

Table 73 — Syntax of differential_object_metadata()

Syntax	No. of bits	Mnemonic
differential_object_metadata() {		
bits_per_point;	**4**	**uimsbf**
fixed_azimuth;	**1**	**bslbf**
if (!fixed_azimuth) {		
for (o = 0; o < num_objects; o++) {		
flag_azimuth;	**1**	**bslbf**
if (flag_azimuth) {		
num_points_azimuth = offset_data(bits_per_point);		
nbits_azimuth;	**3**	**uimsbf**
for (p = 0; p < num_points_azimuth; p++) {		
differential_azimuth[o][p];	**nbits_azimuth + 2**	**tcimsbf**
}		
}		
}		
}		
fixed_elevation;	**1**	**bslbf**
if (!fixed_elevation) {		
for (o = 0; o < num_objects; o++) {		
flag_elevation;	**1**	**bslbf**
if (flag_elevation) {		
num_points_elevation = offset_data(bits_per_point);		
nbits_elevation;	**3**	**uimsbf**
for (p = 0; p < num_points_elevation; p++) {		
differential_elevation[o][p];	**nbits_elevation + 2**	**tcimsbf**
}		
}		
}		
}		
fixed_radius;	**1**	**bslbf**
if (!fixed_radius) {		
for (o = 0; o < num_objects; o++) {		
flag_radius;	**1**	**bslbf**
if (flag_radius) {		
num_points_radius = offset_data(bits_per_point);		
nbits_radius;	**3**	**uimsbf**
for (p = 0; p < num_points_radius; p++) {		
differential_radius[o][p];	**nbits_radius + 2**	**tcimsbf**
}		
}		
}		
}		
fixed_gain;	**1**	**bslbf**
if (!fixed_gain) {		
for (o = 0; o < num_objects; o++) {		
flag_gain;	**1**	**bslbf**
if (flag_gain) {		
num_points_gain = offset_data(bits_per_point);		
nbits_gain;	**3**	**uimsbf**
for (p = 0; p < num_points_gain; p++) {		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
                differential_gain[o][p];                        nbits_gain + 2   tcimsbf
            }
        }
    }
}
fixed_spread;                                                   1                bslbf
if (!fixed_spread) {
    for (o = 0; o < num_objects; o++) {
        if (hasUniformSpread) {
            flag_spread;                                        1                bslbf
            if (flag_spread) {
                num_points_spread = offset_data(bits_per_point);
                nbits_spread;                                   3                uimsbf
                for (p = 0; p < num_points_spread; p++) {
                    differential_spread[o][p];                  nbits_spread +   tcimsbf
                                                                2
                }
            }
        }
        else {
            flag_spread_width;                                  1                bslbf
            if (flag_spread_width) {
                num_points_spread_width = offset_data(bits_per_point);
                nbits_spread_width;                             3                uimsbf
                for (p = 0; p < num_points_spread_width; p++) {
                    differential_spread_width[o][p];        nbits_spread_width   tcimsbf
                                                            + 2
                }
            }
            flag_spread_height;                                 1                bslbf
            if (flag_spread_height) {
                num_points_spread_height = offset_data(bits_per_point);
                nbits_spread_height;                            3                uimsbf
                for (p = 0; p < num_points_spread_height; p++) {
                    differential_spread_height[o][p];      nbits_spread_height   tcimsbf
                                                           + 2
                }
            }
            flag_spread_depth;                                  1                bslbf
            if (flag_spread_depth) {
                num_points_spread_depth = offset_data(bits_per_point);
                nbits_spread_depth;                             3                uimsbf
                for (p = 0; p < num_points_spread_depth; p++) {
                    differential_spread_depth[o][p];       nbits_spread_depth +  tcimsbf
                                                           2
                }
            }
        }
    }
}
if (hasDynamicObjectPriority) {
    fixed_dynamic_object_priority;                              1                bslbf
    if (!fixed_dynamic_object_priority) {
        for (o = 0; o < num_objects; o++) {
            flag_dynamic_object_priority;                       1                bslbf
            if (flag_dynamic_object_priority) {
                num_points_dynamic_object_priority = offset_data(bits_per_point);
                nbits_dynamic_object_priority;                  2                uimsbf
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

for (p = 0; p < num_points_dynamic_object_priority; p++) {		
differential_dynamic_object_priority[o][p];	**nbits_dynamic_object_priority + 2**	**tcimsbf**
}		
}		
}		
}		
}		
}		

Update ***Table 76*** *in* ***7.3.4*** *as follows:*

Table 76 — Syntax of intracoded_object_metadata_low_delay()

Syntax	No. of bits	Mnemonic
intracoded_object_metadata_low_delay()		
{		
if (num_objects>1) {		
fixed_azimuth;	**1**	**bslbf**
if (fixed_azimuth) {		
default_azimuth;	**8**	**tcimsbf**
}		
else {		
common_azimuth;	**1**	**bslbf**
if (common_azimuth) {		
default_azimuth;	**8**	**tcimsbf**
}		
else {		
for (o = 0; o < num_objects; o++) {		
position_azimuth[o];	**8**	**tcimsbf**
}		
}		
}		
fixed_elevation;	**1**	**bslbf**
if (fixed_elevation) {		
default_elevation;	**6**	**tcimsbf**
}		
else {		
common_ elevation;	**1**	**bslbf**
if (common_elevation) {		
default_elevation;	**6**	**tcimsbf**
}		
else {		
for (o = 0; o < num_objects; o++) {		
position_elevation[o];	**6**	**tcimsbf**
}		
}		
}		
fixed_radius;	**1**	**bslbf**
if (fixed_radius) {		
default_radius;	**4**	**uimsbf**
}		
else {		
common_radius;	**1**	**bslbf**
if (common_radius) {		
default_radius;	**4**	**uimsbf**
}		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
        else {
            for (o = 0; o < num_objects; o++) {
                position_ radius[o];                          4       uimsbf
            }
        }
    }
    fixed_gain;                                               1       bslbf
    if (fixed_gain) {
        default_gain;                                         7       tcimsbf
    }
    else {
        common_gain;                                          1       bslbf
        if (common_gain) {
            default_gain;                                     7       tcimsbf
        }
        else {
            for (o = 0; o < num_objects; o++) {
                gain_factor[o];                               7       tcimsbf
            }
        }
    }
    fixed_spread;                                             1       bslbf
    if (fixed_spread) {
        if (hasUniformSpread) {
            default_spread;                                   7       uimsbf
        }
        else {
            default_spread_width;                             7       uimsbf
            default_spread_height;                            5       uimsbf
            default_spread_depth;                             4       uimsbf
        }
    }
    else {
        common_spread;                                        1       bslbf
        if (common_spread) {
            if (hasUniformSpread) {
                default_spread;                               7       uimsbf
            }
            else {
                default_spread_width;                         7       uimsbf
                default_spread_height;                        5       uimsbf
                default_spread_depth;                         4       uimsbf
            }
        }
        else {
            for (o = 0; o < num_objects; o++) {
                if (hasUniformSpread) {
                    spread[o];                                7       uimsbf
                }
                else {
                    spread_width[o];                          7       uimsbf
                    spread_height[o];                         5       uimsbf
                    spread_depth[o];                          4       uimsbf
                }
            }
        }
    }
    if (hasDynamicObjectPriority) {
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

fixed_dynamic_object_priority;	**1**	**bslbf**
if (fixed_dynamic_object_priority) {		
default_dynamic_object_priority;	**3**	**uimsbf**
}		
else {		
common_dynamic_object_priority;	**1**	**bslbf**
if (common_dynamic_object_priority) {		
default_dynamic_object_priority;	**3**	**uimsbf**
}		
else {		
for (o = 0; o < num_objects; o++) {		
dynamic_object_priority[o];	**3**	**uimsbf**
}		
}		
}		
}		
}		
else {		
position_azimuth;	**8**	**tcimsbf**
position_elevation;	**6**	**tcimsbf**
position_radius;	**4**	**uimsbf**
gain_factor;	**7**	**tcimsbf**
if (hasUniformSpread) {		
spread;	**7**	**uimsbf**
}		
else {		
spread_width;	**7**	**uimsbf**
spread_height;	**5**	**uimsbf**
spread_depth;	**4**	**uimsbf**
}		
if (hasDynamicObjectPriority) {		
dynamic_object_priority;	**3**	**uimsbf**
}		
}		
}		

*Update **Table 78** in 7.3.4 as follows*

Table 78 — Syntax of single_dynamic_object_metadata()

Syntax	No. of bits	Mnemonic
single_dynamic_object_metadata (flag_absolute) {		
if (flag_absolute) {		
if (!fixed_azimuth*) {		
position_azimuth;	**8**	**tcimsbf**
}		
if (!fixed_elevation*) {		
position_elevation;	**6**	**tcimsbf**
}		
if (!fixed_radius*) {		
position_radius;	**4**	**uimsbf**
}		
if (!fixed_gain*) {		
gain_ factor;	**7**	**tcimsbf**
}		
if (!fixed_spread*) {		
if (hasUniformSpread) {		
spread;	**7**	**uimsbf**

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
            }
            else {
                **spread_width;**                               7               uimsbf
                **spread_height;**                              5               uimsbf
                **spread_depth;**                               4               uimsbf
            }
        }
        if (hasDynamicObjectPriority) {
            if (!fixed_dynamic_object_priority*) {
                **dynamic_object_priority;**                    3               uimsbf
            }
        }
    }
    else {
        **nbits;**                                              3               uimsbf
        num_bits = nbits +2;
        if (!fixed_azimuth*) {
            **flag_azimuth;**                                   1               bslbf
            if (flag_azimuth)  {
                **position_azimuth_difference;**                num_bits        tcimsbf
            }
        }
        if (!fixed_elevation*) {
            **flag_elevation;**                                 1               bslbf
            if (flag_elevation) {
                **position_elevation_difference;**              min(num_bits,7) tcimsbf
            }
        }
        if (!fixed_radius*) {
            **flag_radius;**                                    1               bslbf
            if (flag_radius)  {
                **position_radius_difference;**                 min(num_bits,5) tcimsbf
            }
        }
        if (!fixed_gain*) {
            **flag_gain;**                                      1               bslbf
            if (flag_gain)  {
                **gain_factor_difference ;**                    min(num_bits,8) tcimsbf
            }
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
        if (!fixed_spread*) {
            if (hasUniformSpread) {
                flag_spread;                              1               bslbf
                if (flag_spread) {
                    spread_difference ;                   min(num_bits,8)  tcimsbf
                }
            }
            else {
                flag_spread_width;                        1               bslbf
                if (flag_spread_width) {
                    spread_width_difference ;             min(num_bits,8)  tcimsbf
                }
                flag_spread_height;                       1               bslbf
                if (flag_spread_height) {
                    spread_height_difference ;            min(num_bits,6)  tcimsbf
                }
                flag_spread_depth;                        1               bslbf
                if (flag_spread_depth) {
                    spread_depth_difference ;             min(num_bits,5)  tcimsbf
                }
        }
        if (hasDynamicObjectPriority) {
            if (!fixed_dynamic_object_priority*) {
                flag_dynamic_object_priority;             1               bslbf
                if (flag_dynamic_object_priority) {
                    dynamic_object_priority_difference;   min(num_bits,4)  tcimsbf
                }
            }
        }
    }
}
* Given by the preceding intracoded_object_metadata_low_delay() frame
```

*In **7.4.1** after the semantics of **hasDynamicObjectPriority** add*

hasUniformSpread	This flag indicates whether the spread of an object is given as uniform spread (flag is equal to 1) or as three independent values for width, height and depth (flag is equal to 0).

*In **7.4.2.1.2** replace*

default_spread	defines the value of the common spread parameter.
spread	if there is no common spread parameter, a value for each object is transmitted. If there is only one object, this is its spread parameter.

with

default_spread	Defines the value of the common spread parameter for the case of one uniform spread value.
default_spread_width	Defines the value of the common spread parameter for the spread in the dimension of width for the case of three independent spread values.
default_spread_height	Defines the value of the common spread parameter for the spread in the dimension of height for the case of three independent spread values.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

default_spread_depth	Defines the value of the common spread parameter for the spread in the dimension of depth for the case of three independent spread values.
spread	If there is no common spread parameter, a value for each object is transmitted. If there is only one object, this is its spread parameter.
spread_width	If there is no common spread parameter, one value for the spread in the dimension of width is transmitted for each object. If there is only one object, this is its spread parameter in the dimension of width.
spread_height	If there is no common spread parameter, one value for the spread in the dimension of height is transmitted for each object. If there is only one object, this is its spread parameter in the dimension of height.
spread_depth	If there is no common spread parameter, one value for the spread in the dimension of depth is transmitted for each object. If there is only one object, this is its spread parameter in the dimension of depth.

In 7.4.2.1.3 after differential_spread add

flag_spread_width	flag per object indicating whether the spread parameter in width dimension changes for this iframe_period.
nbits_spread_width	how many bits are required to represent the differential value minus 2.
differential_spread_width	value of the difference between the linearly interpolated and the actual value.
flag_spread_height	flag per object indicating whether the spread parameter in height dimension changes for this iframe_period.
nbits_spread_height	how many bits are required to represent the differential value minus 2.
differential_spread_height	value of the difference between the linearly interpolated and the actual value.
flag_spread_depth	flag per object indicating whether the spread parameter in depth dimension changes for this iframe_period.
nbits_spread_depth	how many bits are required to represent the differential value minus 2.
differential_spread_depth	value of the difference between the linearly interpolated and the actual value.

In 7.4.2.4.1 and in 7.4.3.2.3.1 replace

```
descale_multidata()
{
    for (o = 0; o < num_objects; o++)
        azimuth[o] = azimuth[o] * 1.5;

    for (o = 0; o < num_objects; o++)
        elevation[o] = elevation[o] * 3.0;

    for (o = 0; o < num_objects; o++)
        radius[o] = pow(2.0, (radius[o] / 3.0)) / 2.0;

    for (o = 0; o < num_objects; o++)
        gain[o] = pow(10.0, (gain[o] - 32.0) / 40.0);

    for (o = 0; o < num_objects; o++)
        spread[o] = spread[o] * 1.5;
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

with

```
descale_multidata()
{
   for (o = 0; o < num_objects; o++)
      azimuth[o] = azimuth[o] * 1.5;

   for (o = 0; o < num_objects; o++)
      elevation[o] = elevation[o] * 3.0;

   for (o = 0; o < num_objects; o++)
      radius[o] = pow(2.0, (radius[o] / 3.0)) / 2.0;

   for (o = 0; o < num_objects; o++)
      gain[o] = pow(10.0, (gain[o] - 32.0) / 40.0);

   if (uniform_spread == 1)
   {
      for (o = 0; o < num_objects; o++)
         spread[o] = spread[o] * 1.5;
   }
   else
   {
      for (o = 0; o < num_objects; o++)
         spread_width[o] = spread_width[o] * 1.5;

      for (o = 0; o < num_objects; o++)
         spread_height[o] = spread_height[o] * 3.0;

      for (o = 0; o < num_objects; o++)
         spread_depth[o] = (pow(2.0, (spread_depth[o] / 3.0)) / 2.0) – 0.5;
   }
   for (o = 0; o < num_objects; o++)
      dynamic_object_priority[o] = dynamic_object_priority[o];
}
```

In 7.4.2.4.2 and in 7.4.3.2.3.2 replace

```
limit_range()
{
   minval = -180;
   maxval =  180;
   for (o = 0; o < num_objects; o++)
      azimuth[o] = MIN(MAX(azimuth[o], minval), maxval);

   minval = -90;
   maxval =  90;
   for (o = 0; o < num_objects; o++)
      elevation[o] = MIN(MAX(elevation[o], minval), maxval);

   minval =    0.5;
    maxval = 16;
   for (o = 0; o < num_objects; o++)
      radius[o] = MIN(MAX(radius[o], minval), maxval);

   minval =    0.004;
   maxval = 5.957;
   for (o = 0; o < num_objects; o++)
      gain[o] = MIN(MAX(gain[o], minval), maxval);

   minval =    0;
   maxval =  180;
   for (o = 0; o < num_objects; o++)
      spread[o] = MIN(MAX(spread[o], minval), maxval);
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

with

```
limit_range()
{
   minval = -180;
   maxval =  180;
   for (o = 0; o < num_objects; o++)
      azimuth[o] = MIN(MAX(azimuth[o], minval), maxval);

   minval = -90;
   maxval =  90;
   for (o = 0; o < num_objects; o++)
      elevation[o] = MIN(MAX(elevation[o], minval), maxval);

   minval =    0.5;
   maxval = 16;
   for (o = 0; o < num_objects; o++)
      radius[o] = MIN(MAX(radius[o], minval), maxval);

   minval =    0.004;
   maxval = 5.957;
   for (o = 0; o < num_objects; o++)
      gain[o] = MIN(MAX(gain[o], minval), maxval);

   if (uniform_spread == 1)
   {
      minval =    0;
      maxval =  180;
      for (o = 0; o < num_objects; o++)
         spread[o] = MIN(MAX(spread[o], minval), maxval);
   }
   else
   {
      minval =    0;
      maxval =  180;
      for (o = 0; o < num_objects; o++)
         spread_width[o] = MIN(MAX(spread_width[o], minval), maxval);

      minval =   0;
      maxval =  90;
      for (o = 0; o < num_objects; o++)
         spread_height[o] = MIN(MAX(spread_height[o], minval), maxval);

      minval =   0;
      maxval =  15.5;
      for (o = 0; o < num_objects; o++)
         spread_depth[o] = MIN(MAX(spread_depth[o], minval), maxval);
   }
   minval = 0;
   maxval = 7;
   for (o = 0; o < num_objects; o++)
      dynamic_object_priority[o] = MIN(MAX(dynamic_object_priority[o], minval),
       maxval);
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In 7.4.3.1.2 after default_spread add

default_spread_width	defines the value of the fixed or common spread parameter in width dimension.
default_spread_height	defines the value of the fixed or common spread parameter in height dimension.

© ISO/IEC 2017 – All rights reserved

default_spread_depth	defines the value of the fixed or common spread parameter in depth dimension.

In 7.4.3.1.2 after spread add

spread_width	if there is no common spread parameter, a value for the spread in width dimension is transmitted for each object. If there is only one object, this is its spread parameter in width dimension.
spread_height	if there is no common spread parameter, a value for the spread in height dimension is transmitted for each object. If there is only one object, this is its spread parameter in height dimension.
spread_depth	if there is no common spread parameter, a value for the spread in depth dimension is transmitted for each object. If there is only one object, this is its spread parameter in depth dimension.

In 7.4.3.1.4 after spread add

spread_width	the absolute value of the spread parameter in width dimension if the value is not fixed.
spread_height	the absolute value of the spread parameter in height dimension if the value is not fixed.
spread_depth	the absolute value of the spread parameter in depth dimension if the value is not fixed.

In 7.4.3.1.4 after spread_difference add

flag_spread_width	flag per object indicating whether the spread parameter in width dimension changes.
spread_width_difference	difference between the previous and the active value.
flag_spread_height	flag per object indicating whether the spread parameter in height dimension changes.
spread_height_difference	difference between the previous and the active value.
flag_spread_depth	flag per object indicating whether the spread parameter in depth dimension changes.
spread_depth_difference	difference between the previous and the active value.

*In **C.3.1** replace*

```
for (o = 0; o < num_objects; o++)
    azimuth_scaled[o][n] = azimuth[o][n] / 1.5f;

for (o = 0; o < num_objects; o++)
    elevation_scaled[o][n] = elevation[o][n] / 3.0f;

for (o = 0; o < num_objects; o++)
    radius_scaled[o][n] = 3.0f * log2(2.0 * radius[o][n]);

for (o = 0; o < num_objects; o++)
    gain_scaled[o][n] = 2.0f * db(gain[o][n]) + 32.0f;
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

with

```
for (o = 0; o < num_objects; o++)

    azimuth_scaled[o][n] = azimuth[o][n] / 1.5f;

for (o = 0; o < num_objects; o++)

    elevation_scaled[o][n] = elevation[o][n] / 3.0f;

for (o = 0; o < num_objects; o++)

    radius_scaled[o][n] = 3.0f * log2(2.0 * radius[o][n]);

for (o = 0; o < num_objects; o++)

        for (o = 0; o < num_objects; o++)][n]) + 32.0f;
            spread_scaled[o] = spread[o] / 1.5f;
    }
    else
    {
        for (o = 0; o < num_objects; o++)
            spread_width_scaled[o] = spread_width[o] / 1.5f;

        for (o = 0; o < num_objects; o++)
            spread_height_scaled[o] = spread_height[o] / 3.0f;

        for (o = 0; o < num_objects; o++)
            spread_depth_scaled[o] = 3.0f * log2(2.0 * (spread_depth[o] + 0.5f);
    }

for (o = 0; o < num_objects; o++)
    dynamic_object_priority_scaled[o][n] = dynamic_object_priority_scaled[o][n];
```

9.2.2 2D spread rendering

*In **8.3** replace*

— Spread parameter α ∈ $[0°, 180°]$.

with

Uniform spread parameter $\alpha \in [0°, 180°]$.

*In **8.4.3.7** remove the following text*

— the local loudspeaker setup is signalled in the LoudspeakerRendering(), and

— the speakerLayoutType is 0, and

— the CICPspeakerLayoutIdx being one of the following values: 4, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18

and remove

If at least one of the conditions is not met, the spread processing specified in the following text is not applied.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Replace:

Audio objects with a spatial extent are processed by means of the Multiple Direction Amplitude Panning (MDAP) method[1]. This method involves the computation of a set of panning gains $g_{\mathrm{scaled,m}}$ for $M = 18$ MDAP directions p_m around the panning direction $p_0 = \hat{p}$.

with:

Audio objects with a spatial extent (spread α > 0.0° (uniform spread) or spread_width α_{width} > 0.0° (non-uniform spread)) are processed by means of the Multiple Direction Amplitude Panning (MDAP) method[2]. This method involves the computation of a set of panning gains $g_{\mathrm{scaled,m}}$ for $M = 18$ MDAP directions p_m around the panning direction $p_0 = \hat{p}$.

*Further, in **8.4.3.7** after* $v = [0,\ldots,0]^T$ *add*

If non-uniform spread values are transmitted, the ratio of the spread parameter in width direction α_{width} $\alpha_{\mathrm{width}}$and the spread parameter in height direction $\alpha_{\mathrm{height}}$are used to determine the ratio

$$a_{r=}\frac{\alpha_{\mathrm{height}}}{\alpha_{\mathrm{width}}}$$

This ratio is used to scale the base vector *v:*

$$v = v \cdot a_r \, .$$

Further replace

Together with the spread parameter α, this yields the MDAP directions p_m:

$$p_m = p_m' + \frac{p_0}{\tan(\alpha')}$$

Where α' is equal to α limited to [0.001°,90°]$[0.001°, 90°]$. The normalization to unit norm is not necessary since the normalization to unit norm is performed when $g_{\mathrm{scaled,m}}$ is computed for the MDAP directions p_m.

If the spread parameter α exceeds 90°, then a cross-fading towards power-normalized unit gain$1/\sqrt{N}$ for all of the N loudspeakers is applied to yield the final panning gains:

$$\lambda = \frac{\alpha\text{-}90°}{90°}.$$

with

Together with the spread parameter α (uniform spread) or the spread parameter α_{width} (non-uniform spread), this yields the MDAP directions p_m:

$$p_m = p_m' + \frac{p_0}{\tan(\alpha')}$$

Where α' is equal to α limited to$[0.001°, 90°]$, or respectively to α_{width} limited to $[0.001°, 90°]$ for the transmission of non-uniform spread. The normalization to unit norm is not necessary since the normalization to unit norm is performed when $g_{\mathrm{scaled,m}}$ is computed for the MDAP directions p_m.

1) Pulkki, Ville, “Generic panning tools for MAX/MSP”, Proceedings of International Computer Music Conference, 2000

2) Pulkki, Ville, “Generic panning tools for MAX/MSP”, Proceedings of International Computer Music Conference, 2000

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

If the spread parameter α (or respectively α_{width} for non-uniform spread) exceeds 90°, then a cross-fading towards power-normalized unit gain $1/\sqrt{\mathrm{N}}$ for all of the N loudspeakers is applied to yield the final panning gains:

$\lambda = \frac{\alpha\text{-}90°}{90°}$, or $\lambda = \frac{\alpha_{\text{width}}\text{-}90°}{90°}$ (non-uniform spread) respectively.

9.2.3 Informative distance and depth spread rendering

*Add a new **Annex K***

Annex K

(informative)

Distance and depth spread rendering

This annex describes an informative approach to distance and depth spread rendering.

Distance rendering:

The approach for distance rendering uses methods of signal processing to render objects with a defined distance that can be nearer or farer away than the original loudspeaker distance [4].

Therefore, the following mapping of OAM radius values to a distance factor shall be applied:

— The reference distance, r_{ref}, is assumed to be the maximum local loudspeaker distance signaled in the localSetupInformationInterface(). If no loudspeaker distances are given, a maximum distance of 1023cm shall be assumed.

— The *doubling factor* d is defined as

$$d = \begin{cases} \log_2\left(\frac{r}{r_{ref}}\right) & r > r_{ref} \\ \log_2\left(\frac{r_{ref}}{r}\right) & r < r_{ref} \end{cases}$$

— If r is equal to r_{ref},then $d = 0$.

After determination of d, the original object is copied to the positions $az + 30°$ and $az - 30°$.

For $r > r_{ref}$, the copied object signals are decorrelated with decorrelation filters (individual filters), e.g. according to [1]. This results in a lowering of phase coherence and interaural coherence.

The gains of the two copies objects, as well as the gain of the original object, are then adjusted based on the doubling factor d.

For $r > r_{ref}$: $\mathrm{d}_1 = \mathrm{d}$ (with 0 = original LS distance and 1 = doubled distance)

- Gain of the original object $g_{\text{dry}} = \max(0, 1 - 0.5 \cdot d_1)$
- Gain of the copied objects $g_{\text{dec}} = 0.25 \cdot d_1$
- In addition to that, a loudness normalization may be applied:
    - o $g_{\text{norm}} = \sqrt{g_{\text{dry}}^{\;2} + 2 \cdot g_{\text{dec}}^{\;2}}$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

- o $g_{dry}' = \frac{g_{dry}}{g_{norm}} g_{dry}' = \frac{g_{dry}}{g_{norm}}$
- o $g_{dec}' = \frac{g_{dec}}{g_{norm}} g_{dec}' = \frac{g_{dec}}{g_{norm}}$

For $r < r_{ref}$: $d_2 = \min(d, 1)$ (with 0 = original LS distance and 1 = half the original distance)

— A specific gain ratio implies the perception of “nearness”

— Gain of the original object $g_{dry} = 1$

— Gain of the copied objects $g_{corr} = \left(\frac{2}{2\text{-}d2}\text{-}1\right) \cdot 0.6$

$$g_{corr} = \begin{cases} \left(\dfrac{2}{2-d_2} - 1\right) \cdot 0.6 & d_2 \le 1 \\ \min(3.25,\ 0.8 \cdot \log_2(d_2) + 0.6 & d_2 > 1 \end{cases}$$

— In addition to that, a loudness normalization may be applied:

- o $g_{norm} = \sqrt{g_{dry}{}^2 + 2 \cdot g_{corr}{}^2}$
- o $g_{dry}' = \frac{g_{dry}}{g_{norm}}$
- o $g_{dec}' = \frac{g_{dec}}{g_{norm}} g_{corr}' = \frac{g_{corr}}{g_{norm}}$

Depth spread rendering:

With the described rendering of distance, rendering of depth spread can be realized by the means of the proposed informal distance rendering.

To make use of the depth spread value, which is transmitted in case of non-uniform spread, the corresponding object is rendered at multiple distances:

- Once at the ‘front’ of the depth expansion
- Once at the ‘back’ of the depth expansion
- Once at the original distance

The depth spread value is translated to the following ‘front’ and ‘back’ radius values:

$$r_{front} = r + \text{spread}_{depth} \cdot 0.5$$

$$r_{back} = \max(0, r - \text{spread}_{depth} \cdot 0.5)$$

The ‘front’ radius is restricted to a minimum of 0.

The rendering of 2D spread by the use of 18 additional MDAP directions shall only be applied to the object at the original distance.

9.3 Signaling and Processing of Scene Displacement Angles for CO content

*In **17** add a new subclause*

17.X Interface for scene displacement data

17.X.1 Introduction

This subclause describes syntax for a normative interface for scene displacement data received by the decoder. Use-cases are the deployment of a tracking device that tracks the user’s head (e.g. a Virtual Reality

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

(VR) device) resulting in a scene displacement that needs to be processed by the decoder framework or a tracking device that directly tracks the scene displacement (e.g. 3D mouse or data glove).

17.X.2 Definition of an interface for scene-displacement data

The mpegh3daSceneDisplacementData() syntax element provides an interface for the scene displacement, given by three Tait-Bryan angles:

- 'yaw' (α_{yaw} α_{yaw}) is a rotation around the z axis
- 'pitch' (β_{pitch}) is a rotation around the x axis
- 'roll' (θ_{roll}) is a rotation around the y axis

The three angles describe the relative orientation between two orthogonal right-handed 3D Cartesian coordinate systems by three rotation angles alongside the three coordinate axes, namely between the so-called 'fixed scene setup coordinate system' and the 'deflected scene coordinate system'.

The coordinate systems are right-handed coordinate systems that use the following direction of the axes:

- x axis pointing to the right
- y axis pointing straight ahead
- z axis pointing straight up

The fixed scene setup coordinate system is the original coordinate system, in which the positioning of loudspeakers (azimuth, elevation), reproductions screen and objects (azimuth, elevation, and radius) happens.

The deflected scene coordinate system describes the orientation of the virtual scene after the objects positions have been updated. This coordinate system rotates with the deflection of a tracker device from its idle state. The origin is assumed to be the same as the scene coordinate system, but the coordinate system may be deflected along all three axes, indicated by the values given by the scene displacement interface.

Positive rotations correspond to the direction of rotation about each axis as indicated below. For this definition, clockwise and anti-clockwise is determined by looking directly along the axis of rotation, towards the origin of the coordinate system. The positive rotations are illustrated in Figure AMD7.1.

- A **positive yaw value** corresponds to a clockwise rotation about **z axis**.
- A **positive pitch value** corresponds to an anti-clockwise rotation about the **x axis**.
- A **positive roll value** corresponds to an anti-clockwise rotation about the **y axis**.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

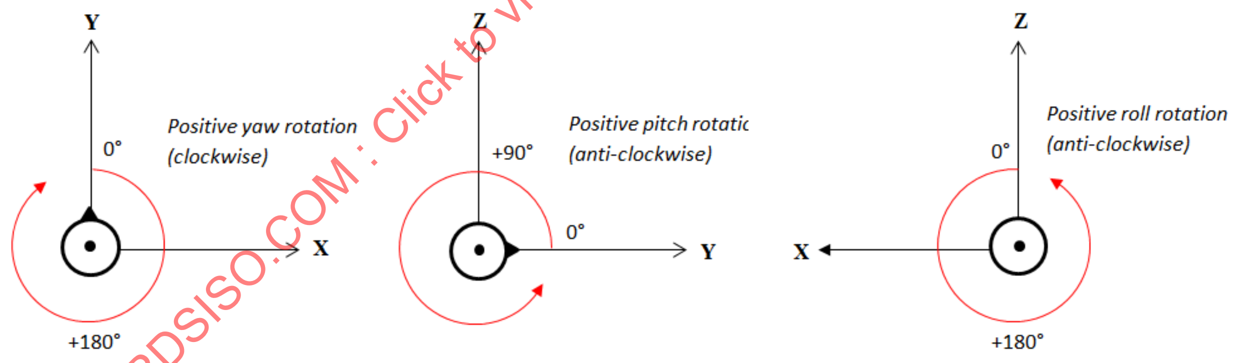


Figure AMD7.1 - Directions of positive rotations

17.X.3 Syntax of the scene displacement interface

Table AMD7.7 – Syntax of mpegh3daSceneDisplacementData()

Syntax	No. of bits	Mnemonic
mpegh3daSceneDisplacementData()		
{		
sd_yaw;	**9**	**uimsbf**
sd_pitch;	**9**	**uimsbf**

© ISO/IEC 2017 – All rights reserved

sd_roll;	9	uimsbf
}		

17.X.4 Semantics of the scene displacement interface

sd_yaw	This field defines the scene displacement angle about the z axis. This field can take values between α_{yaw} == -180° and α_{yaw} == 180°.

$$\alpha_{yaw} = \left(\frac{sd\_yaw}{2^8}\text{-}1\right) \cdot 180\ \alpha_{yaw} = \left(\frac{sd\_yaw}{2^8} - 1\right) \cdot 180;$$

α_{yaw} = min (max (α_{yaw}, -180), 180);

A positive yaw value corresponds to a clockwise rotation about the z axis.

sd_pitch	This field defines the scene displacement angle about the x axis. This field can take values between β_{pitch}= -180° and β_{pitch}= 180°.

$$\alpha_{yaw} = \left(\frac{sd\_yaw}{2^8}\text{-}1\right) \cdot 180\ \beta_{pitch} = \left(\frac{sd\_pitch}{2^8} - 1\right) \cdot 180;$$

β_{pitch} = min (max (β_{pitch}, -180), 180);

A positive pitch value corresponds to an anti-clockwise rotation about the x axis.

sd_roll	This field defines the scene displacement angle about the y axis. This field can take values between θ_{roll}= -180° and θ_{roll}= 180°.

$$\alpha_{yaw} = \left(\frac{sd\_yaw}{2^8}\text{-}1\right) \cdot 180\ \theta_{roll} = \left(\frac{sd\_roll}{2^8} - 1\right) \cdot 180;$$

θ_{roll} = min (max (θ_{roll}, -180), 180);

A positive roll value corresponds to an anti-clockwise rotation about the y axis.

*In **18** add a new subclause*

18.X1 Processing of scene displacement angles for channels and objects (CO)

If the 'useTrackingMode' flag of either binauralRendering() or loudspeakerRendering() is equal to one, the data received by the scene displacement interface mpegh3daSceneDisplacementData() shall be processed. The received angles 'yaw' (α_{yaw}), 'pitch' (β_{pitch}) and 'roll' (θ_{roll}) describe how each object's position has to be updated (i.e. the scene displacement around the z axis, x axis and y axis):

- A **positive yaw value** corresponds to a clockwise rotation about **z axis**.
- A **positive pitch value** corresponds to an anti-clockwise rotation about the **x axis**.
- A **positive roll value** corresponds to an anti-clockwise rotation about the **y axis**.

For each object and channel, it has to be checked if the 'fixedPosition' flag indicated in mae_GroupDefinitionTrackingExtension() of mae_Data() is equal to 0 or 1. Afterwards, for each element with 'fixedPosition' = 0', the position has to be updated.

Channels are therefore interpreted as objects with the position of their corresponding speaker as the object's position: The real playback speaker position shall be taken into account if given by 'known Position'. Otherwise the ideal speaker position shall be assumed.

No position interaction shall be processed for these former channel-based elements, as well as no closest speaker processing, no screen-related remapping, no spread rendering, no excluded sectors processing and no divergence or diffuseness processing.

The position update consists of the following steps:

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

- First, the object or channel position $p = (az, el, r)$ is determined. For channel-based signals the radius r is determined as follows:
  - If the intended speaker exists in the reproduction speaker setup and the distance of the reproduction setup is known, the radius r is set to the speaker distance (in cm).
  - If the intended speaker does not exist in the reproduction speaker setup, but the distance of the reproduction speakers is known, the radius r is set to the maximum reproduction speaker distance.
  - If the intended speaker does not exist in the reproduction speaker setup and no reproduction speaker distance is known, the radius r is set to 1023cm.

- After that, the position p is converted to the position p', according to the 'common' convention, where 0° azimuth is at the right ear (positive values going anti-clockwise) and 0° elevation is top of the head (positive values going downwards), resulting in $p' = (az', el', r)$

$$az' = az + 90°$$
$$el' = 90° - el$$

- The position p' is then transferred to Cartesian coordinates (x,y,z), assuming the following direction of coordinate axes:
  - o x axis pointing to the right
  - o y axis pointing straight ahead
  - o z axis pointing straight up

$$x = r \cdot \sin(el') \cdot \cos(az')$$
$$y = r \cdot \sin(el') \cdot \sin(az')$$
$$z = r \cdot \cos(el')$$

- The resulting position $v = (x, y, z)$ is then rotated with the rotation being dependent on the scene displacement input data.
- An intrinsic rotation shall be calculated using a z-x-y convention ('Yaw-Pitch-Roll Convention' (YPR)). The rotation can e.g. be realized by multiplication with a rotation matrix $\mathbf{T}_{\text{rot}}$ $\mathrm{T}_{\mathrm{rot}}$.
- This rotation results in a position $v' = (x', y', z')$
- This updates position v' is then transferred back to an azimuth-elevation notation according to the 'common' convention:

$$el_2' = \arccos\left(\frac{z'}{\sqrt{x'^2 + y'^2 + z'^2}}\right)$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

$$az_2{'} = \operatorname{atan2}(y', x') = \begin{cases} 90° - \operatorname{sign}(x') \cdot 90° & \text{if} \quad y' == 0 \\ \operatorname{sign}(y') \cdot 90° & \text{else if} \quad x' == 0 \\ \arctan\left(\frac{y'}{x'}\right) & \text{else if} \quad x' > 0 \\ \arctan\left(\frac{y'}{x'}\right) + 180° & \text{else if} \quad (y' > 0) \\ \arctan\left(\frac{y'}{x'}\right) - 180° & \text{else} \end{cases}$$

$$r_2{'} = \sqrt{x'^2 + y'^2 + z'^2}$$

- Position $p_2' = (az_2', el_2')$ is then transferred back to the MPEG-H notation, resulting in a final updated position $p_2 = (az_2, el_2)$

$$az_2 = az_2{'} - 90°$$

$$el_2 = 90° - el_2{'}$$

$$r_2 = r_2{'}$$

After that, objects and channels are fed to the object renderer with their updated positions. The object renderer renders them using the new object positions. The underlying triangulation stays the same, independent of the tracking data.

The DRC set selection process according to 6.4.4 shall ignore DRC sets for DRC-3 if scene displacement processing is enabled at the decoder.

*In **17.3.2** update **Table 156** according to*

Table 156 — Syntax for LoudspeakerRendering()

Syntax	No. of bits	Mnemonic
LoudspeakerRendering()		
{		
bsNumLoudspeakers;	**16**	**uimsbf**
localLoudspeakerSetup = SpeakerConfig3d();		
hasLoudspeakerDistance;	**1**	**bslbf**
hasLoudspeakerCalibrationGain;	**1**	**bslbf**
useTrackingMode;	**1**	**bslbf**
for (n = 0; n < bsNumLoudspeakers; n++) {		
if (speakerLayoutType <= 1) {		
hasKnownPosition[n];	**1**	**bslbf**
if (hasKnownPosition[n]) {		
loudspeakerAzimuth[n];	**9**	**uimsbf**
loudspeakerElevation[n];	**8**	**uimsbf**
}		
}		
if (hasLoudspeakerDistance) {		
loudspeakerDistance[n];	**10**	**uimsbf**
}		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

if (hasLoudspeakerCalibrationGain) {		
loudspeakerCalibrationGain[n];	**7**	**uimsbf**
}		
}		
externalDistanceCompensation;	**1**	**bslbf**
}		

*In **17.3.3** update the semantics according to*

bsNumLoudspeakers	Number of loudspeakers in the local reproduction setup.
hasLoudspeakerDistance	This field defines if a loudspeaker distance is given in the bit stream.
hasLoudspeakerCalibrationGain	This field defines if a loudspeaker calibration gain is given in the bit stream.
useTrackingMode	This field defines if a processing of scene displacement values sent via the mpegh3daSceneDisplacementData() interface shall happen or not.
hasKnownPosition	This flag defines if an explicit signaling of the known position of the loudspeaker is following in the bit stream.

loudspeakerAzimuth	This field defines the azimuth angle of a speaker. It can take values between -180° and 180° in 1° steps.
	
	Azimuth = (loudspeakerAzimuth – 256)
	
	Azimuth = min (max (Azimuth, -180), 180)

loudspeakerElevation	This field defines the elevation angle of a speaker. It can take values between -90° and 90° in 1° steps.
	
	Elevation = (loudspeakerElevation – 128)
	
	Elevation= min (max (Elevation, -90), 90)

loudspeakerDistance	This field defines the distance of a loudspeaker to the reference point centered in the loudspeaker setup in cm. The field can take values between 0 and 1023, corresponding to 0 to 1023 cm in 1cm steps.
loudspeakerCalibrationGain	This field defines a loudspeaker calibration gain in dB. The field can take values between 0 and 127, corresponding to dB values between Gain = -32 dB and Gain = 31.5 dB in 0.5 dB steps: Gain [dB] = 0.5 × (loudspeakerGain – 64)
externalDistanceCompensation	This flag defines whether loudspeaker compensation shall be a applied to the decoder output signals. If this flag is 1, the loudspeaker compensation shall be omitted and loudspeakerDistance and loudspeakerCalibrationGain shall not be applied in the decoder..

*In **17.4.2** update **Table 157** according to*

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Table 157 — Syntax of BinauralRendering()

Syntax	No. of bits	Mnemonic
BinauralRendering()		
{		
bsFileSignature;	**32**	**bslbf**
bsFileVersion;	**8**	**uimsbf**
bsNumCharName;	**8**	**uimsbf**
for (i=0; i<bsNumCharName; i++) {		
bsName[i];	**8**	**bslbf**
}		
useTrackingMode;	**1**	**bslbf**
bsNumBinauralDataRepresentation;	**4**	**uimsbf**
for (r = 0; r < bsNumBinauralDataRepresentation; r++) {		
brirSamplingFrequencyIndex;	**5**	**uimsbf**
if (brirSamplingFrequencyIndex == 0x1f) {		
brirSamplingFrequency;	**24**	**uimsbf**
}		
isHoaData;	**1**	**bslbf**
if (isHoaData) {		
hoaOrderBinaural = escapedValue(3,5,0);	**3,8**	**uimsbf**
nBrirPairs = (hoaOrderBinaural+1)^2;		
} else {		
MeasurementSetup = SpeakerConfig3d();		
if (speakerLayoutType == 0) { /* See SpeakerConfig3d() */		
nBrirPairs = escapedValue(5,8,0)+1;	**5,13**	**uimsbf**
} else {		
nBrirPairs = numSpeakers; /* See SpeakerConfig3d() */		
}		
}		
bsBinauralDataFormatID;	**2**	**uimsbf**
ByteAlign();		
switch (bsBinauralDataFormatID) {		
case 0:		
BinauralFIRData();		
break;		
case 1:		
FdBinauralRendererParam();		
break;		
case 2:		
TdBinauralRendererParam();		
break;		
}		
}		
}		

In 17.4.3 replace

useHeadTrackingMode	This flag defines if a headtracker is connected and the binaural rendering should be processed in a special headtracking mode.

with

useTrackingMode	This flag defines if a tracker device is connected and the binaural rendering shall be processed in a special headtracking mode, meaning a processing of

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

scene displacement values sent via the mpegh3daSceneDisplacementData() interface shall happen.

*In **18** add a new subclause*

18.X2 Processing of scene displacement angles for scene based content (HOA)

The rotation of the HOA representation takes place after spatial HOA decoding and before the optional warping for the screen adaptation and rendering. It can be achieved by two equivalent approaches:

1. The first approach itself consists of three following steps:
   a. Transforming the original HOA representation of order N to the spatial domain, i.e. representing it by general plane waves from $O = (N+1)^2$ directions of incidence $\boldsymbol{\Omega}_q^{(N)}$, $q = 1, \dots, O$, defined in sections F.2 to F.9.
   b. Rotating the directions according to the desired rotation of the HOA representation to provide O rotated directions $\widehat{\boldsymbol{\Omega}}_q^{(N)}$, $q = 1, \dots, O$.
   c. Applying an inverse spatial transform to the general plane waves assuming the rotated directions to provide the rotated HOA representation.

The three operations can be expressed by

$$\widehat{\boldsymbol{C}} = \boldsymbol{\Psi}_{new}\ \boldsymbol{\Psi}_{\boldsymbol{O}}^{-1}\ \boldsymbol{C}\,,$$

with $\boldsymbol{C} \in \mathbb{R}^{(N+1)^2 \text{ x } L}$ and $\widehat{\boldsymbol{C}} \in \mathbb{R}^{(N+1)^2 \text{ x } L}$ denoting the frames of the original and rotated HOA representations, $\boldsymbol{\Psi}_{\mathbf{O}}^{-1}$ denoting the inverse of the mode matrix with respect to the directions $\boldsymbol{\Omega}_q^{(N)}$ expressing the spatial transform and $\boldsymbol{\Psi}_{\mathbf{new}}$ indicating the mode matrix with respect to the rotated directions $\widehat{\boldsymbol{\Omega}}_q^{(N)}$. The mode matrices are computed from the directions as described in according in section F.1.5.

The approach can be beneficial for lowering the computational complexity in cases the HOA representation is already given by in the spatial domain.

The rotation of the directions can be accomplished as follows:

First, the directions $\boldsymbol{\Omega}_q^{(N)} = (\theta_q^{(N)}, \phi_q^{(N)}\ ), q = 1, \dots, O,$ defined in sections F.2 to F.9 and given in spherical coordinates are first converted to positions $\boldsymbol{\Omega}_{\boldsymbol{xyz}}^{(N)} = [x_q, y_q, z_q]^T$ (assuming a radius of one) in Cartesian Coordinates , i.e. $\boldsymbol{\Omega}_q^{(N)}, \rightarrow \boldsymbol{\Omega}_{\boldsymbol{xyz}}^{(N)}$ by

$$x_q = sin\ \theta_q^N\ \cos \phi_q^N$$

$$y_q = sin\ \theta_q^N\ \sin \phi_q^N$$

$$z_q = cos\ \theta_q^N$$

Then, the rotated positions are computed by

$$\widehat{\boldsymbol{\Omega}}_{\mathbf{xyz}}^{(N)} = \boldsymbol{T}_{\boldsymbol{rot}_{\boldsymbol{hoa}}} \boldsymbol{\Omega}_{\boldsymbol{xyz}}^{(N)}$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

where $\boldsymbol{T}_{rot_{hoa}}$ is the rotation matrix for the HOA coordinate system derived from the scene displacement angles (yaw, pitch roll), see Annex I.

Eventually, the rotated positions are converted back to the spherical coordinate system $\widehat{\boldsymbol{\Omega}}_{\mathbf{xyz}}^{(N)} \rightarrow \widehat{\boldsymbol{\Omega}}^{(N)}$ by

$$\hat{\theta}_q^{(N)} = atan2\left(\sqrt{\hat{x}^2 + \hat{y}^2}, \hat{z}\right),$$
$$\hat{\phi}_q^{(N)} = atan2(\hat{y}, \hat{x})$$

where *atan2()* is defined according to **18.X1**.

2. By expressing the rotation directly as one single matrix multiplication applied to the frame of the original HOA representation as

$$\widehat{\boldsymbol{C}} = \boldsymbol{M}\,\boldsymbol{C},$$

where the elements of the transform matrix $\boldsymbol{M}$ can be computed by $\boldsymbol{M} = \boldsymbol{\Psi}_{new}\ \boldsymbol{\Psi}_{0}^{-1}$ or directly using so –called Wigner-D functions, J. Sakurai, J. Napolitano, "Modern Quantum Mechanics", Addison-Wesley, 2010. Note that the transform matrix $\boldsymbol{M}$ has a special block-diagonal structure, which looks for an HOA order of 2 as follows:

$$\mathbf{M}|_{N=2} = \begin{bmatrix} 1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & [3x3] & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & [5x5] \end{bmatrix}.$$

Due to this rather sparse structure, the matrix multiplication with $\boldsymbol{M}$ can be accomplished very efficiently.

The DRC set selection process according to 6.4.4 shall ignore DRC sets for DRC-3 if scene displacement processing is enabled at the decoder.

*Add a new **Annex I***

Annex I

(informative)

Determination of a rotation matrix for processing of scene displacement data

The determination of a rotation matrix for an intrinsic rotation (also called ‘intermediate frames’, ‘rotation of moving body axes’, meaning a rotation about the axes of a rotating coordinate system. The rotating coordinate system is initially aligned with the fixed one and modifies its orientation after each elemental rotation) with the z-x-y convention (‘Yaw-Pitch-Roll Convention’ (YPR)) is shown below.

Note that the following text defines rotation matrixes that are designed to be pre-multiplied with a column-vector representing a position in 3D space.

The overall rotation matrix is defined as the matrix product of three elemental rotations. The three elemental rotations (also called coordinate rotations) are:

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

$$\mathbf{T}_{\text{rot},x}(\cdot) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\cdot) & \sin(\cdot) \\ 0 & -\sin(\cdot) & \cos(\cdot) \end{bmatrix}$$

$$\mathbf{T}_{\text{rot},y}(\cdot) = \begin{bmatrix} \cos(\cdot) & 0 & -\sin(\cdot) \\ 0 & 1 & 0 \\ \sin(\cdot) & 0 & \cos(\cdot) \end{bmatrix}$$

$$\mathbf{T}_{\text{rot},z}(\cdot) = \begin{bmatrix} \cos(\cdot) & \sin(\cdot) & 0 \\ -\sin(\cdot) & \cos(\cdot) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The three different elemental rotations are combined by matrix multiplication to form the overall multiplication matrix (‘Yaw-Pitch-Roll Convention’). The signs for the pitch and roll angles are negative, because the elemental multiplication are always defined as clockwise rotations.

$$\begin{aligned}\mathbf{T}_{\text{rot}} &= \mathbf{T}_{\text{rot},z}(\alpha_{\text{yaw}}) \cdot \mathbf{T}_{\text{rot},x}(-\theta_{\text{pitch}}) \cdot \mathbf{T}_{\text{rot},y}(-\beta_{\text{roll}}) \\ &= \mathbf{T}_{\text{rot},z}(\alpha_{\text{yaw}}) \cdot \mathbf{T}_{\text{rot},x}(\theta_{\text{pitch}})^T \cdot \mathbf{T}_{\text{rot},y}(\beta_{\text{roll}})^T\end{aligned}$$

$$= \begin{bmatrix} \cos(\alpha_{\text{yaw}}) & \sin(\alpha_{\text{yaw}}) & 0 \\ -\sin(\alpha_{\text{yaw}}) & \cos(\alpha_{\text{yaw}}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_{\text{pitch}}) & -\sin(\theta_{\text{pitch}}) \\ 0 & \sin(\theta_{\text{pitch}}) & \cos(\theta_{\text{pitch}}) \end{bmatrix} \cdot \begin{bmatrix} \cos(\beta_{\text{roll}}) & 0 & \sin(\beta_{\text{roll}}) \\ 0 & 1 & 0 \\ -\sin(\beta_{\text{roll}}) & 0 & \cos(\beta_{\text{roll}}) \end{bmatrix}$$

If a matrix is needed to be post-multiplied with a row-vector, the order of elemental rotations has to be inverted.

Determination of rotation matrix for scene based content:

From the scene displacement angles (roll, pitch and yaw), a rotation matrix $\boldsymbol{T}_{\boldsymbol{rot_{hoa}}}$ suited for the HOA coordinate system (ISO/IS 23008-3, Annex F.1.1) is calculated:

$$\boldsymbol{T}_{\boldsymbol{rot_{hoa}}} = \boldsymbol{T}_{rot,z}(\alpha_{\text{yaw}})\, \boldsymbol{T}_{rot,y}(\beta_{\text{pitch}})\, \boldsymbol{T}^{T}_{rot,x}(\theta_{\text{roll}}),$$

where $\boldsymbol{T}^T$ denotes the transposed of matrix $\boldsymbol{T}$ and the basic rotation matrices $\boldsymbol{T}_{rot,x}, \boldsymbol{T}_{rot,y}, \boldsymbol{T}_{rot,z}$ are defined above.

9.4 Extension of screen-related processing for off-centered screens

*In **18.3** replace*

The screen size of a nominal reference screen (used in the mixing and monitoring process) and local screen size information in the playback room are taken into account for the remapping.

If no nominal reference screen size is given, default reference values are used assuming a 4k display and an optimal viewing distance.

with

The screen size of a nominal reference screen (used in the mixing and monitoring process) and local screen size information in the playback room are taken into account for the remapping.

The size of the applicable nominal reference screen is in general read from the bitstream, located in the mae_ProductionScreenSizeData() syntax structure.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

If the structure mae_ProductionScreenSizeDataExtension() is present in the bitstream, mae_NumPresetProductionScreens is bigger than 0 and the interaction mode is set to 'basic interaction mode', the groupPresetID of the currently chosen/valid group preset shall be used to identify the applicable screen size of the reference screen from the mae_ProductionScreenSizeDataExtension() structure before the remapping is applied.

If no is chosen or the chosen preset has no associated production screen, the production screen from the mae_ProductionScreenSizeData() structure shall be used as a default production screen.

If the structure mae_ProductionScreenSizeDataExtension() is present in the bitstream, and the mae_overwriteProductionScreenSizeData flag as well as the hasNonStandardScreenSize flag from mae_ProductionScreenSizeData() are equal to 1, the azimuth angle data originating from mae_ProductionScreenSizeDataExtension() shall be used instead of the azimuth angle data from mae_ProductionScreenSizeData() to define the default production screen.

If no nominal reference screen size is given, default reference values are used assuming a 4k display and an optimal viewing distance.

Further, in ***18.3*** *after*

The reproduction screen edges are abbreviated by:

$\varphi_{left}^{repro}, \varphi_{right}^{repro}, \theta_{top}^{repro}, \theta_{bottom}^{repro}$

add

Azimuth values of a temporary reproduction screen are defined by:

$$\varphi_{\text{offset}}^{\text{repro}} = \begin{cases} (360° + \varphi_{\text{left}}^{\text{repro}} + \varphi_{\text{right}}^{\text{repro}}) \cdot \frac{1}{2} & \text{if } \left(\varphi_{\text{right}}^{\text{repro}} > \varphi_{\text{left}}^{\text{repro}}\right) \\ (\varphi_{\text{left}}^{\text{repro}} + \varphi_{\text{right}}^{\text{repro}}) \cdot \frac{1}{2} & \text{else} \end{cases}$$

$$\varphi_{\text{left, temp}}^{\text{repro}} = \varphi_{\text{left}}^{\text{repro}} - \varphi_{\text{offset}}^{\text{repro}}$$

$$\varphi_{\text{right, temp}}^{\text{repro}} = \varphi_{\text{right}}^{\text{repro}} - \varphi_{\text{offset}}^{\text{repro}}$$

$$\varphi_{\text{left, temp}}^{\text{repro}} = \begin{matrix} \varphi_{\text{left, temp}}^{\text{repro}} - 360° & \text{if } (\varphi_{\text{left, temp}}^{\text{repro}} > 180°) \\ \varphi_{\text{left, temp}}^{\text{repro}} + 360° & \text{if } (\varphi_{\text{left, temp}}^{\text{repro}} < -180°) \\ \varphi_{\text{left, temp}}^{\text{repro}} & \text{else} \end{matrix}$$

$$\varphi_{\text{right, temp}}^{\text{repro}} = \begin{matrix} \varphi_{\text{right, temp}}^{\text{repro}} - 360° & \text{if } (\varphi_{\text{right, temp}}^{\text{repro}} > 180°) \\ \varphi_{\text{right, temp}}^{\text{repro}} + 360° & \text{if } (\varphi_{\text{right, temp}}^{\text{repro}} < -180°) \\ \varphi_{\text{right, temp}}^{\text{repro}} & \text{else} \end{matrix}$$

Azimuth values of a temporary reference screen are defined by:

$$\varphi_{\text{offset}}^{\text{nominal}} = \begin{cases} (360° + \varphi_{\text{left}}^{\text{nominal}} + \varphi_{\text{right}}^{\text{nominal}}) \cdot \frac{1}{2} & \text{if } \left(\varphi_{\text{right}}^{\text{nominal}} > \varphi_{\text{left}}^{\text{nominal}}\right) \\ (\varphi_{\text{left}}^{\text{nominal}} + \varphi_{\text{right}}^{\text{nominal}}) \cdot \frac{1}{2} & \text{else} \end{cases}$$

$$\varphi_{\text{left, temp}}^{\text{nominal}} = \varphi_{\text{left}}^{\text{nominal}} - \varphi_{\text{offset}}^{\text{nominal}}$$

$$\varphi_{\text{right, temp}}^{\text{nominal}} = \varphi_{\text{right}}^{\text{nominal}} - \varphi_{\text{offset}}^{\text{nominal}}$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

$$\varphi_{\text{left, temp}}^{\text{nominal}} = \begin{cases} \varphi_{\text{left, temp}}^{\text{nominal}} - 360^\circ & \text{if } (\varphi_{\text{left, temp}}^{\text{nominal}} > 180^\circ) \\ \varphi_{\text{left, temp}}^{\text{nominal}} + 360^\circ & \text{if } (\varphi_{\text{left, temp}}^{\text{nominal}} < -180^\circ) \\ \varphi_{\text{left, temp}}^{\text{nominal}} & \text{else} \end{cases}$$

$$\varphi_{\text{right, temp}}^{\text{nominal}} = \begin{cases} \varphi_{\text{right, temp}}^{\text{nominal}} - 360^\circ & \text{if } (\varphi_{\text{right, temp}}^{\text{nominal}} > 180^\circ) \\ \varphi_{\text{right, temp}}^{\text{nominal}} + 360^\circ & \text{if } (\varphi_{\text{right, temp}}^{\text{nominal}} < -180^\circ) \\ \varphi_{\text{right, temp}}^{\text{nominal}} & \text{else} \end{cases}$$

A temporal input azimuth value is defined according to:

$$\varphi_{\text{temp}} = \varphi - \varphi_{\text{offset}}^{\text{nominal}}$$

$$\varphi_{\text{temp}} = \begin{cases} \varphi_{\text{temp}} - 360^\circ & \text{if } (\varphi_{\text{temp}} > 180^\circ) \\ \varphi_{\text{temp}} + 360^\circ & \text{if } (\varphi_{\text{temp}} < -180^\circ) \\ \varphi_{\text{temp}} & \text{else} \end{cases}$$

Replace

$$\varphi' = \begin{cases} \dfrac{\varphi_{right}^{repro} + 180^\circ}{\varphi_{right}^{\text{nominal}} + 180^\circ} \cdot (\varphi + 180^\circ) - 180^\circ & for \quad -180^\circ \le \varphi < \varphi_{right}^{\text{nominal}} \\ \dfrac{\varphi_{left}^{repro} - \varphi_{right}^{repro}}{\varphi_{left}^{\text{nominal}} - \varphi_{right}^{\text{nominal}}} \cdot \left(\varphi - \varphi_{right}^{\text{nominal}}\right) + \varphi_{right}^{repro} & for \quad \varphi_{right}^{\text{nominal}} \le \varphi < \varphi_{left}^{\text{nominal}} \\ \dfrac{180^\circ - \varphi_{left}^{repro}}{180^\circ - \varphi_{left}^{\text{nominal}}} \cdot \left(\varphi - \varphi_{left}^{\text{nominal}}\right) + \varphi_{left}^{repro} & for \quad \varphi_{left}^{\text{nominal}} \le \varphi < 180^\circ \end{cases}$$

with

$$\varphi'_{\text{temp}} = \begin{cases} \dfrac{\varphi_{\text{right,temp}}^{\text{repro}} + 180^\circ}{\varphi_{\text{right,temp}}^{\text{nominal}} + 180^\circ} \cdot (\varphi_{\text{temp}} + 180^\circ) - 180^\circ & for \quad -180^\circ \le \varphi_{\text{temp}} < \varphi_{\text{right,temp}}^{\text{nominal}} \\ \dfrac{\varphi_{\text{left,temp}}^{\text{repro}} - \varphi_{\text{right,temp}}^{\text{repro}}}{\varphi_{\text{left,temp}}^{\text{nominal}} - \varphi_{\text{right,temp}}^{\text{nominal}}} \cdot \left(\varphi_{\text{temp}} - \varphi_{\text{right,temp}}^{\text{nominal}}\right) + \varphi_{\text{right,temp}}^{\text{repro}} & for \quad \varphi_{\text{right}}^{\text{nominal}} \le \varphi_{\text{temp}} < \varphi_{\text{left,temp}}^{\text{nominal}} \\ \dfrac{180^\circ - \varphi_{\text{left,temp}}^{\text{repro}}}{180^\circ - \varphi_{\text{left,temp}}^{\text{nominal}}} \cdot \left(\varphi_{\text{temp}} - \varphi_{\text{left,temp}}^{\text{nominal}}\right) + \varphi_{\text{left,temp}}^{\text{repro}} & for \quad \varphi_{\text{left,temp}}^{\text{nominal}} \le \varphi_{\text{temp}} < 180^\circ \end{cases}$$

$$\varphi' = \varphi'_{\text{temp}} + \varphi_{\text{offset}}^{\text{repro}}$$

$$\varphi' = \begin{cases} \varphi' - 360^\circ & \text{if } (\varphi' > 180^\circ) \\ \varphi' + 360^\circ & \text{if } (\varphi' < -180^\circ) \\ \varphi' & \text{else} \end{cases}$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Further, in 18.3 replace Figure 71 with:

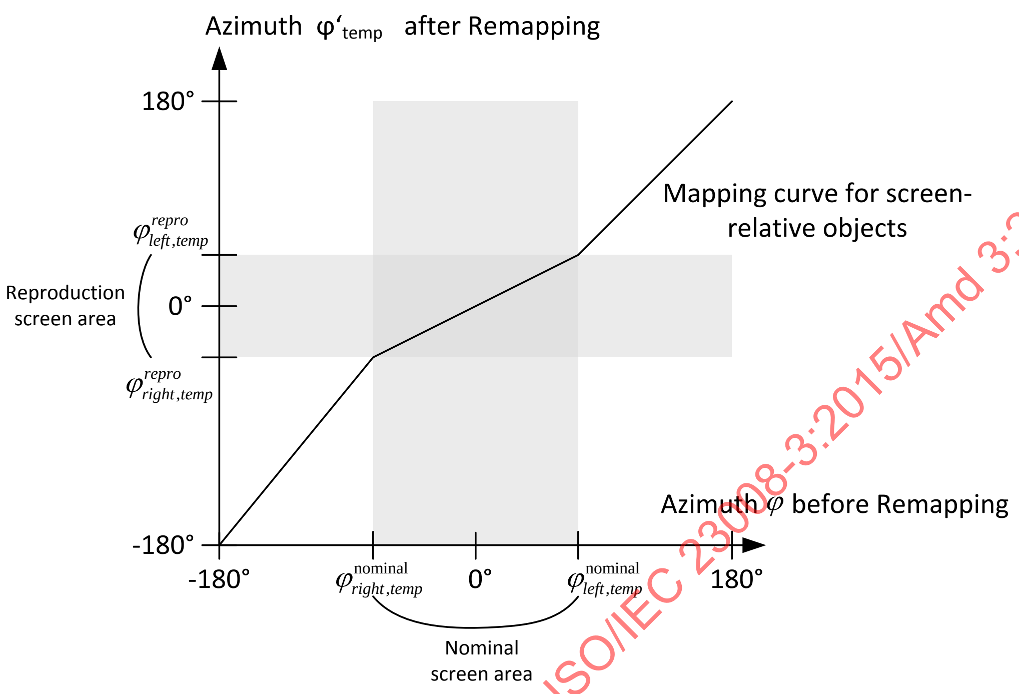


Figure 1 — Remapping function of position data

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

18.X Screen-Related Adaptation and Zooming for Higher Order Ambisonics (HOA)

Screen-related adaptation and zooming is only possible if the IsScreenRelative flag in the HOAConfig() (c.f. table 119) is signaled as 1.

The screen-related adaptation process modifies the HOA rendering matrix and is only computed during the initialization phase. Figure 18.X.1 depicts the process. If no local screen size information is available, no screen-related adaptation shall be applied. In case only azimuth screen size information is given, the decoder shall not apply any screen-related adaptation in the vertical dimension.

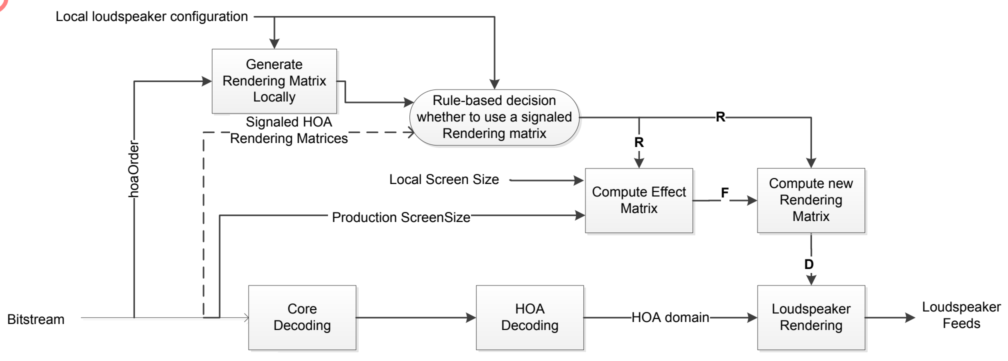

Local loudspeaker configuration
Generate Rendering Matrix Locally
Signaled HOA Rendering Matrices
Rule-based decision whether to use a signaled Rendering matrix
R
R
hoaOrder
Local Screen Size
Production ScreenSize
Compute Effect Matrix
F
Compute new Rendering Matrix
D
Bitstream
Core Decoding
HOA Decoding
HOA domain
Loudspeaker Rendering
Loudspeaker Feeds

Figure 18.X.1 Screen-related processing for Higher Order Ambisonics

The screen-related adaptation of the rendering matrix is achieved by:

1. Generating a mode matrix $\boldsymbol{\Psi}^{(O,M)}$ as described in Annex F.1.5 with M=900 sampling points which directions (θ, ϕ) are defined in Annex F.9. $\boldsymbol{\Psi}^{(O,M)} := [\,S_1^O\ S_2^O\ \dots\ S_M^O] \,\epsilon \mathbb{R}^{O\times M}$.

2. The directions of those M sampling points are first modified using the mapping function defined in clause 18.3. Then a mode matrix $\boldsymbol{\Psi}_m^{(O,M)}$ based on these modified points is computed accordingly.

3. Computing a preliminary effect matrix:

$$\widetilde{\boldsymbol{F}} = \boldsymbol{\Psi}_m^{(O,M)}\ \boldsymbol{\Psi}^{(O,M)^\dagger},$$

   where $\boldsymbol{\Psi}^{(O,M)^\dagger}$ denotes the pseudo-inverse of the mode matrix $\boldsymbol{\Psi}^{(O,M)}$.

4. Computing a loudness correction value by using the HOA rendering matrix R (c.f. clause 12.4.3.2) for each spatial direction $l = 1 \dots M$:

$$A(l) = \sqrt{\frac{(\boldsymbol{R}\, S_{m,l}^{O})^T (\boldsymbol{R}\, S_{m,l}^{O})}{(\boldsymbol{R}\, \widetilde{\boldsymbol{F}} S_l^O)^T (\boldsymbol{R}\, \widetilde{\boldsymbol{F}}\, S_l^O)}}$$

5. Computing the final effect matrix:

$$\boldsymbol{F} = \boldsymbol{\Psi}_m^{(O,M)} diag(A)\ \boldsymbol{\Psi}^{(O,M)^\dagger},$$

   where $diag(A)$ denotes a diagonal matrix including the vector A.

6. Computing the new rendering matrix:

$$\boldsymbol{D} = \boldsymbol{RF}$$

The zoom-depending adaptation of Higher Order Ambisonics is depicted in Figure 18.X.2. If no local zoom information is available, zooming shall not be applied. The same algorithmic principles as described for the screen-related processing for Higher Order Ambisonics are applied, but the rendering matrix must be adapted at runtime according to the data provided by the LocalZoomAreaSize() interface. During a dynamic zooming event, a new effect matrix $\boldsymbol{F}$ shall be computed based a mode matrix $\boldsymbol{\Psi}^{(O,M)}$with $M = (N+2)^2$ equally spatial sampling points which directions are given in Annex F.2 to F.9. Once the zoom is stationary, the new effect matrix $\boldsymbol{F}$ is computed based a mode matrix $\boldsymbol{\Psi}^{(O,M)}$with $M = 900$ spatial sampling points as decribed above.

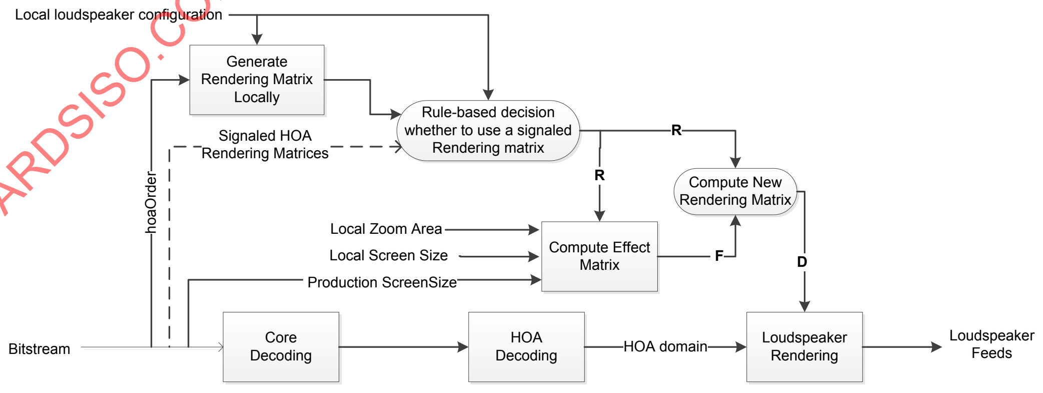

Local loudspeaker configuration
Generate Rendering Matrix Locally
Signaled HOA Rendering Matrices
Rule-based decision whether to use a signaled Rendering matrix
R
R
Compute New Rendering Matrix
hoaOrder
Local Zoom Area
Local Screen Size
Production ScreenSize
Compute Effect Matrix
F
D
Bitstream
Core Decoding
HOA Decoding
HOA domain
Loudspeaker Rendering
Loudspeaker Feeds

Figure 18.X.2— Overview of Zooming for Higher Order Ambisonics

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

9.5 Update of closest speaker playout for the conditioned case

*In **18.5** replace*

This distance has to be calculated for all known position P_1 to P_N of the N output speakers with respect to the wanted position of the audio element P_{wanted}.

with

This distance has to be calculated for all known position P_1 to P_N of a defined list of N output speakers with respect to the wanted position of the audio element P_{wanted}.

*Add a new subclause **18.X***

18.X Determination of a list of speakers for conditioned closest speaker playback

If the closest speaker playback is conditioned, the 'closest speaker processing' shall only be conducted if one or more speakers are located in a defined area around the members of the group.

The corresponding object should then be played back by the closest speaker within this defined range. The area is defined by minimum and maximum values based on the given threshold angle (see 15.3 for semantics):

- $\varphi_{\min} = \varphi_{\text{obj}} - \varphi_{\text{thresh}}$
- $\varphi_{\max} = \varphi_{\text{obj}} + \varphi_{\text{thresh}}$
- $\theta_{\min} = \theta_{\text{obj}} - \theta_{\text{thresh}}$
- $\theta_{\max} = \theta_{\text{obj}} + \theta_{\text{thresh}}$

If a speaker lies within the closest speaker playout processing range, both the following conditions have to be true:

— $\varphi_{\text{speaker}} \geq \varphi_{\text{obj}} - \varphi_{\text{thresh}}$ && $\varphi_{\text{speaker}} \leq \varphi_{\text{obj}} + \varphi_{\text{thresh}}$

— $\theta_{\text{speaker}} \geq \theta_{\text{obj}} - \theta_{\text{thresh}}$ && $\theta_{\text{speaker}} \leq \theta_{\text{obj}} + \theta_{\text{thresh}}$

Special cases have to be considered for values out of the allowed range:

- $\varphi_{\min} < -180°$
- $\varphi_{\max} > +180°$
- $\theta_{\min} < -90°$
- $\theta_{max} > +90°$

Minimum or maximum elevation is out of range, minimum and maximum azimuth are within the allowed range:

If (($\theta_{\max} > 90°$ || $\theta_{\min} < -90°$) && ($\varphi_{\min} \geq -180°$ && $\varphi_{\max} \leq 180°$), the following condition has to be true:

$$(\varphi_{\text{speaker}} \geq \varphi_{\min} \ \&\& \ \varphi_{\text{speaker}} \leq \varphi_{\max}) \ \&\& \ (\theta_{\text{speaker}} \geq \theta_{\min,1} \ \&\& \ \theta_{\text{speaker}} \leq \theta_{\max,1})$$

$$||$$

$$(\varphi_{\text{speaker}} \geq \varphi_{\min,2} \ \&\& \ \varphi_{\text{speaker}} \leq \varphi_{\max,2}) \ \&\& \ (\theta_{\text{speaker}} \geq \theta_{\min,2} \ \&\& \ \theta_{\text{speaker}} \leq \theta_{\max,2})$$

with

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

$$\left.\begin{array}{l}\theta_{\min,1}=\theta_{\min}\\ \theta_{\max,1}=90^\circ\\ \theta_{\min,2}=90^\circ-\left|90^\circ-\theta_{\max}\right|\\ \theta_{\max,2}=90^\circ\\ \varphi_{\min,2}=\varphi_{\min}+180^\circ\\ \varphi_{\max,2}=\varphi_{\max}+180^\circ\end{array}\right\}\text{if }\ \theta_{\max}>90^\circ \qquad \left.\begin{array}{l}\theta_{\min,1}=-90^\circ\\ \theta_{\max,1}=\theta_{\max}\\ \theta_{\min,2}=-90^\circ\\ \theta_{\max,2}=90^\circ+\left|90^\circ+\theta_{\min}\right|\\ \varphi_{\min,2}=\varphi_{\min}+180^\circ\\ \varphi_{\max,2}=\varphi_{\max}+180^\circ\end{array}\right\}\text{if }\ \theta_{\min}<-90^\circ$$

Minimum or maximum azimuth is out of range, minimum and maximum elevation are within the allowed range:

If ((φ_{max} > 180° || φ_{min} < -180°) && (θ_{min} ≥ -90° && θ_{max} ≤ 90°), the following condition has to be true:

$$(\varphi_{speaker} \geq \varphi_{min,1} \ \&\& \ \varphi_{speaker} \leq \varphi_{max,1}) \ \&\& \ (\theta_{speaker} \geq \theta_{min} \ \&\& \ \theta_{speaker} \leq \theta_{max})$$

$$||$$

$$(\varphi_{speaker} \geq \varphi_{min,2} \ \&\& \ \varphi_{speaker} \leq \varphi_{max,2}) \ \&\& \ (\theta_{speaker} \geq \theta_{min} \ \&\& \ \theta_{speaker} \leq \theta_{max})$$

with

$$\left.\begin{array}{l}\varphi_{\min,1}=-180^\circ\\ \varphi_{\max,1}=\varphi_{\max}\\ \varphi_{\min,2}=180^\circ-(\varphi_{\min}+180^\circ)\\ \varphi_{\max,2}=180^\circ\end{array}\right\}\text{if }\ \varphi_{\min}<-180^\circ \qquad \left.\begin{array}{l}\varphi_{\min,1}=\varphi_{\min}\\ \varphi_{\max,1}=180^\circ\\ \varphi_{\min,2}=-180^\circ\\ \varphi_{\max,2}=-180^\circ+(\varphi_{\max}-180^\circ)\end{array}\right\}\text{if }\ \varphi_{\max}>180^\circ$$

Minimum or maximum azimuth is out of range, and minimum or maximum elevation is out of range:

If ((φ_{max} > 180° || φ_{min} < -180°) && (θ_{min} < -90° || θ_{max} > 90°), the following condition has to be true:

$$(\varphi_{speaker} \geq \varphi_{min,1} \ \&\& \ \varphi_{speaker} \leq \varphi_{max,1}) \ \&\& \ (\theta_{speaker} \geq \theta_{min,1} \ \&\& \ \theta_{speaker} \leq \theta_{max,1})$$

$$||$$

$$(\varphi_{speaker} \geq \varphi_{min,2} \ \&\& \ \varphi_{speaker} \leq \varphi_{max,2}) \ \&\& \ (\theta_{speaker} \geq \theta_{min,1} \ \&\& \ \theta_{speaker} \leq \theta_{max,1})$$

$$||$$

$$(\varphi_{speaker} \geq \varphi_{min,3} \ \&\& \ \varphi_{speaker} \leq \varphi_{max,3}) \ \&\& \ (\theta_{speaker} \geq \theta_{min,2} \ \&\& \ \theta_{speaker} \leq \theta_{max,2})$$

with

$$\left.\begin{array}{l}\varphi_{\min,1}=-180^\circ\\ \varphi_{\max,1}=\varphi_{\max}\\ \varphi_{\min,2}=180^\circ-(\varphi_{\min}+180^\circ)\\ \varphi_{\max,2}=180^\circ\end{array}\right\}\text{if }\ \varphi_{\min}<-180^\circ \qquad \left.\begin{array}{l}\varphi_{\min,1}=\varphi_{\min}\\ \varphi_{\max,1}=180^\circ\\ \varphi_{\min,2}=-180^\circ\\ \varphi_{\max,2}=-180^\circ+(\varphi_{\max}-180^\circ)\end{array}\right\}\text{if }\ \varphi_{\max}>180^\circ$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

$$\left.\begin{aligned}\theta_{\min,1} &= \theta_{\min}\\ \theta_{\max,1} &= 90^\circ\\ \theta_{\min,2} &= 90^\circ - \left|90^\circ - \theta_{\max}\right|\\ \theta_{\max,2} &= 90^\circ\\ \varphi_{\min,3} &= \varphi_{\min} + 180^\circ\\ \varphi_{\max,3} &= \varphi_{\max} + 180^\circ\end{aligned}\right\}\text{if } \theta_{\max} > 90^\circ \qquad \left.\begin{aligned}\theta_{\min,1} &= -90^\circ\\ \theta_{\max,1} &= \theta_{\max}\\ \theta_{\min,2} &= -90^\circ\\ \theta_{\max,2} &= 90^\circ + \left|90^\circ + \theta_{\min}\right|\\ \varphi_{\min,3} &= \varphi_{\min} + 180^\circ\\ \varphi_{\max,3} &= \varphi_{\max} + 180^\circ\end{aligned}\right\}\text{if } \theta_{\min} < -90^\circ$$

For each of the output speaker, it has to be determined if it lies within the defined area. The closest speaker playout should then only take into account the speakers within this range.

9.6 Processing of excluded sectors

*Add a new subclause **18.X2***

18.X2 Determination of a reduced reproduction layout based on excluded sectors

If it should be determined for a specific list of reproduction speakers which ones shall be excluded from rendering based on a list of signaled exclusion sectors, the following processing has to be conducted:

- For each relevant signal group a 'group-related' reproduction layout is determined, based on the signaling of excluded sectors in the EnhancedObjectMetadataConfig() and EnhancedObjectMetadataFrame() structures.
- Therefore, each excluded sector of the current group is associated with a 'condition', conditioning the positions of speakers that shall be excluded for rendering.
    - If the excluded sector is given by a pre-defined sector index, the condition is defined as stated in Table 18.X2.1.
    - If the excluded sector is an arbitrary sector defined by minimum and maximum excluded elevation and azimuth values, the condition is given by

$$\left((el \geq el_{\min})\,\&\&(el \leq el_{\max})\right)\&\&\left((az \geq az_{\min})\,\&\&(az \leq az_{\max})\right)$$

- Each reproduction speaker's position (az, el) is compared to all conditions given by the list of excluded sectors associated with the current group. If a condition is true, the current speaker shall be excluded from rendering.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table AMD7.8 - Conditions of excluded sectors

Sector Index	Short description	Condition
0	No positive elevation	el > 0
1	No negative elevation	el < 0
2	No front	((abs(az) < 90) && (abs(el) < 90))
3	No right side	((az > -180) && (az < 0))
4	No left side	((az > 0) && (az < 180))
5	No surround	((abs(az) >= 90) || ((az=0) && (el=90)))
6	Screen only	$(((az > \varphi_{left}^{repro}) \| (az < \varphi_{right}^{repro}))$ && $((el > \theta_{top}^{repro}) \| (el < \theta_{bottom}^{repro})))$
7-15	Reserved	-

This processing is only applied if

— the local loudspeaker setup is signalled in the LoudspeakerRendering(), and

— the speakerLayoutType is 0 or 1.

Any signaling of 'known Positions' (in LoudspeakerRendering()) shall not be taken into account.

Multiple instances of the object renderer module may be needed to render the content to the 'group-related' reproduction layouts (group-related target rendering layouts).

9.7 Interface for channel-based, object-based, and HOA metadata and audio

*Add a new subclause **17.X1***

17.X1 Interfaces for channel-based, object-based, and HOA metadata and audio data

17.X1.1 General

This subclause **17.X1** describes output interfaces for output of un-rendered channels, objects, and HOA content and associated metadata. The implementation of these interfaces is optional, however, if implemented, there are specific parts which shall be included.

17.X1.2 Expectations on external renderers

If the output interfaces defined in this subclause are implemented and used for connecting to external, non-MPEG-H renderers, then these external renderers should apply and handle the metadata provided in this interface and related audio data in the same manner as if internal rendering would be applied. This includes, but is not limited to:

— Correct handling of loudness-related metadata in particular with the aim to preserve intended target loudness
— Preserving artistic intent, such as applying transmitted Downmix matrices correctly
— Rendering spatial attributes of objects appropriately (position, spatial extent, etc.)
— Correct handling of diffuseness, divergence azimuth range and exclusion sector metadata
— Correct handling of group priorities

17.X1.3 Object-based metadata and audio data (object output interface)

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

17.X1.3.1 Introduction

This interface provides the output of metadata and audio data for object-based audio content (distinct objects which have accompanying OAM data, SignalGroupType is equal to SignalGroupTypeObject (given by Signals3d())). It supports the output of metadata and audio data of object-based content after the processing of the element metadata preprocessor module.

The interface is restricted to metadata and audio data for the initial object-based audio elements, that are enabled for playback (switched on), when the interface output data is requested.

The interface does **not support** the output of metadata or audio data of other audio elements.

Note that the output of channel-based elements that are routed to the Object Renderer during tracking processing (see 6.3) is not supported by this interface.

Content that is received by the object output interface should be processed with the same Loudness/DRC processing before output to the interface as if they would be part of the regular output signals. The application of a peak limiter at the end of the processing chain is highly recommended.

The delay of the received signals should be adjusted before output to the interface, such that they have the same delay as the regular output signals. The object renderer output should be muted and not played back in case an output via the object interface is requested.

If an object output interface is provided by an implementation, the following processing shall be applied to the object-based content before the object output interface:

— Processing of the object metadata preprocessor module (e.g. user interaction, screen-related processing, etc.), except diffuseness processing and divergence processing.
— DRC-1
— resampling
— frame truncation (truncation of PCM data)
— DRC-2
— loudness normalization
— application of the peak limiter

Diffuseness processing (i.e. the creation of the diffuse audio part as well as weighting of the direct sound part using the diffuseness-dependent weighting factor as described in 18.X3) is bypassed in case an output of object audio data plus metadata is requested via the object output interface. Diffuseness processing should instead be applied by an external renderer.

The object-based audio element replica created during divergence processing (as defined in 18.1) shall not be put out via the object output interface. The weighting of the original objects with the divergence-dependent weighting factor $\rho_{original}$ shall be bypassed in case an output of object audio data plus metadata is requested via the object output interface. Divergence processing should instead be applied by an external renderer.

If an object output interface is provided by an implementation, the following metadata shall be provided via the application specific interface to be evaluated by possible external renderers:

— number of output objects
— information about audio truncation and number of valid PCM frames for the current frame
— OAM metadata
    — dynamic object priority (if available)
    — object position (azimuth, elevation, radius)
    — spread
    — object gain
— Signal group related metadata
    — static group priority
    — “fixed position” flag
— Enhanced object metadata
    — diffuseness
    — divergence and divergence azimuth range
    — exclusion sector metadata

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

The following sub-clauses provide syntax and semantics for a preferred implementation of such interface. Implementers may choose to provide the information outlined in this preferred implementation in a different way, e.g. using a different structure, different ordering of data, or different data types for conveying the object metadata.

17.X1.3.2 Syntax of an interface for object-based metadata (informative)

Syntax	No. of bits	Mnemonic
mpegh3da_getObjectAudioAndMetadata()		
{		
/* FRAME CONFIGURATION */		
goa_frameLength;	**6**	**uimsbf**
goa_audioTruncation;	**2**	**bslbf**
if (goa_audioTruncation>0) {		
goa_numSamples;	**13**	**uimsbf**
} else {		
goa_numSamples = goa_frameLength << 6;		
}		
/* OBJECT METADATA */		
goa_numberOfOutputObjects;	**9**	**uimsbf**
for (o = 0; o < goa_numberOfOutputObjects; o++) {		
goa_elementID[o];	**9**	**uimsbf**
goa_hasDynamicObjectPriority[o];	**1**	**bslbf**
goa_hasUniformSpread[o];	**1**	**bslbf**
/* OAM Data */		
goa_numOAMframes[o]	**6**	**uimsbf**
for (nf = 0; nf < goa_numOAMframes[o]; nf++) {		
goa_objectMetadataPresent;	**1**	**bslbf**
if (goa_objectMetadataPresent==1) {		
goa_positionAzimuth[o][nf];	**8**	**uimsbf**
goa_positionElevation[o][nf];	**6**	**uimsbf**
goa_positionRadius[o][nf];	**4**	**uimsbf**
goa_objectGainFactor[o][nf];	**7**	**uimsbf**
if (goa_hasDynamicObjectPriority[o]) {		
goa_dynamicObjectPriority[o][nf];	**3**	**uimsbf**
}		
if (goa_hasUniformSpread[o]) {		
goa_uniformSpread[o][nf];	**7**	**uimsbf**
} else {		
goa_spreadWidth[o][nf];	**7**	**uimsbf**
goa_spreadHeight[o][nf];	**5**	**uimsbf**
goa_spreadDepth[o][nf];	**4**	**uimsbf**
}		
}		
}		
/* Signal group related data */		
goa_fixedPosition[o];	**1**	**bslbf**
goa_groupPriority[o];	**3**	**uimsbf**
/* Enhanced Object Metadata */		
goa_diffuseness[o];	**7**	**uimsbf**
goa_divergence[o];	**7**	**uimsbf**

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
        goa_divergenceAzimuthRange[o];                          6      uimsbf
        goa_numExclusionSectors[o];                             4      uimsbf
        for ( s = 0; s < goa_numExclusionSectors[o]; s++) {
            goa_usePredefinedSector[o][s];                      1      bslbf
            if ( goa_usePredefinedSector[o][s] ) {
                goa_excludeSectorIndex[o][s];                   4      uimsbf
            } else {
                goa_excludeSectorMinAzimuth[o][s];              7      uimbsf
                goa_excludeSectorMaxAzimuth[o][s]               7      uimbsf
                goa_excludeSectorMinElevation[o][s];            5      uimbsf
                goa_excludeSectorMaxElevation[o][s]             5      uimbsf
            }
        } /* for ( s = 0; s < goa_numExclusionSectors[o]; s++) */
    } /* for ( o = 0; o < goa_numberOfOutputObjects; o++ ) */
}
```

17.X1.3.3 Semantics of the interface for object-based metadata (informative)

goa_frameLength	This field is used to determine the length of a nominal audio frame in samples: nominal audio frame length = goa_frameLength << 6;
goa_audioTruncation	Determines whether the audio output corresponding to this goa frame has been truncated according to the following table:

Table XXX – truncation direction signaling

value	truncation applied
0	no truncation
1	from the left (beginning of buffer)
2	from the right (end of buffer)

goa_numSamples	Valid output PCM samples the decoder produced corresponding to this goa frame. In case of truncation this can be different from the nominal audio frame length. This value should be evaluated by possible external renderers to ensure the application of the received metadata to the correct audio PCM samples.
goa_numberOfOutputObjects	Number of Output Objects, $n_{\text{obj, out}}$ = goa_numberOfOutputObjects;
goa_elementID	This field gives the mae_metaDataElementID of the current audio object.
goa_hasDynamicObjectPriority	This field defines whether the current object contains a dynamic object priority value.
goa_hasUniformSpread	This field defines whether the current object contains uniform or non-uniform spread values.
goa_numOAMframes	This field gives the number of OAM frames that are available in the current audio frame for the current object. The minimum is 1

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

	OAM frame per audio frame.
goa_objectMetadataPresent	Indicates whether object metadata is present in the current OAM frame

goa_positionAzimuth	This field defines the azimuth position of the current OAM frame of the current object.
	azimuth = (positionAzimuth - 128) * 1.5;
	azimuth = min(max(azimuth, -180), 180);

goa_positionElevation	This field defines the elevation position of the current OAM frame of the current object.
	elevation = (positionElevation - 32) * 3.0;
	elevation = min(max(elevation, -90), 90);

goa_positionRadius	This field defines the radius of the current OAM frame of the current object.
	radius = pow(2.0, (positionRadius / 3.0)) / 2.0;
	radius = min(max(radius[o], 0.5), 16);

goa_objectGainFactor	This field defines the gain of the current OAM frame of the current object.
	gain = pow(10.0, (objectGainFactor - 32.0) / 40.0);
	gain = min(max(gain, 0.004), 5.957);

goa_dynamicObjectPriority	This field defines the dynamic object priority value of the current OAM frame of the current object. This field can take values between 0 and 7.

goa_uniformSpread	This field contains the uniform spread value of the current OAM frame of the current object.
	spread = uniformSpread * 1.5;
	spread = min(max(spread, 0), 180);

goa_spreadWidth	This field contains the spread value in width dimension of the current OAM frame of the current object.
	spread_width = spreadWidth[o] * 1.5;
	spread_width = min(max(spread_width, 0), 180);

goa_spreadHeight	This field contains the spread value in height dimension of the current OAM frame of the current object.
	spread_height = spreadHeight[o] * 3.0;
	spread_height = min(max(spread_height, 0), 90);

goa_spreadDepth	This field contains the spread value in depth dimension of the current OAM frame of the current object.
	spread_depth = (pow(2.0, (spread_depth / 3.0)) / 2.0) – 0.5;
	spread_depth = min(max(spread_depth, 0), 16);

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

goa_fixedPosition	This field defines if the current object's position shall be updated during processing of scene displacement (tracking) data. If the position shall not be updated, the flag is set to 1.
goa_groupPriority	This field defines the priority of the group to which the current object belongs to. It can take integer values between 0 and 7.
goa_diffuseness	This field defines the diffuseness of the group to which the current object belongs to. This field can take values between 0 and 127, corresponding to diffuseness values between 0.0 and 1.0: diffuseness = (goa_diffuseness / 127);
goa_divergence	This field defines the divergence of the objects. The field can take values between 0 and 127, corresponding to divergence values between 0.0 and 1.0: goa_divergence = (goa_divergence / 127);
goa_divergenceAzimuthRange	If the divergence of the object or group is larger than 0.0 (goa_divergence > 0), the goa_divergenceAzimuthRange defines the positioning of the virtual sources. The field can take values between 0 and 63, resulting in azimuth offset angles between 0° and 180°: φ_{offset} = 3.0 · goa_divergenceAzimuthRange; φ_{offset} = min (max (φ_{offset} , 0), 180);
goa_numExclusionSectors	This field defines the number of exclusion sectors that are defined for the current object.
goa_usePredefinedSector	This field defines if an exclusion sector is signaled by a predefined sector index (flag is equal to 1) or if an arbitrary sector is specified (flag is equal to 0).
goa_excludeSectorIndex	This field defines the sector index of a defined exclusion sector.
goa_excludeSectorMinAzimuth	This field defines the minimum excluded azimuth value for a defined exclusion sector. $\varphi_{\text{sector, min}}$ = 3.0 · (excludeSectorMinAzimuth - 63); $\varphi_{\text{sector, min}}$ = min (max ($\varphi_{\text{sector, min}}$, -180), 180);
goa_excludeSectorMaxAzimuth	This field defines the maximum excluded azimuth value for a defined exclusion sector. $\varphi_{\text{sector, max}}$ = 3.0 · (excludeSectorMaxAzimuth - 63); $\varphi_{\text{sector, max}}$ = min (max ($\varphi_{\text{sector, max}}$, -180), 180);
goa_excludeSectorMinElevation	This field defines the minimum excluded elevation value for a defined exclusion sector. $\theta_{\text{sector, min}}$ = 6.0 · (excludeSectorMinElevation - 15);

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

$$\theta_{\text{sector, min}} = \min(\max(\theta_{\text{sector, min}}, -90), 90);$$

goa_excludeSectorMaxElevation This field defines the maximum excluded elevation value for a defined exclusion sector.

$$\theta_{\text{sector, max}} = 6.0 \cdot (\text{excludeSectorMaxElevation} - 15);$$

$$\theta_{\text{sector, max}} = \min(\max(\theta_{\text{sector, max}}, -90), 90);$$

17.X1.4 Channel-based metadata and audio data

17.X1.4.1 Introduction

This interface provides the output of metadata for channel-based audio content (usually one complete signal group with SignalGroupType set to SignalGroupTypeChannels (given by Signals3d())). It supports the output of metadata data of channel-based content after the processing of the element metadata preprocessor module.

The interface is restricted to metadata for the initial channel-based audio elements, that are enabled for playback (switched on), when the interface output data is requested.

The interface does **not support** the output of metadata of other audio elements.

Note that the output of channal-based elements whose positions shall be modified by the element metadata preprocessor is not supported. Instead those elements may be treated as object content with positions defined by the indicated loudspeaker positions.

Content that is received by the channel output interface should be processed with the same Loudness/DRC processing before output to the interface as if they would be part of the regular output signals. The application of a peak limiter at the end of the processing chain is highly recommended.

The delay of the received signals should be adjusted before output to the interface, such that they have the same delay as the regular output signals. The format converter output should be muted and not played back in case an output via the channel interface is requested.

If a channel output interface is provided by an implementation, the following processing shall be applied to the channel-based content before the channel output interface:

- Processing of the metadata preprocessor module (e.g. user interaction)
- DRC-1
- Resampling
- Frame truncation (truncation of PCM data)
- DRC-2
- Loudness normalization
- Application of the peak limiter

If a channel output interface is provided by an implementation, the following metadata shall be provided via the interface to be evaluated by possible external renderers:

- Number of channels
- Number of valid PCM samples for the current frame
- elementIDs for the referenced audio chanels
- Channel configuration
- “fixed position” flag
- Static group priority
- Downmix matrix elements, if transmitted and matching the selected Reproduction Layout (according to 10.3.1)

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

The following sub-clauses provide syntax and semantics for a preferred implementation of such interface. Implementers may choose to provide the information outlined in this preferred implementation in a different way, e.g. using a different structure, different ordering of data, or different data types for conveying the object metadata.

17.X1.4.2 Syntax of an interface for channel-based metadata (informative)

This sub-clause provides syntax and semantics for a preferred implementation of the channel-based metadata interface.

Syntax	No. of bits	Mnemonic
mpegh3da_getChannelMetadata()		
{		
/* FRAME CONFIGURATION */		
gca_frameLength;	**6**	**uimsbf**
gca_audioTruncation;	**2**	**bslbf**
if (gca_audioTruncation>0) {		
gca_numSamples;	**13**	**uimsbf**
} else {		
gca_numSamples = gca_frameLength << 6;		
}		
/* CHANNEL METADATA */		
gca_numberOfOutputChannelGroups;	**9**	**uimsbf**
for (cGrp = 0; cGrp < gca_numberOfOutputChannelGroups; cGrp ++) {		
gca_numberOfChannels[cGrp]	**16**	**uimsbf**
gca_channelLayout[cGrp] = SpeakerConfig3d();		
for (nChn = 0; nChn < gca_numberOfChannels[cGrp]; nChn++ {		
gca_elementID[cGrp][nChn];	**9**	**uimsbf**
}		
/* TRACKING-RELATED METADATA */		
gca_fixedChannelsPosition[cGrp];	**1**	**bslbf**
/* GROUP-RELATED METADATA */		
gca_groupPriority[cGrp];	**3**	**uimsbf**
gca_channelGain[cGrp];	**8**	**uimsbf**
/* DOWNMIX MATRIX ELEMENT */		
gca_downmixAvailable	**1**	**bslbf**
if (gca_downmixAvailable) {		
gca_downmixConfig();		
}		
}		
}		

17.X1.4.3 Semantics of the interface for channel-based metadata (informative)

gca_frameLength	This field is used to determine the length of a nominal audio frame in samples: nominal audio frame length = gca_frameLength << 6
gca_audioTruncation	Determines whether the audio output corresponding to this gca frame has been truncated according to the following table.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table XXX – truncation direction signaling

value	truncation applied
0	no truncation
1	from the left (beginning of buffer)
2	from the right (end of buffer)

gca_numSamples Valid output PCM samples the decoder produced corresponding to this gca frame. In case of truncation this can be different from the nominal audio frame length.

gca_numberOfOutputChannelGroups Definition of the number of output channel groups.

gca_elementID This field gives the elementIDs of the audio channels out of the current channel group.

gca_numberOfChannels This field gives the number of channels present in the current channel group.

gca_channelLayout This field defines the position of each channel

gca_fixedPosition This field defines if the current channels' position shall be updated during processing of scene displacement (tracking) data. If the position shall not be updated, the flag is set to 1.

gca_groupPriority This field defines the priority of the group to which the current object belongs to. It can take integer values between 0 and 7.

gca_channelGain This field defines the gain of the current frame of the current channel group.

gain = pow(10.0, (objectGainFactor - 32.0) / 40.0);

gain = min(max(gain, 0.004), 5.957);

gca_downmixAvailable This flag signals the presence of downmix matrix elements for the given channel group. The transmitted downmix matrix information in downmixConfig() should be evaluated by possible external renderers to preserve highest possible artistic intend from the content creator during the rendering process.

gca_downmixConfig() This is a subset of the original downmixConfig()-element, only including the elements matching the selected Reproduction Layout (according to 10.3.1)

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

17.X1.5 HOA metadata and audio data

17.X1.5.1 Introduction

© ISO/IEC 2017 – All rights reserved

This interface provides the output of metadata for HOA audio content (usually one complete signal group with SignalGroupType set to SignalGroupTypeHOA (given by Signals3d())). It supports the output of the metadata of the HOA content prior the HOA rendering process.

The interface is restricted to metadata for the decoded HOA coefficients, that are enabled for playback (switched on), when the interface output data is requested.

The interface does not support the output of metadata of other audio elements.

Content that is received by the HOA output interface should be processed with the same Loudness/DRC processing before output to the interface as if they would be part of the regular output signals. The application of a peak limiter at the end of the processing chain is highly recommended.

The delay of the received signals should be adjusted before output to the interface, such that they have the same delay as the regular output signals. The HOA interface and the HOA renderer should not perform simultaneously: In case an output via the HOA interface is requested, the output of the HOA renderer should be muted and not play back audio. In opposite, in case the output of the HOA renderer is requested the HOA output interface should be inactive.

If the HOA output interface is provided by an implementation, the following processing shall be applied to the HOA content before the HOA output interface:

— Processing of the metadata preprocessor module
— DRC-1
— Resampling
— Frame truncation (truncation of PCM data)
— DRC-2
— Loudness Normalization
— Peak limitation

If the HOA output interface is provided by an implementation, the following metadata shall be provided via the interface to be evaluated by possible external renderers:

— HOA order
— Number of valid PCM samples for the current frame
— NFC metadata
— A flag that indicates if HOA content is relative to a screen and if so, the local screen size information
— HOA rendering matrix elements, if transmitted and matching the selected reproduction layout

The following sub-clauses provide syntax and semantics for a preferred implementation of such interface. Implementers may choose to provide the information outlined in this preferred implementation in a different way, e.g. using a different structure, different ordering of data.

17.X1.5.2 Syntax of an interface for HOA metadata (informative)

Syntax	No. of bits	Mnemonic
mpegh3da_getHoaMetadata()		
{		
/* FRAME CONFIGURATION */		
gha_frameLength;	**6**	**uimsbf**
gha_audioTruncation;	**2**	**bslbf**
if (gha_audioTruncation>0) {		
gha_numSamples;	**13**	**uimsbf**
} else {		
gha_numSamples = gha_frameLength << 6;		
}		
/* HOA METADATA */		
gha_numberOfHoaGroups;		**9**
for (hGrp = 0; hGrp < gha_numberOfHoaGroups; hGrp ++) {		
gha_HoaOrder[hGrp]**;**	**9**	**uimsbf**
gha_UsesNfc[hGrp];	**1**	**bslbf**
if (gha_UsesNfc[hGrp]) {		
gha_NfcReferenceDistance[hGrp];	**32**	**bslbf**

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

}		
gha_hasSignalledHoaMatrix[hGrp];	**1**	**uimsbf**
if (gha_hasSignalledHoaMatrix[hGrp]) {		
gha_HoaRenderingMatrixSet();		
}		
gha_isScreenRelative[hGrp];	**1**	**uimsbf**
if (gha_isScreenRelated[hGrp]) {		
mae_ProductionScreenSizeData();		
}		
}		
}		

17.X1.5.3 Semantics of the interface for HOA metadata (informative)

gha_frameLength	This field is used to determine the length of a nominal audio frame in samples: nominal audio frame length = gca_frameLength << 6
gha_audioTruncation	Determines whether the audio output corresponding to this gha frame has been truncated according to the following table:

Table XXX – truncation direction signaling

value	truncation applied
0	no truncation
1	from the left (beginning of buffer)
2	from the right (end of buffer)

gha_numSamples	Valid output PCM samples the decoder produced corresponding to this gha frame. In case of truncation this can be different from the nominal audio frame length.
gha_HoaOrder	The HOA order of the outputted audio content
gha_ numberOfHoaGroups	This elements indicates the number signal groups of type SignalGroupTypeHOA.
gha_UsesNfc	This element determines whether or not the HOA Near Field Compensation (NFC) has been applied to the coefficient signals.
gha_NfcReferenceDistance	This element determines the radius in meter that has been used for the HOA NFC (interpreted as float in IEEE 754 format in little-endian).
gha_hasSignalledHoaMatrix	This element indicates if the content was transmitted with HOA rendering matrices in the bitstream.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

gha_HoaRenderingMatrixSet	This is a subset of the original HOARenderingMatrixSet()-element, only including the elements matching the selected Reproduction Layout.
gha_isScreenRelative	This element indicates if the HOA representation shall be rendered with respect to the reproduction screen size.

17.X1.6 Audio PCM data

The PCM data of the channels and objects interfaces shall be provided through the decoder PCM buffer, which first contains the regular rendered PCM signals (e.g. 12 signals for a 7.1+4 setup). Subsequently $n_{\text{chan, out}}$ additional signals carry the PCM data of the originally transmitted channel representation. These are followed by $n_{\text{obj, out}}$ signals carrying the PCM data of the un-rendered output objects. Then additional signals carry the $n_{\text{HOA, out}}$ HOA data which number is indicated in the HOA metadata interface via the HOA order. The HOA audio data in the HOA output interface is provided in the so-called equivalent spatial domain representation. The conversion from the HOA domain into the equivalent spatial domain representation and vice versa is described in Annex C.5.1.

The decoder shall signal the offset index of the PCM buffer for the first un-rendered output object.

9.8 Diffuseness Rendering

9.8.1 Diffuseness Processing

Add a new subclause ***18.X3***

18.X3 Diffuseness Rendering

To render objects whose groups have a ‘diffuseness’ parameter bigger than zero, a weighted sum of a so-called direct sound part and a diffuse sound part has to be calculated depending on the value of the diffuseness parameter before playback.

Two signal versions are created for each object with a diffuseness value bigger than zero: A ‘direct sound part’ and a ‘diffuse sound part’.

The direct sound part is the normal output of the metadata preprocessor, including all processing steps that are related to element metadata preprocessing, such as

- Spread processing
- Position and gain interaction
- Closest speaker playout processing
- Screen-related remapping and zooming
- Processing of excluded sectors
- etc.

The diffuse sound part is created by replicating the object audio content to the number N, with N being the number of total available reproduction loudspeakers without LFEs.

Each of these N signals is filtered with a decorrelation filter. Gain interaction is applied here as well, such that an interaction gain affects both parts equally.

Both the direct sound part and the diffuse sound part are in addition weighted with a gain factor, which is dependent of the diffuseness value of the group, to which the current object belongs to.

The following functions for the two weights g_{dir}, g_{diff} are defined:

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

$$g'_{\mathrm{diff}} = \sqrt{\mathrm{diffuseness}}$$
$$g_{\mathrm{dir}} = \sqrt{1.0 - g'_{\mathrm{diff}}}$$
$$g_{\mathrm{diff}} = g'_{\mathrm{diff}} \cdot \sqrt{N}$$

EXAMPLE — diffuseness = 0.0 → only the direct part is playing

$$g_{\mathrm{dir}} = 1.0, \quad g'_{\mathrm{diff}} = 0.0$$

EXAMPLE — diffuseness = 1.0 → only the diffuse part is playing

$$g_{\mathrm{dir}} = 0.0, \quad g'_{\mathrm{diff}} = 1.0$$

EXAMPLE — diffuseness = 0.5 → both parts are playing with equal weights

$$g_{\mathrm{dir}} = \sqrt{0.5}, \quad g'_{\mathrm{diff}} = \sqrt{0.5}$$

The direct sound part is sent to the object renderer; the diffuse sound part is directly sent to the mixer, where the two paths are combined and a mix of direct sound and diffuse sound is created. Note that the diffuse part does not contain signals for LFE speakers.

The routing of the diffuse part to the mixer shall be disabled in case an output via the object interface is requested. The weighting of the direct sound part with the factor g_{dir} shall also be omitted.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

The overall processing is depicted in Figure AMD7.2.

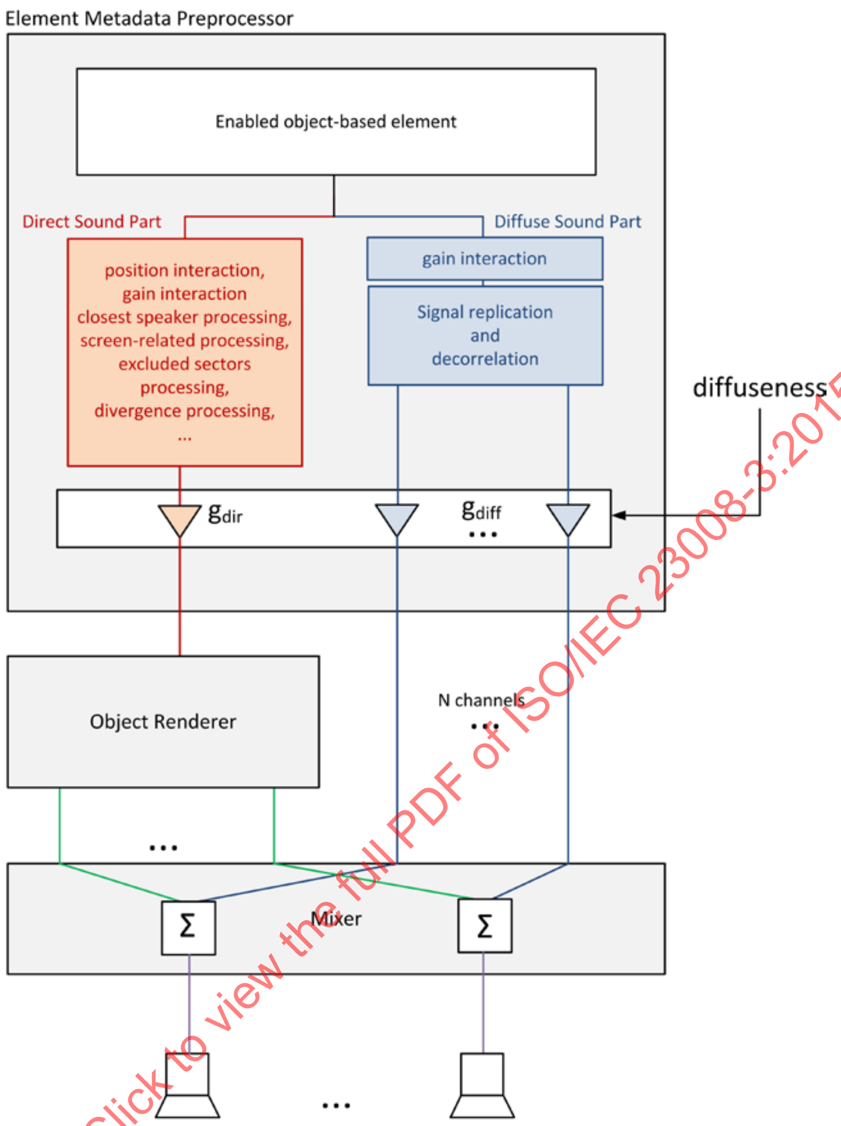

Element Metadata Preprocessor
Enabled object-based element
Direct Sound Part
Diffuse Sound Part
position interaction, gain interaction closest speaker processing, screen-related processing, excluded sectors processing, divergence processing, ...
gain interaction
Signal replication and decorrelation
diffuseness
g_{dir}
g_{diff}
Object Renderer
N channels
Mixer
Σ
Σ

Figure AMD7.2 - Diffuseness processing

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

9.8.2 Informative decorrelation filtering for diffuseness processing

Add a new ***Annex J***

Annex J

(informative)

Decorrelation filtering for 'diffuseness' processing

Decorrelation filters for the 'diffuseness' processing are designed in time-frequency domain (e.g. QMF domain, ERB (Equivalent Rectangular Bandwidth) domain, STFT domain).

A set of frequency-dependent (random / pseudo-random) delays is defined, resulting in one specific delay value $\delta[b]$ per frequency band b. One decorrelation filter is defined for each reproduction loudspeaker / virtual speaker. The delays are static within time.

The decorrelation filters are implemented as FIR filters and filtering is performed to time-domain signals.

The following parameters can be adjusted:

- Overall (broadband) minimum and maximum delay values $\delta_{\min}$ and $\delta_{\max}$
- Frequency-dependent minimum and maximum delay values $\delta_{\min}[b]$ and $\delta_{\max}[b]$, e.g. limitation of the delay per band to be proportional to the longest waveform period present in the current band, resulting in a maximum absolute delay per band

Decorrelation filters according to the described design paradigm are already in use for different purposes and applications, for examples see References [1], [2] and [3].

Based on the existing implementations, the following parameter values are recommended to use:

- The minimum overall delay $\delta_{\min}$ is set to 5ms [1, 2].
- Below 1500Hz: The maximum delay $\delta_{\max}[b]$ is 50 times the cycle time of the frequency band with an upper limit of 100ms [1].
- Above 1500Hz: The maximum delay $\delta_{\max}[b]$ is always 50ms. The minimum delay $\delta_{\min}[b]$ is 10 times the cycle time of the frequency band [1].

In addition, it is also possible to define further enhancement steps, which improve the diffuseness perception and minimize temporal artifacts. Examples for enhancement processing are, e.g.:

- Delays at cutoff frequencies are chosen in a specific way to avoid artifacts at borders between frequency bands, e.g. by defining the delays in a way such that the phase matches at the cutoff frequency where the delay is changed [1], for instance by ensuring that the phase shift at cutoff frequencies have to be multiples of 360° [3].
- Pseudo-randomization of the delays, such that they are distributed equally to the full delay range in each frequency band. This can be ensured by dividing the delay range into as many equal-length intervals as there are channels, and randomizing a delay from a uniform distribution for each interval. Resulting delays are then assigned to the different filters (channels) in random order [2].

Exception of transient signal parts before application of decorrelation filters: An additional step is added before the filtering that separates the transients from the signal (e.g. by using an energy-based transient detection). The signal without transients is then filtered with the decorrelation filters and the original transients are added back to the decorrelated signal afterwards [1].

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

9.9 Updates of the element metadata preprocesssor

*In **17.7.3** replace*

The second mode is a basic interaction mode, where the user may choose one of the group presets that are defined in the metadata audio element syntax. With a group preset, the on/off statuses of the groups that are referenced in the conditions of the chosen preset are defined and cannot be changed by the user. The user may only change the on/off status of the other groups that are not referenced in the conditions of the chosen preset. Position and gain of all groups can be changed according to the defined restrictions and ranges.

with

The second mode is a basic interaction mode, where the user may choose one of a set group presets that are defined in the metadata audio element syntax. With a group preset, the on/off statuses of the groups and switch groups that are referenced in the conditions of the chosen preset are defined and cannot be changed by the user. The user may only change the on/off status of the other groups and switch groups that are not referenced in the conditions of the chosen preset. Position and gain of all groups can be changed according to the defined restrictions and ranges.

*In **17.7.3** add*

If the current group preset or group preset extension contains one or more switch group conditions, the following behavior has to be ensured by the metadata preprocessor:

— If a switch group condition is equal to 1, it has to be ensured that one member of the corresponding switch group is enabled (on/off status equal to 1). If no member is switched on by the user (via interaction), the default member has to be switched on by the metadata preprocessor.
— All preset-dependent object characteristics and parameters (disable gain interactivity, group preset gain, disable position interactivity, group preset azimuth offset, group preset elevation offset, group preset distance factor) shall be applied to the active member, independent on which member is currently active.
— If a switch group condition is equal to 0, all members of the corresponding switch group shall be set to inactive/disabled.

Switch group conditions with value 0 are only applicable for switch groups whose mae_switchGroupAllowOnOff flag is equal to 1.

*Change the title of **18** from*

Application and processing of local setup information and interaction data

to

Application and processing of local setup information, interaction data and scene displacement data

*In **18.1** replace*

A processing block is defined that processes the user interaction interface data and prepares the audio elements for rendering and play-out. This “Element Metadata Preprocessor” also applies the logic that is imposed by the user interaction and the element metadata.

The following processing structure and order of processing the interface data (interactivity interface mpegh3daElementInteraction() and interface for local setup information mpegh3daLocalSetupInformation()) and metadata audio elements is defined. An overview is given in Figure 70.

with

A processing block is defined that processes the user interaction interface data, the scene displacement interface data and prepares the audio elements for rendering and play-out. This “Element Metadata Preprocessor” also applies the logic that is imposed by the user interaction and the element metadata.

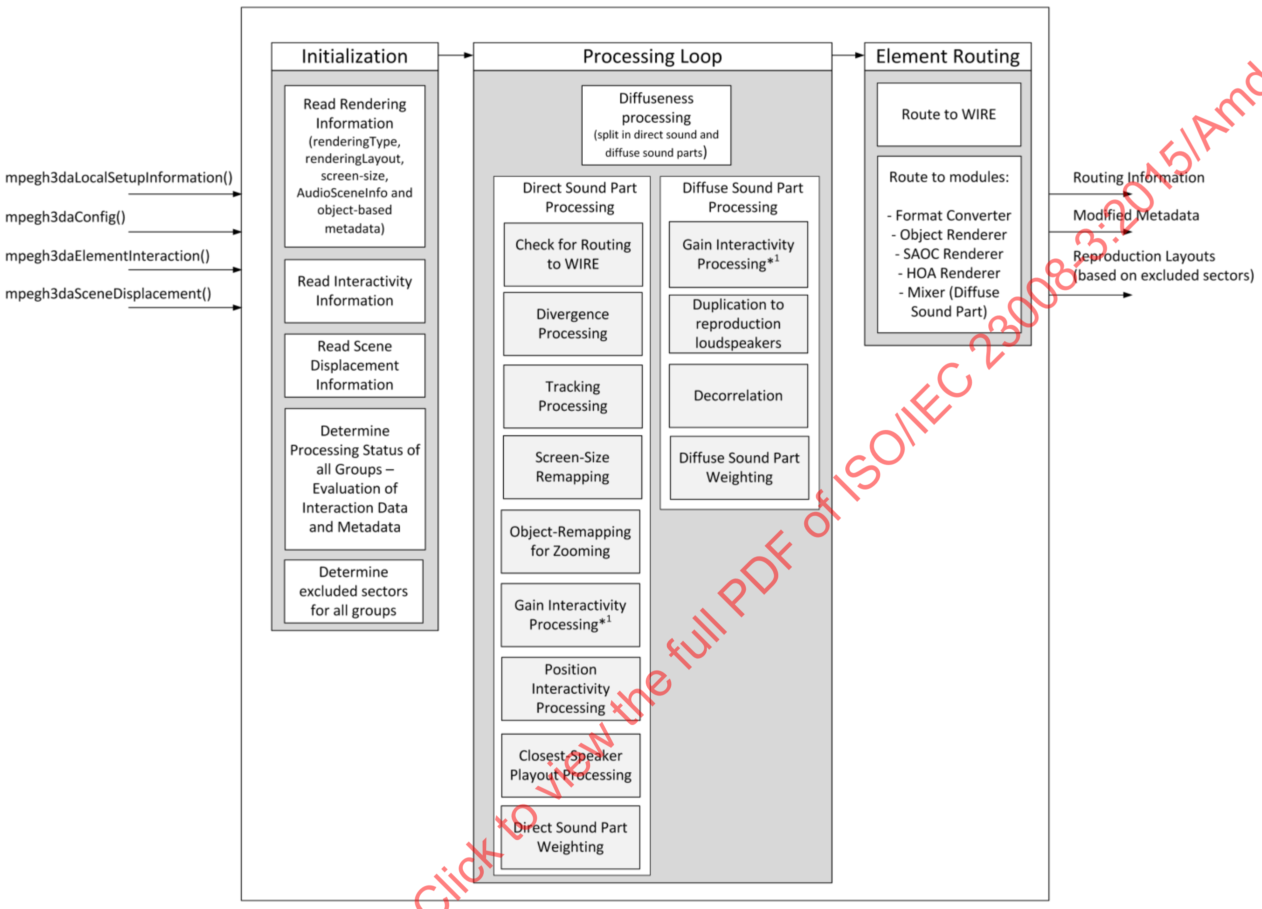
The following processing structure and order of processing the interface data (interactivity interface mpegh3daElementInteraction(), scene displacement interface mpegh3daSceneDisplacementData() and

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

interface for local setup information mpegh3daLocalSetupInformation()) and metadata audio elements is defined. An overview is given in Figure 70.

The divergence processing has to be conducted before the processing of scene displacement angles (tracking processing) is carried out.

*Replace Figure **70** in **18.1** by the following Figure:*


mpegh3daLocalSetupInformation()
mpegh3daConfig()
mpegh3daElementInteraction()
mpegh3daSceneDisplacement()
Initialization
Read Rendering Information (renderingType, renderingLayout, screen-size, AudioSceneInfo and object-based metadata)
Read Interactivity Information
Read Scene Displacement Information
Determine Processing Status of all Groups – Evaluation of Interaction Data and Metadata
Determine excluded sectors for all groups
Processing Loop
Diffuseness processing (split in direct sound and diffuse sound parts)
Direct Sound Part Processing
Check for Routing to WIRE
Divergence Processing
Tracking Processing
Screen-Size Remapping
Object-Remapping for Zooming
Gain Interactivity Processing*1
Position Interactivity Processing
Closest Speaker Playout Processing
Direct Sound Part Weighting
Diffuse Sound Part Processing
Gain Interactivity Processing*1
Duplication to reproduction loudspeakers
Decorrelation
Diffuse Sound Part Weighting
Element Routing
Route to WIRE
Route to modules:
- Format Converter
- Object Renderer
- SAOC Renderer
- HOA Renderer
- Mixer (Diffuse Sound Part)
Routing Information
Modified Metadata
Reproduction Layouts (based on excluded sectors)

*1 The gain interactivity block may be processed at an arbitrary position in the metadata processing loop, hoewever it has to be ensured that both the direct sound part and the diffuse sound part are adjusted in their gain according to the gain interaction data.

Figure 70 — Processing of Metadata and Interface Data

*In **18.1** replace*

As a first step the rendering type, rendering layout and screen size information is determined. Then, the syntax element mae_AudioSceneInfo() is retrieved. Afterwards, the interactivity data is read from the interactivity interface.

The interaction type (basic or advanced) is determined. Next, it is determined which element groups have to be processed. This is defined by their on/off status, the WIRE routing information and the implicit logic by switch group definitions.

with

As a first step the rendering type, rendering layout and screen size information is determined. Then, the syntax element mae_AudioSceneInfo() is retrieved.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

The overall number of object-based audio element is determined that need to be rendered by the object renderer. The overall number of object-based audio elements is

— the sum of the number of elements with OAM data
— plus 2 times the number of elements with OAM data with a divergence value bigger than 0 and whose groups are not marked to be sent to a WIRE output.

Afterwards, the interactivity data is read from the interactivity interface.

The interaction type (basic or advanced) is determined. Next, it is determined which elements have to be processed. This is defined by the on/off status of their corresponding groups, the WIRE routing information and the implicit logic by switch group definitions.

*Further, in **18.1** replace*

The following groups have to be processed:

— All groups that are switched on and are not member of a switch group.

— All groups that are switched on and that are member of a switch group if the switch group condition is fulfilled (a maximum of one member of this switch group is switched on).

— All groups that are switched off but are default members of a switch group from whose members no other member is marked as switched on. The on/off status of this default group shall then be set to 1 (on).

— All groups that are switched off and are marked to be routed to a WIRE output.

with

The elements from the following groups have to be processed:

— All groups that are switched on and are not member of a switch group.

— All groups that are switched on and that are member of a switch group if the switch group condition is fulfilled (a maximum of one member of this switch group is switched on).

— All groups that are switched off but are default members of a switch group from whose members no other member is marked as switched on. The on/off status of this default group shall then be set to 1 (on).

— All groups that are switched off and are marked to be routed to a WIRE output.

*Further, in **18.1** replace*

In the basic interaction mode, the user can choose one of the defined group presets. The on/off statuses of the groups that are referenced in the chosen group preset's conditions are set according to these conditions. Any discrepancy to these values by the values in the ei_GroupInteractivityStatus() syntax element shall be ignored.

with

In the basic interaction mode, the user can choose one of a set of defined group presets.

First, it has to be checked if a downmixId is present and if the chosen preset has a group present extension (defined in mae_GroupPresetDefinitionExtension()) that references the current downmixId. If this is the case, the conditions of the corresponding group preset extension have to be used to determine which groups shall be played back / rendered. The on/off statuses of the groups and switch groups that are referenced in the chosen group preset's or the applicable group preset extension's conditions are set according to these

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

conditions. Any discrepancy to these values by the values in the ei_GroupInteractivityStatus() syntax element shall be ignored.

*In **18.1** after*

NOTE The relevant groups are known at this point in time so the core decoder could be controlled such that irrelevant elements are not decoded.

add

Next, group-related reproduction layouts are determined by evaluation of excluded sectors per group, according to 18.X2. The group-related reproduction layouts (group-related target rendering layouts) are used to control one or more instances of the object renderer.

Next, the possible different reproduction layouts are determined by evaluation of excluded sectors, according to 18.X2. The different reproduction layouts (target rendering layouts) are used to control one or more instances of the object renderer.

and replace

Then the relevant groups are prepared for playout and rendering. Therefore, updated output element characteristics are calculated for each element of each group: Gain, azimuth, elevation, and distance. Azimuth and elevation either represent the element's position or the element's speaker's position (in case the mae_closestSpeakerPlayout flag is set to one). In case of groups with SignalGroupTypeChannels, azimuth and elevation contain the unmodified speaker's position.

with

Then the elements of the relevant groups are prepared for playout and rendering. Therefore, updated output element characteristics are calculated for each element of each group and the additional virtual objects for groups with a divergence value bigger than 0 that are not to be sent to a WIRE output: Gain, azimuth, elevation, and distance. Azimuth and elevation either represent the element's position or the element's speaker's position (in case the closestSpeakerPlayout flag is set to one and the divergence of the corresponding group is equal to 0 or not existing). In case of groups with SignalGroupTypeChannels, azimuth and elevation contain the unmodified speaker's position (in case the divergence related to the element is equal to 0 or not existing and the closestSpeakerPlayout flag is set to one and the closest speaker playout is unconditioned, or in case the closestSpeakerPlayout flag is set to one and the speaker is located within the range defined by the closestSpeakerThresholdAngle as defined in 18.X if the closest speaker playout is conditioned).

*Further, in **18.1** replace*

Azimuth and elevation either represent the element's position or the element's speaker's position (in case the mae_closestSpeakerPlayout flag is set to one).

with

Azimuth and elevation either represent the element's position or the element's speaker's position (in case the closestSpeakerPlayout flag is set to one and the closest speaker playout is unconditioned, or in case the closestSpeakerPlayout flag is set to one and the speaker is located within the range defined by closestSpeakerThresholdAngle as defined in 18.X if the closest speaker playout is conditioned).

*In **18.1** after*

For the determination of the updated element characteristics, it shall first be checked if the element group are marked to be routed to a WIRE output.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

add

Any evaluation of a valid/selected preset (and therefore the application of preset-dependent values) shall only happen in the defined 'basic interaction mode' (see 17.7.3). The chosen/selected preset is indicated by the presetID that is received by the mae_elementInteraction() interface. Any preset-dependent value (e.g. *group preset gain)* is therefore only applied if basic interaction mode is enabled.

*Further, in **18.1** replace*

If a group is not routed to a WIRE output (non-WIRE group), the Screen-Related Remapping is conducted for all group members that are marked as screen-related. If a group is marked as to be routed to a WIRE output (WIRE group), this step should be skipped.

with

If a group is not routed to a WIRE output (non-WIRE group), the Screen-Related Remapping is conducted for all group members that are marked as screen-related. If a group is marked as to be routed to a WIRE output (WIRE group), this step should be skipped.

After that, the divergence processing is conducted. Therefore, the objects with a divergence bigger than 0 are replicated: Two additional 'virtual objects' are created in addition to each object with divergence > 0. The 'unmodified' positions and gains for the 'virtual' objects for the reproduction of objects with a divergence bigger than 0 are determined according to:

$$\varphi_{\text{virtual, left}} = (\varphi_{\text{orig, OAM}} + \varphi_{\text{offset}})$$

if ($\varphi_{\text{virtual, left}} > 180$) { $\varphi_{\text{virtual, left}} = \varphi_{\text{virtual, left}} - 360$) }

if ($\varphi_{\text{virtual, left}} < -180$) { $\varphi_{\text{virtual, left}} = \varphi_{\text{virtual, left}} + 360$) }

$$\varphi_{\text{virtual, right}} = (\varphi_{\text{orig, OAM}} - \varphi_{\text{offset}})$$

if ($\varphi_{\text{virtual, right}} > 180$) { $\varphi_{\text{virtual, right}} = \varphi_{\text{virtual, right}} - 360$) }

if ($\varphi_{\text{virtual, right}} < -180$) { $\varphi_{\text{virtual, right}} = \varphi_{\text{virtual, right}} + 360$) }

The gains ρ are based on the current OAM gain~~, the current gain interaction~~ and the divergence value:

$$\rho_{\text{orig, temp}} = \sqrt{1 - \varepsilon_{\text{divergence}}{}^{0.58497}}$$

$$\rho_{\text{original}} = \rho_{\text{orig, temp}} \cdot \rho_{\text{orig, OAM}}$$

$$\rho_{\text{virtual, temp}} = \sqrt{0.5 \cdot (\varepsilon_{\text{divergence}})^{0.58497}}$$

$$\rho_{\text{virtual, left}} = \rho_{\text{virtual, right}} = \rho_{\text{virtual, temp}} \cdot \rho_{\text{orig, OAM}}$$

After that, the virtual objects are treated the same way as the original objects from the bitstream for the calculation of the updated output element characteristics.

In particular, the virtual objects are assigned with the same isScreenRelativeObject-value as their corresponding original object for screen-related remapping and zooming. If the gain interaction is applied after the divergence processing, all ranges and restrictions of the original objects shall apply to their corresponding virtual object copies as well.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

The duplicated objects are added to the same group as the original object. They are assigned the original group-dependent characteristics, e.g. for the subsequent interaction processing. As a next step, the scene displacement data is processed if the 'useTrackingMode' flag in either binauralRendering() or loudspeakerRendering() is equal to one. Therefore the data from the scene displacement interface is read and updated positions for all tracked objects and channels (indicated by 'fixedPosition' = 0) are calculated as defined in 18.X1.

and replace

Afterwards, the gain interactivity data is processed (for non-WIRE groups and WIRE groups). Therefore, it is first checked if the current group allows for gain interactivity (mae_allowGainInteractivity flag is equal to 1).

with

Afterwards, the gain interactivity data is processed. This step is applied to all elements of non-WIRE groups and WIRE groups. Therefore, it is first checked if the group of the current element allows for gain interactivity (mae_allowGainInteractivity flag is equal to 1).

Further replace

If the gain interactivity is still enabled, the output gain is calculated for each element of the group, taking into account the interactivity gain modification, and either the OAM gain (if available) or the current gain of an element (in case no OAM data is available). If the basic interaction mode is enabled, the possible group gain value of the currently active/valid preset is also taken into account.

The overall gain modification (interactivity gain in dB plus group gain in dB) shall be restricted according to the given mae_interactivityMinGain and mae_interactivityMaxGain values. If no gain interaction is allowed, the output gain shall either contain 0dB (in case no OAM data is available) or the OAM gain in dB.

If the group is marked as WIRE output, then no further interactivity processing shall be applied.

If the group is not marked as WIRE output, then the application of further processing depends on the signal type. If the group contains channel-based elements (SignalGroupTypeChannels), then no further interactivity processing shall be applied. Instead the position of the predefined associated speaker shall be determined from the audioChannelLayout information for each element of the group. It is assumed that the member elements in the member list are ordered with respect to the channel ordering signaled in the syntax element Signals3d().

If the group is accompanied by OAM data, then the position interactivity shall be processed. Therefore, it is first checked if the current group allows for position interactivity (mae_allowPositionInteractivity flag is equal to1).

with

Note: The gain interactivity may be processed at an arbitrary position in the metadata processing chain. If the gain interactivity is conducted after the divergence processing, the replicated objects should be processed with the same gain interaction as the corresponding original objects.

If the gain interactivity is still enabled, the output gain is calculated for each element of the group plus additional virtual objects if the group has a divergence value bigger than 0 (in case divergence processing is conducted before gain interaction), taking into account the interactivity gain modification, and either the OAM gain (if available), the unmodified gain of the virtual objects (in case divergence processing is conducted before gain interaction) or the current gain of an element (in case no OAM data is available). If the basic interaction mode is enabled, the possible group gain value of the currently active/valid preset is also taken into account.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

The overall gain modification (interactivity gain in dB plus group gain in dB) shall be restricted according to the given mae_interactivityMinGain and mae_interactivityMaxGain values of the group of the current element. If no gain interaction is allowed, the output gain shall either contain 0dB (in case no OAM data is available), the OAM gain in dB or respectively the unmodified gain in dB for the virtual objects if divergence processing is conducted before gain interaction.

If the group of the current element is marked as WIRE output, then no further interactivity processing shall be applied.

If the group of the current element is not marked as WIRE output, then the application of further processing depends on the signal type. If the element is a channel-based elements (SignalGroupTypeChannels), then no further interactivity processing shall be applied. Instead the position of the predefined associated speaker shall be determined from the audioChannelLayout information. It is assumed that the member elements in the member list of the associated group are ordered with respect to the channel ordering signaled in the syntax element Signals3d().

If the current element is accompanied by OAM data, then the position interactivity shall be processed. Therefore, it is first checked if the group of the current element allows for position interactivity (mae_allowPositionInteractivity flag is equal to1).

and replace

If that is the case and if the basic interaction mode is active, it is then checked if the currently active/valid preset disables the gain interactivity.

If the gain interactivity is still enabled, the output gain is calculated for each element of the group, taking into account the interactivity gain modification, and either the OAM gain (if available) or the current gain of an element (in case no OAM data is available). If the basic interaction mode is enabled, the possible group gain value of the currently active/valid preset is also taken into account.

with

If that is the case and if the basic interaction mode is active, it is then checked if the currently active/selected preset or the applicable group preset extension for the current downmixId disables the gain interactivity.

If the gain interactivity is still enabled, the output gain is calculated for each element of the group, taking into account the interactivity gain modification, and either the OAM gain (if available) or the current gain of an element (in case no OAM data is available). If the basic interaction mode is enabled, the possible group gain value of the currently active/selected preset or the applicable group preset extension for the current downmixId is also taken into account.

Further, replace

If that is the case and if basic interaction mode is active, it is then checked if the currently active/valid preset disables the position interactivity.

If the position interactivity is still enabled for the current group, azimuth, elevation and distance values are determined for each element of the group. The calculation of the output values takes into account the interactivity modification (offset values and distance multiplication factor) and the values from the OAM data. If the basic interaction mode is enabled, the possible values for a group azimuth offset, group elevation offset and group distance factor defined in the preset shall be taken into account.

The overall interactivity modification (interactivity modification plus the corresponding values in the preset) shall be restricted according to the given interactivity ranges.

with

If that is the case and if basic interaction mode is active, it is then checked if the currently active/valid preset or the applicable group preset extension for the current downmixId disables the position interactivity.

If the position interactivity is still enabled for the current group, azimuth, elevation and distance values are determined for each element of the group and the additional virtual objects if the divergence of the group is

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

bigger than 0. The calculation of the output values takes into account the interactivity modification (offset values and distance multiplication factor) and the values from the OAM data or respectively the unmodified gain in dB for the virtual objects. If the basic interaction mode is enabled, the possible values for a group azimuth offset, group elevation offset and group distance factor defined in the preset or the applicable group preset extension for the current downmixId shall be taken into account.

The overall interactivity modification (interactivity modification plus the corresponding values in the preset) shall be restricted according to the given interactivity ranges.

In addition to that, replace

As a next step, the closest speaker playout processing shall be conducted. If the group is marked as closest speaker playout, then the position of the closest speaker to the output position data is determined for each member element as defined in 18.5. No rendering shall be applied; therefore it has to be ensured that the determined speaker position exists in the reproduction speaker setup.

After processing the interactivity data, the routing information for the elements is determined. If an element group is marked as WIRE output, it should be directly sent to the according WIRE ID. The output gain is sent along and shall be applied before WIRE output. Metadata regarding gain interactivity as well as Loudness/DRC processing should be applied before outputting the audio signal. Content that is routed to a WIRE output should be processed with the same Loudness/DRC processing as if they would be part of the regular output signals. The application of a peak limiter at the end of the processing chain is highly recommended. The signals on the WIRE output should have the same delay as signals after the mixer.

If a group is not marked as WIRE output, then the output depends on the signal type.

with

As a next step, the closest speaker playout processing shall be conducted. If an element is marked as closest speaker playout, it shall be evaluated as the first step if the closest speaker playout processing shall happen unconditioned or conditioned.

If the closest speaker playout is unconditioned, the position of the closest speaker to the output position data is determined as defined in 18.5, taking into account all reproduction speakers. No rendering shall be applied; therefore it has to be ensured that the determined speaker position exists in the reproduction speaker setup.

If the closest speaker playout is conditioned (a closestSpeakerThresholdAngle value is given), it has to be determined, which of the reproduction speakers lies in the given range according to 18.X.

After determining the list of speakers in the range, the position of the closest speaker with respect to the output position data is determined as defined in 18.5, taking into account only the reproduction speakers in the range. No rendering shall be applied; therefore it has to be ensured that the determined speaker position exists in the reproduction speaker setup.

After processing the interactivity data, the routing information for the elements is determined. If an element group is marked as WIRE output, the members of this group should be directly sent to the according WIRE ID. The output gain is sent along and shall be applied before WIRE output. Metadata regarding gain interactivity as well as Loudness/DRC processing should be applied before outputting the audio signal. Content that is routed to a WIRE output should be processed with the same Loudness/DRC processing as if they would be part of the regular output signals. The application of a peak limiter at the end of the processing chain is highly recommended. The signals on the WIRE output should have the same delay as signals after the mixer.

If an element group is not marked as WIRE output, then the output depends on the signal type.

Further, replace

Channel-based groups are marked to be sent to the format converter, object-based groups are marked to be sent to the object renderer, SAOC-based groups are marked to be sent to the SAOC renderer, HOA-based groups are marked to be sent to the HOA renderer.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

with

Channel-based elements with 'fixedPosition = 1' are marked to be sent to the format converter, object-based element as well as channel-based groups with 'fixedPosition = 0' are marked to be sent to the object renderer, SAOC-based elements are marked to be sent to the SAOC renderer, HOA-based elements are marked to be sent to the HOA renderer.

After

In addition to the decoded audio data, each renderer gets azimuth, elevation, distance (either element's position or element's speaker's position), gain and the rendering layout. The SAOC renderer and the HOA renderer might need additional side information.

add

The group-dependent reproduction layouts depending on the signaled excluded sectors are sent along to the object renderer. Dependent on the number of individual reproduction layouts, multiple instances of the object renderer are used.

*In **18.2.3** replace*

Position interactivity is only defined for groups where positional information is provided (OAM data in the syntax element object_metadata(), see **Error! Reference source not found.**).

For groups with SignalGroupTypeChannels, no position interactivity is defined, because they are intended to be played back by a specific speaker.

For groups that are marked as "closestSpeakerPlayout", the determination of the closest speaker should be conducted after processing the position interactivity data.

If an element group is marked to be routed to a WIRE output, position interactivity shall not be applied. The unrendered audio content is sent to the WIRE output and no positional information is sent along.

with

Position interactivity is only defined for elements where positional information is provided (OAM data in the syntax element object_metadata(), see **Error! Reference source not found.**).

For elements with SignalGroupTypeChannels, no position interactivity is defined, because they are intended to be played back by a specific speaker.

For elements that are marked as "closestSpeakerPlayout", the determination of the closest speaker should be conducted after processing the position interactivity data.

If an element group is marked to be routed to a WIRE output, position interactivity shall not be applied to the members of the group. The unrendered audio content is sent to the WIRE output and no positional information is sent along.

*In **18.2.4** replace*

Screen-Related Element Remapping and Object Remapping for Zooming are only defined for groups that are accompanied by OAM data in the syntax element object_metadata().

If no local screen size information is available, no screen-related element remapping shall be applied. In case only azimuth screen size information is given, the decoder shall not apply any screen-related element remapping of the elevation of screen-related elements.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

If no local zoom information is available, no object remapping for zooming shall be applied.

If an element group is marked to be routed to a WIRE output, screen-related element remapping and object remapping for zooming shall be skipped, because the unrendered audio content is sent to the WIRE output without positional information.

with

Screen-Related Element Remapping and Object Remapping for Zooming are only defined for elements that are accompanied by OAM data in the syntax element object_metadata().

If no local screen size information is available, no screen-related element remapping shall be applied. In case only azimuth screen size information is given, the decoder shall not apply any screen-related element remapping of the elevation of screen-related elements.

If no local zoom information is available, no object remapping for zooming shall be applied.

If an element group is marked to be routed to a WIRE output, screen-related element remapping and object remapping for zooming shall be skipped for the members of this group, because the unrendered audio content is sent to the WIRE output without positional information.

*In **18.2.5** replace*

Closest speaker playout is only defined for groups that are accompanied by OAM data in the syntax element object_metadata().

If an element group is marked to be routed to a WIRE output, signaling of the closestSpeakerPlayout option shall be ignored, because the unrendered audio content is sent to the WIRE output.

with

Closest speaker playout is only defined for elements that are accompanied by OAM data in the syntax element object_metadata().

If an element group is marked to be routed to a WIRE output, signaling of the closestSpeakerPlayout option shall be ignored for the members of this group, because the unrendered audio content is sent to the WIRE output.

*In **18.2.5** add*

If an element has a divergence value bigger than 0, the signaling of the closestSpeakerPlayout option shall be ignored and no closest speaker playout processing shall be conducted. Closest speaker playout processing shall only be conducted for objects with spread=0° (uniform spread signaling) or spread_width=spread_height=0° (non-uniform spread signaling).

9.10 Review of Metadata

Add new subclause 5.2.2.X and add the following syntax table:

5.2.2.X Syntax of Signal Group Information

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Table 4 — Syntax of SignalGroupInformation()

Syntax	No. of bits	Mnemonic
SignalGroupInformation(bsNumSignalGroups)		
{		
for (grp = 0; grp < bsNumSignalGroups + 1 ; grp++) {		
groupPriority[grp];	**3**	**uimsbf**
fixedPosition[grp];	**1**	**uimsbf**
}		
}		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

At the end of 5.3 add new subclause 5.3.X Signal Group Information Elements

5.3.X Signal Group Information Elements

groupPriority	This field defines the priority of the group. It can take integer values between 0 and 7. The group may be discarded from rendering and decoding if the priority is lower than 7. If groups are discarded, the groups with lowest priority should be discarded first.
fixedPosition	This field defines if the positions of the members of a group shall be updated in the context of processing of tracking data (scene displacement data). In case the flag is equal to zero, the position of the corresponding group's members are updated during the processing of scene displacement angles. In case the flag is equal to one, the positions of the corresponding group's members are not updated during the processing of scene displacement angles. The default value for fixedPosition is zero for all groups.

Replace all occurrences of mae_fixedObjectPosition *with* fixedPosition.

Replace all occurrences of fixedObjectPosition *with* fixedPosition.

In 15.3. remove the following definitions:

mae_groupPriority	This field defines the priority of the group. It can take integer values between 0 and 7. The group may be discarded from rendering and decoding if the priority is lower than 7. If groups are discarded, the groups with lowest priority should be discarded first.
mae_closestSpeakerPlayout	This flag defines that the members of the metadata element group should not be rendered but directly be played back by the loudspeakers which are nearest to the geometric position of the members as defined in **Error! Reference source not found.**.

At the end of 7.3, add new subclause 7.3.X Enhanced Object Metadata and add the following syntax:

7.3.X Enhanced Object Metadata

Table 5 — Syntax of EnhancedObjectMetadataConfig()

Syntax	No. of bits	Mnemonic
EnhancedObjectMetadataConfig()		
{		
/* static per group */		
hasDiffuseness;	**1**	**bslbf**
if (hasDiffuseness) {		
hasCommonGroupDiffuseness;	**1**	**bslbf**
}		
}		
hasExcludedSectors;	**1**	**bslbf**
if (hasExcludedSectors) {		
hasCommonGroupExcludedSectors;	**1**	**bslbf**
if (hasCommonGroupExcludedSectors) {		
useOnlyPredefinedSectors;	**1**	**bslbf**
}		
}		
hasClosestSpeakerCondition;	**1**	**bslbf**
if (hasClosestSpeakerCondition) {		
closestSpeakerThresholdAngle;	**7**	**uimbsf**
}		
/* static per object */		
for (o = 0; o < num_objects; o++) {		
hasDivergence[o]**;**	**1**	**bslbf**
if (hasDivergence[o]) {		
divergenceAzimuthRange[o]**;**	**6**	**uimsbf**
}		
if (hasCommonGroupExcludedSectors == 0) {		
useOnlyPredefinedSectors[o];	**1**	**bslbf**
}		
}		
}		

Table 6 — Syntax of EnhancedObjectMetadataFrame()

Syntax	No. of bits	Mnemonic
EnhancedObjectMetadataFrame()		
{		
/* dynamic per group */		
if (hasDiffuseness && hasCommonGroupDiffuseness) {		
if (independencyFlag == 0) {		
keepDiffuseness;	**1**	**bslbf**
} else {		
keepDiffuseness = 0;		
}		
if (keepDiffuseness == 0) {		
diffuseness;	**7**	**uimsbf**
}		
}		
if (hasCommonGroupExcludedSectors) {		
if (independencyFlag == 0) {		
keepExclusion;	**1**	**bslbf**
} else {		
keepExclusion = 0;		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
        }
        if ( keepExclusion == 0 ) {
            numExclusionSectors;                                          4          uimbsf
            if ( useOnlyPredefinedSectors ) {
                for ( sc = 0; sc < numExclusionSectors; sc++) {
                    excludeSectorIndex[sc];                               4          uimbsf
                }
            } else {
                for ( sc = 0; sc < numExclusionSectors; sc++) {
                    usePredefinedSector[sc];                              1          uimbsf
                    if (usePredefinedSector[sc] ) {
                        excludeSectorIndex[sc];                           4          uimbsf
                    } else {
                        excludeSectorMinAzimuth[sc];                      7          uimbsf
                        excludeSectorMaxAzimuth[sc];                      7          uimbsf
                        excludeSectorMinElevation[sc];                    5          uimbsf
                        excludeSectorMaxElevation[sc];                    5          uimbsf
                    }
                }
            }
        }
    }

    /* dynamic per object */
    for ( o = 0; o < num_objects; o++ ) {
        closestSpeakerPlayout[o];                                         1          bslbf

        if ( hasDiffuseness && ( hasCommonGroupDiffuseness == 0 ) ) {
            if ( independencyFlag == 0 ) {
                keepDiffuseness[o];                                       1          bslbf
            } else {
                keepDiffuseness[o] = 0;
            }
            if (keepDiffuseness[o] == 0) {
                diffuseness[o];                                           7          uimsbf
            }
        }

        if ( hasDivergence[o] ) {
            if ( independencyFlag == 0 ) {
                keepDivergence[o];                                        1          bslbf
            } else {
                keepDivergence[o] = 0;
            }
            if (keepDivergence[o] == 0) {
                divergence[o];                                            7          uimsbf
            }
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
            if ( hasCommonGroupExcludedSectors == 0 ) {
                if ( independencyFlag == 0 ) {
                    keepExclusion[o];                                   1          bslbf
                } else {
                    keepExclusion[o] = 0;
                }
                if (keepExclusion[o] == 0 )
                {
                    numExclusionSectors[o];                             4          uimbsf
                    if ( useOnlyPredefinedSectors[o] ) {
                        for ( sc = 0; sc < numExclusionSectors[o]; sc++) {
                            excludeSectorIndex[o][sc];                  4          uimbsf
                        }
                    } else {
                        for ( sc = 0; sc < numExclusionSectors[o]; sc++) {
                            usePredefinedSector[o][sc];                 1          uimbsf
                            if (usePredefinedSector[o][sc] ) {
                                excludeSectorIndex[o][sc];              4          uimbsf
                            } else {
                                excludeSectorMinAzimuth[o][sc];         7          uimbsf
                                excludeSectorMaxAzimuth[o][sc];         7          uimbsf
                                excludeSectorMinElevation[o][sc];       5          uimbsf
                                excludeSectorMaxElevation[o][sc];       5          uimbsf
                            }
                        }
                    }
                }
            }
        }
}
```

At the end of 7.4 add a new subclause 7.4.X :

7.4.X Enhanced Object Metadata

7.4.X.1 Enhanced Object Metadata Configuration Semantics

hasDiffuseness	Flag which indicates whether diffuseness information is present in the payload frame.
hasCommonGroupDiffuseness	Flag which indicates whether the transmitted diffuseness information applies to a whole group or to individual objects.
hasExcludedSectors	Flag which indicates whether information about exclusion sectors is present in the payload frame.
hasCommonGroupExcludedSectors	Flag which indicates whether the transmitted information about exclusion sectors applies to a whole group or to individual objects.
useOnlyPredefinedSectors	This flag determines whether all exclusion sectors are chosen from the set of predefined sectors (value of 1) as identified by a table entry or whether sectors may also be signaled by means of explicit azimuth and elevation ranges in the bitstream (value of 0).
hasClosestSpeakerCondition	If the 'closest speaker playout flag' of the current group is set to 1, it is possible to restrict the processing to loudspeakers that are located in a specified area around the members of the group. This flag defines if the

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

	‘closest speaker processing’ shall happen unconditioned (value of 0) or conditioned (value of 1).
closestSpeakerThresholdAngle	If the ‘closest speaker processing’ shall only happen if one or more loudspeakers are located in a defined area around the members of the group, the threshold angle for this area is given by this field. φ_{thresh} = 1.5 · closestSpeakerThresholdAngle; φ_{thresh} = min (max (φ_{thresh} , 0), 180); θ_{thresh} = 1.5 · closestSpeakerThresholdAngle; θ_{thresh} = min (max (θ_{thresh} , 0), 90);
hasDivergence	Flag which indicates whether divergence information is present in the payload frame.
divergenceAzimuthRange	If the divergence of the object or group is larger than 0.0 (divergence > 0), the divergenceAzimuthRange defines the positioning of the virtual sources. The field can take values between 0 and 63, resulting in azimuth offset angles between 0° and 180°: φ_{offset} = min(max(3.0 · divergenceAzimuthRange, 0),180);

7.4.X.2 Enhanced Object Metadata Frame Semantics

keepDiffuseness	Flag which indicates whether the diffuseness from the previous frame shall be re-used for the current frame (value of 0) or whether a new diffuseness shall be transmitted in the subsequent bit stream field.
diffuseness	Definition of the diffuseness of the objects (of a group); this field can take values between 0 and 127, corresponding to diffuseness values between 0.0 and 1.0: diffuseness = (diffuseness / 127);
keepExclusion	Flag which indicates whether the exclusion sectors from the previous frame shall be re-used for the current frame (value of 0) or whether new exclusion sectors shall be transmitted in the subsequent bit stream fields.
numExclusionSectors	This field defines the number of sectors/areas that shall be excluded from rendering the of the affected object(s). It allows for values between 0 and 15. A value of 0 indicates that no loudspeakers shall be excluded.
usePredefinedSector	This flag defines if the following sector is a predefined one (value of 1) identified by a table entry or if a detailed sector definition by azimuth and elevation ranges follows in the bitstream (value of 0).
excludeSectorIndex	Identifier of the predefined exclusion sector as defined in Table AMD7.5.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table AMD7.5 — Value of excludeSectorIndex

Value of excludeSectorIndex	Short description	Explanation
0	No positive elevation	Exclude all speaker with positive elevation angles
1	No negative elevation	Exclude all speakers with negative elevation angles
2	No front	Exclude all front speakers
3	No right side	Exclude all right side speakers
4	No left side	Exclude all left side speakers
5	No surround	Exclude all surround speakers
6	Screen only	Exclude all speakers that are not located in the reproduction screen area
7-15	Reserved	n/a

excludeSectorMinAzimuth This field defines the minimum azimuth of the excluded area.

$\varphi_{\text{sector, min}}$ = 3.0 · (excludeSectorMinAzimuth - 63);

$\varphi_{\text{sector, min}}$ = min (max ($\varphi_{\text{sector, min}}$, -180), 180);

excludeSectorMaxAzimuth This field defines the maximum azimuth of the excluded area.

$\varphi_{\text{sector, max}}$ = 3.0 · (excludeSectorMaxAzimuth - 63);

$\varphi_{\text{sector, max}}$ = min (max ($\varphi_{\text{sector, max}}$, -180), 180);

excludeSectorMinElevation This field defines the minimum elevation of the excluded area.

$\theta_{\text{sector, min}}$ = 6.0 · (excludeSectorMinElevation - 15);

$\theta_{\text{sector, min}}$ = min (max ($\theta_{\text{sector, min}}$, -90), 90);

excludeSectorMaxElevation This field defines the maximum elevation of the excluded area.

$\theta_{\text{sector, max}}$ = 6.0 · (excludeSectorMaxElevation - 15);

$\theta_{\text{sector, max}}$ = min (max ($\theta_{\text{sector, max}}$, -90), 90);

closestSpeakerPlayout This flag defines that the object shall not be rendered with the object renderer but instead directly be played back by the loudspeaker which is nearest to the geometric position of the object as defined in **Error! Reference source not found.**.

keepDivergence Flag which indicates whether the divergence from the previous frame shall be re-used for the current frame (value of 0) or whether a new divergence shall be transmitted in the subsequent bit stream field.

divergence This field defines the divergence of the objects (of a group). The field can take values between 0 and 127, corresponding to divergence values between 0.0 and 1.0:

divergence = (divergence / 127);

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

In 15.2 replace Table 143 with the following:

Table 143 — Syntax of mae_AudioSceneInfo()

Syntax	No. of bits	Mnemonic
mae_AudioSceneInfo()		
{		
mae_isMainStream;	**1**	**bslbf**
if (mae_isMainStream) {		
mae_audioSceneInfoIDPresent;	**1**	**bslbf**
if (mae_audioSceneInfoIDPresent) {		
mae_audioSceneInfoID;	**8**	**uimsbf**
}		
mae_numGroups;	**7**	**uimsbf**
mae_GroupDefinition(mae_numGroups);		
mae_numSwitchGroups;	**5**	**uimsbf**
mae_SwitchGroupDefinition(mae_numSwitchGroups);		
mae_numGroupPresets;	**5**	**uimsbf**
mae_GroupPresetDefinition(mae_numGroupPresets);		
mae_Data();		
mae_metaDataElementIDoffset = 0;		
mae_metaDataElementIDmaxAvail;	**7**	**uimsbf**
}		
else {		
mae_bsMetaDataElementIDoffset;	**7**	**uimsbf**
mae_metaDataElementIDoffset = mae_bsMetaDataElementIDoffset + 1;		
mae_metaDataElementIDmaxAvail;	**7**	**uimsbf**
}		
}		

In 15.2 modify Table 145 as follows:

Table 7 — Syntax of mae_GroupDefinition()

Syntax	No. of bits	Mnemonic
mae_GroupDefinition(numGroups)		
{		
for (grp = 0; grp < numGroups; grp++) {		
mae_groupID[grp];	**7**	**uimsbf**
mae_allowOnOff[grp];	**1**	**bslbf**
mae_defaultOnOff[grp];	**1**	**bslbf**
mae_allowPositionInteractivity[grp];	**1**	**bslbf**
if (mae_allowPositionInteractivity[grp]) {		
mae_interactivityMinAzOffset[grp];	**7**	**uimsbf**
mae_interactivityMaxAzOffset[grp];	**7**	**uimsbf**
mae_interactivityMinElOffset[grp];	**5**	**uimsbf**
mae_interactivityMaxElOffset[grp];	**5**	**uimsbf**
mae_interactivityMinDistFactor[grp];	**4**	**uimsbf**
mae_interactivityMaxDistFactor[grp];	**4**	**uimsbf**
}		
mae_allowGainInteractivity[grp];	**1**	**bslbf**
if (mae_allowGainInteractivity[grp]) {		
mae_interactivityMinGain[grp];	**6**	**uimsbf**
mae_interactivityMaxGain[grp];	**5**	**uimsbf**
}		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

~~**mae_groupPriority**[grp];~~	~~3~~	~~uimsbf~~
~~**mae_closestSpeakerPlayout**[grp];~~	~~1~~	~~bslbf~~
mae_bsGroupNumMembers[grp];	7	uimsbf
mae_hasConjunctMembers[grp];	1	bslbf
if (mae_hasConjunctMembers[grp]) {		
mae_startID[grp];	7	uimsbf
}		
else {		
for (obj = 0; obj < mae_bsGroupNumMembers[grp] + 1; obj++) {		
mae_metaDataElementID[grp][obj];	7	uimsbf
}		
}		
}		
}		

At the end of 15.2 add the following syntax table:

Table AMD.15.2.1 — Syntax of mae_DrcUserInterfaceInfo()

Syntax	No. of bits	Mnemonic
mae_DrcUserInterfaceInfo()		
{		
version;	**2**	**uimsbf**
if (version==0) {		
bsNumTargetLoudnessConditions;	**3**	**uimsbf**
targetLoudnessValueLower[0] = -63;		
for(c=0; c<numTargetLoudnessConditions; c++) {		
bsTargetLoudnessValueUpper[c];	**6**	**uimsbf**
drcSetEffectAvailable[c];	**16**	**bslbf**
targetLoudnessValueLower[c+1] = targetLoudnessValueUpper[c];		
}		
} else {		
/* discard remaining bits signaled by mae_dataLength */		
}		
}		

In 15.3, add the following Semantics

mae_metaDataElementIDmaxAvail	This field signals the maximum available mae_metaDataElementID in a Main Stream or Sub-Stream.
version	A version field that shall be set to zero.
bsNumTargetLoudnessConditions	A field that signals the number of target loudness conditions. The field can take values between 0 and 7.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

numTargetLoudnessConditions = **bsNumTargetLoudnessConditions** + 1;

bsTargetLoudnessValueUpper A field that signals the upper limit of the target loudness range defined by targetLoudnessValueUpper/-Lower. The field can take values between -63 and 0 dB. A range check shall include the upper boundary value and exclude the lower boundary value. If the requested decoder target loudness (targetLoudness) includes any of the signaled ranges, the corresponding drcSetEffectAvailable field can, e.g. be used for guidance of a user interface if the content of the DRC bitstream is not available.

targetLoudnessValueUpper in dB = **bsTargetLoudnessValueUpper** – 63;

drcSetEffectAvailable This field signals the available DRC set effects in the audio bitstream for a certain target loudness range. Each bit represents a requestable DRC set effect type, where Table 11 of ISO/IEC 23003-4:2015 defines bit positions by index (0 corresponds to LSB) and the corresponding semantics. If not present, all bits shall be set to one. Note that the DRC set selection process according to Clause 6.4.4 can handle any request independent of availability in the bitstream.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

9.11 References

Add the following entries to the clause "Bibliography" which is located after the last Annex:

[1] Laitinen, M.-V. et al.: "Reproducing Applause-Type Signals with Directional Audio Coding". J. Audio Eng. Soc., Vol. 59, No. 1/2, 2011

[2] Vilkamo, J. et al.: "Directional Audio Coding: Virtual Microphone-Based Synthesis and Subjective Evaluation". J. Audio Eng. Soc., Vol. 57, No. 9, 2009

[3] Bouéri, M., Kyriakakis, C.: "Audio Signal Decorrelation Based on a Critical Band Approach". 117th AES Convention, San Francisco, USA, 2004

[4] Laitinen, M.-V. et al.: "Auditory Distance rendering using a standard 5.1 loudspeaker setup". 139th AES Convention, New York, USA, 2015

10 Improvements for use in broadcast ecosystems

10.1 Order of elements in mpegh3daDecoderConfig() and mpegh3daFrame()

In 5.3.1 replace:

Signal groups are represented by audio data elements which are configured in the mpegh3daDecoderConfig(). The bitstream element Signals3d() specifies the assignment of audio data elements to signal groups. In case a signal group requires extension payloads, the corresponding mpegh3daExtElementConfig()s shall directly follow the configuration elements for the audio data elements which belong to the associated signal group.

with:

Signal groups are represented by audio data elements which are configured in the mpegh3daDecoderConfig(). The bitstream element Signals3d() specifies the assignment of audio data elements to signal groups. In case a signal group requires extension payloads, the corresponding mpegh3daExtElementConfig()s shall immediately precede the configuration elements for the audio data elements which belong to the associated signal group.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

10.2 Overall delay alignment and constant decoder delay

In 4.3 replace Table 1 with

Table 1 — MPEG-H 3DA functional blocks, internal processing domain and delay numbers. $f_{s,core}$ denotes the core decoder output sampling rate, $f_{s,out}$ denotes the decoder output sampling rate"

Pro-cessing Context	**Functional Block**	**Processing Domain**	**Delay** Samples $[1/f_{s,core}]$ or $[1/f_{s,out}]$	**Contribution to Maximum Delay** **High Profile** Samples $[1/f_{s,out}]$	**Contribution to Maximum Delay** **Low Complexity Profile** Samples $[1/f_{s,out}]$
Audio Core		**TD,** Core frame length = 1024	0		
	SBR / MPS212	**FD,** Core frame length = 2048 or 4096	0		
	SBR/MPS212, stereoConfigIndex==3	**FD,** Core frame length = 2048 or 4096	384	**384 * *RSR*$_{max}$**	
	MPS	**FD**	640	**640 * *RSR*$_{max}$**	
	HREP	**TD,** Core frame length = 1024	64		
	HREP, QMF-Synthesis and QMF-Analysis pair and alignment to 64 sample grid	**FD TD FD**	64 + 257 + 320 + 63	**(64 + 257 + 320 + 63) * *RSR*$_{max}$**	
	QMF-Analysis	**TD, FD**	320	**320 * *RSR*$_{max}$**	
	QMF-Synthesis	**FD, TD**	257	**257 * *RSR*$_{max}$**	
	Hybrid filter/SBR	**FD**	384	**384 * *RSR*$_{max}$**	
Rendering	DRC-1	if multiband: **TD (STFT)** or **FD** else: **neutral**	0		
	Format Converter, Core frame length = 1024	**FD**	3072	**3072 * *RSR*$_{max}$**	
	Format Converter, Core frame length = 2048 or 4096	**FD**	2048		
	STFT Format Converter	**TD (STFT)**	256		**256 * *RSR*$_{max}$**
	Object Renderer	**Neutral**	0		
	SAOC 3D Decoder, bsDoubleFrameLengthFlag=0	**FD**	0		
	SAOC 3D Decoder, bsDoubleFrameLengthFlag=1 Core frame length = 1024	**FD**	1024		
	SAOC 3D Decoder, bsDoubleFrameLengthFlag=1 Core frame length = 2048	**FD**	2048		
	HOA Decoder	**TD, (FD)***	0 | 577		
	HOA Decoder with multiband DRC-1	**TD, FD or TD (STFT),**	**577 | 256**		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

		TD			
	Sample Rate Converter	**FD, TD**	256	**256**	**256**
Mixing	Mixer	**FD, TD**	0		
Maximum accumulated delay to mixer:				**5057 * RSR_{max} + 256**	**256 * RSR_{max} + 256**
Post-processing	QMF-Analysis	**TD, FD**	320	**320**	**320**
	QMF-Synthesis	**FD, TD**	257	**257**	**257**
	DRC-2	if multiband: **FD** else: **neutral**	0		
	FD Binauralizer	**FD**	0		
	TD Binauralizer	**TD**	0		
End of chain	DRC-3 (only singleband)	**TD**	0		
	Loudness Normalization	**TD**	0		
	Peak Limiter (NOTE1)	**TD**	240	**240**	**240**
	LS Distance Compensation	**TD**	0		
Maximum accumulated delay **to decoder output: (NOTES 1, 2)**				5057 * RSR_{max} + 1073 **= 16244**	256 * RSR_{max} + 1073 **= 1841**
NOTE 1: The given values are valid for $f_{s,out}$ =48kHz. The peak limiter shall introduce a delay of 5ms, i.e. the limiter delay in samples shall be determined as floor($f_{s,out}$ · 5/kHz). The maximum accumulated delay to the decoder output has to be adapted accordingly for output sampling rates the differ from $f_{s,out}$ =48kHz.					
NOTE 2: The maximum resampling factor RSR_{max} of the sampling rate converter is 3.					
*) HOA Decoder: FD in case Subband Directional Prediction or PAR is used (only High Profile), Low Complexity Profile only operates in TD					

Replace 4.4.2 with

4.4.2 Mixing

— The mixer may operate in **TD** or in **FD**

After 4.4.2 add a new subclause (4.4.3):

4.4.3 DRC-1 Operation Domains (DRC in Rendering Context)

— If multiband DRC-1 data is present in the bitstream for a DRC-1 module
  — the corresponding DRC-1 processing block should operate in **FD** or **TD (STFT)**

After subclause 4.4, add the following subclause

4.5 Decoder Delay

If a constant decoder delay application is signaled in the bitstream (**receiverDelayCompensation**==1, as defined in 5.2.2) then the decoding and rendering delay from the IMDCT output of the core decoder to the mixing block of the MPEG-H 3D Audio decoder shall be kept constant by introducing delay lines where

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

required. Similarly, the overall delay from the IMDCT output to the decoder output shall be kept constant by introducing delay lines where required. The constant delay values that shall be fulfilled are determined by the ***Maximum accumulated delay*** numbers in Table 1.

Enforcing a constant delay from the IMDCT output to the mixer implies that all signals at the IMDCT output shall be aligned. Accordingly, the decoder shall compensate for different delays that may occur in the processing of the IMDCT output signals, e.g. different SBR/MPS processing delays for SCEs and CPEs in case of stereoConfigIndex==3.

Further, all side information utilized in the rendering blocks as well as for DRC processing shall be sent aligned to the IMDCT output waveforms independent of the value of receiverDelayCompensation. The rendering and DRC side information shall be applied delayed in the rendering/DRC blocks. The side information delays are determined by the delays the waveforms encounter in the processing pipeline from the IMDCT output until reaching the corresponding rendering/DRC blocks. The following kinds of side information shall be aligned to the iMDCT output waveforms and shall be applied delayed as defined above:

- OAM object metadata
- DRC side information
- SAOC side information
- HOA side information

In case a bitstream contains multiple signal groups with different signalGroupTypes, the bitstream element receiverDelayCompensation shall be set to 1.

In Table 2 in 5.2.2.1, General Configuration Syntax, replace

coreQmfDelayCompensation

with

reserved

In Table 2 in 5.3.2, General Configuration Data Elements, replace

coreQmfDelayCompensation	This flag forces the MPEG-H 3D Audio Core coder to apply a delay compensation for the case that eSBR is not applied. If its value is 0, no delay compensation shall be applied. If its value is 1, a delay of 577 samples shall be applied to all audio elements which do not use eSBR, i.e. which do not employ a QMF analysis during decoding. Further, if processing blocks immediately following the MPEG-H 3D Audio Core require QMF domain input, the delay is bypassed and the output is fed to a QMF analysis instead, such that the MPEG-H 3D Audio Core decoder and the subsequent processing blocks can be efficiently combined in the QMF domain. Audio elements which *do* employ eSBR are not affected.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

receiverDelayCompensation	This flag forces the format converter to apply a delay compensation for the case that it is not active. When its value is 0, no delay compensation shall be applied in the bypass path. When its value is 1, a delay of 4033 samples shall be applied, when the format converter is fed with time-domain input, a delay of 2432 samples (corresponding to 38 QMF time slots) shall be applied, when the format converter is fed with QMF domain input.

with

reserved	reserved.
receiverDelayCompensation	This flag forces the decoder to operate in a constant delay. The decoder delay shall be kept constant by employing appropriate delay lines to obtain the delays noted in Table 1.

10.3 Broadcast Contribution Mode Operation of MPEG-H

Add the following new subclause "4.5 Contribution Mode of MPEG-H 3d audio":

4.5 Contribution Mode of MPEG-H 3d audio

The Contribution Mode of MPEG-H 3DA specifies a generic transport mechanism for audio signals with accompanying metadata and it is designed to be unaware of the signal type and of the content and structure of the associated metadata.

A value of **speakerLayoutType** == 3 as defined in Table 41 in the signaling of the **referenceLayout** in the **mpegh3daConfig()** indicates that MPEG-H 3D audio shall operate in Contribution Mode. In Contribution Mode the rendering context shall operate in a pass-through mode, i.e. the format converter shall apply an identity matrix to the signal. Contribution Mode bitstreams shall have the following additional restrictions:

— bsNumSignalGroups == 0 (one single signal group)

— SignalGroupType == 0 (channel signal group)

— differsFromReferenceLayout[0] == 0 (no audioChannelLayout in the signal group)

— core coder delay lines for compensation of SBR, MPS212, shall be applied

— Content of extension elements shall not be processed in the decoder but shall be made available to an external framework

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

In 5.3.3 replace Table 41:

Table 41 — Meaning of speakerLayoutType

Value	Meaning
0	Loudspeaker layout is signaled by means of ChannelConfiguration index as defined in ISO/IEC 23001-8.
1	Loudspeaker layout is signaled by means of a list of LoudspeakerGeometry indices as defined in ISO/IEC 23001-8
2	Loudspeaker layout is signaled by means of a list of explicit geometric position information.
3	*Reserved*

with:

Table 41 — Meaning of speakerLayoutType

Value	Meaning
0	Loudspeaker layout is signaled by means of ChannelConfiguration index as defined in ISO/IEC 23001-8.
1	Loudspeaker layout is signaled by means of a list of LoudspeakerGeometry indices as defined in ISO/IEC 23001-8
2	Loudspeaker layout is signaled by means of a list of explicit geometric position information.
3	Contribution Mode: No associated loudspeaker layout; Renderers shall operate as defined in 4.5. This value is only allowed when signaling the **referenceLayout**.

10.4 Audio Pre-Roll

In 5.2.3.3, replace Table 38 with:

Table 8 — Syntax of AudioPreRoll()

Syntax	No. of bits	Mnemonic
AudioPreRoll()		
{		
configLen = escapedValue(4,4,8);	**4..16**	
Config()	**8*configLen**	
applyCrossfade;	**1**	**bool**
reserved;	**1**	**bool**
numPreRollFrames = escapedValue(2,4,0);	**2..6**	
for (frameIdx=0; frameIdx < numPreRollFrames; ++frameIdx) {		
auLen = escapedValued(16,16,0)	**16..32**	**uimsbf**
AccessUnit()	**8*auLen**	
}		
}		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Replace the content of 5.5.6.1, General:

The *AudioPreRoll()* syntax element is used to transmit audio information of previous frames along with the data of the present frame. The additional audio data can be used to compensate the decoder startup delay (pre-roll), thus enabling random access at stream access points (SAP) that make use of *AudioPreRoll().*

A *mpegh3daExtElement()* with the usacExtElementType of ID_EXT_ELE_AUDIOPREROLL shall be used to transmit the *AudioPreRoll()*.

with the following:

The *AudioPreRoll()* syntax element is used to transmit audio information of previous frames along with the data of the present frame. The additional audio data can be used to initialize the decoder processing pipeline (pre-roll), thus enabling random access at stream access points (SAP) that make use of *AudioPreRoll()*, seamless transition between different codec configurations including bitrate adaptation.

A *mpegh3daExtElement()* with the usacExtElementType of ID_EXT_ELE_AUDIOPREROLL shall be used to transmit the *AudioPreRoll()*.

Replace the content of 5.5.6.2, Semantics:

configLen	Size of the configuration syntax element in bytes.
Config()	The decoder configuration syntax element. In the context of this standard this shall be the Mpegh3daConfig() as defined in 5.2.2.1. The Config() field may be transmitted to be able to respond to changes in the audio configuration (e.g. switching of streams).
numPreRollFrames	The number of pre-roll access units (AUs) transmitted as audio pre-roll data. The reasonable number of AUs depends on the decoder start-up delay.
auLen	AU length in bytes.
AccessUnit()	The pre-roll AU(s).
applyCrossfade	If this flag is set to 1, a linear crossfade shall be applied in case of configuration change, as defined in 5.5.6.3.3.
reserved	reserved bit shall be zero.

NOTE The pre-roll data carried in the extension element may be transmitted “out of band”, i. e. the buffer requirements may not be satisfied

In order to use *AudioPreRoll()* for both random access and bitrate adaptation the following restrictions apply:

— The first element of every frame shall be an extension element (mpegh3daExtElement) of type ID_EXT_ELE_AUDIOPREROLL.

— The corresponding mpegh3daExtElement() shall be configured as specified in Table 60.

— Consequently, if pre-roll data is present, this mpegh3daFrame() shall start with the following bit sequence:

  — “1”: usacIndependencyFlag.

  — “1”: usacExtElementPresent (referring to audio pre-roll extension element).

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

- "0": usacExtElementUseDefaultLength (referring to audio pre-roll extension element).

— If no *AudioPreRoll()* is transmitted, the extension payload shall not be present (usacExtElementPresent = 0).

— The pre-roll frames with index "0" shall be independently decodable, i.e. *usacIndependencyFlag* shall be set to "1".

- If an extension element (mpegh3daExtElement) of type ID_EXT_ELE_SAOC_3D is present, in the pre-roll frames with index "0" bsIndependencyFlag shall be set to "1" in the extension payload Saoc3DFrame().

Table 60 — Setup of mpegh3daExtElementConfig() for AudioPreRoll()

Bitstream Field	Value
usacExtElementType	ID_EXT_ELE_AUDIOPREROLL
usacExtElementConfigLength	0
usacExtElementDefaultLengthPresent	0
usacExtElementPayloadFrag	0

with the following:

configLen	Size of the configuration syntax element in bytes.
Config()	The decoder configuration syntax element. In the context of this standard this shall be the Mpegh3daConfig() as defined in 5.2.2.1. The Config() field may be transmitted to be able to respond to changes in the audio configuration (e.g. switching of streams).
numPreRollFrames	The number of pre-roll access units (AUs) transmitted as audio pre-roll data. Typically a value of 1 is signaled for initializing inverse transform filterbanks of the core decoder and the renderers.
auLen	AU length in bytes.
AccessUnit()	The pre-roll AU(s).

NOTE The pre-roll data carried in the extension element may be transmitted "out of band", i. e. the buffer requirements may not be satisfied

In order to use *AudioPreRoll()* for both random access and seamless configuration changes the following restrictions apply:

— The first element of every frame shall be an extension element (mpegh3daExtElement) of type ID_EXT_ELE_AUDIOPREROLL.

— The corresponding mpegh3daExtElement() shall be configured as specified in Table 60.

— Consequently, if pre-roll data is present, this mpegh3daFrame() shall start with the following bit sequence:

- "1": usacIndependencyFlag.

- "1": usacExtElementPresent (referring to audio pre-roll extension element).

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

  - "0": usacExtElementUseDefaultLength (referring to audio pre-roll extension element).
- If no *AudioPreRoll()* is transmitted, the extension payload shall not be present (usacExtElementPresent = 0).
- The pre-roll frames with index "0" shall be independently decodable, i.e. *usacIndependencyFlag* shall be set to "1".
- To enable seamless configuration changes and bitrate adaptations the involved bitstreams shall have the element **receiverDelayCompensation** set to one.

Table 60 — Setup of mpegh3daExtElementConfig() for AudioPreRoll

Bitstream Field	Value
usacExtElementType	ID_EXT_ELE_AUDIOPREROLL
usacExtElementConfigLength	0
usacExtElementDefaultLengthPresent	0
usacExtElementPayloadFrag	0

Replace the content of 5.5.6.3, Decoding Process:

5.5.6.3.1 Introduction

This section describes the decoding process for both random access / immediate play-out and bitrate adoption scenarios.

5.5.6.3.2 Random access and immediate play-out

Random access and immediate play-out is possible at every frame that utilizes the *AudioPreRoll*() structure as specified in this subclause. The following pseudo-code describes the decoding process:

```
if(usacIndependencyFlag == 1){
  if(usacExtElementPresent == 1){

    /* In this case usacExtElementUseDefaultLength must be 0! */
    if(usacExtElementUseDefaultLength != 0) goto error;

    /* Not used */
    getmpegh3daExtElementPayloadLength();

    /* Check for presence of config and re-initialize if necessary */
    int configLen = getConfigLen();
    if(configLen > 0){
      config c = getConfig(configLen);
      ReConfigureDecoder(c);
      if (ID_EXT_ELE_SAOC_3D) { /*if SAOC 3D payload is present in the bitstream */
        config saoc3dConfig = getSaoc3dConfig(c);
        ReConfigureSaoc3dDecoder(saoc3dConfig);
      }
    }

    /* Get pre-roll AUs and decode, discard output samples */
    int numPreRollFrames = getNumPreRollFrames();
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
      for(auIdx = 0; auIdx < numPreRollFrames; auIdx++)
        int auLen = getAuLen();
        AU nextAU = getPreRollAU(auLen);
        if (ID_EXT_ELE_SAOC_3D) { /*if SAOC 3D payload is present in the bitstream */
          frame nextSaoc3dFrame = Saoc3DFrame(nextAU);
          [outSamplesFrame, saocInSamplesFrame] = DecodeAU(nextAU);
          saocOutSamplesFrame = Saoc3dDecode(saocInSamplesFrame, nextSaoc3dFrame);
          /* outSamplesFrame and saocOutSamplesFrame are discarded */
        } else {
          DecodeAU(nextAU);
        }
      }
    }
}
/* Internal decoder states are initialized at this point. Continue normal decoding */
```

5.5.6.3.3 Bitrate adaption

Bitrate adaption may be utilized by switching between different encoded representations of the same audio content. The *AudioPreRoll*() structure as specified in this subclause may be used for that purpose. The decoding process in case of bitrate adaption is specified by the following pseudo-code:

```
if(usacIndependencyFlag == 1){
  if(usacExtElementPresent == 1{

    /* In this case usacExtElementUseDefaultLength must be 0! */
    if(usacExtElementUseDefaultLength != 0) goto error;

    /* Not used */
    getmpegh3daExtElementPayloadLength();

    int configLen = getConfigLen();
    if(configLen > 0){
      config newConfig = getConfig(configLen);

      /* Configuration did not change, skip AudioPreRoll and continue decoding as normal
*/
      if(newConfig == currentConfig){
        SkipAudioPreRoll();
        goto finish;
      }

      /* Configuration changed, prepare for bitstream switching*/
      outSamplesFlush = FlushDecoder();
      ReConfigureDecoder(c);

      if (ID_EXT_ELE_SAOC_3D) { /*if SAOC 3D payload is present in the bitstream */
        int differentSaoc3dStream = 0;
        config newSaoc3dConfig = getSaoc3dConfig(newConfig);
        if (newSaoc3dConfig != currentSaoc3dConfig) {
          differentSaoc3dStream = 1;
          ReConfigureSaoc3dDecoder(newSaoc3dConfig);
        }
      }

      /* Get pre-roll AUs and decode, discard output samples */
      int numPreRollFrames = getNumPreRollFrames();
      for(auIdx = 0; auIdx < numPreRollFrames; auIdx++)
        int auLen = getAuLen();
        AU nextAU = getPreRollAU(auLen);
        /*if SAOC 3D payload is present in the bitstream */
        if (ID_EXT_ELE_SAOC_3D) && (differentSaoc3dStream == 1) {
          frame nextSaoc3dFrame = Saoc3DFrame();
          [outSamplesFrame, saocInSamplesFrame] = DecodeAU(nextAU);
          saocOutSamplesFrame = Saoc3dDecode(saocInSamplesFrame, nextSaoc3dFrame);
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
          /* outSamplesFrame and saocOutSamplesFrame are discarded */
        } else {
          DecodeAU(nextAU);
        }
      }

      /* Get "regular" AU and decode */
      AU au = mpegh3daFrame();
      /*if SAOC 3D payload is present in the bitstream */
      if (ID_EXT_ELE_SAOC_3D) && (differentSaoc3dStream == 1) {
        frame Saoc3dFrame = Saoc3DFrame();
        [outSamplesFrame, saocInSamplesFrame] = DecodeAU(au);
         saocOutSamplesFrame = Saoc3dDecode(saocInSamplesFrame, Saoc3dFrame);
      } else {
        outSamplesFrame = DecodeAU(au);
      }

      /* Apply crossfade only on the non-SAOC output samples*/
      for(i = 0; i < 128; i++){
        outSamples[i] = outSamplesFlush[i] * (1-i/127) +
                        outSamplesFrame[i] * (i/127)
      }
      for(i = 128; i < outputFrameLength; i++){
        outSamples[i] = outSamplesFrame[i];
      }
    }
  }
}
```

If a configuration change is detected by the decoder the following steps shall be applied:

— Flush the internal decoder states and buffers (`FlushDecoder()`), i.e. decode a hypothetical access unit composed of all zero samples. Store the resulting output samples corresponding to non-SAOC transport channels (`outSamplesFlush`) in a temporary buffer.

— Re-initialize the decoder with the new configuration (`ReConfigureDecoder()`).

— If an SAOC 3D configuration change happened, re-initialize the SAOC 3D decoder with the new configuration (`ReConfigureSaoc3dDecoder`).

— Decode all contained pre-roll AUs and discard the resulting output samples corresponding to non-SAOC transport channels.
The output samples corresponding to SAOC transport channels (`saocInSamplesFrame`), if present, are further decoded together with the SAOC 3D payload (`nextSaoc3dFrame`) by the SAOC 3D decoder. Discard the output samples of the SAOC 3D decoder.

— Decode the current AU (`mpegh3daFrame()`). Store the resulting output samples corresponding to non-SAOC transport channels (`outSamplesFrame`) in a temporary buffer.
Decode the output samples corresponding to SAOC transport channels (`saocInSamplesFrame`), if present, together with the SAOC 3D payload (`Saoc3dFrame`). Store the output of the SAOC 3D decoder in a temporary buffer (`saocOutSamplesFrame`).

— To avoid switching artifacts apply a linear cross-fade of length 128 on `outSamplesFlush` and `outSamplesFrame`. Play-out the result of the cross-fade (`outSamples`) and the SAOC 3D decoder output (`saocOutSamplesFrame`).
If the decoded signal is available in the QMF domain, no cross-fade is applied.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

with the following:

5.5.6.3.1 Introduction

This section describes the decoding process for both random access / immediate play-out and configuration changes including bitrate adoption scenarios.

5.5.6.3.2 Random access and immediate play-out

Random access and immediate play-out is possible at every frame that utilizes the *AudioPreRoll()* structure as specified in this subclause. The following pseudo-code describes the decoding process:

```
if(usacIndependencyFlag == 1){
  if(usacExtElementPresent == 1){

    /* In this case usacExtElementUseDefaultLength must be 0! */
    if(usacExtElementUseDefaultLength != 0) goto error;

    /* parse over length information and discard */
    getmpegh3daExtElementPayloadLength();

    /* Check for presence of config and re-initialize if necessary */
    int configLen = getConfigLen();
    if(configLen > 0){
      config c = getConfig(configLen);
      ReConfigureDecoder(c);
    }

    /* Get pre-roll AUs and decode, discard output samples */
    int numPreRollFrames = getNumPreRollFrames();
    for(auIdx = 0; auIdx < numPreRollFrames; auIdx++)
      int auLen = getAuLen();
      AU nextAU = getPreRollAU(auLen);
      DecodeAU(nextAU);
      /* outSamplesFrame are discarded */
    }
  }
}
/* Internal decoder states are initialized at this point. Continue normal decoding */
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

5.5.6.3.3 Configuration change and bitrate adaption

If **receiverDelayCompensation==1** the *AudioPreRoll()* structure as specified in this subclause can be used to enable seamless configuration changes and bitrate adaptions. The decoding process is specified below.

If a configuration change is detected by the decoder the following steps shall be applied. Note that a configuration (and thus a configuration change) may be signaled to the decoder either by a new mpegh3daConfig() element (e.g. in an MHAS packet of type PACTYP_MPEGH3DACFG) or within the *AudioPreRoll()* structure.

— Flush the internal decoder states and buffers for the core decoder and rendering path (`FlushDecoder()`) by decoding hypothetical access units composed of all zero samples. Store the resulting output samples (`outSamplesFlush`) in a temporary buffer. The number of flushed samples (`numSamplesFlushed`) shall equal the core decoder and rendering path delay.

— Re-initialize the decoder with the new configuration.

— Decode and render all contained pre-roll AUs and discard the resulting rendered samples at the mixer.

— Decode and render the current AU and following AUs and store the resulting output samples (`outSamplesFrame`). Discard the first `numSamplesFlushed` samples of `outSamplesFrame`. After discarding, the first remaining sample in `outSamplesFrame` is the first valid decoded and rendered sample.

— During startup of the reinitialized decoder feed samples from the temporary buffer (`outSamplesFlush`) into the post-processor. Note that due to the constant decoder delay enforced by **receiverDelayCompensation==1** the temporary buffer will cover exactly the startup phase of length `numSamplesFlushed` until valid samples from the reinitialized decoder will arrive at the mixer.

— After the startup of the reinitialized decoder feed samples from the reinitialized decoder (`outSamplesFrame`) into the post-processor.

— In case **applyCrossfade** is set to 1 and the mixer operates in the time domain, an additional buffer of 128 samples (`crossfadeFlush`) shall be flushed in the `FlushDecoder()` call. The `crossfadeFlush` buffer shall be used for a 128 sample linear crossfade between `crossfadeFlush` and the first 128 valid samples of the reinitialized decoder arriving at the mixer:

```
/* Apply crossfade */
if(applyCrossfade) {
  for(i = 0; i < 128; i++){
    outSamples[i] = crossfadeFlush [i] * (1-i/127) +
                    outSamplesFrame[i] * (i/127)
  }
}
```

Add in 5.7, Stream Access Point requirements and inter-frame dependency:

— if MPS212 is used, bsIndependencyFlag shall be 1.

— if SAOC 3D is used, bsIndependencyFlag shall be 1.

— if HOA is used, hoaIndependencyFlag shall be 1.

— If object_metadata (OAM) is used, independently decodable oam_metadata() has to be available (intracoded_object_metadata_efficient() in case of efficient object metadata coding or intracoded_object_metadata_low_delay() in case of object metadata coding with low delay).

10.5 Multi-stream Handling

Add new clause XX:

XX Multi-stream Handling

In multi-stream scenarios the main and side-streams are determined by the mae_isMainStream flag as defined in clause 15.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

XX.1 Restrictions on extension payloads

In multi-stream scenarios extension payloads may be present in both main and side-stream(s). The restrictions as defined in Table AMDXX.1 and AMDXX.2 shall apply.

Table AMDXX.1 — Allowed configuration extensions in main- and side-stream(s)

usacConfigExtType	Main-stream	Side-stream(s)
ID_CONFIG_EXT_DOWNMIX	One element for all present signal groups of type SignalGroupTypeChannels.	One element for all present signal groups of type SignalGroupTypeChannels.
ID_CONFIG_EXT_LOUDNESS_INFO	One element that comprises group specific metadata (loudnessInfoType=1) for present signal groups and full scene metadata including side-stream(s) (loudnessInfoType=0/2/3).	One element that comprises group specific metadata (loudnessInfoType=1) for present signal groups only.
ID_CONFIG_EXT_AUDIOSCENE_INFO	One element with mae_isMainStream=1	One element with mae_isMainStream=0
ID_CONFIG_EXT_HOA_MATRIX	One element for each present signal group of type SignalGroupTypeHOA	One element for each present signal group of type SignalGroupTypeHOA

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table AMDXX.2 — Allowed bit stream extensions in main- and side-stream(s)

usacExtElementType	Main-stream	Side-stream(s)
ID_EXT_ELE_MPEGS	One element for each present signal group of type SignalGroupTypeChannels.	One element for each present signal group of type SignalGroupTypeChannels.
ID_EXT_ELE_AUDIOPREROLL	One element for all present signal groups.	One element for all present signal groups.
ID_EXT_ELE_UNI_DRC	One element that comprises group specific metadata (DRC-1) for present signal groups and full scene metadata including side-stream(s) (DRC-2/3).	One element that comprises group specific metadata (DRC-1) for present signal groups only.
ID_EXT_ELE_OBJ_METADATA	One element for each present signal group of type SignalGroupTypeObject.	One element for each present signal group of type SignalGroupTypeObject.
ID_EXT_ELE_SAOC_3D	One element for each present signal group of type SignalGroupTypeSAOC.	One element for each present signal group of type SignalGroupTypeSAOC.
ID_EXT_ELE_HOA	One element for each present signal group of type SignalGroupTypeHOA.	One element for each present signal group of type SignalGroupTypeHOA.
ID_EXT_ELE_FMT_CNVRTR	One element for all present signal groups of type SignalGroupTypeChannels. and immersiveDownmixFlag=1.	One element for all present signal groups of type SignalGroupTypeChannels. and immersiveDownmixFlag=1.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

11 SAOC signaling update

Add in "Table 4 — Syntax of Signals3d()":

```
        if ( SignalGroupType[grp] == SignalGroupTypeSAOC ) {
            numSAOCTransportChannels += bsNumberOfSignals[grp] + 1;
            saocDmxLayoutPresent;                              1        bslbf
            if ( saocDmxLayoutPresent == 1 ) {
                saocDmxChannelLayout = SpeakerConfig3d();
            }
        }
```

Amend in "15.3 Semantics":

mae_metaDataElementID	This field uniquely defines an ID for a metadata element (i.e. channels, dynamic objects, SAOC channels, SAOC objects, SignalGroupTypeSAOC signals, HOA). This field can take values between 0 and 127, resulting in a maximum of 128 metadata elements. One group shall only contain elements of the same signal type (static objects (channel-based signals without accompanying OAM data), objects (audio tracks with accompanying OAM data), SAOC channels, SAOC object, SignalGroupTypeSAOC signals, HOA).

Add in "15.4 Definition of mae_metaDataElementIDs":

```
else if ( SignalGroupType[grp] == SignalGroupTypeSAOC ) {
   for ( id = 0; id < numSpeakers + bsNumSaocObjects; id++ )  {
      mae_metaDataElementID++;
   }
   if(saocDmxLayoutPresent == 1) {
      mae_metaDataElementID++;
   }
 }
```

Add in "15.4 Definition of mae_metaDataElementIDs":

If saocDmxLayoutPresent == 1 the MAE information shall contain one additional group with exactly one data element. The additional group shall be located after the last group containing elements of signal type SAOC channels or SAOC objects. The data element in the additional group shall be associated with the complete group of SAOC Transport Channels. Additionally, if saocDmxLayoutPresent == 1, the value of the variable baseChannelCount, defined in 6.3.3, shall be increased by 1, such that an individual DRC gain sequence can be assigned to the complete group of SAOC Transport Channels.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Add in "5.3.2 General Configuration Data Elements":

saocDmxLayoutPresent	This flag indicates if the SAOC audio transport channels contain a meaningful downmix of the input channels and objects. If saocDmxLayoutPresent == 1 the saocDmxChannelLayout is associated with the SAOC audio transport channels. Additionally if the number of output channels is smaller than the number of SAOC audio transport channels, the SAOC audio transport channels should be further processed as signals of type

	SignalGroupTypeChannels with the audio channel layout saocDmxChannelLayout. and the SAOC payload should be discarded.
saocDmxChannelLayout	This structure describes the loudspeaker layout of the SAOC audio transport channels, if saocDmxLayoutPresent == 1. The number of speakers signaled in saocDmxChannelLayout shall be equal to (bsNumberOfSignals[grp] + 1), if SignalGroupType[grp] == SignalGroupTypeSAOC.

Add in "5.3.2 General Configuration Data Elements":

numSAOCTransportChannels	Explicit number of SAOC audio transport channels that are conveyed in the present stream by means of SCEs, CPEs, QCEs, and LFEs and that are further processed in the SAOC 3D Decoder. In the element loop of mpegh3daDecoderConfig() these elements shall be located after all audio channel and audio object related elements.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

12 Tool for Advanced Loudness Control

Add new Table after Table 151:

Table AMD11.1 — Syntax of mae_LoudnessCompensationData()

Syntax	No. of bits	Mnemonic
mae_LoudnessCompensationData(numGroups,numGroupPresets)		
{		
mae_loudnessCompGroupLoudnessPresent;	1	bslbf
if (mae_loudnessCompGroupLoudnessPresent == 1) {		
for(grp=0; grp<numGroups; grp++) {		
mae_bsLoudnessCompGroupLoudness[grp];	8	uimsbf
}		
} else {		
/* not present or provided by mpegh3daLoudnessInfoSet() */		
}		
mae_loudnessCompDefaultParamsPresent;	1	bslbf
if (mae_loudnessCompDefaultParamsPresent == 1) {		
for(grp=0; grp<numGroups; grp++) {		
groupID = mae_groupID[grp];		
mae_loudnessCompDefaultIncludeGroup[grp];	1	bslbf
}		
mae_loudnessCompDefaultMinMaxGainPresent;	1	bslbf
if (mae_loudnessCompDefaultMinMaxGainPresent == 1) {		
mae_bsLoudnessCompDefaultMinGain;	4	uimsbf
mae_bsLoudnessCompDefaultMaxGain;	4	uimsbf
}		
}		
for (gp=0; gp<numGroupPresets; gp++) {		
groupPresetID = mae_groupPresetID[gp];		
mae_loudnessCompPresetParamsPresent[gp];	1	bslbf
if (mae_loudnessCompPresetParamsPresent[gp] == 1) {		
for (grp=0; grp<numGroups; grp++) {		
groupID = mae_groupID[grp];		
mae_loudnessCompPresetIncludeGroup[gp][grp];	1	bslbf
}		
mae_loudnessCompPresetMinMaxGainPresent[gp];	1	bslbf
if (mae_loudnessCompPresetMinMaxGainPresent[gp]) {		
mae_bsLoudnessCompPresetMinGain[gp];	4	uimsbf
mae_bsLoudnessCompPresetMaxGain[gp];	4	uimsbf
}		
}		
}		
}		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Add new semantics at end of Section 15.3:

mae_loudnessCompGroupLoudnessPresent A field that indicates whether group loudness values for loudness compensation follow in the bitstream.

mae_bsLoudnessCompGroupLoudness A field that signals a loudness value for the current metadata element group (groupID).

loudnessCompGroupLoudness in dB = 0.25 · **mae_bsLoudnessCompGroupLoudness** - 57.75

mae_loudnessCompDefaultParamsPresent A field that indicates whether loudness compensation parameters for the default scene follow in the bitstream. If not present, all metadata element groups shall be incorporated in the computation of the loudness compensation gain.

mae_loudnessCompDefaultIncludeGroup A field that signals whether the current metadata element group (groupID) shall be incorporated in the computation of the loudness compensation gain of the default scene.

mae_loudnessCompDefaultMinMaxGainPresent A field that indicates whether min/max values for loudness compensation gain of the default scene follow in the bitstream.

mae_bsLoudnessCompDefaultMinGain A field that signals a minimum value for the loudness compensation gain of the default scene. It can take values between 0 and minus 42 dB in 3 dB steps. A value of minus infinity can be signaled by the largest number of mae_bsLoudnessCompDefaultMinGain (15). If not present the default value is minus infinity.

loudnessCompDefaultMinGain in dB = -3 · **mae_bsLoudnessCompDefaultMinGain**

mae_bsLoudnessCompDefaultMaxGain A field that signals a maximum value for the loudness compensation gain of the default scene. It can take values between 0 and 45 dB in 3 dB steps. If not present the default value is plus 21 dB.

loudnessCompDefaultMaxGain in dB = 3 · **mae_bsLoudnessCompDefaultMaxGain**

mae_loudnessCompPresetParamsPresent A field that indicates whether loudness compensation parameters for the current preset (groupPresetID) follow in the bitstream. If not present, all metadata element groups shall be incorporated in the computation of the loudness compensation gain.

mae_loudnessCompPresetIncludeGroup A field that signals whether the current metadata element group (groupID) shall be incorporated in the computation of the loudness compensation gain of the current preset (groupPresetID).

mae_loudnessCompPresetMinMaxGainPresent A field that indicates whether min/max values for loudness compensation gain of the current preset (groupPresetID) follow in the bitstream.

mae_loudnessCompPresetMinGain A field that signals a minimum value for the loudness compensation gain of the current preset (groupPresetID). It can take values between 0 and minus 42 dB in 3 dB steps. A value of minus infinity can be signaled by the largest number of mae_bsLoudnessCompPresetMinGain (15). If not present the default value is minus infinity.

loudnessCompPresetMinGain in dB = -3 · **mae_bsLoudnessCompPresetMinGain**

mae_loudnessCompPresetMaxGain A field that signals a maximum value for the loudness compensation gain of the current preset (groupPresetID). It can take values between 0 and 45 dB in 3 dB steps. If not present the default value is plus 21 dB.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

loudnessCompPresetMaxGain in dB = 3 · **mae_bsLoudnessCompPresetMaxGain**

Add new sub-clause after 15.4:

15.5 Loudness Compensation after Gain Interactivity

An important feature of MPEG-H 3DA is the support of user interaction at the decoder: The user can, e.g. adjust the volume of metadata element groups or even switch them on and off. Changing the level of groups also implies that the overall loudness of the rendered audio scene is changed compared to the unmodified case.

The loudness normalization feature as specified in sub-clause 6.4.7, applies to unmodified reference scenes dependent on the selected preset, target layout or DRC configuration. The loudness compensation tool specified in this sub-clause operates in addition to loudness normalization according to 6.4.7 and compensates for any loudness change relative to the unmodified reference scene due to user interaction with the reference scene.

If the structure mae_LoudnessCompensationData() is present in the metadata bitstream (mae_dataType == ID_MAE_LOUDNESS_COMPENSATION), the loudness compensation tool shall be enabled. If not present, the loudness compensation tool shall be disabled by default.

If the loudness compensation tool is enabled, a loudness compensation gain shall be computed after any gain interaction or preset selection according to Table AMD11.2. The computed compensation gain shall be applied to each audio element. The input parameters for the computation of the loudness compensation gain are extracted as specified in Table AMD11.3.

Group loudness values can be either transmitted within mae_LoudnessCompensationData() by using mae_loudnessCompGroupLoudnessPresent==1 (see 15.2) or within mpegh3aLoudnessInfoSet() by using loudnessInfoType==1 (see 6.3). If group loudness values for one or more metadata element groups are missing within mpegh3aLoudnessInfoSet(), the loudness compensation gain shall be computed as listed in Table AMD11.2 for groupLoudnessValueMissingFlag==1.

Table AMD11.2 — Pseudo Code for Computation of Loudness Compensation Gain

```
computeLoudnessCompensationGain( numGroups,
                                 includeGroup[],
                                 groupLoudnessValueMissingFlag,
                                 groupLoudness[],
                                 groupGainDefaultDb[],
                                 groupGainInteractivityDb[],
                                 groupStateDefault[],
                                 groupStateInteractivity[],
                                 minGainDb,
                                 maxGainDb)
{
    /* init */
    loudnessReference = 0;
    loudnessAfterInteract = 0;

    /* compute components of loudness compensation gain */
    for (n=0; n<numGroups; n++) {
      if (groupLoudnessValueMissingFlag == 0) {
          tmp1 = pow(10, (groupGainDefaultDb[n] + groupLoudness[n]) / 10.0);
          tmp2 = pow(10, (groupGainInteractivityDb[n] + groupLoudness[n]) / 10.0);
      } else { /* group loudness value missing for one or more groups */
          tmp1 = pow(10, groupGainDefaultDb[n] / 10.0);
          tmp2 = pow(10, groupGainInteractivityDb[n] / 10.0);
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
        }
        loudnessReference += includeGroup[n]*groupStateDefault[n] * tmp1;
        loudnessAfterInteract += includeGroup[n]*groupStateInteractivity[n] * tmp2;
    }

    /* loudness compensation gain in dB */
    loudnessCompensationGainDb = 10 * log10(
                                    loudnessReference / loudnessAfterInteract);

    /* clip loudness compensation gain to min/max gain */
    if (loudnessCompensationGainDb < minGainDb) {
        loudnessCompensationGainDb = minGainDb;
    }
    if (loudnessCompensationGainDb > maxGainDb) {
        loudnessCompensationGainDb = maxGainDb;
    }

    return loudnessCompensationGainDb;
}
```

Table AMD11.3 — Input Parameters for Computation of Loudness Compensation Gain

Input Parameters	**Default Scene (no preset selected)**	**Preset Scene (groupPresetID selected, gp is the index of the selected preset)**
numGroups	mae_numGroups	mae_numGroups
includeGroup[]	mae_loudnessCompDefault-IncludeGroup[]	mae_loudnessCompPreset-IncludeGroup[gp][]
groupLoudness[]	loudnessCompGroupLoudness[] if present; alternatively, extracted from mpegh3daLoudnessInfoSet().	loudnessCompGroupLoudness[] if present; alternatively, extracted from mpegh3daLoudnessInfoSet().
groupGainDefaultDb[]	0 dB for all groups.	If present, groupPresetGain[gp][] and otherwise 0 dB for all groups.
groupGainInteractivityDb[]	Current interactivity gain in dB.	Current interactivity gain in dB.
groupStateDefault[]	"1" for groups that are switched on in the default scene, "0" for groups that are switched off.	"1" for groups that are switched on in the preset definition, "0" for groups that are switched off. For groups that are not explicitly referenced in the preset definition, the respective state of the default scene applies.
groupStateInteractivity[]	"1" for groups that are switched on after interactivity, "0" for groups that are switched off.	"1" for groups that are switched on after interactivity, "0" for groups that are switched off.
minGain	loudnessCompDefaultMinGain	loudnessCompPresetMinGain[gp]
maxGain	loudnessCompDefaultMaxGain	loudnessCompPresetMaxGain[gp]

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

13 Frequency-Domain Prediction and Time-Domain Post-Filtering

In Table 32 – Syntax of mpegh3daCoreCoderData(), replace:

```
if (nrChannels == 2) {
    StereoCoreToolInfo(core_mode);
}
```

with:

```
if (nrChannels == 2) {
    StereoCoreToolInfo(core_mode);
} else {
    common_ltpf = 0;
}
```

In Table 33 – Syntax of StereoCoreToolInfo(),replace:

```
    }
    if (tns_active) {
```

with:

```
    }
    if (common_ltpf) {                          1        uimsbf
        if (ltpf_data_present) {                1        uimsbf
            ltpf_pitch_lag_index;               9        uimsbf
            ltpf_gain_index;                    2        uimsbf
        }
    }
    if (tns_active) {
```

In Table 33 – Syntax of StereoCoreToolInfo(),replace:

```
} else {
     common_window = 0;
     common_tw = 0;
 }
```

with:

```
} else {
     common_window = 0;
     common_ltpf = 0;
     common_tw = 0;
 }
```

In Table 34 – Syntax of fd_channel_stream(), replace:

```
}
scale_factor_data();
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

with:

```
    }
    if (!common_ltpf) {
        if (ltpf_data_present) {                        1       uimsbf
            ltpf_pitch_lag_index;                       9       uimsbf
            ltpf_gain_index;                            2       uimsbf
        }
    }
    if ((!indepFlag) &&
      (window_sequence != EIGHT_SHORT_SEQUENCE)) {
        if (fdp_data_present) {                         1       uimsbf
            fdp_spacing_index;                          8       uimsbf
        }
    } else {
        fdp_data_present = 0;
    }
    scale_factor_data();
```

At the end of 5.5 add the following subclauses:

5.5.X Frequency Domain Prediction

5.5.X.1 Tool Description

The frequency domain prediction (FDP) tool can be utilized for subjective quality improvement of low-frequency harmonic signal components. It is largely designed in a fixed-point way in order to ensure consistent operation across different platforms. FDP is applied individually for each channel of the given element in the TNS filtered (and in the case of channel pair elements, the joint-stereo coded) MDCT spectral domain, as obtained after the entropy decoding and noise filling steps, and is supported in both the MDCT based TCX and FD coding modes.

5.5.X.2 Operational Constraints

The FDP tool is only available in dependently coded channels/frames (i. e. indepFlag == 0) which are transform coded using the maximum MDCT length (i. e. largest available mod[k] in case of TCX and window_sequence != EIGHT_SHORT_SEQUENCE in case of FD coding) and for which no transition from TCX to FD coding, or vice versa, occurred between the last and current frame. If these requirements are not satisfied, the FDP indicator, **fdp_data_present**, should equal zero, and all FDP helper states (see below) must be set to zero.

5.5.X.3 Terms and Definitions

fdp_data_present	binary flag indicating whether the FDP tool is active (1) or disabled (0) in the channel.
fdp_spacing_index	eight-bit integer holding the harmonic spacing index used during the FDP processing.
ccfl	coreCoderFrameLength, the transform length, see ISO/IEC 23003-3:2012, subclause 6.1.
g	re-scaling gain, based on global_gain value, see ISO/IEC 23003-3:2012, subclause 7.15.
lg	number of quantized MDCT bins, see ISO/IEC 23003-3:2012, subcl. 6.2.9.2 and 7.15.
noiseFillingStartOffset	noise filling start line, defined depending on ccfl in ISO/IEC 23003-3:2012, Table 109.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

samplingFrequency	core-coder sample rate defined by usacSamplingFrequency(Index) located in Table 2.
harmonicSpacing	helper element, unsigned integer holding a harmonic spacing value for FDP decoding.
predictionBandwidth	helper element, unsigned integer holding the maximum line count for FDP decoding.
quantSpecPrev[ ][ ]	helper array, internal signed-integer MDCT line memory for the inter-frame prediction.
fdp_exp[ ]	constant array holding the integer line-expansion data $NINT\left(64 \cdot i^{4/3}\right)$, with 0 ≤ i ≤ 181.
fdp_scf[ ]	constant array holding the integer scale-factor power data $NINT\left(2^{(s+1)/4}\right)$, with 0 ≤ s ≤ 63.
fdp_sin[ ]	constant array holding the floating-point sine values of $\sin\left(\pi \cdot {}^{i}/_{256}\right)$, with 0 ≤ i ≤ 128.
fdp_int[ ]	output array holding the signed-integer predictor values derived during FDP decoding.

5.5.X.4 Decoding Process

The FDP decoding procedure is performed in four consecutive operations, which are described in the following.

Step 1: Derivation of Harmonic Spacing Value

If **fdp_data_present** == 0, this step is skipped. Otherwise, harmonicSpacing is derived from **fdp_spacing_index**:

$$\text{harmonicSpacing} = (894 \cdot 512 + \text{fdp\_spacing\_value}) / (2 \cdot \text{fdp\_spacing\_value})$$

with fdp_spacing_value = 894 / 3 – **fdp_spacing_index**. The division in the above equation is an integer division.

Step 2: Determination of Prediction Bandwidth

If ccfl ≠ 768 and samplingFrequency ≥ 44100 Hz (i. e. usacSamplingFrequencyIndex < 5), predictionBandwidth equals 132. Otherwise, predictionBandwidth equals the long-window-sequence value of noiseFillingStartOffset. Also, predictionBandwidth is limited to lg, the number of quantized MDCT lines given by the arithmetic decoder:

$$\text{predictionBandwidth} = \min(\text{lg}, \text{predictionBandwidth}).$$

Step 3: Execution of MDCT-Domain Prediction

The FDP decoding process, which returns the predictor values fdp_int[i], 0 ≤ i < predictionBandwidth, depends on the mode of the current frame and channel. If **fdp_data_present** == 0, all fdp_int[i] = 0. Otherwise, in case of FD coding, FDP decoding is applied to the expanded, scaled MDCT values outputSpecCurr[i] after noise filling:

```
harmIndex = -128, compIndex = 256; /* harmonic and compare indices */
s1 = 0; s2 = 0; /* LPC coefficients, adapt both for each harmonic */

if (fdp_data_present) { /* FDP active and allowed, obtain estimate */
  for (i = 0; i < predictionBandwidth; i++) {
    if (abs(i * 256 - harmIndex) >= 384) {  /* bin is not harmonic */
      fdp_int[i] = 0;
    } else {  /* bin is part of the currently active harmonic line */
      reg32 = s1 * quantSpecPrev[0][i] + s2 * quantSpecPrev[1][i];
      fdp_int[i] = sign(reg32) * (((unsigned int)abs(reg32) + 16384) >> 15);
      outputSpecCurr[i] += i_gain * fdp_int[i]; /* actual decoding */
    }
    if (i * 256 == compIndex) {   /* update indices and LPC coeffs */
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
        harmIndex += harmonicSpacing;
        compIndex = harmIndex & 255;
        if (compIndex > 128) {
          compIndex = 256 - compIndex; /* exploit trigonom. symmetry */
        }
        s1 = NINT( 768*min(82-18*fdp_sin[compIndex]^2, 77-6*fdp_sin[compIndex]^2)*
             fdp_sin[compIndex]);
        s2 = NINT(-4.5*min(82-18*fdp_sin[compIndex]^2, 77-6*fdp_sin[compIndex]^2)^2);
        compIndex = harmIndex >> 8;  /* integer unscaled harm. index */
        if ((compIndex & 1) == 0) {
          s1 *= -1; /* negate first LPC coeff for even harm. indices */
        }
        compIndex = 256 + ((harmIndex + 128) >> 8) * 256;  /* update */
      }
    }
  }
```

Note that, in case of MDCT based TCX coding in the given frame and channel (core_mode[ch] == 1), inverse gain `i_gain` = 64 / g, i. e. an amplified version of the re-scaling gain, and in case of FD coding, `i_gain` = 512.

Step 4: Update of Spectral Prediction Memory

For each line at index 0 ≤ i < predictionBandwidth an integer representation x_int[i] of the expanded, scaled line value is computed. In case of FD coding, this depends on the quantized value x_ac_quant[i] and its associated scale factor scf[sfb] for band sfb (see ISO 23003-3:2012, subcl. 7.1 and 7.2). If scf[sfb] < 21, x_int[i] = fdp_int[i]. Otherwise,

$$x[i] = fdp\_exp[\min(abs(x\_ac\_quant[i]), 181)] \cdot fdp\_scf[\min(scf[sfb] - 21, 63)],$$

$$x\_int[i] = sign(x\_ac\_quant[i]) \cdot ((x[i] + 512) >> 10) + fdp\_int[i] \quad \text{(">>" is a binary shift).}$$

In case of MDCT based TCX coding, x_int[i] is derived from the x_tcx_invquant[i] coefficients and the gain g:

$$x\_int[i] = x\_tcx\_invquant[i] \cdot NINT(g / 64) + fdp\_int[i].$$

For all 0 ≤ i < predictionBandwidth, the update is finalized using quantSpecPrev[1][i] = quantSpecPrev[0][i] and, afterwards,

$$quantSpecPrev[0][i] = \min(\max(x\_int[i], -31775), 31775).$$

Note: all x_int[ ] associated with uncoded scale factor bands (i. e. bands whose sfb ≥ max_sfb) must equal zero.

5.5.Y Long-Term Postfilter

5.5.Y.1 Tool Description

The long-term postfilter (LTPF) tool can be utilized for subjective quality improvement of low-frequency harmonic signal components. LTPF is applied individually for each channel of the given element in the time domain as obtained after FD/LPD decoding. More specifically, it is applied on the time-domain signal obtained after the overlap-and-add operation in case of the FD mode (see 7.9.3.3 in ISO/IEC 23003-3) and after the bass postfilter in case of the LPD mode (see. 7.17 in ISO/IEC 23003-3).

5.5.Y.2 Operational Constraints

The LTPF tool is supported in both the FD mode and in the longest MDCT based TCX mode, but not in the shorter MDCT based TCX modes nor in the ACELP mode. However, to avoid any discontinuities that could be introduced when switching from a frame where the tool is supported to a frame where the tool is not supported,

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

the LTPF decoding process is still applied in the modes where the tool is not supported but with **ltpf_data_present** equal zero.

5.5.Y.3 Terms and Definitions

ltpf_data_present	binary flag indicating whether the LTPF tool is active (1) or disabled (0) in the channel.
ltpf_pitch_lag_index	nine-bit integer holding the pitch lag index used during the LTPF decoding process.
ltpf_gain_index	two-bit integer holding the gain index used during the LTPF decoding process.
pit_int	integer part of the pitch lag used during the LTPF decoding process.
pit_fr	fractional part of the pitch lag used during the LTPF decoding process.
gain	gain used during the LTPF decoding process.
Fs	the sampling frequency at which the core coder operates.
ccfl	core coder frame length in samples

5.5.Y.4 Decoding Process

5.5.Y.4.1 General

The LTPF tool processes the output signal of the FD/LPD core decoder with an IIR filter, whose coefficients are derived from three parameters that are decoded from the bit stream. These parameters are estimated at the encoder side on a frame of length ccfl whose middle point coincides with the middle point of the MDCT window. The frame of output signal coming from the FD/LPD core decoder is however delayed by ccfl/2. At the decoder side, the LTPF tool then filters the first half of the current frame using the parameters decoded in the previous frame and filters the second half of the current frame using the parameters decoded in the current frame. To avoid any discontinuities that could be introduced when the filter parameters change between the previous and the current frame, the beginning portion of the second half of the current frame is processed with a transition filter.

The decoding of the filter parameters is described in **5.5.Y.4.1**. The IIR filtering of the core decoder output signal is described in **5.5.Y.4.2**. The transition filter used to remove possible discontinuities is described in **5.5.Y.4.3**.

5.5.Y.4.2 Decoding of the filter parameters

5.5.Y.4.2.1 General

There are three parameters per frame: the integer part of the pitch lag, the fractional part of the pitch lag, and the gain. If ltpf_data_present equals zero, then the three parameters are set to zero. Otherwise, the parameters are decoded as described in the following subclauses.

5.5.Y.4.2.2 Decoding of the integer and fractional parts of the pitch lag

A fractional pitch delay is used with resolutions 1⁄2 in the range [pit_min, pit_fr2-1⁄2], integers only in the range [pit_fr2, pit_fr1-1], and integers with increments by 2 in the range [pit_fr1, pit_max]. pit_min, pit_fr2, pit_fr1 and pit_max are the boundaries of the segments of the quantizers which depend on Fs and they are determined as follows:

```
pit_min = round( 34 * ( Fs / 2 ) / 12800 ) * 2;
pit_fr2 = 324 - pit_min;
pit_fr1 = 320;
pit_max = 54 + 6 * pit_min;
```

The integer and fractional parts of the pitch lag are then decoded as follows:

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
if ( ltpf_pitch_lag_index < (pit_fr2-pit_min)*2 )
{
  pit_int = pit_min + (ltpf_pitch_lag_index/2);
  pit_fr = ltpf_pitch_lag_index – (pit_int - pitmin)*2;
}
else if ( ltpf_pitch_lag_index < (pit_fr2-pit_min)*2 + (pit_fr1-pit_fr2) )
{
  pit_int = pit_fr2 + ltpf_pitch_lag_index - (pit_fr2-pit_min)*2;
  pit_fr = 0;
}
else
{
  pit_int = (ltpf_pitch_lag_index-(pit_fr2-pit_min)*2-(pit_fr1-pit_fr2))*2 + pit_fr1;
  pit_fr = 0;
}
```

5.5.Y.4.2.3 Decoding of the gain

The gain is decoded as follows:

```
gain = (ltpf_gain_index + 1) * 0.0625;
```

5.5.Y.4.3 IIR filtering

The LTPF processes the core decoder output with an IIR filter whose coefficients are derived from the integer and fractional parts of the pitch lag and from the gain. The IIR filter is implemented with the function given below, assuming filtering a portion of signal, where *x points to the first sample of the portion of input signal, *y points to the first sample of the portion of output signal, and N is the length of the portion of signal.

```
function ltpf_filter( *x, *y, N, gain, ltpf_gain_index, pit_int, pit_fr )

if ( gain == 0 )
{
  for ( n = 0; n < N; n++ )
  {
    y[n] = x[n];
  }
}
else
{
  for ( n = 0; n < N; n++ )
  {
    s1 = 0;
    for ( k = 0; k < 8; k++ )
    {
      s1 += y[n-pit_int+k-4] * ltpf_filter_coef1[pit_fr][k];
    }
    s2 = 0;
    for ( k = 0; k < 7; k++ )
    {
      s2 += x[n-k] * ltpf_filter_coef2[ltpf_gain_index][k];
    }
    y[n] = x[n] + gain * s1 - 0.95 * gain * s2;
  }
}
```

The two tables ltpf_filter_coef1 and ltpf_filter_coef2 are given below:

```
ltpf_filter_coef1[2][8] =
{
  {0.0000000,0.0304386,0.1162701,0.2195613,0.2674597,0.2195613,0.1162701,0.0304386},
  {0.0076226,0.0676508,0.1700032,0.2547232,0.2547232,0.1700032,0.0676508,0.0076226}
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
ltpf_filter_coef2[4][7] =
{
  {0.27150189,0.44286013,0.23027992,0.05759155,-0.00172290,-0.00045168,-0.00005891},
  {0.27581838,0.44682277,0.22783915,0.05410054,-0.00353758,-0.00092331,-0.00011995},
  {0.28044685,0.45103979,0.22519192,0.05037740,-0.00545541,-0.00141719,-0.00018336},
  {0.28543320,0.45554676,0.22230634,0.04638935,-0.00749011,-0.00193612,-0.00024943}
}
```

5.5.Y.4.4 Transition filtering

The first half of the current frame is always filtered with the function `ltpf_filter` using the parameters of the previous frame and N=ccfl/2.

If the parameters of the current frame are the same as the ones from the previous frame, the second half of the current frame is also filtered with the function `ltpf_filter` using the same parameters and N=ccfl/2. However, if the parameters change between the previous and the current frame, a discontinuity can be introduced. In that case, the beginning portion (ccfl/8 samples) of the second half of the current frame is processed with a transition filter described in the following. The remaining 3*ccfl/8 samples are then processed with the function `ltpf_filter` using the parameters from the current frame and N=3*ccfl/8.

Three cases are considered:

a) The gain of the previous frame is equal to zero and the gain of the current frame is not equal to zero

A simple fade-in mechanism is used where the function `ltpf_filter` is applied using the filter parameters of the current frame, N=ccfl/8, and changing the line of code

```
    y[n] = x[n] + gain * s1 - 0.95 * gain * s2;
```

by the following lines of code

```
    y[n] = x[n] + alpha * ( gain * s1 - 0.95 * gain * s2 );
    alpha += 1/N;
```

and by setting alpha to zero at the beginning of the function.

b) The gain of the previous frame is not equal to zero and the gain of the current frame is equal to zero

A simple fade-out mechanism is used where the function `ltpf_filter` is applied using the filter parameters of the previous frame, N=ccfl/8, and changing the line of code

```
    y[n] = x[n] + gain * s1 - 0.95 * gain * s2;
```

by the following lines of code

```
    y[n] = x[n] + alpha * ( gain * s1 - 0.95 * gain * s2 );
    alpha -= 1/N;
```

and by setting alpha to one at the beginning of the function.

c) The gain of the previous frame is not equal to zero and the gain of the current frame is not equal to zero

The discontinuity is removed using the zero-impulse-response (ZIR) of a LPC synthesis filter estimated on the previous frame, and with memories computed using the filter parameters of the current frame.

The coefficients of the LPC synthesis filter are estimated using the classic autocorrelation and levinson-durbin approach and is implemented using the function given below, where x[] is the portion of LTPF output signal corresponding to the last ccfl/4 samples, a[] are the LPC coefficients, N=ccfl/4 is the length of the portion of signal, and M=24 is the order of the LPC synthesis filter.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
function ltpf_get_lpc( x[], a[], N, M )

for ( m = 0; m <= M; m++ )
{
  s = 0.0;
  for ( n = 0; n < N-m; n++ )
  {
    s += x[n] * x[n+m];
  }
  r[m] = s;
}
if (r[0] < 100.0)
{
  r[0] = 100.0;
}
r[0] *= 1.0001;
a[0] = 1.0;
rc[0] = -r[1] / r[0];
a[1] = rc[0];
sigma2 = r[0] + r[1] * rc[0];
for ( m = 2; m <= M; m++ )
{
  sum = 0.0f;
  for ( i = 0; i < m; i++ )
  {
    sum += r[m-i] * a[i];
  }
  rc[m-1] = -sum / sigma2;
  sigma2 = sigma2 * ( 1.0 - rc[m-1] * rc[m-1] );
  if ( sigma2 <= 1.0E-09 )
  {
    sigma2 = 1.0E-09;
    for ( i = m; i <= M; i++ )
    {
      rc[i-1] = 0.0;
      a[i] = 0.0;
    }
    break;
  }
  for ( i = 1; I <= (m/2); i++ )
  {
    value = a[i] + rc[m-1] * a[m-i];
    a[m-i] += rc[m-1] * a[i];
    a[i] = value;
  }
  a[m] = rc[m-1];
}
```

Then, the ZIR of the LPC synthesis filter is computed using the following function, where *x points to the first sample of the beginning portion of the second half of the current frame of input signal, *y points to the first sample of the beginning portion of the second half of the current frame of output signal, and Lz=ccfl/8 is the length of the ZIR.

```
function ltpf_get_zir( *x, *y, a[], zir[], M, Lz )

for ( m = 0; m < M; n++ )
{
  s1 = 0;
  for ( k = 0; k < 8; k++ )
  {
    s1 += y[m-M-pit_int+k-4] * ltpf_filter_coef1[pit_fr][k];
  }
  s2 = 0;
  for ( k = 0; k < 7; k++ )
  {
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
    s2 += x[m-M-k] * ltpf_filter_coef2[ltpf_gain_index][k];
  }
  buf[n] = ( x[m-M] - 0.95 * gain * s2 ) – ( y[m-M] - gain * s1 );
}
for ( i = 0; i < Lz; i++ )
{
  for ( j = 1; j <= M; j++ )
  {
    buf[M+i] -= a[M] * buf[M+i-j];
  }
}
for ( i = 0; i < Lz/2; i++ )
{
  zir[i] = buf[M+i];
}
alpha = 1;
for ( i = Lz/2; i < Lz; i++ )
{
  zir[i] = buf[M+i]*alpha;
  alpha -= 2/Lz;
}
```

And finally, the function `ltpf_filter` is applied using the filter parameters of the current frame, N=ccfl/8, and changing the line of code

```
    y[n] = x[n] + gain * s1 - 0.95 * gain * s2;
```

by the following line of code

```
    y[n] = x[n] + gain * s1 - 0.95 * gain * s2 – zir[n];
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

14 Sample Rate Converter

In subclause 4.1 „Overview of the codec building blocks" add the following paragraph before the „Mixer" paragraph:

The sample rate converter block converts between the core decoder sampling rate and the decoder output sampling rate. It enables a constant output sampling rate in case of varying core coding sampling rates and allows for resampling of audio scenes, which may be coded in multiple sub-streams with different core sampling rates, to a common output sampling rate.

Note that in case DRC-2/3 gains need to be applied in the post-processing context (see Figure 2), the DRC-2/3 gains should be resampled similar to the audio samples.

Add a new clause Amd3.3 „Sample Rate Converter " for the sample rate converter block to the specification with the following text:

The sample rate converter performs sample rate conversion between the core coder sampling rate and the output sampling rate of the decoder. The signal delay introduced by the sample rate converter shall not exceed 256 samples, measured in samples at the output sampling rate of the decoder. The sample rate converter shall be able to perform sampling rate conversions with resampling ratios (defined as decoder output sampling rate divided by core decoder output sampling rate) of 3/2, 2, and 3. The sample rate converter block shall further be able to pass through the signal(s) if the core decoder sampling rate matches the decoder output sampling rate.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

15 Low Complexity Downmix

In "4.4 Rule set for determining processing domains" add to "4.4.4 Rendering Context" the following sentence:

In case the core decoder operates in time domain (TD), i.e. if the core decoder does not apply QMF domain processing like SBR or MPS-212, the low-complexity STFT domain downmix as specified in clauses *FC* and *FC5* shall be applied for format conversion if format conversion is requested.

Replace Table 56

immersiveDownmixFlag	Meaning
0	Generic format converter shall be applied as defined in clause 10.
1	If the local loudspeaker setup, signaled by LoudspeakerRendering(), is signaled as (speakerLayoutType==0,CICPspeakerLayoutIdx==5) or as (speakerLayoutType==0,CICPspeakerLayoutIdx==6), independently of potentially signaled loudspeaker displacement angles, then immersive rendering format converter shall be applied as defined in clause 11. In all other case the generic format converter shall be applied as defined in clause 10.

With

immersiveDownmixFlag	Meaning
0	Generic format converter shall be applied as defined in clauses 10 and *FC* according to 4.4.4
1	If the local loudspeaker setup, signaled by LoudspeakerRendering(), is signaled as (speakerLayoutType==0,CICPspeakerLayoutIdx==5) or as (speakerLayoutType==0,CICPspeakerLayoutIdx==6), independently of potentially signaled loudspeaker displacement angles, then immersive rendering format converter shall be applied as defined in clauses 11 and *FC5* according to 4.4.4. In all other case the generic format converter shall be applied as defined in clauses 10 and *FC* according to 4.4.4.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Add a new clause FC, "Low Complexity Generic Loudspeaker Rendering/Format Conversion" with the following text:

FC Low Complexity Generic Loudspeaker Rendering/Format Conversion

FC.1 Description

The loudspeaker renderer converts multichannel signals from transmitted channel configurations to desired reproduction formats. It is thus also called 'format converter'. If the channel configuration of the channels routed to the format converter exactly matches the signaled reproduction layout (i.e. the target channel configuration), the format converter shall be bypassed. The system consists of two major building blocks:

— An initialization algorithm that takes into account static parameters like the input and output format,

— a signal adaptive downmixing process that operates in a subband domain.

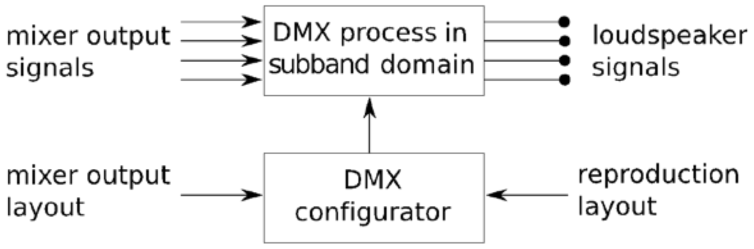


Figure FC.1 — Main building blocks of the low complexity generic format converter

In the initialization phase the format converter automatically generates optimized downmixing parameters (like the downmixing matrix) for the given combination of input and output formats: It applies an algorithm that selects for each input loudspeaker the most appropriate mapping rule from a list of rules that has been designed to incorporate psychoacoustic considerations. Each rule describes the mapping from one input channel to one or several output loudspeaker channels.

Input channels are

— either mapped to a single output channel,

— or panned to two output channels,

— or (in case of the 'Voice of God' channel) distributed over a larger number of output channels.

The optimal mapping for each input channel is selected depending on the list of output loudspeakers that are available in the desired output format. Each mapping defines downmix gains for the input channel under consideration as well as potentially also an equalizer that is applied to the input channel under consideration.

Output setups with non-standard loudspeaker positions can be signaled to the system by providing the azimuth and elevation deviations from a regular loudspeaker setup.

The actual downmixing of the audio signals is performed on a short time Fourier transform (STFT) representation of the signals. The energy-preserving algorithm avoids signal deteriorations like comb-filtering, coloration, or modulation artifacts.

FC.2 Definitions

FC.2.1 General remarks

Audio signals that are fed into the format converter are referred to as *input signals* in the following. Audio signals that are the result of the format conversion process are referred to as *output signals*. Note that the audio input signals of the format converter are audio output signals of the core decoder.

Vectors and matrices are denoted by bold-faced symbols. Vector elements or matrix elements are denoted as italic variables supplemented by indices indicating the row/column of the vector/matrix element in the vector/matrix, e.g. $[y_1 \cdots y_a \cdots y_N] = \mathbf{y}$ denotes a vector and its elements. Similarly, $M_{a,b}$ denotes the element in the a th row and b th column of a matrix $\mathbf{M}$.

FC.2.2 Variable definitions

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

N_{in}	Number of channels in the input channel configuration.
N_{out}	Number of channels in the output channel configuration.
$\mathbf{M}_{\text{DMX}}$	Downmix matrix containing real-valued non-negative downmix coefficients (downmix gains). $\mathbf{M}_{\text{DMX}}$ is of dimension (N_{out} x N_{in}).
$\mathbf{G}_{\text{EQ}}$	Matrix consisting of gain values per processing band determining frequency responses of equalizing filters.
$\mathbf{I}_{\text{EQ}}$	Vector signalling which equalizer filters to apply to the input channels (if any).
L	Frame length measured in time domain audio samples.
ν	Time domain sample index.
F	Frame index (frame number).
PB	Number of processing band, *PB*=58.
pb	Processing band index ($0 \le pb < PB$).
N	DFT length
K	Number of STFT frequency bins, K = 257.
k	STFT frequency bin index ($0 \le k < K$).
α	Filter parameter, $\alpha = 0.0435$.
A, B	Channel indices.
eps	Numerical constant, $\text{eps} = 10^{-35}$.

FC.3 Processing

FC.3.1 Application of transmitted downmix matrices

FC.3.1.1 Introduction

MPEG-H 3D audio allows transmission of downmix specifications for specific target channel configurations. downmixIds are assigned to the transmitted downmix specifications, allowing DRC to adapt to the downmix specification applied in the MPEG-H decoder (e.g. to select an appropriate DRC gain sequence). Further, loudness metadata values may be coupled with downmixIds. downmixIds are transmitted in the bitstream together with the downmixType as well as the nominal loudspeaker layouts the embedded downmix matrices (and/or DRC and loudness data) have been designed for.

In the MPEG-H 3D audio decoder the selection of a downmixIds thus

— determines whether transmitted downmix coefficients (downmixType=1) or decoder side generated downmix coefficients (downmixType=0) are applied in the downmix process;

— influences DRC/loudness processing.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Two arguments speak in favor of applying transmitted downmix coefficients also for moderately displaced reproduction layouts:

— The selection of a significantly different downmix matrix (transmitted vs. decoder generated) results in large perceptual changes of the downmix result.

— The artistic intent of transmitted downmix coefficients would be lost if a transmitted downmix matrix is not applied.

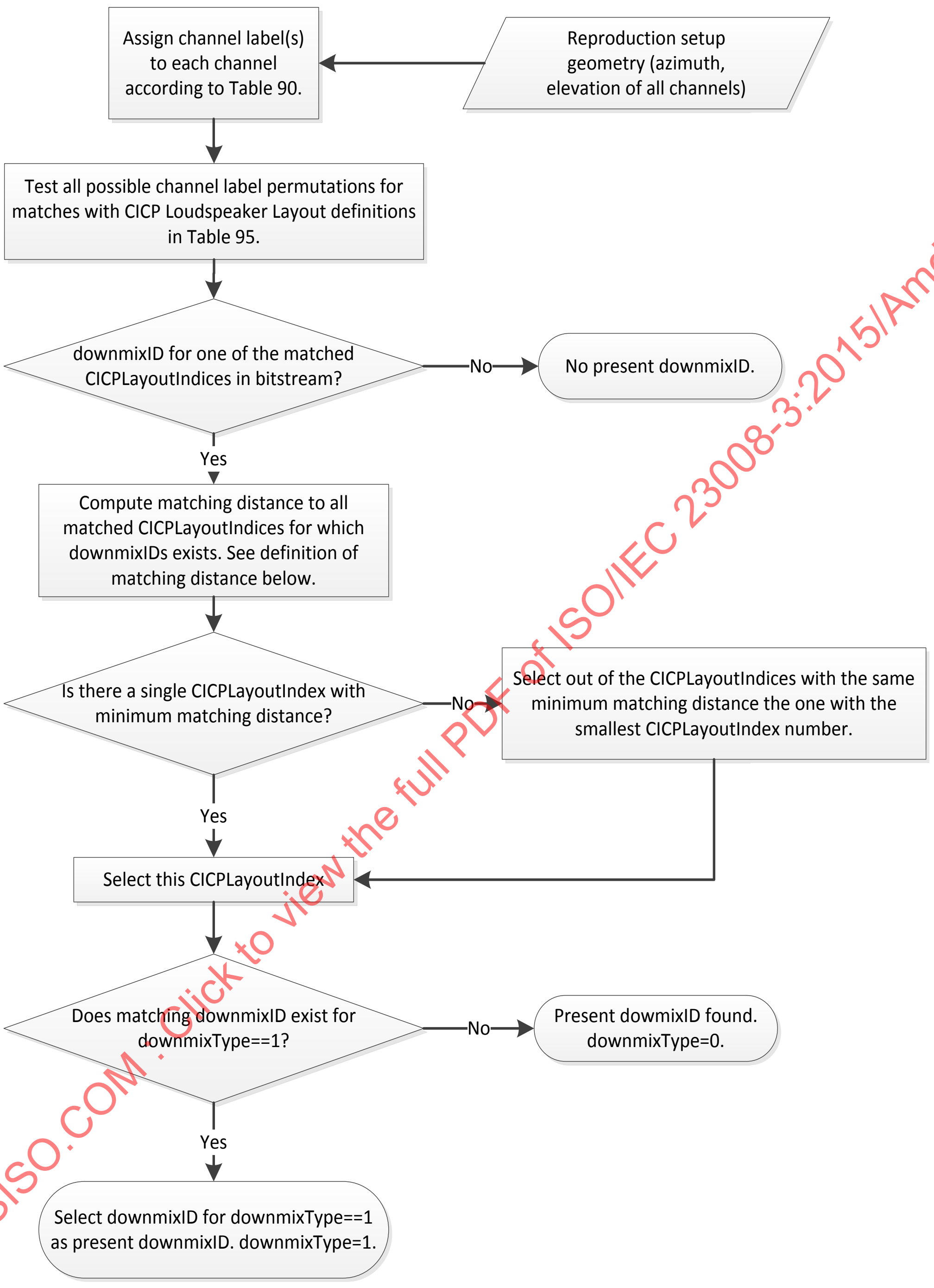
Of course, in case the loudspeaker displacements of the reproduction setup are too large, the application of the transmitted downmix coefficients may result in a larger perceptual distance from the intended reproduction than the application of decoder generated downmix coefficients. As a consequence, the allowed loudspeaker displacement values are restricted in the following matching scheme.

FC.3.1.2 Loudspeaker layout matching scheme

The downmixId matching algorithm is specified by the flow chart of Figure FC.2 below. It takes as input the geometry of the actual reproduction setup and outputs as result the present downmixId (if any). The present downmixId determines the downmix processing in the decoder as shown in FC.3.1.4. Further, the present downmixId may affect DRC and loudness processing.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved


Assign channel label(s) to each channel according to Table 90.
Reproduction setup geometry (azimuth, elevation of all channels)
Test all possible channel label permutations for matches with CICP Loudspeaker Layout definitions in Table 95.
downmixID for one of the matched CICPLayoutIndices in bitstream?
No
No present downmixID.
Yes
Compute matching distance to all matched CICPLayoutIndices for which downmixIDs exists. See definition of matching distance below.
Is there a single CICPLayoutIndex with minimum matching distance?
No
Select out of the CICPLayoutIndices with the same minimum matching distance the one with the smallest CICPLayoutIndex number.
Yes
Select this CICPLayoutIndex
Does matching downmixID exist for downmixType==1?
No
Present dowmixID found. downmixType=0.
Yes
Select downmixID for downmixType==1 as present downmixID. downmixType=1.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Figure FC.2 — Flow diagram for loudspeaker layout matching and to determine downmixId

The *matching distance* is defined as the sum of all absolute azimuth and elevation angle differences between the channel positions of the reproduction layout and the tested CICP loudspeaker layout, summed over all channels of the reproduction setup, excluding the LFE channels:

$$d_{\text{matching}} = \sum_{ch \in \left\{ \begin{smallmatrix} \text{all channels} \\ \text{except LFEs} \end{smallmatrix} \right\}} \left| \varphi_{\text{CICPLayout},ch} - \varphi_{\text{reprodcutionLayout},ch} \right| + \left| \vartheta_{\text{CICPLayout},ch} - \vartheta_{\text{reprodcutionLayout},ch} \right| ,$$

where φ denotes azimuth angles and ϑ denotes elevation angles.

Table FC.1 — Channel matching tolerances for matching downmixIds to reproduction layouts. Azimuth and elevation tolerance intervals are defined as sectors, where azimuth start and end values are connected in counterclockwise direction and elevation start and end values are connected in ascending elevations. Start and end values of the sectors are considered part of the sectors.

LoudspeakerGeometry as defined in ISO/IEC 23001-8)	Channel	Azimuth [deg]	Elevation [deg]	Azimuth start angle of sector [deg]	Azimuth end angle of sector [deg]	Elevation start angle of sector [deg]	Elevation end angle of sector [deg]	Ch. is LFE	Position is relative
	CH_EMPTY	n/a	n/a	n/a	n/a	n/a	n/a	0	0
0	CH_M_L030	30	0	15	45	-15	15	0	0
1	CH_M_R030	-30	0	-45	-15	-15	15	0	0
2	CH_M_000	0	0	-10	10	-15	15	0	0
3	CH_LFE1	0	n/a	n/a	n/a	n/a	n/a	1	0
4	CH_M_L110	110	0	90	130	-15	15	0	0
5	CH_M_R110	-110	0	-130	-90	-15	15	0	0
6	CH_M_L022	22	0	7	37	-15	15	0	0
7	CH_M_R022	-22	0	-37	-7	-15	15	0	0
8	CH_M_L135	135	0	120	150	-15	15	0	0
9	CH_M_R135	-135	0	-150	-120	-15	15	0	0
10	CH_M_180	180	0	170	190	-15	15	0	0
13	CH_M_L090	90	0	70	110	-15	15	0	0
14	CH_M_R090	-90	0	-110	-70	-15	15	0	0
15	CH_M_L060	60	0	40	80	-15	15	0	0
16	CH_M_R060	-60	0	-80	-40	-15	15	0	0
17	CH_U_L030	30	35	15	45	15	55	0	0
18	CH_U_R030	-30	35	-45	-15	15	55	0	0
19	CH_U_000	0	35	-15	15	15	55	0	0
20	CH_U_L135	135	35	115	155	15	55	0	0
21	CH_U_R135	-135	35	-155	-115	15	55	0	0
22	CH_U_180	180	35	165	195	15	55	0	0
23	CH_U_L090	90	35	70	110	15	55	0	0
24	CH_U_R090	-90	35	-110	-70	15	55	0	0
25	CH_T_000	0	90	-180	180	60	90	0	0
26	CH_LFE2	45	n/a	n/a	n/a	n/a	n/a	1	0
27	CH_L_L045	45	-15	25	65	-40	0	0	0
28	CH_L_R045	-45	-15	-65	-25	-40	0	0	0
29	CH_L_000	0	-15	-15	15	-40	0	0	0
30	CH_U_L110	110	35	90	130	15	55	0	0
31	CH_U_R110	-110	35	-130	-90	15	55	0	0
32	CH_U_L045	45	35	30	60	15	55	0	0
33	CH_U_R045	-45	35	-60	-30	15	55	0	0
34	CH_M_L045	45	0	30	60	-15	15	0	0
35	CH_M_R045	-45	0	-60	-30	-15	15	0	0
36	CH_LFE3	-45	n/a	n/a	n/a	n/a	n/a	1	0

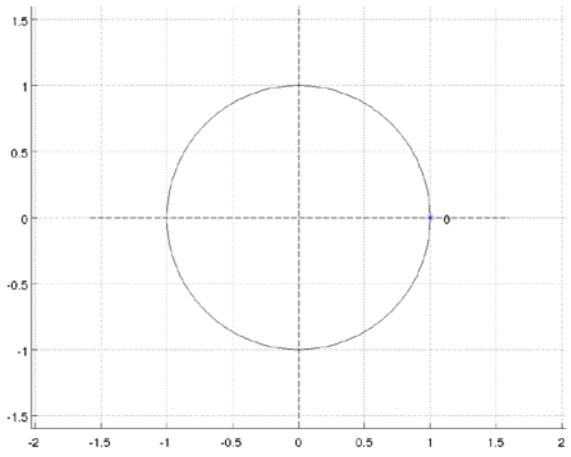
STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

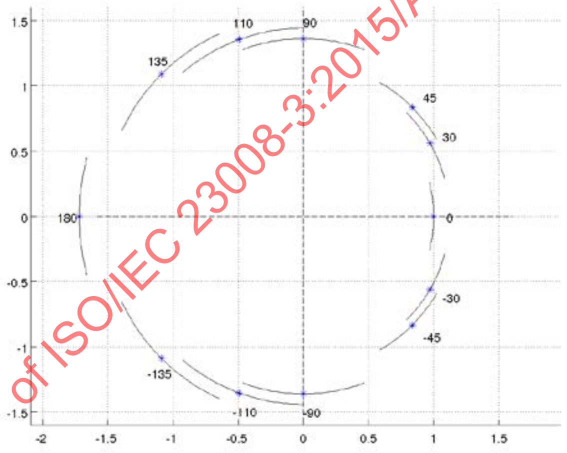
37	CH_M_LSCR	60	0	15	80	-15	15	0	1
38	CH_M_RSCR	-60	0	-80	-15	-15	15	0	1
39	CH_M_LSCH	30	0	7	40	-15	15	0	1
40	CH_M_RSCH	-30	0	-40	-7	-15	15	0	1
41	CH_M_L150	150	0	135	165	-15	15	0	0
42	CH_M_R150	-150	0	-165	-135	-15	15	0	0

FC.3.1.3 Visualization of azimuth tolerances

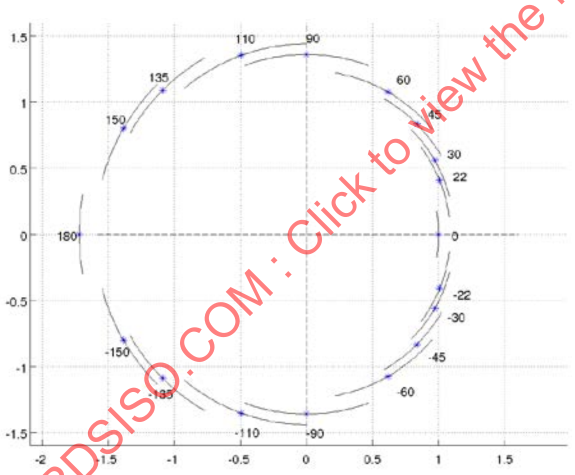
The following figures depict the azimuth matching sectors for each loudspeaker. The tolerances reflect the non-isotropic sound localization performance of the human auditory system.



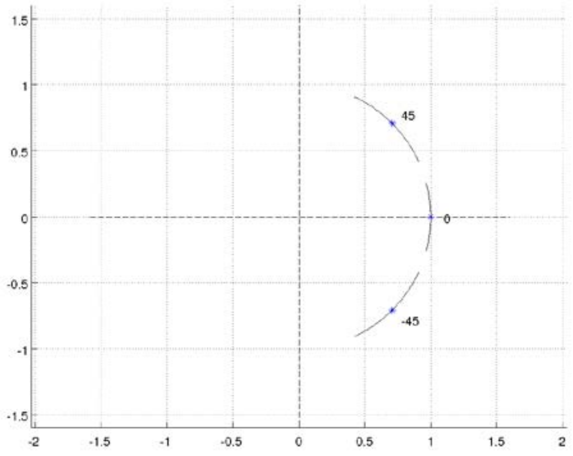
a) top layer



b) upper layer



c) middle layer



d) lower / bottom layer

Figure FC.3 — Visualization of azimuth tolerances.
Arcs have been plotted on different radii just for clarity of presentation. Nominal positions of the channels have been marked with asterisks and labels indicating the nominal azimuth angle in degrees.

FC.3.1.4 Determination of downmix processing depending on present downmixId

The downmix processing in the MPEG-H decoder is determined by the present downmixId as follows:

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

— If there is no present downmixId for the current reproduction setup, the downmix coefficients shall be derived as specified in the format converter initialization.

— If downmixType==0 for the present downmixId, the downmix coefficients shall be derived as specified in the format converter initialization.

— If downmixType==1 for the present downmixId, the downmix coefficients transmitted with the downmixId shall be applied in the downmix.

FC.3.2 Application of transmitted equalizer settings

Equalizer settings may be transmitted together with downmix matrices, as indicated by the **equalizerPresent** bitstream element. In case equalizer settings have been transmitted together with a downmix matrix that is applied in the format converter, the equalizers shall be applied to this downmix matrix as follows:

The transmitted equalizer parameters shall be decoded into frequency dependent gains (i.e. into equalizer frequency responses) according to FC.3.4.6.4. Next, the transmitted downmix matrix gains shall be multiplied by the frequency dependent equalizer gains to arrive at the final frequency dependent downmix matrix that shall be applied in the downmix. Note that the assignment of equalizer gains to downmix matrix elements is given by the vector **equalizerIndex** that is derived according to Table 21 - Syntax of EqualizerConfig: **equalizerIndex** tells for each input channel whether an equalizer (and if any: which) shall be applied to an input channel by applying the corresponding equalizer gains to all downmix matrix coefficients associated with the respective input channel.

FC.3.3 Downmix processing involving multiple channel groups

FC.3.3.1 General

In case multiple channel groups are transmitted in the MPEG-H 3D Audio bitstream and routed to the format converter, one instance of the format converter shall perform a downmix of all input channels routed to the format converter to the desired target channel configuration. Therefore all channels routed to the format converter are compiled in a group of channels that constitutes the input to the format conversion process.

If downmix matrices to one or more target channel configurations are transmitted in the bitstream, downmix matrices for those target channel configurations shall be transmitted for all channel groups/channel elements present in the bitstream.

FC.3.3.2 Downmix processing with decoder generated downmix gains

In case no appropriate downmix matrices have been transmitted for the signaled target loudspeaker configuration, the downmix gains are generated during the initialization of the format converter according to FC.3.4. The channels are fed to the format converter as a group of all input channels routed to the format converter. The input channel configuration signaled to the format converter shall reflect the channel geometry of this group of channels.

FC.3.3.3 Downmix processing with transmitted downmix gains

In case downmix matrices have been transmitted that are applicable to the desired target channel configuration, those downmix matrices shall be applied in the format converter instead of generating downmix gains in the format converter initialization process. Whether a downmix matrix is applicable for a desired target setup, or not, is determined according to FC.3.1.

All channels (and/or groups of channels) shall be compiled in one group of channels, where this group of channels consists of blocks of channels that are the channels (and/or groups of channels) routed to the format converter.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

The transmitted downmix matrices assigned to the channels routed to the format converter shall be concatenated according to the order of the blocks of channels in the group of channels that forms the input to the format converter. The concatenated downmix matrix shall then be applied in the format converter to derive the downmixed signal in the desired target channel configuration.

FC.3.4 Initialization of the format converter

FC.3.4.1 General description of the initialization

The initialization of the format converter is carried out before processing of the audio samples delivered by the core decoder takes place.

The initialization takes into account as input parameters:

— The sampling rate of the audio data to process.

— The channel configuration of the audio data to process with the format converter (number and geometric positions of input channels).

— The channel configuration of the desired output format (number and geometric positions of output channels).

— Optional: Parameters signaling the deviation of the output loudspeaker positions from a standard loudspeaker setup (random setup functionality).

It returns

— A frequency dependent downmix matrix $\mathbf{M}_{\mathrm{DMX}}$ that is applied in the audio signal processing of the format converter. $\mathbf{M}_{\mathrm{DMX}}$ is also taken into account in the core decoding process, see 5.5.3.1.2.

The input parameters to the initialization algorithm are listed in Table FC.2.

Table FC.2 — Format converter initialization input parameters

	Input format: number of channels and nominal channel setup geometry
	Output format: number of channels and nominal channel setup geometry
f_s	Sampling frequency in Hertz.
$r_{\mathrm{azi},A}$	For each output channel *A*, an azimuth angle is specified, determining the deviation from the standard format loudspeaker azimuth.
$r_{\mathrm{ele},A}$	For each output channel *A*, an elevation angle is specified, determining the deviation from the standard format loudspeaker elevation.

Table FC.3 lists the output parameters that are derived during the initialization of the format converter.

Table FC.3 — Format converter initialization output parameters

$\mathbf{M}_{\mathrm{DMX}}$	Downmix matrix [linear gains]

FC.3.4.2 Assignment of format converter channel labels to input/output format channels

The format converter initialization is based on a system of rules that are defined in terms of *format converter channel labels*, see Table FC.5. To allow the application of the initialization rules, the channel labels have to be assigned to the channels of the input and output formats. Each format converter channel label is

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

associated with a segment of the surface of the unit sphere, as defined in Table FC.5. The segments are designed non-overlapping.

The assignment of channel labels to channels is done by geometrically matching the segments to the position data associated with the channels of the input and output formats. The azimuth and elevation angles in degrees of the position data associated with the channels shall be rounded towards the nearest integer number before performing the channel label assignment. Note that the *nominal* channel positions shall be applied in the following matching to channel label sectors, i.e. the azimuth and elevation angles *without* taking into account potential angle deviations signalled in $r_{\mathrm{azi},A}$ and/or $r_{\mathrm{ele},A}$.

For each channel that is not an LFE (low-frequency effects) channel:

If the nominal position of the current channel, defined by its azimuth angle and elevation angle, is within or on the border of one of the segments defined in Table FC.5 then:

Assign the corresponding channel label (e.g. CH_M_L030) associated with the matching segment.

Add the angle differences between the nominal position of the current channel and the nominal position associated with the matching segment (i.e. the angles in the second and third column of Table FC.5) to the angle deviations stored in $r_{\mathrm{azi},A}$ and $r_{\mathrm{ele},A}$.

Else (i.e. no matching sector found), then:

Assign the CH_EMPTY label.

If an input or output format contains exactly one LFE channel, then the label CH_LFE2 shall be assigned to this channel.

If an input or output format contains exactly two LFE channels, then the labels CH_LFE2 and CH_LFE3 shall be assigned to the two LFE channels in the order that minimizes the maximum azimuth distance from the channels to the assigned CH_LFE2 and CH_LFE3 nominal azimuth positions.

If an input or output format contains more than 2 LFE channels, then those 2 LFE channels out of the considered setup shall be selected that minimize the maximum azimuth distance to the CH_LFE2 and CH_LFE3 nominal azimuth positions. The labels CH_LFE2 and CH_LFE3 shall be assigned as in the case of two LFE channels. The remaining LFE channels shall not be considered further in the calculation of downmix coefficients, i.e. the corresponding lines/columns of the downmix matrix shall remain filled with zeros.

FC.3.4.3 Handling for unknown input channels

If the label CH_EMPTY is assigned to an input channel, this channel shall be considered unknown to the rules-based initialization and the downmix coefficients for mapping this input channel to the output channels shall be derived as specified in FC.3.4.6.7.

FC.3.4.4 Handling for unknown output formats

If the output format contains at least one channel with the label CH_EMPTY assigned to it, or if at least one channel label is assigned to more than one channel of the output format, the output format shall be considered unknown and the derivation of the downmixing coefficients shall be carried out as specified in FC.3.4.6.7. The rules-based derivation of downmix coefficients shall not be applied for unknown output formats.

FC.3.4.5 Handling of deviations from standard loudspeaker positions

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

If the below conditions are not met, the rules-based initialization is considered to have failed, the output format shall be considered to be unknown, and the downmixing gains shall be obtained as defined in FC.3.4.6.7.

The absolute values of $r_{\mathrm{azi},A}$ and $r_{\mathrm{ele},A}$ shall not exceed 35 and 55 degrees, respectively. The minimum angle between any loudspeaker pair (without LFE channels) shall not be smaller than 15 degrees.

The values of $r_{\mathrm{azi},A}$ shall be such that the ordering by azimuth angles of the horizontal loudspeakers does not change. Likewise, the ordering of the height and low loudspeakers shall not change.

The values of $r_{\mathrm{ele},A}$ shall be such that the ordering by elevation angles of loudspeakers which are (approximately) above/below each other does not change. To verify this, the following procedure is applied:

For each row of Table FC.11 which contains two or three channels of the output format, do:

— Order the channels by elevation without randomization

— Order the channels by elevation with considering randomization

— If the two orderings differ, return an initialization error

FC.3.4.6 Rules-Based Initialization algorithm

FC.3.4.6.1 General

The rules-based initialization algorithm is defined in the following. It shall not be applied if the output format is considered unknown as defined above. Note that the following description makes use of the intermediate parameters defined in Table FC.4 only for clarity reasons. An implementation may omit the introduction of these intermediate parameters.

Table FC.4 — Format converter initialization intermediate parameters

$\mathbf{S}$	Vector of converter source channels [input channel indices]
$\mathbf{D}$	Vector of converter destination channels [output channel indices]
$\mathbf{G}$	Vector of converter gains [linear]
$\mathbf{E}$	Vector of converter EQ indices
$\mathbf{G}_{\mathrm{EQ}}$	Matrix containing equalizer gain values for all EQ indices and processing bands

The intermediate parameters describe the dowmixing parameters in a mapping-oriented way, i.e. as sets of parameters S_i, D_i, G_i, E_i per mapping *i*.

The format converter initialization output parameters are derived as described in the following steps:

FC.3.4.6.2 Random setups Pre-Processing:

Random output loudspeaker setups, i.e. output setups that contain loudspeakers at positions deviating from the positions defined for the desired output format are signalled by specifying the loudspeaker position deviation angles as input parameters $r_{\mathrm{azi},A}$ and $r_{\mathrm{ele},A}$. The angle deviations are taken into account as a pre-processing step:

Modify in Table FC.5 the channels' azimuth and elevation angles by adding $r_{\mathrm{azi},A}$ and $r_{\mathrm{ele},A}$ to the corresponding channels' azimuth and elevation angles.

FC.3.4.6.3 Derivation of input channel/output channel mapping

parameters:

© ISO/IEC 2017 – All rights reserved

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

The parameters vectors $\mathbf{S}$, $\mathbf{D}$, $\mathbf{G}$, $\mathbf{E}$ define the mapping of input channels to output channels. For each mapping *i* from an input channel to an output channel with non-zero downmix gain they define the downmix gain as well as an equalizer index that indicates which equalizer curve has to be applied to the input channel under consideration in mapping *i*.

The elements of the parameter vectors $\mathbf{S}$, $\mathbf{D}$, $\mathbf{G}$, $\mathbf{E}$ are derived by the following algorithm:

Initialize the mapping counter *i*: $i = 1$;

For each input channel, ignoring channels with label CH_EMPTY assigned to them:

If the input channel also exists in the output format (e.g. input channel under consideration is CH_M_R030 and channel CH_M_R030 exists in the output format), then:

> S_i = index of source channel in input ***Example:*** *channel CH_M_R030 in ChannelConfiguration 6 is at second place according to Table FC.6 i.e. has index 2 in this format)*
>
> D_i = index of same channel in output
>
> G_i = 1.0
>
> E_i = 0
>
> $i = i + 1$

Else (i.e. if the input channel does not exist in the output format)

> search the first entry of this channel in the **Source** column of Table FC.7, for which the channels in the corresponding row of the **Destination** column exist. The ALL_U destination shall be considered valid (i.e. the relevant output channels exist) if the output format contains at least one "CH_U_" channel. The ALL_M destination shall be considered valid (i.e. the relevant output channels exist) if the output format contains at least one "CH_M_" channel. If for no entry in Table FC.7 corresponding to the input channel the channels in the **Destination** column exist, the rules-based initialization shall terminate and the downmix gains shall be derived according to FC.3.4.6.7.
>
> If **Destination** column contains ALL_U, then:
>
> > For each output channel x with "CH_U_" in its name, do:
> >
> > > S_i = index of source channel in input
> > >
> > > D_i = index of channel x in output
> > >
> > > G_i = (value of Gain column) / sqrt(number of "CH_U_" output channels)
> > >
> > > E_i = value of EQ column
> > >
> > > $i = i + 1$
>
> Else if **Destination** column contains ALL_M, then:

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
        For each output channel x with “CH_M_” in its name, do:
                $S_i$ = index of source channel in input
                $D_i$ = index of channel x in output
                $G_i$ = (value of Gain column) / sqrt(number of “CH_M_” output channels)
                $E_i$ = value of EQ column
                $i = i + 1$
Else If there is one channel in the **Destination** column, then:
        $S_i$ = index of source channel in input
        $D_i$ = index of destination channel in output
        $G_i$ = value of Gain column
        $E_i$ = value of EQ column
        $i = i + 1$
Else (two channels in **Destination** column)
        $S_i$ = index of source channel in input
        $D_i$ = index of first destination channel in output
        $G_i$ = (value of Gain column) * $g_1$
        $E_i$ = value of EQ column
        $i = i + 1$

        $S_i$ = $S_{i-1}$
        $D_i$ = index of second destination channel in output
        $G_i$ = (value of Gain column) * $g_2$
        $E_i$ = $E_{i-1}$
        $i = i + 1$
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

The gains g_1 and g_2 are computed by applying tangent law amplitude panning in the following way:

— Unwrap source destination channel azimuth angles to be positive.

— The azimuth angles of the destination channels are α_1 and α_2 (see Table FC.5).

— The azimuth angle of the source channel (= panning target) is α_{src}.

— $\alpha_0 = \dfrac{|\alpha_1 - \alpha_2|}{2}$

— $\alpha_{\mathrm{center}} = \dfrac{\alpha_1 + \alpha_2}{2}$

— $\alpha = (\alpha_{\mathrm{center}} - \alpha_{\mathrm{src}}) \cdot \mathrm{sgn}(\alpha_2 - \alpha_1)$

— $g_1 = \dfrac{g}{\sqrt{1+g^2}}, \quad g_2 = \dfrac{1}{\sqrt{1+g^2}} \qquad \text{with} \qquad g = \dfrac{\tan\alpha_0 - \tan\alpha + 10^{-10}}{\tan\alpha_0 + \tan\alpha + 10^{-10}}$

FC.3.4.6.4 Derivation of equalizer gains $\mathbf{G}_{\mathrm{EQ}}$:

$\mathbf{G}_{\mathrm{EQ}}$ consists of gain values per processing band *pb* and equalizer index *e*. The 5 predefined equalizers are combinations of different peak filters. Each equalizer is a serial cascade of one or more peak filters and a gain:

$$G_{EQ,e}^{pb} = 10^{g/20} \prod_{n=1}^{N} peak\left(band(pb)\frac{f_s}{2}, P_{f,n}, P_{Q,n}, P_{g,n} \right)$$

where *band*(*pb*) is the normalized center frequency of processing band *pb*, specified in Table FC.8, f_s is the sampling frequency, and function *peak()* is for negative *G*

$$\mathrm{peak}(b, f, Q, G) = \sqrt{\frac{b^4 + \left(\frac{1}{Q^2} - 2\right) f^2 b^2 + f^4}{b^4 + \left(\frac{10^{\frac{-G}{10}}}{Q^2} - 2\right) f^2 b^2 + f^4}}$$

and otherwise

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

$$\mathrm{peak}(b, f, Q, \mathrm{G}) = \sqrt{\frac{b^4 + \left(\frac{10^{\frac{G}{10}}}{Q^2} - 2\right) f^2 b^2 + f^4}{b^4 + \left(\frac{1}{Q^2} - 2\right) f^2 b^2 + f^4}}$$

The parameters for the equalizers are specified in Table FC.10.

FC.3.4.6.5 Post-Processing for Random Setups

Once the output parameters are computed, they are modified related to the specific random azimuth and elevations angles. This step has only to be carried out, if not all $r_{\mathrm{ele},A}$ are zero. Definition of the post-processing algorithm:

For each element *i* in D_i, do:

if output channel with index D_i is a horizontal channel by definition (i.e. output channel label contains the label '_M_'), **and**

> **if** this output channel is now a height channel (elevation in range 0..60 degrees), **and**
>
> > **if** input channel with index S_i is a height channel (i.e. label contains '_U_'), **then**
> >
> > — *h* = min(elevation of randomized output channel, 35) / 35
> >
> > — $G_{\mathrm{comp}} = h \cdot \frac{1}{0.85} + (1-h)$
> >
> > — Apply compensation gain to DMX gain: $G_i = G_i \cdot G_{\mathrm{comp}}$
> >
> > — Define new equalizer with a new index *e*, where $G_{EQ,e}^{pb} = h + (1-h) \cdot \mathrm{G}_{EQ,E_i}^{pb}$
> >
> > — $E_i = e$
> >
> > **else if** input channel with index S_i is a horizontal channel (label contains '_M_')
> >
> > — *h* = min(elevation of randomized output channel, 35) / 35
> >
> > — Define new equalizer with a new index *e*, where $G_{EQ,e}^{pb} = h \cdot \mathrm{G}_{EQ,5}^{pb} + (1-h) \cdot \mathrm{G}_{EQ,E_i}^{pb}$
> >
> > — $E_i = e$

Explanation of the post-processing steps defined above:

h is a normalized elevation parameter indicating the elevation of a nominally horizontal output channel ('_M_') due to a random setup elevation offset $r_{\mathrm{ele},A}$. For zero elevation offset *h*=0 follows and effectively no post-processing is applied.

The rules table (Table FC.7) in general applies a gain of 0.85 when mapping an upper input channel ('_U_' in channel label) to one or several horizontal output channels ('_M_' in channel label(s)). In case the output channel gets elevated due to a random setup elevation offset $r_{\mathrm{ele},A}$, the gain of 0.85 is partially (0<*h*<1) or fully (*h*=1) compensated for. Similarly the equalizer definitions fade towards a flat EQ-curve ($G_{EQ,e}^{pb} = const. = 1.0$) for *h* approaching *h* =1.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In case a horizontal input channel gets mapped to an output channel that gets elevated due to a random setup elevation offset $r_{\mathrm{ele},A}$, the equalizer $G_{EQ,5}^{pb}$ is partially (0<*h*<1) or fully (*h*=1) applied.

FC.3.4.6.6 Derivation of rules-based initialization downmix matrix:

$\mathbf{M}_{\mathrm{DMX}}$ is derived by rearranging the temporary parameters from the mapping-oriented representation (enumerated by mapping counter *i*) to a channel-oriented representation as defined in the following:

Initialize $\mathbf{M}_{\mathrm{DMX}}^{k}$ as an N_{out} x N_{in} zero matrix for all STFT bins *k*.

For each *i* do:

If ($E_i = 0$)

$$M_{\mathrm{DMX},A,B}^{k} = G_i \quad \text{for} \quad A = D_i,\ B = S_i,\ 0 \le k < K$$

Else

$$M_{\mathrm{DMX},A,B}^{k} = G_i \cdot G_{EQ,E_i}^{pb=pbm(k)} \quad \text{for} \quad A = D_i,\ B = S_i,\ 0 \le k < K$$

with the mapping *pbm* between processing bands *pb* and DFT frequency bins as defined in **Table** , and where $M_{\mathrm{DMX},A,B}^{k}$ denotes the matrix element in the Ath row and Bth column of $\mathbf{M}_{\mathrm{DMX}}^{k}$. Note that after the rules-based initialization this matrix of downmix coefficients will contain columns of zeros, if unknown channels are present in the input format. Those columns are filled with downmix gains as specified in FC.3.4.6.7.

FC.3.4.6.7 VBAP-based downmix coefficients derivation

Handling of unknown output formats:

In case the output format is considered unknown, the downmix coefficients for all input channels shall be derived as follows:

Each channel of the input setup is regarded as a static audio object at the position defined by the azimuth and elevation angles associated with the input channel. For each input channel the mixing gains to all output loudspeakers are calculated as VBAP panning gains $\mathbf{g}_{\mathrm{scaled}}$ according to 8.4.3, where the same output format shall be signaled to the VBAP algorithm as to the format converter. The panning gain vectors $\mathbf{g}_{\mathrm{scaled}}$ shall be post-processed according to FC.3.4.6.8.

The downmix matrix $\mathbf{M}_{\mathrm{DMX}}^{k}$ is finally derived by filling each matrix column with the post-processed panning gain vector elements of the corresponding input channel, independently of the DFT bin index k.

Handling of unknown input channels:

In case the input format contains unknown input channels, the downmix coefficients for these channels shall be derived as follows:

Each unknown channel of the input setup is regarded as a static audio object at the position defined by the azimuth and elevation angles associated with the input channel. For each unknown input channel the mixing gains to all output loudspeakers are calculated as VBAP panning gains $\mathbf{g}_{\mathrm{scaled}}$ according to 8.4.3, where the

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

same output format shall be signaled to the VBAP algorithm as to the format converter. The panning gain vectors $\mathbf{g}_{\mathbf{scaled}}$ shall be post-processed according to FC.3.4.6.8.

The downmix matrix $\mathbf{M}_{\mathrm{DMX}}^{k}$ is finally derived by filling each matrix column corresponding to an unknown input channel with the post-processed panning gain vector elements of the corresponding unknown input channel, independently of the DFT bin index k.

FC.3.4.6.8 VBAP gains post-processing

The mixing gains obtained from the VBAP rendering algorithm shall be post-processed to avoid excessive use of phantom sources. Therefore, small matrix gains are set to zero, followed by a renormalization of the panning gains to ensure energy-preservation:

For each panning gain vector $\mathbf{g}_{\mathbf{scaled}}$ do:

If the vector contains at least one panning gain that exceeds the threshold value 0.3, then

Set all vector elements smaller or equal to 0.3 to the value 0.0

Normalize the gain vector such that the sum of squares of the vector elements remains the same as before the post-processing

FC.3.4.6.9 Format converter initialization tables

Table FC.14 lists channel labels, corresponding azimuth and elevation angles, and associated sectors. The sectors are defined as points on the unit sphere, whose azimuth/elevation angles are within or on the borders of the intervals given by the azimuth/elevation start and end values in the table, connecting azimuth start and end values in counter-clockwise direction and connecting elevation start and end values in direction of increasing elevation angles.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table FC.14 — Channels definitions: Channel labels, corresponding azimuth and elevation angles, and associated sectors

LoudspeakerGeometry as defined in ISO/IEC 23001-8)	Channel	Azimuth [deg]	Elevation [deg]	Azimuth start angle of sector [deg]	Azimuth end angle of sector [deg]	Elevation start angle of sector [deg]	Elevation end angle of sector [deg]	Ch. is LFE	Position is relative
	CH_EMPTY	n/a	n/a	n/a	n/a	n/a	n/a	0	0
0	CH_M_L030	+30	0	+23	+37	-9	+20	0	0
1	CH_M_R030	-30	0	-37	-23	-9	+20	0	0
2	CH_M_000	0	0	-7	+7	-9	+20	0	0
3	CH_LFE1	0	n/a	n/a	n/a	n/a	n/a	1	0
4	CH_M_L110	+110	0	+101	+124	-45	+20	0	0
5	CH_M_R110	-110	0	-124	-101	-45	+20	0	0
6	CH_M_L022	+22	0	+8	+22	-9	+20	0	0
7	CH_M_R022	-22	0	-22	-8	-9	+20	0	0
8	CH_M_L135	+135	0	125	142	-45	+20	0	0
9	CH_M_R135	-135	0	-142	-125	-45	+20	0	0
10	CH_M_180	180	0	158	-158	-45	+20	0	0
13	CH_M_L090	+90	0	+76	+100	-45	+20	0	0
14	CH_M_R090	-90	0	-100	-76	-45	+20	0	0
15	CH_M_L060	+60	0	+53	+75	-9	+20	0	0
16	CH_M_R060	-60	0	-75	-53	-9	+20	0	0
17	CH_U_L030	+30	+35	+11	+37	+21	+60	0	0
18	CH_U_R030	-30	+35	-37	-11	+21	+60	0	0
19	CH_U_000	0	+35	-10	+10	+21	+60	0	0
20	CH_U_L135	+135	+35	+125	+157	+21	+60	0	0
21	CH_U_R135	-135	+35	-157	-125	+21	+60	0	0
22	CH_U_180	180	+35	+158	-158	+21	+60	0	0
23	CH_U_L090	+90	+35	+67	+100	+21	+60	0	0
24	CH_U_R090	-90	+35	-100	-67	+21	+60	0	0
25	CH_T_000	0	+90	-180	+180	+61	+90	0	0
26	CH_LFE2	+45	n/a	n/a	n/a	n/a	n/a	1	0
27	CH_L_L045	+45	-15	+11	+75	-45	-10	0	0
28	CH_L_R045	-45	-15	-75	-11	-45	-10	0	0
29	CH_L_000	0	-15	-10	+10	-45	-10	0	0
30	CH_U_L110	+110	+35	+101	+124	+21	+60	0	0
31	CH_U_R110	-110	+35	-124	-101	+21	+60	0	0
32	CH_U_L045	+45	+35	+38	+66	+21	+60	0	0
33	CH_U_R045	-45	+35	-66	-38	+21	+60	0	0
34	CH_M_L045	+45	0	+38	+52	-9	+20	0	0
35	CH_M_R045	-45	0	-52	-38	-9	+20	0	0
36	CH_LFE3	-45	n/a	n/a	n/a	n/a	n/a	1	0
37	CH_M_LSCR	+60	0	n/a	n/a	n/a	n/a	0	1
38	CH_M_RSCR	-60	0	n/a	n/a	n/a	n/a	0	1
39	CH_M_LSCH	+30	0	n/a	n/a	n/a	n/a	0	1
40	CH_M_RSCH	-30	0	n/a	n/a	n/a	n/a	0	1
41	CH_M_L150	+150	0	143	157	-45	+20	0	0
42	CH_M_R150	-150	0	-157	-143	-45	+20	0	0

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Table FC.6 — Formats with corresponding number of channels and channel ordering

Loudspeaker Layout Index or ChannelConfiguration as defined in ISO/IEC 23001-8	Number of Channels	Channels (with ordering)
1	1	CH_M_000
2	2	CH_M_L030, CH_M_R030
3	3	CH_M_L030, CH_M_R030, CH_M_000
4	4	CH_M_L030, CH_M_R030, CH_M_000, CH_M180
5	5	CH_M_L030, CH_M_R030, CH_M_000, CH_M_L110, CH_M_R110
6	6	CH_M_L030, CH_M_R030, CH_M_000, CH_LFE1, CH_M_L110, CH_M_R110
7	8	CH_M_L030, CH_M_R030, CH_M_000, CH_LFE1, CH_M_L110, CH_M_R110, CH_M_L060, CH_M_R060
8		n.a.
9	3	CH_M_L030, CH_M_R030, CH_M_180
10	4	CH_M_L030, CH_M_R030, CH_M_L110, CH_M_R110
11	7	CH_M_L030, CH_M_R030, CH_M_000, CH_LFE1, CH_M_L110, CH_M_R110, CH_M_180
12	8	CH_M_L030, CH_M_R030, CH_M_000, CH_LFE1, CH_M_L110, CH_M_R110, CH_M_L135, CH_M_R135
13	24	CH_M_L060, CH_M_R060, CH_M_000, CH_LFE2, CH_M_L135, CH_M_R135, CH_M_L030, CH_M_R030, CH_M_180, CH_LFE3, CH_M_L090, CH_M_R090, CH_U_L045, CH_U_R045, CH_U_000, CH_T_000, CH_U_L135, CH_U_R135, CH_U_L090, CH_U_R090, CH_U_180, CH_L_000, CH_L_L045, CH_L_R045
14	8	CH_M_L030, CH_M_R030, CH_M_000, CH_LFE1, CH_M_L110, CH_M_R110, CH_U_L030, CH_U_R030
15	12	CH_M_L030, CH_M_R030, CH_M_000, CH_LFE2, CH_M_L135, CH_M_R135, CH_LFE3, CH_M_L090, CH_M_R090, CH_U_L045, CH_U_R045, CH_U_180
16	10	CH_M_L030, CH_M_R030, CH_M_000, CH_LFE1, CH_M_L110, CH_M_R110, CH_U_L030, CH_U_R030, CH_U_L110, CH_U_R110
17	12	CH_M_L030, CH_M_R030, CH_M_000, CH_LFE1, CH_M_L110, CH_M_R110, CH_U_L030, CH_U_R030, CH_U_000, CH_U_L110, CH_U_R110, CH_T_000
18	14	CH_M_L030, CH_M_R030, CH_M_000, CH_LFE1, CH_M_L110, CH_M_R110, CH_M_L150, CH_M_R150, CH_U_L030, CH_U_R030, CH_U_000, CH_U_L110, CH_U_R110, CH_T_000
19	12	CH_M_L030, CH_M_R030, CH_M_000, CH_LFE1, CH_M_L135, CH_M_R135, CH_M_L090, CH_M_R090, CH_U_L030, CH_U_R030, CH_U_L135, CH_U_R135
20	14	CH_M_L030, CH_M_R030, CH_M_000, CH_LFE1, CH_M_L135, CH_M_R135, CH_M_L090, CH_M_R090, CH_U_L045, CH_U_R045, CH_U_L135, CH_U_R135, CH_M_LSCR, CH_M_RSCR

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table FC.7 — Converter rules matrix

Source	Destination	Gain	EQ index
CH_M_000	CH_M_L022, CH_M_R022	1.0	0 (off)
CH_M_000	CH_M_L030, CH_M_R030	1.0	0 (off)
CH_M_L022	CH_M_000, CH_M_L030	1.0	0 (off)
CH_M_L022	CH_M_L030	1.0	0 (off)
CH_M_R022	CH_M_000, CH_M_R030	1.0	0 (off)
CH_M_R022	CH_M_R030	1.0	0 (off)
CH_M_L045	CH_M_L030, CH_M_L060	1.0	0 (off)
CH_M_L045	CH_M_L030	1.0	0 (off)
CH_M_R045	CH_M_R030, CH_M_R060	1.0	0 (off)
CH_M_R045	CH_M_R030	1.0	0 (off)
CH_M_L060	CH_M_L045, CH_M_L110	1.0	0 (off)
CH_M_L060	CH_M_L030, CH_M_L110	1.0	0 (off)
CH_M_L060	CH_M_L030	0.8	0 (off)
CH_M_R060	CH_M_R045, CH_M_R110,	1.0	0 (off)
CH_M_R060	CH_M_R030, CH_M_R110,	1.0	0 (off)
CH_M_R060	CH_M_R030,	0.8	0 (off)
CH_M_L090	CH_M_L045, CH_M_L110	1.0	0 (off)
CH_M_L090	CH_M_L030, CH_M_L110	1.0	0 (off)
CH_M_L090	CH_M_L030	0.8	0 (off)
CH_M_R090	CH_M_R045, CH_M_R110	1.0	0 (off)
CH_M_R090	CH_M_R030, CH_M_R110	1.0	0 (off)
CH_M_R090	CH_M_R030	0.8	0 (off)
CH_M_L110	CH_M_L135	1.0	0 (off)
CH_M_L110	CH_M_L090	0.8	0 (off)
CH_M_L110	CH_M_L045	0.8	0 (off)
CH_M_L110	CH_M_L030	0.8	0 (off)
CH_M_R110	CH_M_R135	1.0	0 (off)
CH_M_R110	CH_M_R090	0.8	0 (off)
CH_M_R110	CH_M_R045	0.8	0 (off)
CH_M_R110	CH_M_R030	0.8	0 (off)
CH_M_L135	CH_M_L110	1.0	0 (off)
CH_M_L135	CH_M_L150	1.0	0 (off)
CH_M_L135	CH_M_L090	0.8	0 (off)
CH_M_L135	CH_M_L045	0.8	0 (off)
CH_M_L135	CH_M_L030	0.8	0 (off)
CH_M_R135	CH_M_R110	1.0	0 (off)
CH_M_R135	CH_M_R150	1.0	0 (off)
CH_M_R135	CH_M_R090	0.8	0 (off)
CH_M_R135	CH_M_R045	0.8	0 (off)
CH_M_R135	CH_M_R030	0.8	0 (off)
CH_M_L150	CH_M_L135	1.0	0 (off)
CH_M_L150	CH_M_L110	1.0	0 (off)
CH_M_L150	CH_M_L045	0.8	0 (off)
CH_M_L150	CH_M_L030	0.8	0 (off)
CH_M_R150	CH_M_R135	1.0	0 (off)
CH_M_R150	CH_M_R110	1.0	0 (off)
CH_M_R150	CH_M_R045	0.8	0 (off)
CH_M_R150	CH_M_R030	0.8	0 (off)
CH_M_180	CH_M_R150, CH_M_L150	1.0	0 (off)
CH_M_180	CH_M_R135, CH_M_L135	1.0	0 (off)
CH_M_180	CH_M_R110, CH_M_L110	1.0	0 (off)
CH_M_180	CH_M_R090, CH_M_L090	0.8	0 (off)

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

CH_M_180	CH_M_R045, CH_M_L045	0.6	0 (off)
CH_M_180	CH_M_R030, CH_M_L030	0.6	0 (off)
CH_U_000	CH_U_L030, CH_U_R030	1.0	0 (off)
CH_U_000	CH_M_L030, CH_M_R030	0.85	0 (off)
CH_U_L045	CH_U_L030	1.0	0 (off)
CH_U_L045	CH_M_L045	0.85	1
CH_U_L045	CH_M_L030	0.85	1
CH_U_R045	CH_U_R030	1.0	0 (off)
CH_U_R045	CH_M_R045	0.85	1
CH_U_R045	CH_M_R030	0.85	1
CH_U_L030	CH_U_L045	1.0	0 (off)
CH_U_L030	CH_M_L030	0.85	1
CH_U_R030	CH_U_R045	1.0	0 (off)
CH_U_R030	CH_M_R030	0.85	1
CH_U_L090	CH_U_L030, CH_U_L110	1.0	0 (off)
CH_U_L090	CH_U_L030, CH_U_L135	1.0	0 (off)
CH_U_L090	CH_U_L045	0.8	0 (off)
CH_U_L090	CH_U_L030	0.8	0 (off)
CH_U_L090	CH_M_L045, CH_M_L110	0.85	2
CH_U_L090	CH_M_L030, CH_M_L110	0.85	2
CH_U_L090	CH_M_L030	0.85	2
CH_U_R090	CH_U_R030, CH_U_R110	1.0	0 (off)
CH_U_R090	CH_U_R030, CH_U_R135	1.0	0 (off)
CH_U_R090	CH_U_R045	0.8	0 (off)
CH_U_R090	CH_U_R030	0.8	0 (off)
CH_U_R090	CH_M_R045, CH_M_R110	0.85	2
CH_U_R090	CH_M_R030, CH_M_R110	0.85	2
CH_U_R090	CH_M_R030	0.85	2
CH_U_L110	CH_U_L135	1.0	0 (off)
CH_U_L110	CH_U_L045	0.8	0 (off)
CH_U_L110	CH_U_L030	0.8	0 (off)
CH_U_L110	CH_M_L110	0.85	2
CH_U_L110	CH_M_L045	0.85	2
CH_U_L110	CH_M_L030	0.85	2
CH_U_R110	CH_U_R135	1.0	0 (off)
CH_U_R110	CH_U_R045	0.8	0 (off)
CH_U_R110	CH_U_R030	0.8	0 (off)
CH_U_R110	CH_M_R110	0.85	2
CH_U_R110	CH_M_R045	0.85	2
CH_U_R110	CH_M_R030	0.85	2
CH_U_L135	CH_U_L110	1.0	0 (off)
CH_U_L135	CH_U_L045	0.8	0 (off)
CH_U_L135	CH_U_L030	0.8	0 (off)
CH_U_L135	CH_M_L110	0.85	2
CH_U_L135	CH_M_L045	0.85	2
CH_U_L135	CH_M_L030	0.85	2
CH_U_R135	CH_U_R110	1.0	0 (off)
CH_U_R135	CH_U_R045	0.8	0 (off)
CH_U_R135	CH_U_R030	0.8	0 (off)
CH_U_R135	CH_M_R110	0.85	2
CH_U_R135	CH_M_R045	0.85	2
CH_U_R135	CH_M_R030	0.85	2
CH_U_180	CH_U_R135, CH_U_L135	1.0	0 (off)
CH_U_180	CH_U_R110, CH_U_L110	1.0	0 (off)
CH_U_180	CH_M_180	0.85	2
CH_U_180	CH_M_R110, CH_M_L110	0.85	2
CH_U_180	CH_U_R030, CH_U_L030	0.8	0 (off)

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

CH_U_180	CH_M_R030, CH_M_L030	0.85	2
CH_T_000	ALL_U	1.0	3
CH_T_000	ALL_M	1.0	4
CH_L_000	CH_M_000	1.0	0 (off)
CH_L_000	CH_M_L030, CH_M_R030	1.0	0 (off)
CH_L_L045	CH_M_L045	1.0	0 (off)
CH_L_L045	CH_M_L030	1.0	0 (off)
CH_L_R045	CH_M_R045	1.0	0 (off)
CH_L_R045	CH_M_R030	1.0	0 (off)
CH_LFE2	CH_LFE3	1.0	0 (off)
CH_LFE2	CH_M_L030, CH_M_R030	1.0	0 (off)
CH_LFE3	CH_LFE2	1.0	0 (off)
CH_LFE3	CH_M_L030, CH_M_R030	1.0	0 (off)

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Table FC.8 — Normalized center frequencies of the 58 processing bands

Normalized Frequency [0, 1]
0.000000000000000
0.003891050583658
0.007782101167315
0.011673151750973
0.015564202334630
0.019455252918288
0.023346303501946
0.027237354085603
0.031128404669261
0.035019455252918
0.038910505836576
0.042801556420233
0.046692607003891
0.050583657587549
0.054474708171206
0.058365758754864
0.062256809338521
0.066147859922179
0.070038910505837
0.073929961089494
0.077821011673152
0.081712062256809
0.085603112840467
0.089494163424125
0.093385214007782
0.097276264591440
0.101167315175097
0.105058365758755
0.108949416342412
0.112840466926070
0.116731517509728
0.120622568093385
0.124513618677043
0.132295719844358
0.143968871595331
0.157587548638132
0.173151750972763
0.188715953307393
0.204280155642023
0.221789883268482
0.241245136186770
0.260700389105058
0.284046692607004
0.311284046692607
0.338521400778210
0.365758754863813
0.394941634241245
0.428015564202335
0.464980544747082
0.505836575875486
0.550583657587549
0.597276264591440
0.647859922178988
0.704280155642023

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

0.764591439688716
0.828793774319066
0.898832684824903
0.966926070038911

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Table FC.9 — Processing band mapping table *pbm*(*k*)

k	*pbm(k)*	*k*	*pbm(k)*	*k*	*pbm(k)*	*k*	*pbm(k)*	*k*	*pbm(k)*	*k*	*pbm(k)*
0	0	50	37	100	46	150	51	200	54	250	57
1	1	51	38	101	46	151	51	201	54	251	57
2	2	52	38	102	46	152	51	202	54	252	57
3	3	53	38	103	46	153	51	203	54	253	57
4	4	54	38	104	46	154	51	204	54	254	57
5	5	55	39	105	46	155	51	205	55	255	57
6	6	56	39	106	47	156	51	206	55	256	57
7	7	57	39	107	47	157	51	207	55		
8	8	58	39	108	47	158	51	208	55		
9	9	59	39	109	47	159	51	209	55		
10	10	60	40	110	47	160	52	210	55		
11	11	61	40	111	47	161	52	211	55		
12	12	62	40	112	47	162	52	212	55		
13	13	63	40	113	47	163	52	213	55		
14	14	64	40	114	47	164	52	214	55		
15	15	65	41	115	48	165	52	215	55		
16	16	66	41	116	48	166	52	216	55		
17	17	67	41	117	48	167	52	217	55		
18	18	68	41	118	48	168	52	218	55		
19	19	69	41	119	48	169	52	219	55		
20	20	70	42	120	48	170	52	220	55		
21	21	71	42	121	48	171	52	221	55		
22	22	72	42	122	48	172	52	222	56		
23	23	73	42	123	48	173	52	223	56		
24	24	74	42	124	48	174	53	224	56		
25	25	75	42	125	49	175	53	225	56		
26	26	76	42	126	49	176	53	226	56		
27	27	77	43	127	49	177	53	227	56		
28	28	78	43	128	49	178	53	228	56		
29	29	79	43	129	49	179	53	229	56		
30	30	80	43	130	49	180	53	230	56		
31	31	81	43	131	49	181	53	231	56		
32	32	82	43	132	49	182	53	232	56		
33	33	83	43	133	49	183	53	233	56		
34	33	84	44	134	49	184	53	234	56		
35	33	85	44	135	49	185	53	235	56		
36	34	86	44	136	50	186	53	236	56		
37	34	87	44	137	50	187	53	237	56		
38	34	88	44	138	50	188	53	238	56		
39	35	89	44	139	50	189	54	239	56		
40	35	90	44	140	50	190	54	240	56		
41	35	91	45	141	50	191	54	241	57		
42	35	92	45	142	50	192	54	242	57		
43	36	93	45	143	50	193	54	243	57		
44	36	94	45	144	50	194	54	244	57		
45	36	95	45	145	50	195	54	245	57		
46	36	96	45	146	50	196	54	246	57		
47	37	97	45	147	50	197	54	247	57		
48	37	98	46	148	51	198	54	248	57		
49	37	99	46	149	51	199	54	249	57		

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table FC.10 — Equalizer parameters

Equalizer	P_f [Hz]	P_Q	P_g[dB]	g [dB]
$G_{EQ,1}$	12000	0.3	-2	1.0
$G_{EQ,2}$	12000	0.3	-3.5	1.0
$G_{EQ,3}$	200,1300, 600	0.3, 0.5, 1.0	-6.5, 1.8, 2.0	0.7
$G_{EQ,4}$	5000, 1100	1.0, 0.8	4.5, 1.8	-3.1
$G_{EQ,5}$	35	0.25	-1.3	1.0

Table FC.11 — Vertically corresponding channels: Each row lists channels which are considered to be above/below each other.

CH_L_000	CH_M_000	CH_U_000
CH_L_L045	CH_M_L030	CH_U_L030
CH_L_L045	CH_M_L030	CH_U_L045
CH_L_L045	CH_M_L045	CH_U_L030
CH_L_L045	CH_M_L045	CH_U_L045
CH_L_L045	CH_M_L060	CH_U_L030
CH_L_L045	CH_M_L060	CH_U_L045
CH_L_R045	CH_M_R030	CH_U_R030
CH_L_R045	CH_M_R030	CH_U_R045
CH_L_R045	CH_M_R045	CH_U_R030
CH_L_R045	CH_M_R045	CH_U_R045
CH_L_R045	CH_M_R060	CH_U_R030
CH_L_R045	CH_M_R060	CH_U_R045
CH_M_180	CH_U_180	
CH_M_L090	CH_U_L090	
CH_M_L110	CH_U_L110	
CH_M_L135	CH_U_L135	
CH_M_L090	CH_U_L110	
CH_M_L090	CH_U_L135	
CH_M_L110	CH_U_L090	
CH_M_L110	CH_U_L135	
CH_M_L135	CH_U_L090	
CH_M_L135	CH_U_L135	
CH_M_R090	CH_U_R090	
CH_M_R110	CH_U_R110	
CH_M_R135	CH_U_R135	
CH_M_R090	CH_U_R110	
CH_M_R090	CH_U_R135	
CH_M_R110	CH_U_R090	
CH_M_R110	CH_U_R135	
CH_M_R135	CH_U_R090	
CH_M_R135	CH_U_R135	

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

FC.3.5 Audio signal processing

FC.3.5.1 General

The audio processing block of the format converter obtains time domain audio samples for N_{in} channels from the core decoder and generates a downmixed time domain audio output signal consisting of N_{out} channels.

The processing takes as input

— the audio data decoded by the core decoder,

— the static downmix matrix $\mathbf{M}_{\text{DMX}}$ returned by the initialization of the format converter.

It returns an N_{out}-channel time domain output signal for the OutConf channel configuration signaled during the initialization of the format converter.

The format converter operates on contiguous, non-overlapping frames of length L = 256 time domain samples of the input audio signals and outputs one frame of L samples per processed input frame of length L. The algorithm performs a short time Fourier transform (STFT) of length *N* = 512 with 50% overlap, i.e. the overlap length as well as the hop size of the STFT is 256 time domain samples. The STFT domain processing takes place in *K*=256 frequency bins, which are partitioned into *PB*=58 processing bands.

FC.3.5.2 T/F-transform (STFT analysis)

As the first processing step the converter updates the STFT input buffer $\hat{y}^{\nu}_{ch,i}$ by one frame (*L*=256 samples) of the N_{in} channel time domain input signal $\left[\tilde{y}^{\nu}_{\text{ch},1} \cdots \tilde{y}^{\nu}_{\text{ch},N_{\text{in}}}\right] = \tilde{\mathbf{y}}^{\nu}_{\text{ch}}$,

$$\hat{y}^{\nu}_{ch,i} = \begin{cases} \tilde{y}^{\nu,F-1}_{ch,i} & \text{for} \quad 0 \le \nu < L, \quad 1 \le i \le N_{\text{in}} \\ \tilde{y}^{\nu-L,F}_{ch,i} & \text{for} \quad L \le \nu < N, \quad 1 \le i < N_{\text{in}} \end{cases},$$

where *F* denotes the frame index and $\tilde{y}^{\nu,F-1}_{ch,i} = 0$ for the first processing frame. An analysis window is applied and a DFT of length *N*=512 is calculated for each of the N_{in} signals in the windowed STFT input buffer:

$$y^{k}_{ch,i} = \sum_{\nu=0}^{N-1} w[\nu] \cdot \hat{y}^{\nu}_{ch,i} \cdot e^{-2\pi jk\nu/N} \quad \text{for } 0 \le k < K,\ 1 \le i \le N_{in}$$

with $w[\nu] = \sin\left(\left(\nu + 0.5\right)\pi / N\right)$.

FC.3.5.2 Intermediate downmix signals

Intermediate downmix signals and corresponding energies are calculated according to:

$$\breve{z}^{k}_{ch,o} = \sum_{i=1}^{N_{in}} y^{k}_{ch,i} M^{k}_{\text{DMX},o,i} \quad \text{for} \quad 1 \le o \le N_{out}, \quad 0 \le k < K$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

$$\widehat{Z}^{pb}_{ch,o} = \sum_{i=1}^{N_{in}} \sum_{\substack{k,\\ pbm(k)\\ =pb}} \left|y^{k}_{ch,i}\right|^2 \left(M^{k}_{\mathrm{DMX},o,i}\right)^2 \quad \text{for} \quad 1 \le o \le N_{out},\ 0 \le pb < PB$$

Similarly, the energies of the intermediate downmix signals are derived in the processing bands as

$$\widehat{\widehat{Z}}^{pb}_{ch,o} = \sum_{\substack{k,\\ pbm(k)\\ =pb}} \left|\overset{\frown}{z}^{k}_{ch,o}\right|^2 \quad \text{for} \quad 1 \le o \le N_{out},\ 0 \le pb < PB\,.$$

FC.3.5.3 Final frequency domain downmix signals

The final downmix signals are obtained in the STFT domain according to

$$z^{k}_{ch,o} = \overset{\frown}{z}^{k}_{ch,o} EQ^{k}_{ch,o} \quad \text{for} \quad 1 \le o \le N_{out},\ 0 \le k < K$$

with

$$EQ^{k}_{ch,o} = \begin{cases} 1 & \text{if} \quad \text{passiveDownmixFlag} == 1 \\ \min\left(10^{0.4}, \max\left(10^{-0.5}, \widehat{EQ}^{k}_{ch,o}\right)\right) & \text{else} \end{cases}$$

and

$$\widehat{EQ}^{k}_{ch,o} = \sqrt{\frac{\overline{\widehat{Z}}^{pb}_{ch,o}}{\mathrm{eps} + \overline{\widehat{\widehat{Z}}}^{pb}_{ch,o}}} \quad \text{where} \quad pb = pbm(k)\,,$$

$$\overline{\widehat{Z}}^{pb,F}_{ch,o} = \alpha \widehat{Z}^{pb,F}_{ch,o} + (1-\alpha)\overline{\widehat{Z}}^{pb,F-1}_{ch,o} \quad \text{and} \quad \overline{\widehat{\widehat{Z}}}^{pb,F}_{ch,o} = \alpha \widehat{\widehat{Z}}^{pb,F}_{ch,o} + (1-\alpha)\overline{\widehat{\widehat{Z}}}^{pb,F-1}_{ch,o}\,.$$

$\overline{\widehat{Z}}^{pb,F-1}_{ch,o}$ as well as $\overline{\widehat{\widehat{Z}}}^{pb,F-1}_{ch,o}$ shall be initialized to zero for the first processing frame.

FC.3.5.4 F/T-transform (STFT synthesis)

As last processing step per processed frame of 256 samples, the downmix signal is transformed to the time domain by application of an inverse DFT, windowing and overlap-add update, yielding *L* time domain output samples $\tilde{z}^{\nu}_{ch,o}$ per output channel. For the current frame (frame index *F*) the operations read:

$$\hat{z}^{\nu,F}_{ch,o} = w[\nu] \cdot \frac{1}{N} \sum_{k=0}^{N-1} z^{k,compl}_{ch,o} \cdot e^{2\pi jk\nu/N} \quad \text{for } 0 \le \nu < N,\ 1 \le o \le N_{out}\,,$$

with $z^{k,compl}_{ch,o} = \begin{cases} z^{k}_{ch,o} & 0 \le k < K \\ conj(z^{N-k}_{ch,o}) & K \le k < N \end{cases}$, where $conj(z)$ denotes the complex conjugate of z.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

$\tilde{z}_{ch,o}^{\nu,F} = \hat{z}_{ch,o}^{\nu,F} + \hat{z}_{ch,o}^{\nu+L,F-1}$ for $0 \le \nu < L$, $1 \le o \le N_{out}$,

where $\hat{z}_{ch,o}^{\nu,F-1}$ shall be initialized with zeros for the first processing frame.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Add a new clause FC5, "Low Complexity Immersive Loudspeaker Rendering/Format Conversion" with the following text:

FC5 Low Complexity Immersive Loudspeaker Rendering/Format Conversion

FC5.1 Description

For the 5.0 and 5.1 channel output layouts, immersive loudspeaker rendering shall be chosen to provide overhead sound images using surround channel loudspeakers depending on the immersiveDownmixFlag in downmixConfig(). The immersive loudspeaker renderer is a downmixer that converts multichannel signals from transmitted channel configurations with N_{in} channels to desired reproduction format of either 5.1 or 5.0 system. It has a switching scheme between 3D rendering and 2D rendering using different elevation rendering for height input channels depending on the transmitted bitstream **rendering3DType** to provide overhead sound image properly. It is thus also called 'immersive format converter'. The system consists of two major building blocks:

— An initialization algorithm that takes into account static parameters like the input and output format,

— a signal adaptive downmixing process that operates in a subband domain with a switching scheme according to the transmitted flag **rendering3DType**.

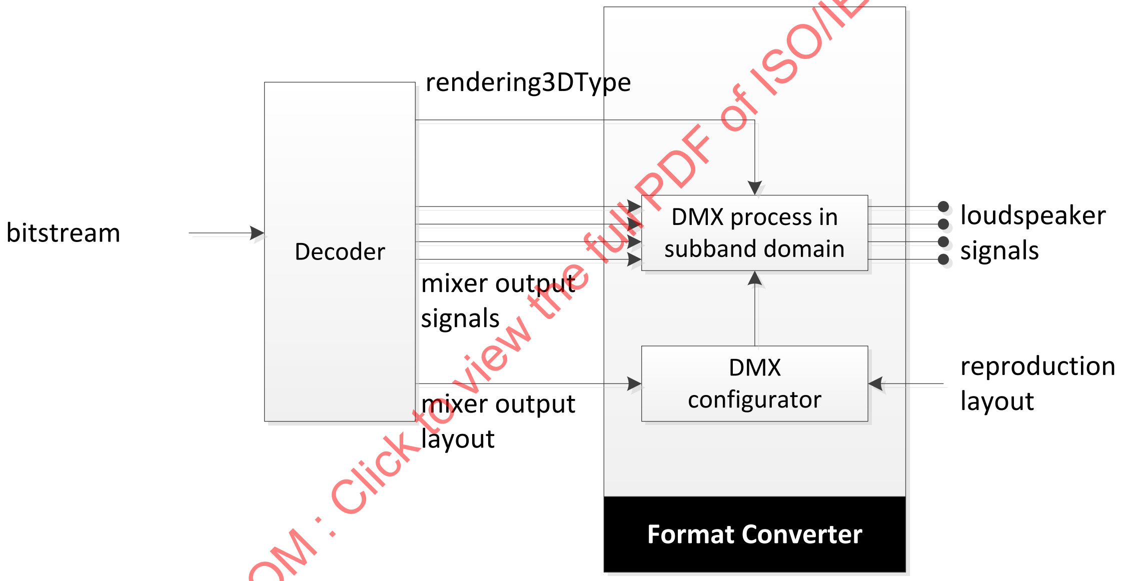


Figure FC5.1 — Main building blocks of the immersive format converter

In the initialization phase the format converter automatically generates optimized downmixing parameters (like the downmixing matrix) for the given combination of input and output formats: It applies an algorithm that selects for each input loudspeaker the most appropriate mapping rule from a list of rules that has been designed to incorporate psychoacoustic considerations. Each rule describes the mapping from one input channel to one or several output loudspeaker channels.

Input channels are

— either mapped to a single output channel,

— or panned to two output channels,

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

— or (in case of the 'Voice of God' channel) distributed over a larger number of output cannels.

— or (in case of the 'elevation rendering') panned to multiple output channels with different panning coefficients over frequency.

The optimal mapping for each input channel is selected depending on the list of output loudspeakers that are available in the desired output format. Each mapping defines downmix gains for the input channel under consideration as well as potentially also an equalizer that is applied to the input channel under consideration.

Output setups with non-standard loudspeaker positions can be signaled to the system by providing the azimuth and elevation deviations from a regular loudspeaker setup. Further, distance variations of the desired target loudspeaker positions are taken into account.

For each frame, a bit called rendering3DType is decoded by the decoder and passed to the immersive format converter. The rendering3DType indicates whether the sound scene is appropriate for the 3D rendering or 2D rendering over either 5.0 or 5.1 channel layout. For the height input channels, the immersive format converter uses the "spatial elevation rendering" for 3D rendering when the rendering3DType is TRUE and "timbral elevation rendering" for the 2D rendering when the rendering3DType is FALSE. For the non-height input channels, non-elevation rendering uses the same downmix coefficients regardless of the rendering3DType.

The actual downmixing of the audio signals is performed on a short time Fourier transform (STFT) representation of the signals. The energy-preserving algorithm avoids signal deteriorations like comb-filtering, coloration, or modulation artifacts.

FC5.2 Syntax

The FormatConverterFrame defines the proper rendering type for the immersive format converter.

Table FC5.1 — Syntax of FormatConverterFrame()

Syntax	No. of bits	Mnemonic
FormatConverterFrame()		
{		
rendering3DType;	**1**	**uimsbf**
}		

The flag **rendering3DType** is created at the encoder based on the audio scene. When the audio scene is wideband and highly decorrelated at a frame, the flag **rendering3DType** becomes false and rendering is done by the secondary downmix matrix $\mathbf{M}_{\text{DMX2}}$. In all other cases, the flag becomes true and rendering is done by the primary downmix matrix $\mathbf{M}_{\text{DMX}}$, which provides elevated sound images.

FC5.3 Definitions

FC5.3.1 General remarks

Audio signals that are fed into the format converter are referred to as *input signals* in the following. Audio signals that are the result of the format conversion process are referred to as *output signals*. Note that the audio input signals of the format converter are audio output signals of the core decoder.

Vectors and matrices are denoted by bold-faced symbols. Vector elements or matrix elements are denoted as italic variables supplemented by indices indicating the row/column of the vector/matrix element in the vector/matrix, e.g. $[y_1 \cdots y_a \cdots y_N] = \mathbf{y}$ denotes a vector and its elements. Similarly, $M_{a,b}$ denotes the element in the a th row and b th column of a matrix $\mathbf{M}$.

FC5.3.2 Variable definitions

N_{in}	Number of channels in the input channel configuration.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

N_{out}	Number of channels in the output channel configuration.
$\mathbf{M}_{\text{DMX}}$	Primary downmix matrix containing real-valued non-negative downmix coefficients (downmix gains) for the non-elevation rendering and spatial elevation rendering for 3D rendering over 2D layout. $\mathbf{M}_{\text{DMX}}$ is basically of dimension (N_{out} x N_{in}) but possibly increased to (N_{out} x (N_{in}+5)) depending on the input and output layouts. See how the dimension is changed in FC5.4.2.3.
$\mathbf{M}_{\text{DMX2}}$	Secondary downmix matrix containing real-valued non-negative downmix coefficients (downmix gains) for the non-elevation rendering and timbral elevation rendering for 2D rendering over 2D layout. The downmix coefficients for the horizontal input channels are identical to those in $\mathbf{M}_{\text{DMX}}$. The $\mathbf{M}_{\text{DMX}}$ is dimension of (N_{out} x N_{in}) but possibly increased to (N_{out} x (N_{in}+5)) depending on the input and output layouts. See how the dimension is changed in FC5.4.2.3.
$\mathbf{G}_{\text{EQ}}$	Matrix consisting of gain values per processing band determining frequency responses of equalizing filters for all rendering mapping. $\mathbf{G}_{\text{EQ},1\sim5}$ are used for the non-elevation rendering and timbral elevation rendering, $\mathbf{G}_{\text{EQ},7\sim14}$ are used for spatial elevation rendering, $\mathbf{G}_{\text{EQ},15\sim20}$ are used for spatial coloration filter, and $\mathbf{G}_{\text{EQ},21\sim}$ are used for modified EQ for randomized setup in FC5.4.1.6.5 and coloration filter in FC5.4.1.6.7.6.
$\mathbf{I}_{\text{EQ}}$	Vector signalling which equalizer filters to apply to the input channels (if any).
L	Frame length measured in time domain audio samples.
ν	Time domain sample index.
F	Frame index (frame number).
PB	Number of processing bands, *PB*=58.
pb	Processing band index $(0 \le pb < PB)$.
N	DFT length
K	Number of STFT frequency bins, K = 257.
k	STFT frequency bin index ($0 \le k < K$).
α	Filter parameter, α = 0.0435.
A, B	Channel indices.
eps	Numerical constant, $\text{eps} = 10^{-35}$.
rendering3DType	Flag from the bitstream identifying the rendering type for the elevation rendering. True for the general audio scene and false for the highly decorrelated wideband scene (e.g. applause). Accordingly, the primary downmix matrix $\mathbf{M}_{\text{DMX}}$ is chosen when the **rendering3DType** is TRUE and

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

	the secondary downmnix matrix $\mathbf{M}_{DMX2}$ is chosen when the **rendering3DType** is FALSE in the immersive format converter.
i_{in}	Label of the input channel to be rendered by immersive format converter (e.g. CH_U_000)
$G_{vH0,1\sim6}(i_{in})$	Spatial elevation panning coefficients for the input channel (i_{in}) for the 2.8~10kHz in order to provide overhead image. Note that the coefficient is always normalized to preserve the input power after mixing. The number index represents output channels as shown in Table FC5.10.
$G_{vL0,1\sim6}(i_{in})$	Spatial elevation panning coefficients for the input channel (i_{in}) for below 2.8 kHz and above 10kHz. Note that the coefficient is always normalized to reserve the input power after mixing. The number index represents output channels as shown in Table FC5.10.
COLOR_A_B	Tone coloration filter to the output at the azimuth of ±B degree from the input at the azimuth of ±A degree based on the ratio between HRTF at A and HRTF at B. It is determined by a frequency dependent dynamic cue which represents speaker-to-listener orientation.

FC5.4 Processing

FC5.4.1 Initialization of the format converter

FC5.4.1.1 General description of the initialization

The initialization of the format converter is carried out before processing of the audio samples delivered by the core decoder takes place.

The initialization takes into account as input parameters

— The sampling rate of the audio data to process.

— The channel configuration of the audio data to process with the format converter (number and geometric positions of input channels).

— The channel configuration of the desired output format (number and geometric positions of output channels).

— Optional: Parameters signaling the deviation of the output loudspeaker positions from a standard loudspeaker setup (random setup functionality).

It returns

— The primary frequency dependent downmix matrix $\mathbf{M}_{DMX}$ that is applied in the audio signal processing of the format converter when the rendering3DType is TRUE. Note that the $\mathbf{M}_{DMX}$ is independent variable in the Format Converter and shall not be taken into account in the core decoding process in 5.5.3.1.2.

— The secondary frequency dependent downmix matrix $\mathbf{M}_{DMX2}$ that is applied in the audio signal processing of the format converter when the rendering3DType is FALSE. The $\mathbf{M}_{DMX2}$ is identical to $\mathbf{M}_{DMX}$ if there is no 'height' input channel or 'spatial elevation rendering' is not possible.

The input parameters to the initialization algorithm are listed in Table FC5.2.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Table FC5.2 — Format converter initialization input parameters

	Input format: number of channels and nominal channel setup geometry
	Output format: number of channels and nominal channel setup geometry
f_s	Sampling frequency in Hertz.
$r_{\mathrm{azi},A}$	For each output channel *A*, an azimuth angle is specified, determining the deviation from the standard format loudspeaker azimuth.
$r_{\mathrm{ele},A}$	For each output channel *A*, an elevation angle is specified, determining the deviation from the standard format loudspeaker elevation.

Table FC5.3 lists the output parameters that are derived during the initialization of the format converter.

Table FC5.3 — Format converter initialization output parameters

$\mathbf{M}_{\mathrm{DMX}}$	Primary Downmix matrix [linear gains] for spatial elevation rendering (for rendering3DType == 1)
$\mathbf{M}_{\mathrm{DMX2}}$	Secondary Downmix matrix [linear gains] for timbral elevation rendering (for rendering3DType == 0)

Note that the $\mathbf{M}_{\mathrm{DMX1}}$ and $\mathbf{M}_{\mathrm{DMX2}}$ include the same input-output downmix matrix for non-elevation rendering input channels.

FC5.4.1.2 Assignment of format converter channel labels to input/output format channels

The format converter initialization is based on a system of rules that are defined in terms of *format converter channel labels*, see Table FC5.12. To allow the application of the initialization rules, the channel labels have to be assigned to the channels of the input and output formats. Each format converter channel label is associated with a segment of the surface of the unit sphere, as defined in Table FC5.12. The segments are designed non-overlapping.

The assignment of channel labels to channels is done by geometrically matching the segments to the position data associated with the channels of the input and output formats. The azimuth and elevation angles in degrees of the position data associated with the channels shall be rounded towards the nearest integer number before performing the channel label assignment. Note that the *nominal* channel positions shall be applied in the following matching to channel label sectors, i.e. the azimuth and elevation angles *without* taking into account potential angle deviations signalled in $r_{\mathrm{azi},A}$ and/or $r_{\mathrm{ele},A}$.

For each channel that is not an LFE (low-frequency effects) channel:

If the nominal position of the current channel, defined by its azimuth angle and elevation angle, is within or on the border of one of the segments defined in Table FC5.12 then:

Assign the corresponding channel label (e.g. CH_M_L030) associated with the matching segment.

Add the angle differences between the nominal position of the current channel and the nominal position associated with the matching segment (i.e. the angles in the second and third column of Table FC5.2) to the angle deviations stored in $r_{\mathrm{azi},A}$ and $r_{\mathrm{ele},A}$.

Else (i.e. no matching sector found), then:

Assign the CH_EMPTY label.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

If an input or output format contains exactly one LFE channel, then the label CH_LFE1 shall be assigned to this channel.

If an input or output format contains exactly two LFE channels, then the labels CH_LFE1 and CH_LFE2 shall be assigned to the two LFE channels in the order that minimizes the maximum azimuth distance from the channels to the assigned CH_LFE1 and CH_LFE2 nominal azimuth positions.

If an input or output format contains more than 2 LFE channels, then those 2 LFE channels out of the considered setup shall be selected that minimize the maximum azimuth distance to the CH_LFE1 and CH_LFE2 nominal azimuth positions. The labels CH_LFE1 and CH_LFE2 shall be assigned as in the case of two LFE channels. The remaining LFE channels shall not be considered further in the calculation of downmix coefficients, i.e. the corresponding lines/columns of the downmix matrix shall remain filled with zeros.

FC5.4.1.3 Handling for unknown input channels

If the label CH_EMPTY is assigned to an input channel, this channel shall be considered unknown to the rules-based initialization and the downmix coefficients for mapping this input channel to the output channels shall be derived as specified in FC.3.4.6.7.

FC5.4.1.4 Handling for unknown output formats

If the output format contains at least one channel with the label CH_EMPTY assigned to it, or if at least one channel label is assigned to more than one channel of the output format, the output format shall be considered unknown and the derivation of the downmixing coefficients shall be carried out as specified in FC5.4.1.6.9. The rules-based derivation of downmix coefficients shall not be applied for unknown output formats.

FC5.4.1.5 Handling of deviations from standard loudspeaker positions

If the below conditions are not met, the rules-based initialization is considered to have failed, the output format shall be considered to be unknown, and the downmixing gains shall be obtained as defined in FC5.4.1.6.9.

The absolute values of $r_{\text{azi},A}$ and $r_{\text{ele},A}$ shall not exceed 35 and 55 degrees, respectively. The minimum angle between any loudspeaker pair (without LFE channels) shall not be smaller than 15 degrees.

The values of $r_{\text{azi},A}$ shall be such that the ordering by azimuth angles of the horizontal loudspeakers does not change. Likewise, the ordering of the height and low loudspeakers shall not change.

The values of $r_{\text{ele},A}$ shall be such that the ordering by elevation angles of loudspeakers which are (approximately) above/below each other does not change. To verify this, the following procedure is applied:

For each row of Table FC5.18 which contains two or three channels of the output format, do:

— Order the channels by elevation without randomization

— Order the channels by elevation with considering randomization

— If the two orderings differ, return an initialization error

If the below conditions are not met, converter initialization is considered to have failed, and an error shall be returned.

FC5.4.1.6 Rules-Based Initialization algorithm

FC5.4.1.6.1 General

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

The rules-based initialization algorithm is defined in the following. It shall not be applied if the output format is considered unknown as defined above. Note that the following description makes use of the intermediate parameters defined in Table FC5.4 only for clarity reasons. An implementation may omit the introduction of these intermediate parameters.

Table FC5.4 — Format converter initialization intermediate parameters

$\mathbf{S}, \mathbf{S}^P, \mathbf{S}^S$	Vector of converter source channels [input channel indices]
$\mathbf{D}, \mathbf{D}^P, \mathbf{D}^S$	Vector of converter destination channels [output channel indices]
$\mathbf{G}, \mathbf{G}^P, \mathbf{G}^S$	Vector of converter gains [linear]
$\mathbf{E}, \mathbf{E}^P, \mathbf{E}^S$	Vector of converter EQ indices
$\mathbf{G}_{\mathrm{EQ}}$	Matrix containing equalizer gain values for all EQ indices and processing bands

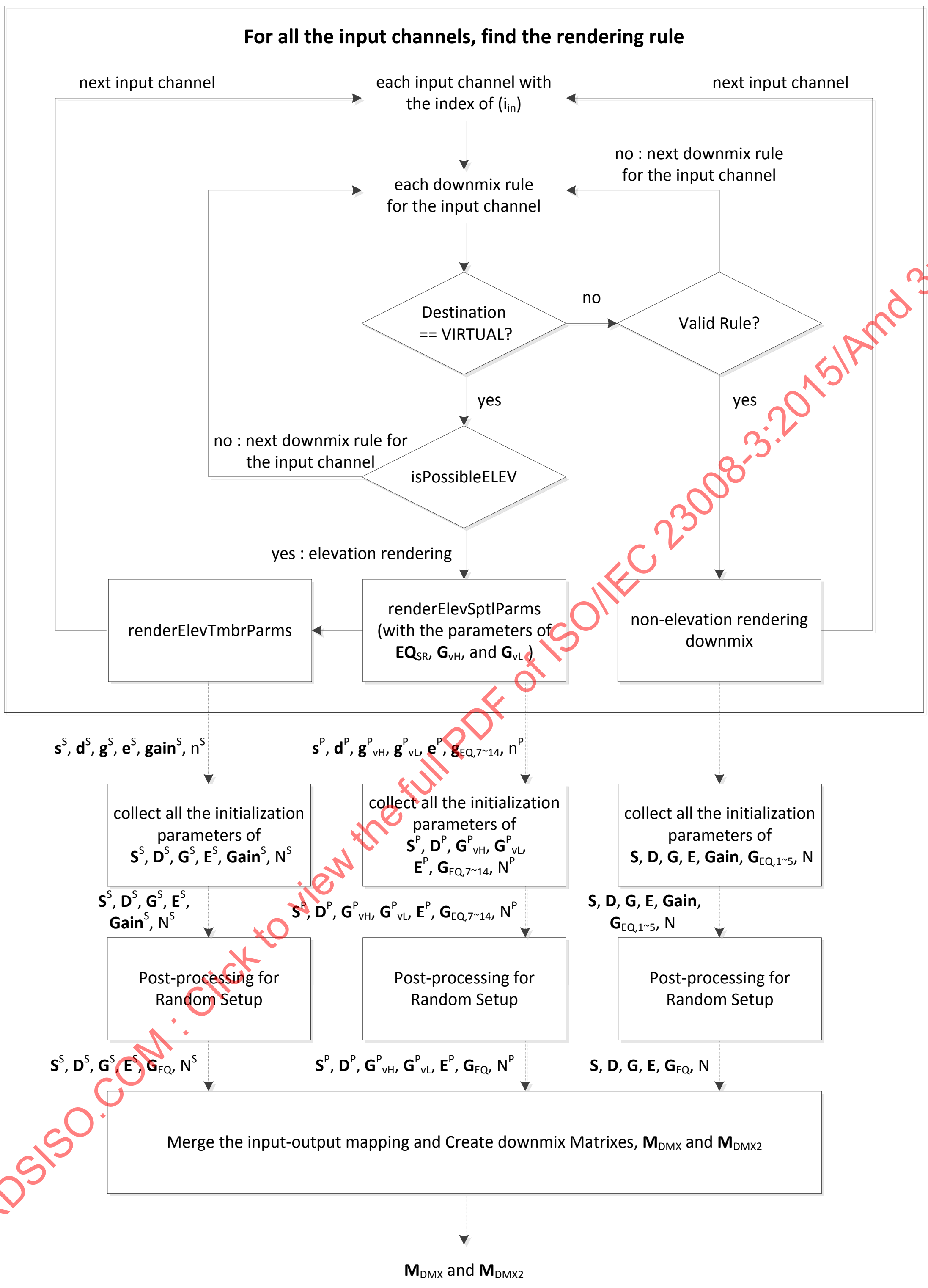
* The superscript S/P is the discriminator for the elevation rendering type. Those with P are initialized to be used for the ‘spatial elevation rendering’ and used to create the primary downmix matrix $\mathbf{M}_{\mathrm{DMX}}$, those with S are for the ‘timbral elevation rendering’ and used to create the secondary downmix matrix, and those without superscript are for the non-elevation rendering and used to create both the primary and secondary downmix matrixes.

The intermediate parameters describe the dowmixing parameters in a mapping-oriented way, i.e. as sets of parameters S_i, D_i, G_i, E_i per mapping *i.*

The format converter initialization output parameters are derived as described in the following figure,

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved


For all the input channels, find the rendering rule
next input channel
each input channel with the index of (i_in)
next input channel
no : next downmix rule for the input channel
each downmix rule for the input channel
Destination == VIRTUAL?
no
Valid Rule?
yes
yes
no : next downmix rule for the input channel
isPossibleELEV
yes : elevation rendering
renderElevTmbrParms
renderElevSptlParms (with the parameters of EQ_SR, G_vH, and G_vL)
non-elevation rendering downmix
s^S, d^S, g^S, e^S, gain^S, n^S
s^P, d^P, g^P_vH, g^P_vL, e^P, g_EQ,7~14, n^P
collect all the initialization parameters of S^S, D^S, G^S, E^S, Gain^S, N^S
collect all the initialization parameters of S^P, D^P, G^P_vH, G^P_vL, E^P, G_EQ,7~14, N^P
collect all the initialization parameters of S, D, G, E, Gain, G_EQ,1~5, N
S^S, D^S, G^S, E^S, Gain^S, N^S
S^P, D^P, G^P_vH, G^P_vL, E^P, G_EQ,7~14, N^P
S, D, G, E, Gain, G_EQ,1~5, N
Post-processing for Random Setup
Post-processing for Random Setup
Post-processing for Random Setup
S^S, D^S, G^S, E^S, G_EQ, N^S
S^P, D^P, G^P_vH, G^P_vL, E^P, G_EQ, N^P
S, D, G, E, G_EQ, N
Merge the input-output mapping and Create downmix Matrixes, M_DMX and M_DMX2
M_DMX and M_DMX2

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

Figure FC5.2 — Rule based downmix initialization flow

, with following steps for each channel:

1. If the destination of the downmix rule for the input channel (i_{in}) is VIRTUAL, isPossibleElev decides whether the input channel should be rendered by the elevation rendering defined in FC5.4.1.6.7.
   A. If isPossibleElev returns TRUE,

i. A set of parameters of $\mathbf{s}^\mathrm{P}$, $\mathbf{d}^\mathrm{P}$, $\mathbf{g}^\mathrm{P}_\mathrm{H}$, $\mathbf{g}^\mathrm{P}_\mathrm{L}$, $\mathbf{e}^\mathrm{P}$, $\mathbf{G}_{\mathrm{EQ,eq(i_{in})}}$, and n^P and another set of parameters of $\mathbf{s}^\mathrm{S}$, $\mathbf{d}^\mathrm{S}$, $\mathbf{g}^\mathrm{S}$, $\mathbf{e}^\mathrm{S}$, $\mathbf{gain}^\mathrm{S}$, and n^S are initialized by the renderElevParms and renderTmbrParms, respectively. Note that the parameters of n^P and n^S indicates the number of output loudspeaker required for the input channel ($\mathrm{i_{in}}$), $\mathbf{s}^\mathrm{P}$, $\mathbf{d}^\mathrm{P}$, $\mathbf{g}^\mathrm{P}_\mathrm{H}$, $\mathbf{g}^\mathrm{P}_\mathrm{L}$, and $\mathbf{e}^\mathrm{P}$ are n^P column vectors, $\mathbf{s}^\mathrm{S}$, $\mathbf{d}^\mathrm{S}$, $\mathbf{g}^\mathrm{S}$, $\mathbf{e}^\mathrm{S}$, and $\mathbf{gain}^\mathrm{S}$ are n^S column vectors, $\mathbf{G}_{\mathrm{EQ,eq(i_{in})}}$ is a 58 row vector representing the EQ coefficients in 58 processing bands for the input channel ($\mathrm{i_{in}}$).

ii. The initialization parameters are collected among the group of parameters, each primary or secondary, and the downmix rules for next input channel are investigated until all the input channel mapping is found.

B. If isPossibleElev returns FALSE, the current downmix rule is ignored and the next downmix rule is investigated

2. If the destination is not VIRTUAL, the downmix rule is investigated whether the rule is valid checking the output layout includes all the channels in the destination column.

A. If the downmix rule is valid,

i. A set of parameters is initialized by the non-elevation rendering initialization and added directly to the $\mathbf{S}$, $\mathbf{D}$, $\mathbf{G}$, $\mathbf{E}$, and $\mathbf{Gain}$ as specified in FC5.4.1.6.3 and to the $\mathbf{G}_\mathrm{EQ}$ as specified in FC5.4.1.6.4.

ii. The downmix rules for next input channel are investigated until all the input channel mappings are found.

B. If the downmix rule is invalid, the current downmix rule is ignored and the next downmix rule is investigated.

3. After collecting all the initializations parameters for the 'spatial elevation rendering,' 'timbral elevation rendering,' and 'non-elevation rendering,' post-processing for random setup shall be applied.

4. Create the primary downmix matrix combining the 'spatial elevation rendering' and 'non-elevation rendering' parameters and the secondary downmix matrix combining the 'timbral elevation rendering' and 'non-elevation rendering' parameters.

FC5.4.1.6.2 Random setups Pre-Processing

Random output loudspeaker setups, i.e. output setups that contain loudspeakers at positions deviating from the positions defined for the desired output format are signalled by specifying the loudspeaker position deviation angles as input parameters $r_{\mathrm{azi},A}$ and $r_{\mathrm{ele},A}$. The angle deviations are taken into account as a pre-processing step:

Modify in Table FC5.12 the channels' azimuth and elevation angles by adding $r_{\mathrm{azi},A}$ and $r_{\mathrm{ele},A}$ to the corresponding channels' azimuth and elevation angles.

FC5.4.1.6.3 Derivation of input channel/output channel mapping parameters:

The parameters vectors $\mathbf{S}$, $\mathbf{D}$, $\mathbf{G}$, $\mathbf{E}$ define the mapping of input channels to output channels. For each mapping *i* from an input channel to an output channel with non-zero downmix gain they define the downmix gain as well as an equalizer index that indicates which equalizer curve has to be applied to the input channel under consideration in mapping *i*.

The elements of the parameter vectors $\mathbf{S}$, $\mathbf{D}$, $\mathbf{G}$, $\mathbf{E}$ are derived by the following algorithm:

Initialize the mapping counter *i*: $i=1$, $i^\mathrm{P}=1$, $i^\mathrm{S}=1$;

Initialize the EQ counter e: $e=21$ (EQ slots for e from 1 to 5 are occupied by FC5.4.1.6.4, that for e from 7 to 14 are occupied by Table FC5.9, and that for e from 15 to 20 are occupied by Table FC5.6. The EQ counter e will be shared in FC5.4.1.6.5 and FC5.4.1.6.7.6 in an incremental way)

For each input channel, ignoring channels with label CH_EMPTY assigned to them:

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

If the input channel also exists in the output format (e.g. input channel under consideration is CH_M_R030 and channel CH_M_R030 exists in the output format), then:

S_i = index of source channel in input (***Example:*** *Example: channel CH_M_R030 in ChannelConfiguration 6 is at second place according to Table FC5.13, i.e. has index 2 in this forma)*

D_i = index of same channel in output

G_i = 1.0

E_i = 0

$i = i + 1$

Else (i.e. if the input channel does not exist in the output format)

search the first entry of this channel in the **Source** column of Table FC5.14, for which the channels in the corresponding row of the **Destination** column exist. The **VIRTUAL** destination shall be considered valid if the **isPossibleElev** returns **TRUE**, which indicates the output format contains required channels for the elevation rendering of the input channel. The **isPossibleElev**, **renderElevSptlParms,** and **renderElevTmbrParms** are defined in FC5.4.1.6.7. The ALL_U destination shall be considered valid (i.e. the relevant output channels exist) if the output format contains at least one "CH_U_" channel. The ALL_M destination shall be considered valid (i.e. the relevant output channels exist) if the output format contains at least one "CH_M_" channel. If for no entry in Table FC5.14. corresponding to the input channel the channels in the **Destination** column exist, the rules-based initialization shall terminate and the downmix gains shall be derived according to FC5.4.1.6.9.

If **Destination** column contains **VIRTUAL**, then:

```
[ $\mathbf{s}^P$, $\mathbf{d}^P$, $\mathbf{g}^P_H$, $\mathbf{g}^P_L$, $\mathbf{e}^P$, $n^P$]= renderElevSptlParms (specified in FC5.4.1.6.7.4)
for $n$ = 1 to $n^P$
    $m$          $= i^P$
    $S^P_m$      $= s^P_n(\mathrm{i_{in}})$
    $D^P_m$      $= d^P_n(\mathrm{i_{in}})$
    $G^P_{mH}$   $= g^P_{nH}(\mathrm{i_{in}})$
    $G^P_{mL}$   $= g^P_{nH}(\mathrm{i_{in}})$
    $E^P_m$      $= e^P_n(\mathrm{i_{in}})$
    $C^P_{mi}$   $= (\mathrm{i_{in}})$
    $C^P_{mo}$   = label of the output channel $d^P_n(\mathrm{i_{in}})$
    $i^P$        $= i^P + 1$
[ $\mathbf{s}^S$, $\mathbf{d}^S$, $\mathbf{g}^S$, $\mathbf{e}^S$, $\mathbf{gain}$, $n^S$ ]= renderElevSptlParms (described in FC5.4.1.6.7.5)
for $n$ = 1 to $n^S$
    $m$          $= i^S$
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

S^{S}_{m} $= s^{S}_{n}(\mathrm{i}_{\mathrm{in}})$

D^{S}_{m} $= d^{S}_{n}(\mathrm{i}_{\mathrm{in}})$

G^{S}_{m} $= g^{S}_{n}(\mathrm{i}_{\mathrm{in}})$

E^{S}_{m} $= e^{S}_{n}(\mathrm{i}_{\mathrm{in}})$

C^{S}_{mi} $= (\mathrm{i}_{\mathrm{in}})$

C^{S}_{mo} = label of the output channel $d^{P}_{n}(\mathrm{i}_{\mathrm{in}})$

$Gain^{S}_{m}$ $= gain_{n}$

i^{S} $=i^{S} + 1$

where i_{in} is the input channel label.

Else, if **Destination** column contains ALL_U, then:

For each output channel x with “CH_U_” in its name, do:

S_i = index of source channel in input

D_i = index of channel x in output

G_i = (value of Gain column) / sqrt(number of “CH_U_” output channels)

E_i = value of EQ column

$Gain_i$ = (value of Gain column)

$i = i + 1$

Else if **Destination** column contains ALL_M, then:

For each output channel x with “CH_M_” in its name, do:

S_i = index of source channel in input

D_i = index of channel x in output

G_i = (value of Gain column) / sqrt(number of “CH_M_” output channels)

E_i = value of EQ column

$Gain_i$ = (value of Gain column)

$i = i + 1$

Else If there is one channel in the **Destination** column, then:

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

S_i = index of source channel in input

D_i = index of destination channel in output

G_i = value of Gain column

E_i = value of EQ column

$Gain_i$ = (value of Gain column)

$i = i + 1$

Else (two channels in **Destination** column)

S_i = index of source channel in input

D_i = index of first destination channel in output

G_i = (value of Gain column) * g_1

E_i = value of EQ column

$Gain_i$ = (value of Gain column)

$i = i + 1$

S_i = S_{i-1}

D_i = index of second destination channel in output

G_i = (value of Gain column) * g_2

E_i = E_{i-1}

$Gain_i$ = (value of Gain column)

$i = i + 1$

The gains g_1 and g_2 are computed by applying tangent law amplitude panning in the following way:

— Unwrap source destination channel azimuth angles to be positive.

— The azimuth angles of the destination channels are α_1 and α_2 (see Table FC5.12).

— The azimuth angle of the source channel (= panning target) is α_{src}.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

— $\alpha_0 = \dfrac{|\alpha_1 - \alpha_2|}{2}$

— $\alpha_{\text{center}} = \dfrac{\alpha_1 + \alpha_2}{2}$

— $\alpha = (\alpha_{\text{center}} - \alpha_{\text{src}}) \cdot \text{sgn}(\alpha_2 - \alpha_1)$

— $g_1 = \dfrac{g}{\sqrt{1+g^2}}, \quad g_2 = \dfrac{1}{\sqrt{1+g^2}} \quad \text{with} \quad g = \dfrac{\tan\alpha_0 - \tan\alpha + 10^{-10}}{\tan\alpha_0 + \tan\alpha + 10^{-10}}$

FC5.4.1.6.4 Derivation of equalizer gains $\mathbf{G}_{\text{EQ}}$:

$\mathbf{G}_{\text{EQ}}$ consists of gain values per processing band *pb* and equalizer index *e*. The 5 predefined equalizers are combinations of different peak filters for $1 \le e \le 5$. Each equalizer is a serial cascade of one or more peak filters and a gain:

$$G_{EQ,e}^{pb} = 10^{g/20} \prod_{n=1}^{N} peak\left(band(pb)\frac{f_s}{2}, P_{f,n}, P_{Q,n}, P_{g,n} \right)$$

where *band*(*pb*) is the normalized center frequency of processing band *pb*, specified in Table FC5.15, f_s is the sampling frequency, and function *peak()* is for negative G

$$\text{peak}(b, f, Q, \text{G}) = \sqrt{\frac{b^4 + \left(\frac{1}{Q^2} - 2\right) f^2 b^2 + f^4}{b^4 + \left(\frac{10^{\frac{-G}{10}}}{Q^2} - 2\right) f^2 b^2 + f^4}}$$

and otherwise

$$\text{peak}(b, f, Q, \text{G}) = \sqrt{\frac{b^4 + \left(\frac{10^{\frac{G}{10}}}{Q^2} - 2\right) f^2 b^2 + f^4}{b^4 + \left(\frac{1}{Q^2} - 2\right) f^2 b^2 + f^4}}$$

The parameters for the equalizers are specified in Table FC5.17.

FC5.4.1.6.5 Post-Processing for Random Setups

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Once the output parameters are computed, they are modified related to the specific random azimuth and elevations angles. This step has only to be carried out, if not all $r_{\mathrm{ele},A}$ are zero. Definition of the post-processing algorithm for non-elevation rendering:

For each element *i* in D_i , do:

```
if output channel with index $D_i$ is a horizontal channel by definition (i.e. output channel label contains the label '_M_'), and
    if this output channel is now a height channel (elevation in range 0..60 degrees), and
        if input channel with index $S_i$ is a height channel (i.e. label contains '_U_'), then
            — $h$ = min(elevation of randomized output channel, 35) / 35
            — $G_{comp} = h \cdot \frac{1}{Gain_i} + (1 - h)$
            — Apply compensation gain to DMX gain: $G_i = G_i \cdot G_{\mathrm{comp}}$
            — Define new equalizer $\mathbf{G}_{\mathrm{EQ},e}$ with the index $e$, where $G_{EQ,e}^{pb} = h + (1-h) \cdot \mathrm{G}_{EQ,E_i}^{pb}$
            — $E_i = e$
            — $e = e + 1$
        else if input channel with index $S_i$ is a horizontal channel (label contains '_M_')
                — $h$ = min(elevation of randomized output channel, 35) / 35
                — Define new equalizer $\mathbf{G}_{\mathrm{EQ},e}$ with the index $e$,
                  where $G_{EQ,e}^{pb} = \max\left(0.6310, \min\left(1.5849, \left(h \cdot IEQ(D_i)^{pb} + (1-h)\right) \cdot \mathrm{G}_{EQ,E_i}^{pb}\right)\right)$
                  and $IEQ(D_i)^{pb}$ $\mathrm{IEQ(D_i)^k}$ is defined in Table FC5.5
                — $E_i = e$
                — $e = e + 1$
```

Table FC5.5 — Inverse spatial elevation filter (EQ$_{0,\mathrm{lin}}$ is defined in Table FC5.9)

Azimuth of D_i	**Front Center (-15 15)**	**Front (-90 -15) or (15 90)**	**Side/Rear [-180 -90] or [90 180]**
$IEQ(D_i)^{pb}$	$\frac{1}{G_{EQ,9}^{pb}}$	$\frac{1}{G_{EQ,7}^{pb}}$	$\frac{1}{G_{EQ,8}^{pb}}$

Explanation of the post-processing steps defined above:

h is a normalized elevation parameter indicating the elevation of a nominally horizontal output channel ('_M_') due to a random setup elevation offset $r_{\mathrm{ele},A}$. For zero elevation offset *h*=0 follows and effectively no post-processing is applied.

The rule table (Table FC5.14) in general applies a gain of the value in the gain column when mapping an upper input channel ('_U_' in channel label) to one or several horizontal output channels ('_M_' in channel label(s)). In case the output channel gets elevated due to a random setup elevation offset $r_{\mathrm{ele},A}$, the gain is partially (0<*h*<1) or fully (*h*=1) compensated for. Similarly the equalizer definitions fade towards a flat EQ-curve ($G_{EQ,e}^{pb} = const. = 1.0$) for *h* approaching *h* =1.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

In case a horizontal input channel gets mapped to an output channel that gets elevated due to a random setup elevation offset $r_{\mathrm{ele},A}$, the equalizer $\mathrm{G}^{\mathrm{k}}_{\mathrm{EQ,E_i}}$ fully applied and $IEQ(D_i)^{pb}$, an inverse form of spatial elevation filter defined in Table FC5.5, is partially (0<*h*<1) or fully (*h*=1) applied. As the spatial elevation filter is defined to provide the tone color of overhead loudspeakers on horizontal loudspeakers, the inverse of the spatial elevation filter is used to provide the tone color of horizontal loudspeakers on overhead loudspeakers. Note that the modified EQ is thresholded within the level of 4 dB, [0.6310, 1.5849].

FC5.4.1.6.6 Spatial coloration filter for the horizontal input channels

When a horizontal input channel at side or rear is panned by two output loudspeakers, e.g. CH_M_L090 is panned by CH_M_L030 and CH_M_L110, the tone color changes. In order to avoid such a change in tone color, a set of horizontal coloration filters of $\mathbf{G}_{\mathrm{EQ},15\sim20}$ are defined as in Table FC5.6. Here, the filter name COLOR_A_B means the coloration filter for the output at the azimuth of ±B from the input at the azimuth of ±A. The filtering algorithm is:

```
For each element i in S do:
    A : the magnitude of the azimuth of the input channel with index $S_i$
    B : the magnitude of the azimuth of the output channel with index $D_i$
    If both $S_i$ and $D_i$ are horizontal channels and there exists a filter COLOR_A_B_ in Table FC5.6,
        If the output channel has no deviation in azimuth and elevation, ( $r_{azi,D_i}$=0 and $r_{ele,D_i}$=0 )
            If      A==60   && B == 30    $E_i$ = 15 (COLOR_60_30)
            Elseif  A==90   && B == 30    $E_i$ = 16 (COLOR_90_30)
            Elseif  A==60   && B == 110   $E_i$ = 17 (COLOR_60_110)
            Elseif  A==90   && B == 110   $E_i$ = 18 (COLOR_90_110)
            Elseif  A==135  && B == 110   $E_i$ = 19 (COLOR_135_110)
            Elseif  A==180  && B == 110   $E_i$ = 20 (COLOR_180_110)
```

Table FC5.6 — Spatial coloration filters for horizontal channels

processing band	COLOR_180_110 $\mathbf{G}_{\mathrm{EQ},20}$	COLOR_090_030 $\mathbf{G}_{\mathrm{EQ},16}$	COLOR_060_110 $\mathbf{G}_{\mathrm{EQ},17}$	COLOR_135_110 $\mathbf{G}_{\mathrm{EQ},19}$	COLOR_090_110 $\mathbf{G}_{\mathrm{EQ},18}$	COLOR_060_030 $\mathbf{G}_{\mathrm{EQ},15}$
0	1.257512901	1.016393	0.975283	1.057872	0.967818	1.024233
1	1.158560317	0.940154	1.020111	1.025689	1.000571	0.958515
2	0.975947998	0.924468	1.052661	0.972375	1.03672	0.938683
3	0.812670539	1.019364	1.034273	0.940191	1.0457	1.008225
4	0.793650794	1.163062	1.007329	0.931031	1.043259	1.123006
5	0.793650794	1.255373	1.010088	0.925814	1.048451	1.209438
6	0.793651	1.221591	1.044259	0.911691	1.054614	1.209597

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

7	0.793651	1.164189	1.06016	0.895357	1.033851	1.193814
8	0.793651	1.151248	1.028108	0.894999	0.988836	1.196971
9	0.793651	1.174002	0.966481	0.915049	0.963343	1.177826
10	0.793651	1.235628	0.915263	0.954878	1.003832	1.126607
11	0.793651	1.26	0.914371	0.9984	1.075405	1.132111
12	0.793651	1.26	0.942626	1.002731	1.061876	1.26
13	0.793651	1.26	0.931554	0.959758	0.946306	1.26
14	0.793651	1.26	0.880058	0.919532	0.851115	1.26
15	0.793651	1.26	0.825	0.9062	0.840038	1.26
16	0.793651	1.22276	0.814011	0.917664	0.892063	1.115775
17	0.793651	1.160128	0.864691	0.949673	0.94146	1.065528
18	0.793651	1.089699	0.936597	0.993391	0.956656	1.066851
19	0.793651	1.037545	0.997666	1.029246	0.96065	1.077524
20	0.793651	0.986758	1.042468	1.031808	0.976646	1.053261
21	0.793651	0.929318	1.071918	1.001631	1.011122	0.985194
22	0.793651	0.883289	1.103262	0.96871	1.066606	0.913645
23	0.793651	0.858581	1.176045	0.959278	1.143043	0.88337
24	0.839215	0.849521	1.26	0.97236	1.234495	0.901514
25	0.895853	0.85036	1.26	0.976487	1.26	0.93963
26	0.921032	0.867297	1.26	0.950306	1.26	0.976639
27	0.915816	0.902004	1.26	0.916408	1.26	1.021141
28	0.907476	0.943387	1.26	0.910758	1.26	1.085708
29	0.902366	0.977346	1.26	0.928637	1.26	1.146713
30	0.900041	1.008207	1.26	0.934616	1.26	1.170319
31	0.903678	1.043796	1.26	0.90898	1.26	1.167112
32	0.895045	1.063373	1.26	0.866561	1.26	1.176141
33	0.827595	1.056064	1.26	0.794462	1.26	1.26
34	0.81639	1.075206	1.26	0.807225	1.26	1.229941
35	0.793651	1.002088	1.26	0.793651	1.26	1.094809
36	0.848212	0.893956	1.26	0.793651	1.26	0.991888
37	0.843676	0.944609	1.26	0.793651	1.26	1.054892
38	0.807275	1.002863	1.26	0.793651	1.26	1.176083
39	0.793651	1.199923	1.26	0.793651	1.26	1.26
40	0.793651	1.26	1.26	0.793651	1.26	1.26
41	0.793651	1.26	1.26	0.793651	1.26	1.26
42	0.793651	1.26	1.119248	0.793651	1.26	1.26
43	0.793651	1.26	0.912938	0.793651	1.186579	1.26
44	0.793651	1.26	0.793651	0.793651	1.047277	1.26
45	0.793651	1.26	0.793651	0.793651	0.958699	1.26
46	0.793651	1.26	0.793651	0.836434	0.937138	1.26

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

47	0.793651	1.26	1.102116	0.793651	1.092959	1.26
48	0.793651	1.26	1.092848	0.793651	1.26	1.26
49	0.793651	1.26	0.793651	0.815618	1.207313	1.26
50	0.793651	1.26	0.793651	0.818016	1.021066	0.793651
51	0.793651	0.922395	0.793651	0.899999	1.008149	0.793651
52	0.793651	0.793651	1.114079	0.856971	1.047784	0.793651
53	0.793651	0.793651	1.26	0.802702	1.26	0.793651
54	0.793651	0.976107	1.26	0.793651	1.26	0.992957
55	0.793651	1.153876	1.26	0.793651	1.26	1.034704
56	0.793651	1.26	1.231165	0.793651	1.26	1.26
57	0.793651	1.26	0.976335	0.793651	1.067507	1.26

FC5.4.1.6.7 Elevation rendering

FC5.4.1.6.7.1 General

Elevation rendering provides overhead sound image using 5.1 channel layout. When the **Destination** column contains **VIRTUAL**, the **isPossibleElev** returns whether the input channel shall be rendered by the elevation rendering by comparing the output channel configuration and the required output channels for the spatial elevation rendering.

Table FC5.7 — Detail information of the Elevation rendering initialization parameters

$\mathbf{S}^{\mathrm{P}}$	Vector of converter source channels [input channel indices]	Initialization parameters for spatial elevation rendering (renderElevSptlParms)
$\mathbf{D}^{\mathrm{P}}$	Vector converter destination channels [output channel indices]	
$\mathbf{G}^{\mathrm{P}}_{\mathrm{H}}$	Vector of converter gains for 2.8~10kHz components	
$\mathbf{G}^{\mathrm{P}}_{\mathrm{L}}$	Vector of converter gains below 2.8kHz and above 10kHz components	
$\mathbf{E}^{\mathrm{P}}$	Vector of converter EQ indices	
$\mathbf{S}^{\mathrm{S}}$	Vector of converter source channels [input channel indices]	Initialization parameters for timbral elevation rendering (renderElevTmbrParms)
$\mathbf{D}^{\mathrm{S}}$	Vector converter destination channels [output channel indices]	
$\mathbf{G}^{\mathrm{S}}$	Vector of converter gains [linear]	
$\mathbf{E}^{\mathrm{S}}$	Vector of converter EQ indices	
$\mathbf{G}_{\mathrm{EQ}}$	Matrix containing equalizer gain values for all EQ indices and frequency bands	EQ matrix for all rendering types

When **isPossibleElev** is **TRUE**, two sets of initialization parameters for the spatial elevation rendering and timbral elevation rendering is initialized using **renderElevSptlParms** and **renderElevTmbrParms**. The parameters of $\mathbf{S}^{\mathrm{P}}$, $\mathbf{D}^{\mathrm{P}}$, $\mathbf{G}^{\mathrm{P}}_{\mathrm{H}}$, $\mathbf{G}^{\mathrm{P}}_{\mathrm{L}}$, and $\mathbf{E}^{\mathrm{P}}$ initialized by **renderElevSptlParms** are for the spatial elevation rendering, and $\mathbf{S}^{\mathrm{S}}$, $\mathbf{D}^{\mathrm{S}}$, $\mathbf{G}^{\mathrm{S}}$, and $\mathbf{E}^{\mathrm{S}}$ by **renderElevTmbrParms** are for the timbral elevation rendering. The intermediate parameters describe the dowmixing parameters in a mapping-oriented way, i.e. as sets of parameters S^{P}_{i}, D^{P}_{i}, G^{P}_{iH}, G^{P}_{iL}, and E^{P}_{i} per mapping *i*.

FC5.4.1.6.7.2 isPossibleElev : Decision whether elevation rendering is valid for the output layout

The **isPossibleElev** returns **TRUE** when all the required output channels exist. The required output channels are defined in Table FC5.8, indicating 1 for the required channel and 0 for the unnecessary channel. For example, CH_M_L030, CH_M_L110, and CH_M_R110 are required in order to use elevation rendering for the input channel CH_U_L030. If any of the required output channels for the input channel is not in the output channel configuration, **isPossibleElev** returns **FALSE** and elevation rendering is not applied for the input channel.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

Table FC5.8 — Required Output Channels for Elevation Rendering for isPossibleElev

Input Channel	Required Output Channels					
	CH_M_L030	CH_M_R030	CH_M_000	CH_LFE1	CH_M_L110	CH_M_R110
CH_U_000	1	1	1	0	1	1
CH_U_L045	1	1	0	0	1	1
CH_U_R045	1	1	0	0	1	1
CH_U_L030	1	1	0	0	1	1
CH_U_R030	1	1	0	0	1	1
CH_U_L090	1	0	0	0	1	1
CH_U_R090	0	1	0	0	1	1
CH_U_L110	1	0	0	0	1	1
CH_U_R110	0	1	0	0	1	1
CH_U_L135	1	0	0	0	1	1
CH_U_R135	0	1	0	0	1	1
CH_U_180	1	1	0	0	1	1
CH_T_000	1	1	1	0	1	1

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

FC5.4.1.6.7.3 initElevSptlParms: Initialization of elevation rendering parameters based on the input channel elevation

The initial values of the spatial elevation filters and the elevation panning coefficients are defined in Table FC5.9, Table FC5.10, and Table FC5.11 for the 'height' input channels, except CH_T_000. When a 'height' input channel except CH_T_000 has the elevation higher than 35 degree, the spatial elevation filter coefficients and elevation panning coefficients shall be updated according to the degree of the elevation. Note that the elevation sector is defined from +21 to +60 degree for the height channels in Table FC5.14.

For the elevation rendering of an input channel, an EQ is selected from the EQ column of the Table FC5.14. For convenience, $\mathrm{eq}(\mathrm{i}_{\mathrm{in}})$ is defined as the EQ column, e.g. if the i_{in} is CH_U_000, $\mathrm{eq}(\mathrm{i}_{\mathrm{in}})$ is 9 (EQVFC).

Table FC5.9 — Spatial elevation filter initial values (for the 35 degree in elevation)

processing bands (58 bands)	$EQ_{0,lin}$ (.)							
	7 (EQVF)	8 (EQVB)	9 (EQVFC)	10 (EQVBC)	11 (EQVOG)	12 (EQVS)	13 (EQBTM)	14 (EQVBA)
0	0.841395	0.819093	0.685085	0.770312	0.625280	0.877674	0.922571	0.877674
1	0.841395	0.819093	0.685085	0.770312	0.625280	0.877674	0.922571	0.877674
2	0.926119	0.922571	0.830000	0.940445	0.625280	0.901571	1.143756	0.988553
3	0.926119	0.980995	0.830000	1.015469	0.675162	1.000000	1.117721	1.051155
4	0.926119	0.980995	0.830000	1.015469	0.675162	1.000000	1.117721	1.051155
5	0.944061	0.887837	0.768679	1.000000	0.729024	0.905038	1.023293	0.951335
6	0.944061	0.887837	0.768679	1.000000	0.729024	0.905038	1.023293	0.951335
7	0.944061	0.887837	0.768679	1.000000	0.729024	0.905038	1.023293	0.951335
8	0.944061	0.887837	0.768679	1.000000	0.729024	0.905038	1.023293	0.951335
9	0.944061	0.887837	0.768679	1.000000	0.729024	0.905038	1.023293	0.951335

10	0.944061	0.887837	0.768679	1.000000	0.729024	0.905038	1.023293	0.951335
11	0.908518	0.958665	0.768679	0.887837	0.701576	1.039122	0.922571	1.027228
12	0.908518	0.958665	0.768679	0.887837	0.701576	1.039122	0.922571	1.027228
13	0.908518	0.958665	0.768679	0.887837	0.701576	1.039122	0.922571	1.027228
14	0.908518	0.958665	0.768679	0.887837	0.701576	1.039122	0.922571	1.027228
15	0.908518	0.958665	0.768679	0.887837	0.701576	1.039122	0.922571	1.027228
16	0.908518	0.958665	0.768679	0.887837	0.701576	1.039122	0.922571	1.027228
17	0.908518	0.958665	0.768679	0.887837	0.701576	1.039122	0.922571	1.027228
18	0.908518	0.958665	0.768679	0.887837	0.701576	1.039122	0.922571	1.027228
19	0.944061	0.936843	0.798751	0.838172	0.729024	1.023293	0.940445	0.860994
20	0.944061	0.936843	0.798751	0.838172	0.729024	1.023293	0.940445	0.860994
21	0.944061	0.936843	0.798751	0.838172	0.729024	1.023293	0.940445	0.860994
22	0.944061	0.936843	0.798751	0.838172	0.729024	1.023293	0.940445	0.860994
23	0.944061	0.936843	0.798751	0.838172	0.729024	1.023293	0.940445	0.860994
24	0.944061	0.936843	0.798751	0.838172	0.729024	1.023293	0.940445	0.860994
25	0.944061	0.936843	0.798751	0.838172	0.729024	1.023293	0.940445	0.860994
26	0.944061	0.936843	0.798751	0.838172	0.729024	1.023293	0.940445	0.860994
27	0.944061	0.936843	0.798751	0.838172	0.729024	1.023293	0.940445	0.860994
28	0.980995	0.962351	0.830000	0.922571	0.649743	0.940445	0.940445	0.992354
29	0.980995	0.962351	0.830000	0.922571	0.649743	0.940445	0.940445	0.992354
30	0.980995	0.962351	0.830000	0.922571	0.649743	0.940445	0.940445	0.992354
31	0.980995	0.962351	0.830000	0.922571	0.649743	0.940445	0.940445	0.992354
32	0.980995	0.962351	0.830000	0.922571	0.649743	0.940445	0.940445	0.992354
33	0.980995	0.962351	0.830000	0.922571	0.649743	0.940445	0.940445	0.992354
34	0.980995	0.962351	0.830000	0.922571	0.649743	0.940445	0.940445	0.992354
35	0.980995	0.962351	0.830000	0.922571	0.649743	0.940445	0.940445	0.992354
36	0.929311	0.892125	0.786271	0.855249	0.615512	0.871818	0.871818	0.919939
37	0.881628	0.908657	0.717845	0.908657	0.630513	0.944205	0.822369	0.901709
38	0.870100	0.892065	0.708457	0.892065	0.622267	0.926965	0.807353	0.885244
39	0.926766	0.855247	0.754596	0.823047	0.662793	0.909407	0.875168	0.848708
40	0.913913	0.838687	0.744132	0.807111	0.653601	0.891798	0.858223	0.832275
41	0.902215	0.826864	0.734606	0.823697	0.620943	0.875859	0.934905	0.886001
42	0.889454	0.810537	0.724216	0.807432	0.612160	0.858564	0.916444	0.868506
43	0.876028	0.793461	0.713285	0.790422	0.602920	0.840476	0.897137	0.850208
44	0.897706	0.793209	0.730935	0.775153	0.642011	0.856488	0.924814	0.907238
45	0.789863	0.779063	0.643126	0.809542	0.564885	0.809542	0.995954	0.911813
46	0.779856	0.733647	0.634979	0.706025	0.536730	0.765280	0.835899	0.786116
47	0.660004	0.546217	0.537392	0.484952	0.490480	0.567586	0.692945	0.585282
48	0.650985	0.535796	0.530049	0.475700	0.483778	0.556758	0.679725	0.574116
49	0.641943	0.525406	0.522686	0.466475	0.477058	0.545961	0.666543	0.562982

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

50	0.632968	0.515151	0.515379	0.457370	0.388260	0.535305	0.653533	0.551994
51	0.624467	0.505491	0.508457	0.448794	0.383046	0.525267	0.641279	0.541644
52	0.616092	0.496025	0.501638	0.440390	0.377909	0.515431	0.629270	0.531501
53	0.607606	0.486487	0.494729	0.431921	0.372704	0.505519	0.617170	0.521280
54	0.599370	0.477280	0.488023	0.423747	0.367652	0.495952	0.605490	0.511415
55	0.591397	0.468415	0.481530	0.415876	0.362761	0.486740	0.594243	0.501916
56	0.583481	0.459661	0.475085	0.408104	0.357906	0.477644	0.583138	0.492536
57	0.576447	0.451922	0.469358	0.401233	0.353590	0.469602	0.573319	0.484243

When the 'height' input channel i_{in} has higher elevation than 35 degree, the spatial elevation filter is calculated from $EQ_{0,lin}^{pb}(\mathrm{eq}(\mathrm{i}_{in}))$ in Table FC5.9 by

If the input channel, i_{in}, is frontal side : azimuth is in the range of (-90, 90)

$$EQ_{1,db}^{pb}\left(eq\left(i_{in}\right)\right)=\begin{cases}20\times\log_{10}\left(EQ_{0,lin}^{pb}\left(eq\left(i_{in}\right)\right)\right)+0.05\times\log_{2}\left(f_{pb}\times f_{s}/6000\right)+1 & \text{for } f_{pb}\times f_{s}>8000\\ 20\times\log_{10}\left(EQ_{0,lin}^{pb}\left(eq\left(i_{in}\right)\right)\right)+1 & \text{otherwise}\end{cases}$$

$$EQ_{2,db}^{pb}\left(eq\left(i_{in}\right)\right)=EQ_{1,lin}^{pb}\left(eq\left(i_{in}\right)\right)\times\left(1+\min(\max(\mathrm{elv}-35,0),25)\times 0.05\right)$$

$$G_{\mathrm{EQ},eq\left(i_{in}\right)}^{pb}=\begin{cases}10^{\left(EQ_{2,db}^{pb}\left(eq\left(i_{in}\right)\right)-1\right)/20-0.05\times\log_{2}\left(f_{pb}\times f_{s}/6000\right)} & \text{for } f_{pb}\times f_{s}>8000\\ 10^{\left(EQ_{2,db}^{pb}\left(eq\left(i_{in}\right)\right)-1\right)/20} & \text{otherwise}\end{cases}$$

else (the input channel is rear side : azimuth is either in [-180, -90] or in [90, 180])

$$EQ_{1,db}^{pb}\left(eq\left(i_{in}\right)\right)=\begin{cases}20\times\log_{10}\left(EQ_{0,lin}^{pb}\left(eq\left(i_{in}\right)\right)\right)+0.07\times\log_{2}\left(f_{pb}\times f_{s}/6000\right)+1 & \text{for } f_{pb}\times f_{s}>8000\\ 20\times\log_{10}\left(EQ_{0,lin}^{pb}\left(eq\left(i_{in}\right)\right)\right)+1 & \text{otherwise}\end{cases}$$

$$EQ_{2,db}^{pb}\left(eq\left(i_{in}\right)\right)=EQ_{1,lin}^{pb}\left(eq\left(i_{in}\right)\right)\times\left(1+\min(\max(\mathrm{elv}-35,0),25)\times 0.05\right)$$

$$G_{\mathrm{EQ},eq\left(i_{in}\right)}^{pb}=\begin{cases}10^{\left(EQ_{2,db}^{pb}\left(eq\left(i_{in}\right)\right)-1\right)/20-0.07\times\log_{2}\left(f_{pb}\times f_{s}/6000\right)} & \text{for } f_{pb}\times f_{s}>8000\\ 10^{\left(EQ_{2,db}^{pb}\left(eq\left(i_{in}\right)\right)-1\right)/20} & \text{otherwise}\end{cases}$$

where f_{pb} is the normalized center frequency of processing band pb, specified in Table FC5.15, f_s is the sampling frequency, and elv is the elevation of the input channel.

When the 'height' input channel i_{in} does not have higher elevation than 35 degree, use the initial filter coefficients by:

$$G_{\mathrm{EQ,eq}(\mathrm{i}_{in})}^{pb}=EQ_{0,lin}^{pb}(\mathrm{eq}(\mathrm{i}_{in}))$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

The spatial elevation panning coefficients shall also be updated for the 'height' input channels, except CH_T_000 and CH_U_180, for the different elevation degree. Table FC5.10 and Table FC5.11 show the initial panning coefficients for the input channels with the elevation of 35 degree. When the elevation of the input channel is higher than 35 degree, the ipsilateral gain to the input channel shall be reduced and the contralateral channel shall be boosted with the gain difference $g_I(\mathrm{elv})$ and $g_C(\mathrm{elv})$.

Table FC5.10 — Initial spatial localization panning coefficients for 2.8 kHz ~ 10 kHz (35 degree in elevation)

Channel Label, i_{in}	$G_{vH0,1\sim6}(i_{in})$					
	CH_M_L030 ($G_{vH0,1}(i_{in})$)	CH_M_R030 ($G_{vH0,2}(i_{in})$)	CH_M_000 ($G_{vH0,3}(i_{in})$)	CH_LFE1 ($G_{vH0,4}(i_{in})$)	CH_M_L110 ($G_{vH0,5}(i_{in})$)	CH_M_R110 ($G_{vH0,6}(i_{in})$)
CH_U_000	0.49146774	0.49146774	0.34746769	0	0.44507593	0.44507593
CH_U_L045	0.70918131	0.27444959	0	0	0.56982642	0.31150770
CH_U_R045	0.27444959	0.70918131	0	0	0.31150770	0.56982642
CH_U_L030	0.70918131	0.27444959	0	0	0.56982642	0.31150770
CH_U_R030	0.27444959	0.70918131	0	0	0.31150770	0.56982642
CH_U_L090	0.56040317	0	0	0	0.81550622	0.14456093
CH_U_R090	0	0.56040317	0	0	0.14456093	0.81550622
CH_U_L110	0.34278116	0	0	0	0.91200900	0.22525696
CH_U_R110	0	0.34278116	0	0	0.22525696	0.91200900
CH_U_L135	0.34278116	0	0	0	0.91200900	0.22525696
CH_U_R135	0	0.34278116	0	0	0.22525696	0.91200900
CH_U_180	0.22851810	0.22851810	0	0	0.66916323	0.66916323
CH_T_000	0.45328009	0.45328009	0.33519593	0	0.48822021	0.48822021

Table FC5.11 — Initial spatial localization panning coefficients below 2.8 kHz and above 10 kHz (35 degree in elevation)

Channel Label, i_{in}	$G_{vL0,1\sim6}(i_{in})$					
	CH_M_L030 ($G_{vL0,1}(i_{in})$)	CH_M_R030 ($G_{vL0,2}(i_{in})$)	CH_M_000 ($G_{vL0,3}(i_{in})$)	CH_LFE1 ($G_{vL0,4}(i_{in})$)	CH_M_L110 ($G_{vL0,5}(i_{in})$)	CH_M_R110 ($G_{vL0,6}(i_{in})$)
CH_U_000	0.61940062	0.61940062	0.43791625	0	0	0
CH_U_L045	1	0	0	0	0	0
CH_U_R045	0	1	0	0	0	0
CH_U_L030	1	0	0	0	0	0
CH_U_R030	0	1	0	0	0	0
CH_U_L090	0.36730000	0	0	0	0.93010002	0
CH_U_R090	0	0.36730000	0	0	0	0.93010002
CH_U_L110	0	0	0	0	1	0
CH_U_R110	0	0	0	0	0	1
CH_U_L135	0.34278116	0	0	0	0.91200900	0.22525696

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

CH_U_R135	0	0.34278116	0	0	0.22525696	0.91200900
CH_U_180	0.22851810	0.22851810	0	0	0.66916323	0.66916323
CH_T_000	0.45328009	0.45328009	0.33519593	0	0.48822021	0.48822021

For all height input channel $\mathrm{i_{in}}$, the $\mathrm{G_{vH,1\sim6}}$ and $\mathrm{G_{vL,1\sim6}}$ shall be initialized with $\mathrm{G_{vH0,1\sim6}}$:

$$\mathrm{G_{vH,1\sim6}(i_{in}) = G_{vH0,1\sim6}(i_{in})}$$

$$\mathrm{G_{vL,1\sim6}(i_{in}) = G_{vL0,1\sim6}(i_{in})}$$

For each height input channel $\mathrm{i_{in}}$, the $\mathrm{G_{vH,1\sim6}}$ shall be calculated from $\mathrm{G_{vH0,1\sim6}}$:

If the input channel is CH_U_000,

$$\mathrm{G_{vH,5}(i_{in}) = 10^{(0.25\times\min(\max(elv-35,0),25))/20} \times G_{vH0,5}(i_{in})}$$

$$\mathrm{G_{vH,6}(i_{in}) = 10^{(0.25\times\min(\max(elv-35,0),25))/20} \times G_{vH0,6}(i_{in})}$$

Elseif the input channel is not CH_U_180

if the input channel is side : azimuth is either in (-110, -70) or in (70, 110),

$$\mathrm{g_I(elv) = 10^{(-0.05522\times\min(\max(elv-35,0),25))/20}}$$

$$\mathrm{g_C(elv) = 10^{(0.41879\times\min(\max(elv-35,0),25))/20}}$$

else (the input channel is frontal or rear in [-70, 70], [-180, -110], or [110 180])

$$\mathrm{g_I(elv) = 10^{(-0.047401\times\min(\max(elv-35,0),25))/20}}$$

$$\mathrm{g_C(elv) = 10^{(0.14985\times\min(\max(elv-35,0),25))/20}}$$

For each output channels ipsilateral to the input channel,
(e.g. CH_M_L030 and CH_M_L110 for CH_U_L045)

$$\mathrm{G_{vH,I}(i_{in}) = g_I(elv) \times G_{vH0,I}(i_{in})}$$

where the I of $\mathrm{G_{vH,I}(i_{in})}$ represents the indices ipsilateral to the $\mathrm{i_{in}}$

For each output channels contralateral to the input channel,
(e.g. CH_M_R030 and CH_M_R110 for CH_U_L045)

$$\mathrm{G_{vH,C}(i_{in}) = g_C(elv) \times G_{vH0,C}(i_{in})}$$

where the C of $\mathrm{G_{vH,C}(i_{in})}$ represents the indices contralateral to the $\mathrm{i_{in}}$

A set of spatial localization panning coefficients for the input channel is normalized to make the sum of powers be 1.

$$\mathrm{P_{G_{vH}}(i_{in}) = \sqrt{\textstyle\sum_{o=1}^{6} G_{vH,o}^2\,(i_{in})}}$$

$$\mathrm{G_{vH,1\sim6}(i_{in}) = \frac{1}{P_{G_{vH}}} G_{vH,1\sim6}(i_{in})}$$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

If the input channel is in [-160, -110) or (110 160], $\mathrm{G_{vL,C}(i_{in})}$ shall be updated :

$$\mathrm{G_{vL,I}(i_{in}) = g_I(elv) \times G_{vL0,I}(i_{in})}$$

$$\mathrm{G_{vL,C}(i_{in}) = g_C(elv) \times G_{vL0,C}(i_{in})}$$

$$\mathrm{P_{G_{vL}}(i_{in}) = \sqrt{\textstyle\sum_{o=1}^{6} G_{vL,o}^2(i_{in})}}$$

$$\mathrm{G_{vL,1\sim6}(i_{in}) = \frac{1}{P_{G_{vL}}} G_{vL,1\sim6}(i_{in})}$$

Note that only the panning coefficients of the $\mathrm{G_{vL,1\sim6}}$ for the rear input channels change and those for the frontal/side input channel do not.

As a result of **initElevSptlParms** following parameters are derived :

$G^{pb}_{\mathrm{EQ,eq(i_{\mathit{in}})}}$: Updated spatial elevation filter coefficient vector (58 bands) for the input channel $\mathrm{i_{in}}$.

$\mathrm{G_{vH,1\sim6}(i_{in})}$: Updated spatial elevation panning coefficients for the input signal in the range of 2.8~10 kHz

$\mathrm{G_{vL,1\sim6}(i_{in})}$: Updated spatial elevation panning coefficients for the input signal below 2.8 kHz and above 10kHz

FC5.4.1.6.7.4 renderElevSptlParms : Derivation of input-output channel mapping and equalizer for spatial elevation rendering

renderElevSptlParms initializes the input-output channel mapping for spatial elevation rendering for an input channel (i_{in}).

```
Initialize the mapping counter $i=1$;

For $m$=1 to 6

    If    $G_{vH,m}(i_{in}) > 1$

          $s^P_i$ = index of source channel $i_{in}$

          $d^P_i$ = index of channel $m$ in output

          $g^P_{iH}$ = (value of Gain column) * $G_{vH,m}(i_{in})$

          $g^P_{iL}$ = (value of Gain column) * $G_{vL,m}(i_{in})$

          $e^P_i$ = $\mathrm{eq(i_{in})}$

          $i = i + 1$

$n^P = i - 1$;

return { $\mathbf{s}^P$, $\mathbf{d}^P$, $\mathbf{g}^P_H$, $\mathbf{g}^P_L$, $\mathbf{e}^P$, and $n^P$ }
```

Note that the initialization does not add the input-output channel mapping that the gain g^P_{iH} is zero and $\mathrm{eq(i_{in})}$ is from 7 to 15.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

By applying the downmix rules defined in Table FC5.14 with the spatial elevation filter and spatial localization panning coefficients, the spatial elevation rendering parameters for each input channel is summarized as :

- CH_U_L135, CH_U_R135, CH_U_180, or CH_T_000
  - Spatial Elevation Filter : the HRTF-based EQs (EQVB , EQVBC, or EQVOG)
  - Spatial Elevation Panning : Filtered signal is multiplied with the same panning coefficients over all frequency (the $\mathbf{g}^{P}_{H}$ is identical to the $\mathbf{g}^{P}_{L}$)
- CH_U_L110, and CH_U_R110
  - Spatial Elevation Filter : the HRTF-based EQ (EQVBA)
  - Spatial Elevation Panning : Filtered EQ signal is multiplied with the different panning coefficients over frequency range
    - $\mathbf{g}^{P}_{H}$ for the elevation-effective range (2.8k~10kHz)
    - $\mathbf{g}^{P}_{L}$ for the rest frequency range : Use the “add-to-the-closest channel” method in order to provide enough envelopment and keep the audio channel wide enough.
- CH_U_L090 and CH_U_R090
  - Spatial Elevation Filter : the HRTF-based EQs (EQVS)
  - Spatial Elevation Panning : Filtered EQ signal is multiplied with the different panning coefficients over frequency range
    - $\mathbf{g}^{P}_{H}$ for the elevation-effective range (2.8k~10kHz)
    - $\mathbf{g}^{P}_{L}$ for the rest frequency range : Panned at 90 degree using front and surround loudspeakers in order to provide enough envelopment and keep the audio channel wide enough.
- CH_U_L030, CH_U_R030, CH_U_L045, CH_U_R045
  - Spatial Elevation Filter : the HRTF-based EQs (EQVF)
  - Spatial Elevation Panning : Filtered EQ signal is multiplied with the different panning coefficients over frequency range
    - $\mathbf{g}^{P}_{H}$ for the elevation-effective range (2.8k~10kHz)
    - $\mathbf{g}^{P}_{L}$ for the rest frequency range : Use the “add-to-the-closest channel” method in order to provide enough envelopment and keep the audio channel wide enough.
  - Signals for the surround speakers, CH_M_L110 and CH_M_R110, are delayed by one STFT hop size in order to avoid front-back confusion using precedence effect. See details in FC5.4.2.3
- CH_U_000
  - Spatial Elevation Filter : : the HRTF-based EQ (EQVFC)
  - Spatial Elevation Panning : Filtered EQ signal is multiplied with the different panning coefficients over frequency range
    - $\mathbf{g}^{P}_{H}$ for the elevation-effective range (2.8k~10kHz)
    - $\mathbf{g}^{P}_{L}$ for the rest frequency range : panned among three frontal output channels, CH_M_L030, CH_M_000, and CH_M_R030.
  - Signals for the surround speakers, CH_M_L110 and CH_M_R110, are delayed by one STFT hop size in order to avoid front-back confusion using precedence effect. See details in FC5.4.2.3

FC5.4.1.6.7.5 renderElevTmbrParms : Derivation of input-output channel mapping and equalizer for timbral elevation rendering

After the **renderElevSptlParms**, **renderElevTmbrParms** initializes another input-output channel mapping for timbral elevation rendering for the same input channel i_{in}. The parameters are initialized following FC5.4.1.6.3 but ignoring the downmix rules with a destination field of **VIRTUAL**. As a result, a set of $\mathbf{s}^{S}$, $\mathbf{d}^{S}$, $\mathbf{g}^{S}$, $\mathbf{e}^{S}$, and $\mathbf{gain}^{S}$ are defined for the input channel i_{in} by the process:

```
Initialize the mapping counter $i=1$;

Search the first entry of the input channel in the Source column of the Table FC5.4 and the channels in
the Destination column exist below the entry with VIRTUAL destination.

   If Destination column contains ALL_U, then:
         For each output channel x with “CH_U_” in its name, do:
               $s^{S}_{i}$ = index of source channel in input
               $d^{S}_{i}$ = index of channel x in output
               $g^{S}_{i}$ = (value of Gain column) / sqrt(number of “CH_U_” output channels)
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
                    $e^S_i$ = value of EQ column
                    $gain_i$ = (value of Gain column)
                    $i = i + 1$
    Else if **Destination** column contains ALL_M, then:
            For each output channel x with “CH_M_” in its name, do:
                    $s^S_i$ = index of source channel in input
                    $d^S_i$ = index of channel x in output
                    $g^S_i$ = (value of Gain column) / sqrt(number of “CH_M_” output channels)
                    $e^S_i$ = value of EQ column
                    $gain_i$ = (value of Gain column)
                    $i = i + 1$
    Else If there is one channel in the **Destination** column, then:
            $s^S_i$ = index of source channel in input
            $d^S_i$ = index of destination channel in output
            $g^S_i$ = value of Gain column
            $e^S_i$ = value of EQ column
            if $e^S_i$ == 13 (EQBTM)
              $e^S_i$ = 0 (No Process)
            $gain_i$ = (value of Gain column)
            $i = i + 1$
    Else (two channels in **Destination** column)
            $s^S_i$ = index of source channel in input
            $d^S_i$ = index of first destination channel in output
            $g^S_i$ = (value of Gain column) * $g_1$
            $e^S_i$ = value of EQ column
            if $e^S_i$ == 13 (EQBTM)
              $e^S_i$ = 0 (No Process)
            $gain_i$ = (value of Gain column)
            $i = i + 1$

            $s^S_i$ = $s^S_{i-1}$
            $d^S_i$ = index of second destination channel in output
            $g^S_i$ = (value of Gain column) * $g_2$
            $e^S_i$ = $e^S_{i-1}$
            $gain_i$ = (value of Gain column)
            $i = i + 1$
     $n^S$ = $i$ - 1;
     return { $\mathbf{s}^S$, $\mathbf{d}^S$, $\mathbf{g}^S$, $\mathbf{e}^S$, **gain** and $n^S$ }
```

The gains g_1 and g_2 are computed by applying tangent law amplitude panning in the following way:

— Unwrap source destination channel azimuth angles to be positive.

— The azimuth angles of the destination channels are α_1 and α_2 (see Table FC5.12).

— The azimuth angle of the source channel (= panning target) is α_{src}.

— $\alpha_0 = \dfrac{|\alpha_1 - \alpha_2|}{2}$

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

— $\alpha_{\text{center}} = \dfrac{\alpha_1 + \alpha_2}{2}$

— $\alpha = (\alpha_{\text{center}} - \alpha_{\text{src}}) \cdot \operatorname{sgn}(\alpha_2 - \alpha_1)$

— $g_1 = \dfrac{g}{\sqrt{1+g^2}}, \quad g_2 = \dfrac{1}{\sqrt{1+g^2}} \quad \text{with} \quad g = \dfrac{\tan\alpha_0 - \tan\alpha + 10^{-10}}{\tan\alpha_0 + \tan\alpha + 10^{-10}}$

FC5.4.1.6.7.6 Post-Processing for Random Setups with Elevation Rendering

After the parameters of $\mathbf{S}^{\mathrm{P}}$, $\mathbf{D}^{\mathrm{P}}$, $\mathbf{G}^{\mathrm{P}}{}_{\mathrm{H}}$, $\mathbf{G}^{\mathrm{P}}{}_{\mathrm{L}}$, and $\mathbf{E}^{\mathrm{P}}$ are initialized based on channel information, they are modified according to the azimuth and elevation deviations. For the convenience, $C^{\mathrm{P}}{}_{ii}$ refers the label of $S^{\mathrm{P}}{}_{i}$, $C^{\mathrm{P}}{}_{io}$ refers the label of $D^{\mathrm{P}}{}_{i}$, r_{ele,D_i}refers the elevation deviation of the $C^{\mathrm{P}}{}_{io}$, and r_{azi,D_i}refers the azimuth deviation of the $C^{\mathrm{P}}{}_{io}$,

(1) Elevation Post-Processing 1 : Find the “practically identical” channel

```
For each element $i$ in $\mathbf{S}^{\mathrm{P}}$, do
    $flag(i)=0$
For each element $i$ in $\mathbf{S}^{\mathrm{P}}$, do
    If $r_{ele,D_i}$ > 20 and $r_{azi,D_i} \le$ 15
        If ( $C^{\mathrm{P}}{}_{io}$ == CH_M_L030 and ( $C^{\mathrm{P}}{}_{ii}$ == CH_U_L030 || $C^{\mathrm{P}}{}_{ii}$ == CH_M_L045 ) ) ||
          ( $C^{\mathrm{P}}{}_{io}$ == CH_M_R030 and ( $C^{\mathrm{P}}{}_{ii}$ == CH_U_R030 || $C^{\mathrm{P}}{}_{ii}$ == CH_M_R045 ) ) ||
          ( $C^{\mathrm{P}}{}_{io}$ == CH_M_000 and $C^{\mathrm{P}}{}_{ii}$ == CH_U_000 ) ||
          {
              $G^{\mathrm{P}}{}_{iH}$ = 1
              $G^{\mathrm{P}}{}_{iL}$ = 1
              $flag(i)$= 1
              For each element $j$ in $\mathbf{S}^{\mathrm{P}}$, do
                  If $C^{\mathrm{P}}{}_{ii}$ == $C^{\mathrm{P}}{}_{ji}$ && $i \neq j$
                      $G^{\mathrm{P}}{}_{jH}$ = 0
                      $G^{\mathrm{P}}{}_{jL}$ = 0
                      $flag(j)$= 1
          }
    If $r_{ele,D_i}$ > 20 and $r_{azi,D_i} \le$ 25
        If ( $C^{\mathrm{P}}{}_{io}$ == CH_M_L110 and ( $C^{\mathrm{P}}{}_{ii}$ == CH_U_L110 || $C^{\mathrm{P}}{}_{ii}$ == CH_M_L135 ) ) ||
          ( $C^{\mathrm{P}}{}_{io}$ == CH_M_R110 and ( $C^{\mathrm{P}}{}_{ii}$ == CH_U_R110 || $C^{\mathrm{P}}{}_{ii}$ == CH_M_R135 ) )
          {
              $G^{\mathrm{P}}{}_{iH}$ = 1
              $G^{\mathrm{P}}{}_{iL}$ = 1
              $flag(i)$= 1
              For each element $j$ in $\mathbf{S}^{\mathrm{P}}$, do
                  If $C^{\mathrm{P}}{}_{ii}$ == $C^{\mathrm{P}}{}_{ji}$ && $i \neq j$
                  $G^{\mathrm{P}}{}_{jH}$ = 0
                  $G^{\mathrm{P}}{}_{jL}$ = 0
                  $flag(j)$= 1
          }
    For each element $i$ in $\mathbf{S}^{\mathrm{P}}$, do
        If $r_{ele,D_i}$ > 20 and $C^{\mathrm{P}}{}_{io}$ == CH_M_L110 and $C^{\mathrm{P}}{}_{ii}$ == CH_U_L090
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
            For each element $j$ in $\mathbf{S}^{\mathrm{P}}$, do
                **If** $r_{ele,D_j}$ > 20 and $C^{\mathrm{P}}{}_{jo}$ == CH_M_L030 and $C^{\mathrm{P}}{}_{ii}$ == CH_U_L090
                    For each element $k$ in $\mathbf{S}^{\mathrm{P}}$, do
                        **If** $C^{\mathrm{P}}{}_{ki}$ == CH_U_L090
                            $G^{\mathrm{P}}{}_{kH}$ = 0
                            $G^{\mathrm{P}}{}_{kL}$ = 0
                            $flag(k)$= 1
                    $G^{\mathrm{P}}{}_{iH}$ = $g_1$
                    $G^{\mathrm{P}}{}_{iL}$ = $g_1$
                    $G^{\mathrm{P}}{}_{jH}$ = $g_2$
                    $G^{\mathrm{P}}{}_{jL}$ = $g_2$
                    $flag(i)$= 1
                    $flag(j)$= 1
        **If** $r_{ele,D_i}$ > 20 and $C^{\mathrm{P}}{}_{io}$ == CH_M_R110 and $C^{\mathrm{P}}{}_{ii}$ == CH_U_R090
            For each element $j$ in $\mathbf{S}^{\mathrm{P}}$, do
                **If** $r_{ele,D_j}$ > 20 and $C^{\mathrm{P}}{}_{jo}$ == CH_M_R030 and $C^{\mathrm{P}}{}_{ii}$ == CH_U_R090
                    For each element $k$ in $\mathbf{S}^{\mathrm{P}}$, do
                        **If** $C^{\mathrm{P}}{}_{ki}$ == CH_U_R090
                            $G^{\mathrm{P}}{}_{kH}$ = 0
                            $G^{\mathrm{P}}{}_{kL}$ = 0
                            $flag(k)$= 1
                    $G^{\mathrm{P}}{}_{iH}$ = $g_1$
                    $G^{\mathrm{P}}{}_{iL}$ = $g_1$
                    $G^{\mathrm{P}}{}_{jH}$ = $g_2$
                    $G^{\mathrm{P}}{}_{jL}$ = $g_2$
                    $flag(i)$= 1
                    $flag(j)$= 1
```

The gains g_1 and g_2 are computed by applying tangent law amplitude panning in the following way:

— Unwrap source destination channel azimuth angles to be positive.

— The azimuth with the deviation for $C^{\mathrm{P}}{}_{io}$ and $C^{\mathrm{P}}{}_{jo}$ are α_1 and α_2

— The azimuth angle of the source channel (= panning target) is α_{src}.

— $\alpha_0 = \dfrac{|\alpha_1 - \alpha_2|}{2}$

— $\alpha_{\mathrm{center}} = \dfrac{\alpha_1 + \alpha_2}{2}$

— $\alpha = (\alpha_{\mathrm{center}} - \alpha_{\mathrm{src}}) \cdot \mathrm{sgn}(\alpha_2 - \alpha_1)$

— $g_1 = \dfrac{g}{\sqrt{1+g^2}}, \quad g_2 = \dfrac{1}{\sqrt{1+g^2}} \qquad \text{with} \qquad g = \dfrac{\tan\alpha_0 - \tan\alpha + 10^{-10}}{\tan\alpha_0 + \tan\alpha + 10^{-10}}$

(2) Elevation Post-Processing 2 on panning coefficients : Find the “practically dual mono” channel

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
For each element $i$ in $\mathbf{S}^{\mathrm{P}}$, if $flag(i)==0$

    If both CH_M_L030 and CH_M_R030 have elevation deviations more than 20 degree
        if $C^{\mathrm{P}}_{ii}$ == CH_U_000
            if $C^{\mathrm{P}}_{io}$ == CH_M_L030 || CH_M_R030
                $G^{\mathrm{P}}_{iH} = 1$
                $G^{\mathrm{P}}_{iL} = 1$
                $flag(i)=1$
            else
                $G^{\mathrm{P}}_{iH} = 0$
                $G^{\mathrm{P}}_{iL} = 0$
                $flag(i)=1$
    If both CH_M_L110 and CH_M_R110 are elevated
        if $C^{\mathrm{P}}_{ii}$ == CH_U_180
            if $C^{\mathrm{P}}_{io}$ == CH_M_L110 || CH_M_R110
                $G^{\mathrm{P}}_{iH} = 1$
                $G^{\mathrm{P}}_{iL} = 1$
                $flag(i)=1$
            else
                $G^{\mathrm{P}}_{iH} = 0$
                $G^{\mathrm{P}}_{iL} = 0$
                $flag(i)=1$
```

(3) Elevation Post-Processing 3 on panning coefficients : Keep the central image

```
For each element $i$ in $\mathrm{S}^{\mathrm{P}}$, if $flag(i)==0$

    if $C^{\mathrm{P}}_{ii}$ == CH_U_000 || CH_T_000 || CH_U_180
        if only one of the output channels of CH_M_L030 or CH_M_R030 has an elevation deviation more
        than 20 degree, then
            if $C^{\mathrm{P}}_{ii}$ == CH_U_000 || CH_T_000
                if $C^{\mathrm{P}}_{io}$ == CH_M_L030 || CH_M_R030
                    if $C^{\mathrm{P}}_{io}$ is elevated
                        $G^{\mathrm{P}}_{iH} = G^{\mathrm{P}}_{iH} \times 10^{\frac{3}{20}\times\frac{r_{ele,D_i}}{35}}$
                        if $C^{\mathrm{P}}_{ii}$ == CH_U_000
                            $G^{\mathrm{P}}_{iL} = G^{\mathrm{P}}_{iL} \times 10^{\frac{2}{20}\times\frac{r_{ele,D_i}}{35}}$
                        if $C^{\mathrm{P}}_{ii}$ == CH_T_000
                            $G^{\mathrm{P}}_{iL} = G^{\mathrm{P}}_{iL} \times 10^{\frac{3}{20}\times\frac{r_{ele,D_i}}{35}}$
                        $flag(i)=1$
                    else
                        $G^{\mathrm{P}}_{iH} = G^{\mathrm{P}}_{iH} \times 10^{\frac{2}{20}\times\frac{r_{ele,D_i}}{35}}$
                        $G^{\mathrm{P}}_{iL} = G^{\mathrm{P}}_{iL} \times 10^{\frac{2}{20}\times\frac{r_{ele,D_i}}{35}}$
                        $flag(i)=1$
                if $C^{\mathrm{P}}_{io}$ == CH_M_000
                        $G^{\mathrm{P}}_{iH} = G^{\mathrm{P}}_{iH} \times 10^{\frac{2}{20}\times\frac{r_{ele,D_i}}{35}}$
                        $G^{\mathrm{P}}_{iL} = G^{\mathrm{P}}_{iL} \times 10^{\frac{2}{20}\times\frac{r_{ele,D_i}}{35}}$
                        $flag(i)=1$
            if $C^{\mathrm{P}}_{ii}$ == CH_U_180
                if $C^{\mathrm{P}}_{io}$ == CH_M_L030 || CH_M_R030
                    if $C^{\mathrm{P}}_{io}$ is not elevated
                        $G^{\mathrm{P}}_{iH} = G^{\mathrm{P}}_{iH} \times 10^{-\frac{2}{20}\times\frac{r_{ele,D_i}}{35}}$
                        $G^{\mathrm{P}}_{iL} = G^{\mathrm{P}}_{iL} \times 10^{-\frac{2}{20}\times\frac{r_{ele,D_i}}{35}}$
                        $flag(i)=1$
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
if only one of the output channels of CH_M_L110 or CH_M_R110 has an elevation deviation more than 20 degree, then
        if $C^{P}_{ii}$ == CH_U_180 || CH_T_000
                if $C^{P}_{io}$ == CH_M_L110 || CH_M_R110
                        if $C^{P}_{io}$ is elevated
                            $G^{P}_{iH} = G^{P}_{iH} \times 10^{\frac{2}{20}\times\frac{r_{ele,D_i}}{35}}$
                            $G^{P}_{iL} = G^{P}_{iL} \times 10^{\frac{2}{20}\times\frac{r_{ele,D_i}}{35}}$
                            $flag(i)=1$
        if $C^{P}_{ii}$ == CH_U_000
                if $C^{P}_{io}$ == CH_M_L110 || CH_M_R110
                        if $C^{P}_{io}$ is not elevated
                            $G^{P}_{iH} = G^{P}_{iH} \times 10^{-\frac{2}{20}\times\frac{r_{ele,D_i}}{35}}$
                            $G^{P}_{iL} = G^{P}_{iL} \times 10^{-\frac{2}{20}\times\frac{r_{ele,D_i}}{35}}$
                            $flag(i)=1$
```

(4) Elevation Post-Processing 4 on panning coefficients : Keep the L/R balance when the Contralateral Frontal Channel Elevated

```
For each element $i$ in $\mathbf{S}^{P}$, if $flag(i)==0$
    if CH_M_L030 is elevated more than 20 degree and CH_M_R030 is not
      if $C^{P}_{ii}$ == CH_U_R030 || $C^{P}_{ii}$ == CH_U_R045, do
        if $C^{P}_{io}$ == CH_M_L030
            $G^{P}_{iH} = G^{P}_{iH} \times 10^{-\frac{1}{20}\times\frac{r_{ele,D_i}}{35}}$
            $G^{P}_{iL} = G^{P}_{iL} \times 10^{-\frac{1}{20}\times\frac{r_{ele,D_i}}{35}}$
            $flag(i)=1$
        elseif $C^{P}_{io}$ == CH_M_L110
            $G^{P}_{iH} = G^{P}_{iH} \times 10^{-\frac{4}{20}\times\frac{r_{ele,D_i}}{35}}$
            $G^{P}_{iL} = G^{P}_{iL} \times 10^{-\frac{4}{20}\times\frac{r_{ele,D_i}}{35}}$
            $flag(i)=1$
      elseif $C^{P}_{ii}$ == CH_U_L090, do
        if $C^{P}_{io}$ == CH_M_L030
            $G^{P}_{iH} = G^{P}_{iH} \times 10^{\frac{8.7}{20}\times\frac{r_{ele,D_i}}{35}}$
            $flag(i)=1$
        elseif $C^{P}_{io}$ == CH_M_L110
            $G^{P}_{iH} = G^{P}_{iH} \times 10^{-\frac{4}{20}\times\frac{r_{ele,D_i}}{35}}$
            $flag(i)=1$
        elseif $C^{P}_{io}$ == CH_M_R110
            $G^{P}_{iH} = 0$
            $flag(i)=1$
      elseif $C^{P}_{ii}$ == CH_U_L110 || $C^{P}_{ii}$ == CH_U_L135, do
        if $C^{P}_{io}$ == CH_M_L030
            $G^{P}_{iH} = G^{P}_{iH} \times 10^{\frac{5.6}{20}\times\frac{r_{ele,D_i}}{35}}$
            $G^{P}_{iL} = G^{P}_{iL} \times 10^{\frac{5.6}{20}\times\frac{r_{ele,D_i}}{35}}$
            $flag(i)=1$
        elseif $C^{P}_{io}$ == CH_M_L110
            $G^{P}_{iH} = G^{P}_{iH} \times 10^{-\frac{4.6}{20}\times\frac{r_{ele,D_i}}{35}}$
            $G^{P}_{iL} = G^{P}_{iL} \times 10^{-\frac{4,6}{20}\times\frac{r_{ele,D_i}}{35}}$
            $flag(i)=1$
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
            elseif $C^{P}{}_{io}$ == CH_M_R110
                $G^{P}{}_{iH}$ = $G^{P}{}_{iH}$ x $10^{-\frac{1}{20}\times\frac{r_{ele,D_i}}{35}}$
                $G^{P}{}_{iL}$ = $G^{P}{}_{iL}$ x $10^{-\frac{1}{20}\times\frac{r_{ele,D_i}}{35}}$
                $flag(i)=1$
    elseif CH_M_R030 is elevated more than 20 degree and CH_M_L030 is not
        if $C^{P}{}_{ii}$ == CH_U_L030 || $C^{P}{}_{ii}$ == CH_U_L045, do
            if $C^{P}{}_{io}$ == CH_M_R030
                $G^{P}{}_{iH}$ = $G^{P}{}_{iH}$ x $10^{-\frac{1}{20}\times\frac{r_{ele,D_i}}{35}}$
                $G^{P}{}_{iL}$ = $G^{P}{}_{iL}$ x $10^{-\frac{1}{20}\times\frac{r_{ele,D_i}}{35}}$
                $flag(i)=1$
            elseif $C^{P}{}_{io}$ == CH_M_R110
                $G^{P}{}_{iH}$ = $G^{P}{}_{iH}$ x $10^{-\frac{4}{20}\times\frac{r_{ele,D_i}}{35}}$
                $G^{P}{}_{iL}$ = $G^{P}{}_{iL}$ x $10^{-\frac{4}{20}\times\frac{r_{ele,D_i}}{35}}$
                $flag(i)=1$
        elseif $C^{P}{}_{ii}$ == CH_U_R090, do
            if $C^{P}{}_{io}$ == CH_M_R030
                $G^{P}{}_{iH}$ = $G^{P}{}_{iH}$ x $10^{\frac{8.7}{20}\times\frac{r_{ele,D_i}}{35}}$
                $flag(i)=1$
            elseif $C^{P}{}_{io}$ == CH_M_R110
                $G^{P}{}_{iH}$ = $G^{P}{}_{iH}$ x $10^{-\frac{4}{20}\times\frac{r_{ele,D_i}}{35}}$
                $flag(i)$ =1
            elseif $C^{P}{}_{io}$ == CH_M_R110
                $G^{P}{}_{iH}$ = 0
                $flag(i)=1$
        elseif $C^{P}{}_{ii}$ == CH_U_R110 || $C^{P}{}_{ii}$ == CH_U_R135, do
            if $C^{P}{}_{io}$ == CH_M_R030
                $G^{P}{}_{iH}$ = $G^{P}{}_{iH}$ x $10^{\frac{5.6}{20}\times\frac{r_{ele,D_i}}{35}}$
                $G^{P}{}_{iL}$ = $G^{P}{}_{iL}$ x $10^{\frac{5.6}{20}\times\frac{r_{ele,D_i}}{35}}$
                $flag(i)=1$
            elseif $C^{P}{}_{io}$ == CH_M_R110
                $G^{P}{}_{iH}$ = $G^{P}{}_{iH}$ x $10^{-\frac{4.6}{20}\times\frac{r_{ele,D_i}}{35}}$
                $G^{P}{}_{iL}$ = $G^{P}{}_{iL}$ x $10^{-\frac{4,6}{20}\times\frac{r_{ele,D_i}}{35}}$
                $flag(i)=1$
            elseif $C^{P}{}_{io}$ == CH_M_L110
                $G^{P}{}_{iH}$ = $G^{P}{}_{iH}$ x $10^{-\frac{1}{20}\times\frac{r_{ele,D_i}}{35}}$
                $G^{P}{}_{iL}$ = $G^{P}{}_{iL}$ x $10^{-\frac{1}{20}\times\frac{r_{ele,D_i}}{35}}$
                $flag(i)=1$
```

(5) Azimuth Post-Processing on panning coefficients

```
    $fbias$ = $r_{azi,A}$(CH_M_L030)+ $r_{azi,A}$(CH_M_R030)
    $bbias$ = $r_{azi,A}$(CH_M_L110)+ $r_{azi,A}$(CH_M_R110)
    For each element $i$ in $S^{P}$,
    If $fbias > 10$
        If ( $C^{P}{}_{ii}$ == CH_U_000 || $C^{P}{}_{ii}$ == CH_T_000 ) && ( $C^{P}{}_{io}$ == CH_M_L030 )
                    $G^{P}{}_{iH}$ = $G^{P}{}_{iH}$ x $10^{-\frac{1}{20}\times\frac{(fbias)}{15}}$
                    If $C^{P}{}_{ii}$ == CH_U_000
                        $G^{P}{}_{iL}$ = $G^{P}{}_{iL}$ x $10^{-\frac{2}{20}\times\frac{(fbias)}{15}}$
                    elseif $C^{P}{}_{ii}$ == CH_T_000
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

```
                $G^{P}_{iL} = G^{P}_{iL} \times 10^{-\frac{1}{20}\times\frac{(fbias)}{15}}$
            $flag(i)=1$
If $fbias < -10$
    If ( $C^{P}_{ii}$ == CH_U_000 || $C^{P}_{ii}$ == CH_T_000 ) && ( $C^{P}_{io}$ == CH_M_R030 )
            $G^{P}_{iH} = G^{P}_{iH} \times 10^{\frac{1}{20}\times\frac{(fbias)}{15}}$
            If $C^{P}_{ii}$ == CH_U_000
                $G^{P}_{iL} = G^{P}_{iL} \times 10^{\frac{2}{20}\times\frac{(fbias)}{15}}$
            elseif $C^{P}_{ii}$ == CH_T_000
                $G^{P}_{iL} = G^{P}_{iL} \times 10^{\frac{1}{20}\times\frac{(fbias)}{15}}$

            $flag(i)=1$
If $bbias > 10$
    If ( $C^{P}_{ii}$ == CH_U_180 || $C^{P}_{ii}$ == CH_T_000 ) && ( $C^{P}_{io}$ == CH_M_L110 )
            $G^{P}_{iH} = G^{P}_{iH} \times 10^{\frac{1}{20}\times\frac{(bbias)}{15}}$
            $G^{P}_{iL} = G^{P}_{iL} \times 10^{\frac{1}{20}\times\frac{(bbias)}{15}}$
            $flag(i)=1$
If $bbias < -10$
    If ( $C^{P}_{ii}$ == CH_U_180 || $C^{P}_{ii}$ == CH_T_000 ) && ( $C^{P}_{io}$ == CH_M_R110 )
            $G^{P}_{iH} = G^{P}_{iH} \times 10^{-\frac{1}{20}\times\frac{(bbias)}{15}}$
            $G^{P}_{iL} = G^{P}_{iL} \times 10^{-\frac{1}{20}\times\frac{(bbias)}{15}}$
            $flag(i)=1$
```

(6) Spatial elevation coefficient normalization

```
For each element $i$ in $\mathbf{S}^{P}$, if $flag(i)==1$
    $P_{iL} = 0$
    $P_{iH} = 0$
For each element $i$ in $\mathbf{S}^{P}$, if $flag(i)==1$
    For each element $k$ in $\mathbf{S}^{P}$
        If $C^{P}_{ii} == C^{P}_{ki}$
            $P_{iL} = P_{iL} + (G^{P}_{iL})^{2}$
            $P_{iH} = P_{iH} + (G^{P}_{iH})^{2}$
            $flag(i)==0$
    For each element $k$ in $\mathbf{S}^{P}$
        If $C^{P}_{ii} == C^{P}_{ki}$
            $G^{P}_{iL} = G^{P}_{iL} \times \sqrt{P_{iL}}$
            $G^{P}_{iH} = G^{P}_{iH} \times \sqrt{P_{iH}}$
            $flag(i)==0$
```

(7) Elevation Post-Processing on spatial elevation filters

```
For each element $i$ in $\mathbf{S}^{P}$

    If $C^{P}_{io}$ is elevated ($r_{ele,A}(C^{P}_{io}) > 20$)

        $E^{P}_{i}$ = 0 (no EQ)
```

(8) Update the $\mathbf{E}^{S}$, $\mathbf{G}_{EQ}$ by same process defined in FC5.4.1.6.5 as below :

For each element *i* in D^{S}_{i}, do:

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved

```
if output channel with index $D^S_i$ is a horizontal channel by definition (i.e. output channel label contains the label '_M_'), and
    if this output channel is now a height channel (elevation in range 0..60 degrees), and
        if input channel with index $S^S_i$ is a height channel (i.e. label contains '_U_'), then
            — $h$ = min(elevation of randomized output channel, 35) / 35
            — $G_{comp} = h \cdot \frac{1}{Gain_i^S} + (1 - h)$
            — Apply compensation gain to DMX gain: $G^S_i = G^S_i \times G_{comp}$
            — Define new equalizer $\mathbf{G}_{\mathrm{EQ},e}$ with the index $e$, where $G^{pb}_{EQ,e} = h + (1-h) \cdot \mathrm{G}^{pb}_{EQ,E_i}$
            — $E^S_i = e$
            — $e = e + 1$
        else if input channel with index $S^S_i$ is a horizontal channel (label contains '_M_')
            — $h$ = min(elevation of randomized output channel, 35) / 35
            — Define new equalizer $\mathbf{G}_{\mathrm{EQ},e}$ with the index $e$,
              where $G^{pb}_{EQ,e} = h \cdot \mathrm{G}^{pb}_{EQ,5} + (1-h) \cdot \mathrm{G}^{pb}_{EQ,E_i}$
            — $E^S_i = e$
            — $e = e + 1$
```

FC5.4.1.6.7.7 Merge general downmix rules and elevation rules

After the elevation rendering parameters are initialized, the vectors of $\mathbf{S}^\mathrm{P}$, $\mathbf{D}^\mathrm{P}$, $\mathbf{G}^\mathrm{P}{}_\mathrm{H}$, $\mathbf{G}^\mathrm{P}{}_\mathrm{L}$, and $\mathbf{E}^\mathrm{P}$ that covers elevation rendered height input channel shall be merged with $\mathbf{S}$, $\mathbf{D}$, $\mathbf{G}$, $\mathbf{E}$, and $\mathbf{G}_\mathrm{EQ}$ that covers the rest input channels by

$$\mathbf{S}^\mathrm{P} = \begin{bmatrix}\mathbf{S}^\mathrm{P}\\ \mathbf{S}\end{bmatrix},\quad \mathbf{D}^\mathrm{P} = \begin{bmatrix}\mathbf{D}^\mathrm{P}\\ \mathbf{D}\end{bmatrix},\quad \mathbf{G}^\mathrm{P}{}_\mathrm{H} = \begin{bmatrix}\mathbf{G}^\mathrm{P}{}_\mathrm{H}\\ \mathbf{G}\end{bmatrix},\quad \mathbf{G}^\mathrm{P}{}_\mathrm{L} = \begin{bmatrix}\mathbf{G}^\mathrm{P}{}_\mathrm{L}\\ \mathbf{G}\end{bmatrix},\quad \mathbf{E}^\mathrm{P} = \begin{bmatrix}\mathbf{E}^\mathrm{P}\\ \mathbf{E}\end{bmatrix}$$

$\mathbf{S}^\mathrm{S}$, $\mathbf{D}^\mathrm{S}$, $\mathbf{G}^\mathrm{S}$, and $\mathbf{E}^\mathrm{S}$ are also merged with $\mathbf{S}$, $\mathbf{D}$, $\mathbf{G}$, and $\mathbf{E}$ by

$$\mathbf{S}^\mathrm{S} = \begin{bmatrix}\mathbf{S}^\mathrm{S}\\ \mathbf{S}\end{bmatrix},\quad \mathbf{D}^\mathrm{S} = \begin{bmatrix}\mathbf{D}^\mathrm{S}\\ \mathbf{D}\end{bmatrix},\quad \mathbf{G}^\mathrm{S} = \begin{bmatrix}\mathbf{G}^\mathrm{S}\\ \mathbf{G}\end{bmatrix},\quad \mathbf{E}^\mathrm{S} = \begin{bmatrix}\mathbf{E}^\mathrm{S}\\ \mathbf{E}\end{bmatrix},$$

When no input channel is rendered by elevation rendering, the vectors of $\mathbf{S}^\mathrm{P}$, $\mathbf{D}^\mathrm{P}$, $\mathbf{G}^\mathrm{P}{}_\mathrm{H}$ (or $\mathbf{G}^\mathrm{P}{}_\mathrm{L}$), and $\mathbf{E}^\mathrm{P}$ are identical with $\mathbf{S}^\mathrm{S}$, $\mathbf{D}^\mathrm{S}$, $\mathbf{G}^\mathrm{S}$, and $\mathbf{E}^\mathrm{S}$ and also with $\mathbf{S}$, $\mathbf{D}$, $\mathbf{G}$, and $\mathbf{E}$.

FC5.4.1.6.8 Derivation of rules-based initialization downmix matrix:

$\mathbf{M}_\mathrm{DMX}$ and $\mathbf{M}_\mathrm{DMX2}$ are derived by rearranging the temporary parameters from the mapping-oriented representation (enumerated by mapping counter *i*) to a channel-oriented representation as defined in the following:

Initialize $\mathbf{M}_\mathrm{DMX}$ and $\mathbf{M}_\mathrm{DMX2}$ as N_out x N_in zero matrixes for all STFT bins *k*.

```
For each $i$ in $\mathbf{S}^\mathrm{P}$ do:
    If ($E^\mathrm{P}{}_i$ = 0)
        $M^k_{DMX,A,B} = G^P{}_{iL}$ for $A = D^\mathrm{P}{}_i$, $B = S^\mathrm{P}{}_i$,    for $k$ such that $f_{pb=pbm(k)} \times f_\mathrm{s} / 2 < 2800$
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

$M^k_{DMX,A,B} = G^P{}_{iH}$ for $A = D^P{}_i$, $B = S^P{}_i$, for k such that $2800 \le f_{pb=pbm(k)} \times f_s / 2 \le 10000$

$M^k_{DMX,A,B} = G^P{}_{iL}$ for $A = D^P{}_i$, $B = S^P{}_i$, for k such that $f_{pb=pbm(k)} \times f_s / 2 > 10000$

Else

$M^k_{DMX,A,B} = G^P{}_{iL} \times G^k{}_{EQ,E^P_i}$ for $A = D^P{}_i$, $B = S^P{}_i$, for k such that $f_{pb=pbm(k)} \times f_s / 2 < 2800$

$M^k_{DMX,A,B} = G^P{}_{iH} \times G^k{}_{EQ,E^P_i}$ for $A = D^P{}_i$, $B = S^P{}_i$, for k such that $2800 \le f_{pb=pbm(k)} \times f_s / 2 \le 10000$

$M^k_{DMX,A,B} = G^P{}_{iL} \times G^k{}_{EQ,E^P_i}$ for $A = D^P{}_i$, $B = S^P{}_i$, for k such that $f_{pb=pbm(k)} \times f_s / 2 > 10000$

For each i in $\mathbf{S}^S$ do:

If ($E^S{}_i = 0$)

$M^k_{DMX2,A,B} = G^S{}_i$ for $A = D^S{}_i$, $B = S^S{}_i$, for $0 \le k < K$

Else

$$M^k_{\mathrm{DMX2},A,B} = G^S_i \cdot G^{pb=pbm(k)}_{EQ,E^S_i} \quad \text{for} \quad A = D^S_i,\ B = S^S_i,\ 0 \le k < K$$

where f_{pb} is the normalized center frequency of processing band *pb*, specified in Table FC5.15, *pbm* defines the processing band to STFT frequency bin mapping as specified in Table FC5.16, f_s is the sampling frequency, $M^k_{DMX,A,B}$ and $M^k_{DMX2,A,B}$ denotes the matrix element in the Ath row and Bth column of $\mathbf{M}_{\mathrm{DMX}}$ and $\mathbf{M}_{\mathrm{DMX2}}$. Note that after the rules-based initialization this matrix of downmix coefficients will contain columns of zeros, if unknown channels are present in the input format. Those columns are filled with downmix gains as specified in FC5.4.1.6.9

FC5.4.1.6.9 VBAP-based downmix coefficients derivation

Handling of unknown output formats:

In case the output format is considered unknown, the downmix coefficients for all input channels shall be derived as follows:

Each channel of the input setup is regarded as a static audio object at the position defined by the azimuth and elevation angles associated with the input channel. For each input channel the mixing gains to all output loudspeakers are calculated as VBAP panning gains $\mathbf{g}_{\mathrm{scaled}}$ according to 8.4.3, where the same output format shall be signaled to the VBAP algorithm as to the format converter. The panning gain vectors $\mathbf{g}_{\mathrm{scaled}}$ shall be post-processed according to FC5.4.1.6.10.

The downmix matrix $\mathbf{M}^k_{\mathrm{DMX}}$ is finally derived by filling each matrix column with the post-processed panning gain vector elements of the corresponding input channel, independently of the STFT bin index k.

Handling of unknown input channels:

In case the input format contains unknown input channels, the downmix coefficients for these channels shall be derived as follows:

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 23008-3:2015/Amd 3:2017

© ISO/IEC 2017 – All rights reserved