INTERNATIONAL STANDARD ISO/IEC 14496-3:2005/Amd.2:2006
TECHNICAL CORRIGENDUM 2

Published 2008-12-01

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION • МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ • ORGANISATION INTERNATIONALE DE NORMALISATION
INTERNATIONAL ELECTROTECHNICAL COMMISSION • МЕЖДУНАРОДНАЯ ЭЛЕКТРОТЕХНИЧЕСКАЯ КОМИССИЯ • COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

Information technology — Coding of audio-visual objects —
Part 3:
Audio

AMENDMENT 2: Audio Lossless Coding (ALS), new audio profiles and BSAC extensions

TECHNICAL CORRIGENDUM 2

Technologies de l'information — Codage des objets audiovisuels —

Partie 3: Codage audio

AMENDEMENT 2: Codage audio sans perte (ALS), nouveaux profils audio et extensions BSAC

RECTIFICATIF TECHNIQUE 2

Technical Corrigendum 2 to ISO/IEC 14496-3:2005/Amd.2:2006 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

Throughout this Corrigendum, modifications or additions to existing text are highlighted in grey.

Page 4, after Table 1.11A, replace:

A HE AAC v2 Profile decoder of a certain level shall operate the HQ SBR tool for streams containing Parametric Stereo data. For streams not containing Parametric Stereo data, the HE AAC v2 Profile decoder may operate the HQ SBR tool, or the LP SBR tool.

ICS 35.040 **Ref. No. ISO/IEC 14496-3:2005/Amd.2:2006/Cor.2:2008(E)**

© ISO/IEC 2008 – All rights reserved

Published in Switzerland

Documents
Document Sharing Hub

Full document details: ISO_IEC 14496-3_2005_AMD2_2006_COR2_2008

with:

An HE AAC v2 Profile decoder shall operate the HQ SBR tool for bitstreams containing Parametric Stereo data. For bitstreams not containing Parametric Stereo data, the HE AAC v2 Profile decoder may operate the HQ SBR tool, or the LP SBR tool.
Only bitstreams consisting of exactly one AAC single channel element may contain Parametric Stereo data. Bitstreams containing more than one channel in the AAC part shall not contain Parametric Stereo data.

Page 6, in Table 1.13 replace the rows:

```
    case 36:
        ALSSpecificConfig();
        break;
```

with:

```
    case 36:
        fillBits;                                   5       bslbf
        ALSSpecificConfig();
        break;
```

In ISO/IEC 14496-3:2005, after 1.6.3.14 add:

1.6.3.15 FillBits

Fill bits for byte alignment of ALSSpecificConfig() relative to the start of AudioSpecificConfig().

Page 8, replace the final sentence:

Similarly the HE AAC v2 decoder can handle all HE AAC Profile streams as well as all AAC Profile streams.

with:

Similarly an HE AAC v2 profile decoder of a certain level can handle all HE AAC Profile streams of the same or lower level as well as all AAC Profile streams of the same or lower level.

In 11.2.1, replace:

(see subclause 11.6.7)

with:

(see subclause 11.6.5.2.1 and 11.6.7)

Page 18, at the end of 11.3.3 add:

x.y Fixed-point signed fractional representation, where x is the number of bits to the left of the binary point, and y is the number of bits to the right of the binary point (two's complement sign representation). 64-bit signed integer (two's complement)

Page 18, replace Table 11.1 with the following table:

© ISO/IEC 2008 – All rights reserved
Documents
Document Sharing Hub
Full document details: ISO_IEC 14496-3_2005_AMD2_2006_COR2_2008

Table 11.1 — Syntax of ALSSpecificConfig

Syntax	No. of bits	Mnemonic
ALSSpecificConfig()		
{		
als_id;	**32**	**uimsbf**
samp_freq;	**32**	**uimsbf**
samples;	**32**	**uimsbf**
channels;	**16**	**uimsbf**
file_type;	**3**	**uimsbf**
resolution;	**3**	**uimsbf**
floating;	**1**	**uimsbf**
msb_first;	**1**	**uimsbf**
frame_length;	**16**	**uimsbf**
random_access;	**8**	**uimsbf**
ra_flag;	**2**	**uimsbf**
adapt_order;	**1**	**uimsbf**
coef_table;	**2**	**uimsbf**
long_term_prediction;	**1**	**uimsbf**
max_order;	**10**	**uimsbf**
block_switching;	**2**	**uimsbf**
bgmc_mode;	**1**	**uimsbf**
sb_part;	**1**	**uimsbf**
joint_stereo;	**1**	**uimsbf**
mc_coding;	**1**	**uimsbf**
chan_config;	**1**	**uimsbf**
chan_sort;	**1**	**uimsbf**
crc_enabled;	**1**	**uimsbf**
RLSLMS	**1**	**uimsbf**
(reserved)	**5**	
aux_data_enabled;	**1**	**uimsbf**
if (chan_config) {		
chan_config_info;	**16**	**uimsbf**
}		
if (chan_sort) {		
for (c = 0; c < channels; c++)		
chan_pos[c];	**1..16**	**uimsbf**
}		
byte_align;	**0..7**	**bslbf**
header_size;	**32**	**uimsbf**
trailer_size;	**32**	**uimsbf**
orig_header[];	**header_size * 8**	**bslbf**
orig_trailer[];	**trailer_size * 8**	**bslbf**
if (crc_enabled) {		
crc;	**32**	**uimsbf**
}		
if ((ra_flag == 2) && (random_access > 0)) {		
for (f = 0; f < ((samples-1) / (frame_length+1)) + 1; f++) {		
ra_unit_size[f]	**32**	**uimsbf**
}		
}		
if (aux_data_enabled) {		
aux_size;	**32**	**uimsbf**
aux_data[];	**aux_size * 8**	**bslbf**
}		
}		

Note: “byte_align” denotes byte alignment of subsequent data relative to the start of ALSSpecificConfig().

© ISO/IEC 2008 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14496-3_2005_AMD2_2006_COR2_2008

Page 21, in Table 11.3, replace:

if (adapt_order == 1) {
opt_order;
}
for (p = 0; p < opt_order; p++) {
quant_cof[p];
}
	1..10
	
	varies
		uimsbf
		
		Rice code

if (RLSLMS) {
RLSLMS_extension_data()
}
}		

with:

if (adapt_order == 1) {
opt_order;
}
else {
opt_order = max_order;
}
for (p = 0; p < opt_order; p++) {
quant_cof[p];
}
	1..10
	
	varies
		uimsbf
		
		Rice code

if (RLSLMS) {
RLSLMS_extension_data()
}
byte_align;
}
	
	0..7
		
		bslbf

© ISO/IEC 2008 – All rights reserved
Documents
Document Sharing Hub
Full document details: ISO_IEC 14496-3_2005_AMD2_2006_COR2_2008

Page 23, replace Table 11.5 with the following table:

Table 11.5 – Syntax of RLSLMS_extension_data

Syntax	No. of bits	Mnemonic
RLSLMS_extension() {		
mono_block	**1**	**uimsbf**
ext_mode	**1**	**uimsbf**
if (ext_mode) {		
extension_bits	**3**	**uimsbf**
if (extension_bits&0x01) {		
RLS_order	**4**	**uimsbf**
LMS_stage	**3**	**uimsbf**
for(i=0; i<LMS_stage;i++){		
LMS_order[i]	**5**	**uimsbf**
}		
}		
if (extension_bits&0x02) {		
if (RLS_order) {		
RLS_lambda	**10**	**uimsbf**
if (RA)		
RLS_lambda_ra	**10**	**uimsbf**
}		
}		
if (extension_bits&04) {		
for(i=0; i<LMS_stage;i++) {		
LMS_mu[i]	**5**	**uimsbf**
}		
LMS_stepsize	**3**	**uimsbf**
}		
}		
}		

Page 25, replace Table 11.9 with the following table:

Table 11.9 — Elements of ALSSpecificConfig

Field	#Bits	Description / Values
als_id	32	ALS identifier fixed value = 1095521024 = 0x414C5300 (Hex)
samp_freq	32	Sampling frequency in Hz
samples	32	Number of samples (per channel) If samples = 0xFFFFFFFF (Hex), the number of samples is not specified (see 11.6.1.3)
channels	16	Number of channels-1 (0 = mono, 1 = stereo, …)
file_type	3	000 = unknown / raw file 001 = wave file 010 = aiff file 011 = bwf file (other values are reserved)

© ISO/IEC 2008 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14496-3_2005_AMD2_2006_COR2_2008

resolution	3	000 = 8-bit 001 = 16-bit 010 = 24-bit 011 = 32-bit (other values are reserved)
floating	1	1 = IEEE 32-bit floating-point, 0 = integer
msb_first	1	Original byte order of the input audio data: 0 = least significant byte first (little-endian) 1 = most significant byte first (big-endian) If resolution = 0 (8-bit data), msb_first = 0 indicates unsigned data (0…255), while msb_first = 1 indicates signed data (-128…127).
frame_length	16	Frame Length - 1 (e.g. frame_length = 0x1FFF signals a frame length of N = 8192)
random_access	8	Distance between RA frames (in frames, 0…255). If no RA is used, the value is zero. If each frame is an RA frame, the value is 1.
ra_flag	2	Indicates where the size of random access units (ra_unit_size) is stored: 00: not stored 01: stored at the beginning of frame_data() 10: stored at the end of ALSSpecificConfig()
adapt_order	1	Adaptive Order: 1 = on, 0 = off
coef_table	2	Table index (00, 01, or 10, see Table 11.20) of Rice code parameters for entropy coding of predictor coefficients, 11 = no entropy coding
long_term_prediction	1	Long term prediction (LTP): 1 = on, 0 = off
max_order	10	Maximum prediction order (0..1023)
block_switching	2	Number of block switching levels: 00 = no block switching 01 = up to 3 levels 10 = 4 levels 11 = 5 levels
bgmc_mode	1	BGMC Mode: 1 = on, 0 = off (Rice coding only)

sb_part	1	Sub-block partition for entropy coding of the residual.
		if bgmc_mode = 0:
		0 = no partition, no ec_sub bit in block_data
		1 = 1:4 partition, one ec_sub bit in block_data
		if bgmc_mode = 1:
		0 = 1:4 partition, one ec_sub bit in block_data
		1 = 1:2:4:8 partition, two ec_sub bits in block_data

joint_stereo	1	Joint Stereo: 1 = on, 0 = off If channels = 0 (mono), joint_stereo = 0

Documents
Document Sharing Hub

© ISO/IEC 2008 – All rights reserved

Full document details: ISO_IEC 14496-3_2005_AMD2_2006_COR2_2008

mc_coding	1	Extended inter-channel coding: 1 = on, 0 = off If channels = 0 (mono), mc_coding = 0
chan_config	1	Indicates that a chan_config_info field is present
chan_sort	1	Channel rearrangement: 1 = on, 0 = off If channels = 0 (mono), chan_sort = 0
crc_enabled	1	Indicates that the crc field is present
RLSLMS	1	Use RLS-LMS predictor: 1 = on, 0 = off
(reserved)	5	
aux_data_enabled	1	Indicates that auxiliary data is present (fields aux_size and aux_data)
chan_config_info	16	Mapping of channels to loudspeaker locations. Each bit indicates whether a channel for a particular predefined location exists (see 11.6.1.5).
chan_pos[]	(channels+1)*ChBits	If channel rearrangement is on (chan_sort = 1), these are the original channel positions. The number of bits per channel is ChBits = ceil[log2(channels+1)] = 1..16 where channels+1 is the number of channels.
header_size	32	Header size of original audio file in bytes If header_size = 0xFFFFFFFF (Hex), there is no orig_header[] field, but the original header may be stored elsewhere, e.g. in the meta data of an MPEG-4 file.
trailer_size	32	Trailer size of original audio file in bytes If trailer_size = 0xFFFFFFFF (Hex), there is no orig_trailer[] field, but the original trailer may be stored elsewhere, e.g. in the meta data of an MPEG-4 file.
orig_header[]	header_size*8	Header of original audio file
orig_trailer[]	trailer_size*8	Trailer of original audio file
crc	32	32-bit CCITT-32 CRC checksum of the original audio data bytes (polynomial: $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$).
ra_unit_size[]	#frames*32	Distances (in bytes) between the random access frames, i.e. the sizes of the random access units, where the number of frames is #frames = ((samples-1) / (frame_length+1)) +1 In ALSSpecificConfig(), this field appears only if ra_flag = 2.
aux_size	32	Size of the aux_data field in bytes If aux_size = 0xFFFFFFFF (Hex), there is no aux_data[] field, but the auxiliary data may be stored elsewhere, e.g. in the meta data of an MPEG-4 file.
aux_data[]	aux_size*8	Auxiliary data (not required for decoding)

© ISO/IEC 2008 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14496-3_2005_AMD2_2006_COR2_2008

Page 28, in Table 11.10 replace:

ra_flag = 2

with:

ra_flag = 1

Page 30, replace Table 11.13 with the following table:

Table 11.13 — Elements of RLSLMS_extension_data

Field	#Bits	Description / Values
mono_block	1	mono_frame == 0: CPE coded with joint-stereo RLS mono_frame == 1: CPE coded with mono RLS
ext_mode	1	RLS-LMS predictor parameters are updated in extension block. 1 == extension block 0 == non-extension block
extension_bits	3	Type of RLS-LMS parameters carried in extension block xtension&01 == RLS-LMS predictors orders extension&02 == RLS_lambda and RLS_lambda_ra extension&04 == LMS_mu and LMS_stepsize
RLS_order	4	RLS predictor order
LMS_stage	3	Number of LMS predictors in cascade
LMS_order[]	5*LMS_stage	LMS predictor order
RLS_lambda	10	RLS predictor parameter lambda.
RLS_lambda_ra	10	RLS predictor parameter lambda for random access frame
LMS_mu[]	5*LMS_stage	LMS predictor parameter – NLMS stepsize
LMS_stepsize	3	Linear combiner parameter – Sign-Sign LMS stepsize

Page 33, 11.6.1.2, add at the top of the list of bullet items:

- ALS identifier: This field must contain the value 1095521024 = 0x414C5300 (Hex). Using byte-wise reading, the first three bytes are equivalent to the ASCII codes for 'ALS'.

© ISO/IEC 2008 – All rights reserved
Documents
Document Sharing Hub
Full document details: ISO_IEC 14496-3_2005_AMD2_2006_COR2_2008

Page 34, 11.6.1.3, replace the pseudo code with the following pseudo code:

```
N = frame_length + 1.
frames = samples / N;
remainder = samples % N;
if (remainder)
{
    frames++;
    N_last = remainder;
}
else
    N_last = N;
```

and add the following paragraphs at the end of the subclause:

If the value of samples is 0xFFFFFFFF (Hex), the number of samples is not specified. If the ALS payload is stored using the MPEG-4 file format, the number of samples can be obtained from the meta data of the file. In that case, the number of frames and the length of the last frame are determined as described above.

If the number of samples is not available, the number of frames is undefined, and all frames are assumed to have the same length N. In that case, the sizes of random access units shall not be stored in ALSSpecifcConfig (i.e. only ra_flag = 0 or ra_flag = 1 shall be used), since the number of random access units is undefined as well.

Page 37, 11.6.2, immediately before Table 11.18, add:

The frame length N must be divisible by 2^{levels} without remainder, in order to obtain integer block lengths N_B.

Page 43, 11.6.3.2.1, replace:

The following algorithm describes the calculation of the residual *d* for an input signal *x*, a predictor order *K* and LPC coefficients *cof*:

with:

The following algorithm describes the calculation of the residual *d* for an input signal *x*, a block length *N*, a predictor order *K* and LPC coefficients *cof*:

Page 44, 11.6.3.2.1, second pseudo-code listing and Page 45, 11.6.3.2.2, second pseudo-code listing, replace in both cases the fourth line:

```
for (n = 0; n < K; n++)
```

with:

```
for (n = 0; n < min(K, N); n++)
```

Page 48, replace 11.6.5 entirely, with the following text:

11.6.5 RLS-LMS Predictor

This subclause describes the backward-adaptive prediction scheme that uses adaptive RLS-LMS predictor. Block diagram of the corresponding encoder and decoder are shown in Figure 11.8 and Figure 11.9, respectively.

© ISO/IEC 2008 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14496-3_2005_AMD2_2006_COR2_2008