**3GPP TSG- Meeting # *2000***

**, , - Revision of S4-251950**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *CR-Form-v12.4* | | | | | | | | |
| **CHANGE REQUEST** | | | | | | | | |
|  | | | | | | | | |
|  |  | **CR** |  | **rev** | 2 | **Current version:** |  |  |
|  | | | | | | | | |
| *For* [***HE******LP***](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form: comprehensive instructions can be found at* [*https://www.3gpp.org/Change-Requests*](https://www.3gpp.org/Change-Requests)*.* | | | | | | | | |
|  | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Proposed change affects:*** | UICC apps |  | ME | **x** | Radio Access Network |  | Core Network | **x** |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | | |
| ***Title:*** |  | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Source to WG:*** |  | | | | | | | | | |
| ***Source to TSG:*** | S4 | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Work item code:*** |  | | | | |  | ***Date:*** | | | 19 |
|  |  | | | |  | |  | | |  |
| ***Category:*** |  |  | | | | | ***Release:*** | | |  |
|  | *Use one of the following categories:* ***F*** *(correction)* ***A*** *(mirror corresponding to a change in an earlier release)* ***B*** *(addition of feature),* ***C*** *(functional modification of feature)* ***D*** *(editorial modification)*  Detailed explanations of the above categories can be found in 3GPP [TR 21.900](http://www.3gpp.org/ftp/Specs/html-info/21900.htm). | | | | | | | | *Use one of the following releases: Rel-8 (Release 8) Rel-9 (Release 9) Rel-10 (Release 10) Rel-11 (Release 11) … Rel-17 (Release 17) Rel-18 (Release 18) Rel-19 (Release 19)  Rel-20 (Release 20)* | |
|  |  | | | | | | | | | |
| ***Reason for change:*** | | Various corrections to the IVAS Codec software (floating-point), which improve software stability, audio quality and interoperability. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Summary of change:*** | | Numerous bugfixes have been applied to the IVAS Codec software and framework. The corrections include:   * Corrections for various crashes, sanitizer errors * Corrections to improve interoperability FL <-> FX * Corrections to reduce memory demand and complexity * Corrections to various quality outliers (clicks, noise-burst, differences in loudness, etc.) * Provide missing implementation of RTP payload writing and parsing (conformant with TS 26.253) * Corrections to HRTF File-Format: Use fixed-point representation for interoperability with FX * Various corrections to codec interface to allow for a decoder-side object editing * Various corrections to the codec framework to allow for format switching   The full list of changes is available at <https://forge.3gpp.org/rep/ivas-codec-pc/ivas-codec/-/wikis/Documentation/Releases/IVAS-3.0-Release#floating-point-code>.  Further on, corrections have to be made to the specification text in TS 26.258, to mirror the changes to the software. In addition, the specification text has to be corrected wrt 6-DoF support for head-rotation and missing parameters in the renderer text file. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Consequences if not approved:*** | | Erroneous IVAS codec software, which exhibits severe quality and interoperability issues. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Clauses affected:*** | | Clauses 2, 4, 4.1, 4.2, 5.10, 5.11, 5.14.2, 5.18 (new), 5.19 (new), Electronic Attachment | | | | | | | | |
|  | |  | | | | | | | | |
|  | | **Y** | **N** |  | | | |  | | |
| ***Other specs*** | | **X** |  | Other core specifications | | | | TS 26.251 | | |
| ***affected:*** | | **X** |  | Test specifications | | | | TS 26.252 CR 0004 | | |
| ***(show related CRs)*** | |  | **X** | O&M Specifications | | | | TS/TR ... CR ... | | |
|  | |  | | | | | | | | |
| ***Other comments:*** | |  | | | | | | | | |
|  | |  | | | | | | | | |
| ***This CR's revision history:*** | | Rev 1: Inclusion of electronic attachment Rev 2: Inclusion of corrections to 5.10, update electronic attachment | | | | | | | | |

==============First change==============

# 2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non‑specific.

- For a specific reference, subsequent revisions do not apply.

- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[2] 3GPP TS 26.253: "Codec for Immersive Voice and Audio Services (IVAS); Detailed Algorithmic Description including RTP payload format and SDP parameter definitions".

[3] 3GPP TS 26.254: "Codec for Immersive Voice and Audio Services (IVAS); Rendering ".

[4] 3GPP TS 26.255: "Codec for Immersive Voice and Audio Services (IVAS); Error concealment of lost packets ".

[5] 3GPP TS 26.256: "Codec for Immersive Voice and Audio Services (IVAS); Jitter Buffer Management ".

[6] 3GPP TS 26.252: "Codec for Immersive Voice and Audio Services (IVAS); Test sequences ".

[7] IETF RFC 3550: "RTP: A Transport Protocol for Real-Time Applications".

[8] Recommendation ITU-T G.191 (03/23): "Software tools for speech and audio coding standardization".

[9] Recommendation ITU-T G.192: "A common digital parallel interface for speech standardization activities".

[10] ISO/IEC 23008-3:2015: “High efficiency coding and media delivery in heterogeneous environments — Part 3: 3D audio”

[11] ISO/IEC 23091-3:2018: “Coding-independent code points — Part 3: Audio“

[12] 3GPP TS 26.442: "Codec for Enhanced Voice Services (EVS); ANSI C code (fixed-point)".

[13] 3GPP TS 26.443: "Codec for Enhanced Voice Services (EVS); ANSI C code (floating-point)".

[14] 3GPP TS 26.452: "Codec for Enhanced Voice Services (EVS); ANSI C code; Alternative fixed-point using updated basic operators".

[15] 3GPP TS 26.114: "IP Multimedia Subsystem (IMS); Multimedia Telephony; Media handling and interaction".

[16] 3GPP TR 26.902: "Video codec performance".

==============Next change==============

# 4 C code structure

This clause gives an overview of the structure of the floating-point C code and provides an overview of the contents and organization of the C code attached to the present document.

C was selected as the programming language because portability was desirable.

==============Next change==============

## 4.1 Contents of the C source code

The C code is organized as listed in Table 1:

Table 1: Source code directory structure

|  |  |
| --- | --- |
| **Directory** | **Description** |
| readme.txt | information on how to compile and use |
| Makefile | UNIX style encoder Makefile |
| Workspace\_msvc/ | Directory for the MSVC 2017 (or newer) project files |
| apps/ | Source code files used solely for the encoder/decoder/renderer applications; these applications make use of the libraries built from lib\_com, lib\_dec, lib\_enc, lib\_rend, and lib\_util |
| lib\_com/ | Source code files used both in encoder and decoder |
| lib\_dec/ | Source code files used solely in the decoder |
| lib\_enc/ | Source code files used solely in the encoder |
| lib\_isar/ | Source code files used solely for split rendering |
| lib\_lc3plus/ | Source code files used solely for split rendering |
| lib\_rend/ | Source code files used solely in the renderer |
| lib\_util/ | Source code files solely for utility functions used by the applications |
| scripts/ | Auxiliary scripts for the conversion of HRTFs to the HRTF filter files (clause 5.10) and generation of binary renderer config metadata format" (clause 5.14.1) |

The distributed files with suffix "c" contain the source code and the files with suffix "h" are the header files. The table ROM data is contained in files named "rom\_\*" and “ivas\_rom\_\*” with suffix "c".

Makefiles are provided for the platforms in which the C code has been verified (listed above). Once the software is installed, this directory will have a compiled version of the encoder (named IVAS\_cod), the decoder (named IVAS\_dec), the renderer (named IVAS\_rend) and the split rendering post-renderer (named ISAR\_post\_rend). In addition, this directory will have a compiled version of the encoder with support for format switching (named IVAS\_cod\_fmtsw) and an example program for Ambisonics format conversion (named ambi\_converter).

==============Next change==============

## 4.2 Program execution

The codec for Immersive Voice and Audio Services is implemented in four programs and two utility executables:

*-* IVAS\_cod: encoder;

*-* IVAS\_dec: decoder;

*-* IVAS\_rend: renderer;

- ISAR\_post\_rend: split rendering post-renderer;

- IVAS\_cod\_fmtsw: encoder with support for format switching;

- ambi\_converter: example program for Ambisonics format conversion.

The programs should be called like:

- IVAS\_cod [encoder options] <input file> <bitstream file>;

- IVAS\_dec [decoder options] <bitstream file> <output file>;

- IVAS\_rend [renderer options] -i <input file> -if <input format> -o <output file> -of <output format>;

- ISAR\_post\_rend [post-renderer options] -i < bitstream file or input file> -if <input format> -o <output file>;

- IVAS\_cod\_fmtsw <format\_switching\_file>;

- ambi\_converter <input file> <output file> <input convention> <output convention>.

The input and output files contain 16-bit linear encoded PCM samples (headerless or in WAVE format) and the bitstream file contains encoded data.

The encoder, decoder, and renderer options will be explained by running the programs without any input arguments. See the file readme.txt for more information on how to run the *IVAS\_cod*, *IVAS\_dec*, *IVAS\_rend*, *ISAR\_post\_rend, IVAS\_cod\_fmtsw, ambi\_converter* programs.

==============Next change==============

## 5.1 Audio Input/output file format

For the input files read by the encoder/renderer and output files written by the decoder/renderer the following formats are supported:

- Headerless format: 16-bit integer words per each data sample. The byte order in each word depends on the host architecture (e.g. LSB first on PCs, etc.).

- WAVE format: 16-bit little-endian integer words per each data sample.

Both the encoder and the decoder program process complete frames corresponding to multiples of 20 ms.

The encoder will pad the last frame to integer multiples of 20ms frames, i.e. n speech frames will be produced from an input file with a length between [(n-1)\*20ms+1 sample; n\*20ms]. The files produced by the decoder will always have a length of n\*20ms.

Input/output audio shall follow configurations as specified in Table 2. Ambisonics components follow the ACN ordering where for real-valued spherical harmonics components of order and degree , where and .

Table 2: Audio track configurations

| **Audio format (designator)** | **Number of tracks** | **Index** | **Configuration  (incl. ordering)** | **Azimuth Range** | **Elevation Range** |
| --- | --- | --- | --- | --- | --- |
| Mono (M) | 1 | 1 | M | - | - |
| Stereo (ST) | 2 | 1,2 | L, R | - | - |
| Binaural (BIN) | 2 | 1,2 | L, R | - | - |
| Multi-channel 5.1 (MC51) | 6 | 1 | CH\_A+030\_E+00 | +30 | 0 |
| 2 | CH\_A-030\_E+00 | -30 | 0 |
| 3 | CH\_A+000\_E+00 | 0 | 0 |
| 4 | LFE | - | - |
| 5 | CH\_A+110\_E+00 | +100 … +120 | 0 … +15 |
| 6 | CH\_A-110\_E+00 | -100 … -120 | 0 ... +15 |
| Multi-channel 7.1 (MC71) | 8 | 1 | CH\_A+030\_E+00 | +30 ... +45 | 0 |
| 2 | CH\_A-030\_E+00 | -30 … -45 | 0 |
| 3 | CH\_A+000\_E+00 | 0 | 0 |
| 4 | LFE | - | - |
| 5 | CH\_A+110\_E+00 | +85 … +110 | 0 |
| 6 | CH\_A-110\_E+00 | -85 … -110 | 0 |
| 7 | CH\_A+135\_E+00 | +120 … +150 | 0 |
| 8 | CH\_A-135\_E+00 | -120 … -150 | 0 |
| Multi-channel 5.1+4 (MC514) | 10 | 1 | CH\_A+030\_E+00 | +30 | 0 |
| 2 | CH\_A-030\_E+00 | -30 | 0 |
| 3 | CH\_A+000\_E+00 | 0 | 0 |
| 4 | LFE | - | - |
| 5 | CH\_A+110\_E+00 | +100 … +120 | 0 … +15 |
| 6 | CH\_A-110\_E+00 | -100 … -120 | 0 … +15 |
| 7 | CH\_A+030\_E+35 | +30 … +45 | +30 … +55 |
| 8 | CH\_A-030\_E+35 | -30 … -45 | +30 … +55 |
| 9 | CH\_A+110\_E+35 | +100 … +135 | +30 … +55 |
| 10 | CH\_A-110\_E+35 | -100 … -135 | +30 … +55 |
| Multi-channel 7.1+4 (MC714) | 12 | 1 | CH\_A+030\_E+00 | +30 … +45 | 0 |
| 2 | CH\_A-030\_E+00 | -30 … -45 | 0 |
| 3 | CH\_A+000\_E+00 | 0 | 0 |
| 4 | LFE | - | - |
| 5 | CH\_A+135\_E+00 | +120 … +150 | 0 |
| 6 | CH\_A-135\_E+00 | -120 … -150 | 0 |
| 7 | CH\_A+090\_E+00 | +85 … +110 | 0 |
| 8 | CH\_A-090\_E+00 | -85 … -110 | 0 |
| 9 | CH\_A+030\_E+35 | +30 … +45 | +30 … +55 |
| 10 | CH\_A-030\_E+35 | -30 … -45 | +30 … +55 |
| 11 | CH\_A+135\_E+35 | +100 … +150 | +30 … +55 |
| 12 | CH\_A-135\_E+35 | -100 … -150 | +30 … +55 |
| FOA (SBA1) | 4 | 1…4 | Ambisonics components with 0,1,2,3 | - | - |
| HOA\*  (SBA) |  | 1… | Ambisonics components with 0,1, 2,… -1 | - | - |
| Mono objects (OBA) | 1…4 | 1…4 | Object(s) with ID 1…4 | - | - |
| Metadata-assisted spatial audio, mono (MASA1) | 1 | 1 | M | - | - |
| Metadata-assisted spatial audio, stereo (MASA2) | 2 | 1,2 | L, R | - | - |
| Combined mono MASA and OBA | 2...5 | 1..4  2...5 | Object(s) with ID 1…4  M MASA | -  - | -  - |
| Combined stereo MASA and OBA | 3...6 | 1..4  5,6 | Object(s) with ID 1…4  L, R MASA | -  - | -  - |
| Combined  HOA\*  (SBA)  and OBA |  | 1… | Object(s) with ID 1…4  Ambisonics components with 0,1, 2,… -1 | - | - |

\* = Ambisonics order

For Ambisonics, SN3D normalization is assumed.

The azimuth ranges are expressed in degrees; positive values rotate to the left when facing the front, i.e. counter clockwise when looking from above. The elevation ranges are expressed in degrees where positive values indicate angles above the horizontal plane.

==============Next change==============

## 5.10 HRTF filter file (decoder/renderer input)

Head related filters for the binaural rendering may be provided to the decoder or the renderer by using dynamic loading of external binary file. Examples code to generate such a binary file from a set of SOFA file is provided in the folder “binauralRenderer\_interface” in the “script” folder of the C source code. Please refer to the readme file of this folder and sub-folder.

The main script to is the matlab script called “generate\_ivas\_binauralizer\_tables\_from\_sofa.m”. It required matlab (version >= R2017b and Signal Processing Toolbox). It also requires to first generate two executables “generate\_crend\_ivas\_tables” and “tables\_format\_converter”. The process to generate these two executables is decribed in the readme file. It requires c compiler and CMake to be installed.

Running the matlab script whitout modifications will generate the binaural rom tables for the different renderers for floating code (ivas\_rom\_binaural\_crend\_head.c|h, ivas\_rom\_binauralRenderer.c|h, ivas\_rom\_TDbinauralRenderer.c|h) and fixed-point code (ivas\_rom\_binaural\_crend\_head\_fx.c|h, ivas\_rom\_binauralRenderer\_fx.c|h, ivas\_rom\_TDbinauralRenderer\_fx.c|h). It will also generate 3 binaural binary files (ivas\_binaural\_48kHz.bin, ivas\_binaural\_32kHz.bin, ivas\_binaural\_16kHz.bin). These 3 binary files contain default values corresponding to the values in the rom tables. By changing the sofa file used by the matlab script you can generate custom binaural binary files. The scripts are provided as example as they may not work will all sofa files.

The decoder program should be called with option -hrtf <binary\_file>. This option can be used with the output configurations BINAURAL, BINAURAL\_ROOM\_IR and BINAURAL\_ROOM\_REVERB.

The binaural binary file has a specific container format with a header and a sequence of entries.

The header of a binaural binary file is defined according to Table 3A as follows:

**Table 3A: Binary file header**

|  |  |  |  |
| --- | --- | --- | --- |
| **Offset** | **Format** | **Length**  **(in bytes)** | **Description** |
| 0 | string | 8 | File identifier: “IVASHRTF” |
| 8 | integer | 4 | Size of file in bytes (header of file included) |
| 12 | integer | 2 | Number of entries (HR filters) |
| 14 | integer | 4 | Max size of raw data (HR filter in binary format) |

Every entry contains a header followed by the related raw data which is the binary representation of the HR filter. The binary format for the different renderers are described in tables 3B through 3F.

The header of each entry is defined as given in Table 3B:

**Table 3B: Entry headers**

|  |  |  |  |
| --- | --- | --- | --- |
| Offset | Format | Length  (in bytes) | Description |
| 0 | integer | 4 | Renderer type  The renderer type is defined according to the enumeration RENDERER\_TYPE among the following values :  - HRTF\_READER\_RENDERER\_BINAURAL\_FASTCONV  - HRTF\_READER\_RENDERER\_BINAURAL\_FASTCONV\_ROOM  - HRTF\_READER\_RENDERER\_BINAURAL\_PARAMETRIC  - HRTF\_READER\_RENDERER\_BINAURAL\_OBJECTS\_TD  - HRTF\_READER\_RENDERER\_BINAURAL\_MIXER\_CONV  - HRTF\_READER\_RENDERER\_BINAURAL\_MIXER\_CONV\_ROOM  - HRTF\_READER\_RENDERER\_REVERB\_ALL |
| 4 | integer | 4 | Input audio configuration  The input audio configuration is defined according to the enumeration BINAURAL\_INPUT\_AUDIO\_CONFIG among the following values :  - BINAURAL\_INPUT\_AUDIO\_CONFIG\_COMBINED  - BINAURAL\_INPUT\_AUDIO\_CONFIG\_HOA3  - BINAURAL\_INPUT\_AUDIO\_CONFIG\_HOA2  - BINAURAL\_INPUT\_AUDIO\_CONFIG\_FOA  - BINAURAL\_INPUT\_AUDIO\_CONFIG\_UNDEFINED |
| 8 | integer | 4 | Sampling frequency (16000, 32000, 48000) |
| 12 | integer | 4 | Raw data size in bytes |

The format of the raw data depends on the rendering and the HR filters are represented in fix point.

Note:

- The HR filters for the renderer types HRTF\_RENDERER\_BINAURAL\_PARAMETRIC, HRTF\_RENDERER\_BINAURAL\_FASTCONV and HRTF\_RENDERER\_BINAURAL\_FASTCONV\_ROOM are fully defined at 48kHz.

- For the renderer type HRTF\_RENDERER\_BINAURAL\_OBJECTS\_TD the input audio configuration is always BINAURAL\_INPUT\_AUDIO\_CONFIG\_UNDEFINED.

- renderer type HRTF\_READER\_RENDERER\_REVERB\_ALL should be associated with HRTF\_READER\_RENDERER\_BINAURAL\_OBJECTS\_TD and/or HRTF\_READER\_RENDERER\_BINAURAL\_CREND to specify the binaural reverberation parameters jointly with new HRIR parameters. They shall be computed on the same HRIR set.

- The binary file does not have to contain all data (HR filter) for all renderers. The following minimal configurations are accepted or any combination of those:

|  |  |  |
| --- | --- | --- |
| HRTF\_READER\_RENDERER\_BINAURAL\_FASTCONV | BINAURAL\_INPUT\_AUDIO\_CONFIG\_COMBINED | Contains data for Combined HRIR |
| HRTF\_READER\_RENDERER\_BINAURAL\_FASTCONV | BINAURAL\_INPUT\_AUDIO\_CONFIG\_HOA3 | Contains data for HOA3 |
| HRTF\_READER\_RENDERER\_BINAURAL\_FASTCONV | BINAURAL\_INPUT\_AUDIO\_CONFIG\_HOA2 | Contains data for HOA2 |
| HRTF\_READER\_RENDERER\_BINAURAL\_FASTCONV | BINAURAL\_INPUT\_AUDIO\_CONFIG\_FOA | Contains data for FOA |
| HRTF\_READER\_RENDERER\_BINAURAL\_FASTCONV\_ROOM | BINAURAL\_INPUT\_AUDIO\_CONFIG\_COMBINED | Contains data for combined BRIR |
| HRTF\_READER\_RENDERER\_BINAURAL\_PARAMETRIC | BINAURAL\_INPUT\_AUDIO\_CONFIG\_HOA3 | Contains data for HOA3, HOA2, FOA and reverberation from BRIR |
| HRTF\_READER\_RENDERER\_BINAURAL\_OBJECTS\_TD | BINAURAL\_INPUT\_AUDIO\_CONFIG\_UNDEFINED | Contains data for HRIR |
| HRTF\_READER\_RENDERER\_BINAURAL\_CREND | BINAURAL\_INPUT\_AUDIO\_CONFIG\_COMBINED | Contains data for combined HRIR |
| HRTF\_READER\_RENDERER\_BINAURAL\_CREND | BINAURAL\_INPUT\_AUDIO\_CONFIG\_HOA3 | Contains data for HOA3 |
| HRTF\_READER\_RENDERER\_BINAURAL\_CREND | BINAURAL\_INPUT\_AUDIO\_CONFIG\_HOA2 | Contains data for HOA2 |
| HRTF\_READER\_RENDERER\_BINAURAL\_CREND | BINAURAL\_INPUT\_AUDIO\_CONFIG\_FOA | Contains data for FOA |
| HRTF\_READER\_RENDERER\_BINAURAL\_CREND\_ROOM | BINAURAL\_INPUT\_AUDIO\_CONFIG\_COMBINED | Contains data for combined BRIR only (BINAURAL\_ROOM\_IR) |
| HRTF\_READER\_RENDERER\_REVERB\_ALL |  | Contains data for HRIR with reverberation (BINAURAL\_ROOM\_REVERB)when TD renderer or mixerconv are used |

Table 3C: HR filters for binaural renderer Fastconv Impulse response binary entries

|  |  |  |  |
| --- | --- | --- | --- |
| Offset | Format | Length  (in bytes) | Description |
| 0 | integer | 2 | Scaling factor for latency value |
| 2 | integer | 4 | Latency value\* |
| 6 | integer | 2 | Number of Binaural convolution bands (Nb) |
| 8 | integer | 2 | Number of channels (Nc) |
| 10 | integer | 2 | Number of taps per filter (Nt) |
| 12 | integer | 2 | Scaling factor for filters taps |
| 14 | integers | 2 \* Nb \* Nc \* Nt | Left ear real taps values\* |
| 14 + 2 \* Nb \* Nc \* Nt | integers | 2 \* Nb \* Nc \* Nt | Left ear imaginary taps values\* |
| 14 + 2 \* 2 \* Nb \* Nc \* Nt | integers | 2 \* Nb \* Nc \* Nt | Right ear real taps values\* |
| 14 + 3 \* 2 \* Nb \* Nc \* Nt | integers | 2 \* Nb \* Nc \* Nt | Right ear imaginary taps values\* |

Table 3D: HR filters for binaural renderer Fastconv Room Impulse Response binary entries

|  |  |  |  |
| --- | --- | --- | --- |
| Offset | Format | Length  (in bytes) | Description |
| 0 | integer | 2 | Scaling factor for latency value |
| 2 | integer | 4 | Latency value\* |
| 6 | integer | 2 | Number of Binaural convolution bands (Nb) |
| 8 | integer | 2 | Number of channels (Nc) |
| 10 | integer | 2 | Number of taps per filter (Nt) |
| 12 | integer | 2 | Scaling factor for filters taps |
| 14 | integers | 2 \* Nb \* Nc \* Nt | Left ear real taps values\* |
| 14 + 2 \* Nb \* Nc \* Nt | integers | 2 \* Nb \* Nc \* Nt | Left ear imaginary taps values\* |
| 14 + 2 \* 2 \* Nb \* Nc \* Nt | integers | 2 \* Nb \* Nc \* Nt | Right ear real taps values\* |
| 14 + 3 \* 2 \* Nb \* Nc \* Nt | integers | 2 \* Nb \* Nc \* Nt | Right ear imaginary taps values\* |
| 14 + 4 \* 2 \* Nb \* Nc \* Nt | integer | 2 | CLDFB max number of channels (Nm) |
| 16 + 4 \* 2 \* Nb \* Nc \* Nt | integer | 2 | Scaling factor for reverberation time values |
| 18 + 4 \* 4 \* Nb \* Nc \* Nt | integers | 2 \* Nm | reverberation time values\* |
| 18 + 4 \* 4 \* Nb \* Nc \* Nt + 2 \* Nm | integer | 2 | Scaling factor for energies corrections values |
| 20 + 4 \* 4 \* Nb \* Nc \* Nt + 2 \* Nm | integers | 2 \* Nm | Energies corrections values \* |

Table 3E: HR filters for binaural renderer parametric

|  |  |  |  |
| --- | --- | --- | --- |
| Offset | Format | Length  (in bytes) | Description |
| 0 | integer | 2 | Number of channels (Nc) |
| 2 | integer | 2 | Number of bins (Nb) |
| 4 | integer | 2 | Scaling factor for filters taps |
| 6 | integers | 2 \* 2 \* Nc \* Nb | Real taps values\* one for each ear |
| 6 + 2 \* 2 \* Nc \* Nb | integers | 2 \* 2 \* Nc \* Nb | Imaginary taps values\* one for each ear |
| 6 + 2 \* 2 \* 2 \* Nc \* Nb | integer | 2 | Scaling factor for reverberation time values |
| 8 + 2 \* 2 \* 2 \* Nc \* Nb | integers | 2 \* Nm | reverberation time values\* |
| 8 + 2 \* 2 \* 2 \* Nc \* Nb + 2 \* Nm | integer | 2 | Scaling factor for energies corrections values |
| 10 + 2 \* 2 \* 2 \* Nc \* Nb + 2 \* Nm | integers | 2 \* Nm | Energies corrections values \* |
| 10 + 2 \* 2 \* 2 \* Nc \* Nb + 4 \* Nm | integer | 2 | Scaling factor for early part energies corrections values |
| 12 + 2 \* 2 \* 2 \* Nc \* Nb + 4 \* Nm | integers | 2 \* Nm | Early part energies corrections values \* |

Table 3F: HR filters for binaural renderer Crend entries

|  |  |  |  |
| --- | --- | --- | --- |
| Offset | Format | Length  (in bytes) | Description |
| 0 | integer | 2 | Scaling factor for latency value |
| 2 | integer | 4 | Latency value\* |
| 6 | integer | 2 | Number of HRIR/BRIR (Nc) |
| 8 | integer | 2 | Number of Binaural channels (Nb = 2) |
| 10 | integer | 2 | Max number of block iterations (Ni) |
| 12 | integers | 2 \* Nc \* Nb | Number of iteration per channel |
| 12 + 2 \* Nc \* Nb | integer | 2 \* Nc \* Nb \* Ni | Max frequency value for each block of direct part (Nf[c][b][i] Tri dimensional tab of size [Nc][Nb][Ni]) |
| 12 + 2 \* Nc \* Nb + 2 \* Nc \* Nb \* Ni | integers | 2 | Max number of iterations for diffuse part (Nid) |
| 14 + 2 \* Nc \* Nb + 2 \* Nc \* Nb \* Ni | integers | 2 \* Nb | Number of diffuse iterations per binaural channel |
| 14 + 2 \* Nc \* Nb + 2 \* Nc \* Nb \* Ni + 2 \* Nb | integers | 2 \* Nb \* Nid | Max frequency value for each block of diffuse part (Nfdiff[b][i] Twoi dimensional tab of size [Nb][Ni]) |
| 14 + 2 \* Nc \* Nb + 2 \* Nc \* Nb \* Ni + 2 \* Nb + 2 \* Nb \* Nid | integer | 2 | Max frequency value over all diffuse blocks |
| 16 + 2 \* Nc \* Nb + 2 \* Nc \* Nb \* Ni + 2 \* Nb + 2 \* Nb \* Nid | integer | 2 | Scaling factor for inverse diffuse weight values |
| 18 + 2 \* Nc \* Nb + 2 \* Nc \* Nb \* Ni + 2 \* Nb + 2 \* Nb \* Nid | integers | 2 \* Nc | Left ear inverse diffuse weight values\* |
| 18 + 2 \* Nc \* Nb + 2 \* Nc \* Nb \* Ni + 2 \* Nb + 2 \* Nb \* Nid + 2 \* Nc | integers | 2 \* Nc | Right ear inverse diffuse weight values\* |
| 18 + 2 \* Nc \* Nb + 2 \* Nc \* Nb \* Ni + 2 \* Nb + 2 \* Nb \* Nid + 2 \* 2 \* Nc | integer | 4 | Max number of bins over all HRIR/BRIR for direct part (Nbin = ) |
| 22 + 2 \* Nc \* Nb + 2 \* Nc \* Nb \* Ni + 2 \* Nb + 2 \* Nb \* Nid + 2 \* 2 \* Nc | integer | 2 | Scaling factor for filters taps |
| 24 + 2 \* Nc \* Nb + 2 \* Nc \* Nb \* Ni + 2 \* Nb + 2 \* Nb \* Nid + 2 \* 2 \* Nc | integers | 2 \* Nbin | Direct part real taps values\* |
| 24 + 2 \* Nc \* Nb + 2 \* Nc \* Nb \* Ni + 2 \* Nb + 2 \* Nb \* Nid + 2 \* 2 \* Nc + 2 \* Nbin | integers | 2 \* Nbin | Direct part imaginary taps values\* |
| 24 + 2 \* Nc \* Nb + 2 \* Nc \* Nb \* Ni + 2 \* Nb + 2 \* Nb \* Nid + 2 \* 2 \* Nc + 2 \* 2 \* Nbin | integer | 4 | Max number of bins over all HRIR/BRIR for diffuse part (Nbindiff = ) |
| 24 + 2 \* Nc \* Nb + 2 \* Nc \* Nb \* Ni + 2 \* Nb + 2 \* Nb \* Nid + 2 \* 2 \* Nc + 2 \* 2 \* Nbin | integers | 2 \* Nbindiff | Diffuse part Real taps values\* |
| 24 + 2 \* Nc \* Nb + 2 \* Nc \* Nb \* Ni + 2 \* Nb + 2 \* Nb \* Nid + 2 \* 2 \* Nc + 2 \* Nbdir + 2 \* Nbindiff | integers | 2 \* Nbindiff | Diffuse part imaginary taps values\* |







==============Next change==============

## 5.11 Head rotation trajectory file (decoder/renderer input)

In the reference implementation of the codec, input data representing the current rotation of the listeners head can be provided to the decoder in an ASCII formatted file comprising four columns separated by commas. These columns contain floating-point numbers representing either a quaternion or a Euler angle. The distinction between these two input formats is made by a magic number in the first column. If this value is set to -3.0, it is assumed that the remaining three columns contain three Euler angles. Otherwise, all four columns are interpreted as a Quaternion. The input is expected to have one line for each subframe of 5 ms.

In the case of Quaternion-based input, the columns are the w, x, y, z components of a unit quaternion. Proper normalization to 1 shall be maintained in the input. The coordinate system is defined such that the x-axis points into the direction of view, the y axis points towards the left ear, and the z axis point from bottom to top. The origin is in the center of the head. See also TS 26.253, clause 7.4.3 [2]. For example, an approximate 90-degree rotation around the vertical (z) axis would be represented by the following input line:

0.707107,0.000000,0.000000,0.70710

In the case of Euler-angle input, the first column contains the magic number -3.0, and the next three columns are the Euler angles yaw, pitch, and roll. The rotations are applied in the order yaw-pitch-roll. The yaw angle rotates around the z axis. The pitch angle rotates around the new y axis. The roll angle rotates around the new x axis. The equivalent of the example line above is then:

-3.0,90.000035,0.000000,0.000000

In case of 6 DoF support in the renderer, the head rotation trajectory file may also include a listener position in absolute Cartesian coordinates on the x-, y- and z-axis. Note that the listener position is expressed in absolute coordinates, while the listener orientation is expressed as scene displacement. An example of a listener positioned at x=3.0, y=4.0 and z=0 would be:

-3.0,90.000035,0.000000,0.000000,3.0,4.0,0.0

Note that the listener position applies for listener orientation expressed both in Quaternions and Euler angles.

==============Next change==============

### 5.14.2 Text renderer config metadata format

The text based renderer configuration file contains the following syntax elements:

[general] header of general metadata

binaryConfig = path; path to the binary configuration file

[roomAcoustics] header of room acoustic metadata group

frequencyGridCount = N; number of frequency grids

acousticEnvironmentCount = N; number of acoustic environments

[frequencyGrid:N] header of a frequency grid, where N is a zero-based, sequential grid index

method = individualFrequencies | startHopAmount | defaultBanding;  
specifies frequency grid representation method

nrBands = N; number of frequency bands, applicable for individual frequencies and start-hop-amount representation methods

frequencies = [...]; center frequencies for individualFrequencies representation method, a comma separated list of N numeric values (ints or floats)

startFrequency = value; starting frequency for start-hop-amount representation method

frequencyHop = value; frequency hop for start-hop-amount representation method. Center frequencies for a grid are computed as fcn = fcn-1 \* hop

defaultGrid = N; default grid identifier. The available default grids are as in Annex B.1, Table B.4.

defaultGridOffset = N; it is possible to use a subset of a default grid by specifying an offset - index of the first center frequency of the default grid and

defaultGridNrBands = N; number of bands from the default grid to be used

[acousticEnvironment:N] header of an acoustic environment element, where N is a zero-based grid index (does not have to be sequential)

frequencyGridIndex = N; index of the frequency grid (see above) used for frequency dependent parameters

preDelay = value; a delay at which DSR (diffuse to source ratios) were measured

rt60 = [...]; RT60 values per frequency band

dsr = [...]; diffuse to source sound energy ratio per frequency band

earlyReflectionsSize = [x, y, z]; shoebox model room size in x, y, z dimension in meters

absorptionCoeffs = [x1, x2, y1, y2, z1, z2];  
early reflections absorption coefficients per wall

listenerOrigin = [x, y, z]; early reflections listener origin (optional) as offset from the room center

lowComplexity = TRUE | FALSE; early reflection low-complexity mode flag (FALSE by default)

[directivitySetting] header of the directivity data group

directivityCount = N; number of directivity components

[directivityPattern:N] header of a directivity pattern element, where N is a zero-based element index

[distanceAttenuation] header of the distance attenuation data group

maxDist = md; Max distance for distance attenuation function

refDist = rd; Ref (minimum) distance for distance attenuation function

rolloffFactor = rf; Rolloff-factor for distance attenuation function

directivity = [ia, oa, og]; directivity data: ia – inner angle, oa – outer angle, og – outer gain.

[SPLITREND] header of split rendering group

BITRATE = R; split rendering bitrate

DOF = N; degree of freedom (N ranging from 0 to 3)

HQMODE = N; High quality mode for 3DOF (N can be 0 or 1), adds more complexity at pre-renderer

CODEC = X; split rendering transport codec (X can be LCLD or LC3plus or NONE)

FRAMESIZE = [5, 10, 20] frame size in ms of the split rendering transport codec. Note: LC3plus supports 5 and 10 ms framesize, LCLD supports 5, 10 and 20 ms framesize.

The config file format supports comments starting with a hash sign #. It also supports splitting data into multiple lines, useful in case of larger arrays.

==============Next change==============

## 5.18 Object editing file (decoder input)

For object based audio input (including the combined formats OBA + MASA and OBA + SBA), the decoder supports editing of object characteristics while decoding/rendering. This allows for a scene adjustment on receiver side.

The parameters for the object editing in decoder for the supported formats are provided via a text parameter file. Each row of the file corresponds to one 20ms IVAS frame. The row contains one or more of the following parameters separated by a comma, as described in Table 7.

Table 7: Object Editing File Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| bg\_gain=<float> | linear gain to be applied on the SBA/MASA component in OSBA/OMASA, no effect for ISM |
| obj\_<int>\_gain=<float> | linear gain to be applied on object <int>, 0-based indexing |
| obj\_<int>\_relgain=0|1 | if 1, obj\_<int>\_gain is interpreted as a relative modification. default is absolute modification |
| obj\_<int>\_azi=<float> | azimuth angle in degrees to be applied on object <int>, 0-based indexing |
| obj\_<int>\_relazi=0|1 | if 1, obj\_<int>\_azi is interpreted as a relative modification. default is absolute modification |
| obj\_<int>\_ele=<float> | elevation angle in degrees to be applied on object <int>, 0-based indexing |
| obj\_<int>\_relele=0|1 | if 1, obj\_<int>\_ele is interpreted as a relative modification. default is absolute modification |

In addition to these metadata parameters, editing of extended metadata parameters is supported for Discrete ISM, OMASA Discrete ISM and OSBA Discrete ISM input formats. Extended metadata parameters consist of radius, yaw and pitch, and they are described in Table 8.

Table 8: Object Editing File Extended Metadata Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| obj\_<int>\_radius=<float> | linear radius to be applied on object <int>, 0-based indexing |
| obj\_<int>\_relradius=0|1 | if 1, obj\_<int>\_radius is interpreted as a relative modification. default is absolute modification |
| obj\_<int>\_yaw=<float> | yaw angle in degrees to be applied on object <int>, 0-based indexing |
| obj\_<int>\_relyaw=0|1 | if 1, obj\_<int>\_yaw is interpreted as a relative modification. default is absolute modification |
| obj\_<int>\_pitch=<float> | pitch angle in degrees to be applied on object <int>, 0-based indexing |
| obj\_<int>\_relpitch=0|1 | if 1, obj\_<int>\_pitch is interpreted as a relative modification. default is absolute modification |

If a parameter is not specified, that parameter is not edited. An empty line in the file corresponds to not editing any parameter in the item.

## 5.19 RTPDUMP file (encoder output, decoder input)

The rtpdump file format is used as the interchange format of IVAS when RTP packing or unpacking is included as part of the encoder or decoder operation, respectively.

The rtpdump file format has already been used in the EVS decoder [12, 13, 14], MTSI [15] and TR 26.902 [16]. It has been originally proposed by Henning Schulzrinne, see <http://www.cs.columbia.edu/IRT/software/rtptools/>. Within the scope of this IVAS, only the binary version of the file format is of relevance. The file is constructed as follows:

The file starts with one line of ASCII coded text, indicating:

#!rtpplay1.0 address/port\n

wherein "address" stands for an IP address (e.g. 192.168.1.2) and port stands for a port number, e.g. 1234. Neither value is used by the toolchain employed in this report. "\n" stands for carriage return/linefeed.

The ASCII header is followed by one binary header (RD\_hdr\_t) and one RD\_packet\_t structure for each received packet. All fields are in network byte order. The RTP and RTCP packets are recorded as-is.

typedef struct {

struct timeval start; /\* start of recording (GMT) \*/

u\_int32 source; /\* network source (multicast address) \*/

u\_int16 port; /\* UDP port \*/

} RD\_hdr\_t;

typedef struct {

u\_int16 length; /\* length of packet, including this header (may

be smaller than plen if not whole packet recorded) \*/

u\_int16 plen; /\* actual header+payload length for RTP, 0 for RTCP \*/

u\_int32 offset; /\* milliseconds since the start of recording \*/

} RD\_packet\_t;

==============End of change==============