**Source: Editor1**

**Title:** **IVAS-7a: Processing plan for selection phase**

**Version: v.0.8.0**

**Agenda Item: 14.2**

Document History:

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Detail of changes** |
| 0.1.0 | 25 Oct 2019 | Initial version |
| 0.1.1 | 22 Jan 2020 | Implementation of S4-200119 according to agreements in EVS SWG |
| 0.2.0 | 5 Feb 2021 | Changed scope to selection phase |
| 0.3.0 | 23 Feb 2022 | Implementation of S4-220164, S4-220165, S4-220166 with agreed amendments of the EVS SWG at SA4#117-e. |
| 0.4.0 | 13 Apr 2022 | Implementation of S4-220432 |
| 0.5.0 | 13 Apr 2022 | Agreement of clause 3.5-3.7 |
| 0.6.0 | 16 May 2022 | Implementation of S4-220733 |
| 0.7.0 | 17 Nov 2022 | Implementation of S4-221464 |
| 0.8.0 | 23 Feb 2023 | Agreements based on S4-230221, S4-230239, S4-230261, S4-230262 implemented at SA4#122. |

**CONTENTS**

[3.1 Acronyms 4](#_Toc128002043)

[3.2 Definitions 4](#_Toc128002044)

[3.3 Experiments 4](#_Toc128002045)

[3.4 Listening lab designators 5](#_Toc128002046)

[3.5 Bitstream format 5](#_Toc128002047)

[3.6 Audio format 5](#_Toc128002048)

[3.7 Audio track designators 5](#_Toc128002049)

[3.8 Audio track configurations 6](#_Toc128002050)

[3.9 Preamble definition 7](#_Toc128002051)

[3.10 File naming 8](#_Toc128002052)

[3.10.1 Generic definitions 8](#_Toc128002053)

[3.10.2 Naming examples of the input files and processed output files 8](#_Toc128002054)

[3.10.3 File naming error patterns 8](#_Toc128002055)

[3.10.4 File naming jitter profiles 8](#_Toc128002056)

[3.10.5 File naming jitter derived error patterns 8](#_Toc128002057)

[3.10.6 File naming rate switching profiles 8](#_Toc128002058)

[4.1 General considerations for processing 9](#_Toc128002059)

[4.1.1 Source material requirements 9](#_Toc128002060)

[4.1.2 Concatenated sequences processing 9](#_Toc128002061)

[4.1.3 Frame error application 9](#_Toc128002062)

[4.2 Pre-processing 10](#_Toc128002063)

[4.2.1 Pre-processing stages 10](#_Toc128002064)

[4.2.2 High-pass filtering 10](#_Toc128002065)

[4.2.3 Level adjustment 10](#_Toc128002066)

[4.3 Processing for anchor conditions 10](#_Toc128002067)

[4.3.1 MNRU/ESDRU 10](#_Toc128002068)

[4.4 Processing for stereo inputs 11](#_Toc128002069)

[4.4.1 Reference conditions 11](#_Toc128002070)

[4.4.2 CuT conditions 11](#_Toc128002071)

[4.5 Processing for binaural audio inputs 11](#_Toc128002072)

[4.5.1 Pre-processing 11](#_Toc128002073)

[4.5.2 Reference conditions 11](#_Toc128002074)

[4.5.3 CuT conditions 11](#_Toc128002075)

[4.6 Processing for multi-channel audio inputs 12](#_Toc128002076)

[Reference conditions 12](#_Toc128002077)

[4.6.1 CuT conditions 12](#_Toc128002078)

[4.7 Processing for scene-based audio inputs 12](#_Toc128002079)

[4.7.1 Reference conditions 12](#_Toc128002080)

[4.7.2 CuT conditions 12](#_Toc128002081)

[4.8 Processing for metadata-assisted spatial audio (MASA) inputs 13](#_Toc128002082)

[4.8.1 Reference conditions 13](#_Toc128002083)

[4.8.2 CuT conditions 13](#_Toc128002084)

[4.9 Processing for object-based audio inputs 14](#_Toc128002085)

[4.9.1 Reference conditions 14](#_Toc128002086)

[4.9.2 CuT conditions 14](#_Toc128002087)

[5.1 Pre- and post-processing operations 15](#_Toc128002088)

[5.1.1 General delay compensation for the STL filter tool 15](#_Toc128002089)

[5.1.2 Filtering operations 15](#_Toc128002090)

[5.1.3 Level adjustment 15](#_Toc128002091)

[5.1.4 Scaling by Factor F 16](#_Toc128002092)

[5.1.5 Low/High level De-/Scaling 16](#_Toc128002093)

[5.1.6 Summation of speech and noise files 17](#_Toc128002094)

[5.1.7 File concatenation 17](#_Toc128002095)

[5.1.8 Audio format conversion 17](#_Toc128002096)

[5.1.9 Sampling Rate changes (resampling) 17](#_Toc128002097)

[5.1.10 Windowing and segmentation 18](#_Toc128002098)

[5.1.11 Bit conversion 19](#_Toc128002099)

[5.1.12 Interleave audio files 19](#_Toc128002100)

[5.1.13 De-interleave audio files 19](#_Toc128002101)

[5.2 Processing 19](#_Toc128002102)

[5.2.1 MNRU reference conditions 19](#_Toc128002103)

[5.2.2 ESDRU reference conditions 20](#_Toc128002104)

[5.2.3 Reference codecs for Narrowband conditions 20](#_Toc128002105)

[5.2.4 Reference codecs for Wideband conditions 20](#_Toc128002106)

[5.2.5 Reference codecs for super-wideband conditions 22](#_Toc128002107)

[5.2.6 Delay compensation for reference conditions 22](#_Toc128002108)

[5.2.7 EVS operation (non-JBM case) 22](#_Toc128002109)

[5.2.8 EVS operation (JBM case) 23](#_Toc128002110)

[5.2.9 CuT operation (non-JBM case) 23](#_Toc128002111)

[5.2.10 CuT operation (JBM case) 23](#_Toc128002112)

[5.3 Encoder and decoder CuT executable requirements 23](#_Toc128002113)

[5.4 Error Insertion (EID) 23](#_Toc128002114)

[5.4.1 Frame error tool 24](#_Toc128002115)

[5.4.2 Pattern generation 24](#_Toc128002116)

[5.4.3 Derive EPFs from jitter profiles 24](#_Toc128002117)

[5.4.4 Rate switching profile generation 25](#_Toc128002118)

[5.4.5 Network simulator for packet jitter generation 25](#_Toc128002119)

[5.4.6 Cutting Tool for JBM tests 25](#_Toc128002120)

[A.1 Delay compensation for filter operations and reference codecs 27](#_Toc128002121)

[A.2 Binaries used 28](#_Toc128002122)

[A.2.1 ITU-T STL processing tools 28](#_Toc128002123)

[A.2.2 AMR Error insertion device 28](#_Toc128002124)

[A.2.3 Reference codecs 29](#_Toc128002125)

[A.2.4 Other tools 31](#_Toc128002126)

[B.1 Databases for objective evaluation 34](#_Toc128002127)

[B.2 Active Frame Ratio (AFR) 34](#_Toc128002128)

[B.3 Gain measuring of active frames and inactive frames 35](#_Toc128002129)

[B.4 JBM performance evaluation 35](#_Toc128002130)

[B.5 Bitrate evaluation 36](#_Toc128002131)

[C.1 Delay and error profile offset 36](#_Toc128002132)

[C.2 Seeds for rate switching 36](#_Toc128002133)

[C.3 Initialization of gen-patt tool 36](#_Toc128002134)

[D.1.1 Delay Jitter 37](#_Toc128002135)

[D.1.2 Error burstiness 37](#_Toc128002136)

# Introduction

This document defines how audio material must be prepared for the selection tests of the 3GPP IVAS codec. It should be read in conjunction with its associated IVAS selection phase test plan [1].

Editor’s Note: The processing modules need to be reviewed for processing beyond mono.

Editor’s Note: The following main remaining work has been identified:

* The permanent document lacks the definition of the following items/processing steps:
  + Definition of Preamble -> minor
  + Definition of rendering (binaural/loudspeaker) algorithms for IVAS and reference codecs for all applicable formats -> major missing item, Fraunhofer aim to bring input
  + Pre-processing:
    - Filtering operation: Definition of filters (HP50\_SHQ filters were used in EVS but are not part of STL2022) -> filters are available, just need to decide
  + Post-processing:
    - Clarification whether additional post-processing steps are needed, e.g. additional level/loudness adjustment -> post normalization still discussed, VoiceAge aim to bring input
  + Processing for test item generation for the immersive conversation use-case -> major missing item, see comments below
    - Stereo/binaural -> only SWB impulse responses available (ITU-T)
    - SBA impulse responses -> examples from Dolby, HEAD Acoustics
    - Objects -> pretty straight forward
    - MASA -> using ambisonics impulse responses as a backup plan (both FOA and HOA2 would be beneficial)
  + Collection of data base for objective Performance Requirements -> major missing item
    - Would parties be able to contribute? Under what conditions?
      * Legal framework might be needed
  + Collection of background material -> major missing item
  + Generation of Speech+Background material
    - Decide on e.g., SNR
  + Random selection of test material for Generic Audio, mixed/music? -> unclear whether material from labs could be used, or random selection from a pool of collected material would be needed
  + Processing for objective Performance Requirements -> missing, minor issue
  + Assess whether the specified version of G.191 STL (07/22) is appropriate or whether an updated version is required
* A set of corresponding processing scripts which implements the agreed processing steps -> major missing item, a proposed way forward was agreed in the Audio SWG but there are some hurdles.

# Responsibilities

The host lab (HL) [TBD] is responsible for the processing of the experiments while the cross-check lab (CL) [TBD] verifies the correctness of the processing.

The listening tests will be conducted by the listening labs (LL): TBD.

Editor’s Note: We need to define who will develop processing scripts. For EVS, one set of processing scripts was developed by the host lab, and another set by proponent companies to be used by the cross-check lab.

# Definitions and formats

## Acronyms

CL Cross-check Laboratory

CuT Codec under Test

DCR Degradation Category Rating

DTX Discontinuous transmission

ESDRU Energy-based Spatial Distortion Reference Unit

EVS Enhanced Voice Services

FB Fullband

FE Frame Erasure

FOA First-Order Ambisonics

GAL Global Analysis Laboratory

HL Host Laboratory

HOA3 Higher-Order Ambisonics, 3rd order

IVAS Immersive Voice and Audio Services

LKFS Loudness, K-weighted, relative to Full Scale

LL Listening Laboratory

MASA Metadata-Assisted Spatial Audio

MNRU Modulated Noise Reference Unit

MUSHRA Multi Stimulus test with Hidden Reference and Anchor

NB Narrowband

SWB Super-wideband

WB Wideband

## Definitions

Anchor condition Controlled impairment, e.g., using MNRU, ESDRU, low-pass filtered signal

CuT condition Codec under test processed for testing

Reference condition Quality target for comparative tests, sometimes also called Direct condition

Reference codec condition Quality comparison to reference codec, e.g. EVS, for evaluation of performance requirements

## Experiments

The selection test plan‎ (IVAS-8a) [1] describes the listening experiments. For the processing of each experiment, a two-letter experiment designator shall be used for the file naming convention. An overview of all experiments is presented in Table 1.

Editor’s Note: Table 1 structure is TBD based on the test plan.

Table 1: Overview of experiments

| Experiment Designator | Audio format | Bandwidth | Tested Conditions |
| --- | --- | --- | --- |
|  |  |  |  |

## Listening lab designators

The following table lists the responsible listening labs (LL) with their dedicated LL designator to be used for the file naming convention.

Table 2: List of listening lab designators

|  |  |
| --- | --- |
| **LL designator** | **Listening Lab Company** |
| **a** | TBD |

## Bitstream format

The candidate codec shall be based on ITU-T G.192 [2] format as a common bit-stream interface.

## Audio format

The format of input/output audio files is WAVE, 16-bit little endian format. For multi-track audio, the audio tracks are ordered according to Table 5.

## Audio track designators

Audio tracks are designated according to Table 3 with further specification according to Table 4.

Table 3: Track designators

| Track designator | Definition | |
| --- | --- | --- |
|  | Index: 1, 2, … |
| M | Mono channel | |
| L | Left channel | |
| R | Right channel | |
| CH\_A\_E | Channel at nominal azimuth angle  and nominal elevation angle | |
| LFE | LFE channel | |

Table 4: Channel specification

| Specification variable | Definition |
| --- | --- |
|  | Nominal azimuth angle in  range: degrees  Represented by three-digit number always including a sign (padded from the left with zeros if necessary).  Positive values indicate positions left of the frontal direction. The sign assigned to azimuth value zero shall be +. |
|  | Nominal elevation angle in  range: degrees  Represented by two-digit number always including a sign (padded from the left with zeros if necessary).  Positive values indicate positions above the horizontal plane. The sign assigned to elevation value zero  shall be +. |

## Audio track configurations

Input/output audio shall follow configurations as specified in Table 5. Ambisonics components follow the ACN ordering where for real-valued spherical harmonics components of order and degree , where .

Table 5: 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 | - | - |

\* = Ambisonics order

## Preamble definition

[

A preamble shall be low-level noise obtained as a digital silence of 16-bit values with amplitudes between +4 and -4.

Editor’s Note: For multi-channel, the correlation between channels of preamble should be further studied.

]

## File naming

### Generic definitions

The following generic filename convention is used for input/output audio files:

*input* Input to a processing module (or sequence of modules)

*output* Output from a processing module (or sequence of modules)

Table 6 specifies suffixes to be used whenever needed for differentiation and may otherwise be left out.

Table 6: File name suffixes

| Suffix | Specification |
| --- | --- |
|  | Track designator according to Table 3 or track index according to Table 5 |
|  | Sampling frequency in kHz being .8k, .16k, .32k or .48k |

### Naming examples of the input files and processed output files

The following sections describe the file naming conventions where all letters in blue indicate variables in the filename while black letters indicate constants.

Editor’s Note: File naming conventions TBD based on scope of input/output/processed material.

[

### File naming error patterns

Frame erasure files will have the name “*patterns****\***eefrr.g192”, where:

* ee stands for the experiment designator (see Table 1)
* f stands for frame loss pattern file
* rr stands for the frame loss rate in per cent, i.e. 03 or 06.

### File naming jitter profiles

According to [4], the MTSI jitter profiles have the file names “*patterns****\***dly\_error\_profile\_x.dat” where x is the profile number, i.e., 1,2,3,4,5,6.

### File naming jitter derived error patterns

The error patterns which are derived from the MTSI jitter profiles have the file names “*patterns****\***f\_profile\_x.g192”, where:

* f stands for frame loss pattern file
* profile\_x stands for the number of the MTSI jitter profile, see 3.9.4.

### File naming rate switching profiles

Rate switching profiles will have the name “*patterns****\***eersx”, where:

* ee stands for the experiment designator (see Table 1)
* rs stands for rate switching file
* x stands for the sequence in the experiment, i.e. 1, 2, etc.

]

# Processing stages

This section defines, in the form of diagrams, the processing stages that shall be supported by the IVAS candidate codec under test.

Editor’s note: The scope of the IVAS codec selection is not yet agreed. Agreed processing stages which are not yet agreed to be in scope for the IVAS codec selection are kept within brackets.

## General considerations for processing

Editor’s Note: Section 4.1 is TBD, currently based on EVS processing plans.

[

### Source material requirements

The input source material shall follow the format specification in section 3.5.

### Concatenated sequences processing

In all Experiments, the pre-processed material will be processed in concatenated files comprising a preamble followed by the samples being evaluated. The preamble concatenation will be 10 seconds long.

Audio files are concatenated after level adjustment. The concatenated files need to be 20 ms block aligned per audio track for further processing. Therefore, the concatenated sequence needs to be padded at the end with the first samples of the preamble to reach an integer number of 20 ms blocks for each audio track.

As reported in AHEVS-226, the VAD of AMR-WB might need an unexpected long adaptation time to produce reliable results. Therefore, the concatenated input sequence for the AMR-WB encoder conditions (AMR-WB codec, AMR-WB encoder/G.718-IO decoder, AMR-WB encoder/EVS-IO decoder) with DTX on in noisy speech experiments shall be extended by the repetition of the preamble and the concatenated speech files.

### Frame error application

In frame erasure conditions, the erasures shall affect the same segments of speech signal for the CuT and reference codecs. This shall be done by compensating for all encoder-side delays (or for all encoder-side delays of the core layer in case of embedded codecs). The CuT encoder and decoder executables shall compensate the delay of the output files. The delay of reference codecs shall be compensated prior to the processing by the reference encoder, as specified in 5.2.6.

Note: No delay compensation is applied for jitter related delay.

Note: Exact delay compensation might not be possible.

]

[

## Pre-processing

### Pre-processing stages



Figure 1: Common pre-processing for input audio

### High-pass filtering

Editor’s note: TBD

### Level adjustment



Figure 2: Level adjustment using BS.1770

## Processing for anchor conditions

### MNRU/ESDRU

Editor’s note: Rendering is currently not specified.

Note: ESDRU is only available for stereo/binaural audio.



Figure 3: Processing for MNRU/ESDRU conditions. Rendering is bypassed when audio format is already stereo/binaural/7.1+4.

]

[

## Processing for stereo inputs

### Reference conditions

Level normalization for stereo is assuming (virtual) loudspeakers at azimuth for both loudspeaker and binaural rendering.



Figure 8: Processing for stereo reference conditions.

### CuT conditions



Figure 9: Processing for stereo CuT conditions. Error insertion / Network simulation is bypassed for error-free/non-JBM conditions.

]

[

## Processing for binaural audio inputs

### Pre-processing

Editor’s Note: TBD, likely similar to stereo pre-processing.

### Reference conditions

Editor’s Note: Resampling step of the pre-processing may depend on the experiment.



Figure 11: Processing for binaural audio reference conditions.

### CuT conditions



Figure 12: Processing for binaural audio CuT conditions. Error insertion / Network simulation is bypassed for error-free/non-JBM conditions.

]

[

## Processing for multi-channel audio inputs

### Reference conditions

Editor’s Note: Resampling step of the pre-processing may depend on the experiment.

Editor’s note: Rendering is currently not specified.



Figure 14: Processing for multi-channel audio reference conditions

### CuT conditions

Editor’s note: Rendering is currently not specified.



Figure 15: Processing for multi-channel audio CuT conditions

## Processing for scene-based audio inputs

### Reference conditions

Editor’s Note: Resampling step of the pre-processing may depend on the experiment.

Editor’s note: Rendering is currently not specified.



Figure 16: Processing for scene-based audio reference conditions

### CuT conditions

Editor’s note: Rendering is currently not specified.



Figure 17: Processing for scene-based audio CuT conditions

## Processing for metadata-assisted spatial audio (MASA) inputs

Editor’s Note: Generation of 48 kHz input signals using IVAS MASA C Reference Software is missing.

### Reference conditions

Editor’s Note: Resampling step of the pre-processing may depend on the experiment.

Editor’s note: Rendering is currently not specified.



Figure 18: Processing for MASA reference conditions

### CuT conditions

Editor’s note: Rendering is currently not specified.

Figure 19: Processing for MASA CuT conditions

## Processing for object-based audio inputs

### Reference conditions

Editor’s Note: Resampling step of the pre-processing may depend on the experiment.

Editor’s note: Rendering is currently not specified.



Figure 20: Processing for object-based audio reference conditions

### CuT conditions

Editor’s note: Rendering is currently not specified. It may be performed by the IVAS decoder or be carried out by an external renderer.



Figure 21: Processing for object-based audio CuT conditions

]

# Processing Modules

This section specifies the operation of pre- and post-processing modules being utilized by the processing stages.

## Pre- and post-processing operations

[

### General delay compensation for the STL filter tool

All filtering steps include a delay compensation step. For preparing the delay compensation, samples of the preamble are added to the end of the input file before applying the filter step. After completion of the filtering step, the samples are to be removed from the beginning of the filtered file.

### Filtering operations

#### HP50\_48KHZ filtering

To produce a 50Hz high pass filtered 48kHz sampling file use:

filter.exe HP50\_48KHZ *input.48k* *output.48k* 960

#### MSIN filtering

To produce an MSIN high pass filtered 16kHz sampling file use:

filter.exe MSIN *input.16k* *output.16k* 320

]

### Level adjustment

[

#### P.56 active speech level adjustment

To normalize the P.56 ASL of an 8-kHz sampling file to -26 dBov, use:

sv56demo.exe -lev -26 -sf 8000 *input.8k* *output.8k* 160

To normalize the P.56 ASL of a 16-kHz sampled file to -26 dBov, use:

sv56demo.exe -lev -26 -sf 16000 *input.16k* *output.16k* 320

To normalize the P.56 ASL level of a 32-kHz sampled file to -26 dBov, use:

sv56demo.exe -lev -26 -sf 32000 *input.32k* *output.32k* 640

To normalize the P.56 ASL level of a 48-kHz sampled file to ‑26 dBov, use:

sv56demo.exe -lev -26 -sf 48000 *input.48k* *output.48k* 960

#### RMS Level adjustment

To normalize the RMS level of an 8-kHz sampled file, use:

sv56demo.exe –rms –lev xx –sf 8000 *input.8k* *output.8k* 160

To normalize the RMS level of a 16-kHz sampled file, use:

sv56demo.exe –rms –lev xx –sf 16000 *input.16k* *output.16k* 320

To normalize the RMS level of a 32-kHz sampled file, use:

sv56demo.exe –rms –lev xx –sf 32000 *input.32k* *output.32k* 640

To normalize the RMS level of a 48-kHz sampled file, use:

sv56demo.exe –rms -lev xx -sf 48000 *input.48k* *output.48k* 960

where xx is the desired level.

#### RMS Level measuring

To measure the RMS level of a MSIN filtered noise file, use:

actlev.exe –rms –blk 160 –sf 8000 *input.8k*

and extract the dBov value of the long-term energy (RMS).

]

#### BS.1770 Level adjustment

To normalize the level of a 48 kHz sampled file, use:

bs1770demo.exe –nchan N –lev L -conf xxxx *input.48k* *output.48k*

where N is the number of channels, L the target level in LKFS (default: -26), and xxxx a configuration string with one value per channel specifying channel weighting according to:

x = ’1’ loudspeaker position within |elevation| < 30 deg, 60 deg <= |azimuth| <= 120 deg,   
(weighted by 1.41)

x = ’L’ LFE channel (weighted by 0)

x = ’0’ otherwise (weighted by 1)

To normalize stereo or binaural to -26 LKFS:

bs1770demo.exe –nchan 2 –lev -26 -conf 00 *input.48k* *output.48k*

To normalize 7.1+4 multi-channel configuration, as specified in Table 5, to -26 LKFS:

bs1770demo.exe –nchan 12 –lev -26 -conf 000L00110000 *input.48k* *output.48k*

#### BS.1770 Level measuring

To measure the level of a 48 kHz sampled file, use:

bs1770demo.exe –nchan N -conf xxxx *input.48k*

where N is the number of channels, and xxxx a configuration string with one value per channel specifying channel weighting according to:

x = ’1’ loudspeaker position within |elevation| < 30 deg, 60 deg <= |azimuth| <= 120 deg,   
(weighted by 1.41)

x = ’L’ LFE channel (weighted by 0)

x = ’0’ otherwise (weighted by 1)

[

### Scaling by Factor F

To scale a noise signal at the sampling rate 8 kHz, 16 kHz, 32 kHz or 48 kHz by the Factor F use:

scaldemo.exe -dB -gain F -bits 16 -round -nopremask -blk BBB input output

where BBB is 160 for 8 kHz sampled files, 320 for 16 kHz sampled files, 640 for 32 kHz sampled files and 960 for 48 kHz sampled files. The Factor F is to be determined as TBD.

### Low/High level De-/Scaling

In case the speech level shall be -26 dBov, this processing step shall be bypassed. Otherwise, the following gain needs to be applied to the signal:

|  |  |  |
| --- | --- | --- |
| **Speech level** | **Scaling** | **Descaling** |
| -16 dBov | G=10 | G=-10 |
| -36 dBov | G=-10 | G=10 |

To apply the scaling operation with the gain G, use:

scaldemo.exe -dB -gain G -bits 16 -round –nopremask -blk BBB input output

where BBB is 160 for 8 kHz sampled files, 320 for 16 kHz sampled files, 640 for 32 kHz sampled files and 960 for 48 kHz sampled files.

### Summation of speech and noise files

#### Summation of a speech and a noise file

To add files in the same sampling frequency (8, 16 or 32 kHz), e.g., to produce a 32 kHz speech file mixed with noise file called *output.32k* by summing a 32 kHz speech file called *input.32k* and a 32 kHz noise file called *noise.32k,* use:

oper.exe –size 0 1 input.32k + 1 noise.32k 0 output.32k BBB

where BBB is 160 for 8 kHz sampled files, 320 for 16 kHz sampled files and 640 for 32 kHz sampled files.

### File concatenation

To concatenate files, the concat command is used:

concat.exe –undo undo\_concat.txt file1 [file2 file3 …] catfile

Where file1, file2, … are the files to be concatenated and catfile is the concatenated file. The undo\_concat.txt contains the parameters for segmentation.

In case additional silence sections needs to be inserted before concatenation, use the following for all input files

concat.exe silence fileX silence tmp\_file

copy tmp\_file fileX

where the silence file contains 0.2 sec of digital silence and fileX is one of the input files.

### Audio format conversion

#### PCM to WAV conversion

To convert a 32kHz sampled PCM file to a WAV file for AMR-WB+, use:

CopyAudio.exe -F WAVE-NOEX -P integer16,0,32000,native,1,default -I “” input.32k output\_32k.wav

#### WAV to PCM conversion

To convert a WAV file to a 32kHz sampled PCM file for AMR-WB+, use:

CopyAudio.exe -F noheader -D integer16 input\_32k.wav output.32k

### Sampling Rate changes (resampling)

#### Rate-change from 48- to 8-kHz sampling

To produce an 8-kHz sampling file from a 48-kHz sampling file, use:

filter.exe –down SHQ3 input.48k tmp.16k 960

filter.exe –down SHQ2 tmp.16k output.8k 320

#### Rate-change from 48- to 16-kHz sampling

To produce a 16-kHz sampling file from a 48-kHz sampling file, use:

filter.exe –down SHQ3 input.48k output.16k 960

#### Rate-change from 48- to 32-kHz sampling

To produce a 32 kHz sampling file from a 48 kHz sampling file, use:

filter.exe –up SHQ2 input.48k tmp.96k 960

filter.exe –down SHQ3 tmp.96k output.32k 1920

#### Rate-change from 8- to 48-kHz sampling

To produce a 48 kHz sampling file from an 8 kHz sampling file, use:

filter.exe –up SHQ2 input.8k tmp.16k 160

filter.exe –up SHQ3 tmp.16k output.48k 320

#### Rate-change from 16- to 48-kHz sampling

To produce a 48 kHz sampling file from a 16 kHz sampling file, use:

filter.exe –up SHQ3 input.16k output.48k 320

#### Rate-change from 32- to 48-kHz sampling

To produce a 48 kHz sampling file from a 32 kHz sampling file, use:

filter.exe –up SHQ3 input.32k tmp.96k 640

filter.exe –down SHQ2 tmp.96k output.48k 1920

#### Rate-change from 32- to 16-kHz sampling

To produce a 16 kHz sampling file from a 32 kHz sampling file, use:

filter.exe –down SHQ2 input.32k output.16k 640

#### Rate-change from 16- to 32-kHz sampling

To produce a 32 kHz sampling file from a 16 kHz sampling file, use:

filter.exe –up SHQ2 input.16k output.32k 320

#### Rate-change from 16- to 8-kHz sampling

To produce an 8 kHz sampling file from a 16 kHz sampling file, use:

filter.exe –down SHQ2 input.16k output.8k 320

### Windowing and segmentation

#### Segmentation for NB conditions

To extract an m sample long file beginning at sample s from an 8 kHz single channel concatenated file, use:

astrip.exe –sample –smooth –wlen 800 –start s -n m input.8k output.8k

#### Segmentation for WB conditions

To extract an m sample long file beginning at sample s from a 16 kHz single channel concatenated file, use:

astrip.exe –sample –smooth –wlen 1600 –start s -n m input.16k output.16k

#### Segmentation for SWB conditions

To extract an m sample long file beginning at sample s from a 32 kHz single channel concatenated file, use:

astrip.exe –sample –smooth –wlen 3200 –start s -n m input.32k output.32k

#### Initial windowing

To apply the initial windowing of a 48 kHz input speech or music file, use:

astrip.exe –sample –smooth –wlen 4800 –start s -n m input.48k output.48k

### Bit conversion

#### Conversion from 16 to 13 bits

To convert an 8kHz narrowband signal from a 16-bit representation to a 13-bit representation including rounding, use:

scaldemo.exe -lin -gain 1 -bits 13 -round -nopremask -blk 160 input.8k output.8k

#### Conversion from 16 to 14 bits

To convert a 16kHz wideband signal from a 16-bit representation to a 14-bit representation including rounding, use:

scaldemo.exe -lin -gain 1 -bits 14 -round -nopremask -blk 320 input.16k output.16k

### Interleave audio files

#### Interleave stereo files

To interleave stereo files, use:

stereoop.exe -interleave inputL inputR output

### De-interleave audio files

#### De-interleave stereo files

To de-interleave stereo files, use:

stereoop.exe -split input outputL outputR

]

## Processing

This section describes the main processing modules utilized by the processing stages.

[

### MNRU reference conditions

To generate an MNRU reference at XXX dB for a file *input.8k*, use:

mnrudemo.exe –Q XXX input.8k output.8k 160

To generate an MNRU reference at XXX dB for a file *input.16k*, use:

mnrudemo.exe –Q XXX input.16k output.16k 320

To generate a P.50 MNRU reference at XXX dB for a file *input.48k*, use:

p50mnru.exe input.48k output.48k XXX M

Note that the P.50 MNRU processing for SWB conditions requires rate-change steps between 32kHz and 48kHz.

]

### ESDRU reference conditions

To generate an ESDRU reference at distortion level alpha for a file *input*, use:

esdru.exe -e\_step 0.5 -seed 1 -sf fs alpha input output

where fs is the sample frequency, default 48 kHz, where and corresponds to the highest distortion level.

[

### Reference codecs for Narrowband conditions

#### AMR

To process a file *input.8k* through the AMR codec at XXX kbit/s, use:

amr\_cod\_vad2.exe [-dtx] BBB *input.8k* bitstream

amr\_dec.exe bitstream output.8k

where BBB is the bitrate mode corresponding to XXX as given in the following table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| XXX [kbit/s] | 7.4 | 7.95 | 10.2 | 12.2 |
| BBB | MR74 | MR795 | MR102 | MR122 |

Note that AMR operates at 20 ms frame length and has encoder and decoder delay of 5 ms and 0 ms, respectively

#### G.711

To process a file *input.8k* through the G.711codec with 20 ms blocks, use:

g711demo.exe LAW lili *input.8k* *output.8k* 160

where LAW is A for A-law and u for u-law. Note that G.711 does not have algorithmic delay.

#### G.718 narrowband mode

To process a file *input.8k* through the G.718 codec in narrowband mode at XXX bit/s, use:

g718\_enc.exe [-dtx 8] –maxBR XXX 8 *input.8k* bitstream

g718\_dec.exe -disabled\_NG –maxBR XXX 8 bitstream *output.8k*

where XXX is either 8000 or 12000. Note that G.718 operates at 20 ms frame length and has an algorithmic delay of 33.875 ms, when operating on narrowband input and in low-delay mode.

### Reference codecs for Wideband conditions

#### AMR-WB

To process a file *input.16k* through the AMR-WB codec at XXX kbit/s, use ITU-T G.722.2 as:

amrwb\_cod.exe [–dtx] –itu BBB *input.16k* bitstream

amrwb\_dec.exe -itu bitstream *output.16k*

where BBB is the bitrate mode corresponding to XXX as given in the following table.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| XXX [kbit/s] | 8.85 | 12.65 | 14.25 | 15.85 | 18.25 | 19.85 | 23.05 | 23.85 |
| BBB | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

Note that AMR-WB operates at 20 ms frame length and has encoder and decoder delay of 5 ms and 0.9375 ms, respectively.

#### AMR-WB encoder - G.718 IO decoder

To process a file *input.16k* through the AMR-WB encoder and the G.718 IO decoder at XXX bit/s, use ITU-T G.722.2 as:

amrwb\_cod.exe [–dtx] –itu BBB *input.16k* bitstream

where BBB is described in 5.2.4.1. For decoding call

g718\_dec.exe -disabled\_NG -IO\_G722\_2 –maxBR YYY 16 bitstream o*utput.16k*

where YYY is the bitrate mode corresponding to BBB as given in the following table.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| BBB | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| YYY | 8850 | 12650 | 14250 | 15850 | 18250 | 19850 | 23050 | 23850 |

Note that AMR-WB encoder operates at 20 ms frame length and has a delay of 5 ms. G.718 IO decoder operates at 20 ms frame length and has a delay of 1.9375 ms when operating on wideband input and in low-delay mode.

#### G.718

To process a file *input.16k* through the G.718 codec at XXX bit/s, use:

g718\_enc.exe –maxBR XXX 16 *input.16k* bitstream

g718\_dec.exe -disabled\_NG –maxBR XXX 16 bitstream *output.16k*

where XXX is 32000.

Note that G.718 operates at 20 ms frame length and has an overall algorithmic delay of 42.875 ms, when operating on wideband input.

#### G.722

To process a file *input.16k* through the G.722 codec at XXX kbit/s using 20 ms block size, use:

encg722.exe –fsize 320 –mode XXX *input.16k* bitstream

decg722.exe -fsize 320 –mode XXX bitstream *output.16k*

where XXX is either 56 or 64. Note that G.722 operates at 20 ms frame length and has an overall algorithmic delay of 1.625 ms.

#### G.722.1

To process a file *input.16k* through the G.722.1 codec at XXX bit/s, use:

g7221\_enc.exe 1 *input.16k* bitstream XXX 7000

g7221\_dec.exe 1 bitstream *output.16k* XXX 7000

where XXX is one of (24000, 32000). Note that G.722.1 operates at 20 ms frame length and has an overall algorithmic delay of 40 ms.

#### G.711.1

To process a file *input.16k* through the G.711.1 codec at 80 bit/s, use:

g7111\_enc.exe –mode 3 A *input.16k* bitstream

g7111\_dec.exe A –mode 3 bitstream *output.16k*

Note that G.711.1 operates at 20 ms frame length and has an overall algorithmic delay of 11.875 ms.

### Reference codecs for super-wideband conditions

#### AMR-WB+

To process a file *input\_32.wav* through the AMR-WB+ codec at XXX kbit/s, use:

amrwbplus\_cod.exe –rate XXX –ff raw –if *input\_32.wav* –of bitstream

amrwbplus\_dec.exe –ff raw –if bitstream –fs 32000 –mono –of *output\_32.wav*

where XXX is one of 9.75, 12, 16. Note that AMR-WB+ has an overall algorithmic delay of approx. 72/68 ms at 9.75/12kbps.

#### G.718 Annex B

To process a file *input.32k* through the G.718 Annex B codec at 36 kbit/s, use:

g718\_enc.exe –maxBR 36000 32 *input.32k* bitstream

g718\_dec.exe –disabled\_NG 32 bitstream *output.32k*

Note that G.718 Annex B operates at 20 ms frame length and has an overall algorithmic delay of 49.625 ms.

#### G.719

To process a file *input.48k* through the G.719 codec at XXX bit/s, use:

g719\_enc.exe –r XXX –i *input.48k* -o bitstream

g719\_dec.exe –i bitstream -o *output.48k*

where XXX is one of 32000, 48000, 56000, 64000, 80000, 96000, 128000. Note that G.719 operates at 20 ms frame length and has an overall algorithmic delay of 40 ms. G.719 operates at 48 kHz which implies resampling steps from and to 32 kHz.

#### G.722.1 Annex C

To process a file *input.32k* through the G.722.1 Annex C codec at XXX bit/s, use:

g7221\_enc.exe 1 *input.32k* bitstream XXX 14000

g7221\_dec.exe 1 bitstream *output.32k* XXX 14000

where XXX is one of 24000, 32000 or 48000. Note that G.722.1 Annex C operates at 20 ms frame length and has an overall algorithmic delay of 40 ms.

### Delay compensation for reference conditions

Details on the delay compensation implementation can be found in Annex A.1.

### EVS operation (non-JBM case)

To process a file *input.8k* through the EVS codec at XXX bit/s, use:

EVS\_cod.exe [-dtx] XXX/SWF 8 *input.8k* bitstream

EVS\_dec.exe 8 bitstream *output.8k*

where XXX is one of 5900, 7200, 8000, 9600 or 13200. For rate switching operation, XXX is replaced by a switching file (SWF).

To process a file *input.16k* through the EVS codec at XXX bit/s, use:

EVS\_cod.exe [-dtx] XXX/SWF 16 *input.16k* bitstream

EVS\_dec.exe 16 bitstream *output.16k*

where XXX is one of 5900, 7200, 8000, 9600, 13200, 16400, 24400, 32000, 48000, 64000, 96000 for testing non-IO modes or 6600, 8850, 12650, 14250, 15850, 18250, 19850, 23050, 23850 for testing AMR-WB IO modes. For rate switching operation, XXX is replaced by a switching file (SWF).

To process a file *input.32k* through the EVS codec at XXX bit/s, use:

EVS\_cod.exe [-dtx] XXX/SWF 32 *input.32k* bitstream

EVS\_dec.exe 32 bitstream *output.32k*

where XXX is one of 13200, 16400, 24400, 32000, 48000, 64000, 96000, 128000. For rate switching operation, XXX is replaced by a switching file (SWF).

The switching file consists of XXX values indicating the bit rate for each frame in bit/s. These values are stored in binary format using 4 bytes per value.

Note: EVS encoder and decoder provide delay compensated output files

### EVS operation (JBM case)

To process a file *input.8k* through the EVS codec at XXX bit/s, use:

EVS\_cod.exe [-dtx] XXX 8 *input.8k* bitstream

EVS\_dec.exe –Tracefile tracefile\_dec –VOIP 8 netsimoutput *output.8k*

where XXX is 9600.

To process a file *input.16k* through the EVS codec at XXX bit/s, use:

EVS\_cod.exe [-dtx] XXX 16 *input.16k* bitstream

EVS\_dec.exe –Tracefile tracefile\_dec –VOIP 16 netsimoutput *output.16k*

where XXX is 13200.

To process a file *input.32k* through the EVS codec at XXX bit/s, use:

EVS\_cod.exe [-dtx] XXX 32 *input.32k* bitstream

EVS\_dec.exe –Tracefile tracefile\_dec –VOIP 32 netsimoutput *output.32k*

where XXX is one of 13200 or 24400. The tracefile\_dec is the tracefile written by the decoder and netsimoutput is the RTP+G.192 input file written by the network simulator.

### CuT operation (non-JBM case)

Editor’s Note: TBD

### CuT operation (JBM case)

Editor’s Note: TBD

## Encoder and decoder CuT executable requirements

The executables shall be compatible to WIN32.

## Error Insertion (EID)

For the conditions where random frame erasures are desired, frame erasure patterns are applied to the bitstream using tools from the [ITU-T STL2009 library].

### Frame error tool

For the AMR codec, the following processing shall be used:

eid-amr.exe amrbsin ep.g192 amrbsout

where:

amrbsin is the AMR input bit stream

ep.g192 is the error pattern file

amrbsout is the AMR output bit stream

For the G.711 A/µ-law, the following processing shall be used:

eid-int –ep g192 –factor 2 ep.g192 ep10.g192

g711iplc.exe ep10.g192 input.8k output.8k

where:

ep.g192 is the error pattern file assuming 20ms frames and ep10.g192 is the error pattern file for the g711iplc tool where all entries are doubled to take the 10ms frame grid of the g711iplc tool into account.

input.8k is the G.711 decoded output file

output.8k is the G.711 decoded output file with packet loss concealment

For all other reference codecs and CuTs, the following processing shall be used:

eid-xor.exe –vbr –fer g192bsin ep.g192 g192bsout

where:

g192bsin is the input bit stream

ep.g192 is the error pattern file

g192bsout is the output bit stream

### Pattern generation

Before generating error patterns, the sta file of the gen-patt tool needs to be initialized as specified in Annex C.3.

The error patterns used are generated using the gen-patt tool as follows:

gen-patt.exe -tailstat -fer -g192 -gamma 0 -rate XXX -tol 0.001 -reset -n LENGTH -start 501 ep.g192

where XXX is the required erasure rate, i.e. 0.03 for 3% and 0.06 for 6% FER.

Different error patterns should be generated for each experiment. For all speech experiments the parameter LENGTH is exactly 12500, assuming a 10 sec preamble and 6 talker with 5 samples of 8 sec length each. For the mixed/music experiments, this parameter needs to be adapted to the length of the concatenated sequence.

For the AMR-WB encoder conditions with enabled DTX, the error pattern needs to be repeated due to the repetition of the input sequence. To generate an EPF with repetition, use:

concate.exe ep.g192 ep.g192 ep.repeated.g192

### Derive EPFs from jitter profiles

To derive an error pattern file from one of the delay and error profiles 1, 2, or 3 given in the MTSI specification, use:

dlyerr\_2\_errpat.exe –d 200 –f 1 –w –s YYY –i *dly\_err\_profile\_XXX.dat -o* f\_profile\_*XXX*.g192

where XXX is one of 1, 2, or 3, and YYY is the random offset into the profile.

To derive an error pattern file from one of the delay and error profiles 4 or 6 given in the MTSI specification, use:

dlyerr\_2\_errpat.exe –l 1 –f 1 –w –s YYY –i *dly\_err\_profile\_XXX.dat -o* f\_profile\_*XXX*.g192

where XXX is one of 4, or 6, and YYY is the random offset into the profile.

To derive an error pattern file from one of the delay and error profile 5 given in the MTSI specification, use:

dlyerr\_2\_errpat.exe –l 1 –f 2 –w –s YYY –i *dly\_err\_profile\_5.dat -o* f\_profile\_5.g192

where YYY is the random offset into the profile.

### Rate switching profile generation

To generate rate switching profiles for CuT conditions, use

gen-rate-profile.exe –layers B1,B2,…,Bx SWF 10 B1 LENGTH SEEDx

where B1,B2,…,Bx are all bit rates starting from the lowest one (B1) up to the highest one (Bx). The following list shows all possible rates for the CuT:

* no AMR-WB IO: 7200, 8000, 9600, 13200, 24400, 32000, 48000, 64000, 96000, 128000
* AMR-WB IO: 6600, 8850, 12650, 14250, 15850, 18250, 19850, 23050, 23850

SWF is the file name and SEEDx is a random seed, see C.2. For mixed/music experiments, the parameter LENGTH needs to be adapted to the length of the concatenated sequence. For speech experiments, the parameter LENGTH is exactly 12500.

To generate for instance the rate switching profile for a non-IO CuT condition from 9.6kbps to 64kbps, use

gen-rate-profile.exe –layers 9600,13200,16400,24400,32000,48000,64000 SWF 10 9600 LENGTH SEEDx

To generate a rate switching profiles consisting only of AMR-WB IO modes, use the Bx rates listed under AMR-WB IO. Mixing the non-IO and IO rates is required in case WB non-IO / IO switching is tested, e.g.to process a switched CuT non-IO / IO condition from 8.85kbps to 13.2kbps, use

gen-rate-profile.exe –layers 8850,9600,12650,13200 SWF 10 8850 LENGTH SEEDx

Note: Case B) AMR-WB encoder side rate switching is not supported so far

### Network simulator for packet jitter generation

To simulate packet jitter for CuT conditions, use

network\_simulator.exe dly\_error\_profile\_x.dat evsbitstream netsimoutput tracefile\_sim nFramesPerPacket [offset]

where

* dly\_error\_profile\_x.dat is the MTSI delay and error profile number x
* evsbitstream is the G.192 input bitstream file name
* netsimoutput is the RTP+G.192 output file name
* tracefile\_sim is the trace output file name of the network simulator
* nFramesPerPacket is the number of frames per packet (1, 2)
* offset is the shift/offset in delay and error profile in frames (default: 0)

### Cutting Tool for JBM tests

To trim parameters for the file segmentation after de-jittering, use

jbmtrim.exe –file Fs tracefile\_dec undo\_concat.txt undo\_concat\_trim.txt

where

* file is to enable the undo\_concat.txt support
* Fs is the operating sampling rate, i.e. 8000, 16000 or 32000
* tracefile\_dec is the trace output file of the CuT decoder
* undo\_concat.txt is the file name where parameters for segmentation are stored
* undo\_concat\_trim.txt is the file name containing the corrected parameters for segmentation using the astrip tool

]

# References

1. Pdoc IVAS-8a: “Test plan for selection phase”
2. Recommendation ITU-T G.192: “A common digital parallel interface for speech standardization activities”, March 1993
3. Recommendation ITU-T G.191 “Software tools for speech and audio coding standardization”, March 2010
4. TS 26.114, “IP Multimedia Subsystem (IMS); Multimedia Telephony; Media handling and interaction”
5. Pdoc IVAS-4: “IVAS Design Constraints”

[

1. External Resources
   1. Delay compensation for filter operations and reference codecs

The processing steps are delay-compensated in order to apply error insertion on the same parts of the audio signal and to be able to extract the original length and offset for each audio sample used in the tests.

The delay compensation is initialized by concatenating the file to be filtered and the first 960 samples of the preamble. After processing, the delay of the processing operation is compensated, and the original file length is restored.

To compensate the delay for filter operations and reference conditions for encoder and decoder in the common scripts, use:

astrip.exe –sample –start S+1 –n FILELENGTH input output

where FILELENGTH denotes the size in samples and the value for S for each filtering operation is given in Table 7 and for the reference conditions in Table 8.

Table 7: Delay compensation values for filter operations

|  |  |
| --- | --- |
| **Filter operation** | **Value for delay compensation after filtering operation** |
| -up SHQ2 | 436 |
| -up SHQ3 | 436 |
| -down SHQ2 | 218 |
| -down SHQ3 | 145 |
| HP50\_48KHZ | 839 |
| MSIN | 92 |

Table 8: Delay compensation values for reference codecs

| **Codec** | **Value for encoder delay compensation** | **Value for decoder delay compensation** |
| --- | --- | --- |
| AMR | 40 | 0 |
| AMR-WB | 80 | 15 |
| AMR-WB encoder – EVS IO decoder (Case B) | 80 | 0 |
| AMR-WB encoder - G.718 IO decoder | 80 | 0 |
| AMR-WB+ | - | 2314 (9.75kbps)  2187 (12kbps)  2187 (16kbps) |
| EVS | 0 | 0 |
| EVS IO encoder – AMR-WB decoder (Case A) | 0 | 15 |
| G.711 (A and µlaw) including the g711iplc tool | 0 | 0 |
| G.711.1 | 175 | 15 |
| G.718 | 0 | 0 |
| G.718 Annex B | 0 | 0 |
| G.718 narrow band | 0 | 0 |
| G.719 | 480 | Add 480 samples to beginning of decoded output |
| G.722 | 11 | 11 |
| G.722.1 | 160 | 160 |
| G.722.1 Annex C | 320 | 320 |

* 1. Binaries used

All binaries are compiled and tested under Win32 platforms. The following section documents the origin and the compilation of the binaries.

Editor’s Note: Usage rights of tools need review and updated rights may be needed for use in IVAS standardization.

* + 1. ITU-T STL processing tools

|  |  |
| --- | --- |
| Source | * ITU-T G.191 * S4-120344 “Filter masks for EVS testing” |
| URL | * https://www.itu.int/rec/T-REC-G.191-202207-I * http://ftp.3gpp.org/tsg\_sa/WG4\_CODEC/TSGS4\_68/Docs/S4-120344.zip |
| Version / Release | * G.191 (07/22) |
| Description | Software tools for speech and audio coding standardization |
| Comments | G.191 filter tool patched with S4-120344 to enable support for HP50 and SHQ filter |
| Executables | oper, astrip, concat, sv56demo, filter, scaldemo, actlev, eid-xor, gen-patt, mnrudemo, gen-rate-profile, eid-int, stereoop, bs1770demo, esdru |
| Status | Available |

* + 1. AMR Error insertion device

|  |  |
| --- | --- |
| Source | Orange: S4-120998 |
| URL | http://ftp.3gpp.org/tsg\_sa/WG4\_CODEC/TSGS4\_70/Docs/S4-120998.zip |
| Version / Release | - |
| Description | Error insertion device for AMR bit streams |
| Comments |  |
| Executables | eid-amr |
| Status | Available |

* + 1. Reference codecs
       1. AMR

|  |  |
| --- | --- |
| Source | 3GPP TS  [26.073](http://www.3gpp.org/ftp/Specs/archive/26_series/26.073): ANSI-C code for the Adaptive Multi Rate (AMR) speech codec |
| URL | http://www.3gpp.org/ftp/Specs/archive/26\_series/26.073/26073-a00.zip |
| Version / Release | 3GPP Rel. 10; Software version 5.1.0 March 26, 2003 |
| Description | AMR narrow band fix point encoder and decoder software |
| Comments | Compiled with VAD version 1 and 2 |
| Executables | amr\_cod\_vad1.exe, amr\_cod\_vad2.exe, amr\_dec.exe |
| Status | Available |

* + - 1. G.711

|  |  |
| --- | --- |
| Source | ITU-T G.191 |
| URL | <http://www.itu.int/rec/T-REC-G.191-201003-I> |
| Version / Release | Version v3.3 of 02.Feb. 2010 |
| Description | G.711 codec |
| Comments |  |
| Executables | g711demo.exe, g711iplc.exe |
| Status | Available |

* + - 1. AMR-WB

|  |  |
| --- | --- |
| Source | 3GPP TS  [26.173](http://www.3gpp.org/ftp/Specs/archive/26_series/26.173):  ANSI-C code for the Adaptive Multi-Rate - Wideband (AMR-WB) speech codec |
| URL | http://www.3gpp.org/ftp/Specs/archive/26\_series/26.173/26173-a00.zip |
| Version / Release | 3GPP Rel. 10; AMR Wideband Codec 3GPP TS26.190 / ITU-T G.722.2, Aug 25, 2003. Software version 7.0.0. |
| Description | AMR-WB fixed point encoder and decoder software |
| Comments |  |
| Executables | amrwb\_cod.exe, amrwb\_dec.exe |
| Status | Available |

* + - 1. G.722

|  |  |
| --- | --- |
| Source | ITU-T G.191 |
| URL | <http://www.itu.int/rec/T-REC-G.191-201003-I> |
| Version / Release | COPYRIGHT CNET LANNION A TSS/CMC Date 24/Aug/90  COPYRIGHT Ericsson AB. Date 22/May/06  COPYRIGHT France Telecom R&D Date 23/Aug/06 |
| Description | G.722 encoder and decoder software |
| Comments |  |
| Executables | encg722.exe, decg722.exe |
| Status | Available |

* + - 1. G.711.1

|  |  |
| --- | --- |
| Source | G.711.1: Wideband embedded extension for ITU-T G.711 pulse code modulation |
| URL | http://www.itu.int/rec/T-REC-G.711.1-201209-I/en |
| Version / Release | V2.00 / (09 / 2012) |
| Description | G.711.1 encoder and decoder software |
| Comments | Sub folder: G.711.1\_MB-Fix\_v2.00 |
| Executables | g7111\_enc.exe, g7111\_dec.exe |
| Status | Available |

* + - 1. G.722.1 (Annex C)

|  |  |
| --- | --- |
| Source | G.722.1 : Low-complexity coding at 24 and 32 kbit/s for hands-free operation in systems with low frame loss |
| URL | http://www.itu.int/rec/T-REC-G.722.1-200505-I/en |
| Version / Release | Release 2.1 (05/2005); Source folder: Fixed-200505-Rel.2.1 |
| Description | G.722.1 (Annex C) fixed point encoder and decoder software |
| Comments | Includes WB and SWB operations |
| Executables | g7221\_enc.exe, g7221\_dec.exe |
| Status | Available |

* + - 1. AMR-WB+

|  |  |
| --- | --- |
| Source | 3GPP TS  [26.273](http://www.3gpp.org/ftp/Specs/archive/26_series/26.273): ANSI-C code for the fixed-point Extended Adaptive Multi-Rate - Wideband (AMR-WB+) speech codec |
| URL | http://www.3gpp.org/ftp/Specs/archive/26\_series/26.273/26273-a00.zip |
| Version / Release | 3GPP Rel. 10; |
| Description | AMR-WBplus fixed point encoder and decoder software |
| Comments | er-libisomedia.dll required |
| Executables | amrwbplus\_cod.exe, amrwbplus\_dec.exe |
| Status | Available |

* + - 1. G.718 (Annex B)

|  |  |
| --- | --- |
| Source | G.718 : Frame error robust narrow-band and wideband embedded variable bit-rate coding of speech and audio from 8-32 kbit/s |
| URL | http://www.itu.int/rec/T-REC-G.718-201101-I!Cor3/en |
| Version / Release | Corrigendum 3; Approved 2011-01; Version v1.6; Software version v1.0; Source folder: G.718\_V1.6\_Part3\_swb\_mono\_AnnexB\_26072010 |
| Description | G.718 (Annex B) fixed point encoder and decoder software |
| Comments | Includes NB, WB and SWB codec |
| Executables | g718\_enc.exe, g718\_dec.exe |
| Status | Available |

* + - 1. G.719

|  |  |
| --- | --- |
| Source | G.719 : Low-complexity, full-band audio coding for high-quality, conversational applications |
| URL | http://www.itu.int/rec/T-REC-G.719-200806-I/en |
| Version / Release | Approved in 2008-06 including G.719 (2008) Amend.1 and G.719 (2008) Amend.2.; v1.0a (2008-04-16); Source folder: Fix-point-200806-Release-1.0a |
| Description | G.719 fixed point encoder and decoder software |
| Comments |  |
| Executables | g719\_enc.exe, g719\_dec.exe |
| Status | Available |

* + - 1. EVS

|  |  |
| --- | --- |
| Source | TBD, [3GPP TS 26.442/TS 26.443/TS 26.452: Codec for Enhanced Voice Services (EVS)] |
| URL | TBD |
| Version / Release | [Latest version available, TBD] |
| Description | EVS encoder and decoder software |
| Comments |  |
| Executables | EVS\_cod.exe, EVS\_dec.exe |
| Status | Available |

* + 1. Other tools
       1. Audio format converter

|  |  |
| --- | --- |
| Source | McGill University, Telecommunications & Signal Processing Laboratory, Audio File Programs and Routines |
| URL | https://www-mmsp.ece.mcgill.ca/Documents/Downloads/AFsp/AFsp-v10r4a.tar.gz |
| Version / Release | Release v10r4; Software version of CopyAudio: v6r0 2003-05-08 |
| Description | CopyAudio tool from the AFsp |
| Comments |  |
| Executables | CopyAudio.exe |
| Status | Available |

* + - 1. Network simulator

|  |  |
| --- | --- |
| Source | Fraunhofer |
| URL | <http://ftp.3gpp.org/tsg_sa/WG4_CODEC/TSGS4_76/Docs/S4-131277.zip> |
| Version / Release | - |
| Description | Jitter Simulator |
| Comments |  |
| Executables | network\_simulator.exe |
| Status | Available |

* + - 1. Jitter profile to EPF converter

|  |  |
| --- | --- |
| Source | Fraunhofer: S4-121077 |
| URL | http://ftp.3gpp.org/tsg\_sa/WG4\_CODEC/TSGS4\_70/Docs/S4-121077.zip |
| Version / Release | - |
| Description | Converts MTSI jitter profiles to error pattern for reference codecs |
| Comments |  |
| Executables | dlyerr\_2\_errpat.exe |
| Status | Available |

* + - 1. JBM trim tool

|  |  |
| --- | --- |
| Source | Fraunhofer: AHEVS-181 |
| URL | http://ftp.3gpp.org/tsg\_sa/WG4\_CODEC/Ad-hoc\_EVS/Docs/AHEVS-181.zip |
| Version / Release | - |
| Description | Tool for trim parameters for segmentation of samples |
| Comments |  |
| Executables | jbmtrim.exe |
| Status | Available |

* + - 1. AFR measure tool

|  |  |
| --- | --- |
| Source | Qualcomm, Fraunhofer |
| URL | Latest Executable: <http://ftp.3gpp.org/tsg_sa/WG4_CODEC/TSGS4_76/Docs/S4-131277.zip>  Documentation:  <http://ftp.3gpp.org/tsg_sa/WG4_CODEC/TSGS4_75/Docs/S4-131020.zip> |
| Version / Release | - |
| Description | Tool for measuring active frame ratio, bit rate and for identification of active frames |
| Comments |  |
| Executables | afr\_br.exe |
| Status | Available |

* + - 1. Jitter buffer verification tool

|  |  |
| --- | --- |
| Source | Fraunhofer: AHEVS-235 |
| URL | http://ftp.3gpp.org/tsg\_sa/WG4\_CODEC/ Ad-hoc\_EVS/ Docs/AHEVS-235.zip |
| Version / Release | V4.0 |
| Description | Tool for checking the objective performance requirements of the jitter buffer manager |
| Comments |  |
| Executables | checkObjectiveJbmPR.exe |
| Status | Available |

* + - 1. Gain check tool

|  |  |
| --- | --- |
| Source | VoiceAge: AHEVS-231 |
| URL | http://ftp.3gpp.org/tsg\_sa/WG4\_CODEC/ Ad-hoc\_EVS/Docs/AHEVS-231.zip |
| Version / Release | V3.0 |
| Description | Tool for verification of amplification of active and attenuation of background noise |
| Comments |  |
| Executables | gain\_chk\_v2.exe |
| Status | Available |

* + - 1. SWB MNRU

|  |  |
| --- | --- |
| Source | NTT: AHEVS-165 |
| URL | http://ftp.3gpp.org/tsg\_sa/WG4\_CODEC/Ad-hoc\_EVS/Docs/AHEVS-165.zip |
| Version / Release | - |
| Description | P.50 MNRU |
| Comments |  |
| Executables | p50mnru.exe |
| Status | Available |

* + - 1. Randomization tool

|  |  |
| --- | --- |
| Source | S4-121078 |
| URL | http://ftp.3gpp.org/tsg\_sa/WG4\_CODEC/TSGS4\_70/Docs/S4-121078.zip |
| Version / Release | - |
| Description | Tool for providing all randomizations depending on a master seed |
| Comments |  |
| Executables | random.exe |
| Status | Available |

* + - 1. MD5 checksum tool

|  |  |
| --- | --- |
| Source | GNU utilities for Win32 |
| URL | http://unxutils.sourceforge.net |
| Version / Release | 2.1 |
| Description | Tool for crosschecking common script result via MD5 hashes |
| Comments |  |
| Executables | md5sum.exe |
| Status | Available |

* + - 1. IVAS MASA C Reference Software

|  |  |
| --- | --- |
| Source | Nokia, Orange: S4-230221 |
| URL | https://www.3gpp.org/ftp/TSG\_SA/WG4\_CODEC/TSGS4\_122\_Athens/Docs/S4-230221.zip |
| Version / Release | - |
| Description | masaAnalyzer is the tool that takes in input signal and processes it into transport signals and metadata specified in MASA format.  masaRenderer is the tool that takes in the created MASA format signals and renders it to various listenable output configurations. |
| Comments |  |
| Executables | masaAnalyzer.exe, masaRenderer.exe |
| Status | Available |

1. Objective evaluation
   1. Databases for objective evaluation

For evaluation of any objective performance, the common database used in selection testing will be available subject to an appropriate legal framework. The database will consist of the following parts:

Table 9: Available databases for objective evaluation

|  |  |  |
| --- | --- | --- |
| **Database** | **Description** | **File name** |
| 1 |  |  |

* 1. Active Frame Ratio (AFR)

To measure the active frame ratio, use

afr\_br [OPTIONS] G192FILE

where the OPTIONS are:

-a needs to be enabled when the bit stream is in AMR format instead of G.192

-s SID\_SIZE specifies one valid SID\_SIZE in bits (up to five)

-f SID\_FILE specifies a SID\_FILE with valid SID frame sizes (up to five) as one line per size

-p PREAMBLE is the number of PREAMBLE frames (each 20ms) to skip

-b BR\_IN\_FILE input file BR\_IN\_FILE which contains the bit rates as external input

-v VAD\_IN\_FILE input file VAD\_IN\_FILE which contains VAD decisions as external input

-V VAD\_OUT\_FILE output VAD\_OUT\_FILE file which contains VAD decisions based on SID sizes

-o RESULT\_FILE output file RESULT\_FILE which contains the textual result

-O RESULT\_BR\_F output file RESULT\_BR\_F which contains the bit rate mismatches

The AFR is measured separately on all noisy speech databases as defined in Section B.1.

* 1. Gain measuring of active frames and inactive frames

Editor’s Note: To be updated based on design constraint on “Output gain tolerance”, see IVAS-4 [5].

To identify the active signal parts of the objective database, the AMR-WB VAD is to be used. Use

amrwb\_cod.exe –dtx –itu 2 EVSDB\_clean\_speech.16k cleanspeech\_bs

to generate the AMR-WB bit stream for the clean speech database. The delay compensation for the AMR-WB needs to be taken into account. To extract the information on the active frames, use

afr\_br.exe –s 35 –p 500 –V VAD\_f\_cleanspeech cleanspeech\_bs

where VAD\_f\_cleanspeech is the file name where VAD flags per frame are stored.

Process the databases by the required CuT conditions. Before starting to measure, strip the 10 sec preamble off the original database and the processed database.

To measure the amplification level of active signals, use the following command line:

gain\_chk -i original\_DB -o processed\_DB -t results\_f -r Fs   
[–v VAD\_f\_cleanspeech]

where

* original\_DB is one of the databases defined in B.1.
* processed\_DB is the original\_DB processed with one CuT condition
* results\_f is the results file to be reported for all DB files and CuT conditions
* Fs is the sampling frequency
* VAD\_f\_cleanspeech is the VAD file to be generated on the clean speech database. This file is required for the evaluation of clean speech and noisy speech. For mixed content and music, this option needs to be disabled.

The results contain two parts, one for the active frames and one for the inactive frames.

* 1. JBM performance evaluation

To evaluate the performance of the EVS JBM solution, it is recommended to use

checkObjectiveJbmPR.exe -v dly\_error\_profile\_x.dat nFramesPerPacket tracefile\_sim tracefile\_dec

where

* dly\_error\_profile\_x.dat is the MTSI delay and error profile number x
* nFramesPerPacket is the number of frames per packet (1, 2)
* tracefile\_sim is the trace output file name of the network simulator
* tracefile\_dec is the trace output file of the CuT decoder
  1. Bitrate evaluation

To evaluate the bitrate of the CuT, it is recommended to use the AFR tool as specified in Section B.2. This tool reports on the bit rate of a bitstream. The preamble shall be excluded from calculation.

1. Randomization scripts

The randomization scripts require one master seed, which will be provided by the HL. The master seed (MASTER\_SEED) determines all parameters for noise files, EPFs, jitter and rate switching.

* 1. Delay and error profile offset

As the JBM profiles can be looped, the range should be the entire profile, i.e. 7500 or 8000 entries.

To randomly select an JBM offset for profiles 1-6, use:

random -n 1 -s MASTER\_SEED –d PROFILE\_NUMBER –r 0 7499

where PROFILE\_NUMBER is the number of the profile for profiles 1-6.

To randomly select an JBM offset for profiles 7-10, use:

random -n 1 -s MASTER\_SEED –d PROFILE\_NUMBER –r 0 7999

where PROFILE\_NUMBER is the number of the profile for profiles 7-10.

* 1. Seeds for rate switching

The seeds to generating the rate switching profiles (Section 5.4.4) can be obtained by

* SEEDx = MASTER\_SEED
  1. Initialization of gen-patt tool

In order to initialize the gen-patt tool, an *sta file* needs to written. Therefore, two templates are provided:

Template for 6% frame error rate:

EID

BER = 0.060000

GAMMA = 0.000000

RAN-seed = 0x176d71ac

Current State = G

GOOD->GOOD = 0.880000

GOOD->BAD = 1.000000

BAD ->GOOD = 0.880000

BAD ->BAD = 1.000000

Template for 3% frame error rate:

EID

BER = 0.030000

GAMMA = 0.000000

RAN-seed = 0xab5d4825

Current State = G

GOOD->GOOD = 0.940000

GOOD->BAD = 1.000000

BAD ->GOOD = 0.940000

BAD ->BAD = 1.000000

where the RAN-seed needs to be replaced in HEX format. To generate these numbers, use

random -n 1 -s MASTER\_SEED –d PRERUN\_EXP\_RATE –r 0 99999999

where PRERUN\_EXP\_RATE for each EPF and experiment is given the table below.

by the following numbers:

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment** | **EPF for 3%** | **EPF for 6%** | **EPF for 10%** |
|  |  |  |  |

1. Additional Delay and Loss Profiles for Characterization

A number of VoLTE RTP packet logs under different network conditions were analyzed and the set of delay loss profiles dlp7, dlp8, dlp9 and dlp10 was derived to reflect the properties of the analyzed logs. The VoLTE logs were captured in different operator networks across different regions (South Korea and US). While different VoLTE network operating parameters affect the channel error characteristics and therefore the profile generation, it is noted that certain parameters such as the delay jitter, error burstiness, and jitter temporal evolution have more bearing on the VoLTE error profile generation. These VoLTE parameters are described below in the process of dlp7 through dlp10 generation.

* + 1. Delay Jitter

The delay jitter is a characteristic of the VoLTE network under consideration. Jitter of the arrived packets depends on factors such as network buffering characteristics, scheduler delays, network load, fading characteristics and mobility of the UE. The logs were derived to reflect on the typical characteristics of VoLTE networks by extracting the delay jitter from four different real world VoLTE logs under different network conditions. In particular, typical characteristics such as random jitter, ramp up and ramp down of jitter and jitter under increased network loss were included in the four delay loss profiles.

The delay jitter histograms in Figure 23 show the different delay jitter distributions of the four delay loss profiles derived from VoLTE logs under challenging radio conditions. Figure 24 shows the first 1000 elements of the proposed profiles.

Figure 22: Delay histograms for delay loss profiles 7, 8, 9, and 10

* + 1. Error burstiness

The error burstiness varies greatly with different channel conditions. The typical error burstiness of VoLTE logs under impaired channel conditions has been analyzed and is reflected in the four delay loss profiles from VoLTE logs under challenging radio conditions; the most frequent burst length varies for different profiles and that is 1, 2 or 3 in most cases.



Figure 23: Delay variation for delay loss profiles 7, 8, 9, and 10

The graphs show only first 1000 entries in each delay loss profile due to space limitations. Note that the “-1” entries represent network losses.

]