3GPP TS 26.443 V16.3.0 (2021-12)

Technical Specification

3rd Generation Partnership Project;

Technical Specification Group Services and System Aspects;

Codec for Enhanced Voice Services (EVS);

ANSI C code (floating-point)

(Release 16)

** 

The present document has been developed within the 3rd Generation Partnership Project (3GPP TM) and may be further elaborated for the purposes of 3GPP..  
The present document has not been subject to any approval process by the 3GPPOrganizational Partners and shall not be implemented.  
This Specification is provided for future development work within 3GPPonly. The Organizational Partners accept no liability for any use of this Specification.  
Specifications and Reports for implementation of the 3GPP TM system should be obtained via the 3GPP Organizational Partners' Publications Offices.

Keywords

UMTS, LTE, EVS, telephony, codec

***3GPP***

Postal address

3GPP support office address

650 Route des Lucioles - Sophia Antipolis

Valbonne - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Internet

http://www.3gpp.org

***Copyright Notification***

No part may be reproduced except as authorized by written permission.  
The copyright and the foregoing restriction extend to reproduction in all media.

© 2021, 3GPP Organizational Partners (ARIB, ATIS, CCSA, ETSI, TSDSI, TTA, TTC).

All rights reserved.

UMTS™ is a Trade Mark of ETSI registered for the benefit of its members

3GPP™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners  
LTE™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners

GSM® and the GSM logo are registered and owned by the GSM Association

Contents

Foreword [4](#__RefHeading___Toc26263316)

1 Scope [5](#__RefHeading___Toc26263317)

2 References [5](#__RefHeading___Toc26263318)

3 Definitions and abbreviations [6](#__RefHeading___Toc26263319)

3.1 Definitions [6](#__RefHeading___Toc26263320)

3.2 Abbreviations [6](#__RefHeading___Toc26263321)

4 C code structure [6](#__RefHeading___Toc26263322)

4.1 Contents of the C source code [7](#__RefHeading___Toc26263323)

4.2 Program execution [7](#__RefHeading___Toc26263324)

5 File formats [7](#__RefHeading___Toc26263325)

5.1 Input/output file format [7](#__RefHeading___Toc26263326)

5.2 Rate switching profile (encoder input) [8](#__RefHeading___Toc26263327)

5.3 Parameter bitstream file (encoder output / decoder input) [8](#__RefHeading___Toc26263328)

5.3.1 ITU-T G.192 compliant format [8](#__RefHeading___Toc26263329)

5.3.2 Compact storage format file [8](#__RefHeading___Toc26263330)

5.4 VoIP parameter bitstream file (decoder input) [8](#__RefHeading___Toc26263331)

5.5 Bandwidth switching profile (encoder input) [9](#__RefHeading___Toc26263332)

5.6 Channel-aware configuration file (encoder input and decoder output) [9](#__RefHeading___Toc26263333)

5.7 JBM trace file (decoder output) [9](#__RefHeading___Toc26263334)

Annex A (informative): Change history [10](#__RefHeading___Toc26263335)

# Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

x the first digit:

1 presented to TSG for information;

2 presented to TSG for approval;

3 or greater indicates TSG approved document under change control.

y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.

z the third digit is incremented when editorial only changes have been incorporated in the document.

# 1 Scope

The present document contains an electronic copy of the ANSI C floating-point code for the Enhanced Voice Services (EVS) Codec. This ANSI C code is the unique alternative reference specification besides the ANSI-C fixed-point code for the EVS Codec (3GPP TS 26.442) for a standard compliant implementation of the EVS Codec (3GPP TS 26.445), Voice Activity Detection (VAD) (3GPP TS 26.451), Comfort Noise Generation (CNG) (3GPP TS 26.449), Discontinuous Transmission (DTX) (3GPP TS 26.450), Packet Loss Concealment (PLC) of Lost Packets (3GPP TS 26.447), Jitter Buffer Management (JBM) (3GPP TS 26.448), and AMR-WB Interoperable Function (3GPP TS 26.446).

The bit-exact fixed-point ANSI-C code in 3GPP TS 26.442 or alternatively in 3GPP TS 26.452 remain the preferred implementation for all applications, but the floating-point codec may be used instead of the fixed-point codec when the implementation platform is better suited for a floating-point implementation.

Requirements for any implementation of the EVS codec to be standard compliant are specified in 3GPP TS 26.444 (Test sequences).

# 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 26.445: "Codec for Enhanced Voice Services (EVS); Detailed Algorithmic Description".

[3] 3GPP TS 26.451: "Codec for Enhanced Voice Services (EVS); Voice Activity Detection (VAD)".

[4] 3GPP TS 26.449: "Codec for Enhanced Voice Services (EVS); Comfort Noise Generation (CNG) Aspects".

[5] 3GPP TS 26.450: "Codec for Enhanced Voice Services (EVS); Discontinuous Transmission (DTX)".

[6] 3GPP TS 26.447: "Codec for Enhanced Voice Services (EVS); Error Concealment of Lost Packets".

[7] 3GPP TS 26.448: "Codec for Enhanced Voice Services (EVS); Jitter Buffer Management".

[8] 3GPP TS 26.446: "Codec for Enhanced Voice Services (EVS); AMR-WB Backward Compatible Functions".

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

[10] Recommendation ITU-T G.191 (03/10): "Software tools for speech and audio coding standardization".

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

# 3 Definitions and abbreviations

## 3.1 Definitions

For the purposes of the present document, the definitions given in TR 21.905 [1] apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in TR 21.905 [1].

## 3.2 Abbreviations

For the purposes of the present document, the abbreviations given in TR 21.905 [1] and the following apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in TR 21.905 [1].

ACELP Algebraic Code-Excited Linear Prediction

AMR-WB Adaptive Multi Rate Wideband (codec)

CNG Comfort Noise Generator

DTX Discontinuous Transmission

EVS Enhanced Voice Services

FB Fullband

FEC Frame Erasure Concealment

IP Internet Protocol

JBM Jitter Buffer Management

MSB Most Significant Bit

MTSI Multimedia Telephony Service for IMS

NB Narrowband

PS Packet Switched

PSTN Public Switched Telephone Network

SAD Sound Activity Detection

SC-VBR Source Controlled - Variable Bit Rate

SID Silence Insertion Descriptor

SWB Super Wideband

VAD Voice Activity Detection

WB Wideband

WMOPS Weighted Millions of Operations Per Second

# 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.

The ANSI‑C code has been verified on the following platforms:

- IBM PC compatible computers with Windows 7 operating systems and Microsoft Visual C++ 2010 compiler, 32-bit.

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

## 4.1 Contents of the C source code

The C code is organized as follows:

Table 1: Source code directory structure

|  |  |
| --- | --- |
| Directory | Description |
| README.txt | information on how to compile |
| Makefile | UNIX style encoder Makefile |
| Workspace\_msvc/ | Directory for the MSVC 2008 (or newer) project files |
| 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 |

The distributed files with suffix "c" contain the source code and the files with suffix "h" are the header files. The ROM data is contained in files named "rom\_xxx" 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 EVS\_cod) and the decoder (named EVS\_dec).

## 4.2 Program execution

The codec for Enhanced Voice Services is implemented in two programs:

*-* EVS\_cod: encoder;

*-* EVS\_dec: decoder.

The programs should be called like:

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

- EVS\_dec [decoder options]<bitstream file> < output file>.

The input and output files contain 16-bit linear encoded PCM samples and the bitstream file contains encoded data.

The encoder and decoder 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 *encoder* and *decoder* programs.

# 5 File formats

This clause describes the file formats used by the encoder and decoder programs. The test sequences defined in [1] also use the file formats described here.

## 5.1 Input/output file format

Input files read by the encoder and output files written by the decoder consist of 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.). Both the encoder and the decoder program process complete frames corresponding to multiples of 20 ms. The remaining samples are discarded.

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.

## 5.2 Rate switching profile (encoder input)

The encoder program can optionally read in a rate switching profile file which specifies the encoding bitrate for each frame of the input data. The rate switching profile is a binary file, generated by 'gen-rate-profile' tool, which is part of STL 2009, as contained in ITU-T G.191 [10]. The rate switching profile contains 32-bit integer words where each word represents the encoding bitrate for each particular frame. The rate switching profile is recycled if it contains less entries than the total number of frames in the input file. The rate switching profile can contain EVS primary mode bitrates and AMR-WB IO mode bitrates arbitrarily. I.e. switching between the two modes can be specified by the rate switching profile.

## 5.3 Parameter bitstream file (encoder output / decoder input)

The files produced by the speech/audio encoder/expected by the speech decoder contain an arbitrary number of frames in the following available formats.

### 5.3.1 ITU-T G.192 compliant format

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| SYNC\_WORD | DATA\_LENGTH | B1 | B2 | … | Bnn |

Each box corresponds to one Word16 value in the bitstream file, for a total of 2+nn words or 4+2nn bytes per frame, where nn is the number of encoded bits in the frame. Each encoded bit is represented as follows: Bit 0 = 0x007f, Bit 1 = 0x0081. The fields have the following meaning:

- SYNC\_WORD: Word to ensure correct frame synchronization between the encoder and the decoder. It is also used to indicate the occurrences of bad frames.

In the encoder output: (0x6b21)

In the decoder input: Good frames (0x6b21)

Bad frames (0x6b20)

- DATA\_LENGTH: Length of the speech data. Codec mode and frame type is extracted in the decoder using this parameter

### 5.3.2 Compact storage format file

The encoder and decoder programs can optionally write and read a file in the octet-based compact storage format. The compact storage format is specified in Annex A.2.6 of [2].

## 5.4 VoIP parameter bitstream file (decoder input)

|  |  |  |  |
| --- | --- | --- | --- |
| Packet size | Arrival time | RTP header | G.192 format (see 6.3.1) |

The fields have the following size and meaning:

- Packet size: 32 bit unsigned integer (= 12 + 2 + DATA\_LENGTH).

- Arrival time: 32 bit unsigned integer in ms.

- RTP header: 96 bits (see RFC 3550 [9]), including RTP timestamp and SSRC.

## 5.5 Bandwidth switching profile (encoder input)

The encoder program can optionally read in a bandwidth switching profile, which specifies the encoding bandwidth for each frame of speech processed. The file is a text file where each line contains "nb\_frames B". B specifies the signal bandwidth that is one of the supported four bandwidths, i.e. NB, WB, SWB or FB. And "nb\_frames" is an integer number of frames and specifies the duration of activation of the accompanied signal bandwidth B.

## 5.6 Channel-aware configuration file (encoder input and decoder output)

The encoder program can optionally read in a configuration file which specifies the values of FEC indicator p and FEC offset o, where FEC indicator, p: LO or HI, and FEC offset, o: 2, 3, 5, or 7 in number of frames. Each line of the configuration file contains the values of p and o separated by a space.

The channel-aware configuration file is meant to simulate channel feedback from a receiver to a sender, i.e. the decoder would generate FEC indication and FEC offset values for receiver feedback that correspond to the current transmission channel characteristics, thereby allowing optimization of the transmission by the encoder which applies the FEC offset and FEC indication when in the channel-aware mode.

## 5.7 JBM trace file (decoder output)

The decoder can generate a JBM trace file with the –Tracefile switch as a by-product of the decoder operation in case of JBM operation (which is triggered with the –VOIP switch on the decoder side).

The trace file is a CSV file with semi-colon as separator. The trace file starts with one header line that contains the column names in the following order:

rtpSeqNo;rtpTs;rcvTime;playtime;active

For each played out speech frame one entry is written to the trace file. The interval of the playtime values is usually 20ms, but may differ, depending on the JBM operation. Each entry is a line in the trace file that contains values as specified in Table 1.

Table 1: JBM trace file entry format

|  |  |  |
| --- | --- | --- |
| Name | Unit | Description |
| rtpSeqNo | 1 | RTP sequence number of played out speech frame. -1 if no corresponding RTP packet for the speech frame exists. |
| rtpTs | ms | RTP time stamp of played out speech frame. -1 if no corresponding RTP packet for the speech frame exists |
| rcvTime | ms | Absolute reception time of the RTP packet that corresponds to the speech frame. -1 if no corresponding RTP packet for the speech frame exists. |
| playtime | ms | Absolute play time (i.e. the time at which the PCM data is made available by the decoder). Can be floating-point value. |
| active | 0 or 1 | Binary entry, which is set to 1 for active speech frames (i.e. frames that are neither SID nor NO\_DATA) |

Annex A (informative):  
Change history

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Change history** | | | | | | | |
| **Date** | **Meeting** | **TDoc** | **CR** | **Rev** | **Cat** | **Subject/Comment** | **New version** |
|  |  |  |  |  |  |  |  |
| 2014-12 | SA#66 | SP-140729 |  |  |  | Presented to TSG SA#66 for approval | 1.0.0 |
| 2014-12 | SA#66 |  |  |  |  |  | 12.0.0 |
| 2015-03 | SA#67 | SP-150084 | 0001 | 1 |  | Corrections to the text of the specification | 12.1.0 |
| 2015-03 | SA#67 | SP-150084 | 0002 | 2 |  | Bugfixes to EVS Floating Point Source Code | 12.1.0 |
| 2015-03 | SA#67 | SP-150084 | 0003 | 3 |  | Implementation of the compact storage format in the EVS Floating Point Source Code | 12.1.0 |
| 2015-06 | SA#68 | SP-150201 | 0005 | - |  | Bugfixes to EVS Floating Point Source Code | 12.2.0 |
| 2015-09 | SA#69 | SP-150434 | 0006 | 1 |  | Corrections to EVS Floating-Point Source Code | 12.3.0 |
| 2015-12 | SA#70 | SP-150639 | 0007 | - |  | Corrections to EVS Floating-Point Source Code | 12.4.0 |
| 2015-12 | SA#70 |  |  |  |  | Version for Release 13 | 13.0.0 |
| 2016-03 | SA#71 | SP-160064 | 0009 | - |  | Corrections to EVS Floating-Point Source Code | 13.1.0 |
| 2016-06 | SA#72 | SP-160257 | 0011 | - | A | Corrections to EVS Floating-Point Source Code | 13.2.0 |
| 2016-09 | SA#73 | SP-160589 | 0014 | - | A | Corrections to EVS Floating-Point Source Code | 13.3.0 |
| 2016-09 | SA#73 | SP-160589 | 0016 | - | A | Corrections to EVS Floating-Point Source Code | 13.3.0 |
| 2017-03 | SA#75 | SP-170019 | 0018 | - | A | Corrections to EVS Floating-Point Source Code | 13.4.0 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Change history** | | | | | | | |
| **Date** | **Meeting** | **TDoc** | **CR** | **Rev** | **Cat** | **Subject/Comment** | **New version** |
| 2017-03 | 75 |  |  |  |  | Version for Release 14 | 14.0.0 |
| 2017-12 | 78 | SP-170820 | 0020 | 2 | A | Corrections to EVS Floating-Point Source Code | 14.1.0 |
| 2018-06 | 80 | SP-180261 | 0024 | - | A | Corrections to EVS Floating-Point Source Code | 14.2.0 |
| 2018-06 | 80 |  |  | - |  | Version for Release 15 | 15.0.0 |
| 2018-12 | 82 | SP-180965 | 0028 | - | A | Corrections to EVS Floating-Point Source Code | 15.1.0 |
| 2019-12 | 86 | SP-191000 | 0029 | 1 | F | Correction of Scope | 16.0.0 |
| 2020-06 | SA#88-e | SP-200585 | 0033 | 1 | A | Corrections to EVS Floating-Point Source Code | 16.1.0 |
| 2021-09 | SA#93-e | SP-210825 | 0040 | 1 | A | Corrections to EVS Floating-Point Source Code | 16.2.0 |
| 2021-12 | SA#94-e | SP-211345 | 0045 | - | A | Corrections to EVS Floating-Point Source Code | 16.3.0 |