



# Experimental AMR-WB Codec

(NON-COMMERCIAL use only)

## Technical Documentation

Version 1.1 Revision A  
February 2024

CodecPro Incorporated  
2162 Laurier East  
Montreal, QC H2H 1C2 CANADA  
[sales@codecpro.com](mailto:sales@codecpro.com)  
[www.codecpro.com](http://www.codecpro.com)

## Contents

<b>Revision history .....</b>	<b>4</b>
<b>References .....</b>	<b>4</b>
<b>Experimental AMR-WB Codec .....</b>	<b>5</b>
<b>AMR-WB Codec .....</b>	<b>5</b>
<b>Data input/output format .....</b>	<b>6</b>
Bitstream frame layout.....	6
Byte 0 layout.....	6
Modes .....	7
Discontinuous Transmission (DTX) mode.....	7
Bad Frame Indicator (BFI) .....	7
<b>Package Contents.....</b>	<b>8</b>
<b>AMR-WB API functions .....</b>	<b>9</b>
E_IF_amrwb_init .....	9
E_IF_amrwb_encode .....	9
E_IF_amrwb_exit .....	10
D_IF_amrwb_init .....	11
D_IF_amrwb_decode .....	11
D_IF_amrwb_exit .....	12

## Revision history

November 2013      First release of this document.

February 2024      Changed package content section to reflect the use of Visual Studio 2022.

## References

- [1] 3GPP 2001 TS 26.171, "Speech codec speech processing functions; Adaptive Multi-Rate - Wideband (AMR-WB) speech codec; General description." <http://www.3gpp.org/ftp/Specs/html-info/26171.htm>.
- [2] ITU-T 2002 G.722.2, "Wideband coding of speech at around 16 kbit/s using Adaptive Multi-Rate Wideband (AMR-WB)." <http://www.itu.int/rec/recommendation.asp?type=folders&lang=e&parent=T-REC-G.722.2>
- [3] IETF (2007) RFC 4867, "Real-Time Transport Protocol (RTP) Payload Format and File Storage Format for the Adaptive Multi-Rate (AMR) and Adaptive Multi-Rate Wideband (AMR-WB) Audio Codec." <http://www.ietf.org/rfc/rfc4867.txt>

## Experimental AMR-WB Codec

We have chosen to offer a free AMR-WB codec implementation that enables the developer community to better appreciate the strengths and capabilities of this technology.

The source code published by the standardization body is copyrighted by many organizations. Consequently, contributions to open forums are made in the form of object code, under specific licenses that place certain restrictions on the object code but do not prevent the implementation from being used in harmony with the typical open-source project licenses.

Please note that this Experimental AMR-WB Codec is intended for personal or community experimentation usage. It cannot be used for commercial purposes as specified in the license terms to which it was agreed to during the download process from CodecPro's web site. For a commercial version please contact CodecPro at [sales@codecpro.com](mailto:sales@codecpro.com).

## AMR-WB Codec

The implementation provided in this package is the Adaptive Multi-Rate Wideband (AMR-WB) encoder/decoder.

AMR-WB is the mandatory standard codec for wideband speech in GSM and WCDMA networks. It is also standardized in ITU-T as Recommendation G.722.2, and is included as optional in the CableLabs® PacketCable™ 2.0 specification. AMR-WB/G.722.2 interoperates with VMR-WB, which was standardized by 3GPP2 for wideband telephony and multimedia streaming services in cdma2000® networks.

The AMR-WB speech codec consists of nine speech codec modes with bit rates of 23.85, 23.05, 19.85, 18.25, 15.85, 14.25, 12.65, 8.85 and 6.6 kbps. It also includes VAD/DTX/CNG (voice activity detection/discontinuous transmission/comfort noise generation) features for increased efficiency.

## Data input/output format

The input to the encoder and output from the decoder is 16-bit linear PCM speech, sampled at 16 kHz.

## Bitstream frame layout

The output from the encoder and input to the decoder use the bitstream format described in RFC 4867 (Sections 5.1 and 5.3) and illustrated in the following graphic:

Coding mode and Quality indicator	Bitstream data
(1 byte)	(size depends on mode)

The first byte of every frame (Byte 0) contains the coding mode and a quality indicator. The rest of the bytes in the frame are the encoded speech bitstream.

## Byte 0 layout

Byte 0 contains the coding mode in bits 1 to 4 and the quality of this frame (0 for bad frame or 1 for good frame) in bit 5. The other bits in Byte 0 are not used.

Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
Pad	——	Mode	——	Quality	Pad	Pad	

## Modes

Possible mode values are 0-8, 9 and 15. Each mode has a corresponding number of bytes per frame, as shown in the following table. Modes 9 and 15 are automatically generated by the encoder when Voice Activity Detection (VAD) is enabled. Modes 10-14 are not used.

Mode	Bit Rate	Bytes per frame (zero-padded)
0	6.60 kbps	18
1	8.85 kbps	24
2	12.65 kbps	33
3	14.25 kbps	37
4	15.85 kbps	41
5	18.25 kbps	47
6	19.85 kbps	51
7	23.05 kbps	59
8	23.85 kbps	61
9	SID	6
15	No data	1

## Discontinuous Transmission (DTX) mode

In a typical telephone conversation, voice transmission alternates frequently between the speaking parties, leaving long pauses of silence. These can be more efficiently represented as background noise that is transmitted at a much lower bit rate. The discontinuous transmission mode is used to encode frames that contain only background noise.

When AMR-WB operates in DTX mode, a voice activity detector (VAD) on the transmission (TX) side evaluates whether a frame contains any voice data. In the absence of speech, a silence information descriptor (SID) frame, which contains characteristics describing the background noise, is transmitted. On the reception (RX) side, a comfort noise generator (CNG) is used to synthesize background noise based on the SID frame parameters. On the TX side, the encoder generates No data frames until it detects a change in the input signal (as background noise or speech).

## Bad Frame Indicator (BFI)

To hide the effects of a corrupted or lost speech frame, the decoder can perform error concealment, using information from previous good frames. The `iBadFrame` argument is used to indicate the quality of each frame to the decoder and to enable error concealment. (See [D\\_IF\\_amrwb\\_decode](#) on page 11.)

## Package Contents

cp_amrwb.lib	Single channel, 64-bit static library of the AMR-WB codec for Windows. Built using Microsoft Visual Studio 2022 and the following command line options: /permissive- /GS /GL /W3 /Gy /Zc:wchar_t /Zi /Gm- /O2 /sdl /Zc:inline /fp:fast /D "NDEBUG" /D "_LIB" /D "_UNICODE" /D "UNICODE" /errorReport:prompt /WX- /Zc:forScope /Gd /Oi /MT /FC /Fa"x64\Release\" /EHsc /nologo /Fo"x64\Release\" /Fp"x64\Release\cp_amrwb.pch" /diagnostics:column
cp_amrwb.h	Header file with API prototypes to cp_amrwb.lib and constant definitions used by decoder.c and encoder.c.
decoder.c	Sample application that demonstrates how to use the cp_amrwb.lib decoder API.
encoder.c	Sample application that demonstrates how to use the cp_amrwb.lib encoder API.



## AMR-WB API functions

### E\_IF\_amrwb\_init

Description	Initializes resources needed for the encoder. To “reset” the encoder during normal operation, call <code>E_IF_amrwb_exit</code> and then call this function again.
Syntax	<pre>#include "typedef.h" #include "cp_amrwb.h"  Word32 E_IF_amrwb_init(     void);</pre>
Arguments	None
Return value	-1 if failed, 0 otherwise.

### E\_IF\_amrwb\_encode

Description	Encodes one frame of 16-bit linear PCM speech data.								
Syntax	<pre>#include "typedef.h" #include "cp_amrwb.h "  Word32 E_IF_amrwb_encode(     Word16 iMode,     Word16 *pSpeech,     UWord8 *pBitstream,     Word16 *iDtx );</pre>								
Arguments	<table><tr><td><code>iMode</code></td><td>(Input) Encoding mode {0..8}.</td></tr><tr><td><code>pSpeech</code></td><td>(Input) Buffer containing one frame of speech samples.</td></tr><tr><td><code>pBitstream</code></td><td>(Output) Buffer containing one frame of encoded bitstream data.</td></tr><tr><td><code>iDtx</code></td><td>(Input) Set to 1 to enable voice activity detection (VAD), 0 otherwise. For a description of codec operation when VAD is enabled, please see “Discontinuous Transmission (DTX) mode” on page 7.</td></tr></table>	<code>iMode</code>	(Input) Encoding mode {0..8}.	<code>pSpeech</code>	(Input) Buffer containing one frame of speech samples.	<code>pBitstream</code>	(Output) Buffer containing one frame of encoded bitstream data.	<code>iDtx</code>	(Input) Set to 1 to enable voice activity detection (VAD), 0 otherwise. For a description of codec operation when VAD is enabled, please see “Discontinuous Transmission (DTX) mode” on page 7.
<code>iMode</code>	(Input) Encoding mode {0..8}.								
<code>pSpeech</code>	(Input) Buffer containing one frame of speech samples.								
<code>pBitstream</code>	(Output) Buffer containing one frame of encoded bitstream data.								
<code>iDtx</code>	(Input) Set to 1 to enable voice activity detection (VAD), 0 otherwise. For a description of codec operation when VAD is enabled, please see “Discontinuous Transmission (DTX) mode” on page 7.								
Return value	Frame size in bytes.								

**E\_IF\_amrwb\_exit**

Description Frees resources needed for the encoder.

**Syntax**

```
#include "typedef.h"
#include "cp_amrwb.h"

Word32 E_IF_amrwb_exit(
    void);
```

Arguments none

Return value none

**D\_IF\_amrwb\_init**

## Description

Initializes the resources needed for the decoder. To “reset” the decoder during normal operation, call `D_IF_amrwb_exit` and then call this function again.

**Syntax**

```
#include "typedef.h"
#include "cp_amrwb.h"

Word32 D_IF_amrwb_init(
    void
);
```

## Arguments

none

## Return value

-1 if failed, 0 otherwise.

**D\_IF\_amrwb\_decode**

## Description

Decodes one frame of encoded bitstream data.

**Syntax**

```
#include "typedef.h"
#include "cp_amrwb.h"

void D_IF_amrwb_decode(
    UWord8 *pBitstream,
    Word16 *pSynthSpeech,
    Word32 iBadFrame
);
```

## Arguments

<code>pBitstream</code>	(Input) Buffer containing one frame of encoded bitstream data.
<code>pSynthSpeech</code>	(Output) Buffer containing one frame of speech samples.
<code>iBadFrame</code>	<p>(Input) Bad frame indicator is used to inform the decoder of problems in the received frame. ‘_good_frame’ should be used under normal conditions (see <code>dec_if.h</code>).</p> <p>_good_frame: good speech or SID frame is received.</p> <p>_bad_frame: frame with possible bit errors.</p> <p>_lost_frame: speech or SID frame is lost in transmission.</p> <p>_no_frame: indicates non-received frames in DTX operation.</p>

## Return value

None

**D\_IF\_amrwb\_exit**

Description Frees resources needed for the decoder.

**Syntax**

```
#include "typedef.h"
#include "cp_amrwb.h"

Word32 D_IF_amrwb_exit(
    void);
```

Arguments none

Return value none