FAAC - ISO/MPEG 2/4 AAC Encoder Library V1.0

Freeware Advanced Audio Coding
(http://www.audiocoding.com/)

1 Contents

2 Scope

This document describes the interface and usage of the

FAAC - ISO / MPEG 2 / 4 AAC Encoder Library

Developed for the Freeware Advanced Audio Coding project.

3 Interface description

The ISO/MPEG 2/4 AAC Encoder Library provides a high-level interface for encoding MPEG2 and MPEG4 ISO AAC files. The following header file is provided for usage in C/C++ programs:

faac.h: function prototypes

The encoder core resides in a statically linkable library called libfaac.lib (*Microsoft Windows*) or libfaac.a (*UNIX*). There are various example programs that show how to use the library.

4 Usage

4.1 Calling sequence

For encoding AAC bitstreams the following calling sequence is mandatory:

- Call **`faacEncOpen()`** for every encoder instance you need.
- To set encoder options, call **`faacEncGetCurrentConfiguration()`**, change the parameters in the structure accessible by the returned pointer and then call **`faacEncSetConfiguration()`**.
- As long as there are still samples left to encode, call **`faacEncEncode()`** to encode the data. The encoder returns the bitstream data in a client-supplied buffer.
- Once you call **`faacEncEncode()`** with zero samples of input the flushing process is initiated; afterwards you may call **`faacEncEncode()`** with zero samples input only.

  faacEncEncode() will continue to write out data until all audio samples have been encoded.
- Once **`faacEncEncode()`** has returned with zero bytes written, call **`faacEncClose()`** to destroy this encoder instance.

5 Function reference

5.1 Initialization / De-initialization

5.1.1 faacEncOpen()

Prototype

```
faacEncHandle FAACAPI faacEncOpen
    (
    unsigned long sampleRate,
    unsigned int numChannels,
    unsigned long *inputSamples,
    unsigned long *maxOutputBytes
    );
```

Description

Open and initialize one instance of the encoder.

Parameters

- sampleRate
  The samplerate of the encoder input data.
- numChannels
  The number of channels of the encoder input data.
- inputSamples
  Receives the total number of samples that should be fed to **`faacEncEncode()`** in each call.
- maxOutputBytes
  Receives the maximum number of bytes that can be in the output buffer after a call to **`faacEncEncode()`**.

Return value

An initialized encoder handle. If anything goes wrong NULL is returned.

5.1.2 faacEncClose()

Prototype

```
void FAACAPI faacEncClose
    (
    faacEncHandle hEncoder
    );
```

Description

Closes an encoder instance.

Parameters

- hEncoder
  An encoder handle returned by **`faacEncOpen()`**.

5.2 Encoder configuration

5.2.1 faacEncGetCurrentConfiguration()

Prototype

```
faacEncConfigurationPtr FAACAPI
faacEncGetCurrentConfiguration
    (
    faacEncHandle hEncoder
    );
```

Description

Get a pointer to a structure describing the current encoder configuration. You may change this structure and feed it into **`faacEncSetConfiguration()`**.

5.2.2 faacEncSetConfiguration()

Prototype

```
int FAACAPI faacEncSetConfiguration
    (
    faacDecHandle hDecoder,
    faacEncConfigurationPtr config
    );
```

Description

Set a new encoder configuration. See **`faacEncGetCurrentConfiguration()`**.

5.3 Encoding functions

5.3.1 faacEncEncode()

Prototype

```
int FAACAPI faacEncEncode
    (
    faacEncHandle hEncoder,
    short *inputBuffer,
    unsigned int samplesInput,
    unsigned char *outputBuffer,
    unsigned int bufferSize
    );
```

Description

Encode one frame of samples.

Parameters

- hEncoder
  An encoder handle.
- inputBuffer
  Contains audio samples to be encoded.
- samplesInput

The number of valid samples in inputBuffer, this should be the number received in **`inputSamples`** in the call to **`faacEncOpen()`**, as long as that number of samples is available. Once you have called **`faacEncEncode()`** with zero samples input, the flushing process is initiated.

- outputBuffer
  Pointer to a buffer receiving the bitstream data. This buffer should at least be of size **`maxOutputBytes`** received in the call to **`faacEncOpen()`**.

Return value

A negative value to indicate a failure, the number of vaid bytes in the output buffer otherwise. A return value of zero does not indicate failure.

6 Data structures reference

6.1 faacEncConfiguration

Definition

```
typedef struct faacEncConfiguration
    {
    unsigned int mpegVersion;
    unsigned int aacObjectType;
    unsigned int allowMidside;
    unsigned int useLfe;
    unsigned int useTns;
    unsigned long bitRate;
    unsigned int bandWidth;
    }
faacEncConfiguration, *faacEncConfigurationPtr;
```

Description

Through this structure you can change the encoder configuration.

Fields

- mpegVersion
  The MPEG version. Can be either **`MPEG2`** or **`MPEG4`**.
- aacObjectType
  The AAC object type. Can be one of these values: **`MAIN`**, **`LOW`** or **`LTP`**.
- allowMidside
  Set to 1 to allow the usage of mid/side coding, 0 for no mid/side coding.
- useLfe
  Set to 1 to use one LFE channel. *This flag is not supported yet*.
- useTns
  Set to 1 to use TNS, 0 for no TNS.
- bitRate
  Holds the bitrate in bits per second per channel.
- bandwidth
  Holds the maximum bandwith in Hz.