Chapter 1

libtheora Main Page

1.1 Introduction

This is the documentation for the libtheora C API. libtheora is the reference implementation for Theora, a free video codec. Theora is derived from On2's VP3 codec with improved integration for Ogg multimedia formats by Xiph.Org.
Chapter 2

libtheora Data Structure Index

2.1 libtheora Data Structures

Here are the data structures with brief descriptions:

- **yuv_buffer** .......................................................... 7
Chapter 3

libtheora File Index

3.1 libtheora File List

Here is a list of all documented files with brief descriptions:

theora.h

.................................

.................................... 9
Chapter 4

libtheora Data Structure Documentation

4.1 yuv_buffer Struct Reference

#include <theora.h>

Data Fields

- int y_width
- int y_height
- int y_stride
- int uv_width
- int uv_height
- int uv_stride
- unsigned char * y
- unsigned char * u
- unsigned char * v

4.1.1 Detailed Description

A YUV buffer.

The documentation for this struct was generated from the following file:

- theora.h
Chapter 5

libtheora File Documentation

5.1 theora.h File Reference

#include <ogg/ogg.h>

Data Structures

- struct yuv_buffer
- struct theora_info
- struct theora_state
- struct theora_comment

Defines

- #define OC_FAULT -1
- #define OC_EINVAL -10
- #define OC_DISABLED -11
- #define OC_BADHEADER -20
- #define OC_NOTFORMAT -21
- #define OC_VERSION -22
- #define OC_IMPL -23
- #define OC_BADPACKET -24
- #define OC_NEWPACKET -25

Typedefs

- typedef theora_comment theora_comment

Enumerations

- enum theora_colorspace { OC_CS_UNSPECIFIED, OC_CS_ITU_REC_470M, OC_CS_ITU_REC_470BG }
Functions

- `const char * theora_version_string(void)`
- `ogg_uint32_t theora_version_number(void)`
- `int theora_encode_init( theora_state *th, theora_info *c)`
- `int theora_encode_YUVin( theora_state *t, yuv_buffer *yuv)`
- `int theora_encode_packetout( theora_state *t, int last_p, ogg_packet *op)`
- `int theora_encode_header( theora_state *t, ogg_packet *op)`
- `int theora_encode_comment( theora_comment *tc, ogg_packet *op)`
- `int theora_encode_tables( theora_state *t, ogg_packet *op)`
- `int theora_decode_header( theora_info *ci, theora_comment *cc, ogg_packet *op)`
- `int theora_decode_init( theora_state *th, theora_info *c)`
- `int theora_decode_packetin( theora_state *th, ogg_packet *op)`
- `int theora_decode_YUVout( theora_state *th, yuv_buffer *yuv)`
- `double theora_granule_time( theora_state *th, ogg_int64_t granulepos)`
- `ogg_int64_t theora_granule_frame( theora_state *th, ogg_int64_t granulepos)`
- `void theora_info_init( theora_info *c)`
- `void theora_info_clear( theora_info *c)`
- `void theora_clear( theora_state *t)`
- `void theora_comment_init( theora_comment *tc)`
- `void theora_comment_add( theora_comment *tc, char *comment)`
- `void theora_comment_add_tag( theora_comment *tc, char *tag, char *value)`
- `char * theora_comment_query( theora_comment *tc, char *tag, int count)`
- `int theora_comment_query_count( theora_comment *tc, char *tag)`
- `void theora_comment_clear( theora_comment *tc)`

5.1.1 Detailed Description

The libtheora C API.

5.1.2 Enumeration Type Documentation

5.1.2.1 `enum theora_colorspace`

A Colorspace.

5.1.3 Function Documentation

5.1.3.1 `void theora_clear( theora_state * t)`

Free all internal data associated with a theora_state handle.

Parameters:

- `t` A theora_state handle.
5.1.3.2 int theora_decode_header (theora_info * ci, theora_comment * cc, 
ogg_packet * op)

Decode an Ogg packet, with the expectation that the packet contains an initial header, comment 
data or codebook tables.

Parameters:

- ci A theora_info structure to fill. This must have been previously initialized with 
  `theora_info_init()` (p. 15). If op contains an initial header, `theora_decode_header()` (p. 11) 
  will fill ci with the parsed header values. If op contains codebook tables, `theora_decode_ 
  header()` (p. 11) will parse these and attach an internal representation to ci->
  > codec_setup.
- cc A theora_comment structure to fill. If op contains comment data, `theora_decode_ 
  header()` (p. 11) will fill cc with the parsed comments.
- op An ogg_packet structure which you expect contains an initial header, comment data or 
  codebook tables.

Return values:

- **OC_BADHEADER** op is NULL; OR the first byte of op->packet has the signature of an 
  initial packet, but op is not a b_o_s packet; OR this packet has the signature of an 
  initial header packet, but an initial header packet has already been seen; OR this packet 
  has the signature of a comment packet, but the initial header has not yet been seen; OR 
  this packet has the signature of a comment packet, but contains invalid data; OR this 
  packet has the signature of codebook tables, but the initial header or comments have not 
  yet been seen; OR this packet has the signature of codebook tables, but contains invalid 
  data; OR the stream being decoded has a compatible version but this packet does not 
  have the signature of a theora initial header, comments, or codebook packet
- **OC_VERSION** The packet data of op is an initial header with a version which is incom-
  patible with this version of libtheora.
- **OC_NEWPACKET** the stream being decoded has an incompatible (future) version and 
  contains an unknown signature.
- 0 Success

Note:

The normal usage is that `theora_decode_header()` (p. 11) be called on the first three 
packets of a theora logical bitstream in succession.

5.1.3.3 int theora_decode_init (theora_state * th, theora_info * c)

Initialize a theora_state handle for decoding.

Parameters:

- th The theora_state handle to initialize.
- c A theora_info struct filled with the desired decoding parameters. This is of course usually 
  obtained from a previous call to `theora_decode_header()` (p. 11).

Returns:

- 0 Success
5.1.3.4  int theora_decode_packetin (theora_state * th, ogg_packet * op)

Input a packet containing encoded data into the theora decoder.

Parameters:
   th  A theora_state handle previously initialized for decoding.
   op  An ogg_packet containing encoded theora data.

Return values:
   OC_BADPACKET  op does not contain encoded video data

5.1.3.5  int theora_decode_YUVout (theora_state * th, yuv_buffer * yuv)

Output the next available frame of decoded YUV data.

Parameters:
   th  A theora_state handle previously initialized for decoding.
   yuv A yuv_buffer (p. 7) in which libtheora should place the decoded data.

Return values:
   0  Success

5.1.3.6  int theora_encode_comment (theora_comment * tc, ogg_packet * op)

Request a comment header packet from provided metadata. A pointer to the comment data is
placed in a user-provided ogg_packet structure.

Parameters:
   tc  A theora_comment structure filled with the desired metadata
   op  An ogg_packet structure to fill. libtheora will set all elements of this structure, including
        a pointer to the encoded comment data. The memory for the comment data is owned
        by libtheora.

Return values:
   0  Success

5.1.3.7  int theora_encode_header (theora_state * t, ogg_packet * op)

Request a packet containing the initial header. A pointer to the header data is placed in a user-
provided ogg_packet structure.

Parameters:
   t   A theora_state handle previously initialized for encoding.
   op  An ogg_packet structure to fill. libtheora will set all elements of this structure, including
        a pointer to the header data. The memory for the header data is owned by libtheora.

Return values:
   0  Success
5.1.3.8  int theora_encode_init (theora_state * th, theora_info * c)

Initialize the theora encoder.

Parameters:
- th The theora_state handle to initialize for encoding.
- ti A theora_info struct filled with the desired encoding parameters.

Returns:
- 0 Success

5.1.3.9  int theora_encode_packetout (theora_state * t, int last_p, ogg_packet * op)

Request the next packet of encoded video. The encoded data is placed in a user-provided ogg_packet structure.

Parameters:
- t A theora_state handle previously initialized for encoding.
- last_p ???
- op An ogg_packet structure to fill. libtheora will set all elements of this structure, including a pointer to encoded data. The memory for the encoded data is owned by libtheora.

Return values:
- 0 No internal storage exists OR no packet is ready
- -1 The encoding process has completed
- 1 Success

5.1.3.10  int theora_encode_tables (theora_state * t, ogg_packet * op)

Request a packet containing the codebook tables for the stream. A pointer to the codebook data is placed in a user-provided ogg_packet structure.

Parameters:
- t A theora_state handle previously initialized for encoding.
- op An ogg_packet structure to fill. libtheora will set all elements of this structure, including a pointer to the codebook data. The memory for the header data is owned by libtheora.

Return values:
- 0 Success

5.1.3.11  int theora_encode_YUVin (theora_state * t, yuv_buffer * yuv)

Input a YUV buffer into the theora encoder.

Parameters:
- t A theora_state handle previously initialized for encoding.
**yuv** A buffer of YUV data to encode.

**Return values:**
- **OC_EINVAL** Encoder is not ready, or is finished.
- -1 The size of the given frame differs from those previously input
- 0 Success

### 5.1.3.12 `ogg_int64_t theora_granule_frame (theora_state * th, ogg_int64_t granulepos)`

Convert a granulepos to an absolute frame number. The granulepos is interpreted in the context of a given theora_state handle.

**Parameters:**
- **th** A previously initialized theora_state handle (encode or decode)
- **granulepos** The granulepos to convert.

**Returns:**
- The frame number corresponding to `granulepos`.

**Return values:**
- -1 The given granulepos is invalid (ie. negative)

### 5.1.3.13 `double theora_granule_time (theora_state * th, ogg_int64_t granulepos)`

Convert a granulepos to absolute time in seconds. The granulepos is interpreted in the context of a given theora_state handle.

**Parameters:**
- **th** A previously initialized theora_state handle (encode or decode)
- **granulepos** The granulepos to convert.

**Returns:**
- The absolute time in seconds corresponding to `granulepos`.

**Return values:**
- -1 The given granulepos is invalid (ie. negative)

### 5.1.3.14 `void theora_info_clear (theora_info * c)`

Clear a theora_info structure. All values within the given theora_info structure are cleared, and associated internal codec setup data is freed.

**Parameters:**
- **c** A theora_info struct to initialize.
5.1 theora.h File Reference

5.1.3.15  

```c
void theora_info_init (theora_info * c)
```

Initialize a theora_info structure. All values within the given theora_info structure are initialized, and space is allocated within libtheora for internal codec setup data.

**Parameters:**
- `c` A theora_info struct to initialize.

5.1.3.16  

```c
ogg_uint32_t theora_version_number (void)
```

Retrieve a 32-bit version number. This number is composed of a 16-bit major version, 8-bit minor version and 8 bit sub-version, composed as follows:

```
(VERSION_MAJOR<<16) + (VERSION_MINOR<<8) + (VERSION_SUB)
```

**Returns:**
- the version number.

5.1.3.17  

```c
const char * theora_version_string (void)
```

Retrieve a human-readable string to identify the vendor and version.

**Returns:**
- a version string.
Index

theora.h, 9
  theora_clear, 10
theora_colorspace, 10
theora_decode_header, 10
theora_decode_init, 11
theora_decode_packetin, 11
theora_decode_YUVout, 12
theora_encode_comment, 12
theora_encode_header, 12
theora_encode_init, 12
theora_encode_packetout, 13
theora_encode_tables, 13
theora_encode_YUVin, 13
theora_granule_frame, 14
theora_granule_time, 14
theora_info_clear, 14
theora_info_init, 14
theora_version_number, 15
theora_version_string, 15
theora_clear
  theora.h, 10
theora_colorspace
  theora.h, 10
theora_decode_header
  theora.h, 10
theora_decode_init
  theora.h, 11
theora_decode_packetin
  theora.h, 11
theora_decode_YUVout
  theora.h, 12
theora_encode_comment
  theora.h, 12
theora_encode_header
  theora.h, 12
theora_encode_init
  theora.h, 12
theora_encode_packetout
  theora.h, 13
theora_encode_tables
  theora.h, 13
theora_encode_YUVin
  theora.h, 13
theora_granule_frame
  theora.h, 14
theora_granule_time
  theora.h, 14
theora_info_clear
  theora.h, 14
theora_info_init
  theora.h, 14
theora_version_number
  theora.h, 15
theora_version_string
  theora.h, 15
yuv_buffer, 7