原 FFMPEG结构体分析：AVStream

2013年11月10日 00:02:04 　阅读数：34879

注：写了一系列的结构体的分析的文章，在这里列一个列表：

[FFMPEG结构体分析：AVFrame](#)

[FFMPEG结构体分析：AVFormatContext](#)

[FFMPEG结构体分析：AVCodecContext](#)

[FFMPEG结构体分析：AVIOContext](#)

[FFMPEG结构体分析：AVCodec](#)

[FFMPEG结构体分析：AVStream](#)

[FFMPEG结构体分析：AVPacket](#)

FFMPEG有几个最重要的结构体，包含了解协议，解封装，解码操作，此前已经进行过分析：

[FFMPEG中最关键的结构体之间的关系](#)

在此不再详述，其中AVStream是存储每一个视频/音频流信息的结构体。本文将会分析一下该结构体里重要变量的含义和作用。

首先看一下结构体的定义（位于avformat.h文件中）：

```cpp
/* 雷霄骅
 * 中国传媒大学/数字电视技术
 * leixiaohua1020@126.com
 *
 */
/**
 * Stream structure.
 * New fields can be added to the end with minor version bumps.
 * Removal, reordering and changes to existing fields require a major
 * version bump.
 * sizeof(AVStream) must not be used outside libav*.
 */
typedef struct AVStream {
    int index;    /**< stream index in AVFormatContext */
    /**
     * Format-specific stream ID.
     * decoding: set by libavformat
     * encoding: set by the user
     */
    int id;
    AVCodecContext *codec; /**< codec context */
    /**
     * Real base framerate of the stream.
     * This is the lowest framerate with which all timestamps can be
     * represented accurately (it is the least common multiple of all
     * framerates in the stream). Note, this value is just a guess!
     * For example, if the time base is 1/90000 and all frames have either
     * approximately 3600 or 1800 timer ticks, then r_frame_rate will be 50/1.
     */
    AVRational r_frame_rate;
    void *priv_data;

    /**
     * encoding: pts generation when outputting stream
     */
    struct AVFrac pts;

    /**
     * This is the fundamental unit of time (in seconds) in terms
     * of which frame timestamps are represented. For fixed-fps content,
     * time base should be 1/framerate and timestamp increments should be 1.
     * decoding: set by libavformat
     * encoding: set by libavformat in av_write_header
     */
    AVRational time_base;

    /**
     * Decoding: pts of the first frame of the stream in presentation order, in stream time base.
     * Only set this if you are absolutely 100% sure that the value you set
     * it to really is the pts of the first frame.
     * This may be undefined (AV_NOPTS_VALUE).
     * @note The ASF header does NOT contain a correct start_time the ASF
     * demuxer must NOT set this.
     */
    int64_t start_time;
```

```c
57.        /**
58.         * Decoding: duration of the stream, in stream time base.
59.         * If a source file does not specify a duration, but does specify
60.         * a bitrate, this value will be estimated from bitrate and file size.
61.         */
62.        int64_t duration;
63.
64.        int64_t nb_frames;                 ///< number of frames in this stream if known or 0
65.
66.        int disposition; /**< AV_DISPOSITION_* bit field */
67.
68.        enum AVDiscard discard; ///< Selects which packets can be discarded at will and do not need to be demuxed.
69.
70.        /**
71.         * sample aspect ratio (0 if unknown)
72.         * - encoding: Set by user.
73.         * - decoding: Set by libavformat.
74.         */
75.        AVRational sample_aspect_ratio;
76.
77.        AVDictionary *metadata;
78.
79.        /**
80.         * Average framerate
81.         */
82.        AVRational avg_frame_rate;
83.
84.        /**
85.         * For streams with AV_DISPOSITION_ATTACHED_PIC disposition, this packet
86.         * will contain the attached picture.
87.         *
88.         * decoding: set by libavformat, must not be modified by the caller.
89.         * encoding: unused
90.         */
91.        AVPacket attached_pic;
92.
93.        /*****************************************************************
94.         * All fields below this line are not part of the public API. They
95.         * may not be used outside of libavformat and can be changed and
96.         * removed at will.
97.         * New public fields should be added right above.
98.         *****************************************************************
99.         */
100.
101.        /**
102.         * Stream information used internally by av_find_stream_info()
103.         */
104. #define MAX_STD_TIMEBASES (60*12+5)
105.        struct {
106.            int64_t last_dts;
107.            int64_t duration_gcd;
108.            int duration_count;
109.            double duration_error[2][2][MAX_STD_TIMEBASES];
110.            int64_t codec_info_duration;
111.            int nb_decoded_frames;
112.            int found_decoder;
113.        } *info;
114.
115.        int pts_wrap_bits; /**< number of bits in pts (used for wrapping control) */
116.
117.        // Timestamp generation support:
118.        /**
119.         * Timestamp corresponding to the last dts sync point.
120.         *
121.         * Initialized when AVCodecParserContext.dts_sync_point >= 0 and
122.         * a DTS is received from the underlying container. Otherwise set to
123.         * AV_NOPTS_VALUE by default.
124.         */
125.        int64_t reference_dts;
126.        int64_t first_dts;
127.        int64_t cur_dts;
128.        int64_t last_IP_pts;
129.        int last_IP_duration;
130.
131.        /**
132.         * Number of packets to buffer for codec probing
133.         */
134. #define MAX_PROBE_PACKETS 2500
135.        int probe_packets;
136.
137.        /**
138.         * Number of frames that have been demuxed during av_find_stream_info()
139.         */
140.        int codec_info_nb_frames;
141.
142.        /**
143.         * Stream Identifier
144.         * This is the MPEG-TS stream identifier +1
145.         * 0 means unknown
146.         */
147.        int stream_identifier;
```

```
148.
149.        int64_t interleaver_chunk_size;
150.        int64_t interleaver_chunk_duration;
151.
152.        /* av_read_frame() support */
153.        enum AVStreamParseType need_parsing;
154.        struct AVCodecParserContext *parser;
155.
156.        /**
157.         * last packet in packet_buffer for this stream when muxing.
158.         */
159.        struct AVPacketList *last_in_packet_buffer;
160.        AVProbeData probe_data;
161. #define MAX_REORDER_DELAY 16
162.        int64_t pts_buffer[MAX_REORDER_DELAY+1];
163.
164.        AVIndexEntry *index_entries; /**< Only used if the format does not
165.                                        support seeking natively. */
166.        int nb_index_entries;
167.        unsigned int index_entries_allocated_size;
168.
169.        /**
170.         * flag to indicate that probing is requested
171.         * NOT PART OF PUBLIC API
172.         */
173.        int request_probe;
174. } AVStream;
```

AVStream重要的变量如下所示：

int index：标识该视频/音频流

AVCodecContext *codec：指向该视频/音频流的AVCodecContext（它们是一一对应的关系）

AVRational   time_base：时基。通过该值可以把PTS，DTS转化为真正的时间。FFMPEG其他结构体中也有这个字段，但是根据我的经验，只有AVStream中的 time_base是可用的 。PTS*time_base=真正的时间

int64_t duration：该视频/音频流长度

AVDictionary *metadata：元数据信息

AVRational avg_frame_rate：帧率（注：对视频来说，这个挺重要的）

AVPacket attached_pic：附带的图片。比如说一些MP3，AAC音频文件附带的专辑封面。

该结构体其他字段的作用目前还有待于探索。