

中科院声学所离线识别引擎

FFMPEG接口设计

版 本 历 史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本/状态 | 作者 | 参与者 | 起止日期 | 备注 |
| V1.0 | 王超锋 |  | 2021-11-17 | 创建文件 |
|  |  |  |  |  |

目录

[1.概述 3](#_Toc22086)

[1.1 编写目的 3](#_Toc18325)

[2.总体设计 4](#_Toc28276)

[2.1 总体需求 4](#_Toc24096)

[3.ffmpeg接口设计 5](#_Toc9269)

[3.1 通用 API 5](#_Toc3292)

[3.2 解码API 7](#_Toc29606)

[3.3 编码API 8](#_Toc10425)

[4.代码目录 10](#_Toc10776)

[4.1 代码目录结构 10](#_Toc28003)

[4.2 主要代码模块说明 11](#_Toc4544)

[4.2.1 libavutil 11](#_Toc19618)

[4.2.2 libavformat 11](#_Toc3070)

[4.2.3 libavcodec 12](#_Toc1549)

# 1.概述

1

**1.1 编写目的**

本文档主要描述ffmpeg内部接口设计及接口中参数的数据结构，本文档的读者对象为离线识别系统的开发人员、测试人员、系统维护人员及接入识别系统的第三方业务人员，通过本文档能够从总体上了解识别系统内部的数据流向及处理方式。

本说明给出FFMPEG的接口设计说明，包括接口安全、版本兼容性、数据格式、服务端的异常处理等。

目的在于：

* 为开发人员提供依据；
* 为代码修改、维护提供条件；

# 2.总体设计

2

## 2.1 总体需求

接口需求：

1，接口参数简单明了，参数有实际意义；

2，接口功能定义明确；

3，接口的数据特性明确；

4，接口的规格和技术要求明确；

5，接口程序的数据处理逻辑清晰；

# 3.ffmpeg接口设计

3

ffmpeg包含通用API、解码API、编码API。

## 3.1 通用 API

1.av\_register\_all：

编译配置（–enable、–disable） FFmpeg 的时候，会生成两个文件：config.mk、config.h。config.mk 会实际上就是 makefile 文件需要包含进去的子模块，会作用在编译阶段，帮助开发者编译出正确的库；而 config.h 是作用在运行阶段，这一阶段将确保需要注册哪些容器以及编解码格式到 FFmpeg 框架中。所以该函数的内部实现会先调用 avcodec\_register\_all 来注册所有 config.h 里面开放的编解码器，然后会注册所有的 Muxer 和 Demuxer，最后注册所有的 Protocol。如此，在 config 的过程中，enable、disable 的选项就作用到了运行时，该函数的源码分析涉及的源文件包括 url.c、alformats.c、mux.c、format.c 等文件。

av\_register\_all()的作用是：初始化所有组件，只有调用了该函数，才能使用复用器和编解码器，该接口内部的调用为：

(1).avcodec\_register\_all()，该接口内部执行步骤：

- 注册硬件加速器：REGISTER\_HWACCEL()

- 注册音视频编码器：REGISTER\_ENCODER()

- 注册音视频解码器：REGISTER\_DECODER()

- 打包注册：REGISTER\_ENCDEC()

- 注册解析器：REGISTER\_PARSER()

(2).执行复用器和解复用器的注册：

- 注册复用器：REGISTER\_MUXER()

- 注册解复用器：REGISTER\_DEMUXER()

- 打包注册：REGISTER\_MUXDEMUX()

2.av\_find\_codec：

在 avcodec\_register\_all 函数里面已经把编码器和解码器都存放到了一个链表中，因此都是从该链表进行遍历查找

3.avcodec\_open2：

打开编解码其的时候就会用到该函数，参数有三个，第一个是 AVCodecContext，如果想要传入私有参数，比如 preset、tune、profile，则可以为设置到 priv\_data 参数中。具体到函数实现时，它会找到对应的实现文件，比如，如果打开的是 libx264 编码器，那么实际上的 Codec 为 libx264.c 中的 ff\_libx264\_encoder，Codec 的生命周期方法就会委托给该结构体对应的函数指针所指向的函数。open 对应的就是 init 函数指针所指向的函数，该函数会调用具体的编码库的 API，并以对应的 AVCodecContext 中的 priv\_data 来填充对应第三方库所需要的私有参数。

4.av\_codec\_close：

和 open 类似，找到对应的实现文件中的 close 函数指针所指向的函数，然后该函数会调用对应第三方库的 API 来关闭掉对应编码库。其实 FFmpeg 所做的事情就是透明化所有的编码库，用自己的封装来为开发者提供统一的接口，开发者只需要在打开编解码库时指定编解码器的 ID 即可，之后编码、解码、关闭资源都会找到对应的实现文件去做具体的事情。

## 3.2 解码API

1.avformat\_open\_input

该函数会根据提供的的文件路径判断文件的格式，继而决定使用哪一个 Demuxer。比如，如果是 flv 文件，那么 Demuxer 就会使用对应的 ff\_flv\_demuxer，之后关键的生命周期方法 read\_header、read\_packet、read\_seek、read\_close 都会使用 ff\_flv\_demuxer 中函数指针指定的函数。read\_header 函数会将 AVStream 结构体构造好。

2.avformat\_find\_stream\_info

这个函数非常重要，该方法的作用是将所有 Stream 的MetaData信息填充好，方法内部会先查找对应的解码器，并打开，紧接着利用 Demuxer 中的 read\_packet 函数读取一段数据进行解码，解码数据越多，分析出的流数据就会越准确，本地资源会比较快，网络资源则较慢。该函数提供了几个参数可以控制读取数据的长度，分别为：probe\_size、max\_analyze、fps\_probe\_size，这几个参数的值越小，读取速度越快，信息则相对不够准确。

3.av\_read\_frame

该方法读取出来的数据是 AVPacket，该函数的实现首先会委托到 Demuxer 的 read\_packet 方法，然后在该函数中把未处理完的压缩数据进行缓存处理。

4.avcodec\_decode

如果要解码 H264，会找到 ff\_h264\_decoder，其中最重要的三个声明周期方法为 init、decode、close。

## 3.3 编码API

1.avformat\_alloc\_output\_context

和avformat\_open\_input类似，该函数最终会找到对应的格式复制给 AVFormatContext中的oformat。

2.avio\_open2

该方法首先调用函数ffurl\_open，构造除URLContext，这个结构体包含了 URLProtocol，接着调用avio\_alloc\_context方法，分配除AVIOContext结构体，并将上一步构造出来的URLProtocol传递进来，然后复制给AVFormatContext的属性。

编码步骤其实是解码的一个逆过程，解码过程中的av\_find\_stream\_info对应到编码就是avformat\_new\_stream和av\_format\_write\_header，该步骤会将音频流或视频流的信息填充好，分配出AVStream结构体。read\_header则对应于av\_write\_header，再之后是av\_write\_frame、av\_write\_tailer。

# 4.代码目录

4

## 4.1 代码目录结构

ffmpeg-4.3

|--compat

|--configure 配置项

|--doc 说明文档

|--ffbuild

|--fftools

|--libavcodec 编解码库

|--libavdevice 输入输出设备库

|--libavfilter 滤镜库

|--libavformat 文件格式与协议库

|--libavresample

|--libavutil 核心工具库

|--libpostproc

|--libswresample 重采样、转换声道数

|--libswscale 像素格式转换

|--Makefile 编译规则与命令

|--presets

|--README.md 说明

|--tests

|--tools

|--VERSION

## 4.2 主要代码模块说明

### 4.2.1 libavutil

核心工具库，最基础的模块之一，其它模块经常依赖该库做一些基本的音视频处理操作，比如 av\_image\_fill\_arrays（填充原始图像数据到 AVFrame）、av\_image\_get\_buffer\_size（根据图像宽高、格式获取填充该图像需要的字节数）、av\_get\_pix\_fmt\_name（获取像素格式的名称） 等等。

### 4.2.2 libavformat

文件格式和协议库，最重要的模块之一，封装了 Protocol 层和 Demuxer、Muxer 层。常用于读写文件及文件信息，比如 avformat\_write\_header（写文件头）、av\_write\_trailer（写文件尾）、av\_read\_frame （从文件中读取一帧编码后的图像/音频数据）、av\_write\_frame（往文件中写一帧编码后的图像/音频数据）、av\_seek\_frame（给定一个时间戳，移动读指针到对应位置）等等。

### 4.2.3 libavcodec

编解码库，最重要的模块之一。FFmpeg 默认不会添加 libx264、FDK-AAC 等库，但 FFmpeg 可以像一个平台一样，将其它第三方的 Codec 以插件的形式添加进来，并为开发者提供统一的接口。编解码需要用到的函数基本都在该库中，比如 avcodec\_find\_decoder（找到对应的第三方解码器）、avcodec\_decode\_video2（使用对应的解码器解码一帧图像/音频数据）。