## 一、Ffmpeg基本使用

<命令分类.txt>

<http://blog.csdn.net/sand_wiliam/article/details/53380881> ------网页版

## 二、ffmpeg目录架构与基本框架机制

<http://blog.csdn.net/sand_wiliam/article/details/53381263>

## 三、编解码、滤镜基本流程

<http://blog.csdn.net/leixiaohua1020/article/details/39760711> ----------------ffmpeg.c基本框架

### 1.ffmpeg解码

<http://blog.csdn.net/leixiaohua1020/article/details/44220151> --------解码的简单分析和相关函数的简单的分析

以cuvid解码器为例：（由于涉及的源码较多，只是简单罗列了调用关系）

进入解码器cuvid\_decode\_init初始化的流程：

第一次，与试解有关

  ffmpeg\_parse\_options ----------------- at ffmpeg\_opt.c:

open\_files ---------------------at ffmpeg\_opt.c

open\_input\_file ------------------at ffmpeg\_opt.c

 avformat\_find\_stream\_info  ---------------------at libavformat/utils.c:3375 <http://blog.csdn.net/leixiaohua1020/article/details/44084321>（avformat\_find\_stream\_info  分析）

 avcodec\_open2 ---------------------------------- at libavcodec/utils.c

  ret = avctx->codec->init(avctx);     ------ -🡪cuvid\_decode\_init

第二次，从transcode 进入

transcode

transcode\_init

init\_input\_stream

avcodec\_open2

 ret = avctx->codec->init(avctx);     ------ ---》cuvid\_decode\_init

进入解码的的流程：

第一次,试解

ffmpeg\_parse\_options

open\_files

open\_input\_file

avformat\_find\_stream\_info

try\_decode\_frame  ------------------------ at libavformat/utils.c

avcodec\_send\_packet  ------------------------ at libavcodec/utils.c

do\_decode ---------循环被调用 ----------------------at libavcodec/utils.c:

avcodec\_decode\_video2  -------------------------at libavcodec/utils.c

  ret = avctx->codec->decode --------------> cuvid\_decode\_frame

真正的解码是在 transcode\_step中，

transcode\_step  -----------------------at ffmpeg.c 循环

process\_input  -------------------- at ffmpeg.c:

process\_input\_packet -----------------------  at ffmpeg.c

decode\_video  ----------------------------at ffmpeg.c

avcodec\_decode\_video2 --------------- at ffmpeg.c

### 2.ffmpeg编码

<http://blog.csdn.net/leixiaohua1020/article/details/44226355>----------编码流程记录以及函数的简单的分析

<http://blog.csdn.net/leixiaohua1020/article/details/44117891> ---------**avcodec\_open2()**

以nvenc为例，简单分析编码器

nvenc编码器的初始化简单调用流程

transcode (argc=21, argv=0x7fffffffea68)  --------------------at ffmpeg.c:

transcode\_init ()  --------------------------------at ffmpeg.c:

  init\_output\_stream () ----------------------at ffmpeg.c: /\* open each encoder \*/

 avcodec\_open2 () -----------------------  at libavcodec/utils.c:1572

  ret = avctx->codec->init(avctx);     -------------开始初始化编码器 --------->ff\_nvenc\_encode\_init

nvenc编码器的开始解码的调用流程

transcode

reap\_filters --------------- at ffmpeg.c:

do\_video\_out -----------------at ffmpeg.c:

avcodec\_encode\_video2 -----------------at libavcodec/utils.c:

     ret = avctx->codec->encode2(avctx, avpkt, frame, got\_packet\_ptr); ---------------->ff\_nvenc\_encode\_frame

### 3.ffmpeg滤镜

<http://blog.csdn.net/newchenxf/article/details/51364105>（过滤器的基本流程和机制简单分析）

<http://blog.csdn.net/lius1984/article/details/9067555>

进入过滤器的初始化的流程：

     ffmpeg\_parse\_options (argc=<value optimized out>, argv=<value optimized out>) at ffmpeg\_opt.c:

            init\_complex\_filters

                    init\_complex\_filtergraph (fg=0x1f024a0) at ffmpeg\_filter.c:

                         avfilter\_graph\_parse2   ------------------at libavfilter/graphparser.c:

                               parse\_filter     -------------------------at libavfilter/graphparser.c

                                  create\_filter    -------------------------at libavfilter/graphparser.c

                                     avfilter\_init\_str -------------------------at libavfilter/avfilter.c:1010

                                            int avfilter\_init\_dict ------ -------------at libavfilter/avfilter.c:1010

                                                    ret = ctx->filter->init(ctx);

开始进入滤镜工作的流程：

   transcode\_step

         process\_input

            process\_input\_packet

                  decode\_video    --------------------at ffmpeg.c:

                         av\_buffersrc\_add\_frame\_flags        ------------------------at libavfilter/buffersrc.c:174

                                  av\_buffersrc\_add\_frame\_internal

                                         request\_frame

                                              ff\_filter\_frame        -----------------------at libavfilter/avfilter.c:1228

                                                    ff\_filter\_frame\_framed

                                                           filter\_frame

                                                                if (!(filter\_frame = dst->filter\_frame))     ------------------》指向某个插件的接口

                                                                              ret = filter\_frame(link, out);

--------------------------------------------------------------------------------------------------------------------------------

  分析configure\_filtergraph接口的流程

               avfilter\_graph\_free  ----释放以前

                avfilter\_graph\_alloc -----分配结构体AVFilterGraph，设置初始参数

               avfilter\_graph\_parse2 --------分析命令，创建以输入命令相关的滤镜滤镜，初始化参数

               configure\_input\_filter  ------创建输入buffer滤镜

               configure\_output\_filter ------创建输出滤镜，format，buffersink，还会根据情况创建其他滤镜，如指定改变分辨率，创建scale

               avfilter\_graph\_config  -----配置各种滤镜

## 四、ffmpeg命令处理机制分析

### 1.参数是如何被设置？

<http://blog.csdn.net/leixiaohua1020/article/details/44279329>（结构体成员管理系统-AVOption）

<http://blog.csdn.net/leixiaohua1020/article/details/44268323>（结构体成员管理系统-AVClass）

<http://blog.csdn.net/sand_wiliam/article/details/53377721> （命令参数设置总结和分析）

### 2.命令如何被分组？

<http://blog.csdn.net/leixiaohua1020/article/details/39760711> （ffmpeg.c基本框架）

<http://blog.csdn.net/cugkuan/article/details/17632115> （[ffmpeg parse\_options函数解析](http://blog.csdn.net/cugkuan/article/details/17632115) ）

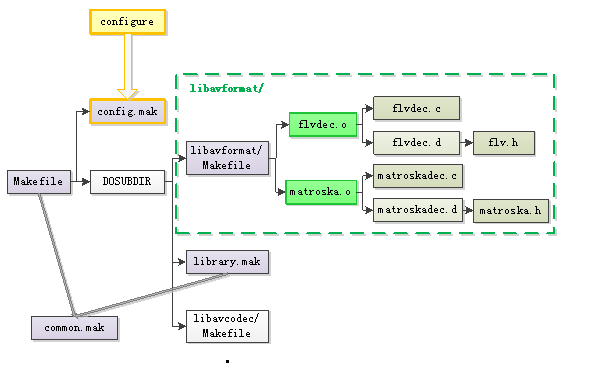
## 五、ffmpeg日志系统

<http://blog.csdn.net/leixiaohua1020/article/details/44243155-------------> 日志系统介绍

<http://blog.csdn.net/sand_wiliam/article/details/53389682> ------日志系统简单分析

## 六、makefile和configure

<http://blog.csdn.net/leixiaohua1020/article/details/44556525> ------------makefile分析



<http://blog.csdn.net/leixiaohua1020/article/details/44587465> --------------configure分析 -------本质是一个shell脚本

Configure一方面用于检测FFmpeg的编译环境，另一方面根据用户配置的选项生成config.mak，config.h文件（可能还有config.asm），提供给Makefile使用

#### 1.添加新模块：

<http://blog.csdn.net/sand_wiliam/article/details/53397027> （添加新模块的简单分析）

## 七、GPU-nvidia

<https://developer.nvidia.com/nvidia-video-codec-sdk#CodecSupportMatrix> （nvidia-sdk）

### 1.GPU编码

<http://blog.csdn.net/sand_wiliam/article/details/53392543> （nvenc简单分析）

## 八、代码demo

### 1.命令参数设置demo

<http://blog.csdn.net/sand_wiliam/article/details/53385569>

--- ----仿照ffmpeg的命令机制和日志系统以及基本框架，仿写一个小型的命令参数设置demo