5min

第 3 章 FFmpeg 三大常用工具及应用选项详解

FFmpeg 多媒体库支持的命令行调用分为 3 个模块：ffmpeg、ffplay、ffprobe，其中 ffmpeg 模块常用于音视频剪切、转码、滤镜、拼接、混音、截图等；ffplay 模块用于播放视频；ffprobe 模块用于检测多媒体流格式。详情可查阅 FFmpeg 的官方文档，网址为 https://ffmpeg.org/ffmpeg-all.html。

注意：本章的知识侧重于讲解命令行选项的使用，读者在学习过程中可能会感觉比较抽象。可以先对这些选项大体上有个印象，后续章节笔者准备了大量的案例来应用这些选项。

3.1 ffmpeg 工具简介

ffmpeg 工具主要用于编解码，主要工作流程相对比较简单，如图 3-1 所示。

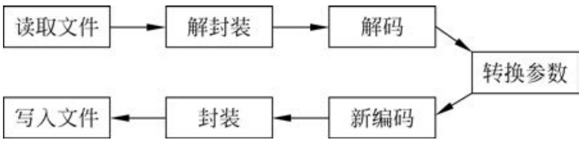

图 3-1 ffmpeg 工具的编解码流程

（1）读取输入源。

（2）进行音视频的解封装。

（3）解码每帧音视频数据。

（4）编码每帧音视频数据。

（5）进行音视频的重写封装。

（6）输出到目标。

ffmpeg 工具首先读取输入源，然后通过解封装器（Demuxer）将音视频包进行解封装，这个动作通过调用 libavformat 中的接口实现；接下来通过解码器（Decoder）进行解码，将音视频通过 Decoder 解包为 YUV 或者 PCM 这样的音视频裸数据，Decoder 通过 libavcodec 中的接口实现，然后通过编码器（Encoder）将对应的数据进行编码，编码可以通过

libavcodec 中的接口实现；接下来将编码后的音视频数据包通过封装器(Muxer)进行封装，Muxer 封装可以通过 libavformat 中的接口实现，最终的输出称为输出流。

ffmpeg 命令行工具是一个快速的音视频转换工具，使用方法及格式如下：

```
ffmpeg [全局选项] {[输入文件选项] -i'输入文件'} … {[输出文件选项]'输出文件'}
```

(1) [全局选项]：一对方括号[]代表这些选项是可选的。

(2) {[输入文件选项]-i'输入文件'}：一对花括号{}代表可以读取任意数量的“输入文件”，这里的文件可以是常规文件、管道、网络流、抓取设备等，输入文件前必须带有-i 选项，也可以带有其他的输入文件选项。

(3) {[输出文件选项]'输出文件'}：一对花括号{}代表可以写入任意数量的“输出文件”，可以带有相关的输出文件选项，任何不能被解释为选项的字符串都被认为是一个输出文件或者称为 URL。

ffmpeg 是多媒体领域应用最广泛的一个开源框架，包含了传输协议、视频格式、编解码、图像滤镜、语音处理等功能，并可以与 CUDA、OpenCV 等其他软件对接，几乎是事实上的音视频标准库。它的架构清晰简洁，大体上分为以下几部分：

(1) 传输协议，包括 TCP、UDP、HTTP、RTMP、RTSP 等常见的协议，本地文件也被实现为一个协议 file。

(2) 视频封装格式，包括 TS、FLV、MP4、MP3、MOV 等市面上常见的格式，几乎没有 ffmpeg 解析不了的。

(3) 编解码协议，包括视频的 H.264、H.265、VP8/9 等，音频的 MP3、AAC、Opus 等。大概支持几十种编解码协议，常见的不常见的都有，目前最流行的还是 H.264 和 AAC。

(4) 图像滤镜和语音滤镜，在 ffmpeg 里都通过 avfilter 实现，它是 ffmpeg 提供的对解码后的图像和语音进行处理的一个框架。

(5) 模块化设计，足以与 Nginx 和 Linux 内核相媲美，添加第三方模块或者自定义模块都非常方便。它自带的各个功能也是以模块的形式进行管理的，可以在编译之前灵活地配置。

3.2 ffplay 工具简介

在编译 FFmpeg 源代码时，如果系统中包含了 SDL 库，则会默认编译生成 ffplay 工具，如果不包含 SDL 时，无法生成 ffplay 工具，所以如果想使用 ffplay 进行流媒体播放测试，则需要安装 SDL 库。

通常使用 ffplay 作为播放器，其实 ffplay 不但可以作为播放器，同样可以作为很多图像化音视频数据的分析工具，通过 ffplay 可以看到视频图像的运动估计方向、音频数据的波形等，在本节将会对更多的参数进行介绍并举例。

3.2.1 ffplay 常用参数

ffplay 不仅是播放器，同时也是测试 ffmpeg 的 codec 引擎、format 引擎及 filter 引擎的工具，并且也可以作为可视化的媒体参数分析工具，可以通过 ffplay -help 命令查看这些参数。常见参数读者可以手动进行测试，举例如下。

如果希望从视频的第 30s 开始播放，播放 10s，命令如下：

```
ffplay -ss 30 -t 10 input.mp4
```

视频播放时播放器的窗口可以显示自定义标题，显示效果如图 3-2 所示，命令如下：

```
ffplay -window_title "Hello World,This is a sample" in.avi
```

图 3-2　ffplay 自定义窗口标题

使用 ffplay 还可以打开网络直播流，实时网络直播视频流的播放效果如图 3-3 所示，命令如下：

图 3-3　ffplay 播放网络直播流并自定义窗口标题

```
ffplay -window_title "播放测试" rtsp://127.0.0.1:8554/test1
```

注意：这里的网络直播流的网址为 rtsp://127.0.0.1:8554/test1，直播流是用 VLC 模拟出来的 RTSP 直播流，具体的操作步骤可参考本节的“3.2.4 VLC 作为 RTSP 流媒体服务器”。

3.2.2 ffplay 高级参数

通过 ffplay -help 命令可以看到的帮助信息比较多，在帮助信息中包含了许多高级参数，这里列举几个重要的参数选项，如表 3-1 所示。

表 3-1 ffplay 的参数选项及说明信息

参数	说明
x	强制设置视频显示窗口的宽度
y	强制设置视频显示窗口的高度
s	设置视频显示的宽和高
fs	强制全屏显示(Full Screen)
an	屏蔽音频
vn	屏蔽视频
sn	屏蔽字幕
ss	根据设置的时间(单位为秒)进行定位拖动
t	设置播放视频/音频长度
Bytes	设置定位拖动的策略，0 为不可拖动，1 为可拖动，-1 为自动
nodisp	关闭图形化显示窗口
f	强制使用设置的格式进行解析
window_title	设置显示窗口的标题
af	设置音频的滤镜
codec	强制使用设置的 codec 进行解码
autorotate	自动旋转视频
ast	设置将要播放的音频流
vst	设置将要播放的视频流
sst	设置将要播放的字幕流
stats	输出多媒体播放状态

参数	说明
fast	非标准化规范的多媒体兼容优化
sync	音视频同步设置，可根据音频视频进行参考，可根据视频时间参考，也可根据外部扩展时间进行参考
autoexit	多媒体播放完毕自动退出 ffplay，ffplay 默认播放完毕不退出播放器
exitonkeydown	当有按键按下事件产生时退出 ffplay
exitonmousedown	当有鼠标按键事件产生时退出 ffplay
loop	设置多媒体文件循环播放次数
framedrop	当 CPU 资源占用过高时，自动丢帧
infbuf	设置无极限的播放器 buffer，这个选项常见于实时流媒体播放场景
vf	视频滤镜设置
acodec	强制使用设置的音频解码器
vcodec	强制使用设置的视频解码器
scodec	强制使用设置的字幕解码器

下面根据这些参数与前面介绍过的一些参数进行组合，案例如下。

(1) 从 20s 开始播放一个视频，播放时长为 10s，播放完成后自动退出 ffplay，播放器的窗口标题为 Hello World，命令如下：

```
ffplay -window_title "Hello World" -ss 20 -t 10 -autoexit ande10.mp4
# -autoexit: 播放完毕后自动退出
```

(2) 强制使用 H.264 解码器解码 MPEG-4 格式的视频将会报错，test3.avi 的音视频流信息如图 3-4 所示，命令如下：

```
ffplay -vcodec h264 test3.avi
```

图 3-4　test3.avi 的视频流格式

注意： 这里的 test3.avi 中的视频流是用 MPEG-4 编码的，而不是 H.264。

该案例中会提示一些报错信息，因为强制使用 H.264 解码器，所以会提示找不到 NAL unit，代码如下：

```
//chapter3/3.1.txt
D:\_movies\__test>ffplay -vcodec h264 test3.avi
ffplay version 4.3.1 Copyright(c) 2003-2020 the FFmpeg developers
  built with gcc 10.2.1(GCC) 20200726
  configuration: --enable-gpl --enable-version3 --enable-sdl2 --enable-
fontconfig ......
  libavutil       56. 51.100 / 56. 51.100
  libavcodec      58. 91.100 / 58. 91.100
  libavformat     58. 45.100 / 58. 45.100
  libavdevice     58. 10.100 / 58. 10.100
  libavfilter      7. 85.100 /  7. 85.100
  libswscale       5.  7.100 /  5.  7.100
  libswresample    3.  7.100 /  3.  7.100
  libpostproc     55.  7.100 / 55.  7.100
Input #0,avi,from 'test3.avi':   0KB vq=   0KB sq=   0B f=0/0
  Metadata:
```

```
    encoder         : Lavf57.57.100
  Duration: 00: 00: 10.03,start: 0.000000,bitrate: 656 kb/s
    Stream #0: 0: Video: mpeg4(Simple Profile)(FMP4 / 0x34504D46),yuv420p,1280x720
[SAR 1: 1 DAR 16: 9],509 kb/s,30 fps,30 tbr,30 tbn,30 tbc
    Stream #0: 1: Audio: mp3(U[0][0][0] / 0x0055),44100 Hz,stereo,fltp,128 kb/s
    #注意: 这里提示找不到 H.264 码流规定的 NAL unit
[h264 @ 00de5740] Invalid NAL unit 0,skipping.
    Last message repeated 4 times
[h264 @ 00de5740] Invalid NAL unit 0,skipping.KB sq =  0B f = 0/0
```

（3）如果使用 ffplay 播放视频时希望加载字幕文件，则可以通过加载 ASS 或者 SRT 字幕文件来解决，下面举一个加载 SRT 字幕的例子，首先编辑 SRT 字幕文件，将这些内容存储为普通文本格式，文件名为 0.srt（在本书课件资料的 chapter3 目录下），并与要播放的视频文件放到同一个目录下，具体内容如下：

```
1
00: 00: 00,009 --> 00: 00: 03,490
这里是【梅老师】FFmpeg 的课程

2
00: 00: 05,619 --> 00: 00: 07,420
欢迎您进入 FFmpeg 的音视频世界

3
00: 00: 09,549 --> 00: 00: 12,170
祝大家万事如意,学有所成
```

然后通过 filter 将字幕文件加载到播放数据中，显示效果如图 3-5 所示，命令如下：

```
ffplay -window_title "Test Subtitle" -vf "subtitles = 0.srt" ande_302.mp4
```

图 3-5　ffplay 播放字幕文件

在该案例中，ffplay 完整的输出信息如下(注意#开头的内容是笔者添加的注释信息)：

```
//chapter3/3.1-ffplay-out.txt
D:\__test>ffplay -window_title "Test Subtitle" -vf "subtitles=0.srt" ande_302.mp4
#ffplay的编译选项,这里开启的第三方编码器比较多,读者可以顺便熟悉一下
ffplay version 4.3.1 Copyright(c) 2003-2020 the FFmpeg developers
  built with gcc 10.2.1(GCC) 20200726
  configuration: --enable-gpl --enable-version3 --enable-sdl2 --enable-
fontconfig --en
able-gnutls --enable-iconv --enable-libass --enable-libdav1d --enable-libbluray
--enab
le-libfreetype --enable-libmp3lame --enable-libopencore-amrnb --enable-
libopencore-amr
wb --enable-libopenjpeg --enable-libopus --enable-libshine --enable-libsnappy
--enable
-libsoxr --enable-libsrt --enable-libtheora --enable-libtwolame --enable-libvpx
--enab
le-libwavpack --enable-libwebp --enable-libx264 --enable-libx265 --enable-
libxml2 --en
able-libzimg --enable-lzma --enable-zlib --enable-gmp --enable-libvidstab --
enable-lib
vmaf --enable-libvorbis --enable-libvo-amrwbenc --enable-libmysofa --enable-
libspeex -
-enable-libxvid --enable-libaom --enable-libgsm --disable-w32threads --enable-
libmfx -
-enable-ffnvcodec --enable-CUDA-llvm --enable-cuvid --enable-d3d11va --enable
-nvenc --
enable-nvdec --enable-dxva2 --enable-avisynth --enable-libopenmpt --enable-amf
  libavutil      56. 51.100 / 56. 51.100
  libavcodec     58. 91.100 / 58. 91.100
  libavformat    58. 45.100 / 58. 45.100
  libavdevice    58. 10.100 / 58. 10.100
  libavfilter     7. 85.100 /  7. 85.100
  libswscale      5.  7.100 /  5.  7.100
  libswresample   3.  7.100 /  3.  7.100
  libpostproc    55.  7.100 / 55.  7.100
Input #0,mov,mp4,m4a,3gp,3g2,mj2,from 'ande_302.mp4': 0B f=0/0
  Metadata:
    major_brand     : isom
    minor_version   : 512
    compatible_brands: isomiso2avc1mp41
    encoder         : Lavf57.57.100
  Duration: 00:00:39.87,start: 0.000000,bitrate: 724 kb/s
#输入文件的第0路流,格式为H.264、1280x720、30帧/秒等
    Stream #0:0(und): Video: h264(High)(avc1 / 0x31637661),yuv420p,1280x720
[SAR 1:1 DAR 16:9],590 kb/s,30 fps,30 tbr,15360 tbn,60 tbc(default)
    Metadata:
```

```
      handler_name    : VideoHandler
#输入文件的第 1 路流,格式为 AAC、44100Hz、立体声、fltp 采样格式等
    Stream #0: 1(und): Audio: aac(LC)(mp4a / 0x6134706D),44100 Hz,stereo,fltp,127
kb/s(default)
    Metadata:
      handler_name    : SoundHandler
#解析字幕信息
[Parsed_subtitles_0 @ 0c4f5f80] Shaper: FriBidi 1.0.9(SIMPLE)0/0
[Parsed_subtitles_0 @ 0c4f5f80] Using font provider directwrite/0
[Parsed_subtitles_0 @ 0c4f5f80] fontselect: (Arial,400,0) -> ArialMT,0,ArialMT
[Parsed_subtitles_0 @ 0c4f5f80] Glyph 0x8FD9 not found,selecting one more font for(A
rial,400,0)
[Parsed_subtitles_0 @ 0c4f5f80] fontselect: (Arial,400,0) -> MicrosoftYaHeiUI,1,Mi
crosoftYaHeiUI
  12.17 A-V: 0.001 fd= 54 aq= 19KB vq= 40KB sq= 0B f=0/0
```

3.2.3 ffplay 的数据可视化分析应用

使用 ffplay 除了可以播放视频流媒体文件之外，还可以作为可视化的视频流媒体分析工具。例如当播放音频文件时，有时不确定文件的声音是否正常，此时可以直接使用 ffplay 播放音频文件，播放时将会把解码后的音频数据以音频波形显示出来，如图 3-6 所示。从图中可以看到，音频在播放时的波形可以通过振幅显示出来，通过振幅可以判断音频的播放情况，命令如下：

```
ffplay -showmode 1 test3.mp3
```

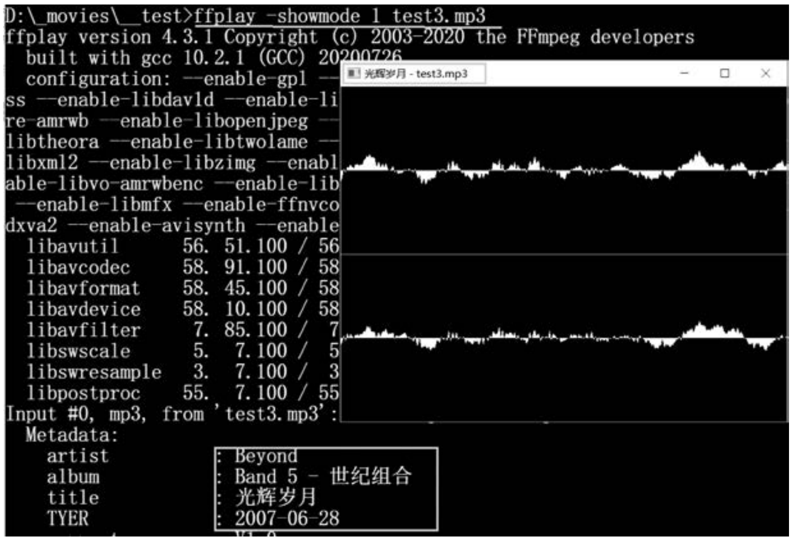

图 3-6 ffplay 播放音频并显示波形图

3.2.4 VLC 作为 RTSP 流媒体服务器

VLC 的功能很强大，不仅是一个视频播放器，也可作为小型的视频服务器，还可以一边播放一边转码，把视频流发送到网络上。VLC 作为视频服务器的具体步骤如下：

（1）单击主菜单中“媒体”下的“流”。

（2）在弹出的对话框中单击“添加”按钮，选择一个本地视频文件，如图 3-7 所示。

图 3-7 VLC 流媒体服务器之打开本地文件

（3）单击页面下方的“串流”，添加串流协议，如图 3-8 所示。

图 3-8 VLC 流媒体服务器之添加串流协议

(4) 该页面会显示刚才选择的本地视频文件,然后单击“下一步”按钮,如图 3-9 所示。

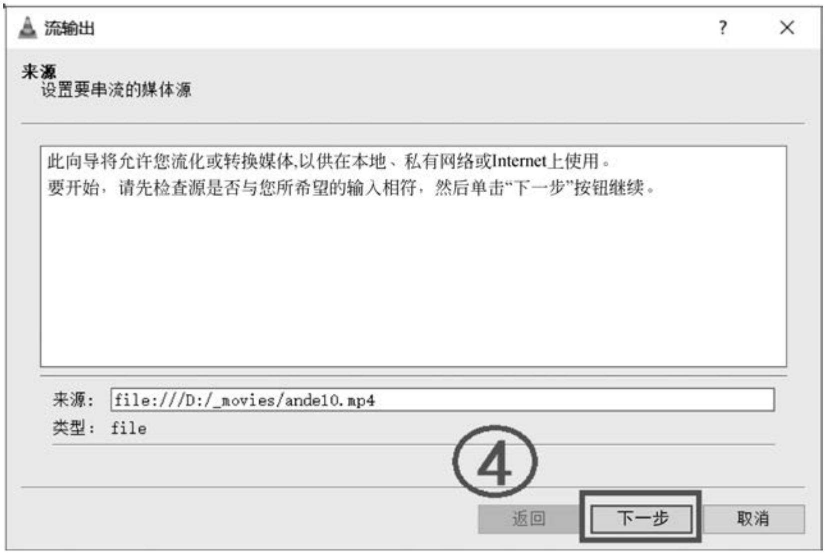

图 3-9 VLC 流媒体服务器之文件来源

(5) 在该页面单击“添加”按钮,选择具体的流协议,例如这里选择 RTSP,然后单击“下一步”按钮,如图 3-10 所示。

图 3-10 VLC 流媒体服务器之选择 RTSP 协议

(6) 在该页面的下拉列表框列表中选择 Video-H.264+MP3(TS),然后单击“下一步”按钮,如图 3-11 所示。

注意:一定要选中“激活转码”,并且需要是 TS 流格式。

(7) 在该页面可以看到 VLC 生成的所有串流输出参数,然后单击“流”按钮即可,如图 3-12 所示。

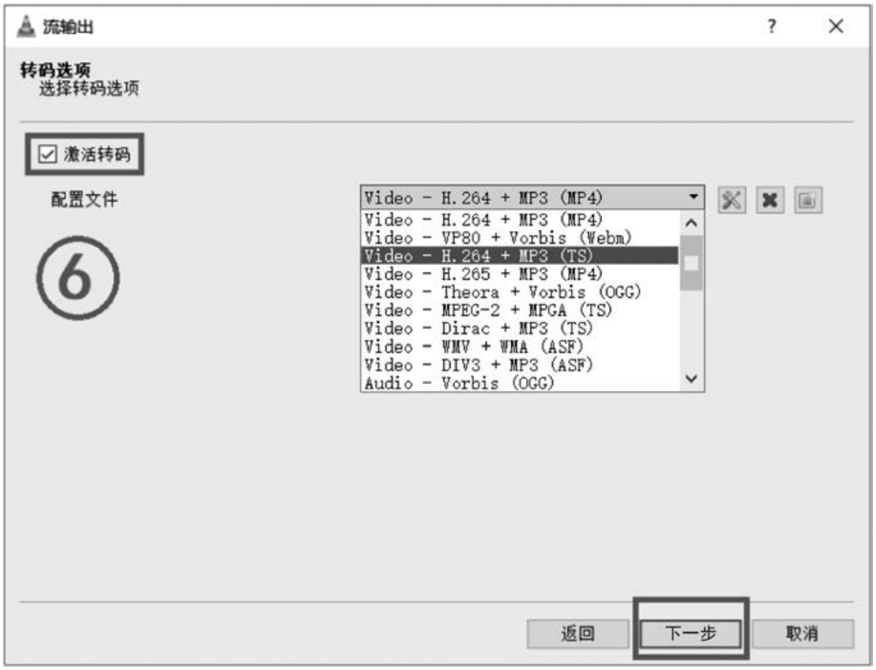

图 3-11 VLC 流媒体服务器之 H.264＋MP3(TS)

图 3-12 VLC 流媒体服务器之串流输出字符串

3.3 ffprobe 工具简介

ffprobe 工具主要用于检测多媒体信息，包括时长、视频分辨率、帧率、音频采样率、声道数、每个流（stream）信息等。支持的显示形式包括 json 和 xml，例如-print_format json 指定用 json 方式输出信息，命令如下：

```
ffprobe -i input.mp4 -show_streams -show_format -print_format json
```

如果要显示每帧数据信息，则可使用-show_frames，可以显示 pts、packet_size、duration、frame_type 等信息，命令如下：

```
ffprobe -i input.mp4 -show_frames
```

如果只显示视频流，则可使用-select_streams v，其中 v 代表 video，a 代表 audio，s 代表 subtitle，命令如下：

```
ffprobe -i input.mp4 -show_frames -select_streams v
```

ffprobe 常用参数比较多，可以通过 ffprobe--help 命令来查看详细的信息。

3.3.1 show_packets

show_packets：查看的多媒体包信息使用 PACKET 标签包括起来，其中包含的信息主要如下。

（1）codec_type：多媒体类型，例如视频包、音频包等。
（2）stream_index：多媒体的流索引（Stream Index）。
（3）pts：多媒体的显示时间值（Presentation Timestamp）。
（4）pts_time：根据不同格式计算过后的多媒体的显示时间。
（5）dts：多媒体解码时间值（Decoding Timestamp）。
（6）dts_time：根据不同格式计算过后的多媒体解码时间。
（7）duration：多媒体包占用的时间值。
（8）duration_time：根据不同格式计算过后的多媒体包占用的时间值。
（9）size：多媒体包的大小。
（10）pos：多媒体包所在的文件偏移位置。
（11）flags：多媒体包标记，关键包与非关键包的标记。

3.3.2 show_format

ffprobe 还可以分析多媒体的封装格式，通过-show_format 参数可以查看多媒体的封

装格式，封装可以使用 FORMAT 标签括起来显示，命令如下：

```
ffprobe -show_format test3.mp4
```

通过读取 format 信息，可以看到这个视频文件有两个流通道，包括视频流 Stream ＃0:0(und):Video:h264 和音频流 Stream ＃0:1(und):Audio:aac。这个文件的起始时间是 0.000000，长度为 00:00:08.55，文件大小为 659 171B，码率为 617kb/s。该文件的封装格式可能为 MOV、MP4、M4A、3GP、3G2、MJ2，之所以 ffprobe 会这么输出，是因为这几种封装格式在 FFmpeg 中所识别的标签基本相同，所以才会有这么多种显示方式。完整的输出信息，代码如下(以＃开头的部分是笔者的注释信息)：

```
//chapter3/3.1-ffpbore-out.txt
D:\_movies\__test>ffprobe -show_format test3.mp4
ffprobe version 4.3.1 Copyright(c) 2007-2020 the FFmpeg developers
  built with gcc 10.2.1(GCC) 20200726
  configuration: --enable-gpl --enable-version3 --enable-sdl2 --enable-
fontconfig --e
able-gnutls --enable-iconv --enable-libass --enable-libdav1d --enable-libbluray
--ena
le-libfreetype --enable-libmp3lame ......
  libavutil        56. 51.100 / 56. 51.100
  libavcodec       58. 91.100 / 58. 91.100
  libavformat      58. 45.100 / 58. 45.100
  libavdevice      58. 10.100 / 58. 10.100
  libavfilter      7. 85.100 / 7. 85.100
  libswscale       5. 7.100 / 5. 7.100
  libswresample    3. 7.100 / 3. 7.100
  libpostproc      55. 7.100 / 55. 7.100
#输入文件的封装格式,这里为 MP4 封装格式
Input #0,mov,mp4,m4a,3gp,3g2,mj2,from 'test3.mp4':
  Metadata:
    major_brand     : isom
    minor_version   : 512
    compatible_brands: isomiso2avc1mp41
    encoder         : Lavf57.57.100
  Duration: 00:00:08.55,start: 0.000000,bitrate: 617 kb/s

#第 0 路流: 视频流,H.264 编码格式
    Stream #0:0(und): Video: h264(High)(avc1 / 0x31637661),yuv420p,1280x720
[SAR: 1 DAR 16:9],553 kb/s,30 fps,30 tbr,16k tbn,60 tbc(default)
    Metadata:
      handler_name  : VideoHandler

#第 1 路流: 音频流,AAC 编码格式
```

```
Stream #0:1(und): Audio: aac(LC)(mp4a / 0x6134706D),44100 Hz,stereo,fltp,12
kb/s(default)
    Metadata:
      handler_name : SoundHandler

#[FORMAT]标签格式的输出信息,格式为 name = value
[FORMAT]
filename = test3.mp4
nb_streams = 2
nb_programs = 0
format_name = mov,mp4,m4a,3gp,3g2,mj2
format_long_name = QuickTime / MOV
start_time = 0.000000
duration = 8.545000
size = 659171
bit_rate = 617129
probe_score = 100
TAG: major_brand = isom
TAG: minor_version = 512
TAG: compatible_brands = isomiso2avc1mp41
TAG: encoder = Lavf57.57.100
[/FORMAT]
```

3.3.3 show_frames

通过 ffprobe 的 show_frames 参数可以查看视频文件中的帧信息,输出的帧信息使用 FRAME 标签括起来,能够看到每帧的信息,命令如下:

```
ffprobe -show_frames -i test3.mp4
```

下面介绍重要的属性说明,如表 3-2 所示。

表 3-2 show_frames 的属性说明

属 性	值	属 性	值
media_type	帧的类型(视频、音频、字幕等)	pkt_duration	Frame 包的时长
stream_index	帧所在的索引区域	pkt_duration_time	Frame 包的时长时间显示
key_frame	是否是关键帧	pkt_pos	Frame 包所在文件的偏移位置
pkt_pts	Frame 包的 pts	width	帧显示的宽度
pkt_pts_time	Frame 包的 pts 的时间显示	height	帧显示的高度
pkt_dts	Frame 包的 dts	pix_fmt	帧的图像色彩格式(YUV、RGB 等)
pkt_dts_time	Frame 包的 dts 的时间显示	pict_fmt	帧类型(I、P、B)

从该案例中的部分输出信息可以看出,音视频帧是交错输出的,一般情况是一帧视频,

然后是多帧音频，以此交错类推，输出信息如下（以＃开头的部分是笔者的注释信息）：

```
//chapter3/test3.mp4.frames.txt
//chapter3/完整的 ffprobe - show_frames 输出的帧信息
D:\_movies\__test > ffprobe - show_frames - i test3.mp4
ffprobe version 4.3.1 Copyright(c) 2007 - 2020 the FFmpeg developers
  built with gcc 10.2.1(GCC) 20200726
  configuration: -- enable - gpl -- enable - version3 -- enable - sdl2 -- enable -
fontconfig -- en
able - gnutls -- enable - iconv -- enable - libass -- enable - libdav1d -- enable - libbluray
-- enab
le - libfreetype -- enable - libmp3lame ......
  libavutil        56. 51.100 / 56. 51.100
  libavcodec       58. 91.100 / 58. 91.100
  libavformat      58. 45.100 / 58. 45.100
  libavdevice      58. 10.100 / 58. 10.100
  libavfilter      7. 85.100 / 7. 85.100
  libswscale       5. 7.100 / 5. 7.100
  libswresample    3. 7.100 / 3. 7.100
  libpostproc      55. 7.100 / 55. 7.100
Input #0,mov,mp4,m4a,3gp,3g2,mj2,from 'test3.mp4':
  Metadata:
    major_brand     : isom
    minor_version   : 512
    compatible_brands: isomiso2avc1mp41
    encoder         : Lavf57.57.100
  Duration: 00: 00: 08.55,start: 0.000000,bitrate: 617 kb/s
    Stream #0: 0(und): Video: h264(High)(avc1 / 0x31637661),yuv420p,1280x720 [SAR 1: 1
DAR 16: 9],553 kb/s,30 fps,30 tbr,16k tbn,60 tbc(default)
  Metadata:
    major_brand     : isom
    minor_version   : 512
    compatible_brands: isomiso2avc1mp41
    encoder         : Lavf57.57.100
  Duration: 00: 00: 08.55,start: 0.000000,bitrate: 617 kb/s
    Stream #0: 0(und): Video: h264(High)(avc1 / 0x31637661),yuv420p,1280x720 [SAR 1: 1
DAR 16: 9],553 kb/s,30 fps,30 tbr,16k tbn,60 tbc(default)
    Metadata:
      handler_name  : VideoHandler
    Stream #0: 1(und): Audio: aac(LC)(mp4a / 0x6134706D),44100 Hz,stereo,fltp,127
kb/s(default)
    Metadata:
      handler_name  : SoundHandler
#音频帧: audio
[FRAME]
media_type = audio
stream_index = 1
key_frame = 1
```

```
pkt_pts = 0
pkt_pts_time = 0.000000
pkt_dts = 0
pkt_dts_time = 0.000000
best_effort_timestamp = 0
best_effort_timestamp_time = 0.000000
pkt_duration = 1024
pkt_duration_time = 0.023220
pkt_pos = 48
pkt_size = 378
sample_fmt = fltp
nb_samples = 1024
channels = 2
channel_layout = stereo
[/FRAME]

#音频帧: audio
[FRAME]
media_type = audio
stream_index = 1
key_frame = 1
pkt_pts = 1024
pkt_pts_time = 0.023220
pkt_dts = 1024
pkt_dts_time = 0.023220
best_effort_timestamp = 1024
best_effort_timestamp_time = 0.023220
pkt_duration = 1024
pkt_duration_time = 0.023220
pkt_pos = 426
pkt_size = 373
sample_fmt = fltp
nb_samples = 1024
channels = 2
channel_layout = stereo
[/FRAME]

#音频帧: audio
[FRAME]
media_type = audio
stream_index = 1
key_frame = 1
pkt_pts = 2048
pkt_pts_time = 0.046440
pkt_dts = 2048
```

```
pkt_dts_time = 0.046440
best_effort_timestamp = 2048
best_effort_timestamp_time = 0.046440
pkt_duration = 1017
pkt_duration_time = 0.023061
pkt_pos = 799
pkt_size = 369
sample_fmt = fltp
nb_samples = 1024
channels = 2
channel_layout = stereo
[/FRAME]

#音频帧: audio
[FRAME]
media_type = audio
stream_index = 1
key_frame = 1
pkt_pts = 3065
pkt_pts_time = 0.069501
pkt_dts = 3065
pkt_dts_time = 0.069501
best_effort_timestamp = 3065
best_effort_timestamp_time = 0.069501
pkt_duration = 1024
pkt_duration_time = 0.023220
pkt_pos = 1168
pkt_size = 367
sample_fmt = fltp
nb_samples = 1024
channels = 2
channel_layout = stereo
[/FRAME]

#音频帧: audio
[FRAME]
media_type = audio
stream_index = 1
key_frame = 1
pkt_pts = 5113
pkt_pts_time = 0.115941
pkt_dts = 5113
pkt_dts_time = 0.115941
best_effort_timestamp = 5113
best_effort_timestamp_time = 0.115941
pkt_duration = 1024
```

```
pkt_duration_time = 0.023220
pkt_pos = 1908
pkt_size = 356
sample_fmt = fltp
nb_samples = 1024
channels = 2
channel_layout = stereo
[/FRAME]
...
#视频帧: video
[FRAME]
media_type = video
stream_index = 0
key_frame = 1
pkt_pts = 19088
pkt_pts_time = 1.193000
pkt_dts = 19088
pkt_dts_time = 1.193000
best_effort_timestamp = 19088
best_effort_timestamp_time = 1.193000
pkt_duration = 533
pkt_duration_time = 0.033313
pkt_pos = 18360
pkt_size = 107280
width = 1280
height = 720
pix_fmt = yuv420p
sample_aspect_ratio = 1:1
pict_type = I
coded_picture_number = 0
display_picture_number = 0
interlaced_frame = 0
top_field_first = 0
repeat_pict = 0
color_range = unknown
color_space = unknown
color_primaries = unknown
color_transfer = unknown
chroma_location = left
[/FRAME]
...
```

3.3.4 show_streams

通过 ffprobe 的 show_streams 参数可以查看多媒体文件中的流信息，流信息使用 STREAMS 标签括起来，命令如下：

```
ffprobe -show_streams -i test3.mp4
```

下面介绍重要的属性说明,如表 3-3 所示。

表 3-3　show_streams 的属性说明

属　　性	值	属　　性	值
index	流所在的索引区域	codec_type	编码类型
codec_name	编码名	codec_time_base	编码的时间戳的基础单位
codec_long_name	编码全名	pix_fmt	图像显示的图像色彩格式
profile	编码的能力	coded_width	图像的宽度
level	编码的层级	coded_height	图像的高度
has_b_frames	包含 B 帧信息	codec_tag_string	编码的标签数据

该案例中的部分输出信息如下(以#开头的部分是笔者的注释信息):

```
//chapter3/test3.mp4.streams.txt
#STREAM: 0, 视频流,H.264,1280x720,......
[STREAM]
index=0
codec_name=h264
codec_long_name=H.264 / AVC / MPEG-4 AVC / MPEG-4 part 10
profile=High
codec_type=video
codec_time_base=1/60
codec_tag_string=avc1
codec_tag=0x31637661
width=1280
height=720
coded_width=1280
coded_height=720
closed_captions=0
has_b_frames=2
sample_aspect_ratio=1:1
display_aspect_ratio=16:9
pix_fmt=yuv420p
level=31
color_range=unknown
color_space=unknown
color_transfer=unknown
color_primaries=unknown
chroma_location=left
field_order=unknown
timecode=N/A
```

```
refs = 1
is_avc = true
nal_length_size = 4
id = N/A
r_frame_rate = 30/1
avg_frame_rate = 30/1
time_base = 1/16000
start_pts = 19088
start_time = 1.193000
duration_ts = 118400
duration = 7.400000
bit_rate = 553246
max_bit_rate = N/A
bits_per_raw_sample = 8
nb_frames = 222
nb_read_frames = N/A
nb_read_packets = N/A
DISPOSITION: default = 1
DISPOSITION: dub = 0
DISPOSITION: original = 0
DISPOSITION: comment = 0
DISPOSITION: lyrics = 0
DISPOSITION: karaoke = 0
DISPOSITION: forced = 0
DISPOSITION: hearing_impaired = 0
DISPOSITION: visual_impaired = 0
DISPOSITION: clean_effects = 0
DISPOSITION: attached_pic = 0
DISPOSITION: timed_thumbnails = 0
TAG: language = und
TAG: handler_name = VideoHandler
[/STREAM]

#STREAM: 1, 音频流,AAC,44100Hz,2 声道,......
[STREAM]
index = 1
codec_name = aac
codec_long_name = AAC(Advanced Audio Coding)
profile = LC
codec_type = audio
codec_time_base = 1/44100
codec_tag_string = mp4a
codec_tag = 0x6134706d
sample_fmt = fltp
sample_rate = 44100
channels = 2
```

```
channel_layout = stereo
bits_per_sample = 0
id = N/A
r_frame_rate = 0/0
avg_frame_rate = 0/0
time_base = 1/44100
start_pts = 0
start_time = 0.000000
duration_ts = 376811
duration = 8.544467
bit_rate = 127887
max_bit_rate = 127887
bits_per_raw_sample = N/A
nb_frames = 368
nb_read_frames = N/A
nb_read_packets = N/A
DISPOSITION: default = 1
DISPOSITION: dub = 0
DISPOSITION: original = 0
DISPOSITION: comment = 0
DISPOSITION: lyrics = 0
DISPOSITION: karaoke = 0
DISPOSITION: forced = 0
DISPOSITION: hearing_impaired = 0
DISPOSITION: visual_impaired = 0
DISPOSITION: clean_effects = 0
DISPOSITION: attached_pic = 0
DISPOSITION: timed_thumbnails = 0
TAG: language = und
TAG: handler_name = SoundHandler
[/STREAM]
```

3.3.5 print_format

ffprobe使用前面的参数可以获得到对应的key-value，但是阅读起来因个人习惯不同，所以有的人认为方便，而有的人认为不方便，这样就需要用到ffprobe的print_format参数进行相应的格式输出(包括XML、JSON、INI、CSV、FLAT)，下面举几种输出的例子：

```
//chapter3/print_format-help.txt
ffprobe -show_frames test3.mp4 -print_format json
ffprobe -show_frames test3.mp4 -print_format xml
ffprobe -show_frames test3.mp4 -print_format flat
```

JSON格式的部分输出信息如下：

```
//chapter3/test3.mp4.print_format.json.txt
{
    "frames": [
        {
            "media_type": "audio",
            "stream_index": 1,
            "key_frame": 1,
            "pkt_pts": 0,
            "pkt_pts_time": "0.000000",
            "pkt_dts": 0,
            "pkt_dts_time": "0.000000",
            "best_effort_timestamp": 0,
            "best_effort_timestamp_time": "0.000000",
            "pkt_duration": 1024,
            "pkt_duration_time": "0.023220",
            "pkt_pos": "48",
            "pkt_size": "378",
            "sample_fmt": "fltp",
            "nb_samples": 1024,
            "channels": 2,
            "channel_layout": "stereo"
        },
        {
            "media_type": "audio",
            "stream_index": 1,
            "key_frame": 1,
            "pkt_pts": 1024,
            "pkt_pts_time": "0.023220",
            "pkt_dts": 1024,
            "pkt_dts_time": "0.023220",
            "best_effort_timestamp": 1024,
            "best_effort_timestamp_time": "0.023220",
            "pkt_duration": 1024,
            "pkt_duration_time": "0.023220",
            "pkt_pos": "426",
            "pkt_size": "373",
            "sample_fmt": "fltp",
            "nb_samples": 1024,
            "channels": 2,
            "channel_layout": "stereo"
        }
...
  ]
}
```

XML 格式的部分输出信息如下：

```
//chapter3/test3.mp4.print_format.xml.txt
<?xml version = "1.0" encoding = "UTF - 8"?>
< ffprobe >
    < frames >
        < frame media_type = "audio" stream_index = "1" key_frame = "1" pkt_pts = "0" pkt_pts_time = "0.000000" pkt_dts = "0" pkt_dts_time = "0.000000" best_effort_timestamp = "0" best_effort_timestamp_time = "0.000000" pkt_duration = "1024" pkt_duration_time = "0.023220" pkt_pos = "48" pkt_size = "378" sample_fmt = "fltp" nb_samples = "1024" channels = "2" channel_layout = "stereo"/>
        < frame media_type = "audio" stream_index = "1" key_frame = "1" pkt_pts = "1024" pkt_pts_time = "0.023220" pkt_dts = "1024" pkt_dts_time = "0.023220" best_effort_timestamp = "1024" best_effort_timestamp_time = "0.023220" pkt_duration = "1024" pkt_duration_time = "0.023220" pkt_pos = "426" pkt_size = "373" sample_fmt = "fltp" nb_samples = "1024" channels = "2" channel_layout = "stereo"/>
        < frame media_type = "audio" stream_index = "1" key_frame = "1" pkt_pts = "2048" pkt_pts_time = "0.046440" pkt_dts = "2048" pkt_dts_time = "0.046440" best_effort_timestamp = "2048" best_effort_timestamp_time = "0.046440" pkt_duration = "1017" pkt_duration_time = "0.023061" pkt_pos = "799" pkt_size = "369" sample_fmt = "fltp" nb_samples = "1024" channels = "2" channel_layout = "stereo"/>
        < frame media_type = "audio" stream_index = "1" key_frame = "1" pkt_pts = "3065" pkt_pts_time = "0.069501" pkt_dts = "3065" pkt_dts_time = "0.069501" best_effort_timestamp = "3065" best_effort_timestamp_time = "0.069501" pkt_duration = "1024" pkt_duration_time = "0.023220" pkt_pos = "1168" pkt_size = "367" sample_fmt = "fltp" nb_samples = "1024" channels = "2" channel_layout = "stereo"/>
...
</frames >
</ffprobe >
```

FLAT 格式的部分输出信息如下：

```
//chapter3/test3.mp4.print_format.flat.txt
frames.frame.0.media_type = "audio"
frames.frame.0.stream_index = 1
frames.frame.0.key_frame = 1
frames.frame.0.pkt_pts = 0
frames.frame.0.pkt_pts_time = "0.000000"
frames.frame.0.pkt_dts = 0
frames.frame.0.pkt_dts_time = "0.000000"
frames.frame.0.best_effort_timestamp = 0
frames.frame.0.best_effort_timestamp_time = "0.000000"
frames.frame.0.pkt_duration = 1024
frames.frame.0.pkt_duration_time = "0.023220"
frames.frame.0.pkt_pos = "48"
frames.frame.0.pkt_size = "378"
frames.frame.0.sample_fmt = "fltp"
```

```
frames.frame.0.nb_samples=1024
frames.frame.0.channels=2
frames.frame.0.channel_layout="stereo"
frames.frame.1.media_type="audio"
frames.frame.1.stream_index=1
frames.frame.1.key_frame=1
frames.frame.1.pkt_pts=1024
frames.frame.1.pkt_pts_time="0.023220"
frames.frame.1.pkt_dts=1024
frames.frame.1.pkt_dts_time="0.023220"
frames.frame.1.best_effort_timestamp=1024
frames.frame.1.best_effort_timestamp_time="0.023220"
frames.frame.1.pkt_duration=1024
frames.frame.1.pkt_duration_time="0.023220"
frames.frame.1.pkt_pos="426"
frames.frame.1.pkt_size="373"
frames.frame.1.sample_fmt="fltp"
frames.frame.1.nb_samples=1024
frames.frame.1.channels=2
frames.frame.1.channel_layout="stereo"
...
```

3.3.6 select_streams

如果只查看音频流或视频流，则可使用-select_streams 参数，例如只查看视频流的 frames 信息，命令如下：

```
ffprobe -show_frames -select_streams v -of json test3.mp4
```

命令行执行后可以看到输出信息全部为视频的 frames，内容如下：

```
//chapter3/test3.mp4.select_streams-v-ofjson.txt
{
    "frames": [
        {
            "media_type": "video",
            "stream_index": 0,
            "key_frame": 1,
            "pkt_pts": 19088,
            "pkt_pts_time": "1.193000",
            "pkt_dts": 19088,
            "pkt_dts_time": "1.193000",
            "best_effort_timestamp": 19088,
            "best_effort_timestamp_time": "1.193000",
```

```
            "pkt_duration": 533,
            "pkt_duration_time": "0.033313",
            "pkt_pos": "18360",
            "pkt_size": "107280",
            "width": 1280,
            "height": 720,
            "pix_fmt": "yuv420p",
            "sample_aspect_ratio": "1:1",
            "pict_type": "I",
            "coded_picture_number": 0,
            "display_picture_number": 0,
            "interlaced_frame": 0,
            "top_field_first": 0,
            "repeat_pict": 0,
            "chroma_location": "left"
        },
        {
            "media_type": "video",
            "stream_index": 0,
            "key_frame": 0,
            "pkt_pts": 19616,
            "pkt_pts_time": "1.226000",
            "pkt_dts": 19616,
            "pkt_dts_time": "1.226000",
            "best_effort_timestamp": 19616,
            "best_effort_timestamp_time": "1.226000",
            "pkt_duration": 533,
            "pkt_duration_time": "0.033313",
            "pkt_pos": "130212",
            "pkt_size": "170",
            "width": 1280,
            "height": 720,
            "pix_fmt": "yuv420p",
            "sample_aspect_ratio": "1:1",
            "pict_type": "B",
            "coded_picture_number": 3,
            "display_picture_number": 0,
            "interlaced_frame": 0,
            "top_field_first": 0,
            "repeat_pict": 0,
            "chroma_location": "left"
        },
...
    ]
}
```

3.4 通用选项

通用选项是指对音频、视频都能使用的命令参数，这里介绍几个重要的选项。

（1）-f fmt(input/output)：强制输入或输出文件格式。通常，输入文件的格式是自动检测的，输出文件的格式是通过文件扩展名进行推断的。

（2）-i filename(input)：输入文件名。

（3）-y(global)：如果存在同名的输出文件，则直接强制覆盖。

（4）-n(global)：不覆盖输出文件，如果一个给定的输出文件已经存在，则立即退出。

（5）-c[:stream_specifier] codec(input/output、per-stream)：编解码器选项，例如-c:v 代表视频编码器，-c:a 代表音频编码器，-c copy 代表音视频编码格式都复制原来的。也可以写成-codec[:stream_specifier] codec(input/output,per-stream)，为一个或多个流选择一个编码器（当使用在一个输出文件之前时）或者一个解码器（当使用在一个输入文件之前时）。codec 是一个编码器/解码器名称（例如 libx264、aac、libmp3lame 等）或者一个特定值 copy（只适用输出）。

（6）-t duration(output)：当到达 duration 时，停止写输出。duration 可以是一个数字（单位为秒），或者使用 hh:mm:ss[.xxx]形式。

（7）-to position(output)：在 position 处停止写输出。duration 可以是一个数字（单位为秒），或者使用 hh:mm:ss[.xxx]形式。选项-to 和-t 是互斥的，-t 的优先级更高。

（8）-fs limit_size(output)：设置文件大小限制，以字节表示。

（9）-ss position(input/output)：当作为输入选项时（在 -i 之前），在输入文件中跳转到 position。需要注意的是，在大多数格式中，不太可能精确地跳转，因此，FFmpeg 将跳转到 position 之前最接近的位置。当进行转码并且-accurate_seek 打开时（默认会打开），位于跳转点和 position 之间的额外部分将被解码并且丢弃。当进行流复制或者使用-noaccurate_seek 时，它将被保留下来。当作为输出选项时（在输出文件名前），会解码，但是丢弃输入，直到时间戳到达 position。position 可以是一个数字（单位为秒）或者 hh:mm:ss[.xxx]形式。

（10）-itsoffset offset(input)：设置输入时间偏移。offset 将被添加到输入文件的时间戳。指定一个正偏移，意味着相应的流将被延时指定时间。

（11）-timestamp date(output)：在容器中设置录音时间戳。

（12）-metadata[:metadata_specifier] key=value(output,per-metadata)：设置 metadata 的 key-value 对。

（13）-target type(output)：指定目标文件类型（VCD、SVCD、DVD、DV 和 DV50）。type 可以带有 pal-、ntsc-或 film-前缀，以使用相应的标准。所有的格式选项（bitrate、codecs、buffer sizes）将自动设定。

(14) -dframes number(output):设置要录制数据帧的个数。这是-frames:d 的别名。

(15) -frames[:stream_specifier] framecount(output、per-stream):framecount 帧以后停止写流。

(16) -q[:stream_specifier] q(output、per-stream):指定音视频质量,如-q:v 2,其中 2 代表保存为高质量。或者写成-qscale[:stream_specifier] q(output、per-stream),可以使用固定质量范围。

(17) -filter[:stream_specifier] filtergraph(output,per-stream):创建 filtergraph 指定的过滤图,并使用它过滤流。

(18) -filter_script[:stream_specifier] filename(output、per-stream):该选项与-filter 相似,唯一的不同是,它的参数是一个存放过滤图的文件的名称。

(19) -pre[:stream_specifier] preset_name(output,per-stream):指定匹配流的预设。

(20) -stats(global):打印编码进程/统计信息。默认打开,可以使用-nostats 禁用。

(21) -stdin:开启标准输入交互。默认打开,除非标准输入作为一个输入。可以使用-nostdin 禁止。

(22) -DeBug_ts(global):打印时间戳信息,默认关闭。

(23) -attach filename(output):将一个附件添加到输出文件中。

(24) -dump_attachment[:stream_specifier] filename(input,per-stream):将匹配的附件流提取到 filename 指定的文件中。

3.5 视频选项

主要应用于视频的选项,列举如下。

(1) -vframes number(output):设置录制视频帧的个数,它是-frames:v 的别名。

(2) -r[:stream_specifier] fps(input/output、per-stream):设置帧率。

(3) -s[:stream_specifier] size(input/output、per-stream):设置帧大小,格式为 width x height,默认与原始的宽和高相同,例如 640x360(注意中间是小写的 x)。

(4) -aspect[:stream_specifier] aspect(output、per-stream):设置视频显示的宽高比。

(5) -vn(output):禁止视频录制。

(6) -vcodec codec(output):设置视频 codec,它是-codec:v 的别名,例如-vcodec libx264 代表用 libx264 进行视频编码,-vcodec copy 代表不用重新编码而直接复制原始视频流。

(7) -pass[:stream_specifier] n(output、per-stream):选择 pass number(1 or 2),用来进行双行程视频编码。

(8) -passlogfile[:stream_specifier] prefix(output、per-stream):设置 two-pass 日志文件名前缀,默认为 ffmpeg2pass。

(9) -vf filtergraph(output)：创建 filtergraph 指定的过滤图,并使用它来过滤流。

(10) -pix_fmt[:stream_specifier] format(input/output、per-stream)：设置像素格式。

(11) -sws_flags flags(input/output)：设置软缩放标志。

(12) -vdt n：丢弃阈值。

(13) -psnr：用于计算压缩帧的 PSNR(Peak Signal to Noise Ratio),即峰值信噪比,是一种评价图像的客观标准。

(14) -vstats：将视频编码统计信息复制到 vstats_HHMMSS.log。

(15) -vstats_file file：将视频编码统计信息复制到指定的 file。

(16) -force_key_frames[:stream_specifier] time[,time...](output、per-stream)：强制关键帧。

(17) -force_key_frames[:stream_specifier] expr:expr(output、per-stream)：在指定的时间戳强制关键帧。

(18) -copyinkf[:stream_specifier](output、per-stream)：当进行流复制时,同时复制开头的非关键帧。

(19) -hwaccel[:stream_specifier] hwaccel(input、per-stream)：使用硬件加速来解码匹配的流。

(20) -hwaccel_device[:stream_specifier] hwaccel_device(input、per-stream)：选择硬件加速所使用的设备,该选项只有-hwaccel 同时指定时才有意义。

3.6 音频选项

主要应用于音频的选项,列举如下。

(1) -aframes number(output)：设置录制音频帧的个数,是-frames:a 的别名。

(2) -ar[:stream_specifier] freq(input/output、per-stream)：设置音频采样率。

(3) -aq q(output)：设置音频质量,是-q:a 的别名。

(4) -ac[:stream_specifier] channels(input/output、per-stream)：设置音频通道数。

(5) -an(output)：禁止音频录制。

(6) -acodec codec(input/output)：设置音频 codec,是-codec:a 的别名。例如-acodec aac 代表使用 aac 进行音频编码。

(7) -sample_fmt[:stream_specifier] sample_fmt(output、per-stream)：设置音频采样格式。

(8) -af filtergraph(output)：创建 filtergraph 所指定的过滤图,并使用它来过滤流。

3.7 字幕选项

主要应用于字幕的选项,列举如下。

(1) -s size：设置框架大小(width x height 或缩写)。

(2) -sn：禁用字幕。

(3) -scodec：编解码器强制字幕编解码器。

(4) -stag fourcc/tag：字幕标签/fourcc。

(5) -fix_sub_duration：修复字幕持续时间。

(6) -canvas_size size：设置画布大小(width x height)。

(7) -spre：预设将字幕选项设置为指示的预设。

3.8 高级选项

(1) -map [-]input_file_id[:stream_specifier][,sync_file_id[:stream_specifier]] | [linklabel](output)：指定一个或多个流作为输出文件的源。

(2) -re(input)：以本地视频文件的帧率来读取数据，主要用来模拟一个采集设备，例如可以读取一个本地文件来模拟实时输入流。

(3) -map_metadata[:metadata_spec_out] infile[:metadata_spec_in](output、per-metadata)：设置下一个输出文件的 metadata 信息。

(4) -map_chapters input_file_index(output)：从索引号为 input_file_index 的输入文件中将章节复制到下一个输出文件中。

(5) -timelimit duration(global)：FFmpeg 运行 duration 秒后退出。

(6) -dump(global)：将每个输入包复制到标准输出。

(7) -hex(global)：复制包时，同时复制负载。

(8) -vsync parameter：视频同步方法。

(9) -async samples_per_second：音频同步方法。

(10) -shortest(output)：当最短的输入流结束时，终止编码。

(11) -muxdelay seconds(input)：设置最大解封装、解码延时。

(12) -muxpreload seconds(input)：设置初始解封装、解码延时。

(13) -streamid output-stream-index:new-value(output)：为一个输出流分配一个新的 stream-id。

(14) -bsf[:stream_specifier] bitstream_filters(output、per-stream)：为匹配的流设置比特流滤镜。

(15) -filter_complex filtergraph(global)：定义一个复杂的过滤图。

(16) -lavfi filtergraph(global)：定义一个复杂的过滤图，相当于-filter_complex。

(17) -filter_complex_script filename(global)：该选项类似于-filter_complex，唯一的不同是它的参数是一个定义过滤图的文件的文件名。

(18) -accurate_seek(input)：打开或禁止在输入文件中的精确跳转，默认打开。

3.9 map 详解

FFmpeg 的-map 选项用于指定一个或多个流作为输出文件的源，语法如下：

```
- map [ - ] input_file_id[: stream_specifier][, sync_file_id[: stream_specifier]] |
[linklabel](output)
```

该命令行中的第 1 个-map 选项用于指定输出流 0 的源；第 2 个-map 选项用于指定输出流 1 的源，以此类推。以下是一些特别流符号的说明：

（1）-map 0 表示选择第 1 个文件的所有流。

（2）-map i:v 表示从文件序号 i(index)中获取所有视频流。

（3）-map i:a 表示获取所有音频流。

（4）-map i:s 表示获取所有字幕流等。

（5）特殊参数-an、-vn、-sn 表示分别排除所有的音频流、视频流、字幕流。

注意：如果文件索引号前带有减号(—)，则表示排除指定文件中的指定流。例如-map -0:v:0 表示排除第 1 个输入文件中的第 1 路视频流。

-map 的通常用法是-map file_number[:stream_type][:stream_number]，即"-map 文件索引号:流类型:流索引号"，下面列举几个案例，如图 3-13 所示。

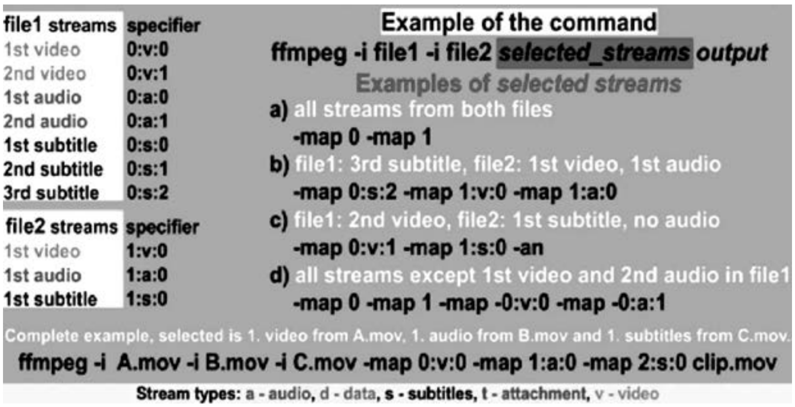

图 3-13　map 用法及案例

在 map 用法中，文件、流(包括音频流、视频流、字幕流)的索引号都是从 0 开始的，例如第 1 个文件的第 1 路视频流的标识符(specifier)为 0:v:0、第 1 个文件的第 2 路视频流的标识符为 0:v:1、第 2 个文件的第 3 路字幕流的标识符为 1:s:2。

在上述案例中有两个输入文件 file1 和 file2，一个输出文件 output，命令行语法如下：

```
ffmpeg -i file1 -i file2 selected_streams output
```

其中，列举几种 selected_streams 可用的情况。

a）从所有输入文件中选择所有流，代码如下：

```
-map 0 -map 1
#-map 0 表示从第 1 个文件中获取所有流，包括音频流、视频流和字幕流
#-map 1 表示从第 2 个文件中获取所有流，包括音频流、视频流和字幕流
```

b）从第 1 个输入文件中获取第 3 路字幕流，从第 2 个文件中获取第 1 路视频流和第 1 路音频流，代码如下：

```
-map 0:s:2 -map 1:v:0 -map 1:a:0
#-map 0:s:2 表示从第 1 个文件中获取第 3 路字幕流
#-map 1:v:0 表示从第 2 个文件中获取第 1 路视频流
#-map 1:a:0 表示从第 2 个文件中获取第 1 路音频流
```

c）从第 1 个输入文件中获取第 2 路视频流，从第 2 个文件中获取第 1 路字幕流，然后屏蔽所有的音频流，代码如下：

```
-map 0:v:1 -map 1:s:0 -an
#-map 0:v:1 表示从第 1 个文件中获取第 2 路视频流
#-map 1:s:0 表示从第 2 个文件中获取第 1 路字幕流
#-an 表示屏蔽所有的音频流
```

d）从所有输入文件中选择所有流，但需要排除第 1 个输入文件中的第 1 路视频流和第 2 路音频流，代码如下：

```
-map 0 -map 1 -map -0:v:0 -map -0:a:1
#-map 0 表示从第 1 个文件中获取所有流，包括音频流、视频流和字幕流
#-map 1 表示从第 2 个文件中获取所有流，包括音频流、视频流和字幕流
#-map -0:v:0 注意文件索引号前边有一个减号(-)，表示从第 1 个文件中排除第 1 路视频流
#-map -0:a:1 注意文件索引号前边有一个减号(-)，表示从第 1 个文件中排除第 2 路音频流
```

3.10 ffmpeg -h 详解

打开命令行窗口，输入 ffmpeg -h 命令，会打印出 FFmpeg 的帮助信息，输出内容如下：

```
//chapter3/ffmpeg.h.10.comment.txt
//chapter3/ffmpeg-h-注释说明
Hyper fast Audio and Video encoder
#ffmpeg 的语法格式
usage: ffmpeg [options] [[infile options] -i infile]... {[outfile options] outfile}...
```

```
#ffmpeg的获取帮助的参数,包括 -h、-h long 和 -h full
#由于内容比较多,读者可以自己测试 -h long 和 -h full
Getting help:
    -h      -- print basic options
    -h long -- print more options
    -h full -- print all options(including all format and codec specific options,very long)
    -h type=name -- print all options for the named decoder/encoder/demuxer/muxer/filter/
bsf/protocol
    See man ffmpeg for detailed description of the options.

Print help / information / capabilities:
-L                      show license                          #显示授权信息
-h topic                show help                             #显示帮助信息
-? topic                show help                             #显示帮助信息
-help topic             show help
--help topic            show help
-version                show version                          #显示版本信息
-buildconf              show build configuration              #显示编译配置信息
-formats                show available formats                #显示可用的格式
-muxers                 show available muxers                 #显示可用的封装器
-demuxers               show available demuxers               #显示可用的解封装器
-devices                show available devices                #显示可用的设备
-codecs                 show available codecs                 #显示可用的编解码器
-decoders               show available decoders               #显示可用的解码器
-encoders               show available encoders               #显示可用的编码器
-bsfs                   show available bit stream filters     #显示可用的位流滤镜
-protocols              show available protocols              #显示可用的协议
-filters                show available filters                #显示可用的滤镜
-pix_fmts               show available pixel formats          #显示可用的像素格式
-layouts                show standard channel layouts         #显示可用的声道模式
-sample_fmts            show available audio sample formats   #显示可用的采样格式
-colors                 show available color names            #显示可用的颜色名称
-sources device         list sources of the input device      #显示可用的输入设备
-sinks device           list sinks of the output device       #显示可用的输出设备
-hwaccels               show available HW acceleration methods #显示可用的硬件加速方法

#全局选项
Global options(affect whole program instead of just one file):
-loglevel loglevel      set logging level
-v loglevel             set logging level
-report                 generate a report
-max_alloc Bytes        set maximum size of a single allocated block
-y                      overwrite output files
-n                      never overwrite output files
-ignore_unknown         Ignore unknown stream types
-filter_threads         number of non-complex filter threads
```

```
-filter_complex_threads number of threads for -filter_complex
-stats                    print progress report during encoding
-max_error_rate maximum error rate ratio of errors(0.0: no errors,1.0: 100% errors) above
which ffmpeg returns an error instead of success.
-bits_per_raw_sample number set the number of bits per raw sample
-vol volume            change audio volume(256=normal)

#文件主要选项
Per-file main options:
-f fmt                 force format        #强制规定文件格式
-c codec               codec name
-codec codec           codec name
-pre preset            preset name
-map_metadata outfile[,metadata]: infile[,metadata] set metadata information of outfile
from infile
-t duration            record or transcode "duration" seconds of audio/video
-to time_stop          record or transcode stop time
-fs limit_size         set the limit file size in Bytes
-ss time_off           set the start time offset
-sseof time_off        set the start time offset relative to EOF
-seek_timestamp        enable/disable seeking by timestamp with -ss
-timestamp time        set the recording timestamp('now' to set the current time)
-metadata string=string add metadata
-program title=string: st=number... add program with specified streams
-target type           specify target file type("vcd","svcd","dvd","dv" or "dv50" with
optional prefixes "pal-","ntsc-" or "film-")
-apad                  audio pad
-frames number         set the number of frames to output
-filter filter_graph   set stream filtergraph
-filter_script filename read stream filtergraph description from a file
-reinit_filter         reinit filtergraph on input parameter changes
-discard               discard
-disposition           disposition

#视频选项
Video options:
-vframes number        set the number of video frames to output
-r rate                set frame rate(Hz value,fraction or abbreviation)
-s size                set frame size(WxH or abbreviation)
-aspect aspect         set aspect ratio(4:3,16:9 or 1.3333,1.7777)
-bits_per_raw_sample number set the number of bits per raw sample
-vn                    disable video
-vcodec codec          force video codec('copy' to copy stream)
-timecode hh:mm:ss[:;.]ff set initial TimeCode value.
-pass n                select the pass number(1 to 3)
-vf filter_graph       set video filters
```

```
-ab bitrate             audio bitrate(please use -b: a)
-b bitrate              video bitrate(please use -b: v)
-dn                     disable data

#音频选项
Audio options:
-aframes number         set the number of audio frames to output
-aq quality             set audio quality(codec-specific)
-ar rate                set audio sampling rate(in Hz)
-ac channels            set number of audio channels
-an                     disable audio
-acodec codec           force audio codec('copy' to copy stream)
-vol volume             change audio volume(256 = normal)
-af filter_graph        set audio filters

#字幕选项
Subtitle options:
-s size                 set frame size(WxH or abbreviation)
-sn                     disable subtitle
-scodec codec           force subtitle codec('copy' to copy stream)
-stag fourcc/tag        force subtitle tag/fourcc
-fix_sub_duration       fix subtitles duration
-canvas_size size       set canvas size(WxH or abbreviation)
-spre preset            set the subtitle options to the indicated preset
```

3.11 FFmpeg 其他选项

FFmpeg 提供了非常多的编码器、解码器、封装器、协议、硬件加速等功能，这里重点介绍以下几项。

(1) 可用的比特流：ffmpeg -bsfs。

(2) 可用的编解码器：ffmpeg -codecs。

(3) 可用的解码器：ffmpeg -decoders。

(4) 可用的编码器：ffmpeg -encoders。

(5) 可用的滤镜：ffmpeg -filters。

(6) 可用的视频格式：ffmpeg -formats。

(7) 可用的声道布局：ffmpeg -layouts。

(8) 可用的许可证：ffmpeg -L。

(9) 可用的像素格式：ffmpeg -pix_fmts。

(10) 可用的协议：ffmpeg -protocols。

3.11.1 -formats：支持的文件格式

-formats 用于列举所有可用的文件格式，D 代表解封装支持的格式，E 代表封装支持的格式，DE 代表封装/解封装都支持的格式。由于内容太多，笔者删除了一些不常用的格式，输出信息如下：

```
//chapter3//ffmpeg - 311 - formats.txt
File formats:                                                    #文件格式
D. = Demuxing supported                                          #解封装支持
.E = Muxing supported                                            #封装支持
--
D 3dostr                3DO STR
  E 3g2                 3GP2(3GPP2 file format)
  E 3gp                 3GP(3GPP file format)
D aac                   raw ADTS AAC(Advanced Audio Coding)      #AAC格式
DE ac3                  raw AC - 3                               #AC - 3格式
D avs                   Argonaut Games Creature Shock
DE avs2                 raw AVS2 - P2/IEEE1857.4 video

D dshow                 DirectShow capture
D dsicin                Delphine Software International CIN
DE dv                   DV(Digital Video)          #DV格式,用于数字视频
D dvbsub                raw dvbsub
D dvbtxt                dvbtxt
  E dvd                 MPEG - 2 PS(DVD VOB)       #DVD格式,包括DVD和VOB,属于PS格式

DE f32be                PCM 32 - bit floating - point big - endian
DE f32le                PCM 32 - bit floating - point little - endian
  E f4v                 F4V Adobe Flash Video      #Flash格式,包含视频流信息
DE f64be                PCM 64 - bit floating - point big - endian
DE f64le                PCM 64 - bit floating - point little - endian
DE ffmetadata           FFmpeg metadata in text

DE flv                  FLV(Flash Video)           #FLV格式,包含视频、音频、字幕等

DE g722                 raw G.722                  #音频格式
DE g723_1               raw G.723.1                #音频格式
DE g726                 raw big - endian G.726("left - justified")
DE g726le               raw little - endian G.726("right - justified")
D g729                  G.729 raw format demuxer
D gdigrab               GDI API Windows frame grabber
DE gif                  CompuServe Graphics Interchange Format(GIF)
D gif_pipe              piped gif sequence
```

```
DE h261                       raw H.261                    # 视频格式
DE h263                       raw H.263                    # 视频格式
DE h264                       raw H.264 video              # 视频格式
DE hevc                       raw HEVC video               # 视频格式,即 H.265

DE image2                     image2 sequence
DE image2pipe                 piped image2 sequence

D jpeg_pipe                   piped jpeg sequence
D jpegls_pipe                 piped jpegls sequence

DE m4v                        raw MPEG-4 video             # 视频格式,包含 MPEG-4 视频流信息
  E matroska                  Matroska                     # 视频格式,MKV

DE mjpeg                      raw MJPEG video
D mjpeg_2000                  raw MJPEG 2000 video

  E mov                       QuickTime / MOV
D mov,mp4,m4a,3gp,3g2,mj2 QuickTime / MOV
  E mp2                       MP2(MPEG audio layer 2)      # 音频格式
DE mp3                        MP3(MPEG audio layer 3)      # 音频格式
  E mp4                       MP4(MPEG-4 Part 14)          # 视频格式,.mp4
DE mpeg                       MPEG-1 Systems / MPEG program stream
  E mpeg1video                raw MPEG-1 video
  E mpeg2video                raw MPEG-2 video
DE mpegts                     MPEG-TS(MPEG-2 Transport Stream)
D mpegtsraw                   raw MPEG-TS(MPEG-2 Transport Stream)
D mpegvideo                   raw MPEG video
DE mpjpeg                     MIME multipart JPEG
D mpl2                        MPL2 subtitles
D mv                          Silicon Graphics Movie

  E oga                       Ogg Audio
DE ogg                        Ogg
  E ogv                       Ogg Video
DE oma                        Sony OpenMG audio
  E opus                      Ogg Opus

DE rawvideo                   raw video
DE rtp                        RTP output
  E rtp_mpegts                RTP/mpegts output format
DE rtsp                       RTSP output
DE s16be                      PCM signed 16-bit big-endian
DE s16le                      PCM signed 16-bit little-endian
DE s24be                      PCM signed 24-bit big-endian
DE s24le                      PCM signed 24-bit little-endian
```

```
DE s32be                PCM signed 32 - bit big - endian
DE s32le                PCM signed 32 - bit little - endian

DE u16be                PCM unsigned 16 - bit big - endian
DE u16le                PCM unsigned 16 - bit little - endian
DE u24be                PCM unsigned 24 - bit big - endian
DE u24le                PCM unsigned 24 - bit little - endian
DE u32be                PCM unsigned 32 - bit big - endian
DE u32le                PCM unsigned 32 - bit little - endian
DE u8                   PCM unsigned 8 - bit
DE vc1                  raw VC - 1 video
DE vc1test              VC - 1 test bitstream
  E vcd                 MPEG - 1 Systems / MPEG program stream(VCD)
D vfwcap                VfW video capture
  E vob                 MPEG - 2 PS(VOB)
D vobsub                VobSub subtitle format

DE wav                  WAV / WAVE(Waveform Audio)
   E webm               WebM
  E webm_chunk          WebM Chunk Muxer
DE webm_dash_manifest WebM DASH Manifest
  E webp                WebP
D webp_pipe             piped webp sequence

DE yuv4mpegpipe         YUV4MPEG pipe
```

3.11.2 -muxers：支持的封装器格式

-muxers 用于列举所有可用的封装器格式，D 代表解封装支持的格式，E 代表封装支持的格式，DE 代表封装/解封装都支持的格式。-muxers 属于封装格式，所以只包含 E 类型的格式，具体输出信息如下：

```
//chapter3//ffmpeg - 311 - muxers.txt
File formats:
D. = Demuxing supported
.E = Muxing supported
 --
  E 3g2                 3GP2(3GPP2 file format)
  E 3gp                 3GP(3GPP file format)
  E a64                 a64 - video for Commodore 64
  E ac3                 raw AC - 3
  E adts                ADTS AAC(Advanced Audio Coding)
  E adx                 CRI ADX
  E aiff                Audio IFF
```

```
E alaw            PCM A - law
E amr             3GPP AMR
E apng            Animated Portable Network Graphics
E aptx            raw aptX(Audio Processing Technology for Bluetooth)
E aptx_hd         raw aptX HD(Audio Processing Technology for Bluetooth)
E asf             ASF(Advanced / Active Streaming Format)
E asf_stream      ASF(Advanced / Active Streaming Format)
E ass             SSA(SubStation Alpha) subtitle
E ast             AST(Audio Stream)
E au              Sun AU
E avi             AVI(Audio Video Interleaved)
E avm2            SWF(ShockWave Flash)(AVM2)
E avs2            raw AVS2 - P2/IEEE1857.4 video
E bit             G.729 BIT file format
E caf             Apple CAF(Core Audio Format)
E cavsvideo       raw Chinese AVS(Audio Video Standard) video
E codec2          codec2 .c2 muxer
E codec2raw       raw codec2 muxer
E crc             CRC testing
E dash            DASH Muxer
E data            raw data
E daud            D - Cinema audio
E dirac           raw Dirac
E dnxhd           raw DNxHD(SMPTE VC - 3)
E dts             raw DTS
E dv              DV(Digital Video)
E dvd             MPEG - 2 PS(DVD VOB)
E eac3            raw E - AC - 3
E f32be           PCM 32 - bit floating - point big - endian
E f32le           PCM 32 - bit floating - point little - endian
E f4v             F4V Adobe Flash Video
E f64be           PCM 64 - bit floating - point big - endian
E f64le           PCM 64 - bit floating - point little - endian
E ffmetadata      FFmpeg metadata in text
E fifo            FIFO queue pseudo - muxer
E fifo_test       Fifo test muxer
E film_cpk        Sega FILM / CPK
E filmstrip       Adobe Filmstrip
E fits            Flexible Image Transport System
E flac            raw FLAC
E flv             FLV(Flash Video)
E framecrc        framecrc testing
E framehash       Per - frame hash testing
E framemd5        Per - frame MD5 testing
E g722            raw G.722
E g723_1          raw G.723.1
```

```
E g726                  raw big-endian G.726("left-justified")
E g726le                raw little-endian G.726("right-justified")
E gif                   CompuServe Graphics Interchange Format(GIF)
E gsm                   raw GSM
E gxf                   GXF(General eXchange Format)
E h261                  raw H.261
E h263                  raw H.263
E h264                  raw H.264 video
E hash                  Hash testing
E hds                   HDS Muxer
E hevc                  raw HEVC video
E hls                   Apple HTTP Live Streaming
E ico                   Microsoft Windows ICO
E ilbc                  iLBC storage
E image2                image2 sequence
E image2pipe            piped image2 sequence
E ipod                  iPod H.264 MP4(MPEG-4 Part 14)
E ircam                 Berkeley/IRCAM/CARL Sound Format
E ismv                  ISMV/ISMA(Smooth Streaming)
E ivf                   On2 IVF
E jacosub               JACOsub subtitle format
E kvag                  Simon & Schuster Interactive VAG
E latm                  LOAS/LATM
E lrc                   LRC lyrics
E m4v                   raw MPEG-4 video
E matroska              Matroska
E md5                   MD5 testing
E microdvd              MicroDVD subtitle format
E mjpeg                 raw MJPEG video
E mkvtimestamp_v2 extract pts as timecode v2 format,as defined by mkvtoolnix
E mlp                   raw MLP
E mmf                   Yamaha SMAF
E mov                   QuickTime / MOV
E mp2                   MP2(MPEG audio layer 2)
E mp3                   MP3(MPEG audio layer 3)
E mp4                   MP4(MPEG-4 Part 14)
E mpeg                  MPEG-1 Systems / MPEG program stream
E mpeg1video            raw MPEG-1 video
E mpeg2video            raw MPEG-2 video
E mpegts                MPEG-TS(MPEG-2 Transport Stream)
E mpjpeg                MIME multipart JPEG
E mulaw                 PCM mu-law
E mxf                   MXF(Material eXchange Format)
E mxf_d10               MXF(Material eXchange Format) D-10 Mapping
E mxf_opatom            MXF(Material eXchange Format) Operational Pattern Atom
E null                  raw null video
E nut                   NUT
```

```
E oga                    Ogg Audio
E ogg                    Ogg
E ogv                    Ogg Video
E oma                    Sony OpenMG audio
E opus                   Ogg Opus
E psp                    PSP MP4(MPEG-4 Part 14)
E rawvideo               raw video
E rm                     RealMedia
E roq                    raw id RoQ
E rso                    Lego Mindstorms RSO
E rtp                    RTP output
E rtp_mpegts             RTP/mpegts output format
E rtsp                   RTSP output
E s16be                  PCM signed 16-bit big-endian
E s16le                  PCM signed 16-bit little-endian
E s24be                  PCM signed 24-bit big-endian
E s24le                  PCM signed 24-bit little-endian
E s32be                  PCM signed 32-bit big-endian
E s32le                  PCM signed 32-bit little-endian
E s8                     PCM signed 8-bit
E sap                    SAP output
E sbc                    raw SBC
E scc                    Scenarist Closed Captions
E sdl,sdl2               SDL2 output device
E segment                segment
E singlejpeg             JPEG single image
E smjpeg                 Loki SDL MJPEG
E smoothstreaming Smooth Streaming Muxer
E sox                    SoX native
E spdif                  IEC 61937(used on S/PDIF-IEC958)
E spx                    Ogg Speex
E srt                    SubRip subtitle
E stream_segment,ssegment streaming segment muxer
E streamhash             Per-stream hash testing
E sup                    raw HDMV Presentation Graphic Stream subtitles
E svcd                   MPEG-2 PS(SVCD)
E swf                    SWF(ShockWave Flash)
E tee                    Multiple muxer tee
E truehd                 raw TrueHD
E tta                    TTA(True Audio)
E u16be                  PCM unsigned 16-bit big-endian
E u16le                  PCM unsigned 16-bit little-endian
E u24be                  PCM unsigned 24-bit big-endian
E u24le                  PCM unsigned 24-bit little-endian
E u32be                  PCM unsigned 32-bit big-endian
E u32le                  PCM unsigned 32-bit little-endian
```

```
E u8                    PCM unsigned 8-bit
E uncodedframecrc uncoded framecrc testing
E vc1                   raw VC-1 video
E vc1test               VC-1 test bitstream
E vcd                   MPEG-1 Systems / MPEG program stream(VCD)
E vidc                  PCM Archimedes VIDC
E vob                   MPEG-2 PS(VOB)
E voc                   Creative Voice
E w64                   Sony Wave64
E wav                   WAV / WAVE(Waveform Audio)
E webm                  WebM
E webm_chunk            WebM Chunk Muxer
E webm_dash_manifest WebM DASH Manifest
E webp                  WebP
E webvtt                WebVTT subtitle
E wtv                   Windows Television(WTV)
E wv                    raw WavPack
E yuv4mpegpipe          YUV4MPEG pipe
```

3.11.3 -demuxers：支持的解封装器格式

-demuxers 用于列举所有可用的解封装器格式，D 代表解封装支持的格式，E 代表封装支持的格式，DE 代表封装/解封装都支持的格式。-demuxers 属于解封装格式，所以只包含 D 类型的格式。由于内容太多，笔者删除了一些不常用的格式，具体输出信息如下：

```
//chapter3//ffmpeg-311-demuxers.txt
File formats:
D. = Demuxing supported
.E = Muxing supported
--
D 3dostr                3DO STR
D 4xm                   4X Technologies
D aa                    Audible AA format files
D aac                   raw ADTS AAC(Advanced Audio Coding)
D ac3                   raw AC-3
D acm                   Interplay ACM
D act                   ACT Voice file format
D adf                   Artworx Data Format
D adp                   ADP
...
D dv                    DV(Digital Video)
D dvbsub                raw dvbsub
D dvbtxt                dvbtxt
D dxa                   DXA
```

```
D ea                          Electronic Arts Multimedia
D ea_cdata                    Electronic Arts cdata
D eac3                        raw E-AC-3
D epaf                        Ensoniq Paris Audio File
D exr_pipe                    piped exr sequence
D f32be                       PCM 32-bit floating-point big-endian
D f32le                       PCM 32-bit floating-point little-endian
D f64be                       PCM 64-bit floating-point big-endian
D f64le                       PCM 64-bit floating-point little-endian
D ffmetadata                  FFmpeg metadata in text
D film_cpk                    Sega FILM / CPK
D filmstrip                   Adobe Filmstrip
D fits                        Flexible Image Transport System
D flac                        raw FLAC
D flic                        FLI/FLC/FLX animation
D flv                         FLV(Flash Video)
D frm                         Megalux Frame
D fsb                         FMOD Sample Bank
...
D h261                        raw H.261
D h263                        raw H.263
D h264                        raw H.264 video
D hca                         CRI HCA
D hcom                        Macintosh HCOM
D hevc                        raw HEVC video
...
D j2k_pipe                    piped j2k sequence
D jacosub                     JACOsub subtitle format
D jpeg_pipe                   piped jpeg sequence
D jpegls_pipe                 piped jpegls sequence
...
D m4v                         raw MPEG-4 video
D matroska,webm               Matroska / WebM
D mgsts                       Metal Gear Solid: The Twin Snakes
D microdvd                    MicroDVD subtitle format
D mjpeg                       raw MJPEG video
D mjpeg_2000                  raw MJPEG 2000 video
...
D mov,mp4,m4a,3gp,3g2,mj2 QuickTime / MOV
D mp3                         MP2/3(MPEG audio layer 2/3)
D mpc                         Musepack
D mpc8                        Musepack SV8
D mpeg                        MPEG-PS(MPEG-2 Program Stream)
D mpegts                      MPEG-TS(MPEG-2 Transport Stream)
D mpegtsraw                   raw MPEG-TS(MPEG-2 Transport Stream)
D mpegvideo                   raw MPEG video
```

```
D mpjpeg                 MIME multipart JPEG
...
D mv                     Silicon Graphics Movie
D mvi                    Motion Pixels MVI
D mxf                    MXF(Material eXchange Format)
...
D ogg                    Ogg
...
D rtp                    RTP input
D rtsp                   RTSP input
D s16be                  PCM signed 16 - bit big - endian
D s16le                  PCM signed 16 - bit little - endian
D s24be                  PCM signed 24 - bit big - endian
D s24le                  PCM signed 24 - bit little - endian
D s32be                  PCM signed 32 - bit big - endian
D s32le                  PCM signed 32 - bit little - endian
D s337m                  SMPTE 337M
...
D u16be                  PCM unsigned 16 - bit big - endian
D u16le                  PCM unsigned 16 - bit little - endian
D u24be                  PCM unsigned 24 - bit big - endian
D u24le                  PCM unsigned 24 - bit little - endian
D u32be                  PCM unsigned 32 - bit big - endian
D u32le                  PCM unsigned 32 - bit little - endian
D u8                     PCM unsigned 8 - bit
...
D wav                    WAV / WAVE(Waveform Audio)
D wc3movie               Wing Commander III movie
D webm_dash_manifest WebM DASH Manifest
D webp_pipe              piped webp sequence
...
D yuv4mpegpipe           YUV4MPEG pipe
```

3.11.4 -devices：支持的设备

-devices 用于列举所有可用的设备，D 代表解封装支持的格式，E 代表封装支持的格式，DE 代表封装/解封装都支持的格式。具体输出信息如下：

```
//chapter3//ffmpeg - 311 - devices.txt
Devices:
D. = Demuxing supported
.E = Muxing supported
--
D dshow                  DirectShow capture #DirectShow 类型的捕获设备
```

```
D gdigrab              GDI API Windows frame grabber      #Windows 下的帧抓取器,可以录屏
D lavfi                Libavfilter virtual input device   #虚拟输入设备
  E sdl,sdl2           SDL2 output device                 #SDL2 的输出设备
D vfwcap               VfW video capture                  #Windows 下的 vfwcap,用于视频捕获
```

3.11.5 -encoders：支持的编码器格式

-encoders 用于列举所有可用的编码器格式，V 代表视频格式，A 代表音频格式，S 代表字幕格式，F 代表帧级别的多线程运算，S 代表条带级别的多线程运算，X 代表实验的编解码器，B 代表支持绘制水平带，D 代表直接渲染。由于内容太多，笔者删除了一些不常用的格式，具体输出信息如下：

```
//chapter3//ffmpeg-311-encoders.txt
Encoders:
V ..... = Video
A ..... = Audio
S ..... = Subtitle
.F .... = Frame-level multithreading
..S ... = Slice-level multithreading
.....X = Codec is experimental
....B. = Supports draw_horiz_band
.....D = Supports direct rendering method 1
------
...
V..... ayuv              Uncompressed packed MS 4:4:4:4      #YUV444 采样格式
V..... bmp               BMP(Windows and OS/2 bitmap)        #BMP 格式
...
V..... flashsv           Flash Screen Video
V..... flashsv2          Flash Screen Video Version 2
V..... flv               FLV / Sorenson Spark / Sorenson H.263(Flash Video)(codec flv1)
                                                             #FLV 编码
V..... gif               GIF(Graphics Interchange Format)
V..... h261              H.261                               #H.261 视频编码
V..... h263              H.263 / H.263-1996                  #H.263 视频编码
V.S... h263p             H.263+ / H.263-1998 / H.263 version 2
V..... libx264           libx264 H.264 / AVC / MPEG-4 AVC / MPEG-4 part 10(codec h264)
                                                   #libx264 编码,属于 H.264 的开源编码器
V..... libx264rgb        libx264 H.264 / AVC / MPEG-4 AVC / MPEG-4 part 10 RGB(codec h264)
V..... h264_amf          AMD AMF H.264 Encoder(codec h264)
V..... h264_mf           H264 via MediaFoundation(codec h264)
V..... h264_nvenc        NVIDIA NVENC H.264 encoder(codec h264)
V..... h264_qsv          H.264 / AVC / MPEG-4 AVC / MPEG-4 part 10(Intel Quick Sync Video
acceleration)(codec h264)
V..... nvenc             NVIDIA NVENC H.264 encoder(codec h264)
```

```
V..... nvenc_h264           NVIDIA NVENC H.264 encoder(codec h264)
V..... hap                  Vidvox Hap
V..... libx265              libx265 H.265 / HEVC(codec hevc) # libx265 编码
V..... nvenc_hevc           NVIDIA NVENC hevc encoder(codec hevc)
V..... hevc_amf             AMD AMF HEVC encoder(codec hevc)
V..... hevc_mf              HEVC via MediaFoundation(codec hevc)
V..... hevc_nvenc           NVIDIA NVENC hevc encoder(codec hevc)
V..... hevc_qsv             HEVC(Intel Quick Sync Video acceleration)(codec hevc)
VF.... huffyuv              Huffyuv / HuffYUV
V..... jpeg2000             JPEG 2000
VF.... libopenjpeg          OpenJPEG JPEG 2000(codec jpeg2000)
VF.... jpegls               JPEG-LS
VF.... ljpeg                Lossless JPEG
VF.... magicyuv             MagicYUV video
VFS... mjpeg                MJPEG(Motion JPEG)
V..... mjpeg_qsv            MJPEG(Intel Quick Sync Video acceleration)(codec mjpeg)
V.S... mpeg1video           MPEG-1 video
V.S... mpeg2video           MPEG-2 video
V..... mpeg2_qsv            MPEG-2 video(Intel Quick Sync Video acceleration)(codec mpeg2video)
V.S... mpeg4                MPEG-4 part 2
V..... libxvid              libxvidcore MPEG-4 part 2(codec mpeg4)
V..... msmpeg4v2            MPEG-4 part 2 Microsoft variant version 2
V..... msmpeg4              MPEG-4 part 2 Microsoft variant version 3(codec msmpeg4v3)
V..... msvideo1             Microsoft Video-1
...
V..... v210                 Uncompressed 4:2:2 10-bit
V..... v308                 Uncompressed packed 4:4:4
V..... v408                 Uncompressed packed QT 4:4:4:4
V..... v410                 Uncompressed 4:4:4 10-bit
V..... libvpx               libvpx VP8(codec vp8)
V..... libvpx-vp9           libvpx VP9(codec vp9)
V..... vp9_qsv              VP9 video(Intel Quick Sync Video acceleration)(codec vp9)
V..... libwebp_anim         libwebp WebP image(codec webp)
V..... libwebp              libwebp WebP image(codec webp)
V..... wmv1                 Windows Media Video 7
V..... wmv2                 Windows Media Video 8

V..... y41p                 Uncompressed YUV 4:1:1 12-bit
V..... yuv4                 Uncompressed packed 4:2:0
VF.... zlib                 LCL(LossLess Codec Library) ZLIB
V..... zmbv                 Zip Motion Blocks Video
A..... aac                  AAC(Advanced Audio Coding)
A..... aac_mf               AAC via MediaFoundation(codec aac)
A..... ac3                  ATSC A/52A(AC-3)
A..... ac3_fixed            ATSC A/52A(AC-3)(codec ac3)
```

```
A..... ac3_mf              AC3 via MediaFoundation(codec ac3)
A..... adpcm_adx           SEGA CRI ADX ADPCM
A..... g722                G.722 ADPCM(codec adpcm_g722)
A..... g726                G.726 ADPCM(codec adpcm_g726)
A..... g726le              G.726 little endian ADPCM("right - justified")(codec adpcm_g726le)
A..... adpcm_ima_qt        ADPCM IMA QuickTime
A..... adpcm_ima_ssi       ADPCM IMA Simon & Schuster Interactive
A..... adpcm_ima_wav       ADPCM IMA WAV
A..... adpcm_ms            ADPCM Microsoft
A..... adpcm_swf           ADPCM Shockwave Flash
A..... adpcm_yamaha        ADPCM Yamaha

A..... flac                FLAC(Free Lossless Audio Codec)
A..... g723_1              G.723.1

A..... mp2                 MP2(MPEG audio layer 2)
A..... mp2fixed            MP2 fixed point(MPEG audio layer 2)(codec mp2)
A..... libtwolame          libtwolame MP2(MPEG audio layer 2)(codec mp2)
A..... mp3_mf              MP3 via MediaFoundation(codec mp3)
A..... libmp3lame          libmp3lame MP3(MPEG audio layer 3)(codec mp3)

A..... libopus             libopus Opus(codec opus)
A..... pcm_alaw            PCM A - law / G.711 A - law
A..... pcm_dvd             PCM signed 16|20|24 - bit big - endian for DVD media
A..... pcm_f32be           PCM 32 - bit floating point big - endian
A..... pcm_f32le           PCM 32 - bit floating point little - endian
A..... pcm_f64be           PCM 64 - bit floating point big - endian
A..... pcm_f64le           PCM 64 - bit floating point little - endian
A..... pcm_mulaw           PCM mu - law / G.711 mu - law
A..... pcm_s16be           PCM signed 16 - bit big - endian
A..... pcm_s16be_planar    PCM signed 16 - bit big - endian planar
A..... pcm_s16le           PCM signed 16 - bit little - endian
A..... pcm_s16le_planar    PCM signed 16 - bit little - endian planar
A..... pcm_s24be           PCM signed 24 - bit big - endian
A..... pcm_s24daud         PCM D - Cinema audio signed 24 - bit
A..... pcm_s24le           PCM signed 24 - bit little - endian
A..... pcm_s24le_planar    PCM signed 24 - bit little - endian planar
A..... pcm_s32be           PCM signed 32 - bit big - endian
A..... pcm_s32le           PCM signed 32 - bit little - endian
A..... pcm_s32le_planar    PCM signed 32 - bit little - endian planar
A..... pcm_s64be           PCM signed 64 - bit big - endian
A..... pcm_s64le           PCM signed 64 - bit little - endian
A..... pcm_s8              PCM signed 8 - bit
A..... pcm_s8_planar       PCM signed 8 - bit planar
A..... pcm_u16be           PCM unsigned 16 - bit big - endian
A..... pcm_u16le           PCM unsigned 16 - bit little - endian
```

```
A..... pcm_u24be            PCM unsigned 24 - bit big - endian
A..... pcm_u24le            PCM unsigned 24 - bit little - endian
A..... pcm_u32be            PCM unsigned 32 - bit big - endian
A..... pcm_u32le            PCM unsigned 32 - bit little - endian
A..... pcm_u8               PCM unsigned 8 - bit
A..... pcm_vidc             PCM Archimedes VIDC
A..... real_144             RealAudio 1.0(14.4K)(codec ra_144)
A..... roq_dpcm             id RoQ DPCM

A..... wmav1                Windows Media Audio 1
A..... wmav2                Windows Media Audio 2
S..... ssa                  ASS(Advanced SubStation Alpha) subtitle(codec ass)
S..... ass                  ASS(Advanced SubStation Alpha) subtitle
S..... dvbsub               DVB subtitles(codec dvb_subtitle)
S..... dvdsub               DVD subtitles(codec dvd_subtitle)
S..... mov_text             3GPP Timed Text subtitle
S..... srt                  SubRip subtitle(codec subrip)
S..... subrip               SubRip subtitle
S..... text                 Raw text subtitle
S..... webvtt               WebVTT subtitle
S..... xsub                 DivX subtitles(XSUB)
```

3.11.6 -decoders：支持的解码器格式

-decoders 用于列举所有可用的解码器格式，V 代表视频格式，A 代表音频格式，S 代表字幕格式，F 代表帧级别的多线程运算，S 代表条带级别的多线程运算，X 代表实验的编解码器，B 代表支持绘制水平带，D 代表直接渲染。由于内容太多，笔者删除了一些不常用的格式，具体输出信息如下：

```
//chapter3//ffmpeg - 311 - decoders.txt
Decoders:
V..... = Video
A..... = Audio
S..... = Subtitle
.F.... = Frame - level multithreading
..S... = Slice - level multithreading
...X.. = Codec is experimental
....B. = Supports draw_horiz_band
.....D = Supports direct rendering method 1
------
V....D 012v                 Uncompressed 4:2:2 10 - bit
  V....D 8bps               QuickTime 8BPS video

V....D avs                  AVS(Audio Video Standard) video
```

```
V....D avui                 Avid Meridien Uncompressed
V....D ayuv                 Uncompressed packed MS 4:4:4:4
V....D cyuv                 Creative YUV(CYUV)
V.S..D dds                  DirectDraw Surface image decoder
  VFS..D ffv1               FFmpeg video codec #1
  V...BD flv                FLV / Sorenson Spark / Sorenson H.263(Flash Video)(codec flv1)

V....D gif                  GIF(Graphics Interchange Format)
V....D h261                 H.261
V...BD h263                 H.263 / H.263-1996,H.263+ / H.263-1998 / H.263 version 2
V...BD h263i                Intel H.263
V...BD h263p                H.263 / H.263-1996,H.263+ / H.263-1998 / H.263 version 2
VFS..D h264                 H.264 / AVC / MPEG-4 AVC / MPEG-4 part 10
V....D h264_qsv             H.264 / AVC / MPEG-4 AVC / MPEG-4 part 10(Intel Quick Sync Video
acceleration)(codec h264)
V..... h264_cuvid           NVIDIA CUVID H264 decoder(codec h264)
VFS..D hap                  Vidvox Hap
VFS..D hevc                 HEVC(High Efficiency Video Coding)
V....D hevc_qsv             HEVC(Intel Quick Sync Video acceleration)(codec hevc)
V..... hevc_cuvid           NVIDIA CUVID HEVC decoder(codec hevc)

VFS..D jpeg2000             JPEG 2000
VF...D libopenjpeg          OpenJPEG JPEG 2000(codec jpeg2000)

V....D mjpeg                MJPEG(Motion JPEG)
V..... mjpeg_cuvid          NVIDIA CUVID MJPEG decoder(codec mjpeg)
V....D mjpeg_qsv            MJPEG video(Intel Quick Sync Video acceleration)(codec mjpeg)
V....D mjpegb               Apple MJPEG-B
V....D mmvideo              American Laser Games MM Video
V....D motionpixels         Motion Pixels video
V.S.BD mpeg1video           MPEG-1 video
V..... mpeg1_cuvid          NVIDIA CUVID MPEG1VIDEO decoder(codec mpeg1video)
V.S.BD mpeg2video           MPEG-2 video
V.S.BD mpegvideo            MPEG-1 video(codec mpeg2video)
V....D mpeg2_qsv            MPEG-2 video(Intel Quick Sync Video acceleration)(codec mpeg2video)
V..... mpeg2_cuvid          NVIDIA CUVID MPEG2VIDEO decoder(codec mpeg2video)
VF..BD mpeg4                MPEG-4 part 2
V..... mpeg4_cuvid          NVIDIA CUVID MPEG4 decoder(codec mpeg4)

V....D vc1                  SMPTE VC-1
V....D vc1_qsv              VC-1 video(Intel Quick Sync Video acceleration)(codec vc1)
VF..BD vp3                  On2 VP3
VF..BD vp4                  On2 VP4
V....D vp5                  On2 VP5
V....D vp6                  On2 VP6
V.S..D vp6a                 On2 VP6(Flash version,with alpha channel)
```

```
V....D vp6f                 On2 VP6(Flash version)
V....D vp7                  On2 VP7
VFS..D vp8                  On2 VP8
V....D libvpx               libvpx VP8(codec vp8)
V..... vp8_cuvid            NVIDIA CUVID VP8 decoder(codec vp8)
V....D vp8_qsv              VP8 video(Intel Quick Sync Video acceleration)(codec vp8)
VFS..D vp9                  Google VP9
V..... libvpx-vp9           libvpx VP9(codec vp9)
V..... vp9_cuvid            NVIDIA CUVID VP9 decoder(codec vp9)
V....D vp9_qsv              VP9 video(Intel Quick Sync Video acceleration)(codec vp9)

VF...D webp                 WebP image
V...BD wmv1                 Windows Media Video 7
V...BD wmv2                 Windows Media Video 8
V....D wmv3                 Windows Media Video 9
V....D wmv3image            Windows Media Video 9 Image

V....D yuv4                 Uncompressed packed 4:2:0
VF...D zlib                 LCL(LossLess Codec Library) ZLIB

A....D aac                  AAC(Advanced Audio Coding)
A....D aac_fixed            AAC(Advanced Audio Coding)(codec aac)
A....D aac_latm             AAC LATM(Advanced Audio Coding LATM syntax)
A....D ac3                  ATSC A/52A(AC-3)
A....D ac3_fixed            ATSC A/52A(AC-3)(codec ac3)
  A....D adpcm_4xm          ADPCM 4X Movie
A....D adpcm_adx            SEGA CRI ADX ADPCM
A....D adpcm_afc            ADPCM Nintendo Gamecube AFC
A....D adpcm_agm            ADPCM AmuseGraphics Movie
A....D adpcm_aica           ADPCM Yamaha AICA
A....D adpcm_argo           ADPCM Argonaut Games
A....D adpcm_ct             ADPCM Creative Technology
A....D adpcm_dtk            ADPCM Nintendo Gamecube DTK
A....D adpcm_ea             ADPCM Electronic Arts
A....D adpcm_ea_maxis_xa    ADPCM Electronic Arts Maxis CDROM XA
A....D adpcm_ea_r1          ADPCM Electronic Arts R1
A....D adpcm_ea_r2          ADPCM Electronic Arts R2
A....D adpcm_ea_r3          ADPCM Electronic Arts R3
A....D adpcm_ea_xas         ADPCM Electronic Arts XAS
A....D g722                 G.722 ADPCM(codec adpcm_g722)
A....D g726                 G.726 ADPCM(codec adpcm_g726)
A....D g726le               G.726 ADPCM little-endian(codec adpcm_g726le)
A....D adpcm_ima_alp        ADPCM IMA High Voltage Software ALP
A....D adpcm_ima_amv        ADPCM IMA AMV
A....D adpcm_ima_apc        ADPCM IMA CRYO APC
A....D adpcm_ima_apm        ADPCM IMA Ubisoft APM
```

```
A....D adpcm_ima_cunning      ADPCM IMA Cunning Developments
A....D adpcm_ima_dat4         ADPCM IMA Eurocom DAT4
A....D adpcm_ima_dk3          ADPCM IMA Duck DK3
A....D adpcm_ima_dk4          ADPCM IMA Duck DK4
A....D adpcm_ima_ea_eacs      ADPCM IMA Electronic Arts EACS
A....D adpcm_ima_ea_sead      ADPCM IMA Electronic Arts SEAD
A....D adpcm_ima_iss          ADPCM IMA Funcom ISS
A....D adpcm_ima_mtf          ADPCM IMA Capcom's MT Framework
A....D adpcm_ima_oki          ADPCM IMA Dialogic OKI
A....D adpcm_ima_qt           ADPCM IMA QuickTime
A....D adpcm_ima_rad          ADPCM IMA Radical
A....D adpcm_ima_smjpeg       ADPCM IMA Loki SDL MJPEG
A....D adpcm_ima_ssi          ADPCM IMA Simon & Schuster Interactive
A....D adpcm_ima_wav          ADPCM IMA WAV
A....D adpcm_ima_ws           ADPCM IMA Westwood
A....D adpcm_ms               ADPCM Microsoft
A....D adpcm_mtaf             ADPCM MTAF
A....D adpcm_psx              ADPCM Playstation
A....D adpcm_sbpro_2          ADPCM Sound Blaster Pro 2-bit
A....D adpcm_sbpro_3          ADPCM Sound Blaster Pro 2.6-bit
A....D adpcm_sbpro_4          ADPCM Sound Blaster Pro 4-bit
A....D adpcm_swf              ADPCM Shockwave Flash
A....D adpcm_thp              ADPCM Nintendo THP
A....D adpcm_thp_le           ADPCM Nintendo THP(little-endian)
A....D adpcm_vima             LucasArts VIMA audio
A....D adpcm_xa               ADPCM CDROM XA
A....D adpcm_yamaha           ADPCM Yamaha
A....D adpcm_zork             ADPCM Zork

A....D mp2                    MP2(MPEG audio layer 2)
A....D mp2float               MP2(MPEG audio layer 2)(codec mp2)
A....D mp3float               MP3(MPEG audio layer 3)(codec mp3)
A....D mp3                    MP3(MPEG audio layer 3)
A....D mp3adufloat            ADU(Application Data Unit) MP3(MPEG audio layer 3)(codec mp3adu)
A....D mp3adu                 ADU(Application Data Unit) MP3(MPEG audio layer 3)
A....D mp3on4float            MP3onMP4(codec mp3on4)
A....D mp3on4                 MP3onMP4
A....D als                    MPEG-4 Audio Lossless Coding(ALS)(codec mp4als)
A....D mpc7                   Musepack SV7(codec musepack7)
A....D mpc8                   Musepack SV8(codec musepack8)
A....D nellymoser             Nellymoser Asao
A....D opus                   Opus
A....D libopus                libopus Opus(codec opus)
A....D paf_audio              Amazing Studio Packed Animation File Audio
A....D pcm_alaw               PCM A-law / G.711 A-law
A....D pcm_bluray             PCM signed 16|20|24-bit big-endian for Blu-ray media
```

```
A....D pcm_dvd              PCM signed 16|20|24 - bit big - endian for DVD media
A....D pcm_f16le            PCM 16.8 floating point little - endian
A....D pcm_f24le            PCM 24.0 floating point little - endian
A....D pcm_f32be            PCM 32 - bit floating point big - endian
A....D pcm_f32le            PCM 32 - bit floating point little - endian
A....D pcm_f64be            PCM 64 - bit floating point big - endian
A....D pcm_f64le            PCM 64 - bit floating point little - endian
A....D pcm_lxf              PCM signed 20 - bit little - endian planar
A....D pcm_mulaw            PCM mu - law / G.711 mu - law
A....D pcm_s16be            PCM signed 16 - bit big - endian
A....D pcm_s16be_planar     PCM signed 16 - bit big - endian planar
A....D pcm_s16le            PCM signed 16 - bit little - endian
A....D pcm_s16le_planar     PCM signed 16 - bit little - endian planar
A....D pcm_s24be            PCM signed 24 - bit big - endian
A....D pcm_s24daud          PCM D - Cinema audio signed 24 - bit
A....D pcm_s24le            PCM signed 24 - bit little - endian
A....D pcm_s24le_planar     PCM signed 24 - bit little - endian planar
A....D pcm_s32be            PCM signed 32 - bit big - endian
A....D pcm_s32le            PCM signed 32 - bit little - endian
A....D pcm_s32le_planar     PCM signed 32 - bit little - endian planar
A....D pcm_s64be            PCM signed 64 - bit big - endian
A....D pcm_s64le            PCM signed 64 - bit little - endian
A....D pcm_s8               PCM signed 8 - bit
A....D pcm_s8_planar        PCM signed 8 - bit planar
A....D pcm_u16be            PCM unsigned 16 - bit big - endian
A....D pcm_u16le            PCM unsigned 16 - bit little - endian
A....D pcm_u24be            PCM unsigned 24 - bit big - endian
A....D pcm_u24le            PCM unsigned 24 - bit little - endian
A....D pcm_u32be            PCM unsigned 32 - bit big - endian
A....D pcm_u32le            PCM unsigned 32 - bit little - endian
A....D pcm_u8               PCM unsigned 8 - bit

S..... mov_text             3GPP Timed Text subtitle

S..... text                 Raw text subtitle
S..... vplayer              VPlayer subtitle
S..... webvtt               WebVTT subtitle
S..... xsub                 XSUB
```

3.11.7 -protocols：支持的协议格式

-protocols 用于列举所有可用的协议格式，分为输入协议和输出协议，具体输出信息如下（#以开头的部分是笔者的注释）：

```
//chapter3/ffmpeg-311--protocols.txt
Supported file protocols:
Input:           #输入类型的协议
  async
  bluray
  cache
  concat
  crypto
  data
  ffrtmpcrypt
  ffrtmphttp
  file           #本地文件
  ftp            #FTP 文件传输协议
  gopher
  hls            #HLS(HTTP Live Streaming)直播协议
  http           #http,
  httpproxy
  https
  mmsh
  mmst
  pipe
  rtmp           #RTMP,直播推流协议
  rtmpe
  rtmps
  rtmpt
  rtmpte
  rtmpts
  rtp            #RTP,实时流协议,用于音视频数据的传输
  srtp
  subfile
  tcp
  tls
  udp
  udplite
  Srt

Output:          #输出类型的协议
  crypto
  ffrtmpcrypt
  ffrtmphttp
  file
  ftp
  gopher
  http
  httpproxy
  https
```

```
icecast
md5
pipe
prompeg
rtmp
rtmpe
rtmps
rtmpt
rtmpte
rtmpts
rtp
srtp
tee
tcp
tls
udp
udplite
srt
```

3.11.8 -hwaccels：支持的硬件加速格式

-hwaccels 用于列举所有可用的硬件加速格式，具体输出信息如下：

```
//chapter3/ffmpeg-311 -- hwaccels.txt
Hardware acceleration methods:
CUDA     #CUDA(Compute Unified Device Architecture)是显卡厂商 NVIDIA 推出的运算平台
dxva2    #dxva2: DXVA 是 DirectX Video Acceleration 的简称，中文译为视频硬件加速。DXVA 是微
         #软公司专门制定的视频加速规范，它共有两个版本，分别是 DXVA 1.0 和 DXVA 2.0
qsv      #QSV: Intel 的硬件加速方案
d3d11va  #d3d11va 硬件加速
```

3.11.9 -layouts：支持的声道模式

-layouts 用于列举所有可用的声道模式，具体输出信息如下：

```
//chapter3/ffmpeg-311 -- layouts.txt
Individual channels:                #独立声道
NAME            DESCRIPTION
FL              front left          #左前
FR              front right         #右前
FC              front center        #中前
LFE             low frequency       #低频
BL              back left           #左后
BR              back right          #右后
```

```
FLC                 front left - of - center      #左中前
FRC                 front right - of - center     #右中前
BC                  back center                   #中后
SL                  side left                     #左侧
SR                  side right                    #右侧
TC                  top center                    #中上
TFL                 top front left
TFC                 top front center
TFR                 top front right
TBL                 top back left
TBC                 top back center
TBR                 top back right
DL                  downmix left
DR                  downmix right
WL                  wide left
WR                  wide right
SDL                 surround direct left
SDR                 surround direct right
LFE2                low frequency 2

Standard channel layouts:                         #标准声道模式
NAME                DECOMPOSITION
mono                FC                            #单声道
stereo              FL + FR                       #双声道：左前、右前
2.1                 FL + FR + LFE                 #2.1 声道
3.0                 FL + FR + FC
3.0(back)           FL + FR + BC
4.0                 FL + FR + FC + BC
quad                FL + FR + BL + BR
quad(side)          FL + FR + SL + SR
3.1                 FL + FR + FC + LFE
5.0                 FL + FR + FC + BL + BR
5.0(side)           FL + FR + FC + SL + SR
4.1                 FL + FR + FC + LFE + BC
5.1                 FL + FR + FC + LFE + BL + BR
5.1(side)           FL + FR + FC + LFE + SL + SR
6.0                 FL + FR + FC + BC + SL + SR
6.0(front)          FL + FR + FLC + FRC + SL + SR
hexagonal           FL + FR + FC + BL + BR + BC
6.1                 FL + FR + FC + LFE + BC + SL + SR
6.1(back)           FL + FR + FC + LFE + BL + BR + BC
6.1(front)          FL + FR + LFE + FLC + FRC + SL + SR
7.0                 FL + FR + FC + BL + BR + SL + SR
7.0(front)          FL + FR + FC + FLC + FRC + SL + SR
7.1                 FL + FR + FC + LFE + BL + BR + SL + SR
7.1(wide)           FL + FR + FC + LFE + BL + BR + FLC + FRC
```

```
7.1(wide-side)  FL+FR+FC+LFE+FLC+FRC+SL+SR
octagonal       FL+FR+FC+BL+BR+BC+SL+SR
hexadecagonal   FL+FR+FC+BL+BR+BC+SL+SR+TFL+TFC+TFR+TBL+TBC+TBR+WL+WR
downmix         DL+DR
```

3.11.10 -sample_fmts：支持的采样格式

-sample_fmts 用于列举所有可用的采样格式，具体输出信息如下：

```
//chapter3/ffmpeg-311--sample_fmts.txt
name depth
u8      8    #无符号 8 位
s16     16   #有符号 16 位
s32     32   #有符号 32 位
flt     32   #float: 32 位
dbl     64   #double: 64 位

##p: 代表 planar,平面模式,例如 S16 和 S16P 的不同点在于数据的排列方式,前者相邻连续
##排列,后者则分离排列,但是现在有相当多的音频文件采用 planar 方案,如 S16、U8、S32、F32、F64
##都有对应的 planar 方式
u8p     8    #无符号 8 位: planar,平面模式
s16p    16
s32p    32
fltp    32
dblp    64
s64     64
s64p    64
```

3.11.11 -colors：支持的颜色名称

-colors 用于列举所有可用的颜色名称，具体输出信息如下：

```
//chapter3/ffmpeg-311--colors.txt
name                      #RRGGBB
AliceBlue                 #f0f8ff   #爱丽丝蓝
AntiqueWhite              #faebd7   #古董白
Aqua                      #00ffff
...
Black                     #000000
BlanchedAlmond            #ffebcd
Blue                      #0000ff   #蓝色
BlueViolet                #8a2be2
Brown                     #a52a2a   #棕色
```

```
BurlyWood            #deb887
...
DarkGoldenRod        #b8860b
DarkGray             #a9a9a9   #深灰色
DarkGreen            #006400
DarkKhaki            #bdb76b
DarkMagenta          #8b008b
DarkOliveGreen       #556b2f
Darkorange           #ff8c00
DarkOrchid           #9932cc
DarkRed              #8b0000   #深红色
DarkSalmon           #e9967a
DarkTurquoise        #00ced1
DarkViolet           #9400d3   #深紫色
DeepPink             #ff1493
Gold                 #ffd700
GoldenRod            #daa520
Gray                 #808080   #灰色
Green                #008000   #绿色
GreenYellow          #adff2f
HoneyDew             #f0fff0
...
Violet               #ee82ee  #紫色
Wheat                #f5deb3  #小麦色
White                #ffffff  #白色
WhiteSmoke           #f5f5f5
Yellow               #ffff00  #黄色
YellowGreen          #9acd32
```

3.11.12 -pix_fmts：支持的像素格式

-pix_fmts 用于列举所有可用的像素格式，I 代表支持输入，O 代表支持输出，P 代表支持调色板(Paletted)，H 代表支持硬件加速，B 代表支持位流，具体输出信息如下：

```
//chapter3/ffmpeg-311--pix_fmts.txt
Pixel formats:
I.... = Supported Input format for conversion
.O... = Supported Output format for conversion
..H.. = Hardware accelerated format
...P. = Paletted format
....B = Bitstream format
FLAGS NAME          NB_COMPONENTS BITS_PER_PIXEL
#名称                    单元数       每像素的位数
-----
IO... yuv420p            3            12    #每像素占12位,Y、U、V分别存储
```

```
IO... yuyv422        3     16   # 每像素占 16 位
IO... rgb24          3     24   # 每像素占 24 位
IO... bgr24          3     24
IO... yuv422p        3     16
IO... yuv444p        3     24
IO... yuv410p        3     9
IO... yuv411p        3     12
IO... gray           1     8
IO..B monow          1     1
IO..B monob          1     1
I..P. pal8           1     8
IO... yuvj420p       3     12
IO... yuvj422p       3     16
IO... yuvj444p       3     24
IO... uyvy422        3     16
..... uyyvyy411      3     12
IO... bgr8           3     8
.O..B bgr4           3     4
IO... bgr4_Byte      3     4
IO... rgb8           3     8
.O..B rgb4           3     4
IO... rgb4_Byte      3     4
IO... nv12           3     12
IO... nv21           3     12
IO... argb           4     32
IO... rgba           4     32
IO... abgr           4     32
IO... bgra           4     32
IO... gray16be       1     16
IO... gray16le       1     16
IO... yuv440p        3     16
IO... yuvj440p       3     16
IO... yuva420p       4     20
IO... rgb48be        3     48
IO... rgb48le        3     48
IO... rgb565be       3     16
IO... rgb565le       3     16
IO... rgb555be       3     15
IO... rgb555le       3     15
IO... bgr565be       3     16
IO... bgr565le       3     16
IO... bgr555be       3     15
IO... bgr555le       3     15
..H.. vaapi_moco     0     0
..H.. vaapi_idct     0     0
..H.. vaapi_vld      0     0
```

```
IO... yuv420p16le           3           24
IO... yuv420p16be           3           24
IO... yuv422p16le           3           32
IO... yuv422p16be           3           32
IO... yuv444p16le           3           48
IO... yuv444p16be           3           48
..H.. dxva2_vld             0           0
IO... rgb444le              3           12
IO... rgb444be              3           12
IO... bgr444le              3           12
IO... bgr444be              3           12
IO... ya8                   2           16
IO... bgr48be               3           48
IO... bgr48le               3           48
IO... yuv420p9be            3           13
IO... yuv420p9le            3           13
IO... yuv420p10be           3           15
IO... yuv420p10le           3           15
IO... yuv422p10be           3           20
IO... yuv422p10le           3           20
IO... yuv444p9be            3           27
IO... yuv444p9le            3           27
IO... yuv444p10be           3           30
IO... yuv444p10le           3           30
IO... yuv422p9be            3           18
IO... yuv422p9le            3           18
IO... gbrp                  3           24
IO... gbrp9be               3           27
IO... gbrp9le               3           27
IO... gbrp10be              3           30
IO... gbrp10le              3           30
IO... gbrp16be              3           48
IO... gbrp16le              3           48
IO... yuva422p              4           24
IO... yuva444p              4           32
IO... yuva420p9be           4           22
IO... yuva420p9le           4           22
IO... yuva422p9be           4           27
IO... yuva422p9le           4           27
IO... yuva444p9be           4           36
IO... yuva444p9le           4           36
IO... yuva420p10be          4           25
IO... yuva420p10le          4           25
IO... yuva422p10be          4           30
IO... yuva422p10le          4           30
IO... yuva444p10be          4           40
```

```
IO... yuva444p10le           4            40
IO... yuva420p16be           4            40
IO... yuva420p16le           4            40
IO... yuva422p16be           4            48
IO... yuva422p16le           4            48
IO... yuva444p16be           4            64
IO... yuva444p16le           4            64
..H.. vdpau                  0            0
IO... xyz12le                3            36
IO... xyz12be                3            36
..... nv16                   3            16
..... nv20le                 3            20
..... nv20be                 3            20
IO... rgba64be               4            64
IO... rgba64le               4            64
IO... bgra64be               4            64
IO... bgra64le               4            64
IO... yvyu422                3            16
IO... ya16be                 2            32
IO... ya16le                 2            32
IO... gbrap                  4            32
IO... gbrap16be              4            64
IO... gbrap16le              4            64
..H.. qsv                    0            0
..H.. mmal                   0            0
..H.. d3d11va_vld            0            0
..H.. CUDA                   0            0
IO... 0rgb                   3            24
IO... rgb0                   3            24
IO... 0bgr                   3            24
IO... bgr0                   3            24
IO... yuv420p12be            3            18
IO... yuv420p12le            3            18
IO... yuv420p14be            3            21
IO... yuv420p14le            3            21
IO... yuv422p12be            3            24
IO... yuv422p12le            3            24
IO... yuv422p14be            3            28
IO... yuv422p14le            3            28
IO... yuv444p12be            3            36
IO... yuv444p12le            3            36
IO... yuv444p14be            3            42
IO... yuv444p14le            3            42
IO... gbrp12be               3            36
IO... gbrp12le               3            36
IO... gbrp14be               3            42
```

```
IO... gbrp14le              3          42
IO... yuvj411p              3          12
I.... bayer_bggr8           3          8
I.... bayer_rggb8           3          8
I.... bayer_gbrg8           3          8
I.... bayer_grbg8           3          8
I.... bayer_bggr16le        3          16
I.... bayer_bggr16be        3          16
I.... bayer_rggb16le        3          16
I.... bayer_rggb16be        3          16
I.... bayer_gbrg16le        3          16
I.... bayer_gbrg16be        3          16
I.... bayer_grbg16le        3          16
I.... bayer_grbg16be        3          16
..H.. xvmc                  0          0
IO... yuv440p10le           3          20
IO... yuv440p10be           3          20
IO... yuv440p12le           3          24
IO... yuv440p12be           3          24
IO... ayuv64le              4          64
..... ayuv64be              4          64
..H.. videotoolbox_vld      0          0
IO... p010le                3          15
IO... p010be                3          15
IO... gbrap12be             4          48
IO... gbrap12le             4          48
IO... gbrap10be             4          40
IO... gbrap10le             4          40
..H.. mediacodec            0          0
IO... gray12be              1          12
IO... gray12le              1          12
IO... gray10be              1          10
IO... gray10le              1          10
IO... p016le                3          24
IO... p016be                3          24
..H.. d3d11                 0          0
IO... gray9be               1          9
IO... gray9le               1          9
IO... gbrpf32be             3          96
IO... gbrpf32le             3          96
IO... gbrapf32be            4          128
IO... gbrapf32le            4          128
..H.. drm_prime             0          0
..H.. opencl                0          0
IO... gray14be              1          14
IO... gray14le              1          14
```

```
IO... grayf32be        1        32
IO... grayf32le        1        32
IO... yuva422p12be     4        36
IO... yuva422p12le     4        36
IO... yuva444p12be     4        48
IO... yuva444p12le     4        48
IO... nv24             3        24
IO... nv42             3        24
..H.. vulkan           0        0
..... y210be           3        20
I.... y210le           3        20
```