VCB-Studio教程04 BDRip的制作流程

0. 前言

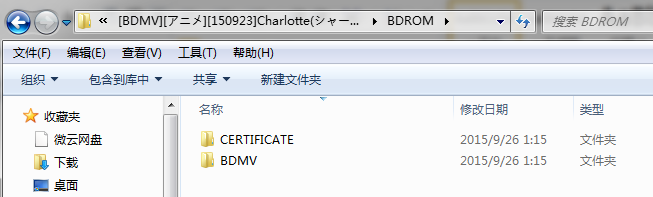
本教程的目的是作为VCB-Studio群内教程，按照VCB-Studio的标准流程，来讲述BDRip的制作。这只是vcb-s的习惯流程，针对vcb-s团队分工习惯而编写。而没有任何设定“业界标准”或标榜“最佳步骤”的意思。实际操作中，非vcb-s组员的读者没有必要完全参照，因为可能有很多步骤和规范，有更好、更简便的做法。

1. 认识BDMV目录结构

BDMV因为抓取步骤和放流人士的区别，上层目录可能有各种不同的样子。然而，一定有一个目录包括两个子目录：BDMV和CERTIFICATE。这两个目录的上级，我们称为根目录。比如说这是U2上《四月是你的謊言Finale Event》的蓝光，其根目录名称为BD\_VIDEO:

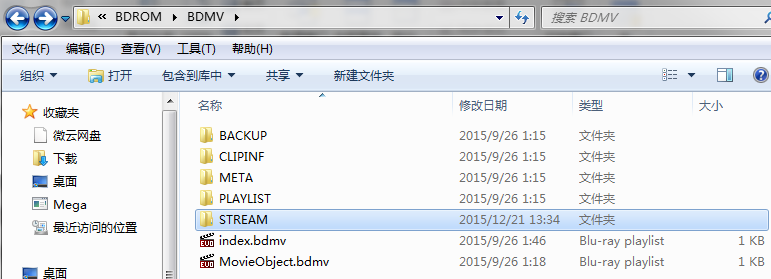


另一个例子是Charlotte的Vol.1:

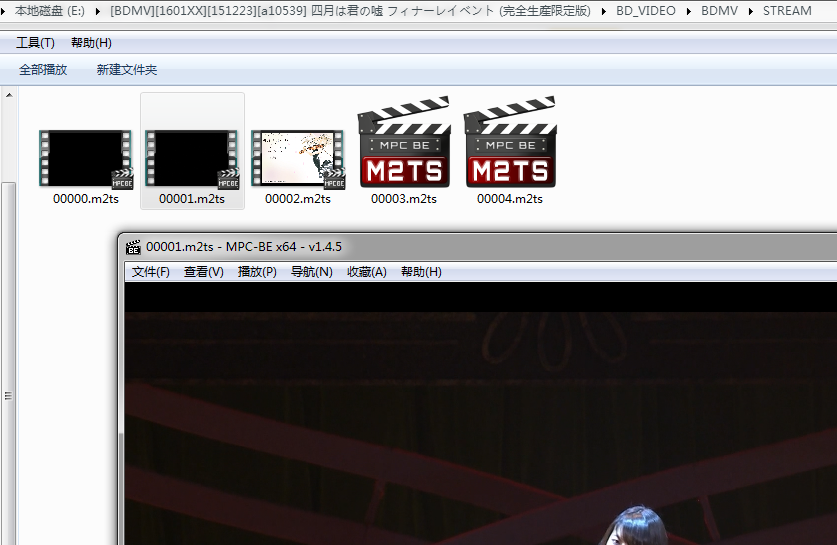


这里蓝光的根目录叫做BDROM，可见不同蓝光，甚至不同卷之间，根目录的名称不固定，但是根目录下一定有BDMV和CERTIFICATE两个子目录。

BDMV目录里的内容大致如下，其中重要的文件夹有两个，STREAM和PLAYLIST：

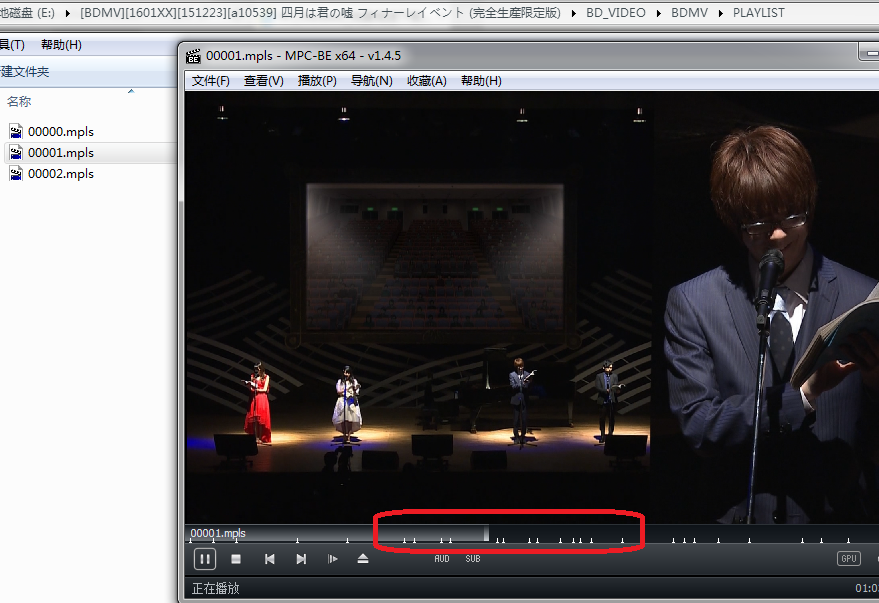


STREAM文件夹就是蓝光视频所在的地方：



一般来说，m2ts后缀的就是蓝光视频文件。有些不能播放的，多是菜单按钮等信息。

PLAYLIST文件夹下的文件后缀为mpls，是蓝光的播放列表信息，多半可以被播放器直接播放。它当中还储存着蓝光的章节信息（以后讲述提取章节信息，就是从这里获取的），可以在播放器中被显示：



BDMV中对BDRip的有效信息大概就这么多。

2. 用avs/vs载入视频

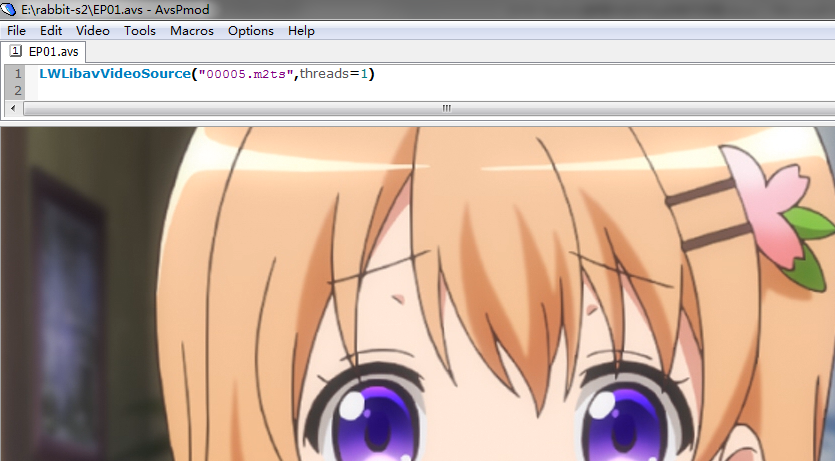
压制视频，我们一般是先用avs/vs载入视频，并做一定的处理。这份教程中我们只介绍最基本的载入视频的方法。

avs的写法如下：

LWLibavVideoSource("00000.m2ts",threads=1)

就是用Libav(LAV)分离并解码00000.m2ts的视频部分。文件名自己改；threads=1是保证单线程解码，避免解码过程中出现未知问题（多线程解码在一些视频上容易出现问题）

用avspmod调试，效果如下：



VS的写法如下：

import vapoursynth as vs

import sys

core = vs.get\_core(accept\_lowercase=True,threads=8)

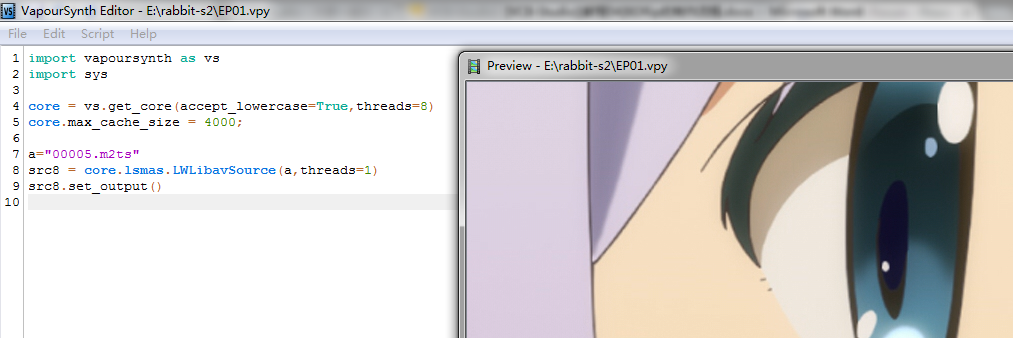
core.max\_cache\_size = 4000;

src8 = core.lsmas.LWLibavSource("00000.m2ts",threads=1)

src8.set\_output()

最上面4行是在Python环境中定义VS的运行，threads指定vs最多可以多线程优化的线程数，max\_cache\_size则是可以分配的内存数。后两行是载入视频，并输出(set\_output)

用vseditor预览，可以看到画面：

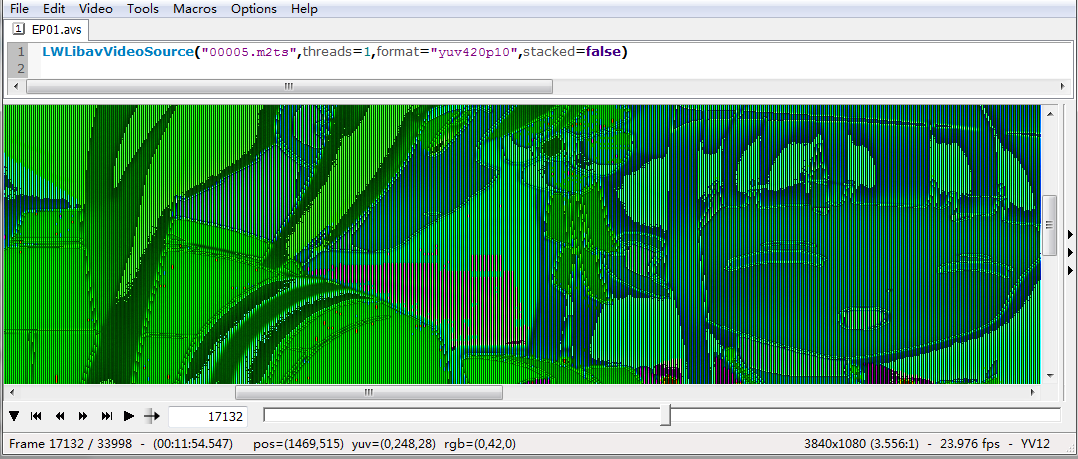


3. avs/vs的纪律性检查

在日常压制任务中，调试avs/vs的时候，对avs/vs进行纪律性检查是非常重要的，不然很有可能跑了几十小时后，最终发现做错了，得全盘重做。无数血淋淋的经验表明：纪律性检查不会让你压片效率提升多少，但是做不到这个绝对会有一天让你效率变成负数。

1. 首先检查的因素是画面。简单说，要压第一集，你是不是载入正确的m2ts, 别压成了第二集。这个通过avspmod/vseditor能很容易的检查到。接着检查画面正常与否。对于8bit压制任务，avs/vs都应该显示正常的画面，你可以检查是否有不正常的输出效果，花屏，等等。

如果是10bit压制任务，vcb-s一般会使用Interleaved 10bit输出（以后会细讲），画面看起来是这样的：



这时候可以选择Video - Bitdepth - Interleaved yuv420p10 or yuv444p10, avspmod就可以显示正常。

而vs中画面始终正常，不论是8bit还是10bit。

2. 随后需要检查的是avspmod/vseditor的状态栏。这两个状态栏可以给你非常丰富的信息。首先是avspmod的：



从左到右，先是先是总帧数。一般一集24分钟，24p帧率的番，总帧数在35000帧左右。如果差的太多（10%以上）一般就有问题。

然后是分辨率。8bit输出的前提下，分辨率该是多少就是多少；Interleaved 10bit输出，分辨率一般是横向\*2, 纵向不变。比如Interleaved 10bit输出，如果不调整，分辨率会显示3840x1080；调整bitdepth之后则显示1920x1080.

3. 接着需要检查的是帧率。avspmod 会显示帧率是23.976，还是29.970，或者是59.940。一般不同帧率的东西不能用一套制作方案；比如说，用于24p正片的参数，一般不能直接用于30fps的特典。

现在动漫蓝光基本上以24p为主，然而，你一旦发现30fps特典，应该第一时间有警觉，这不能按照24p的做法去制作。

4. 最后要检查的是格式。YV12=YUV420, YV24=YUV444, RGB则表示输出是RGB。一般我们是不可能把RGB作为压制输出的，所以如果你拿到的avs输出是RGB，那么就是调试开关忘记关闭了。

vs的检查基本类似：



跟avs的区别在于这几点：

1. vs的size直接就是原生的分辨率，不管输出精度是8/10/16;

2. Format中附带了精度。所以检查8bit/10bit，在format中检查。一般8bit压制输出是YUV420P8，10bit是YUV420P10

3. 千万不要忘记检查帧率是23.976/29.970还是59.940

一般拿到avs/vs脚本，或者自己改写，只有当你调试完毕，确认没问题，才可以开始下一步骤的操作。

4. x264/x265的使用

avs和vpy就位之后，就可以送压x264/x265。一般来说，我们使用64bit版本。x264/x265有8bit/10bit的区分，一般来说，我们使用的x264，8bit/10bit是不同的exe（不同源码编译成不同的执行文件）；而x265官方版，8/10/12bit是一个exe，通过参数来控制输出；也有分开编译的第三方。（比如Yukki Mod）

本系列教程中，我们一般认为x264有8bit/10bit之分；10bit x264往往会以 x264\_10bit这种名称标注。

将预处理脚本转换为YUV数据，送给编码器，需要一个中间“转换器”。对于avs，转换器为avs4x264.exe、avs4x265.exe(这两个可以在megui\tools\x264和x265中找到), 对于vs，转换器为vspipe.exe，这个可以从VapourSynth\Core64里面找。

因为其高度相似性，avs4x264和avs4x265可以被合二为一，比如最新的tMod版本就支持启动x262/x264/x265(<https://www.nmm-hd.org/newbbs/viewtopic.php?f=8&t=403&hilit=avs4x26x>)。本教程中，我们还是区别对待。

所以压片的流程，如果跟播放的分离-解码-修改-渲染相比，那就是：

“转换器”执行avs/vs脚本，对（本身是压缩格式的）片源做分离、解码和修改，输出YUV数据；

x264/x265把YUV数据编码成AVC/HEVC的视频压缩格式。

压片一般会写一个批处理。以下教程假设我们做8bit压制，就是预处理输出，和压制工具都是8bit。

用x264压制avs，使用avs4x264.exe:

avs4x264 --x264-binary "x264\_64.exe" --preset slow --crf 18 -o "EP01.264" "EP01.avs"

其中文件名（输入、输出、exe）可以自己改。--preset slow --crf 18属于x264参数设置，后续教程会介绍。

如果是用x265压制的话，大同小异：

avs4x265 --x265-binary "x265\_64.exe" --preset slow --crf 18 -o "EP01.265" "EP01.avs"

同理，--preset slow --crf 18属于x265的参数设置。

使用tMod版本，直接把--x264-binary和--x265-binary 合并为--x26x-binary

如果是vs压制，则使用vspipe.exe:

vspipe.exe --y4m "EP01.vpy" - | "x264\_64" --demuxer y4m --preset slow --crf 18 --output "00010.264" -

vspipe.exe --y4m "EP01.vpy" - | "x265\_64" --y4m --preset slow --crf 18 --output "00010.265" -

同样，输入输出和编码器名称自己设定。编码器的名称，exe后缀可以省略。

输入文件名没啥好说的，一般为.avs或者.vpy;

输出文件名，用x265的老老实实把后缀设置为.265，用x264则建议设置为mkv。输出mkv的好处在于，可以直接预览半成品，方便检查。如果是准备封装成mp4成品，后缀名也可以设置为.mp4。

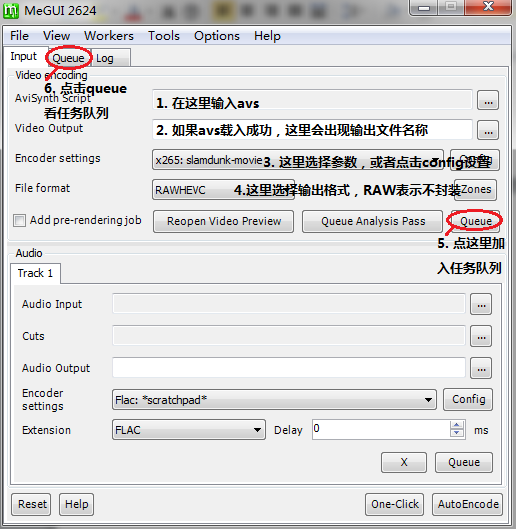
x264会根据后缀名，自动选择输出文件是否自带容器。.mp4则用MP4容器，.mkv则用mkv，否则就是纯AVC数据流。

x264输出的mkv，并非是最标准的封装，表现为在一些时候无法拖动进度条，花屏等。经过正规封装之后（后文会说）一般问题解决。

如果需要进行10bit的压制，x264的做法是把exe换成10bit版本的（例如MeGUI自带的x264-10b\_64.exe）；x265的做法是参数中加一个 -D 10 表示输出精度为10bit。、

以后的教程中我们还会讲原生高精度的输出，如果avs的输出是10bit/16bit精度，参数中需要加一个--input-depth 10或者--input-depth 16，来保证avs4x264能正确的把高精度颜色喂给编码器。vs压制无需照顾这一点，因为vs在设计的时候就有原生的高精度支持。5. 用MeGUI压制avs任务

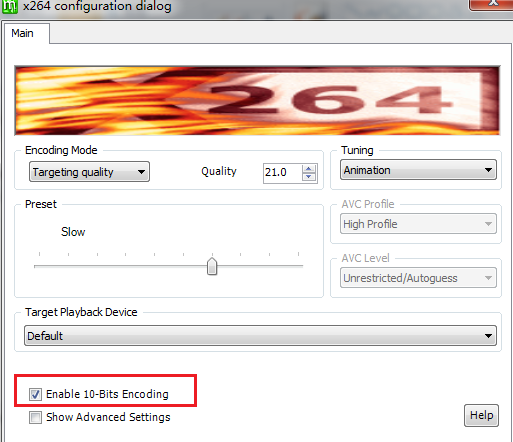
如果是avs输入，上述x264/x265的使用可以通过MeGUI实现：



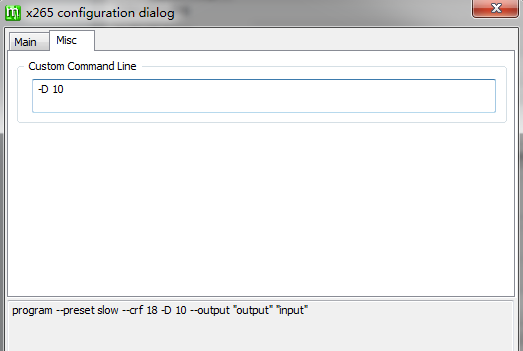
第1步中，如果avs有问题，MeGUI会弹窗报错。

第3步中，保存好的参数文件，一般是.xml格式，比如x265\_dp\_ slamdunk-movie.xml。对于给定的xml文件，正确的载入方式是**在关闭MeGUI的前提下**，把xml文件放入MeGUI\allprofiles\x265或者MeGUI\allprofiles\x264中（取决于这是x264还是x265的参数）

手写参数的前提下，如果你用x264，要进行10bit压制，你需要在设置中勾选Enable 10-bits Encoding:



如果你用x265，则需要在设置中手动写入-D 10:

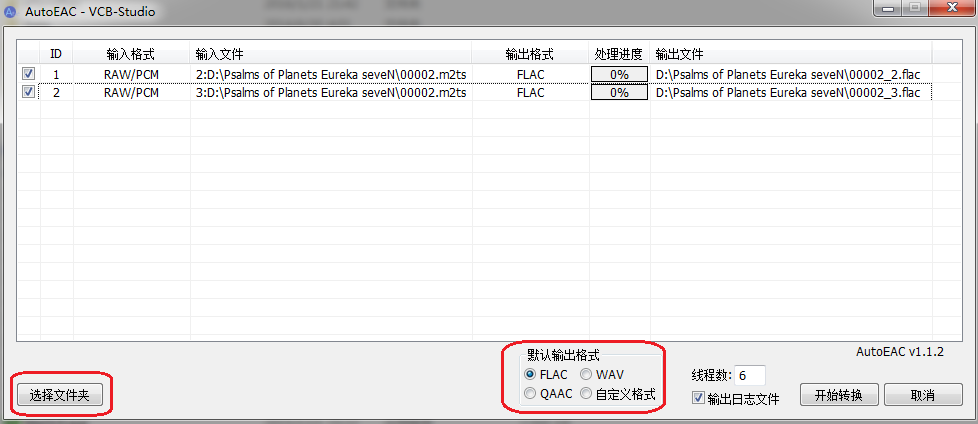


6. 抽取原盘音轨

抽取原盘音轨，最标准的方法是在MeGUI中，使用Tools-HD Stream Extractor来调用eac3to进行抽取。这部分知识我们会在后续教程中详细说，本教程则教大家使用VCB-Studio AutoEAC来便捷抽取。

把AutoEAC解压到某个地方，就可以使用了。AutoEAC自带了eac3to和qaac，如果你已经按照上文设置了MeGUI，可以只解压exe和ini放到megui根目录下，这样AutoEAC会自动调用MeGUI的tools。

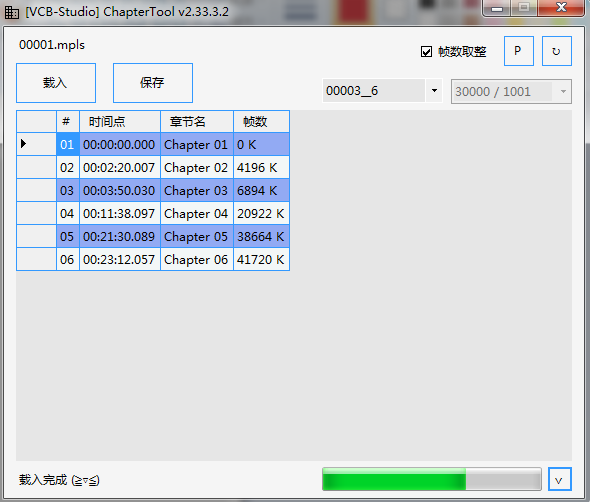
AutoEAC的使用非常简单，左下角选择BDMV目录，再选择输出格式就好。如果不想要某条轨道，可以通过最前面的选择框取消，或者直接双击删除轨道。



AutoEAC输出的AAC，使用qaac --cvbr 128K编码。

7. 抽取原盘章节

原盘章节的抽取可以用Chapter Tools来制作，这里只讲述它最简单的用法：

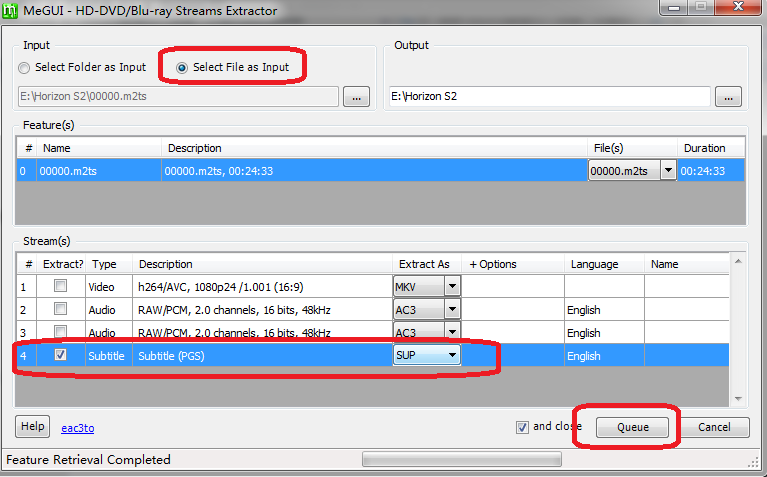


点击载入，选择mpls（前文说过，这玩意就是章节文件所在处）

载入后，点击保存，就可以去playlist文件夹下找对应的txt

8. 抽取原盘字幕

原盘字幕的抽取，需要用MeGUI-tools-HD Stream Extractor：



如图所示，选择Select File As Input，选中m2ts。MeGUI会自动分析，分析完毕后状态栏显示Feature Retrieved Completed。

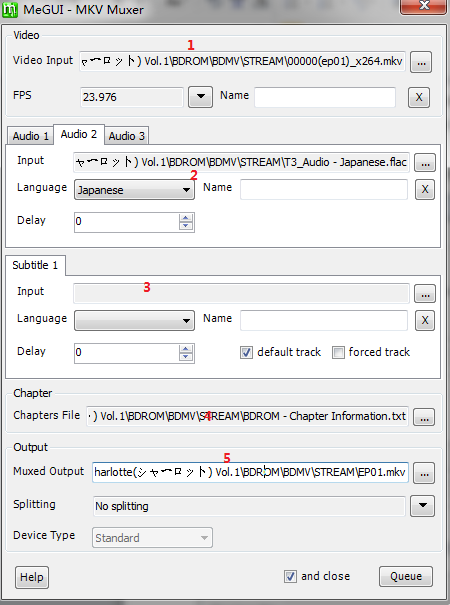
然后就选中Subtitle，queue，执行。抽取出来的字幕后缀是\*.sup，在m2ts同目录中。

这个工具其实也可以用来抽音轨，效果和操作逻辑跟AutoEAC类似，都是调用eac3to。但是切记不要用它抽取aac。（因为MeGUI抽aac调用的是NeroAAC，需要另外配置）

9. 成品的封装与检查

准备好压制完毕的视频、音频、章节和字幕之后，就可以着手准备封装成品了。MeGUI提供了简单的MP4/MKV封装工具，在tools-muxer中。其他更高级工具，例如mkvtoolnix，会在以后的教程中讲解。

下图以MKV muxer为例，MP4的大同小异：



1：输入视频文件。注意封装MKV的时候，输入可以是264/mp4/mkv，封装mp4则必须是264/mp4。如果压制出来纯视频是mkv格式，则先要抽取纯264视频流。检查帧率，一般都是24p 23.976，如果是60fps版就是59.940。

2：输入音频。如果有多条音轨，输入第一条后，旁边自动出现第二条的位置，如图所示。

3：输入字幕（如果有），类似输入音频，然后设置音频语言信息，一般动漫么都是日语音轨，选japanese，如果有英配，中配，则选English或者Chinese，诸如此类。

4：输入章节文件，一般为txt格式

5：设置输出名称。vcb-s的规范一般以卷数-标号为准，比如v1-00000.mkv，表明是Vol.1的00000.m2ts压制而来。设置好后就可以点击Queue加入队列，在队列中启动项目

封装好的成品，要进行再一次的检查，主要包括：

1. 播放效果是否正常，这时候的成品应该完全符合BDRip成品要求；

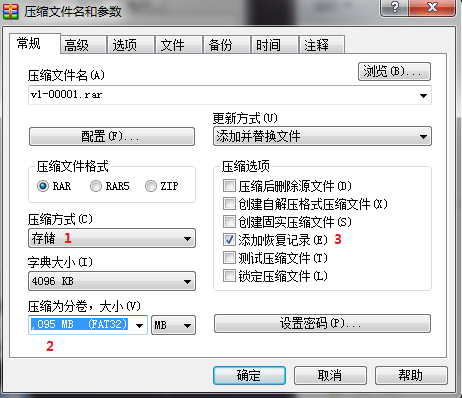
2. 章节是否正确，能否通过章节来实现跳段；

3. 音轨/字幕是否齐全，能不能正确播放，切换时候语言显示正确与否，是否与画面时间匹配；

4. mediainfo中，分辨率、比例、多音轨字幕、bitdepth等信息是否正确。

10. 成品的上传规范

VCB-Studio压制组的成品一般是传到百度网盘，然后给整理组接手。百度网盘默认只允许最大4GB的文件，所以任何更大的文件必须通过rar分卷解决：



打包时候，有几点需要注意：

1. 压缩方式选储存。这种是不做任何压缩，直接复制。觉得想通过压缩包省点体积的不妨试试能省多少。（事实上，x264/x265编码最后一步就是类似7z做的事情）

2. 分卷大小，选择4GB

3. 添加恢复记录，这是为了防止传输过程中，数据损坏导致压缩包不能正常解压。有恢复记录（默认3%）足够解决绝大多数时候的传输数据损坏。

其他时候对于小文件，我们更多使用RapidCRC来添加md5/CRC32. 使用方法很简单，打开界面，把成品mkv拖进去，点击Put CRC into Filename。软件会提示你是否把你选中的，所有不带CRC32后缀的文件打上后缀，点击Yes，文件名后就自动带上了CRC32，类似v1-00005 [7D7C6223].mkv。

这个机制可以帮助整理组快速检查文件损坏。你可以试着把某个文件的CRC32改动下，然后再拖到软件中看看。

设置好了之后，就可以上传到度盘准备交作业了。