Toybrick

标题: RK3399Pro入门教程（6）硬件编解码器MPP库的使用 **[打印本页]**

作者: jefferyzhang **时间:** 2019-4-16 11:57
标题: RK3399Pro入门教程（6）硬件编解码器MPP库的使用
本帖最后由 jefferyzhang 于 2021-3-16 09:39 编辑

[attach]239[/attach]

MPP库安装方式

1. dnf 安装（详见wiki）

```
1. sudo dnf install librockchip_mpp-devel
```

复制代码
2. 源码编译

MPP库源码下载地址

```
1. https://github.com/rockchip-linux/mpp
2. 或 https://github.com/HermanChen/mpp
```

复制代码
MPP兼容的gstreamer源码下载地址

```
1. https://github.com/rockchip-linux/gstreamer-rockchip
```

复制代码

如果大家习惯使用gstreamer或者ffmpeg的接口的话，可以直接使用以上源码进行快速开发或代码迁移。但是这两个库已经交给开源社区维护了，遇到问题大家只能自己按着源码自己去debug。不是很建议项目中去使用。

MPP库简介

MPP库是Rockchip根据自己的硬编解码器开发的应用程序编解码库，如果想达到最好的效果，必须要通过librockchip_mpp来直接编码实现编解码。gstreamer和ffmpeg都会因为兼容api的原因，徒增几次无用的帧拷贝动作，并且使用的都是虚拟地址。在上一篇RGA的教学中，我们知道纯物理连续地址的硬件操作是非常快的，转到虚拟地址后效率就会降低。如果想榨干Toybrick的性能，开发最完美的代码，纯连续的物理Buffer、mpp+rga是离不开的。

Mpp的API思路其实跟目前绝大多数的编解码库是一致的，都是queue/dequeue的队列操作方式，先设置好编解码状态，然后不停的queue/dequeue input/output buffer就可以实现编解码控制了。如果大家熟悉FFMPEG，那学习MPP会非常容易，MPP和FFMPEG的api非常相像。

Mpp库自带的sample基本可以带大家入手。

MPP编译

1. （没有交叉编译环境的建议还是直接放在板子上编译） cd 到build目录里对应平台的目录

```
1. cd build/linux/aarch64
```

复制代码

2. 如果是交叉编译环境，需要修改该目录下编译链的配置。然后执行编译脚本。

```
1. ./make-Makefiles.bash
```

复制代码

MPP套路讲解

解码范例在mpp源码内：test/mpi_dec_test.c
编码范例在mpp源码内：test/mpi_enc_test.c

1. 创建 MPP context 和 MPP api 接口。 （注意，和RGA一样，多个线程多个实例需要多个独立的的context）

```
1. ret = mpp_create(&ctx, &mpi);
2.
3. if (MPP_OK != ret) {
4.       mpp_err("mpp_create failed\n");
5.       goto MPP_TEST_OUT;
6. }
```

复制代码

2. 设置一些MPP的模式（这里设置的是 MPP_DEC_SET_PARSER_SPLIT_MODE）

```
1. mpi_cmd = MPP_DEC_SET_PARSER_SPLIT_MODE;
2. param = &need_split;
3. ret = mpi->control(ctx, mpi_cmd, param);
4. if (MPP_OK != ret) {
5.       mpp_err("mpi->control failed\n");
6.       goto MPP_TEST_OUT;
7. }
```

复制代码
常用设置的一些模式解释如下：（其余的可以看MPP自带的开发文档，在doc目录下有详细说明）

MPP_DEC_SET_PARSER_SPLIT_MODE： （仅限解码）
自动拼包（建议开启），硬编解码器每次解码就是一个Frame，所以如果输入的数据不确定是不是一个Frame（例如可能是一个Slice、一个Nalu或者一个FU-A分包，甚至可能随意读的任意长度数据），那就必须把该模式打开，MPP会自动分包拼包成一个完整Frame送给硬解码器。

MPP_DEC_SET_IMMEDIATE_OUT： （仅限解码）
立即输出模式（不建议开启），如果未开立即输出模式，MPP会按预先设定的节奏间隔输出解码的帧（例如33ms输出一帧）。但是实际硬件解码过程并不是均匀输出的，有时候两帧间隔可能就1ms一下子输出2-3帧，有时候两帧间又会有较长的间隔。如果打开立即输出模式，MPP就会在解码成功后立即输出一帧，那后续处理显示的节奏就需要用户自己控制。该模式适用于一些对实时性要求比较高的客户产品，需要自己把握输出节奏。

MPP_SET_INPUT_BLOCK:
MPP_SET_INTPUT_BLOCK_TIMEOUT:
MPP_SET_OUTPUT_BLOCK:
MPP_SET_OUTPUT_BLOCK_TIMEOUT:
设置输入输出的block模式，如果block模式打开，喂数据时候会block住直到编解码成功入队列或者出队列或者达到TIMEOUT时间，才会返回。

3. 初始化 MPP

```
1. ret = mpp_init(ctx, MPP_CTX_DEC, MppCodingType::MPP_VIDEO_CodingAVC);
2. if (MPP_OK != ret) {
3.       mpp_err("mpp_init failed\n");
4.       goto MPP_TEST_OUT;
5. }
```

复制代码
初始化编码还是解码，以及编解码的格式。
MPP_CTX_DEC： 解码
MPP_CTX_ENC： 编码
MPP_VIDEO_CodingAVC： H.264
MPP_VIDEO_CodingHEVC： H.265
MPP_VIDEO_CodingVP8： VP8
MPP_VIDEO_CodingVP9： VP9
MPP_VIDEO_CodingMJPEG : MJPEG
等等，详细参看rk_mpi.h定义

4. 解码的话到这里初始化就完成了，编码的话需要多设置一些参数

设置编码宽高、对齐后宽高参数

```
1.       mPrepCfg.change = MPP_ENC_PREP_CFG_CHANGE_INPUT | MPP_ENC_PREP_CFG_CHANGE_FORMAT;
2.       mPrepCfg.width = mWidth;
3.       mPrepCfg.height = mHeight;
4.       mPrepCfg.hor_stride = mHStride;
5.       mPrepCfg.ver_stride = mVStride;
6.       mPrepCfg.format = mFrameFormat;
7.
8.       int ret = mMppApi->control(mMppCtx, MPP_ENC_SET_PREP_CFG, &mPrepCfg);
```

复制代码

设置编码码率、质量、定码率变码率

```
1.       mRcCfg.change = MPP_ENC_RC_CFG_CHANGE_ALL;
2.
3.       /*
4.          * rc_mode - rate control mode
5.          * Mpp balances quality and bit rate by the mode index
6.          * Mpp provide 5 level of balance mode of quality and bit rate
7.          * 1 - only quality mode: only quality parameter takes effect
8.          * 2 - more quality mode: quality parameter takes more effect
9.          * 3 - balance mode        : balance quality and bitrate 50 to 50
10.         * 4 - more bitrate mode: bitrate parameter takes more effect
11.         * 5 - only bitrate mode: only bitrate parameter takes effect
12.         */
13.      if (mIsCBR) {
14.            mRcCfg.rc_mode = (MppEncRcMode) MPP_ENC_RC_MODE_CBR;
15.      } else {
16.            mRcCfg.rc_mode = (MppEncRcMode) MPP_ENC_RC_MODE_VBR;
17.      }
18.
19.      /*
20.         * quality - quality parameter
21.         * mpp does not give the direct parameter in different protocol.
22.         * mpp provide total 5 quality level 1 ~ 5
23.         * 0 - auto
24.         * 1 - worst
25.         * 2 - worse
26.         * 3 - medium
27.         * 4 - better
28.         * 5 - best
29.         */
30.      if (mQuality > 4) {
31.            mRcCfg.quality = (MppEncRcQuality)MPP_ENC_RC_QUALITY_BEST;
32.      } else {
33.            mRcCfg.quality = (MppEncRcQuality)mQuality;
34.      }
35.
36.
37.      int bps = mBps;
38.      switch (mRcCfg.rc_mode) {
39.            case MPP_ENC_RC_MODE_CBR:
```

```
                // constant bitrate has very small bps range of 1/16 bps
                mRcCfg.bps_target = bps;
                mRcCfg.bps_max = bps * 17 / 16;
                mRcCfg.bps_min = bps * 15 / 16;
                break;
            case MPP_ENC_RC_MODE_VBR:
                // variable bitrate has large bps range
                mRcCfg.bps_target = bps;
                mRcCfg.bps_max = bps * 3 / 2;
                mRcCfg.bps_min = bps * 1 / 2;
                break;
            default:
                abort();
        }

        /* fix input / output frame rate */
        mRcCfg.fps_in_flex          = 0;
        mRcCfg.fps_in_num           = mFps;
        mRcCfg.fps_in_denorm        = 1;
        mRcCfg.fps_out_flex         = 0;
        mRcCfg.fps_out_num          = mFps;
        mRcCfg.fps_out_denorm       = 1;

        mRcCfg.gop                      = mIInterval; /* i frame interval */
        mRcCfg.skip_cnt               = 0;

        int ret = mMppApi->control(mMppCtx, MPP_ENC_SET_RC_CFG, &mRcCfg);
```

复制代码

设置264相关的其他编码参数

```
mCodecCfg.h264.change = MPP_ENC_H264_CFG_CHANGE_PROFILE |
                                        MPP_ENC_H264_CFG_CHANGE_ENTROPY |
                                        MPP_ENC_H264_CFG_CHANGE_TRANS_8x8 |
                                        MPP_ENC_H264_CFG_CHANGE_QP_LIMIT;

        /*
           * H.264 profile_idc parameter
           * 66    - Baseline profile
           * 77    - Main profile
           * 100 - High profile
           */
        mCodecCfg.h264.profile    = 100;

        /*
           * H.264 level_idc parameter
           * 10 / 11 / 12 / 13      - qcif@15fps / cif@7.5fps / cif@15fps / cif@30fps
           * 20 / 21 / 22              - cif@30fps / half-D1@@25fps / D1@12.5fps
           * 30 / 31 / 32              - D1@25fps / 720p@30fps / 720p@60fps
           * 40 / 41 / 42              - 1080p@30fps / 1080p@30fps / 1080p@60fps
           * 50 / 51 / 52              - 4K@30fps
           */
        mCodecCfg.h264.level      = 40;
        mCodecCfg.h264.entropy_coding_mode    = 1;
        mCodecCfg.h264.cabac_init_idc    = 0;
        mCodecCfg.h264.transform8x8_mode = 1;

        if (mRcCfg.rc_mode == MPP_ENC_RC_MODE_CBR) {
                mCodecCfg.h264.qp_init    = 10;
                mCodecCfg.h264.qp_min = 4;
                mCodecCfg.h264.qp_max = 30;
                mCodecCfg.h264.qp_max_step = 16;
        }

        int ret = mMppApi->control(mMppCtx, MPP_ENC_SET_CODEC_CFG, &mCodecCfg);
```

复制代码

5. 接下来就是喂数据和拿输出数据的过程了，具体可以直接看sample代码，这里解释下一些基本概念，方便大家看Sample代码时候不懵逼。

MppPacket ： 存放编码数据，例如264、265数据
MppFrame ： 存放解码的数据，例如YUV、RGB数据
MppTask ： 一次编码或者解码的session

编码就是喂MppFrame，输出MppPacket；
解码就是喂MppPacket，输出MppFrame；

MPI包含两套接口做编解码：
一套是简易接口， 类似 decode_put_packet / decode_get_frame 这样put/get即可
一套是高级接口， 类似 poll / enqueue/ dequeue 这样的对input output队列进行操作

解码得到的output buffer一般都拥有虚拟地址和物理地址的fd，紧接着就可以通过RGA做对应操作或者拷贝，速度是相当快的。

MPP时间打印和FAQ：

传送门 -> http://t.rock-chips.com/forum.php?mod=viewthread&tid=785&extra=page%3D1

作者: ronyuzhang **时间:** 2019-4-17 10:08
大赞的教程，良心，有求必应@jefferyzhang 希望再接再厉，嘉惠码农。

作者: yaowei **时间:** 2019-4-18 11:44
那么视频解码必须要用C++写的？python代码是没有的吗?

作者: jefferyzhang **时间:** 2019-4-18 12:26

> yaowei 发表于 2019-4-18 11:44
> 那么视频解码必须要用C++写的？python代码是没有的吗?

python可以使用gstreamer-rockchip来转接。例如opencv内部调用的是gstreamer的接口，是可以调用到硬解码的。具体可以问下其他开发者，他们有成功用起来过

作者: yaowei **时间:** 2019-4-18 17:17
我也成功用起来了，不止需要gstreamer-rockchip，还需要其他一些库，现在可以解码视频和rtsp摄像头。

作者: ronyuzhang **时间:** 2019-4-30 18:42
官方mipi的camera 什么时候发售

作者: jefferyzhang **时间:** 2019-5-1 19:12

> ronyuzhang 发表于 2019-4-30 18:42
> 官方mipi的camera 什么时候发售

快了...

作者: kiwi **时间:** 2019-5-8 18:18
MPP兼容的ffmpeg，是指用用这个ffmpeg去编解码会调用到vpu吗

作者: jefferyzhang **时间:** 2019-5-9 08:50

> kiwi 发表于 2019-5-8 18:18
> MPP兼容的ffmpeg，是指用用这个ffmpeg去编解码会调用到vpu吗

是的，GitHub 上有我们对接好的ffmpeg和gsteamer

作者: hzk8656511 **时间:** 2019-5-16 11:21
板子跑的是android，MPP有没有在linux下编译andorid mp.so 的详细步骤

作者: jefferyzhang **时间:** 2019-5-17 11:05

> hzk8656511 发表于 2019-5-16 11:21
> 板子跑的是android，MPP有没有在linux下编译andorid mp.so 的详细步骤

mpp的build目录里有安卓编译。
我们的android也是自带libmpp的，无需编译。
android的media_codec默认就会调用到硬解码，也不需要直接去操作mpp。

作者: hzk8656511 **时间:** 2019-5-17 14:16
本帖最后由 hzk8656511 于 2019-5-17 14:19 编辑

> jefferyzhang 发表于 2019-5-17 11:05
> mpp的build目录里有安卓编译。
> 我们的android也是自带libmpp的，无需编译。
> android的media_codec默认就 ...

目前板子上的mpplib有点老，每次调用mpi->control 就挂掉了，mpi->control函数指针为空，打印发现mpi->size为176，而实际上最新代码 sizeof(MppApi)为184， 应该是mpi->control 越界了，板子上面的mpplib的应该和最新代码头文件不对应的造成的，经过查看rk_mpi.h这个文件提交日志发现应该是提交 MPP_RET (*poll)(MppCtx ctx, MppPortType type, MppPollType timeout); 新增这个功能导致的，多个一个函数指针8个字节造成control越界了

我们的程序因为是驱动的问题用的是android的系统，不能直接调用media_code 是用C写的，
android编译不是很熟悉，能不能帮忙指导下具体详细编译步骤，或者帮忙给编个aarch64 android最新mpp的，非常感谢

作者: guanyuqin **时间:** 2019-5-17 15:27

> yaowei 发表于 2019-4-18 17:17
> 我也成功用起来了，不止需要gstreamer-rockchip，还需要其他一些库，现在可以解码视频和rtsp摄像头。 ...

你好，我想问下你解码rtsp摄像头是硬解码么，视频流的压缩格式是什么

作者: yaowei **时间:** 2019-5-20 08:47

> guanyuqin 发表于 2019-5-17 15:27
> 你好，我想问下你解码rtsp摄像头是硬解码么，视频流的压缩格式是什么

怎么看是硬解还是软解，反正我之前看不了摄像头的，装了一些东西之后就可以看到摄像头的视频了，而且还挺流畅的

作者: guanyuqin **时间:** 2019-5-20 09:49

> yaowei 发表于 2019-5-20 08:47
> 怎么看是硬解还是软解，反正我之前看不了摄像头的，装了一些东西之后就可以看到摄像头的视频了，而且还挺 ...

能告诉你gstreamer用的指令么，你的相机是什么视频格式的?

作者: newstarqu **时间:** 2019-6-11 09:55
MPP 解码出来的YUV可以再继续解码成RGB吗？用硬解码

作者: jefferyzhang **时间:** 2019-6-12 12:28

> newstarqu 发表于 2019-6-11 09:55
> MPP 解码出来的YUV可以再继续解码成RGB吗？用硬解码

可以通过RGA硬件转换RGB

作者: ehome4407 **时间:** 2019-6-14 09:12
楼主你好，我在rk3288上用MPP和RGA将camera的输出转成了RGB，数据正常，但是就是CPU的占用率跟之前比没有任何降低，反而高了一点，我是用的系统性能监视器app实时看的cpu的占用率的，把转换部分注释掉，cpu的占用率很低，一旦转换部分（MPP+RGA）打开，cpu占用率就上去了，但是我的确是用的MPP+RGA的硬解码实现的mjpeg转rgb，求楼主指点一下，看看是哪里用错了，搞了两天了

作者: mayl88222 **时间:** 2019-7-15 18:28
怎么可以通过jni方式调用呢

作者: jefferyzhang **时间:** 2019-7-16 08:40

> mayl88222 发表于 2019-7-15 18:28
> 怎么可以通过jni方式调用呢

jni直接调用amediacodec就可以了，安卓上都不需要直接调用mpp

作者: 盗骊_I **时间:** 2019-7-16 17:29
我在Android上编译了mpp,demo程序也能正常跑。我自己写了个程序按照demo的使用方式，也能正常解码，但是发现解码出数据是不均匀的，不是传一帧，解码出一帧数据。有的时候，传十几帧，然后才连续出十几帧的数据。请问这是什么情况？过程当中还会遇到很多

```
1. decoder_get_frame get err info:1 discard:0
```

复制代码这个错误，有是什么情况?

作者: jefferyzhang **时间:** 2019-7-18 10:06

> 盗骊_I 发表于 2019-7-16 17:29
> 我在Android上编译了mpp,demo程序也能正常跑。我自己写了个程序按照demo的使用方式，也能正常解码，但是发 ...

可能原因：
1. I帧不完整，丢失，造成后面P帧无法解码
2. 码流带有B帧
3. 不管怎么样都不可能一帧进一帧出，除非你码流全是I帧

你发的错误看不出什么问题，拿不到帧很多情况是帧不完整无法解码造成的

作者: zhouzhouzlove **时间:** 2019-8-20 09:18
楼主大神，我在板子上安装了瑞芯微官网的ffmpeg，并且跑了个存储rtsp到本地的例子。但是我怎么知道是用的硬解码？咋说呢，我怎么验证。。。。。。。。

作者: jefferyzhang **时间:** 2019-8-20 10:58

> zhouzhouzlove 发表于 2019-8-20 09:18
> 楼主大神，我在板子上安装了瑞芯微官网的ffmpeg，并且跑了个存储rtsp到本地的例子。但是我怎么知道是用的硬 ...

最简单看CPU占用率就知道了，ARM的cpu做264软件速度很慢的，甚至大一点的视频码率是无法达到30fps的

作者: zhouzhouzlove **时间:** 2019-8-20 11:06

> jefferyzhang 发表于 2019-8-20 10:58
> 最简单看CPU占用率就知道了，ARM的cpu做264软件速度很慢的，甚至大一点的视频码率是无法达到30fps的 ...

是不是还可以这样：
从ffmpeg官网下载一个，编译安装这个库。
然后在编译自己的程序时，链接这个新的库，那么理论上来讲，这两个例子在跑时cpu占用率会有明显的差距?

作者: zhouzhouzlove **时间:** 2019-8-20 11:44

> zhouzhouzlove 发表于 2019-8-20 11:06
> 是不是还可以这样：
> 从ffmpeg官网下载一个，编译安装这个库。
> 然后在编译自己的程序时，链接这个新的库， ...

实验结果表明，这样做没有区别，使用top查看cpu占用率，都在2-3%。
有没有对比比较明显的例子。。。。。

作者: zhouzhouzlove **时间:** 2019-8-20 14:54
本帖最后由 zhouzhouzlove 于 2019-8-20 15:32 编辑

> jefferyzhang 发表于 2019-8-20 10:58
> 最简单看CPU占用率就知道了，ARM的cpu做264软件速度很慢的，甚至大一点的视频码率是无法达到30fps的 ...

ffmpeg的编译安装是这样的，大神看下是否有问题，三部曲如下：
./configure --enable-shared --prefix=/opt/ffmpeg

make

make install
在安装完后，修改/etc/ld.so.conf添加安装目录/opt/ffmpeg/lib，sudo ldconfig 后使用如下的命令编译：
g++ -Wall -fexceptions -std=c++11 -g -I"/opt/ffmpeg/include" -c ./save-rtsp.cpp -o ./save-rtsp.o

g++ -o save-rtsp ./save-rtsp.o   /opt/ffmpeg/lib/libavcodec.so /opt/ffmpeg/lib/libavdevice.so /opt/ffmpeg/lib/libavfilter.so /opt/ffmpeg/lib/libavformat.so /opt/ffmpeg/lib/libavutil.so /opt/ffmpeg/lib/libswresample.so /opt/ffmpeg/lib/libswscale.so

生成了可执行程序save-rtsp。同样的，从ffmpeg官网下载了一ffmpeg，安装在另一个目录，编译时修改系统环境变量和链接命令，同样生成了可执行文件。

但是这两个可执行文件执行时，cpu的占用率没有差别，都是2%左右。
讲道理的话，是不是ffmpeg官网的版本会有较高的cpu占用率。。。。

  --enable-rkmpp   enable Rockchip Media Process Platform code [no]
是不是应该在配置时打开这个选项？或者说，有没有官方的指导，关于怎么配置ffmpeg或者mpp的，都需要打开哪些支持

作者: jefferyzhang **时间:** 2019-8-20 15:41

> zhouzhouzlove 发表于 2019-8-20 14:54
> ffmpeg的编译安装是这样的，大神看下是否有问题，三部曲如下：
> ./configure --enable-shared --prefix=/op ...

是的，ffmpeg软解会有较高的CPU占用率，当然你码率要不高，就320p的264，那区别不大。
走VPU解码CPU是几乎不会有占用率的。

作者: zhouzhouzlove **时间:** 2019-8-20 16:20

> jefferyzhang 发表于 2019-8-20 15:41
> 是的，ffmpeg软解会有较高的CPU占用率，当然你码率要不高，就320p的264，那区别不大。
> 走VPU解码CPU是几 ...

首先多谢大神的耐心回复！

--enable-rkmpp   enable Rockchip Media Process Platform code [no]
是不是应该在配置时打开这个选项？我没打开，当添加这个选项时，配置报错会说：rkmpp is version3 and --enable-version3 is not specified.
当加上--enable-version3后会报错，pkg-config未安装。
[attach]488[/attach]

我不知道需不需要额外的打开一些开关，还是说，直接指定编译为共享库就行了。就像之前提到的这样，/configure --enable-shared --prefix=/opt/ffmpeg
另外，问个白痴的问题，安装了ffmpeg之后是否还需要安装mpp？现在是未安装mpp，只安装了ffmpeg。。。。。。。。。。

作者: jefferyzhang **时间:** 2019-8-21 09:09

> zhouzhouzlove 发表于 2019-8-20 16:20
> 首先多谢大神的耐心回复！
>
> --enable-rkmpp   enable Rockchip Media Process Platform code [n ...

这个我真不知道- -# 因为这个库是另外一个部门维护的，你们可以直接在github上的issues里问他们问题。
我们部门跟你们一样也只是用了这个库而已。。。

作者: zhouzhouzlove **时间:** 2019-8-21 09:37

> jefferyzhang 发表于 2019-8-21 09:09
> 这个我真不知道- -# 因为这个库是另外一个部门维护的，你们可以直接在github上的issues里问他们问题。
> 我 ...

好的，收到！
多谢大神的耐心解答~
么么哒
——抠脚大汉

作者: kiwi **时间:** 2019-8-21 10:23

> zhouzhouzlove 发表于 2019-8-20 16:20
> 首先多谢大神的耐心回复！
>
> --enable-rkmpp enable Rockchip Media Process Platform code [n ...

你这个想法，我之前做过同样的事情，编译ffmpeg打开--enable-rkmpp需要先编译安装mpp，然后配置好mpp的pkgconfig，然后配置ffmpeg能查找到mpp的pkgconfig才能编译成功。
不过最后我用ffmpeg来解码明显还是调用不到vpu，建议还是直接使用mpp的api来解码，这样百分百能调用到vpu。

作者: zhouzhouzlove **时间:** 2019-8-21 17:21

> kiwi 发表于 2019-8-21 10:23
> 你这个想法，我之前做过同样的事情，编译ffmpeg打开--enable-rkmpp需要先编译安装mpp，然后配置好mpp的pk ...

然后配置好mpp的pkgconfig
少侠，这句话可以具体一点么？

作者: kiwi **时间:** 2019-8-22 16:04

> zhouzhouzlove 发表于 2019-8-21 17:21
> 然后配置好mpp的pkgconfig
> 少侠，这句话可以具体一点么？

就是配置其lib下的pkgconfig目录下的pc文件

作者: zhouzhouzlove **时间:** 2019-8-23 10:27

> kiwi 发表于 2019-8-22 16:04
> 就是配置其lib下的pkgconfig目录下的pc文件

多谢少侠的回复！
我做了个实验：
1、只安装ffmpeg官网的ffmpeg4.0版本，用自己的程序去测试（解码rtsp流），发现可以软解；
2、只安装瑞芯微的ffmpeg4.0版本，提示无法找到解码器，打开流失败；
3、安装了mpp，同样提示无法找到解码器，打开流失败。
结论如下：我不知道怎么配置ffmpeg和mpp，才能使得ffmpeg可以调用到VPU。
希望官方能够给出一个教程，否则只说怎么调用API，但是如果不会配置安装的话，这都是没用的。

强烈建议楼主给出相关部门人员的帖子链接或者联系方式，以便留言询问，自github上提了问题，但是感觉应该没人看。

报错截图如下：27是codec_id,在3。3的版本下是28，但是4.0以上就是27了，不知道为什么，但是应该不打紧。
[attach]495[/attach]

作者: kiwi **时间:** 2019-8-23 14:10

> zhouzhouzlove 发表于 2019-8-23 10:27
> 多谢少侠的回复！
> 我做了个实验：
> 1、只安装ffmpeg官网的ffmpeg4.0版本，用自己的程序去测试（解码rtsp流 ...

建议别捣鼓ffmpeg了，他们也只是对接了api，据说是有bug的，也不会提供ffmpeg的支持，用mpp是最好的，官方也只管这个，mpp的解码器需要类型需要自己配置，也并没有支持rtsp取流的，自己rtsp取流后把packet送给mpp去解码

作者: zhouzhouzlove **时间:** 2019-8-23 14:13

> kiwi 发表于 2019-8-23 14:10
> 建议别捣鼓ffmpeg了，他们也只是对接了api，据说是有bug的，也不会提供ffmpeg的支持，用mpp是最好的，官 ...

我现在在看mpp开发手册，感觉需要例子，也正在网上找，大神有没有小例子，我想这样可能会快点，随便硬解码的编解码例子都行，我看下流程和各个api的使用。
当然，工作中的内容涉及保密的话，就算了。。。。
还是希望能够得到你的帮助，，，，，

作者: kiwi **时间:** 2019-8-23 14:16

> zhouzhouzlove 发表于 2019-8-23 14:13
> 我现在在看mpp开发手册，感觉需要例子，也正在网上找，大神有没有小例子，我想这样可能会快点，随便硬解 ...

https://github.com/rockchip-linux/mpp
里面有相应的例子，结合mpp开发文档，容易理解

作者: zhouzhouzlove **时间:** 2019-8-23 14:40

> kiwi 发表于 2019-8-23 14:16
> https://github.com/rockchip-linux/mpp
> 里面有相应的例子，结合mpp开发文档，容易理解 ...

好的，多谢~
么么哒~~~~~

作者: zhouzhouzlove **时间:** 2019-8-23 17:43
本帖最后由 zhouzhouzlove 于 2019-8-23 19:40 编辑

大神，我安装mpp的步骤是这样的，请看下是是否正确：
1、瑞芯微github下载mpp，
2、修改mpp/build/linux/aarch64/arm.linux.cross.cmake,修改了gcc和g++还有armv8-a这三条。修改完后，执行同目录下的make-Makefiles.bash，然后make ，sudo make install。

```
SET(CMAKE_SYSTEM_NAME Linux)
SET(CMAKE_C_COMPILER "aarch64-linux-gnu-gcc")
#SET(CMAKE_C_COMPILER "arm-linux-gnueabi-gcc")
SET(CMAKE_CXX_COMPILER "aarch64-linux-gnu-g++")
#SET(CMAKE_CXX_COMPILER "arm-linux-gnueabi-g++")
SET(CMAKE_SYSTEM_PROCESSOR "armv8-a")
#SET(CMAKE_SYSTEM_PROCESSOR "armv7-a_hardfp")
```

配置时有警告：
CMake Warning:
Manually-specified variables were not used by the project:

RKPLATFORM
3、默认的安装路径是/usr/local/，修改/etc/profile，添加环境变量，并且执行sudo ldconfig使之生效

```
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib
export LD_LIBRARY_PATH

export PATH =$PATH:/usr/local/bin

C_INCLUDE_PATH=$C_INCLUDE_PATH:/usr/local/include/rockchip
export C_INCLUDE_PATH

CPLUS_INCLUDE_PATH=$CPLUS_INCLUDE_PATH:/usr/local/include/rockchip
export CPLUS_INCLUDE_PATH
```

4、执行mpi_dec_test程序，提示信息正常，当下载了h264文件进行测试时，sudo mpi_dec_test -t 7 -i ~/128x128.264 -n 10
输出如下：

```
mpi_dec_test: cmd parse result:
mpi_dec_test: input  file name: /home/perfxlab/128x128.264
mpi_dec_test: output file name:
mpi_dec_test: config file name:
mpi_dec_test: width      :    0
mpi_dec_test: height     :    0
mpi_dec_test: type       : 7
mpi_dec_test: debug flag : 0
mpi_dec_test: max frames : 10
mpi_dec_test: mpi_dec_test start
mpi_dec_test: input file size 174416
mpi_dec_test: mpi_dec_test decoder test start w 0 h 0 type 7
mpi: mpp version: Without VCS info
mpp_rt: NOT found ion allocator
mpp_rt: found drm allocator
h264d_dpb: dpb_size error.
mpi_dec_test: decode_get_frame get info changed found
mpi_dec_test: decoder require buffer w:h [128:128] stride [128:128] buf_size 32768
mpi_dec_test: decode_get_frame get frame 1
mpi_dec_test: decode_get_frame get frame 2
mpi_dec_test: decode_get_frame get frame 3
mpi_dec_test: decode_get_frame get frame 4
mpi_dec_test: decode_get_frame get frame 5
mpi_dec_test: decode_get_frame get frame 6
mpi_dec_test: decode_get_frame get frame 7
mpi_dec_test: decode_get_frame get frame 8
mpi_dec_test: decode_get_frame get frame 9
```

mpi_dec_test: decode_get_frame get frame 10
mpi_dec_test: reach max frame number 10
mpi_dec_test: test success max memory 0.16 MB

请问，这个mpp库的安装有问题吗？rochip-ffmpeg的安装过程有没有详细教程？

作者: jefferyzhang **时间:** 2019-8-25 17:37

> zhouzhouzlove 发表于 2019-8-23 17:43
> 大神，我安装mpp的步骤是这样的，请看下是是否正确：
> 1、瑞芯微github下载mpp，
> 2、修改mpp/build/linux/aa ...

看过去编译是没问题的，都能正常跑了。
没有要改源码的话可以用我们dnf安装即可。

作者: zhouzhouzlove **时间:** 2019-8-26 09:13

> jefferyzhang 发表于 2019-8-25 17:37
> 看过去编译是没问题的，都能正常跑了。
> 没有要改源码的话可以用我们dnf安装即可。 ...

在上面的解码的时候log里，有一个报错：

h264d_dpb: dpb_size error.

这在mpp开发参考（0.3版）.pdf里的截图中是没有的。

作者: jefferyzhang **时间:** 2019-8-26 09:31

> zhouzhouzlove 发表于 2019-8-26 09:13
> 在上面的解码的时候log里，有一个报错：
>
> h264d_dpb: dpb_size error.

没花屏的话无所谓
说的是码流中 vui 信息与前面码流的 dpb 信息不一致
这跟编译没关系，是你的码流告警的，不用在意

作者: zhouzhouzlove **时间:** 2019-8-28 17:59
mpi_dec_test
这个程序如果指定了输出文件，文件的类型是什么？我看好像不能指定输出类型。

作者: jefferyzhang **时间:** 2019-8-29 08:16

> zhouzhouzlove 发表于 2019-8-28 17:59
> mpi_dec_test
> 这个程序如果指定了输出文件，文件的类型是什么？我看好像不能指定输出类型。
> ...

默认是NV12，可以设置输出rgb我记得。具体看下help，我也记不清楚了。

作者: zhouzhouzlove **时间:** 2019-8-29 10:51

> jefferyzhang 发表于 2019-8-29 08:16
> 默认是NV12，可以设置输出rgb我记得。具体看下help，我也记不清楚了。

斑竹辛苦了，这个属性，是否就是制定了默认的输出类型？
我看了代码，是输入命令-f后面的参数。但是这个参数非必须，那么默认值就是MPP_FMT_BUTT，但是这个格式没有理解具体含义，不像yuv和rgb那样容易理解。还希望能指导一下。

另外，解码时，分了两种情况，分别使用decode_simple和decode_advanced两种流程，选择依据是 cmd->simple = (cmd->type != MPP_VIDEO_CodingMJPEG) ? (1) : (0);
意为如果输入文件的类型-t 的参数是MPP_VIDEO_CodingMJPEG，编号为4，那么simple=0，调用decode_advanced进行解码；
如果输入文件的类型不是MPP_VIDEO_CodingMJPEG，如h264文件为-t 7，那么simple=1，调用decode_simple进行解码。
我没有理解为什么要分两种情况，是否可以简单指导一下？

貌似这个帖子就我最笨，问的问题也比较low，给斑竹增加了工作量，深感抱歉，但是也是无奈之举。
[attach]518[/attach]

作者: zhouzhouzlove **时间:** 2019-8-29 10:55

> kiwi 发表于 2019-8-23 14:10
> 建议别捣鼓ffmpeg了，他们也只是对接了api，据说是有bug的，也不会提供ffmpeg的支持，用mpp是最好的，官 ...

大佬好，我看了mpi_dec_test的源码，好像确实是没有解析rtsp协议得到h264裸流数据包的功能。
你看这样想行不行，利用ffmpeg的接口进行解协议，从rtsp的网址得到得到avpacket，即从网络流中得到h264的数据包。
然后再把这个avpacket送给mpp。

但是问题就是怎么给，是把这两个库的packet结构体进行比较，然后对应的赋值吗？比如都有pts成员。

可否简单的提点我一下怎么喂数据。。。。。。。。

作者: jefferyzhang **时间:** 2019-8-29 12:41

> zhouzhouzlove 发表于 2019-8-29 10:55
> 大佬好，我看了mpi_dec_test的源码，好像确实是没有解析rtsp协议得到h264裸流数据包的功能。
> 你看这样想 ...

mpp_dec_test一直就在讲怎么喂数据，h264数据放package喂进去就可以了，并不复杂。
整个mpp库接口都是仿照ffmpeg设计的。
好好看下源码，所有源码都开放了，sample都有了，又来问别人怎么写代码，这就很不合适了吧。

作者: zhouzhouzlove **时间:** 2019-8-29 13:57

> jefferyzhang 发表于 2019-8-29 12:41
> mpp_dec_test一直就在讲怎么喂数据，h264数据放package喂进去就可以了，并不复杂。
> 整个mpp库接口都是仿 ...

我错了，大佬，，，，，，，多谢回复，正在看。

作者: zhouzhouzlove **时间:** 2019-8-31 09:57
本帖最后由 zhouzhouzlove 于 2019-8-31 16:51 编辑

楼主大神，是的，没错，我又来了。。。。。。。。
是这样的，我运行mpi_dec_test时，运行的时候，cpu占用率是处于比较高的水平的，cpu5已经达到了70%。

当运行mpi_dec_multi_test时，程序报错，当然，这可能是维护未更新的问题，但是从cpu占用率来看，三个cpu的占用率加起来也是达到了70%。
mpi: mpp version: 3d35398 author: Johnson Ding [jpege]: Fix jpeg encoder stride problem
[attach]528[/attach]

得到的文件时yuv420sp格式，用yuv播放器确实可以看到一幅幅的画面。
这是正常的吗？按理说，mpp肯定是调用了硬件解码，应该比较少才对，感觉怎么着也得降到10%吧。。。。。

作者: jefferyzhang **时间:** 2019-9-1 00:37

> zhouzhouzlove 发表于 2019-8-31 09:57
> 楼主大神，是的，没错，我又来了。。。。。。。。
> 是这样的，我运行mpi_dec_test时，运行的时候，cpu占用率 ...

正不正常放到项目里看。
你这里一帧一个print，cpu低于10%都不可能，还不限帧率，你这里应该解码都有好几百fps了，怎么可能cpu在10%以内。

作者: 15992605143 **时间:** 2019-9-3 00:40
兼容MPP的ffmpeg解码出来的帧格式是：AV_PIX_FMT_DRM_PRIME，我尝试使用libswscale库转成RGB24，
结果该库不认识这个格式，报错退出了。请问AV_PIX_FMT_DRM_PRIME这是什么格式，怎样才能转换成RGB24?

作者: jefferyzhang **时间:** 2019-9-3 08:02

> 15992605143 发表于 2019-9-3 00:40
> 兼容MPP的ffmpeg解码出来的帧格式是：AV_PIX_FMT_DRM_PRIME，我尝试使用libswscale库转成RGB24，
> 结果该库 ...

我不是很懂ffmpeg，我从mpp角度说下。
一般来说H.264压缩的是YUV编码，解压出来的一般来说就是YUV NV12格式的帧。

作者: kiwi **时间:** 2019-9-4 21:03

> 15992605143 发表于 2019-9-3 00:40
> 兼容MPP的ffmpeg解码出来的帧格式是：AV_PIX_FMT_DRM_PRIME，我尝试使用libswscale库转成RGB24，
> 结果该库 ...

没听过FFmpeg有这个色彩空间格式，你应该是搞错了

作者: kiwi **时间:** 2019-9-4 21:06

> zhouzhouzlove 发表于 2019-8-29 10:55
> 大佬好，我看了mpi_dec_test的源码，好像确实是没有解析rtsp协议得到h264裸流数据包的功能。
> 你看这样想 ...

这样做是可以的，看看MPP文档中的MppPacket的用法，就直接把packet喂进去就行

作者: zhouzhouzlove **时间:** 2019-9-5 14:18
本帖最后由 zhouzhouzlove 于 2019-9-5 14:47 编辑

> kiwi 发表于 2019-9-4 21:06
> 这样做是可以的，看看MPP文档中的MppPacket的用法，就直接把packet喂进去就行 ...

我目前这样做了，目前情况如下：

我是在mpi_dec_test的基础上做的修改，增加了ffmpeg的功能，添加了ffmpeg头文件和库后，可以编译并运行，并且将解码后的yuv数据写成了文件，用yuv播放器可以看到一帧帧的图像，正常。

由于我是使用了mpp库的编译流程，所以，在想着把流程简化出来，使用安装好的mpp动态库和头文件来编译程序。
卡在了一个地方，

```
buf = mpp_malloc(char, packet_size);                    //这一步失败，我不知道怎么引用头文件和库，才能调用到这个函数。
//buf = malloc(packet_size);
ret = mpp_packet_init(&packet, buf, packet_size);               //第一步，初始化MPP的packet
```

在库目录下grep，发现没有这个函数，但是有它调用的mpp_osal_malloc
做了实验，如果直接换成malloc。那么解码的时候会报错——也不是报错，就是会一卡一卡的，导致数据丢失。yuv文件会不连续。
求大佬和楼主给指导一下，我尝试了将 mpp_malloc的函数实现直接写到程序里，但是这样会引入更多的函数和宏，感觉不太现实，我想知道怎么解决这个问题。

作者: zhouzhouzlove **时间:** 2019-9-5 14:21
本帖最后由 zhouzhouzlove 于 2019-9-5 14:31 编辑

> jefferyzhang 发表于 2019-9-1 00:37
> 正不正常放到项目里看。
> 你这里一帧一个print，cpu低于10%都不可能，还不限帧率，你这里应该解码都有好几 ...

大佬，我现在基本上流程走通了，就只剩下一个问题：
如何使用mpp的动态库和头文件编译程序，之前是直接将程序添加到mpp的编译工程里的。
在直接将ffmpeg的packet中的data和datasize给到mpp时，运行时报错如下：但是经过检查解码的yuv文件，图像并无问题。
mpp_packet: Assertion p->data <= p->pos failed at mpp_packet_set_pos:196
mpp_packet: Assertion p->size >= p->length failed at mpp_packet_set_pos:197

发现mpp_log和mpp_err，msleep(已通过将宏定义直接引入解决)、 mpp_malloc()无法使用。报错未定义的引用。
前面的log可以用其他办法实现，但是最后的一个 mpp_malloc（）却没有通过引入函数实现来解决，请问可否指导一下?

作者: jefferyzhang **时间:** 2019-9-5 15:26

> zhouzhouzlove 发表于 2019-9-5 14:21
> 大佬，我现在基本上流程走通了，就只剩下一个问题：
> 如何使用mpp的动态库和头文件编译程序，之前是直接将 ...

看不懂你在说什么。。。。

作者: zhouzhouzlove **时间:** 2019-9-5 15:45

> jefferyzhang 发表于 2019-9-5 15:26
> 看不懂你在说什么。。。。

抱歉，我的描述有问题：
是这样的，一开始我不太熟悉怎么使用mpp，所以就直接在mpp库自带的mpi_dec_test.c上修改，然后直接编译mpp库，那么这个测试程序也会被相应的编译。

我在里面，去掉读h264文件的操作，增加了ffmpeg读取rtsp流的操作，将ffmpeg解出来的h264数据直接喂给了mpp解码器。
然后编译mpp库，得到了这个新的mpi_dec_test程序，经过测试，可以从rtsp流解码到yuv数据。

现在是想着，不套用mpp库的编译流程，而是自己写个单独的程序，来调用安装好的mpp头文件和so库，结果发现有一个函数，在mpp库中没有存在，即mpp_malloc,如果没有这个函数，那么mpp_packet_init也不能正确执行，所以想问下，这个函数我需要怎么调用到，不知道我说明白了没有，。。。。。。

```
buf = mpp_malloc(char, packet_size);                    //这一步失败，我不知道怎么引用头文件和库，才能调用到这个函数。
//buf = malloc(packet_size);
ret = mpp_packet_init(&packet, buf, packet_size);               //第一步，初始化MPP的packet
```

作者: jefferyzhang **时间:** 2019-9-5 18:06

> zhouzhouzlove 发表于 2019-9-5 15:45
> 抱歉，我的描述有问题：
> 是这样的，一开始我不太熟悉怎么使用mpp，所以就直接在mpp库自带的mpi_dec_test. ...

我没用过这个接口，为啥会用这个接口分配buffer?
demo里有这么用的么?

作者: zhouzhouzlove **时间:** 2019-9-5 19:21

> jefferyzhang 发表于 2019-9-5 18:06
> 我没用过这个接口，为啥会用这个接口分配buffer?
> demo里有这么用的么?

哈哈哈，终于问了个有意义的问题，demo里面确实有，

打开mpp的编译verbose=1后，得到这个文件的编译命令如下：

```
/usr/bin/aarch64-linux-gnu-g++
  -O3 -DNDEBUG
/mpp-release/build/linux/aarch64/test/CMakeFiles/mpi_dec_test.dir/mpi_dec_test.c.o
/mpp-release/build/linux/aarch64/test/CMakeFiles/mpi_dec_test.dir/mpp_event_trigger.c.o
/mpp-release/build/linux/aarch64/test/CMakeFiles/mpi_dec_test.dir/mpp_parse_cfg.c.o
-o mpi_dec_test
-L/usr/local/lib -rdynamic /mpp-release/build/linux/aarch64/utils/libutils.a
/mpp-release/build/linux/aarch64/mpp/librockchip_mpp.so.0
/mpp-release/build/linux/aarch64/mpp/librockchip_mpp_static.a
/mpp-release/build/linux/aarch64/mpp/codec/libmpp_codec.a
/mpp-release/build/linux/aarch64/mpp/codec/dec/avs/libcodec_avsd.a
/mpp-release/build/linux/aarch64/mpp/codec/dec/h263/libcodec_h263d.a
/mpp-release/build/linux/aarch64/mpp/codec/dec/h264/libcodec_h264d.a
/mpp-release/build/linux/aarch64/mpp/codec/dec/h265/libcodec_h265d.a
/mpp-release/build/linux/aarch64/mpp/codec/dec/m2v/libcodec_mpeg2d.a
/mpp-release/build/linux/aarch64/mpp/codec/dec/mpg4/libcodec_mpeg4d.a
/mpp-release/build/linux/aarch64/mpp/codec/dec/vp8/libcodec_vp8d.a
/mpp-release/build/linux/aarch64/mpp/codec/dec/vp9/libcodec_vp9d.a
/mpp-release/build/linux/aarch64/mpp/codec/dec/jpeg/libcodec_jpegd.a
/mpp-release/build/linux/aarch64/mpp/codec/enc/h264/libcodec_h264e.a
/mpp-release/build/linux/aarch64/mpp/codec/libmpp_rc.a
/mpp-release/build/linux/aarch64/mpp/codec/enc/jpeg/libcodec_jpege.a
/mpp-release/build/linux/aarch64/mpp/codec/enc/h265/libcodec_h265e.a
/mpp-release/build/linux/aarch64/mpp/codec/enc/vp8/libcodec_vp8e.a
/mpp-release/build/linux/aarch64/mpp/codec/enc/dummy/libcodec_dummy_enc.a
/mpp-release/build/linux/aarch64/mpp/codec/dec/dummy/libcodec_dummy_dec.a
/mpp-release/build/linux/aarch64/mpp/hal/libmpp_hal.a
/mpp-release/build/linux/aarch64/mpp/hal/rkdec/h264d/libhal_h264d.a
/mpp-release/build/linux/aarch64/mpp/hal/libmpp_hal.a
/mpp-release/build/linux/aarch64/mpp/hal/rkdec/h264d/libhal_h264d.a
/mpp-release/build/linux/aarch64/mpp/hal/rkdec/avsd/libhal_avsd.a
/mpp-release/build/linux/aarch64/mpp/hal/vpu/h263d/libhal_h263d.a
/mpp-release/build/linux/aarch64/mpp/hal/rkdec/h265d/libhal_h265d.a
/mpp-release/build/linux/aarch64/mpp/hal/vpu/m2vd/libhal_mpeg2d.a
/mpp-release/build/linux/aarch64/mpp/hal/vpu/mpg4d/libhal_mpeg4d.a
/mpp-release/build/linux/aarch64/mpp/hal/vpu/vp8d/libhal_vp8d.a
/mpp-release/build/linux/aarch64/mpp/hal/rkdec/vp9d/libhal_vp9d.a
/mpp-release/build/linux/aarch64/mpp/hal/vpu/jpegd/libhal_jpegd.a
/mpp-release/build/linux/aarch64/mpp/hal/common/h264/libhal_h264e.a
/mpp-release/build/linux/aarch64/mpp/hal/rkenc/h264e/libhal_h264e_rkv.a
/mpp-release/build/linux/aarch64/mpp/hal/vpu/h264e/libhal_h264e_vpu.a
/mpp-release/build/linux/aarch64/mpp/hal/common/h264/libhal_h264e.a
/mpp-release/build/linux/aarch64/mpp/hal/rkenc/h264e/libhal_h264e_rkv.a
/mpp-release/build/linux/aarch64/mpp/hal/vpu/h264e/libhal_h264e_vpu.a
/mpp-release/build/linux/aarch64/mpp/hal/common/libhal_common.a
```

/mpp-release/build/linux/aarch64/mpp/hal/vpu/jpege/libhal_jpege.a
/mpp-release/build/linux/aarch64/mpp/hal/rkenc/h265e/libhal_h265e.a
/mpp-release/build/linux/aarch64/mpp/hal/vpu/vp8e/libhal_vp8e.a
/mpp-release/build/linux/aarch64/mpp/hal/dummy/libhal_dummy.a
/mpp-release/build/linux/aarch64/mpp/hal/worker/mpp_device/libmpp_device.a
/mpp-release/build/linux/aarch64/mpp/vproc/libmpp_vproc.a
/mpp-release/build/linux/aarch64/mpp/vproc/rga/libvproc_rga.a
/mpp-release/build/linux/aarch64/mpp/base/libmpp_base.a
/mpp-release/build/linux/aarch64/osal/libosal.a
-lpthread
/mpp-release/build/linux/aarch64/mpp/vproc/iep/libvproc_iep.a
-Wl,-rpath,/mpp-release/build/linux/aarch64/mpp

[attach]545[/attach]

作者: zhouzhouzlove **时间:** 2019-9-5 19:24
本帖最后由 zhouzhouzlove 于 2019-9-5 19:33 编辑

请删除这一层，刚才论坛出bug了，无法回帖，现在又突然全部都出来了。。。。。

作者: zhouzhouzlove **时间:** 2019-9-5 19:26
论坛有bug，我现在无法直接回复你的问题（提交不了），只能重新来一层
确实是的demo里面的用法
[attach]547[/attach]

作者: zhouzhouzlove **时间:** 2019-9-5 19:28
论坛有bug，我现在无法直接回复你的问题（提交不了），只能重新来一层
确实是的demo里面的用法
详见demo的538行，我无法发送图片，论坛有bug

作者: zhouzhouzlove **时间:** 2019-9-5 19:34
你们的论坛有bug。。。。。。。。。。请删除重复的楼层、。、、、、、、、、
要不整的我跟个二愣子一样。。。。。

作者: kiwi **时间:** 2019-9-5 21:47

zhouzhouzlove 发表于 2019-9-5 14:18
我目前这样做了，目前情况如下：

我是在mpi_dec_test的基础上做的修改，增加了ffmpeg的功能，添加了ffmpe ...

mpp_malloc就是申请一块内存罢了，用malloc也是一样的，二者没有区别

作者: zhouzhouzlove **时间:** 2019-9-6 09:50

kiwi 发表于 2019-9-5 21:47
mpp_malloc就是申请一块内存罢了，用malloc也是一样的，二者没有区别

当直接把ffmpeg的packet中的data给到mpp的packet的时候，是否遇见过这个报错：
mpp_packet: Assertion p->data <= p->pos failed at mpp_packet_set_pos:196
mpp_packet: Assertion p->data <= p->pos failed at mpp_packet_set_pos:196
mpp_packet: Assertion p->data <= p->pos failed at mpp_packet_set_pos:196
mpp_packet: Assertion p->data <= p->pos failed at mpp_packet_set_pos:196
mpp_packet: Assertion p->size >= p->length failed at mpp_packet_set_pos:197
mpp_packet: Assertion p->data <= p->pos failed at mpp_packet_set_pos:196
mpp_packet: Assertion p->data <= p->pos failed at mpp_packet_set_pos:196
mpp_packet: Assertion p->size >= p->length failed at mpp_packet_set_pos:197

作者: swlmx **时间:** 2019-9-10 14:33
楼主好。我在Android环境下用java的MediaCodec API读取YUV编码h.264，然后demsg显示要20毫秒编码一帧，达不到同时两路1080p@30fps呢？

下面是编码的代码

```
1.
2. public void encode(byte[] yuvFrame){
3.              int inputBufferIndex = codec.dequeueInputBuffer(-1);
4.
5.              if (inputBufferIndex >= 0) {
6.                    ByteBuffer inputBuffer;
7.                    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) {
8.                          inputBuffer = codec.getInputBuffer(inputBufferIndex);
9.                    } else {
10.                         inputBuffer = codec.getInputBuffers()[inputBufferIndex];
11.                   }
12.                   pts = computePresentationTime(generateIndex);
13.                   if (inputBuffer != null) {
14.                         inputBuffer.clear();
15.                         inputBuffer.put(yuvFrame, 0, yuvFrame.length);
16.                         codec.queueInputBuffer(inputBufferIndex, 0, yuvFrame.length, pts, 0);
17.                   }
18.                   generateIndex += 1;
19.             }
20.             MediaCodec.BufferInfo bufferInfo = new MediaCodec.BufferInfo();
21.
22.             int outputBufferIndex = codec.dequeueOutputBuffer(bufferInfo, DEFAULT_TIMEOUT_US);
23.
24.             while (outputBufferIndex >= 0) {
25.                   ByteBuffer outputBuffer;
26.                   if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) {
27.                         outputBuffer = codec.getOutputBuffer(outputBufferIndex);
28.                   } else {
29.                         outputBuffer = codec.getOutputBuffers()[outputBufferIndex];
30.                   }
31.                   if (outputBuffer != null) {
32.                         byte[] outData = new byte[bufferInfo.size];
33.                         outputBuffer.get(outData);
34.                         if (bufferInfo.flags == 2) {
35.                               configByte = outData;
36.                         } else if (bufferInfo.flags == 1) {
37.                               byte[] keyframe = new byte[bufferInfo.size + configByte.length];
38.                               System.arraycopy(configByte, 0, keyframe, 0, configByte.length);
39.                               System.arraycopy(outData, 0, keyframe, configByte.length, outData.length);
40.
41.                               if (null != callback) {
42.                                     callback.onFrame(this, keyframe);
43.                               }
44.                         } else {
45.                               if (null != callback) {
46.                                     callback.onFrame(this, outData);
47.                               }
48.                         }
49.
50.                         codec.releaseOutputBuffer(outputBufferIndex, false);
51.                   }
52.                   outputBufferIndex = codec.dequeueOutputBuffer(bufferInfo, DEFAULT_TIMEOUT_US);
53.             }
54.       }
55.
56.       /**
57.          * Generates the presentation time for frame N, in microseconds.
58.          */
59.       private long computePresentationTime(long frameIndex) {
60.             return frameIndex * 1000000 / FPS;
61.       }
62.
```

复制代码

作者: jefferyzhang **时间:** 2019-9-11 08:18

swlmx 发表于 2019-9-10 14:33
楼主好。我在Android环境下用java的MediaCodec API读取YUV编码h.264，然后demsg显示要20毫秒编码一帧，达不 ...

编码在VPU跑 300MHz情况下只能一路1080p30，
跑2路需要将频率提到400MHz或者更高

作者: swlmx **时间:** 2019-9-12 15:54

jefferyzhang 发表于 2019-9-11 08:18
编码在VPU跑 300MHz情况下只能一路1080p30，
跑2路需要将频率提到400MHz或者更高 ...

那怎么提升VPU频率呢

作者: jefferyzhang **时间:** 2019-9-13 11:23

swlmx 发表于 2019-9-12 15:54
那怎么提升VPU频率呢

需要修改kernel，不在这贴里讨论，有需求请单独开一个帖子问下。
我会去问下我们驱动人员能否在下一版直接测试和发布400M的kernel，

并且我们不会在Toybrick开发板上建议任何的抬频抬压操作，这种操作没经过压力测试和稳定性测试我们也不敢对外发布，如果自行抬压抬频造成的设备损坏是无法进行保修的。

作者: shopping **时间:** 2019-9-24 10:05

kiwi 发表于 2019-8-23 14:10
建议别捣鼓ffmpeg了，他们也只是对接了api，据说是有bug的，也不会提供ffmpeg的支持，用mpp是最好的，官 ...

你好，您这个意思是不是 MPP 不能直接调用摄像头，我还需要用其他库 取rtsp流，然后再喂进 mpp 进行解码？如果是上面这样，请问您用的是上面库取流，能否告知，我学习一下。

作者: swlmx **时间:** 2019-9-24 10:09
4. you can get demo about mpp applied to linux and android.
Liunx : https://github.com/WainDing/mpp_linux_cpp

Android : https://github.com/c-xh/MediaCodecDecodeMulti_h264

安卓的失效了，能重新放一个demo吗

作者: jefferyzhang **时间:** 2019-9-24 10:38

> swlmx 发表于 2019-9-24 10:09
> 4. you can get demo about mpp applied to linux and android.
> Liunx : https://github.com/WainDing ...

https://github.com/c-xh/mediacodec_gl_decode_multi_h264_file

作者: swlmx **时间:** 2019-9-24 17:12

> jefferyzhang 发表于 2019-9-24 10:38
> https://github.com/c-xh/mediacodec_gl_decode_multi_h264_file

好的 谢谢

作者: shopping **时间:** 2019-9-25 11:02
楼主你好，看了贵司的 mpp开发参考 这一文档，发现其并未提及到用哪个接口调用摄像头。那我要怎样获取摄像头的码流，双路USB摄像头。

作者: jefferyzhang **时间:** 2019-9-25 11:39

> shopping 发表于 2019-9-25 11:02
> 楼主你好，看了贵司的 mpp开发参考 这一文档，发现其并未提及到用哪个接口调用摄像头。那我要怎样获取摄像 ...

mpp是硬件编解码，跟usb摄像头没有任何关系。
linux获取usb摄像头走的是v4l2框架，建议你先百度学习下。。。。

作者: kiwi **时间:** 2019-9-25 19:30

> shopping 发表于 2019-9-25 11:02
> 楼主你好，看了贵司的 mpp开发参考 这一文档，发现其并未提及到用哪个接口调用摄像头。那我要怎样获取摄像 ...

usb摄像头可以不用mpp，直接用v4l2框架取yuv视频数据就行

作者: shopping **时间:** 2019-9-26 09:45

> kiwi 发表于 2019-9-25 19:30
> usb摄像头可以不用mpp，直接用v4l2框架取yuv视频数据就行

嗯，已经在看这个V4L2框架了，大把的bug。

作者: shopping **时间:** 2019-10-8 14:17
本帖最后由 shopping 于 2019-10-8 14:23 编辑

> kiwi 发表于 2019-9-25 19:30
> usb摄像头可以不用mpp，直接用v4l2框架取yuv视频数据就行

你好，按您的说法，不用MPP库而直接用V4L2框架解码，速度上二者差异大不大，您有试验过吗？目前我VAL2+RGA demo 已经写好了，本来想在中间插一个MPP解码，以为这样速度会更快地。

作者: 15992605143 **时间:** 2019-10-18 14:43
MPP_DEC_SET_IMMEDIATE_OUT: （仅限解码）
立即输出模式（不建议开启），如果未开立即输出模式，MPP会按预先设定的节奏间隔输出解码的帧（例如33ms输出一帧）
请问这个33ms的时间间隔是如何设置的?

作者: jefferyzhang **时间:** 2019-10-18 15:06

> shopping 发表于 2019-10-8 14:17
> 你好，按您的说法，不用MPP库而直接用V4L2框架解码，速度上二者差异大不大，您有试验过吗？目前我VAL2+RGA ...

请搞清楚你数据源的数据格式是什么。
V4L2和MPP没有任何关系。

作者: shopping **时间:** 2019-10-18 15:11

> jefferyzhang 发表于 2019-10-18 15:06
> 请搞清楚你数据源的数据格式是什么。
> V4L2和MPP没有任何关系。

这是以前没有了解时候问的，见笑了。

作者: jefferyzhang **时间:** 2019-10-18 15:19

> 15992605143 发表于 2019-10-18 14:43
> MPP_DEC_SET_IMMEDIATE_OUT: （仅限解码）
> 立即输出模式（不建议开启），如果未开立即输出模式，MPP会按预 ...

这个是mpp内部sleep的。按照pps sps的帧率算的

作者: jefferyzhang **时间:** 2019-10-18 15:20

> shopping 发表于 2019-10-18 15:11
> 这是以前没有了解时候问的，见笑了。

哦，以前看漏了，现在才看到这问题。
以后有问题还是单独发帖子问比较好

作者: swlmx **时间:** 2019-10-24 13:50

> jefferyzhang 发表于 2019-5-17 11:05
> mpp的build目录里有安卓编译。
> 我们的android也是自带libmpp的，无需编译。
> android的media_codec默认就 ...

楼主，我现在想解码 Mjpeg，但是 mediacodec api 好像不支持 video/mjpeg。
应用层又无法通过 System.loadLibrary() 载入 libmpp (报错 E/linker: library "/vendor/lib/libmpp.so" ("/vendor/lib/libmpp.so") needed or dlopened by "/system/lib/libnativeloader.so" is not accessible for the namespace)。看起来android N开始应用层无法载入系统私有库。
所以要怎么测试解码mjpeg?

作者: jefferyzhang **时间:** 2019-10-24 18:04

> swlmx 发表于 2019-10-24 13:50
> 楼主，我现在想解码 Mjpeg，但是 mediacodec api 好像不支持 video/mjpeg。
> 应用层又无法通过 System.loa ...

可以直接用安卓jpeg loader来读，底层我记得是对接过硬件的（大于多少像素会用VPU解），具体我也不是很清楚该版本有没有。
当然不放心的话直接调用mpp来做路径是最短的

作者: swlmx **时间:** 2019-10-30 11:20

> jefferyzhang 发表于 2019-10-24 18:04
> 可以直接用安卓jpeg loader来读，底层我记得是对接过硬件的（大于多少像素会用VPU解），具体我也不是很清 ...

libmpp.so是armv7的？1.3固件。
我用v8a的库链接报错

作者: jefferyzhang **时间:** 2019-10-30 11:43

> swlmx 发表于 2019-10-30 11:20
> libmpp.so是armv7的？1.3固件。
> 我用v8a的库链接报错

toybrick里带的都是64bit的

作者: swlmx **时间:** 2019-11-5 10:40

> jefferyzhang 发表于 2019-10-30 11:43
> toybrick里带的都是64bit的

我yuv编码jpeg出现 jpege_api: jpege: hardware return error status 40 。存下来的图底部是花的，是哪里出了问题？代码如下

```
1.
2. void doCodec(){
3.       MppCtx ctx;
4.       MppApi *mpi;
5.       MpiCmd cmd = MPP_SET_OUTPUT_BLOCK;
6.       MppParam param          = NULL;
7.       MppPollType block     = MPP_POLL_BLOCK;
8.       MppCodingType type = MPP_VIDEO_CodingMJPEG;
9.       int MPI_ENC_LOOP_COUNT = 1;
10.      __android_log_print(ANDROID_LOG_DEBUG, "JNI","mpp_create");
11.
12.      if(MPP_OK != mpp_create(&ctx, &mpi)){
13.              __android_log_print(ANDROID_LOG_DEBUG, "JNI","mpp_create failed");
14.              goto MPP_TEST_FAILED;
15.      }
16.      __android_log_print(ANDROID_LOG_DEBUG, "JNI","mpp_init");
17.
```

```
    if(MPP_OK != mpp_init(ctx, MPP_CTX_ENC, type)){
            __android_log_print(ANDROID_LOG_DEBUG, "JNI","mpp_init failed");
            goto MPP_TEST_FAILED;
    }

    MppEncRcCfg rcCfg;


    __android_log_print(ANDROID_LOG_DEBUG, "JNI","rcCfg");

    rcCfg.change = MPP_ENC_RC_CFG_CHANGE_ALL;
    rcCfg.rc_mode = (MppEncRcMode) MPP_ENC_RC_MODE_CBR;
    rcCfg.quality = (MppEncRcQuality)MPP_ENC_RC_QUALITY_BEST;
    rcCfg.bps_target = bps;
    rcCfg.bps_max = bps * 17 / 16;
    rcCfg.bps_min = bps * 15 / 16;
    rcCfg.fps_in_flex         = 0;
    rcCfg.fps_in_num          = 24;
    rcCfg.fps_in_denorm      = 1;
    rcCfg.fps_out_flex        = 0;
    rcCfg.fps_out_num         = 24;
    rcCfg.fps_out_denorm     = 1;
    rcCfg.gop = 50;
    if(MPP_OK != mpi->control(ctx, MPP_ENC_SET_RC_CFG, &rcCfg)){
            __android_log_print(ANDROID_LOG_DEBUG, "JNI","rcCfg failed");
            goto MPP_TEST_FAILED;
    }

    __android_log_print(ANDROID_LOG_DEBUG, "JNI","prepCfg");

    MppEncPrepCfg prepCfg;
    prepCfg.change = MPP_ENC_PREP_CFG_CHANGE_INPUT | MPP_ENC_PREP_CFG_CHANGE_FORMAT;
    prepCfg.width = 1920;
    prepCfg.height = YUV_HEIGHT;
    prepCfg.hor_stride = MPP_ALIGN(1920, 8);
    prepCfg.ver_stride = MPP_ALIGN(YUV_HEIGHT, 8);
    prepCfg.format = MPP_FMT_YUV420P;

    if(MPP_OK != mpi->control(ctx, MPP_ENC_SET_PREP_CFG, &prepCfg)){
            __android_log_print(ANDROID_LOG_DEBUG, "JNI","rcCfg failed");
            goto MPP_TEST_FAILED;
    }

    __android_log_print(ANDROID_LOG_DEBUG, "JNI","codecCfg");
    MppEncCodecCfg codecCfg;


    if(MPP_OK != mpi->control(ctx, MPP_ENC_SET_CODEC_CFG, &codecCfg)){
            __android_log_print(ANDROID_LOG_DEBUG, "JNI","codecCfg failed");
            goto MPP_TEST_FAILED;
    }


    MppPacket enc_out;
    MppFrame enc_in;

    param = &block;

    MppBuffer frm_buf;
    unsigned char *buf2;
    mpp_buffer_get(NULL, &frm_buf, MPP_ALIGN(1920, 8) * MPP_ALIGN(YUV_HEIGHT, 8) * 3 / 2);


    FILE *file;
    file = fopen("/sdcard/single_frame.yuv", "rb");
    buf2 = (unsigned char *)malloc(YUV_SIZE);
    memset(buf2, 0, YUV_SIZE);

    fread(buf2, 1, YUV_SIZE, file);



    if(MPP_OK != mpi->control(ctx, MPP_SET_OUTPUT_BLOCK, param)){
            __android_log_print(ANDROID_LOG_DEBUG, "JNI","control failed");
            goto MPP_TEST_FAILED;
    }

    // interface with both input and output
    for (int i = 0; i < MPI_ENC_LOOP_COUNT; i++) {
            unsigned char *buf;
            buf = (unsigned char *)mpp_buffer_get_ptr(frm_buf);


            memcpy(buf, buf2, YUV_SIZE);


            mpp_frame_init(&enc_in);


            mpp_frame_set_width(enc_in, 1920);
            mpp_frame_set_height(enc_in, YUV_HEIGHT);
            mpp_frame_set_hor_stride(enc_in, MPP_ALIGN(1920, 8));
            mpp_frame_set_ver_stride(enc_in, MPP_ALIGN(YUV_HEIGHT, 8));
            mpp_frame_set_fmt(enc_in, MPP_FMT_YUV420P);
            mpp_frame_set_buffer(enc_in, frm_buf);
            mpp_frame_set_eos(enc_in, 0);




            if (MPP_OK != mpi->encode_put_frame(ctx, enc_in)) {
                 __android_log_print(ANDROID_LOG_DEBUG, "JNI","encode_put_frame failed");
                 goto MPP_TEST_FAILED;
            } else{

                 if (MPP_OK != mpi->encode_get_packet(ctx, &enc_out)) {
                      __android_log_print(ANDROID_LOG_DEBUG, "JNI","encode_get_packet failed");
                      goto MPP_TEST_FAILED;
                 } else{
                      if(enc_out){
                              __android_log_print(ANDROID_LOG_DEBUG, "JNI","enc_out!");
                              void *ptr     = mpp_packet_get_pos(enc_out);
                              size_t len    = mpp_packet_get_length(enc_out);
                              RK_U32 eos = mpp_packet_get_eos(enc_out);
                              unsigned char* out_buf = new unsigned char[len];
                              memcpy(out_buf, ptr, len);

                              FILE *outfile = fopen("/sdcard/output.jpeg", "wb+");
                              fwrite(out_buf, 1, len, outfile);
                              fclose(outfile);

                              mpp_packet_deinit(&enc_out);
                      }
                 }
            }

            mpp_frame_deinit(&enc_in);
    }

    mpi->reset(ctx);

    if (enc_in){
            mpp_frame_deinit(&enc_in);

    }

    mpp_destroy(ctx);
    free(buf2);

    __android_log_print(ANDROID_LOG_DEBUG, "JNI","mpi_test success");


    return;

    MPP_TEST_FAILED:

    if (enc_in){
            mpp_frame_deinit(&enc_in);
    }
    if (ctx){
            mpp_destroy(ctx);
    }
    if (buf2){
            free(buf2);
    }

}
```

复制代码

作者: swlmx **时间:** 2019-11-6 11:31

> jefferyzhang 发表于 2019-10-30 11:43
> toybrick里带的都是64bit的

你们固件里面libmpp.so确实是armv7的，我下载github上mpp的testcode编译出来armv8找不到libmpp.so，只有v7才能运行，而且输出也不正常……

作者: jefferyzhang **时间:** 2019-11-6 12:52

> swlmx 发表于 2019-11-6 11:31
> 你们固件里面libmpp.so确实是armv7的，我下载github上mpp的testcode编译出来armv8找不到libmpp.so，只有v ...

你说的是安卓还是linux?

作者: swlmx **时间:** 2019-11-6 13:58

> jefferyzhang 发表于 2019-11-6 12:52
> 你说的是安卓还是linux?

安卓 字数补丁

作者: jefferyzhang **时间:** 2019-11-6 15:15

> swlmx 发表于 2019-11-6 13:58
> 安卓 字数补丁

哦，安卓是v7 32bit的，因为安卓的框架mediacodec他是32 only的，所以mpp也必须是32才能被系统调用

作者: swlmx **时间:** 2019-11-8 16:44
本帖最后由 swlmx 于 2019-11-8 16:54 编辑

> jefferyzhang 发表于 2019-11-6 15:15
> 哦，安卓是v7 32bit的，因为安卓的框架mediacodec他是32 only的，所以mpp也必须是32才能被系统调用 ...

我用mpp demo里的decode_advanced代码循环300次平均每解码1帧1080p jpg要26ms，而且dmesg显示解码耗时很离谱。我另外测用jpegturbo解码平均才17ms。
另外demo里decode_simple解不了mjpeg……是不是有bug?

mpp流程循环中的伪代码：

```
mpp_packet_set_pos
mpp_packet_set_length
//input
mpi->poll
mpi->dequeue
mpp_task_meta_set_packet
mpp_task_meta_set_frame
mpi->enqueue
//output
mpi->poll
mpi->dequeue
mpp_task_meta_get_frame
mpi->enqueue
```

复制代码

dmesg里的log：

```
[83150.339427] rk_vcodec: vpu2_dec task: 1573200164298 ms
[83150.368903] rk_vcodec: vpu2_dec task: 1573200164327 ms
[83150.394374] rk_vcodec: vpu2_dec task: 1573200164353 ms
[83150.419830] rk_vcodec: vpu2_dec task: 1573200164378 ms
[83150.445152] rk_vcodec: vpu2_dec task: 1573200164404 ms
[83150.470661] rk_vcodec: vpu2_dec task: 1573200164429 ms
[83150.496018] rk_vcodec: vpu2_dec task: 1573200164455 ms
[83150.521544] rk_vcodec: vpu2_dec task: 1573200164480 ms
[83150.550334] rk_vcodec: vpu2_dec task: 1573200164509 ms
[83150.579556] rk_vcodec: vpu2_dec task: 1573200164538 ms
[83150.620477] rk_vcodec: vpu2_dec task: 1573200164579 ms
[83150.649453] rk_vcodec: vpu2_dec task: 1573200164608 ms
[83150.678954] rk_vcodec: vpu2_dec task: 1573200164637 ms
[83150.702283] rk_vcodec: vpu2_dec task: 1573200164661 ms
```

复制代码

感觉是把当前时间戳打出来了

作者: jefferyzhang **时间:** 2019-11-8 18:06

> swlmx 发表于 2019-11-8 16:44
> 我用mpp demo里的decode_advanced代码循环300次平均每解码1帧1080p jpg要26ms，而且dmesg显示解码耗时很离 ...

1. decode_simple 确实解不了mjpeg，很早前就提过需求给相关部门了，他们还没有实现。
2. MJpeg解码比cpu慢是正常的，我们VPU是用来解码264、265帧间编码速度快的，这种帧内编码的解码肯定不如CPU，我们A72CPU跑满1.8GHz的，这种运算肯定是CPU快。
3. 显示不对这个我回头问问相关人员。

作者: swlmx **时间:** 2019-11-11 16:44

> jefferyzhang 发表于 2019-11-8 18:06
> 1. decode_simple 确实解不了mjpeg，很早前就提过需求给相关部门了，他们还没有实现。
> 2. MJpeg解码比cpu ...

了解 谢谢楼主

作者: coolfly19 **时间:** 2019-11-16 15:52
本帖最后由 coolfly19 于 2019-11-16 15:53 编辑

我在https://github.com/rockchip-linux/mpp下载的源代码，用android-ndk-r16b编译mpp后，写了一个采集编码程序，但程序启动时在初始化摄像头时就失败了，cam_hal_init返回错误，这怎么解决啊？是不是mpp和摄像头驱动有关联，需要用同一个ndk版本编译才行？
单独运行采集而不调用mpp编码接口的程序可以正常采集图像。

作者: dawnmaples **时间:** 2019-11-25 15:48
本帖最后由 dawnmaples 于 2019-11-25 16:10 编辑

我在debian下编译的mpp，运行./make-Makefiles.bash，显示...
--compile without drm support
...

CMake Warning:
Maually-specified variables were **not used by the project :**
HAVE-DRM
RKPLATFORM

您好，这里是不是编译了也不能调用GPU，还有我看系统的/usr/lib/aarch64-linux-gnu/下是有libdrm的，用pkg-config也可以看到libdrm的多个支持平台的lib名，但是问题是，就是没有libdrm_rockchip。我看/usr/lib/aarch64-linux-gnu/目录下是有libdrm_rockchip的库，但是没有libdrm_rockchip.so库，只有libdrm_rockchip.so.1和libdrm_rockchip.1.0.0这俩，不明白这是为什么?

同样这个/usr/lib/aarch64-linux-gnu/目录下也有librockchip_mpp.so和librockchip_mpp.so.1，但是却没有头文件对应，这是为啥?

作者: jefferyzhang **时间:** 2019-11-25 16:24

> dawnmaples 发表于 2019-11-25 15:48
> 我在debian下编译的mpp，运行./make-Makefiles.bash，显示...
> --compile without drm support
> ...

mpp跟gpu、drm有啥关系？ mpp只是vpu的api而已。