密级状态：绝密(　　)　　秘密(　　)　　内部(　)　　公开(√)

RKISPV1_Camera_驱动调试方法

（技术部，产品二部）

文件状态： [　] 正在修改 [√] 正式发布	当前版本：	V1.0
	作　　者：	陈潇
	完成日期：	2017-08-25
	审　　核：	
	完成日期：	2017-08-25

福州瑞芯微电子股份有限公司

Fuzhou Rockchips Electronics Co . , Ltd

版 本 历 史

版本号	作者	修改日期	修改说明	备注
V1.0	陈潇	2017-08-25	发布初版	

目 录

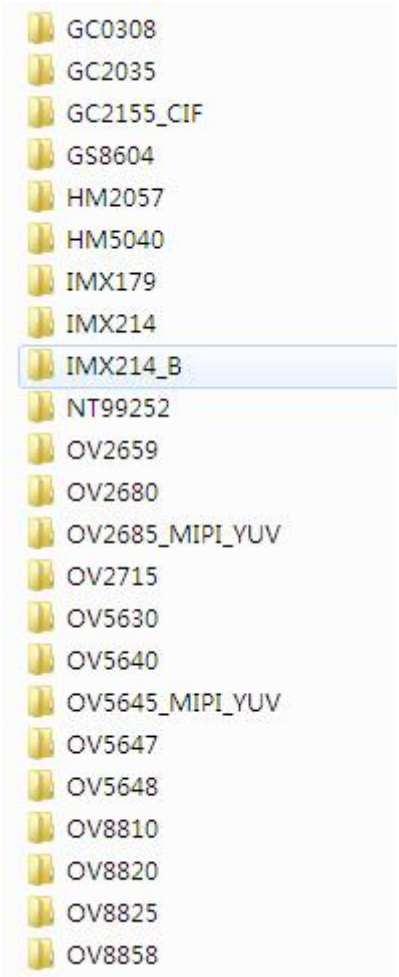
1. RK Camera 驱动调试方法

1.1 概述

本文以 gt5005 来说明拿到新的 camera 如何调试。

RK 平台摄像头驱动路径：hardware\rockchip\camera\SiliconImage\isi\drv

里面各大摄像头驱动文件夹：



首先要知道当前需要调试的摄像头是 dvp yuv，mipi yuv 还是 mipi raw 的。

Dvp yuv 可以参考 GC0308(30W) OV5640(500W),文件添加驱动。驱动文件相对简单，这里不再详述。

关于 mipi yuv 和 mipi raw 这边简单说明下:

目前 rk 平台可以支持的摄像头数据 yuv 和 raw 数据。

Raw 数据是没有经过 isp 信号处理的原始数据，需要 rk 芯片内部 isp 图像处理才能还原真实图像。

Yuv 数据是 camera 端的 isp 经过处理后送出来的数据，所以不需要关心 isp 图像的后期处理。所以代码相对也简单。

1.2 MIPI yuv 输出摄像头调试

Mipi yuv camera 可以参考上图 OV2685_MIPI_YUV(200w),OV5645_MIPI_YUV(500w).

以 2685 为例，需要改动的地方：

```
#define OV2685_DELAY_5MS                (0x0000) //delay 5 ms
#define OV2685_MODE_SELECT              (0x0100) // rw - Bit[7:1
#define OV2685_SOFTWARE_RST             (0x0103) // rw - Bit[7:1

#define OV2685_CHIP_ID_HIGH_BYTE_DEFAULT          (0x26) // r -
#define OV2685_CHIP_ID_MIDDLE_BYTE_DEFAULT        (0x85) // r -
#define OV2685_CHIP_ID_LOW_BYTE_DEFAULT           (0x00) // r -

#define OV2685_CHIP_ID_HIGH_BYTE          (0x300a) // r -
#define OV2685_CHIP_ID_MIDDLE_BYTE        (0x300b) // r -
#define OV2685_CHIP_ID_LOW_BYTE           (0x300c) // r -
/*****************************************************************
```

第一个不用关心， 第二个数据流开始的寄存器，第三个软复位寄存器；

中间三个对应高中低 id 的值，后面三个对应高中低位 id 寄存器；

如果只有一个寄存器则值全部写一样的。

```
#define OV2685_SLAVE_ADDR       0x78U //0x20                    /**< i2c

#define OV2685_SLAVE_ADDR2       0x20

#define OV2685_SLAVE_AF_ADDR    0x00U
```

I2c 地址可以填写 2 个，如果第一个不成功会去通信第二个，一般只要填第一个，第二个可以不管，下面是 af 马达的 i2c 地址，如果没对焦，在配置里面关掉即可，有的话填上对应地址。

```
extern const IsiRegDescription_t OV2685_g_aRegDescription[];

extern const IsiRegDescription_t OV2685_g_svga[];

extern const IsiRegDescription_t OV2685_g_1600x1200[];
```

这几个序列是根据自行调试的摄像头的属性来的，第一个初始化序列，后面跟的都是支持的分辨率，这几个序列定义到摄像头，根据相关

```
#define OV2685_I2C_NR_ADR_BYTES      (2U)
#define OV2685_I2C_NR_DAT_BYTES      (1U)
```

I2c 地址以及数据字节数，根据具体 camera 配置

```
static uint16_t g_suppoted_mipi_lanenum_type = SUPPORT_MIPI_ONE_LANE;
#define DEFAULT_NUM_LANES SUPPORT_MIPI_ONE_LANE
```

上面的是支持的 lane 数，可以用|符号添加如 2lane 和 4lane 的支持，下面是默认使用的 lane 数，当然起作用的还是 cam_board.xml 里面的配置 lane；

有两个比较重要的函数：

OV2685_IsiGetCapsIssInternal

这个是遍历支持的 lane 数以及 lane 数下面的分辨率

比如：

```
        }
    } else if(mipi_lanes == SUPPORT_MIPI_TWO_LANE) {
        switch (pIsiSensorCaps->Index)
        {
            default:
            {
                result = RET_OUTOFRANGE;
                goto end;
            }

        }
    } else if(mipi_lanes == SUPPORT_MIPI_ONE_LANE) {

        switch (pIsiSensorCaps->Index)
        {
            case 0:
            {
                pIsiSensorCaps->Resolution = ISI_RES_1600_1200P7;
                break;
            }
            case 1:
            {
                pIsiSensorCaps->Resolution = ISI_RES_SVGAP30;
                break;
            }
            default:
```

说明 2lane 下没有支持的分辨率 ，1lane 下有两个支持的分辨率，根据摄像头情况来填写。

如果不需要太多，可以只填写自己需要的分辨率即可。

上层会根据获得的支持的分辨率设置里面的其中一种调用。

OV2685_SetupOutputWindow 函数写寄存器，函数里面就是写寄存器的代码。

剩下少量要修改的，直接看下驱动里面的数字，根据摄像头来修改即可。

1.3 MIPI raw 输出摄像头调试

Mipi raw 和上面的调试基本上差不多，但是比 yuv 的多了 isp 方面的一些属性要添加。摄像头因镜头的型号不一样，效果也会相应不一样。

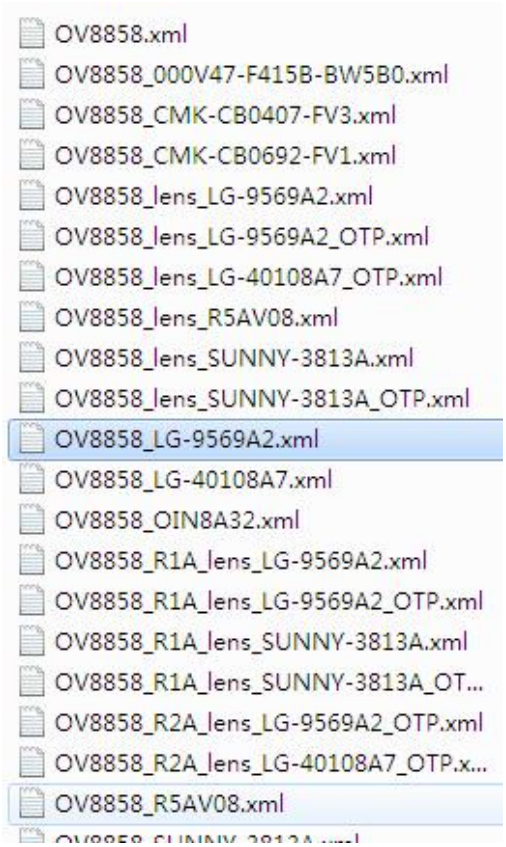
所以效果 xml 文件名增加镜头名来命名。

以 8858 为例：

<SensorLens name="LG-9569A2"></SensorLens>

如果 8858 配置的 LG-9569A2，那么使用的 tuning 文件为

hardware\rockchip\camera\SiliconImage\isi\drv\OV8858\calib



这个文件最终会放到系统的/system/etc

所以机器起来看看是否有对应的文件。

下面介绍下 xml 文件的修改

Xml 文件一般我们做两个分辨率，一个拍照分辨率，一个预览分辨率（大部分是预览的 1/2*1/2），一般是针对这两个分辨率调效果，所以尽量建议驱动支持这两个分辨率，然后就是修改 xml 文件里面的预览分辨率和拍照分辨率。

以 OV8858_LG-9569A2.xml 为例：

1632x1224 为预览分辨率

3264x2448 为拍照分辨率

文件开始

[1632]

[1224]

[3264]

[2448]

这个单独修改成对应的分辨率

1632x1224

3264x2448

请搜索，全部替换成对应的分辨率。

[102 102 102 102 102 102 102 102]

[77 76 77 76 77 76 77 76]

[204 204 204 204 204 204 204 204]

[153 153 153 153 153 153 153 153]

前面两个分别是对应的预览分辨率的 1/16

后面两个分别是对应的拍照分辨率的 1/16

如果除不尽的请前大后小。

修改为相应摄像头的分辨率。

除了效果文件，驱动里面还有

usFrameLengthLines

usLineLengthPck

ulMipiFreq

行场和 mipi 的频率在 datasheet 或者在 camera 原厂给的寄存器列表有对应的值，直接填进去就好。

还有 camera 的 gain 值和曝光对应的寄存器要修改，可以在 datasheet 中找。

另外部分 camera 厂家为了减少出厂摄像头的差异性，里面会带 otp 功能，建议刚开始调试的时候先不要加 otp 的代码，方法可以对比 OV13850 代码把 otp 代码删掉，默认代码 13850 不带 otp。等摄像头调好了，再加上，有少数厂家 otp 读出来的信息位置也可能有差异，要根据对应厂家来改。