


qq69696698的专栏

为学习做一点笔记

目录视图摘要视图RSS 订阅

个人资料



qq69696698

访问: 108015次

积分: 1731

等级: BLOG 4

排名: 第14261名

原创: 63篇 转载: 18篇

译文: 2篇 评论: 36条

文章搜索

文章分类

文章存档

2015年04月 (1)

2015年01月 (1)

2014年11月 (5)

2014年10月 (3)

2014年05月 (1)

展开

阅读排行

Qualcomm Camera基础 (13711)

linux poll函数 (8172)

OpenMAX介绍(总括) (4705)

android Sim卡锁定 pin解 (3859)

(ISP) 图像信号处理器的 (3715)

Unity3D中基于订阅者模式实现事件机制 云计算行业圆桌论坛 【征文】Hadoop十周年特别策划——我与Hadoop不得不说的故事

Qualcomm Camera基础

标签: android callback 照片 图像处理 平台 cam

2012-03-27 16:54 13719人阅读 评论(1)

分类: 图像处理 (9)

版权声明: 本文为博主原创文章, 未经博主允许不得转载。

目录 (?) [+]

高通将android的camera模块重新修改了一下, 与原生的方式存在一些差异。这里将前段时间学习的一些零散知识进行一下总结, 便于以后查阅。

1. 整个模块主要运行三个主线程: control、config及frame, control用来执行总的控制, 是上层控制接口 (这个线程还未去了解)? config主要进行一些配置, 这个线程里面主要进行3A的工作, 另外还有一些跟效果有关的设置; 至于frame线程好像主要用来做预览吧。目前还只是大致了解config线程。

2. 在Qualcomm执行初始化时就会调用到mm_camera_exec()函数来建立config线程launch_cam_conf_thread(); 阅读此线程函数体会发现里面使用了select机制来检测配置指令并进行分发 (调用不同的分支函数)。后面就是一连串的function call了。关于select机制还有不少疑点需要进一步学习: 指令的来源? 如何对文件进行控制的? 比如下面这一段LOG就可以看到对AE、AWB及HIS设置的过程 (只贴了部分):

```
E/CAM_FD ( 194): ..... entering config duty loop .....
E/CAM_FD ( 194): cam_conf: MSM_CAM_IOCTL_GET_STATS: resptype=1 ctrl_cmd.type=4
E/mm-camera( 194): ++++++ config_proc_vfe_event_message type 0 // event类型
E/mm-camera( 194): camconfig_proc_vfe_event_message received msgId = 9
E/mm-camera( 194): vfe_process_msg_evt msg_id = 9
E/mm-camera( 194): vfe_process_VFE_ID_COMMON, vfe common message = 0x4a000 // AE、AWB、IHS
E/mm-camera( 194): received AEC stats: buf = 0x40821000, fd = 52
E/mm-camera( 194): VFE_ID_STATS_AE numReg 256, opt_mode 4
E/mm-camera( 194): vfe_util_do_aec: numReg = 256, num_pixels_per_region_aec = 30856
E/CAM_FD ( 194): isp3a_execute stats_type: 0 // 执行AEC
E/mm-camera( 194): vfe_util_do_aec: no pendingPrepSnapCtrlCmd
.....
```

看过Qualcomm摄像头部分代码的都知道, 里面充斥了一些全局及静态变量, 在未完全理清前看起来很费劲, 比如我现在就处于这种状态。下面就将一些已发现的变量总结一下写在这里, 方便以后查阅。在海量代码中查阅一些数据结构真是一件令人头痛的事情。

1. config线程相关:

config线程主要负责摄像头的配置工作, 由它完成模块的大部分工作, 调试的重点也在这部分。

Camera Face Detection	(3703)
Android Camera架构浅析	(3581)
Qualcomm Camera 开发	(3154)
git stash相关使用	(2979)
MIPI Spec	(2364)

评论排行	
Qualcomm Camera基础	(13)
Camera Face Detection	(7)
Android HDMI 介绍	(4)
(ISP) 图像信号处理器的	(3)
Android Camera架构浅析	(3)
Qualcomm Camera 开发	(3)
git stash相关使用	(2)
open函数的流程	(1)
编程修养(三)	(0)
网络编程学习(十)	(0)

推荐文章	
* HDF5如何检测并删除多余副本块	
* Project Perfect让Swift在服务器端跑起来—让Perfect更Rails (五)	
* 数据库性能优化之SQL语句优化	
* Animation动画详解(七)——ObjectAnimator基本使用	
* 机器学习系列(7)_机器学习路线图(附资料)	
* 大数据三种典型云服务模式	

最新评论	
Qualcomm Camera基础	
无敌小电工: 楼主的葵花宝典能不能给在下发一份 409173854@qq.com	
Camera Face Detection	
GearCrow: @qq69696698:好吧, 依旧感谢你的回复	
Camera Face Detection	
qq69696698: @wuhao1993:抱歉, 我这边也没有人脸识别的算法, 大部分的人脸识别应该都是以人脸的轮廓曲线去比...	
Camera Face Detection	
GearCrow: 博主你好, Camera的FaceDetection 是使用什么算法来检测人脸的?我很想看下它的算法, ...	
Camera Face Detection	
qq69696698: @ashank:一般来说, 人脸识别是底层做的, 只需要摄像头捕捉到的图像可以被底层识别出来, 就算支持了...	
Camera Face Detection	
春水煎茶: 您好, 请问一下 人脸识别为什么会有硬件不支持呢 人脸识别和摄像头哪方面有关 什么情况下 g...	
Android HDMI 介绍	
kangear: @qq69696698:是的, 知识是不变的。	

首先定义了一个全局的变量`config_ctrl_t cfgctrl`,所有的config相关的数据都跟这个变量相关,比如调试过程中常用到的`isp3a_ctrl_t`及`sensor_ctrl_t`等,都挂在这里。从这个结构体层层往下找就能慢慢理清Qualcomm中的数据结构。在config线程里面发生的ctrl command和vfe event message都将传入`cfgctrl`地址及需要刷新的值,进行后续的操作并保持一些状态到`cfgctrl`中。所以如果需要了解某个状态,只要读取`cfgctrl`中的相关字段即可。ctrl command主要是一些菜单的设置,而vfe event message主要是vfe一些event的反馈处理。

Qualcomm提供了一些接口来设置,但获取状态的接口基本没有,只能自己编写。

2.AWB相关(Bayer格式):

在AWB的处理过程中,除了上面提到的`isp3a_ctrl_t`结构体用于对外交互外,AWB模块内部还定义了一个静态变量`awb_algo_ctrl_t*awbCtrl`用于保存AWB相关的数据及状态。这里面最重要的一个结构体`awb_advanced_grey_world_t`就是为增强的灰度世界法算法服务的。

Android高通平台调试Camera驱动全纪录

项目比较紧,3周内把一个带有外置ISP,MIPI数据通信,800万像素的camera从无驱动到实现客户全部需求。

1日 搭平台,建环境,编译内核,烧写代码。

我是一直在Window下搭个虚拟机登服务器搞开发的,对Linux系统环境实在无爱,每每一到项目刚开始内心总有点排斥,过程就比较纠结,看来以后还是要搞个linux真机玩玩。

2日 编写camera驱动大致框架,配置GPIO,I2C,MIPI,电压,时钟等。

很少能碰到FAE只给硬件手册,没有Linux和Android驱动的。因为是camerasensor外接ISP芯片,杯具就发生了。整个系统是这样,高通平台的开发板,自己写驱动来控制ISP芯片,ISP芯片与camerasensor封装在一起,ISP控制sensor,实质就是sensor写寄存器。

开始写驱动了,说好听的那是站在巨人的肩膀上借鉴别的驱动,说难听的就是照葫芦画瓢,反正再改下Kconfig,Makefile,这驱动框架就算是有。

对驱动开发而言,前期的主要工作应该就是配置GPIO口和芯片上电时序了。

每个特定平台在操作GPIO,电压,时钟上都会有自己的一套内核API封装实现,只要能看懂会用这些API即可。配置完后,须在驱动初始化函数里,正确设置芯片的上电时序,确保芯片硬件上能正常工作起来。

3日 编写I2C通信的封装函数,调试CPU与ISP间的I2C通信

对于一些成熟方案,上面的工作完成顺利的话,驱动就差不多了。。很可惜,这块ISP芯片在提升800万camera性能的同时,并没有给我带来足够多的技术支持,只能说,成也ISP,败也ISP,解决方案全都自己来吧。万里长征第一道坎便是I2C。

I2C通信本身要注意两点,

1) SDA第9位ACK位为低时说明从设备有响应。

2) Slave address

芯片手册对这个从设备地址没有统一的写法,有的给出8位地址,有的给出7位地址,一开始容易混淆。如果给出的是8位地址,那第8位是指Write-0或者Read-1,实际的I2C芯片地址是7位的。Linux源码里struct i2c_board_info的板基信息应填写7位I2C地址,另外,I2C芯片地址可以通过开发板shell环境下\$ ls/sys/bus/i2c/devices/ 查看。举个例子,static struct i2c_board_info msm_camera_boardinfo[] __initdata = {

```
{
    I2C_BOARD_INFO("ov8820", 0x78>> 1),
},
```

4日 FAE现场支持

FAE过来了,就确认了一件事,没有现成驱动了,我彻底死心了。后来还发现一个规律,只要FAE来现场那就意味着啥

Android HDMI 介绍

qq69696698: @kangear:这说的是Android2.1、2.2时候的状况,是比较老的了,就简单了解一下好了

git stash相关使用

qq69696698: @jbas:不客气,大家一起学习

Android Camera架构浅析

Brady的博客: camera讨论群:330777216.欢迎camera达人,驱动调试工程师,效果turning工程...

都搞不出来了。。几个人汇聚思想还不如一个人静下心来研究。不过他们此行至少留下一份重要的资料-ISP芯片指令序列, camera所有功能的实现就靠它了。

5日 调通I2C

I2C的调通具有里程碑式的意义,它不仅标志着硬件性能正常开启,更为后来璀璨绚烂的camera世界奠定了坚实的基础。。

有段时间卡在I2C 通信上,给ISP芯片0x3c写入开启芯片命令0xf0成功,但是再发送其他命令全部失败。

分析现象,I2C总线已经可以通信了,问题只能是在ISP芯片上,于是,查电路图,抄家伙起来把电路板上的电和时钟全部再量一遍。。

结果发现,有一路来自自动对焦马达的电压只有1.7V,没有达到要求,驱动里没有把它的GPIO拉高,导致芯片无法正常开启工作。

6日 编写预览驱动,测量MIPI数据

根据葵花宝典里的ISP指令序列,在Linux驱动里和Android高通抽象层里填写相关代码,便可实现预览功能。不过很不幸,光靠那两下子预览还是出不来的。开启预览程序时,用示波器量MIPI总线上的图像数据,能够得:

波形,说明底层驱动的预览功能OK,问题在于高通平台的CAMIFVFE上,于是,翻阅高通的技术资料,学习添加VFE的一些配置。

7日 配置VFE, 点亮预览

预览的成功具有划时代的意义,它不仅标志着camera模块在整个Android系统架构中的成型,更为后来的拍照,录像,图像效果等功能奠定了坚实的基础。预览的出现,意味着我不用再回答那些类似像“camera亮没”之类的只注重表面现象的问题,从那一刻起,我仿佛站上了另一个高度,有种梦回汉唐的感觉。。

8日 健壮代码,编写拍照功能,对焦功能

至此,整个camera模块从上层应用到底层驱动已全部打通,接下来就可以见神杀神,见佛杀佛了。。

9日 编写白平衡,色彩效果,场景模式,ISO,防震,闪光灯等功能

这年头码农伤不起啊!就按葵花宝典上的ISP指令序列往里使劲填充。

10日 登陆服务器提交代码

高通Android平台

最近负责一个项目(手机)上camera的功能,其中有要求做zoom这个功能(项目上要求对所有的分辨率都可以支持4X的zoom),所以把这个部分比较全面的学习了一下,本文对高通在android平台上zoom的实现原理做一个深入的分析,包括的部分主要有zoom功能所涉及高通HW模块的原理架构、高通在android软件中digitalzoom的实现流程以及具体的相关接口参数介绍,旨在让读者能够对高通android平台下digitalzoom的实现原理及架构有一个清楚的了解。

1. Digital zoom原理介绍

这里提到的digitalzoom,即数码变焦,是相机变焦的一种;另外一种为光学变焦,主要是在数码相机中有所应用,它通过相机镜头的移动来放大与缩小需要拍摄的景物,光学变焦倍数越大,能拍摄的景物

就越远,而且不会影响画质。本章节重点介绍数码变焦部分,这种变焦在手机中应该较为广泛。手机上的数码变焦是通过手机内的处理器,把图片内的每个像素面积增大,从而达到放大目的。这种手法如同用图像处理软件把图片的面积改大,不过程序在手机内进行,把原来sensor上的一部份像素使用"插值"处理手段做放大,将sensor上的像素用插值算法将画面放大到整个画面。通过数码变焦,拍摄的景物放大了,但它的清晰度会有一定程度的下降,所以数码变焦并没有太大的实际意义。下图是数码变焦和光学变焦的效果对比,一目了然。

数码变焦一般分为分为2个步骤,crop和插值放大。另外,数码变焦有2种状况:一种是需要拍照画面的尺寸和sensor输出画面的尺寸是一致的;另一种则是需要拍照画面的尺寸 \times zoom等级后比sensor输出画面的尺寸小。在第二种情况下只需要crop就可以了(具体要根据zoom等级计算)

关于zoom的等级,倍数越高,crop的像素就越少,以拍照3M(2048 \times 1536,sensor输出的原始照片为3M)的照片为例,如果做2X的zoom,那么需要从原始照片中crop出1024 \times 768的画面,然后再插值放大成2048 \times 1536;如果是4X的zoom,那么需要从原始照片中crop出512 \times 384的画面,然后再插值放大成2048 \times 1536,以此类推。倍数是指宽和高的倍数,而非面积。

下面结合图片来说明一下数码变焦的原理:

- 1、当需要拍照画面的尺寸和sensor输出画面的尺寸一致

原始照片,红色部分为zoom后需要crop的部分

Zoom后的照片,crop后并插值放大

可以发现,zoom前后照片大小是一致的,但照片的范围变小了,感觉是镜头拉近了,其实就是通过crop后再插值放大来完成,zoom的倍数越高,需要插值的像素就越多,zoom后的照片就会越模糊。

Note:这里的插值放大是指软件算法,和图像处理软件中那种线性放大是不同的,关于算法的实现这里不多介绍了。

- 2、需要拍照画面的尺寸 \times zoom等级后比sensor输出画面的尺寸小

下图是sensor输出的原始照片,1600 \times 1200。需要拍照的分辨率为800 \times 600,即可以作2X的digitalzoom

在这种状况下是不需要插值放大的,图片的质量并没有降低;但如果zoom的等级比较大,比如要4x的zoom,那么光靠crop是不行的,还是得再通过插值放大来完成。通常说来,如果拍照的分辨率比较小,zoom大都可以只通过crop的方式来完成。在上章节中介绍了digitalzoom的效果以及基本的实现原理,本章将着重介绍高通平台上实现digitalzoom所涉及的相关模块架构,因为digitalzoom是camera中的一个feature,分别需要在preview和snapshot中完成,

所涉及的相关模块也都是和camera相关的,如下所示:

上图是preview的时候,digitalzoom所涉及的相关模块,各个模块的用途如下(这里主要介绍sensor、VFE和MDP):

- 1、Sensor

虽然本身也有zoom的功能,但在这里并未使用。Preview时如果做digitalzoom,只是正常的输出frame而已,比如输出30fps的VGA数据(YUV)。

- 2、VFE

DSP的一部分,功能主要都是和图像处理相关,在zoom的时候,它的用途主要就是Crop(剪裁),它会把sensor输出的VGA数据crop成preview时所需要大小的数据,如CIF(352 \times 288),这样的话相当于已经做了一部分zoom(640/352),大概是1.8X,如果不够,剩余的zoom将由后面的MDP来完成。

Note:VFE只有crop的功能,没有upscale(放大)的能力,所以VFE最多只能完成有限的zoom。

- 3、MDP

这是一个专门处理显示数据的处理器,功能比较齐全,在zoom的时候主要的用处就是crop+upscale。因为VFE的zoom能力有限,所以当VFE不能满足要求的时候,MDP则继续完成剩余的zoom,比如:如果要求preview画面的大小为QVGA,现在要做4X的zoom,那么VFE会从原始的VGA数据中crop出

QVGA大小的数据,相当于已经做了2X的zoom,那么剩下的2Xzoom怎么做呢?MDP会从VFE输出的QVGA(320×240)数据中crop出160×120大小的数据(从中间截取),然后再upscale成QVGA大小的数据送到LCD显示,这样相当于又做了2X的zoom,所以加起来一共做了4X的zoom。

Note:MDP最大可以进行4X的upscale。

上面介绍了preview时zoom的实现,下面来看一下拍照时zoom是如何实现的?

Note:为了满足所拍为所看,即拍下来照片的景物范围和preview时所看到的景物范围要保持一致,preview时和snapshot时的zoomlevel必须保持一致。

1、 Sensor

输出拍照需要的原始数据。在当前应用中,不管设置的拍照分辨率是多少,我们要求sensor输出的拍照数据是固定的,即最大3M(2048×1536,以ICE为例)。

2、 VFE

功能和preview时候是一致的,只不过在拍照的时候,VFE会根据zoom的等级以及需要拍照的分辨率来自动crop出合适大小的数据。

例如选择拍照的分辨率为2048×1536,zoom的level为4X,那么VFE将从原始的2048×1536的数据中crop出512×384大小的数据,后面的zoom由Videocore中的jpegencoder完成。

还有一种状况,如果拍照的分辨率较小,那么有可能只通过VFE的crop就可以完成zoom功能,比如拍照的分辨率为1024×768,这个时候如果做2X的zoom,那么VFE只需要从原始的2048×1536的数据中间直接crop出1024×768的数据即可,后面就不需要再用jpegencoder来zoom了。但如果zoom的等级比较高,后面的2Xzoom还是要通过jpegencoder来做了。

3、 Video core

负责把VFE输出的数据encoder成jpeg文件,这里的jpegencoder还有一个比较重要的功能,那就是upscale,通过这个功能,再搭配之前VFE的crop功能,zoom就可以完成了。

Note:Jpeg encoder的upscale功能是有限的,最大可以进行4X的放大,目前可以满足ICE上的需求。

可以看出,从HW架构来说,preview和snapshot只是在后面的upscale部分有所区别,前者是通过MDP来完成,后者则是通过jpegencoder(DSP)来完成。

3.高通 Android平台软件架构分析(digitalzoom相关)

本章将从软件角度来分析一下高通Android平台下digitalzoom的架构以及实现流程,下面先来看一下Android中camera部分的软件架构。

Note:目前以Andrioddonut版本为例,高通在androidclair版本上还没有导入

先来看一下Preview的流程:

1、VFEdriver会把从sensor传送来的frame数据crop成上层需要的大小(具体如果crop要根据zoom的level),然后连同crop信息一起把数据传送到HAL。

2、HAL层不会对preview数据做任何处理,它会这些数据原封不动的callback到camera service,同样包含cropinfo(下章节会详细介绍crop info)。

3、Camera service在一开始的时候会在surfaceflinger中创建一个surface。当cameraservice收到preview数据的时候,2个主要接口会被调用:

1) zoomUpScale_callback

通过调用mSurface->updateCropRect接口把crop相关信息通知给surfaceflinger

2) previewCallback

通过调用mSurface->postBuffer接口把preview的数据传递给surfaceflinger。

4、Surfaceflinger收到数据和crop信息后会调用copybit的接口来驱动MDP去做相关的动作(crop&upscale),然后就去画屏了。

再来看一下snapshot时的流程:

1、VFEdriver把从sensor传递来的原始拍照数据(最大分辨率:2048×1536)crop成zoom需要大小的数据,连同crop信息一起传递给HAL。

2、HAL层收到snapshot的数据后会先去检查crop info,判断是否需要jpegencoder去做upscale的动作。如果不需要就直接encode成jpeg数据;如果需要,填好upscale的参数再做encode。

3、Jpeg encoder后的数据会从HAL callback到camera service, camerbservice会在通知上层去把数据写成文件。

由此可见,在snapshot的时候,整个zoom在HAL就可以完成了,而不像preview的时候,需要在surfaceflinger中配置MDP协助完成。具体的原理在第二章中有详细的叙述,这里就不重复了。

4.Zoom相关接口及参数介绍

本章将从代码的层次来分析一下zoom的实现原理及流程。

Note:基于高通5110 release的代码

首先来看一些配置参数(基于HAL):

```
#define MAX_ZOOM_LEVEL5//对于user来说可以zoom的等级
```

```
static const int ZOOM_STEP =6;//每次zoom时的幅度,可以修改
```

另外需要说明高通VFE中zoom的最大值为60(和分辨率无关)。所以在ICE上我们应该让MAX_ZOOM_LEVEL×ZOOM_STEP=60。要么增大MAX_ZOOM_LEVEL,要么增大ZOOM_STEP。

下面再来看一下zoom等级的对应关系(高通可以做到最大的就是zoom4X,这个是HW(MDP和jpegencoder)的限制):

下面看一下HAL层zoom的接口

HAL层的接口比较简单,就是setZoom,上层传递一个zoom的level即可,执行时会判断参数,如果没有超出则通知VFE进行crop,如下:

所以surfaceflinger在更新画面的时候就会根据这些参数来配置MDP,完成后续的操作了。

拍照的时候同样也是这样的原理,差别在于crop中的信息不需要传递给上层,而是直接传递给jpegencoder即可(写到mDimension这个结构体中),如下(HAL中的receiveRawPicture函数):

Jpegencoder完成后,HAL只需要把zoom好的jpegdata callback给上层就OK了,所以拍照部分的zoom不需要上层额外的处理。

下面看看Camera service里面是怎么处理的?

Cameraservice收到callback后会把crop相关信息及标志更新到preview所申请的surface中。

下面看一下VFE输出数据的格式:

在Preview的时候,通过MSM_CAM_IOCTL_GETFRAME系统命令从底层得到preview的数据,格式如下:

buffer为数据地址,y_off和cbcr_off分辨为Y的偏移和CBCR的偏移,通过y_off=0,cbcr_off=w*h,这里和zoom相关的是cropinfo,比较重要,如下:

可以看出有2个buffer的参数,其中1是preview的,2是snapshot的。如果没有开启zoom功能,这些参数都是空的;如果zoom的level比较低,VFE足以处理,那么这些参数也是空的。

只有当VFE不足以处理所需要的zoom level时,这些参数的值才有意义。具体含义如下:

out的值代表上层需要数据的宽和高,比如说上层设置的preview大小为480×320,那么out1_w=480; out1_h=320;而in的值则代表后端的MDP或是jpegencoder需要crop的大小,举例来说明:

Sensor输出VGApreview画面,MMI设置HVGApreview大小,如果要做2X的zoom,VFE能力有限,只能做640/480=1.3X,这个时候VFE输出数据是crop后的HVGA数据,crop信息中的in1_w=320;

out1_w=216,意思是后面的MDP需要从HVGA的数据中crop出320×216大小的数据,然后在scale成HVGA,这样整体算起来就是zoom2X了。

所以HAL只需要将preview数据以及crop info传递给上层即可,这里是通过callback进行的。

顶

0

踩

0

上一篇

Android Camera架构浅析

下一篇

Qualcomm Camera 开发遇到的错误及解决方法

我的同类文章

图像处理 (9)

• Qualcomm Camera 开发遇...

2012-03-29

阅读 3153

• Android Camera架构浅析

2012-03-27

阅读 3579

• (ISP)图像信号处理器的相...

2012-03-26

阅读 3714

• Camera Face Detection

2012-03-06

阅读 3702

• YUV格式的解析

2012-02-01

阅读 371

• android4.0 camera hal 移植

2012-01-04

阅读 1104

• Multimedia and Camera - ...

2012-01-04

阅读 485

• Camera

2012-01-04

阅读 1102

• 入门视频采集与处理(学会...

2011-05-23

阅读 845

猜你在找

高通骁龙处理

高通骁龙处理

oa系统

高通处理器排

复式单身公寓

透明手机


led显示屏

查看评论

- 10楼

无敌小电工


2015-12-05 16:57发表



楼主的葵花宝典能不能给在下发一份 409173854@qq.com
- 9楼

Brady的博客


2014-08-18 17:49发表



camera讨论群:330777216.欢迎camera达人, 驱动调试工程师, 效果turning工程师, CAMERA的APK技术牛人入内。纯粹技术, 谢绝广告。集聚高通、MTK因特尔等平台各路人马, 为打造中国camera第一联盟而不断努力。
- 8楼

qq69696698


2014-03-19 17:54发表



回3楼:如果sensor支持对应的function,则可以, 否则需要对应的isp或通过别的演算法来处理
- 7楼

qq69696698


2014-03-19 17:52发表



sensor和isp各自有各自的驱动, 只要确认isp有收到frame, 就代表sensor驱动已经起来了
- 6楼

qq69696698

2014-03-19 17:50发表



您这个问题看起来是display的问题, 可以看看preview收到帧后面的耗时如何

5楼 [intelSword](#) 2013-09-02 11:33发表



请教个问题啊 准备开发一个方案 除了isp的驱动 sensor是不是也需要驱动啊? sensor是ov9712 isp是集成到主芯片内的

Re: [qq69696698](#) 2014-03-19 17:59发表



回复intelSword: 7楼有回复您的问题

4楼 [qhaiwyy](#) 2013-08-12 23:08发表



讲的非常棒 佩服啊 特别是 sensor -VFE -MDP 数据大小关系分析非常好, 我遇到一个问题, 录像时候预览会卡(估计掉帧), 但是播放录制的视频没有这个现象, 请问是不是MDP 那里出的问题呢?

Re: [qq69696698](#) 2014-03-19 17:58发表



回复qhaiwyy: 6楼有回复您的问题

3楼 [suheng007](#) 2013-04-03 13:16发表



能不能解释一下插值是驱动里面做的还是上层做的?
插值是给sensor发相应命令, 从sensori读出来的就是插值完成的还是说要将读出来的图片进行加工处理?

Re: [qq69696698](#) 2014-03-19 17:58发表



回复suheng007: 在8楼有回复您的问题

2楼 [suheng007](#) 2013-03-26 20:14发表



怎么看不到图啊

1楼 [zzb_boy](#) 2012-09-07 16:58发表



这是我见过讲得最好的一篇。

您还没有登录,请[\[登录\]](#)或[\[注册\]](#)

* 以上用户言论只代表其个人观点, 不代表CSDN网站的观点或立场

核心技术类目

全部主题

Hadoop

AWS

移动游戏

Java

Android

iOS

Swift

智能硬件

Docker

OpenStack

VPN

Spark

ERP

IE10

Eclipse

CRM

JavaScript

数据库

Ubuntu

NFC

WAP

jQuery

BI

HTML5

Spring

Apache

.NET

API

HTML

SDK

IIS

Fedora

XML

LBS

Unity

Splashtop

UML

components

Windows Mobile

Rails

QEMU

KDE

Cassandra

CloudStack

FTC

coremail

OPhone

CouchBase

云计算

iOS6

Rackspace

Web App

SpringSide

Maemo

Compuware

大数据

aptech

Perl

Tornado

Ruby

Hibernate

ThinkPHP

HBase

Pure

Solr

Angular

Cloud Foundry

Redis

Scala

Django

Bootstrap

公司简介 | 招贤纳士 | 广告服务 | 银行汇款帐号 | 联系方式 | 版权声明 | 法律顾问 | 问题报告 | 合作伙伴 | 论坛反馈

网站客服 杂志客服 微博客服 webmaster@csdn.net 400-600-2320 | 北京创新乐知信息技术有限公司 版权所有 |

江苏乐知网络技术有限公司 提供商务支持

京 ICP 证 09002463 号 | Copyright © 1999-2014, CSDN.NET, All Rights Reserved

8 of 8

03/02/2016 04:18 PM