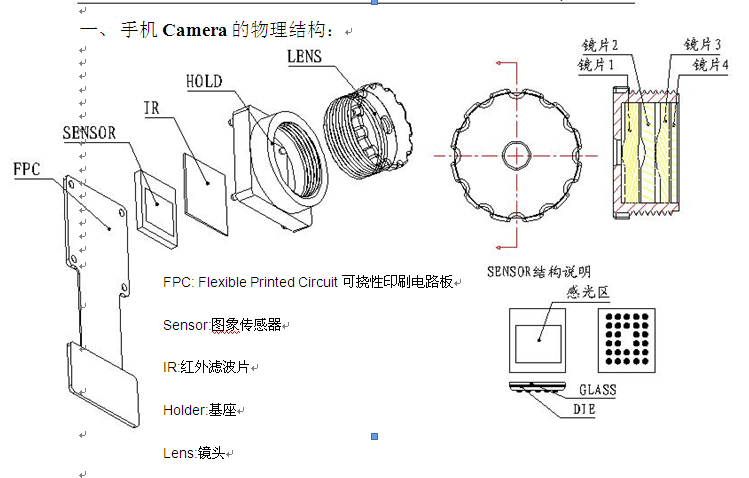
# CAM框架小结及各场景模式效果的实现

## 一．写文档初衷

先前准备写一个camera的各个菜单功能和实现原理的文档小结，不过赵树超总结的文档 和我整理的对CAM各个菜单的功能以及各平台的Feature介绍已比较全面，这里我就不赘述了。我这里就小结一个CAM框架及各场景模式效果实现的文档，主要针对RAW格式CAM，（YUV格式CAM雷同，两者代码主要区别在于效果调试代码位置不一样），文档内容比较简单，适合以前未接触过MTK CAM架构的新员工学习，以便能够尽快的上手CAM调试。

## 二．手机CAM的物理结构：



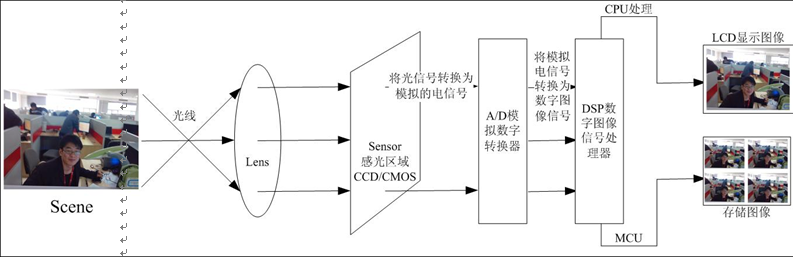
## 三．Camera 的成像原理

景物通过镜头（LENS）生成的光学图像投射到图像传感器(Sensor)表面上，然后转为模

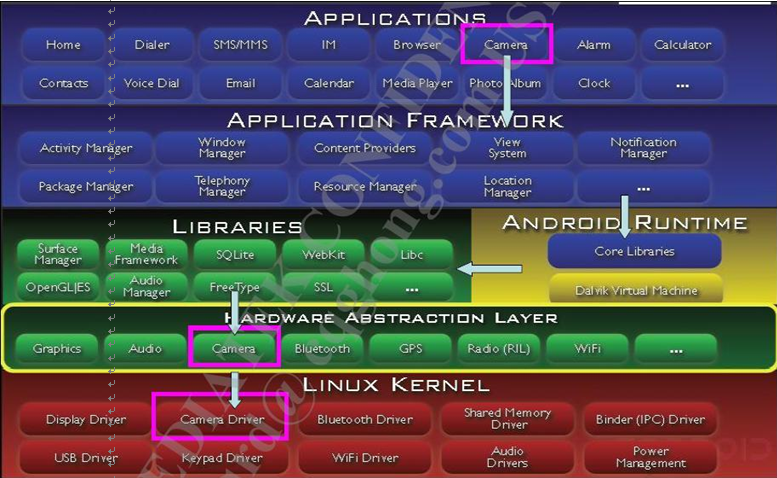
拟的电信号，经过 A/D（模数转换）转换后变为数字图像信号，再送到数字信号处理芯片

（DSP）中加工处理，再通过 IO 接口传输到 CPU 中处理，通过 LCD 就可以看到图像了。

MCU(微控制单元，单片微型计算机)



## 四．MTK 平台 Camera 驱动架构



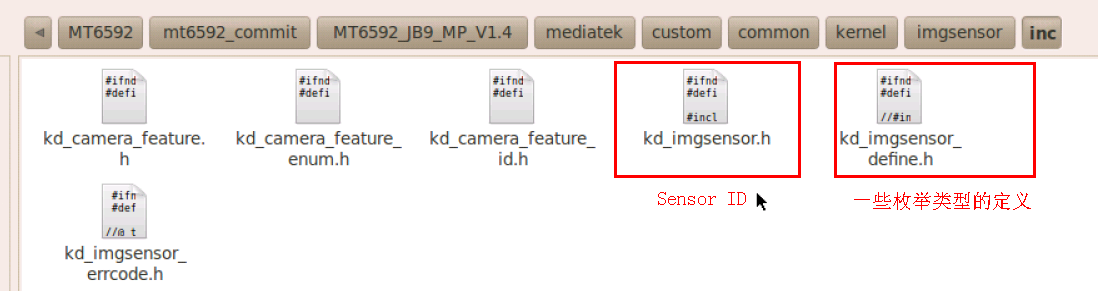
上图的架构大家通过看一些MTK datasheet，估计都有了一定的了解，android 将系统大致分为应用层、库文件和硬件抽象层、Linux 内核三层。在底层的内核空间，Camera 的 driver 将其驱动起来以后，将硬件驱动的接口交给硬件抽象层，android 上层的 Camera 应用程序在 android 实时系统中的虚拟机中，加载 android 留给 Camera 公用的一些库文件，调用硬件抽象层的接口来控制 Camera硬件来实现功能。当然，如果是 Raw 模式的 Camera，还需要在硬件抽象层调用一些参数来控制 Camera 的效果。

## 五． MTK 平台 Camera 相关代码文件（以下代码均为 MTK6592 平台）

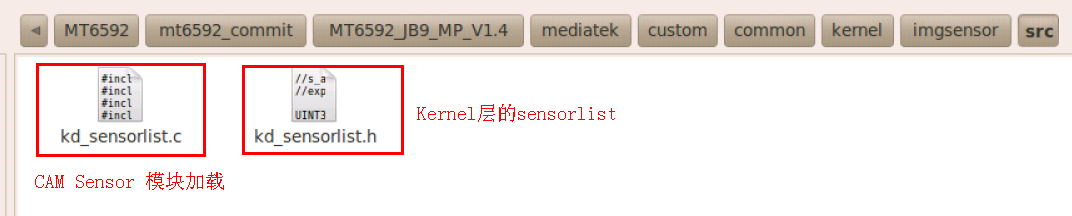
###### 1．CameraSensor 驱动相关文件



###### 2．Sensor ID 和一些枚举类型的定义



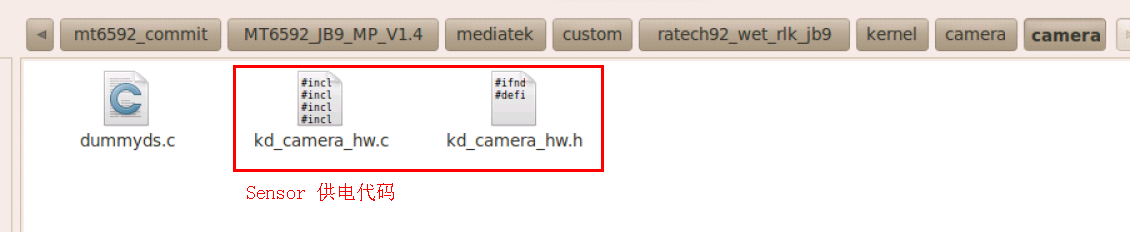
###### 3． Kernel Space 的 SensorList（要和用户层的sensorlist前后匹配）imgsensor 模块注册



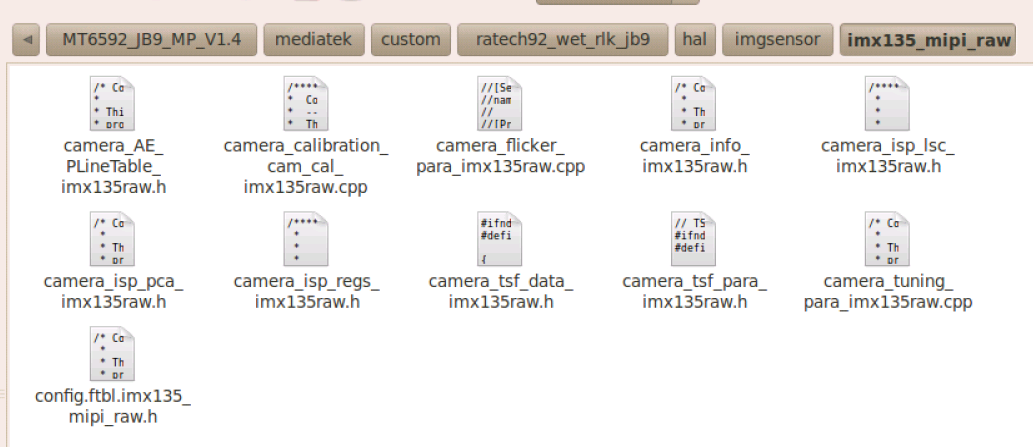
###### 4． User Space 的 SensorList，向用户空间提供支持的 SensorList（要和Kernel层的sensorlist前后匹配）



###### 5．SENSOR供电



###### 6．调整Sensor 效果的代码文件（RAW格式CAM的重中之重）



camera\_AE\_PLineTable\_imx135raw.h ------- 调整自动曝光值。

camera\_flicker\_para\_imx135raw.cpp -------- 自动调整flicker（工频干扰引起的波纹现象），可以使flicker自动调整效果更好，使用 generate\_flicker\_params\_v3.exe软件生成。

camera\_isp\_lsc\_imx135raw.h -------- 调整镜头阴影补偿。

camera\_isp\_pca\_imx135raw.h -------- 使用HSV Domain调整颜色，H:Hue色调，

S:Sat饱和度，V:Lum亮度

camera\_isp\_regs\_imx135raw.h ------- 包括调整，ANR(躁点)，EE(锐利度)，PCA(饱和度)，Brightness（亮度），Contrast(对比度)，knee(暗阶压抑度)等等。

camera\_tsf\_data\_imx135raw.h

camera\_tsf\_para\_imx135raw.h ------- 镜头阴影补偿自动调整，可以适合多种环境的光源，我们需提供各种环境的RAW格式图片让MTK帮忙生成这两个代码文件。

camera\_tuning\_para\_imx135raw.cpp ------- 调整AE曝光参数，各种模式下（preview，capture，video）的flare调试。

config.ftbl.imx135\_mipi\_raw.h -------- CAM的各种功能（场景模式，拍照模式，图片大小，对焦，ISO等）的配置和开关文件。

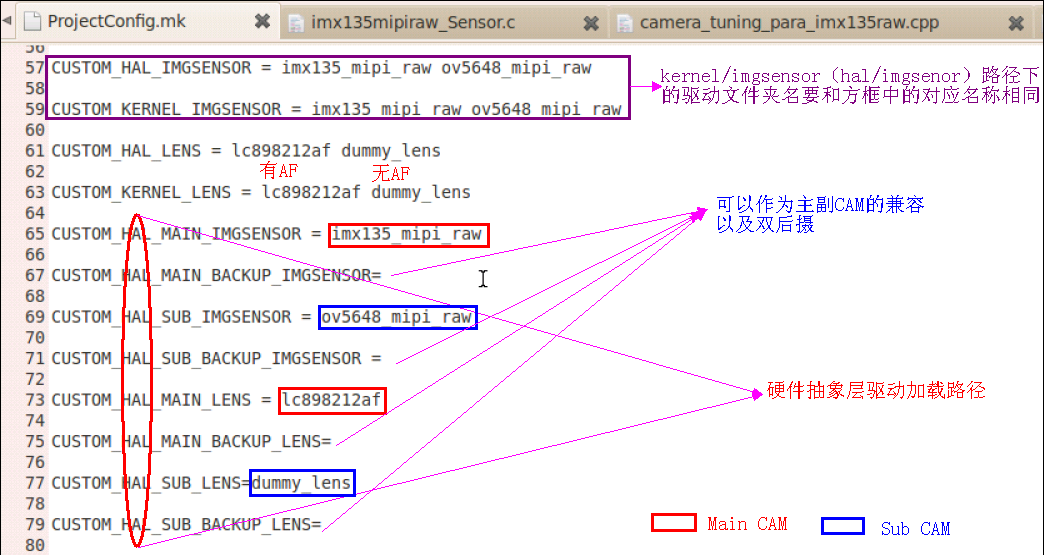
###### 7.Sensor 对焦效果文件

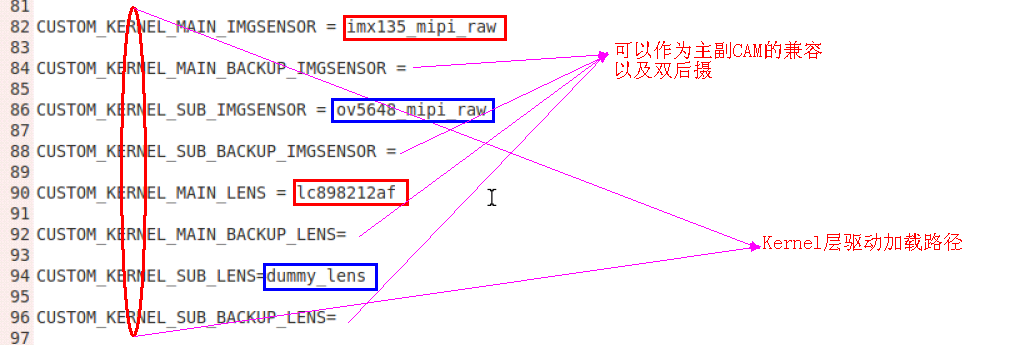


## 六．Camera 驱动添加、调试流程

###### 1、修改系统配置文件 ProjectConfig.mk

**mediatek\config\$project$\ProjectConfig.mk**



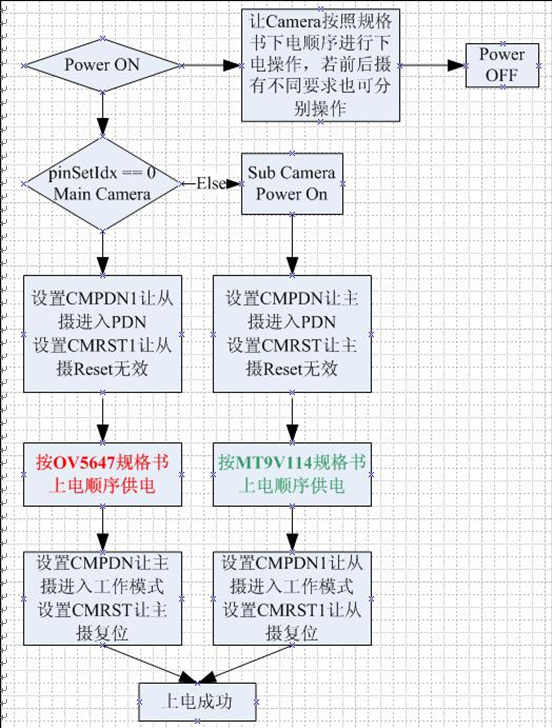


###### 2、 检查、配置供电文件

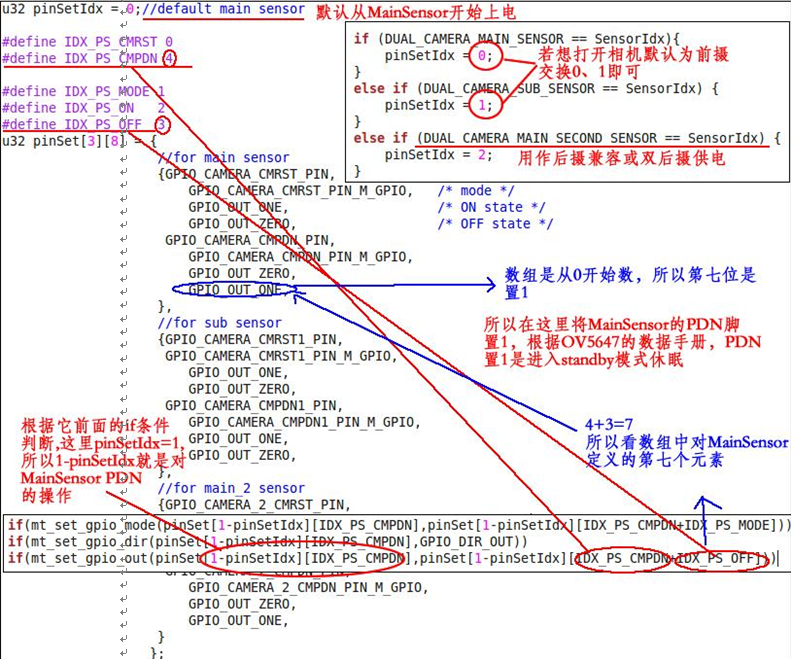
**mediatek\custom\$project$\Kernel\Camera\Camera\kd\_camera\_hw.c**

这个为了省事，找了个现成的图为示例，换汤不换药，原理都是一样的

Camera 供电流程（以 3M 前摄 MT9V114+5M 后摄 OV5647 为例）



其实在 kd\_camera\_hw.c 中只有一个函数 kdCISModulePowerOn（），在这个函数中需要注意的是通过 GPIO 口控制 PDN 和 RST 引脚的时候，对于其相关的定义，由其在切换平台的时候，可能定义顺序不同。



###### 3、 添加 Camera 驱动代码

具体添加Camera kernel层驱动代码和硬件抽象层（hal）的效果调试代码步骤请见以下文档，该文档介绍的比较清晰。

## 七．各场景模式效果的实现

接下来我们了解一下RAW格式CAM是怎样控制各场景模式的效果显示的。首先理解一些场景模式的意义，设置这么多场景模式的意义是为了在某些特殊场景下切换到相对应的场景模式下拍照效果比较好，而不是随便找个场景进行切换就能有明显的变化，那样的话和color effect特效就没有什么区别了，例如Sport模式，顾名思义，就是运动，我们要拍摄正在运动的物体，那么如何让运动的物体被拍的清晰，而没有拖影，那么就必须提高帧率，提高帧率就势必会降低AE，所以可以看到，切换到Sport模式下，帧率会明显提高，但是预览界面会变暗，这都是正常的。

场景模式是由哪些参数控制的呢。场景模式是由

BPC : 没弄明白，一般调试时也没有调试到该参数。

NR1 : Noise Reduction。

CFA : Noise Reduction的第一道关卡 RGB Domain的NR。

GGM:控制Gama R，B，G，三基色。

ANR : Noise Reduction的第二道关卡 YUV Domain的NR，效果大于CFA。

CCR : 控制Color Saturation(减少)。

EE : Sharpness Setting，锐利度的控制。

PCA : Saturation Setting，饱和度的控制。

**1、上述参数所属代码文件路径：**

**/mediatek/custom/$project$/hal/imgsensor/imx135\_mipi\_raw/camera\_isp\_regs\_imx135raw.h（以MT6592平台IMX135 CAM为例）  
 2、但是控制某一场景模式所需上述参数哪一个程度的索引值位于：**

**/mediatek/custom/$project$/hal/camera/camera/cfg\_isp\_tuning\_idx.h**

**例如正常拍照Sport模式所需的参数的索引值如下图显示：**

****

**3、camera\_isp\_regs\_imx135raw.h代码可由CCT工具调试生成，也可以手动进行更改，但是手动更改工作量太大也不好控制效果，故一般由CCT工具可视化调试后自动生成。**

**4、具体如何打开或关闭某一个场景模式的功能，其控制文件在以下路径的文件里：**

**/mediatek/custom/$project$/hal/imgsensor/imx135\_mipi\_raw/config.ftbl.imx135\_mipi\_raw.h**

**控制打开或关闭方法请见以及**参考 **葵花宝典FAQ08087。**