4.9--4.10关于drm的分析

https://en.wikipedia.org/wiki/Direct_Rendering_Manager
通过这里的介绍可以确定分辨率的设置存在于KMS中。

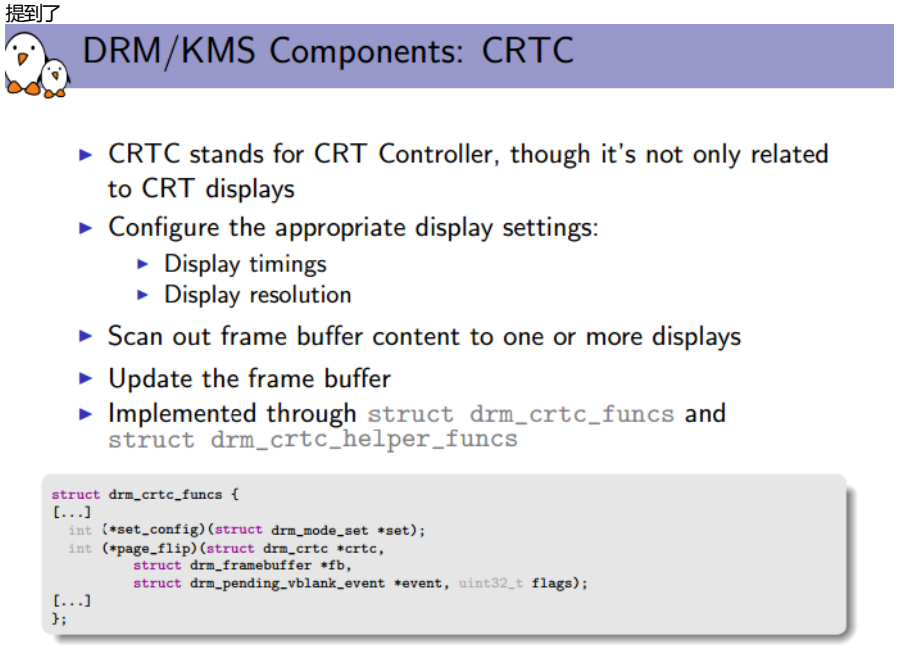
DRM由两部分组成：
DRM core：提供一个基本的框架，对下，不同的DRM Driver可以向DRM Core注册，对上，为用户空间提供一个独立于硬件的最小且公共的ioctl操作集合。
DRM Driver：实现依赖于硬件的ioctl API，也可以额外提供在特定硬件上支持的API。如果DRM Driver提供了额外的API，用户空间的libdrm库也可以提供额外的功能API。

KMS：Kernel Mode Setting
从wiki上这段话的描述确定对分辨率的设置被写到了DRM中的KMS模块：
Finally, it was decided that the best approach was to move the mode-setting code to an single place inside the kernel, specifically to the existing DRM module.[26][27][31][32] Then, every process —including the X Server— should be able to command the kernel to perform mode-setting operations, and the kernel would ensure that concurrent operations don't result in an inconsistent state. The new kernel API and code added to the DRM module to perform these mode-setting operations was called *Kernel Mode-Setting* (KMS).

KMS信息：
https://lwn.net/Articles/653071/这篇文章中提到：
Besides being the central object for routing, a CRTC also keeps track of other settings like the display mode (i.e. refresh rate and resolution) used by the display pipeline and the background color that should be shown where no plane is visible at all.
由此可以锁定，分辨率相关的东西基本可以锁定在CRTC中了。
注：KMS除FrameBuffer外的三大控件

1. **CRTC 的常用行为如下：**
   - DPMS (Display Power Manage System) 电源状态管理 (crtc_funcs->dpms)
   - 将 Framebuffer 转换成标准的 LCDC Timing， 其实就是一帧图像刷新的过程（crtc_funs->mode_set）
   - 帧切换，即在 VBlank 消影期间，切换 Framebuffer（crtc_funcs->page_flip）
   - Gamma 校正值调整（crtc_funcs->gamma_set）
2. **Encoder 的常用行为如下：**
   - DPMS (Display Power Manage System) 电源状态管理 (encoder_funcs->dpms)
   - 将 VOP 输出的 lcdc Timing 打包转化为对应接口时序 *HDMI TMDS* / ... (encoder_funcs->mode_set)
3. **Connector 的常用行为如下：**
   - 获取上报 热拔插 Hotplug 状态
   - 读取并解析屏 (Panel) 的 EDID 信息

在这篇文章中http://events.linuxfoundation.org/sites/events/files/slides/brezillon-drm-kms.pdf
提到了



到这里基本上锁定了显示器分辨率设置在DRM---KMS中的位置。

分析关联的几个文件中的源码找到以下关于显示器分辨率设置的接口：
int drm_crtc_helper_set_config(struct drm_mode_set *set)
设置新的配置，提供给上层调用。（比如ioctl和fbdev）

查看drm_mode_set 结构体定义：
connectors参数存储了此crtc所有的connector

```
struct drm_mode_set {
    struct drm_framebuffer *fb;
    struct drm_crtc *crtc;
    struct drm_display_mode *mode;
    uint32_t x;
    uint32_t y;
    struct drm_connector **connectors;
```

```
    size_t num_connectors;
};
```

drm_mode_config结构体存储了所有CRTC, encoders和connectors。

三大控件中connector是更偏向于硬件的，暂时猜测显示器分辨率的信息应该在connector中有所体现。
在drm_connector结构体定义中也发现了一些字段可能为分辨率，但是还需要进一步分析验证。

附：
在其头文件中又找到一个以后可能会用到的函数：
void drm_kms_helper_hotplug_event(struct drm_device *dev)
注释说明此函数的作用是：KMS的热拔插事件发生时，KMS的处理流程。

DRM介绍
http://landley.net/kdocs/htmldocs/drm.html#drm-kms-init 关于DRM介绍以及暴露给用户空间的详细接口。
http://elinux.org/images/7/71/Elce11_dae.pdf关于嵌入式开发的文章，里面有一些关于DRM和KMS的介绍