

Rockchip HDMI 软件开发指南

文档标识: RK-SM-YF-119

发布版本: V1.2.0

日期: 2024-03-28

文件密级: ☐绝密 ☐秘密 ☐内部资料 ☒公开

免责声明

本文档按“现状”提供，瑞芯微电子股份有限公司（“本公司”，下同）不对本文档的任何陈述、信息和内容的准确性、可靠性、完整性、适销性、特定目的性和非侵权性提供任何明示或暗示的声明或保证。本文档仅作为使用指导的参考。

由于产品版本升级或其他原因，本文档将可能在未经任何通知的情况下，不定期进行更新或修改。

商标声明

“Rockchip”、“瑞芯微”、“瑞芯”均为本公司的注册商标，归本公司所有。

本文档可能提及的其他所有注册商标或商标，由其各自所有者所有。

版权所有 © 2024 瑞芯微电子股份有限公司**

超越合理使用范畴，非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd.

地址：福建省福州市铜盘路软件园A区18号

网址：www.rock-chips.com

客户服务电话：+86-4007-700-590

客户服务传真：+86-591-83951833

客户服务邮箱：fae@rock-chips.com

前言

文本主要介绍 Rockchip 平台基于 DRM 显示框架的 HDMI 的使用与调试方法。

概述

产品版本

芯片名称	内核版本
RK322X/RK3328/RK3368/RK3399/RK3288/RK3528/RK356X/RK3588	LINUX kernel 6.1/5.10/4.19/4.4

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

软件开发工程师

修订记录

日期	版本	作者	修改说明
2020-06-24	V1.0.0	操瑞杰	初始发布
2020-08-26	V1.1.0	操瑞杰	新增 HDCP 2.2 使用说明 新增 RK3288/RK3399 HDMI-PHY-PLL 修改方法
2024-03-29	V1.2.0	操瑞杰	删除 FB 框架介绍，独立成新文档 新增 RK3528/RK3576/RK3588 说明 新增 DTS 配置 INNO HDMI PHY 信号强度的说明 修复颜色格式描述错误 更新 HDMI 强制输出分辨率的方法

Rockchip HDMI 软件开发指南

1. Rockchip 平台 HDMI 简介
2. DRM 框架 HDMI 介绍
 - 2.1 HDMI 软件功能配置
 - 2.1.1 使能 HDMI
 - 2.1.2 绑定 VOP
 - 2.1.3 打开开机 logo
 - 2.1.4 VOP dclk 绑定 PLL
 - 2.1.4.1 RK3288 绑定 PLL
 - 2.1.4.2 RK3399 绑定 PLL
 - 2.1.4.3 RK356X 绑定 PLL
 - 2.1.4.4 RK3588/RK3576 绑定 PLL
 - 2.1.5 RK3288/RK3399/RK3528/RK356X HDCP 使能
 - 2.1.5.1 HDCP 1.4 使能
 - 2.1.5.2 HDCP 2.2 使能
 - 2.1.6 RK3588/RK3576 HDCP 使能
 - 2.1.7 DDC 的 I2C 速率配置
 - 2.1.8 HDMI 信号强度配置
 - 2.1.8.1 RK322X/RK3328/RK3528/RK356X 信号强度配置
 - 2.1.8.1.1 RK322X 信号配置
 - 2.1.8.1.2 RK3328 信号配置
 - 2.1.8.1.3 RK3528 信号配置
 - 2.1.8.2 RK3288/RK3368/RK3399/RK356X 信号配置
 - 2.1.9 新增特殊分辨率
 - 2.1.9.1 新增特殊分辨率时序
 - 2.1.9.2 RK322X/RK3328/RK3528 新增 PLL 配置
 - 2.1.9.3 RK3288/RK3368/RK3399/RK356X 新增 PLL 配置
 - 2.1.9.4 RK3588/RK3576 新增 PLL 配置
 - 2.1.10 打开音频
 - 2.2 Android 显示框架配置
 - 2.2.1 主副显示接口配置
 - 2.2.2 主副显示接口查询
 - 2.2.3 Framebuffer 分辨率配置
 - 2.2.4 分辨率过滤配置
 - 2.2.5 HDMI 设置选项
 - 2.3 常用调试方法
 - 2.3.1 查看 VOP 状态
 - 2.3.2 查看 Connector 状态
 - 2.3.3 查看 HDMI 工作状态
 - 2.3.4 查看 HDMI CEC 状态
 - 2.3.5 强制使能/禁用 HDMI
 - 2.3.6 命令行设置分辨率
 - 2.3.6.1 Android 7.x & Android 8.x 分辨率设置
 - 2.3.6.2 Android 9.0 及以上版本分辨率设置
 - 2.3.7 命令行设置颜色
 - 2.3.7.1 Android 7.x & Android 8.x 颜色设置
 - 2.3.7.2 Android 9.0 及以上版本颜色设置
 - 2.3.8 设置过扫描
 - 2.3.8.1 Android 7.x & Android 8.x 过扫描设置
 - 2.3.8.2 Android 9.0 及以上版本过扫描设置
 - 2.3.9 设置亮度、对比度、饱和度、色度
 - 2.3.9.1 Android 7.x & Android 8.x 亮度、对比度、饱和度、色度设置
 - 2.3.9.2 Android 9.0 及以上版本亮度、对比度、饱和度、色度设置
 - 2.4 常见问题排查
 - 2.4.1 插入或切换分辨率，电视提示无信号或格式不支持或画面不稳定，时有时无，或有大量彩色亮点、亮线
 - 2.4.2 播放视频时电视提示无信号或格式不支持

- 2.4.3 部分电视提示无信号、黑屏、花屏
- 2.4.4 读取 EDID 失败时，如何设置默认分辨率
- 2.4.5 强制输出指定分辨率
- 2.4.6 Recovery HDMI 无显示
- 2.4.7 settings 无法设置 HDMI 分辨率
- 2.4.8 DDR 带宽不足导致的问题
- 2.4.9 4K UI 相关问题
- 2.4.10 setting 中 HDMI 分辨率列表中没有 4K 分辨率
- 2.4.11 RK3588 setting 中 HDMI 分辨率列表中没有 8K 分辨率
- 2.4.12 RK3588/RK3576 HDMI 8K/4K120 等 分辨率闪屏、显示异常
- 2.4.13 HDMI 认证申请表的填写

1. Rockchip 平台 HDMI 简介

Rockchip 各平台的 HDMI 功能如下：

功能	RK3288	RK3368	RK322X	RK3328	RK3399	RK3528	RK356X	RK3588	RK3576
最大输出分辨率	3840x2160p60	4096x2160p60	4096x2160p60	4096x2160p60	4096x2160p60	4096x2160p60	4096x2160p60	7680x4320p60	4096x2160p120
隔行模式	N	N	Y	Y	Y	Y	Y	Y	Y
4K-60/50 Hz 支持的颜色	RGB YCbCr444 YCbCr422 YCbCr420(只有 RK3288W 支持)	YCbCr420	YCbCr420	YCbCr420	RGB YCbCr444 YCbCr422 YCbCr420	RGB YCbCr444 YCbCr422 YCbCr420	RGB YCbCr444 YCbCr422 YCbCr420	RGB YCbCr444 YCbCr420	RGB YCbCr444 YCbCr422 YCbCr420
是否支持 10 bit 色深	Y	N	Y	Y	Y	Y	Y	Y	Y
支持HDMI协议版本	HDMI 2.0	HDMI 2.0	HDMI 2.0	HDMI 2.0	HDMI 2.0	HDMI 2.0	HDMI 2.0	HDMI 2.1	HDMI 2.1

- DRM:
DRM 全称是 Direct Rendering Manager 是 DRI (Direct Rendering Infrastructure) 框架的一个组件。
LINUX 4.4 及其以后的内核采用 DRM 框架， HDMI 驱动的路径为：

```
kernel/drivers/gpu/drm/rockchip/dw_hdmi-rockchip.c
kernel/drivers/gpu/drm/rockchip/inno_hdmi.c
kernel/drivers/gpu/drm/bridge/synopsys/
```

2. DRM 框架 HDMI 介绍

2.1 HDMI 软件功能配置

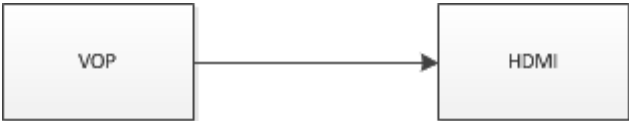
2.1.1 使能 HDMI

打开HDMI需要添加：

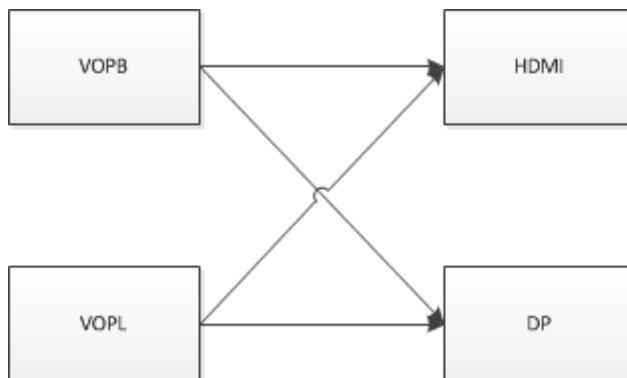
```
&hdmi {
    status = "okay";
};
```

2.1.2 绑定 VOP

在 Rockchip 的各个平台中，各种显示接口（HDMI、DP、CVBS等）输出的图像数据来自于 VOP：



如果平台存在两个 VOP（RK3288、RK3399）：VOPB（支持 4K）、VOPL（只支持 2K），两个 VOP 可以分别与两个显示接口绑定（一个显示接口只能和一个 VOP 绑定），且可以相互交换：



当 dts 中显示设备节点打开时，显示接口对应 VOPB 和 VOPL 的 ports 都会打开，所以需要关闭用不到的那个 VOP 对应的 port。

比如 HDMI 绑定到 VOPB 需要添加：

```
&hdmi_in_vopl {
    status = "disabled";
};
```

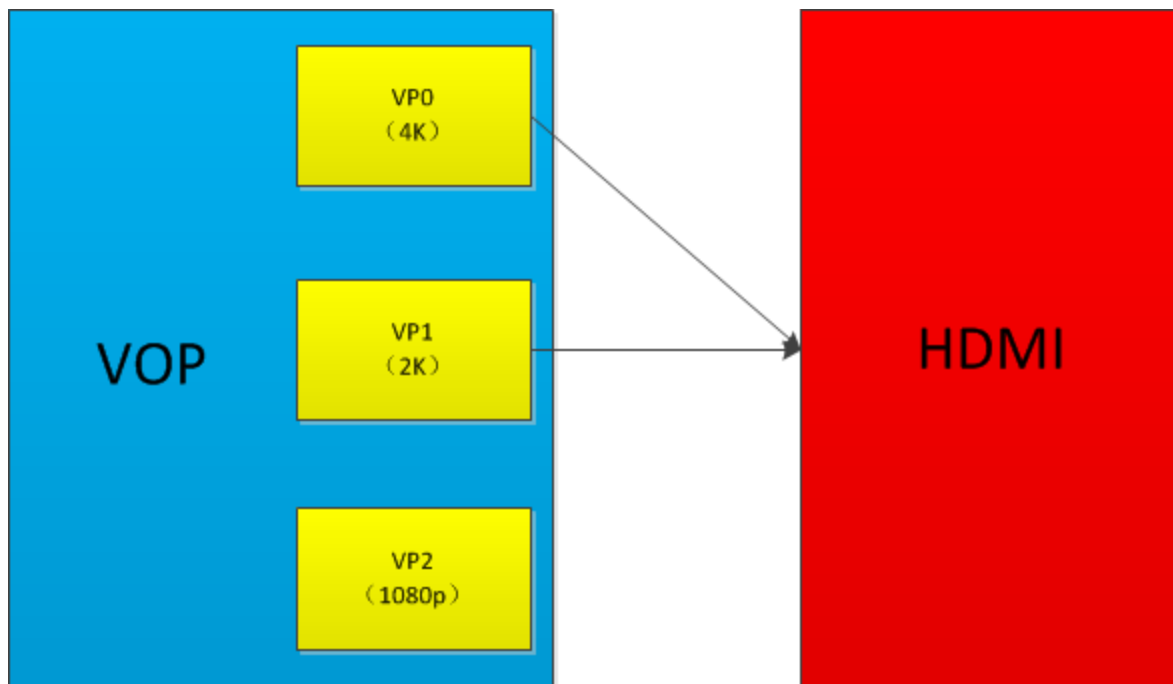
反之若绑定到 VOPL 则添加：

```
&hdmi_in_vopb {
    status = "disabled";
};
```

如果平台只有一个 VOP，则不需要该步骤。

VOP2 及其后版本，一个平台不再有多多个 VOP。取而代之的是只有一个 VOP，而 VOP 中有多个 VP（Video Port）输出。

RK356X VOP 与 HDMI 的通路如图所示：



HDMI 可以绑定在 VP0 或 VP1 上，建议绑定 VP0 可以支持 4K 输出：

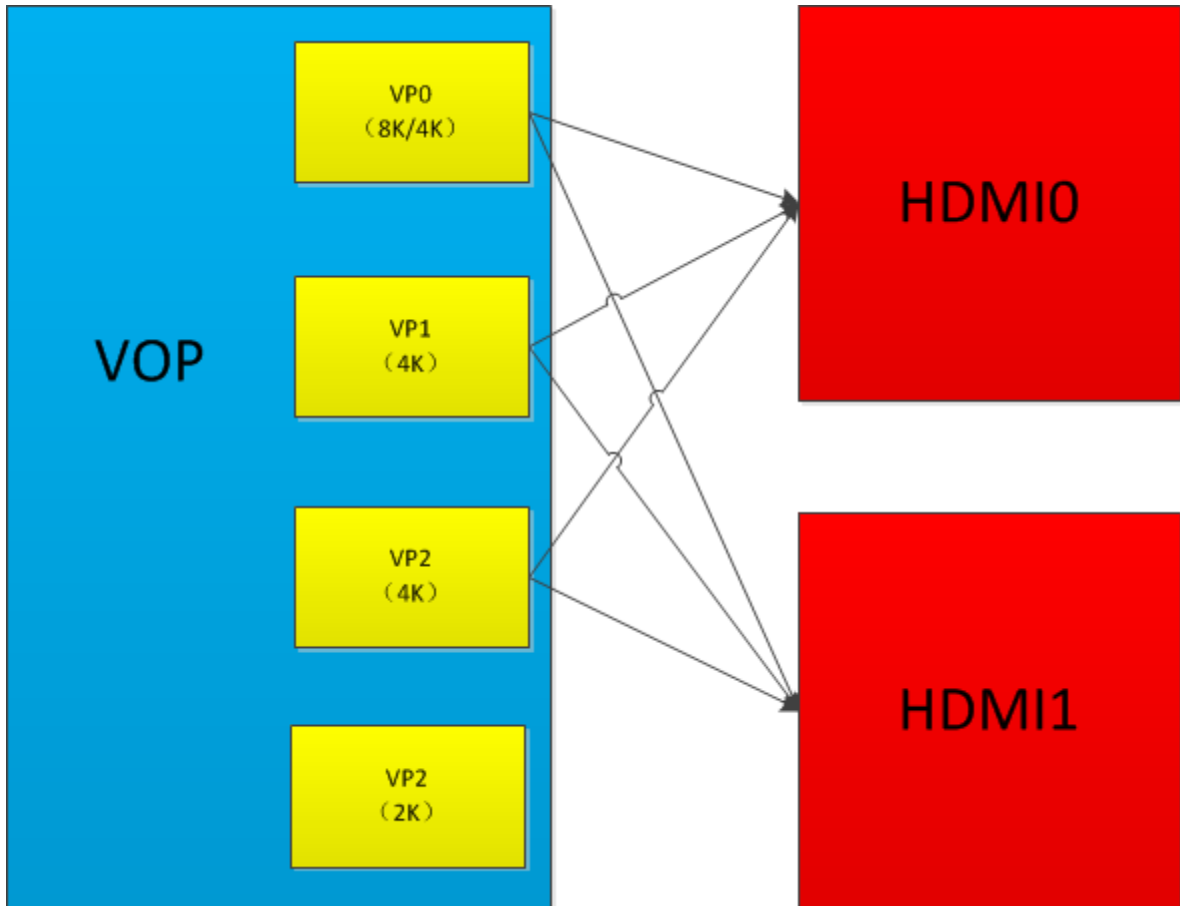
```

&hdmi_in_vp0 {
    status = "okay";
};

&hdmi_in_vp1 {
    status = "disabled";
};

```

RK3588 VOP 与 HDMI 的通路如图所示：



RK3588 有两个 HDMITX，两者在性能上完全相同，分别可以绑定在 VP0/1/2。

如果最高只需要输出 4K 分辨率，建议 HDMI0/1 分别绑定在 VP0/1：

```

&hdmi0_in_vp0 {
    status = "okay";
};

&hdmi0_in_vp1 {
    status = "disabled";
};

&hdmi0_in_vp2 {
    status = "disabled";
};

&hdmi1_in_vp1 {
    status = "okay";
};

&hdmi1_in_vp0 {

```



```

        status = "disabled";
    };

    &hdmi1_in_vp2 {
        status = "disabled";
    };

```

RK3588 平台如果需要输出 8K 分辨率，必须占用 VP0 和 VP1 两个 port 进行拼接。在 DTS 中，必须要将输出 8K 的 HDMI 绑定在 VP0 上，以 HDMI0 为例：

```

&hdmi0_in_vp0 {
    status = "okay";
};

&hdmi0_in_vp1 {
    status = "disabled";
};

&hdmi0_in_vp2 {
    status = "disabled";
};

```

同时，还需要将 VOP ACLK 设置为 800M，详见 3.1.4。

RK3588 HDMI 与 eDP 共用 COMBPHY，以 HDMI0 和 eDP0 为例，DTS 中 hdptxphy_hdmi0 为 HDMI PHY 节点，hdptxphy0 为 eDP PHY 节点：

```

hdptxphy0: phy@fed60000 {
    compatible = "rockchip,rk3588-hdptx-phy";
    reg = <0x0 0xfed60000 0x0 0x2000>;
    clocks = <&cru CLK_USB2PHY_HDPTXRXPHY_REF>, <&cru PCLK_HDPTX0>;
    clock-names = "ref", "apb";
    resets = <&cru SRST_P_HDPTX0>, <&cru SRST_HDPTX0_INIT>,
            <&cru SRST_HDPTX0_CMN>, <&cru SRST_HDPTX0_LANE>;
    reset-names = "apb", "init", "cmn", "lane";
    rockchip,grf = <&hdptxphy0_grf>;
    #phy-cells = <0>;
    status = "disabled";
};

hdptxphy_hdmi0: hdmiphy@fed60000 {
    compatible = "rockchip,rk3588-hdptx-phy-hdmi";
    reg = <0x0 0xfed60000 0x0 0x2000>;
    clocks = <&cru CLK_USB2PHY_HDPTXRXPHY_REF>, <&cru PCLK_HDPTX0>;
    clock-names = "ref", "apb";
    clock-output-names = "clk_hdmiphy_pixel0";
    #clock-cells = <0>;
    resets = <&cru SRST_HDPTX0>, <&cru SRST_P_HDPTX0>,
            <&cru SRST_HDPTX0_INIT>, <&cru SRST_HDPTX0_CMN>,
            <&cru SRST_HDPTX0_LANE>, <&cru SRST_HDPTX0_ROPLL>,
            <&cru SRST_HDPTX0_LCPPLL>;
    reset-names = "phy", "apb", "init", "cmn", "lane", "ropll",
            "lcppll";
    rockchip,grf = <&hdptxphy0_grf>;
    #phy-cells = <0>;
    status = "disabled";
};

```

```
};
```

所以当使用 HDMI 的时候必须关闭对应的 eDP 和 eDP PHY。以 HDMI0 为例：

```
&hdmi0 {
    status = "okay";
};

&hdptxphy_hdmi0 {
    status = "okay";
};

&edp0 {
    status = "disabled";
};

&hdptxphy0 {
    status = "disabled";
};
```

2.1.3 打开开机 logo

如果 U-Boot logo 未开启，那 kernel 阶段也无法显示开机 logo，只能等到系统启动后才能看到应用显示的图像。在 dts 里面将 `route_hdmi` 使能即可打开 U-Boot logo 支持：

```
&route_hdmi {
    status = "okay";
};
```

同时，在双 VOP 的平台，需要注意下方代码中的 `connect` 指定的 VOP 必须要与 HDMI 绑定的 VOP 一致（详见 3.1.2），否则可能出现花屏等问题。

```
route_hdmi: route-hdmi {
    status = "disabled";
    logo,uboot = "logo.bmp";
    logo,kernel = "logo_kernel.bmp";
    logo,mode = "center";
    charge_logo,mode = "center";
    connect = <&vopb_out_hdmi>;
};
```

2.1.4 VOP dclk 绑定 PLL

HDMI 绑定的 VOP/VP dclk，需要指定对应的 PLL 作为时钟源。在 RK 的平台中，RK322X/RK3328/RK3528 HDMI 都与固定的 VOP/VP 绑定，且固定使用 HDMI PHY PLL 作为 dclk 时钟源，无需配置。

2.1.4.1 RK3288 绑定 PLL

RK3288 VOPB/VOPL dclk 可以挂载到 GPLL/CPLL。由于这两个 PLL 都不能小数分频，所以 RK3288 的 HDMI 只能输出 594M 整数分频的标准分辨率（如 4K60/1080P60/720P60）。举例 VOPB dclk 挂载到 GPLL，VOPL dclk 挂载到 CPLL 如下：

```
&vopb {
    assigned-clocks = <&cru DCLK_VOP0>;
    assigned-clock-parents = <&cru PLL_GPLL>;
};

&voPl {
    assigned-clocks = <&cru DCLK_VOP1>;
    assigned-clock-parents = <&cru PLL_CPLL>;
};
```

2.1.4.2 RK3399 绑定 PLL

RK3399 的 HDMI 所绑定的 VOP dclk 需要挂载到 VPLL 上，若是双显则需要将另一个 VOP dclk 挂到 CPLL，这样可以分出任意频率的 dclk，实现双显任意分辨率的支持。如当 HDMI 绑定到 VOPB 时配置：

```
&vopb {
    assigned-clocks = <&cru DCLK_VOP0_DIV>;
    assigned-clock-parents = <&cru PLL_VPLL>;
};

&voPl {
    assigned-clocks = <&cru DCLK_VOP1_DIV>;
    assigned-clock-parents = <&cru PLL_CPLL>;
};
```

当 HDMI 绑定到 VOPL 时配置：

```
&vopb {
    assigned-clocks = <&cru DCLK_VOP0_DIV>;
    assigned-clock-parents = <&cru PLL_CPLL>;
};

&voPl {
    assigned-clocks = <&cru DCLK_VOP1_DIV>;
    assigned-clock-parents = <&cru PLL_VPLL>;
};
```

2.1.4.3 RK356X 绑定 PLL

RK356X HDMI 所绑定的 VP dclk 必须需要挂载到 HPLL 上，举例 VP0 dclk：

```
&vop {
    status = "okay";
    assigned-clocks = <&cru DCLK_VOP0>;
    assigned-clock-parents = <&pmucru PLL_HPLL>;
};
```

2.1.4.4 RK3588/RK3576 绑定 PLL

RK3588 如果需要输出 4K60 以上的分辨率，需要将 VOP ACLK 设置为 800Mhz:

```
&vop {
    assigned-clocks = <&cru ACLK_VOP>;
    assigned-clock-rates = <800000000>;
    status = "okay";
};
```

RK3576 VOP ACLK 无需专门配置。

RK3588 HDMI 0/1 可以绑定 VP 0/1/2 详见 2.1.2。这三个 VP dclk 都可以挂载到 GPLL/HDMI0 PHY PLL/HDMI 1 PHY PLL/V0PLL。

具体的分配策略和相关限制可以参考《Rockchip_Developer_Guide_DRM_Display_Driver_CN.pdf》中的 10.11 章节。

RK3588/RK3576 如果要支持非标准分辨率，需要指定 PHY PLL 作为 VOP dclk 时钟源。

RK3588 DTS 配置如下:

```
&display_subsystem {
    clocks = <&hdptxphy_hdmi_clk0>, <&hdptxphy_hdmi_clk1>;
    clock-names = "hdmi0_phy_pll", "hdmi1_phy_pll";
};

&hdptxphy_hdmi_clk0 {
    status = "okay";
};

&hdptxphy_hdmi_clk1 {
    status = "okay";
};
```

RK3576 DTS 配置如下:

```
&display_subsystem {
    clocks = <&hdptxphy_hdmi>;
    clock-names = "hdmi0_phy_pll";
};
```

DTS 时钟分配是否生效，可以使用命令 `cat /sys/kernel/debug/clk/clk_summary` 进行确认。

以 VP0 的 dclk 绑定 HDMI0 PHY PLL 为例，`dclk_vop0` 在时钟树中挂在 `clk_hdmiphy_pixel0` 下:

xin24m	26	28	0	24000000	0
0 50000					
clk_usbdpphy_mipidcpiphy_ref	2	2	0	24000000	0
0 50000					
clk_usb2phy_hdptxrphy_ref	11	11	0	24000000	0
0 50000					
clk_hdmiphy_pixel0	2	3	0	594000000	0
0 50000					
dclk_vop0	2	4	0	594000000	0
0 50000					

2.1.5 RK3288/RK3399/RK3528/RK356X HDCP 使能

2.1.5.1 HDCP 1.4 使能

```
&hdmi {  
    hdcplx-enable = <1>;  
}
```

使能 HDCP 1.4 后还需要通过工具烧录对应 key，工具可以在 SDK 的 RKTools 目录下获取。不同 Android 工具可能不同，可向 FAE 咨询，使用说明见工具的 readme。对应的 Key 需要客户自行向 Digital Content Protection LLC 申请。

通过以下节点可以开启或关闭 HDCP 功能：

```
echo 1 > /sys/class/misc/hdmi_hdcplx/enable
```

1 为开启 HDCP 功能，0 为关闭 HDCP 功能。

开启 HDCP 功能后，可以通过以下方法确认 HDCP 是否生效：

- 通过以下节点查看 HDCP 工作状态：

```
cat /sys/class/misc/hdmi_hdcplx/status
```

不同值对应的 HDCP 状态如下：

status 节点值	说明
hdcpl disable	HDCP 功能关闭。
hdcpl_auth_start	HDCP 开始认证。
hdcpl_auth_success	HDCP 认证成功，开始传输加密的视频数据。
hdcpl_auth_fail	HDCP 认证失败

- 分别找一台不支持 HDCP 1.4 的电视和支持 HDCP 1.4 的电视。若开启 HDCP 功能后，不支持 HDCP 1.4 的电视显示粉屏而支持 HDCP 1.4 的电视可以正常显示，则说明 HDCP 工作正常。

2.1.5.2 HDCP 2.2 使能

RK3288/RK3399/RK3528/RK356X 支持 DRM 框架下的 HDCP 2.2 功能，需要注意的是，想使用 HDCP 2.2 功能必须确保 HDCP 1.4 工作正常。想要开启该功能需要以下步骤：

- 向 FAE 申请 HDCP 2.2 Key 打包工具以及补丁包，按照 readme 将 Key 打包并打上补丁。
- 重新编译并烧写后，使用以下节点开启/关闭 HDCP 2.2 功能：

```
echo 1 > /sys/class/misc/hdcp2_node/enable
```

使能后可以通过以下方法判断 HDCP 2.2 是否正常工作：

- 分别找一台不支持 HDCP 2.2 的电视和支持 HDCP 2.2 的电视。若开启 HDCP 功能后，不支持 HDCP 2.2 的电视显示白屏而支持 HDCP 2.2 的电视可以正常显示，则说明 HDCP 工作正常。
- 通过以下节点获取 HDCP 2.2 工作状态：

```
cat /sys/class/misc/hdcp2_node/status
```

status 值	说明
hdcp2 auth sucess	认证成功。
no enable hdcp2	HDCP 2.2 已关闭。
hdcp2 no auth	HDMI 未连接或是设备不支持 HDCP 2.2。
no already auth sucess	认证失败。

- 如果出现了认证失败的情况，请在 redmine 上传以下 Log 文件：

```
/cache/hdcp_tx0.log
```

或是执行以下命令抓取 Log：

```
logcat -s HDMI_HDCP2
dmesg | grep hdcp
```

2.1.6 RK3588/RK3576 HDCP 使能

RK3588/RK3576 支持 HDCP 1.4/2.3，两者的使能需要调用 DRM 框架的 HDCP 接口。通过配置 DRM PROPERTY 进行 HDCP 开关以及 HDCP 状态查询。相关 DEMO 代码 `hdcpctest.c` 请向业务申请。调试时，可以使用 `modetest` 进行测试。

Android 系统 `modetest` 代码路径为：

```
external/libdrm/tests/modetest/
```

Linux 系统可以使用 `buildroot` 编译的 `modetest`，编译 `rootfs` 之后，代码路径为：

```
buildroot/output/rockchip_rk3588/build/libdrm-2.4.115/testst/modetest/
```

使用 `modetest` 测试 HDCP，需要用到以下属性：

```
Content Protection:
    flags: enum
    enums: Undesired=0 Desired=1 Enabled=2
    value: 2

Undesired: 关闭hdcp
Desired: 开启hdcp
Enabled: hdcp已经开启并认证成功

hdcp_encrypted:
```

```
flags: range
values: 0 2
value: 2
```

hdcp认证等级:

- 0: hdcp未认证。
- 1: hdcp1.4。
- 2: hdcp2.3。

HDCP 的开关可以使用命令 `modetest -w` 进行, 使用方式举例:

```
modetest -w 423:"Content Protection":1
----->开启hdcp, 优先2.3, 若电视不支持2.3或2.3认证失败, 自动切换到1.4。

modetest -w 423:"Content Protection":0
----->关闭hdcp。
```

其中 423 为 HDMI 的 `connector id`, 可以使用命令 `modetest -c` 进行查询:

```
rk3588_s:/ # modetest -c
trying to open device 'i915'...failed
trying to open device 'amdgpu'...failed
trying to open device 'radeon'...failed
trying to open device 'nouveau'...failed
trying to open device 'vmwgfx'...failed
trying to open device 'omapdrm'...failed
trying to open device 'exynos'...failed
trying to open device 'tilcdc'...failed
trying to open device 'msm'...failed
trying to open device 'sti'...failed
trying to open device 'tegra'...failed
trying to open device 'imx-drm'...failed
trying to open device 'rockchip'...done
Connectors:
id      encoder status      name      size (mm)      modes      encoders
423     422      connected   HDMI-A-1    0x0        6          422
  modes:
    name refresh (Hz) hdisp hss hse htot vdisp vss vse vtot)
    1920x1080 60 1920 2008 2052 2200 1080 1084 1089 1125 148500 flags: phsync,
pvsync; type: preferred
    1920x1080 50 1920 2448 2492 2640 1080 1084 1089 1125 148500 flags: phsync,
pvsync; type: driver
    1280x720 60 1280 1390 1430 1650 720 725 730 750 74250 flags: phsync, pvsync;
type: driver
    1280x720 50 1280 1720 1760 1980 720 725 730 750 74250 flags: phsync, pvsync;
type: driver
    720x576 50 720 732 796 864 576 581 586 625 27000 flags: nhsync, nvsync; type:
driver
    720x480 60 720 736 798 858 480 489 495 525 27000 flags: nhsync, nvsync; type:
driver
```

详细 HDCP 的使用说明见《Rockchip_RK3588_Developer_Guide_HDCP_CN.pdf》

2.1.7 DDC 的 I2C 速率配置

当插拔 HDMI 读取 EDID 失败时，可以尝试降低 DDC I2C 的速率

```
[ 163.026743] dwhdmi-rockchip fdea0000.hdmi: i2c read time out!  
[ 163.130075] dwhdmi-rockchip fdea0000.hdmi: i2c read time out!  
[ 163.233407] dwhdmi-rockchip fdea0000.hdmi: i2c read time out!  
[ 163.336741] dwhdmi-rockchip fdea0000.hdmi: i2c read time out!  
[ 163.440074] dwhdmi-rockchip fdea0000.hdmi: i2c read time out!  
[ 163.440110] dwhdmi-rockchip fdea0000.hdmi: failed to get edid
```

目前 I2C 速率通过 clk 高电平和低电平的时间来调整，如下为实测 I2C 速率为 50 khz 时候的配置。

```
&hdmi {  
    ddc-i2c-scl-high-time-ns = <9625>;  
    ddc-i2c-scl-low-time-ns = <10000>;  
}
```

调整 I2C 速率只需将这两个值按对应的比例修改即可，例如调整速率为 100 khz:

```
&hdmi {  
    ddc-i2c-scl-high-time-ns = <4812>;  
    ddc-i2c-scl-low-time-ns = <5000>;  
}
```

2.1.8 HDMI 信号强度配置

由于硬件走线差异,不同板子有可能需要不同的驱动强度配置，当遇到电视兼容性问题时可以尝试进行修改。

2.1.8.1 RK322X/RK3328/RK3528/RK356X 信号强度配置

2.1.8.1.1 RK322X 信号配置

RK322X HDMI 信号强度可通过 dts 的 `rockchip,phy-table` 属性配置，格式定义：

```
rockchip,phy-table =  
    <165000000 0xaa 0x00 0x44 0x44 0x00 0x00 0x00  
        0x00 0x00 0x00 0x00 0x00 0x00 0x00>,  
    <340000000 0xaa 0x15 0x6a 0xaa 0x00 0x00 0x00  
        0x00 0x00 0x00 0x00 0x00 0x00 0x00>,  
    <594000000 0xaa 0x15 0x7a 0xaa 0x00 0x00 0x00  
        0x00 0x00 0x00 0x00 0x00 0x00 0x00>;
```

以上表 340000000 这一栏为例：

参数	说明
340000000	表示该栏参数所对应的最大 tmds clock 频率。也即这里一栏配置适用于 tmds clock 低于 165Mhz 的分辨率
0xaa	data lane0 slew rate: bit[1:0] data lane1 slew rate: bit[3:2] data lane2 slew rate: bit[5:4] clock lane slew rate: bit[7:6] slew rate:调整上升沿和下降沿时间，值越大，时间越短。
0x15	data lane0 pre-emphasis: bit[1:0] data lane1 pre-emphasis: bit[3:2] data lane2 pre-emphasis: bit[5:4] pre-emphasis: 预加重，值越大预加重越大。
0x6a	data lane2 swing: bit[3:0] clock lane swing: bit[7:4] swing:幅值，值越大幅值越大。
0xaa	data lane0 swing: bit[3:0] data lane1 swing: bit[7:4] swing:幅值，值越大幅值越大。
后续所有0值	无效

2.1.8.1.2 RK3328 信号配置

RK3328 HDMI 信号强度可通过 dts 的 `rockchip,phy-table` 属性配置，格式定义:

```
rockchip,phy-table =
    <165000000 0x07 0x0a 0x0a 0x0a 0x00 0x00 0x08
        0x08 0x08 0x00 0xac 0xcc 0xcc 0xcc>,
    <340000000 0x0b 0x0d 0x0d 0x0d 0x07 0x15 0x08
        0x08 0x08 0x3f 0xac 0xcc 0xcd 0xdd>,
    <594000000 0x10 0x1a 0x1a 0x1a 0x07 0x15 0x08
        0x08 0x08 0x00 0xac 0xcc 0xcc 0xcc>;
```

以上表 340000000 这一栏为例:

参数	说明
340000000	表示该栏参数所对应的最大 tmds clock 频率。也即这里一栏配置适用于 tmds clock 低于 165Mhz 的分辨率
0x0b	clock lane swing swing:幅值， 值越大幅值越大。
0x0d	data lane2 swing swing:幅值， 值越大幅值越大。
0x0d	data lane1 swing swing:幅值， 值越大幅值越大。
0x0d	data lane0 swing swing:幅值， 值越大幅值越大。
0x07	data lane0 pre-emphasis mode: bit[0] data lane1 pre-emphasis mode: bit[1] data lane2 pre-emphasis mode: bit[2] pre-emphasis mode:0--full mode 1--half mode
0x15	data lane0 pre-emphasis level: bit[1:0] data lane1 pre-emphasis level: bit[3:2] data lane2 pre-emphasis level: bit[5:4] pre-emphasis level:预加重强度， 值越大加重强度越大， 为0时关闭预加重。
0x08	data lane2 pre-emphasis swing: bit[4:0] pre-emphasis swing:预加重幅值， 值越大幅值越大。
0x08	data lane1 pre-emphasis swing: bit[4:0] pre-emphasis swing:预加重幅值， 值越大幅值越大。
0x08	data lane0 pre-emphasis swing: bit[4:0] pre-emphasis swing:预加重幅值， 值越大幅值越大。
0x3f	data lane0 pre-emphasis driver path gate: bit[0], main driver path gate: bit[1] data lane1 pre-emphasis driver path gate: bit[2], main driver path gate: bit[3] data lane2 pre-emphasis driver path gate: bit[4], main driver path gate: bit[5] clock lane pre-emphasis driver path gate: bit[6], main driver path gate: bit[7] 写1 enable path。
0xac	data lane2 pre-emphasis second delay gate: bit[0], pre-emphasis first delay gate: bit[1] data lane2 main driver second delay gate: bit[2], main driver first delay gate: bit[3] clock lane pre-emphasis second delay gate: bit[4], pre-emphasis first delay gate: bit[5] clock lane main driver second delay gate: bit[6], main driver first delay gate: bit[7] 写1 enable delay。
0xcc	data lane0 pre-emphasis second delay gate: bit[0], pre-emphasis first delay gate: bit[1] data lane0 main driver second delay gate: bit[2], main driver first delay gate: bit[3] data lane1 pre-emphasis second delay gate: bit[4], pre-emphasis first delay gate: bit[5] data lane1 main driver second delay gate: bit[6], main driver first delay gate: bit[7] 写1 enable delay。

参数	说明
0xcd	data lane2 delay time: bit[3:0] clock lane delay time: bit[7:4] delay time: 延迟时间，值越大延迟越大。
0xdd	data lane0 delay time: bit[3:0] data lane1 delay time: bit[7:4] delay time: 延迟时间，值越大延迟越大。

2.1.8.1.3 RK3528 信号配置

RK3528 HDMI 信号强度可通过 dts 的 `rockchip,phy-table` 属性配置，格式定义：

```
rockchip,phy-table =
    <165000000 0x03 0x04 0x0c 0x12 0x00 0x00 0x00
      0x00 0x00 0x00 0x00 0x00 0x00 0x00>,
    <340000000 0x03 0x04 0x0c 0x12 0x00 0x00 0x00
      0x00 0x00 0x00 0x00 0x00 0x00 0x00>,
    <594000000 0x02 0x08 0x0d 0x18 0x00 0x00 0x00
      0x00 0x00 0x00 0x00 0x00 0x00 0x00>;
```

以上表 340000000 这一栏为例：

参数	说明
340000000	表示该栏参数所对应的最大 tmds clock 频率。也即这里一栏配置适用于 tmds clock 低于 165Mhz 的分辨率
0x03	clock lane current bias control: 0x00:320uA 0x0f:920uA 步进40uA
0x04	data lane current bias control: 0x00:320uA 0x0f:920uA 步进40uA
0x0c	clock lane swing: bit[4:0] swing:幅值，值越大幅值越大。
0x12	data lane swing: bit[4:0] ESD event detection threshold: bit[7:5] swing:幅值，值越大幅值越大。 ESD event detection threshold:触发ESD事件阈值，值越大阈值越大。
0x00	Pre-cursor pre-emphasis:bit[2:0] Post-cursor pre-emphasis:bit[7:4] pre-emphasis:预加重，值越大预加重越强，0值关闭预加重。
后续所有0值	无效

2.1.8.2 RK3288/RK3368/RK3399/RK356X 信号配置

RK3288/RK3368/RK3399/RK356X HDMI 信号强度可通过 dts 的 `rockchip,phy-table` 属性配置，格式定义：

<PIXELCLOCK PHY_CKSYMTXCTRL PHY_TXTERM PHY_VLEVCTRL>。

PIXELCLOCK 表示该行参数所对应的最大 pixelclock 频率。

PHY_CKSYMTXCTRL 寄存器 (0x09) 值用于调整 HDMI 信号的预加重和上升斜率，加大预加重或 sloop boost，可以提升 DATA 信号的上升/下降斜率，但会降低信号的上升/下降时间：

Bit[0]: CLOCK 信号使能。

Bit[3:1]: DATA 信号预加重。

Bit[4:5]: DATA 信号sloop boost。

PHY_TXTERM 寄存器 (0x19) 值用于调整端接电阻值

Bit[0:2]: 值越大，端接电阻值越大。

PHY_VLEVCTRL寄存器 (0x0e) 值用于调整 HDMI 的信号幅度，具体定义如下：Bit[0:4]: tmds_clk +/- 信号幅度，值越低，信号幅度越大；

Bit[5:9]: tmds_data +/- 信号幅度，值越低，信号幅度越大。

举例如下：

```
&hdmi {
    rockchip,phy-table =
        <74250000 0x8009 0x0004 0x0272>,
        <165000000 0x802b 0x0004 0x0209>,
        <297000000 0x8039 0x0005 0x028d>,
        <594000000 0x8039 0x0000 0x019d>,
        <000000000 0x0000 0x0000 0x0000>;
};
```

其中 <74250000 0x8009 0x0004 0x0272>，表示 pixeclock 为 74.25MHz (720p 分辨率) 以下是 PHY_CKSYMTXCTRL 寄存器值为0x8009；PHY_TXTERM 值为 0x0004; PHY_VLEVCTRL 值为 0x0272。修改后也可用 `cat /sys/kernel/debug/dw-hdmi/phy` 命令查看对应的寄存器值确认是否有修改成功。

2.1.9 新增特殊分辨率

2.1.9.1 新增特殊分辨率时序

DRM 框架目前代码已经支持了绝大部分分辨率时序，但是部分 HDMI 屏幕旋转的场景下，可能还有一些特殊分辨率不支持。需要在 `kernel\drivers\gpu\drm\drm_edid.c` 中的 `drm_dmt_modes` 的末尾新增项目：

```
/* 0x58 - 4096x2160@59.94Hz RB */
{ DRM_MODE("4096x2160", DRM_MODE_TYPE_DRIVER, 556188, 4096, 4104,
    4136, 4176, 0, 2160, 2208, 2216, 2222, 0,
    DRM_MODE_FLAG_PHSYNC | DRM_MODE_FLAG_NVSYNC) },
```

参数	说明
"4096x2160"	mode name，为分辨率的 hdisplay x vdisplay
DRM_MODE_TYPE_DRIVER	mode type，配置为 DRM_MODE_TYPE_DRIVER
556188	像素时钟
4096	行有效像素
4104	行同步起始像素
4136	行同步结束像素
4176	一行总像素
0	hskew，通常为 0
2160	帧有效行
2208	帧同步开始行
2216	帧同步结束行
2222	一帧总行数
0	vscan，通常为 0
vrefresh	显示设备帧率
DRM_MODE_FLAG_PHSYNC DRM_MODE_FLAG_NVSYNC	hsync 和 vsync 极性，flags 的定义如下： DRM_MODE_FLAG_PHSYNC (1<<0) DRM_MODE_FLAG_NHSYNC (1<<1) DRM_MODE_FLAG_PVSYNC (1<<2) DRM_MODE_FLAG_NVSYNC (1<<3) DRM_MODE_FLAG_INTERLACE (1<<4)

各项参数描述如上表，具体时序的含义可以参考 3.2.4 节。

2.1.9.2 RK322X/RK3328/RK3528 新增 PLL 配置

RK322X/RK3328/RK3528 芯片新增特殊分辨率还需要新增 HDMI-PHY-PLL 的配置。具体配置的计算过程请参考 2.1.5.2 节。

DRM 框架需要新增 PHY 配置时，需在 PRE-PLL 的配置 TABLE: `pre_pll_cfg_table` 中新增对应的配置，而 POST-PLL 的配置 TABLE: `post_pll_cfg_table` 目前配置已经涵盖了 PHY 所支持的所有分辨率，无需再新增配置。路径为：

```

static const struct pre_pll_config pre_pll_cfg_table[] = {
    { 27000000, 27000000, 1, 90, 3, 2, 2, 10, 3, 3, 4, 0, 0},
    { 27000000, 33750000, 1, 90, 1, 3, 3, 10, 3, 3, 4, 0, 0},
    { 40000000, 40000000, 1, 80, 2, 2, 2, 12, 2, 2, 2, 0, 0},
    { 40000000, 50000000, 1, 100, 2, 2, 2, 1, 0, 0, 15, 0, 0},
    { 59341000, 59341000, 1, 98, 3, 1, 2, 1, 3, 3, 4, 0, 0xE6AE6B},
    { 59400000, 59400000, 1, 99, 3, 1, 1, 1, 3, 3, 4, 0, 0},
    { 59341000, 74176250, 1, 98, 0, 3, 3, 1, 3, 3, 4, 0, 0xE6AE6B},
    { 59400000, 74250000, 1, 99, 1, 2, 2, 1, 3, 3, 4, 0, 0},
    { 65000000, 65000000, 1, 130, 2, 2, 2, 1, 0, 0, 12, 0, 0},
    { 65000000, 81250000, 3, 325, 0, 3, 3, 1, 0, 0, 10, 0, 0},
    { 71000000, 71000000, 3, 284, 0, 3, 3, 1, 0, 0, 8, 0, 0},
    { 71000000, 88750000, 3, 355, 0, 3, 3, 1, 0, 0, 10, 0, 0},
    { 74176000, 74176000, 1, 98, 1, 2, 2, 1, 2, 3, 4, 0, 0xE6AE6B},
    { 74250000, 74250000, 1, 99, 1, 2, 2, 1, 2, 3, 4, 0, 0},
    { 74176000, 92720000, 4, 494, 1, 2, 2, 1, 3, 3, 4, 0, 0x816817},
    { 74250000, 92812500, 4, 495, 1, 2, 2, 1, 3, 3, 4, 0, 0},
    { 83500000, 83500000, 2, 167, 2, 1, 1, 1, 0, 0, 6, 0, 0},
    { 83500000, 104375000, 1, 104, 2, 1, 1, 1, 1, 0, 5, 0, 0x600000},
    { 85750000, 85750000, 3, 343, 0, 3, 3, 1, 0, 0, 8, 0, 0},
    { 88750000, 88750000, 3, 355, 0, 3, 3, 1, 0, 0, 8, 0, 0},
    { 88750000, 110937500, 1, 110, 2, 1, 1, 1, 1, 0, 5, 0, 0xF00000},
    { 108000000, 108000000, 1, 90, 3, 0, 0, 1, 0, 0, 5, 0, 0},
    { 108000000, 135000000, 1, 90, 0, 2, 2, 1, 0, 0, 5, 0, 0},
    { 119000000, 119000000, 1, 119, 2, 1, 1, 1, 0, 0, 6, 0, 0},
    { 119000000, 148750000, 1, 99, 0, 2, 2, 1, 0, 0, 5, 0, 0x2AAAAA},
    { 148352000, 148352000, 1, 98, 1, 1, 1, 1, 2, 2, 2, 0, 0xE6AE6B},
    { 148500000, 148500000, 1, 99, 1, 1, 1, 1, 2, 2, 2, 0, 0},
    { 148352000, 185440000, 4, 494, 0, 2, 2, 1, 3, 2, 2, 0, 0x816817},
    { 148500000, 185625000, 4, 495, 0, 2, 2, 1, 3, 2, 2, 0, 0},
    { 162000000, 162000000, 1, 108, 0, 2, 2, 1, 0, 0, 4, 0, 0},
    { 162000000, 202500000, 1, 135, 0, 2, 2, 1, 0, 0, 5, 0, 0},
    { 296703000, 296703000, 1, 98, 0, 1, 1, 1, 0, 2, 2, 0, 0xE6AE6B},
    { 297000000, 297000000, 1, 99, 0, 1, 1, 1, 0, 2, 2, 0, 0},
    { 296703000, 370878750, 4, 494, 1, 2, 0, 1, 3, 1, 1, 0, 0x816817},
    { 297000000, 371250000, 4, 495, 1, 2, 0, 1, 3, 1, 1, 0, 0},
    { 593407000, 296703500, 1, 98, 0, 1, 1, 1, 0, 2, 1, 0, 0xE6AE6B},
    { 594000000, 297000000, 1, 99, 0, 1, 1, 1, 0, 2, 1, 0, 0},
    { 593407000, 370879375, 4, 494, 1, 2, 0, 1, 3, 1, 1, 1, 0x816817},
    { 594000000, 371250000, 4, 495, 1, 2, 0, 1, 3, 1, 1, 1, 0},
    { 593407000, 593407000, 1, 98, 0, 2, 0, 1, 0, 1, 1, 0, 0xE6AE6B},
    { 594000000, 594000000, 1, 99, 0, 2, 0, 1, 0, 1, 1, 0, 0},
    { ~0UL, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
};

static const struct post_pll_config post_pll_cfg_table[] = {
    {33750000, 1, 40, 8, 1},
    {33750000, 1, 80, 8, 2},
    {33750000, 1, 10, 2, 4},
    {74250000, 1, 40, 8, 1},
    {74250000, 18, 80, 8, 2},
    {74250000, 1, 20, 4, 8},
    {148500000, 2, 40, 4, 3},
    {148500000, 1, 10, 2, 8},
};

```

```

{297000000, 4, 40, 2, 3},
{297000000, 2, 20, 2, 8},
{594000000, 8, 40, 1, 3},
{594000000, 4, 20, 1, 8},
{      ~0UL, 0,  0, 0, 0}
};

```

`struct pre_pll_config` 和 `struct post_pll_config` 定义如下，LINUX 4.4/4.19 内核相当于拆分了 3.10 内核中的 `struct ext_pll_config_tab`。

```

struct pre_pll_config {
    unsigned long pixclock;
    unsigned long tmdsclock;
    u8 prediv;
    u16 fbdiv;
    u8 tmds_div_a;
    u8 tmds_div_b;
    u8 tmds_div_c;
    u8 pclk_div_a;
    u8 pclk_div_b;
    u8 pclk_div_c;
    u8 pclk_div_d;
    u8 vco_div_5_en;
    u32 fracdiv;
};

struct post_pll_config {
    unsigned long tmdsclock;
    u8 prediv;
    u16 fbdiv;
    u8 postdiv;
    u8 version;
};

```

`pre_pll_config` 各项参数说明见下表：

参数	说明
pixclock	HDMI输出分辨率的pixel clock
tmdsclock	HDMI输出分辨率的tmds clock
prediv	pre-pll-pre-divider
fbdiv	pre-pll-feedback-divider
tmds_div_a	tmds-dividera
tmds_div_b	tmds-dividerb
tmds_div_c	tmds-dividerc
pclk_div_a	pclk-dividera
pclk_div_b	pclk-dividerb
pclk_div_c	pclk-dividerc
pclk_div_d	pclk-dividerd
vco_div_5_en	pin_hd20_pclk是否直接由VCO / 5所得，特定clock情况下使用
fracdiv	pre-pll-fractional-feedback-divider

`post_pll_config` 各项参数说明见下表：

参数	说明
tmdsclock	HDMI输出分辨率的tmds clock
prediv	post-pll-pre-divider
fbdiv	post-pll-feedback-divider
postdiv	post-pll-post-divider
version	芯片版本，POST-PLL配置需根据时钟和芯片版本确定，其值含义： 1--RK322X与RK322XH早期样片，tmds clock为74.25Mhz及以下的配置 2--RK322XH量产芯片，tmds clock为74.25Mhz及以下的配置 3--RK322X与RK322XH芯片，tmds clock为74.25Mhz以上的配置，两者配置相同 4--RK322X部分芯片POST VCO为1080Mhz时不稳定，为270Mhz时工作稳定，需要特别区分出来 8--RK3528专有配置

以 TMDS CLOCK 为 74.25Mhz，RK3328 量产芯片为例，POST-PLL 配置选择方法如下：

- 首先在 `post_pll_cfg_table` 中根据 TMDS CLOCK 找到对应的区间。如 TMDS CLOCK 为 74.25Mhz 时， $33.75\text{Mhz} < \text{TMDS CLOCK} \leq 74.25\text{Mhz}$ ，找到对应的二项：

```
{74250000, 1, 40, 8, 1},  
{74250000, 18, 80, 8, 2},
```

- 根据芯片版本进一步选择，此时是 RK3328 量产芯片，TMDS CLOCK $\leq 74.25\text{Mhz}$ ，所以 version 的值应选择 2，所以最终选择：


```
{74250000, 18, 80, 8, 2},
```

3. 最终配置值为: `prediv = 18`, `fbdiv = 80`, `postdiv = 8`。在 LINUX 3.10 内核的驱动中对应 `struct ext_pll_config_tab` 中的 `pll_nd`, `pll_nf`, `pll_no` 三项。由于是 RK328 量产芯片切 TMD5 `CLOCK <= 74.25Mhz`, 所以需要添加在 `RK322XH_V1_PLL_TABLE` 当中。

2.1.9.3 RK3288/RK3368/RK3399/RK356X 新增 PLL 配置

RK3288/RK3368/RK3399/RK356X 的 HDMI-PHY-PLL 配置保存在 `rockchip_mpll_cfg` 和 `rockchip_mpll_cfg_420` 中:

```
static const struct dw_hdmi_mpll_config rockchip_mpll_cfg[] = {
    {
        30666000, {
            { 0x00b3, 0x0000 },
            { 0x2153, 0x0000 },
            { 0x40f3, 0x0000 },
        },
    }, {
        36800000, {
            { 0x00b3, 0x0000 },
            { 0x2153, 0x0000 },
            { 0x40a2, 0x0001 },
        },
    }, {
        46000000, {
            { 0x00b3, 0x0000 },
            { 0x2142, 0x0001 },
            { 0x40a2, 0x0001 },
        },
    }, {

```

路径为:

```
kernel/drivers/gpu/drm/rockchip/dw_hdmi-rockchip.c
```

其中 `rockchip_mpll_cfg` 为 RGB/YUV444/YUV422 的配置, `rockchip_mpll_cfg_420` 为 YUV420 的配置。

结构体 `dw_hdmi_mpll_config` 定义如下:

```
struct dw_hdmi_mpll_config {
    unsigned long mpixelclock;
    struct {
        ul6 cpce;
        ul6 gmp;
    } res[DW_HDMI_RES_MAX];
};
```

各项参数说明如下:

参数	说明
mpixelclock	像素时钟
cpce	OPMODE_PLLCFG 寄存器值
gmp	PLLGMPCTRL 寄存器值

以 `rockchip_mpll_cfg` 中的第一项配置为例：

```
static const struct dw_hdmi_mpll_config rockchip_mpll_cfg[] = {
    {
        30666000, {
            { 0x00b3, 0x0000 },
            { 0x2153, 0x0000 },
            { 0x40f3, 0x0000 },
        },
    },
    {
```

首先，HDMI 驱动会判断是否颜色格式为 YUV420，若是，则选择 `rockchip_mpll_cfg_420`，否则选择 `rockchip_mpll_cfg` 30666000 表示像素时钟为 30666000 及以下的分辨率适用该项配置，`{ 0x00b3, 0x0000 }`、`{ 0x2153, 0x0000 }`、`{ 0x40f3, 0x0000 }` 三项依次对应色深为 8 BIT、10 BIT、12 BIT（目前 Rockchip 方案实际只支持 8/10 bit 两种模式）情况下使用的配置。

由于参数的取值需要查阅 PHY 的 DATASHEET 获取，若需要新增 HDMI-PHY-PLL 配置，可以向 FAE 提出所需的像素时钟。然后根据上述的规则，将新增的配置添加到 `rockchip_mpll_cfg` 或 `rockchip_mpll_cfg_420` 中。

2.1.9.4 RK3588/RK3576 新增 PLL 配置

RK3588/RK3576 驱动支持自动计算 PLL 频率，当新增分辨率时不需要特意配置。

PLL 分配策略详见 2.1.4。

2.1.10 打开音频

3368 和 3288 默认 HDMI 声卡和 Codec 公用，需确认配置如下：

```
&hdmi_analog_sound {
    status = "okay";
}
```

3399 目前 HDMI 声卡和 DP 公用：

```
&hdmi_dp_sound {
    status = "okay";
};
```

2.2 Android 显示框架配置

Rockchip 在 Android 显示框架增加了一些系统属性，用于帮助客户能够根据需求配置显示。

2.2.1 主副显示接口配置

属性	功能说明
<code>sys.hwc.device.primary</code> <code>vendor.hwc.device.primary</code> (Android 9.0 后使用)	设置显示接口做为主显
<code>sys.hwc.device.extend</code> <code>vendor.hwc.device.extend</code> (Android 9.0 后使用)	设置显示接口做为副显

以上两个属性的配置可加在产品配置目录下的system.prop里，如：

```
device/rockchip/rk3368/rk3368_box/system.prop
```

默认情况下(即以上属性未配置时)，不支持热拔插设备（如 CVBS/MIPI/LVDS 等）会作为主显，支持热拔插设备（如 HDMI/DP 等）会作为副显。

通常主、副显只配置一个显示接口，例如 RK3399 BOX SDK 默认采用的配置，HDMI 作为主显，DP 作为副显。

```
sys.hwc.device.primary=HDMI-A
sys.hwc.device.extend=DP
```

9.0之后属性改成：

```
vendor.hwc.device.primary=HDMI-A
vendor.hwc.device.extend=DP
```

当主/副显配置多个显示接口时，优先使用支持热拔插的设备。例如RK3368 BOX SDK默认采用的配置：

```
sys.hwc.device.primary=HDMI-A,TV
```

9.0之后属性改成：

```
vendor.hwc.device.primary=HDMI-A,TV
```

当 HDMI 插入时，主显使用 HDMI 作为显示，HDMI 拔出时，主显使用 CVBS 作为显示。

注意：由于主显的 framebuffer 分辨率无法动态更改，所以有两个或以上设备作为主显时，最好设定一个主显的 framebuffer 分辨率。

关于接口名称可以参见 `hardware/rockchip/hwcomposer/drmresources.cpp` 里的定义：

```
struct type_name connector_type_names[] = {
    { DRM_MODE_CONNECTOR_Unknown, "unknown" }, //未知接口
    { DRM_MODE_CONNECTOR_VGA, "VGA" }, //VGA
    { DRM_MODE_CONNECTOR_DVII, "DVI-I" }, //DVI, 暂不支持
    { DRM_MODE_CONNECTOR_DVID, "DVI-D" }, //DVI, 暂不支持
    { DRM_MODE_CONNECTOR_DVIA, "DVI-A" }, //DVI, 暂不支持
    { DRM_MODE_CONNECTOR_Composite, "composite" }, //不支持
    { DRM_MODE_CONNECTOR_SVIDEO, "s-video" }, //S端子
    { DRM_MODE_CONNECTOR_LVDS, "LVDS" }, //LVDS
}
```

```

{ DRM_MODE_CONNECTOR_Component, "component" },//分量信号YPbPr
{ DRM_MODE_CONNECTOR_9PinDIN, "9-pin DIN" },//不支持
{ DRM_MODE_CONNECTOR_DisplayPort, "DP" },//DP
{ DRM_MODE_CONNECTOR_HDMIA, "HDMI-A" },//HDMI A型口
{ DRM_MODE_CONNECTOR_HDMIB, "HDMI-B" },//HDMI B型口, 不支持
{ DRM_MODE_CONNECTOR_TV, "TV" },// CVBS
{ DRM_MODE_CONNECTOR_eDP, "eDP" },//EDP
{ DRM_MODE_CONNECTOR_VIRTUAL, "Virtual" },//不支持
{ DRM_MODE_CONNECTOR_DSI, "DSI" },//MIPI
};

```

2.2.2 主副显示接口查询

可以通过以下两个只读属性来分别查询主副显示输出接口的名称。

属性	功能说明
sys.hwc.device.main vendor.hwc.device.main(Android 9.0 后使用)	查询当前主显的输出接口
sys.hwc.device.aux vendor.hwc.device.main (Android 9.0 后使用)	查询当前副显的输出接口

2.2.3 Framebuffer 分辨率配置

Framebuffer 分辨率是 UI 绘制时采用的分辨率，与 HDMI 输出分辨率不同。Framebuffer 分辨率与 HDMI 输出分辨率不同时，会进行相应的缩放。可以通过配置以下属性来设置 Framebuffer 的分辨率：

```
persist.sys.framebuffer.main=1920x1080
```

9.0之后属性改为：

```
persist.vendor.framebuffer.main=1920x1080
```

2.2.4 分辨率过滤配置

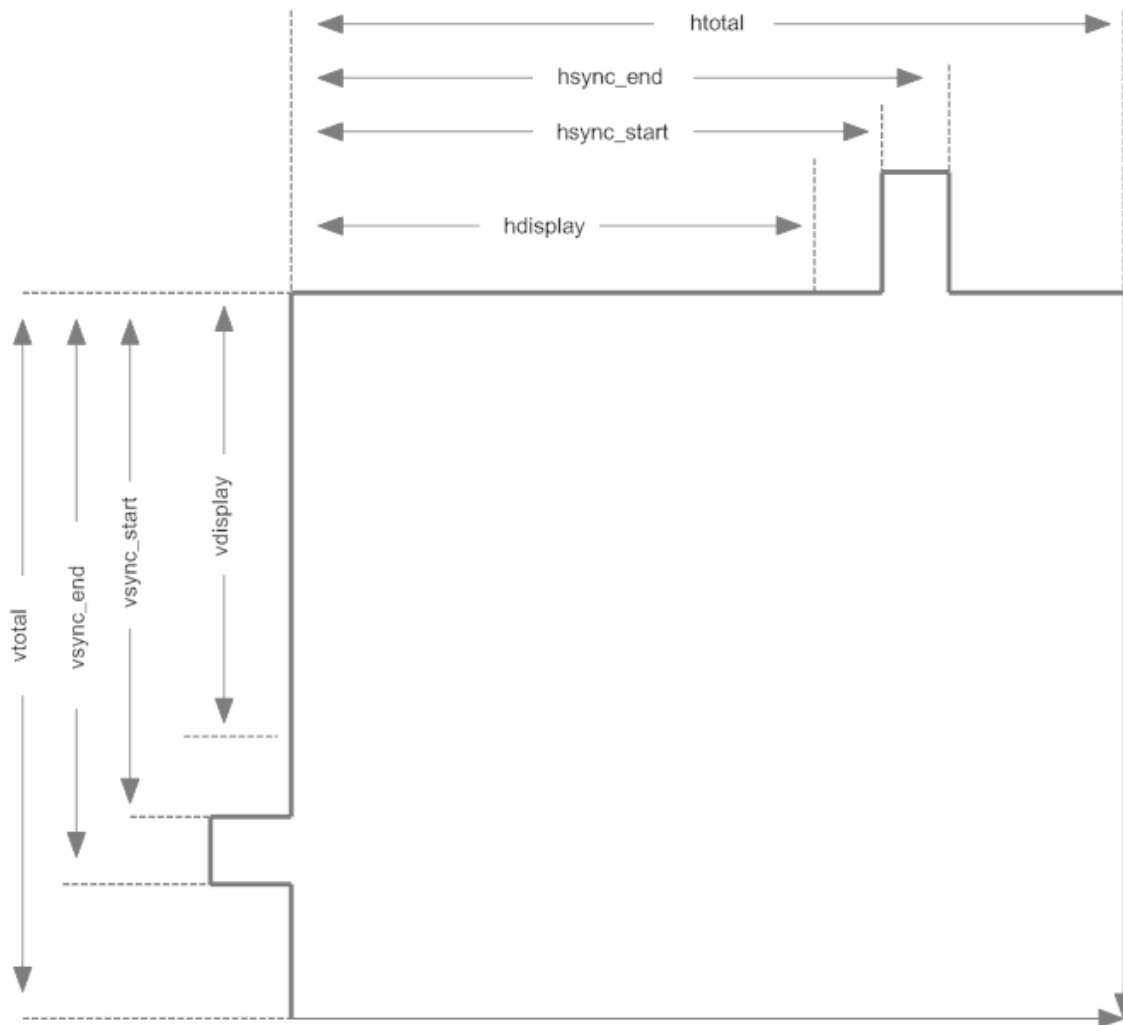
因为初始获取到的全部分辨率过多，有些分辨率对用户来说并不需要，因此在 SDK 的 HWC 模块中对分辨率进行了过滤。采用了白名单的方式对分辨率进行过滤：

```
device/rockchip/common/resolution_white.xml
```

HWC 中会根据该配置文件对初始的分辨率进行过滤筛选后再传递给上层，该 XML 文件的每一个 resolution 块定义了一个能够通过过滤的分辨率，其中详细项的定义如下：

项定义	说明
clock	像素时钟
hdisplay	行有效像素
hsync_start	行同步起始像素
hsync_end	行同步结束像素
htotal	一行总像素
hskew	行偏差，通常为 0
vdisplay	帧有效行
vsync_start	帧同步开始行
vsync_end	帧同步结束行
vtotal	一帧总行数
vscan	帧扫描信号，通常为 0
vrefresh	显示设备帧率
flags	flags 的定义如下： DRM_MODE_FLAG_PHSYNC (1<<0) DRM_MODE_FLAG_NHSYNC (1<<1) DRM_MODE_FLAG_PVSYNC (1<<2) DRM_MODE_FLAG_NVSYNC (1<<3) DRM_MODE_FLAG_INTERLACE (1<<4)
vic	HDMI 标准对应定义的 VIC 值，如HDMI标准中未定义置 0

具体时序说明见下图：



2.2.5 HDMI 设置选项

系统的设置 app 可以从 UI 上对当前的 HDMI 分辨率等属性进行修改。

要在设置中显示出 HDMI 选项，android7.X 是默认显示的；android 8.X 及以上需对 device 下的产品目录，添加配置属性如下：

```
BOARD_SHOW_HDMI_SETTING := true
```

UI 界面默认只显示副屏的配置，如要修改，请在 `package/apps/Settings` 的代码中，对 `HdmiSettings.java`，修改如下内容：

```
int value = SystemProperties.getInt("persist.hdmi.ui.state", ???);
```

代码中 `???` 的取值为 0：显示副屏配置 UI； 1：显示主屏配置 UI； 2：显示主副屏 2 个 UI 配置。

2.3 常用调试方法

2.3.1 查看 VOP 状态

执行以下命令可查看 VOP 状态:

```
cat /sys/kernel/debug/dri/0/summary
```

```
rk3399_box:/ # cat /d/dri/0/summary
VOP [ff900000.vop]: ACTIVE
Connector: HDMI-A
  bus_format[2025] output_mode[f]
Display mode: 1920x1080p60
  clk[148500] real_clk[148500] type[48] flag[5]
  H: 1920 2008 2052 2200
  V: 1080 1084 1089 1125
win0-0: ACTIVE
  format: NV12 little-endian (0x3231564e)
  zpos: 0
  src: pos[0x0] rect[3840x1080]
  dst: pos[0x0] rect[1920x1080]
  buf[0]: addr: 0x0000000000ebc7000 pitch: 7680 offset: 0
  buf[1]: addr: 0x0000000000ebc7000 pitch: 7680 offset: 8294400
win1-0: DISABLED
win2-0: ACTIVE
  format: AB24 little-endian (0x34324241)
  zpos: 1
  src: pos[0x0] rect[29x37]
  dst: pos[385x543] rect[29x37]
  buf[0]: addr: 0x0000000001abb000 pitch: 128 offset: 0
win2-0: DISABLED
win2-1: DISABLED
win2-2: DISABLED
win3-0: DISABLED
win3-0: DISABLED
win3-1: DISABLED
win3-2: DISABLED
VOP [ff8f0000.vop]: DISABLED
```

VOP状态

Connector状态

图层配置信息

VOP状态

上是 RK3399 连接 HDMI 时上述命令输出的 LOG，可以提供三种信息：

- VOP 状态：VOPB 处于使能状态，VOPL 处于禁用状态。
- VOP 对应的 Connector 状态：VOPB 输出信号给 HDMI，bus_format = 0x2025 表示 YUV444 8bit，output_mode = 0x0f 表示 VOP 输出总线为 ROCKCHIP_OUT_MODE_AAAA，输出 1920x1080P60。常用的 bus_format 由内核 `uapi/linux/media-bus-format.h` 定义：

```
#define MEDIA_BUS_FMT_RGB888_1X24      0x100a  //RGB888
#define MEDIA_BUS_FMT_RGB101010_1X30   0x1018  //RGB101010
#define MEDIA_BUS_FMT_YUV8_1X24        0x2025  //YUV444 8bit
#define MEDIA_BUS_FMT_YUV10_1X30       0x2016  //YUV444 10bit
#define MEDIA_BUS_FMT_UYYVYY8_0_5X24   0x2026  //YUV420 8bit
#define MEDIA_BUS_FMT_UYYVYY10_0_5X30   0x2027  //YUV420 10bit
```

常用的 output_mode 由内核 `drivers/gpu/drm/rockchip/rockchip_drm_vop.h` 定义：

```
#define ROCKCHIP_OUT_MODE_P888          0
#define ROCKCHIP_OUT_MODE_P666          1
#define ROCKCHIP_OUT_MODE_P565          2
#define ROCKCHIP_OUT_MODE_S888          8
#define ROCKCHIP_OUT_MODE_S888_DUMMY    12
#define ROCKCHIP_OUT_MODE_YUV420        14
/* for use special outface */
#define ROCKCHIP_OUT_MODE_AAAA          15
```

- 图层配置信息：win0 和 win2 使能，win2 buffer 格式为 ARGB，buffer 大小为 29x37；目标窗口为 29x37，窗口左上角坐标（385，543）。Win0 buffer 格式为 NV12，大小为 3840x2160；目标窗口大小为 1920x1080，窗口左上角坐标（0，0）。

2.3.2 查看 Connector 状态

`/sys/class/drm` 目录下可以看到驱动注册的各个显卡。下图是 RK3399 BOX 平台 DRM 目录结构，可以看到注册了 `card0-HDMI-A-1` 和 `card0-DP-1` 两种显示设备，分别表示 HDMI 和 DP。

```
rk3399_box:/ $ ls /sys/class/drm/
card0-DP-1/          card0/          renderD128/
card0-HDMI-A-1/      controlD64/     version
```

以 `card0-HDMI-A-1` 为例，其目录下有以下文件：

- Enabled:使能状态
- Status:连接状态
- Mode:当前输出分辨率
- Modes:连接设备支持的分辨率列表
- Audioformat:连接设备支持的音频格式
- Edid:连接设备的 EDID，可以通过命令 `cat edid > /data/edid.bin` 保存下来。

2.3.3 查看 HDMI 工作状态

如果包含以下提交，则可以查看 HDMI 工作状态：

```
commit eaca91814449199b1e6ad0b9fe0bba2215497c97
Author: Zheng Yang <zhengyang@rock-chips.com>
Date:   Mon Nov 27 16:56:21 2017 +0800

    drm: bridge: dw-hdmi: add hdmi status debugfs node
```

使用以下命令查看当前 HDMI 输出状态：

```
cat /sys/kernel/debug/dw-hdmi/status
```

```
HDMI Output Status: PHY disabled

HDMI Output Status: PHY enabled
Pixel Clk: 148500000Hz          TMDS Clk: 148500000Hz
Color Format: YUV444           Color Depth: 8 bit
Colorimetry: ITU.BT709        EOTF: SDR
x0: 0                          y0: 0
x1: 0                          y1: 0
x2: 0                          y2: 0
white x: 0                     white y: 0
max lum: 0                     min lum: 0
max cll: 0                     max fall: 0
```

- HDMI Output Status 表示当前 PHY 状态，只有当 PHY 使能的时候才会有后续打印。
- Pixel Clk 表示当前输出的像素时钟。
- TMDS Clk 表示当前输出 HDMI character rate。
- Color Format 表示输出的颜色格式，取值 RGB、YUV444、YUV422、YUV420。

- Color Depth 表示输出的颜色深度，取值 8bit、10bit、12bit、16bit。
- Colorimetry 表示输出的颜色标准，取值 ITU.BT601、ITU.BIT709、ITU.BT2020。
- EOTF 表示输出的 HDR 电光转换曲线方式，有如下取值：

EOTF	含义
Unsupported	HDMI不支持发送HDR信息
Not Defined	未定义
Off	不发送HDR信息
SDR	采用SDR曲线
ST2084	采用ST2084 EOTF曲线
HLG	采用HLG EOTF曲线

- (x0, y0)、(x1, y1)、(x2, y2)、(white x, white y)、max lum、min lum、max cll、maxfall为静态 HDR 描述子信息，只有 EOTF值为 SDR、ST2084、HLG 值时才会存在。

执行以下命令可以查看 HDMI 控制器寄存器：

```
cat /sys/kernel/debug/dw-hdmi/ctrl
```

可以使用命令来修改寄存器，例如要修改0x1000寄存器为 0xF8，输入命令：

```
echo 1000 f8 > /sys/kernel/debug/dw-hdmi/ctrl
```

RK3288、RK3368、RK3399 平台可以用以下命令查看 HDMI PHY 寄存器：

```
cat /sys/kernel/debug/dw-hdmi/phy
```

修改 PHY 寄存器与控制器类似，例如修改 0x06 寄存器为 0x8002，输入命令：

```
echo 06 8002 > /sys/kernel/debug/dw-hdmi/phy
```

2.3.4 查看 HDMI CEC 状态

执行以下命令查看 HDMI CEC 状态：

```
cat /sys/kernel/debug/cec/cec0/status
```

```
configured: 1
configuring: 0
phys_addr: 2.0.0.0
number of LAS: 1
LA mask: 0x0010
has CEC follower (in passthrough mode)
pending transmits: 0
```

打印结果如上图所示：

- configured 表示 cec adapter 是否配置完毕，1 为配置完毕，0 为未完毕。
- configuring 表示 cec adapter 是否正在配置，1 为正在，0 为配置完毕或未开始配置。

- `phys_addr` 表示 `cec` 的物理地址，未获取物理地址时为 `f.f.f.f`。
- `number of LAs` 表示该 `cec` 设备的逻辑地址数量，大部分设备为 1，极少数为 2。
- `LA mask` 表示当前绑定的逻辑地址，具体取值为（1 << 绑定的逻辑地址）。如：取值为 `0x0010` 时，右移 4 位后为 1，则说明当前的逻辑地址为 4，取值为 `0x0800` 时，右移 11 位后为 1，则说明当前的逻辑地址为 11.如未绑定任何逻辑地址时为 `0x0000`。
- `has CEC follower` 表示当前收到的 `cec` 消息交由上层用户空间处理，`in passthrough mode` 表示 `kernel` 不会处理 `cec core message`，而是全部上报给上层用户空间处理。
- `pending transmits` 表示当前还有多少待发送的 `cec` 消息。

2.3.5 强制使能/禁用 HDMI

强制使能 HDMI:

```
echo on > /sys/class/drm/card0-HDMI-A-1/status
```

强制禁用 HDMI:

```
echo off > /sys/class/drm/card0-HDMI-A-1/status
```

恢复检测热插拔:

```
echo detect > /sys/class/drm/card0-HDMI-A-1/status
```

2.3.6 命令行设置分辨率

在 `Android` 系统中，可以用命令行设置属性的方式来设置分辨率。此外，当用户在 `Android` 的 `setting` 中设置分辨率时，对应的属性值也会被设置，详见下面的说明。

2.3.6.1 Android 7.x & Android 8.x 分辨率设置

属性	说明
<code>persist.sys.resolution.main</code>	设置主屏分辨率，参数为该分辨率的时序，详见 3.2.4 节。
<code>persist.sys.resolution.aux</code>	设置副屏分辨率，参数为该分辨率的时序，详见 3.2.4 节。
<code>sys.display.timeline</code>	刷新显示时间线，每次设置新的分辨率需要加一。

通过 `persist.sys.resolution.main` 以及 `persist.sys.resolution.aux` 设置主副屏分辨率，每次设置完更新 `sys.display.timeline` (每次加 1) 再进行移动鼠标等更新 UI 的操作使新分辨率生效，例子如下:

- 设置4k60:

```
setprop persist.sys.resolution.main 3840x2160@60-3840-4016-4104-4400-2160-2168-2178-2250-5
setprop sys.display.timeline 1
```

- 设置1080p60:

```
setprop persist.sys.resolution.main 1920x1080@60-1920-2008-2052-2200-1080-1084-1089-1125-5
setprop sys.display.timeline 2
```

- 设置720P60:

```
setprop persist.sys.resolution.main 1280x720@60.00-1390-1430-1650-725-730-750-5
setprop sys.display.timeline 3
```

- 设置480P60:

```
setprop persist.sys.resolution.main 720x480@59.94-736-798-858-489-495-525-a
setprop sys.display.timeline 4
```

2.3.6.2 Android 9.0 及以上版本分辨率设置

Android 9.0 新旧版本配置分辨率的属性有所区别，需要根据 HWC 版本进行区分。新版 Android 9.0 SDK 以及更高 Android 版本都使用新属性。

```
commit 6f772a162893dd0242b6644fa32166e1ac15b2c0
Author: libin <bin.li@rock-chips.com>
Date: Wed May 12 15:00:05 2021 +0800

DrmConnector : Add UpdateDisplayMode function

In order to adapt to the three-screen version.

The hwc switching resolution will be in accordance with
the following priority:
1.persist.vendor.resolution.<Connector-type>-<Connector-ID>
2.persist.vendor.resolution.main(aux)
3.baseparameter(if exist)
4.use "Auto"

Version: 1.1.137

Signed-off-by: libin <bin.li@rock-chips.com>
Change-Id: I73e216047ed5423929d7a091572d717c4ffdf50c
```

当 HWC 代码包含以上提交时，使用的是新版属性：

属性	说明
persist.vendor.resolution.HDMI-A-0	设置 HDMI0 分辨率，若设置 HDMI1 则是 HDMI-A-1，参数为该分辨率的时序，详见 3.2.4 节
vendor.display.timeline	刷新显示时间线，每次设置新的分辨率需要加一。

如果 HWC 不含上述属性时，使用的是旧版属性：

属性	说明
<code>persist.vendor.resolution.main</code>	设置主屏分辨率，参数为该分辨率的时序，详见 3.2.4 节。
<code>persist.vendor.resolution.aux</code>	设置副屏分辨率，参数为该分辨率的时序，详见 3.2.4 节。
<code>vendor.display.timeline</code>	刷新显示时间线，每次设置新的分辨率需要加一。

通过 `persist.vendor.resolution.main` 以及 `persist.vendor.resolution.aux` 设置主副屏分辨率，每次设置完更新 `vendor.display.timeline` (每次加 1)再进行移动鼠标等更新UI 的操作使新分辨率生效，例子如下：

- 设置4k60:

```
setprop persist.vendor.resolution.main 3840x2160@60-3840-4016-4104-4400-2160-2168-2178-2250-5
setprop vendor.display.timeline 1
```

- 设置1080p60:

```
setprop persist.vendor.resolution.main 1920x1080@60-1920-2008-2052-2200-1080-1084-1089-1125-5
setprop vendor.display.timeline 2
```

- 设置720P60:

```
setprop persist.vendor.resolution.main 1280x720@60.00-1390-1430-1650-725-730-750-5
setprop vendor.display.timeline 3
```

- 设置480P60:

```
setprop persist.vendor.resolution.main 720x480@59.94-736-798-858-489-495-525-a
setprop vendor.display.timeline 4
```

2.3.7 命令行设置颜色

在 Android 系统中，可以用命令行设置属性的方式来设置颜色。此外，当用户在 Android 的 setting 中设置颜色时，对应的属性值也会被设置，详见下面的说明。

2.3.7.1 Android 7.x & Android 8.x 颜色设置

属性	说明
<code>persist.sys.color.main</code>	设置主屏颜色，参数为：颜色格式-色深 例如想要设置颜色为 RGB，色深 8 bit（24 bit），则参数为 RGB-8bit 支持颜色格式有： RGB YCBCR444 YCBCR422 YCBCR420 支持色深有： 8bit 10bit
<code>persist.sys.color.aux</code>	设置副屏颜色，参数同主屏。
<code>sys.display.timeline</code>	刷新显示时间线，每次设置新的分辨率需要加一。

通过 `persist.sys.color.main` 以及 `persist.sys.color.aux` 设置主副屏颜色，每次设置完更新 `sys.display.timeline` (每次加 1) 再进行移动鼠标等更新 UI 的操作使新颜色生效，例子如下：

```
setprop persist.sys.color.main RGB-8bit
setprop sys.display.timeline 1
```

将输出颜色设置为 RGB，色深 8 bit（RGB 24 bit）。

2.3.7.2 Android 9.0 及以上版本颜色设置

与 3.3.6.2 中描述的设置分辨率相同，新旧 Android 版本使用的颜色设置属性不同。

```
commit 6f772a162893dd0242b6644fa32166e1ac15b2c0
Author: libin <bin.li@rock-chips.com>
Date:   Wed May 12 15:00:05 2021 +0800

    DrmConnector : Add UpdateDisplayMode function

    In order to adapt to the three-screen version.

    The hwc switching resolution will be in accordance with
    the following priority:
        1.persist.vendor.resolution.<Connector-type>-<Connector-ID>
        2.persist.vendor.resolution.main(aux)
        3.baseparameter(if exist)
        4.use "Auto"

    Version: 1.1.137

    Signed-off-by: libin <bin.li@rock-chips.com>
    Change-Id: I73e216047ed5423929d7a091572d717c4ffdf50c
```

当 HWC 代码包含以上提交时，使用的是新版属性：

属性	说明
<code>persist.vendor.color.HDMI-A-0</code>	设置 HDMI0 颜色，若设置 HDMI1 则是 HDMI-A-1，参数为：颜色格式-色深 例如想要设置颜色为 RGB，色深 8 bit（24 bit），则参数为 RGB-8bit 支持颜色格式有： RGB YCBCR444 YCBCR422 YCBCR420 支持色深有： 8bit 10bit
<code>vendor.display.timeline</code>	刷新显示时间线，每次设置新的分辨率需要加一。

如果 HWC 不含上述属性时，使用的是旧版属性：

属性	说明
<code>persist.vendor.color.main</code>	设置主屏颜色，参数为：颜色格式-色深 例如想要设置颜色为 RGB，色深 8 bit（24 bit），则参数为 RGB-8bit 支持颜色格式有： RGB YCBCR444 YCBCR422 YCBCR420 支持色深有： 8bit 10bit
<code>persist.vendor.color.aux</code>	设置副屏颜色，参数同主屏。
<code>vendor.display.timeline</code>	刷新显示时间线，每次设置新的分辨率需要加一。

通过 `persist.vendor.color.main` 以及 `persist.vendor.color.aux` 设置主副屏颜色，每次设置完更新 `vendor.display.timeline`（每次加 1）再进行移动鼠标等更新 UI 的操作使新颜色生效，例子如下：

```
setprop persist.vendor.color.main RGB-8bit
setprop vendor.display.timeline 1
```

将输出颜色设置为 RGB，色深 8 bit（RGB 24 bit）。

2.3.8 设置过扫描

由于不同电视之间存在差异，画面显示可能存在画面超出屏幕范围，或是画面有黑边等现象。此时可以设置过扫描来调整缩放大小，以修正这些问题。

在 Android 系统中，可以用命令行设置属性的方式来设置过扫描。此外，当用户在 Android 的 setting 中也可以设置过扫描，设置完成后，对应的属性值也会被设置，详见下面的说明。

2.3.8.1 Android 7.x & Android 8.x 过扫描设置

属性	说明
<code>persist.sys.overscan.main</code>	设置主屏过扫描，属性格式： <code>overscan left,top,right,bottom</code> <code>left</code> 、 <code>top</code> 、 <code>right</code> 、 <code>bottom</code> 分别为左、上、右、下四个方向的过扫描值，最小值为 1，最大值由属性 <code>persist.sys.overscan.max</code> 定义，如 <code>persist.sys.overscan.max</code> 不存在，默认取 100。
<code>persist.sys.overscan.aux</code>	设置副屏过扫描，参数同主屏。

例子如下：

```
setprop persist.sys.overscan.main "overscan 70,70,70,70"
```

将四个方向的过扫描设置为 70。

2.3.8.2 Android 9.0 及以上版本过扫描设置

参考 3.3.6.2 说明，根据 Android 版本选择属性进行设置。

新版属性：

属性	说明
<code>persist.vendor.overscan.HDMI-A-0</code>	设置 HDMI0 颜色，若设置 HDMI1 则是 HDMI-A-1，属性格式： <code>overscan left,top,right,bottom</code> <code>left</code> 、 <code>top</code> 、 <code>right</code> 、 <code>bottom</code> 分别为左、上、右、下四个方向的过扫描值，最小值为 1，最大值由属性 <code>persist.vendor.overscan.max</code> 定义，如 <code>persist.vendor.overscan.max</code> 不存在，默认取 100。

旧版属性：

属性	说明
<code>persist.vendor.overscan.main</code>	设置主屏过扫描，属性格式： <code>overscan left,top,right,bottom</code> <code>left</code> 、 <code>top</code> 、 <code>right</code> 、 <code>bottom</code> 分别为左、上、右、下四个方向的过扫描值，最小值为 1，最大值由属性 <code>persist.vendor.overscan.max</code> 定义，如 <code>persist.vendor.overscan.max</code> 不存在，默认取 100。
<code>persist.vendor.overscan.aux</code>	设置副屏过扫描，参数同主屏。

例子如下：

```
setprop persist.vendor.overscan.main "overscan 70,70,70,70"
```

将四个方向的过扫描设置为 70。

2.3.9 设置亮度、对比度、饱和度、色度

在 Android 系统中，可以用命令行设置属性的方式来设置这些参数。此外，当用户在 Android 的 setting 中也可以设置这些参数。设置完成后，对应的属性值也会被设置，详见下面的说明。

2.3.9.1 Android 7.x & Android 8.x 亮度、对比度、饱和度、色度设置

BCSH	取值范围	说明
亮度	整形数，0 - 100，默认值 50	persist.sys.brightness.main persist.sys.brightness.aux
对比度	整形数，0 - 100，默认值 50	persist.sys.contrast.main persist.sys.contrast.aux
饱和度	整形数，0 - 100，默认值 50	persist.sys.saturation.main persist.sys.saturation.aux
色度	整形数，0 - 100，默认值 50	persist.sys.hue.main persist.sys.hue.aux

举例如下：

```
setprop persist.sys.brightness.main 70
setprop vendor.display.timeline 1
```

通过 `persist.sys.brightness.main` 设置主屏亮度，每次设置完更新 `vendor.display.timeline` (每次加 1) 再进行移动鼠标等更新 UI 的操作使新亮度生效。

2.3.9.2 Android 9.0 及以上版本亮度、对比度、饱和度、色度设置

参考 3.3.6.2 说明，根据 Android 版本选择属性进行设置。

新版属性：

BCSH	取值范围	说明
亮度	整形数，0 - 100，默认值 50	persist.vendor.brightness.HDMI-A-0
对比度	整形数，0 - 100，默认值 50	persist.vendor.contrast.HDMI-A-0
饱和度	整形数，0 - 100，默认值 50	persist.vendor.saturation.HDMI-A-0
色度	整形数，0 - 100，默认值 50	persist.vendor.hue.HDMI-A-0

旧版属性：

BCSH	取值范围	说明
亮度	整形数，0 - 100，默认值 50	persist.vendor.brightness.main persist.vendor.brightness.aux
对比度	整形数，0 - 100，默认值 50	persist.vendor.contrast.main persist.vendor.contrast.aux
饱和度	整形数，0 - 100，默认值 50	persist.vendor.saturation.main persist.vendor.saturation.aux
色度	整形数，0 - 100，默认值 50	persist.vendor.hue.main persist.vendor.hue.aux

举例如下：

```
setprop persist.vendor.brightness.main 70
setprop sys.display.timeline 1
```

通过 `persist.vendor.brightness.main` 设置主屏亮度，每次设置完更新 `vendor.display.timeline` (每次加 1) 再进行移动鼠标等更新 UI 的操作使新亮度生效。

2.4 常见问题排查

2.4.1 插入或切换分辨率，电视提示无信号或格式不支持或画面不稳定，时有时无，或有大量彩色亮点、亮线

1. 确认HDMI当前的分辨率，命令见 3.3.2。
2. 降低HDMI分辨率，看电视能否正常显示，命令见 3.3.6。
3. 更换好的HDMI线，看电视能否正常显示。
4. 如果步骤2、3可以恢复画面，一般与 HDMI 物理信号的兼容性有关，需要检查硬件，测试 HDMI 信号进一步分析。
5. 如果 HDMI 信号不达标，可以通过调整 HDMI PHY 的配置进行信号的调整，参考 3.1.7。

2.4.2 播放视频时电视提示无信号或格式不支持

检查内核代码 `dtb` 中是否有针对视频的 DDR 变频功能，若有请设置 `auto-freq-en = <0>`；关闭自动变频功能。

```
dmc: dmc {
    compatible = "rockchip,rk3328-dmc";
    devfreq-events = <&dfi>;
    clocks = <&cru SCLK_DDRCLK>;
    clock-names = "dmc_clk";
    operating-points-v2 = <&dmc_opp_table>;
    ddr_timing = <&ddr_timing>;
    upthreshold = <40>;
    downdifferential = <20>;
    system-status-freq = <
        /*system status          freq(KHz) */
```

```

SYS_STATUS_NORMAL      786000
SYS_STATUS_REBOOT      786000
SYS_STATUS_SUSPEND     786000
SYS_STATUS_VIDEO_1080P 786000
SYS_STATUS_VIDEO_4K     786000
SYS_STATUS_VIDEO_4K_10B 933000
SYS_STATUS_PERFORMANCE 933000
SYS_STATUS_BOOST       933000

>;
auto-min-freq = <786000>;
auto-freq-en = <0>;
#cooling-cells = <2>;
status = "disabled";

```

2.4.3 部分电视提示无信号、黑屏、花屏

1. 确认HDMI当前的分辨率，命令见 3.3.2。
2. 降低HDMI分辨率，看电视能否正常显示，命令见 3.3.6。
3. 更换好的HDMI线，看电视能否正常显示。
4. 如果步骤 2、3 可以恢复画面，一般与 HDMI 物理信号的兼容性有关，需要检查硬件，测试 HDMI 信号进一步分析。
5. 如果 HDMI 信号不达标，可以通过调整 HDMI PHY 的配置进行信号的调整，参考 3.1.7。
6. 若测试 HDMI 信号达标，可尝试修改以下寄存器值：

```
kernel\drivers\gpu\drm\bridge\synopsys\dw-hdmi.c
```

```

/* HDMI Initialization Step B.4 */
static void dw_hdmi_enable_video_path(struct dw_hdmi *hdmi)
{
    /* control period minimum duration */
    hdmi_writeb(hdmi, 12, HDMI_FC_CTRLDUR);
}

```

将 HDMI_FC_CTRLDUR 的值逐步增大（最大 223）看显示是否能恢复正常。

2.4.4 读取 EDID 失败时，如何设置默认分辨率

```

Commit 727e0fe68d8f422698f4e257cb7c04f90b8692c0
Author: xuhuicong xhc@rock-chips.com
Date: Tue Sep 26 17:32:56 2017 +0800
drm/edid: output common tv resolution and hdmi mode if no read the correct edid
Change-Id: Ib7379340e8c1d59382553d21b60165fe5fb371e8
Signed-off-by: xuhuicong xhc@rock-chips.com

```

在有上面的提交基础上，修改 `def_modes` 的值，对应的是 VIC 值，如下面代码中的 4 对应 720P60 的分辨率。

```
kernel\drivers\gpu\drm\bridge\synopsys\dw-hdmi.c
```

```
static int dw_hdmi_connector_get_modes(struct drm_connector *connector)
{
    struct dw_hdmi *hdmi = container_of(connector, struct dw_hdmi,
                                         connector);

    struct edid *edid;
    struct drm_display_mode *mode;
    const u8 def_modes[6] = {4, 16, 31, 19, 17, 2};
    struct drm_display_info *info = &connector->display_info;
```

2.4.5 强制输出指定分辨率

当需要无视 EDID 的限制，强制输出某个分辨率时，RK3288/RK3368/RK3399/RK356X 确认当前代码是否包含以下提交：

```
commit b318f175080ca98173d75fcf436beeee64092303
Author: Algea Cao <algea.cao@rock-chips.com>
Date:   Fri Nov 17 09:55:52 2023 +0800

    drm/bridge: synopsys: Support hdmi force output

    Support hdmi output specific resolution and color format
    regardless of whether hdmi is connected.

    Signed-off-by: Algea Cao <algea.cao@rock-chips.com>
    Change-Id: I228a74d128aa818166f589798897729473d97610
```

RK3588/RK3576 确认当前代码是否包含以下提交：

```
commit ed5631fa515a297b45cef5a1330b516f56e113dc
Author: Algea Cao <algea.cao@rock-chips.com>
Date:   Thu Dec 28 10:16:17 2023 +0800

    drm/bridge: dw-hdmi-qp: Support hdmi force output

    Support hdmi output specific resolution and color format
    regardless of whether hdmi is connected.

    Signed-off-by: Algea Cao <algea.cao@rock-chips.com>
    Change-Id: I3f18aeb04427846e06b6a4397a4c6df77bbbcab2
```

若包含以上提交，则可以在 DTS 中的直接配置所需的分辨率及颜色格式：

```
&hdmi {
    status = "okay";
    force-output;
    force-bus-format = <MEDIA_BUS_FMT_RGB888_1X24>;
    force_timing {
        clock-frequency = <594000000>;
        hactive = <3840>;
        vactive = <2160>;
        hback-porch = <296>;
        hfront-porch = <176>;
```

```

vback-porch = <72>;
vfront-porch = <8>;
hsync-len = <88>;
vsync-len = <10>;
hsync-active = <1>;
vsync-active = <1>;
de-active = <0>;
pixelclk-active = <0>;

};

};

```

1. `force-output`：表示强制输出 HDMI 分辨率的 FLAG。
2. `force-bus-format`：强制输出指定颜色格式。
3. `force_timing`：强制输出指定的分辨率时序。

若当前代码不包含以上提交时，请做以下修改：

```
diff --git a/drivers/gpu/drm/bridge/synopsys/dw-hdmi.c b/drivers/gpu/drm/bridge/synopsys/dw-hdmi.c
index 5ed021b..9a7f18c 100644
--- a/drivers/gpu/drm/bridge/synopsys/dw-hdmi.c
+++ b/drivers/gpu/drm/bridge/synopsys/dw-hdmi.c
@@ -2497,7 +2497,7 @@ static int dw_hdmi_connector_get_modes(struct drm_connector *connector)
connector);

struct edid *edid;
struct drm_display_mode *mode;
const u8 def_modes[6] = {4, 16, 31, 19, 17, 2};
+ const u8 def_modes[6] = {107, 16, 31, 19, 17, 2};
struct hdr_static_metadata *metadata =
&connector->display_info.hdmi.hdr_panel_metadata;

int i, ret = 0;
@@ -2506,6 +2506,7 @@ static int dw_hdmi_connector_get_modes(struct drm_connector *connector)
return 0;

edid = drm_get_edid(connector, hdmi->ddc);
edid = NULL;
if (edid) {
dev_dbg(hdmi->dev, "got edid: width[%d] x height[%d]\n",
edid->width_cm, edid->height_cm);
diff --git a/drivers/gpu/drm/rockchip/dw_hdmi-rockchip.c b/drivers/gpu/drm/rockchip/dw_hdmi-rockchip.c
index bdc96cd4..6653dfa 100644
--- a/drivers/gpu/drm/rockchip/dw_hdmi-rockchip.c
+++ b/drivers/gpu/drm/rockchip/dw_hdmi-rockchip.c
@@ -488,10 +488,12 @@ dw_hdmi_rockchip_mode_valid(struct drm_connector *connector,
* If sink max TMDS clock < 340MHz, we should check the mode pixel
* clock > 340MHz is YCbCr420 or not.
*/
+#if 0
if (mode->clock > 340000 &&
connector->display_info.max_tmds_clock < 340000 &&
!drm_mode_is_420(&connector->display_info, mode))
return MODE_BAD;
+#endif

if (!encoder) {
const struct drm_connector_helper_funcs *funcs;
```

1. 把 `def_mode` 数组的第一个值改成所需分辨率对应的 VIC。
2. `edid = NULL`; 强制进入 EDID 读取失败的流程，不管有没有读到 EDID 都强制按 `def_modes` 的分辨率来显示。
3. 如果需要强制显示 4K 的分辨率，还需要注释掉上图的这段代码，解除对 4K 分辨率的限制。

需要注意的是，强制输出 HDMI 2.0（4K30以上）及以上的分辨率时，需要确认 SINK 端是否需要进行 SCDC 通信。如果需要 SCDC 通信，则必须保证 HDMI DDC 功能正常。且强制输出时，不支持 HDMI 热插拔。

2.4.6 Recovery HDMI 无显示

Recovery 下不支持双显，也不支持热插拔。若需要从 HDMI 显示，如果代码中没有如下修改，请添加如下的修改，然后插着 HDMI 开机。

```
diff --git a/minui/graphics_drm.cpp b/minui/graphics_drm.cpp
index 03e33b7..f4d5fc7 100644
--- a/minui/graphics_drm.cpp
+++ b/minui/graphics_drm.cpp
@@ -295,6 +295,7 @@ static drmModeConnector *find_main_monitor(int fd, drmModeRes *resources,
int modes;
/* Look for LVDS/eDP/DSI connectors. Those are the main screens. */
unsigned kConnectorPriority[] = {
+   DRM_MODE_CONNECTOR_HDMI_A,
   DRM_MODE_CONNECTOR_LVDS,
   DRM_MODE_CONNECTOR_eDP,
   DRM_MODE_CONNECTOR_DSI,
```

2.4.7 settings 无法设置 HDMI 分辨率

1. 请确认 3.2.1 中的主副屏配置是否正确，3.2.5 中的设置是否正确。
2. 请确认 3.3.6 中的属性配置是否正确。
3. 如果是 Android 9.X 及以上系统，需要启用 RkOutputManager 服务。3399 代码需更新到以下提交点。

```
commit 3550c7dd16aea55c67b58e45d42301d94216f665
Author: wh <wanghang@rock-chips.com>
Date:   Wed Apr 3 15:56:06 2019 +0800

    rk3399: mid add hdmi config

    Change-Id: I0705b537f9fe110bb06ae6390caaa8886f8d36a6
    Signed-off-by: wh <wanghang@rock-chips.com>

wh@ubuntu:~/3399_9.0/device/rockchip/rk3399$
```

4. 9.0 其他平台需打上相应的补丁，补丁为当前工程中执行完 source 和 lunch 相关的命令后，执行 `get_build_var DEVICE_MANIFEST_FILE`，会打印当前所使用的 manifest 文件，例如输出为：
`device/rockchip/common/manifest.xml`，就将下方代码添加到 manifest 文件中：

```
<hal format="hidl">
  <name>rockchip.hardware.outputmanager</name>
  <transport>hwbinder</transport>
  <version>1.0</version>
  <interface>
    <name>IRkOutputManager</name>
    <instance>default</instance>
  </interface>
</hal>
```

2.4.8 DDR 带宽不足导致的问题

如果在 4K 的高分辨下，出现闪屏或者闪绿线的问题，确认内核 log 是否有以下打印：

```
[drm:vop_isr] ERROR POST_BUF_EMPTY irq err
```

若有以上打印则为 DDR 带宽不足导致的问题，请参考

《Rockchip_RK3399_Developer_Guide_Android7.1_Software_CN&EN.pdf》的 9.7 节进行处理。

2.4.9 4K UI 相关问题

1. 是否一定要 4K UI?

4K UI 占用系统资源较多，最高只能支持到 4K25Hz 左右，不推荐使用 4K UI。如果只是想要播放 4K 视频或是查看 4K 图片，那可以不需要配置 4K UI，系统默认的视频播放器和图片浏览器可以支持。

2. 如何配置 4K UI?

请参考 3.2.3 节，将 Framebuffer 分辨率配置为 4K。

3. 配置成 4K UI 之后出现闪屏DDR 带宽问题，请参考 3.4.10 节进行处理。

2.4.10 setting 中 HDMI 分辨率列表中没有 4K 分辨率

1. 确认电视是否支持 4K 分辨率。

2. 执行以下命令，确认内核的 HDMI 分辨率列表中是否包含 4K 分辨率。

```
cat /sys/class/drm/card0-HDMI-A-1/modes
```

3. 如果上述列表中不包含 4K 分辨率，双 VOP 平台（RK3288、RK3399）请确认 HDMI 是否绑定的是 VOPB。或是该电视的 4K-50/60Hz 不支持 YUV420，当前平台不支持这么高的分辨率（平台支持 HDMI 最高分辨率请参考第 1 章表格）。

4. 若内核的 HDMI 分辨率列表中包含 4K 分辨率，而 setting 中的分辨率列表不包含该分辨率，请确认白名单中是否包含了该分辨率（参考 3.2.4 节）。

2.4.11 RK3588 setting 中 HDMI 分辨率列表中没有 8K 分辨率

1. 确认电视设置是否选择了 HDMI 2.1 模式或者游戏模式，大多数 8K 电视需要手动配置，否则不会有 8K 分辨率。

2. 确认 VOP ACLK 是否设置为了 800 Mhz，详见 3.1.5。

2.4.12 RK3588/RK3576 HDMI 8K/4K120 等 分辨率闪屏、显示异常

1. 确认使用的 HDMI 线缆是否为 HDMI 2.1 线缆。HDMI 2.1 分辨率必须使用 HDMI 2.1 线缆。

2. 提供 HDMI 原理图 / PCB 图供 RK 硬件审核。

3. 联系外部实验室进行 SI 测试或者向业务申请由 RK 硬件实验室进行 SI 测试。

2.4.13 HDMI 认证申请表的填写

如果要对设备进行 HDMI 认证，通常都会从认证机构获取到一份认证申请表，该表格一般都是 excel 形式。

首先关注表格下方的分页标签，如下方的例子：

General / Source CDF / Sink CDF / Repeater CDF / Rptr-Mini-Source / Rptr-Mini-Sink / Cable CDF / CEC General / CEC Features

HDMI 各项功能的认证需要填写各个分页中的内容，如果设备不支持某些功能，则对应分页不必填写。

常见分页说明如下：

- **General:** 一般申请表都有类似的页面，且必须填写。通常都是关于设备 HDMI 的一些基本信息。如设备有几个 HDMI IN 口、有几个 HDMI OUT 口、设备 HDMI 是否支持 HDCP, CEC 等功能。该分页选项往往也决定了后续分页是否需要填写。如 HDMI_input_count 填 0 的话，则说明设备不支持 HDMI IN，Sink CDF 分页就不必填写。

Source/Sink/Repeater Characteristics			
Must be filled out for any Source, Sink or Repeater. Do not fill out if Cable.			
Product Category			
Field Name	Field Definition	Choices	Value
HDMI_output_count	How many HDMI output ports are on the product?	0...X	1
HDMI_input_count	How many HDMI input ports are on the product?	0...X	0
HDCP_Supported	Is HDCP supported on the product?	Y/N	N
CEC Characteristics			
Field Name	Field Definition	Choices	Value
CEC_protocol	Is CEC protocol supported? If this field is "Y", then CEC CDF shall be submitted.	Y/N	N
Independent_CEC	Are the CEC signals on input connectors independent? (Meaning: no physical connection between inputs and the product has a logical address of 0 for all inputs). [Note: If the product has no HDMI inputs, answer "N".]	Y/N	N
CEC_root_device	Does the product act as a CEC root device? (Meaning: the product is a Sink or Repeater and the product's Physical Address is 0.0.0.0 and the product's EDID(s) [if present] contain Source Physical Address of P.0.0.0). [Note: If the product has no HDMI inputs, answer "N".]	Y/N	N
HEAC Characteristics			
Field Name	Field Definition	Choices	Value
HEC_supported	Is HEC supported? If this field is "Y", then HEAC CDF shall be submitted.	Y/N	N
ARC_supported	Is ARC supported? If this field is "Y", then HEAC CDF shall be submitted.	Y/N	N

- **Source CDF:** 设备如果包含 HDMI OUT 口（General 分页中需要填写支持几个 HDMI OUT 口），需要填写这个分页，其中一般需要填写 HDMI OUT 支持输出哪些分辨率，支持哪些颜色格式等。

Which HDMI output ports are covered by this section?		port1	(type in port names/numbers)
Field Name	Field Definition	Choices	Value
Electrical			
Source_DDC_cap_power_on	Should the DDC capacitance be measured with the product powered on? (Note: HPD will be false during measurement.)	Y/N	N
Video			
Source_HDMI_YCbCr	Will the product transmit an HDMI video signal using YCbCr (4:4:4 or 4:2:2) pixel encoding under some conditions (user selection, EDID indication etc.)?	Y/N	N
Source_AVI_Required	Is the product ever required to transmit an AVI InfoFrame	Y/N	Y
Source_AVI_Supported	Does the product support the transmission of the AVI InfoFrame under some conditions?	Y/N	Y
Source_AVI_Info_Available	Is any of the following information available and valid at the product?: Active Format Aspect Ratio, bar widths, overscan vs. underscan, non-uniform picture scaling, or the colorimetry of the video.	Y/N	Y
Source_Alt_Colorimetry	Will the product ever transmit video using a non-default (i.e. alternate) colorimetry under some condition? (e.g. using BT.709 for 480p or BT.601 for 1080i)	Y/N	N
Source_xvYCC	Is the product capable of transmitting video using xvYCC colorimetry?	Y/N	N
Source_AR_Converter	Does the product have the ability to convert between aspect ratios of 4:3 and 16:9 (and vice versa)?	Y/N	N

- **Sink CDF:** 设备如果包含 HDMI IN 口，需要填写这个分页，其中一般需要填写 HDMI IN 支持输入哪些分辨率，支持哪些颜色格式以及 EDID 相关信息等。

- Repeater CDF：设备如果作为 HDMI Repeater 需要填写这个分页。Rockchip 方案一般不包含这种产品形态。

每个分页一般包含若干个表格需要填写，填写方法与普通 excel 表格一致，以 Source CDF 分页下的 Video 表格为例：

Video			
Source_HDMI_YCbCr	Will the product transmit an HDMI video signal using YCbCr (4:4:4 or 4:2:2) pixel encoding under some conditions (user selection, EDID indication etc.)?	Y/N	<div>N</div>
Source_AVI_Required	Is the product ever required to transmit an AVI InfoFrame	Y/N	<div>Y N</div>
Source_AVI_Supported	Does the product support the transmission of the AVI InfoFrame under some conditions?	Y/N	Y
Source_AVI_Info_Available	Is any of the following information available and valid at the product?: Active Format Aspect Ratio, bar widths, overscan vs. underscan, non-uniform picture scaling, or the colorimetry of the video.	Y/N	Y
Source_Alt_Colorimetry	Will the product ever transmit video using a non-default (i.e. alternate) colorimetry under some condition? (e.g. using BT.709 for 480p or BT.601 for 1080i)	Y/N	N
Source_xvYCC	Is the product capable of transmitting video using xvYCC colorimetry?	Y/N	N
Source_AR_Converter	Does the product have the ability to convert between aspect ratios of 4:3 and 16:9 (and vice versa)?	Y/N	N
Source_60Hz	Does the product output standard, enhanced or high-definition 60Hz video formats on any video output in addition to 640x480p @ 60Hz?	Y/N	Y
Source_50Hz	Does the product output standard, enhanced or high-definition 50Hz video formats on any video output?	Y/N	Y
Source_Above_165	Does the product support any video format/color mode with a TMDS clock frequency above 165MHz?	Y/N	N
Source_Deep_Color	Does the product support any Deep Color modes?	Y/N	N

针对表格中的选项，根据设备的实际情况逐条填写。如上图中的 Source_HDMI_YCbCr 项目，表格第一列为项目名称，第二列为项目说明，第三列为可选值，第四列为申请人需要填写的值。根据描述，该项表明 HDMI 是否支持输出 YCbCr 颜色格式。如果支持该功能，则在右侧下拉列表中选择 Y，反之则选择 N。

由于申请表格通常项目较多，在此就不一一说明，如果对任何项目的填写有疑问，请在 redmine 上提出。