Rockchip eDP 接口开发指南

文件标识：RK-YH-YF-804

发布版本：V1.0.0

日期：2024-04-16

文件密级：□绝密 □秘密 □内部资料 ■公开

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

概述

本文描述了 Rockchip 平台 eDP 接口的使用说明和常用问题的解决办法，本文档将不定期进行更新或修改。

产品版本

芯片名称	内核版本
RK3288	Linux kernel 4.4 及以上内核
RK3399	Linux kernel 4.4 及以上内核
RK356x	Linux kernel 4.19 及以上内核
RK3588	Linux kernel 5.10 及以上内核
RK3576	Linux kernel 6.1 及以上内核

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

版本号	作者	修改日期	修改说明
V1.0.0	吴小平	2024-04-16	初始版本

目录

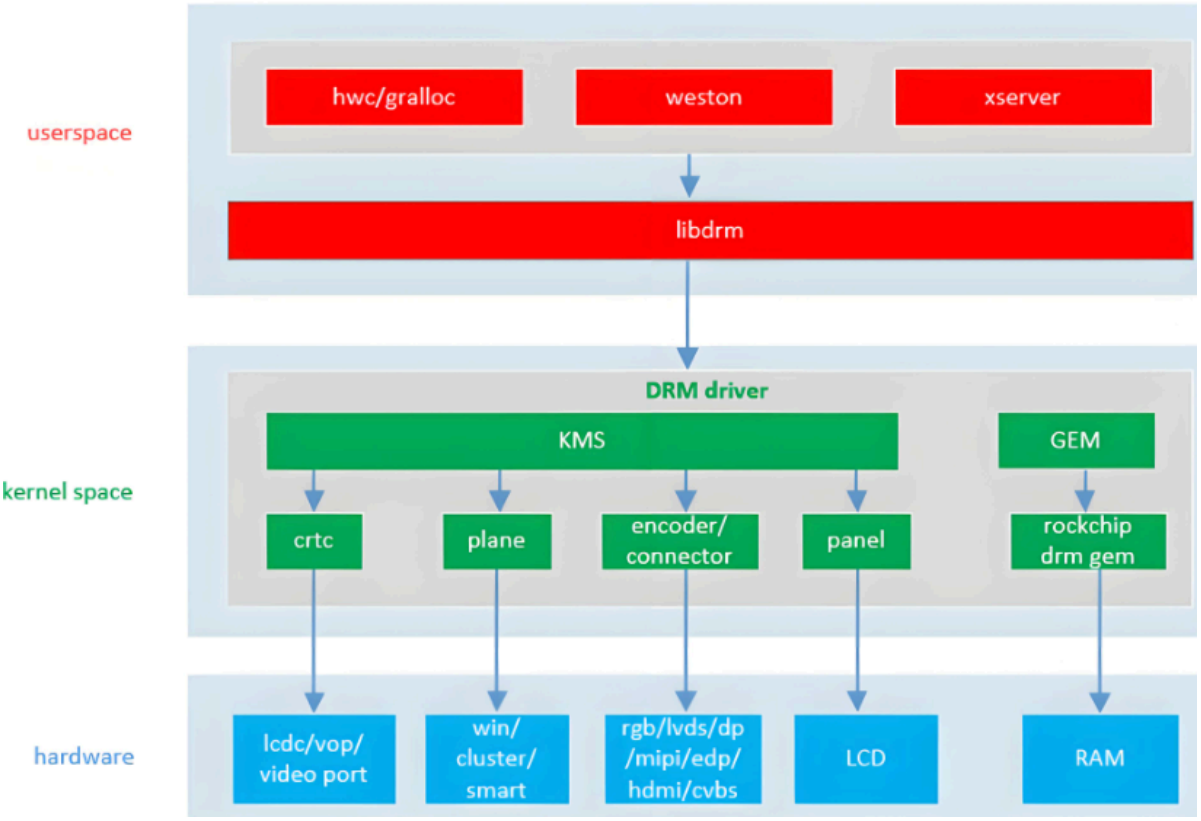
1. 概述

本文主要介绍 Rockchip 平台处理器基于 eDP 显示接口的基本特性、工作流程和常见问题的分析指南。目的是为了相关工程师能对 Rockchip 平台的 eDP 接口显示驱动框架和硬件接口有更好的理解，并通过常见问题的分析能快速定位问题、优化问题和解决问题。

2. eDP 基础介绍

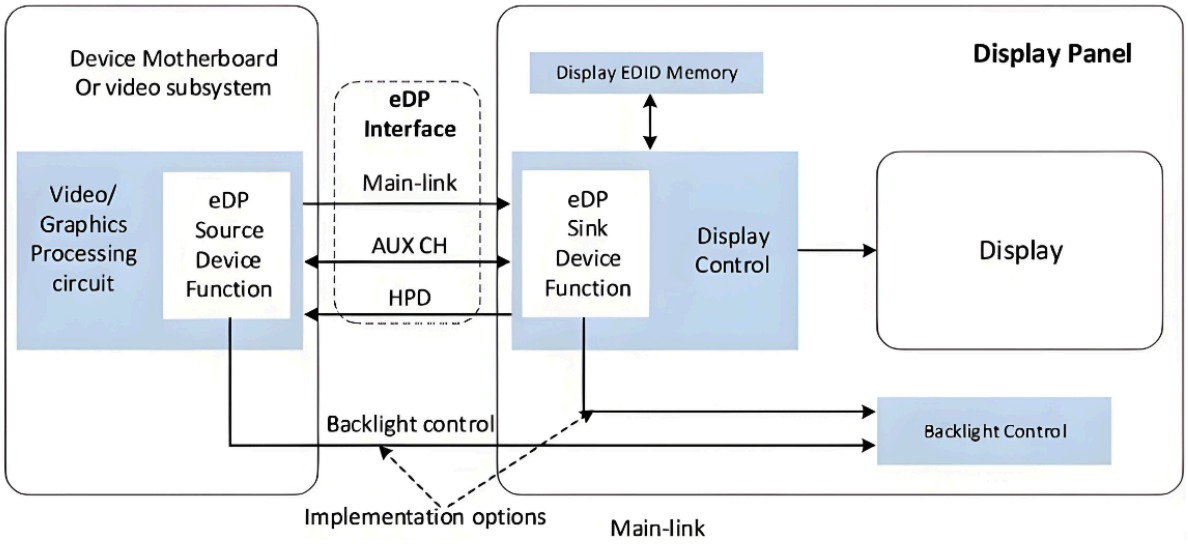
2.1 DRM 显示框架

DRM 驱动和 Libdrm 的显示交互过程图如下：



2.2 eDP 硬件接口交互

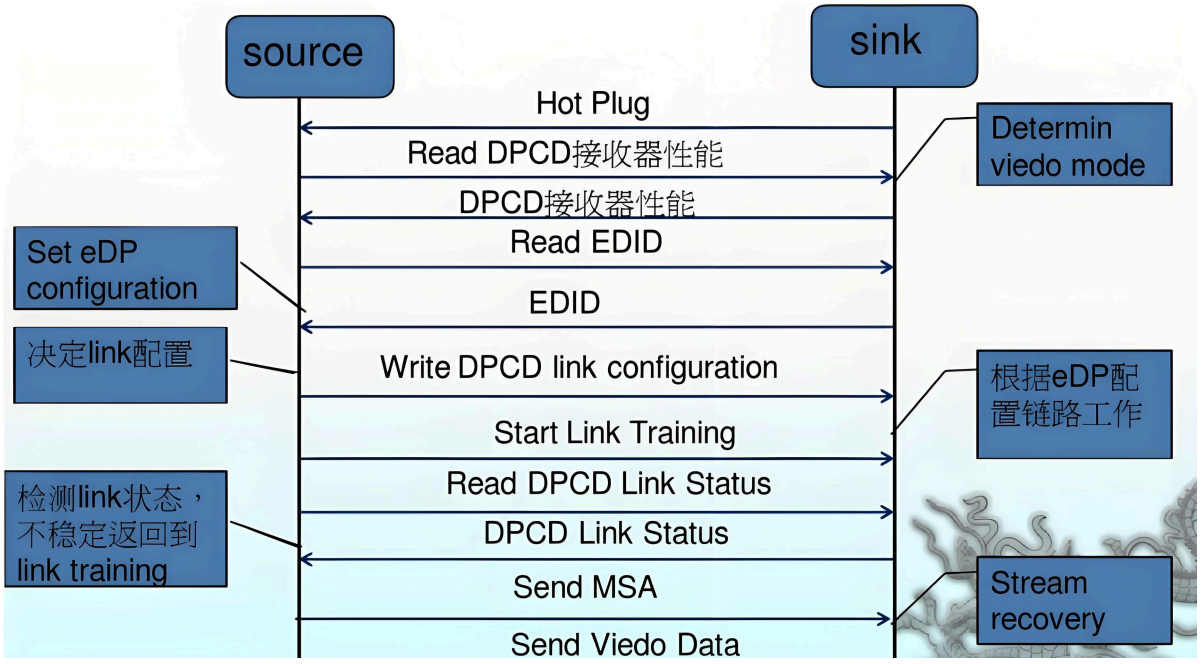
eDP 硬件接口信号主要由 Main-Link、AUX CH 与 HPD 三部分组成。



链路说明:

- Main-Link：表示主通道，用来传输各类型视频数据和音频数据；
- AUX CH：表示辅助通道，用于传输配置参数与指令数据，以及设备控制信号，最大速率1Mbps；
- HPD：表示热插拔检测通道，用于检测 HPD 信号由 Sink 端驱动，通知 Source 端是否有设备接入（选配）。

2.3 eDP 屏幕视频数据流 Linking 过程

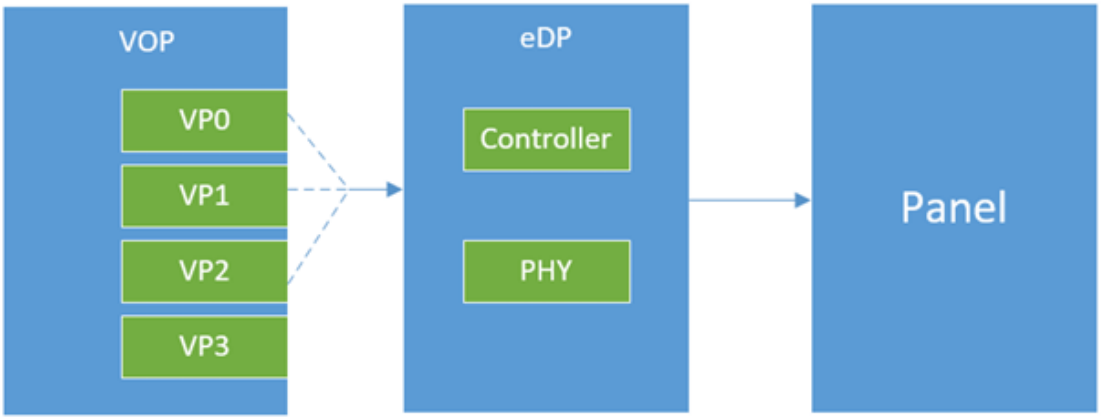


如上图所示 Source 端和 Sink 端的详细通信过程如下：

1. 上电开机，Sink 端通过 HPD 向 Source 端发射一个脉冲信号；
2. Source 端通过读取 Sink 端的 DPCD 接收器，来确定Sink端的接收能力；
3. Sink 端使用 DPCD 接收能力字段进行应答；
4. 确定 Sink 端支持的视频模式，Source 端通过 I2C/AUX 启动 EDID 读取；
5. Sink 端通过 I2C/AUX 应答 EDID；
6. Source 端向 Sink 端写入 DPCD 链路配置；
7. Source 端启动链接培训 Training，Sink 端可进行时钟恢复和均衡化；

8. Source 端读取 DPCD 链路和 Sink 端的状态；
9. Sink 端回应 DPCD 链路和 Sink 端的状态，如果 Main-link 不稳定，返回步骤7；
10. Source 端发送 MSA （主流属性）数据；
11. Source 端发送视频数据流。

2.4 eDP-VOP 显示通路



eDP-VOP 显示通路描述了 eDP、VOP 和 Panel 之间的链路关系。以 RK3588 为例，eDP 最大输出 4K@60Hz，eDP 只能挂载到 VP0/VP1/VP2 上，其中 VP0 最大可以支持到 8K，VP1 和 VP2 最大只能支持到 4K。

3. eDP 驱动配置

3.1 eDP 驱动文件路径

```
#eDP Controller 驱动文件路径(RK3588)
Kernel-5.10:
drivers/gpu/drm/bridge/analogix/analogix_dp_core.c
drivers/gpu/drm/bridge/analogix/analogix_dp_core.h
drivers/gpu/drm/bridge/analogix/analogix_dp_reg.c
drivers/gpu/drm/bridge/analogix/analogix_dp_reg.h
drivers/gpu/drm/rockchip/analogix_dp-rockchip.c
include/drm/bridge/analogix_dp.h

U-Boot:
drivers/video/drm/analogix_dp.c
drivers/video/drm/analogix_dp.h
drivers/video/drm/analogix_dp_reg.c

#eDP PHY 驱动文件路径(RK3588)
Kernel-5.10:
drivers/phy/rockchip/phy-rockchip-samsung-hdptx.c
U-Boot:
drivers/phy/phy-rockchip-samsung-hdptx.c

#eDP Panel 驱动文件路径:
Kernel-5.10:
drivers/gpu/drm/panel/panel-simple.c
u-boot:
drivers/video/drm/rockchip_panel.c
```

Kernel 下参考相关的 dts 配置文件路径如下（以 RK3588 为例）：

```
arch/arm64/boot/dts/rockchip/rk3588s-evb1-lp4x.dtsi
arch/arm64/boot/dts/rockchip/rk3588-evb2-lp4-v10-edp2dp.dts
```

Note：RK3588 eDP 的输入源可以选择 VP0/VP1/VP2，建议优先选择 VP2，因为 VP2 可以任意分配到所需时钟。

3.2 eDP dts 配置

1. 作为外设 Panel，不支持 HPD 参考配置如下（以 RK3588 为例）：

```
&edp0 {
	force-hpd;
	status = "okay";
	ports {
		port@1 {
		reg = <1>;
		edp0_out_panel: endpoint {
			remote-endpoint = <&panel_in_edp0>;
			};
		};
	};
};
```

2. eDP 外接显示器，支持 HPD 参考配置如下（HPD FUNC 功能引脚，以 RK3588 为例）：

- hdmim0_tx0_hpd (gpio1_a5)
- hdmim1_tx0_hpd (gpio3_d4)

```
&edp0 {
	pinctrl-names = "default";
	pinctrl-0 = <&hdmim0_tx0_hpd>;
	status = "okay";
};
&edp0_in_vp0 {
	status = "disabled";
};
&edp0_in_vp1 {
	status = "disabled";
};
&edp0_in_vp2 {
	status = "okay";
};
```

3. HPD GPIO 自定义引脚设置：

```
&edp0 {
	pinctrl-names = "default";
	pinctrl-0 = <&edp0_hpd>;
	hpd-gpios = <&gpio1 RK_PA5 GPIO_ACTIVE_HIGH>;
	status = "okay";
};
&pinctrl {
	edp {
	    edp0_hpd: edp0-hpd {
		 rockchip,pins = <1 RK_PA5 0 &pcfg_pull_none>;
	    };
	};
};
&edp0_in_vp0 {
	status = "disabled";
};
&edp0_in_vp1 {
	status = "disabled";
};
```

```
&edp0_in_vp2 {
        status = "okay";
};
```

3.3 eDP PHY dts 配置

1. 在 RK3588 和 RK3576 上，eDP 和 HDMI 复用 PHY，所以在使用 eDP 的时候，要确保其对应的 HDMI PHY hdptxphy_hdmi 是 disabled 。

```
&hdptxphy0 {
        status = "okay";
};
&hdptxphy_hdmi0 {
        status = "disabled";
};
```

2. eDP Training Table：本属性为可选属性，默认不需配置，使用驱动默认参数即可，如果眼图不符合要求，可以配置以下属性。使用单一参数配置，调整眼图相关参数使眼图达标。

```
&hdptxphy0 {
    /* Single Vdiff Training Table (optional) */
        training-table = /bits/ 8 <
        /* voltage swing 0, pre-emphasis 0->3 */
        0x0d 0x0a 0x04 0x06 0x00 0x04
        0x0d 0x0a 0x04 0x06 0x00 0x04
        0x0d 0x0a 0x04 0x06 0x00 0x04
        0x0d 0x0a 0x04 0x06 0x00 0x04
        /* voltage swing 1, pre-emphasis 0->2 */
        0x0d 0x0a 0x04 0x06 0x00 0x04
        0x0d 0x0a 0x04 0x06 0x00 0x04
        0x0d 0x0a 0x04 0x06 0x00 0x04
        /* voltage swing 2, pre-emphasis 0->1 */
        0x0d 0x0a 0x04 0x06 0x00 0x04
        0x0d 0x0a 0x04 0x06 0x00 0x04
        /* voltage swing 3, pre-emphasis 0 */
        0x0d 0x0a 0x04 0x06 0x00 0x04
        >;
        status = "okay";
};
```

参数说明：

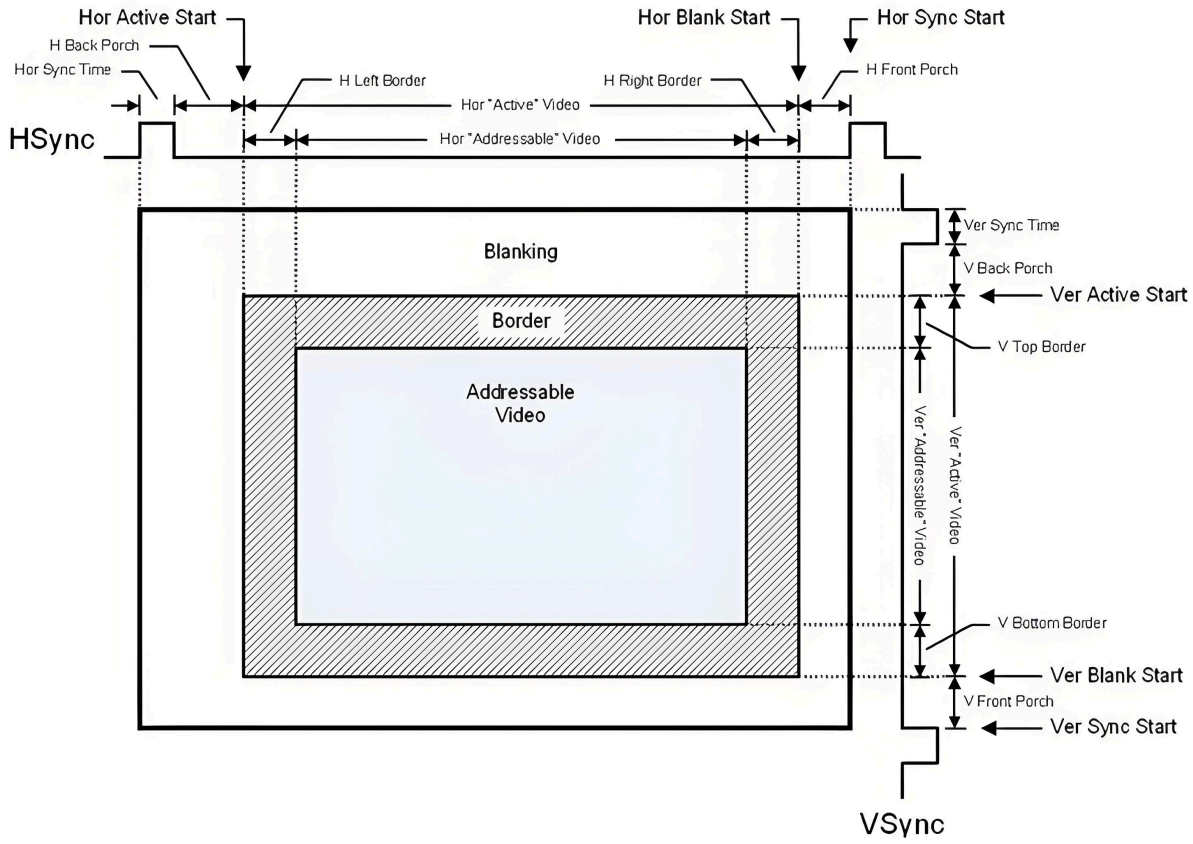
- 第一列参数含义：
  ln_tx_drv_lvl_ctrl
  TX driver main-tap level(TX_AMP)
  01101: max main-tap level(max swing)...
  00000: min main-tap level(min swing)Others: N/A
- 第二列参数含义：
  ln0_tx_drv_post_lvl_ctrl
  TX driver de-emphasis level(TX_DE_EMP)0000: min de-emphasis level
  ...

1110: max de-emphasis level（详细调试方法可以参考眼图章节)
Others: N/A

3. eDP Lane 极性 P/N 交换：如果遇到硬件设计 LANEx 的 P/N 极性接反需要通过软件调整过来，则可以配置修改下面这个属性：lane-polarity-invert （目前只有 RK3588/RK3576 支持此功能)。

```
&hdptxphy0 {
        lane-polarity-invert = <0 1 0 0>;
        status = "okay";
};
```

4. eDP DT-Timing：Rockchip eDP Dt-Timing 各个参数的计算方法。



- htotal = hsync_len + hback_porch + hactive + hfront_porch
- hsync_start = hfront_porch + hactive
- hsync_end = htotal - hback_porch = hsync_len + hactive + hfront_porch
- vtotal = vsync_len + vback_porch + vactive + vfront_porch
- vsync_start = vfront_porch + vactive
- vsync_end = vtotal - vback_porch = vsync_len + vactive + vfront_porch

5. Rockchip eDP Lane Rate 像素带宽计算方法。

- eDP Lane 与传输速率和像素带宽之间的计算关系：带宽 = Link 连接的速率 × Lane 的数量 × 0.8（8/10B)；
  例如：2Lane且传输速率为 2.7GHz 的比特率容量：带宽 = 2 × 2.7Gbps × 0.8 = 4.32Gbps。
- eDP 的比特率传输带宽：比特率传输带宽 = 像素时钟频率 x bpp；
  例如：以 1920x1080@60Hz 计算，像素时钟为 148.5MHz（可以查询 eDPv1.4 协议的Table A-1表格）传输像素单元为24bpp，则其比特率传输带宽 = 148.5MHz × 24bpp= 3.564Gbps。

- eDP 的有效数据带宽：有效数据带宽 = Link连接的速率 x 0.8则有效数据带宽 = 5.4Gbps × 0.8 = 4.32Gbps。

3.4 eDP Panel dts 配置

Rockchip 平台的 eDP Panel 主要有以下三种配置方式，分别是：

1. 使用 compatible 字段匹配固定 Timing 方式：把 Timings 写在drivers/gpu/drm/panel/panel-simple.c中，通过compatible 字段匹配，该方式为 Upstream 推荐的使用方式：

```
panel: panel {
    compatible = "cptt,claa101wb01","lg,lp097qx1-spa1“;
    power-supply = <&vdd_pnl_reg>;
    enable-gpios = <&gpio 90 0>;
    backlight = <&backlight>;
    prepare-delay-ms = <20>;
    reset-delay-ms = <20>;
    enable-delay-ms = <20>;
};
static const struct drm_display_mode lg_lp097qx1_spa1_mode = {
    .clock = 205210,
    .hdisplay = 2048,
    .hsync_start = 2048 + 150,
    .hsync_end = 2048 + 150 + 5,
    .htotal = 2048 + 150 + 5 + 5,
    .vdisplay = 1536,
    .vsync_start = 1536 + 3,
    .vsync_end = 1536 + 3 + 1,
    .vtotal = 1536 + 3 + 1 + 9,
    .vrefresh = 60,
};
static const struct panel_desc lg_lp097qx1_spa1 = {
    .modes = &lg_lp097qx1_spa1_mode,
    .num_modes = 1,
    .size = {
        .width = 320,
        .height = 187,
    },
};
static const struct of_device_id platform_of_match[] = {
    [...]
    }, {
       .compatible = "lg,lp097qx1-spa1",
       .data = &lg_lp097qx1_spa1,
    }, {
    [...]
};
```

2. 读 EDID 方式：不填写任何 Timing，直接使用 EDID 来获取 Timing，不需要填写 Display-Timings 结构参考文档如下：Documentation/devicetree/bindings/display/panel/simple-panel.txt

```
panel-edp0 {
    compatible = "simple-panel";
    power-supply = <&vcc3v3_lcd_edp0>;
    enable-gpios = <&gpio 90 0>;
    backlight = <&backlight>;
    prepare-delay-ms = <120>;
    enable-delay-ms = <120>;
    unprepare-delay-ms = <120>;
    disable-delay-ms = <120>;
    status = "okay";
};
```

3. 固定Timing 方式：是直接将 Timing 写在 DTS 文件中，使用 Display-Timings 结构，这类适用于简单的屏配置（参考屏 Datasheet 上的参数）。

```
panel-edp0 {
    status = "okay";
    compatible = "simple-panel";
    power-supply = <&vcc3v3_lcd_edp0>;
    enable-gpios = <&gpio 90 0>;
    backlight = <&backlight>;
    prepare-delay-ms = <120>;
    enable-delay-ms = <120>;
    unprepare-delay-ms = <120>;
    disable-delay-ms = <120>;
    width-mm = <129>;
    height-mm = <171>;
    bpc = <8>;  //根据屏幕的特性填写8bit 6bit屏幕
    display-timings {
        native-mode = <&timing0>;
        timing0: timing0 {
            clock-frequency = <160000000>;
            hactive = <1200>;
            vactive = <1920>;
            hback-porch = <21>;
            hfront-porch = <120>;
            vback-porch = <18>;
            vfront-porch = <21>;
            hsync-len = <20>;
            vsync-len = <3>;
            hsync-active = <0>;
            vsync-active = <0>;
            de-active = <0>;
            pixelclk-active = <0>;
        };
    };
    ports {
         panel_in_edp0: endpoint {
              remote-endpoint = <&edp0_out_panel>;
         };
    };
};
```

接入 eDP 屏幕开机后，在 U-Boot 的串口 Log 中，有类似下面的打印信息：

```
Maximum visible display size: 26 cm x 17 cm
Power management features: no active off, no suspend, no standby
Estabilished timings:
Standard timings:
2400x1600 59 Hz
Monitor name: LQ123P1JX31
Detailed mode clock 252750 kHz, flags[a]
H: 2400 2448 2480 2560
V: 1600 1603 1613 1646
bus_format: 1009
```

通过上面的信息，我们可以获取到屏参信息如下：

```
dclk        = 252750 kHz
flags       = 0xa
hactive     = 2400
hsync_start = 2448
hsync_end   = 2480
htotal      = 2560
vactive     = 1600
vsync_start = 1603
vsync_end   = 1613
vtotal      = 1646
bus_format  = 1009
```

flags 对应标志位代表信息如下：

```
flags = 0xa = 1010 即 DRM_MODE_FLAG_NVSYNC | DRM_MODE_FLAG_NHSYNC
/* Video mode flags */
/* bit compatible with the xorg definitions. */
#define DRM_MODE_FLAG_PHSYNC            (1 << 0)
#define DRM_MODE_FLAG_NHSYNC            (1 << 1)
#define DRM_MODE_FLAG_PVSYNC            (1 << 2)
#define DRM_MODE_FLAG_NVSYNC            (1 << 3)
#define DRM_MODE_FLAG_INTERLACE         (1 << 4)
#define DRM_MODE_FLAG_DBLSCAN           (1 << 5)
#define DRM_MODE_FLAG_CSYNC             (1 << 6)
#define DRM_MODE_FLAG_PCSYNC            (1 << 7)
#define DRM_MODE_FLAG_NCSYNC            (1 << 8)
#define DRM_MODE_FLAG_HSKEW             (1 << 9)
#define DRM_MODE_FLAG_BCAST             (1 << 10)
#define DRM_MODE_FLAG_PIXMUX            (1 << 11)
#define DRM_MODE_FLAG_DBLCLK            (1 << 12)
#define DRM_MODE_FLAG_CLKDIV2           (1 << 13)
```

Timing 计算和参数说明如下：

- hfront-porch = hsync_start - hactive
- hsync-len = hsync_end - hsync_start
- hback-porch = htotal - hsync_end
- vfront-porch = vsync_start - vactive
- vsync-len = vsync_end - vsync_start

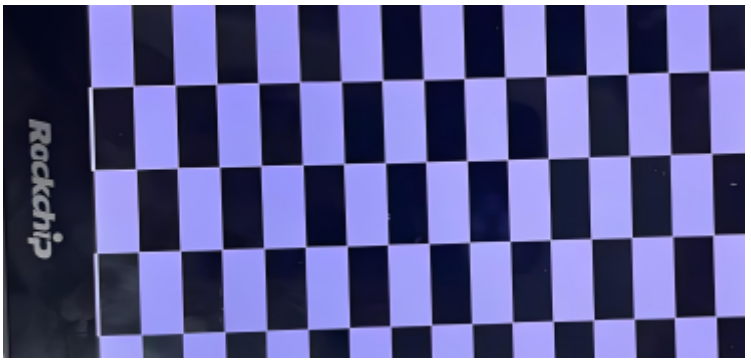
- vback-porch = vtotal - vsync_end
- hsync-active 和 vsync-active 即对应 flags 中的 (BIT(0) - BIT(3))
- hsync-active = <0>：表示 DRM_MODE_FLAG_PHSYNC 反之为 DRM_MODE_FLAG_NHSYNC
- vsync-active = <0>：表示 DRM_MODE_FLAG_PVSYNC 反之为 DRM_MODE_FLAG_NVSYNC
- clock-frequency：即像素时钟 dclk，通常精确到兆即可，尽量写成整数如252750000写252000000。有些特殊屏幕，对时钟参数非常敏感，差距太大会出现闪屏和花屏现象。
- pixelclk-active：表示 像素时钟 dclk 的极性。
  0：表示 active low = drive pixel data on falling edge/sample data on rising edge
  1：表示 active high = drive pixel data on rising edge/sample data on fallin gedge
- prepare-delay-ms：是 eDP AUX 通信最前面的 Delay，在 panel_simple_prepare 函数中会先使能电源在 Delay 对应时间；若不需要则不设置，如果出现点屏失败的情况，确认是时序的问题可以延长 prepare-delay-ms至200-1000。

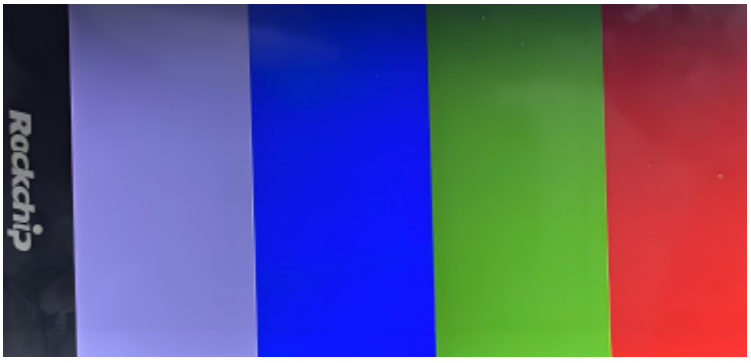
Property	Description	Value
power-supply	Display panels require power to be supplied.	power-supply = <&vcc3v3_lcd_n>;
enable-gpios	Specifier for a GPIO connected to the panel enable control signal.	enable-gpios = <&gpio3 RK_PA7 GPIO_ACTIVE_HIGH>;
reset-gpios	Specifier for a GPIO connected to the panel reset control signal.	reset-gpios = <&gpio3 RK_PA6 GPIO_ACTIVE_HIGH>;
bus-format	Pixel data format	MEDIA_BUS_FMT_RGB888_1X24 ; MEDIA_BUS_FMT_RGB666_1X24_CPADHI ; MEDIA_BUS_FMT_RGB101010_1X30;
width-mm	Physical width in mm	width-mm = <240>;
height-mm	Physical height in mm	height-mm = <120>;
bpc	Bits per color	bpc =<8/6/10>;

4. eDP 工作模式

4.1 eDP 屏幕自测模式

如果使能 Panel 自测模式后，Panel 可以显示，那么说明 Panel 已经正常工作。AUX 可以正常通信，打开 Panel 自测模式，效果图交替显示如下所示：

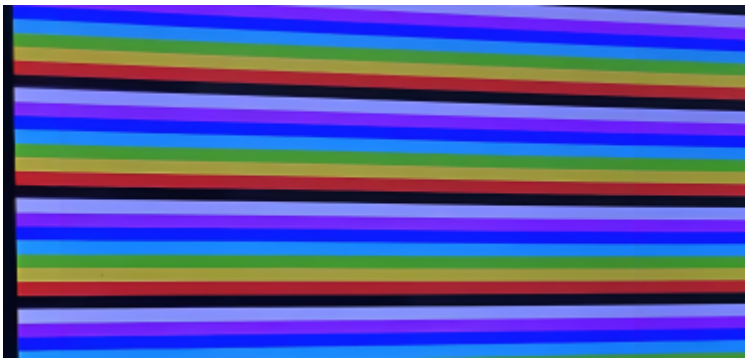





```
diff --git a/arch/arm64/boot/dts/rockchip/rk3588s-evb1-lp4x.dtsi
b/arch/arm64/boot/dts/rockchip/rk3588s-evb1-lp4x.dtsi
index a95bd09749db..4888d2aeadd2 100644
--- a/arch/arm64/boot/dts/rockchip/rk3588s-evb1-lp4x.dtsi
+++ b/arch/arm64/boot/dts/rockchip/rk3588s-evb1-lp4x.dtsi
@@ -223,6 +223,7 @@
};
&edp0 {
+       panel-self-test;
        force-hpd;
        status = "okay";
};
```

4.2 eDP 自测模式

如果使能 eDP 自测模式后，Panel 可以显示那么说明 Panel 已经正常工作。AUX 可以正常通信，eDP 主链路正常，再进行屏端参数的调整，打开 eDP 自测模式，效果图如下所示：



```
diff --git a/arch/arm64/boot/dts/rockchip/rk3588s-evb1-lp4x.dtsi
b/arch/arm64/boot/dts/rockchip/rk3588s-evb1-lp4x.dtsi
index a95bd09749db..4888d2aeadd2 100644
--- a/arch/arm64/boot/dts/rockchip/rk3588s-evb1-lp4x.dtsi
+++ b/arch/arm64/boot/dts/rockchip/rk3588s-evb1-lp4x.dtsi
@@ -223,6 +223,7 @@
};
&edp0 {
+       analogix,video-bist-enable;
        force-hpd;
        status = "okay";
};
```

4.3 eDP 屏幕自刷新 PSR 模式

针对 Source 和 Sink 来说，PSR 属于可选功能，如果要启动 PSR 模式，双方都需要支持。自刷新的最终目的是降低系统功耗，而且是 eDP 嵌入式应用最主要的节省功耗方式，并且效果明显，当系统需要长时间显示静态图片的时候就可以使用此功能。但是市面上支持自刷新的屏幕比较少（以 RK3588 eDP0 为例）。

```
&edp0 {
        support-psr;//配置这个属性即可
        status = "okay";
};
```

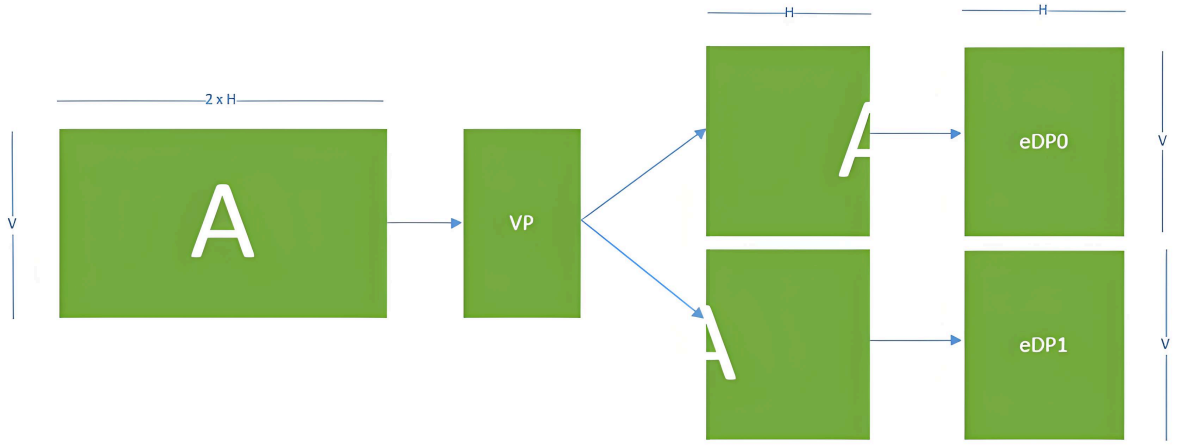
下面是一个 PSR 功能的相关案例，打开 PSR 功能后闪屏，由于在退出 PSR 时候图层没有做恢复，需要等到应用刷新的时候才显示正常图像，这就导致会引入闪屏的现象，如遇到类似问题可以确认如下代码提交是否已合入：

```
commit 16b27eed7c3980eb9ed04a39c050bce7a5e24b15
Author: Sandy Huang <hjc@rock-chips.com>
Date:   Mon Apr 3 16:41:25 2023 +0800

    drm/rockchip: vop2: recover win state when exit psr

    Change-Id: I0d30042473ae84f49b4d326b31732180995b8b52
    Signed-off-by: Sandy Huang <hjc@rock-chips.com>
```

4.4 eDP 的 Split-Mode

Split-Mode 此模式下可以把 VOP 输出分成左右半屏分别给两个 eDP 屏幕显示如下图所示：

Note: Split-Mode 要求 eDP0 和 eDP1 的 Timing 完全一样，必须两个接口/时序完全相同的屏幕。



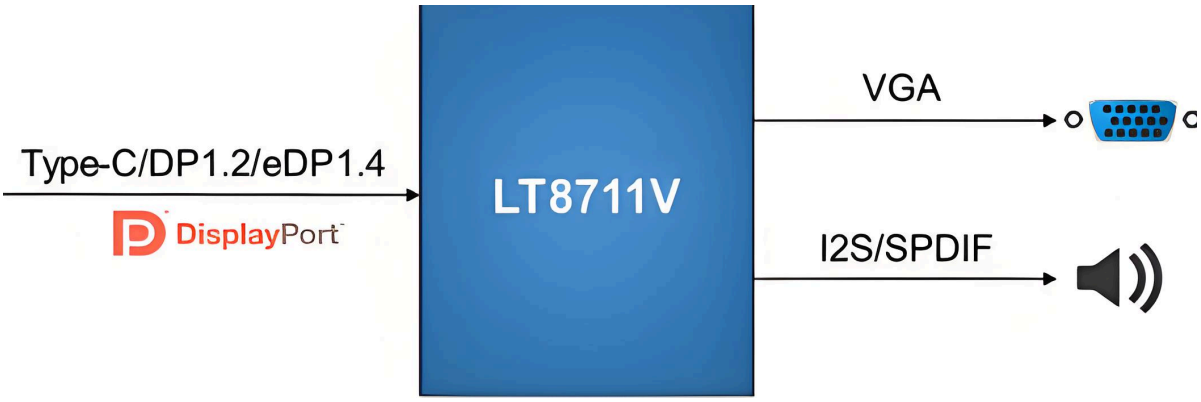
其中 VOP2.0 Split-Mode DTS 代码修改支持如下（目前只有 RK3588/RK3576 支持此功能）：

```
diff --git a/arch/arm64/boot/dts/rockchip/rk3588s-evb1-lp4x.dtsi
b/arch/arm64/boot/dts/rockchip/rk3588s-evb1-lp4x.dtsi
index 8eb4fcf75aae..7b92b00d3759 100644
--- a/arch/arm64/boot/dts/rockchip/rk3588s-evb1-lp4x.dtsi
+++ b/arch/arm64/boot/dts/rockchip/rk3588s-evb1-lp4x.dtsi
@@ -223,6 +223,7 @@
 };
&edp0 {
+       rockchip,split-mode;
        force-hpd;
        status = "okay";
};

&edp1 {
        force-hpd;
        status = "okay";
};

&edp0_in_vp0 {
         status = "okay";
};

&edp1_in_vp0 {
         status = "okay";
};

&edp0_in_vp1 {
         status = "disabled";
};

&edp0_in_vp2 {
         status = "disabled";
};
```

```
&edp1_in_vp1 {
        status = "disabled";
};

&edp1_in_vp2 {
        status = "disabled";
};
```

5. eDP2VGA 相关问题的调试

VGA 接口（Video Graphics Array）视频图形阵列是一个使用模拟信号的电脑显示标准，即电脑采用 VGA 标准输出数据的专用接口。接口共有15针：分成3排，每排5个孔，它是显卡上应用最为广泛的接口类型之一，它能传输红、绿、蓝模拟信号以及同步信号（水平和垂直信号）。



Rockchip 平台有适配过多种 eDP 转 VGA 方案，有 CH751X/IT6516/RTD2166/CS5212/LT8711V 等。VGA 显示器要支持即插即用，即支持热插拔功能，主要的工作流程是上电后转换芯片上报所接入的 VGA 设备的 EDID 信息到主控发起视频传输，要确保转换芯片比主控提前上电 Ready。转换芯片自身的初始化过程需要一定时间和读屏端EDID 信息的时间，下面列出相关案例的解决方法。

5.1 U-Boot Logo 无法显示 Kernel Logo 可以正常显示（以 IT6516 为例）

由于转换芯片 IT6516 默认上报的 EDID 信息中标记为模拟信号，而 Rockchip 平台 U-Boot 代码中读取 EDID 信息时会解析并判断是不是数字信号显示器，导致返回错误信息，最终结果 U-Boot Logo 无法正常显示，系统正常启动之后会再次读取 EDID 信息，Kernel 下没有这部分判断逻辑所以可以正常显示，最终表现为系统完全启动后可以正常显示，解决的办法可以在驱动代码中忽略此判断或者由转换芯片修改默认的上报信息中标记为数字信号代码如下：

```
xx@ubuntu:~$vi uboot/common/edid.c
int edid_get_drm_mode(u8 *buf, int buf_size, struct drm_display_mode *mode,
-if (!EDID1_INFO_VIDEO_INPUT_DIGITAL(*edid)) {
+/*if (!EDID1_INFO_VIDEO_INPUT_DIGITAL(*edid)) {
        debug("%s: Not a digital display\n", __func__);
        //判断edid 里面的info信息:读出来video_input_definition=0xe最高位为0判断为不是数字信号
        return -ENOSYS;
-   }
+   }*/
Note: EDID1_INFO_VIDEO_INPUT_DIGITAL-> GET_BIT(((_x).video_input_definition), 7)
edid info 14H寄存器的第7bit模拟/数字信号: 模拟= 0,数字= 1;
//与对应的转换芯片厂商协调这个上报信息，确保正确读取 edid 获得分辨率等信息正常显示,默认这个转换芯片上报的是 VGA 转换芯片固定报过来的模拟 edid,要求后端接的是模拟显示器,默认当 monitor 使用的模拟显示器,所以报的是模拟edid,才会出现此问题。
```

5.2 U-Boot Logo 能显示 Kernel Logo 无法正常显示（以 IT6516 为例）

U-Boot 传递的 Mode 参数和 Kernel 阶段不匹配导致，通过下面的调试方法找到不匹配的原因。

```
xx@ubuntu:~$vi kernel/drivers/gpu/drm/rockchip/rockchip_drm_logo.c
       list_for_each_entry(mode, &connector->modes, head) {
+      if(connector->connector_type == DRM_MODE_CONNECTOR_DisplayPort)
+          drm_mode_convert_to_origin_mode(mode);
+          DRM_INFO("wxp add uboot convert mode: clock[%d], h/v display[%d,%d] h/v +sync_end[%d,%d] flags[0x%x], aspect_ratio[%d]\n",mode->clock,mode->hdisplay,
+                   mode->vdisplay, mode->crtc_hsync_end, mode->crtc_vsync_end,
+                   set->flags, mode->picture_aspect_ratio);
+
                    if (mode->clock == set->clock &&
                    mode->hdisplay == set->hdisplay &&
                    mode->vdisplay == set->vdisplay &&
-                   mode->crtc_hsync_end == set->crtc_hsync_end &&
-                   mode->crtc_vsync_end == set->crtc_vsync_end &&
-                   drm_mode_vrefresh(mode) == set->vrefresh &&
-                   /* we just need to focus on DRM_MODE_FLAG_ALL flag, so here
+                   mode->hsync_end == set->crtc_hsync_end &&
+                   mode->vsync_end == set->crtc_vsync_end &&
+                   drm_mode_vrefresh(mode) == set->vrefresh)
+                   {/* we just need to focus on DRM_MODE_FLAG_ALL flag, so here
                     * we compare mode->flags with set->flags & DRM_MODE_FLAG_ALL.
                     */
-                   mode->flags == (set->flags & DRM_MODE_FLAG_ALL) &&
-                   mode->picture_aspect_ratio == set->picture_aspect_ratio) {
+                   //mode->flags == (set->flags & DRM_MODE_FLAG_ALL) &&
+                   //mode->picture_aspect_ratio == set->picture_aspect_ratio) {
                    -->该问题主要是这里不匹配导致
                       found = 1;
                       match = 1;
+                      DRM_INFO("---> This can't match mode ---->\r\n");
```

```
                        //这里找到对应的mode才可以正常显示kernel logo
                        break;
                }
+                if(connector->connector_type == DRM_MODE_CONNECTOR_DisplayPort)
+                        drm_mode_convert_to_split_mode(mode);
        }
```

5.3 U-Boot Logo 能显示 Kernel Logo 正常显示之后无法显示桌面

如果 U-Boot 不能正常读到 EDID 信息，那么会使用系统默认分辨率输出（如：1280x720），结果出现无法显示桌面的情况。因为在 Kernel 起来之后会再次读显示的 Prefer 分辨率（如：1920x1080），由于U-Boot 下输出的分辨率和 Kernel 下输出的分辨率不一致，系统启动之后由 Weston 自动切换分辨率切换失败导致无法正常显示桌面，关键性 Log 信息如下（以 IT6516 为例）：

```
Update mode to 1920x1080p60, type: 14(if:0, flag:0x0) for vp2 dclk: 148500000
```

从log中可以发现 type: 14的(if:0, flag:0x0)里面if的值清零了，正常的值是100。

```
xx@ubuntu:~$ vi kernel\drivers\gpu\drm\rockchip\analogix_dp-rockchip.c
rockchip_dp_drm_encoder_disable
        s->output_if &= ~(dp->id ? VOP_OUTPUT_IF_eDP1:VOP_OUTPUT_IF_eDP0);//这里清除了
```

完整异常log信息如下：

```
[   32.715314] rockchip-vop2 fdd90000.vop: [drm:vop2_crtc_atomic_enable] Update mode to 1920x1080p60, type: 14(if:0,flag:0x0) for vp2 dclk: 148500000
1024x768@70.1, 75.0 MHz
[   32.715963] rockchip-vop2 fdd90000.vop: [drm:vop2_crtc_atomic_enable] #wxp clk_name=dclk_out2 800x600@72.2, 50.0 MHz
[   32.715995] rockchip-vop2 fdd90000.vop: [drm:vop2_crtc_atomic_enable] dclk_out2 div: 0 dclk_core2 div: 2 800x600@60.3, 40.0 MHz
[   32.716009] rockchip-vop2 fdd90000.vop: [drm:vop2_crtc_atomic_enable] #wxp clk_name=dclk2 640x480@75.0, 31.5 MHz
[   32.716022] rockchip-vop2 fdd90000.vop: [drm:vop2_crtc_atomic_enable] #wxp vop2_clk dclk->rate = 148500000 640x480@72.8, 31.5 MHz
[   32.716032] rockchip-vop2 fdd90000.vop: [drm:vop2_crtc_atomic_enable] #wxp set dclk->rate = 0 640x480@66.7, 30.2 MHz
[   32.716182] rockchip-vop2 fdd90000.vop: [drm:vop2_crtc_atomic_enable] set dclk_vop2 to 0 ,get 148500000
```

如遇到上面的问题，可以确认如下代码提交是否已合入：

```
commit e07cb9adcc9e6f087a8f4672a7ca76e01eb56cd6 (HEAD)
Author: Zhang Yubing <yubing.zhang@rock-chips.com>
Date:   Sat May 6 10:11:51 2023 +0800

    drm/bridge: analogix_dp: clear old output_if when crtc active change

    Signed-off-by: Zhang Yubing <yubing.zhang@rock-chips.com>
    Change-Id: Ibb2fe312ec1f9e72471eefe25de37883f114564c
```

5.4 VGA 的热插拔

以 Linux SDK 为例，若有遇到热插拔 VGA 显示器不显示的情况，通过抓 VOP Buffer 来确认 Weston 传递的图是黑色，这定位到是 Weston 异常，请通过提交 Case 获取最新版本 Weston。比如更新 Weston-12.02 版本解决黑屏无显示问题：

- 下载链接：从新提的客户 Readmine 上索要最新版本的 Weston；
- 添加最新版本：weston-12.02.tar 目录 /buildroot/dl/weston；
- 替换目录：buildroot/package/weston。

5.5 VGA 的 HPD 信号

如果此转换芯片没有 HPD 信号或者由于硬件设计原因没有将 HPD 信号引到主控上，热插拔时无法上报 HPD 事件，导致无法正常显示的异常，可以参考如下修改。此修改是软件定时进行 POLL Detect Connector 取代 HPD 功能，针对只有固定分辨率的应用场景（以 CS5202 为例）。

```
diff --git a/drivers/gpu/drm/drm_probe_helper.c
b/drivers/gpu/drm/drm_probe_helper.c
index 225a84f97b50..da571e358485 100755
--- a/drivers/gpu/drm/drm_probe_helper.c
+++ b/drivers/gpu/drm/drm_probe_helper.c
@@ -223,7 +223,7 @@ drm_connector_mode_valid(struct drm_connector *connector,
        return ret;
 }
-#define DRM_OUTPUT_POLL_PERIOD (10*HZ)
+#define DRM_OUTPUT_POLL_PERIOD (1*HZ) //修改POLL的时间
 /**
  * drm_kms_helper_poll_enable - re-enable output polling.
  * @dev: drm_device
drivers/gpu/drm/bridge/analogix/analogix_dp_core.c
diff --git a/drivers/gpu/drm/bridge/analogix/analogix_dp_core.c
b/drivers/gpu/drm/bridge/analogix/analogix_dp_core.c
index b1467e2d25a8..45b5a71bc84e 100755
--- a/drivers/gpu/drm/bridge/analogix/analogix_dp_core.c
+++ b/drivers/gpu/drm/bridge/analogix/analogix_dp_core.c
@@ -1534,7 +1534,8 @@ static int analogix_dp_bridge_attach(struct drm_bridge
*bridge,
                DRM_ERROR("Failed to initialize connector with drm\n");
                return ret;
        }
```

```
-
+          printk("wxp set polled connect:disconnect\r\n");
+          connector->polled = DRM_CONNECTOR_POLL_CONNECT |
DRM_CONNECTOR_POLL_DISCONNECT;
           private = connector->dev->dev_private;
           if (dp->split_area)
```

5.6 VGA 的兼容性

调试中若遇到 VGA 显示器：部分显示器能正常显示，部分不能正常显示（以 RK3588+型号 CH7517 为例），分析如下：

- CH7517 读到 EDID 信息中的 hsync 极性为 + 时，即显示器实际 hsync 为 + 则 SOC 输出到 CH7517 数据异常无法正常显示；
- CH7517 读到 EDID 信息中的 hsync 极性为 - 时，即显示器实际 hsync 为 - 则 SOC 输出到 CH7517 正常显示。

由于 RK3588 只能 Hync & Vsync 支持输出单一极性，参考代码提交如下：

```
commit e9cac7f1fea932c845bd5b4608b1b003d6b4b930
Author: Wyon Bi <bivvy.bi@rock-chips.com>
Date:   Thu Jun 16 11:34:07 2022 +0000

    video/drm: analogix_dp: Use video format information from register

    Force sync polarity to active low for RK3588.

    Signed-off-by: Wyon Bi <bivvy.bi@rock-chips.com>
    Change-Id: I73270addc6118279accf6d9ba01f3f018a4e0850
```

6. eDP 支持 v1.4b 协议

对于 eDPv1.4b 版本，由于 Link Rate 的读取和设置不再向下兼容，若读取 MAX LINK RATE 为00可以尝试使用此补丁，这用于解决 eDP v1.4 版本读取和设置 Link Rate 的问题，eDP v1.4 中读取 MAX_LINK_RATE 会读出00表明此 Panel 仅支持查询 SUPPORTED_LINK_RATES 且需设置 Link Rate 于 DP_LINK_RATE_SET 寄存器，代码提交记录如下：

```
commit 883306549c0c048ed259c464744eebe7bc1f4851 (HEAD)
Author: Damon Ding <damon.ding@rock-chips.com>
Date:   Tue May 21 15:13:45 2024 +0800

    phy/rockchip: samsung-hdptx: add support for rate R216/R243/R324/R432

    R216/R243/R324/R432 are the new recommended link rates in
    eDP v1.4, which may be read from DPCD SUPPORTED_LINK_RATES
    (0x00010h through 0x0001fh). And set the link rate by DPCD
    LINK_BW_SET(0x00100h).

    Signed-off-by: Damon Ding <damon.ding@rock-chips.com>
    Change-Id: Ib692d18cd79765eb172fb0455cbc1bc1f0594d79
---------------------------------------------------------------------------
commit 96d96c31ea5c219f483efc62e5be7a6d81192a41 (HEAD)
Author: Damon Ding <damon.ding@rock-chips.com>
Date:   Tue May 21 15:12:16 2024 +0800

    drm/bridge: analogix_dp: add support for rate R216/R243/R324/R432

    R216/R243/R324/R432 are the new recommended link rates in
    eDP v1.4, which may be read from DPCD SUPPORTED_LINK_RATES
    (0x00010h through 0x0001fh). And set the link rate by DPCD
    LINK_BW_SET(0x00100h).

    Signed-off-by: Damon Ding <damon.ding@rock-chips.com>
    Change-Id: I2d46c910bc763f1dac282a2156f9daec88d60cac
---------------------------------------------------------------------------
commit 936d3dfad8999d82f36bf6837a2ffa318b84ef90 (HEAD)
Author: Damon Ding <damon.ding@rock-chips.com>
Date:   Fri May 17 17:52:03 2024 +0800

    drm/bridge: analogix_dp: add support to parse link rate for eDP v1.4

    As Table 4-23 in eDP v1.4 spec, the link rate can be
    read from SUPPORTED_LINK_RATES(0x00010h through 0x0001fh),
    which is required for sink devices.

    Signed-off-by: Damon Ding <damon.ding@rock-chips.com>
    Change-Id: Ibff01e7eee16be735b5f4b10c6ef51e9b2954274
---------------------------------------------------------------------------
commit 44519c1186f44cecbfeefd1617a90dc393c0f5b5 (HEAD)
Author: Damon Ding <damon.ding@rock-chips.com>
Date:   Mon May 27 17:48:55 2024 +0800
```

```
    drm/bridge: analogix_dp: remove getting bandwidth/lane_count in
analogix_dp_full_link_train()

    The bandwidth/lane_count have been got in analogix_dp_detect(),
    which is earlier than analogix_dp_full_link_train().
    And getting link configs in .detect() will help to
    check mode support in .mode_valid().

    Signed-off-by: Damon Ding <damon.ding@rock-chips.com>
    Change-Id: Ia98bae1d3be20243f1a81b5beec0fb53f2ea3ca1
```

7. eDP 点屏常见问题分析方法

7.1 AUX CH 报错 Log

```
[ 11.591040] rockchip-dp fe0c0000.edp: AUX CH command reply failed!
[ 11.591123] rockchip-dp fe0c0000.edp: AUX CH command reply failed!
[ 11.591242] rockchip-dp fe0c0000.edp: AUX CH error happens: 0x2 (1)
[ 11.591242] rockchip-dp fe0c0000.edp: AUX CH error happened: 0x2 (1)
[ 11.592663] rockchip-dp fe0c0000.edp: AUX CH error happened: 0x2 (1)
[ 11.593026] rockchip-dp fe0c0000.edp: AUX CH error happened: 0x2 (1)
[ 11.596047] rockchip-dp fe0c0000.edp: failed to get Rx Max Link Rate
[ 11.597051] rockchip-dp fe0c0000.edp: failed to init training
[ 11.598089] rockchip-dp fe0c0000.edp: unable to do link train
```

7.1.1 检查 Power Sequence 是否符合协议要求

遇到上面的 AUX Command 相关的报错信息，可以检查屏端的 Power Sequence。

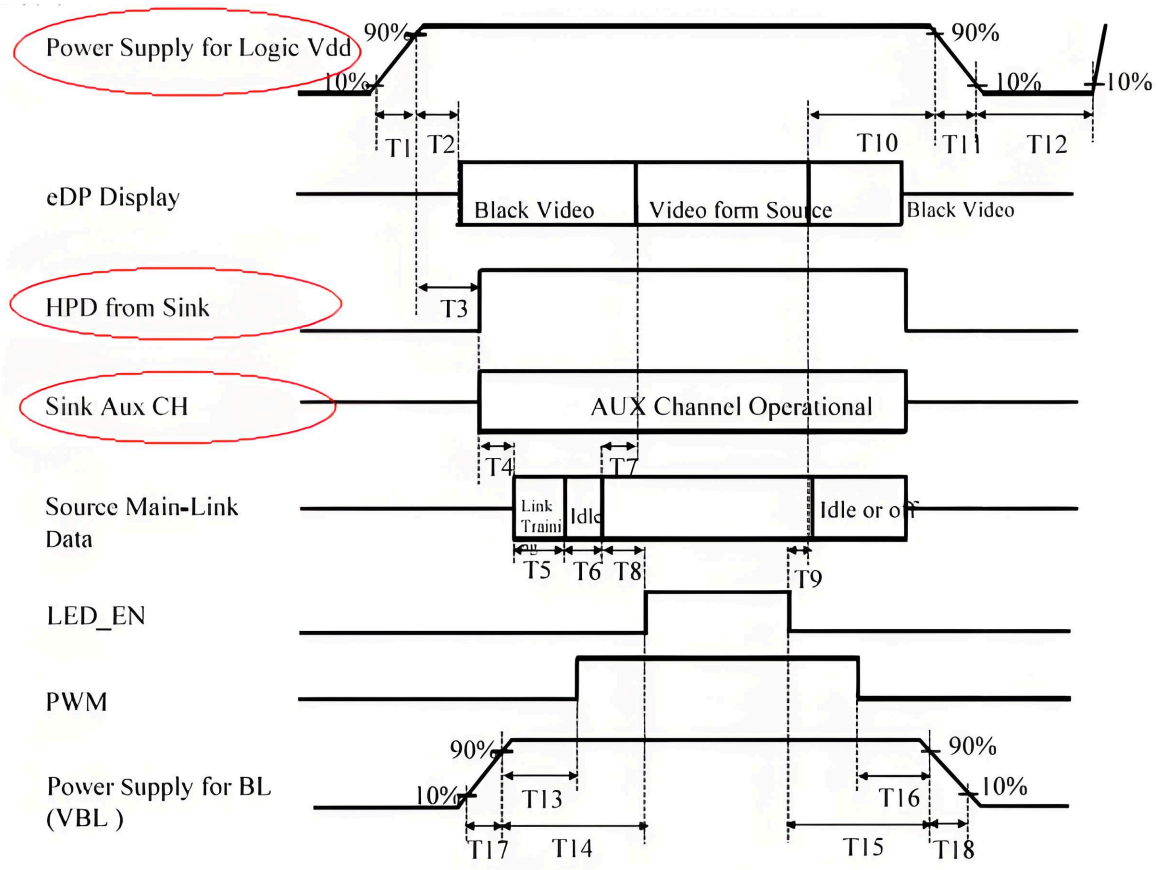


Figure 15. Power Sequence

7.1.2 检查 HPD 信号是否正常

确认是否填写了 force-hpd 导致上电时序不对，AUX 过早进行通信。确认填写的 hpd-gpios 检查 HPD Pin 是否正确，用示波器检查上电时序是否符合要求。

```
xx@ubuntu:~$ cat /sys/class/drm/card0-eDP-1/status
connected
```

Note：如果 status 为 disconnected，请检查 HPD 信号。

7.1.3 检查 Prepare Delay 延时

对于部分屏幕 AUX 报错可以通过延长上电延时来规避，这种情况是上电时序不对，通过 Panel 结点的 prepare-delay-ms 来优化，尝试用如200/500/1000等值，参考修改 eDP Panel 这个属性。

```
edp-panel {
+    prepare-delay-ms = <500>;  //200-500-1000根据屏幕需求做对应的调整
};
```

7.2 AUX Link Rate 报错 Log

```
xx@ubuntu:~$ dmesg | grep edp
[ 3.236549] rockchip-dp fdec0000.edp: failed to read max link rate
[ 3.260422] rockchip-dp fdec0000.edp: failed to read max link rate
[ 3.284163] rockchip-dp fdec0000.edp: failed to read max link rate
```

7.2.1 确认主控端硬件接口的走线 PCB（先检查硬件）

确认主控端 eDP 接口的走线是否符合规范，可以参考我们官方的硬件设计指南，每个平台都有对应的文档。检查接口是否符合如下要求：

- 屏端的上电时序；
- 屏线长度不要超过30cm；
- 屏线应有屏蔽；
- 确认屏线接口没有接错；
- 确认线材本身质量。

7.2.2 确认降低速率是否可以消除

U-Boot 修改这个文件：vi u-boot/drivers/video/rockchip_analogix_dp.c

```
--- a/drivers/video/rockchip_analogix_dp.c
+++ b/drivers/video/rockchip_analogix_dp.c
@@ -506,7 +506,8 @@ static void analogix_dp_init_training(struct
analogix_dp_device *dp,
        /* Initialize by reading RX's DPCD */
        analogix_dp_get_max_rx_bandwidth(dp, &dp->link_train.link_rate);
        analogix_dp_get_max_rx_lane_count(dp, &dp->link_train.lane_count);
-
+       dp->link_train.link_rate = DP_LINK_BW_1_62;
        if ((dp->link_train.link_rate != DP_LINK_BW_1_62) &&
            (dp->link_train.link_rate != DP_LINK_BW_2_7) &&
            (dp->link_train.link_rate != DP_LINK_BW_5_4)) {
```

Kernel-5.10 修改这个文件：kernel\drivers\gpu\drm\bridge\analogix\analogix_dp_core.c

```
static void analogix_dp_init_training(struct analogix_dp_device *dp,
                      enum link_lane_count_type max_lane,
                      int max_rate)
{
    /*
     * MACRO_RST must be applied after the PLL_LOCK to avoid
     * the DP inter pair skew issue for at least 10 us
     */
    analogix_dp_reset_macro(dp);

    /* Initialize by reading RX's DPCD */
    analogix_dp_get_max_rx_bandwidth(dp, &dp->link_train.link_rate);
    analogix_dp_get_max_rx_lane_count(dp, &dp->link_train.lane_count);

    if ((dp->link_train.link_rate != DP_LINK_BW_1_62) &&
        (dp->link_train.link_rate != DP_LINK_BW_2_7) &&
        (dp->link_train.link_rate != DP_LINK_BW_5_4)) {
        dev_err(dp->dev, "Rx Max Link Rate is abnormal :%x !\n",
            dp->link_train.link_rate);
        dp->link_train.link_rate = DP_LINK_BW_1_62;
    }
+   dp->link_train.link_rate = DP_LINK_BW_1_62;
    if (dp->link_train.lane_count == 0) {
        dev_err(dp->dev, "Rx Max Lane count is abnormal :%x !\n",
            dp->link_train.lane_count);
        dp->link_train.lane_count = (u8)LANE_COUNT1;
    }
```

7.3 Link Training Failed

针对 Link Training Failed 的问题，报错通常如下：

```
# Link Training CR 阶段失败
CR Max reached (5,1,0) LT CR failed!
CR Max reached (5,1,0) LT CR failed!
CR Max reached (5,1,0) LT CR failed!
CR Max reached (5,1,0) LT CR failed!
CR Max reached (5,1,0) LT CR failed!
unable to do link train
# Link Training EQ 阶段失败
EQ Max loop LT EQ failed
```

这是 eDP/DP Link Training 不通过，除了检查硬件之外，还可以通过如下方法进一步确认和分析。

7.3.1 增加 Retry 机制

增加 Retry 的次数和增加延时，具体代码修改如下：

```
diff --git a/drivers/video/drm/analogix_dp.h b/drivers/video/drm/analogix_dp.h
index 26815c79f5..740f4beec4 100644
--- a/drivers/video/drm/analogix_dp.h
++ b/drivers/video/drm/analogix_dp.h
`@ -379,7 +379,7 `@
\#define VIDEO_MODE_SLAVE_MODE (0x1 << 0)
\#define VIDEO_MODE_MASTER_MODE (0x0 << 0)
-#define DP_TIMEOUT_LOOP_COUNT 100
+#define DP_TIMEOUT_LOOP_COUNT 200
-#define MAX_CR_LOOP 5
+#define MAX_CR_LOOP 10
-#define MAX_EQ_LOOP 5
+#define MAX_EQ_LOOP 10
```

7.3.2 增加 Retrain 机制

U-Boot 代码修改如下:

```
diff --git a/drivers/video/drm/analogix_dp.c b/drivers/video/drm/analogix_dp.c
index 70cd620fa8..a5b7c5ac1a 100644
--- a/drivers/video/drm/analogix_dp.c
+++ b/drivers/video/drm/analogix_dp.c
@@ -458,6 +458,13 @@ static int analogix_dp_init_training(struct
analogix_dp_device *dp,*
/* All DP analog module power up */
   analogix_dp_set_analog_power_down(dp, POWER_ALL, 0);
+  if (dp->connector.panel->funcs->unprepare)
+       dp->connector.panel->funcs->unprepare(dp->connector.panel);
+  if (dp->connector.panel->funcs->prepare)
+       dp->connector.panel->funcs->prepare(dp->connector.panel);
   return 0;
}
```

Kernel-5.10 代码修改:

```
--- a/drivers/gpu/drm/bridge/analogix/analogix_dp_core.c
+++ b/drivers/gpu/drm/bridge/analogix/analogix_dp_core.c
static int analogix_dp_full_link_train(struct analogix_dp_device *dp,
                       u32 max_lanes, u32 max_rate)
       /* All DP analog module power up */
       analogix_dp_set_analog_power_down(dp, POWER_ALL, 0);
+      if (dp->plat_data->panel) {
+            analogix_dp_panel_unprepare(dp);
+            analogix_dp_panel_prepare(dp);
+      }
       dp->link_train.lt_state = START;
}
```

7.3.3 确认 eDP 屏幕的协议版本

现在阶段部分屏幕只支持 eDPv1.4 版本的协议，且不支持向下兼容，这需要同屏厂确认此屏幕所用的芯片规格书，查看支持的 eDP 最高版本是否符合要求。如果需要支持 eDPv1.4 协议，可以通过如下修改确认问题点（Rockchip 平台已支持 eDPv1.4）。

```
static int analogix_dp_init_training(struct analogix_dp_device *dp,
                                     enum link_lane_count_type max_lane,
                                     int max_rate)
{
        u8 dpcd;
        /*
         * MACRO_RST must be applied after the PLL_LOCK to avoid
         * the DP inter pair skew issue for at least 10 us
         */
        analogix_dp_reset_macro(dp);

        /* Initialize by reading RX's DPCD */
        analogix_dp_get_max_rx_bandwidth(dp, &dp->link_train.link_rate);
        analogix_dp_get_max_rx_lane_count(dp, &dp->link_train.lane_count);

        if ((dp->link_train.link_rate != DP_LINK_BW_1_62) &&
            (dp->link_train.link_rate != DP_LINK_BW_2_7) &&
            (dp->link_train.link_rate != DP_LINK_BW_5_4)) {
+       dev_err(dp->dev, "failed to get Rx Max Link Rate, \
                dp->link_train.link_rate=0x%x\n", dp->link_train.link_rate);
                return -ENODEV;
        }
        if (dp->link_train.lane_count == 0) {
                dev_err(dp->dev, "failed to get Rx Max Lane Count\n");
                return -ENODEV;
        }
```

从上面的打印信息中能确认是否为 0x1e：define DP_LINK_BW_8_1 0x1e /* 1.4 */，表示的是 eDPv1.4 版本。

7.3.4 强制默认输出

如果依然不能正常显示或者是概率性异常输出，可以尝试强制输出来确认问题点；正式版本的代码中不推荐使用强制输出，这可能仍会有部分屏幕无法点亮，U-Boot 代码修改如下：

```
diff --git a/drivers/video/drm/analogix_dp.c b/drivers/video/drm/analogix_dp.c
 index 524cc6bd48..dc601e1fa1 100644
 --- a/drivers/video/drm/analogix_dp.c
 +++ b/drivers/video/drm/analogix_dp.c
 @@ -907,8 +908,12 @@ static int analogix_dp_connector_enable(struct
display_state *state)
 ret = analogix_dp_set_link_train(dp, dp->video_info.max_lane_count,
                                        dp->video_info.max_link_rate);
       if (ret) {
-            dev_err(dp->dev, "unable to do link train\n");
-            return ret;
+            dev_err(dp->dev, "unable to do link train ,using default
strength\n");
+            //return ret;
+           dp->link_train.training_lane[0] =
+                     DP_TRAIN_VOLTAGE_SWING_LEVEL_0 |
+                     DP_TRAIN_PRE_EMPH_LEVEL_0;
+                     analogix_dp_set_lane_link_training(dp);
         }
         analogix_dp_enable_scramble(dp, 1);
```

7.4 eDP 黑屏和花屏闪屏问题

7.4.1 屏幕本身支持的颜色数据位数不匹配导致的花屏

```
panel-edp {
    status = "okay";
    compatible = "innolux,p120zdg-bf4", "simple-panel";
    backlight = <&backlight>;
    power-supply = <&vcc3v3_lcd_edp>;
    prepare-delay-ms = <120>;
    enable-delay-ms = <120>;
    unprepare-delay-ms = <120>;
    disable-delay-ms = <120>;
    width-mm = <129>;
    height-mm = <171>;
+   bpc = <6>;  //6bit 8bit 10bit屏幕本身支持的对应这里修改
    bus-format = <MEDIA_BUS_FMT_RGB888_1X24>;
```

7.4.2 开机黑屏

针对 Android 版本，当遇到开机过程中 U-Boot Logo 和 Kernel Logo 正常显示，到 Android Logo 直接黑屏的问题。需要确认驱动层的原因还是 HWC 层的问题，通过如下步骤进行查看问题：

1. HWC 相关问题确认（参考 RKDocs\common\display\Rockchip FAQ DRM Hardware Composer V1.00-20181213.pdf）通过查看当前开机或者休眠唤醒异常时的串口信息是否有 AUX Failed 等重要错误信息，来用于判断开机或休眠唤醒时，eDP 和屏端是否 AUX 通信正常并使能成功，通信异常会导致黑屏或者不显示问题。

```
xx@ubuntu:~$ adb shell getprop | grep ghwc    //查看版本信息
xx@ubuntu:~$ setprop sys.hwc.compose_policy 6 //HWC enable
xx@ubuntu:~$ setprop sys.hwc.compose_policy 0 //HWC disable
xx@ubuntu:~$ adb shell "setprop sys.hwc.log 511"
xx@ubuntu:~$ adb shell "logcat -c ;logcat" > hwc.log //提供log信息
```

2. 通过查看 Summary 节点，确认当前 VOP、图层和屏幕等参数信息。

```
xx@ubuntu:~$ cat /sys/kernel/debug/dri/0/summary
Video Port1: DISABLED
Video Port2: ACTIVE
    Connector: eDP-1
        bus_format[100a]: RGB888_1X24
        overlay_mode[0] output_mode[0] color_space[0], eotf:0
    Display mode: 1536x2048p60
        clk[200000] real_clk[200000] type[48] flag[a]
        H: 1536 1548 1564 1612
        V: 2048 2056 2060 2068
    Cluster2-win0: ACTIVE
        win_id: 4
        format: AB24 little-endian (0x34324241)[AFBC] SDR[0] color_space[0]
glb_alpha[0xff]
        rotate: xmirror: 0 ymirror: 0 rotate_90: 0 rotate_270: 0
        csc: y2r[0] r2y[0] csc mode[0]
        zpos: 1
        src: pos[0, 0] rect[1536 x 2048]
        dst: pos[0, 0] rect[1536 x 2048]
        buf[0]: addr: 0x000000007f5e0000 pitch: 4352 offset: 0
```

3. 通过查看 CLK 节点，查看时钟分配是否正常。

```
xx@ubuntu:~$ cat /sys/kernel/debug/clk/clk_summary
enable  prepare protect duty clock count count  count rate  accuracy phase  cycle
---------------------------------------------------------------------------------
---------
 rk808-clkout2           2        2        0       32768          0     0  50000
 xin32k                  0        0        0       32768          0     0  50000
 clkin_gmac              1        1        0   125000000          0     0  50000
    clk_rmii_src         1        4        0   125000000          0     0  50000
       clk_rmii_tx       1        2        0   125000000          0     0  50000
       clk_rmii_rx       0        1        0   125000000          0     0  50000
       clk_mac_ref       0        1        0   125000000          0     0  50000
       clk_mac_refout    0        1        0   125000000          0     0  50000
```

```
 dummy_vpll              0        0        0          0          0     0  50000
 dummy_cpll              0        0        0          0          0     0  50000
    clk_test_pre         0        0        0          0          0     0  50000
       clk_test          0        0        0          0          0     0  50000
       clk_test_frac     0        0        0          0          0     0  50000
    clk_cifout_src       0        0        0          0          0     0  50000
    clk_testout2_pll_src 0        0        0          0          0     0  50000
    clk_testout1_pll_src 0        0        0          0          0     0  50000
    clk_uart_src         0        0        0          0          0     0  50000
       clk_uart3_div     0        0        0          0          0     0  50000
          clk_uart3_frac 0        0        0          0          0     0  50000
       clk_uart2_div     0        0        0          0          0     0  50000
          clk_uart2_frac 0        0        0          0          0     0  50000
       clk_uart1_div     0        0        0          0          0     0  50000
          clk_uart1_frac 0        0        0          0          0     0  50000
    clk_i2s0_div         0        0        0          0          0     0  50000
       clk_i2s0_frac     0        0        0          0          0     0  50000
 xin24m                  20       23       0   24000000          0     0  50000
    clk_timer11          0        0        0   24000000          0     0  50000
    clk_timer10          0        0        0   24000000          0     0  50000
    clk_timer09          0        0        0   24000000          0     0  50000
    clk_timer08          0        0        0   24000000          0     0  50000
```

通过上面的信息检查 CLK 是否满足预期的值，若 EDID 或 DTS 中设置的 DCLK 与实际设置出来的值不同，这是CLK 不能分出对应频点，差距过大而导致无法点屏。要重点核对 summary信息中的 clk[]与 real_clk[] 是否一一对应，时钟树详细信息可以通过#3命令查询。

4. 确认 eDP 接口的链接状态：

```
xx@ubuntu:~$ cat /sys/class/drm/card0-eDP-1/status
connected
```

- connected：代表连接正常；
- disconnected：代表未连接检查hpd信号。

5. 确认当前 eDP 接口寄存器的状态信息，根据寄存器的状值进行对比差异，结合 eDP 挂载情况选择对应的 dump 寄存器如下（以 RK3288 为例）：

```
vopb:
     xx@ubuntu:~$ io -r -4 -l 0x19c 0xff930000
     xx@ubuntu:~$ io -r -4 -l 0x100 0xff931000
vopl:
     xx@ubuntu:~$ io -r -4 -l 0x19c 0xff940000
     xx@ubuntu:~$ io -r -4 -l 0x100 0xff941000
edp:
     xx@ubuntu:~$ io -r -4 -l 0x400 0xff970000
```

Note：通过分析对比寄存器值的状态来判断当前 eDP 接口可能的异常原因。

6. eDP 异常状态之后需要确认 eDP 接口本身是否可以正常输出，通过输出彩条模式看看是否正常，或者休眠唤醒是否可恢复？

```
xx@ubuntu:~$sh display_fault_analyzer.sh -v colorbar
```

7. 异常之后确认重新进行休眠唤醒动作确认显示子系统能否正常重新关流和开流，通过输入如下命令打开Dpcd相关流程的打印信息，休眠唤醒后执行如下命令查看信息并进一步分析。

```
xx@ubuntu:~$ echo 0x100 > /sys/module/drm/parameters/debug
xx@ubuntu:~$ dmesg | grep drm
```

7.4.3 开机闪屏

如果遇到概率性的开机 eDP 屏闪屏现象，通过如下步骤进行优化解决。

1. 提高 eDP 驱动强度是否有改善，参考修改如下：

```
diff --git a/drivers/video/drm/analogix_dp.c b/drivers/video/drm/analogix_dp.c
index 524cc6bd48..dc601e1fa1 100644
--- a/drivers/video/drm/analogix_dp.c
++ b/drivers/video/drm/analogix_dp.c
@ -907,8 +908,12 @ static int analogix_dp_connector_enable(struct display_state
*state)
ret = analogix_dp_set_link_train(dp, dp->video_info.max_lane_count,
dp->video_info.max_link_rate);
if (ret) {
    dev_err(dp->dev, "unable to do link train\n");
    return ret;
}
dp->link_train.training_lane0 =
-   DP_TRAIN_VOLTAGE_SWING_LEVEL_0 |
-   DP_TRAIN_PRE_EMPH_LEVEL_0;
+   DP_TRAIN_VOLTAGE_SWING_LEVEL_3 |
+   DP_TRAIN_PRE_EMPH_LEVEL_3;
```

2. 调整 eDP 眼图。
3. 时钟异常引起的闪屏或者不显示问题。

   有的屏幕对时钟要求比较敏感，非整数或者特殊频点会有闪屏现象。通过如下修改纠正 CLK，U-Boot 阶段增加CLK 矫正后是否正常显示（正常显示的 CLK 预期是141700 Kernel 阶段也需要修改掉）。

```
diff --git a/drivers/video/drm/rockchip_display.c
b/drivers/video/drm/rockchip_display.c
index dfca9ed667..87cb5e4608 100644
--- a/drivers/video/drm/rockchip_display.c
+++ b/drivers/video/drm/rockchip_display.c
-747,6 +747,7 static int display_get_edid_mode(struct display_state *state)
int bpc;

ret = edid_get_drm_mode(conn_state->edid, sizeof(conn_state->edid), mode, &bpc);
+ mode->clock = 270000;(矫正成能正常显示的时钟)
if (!ret) {
conn_state->bpc = bpc;
edid_print_info((void *)&conn_state->edid);
```

4. 如果异常仍然存在，其他参数也得修改为正常显示的屏参。理论上能正确读到 EDID 信息且串口信息上显示Training也通过，是能显示 U-Boot Logo，否则读取的 EDID 信息是错的，参考修改如下：

```
#u-boot$ vi drivers/video/drm/rockchip_display.c
static int display_get_edid_mode(struct display_state *state)
+mode->clock = 270000;//270000000此后面继续增加屏参,补充完整屏参就相当于你dts文件里面填写
timing
+conn_state->bpc = 8;
+mode->flags = 0x5;
+mode->hdisplay = 1280;//对应填写你的屏参
+mode->hsync_start = 1390;//对应填写你的屏参
+mode->hsync_end = 1430;
+mode->htotal = 1650;
+mode->hskew = 0;
+mode->vdisplay = 720;
+mode->vsync_start = 725;
+mode->vsync_end = 730;
+mode->vtotal = 750;
+mode->vrefresh = 60;
```

5. 修改预加重解决闪屏，参考这个修改调整 VP 数组的下标值（RK3568+Android11）：

```
diff --git a/drivers/phy/rockchip/phy-rockchip-naneng-edp.c
b/drivers/phy/rockchip/phy-rockchip-naneng-edp.c
index 379e4318c50a..e8a71f1a5e0a 100644
--- a/drivers/phy/rockchip/phy-rockchip-naneng-edp.c
+++ b/drivers/phy/rockchip/phy-rockchip-naneng-edp.c
@@ -92,6 +92,8 @@ static int rockchip_edp_phy_set_voltages(struct
rockchip_edp_phy *edpphy,
        u32 val;
        for (lane = 0; lane < dp->lanes; lane++) {
+               dp->voltage[lane] = 1;
+               dp->pre[lane]]    = 1;
                val = vp[dp->voltage[lane]][dp->pre[lane]].amp;
                writel(EDP_PHY_TX_AMP(lane, val),
                       edpphy->regs + EDP_PHY_GRF_CON4);
```

7.4.4 双屏或者多屏显示闪屏问题

单屏幕显示正常，三屏幕同时打开 eDP 屏幕闪烁或者 LVDS 花屏等问题。

1. 首先确认三显的时钟分配问题，通过查看 CLK 树可以发现 dclk_vop0 和 dclk_vop2 挂载同一个 HPLL 上，需要把dclk_vop2 挂载 GPLL 上去。

```
xx@ubuntu:~$ cat /sys/kernel/debug/clk/clk_summary
pll_hpll         1    1    0    267969999   0    0  50000
  hpll           2    2    0    267969999   0    0  50000
    dclk_vop2    1    2    0    267969999   0    0  50000
    dclk_vop0    1    2    0    267969999   0    0  50000
    clk_hdmi_ref 0    0    0    267969999   0    0  50000
    hpll_ph0     0    0    0    133984999   0    0  50000
  pll_ppll       1    1    0    200000000   0    0  50000
```

2. 修改三屏幕的 CLK 分配问题。

```
xx@ubuntu:~$ vi ./drivers/clk/rockchip/clk-rk3568.c
diff --git a/drivers/clk/rockchip/clk-rk3568.c b/drivers/clk/rockchip/clk-
rk3568.c
index 73dda8e8046e..fd0de9ffd849 100644
--- a/drivers/clk/rockchip/clk-rk3568.c
+++ b/drivers/clk/rockchip/clk-rk3568.c
@@ -1102,13 +1102,13 @@ static struct rockchip_clk_branch rk3568_clk_branches[]
__initdata = {
        RK3568_CLKGATE_CON(20), 8, GFLAGS),
        GATE(HCLK_VOP, "hclk_vop", "hclk_vo", 0, RK3568_CLKGATE_CON(20), 9,
GFLAGS),
-       COMPOSITE(DCLK_VOP0, "dclk_vop0", hpll_vpll_gpll_cpll_p,
CLK_SET_RATE_PARENT | CLK_SET_RATE_NO_REPARENT,
+       COMPOSITE(DCLK_VOP0, "dclk_vop0", hpll_vpll_gpll_cpll_p, 0,
        RK3568_CLKSEL_CON(39), 10, 2, MFLAGS, 0, 8, DFLAGS,
RK3568_CLKGATE_CON(20), 10, GFLAGS),
        COMPOSITE_DCLK(DCLK_VOP1, "dclk_vop1", hpll_vpll_gpll_cpll_p,
CLK_SET_RATE_PARENT | CLK_SET_RATE_NO_REPARENT,
        RK3568_CLKSEL_CON(40), 10, 2, MFLAGS, 0, 8, DFLAGS,
RK3568_CLKGATE_CON(20), 11, GFLAGS, RK3568_DCLK_PARENT_MAX_PRATE),
-       COMPOSITE(DCLK_VOP2, "dclk_vop2", hpll_vpll_gpll_cpll_p, 0,
+       COMPOSITE(DCLK_VOP2, "dclk_vop2", hpll_vpll_gpll_cpll_p,
CLK_SET_RATE_PARENT | CLK_SET_RATE_NO_REPAREN,
        RK3568_CLKSEL_CON(41), 10, 2, MFLAGS, 0, 8, DFLAGS,
RK3568_CLKGATE_CON(20), 12, GFLAGS),
        GATE(CLK_VOP_PWM, "clk_vop_pwm", "xin24m", 0,
```

3. DTS/VOP/PLL 时钟绑定关系遵循三个 VOP 各自绑定一个 PLL 原则，也是独占 PLL，VP 目标时钟如下：

```
vp0: clock-frequency = <270000000>
vp1: clock-frequency = <71000000>;
vp2: clock-frequency = <72000000>; //1188Mhz / 16 倍分频后为74.25Mhz
&vop {
     status = "okay";
     assigned-clocks = <&cru DCLK_VOP0>, <&cru DCLK_VOP1>, <&cru DCLK_VOP2>;
     assigned-clock-parents = <&cru PLL_GPLL>, <&pmucru PLL_HPLL>, <&cru
PLL_VPLL>;
}; //指定对应的父时钟,就近分频分到接近频率即可。
```

4. 查看时钟是否符合预期，如下独占 PLL 结构：

```
xx@ubuntu:~$ cat /sys/kernel/debug/clk/clk_summary |grep vop
pll_hpll       1    1    0   148500000    0    0    50000
  hpll         2    3    0   148500000    0    0    50000
    dclk_vop1  1    2    0   148500000    0    0    50000
pll_vpll       0    0    0   500000000    0    0    50000
  vpll         0    1    0    24000000    0    0    50000
    dclk_vop2  0    1    0     6000000    0    0    50000
pll_gpll       1    1    0  1188000000    0    0    50000
  gpll         6   14    0  1188000000    0    0    50000
    dclk_vop0  0    1    0  1188000000    0    0    50000
```

7.5 SSC 展频 EMI 干扰问题

对于 EMI 类问题（电磁干扰类）打开 SSC 功能使频谱能量不过于集中，不同的 SOC 对于 SSC 展频功能的默认状态不同。

1. 详细查询 Rockchip 平台的 eDP 驱动文件，我们默认是打开的，详细代码如下：

```
static const struct rockchip_dp_chip_data rk3399_edp = {
 .lcdsel_grf_reg = RK3399_GRF_SOC_CON20,
 .lcdsel_big = HIWORD_UPDATE(0, RK3399_EDP_LCDC_SEL),
 .lcdsel_lit = HIWORD_UPDATE(RK3399_EDP_LCDC_SEL, RK3399_EDP_LCDC_SEL),
 .chip_type = RK3399_EDP,
 .ssc = true,
 };
 static const struct rockchip_dp_chip_data rk3368_edp = {
 .chip_type = RK3368_EDP,
 .ssc = true,
 };
 static const struct rockchip_dp_chip_data rk3288_dp = {
 .lcdsel_grf_reg = RK3288_GRF_SOC_CON6,
 .lcdsel_big = HIWORD_UPDATE(0, RK3288_EDP_LCDC_SEL),
 .lcdsel_lit = HIWORD_UPDATE(RK3288_EDP_LCDC_SEL, RK3288_EDP_LCDC_SEL),
 .chip_type = RK3288_DP,
 .ssc = true,
 static const struct rockchip_dp_chip_data rk3568_edp = {
 .chip_type = RK3568_EDP,
 .ssc = true,
 .audio = true,
 };
 static const struct rockchip_dp_chip_data rk3588_edp[] = {
 {
        .chip_type = RK3588_EDP,
        .spdif_sel = GRF_REG_FIELD(0x0000, 4, 4),
        .i2s_sel = GRF_REG_FIELD(0x0000, 3, 3),
        .edp_mode = GRF_REG_FIELD(0x0000, 0, 0),
        .ssc = true,
        .audio = true,
        .split_mode = true,
    },
    {
        .chip_type = RK3588_EDP,
        .spdif_sel = GRF_REG_FIELD(0x0004, 4, 4),
```

```
        .i2s_sel = GRF_REG_FIELD(0x0004, 3, 3),
        .edp_mode = GRF_REG_FIELD(0x0004, 0, 0),
        .ssc = true,
        .audio = true,
        .split_mode = true,
    },
    { /* sentinel */ }
};
```

2. Rockchip eDP 驱动代码默认 SSC 功能是打开的，但实际是否开启了此功能需要通过下面函数的返回结果来确认 analogix_dp_ssc_supported 。如果需要关闭 SSC 功能仅需将1# 的 SSC 标志位置为 false 即可，SSC 能代码确认如下：

```
if (analogix_dp_ssc_supported(dp))
     analogix_dp_ssc_enable(dp);
else
     analogix_dp_ssc_disable(dp);
bool analogix_dp_ssc_supported(struct analogix_dp_device *dp)
{
   /* Check if SSC is supported by both sides */
     return dp->plat_data->ssc && dp->link_train.ssc;  //这里确认是否真的有打开
}
```

3. 需要屏幕支持和同时打开主控设置，才会启用 SSC 功能；如果要强制打开，修改 analogix_dp_ssc_supported 函数返回值为 true 即可。

   Note：若需要强制开启 SSC 功能，U-Boot 和 Kernel 都需要打开。

4. 打开 SSC 功能后还是有 EMI 的问题，将 eDP/DP 挂到 GPLL 下，打开 GPLL 展频，参考文档如下：RKDocs/common/CLK/Rockchip_Develop_Guide_Pll_Ssmod_Clock_CN.pdf 通过命令行修改成功之后，再把对应的值填写到代码里，参考如下:

```
xx@ubuntu:~$ io -4 0xfd7c01cc 0x00ff000c #设置速率30khz
xx@ubuntu:~$ io -4 0xfd7c01cc 0x3f000500 #设置幅度0.5%
xx@ubuntu:~$ io -4 0xfd7c01cc 0xc0008000 #设置展频模式
xx@ubuntu:~$ io -4 0xfd7c01d0 0x00010001 #打开展频功能
```

Note：RK3036/RK312X/RK322X/RK3188/RK3288/RK3368不支持展频功能。如果是其他的 PLL 类似处理，其基地址修改成对应的 PLL，展频的速率建议使用30KHZ；但展频的幅度没有参考值，需实测以 EMI 测试是否通过为准。

5. 当如步骤#4通过了 EMI 测试，把修改合入系统中，U-Boot 代码修改如下（以 RK356x-Android11 为例）：

```
diff --git a/drivers/clk/rockchip/clk_rk3568.c
b/drivers/clk/rockchip/clk_rk3568.c
index 3ed285c6fb..7ff7e321a8 100644
--- a/drivers/clk/rockchip/clk_rk3568.c
+++ b/drivers/clk/rockchip/clk_rk3568.c
@@ -485,7 +485,10 @@ static int rk3568_pmuclk_probe(struct udevice *dev)
 {
     struct rk3568_pmuclk_priv *priv = dev_get_priv(dev);
     int ret = 0;
-
```

```
+    writel(0x1f000800, &priv->pmucru->pll[2].con3);
+    writel(0x00f00060, &priv->pmucru->pll[2].con3);
+    writel(0x10000000, &priv->pmucru->pll[2].con1);
+    writel(0x00070000, &priv->pmucru->pll[2].con3);
     if (priv->ppll_hz != PPLL_HZ) {
         ret = rockchip_pll_set_rate(&rk3568_pll_clks[PPLL],priv->pmucru,
```

7.6 eDP 眼图调试

1. eDP 屏幕概率性出现黑屏、闪屏等异常，通过示波器测试 eDP 信号眼图；Link Training 概率性不过的情况，可通过修改摆幅和预加重。Rockchip eDP 驱动代码默认使用 Level0 进行强制输出，对于眼图仍需优化的情况，可以修改对应的幅值档位，参考按照协议手册查到对应的数值表格，并按照表格允许的组合方式进行调试，设置后通过抓取的眼图对比效果是否有更优，如下图所示对应的宏代码：

Table 3-1: Allowed Vdiff_pp and Pre-emphasis Combinations

Voltage Swing Level	Pre-emphasis Level			
	Level 0	Level 1	Level 2	Level 3
	Vdiff_pre_pp	Vdiff_pre_pp	Vdiff_pre_pp	Vdiff_pre_pp
0	Required	Required	Required	Optional
1	Required	Required	Required	Not allowed
2	Required	Required	Not allowed	Not allowed
3	Optional	Not allowed	Not allowed	Not allowed

```
#define DP_TRAIN_MAX_SWING_REACHED       (1 << 2)
#define DP_TRAIN_VOLTAGE_SWING_LEVEL_0   (0 << 0)
#define DP_TRAIN_VOLTAGE_SWING_LEVEL_1   (1 << 0)
#define DP_TRAIN_VOLTAGE_SWING_LEVEL_2   (2 << 0)
#define DP_TRAIN_VOLTAGE_SWING_LEVEL_3   (3 << 0)
#define DP_TRAIN_PRE_EMPHASIS_MASK       (3 << 3)
#define DP_TRAIN_PRE_EMPH_LEVEL_0        (0 << 3)
#define DP_TRAIN_PRE_EMPH_LEVEL_1        (1 << 3)
#define DP_TRAIN_PRE_EMPH_LEVEL_2        (2 << 3)
#define DP_TRAIN_PRE_EMPH_LEVEL_3        (3 << 3)
```

2. 若通过强制输出测试后确认是最优组合，请在 Link Training 前先写入该固定组合，确认 Link Training 能否通过。根据实测情况修改对应 Lane 的摆幅和预加重，具体修改 U-Boot 部分代码的位置参考如下：

```
diff --git a/drivers/gpu/drm/bridge/analogix/analogix_dp_core.c
b/drivers/gpu/drm/bridge/analogix/analogix_dp_core.c
 index 866bd9ba2fb9..cf8b6e7c1175 100644---
a/drivers/gpu/drm/bridge/analogix/analogix_dp_core.c
 +++ b/drivers/gpu/drm/bridge/analogix/analogix_dp_core.c
 @@ -392,8 +392,8 @@ static int analogix_dp_link_start(struct analogix_dp_device
*dp)
 /* Set TX voltage-swing and pre-emphasis to minimum */
 for (lane = 0; lane < lane_count; lane++)
        dp->link_train.training_lane[lane] =
 -                DP_TRAIN_VOLTAGE_SWING_LEVEL_0 |
 -                DP_TRAIN_PRE_EMPH_LEVEL_0;
```

```
+                DP_TRAIN_VOLTAGE_SWING_LEVEL_1 |
+                DP_TRAIN_PRE_EMPH_LEVEL_1;
        analogix_dp_set_lane_link_training(dp);
/* Set training pattern 1 *
```

Note：Lane 对应的数据链路通道Lane0/1/2/3，根据实际需要的 Lane 修改进行验证测试。

3. RK3588 eDP 屏幕上电开机后，通过命令行手动设置 eDP 各项参数，并手动休眠唤醒系统后，使写入参数生效；接着配置 eDP 控制器输出特定的码型，然后测试眼图，对于 RK3588 有两个 eDP 接口，基地址如下:

- edp0: 0xfdec0000
- edp1: 0xfded0000
- hdptxphy0：0xfed60000
- hdptxphy1：0xfed70000

```
xx@ubuntu:~$ echo 162000 > /sys/devices/platform/fdec0000.edp/link_rate
162000: 1.62Gbps/lane
270000: 2.7Gbps/lane
540000: 5.4Gbps/lane
xx@ubuntu:~$ echo 4 > /sys/devices/platform/fdec0000.edp/lane_count
1: 1 lane
2: 2 lane
4: 4 lane
xx@ubuntu:~$ echo 2 > /sys/devices/platform/fdec0000.edp/vs
0: voltage swing level 0
1: voltage swing level 1
2: voltage swing level 2
3: voltage swing level 3
xx@ubuntu:~$ echo 0 > /sys/devices/platform/fdec0000.edp/pe
0: pre emphasis level 0
1: pre emphasis level 1
2: pre emphasis level 2
3: pre emphasis level 3
xx@ubuntu:~$ echo 0 > /sys/devices/platform/fdec0000.edp/ssc
0: ssc off
1: ssc on
xx@ubuntu:~$ echo 0 > /sys/devices/platform/fdec0000.edp/pattern
0: Training pattern not sent
1: PRBS 7 bit
2: D10.2 test pattern is sent
3: Sending training pattern 1
4: Sending training pattern 2
5: Sending training pattern 3
6: 80 bit test pattern
7: HBR2 Compliance
```

4. 步骤#3调试命令，需要合入补丁文件这个0001-drm-bridge-analogix_dp-support-for-test-mode-rk3588.patch 文件路径为：Readmine链接。

7.7 eDP 主副屏设置

本章列举不同的 Android 平台设置多屏显示的主副屏幕参数。

1. Android7-Android9 的如下:

- sys.hwc.device.primary //主屏设备类型，用户设置
- sys.hwc.device.extend //副屏设备类型，用户设置
- sys.hwc.device.main //主屏当前设置设备，系统设置
- sys.hwc.device.aux //副屏当前设置设备，系统设置

```
//在system/build.prop加入格式字段,主副屏设备可根据产品自行设置,即可配置主
//副屏,对于HDMI-A-1,HDMI-A-2 问题,目前HWC最新代码支持对应判断,如果不支持
//请升级hwc代码sys.hwc.device.xxx=xx,xx
xx@ubuntu:~$ setprop sys.hwc.device.primary HDMI-A-1//设置HDMI-A-1为主屏
xx@ubuntu:~$ setprop sys.hwc.device.extend eDP//设置eDP为副屏
xx@ubuntu:~$ stop;start//设置完需要Android重启生效,需要reboot生效需要写入build.prop中
```

2. Android12-Android14（以 RK3588+Android12 为例）:

```
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=DSI-1,eDP-2//主屏
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=HDMI-A-1,eDP-1//副屏
```

3. 主副屏幕旋转（以 RK3588+Android12 为例）:

```
setprop persist.sys.rotation.einit-1 1   //0 1 2 3 主屏旋转角度0 90 180 360
setprop persist.sys.rotation.efull-1 1   //主屏满屏显示
setprop persist.sys.rotation.einit-3 1   //0 1 2 3 副屏旋转角度0 90 180 360
setprop persist.sys.rotation.efull-3 1   //副屏满屏显示
```

4. 竖屏横显修改（以 RK3588+Android12 为例）:

```
xx@ubuntu:~$ cd device/rockchip/rk3588/
#rotate screen to 0, 90, 180, 270
#0:   ROTATION_NONE      ORIENTATION_0   : 0
#90:  ROTATION_RIGHT     ORIENTATION_90  : 90
#180: ROTATION_DOWN      ORIENTATION_180: 180
#270: ROTATION_LEFT      ORIENTATION_270: 270
diff --git a/BoardConfig.mk b/BoardConfig.mk
index 71d04b0..d293aee 100644
--- a/BoardConfig.mk
+++ b/BoardConfig.mk
@@ -28,7 +28,7 @@ PRODUCT_KERNEL_CONFIG ?= rockchip_defconfig pcie_wifi.config
 PRODUCT_BOOT_DEVICE := fe2e0000.mmc
 PRODUCT_SDMMC_DEVICE := fe2c0000.mmc
-SF_PRIMARY_DISPLAY_ORIENTATION := 0
+SF_PRIMARY_DISPLAY_ORIENTATION := 90
```

8. 参考文档

1. RKDocs\common\display\Rockchip_Developer_Guide_DRM_Panel_Porting_CN&EN.pdf
2. RKDocs\common\display\Rockchip_Introduction_DRM_Integration_Helper_CN.pdf
3. RKDocs\common\display\Rockchip_Developer_Guide_DRM_Display_Driver_CN.pdf