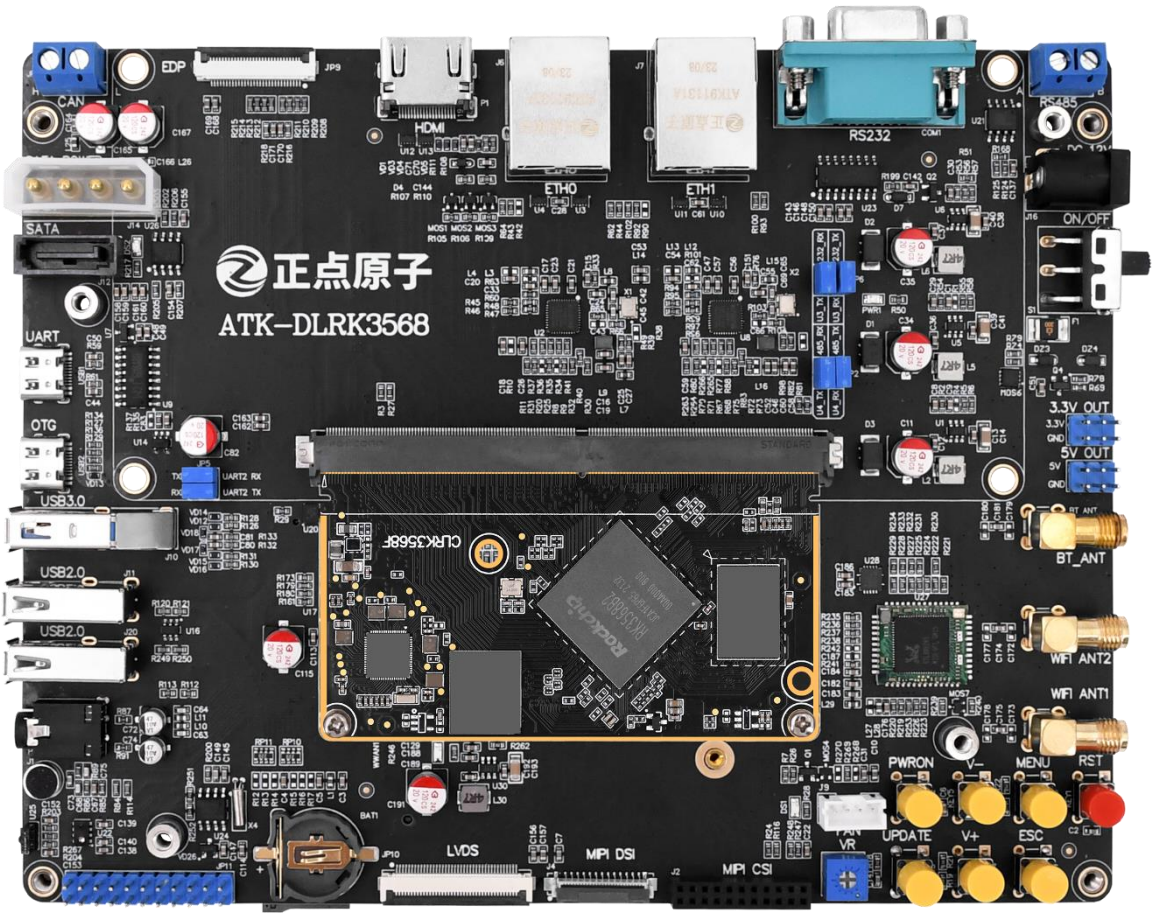
ATK-DLRK3568 开发板三屏显示参考手册 V1.0

-正点原子 ATK-DLRK3568 开发板文档



修订历史:

版本	日期	修改内容
V1.0	2023/06/20	第一次发布

正点原子公司名称 ：广州市星翼电子科技有限公司
原子哥在线教学平台：www.yuanzige.com
开源电子网 / 论坛 ：www.openedv.com
正点原子官方网站 ：www.alientek.com
正点原子淘宝店铺 ：https://openedv.taobao.com
正点原子 B 站视频 ：https://space.bilibili.com/394620890

电话：020-38271790 传真：020-36773971

请下载原子哥 APP，数千讲视频免费学习，更快更流畅。
请关注正点原子公众号，资料发布更新我们会通知。

扫码下载“原子哥”APP

扫码关注正点原子公众号

前言

本文档将向用户介绍正点原子 ATK-DLRK3568 开发板如何使用三屏显示。

第一章 RK3568 显示系统概述

显示系统是指 Rockchip 平台显示输出相关硬件系统的统称，它包括 VOP（Video Output Processor，相当于其它 SoC 的 LCDC，也就是 LCD 控制器）、Display Interface、Panel 等显示信号输出模块。

目前 Rockhip 平台上存在两种 VOP 架构----VOP 1.0 和 VOP 2.0，RK3568 使用的是 VOP 2.0 架构。

1.1 显示系统硬件框图

对于 VOP 2.0 显示架构来说，整个显示系统的硬件框图如下所示：

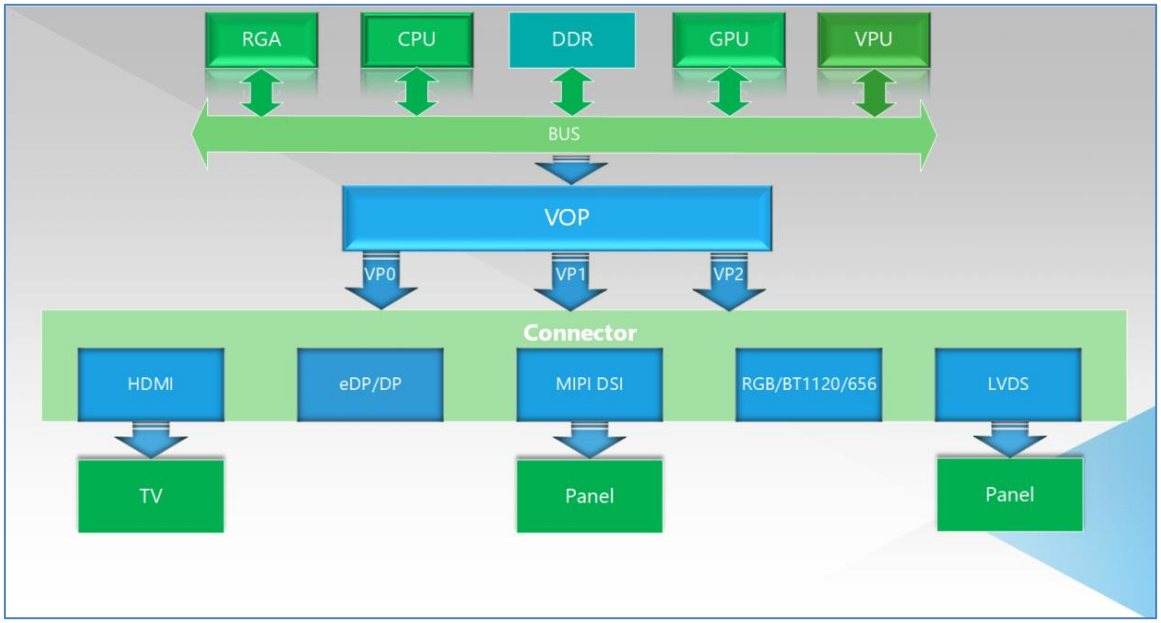


图 1.1.1 VOP 2.0 显示系统硬件框图

上图取自于 RK 文档：

<SDK>/docs/Common/DISPLAY/Rockchip_Developer_Guide_DRM_Display_Driver_CN.pdf

从框图可知，在整个显示通路的最后端，是由 RGA、GPU、VPU 组成的显示图形加速模块构成，它们是专门针对图像处理优化设计的硬件 IP，能够高效的进行图像的生成和进一步处理（譬如 GPU 通过 OpenGL 功能提供图像渲染功能，RGA 可以对图像数据进行缩放、旋转、合成等 2D 处理，VPU 可以高效的进行视频解码），从而减轻 CPU 负担。

经过这些图像加速模块处理后的数据会存放在 DDR 中，然后由 VOP 读取，根据应用需求进行 Alpha 叠加、颜色空间转换、gamma 矫正、HDR 转换等处理后，再发送到对应的显示接口模块（HDMI/MIPI/RGB/LVDS/EDP），这些接口模块会把接收到的数据转换成符合各自协议的数据流，发送到显示器或 LCD 显示屏，最终呈现在用户眼前。

目前 Rockchip 平台上存在两种 VOP 架构----VOP 1.0 和 VOP 2.0（RK3568 对应 VOP 2.0 架构）。它们的主要区别是对多显的支持方式不同，VOP 2.0 采用了统一显示架构，即整个 SoC 上只存在一个 VOP，但是在 VOP 的后端设计了多路独立的 Video Port（以下简称 VP）输出端口，这些 VP 能够同时独立工作，并且输出相互独立的显示时序。譬如 RK3568，有三个 VP，就能同时实现三屏异显。

1.2 显示特性

Rockchip 各平台 VOP 基础特性如下图所示：

SOC	VOP-version	VOP base feature									
		8K	4K	MMU	i-MODE	A/I FBDC	MULTI AREA	BCSH	gamma	3D-LUT	POST-SCALE
RK3066/PX2	V1.0	×	×	×	×	×	×	×	√	×	×
RK3188/PX3	V1.0	×	×	×	×	×	×	×	√	×	×
RK3126/RK3126C	V1.0	×	×	√	×	×	×	√	√	×	×
RK3128/PX3SE	V1.0	×	×	√	√	×	×	√	√	×	×
RK3036	V1.0	×	×	√	√	×	×	√	√	×	×
RK322X/RK312XH	V1.0	×	√	√	√	×	×	√	×	×	√
RK322XH/RK332X	V1.0	×	√	√	√	×	×	√	×	×	√
SOFIA 3GR	V1.0	×	×	√	×	×	×	√	√	×	×
RV1108	V1.0	×	×	×	√	×	×	√	√	×	×
RK3288	V1.0	×	√	√	×	×	√	√	√	×	√
RK3368/PX5	V1.0	×	√	√	×	√	√	√	√	×	√
RK3399	V1.0	×	√	√	√	√	√	√	√	×	√
RK3326/PX30	V1.0	×	×	√	√	√	√	√	√	×	×
RK3308	V1.0	×	×	×	×	×	×	√	√	×	×
RK1808	V1.0	×	×	√	×	×	×	√	√	×	×
RV1109/RV1126	V1.0	×	×	√	×	×	×	√	√	×	×
RK356X	V2.0	×	√	√	√	√	√	√	√	√	√
RK3588	V2.0	√	√	√	√	√	√	√	√	√	×
RV1103/RV1106	V1.0	×	×	×	×	×	×	√	√	×	×

图 1.2.1 各平台 VOP 基础特性

上图取自于 RK 文档：

<SDK>/docs/Common/DISPLAY/Rockchip_Developer_Guide_DRM_Display_Driver_CN.pdf

RK3568 各显示接口最大输出分辨率和协议标准，如下图所示：

RK356X	RGB	1920x1080@60hz	支持 RGB888/RGB666/BT.656/BT.1120
	MIPI	单通道：1920x1080@60hz 双通道：2560x1600@60hz	支持 DSI v1.1，DCS v1.1，DPHY v1.1 协议标准准
	eDP	2560x1600@60hz	支持 DP 1.2a 和 eDP 1.3 协议标准
	HDMI	4096x2160@60hz	支持 HDMI 1.4a 和 2.0a 协议标准

图 1.2.2 各显示接口最大输出分辨率及协议标准

上图取自于 RK 文档：

<SDK>/docs/Common/DISPLAY/Rockchip_Developer_Guide_DRM_Display_Driver_CN.pdf

RK3568 支持多种显示接口，包括 RGB、MIPI DSI、eDP、HDMI 以及 LVDS。

RK3568 VP 与各显示接口的连接关系，如下图所示：

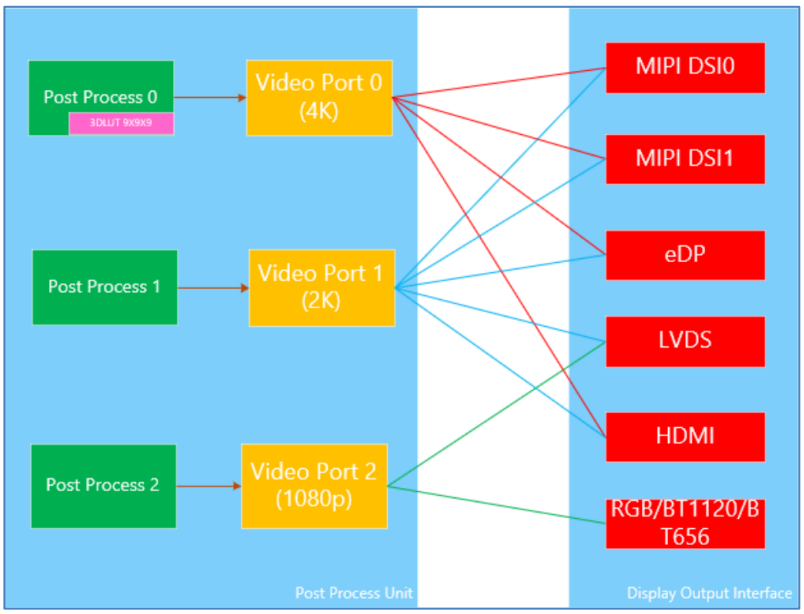


图 1.2.3 RK3568 VP 与各显示接口的关系

RK3568 的 VP0 最大支持 4K 60Hz 输出；VP1 最大支持 2K 60Hz 输出；VP2 最大支持 1080p 60Hz 输出。由上图可知，VP0 可以输出到 MIPI DSI0、MIPI DSI1、eDP 以及 HDMI 这 4 种接口之一，VP1 可以输出到 MIPI DSI0、MIPI DSI1、eDP、LVDS 以及 HDMI 这五种接口之一，而 VP2 只能输出到 LVDS 或 RGB 接口。

以上内容均来自于 RK 提供的文档：

<SDK>/docs/Common/DISPLAY/Rockchip_Developer_Guide_DRM_Display_Driver_CN.pdf

更加详细的内容请参考该文档以及 RK3568 Datasheet：

<SDK>/docs/RK356X/Datasheet/Rockchip_RK3568_Datasheet_V1.1-20210305.pdf

第二章 ATK-DLRK3568 三屏显示介绍

正点原子 ATK-DLRK3568 SDK、内核设备树默认只使能了 MIPI 和 HDMI 显示接口，也就是双屏显示。本章向用户介绍如何在 ATK-DLRK3568 开发板上使用三屏显示。

2.1 ATK-DLRK3568 开发板显示接口介绍

正点原子 ATK-DLRK3568 开发板提供了多种显示接口给到用户，包括 HDMI 显示接口、MIPI DSI 显示接口、LVDS 显示接口、eDP 显示接口，可支持单屏显示、双屏显示以及三屏显示，让用户有更多的选择。

开发板上各显示接口示意图如下所示：

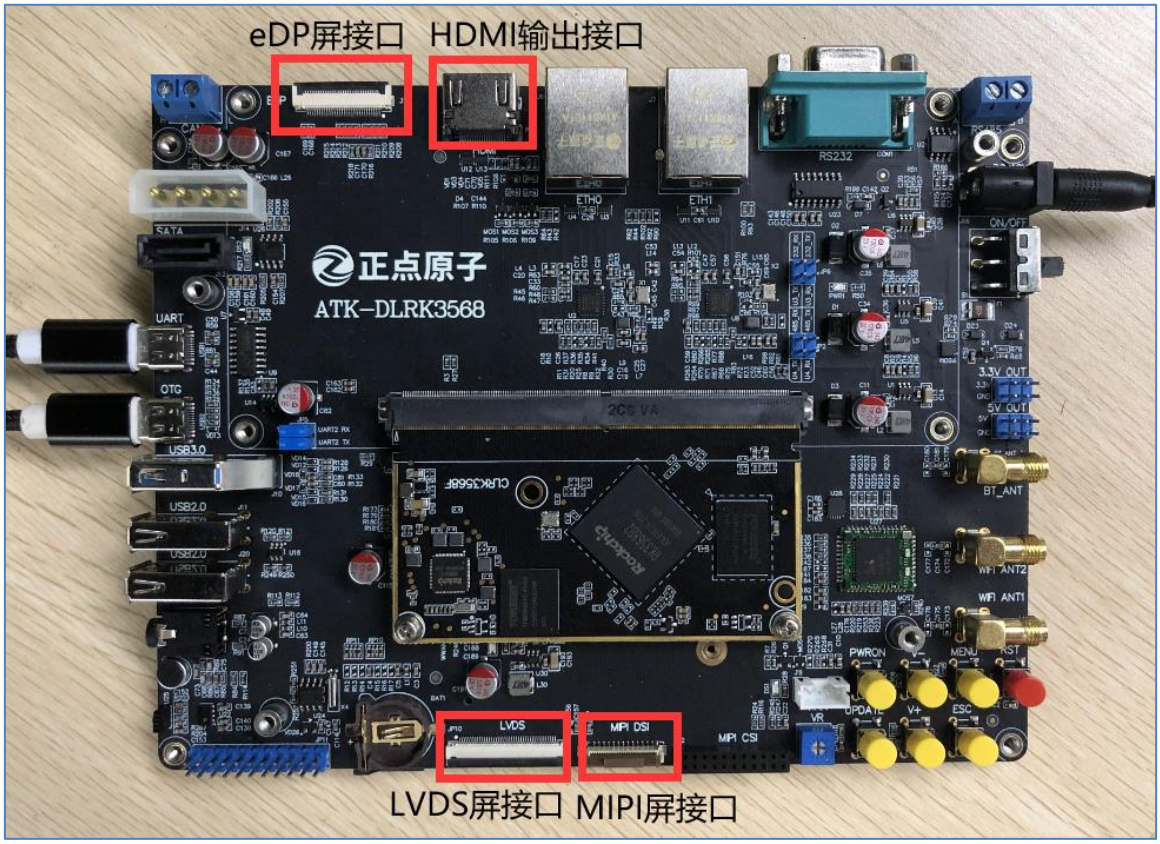


图 2.1.1 显示接口示意图

- **HDMI 显示接口**：用于连接 HDMI 显示器。最大支持 4096x2160@60Hz 或 1080p@120Hz；
- **MIPI DSI 显示接口**：用于连接 MIPI 显示屏。软硬件上仅支持正点原子 5.5 寸 MIPI 屏，包括 1080p MIPI 屏和 720p MIPI 屏以及后续将会推出的 10.1 寸 800x1280 MIPI 屏；
- **LVDS 显示接口**：用于连接 LVDS 显示屏。软硬件上仅支持正点原子 10.1 寸 LVDS 屏；
- **eDP 显示接口**：用于连接 eDP 显示屏。正点原子不做 eDP 屏，如果用户有需求，可参考 ATK-DLRK3568 底板原理图的 eDP 接口部分、自己画转接板连接 eDP 屏（eDP 屏一般在笔记本电脑上用的比较多，可在淘宝自行购买）；

接下来看一下各显示接口的原理图，首先打开 ATK-DLRK3568 底板原理图，底板原理图所在路径为：**开发板光盘 A 盘-基础资料→02、开发板原理图→01、底板原理图→正点原子 ATK-DLRK3568 底板原理图.pdf**。

MIPI DSI 显示接口：

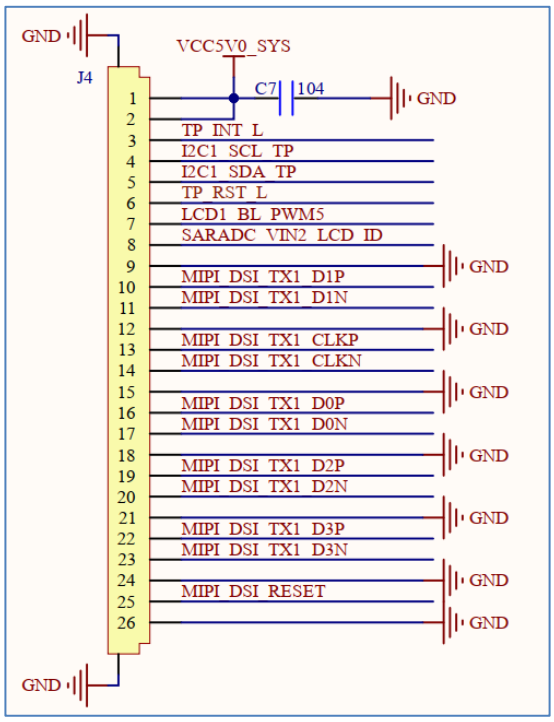


图 2.1.2 MIPI DSI 接口原理图

TP_INT_L：触摸屏中断引脚，连接到 RK3568 GPIO0_B5，触摸屏使用的是 GT911；

I2C1_SCL_TP：触摸屏的 I2C_SCL 引脚，连接到 RK3568 I2C1_SCL；

I2C1_SDA_TP：触摸屏的 I2C_SDA 引脚，连接到 RK3568 I2C1_SDA；

TP_RST_L：触摸屏复位引脚，连接到 RK3568 GPIO0_B6；

LCD1_BL_PWM5：MIPI 屏背光引脚，连接到 RK3568 PWM5；

SARADC_VIN2_LCD_ID：MIPI 屏的 ADC 引脚，连接到 RK3568 SARADC_VIN2，底板的 MIPI DSI 接口可支持正点原子 5.5 寸 1080p MIPI 屏和 5.5 寸 720p MIPI 屏，两种屏读取到的 ADC 值是不一样的，软件上可判断读取到的 ADC 值来识别当前接入的是 720p 还是 1080p MIPI 屏，然后加载对应的 DTB。

MIPI_DSI_TX1_XX：MIPI DSI 信号相关引脚，需要注意的是，RK3568 支持 2 路 MIPI DSI：分别为 DSI0 和 DSI1，底板的 MIPI DSI 接口对应的是 DSI1；

MIPI_DSI_RESET：MIPI 屏复位引脚，连接到 RK3568 GPIO4_B5。

HDMI 显示接口：

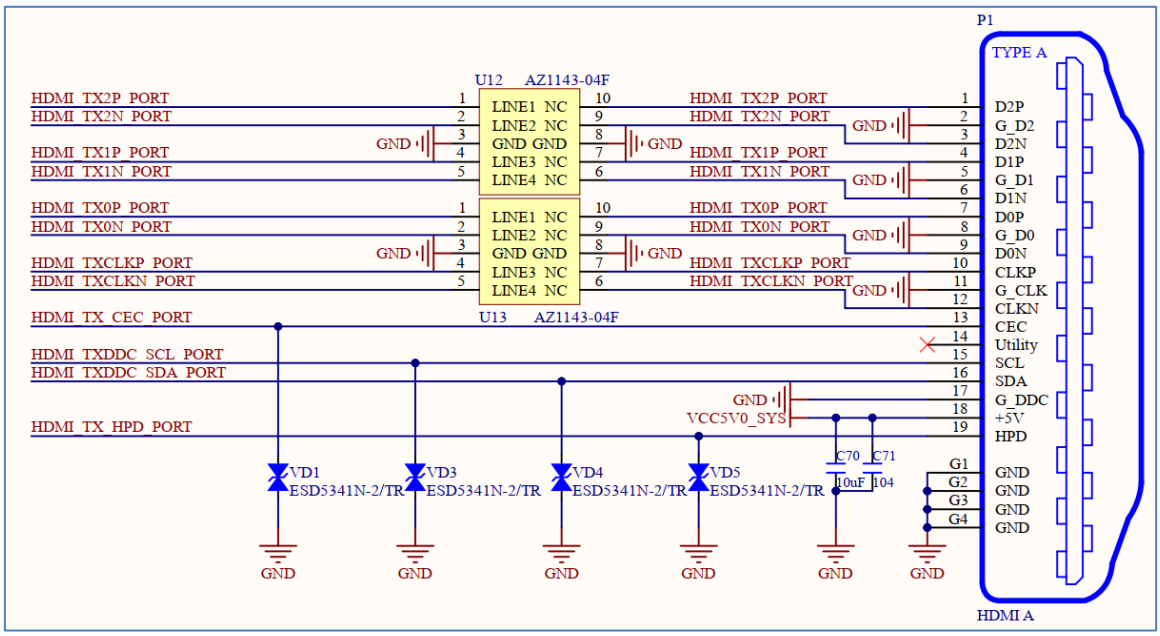


图 2.1.3 HDMI 接口原理图

LVDS 显示接口：

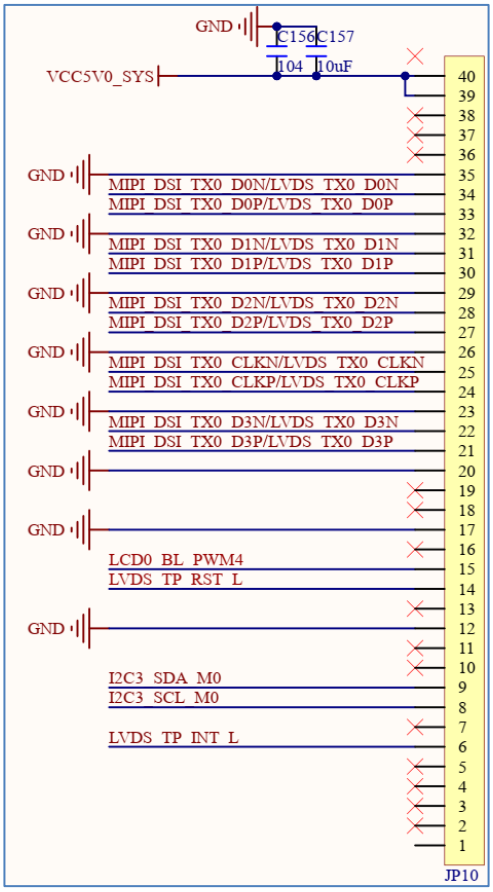


图 2.1.4 LVDS 接口原理图

LVDS_TX0_XX：LVDS 信号相关引脚；
LCD0_BL_PWM4：LVDS 屏背光引脚，连接到 RK3568 PWM4；
LVDS_TP_RST_L：触摸屏复位引脚，连接到 RK3568 GPIO3_C4；

I2C3_SDA_M0：触摸屏的 I2C_SDA 引脚，连接到 RK3568 I2C3_SDA_M0，触摸屏使用的是 GT911；

I2C3_SCL_M0：触摸屏的 I2C_SCL 引脚，连接到 RK3568 I2C3_SCL_M0；

LVDS_TP_INT_L：触摸屏中断引脚，连接到 RK3568 GPIO0_C5。

eDP 显示接口：

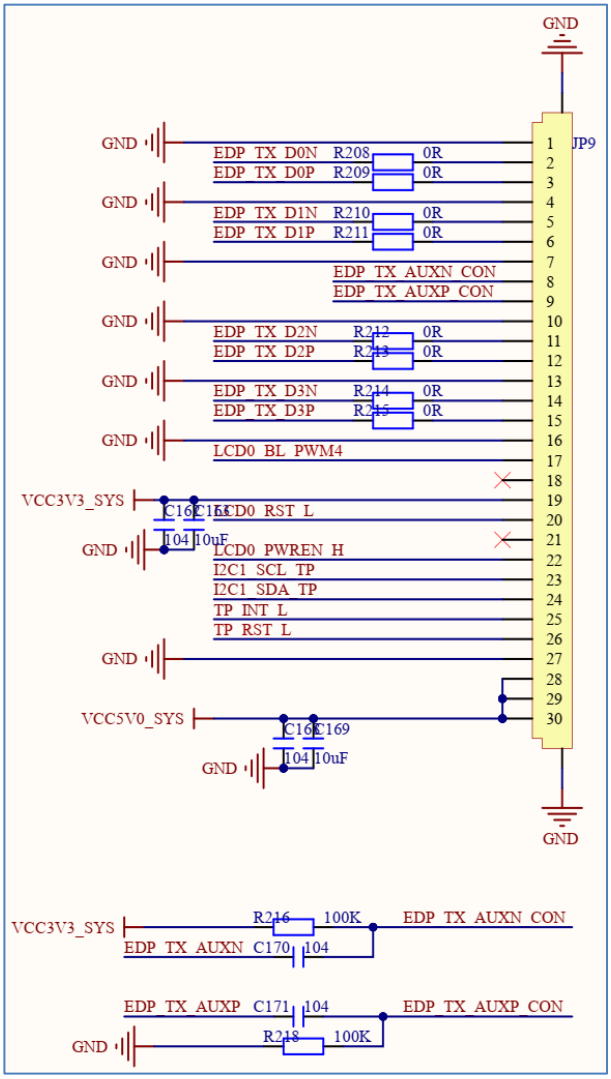


图 2.1.5 eDP 接口原理图

EDP_TX_XX：eDP 信号相关引脚；

LCD0_BL_PWM4：eDP 屏背光引脚，连接到 RK3568 PWM4，与 LVDS 屏共用同一路背光；

LCD0_RST_L：eDP 屏复位引脚，连接到 RK3568 GPIO1_B2；

LCD0_PWREN_H：eDP 屏电源使能控制引脚，连接到 RK3568 GPIO0_C7；

I2C1_SCL_TP：触摸屏的 I2C_SCL 引脚，请忽略！

I2C1_SDA_TP：触摸屏的 I2C_SDA 引脚，请忽略！

TP_INT_L：触摸屏中断引脚，请忽略！

TP_RST_L：触摸屏复位引脚，请忽略！

2.2 设备树配置

Rockchip 平台的所有设备树文件都存放在**<Kernel>arch/arm64/boot/dts/rockchip/**目录下。以 Linux 系统为例，ATK-DLRK3568 开发板对应的设备树文件为 **rk3568-atk-evb1-ddr4-v10-linux.dts**。

rk3568-atk-evb1-ddr4-v10-linux.dts 作为顶层设备树文件，该设备树包含有多个.dtsi 设备树，如下所示：

rk3568-atk-evb1-ddr4-v10-linux.dts
- rk3568-atk-evb1-ddr4-v10.dtsi
  - rk3568.dtsi
  - rk3568-evb.dtsi
- rk3568-linux.dtsi
- rk3568-screen_choose.dtsi
- rk3568-lcds.dtsi

- rk3568.dtsi：RK3568 平台级设备树文件，与具体开发板硬件无关，纯 SoC 级别的设备树文件，由 RK 提供，开发者无需改动该文件！
- rk3568-evb.dtsi：板级通用设备树文件，通常被板级设备树文件所包含；
- rk3568-linux.dtsi：包含 linux 部分特有配置信息，如果是 Android 平台则使用 rk3568-android.dtsi；
- rk3568-atk-evb1-ddr4-v10.dtsi：板级设备树文件，该设备树文件包含 rk3568-evb.dtsi 和 rk3568.dtsi；
- rk3568-screen_choose.dtsi：该设备树用于选择需要使能的 LCD 屏，譬如 HDMI、MIPI、LVDS、VGA，支持单显、双显、三显。用户可在该文件中定义 ATK_LCD_TYPE_XXX 宏（详细内容请看该文件）来使能对应的屏幕，譬如 ATK_LCD_TYPE_HDMI 宏表示使能 HDMI，没定义该宏表示禁用；
- rk3568-lcds.dtsi：该设备树实现了 **ATK_LCD_TYPE_XXX 宏控制 LCD 屏使能**的逻辑 ，以及与该屏相关联的外设，譬如触摸屏（只有使能 MIPI 或 LVDS 时才会使能触摸屏）、HDMI 音频（只有使能 HDMI 时才会使能 HDMI 音频）、背光（只有使能 MIPI 或 LVDS 时才会使能背光）。

通过 rk3568-screen_choose.dtsi 文件来配置需要使能的 LCD 屏，该文件内容如下所示：

```
/*
 * 屏幕选择
 * ATK_LCD_TYPE_MIPI_720P:   正点原子5.5寸 720*1280 MIPI屏
 * ATK_LCD_TYPE_MIPI_1080P:  正点原子5.5寸 1080*1920 MIPI屏
 * ATK_LCD_TYPE_LVDS:        正点原子10.1寸 1280*800 LVDS屏
 * ATK_LCD_TYPE_HDMI:        HDMI显示器
 * ATK_LCD_TYPE_EDP_VGA:     eDP屏或者VGA显示器（硬件默认使能的是VGA接口，若用户需要使用eDP屏，则需修改硬件
 *                           具体情况可以看正点原子RK3568底板原理图！）
 */
/*
 * RK3568可支持三屏显示，也就是三路显示 VP0 VP1 VP2
 * 但是三屏显示需要注意一些问题，具体情况可以看正点原子提供的文档<RK3568三屏显示参考手册>!
 */

/*
 * ATK_LCD_TYPE_MIPI_720P 和 ATK_LCD_TYPE_MIPI_1080P 二选一
 */
#define ATK_LCD_TYPE_MIPI_720P          // 从VP1输入
//#define ATK_LCD_TYPE_MIPI_1080P       // 从VP1输入

/*
 * ATK_LCD_TYPE_HDMI 和 ATK_LCD_TYPE_EDP_VGA 二选一
 */
#define ATK_LCD_TYPE_HDMI               // 从VP0输入
//#define ATK_LCD_TYPE_EDP_VGA          // 从VP0输入

//#define ATK_LCD_TYPE_LVDS             // 从VP2输入

~
```

图 2.2.1 rk3568-screen_choose.dtsi 文件的内容

具体使用方法上图都有说明，这里不再重述！从图中可知，默认情况下，rk3568-screen_choose.dtsi 设备树只使能了 MIPI DSI 接口（支持正点原子 720p MIPI 屏，连接到 VP1）和 HDMI 接口（连接到 VP0），客户可根据自己的需求对其进行更改。

2.2.1 rk3568-lcds.dtsi 设备树详解

在 rk3568-lcds.dtsi 设备树中，会根据用户定义的 ATK_LCD_TYPE_XXX 宏，使能对应的显示接口模块并使能与该屏相关的外设，下面举几个例子给大家说明。

1. HDMI

当用户定义了 ATK_LCD_TYPE_HDMI 宏时：

```
#if defined(ATK_LCD_TYPE_HDMI)
&hdmi {
	status = "okay";
};

&hdmi_in_vp0 {
	status = "okay";
};

&route_hdmi {
	connect = <&vp0_out_hdmi>;
	status = "okay";
};

&hdmi_sound {
	status = "okay";
};
#elif defined(ATK_LCD_TYPE_EDP_VGA)
&edp {
	status = "okay";
};

&edp_in_vp0 {
	status = "okay";
};

&route_edp {
	connect = <&vp0_out_edp>;
	status = "okay";
};

&edp_phy {
	status = "okay";
};

&edp_panel {
	status = "okay";
};

&backlight {
	status = "okay";
};
#endif
```

从上面的内容可知，如果用户定义了 ATK_LCD_TYPE_HDMI 宏：

①、它会使能 hdmi 显示接口：

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

②、接着使能 hdmi_in_vp0 和 route_hdmi 节点：

```
&hdmi_in_vp0 {
        status = "okay";
};

&route_hdmi {
        connect = <&vp0_out_hdmi>;
        status = "okay";
};
```

从图 1.2.3 可知，HDMI 接口的数据可以来自于 VP0（VP0→HDMI）、同样也可来自于 VP1（VP1→HDMI），二选其一，rk3568-lcds.dtsi 设备树中默认将其配置为 VP0。

使能 **hdmi_in_vp0** 节点表示使能 VP0 → HDMI 这条显示通路，同时还需指定 route_hdmi 节点中的 **connect** 属性为<&vp0_out_hdmi>。

如果用户需要使能 VP1 → HDMI 这条显示通路，则需使能 hdmi_in_vp1 节点（禁用 hdmi_in_vp0，它们互斥、不可同时使能），并且将 route_hdmi 节点中的 connect 属性设置为<&vp1_out_hdmi>，如下所示：

```
&hdmi_in_vp1 {
    status = "okay";
};

&route_hdmi {
    connect = <&vp1_out_hdmi>;
    status = "okay";
};
```

③、最后使能与 HDMI 相关联的外设：HDMI 音频（只有使能了 HDMI 接口，HDMI 音频才有效）

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

从图 1.2.3 可知，eDP 接口的数据可以来自于 VP0（VP0→eDP）、同样也可来自于 VP1（VP1→eDP），二选其一，rk3568-lcds.dtsi 设备树中默认将其配置为 VP0。所以在这个默认配置情况下，HDMI 和 eDP/VGA 无法同时使用，只能二选其一，因为一个 VP 只能连接一个显示接口。当然，用户可以对其进行更改，譬如将 eDP 连接到 VP1，这样就可以同时使用 HDMI 和 eDP/VGA。

2. MIPI DSI1

再来看下 MIPI 屏的配置情况，当用户定义了 ATK_LCD_TYPE_MIPI_720P 宏时：

```
#if defined(ATK_LCD_TYPE_MIPI_720P)
&dsi1 {
       status = "okay";
};

&dsi1_in_vp1 {
       status = "okay";
};

&route_dsi1 {
       connect = <&vp1_out_dsi1>;
       status = "okay";
};
```

```
&video_phy1 {
	status = "okay";
};

&dsi_touch {
	status = "okay";
};

&backlight1 {
	status = "okay";
};

&disp_timings1 {		//720p mipi 屏时序
	native-mode = <&dsi1_timing1>;
};

&dsi1_panel {
	status = "okay";
	panel-init-sequence = [ //720p mipi 屏参数
		39 00 04 B9 FF 83 94
		39 00 07 BA 63 03 68 6B B2 C0
		//15 00 02 36 01(倒向显示)
		//15 00 02 36 02(正向显示)
		15 00 02 36 01
		39 00 0B B1 48 12 72 09 32 54 71 71 57 47
		39 00 07 B2 00 80 64 0C 0D 2F
		39 00 16 B4 73 74 73 74 73 74 01 0C 86 75 00 3F 73 74 73 74 73 74
01 0C 86
		39 00 03 B6 6E 6E
		39 00 22 D3 00 00 07 07 40 07 0C 00 08 10 08 00 08 54 15 0A 05 0A
02 15 06 05 06 47 44 0A 0A 4B 10 07 07 0C 40
		39 00 2D D5 1C 1C 1D 1D 00 01 02 03 04 05 06 07 08 09 0A 0B 24 25
18 18 26 27 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 20 21 18 18 18 18
		39 00 2D D6 1C 1C 1D 1D 07 06 05 04 03 02 01 00 0B 0A 09 08 21 20
18 18 27 26 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 25 24 18 18 18 18
		39 00 3B E0 00 0A 15 1B 1E 21 24 22 47 56 65 66 6E 82 88 8B 9A 9D
98 A8 B9 5D 5C 61 66 6A 6F 7F 7F 00 0A 15 1B 1E 21 24 22 47 56 65 65 6E 81 87 8B
98 9D 99 A8 BA 5D 5D 62 67 6B 72 7F 7F
		39 00 03 C0 1F 31
		15 00 02 CC 03
		15 00 02 D4 02
		15 00 02 BD 02
		39 00 0D D8 FF FF FF FF FF FF FF FF FF FF FF FF
		15 00 02 BD 00
		15 00 02 BD 01
		15 00 02 B1 00
		15 00 02 BD 00
		39 00 08 BF 40 81 50 00 1A FC 01
		15 00 02 C6 ED
		05 64 01 11
		05 78 01 29
	];
};
#elif defined(ATK_LCD_TYPE_MIPI_1080P)
……
……
#endif
```

不管是定义 ATK_LCD_TYPE_MIPI_720P、还是 ATK_LCD_TYPE_MIPI_1080P，都会使能 mipi dsi1 显示接口模块，它们之间的区别在于初始化参数以及时序参数的不同。

从上面的内容可知，如果用户定义了 ATK_LCD_TYPE_MIPI_720P 宏：

①、使能 mipi dsi1 显示接口：

```
&dsi1 {
        status = "okay";
};
```

②、使能 dsi1_in_vp1 和 route_dsi1 节点：

```
&dsi1_in_vp1 {
        status = "okay";
};

&route_dsi1 {
        connect = <&vp1_out_dsi1>;
        status = "okay";
};
```

从图 1.2.3 可知，MIPI DSI1 接口的数据可以来自于 VP0（VP0→MIPI DSI1）、同样也可来自于 VP1（VP1→MIPI DSI1），二选其一，rk3568-lcds.dtsi 设备树中默认将其配置为 VP1。

使能 **dsi1_in_vp1** 节点表示使能 VP1 → MIPI DSI1 这条显示通路，同时还需指定 route_dsi1 节点中的 **connect** 属性为<&vp1_out_dsi1>。

如果用户需要使能 VP0 → MIPI DSI1 这条显示通路，则需使能 dsi1_in_vp0 节点（禁用 dsi1_in_vp1，它们互斥、不可同时使能），并且将 route_dsi1 节点中的 connect 属性设置为<&vp0_out_dsi1>，如下所示：

```
&dsi1_in_vp0 {
    status = "okay";
};

&route_dsi1 {
    connect = <&vp0_out_dsi1>;
    status = "okay";
};
```

③、使能 video_phy1

```
&video_phy1 {
        status = "okay";
};
```

有些显示接口模块需要使能对应的 phy 才能工作，譬如 MIPI DSI1（video_phy1）、LVDS（video_phy0）、eDP（edp_phy）。

④、使能触摸屏设备和背光设备：

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

&backlight1 {
        status = "okay";
};
```

dsi_touch 是 MIPI 屏对应的触摸设备节点，该节点定义在 rk3568-atk-evb1-ddr4-v10.dtsi 设备树文件中，具体配置情况请参考该文件。

对于 720p MIPI 屏和 1080p MIPI 屏，以上这些配置都是相同的，区别在于屏幕的初始化参数和时序参数，720p 如下：

```
&disp_timings1 {       //720p mipi 屏时序
       native-mode = <&dsi1_timing1>;
};
```

```
&dsi1_panel {
	status = "okay";
	panel-init-sequence = [ //720p mipi 屏参数
		39 00 04 B9 FF 83 94
		39 00 07 BA 63 03 68 6B B2 C0
		//15 00 02 36 01(倒向显示)
		//15 00 02 36 02(正向显示)
		15 00 02 36 01
		39 00 0B B1 48 12 72 09 32 54 71 71 57 47
		39 00 07 B2 00 80 64 0C 0D 2F
		39 00 16 B4 73 74 73 74 73 74 01 0C 86 75 00 3F 73 74 73 74 73 74
01 0C 86
		39 00 03 B6 6E 6E
		39 00 22 D3 00 00 07 07 40 07 0C 00 08 10 08 00 08 54 15 0A 05 0A
02 15 06 05 06 47 44 0A 0A 4B 10 07 07 0C 40
		39 00 2D D5 1C 1C 1D 1D 00 01 02 03 04 05 06 07 08 09 0A 0B 24 25
18 18 26 27 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 20 21 18 18 18 18
		39 00 2D D6 1C 1C 1D 1D 07 06 05 04 03 02 01 00 0B 0A 09 08 21 20
18 18 27 26 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 25 24 18 18 18 18
		39 00 3B E0 00 0A 15 1B 1E 21 24 22 47 56 65 66 6E 82 88 8B 9A 9D
98 A8 B9 5D 5C 61 66 6A 6F 7F 7F 00 0A 15 1B 1E 21 24 22 47 56 65 65 6E 81 87 8B
98 9D 99 A8 BA 5D 5D 62 67 6B 72 7F 7F
		39 00 03 C0 1F 31
		15 00 02 CC 03
		15 00 02 D4 02
		15 00 02 BD 02
		39 00 0D D8 FF FF FF FF FF FF FF FF FF FF FF FF
		15 00 02 BD 00
		15 00 02 BD 01
		15 00 02 B1 00
		15 00 02 BD 00
		39 00 08 BF 40 81 50 00 1A FC 01
		15 00 02 C6 ED
		05 64 01 11
		05 78 01 29
	];
};
```

1080p 如下：

```
&disp_timings1 {	//1080p mipi 屏时序
	native-mode = <&dsi1_timing0>;
};

&dsi1_panel {
	status = "okay";
	panel-init-sequence = [  //1080p mipi 屏参数
		39 00 04 B9 FF 83 99
		15 00 02 D2 77
		//15 00 02 CC 04(倒向显示)
		//15 00 02 CC 08(正向显示)
		15 00 02 CC 04
		39 00 10 B1 02 04 74 94 01 32 33 11 11 AB 4D 56 73 02 02
		39 00 10 B2 00 80 80 AE 05 07 5A 11 00 00 10 1E 70 03 D4
		39 00 2D B4 00 FF 02 C0 02 C0 00 00 08 00 04 06 00 32 04 0A 08 21
03 01 00 0F B8 8B 02 C0 02 C0 00 00 08 00 04 06 00 32 04 0A 08 01 00 0F B8 01
		39 05 22 D3 00 00 00 00 00 00 06 00 00 10 04 00 04 00 00 00 00 00
00 00 00 00 00 01 00 05 05 07 00 00 00 05 40
		39 05 21 D5 18 18 19 19 18 18 21 20 01 00 07 06 05 04 03 02 18 18
18 18 18 18 2F 2F 30 30 31 31 18 18 18 18
		39 05 21 D6 18 18 19 19 40 40 20 21 02 03 04 05 06 07 00 01 40 40
40 40 40 40 2F 2F 30 30 31 31 40 40 40 40
		39 00 11 D8 A2 AA 02 A0 A2 A8 02 A0 B0 00 00 00 B0 00 00 00
		15 00 02 BD 01
		39 00 11 D8 B0 00 00 00 B0 00 00 00 E2 AA 03 F0 E2 AA 03 F0
		15 00 02 BD 02
```

```
                39 00 09 D8 E2 AA 03 F0 E2 AA 03 F0
                15 00 02 BD 00
                39 00 03 B6 8D 8D
                39 05 37 E0 00 0E 19 13 2E 39 48 44 4D 57 5F 66 6C 76 7F 85 8A 95
9A A4 9B AB B0 5C 58 64 77 00 0E 19 13 2E 39 48 44 4D 57 5F 66 6C 76 7F 85 8A 95
9A A4 9B AB B0 5C 58 64 77
                05 c8 01 11
                05 ff 01 29
        ];
};
```

dsi1_timing1 和 dsi1_timing0 这两个节点分别对应 720p 和 1080p MIPI 屏的时序参数，定义在 rk3568-atk-evb1-ddr4-v10.dtsi 设备树文件中，详细内容请参考该文件。

3. LVDS

最后再来看下 LVDS 屏的配置情况，当用户定义了 ATK_LCD_TYPE_LVDS 宏时：

```
#if defined(ATK_LCD_TYPE_LVDS)
&lvds {
        status = "okay";
};

&lvds_in_vp2 {
        status = "okay";
};

&route_lvds {
        connect = <&vp2_out_lvds>;
        status = "okay";
};

&video_phy0 {
        status = "okay";
};

&lvds_touch {
        status = "okay";
};

&lvds_panel {
        status = "okay";
};

&backlight {
        status = "okay";
};
#endif
```

从上面的内容可知，如果用户定义了 ATK_LCD_TYPE_LVDS 宏：

①、使能 lvds 显示接口：

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

②、使能 lvds_in_vp2 和 route_lvds 节点：

```
&lvds_in_vp2 {
        status = "okay";
};

&route_lvds {
        connect = <&vp2_out_lvds>;
        status = "okay";
};
```

从图 1.2.3 可知，LVDS 接口的数据可以来自于 VP1（VP1→LVDS）、同样也可来自于 VP2（VP2→LVDS），二选其一，rk3568-lcds.dtsi 设备树中默认将其配置为 VP2。

使能 **lvds_in_vp2** 节点表示使能 VP2 → LVDS 这条显示通路，同时还需指定 route_lvds 节点中的 **connect** 属性为<&vp2_out_lvds>。

如果用户需要使能 VP1 → LVDS 这条显示通路，则需使能 lvds_in_vp1 节点（禁用 lvds_in_vp2，它们互斥、不可同时使能），并且将 route_lvds 节点中的 connect 属性设置为<&vp1_out_lvds>，如下所示：

```
&lvds_in_vp1 {
    status = "okay";
};

&route_lvds {
    connect = <&vp1_out_lvds>;
    status = "okay";
};
```

③、使能 video_phy0：

```
&video_phy0 {
        status = "okay";
};
```

④、使能触摸屏、panel 设备、背光设备：

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

&lvds_panel {
        status = "okay";
};

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

LVDS 屏与 eDP 屏公用共一个背光设备<&backlight>，MIPI 屏使用<&backlight1>。LVDS 屏的时序参数定义在 rk3568-atk-evb1-ddr4-v10.dtsi 设备树文件中（& lvds_timing0），详细内容请参考该文件。

4. 总结

关于 rk3568-evb.dtsi 设备树文件的配置情况就给大家介绍这么多，该设备树文件由正点原子提供，其显示通路默认配置情况如下：

VP0 → HDMI
VP0 → eDP
VP1 → MIPI DSI
VP2 → LVDS

所以在上述配置情况下，HDMI 和 eDP/VGA 屏不能同时使用，客户可根据自己的需求对其进行修改，只需满足图 1.2.3 所示对应关系即可！

2.3 使能三屏显示

rk3568-screen_choose.dtsi 设备树中默认只使能了 HDMI 和 MIPI，如果用户需要使用三屏显示，需要用户对该文件进行配置。

2.3.1 配置 rk3568-screen_choose.dtsi

正点原子默认提供了两种三屏显示方案，如下所示：

①：HDMI 显示器(VP0) + MIPI 屏(VP1) + LVDS 屏(VP2)

②：eDP(VP0) + MIPI 屏(VP1) + LVDS 屏(VP2)

除了以上两种方案之外，还可以使用“HDMI(VP0) + eDP/VGA(VP1) + LVDS(VP2)”和“eDP/VGA(VP0)+HDMI(VP1)+LVDS(VP2)”这两种方案，需要用户自行修改 rk3568-evb.dtsi 文件（如果是双屏显示，组合的方案就更多了）。

对于第①种方案，rk3568-screen_choose.dtsi 设备树文件配置方式如下：

```
#define ATK_LCD_TYPE_MIPI_720P      //或者 ATK_LCD_TYPE_MIPI_1080P
#define ATK_LCD_TYPE_LVDS
#define ATK_LCD_TYPE_HDMI
```

如果客户使用的是 720p MIPI 屏，则定义 ATK_LCD_TYPE_MIPI_720P；如果是 1080p MIPI 屏则定义 ATK_LCD_TYPE_MIPI_1080P。

对于第②种方案，rk3568-screen_choose.dtsi 设备树文件配置方式如下：

```
#define ATK_LCD_TYPE_MIPI_720P      //或者 ATK_LCD_TYPE_MIPI_1080P
#define ATK_LCD_TYPE_LVDS
#define ATK_LCD_TYPE_EDP_VGA
```

2.3.2 配置 rk3568-atk-evb1-ddr4-v10.dtsi

除了修改 rk3568-screen_choose.dtsi 文件，还需要对 rk3568-atk-evb1-ddr4-v10.dtsi 文件进行简单的修改，主要是修改显示时钟大小。

先解释下为什么需要修改显示时钟？

显示相关的时钟要求比较多，DCLK（显示时钟）一般要求任意频率，能够满足任意分辨率（任意时序参数）的屏，因为不同的显示分辨率对应的 DCLK 频率也是不同的。所以在 RK 平台上，DCLK 一般是独占一个 PLL 的，由于独占了一个 PLL，所以这个 PLL 的频率会根据屏的要求变化，DCLK 也就可以输出任意频率了。

RK3568 有三个 VP 端口：VP0、VP1、VP2，能同时实现三屏异显。所以对于 RK3568 来说，三屏显示理论上需要三个独立的 PLL，每个 VP 独占一个。但实际上，RK3568 无法满足这个要求，它只有两个 PLL 给显示用（HPLL 和 VPLL），其余 PLL（NPLL、CPLL、GPLL、DPLL、APLL、PPLL、MPLL）都给其它外设使用了。

所以 RK3568 支持双显独占 HPLL 和 VPLL，支持双路任意分辨率异显。对于三路显示，也就意味着三路 VP 共用两个 PLL：一路 VP 独占一个 PLL，另外两路 VP 共用一个 PLL；在这种情况下，独占一个 PLL 的那路 VP 可以支持任意分辨率显示，而共用一个 PLL 的那两路 VP 无法实现任意分辨率显示，因为一个 PLL 无法同时满足两路 DCLK 输出任意频率；为了保证这两路 VP 能够正常显示，要求它们的 DCLK 频率相等或相接近，否则可能会显示异常。**所以，其实是不建议客户使用三屏显示。**

通过上面的分析可知，对于三屏显示来说，共用一个 PLL 的那两路 VP、它们的 DCLK 频率要求相等或接近，否则可能会出现问题。那我们就根据这个要求对 rk3568-atk-evb1-ddr4-v10.dtsi 文件进行修改。

打开 rk3568-atk-evb1-ddr4-v10.dtsi 设备树文件，可以找到如下内容：

```
&vop {
        assigned-clocks = <&cru DCLK_VOP0>, <&cru DCLK_VOP1>, <&cru DCLK_VOP2>;
        assigned-clock-parents = <&pmucru PLL_HPLL>, <&cru PLL_VPLL>, <&cru PLL_VPLL>;
};
```

这个配置表示：VP0 独占 HPLL，VP1 和 VP2 共用 VPLL。当然客户也可对其进行更改，这里不再多说。

正点原子默认提供了两种三屏显示方案，如下所示：

①：HDMI 显示器(VP0) + MIPI 屏(VP1) + LVDS 屏(VP2)

②：eDP/VGA 显示器(VP0) + MIPI 屏(VP1) + LVDS 屏(VP2)

因为 VP1 和 VP2 共用 VPLL，所以不管是第一种方案还是第二种方案，都要求 MIPI 屏和 LVDS 屏的 DCLK 相等或相近。

1. 720p MIPI 屏

如果使用 720p MIPI 屏（该屏 60Hz 刷新率所对应的时钟频率为 65MHz），只需将 LVDS 屏的时钟频率修改为 65MHz，修改如下(rk3568-atk-evb1-ddr4-v10.dtsi)：

```
97        lvds_panel: lvds-panel {
98                compatible = "simple-panel";
99                backlight = <&backlight>;
100               power-supply = <&vcc5v0_sys>;
101               enable-delay-ms = <20>;
102               prepare-delay-ms = <20>;
103               unprepare-delay-ms = <20>;
104               disable-delay-ms = <20>;
105               bus-format = <MEDIA_BUS_FMT_RGB888_1X7X4_SPWG>;
106               width-mm = <230>;
107               height-mm = <150>;
108
109               display-timings {
110                       native-mode = <&lvds_timing0>;
111
112                       lvds_timing0: timing0 {
113                               //clock-frequency = <71100000>; //三屏显示情况下,可将频率设置为65MHz,以保证...
114                               clock-frequency = <65000000>;
115                               hactive = <1280>;
116                               vactive = <800>;
117                               hback-porch = <80>;
118                               hfront-porch = <70>;
119                               vback-porch = <10>;
120                               vfront-porch = <10>;
121                               hsync-len = <10>;
122                               vsync-len = <3>;
123                               hsync-active = <0>;
124                               vsync-active = <0>;
125                               de-active = <0>;
126                               pixelclk-active = <0>;
127                       };
128               };
```

将原本的71.1M修改为65M

将 LVDS 屏的时钟频率修改为 65MHz，因为 720p MIPI 的时钟频率也是 65MHz。

2. 1080p MIPI 屏

如果使用 1080p MIPI 屏，需将 LVDS 屏时钟频率以及 1080p MIPI 屏时钟频率都设置为 65MHz，如下：

```
97          lvds_panel: lvds-panel {
98                  compatible = "simple-panel";
99                  backlight = <&backlight>;
100                 power-supply = <&vcc5v0_sys>;
101                 enable-delay-ms = <20>;
102                 prepare-delay-ms = <20>;
103                 unprepare-delay-ms = <20>;
104                 disable-delay-ms = <20>;
105                 bus-format = <MEDIA_BUS_FMT_RGB888_1X7X4_SPWG>;
106                 width-mm = <230>;
107                 height-mm = <150>;
108
109                 display-timings {
110                         native-mode = <&lvds_timing0>;
111
112                         lvds_timing0: timing0 {
113                                 //clock-frequency = <71100000>; //三屏显示情况下,可将频率设置为65MHz,以保证...
114                                 clock-frequency = <65000000>;
115                                 hactive = <1280>;
116                                 vactive = <800>;
117                                 hback-porch = <80>;
118                                 hfront-porch = <70>;
119                                 vback-porch = <10>;
120                                 vfront-porch = <10>;
121                                 hsync-len = <10>;
122                                 vsync-len = <3>;
123                                 hsync-active = <0>;
124                                 vsync-active = <0>;
125                                 de-active = <0>;
126                                 pixelclk-active = <0>;
127                         };
128                 };
```

将原本的71.1M修改为65M

```
311         dsi1_timing0: timing0 {
312                 //clock-frequency = <75000000>; //三屏显示情况下,可将频率设置为65MHz,以保证...
313                 clock-frequency = <65000000>;
314                 hactive = <1080>;
315                 vactive = <1920>;
316                 hfront-porch = <48>;
317                 hsync-len = <8>;
318                 hback-porch = <52>;
319                 vfront-porch = <16>;
320                 vsync-len = <6>;
321                 vback-porch = <15>;
322                 hsync-active = <0>;
323                 vsync-active = <0>;
324                 de-active = <0>;
325                 pixelclk-active = <0>;
326         };
```

将原本的75M修改为65M

1080p MIPI 屏 60Hz 刷新率所对应的时钟频率为 136MHz，如果将其修改为 65MHz，则会导致其刷新率直接减半，严重影响其响应速度！这里只是作为一个测试，在实际项目应用当中，应谨慎选择！

2.3.3 编译设备树

修改完设备树文件后保存退出，然后在内核源码目录下执行如下命令重新编译 rk3568-atk-evb1-ddr4-v10-linux.dts 设备树文件（如果没编译内核、可先运行./make.sh 脚本编译）：

```
make ARCH=arm64 rockchip/rk3568-atk-evb1-ddr4-v10-linux.dtb -j24
```

```
tgg@tgg-virtual-machine:~/rk3568_linux_sdk/kernel$
tgg@tgg-virtual-machine:~/rk3568_linux_sdk/kernel$ make ARCH=arm64 rockchip/rk3568-atk-evb1-ddr4-v10-linux.dtb -j24
  DTC     arch/arm64/boot/dts/rockchip/rk3568-atk-evb1-ddr4-v10-linux.dtb
tgg@tgg-virtual-machine:~/rk3568_linux_sdk/kernel$
```

接着执行如下命令将 rk3568-atk-evb1-ddr4-v10-linux.dtb 和 logo 图片一起打包成 resource.img：

```
cp arch/arm64/boot/dts/rockchip/rk3568-atk-evb1-ddr4-v10-linux.dtb ./
./scripts/resource_tool rk3568-atk-evb1-ddr4-v10-linux.dtb logo.bmp logo_kernel.bmp
```

```
tgg@tgg-virtual-machine:~/rk3568_linux_sdk/kernel$ cp arch/arm64/boot/dts/rockchip/rk3568-atk-evb1-ddr4-v10-linux.dtb ./
tgg@tgg-virtual-machine:~/rk3568_linux_sdk/kernel$
tgg@tgg-virtual-machine:~/rk3568_linux_sdk/kernel$ ./scripts/resource_tool rk3568-atk-evb1-ddr4-v10-linux.dtb logo.bmp logo_kernel.bmp
Pack to resource.img successed!
tgg@tgg-virtual-machine:~/rk3568_linux_sdk/kernel$
```

最后执行如下命令，将 resource.img、内核镜像 Image 打包成 boot.img：

../device/rockchip/common/mk-fitimage.sh kernel/boot.img device/rockchip/rk356x/boot.its

```
tgg@tgg-virtual-machine:~/rk3568_linux_sdk/kernel$
tgg@tgg-virtual-machine:~/rk3568_linux_sdk/kernel$ ../device/rockchip/common/mk-fitimage.sh kernel/boot.img device/rockchip/rk356x/boot.its
fdt {
kernel {
resource {
FIT description: U-Boot FIT source file for arm
Created:         Tue Apr 18 18:25:24 2023
 Image 0 (fdt)
  Description:  unavailable
  Created:      Tue Apr 18 18:25:24 2023
  Type:         Flat Device Tree
  Compression:  uncompressed
  Data Size:    142023 Bytes = 138.69 KiB = 0.14 MiB
  Architecture: AArch64
  Load Address: 0xffffff00
  Hash algo:    sha256
  Hash value:   51fce36661f13c5252f194e5bf5eb57b67decb8b7d02803ab578eefbc265b8a5
 Image 1 (kernel)
  Description:  unavailable
  Created:      Tue Apr 18 18:25:24 2023
  Type:         Kernel Image
  Compression:  uncompressed
  Data Size:    22603784 Bytes = 22074.01 KiB = 21.56 MiB
  Architecture: AArch64
  OS:           Linux
  Load Address: 0xffffff01
  Entry Point:  0xffffff01
  Hash algo:    sha256
  Hash value:   7cd97842435529df084398be14bbb13708240cbd5144841dc643a260ddf944ba
 Image 2 (resource)
  Description:  unavailable
  Created:      Tue Apr 18 18:25:24 2023
  Type:         Multi-File Image
  Compression:  uncompressed
  Data Size:    500736 Bytes = 489.00 KiB = 0.48 MiB
  Hash algo:    sha256
  Hash value:   73414dc8a2cef20c69df681be907276b13bbfc4786141d38c12ec76439e8553b
 Default Configuration: 'conf'
 Configuration 0 (conf)
  Description:  unavailable
  Kernel:       kernel
  FDT:          fdt
tgg@tgg-virtual-machine:~/rk3568_linux_sdk/kernel$ _
```

将生成的<Kernel>/boot.img 镜像烧写到开发板 boot 分区，烧写完成后重启进入系统。正常情况下，系统启动过程中三个屏都会显示启动 logo（先提前连接 MIPI 屏、LVDS 屏以及 HDMI 显示器）。

第三章 常用的 Debug 手段

详细内容请参考 RK 官方文档。

3.1 dump 当前的显示状态

进入系统后，执行命令查看当前显示信息：

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

```
root@RK356X:/# cat /sys/kernel/debug/dri/0/summary
Video Port0: ACTIVE
    Connector: HDMI-A-1
        bus_format[2025]: YUV8_1X24
        overlay_mode[1] output_mode[f] color_space[3]
    Display mode: 1920x1080p61
        clk[151000] real_clk[151000] type[48] flag[9]
        H: 1920 2008 2052 2200
        V: 1080 1084 1089 1125
    Smart1-win0: ACTIVE
        win_id: 1
        format: XR24 little-endian (0x34325258) SDR[0] color_space[0] glb_alpha[0xff]
        rotate: xmirror: 0 ymirror: 0 rotate_90: 0 rotate_270: 0
        csc: y2r[0] r2y[1] csc mode[1]
        zpos: 1
        src: pos[0, 0] rect[1280 x 800]
        dst: pos[95, 0] rect[1729 x 1080]
        buf[0]: addr: 0x0000000000d58000 pitch: 5120 offset: 0
Video Port1: ACTIVE
    Connector: DSI-1
        bus_format[100a]: RGB888_1X24
        overlay_mode[0] output_mode[0] color_space[0]
    Display mode: 720x1280p60
        clk[65000] real_clk[65000] type[48] flag[a]
        H: 720 768 776 828
        V: 1280 1296 1302 1317
    Smart0-win0: ACTIVE
        win_id: 0
        format: XR24 little-endian (0x34325258) 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: 0
        src: pos[0, 0] rect[1280 x 800]
        dst: pos[0, 415] rect[720 x 450]
        buf[0]: addr: 0x0000000000d58000 pitch: 5120 offset: 0
Video Port2: ACTIVE
    Connector: LVDS-1
        bus_format[1011]: RGB888_1X7X4_SPWG
        overlay_mode[0] output_mode[0] color_space[0]
    Display mode: 1280x800p55
        clk[65000] real_clk[65000] type[48] flag[a]
        H: 1280 1350 1360 1440
        V: 800 810 813 823
    Esmart1-win0: ACTIVE
        win_id: 2
        format: XR24 little-endian (0x34325258) 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: 2
        src: pos[0, 0] rect[1280 x 800]
        dst: pos[0, 0] rect[1280 x 800]
        buf[0]: addr: 0x0000000000d58000 pitch: 5120 offset: 0
root@RK356X:/#
```

HDMI的信息

MIPI屏的信息

LVDS屏的信息

查看 Connector 当前连接状态：

```
cat /sys/class/drm/card0-DSI-1/status
```

```
root@RK356X:/#
root@RK356X:/# cat /sys/class/drm/card0-DSI-1/status
connected
root@RK356X:/#
```

查看 Connector 当前使能状态：

```
cat /sys/class/drm/card0-DSI-1/enabled
```

```
root@RK356X:/#
root@RK356X:/# cat /sys/class/drm/card0-DSI-1/enabled
enabled
root@RK356X:/#
root@RK356X:/#
```

查看 Connector 支持的显示模式：

```
cat /sys/class/drm/card0-DSI-1/modes
```

```
root@RK356X:/#
root@RK356X:/# cat /sys/class/drm/card0-DSI-1/modes
720x1280
root@RK356X:/#
```

3.2 查看当前显示时钟

执行如下命令获取整个时钟树：

```
cat /sys/kernel/debug/clk/clk_summary
```

执行如下命令只获取 VOP 相关时钟：

```
cat /sys/kernel/debug/clk/clk_summary | grep vop
```

```
root@RK356X:/# cat /sys/kernel/debug/clk/clk_summary | grep vop
    clk_vop_pwm                     0        0        0    24000000          0     0  50000
        dclk_vop2                   1        2        0    65000000          0     0  50000
        dclk_vop1                   1        2        0    65000000          0     0  50000
        aclk_vop_pre                1        2        0   500000000          0     0  50000
           aclk_vop                 1        3        0   500000000          0     0  50000
               hclk_vop             2        5        0   148500000          0     0  50000
        dclk_vop0                   1        2        0   151000000          0     0  50000
root@RK356X:/#
```

从上图可知，vp0 的时钟频率为 151MHz（HDMI），vp1 和 vp2 的时钟频率均为 65MHz。

3.3 强行开启/关闭显示设备

关闭或开启 HDMI 显示：

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

关闭或开启 MIPI 屏显示：

```
echo off > /sys/class/drm/card0-DSI-1/status
echo on > /sys/class/drm/card0-DSI-1/status
```

关闭或开启 LVDS 屏显示：

```
echo off > /sys/class/drm/card0-LVDS-1/status
echo on > /sys/class/drm/card0-LVDS-1/status
```

3.4 HDMI 相关

获取 EDID 信息：

```
cat /sys/class/drm/card0-HDMI-A-1/edid > /userdata/edid.bin
```

查看 HDMI 状态：

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

```
root@RK356X:/#
root@RK356X:/# cat /sys/kernel/debug/dw-hdmi/status
PHY: enabled                    Mode: HDMI
Pixel Clk: 151000000Hz          TMDS Clk: 151000000Hz
Color Format: YUV444            Color Depth: 8 bit
Colorimetry: ITU.BT709          EOTF: Off
root@RK356X:/#
```

3.5 modetest 命令

modetest 是由 libdrm 提供的测试程序，可以查询显示设备的特性，进行基本的显示测试，以及设置显示模式等。执行如下命令可查看 modetest 工具的帮助信息：

```
modetest --help
```

```
root@RK356X:/# modetest --help
usage: modetest [-acDdefMPpsCvrw]

 Query options:

        -c      list connectors
        -e      list encoders
        -f      list framebuffers
        -p      list CRTCs and planes (pipes)

 Test options:

        -P <plane_id>@<crtc_id>:<w>x<h>[+<x>+<y>][*<scale>][@<format>]  set a plane
        -s <connector_id>[,<connector_id>][@<crtc_id>]:[#<mode index>]<mode>[-<vrefresh>][@<format>]    set a mode
        -C      test hw cursor
        -v      test vsynced page flipping
        -r      set the preferred mode for all connectors
        -w <obj_id>:<prop_name>:<value> set property
        -a      use atomic API
        -F pattern1,pattern2    specify fill patterns

 Generic options:

        -d      drop master after mode set
        -M module       use the given driver
        -D device       use the given device

        Default is to dump all info.
root@RK356X:/#
```

Dump 所有信息：

```
modetest -M rockchip
```

执行上述命令打印信息特别多，以下只截取部分信息：

```
root@RK356X:/# modetest -M rockchip
Encoders:
id      crtc    type    possible crtcs  possible clones
151     0       Virtual 0x00000007      0x00000000
153     131     LVDS    0x00000004      0x00000000
155     85      TMDS    0x00000001      0x00000000
166     115     DSI     0x00000002      0x00000000

Connectors:
id      encoder status          name            size (mm)       modes   encoders
154     153     connected       LVDS-1          230x150         1       153
  modes:
        index name refresh (Hz) hdisp hss hse htot vdisp vss vse vtot
  #0 1280x800 54.85 1280 1350 1360 1440 800 810 813 823 65000 flags: nhsync,
nvsync; type: preferred, driver
……
……
156     155     connected       HDMI-A-1        520x310         35      155
  modes:
        index name refresh (Hz) hdisp hss hse htot vdisp vss vse vtot
  #0 1920x1080 61.01 1920 2008 2052 2200 1080 1084 1089 1125 151000 flags: phsync,
nvsync; type: preferred, driver
……
……
167     166     connected       DSI-1           0x0             1       166
  modes:
```

```
        index name refresh (Hz) hdisp hss hse htot vdisp vss vse vtot
  #0 720x1280 59.61 720 768 776 828 1280 1296 1302 1317 65000 flags: nhsync,
nvsync; type: preferred, driver
……
……
CRTCs:
id      fb      pos     size
85      218     (0,0)   (1920x1080)
  #0 1920x1080 61.01 1920 2008 2052 2200 1080 1084 1089 1125 151000 flags: phsync,
nvsync; type: preferred, driver
……
……
115     218     (0,0)   (720x1280)
  #0 720x1280 59.61 720 768 776 828 1280 1296 1302 1317 65000 flags: nhsync,
nvsync; type: preferred, driver
……
……
131     218     (0,0)   (1280x800)
  #0 1280x800 54.85 1280 1350 1360 1440 800 810 813 823 65000 flags: nhsync,
nvsync; type: preferred, driver
……
……
Planes:
id      crtc    fb      CRTC x,y        x,y     gamma size      possible crtcs
57      85      218     0,0             0,0     0               0x00000001
  formats: XR24 AR24 XB24 AB24 RG24 BG24 RG16 BG16
  props:
        8 type:
                flags: immutable enum
                enums: Overlay=0 Primary=1 Cursor=2
                value: 1
……
……
71      0       0       0,0             0,0     0               0x00000001
  formats: XR24 AR24 XB24 AB24 RG24 BG24 RG16 BG16 NV12 NV16 NV24 NA12 NA16 NA24
YUYV
  props:
        8 type:
                flags: immutable enum
                enums: Overlay=0 Primary=1 Cursor=2
                value: 2
……
……
87      115     218     0,0             0,0     0               0x00000002
  formats: XR24 AR24 XB24 AB24 RG24 BG24 RG16 BG16
  props:
        8 type:
                flags: immutable enum
                enums: Overlay=0 Primary=1 Cursor=2
                value: 1
……
……
101     0       0       0,0             0,0     0               0x00000002
  formats: XR24 AR24 XB24 AB24 RG24 BG24 RG16 BG16 NV12 NV16 NV24 NA12 NA16 NA24
YUYV
  props:
        8 type:
                flags: immutable enum
                enums: Overlay=0 Primary=1 Cursor=2
                value: 2
……
……
117     131     218     0,0             0,0     0               0x00000004
  formats: XR24 AR24 XB24 AB24 RG24 BG24 RG16 BG16 NV12 NV16 NV24 NA12 NA16 NA24
YVYU VYUY
  props:
        8 type:
```

```
                flags: immutable enum
                enums: Overlay=0 Primary=1 Cursor=2
                value: 1
……
……
133     0       0       0,0             0,0     0               0x00000002
  formats: XR24 AR24 XB24 AB24 RG24 BG24 RG16 BG16 NV12 NV16 NV24 NA12 NA16 NA24
YVYU VYUY
  props:
        8 type:
                flags: immutable enum
                enums: Overlay=0 Primary=1 Cursor=2
                value: 0
……
……
```

上述命令会打印所有组件的信息，包括：Connector、encoder、framebuffer、CRTC 以及 plane，每个组件都有一个 id，modetest 通过这些 id 来引用这些组件。

从打印信息可知，Connector LVDS-1 的 id 为 154，它对应的 Encoder id 为 153，其它处于 connected 状态。

Connector HDMI-A-1 的 id 为 156，它对应的 Encoder id 为 155，其它处于 connected 状态。

Connector DSI-1 的 id 为 167，它对应的 Encoder id 为 166，其它处于 connected 状态。

通过如下命令可以单独查看所有 Encoder 的信息：

```
modetest -M rockchip -e
```

```
root@RK356X:/# modetest -M rockchip -e
Encoders:
id      crtc    type    possible crtcs  possible clones
151     0       Virtual 0x00000007      0x00000000
153     131     LVDS    0x00000004      0x00000000
155     85      TMDS    0x00000001      0x00000000
166     115     DSI     0x00000002      0x00000000
```

可以通过 modetest 在对应的显示接口上输出各种颜色的彩条，利用该功能对 DRM 驱动进行简单的验证。

执行如下命令在 HDMI 屏上输出彩条：

```
modetest -M rockchip -s 156@85:1920x1080
```

```
root@RK356X:/#
root@RK356X:/# modetest -M rockchip -s 156@85:1920x1080
setting mode 1920x1080-61.01Hz on connectors 156, crtc 85
```

其中 156 是 HDMI-A-1 的 id，85 是 CRTC 的 id（也就是 VP0 的 id）。显示效果如下：

执行如下命令在 MIPI 屏上输出彩条：

```
modetest -M rockchip -s 167@115:720x1280
```

```
root@RK356X:/#
root@RK356X:/# modetest -M rockchip -s 167@115:720x1280
setting mode 720x1280-59.61Hz on connectors 167, crtc 115
```

其中 167 是 DSI-1 的 id，115 是 CRTC 的 id（也就是 VP1 的 id），如果客户用的是 1080p MIPI 屏，则将分辨率改成 1080x1920 即可。显示效果如下：

执行如下命令在 LVDS 屏上输出彩条：

```
modetest -M rockchip -s 154@131:1280x800
```

```
root@RK356X:/#
root@RK356X:/# modetest -M rockchip -s 154@131:1280x800
setting mode 1280x800-54.85Hz on connectors 154, crtc 131
```

其中 154 是 LVDS-1 的 id，131 是 CRTC 的 id（也就是 VP2 的 id）。

第四章 参考资料汇总

更加详细的内容请参考资料以下文档：

<SDK>/docs/Common/DISPLAY/Rockchip_Developer_Guide_DRM_Display_Driver_CN.pdf
<SDK>/docs/RK356X/Datasheet/Rockchip_RK3568_Datasheet_V1.1-20210305.pdf
<SDK>/docs/Linux/Graphics/Rockchip_Developer_Guide_Linux_Graphics_CN.pdf
<SDK>/docs/Common/DISPLAY/Rockchip_Developer_Guide_HDMI_CN.pdf
<SDK>/docs/Common/DISPLAY/Rockchip_DRM_Display_Driver_Development_Guide_V1.0.pdf
<SDK>/docs/Common/DISPLAY/Rockchip_DRM_Panel_Porting_Guide_V1.6_20190228.pdf
<SDK>/docs/Common/DISPLAY/Rockchip_VOP2_Plane_Assign.pdf
<SDK>/docs/Common/CLK/Rockchip_Developer_Guide_Linux4.4_4.19_Clock_CN.pdf
<SDK>/docs/Common/CLK/Rockchip_Develop_Guide_Pll_Ssmod_Clock_CN.pdf