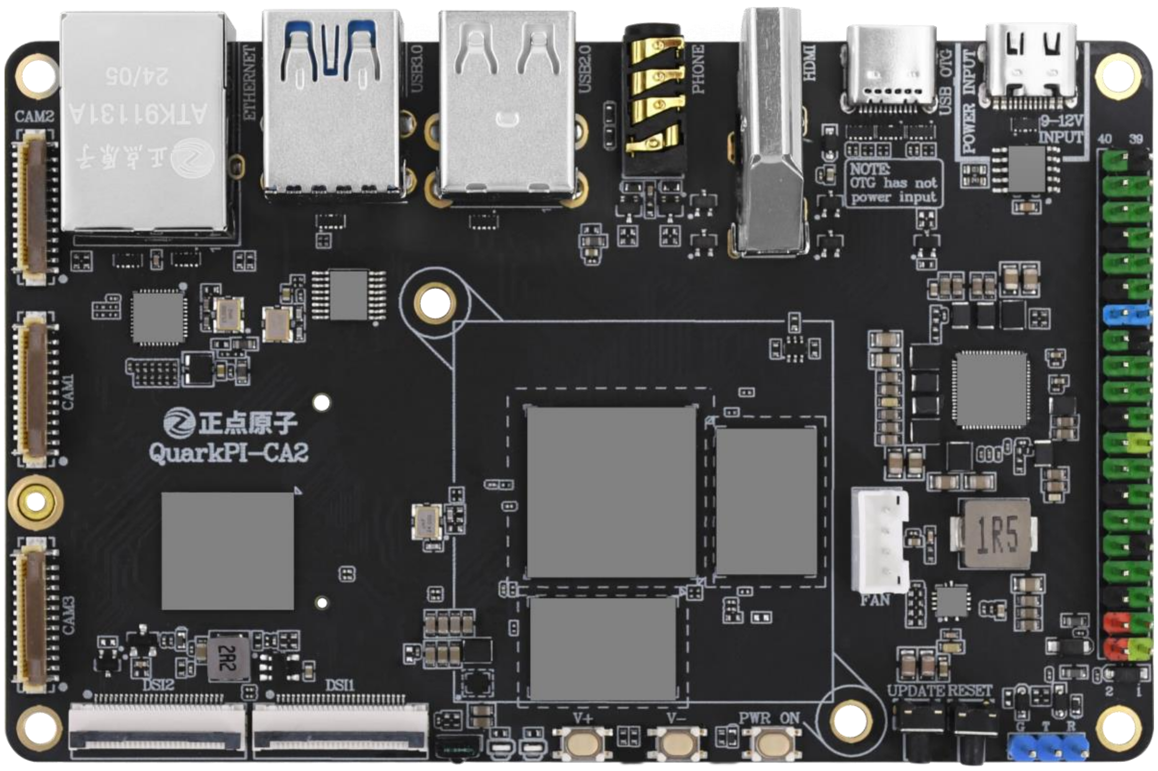
QuarkPi-CA2 Multi

Display Development manual

V1.0



1. Shopping:

TMALL: https://zhengdianyuanzi.tmall.com

TAOBAO: https://openedv.taobao.com

2. Download

Address: http://www.openedv.com/docs/index.html

3. FAE

Website	:	www.alientek.com
Forum	:	http://www.openedv.com/forum.php
Videos	:	www.yuanzige.com
Fax	:	+86 - 20 - 36773971
Phone	:	+86 - 20 - 38271790

Disclaimer

The product specifications and instructions mentioned in this document are for reference only and subject to update without prior notice; Unless otherwise agreed, this document is intended as a product guide only, and none of the representations made herein constitutes a warranty of any kind. The copyright of this document belongs to Guangzhou Xingyi Electronic Technology Co., LTD. Without the written permission of the company, any unit or individual shall not be used for profit-making purposes in any way of dissemination.

In order to get the latest version of product information, please regularly visit the download center or contact the customer service of Taobao ALIENTEK flagship store. Thank you for your tolerance and support.

Revision History:

Version	Version Update Notes	Responsible person	Proofreading	Date
V1.0	release officially	ALIENTEK	ALIENTEK	2025.04.07

Catalogue

Brief

This document mainly records the content related to screen display on the QuarkPi-CA2 card computer platform, including the following sections:

1. Overview of the RK3588S display subsystem;
2. Introduction to the display interface of the QuarkPi-CA2 platform;
3. Introduction to multi-screen display in Android;
4. Introduction to multi-screen display in Linux buildroot;

Chapter 1. Overview of the RK3588 Display Subsystem

The display subsystem refers to the collective term for the software and hardware systems related to display output on the Rockchip platform. It includes VOP (Video Output Processor, equivalent to the LCDC in other SoCs, which is the LCD controller) and the Display Interface (such as RGB, LVDS, MIPI DSI, EDP, HDMI, etc. for display interfaces) as well as the corresponding software drivers.

Currently, there are two VOP architectures on the Rockchip platform - VOP 1.0 and VOP 2.0. The RK3588 platform uses the VOP 2.0 architecture.

1.1 Hardware framework of the display system

For the VOP 2.0 display architecture, the hardware block diagram of the entire display system is as follows:

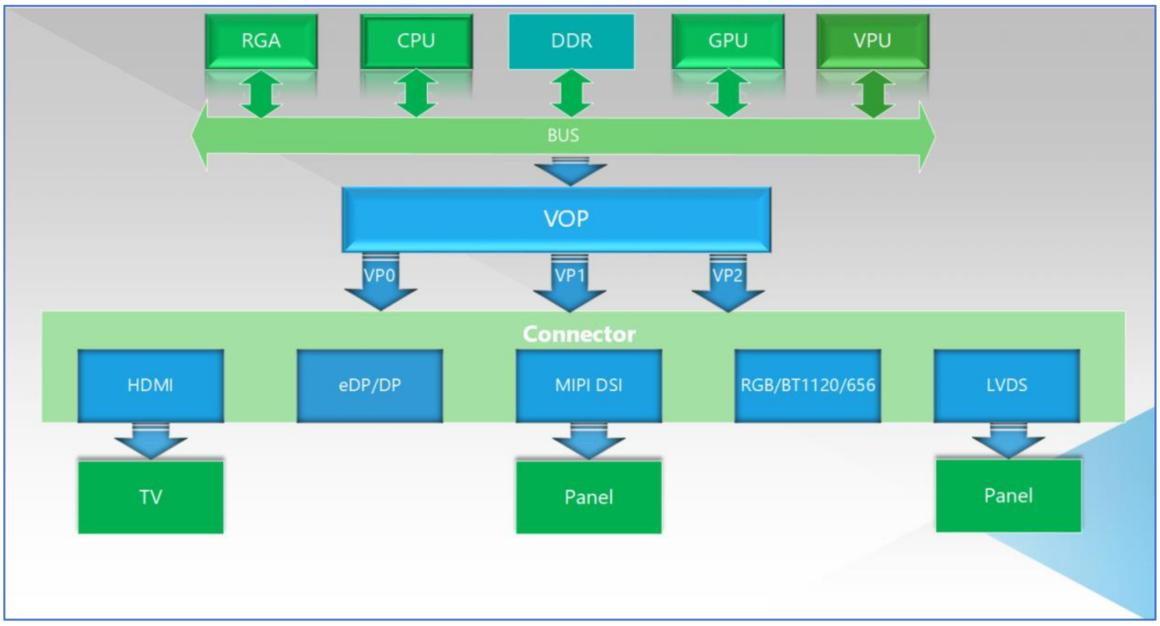


Figure 1.1-1 VOP 2.0 Display Subsystem Architecture

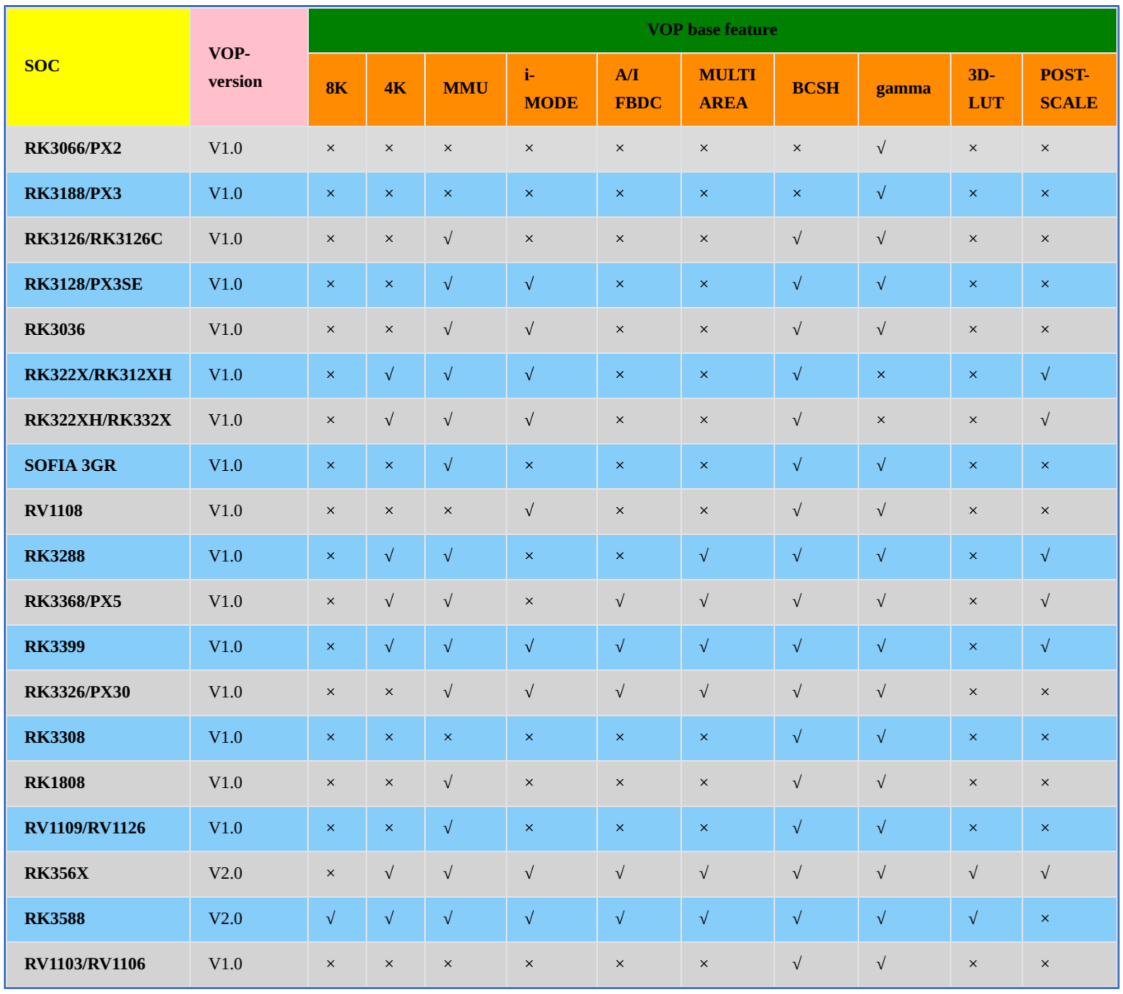
From the block diagram, it can be seen that at the very end of the entire display path, the display graphics acceleration module composed of RGA, GPU and VPU is formed. These are hardware IPs specifically designed for image processing optimization, capable of efficiently generating and further processing images (for example, GPU provides image rendering functions through OpenGL, RGA can perform 2D processing such as scaling, rotation, and synthesis on image data, and VPU can efficiently perform video decoding). Thus, the burden on the CPU is reduced.

After being processed by these image acceleration modules, the data will be stored in DDR and then read by VOP. Based on the application requirements, it will undergo Alpha overlay, color space conversion, gamma correction, HDR conversion, etc. and then sent to the corresponding display interface module (HDMI/MIPI/RGB/LVDS/EDP). These interface modules will convert the received data into data streams that comply with their respective protocols and send them to the display or LCD screen, ultimately presenting the image to the user.

Currently, there are two VOP architectures on the Rockchip platform - VOP 1.0 and VOP 2.0 (the VOP 2.0 architecture is used on the RK3588 platform). Their main difference lies in the way they support multiple displays. VOP 2.0 adopts a unified display architecture, meaning that only one VOP exists on the entire SoC, but a multi-channel independent Video Port (hereinafter referred to as VP) output port is designed at the back end of VOP. These VP can work independently and output independent display timing. For example, the RK3588 platform has four VP, which can achieve quad-screen heterogeneous display.

1.2 Display Characteristics

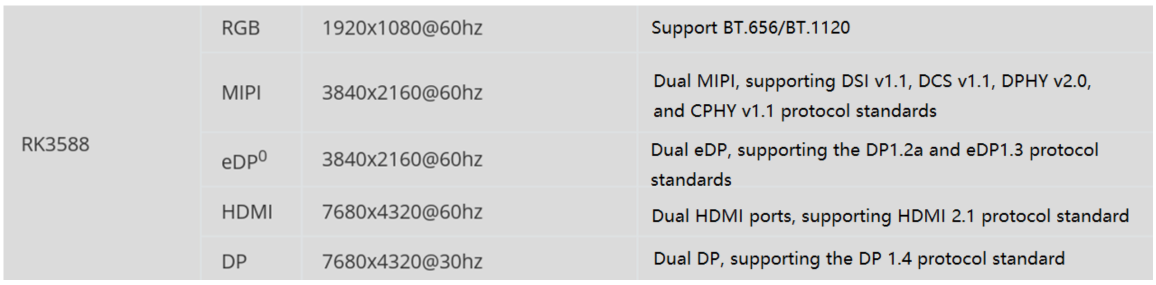
The basic display features of Rockchip's various platforms are shown in the following figure:



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	×	×	×	×	×	×	√	√	×	×

Figure 1.2-1 Basic features of VOP on each platform

The maximum output resolution and protocol standards of each display interface of RK3588 are as shown in the following figure:



RK3588	RGB	1920x1080@60hz	Support BT.656/BT.1120
	MIPI	3840x2160@60hz	Dual MIPI, supporting DSI v1.1, DCS v1.1, DPHY v2.0, and CPHY v1.1 protocol standards
	eDP^0	3840x2160@60hz	Dual eDP, supporting the DP1.2a and eDP1.3 protocol standards
	HDMI	7680x4320@60hz	Dual HDMI ports, supporting HDMI 2.1 protocol standard
	DP	7680x4320@30hz	Dual DP, supporting the DP 1.4 protocol standard

Figure 1.2-2 The maximum output resolution and protocol standards of each display interface

The RK3588 platform supports multiple display interfaces, including 1 RGB interface, 2 MIPI DSI interfaces (dsi0, dsi1), 2 eDP interfaces (edp0, edp1), 2 HDMI interfaces (hdmi0, hdmi1), and 2 DP interfaces (dp0, dp1). It should be noted that the PHYs of eDP and HDMI are combined. HDMI and eDP functions can only be selected from one of them. That is, in the same product, if hdmi0 is used, edp0 cannot be used; if hdmi1 is used, edp1 cannot be used.

The connection relationship between the RK3588 VP and each display interface is as shown in the following figure:

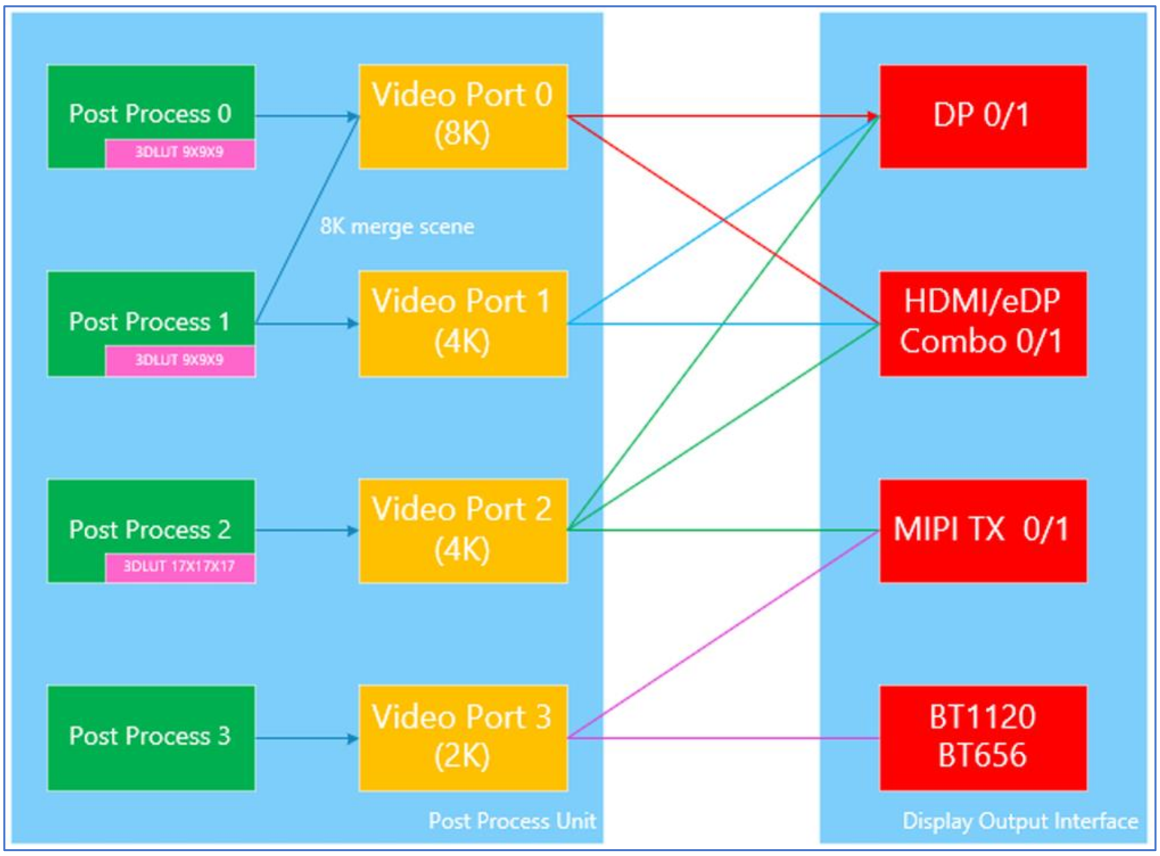


Figure 1.2-3 The relationship between RK3588 VP and various display interfaces

It should be noted that the HDMI and DP interfaces of the RK3588 platform support 8K output. However, in the 8K output mode, one display interface needs to occupy both VP0 and VP1 simultaneously. Therefore, if the product needs to support 8K display output, be sure not to connect other display interfaces on VP1.

The maximum resolution output by each display path (VP → display interface) is limited by the maximum resolution of the corresponding VP and display interface.

For more detailed information, please refer to the "Rockhip RK3588 Datasheet" - Display Interface and Video Output Processor section.

Chapter 2. Introduction to the Display Interface of the QuarkPi-CA2 Platform

This chapter describes the various display interfaces available on the ALIENTEK QuarkPi-CA2 card computer and the device tree configuration.

2.1 Display Interface

The QuarkPi-CA2 card computer of ALIENTEK provides multiple display interfaces for users, including 2 MIPI DSI interfaces, 1 HDMI display interface, and 1 full-function type-C interface (supporting DP display), which can support up to 4 screens simultaneously, offering users more choices.

The schematic diagrams of each display interface on the development board are as follows:

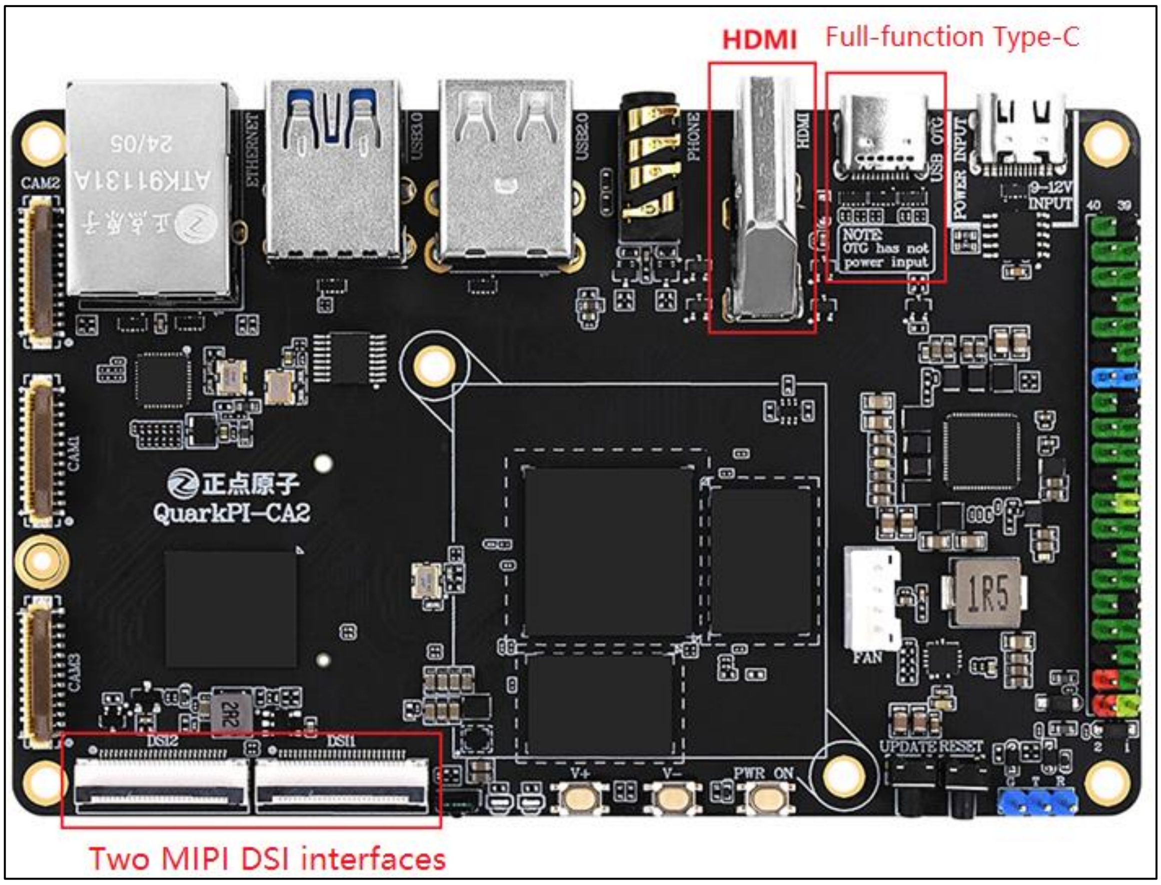


Figure 2.1-1 Display interface distribution diagram

One HDMI display interface (HDMI TX0): used for connecting HDMI display screens. It can support the maximum output of 7680x4320@60Hz;

Two MIPI DSI interfaces (DSI1, DSI2): used for connecting MIPI display screens. It supports the 5.5-inch 1080x1920 MIPI screen, 5.5-inch 720x1280 MIPI screen, and 10.1-inch 800x1280 MIPI screen sold by ALIENTEK Store;

One full-function type-c interface (TYPE-C): The full-function type-c interface supports USB 3.0 and DP display, and can be connected to DP display (using type-c to DP cable connection). It can support the maximum output of 7680x4320@30Hz;

2.2 Device Tree Configuration

The kernel device tree files for the Rockchip platform are all stored in the <Kernel_PATH>/arch/arm64/boot/dts/rockchip/ directory. For the QuarkPi-CA2 card computer platform, the corresponding kernel device tree file is:

```
Linux system：      arch/arm64/boot/dts/rockchip/rk3588s-atk-quarkpi-ca2.dts
Android system：    arch/arm64/boot/dts/rockchip/rk3588s-atk-quarkpi-ca2.dts
```

Kernel source code directory:

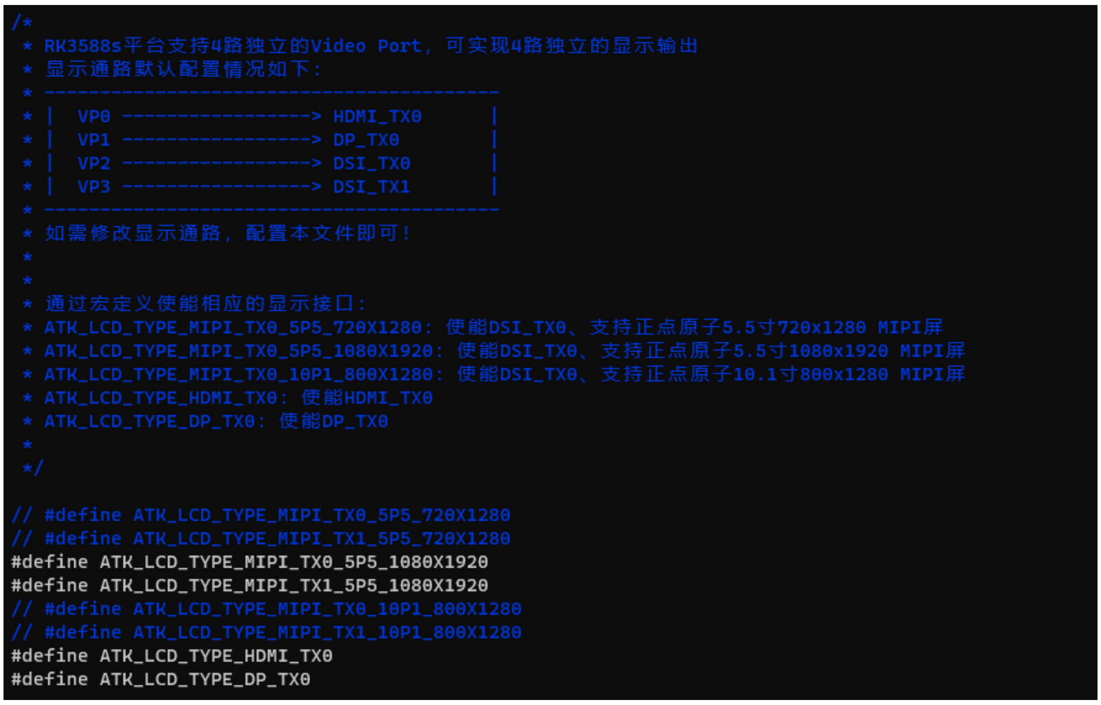
```
Linux SDK：         <SDK_PATH>/kernel/
Android SDK：       <SDK_PATH>/kernel-5.10/
```

Note: SDK_PATH refers to the root directory of the SDK.

The structure of these two device tree files is as follows:

```
# buildroot linux
rk3588s-atk-quarkpi-ca2.dts
        rk3588s-atk-quarkpi-ca2..dtsi
                rk3588s.dtsi
                rk3588s-evb.dtsi
                rk3588-rk806-single.dtsi
        rk3588s-atk-camera.dtsi
        rk3588s-atk-screen_choose.dtsi
        rk3588s-atk-lcds.dtsi
        rk3588-linux.dtsi

# android12/android13
rk3588s-atk-quarkpi-ca2.dts
        rk3588s-atk-quarkpi-ca2.dtsi
                rk3588s.dtsi
                rk3588s-evb.dtsi
                rk3588-rk806-single.dtsi
        rk3588s-atk-camera.dtsi
        rk3588s-atk-screen_choose.dtsi
        rk3588s-atk-lcds.dtsi
        rk3588-android.dtsi
```

For the screen display, we focus on the two device tree files: rk3588s-atk-lcds.dtsi and rk3588s-atk-screen_choose.dtsi. The rk3588s-atk-lcds.dtsi file contains the configuration information related to the display interface on the quarkpi-ca2 card computer; while the rk3588s-atk-screen_choose.dtsi file defines some macros for selecting which display interfaces on the development board we need to enable. The content is as follows:



```
/*
 * RK3588s平台支持4路独立的Video Port, 可实现4路独立的显示输出
 * 显示通路默认配置情况如下:
 * ------------------------------------------
 * |  VP0 -----------------> HDMI_TX0       |
 * |  VP1 -----------------> DP_TX0         |
 * |  VP2 -----------------> DSI_TX0        |
 * |  VP3 -----------------> DSI_TX1        |
 * ------------------------------------------
 * 如需修改显示通路, 配置本文件即可!
 *
 *
 * 通过宏定义使能相应的显示接口:
 * ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280: 使能DSI_TX0、支持正点原子5.5寸720x1280 MIPI屏
 * ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920: 使能DSI_TX0、支持正点原子5.5寸1080x1920 MIPI屏
 * ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280: 使能DSI_TX0、支持正点原子10.1寸800x1280 MIPI屏
 * ATK_LCD_TYPE_HDMI_TX0: 使能HDMI_TX0
 * ATK_LCD_TYPE_DP_TX0: 使能DP_TX0
 *
 */

// #define ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_720X1280
#define ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920
#define ATK_LCD_TYPE_MIPI_TX1_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280
// #define ATK_LCD_TYPE_MIPI_TX1_10P1_800X1280
#define ATK_LCD_TYPE_HDMI_TX0
#define ATK_LCD_TYPE_DP_TX0
```

Figure 2.2-1 The content of the file rk3588s-atk-screen_choose.dtsi

As mentioned earlier, the RK3588s platform has 4 independent VPs, enabling 4 independent display outputs. However, compared to RK3588, RK3588S has one less HDMI and DP output interface each, so there are only 4 output interfaces in total. The default display path configuration is as follows:

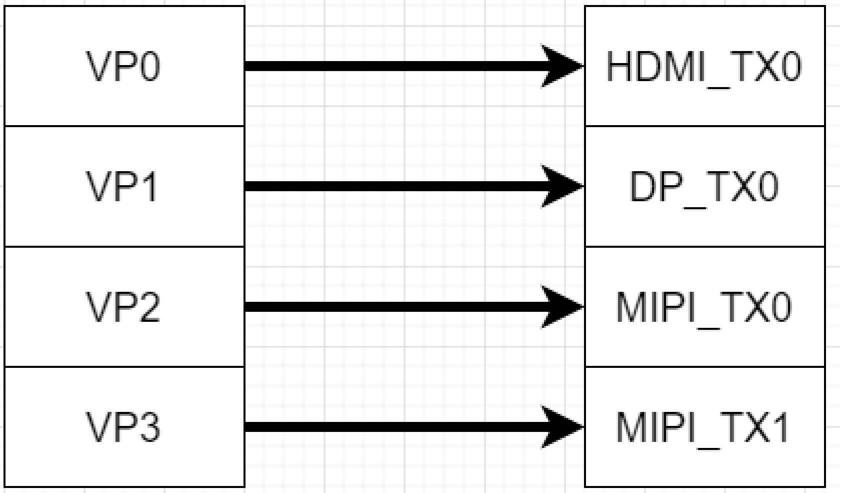
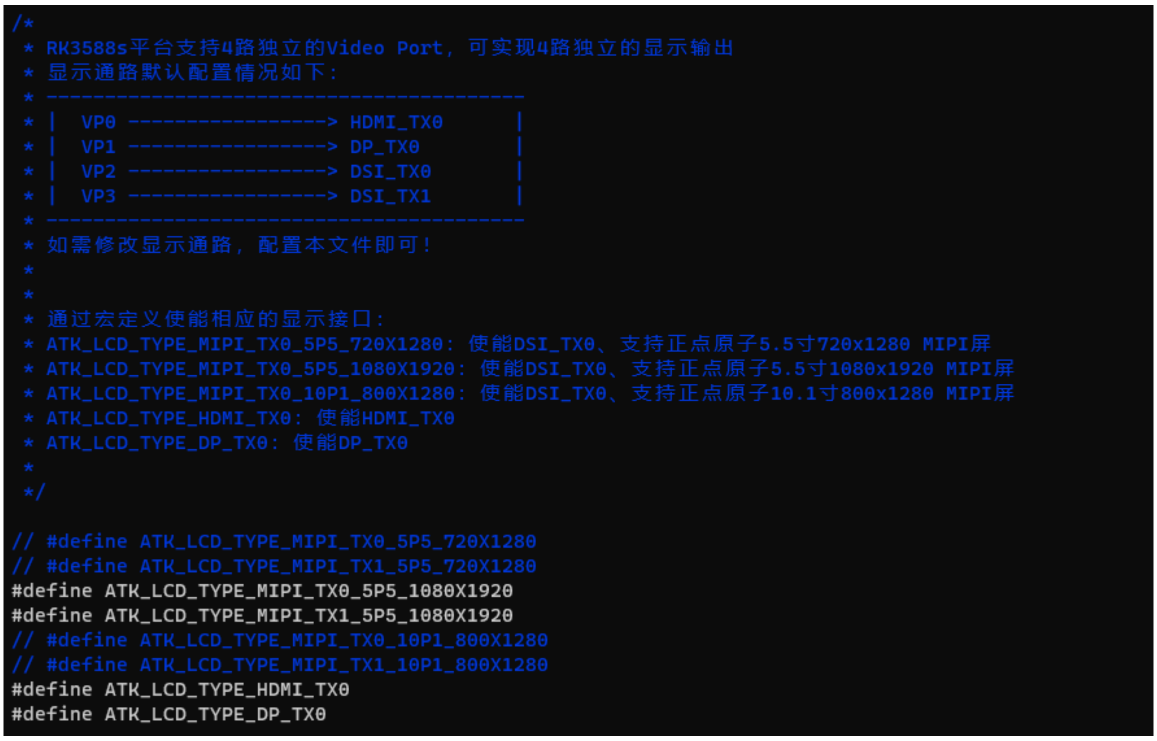


Figure 2.2-2 Default display path configuration status

Depending on different display modes, adjustments to the display paths may be necessary.

Users can define or comment out the corresponding ATK_LCD_TYPE_xxx macros to enable or disable the corresponding display interfaces.



```
/*
 * RK3588s平台支持4路独立的Video Port, 可实现4路独立的显示输出
 * 显示通路默认配置情况如下:
 * -----------------------------------------
 * |  VP0 -----------------> HDMI_TX0      |
 * |  VP1 -----------------> DP_TX0        |
 * |  VP2 -----------------> DSI_TX0       |
 * |  VP3 -----------------> DSI_TX1       |
 * -----------------------------------------
 * 如需修改显示通路, 配置本文件即可!
 *
 *
 * 通过宏定义使能相应的显示接口:
 * ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280: 使能DSI_TX0、支持正点原子5.5寸720x1280 MIPI屏
 * ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920: 使能DSI_TX0、支持正点原子5.5寸1080x1920 MIPI屏
 * ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280: 使能DSI_TX0、支持正点原子10.1寸800x1280 MIPI屏
 * ATK_LCD_TYPE_HDMI_TX0: 使能HDMI_TX0
 * ATK_LCD_TYPE_DP_TX0: 使能DP_TX0
 *
 */

// #define ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_720X1280
#define ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920
#define ATK_LCD_TYPE_MIPI_TX1_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280
// #define ATK_LCD_TYPE_MIPI_TX1_10P1_800X1280
#define ATK_LCD_TYPE_HDMI_TX0
#define ATK_LCD_TYPE_DP_TX0
```

Figure 2.2-3 Default display path configuration status

Depending on different display modes, adjustments to the display paths may be necessary.

Users can define or comment out the corresponding ATK_LCD_TYPE_xxx macros to enable or disable the corresponding display interfaces.

2.3 Display Interface Configuration Explanation

Next, a brief explanation of the content in the rk3588s-atk-lcds.dtsi file will be provided. This device tree file mainly contains the configuration information related to the display interfaces on the development board.

2.3.1 MIPI Screen Configuration Information

There are two MIPI screen interfaces on the development board, and there are also two screen configuration entries in the device tree.

MIPI DSI0:

```
/******************** Backlight ********************/
&backlight {
        pwms = <&pwm5 0 25000 0>;
        default-brightness-level = <255>;
        status = "disabled";
};
/******************** END ********************/
&disp_timings0 {
        mipi0_5p5_1080x1920_timing: timing0 {
                /*
```

```
		clock-frequency = <136000000>;
		hactive = <1080>;
		vactive = <1920>;
		hfront-porch = <45>;
		hsync-len = <45>;
		hback-porch = <5>;
		vfront-porch = <9>;
		vsync-len = <4>;
		vback-porch = <3>;
		hsync-active = <0>;
		vsync-active = <0>;
		de-active = <0>;
		pixelclk-active = <0>;
		*/

		clock-frequency = <119000000>;
		hactive = <1080>;
		vactive = <1920>;
		hfront-porch = <10>;
		hsync-len = <6>;
		hback-porch = <32>;
		vfront-porch = <20>;
		vsync-len = <6>;
		vback-porch = <10>;
		hsync-active = <0>;
		vsync-active = <0>;
		de-active = <0>;
		pixelclk-active = <0>;
	};

	mipi0_5p5_720x1280_timing: timing1 {
		clock-frequency = <65000000>;
		hactive = <720>;
		vactive = <1280>;
		hfront-porch = <48>;
		hsync-len = <8>;
		hback-porch = <52>;
		vfront-porch = <16>;
		vsync-len = <6>;
		vback-porch = <15>;
		hsync-active = <0>;
		vsync-active = <0>;
		de-active = <0>;
```

```
                pixelclk-active = <0>;
        };

        mipi0_10p1_800x1280_timing: timing2 {
                clock-frequency = <67000000>;
                hactive = <800>;
                vactive = <1280>;
                hfront-porch = <12>;
                hsync-len = <24>;
                hback-porch = <24>;
                vfront-porch = <7>;
                vsync-len = <2>;
                vback-porch = <9>;
                hsync-active = <0>;
                vsync-active = <0>;
                de-active = <0>;
                pixelclk-active = <0>;
        };
};
/******************** touch screen ********************/
&i2c4 {
        dsi0_touch: gt911@14 {
                compatible = "goodix,gt9xx";
                reg = <0x14>;
                pinctrl-names = "default";
                pinctrl-0 = <&touch0_gpio>;
                touch-gpio = <&gpio1 RK_PA0 IRQ_TYPE_LEVEL_LOW>;
                reset-gpio = <&gpio1 RK_PA1 GPIO_ACTIVE_HIGH>;
                max-x = <1080>;
                max-y = <1920>;
                tp-size = <911>;
                tp-supply = <&vcc5v0_sys>;
                wakeup-source;
                goodix-ts-name = "dsi0_ts_gt9xx";
                status = "disabled";
        };
};
/******************** END ********************/
/******************** MIPI DSI TX0 ********************/
&dsi0 {
        status = "disabled";
};
```

```
&dsi0_in_vp2 {
        status = "disabled";
};

&dsi0_in_vp3 {
        status = "disabled";
};

&route_dsi0 {
        status = "disabled";
};



&dsi0_panel {
        power-supply = <&vcc5v0_sys>;
        reset-gpios = <&gpio0 RK_PB2 GPIO_ACTIVE_LOW>;
        pinctrl-names = "default";
        pinctrl-0 = <&lcd0_rst_gpio>;
        reset-delay-ms = <100>;
        init-delay-ms = <80>;

        panel-exit-sequence = [
                05 32 01 28
                05 C8 01 10
        ];
        status = "disabled";
};

/******************** END ********************/
#if defined(ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280) ||
defined(ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920) ||
defined(ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280)
&dsi0 {
        status = "okay";
};

&dsi0_in_vp2 {
        status = "okay";
};

&route_dsi0 {
        status = "okay";
```

```
		connect = <&vp2_out_dsi0>;
	};

	&dsi0_panel {
		status = "okay";
	};

	&dsi0_touch {
		status = "okay";
	};

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

	#if defined(ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280)
	&disp_timings0 {
		native-mode = <&mipi0_5p5_720x1280_timing>;
	};

	&dsi0_panel {
		panel-init-sequence = [
			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_TX0_5P5_1080X1920)
&disp_timings0 {
        native-mode = <&mipi0_5p5_1080x1920_timing>;
};

&dsi0_panel {
        panel-init-sequence = [
                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 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
            ];
    };

    #elif defined(ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280)
    &disp_timings0 {
            native-mode = <&mipi0_10p1_800x1280_timing>;
    };

    &dsi0_panel {
            panel-init-sequence = [
                    39 05 04 FF 98 81 03
                    05 05 02 01 00
                    05 05 02 02 00
                    05 05 02 03 53
                    05 05 02 04 D3
                    05 05 02 05 00
                    05 05 02 06 0D
                    05 05 02 07 08
                    05 05 02 08 00
                    05 05 02 09 00
                    05 05 02 0a 00
                    05 05 02 0b 00
                    05 05 02 0c 00
                    05 05 02 0d 00
                    05 05 02 0e 00
                    05 05 02 0f 28
                    05 05 02 10 28
                    05 05 02 11 00
                    05 05 02 12 00
                    05 05 02 13 00
                    05 05 02 14 00
                    05 05 02 15 00
                    05 05 02 16 00
                    05 05 02 17 00
                    05 05 02 18 00
                    05 05 02 19 00
                    05 05 02 1a 00
                    05 05 02 1b 00
```

```
05 05 02 1c 00
05 05 02 1d 00
05 05 02 1e 40
05 05 02 1f 80
05 05 02 20 06
05 05 02 21 01
05 05 02 22 00
05 05 02 23 00
05 05 02 24 00
05 05 02 25 00
05 05 02 26 00
05 05 02 27 00
05 05 02 28 33
05 05 02 29 33
05 05 02 2a 00
05 05 02 2b 00
05 05 02 2c 00
05 05 02 2d 00
05 05 02 2e 00
05 05 02 2f 00
05 05 02 30 00
05 05 02 31 00
05 05 02 32 00
05 05 02 33 00
05 05 02 34 03
05 05 02 35 00
05 05 02 36 00
05 05 02 37 00
05 05 02 38 96
05 05 02 39 00
05 05 02 3a 00
05 05 02 3b 00
05 05 02 3c 00
05 05 02 3d 00
05 05 02 3e 00
05 05 02 3f 00
05 05 02 40 00
05 05 02 41 00
05 05 02 42 00
05 05 02 43 00
05 05 02 44 00
05 05 02 50 00
05 05 02 51 23
```

```
05 05 02 52 45
05 05 02 53 67
05 05 02 54 89
05 05 02 55 AB
05 05 02 56 01
05 05 02 57 23
05 05 02 58 45
05 05 02 59 67
05 05 02 5a 89
05 05 02 5b AB
05 05 02 5c CD
05 05 02 5d EF
05 05 02 5e 00
05 05 02 5f 08
05 05 02 60 08
05 05 02 61 06
05 05 02 62 06
05 05 02 63 01
05 05 02 64 01
05 05 02 65 00
05 05 02 66 00
05 05 02 67 02
05 05 02 68 15
05 05 02 69 15
05 05 02 6a 14
05 05 02 6b 14
05 05 02 6c 0D
05 05 02 6d 0D
05 05 02 6e 0C
05 05 02 6f 0C
05 05 02 70 0F
05 05 02 71 0F
05 05 02 72 0E
05 05 02 73 0E
05 05 02 74 02
05 05 02 75 08
05 05 02 76 08
05 05 02 77 06
05 05 02 78 06
05 05 02 79 01
05 05 02 7a 01
05 05 02 7b 00
05 05 02 7c 00
```

```
05 05 02 7d 02
05 05 02 7e 15
05 05 02 7f 15
05 05 02 80 14
05 05 02 81 14
05 05 02 82 0D
05 05 02 83 0D
05 05 02 84 0C
05 05 02 85 0C
05 05 02 86 0F
05 05 02 87 0F
05 05 02 88 0E
05 05 02 89 0E
05 05 02 8A 02
39 05 04 FF 98 81 04
05 05 02 6E 2B
05 05 02 6F 37
05 05 02 3A 24
05 05 02 8D 1A
05 05 02 87 BA
05 05 02 B2 D1
05 05 02 88 0B
05 05 02 38 01
05 05 02 39 00
05 05 02 B5 02
05 05 02 31 25
05 05 02 3B 98
39 05 04 FF 98 81 01
05 05 02 22 0A
05 05 02 31 00
05 05 02 53 3D
05 05 02 55 3D
05 05 02 50 B5
05 05 02 51 AD
05 05 02 60 06
05 05 02 62 20
05 05 02 A0 00
05 05 02 A1 21
05 05 02 A2 35
05 05 02 A3 19
05 05 02 A4 1E
05 05 02 A5 33
05 05 02 A6 27
```

```
                05 05 02 A7 26
                05 05 02 A8 AF
                05 05 02 A9 1B
                05 05 02 AA 27
                05 05 02 AB 8D
                05 05 02 AC 1A
                05 05 02 AD 1B
                05 05 02 AE 50
                05 05 02 AF 26
                05 05 02 B0 2B
                05 05 02 B1 54
                05 05 02 B2 5E
                05 05 02 B3 23
                05 05 02 C0 00
                05 05 02 C1 21
                05 05 02 C2 35
                05 05 02 C3 19
                05 05 02 C4 1E
                05 05 02 C5 33
                05 05 02 C6 27
                05 05 02 C7 26
                05 05 02 C8 AF
                05 05 02 C9 1B
                05 05 02 CA 27
                05 05 02 CB 8D
                05 05 02 CC 1A
                05 05 02 CD 1B
                05 05 02 CE 50
                05 05 02 CF 26
                05 05 02 D0 2B
                05 05 02 D1 54
                05 05 02 D2 5E
                05 05 02 D3 23
                39 05 04 FF 98 81 00
                05 05 02 11 00
                05 78 02 29 00
                05 78 02 35 00
        ];
};
#endif
```

MIPI DSI1

```
05 05 02 24 00
05 05 02 25 00
05 05 02 26 00
05 05 02 27 00
05 05 02 28 33
05 05 02 29 33
05 05 02 2a 00
05 05 02 2b 00
05 05 02 2c 00
05 05 02 2d 00
05 05 02 2e 00
05 05 02 2f 00
05 05 02 30 00
05 05 02 31 00
05 05 02 32 00
05 05 02 33 00
05 05 02 34 03
05 05 02 35 00
05 05 02 36 00
05 05 02 37 00
05 05 02 38 96
05 05 02 39 00
05 05 02 3a 00
05 05 02 3b 00
05 05 02 3c 00
05 05 02 3d 00
05 05 02 3e 00
05 05 02 3f 00
05 05 02 40 00
05 05 02 41 00
05 05 02 42 00
05 05 02 43 00
05 05 02 44 00
05 05 02 50 00
05 05 02 51 23
05 05 02 52 45
05 05 02 53 67
05 05 02 54 89
05 05 02 55 AB
05 05 02 56 01
05 05 02 57 23
05 05 02 58 45
05 05 02 59 67
```

```
05 05 02 5a 89
05 05 02 5b AB
05 05 02 5c CD
05 05 02 5d EF
05 05 02 5e 00
05 05 02 5f 08
05 05 02 60 08
05 05 02 61 06
05 05 02 62 06
05 05 02 63 01
05 05 02 64 01
05 05 02 65 00
05 05 02 66 00
05 05 02 67 02
05 05 02 68 15
05 05 02 69 15
05 05 02 6a 14
05 05 02 6b 14
05 05 02 6c 0D
05 05 02 6d 0D
05 05 02 6e 0C
05 05 02 6f 0C
05 05 02 70 0F
05 05 02 71 0F
05 05 02 72 0E
05 05 02 73 0E
05 05 02 74 02
05 05 02 75 08
05 05 02 76 08
05 05 02 77 06
05 05 02 78 06
05 05 02 79 01
05 05 02 7a 01
05 05 02 7b 00
05 05 02 7c 00
05 05 02 7d 02
05 05 02 7e 15
05 05 02 7f 15
05 05 02 80 14
05 05 02 81 14
05 05 02 82 0D
05 05 02 83 0D
05 05 02 84 0C
```

```
05 05 02 85 0C
05 05 02 86 0F
05 05 02 87 0F
05 05 02 88 0E
05 05 02 89 0E
05 05 02 8A 02
39 05 04 FF 98 81 04
05 05 02 6E 2B
05 05 02 6F 37
05 05 02 3A 24
05 05 02 8D 1A
05 05 02 87 BA
05 05 02 B2 D1
05 05 02 88 0B
05 05 02 38 01
05 05 02 39 00
05 05 02 B5 02
05 05 02 31 25
05 05 02 3B 98
39 05 04 FF 98 81 01
05 05 02 22 0A
05 05 02 31 00
05 05 02 53 3D
05 05 02 55 3D
05 05 02 50 B5
05 05 02 51 AD
05 05 02 60 06
05 05 02 62 20
05 05 02 A0 00
05 05 02 A1 21
05 05 02 A2 35
05 05 02 A3 19
05 05 02 A4 1E
05 05 02 A5 33
05 05 02 A6 27
05 05 02 A7 26
05 05 02 A8 AF
05 05 02 A9 1B
05 05 02 AA 27
05 05 02 AB 8D
05 05 02 AC 1A
05 05 02 AD 1B
05 05 02 AE 50
```

```
                05 05 02 AF 26
                05 05 02 B0 2B
                05 05 02 B1 54
                05 05 02 B2 5E
                05 05 02 B3 23
                05 05 02 C0 00
                05 05 02 C1 21
                05 05 02 C2 35
                05 05 02 C3 19
                05 05 02 C4 1E
                05 05 02 C5 33
                05 05 02 C6 27
                05 05 02 C7 26
                05 05 02 C8 AF
                05 05 02 C9 1B
                05 05 02 CA 27
                05 05 02 CB 8D
                05 05 02 CC 1A
                05 05 02 CD 1B
                05 05 02 CE 50
                05 05 02 CF 26
                05 05 02 D0 2B
                05 05 02 D1 54
                05 05 02 D2 5E
                05 05 02 D3 23
                39 05 04 FF 98 81 00
                05 05 02 11 00
                05 78 02 29 00
                05 78 02 35 00
        ];
};
#endif
```

2.3.2 HDMI Screen Configuration Information

There is one HDMI output interface on the card computer.

HDMI TX0：

```
/******************** HDMI TX0 ********************/
//<!- HDMI TX0 and EDP TX0 cannot be enabled simultaneously.->
&hdmi0 {
        enable-gpios = <&gpio4 RK_PB6 GPIO_ACTIVE_HIGH>;
        status = "disabled";
};
```

```
&hdmi0_in_vp0 {
        status = "disabled";
};

&hdmi0_in_vp1 {
        status = "disabled";
};

&hdmi0_in_vp2 {
        status = "disabled";
};

&route_hdmi0 {
        status = "disabled";
};

&hdptxphy_hdmi0 {
        status = "disabled";
};

&hdmi0_sound { //i2s5_8ch
        status = "disabled";
};
/******************** END ********************/
#if defined(ATK_LCD_TYPE_HDMI_TX0)
&hdmi0 {
        status = "okay";
};

&hdmi0_in_vp0 {
        status = "okay";
};

&route_hdmi0 {
        status = "okay";
        connect = <&vp0_out_hdmi0>;
};

&hdptxphy_hdmi0 {
        status = "okay";
};

&hdmi0_sound {
        status = "okay";
};
#endif
```

2.3.3 DP Display Port Configuration Information

There is one DP display port on the card computer.

DP TX0：

```
/******************** DP TX0 ********************/
&dp0 {
        status = "disabled";
};

&dp0_in_vp0 {
        status = "disabled";
};

&dp0_in_vp1 {
        status = "disabled";
};

&dp0_in_vp2 {
        status = "disabled";
};

&route_dp0 {
        status = "disabled";
};

//RK3588S 支持一个 USB3.0/DP1.4 Combo PHY
&usbdp_phy0_dp { // The child node of usbdp_phy0
        status = "disabled";
};

&dp0_sound { //spdif_tx2
        status = "disabled";
};
/******************** END ********************/
#if defined(ATK_LCD_TYPE_DP_TX0)
&dp0 {
        status = "okay";
};
```

```
&dp0_in_vp1 {
        status = "okay";
};

&route_dp0 {
        status = "okay";
        connect = <&vp1_out_dp0>;
};

&usbdp_phy0_dp {
        status = "okay";
};

&dp0_sound {
        status = "okay";
};
#endif
```

Chapter 3. Introduction to Android Multi-screen Display

This chapter introduces the related content of multi-screen display under the Android system, including the concepts of the main screen and secondary screen, screen display direction rotation, touch direction rotation, multi-screen simultaneous display, and multi-screen different display with different touch operations, etc.

3.1 Concept of Main Screen and Secondary Screen

In the Android system, when there are multiple logical screens (corresponding to actual physical screens), there are main screens and secondary screens. The main screen usually has only one, while the secondary screen can have multiple. The logical screens in the Android system are divided into main screen, secondary screen, and virtual screen (not considering for now), corresponding to the actual physical screens; the attributes and display paths of the logical screens in the system may vary according to display requirements, such as multi-screen simultaneous display and multi-screen different display.

3.1.1 Only one logical screen

If the system has only one logical screen, that is, the logical main screen, then the display path of the system is as shown in the following figure:

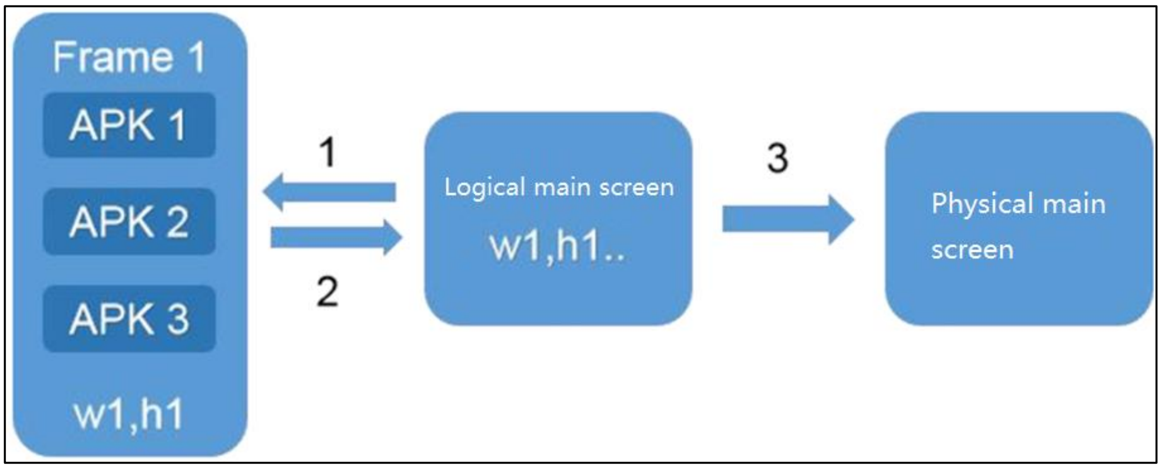


Figure 3.1-1 Single-screen display

1. The APK acquisition logic obtains the attribute information (width/height/fps) of the main screen and uses it as the rendering layout for the APK;

2. Multiple APKs submit display requests to the logic main screen, and the main screen mixes them based on the hierarchical structure;

3. The logic main screen sends the mixed effect to the actual physical main screen. By default, the attribute information of the logic main screen is consistent with that of the physical main screen;

Note: Since the APK is rendered and drawn based on the attribute information of the main screen, when there is only one logic main screen, the display effect is normal and there are no stretching or other issues.

3.1.2 Multi-screen Display

Multi-screen display, as the name suggests, means displaying the same content on multiple screens. For the RK3588S platform, it supports dual-screen display (one main screen and one secondary screen), tri-screen display (one main screen and two secondary screens), and quad-screen display (one main screen and three secondary screens). In the case of multi-screen display, it is the logical secondary screen that transmits the display content of the logical main screen to the physical secondary screen. The display path is as shown in the following figure:

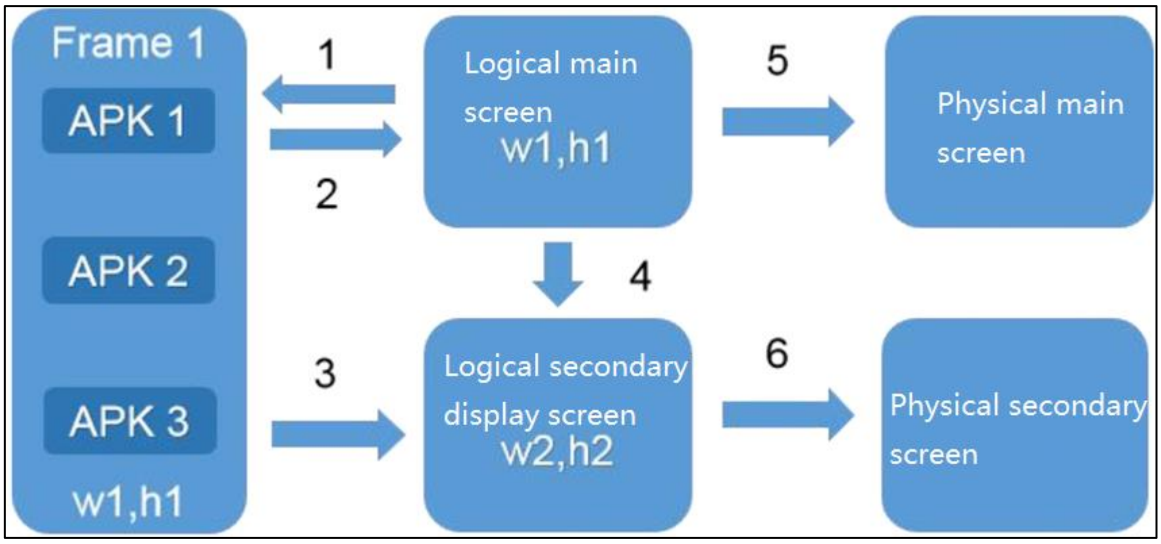

Frame 1
APK 1
APK 2
APK 3
w1,h1
1
2
3
Logical main screen w1,h1
4
Logical secondary display screen w2,h2
5
6
Physical main screen
Physical secondary screen

Figure 3.1-2 Multi-screen display

1. The APK obtains the attribute information (width/height/fps) of the main screen and uses it as the rendering layout for the APK;

2. Multiple APKs submit display requests to the logical main screen, which mixes them based on the hierarchical structure;

3. Multiple APKs submit display requests to the logical secondary screen, which also mixes them based on the hierarchical structure;

4. These three steps can be equivalently regarded as the logical main screen submitting the display content to the logical secondary screen;

5. The logical main screen sends the mixed result to the actual physical main screen. By default, the attribute information of the logical main screen is consistent with that of the physical main screen;

6. The logical secondary screen sends the mixed result to the actual physical secondary screen. By default, the attribute information of the logical secondary screen is consistent with that of the physical secondary screen.

In the multi-screen display mode, the logical secondary screen may have black borders or be stretched. The specific display scenarios are as follows:

- The pixel width and height ratios of the main and secondary screens are the same, that is, w1/h1 = w2/h2:

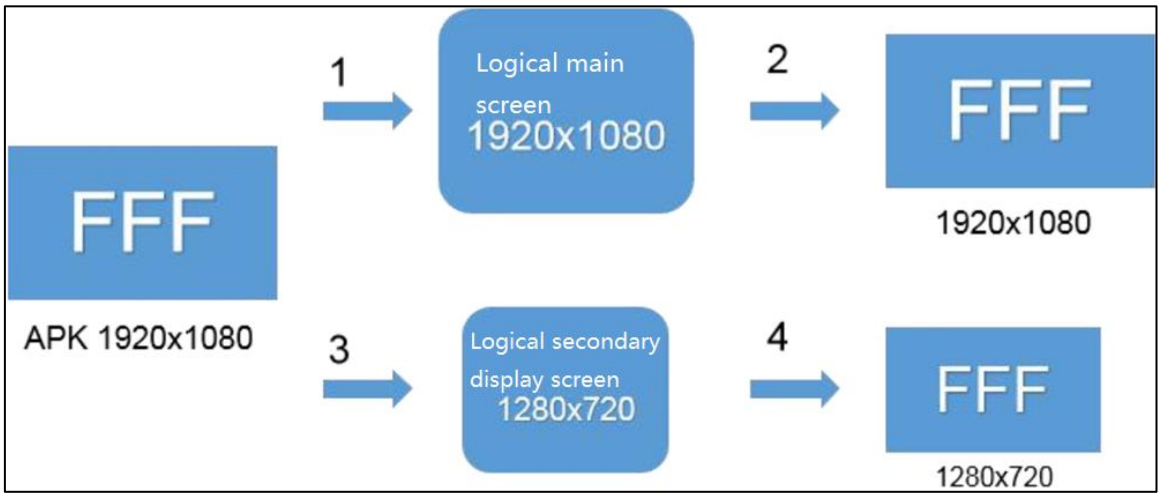

FFF
APK 1920x1080
1
Logical main screen 1920x1080
2
FFF
1920x1080
3
Logical secondary display screen 1280x720
4
FFF
1280x720

Figure 3.1-3 The pixel aspect ratios of the main and secondary screens are the same.

Note: If the aspect ratios of the logical main and secondary screens are the same, the image will be displayed full screen and there will be no stretching or black borders.

- If the pixel aspect ratios of the main and secondary screens are different, that is, w1/h1 != w2/h2:

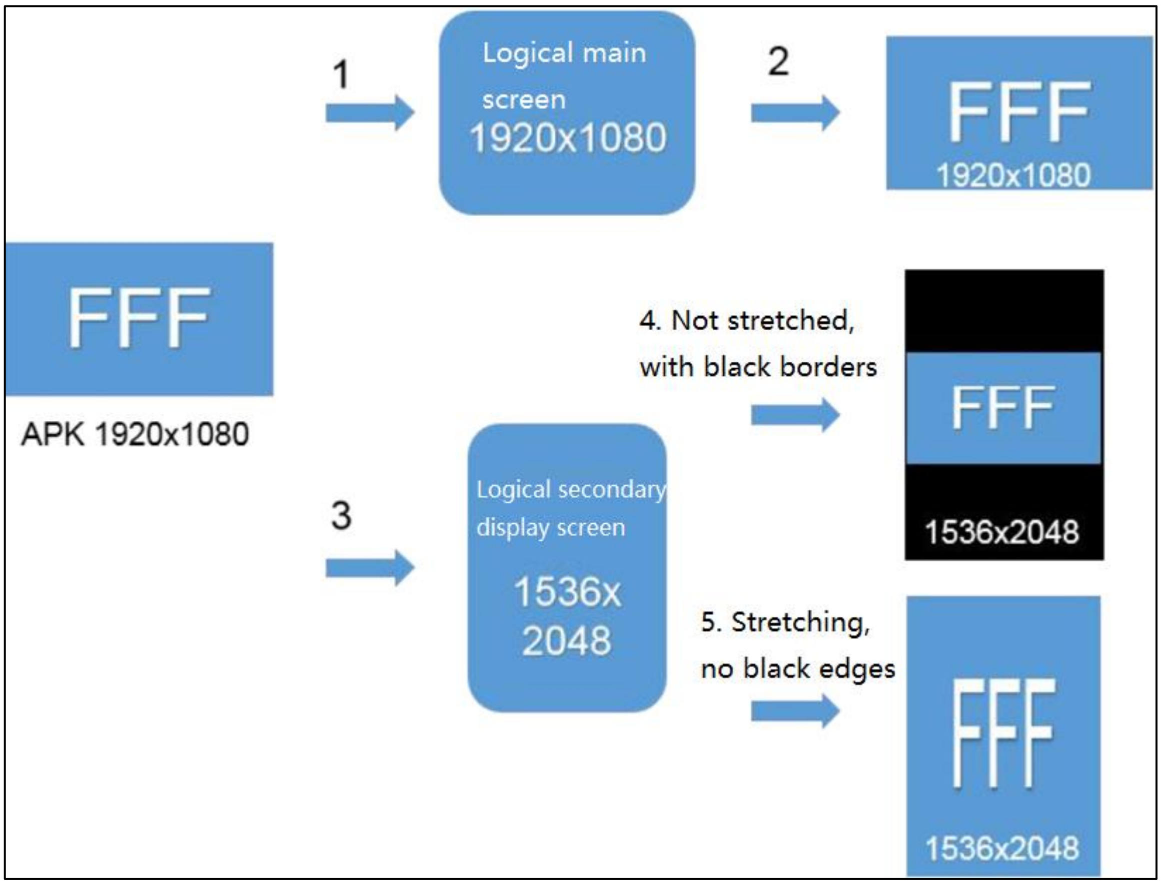

FFF
APK 1920x1080
1
Logical main screen 1920x1080
2
FFF
1920x1080
3
Logical secondary display screen 1536x 2048
4. Not stretched, with black borders
FFF
1536x2048
5. Stretching, no black edges
FFF
1536x2048

Figure 3.1-4 The pixel aspect ratios of the main and secondary screens are different.

1. The APK obtains the attribute information (width/height/refresh rate) of the main screen and uses it as the rendering layout for the APK;

2. If the image rendering layout is the same as the screen resolution, the image will be displayed normally without stretching or black borders;

3. If the image rendering layout is 1920x1080 and the screen resolution is 1536x2048, and the aspect ratios are different, then only two display modes 4 and 5 are available;

4. The image is displayed centered, without stretching, but there are black borders;

5. The image is tiled and stretched, but can be displayed full screen;

Note: In the case of multi-screen simultaneous display, if the aspect ratios of the logical main and secondary screens are different, only one of the display modes 4 and 5 can be selected.

3.1.3 Multi-screen Different Display

Multi-screen heterogeneous display, as the name suggests, means that different content is displayed on each of the multiple screens. For example, the RK3588S platform supports dual-screen heterogeneous display (one main screen and one secondary screen), three-screen heterogeneous display (one main screen and two secondary screens), and four-screen heterogeneous display (one main screen and three secondary screens). In the case of multi-screen heterogeneous display, the attribute information of the logical screen that needs to be displayed is independently drawn by the APK and independently displayed, which is different from multi-screen homogeneous display:

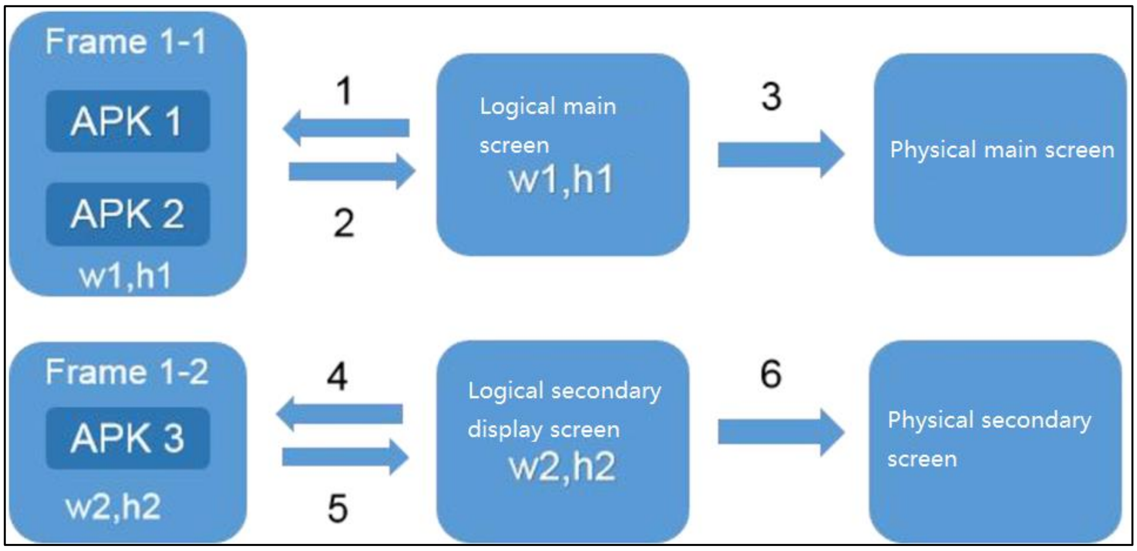


Figure 3.1-5 Multi-screen heterogeneous display

1. APK1 and APK2 obtain the attribute information (width/height/fps) of the logical main screen and use it as the rendering layout for APK1 and APK2;

2. APK1 and APK2 submit the display requests to the logical main screen, and the main screen performs blending based on the hierarchical structure;

3. The logical main screen sends the blended result to the actual physical main screen. By default, the attribute information of the logical main screen is consistent with that of the physical main screen;

4. APK3 obtains the attribute information (width/height/fps) of the logical secondary screen and uses it as the rendering layout for APK3;

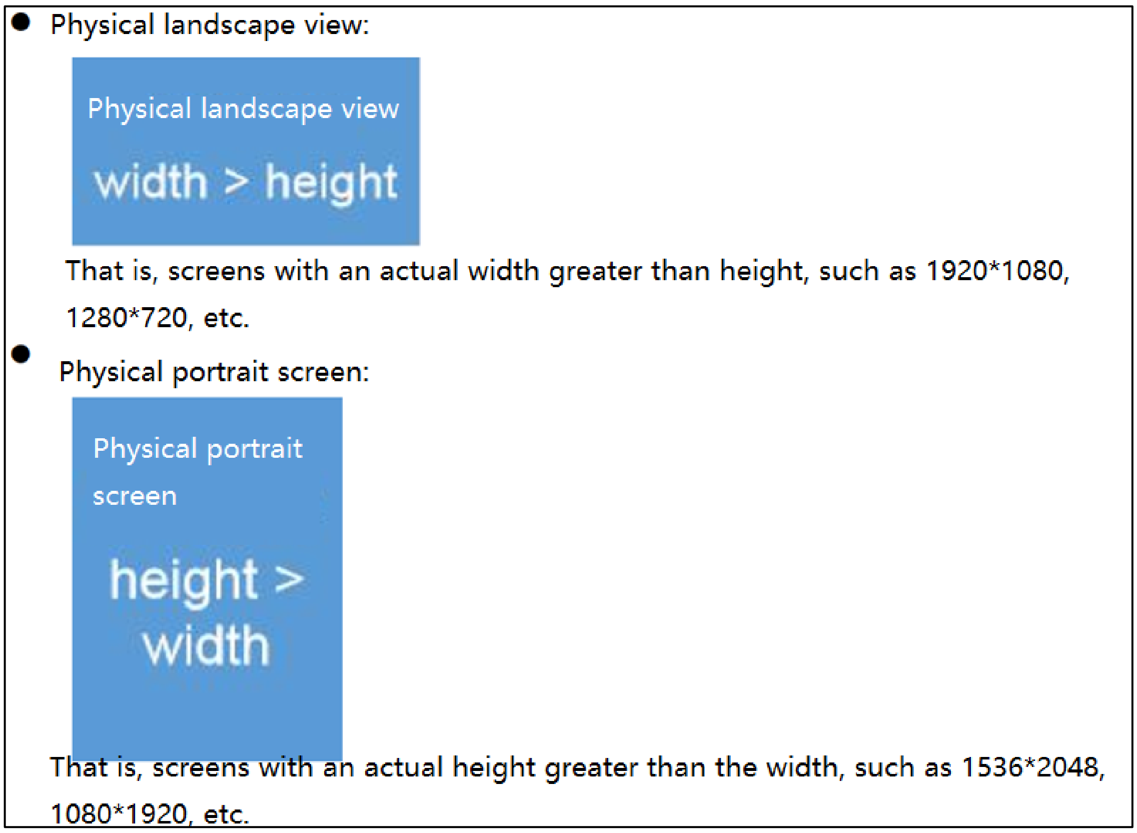
5. APK3 submits the display requests to the logical secondary screen, and the secondary screen performs blending based on the hierarchical structure;

6. The logical secondary screen sends the blended result to the actual physical secondary screen. By default, the attribute information of the logical secondary screen is consistent with that of the physical secondary screen;

Note: Due to the fact that the display content sent by APK is independently drawn, the rendering layout of APK always remains consistent with the attribute information of the physical screen that needs to be displayed. Therefore, there is no stretching or black border issue, and all screens display normally. The displayed content depends on each APK.

3.1.4 Physical Landscape Orientation and Physical Portrait Orientation

Distinguish between physical landscape orientation and physical portrait orientation, and introduce concepts such as landscape display on portrait orientation and portrait display on landscape orientation.



- Physical landscape view:

  That is, screens with an actual width greater than height, such as 1920*1080, 1280*720, etc.

- Physical portrait screen:

  That is, screens with an actual height greater than the width, such as 1536*2048, 1080*1920, etc.

Figure 3.1-6 Physical landscape mode and physical portrait mode

Introduction to landscape display and portrait display.

Take landscape display as an example, as shown below:

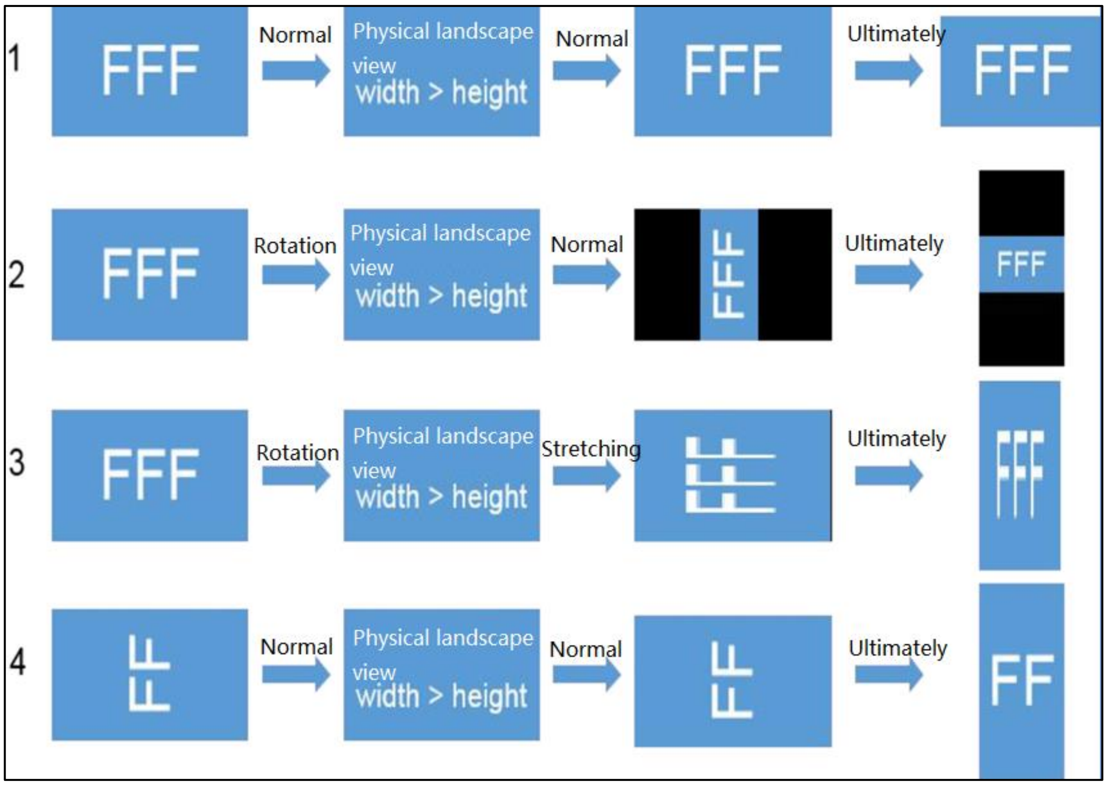

1
FFF
Normal
Physical landscape view width > height
Normal
FFF
Ultimately
FFF
2
FFF
Rotation
Physical landscape view width > height
Normal
FFF
Ultimately
FFF
3
FFF
Rotation
Physical landscape view width > height
Stretching
Ultimately
4
FF
Normal
Physical landscape view width > height
Normal
FF
Ultimately
FF

Figure 3.1-7 Portrait and Landscape Display Examples

1. Normal Portrait Display;
2. Normal Portrait + Back-end Rotation Display;
3. Normal Portrait + Back-end Rotation + Back-end Stretch Display;
4. Normal Portrait + APK End Rotation Display.

Note: 2/3/4 are all cases of portrait and landscape display, and all of them are achievable situations.

3.1.5 Introduction of Relevant Attributes

Next, we will introduce four attributes:

1.vendor.hwc.device.primary
2.vendor.hwc.device.extend
3. persist.sys.rotation.efull-n (n is a number, such as 1, 2, 3)
4. persist.sys.rotation.einit-n (n is a number, such as 1, 2, 3)

- vendor.hwc.device.primary

This attribute is used to set the main screen, for example:

```
vendor.hwc.device.primary=DSI            # Set the MIPI screen as the main screen
vendor.hwc.device.primary=HDMI-A         # Set the HDMI display as the main screen
vendor.hwc.device.primary=LVDS           # Set the LVDS screen as the main screen
vendor.hwc.device.primary=eDP            # Set the eDP display as the main screen
vendor.hwc.device.primary=DP             # Set the DP display as the main screen
```

Note: The main screen usually has only one, while the secondary screen can have multiple. If the system has only one logical screen, that screen is the logical main screen (corresponding to the physical main screen).

- vendor.hwc.device.extend

This attribute is used to set the secondary screen, for example:

```
vendor.hwc.device.extend=DSI            # Set the MIPI screen as the secondary display screen
vendor.hwc.device.extend=HDMI-A         # Set the HDMI display as the secondary screen
vendor.hwc.device.extend=LVDS           #Set the LVDS screen as the secondary display screen
vendor.hwc.device.extend=eDP            # Set the eDP screen as the secondary display screen
vendor.hwc.device.extend=DP             # Set the DP display as the secondary screen
vendor.hwc.device.extend=DSI,HDMI-A     # Set the MIPI screen and the HDMI display as secondary screens
```

- persist.sys.rotation.einit-n（n=1、2、3、….）

This attribute is used to set the display direction of the secondary screen (Secondary Screen 1, Secondary Screen 2, etc.). Its possible values are 0/1/2/3, which respectively represent rotating the display direction of the secondary screen (clockwise) by 0 degrees, 90 degrees, 180 degrees, and 270 degrees. As shown in the following figure:

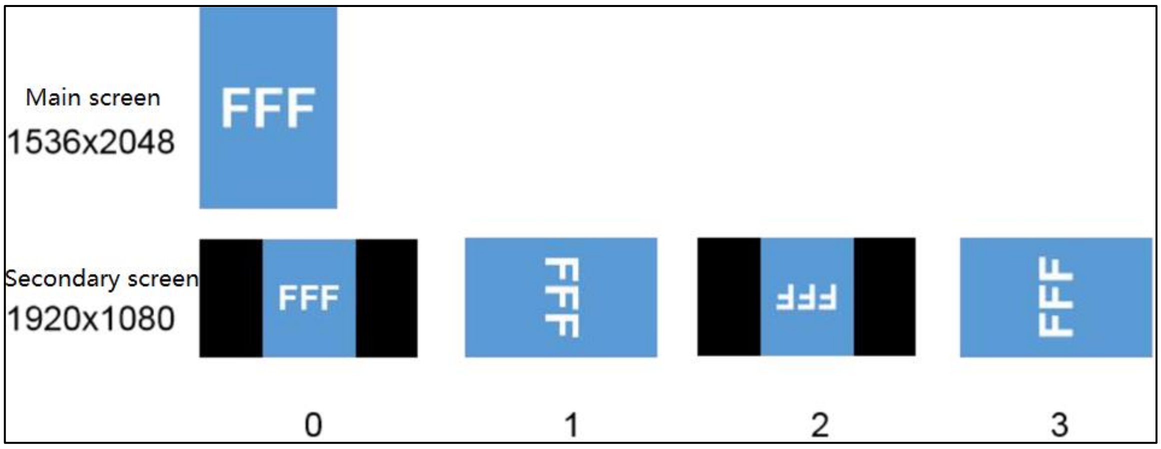
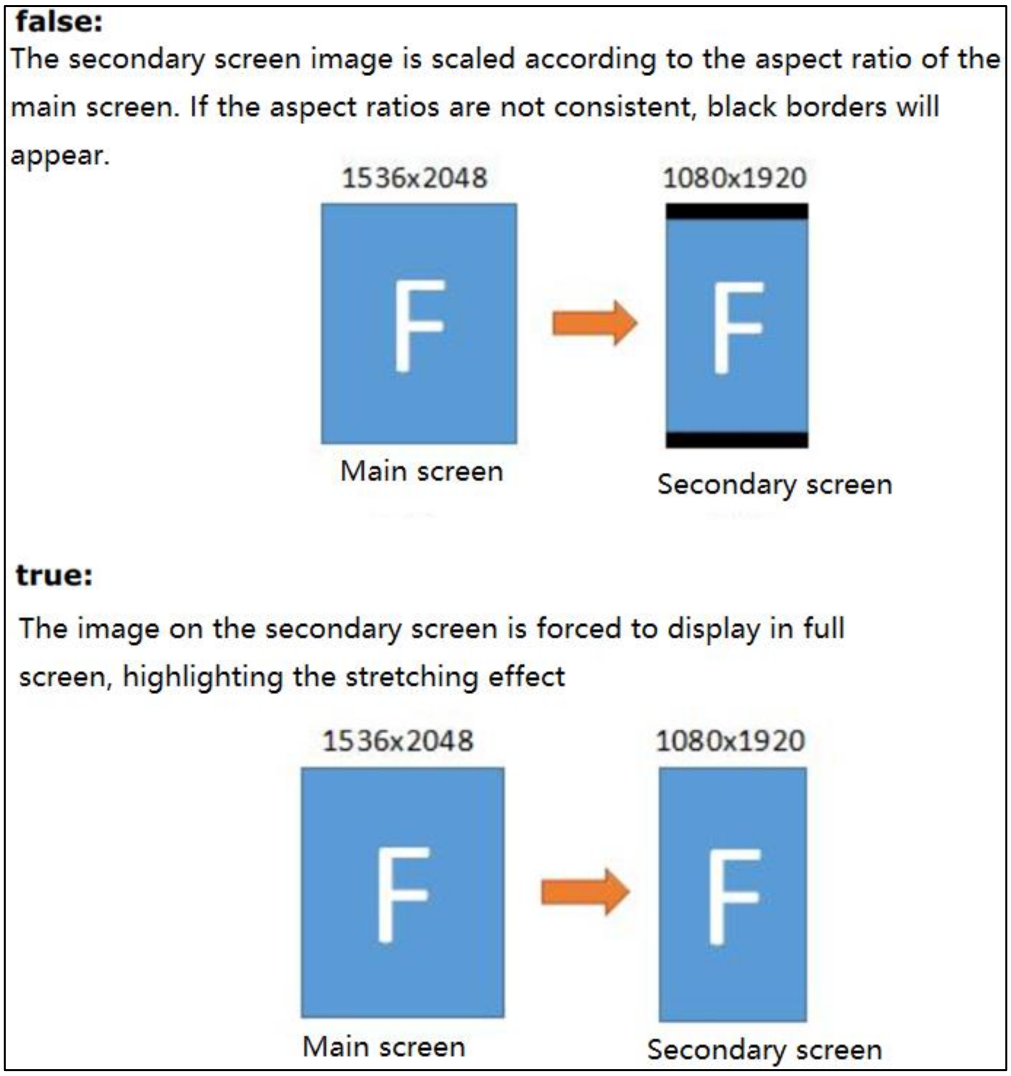


Figure 3.1-8 The display of the secondary screen shows the schematic diagram of the direction rotation.

- persist.sys.rotation.efull-n（n=1、2、3、….）

This attribute is used to set whether the secondary screens (such as Screen 1, Screen 2, etc.) are displayed in full screen mode. The possible values are false and true, as shown in the following figure:


false:

The secondary screen image is scaled according to the aspect ratio of the main screen. If the aspect ratios are not consistent, black borders will appear.

true:

The image on the secondary screen is forced to display in full screen, highlighting the stretching effect

Figure 3.1-9 The secondary screen displays the stretching diagram.

3.2 Screen Display Orientation Configuration

The QuarkPi-CA2 RK3588S card computer supports various screens. It supports the 5.5-inch 720x1280, 5.5-inch 1080x1920, and 10.1-inch 800x1280 MIPI screens sold by ALIENTEK Store, as well as HDMI display and DP display. The MIPI screens of ALIENTEK are all physical vertical screens, while HDMI displays and DP displays are usually physical horizontal screens.

The Android system supports both landscape and portrait display, as shown in the following figure:

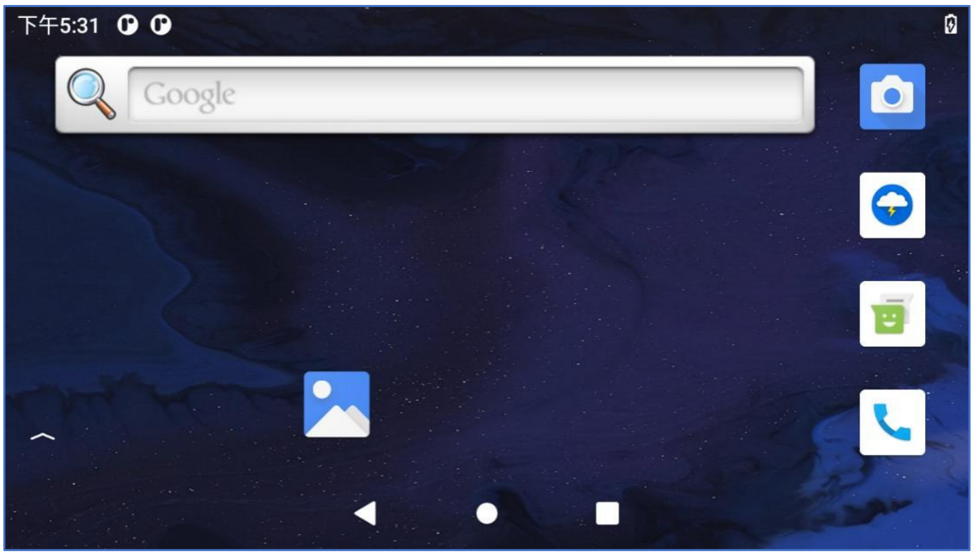


Figure 3.2-1 Horizontal screen display effect

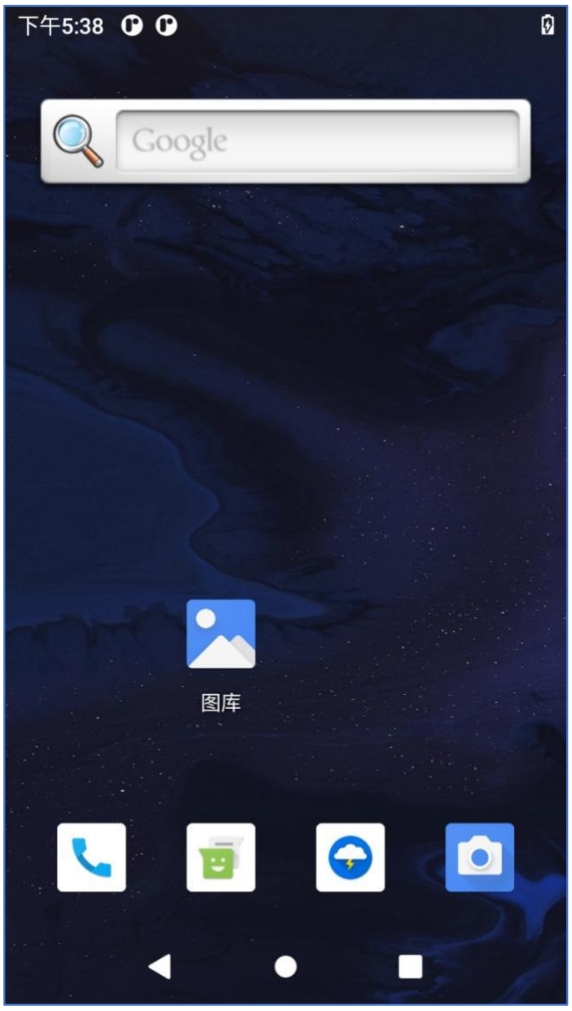


Figure 3.2-2 Vertical screen display effect

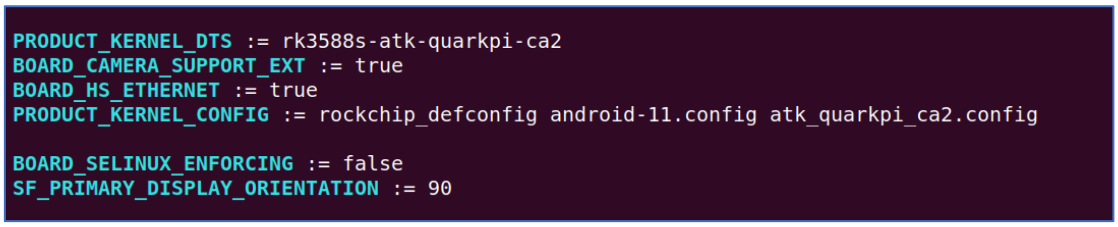
The screen we use can be either a physical landscape screen or a physical portrait screen. This section will explain how to rotate the display orientation of the screen. By rotating, you can achieve landscape display in portrait mode or portrait display in landscape mode.

3.2.1 Main Screen Display Orientation

The rotation angle of the main screen can be controlled by the SF_PRIMARY_DISPLAY_ORIENTATION variable. This variable can take values of 0, 90, 180, and 270, respectively, representing a rotation (clockwise rotation) of 0 degrees, 90 degrees, 180 degrees, and 270 degrees for the main screen display orientation.

Open the file device/rockchip/rk3588/atk_quarkpi_ca2/BoardConfig.mk and add the following content at the end of the file:

```
SF_PRIMARY_DISPLAY_ORIENTATION := 90            # Rotate 90 degrees
```



```
PRODUCT_KERNEL_DTS := rk3588s-atk-quarkpi-ca2
BOARD_CAMERA_SUPPORT_EXT := true
BOARD_HS_ETHERNET := true
PRODUCT_KERNEL_CONFIG := rockchip_defconfig android-11.config atk_quarkpi_ca2.config

BOARD_SELINUX_ENFORCING := false
SF_PRIMARY_DISPLAY_ORIENTATION := 90
```

Figure 3.2-3 Set the rotation angle of the main screen

After the modification is completed, save and exit. Then, execute the following command in the root directory of the SDK to recompile the Android source code:

```
source build/envsetup.sh
lunch atk_quarkpi_ca2-userdebug
./build.sh -A -J10
```

Burn the compiled super.img onto the development board, then power on the development board and start it. After entering the Android system, you will notice that the miPI screen (currently the miPI screen is the main screen) has changed from vertical display to horizontal display, as shown below:

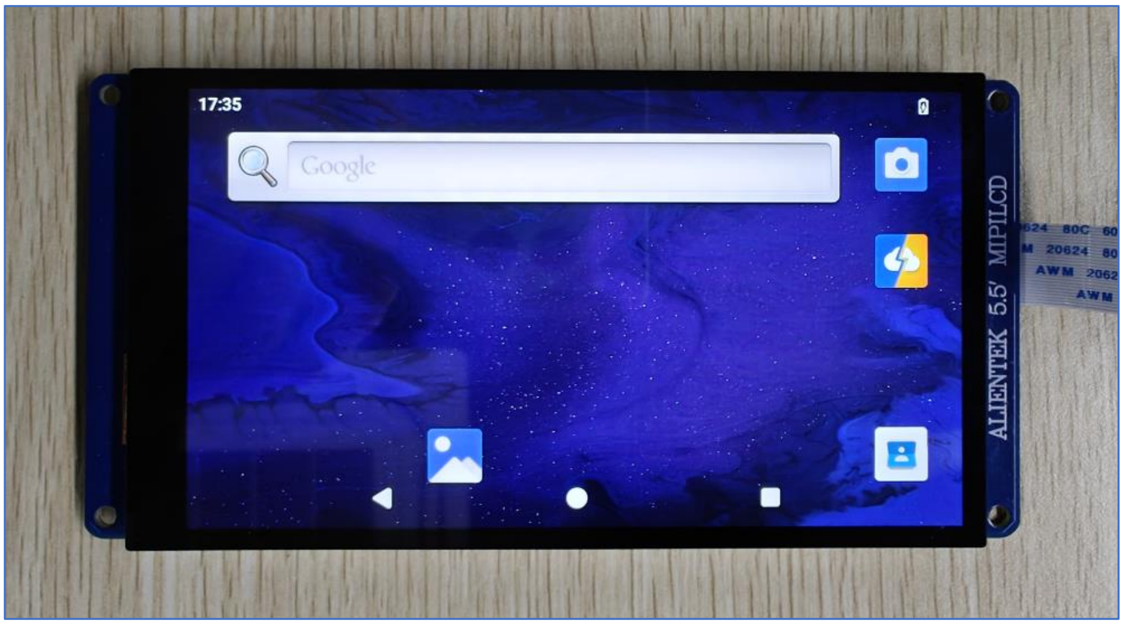
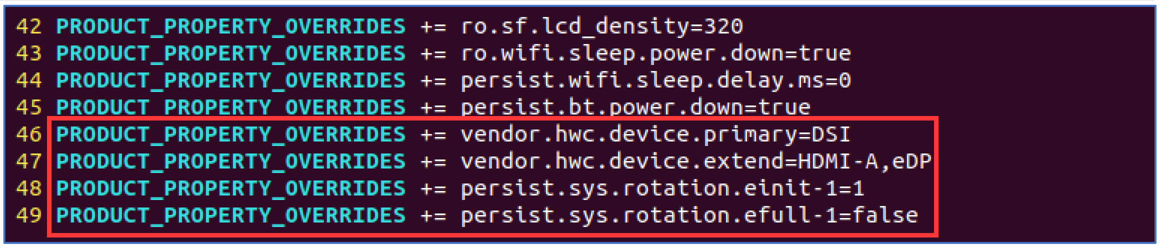


Figure 3.2-4 Horizontal screen display effect

3.2.2 Display Orientation of Secondary Screen

The rotation angle of the secondary screen can be adjusted via persist.sys.rotation.einit-n (where n = 1, 2, 3, ...) .） Attribute control, as introduced in Section 3.1.5. Taking the MIPI main screen and HDMI secondary screen as examples, by adding the following content in the device/rockchip/rk3588/atk_quarkpi_ca2/atk_quarkpi_ca2.mk file, the display direction of the HDMI secondary screen can be rotated by 90 degrees:

```
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=DSI
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=HDMI-A,eDP
PRODUCT_PROPERTY_OVERRIDES += persist.sys.rotation.einit-1=1
PRODUCT_PROPERTY_OVERRIDES += persist.sys.rotation.efull-1=false
```



```
PRODUCT_PROPERTY_OVERRIDES += ro.sf.lcd_density=320
PRODUCT_PROPERTY_OVERRIDES += ro.wifi.sleep.power.down=true
PRODUCT_PROPERTY_OVERRIDES += persist.wifi.sleep.delay.ms=0
PRODUCT_PROPERTY_OVERRIDES += persist.bt.power.down=true
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=DSI
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=HDMI-A,eDP
PRODUCT_PROPERTY_OVERRIDES += persist.sys.rotation.einit-1=1
PRODUCT_PROPERTY_OVERRIDES += persist.sys.rotation.efull-1=false
```

Figure 3.2-5 Modified example:

After the modification is completed, save and exit. Then, execute the following command in the root directory of the SDK to recompile the Android source code:

```
source build/envsetup.sh
lunch atk_quarkpi_ca2-userdebug
./build.sh -A -J10
```

Burn the compiled super.img onto the development board, then power on the development board, connect the HDMI display, and after entering the Android system, you will notice that the display direction of the HDMI screen has rotated by 90 degrees, as shown below:

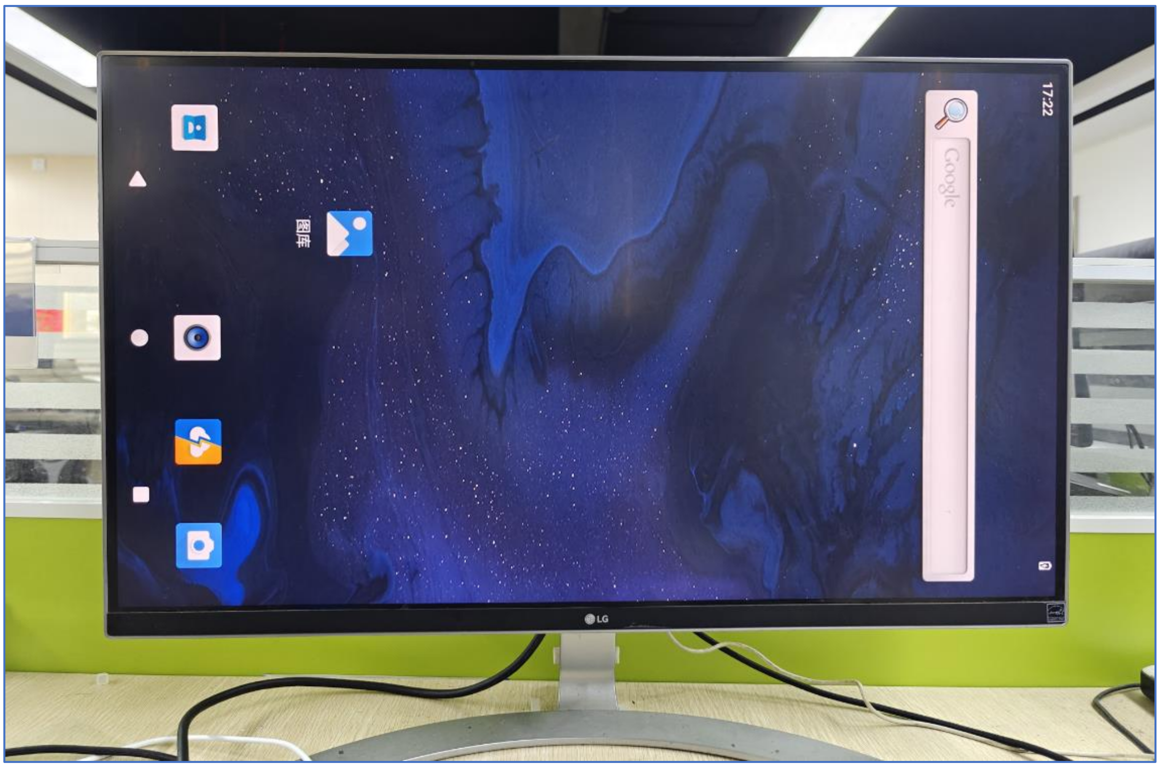


Figure 3.2-6 HDMI display effect

3.2.3 Logo display direction

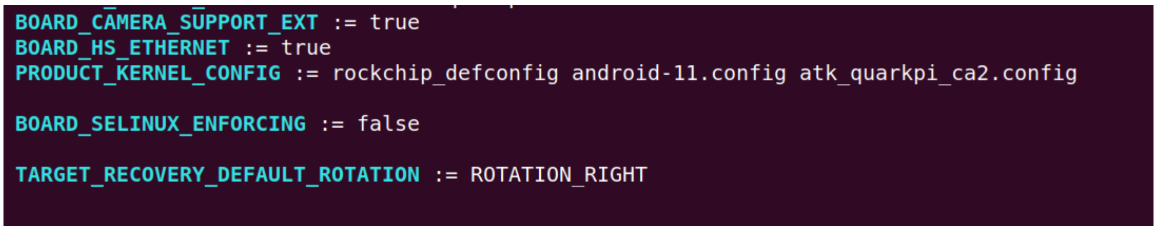
The RK platform's U-Boot does not default to supporting the logo rotation function. ALIENTEK has implemented the logo rotation function based on the RK U-Boot. The usage method is very simple. Just make a simple configuration of the kernel device tree. For specific usage methods, please refer to: [ALIENTEK] QuarkPi-CA2 Startup Logo Customization Reference Manual V1.0.pdf.

3.2.4 Screen Display Direction in Recovery Mode

The screen display direction in recovery mode can be controlled by the TARGET_RECOVERY_DEFAULT_ROTATION variable. This variable can take values of: ROTATION_NONE, ROTATION_RIGHT, ROTATION_DOWN, and ROTATION_LEFT, respectively, indicating that the screen display direction in recovery mode will rotate 0 degrees, 90 degrees, 180 degrees, and 270 degrees.

For example, if you want to rotate the screen display direction in recovery mode by 90 degrees, you can open the file device/rockchip/rk3588/atk_quarkpi_ca2/BoardConfig.mk, and then add the following content:

```
TARGET_RECOVERY_DEFAULT_ROTATION := ROTATION_RIGHT
```



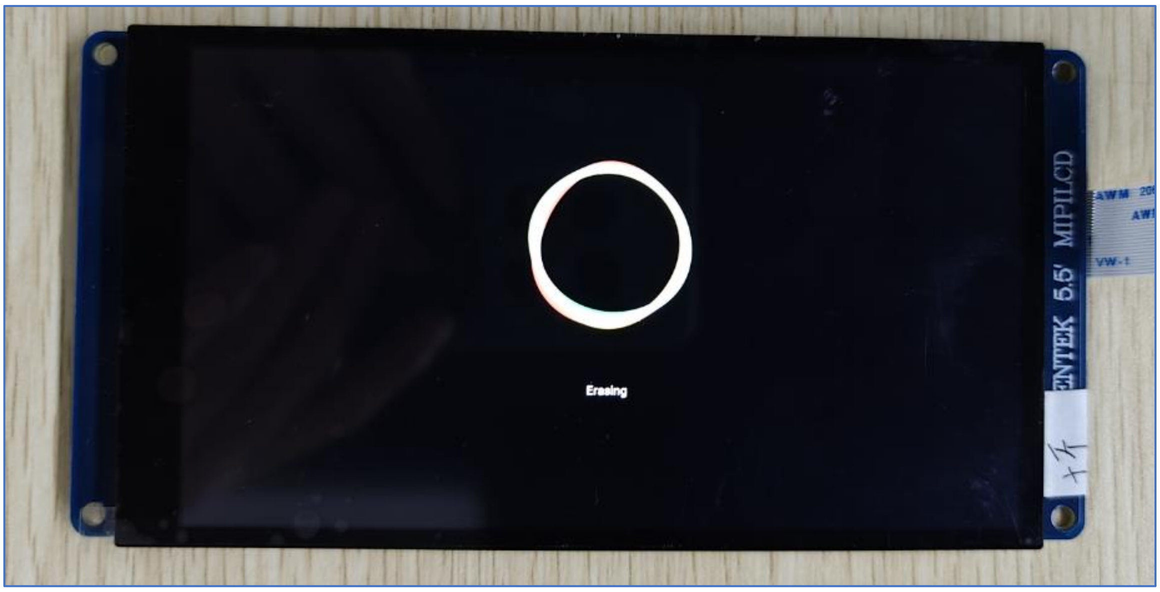
```
BOARD_CAMERA_SUPPORT_EXT := true
BOARD_HS_ETHERNET := true
PRODUCT_KERNEL_CONFIG := rockchip_defconfig android-11.config atk_quarkpi_ca2.config

BOARD_SELINUX_ENFORCING := false

TARGET_RECOVERY_DEFAULT_ROTATION := ROTATION_RIGHT
```

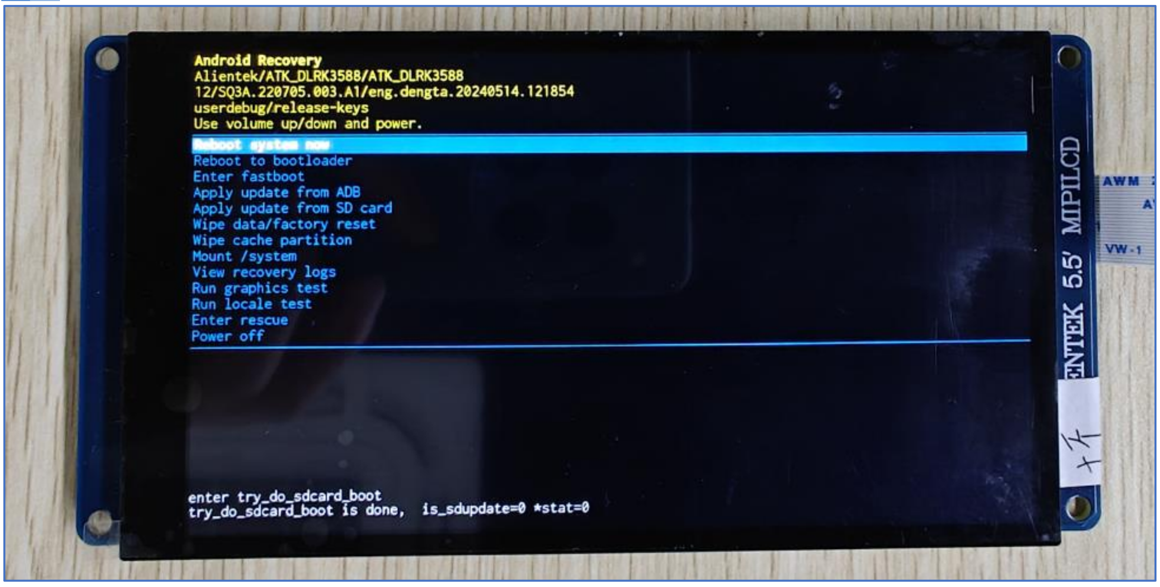
Figure 3.2-7 Modify the screen rotation angle in the recovery mode.

After the modification is completed, save and exit. Then, execute the following command in the root directory of the SDK to recompile the Android source code:

```
source build/envsetup.sh
lunch atk_quarkpi_ca2-userdebug
./build.sh -A -J10
```

Burn the generated recovery.img and misc.img files onto the development board. Once the burning is completed, the device will automatically restart and enter the recovery mode (for the first startup after burning misc.img, it will also enter the recovery mode). At this time, you will notice that the screen display has rotated 90 degrees, as shown in the following picture:





```
Android Recovery
Alientek/ATK_DLRK3588/ATK_DLRK3588
12/SQ3A.220705.003.A1/eng.dengta.20240514.121854
userdebug/release-keys
Use volume up/down and power.
Reboot system now
Reboot to bootloader
Enter fastboot
Apply update from ADB
Apply update from SD card
Wipe data/factory reset
Wipe cache partition
Mount /system
View recovery logs
Run graphics test
Run locale test
Enter rescue
Power off

enter try_do_sdcard_boot
try_do_sdcard_boot is done,  is_sdupdate=0 *stat=0
```

Figure 3.2-8 Display Effect

3.3 Touch Direction Configuration

When the display direction of the screen rotates, the touch direction should also rotate accordingly; otherwise, the touch operation will not function properly. For the MIPI main screen, the kernel device tree touch screen node needs to be configured.

The attributes related to the touch direction of the MIPI screen are as follows:

```
touchscreen-inverted-x;       # Define this attribute and indicate that the X-axis will be flipped.
touchscreen-swapped-x-y;      # Define this attribute and it represents the operation of swapping the X-axis and the Y-axis.
touchscreen-inverted-y;       # Define this attribute and indicate that the Y-axis will be inverted.
```

Based on the rotation angle, the corresponding two attributes are added to the touchscreen device node.

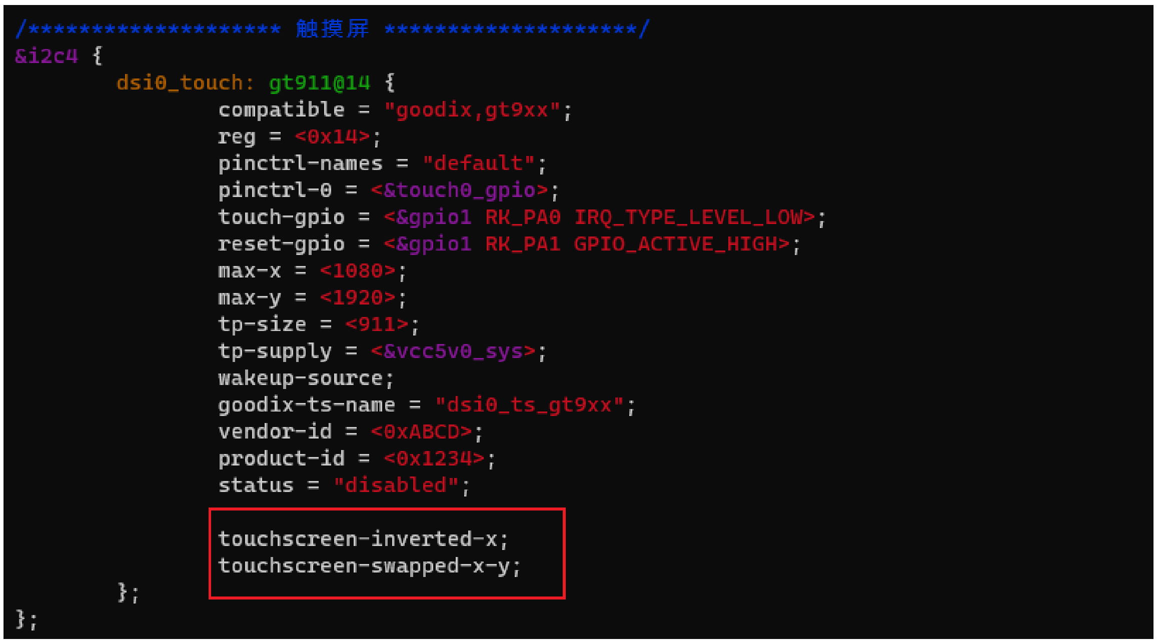
The "flipping" mentioned in the previous text refers to the flipping of coordinate values. For example, in an 800x1280 touchscreen (vertical screen), the coordinate values corresponding to the X-axis increase sequentially from left to right: 0 → 799, and the coordinate values corresponding to the Y-axis increase sequentially from top to bottom: 0 → 1279. If the X-axis is flipped, from left to right it will become sequentially decreasing: 799 → 0. If the Y-axis is flipped, from top to bottom it will become sequentially decreasing: 1279 → 0.

The exchange of the X-axis and Y-axis is easier to understand. Usually, physical horizontal-to-vertical display (rotation by 90 or 270 degrees) or physical vertical-to-horizontal display (rotation by 90 or 270 degrees) all require the exchange of X-axis and Y-axis coordinates.

- Add the following two attributes, and the touch direction of the MIPI screen can be rotated by 90 degrees:

```
touchscreen-inverted-x;
touchscreen-swapped-x-y;
```

Open the device tree file "arch/arm64/boot/dts/rockchip/rk3588s-atk-lcds.dtsi" located in the kernel source code directory. Add the following content to the "dsi0_touch" node (the node corresponding to the touch screen device of MIPI DSI0):



```
/********************* 触摸屏 *********************/
&i2c4 {
        dsi0_touch: gt911@14 {
                compatible = "goodix,gt9xx";
                reg = <0x14>;
                pinctrl-names = "default";
                pinctrl-0 = <&touch0_gpio>;
                touch-gpio = <&gpio1 RK_PA0 IRQ_TYPE_LEVEL_LOW>;
                reset-gpio = <&gpio1 RK_PA1 GPIO_ACTIVE_HIGH>;
                max-x = <1080>;
                max-y = <1920>;
                tp-size = <911>;
                tp-supply = <&vcc5v0_sys>;
                wakeup-source;
                goodix-ts-name = "dsi0_ts_gt9xx";
                vendor-id = <0xABCD>;
                product-id = <0x1234>;
                status = "disabled";

                touchscreen-inverted-x;
                touchscreen-swapped-x-y;
        };
};
```

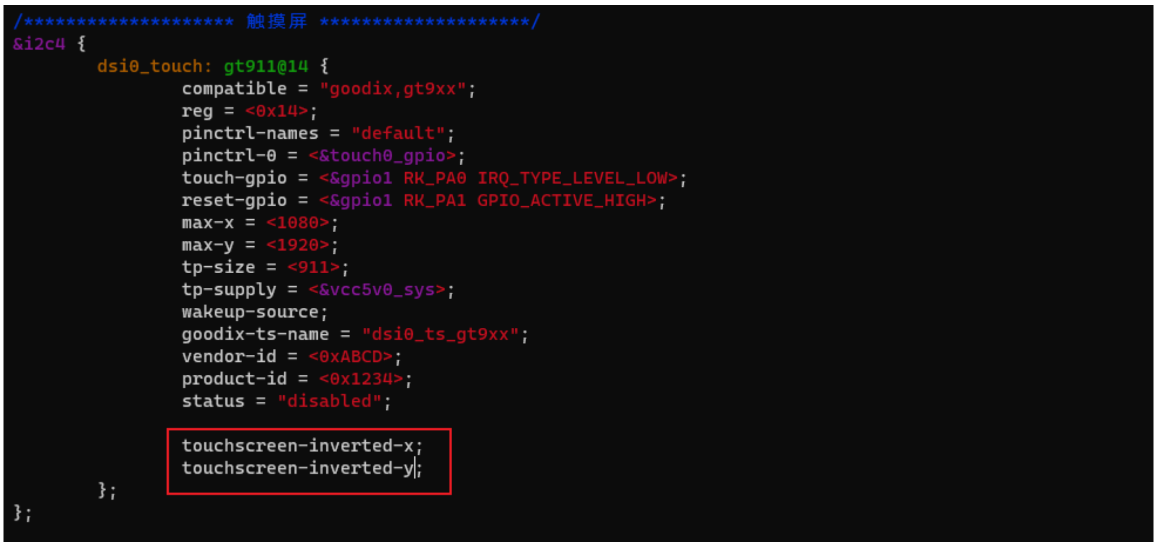
Figure 3.3-1 Rotate the direction by 90 degrees

The touch screen device node corresponding to MIPI DSI0 is dsi0_touch, and the touch screen device node corresponding to MIPI DSI1 is dsi1_touch.

- To rotate the touch direction of the MIPI screen by 180 degrees, add the following two attributes:

```
touchscreen-inverted-x;
touchscreen-inverted-y;
```

Open the device tree file "arch/arm64/boot/dts/rockchip/rk3588s-atk-lcds.dtsi" located in the kernel source code directory. Add the following content to the "dsi0_touch" node (the node corresponding to the touch screen device of MIPI DSI0):



```
/********************* 触摸屏 *********************/
&i2c4 {
        dsi0_touch: gt911@14 {
                compatible = "goodix,gt9xx";
                reg = <0x14>;
                pinctrl-names = "default";
                pinctrl-0 = <&touch0_gpio>;
                touch-gpio = <&gpio1 RK_PA0 IRQ_TYPE_LEVEL_LOW>;
                reset-gpio = <&gpio1 RK_PA1 GPIO_ACTIVE_HIGH>;
                max-x = <1080>;
                max-y = <1920>;
                tp-size = <911>;
                tp-supply = <&vcc5v0_sys>;
                wakeup-source;
                goodix-ts-name = "dsi0_ts_gt9xx";
                vendor-id = <0xABCD>;
                product-id = <0x1234>;
                status = "disabled";

                touchscreen-inverted-x;
                touchscreen-inverted-y;
        };
};
```

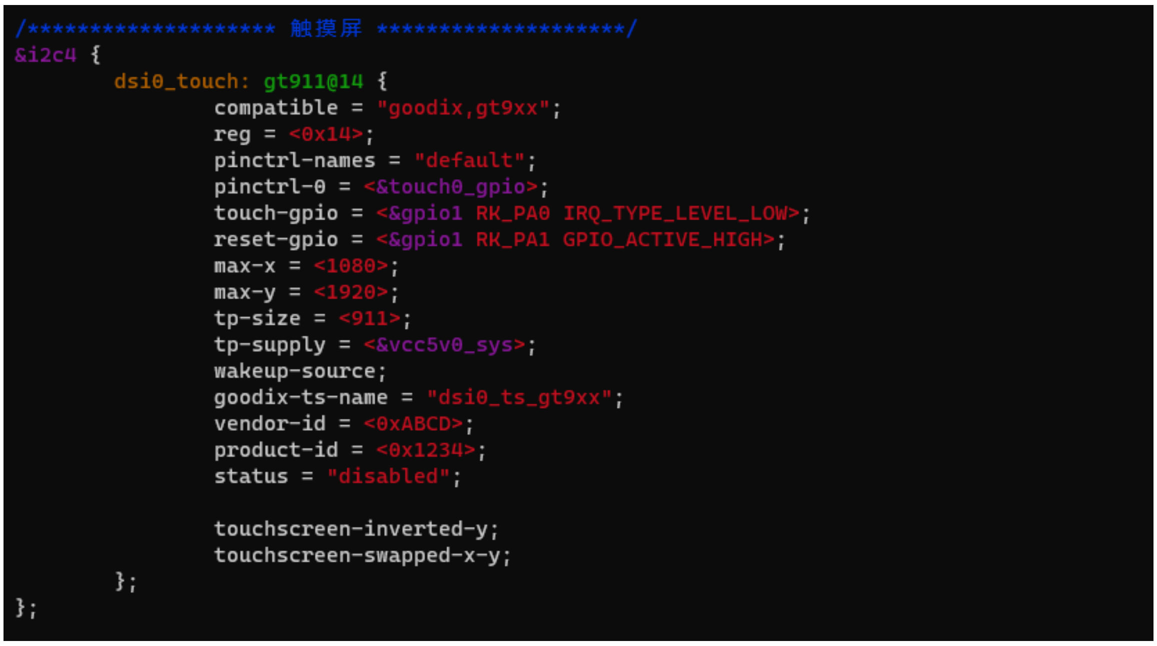
Figure 3.3-2 Rotate the direction by 180 degrees

The dsi0_touch is the touchscreen device node corresponding to MIPI DSI0, and the touchscreen device node corresponding to MIPI DSI1 is dsi1_touch.

- By adding the following two attributes, the touch direction of the MIPI screen can be rotated by 270 degrees:

```
touchscreen-inverted-y;
touchscreen-swapped-x-y;
```

Open the device tree file "arch/arm64/boot/dts/rockchip/rk3588s-atk-lcds.dtsi" located in the kernel source code directory. Add the following content to the "dsi0_touch" node (the node corresponding to the touch screen device of MIPI DSI0):



```
/******************** 触摸屏 ********************/
&i2c4 {
        dsi0_touch: gt911@14 {
                compatible = "goodix,gt9xx";
                reg = <0x14>;
                pinctrl-names = "default";
                pinctrl-0 = <&touch0_gpio>;
                touch-gpio = <&gpio1 RK_PA0 IRQ_TYPE_LEVEL_LOW>;
                reset-gpio = <&gpio1 RK_PA1 GPIO_ACTIVE_HIGH>;
                max-x = <1080>;
                max-y = <1920>;
                tp-size = <911>;
                tp-supply = <&vcc5v0_sys>;
                wakeup-source;
                goodix-ts-name = "dsi0_ts_gt9xx";
                vendor-id = <0xABCD>;
                product-id = <0x1234>;
                status = "disabled";

                touchscreen-inverted-y;
                touchscreen-swapped-x-y;
        };
};
```

Figure 3.3-3 Rotate the direction by 270 degrees.

dsi0_touch is the touchscreen device node corresponding to MIPI DSI0, and the touchscreen device node corresponding to MIPI DSI1 is dsi1_touch.

Based on the rotation angle, add the corresponding attributes to the touchscreen device node. After the modification is completed, save and exit. Execute the following command in the SDK root directory to recompile the Linux kernel and Android source code:

```
source build/envsetup.sh
lunch atk_quarkpi_ca2-userdebug
./build.sh -KA -J10
```

Burn the compiled boot.img onto the development board for testing.

3.4 Multi-screen Touch Configuration

In some Android products that support multi-screen display, sometimes both the primary and secondary screens are equipped with touch screens and both support touch functionality. On both the primary and secondary screens, you can operate the device through touch screens; moreover, in the case of dual display, the touch operations on the primary and secondary screens do not interfere with each other. This is the multi-screen touch function described in this section.

To achieve multi-screen touch interaction, it is necessary to bind the TP (Touch Panel) with the corresponding screen. The main screen TP is bound to the main screen, and the secondary screen TP is bound to the secondary screen. That is, the touch input events generated by each TP are sent to the corresponding screen window.

In the existing Android framework, all external hot-pluggable input devices are by default uniformly designated as secondary screen input devices, such as USB touch screens, Bluetooth touch screens, etc.; while all built-in input devices are uniformly designated as primary screen input devices, such as I2C touch screens. Taking dual-screen display as an example, according to this logic, if our main screen is an I2C touch screen and the secondary screen is a USB touch screen, then the Android system can default support multi-screen touch; but sometimes we have a combination of dual I2C touch screens or dual USB touch screens, then we need to modify the existing logic and reconfigure.

There are generally two methods:

1. Modify the Android source code: frameworks/native/services/inputflinger/reader/EventHub.cpp

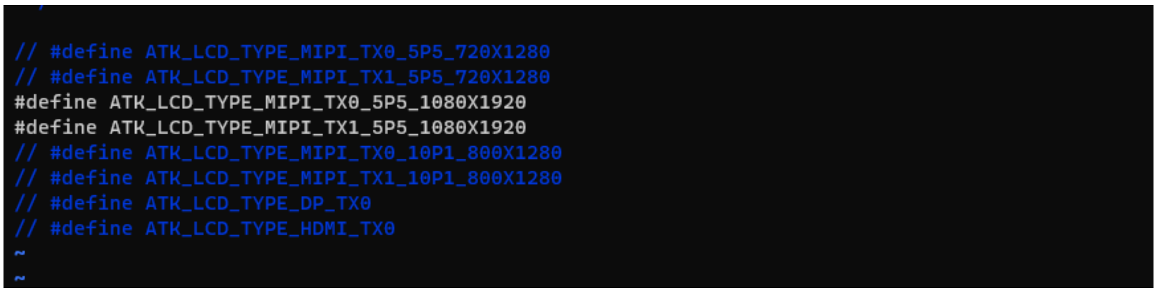
2. Configure the IDC file of the touch screen

The first method will not be introduced. Modifying the Android source code is quite troublesome; it is recommended to use the second method, which is to configure the IDC file of the touch screen. The IDC (Input Device Configuration) file is an input device configuration file, which is used to define and configure the behavior of input devices; the IDC file contains the configuration attributes of specific input devices, and these attributes will affect the behavior of the input devices.

3.4.1 Example: Dual MIPI Screen Touching

The following will be explained through an example: Dual MIPI screen display (dual I2C touch). Since our MIPI display has I2C touch functionality, as mentioned above, the Android system by default will assign all I2C touch input devices to the main screen as the input device. This is definitely not the result we expected. We need to modify the existing logic to make the main screen TP designated as the input device for the main screen and the secondary screen TP designated as the input device for the secondary screen.

Open the Linux kernel device tree file kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi and enable the two MIPI screen display interfaces (for example, both MIPI DSI interfaces are connected to a 5.5-inch 1080x1920 MIPI screen):



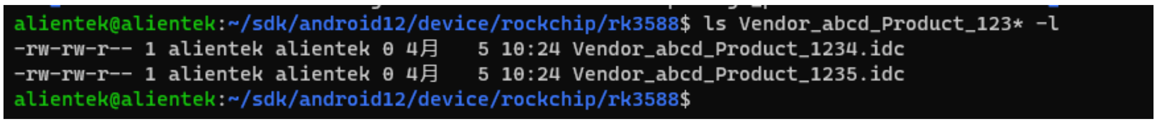
```
// #define ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_720X1280
#define ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920
#define ATK_LCD_TYPE_MIPI_TX1_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280
// #define ATK_LCD_TYPE_MIPI_TX1_10P1_800X1280
// #define ATK_LCD_TYPE_DP_TX0
// #define ATK_LCD_TYPE_HDMI_TX0
~
~
```

Figure 3.4-1 Display interface configuration

In this case, the default MIPI DSI0 will be used as the main screen, and MIPI DSI1 as the secondary screen. After making the modifications, save and exit!

Enter the "device/rockchip/rk3588" directory, and create two.idc files: Vendor_abcd_Product_1234.idc, Vendor_abcd_Product_1235.idc:

```
cd device/rockchip/rk3588/
touch Vendor_abcd_Product_1234.idc Vendor_abcd_Product_1235.idc
```



```
alientek@alientek:~/sdk/android12/device/rockchip/rk3588$ ls Vendor_abcd_Product_123* -l
-rw-rw-r-- 1 alientek alientek 0 4月   5 10:24 Vendor_abcd_Product_1234.idc
-rw-rw-r-- 1 alientek alientek 0 4月   5 10:24 Vendor_abcd_Product_1235.idc
alientek@alientek:~/sdk/android12/device/rockchip/rk3588$
```

Figure 3.4-2 Create touchscreen IDC file

Vendor_abcd_Product_1234.idc is used to define and configure the behavior of the MIPI DSI0 screen TP (in this case, it corresponds to the main screen TP);

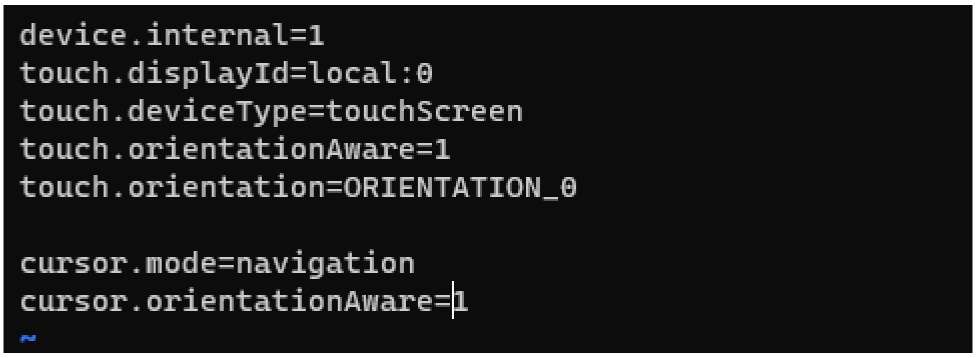
Vendor_abcd_Product_1235.idc is used to define and configure the behavior of the MIPI DSI1 screen TP (in this case, it corresponds to the secondary screen TP).

The naming convention for IDC files is: Vendor_vid_Product_pid.idc

pid and vid respectively refer to the supplier ID and product ID of the input device, and they cannot be filled incorrectly; otherwise, they won't match. How to query the pid and vid of the input device? You can read the 3.4.2 section. Besides this naming method, you can directly name it using the input device name, such as DEVICE_NAME.idc.

Open the Vendor_abcd_Product_1234.idc file and add the following content:

```
device.internal = 1
touch.displayId = local:0
touch.deviceType = touchScreen
touch.orientationAware = 1
touch.orientation = ORIENTATION_0

cursor.mode = navigation
cursor.orientationAware = 1
```



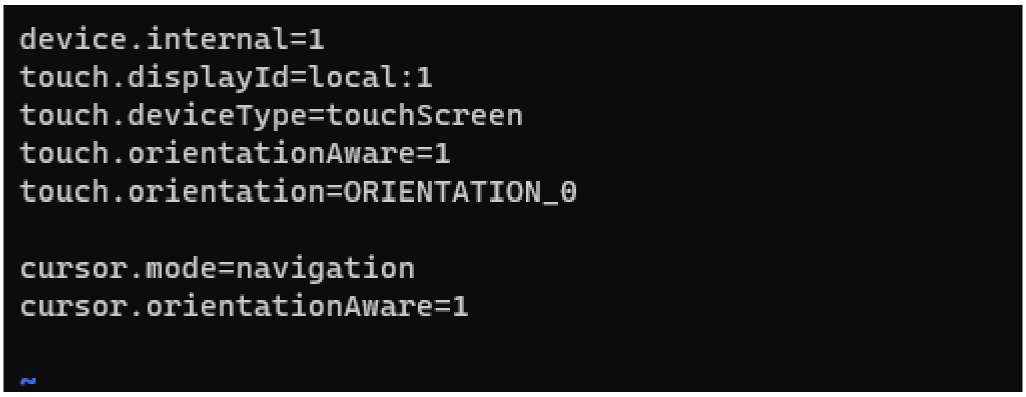
```
device.internal=1
touch.displayId=local:0
touch.deviceType=touchScreen
touch.orientationAware=1
touch.orientation=ORIENTATION_0

cursor.mode=navigation
cursor.orientationAware=1
~
```

Figure 3.4-3 Configure the IDC file

After the modification is completed, save and exit!

Open the Vendor_abcd_Product_1235.idc file and add the following content:

```
device.internal = 1
touch.displayId = local:1
touch.deviceType = touchScreen
touch.orientationAware = 1
touch.orientation = ORIENTATION_0

cursor.mode = navigation
cursor.orientationAware = 1
```



```
device.internal=1
touch.displayId=local:1
touch.deviceType=touchScreen
touch.orientationAware=1
touch.orientation=ORIENTATION_0

cursor.mode=navigation
cursor.orientationAware=1

~
```

Figure 3.4-4 Configure the IDC file

After modification, save and exit!

Here is a brief explanation of the meanings of each attribute in the IDC file:

device.internal: It can take the values of 0 or 1. It is used to define whether the input device is an internal device (such as an I2C touch device) or an externally connected device (which can be hot-swappable and detachable, such as a touch screen with a USB interface, a touch screen connected via Bluetooth, etc.). If the value is 0, then the device is an external connected device; if the value is 1, then the device is an internal device.

touch.deviceType: It is used to define the type of the input device. "touchScreen" indicates that the device is a touch screen device.

touch.orientationAware: Can take values of 0 or 1, used to define whether the touchscreen rotates along with the display. If the value is 0, it does not rotate with the display; if the value is 1, it rotates with the display.

touch.orientation: Used to define the initial rotation angle of the touchscreen, and can take the following values:

- ORIENTATION_0
- ORIENTATION_90
- ORIENTATION_180
- ORIENTATION_270

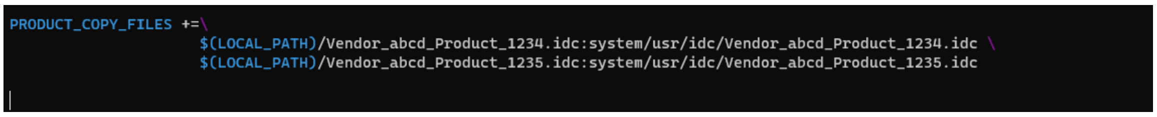
These represent rotations of 0 degrees, 90 degrees, 180 degrees, and 270 degrees respectively.

touch.displayId: Used to specify the display, and will be assigned as the input device of that screen. Its value is equal to the unique ID of the display. In the Android system, it is called uniqueId. In this example, local:0 is the unique ID of the MIPI DSI0 screen, and local:1 is the unique ID of the MIPI DSI1 screen. How to query the unique ID of the display? You can read the 3.4.3 section.

In fact, in our example, we only need to create the idc file of the secondary screen TP. Therefore, the Vendor_abcd_Product_1234.idc (which is the idc file of the main screen TP in this example) can be omitted.

Next, open the file "device/rockchip/rk3588/device.mk" and add the following content:

```
PRODUCT_COPY_FILES += \
$(LOCAL_PATH)/Vendor_abcd_Product_1234.idc:system/usr/idc/Vendor_abcd_Product_1234.
idc \
$(LOCAL_PATH)/Vendor_abcd_Product_1235.idc:system/usr/idc/Vendor_abcd_Product_1235.idc
```



```
PRODUCT_COPY_FILES +=\
                        $(LOCAL_PATH)/Vendor_abcd_Product_1234.idc:system/usr/idc/Vendor_abcd_Product_1234.idc \
                        $(LOCAL_PATH)/Vendor_abcd_Product_1235.idc:system/usr/idc/Vendor_abcd_Product_1235.idc
```

Figure 3.4-5 Modify device.mk

The purpose of adding these three lines is to copy the two idc files to the /system/usr/idc/ directory of the Android system. After adding them, save and exit.

After the above modifications are completed, execute the following command in the root directory of the SDK to recompile the Linux kernel and the Android source code.

```
source build/envsetup.sh
lunch atk_quarkpi_ca2-userdebug
./build.sh -KA -J10
```

Connect the two MIPI screens to the MIPI DSI0 interface and MIPI DSI1 interface on the development board respectively. In this example, MIPI DSI0 is connected to a 5.5-inch 1080x1920 MIPI screen, and MIPI DSI1 is connected to another 5.5-inch 1080x1920 MIPI screen.

Burn the compiled super.img and boot.img onto the development board, then start the development board to enter the Android system, as shown below:

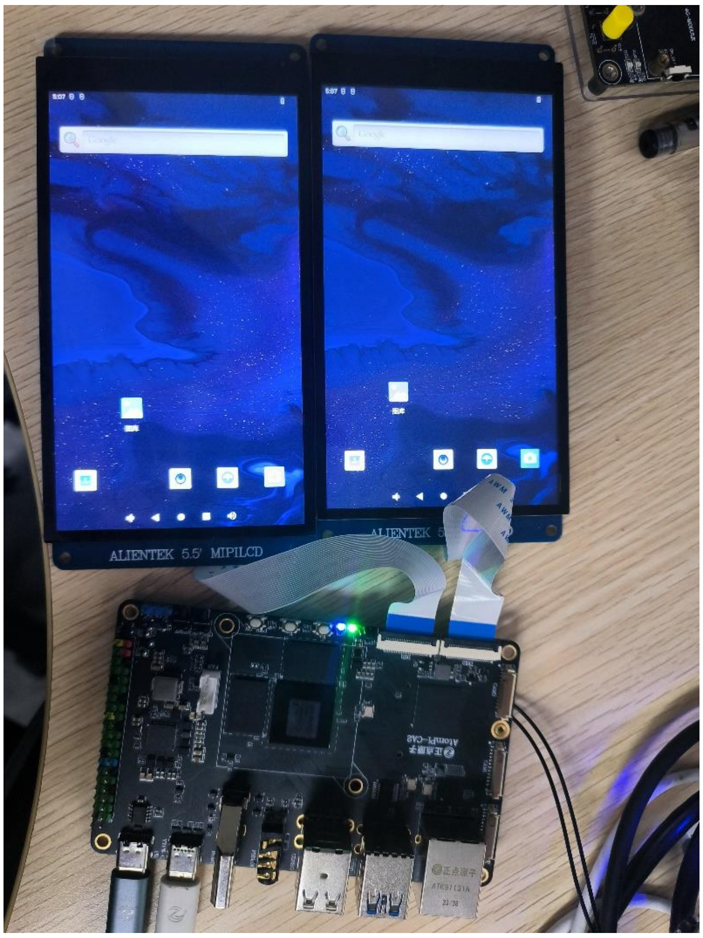
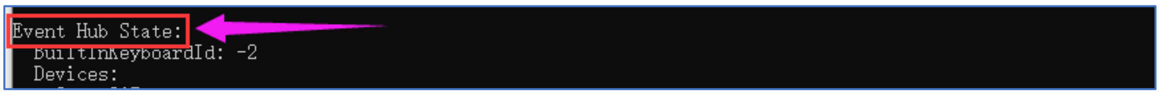


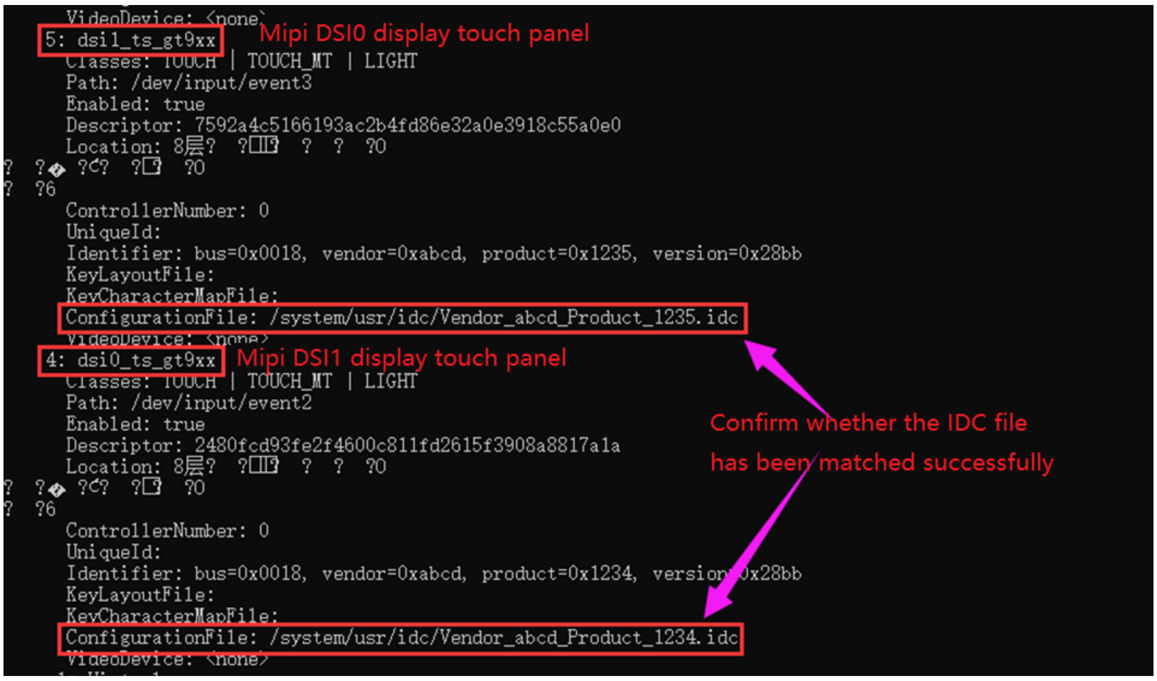
Figure 3.4-6 Dual MIPI display screens

Next, we will conduct the test. First, we need to confirm whether our modifications have taken effect. On the RK3588 Android system, execute the "dumpsys input" command. From the "Event Hub State" information, find the contents of the two MIPI screen touch points and confirm whether the touch screen matches the specified idc file:

```
dumpsys input
```



```
Event Hub State:
  BuiltInKeyboardId: -2
  Devices:
```

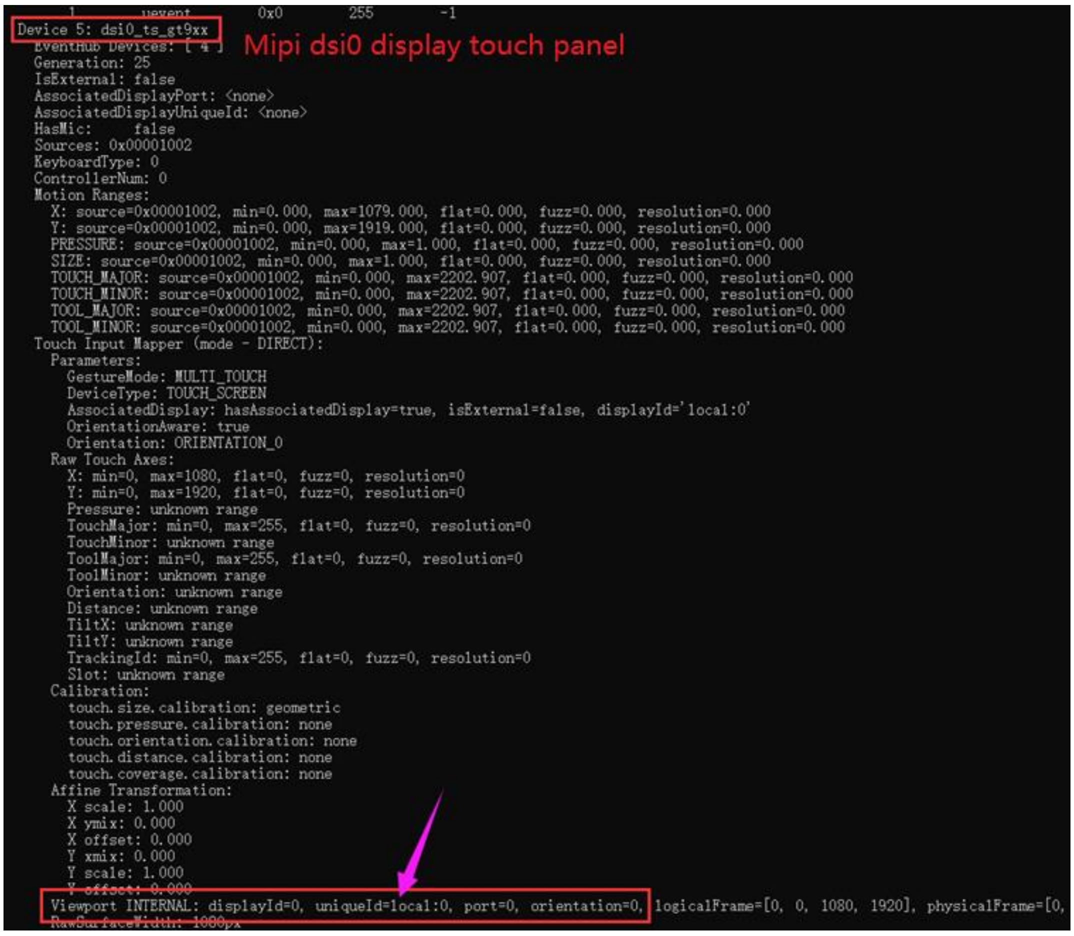


```
    VideoDevice: <none>
  5: dsi1_ts_gt9xx
    Classes: TOUCH | TOUCH_MT | LIGHT
    Path: /dev/input/event3
    Enabled: true
    Descriptor: 7592a4c5166193ac2b4fd86e32a0e3918c55a0e0
    Location: 8层?  ?□?  ?  ?  ?0
?  ?� ?ç? ?□  ?0
?  ?6
    ControllerNumber: 0
    UniqueId:
    Identifier: bus=0x0018, vendor=0xabcd, product=0x1235, version=0x28bb
    KeyLayoutFile:
    KeyCharacterMapFile:
    ConfigurationFile: /system/usr/idc/Vendor_abcd_Product_1235.idc
    VideoDevice: <none>
  4: dsi0_ts_gt9xx
    Classes: TOUCH | TOUCH_MT | LIGHT
    Path: /dev/input/event2
    Enabled: true
    Descriptor: 2480fcd93fe2f4600c811fd2615f3908a8817a1a
    Location: 8层?  ?□?  ?  ?  ?0
?  ?� ?ç? ?□  ?0
?  ?6
    ControllerNumber: 0
    UniqueId:
    Identifier: bus=0x0018, vendor=0xabcd, product=0x1234, version=0x28bb
    KeyLayoutFile:
    KeyCharacterMapFile:
    ConfigurationFile: /system/usr/idc/Vendor_abcd_Product_1234.idc
    VideoDevice: <none>
```

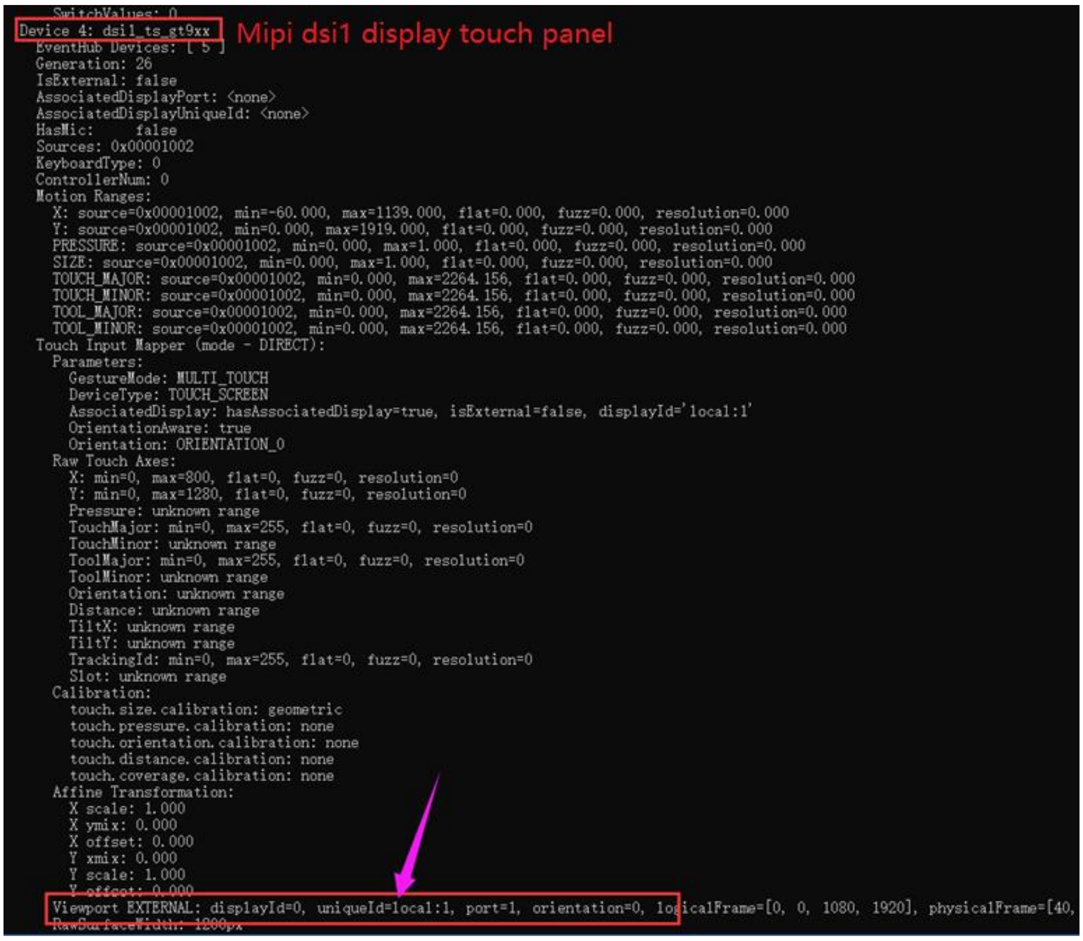
Mipi DSI0 display touch panel
Mipi DSI1 display touch panel
Confirm whether the IDC file has been matched successfully

Figure 3.4-7 Confirm whether the IDC file has been matched successfully.

Then, from the "Input Reader State" information, find the contents of the two MIPI display touch pads, and confirm whether the two touch screens are respectively designated as the main screen touch pad and the secondary screen touch pad. In this example, MIPI DSI0 is the main screen (uniqueId equals local:0), and MIPI DSI1 is the secondary screen (uniqueId equals local:1).



```
      1        uevent        0x0        255        -1
Device 5: dsi0_ts_gt9xx
    EventHub Devices: [ 4 ]
    Generation: 25
    IsExternal: false
    AssociatedDisplayPort: <none>
    AssociatedDisplayUniqueId: <none>
    HasMic:      false
    Sources: 0x00001002
    KeyboardType: 0
    ControllerNum: 0
    Motion Ranges:
      X: source=0x00001002, min=0.000, max=1079.000, flat=0.000, fuzz=0.000, resolution=0.000
      Y: source=0x00001002, min=0.000, max=1919.000, flat=0.000, fuzz=0.000, resolution=0.000
      PRESSURE: source=0x00001002, min=0.000, max=1.000, flat=0.000, fuzz=0.000, resolution=0.000
      SIZE: source=0x00001002, min=0.000, max=1.000, flat=0.000, fuzz=0.000, resolution=0.000
      TOUCH_MAJOR: source=0x00001002, min=0.000, max=2202.907, flat=0.000, fuzz=0.000, resolution=0.000
      TOUCH_MINOR: source=0x00001002, min=0.000, max=2202.907, flat=0.000, fuzz=0.000, resolution=0.000
      TOOL_MAJOR: source=0x00001002, min=0.000, max=2202.907, flat=0.000, fuzz=0.000, resolution=0.000
      TOOL_MINOR: source=0x00001002, min=0.000, max=2202.907, flat=0.000, fuzz=0.000, resolution=0.000
    Touch Input Mapper (mode - DIRECT):
      Parameters:
        GestureMode: MULTI_TOUCH
        DeviceType: TOUCH_SCREEN
        AssociatedDisplay: hasAssociatedDisplay=true, isExternal=false, displayId='local:0'
        OrientationAware: true
        Orientation: ORIENTATION_0
      Raw Touch Axes:
        X: min=0, max=1080, flat=0, fuzz=0, resolution=0
        Y: min=0, max=1920, flat=0, fuzz=0, resolution=0
        Pressure: unknown range
        TouchMajor: min=0, max=255, flat=0, fuzz=0, resolution=0
        TouchMinor: unknown range
        ToolMajor: min=0, max=255, flat=0, fuzz=0, resolution=0
        ToolMinor: unknown range
        Orientation: unknown range
        Distance: unknown range
        TiltX: unknown range
        TiltY: unknown range
        TrackingId: min=0, max=255, flat=0, fuzz=0, resolution=0
        Slot: unknown range
      Calibration:
        touch.size.calibration: geometric
        touch.pressure.calibration: none
        touch.orientation.calibration: none
        touch.distance.calibration: none
        touch.coverage.calibration: none
      Affine Transformation:
        X scale: 1.000
        X ymix: 0.000
        X offset: 0.000
        Y xmix: 0.000
        Y scale: 1.000
        Y offset: 0.000
      Viewport INTERNAL: displayId=0, uniqueId=local:0, port=0, orientation=0, logicalFrame=[0, 0, 1080, 1920], physicalFrame=[0,
      RawSurfaceWidth: 1080px
```



```
    SwitchValues: 0
  Device 4: dsi1_ts_gt9xx
    EventHub Devices: [ 5 ]
    Generation: 26
    IsExternal: false
    AssociatedDisplayPort: <none>
    AssociatedDisplayUniqueId: <none>
    HasMic:     false
    Sources: 0x00001002
    KeyboardType: 0
    ControllerNum: 0
    Motion Ranges:
      X: source=0x00001002, min=-60.000, max=1139.000, flat=0.000, fuzz=0.000, resolution=0.000
      Y: source=0x00001002, min=0.000, max=1919.000, flat=0.000, fuzz=0.000, resolution=0.000
      PRESSURE: source=0x00001002, min=0.000, max=1.000, flat=0.000, fuzz=0.000, resolution=0.000
      SIZE: source=0x00001002, min=0.000, max=1.000, flat=0.000, fuzz=0.000, resolution=0.000
      TOUCH_MAJOR: source=0x00001002, min=0.000, max=2264.156, flat=0.000, fuzz=0.000, resolution=0.000
      TOUCH_MINOR: source=0x00001002, min=0.000, max=2264.156, flat=0.000, fuzz=0.000, resolution=0.000
      TOOL_MAJOR: source=0x00001002, min=0.000, max=2264.156, flat=0.000, fuzz=0.000, resolution=0.000
      TOOL_MINOR: source=0x00001002, min=0.000, max=2264.156, flat=0.000, fuzz=0.000, resolution=0.000
    Touch Input Mapper (mode - DIRECT):
      Parameters:
        GestureMode: MULTI_TOUCH
        DeviceType: TOUCH_SCREEN
        AssociatedDisplay: hasAssociatedDisplay=true, isExternal=false, displayId='local:1'
        OrientationAware: true
        Orientation: ORIENTATION_0
      Raw Touch Axes:
        X: min=0, max=800, flat=0, fuzz=0, resolution=0
        Y: min=0, max=1280, flat=0, fuzz=0, resolution=0
        Pressure: unknown range
        TouchMajor: min=0, max=255, flat=0, fuzz=0, resolution=0
        TouchMinor: unknown range
        ToolMajor: min=0, max=255, flat=0, fuzz=0, resolution=0
        ToolMinor: unknown range
        Orientation: unknown range
        Distance: unknown range
        TiltX: unknown range
        TiltY: unknown range
        TrackingId: min=0, max=255, flat=0, fuzz=0, resolution=0
        Slot: unknown range
      Calibration:
        touch.size.calibration: geometric
        touch.pressure.calibration: none
        touch.orientation.calibration: none
        touch.distance.calibration: none
        touch.coverage.calibration: none
      Affine Transformation:
        X scale: 1.000
        X ymix: 0.000
        X offset: 0.000
        Y xmix: 0.000
        Y scale: 1.000
        Y offset: 0.000
      Viewport EXTERNAL: displayId=0, uniqueId=local:1, port=1, orientation=0, logicalFrame=[0, 0, 1080, 1920], physicalFrame=[40,
      RawSurfaceWidth: 1280px
```

Figure 3.4-8 Confirm the main screen TP and the secondary screen TP.

After confirming there are no errors, execute the following command. On the secondary screen, launch the Android system settings application to test if the dual-screen touch interaction is normal. Please conduct the test yourself:

```
am start --display 2 com.android.settings/.Settings
```

--display option is used to specify the displayId, and to launch the application on the specified screen. How to query the displayId of a screen? You can refer to Section 3.4.3. In this example, the displayId of the secondary screen is 2.

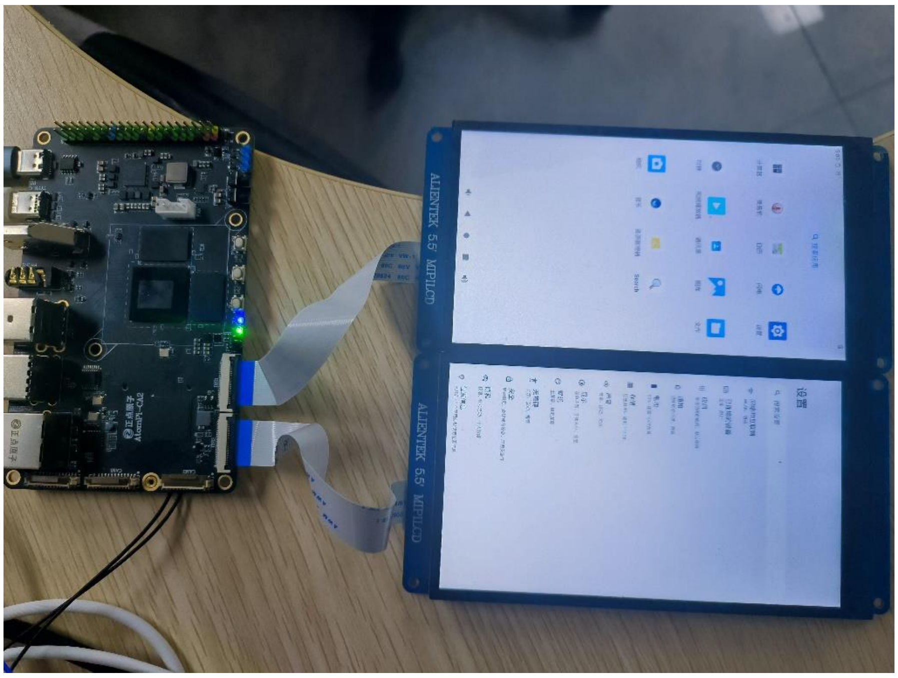
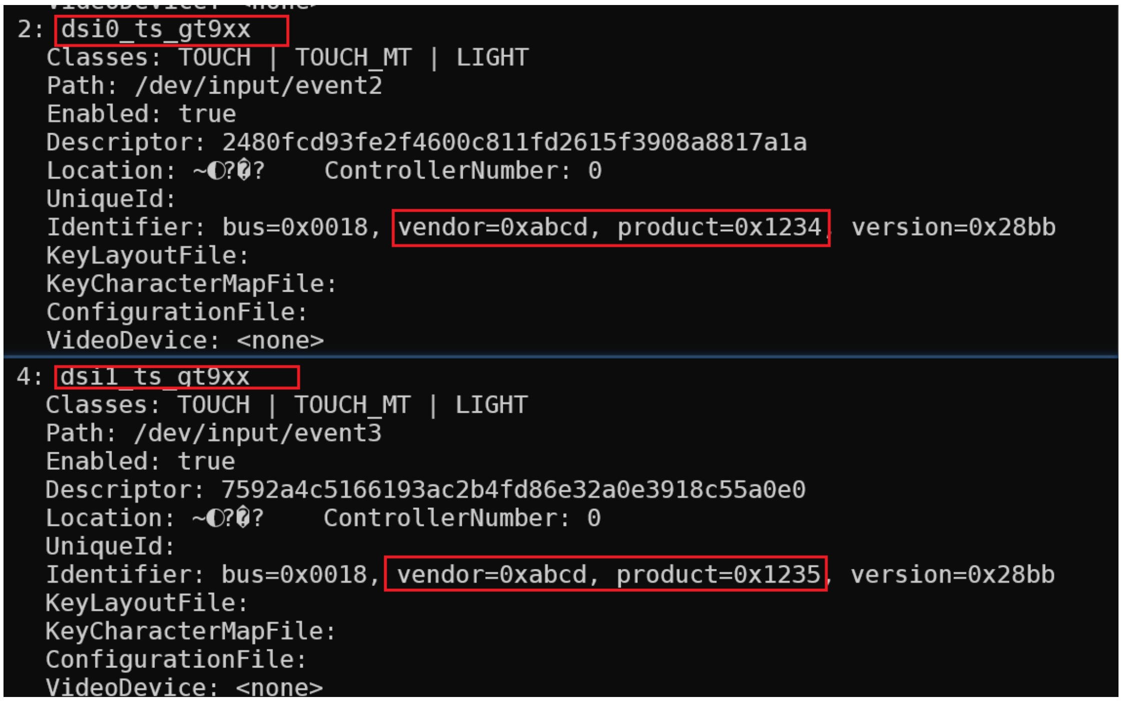


Figure 3.4-9 Dual-screen touch test

3.4.2 Query the pid and vid of the input device

The pid and vid of the input device can be queried using the "dumpsys input" command. On the QuarkPi-CA2 Android system, execute the "dumpsys input" command. This command will print out the status and configuration information of all input devices, with a large amount of output information. It is necessary to search for it yourself. Taking the MIPI screen touch input device as an example, as shown below (only a part is captured):

```
dumpsys input
```



```
2: dsi0_ts_gt9xx
   Classes: TOUCH | TOUCH_MT | LIGHT
   Path: /dev/input/event2
   Enabled: true
   Descriptor: 2480fcd93fe2f4600c811fd2615f3908a8817a1a
   Location: ~◐?�?    ControllerNumber: 0
   UniqueId:
   Identifier: bus=0x0018, vendor=0xabcd, product=0x1234, version=0x28bb
   KeyLayoutFile:
   KeyCharacterMapFile:
   ConfigurationFile:
   VideoDevice: <none>
4: dsi1_ts_gt9xx
   Classes: TOUCH | TOUCH_MT | LIGHT
   Path: /dev/input/event3
   Enabled: true
   Descriptor: 7592a4c5166193ac2b4fd86e32a0e3918c55a0e0
   Location: ~◐?�?    ControllerNumber: 0
   UniqueId:
   Identifier: bus=0x0018, vendor=0xabcd, product=0x1235, version=0x28bb
   KeyLayoutFile:
   KeyCharacterMapFile:
   ConfigurationFile:
   VideoDevice: <none>
```

Figure 3.4-10 Query the PID and VID of the input device

The name of the touch input device corresponding to MIPI DSI0 is dsi0_ts_gt9xx, with PID and VID being 1234 and abcd respectively;

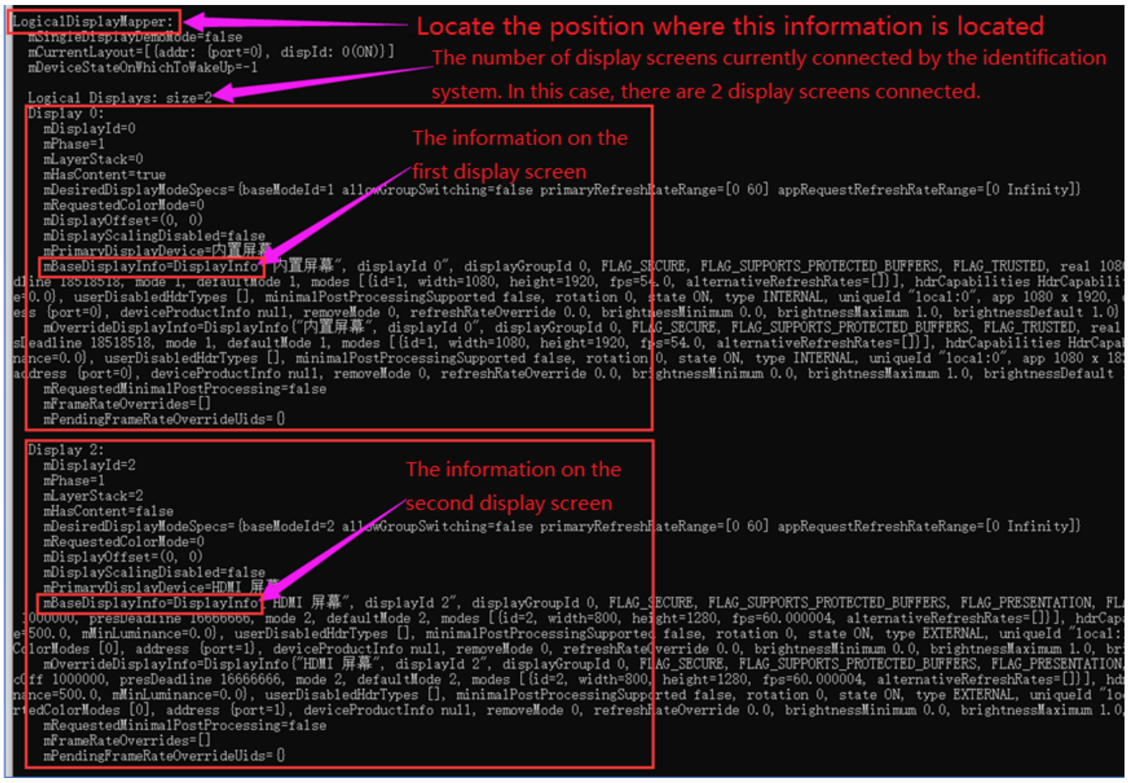
The name of the touch input device corresponding to MIPI DSI1 is dsi1_ts_gt9xx, with PID and VID being 1235 and abcd respectively.

These pieces of information are pre-configured in our device tree and users can also define them themselves.

3.4.3 Query the uniqueId and displayId of the screen

The uniqueId is the unique identification ID of the display device, and the displayId is the display ID of the screen (by specifying the displayId, Android applications can be launched and displayed on the specified screen). On the QuarkPi-CA2 Android system, the command "dumpsys display" can be executed to query the uniqueId and displayId of the display screen. This command outputs a lot of information, and we need to find the following contents from these pieces of information:

```
dumpsys display
```


Locate the position where this information is located
The number of display screens currently connected by the identification system. In this case, there are 2 display screens connected.
The information on the first display screen
The information on the second display screen

```
LogicalDisplayMapper:
  mSingleDisplayDemoMode=false
  mCurrentLayout=[{addr: {port=0}, dispId: 0(ON)}]
  mDeviceStateOnWhichToWakeUp=-1

  Logical Displays: size=2
  Display 0:
    mDisplayId=0
    mPhase=1
    mLayerStack=0
    mHasContent=true
    mDesiredDisplayModeSpecs={baseModeId=1 allowGroupSwitching=false primaryRefreshRateRange=[0 60] appRequestRefreshRateRange=[0 Infinity]}
    mRequestedColorMode=0
    mDisplayOffset=(0, 0)
    mDisplayScalingDisabled=false
    mPrimaryDisplayDevice=内置屏幕
    mBaseDisplayInfo=DisplayInfo{"内置屏幕", displayId 0", displayGroupId 0, FLAG_SECURE, FLAG_SUPPORTS_PROTECTED_BUFFERS, FLAG_TRUSTED, real 108
dline 18518518, mode 1, defaultMode 1, modes [{id=1, width=1080, height=1920, fps=54.0, alternativeRefreshRates=[]}], hdrCapabilities HdrCapabili
e=0.0}, userDisabledHdrTypes [], minimalPostProcessingSupported false, rotation 0, state ON, type INTERNAL, uniqueId "local:0", app 1080 x 1920, 
ess {port=0}, deviceProductInfo null, removeMode 0, refreshRateOverride 0.0, brightnessMinimum 0.0, brightnessMaximum 1.0, brightnessDefault 1.0}
    mOverrideDisplayInfo=DisplayInfo{"内置屏幕", displayId 0", displayGroupId 0, FLAG_SECURE, FLAG_SUPPORTS_PROTECTED_BUFFERS, FLAG_TRUSTED, real
sDeadline 18518518, mode 1, defaultMode 1, modes [{id=1, width=1080, height=1920, fps=54.0, alternativeRefreshRates=[]}], hdrCapabilities HdrCapa
nance=0.0}, userDisabledHdrTypes [], minimalPostProcessingSupported false, rotation 0, state ON, type INTERNAL, uniqueId "local:0", app 1080 x 18
address {port=0}, deviceProductInfo null, removeMode 0, refreshRateOverride 0.0, brightnessMinimum 0.0, brightnessMaximum 1.0, brightnessDefault 
    mRequestedMinimalPostProcessing=false
    mFrameRateOverrides=[]
    mPendingFrameRateOverrideUids={}

  Display 2:
    mDisplayId=2
    mPhase=1
    mLayerStack=2
    mHasContent=false
    mDesiredDisplayModeSpecs={baseModeId=2 allowGroupSwitching=false primaryRefreshRateRange=[0 60] appRequestRefreshRateRange=[0 Infinity]}
    mRequestedColorMode=0
    mDisplayOffset=(0, 0)
    mDisplayScalingDisabled=false
    mPrimaryDisplayDevice=HDMI 屏幕
    mBaseDisplayInfo=DisplayInfo{"HDMI 屏幕", displayId 2", displayGroupId 0, FLAG_SECURE, FLAG_SUPPORTS_PROTECTED_BUFFERS, FLAG_PRESENTATION, FL
1000000, presDeadline 16666666, mode 2, defaultMode 2, modes [{id=2, width=800, height=1280, fps=60.000004, alternativeRefreshRates=[]}], hdrCap
e=500.0, mMinLuminance=0.0}, userDisabledHdrTypes [], minimalPostProcessingSupported false, rotation 0, state ON, type EXTERNAL, uniqueId "local:
ColorModes [0], address {port=1}, deviceProductInfo null, removeMode 0, refreshRateOverride 0.0, brightnessMinimum 0.0, brightnessMaximum 1.0, br
    mOverrideDisplayInfo=DisplayInfo{"HDMI 屏幕", displayId 2", displayGroupId 0, FLAG_SECURE, FLAG_SUPPORTS_PROTECTED_BUFFERS, FLAG_PRESENTATION,
cOff 1000000, presDeadline 16666666, mode 2, defaultMode 2, modes [{id=2, width=800, height=1280, fps=60.000004, alternativeRefreshRates=[]}], hd
nance=500.0, mMinLuminance=0.0}, userDisabledHdrTypes [], minimalPostProcessingSupported false, rotation 0, state ON, type EXTERNAL, uniqueId "lo
rtedColorModes [0], address {port=1}, deviceProductInfo null, removeMode 0, refreshRateOverride 0.0, brightnessMinimum 0.0, brightnessMaximum 1.0,
    mRequestedMinimalPostProcessing=false
    mFrameRateOverrides=[]
    mPendingFrameRateOverrideUids={}
```

Figure 3.4-11 Query the uniqueId and displayId.

You can also use the following command for filtering:

```
dumpsys display | grep mBaseDisplayInfo
```

Received the following information:

```
# In this example, the system is connected to two display screens. The information of these screens is as follows:

# Information of the First Display Screen
mBaseDisplayInfo=DisplayInfo{"内置屏幕", displayId 0", displayGroupId 0, FLAG_SECURE, FLAG_SUPPORTS_PROTECTED_BUFFERS, FLAG_TRUSTED, real 1080 x 1920, largest app 1080 x 1920, smallest app 1080 x 1920, appVsyncOff 1000000, presDeadline 18518518, mode 1, defaultMode 1, modes [{id=1, width=1080, height=1920, fps=54.0, alternativeRe-freshRates=[]}], hdrCapabilities HdrCapabilities{mSupportedHdrTypes=[], mMaxLumi-nance=500.0, mMaxAverageLuminance=500.0, mMinLuminance=0.0}, userDisabledHdrTypes [], minimalPostProcessingSupported false, rotation 0, state ON, type INTERNAL, uniqueId "local:0", app 1080 x 1920, density 320 (320.0 x 320.0) dpi, layerStack 0, colorMode 0, supportedColorModes [0], address {port=0}, deviceProductInfo null, removeMode 0, refreshRateOverride 0.0, brightnessMinimum 0.0, brightnessMaximum 1.0, bright-nessDefault 1.0}


# The information on the second display screen
mBaseDisplayInfo=DisplayInfo{"HDMI 屏 幕 ", displayId 2", displayGroupId 0, FLAG_SECURE, FLAG_SUPPORTS_PROTECTED_BUFFERS, FLAG_PRESENTATION, FLAG_TRUSTED, real 800 x 1280, largest app 800 x 1280, smallest app 800 x 1280, appVsyncOff
1000000, presDeadline 16666666, mode 2, defaultMode 2, modes [{id=2, width=1080, height=1920, fps=54.0, alternativeRefreshRates=[]}], hdrCapabilities HdrCapabilities{mSupportedHdrTypes=[], mMaxLuminance=500.0, mMaxAverageLuminance=500.0, mMinLuminance=0.0}, userDisabledHdrTypes [], minimalPostProcessingSupported false, rotation 0, state ON, type EXTERNAL, uniqueId "local:1", app 800 x 1280, density 213 (320.0 x 320.0) dpi, layerStack 2, colorMode 0, supportedColorModes [0], address {port=1}, deviceProductInfo null, removeMode 0, refreshRateOverride 0.0, brightnessMinimum 0.0, brightnessMaximum 1.0, brightnessDefault 0.5}
```

From the above information, it can be seen that the displayId of the first display is **0**, with a resolution of **1080x1920** and a refresh rate of **54 fps**, and the **uniqueId** is "**local:0**"; the displayId of the second display is **2**, with a resolution of **1080x1920** and a refresh rate of **54 fps**, and the uniqueId is "local:1".

TIPS: The "built-in screen" corresponds to the main screen, while the "HDMI screen" corresponds to the secondary screen (note, this does not mean that the secondary screen is an HDMI display. If the secondary screen is a MIPI screen, it will still be displayed as "HDMI screen" in the ANDROID framework. This is just a "name" given by the ANDROID framework for the main and secondary screens).

3.5 Multi-screen Display

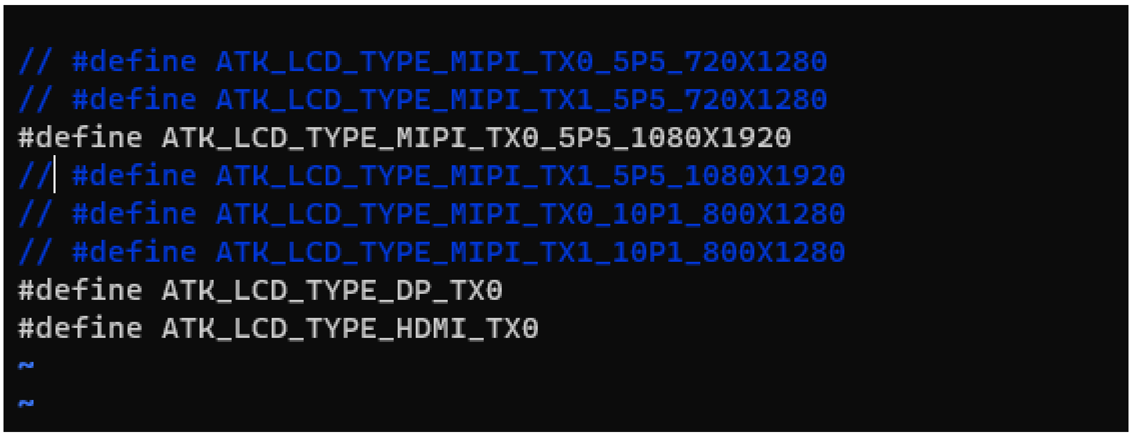
When there are multiple logical screens (corresponding to actual physical screens) in the Android system, the default state is multi-screen display; all screens display the same content. The ALIENTEK QuarkPi-CA2 card computer has 1 HDMI display interface, 1 full-function TYPE-C interface (supporting DP display), and 2 MIPI screen interfaces, enabling 4-screen display.

Take the multi-screen display of the ALIENTEK QuarkPi-CA2 card computer as an example, and introduce how to configure the following situations.

3.5.1 MIPI as the main screen, HDMI/DP as the secondary screen

Open the Linux kernel device tree file kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi, enable the MIPI DSI0 (taking the 5.5-inch 1080x1920 MIPI screen as an example), HDMI TX0, and DP TX0 display interfaces, as shown below:

```
vi kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi
```



```
// #define ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_720X1280
#define ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280
// #define ATK_LCD_TYPE_MIPI_TX1_10P1_800X1280
#define ATK_LCD_TYPE_DP_TX0
#define ATK_LCD_TYPE_HDMI_TX0
~
~
```

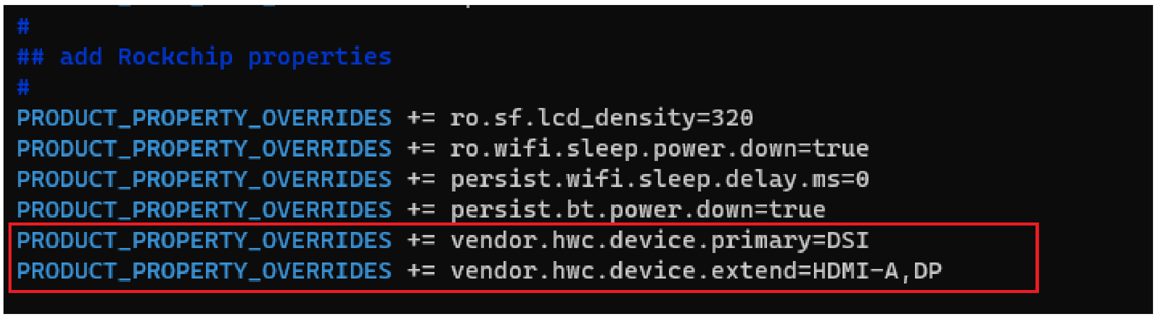
Figure 3.5-1 Display interface configuration

After modification, save and exit!

Open the file "device/rockchip/rk3588/atk_quarkpi_ca2/atk_quarkpi_ca2.mk", and configure the main screen as a MIPI screen and the secondary screen as an HDMI and DP screen, as follows:

```
vim device/rockchip/rk3588/atk_quarkpi_ca2/atk_quarkpi_ca2.mk
```

```
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=DSI
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=HDMI-A,DP
```



```
#
## add Rockchip properties
#
PRODUCT_PROPERTY_OVERRIDES += ro.sf.lcd_density=320
PRODUCT_PROPERTY_OVERRIDES += ro.wifi.sleep.power.down=true
PRODUCT_PROPERTY_OVERRIDES += persist.wifi.sleep.delay.ms=0
PRODUCT_PROPERTY_OVERRIDES += persist.bt.power.down=true
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=DSI
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=HDMI-A,DP
```

Figure 3.5-2 Configure the main and secondary screens

After the modification is completed, save and exit!

Execute the following command in the root directory of the SDK to recompile the Linux kernel and Android source code:

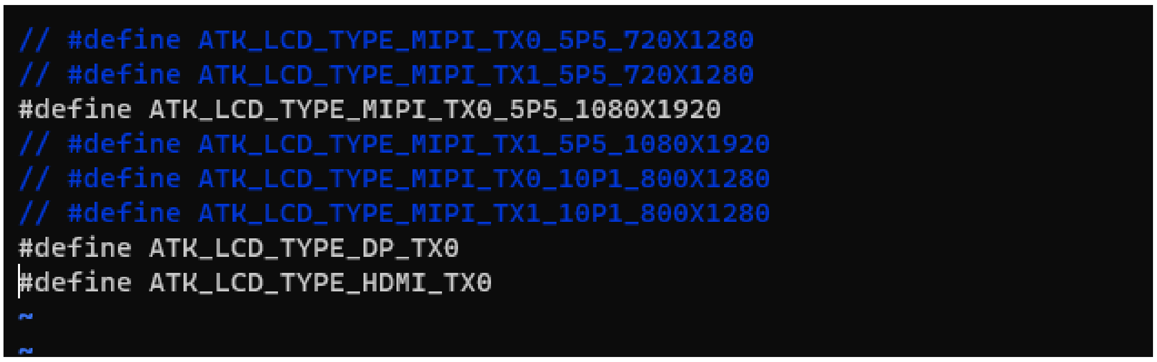
```
source build/envsetup.sh
lunch atk_quarkpi_ca2-userdebug
./build.sh -KA -J10
```

Burn the compiled super.img and boot.img files onto the development board. After the burning process is completed, power off the development board and connect the MIPI screen, HDMI display, and DP display to the MIPI DSI0, HDMI TX0, and TYPE-C0 interfaces of the development board (using a type-C to DP cable for connection). Then power on the development board to start the test.

3.5.2 Main screen using HDMI, secondary screen using MIPI/DP

Open the Linux kernel device tree file kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi, enable the MIPI DSI0 (taking a 10.1-inch 800x1280 MIPI screen as an example), HDMI TX0, and DP TX0 these three display interfaces, as shown below:

```
vi kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi
```



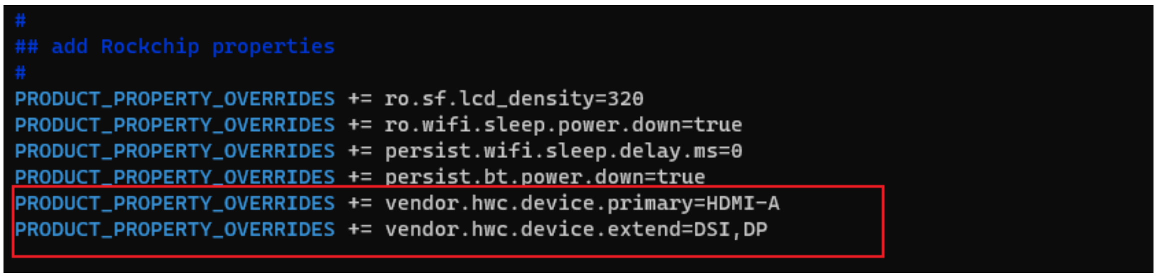
```
// #define ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_720X1280
#define ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280
// #define ATK_LCD_TYPE_MIPI_TX1_10P1_800X1280
#define ATK_LCD_TYPE_DP_TX0
#define ATK_LCD_TYPE_HDMI_TX0
~
~
```

Figure 3.5-3 Display interface configuration

After modification, save and exit!

Open the file device/rockchip/rk3588/atk_quarkpi_ca2/atk_quarkpi_ca2.mk, and configure the main screen as HDMI screen, and the secondary screen as MIPI and DP screen, as follows:

```
vi device/rockchip/rk3588/atk_quarkpi_ca2/atk_quarkpi_ca2.mk
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=HDMI-A
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=DSI,DP
```



```
#
## add Rockchip properties
#
PRODUCT_PROPERTY_OVERRIDES += ro.sf.lcd_density=320
PRODUCT_PROPERTY_OVERRIDES += ro.wifi.sleep.power.down=true
PRODUCT_PROPERTY_OVERRIDES += persist.wifi.sleep.delay.ms=0
PRODUCT_PROPERTY_OVERRIDES += persist.bt.power.down=true
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=HDMI-A
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=DSI,DP
```

Figure 3.5-4 Configure the main and secondary screens

After the modification is completed, save and exit!

Execute the following command in the root directory of the SDK to recompile the Linux kernel and Android source code:

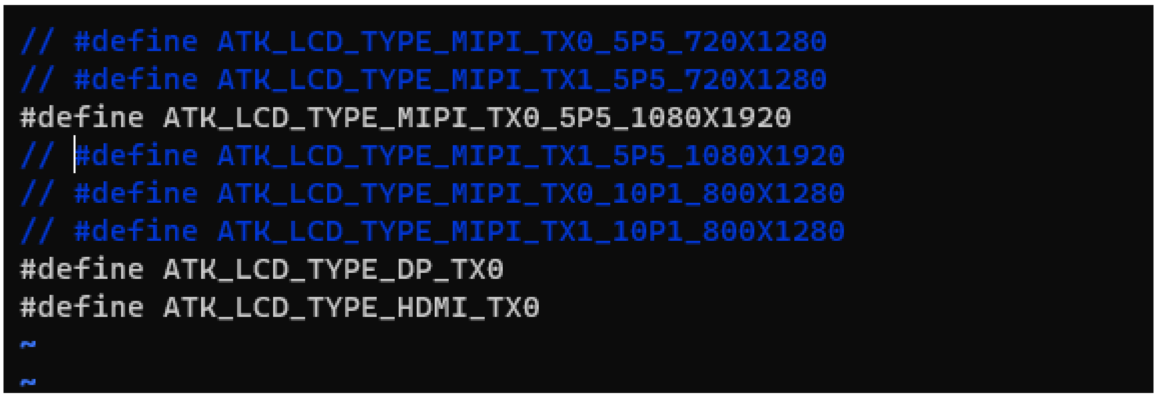
```
source build/envsetup.sh
lunch atk_quarkpi_ca2-userdebug
./build.sh -KA -J10
```

Burn the compiled super.img and boot.img onto the development board. After the burning process is completed, power off the development board and connect the MIPI screen, HDMI display, and DP display to the MIPI DSI0, HDMI TX0, and TYPE-C0 interfaces of the development board (using a type-C to DP cable for connection). Then power on the development board to start the test.

3.5.3 DP as the main screen, MIPI/HDMI as the secondary screens

Open the Linux kernel device tree file kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi, enable the MIPI DSI0 (taking the 10.1-inch 800x1280 MIPI screen as an example), HDMI TX0, and DP TX0 display interfaces as follows:

```
vi kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi
```



```
// #define ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_720X1280
#define ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280
// #define ATK_LCD_TYPE_MIPI_TX1_10P1_800X1280
#define ATK_LCD_TYPE_DP_TX0
#define ATK_LCD_TYPE_HDMI_TX0
~
~
```

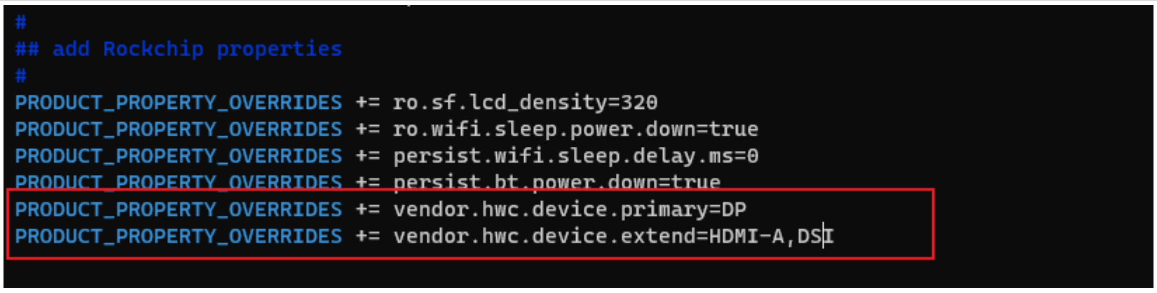
Figure 3.5-5 Display interface configuration

After modification, save and exit!

Open the file device/rockchip/rk3588/atk_quarkpi_ca2/atk_quarkpi_ca2.mk, configure the main screen as a DP screen, and the secondary screens as HDMI and MIPI screens, as follows:

```
vi device/rockchip/rk3588/atk_quarkpi_ca2/atk_quarkpi_ca2.mk
```

```
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=DP
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=DSI,HDMI-A
```



```
#
## add Rockchip properties
#
PRODUCT_PROPERTY_OVERRIDES += ro.sf.lcd_density=320
PRODUCT_PROPERTY_OVERRIDES += ro.wifi.sleep.power.down=true
PRODUCT_PROPERTY_OVERRIDES += persist.wifi.sleep.delay.ms=0
PRODUCT_PROPERTY_OVERRIDES += persist.bt.power.down=true
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=DP
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=HDMI-A,DSI
```

Figure 3.5-6 Configure the main and secondary screens

After the modification is completed, save and exit!

Execute the following command in the root directory of the SDK to recompile the Linux kernel and Android source code:

```
source build/envsetup.sh
lunch atk_quarkpi_ca2-userdebug
./build.sh -KA -J10
```

Burn the compiled super.img and boot.img onto the development board. After the burning process is completed, power off the development board and connect the MIPI screen, HDMI display, and DP display to the MIPI DSI0, HDMI TX, and TYPE-C interfaces of the development board (using a type-C to DP cable for connection). Then power on the development board to start the test.

3.5.4 Dual MIPI Screen Display

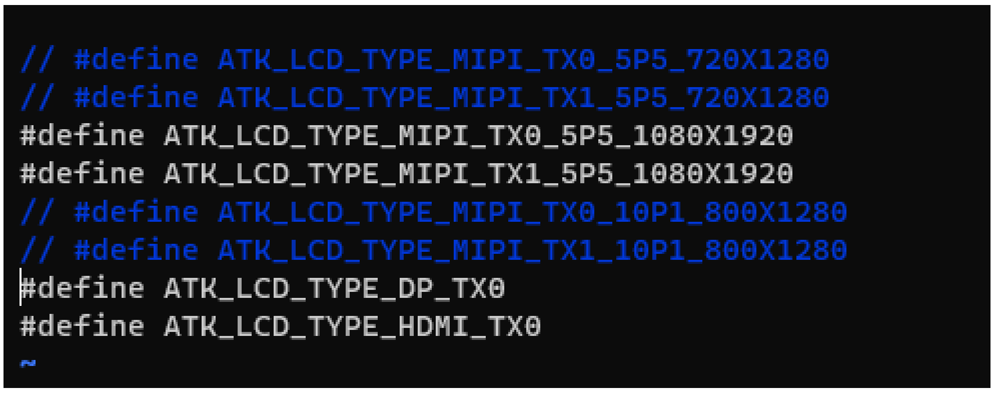
Refer to Section 3.4.1.

3.5.5 Quad-screen Display: Dual MIPI + HDMI + DP

The RK3588S platform has 4 independent VP (VP0/VP1/VP2/VP3) circuits, enabling 4 independent display outputs. It supports up to four screens with independent display. This section will illustrate with the example of dual MIPI + DP + HDMI. In this case, the MIPI screen will be used as the main screen.

Open the Linux kernel device tree file kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi and enable the MIPI DSI0 (for a 5.5-inch 1080x1920 MIPI screen), MIPI DSI1 (for a 5.5-inch 1080x1920 MIPI screen), HDMI TX, and DP TX display interfaces as follows:

```
vi kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi
```



```
// #define ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_720X1280
#define ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920
#define ATK_LCD_TYPE_MIPI_TX1_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280
// #define ATK_LCD_TYPE_MIPI_TX1_10P1_800X1280
#define ATK_LCD_TYPE_DP_TX0
#define ATK_LCD_TYPE_HDMI_TX0
~
```

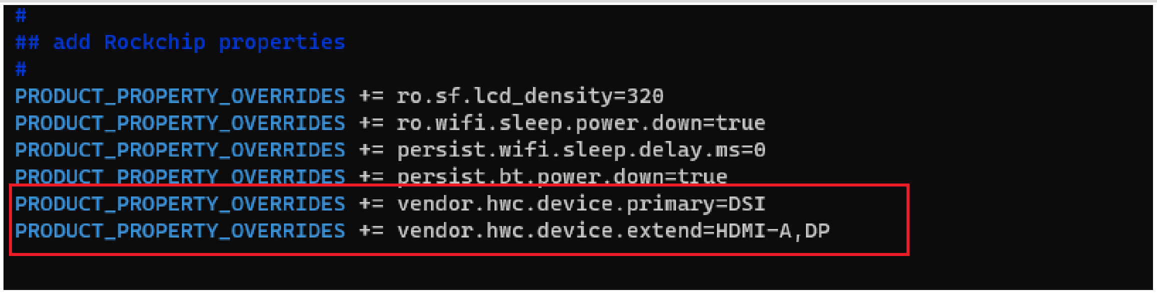
Figure 3.5-7 Display interface configuration

After modification, save and exit!

Open the file "device/rockchip/rk3588/atk_quarkpi_ca2/atk_quarkpi_ca2.mk", and configure the main screen as a MIPI screen and the secondary screen as an HDMI and DP screen, as follows:

```
vi device/rockchip/rk3588/atk_quarkpi_ca2/atk_quarkpi_ca2.mk
```

```
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=DSI
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=HDMI-A,DP
```



```
#
## add Rockchip properties
#
PRODUCT_PROPERTY_OVERRIDES += ro.sf.lcd_density=320
PRODUCT_PROPERTY_OVERRIDES += ro.wifi.sleep.power.down=true
PRODUCT_PROPERTY_OVERRIDES += persist.wifi.sleep.delay.ms=0
PRODUCT_PROPERTY_OVERRIDES += persist.bt.power.down=true
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=DSI
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=HDMI-A,DP
```

Figure 3.5-8 Configure the main and secondary screens

After the modification is completed, save and exit!

Execute the following command in the root directory of the SDK to recompile the Linux kernel and Android source code:

```
source build/envsetup.sh
lunch atk_quarkpi_ca2-userdebug
./build.sh -KA -J10
```

Burn the compiled super.img and boot.img files onto the development board. After the burning process is completed, power off the development board and connect the MIPI screen to the MIPI DSI1 and MIPI DSI2 interfaces of the development board, connect the DP display to the TYPE-C interface of the development board (using a type-c to DP cable), and connect the HDMI display to the HDMI TX interface of the development board.

Then power on the development board to start the test.

3.6 Multi-screen Different Display

3.1.3 Section introduced the concept of multi-screen different display. In the Android system, the main and secondary screens are in the same display state by default. Some methods are needed to achieve different display. Currently, there are two different display schemes: Android Presentation and Android Activity specifying screen startup.

3.6.1 Introduction to Two Different Display Schemes

In Android Presentation, the corresponding interface needs to be called during APP development to make the specified view (the Presentation view is a special type of dialog view) display on the secondary screen.

In Android Activity, the screen is specified to be launched, and the APP can directly display the Activity on the corresponding screen using the displayId parameter.

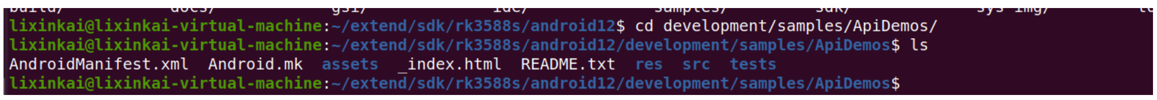
The main differences between the two are as follows:

1. The former's Activity requires independent development. The content to be displayed is projected onto the secondary screen. The latter does not require source code and can use command-line or system interfaces to project the Activity of a third-party APP onto the secondary screen;

2. The former is just an Activity on the top layer, and the specified content is displayed on the secondary screen through a special dialog. The latter has two Activities displaying on the main and secondary screens respectively.

3.6.2 Android Presentation

There are already demo programs using this interface in the SDK. The source code of this demo can be found in the path <SDK>/development/samples/ApiDemos/. As shown below:



```
lixinkai@lixinkai-virtual-machine:~/extend/sdk/rk3588s/android12$ cd development/samples/ApiDemos/
lixinkai@lixinkai-virtual-machine:~/extend/sdk/rk3588s/android12/development/samples/ApiDemos$ ls
AndroidManifest.xml  Android.mk  assets  _index.html  README.txt  res  src  tests
lixinkai@lixinkai-virtual-machine:~/extend/sdk/rk3588s/android12/development/samples/ApiDemos$
```

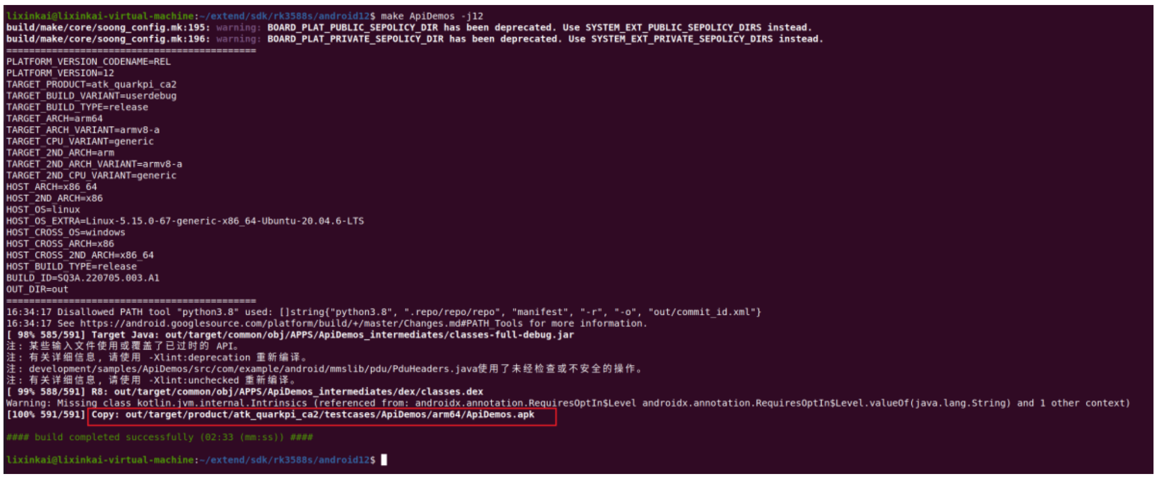
Figure 3.6-1 ApiDemes source code

The specific code used by the Android Presentation interface is located at the following path:

```
<SDK>/development/samples/ApiDemos/src/com/example/android/apis/app/PresentationActivity.java
```

Execute the following command in the SDK root directory to compile the ApiDemos.apk:

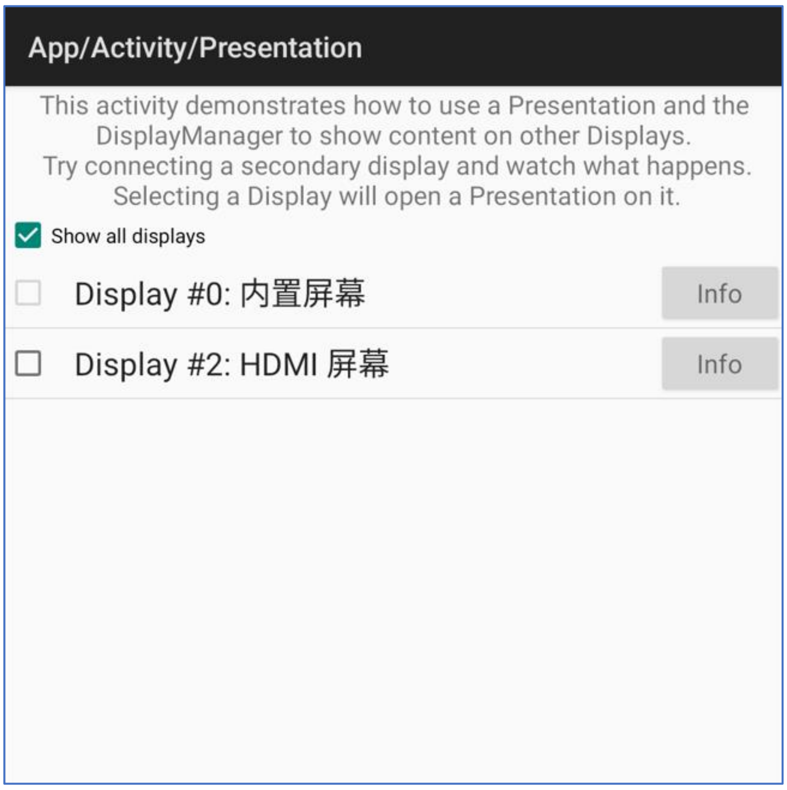
```
source build/envsetup.sh
lunch atk_quarkpi_ca2-userdebug
make ApiDemos -j12
```



```
lixinkai@lixinkai-virtual-machine:~/extend/sdk/rk3588s/android12$ make ApiDemos -j12
build/make/core/soong_config.mk:195: warning: BOARD_PLAT_PUBLIC_SEPOLICY_DIR has been deprecated. Use SYSTEM_EXT_PUBLIC_SEPOLICY_DIRS instead.
build/make/core/soong_config.mk:196: warning: BOARD_PLAT_PRIVATE_SEPOLICY_DIR has been deprecated. Use SYSTEM_EXT_PRIVATE_SEPOLICY_DIRS instead.
============================================
PLATFORM_VERSION_CODENAME=REL
PLATFORM_VERSION=12
TARGET_PRODUCT=atk_quarkpi_ca2
TARGET_BUILD_VARIANT=userdebug
TARGET_BUILD_TYPE=release
TARGET_ARCH=arm64
TARGET_ARCH_VARIANT=armv8-a
TARGET_CPU_VARIANT=generic
TARGET_2ND_ARCH=arm
TARGET_2ND_ARCH_VARIANT=armv8-a
TARGET_2ND_CPU_VARIANT=generic
HOST_ARCH=x86_64
HOST_2ND_ARCH=x86
HOST_OS=linux
HOST_OS_EXTRA=Linux-5.15.0-67-generic-x86_64-Ubuntu-20.04.6-LTS
HOST_CROSS_OS=windows
HOST_CROSS_ARCH=x86
HOST_CROSS_2ND_ARCH=x86_64
HOST_BUILD_TYPE=release
BUILD_ID=SQ3A.220705.003.A1
OUT_DIR=out
============================================
16:34:17 Disallowed PATH tool "python3.8" used: []string{"python3.8", ".repo/repo/repo", "manifest", "-r", "-o", "out/commit_id.xml"}
16:34:17 See https://android.googlesource.com/platform/build/+/master/Changes.md#PATH_Tools for more information.
[ 98% 585/591] Target Java: out/target/common/obj/APPS/ApiDemos_intermediates/classes-full-debug.jar
注：某些输入文件使用或覆盖了已过时的 API。
注：有关详细信息，请使用 -Xlint:deprecation 重新编译。
注：development/samples/ApiDemos/src/com/example/android/mmslib/pdu/PduHeaders.java使用了未经检查或不安全的操作。
注：有关详细信息，请使用 -Xlint:unchecked 重新编译。
[ 99% 588/591] R8: out/target/common/obj/APPS/ApiDemos_intermediates/dex/classes.dex
Warning: Missing class kotlin.jvm.internal.Intrinsics (referenced from: androidx.annotation.RequiresOptIn$Level androidx.annotation.RequiresOptIn$Level.valueOf(java.lang.String) and 1 other context)
[100% 591/591] Copy: out/target/product/atk_quarkpi_ca2/testcases/ApiDemos/arm64/ApiDemos.apk

#### build completed successfully (02:33 (mm:ss)) ####

lixinkai@lixinkai-virtual-machine:~/extend/sdk/rk3588s/android12$
```

Figure 3.6-2 Compile ApiDemos.apk

Use adb to install ApiDemos.apk onto the RK3588S Android system. After installing the APK, click to run the APP. Then, click the "App -> Activity -> Presentation" options one by one, and you can enter the Presentation invocation interface, as shown in the following figure:



App/Activity/Presentation

This activity demonstrates how to use a Presentation and the DisplayManager to show content on other Displays.
Try connecting a secondary display and watch what happens.
Selecting a Display will open a Presentation on it.

Figure 3.6-3 Presentation Test

Checking the "Show all displays" option will list all screens including the main screen and the secondary screen. Unchecking it will only list the secondary screen. "Built-in screen" refers to the main screen, while "HDMI screen" refers to the secondary screen (note: the secondary screen will always be labeled as an HDMI screen, but this does not mean it is necessarily an HDMI display).

Checking "Display #2: HDMI screen" will enable the display of the corresponding image on the secondary screen, as shown below:

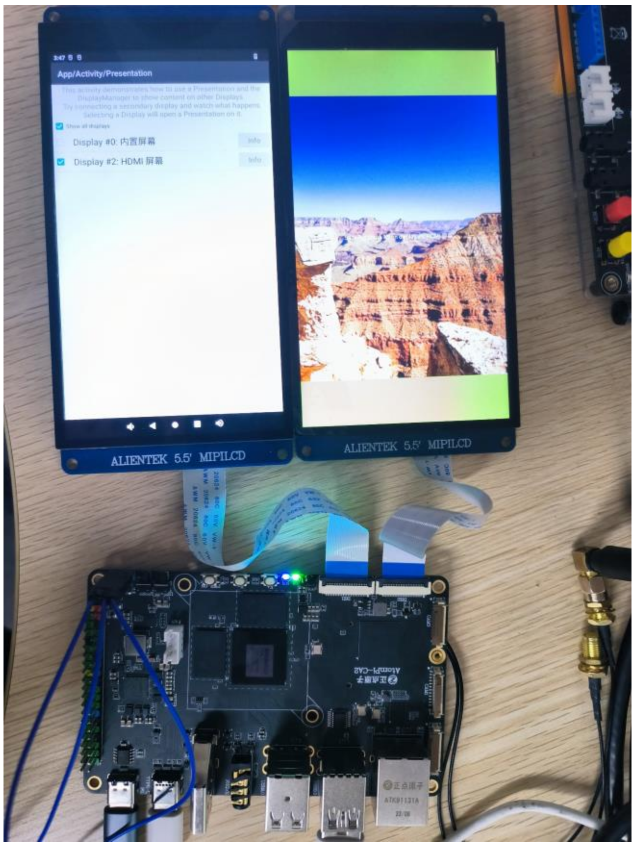


Figure 3.6-4 Secondary screen display

For the specific code, please study the source file PresentationActivity.java by yourself.

3.6.3 Android Activity specifying screen launch

This section will explain how to make the APP directly start and display on a specific screen by invoking the command line. The invocation method is as follows:

```
adb shell am start --display <displayId> <activity_name>
```

displayId represents the display ID of the screen. This has been discussed in Section 3.4.3. No further explanation is necessary! Also, please refer to Section 3.4.3 for information on how to query displayId. The activity_name parameter specifies the Activity to be launched. For example, to launch the Android system settings application, the following code should be used:

```
adb shell am start --display 2 com.android.settings/.Settings
```

Start the clock application:

```
adb shell am start --display 2 com.android.deskclock
```

Launch the gallery application:

```
adb shell am start --display 2 com.android.gallery3d
```

3.6.4 Mouse Primary and Secondary Screen Switching

In the dual-display mode, if the development board is connected to a mouse, by default, the mouse will only display on the primary screen and cannot switch to the secondary screen. At this time, we can set the sys.mouse.presentation property to 1 to enable the mouse primary and secondary screen switching display function:

```
adb shell setprop sys.mouse.presentation 1
```

After enabling this function, when the mouse moves to the edge of the screen, it will automatically switch to the center position of the secondary screen for display.

Chapter 4. Introduction to Multi-Display Function in Linux Buildroot System

This chapter introduces the related content of multi-display functionality in the Linux Buildroot system, including screen display direction rotation, multi-screen simultaneous display, multi-screen splicing display, and multi-screen touch, etc.

Reference document:

docs/cn/Linux/Graphics/Rockchip_Developer_Guide_Buildroot_Weston_CN.pdf

4.1 Multi-screen Display

The RK3588S Linux buildroot system uses Weston (a display server based on the Wayland protocol) as the display server to achieve the display and interaction functions of the graphical user interface (GUI).

Weston supports functions such as multi-screen simultaneous display and multi-screen concatenation. This section introduces how to configure the multi-screen simultaneous display function.

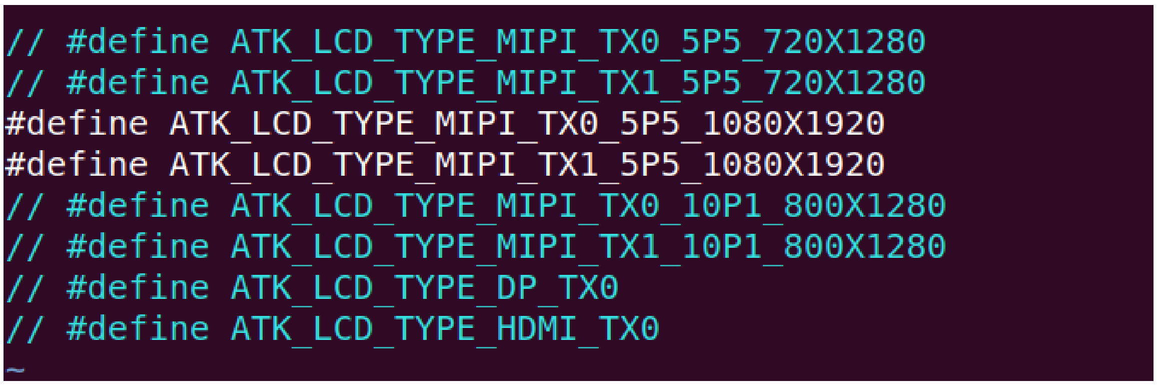
The ALIENTEK Technology QuarkPi-CA2 has 1 HDMI display interface, 1 full-function TYPE-C interface (supporting DP display), and 2 MIPI screen interfaces. It can achieve simultaneous display of up to 4 screens.

Taking the multi-screen simultaneous display of the ALIENTEK Technology QuarkPi-CA2 card computer as an example, this section introduces how to configure the following situations.

4.1.1 Dual MIPI Screen Simultaneous Display

Open the Linux kernel device tree file kernel/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi, enable the two MIPI DSI display interfaces (taking a 5.5-inch 1080x1920 MIPI screen as an example), as follows:

```
vi kernel/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi
```



```
// #define ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_720X1280
#define ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920
#define ATK_LCD_TYPE_MIPI_TX1_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280
// #define ATK_LCD_TYPE_MIPI_TX1_10P1_800X1280
// #define ATK_LCD_TYPE_DP_TX0
// #define ATK_LCD_TYPE_HDMI_TX0
~
```

Figure 4.1-1 Display interface configuration

After modification, save and exit!

Execute the following command in the root directory of the SDK to compile the Linux kernel:

```
./build.sh alientek_quarkpi_ca2_defconfig
./build.sh kernel
```

Connect the two 5.5-inch 1080x1920 MIPI screens to the MIPI DSI0 and MIPI DSI1 interfaces of the development board respectively.

Burn the compiled boot.img onto the development board, and then start the development board to enter the Linux buildroot system, as shown below:

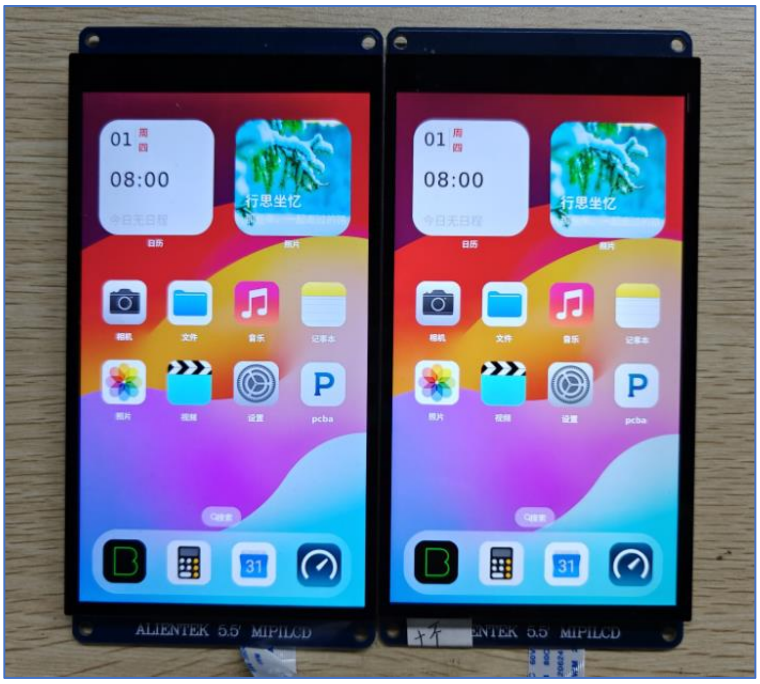


Figure 4.1-2 Dual MIPI screens display simultaneously

Since the default configuration of Weston is for multi-screen simultaneous display, it is possible to see that both MIPI screens display the same content.

4.1.2 Three-screen simultaneous display

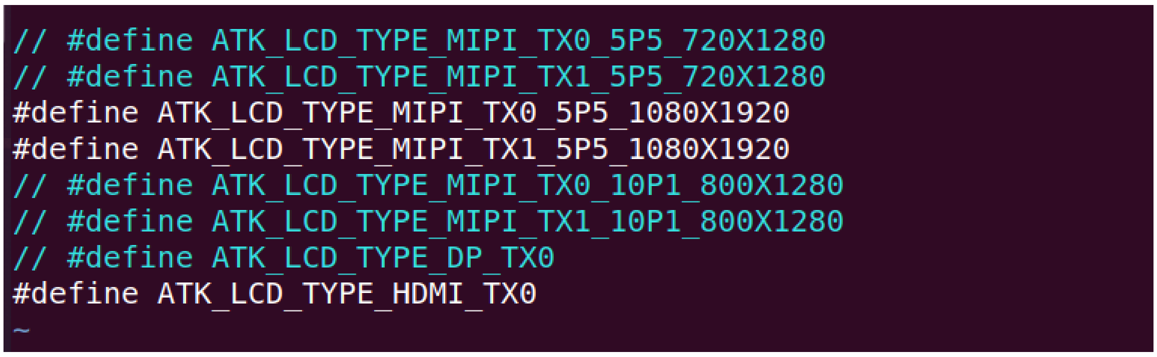
There are many schemes for three-screen simultaneous display:

- Dual MIPI + HDMI
- Dual MIPI + DP
- MIPI + HDMI + DP

It is impossible to introduce each one individually. This section will take the dual MIPI + HDMI as an example.

Open the Linux kernel device tree file kernel/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi, enable the MIPI DSI0, MIPI DSI1 and HDMI TX0 interfaces (taking a 5.5-inch 1080x1920 MIPI screen as an example), as shown below:

```
vi kernel/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi
```



```
// #define ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_720X1280
#define ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920
#define ATK_LCD_TYPE_MIPI_TX1_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280
// #define ATK_LCD_TYPE_MIPI_TX1_10P1_800X1280
// #define ATK_LCD_TYPE_DP_TX0
#define ATK_LCD_TYPE_HDMI_TX0
~
```

Figure 4.1-3 Display interface configuration

After modification, save and exit!

Open the "buildroot/package/weston/weston.sh" file and configure the main screen (or the main display)

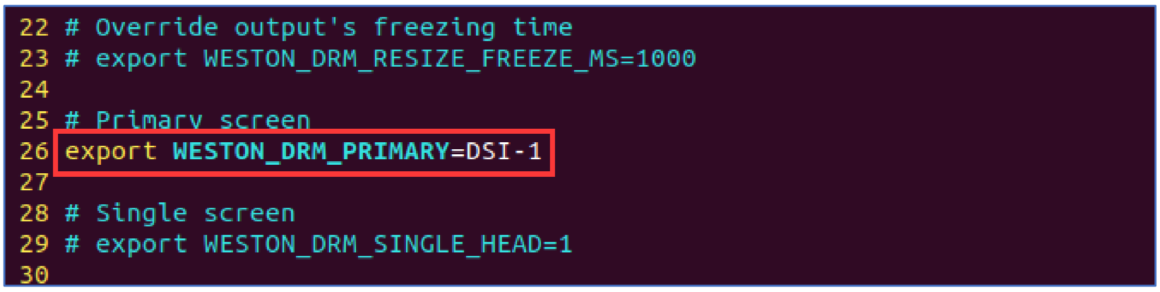
```
vi buildroot/package/weston/weston.sh
```

Find the following line:

```
# export WESTON_DRM_PRIMARY=eDP-1
```

Remove the "#" symbol at the beginning and open it. This variable is used to specify the primary screen. In this example, we set the primary screen to the MIPI screen:

```
export WESTON_DRM_PRIMARY=DSI-1
```



```
# Override output's freezing time
# export WESTON_DRM_RESIZE_FREEZE_MS=1000

# Primary screen
export WESTON_DRM_PRIMARY=DSI-1

# Single screen
# export WESTON_DRM_SINGLE_HEAD=1

```

Figure 4.1-4 Configure the main screen

DSI-1 is the connector corresponding to the MIPI screen. After modification, save and exit!

Tips: After compilation, the weston.sh script will be copied to the etc/profile.d/ directory of the target root file system. Therefore, you can modify the /etc/profile.d/weston.sh file during system operation to configure the Weston environment variables. After modification, save and exit, and restart the system for the new configuration to take effect.

Execute the following command in the SDK root directory to recompile the Linux kernel and the buildroot root file system (the entire SDK has been fully compiled before):

```
./build.sh alientek_quarkpi_ca2_defconfig
./build.sh kernel
./build.sh buildroot-make:weston-rebuild
./build.sh buildroot
```

Connect one HDMI display to the HDMI interface of the card computer, and connect two MIPI screens (in this case, a 5.5-inch 1080x1920 MIPI screen) to the MIPI DSI1 and MIPI DSI2 interfaces of the development board respectively.

Burn the compiled boot.img and rootfs.img onto the development board for testing.

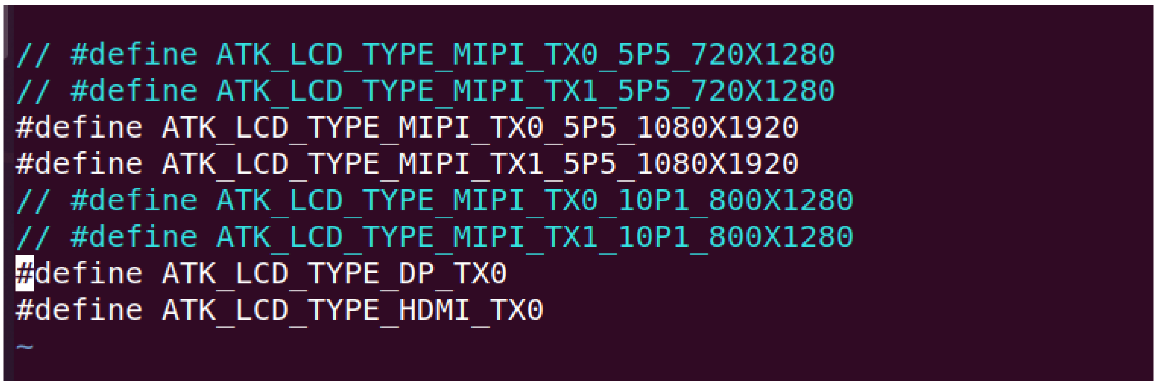
4.1.3 Four-Screen Shared Display

The four-screen shared display solution for the card computer is as follows:

➢ Dual MIPI + HDMI + DP

Open the Linux kernel device tree file kernel/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi, enable the HDMI TX0, DP TX0, MIPI DSI0 and MIPI DSI1 interfaces (taking 2 5.5-inch 1080x1920 MIPI screens as an example), as follows:

```
vi kernel/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi
```



```
// #define ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_720X1280
#define ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920
#define ATK_LCD_TYPE_MIPI_TX1_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280
// #define ATK_LCD_TYPE_MIPI_TX1_10P1_800X1280
#define ATK_LCD_TYPE_DP_TX0
#define ATK_LCD_TYPE_HDMI_TX0
~
```

Figure 4.1-5 Display interface configuration

After modification, save and exit!

Open the "buildroot/package/weston/weston.sh" file and configure the main screen (or the main display)

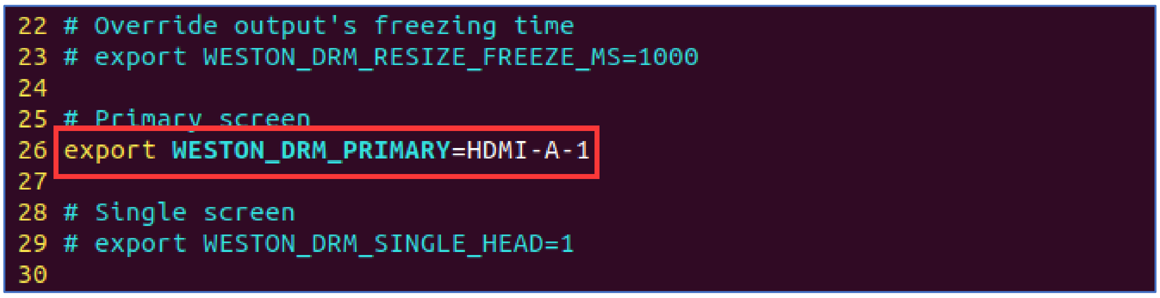
```
vi buildroot/package/weston/weston.sh
```

Find the following line:

```
# export WESTON_DRM_PRIMARY=eDP-1
```

Remove the "#" symbol at the beginning and open it. This variable is used to specify the primary screen. In this example, we set the primary screen to the HDMI display connected to the HDMI TX0 interface:

```
export WESTON_DRM_PRIMARY=HDMI-A-1
```



```
# Override output's freezing time
# export WESTON_DRM_RESIZE_FREEZE_MS=1000

# Primary screen
export WESTON_DRM_PRIMARY=HDMI-A-1

# Single screen
# export WESTON_DRM_SINGLE_HEAD=1

```

Figure 4.1-6 Configure the main screen

HDMI-A-1 is the connector corresponding to the HDMI display. After modification, save and exit!

Tips: After compilation, the weston.sh script will be copied to the etc/profile.d/ directory of the target root file system. Therefore, you can modify the /etc/profile.d/weston.sh file during system operation to configure the Weston environment variables. After modification, save and exit, and restart the system for the new configuration to take effect.

Execute the following command in the SDK root directory to recompile the Linux kernel and the buildroot root file system (the entire SDK has been fully compiled before):

```
./build.sh alientek_quarkpi_ca2_defconfig
./build.sh kernel
./build.sh buildroot-make:weston-rebuild
./build.sh buildroot
```

Connect one HDMI display and one DP display to the HDMI TX0 port and the full-function typec interface of the development board, and connect two MIPI screens to the MIPI DSI1 and MIPI DSI2 interfaces of the development board (in this case, two 5.5-inch 1080x1920 MIPI screens are used).

Burn the compiled boot.img and rootfs.img onto the development board for testing.

4.2 Multi-screen Splicing Display

Weston supports the multi-screen splicing display function. It can splice multiple screens in a horizontal direction in a 1xN (N represents the number of screens) mode. The configuration method is also very simple.

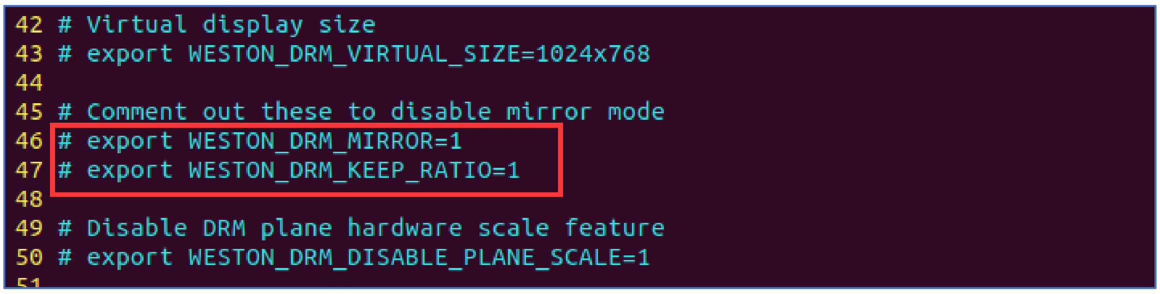
Open the buildroot/package/weston/weston.sh file:

```
vi buildroot/package/weston/weston.sh
```

Find the following two lines:

```
export WESTON_DRM_MIRROR=1
export WESTON_DRM_KEEP_RATIO=1
```

Commenting out these two lines will enable the concatenation mode, as shown below:



```
# Virtual display size
# export WESTON_DRM_VIRTUAL_SIZE=1024x768

# Comment out these to disable mirror mode
# export WESTON_DRM_MIRROR=1
# export WESTON_DRM_KEEP_RATIO=1

# Disable DRM plane hardware scale feature
# export WESTON_DRM_DISABLE_PLANE_SCALE=1
```

Figure 4.2-1 Enable the splicing mode

export WESTON_DRM_MIRROR=1: Use the mirror mode. If this environment variable is not set, it will be the splicing mode.

export WESTON_DRM_KEEP_RATIO=1: In the mirror mode, scaling maintains the aspect ratio. If this environment variable is not set, it will be the forced full-screen mode.

After modification, save and exit!

Tips: After compilation, the weston.sh script will be copied to the etc/profile.d/ directory of the target root file system. Therefore, you can modify the /etc/profile.d/weston.sh file during system operation to configure the weston environment variables. After modification, save and exit, and the new configuration will take effect after a restart.

Execute the following command in the SDK root directory to recompile the buildroot root file system (the entire SDK has been fully compiled before):

```
./build.sh alientek_quarkpi_ca2_defconfig
./build.sh buildroot-make:weston-rebuild
```

```
./build.sh buildroot
```

Burn the compiled rootfs.img onto the development board for testing.

4.3 Screen display orientation rotation

Weston supports screen display orientation rotation and the configuration method is very simple!

Open the "buildroot/board/rockchip/common/overlays/10-weston/etc/xdg/weston/weston.ini" file:

```
vi buildroot/board/rockchip/common/overlays/10-weston/etc/xdg/weston/weston.ini
```

For example, in the "weston.ini" file, add the following content to rotate the HDMI display by 90 degrees:

```
[output]
name=HDMI-A-1
transform=rotate-90
```

Add the following content to the "weston.ini" file to rotate the MIPI screen by 180 degrees:

```
[output]
name=DSI-1
transform=rotate-90
The possible values of "transform" are "rotate-90", "rotate-180", and "rotate-270".
```

Save and exit after modification!

Tips: After compilation, the weston.ini file will be copied to the etc/xdg/weston/ directory of the target root file system. Therefore, you can modify the /etc/xdg/weston/weston.ini file during system operation to configure the screen display direction. After modification, save and exit, and the new configuration will take effect after a restart.

If you need to dynamically configure the screen display direction during operation, you can use dynamic configuration files, such as:

```
echo "output:DSI-1:rotate90" > /tmp/.weston_drm.conf
#Rotate the MIPI screen by 90 degrees
echo "output:DSI-1:rotate180" > /tmp/.weston_drm.conf
#Rotate the MIPI screen by 180 degrees
echo "output:DSI-1:rotate270" > /tmp/.weston_drm.conf
#Rotate the MIPI screen by 270 degrees
echo "output:all:rotate90" > /tmp/.weston_drm.conf
# Rotate all screens by 90 degrees.
```

Execute the following command in the SDK root directory to recompile the buildroot root file system (the entire SDK has been fully compiled before):

```
./build.sh alientek_quarkpi_ca2_defconfig
./build.sh buildroot
```

Burn the compiled rootfs.img onto the development board for testing.

4.4 Multi-screen Touch Configuration

Refer to Section 2.12 of the document docs/cn/Linux/Graphics/Rockchip_Developer_Guide_Buildroot_Weston_CN.pdf.

4.5 Other Configurations of Weston

Read the official documentation provided by RK:
docs/cn/Linux/Graphics/Rockchip_Developer_Guide_Buildroot_Weston_CN.pdf

Chapter 5. 8K Display Configuration

As mentioned earlier, the HDMI display interface and DP display interface of the RK3588S platform support 8K output. The HDMI interface can support up to 8K at 60fps, while the DP interface can support up to 8K at 30fps.

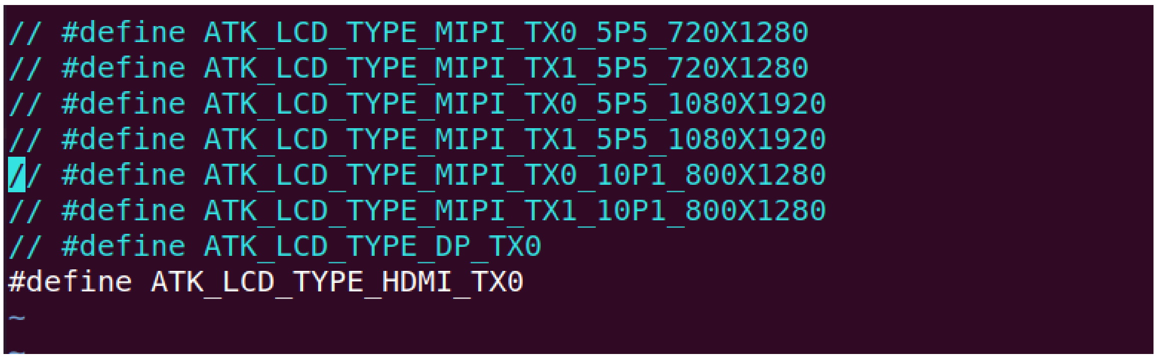
This chapter will explain how to configure the device tree for Android 12 to support 8K output. 8K displays are not very common. If you don't have an 8K display, you won't be able to test it, so there's no need to go through all the trouble!!!

5.1 HDMI 8K Output Configuration

As can be seen from Figure 1.2.3, only VP0 supports 8K output. In the 8K output mode, one display interface needs to occupy both VP0 and VP1 simultaneously. Therefore, if the product needs to support 8K display output, be sure not to connect any other display interfaces on VP1.

Open the Linux kernel device tree file kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi, enable the HDMI TX0 display interface, as shown below:

```
vi kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi
```



```
// #define ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_720X1280
// #define ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280
// #define ATK_LCD_TYPE_MIPI_TX1_10P1_800X1280
// #define ATK_LCD_TYPE_DP_TX0
#define ATK_LCD_TYPE_HDMI_TX0
~
```

Figure 5.1-1 Display interface configuration

After modification, save and exit!

Open the Linux kernel device tree file: kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-quarkpi-ca2.dts

```
vi kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-quarkpi-ca2.dts
```

Add the following content at the end of the document:

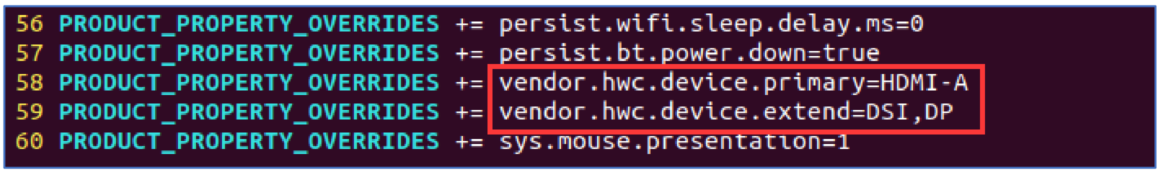
```
&vop {
	assigned-clocks = <&cru ACLK_VOP>;
	assigned-clock-rates = <800000000>;// Set the ACLK frequency to 800M. The default is 500M.
};
```

The HDMI TX0 display interface is already connected to the VP0 by default and does not require any modification. After the modification, save and exit!

Open the file device/rockchip/rk3588/atk_quarkpi_ca2/atk_quarkpi_ca2.mk, and configure the main screen as an HDMI screen as follows:

```
vi device/rockchip/rk3588/atk_quarkpi_ca2/atk_quarkpi_ca2.mk
```

```
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=HDMI-A
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=DSI,DP
```



```
PRODUCT_PROPERTY_OVERRIDES += persist.wifi.sleep.delay.ms=0
PRODUCT_PROPERTY_OVERRIDES += persist.bt.power.down=true
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=HDMI-A
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=DSI,DP
PRODUCT_PROPERTY_OVERRIDES += sys.mouse.presentation=1
```

Figure 5.1-2 Configure the main and secondary screens

After the modification is completed, save and exit!

Execute the following command in the root directory of the SDK to recompile the Linux kernel and Android source code:

```
source build/envsetup.sh
lunch atk_quarkpi_ca2-userdebug
./build.sh -KA -J10
```

Connect the HDMI display to the HDMI TX0 interface of the development board.

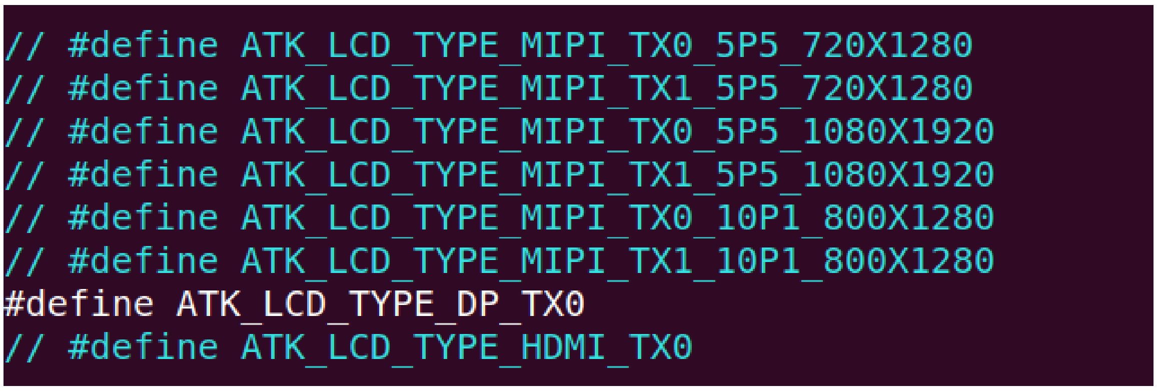
Burn the compiled super.img and boot.img onto the development board for testing.

5.2 DP 8K Output Configuration

As can be seen from Figure 1.2.3, only VP0 supports 8K output. In the 8K output mode, one display interface needs to occupy both VP0 and VP1 simultaneously. Therefore, if the product needs to support 8K display output, be careful not to connect other display interfaces on VP1.

Open the Linux kernel device tree file kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi, enable the DP TX0 display interface as follows:

```
vi kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-screen_choose.dtsi
```



```
// #define ATK_LCD_TYPE_MIPI_TX0_5P5_720X1280
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_720X1280
// #define ATK_LCD_TYPE_MIPI_TX0_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX1_5P5_1080X1920
// #define ATK_LCD_TYPE_MIPI_TX0_10P1_800X1280
// #define ATK_LCD_TYPE_MIPI_TX1_10P1_800X1280
#define ATK_LCD_TYPE_DP_TX0
// #define ATK_LCD_TYPE_HDMI_TX0
```

Figure 5.2-1 Display interface configuration

After modification, save and exit!

Open the Linux kernel device tree file: kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-quarkpi-ca2.dts

```
vi kernel-5.10/arch/arm64/boot/dts/rockchip/rk3588s-atk-quarkpi-ca2.dts
```

Add the following content at the end of the document:

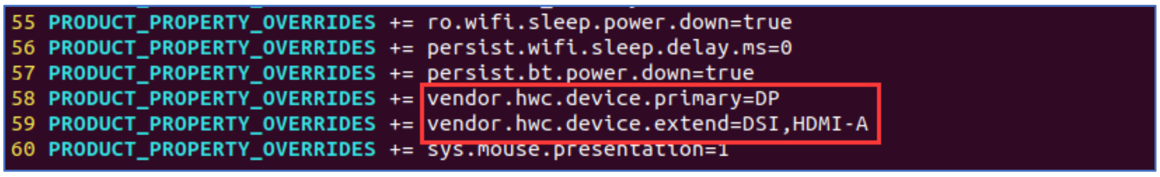
```
&vop {
        assigned-clocks = <&cru ACLK_VOP>;
        assigned-clock-rates = <800000000>;// Set the ACLK frequency to 800M. The default is 500M.
};

// Connect the DP TX0 display interface to VP0. By default, DP TX0 is connected to VP1.
&dp0_in_vp0 {
        status = "okay";
};

&dp0_in_vp1 {
        status = "disabled";
};

&dp0_in_vp2 {
        status = "disabled";
};
```

After modification, save and exit!

Open the file device/rockchip/rk3588/atk_quarkpi_ca2/atk_quarkpi_ca2.mk, and configure the main screen as a DP screen as follows:

```
vi device/rockchip/rk3588/atk_quarkpi_ca2/atk_quarkpi_ca2.mk
```

```
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=DP
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=DSI,HDMI-A
```



```
PRODUCT_PROPERTY_OVERRIDES += ro.wifi.sleep.power.down=true
PRODUCT_PROPERTY_OVERRIDES += persist.wifi.sleep.delay.ms=0
PRODUCT_PROPERTY_OVERRIDES += persist.bt.power.down=true
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.primary=DP
PRODUCT_PROPERTY_OVERRIDES += vendor.hwc.device.extend=DSI,HDMI-A
PRODUCT_PROPERTY_OVERRIDES += sys.mouse.presentation=1
```

Figure 5.2-2 Configure the main and secondary screens

After the modification is completed, save and exit!

Execute the following command in the root directory of the SDK to recompile the Linux kernel and Android source code:

```
source build/envsetup.sh
lunch atk_quarkpi_ca2-userdebug
./build.sh -KA -J10
```

Burn the compiled super.img and boot.img onto the development board. After the burning process is completed, power off the development board and connect the DP display to the full-function TYPE-C interface of the development board.

Then power on the development board to start the test.

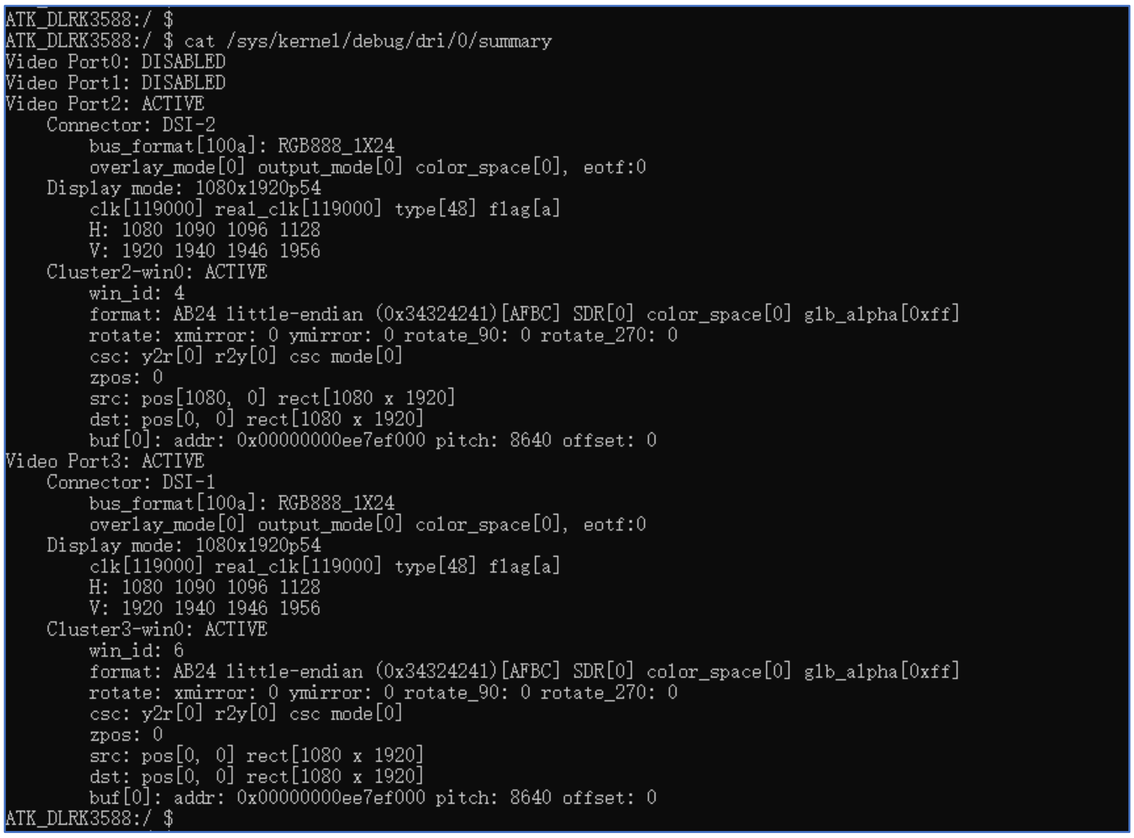
Chapter 6. Common Debugging Techniques

Common debugging methods.

6.1 Dump the current display status

Execute the following command to print the current display status:

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



```
ATK_DLRK3588:/ $
ATK_DLRK3588:/ $ cat /sys/kernel/debug/dri/0/summary
Video Port0: DISABLED
Video Port1: DISABLED
Video Port2: ACTIVE
    Connector: DSI-2
        bus_format[100a]: RGB888_1X24
        overlay_mode[0] output_mode[0] color_space[0], eotf:0
    Display mode: 1080x1920p54
        clk[119000] real_clk[119000] type[48] flag[a]
        H: 1080 1090 1096 1128
        V: 1920 1940 1946 1956
    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: 0
        src: pos[1080, 0] rect[1080 x 1920]
        dst: pos[0, 0] rect[1080 x 1920]
        buf[0]: addr: 0x00000000ee7ef000 pitch: 8640 offset: 0
Video Port3: ACTIVE
    Connector: DSI-1
        bus_format[100a]: RGB888_1X24
        overlay_mode[0] output_mode[0] color_space[0], eotf:0
    Display mode: 1080x1920p54
        clk[119000] real_clk[119000] type[48] flag[a]
        H: 1080 1090 1096 1128
        V: 1920 1940 1946 1956
    Cluster3-win0: ACTIVE
        win_id: 6
        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: 0
        src: pos[0, 0] rect[1080 x 1920]
        dst: pos[0, 0] rect[1080 x 1920]
        buf[0]: addr: 0x00000000ee7ef000 pitch: 8640 offset: 0
ATK_DLRK3588:/ $
```

Figure 6.1-1 Display status

The status of the VP is ACTIVE, indicating that it is running; if it is DISABLED, it means that the VP is not running.

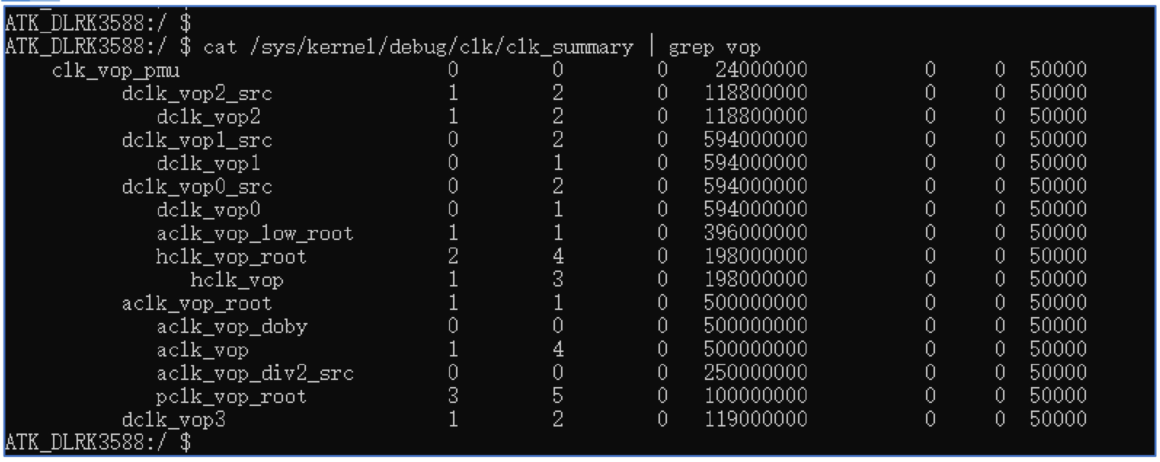
6.2 View the current display clock

Executing the following command will print the entire clock tree:

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

If you only focus on the VOP clock, then execute the following command:

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



```
ATK_DLRK3588:/ $
ATK_DLRK3588:/ $ cat /sys/kernel/debug/clk/clk_summary | grep vop
    clk_vop_pmu                      0        0        0     24000000          0      0  50000
          dclk_vop2_src              1        2        0    118800000          0      0  50000
             dclk_vop2               1        2        0    118800000          0      0  50000
          dclk_vop1_src              0        2        0    594000000          0      0  50000
             dclk_vop1               0        1        0    594000000          0      0  50000
          dclk_vop0_src              0        2        0    594000000          0      0  50000
             dclk_vop0               0        1        0    594000000          0      0  50000
             aclk_vop_low_root       1        1        0    396000000          0      0  50000
             hclk_vop_root           2        4        0    198000000          0      0  50000
                hclk_vop             1        3        0    198000000          0      0  50000
          aclk_vop_root              1        1        0    500000000          0      0  50000
             aclk_vop_doby           0        0        0    500000000          0      0  50000
             aclk_vop                1        4        0    500000000          0      0  50000
             aclk_vop_div2_src       0        0        0    250000000          0      0  50000
             pclk_vop_root           3        5        0    100000000          0      0  50000
          dclk_vop3                  1        2        0    119000000          0      0  50000
ATK_DLRK3588:/ $
```

Figure 6.2-1 VOP-related clocks

dclk_vop0, dclk_vop1, dclk_vop2, and dclk_vop3 correspond to the dclk (display clock, i.e., pixel clock) of the 4 VPs respectively.