密级状态：绝密( ) 秘密( ) 内部( √ ) 公开( )

**基于DRM的Android显示使用指南**

（第一系统产品部）

|  |  |  |
| --- | --- | --- |
| 文件状态：  [√] 正在修改  [ ] 正式发布 | 当前版本： | V1.0 |
| 作 者： | 郑阳、许惠聪、许碧绿、操瑞杰 |
| 完成日期： | 2017-XX-XX |
| 审 核： | XXX |
| 完成日期： | 201X-XX-XX |

福州瑞芯微电子有限公司

Fuzhou Rockchips Semiconductor Co . , Ltd

(版本所有,翻版必究)

版 本 历 史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本号 | 作者 | 修改日期 | 修改说明 | 备注 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目 录

[1. 简介 3](#_Toc499196945)

[2. 屏幕配置 3](#_Toc499196946)

[2.1 配置dts 3](#_Toc499196947)

[2.1.1 使能对应显示设备节点 3](#_Toc499196948)

[2.1.2 使能显示接口组件 3](#_Toc499196949)

[2.1.3 绑定PLL（只有rk3399需要） 4](#_Toc499196950)

[2.2 LCD Panel timing 配置 5](#_Toc499196951)

[2.2.1 dts中配置timing 5](#_Toc499196952)

[2.2.2 panel-simple中配置timing 7](#_Toc499196953)

[2.3 MIPI 10](#_Toc499196954)

[2.3.1 mipi基本配置 10](#_Toc499196955)

[2.3.2 绑定VOP 13](#_Toc499196956)

[2.3.3 初始化命令详解 14](#_Toc499196957)

[2.4 HDMI相关配置说明 17](#_Toc499196958)

[2.4.1信号强度配置(3288/3368/3399)： 17](#_Toc499196959)

[2.4.2 ddc的i2c速率配置: 18](#_Toc499196960)

[2.4.3 使能hdcp1.4 19](#_Toc499196961)

[2.4.4 打开音频 19](#_Toc499196962)

[2.5 DP配置说明（RK3399） 19](#_Toc499196963)

[2.5.1 DP检测： 19](#_Toc499196964)

[2.5.2 绑定typec口 20](#_Toc499196965)

[2.5.3 注册dp 驱动: 20](#_Toc499196966)

[2.5.4不使用fusb302将typec口固定作dp输出： 21](#_Toc499196967)

[2.6 参考配置 21](#_Toc499196968)

[2.6.1 单MIPI屏 21](#_Toc499196969)

[2.6.2 双MIPI单屏 26](#_Toc499196970)

[2.6.3 单EDP屏 29](#_Toc499196971)

[2.6.4 两个单MIPI屏 34](#_Toc499196972)

[2.6.5 单LVDS屏 34](#_Toc499196973)

[2.6.6 单HDMI(vopb) 39](#_Toc499196974)

[2.6.7 单VGA（基于rk3288 PopMatal板） 41](#_Toc499196975)

[2.6.8 EDP(vopb) + HDMI(vopl) 44](#_Toc499196976)

[2.6.9 LVDS(vopl) + HDMI(vopb) 45](#_Toc499196977)

[2.6.10 MIPI(vopb) + HDMI(vopl) 46](#_Toc499196978)

[2.6.11 HDMI(vopb) + DP(vopl) 46](#_Toc499196979)

[2.6.12 HDMI + CVBS（单VOP） 47](#_Toc499196980)

[2.6.13 HDMI（vopb）+ CVBS（vopl） 50](#_Toc499196981)

[2.6.14 MIPI + EDP（3288需要先配置build.prop） 50](#_Toc499196982)

[2.6.15 MIPI + LVDS/RGB（3399不支持LVDS/RGB，需要外接转换IC） 60](#_Toc499196983)

[2.6.16 EDP + LVDS/RGB（3399不支持LVDS/RGB，需要外接转换IC） 65](#_Toc499196984)

[3. 显示框架配置 71](#_Toc499196985)

[3.1 主副显示器配置 71](#_Toc499196986)

[3.2 主副显示器接口查询 72](#_Toc499196987)

[3.3 FrameBuffer分辨率配置 72](#_Toc499196988)

[3.4 分辨率过滤配置 72](#_Toc499196989)

[4. 常用调试方法 75](#_Toc499196990)

[4.1 查看VOP状态 75](#_Toc499196991)

[4.2 查看Connector状态 76](#_Toc499196992)

[4.3 命令行设置分辨率 77](#_Toc499196993)

[5. 参考文档 77](#_Toc499196994)

# 简介

DRM全称是Direct Rendering Manager是DRI(Direct Rendering Infrastructure)框架的一个组件。 Android新版本逐渐从Framebuffer框架迁移到DRM上。从内核4.4开始，RK的显示框架逐渐迁移到DRM上。本文档介绍如何使用新的显示框架，适用于以下SDK：

RK3399 BOX Android7.1

RK3368 BOX Android7.1

RK3288 MID Android7.1

# 屏幕配置

## 2.1 配置dts

### 2.1.1 使能对应显示设备节点

打开显示设备执行相关hdmi的probe函数注册显示设备驱动，如打开HDMI需要添加：

&hdmi {

status = "okay";

};

### 2.1.2 使能显示接口组件

display-subsystem注册会把所有打开的设备以组件的形式加在一起, 等所有的组件加载完毕后, 统一进行bind/unbind。

#### 2.1.2.1 绑定VOP

如果平台存在两个VOP（RK3288、RK3399）：vopb（支持4K）、vopl（只支持2K），当显示设备节点打开时，显示接口对应vopb和vopl的ports都会打开，需要关闭用不到的那个VOP。

比如hdmi绑定到vopb需要添加：

&hdmi\_in\_vopl {

status = "disabled";

};

反之若绑定到vopl则添加：

&hdmi\_in\_vopb {

status = "disabled";

};

如果平台只有一个VOP，可以跳过。

#### 2.1.2.2 开机 logo

如果uboot logo未开启，那kernel阶段也无法显示开机logo，只能等到android启动后才能看到显示。在dts里面将对应的route使能即可打开uboot logo支持，比如打开hdmi的uboot logo显示:

&route\_hdmi {

status = "okay"

};

### 2.1.3 绑定PLL（只有rk3399需要）

rk3399的hdmi所绑定的vop时钟需要挂载到vpll上，若是双显需将另一个vop时钟挂到cpll这样可以分出任意dclk的频率。如当hdmi绑定到vopb时配置：

&vopb {

assigned-clocks = <&cru DCLK\_VOP0\_DIV>;

assigned-clock-parents = <&cru PLL\_VPLL>;

};

&vopl {

assigned-clocks = <&cru DCLK\_VOP1\_DIV>;

assigned-clock-parents = <&cru PLL\_CPLL>;

}；

当hdmi绑定到vopl时配置：

&vopb {

assigned-clocks = <&cru DCLK\_VOP0\_DIV>;

assigned-clock-parents = <&cru PLL\_CPLL>;

};

&vopl {

assigned-clocks = <&cru DCLK\_VOP1\_DIV>;

assigned-clock-parents = <&cru PLL\_VPLL>;

};

## 2.2 LCD Panel timing 配置

目前我们平台LCD panel（MIPI/EDP/LVDS）的timing配置有下面两种：

1. dts文件中配置disp\_timings结构。
2. 在drivers/gpu/drm/panel/panel-simple.c 中添加timing结构。

下面分别对这两种配置进行详细说明，以RK3399 SDK上的EDP panel（lg\_lp079qx1\_sp0v）为例：

### 2.2.1 dts中配置timing

这种方式直接在dts中配置disp\_timings结构，简单明了，推荐客户使用该方式，dts中的配置如下：

edp\_panel: edp-panel {

compatible = "panel-simple";

backlight = <&backlight>;

power-supply = <&vcc3v3\_s0>;

enable-gpios = <&gpio1 13 GPIO\_ACTIVE\_HIGH>;

ports {

panel\_in\_edp: endpoint {

remote-endpoint = <&edp\_out\_panel>;

};

};

disp\_timings: display-timings {

native-mode = <&timing0>;

timing0: timing0 {

clock-frequency = <200000000>;

hactive = <1536>;

vactive = <2048>;

hfront-porch = <12>;

hsync-len = <16>;

hback-porch = <48>;

vfront-porch = <8>;

vsync-len = <4>;

vback-porch = <8>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

|  |  |  |
| --- | --- | --- |
| property | Value | comment |
| clock-frequency | 200000000 | Dclk频率，单位Hz |
| hactive | 1536 | 行有效像素 |
| vactive | 2048 | 列有效像素 |
| hfront-porch | 12 | 行前消隐 |
| hsync-len | 16 | 行同步信号 |
| hback-porch | 48 | 行后消隐 |
| vfront-porch | 8 | 列前消隐 |
| vsync-len | 4 | 列同步信号 |
| vback-porch | 8 | 列后消隐 |
| hsync-active | 0 | Hsync信号极性配置 |
| vsync-active | 0 | Vsync 信号极性配置 |
| de-active | 0 | Den 信号极性配置 |
| pixelclk-active | 0 | Dclk信号极性配置 |

### 2.2.2 panel-simple中配置timing

把Timing写在panel-simple.c中，直接以短字符串匹配，该方式为upstream推荐的使用方式。

这种方式dts中的配置如下：

edp\_panel: edp-panel {

compatible = "lg,lp079qx1-sp0v";

backlight = <&backlight>;

power-supply = <&vcc3v3\_s0>;

enable-gpios = <&gpio1 13 GPIO\_ACTIVE\_HIGH>;

ports {

panel\_in\_edp: endpoint {

remote-endpoint = <&edp\_out\_panel>;

};

};

};

drivers/gpu/drm/panel/panel-simple.c中添加对应的结构。分下面两块：

① 对EDP/LVDS屏来说，首先需要在platform\_of\_match[]结构中添加of\_device\_id结构：

{

.compatible = "lg,lp079qx1-sp0v",

.data = &lg\_lp079qx1\_sp0v,

},

MIPI屏添加的位置为dsi\_of\_match[]结构中，以SDK的MIPI panel为例：

{

.compatible = "boe,tv080wum-nl0",

.data = &boe\_tv080wum\_nl0

},

Note: compatible 中的名字需要跟dts 中的compatible 名字一致。

② 添加下面的上面data所指定的&lg\_lp079qx1\_sp0v结构：

static const struct panel\_desc lg\_lp079qx1\_sp0v = {

.modes = &lg\_lp079qx1\_sp0v\_mode,

.num\_modes = 1,

.size = {

.width = 129,

.height = 171,

},

.bus\_format = MEDIA\_BUS\_FMT\_RGB666\_1X18,

};

|  |  |  |
| --- | --- | --- |
| property | Value | comment |
| num\_modes | 1 | 设置为1即可 |
| size | 129、171 | 屏的物理尺寸 |
| bus\_format | MEDIA\_BUS\_FMT\_RGB666\_1X18 | 主控输出给屏的数据格式 |

③ 添加第②步中modes指定的&lg\_lp079qx1\_sp0v\_mode结构：

static const struct drm\_display\_mode lg\_lp079qx1\_sp0v\_mode = {

.clock = 200000,

.hdisplay = 1536,

.hsync\_start = 1536 + 12,

.hsync\_end = 1536 + 12 + 16,

.htotal = 1536 + 12 + 16 + 48,

.vdisplay = 2048,

.vsync\_start = 2048 + 8,

.vsync\_end = 2048 + 8 + 4,

.vtotal = 2048 + 8 + 4 + 8,

.vrefresh = 60,

.flags = DRM\_MODE\_FLAG\_NVSYNC | DRM\_MODE\_FLAG\_NHSYNC,

};

|  |  |  |
| --- | --- | --- |
| property | Value | comment |
| clock | 200000 | Dclk，单位KHz |
| hdisplay | 1536 | 行有效分辨率 |
| hsync\_start | 1536 + 12 | Hdisplay + HFP |
| hsync\_end | 1536 + 12 + 16 | Hdisplay + HFP + HSYNC |
| htotal | 1536 + 12 + 16 + 48 | 行总像素 Hdisplay + HFP + HSYNC + HBP |
| vdisplay | 2048 | 列有效分辨率 |
| vsync\_start | 2048 + 8 | Vdisplay + VFP |
| vsync\_end | 2048 + 8 + 4 | Vdisplay + VFP + VSYNC |
| vtotal | 2048 + 8 + 4 + 8 | 列总像素 Vdisplay + VFP + VSYNC + VBP |
| vrefresh | 60 | 刷新率 |
| flags | DRM\_MODE\_FLAG\_NVSYNC | DRM\_MODE\_FLAG\_NHSYNC | Video mode flags |

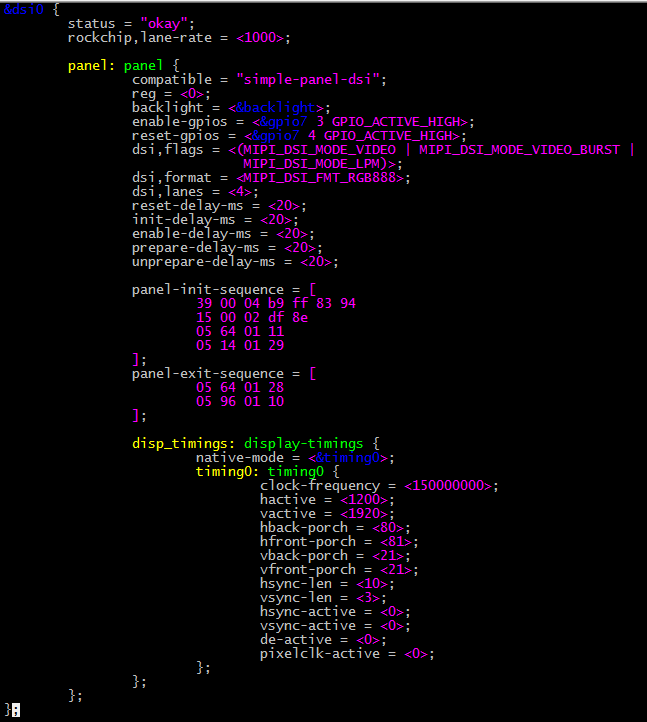
## 2.3 MIPI

### 2.3.1 mipi基本配置

#### 2.3.1.1 MIPI-DSI Host 以及 Panel的配置

关于屏的timing信息我们有两种配置方式：

1. dts文件中配置disp\_timings结构。见下面截图中的结构。
2. 在drivers/gpu/drm/panel/panel-simple.c 中添加timing结构。见2.2章节。



#### 2.3.1.2 属性说明

|  |  |  |
| --- | --- | --- |
| property | value | comment |
| rockchip,lane-rate | 1000 | Mipi clk 速率，单位Mbps/lane。如果没有配置该属性，驱动会根据屏的timing计算lane-rate |
| compatible | simple-panel-dsi | 与mipi panel驱动进行匹配 |
| reg | 0 | Mipi DSI virtual channel |
| backlight | &backlight | 引用背光节点，panel驱动会解析并对背光进行控制 |
| enable-gpios | &gpio7 3 GPIO\_ACTIVE\_HIGH | 屏的enable脚的gpio配置，参考原理图。 |
| reset-gpios | &gpio7 4 GPIO\_ACTIVE\_HIGH | 屏的reset脚的gpio配置，参考原理图。 |
| dsi,flags | MIPI\_DSI\_MODE\_VIDEO | MIPI\_DSI\_MODE\_VIDEO\_BURST | MIPI\_DSI\_MODE\_LPM | Dsi host的一些flag选项。MIPI\_DSI\_MODE\_VIDEO\_BURST：数据发送模式， 绝大部分mipi屏都只支持或者兼容支持该模式。  MIPI\_DSI\_MODE\_LPM：在low power模式下发送命令 |
| dsi,format | MIPI\_DSI\_FMT\_RGB888 | Pixel format 具体看屏的要求，mipi一般是888 |
| dsi,lanes | 4 | Lane的条数，dual-channel屏为8。参考屏规格书 |
| reset-delay-ms | 20 | Reset信号有效脉冲宽度，可选，参考屏规格书 |
| init-delay-ms | 20 | 发送初始化命令之前的延时，可选，参考屏规格书 |
| enable-delay-ms | 20 | 开启背光之前的延时，可选，参考屏规格书 |
| prepare-delay-ms | 20 | Enable信号控制后的延时，可选，参考屏规格书 |
| unprepare-delay-ms | 20 | 屏幕掉电gpio操作后的延时，可选，参考屏规格书 |
| panel-init-sequence | 05 64 01 11等 | 屏上电的初始化序列，一般由屏的fae提供。 |
| panel-exit-sequence | 05 96 01 10等 | 屏下电序列，一般由屏的fae提供 |
| disp\_timings |  | 屏的timing信息，参考屏规格书 |

#### 2.3.1.3 MIPI-DPHY



**Note：**只适用于使用Non-SNPS PHY作为DPHY的Soc系列，比如RK3366/RK3368。对于使用SNPS PHY作为DPHY的Soc系统，比如RK3288/RK3399，不需要配置该node。

#### 2.3.1.4 uboot logo

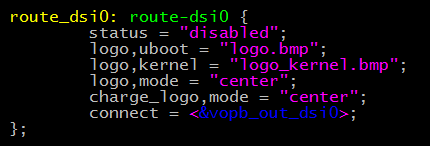


Route\_dsi0 是uboot显示的开关。okay表示开启uboot显示。

### 2.3.2 绑定VOP

关于mipi dsi连接那个vop的配置，配置分uboot和kernel两部分。针对rk3288平台，这部分配置放在rk3288-android.dtsi中。

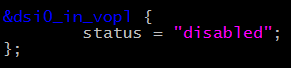
#### 2.3.2.1 uboot 配置



上面connect = <&vopb\_out\_dsi0>表示dsi0连接vopb，如果需要连接vopl，可以设置为<&vopl\_out\_dsi0>。

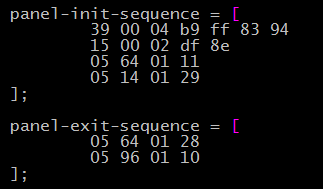
#### 2.3.2.2 kernel中的配置

在没有做任何配置的前提下，我们kernel中默认接vopl， 如需要接跟uboot保持一致接vopb，那么需要把dsi0\_in\_vopl给关闭，如下所示：



### 2.3.3 初始化命令详解

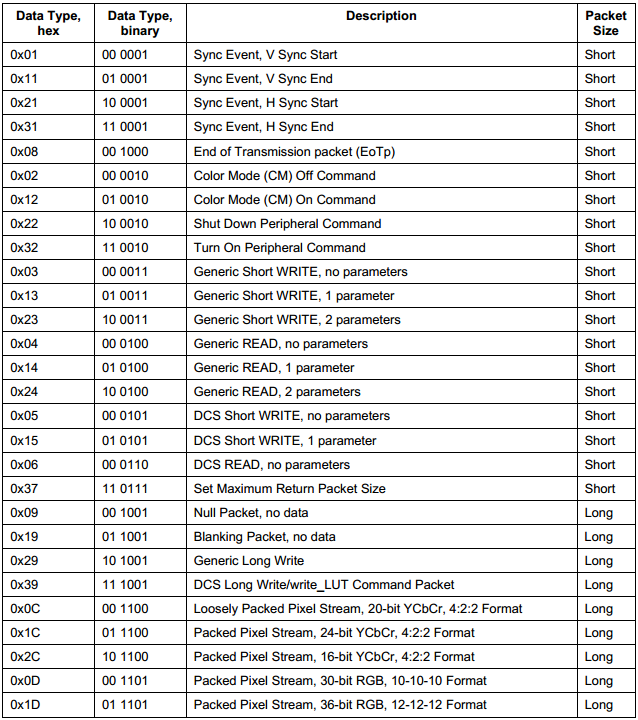
#### 2.3.3.1 命令格式说明

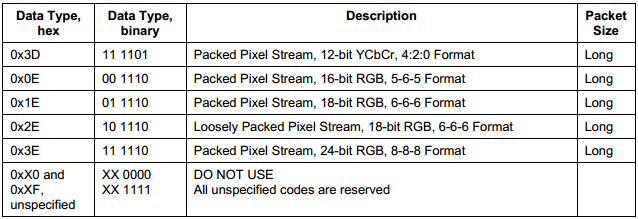


命令的前面三个字节分别表示命令类型、延时和命令净荷长度。从第四个字节开始表示命令的有效payload。这个字节数需要与第三个字节一致。

第一个字节代表的命令类型分两大类：DCS命令和Generic命令。DCS命令是有mipi标准协议里面指定的命令，具体可以参考《MIPI Alliance Specification for Display Command Set.pdf 》。Generic命令一般应用于厂商自定义的命令。具体使用哪种类型需要参考屏规格书或者咨询屏厂FAE。如果没有特别指定，建议按DCS命令类型配置。

DCS命令的类型有三种：0x05/0x15/0x39。Generic命令的类型分为：0x03/0x13/0x23/0x29。详细的含义见下面表格：





第二个字节表示的是发送完该命令以后需要延时的时间，单位为ms。这个时间需要参考屏规格书或者由屏厂FAE提供。

第三个字节表示的是命令payload的长度。

以panel-init-sequence中的第一条命令[39 00 04 b9 ff 83 94]为例，详细解释一下各个自己表示的含义：

0x39 表示这是一个DCS长包命令，payload长度大于2。

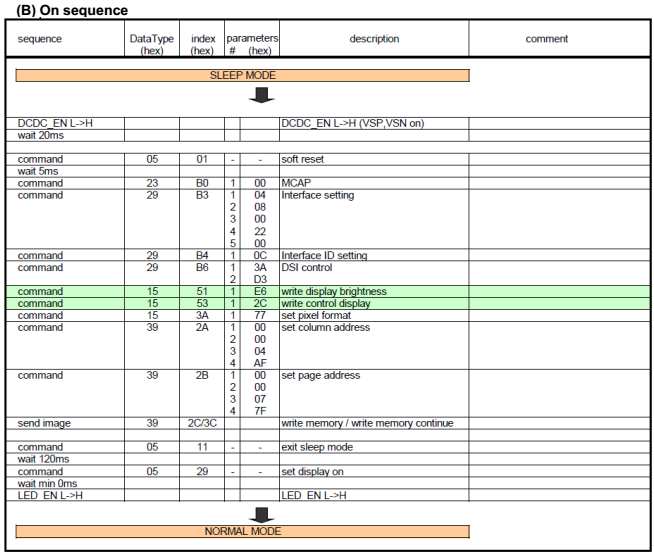
0x00 表示该命令发送后无需延时。

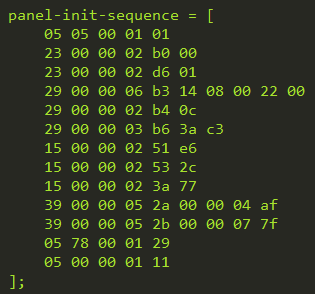
0x04 表示命令长度为4。

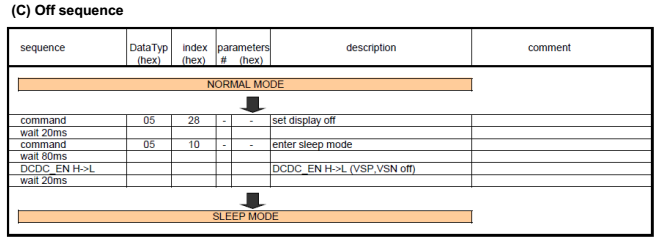
0xb9 0xff 0x83 0x94 表示的是实际发送的命令。

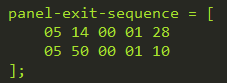
#### 2.3.3.2 示例

以Japan Display 的 H070ME05000 进行说明， 该屏的规格书有详细的初始化序列说明。









## 2.4 HDMI相关配置说明

### 2.4.1信号强度配置(3288/3368/3399)：

由于硬件走线差异,不同板子有可能需要不同的驱动强度配置，当遇到电视兼容性问题时可以尝试修改这个看是否有改善。

hdmi信号强度可通过dts的rockchip.phy-table属性配置，格式定义:<PIXELCLOCK PHY\_CKSYMTXCTRL PHY\_TXTERM PHY\_VLEVCTRL>.

PIXELCLOCK表示所在行参数所对应的最大pixelclock频率。

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

Bit[0]：Clock信号使能

Bit[3:1]：Data信号预加重，定义如下

Bit[4:5]：Data信号sloop boost

PHY\_TXTERM寄存器(0x19)值用于调整端接电阻值

Bit[0:2]：值越大，端接电阻值越大。

PHY\_VLEVCTRL寄存器(0x0e)值用于调整HDMI的信号幅度，具体定义如下：

Bit[0:4] : tmds\_clk +/- 信号幅度，值越低，驱动能力越强；

Bit[5:9]: tmds\_data +/- 信号幅度，值越低，驱动能力越强。

如：

&hdmi {

rockchip,phy-table =

<74250000 0x8009 0x0004 0x0272>,

<165000000 0x802b 0x0004 0x0209>,

<297000000 0x8039 0x0005 0x028d>,

<594000000 0x8039 0x0000 0x019d>,

<000000000 0x0000 0x0000 0x0000>;

};

其中<74250000 0x8009 0x0004 0x0272>，表示pixeclock为74.25M(720p分辨率)以下时PHY\_CKSYMTXCTRL寄存器值为0x8009；PHY\_TXTERM值为0x0004; PHY\_VLEVCTRL值为0x0272。修改后也可用cat /d/dw-hdmi/phy命令查看对应的寄存器值确认是否有修改成功。

### 2.4.2 ddc的i2c速率配置:

目前i2c速率通过clk高电平和低电平的时间来定义，如下为实测i2c速率为50k时候的配置，调整i2c速率只需将这2个值按对应的比例修改即可。

&hdmi {

ddc-i2c-scl-high-time-ns = <9625>;

ddc-i2c-scl-low-time-ns = <10000>;

}

### 2.4.3 使能hdcp1.4

&hdmi {

hdcp1x-enable = <1>;

}

使能hdcp1.4后还需要通过工具烧录对应key，当前工具版本为” ProvisioningTool20170526.zip”具体操作详见工具内的使用文档。

### 2.4.4 打开音频

3368和3288默认hdmi声卡和codec公用，需确认配置如下：

&hdmi\_analog\_sound {

status = "okay";

}

3399目前hdmi声卡和dp公用：

&hdmi\_dp\_sound {

status = "okay";

};

## 2.5 DP配置说明（RK3399）

### 2.5.1 DP检测：

3399的typec支持dp/usb3/usb2, sdk默认使用fusb302来检测接入的设备类型。当设备接入时fusb302通过extcon传递给usb驱动。fusb302是通过i2c外挂的芯片，下面配置是挂到i2c4上时打开的配置，若挂在其他i2c上则需要对应修改。

&i2c4 {

status = "okay";

fusb0: fusb30x@22 {

status = "okay";

};

};

### 2.5.2 绑定typec口

3399有两个功能相同的typec口，都支持dp输出，不过由于dp控制器只有一个所以同一时刻最多只能有一个typec口输出 dp信号。

typec0口包括usb控制器(&usbdrd3\_0); usb3phy(&tcphy0)和usb2phy (&u2phy0);

typec1口包括usb控制器(&usbdrd3\_1);usb3phy(&tcphy1)和usb2phy (&u2phy1);

若fusb302接到typec0口时需配置如下。

&tcphy0 {

extcon = <&fusb0>;

status = "okay";

};

&u2phy0 {

status = "okay";

extcon = <&fusb0>;

};

&usbdrd3\_0 {

extcon = <&fusb0>;

status = "okay";

};

若fusb302接到typec1口时需配置如下

&tcphy1{

extcon = <&fusb0>;

status = "okay";

};

&u2phy1 {

status = "okay";

extcon = <&fusb0>;

};

&usbdrd3\_1 {

extcon = <&fusb0>;

status = "okay";

};

### 2.5.3 注册dp 驱动:

打开dp的dts同时绑定extcon配置：

&cdn\_dp {

status = "okay";

extcon = <&fusb0>;

}

### 2.5.4不使用fusb302将typec口固定作dp输出：

有些产品不接fusb302，而将其中一个typec口固定做dp输出，这时需要自己添加一个extcon驱动如vpd0（目前补丁vpd0.patch还未提交）。然后将dp和usb的extcon设置成vpd0如下：

vpd0: virtual-pd0 {

status = "okay";

}

&cdn\_dp {

status = "okay";

extcon = <&vpd0>;

}

&tcphy0 {

extcon = <&vpd0>;

status = "okay";

};

//以上为接到typec0口是的配置，若是接到typec1口则需将tcphy0改为tcphy1如下：

/\*&tcphy1 {

extcon = <&vpd0 >;

status = "okay";

};\*/

## 2.6 参考配置

### 2.6.1 单MIPI屏

参考dts：

RK3288：arch/arm/boot/dts/rk3288-evb-android-rk818-mipi.dts

RK3399：arch/arm64/boot/dts/rockchip/rk3399-evb-rev3-android.dts

#### 2.6.1.1 使能MIPI-DSI Host以及Panel

* RK3288:

&dsi0 {

status = "okay";

rockchip,lane-rate = <1000>;

panel: panel {

compatible = "simple-panel-dsi";

reg = <0>;

enable-gpios = <&gpio7 4 GPIO\_ACTIVE\_HIGH>;

power-supply = <&vcc\_lcd>;

dsi,flags = <(MIPI\_DSI\_MODE\_VIDEO | MIPI\_DSI\_MODE\_VIDEO\_BURST)>;

dsi,format = <MIPI\_DSI\_FMT\_RGB888>;

dsi,lanes = <4>;

reset-delay-ms = <20>;

init-delay-ms = <20>;

enable-delay-ms = <120>;

prepare-delay-ms = <120>;

status = "okay";

display\_timings: display-timings {

native-mode = <&timing0>;

timing0: timing0 {

clock-frequency = <150000000>;

hactive = <1200>;

vactive = <1920>;

hback-porch = <80>;

hfront-porch = <81>;

vback-porch = <21>;

vfront-porch = <21>;

hsync-len = <10>;

vsync-len = <3>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

};

* RK3399:

&dsi {

status = "okay";

panel@0 {

compatible = "simple-panel-dsi";

reg = <0>;

backlight = <&backlight>;

enable-gpios = <&gpio1 13 GPIO\_ACTIVE\_HIGH>;

dsi,flags = <(MIPI\_DSI\_MODE\_VIDEO | MIPI\_DSI\_MODE\_VIDEO\_BURST)>;

dsi,format = <MIPI\_DSI\_FMT\_RGB888>;

dsi,lanes = <4>;

panel-init-sequence = [

05 64 01 11

05 14 01 29

];

panel-exit-sequence = [

05 64 01 28

05 96 01 10

];

display\_timings: display-timings {

native-mode = <&timing0>;

timing0: timing0 {

clock-frequency = <150000000>;

hactive = <1200>;

vactive = <1920>;

hback-porch = <21>;

hfront-porch = <120>;

vback-porch = <18>;

vfront-porch = <21>;

hsync-len = <20>;

vsync-len = <3>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

};

Note: 如需添加初始化命令，参考2.3章节。

#### 2.6.1.2 使能MIPI-DPHY

&mipi\_dphy {

status = "okay"

};

**Note：**只适用于使用Non-SNPS PHY作为DPHY的Soc系列，比如RK3366/RK3368。对于使用SNPS PHY作为DPHY的Soc系统，比如RK3288/RK3399，不需要配置该node。

#### 2.6.1.3 uboot-logo 开启配置

* RK3288:

&route\_dsi0 {

status = "okay"

};

* RK3399:

&route\_dsi {

status = "okay"

};

#### 2.6.1.4 uboot指定DSI对应 VOP

* RK3288:

DSI连接vopl

&route\_dsi0 {

connect = <&vopl\_out\_dsi0>;

};

DSI连接vopb

&route\_dsi0 {

connect = <&vopb\_out\_dsi0>;

};

* RK3399:

DSI连接vopl

&route\_dsi {

connect = <&vopl\_out\_dsi>;

};

DSI连接vopb

&route\_dsi {

connect = <&vopb\_out\_dsi>;

};

#### 2.6.1.5 kernel指定DSI对应 VOP

* RK3288：

DSI连接VOPL

&dsi0\_in\_vopl {

status = "okay";

};

&dsi0\_in\_vopb {

status = "disabled";

};

DSI连接VOPB

&dsi0\_in\_vopb {

status = "okay";

};

&dsi0\_in\_vopl {

status = "disabled";

};

* RK3399

DSI连接VOPL

&dsi\_in\_vopl {

status = "okay";

};

&dsi\_in\_vopb {

status = "disabled";

};

DSI连接VOPB

&dsi\_in\_vopb {

status = "okay";

};

&dsi\_in\_vopl {

status = "disabled";

};

### 2.6.2 双MIPI单屏

参考dts:

RK3288: arch/arm/boot/dts/rk3288-x7811-rk818-dual-dsi.dts

#### 2.6.2.1 MIPI-DSI DSI0 以及Panel

&dsi0 {

status = "okay";

rockchip,dual-channel = <&dsi1>;

rockchip,lane-rate = <900>;

panel: panel@0 {

compatible = "simple-panel-dsi";

reg = <0>;

backlight = <&backlight>;

reset-gpios = <&gpio1 24 GPIO\_ACTIVE\_LOW>;

power-supply = <&vcc\_lcd>;

dsi,flags = <(MIPI\_DSI\_MODE\_VIDEO | MIPI\_DSI\_MODE\_VIDEO\_BURST |

MIPI\_DSI\_MODE\_LPM | MIPI\_DSI\_MODE\_EOT\_PACKET)>;

dsi,format = <MIPI\_DSI\_FMT\_RGB888>;

dsi,lanes = <8>;

reset-delay-ms = <120>;

init-delay-ms = <120>;

enable-delay-ms = <120>;

prepare-delay-ms = <120>;

panel-init-sequence = [

05 64 01 11

05 14 01 29

];

panel-exit-sequence = [

05 64 01 28

05 96 01 10

];

disp\_timings: display-timings {

native-mode = <&timing0>;

timing0: timing0 {

clock-frequency = <230000000>;

hactive = <1536>;

vactive = <2048>;

hback-porch = <180>;

hfront-porch = <180>;

vback-porch = <10>;

vfront-porch = <14>;

hsync-len = <48>;

vsync-len = <2>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

};

Note: 相比单MIPI屏主要的改动主要是下面两处：

① 添加了rockchip,dual-channel = <&dsi1>;

② 修改lane的条数dsi,lanes = <8>;

#### 2.6.2.2 MIPI-DSI DSI1

&dsi1 {

status = "okay"

};

#### 2.6.2.3 Uboot LOGO

&route\_dsi0 {

status = "okay"

};

#### 2.6.2.4 uboot指定DSI对应的 VOP

* RK3288:

DSI连接vopl

&route\_dsi0 {

connect = <&vopl\_out\_dsi0>;

};

DSI连接vopb

&route\_dsi0 {

connect = <&vopb\_out\_dsi0>;

};

* RK3399:

DSI连接vopl

&route\_dsi {

connect = <&vopl\_out\_dsi>;

};

DSI连接vopb

&route\_dsi {

connect = <&vopb\_out\_dsi>;

};

#### 2.6.2.5 kernel指定DSI对应的 VOP

* RK3288：

DSI连接VOPL

&dsi0\_in\_vopl {

status = "okay";

};

&dsi0\_in\_vopb {

status = "disabled";

};

DSI连接VOPB

&dsi0\_in\_vopb {

status = "okay";

};

&dsi0\_in\_vopl {

status = "disabled";

};

* RK3399

DSI连接VOPL

&dsi\_in\_vopl {

status = "okay";

};

&dsi\_in\_vopb {

status = "disabled";

};

DSI连接VOPB

&dsi\_in\_vopb {

status = "okay";

};

&dsi\_in\_vopl {

status = "disabled";

};

### 2.6.3 单EDP屏

参考dts配置文件：

Rk3288: arch/arm/boot/dts/rk3288-evb-android-rk818-edp.dts

Rk3399: arch/arm64/boot/dts/rockchip/rk3399-evb-rev3-android-edp.dts

#### 2.6.3.1 使能EDP host

* RK3288：

&edp {

status = "okay";

};

* RK3399 & RK3368：

&edp {

status = "okay";

ports {

edp\_out: port@1 {

reg = <1>;

#address-cells = <1>;

#size-cells = <0>;

edp\_out\_panel: endpoint@0 {

reg = <0>;

remote-endpoint = <&panel\_in\_edp>;

};

};

};

};

Note: 3288中的ports结构在rk3288-evb.dtsi中已经有定义，所以无需重复添加。

#### 2.6.3.2 使能EDP phy

* RK3288 & RK3368：

&edp\_phy {

status = "okay"

};

Note: rk3399 无需配置该项

#### 2.6.3.3 配置EDP panel

* RK3288：

&edp\_panel {

compatible = "panel-simple";

enable-gpios = <&gpio7 4 GPIO\_ACTIVE\_HIGH>;

enable-delay-ms = <120>;

pinctrl-0 = <&lcd\_cs>;

power-supply = <&vcc\_lcd>;

status = "status";

disp\_timings: display-timings {

native-mode = <&timing0>;

timing0: timing0 {

clock-frequency = <200000000>;

hactive = <1536>;

vactive = <2048>;

hfront-porch = <12>;

hsync-len = <16>;

hback-porch = <48>;

vfront-porch = <8>;

vsync-len = <4>;

vback-porch = <8>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

* RK3399：

edp\_panel: edp-panel {

compatible = "lg,lp079qx1-sp0v";

backlight = <&backlight>;

power-supply = <&vcc3v3\_s0>;

enable-gpios = <&gpio1 13 GPIO\_ACTIVE\_HIGH>;

ports {

panel\_in\_edp: endpoint {

remote-endpoint = <&edp\_out\_panel>;

};

};

};

#### 属性说明：

|  |  |  |
| --- | --- | --- |
| **Property** | **Value** | **Comment** |
| backlight | &backlight | 引用背光节点，panel驱动会对背光进行处理。 |
| enable-gpios | &gpio7 4 GPIO\_ACTIVE\_HIGH | 屏的enable脚的gpio配置，参考原理图。 |
| enable-delay-ms | 120 | Enable信号有效以后的延时，但是ms |
| power-supply | &vcc\_lcd或&vcc3v3\_s0 | Sdk屏的3.0供电，这个由lcd\_en来控制。这个是SDK特有的控制手段，客户一般无需配置。参考原理图。 |
| pinctrl-0 | &lcd\_cs | 这个就是enable-gpios，我们iomux存在复用，加这个是为了确认为gpio。不加也可以，我默认使用为gpio |

Note:

① 3288 edp panel的backlight 在rk3288-android.dtsi里面已有包含，所以这里无需重复添加。

② 3399的配置中没有disp\_timings的结构。这个是在drivers/gpu/drm/panel/panel-simple.c 中定义相应的结构。详细介绍见2.2章节。

#### 2.6.3.4 开启 Uboot LOGO

&route\_edp {

status = "okay"

};

#### 2.6.3.5 uboot中DSI挂在VOP

RK3288默认指定uboot中EDP对应的VOP为vopl。而RK3399默认指定uboot中EDP对应的VOP为vopb。

连接vopb时：

&route\_edp {

connect = <&vopb\_out\_edp>;

};

连接vopl时：

&route\_edp {

connect = <&vopl\_out\_edp>;

};

#### 2.6.3.6 kernel中DSI挂载VOP

* RK3288：

RK3288 SDK kernel中默认接vopl， 在rk3288-android.dtsi中有把edp\_in\_vopb给关掉。若改成接vopb，需要把edp\_in\_vopl给关闭，把edp\_in\_vopb 开启如下所示：

&edp\_in\_vopl {

status = "disabled";

};

&edp\_in\_vopb {

status = "okay";

};

* RK3399：

RK3399 接vopb只需如下配置：

&edp\_in\_vopl {

status = "disabled";

};

### 2.6.4 两个单MIPI屏

#### 2.6.4.1 两个相同MIPI屏接一个VOP

配置与2.3.2双MIPI单屏一致。请参考该章节

#### 2.6.4.2 两个相同MIPI屏接两个VOP

暂缺

#### 2.6.4.3 两个不同MIPI屏接两个VOP

不支持。

### 2.6.5 单LVDS屏

参考dts：

RK3288: arch/arm/boot/dts/rk3288-evb-android-rk818-lvds.dts

**RK3399: RK3399无LVDS接口，需外接转换IC**

#### 2.6.5.1 配置LVDS host

&lvds {

status = "okay";

};

#### 2.6.5.2 配置LVDS Panel

&lvds\_panel {

status = "okay";

compatible ="simple-panel";

backlight = <&backlight>;

bus-format = <MEDIA\_BUS\_FMT\_RGB666\_1X18>;

enable-gpios = <&gpio7 4 GPIO\_ACTIVE\_HIGH>;

enable-delay-ms = <10>;

power-supply = <&vcc\_lcd>;

rockchip,data-mapping = "jeida";

rockchip,data-width = <24>;

rockchip,output = "lvds";

display-timings {

native-mode = <&timing0>;

timing0: timing0 {

clock-frequency = <71000000>;

hactive = <1280>;

vactive = <800>;

hback-porch = <100>;

hfront-porch = <18>;

vback-porch = <8>;

vfront-porch = <6>;

hsync-len = <10>;

vsync-len = <2>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

#### 属性说明：

|  |  |  |
| --- | --- | --- |
| **Property** | **Value** | **Comment** |
| backlight | &backlight | 引用背光节点，panel驱动会对背光进行处理。 |
| bus-format | MEDIA\_BUS\_FMT\_RGB666\_1X18 | 主控输出给屏的数据格式 |
| enable-gpios | &gpio7 4 GPIO\_ACTIVE\_HIGH | 屏的enable脚的gpio配置，参考原理图。 |
| enable-delay-ms | 10 | Enable信号有效以后的延时，但是ms |
| rockchip,data-mapping | vesa or jeida | LVDS信号的两种编码方式，具体对应关系参考4.3的data mapping说明。 |
| rockchip,data-width | 18 or 24 or 30 | LVDS的数据位，RGB三个分量都是6bit的填18，RGB三个分量都是8bit的填24，RGB三个分量都是10bit的填30。 |
| rockchip,output | lvds or rgb | LVDS输出的两种模式：  lvds：LVDS屏的配置；  rgb: RGB屏的配置。 |

#### 2.6.5.3 开启uboot logo

&route\_lvds {

status = "okay";

};

#### 2.6.5.4 uboot中lvds挂载vop

接vopb:

&route\_lvds {

connect = <&vopb\_out\_lvds>;

};

接vopl:

&route\_lvds {

connect = <&vopl\_out\_lvds>;

};

#### 2.6.5.5 kernel中lvds挂在vop

接vopb:

&lvds\_in\_vopb {

status = "okay";

};

&lvds\_in\_vopl {

status = "disabled";

};

接vopl:

&lvds\_in\_vopl {

status = "okay";

};

&lvds\_in\_vopb {

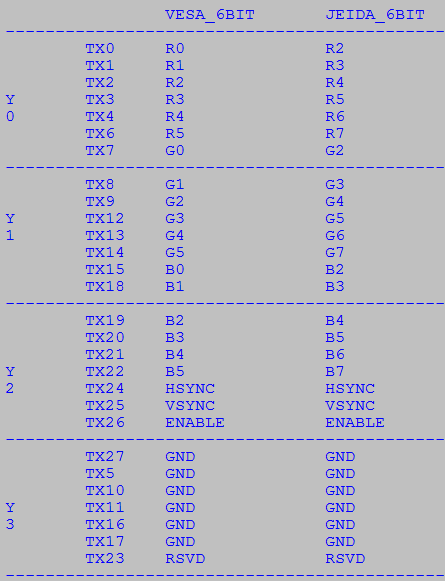
status = "disabled";

};

#### 2.6.5.6 Data\_mapping

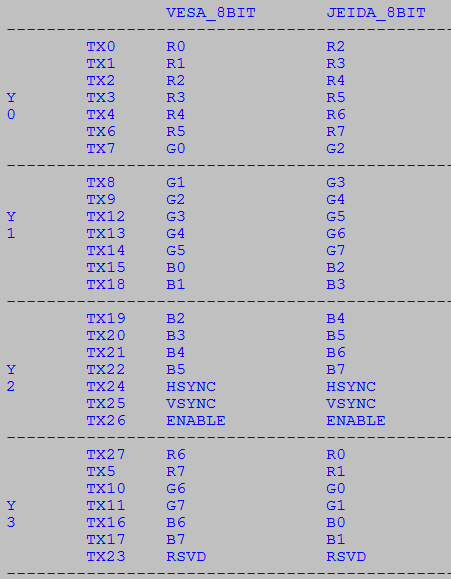
**1）6 bit output mode**

采用4+1的传输模式，即4组数据信号加一组时钟信号，最后一组数据信号传输无效数据。



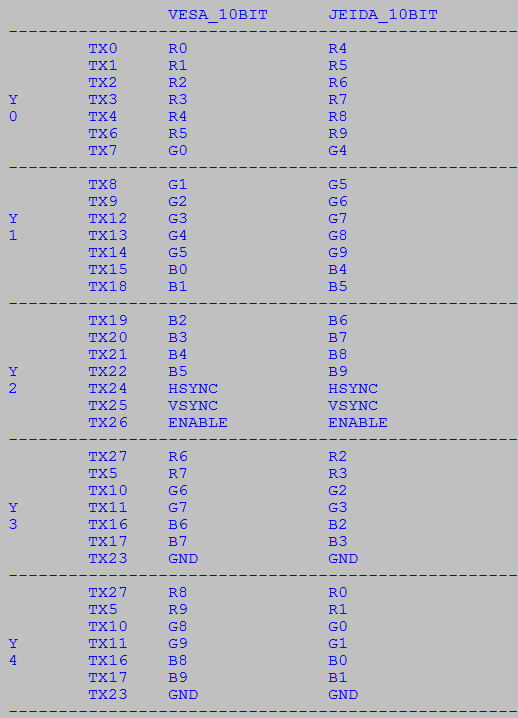
**2）8 bit output mode**

采用4+1的传输模式，即4组数据信号加一组时钟信号。



**3）10 bit output mode**

采用5+1的传输模式，即5组数据信号加一组时钟信号



### 2.6.6 单HDMI(vopb)

* RK3288：

1. hwc需要做修改，我们之前的代码默认只支持MIPI/EDP/LVDS等独显。HDMI独显需要在hardware/rockchip/hwcomposer/drmresources.cpp做如下修改：

diff --git a/drmresources.cpp b/drmresources.cpp

index 9f89687..eb5f23d 100755

--- a/drmresources.cpp

+++ b/drmresources.cpp

@@ -385,12 +385,23 @@ int DrmResources::Init() {

}

}

+ if (!found\_primary) {

+ for (auto &conn : connectors\_) {

+ found\_primary = true;

+ conn->set\_display\_possible(conn->possible\_displays() | HWC\_DISPLAY\_PRIMARY\_BIT);

+ SetPrimaryDisplay(conn.get());

+ break;

+ }

+ }

+

if (!found\_primary) {

ALOGE("failed to find primary display\n");

return -ENODEV;

}

for (auto &conn : connectors\_) {

+ if (GetConnectorFromType(HWC\_DISPLAY\_PRIMARY) == conn.get())

+ continue;

if (!(conn->possible\_displays() & HWC\_DISPLAY\_EXTERNAL\_BIT))

continue;

if (conn->state() != DRM\_MODE\_CONNECTED)

在上面修改的前提下，再在kernel中做如下配置：

/\* 打开hdmi设备节点 \*/

&hdmi {

status = "okay";

};

/\* 绑定hdmi到vopb \*/

&hdmi\_in\_vopl {

status = "disabled";

};

&hdmi\_in\_vopb {

status = "okay";

};

/\* 开启hdmi的uboot logo显示 \*/

&route\_hdmi {

status = "okay";

};

注：目前3288暂不支持hdmi uboot logo显示，还没做验证。

* RK3399：

/\* 打开hdmi设备节点 \*/

&hdmi {

status = "okay";

};

/\* 绑定hdmi到vopb \*/

&hdmi\_in\_vopl {

status = "disabled";

};

/\* 开启hdmi的uboot logo显示 \*/

&route\_hdmi {

status = "okay";

};

/\* 关闭vopl \*/

&vopl{

status = disabled;

};

RK3399还需将vopb时钟挂到vpll上如下：（其他芯片不需要配置）

&vopb {

assigned-clocks = <&cru DCLK\_VOP0\_DIV>;

assigned-clock-parents = <&cru PLL\_VPLL>;

};

### 2.6.7 单VGA（基于rk3288 PopMatal板）

参考dts: arch/arm/boot/dts/rk3288-popmetal.dts

因为是由RGB经过数模转换成VGA输出，所以对于我们主控而言输出的是RGB数据，目前的做法是当作一个可以读取EDID信息的RGB屏来调试。RGB目前在平台上跟LVDS统一，所以配置都是基于LVDS。

关于VGA热插拔时的分辨率切换，发现是在R, G, B 三基色对应的pin插入的时候，会触发I2C的读写操作。发现R、G、B三基色对应的pin的电压值在插入的情况下比空载的情况下降低一倍。由此可以了解VGA热插拔实现分辨率切换的检测机制。

#### 2.6.7.1 打开设备节点

&lvds {

status = "okay";

};

#### 2.6.7.2 配置panel

&lvds\_panel {

status = "okay";

compatible ="simple-panel";

enable-gpios = <&gpio7 21 GPIO\_ACTIVE\_HIGH>;

enable-delay-ms = <10>;

power-supply = <&vcc\_lcd>;

ddc-i2c-bus = <&i2c2>;

rockchip,output = "rgb";

display-timings {

native-mode = <&timing0>;

timing0: timing0 {

clock-frequency = <74250000>;

hactive = <1280>;

vactive = <720>;

hback-porch = <220>;

hfront-porch = <120>;

vback-porch = <20>;

vfront-porch = <5>;

hsync-len = <40>;

vsync-len = <5>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

#### 属性说明：

|  |  |  |
| --- | --- | --- |
| **Property** | **Value** | **Comment** |
| enable-gpios | &gpio7 4 GPIO\_ACTIVE\_HIGH | 屏的enable脚的gpio配置，参考原理图。 |
| enable-delay-ms | 10 | Enable信号有效以后的延时，但是ms |
| power-supply | &vcc\_lcd | Sdk板子独有的供电方式，客户一般都gpio操作 |
| ddc-i2c-bus | &i2c2 | 读取edid的i2c通道 |
| rockchip,output | lvds or rgb | LVDS输出的两种模式：  lvds：LVDS屏的配置；  rgb: RGB屏的配置。 |

#### 2.6.7.3 开启uboot 显示

&route\_lvds {

status = "okay";

};

#### 2.6.7.4 uboot中连接vop

接vopb:

&route\_lvds {

connect = <&vopb\_out\_lvds>;

};

接vopl:

&route\_lvds {

connect = <&vopl\_out\_lvds>;

};

#### 2.6.7.5 kernel中选择连接vop

接vopb:

&lvds\_in\_vopb {

status = "okay";

};

&lvds\_in\_vopl {

status = "disabled";

};

接vopl:

&lvds\_in\_vopl {

status = "okay";

};

&lvds\_in\_vopb {

status = "disabled";

};

### 2.6.8 EDP(vopb) + HDMI(vopl)

/\* 打开edp设备节点 \*/

&edp {

status = "okay";

};

/\* 绑定edp到vopb \*/

&edp\_in\_vopl {

status = "disabled";

};

/\* 开启edp的uboot logo显示 \*/

&route\_edp {

status = "okay"

};

/\* 打开hdmi设备节点 \*/

&hdmi {

status = "okay";

};

/\* 绑定hdmi到vopl \*/

&hdmi\_in\_vopb {

status = "disabled";

};

RK3399还需将hdmi绑定的vopl时钟挂到vpll上如下：（其他芯片不需要配置）

&vopb {

assigned-clocks = <&cru DCLK\_VOP0\_DIV>;

assigned-clock-parents = <&cru PLL\_CPLL>;

};

&vopl {

assigned-clocks = <&cru DCLK\_VOP1\_DIV>;

assigned-clock-parents = <&cru PLL\_VPLL>;

};

### 2.6.9 LVDS(vopl) + HDMI(vopb)

//使能LVDS

&lvds {

status = "okay";

};

//绑定LVDS到vopl

&lvds\_in\_vopb {

status = "disabled";

};

//LVDS屏显示uboot logo

&route\_lvds{

connect = <&vopl\_out\_lvds>;

status = "okay"

};

//使能HDMI

&hdmi {

status = "okay";

};

//绑定HDMI到vopb

&hdmi\_in\_vopl {

status = "disabled";

};

RK3399还需将hdmi绑定的vopb时钟挂到vpll上如下：（其他芯片不需要配置）

&vopb {

assigned-clocks = <&cru DCLK\_VOP0\_DIV>;

assigned-clock-parents = <&cru PLL\_VPLL>;

};

&vopl {

assigned-clocks = <&cru DCLK\_VOP1\_DIV>;

assigned-clock-parents = <&cru PLL\_CPLL>;

};

### 2.6.10 MIPI(vopb) + HDMI(vopl)

&mipi {

status = "okay";

};

&mipi\_in\_vopl {

status = "disabled";

};

&hdmi {

status = "okay";

};

&hdmi\_in\_vopb {

status = "disabled";

};

&route\_mipi{

status = "okay"

};

RK3399还需将hdmi绑定的vopl时钟挂到vpll上如下：（其他芯片不需要配置）

&vopb {

assigned-clocks = <&cru DCLK\_VOP0\_DIV>;

assigned-clock-parents = <&cru PLL\_CPLL>;};

&vopl {

assigned-clocks = <&cru DCLK\_VOP1\_DIV>;

assigned-clock-parents = <&cru PLL\_VPLL>;

};

### 2.6.11 HDMI(vopb) + DP(vopl)

&hdmi {

status = "okay";

};

&hdmi\_in\_vopl {

status = "disabled";

};

&cdn\_dp {

status = "okay";

};

&dp\_in\_vopl {

status = "disabled";

};

&route\_hdmi {

status = "okay"

};

RK3399还需将hdmi绑定的vopb时钟挂到vpll上如下：（其他芯片不需要配置）

&vopb {

assigned-clocks = <&cru DCLK\_VOP0\_DIV>;

assigned-clock-parents = <&cru PLL\_VPLL>;

};

&vopl {

assigned-clocks = <&cru DCLK\_VOP1\_DIV>;

assigned-clock-parents = <&cru PLL\_CPLL>;

};

### 2.6.12 HDMI + CVBS（单VOP）

#### 2.6.12.1 非外接CVBS的平台

RK3228、RK3328内部集成了HDMI、CVBS，参考如下配置：

/\* 打开hdmi设备节点 \*/

&hdmi {

status = "okay";

};

/\* 开启hdmi的uboot logo显示 \*/

&route\_hdmi {

status = "okay"

};

/\* 打开TVE设备节点，输出CVBS \*/

&tve {

status = "okay"

};

#### 2.6.12.2 使用RK1000输出CVBS的平台

/\* 打开hdmi设备节点 \*/

&hdmi {

status = "okay";

};

/\* 开启hdmi的uboot logo显示 \*/

&route\_hdmi {

status = "okay"

};

RK3368使用通过I2C通信的外接芯片RK1000输出CVBS，由于RK1000的CVBS挂载在LVDS下，所以需要使能LVDS，并创建port供TVE挂载，相关详细说明参考：Documentation/devicetree/bindings/media/video-interfaces.txt

&lvds {

status = "okay";

ports {

lvds\_out: port@1 {

reg = <1>;

#address-cells = <1>;

#size-cells = <0>;

lvds\_out\_tve: endpoint@0 {

reg = <0>;

remote-endpoint = <&tve\_in\_lvds>;

};

};

};

};

RK1000通过I2C进行通信，所以对应的I2C必须开启（下面例子为I2C1）。RK1000的cvbs显示需要开启两个模块：rk1000\_ctl和rk1000-tve模块。CTL也即CORE CTRL，为I2C的总控制模块，TVE也即TV ENCODER，为CVBS控制模块。rk1000-tve必须挂载在lvds下（红字部分）。

&i2c1 {

status = "okay";

rk1000\_ctl: rk1000-ctl@40 {

compatible = "rockchip,rk1000-ctl";

status = "okay";

reg = <0x40>;

reset-gpios = <&gpio0 1 GPIO\_ACTIVE\_LOW>;

clocks = <&cru SCLK\_I2S\_8CH\_OUT>;

clock-names = "mclk";

pinctrl-names = "default";

pinctrl-0 = <&i2s\_8ch\_mclk>;

};

rk1000-tve@42 {

status = "okay";

compatible = "rockchip,rk1000-tve";

reg = <0x42>;

rockchip,data-width = <24>;

rockchip,output = "rgb";

rockchip,ctl = <&rk1000\_ctl>;

ports {

#address-cells = <1>;

#size-cells = <0>;

tve\_in: port@0 {

reg = <0>;

tve\_in\_lvds: endpoint {

remote-endpoint = <&lvds\_out\_tve>;

};

};

};

};

};

参数说明如下：

|  |  |  |
| --- | --- | --- |
| property | value | comment |
| reset-gpios | &gpio0 1 GPIO\_ACTIVE\_LOW | 配置RK1000的reset引脚，参考原理图 |
| clocks | &cru SCLK\_I2S\_8CH\_OUT | RK1000 CORE CTRL的I2C通信需要的mclk。 |
| clock-names | mclk | 时钟名称 |
| pinctrl-0 | &i2s\_8ch\_mclk | mclk的pinctrl |
| pinctrl-names | default | mclk的pinctrl名称 |
| rockchip,data-width | 24 | 输出色彩格式，必须为24或18 |
| rockchip,output | rgb | 输出格式，必须为rgb,lvd, duallvds |
| rockchip,ctl | &rk1000\_ctl | 引用RK1000 CORE CTRL |

### 2.6.13 HDMI（vopb）+ CVBS（vopl）

暂缺。

### 2.6.14 MIPI + EDP（3288需要先配置build.prop）

对于双LCD显示，我们建议先单独调试好MIPI屏和EDP屏，再去做双LCD显示。这样步骤流程更清晰，遇到问题也便于定位。

参考dts:

RK3288:arch/arm/boot/dts/rk3288-evb-android-rk818-mipi-edp.dts

RK3399:arch/arm64/boot/dts/rockchip/rk3399-evb-rev3-android-mipi-edp.dts

#### 2.6.14.1 开启mipi以及配置mipi panel

保持事先调试好的单MIPI屏的配置即可。

* RK3288:

&dsi0 {

status = "okay";

rockchip,lane-rate = <1000>;

panel: panel {

compatible = "simple-panel-dsi";

reg = <0>;

power-supply = <&vcc\_lcd>;

dsi,flags = <(MIPI\_DSI\_MODE\_VIDEO | MIPI\_DSI\_MODE\_VIDEO\_BURST)>;

dsi,format = <MIPI\_DSI\_FMT\_RGB888>;

dsi,lanes = <4>;

reset-delay-ms = <20>;

init-delay-ms = <20>;

enable-delay-ms = <120>;

prepare-delay-ms = <120>;

status = "okay";

display\_timings: display-timings {

native-mode = <&timing0>;

timing0: timing0 {

clock-frequency = <150000000>;

hactive = <1200>;

vactive = <1920>;

hback-porch = <80>;

hfront-porch = <81>;

vback-porch = <21>;

vfront-porch = <21>;

hsync-len = <10>;

vsync-len = <3>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

};

* RK3399:

&dsi {

status = "okay";

panel@0 {

compatible = "simple-panel-dsi";

reg = <0>;

backlight = <&backlight>;

enable-gpios = <&gpio1 13 GPIO\_ACTIVE\_HIGH>;

dsi,flags = <(MIPI\_DSI\_MODE\_VIDEO | MIPI\_DSI\_MODE\_VIDEO\_BURST)>;

dsi,format = <MIPI\_DSI\_FMT\_RGB888>;

dsi,lanes = <4>;

panel-init-sequence = [

05 64 01 11

05 14 01 29

];

panel-exit-sequence = [

05 64 01 28

05 96 01 10

];

display\_timings: display-timings {

native-mode = <&timing0>;

timing0: timing0 {

clock-frequency = <150000000>;

hactive = <1200>;

vactive = <1920>;

hback-porch = <21>;

hfront-porch = <120>;

vback-porch = <18>;

vfront-porch = <21>;

hsync-len = <20>;

vsync-len = <3>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

};

#### 2.6.14.2 开启edp

* RK3288:

&edp {

status = "okay";

}；

* RK3399:

&edp {

status = "okay";

ports {

edp\_out: port@1 {

reg = <1>;

#address-cells = <1>;

#size-cells = <0>;

edp\_out\_panel: endpoint@0 {

reg = <0>;

remote-endpoint = <&panel\_in\_edp>;

};

};

};

};

Note: 3288中的ports结构在rk3288-evb.dtsi中已经有定义，所以无需重复添加。

#### 2.6.14.3 配置edp panel

* RK3288:

&edp\_panel {

compatible = "simple-panel";

enable-gpios = <&gpio7 4 GPIO\_ACTIVE\_HIGH>;

enable-delay-ms = <120>;

pinctrl-0 = <&lcd\_cs>;

power-supply = <&vcc\_lcd>;

status = "okay";

display-timings {

native-mode = <&F402>;

F402: timing0 {

clock-frequency = <200000000>;

hactive = <1536>;

vactive = <2048>;

hfront-porch = <12>;

hsync-len = <16>;

hback-porch = <48>;

vfront-porch = <8>;

vsync-len = <4>;

vback-porch = <8>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

* RK3399:

edp\_panel: edp-panel {

compatible = "lg,lp079qx1-sp0v", "panel-simple";

backlight = <&backlight1>;

power-supply = <&vcc3v3\_s0>;

ports {

panel\_in\_edp: endpoint {

remote-endpoint = <&edp\_out\_panel>;

};

};

disp\_timings: display-timings {

native-mode = <&F402>;

F402: timing0 {

clock-frequency = <200000000>;

hactive = <1536>;

vactive = <2048>;

hfront-porch = <12>;

hsync-len = <16>;

hback-porch = <48>;

vfront-porch = <8>;

vsync-len = <4>;

vback-porch = <8>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

Note: backlight如果两个panel共用同样的gpio和pwm，那么就无需添加，不然会资源冲突。如果不同，那么backlight后面的&backlight需要和mipi panel中的背光命名区分。Backlight的详细配置见2.3.13.4。

enable-gpios 如果两个panel共用同样的gpio，则无需添加改行。

#### 2.6.14.4 配置edp背光backlight

我们sdk上edp和mipi共用背光设置，所以无需进行下面的配置，当然进行下面的配置也是可以正常显示的。下面的配置是给各自有独立背光的panel设定的。

* RK3288:

backlight1: backlight1 {

compatible = "pwm-backlight";

pwms = <&pwm1 0 25000 0>;

brightness-levels = <

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47

48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63

64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79

80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95

96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111

112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127

128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143

144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159

160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175

176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191

192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207

208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223

224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239

240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255

>;

default-brightness-level = <128>;

enable-gpios = <&gpio7 2 GPIO\_ACTIVE\_HIGH>;

pinctrl-names = "default";

pinctrl-0 = <&bl\_en>;

pwms = <&pwm0 0 1000000 0>;

};

&pwm1 {

status = "okay";

};

&backlight1 {

status = "okay";

};

* RK3399:

backlight1: backlight1 {

compatible = "pwm-backlight";

pwms = <&pwm1 0 25000 0>;

brightness-levels = <

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47

48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63

64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79

80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95

96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111

112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127

128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143

144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159

160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175

176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191

192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207

208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223

224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239

240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255

>;

default-brightness-level = <200>;

};

&pwm1 {

status = "okay";

};

&backlight1 {

status = "okay";

};

Note: 3399和3288相比，少了gpio的配置。因为3399 sdk上背光的gpio和lcd\_en共用，所以这里无需配置。

#### 2.6.14.5 配置VOP连接

* RK3288:

Mipi接vopb, edp接vopl：

&route\_dsi0 {

connect = <&vopb\_out\_dsi0>;

};

&route\_edp {

connect = <&vopl\_out\_edp>;

};

&dsi0\_in\_vopl {

status = "disabled";

}；

&dsi0\_in\_vopb {

status = "okay";

}；

&edp\_in\_vopl {

status = "okay";

}；

&edp\_in\_vopb {

status = "disabled";

}；

Mipi接vopl, edp接vopb：

&route\_dsi0 {

connect = <&vopl\_out\_dsi0>;

};

&route\_edp {

connect = <&vopb\_out\_edp>;

};

&dsi0\_in\_vopl {

status = "okay";

}；

&dsi0\_in\_vopb {

status = "disabled";

}；

&edp\_in\_vopl {

status = "disabled";

}；

&edp\_in\_vopb {

status = "okay";

}；

* RK3399:

Mipi接vopb, edp接vopl：

&route\_dsi {

connect = <&vopb\_out\_dsi>;

};

&route\_edp {

connect = <&vopl\_out\_edp>;

};

&dsi\_in\_vopl {

status = "disabled";

}；

&dsi\_in\_vopb {

status = "okay";

}；

&edp\_in\_vopl {

status = "okay";

}；

&edp\_in\_vopb {

status = "disabled";

}；

Mipi接vopl, edp接vopb：

&route\_dsi {

connect = <&vopl\_out\_dsi>;

};

&route\_edp {

connect = <&vopb\_out\_edp>;

};

&dsi\_in\_vopb {

status = "disabled";

}；

&dsi\_in\_vopl {

status = "okay";

}；

&edp\_in\_vopb {

status = "okay";

}；

&edp\_in\_vopl {

status = "disabled";

}；

Note: 3288 为了HDMI 4K显示，默认HDMI接VOPB，MIPI和EDP默认接VOPL。

#### 2.6.14.6 开启uboot显示

* RK3288：

&route\_dsi0 {

status = "okay";

};

&route\_edp {

status = "okay";

};

* RK3399：

&route\_dsi {

status = "okay";

};

&route\_edp {

status = "okay";

};

### 2.6.15 MIPI + LVDS/RGB（3399不支持LVDS/RGB，需要外接转换IC）

对于双LCD显示，我们建议先单独调试好MIPI屏和LVDS/RGB屏，再去做双LCD显示。这样步骤流程更清晰，遇到问题也便于定位。

参考dts:

暂缺（客户那边按下面配置已经点亮）

注：3288需要先配置build.prop，关于build.prop配置，具体见3.1章节，调试阶段可以按如下所示做配置：

adb root  
 adb remount  
 adb pull system/build.prop  
 修改这个文件，添加  
 sys.hwc.device.primary=DSI  
 sys.hwc.device.extend=LVDS  
 然后

adb push build.prop system  
 adb shell "chmod 644 system/build.prop"   
 adb reboot

#### 2.6.15.1 开启mipi以及配置mipi panel

保持事先调试好的单MIPI屏的配置即可。

* RK3288:

&dsi0 {

status = "okay";

rockchip,lane-rate = <1000>;

panel: panel {

compatible = "simple-panel-dsi";

reg = <0>;

power-supply = <&vcc\_lcd>;

dsi,flags = <(MIPI\_DSI\_MODE\_VIDEO | MIPI\_DSI\_MODE\_VIDEO\_BURST)>;

dsi,format = <MIPI\_DSI\_FMT\_RGB888>;

dsi,lanes = <4>;

reset-delay-ms = <20>;

init-delay-ms = <20>;

enable-delay-ms = <120>;

prepare-delay-ms = <120>;

status = "okay";

display\_timings: display-timings {

native-mode = <&timing0>;

timing0: timing0 {

clock-frequency = <150000000>;

hactive = <1200>;

vactive = <1920>;

hback-porch = <80>;

hfront-porch = <81>;

vback-porch = <21>;

vfront-porch = <21>;

hsync-len = <10>;

vsync-len = <3>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

};

#### 2.6.15.2 开启LVDS

* RK3288:

&lvds {

status = "okay";

}；

#### 2.6.15.3 配置LVDS panel

* RK3288:

&lvds\_panel {

compatible = "simple-panel";

bus-format = <MEDIA\_BUS\_FMT\_RGB666\_1X18>;

enable-gpios = <&gpio7 4 GPIO\_ACTIVE\_HIGH>;

enable-delay-ms = <10>;

power-supply = <&vcc\_lcd>;

rockchip,data-mapping = "jeida";

rockchip,data-width = <24>;

rockchip,output = "lvds";

status = "okay";

display-timings {

native-mode = <&lvds\_panel\_name>;

lvds\_panel\_name: timing0 {

clock-frequency = <71000000>;

hactive = <1280>;

vactive = <800>;

hfront-porch = <18>;

hsync-len = <10>;

hback-porch = <100>;

vfront-porch = <6>;

vsync-len = <2>;

vback-porch = <8>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

Note: backlight如果两个panel共用同样的gpio和pwm，那么就无需添加，不然会资源冲突。如果不同，那么backlight后面的&backlight需要和mipi panel中的背光命名区分。Backlight的详细配置见2.3.14.4。

enable-gpios 如果两个panel共用同样的gpio，则无需添加改行。

#### 2.6.15.4 配置LVDS背光backlight

我们sdk上LVDS和MIPI共用背光设置，所以无需进行下面的配置，当然进行下面的配置也是可以正常显示的。下面的配置是给各自有独立背光的panel设定的。

* RK3288:

backlight1: backlight1 {

compatible = "pwm-backlight";

pwms = <&pwm1 0 25000 0>;

brightness-levels = <

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47

48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63

64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79

80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95

96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111

112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127

128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143

144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159

160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175

176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191

192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207

208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223

224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239

240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255

>;

default-brightness-level = <128>;

enable-gpios = <&gpio7 2 GPIO\_ACTIVE\_HIGH>;

pinctrl-names = "default";

pinctrl-0 = <&bl\_en>;

pwms = <&pwm0 0 1000000 0>;

};

&pwm1 {

status = "okay";

};

&backlight1 {

status = "okay";

};

#### 2.6.15.5 配置VOP连接

* RK3288:

Mipi接vopb, lvds接vopl：

&route\_dsi0 {

connect = <&vopb\_out\_dsi0>;

};

&route\_lvds {

connect = <&vopl\_out\_lvds>;

};

&dsi0\_in\_vopl {

status = "disabled";

}；

&dsi0\_in\_vopb {

status = "okay";

}；

&lvds\_in\_vopl {

status = "okay";

}；

&lvds\_in\_vopb {

status = "disabled";

}；

Mipi接vopl, lvds接vopb：

&route\_dsi0 {

connect = <&vopl\_out\_dsi0>;

};

&route\_lvds {

connect = <&vopb\_out\_lvds>;

};

&dsi0\_in\_vopl {

status = "okay";

}；

&dsi0\_in\_vopb {

status = "disabled";

}；

&lvds\_in\_vopl {

status = "disabled";

}；

&lvds\_in\_vopb {

status = "okay";

}；

#### 2.6.15.6 开启uboot显示

* RK3288：

&route\_dsi0 {

status = "okay";

};

&route\_lvds {

status = "okay";

};

### 2.6.16 EDP + LVDS/RGB（3399不支持LVDS/RGB，需要外接转换IC）

对于双LCD显示，我们建议先单独调试好EDP屏和LVDS/RGB屏，再去做双LCD显示。这样步骤流程更清晰，遇到问题也便于定位。

参考dts:

暂缺

注：3288需要先配置build.prop，关于build.prop配置，具体见3.1章节，调试阶段可以按如下所示做配置：

adb root  
 adb remount  
 adb pull system/build.prop  
 修改这个文件，添加  
 sys.hwc.device.primary=eDP /\*注意e要小写\*/  
 sys.hwc.device.extend=LVDS  
 然后

adb push build.prop system  
 adb shell "chmod 644 system/build.prop"   
 adb reboot

#### 2.6.16.1 开启EDP以及配置EDP panel

保持事先调试好的单EDP屏的配置即可。

* RK3288:

&edp {

status = "okay";

}；

&edp\_panel {

compatible = "simple-panel";

backlight = <&backlight>;

enable-gpios = <&gpio7 4 GPIO\_ACTIVE\_HIGH>;

enable-delay-ms = <120>;

pinctrl-0 = <&lcd\_cs>;

power-supply = <&vcc\_lcd>;

status = "okay";

display-timings {

native-mode = <&timing0>;

timing0: timing0 {

clock-frequency = <200000000>;

hactive = <1536>;

vactive = <2048>;

hfront-porch = <12>;

hsync-len = <16>;

hback-porch = <48>;

vfront-porch = <8>;

vsync-len = <4>;

vback-porch = <8>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

#### 2.6.16.2 开启LVDS

* RK3288:

&lvds {

status = "okay";

}；

#### 2.6.16.3 配置LVDS panel

* RK3288:

&lvds\_panel {

compatible = "simple-panel";

bus-format = <MEDIA\_BUS\_FMT\_RGB666\_1X18>;

enable-gpios = <&gpio7 4 GPIO\_ACTIVE\_HIGH>;

enable-delay-ms = <10>;

power-supply = <&vcc\_lcd>;

rockchip,data-mapping = "jeida";

rockchip,data-width = <24>;

rockchip,output = "lvds";

status = "okay";

display-timings {

native-mode = <&lvds\_panel\_name>;

lvds\_panel\_name: timing0 {

clock-frequency = <71000000>;

hactive = <1280>;

vactive = <800>;

hfront-porch = <18>;

hsync-len = <10>;

hback-porch = <100>;

vfront-porch = <6>;

vsync-len = <2>;

vback-porch = <8>;

hsync-active = <0>;

vsync-active = <0>;

de-active = <0>;

pixelclk-active = <0>;

};

};

};

Note: backlight如果两个panel共用同样的gpio和pwm，那么就无需添加，不然会资源冲突。如果不同，那么backlight后面的&backlight需要和edp panel中的背光命名区分。Backlight的详细配置见2.3.16.4。

enable-gpios 如果两个panel共用同样的gpio，则无需添加改行。

#### 2.6.16.4 配置LVDS背光backlight

我们sdk上LVDS和EDP共用背光设置，所以无需进行下面的配置，当然进行下面的配置也是可以正常显示的。下面的配置是给各自有独立背光的panel设定的。

* RK3288:

backlight1: backlight1 {

compatible = "pwm-backlight";

pwms = <&pwm1 0 25000 0>;

brightness-levels = <

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47

48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63

64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79

80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95

96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111

112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127

128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143

144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159

160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175

176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191

192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207

208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223

224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239

240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255

>;

default-brightness-level = <128>;

enable-gpios = <&gpio7 2 GPIO\_ACTIVE\_HIGH>;

pinctrl-names = "default";

pinctrl-0 = <&bl\_en>;

pwms = <&pwm0 0 1000000 0>;

};

&pwm1 {

status = "okay";

};

&backlight1 {

status = "okay";

};

#### 2.6.16.5 配置VOP连接

* RK3288:

edp接vopb, lvds接vopl：

&route\_edp {

connect = <&vopb\_out\_edp>;

};

&route\_lvds {

connect = <&vopl\_out\_lvds>;

};

&edp\_in\_vopl {

status = "disabled";

}；

&edp\_in\_vopb {

status = "okay";

}；

&lvds\_in\_vopl {

status = "okay";

}；

&lvds\_in\_vopb {

status = "disabled";

}；

edp接vopl, lvds接vopb：

&route\_edp {

connect = <&vopl\_out\_edp>;

};

&route\_lvds {

connect = <&vopb\_out\_lvds>;

};

&edp\_in\_vopl {

status = "okay";

}；

&edp\_in\_vopb {

status = "disabled";

}；

&lvds\_in\_vopl {

status = "disabled";

}；

&lvds\_in\_vopb {

status = "okay";

}；

#### 2.6.16.6 开启uboot显示

* RK3288：

&route\_edp {

status = "okay";

};

&route\_lvds {

status = "okay";

};

# 显示框架配置

当前SDK的显示框架增加了一些系统属性用于帮助客户能够根据需求配置显示。

## 3.1 主副显示器配置

|  |  |
| --- | --- |
| 属性 | 功能说明 |
| sys.hwc.device.primary | 设置显示接口做为主显 |
| sys.hwc.device.extend | 设置显示接口做为副显 |

以上两个属性的配置可加在产品配置目录下的system.prop里（如device/rockchip/rk3368/rk3368\_box/system.prop）。

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

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

sys.hwc.device.primary=HDMI-A

sys.hwc.device.extend=DP

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

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

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

注意：由于主显的framebuffer分辨率无法动态更改，所以有两个或以上设备作为主显时，最好设定一个主显的framebuffer分辨率。设置方法见章节3.3。

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

struct type\_name connector\_type\_names[] = {

{ DRM\_MODE\_CONNECTOR\_Unknown, "unknown" },//未知接口

{ DRM\_MODE\_CONNECTOR\_VGA, "VGA" }, //VGA

{ DRM\_MODE\_CONNECTOR\_DVII, "DVI-I" },//DVI，暂不支持

{ DRM\_MODE\_CONNECTOR\_DVID, "DVI-D" },//DVI，暂不支持

{ DRM\_MODE\_CONNECTOR\_DVIA, "DVI-A" },//DVI，暂不支持

{ DRM\_MODE\_CONNECTOR\_Composite, "composite" },//不支持

{ DRM\_MODE\_CONNECTOR\_SVIDEO, "s-video" },//S端子

{ DRM\_MODE\_CONNECTOR\_LVDS, "LVDS" },//LVDS

{ DRM\_MODE\_CONNECTOR\_Component, "component" },//分量信号YPbPr

{ DRM\_MODE\_CONNECTOR\_9PinDIN, "9-pin DIN" },//不支持

{ DRM\_MODE\_CONNECTOR\_DisplayPort, "DP" },//DP

{ DRM\_MODE\_CONNECTOR\_HDMIA, "HDMI-A" },//HDMI A型口

{ DRM\_MODE\_CONNECTOR\_HDMIB, "HDMI-B" },//HDMI B型口，不支持

{ DRM\_MODE\_CONNECTOR\_TV, "TV" },// CVBS

{ DRM\_MODE\_CONNECTOR\_eDP, "eDP" },//EDP

{ DRM\_MODE\_CONNECTOR\_VIRTUAL, "Virtual" },//不支持

{ DRM\_MODE\_CONNECTOR\_DSI, "DSI" },//MIPI

};

## 3.2 主副显示器接口查询

可以通过以下两个只读属性来分别查询主副显示器的输出接口的名称。

表6‑3主副显示器查询

|  |  |
| --- | --- |
| 属性 | 功能说明 |
| sys.hwc.device.main | 查询当前主显的输出接口 |
| sys.hwc.device.aux | 查询当前副显的输出接口 |

## 3.3 FrameBuffer分辨率配置

可以通过配置以下属性来设置FrameBuffer的分辨率

persist.sys.framebuffer.main=1920x1080

## 3.4 分辨率过滤配置

因为初始获取到的全部分辨率过多，有些分辨率对用户来说并不需要，因此在SDK的HWC模块中对分辨率进行了过滤。

位于device/rockchip/common/resolution\_white.xml路径的配置文件定义了能够通过过滤的白名单，HWC中会根据该配置文件对初始的分辨率进行过滤筛选后再传递给上层，该XML文件的每一个<resolution>块定义了一个能够通过过滤的分辨率，其中详细项的定义如下：

表3‑4分辨率过滤项定义说明

|  |  |
| --- | --- |
| 项定义 | 说明 |
| clock | 时钟 |
| hdisplay | 行有效像素，见图6-1的标示 |
| hsync\_start | 行同步起始像素，见图6-1的标示 |
| hsync\_end | 行同步结束像素，见图6-1的标示 |
| htotal | 一行总像素，见图6-1的标示 |
| hskew | 见图6-1的标示 |
| vdisplay | 帧有效行，见图6-1的标示 |
| vsync\_start | 帧同步开始行，见图6-1的标示 |
| vsync\_end | 帧同步结束行，见图6-1的标示 |
| vtotal | 一帧总行数，见图6-1的标示 |
| vscan | 见图6-1的标示 |
| vrefresh | 显示设备帧率 |
| flags | flags的定义如下：  DRM\_MODE\_FLAG\_PHSYNC (1<<0)  DRM\_MODE\_FLAG\_NHSYNC (1<<1)  DRM\_MODE\_FLAG\_PVSYNC (1<<2)  DRM\_MODE\_FLAG\_NVSYNC (1<<3)  DRM\_MODE\_FLAG\_INTERLACE (1<<4) |
| vic | HDMI标准对应定义的VIC值，如HDMI标准中未定义置0 |

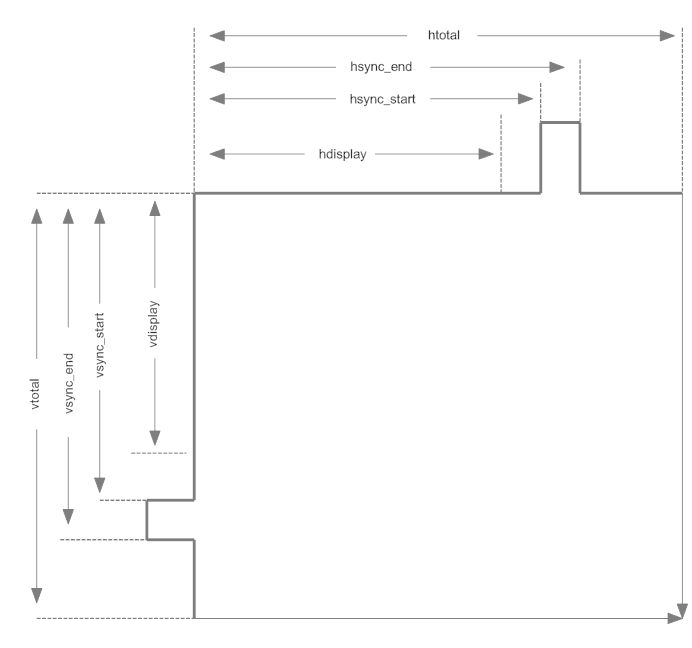


图 3‑1分辨率项定义示意图

# 常用调试方法

## 查看VOP状态

cat /d/dri/0/summary

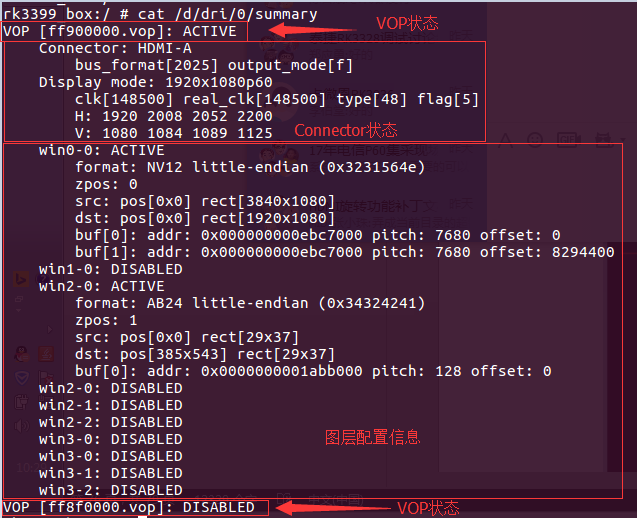


图4-1

图4-1是RK3399连接HDMI时上述命令输出的Log，可以提供三种信息：

* VOP状态： VOPB处于使能状态，VOPL处于禁用状态
* VOP对应的Connector状态：VOPB输出信号给HDMI，bus\_format = 0x2025表示YUV444 8bit，output\_mode = 0x0f表示VOP输出总线为ROCKCHIP\_OUT\_MODE\_AAAA，输出1920x1080P60。

常用的bus\_format由内核uapi/linux/media-bus-format.h定义：

#define MEDIA\_BUS\_FMT\_RGB888\_1X24 0x100a //RGB888

#define MEDIA\_BUS\_FMT\_RGB101010\_1X30 0x1018 //RGB101010

#define MEDIA\_BUS\_FMT\_YUV8\_1X24 0x2025 //YUV444 8bit

#define MEDIA\_BUS\_FMT\_YUV10\_1X30 0x2016 //YUV444 10bit

#define MEDIA\_BUS\_FMT\_UYYVYY8\_0\_5X24 0x2026 //YUV420 8bit

#define MEDIA\_BUS\_FMT\_UYYVYY10\_0\_5X30 0x2027 //YUV420 10bit

常用的output\_mode由内核drivers/gpu/drm/rockchip/rockchip\_drm\_vop.h定义：

#define ROCKCHIP\_OUT\_MODE\_P888 0

#define ROCKCHIP\_OUT\_MODE\_P666 1

#define ROCKCHIP\_OUT\_MODE\_P565 2

#define ROCKCHIP\_OUT\_MODE\_S888 8

#define ROCKCHIP\_OUT\_MODE\_S888\_DUMMY 12

#define ROCKCHIP\_OUT\_MODE\_YUV420 14

/\* for use special outface \*/

#define ROCKCHIP\_OUT\_MODE\_AAAA 15

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

## 查看Connector状态

/sys/class/drm目录下可以看到驱动注册的各个显卡。图4-2是RK3399 BOX平台drm目录结构，可以看到注册了card0-HDMI-A-1和card0-DP-1两种输出，分别表示HDMI和DP。



图4-2

以card0-HDMI-A-1为例，其目录下有以下文件：

* enabled 使能状态
* status 连接状态
* mode 当前输出分辨率
* modes 连接设备支持的分辨率列表
* audioformat 连接设备支持的音频格式
* edid 连接设备的EDID，可以通过命令cat edid > /data/edid.bin保存下来。

## 命令行设置分辨率

# 参考文档

3.1 <https://markyzq.gitbooks.io/rockchip_drm_integration_helper/content/zh/>

3.2 Android drm显示框架介绍.ppt

3.3 kernel的Documentation/devicetree文档