Rockchip RK628 For All Porting 开发指南

文件标识：RK-YH-YF-287

发布版本：V2.6

日期：2024-08-06

文件密级：□绝密 □秘密 □内部资料 ■公开

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd.

地址： 福建省福州市铜盘路软件园 A 区 18 号

网址： www.rock-chips.com

客户服务电话： +86-4007-700-590

客户服务传真： +86-591-83951833

客户服务邮箱： fae@rock-chips.com

前言

RK628 输入支持 RGB / BT1120 / HDMI，输出支持 RGB / BT1120 / LVDS / GVI / DSI / CSI / HDMI 等，具体功能描述参考 datasheet。本文档主要描述 rk628-for-all 软件配置、调试手段以及常见问题处理方式，rk628-for-all 代码期望做到与硬件平台和软件版本无关，还在不断完善中。目前代码包括 MISC（代码路径 drivers/misc/rk628/，支持 RGB / BT1120 / HDMI 输入，RGB / BT1120 / LVDS / GVI / DSI / HDMI 输出）和 MEDIA（代码路径 drivers/media/i2c/rk628/，支持 HDMI 输入，CSI / BT1120 输出）两大部分，分别进行维护。本文档按照不同的模块分别进行介绍和说明。

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

软件开发工程师

修订记录

版本号	作者	修改日期	修改说明
V1.0	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠	2021-08-28	初始发布
V1.1	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠	2021-09-03	新增 DTS DEMO
V1.2	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠	2021-09-17	补充 FAQ 部分
V1.3	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠	2021-09-29	补充 AUDIO 的 FAQ
V1.4	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠	2021-10-08	MISC 新增 HDMIOUT 说明
V1.5	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠	2021-10-27	增加 DSI->CSI 方案说明和 pinctrl 支持
V1.6	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠 / 黄雄山	2021-11-02	增加硬件基础信息
V1.7	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠 / 黄雄山	2021-11-26	支持根据输入源配置 src_mode 参数自动生成 hdmirx 的 edid 数据
V1.8	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠 / 黄雄山	2022-01-15	新增部分 FAQ 及 BT1120 的使用说明
V1.9	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠 / 黄雄山	2022-03-05	补充说明 HDMIRX 支持的频点列表，MISC 新增多 RK628 功能使用说明。

版本号	作者	修改日期	修改说明
V2.0	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠 / 黄雄山	2022-04-10	补充声卡注册失败问题处理和 HDMI-IN 上层问题排查方法。
V2.1	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠 / 黄雄山	2022-12-10	新增部分HDMI-IN和HDMI2GVI 问题的排查思路和方法
V2.2	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠 / 黄雄山 / 黄智斌 / 范建威	2024-01-01	补充对 RK628F 驱动的支持
V2.3	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠 / 黄雄山 /黄智斌 / 范建威	2024-01-03	补充对 RK628H 驱动的支持说明
V2.4	黄国椿 / 陈顺庆 / 兰顺华 / 温定贤 / 罗伟 / 让凌鹏 / 郭立楠 / 黄雄山 /黄智斌 / 范建威	2024-05-22	修复media框架一些相关说明, 增加驱动移植说明章节, 统一 Misc rk628 驱动的接口节点名称
V2.5	黄国椿	2024-07-26	补充对 RK628F/H U-BOOT LOGO 驱动支持说明
V2.6	黄智斌	2024-08-06	补充 GVI 信号调节及展频方式 一些细节修改

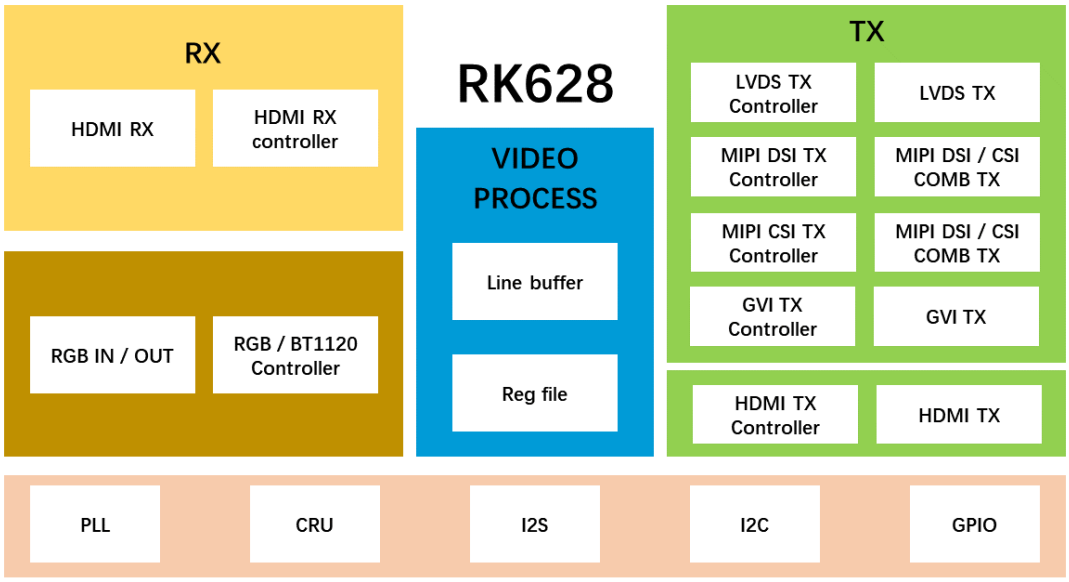
1. 目录

2. 前言

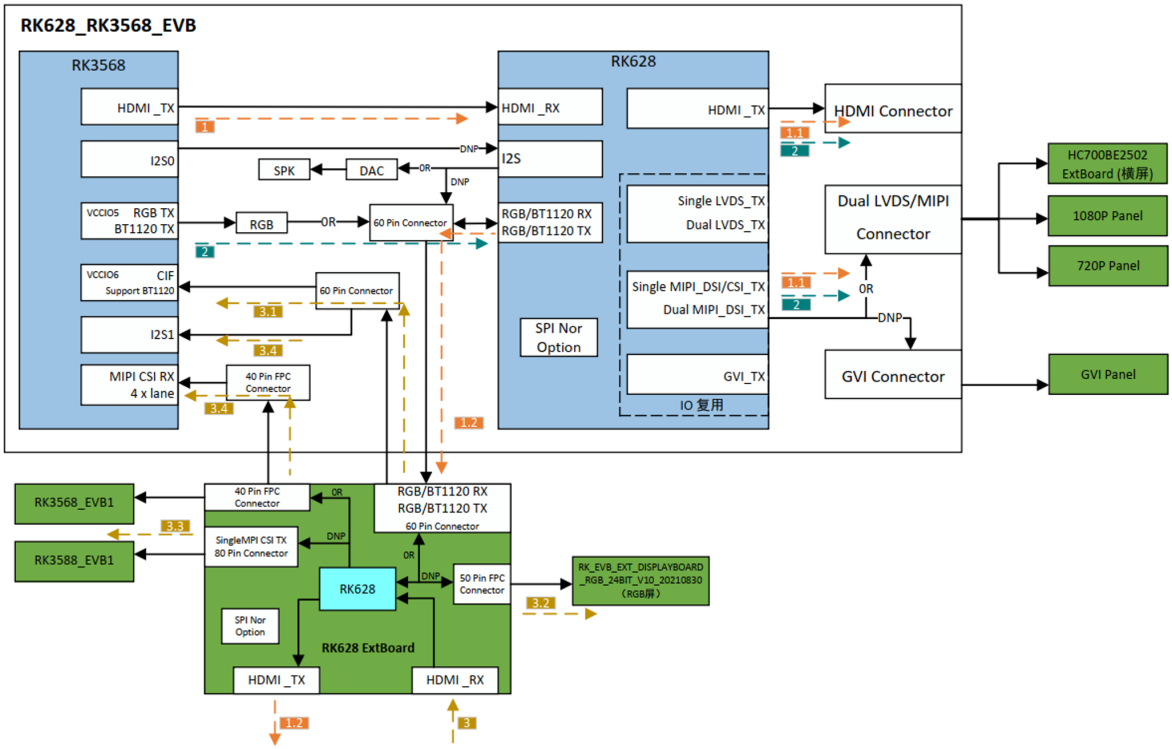
RK628 分为 Display 通路和 HDMI IN 通路，SDK 版本 Display 通路基于 DRM 框架，HDMI IN 通路基于 V4L2 框架，不同的框架或是不同的内核版本需要不同的驱动去适配，而且只适用于 RK 平台。为了适配别的平台，方便驱动移植，推出 For-All 版本驱动，当前驱动支持 RK628D 和 RK628F/H，驱动补丁版本管理参阅 sdk 补丁包的《rk628 驱动版本说明》。

For-All 版本驱动一样也分为 Display 通路和 HDMI IN 通路，Display 通路的驱动于 drivers/misc/rk628/ 下，支持 RGB / BT1120 / HDMI 输入，RGB / BT1120 / LVDS / GVI / DSI / HDMI 输出，HDMI IN 通路的驱动于 drivers/media/i2c/rk628/ 下，支持 HDMI 输入，CSI / BT1120 输出。下面我们介绍的 Misc 和 Media 分别代表这两套驱动，两套代码相互独立，都可单独编译运行。

2.1 RK628 芯片框图



2.2 RK628 典型设计硬件框图



2.3 RK628 主要规格

2.3.1 RK628D 主要规格

Video Iutput Interface	Typical Resolution	Typical Format
HDMI IN	4K@30	YUV420(4K@60) / YUV422 / YUV444 / RGB888
BT1120 IN	1080P@60	YUV422 8bit
RGB IN	1080P@60	RGB888
VIDEO Output Interface	**Typical Resolution**	**Typical Format**
GVI OUT	4K@60	RGB888
MIPI CSI	4K@30	YUV422 8bit
MIPI DSI	1080P@60 / 2.5K@60	RGB888
LVDS OUT	720P@60 / 1080P@60	RGB888
BT1120 OUT	1080P@60	YUV422 8bit
RGB OUT	1080P@60	RGB888
HDMI OUT	1080P@60	RGB888/YUV444 8bit

2.3.2 RK628F 主要规格

Video Iutput Interface	**Typical Resolution**	**Typical Format**
HDMI IN	4K@60	RGB888 / YUV422 8bit / YUV422 10bit / YUV420 8bit (> 4k@30Hz) / YUV420 10bit (> 4k@30Hz)
BT1120 IN	1080P@60	YUV422 8bit
RGB IN	1080P@60	RGB888
VIDEO Output Interface	**Typical Resolution**	**Typical Format**
GVI OUT	4K@60	RGB888
MIPI CSI	双 MIPI，4K@60	YUV422 8bit
MIPI DSI	1080P@60 / 2.5K@60	RGB888
LVDS OUT	720P@60 / 1080P@60	RGB888
BT1120 OUT	1080P@60	YUV422 8bit
RGB OUT	1080P@60	RGB888
HDMI OUT	1080P@60	RGB888 / YUV444 8bit

2.3.3 RK628H 主要规格

对比 **RK628F**，**RK628H** 没有 **GVI** 接口功能。

Video Iutput Interface	Typical Resolution	Typical Format
HDMI IN	4K@60	RGB888 / YUV422 8bit / YUV422 10bit / YUV420 8bit (> 4k@30Hz) / YUV420 10bit (> 4k@30Hz)
BT1120 IN	1080P@60	YUV422 8bit
RGB IN	1080P@60	RGB888
VIDEO Output Interface	**Typical Resolution**	**Typical Format**
MIPI CSI	双 MIPI，4K@60	YUV422 8bit
MIPI DSI	1080P@60 / 2.5K@60	RGB888
LVDS OUT	720P@60 / 1080P@60	RGB888
BT1120 OUT	1080P@60	YUV422 8bit
RGB OUT	1080P@60	RGB888
HDMI OUT	1080P@60	RGB888 / YUV444 8bit

2.4 RK628F/H 与 RK628D 改进及优化对比

功能模块	**RK628D**	**RK628F/H**
HDMIRX	最高支持到 4K@60 YUV420 TMDS CLK 297M	最高可以支持 4K@60 RGB/YUV444 TMDS CLK 594M
	有频点限制，特定分辨率应用需要评估，不一定能满足	支持 27M ~ 594M 任一频点， 满足多样化的分辨率输入
	不支持 DVI MODE	支持 DVI MODE
	无法检测判断输入信号 yuv422/yuv420 视频格式	控制器可以判断输入信号的视频格式
	输入源的 clk lane 幅值低的情况，RK628D rxphy 无法锁定	无该问题
	hdmi rx 存在误码检测	无该问题
MIPI CSI	单通道 4lanes	双通道 8lanes
	最大支持分辨率 4k@30 yuv422 8bit	最大支持分辨率 4k@60 yuv422 8bit
GVI	不同型号的屏存在点不亮和点屏异常	测试多款市面上存在型号的屏暂未发现异常，指导说明见《输出模块配置》-《GVI 输出》-《RK628F GVI 测试型号列表》小节
	需要时钟同源	无需时钟同源
audio	无 audio 的 mclk，接 codec 方案，需要主控提供 mclk	可以提供 mclk
	hdmi 插拔时存在 I2S FIFO 溢出和声道误切换	解决 FIFO 溢出时候，声道顺序不一致问题
csc	不同输入输出接口，部分应用场景存在色域空间转换或图像格式的上下采样，图像色彩少量细节可能会有轻微偏差	无该问题
hdmi tx	需要时钟同源	rgb2hdmi 场景需要同源 hdmi2hdmi 场景不需要同源
多显	支持 rgb/bt1120 和 hdmi tx 同时同分辨率输出	支持 rgb/bt1120、hdmi tx 和 gvi/dsi/csi/lvds 同分辨率同时输出
MCU	没有内置 MCU	内置 MCU

3. 驱动移植说明

For-All 版本驱动分为 Display 通路和 HDMI IN 通路，Display 通路的 u-boot 驱动于 drivers/video/drm/rk628/ 下、内核驱动于 drivers/misc/rk628/ 下，支持 RGB / BT1120 / HDMI 输入，RGB / BT1120 / LVDS / GVI / DSI / HDMI 输出，HDMI IN 通路的驱动于 drivers/media/i2c/rk628/ 下，支持 HDMI 输入，CSI / DSI / BT1120 输出。

可以根据需求，单独移植如下驱动：

3.1 Display 通路

3.1.1 u-boot(drivers/video/drm/rk628)

在 RK 平台 u-boot 中，已经开发 RK628F/H 部分显示通路的驱动如下表，以支持 uboot 开机快速 logo。

IC	Display Route	支持情况
RK628F/H	HDMI2(Dual)DSI	支持
RK628F/H	HDMI2(Dual)LVDS	支持
RK628F	HDMI2GVI	支持
RK628F/H	HDMI2BT1120/RGB	支持
RK628F/H	BT1120/RGB2(Dual)DSI	支持
RK628F/H	BT1120/RGB2(Dual)LVDS	支持
RK628F	BT1120/RGB2GVI	支持

注：u-boot 开机 logo 驱动不对 RK628D 进行支持，因为 HDMIRX 不能工作在任意频点

若为首次添加 RK628 Misc 驱动或 drivers/video/drm 下不存在 RK628 驱动, 则把 u-boot 中 Misc 的 RK628 驱动源码直接拷贝到 drivers/video/drm 下，并修改如下源文件：

```
drivers/video/drm/rockchip_display.c
drivers/video/drm/Kconfig
drivers/video/drm/Makefile
configs/rk3568_defconfig //以rk3568平台为例
```

参考如下修改补丁：

```
diff --git a/drivers/video/drm/rockchip_display.c
b/drivers/video/drm/rockchip_display.c
index a1cb40a000e..041b8bdd84f 100644
--- a/drivers/video/drm/rockchip_display.c
+++ b/drivers/video/drm/rockchip_display.c
@@ -918,6 +918,23 @@ static int display_enable(struct display_state *state)
        if (state->enabled_at_spl == false)
                rockchip_connector_enable(state);

+#ifdef CONFIG_DRM_ROCKCHIP_RK628
+       /*
+        * trigger .probe helper of U_BOOT_DRIVER(rk628) in ./rk628/rk628.c
+        */
```

```
+       struct udevice * dev;
+       int phandle, ret;
+
+       phandle = ofnode_read_u32_default(state->node, "bridge", -1);
+       if (phandle < 0)
+               printf("%s failed to find bridge phandle\n",
ofnode_get_name(state->node));
+
+       ret = uclass_get_device_by_phandle_id(UCLASS_I2C_GENERIC, phandle, &dev);
+       if (ret && ret != -ENOENT)
+               printf("%s:%d failed to get rk628 device ret:%d\n", __func__,
__LINE__, ret);
+
+#endif
+
        if (crtc_state->soft_te)
                crtc_funcs->apply_soft_te(state);
```

```
diff --git a/drivers/video/drm/Kconfig b/drivers/video/drm/Kconfig
index ddc65818cba..c5073b9926b 100644
--- a/drivers/video/drm/Kconfig
+++ b/drivers/video/drm/Kconfig
@@ -191,6 +191,13 @@ config DRM_ROCKCHIP_RK618
          Choose this option to enable support for Rockchip RK618 display
          bridge chips driver. say Y to enable its driver.

+config DRM_ROCKCHIP_RK628
+       bool "Rockchip RK628 display bridge driver"
+       depends on DRM_ROCKCHIP
+       help
+         Choose this option to enable support for Rockchip RK628 display
+         bridge chips driver. say Y to enable its driver.
+
```

```
diff --git a/drivers/video/drm/Makefile b/drivers/video/drm/Makefile
index ea4219d9cf6..f11a8ff4885 100644
--- a/drivers/video/drm/Makefile
+++ b/drivers/video/drm/Makefile
@@ -36,6 +36,7 @@ obj-$(CONFIG_DRM_ROCKCHIP_RGB) += rockchip_rgb.o
 obj-$(CONFIG_DRM_ROCKCHIP_PANEL) += rockchip_panel.o
 obj-$(CONFIG_DRM_ROHM_BU18XL82) += rohm-bu18tl82.o rohm-bu18rl82.o
 obj-$(CONFIG_DRM_ROCKCHIP_RK618) += rk618.o rk618_lvds.o rk618_dsi.o
+obj-$(CONFIG_DRM_ROCKCHIP_RK628) += rk628/
 obj-$(CONFIG_DRM_ROCKCHIP_RK1000) += rk1000.o rk1000_tve.o
 obj-$(CONFIG_DRM_ROCKCHIP_SAMSUNG_MIPI_DCPHY) += samsung_mipi_dcphy.o
 obj-$(CONFIG_DRM_SII902X) += sii902x.o
```

```
diff --git a/configs/rk3568_defconfig b/configs/rk3568_defconfig
index acd1fb18169..94f1050eb4f 100644
--- a/configs/rk3568_defconfig
+++ b/configs/rk3568_defconfig
@@ -224,3 +224,4 @@ CONFIG_RK_AVB_LIBAVB_USER=y
 CONFIG_OPTEE_CLIENT=y
 CONFIG_OPTEE_V2=y
 CONFIG_OPTEE_ALWAYS_USE_SECURITY_PARTITION=y
+CONFIG_DRM_ROCKCHIP_RK628=y
```

最后，使能 u-boot 开机 logo 显示功能需要在 dts 中配置如下：

以 RGB 作为输入：

```
 &route_rgb {
        status = "okay";
+
+       bridge = <&i2c2_rk628>;
 };
```

以 HDMI 作为输入：

```
&route_hdmi {
        status = "okay";
+
+       bridge = <&i2c2_rk628>;
        force-bus-format = <MEDIA_BUS_FMT_RGB888_1X24>;
        force-output;
        force_timing{
```

为了保证 uboot logo 在跳转 kernel logo 时过渡没有闪烁，内核要确保导入如下提交信息的补丁：

```
misc: rk628: add support uboot logo

Type: Function
Redmine ID: #496880 #491750
Associated modifications: NULL
Test: NULL

Change-Id: I311e2c9682f835b377ea9082e1d3a88688167172
Signed-off-by: Guochun Huang <hero.huang@rock-chips.com>
```

3.1.2 kernel（drivers/misc/rk628）

在 RK 平台内核中，若为首次添加 RK628 Misc 驱动或 drivers/misc 下不存在 RK628 驱动，则把内核中 Misc 的 RK628 驱动源码直接拷贝到 drivers/misc 下，并修改 drivers/misc/Kconfig、drivers/misc/Makefile、arch/arm64/configs/rockchip_defconfig，添加 rk628 的编译即可。参考如下补丁：

```
diff --git a/drivers/misc/Kconfig b/drivers/misc/Kconfig
index 276c7c4fef15..8481b9613bc9 100644
--- a/drivers/misc/Kconfig
+++ b/drivers/misc/Kconfig
@@ -5,6 +5,8 @@

 menu "Misc devices"

+source "drivers/misc/rk628/Kconfig"
+
 config RK803
        tristate "RK803"
        default n
```

```
diff --git a/drivers/misc/Makefile b/drivers/misc/Makefile
index b296e760fd47..14642c54a2bc 100644
--- a/drivers/misc/Makefile
+++ b/drivers/misc/Makefile
@@ -3,6 +3,7 @@
 # Makefile for misc devices that really don't fit anywhere else.
 #

+obj-y                          += rk628/
 obj-$(CONFIG_RK803)            += rk803.o
 obj-y                          += rockchip/
 obj-$(CONFIG_LT7911D_FB_NOTIFIER) += lt7911d-fb-notifier.o
```

```
diff --git a/arch/arm64/configs/rockchip_defconfig
b/arch/arm64/configs/rockchip_defconfig
index 28340b4bdfae..dbb7d9e55b62 100644
--- a/arch/arm64/configs/rockchip_defconfig
+++ b/arch/arm64/configs/rockchip_defconfig
@@ -294,6 +294,8 @@ CONFIG_BLK_DEV_LOOP_MIN_COUNT=16
 CONFIG_BLK_DEV_RAM=y
 CONFIG_BLK_DEV_RAM_SIZE=8192
 CONFIG_BLK_DEV_NVME=y
+CONFIG_RK628_MISC=y
+CONFIG_RK628_MISC_HDMITX=y
 CONFIG_LT7911D_FB_NOTIFIER=y
 CONFIG_SRAM=y
 CONFIG_UID_SYS_STATS=y
@@ -557,7 +559,6 @@ CONFIG_ROCKCHIP_THERMAL=y
 CONFIG_WATCHDOG=y
 CONFIG_DW_WATCHDOG=y
 CONFIG_MFD_RK618=y
-CONFIG_MFD_RK628=y
 CONFIG_MFD_RK630_I2C=y
 CONFIG_MFD_RK806_I2C=y
 CONFIG_MFD_RK806_SPI=y
@@ -640,7 +641,6 @@ CONFIG_ROCKCHIP_LVDS=y
 CONFIG_ROCKCHIP_RGB=y
 CONFIG_ROCKCHIP_DW_HDCP2=y
 CONFIG_DRM_ROCKCHIP_RK618=y
-CONFIG_DRM_ROCKCHIP_RK628=y
 CONFIG_DRM_PANEL_SIMPLE=y
 CONFIG_DRM_PANEL_MAXIM_MAX96752F=y
```

```
CONFIG_DRM_PANEL_MAXIM_MAX96772=y
```

需要注意：CONFIG_MFD_RK628 以及 CONFIG_DRM_ROCKCHIP_RK628 为旧版本的 RK628 驱动，在 rockchip_defconfig 文件中需要将其删除。

Misc 的驱动默认都会编译，只需要通过配置 dts 选择运行对应的驱动。

3.2 HDMI IN 通路（drivers/media/i2c/rk628）

kernel-5.10 之前的版本，使用 rk628_for-all 的补丁包，kernel-5.10 的版本，使用 kernel_5.10_rk628_media_patch 的补丁包。把 Media 相关的 rk628 驱动直接拷贝到 drivers/media/i2c/rk628/ 下，并修改 drivers/media/i2c/Kconfig、drivers/media/i2c/Makefile、arch/arm64/configs/rockchip_defconfig，添加 rk628 的编译即可。参考如下补丁：

```
diff --git a/drivers/media/i2c/Kconfig b/drivers/media/i2c/Kconfig
index f3273756c612..609bb5e3db07 100644
--- a/drivers/media/i2c/Kconfig
+++ b/drivers/media/i2c/Kconfig
@@ -414,6 +414,8 @@ config VIDEO_OTP_EEPROM
         help
           This driver supports OTP load from eeprom.

+source "drivers/media/i2c/rk628/Kconfig"
+
 config VIDEO_SAA7110
         tristate "Philips SAA7110 video decoder"
         depends on VIDEO_V4L2 && I2C
```

```
diff --git a/drivers/media/i2c/Makefile b/drivers/media/i2c/Makefile
index 26edd6f7590e..d6f4128b19a5 100644
--- a/drivers/media/i2c/Makefile
+++ b/drivers/media/i2c/Makefile
@@ -141,6 +141,7 @@ obj-$(CONFIG_VIDEO_ML86V7667)       += ml86v7667.o
 obj-$(CONFIG_VIDEO_OV2659)      += ov2659.o
 obj-$(CONFIG_VIDEO_TC358743)    += tc358743.o
 obj-$(CONFIG_VIDEO_TC35874X)    += tc35874x.o
+obj-$(CONFIG_VIDEO_RK628)       += rk628/
 obj-$(CONFIG_VIDEO_AR0230)      += ar0230.o
 obj-$(CONFIG_VIDEO_GC02M2)      += gc02m2.o
 obj-$(CONFIG_VIDEO_GC08A3)      += gc08a3.o
```

```
diff --git a/arch/arm64/configs/rockchip_defconfig
b/arch/arm64/configs/rockchip_defconfig
index 95ad81b12a69..c66f67ecd0c2 100644
--- a/arch/arm64/configs/rockchip_defconfig
+++ b/arch/arm64/configs/rockchip_defconfig
@@ -567,6 +567,8 @@ CONFIG_VIDEO_ROCKCHIP_ISPP=y
 CONFIG_VIDEO_ROCKCHIP_HDMIRX=y
 CONFIG_VIDEO_LT6911UXC=y
 CONFIG_VIDEO_LT7911D=y
+CONFIG_VIDEO_RK628_CSI=y
+CONFIG_VIDEO_RK628_BT1120=y
 CONFIG_VIDEO_TC35874X=y
 CONFIG_VIDEO_GC0312=y
 CONFIG_VIDEO_GC2145=y
```

Media 的驱动默认都会编译，只需要通过配置 dts 选择运行对应的驱动。

4. SOC 与 RK628 I2C 通信

SOC 使用 I2C 与 RK628 进行通信，RK628 典型的 7bit I2C 地址为 0x50，在同一个 I2C 总线下使用多片 RK628 时，可通过 RK628 的 GPIO 改变 I2C 地址，参考如下：

The i2c address consists of 7 bits, where the upper four bits are the identifier of the i2c device and the value is set to 4b'1010, the lower three bits are the device address. In order to meet the application of different scenarios, the I2C slave device address can be programmed through GOIO, the mapping of address to GPIO show as Table 5-1, the typical slave address is 7'b1010000.

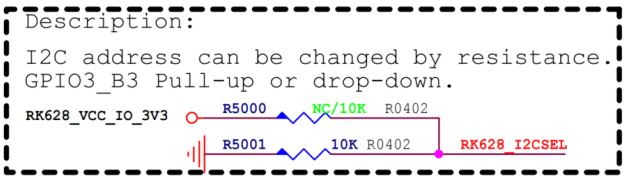
Table 5-2 Mapping of i2c slave address to GPIO

Addr bit	Pad Name	GPIO Setting
cfg_slvadr[2]	IO_GPIO0a1	GPIO0A_OE[1]=1'b1
cfg_slvadr[1]	IO_GPIO0a0	GPIO0A_OE[0]=1'b1
cfg_slvadr[0]	IO_GPIO3b3	GPIO3B_OE[2]=1'b1

以从机地址 0x50 和 0x51 为例：

I2C address configuration:

I2C SEL	I2C ADDR
0	7'b1010000(0x50)
1	7'b1010001(0x51)



GPIO0a0 和 GPIO0a1 保持低电平，GPIO3b3 拉高，则从机地址为 0x51 (7'b1010001)，GPIO3b3 拉低，则从机地址为 0x50 (7'b1010000)。

5. Misc

5.1 MISC 驱动介绍

RK628 Misc 驱动为 RK628 Display 通路的驱动代码，使用 RK628 进行显示协议转换，支持以 RGB、BT1120 或 HDMI 作为输入，输出 LVDS、GVI、DSI、RGB、BT1120 或 HDMI。

5.1.1 驱动移植说明

参考《驱动移植说明》-《Display 通路》章节

5.1.2 u-boot 驱动目录结构

```
drivers/video/drm/rk628/
├── Makefile
├── panel.c
├── panel.h
├── rk628.c
├── rk628_combtxphy.c
├── rk628_combtxphy.h
├── rk628_cru.c
├── rk628_cru.h
├── rk628_dsi.c
├── rk628_dsi.h
├── rk628_gvi.c
├── rk628_gvi.h
├── rk628.h
├── rk628_hdmirx.c
├── rk628_hdmirx.h
├── rk628_hdmitx.h
├── rk628_lvds.c
├── rk628_lvds.h
├── rk628_post_process.c
├── rk628_post_process.h
├── rk628_rgb.c
└── rk628_rgb.h
```

5.1.3 kernel 驱动目录结构

```
drivers/misc/rk628/
├── Kconfig
├── Makefile
├── panel.c
├── panel.h
├── rk628.c
├── rk628_combrxphy.c
├── rk628_combrxphy.h
├── rk628_combtxphy.c
├── rk628_combtxphy.h
├── rk628_config.c
├── rk628_config.h
├── rk628_cru.c
```

```
├── rk628_cru.h
├── rk628_csi.c
├── rk628_csi.h
├── rk628_dsi.c
├── rk628_dsi.h
├── rk628_efuse.c
├── rk628_efuse.h
├── rk628_gpio.h
├── rk628_grf.h
├── rk628_gvi.c
├── rk628_gvi.h
├── rk628.h
├── rk628_hdmirx.c
├── rk628_hdmirx.h
├── rk628_hdmitx.c
├── rk628_hdmitx.h
├── rk628_lvds.c
├── rk628_lvds.h
├── rk628_pinctrl.c
├── rk628_pinctrl.h
├── rk628_post_process.c
├── rk628_post_process.h
├── rk628_rgb.c
├── rk628_rgb.h
```

5.1.4 驱动 DTS 配置总览

以 RGB -> DSI 通路为例

```
&i2c2 {
    clock-frequency = <400000>;
    status = "okay";

    i2c2_rk628: rk628@50 {
        compatible = "rockchip,rk628";
        reg = <0x50>;
        interrupt-parent = <&gpio0>;
        interrupts = <20 IRQ_TYPE_LEVEL_HIGH>;
        enable-gpios = <&gpio0 RK_PC1 GPIO_ACTIVE_HIGH>;
        reset-gpios = <&gpio2 RK_PA2 GPIO_ACTIVE_LOW>;
        pinctrl-names = "default";
        pinctrl-0 = <&rk628_reset>;
        status = "okay";
    };
};

&pinctrl {
    rk628 {
        rk628_reset: rk628-reset {
            rockchip,pins = <2 RK_PA2 RK_FUNC_GPIO &pcfg_pull_none>;
        };
    };
};

&i2c2_rk628 {
```

```
    /* soc_24M optional */
    pinctrl-names = "default";
    pinctrl-0 = <&rk628_reset &refclk_pins>;
    assigned-clocks = <&pmucru CLK_WIFI>;
    assigned-clock-rates = <24000000>;
    clocks = <&pmucru CLK_WIFI>;
    clock-names = "soc_24M";

    panel-backlight = <&backlight>;
    panel-power-supply = <&vcc3v3_lcd0_n>;
    panel-enable-gpios = <&gpio2 RK_PC6 GPIO_ACTIVE_HIGH>;

    panel-reset-delay-ms = <10>;
    panel-enable-delay-ms = <10>;
    panel-prepare-delay-ms = <60>;
    panel-unprepare-delay-ms = <10>;
    panel-disable-delay-ms = <60>;
    panel-init-delay-ms = <60>;

    rk628-rgb-in;
    rk628-dsi-out {
        dsi,eotp;
        dsi,video-mode;
        dsi,format = "rgb888";
        dsi,lanes  = <4>;
        status = "okay";

        rk628-panel {
            panel-init-sequence = [
                05 78 01 11
                05 78 01 29
            ];

            panel-exit-sequence = [
                05 00 01 28
                05 00 01 10
            ];
        };
    };

    display-timings {
        src-timing {
            clock-frequency = <148500000>;
            hactive = <1920>;
            vactive = <1080>;
            hfront-porch = <88>;
            hsync-len = <44>;
            hback-porch = <148>;
            vfront-porch = <4>;
            vsync-len = <5>;
            vback-porch = <36>;
            hsync-active = <1>;
            vsync-active = <1>;
            de-active = <0>;
            pixelclk-active = <0>;
        };

        dst-timing {
```

```
            clock-frequency = <148500000>;
            hactive = <1920>;
            vactive = <1080>;
            hfront-porch = <88>;
            hsync-len = <44>;
            hback-porch = <148>;
            vfront-porch = <4>;
            vsync-len = <5>;
            vback-porch = <36>;
            hsync-active = <1>;
            vsync-active = <1>;
            de-active = <0>;
            pixelclk-active = <0>;
        };
    };
};
```

5.2 驱动核心配置

5.2.1 RK628 节点配置

SOC 使用 I2C 与 RK628 进行通信，以 RK3568 为例，在 DTS 的 I2C 节点（i2c2）下添加 RK628 节点（i2c2_rk628: rk628@50）：

```
&i2c2 {
    clock-frequency = <400000>;
    status = "okay";

    i2c2_rk628: rk628@50 {
        compatible = "rockchip,rk628";
        reg = <0x50>;
        enable-gpios = <&gpio0 RK_PC1 GPIO_ACTIVE_HIGH>;
        reset-gpios = <&gpio2 RK_PA2 GPIO_ACTIVE_LOW>;
        pinctrl-names = "default";
        pinctrl-0 = <&rk628_reset>;
        status = "okay";
    };
};

&pinctrl {
    rk628 {
        rk628_reset: rk628-reset {
            rockchip,pins = <2 RK_PA2 RK_FUNC_GPIO &pcfg_pull_none>;
        };
    };
};
```

RK628 节点属性说明

Property	Description	Option Value
compatible	compatible	RK628 Misc 驱动的 compatible 需设置为 "rockchip,rk628"
reg	RK628 I2C 从机地址	0x50/0x51，根据硬件进行配置
enable-gpios	RK628 使能 GPIO 引用	根据硬件进行配置
reset-gpios	RK628 复位 GPIO 引用	根据硬件进行配置
pinctrl-names	设置 RK628 reset 引脚的复用功能为 GPIO 功能	根据硬件进行配置
pinctrl-0		

5.2.2 24MHz 工作时钟配置

RK628 的 24MHz 工作时钟可以来自硬件上外接的 24M 晶振，也可以取自 SOC 的 24MHz CLK 引脚，使用外接 24M 晶振作为工作时钟可忽略以下配置。

需要注意的是，在 **RK628D HDMI / GVI OUT** 相应的通路场景以及 **RK628F/H RGB ->HDMI** 通路场景下，需要 SOC 输出的 24MHz CLK 引脚作为 RK628 的工作时钟实现时钟同源，接外接的 24M 晶振作为工作时钟可能会导致该场景显示异常。（RK628F/H 在同源问题上做了优化，以 HDMI RX 检测到的 TMDS CLK 作为 CRU PLL 的基准时钟，所以在以 HDMI IN 的通路场景下不需要配置来自 SOC 的 24MHz CLK 作为基准时钟）

以 RK3568 为例（更多 RK 平台配置参考《RK628 24M 晶振来自其他 SOC 的配置方式》章节），RK3568 能输出 24MHz 的引脚比较多，例如 REF_CLKOUT（clk_wifi / gpio0_a0）、CAM_CLKOUT1（clk_cam1_out / gpio4_b0）、ETH_REFCLK_25M_M0（clk_mac1_out / gpio3_b0），现以第一个为例介绍。

1. 在 DTS 的 RK628 节点中引用以下配置

```
&i2c2_rk628 {
    /* soc_24M optional */
    pinctrl-names = "default";
    pinctrl-0 = <&rk628_reset &refclk_pins>;
    assigned-clocks = <&pmucru CLK_WIFI>;
    assigned-clock-rates = <24000000>;
    clocks = <&pmucru CLK_WIFI>;
    clock-names = "soc_24M";
    ......
};
```

2. 以下配置在 rk3568-pinctrl.dtsi 中自带

```
&pinctrl {
    ......
    refclk {
        /omit-if-no-ref/
        refclk_pins: refclk-pins {
            rockchip,pins =
                /* refclk_ou */
                <0 RK_PA0 1 &pcfg_pull_none>;
        };
    };
    ......
};
```

24MHz CLK 属性说明

pinctrl-names	设置 refclk 引脚的复用功能	根据硬件进行配置 （因为是复写上述 RK628 节点属性，所以需要补充上 rk628_reset 引用）
pinctrl-0		
assigned-clocks	为 RK628 分配的时钟引用	根据硬件进行配置
assigned-clock-rates	为 RK628 分配的时钟速率（Hz）	<24000000>
clocks	引用时钟的 handle 列表	根据硬件进行配置
clock-names	为该时钟设置名称	"soc_24M"

5.2.3 输入输出模块选择

在 DTS 的 RK628 节点中，根据输入输出模块需求，配置对应的模块属性（bool 类型）或节点，模块节点中配置模块相应的功能参数：

```
&i2c2_rk628 {
    ......
    rk628-rgb-in;
    rk628-dsi-out {
        ......
    };
    ......
};
```

以 RGB -> DSI 通路为例，设置 “rk628-rgb-in;” 属性配置 RGB 模块作为输入，设置 “rk628-dsi-out” 节点配置 DSI 模块作为输出。目前 Misc 输入模块有 RGB IN、BT1120 IN 或 HDMI IN，输出模块有 LVDS、GVI、DSI、或 HDMI。各个输入输出模块配置详见《输入模块配置》及《输出模块配置》章节。

5.2.4 Panel 端配置

在 DTS 的 RK628 节点中，根据屏 Spec 配置屏相关的属性，包括背光、电源、复位引脚、屏上电时序、显示时序等（HDMITX 通路无需配置）：

```
&i2c2_rk628 {
    ......
```

```
    panel-backlight = <&backlight>;
    panel-power-supply = <&vcc3v3_lcd0_n>;
    panel-enable-gpios = <&gpio2 RK_PC6 GPIO_ACTIVE_HIGH>;

    panel-reset-delay-ms = <10>;
    panel-enable-delay-ms = <10>;
    panel-prepare-delay-ms = <60>;
    panel-unprepare-delay-ms = <10>;
    panel-disable-delay-ms = <60>;
    panel-init-delay-ms = <60>;
    ......
    display-timings {
        src-timing {
            clock-frequency = <148500000>;
            hactive = <1920>;
            vactive = <1080>;
            hfront-porch = <88>;
            hsync-len = <44>;
            hback-porch = <148>;
            vfront-porch = <4>;
            vsync-len = <5>;
            vback-porch = <36>;
            hsync-active = <1>;
            vsync-active = <1>;
            de-active = <0>;
            pixelclk-active = <0>;
        };

        dst-timing {
            clock-frequency = <148500000>;
            hactive = <1920>;
            vactive = <1080>;
            hfront-porch = <88>;
            hsync-len = <44>;
            hback-porch = <148>;
            vfront-porch = <4>;
            vsync-len = <5>;
            vback-porch = <36>;
            hsync-active = <1>;
            vsync-active = <1>;
            de-active = <0>;
            pixelclk-active = <0>;
        };
    };
    ......
};
```

屏相关属性说明

Property	Description	Option Value
panel-enable-gpios	屏使能 GPIO 引用 [option]	根据硬件进行配置
panel-reset-gpios	屏复位 GPIO 引用 [option]	根据硬件进行配置
panel-backlight	屏背光 [option]	根据硬件进行配置
panel-reset-delay-ms	屏上电时序	参考屏规格书
panel-enable-delay-ms		
panel-prepare-delay-ms		
panel-unprepare-delay-ms		
panel-disable-delay-ms		
panel-init-delay-ms		
src-timing	SOC 传输到 RK628 的显示时序节点	与 dst-timing 按行列分别缩放
dst-timing	屏端显示时序节点	参考屏规格书
clock-frequency	显示时序	参考屏规格书与实际 scaler
hactive		
vactive		
hfront-porch		
hback-porch		
hsync-len		
vfront-porch		
vback-porch		
vsync-len		
hsync-active		
vsync-active		
de-active		
pixelclk-active		

如果 RK628 输入端与输出端的分辨率采用缩放方式，则将输入端的 timing 配置到 src-timing，将输出端 timing 配置到 dst-timing。
缩放前后 timing 需要保证显示部分与消隐部分缩放比例一致：

```
src: hactive / dst: hactive = src: hblanking / dst: hblanking
src: vactive / dst: vactive = src: vblanking / dst: vblanking
hblanking = hfront-porch + hback-porch + hsync-len
vblanking = vfront-porch + vback-porch + vsync-len
```

且缩放前后帧率需要保持一致

```
src: clock-frequency / (htotal * vtotal) = dst: clock-frequency / (htotal *
vtotal)
htotal = hactive + hfront-porch + hback-porch + hsync-len
vtotal = vactive + vfront-porch + vback-porch + vsync-len
```

如果输入端与输出端分辨率一致，则 src-timing 和 dst-timing 配置相同的目标 timing。

如果用到两个同分辨率的 DSI 单屏或两个同分辨率的 LVDS 单屏，需要对 src-timing 和 dst-timing 中的 clock-frequency、hactive、hfront-porch、hback-porch 和 hsync-len 在原有单屏配置基础上乘以 2。

5.2.5 不同显示组合通路应用

参考如下 DTS，具体通路介绍详见《目前 DTS 支持的几种组合》章节。

```
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-hdmi2bt1120-ddr4-v10.dts
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-hdmi2dsi-ddr4-v10.dts
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-hdmi2dsi-dual-ddr4-v10.dts
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-hdmi2gvi-ddr4-v10.dts
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-hdmi2lvds-ddr4-v10.dts
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-hdmi2lvds-dual-ddr4-v10.dts
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2dsi-ddr4-v10.dts
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2gvi-ddr4-v10.dts
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2hdmi-ddr4-v10.dts
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2lvds-ddr4-v10.dts
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2lvds-dual-ddr4-v10.dts
```

5.3 输入模块配置

5.3.1 RGB 输入

5.3.1.1 RGB 输入配置

1. 在 DTS 的 RK628 节点中添加 “rk628-rgb-in;” bool 属性即可：

```
&i2c2_rk628 {
    ......
    rk628-rgb-in;
    ......
};
```

需要注意的是，RGB IN / RGB OUT / BT1120 IN / BT1120 OUT 功能共用引脚，所以这四种功能不能同时存在。

2. SOC 配置 RGB 输出，以 RK3568 为例，RGB 输出走 DRM 框架，将 RK628 虚拟作一块屏幕：

```
/ {
    panel@0 {
        compatible = "simple-panel";

        disp_timings3: display-timings {
```

```
            native-mode = <&rgb2dsi_timing>;
            rgb2dsi_timing: timing0 {
                clock-frequency = <148500000>;
                hactive = <1920>;
                vactive = <1080>;
                hfront-porch = <88>;
                hsync-len = <44>;
                hback-porch = <148>;
                vfront-porch = <4>;
                vsync-len = <5>;
                vback-porch = <36>;
                hsync-active = <1>;
                vsync-active = <1>;
                de-active = <0>;
                pixelclk-active = <0>;
            };
        };

        port {
            panel_in_rgb: endpoint {
                remote-endpoint = <&rgb_out_panel>;
            };
        };
    };
};

&route_rgb {
    status = "okay";

    /* for u-boot logo display */
    bridge = <&i2c2_rk628>;
};

&rgb_in_vp2 {
    status = "okay";
};

&rgb {
    status = "okay";

    ports {
        port@1 {
            reg = <1>;

            rgb_out_panel: endpoint {
                remote-endpoint = <&panel_in_rgb>;
            };
        };
    };c
};
```

“panel@0/display-timings/timing0” 节点的时序需要与 “&i2c2_rk628/display-timings/src-timing” 节点的时序参数配置一致。

5.3.2 BT1120 输入

5.3.2.1 BT1120 输入配置

参考 “RGB 输入配置”，将 DTS 的 RK628 节点中的 “rk628-rgb-in;” bool 属性修改 “rk628-bt1120-in;” 为即可：

```
&i2c2_rk628 {
    ......
    rk628-bt1120-in;
    // bt1120-dual-edge;
    // bt1120-yc-swap;
    // bt1120-uv-swap;
    ......
};
```

若需要配置 BT1120 为双边沿，则在 RK628 节点中添加 “bt1120-dual-edge;” bool 属性。

“bt1120-yc-swap” 属性及 “bt1120-uv-swap” 属性修改 RK628 解析 BT1120 信号在线缆上顺序，若画面显示图像正常但颜色异常时，可以尝试这两个属性。

需要注意的是，RGB IN / RGB OUT / BT1120 IN / BT1120 OUT 功能共用引脚，所以这四种功能不能同时存在。

5.3.3 HDMI 输入

5.3.3.1 HDMI 输入配置

目前 HDMI 输入有两种形态，一种是与 SOC 直连，一种是通过 HDMI 线缆连接。

5.3.3.2 与 SOC 直连模式

在 DTS 的 RK628 节点中配置：

```
&i2c2_rk628 {
    ......
    rk628-hdmi-in;
    // src-mode-4k-yuv420;
    ......
};
```

与 SOC 直连的模式，需要 HDMI 源默认输出，所以 RK628 不需要通过 io 引脚检测 HDMI 源端热插拔，且不支持分辨率切换。在该模式下，驱动采用轮询方式检测 HDMI 信号的输入。

若 HDMI 源输出 4K YUV420 （即 force-bus-format 属性设置为 MEDIA_BUS_FMT_UYYVYY8_0_5X24，force_timing 设置为 4K 分辨率），则需要在 RK628 节点下配置 “src-mode-4k-yuv420” bool 属性，因为 HDMIRX 无法提前判断 HDMI 源是否是 YUV420 输出。

SOC 指定输出特定分辨率的 HDMI 源：

SOC 配置 HDMI 输出，以 RK3568 为例，将 HDMI 配置为指定输出，可以指定输出源的 timing 及 bus-format：

```
#include <dt-bindings/display/media-bus-format.h>

&hdmi {
    status = "okay";
    force-bus-format = <MEDIA_BUS_FMT_RGB888_1X24>;
    force-output;
    force_timing{
        clock-frequency = <594000000>;
        hactive = <3840>;
        vactive = <2160>;
        hback-porch = <296>;
        hfront-porch = <176>;
        vback-porch = <72>;
        vfront-porch = <8>;
        hsync-len = <88>;
        vsync-len = <10>;
        hsync-active = <1>;
        vsync-active = <1>;
        de-active = <0>;
        pixelclk-active = <0>;
    };
};

&hdmi_in_vp0 {
    status = "okay";
};

&hdmi_in_vp1 {
    status = "disabled";
};

&route_hdmi {
    status = "okay";

    /* for u-boot logo display */
    bridge = <&i2c2_rk628>;
    force-bus-format = <MEDIA_BUS_FMT_RGB888_1X24>;
    force-output;
    force_timing{
        clock-frequency = <594000000>;
        hactive = <3840>;
        vactive = <2160>;
        hback-porch = <296>;
        hfront-porch = <176>;
        vback-porch = <72>;
        vfront-porch = <8>;
        hsync-len = <88>;
        vsync-len = <10>;
        hsync-active = <1>;
        vsync-active = <1>;
        de-active = <0>;
        pixelclk-active = <0>;
    };
};
```

使能 “&route_hdmi” 节点是为了让 HDMIRX 通路可以显示开机的 kernel logo，“&route_hdmi” 节点下的 “force-output”、“force-bus-format”、 “force_timing” 需要与 “hdmi” 节点保持一致。

配置 force-bus-format 属性时，需要在该 dts 里添加头文件 <dt-bindings/display/media-bus-format.h>，该头文件包含各种显示数据格式。

SOC 强制输出 HDMI 补丁

在 RK SOC 平台上，配置 HDMI 强制输出指定分辨率需要打上如下相应的补丁（可以对照补丁内容与源码内容判断该补丁是否添加过）：

补丁列表	**commit message**	补丁说明
0190-drm-bridge-dw-hdmi-qp-Support-hdmi-force-output-kernel-5-10.patch	drm/bridge: dw-hdmi-qp: Support hdmi force output	在 rk 平台的**kernel5.10**版本 hdmi tx 驱动没有该补丁则需要导入
0190-drm-bridge-synopsys-Support-hdmi-force-output-kernel-5-10.patch	drm/bridge: synopsys: Support hdmi force output	在 rk 平台的**kernel5.10**版本 hdmi tx 驱动没有该补丁则需要导入
0190-drm-bridge-synopsys-Support-hdmi-force-output-kernel-4-19.patch	drm/bridge: synopsys: Support hdmi force output	在 rk 平台的**kernel4.19**版本 hdmi tx 驱动没有该补丁则需要导入
0190-drm-bridge-synopsys-Support-hdmi-force-output.patch	drm/bridge: synopsys: Support hdmi force output	在 rk 平台的**kernel4.4**版本 hdmi tx 驱动没有该补丁则需要导入
0189-drm-bridge-synopsys-dw-hdmi-Support-force-logo-displ.patch	drm/bridge: synopsys: dw-hdmi: Support force logo display	在 rk 平台的**kernel4.4**版本 hdmi tx 驱动没有该补丁则需要导入

5.3.3.3 HDMI 线缆连接模式

在 DTS 的 RK628 节点中配置：

```
&i2c2_rk628 {
    ......
    interrupt-parent = <&gpio0>;
    interrupts = <20 IRQ_TYPE_LEVEL_HIGH>;
    plugin-det-gpios = <&gpio2 RK_PA4 GPIO_ACTIVE_LOW>;
    // hpd-output-inverted;
    rk628-hdmi-in;
    ......
};
```

HDMI 线缆连接 RK628 需要通过 IO 检测 HDMI 源热插拔，同时也有分辨率 / 颜色格式切换的需求，所以该模式下需要在 RK628 节点下配置 “plugin-det-gpios” 属性，且需要配置 “interrupt-parent”、“interrupts” 属性。此处的热拔插脚是 RK628 对 HDMI 源的热拔插检测，不是 HDMI 端对 RK628 的热拔插检测，通常两端都需要检测对方的热拔插情况。

“hpd-output-inverted” bool 属性：上述热插拔检测 IO 取反配置，若 HPD 输出电平在电路上做了取反，则需要使能此配置项，通常也可借助 GPIO 的GPIO_ACTIVE_LOW / GPIO_ACTIVE_HIGH 配置来进行软件取反。

HDMI 线缆连接模式下如果需要修改 HDMIRX 的分辨率，只需要将《Panel 端配置》章节中的 src-timing 修改为指定分辨率即可。

5.3.3.4 音频配置

做为 HDMI 输入的时候，RK628 是输出音频的，I2S 作为 master 把音频数据从 I2S 输出，dts 声卡配置：

```
rk628_dc: rk628-dc {
    compatible = "rockchip,dummy-codec";
    #sound-dai-cells = <0>;
};

&i2s0 {
    rockchip,capture-only;
    status = "okay";
};

/ {
    hdmiin-sound {
        compatible = "simple-audio-card";
        simple-audio-card,format = "i2s";
        simple-audio-card,name = "rockchip,hdmiin";
        simple-audio-card,bitclock-master = <&dailink0_master>;
        simple-audio-card,frame-master = <&dailink0_master>;
        status = "okay";
        simple-audio-card,cpu {
            sound-dai = <&i2s0>;
        };
        dailink0_master: simple-audio-card,codec {
            sound-dai = <&rk628_dc>;
        };
    };
};
```

SOC I2S 做从，在没有时钟信号的时候，驱动会一直等待，在 Android 系统中，等待太长容易出现 crash，可以添加下面补丁修改超时：

```
diff --git a/sound/soc/rockchip/rockchip_i2s.c
b/sound/soc/rockchip/rockchip_i2s.c
index 9bc29fdd13c5..230c350c6765
--- a/sound/soc/rockchip/rockchip_i2s.c
+++ b/sound/soc/rockchip/rockchip_i2s.c
@@ -468,6 +468,9 @@ static int rockchip_i2s_trigger(struct snd_pcm_substream
*substream,
                ret = -EINVAL;
                break;
        }
+       if(substream->stream == SNDRV_PCM_STREAM_CAPTURE) {
+               substream->wait_time = msecs_to_jiffies(100);
+       }

        return ret;
 }
diff --git a/sound/soc/rockchip/rockchip_i2s_tdm.c
b/sound/soc/rockchip/rockchip_i2s_tdm.c
index b0c4ce01e2be..0602df42c309
--- a/sound/soc/rockchip/rockchip_i2s_tdm.c
+++ b/sound/soc/rockchip/rockchip_i2s_tdm.c
```

```
@@ -1776,6 +1776,9 @@ static int rockchip_i2s_tdm_trigger(struct
snd_pcm_substream *substream,
                ret = -EINVAL;
                break;
        }
+       if(substream->stream == SNDRV_PCM_STREAM_CAPTURE) {
+               substream->wait_time = msecs_to_jiffies(100);
+       }

        return ret;
 }
```

5.4 输出模块配置

5.4.1 RGB 输出

5.4.1.1 RGB 输出配置

在 DTS 的 RK628 节点中添加 “rk628-rgb-out;” bool 属性即可：

```
&i2c2_rk628 {
    ......
    rk628-rgb-out;
    ......
};
```

需要注意的是，RGB IN / RGB OUT / BT1120 IN / BT1120 OUT 功能共用引脚，所以这四种功能不能同时存在。

5.4.2 BT1120 输出

5.4.2.1 BT1120 输出配置

在 DTS 的 RK628 节点中添加 “rk628-bt1120-out;” bool 属性即可：

```
&i2c2_rk628 {
    ......
    rk628-bt1120-out;
    // bt1120-dual-edge;
    ......
};
```

若需要配置 BT1120 为双边沿，则在 RK628 节点中添加 “bt1120-dual-edge;” bool 属性。
需要注意的是，RGB IN / RGB OUT / BT1120 IN / BT1120 OUT 功能共用引脚，所以这四种功能不能同时存在。

5.4.3 DSI 输出

5.4.3.1 DSI 输出配置

在 DTS 的 RK628 节点中添加 “rk628-dsi-out” 节点：

```
&i2c2_rk628 {
    ......
    rk628-dsi-out {
        // rockchip,lane-mbps = <1100>;
        // rockchip,dual-channel;
        dsi,eotp;
        dsi,video-mode;
        // dsi,clk-non-continuous;
        dsi,format = "rgb888";
        dsi,lanes  = <4>;
        status = "okay";

        rk628-panel {
            panel-init-sequence = [
                ......
                05 78 01 11
                05 78 01 29
            ];

            panel-exit-sequence = [
                05 00 01 28
                05 00 01 10
            ];
        };
    };
    ......
};
```

“rk628-dsi-out” 节点配置属性说明

Property	Description	Option Value
rockchip,lane-mbps	指定 MIPI 数据带宽 [option]	
rockchip,dual-channel	单双 MIPI [option]	默认为单 MIPI，添加该属性配置为双 MIPI
dsi,eotp	EOT PACKET [option]	
dsi,video-mode	Video 模式 / Command 模式 [option]	默认为 Command 模式，添加该属性配置为 Video 模式
dsi,clk-non-continuous	连续 / 非连续时钟 [option]	默认为连续时钟，添加该属性配置为非连续时钟
dsi,format	DSI 数据格式	"rgb888" / "rgb666" / "rgb666-packed" / "rgb565"
dsi,lanes	DSI lanes	双 MIPI 按单 MIPI lanes 配置
panel-init-sequence	屏 init 序列	第一列表示 data type，第二列表示 mdelays， 第三列表示发送每条命令的 payload_lenth， 后面几列都是表示每条命令的 payload
panel-exit-sequence	屏 exit 序列	同上

常见数据类型

data type	description	packet size
0x03	Generic Short WRITE, no parameters	short
0x13	Generic Short WRITE, 1 parameters	short
0x23	Generic Short WRITE, 2 parameters	short
0x29	Generic long WRITE,	long
0x05	DCS Short WRITE, no parameters	short
0x15	DCS Short WRITE, 1 parameters	short

5.4.4 LVDS 输出

5.4.4.1 LVDS 输出配置

在 DTS 的 RK628 节点中添加 “rk628-lvds-out” 节点：

```
&i2c2_rk628 {
    ......
    rk628-lvds {
        bus-format = <MEDIA_BUS_FMT_RGB888_1X7X4_SPWG>;
        link-type = "single_link";
        status = "okay";
    };
    ......
};
```

“rk628-lvds-out” 节点配置属性说明

Property	Description	Option Value
bus-format	总线格式	<MEDIA_BUS_FMT_RGB666_1X7X3_SPWG> / <MEDIA_BUS_FMT_RGB888_1X7X4_SPWG> / <MEDIA_BUS_FMT_RGB888_1X7X4_JEIDA>
link-type	通道类型	"single_link" （单通道） "dual_link_odd_even_pixels" （双通道，左右通道为奇偶通道） "dual_link_even_odd_pixels" （双通道，左右通道为偶奇通道） "dual_link_left_right_pixels" （双通道，左右通道为左右屏） "dual_link_right_left_pixels" （双通道，左右通道为右左屏）

5.4.5 GVI 输出

5.4.5.1 GVI 输出配置

在 DTS 的 RK628 节点中添加 “rk628-gvi-out” 节点：

```
&i2c2_rk628 {
    ......
    rk628-gvi-out {
        bus-format = "rgb888";
        gvi,lanes = <8>;
        // rockchip,division-mode;
        // rockchip, gvi-frm-rst;
        status = "okay";
    };
    ......
};
```

“rk628-gvi-out” 节点配置属性说明

Property	Description	Option Value
bus-format	总线格式	"rgb666" / "rgb888" / "rgb101010" / "yuyv8" / "yuyv10"
gvi,lanes	GVI lanes	1、2、4、8
rockchip,division-mode	one section / two section [option]	默认为 one section 模式，添加该属性设置为 two section 模式
rockchip, gvi-frm-rst	enable gvi rst when frame start [option]	默认 disabled，添加该属性设置为 enable

5.4.5.2 输出信号调节

1. 可以通过直接写寄存器来调节信号做测试（重启后配置不保存；需要固定配置，参考下方补丁修改）

```
# 1. 电压缩放 N=0(75%), 1/2(100%), 3(125%)
# echo 0x90004 $val > /d/rk628/2-0050/registers/combtxphy
# $val:
#     tx0[Bit1:0], tx1[Bit3:2], tx2[Bit5:4], tx3[Bit7:6], tx4[Bit9:8]
#     tx5[Bit17:16], tx6[Bit19:18], tx7[Bit21:20], tx8[Bit23:22], tx9[Bit25:24]
# 以 1(100%) 为例:
echo 0x90004 0x01550155 > /d/rk628/2-0050/registers/combtxphy

# 2. 电压调节 N=0~9
# echo 0x90008 $val > /d/rk628/2-0050/registers/combtxphy
# $val:
#     tx0[Bit2:0], tx1[Bit5:3], tx2[Bit8:6], tx3[Bit11:9], tx4[Bit14:12]
#     tx5[Bit18:16], tx6[Bit21:19], tx7[Bit24:22], tx8[Bit27:25], tx9[Bit30:28]
# 以 3 档为例:
echo 0x90008 0x36db36db > /d/rk628/2-0050/registers/combtxphy

# 3. 驱动强度调节 N=0~9
# echo 0x9000c $val > /d/rk628/2-0050/registers/combtxphy
# $val: tx0[Bit3:0], tx1[Bit7:4], tx2[Bit11:8], tx3[Bit15:12], tx4[Bit19:16]
# echo 0x90010 $val > /d/rk628/2-0050/registers/combtxphy
# $val: tx5[Bit3:0], tx6[Bit7:4], tx7[Bit11:8], tx8[Bit15:12], tx9[Bit19:16]
# 以 5 档为例:
echo 0x9000c 0x55555 > /d/rk628/2-0050/registers/combtxphy
echo 0x90010 0x55555 > /d/rk628/2-0050/registers/combtxphy

# 4. 展频
# 对 CPLL 全局展频来实现 GVI 展频
# echo 0xc000c $((val | 0xffff0000)) > /d/rk628/2-0050/registers/cru
# $val 参考下表
# 例:
echo 0xc000c 0xffff1f80 > /d/rk628/2-0050/registers/cru
```

展频值设置参考（表格中为实验值，以实际测量为准）：

$val	强度 / dBm	$val	强度 / dBm
0x1f80	-76.46	0x0ff0	-68.76
0x1fc0	-76.75	0x0fc0	-70.94
0x1ff0	-76.76	0x0f80	-68.92
0x14c0	-72.94	0x0f40	-74.63
0x18c0	-74.07	0x1f80	-73.58
		0x1c80	-74.87
		0x1880	-73.7
		0x1480	-69.4
		0x0f80	-68.92
		0x0880	-66.64
		0x0480	-68.01

2. 输出信号调节补丁（配置值参考上述说明）

```
diff --git a/drivers/misc/rk628/rk628_combtxphy.c
b/drivers/misc/rk628/rk628_combtxphy.c
index 3387d8bd73b9..70d7a5422a35 100644
--- a/drivers/misc/rk628/rk628_combtxphy.c
+++ b/drivers/misc/rk628/rk628_combtxphy.c
@@ -139,6 +139,17 @@ static void rk628_combtxphy_gvi_power_on(struct rk628
*rk628)
        usleep_range(100, 200);
        rk628_i2c_update_bits(rk628, COMBTXPHY_CON0,
                              SW_TX_IDLE_MASK, 0);
+
+       // amplitude scale
+       rk628_i2c_write(rk628, COMBTXPHY_CON1, 0x01550155);
+       // amplitude level
+       rk628_i2c_write(rk628, COMBTXPHY_CON2, 0x36db36db);
+       // amplitude level
+       rk628_i2c_write(rk628, COMBTXPHY_CON3, 0x00055555);
+       rk628_i2c_write(rk628, COMBTXPHY_CON4, 0x00055555);
+
+       // ssmod
+       rk628_i2c_write(rk628, CRU_CPLL_CON3, 0xffff1f80);
 }

 void rk628_combtxphy_power_on(struct rk628 *rk628)
```

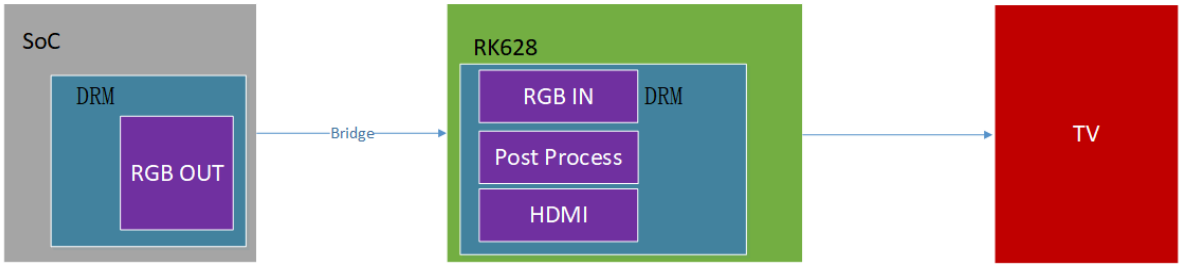
5.4.5.3 RK628F GVI 测试型号列表

TCON	型号	屏幕厂家	分辨率
IN8205A	M280DGJ-L30	群创光电	3840*2160
IN8205A	M315DJJ		
前期是 IN8908A (2021 年年底改为 IN8210)	S500DJ2-KS5		
前期是 IN8908A (2021 年年底改为 IN8210)	V500DJ2-KS5		
SW0894A	LC860EQY-FJA5	巡视	
	LC550EGE	LGDisplay	
Hi3231V530	HV550QUB-N8D	BOE	
SW0894A	MZ860004-07	ACT	
	86S13DCX		
TLi2461MC	ACT-MZ750004-07		
LP71517			
SW08006A	848.2		
TLi2380EP	ist-ITV-016	FunTV	
SW08008			
10301	L22650USWL1	Lenove	
10302	HV860QUB-E1D		
CSQ12-B0S	SG8561D03-1		
CSTU01-A0H	H750D4-BA080AA		
CSTU01-A0W	SL(LX)-23		
KV7636-VPP	ST7461D02-6		
KV7626			
AUO-12417			
AUO-12415	ICB-VN65	HiteVision	
NT71120MFG-001			
EK76603E	V650DK-KS5		
iLITEK-2326			

IN8205A		
MST6M60FV	TN2A1FV4LCZZ	
	HS-86AW-L09PA	
	HV650QUB-F70	
	CEJZ650L07Q1	华星光电
	MT5461D01-1	华星光电
	BOEI650WQ1	
	UV650QUB-N90	
	LC650EQQ-SMA5	

5.4.6 HDMI 输出

考虑到 HDMI 需要支持切不同分辨率的需求，当前 Misc 的 RK628 作为 DRM 框架的 Bridge 实现：



5.4.6.1 HDMI 输出配置

在 DTS 的 RK628 节点中添加 “rk628-hdmi-out” bool 属性，“&rgb” 节点与 RK628 节点通过 “remote-endpoint” 进行 binding：

```
&i2c2_rk628 {
    pinctrl-names = "default";
    pinctrl-0 = <&rk628_reset &refclk_pins>;
    assigned-clocks = <&pmucru CLK_WIFI>;
    assigned-clock-rates = <24000000>;
    clock-names = "soc_24M";
    clocks = <&pmucru CLK_WIFI>;

    rk628-rgb-in;
    rk628-hdmi-out;
    // mode-sync-pol = <0>;
    status = "okay";

    port {
        rgb_in_hdmi: endpoint {
            remote-endpoint = <&rgb_out_hdmi>;
        };
    };
};
```

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

&rgb_in_vp2 {
    status = "okay";
};

&rgb {
    status = "okay";

    ports {
        port@1 {
            reg = <1>;

            rgb_out_hdmi: endpoint {
                remote-endpoint = <&rgb_in_hdmi>;
            };
        };
    };
};
```

HDMITX 分辨率和极性随标准分辨率变化，所有没有配置屏相关参数，若为 RGB/BT1120 输入，则需要根据 RGB/BT1120 实际 hsync-active 和 vsync-active 配置 mode-sync-pol 属性，只有一个属性，所以 hsync-active 和 vsync-active 需要保持一致。若没有配置，则默认极性为 1（正极性）。若画面显示花屏时可以尝试添加该属性。

需要注意的是，HDMITX 通路需要实现 CLK 同源，否则可能会导致该通路显示异常，具体 CLK 同源配置详见《驱动核心配置》章节的《24MHz 工作时钟配置》小节。

5.4.6.2 音频配置

```
&i2s0 {
    rockchip,playback-only;
    status = "oaky";
};

/ {
    rk628_sound: rk628-sound {
        compatible = "simple-audio-card";
        simple-audio-card,format = "i2s";
        simple-audio-card,mclk-fs = <128>;
        simple-audio-card,name = "rockchip,hdmi-rk628";
        status = "oaky";
        simple-audio-card,cpu {
            sound-dai = <&i2s0>;
        };
        simple-audio-card,codec {
            sound-dai = <&i2c2_rk628>;
        };
    };
}
```

5.5 目前 DTS 支持的几种组合

5.5.1 RGB -> DSI 转换

参考 DTS：

```
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-ddr4-v10.dtsi
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2dsi-ddr4-v10.dts
```

以上 DTS 是基于 rk3568-evb.dtsi 增加 RK628 支持 RGB 转 MIPI DSI 的功能，具体差异请结合“输入模块介绍”及“输出模块介绍”章节自行对比代码。

SOC 配置输出指定 timing 的 RGB 参考 《输入模块配置》章节的《RGB 输入》小节。

5.5.2 RGB -> LVDS 转换

参考 DTS：

```
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-ddr4-v10.dtsi
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2lvds-ddr4-v10.dts
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2lvds-dual-ddr4-v10.dts
```

以上 DTS 是基于 rk3568-evb.dtsi 增加 RK628 支持 RGB 转 MIPI DSI 的功能，其中 rk3568-evb-rk628-rgb2lvds-ddr4-v10.dts 为输出单通道 LVDS，rk3568-evb-rk628-rgb2lvds-dual-ddr4-v10.dts 为输出双通道 LVDS，具体差异请结合“输入模块介绍”及“输出模块介绍”章节自行对比代码。

SOC 配置输出指定 timing 的 RGB 参考 《输入模块配置》章节的《RGB 输入》小节。

5.5.3 RGB -> GVI 转换

参考 DTS：

```
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-ddr4-v10.dtsi
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2gvi-ddr4-v10.dts
```

以上 DTS 是基于 rk3568-evb.dtsi 增加 RK628 支持 RGB 转 GVI 的功能，具体差异请结合“输入模块介绍”及“输出模块介绍”章节自行对比代码。

SOC 配置输出指定 timing 的 RGB 参考 《输入模块配置》章节的《RGB 输入》小节。

5.5.4 RGB -> HDMI 转换

参考 DTS：

```
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-ddr4-v10.dtsi
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2hdmi-ddr4-v10.dts
```

以上 DTS 是基于 rk3568-evb.dtsi 增加 RK628 支持 RGB 转 HDMI 的功能，具体差异请结合“输入模块介绍”及“输出模块介绍”章节自行对比代码。

因为 HDMITX 需要检测屏端支持的分辨率作为目标分辨率，所以在 DTS 中没有配置 SOC 输出 RGB 的 timing。

5.5.5 HDMI -> DSI 转换

参考 DTS：

```
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-ddr4-v10.dtsi
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-hdmi2dsi-ddr4-v10.dts
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-hdmi2dsi-dual-ddr4-v10.dts
```

以上 DTS 是基于 rk3568-evb.dtsi 增加 RK628 支持 HDMI 转 DSI 的功能，使用与 SOC 直连的模式，其中 rk3568-evb-rk628-hdmi2dsi-ddr4-v10.dts 为输出单 MIPI DSI，rk3568-evb-rk628-hdmi2dsi-dual-ddr4-v10.dts 为输出双 MIPI DSI，具体差异请结合“输入模块介绍”及“输出模块介绍”章节自行对比代码。

SOC 配置输出指定 timing 的 RGB 参考 《输入模块配置》章节的《HDMI 输入》小节。

5.5.6 HDMI -> LVDS 转换

参考 DTS：

```
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-ddr4-v10.dtsi
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-hdmi2lvds-ddr4-v10.dts
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-hdmi2lvds-dual-ddr4-v10.dts
```

以上 DTS 是基于 rk3568-evb.dtsi 增加 RK628 支持 HDMI 转 LVDS 的功能，使用与 SOC 直连的模式，其中 rk3568-evb-rk628-hdmi2lvds-ddr4-v10.dts 为输出单通道 LVDS，rk3568-evb-rk628-hdmi2lvds-dual-ddr4-v10.dts 为输出双通道 LVDS，具体差异请结合“输入模块介绍”及“输出模块介绍”章节自行对比代码。

SOC 配置输出指定 timing 的 RGB 参考 《输入模块配置》章节的《HDMI 输入》小节。

5.5.7 HDMI -> GVI 转换

参考 DTS：

```
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-ddr4-v10.dtsi
arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-hdmi2gvi-ddr4-v10.dts
```

以上 DTS 是基于 rk3568-evb.dtsi 增加 RK628 支持 HDMI 转 GVI 的功能，使用与 SOC 直连的模式，具体差异请结合“输入模块介绍”及“输出模块介绍”章节自行对比代码。

SOC 配置输出指定 timing 的 RGB 参考 《输入模块配置》章节的《HDMI 输入》小节。

5.6 基础调试命令

5.6.1 寄存器调试节点

/sys/kernel/debug/regmap 节点下

```
rk3568_t:/ # ls /sys/kernel/debug/regmap/
0-001c              2-0050-dsi1   2-0050-hdmirx     4-0050-gpio0
0-0020              2-0050-efuse  4-0050-adapter    4-0050-gpio1
0-0020-rk817-codec  2-0050-gpio0  4-0050-combrxphy  4-0050-gpio2
2-0050-adapter      2-0050-gpio1  4-0050-combtxphy  4-0050-gpio3
2-0050-combrxphy    2-0050-gpio2  4-0050-cru        4-0050-grf
2-0050-combtxphy    2-0050-gpio3  4-0050-csi        4-0050-gvi
2-0050-cru          2-0050-grf    4-0050-dsi0       4-0050-hdmi
2-0050-csi          2-0050-gvi    4-0050-dsi1       4-0050-hdmirx
2-0050-dsi0         2-0050-hdmi   4-0050-efuse      ......
```

或 /sys/kernel/debug/rk628/x-xxxx/registers 节点下

```
rk3568_t:/ # ls /sys/kernel/debug/rk628/2-0050/registers/
adapter  combrxphy  combtxphy  cru  csi  dsi0  dsi1  efuse  gpio0  gpio1  gpio2
gpio3  grf  gvi  hdmi  hdmirx
```

其中，2 表示 I2C 总线号，0050 表示 RK628 SLAVE 地址。

1. 读寄存器（以 GRF 寄存器为例）

/sys/kernel/debug/regmap 节点下

```
rk3568_t:/ # cat /sys/kernel/debug/regmap/2-0050-grf/registers
000: 06000088
004: ffffffff
008: 00000000
00c: 00000000
010: 00000001
014: 00000000
018: 00050000
01c: 002c0898
020: 00c00840
......

rk3568_t:/ # cat /sys/kernel/debug/regmap/2-0050-grf/registers | grep 200:
200: 20230321
```

或 /sys/kernel/debug/rk628/x-xxxx/registers 节点下

```
rk3568_t:/ # cat /sys/kernel/debug/rk628/2-0050/registers/grf
rk628_grf:

0x0000: 06000298 ffffffff 00000000 00000001
0x0010: 00000001 00000000 000404b6 002804ba
0x0020: 0046047e 000a079e 00140794 0046047e
0x0030: 00140794 00000004 00000000 00000000
0x0040: 04380004 80000000 00000000 00000000
```

```
0x0050: 80000000 00000000 10000000 00000000
0x0060: 000002ea 01f0ea00 00000000 01f00000
0x0070: 0000155c 00005602 00000000 00000000
0x0080: 00006000 00000000 00000000 00000000
......

rk3568_t:/ # cat /sys/kernel/debug/rk628/2-0050/registers/grf | grep 0200:
0x0200: 20230321 00000000 00000000 00000000
```

2. 写寄存器（以 GRF 寄存器为例）

/sys/kernel/debug/regmap 节点下

```
rk3568_t:/ # echo 0x010 0x02000200 > /sys/kernel/debug/regmap/2-0050-
grf/registers
```

或 /sys/kernel/debug/rk628/x-xxxx/registers 节点下

```
rk3568_t:/ # echo 0x010 0x02000200 > /sys/kernel/debug/rk628/2-0050/registers/grf
```

5.6.2 自测模式命令

当输出无法正常显示时，可以通过以下命令来判断问题出在 RK628 input 端还是 RK628 output 端（仅 RK628F/H 可以使用）：

```
# Enable horizontal color bar
rk3568_t:/ # echo 1 > /sys/kernel/debug/rk628/2-0050/scaler_color_bar

# Enable vertical color bar
rk3568_t:/ # echo 2 > /sys/kernel/debug/rk628/2-0050/scaler_color_bar

# Disable color bar
rk3568_t:/ # echo 0 > /sys/kernel/debug/rk628/2-0050/scaler_color_bar
```

若 scaler colorbar 正常显示，则主要排查主控输出、RK628 input、RK628 Process 的配置；若无法正常显示，则先排查 RK628 output 的配置，并通过下列各个接口的 colorbar 判断输出模块的控制器、对应的 phy、屏端这条链路是否正常工作。下列的模块 colorbar 正常工作后，scaler color 仍然无法显示，则继续排查 RK628 input 的配置。

以下命令测试输出模块的控制器、对应的 phy、屏端这条链路是否正常工作：

1. HDMITX color bar

```
# Enable normal color bar
rk3568_t:/ # echo 1 > /sys/kernel/debug/rk628/2-0050/hdmitx_color_bar

# Enable special color bar
rk3568_t:/ # echo 2 > /sys/kernel/debug/rk628/2-0050/hdmitx_color_bar

# Enable black color bar
rk3568_t:/ # echo 3 > /sys/kernel/debug/rk628/2-0050/hdmitx_color_bar

# Disable color bar
rk3568_t:/ # echo 0 > /sys/kernel/debug/rk628/2-0050/hdmitx_color_bar
```

2. DSI color bar

```
# Enable dsi color bar
rk3568_t:/ # echo 1 > /sys/kernel/debug/rk628/2-0050/dsi_color_bar

# Disable dsi color bar
rk3568_t:/ # echo 0 > /sys/kernel/debug/rk628/2-0050/dsi_color_bar
```

3. GVI color bar

```
# Enable color bar
rk3568_t:/ # echo 1 > /sys/kernel/debug/rk628/2-0050/gvi_color_bar

# Disable color bar
rk3568_t:/ # echo 0 > /sys/kernel/debug/rk628/2-0050/gvi_color_bar
```

5.6.3 RGB IN 调试命令（仅 RK628F/H 支持）

在 RK628F/H 的应用下，可以通过如下命令，查看 RGB IN 实际收到的分辨率，判断 RGB RX 是否正常

```
rk3568_t:/ # cat /sys/kernel/debug/rk628/2-0050/rgb_resolution
1080x1920 pclk:131967000
```

5.7 显示常见问题处理

5.7.1 没有生成 regmap 节点

如下 log，查看 RK628 相关是否存在 error 导致 RK628 无法正常完成 probe

```
rk3568_t:/ # dmesg | grep rk628
[    0.294331] rk628 2-0050: the driver version is 0.1.0 of RK628-F/B/G
[    0.294426] rk628 2-0050: failed to request enable GPIO: -16
```

5.7.2 I2C 通信异常

如下 log 表示 RK628 的 I2C 通信异常导致 RK628 的各个模块注册不上，需要检查 RK628 的上电时序以及 24MHz 的基准时钟（见常见需求处理章节），以及相关 pin 的 iomux。

```
[    0.960609] rk628 1-0050: failed to access register: -6
```

5.7.3 DSI 或 GVI 显示有内容但花屏

可以尝试修改 dts 中 src-timing 消隐的正负极性，方法对 DSI 和 GVI 均有效。

例如以下修改：

```
--- a/arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2dsi-ddr4-v10.dts
+++ b/arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2dsi-ddr4-v10.dts
@@ -347,8 +347,8 @@ src-timing {
                                vfront-porch = <15>;
                                hsync-len = <2>;
                                vsync-len = <2>;
-                               hsync-active = <0>;
-                               vsync-active = <0>;
+                               hsync-active = <1>;
+                               vsync-active = <1>;
                                de-active = <0>;
                                pixelclk-active = <0>;
```

5.7.4 提高 clock-frequency 后 DSI 无法显示

为了提高刷新率，一般会提高 clock-frequency，例如下面的修改：

```
--- a/arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2dsi-ddr4-v10.dts
+++ b/arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2dsi-ddr4-v10.dts
@@ -338,7 +338,7 @@ rk628-panel {

                display-timings {
                        src-timing {
-                               clock-frequency = <132000000>;
+                               clock-frequency = <148500000>;
                                hactive = <1080>;
                                vactive = <1920>;
                                hback-porch = <30>;
@@ -354,7 +354,7 @@ src-timing {
                        };

                        dst-timing {
-                               clock-frequency = <132000000>;
+                               clock-frequency = <148500000>;
                                hactive = <1080>;
                                vactive = <1920>;
                                hback-porch = <30>;
```

但提高 CLK 可能导致屏幕闪屏或无法显示，如果出现这种情况，则可以尝试按如下方式调整 LANE 的速率解决此问题。

```
--- a/arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2dsi-ddr4-v10.dts
+++ b/arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-rgb2dsi-ddr4-v10.dts
@@ -57,6 +57,7 @@ &i2c2_rk628 {

                rk628-rgb-in;
                rk628-dsi-out {
+                       rockchip,lane-mbps = <1000>;
                        //rockchip,dual-channel;
                        dsi,eotp;
                        dsi,video-mode;
```

5.7.5 显示有偏移问题

例如，HDMI 做为输入，后端显示有偏移，根据偏移方向，在对应的 DTS 中调整屏端的 timing 消隐（src-timing/dst-timing 中的行消隐 hfront-porch/hback-porch/hsync-len 和列消隐 vfront-porch/vback-porch/vsync-len）。同时，要确保后端输出的 clock-frequency 与 HDMI 输入保持相同，例如都是 148500000。

```
--- a/arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-hdmi2dsi-ddr4-v10.dts
+++ b/arch/arm64/boot/dts/rockchip/rk3568-evb-rk628-hdmi2dsi-ddr4-v10.dts
@@ -314,7 +314,7 @@ src-timing {
                        hfront-porch = <60>;
                        vback-porch = <10>;
                        vfront-porch = <10>;
-                       hsync-len = <20>;
+                       hsync-len = <40>;
                        vsync-len = <10>;
                        hsync-active = <1>;
                        vsync-active = <1>;
```

5.7.6 如何操作 RK628 的 GPIO

在需要控制 GPIO 的位置直接调用 GPIO 接口即可（调用时需要包含头文件 rk628_pinctrl.h、rk628_gpio.h），该接口包括 IOMUX 设置，GPIO 方向设置和电平设置，例如以下操作将 I2S_D2_M0 接口设为 GPIO，并输出高电平。

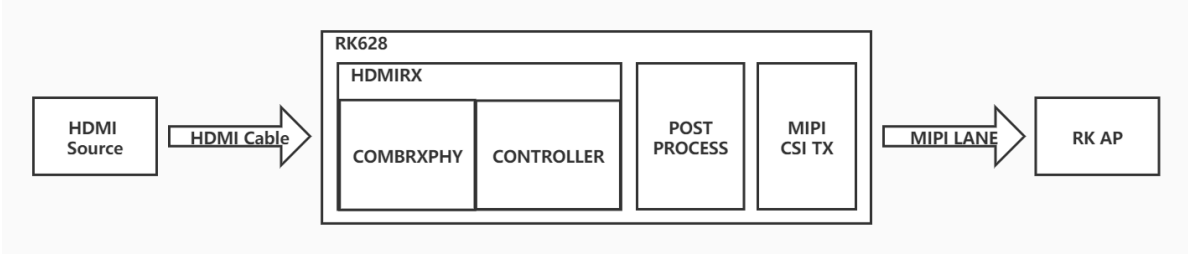
```
#include "rk628_pinctrl.h"
#include "rk628_gpio.h"

rk628_misc_gpio_direction_output(rk628, GPIO0_A6, 1);
```

若需要恢复则需要调用以下接口。

```
rk628_misc_pinctrl_set_mux(rk628, GPIO0_A6, I2SM0D2);
```

6. Media

6.1 驱动介绍



Media 为 RK628 HDMI IN 通路的驱动代码，将 RK628 作为类 camera 设备使用，实现如上功能。

6.1.1 RK628F/H 对比 RK628D 改进

media 框架驱动主要应用在 HDMI2CSI、HDMI2DSI、HDMI2BT1120 对接 SOC 主控平台的场景，对比 RK628D，RK628F/H 这部分主要改进如下：

模块	主要改进点
HDMIRX	支持 HDMI2.0 支持 RGB 4K60 输入 增加更多频点支持 支持 DVI mode 锁定更快更稳定
MIPI-CSI	解决了不同图像格式输入，格式转换下采样引入的色偏问题 增加一路 MIPI-CSI，双 MIPI-CSI 支持 4K60 输出
CSC	新增 matrix，可对图像色域做更灵活转换

6.1.2 驱动目录结构

RK628 在 media 框架下的驱动代码需要需要根据内核版本进行区分。kernel-5.10 之前的版本，使用 rk628_for-all 的补丁包，kernel-5.10 的版本，使用 kernel_5.10_rk628_media_patch 的补丁包。

如果是 kernel-5.10 之前的版本，驱动目录结构如下：

```
drivers/media/i2c/rk628
├── Makefile
├── rk628_bt1120_v4l2.c
├── rk628.c
├── rk628_combrxphy.c
├── rk628_combrxphy.h
├── rk628_combtxphy.c
├── rk628_combtxphy.h
├── rk628_cru.c
├── rk628_cru.h
├── rk628_csi.c
├── rk628_csi.h
```

```
├── rk628_csi_v4l2.c
├── rk628_dsi.c
├── rk628_dsi.h
├── rk628_gpio.h
├── rk628_grf.h
├── rk628.h
├── rk628_hdmirx.c
├── rk628_hdmirx.h
├── rk628_mipi_dphy.c
├── rk628_mipi_dphy.h
├── rk628_pinctrl.c
├── rk628_pinctrl.h
├── rk628_post_process.c
├── rk628_post_process.h
└── rk628_v4l2_controls.h
```

如果是 kernel-5.10 的内核版本，驱动目录如下：

```
drivers/media/i2c/rk628
├── Makefile
├── rk628_bt1120_v4l2.c
├── rk628.c
├── rk628_combrxphy.c
├── rk628_combrxphy.h
├── rk628_combtxphy.c
├── rk628_combtxphy.h
├── rk628_cru.c
├── rk628_cru.h
├── rk628_csi.c
├── rk628_csi.h
├── rk628_csi_v4l2.c
├── rk628_dsi.c
├── rk628_dsi.h
├── rk628.h
├── rk628_hdmirx.c
├── rk628_hdmirx.h
├── rk628_mipi_dphy.c
├── rk628_mipi_dphy.h
├── rk628_post_process.c
└── rk628_post_process.h
```

目前主要的几个驱动文件及所适用的场景如下：

rk628_csi.c: HDMI2CSI 应用场景，适用于 CameraHal1 及第三方 SOC；

rk628_csi_v4l2.c: HDMI2CSI / HDMI2DSI 应用场景，适用于 CameraHal3/TV input 等基于 v4l2 框架的平台；

rk628_bt1120_v4l2.c: HDMI2BT1120 应用场景，适用于 CameraHal3 等基于 v4l2 框架的平台；

6.1.3 驱动移植说明

参考《驱动移植说明》-《HDMI IN 通路（drivers/media/i2c/rk628）》章节

6.2 HDMI IN VIDEO 框架说明

HDMI IN video 部分的软件实现方案是将 RK628 模拟成一个 MIPI SOC camera 设备，通过 camera 框架或者 TV 框架接收 video 数据并在 APK 进行显示，同时基于 HDMI IN 的应用场景需要，增加 HDMI IN 热拔插和 HDMI IN 分辨率自适应支持。

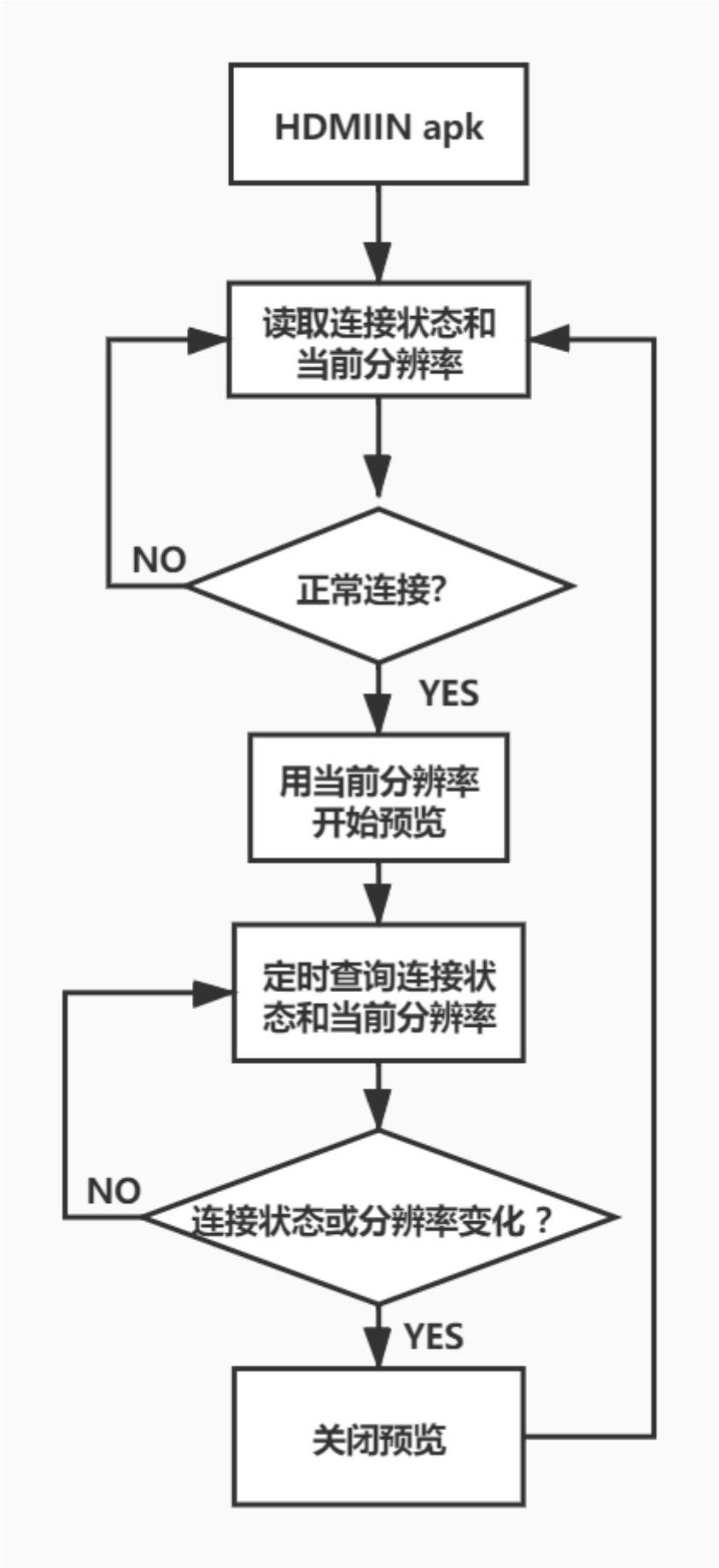
Android9/10/11 基于 cameraHAL3 框架开发，可以参考 SDK 目录 RKDocs/common/camera/hdmi-in 文档：《Rockchip_Developer_Guide_HDMI_IN_Based_On_CameraHal3_CN.pdf》

Android12/13 基于 cameraHAL3/TVinput 框架开发，可以参考 SDK 目录 RKDocs/common/camera/hdmi-in 文档：

《Rockchip_Developer_Guide_Android12+_HDMI_IN_Bridge_CN》

6.2.1 HDMI IN APK 工作流程

APK 大致工作流程如下图所示：



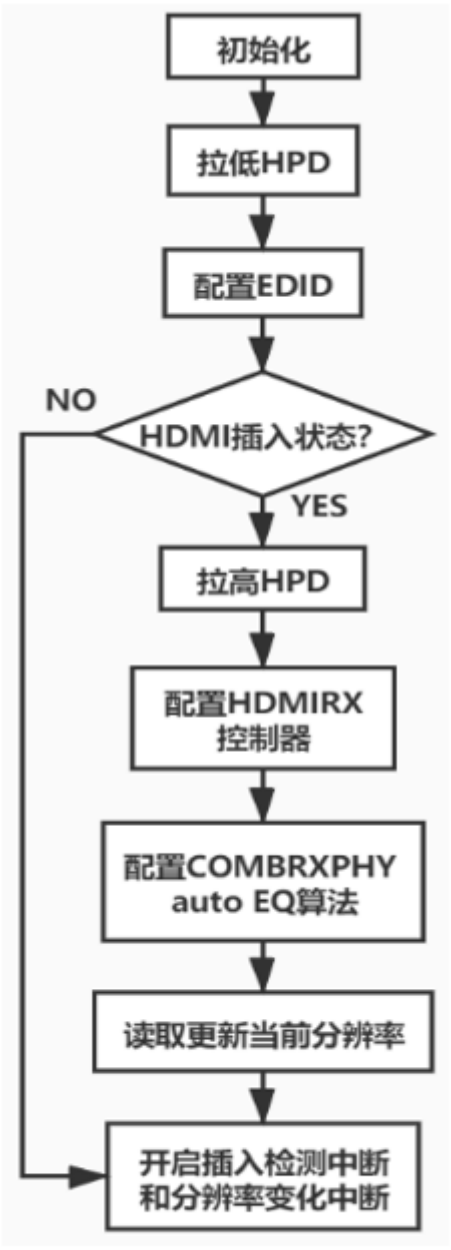
基于 kernel-4.19 或者 kernel-4.4 版本（安卓 9/10/11），可以直接使用 rkCamera2 打开预览。

基于 kernel-5.10 版本（安卓 12/13），需要区分是 camera 框架还是 TV 框架，详细请参考文档：

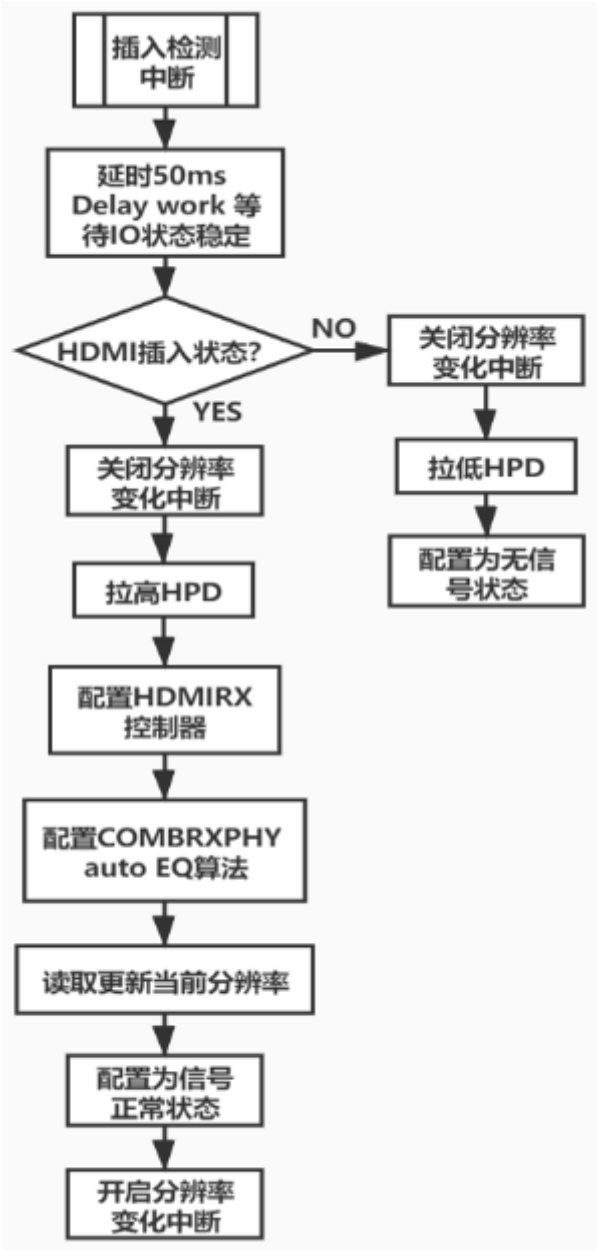
《Rockchip_Developer_Guide_Android12+_HDMI_IN_Bridge_CN》

6.2.2 RK628 驱动架构

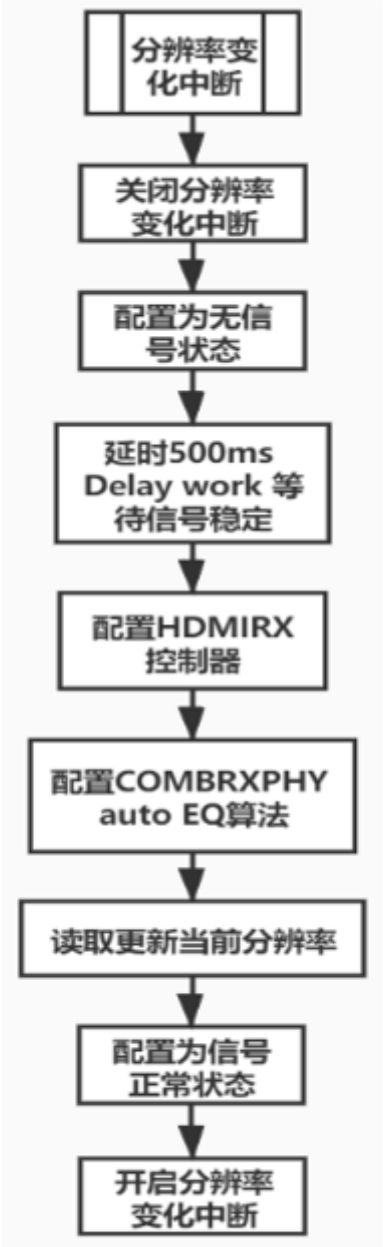
RK628 驱动需要重点关注的主要有三个部分：初始化、热拔插中断处理、分辨率切换中断处理。流程图参考如下：



驱动初始配置流程



热拔插中断处理程序



分辨率变化中断处理程序

6.3 dts 配置说明

6.3.1 RK628 节点配置

```
&i2c5 {
    status = "okay";

    rk628_csi: rk628_csi@50 {
        reg = <0x50>;
        compatible = "rockchip,rk628-csi-v4l2";
        status = "okay";
        power-domains = <&power RK3588_PD_VI>;
        pinctrl-names = "default";
        pinctrl-0 = <&rk628_pin>;
        interrupt-parent = <&gpio2>;
        interrupts = <RK_PC4 IRQ_TYPE_LEVEL_HIGH>;
```

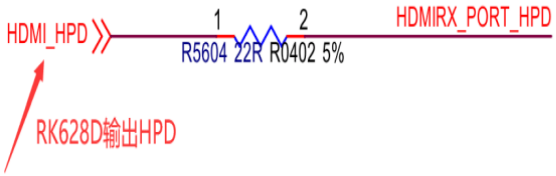
```
        enable-gpios = <&gpio1 RK_PA0 GPIO_ACTIVE_HIGH>;
        reset-gpios = <&gpio4 RK_PC6 GPIO_ACTIVE_HIGH>;
        plugin-det-gpios = <&gpio1 RK_PA1 GPIO_ACTIVE_LOW>;
        continues-clk = <1>;

        rockchip,camera-module-index = <0>;
        rockchip,camera-module-facing = "back";
        rockchip,camera-module-name = "HDMI-MIPI";
        rockchip,camera-module-lens-name = "RK628-CSI";

        multi-dev-info {
            dev-idx-l = <0>;
            dev-idx-r = <1>;
            combine-idx = <0>;
            pixel-offset = <0>;
            dev-num = <2>;
        };

        port {
            hdmiin_out0: endpoint {
                remote-endpoint = <&hdmi_mipi0_in>;
                data-lanes = <1 2 3 4>;
            };
        };
    };
};
```

- **reg**：I2C 地址；RK628 典型的 7bit I2C 地址为 0x50，使用多片 RK628 时，可通过 RK628 的 GPIO 改变 I2C 地址，具体参考《SOC 与 RK628 I2C 通信》章节；
- **compatible**：

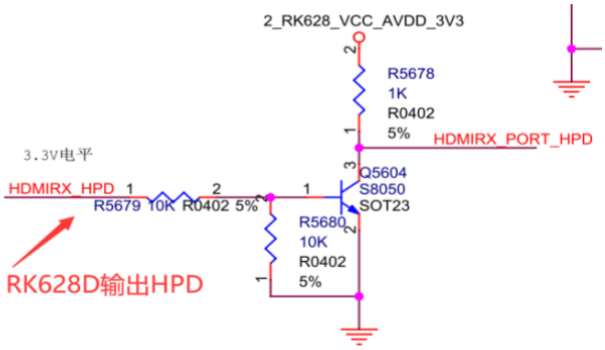
  compatible = "rockchip,rk628-csi" 对应 rk628_csi.c；
  compatible = "rockchip,rk628-csi-v4l2" 对应 rk628_csi_v4l2.c；
  compatible = "rockchip,rk628-bt1120-v4l2" 对应 rk628_bt1120_v4l2.c；

- **interrupt-parent/ interrupts**：连接 RK628 中断的 GPIO 引脚；
- **enable-gpios**：RK628 供电控制 GPIO 引脚（若为常供电可不配置）；
- **reset-gpios**：RK628 复位控制 GPIO 引脚；
- **hpd-output-inverted**：HPD 输出取反配置，若 HPD 输出电平在电路上做了取反，则需要使能此配置项；

◆ **HPD未取反设计：**

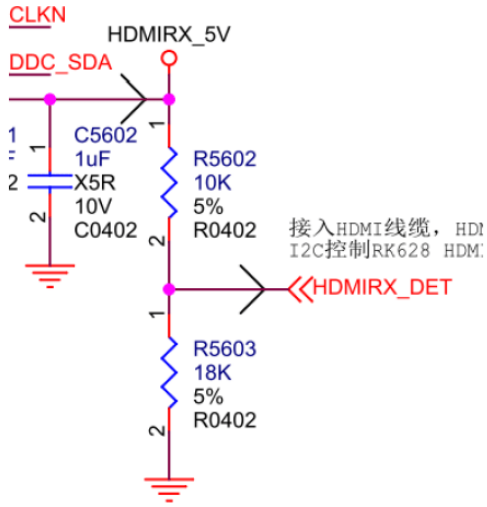


◆ **HPD取反设计：**

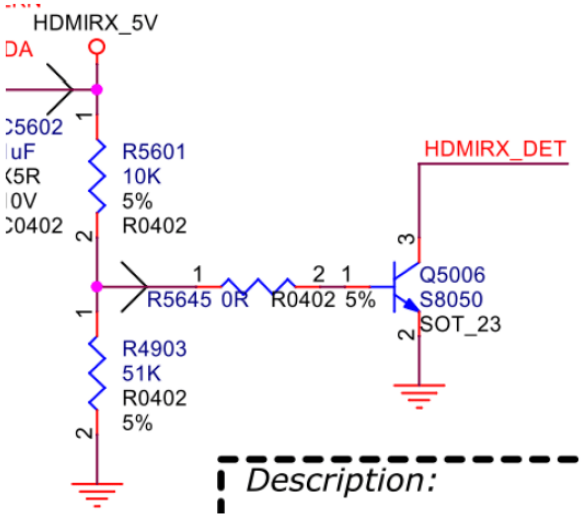


- **plugin-det-gpios**：HDMI 插入检测 GPIO 引脚，注意电路上是否有对电平取反，有效电平需要配置正确；

◆ HDMIRX_DET未取反设计：



◆ HDMIRX_DET取反设计：



- **power-gpios**：RK 主控端 MIPI RX 电源域供电控制 GPIO 引脚（若为常供电可不配置）；
- **hdcp-enable**: 使能 HDCP 功能；
- **continues-clk**：选择 MIPI CSI 输出连续模式或者非连续模式的 CLK。
- **multi-dev-info**：双 MIPI 支持 4K60 场景下需要该配置（仅 RK628F/H 支持），目前 SOC 仅 RK3588/RK3562/RK3576，内核5.10以上版本 支持。

6.3.2 图像接收链路组合

将 RK628 转换芯片作为类 camera 设备进行开发，与 camera sensor 一样需要实现基于 V4L2 框架的驱动，数据链路配置的方法与 MIPI SOC Sensor 一致。常见的链路组合有以下三种：

- RK628 -> HDMI2CSI -> SOC
- RK628 -> HDMI2DSI -> SOC
- RK628 -> HDMI2BT1120 -> SOC

6.3.3 HDMI2CSI 链路

6.3.3.1 RK628D

以 RK3288 + CSI_V4L2 为例，rk628 + isp1 链路配置

```
&rk628_csi {
    status = "okay";
    /*
     * If the hpd output level is inverted on the circuit,
     * the following configuration needs to be enabled.
     */
    /* hpd-output-inverted; */
    plugin-det-gpios = <&gpio0 13 GPIO_ACTIVE_HIGH>;
    power-gpios = <&gpio0 17 GPIO_ACTIVE_HIGH>;
    rockchip,camera-module-index = <0>;
    rockchip,camera-module-facing = "back";
    rockchip,camera-module-name = "RK628-CSI";
    rockchip,camera-module-lens-name = "NC";
```

```
    port {
        hdmiin_out0: endpoint {
            remote-endpoint = <&mipi_in>;
            data-lanes = <1 2 3 4>;
        };
    };
};

&mipi_phy_rx0 {
    status = "okay";

    ports {
        #address-cells = <1>;
        #size-cells = <0>;

        port@0 {
            reg = <0>;
            #address-cells = <1>;
            #size-cells = <0>;

            mipi_in: endpoint@1 {
                reg = <1>;
                remote-endpoint = <&hdmiin_out0>;
                data-lanes = <1 2 3 4>;
            };
        };

        port@1 {
            reg = <1>;
            #address-cells = <1>;
            #size-cells = <0>;

            dphy_rx_out: endpoint@0 {
                reg = <0>;
                remote-endpoint = <&isp_mipi_in>;
            };
        };
    };
};

&rkisp1 {
    status = "okay";
    port {
        #address-cells = <1>;
        #size-cells = <0>;

        isp_mipi_in: endpoint@0 {
            reg = <0>;
            remote-endpoint = <&dphy_rx_out>;
        };
    };
};

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

以 **RK3568 + CSI_V4L2 为例，rk628 + isp2 链路配置：**

```
&i2c2 {
    status = "okay";
    pinctrl-names = "default";
    pinctrl-0 = <&i2c2m1_xfer>;

    rk628_csi: rk628_csi@32 {
        compatible = "rockchip,rk628-csi-v4l2";
        reg = <0x50>;
        interrupt-parent = <&gpio4>;
        interrupts = <16 IRQ_TYPE_LEVEL_LOW>;
        pinctrl-names = "default";
        pinctrl-0 = <&rk628_irq>;
        reset-gpios = <&gpio4 RK_PD2 GPIO_ACTIVE_LOW>;
        plugin-det-gpios = <&gpio0 RK_PD6 GPIO_ACTIVE_LOW>;
        rockchip,camera-module-index = <0>;
        rockchip,camera-module-facing = "back";
        rockchip,camera-module-name = "RK628-CSI";
        rockchip,camera-module-lens-name = "NC";
        port {
            rk628_out: endpoint {
                remote-endpoint = <&mipi_in>;
                data-lanes = <1 2 3 4>;
            };
        };
    };
};

&csi2_dphy_hw {
    status = "okay";
};

&csi2_dphy0 {
    status = "okay";

    ports {
        #address-cells = <1>;
        #size-cells = <0>;
        port@0 {
            reg = <0>;
            #address-cells = <1>;
            #size-cells = <0>;

            mipi_in: endpoint@0 {
                reg = <0>;
                remote-endpoint = <&rk628_out>;
                data-lanes = <1 2 3 4>;
            };
        };
        port@1 {
            reg = <1>;
            #address-cells = <1>;
            #size-cells = <0>;

            csidphy0_out: endpoint@0 {
                reg = <0>;
```

```
                remote-endpoint = <&isp0_in>;
            };
        };
    };
};

&rkisp {
    status = "okay";
};

&rkisp_mmu {
    status = "okay";
};

&rkisp_vir0 {
    status = "okay";

    port {
        #address-cells = <1>;
        #size-cells = <0>;

        isp0_in: endpoint@0 {
            reg = <0>;
            remote-endpoint = <&csidphy0_out>;
        };
    };
};
```

RK3568，rk628 + vicap 链路配置：

```
&i2c2 {
    status = "okay";
    pinctrl-names = "default";
    pinctrl-0 = <&i2c2m1_xfer>;

    rk628_csi: rk628_csi@32 {
        compatible = "rockchip,rk628-csi-v4l2";
        reg = <0x50>;
        //clocks = <&ext_cam_clk>;
        //clock-names = "xvclk";

        interrupt-parent = <&gpio4>;
        interrupts = <16 IRQ_TYPE_LEVEL_LOW>;
        pinctrl-names = "default";
        pinctrl-0 = <&rk628_irq>;
        //power-gpios = <&gpio0 RK_PD5 GPIO_ACTIVE_HIGH>;
        reset-gpios = <&gpio4 RK_PD2 GPIO_ACTIVE_LOW>;
        plugin-det-gpios = <&gpio0 RK_PD6 GPIO_ACTIVE_LOW>;
        rockchip,camera-module-index = <0>;
        rockchip,camera-module-facing = "back";
        rockchip,camera-module-name = "RK628-CSI";
        rockchip,camera-module-lens-name = "NC";
        port {
            rk628_out: endpoint {
                remote-endpoint = <&mipi_in>;
                data-lanes = <1 2 3 4>;
            };
```

```
        };
    };
};

&csi2_dphy_hw {
    status = "okay";
};

&csi2_dphy0 {
    status = "okay";

    ports {
        #address-cells = <1>;
        #size-cells = <0>;
        port@0 {
            reg = <0>;
            #address-cells = <1>;
            #size-cells = <0>;

            mipi_in: endpoint@0 {
                reg = <0>;
                remote-endpoint = <&rk628_out>;
                data-lanes = <1 2 3 4>;
            };
        };
        port@1 {
            reg = <1>;
            #address-cells = <1>;
            #size-cells = <0>;

            csidphy0_out: endpoint@0 {
                reg = <0>;
                remote-endpoint = <&mipi_csi2_input>;
                data-lanes = <1 2 3 4>;
            };
        };
    };
};

&mipi_csi2 {
    status = "okay";

    ports {
        #address-cells = <1>;
        #size-cells = <0>;

        port@0 {
            reg = <0>;
            #address-cells = <1>;
            #size-cells = <0>;

            mipi_csi2_input: endpoint@1 {
                reg = <1>;
                remote-endpoint = <&csidphy0_out>;
                data-lanes = <1 2 3 4>;
            };
        };
```

```
        port@1 {
            reg = <1>;
            #address-cells = <1>;
            #size-cells = <0>;

            mipi_csi2_output: endpoint@0 {
                reg = <0>;
                remote-endpoint = <&cif_mipi_in>;
                data-lanes = <1 2 3 4>;
            };
        };
    };
};

&rkcif {
    status = "okay";
};

&rkcif_mipi_lvds {
    status = "okay";

    port {
        cif_mipi_in: endpoint {
            remote-endpoint = <&mipi_csi2_output>;
            data-lanes = <1 2 3 4>;
        };
    };
};

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

6.3.3.2 RK628F/H

RK628F/H 对接 SOC DTS 配置与 RK628D 基本一致，可以参考 RK628D。

RK628F/H 支持双 MIPI 的配置，目前仅支持搭配 RK3588/RK3562 主控，下面以 RK3588 双 MIPI 为例，rk628F /H + vicap 链路配置，RK628F/H CSI0/CSI1 分别接入 RK3588 CSI0/CSI1

```
&csi2_dphy0 {
    status = "okay";

    ports {
        #address-cells = <1>;
        #size-cells = <0>;
        port@0 {
            reg = <0>;
            #address-cells = <1>;
            #size-cells = <0>;

            hdmi_mipi2_in: endpoint@1 {
                reg = <1>;
                remote-endpoint = <&hdmiin_out1>;
                data-lanes = <1 2 3 4>;
            };
```

```
        };
        port@1 {
            reg = <1>;
            #address-cells = <1>;
            #size-cells = <0>;

            csidphy0_out: endpoint@0 {
                reg = <0>;
                remote-endpoint = <&mipi2_csi2_input>;
            };
        };
    };
};

&csi2_dphy0_hw {
    status = "okay";
};

&csi2_dphy1_hw {
    status = "okay";
};

&i2c3 {
    status = "okay";
    clock-frequency = <400000>;

    rk628_csi: rk628_csi@50 {
        reg = <0x50>;
        compatible = "rockchip,rk628-csi-v4l2";
        status = "okay";
        power-domains = <&power RK3588_PD_VI>;
        pinctrl-names = "default";
        pinctrl-0 = <&rk628_pin>;
        interrupt-parent = <&gpio1>;
        interrupts = <RK_PB2 IRQ_TYPE_LEVEL_HIGH>;
        enable-gpios = <&gpio1 RK_PA7 GPIO_ACTIVE_HIGH>;
        reset-gpios = <&gpio1 RK_PB1 GPIO_ACTIVE_HIGH>;
        plugin-det-gpios = <&gpio2 RK_PB6 GPIO_ACTIVE_LOW>;
        continues-clk = <1>;

        rockchip,camera-module-index = <0>;
        rockchip,camera-module-facing = "back";
        rockchip,camera-module-name = "HDMI-MIPI2";
        rockchip,camera-module-lens-name = "RK628-CSI";

        multi-dev-info {
            dev-idx-l = <2>;
            dev-idx-r = <4>;
            combine-idx = <2>;
            pixel-offset = <0>;
            dev-num = <2>;
        };

        port {
            hdmiin_out1: endpoint {
                remote-endpoint = <&hdmi_mipi2_in>;
                data-lanes = <1 2 3 4>;
            };
```

```
        };
    };
};

&mipi_dcphy0 {
    status = "okay";
};

&mipi_dcphy1 {
    status = "okay";
};

&mipi2_csi2 {
    status = "okay";

    ports {
        #address-cells = <1>;
        #size-cells = <0>;

        port@0 {
            reg = <0>;
            #address-cells = <1>;
            #size-cells = <0>;

            mipi2_csi2_input: endpoint@1 {
                reg = <1>;
                remote-endpoint = <&csidphy0_out>;
            };
        };

        port@1 {
            reg = <1>;
            #address-cells = <1>;
            #size-cells = <0>;

            mipi2_csi2_output: endpoint@0 {
                reg = <0>;
                remote-endpoint = <&cif_mipi_in2>;
            };
        };
    };
};

&rkcif {
    status = "okay";
};

&rkcif_mipi_lvds2 {
    status = "okay";

    port {
        cif_mipi_in2: endpoint {
            remote-endpoint = <&mipi2_csi2_output>;
        };
    };
};

&rkcif_mmu {
```

```
    status = "okay";
};
```

6.3.4 HDMI2DSI 转换

6.3.4.1 RK628D

RK628D 在 HDMI To MIPI CSI 应用场景存在色域空间转换或图像格式的上下采样，图像色彩少量细节可能会有轻微偏差，可以采用 HDMI To MIPI DSI 替代 HDMI To MIPI CSI 方案，主控直接接收 RGB888 格式图像，应用层使用 rgb2yuv 算法库对其进行格式转换。需要注意的是：

最大支持分辨率 1080P60；

目前支持接收 MIPI DSI 的 AP：RK1109、RK1126、RK3566、RK3568、RK3588、RK3562、RK3576；

不支持接收 MIPI DSI 的 AP：RK3288、RK3326、RK3368、RK3399 等早期芯片。

CSI 与 DSI 代码兼容，只是 dts 的 compatible 做区分：

CSI 为 compatible = "rockchip,rk628-csi-v4l2";

DSI 为 compatible = "rockchip,rk628-dsi-v4l2"；

以 **RK3568** 为例

```
&i2c2 {
    status = "okay";
    pinctrl-names = "default";
    pinctrl-0 = <&i2c2m1_xfer>;

    rk628_csi: rk628_csi@32 {
        compatible = "rockchip,rk628-dsi-v4l2";
        reg = <0x50>;

        interrupt-parent = <&gpio4>;
        interrupts = <16 IRQ_TYPE_LEVEL_LOW>;
        pinctrl-names = "default";
        pinctrl-0 = <&rk628_irq>;
        //power-gpios = <&gpio0 RK_PD5 GPIO_ACTIVE_HIGH>;
        reset-gpios = <&gpio4 RK_PD2 GPIO_ACTIVE_LOW>;
        plugin-det-gpios = <&gpio0 RK_PD6 GPIO_ACTIVE_LOW>;
        rockchip,camera-module-index = <0>;
        rockchip,camera-module-facing = "back";
        rockchip,camera-module-name = "RK628-CSI";
        rockchip,camera-module-lens-name = "NC";
        port {
            rk628_out: endpoint {
                remote-endpoint = <&hdmi2mipi_in>;
                data-lanes = <1 2 3 4>;
            };
        };
    };
};

&mipi_csi2 {
    status = "okay";
```

```
    ports {
        #address-cells = <1>;
        #size-cells = <0>;

        port@0 {
            reg = <0>;
            #address-cells = <1>;
            #size-cells = <0>;

            mipi_csi2_input: endpoint@1 {
                reg = <1>;
                remote-endpoint = <&csidphy_out>;
                data-lanes = <1 2 3 4>;
            };
        };

        port@1 {
            reg = <1>;
            #address-cells = <1>;
            #size-cells = <0>;

            mipi_csi2_output: endpoint@0 {
                reg = <0>;
                remote-endpoint = <&cif_mipi_in>;
                data-lanes = <1 2 3 4>;
            };
        };
    };
};

&rkcif {
    status = "okay";
};

&rkcif_mipi_lvds {
    status = "okay";

    port {
        cif_mipi_in: endpoint {
            remote-endpoint = <&mipi_csi2_output>;
            data-lanes = <1 2 3 4>;
        };
    };
};

&rkcif_mmu {
    status = "okay";
};

&csi2_dphy_hw {
    status = "okay";
};

&csi2_dphy0 {
    status = "okay";

    ports {
        #address-cells = <1>;
```

```
        #size-cells = <0>;
        port@0 {
            reg = <0>;
            #address-cells = <1>;
            #size-cells = <0>;

            hdmi2mipi_in: endpoint@0 {
                reg = <1>;
                remote-endpoint = <&rk628_out>;
                data-lanes = <1 2 3 4>;
            };
        };
        port@1 {
            reg = <1>;
            #address-cells = <1>;
            #size-cells = <0>;

            csidphy_out: endpoint@0 {
                reg = <0>;
                remote-endpoint = <&mipi_csi2_input>;
            };
        };
    };
};
```

若主控是 RK3568，代码版本比较旧的话，CIF 需要包含下面的提交来支持 RGB888 接收：

RK3588 kernel-5.10 驱动代码默认支持 RGB888。

```
From e925e385a41a857e8e50020b5df9a2d41b166d83 Mon Sep 17 00:00:00 2001
From: Zefa Chen <zefa.chen@rock-chips.com>
Date: Thu, 02 Sep 2021 15:07:13 +0800
Subject: [PATCH] media: rockchip: cif fix errors in rgb24 data format

Signed-off-by: Zefa Chen <zefa.chen@rock-chips.com>
Change-Id: I0766997860f06dc25e604c2ea8425049a987fdc5
---

diff --git a/drivers/media/platform/rockchip/cif/capture.c
b/drivers/media/platform/rockchip/cif/capture.c
index 6c0aa7d..36bd2c9 100644
--- a/drivers/media/platform/rockchip/cif/capture.c
+++ b/drivers/media/platform/rockchip/cif/capture.c
@@ -1785,13 +1785,11 @@
      * needs aligned with :ALIGN(bits_per_pixel * width * 2, 8), to optimize
reading and
      * writing of ddr, aliged with 256
      */
-    if (fmt->fmt_type == CIF_FMT_TYPE_RAW && stream->is_compact) {
+    if (fmt->fmt_type == CIF_FMT_TYPE_RAW && stream->is_compact &&
+        fmt->csi_fmt_val != CSI_WRDDR_TYPE_RGB888) {
         channel->virtual_width = ALIGN(channel->width * fmt->raw_bpp / 8, 256);
     } else {
-        if (fmt->fmt_type == CIF_FMT_TYPE_RAW && fmt->csi_fmt_val !=
CSI_WRDDR_TYPE_RAW8)
-            channel->virtual_width = ALIGN(channel->width * 2, 8);
-        else
```

```
-            channel->virtual_width = ALIGN(channel->width * fmt->bpp[0] / 8, 8);
+        channel->virtual_width = ALIGN(channel->width * fmt->bpp[0] / 8, 8);
     }

     if (channel->fmt_val == CSI_WRDDR_TYPE_RGB888)
@@ -3240,7 +3238,8 @@

         if (fmt->fmt_type == CIF_FMT_TYPE_RAW && stream->is_compact &&
             (dev->active_sensor->mbus.type == V4L2_MBUS_CSI2 ||
-             dev->active_sensor->mbus.type == V4L2_MBUS_CCP2)) {
+             dev->active_sensor->mbus.type == V4L2_MBUS_CCP2) &&
+             fmt->csi_fmt_val != CSI_WRDDR_TYPE_RGB888) {
             bpl = ALIGN(width * fmt->raw_bpp / 8, 256);
         } else {
             bpp = rkcif_align_bits_per_pixel(stream, fmt, i);
@@ -4659,13 +4658,11 @@
      * needs aligned with :ALIGN(bits_per_pixel * width * 2, 8), to optimize
reading and
      * writing of ddr, aliged with 256
      */
-    if (fmt->fmt_type == CIF_FMT_TYPE_RAW && stream->is_compact) {
+    if (fmt->fmt_type == CIF_FMT_TYPE_RAW && stream->is_compact &&
+        fmt->csi_fmt_val != CSI_WRDDR_TYPE_RGB888) {
         *crop_vwidth = ALIGN(raw_width * fmt->raw_bpp / 8, 256);
     } else {
-        if (fmt->fmt_type == CIF_FMT_TYPE_RAW)
-            *crop_vwidth = ALIGN(raw_width * 2, 8);
-        else
-            *crop_vwidth = ALIGN(raw_width * fmt->bpp[0] / 8, 8);
+        *crop_vwidth = ALIGN(raw_width * fmt->bpp[0] / 8, 8);
     }

     if (channel->fmt_val == CSI_WRDDR_TYPE_RGB888)
```

rk628-dsi 输出默认是 command mode，cif 接收也需要使用 command mode，修改如下:

```
diff --git a/drivers/media/platform/rockchip/cif/capture.c
b/drivers/media/platform/rockchip/cif/capture.c
index 36bd2c99c707..fc4181daed2c 100644
--- a/drivers/media/platform/rockchip/cif/capture.c
+++ b/drivers/media/platform/rockchip/cif/capture.c
@@ -1753,7 +1753,7 @@ static int rkcif_csi_channel_init(struct rkcif_stream
*stream,

        channel->fmt_val = stream->cif_fmt_out->csi_fmt_val;

-       channel->cmd_mode_en = 0; /* default use DSI Video Mode */
+       channel->cmd_mode_en = 1; /* default use DSI Video Mode */

        if (stream->crop_enable) {
                channel->crop_en = 1;
```

对接 camera 框架

若主控是 RK356X 系列，只能对接 camera 框架，camera 框架不支持 RGB 直接送显示，因此需要加上 rgb2yuv 算法库。

CameraHal 加上如下 rgb2yuv 算子库，若 SDK 代码版本是安卓 11-R9，则可以直接加上下面的补丁，具体见补丁 rk356x_HAL3_support_rgb2yuv_patch.rar

若代码版本是 R10 或者 R11 的，在 hardware/rockchip/camera 下先添加如下修改，再打上上述补丁

```
diff --git a/psl/rkisp2/RKISP2GraphConfig.cpp b/psl/rkisp2/RKISP2GraphConfig.cpp
index 2a5fa5a..3d2409c 100755
--- a/psl/rkisp2/RKISP2GraphConfig.cpp
+++ b/psl/rkisp2/RKISP2GraphConfig.cpp
@@ -76,6 +76,7 @@ const string MEDIACTL_POSTVIEWNAME = "postview";

 const string MEDIACTL_STATNAME = "rkisp1-statistics";
 const string MEDIACTL_VIDEONAME_CIF = "stream_cif_dvp_id0";
+const string MEDIACTL_VIDEONAME_CIF_MIPI_ID0 = "stream_cif_mipi_id0";

 RKISP2GraphConfig::RKISP2GraphConfig() :
         mManager(nullptr),
@@ -2620,6 +2621,13 @@ status_t RKISP2GraphConfig::getImguMediaCtlConfig(int32_t
cameraId,
                 addLinkParams("rkisp-isp-subdev", 2, "rkisp_mainpath", 0, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
                 addLinkParams("rkisp-isp-subdev", 2, "rkisp_selfpath", 0, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
             }
+        } else if(mipName2.find("mipi") != std::string::npos) {
+            addLinkParams(mipName, mipSrcPad, mipName2, csiSinkPad, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
+            addLinkParams(mipName2, 1, "stream_cif_mipi_id0", 0, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
+            addLinkParams(mipName2, 2, "stream_cif_mipi_id1", 0, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
+            addLinkParams(mipName2, 3, "stream_cif_mipi_id2", 0, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
+            addLinkParams(mipName2, 4, "stream_cif_mipi_id3", 0, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
+            mSensorLinkedToCIF = true;
         } else {
             addLinkParams(mipName, mipSrcPad, csiName, csiSinkPad, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
             addLinkParams(csiName, csiSrcPad, IspName, ispSinkPad, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
@@ -2628,6 +2636,12 @@ status_t RKISP2GraphConfig::getImguMediaCtlConfig(int32_t
cameraId,
             addLinkParams(csiName, 5, "rkisp_rawwr3", 0, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
         }
     }
+    if(mSensorLinkedToCIF){
+        addImguVideoNode(IMGU_NODE_VIDEO, MEDIACTL_VIDEONAME_CIF_MIPI_ID0,
mediaCtlConfig);
+        addFormatParams(MEDIACTL_VIDEONAME_CIF_MIPI_ID0, mCurSensorFormat.width,
mCurSensorFormat.height,
+                0, V4L2_PIX_FMT_NV12, 0, 0, mediaCtlConfig);
+        return OK;
+    }
     // isp input pad format and selection config
```

```
    addFormatParams(IspName, ispInWidth, ispInHeight, ispSinkPad, ispInFormat,
0, 0, mediaCtlConfig);
    addSelectionParams(IspName, ispInWidth, ispInHeight, 0, 0,
V4L2_SEL_TGT_CROP, ispSinkPad, mediaCtlConfig);
```

请注意，该补丁仅仅适用于 RK3568+ANDROID11 平台使用。

对接 TV 框架

如果 HDMI to DSI 的通路搭配的是 RK3588或者RK3576 使用的话，应用框架可以走 TV 框架，可以支持 RGB888 格式直接送显示，不需要加上述 rgb2yuv 的算法库，具体在 APK 适配方法章节描述。

6.3.4.2 RK628F/H

RK628F/H 解决了 RK628D 在不同图像格式输入时，格式转换下采样引入的色偏问题。

但在某些对画质要求比较高的场景，RK628F/H 可以通过 DSI 接口输出 RGB888 图像格式，RK3588 等主控接收 RGB888 格式图像之后，通过 TV 框架直接送显示，不需要经过颜色转换，保证画质没有丢失。RK628F/H 有两个 MIPI-DSI，同样支持双 MIPI 模式对接 RK3588 主控。

以 RK3588 + RK628F/H 双 MIPI 为例

只需要将 compatible 修改成 DSI 的即可。

```
......
&i2c3 {
    status = "okay";
    clock-frequency = <400000>;

    rk628_csi: rk628_csi@50 {
        reg = <0x50>;
        compatible = "rockchip,rk628-dsi-v4l2";
        status = "okay";
        power-domains = <&power RK3588_PD_VI>;
        pinctrl-names = "default";
        pinctrl-0 = <&rk628_pin>;
        interrupt-parent = <&gpio1>;
        interrupts = <RK_PB2 IRQ_TYPE_LEVEL_HIGH>;
        enable-gpios = <&gpio1 RK_PA7 GPIO_ACTIVE_HIGH>;
        reset-gpios = <&gpio1 RK_PB1 GPIO_ACTIVE_HIGH>;
        plugin-det-gpios = <&gpio2 RK_PB6 GPIO_ACTIVE_LOW>;

        rockchip,camera-module-index = <0>;
        rockchip,camera-module-facing = "back";
        rockchip,camera-module-name = "HDMI-MIPI2";
        rockchip,camera-module-lens-name = "RK628-CSI";

        multi-dev-info {
            dev-idx-l = <2>;
            dev-idx-r = <4>;
            combine-idx = <2>;
            pixel-offset = <0>;
            dev-num = <2>;
        };

        port {
            hdmiin_out1: endpoint {
```

```
                remote-endpoint = <&hdmi_mipi2_in>;
                data-lanes = <1 2 3 4>;
            };
        };
    };
};
......
```

6.3.5 HDMI2BT1120 转换

HDMI2BT1120 链路配置上，RK628D 与 RK628F/H 基本一致，此处不再赘述。

以 RK3568 + BT1120_V4L2 为例

```
&i2c4 {
    pinctrl-names = "default";
    pinctrl-0 = <&i2c4m1_xfer>;
    clock-frequency = <400000>;
    status = "okay";

    rk628_bt1120: rk628_bt1120@50 {
        compatible = "rockchip,rk628-bt1120-v4l2";
        reg = <0x50>;
        status = "disabled";
        pinctrl-names = "default";
        pinctrl-0 = <&cif_dvp_clk &cif_dvp_bus16 &cif_dvp_bus8>;
        interrupt-parent = <&gpio2>;
        interrupts = <15 IRQ_TYPE_LEVEL_HIGH>;
        enable-gpios = <&gpio2 RK_PC0 GPIO_ACTIVE_HIGH>;
        reset-gpios = <&gpio2 RK_PB0 GPIO_ACTIVE_LOW>;
        plugin-det-gpios = <&gpio1 RK_PA2 GPIO_ACTIVE_LOW>;
        rockchip,camera-module-index = <0>;
        rockchip,camera-module-facing = "back";
        rockchip,camera-module-name = "RK628-BT1120";
        rockchip,camera-module-lens-name = "NC";
        dual-edge = <1>;

        port {
            lt8619c_out: endpoint {
                remote-endpoint = <&cif_para_in>;
                bus-width = <16>;
                pclk-sample = <1>;
            };
        };
    };
};

&rkcif_dvp {
    status = "okay";

    port {
        /* Parallel bus endpoint */
        cif_para_in: endpoint {
            remote-endpoint = <&lt8619c_out>;
        };
    };
```

```
};

&rkcif {
    status = "okay";
};

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

6.4 开启 HDCP 功能

RK628 HDMIRX 支持 HDCP1.4，HDCP 功能默认关闭，需要时打开：

1. dtsRK628 节点使能 HDCP

```
                reset-gpios = <&gpio7 RK_PB4 GPIO_ACTIVE_LOW>;
                plugin-det-gpios = <&gpio0 13 GPIO_ACTIVE_HIGH>;
                power-gpios = <&gpio0 17 GPIO_ACTIVE_HIGH>;
+               hdcp-enable = <1>;
```

2. 烧录 HDCP RX Key

   HDCP Key 需要从 HDCP 协会购买，RK 不提供 Key，请注意 HDCP 有区分 TX 和 RK Key，我们需要的是 RK Key；

   拿到 Key 源文件之后，用 KeyConverter 工具拆分 Key，并转成 skf 后缀的烧录文件，具体请看工具使用说明；

   转成.skf 文件之后，使用 RKDevInfoWriteTool 工具进行烧写，工具会对 Key 进行加密，具体请看工具使用说明；

6.5 开启 scaler 功能

有些平台不支持 MIPI 4K 接收，但是 RK628 HDMI IN 可以支持 4K 输入， 所以有客户需求，无论什么分辨率输入，CSI 都按 1080P 输出，dts 只需要添加如下配置：

```
                reset-gpios = <&gpio7 RK_PB4 GPIO_ACTIVE_LOW>;
                plugin-det-gpios = <&gpio0 13 GPIO_ACTIVE_HIGH>;
                power-gpios = <&gpio0 17 GPIO_ACTIVE_HIGH>;
+               scaler-en = <1>;
```

6.6 csi 支持 2 lanes

csi 默认配置是 4 lanes：

```
csi->csi_lanes_in_use = USE_4_LANES;
```

如果需要配置成 2 lanes，那么把 csi->csi_lanes_in_use 配置成 2；

6.7 连续 MIPI 模式与非连续 MIPI

设置连续 MIPI 模式，可在 DTS 配置：

```
continues-clk = <1>;
```

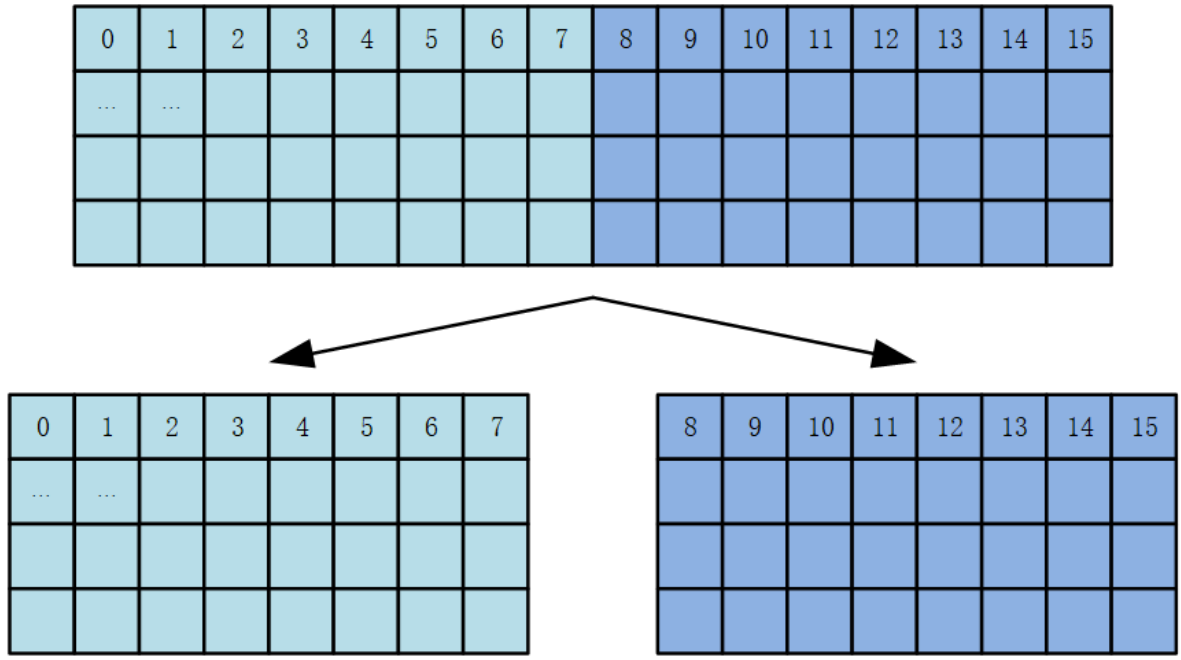
6.8 双 MIPI 模式配置（仅 RK628F/H 支持）

RK628F/H 受限于 MIPI DPHY 速率，一路 MIPI 最大只能支持 4K30，RK628F/H 有两路 MIPI-CSI，在对接 RK3588/RK3562 等有多路 MIPI-CSI 的主控平台时，4K60 的场景下，可以将 4K60 视频数据拆分成左右两半图像分别通过两路 4lane 的 MIPI-CSI 进行传输，由主控平台将图像进行合成为 4K60。

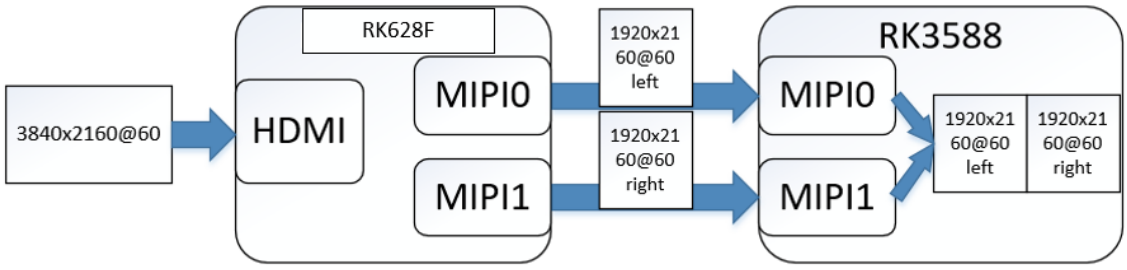
注：对接 RK3588 等 SOC 平台，图像拼接合成直接在 VI 驱动完成，不会额外增加延时以及负载。

6.8.1 双 MIPI split 模式说明

RK628F/H 内部支持可以将 HDMIRX 接收的图像 split 分割成左右两半，分别给到两路 CSI 进行传输，如下图所示：



基于上述，在对接 RK3588 等有多路 MIPI 接口的 SOC 平台时，就可以采用双 MIPI 模式，接收下 4K60 分辨率的图像数据，功能流程如下图所示。



6.8.2 双 MIPI 模式配置

以下介绍一下 RK628F/H 双 MIPI 模式对接 RK3588 主控，实现 4K60 HDMI-IN 功能的配置。

关于双 MIPI mode 对接 RK SOC 调试细节，具体可参考文档（SDK 目录 RKDocs/common/camera/HAL3）《Rockchip_RK3588_HDMI_To_CSI_Dual_Mipi_Developer_Guide》

6.8.2.1 RK628F/H 配置

RK628F/H 驱动版本已经支持双 MIPI 模式，默认在 HDMI-IN 的分辨率为 4K60 时，采用两个 MIPI-CSI 进行传输，DTS 配置 multi-dev-info 解析如下：

> dev-idx-l：左半边图像对应的 mipi 接口编号
>
> dev-idx-r：右半边图像对应的 mipi 接口编号
>
> combine-idx：将左右两张图想合并到其中的一个 idx，dts 的链路以这个 idx 为准
>
> pixel-offset：像素偏移，默认设置 0
>
> dev-num：多少个 mipi 设备进行拼接

举例如下，以 RK628F/H CSI0/1 分别接入 RK3588 CSI0/1 RX 为例：

```
&i2c3 {
    status = "okay";
    clock-frequency = <400000>;

    rk628_csi: rk628_csi@50 {
        reg = <0x50>;
        compatible = "rockchip,rk628-csi-v4l2";
        status = "okay";
        power-domains = <&power RK3588_PD_VI>;
        pinctrl-names = "default";
        pinctrl-0 = <&rk628_pin>;
        interrupt-parent = <&gpio1>;
        interrupts = <RK_PB2 IRQ_TYPE_LEVEL_HIGH>;
        enable-gpios = <&gpio1 RK_PA7 GPIO_ACTIVE_HIGH>;
        reset-gpios = <&gpio1 RK_PB1 GPIO_ACTIVE_HIGH>;
        plugin-det-gpios = <&gpio2 RK_PB6 GPIO_ACTIVE_LOW>;
        continues-clk = <1>;

        rockchip,camera-module-index = <0>;
        rockchip,camera-module-facing = "back";
        rockchip,camera-module-name = "HDMI-MIPI2";
        rockchip,camera-module-lens-name = "RK628-CSI";

        multi-dev-info {
            dev-idx-l = <2>;
            dev-idx-r = <4>;
            combine-idx = <2>;
            pixel-offset = <0>;
            dev-num = <2>;
        };

        port {
            hdmiin_out1: endpoint {
```

```
                remote-endpoint = <&hdmi_mipi2_in>;
                data-lanes = <1 2 3 4>;
            };
        };
    };
};
```

6.8.2.2 SOC 驱动代码版本要求

RK 平台的主控目前支持双 MIPI 模式的只有 RK3588/RK3562/RK3576，kernel-5.10 之后的版本才可支持，对 SOC 主控的代码版本有要求，建议将 SDK（安卓 12/13）代码更新到最新。如仅更新 kernel-5.10 RK628 驱动代码，可能会导致编译不过，即 SDK 代码版本太旧不支持双 MIPI 导致，请更新 SDK 代码版本或者 redmine 联系 RK 工程师解决。

6.9 不同芯片平台的接收能力

由于各个芯片平台 isp/vicap 的性能不同，对图像的最大接收能力也不同。可参考下表：

芯片平台	接收控制器	支持最大分辨率
RK3288/RK3326/RK3368	isp	1920x1080P60
RK3399	isp	3840x2160P30（注：isp 需要超频）
RK3566/RK3568	vicap/isp	3840x2160P30（注：连接ISP最大1080P60）
RK3588/RK3562/RK3576	vicap	3840x2160P60（双 MIPI）

6.9.1 配置 isp 超频的方法

- RK3399 配置 PLL_NPLL 为 650M：

```
--- a/arch/arm64/boot/dts/rockchip/rk3399-vop-clk-set.dtsi
+++ b/arch/arm64/boot/dts/rockchip/rk3399-vop-clk-set.dtsi
@@ -148,7 +148,7 @@
                         <50000000>, <100000000>,
                         <75000000>, <75000000>,
                         <816000000>, <816000000>,
-                        <600000000>, <200000000>,
+                        <650000000>, <200000000>,
                         <800000000>, <150000000>,
                         <75000000>, <37500000>,
                         <300000000>, <100000000>,
```

- 修改 rk3399 isp 最大支持频率为 650M：

```
--- a/drivers/media/platform/rockchip/isp1/dev.c
+++ b/drivers/media/platform/rockchip/isp1/dev.c
@@ -757,7 +757,7 @@ static const unsigned int rk3368_isp_clk_rate[] = {

 /* isp clock adjustment table (MHz) */
 static const unsigned int rk3399_isp_clk_rate[] = {
-       300, 400, 600
+       300, 400, 650
 };

 static struct isp_irqs_data rk1808_isp_irqs[] = {
```

- 如果 rk3288 概率性 isp 报错，isp 也可以抬频尝试，一般只需留下最高频即可，该修改仅用于定位问题即可。

```
--- a/drivers/media/platform/rockchip/isp1/dev.c
+++ b/drivers/media/platform/rockchip/isp1/dev.c
@@ -849,7 +849,7 @@ static const unsigned int rk1808_isp_clk_rate[] = {

 /* isp clock adjustment table (MHz) */
static const unsigned int rk3288_isp_clk_rate[] = {
-       150, 384, 500, 594
+       594
 };
```

- 转换芯片驱动中配置 isp 频率

```
#define RK628_CSI_PIXEL_RATE_HIGH       600000000
...

static int rk628_csi_set_fmt(struct v4l2_subdev *sd,
                struct v4l2_subdev_pad_config *cfg,
                struct v4l2_subdev_format *format)
{
...
        if ((mode->width == 3840) && (mode->height == 2160)) {
                v4l2_dbg(1, debug, sd,
                        "%s res wxh:%dx%d, link freq:%llu, pixrate:%u\n",
                        __func__, mode->width, mode->height,
                        link_freq_menu_items[1], RK628_CSI_PIXEL_RATE_HIGH);
                __v4l2_ctrl_s_ctrl(csi->link_freq, 1);
                __v4l2_ctrl_s_ctrl_int64(csi->pixel_rate,
                        RK628_CSI_PIXEL_RATE_HIGH);
        }
...
}
```

isp 驱动中会对配置的频率再加 25% 的余量，所以在驱动中配置适当的频率 RK628_CSI_PIXEL_RATE_HIGH 即可。

```
drivers/media/platform/rockchip/isp1/dev.c

static int __isp_pipeline_s_isp_clk(struct rkisp1_pipeline *p)
{
...
```

```
        ctrl = v4l2_ctrl_find(sd->ctrl_handler, V4L2_CID_PIXEL_RATE);
        if (!ctrl) {
                v4l2_warn(sd, "No pixel rate control in subdev\n");
                return -EPIPE;
        }

        /* calculate data rate */
        data_rate = v4l2_ctrl_g_ctrl_int64(ctrl) *
                    dev->isp_sdev.in_fmt.bus_width;
        data_rate >>= 3;
        do_div(data_rate, 1000 * 1000);

        /* increase 25% margin */
        data_rate += data_rate >> 2;
...
}
```

6.9.2 配置 ISP 使用 CMA 内存的方法

部分平台 HDMI IN 接收图像数据时，根据实际系统负载，可能会存在带宽不足导致丢帧或 MIPI 接收异常等问题，参考异常 log 如下：

```
[ 228.999567] rkisp1: MIPI mis error: 0x00800000
[ 228.999925] rkisp1: CIF_ISP_PIC_SIZE_ERROR (0x00000001)
[ 228.999976] rkisp1: CIF_ISP_PIC_SIZE_ERROR (0x00000001)rkisp1:
CIF_ISP_PIC_SIZE_ERROR (0x00000001)
[ 229.000081] rkisp1: CIF_ISP_PIC_SIZE_ERROR (0x00000001)rkisp1:
CIF_ISP_PIC_SIZE_ERROR (0x00000001)
[ 229.000187] rkisp1: CIF_ISP_PIC_SIZE_ERROR (0x00000001)rkisp1:
CIF_ISP_PIC_SIZE_ERROR (0x00000001)
[ 229.000294] rkisp1: CIF_ISP_PIC_SIZE_ERROR (0x00000001)rkisp1:
CIF_ISP_PIC_SIZE_ERROR (0x00000001)
```

此时需要提高 DDR 频率，若仍无改善，可给 ISP 预留使用 CMA 内存，以改善解决此问题：

- 在 rockchip_defconfig 配置预留 CMA 内存 128MB

```
CONFIG_CMA=y
CONFIG_CMA_SIZE_MBYTES=128
```

- 在 dts 配置 ISP 关闭 IOMMU，使用 CMA 内存

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

6.10 EDID 的配置方法

RK628 支持 EDID 配置，目前驱动代码中 EDID 支持的分辨率为：

3840x2160P60、3840x2160P30、1920x1080P60、1920x1080P30、1280x720P60、720x576P50、720x480P60 等常用分辨率，若需要修改分辨率支持，可直接在驱动代码中修改 EDID：

```
static u8 rk628f_edid_init_data[] = {
    0x00, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x00,
    0x24, 0xD0, 0x8F, 0x62, 0x01, 0x00, 0x00, 0x00,
    0x2D, 0x21, 0x01, 0x03, 0x80, 0x78, 0x44, 0x78,
    0x0A, 0xCF, 0x74, 0xA3, 0x57, 0x4C, 0xB0, 0x23,
    0x09, 0x48, 0x4C, 0x21, 0x08, 0x00, 0x61, 0x40,
    0x01, 0x01, 0x81, 0x00, 0x95, 0x00, 0xA9, 0xC0,
    0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x08, 0xE8,
    0x00, 0x30, 0xF2, 0x70, 0x5A, 0x80, 0xB0, 0x58,
    0x8A, 0x00, 0xC4, 0x8E, 0x21, 0x00, 0x00, 0x1E,
    0x02, 0x3A, 0x80, 0x18, 0x71, 0x38, 0x2D, 0x40,
    0x58, 0x2C, 0x45, 0x00, 0xB9, 0xA8, 0x42, 0x00,
    0x00, 0x1E, 0x00, 0x00, 0x00, 0xFC, 0x00, 0x49,
    0x46, 0x50, 0x20, 0x44, 0x69, 0x73, 0x70, 0x6C,
    0x61, 0x79, 0x0A, 0x20, 0x00, 0x00, 0x00, 0xFD,
    0x00, 0x3B, 0x46, 0x1F, 0x8C, 0x3C, 0x00, 0x0A,
    0x20, 0x20, 0x20, 0x20, 0x20, 0x20, 0x01, 0xA8,

    0x02, 0x03, 0x39, 0xF2, 0x4D, 0x01, 0x03, 0x12,
    0x13, 0x84, 0x22, 0x1F, 0x90, 0x5D, 0x5E, 0x5F,
    0x60, 0x61, 0x23, 0x09, 0x07, 0x07, 0x83, 0x01,
    0x00, 0x00, 0x6D, 0x03, 0x0C, 0x00, 0x10, 0x00,
    0x00, 0x44, 0x20, 0x00, 0x60, 0x03, 0x02, 0x01,
    0x67, 0xD8, 0x5D, 0xC4, 0x01, 0x78, 0xC0, 0x00,
    0xE3, 0x05, 0x03, 0x01, 0xE4, 0x0F, 0x00, 0x18,
    0x00, 0x02, 0x3A, 0x80, 0x18, 0x71, 0x38, 0x2D,
    0x40, 0x58, 0x2C, 0x45, 0x00, 0xB9, 0xA8, 0x42,
    0x00, 0x00, 0x1E, 0x08, 0xE8, 0x00, 0x30, 0xF2,
    0x70, 0x5A, 0x80, 0xB0, 0x58, 0x8A, 0x00, 0xC4,
    0x8E, 0x21, 0x00, 0x00, 0x1E, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x93,
};
```

推荐一个 EDID 编辑工具：http://www.quantumdata.com/support/downloads/980/release_5_05/R_980mgr_5.05_Win32.msi

6.11 camera3_profiles.xml 配置

走 camera 框架预览需要适配 camera3_profiles.xml，注册 camera 设备。

camera3_profiles.xml 文件对应 SDK 目录下具体芯片平台的文件：

```
hardware/rockchip/camera/etc/camera/camera3_profiles_rk3xxx.xml
```

主要配置注意事项如下，详情可参考 SOC Sensor 的配置方法：

- **name**：需要与驱动名称一致，有大小写区别；
- **moduleId**：需要与驱动 dts 中配置的 index 一致；

```
    </Profiles>
    <Profiles cameraId="0" name="rk628-csi" moduleId="m00">
        <Supported_hardware>
            <hwType value="SUPPORTED_HW_RKISP1"/>
```

```
&rk628_csi {
        status = "okay";
        /*
         * If the hpd output level is inverted on the circuit,
         * the following configuration needs to be enabled.
         */
        /* hpd-output-inverted; */
        plugin-det-gpios = <&gpio0 13 GPIO_ACTIVE_HIGH>;
        power-gpios = <&gpio0 17 GPIO_ACTIVE_HIGH>;
        rockchip,camera-module-index = <0>;
        rockchip,camera-module-facing = "back";
        rockchip,camera-module-name = "RK628-CSI";
        rockchip,camera-module-lens-name = "NC";
```

- scaler.availableStreamConfigurations/scaler.availableMinFrameDurations/ scaler.availableStallDurations：需要正确配置驱动支持的分辨率和最小的帧间隔时间，若驱动中需要增加新的分辨率支持，在这里也要相应地增加配置；

```
<scaler.availableStreamConfigurations value="BLOB,3840x2160,OUTPUT,
    BLOB,1920x1080,OUTPUT,
    BLOB,1280x720,OUTPUT,
    BLOB,720x576,OUTPUT,
    BLOB,720x480,OUTPUT,
    YCbCr_420_888,3840x2160,OUTPUT,
    YCbCr_420_888,1920x1080,OUTPUT,
    YCbCr_420_888,1280x720,OUTPUT,
    YCbCr_420_888,720x576,OUTPUT,
    YCbCr_420_888,720x480,OUTPUT,
    IMPLEMENTATION_DEFINED,3840x2160,OUTPUT,
    IMPLEMENTATION_DEFINED,1920x1080,OUTPUT,
    IMPLEMENTATION_DEFINED,1280x720,OUTPUT,
    IMPLEMENTATION_DEFINED,720x576,OUTPUT,
    IMPLEMENTATION_DEFINED,720x480,OUTPUT" />
<scaler.availableMinFrameDurations value="BLOB,3840x2160,33333333,
    BLOB,1920x1080,16666667,
    BLOB,1280x720,16666667,
    BLOB,720x576,20000000,
    BLOB,720x480,16666667,
    YCbCr_420_888,3840x2160,33333333,
    YCbCr_420_888,1920x1080,16666667,
    YCbCr_420_888,1280x720,16666667,
    YCbCr_420_888,720x576,20000000,
    YCbCr_420_888,720x480,16666667,
    IMPLEMENTATION_DEFINED,3840x2160,33333333,
    IMPLEMENTATION_DEFINED,1920x1080,16666667,
    IMPLEMENTATION_DEFINED,1280x720,16666667,
    IMPLEMENTATION_DEFINED,720x576,20000000,
    IMPLEMENTATION_DEFINED,720x480,16666667"/>
<scaler.availableStallDurations value="BLOB,3840x2160,33333333,
    BLOB,1920x1080,16666667,
    BLOB,1280x720,16666667,
    BLOB,720x576,20000000,
    BLOB,720x480,16666667"/>
```

- **sensor.orientation**：图像旋转角度，支持 0、90、180、270；

```
<sensor.maxAnalogSensitivity value="2400"/> <!-- HAL
<sensor.orientation value="0"/>
<sensor.profileHueSatMapDimensions value="0,0,0"/>
```

6.12 HDMI IN APK 适配方法

6.12.1 安卓 9/10/11 版本

安卓 9/10/11 版本，对应内核版本为 kernel-4.4/kernel-4.19，使用 for-all 代码补丁包，只能 走 cameraHAL3 框架。

6.12.1.1 获取和编译 APK 源码

APK 源码提供在 SDK 目录：

```
RKDocs/common/hdmi-in/apk/rkCamera2_based_on_CameraHal3_V1.3.tar.gz
```

需要将源码拷贝并解压到目录：

```
packages/apps/
```

以 RK3288 平台为例，在 device/rockchip/rk3288/ 目录参照如下修改，增加 rkCamera2 APK 编译：

```
diff --git a/device.mk b/device.mk
index 6667b5c..96f08f1 100644
--- a/device.mk
+++ b/device.mk
@@ -17,7 +17,8 @@
 PRODUCT_PACKAGES += \
     memtrack.$(TARGET_BOARD_PLATFORM) \
     WallpaperPicker \
-    Launcher3
+    Launcher3 \
+    rkCamera2

 #$_rbox_$_modify_$_zhengyang: add displayd
 PRODUCT_PACKAGES += \
```

6.12.1.2 APK 源码的适配

APK 通过 ioctl 的方式访问 RK628 设备节点，获取当前的连接状态和分辨率。RK628 设备节点在 isp1/isp2/vicap 链路上可能会差异。需要根据实际情况修改 APK 源码，获取设备节点的方法可参考调试命令举例章节。

```
rkCamera2/jni/native.cpp
```

```
static void openDevice(JNIEnv *env, jobject thiz)
{
    (void)*env;
    (void)thiz;

    char video_name[64];
    memset(video_name, 0, sizeof(video_name));
    strcat(video_name, "/dev/v4l-subdev2");

    camFd = open(video_name, O_RDWR);
    if (camFd < 0) {
        LOGE("open %s failed,erro=%s",video_name,strerror(errno));
    } else {
        LOGD("open %s success,fd=%d",video_name,camFd);
    }
}
```

获取当前的连接状态和分辨率代码如下：

```
static void getDeviceFormat(int *format)
{
    struct v4l2_control control;
    memset(&control, 0, sizeof(struct v4l2_control));
    control.id = V4L2_CID_DV_RX_POWER_PRESENT;
    int err = ioctl(camFd, VIDIOC_G_CTRL, &control);
    if (err < 0) {
        LOGV("Set POWER_PRESENT failed ,%d(%s)", errno, strerror(errno));
    }

    unsigned int noSignalAndSync = 0;
    ioctl(camFd, VIDIOC_G_INPUT, &noSignalAndSync);
    LOGV("noSignalAndSync ? %s",noSignalAndSync?"YES":"NO");

    struct v4l2_dv_timings dv_timings;
    memset(&dv_timings, 0 ,sizeof(struct v4l2_dv_timings));
    err = ioctl(camFd, VIDIOC_SUBDEV_QUERY_DV_TIMINGS, &dv_timings);
    if (err < 0) {
        LOGV("Set VIDIOC_SUBDEV_QUERY_DV_TIMINGS failed ,%d(%s)", errno, strerror(errno));
    }

    format[0] = dv_timings.bt.width;
    format[1] = dv_timings.bt.height;
    format[2] = control.value && !noSignalAndSync;
}
```

由于在 APK 访问了设备节点，所以需要确认是否关闭了 selinux，可通过 getenforce 命令查看：

```
rk3288:/ # getenforce
Enforcing
rk3288:/ # setenforce 0
rk3288:/ #
rk3288:/ # getenforce
Permissive
```

6.12.1.3 APK 调试前的准备

APK 调试前首先需要先将驱动调试完成，参考驱动调试方法章节。第二步需要确认 camera 设备是否正确注册到 CameraHal，可通过以下命令查看，若未正确注册，则需要检查 camera3_profiles.xml 配置，参考camera3_profiles.xml 配置文件说明章节。

```
rk3288:/ #
rk3288:/ # dumpsys media.camera

== Service global info: ==

Number of camera devices: 1
Number of normal camera devices: 1
    Device 0 maps to "0"
Active Camera Clients:
[]
Allowed user IDs: 0

== Camera service events log (most recent at top): ==
  04-08 11:08:31 : USER_SWITCH previous allowed user IDs: <None>, current allowed user IDs: 0
  01-18 08:50:15 : ADD device 0, reason: (Device status changed from 0 to 1)
  01-18 08:50:15 : ADD device 0, reason: (Device added)
```

6.12.2 安卓 12+ 版本

安卓 12/13 的版本，对应内核版本为 kernel-5.10，驱动补丁使用 kernel-5.10 补丁包。应用框架分为 camera 框架和 TV 框架（后 TIF 同）。具体可以参考文档
《Rockchip_Android12+_HDMI_TO_MIPI_Developer_Guide》进行调试。

安卓14的版本，对应内核版本为kernel-6.1，驱动在SDK目录自带。应用框架推荐使用TV框架。

6.12.2.1 APK 源码

- packages/apps/TV/partner_support/samples ：提供 TV 源数据服务，通过 framework 与 HAL 层、预览 APK 进行交互，由于是开机运行的隐藏服务，该 APK 在桌面上是隐藏图标的。
- hardware/rockchip/tv_input ：TVHAL 层代码，开关流、热拔插和分辨率切换事件等与驱动进行命令交互。
- hardware/rockchip/camera：cameraHAL 层代码，camera 框架取流、拍照等功能实现。
- vendor/rockchip/hardware/interfaces/hdmi：走 camera 框架使用，负责监听分辨率变化与热拔插事件，与驱动以及 APK 交互。
- packages/apps/rkCamera2 ：预览 apk，此应用有 2 个界面。默认的 MainActivity 界面使用 TIF 方案进行预览，通过 framework 层与上述 TV 源数据服务进行交互；RockchipCamera2 界面使用 Camera 方案进行预览，用标准的 Camera API，对 MIPI 的节点的 cameraid 进行 open 操作。
  APK 在桌面上图标名称为 HdmiIn ，通常客户会二次开发替换为自己的预览 APK。
- SDK 默认代码 HDMI IN 功能是关闭的，使能 HDMI IN 功能，需配置如下属性，开启后会编译含上述
  APK 在内的相关模块：

  ```
  vim device/rockchip/rk3588/BoardConfig.mk
  BOARD_HDMI_IN_SUPPORT := true
  ```

- 走 camera 框架需要打开：

  ```
  vim device/rockchip/rk3588/BoardConfig.mk
  CAMERA_SUPPORT_HDMI := true
  ```

只配置 CAMERA_SUPPORT_HDMI，而不配置 BOARD_HDMI_IN_SUPPORT 的情况下。如想要用 rkCamera2 进行 camera 预览，需要将 rkCamera2 加入到编译并配置属性 persist.sys.hdmiinmode 值为 2。
其他补充见文档：

《Rockchip_Developer_Guide_HDMI_RX_CN》

```
PRODUCT_PACKAGES += \
    rkCamera2
```

6.12.2.2 APK 预览说明

APK 支持 RK3588 HDMI RX 通路数据预览和 HDMI 转 MIPI-CSI 通路数据预览，使用时需要切换。

TIF 预览方式，需要设置 MIPI-CSI2 通路：

```
setprop vendor.tvinput.hdmiin.type 1
```

camera 预览方式，需要打开 RockchipCamera2 界面，注意需要使能 CAMERA_SUPPORT_HDMI

```
setprop persist.sys.hdmiinmode 2
```

6.12.2.3 TIF 预览与 camera 预览差异

	TIF	Camera
优点	延迟低	app 端能够拿到预览数据进行后处理
缺点	不支持屏幕旋转、分屏、画中画功能； app 端拿不到预览 buffer 数据； 不支持 screencap 方式的截图命令	延迟高于 TIF

6.13 常驱动调试方法

驱动调试方法与 SOC Sensor 的调试方法一致，更多信息请参考 redmine 文档：

https://redmine.rock-chips.com/documents/53

6.13.1 调试工具获取

需要使用 media-ctl 和 v4l2-ctl 工具，目前 SDK 编译固件时会自动拷贝集成，具体是放置在 SDK 目录：

```
hardware/rockchip/camera/etc/tools/
```

若 SDK 版本较老，可通过 redmine 获取：

https://redmine.rock-chips.com/documents/104

将 media-ctl 和 v4l2-ctl 用 adb push 到设备的 /vendor/bin/ 目录。

6.13.2 调试命令举例

以 RK3288 + RK628 接收 1920x1080P 分辨率为例，具体调试时需要根据实际情况修改。请注意，以下调试命令需要从前往后逐条输入，否则有可能出现缺少配置导致无法工作的问题。

- 查看链路拓扑结构：

执行命令查看 media 节点的拓扑结构，根据不同芯片平台具体的链路连接，有可能是 /dev/media0 或 /dev/media1。

```
media-ctl -d /dev/media0 -p
```

截取 RK628 部分为例，可知 RK628 设备为：/dev/v4l-subdev2，识别到 HDMI IN 分辨率为：1920x1080。

```
...
- entity 8: m00_b_rk628-csi rk628-csi (1 pad, 1 link)
            type V4L2 subdev subtype Sensor
            device node name /dev/v4l-subdev2
        pad0: Source
                [fmt:UYVY2X8/1920x1080]
                -> "rockchip-mipi-dphy-rx":0 [ENABLED]
...
```

- 配置链路连接：

设备复位启动后，链路默认是连接上的。用 HDMI IN APK 打开再退出时，链路会被断开，查看拓扑结构，没有 [ENABLED] 时才需要重新链接。

```
media-ctl -d /dev/media0 -l \
'"m00_b_rk628-csi rk628-csi":0->"rockchip-mipi-dphy-rx":0 [1]'
media-ctl -d /dev/media0 -l \
'"rockchip-mipi-dphy-rx":1->"rkisp1-isp-subdev":0 [1]'
media-ctl -d /dev/media0 -l '"rkisp1-input-params":0->"rkisp1-isp-subdev":1 [1]'
media-ctl -d /dev/media0 -l '"rkisp1-isp-subdev":2->"rkisp1_mainpath":0 [1]'
media-ctl -d /dev/media0 -l '"rkisp1-isp-subdev":2->"rkisp1_selfpath":0 [1]'
media-ctl -d /dev/media0 -l '"rkisp1-isp-subdev":3->"rkisp1-statistics":0 [1]'
```

- 配置分辨率：

```
media-ctl -d /dev/media0 \
--set-v4l2 '"rkisp1-isp-subdev":0[fmt:UYVY2X8/1920x1080]'
media-ctl -d /dev/media0 \
--set-v4l2 '"rkisp1-isp-subdev":0[crop:(0,0)/1920x1080]'
media-ctl -d /dev/media0 \
--set-v4l2 '"rkisp1-isp-subdev":2[fmt:UYVY2X8/1920x1080]'
media-ctl -d /dev/media0 \
--set-v4l2 '"rkisp1-isp-subdev":2[crop:(0,0)/1920x1080]'
```

- 获取图像数据流：

查看配置结果：

```
media-ctl -d /dev/media0 -p
```

获取图像数据流：

```
v4l2-ctl --verbose -d /dev/video0 --set-fmt-
video=width=1920,height=1080,pixelformat='NV12' --stream-mmap=4
```

抓取图像 yuv 文件，可 adb pull 上来用 7yuv 等工具查看：

```
v4l2-ctl -d /dev/video0 --set-fmt-video=width=1920,height=1080,pixelformat='NV12'
--stream-mmap=3 --stream-skip=4 --stream-to=/data/1920x1080p60_nv12.yuv --stream-
count=5 --stream-poll
```

若一切正常，能接收到图像数据，会打出帧率，参考 log 如下：

```
VIDIOC_QUERYCAP: ok
```

```
VIDIOC_G_FMT: ok
VIDIOC_S_FMT: ok
Format Video Capture Multiplanar:
        Width/Height      : 1920/1080
        Pixel Format      : 'NV12'
        Field             : None
        Number of planes  : 1
        Flags             :
        Colorspace        : Default
        Transfer Function : Default
        YCbCr Encoding    : Default
        Quantization      : Full Range
        Plane 0           :
           Bytes per Line : 1920
           Size Image     : 3110400
VIDIOC_G_SELECTION: ok
VIDIOC_S_SELECTION: ok
VIDIOC_REQBUFS: ok
VIDIOC_QUERYBUF: ok
VIDIOC_QUERYBUF: ok
VIDIOC_QBUF: ok
VIDIOC_QUERYBUF: ok
VIDIOC_QBUF: ok
VIDIOC_QUERYBUF: ok
VIDIOC_QBUF: ok
VIDIOC_QUERYBUF: ok
VIDIOC_QBUF: ok
VIDIOC_STREAMON: ok
idx: 0 seq:      0 bytesused: 3110400 ts: 131.560377
idx: 1 seq:      1 bytesused: 3110400 ts: 131.577023 delta: 16.646 ms
idx: 2 seq:      2 bytesused: 3110400 ts: 131.593697 delta: 16.674 ms
idx: 3 seq:      3 bytesused: 3110400 ts: 131.610363 delta: 16.666 ms
idx: 0 seq:      4 bytesused: 3110400 ts: 131.627033 delta: 16.670 ms fps: 60.01
idx: 1 seq:      5 bytesused: 3110400 ts: 131.643721 delta: 16.688 ms fps: 59.99
idx: 2 seq:      6 bytesused: 3110400 ts: 131.660390 delta: 16.669 ms fps: 59.99
idx: 3 seq:      7 bytesused: 3110400 ts: 131.677058 delta: 16.668 ms fps: 59.99
```

6.13.3 打开 log 开关

可通过如下命令打开 RK628 的 log 开关，然后通过 demsg 命令获取 kernel log：

```
echo 1 > /sys/kernel/debug/rk628/x-0050/debug    //x为挂载的i2c总线，50为对应的i2c地址
echo 2 > /sys/module/rk628_csi/parameters/debug
echo 2 > /sys/module/rk628_csi_v4l2/parameters/debug    //内核为linux4.4或者4.19
dmesg -n 8
```

若要抓取上电开机过程的 log，建议直接修改代码并重新编译烧写 kernel 相关部分：

```
diff --git a/drivers/media/i2c/rk628_csi.c b/drivers/media/i2c/rk628_csi.c
index c763a9558169..bd7f3effb45a 100644
--- a/drivers/media/i2c/rk628_csi_v4l2.c
+++ b/drivers/media/i2c/rk628_csi_v4l2.c
@@ -34,7 +34,7 @@
 #include <video/videomode.h>
 #include "rk628_csi.h"
```

```
-static int debug;
+static int debug = 1;
 module_param(debug, int, 0644);
 MODULE_PARM_DESC(debug, "debug level (0-3)");

diff --git a/include/media/v4l2-common.h b/include/media/v4l2-common.h
index 1cc0c5ba16b3..e74f3a85f0b8 100644
--- a/include/media/v4l2-common.h
+++ b/include/media/v4l2-common.h
@@ -75,7 +75,7 @@
 #define v4l2_dbg(level, debug, dev, fmt, arg...)                        \
        do {                                                             \
                if (debug >= (level))                                    \
-                       v4l2_printk(KERN_DEBUG, dev, fmt , ## arg);      \
+                       v4l2_printk(KERN_INFO, dev, fmt , ## arg);       \
        } while (0)
```

6.13.4 寄存器读写

RK628 寄存器调试节点如下，当前示例 RK628 接在 I2C3，地址为 0x50：

```
console:/ # ls /sys/kernel/debug/rk628/3-0050/registers/
adapter    combtxphy  csi    dsi0  gpio0  gpio2  grf     hdmirxphy
combrxphy  cru        csi1   dsi1  gpio1  gpio3  hdmirx
```

寄存器节点可支持读写。

读寄存器：

```
console:/ # cat /sys/kernel/debug/rk628/3-0050/registers/cru
rk628_cru:

0xc0000: 00001031 00000441 00800000 00000007
0xc0010: 00007f00
0xc0020: 00001028 00000441 00f5c28f 00000007
0xc0030: 00007f00
0xc0040: 00005019 00000541 0099735e 00000007
0xc0050: 00007f00
0xc0060: 00000015
0xc0080: 00008989 00000003 00000007 00004b80
0xc0090: 03355460 000080d7 00008004 0000003b
0xc00a0: 00002c00 00000000 00000000 00000000
0xc00b0: 00285f58 00010008 00010008 00000103
0xc00c0: 00000300 00000703 0000000b 00000008
0xc00d0: 0001000a 00000b07
0xc0180: 00000000 00000000 00000000 00000000
0xc0190: 00000000 00000000
0xc0200: 00000000 00000000 00000000 00000000
0xc0210: 00000000
```

写寄存器：

```
rk3288:/ # echo 0x000 0xffffffff > /sys/kernel/debug/rk628/3-0050/registers/grf
```

6.13.5 读取HDMI-RX状态信息

可通过如下指令读取HDMI-RX的信息，包括分辨率信息、锁定状态、拔插状态、颜色格式等等：

```
console:/ # cat /sys/kernel/debug/rk628/3-0050/hdmirx/status
status: plugin
Clk-Ch:Lock     Ch0:Lock        Ch1:Lock        Ch2:Lock
Color Format: RGB
Timing: 1920x1080p60 (2200x1125)                hfp:88  hs:45  hbp:147  vfp:4
vs:5  vbp:36
Pixel Clk: 148500000
Mode: HDMI
Color Depth: 8 bit
Color Range: Limited
Color Space: RGB
```

6.13.6 设置pattern输出

RK628支持设置scaler color bar输出，具体使用命令如下：

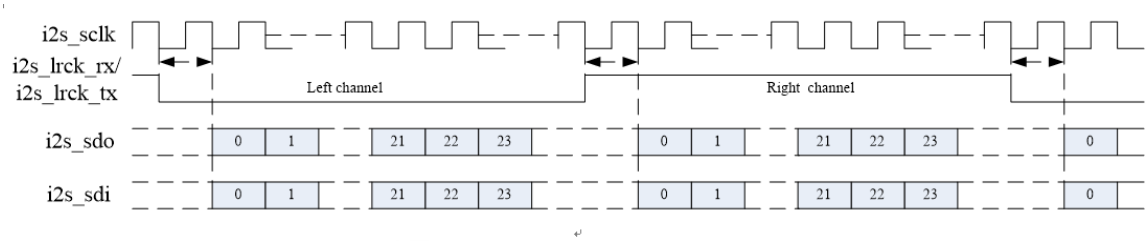
```
echo 1 > /sys/kernel/debug/rk628/3-0050/scaler_color_bar

//pattern 模式选择
echo 2 > /sys/kernel/debug/rk628/3-0050/scaler_color_bar
```

6.14 音频模块介绍

音频部分会根据使用 HDMIRX 或者 HDMITX 的使用场景，可以分别设置为输出与输入。在 HDMIRX 使用场景中，RK628 HDMI 接收的音频数据被解包出来后通过 I2S 接口输出；而 HDMITX 使用场景中，RK628D I2S 是作为数据输入端，接收音频数据，打包后通过 HDMITX 输出。下面是两种场景的配置方式：

6.14.1 HDMIRX

HDMIRX 一般是在驱动加载的时候，就会去初始化音频模块，不再需要额外的配置了。HDMITX 播放音频数据时候，音频信号会通过 RK628 I2S 输出（RK628 必须是 master）。RK628 I2S 可以直接与支持 I2S 接口的 DAC、SOC 连接。I2S 格式如下：



I2S normal mode timing format

与 SOC 连接时候，RK628 I2S 不需要额外的配置，可以使用 dummy_codec 创建一个声卡设备，供系统使用：

```
rk628_dc: rk628-dc {
        compatible = "rockchip,dummy-codec";
        #sound-dai-cells = <0>;
};

hdmiin-sound {
        compatible = "simple-audio-card";
        simple-audio-card,format = "i2s";
        simple-audio-card,name = "rockchip,hdmiin";
        simple-audio-card,bitclock-master = <&dailink0_master>;
        simple-audio-card,frame-master = <&dailink0_master>;
        status = "okay";
        simple-audio-card,cpu {
                        sound-dai = <&i2s0>;
        };
        dailink0_master: simple-audio-card,codec {
                        sound-dai = <&rk628_dc>;
        };
};
```

与 DAC 连接的时候，软件上面不要上面的配置了，DAC 直接就是能输出模拟信号。但是目前大部分的 DAC 都是需要有提供一个 MCLK，RK628 设计的时候，没有预留 MCLK，目前使用一个兼容的方法，使用 RK628 GPIO1_A0 (H7)作为 TEST_CLKO，可以输出 128FS 的时钟。兼容目前大部分的 DAC，有验证过的包括：CS4344、ES7144。使能 GPIO1_A0 输出 MCLK 需要在初始化的时候，调用以下接口，修改如下（请注意自己项目用的是哪个驱动）：

```
--- a/drivers/media/i2c/rk628/rk628_hdmirx.h
+++ b/drivers/media/i2c/rk628/rk628_hdmirx.h
@@ -392,5 +392,5 @@ bool rk628_audio_ctsnints_enabled(HAUDINFO info);
 void rk628_csi_isr_ctsn(HAUDINFO info, u32 pdec_ints);
 void rk628_csi_isr_fifoints(HAUDINFO info, u32 fifo_ints);
 int rk628_is_avi_ready(struct rk628 *rk628, bool avi_rcv_rdy);
+void rk628_hdmirx_audio_set_mclk_output(HAUDINFO info);

--- a/drivers/media/i2c/rk628/rk628_csi.c
+++ b/drivers/media/i2c/rk628/rk628_csi.c
@@ -957,6 +957,9 @@ static void rk628_csi_initial_setup(struct rk628_csi *csi)
 {
        struct v4l2_subdev_edid def_edid;

+       //enable rk628 mclk
+       rk628_hdmirx_audio_set_mclk_output(csi->audio_info);

--- a/drivers/media/i2c/rk628/rk628_csi_v4l2.c
+++ b/drivers/media/i2c/rk628/rk628_csi_v4l2.c
@@ -1096,6 +1096,9 @@ static void rk628_csi_initial_setup(struct v4l2_subdev *sd)
        struct rk628_csi *csi = to_csi(sd);
        struct v4l2_subdev_edid def_edid;

+       //enable rk628 mclk
+       rk628_hdmirx_audio_set_mclk_output(csi->audio_info);
```

对于 RK3288 EVB，使用的是 RK3288 I2S 与 RT5651 的 I2S2 ，RK628 I2S 与 RT5651 的 I2S2 连接。在使用过程中，通过切换 RT5651 内部的 route，达到切换不同通路录音，播放的功能，对应 dts 配置如下：

```
hdmiin-sound {
        compatible = "rockchip,rockchip-rt5651-rk628-sound";
        rockchip,cpu = <&i2s>;
        rockchip,codec = <&rt5651>;
        status = "okay";
};
```

在默认情况下，i2s 只有在预览的时候，才会有输出，如果需要一直输出，DTS 需要添加 i2s-enable-default 配置

```
&i2c4 {
        clock-frequency = <400000>;
        status = "okay";
        rk628_csi_v4l2: rk628_csi_v4l2@50 {
                reg = <0x50>;
                compatible = "rockchip,rk628-csi-v4l2";
                ....
                i2s-enable-default;
                ....
        };
};
```

6.14.2 HDMITX

HDMITX 配置比较简单, 只需要配置好 HDMI 声卡就好了。如下，RK628 I2S 与 SOC 的 I2S0 连接：

```
hdmi_sound: hdmi-sound {
        compatible = "simple-audio-card";
        simple-audio-card,format = "i2s";
        simple-audio-card,name = "hdmi-sound";
        status = "okay";
        simple-audio-card,cpu {
                sound-dai = <&i2s0>;
        };
        simple-audio-card,codec {
                sound-dai = <&rk628_hdmi>;
        };
};
```

6.14.3 音频常见问题处理

6.14.3.1 I2S 没有输出

```
echo 0x000 0x6000220 > /sys/kernel/debug/regmap/0-0050-grf/registers //i2s 的
IOMUX
echo 0x70 0xfffff55c > /sys/kernel/debug/regmap/0-0050-grf/registers
echo 0x70 0x155c155c > /sys/kernel/debug/regmap/0-0050-grf/registers // 设置输出
```

6.14.3.2 使能打印 v4l2_dbg

```
diff --git a/drivers/media/i2c/rk628_csi.c b/drivers/media/i2c/rk628_csi.c
index 5e8e3710a82f..638ac2acc472 100644
--- a/drivers/media/i2c/rk628_csi.c
+++ b/drivers/media/i2c/rk628_csi.c
@@ -34,7 +34,7 @@
 #include <video/videomode.h>
 #include "rk628_csi.h"

-static int debug;
+static int debug = 3;
 module_param(debug, int, 0644);
 MODULE_PARM_DESC(debug, "debug level (0-3)");

diff --git a/include/media/v4l2-common.h b/include/media/v4l2-common.h
index cdc87ec61e54..f159118d4a6b 100644
--- a/include/media/v4l2-common.h
+++ b/include/media/v4l2-common.h
@@ -82,7 +82,7 @@
 #define v4l2_dbg(level, debug, dev, fmt, arg...)                        \
        do {                                                             \
                if (debug >= (level))                                    \
-                       v4l2_printk(KERN_DEBUG, dev, fmt , ## arg);      \
+                       v4l2_printk(KERN_INFO, dev, fmt , ## arg);       \
        } while (0)

 /**
```

6.14.3.3 关于应用录音数据杂音问题

这个可能是 HDMTX 采样率与 HAL 录音采样率不一致导致的

1. 查看 HDMITX 采样率

打开内核 log:`echo 3 > /sys/module/rk628_csi/parameters/debug`

2. 打开之后，从下面 log 看出 TX 的采样率

```
m00_b_rk628-csi rk628-csi: rk628_hdmirx_audio_fs: clkrate:1500 tmdsclk:74250000,
n_decoded:6144, cts_decoded:74250, fs_audio:48000
m00_b_rk628-csi rk628-csi: rk628_csi_delayed_work_audio:
HDMI_RX_AUD_FIFO_FILLSTS1:0xfe, single offset:0, total offset:-2
m00_b_rk628-csi rk628-csi: rk628_csi_delayed_work_audio:
HDMI_RX_AUD_FIFO_ISTS:0x4
m00_b_rk628-csi rk628-csi: rk628_hdmirx_audio_fs: clkrate:1500 tmdsclk:74250000,
n_decoded:6144, cts_decoded:74250, fs_audio:48000
m00_b_rk628-csi rk628-csi: rk628_csi_delayed_work_audio:
HDMI_RX_AUD_FIFO_FILLSTS1:0xfc, single offset:-2, total offset:-4
m00_b_rk628-csi rk628-csi: rk628_csi_delayed_work_audio:
HDMI_RX_AUD_FIFO_ISTS:0x4
m00_b_rk628-csi rk628-csi: rk628_hdmirx_audio_fs: clkrate:1500 tmdsclk:74250000,
n_decoded:6144, cts_decoded:74250, fs_audio:48000
```

```
m00_b_rk628-csi rk628-csi: rk628_csi_delayed_work_audio:
HDMI_RX_AUD_FIFO_FILLSTS1:0xfc, single offset:0, total offset:-4
m00_b_rk628-csi rk628-csi: rk628_csi_delayed_work_audio:
HDMI_RX_AUD_FIFO_ISTS:0x4
m00_b_rk628-csi rk628-csi: rk628_hdmirx_audio_fs: clkrate:1500 tmdsclk:74250000,
n_decoded:6144, cts_decoded:74250, fs_audio:48000
m00_b_rk628-csi rk628-csi: rk628_csi_delayed_work_audio:
HDMI_RX_AUD_FIFO_FILLSTS1:0xfc, single offset:0, total offset:-4
m00_b_rk628-csi rk628-csi: rk628_csi_delayed_work_audio:
HDMI_RX_AUD_FIFO_ISTS:0x4
m00_b_rk628-csi rk628-csi: rk628_hdmirx_audio_fs: clkrate:1500 tmdsclk:74250000,
n_decoded:6144, cts_decoded:74250, fs_audio:48000
m00_b_rk628-csi rk628-csi: rk628_csi_delayed_work_audio:
HDMI_RX_AUD_FIFO_FILLSTS1:0xfa, single offset:-2, total offset:-6
m00_b_rk628-csi rk628-csi: rk628_csi_delayed_work_audio:
HDMI_RX_AUD_FIFO_ISTS:0x4
m00_b_rk628-csi rk628-csi: rk628_hdmirx_audio_fs: clkrate:1500 tmdsclk:74250000,
n_decoded:6144, cts_decoded:74250, fs_audio:48000
m00_b_rk628-csi rk628-csi: rk628_csi_delayed_work_audio:
HDMI_RX_AUD_FIFO_FILLSTS1:0xfc, single offset:2, total offset:-4
m00_b_rk628-csi rk628-csi: rk628_csi_delayed_work_audio:
HDMI_RX_AUD_FIFO_ISTS:0x4
m00_b_rk628-csi rk628-csi: rk628_hdmirx_audio_fs: clkrate:1500 tmdsclk:74250000,
n_decoded:6144, cts_decoded:74250, fs_audio:48000
m00_b_rk628-csi rk628-csi: rk628_csi_delayed_work_audio:
HDMI_RX_AUD_FIFO_FILLSTS1:0xfa, single offset:-2, total offset:-6
m00_b_rk628-csi rk628-csi: rk628_csi_delayed_work_audio:
HDMI_RX_AUD_FIFO_ISTS:0x4
```

3. HAL 录音的采样率：

```
getprop vendor.hdmiin.audiorate
```

6.14.3.4 直接设置 IOMUX

```
 static void rk628_hdmirx_initial_setup(struct rk628_hdmirx *hdmirx)
 {
        struct v4l2_subdev_edid def_edid;
@@ -783,6 +937,8 @@ static void rk628_hdmirx_initial_setup(struct rk628_hdmirx
*hdmirx)

        // ddc and hpd pinctrl
        regmap_write(hdmirx->grf, GRF_GPIO1AB_SEL_CON, 0x07000700);
+       //i2s pinctrl
+       regmap_write(hdmirx->grf, GRF_GPIO0AB_SEL_CON, 0x155c155c);

        rk628_hdmirx_controller_reset(hdmirx);
```

6.14.3.5 关于 tmdsclk 计算错误

1 bitmask 与时钟对齐

```
git sholsh@rk-intel-1:~/rk-sdk/android11-rk3399/kernel$ git show
007131063748ea69facbf4bbe7aaee71c34fd921
commit 007131063748ea69facbf4bbe7aaee71c34fd921
Author: Shunhua Lan <lsh@rock-chips.com>
Date:   Fri Apr 23 18:43:59 2021 +0800

    media: i2c: rk628csi: fix mask for clkrate and fs audio align to 100

    Signed-off-by: Shunhua Lan <lsh@rock-chips.com>
    Change-Id: I15b290319463f1b41e6908e54caa99ef9c6db4f4

diff --git a/drivers/media/i2c/rk628_csi.c b/drivers/media/i2c/rk628_csi.c
index 89c4a926a985..a970bc621e0e 100644
--- a/drivers/media/i2c/rk628_csi.c
+++ b/drivers/media/i2c/rk628_csi.c
@@ -1064,7 +1064,7 @@ static void rk628_csi_delayed_work_audio(struct work_struct
*work)

        /* fout=128*fs=ftmds*N/CTS */
        regmap_read(csi->hdmirx_regmap, HDMI_RX_HDMI_CKM_RESULT, &clkrate);
-       clkrate = clkrate & 0xfff;
+       clkrate = clkrate & 0xffff;
        /* tmdsclk = (clkrate/1000) * 49500000 */
        tmdsclk = clkrate * (49500000 / 1000);
        regmap_read(csi->hdmirx_regmap, HDMI_RX_PDEC_ACR_CTS, &cts_decoded);
@@ -1073,6 +1073,8 @@ static void rk628_csi_delayed_work_audio(struct work_struct
*work)
        if (cts_decoded != 0) {
                fs_audio = div_u64((tmdsclk * n_decoded), cts_decoded);
                fs_audio = div_u64(fs_audio, 128);
+               fs_audio = div_u64(fs_audio + 50, 100);
+               fs_audio *= 100;
        }
        v4l2_dbg(2, debug, sd,
                "%s: clkrate:%d tmdsclk:%llu, n_decoded:%d, cts_decoded:%d,
fs_audio:%llu\n",
```

问题 log 如下：

```
[67.440094] m00_b_rk628-csi rk628-csi:rk628_csi_delayed_work_audio:clkrate:1904
tmdsclk:94248000,n_decoded:6144,cts_decoded:297000,fs_audio:15232
```

4K 屏，计算的 tmdsclk 频率应该是 297M

tmdsclk = clkrate * (49500000 / 1000);

1904 ---> 0x770

6000 ---> 0x1770

tmdsclk 6*49500000 = 297000000

6.14.3.6 6 设置 GPIO 输出 test clk

快捷设置

```
grf: gpio0 相关 iomux 切换，配置为 IO 口:
echo 0x70 0x0fff0000 > registers

gpio0 配置为输出口, 1-0051-rk628-pinctrl:
echo 0xd0008 0x00EC00EC > registers

grf，选择在哪一级拉出信号:
HDMI_RX:
echo 0x300 0x11 > registers
ASYNC_IN:
echo 0x300 0x14 > registers
ASYNC_OUT:
echo 0x300 0x15 > registers
SCALER:
echo 0x300 0x16 > registers
```

6.14.3.7 rk356x 的 IOMUX 特殊处理

对于 RK356X 因为每路 I2S 可以复用多组 pin，需要额外在配置 GRF 寄存器 GRF_IOFUNC_SEL4：

Bit	Attr	Reset Value	Description
14	RW	0x0	i2s3_iomux_sel I2S3 IO mux selection 1'b0:M0 mux solution 1'b1:M1 mux solution
12	RW	0x0	i2s2_iomux_sel I2S2 IO mux selection 1'b0:M0 mux solution 1'b1:M1 mux solution
11:10	RW	0x0	i2s1_iomux_sel I2S1 IO mux selection 2'b00:M0 mux solution 2'b01:M1 mux solution 2'b10:M2 mux solution 2'b11: Reserved

pinctrl 驱动里面，会根据用户选择的 mclk 来配置 GRF 对应的 bit，代码如下：

```
static struct rockchip_mux_route_data rk3568_mux_route_data[] = {
    ....
    RK_MUXROUTE_GRF(1, RK_PA2, 1, 0x0310, WRITE_MASK_VAL(11, 10, 0)), /* I2S1 IO
mux M0 */
    RK_MUXROUTE_GRF(3, RK_PC6, 4, 0x0310, WRITE_MASK_VAL(11, 10, 1)), /* I2S1 IO
mux M1 */
    RK_MUXROUTE_GRF(2, RK_PD0, 5, 0x0310, WRITE_MASK_VAL(11, 10, 2)), /* I2S1 IO
mux M2 */
    RK_MUXROUTE_GRF(2, RK_PC1, 1, 0x0310, WRITE_MASK_VAL(12, 12, 0)), /* I2S2 IO
mux M0 */
    RK_MUXROUTE_GRF(4, RK_PB6, 5, 0x0310, WRITE_MASK_VAL(12, 12, 1)), /* I2S2 IO
mux M1 */
    RK_MUXROUTE_GRF(3, RK_PA4, 4, 0x0310, WRITE_MASK_VAL(14, 14, 0)), /* I2S3 IO
mux M0 */
    RK_MUXROUTE_GRF(4, RK_PC4, 5, 0x0310, WRITE_MASK_VAL(14, 14, 1)), /* I2S3 IO
mux M1 */
    ....
}
```

对于类似 rk628 这类不需要 mclk 应用，因为没有配置 mclk，所以边不会设置对应的 bit，这里需要手动添加下，一般 i2s 都会使用 sclk 的，这里把 sclk 加上去（后续新的 RK356X SDK 会增加此补丁）：

```
diff --git a/drivers/pinctrl/pinctrl-rockchip.c b/drivers/pinctrl/pinctrl-
rockchip.c
index 871c49ef0c2a..3d10e6c2ed27 100644
--- a/drivers/pinctrl/pinctrl-rockchip.c
+++ b/drivers/pinctrl/pinctrl-rockchip.c
@@ -1046,13 +1046,26 @@ static struct rockchip_mux_route_data
rk3568_mux_route_data[] = {
        RK_MUXROUTE_GRF(2, RK_PB0, 3, 0x0310, WRITE_MASK_VAL(9, 8, 0)), /* UART9
IO mux M0 */
        RK_MUXROUTE_GRF(4, RK_PC5, 4, 0x0310, WRITE_MASK_VAL(9, 8, 1)), /* UART9
IO mux M1 */
        RK_MUXROUTE_GRF(4, RK_PA4, 4, 0x0310, WRITE_MASK_VAL(9, 8, 2)), /* UART9
IO mux M2 */
+
        RK_MUXROUTE_GRF(1, RK_PA2, 1, 0x0310, WRITE_MASK_VAL(11, 10, 0)), /* I2S1
IO mux M0 */
        RK_MUXROUTE_GRF(3, RK_PC6, 4, 0x0310, WRITE_MASK_VAL(11, 10, 1)), /* I2S1
IO mux M1 */
        RK_MUXROUTE_GRF(2, RK_PD0, 5, 0x0310, WRITE_MASK_VAL(11, 10, 2)), /* I2S1
IO mux M2 */
+       RK_MUXROUTE_GRF(1, RK_PA3, 1, 0x0310, WRITE_MASK_VAL(11, 10, 0)), /* I2S1
IO mux sclk tx M0 */
+       RK_MUXROUTE_GRF(1, RK_PA4, 1, 0x0310, WRITE_MASK_VAL(11, 10, 0)), /* I2S1
IO mux sclk rx M0 */
+       RK_MUXROUTE_GRF(3, RK_PC7, 4, 0x0310, WRITE_MASK_VAL(11, 10, 1)), /* I2S1
IO mux sclk tx M1 */
+       RK_MUXROUTE_GRF(4, RK_PA6, 5, 0x0310, WRITE_MASK_VAL(11, 10, 1)), /* I2S1
IO mux sclk rx M1 */
+       RK_MUXROUTE_GRF(2, RK_PD1, 5, 0x0310, WRITE_MASK_VAL(11, 10, 2)), /* I2S1
IO mux sclk tx M2 */
+       RK_MUXROUTE_GRF(3, RK_PC3, 5, 0x0310, WRITE_MASK_VAL(11, 10, 2)), /* I2S1
IO mux sclk rx M2 */
        RK_MUXROUTE_GRF(2, RK_PC1, 1, 0x0310, WRITE_MASK_VAL(12, 12, 0)), /* I2S2
IO mux M0 */
```

```
        RK_MUXROUTE_GRF(4, RK_PB6, 5, 0x0310, WRITE_MASK_VAL(12, 12, 1)), /* I2S2
IO mux M1 */
+       RK_MUXROUTE_GRF(2, RK_PC2, 1, 0x0310, WRITE_MASK_VAL(12, 12, 0)), /* I2S2
IO mux sclk tx M0 */
+       RK_MUXROUTE_GRF(2, RK_PB7, 1, 0x0310, WRITE_MASK_VAL(12, 12, 0)), /* I2S2
IO mux sclk rx M0 */
+       RK_MUXROUTE_GRF(4, RK_PB7, 4, 0x0310, WRITE_MASK_VAL(12, 12, 1)), /* I2S1
IO mux sclk tx M1 */
+       RK_MUXROUTE_GRF(4, RK_PC1, 5, 0x0310, WRITE_MASK_VAL(12, 12, 1)), /* I2S1
IO mux sclk rx M1 */
        RK_MUXROUTE_GRF(3, RK_PA2, 4, 0x0310, WRITE_MASK_VAL(14, 14, 0)), /* I2S3
IO mux M0 */
        RK_MUXROUTE_GRF(4, RK_PC2, 5, 0x0310, WRITE_MASK_VAL(14, 14, 1)), /* I2S3
IO mux M1 */
+       RK_MUXROUTE_GRF(3, RK_PA3, 4, 0x0310, WRITE_MASK_VAL(14, 14, 0)), /* I2S3
IO mux sclk M0 */
+       RK_MUXROUTE_GRF(4, RK_PC3, 5, 0x0310, WRITE_MASK_VAL(14, 14, 1)), /* I2S3
IO mux sclk M1 */
        RK_MUXROUTE_GRF(1, RK_PA4, 3, 0x0314, WRITE_MASK_VAL(1, 0, 0)), /* PDM IO
mux M0 */
        RK_MUXROUTE_GRF(1, RK_PA6, 3, 0x0314, WRITE_MASK_VAL(1, 0, 0)), /* PDM IO
mux M0 */
        RK_MUXROUTE_GRF(3, RK_PD6, 5, 0x0314, WRITE_MASK_VAL(1, 0, 1)), /* PDM IO
mux M1 */
```

6.14.3.8 RK3399 的 LRCK 的特殊处理

如果方向配置错误可能导致声卡打开后收不到任何数据，LOG 大概如下（节点路径需要根据实际情况修改）：

```
console:/ # cat /proc/asound/cards
 0 [rockchiphdmi   ]: rockchip_hdmi - rockchip,hdmi
                      rockchip,hdmi
 1 [rockchiphdmiin ]: rockchip_hdmiin - rockchip,hdmiin
                      rockchip,hdmiin

console:/ # cat /proc/asound/card1/pcm0c/sub0/status
state: RUNNING
owner_pid   : 1415
trigger_time: 1632716135.151992694
tstamp      : 0.000000000
delay       : 0
avail       : 0
avail_max   : 0
-----
hw_ptr      : 0
appl_ptr    : 0
```

正常情况应该类似下面的信息：

```
console:/ # cat /proc/asound/card1/pcm0c/sub0/status
state: RUNNING
owner_pid   : 1421
trigger_time: 1632892868.412331044
tstamp      : 1632892881.181113577
delay       : 232
avail       : 232
avail_max   : 264
-----
hw_ptr      : 543736
appl_ptr    : 543504
```

这是因为 RK628 做主，RK3399 当从设备，需要根据电路连接方式进行配置，如果 LRCK_TX 连接 RK628，则“rockchip,clk-trcm”配置如下：

```
+&i2s0 {
+       rockchip,clk-trcm = <1>;
+       status = "okay";
+};

console:/ # io -4 0xff880008
ff880008:  18071f1f
```

如果 LRCK_RX 连接 RK628，则“rockchip,clk-trcm”配置如下：

```
+&i2s0 {
+       rockchip,clk-trcm = <2>;
+       status = "okay";
+};
```

正常寄存器如下：

```
console:/ # io -4 0xff880008
ff880008:  28071f1f
```

I2S_CKR 寄存器解释如下 bit29:28 解释如下：

```
Tx and Rx Common Use
2'b00/2'b11:tx_lrck/rx_lrck are used as synchronous signal for TX
/RX respectively.
2'b01:only tx_lrck is used as synchronous signal for TX and RX.
2'b10:only rx_lrck is used as synchronous signal for TX and RX.
```

RK628 代码可以更新到 v15-210926 或更新，然后参考这个 DTS 的音频配置 arch/arm64/boot/dts/rockchip/rk3399-evb-ind-lpddr4-rk628-hdmi2csi-v4l2-avb.dts

6.14.3.9 HDMI-IN 声卡选择错误

APK 打开时会去根据声卡名称去匹配，早期代码可能存在名称匹配错误问题，从而导致 HDMI-IN 声卡无法打开的问题，错误信息如下：

```
09-28 07:03:13.341   261  1386 D AudioHardwareTiny: card0 id:rockchiphdmi
09-28 07:03:13.341   261  1386 D AudioHardwareTiny: SPEAKER card, got
card=0,device=0
```

```
09-28 07:03:13.341   261  1386 D AudioHardwareTiny: HDMI card, got
card=0,device=0
09-28 07:03:13.341   261  1386 D AudioHardwareTiny: SPDIF card, got
card=0,device=0
09-28 07:03:13.341   261  1386 D AudioHardwareTiny: BT card, got card=0,device=0
09-28 07:03:13.342   261  1386 D AudioHardwareTiny: card1 id:rockchiphdmiin
09-28 07:03:13.342   261  1386 D AudioHardwareTiny: SPEAKER card, got
card=1,device=0
09-28 07:03:13.342   261  1386 D AudioHardwareTiny: HDMI card, got
card=1,device=0
09-28 07:03:13.342   261  1386 D AudioHardwareTiny: SPDIF card, got
card=1,device=0
09-28 07:03:13.342   261  1386 D AudioHardwareTiny: BT card, got card=1,device=0
09-28 07:03:13.342   261  1386 D AudioHardwareTiny: No exist
proc/asound/card2/id, break and finish parsing
09-28 07:03:13.343   261  1386 D AudioHardwareTiny: dump out device info
09-28 07:03:13.343   261  1386 D AudioHardwareTiny: dev_info SPEAKER  card=1,
device:0
09-28 07:03:13.343   261  1386 D AudioHardwareTiny: dev_info HDMI  card=1,
device:0
09-28 07:03:13.343   261  1386 D AudioHardwareTiny: dev_info SPDIF  card=1,
device:0
09-28 07:03:13.343   261  1386 D AudioHardwareTiny: dev_info BT  card=1, device:0
```

HDMI card, got card=1,device=0 信息与实际不符，因此声卡无法打开。

需要更新以下补丁解决或者将 RK628 代码更新到 V15-210926 或更新版本解决。

```
hardware/rockchip/audio
commit 1b51a62fc62e9d63850e4e5a39f1e3cff0aa9b88 (HEAD)
Author: Shunhua Lan <lsh@rock-chips.com>
Date:   Tue Sep 28 17:25:44 2021 +0800

    [audio hal] use levenshtein distance for sound card matching

    Signed-off-by: Shunhua Lan <lsh@rock-chips.com>
    Change-Id: I16f1692715d710a0693ae74875b0272669a04ba2
```

6.14.4 其他音频文档补充

如果 RK628 直接连 SOC，可以参考文档“RK628-DIRECT-TO-SOC.pdf”。

如果 RK628 通过 AUDIO CODEC 再连 SOC，可以参考文档“RK628-RT5640-AUDIO-CONFIG.pdf”。

和 HDMI-IN APK 相关的音频问题，可以参考文档“RK628-HDMIIN-APP-AUDIO.pdf”。

若其他文档与这个文档有不符的地方或者冲突，以这个文档为准。

6.15 常见问题排查方法

6.15.1 clk det 异常问题

6.15.1.1 RK628D

COMBRXPHY 没有检测到信号，有可能是 HDMI 插入有效电平或是 HPD 有效电平配置出错，导致输入源 HDMI 信号没有正常输入。需要检查 rk628_csi 节点下 plugin-det-gpios 和 hpd-output-inverted 配置项，同时可以用万用表测试 HPD 电平状态。

注意异常时会进行 retry，最多到 retry 2。若 retry 后能够正常，则可认为信号正常。

```
rk628-csi-v4l2 1-0051: clk det over cnt:10, reg_0x6654:0x403f0000
rk628-csi-v4l2 1-0051: |d2_p| level detection anomaly
rk628-csi-v4l2 1-0051: clock detected failed, cfg resistance manual!
rk628-csi-v4l2 1-0051: err, clk not stable, reg_0x6630:0x87000d,
reg_0x6608:0x110100
m00_b_rk628-csi 1-0051: hdmi rxphy power on failed
```

如上 log，可以看出是 data2_p 电压检测异常，可能是没信号或是电平幅值比较低。

```
rk628-csi-v4l2 1-0051: clk det over cnt:10, reg_0x6654:0x403f0000
rk628-csi-v4l2 1-0051: Clock detection anomaly
rk628-csi-v4l2 1-0051: clock detected failed, cfg resistance manual!
rk628-csi-v4l2 1-0051: err, clk not stable, reg_0x6630:0x87000d,
reg_0x6608:0x110100
m00_b_rk628-csi 1-0051: hdmi rxphy power on failed
```

如上 log，表示频点检测异常，可能是频点不在 RK628D 的支持范围内，目前 HDMIRX 支持以下频点：

```
25175, 27000, 33750, 40000, 59400, 65000, 68250, 74250,
75000, 83500, 85500, 88750, 928125, 101000, 102250, 108000,
118800, 119000, 135000, 148500, 150000, 162000, 165000, 297000
```

6.15.2 HDMI RX 正常的判断方法

COMBRXPHY 正常锁定后，HDMI RX 控制器能正常解析到 Timing，Timing 是计算出来，可能会有一些小误差，一般可能会差 1。详细的 Timing 可参考 CEA 标准文档。

```
m00_b_rk628-csi rk628-csi: cnt_num:1000, tmds_cnt:3000, hs_cnt:15, vs_cnt:3667, hofs:192
m00_b_rk628-csi rk628-csi: SCDC_REGS1:0xffff0f00, act:1920x1080, total:2200x1125, fps:60,

m00_b_rk628-csi rk628-csi: hfp:88, hs:45, hbp:147, vfp:4, vs:5, vbp:36, interlace:0
m00_b_rk628-csi rk628-csi: rk628_csi_s_dv_timings: 1920x1080p60.0 (2200x1125)
m00_b_rk628-csi rk628-csi: enable_stream: disable
```

6.15.3 Open subdev 权限异常

```
D JNI     : JNI CAMERA CALL init
I HdmiInput-navtive: JNI OnLoad
I HdmiInput-navtive: Apk Version: V1.2
E HdmiInput-navtive: openDevice(91): open /dev/v4l-subdev2 failed,erro=Permission denied
D RockchipCamera2: remove take pic button
D RockchipCamera2: recreatTextureview
I RockchipCamera2: textureView remove
D RockchipCamera2: onResume
```

需要确认是否已经给 /dev/v4l-subdev* 提供 666 的权限，在设备命令行查询：

```
rk3288:/ # cat /vendor/ueventd.rc | grep subdev
/dev/v4l-subdev*          0666   media      camera
```

目前 SDK 代码已经包含了本提交，若未包含，可在 device/rockchip/common/ 参考如下修改：

```
wendingxian@ubuntu:~/rk3288_9.0_int/device/rockchip/common$ git show
commit 17112e1430b0f10a88b57c73ad19203e58f0eeff (HEAD -> mid_9.0, rk/rk/rk33/mid/9.0/develop)
Author: Dingxian Wen <shawn.wen@rock-chips.com>
Date:   Tue Apr 27 21:26:48 2021 +0800

    ueventd.rockchip.rc: modify the /dev/v4l-subdev* permission to 666

    Signed-off-by: Dingxian Wen <shawn.wen@rock-chips.com>
    Change-Id: Id4848209fd983f7e525761f19c7f0420b9fee747

diff --git a/ueventd.rockchip.rc b/ueventd.rockchip.rc
index 1914490b..52935d49 100755
--- a/ueventd.rockchip.rc
+++ b/ueventd.rockchip.rc
@@ -178,7 +178,7 @@
 /dev/ovr0               0664   system          system

 #for rk_isp1
-/dev/v4l-subdev*          0660   media      camera
+/dev/v4l-subdev*          0666   media      camera

 /dev/video0          0660   media      camera
 /dev/video1          0660   media      camera
```

6.15.4 信号识别不到

步骤 1. 拔插 HDMI，串口没有 RK628 相关的 log 打印，说明 det 脚的 gpio 配置错误。

步骤 2. 拔出 HDMI 有 RK628 的 log 打印，而插入没有 RK628 的 log 打印，说明 plugin-det-gpios 脚极性设置反了，如果中间加了三极管反向那么要设置 GPIO_ACTIVE_LOW。 判断拔插状态是否吻合，可以把 rk628_csi_v4l2.c 里面，static int debug; 改成 static int debug =3;
插入的时候会打印 tx_5v_power_present: 1 ，拔出的时候 tx_5v_power_present: 0 就算对了

步骤 3. 插入一直检测不到正常信号，可能有下面几种情况：

情况 1. hpd 脚反向问题，RK628 给 hdmiin 座子那边要有高电平，hdmi 才会有输出，如果加了反向器，那么 RK628 管脚需要输出低，反向之后到座子那边才会变高。那么 dts 要加上 hpd-output-inverted;

情况 2. phy 锁不住，请参考clk det 异常问题；

情况 3. 分辨率锁不住，可能是 HDMIRX 检测到码率超出规定范围；

情况 4. 分辨率能锁住。avi_rcv_rdy 在信号锁住后会设置为 1，但是如果报 avi 信号没有 ready，这种可能是中断脚没有配置。例如以下 LOG 一直打印 avi_rcv_rdy:0，表示能识别到正常分辨率，但是 avi_rcv_rdy 为 0，信号没锁住，优先检查下中断脚配置是否正确。

```
rk628d 1-0050: rk628_is_avi_ready PDEC_AVI_PB:0x10000840, avi_rcv_rdy:0
rk628d 1-0050: SCDC_REGS1:0x80000f00, act:1920x1080, total:2200x1125, fps:60,
pixclk:148500000
```

这种情况下去抓图，测试过 rk3288 抓图会概率性出现图像分割界面

6.15.5 显示异常

1. 确认输入源是否是 DVI Mode， RK628D 不支持 DVI，可以查看 log:

```
rk628-csi-v4l2 1-0051: DVI mode detected
```

2. 确认输入源是否有 HDCP 加密，如果有加密，需要打开 RK628 HDMIRX 的 HDCP 功能。

6.15.6 显示只有一半画面

出现这个问题，最大的可能是测试场景 HDMI 的源输出是 4K60，但是对接的主控是旧的平台，如 RK3399、RK3568、RK3288 等只能支持单 MIPI 的 SOC，这样 RK628F/H 会将 4K60 拆分成左右两半图像，因此主控收到的只有一半的图像。这种场景下，建议直接将 edid 修改为支持到 4K30，即直接使用 RK628D 的 edid 即可：

```
diff --git a/drivers/media/i2c/rk628/rk628_csi_v4l2.c
b/drivers/media/i2c/rk628/rk628_csi_v4l2.c
index 32182d4433605..36568f5a081e0 100644
--- a/drivers/media/i2c/rk628/rk628_csi_v4l2.c
+++ b/drivers/media/i2c/rk628/rk628_csi_v4l2.c
@@ -1347,7 +1347,7 @@ static void rk628_csi_initial_setup(struct v4l2_subdev *sd)
        def_edid.start_block = 0;
        def_edid.blocks = 2;
        if (csi->rk628->version >= RK628F_VERSION)
-               def_edid.edid = rk628f_edid_init_data;
+               def_edid.edid = edid_init_data;
        else
                def_edid.edid = edid_init_data;
        rk628_csi_s_edid(sd, &def_edid);
```

6.15.7 抓图失败

1. Android9.0 之后及 linux 平台

   RK 平台 Android9.0 之后或者 linux 可以通过 v4l2-ctl 抓图。
   先看`media-ctl -d /dev/mediaX -p`, media 节点编号根据各个平台不同而不同。如果不清楚，一般情况把 cif 控制器关掉，会是 media0，rk3399 如果接 phy1，可能就是 media1 等等,请了解平台 camera 配置.
   如果 media-ctl 拓扑下面有 rk628 的节点，如果没有应该就是 dts 配置有问题，自行检查，另外拓扑结构所有分辨率并不和你输入的分辨率匹配，那么也得通过 media-ctl 设置成一致。

   然后执行

```
v4l2-ctl -d /dev/video0 --set-fmt-
video=width=1920,height=1080,pixelformat=NV12 --stream-mmap=3`
```

   如果一直打印<<<<<<<<<< 说明抓图正常，这种情况驱动一般已经正常。
   如果一直报错，或者概率报错：

```
rkisp1: MIPI mis error:
```

一般可能跟硬件相关，检查硬件 hdmi 线有无分叉，阻抗匹配是否一致，另外比较多客户喜欢 hdmi 线上接电阻（默认原理图上不接），这种我们默认非必要情况下阻值为 0，如果不为 0，可能概率性出现报错情况。要具体分析，检查下 rk628 到主控这端的信号；

或者出现

```
rkisp-vir0: MIPI error: overflow: 0x00000001
```

这个 log 是 rk356x rv11xx 容易在 isp 使用 4k 输入的时候出现的问题，这种情况建议走 rk628+vicap 链路，请搜索本文档 vicap 配置方式说明

另外打印如下这种 SIZE_ERROR，有可能是输入 720p，却用 1080p 采集，这种分辨率不匹配的情况，要检查拓扑结构所有分辨率，以及 v4l2-ctl 采集分辨率是否对的问题，当然建议是刚开始先都用 1080p 去测试

```
rkisp1 ff910000.rkisp1: CIF_ISP_PIC_SIZE_ERROR
```

另外系统默认用的是 /dev/v4l-subdev2 去判断分辨率，但是如果用`media-ctl -p`看的拓扑结构为 /dev/v4l-subdev3，如下：

```
 - entity 70: m00_b_rk628-csi rk628-csi (1 pad, 1 link)
          type V4L2 subdev subtype Sensor
          device node name /dev/v4l-subdev3
pad0: Source
    [fmt:UYVY2X8/1280x720]
    -> "rockchip-csi2-dphy0":0 []
```

这样就会识别不到分辨率，应用在判断分辨率会识别不到，也会打印 CIF_ISP_PIC_SIZE_ERROR，为了解决这个问题，需要将 android 的应用 jni/native.cpp 文件做如下修改：

```
strcat(video_name, "/dev/v4l-subdev2");
```

改成

```
strcat(video_name, "/dev/v4l-subdev3");
```

2. 4K 分辨率不收图

4K 因为频率更高，也有可能是硬件信号质量不好的缘故，另外 3399 isp 需要超频，参考RK3399 配置 isp 超频的方法

6.15.8 APK 打开失败

发布的补丁会自带 rkCamera2 源码，请在系统里面编译(可以直接放在根目录下，然后 mmm apk 目录)。应用闪退可能是以下原因导致的：

1. 依赖库问题

   如果只将 apk 安装到系统，那么会报

```
AndroidRuntime: java.lang.UnsatisfiedLinkError: dlopen failed: library
"/system/lib64/libhdmiinput_jni.so" needed or dlopened by
"/apex/com.android.runtime/lib64/libnativeloader.so" is not accessible for
the namespace "classloader-namespace"
```

这种情况建议直接在 SDK 编译并打包，依赖库会打包到 img 里面。

2. 文件权限问题

   APK 通过 ioctl 的方式访问 RK628 设备节点，比如：

```
HdmiInput-navtive: openDevice(113): open /dev/v4l-subdev2
failed,erro=Permission denied
```

   首先确定文件权限，补丁有设置成 666，由于在 APK 访问了设备节点，所以需要确认是否关闭了 selinux，可通过 getenforce 命令查看。

3. 上层配置问题

   Android 9.0 之后配置 camera3_profiles.xml，Android 9.0 之前配置 cam_board.xml，可以通过 dumpsys media.camera 查看，如果没有，可能就是没有配置：

```
$ dumpsys media.camera

== Service global info: ==

Number of camera devices: 1
Number of normal camera devices: 1
```

4. camera 库 crash

   例如出现以下错误，可能是 camera 名称错误导致，因为 getDataFromXmlFile 获取到的名字和驱动的不吻合，导致 CRASH。

```
09-29 04:43:30.673   405   405 W ServiceManagement: Waited one second for
android.hardware.camera.provider@2.4::ICameraProvider/legacy/0. Waiting
another...
09-29 04:43:31.673   405   405 W ServiceManagement: Waited one second for
android.hardware.camera.provider@2.4::ICameraProvider/legacy/0. Waiting
another...
...
09-29 04:43:31.904  1404  1404 F DEBUG   : backtrace:
09-29 04:43:31.904  1404  1404 F DEBUG   :     #00 pc 0007214e
/vendor/lib/hw/camera.rk30board.so
(android::camera2::ChromeCameraProfiles::handleAndroidStaticMetadata(char const*,
char const**)+546)
09-29 04:43:31.904  1404  1404 F DEBUG   :     #01 pc 00007895  /system/lib/vndk-
28/libexpat.so (doContent+432)
09-29 04:43:31.905  1404  1404 F DEBUG   :     #02 pc 0000637b  /system/lib/vndk-
28/libexpat.so (contentProcessor+40)
09-29 04:43:31.905  1404  1404 F DEBUG   :     #03 pc 00003825  /system/lib/vndk-
28/libexpat.so (XML_ParseBuffer+84)
09-29 04:43:31.905  1404  1404 F DEBUG   :     #04 pc 000711ad
/vendor/lib/hw/camera.rk30board.so
(android::camera2::CameraProfiles::getDataFromXmlFile()+148)
09-29 04:43:31.905  1404  1404 F DEBUG   :     #05 pc 00071e97
/vendor/lib/hw/camera.rk30board.so
(android::camera2::ChromeCameraProfiles::init()+86)
09-29 04:43:31.905  1404  1404 F DEBUG   :     #06 pc 000734c7
/vendor/lib/hw/camera.rk30board.so (android::camera2::PlatformData::init()+198)
09-29 04:43:31.905  1404  1404 F DEBUG   :     #07 pc 000cecb3
/vendor/lib/hw/camera.rk30board.so (initCameraHAL()+46)
...
```

对于需要 v4l2（android9.0 及以上）的需要将 camera3_profiles_rk3399.xml 里面的 name="rk628-csi" 改名成 name="rk628-csi-v4l2"，修改如下：

```
hardware/rockchip/camera
--- a/etc/camera/camera3_profiles_rk3399.xml
+++ b/etc/camera/camera3_profiles_rk3399.xml
@@ -769,7 +769,7 @@

 <!-- ******************PSL specific section end
************************************************************-->
     </Profiles>
-    <Profiles cameraId="0" name="rk628-csi" moduleId="m00">
+    <Profiles cameraId="0" name="rk628-csi-v4l2" moduleId="m00">
```

5. APK 权限导致的闪退

   出现该问题的 LOG 大概如下：

   对于需要 v4l2（android9.0 及以上）的需要将"rk628-csi"改名成"rk628-csi-v4l2"

```
09-24 10:21:20.154  1552  1570 E AndroidRuntime: FATAL EXCEPTION: Thread-2
09-24 10:21:20.154  1552  1570 E AndroidRuntime: Process:
com.android.rockchip.camera2, PID: 1552
09-24 10:21:20.154  1552  1570 E AndroidRuntime: java.lang.IllegalStateException:
startRecording() called on an uninitialized AudioRecord.
09-24 10:21:20.154  1552  1570 E AndroidRuntime:        at
android.media.AudioRecord.startRecording(AudioRecord.java:983)
09-24 10:21:20.154  1552  1570 E AndroidRuntime:        at
com.android.rockchip.camera2.AudioStream$recordSound.run(AudioStream.java:199)
09-24 10:21:20.154  1552  1570 E AndroidRuntime:        at
java.lang.Thread.run(Thread.java:764)
09-24 10:21:20.164   464   820 W ActivityManager:   Force finishing activity
com.android.rockchip.camera2/.RockchipCamera2
09-24 10:21:20.183  1552  1552 D RockchipCamera2: onPause
```

可以采用以下补丁解决

```
packages/apps/rkCamera2
--- a/AndroidManifest.xml
+++ b/AndroidManifest.xml
@@ -4,7 +4,8 @@
     package="com.android.rockchip.camera2">

         <uses-permission android:name="android.permission.CAMERA" />
-        <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
+       <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"
/>
+       <uses-permission android:name="android.permission.RECORD_AUDIO" />
     <application
         android:allowBackup="true"
         android:icon="@mipmap/ic_launcher"
diff --git a/src/com/android/rockchip/camera2/RockchipCamera2.java
b/src/com/android/rockchip/camera2/RockchipCamera2.java
index 9dc29b8..02ee104 100755
--- a/src/com/android/rockchip/camera2/RockchipCamera2.java
+++ b/src/com/android/rockchip/camera2/RockchipCamera2.java
@@ -100,9 +100,11 @@ public class RockchipCamera2 extends Activity {
        if (ActivityCompat.checkSelfPermission(this,
            Manifest.permission.CAMERA) != PackageManager.PERMISSION_GRANTED
            && ActivityCompat.checkSelfPermission(this,
-                   Manifest.permission.WRITE_EXTERNAL_STORAGE) !=
PackageManager.PERMISSION_GRANTED) {
+                   Manifest.permission.WRITE_EXTERNAL_STORAGE) !=
PackageManager.PERMISSION_GRANTED
+           && ActivityCompat.checkSelfPermission(this,
+                   Manifest.permission.RECORD_AUDIO) !=
PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(RockchipCamera2.this,
-               new String[] { Manifest.permission.CAMERA,
Manifest.permission.WRITE_EXTERNAL_STORAGE },
+               new String[] { Manifest.permission.CAMERA,
Manifest.permission.WRITE_EXTERNAL_STORAGE, Manifest.permission.RECORD_AUDIO},
                    REQUEST_CAMERA_PERMISSION);
            return;
        }
```

6.15.9 dts 配置连接到 rkcif，apk 预览失败

RK356X，android11 代码版本是 R10/R11，dts 配置 RK628 连接到 rkcif，抓图正常，apk 打开预览失败，可能是 R10 和 R11 的代码 cameraHAL 部分不支持该 pipeline 链路造成的，R9 没有这个问题。排查方法和解决方法如下：

1. 打开 cameraHAL 开关：

```
setprop persist.vendor.camera.hal.debug 5
```

2. 抓取打开 apk 出错的 logcat，log 关键信息如下：

```
I RkCamera: <HAL> RKISP2GraphConfig: @addLinkParams, srcName:rockchip-csi2-dphy0, srcPad:1, sinkName:none, sinkPad:0, enable:1, flags:1
I RkCamera: <HAL> RKISP2GraphConfig: @addLinkParams, srcName:none, srcPad:1, sinkName:none, sinkPad:0, enable:1, flags:1
I RkCamera: <HAL> RKISP2GraphConfig: @addLinkParams, srcName:none, srcPad:2, sinkName:rkisp_rawwr0, sinkPad:0, enable:1, flags:1
I RkCamera: <HAL> RKISP2GraphConfig: @addLinkParams, srcName:none, srcPad:4, sinkName:rkisp_rawwr2, sinkPad:0, enable:1, flags:1
I RkCamera: <HAL> RKISP2GraphConfig: @addLinkParams, srcName:none, srcPad:5, sinkName:rkisp_rawwr3, sinkPad:0, enable:1, flags:1
I RkCamera: <HAL> RKISP2GraphConfig: @addFormatParams, entityName:none, width:1920, height:1080, pad:0, format:0x2006:V4L2_MBUS_FMT_UYVY8_2X8, field:0
I RkCamera: <HAL> RKISP2GraphConfig: @addSelectionParams, width:1920, height:1080, left:0, top:0, target:0, pad:0, entityName:none
I RkCamera: <HAL> RKISP2GraphConfig: @addSelectionParams, width:1920, height:1080, left:0, top:0, target:0, pad:2, entityName:none
I RkCamera: <HAL> RKISP2GraphConfig: @addFormatParams, entityName:none, width:1920, height:1080, pad:2, format:0x2008:V4L2_MBUS_FMT_YUYV8_2X8, field:0
I RkCamera: <HAL> RKISP2GraphConfig: @addLinkParams, srcName:none, srcPad:3, sinkName:none, sinkPad:0, enable:1, flags:1
I RkCamera: <HAL> RKISP2GraphConfig: @addLinkParams, srcName:none, srcPad:0, sinkName:none, sinkPad:1, enable:1, flags:1
D RkCamera: <HAL> RKISP2GraphConfig: @ isNeedPathCrop : stream ratios: 1.777778
D RkCamera: <HAL> RKISP2GraphConfig: @ isNeedPathCrop : mp_need_crop 1, sp_need_crop 0, sp_enabled 0
D RkCamera: <HAL> RKISP2GraphConfig: @cal_crop : src_ratio:1.777778, dst_ratio:1.777778, src(1920x1080), dst(1920x1080)
I RkCamera: <HAL> RKISP2GraphConfig: @addSelectionVideoParams, width:1920, height:1080, left:0, top:0, target:0, type:1, flags:0 entityName:none
I RkCamera: <HAL> RKISP2GraphConfig: @addFormatParams, entityName:none, width:1920, height:1080, pad:0, format:0x3231564e:V4L2_PIX_FMT_NV12, field:0
I RkCamera: <HAL> RKISP2GraphConfig: @addLinkParams, srcName:none, srcPad:2, sinkName:none, sinkPad:0, enable:1, flags:1
I RkCamera: <HAL> RKISP2GraphConfig: @getImguMediaCtlConfig : No need for selfPath
```

如果看到上述 log，则是 ameraHAL 默认将对应的 pipeline 配置到 rkisp 了，而 dts 对应的是配置到 rkcif

3. 解决办法

   hardware/rockchip/camera 目录下，添加如下修改：

```
diff --git a/psl/rkisp2/RKISP2GraphConfig.cpp b/psl/rkisp2/RKISP2GraphConfig.cpp
index 2a5fa5a..3d2409c 100755
--- a/psl/rkisp2/RKISP2GraphConfig.cpp
+++ b/psl/rkisp2/RKISP2GraphConfig.cpp
@@ -76,6 +76,7 @@ const string MEDIACTL_POSTVIEWNAME = "postview";

 const string MEDIACTL_STATNAME = "rkisp1-statistics";
 const string MEDIACTL_VIDEONAME_CIF = "stream_cif_dvp_id0";
+const string MEDIACTL_VIDEONAME_CIF_MIPI_ID0 = "stream_cif_mipi_id0";

 RKISP2GraphConfig::RKISP2GraphConfig() :
         mManager(nullptr),
@@ -2620,6 +2621,13 @@ status_t RKISP2GraphConfig::getImguMediaCtlConfig(int32_t
cameraId,
                 addLinkParams("rkisp-isp-subdev", 2, "rkisp_mainpath", 0, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
                 addLinkParams("rkisp-isp-subdev", 2, "rkisp_selfpath", 0, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
             }
+        } else if(mipName2.find("mipi") != std::string::npos) {
+            addLinkParams(mipName, mipSrcPad, mipName2, csiSinkPad, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
+            addLinkParams(mipName2, 1, "stream_cif_mipi_id0", 0, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
+            addLinkParams(mipName2, 2, "stream_cif_mipi_id1", 0, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
+            addLinkParams(mipName2, 3, "stream_cif_mipi_id2", 0, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
+            addLinkParams(mipName2, 4, "stream_cif_mipi_id3", 0, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
```

```
+            mSensorLinkedToCIF = true;
         } else {
             addLinkParams(mipName, mipSrcPad, csiName, csiSinkPad, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
             addLinkParams(csiName, csiSrcPad, IspName, ispSinkPad, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
@@ -2628,6 +2636,12 @@ status_t RKISP2GraphConfig::getImguMediaCtlConfig(int32_t
cameraId,
             addLinkParams(csiName, 5, "rkisp_rawwr3", 0, 1,
MEDIA_LNK_FL_ENABLED, mediaCtlConfig);
         }
     }
+    if(mSensorLinkedToCIF){
+        addImguVideoNode(IMGU_NODE_VIDEO, MEDIACTL_VIDEONAME_CIF_MIPI_ID0,
mediaCtlConfig);
+        addFormatParams(MEDIACTL_VIDEONAME_CIF_MIPI_ID0, mCurSensorFormat.width,
mCurSensorFormat.height,
+                0, V4L2_PIX_FMT_NV12, 0, 0, mediaCtlConfig);
+        return OK;
+    }
     // isp input pad format and selection config
     addFormatParams(IspName, ispInWidth, ispInHeight, ispSinkPad, ispInFormat,
0, 0, mediaCtlConfig);
     addSelectionParams(IspName, ispInWidth, ispInHeight, 0, 0,
V4L2_SEL_TGT_CROP, ispSinkPad, mediaCtlConfig);
```

6.15.10 如何操作 RK628 的 GPIO

在需要控制 GPIO 的位置直接调用 GPIO 接口即可，该接口包括 IOMUX 设置，GPIO 方向设置和电平设置，例如以下操作将 I2C_SDA_HDMI 和 I2C_SCL_HDMI 接口设为 GPIO。

```
rk628_gpio_direction_output(rk628, GPIO1_B1, GPIO_REG_HIGH);
rk628_gpio_direction_output(rk628, GPIO1_B2, GPIO_REG_HIGH);
```

若需要恢复则需要调用以下接口。

```
rk628_pinctrl_set_mux(rk628, GPIO1_B1, DDCM0SDARX);
rk628_pinctrl_set_mux(rk628, GPIO1_B2, DDCM0SCLRX);
```

6.15.11 声卡注册失败

如下，系统启动后，找不到 hdmiin 声卡

```
# cat /proc/asound/cards
 1 [rockchiphdmiin ]: rockchip_hdmiin - rockchip,hdmiin
                      rockchip,hdmiin
```

那么首先要检查的是，声卡依赖的一个 soc dai、codec dai 设备是否有加载成功：

```
rk3399:/ # cat /sys/kernel/debug/asoc/components
...
```

```
dummy-codec
ff8a0000.i2s
ff8a0000.i2s
ff880000.i2s
ff880000.i2s
...
rk3399:/ # cat /sys/kernel/debug/asoc/dais
...
ff8a0000.i2s
ff880000.i2s
dummy-codec
....
```

以上节点加载失败的话，可以检查 defconfig、dts

在 soc dai codec dai 都注册成功的情况下，出现声卡注册不上，有可能是 dma 资源不够，尤其是 rk3399 情况下出现比较多，在 4.19 以上内核，可以在 i2s 节点添加下面属性来解决：

```
--- a/arch/arm64/boot/dts/rockchip/rk3399.dtsi
+++ b/arch/arm64/boot/dts/rockchip/rk3399.dtsi
@@ -1788,6 +1788,7 @@
                clocks = <&cru SCLK_I2S2_8CH>, <&cru HCLK_I2S2_8CH>;
                resets = <&cru SRST_I2S2_8CH>, <&cru SRST_H_I2S2_8CH>;
                reset-names = "reset-m", "reset-h";
+               rockchip,capture-only;
                power-domains = <&power RK3399_PD_SDIOAUDIO>;
                status = "disabled";
```

7. 常见需求处理

7.1 RK628 24M 晶振来自其他 SOC 的配置方式

下面分别以几个 ROCKCHIP 的主控来提供配置方法，其中 DTS 的配置 CLK 都默认有在代码中使能，如果未使能开机后 24M CLK 会被关闭，从而导致 I2C 通信异常。

7.1.1 RK3399 添加 24M 支持

1. 在 rk628 dts 中引用以下配置

```
pinctrl-names = "default";
pinctrl-0 = <&rk628_rst>, <&clk_testout2>;
assigned-clocks = <&cru SCLK_TESTCLKOUT2>;
assigned-clock-rates = <24000000>;
clocks = <&cru SCLK_TESTCLKOUT2>;
clock-names = "soc_24M";
```

2. 根据实际硬件接法添加 IO 脚的 PINCTRL 配置

```
&pinctrl {
        test {
                clk_testout2: clk_testout2 {
                rockchip,pins = <0 8 RK_FUNC_3 &pcfg_pull_none>;
        };
};
```

7.1.2 RK3288 添加 24M 支持

1. 在 rk628 dts 中引用以下配置

```
pinctrl-names = "default";
pinctrl-0 = <&test_clkout>;
assigned-clocks = <&cru SCLK_TESTOUT_SRC>;
assigned-clock-parents = <&xin24m>;
clocks = <&cru SCLK_TESTOUT>;
clock-names = "soc_24M";
```

2. 根据实际硬件接法添加 IO 脚的 PINCTRL 配置

```
&pinctrl {
        test {
                test_clkout: test-clkout {
                        rockchip,pins = <0 17 RK_FUNC_1 &pcfg_pull_none>;
                };
        };
};
```

7.1.3 RK356X 添加 24M 支持

RK356X 能输出 24MHZ 的引脚比较多，例如 REF_CLKOUT（clk_wifi/gpio0_a0）、CAM_CLKOUT1（clk_cam1_out/gpio4_b0）、ETH_REFCLK_25M_M0（clk_mac1_out/gpio3_b0）,现以 clk_wifi 为例介绍。

1. 在 rk628 dts 中引用以下配置

```
&i2c2_rk628 {
    pinctrl-names = "default";
    pinctrl-0 = <&rk628_reset &refclk_pins>;
    assigned-clocks = <&pmucru CLK_WIFI>;
    assigned-clock-rates = <24000000>;
    clocks = <&pmucru CLK_WIFI>;
    clock-names = "soc_24M";
};

&pinctrl {
    rk628 {
        rk628_reset: rk628-reset {
            rockchip,pins = <2 RK_PA2 RK_FUNC_GPIO &pcfg_pull_none>;
        };
    };
};
```

2. 以下配置在 rk3568-pinctrl.dtsi 中自带

```
&pinctrl {
    ......
    refclk {
        /omit-if-no-ref/
        refclk_pins: refclk-pins {
            rockchip,pins =
                /* refclk_ou */
                <0 RK_PA0 1 &pcfg_pull_none>;
        };
    };
    ......
};
```

其他未写的 RK 芯片，可以参考文档 “Rockchip_Develop_Guide_Gpio_Output_Clocks_CN.pdf” 进行 24M 的处理。

7.2 双 RK628 支持

7.2.1 HDMI2CSI+HDMI2CSI 支持

7.2.1.1 注意事项

1. 两路 RK628 实现类似 camera 双摄的做法，两路 RK628 能够同时工作，需要主控支持两个 isp 控制器，同时处理两个输入数据，例如 RK3399 等。
2. 在硬件设计上，两个 RK628 需要挂在不同的 I2C 下或者两路 RK628 挂在同一个 I2C 下但设计成不同的 I2C 地址。
3. 注意 dts 的配置需要跟硬件设计一致，具体的 rk628 硬件接到 mipi_phy_rx0 或者 mipi_phy_tx1rx1，在 dts 也需要对应的配置。
4. 使用两路 rk628 实现 HDMI2CSI，两路 rk628 的 RESET、INT 控制 io 不能用一样的，否则会异常。

7.2.1.2 kernel dts 配置问题

现在以 RK3399 配置两路 RK628 为例介绍。

1. rk628->mipi_dphy_rx0->rkisp1_0，第一路 index 为 0，facing 配置为 back

```
&i2c1 {
    status = "okay";

    rk628_csi_v4l2: rk628_csi_v4l2@50 {
        compatible = "rockchip,rk628-csi-v4l2";
        reg = <0x50>;
        //clocks = <&ext_cam_clk>;
        //clock-names = "xvclk";

        interrupt-parent = <&gpio4>;
        interrupts = <16 IRQ_TYPE_LEVEL_LOW>;
        pinctrl-names = "default";
```

```
        pinctrl-0 = <&rk628_irq>;
        //power-gpios = <&gpio0 RK_PD5 GPIO_ACTIVE_HIGH>;
        reset-gpios = <&gpio4 RK_PD2 GPIO_ACTIVE_LOW>;
        plugin-det-gpios = <&gpio0 RK_PD6 GPIO_ACTIVE_LOW>;
        rockchip,camera-module-index = <0>;
        rockchip,camera-module-facing = "back";
        rockchip,camera-module-name = "RK628-CSI";
        rockchip,camera-module-lens-name = "NC";
        port {
            rk628_out: endpoint {
                remote-endpoint = <&hdmi2mipi_in>;
                data-lanes = <1 2 3 4>;
            };
        };
    };
};

&mipi_dphy_rx0 {
    status = "okay";

    ports {
        #address-cells = <1>;
        #size-cells = <0>;

        port@0 {
            reg = <0>;
            #address-cells = <1>;
            #size-cells = <0>;

            hdmi2mipi_in: endpoint@1 {
                reg = <1>;
                remote-endpoint = <&rk628_out>;
                data-lanes = <1 2 3 4>;
            };
        };

        port@1 {
            reg = <1>;
            #address-cells = <1>;
            #size-cells = <0>;

            dphy_rx0_out: endpoint@0 {
                reg = <0>;
                remote-endpoint = <&isp0_mipi_in>;
            };
        };
    };
};

&rkisp1_0 {
    status = "okay";

    port {
        #address-cells = <1>;
        #size-cells = <0>;

        isp0_mipi_in: endpoint@0 {
            reg = <0>;
```

```
            remote-endpoint = <&dphy_rx0_out>;
        };
    };
};
```

2. rk628->mipi_dphy_tx1rx1->rkisp1_1，第二路 index 为 1，facing 配置为 front

```
&i2c4 {
    status = "okay";

    rk628_csi_v4l2_1: rk628_csi_v4l2_1@50 {
        compatible = "rockchip,rk628-csi-v4l2";
        reg = <0x50>;
        //clocks = <&ext_cam_clk>;
        //clock-names = "xvclk";

        interrupt-parent = <&gpio3>;
        interrupts = <12 IRQ_TYPE_LEVEL_LOW>;
        pinctrl-names = "default";
        pinctrl-0 = <&rk628_irq1>;
        //power-gpios = <&gpio0 RK_PD5 GPIO_ACTIVE_HIGH>;
        reset-gpios = <&gpio4 RK_PD3 GPIO_ACTIVE_LOW>;
        plugin-det-gpios = <&gpio0 RK_PD7 GPIO_ACTIVE_LOW>;
        rockchip,camera-module-index = <1>;
        rockchip,camera-module-facing = "front";
        rockchip,camera-module-name = "RK628-CSI";
        rockchip,camera-module-lens-name = "NC";
        port {
            rk628_out1: endpoint {
                remote-endpoint = <&hdmi2mipi_in1>;
                data-lanes = <1 2 3 4>;
            };
        };
    };
};

&mipi_dphy_tx1rx1 {
    status = "okay";

    ports {
        #address-cells = <1>;
        #size-cells = <0>;

        port@0 {
            reg = <0>;
            #address-cells = <1>;
            #size-cells = <0>;

            hdmi2mipi_in1: endpoint@1 {
                reg = <1>;
                remote-endpoint = <&rk628_out1>;
                data-lanes = <1 2 3 4>;
            };
        };

        port@1 {
            reg = <1>;
```

```
            #address-cells = <1>;
            #size-cells = <0>;

            dphy_tx1rx1_out: endpoint@0 {
                reg = <0>;
                remote-endpoint = <&isp1_mipi_in>;
            };
        };
    };
};

&rkisp1_1 {
    status = "okay";

    port {
        #address-cells = <1>;
        #size-cells = <0>;

        isp1_mipi_in: endpoint@0 {
            reg = <0>;
            remote-endpoint = <&dphy_tx1rx1_out>;
        };
    };
};
```

7.2.1.3 android 配置问题

1. camera3_profiles.xml 文件配置注意事项：

moduleId：需要与 dts 中的 index 一致

第一路 **rk628**：

```
</Profiles>
<Profiles cameraId="0" name="rk628-csi" moduleId="m00">
    <Supported_hardware>
        <hwType value="SUPPORTED_HW_RKISP1"/>
    </Supported_hardware>
```

第二路 **rk628**:

```
</Profiles>
<Profiles cameraId="1" name="rk628-csi" moduleId="m01">
    <Supported_hardware>
        <hwType value="SUPPORTED_HW_RKISP1"/>
    </Supported_hardware>
```

2. 两路 rk628 如果需要同时使用，cameraHAL 需要添加支持，在 SDK/hardware/rockchip/camera$目录下添加修改：

```
diff --git a/common/platformdata/PlatformData.cpp
b/common/platformdata/PlatformData.cpp
index 36d2ac9..d3fcb4b 100755
--- a/common/platformdata/PlatformData.cpp
+++ b/common/platformdata/PlatformData.cpp
@@ -1047,7 +1047,7 @@ CameraHWInfo::CameraHWInfo() :
     mProductName = "<not_set>";
     mManufacturerName = "<not set>";
     mCameraDeviceAPIVersion = CAMERA_DEVICE_API_VERSION_3_3;
-    mSupportDualVideo = false;
+    mSupportDualVideo = true;
     mSupportExtendedMakernote = false;
     mSupportFullColorRange = true;
     mSupportIPUAcceleration = false;
```

7.2.2 HDMI2CSI+HDMI2DSI 支持

7.2.2.1 kernel dts 配置

HDMI2CSI+HDMI2DSI 组合：一路 RK628 走 camera 框架，接到主控端，一路 RK628 接到屏幕。这里以 RK3399 为例

1. HDMI2CSI 配置

rk628->mipi_phy_rx0->rkisp1_0 或者 rk628->mipi_dphy_tx1rx1->rkisp1_1，以接 rkisp1_0 为例

```
&i2c1 {
    status = "okay";

    rk628_csi_v4l2: rk628_csi_v4l2@50 {
        compatible = "rockchip,rk628-csi-v4l2";
        reg = <0x50>;
        //clocks = <&ext_cam_clk>;
        //clock-names = "xvclk";

        interrupt-parent = <&gpio4>;
        interrupts = <16 IRQ_TYPE_LEVEL_LOW>;
        pinctrl-names = "default";
        pinctrl-0 = <&rk628_irq>;
        //power-gpios = <&gpio0 RK_PD5 GPIO_ACTIVE_HIGH>;
        reset-gpios = <&gpio4 RK_PD2 GPIO_ACTIVE_LOW>;
        plugin-det-gpios = <&gpio0 RK_PD6 GPIO_ACTIVE_LOW>;
        rockchip,camera-module-index = <0>;
        rockchip,camera-module-facing = "back";
        rockchip,camera-module-name = "RK628-CSI";
        rockchip,camera-module-lens-name = "NC";
        port {
            rk628_out: endpoint {
                remote-endpoint = <&hdmi2mipi_in>;
                data-lanes = <1 2 3 4>;
            };
        };
    };
};
```

```
&mipi_dphy_rx0 {
    status = "okay";

    ports {
        #address-cells = <1>;
        #size-cells = <0>;

        port@0 {
            reg = <0>;
            #address-cells = <1>;
            #size-cells = <0>;

            hdmi2mipi_in: endpoint@1 {
                reg = <1>;
                remote-endpoint = <&rk628_out>;
                data-lanes = <1 2 3 4>;
            };
        };

        port@1 {
            reg = <1>;
            #address-cells = <1>;
            #size-cells = <0>;

            dphy_rx0_out: endpoint@0 {
                reg = <0>;
                remote-endpoint = <&isp0_mipi_in>;
            };
        };
    };
};

&rkisp1_0 {
    status = "okay";

    port {
        #address-cells = <1>;
        #size-cells = <0>;

        isp0_mipi_in: endpoint@0 {
            reg = <0>;
            remote-endpoint = <&dphy_rx0_out>;
        };
    };
};
```

2. HDMI2DSI 配置

直接参考 2.4.5 章节的 HDMI2DSI 转换即可。

7.2.2.2 android 配置

1. HDMI2CSI：参考一路 rk628 HDMI2CSI 即可，正确配置 camera3_profiles.xml 即可。
2. HDMI2DSI: android 部分不需要做配置