UNISOC Android 10 UDS710+UDX710 Camera Driver Customization Guide

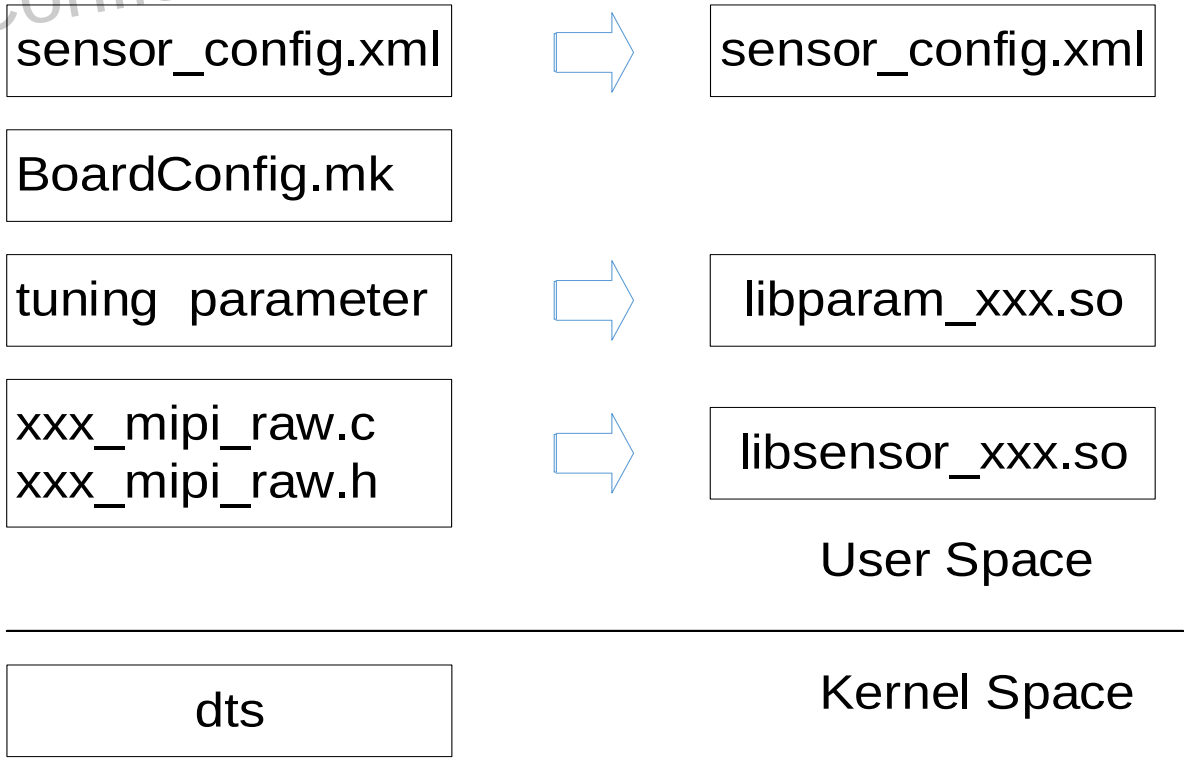
修改历史

版本号	日期	注释
V1.0	2020/05/15	初稿
V1.1	2020/09/22	适配Android 10进行修改

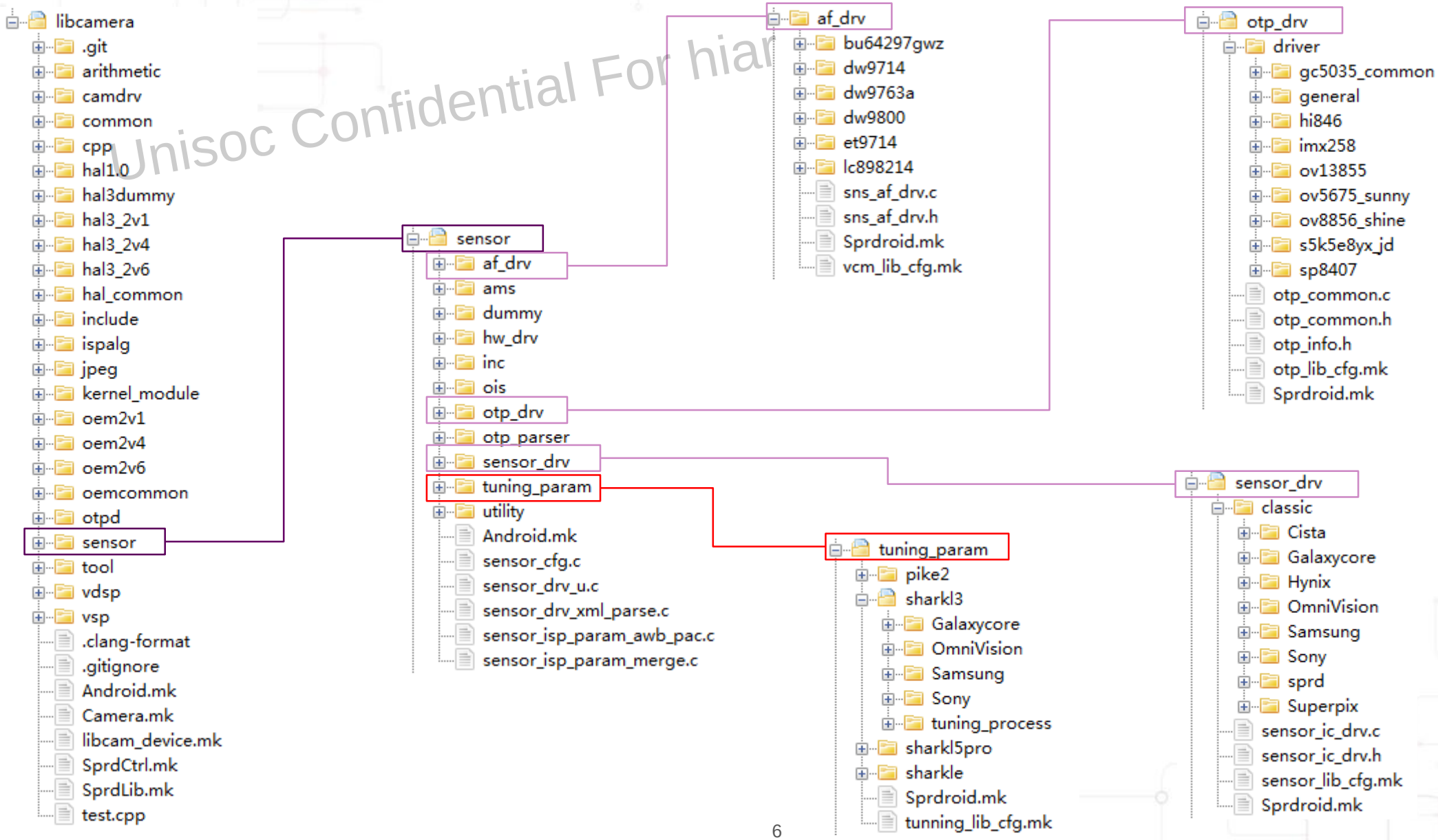
文档信息

适用产品信息	适用版本信息	关键字
UDS710_UDX710	Android 10.0	Sensor driver, OTP driver, AF driver, Flash driver

Contents



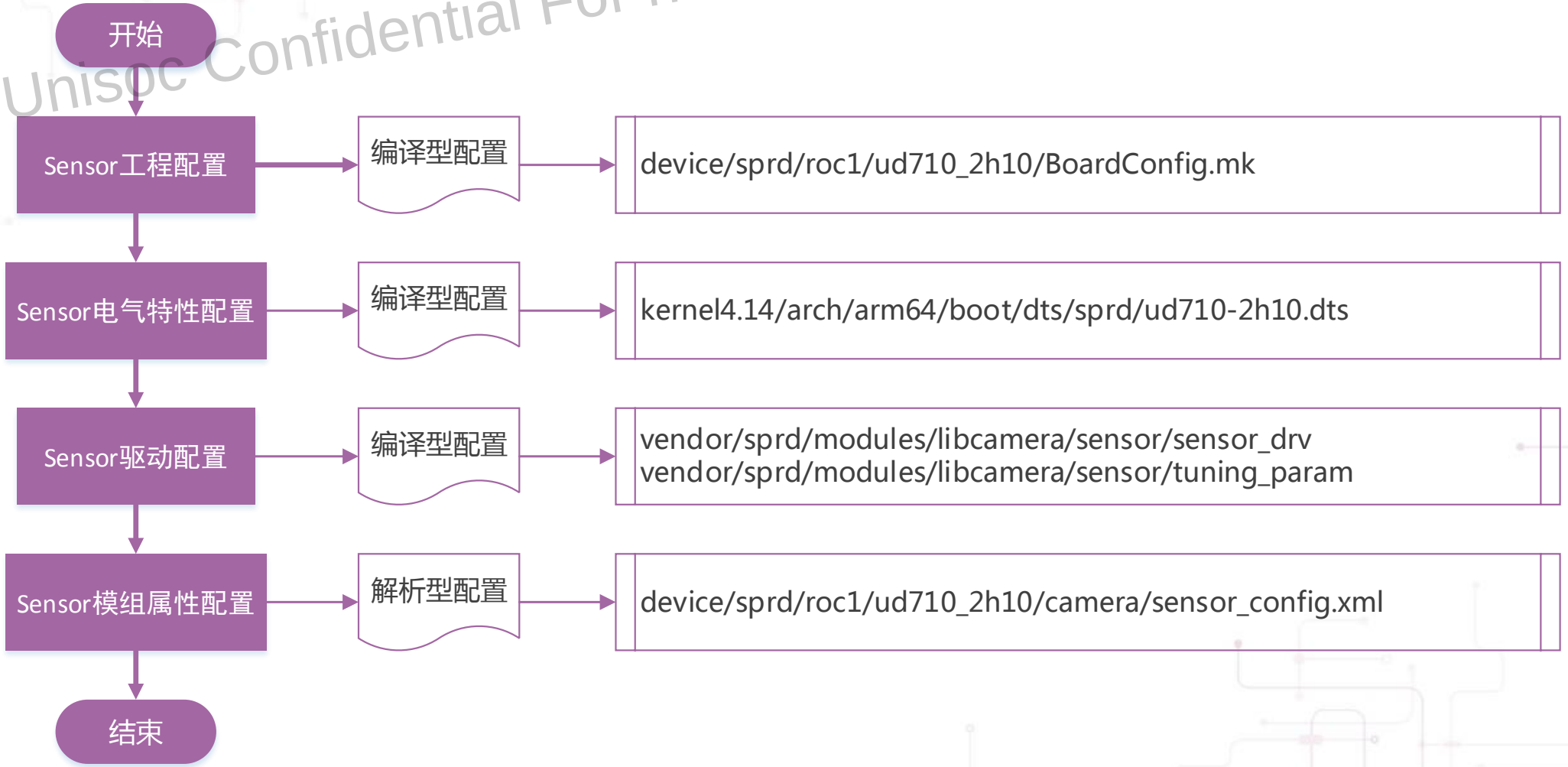
Camera Driver Introduction



Contents

Sensor Driver Porting

文档中以board ud710_2h10为例



Sensor Driver Porting-Sensor工程配置

Path: ddevice/sprd/roc1/ud710_2h10/BoardConfig.mk

- Set sensor resolution

```
#select camera 2M,3M,5M,8M,13M,16M,21M
CAMERA_SUPPORT_SIZE := 13M
FRONT_CAMERA_SUPPORT_SIZE := 8M
BACK_EXT_CAMERA_SUPPORT_SIZE := 5M
FRONT_EXT_CAMERA_SUPPORT_SIZE :=
```

CAMERA_SUPPORT_SIZE：后摄分辨率
FRONT_CAMERA_SUPPORT_SIZE：前摄分辨率
BACK_EXT_CAMERA_SUPPORT_SIZE：后辅摄分辨率
FRONT_EXT_CAMERA_SUPPORT_SIZE：前辅摄分辨率

- Add sensor name

```
#camera sensor support list
#example
#CAMERA_SENSOR_TYPE_BACK :="ov8856,ov8858"
CAMERA_SENSOR_TYPE_BACK := "imx351,xxx"
CAMERA_SENSOR_TYPE_FRONT := "ov8856_shine"
CAMERA_SENSOR_TYPE_BACK_EXT := "ov5675_dual"
CAMERA_SENSOR_TYPE_FRONT_EXT := ""
```

CAMERA_SENSOR_TYPE_BACK：后摄sensor型号
CAMERA_SENSOR_TYPE_FRONT：前摄sensor型号
CAMERA_SENSOR_TYPE_BACK_EXT：后辅摄sensor型号
CAMERA_SENSOR_TYPE_FRONT_EXT：前辅摄sensor型号

Note：sensor型号需与存放driver文件夹同名

- Add tuning parameter

```
#tuning param support list
TUNING_PARAM_LIST := "imx351,ov8865_shine,ov5675,ov5675_dual,xxx"
```

依次添加对应sensor参数名称

Note：sensor参数名称需与存放tuning parameter中sensor名相同

Sensor Driver Porting-Sensor电气特性配置

Path:kernel4.14/arch/arm64/boot/dts/sprd/ud710-2h10.dts

根据硬件原理图修改

i2c bus position

camera position

i2c address

mclk

avdd/dvdd/iovdd

mipi, phyid

csi

pwdn/reset

```
&i2c0 {
    status = "okay";
    sensor_main: sensor-main@34 {
        compatible = "sprd,sensor-main";
        reg = <0x34>;
        clock-names = "clk_src","sensor_eb",
            "clk_96m","clk_76m8",
            "clk_48m","clk_26m";
        clocks = <&mm_clk CLK_SENSOR1>,<&mm_gate CLK_MM_SENSOR1_EB>,
            <&g12_pll CLK_TWPLL_96M>,<&g12_pll CLK_TWPLL_76M8>,
            <&g12_pll CLK_TWPLL_48M>,<&ext_26m>;
        vddio-supply = <&vddcamio>;
        vddcama-supply = <&vddcama0>;
        vddcamd-supply = <&vddcamd0>;
        vddcammot-supply = <&vddcammot>;
        sprd,phyid = <0>;
        csi = <&csi0>;
        reset-gpios = <&ap_gpio 45 0>;
        power-down-gpios = <&ap_gpio 47 0>;
```

main：后主摄
sub：前摄
main2：后辅摄
sub2：前辅摄

非实际使用i2c地址，但需要用来区分i2c设备，不能相同

CSI1: phyid = 0
CSI0@4lane: phyid = 1
CSI0_m@2lane: phyid = 3
CSI0_s@2lane: phyid = 4
CSI2: phyid = 5

dcam0: csi0
dcam1: csi1

Sensor Driver Porting-Sensor电气特性配置

Path: kernel4.14/arch/arm64/boot/dts/sprd/ud710-2h10.dts

Path: u-boot15/board/spreadtrum/ud710_2h10/pinmap-ud710.c

根据硬件原理图修改

LDO供电，gpio口控制

将对应pin脚在pinmap中配置为gpio

dts中设置为gpio控制即可

```
&i2c0 {
    status = "okay";
    sensor_main: sensor-main@34 {
        compatible = "sprd,sensor-main";
        reg = <0x34>;

        clock-names = "clk_src","sensor_eb",
            "clk_96m","clk_76m8",
            "clk_48m","clk_26m";
        clocks = <&mm_clk CLK_SENSOR1>,<&mm_gate CLK_MM_SENSOR1_EB>,
            <&g12_pll CLK_TWPLL_96M>,<&g12_pll CLK_TWPLL_76M8>,
            <&g12_pll CLK_TWPLL_48M>,<&ext_26m>;
        vddio-supply = <&vddcamio>;
        vddcama-supply = <&vddcama0>;
        vddcamd-supply = <&vddcamd0>;
        vddcammot-supply = <&vddcammot>;
        sprd,phyid = <0>;
        csi = <&csi0>;
        dvdd-gpios = <&ap_gpio 62 0>;
        mipi-switch-en-gpios = <&ap_gpio 55 0>;
        mipi-switch-mode-gpios = <&ap_gpio 8 0>;
        reset-gpios = <&ap_gpio 45 0>;
        power-down-gpios = <&ap_gpio 47 0>;
```

mipi switch

将对应pin脚在pinmap中配置为gpio

在sensor驱动中控制

Sensor Driver Porting-Sensor电气特性配置-Example

```
&i2c0 {
    status = "okay";
    sensor_main: sensor-main@34 {
        compatible = "sprd,sensor-main";
        reg = <0x34>;

        clock-names = "clk_src","sensor_eb",
            "clk_96m","clk_76m8",
            "clk_48m","clk_26m";
        clocks = <&mm_clk CLK_SENSOR1>,<&mm_gate CLK_MM_SENSOR1_EB>,
            <&g12_pll CLK_TWPLL_96M>,<&g12_pll CLK_TWPLL_76M8>,
            <&g12_pll CLK_TWPLL_48M>,<&ext_26m>;
        vddio-supply = <&vddcamio>;
        vddcama-supply = <&vddcama0>;
        vddcamd-supply = <&vddcamd0>;
        vddcammot-supply = <&vddcammot>;
        sprd,phyid = <0>;
        csi = <&csi0>;
        reset-gpios = <&ap_gpio 45 0>;
        power-down-gpios = <&ap_gpio 47 0>;
```

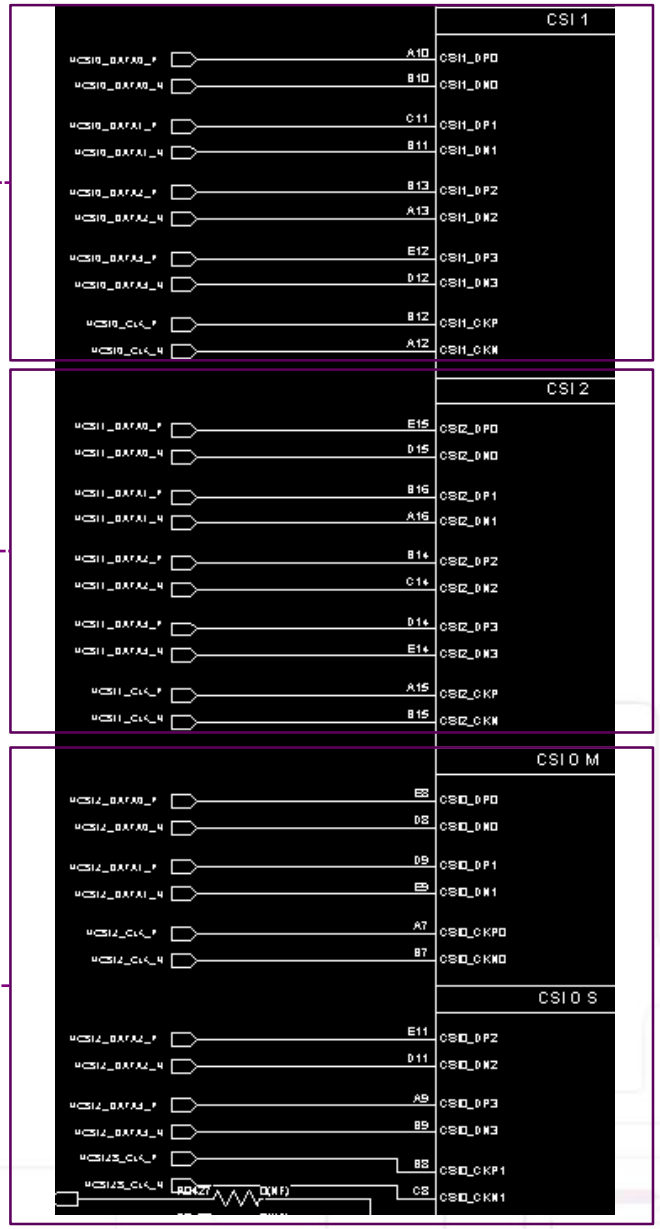
后主摄
I2c: i2c0
csi1: phyid = 0
clock: MCLK1
reset: gpio 45
pwdn: gpio 47

```
&i2c1 {
    status = "okay";
    sensor_main2: sensor-main2@20 {
        compatible = "sprd,sensor-main2";
        reg = <0x20>;

        clock-names = "clk_src","sensor_eb",
            "clk_96m","clk_76m8",
            "clk_48m","clk_26m";
        clocks = <&mm_clk CLK_SENSOR0>,<&mm_gate CLK_MM_SENSOR0_EB>,
            <&g12_pll CLK_TWPLL_96M>,<&g12_pll CLK_TWPLL_76M8>,
            <&g12_pll CLK_TWPLL_48M>,<&ext_26m>;
        vddio-supply = <&vddcamio>;
        vddcama-supply = <&vddcama1>;
        vddcamd-supply = <&vddcamd1>;
        vddcammot-supply = <&vddcammot>;
        sprd,phyid = <5>;
        csi = <&csi1>;
        reset-gpios = <&ap_gpio 41 0>;
        power-down-gpios = <&ap_gpio 40 0>;
```

后辅摄
i2c: i2c1
csi1: phyid = 5
clock: MCLK0
reset: gpio 41
pwdn: gpio 40

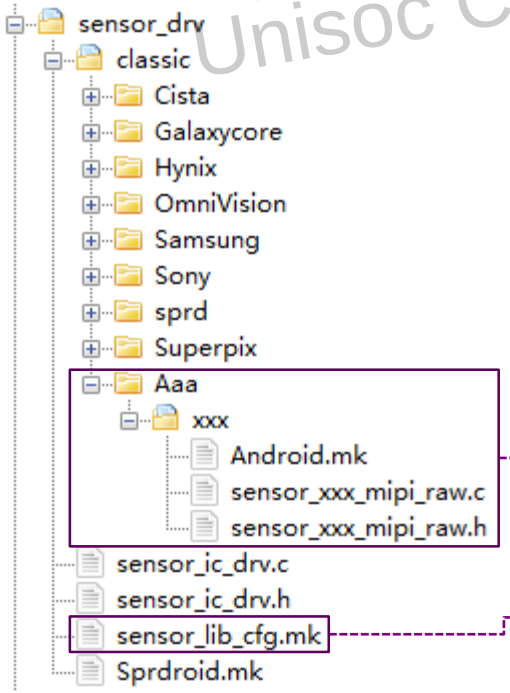
```
&i2c1 {
    status = "okay";

    sensor_sub: sensor-sub@6c {
        compatible = "sprd,sensor-sub";
        reg = <0x6c>;

        clock-names = "clk_src","sensor_eb",
            "clk_96m","clk_76m8",
            "clk_48m","clk_26m";
        clocks = <&mm_clk CLK_SENSOR0>,<&mm_gate CLK_MM_SENSOR0_EB>,
            <&g12_pll CLK_TWPLL_96M>,<&g12_pll CLK_TWPLL_76M8>,
            <&g12_pll CLK_TWPLL_48M>,<&ext_26m>;
        vddio-supply = <&vddcamio>;
        vddcama-supply = <&vddcama1>;
        vddcamd-supply = <&vddcamd1>;
        vddcammot-supply = <&vddcammot>;
        sprd,phyid = <1>;
        csi = <&csi0>;
        reset-gpios = <&ap_gpio 44 0>;
        power-down-gpios = <&ap_gpio 46 0>;
```

前摄
i2c: i2c1
csi0_s: phyid = 1
clock: MCLK0
reset: gpio 44
pwdn: gpio 46



Sensor Driver Porting-Sensor驱动配置-添加驱动

Path：vendor/sprd/modules/libcamera/sensor/sensor_drv



添加驱动文件夹

需与BoardConfig.mk中sensor型号同名

Android.mk

sensor_xxx_mipi_raw.c

sensor_xxx_mipi_raw.h

Android.mk中添加编译关系，参照模板添加

sensor_xxx_mipi_raw.c中添加函数

```
void *sensor_ic_open_lib(void){
        return
&g_xxx_mipi_raw_info;}
```

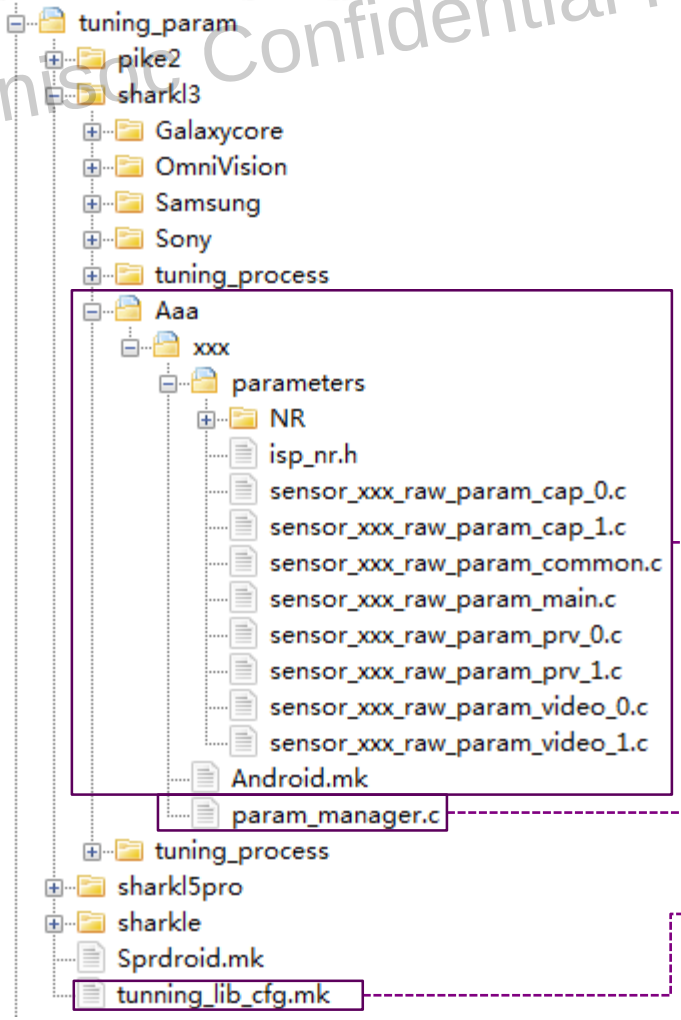
```
PRODUCT_PACKAGES += libsensor_imx351 \
    libsensor_ov5675_dual \
    libsensor_ov8856_shine \
    libsensor_xxx
```

Note:

Aaa: Sensor vendor; xxx: Sensor PN; yyy: Module vendor; zzz: VCM Driver IC PN

Sensor Driver Porting-Sensor驱动配置-添加参数

Path：vendor/sprd/modules/libcamera/sensor/tuning_param



添加参数文件夹

需与BoardConfig.mk中sensor参数名称同名parameter文件夹

Android.mk中添加编译关系，参照模板添加

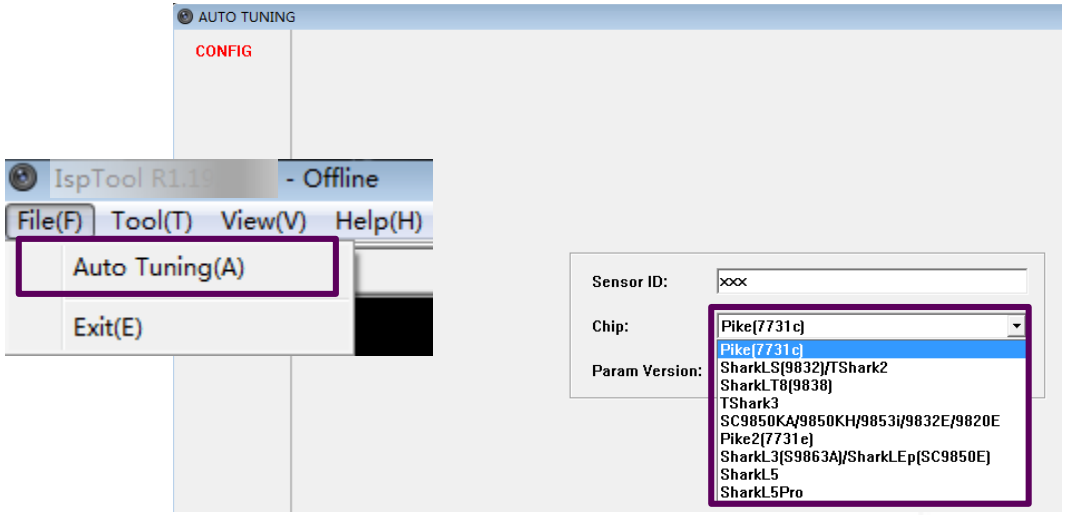
```
#include "sensor_raw.h"

void *tuning_param_get_ptr(void) {
    cmr_int rtn = 0;
    #include "parameters/sensor_xxx_raw_param_main.c"
    return &s_xxx_mipi_raw_info;
}
```

```
#ifeq ($(strip $(CHIP_NAME)),sharkl3)
#s9863a1h10
PRODUCT_PACKAGES += libparam_imx351
PRODUCT_PACKAGES += libparam_ov5675_dual
PRODUCT_PACKAGES += libparam_ov8856_shine
PRODUCT_PACKAGES += libparam_xxx
```

Sensor Driver Porting-Sensor驱动配置-添加参数

第一次点亮sensor，建议使用工具auto tuning生成初始参数

Chip	Auto Tuning Option	Version ID
SC9832E	SC9850KA/9850KH/9853i/9832E/9820E	0x00070005
SC7731E	Pike2(7731)	0x00080006
SC9863A	SharkL3(S9863A)/SharklEp(SC9850E)	0x00090007
UMS312/UDS710_UDX710	Sharkl5	0x000A0008
UMS512(T)	Sharkl5Pro	0x000B0009



Sensor Driver Porting-Sensor模组属性配置

Path：device/sprd/roc1/ud710_2h10/camera/sensor_config.xml

模组属性字段

SensorName：与BoardConfig.mk中sensor型号相同
Facing：sensor朝向
Orientation：sensor安装角度
Resource_cost：该sensor资源占有比例，0/50/100

OTPName：与OTP driver名称中相同
I2cAddr：E2prom i2c地址
E2promNum：E2prom数量及存放方式
E2promsize：E2prom大小，一般为8K/16K

AfName：与Af driver名称中相同
Mode：Af工作模式，默认为0

TuningName：与BoardConfig.mk中sensor参数名称相同

Sensor属性字段

OTP属性字段

VCM属性字段

Tuning参数属性字段

```
<CameraModuleCfg>
    <SlotId>0</SlotId>
    <SensorName>xxx</SensorName>
    <Facing>BACK</Facing>
    <Orientation>90</Orientation>
    <Resource_cost>0</Resource_cost>
    <OTP>
        <E2prom>
            <OtpName>xxx_yyy</OtpName>
            <I2cAddr>0x00</I2cAddr>
            <E2promNum>1</E2promNum>
            <E2promSize>8192</E2promSize>
        </E2prom>
    </OTP>
    <VCM>
        <AfName>zzz</AfName>
        <Mode>0</Mode>
    </VCM>
    <TuningParameter>
        <TuningName>xxx</TuningName>
    </TuningParameter>
</CameraModuleCfg>
```

Note: 如没有该功能，则不配置对应字段，Sensor属性字段必须有！

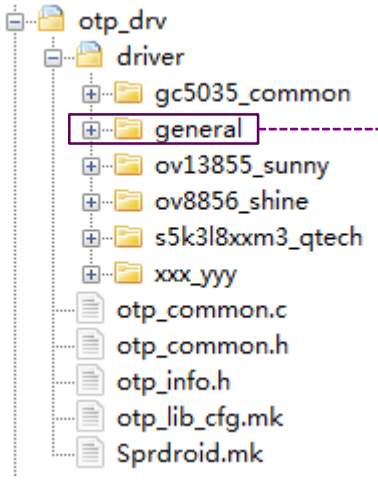
Contents

OTP Driver Porting-平台OTP

Path：vendor/sprd/modules/libcamera/sensor/otp_drv/driver

按照平台OTP规范烧录，数据存储在E2prom中，使用general驱动

参考P16配置sensor_config.xml




```
<CameraModuleCfg>
  <SlotId>0</SlotId>
  <SensorName>xxx</SensorName>
  <Facing>BACK</Facing>
  <Orientation>90</Orientation>
  <Resource_cost>0</Resource_cost>
  <OTP>
      <E2prom>
          <OtpName>general</OtpName>
          <I2cAddr>0x00</I2cAddr>
          <E2promNum>1</E2promNum>
          <E2promSize>8192</E2promSize>
      </E2prom>
  </OTP>
  <VCM>
      <AfName>zzz</AfName>
      <Mode>0</Mode>
  </VCM>
  <TuningParameter>
      <TuningName>xxx</TuningName>
  </TuningParameter>
</CameraModuleCfg>
```

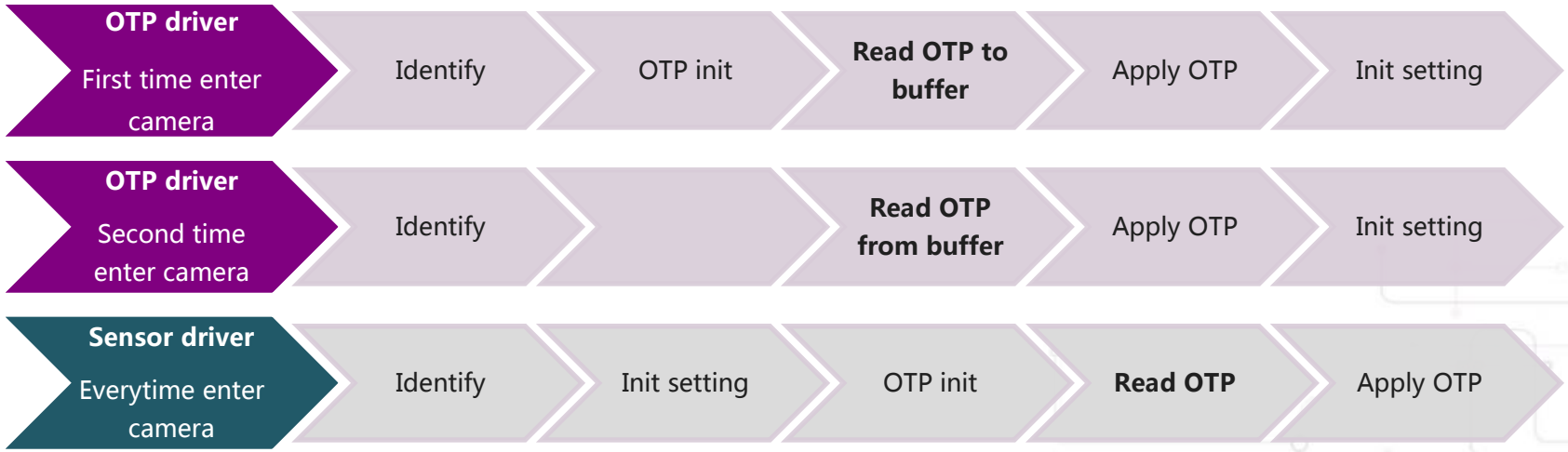
OTP Driver Porting-Sensor OTP

Path：vendor/sprd/modules/libcamera/sensor/otp_drv/driver

可使用两种方式应用sensor OTP

- 将sensor OTP的处理放在sensor driver中
- 借助平台OTP流程处理sensor OTP

方式	优点	缺点
Sensor driver	不需要单独维护OTP驱动	每一次进camera都会处理OTP，增加启动时间
平台OTP	仅开机第一次读取OTP	需按照平台OTP架构嵌入sensor OTP处理



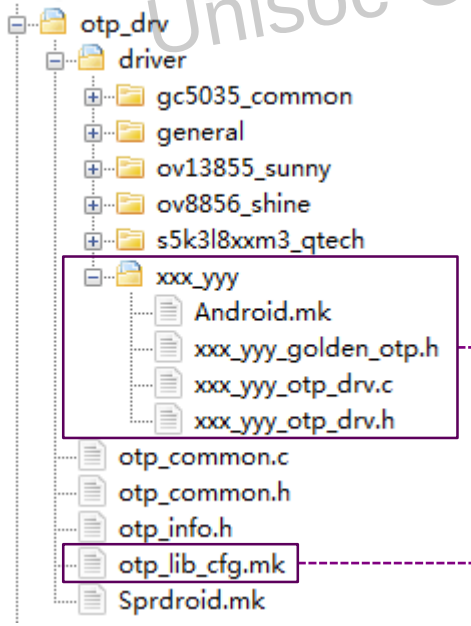
OTP Driver Porting-Sensor OTP in Sensor Driver

Path：vendor/sprd/modules/libcamera/sensor/sensor_drv


```
static cmr_int xxx_drv_access_val(cmr_handle handle, cmr_uint param)
{
    cmr_int ret = SENSOR_FAIL;
    SENSOR_VAL_T *param_ptr = (SENSOR_VAL_T *)param;

    SENSOR_IC_CHECK_HANDLE(handle);
    SENSOR_IC_CHECK_PTR(param_ptr);
    struct sensor_ic_drv_cxt * sns_drv_cxt = (struct sensor_ic_drv_cxt *)handle;

    SENSOR_LOGI("sensor xxx: param_ptr->type=%x", param_ptr->type);

    switch(param_ptr->type)
    {
        case SENSOR_VAL_TYPE_GET_STATIC_INFO:
            ret = xxx_drv_get_static_info(handle, param_ptr->pval);
            break;
        ......

#ifdef FEATURE_OTP
        case SENSOR_VAL_TYPE_READ_OTP:
            ret = xxx_yyy_identify_otp(handle, param_ptr);
            break;
#endif
        default:
            break;
    }
```

OTP Driver Porting-Sensor OTP Through Platform OTP

Path：vendor/sprd/modules/libcamera/sensor/otp_drv/driver



添加OTP驱动文件夹

需与sensor_config.xml中OTP驱动同名

Android.mk

xxx_yyy_golden_otp.h

xxx_yyy_otp_drv.c

xxx_yyy_otp_drv.h

Android.mk中添加编译关系，参照模板添加

xxx_yyy_otp_drv.c中添加函数

```
void *otp_driver_open_lib(void) {
        return
&xxx_yyy_otp_drv_entry; }
```

```
PRODUCT_PACKAGES += libotp_general \
                    libotp_s5k5e8yx_jd \
                    libotp_ov13855 \
                    libotp_gc5035_common \
                    libotp_s5k4h7_tsp \
                    libotp_xxx_yyy
```

OTP Driver Porting-Sensor OTP Through Platform OTP

Path：device/sprd/roc1/ud710_2h10/camera/sensor_config.xml

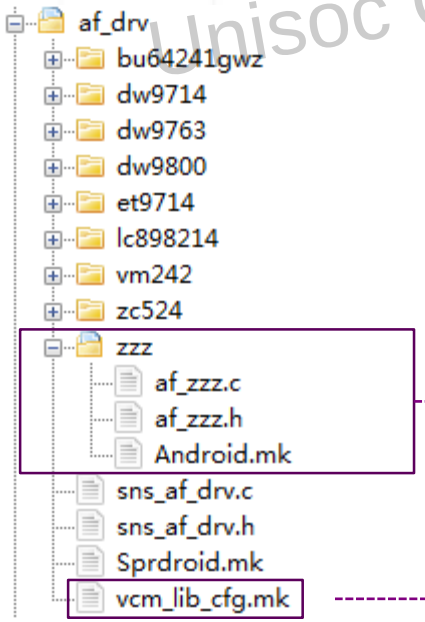
```
<CameraModuleCfg>
  <SlotId>0</SlotId>
  <SensorName>xxx</SensorName>
  <Facing>BACK</Facing>
  <Orientation>90</Orientation>
  <Resource_cost>0</Resource_cost>
  <OTP>
      <E2prom>
          <OtpName>xxx_yyy</OtpName>
          <I2cAddr>0x00</I2cAddr>
          <E2promNum>1</E2promNum>
          <E2promSize>8192</E2promSize>
      </E2prom>
  </OTP>
  <VCM>
      <AfName>zzz</AfName>
      <Mode>0</Mode>
  </VCM>
  <TuningParameter>
      <TuningName>xxx</TuningName>
  </TuningParameter>
</CameraModuleCfg>
```

0: single camera, one EEPROM
1: dual camera, one EEPROM
2: dual camera, two EEPROM
3: multi camera, independent EEPROM

Contents

AF Driver Porting

Path：vendor/sprd/modules/libcamera/sensor/af_driver



添加OTP驱动文件夹

需与sensor_config.xml中AF驱动同名

Android.mk

af_zzz.c

af_zzz.h

```
PRODUCT_PACKAGES += libvcm_dw9714p \
                    libvcm_lc898213 \
                    libvcm_dw9768v \
                    libvcm_dw9800 \
                    libvcm_dw9714 \
                    libvcm_dw9718s \
                    libvcm_zc524 \
                    libvcm_zzz
```

Android.mk中添加编译关系，参照模板添加

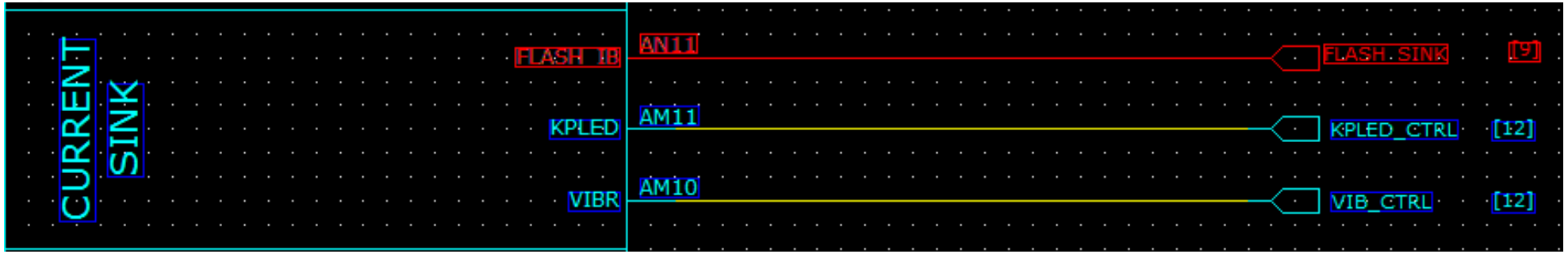
af_zzz.c中添加函数

```
void *vcm_driver_open_lib(void){
return &zzz_drv_entry;}
```

Contents

平台支持两种方式控制闪光灯

- 内置PMIC
  - ✓ 平台PMU 输出PIN脚Flash_IB，提供电源及控制
  - ✓ 例如：SC2720，SC2721，SC2703等



- 外置闪光灯IC
  - ✓ GPIO控制，例如：AW3641等
  - ✓ GPIO+i2c控制，例如：OCP8137，AW3648等

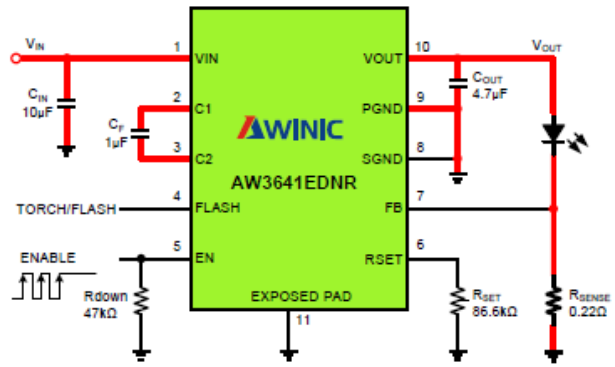


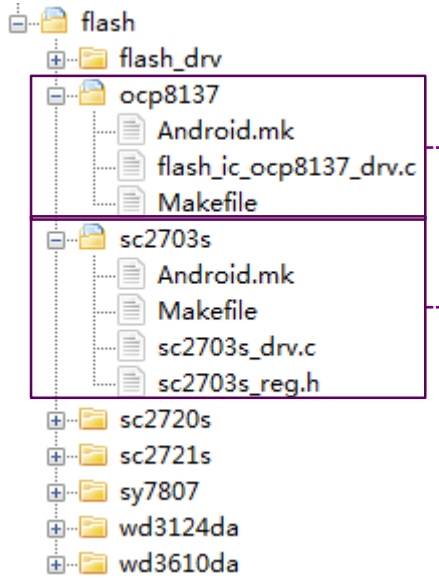
Figure 1 Typical Application Circuit of AW3641E

Flash Driver Porting

Path：vendor/sprd/modules/libcamera/kernel_module/flash

Path ：kernel4.14/arch/arm64/boot/dts/sprd/ud710-2h10.dts

以ocp8137和sc2703为例



添加Flash驱动文件夹

Flash_ic_oc8137.c

Android.mk

Makefile

添加Flash驱动文件夹

sc2703s_drv.c/sc2703s_reg.h

Android.mk

Makefile

```
flash_ic: flash-ic@63 {
    compatible = "sprd,flash-ocp8137";
    reg = <0x63>;
    sprd,flash-ic = <8137>;
    sprd,torch = <1>;
    sprd,preflash = <1>;
    sprd,highlight = <1>;
    sprd,torch-level = <128>;
    sprd,preflash-level = <128>;
    sprd,highlight-level = <128>;
    sprd,lvfm-enable = <1>;
    flash-torch-en-gpios = <&ap_gpio 88 0>;
    flash-chip-en-gpios = <&ap_gpio 89 0>;
    flash-en-gpios = <&ap_gpio 76 0>;
    flash-sync-gpios = <&ap_gpio 141 0>;
};
```

```
flash: sc2703-flash@4a {
    compatible = "sprd,sc2703-flash";
    reg = <0x4a>;
    flash-chip-en-gpios = <&ap_gpio 137 0>;
};
```

Path：vendor/sprd/modules/libcamera/kernel_module/init.sprd_flash.rc

```
on sprd-flash-ko
    insmod ${ro.vendor.ko.mount.point}/socko/flash_ic_sc2703s.ko
    insmod ${ro.vendor.ko.mount.point}/socko/flash_ic_ocp8137.ko
    insmod ${ro.vendor.ko.mount.point}/socko/flash_ic_sy7807.ko
```

Flash Driver Porting-Configuration

Path：device/sprd/roc1/ud710_2h10/BoardConfig.mk

- 双色温闪光灯配置

```
#flash led  feature
TARGET_BOARD_CAMERA_FLASH_LED_0 := true/false   #flash led0 config
TARGET_BOARD_CAMERA_FLASH_LED_1 := true/false   #flash led1 config
```

- 前摄闪光灯模式

```
#front flash type
#lcd,led,flash
TARGET_BOARD_FRONT_CAMERA_FLASH_TYPE := lcd/led/flash
#lcd: lcd补光
#led: led补光
#flash: 闪光灯模式
```

- 闪光灯总亮度等级

```
#Range of value 0~31
CAMERA_TORCH_LIGHT_LEVEL := 16  #0~31 level
```

Contents

Compile and Download

1. Set environment variable, enter root directory

source build/envsetup.sh

2. Choose compile option, show as right

lunch <boardname>

3. Install kernel header, execute when first compiling

kheader

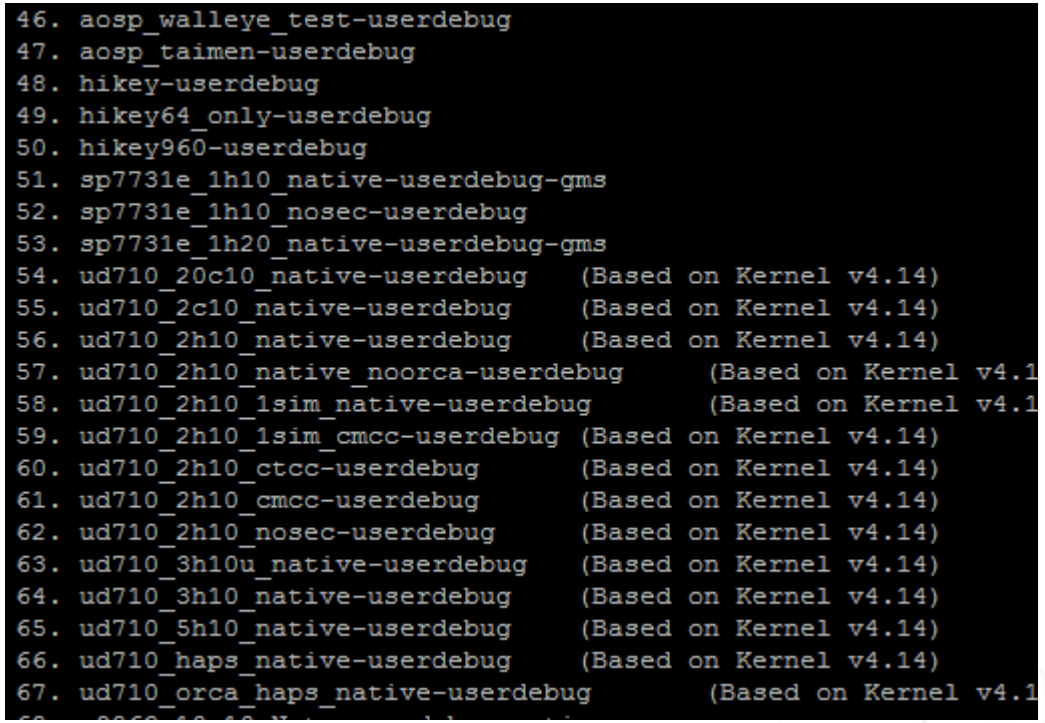
4. Compile whole project

make

5. Compile camera module, enter directory

vendor/sprd/modules/libcamera

mm



Compile and Download

Path：out/target/product/ud710_2h10/vendor/lib

- 修改sensor驱动

Generated	Download
libsensor_xxx_.so	adb push libsensor_xxx_.so vendor/lib

- 修改sensor tuning parameter

Generated	Download
libparam_xxx_.so	adb push libparam_xxx_.so vendor/lib

- 动态更换驱动或tuning parameter
  - 已添加在BoardConfig.mk中，先编译生成sprd_camera.ko，然后编译生成so文件
  - 更新sensor_config.xml中配置

Generated	Download
libsensor_xxx_.so libparam_xxx_.so sensor_config.xml	adb push libsensor_xxx_.so vendor/lib adb push libparam_xxx_.so vendor/lib adb push sensor_config.xml vendor/etc

Note: 所有修改建议重启手机后验证生效

THANKS

WWW.UNISOC.COM
紫 光 展 锐 科 技