SC600Y&SC600T Camera Driver Development Guide

Smart LTE Module Series

Rev. SC600Y&SC600T_Camera_Driver_Development_Guide_V1.0

Date: 2019-04-23

Status: Released

Our aim is to provide customers with timely and comprehensive service. For any assistance, please contact our company headquarters:

Quectel Wireless Solutions Co., Ltd.
7th Floor, Hongye Building, No.1801 Hongmei Road, Xuhui District, Shanghai 200233, China
Tel: +86 21 5108 6236
Email: info@quectel.com

Or our local office. For more information, please visit:
http://www.quectel.com/support/sales.htm

For technical support, or to report documentation errors, please visit:
http://www.quectel.com/support/technical.htm
Or email to: support@quectel.com

GENERAL NOTES

QUECTEL OFFERS THE INFORMATION AS A SERVICE TO ITS CUSTOMERS. THE INFORMATION PROVIDED IS BASED UPON CUSTOMERS' REQUIREMENTS. QUECTEL MAKES EVERY EFFORT TO ENSURE THE QUALITY OF THE INFORMATION IT MAKES AVAILABLE. QUECTEL DOES NOT MAKE ANY WARRANTY AS TO THE INFORMATION CONTAINED HEREIN, AND DOES NOT ACCEPT ANY LIABILITY FOR ANY INJURY, LOSS OR DAMAGE OF ANY KIND INCURRED BY USE OF OR RELIANCE UPON THE INFORMATION. ALL INFORMATION SUPPLIED HEREIN IS SUBJECT TO CHANGE WITHOUT PRIOR NOTICE.

COPYRIGHT

About the Document

History

Revision	Date	Author	Description
1.0	2019-04-23	Aikin CHEN	Initial

Contents

Table Index

Figure Index

1 Introduction

This document provides driver development guidelines for the camera module (such as the camera sensor), and describes how to bring up the camera on the Android platform of Quectel SC600Y&SC600T modules.

The camera sensor framework includes the configuration of the following components:

- Sensor
- CSIPHY
- CSID
- Actuator
- Flash
- EEPROM
- Chromatix™

NOTES

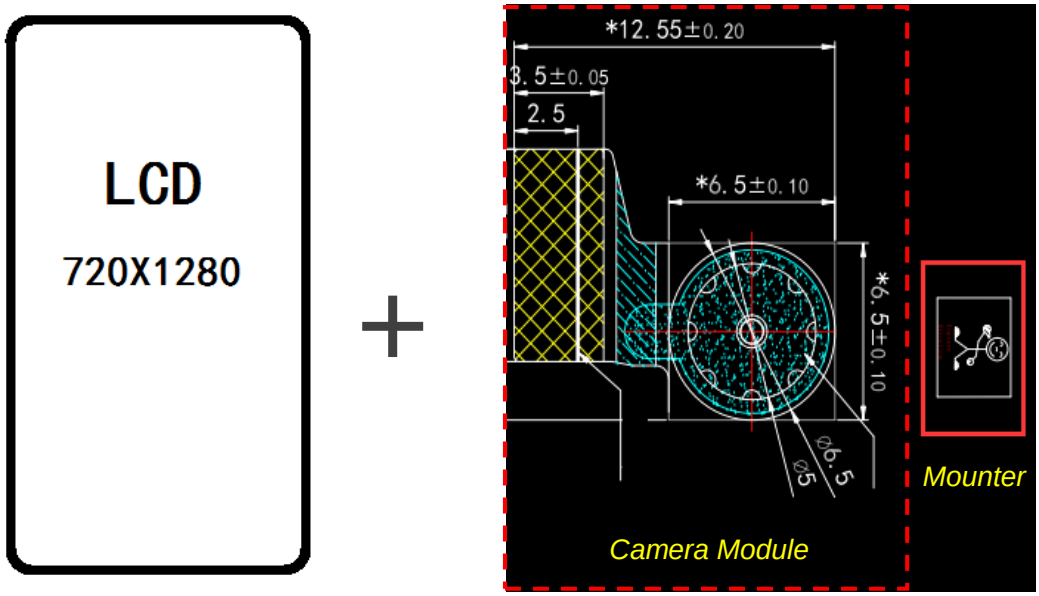
1. The main camera S5K3P3 on SC600Y&SC600T EVB (Smart EVB G2) will be used as an example in this document.
2. If any file under *vendor* directory is modified, then it is necessary to make the vendor image. If any file under *kernel* directory is modified, then it is necessary to make the boot image and dtbo image.

2 Assembling of Camera Module

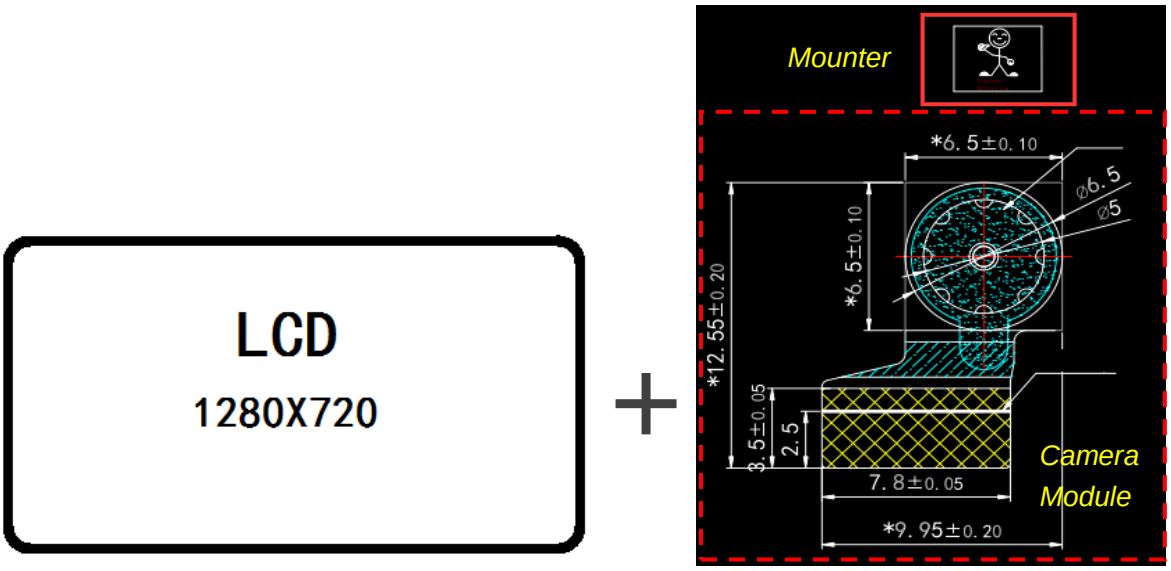
Before assembling the camera module to customers' PCB, please pay special attention to its mounting direction.

In order to ensure proper information display on the LCD, please be aware of the relative position between the LCD and the camera module as illustated below.

- The mounter should always stand on the long side of the LCD.
- When the LCD is mounted in lengthwise direction as illustrated below, the relative position between the LCD and the camera module should be:



- When the LCD is mounted in crosswise direction as illustrated below, the relative position between the LCD and the camera module should be:



NOTES

1. The design schematic of the camera module shown above is provided as an example only. The specific schematics should be acquired from corresponding camera module suppliers.
2. If the camera module has to be rotated by 180 degrees based on the relative directions illustrated above, then it is recommended to modify the information display direction through the user space *vendor/qcom/proprietary-camera-camera2/media-controller/modulesnsors/configs/msm8953_camera.xml*.

```
<CameraConfigurationRoot>
  <CameraModuleConfig>
    <CameraId>0</CameraId>
    <SensorName>s5k3p3</SensorName>
    <ActuatorName>dw9763</ActuatorName>
    <EepromName>dw9763_2d</EepromName>
    <FlashName>pmic</FlashName>
    <ChromatixName>s5k3p3_chromatix</ChromatixName>
    <ModesSupported>1</ModesSupported>
    <Position>BACK</Position>
    <MountAngle>270</MountAngle>
    <CSIInfo>
```

3. If the camera module has to be rotated by 90 degrees based on the relative directions illustrated above, then it is necessary for the camera module manufacturers to change the module's image output direction.

3 Information Provided by Camera Module Manufacturers

The camera module manufacturers should provide:

1. Sensor datasheet, AF datasheet (if the camera module has AF)
2. User-space sensor driver
3. Sensor chromatix code
4. User-space AF actuator driver
5. AF actuator effect code
6. User-space EEPROM driver

Taking the main camera S5K3P3 on EVB as an example, the following code and drivers are all provided by the manufacturer:

1. User-space sensor driver and sensor chromatix code

vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors:
sensor/libs/s5k3p3/
├──── Android.mk
├──── s5k3p3_lib.c
└──── s5k3p3_lib.h
chromatix/0310/chromatix_s5k3p3/
├──── 3A
│ ├──── 4k_preview
│ ├──── 4k_video
│ ├──── default_preview
│ ├──── default_video
│ ├──── hfr_120
│ ├──── hfr_60
│ ├──── hfr_90
│ ├──── zsl_preview
│ └──── zsl_video
├──── common
├──── cpp
│ ├──── cpp_hfr_120
│ ├──── cpp_hfr_60
│ ├──── cpp_hfr_90

```
│   ├── cpp_liveshot
│   ├── cpp_preview
│   ├── cpp_snapshot
│   └── cpp_video
│   └── cpp_video_4k
├── isp
│   ├── hfr_120
│   ├── hfr_60
│   ├── hfr_90
│   ├── preview
│   ├── snapshot
│   └── video
│   └── video_4k
└── postproc
```

2. User-space AF actuator driver

vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/actuator/libs/dw9763/

```
├── Android.mk
├── dw9763_actuator.c
└── dw9763_actuator.h
```

3. User-space EEPROM driver

vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/eeprom/libs/dw9763_2d/

```
├── Android.mk
├── dw9763_2d_eeprom.c
└── dw9763_2d_eeprom.h
```

4 Camera Circuit Diagram

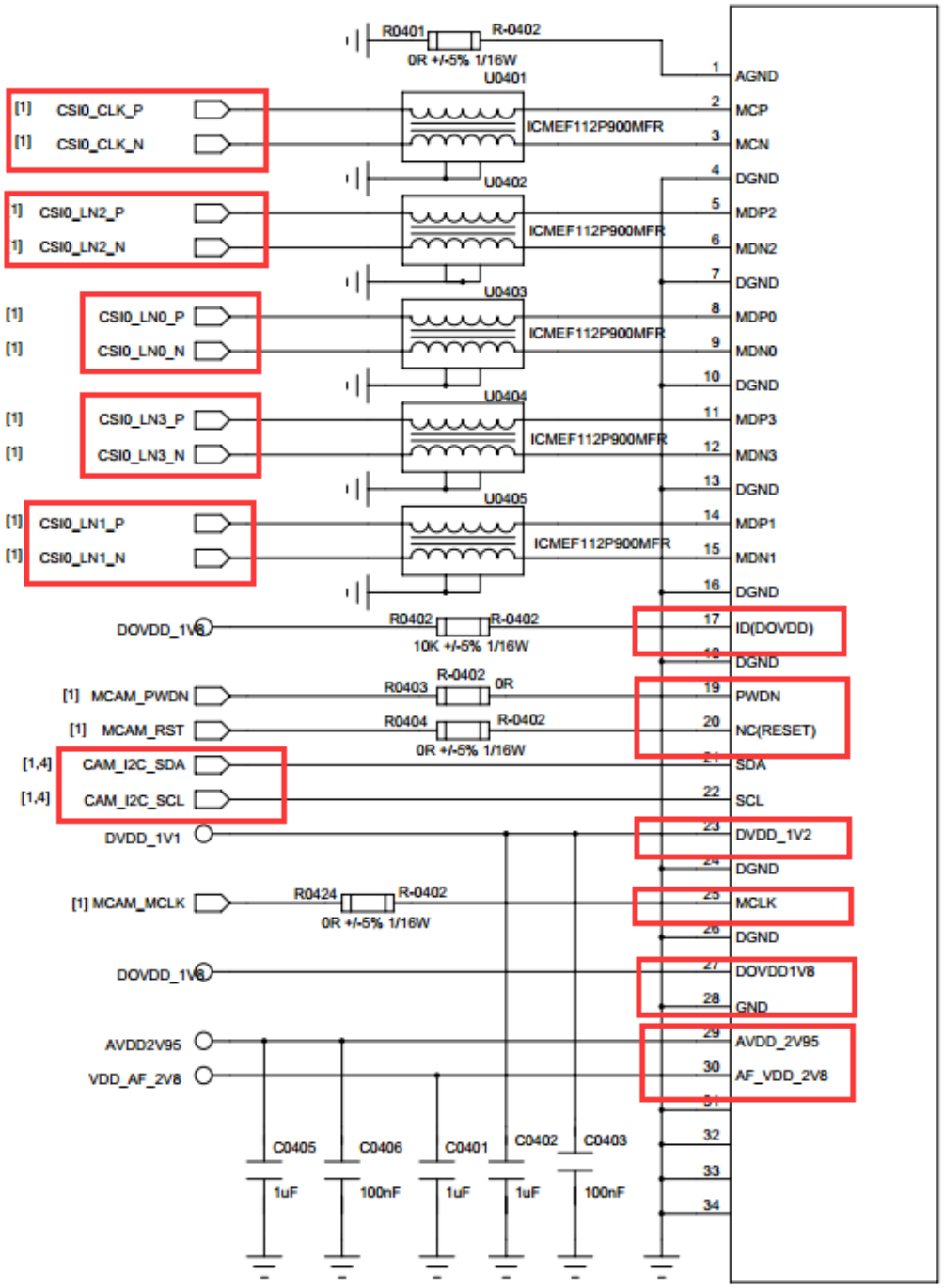


Figure 1: Main Camera Circuit Diagram

- Power supply: DVDD (1.2V), AVDD (2.8V), IOVDD (1.8V) (DOVDD), AFVDD (2.8V)
- Reset: RESET
- Suspend: PWDN
- Clock: MCLK
- MIPI Data: MDP0 MDN0, MDP1 MDN1, MDP2 MDN2, MDP3 MDN3
- MIPI Clock: MCP, MCN
- I2C: SDA, SCL

NOTE

I2C: Qualcomm CCI interface. It is only used for camera.

While debugging sensor, the MCLK, power supply, and the power sequence should be configured first.

- The sensor's power supply pins are mainly DVDD (1.2V), AVDD (2.8V) and IOVDD (1.8V). While AFVDD (2.8V) is not the power supply pin of camera sensor, it's for actuator.
- RESET and PWDN pins are also associated with power on sequence. Sometimes, the PWDN pin is not available for the camera module circuit design/connection, and is internally pulled up.

NOTE

After power, MCLK, and power-on sequence are all configured, the sensor will work properly. Thus, SC600Y&SC600T will be able to communicate with the sensor via I2C, and then read the ID from the sensor ID register.

5 Add Sensor Driver

5.1. Kernel Driver

This section provides the information necessary for adding the kernel driver.

Path: *kernel/msm-4.9/arch/arm/boot/dts/qcom/msm8953-camera-sensor-mtp.dtsi*

5.1.1. GPIO Configuration

As shown below, customers can configure sensor-specific GPIOs based on the target board.

For explanations on each property, please refer to documents in the following path:
kernel/msm-4.9/Documentation/devicetree/bindings/video/

GPIOs can be configured by the following two ways, based on the software used.

1. Using pinctrl

Pinctrl node entries in .dtsi file can be used to configure GPIOs, e.g.:

```
pinctrl-names = "cam_default", "cam_suspend";
pinctrl-0 = <&cam_sensor_mclk0_default &cam_sensor_rear_default>;
pinctrl-1 = <&cam_sensor_mclk0_sleep &cam_sensor_rear_sleep>;
```

2. Using GPIO control

GPIO node entries in .dtsi file can be used to configure GPIOs, e.g.:

```
gpios = <&msm_gpio 26 0>,
        <&msm_gpio 35 0>,
        <&msm_gpio 34 0>;
qcom,gpio-reset = <1>;
qcom,gpio-standby = <2>;
qcom,gpio-req-tbl-num = <0 1 2>;
qcom,gpio-req-tbl-flags = <1 0 0>;
qcom,gpio-req-tbl-label = "CAMIF_MCLK",
```

```
"CAM_RESET1",
"CAM_STANDBY";
```

5.1.2. Clock-related Settings

In the .dts file, for each sensor node, the customer can configure clock source as follows:

```
clocks = <&clock_gcc clk_mclk0_clk_src>,
         <&clock_gcc clk_gcc_camss_mclk0_clk>;
clock-names = "cam_src_clk", "cam_clk";
```

The order of the lists in the two properties is important. The nth clock-name will correspond to the nth entry in the clock's property. Thus, the two properties above, cam_src_clk would correspond to clk_mclk0_clk_src, cam_clk should correspond to clk_gcc_camss_mclk0_clk, etc. The customer does not need to change this, as it is parsed in the clock framework.

5.1.3. Power Handler

1. PMIC case

```
cam_vio-supply = <&pm8953_l6>;
cam_vdig-supply = <&pm8953_l2>;
cam_vaf-supply = <&pm8953_l17>;
cam_vana-supply = <&pm8953_l22>;
qcom,cam-vreg-name = "cam_vio", "cam_vdig", "cam_vaf",
                     "cam_vana";
qcom,cam-vreg-min-voltage = <0 1100000 2850000 2800000>;
qcom,cam-vreg-max-voltage = <0 1100000 2850000 2800000>;
qcom,cam-vreg-op-mode = <0 105000 100000 80000>;
```

2. GPIO case

```
gpios = <&tlmm 26 0>,
        <&tlmm 40 0>,
        <&tlmm 39 0>,
        <&tlmm 3 0>;
qcom,gpio-reset = <1>;
qcom,gpio-standby = <2>;
qcom,gpio-vdig = <3>;
qcom,gpio-req-tbl-num = <0 1 2 3>;
qcom,gpio-req-tbl-flags = <1 0 0 0>;
qcom,gpio-req-tbl-label = "CAMIF_MCLK0",
        "CAM_RESET0",
        "CAM_STANDBY0",
        "CAM_VDIG";
```

- CAM_VANA – Supply voltage (analog)
- CAM_VDIG – Supply voltage (digital)

- CAM_VIO – Input/output voltage (digital)
- CAM_VAF – Supply voltage (actuator voltage)

5.1.4. Taking EVB's Main Camera S5K3P3 as an Example

- DVDD: gpio3 control
- AVDD: LDO22 supply
- IOVDD: LDO6 supply
- RESET: gpio40 control
- PWDN: gpio39 control

The power supply type can be LDO supply or GPIO control, so two places need to be configured.

In the following example codes, vdig configures not only LDO2 but also GPIO3. But actually only GPIO3 needs to be configured.

- *kernel/msm-4.9/arch/arm/boot/dts/qcom/msm8953-camera-sensor-mtp.dtsi*

```
camera0: qcom,camera@0 {
        cell-index = <0>;
        compatible = "qcom,camera";
        reg = <0x0>;
        qcom,csiphy-sd-index = <0>;
        qcom,csid-sd-index = <0>;
        qcom,mount-angle = <270>;
        qcom,eeprom-src = <&eeprom0>;
        qcom,actuator-src = <&actuator0>;
        cam_vio-supply = <&pm8953_l6>;
        cam_vdig-supply = <&pm8953_l2>;
        cam_vaf-supply = <&pm8953_l17>;
        cam_vana-supply = <&pm8953_l22>;
        qcom,cam-vreg-name = "cam_vio", "cam_vdig", "cam_vaf", "cam_vana";
        qcom,cam-vreg-min-voltage = <0 1100000 2850000 2800000>;
        qcom,cam-vreg-max-voltage = <0 1100000 2850000 2800000>;
        qcom,cam-vreg-op-mode = <0 105000 100000 80000>;
        pinctrl-names = "cam_default", "cam_suspend";
        pinctrl-0 = <&cam_sensor_mclk0_default &cam_sensor_rear_default &cam_sensor_rear_vana>;
        pinctrl-1 = <&cam_sensor_mclk0_sleep &cam_sensor_rear_sleep &cam_sensor_rear_vana_sleep>;
        gpios = <&tlmm 26 0>,
                <&tlmm 40 0>,
                <&tlmm 39 0>,
                <&tlmm 3 0>;
        qcom,gpio-reset = <1>;
        qcom,gpio-standby = <2>;
        qcom,gpio-vdig = <3>;
        qcom,gpio-req-tbl-num = <0 1 2 3>;
```

IOVDD
DVDD
AVDD
RESET
PWDN
DVDD

- *kernel/msm-4.9/arch/arm/boot/dts/qcom/msm8953-pinctrl.dtsi*

```
cam_sensor_rear_default: cam_sensor_rear_default {
        /* RESET, STANDBY */
        mux {
                pins = "gpio40", "gpio39";
                function = "gpio";
        };

        config {
                pins = "gpio40","gpio39";
                bias-disable; /* No PULL */
                drive-strength = <2>; /* 2 MA */
        };
};

cam_sensor_rear_sleep: cam_sensor_rear_sleep {
        /* RESET, STANDBY */
        mux {
                pins = "gpio40","gpio39";
                function = "gpio";
        };

        config {
                pins = "gpio40","gpio39";
                bias-disable; /* No PULL */
                drive-strength = <2>; /* 2 MA */
        };
};
```

```
cam_sensor_rear_vana: cam_sensor_rear_vdig {
        /* VDIG */
        mux {
                pins = "gpio3";
                function = "gpio";
        };

        config {
                pins = "gpio3";
                bias-disable; /* No PULL */
                output-low;
                drive-strength = <2>; /* 2 MA */
        };
};

cam_sensor_rear_vana_sleep: cam_sensor_rear_vdig_sleep {
        /* VDIG */
        mux {
                pins = "gpio3";
                function = "gpio";
        };

        config {
                pins = "gpio3";
                bias-disable; /* No PULL */
                drive-strength = <2>; /* 2 MA */
        };
};
```

5.2. User Space Driver

This section describes information necessary for creating the user space driver.

The sensor's user space driver should be provided by the camera module manufacturer. In this section, only the important parts, such as the sensor driver and chromatix code, are shown.

5.2.1. Sensor Driver

5.2.1.1. Add sensor driver

Refer to *s5k3p3_lib.c*, *s5k3p3_lib.h*, *s5k3p3_pdaf_flip_mirror.h*, *s5k3p3_pdaf.h*, and *Android.mk files* in: *vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/sensor/libs/s5k3p3/.*

5.2.1.2. Configure Sensor Driver

Generally, it is no need to modify the parameters mentioned in the following sections, as the manufacturer has configured them already.

1) Configure camera ID, slaver address, sensor ID

After power-on, the driver will read the sensor ID. If it is successfully matched, it will probe successfully.

```
static sensor_lib_t sensor_lib_ptr =
{
  .sensor_slave_info =
  {
    .sensor_name = SENSOR_MODEL,
    .slave_addr = 0x20,
    .i2c_freq_mode = SENSOR_I2C_MODE_FAST,
    .addr_type = CAMERA_I2C_WORD_ADDR,
    .sensor_id_info =
    {
      .sensor_id_reg_addr = 0x0000,
      .sensor_id = 0x3103,
    },
```

2) Configure sensor output information

The following sensor formats are supported: RAW8, RAW10, RAW12 and YUV422-8.

```
    .sensor_output =
    {
      .output_format = SENSOR_BAYER,
      .connection_mode = SENSOR_MIPI_CSI,
      .raw_output = SENSOR_10_BIT_DIRECT,
#ifndef FLIP_MIRROR
      .filter_arrangement = SENSOR_GRBG,
#else
      .filter_arrangement = SENSOR_GBRG,
#endif
    },
```

- Configure the register

  It mainly includes: init register array, start/stop register array, resolution register array.

```
  .res_settings_array =
  {
    .reg_settings =
    {
      /* Res 0 */
      {
        .reg_setting_a = RES0_REG_ARRAY,
        .addr_type = CAMERA_I2C_WORD_ADDR,
        .data_type = CAMERA_I2C_WORD_DATA,
        .delay = 0,
      },
#ifndef DISABLE_RES1_TO_USE_PDAF_IN_VIDEO_OR_NOZSL_MODE
      /* Res 1 */
      {
        .reg_setting_a = RES1_REG_ARRAY,
        .addr_type = CAMERA_I2C_WORD_ADDR,
        .data_type = CAMERA_I2C_WORD_DATA,
        .delay = 0,
      },
#endif
```

```
  .out_info_array =
  {
    .out_info =
    {
      /* Res 0 */
      {
        .x_output = 4632,
        .y_output = 3480,
        .line_length_pclk = 5148,
        .frame_length_lines = 3626,
        .vt_pixel_clk = 560000000,
        .op_pixel_clk = 556800000,
        .binning_factor = 1,
        .min_fps = 4,//7.5
        .max_fps = 30.1,
        .mode = SENSOR_DEFAULT_MODE,
        .offset_x = 0,
        .offset_y = 0,
        .scale_factor = 0,
        .is_pdaf_supported = 1,//when the pdaf cal data
into snashot camera,
                              //and the third camera apk c
      },
#ifndef DISABLE_RES1_TO_USE_PDAF_IN_VIDEO_OR_NOZSL_MODE
      /* Res 1 */
      {
        .x_output = 2316,
        .y_output = 1740,
        .line_length_pclk = 5148,
        .frame_length_lines = 3626,
        .vt_pixel_clk = 560000000,
        .op_pixel_clk = 556800000,
        .binning_factor = 1,
        .min_fps = 7.5,
        .max_fps = 30.1,
        .mode = SENSOR_DEFAULT_MODE,
        .offset_x = 0,
        .offset_y = 0,
        .scale_factor = 0,
        .is_pdaf_supported = 0,
      },
#endif
```

NOTE

Resolution/CLK/Frame should correspond with register configuration.

- line_length_pclk: Width including blanking.
- frame_length_lines: Height including blanking.
- vt_pixel_clk(video timing clk value): Virtual clock value used for calculating shutter time, and used by AEC for correcting banding artifacts.
- vt_pixel_clk = line_length_pclk * frame_length_lines * frame rate.
- op_pixel_clk represents how much data comes out of the camera over MIPI lanes to set the VFE clock.
- op_pixel_clk = (total data rate from sensor)/bits-per-pixel.

For example, if the MIPI DDR clock value (speed of the clock lane of the MIPI camera sensor) is 300MHz,

and the sensor transmits on 4 lanes, each lane has a 600MHz data rate; thus, the total data rate is 2400MHz. For 10 bits per pixel Bayer data, this translates to the op_pixel_clk value of 2400/10 = 240MHz. These values must be filled in accordance with the sensor specifications. These values can be calculated based on the register settings configured for the camera sensor.

- Configure lane number

```
.csi_params =
{
  .lane_cnt = 4,
  .settle_cnt = 0x14,
  .is_csi_3phase = 0,
},
```

3) Configure power on/off sequence

- Power on sequence

S5K3P3 datasheet describes the power-on sequence as below:

7.1 Power-Up Sequence

The digital and analog supply voltages can be powered up in any order, e.g., VDDD/VDDIO then VDDA/VPIX or VDDA/VPIX/VDDIO then VDDD.

On power up, RSTN (XSHUTDOWN) should be low when the power supplies are brought up, then the sensor module will go into hardware standby mode. As long as RSTN is low and VDDD is down, the sensor module stays in hardware standby mode.

The assertion of RSTN ensures that the CCI/SPI register values are initialized correctly to their default values.

When RSTN will go "high", all PADs will exit from FAIL-SAFE mode, and switch to Normal operating mode.

The MCLK clock can either be initially low and then enabled during software standby mode or MCLK can be a free running clock.

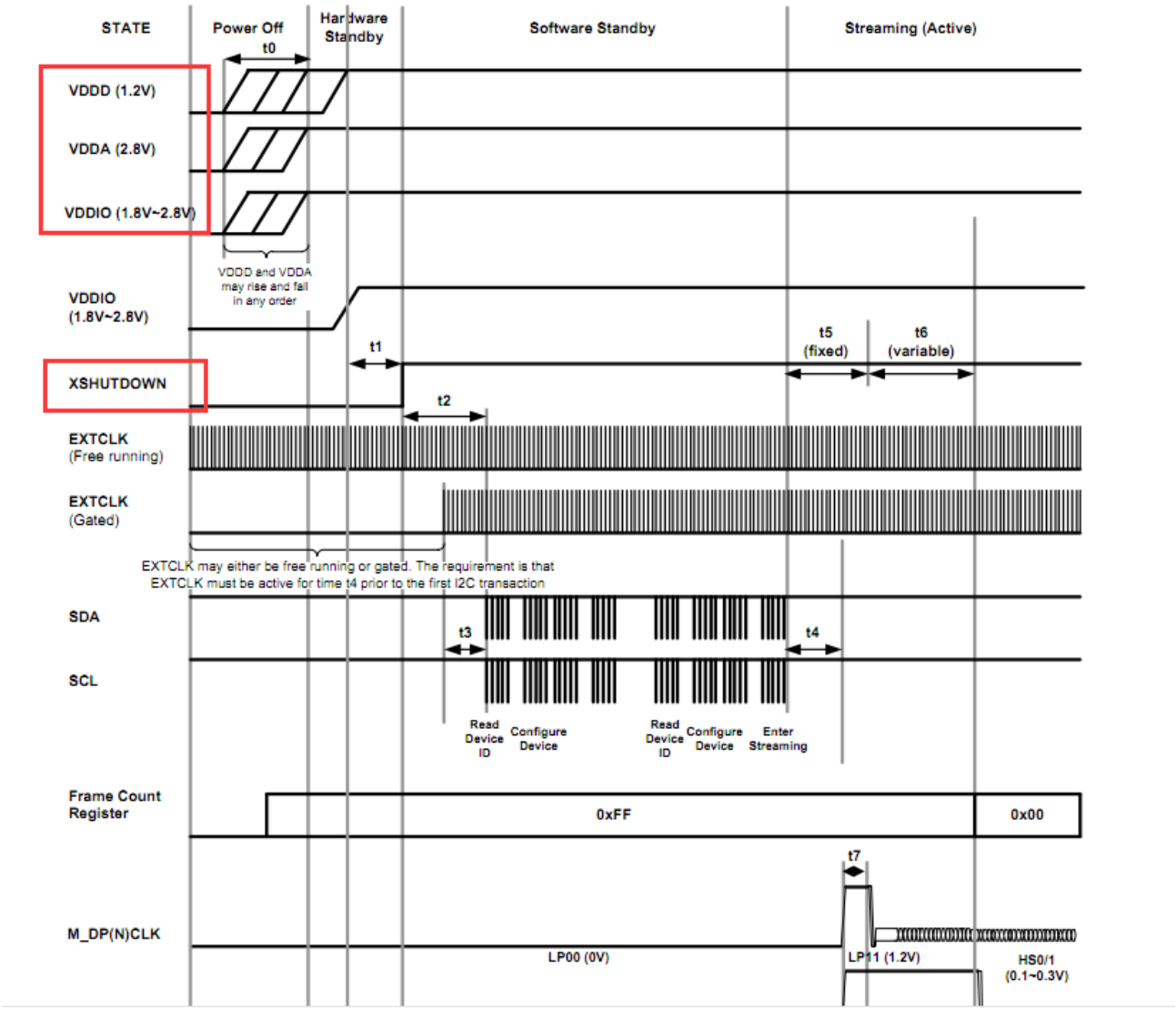


Figure 2: S5K3P3 Power on Sequence

Power on AVDD first, and then IOVDD and DVDD. Wait for the time interval indicated (t1), and then pull up XPWRDWN (PWDN).

NOTE

Sometimes, it is not necessary to strictly conform to the power on sequence, but sometimes it is a must. So it is highly recommended to configure the power on sequence according to the datasheet, otherwise, the camera may fail to be brought up.

- Power on sequence in code

Path:
vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/sensor/libs/s5k3p3/s5k3p3_lib.h

```
.power_setting_array =
{
  .power_setting_a =
  {
    {
      .seq_type = CAMERA_POW_SEQ_GPIO,
      .seq_val = CAMERA_GPIO_RESET,
      .config_val = GPIO_OUT_LOW,
      .delay = 1,
    },
    {
      .seq_type = CAMERA_POW_SEQ_GPIO,
      .seq_val = CAMERA_GPIO_STANDBY,
      .config_val = GPIO_OUT_LOW,
      .delay = 1,
    },
    {
      .seq_type = CAMERA_POW_SEQ_VREG,
      .seq_val = CAMERA_VANA,
      .config_val = 0,
      .delay = 1,
    },
    {
      .seq_type = CAMERA_POW_SEQ_VREG,
      .seq_val = CAMERA_VIO,
      .config_val = 0,
      .delay = 1,
    },
    {
      .seq_type = CAMERA_POW_SEQ_VREG,
      .seq_val = CAMERA_VDIG,
      .config_val = 0,
      .delay = 5,
    },
    {
      .seq_type = CAMERA_POW_SEQ_GPIO,
      .seq_val = CAMERA_GPIO_VDIG,
      .config_val = GPIO_OUT_HIGH,
      .delay = 5,
    },
```

```
{
  .seq_type = CAMERA_POW_SEQ_CLK,
  .seq_val = CAMERA_MCLK,
  .config_val = 24000000,
  .delay = 1,
},
{
  .seq_type = CAMERA_POW_SEQ_GPIO,
  .seq_val = CAMERA_GPIO_RESET,
  .config_val = GPIO_OUT_HIGH,
  .delay = 10,
},
{
  .seq_type = CAMERA_POW_SEQ_GPIO,
  .seq_val = CAMERA_GPIO_STANDBY,
  .config_val = GPIO_OUT_HIGH,
  .delay = 20,
},
{
  .seq_type = CAMERA_POW_SEQ_VREG,
  .seq_val = CAMERA_VAF,
  .config_val = 0,
  .delay = 5,
},
},
size = 10,
```

- Power off sequence

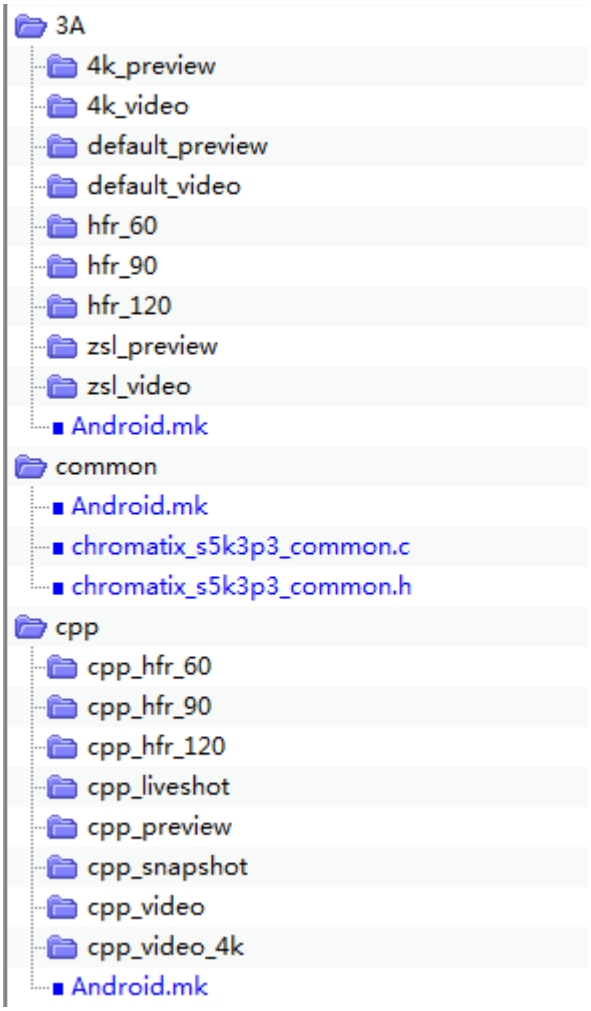
Please refer to the power on sequence.

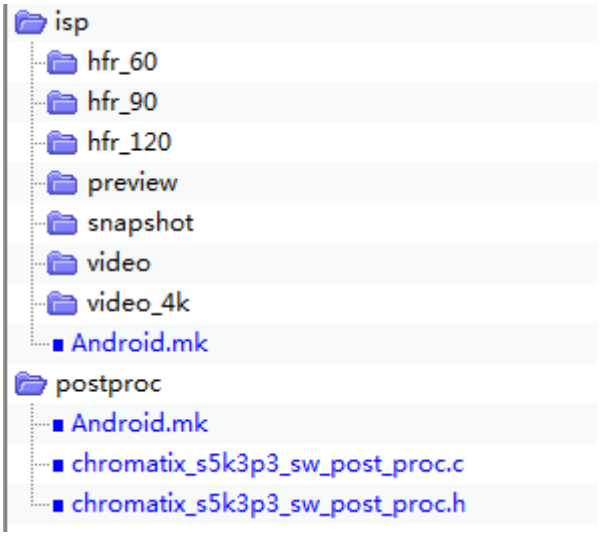
5.2.2. Chromatix Code

1. Adding chromatix code

Path:
vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/chromatix/0310/chromatix_s5k3p3

Compile the codes in the above path to generate .so files, and then configure the .so files by:
vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/configs/s5k3p3_chromatix.xml





2. Configure to use chromatix code

- Add the file in the following path:
  vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/configs/s5k3p3_chromatix.xml

- ISP/CPP/3A code should be configured corresponding to sensor resolution index in the path below:
  vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/sensor/libs/s5k3p3/s5k3p3_lib.h

```
<ChromatixConfigurationRoot>
  <CommonChromatixInfo>
    <ChromatixName special_mode_mask="0">
      <ISPCommon>s5k3p3_common</ISPCommon>
      <PostProc>s5k3p3_postproc</PostProc>
    </ChromatixName>
  </CommonChromatixInfo>
  <ResolutionChromatixInfo>
    <ChromatixName sensor_resolution_index="0" special_mode_mask="0">
      <ISPPreview>s5k3p3_snapshot</ISPPreview>
      <ISPSnapshot>s5k3p3_snapshot</ISPSnapshot>
      <ISPVideo>s5k3p3_default_video</ISPVideo>
      <CPPPreview>s5k3p3_cpp_preview</CPPPreview>
      <CPPSnapshot>s5k3p3_cpp_snapshot</CPPSnapshot>
      <CPPVideo>s5k3p3_cpp_video</CPPVideo>
      <CPPLiveshot>s5k3p3_cpp_liveshot</CPPLiveshot>
      <A3Preview>s5k3p3_zsl_preview_3a</A3Preview>
      <A3Video>s5k3p3_zsl_video_3a</A3Video>
    </ChromatixName>
    <ChromatixName sensor_resolution_index="1" special_mode_mask="0">
      <ISPPreview>s5k3p3_snapshot</ISPPreview>
      <ISPSnapshot>s5k3p3_snapshot</ISPSnapshot>
      <ISPVideo>s5k3p3_default_video_4k</ISPVideo>
      <CPPPreview>s5k3p3_cpp_preview</CPPPreview>
      <CPPSnapshot>s5k3p3_cpp_snapshot</CPPSnapshot>
      <CPPVideo>s5k3p3_cpp_video_4k</CPPVideo>
      <CPPLiveshot>s5k3p3_cpp_liveshot</CPPLiveshot>
      <A3Preview>s5k3p3_4k_preview_3a</A3Preview>
      <A3Video>s5k3p3_4k_video_3a</A3Video>
    </ChromatixName>
```

- Modify the file in the following path:
  vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/configs/Android.mk

```
include $(CLEAR_VARS)
LOCAL_MODULE:= s5k3p3_chromatix.xml
LOCAL_MODULE_CLASS := EXECUTABLES
LOCAL_SRC_FILES := s5k3p3_chromatix.xml
LOCAL_MODULE_TAGS := optional
LOCAL_MODULE_PATH := $(TARGET_OUT_VENDOR)/etc/camera
LOCAL_MODULE_OWNER := qti
include $(BUILD_PREBUILT)
```

5.2.3. Driver Configuration

5.2.3.1. xml configuration

1) Path:

vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/configs/msm8953_camera.xml

2) Main parameters:

- **CameraId: 0, 1, 2**

CameraId corresponds to *qcom,camera@0*, *qcom,camera@1* and *qcom,camera@2* in *kernel/msm-4.9/arch/arm/boot/dts/qcom/msm8953-camera-sensor-mtp.dtsi*.

- **CSIDCore: 0, 1, 2**

CSIDCore corresponds to *qcom,csid-sd-index* in *kernel/msm-4.9/arch/arm/boot/dts/qcom/msm8953-camera-sensor-mtp.dtsi*.

NOTE

SC600Y does not support CSIDCore 2, so CSIDCore and csid-sd-index share index 1 on camera ID 1&2.

- **cci-master: 0, 1**

cci-master corresponds to the cci resource the sensor used, and it is defined in *kernel/msm-4.9/arch/arm/boot/dts/qcom/msm8953-camera-sensor-mtp.dtsi*.

```
camera0: qcom,camera@0 {
        cell-index = <0>;
        compatible = "qcom,camera";
        reg = <0x0>;
        qcom,csiphy-sd-index = <0>;
        qcom,csid-sd-index = <0>;
        qcom,mount-angle = <270>;
        qcom,eeprom-src = <&eeprom0>;
        qcom,actuator-src = <&actuator0>;
        cam_vio-supply = <&pm8953_l6>;
        cam_vdig-supply = <&pm8953_l2>;
        cam_vaf-supply = <&pm8953_l17>;
        cam_vana-supply = <&pm8953_l22>;
        qcom,cam-vreg-name = "cam_vio", "cam_vdig", "cam_vaf",
                             "cam_vana";
        qcom,cam-vreg-min-voltage = <0 1100000 2850000 2800000>;
        qcom,cam-vreg-max-voltage = <0 1100000 2850000 2800000>;
        qcom,cam-vreg-op-mode = <0 105000 100000 80000>;
        pinctrl-names = "cam_default", "cam_suspend";
        pinctrl-0 = <&cam_sensor_mclk0_default
                     &cam_sensor_rear_default
                     &cam_sensor_rear_vana>;
        pinctrl-1 = <&cam_sensor_mclk0_sleep &cam_sensor_rear_sleep
                     &cam_sensor_rear_vana_sleep>;
        gpios = <&tlmm 26 0>,
                <&tlmm 40 0>,
                <&tlmm 39 0>,
                <&tlmm 3 0>;
        qcom,gpio-reset = <1>;
        qcom,gpio-standby = <2>;
        qcom,gpio-vdig = <3>;
        qcom,gpio-req-tbl-num = <0 1 2 3>;
        qcom,gpio-req-tbl-flags = <1 0 0 0>;
        qcom,gpio-req-tbl-label = "CAMIF_MCLK0",
                "CAM_RESET0",
                "CAM_STANDBY0",
                "CAM_VDIG";
        qcom,sensor-position = <0>;
        qcom,sensor-mode = <0>;
        qcom,cci-master = <0>;
        status = "ok";
        clocks = <&clock_gcc clk_mclk0_clk_src>,
                 <&clock_gcc clk_gcc_camss_mclk0_clk>;
```

3) Position

BACK, FRONT and BACK_AUX
(BACK_AUX can be used as a depth camera for dual-camera devices. Generally, it will not be used.)

4) MountAngle: 0, 90, 180, 270.

5) Configure csi_lane_mask, csi_lane_assign and combo_mode:

- **csi_lane_mask:** This 8-bit field indicates which MIPI lanes are valid and enabled. When a single camera is connected on the combo PHY, the value would be interpreted as shown in the following table.

Table 1: Description of csi_lane_mask Bit Fields

Bit Position	Represents
7:5	Reserved
4	Is data lane 3 valid? 0 No 1 Yes
3	Is data lane 2 valid? 0 No 1 Yes
2	Is data lane 1 valid? 0 No 1 Yes
1	Is clock lane valid? 0 No 1 Yes **Note**: this should always be 1.
0	Is data lane 0 valid? 0 No 1 Yes

- **csi_lane_assign:** Sometimes customers' hardware may be designed with different pin mapping compared to the module chipset's reference pin map for camera data lanes, for example, sensor data lane 0 may be connected to MSM data lane 4. The csi_lane_assign parameter can be configured to address such cases. This is a 16-bit value, with the meaning of each bit field presented as follows:

Table 2: Description of csi_lane_mask Bit Fields

Bit Position	Represents
15:12	MSM side PHY lane number, where the sensor's data lane 3 is connected
11:8	MSM side PHY lane number, where the sensor's data lane 2 is connected
7:4	MSM side PHY lane number, where the sensor's data lane 1 is connected
3:0	MSM side PHY lane number, where the sensor's data lane 0 is connected

NOTE

Lane 1 is reserved for the clock. Customers should not use this lane for mapping any data lanes.

```
ComboMode :
    Flag to enable combo mode.
    This flag is enabled if multiple sensors are using same CSI-PHY receiver
```

```
<CameraConfigurationRoot>
  <CameraModuleConfig>
    <CameraId>0</CameraId>
    <SensorName>s5k3p3</SensorName>
    <ActuatorName>dw9763</ActuatorName>
    <EepromName>dw9763_2d</EepromName>
    <FlashName>pmic</FlashName>
    <ChromatixName>s5k3p3_chromatix</ChromatixName>
    <ModesSupported>1</ModesSupported>
    <Position>BACK</Position>
    <MountAngle>270</MountAngle>
    <CSIInfo>
      <CSIDCore>0</CSIDCore>
      <LaneMask>0x1F</LaneMask>
      <LaneAssign>0x4320</LaneAssign>
      <ComboMode>0</ComboMode>
    </CSIInfo>
    <LensInfo>
      <FocalLength>3.57</FocalLength>
      <FNumber>2.0</FNumber>
      <TotalFocusDistance>1.2</TotalFocusDistance>
      <HorizontalViewAngle>64.7</HorizontalViewAngle>
      <VerticalViewAngle>48.5</VerticalViewAngle>
      <MinFocusDistance>0.1</MinFocusDistance>
    </LensInfo>
  </CameraModuleConfig>
```

5.2.3.2. Add All Necessary .so Files

- Make entry of the new libraries in the file to include in the build.

Path: *vendor/qcom/proprietary/common/config/device-vendor.mk*

```
MM_CAMERA += s5k3p3_chromatix.xml
MM_CAMERA += libmmcamera_s5k3p3
MM_CAMERA += libchromatix_s5k3p3_default_video
MM_CAMERA += libchromatix_s5k3p3_hfr_120_3a
MM_CAMERA += libchromatix_s5k3p3_hfr_90_3a
MM_CAMERA += libchromatix_s5k3p3_hfr_90
MM_CAMERA += libchromatix_s5k3p3_cpp_liveshot
MM_CAMERA += libchromatix_s5k3p3_zsl_video_3a
MM_CAMERA += libchromatix_s5k3p3_preview
MM_CAMERA += libchromatix_s5k3p3_default_video_3a
MM_CAMERA += libchromatix_s5k3p3_cpp_hfr_120
MM_CAMERA += libchromatix_s5k3p3_default_preview_3a
MM_CAMERA += libchromatix_s5k3p3_cpp_hfr_60
MM_CAMERA += libchromatix_s5k3p3_snapshot
MM_CAMERA += libchromatix_s5k3p3_cpp_snapshot
MM_CAMERA += libchromatix_s5k3p3_hfr_60
MM_CAMERA += libchromatix_s5k3p3_cpp_preview
MM_CAMERA += libchromatix_s5k3p3_common
MM_CAMERA += libchromatix_s5k3p3_postproc
MM_CAMERA += libchromatix_s5k3p3_cpp_hfr_90
MM_CAMERA += libchromatix_s5k3p3_zsl_preview_3a
MM_CAMERA += libchromatix_s5k3p3_hfr_60_3a
MM_CAMERA += libchromatix_s5k3p3_cpp_video
MM_CAMERA += libchromatix_s5k3p3_hfr_120
MM_CAMERA += libchromatix_s5k3p3_default_video_4k
MM_CAMERA += libchromatix_s5k3p3_4k_video_3a
MM_CAMERA += libchromatix_s5k3p3_cpp_video_4k
MM_CAMERA += libchromatix_s5k3p3_4k_preview_3a
MM_CAMERA += libactuator_dw9763
```

5.2.4. Summary

- Add sensor driver and chromatix code
- Configure power supply in kernel and power-on/off sequence
- Configure the following two .xml files:
  - ✓ *vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/configs/msm8953_camera.xml*
  - ✓ *vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/configs/s5k3p3_chromatix.xml*
- Add compile file *vendor/qcom/proprietary/common/config/device-vendor.mk*

5.3. YUV Sensor Configuration

The above configurations (***Chapter 5.1*** and ***Chapter 5.2***) are based on Bayer sensor. Part of configuration is different when the sensor output type is YUV, and the main differences are listed below:

1. Vendor driver configuration about sensor_output is different.

Reference:
vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/sensor/libs/xxx/xxx_lib.h

```
.sensor_output =
{
  .output_format = SENSOR_YCBCR,
  .connection_mode = SENSOR_MIPI_CSI,
  .raw_output = SENSOR_8_BIT_DIRECT,
  .filter_arrangement = SENSOR_YUYV,
},
```

2. There is no chromatix code, so it is no need to configure *xxx_chromatix.xml*. There is also no need to add ChromatixName in *msm8953_camera.xml*.

6 Add AF Actuator Driver

This chapter provides guidelines to customers who write their own AF actuator driver.

6.1. Updating a Device Tree File

1. In the target's camera .dtsi file, e.g., *msm8953-camera-sensor-mtp.dtsi*, add an entry for the actuator node and assign qcom,actuator-src with actuator node.

Path: *kernel/msm-4.9/arch/arm/boot/dts/qcom/msm8953-camera-sensor-mtp.dtsi*

```
actuator0: qcom,actuator@0 {
        cell-index = <0>;
        reg = <0x0>;
        compatible = "qcom,actuator";
        qcom,cci-master = <0>;
        cam_vaf-supply = <&pm8953_l17>;
        qcom,cam-vreg-name = "cam_vaf";
        qcom,cam-vreg-min-voltage = <2850000>;
        qcom,cam-vreg-max-voltage = <2850000>;
        qcom,cam-vreg-op-mode = <80000>;
};
```

```
camera0: qcom,camera@0 {
        cell-index = <0>;
        compatible = "qcom,camera";
        reg = <0x0>;
        qcom,csiphy-sd-index = <0>;
        qcom,csid-sd-index = <0>;
        qcom,mount-angle = <270>;
        qcom,actuator-src = <&actuator0>;
        cam_vdig-supply = <&pm8953_l2>;
```

2. Please note that the power supply of AF is specified together with the camera sensor and it is the fourth entry in the list of each vreg name, type, min-voltage, max-voltage and op-mode.

```
camera0: qcom,camera@0 {
        cell-index = <0>;
        compatible = "qcom,camera";
        reg = <0x0>;
        qcom,csiphy-sd-index = <0>;
        qcom,csid-sd-index = <0>;
        qcom,mount-angle = <270>;
        qcom,eeprom-src = <&eeprom0>;
        qcom,actuator-src = <&actuator0>;
        cam_vio-supply = <&pm8953_l6>;
        cam_vdig-supply = <&pm8953_l2>;
        cam_vaf-supply = <&pm8953_l17>;
        cam_vana-supply = <&pm8953_l22>;
        qcom,cam-vreg-name = "cam_vio", "cam_vdig", "cam_vaf",
                             "cam_vana";
        qcom,cam-vreg-min-voltage = <0 1100000 2850000 2800000>;
        qcom,cam-vreg-max-voltage = <0 1100000 2850000 2800000>;
        qcom,cam-vreg-op-mode = <0 105000 100000 80000>;
        pinctrl-names = "cam_default", "cam_suspend";
        pinctrl-0 = <&cam_sensor_mclk0_default
                     &cam_sensor_rear_default
```

6.2. Add AF Actuator User Space Driver

1. Adding AF actuator files

vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/actuator_libs/<actuator>/
├─── Android.mk
├─── <actuator>_actuator.c
└─── <actuator>_actuator.h

2. Adding AF algorithm tuning files

About SC600Y&SC600T, there is no individual chromatix code. It is contained in sensor 3A file.

vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/chromatix/0310/chromatix_s5k3p3/3A/
Such as: *default_preview/chromatix_s5k3p3_default_preview_dw9763.h*

```
/* Header Info */
{
   0x0309, /* Version */
   "s5k3p3", /* Module Name */
   "dw9763", /* Actuator Name */
   ACT_MAIN_CAM_0, /* Cam Name */
},
```

3. Update msm8953_camera.xml configuration

vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/configs/msm8953_camera.xml

```
<CameraConfigurationRoot>
  <CameraModuleConfig>
    <CameraId>0</CameraId>
    <SensorName>s5k3p3</SensorName>
    <ActuatorName>dw9763</ActuatorName>
    <EepromName>dw9763_2d</EepromName>
```

6.3. Updating the Device Tree

In our camera target .dtsi file, the actuator node needs to be added. And the AF power supply is set together with that of the sensor, and the *vreg-name/type/min -voltage/max-voltage/op-mode* needs to be set.

```
actuator0: qcom,actuator@0 {
    cell-index = <0>;
    reg = <0x0>;
    compatible = "qcom,actuator";
    qcom,cci-master = <0>;
    cam_vaf-supply = <&pm8953_l17>;
    qcom,cam-vreg-name = "cam_vaf";
    qcom,cam-vreg-min-voltage = <2850000>;
    qcom,cam-vreg-max-voltage = <2850000>;
    qcom,cam-vreg-op-mode = <80000>;
};
```

```
qcom,camera@0 {
    cell-index = <0>;
    compatible = "qcom,camera";
    reg = <0x0>;
    qcom,csiphy-sd-index = <0>;
    qcom,csid-sd-index = <0>;
    qcom,mount-angle = <270>;
    qcom,led-flash-src = <&led_flash0>;
    qcom,eeprom-src = <&eeprom0>;
    qcom,actuator-src = <&actuator0>;
    cam_vio-supply = <&pm8953_l6>;
    cam_vdig-supply = <&pm8953_l2>;      //not used
    cam_vaf-supply = <&pm8953_l17>;
    cam_vana-supply = <&pm8953_l22>;
    qcom,cam-vreg-name = "cam_vio", "cam_vdig", "cam_vaf",
                         "cam_vana";
    qcom,cam-vreg-min-voltage = <0 1100000 2850000 2800000>;
    qcom,cam-vreg-max-voltage = <0 1100000 2850000 2800000>;
    qcom,cam-vreg-op-mode = <0 105000 100000 80000>;
```

7 Add EEPROM Driver

This chapter provides guidelines to customers who write their own EEPROM driver.

7.1. Updating a Device Tree File

In the target camera .dtsi file, e.g., *msm8953-camera-sensor-mtp.dtsi*, add an entry for EEPROM node and assign qcom,eeprom-src with EEPROM node.

Path: *kernel/msm-4.9/arch/arm/boot/dts/qcom/msm8953-camera-sensor-mtp.dtsi*

```
eeprom0: qcom,eeprom@0 {
        cell-index = <0>;
        compatible = "qcom,eeprom";
        qcom,cci-master = <0>;
        reg = <0xb0>;
        cam_vio-supply = <&pm8953_l6>;
        cam_vdig-supply = <&pm8953_l2>;//not use
        cam_vaf-supply = <&pm8953_l17>;
        cam_vana-supply = <&pm8953_l22>;
        qcom,cam-vreg-name = "cam_vio", "cam_vdig", "cam_vaf", "cam_vana";
        qcom,cam-vreg-min-voltage = <0 1100000 2850000 2800000>;
        qcom,cam-vreg-max-voltage = <0 1100000 2850000 2800000>;
        qcom,cam-vreg-op-mode = <0 105000 100000 80000>;
        pinctrl-names = "cam_default", "cam_suspend";
        pinctrl-0 = <&cam_sensor_mclk0_default
                     &cam_sensor_rear_default
                     &cam_sensor_rear_vana>;
        pinctrl-1 = <&cam_sensor_mclk0_sleep &cam_sensor_rear_sleep
                     &cam_sensor_rear_vana_sleep>;
        gpios = <&tlmm 26 0>,
                <&tlmm 40 0>,
                <&tlmm 39 0>,
                <&tlmm 3 0>;
        qcom,gpio-reset = <1>;
        qcom,gpio-standby = <2>;
        qcom,gpio-vdig = <3>;
        qcom,gpio-req-tbl-num = <0 1 2 3>;
        qcom,gpio-req-tbl-flags = <1 0 0 0>;
        qcom,gpio-req-tbl-label = "CAMIF_MCLK0",
                "CAM_RESET0",
                "CAM_STANDBY0",
                "CAM_VDIG";
        status = "ok";
        clocks = <&clock_gcc clk_mclk0_clk_src>,
                     <&clock_gcc clk_gcc_camss_mclk0_clk>;
        clock-names = "cam_src_clk", "cam_clk";
        qcom,clock-rates = <19200000 0>;
}
```

```
camera0: qcom,camera@0 {
        cell-index = <0>;
        compatible = "qcom,camera";
        reg = <0x0>;
        qcom,csiphy-sd-index = <0>;
        qcom,csid-sd-index = <0>;
        qcom,mount-angle = <270>;
        qcom,eeprom-src = <&eeprom0>;
        qcom,actuator-src = <&actuator0>;
        cam_vio-supply = <&pm8953_l6>;
        cam_vdig-supply = <&pm8953_l2>;
        cam_vaf-supply = <&pm8953_l17>;
        cam_vana-supply = <&pm8953_l22>;
```

7.2. Updating a Sensor Driver File

Considering the *s5k3p3* driver as an example, the eeprom_name field should be updated to *msm8953_camera.xml*.

Path:
vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/configs/msm8953_camera.xml

```
<CameraModuleConfig>
  <CameraId>0</CameraId>
  <SensorName>s5k3p3</SensorName>
  <ActuatorName>dw9763</ActuatorName>
  <EepromName>dw9763_2d</EepromName>
  <FlashName>pmic</FlashName>
  <ChromatixName>s5k3p3_chromatix</ChromatixName>
```

7.3. Adding a EEPROM Driver File

The following *<eeprom>.c* file must be added for a new EEPROM driver.

vendor/qcom/proprietary/mm-camera/mm-camera2/media-controller/modules/sensors/eeprom/libs/<eeprom>/
├── Android.mk
├── <eeprom>_eeprom.c
└── <eeprom>_eeprom.h

Any new *<eeprom>.c* file should have the following function pointers mapped and defined in it. Any of these functions not defined in that EEPROM driver must be set to NULL.

```
static eeprom_lib_func_t <eeprom>_lib_func_ptr = {
.get_calibration_items = NULL,
.format_calibration_data = NULL,
.do_af_calibration = NULL,
.do_wbc_calibration = NULL,
.do_lsc_calibration = NULL,
.do_dpc_calibration = NULL,
.get_dpc_calibration_info = NULL,
.get_raw_data = NULL,
};
```

8 LED Flash Driver

This chapter provides guidelines to customers who write their own LED Flash driver. Some important parts of writing LED Flash driver are used here for illustration.

8.1. Updating a Device Tree File

1. In the target camera .dtsi file, e.g., *msm8953-mtp.dtsi*, add an entry for led_flash node and assign qcom,led-flash-src with led_flash node.

```
&cci {
        qcom,camera@0 {
                qcom,led-flash-src = <&led_flash0>;
        };
};
```

2. Depending on the LED Flash hardware, OEMs can decide which type of interface driver to configure. Some LED Flash hardware needs a power supply at input to turn it on/off. For such LED Flash hardware, OEMs can use a PMIC-based LED Flash driver to supply current/power from the PMIC IC. This driver is very simple and just calls PMIC APIs to control the current/power level for different Flash states. Other LED Flash hardware must be programmed with register settings to turn it on/off. For that hardware, OEMs can use either QUP- or I2C-based LED Flash drivers.

Node entry in the device tree file will change based on the type of LED Flash driver (PMIC-based, I2C-based).

For more details and explanation of each field in device tree file, please refer to:
kernel/msm-4.9/Documentation/devicetree/bindings/media/video/msm-camera-flash.txt
kernel/msm-4.9/Documentation/devicetree/bindings/leds/leds-gpio.txt

3. PMIC-based LED Flash driver

PMIC-based LED flash driver is located in *kernel/msm-4.9/arch/arm/boot/dts/qcom/pmi632.dtsi*, such as *pmi632.dtsi*. It defines the source of flash, parameters and handle.

```
&soc {
        led_flash0: qcom,camera-flash {
                cell-index = <0>;
                compatible = "qcom,camera-flash";
                qcom,flash-type = <1>;
                qcom,flash-source = <&pmi632_flash0 &pmi632_flash1>;
                qcom,torch-source = <&pmi632_torch0 &pmi632_torch1>;
                qcom,switch-source = <&pmi632_switch0>;
        };
};
```

```
pmi632_flash0: qcom,flash_0 {
        label = "flash";
        qcom,led-name = "led:flash_0";
        qcom,max-current = <1500>;
        qcom,default-led-trigger = "flash0_trigger";
        qcom,id = <0>;
        qcom,current-ma = <1000>;
        qcom,duration-ms = <1280>;
        qcom,ires-ua = <12500>;
        qcom,hdrm-voltage-mv = <400>;
        qcom,hdrm-vol-hi-lo-win-mv = <100>;
};

pmi632_flash1: qcom,flash_1 {
        label = "flash";
        qcom,led-name = "led:flash_1";
        qcom,max-current = <1500>;
        qcom,default-led-trigger = "flash1_trigger";
        qcom,id = <1>;
        qcom,current-ma = <1000>;
        qcom,duration-ms = <1280>;
        qcom,ires-ua = <12500>;
        qcom,hdrm-voltage-mv = <400>;
        qcom,hdrm-vol-hi-lo-win-mv = <100>;
};

pmi632_torch0: qcom,torch_0 {
        label = "torch";
        qcom,led-name = "led:torch_0";
        qcom,max-current = <500>;
        qcom,default-led-trigger = "torch0_trigger";
        qcom,id = <0>;
        qcom,current-ma = <300>;
        qcom,ires-ua = <12500>;
        qcom,hdrm-voltage-mv = <400>;
        qcom,hdrm-vol-hi-lo-win-mv = <100>;
};

pmi632_torch1: qcom,torch_1 {
        label = "torch";
        qcom,led-name = "led:torch_1";
        qcom,max-current = <500>;
        qcom,default-led-trigger = "torch1_trigger";
```

9 Troubleshooting

9.1. Check Log

1. The correct log

- Probe succeeded

```
[    21.038576] msm_cci_init:1426: hw_version = 0x10020005
[    21.038882] s5k3p3 probe succeeded
[    21.040163] msm_pcm_volume_ctl_get substream runtime not found
```

2. The error log

- If the sensor failed to communicate with the slave device, it is possible that the I2C address is incorrect or the slave device does not work.

```
MASTER_0 error 0x10000000
msm_cci_i2c_read:955 read_words = 0, exp words = 1
msm_cci_i2c_read_bytes:1038 failed rc -22
msm_camera_cci_i2c_read: line 45 rc = -22
```

- If ID matching failed and there is no bus error as shown above, it means that the read ID is different from the configured ID.

```
msm_sensor_match_id: s5k3p3: read id failed
msm_sensor_check_id:1372 match id failed rc -22
s5k3p3 power up failed
```

- If sensor failed to power up, it means that there are some problems in the power up setting. The address of I2C, dts, the pin configuration of the vendor driver and the sequence of power should be checked.

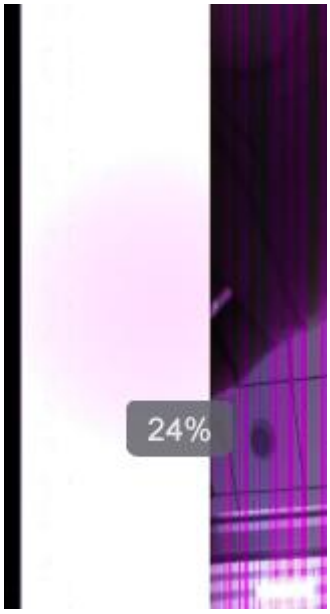
- The following log indicates a failure in getting stream. The possible causes include incorrect register-configuration,problematic MIPI signal and improper lane_cnt in xxx_lib.h

```
Kernel log:
msm_private_ioctl:Notifying subdevs about potential sof freeze
MSM-SENSOR-INIT msm_sensor_init_subdev_ioctl:121 default
```

```
Logcat log:
E/mm-camera( 316): mct_bus_sof_thread_run: SOF freeze; Sending error message
```

If the first frame shows crash and then return, and the log shows ERROR_CRC, ERROR_PHY_DL0_FIFO_OVERFLOW error, please refer to the hardware to find the CRC and DL0 error. Generally, please check settle_cnt value in xxx_lib.h. log as shown below:

```
msm_csid_irq CSID0_IRQ_STATUS_ADDR = 0x1100033
```



3. Settle_cnt calculate

settle_cnt: Also known as settle count. This value must be configured, based on the sensor's output characteristics, to ensure that the sensor's PHY transmitter does not have sync issues with the MSM's PHY receiver.

For 28nm and smaller MSM parts, please use the following formula to calculate settle count:

settle_cnt = T(HS_SETTLE)_avg /T(TIMER_CLK)

- where T(HS_SETTLE)_avg = (T(HS_SETTLE)_min + T(HS_SETTLE)_max) / 2, as indicated by sensor datasheet.
- TIMER_CLK refers to the operating frequency of the PHY interface to which the camera sensor is connected (for example, CAMSS_PHY0_CSI0PHYTIMER_CLK for PHY0).
- T(TIMER_CLK) is the duration of a clock cycle, if the operating frequency is equal to TIMER_CLK, and is represented in Nano second unit. For example, T(TIMER_CLK) for TIMER_CLK 200 MHz is (1 * (10^9))/(200 * (10^6)) = 5ns.

10 Appendix A Reference

Table 1: Terms and Abbreviations

Abbreviation	Description
AF	Auto Focus
CSIPHY	Camera Serial Interface Phy Layer
DT	Device Tree
EEPROM	Electrically Erasable Programmable Read-Only Memory
EVB	Evaluation board
GPIO	General Purpose Input Output
I2C	Inter-integrated Circuit
LDO	Low Dropout Regulator
MIPI	Mobile Industry Processor Interface
PMIC	Power Management IC