Camera Porting Guide

Revision	Date	Description
V1.0	2016/07/01	CAMERA PORTING GUIDE

Archermind

2016/7/20

Contents

1. Introduction

The introduction could help user to porting the camera.

2. M camera driver code path

2.1. Config file

- kernel-3.18/arch/arm64/configs/<$PROJECT>_debug_defconfig (for Eng load)
- kernel-3.18/arch/arm64/configs/<$PROJECT>_defconfig (for user load)
- device/mediatek/<$PROJECT>/ProjectConfig.mk

2.2. Kernel driver & build Makefile

[By project]

- kernel-3.18/drivers/misc/mediatek/mach/mt6797/<project>/imgsensor/

[By Platform]

- kernel-3.18/drivers/misc/mediatek/imgsensor/src/mt6797/

Priority: project > platform

2.3. Kd_sensorlist.c/kd_sensorlist.h

- /kernel-3.18/drivers/misc/mediatek/imgsensor/src/<platform>/

2.4. Kd_camera_hw.c/kd_camera_hw.h

- kernel-3.18/drivers/misc/mediatek/imgsensor/src/mt6755/camera_project/<project>/camera_hw

2.5. Header files

- kd_imgsensor.h
- kd_imgsensor_errcode.h
- kd_imgsensor_define.h
- kd_camera_feature_id.h
- kd_camera_feature.h
- kd_camera_feature_enum.h

file path:

1) Kernel space

kernel-3.18/drivers/misc/mediatek/imgsensor/inc/kd_imgsensor.h

2) User space

device/mediatek/common/kernel-headers/kd_imgsensor.h

These files must be the same in user and kernel space, otherwise you will encounter the io-ctrl error. Why? One rule: Android should not Access Kernel space Source and Header Files directly.

3. DTS

3.1. Power on template

Add your sensor to PowerOnList @ kd_camera_hw.c
No default power on sequence
kdCISModulePowerOn()

kdCISModulePowerOn()

- hwpoweron()
- hwPowerOn() for regulator(aka LDO)
- mtkcam_gpio_set() for gpio
- mt_set_gpio_mode(), mt_set_gpio_dir() and mt_set_gpio_out()
- hwPowerOn() regulator(aka LDO)

3.2. GPIO

3.2.1. Usage

- To control the PWD/RST pin for camera power on
- To control the external LDO for camera

3.2.2. Kernel 3.18

– Kernel standard
– PINCTRL
- For details, please find kernel-3.18/ Documentation/pinctrl.txts

– Need customized for each project manually
- Ex.kernel-3.18/arch/arm64/boot/dts/amt6797_64_open.dts

3.3. PINCTRL

- pinctrl_state
- Functions

pinctrl_lookup_state()

/* GPIO Pin control*/

```
struct platform_device *cam_plt_dev = NULL;
struct pinctrl *camctrl = NULL;
struct pinctrl_state *cam0_pnd_h = NULL;/* main cam */
struct pinctrl_state *cam0_pnd_l = NULL;
struct pinctrl_state *cam0_rst_h = NULL;
struct pinctrl_state *cam0_rst_l = NULL;
struct pinctrl_state *cam1_pnd_h = NULL;/* sub cam */
struct pinctrl_state *cam1_pnd_l = NULL;
struct pinctrl_state *cam1_rst_h = NULL;
struct pinctrl_state *cam1_rst_l = NULL;
struct pinctrl_state *cam2_pnd_h = NULL;/* main2 cam */
struct pinctrl_state *cam2_pnd_l = NULL;
struct pinctrl_state *cam2_rst_h = NULL;
struct pinctrl_state *cam2_rst_l = NULL;
struct pinctrl_state *cam_ldo_vcama_h = NULL;/* for AVDD */
struct pinctrl_state *cam_ldo_vcama_l = NULL;
struct pinctrl_state *cam_ldo_vcamd_h = NULL;/* for DVDD */
struct pinctrl_state *cam_ldo_vcamd_l = NULL;
struct pinctrl_state *cam_ldo_vcamio_h = NULL;/* for DOVDD */
struct pinctrl_state *cam_ldo_vcamio_l = NULL;
struct pinctrl_state *cam_ldo_vcamaf_h = NULL;/* for AFVDD */
struct pinctrl_state *cam_ldo_vcamaf_l = NULL;
struct pinctrl_state *cam_ldo_sub_vcamd_h = NULL;/* for SUB_DVDD */
struct pinctrl_state *cam_ldo_sub_vcamd_l = NULL;
struct pinctrl_state *cam_ldo_main2_vcamd_h = NULL;/* for MAIN2_DVDD */
struct pinctrl_state *cam_ldo_main2_vcamd_l = NULL;
```

– get specified pinctrl_state descripted in xxx.dts

- ❖ am0_pnd_hGPIO state for PDN of main cam @ high,
- ❖ cam0_pnd_l→GPIO state for PDN of main cam @ low

– pinctrl_select_state()

- ❖ pinctrl_select_state(camctrl, cam0_pnd_l); →set PDN of main cam to low

4. How to customized xxx.dts for each project

4.1. kernel-3.18/arch/arm64/boot/dts/amt6797_64_open.dts

4.2. For camera

– &pio→define states of GPIOs

- ❖ camera_pins_cam0_pnd0
- ❖ camera_pins_cam0_pnd1
- ❖ … all the gpio of camera are listed

– kd_camera_hw1→pinctrl mapping

Path ： kernel-3.18/arch/arm64/boot/dts/amt6797_64_open.dts

```
&kd_camera_hw1 {
	pinctrl-names = "default",
		"cam0_rst0", "cam0_rst1", "cam0_pnd0", "cam0_pnd1",
		"cam1_rst0", "cam1_rst1", "cam1_pnd0", "cam1_pnd1",
		"cam2_rst0", "cam2_rst1", "cam2_pnd0", "cam2_pnd1",
		"cam_ldo_vcama_0", "cam_ldo_vcama_1", "cam_ldo_vcamd_0",
"cam_ldo_vcamd_1",
		"cam_ldo_vcamd2_0", "cam_ldo_vcamd2_1";
	pinctrl-0 = <&camera_pins_default>;
	pinctrl-1 = <&camera_pins_cam0_rst0>;
	pinctrl-2 = <&camera_pins_cam0_rst1>;
	pinctrl-3 = <&camera_pins_cam0_pnd0>;
	pinctrl-4 = <&camera_pins_cam0_pnd1>;
	pinctrl-5 = <&camera_pins_cam1_rst0>;
	pinctrl-6 = <&camera_pins_cam1_rst1>;
	pinctrl-7 = <&camera_pins_cam1_pnd0>;
	pinctrl-8 = <&camera_pins_cam1_pnd1>;
	/* for main2 */
	pinctrl-9 = <&camera_pins_cam2_rst0>;
	pinctrl-10 = <&camera_pins_cam2_rst1>;
	pinctrl-11 = <&camera_pins_cam2_pnd0>;
	pinctrl-12 = <&camera_pins_cam2_pnd1>;
	/* for ldo control by gpio */
	pinctrl-13 = <&camera_pins_cam_ldo_vcama_0>;
	pinctrl-14 = <&camera_pins_cam_ldo_vcama_1>;
	pinctrl-15 = <&camera_pins_cam_ldo_vcamd_0>;
	pinctrl-16 = <&camera_pins_cam_ldo_vcamd_1>;
```

```
        pinctrl-17 = <&camera_pins_cam_ldo_e2_vcamd_0>;
        pinctrl-18 = <&camera_pins_cam_ldo_e2_vcamd_1>;
        status = "okay";
};

&pio {
        camera_pins_cam0_rst0: cam0@0 {
                pins_cmd_dat {
                        pins = <PINMUX_GPIO32__FUNC_GPIO32>;
                        slew-rate = <1>; /*direction 0:in, 1:out*/
                        output-low;/*direction out used only. output_low or high*/
                };
        };
        camera_pins_cam0_rst1: cam0@1 {
                pins_cmd_dat {
                        pins = <PINMUX_GPIO32__FUNC_GPIO32>;
```

5. For external LDO control mt6797

Path ： kernel-3.18/arch/arm64/boot/dts/amt6797_64_open.dts

```
camera_pins_cam_ldo_vcamd_0: cam0@vcamd0 {
        pins_cmd_dat {
            pins = <PINMUX_GPIO63__FUNC_GPIO63>;
            slew-rate = <1>;
            output-low;
        };
    };
    camera_pins_cam_ldo_vcamd_1: cam1@vcamd1 {
        pins_cmd_dat {
            pins = <PINMUX_GPIO63__FUNC_GPIO63>;
            slew-rate = <1>;
            output-high;
        };
    };


&kd_camera_hw1 {
    pinctrl-names = "default",
            "cam0_rst0", "cam0_rst1", "cam0_pnd0", "cam0_pnd1",
            "cam1_rst0", "cam1_rst1", "cam1_pnd0", "cam1_pnd1",
            "cam2_rst0", "cam2_rst1", "cam2_pnd0", "cam2_pnd1",
            "cam_ldo_vcama_0", "cam_ldo_vcama_1", "cam_ldo_vcamd_0",
"cam_ldo_vcamd_1",
            "cam_ldo_vcamd2_0", "cam_ldo_vcamd2_1";
    pinctrl-0 = <&camera_pins_default>;
    pinctrl-1 = <&camera_pins_cam0_rst0>;
    pinctrl-2 = <&camera_pins_cam0_rst1>;
    pinctrl-3 = <&camera_pins_cam0_pnd0>;
    pinctrl-4 = <&camera_pins_cam0_pnd1>;
    pinctrl-5 = <&camera_pins_cam1_rst0>;
    pinctrl-6 = <&camera_pins_cam1_rst1>;
    pinctrl-7 = <&camera_pins_cam1_pnd0>;
    pinctrl-8 = <&camera_pins_cam1_pnd1>;
    /* for main2 */
    pinctrl-9 = <&camera_pins_cam2_rst0>;
    pinctrl-10 = <&camera_pins_cam2_rst1>;
    pinctrl-11 = <&camera_pins_cam2_pnd0>;
    pinctrl-12 = <&camera_pins_cam2_pnd1>;
    /* for ldo control by gpio */
    pinctrl-13 = <&camera_pins_cam_ldo_vcama_0>;
```

```
        pinctrl-14 = <&camera_pins_cam_ldo_vcama_1>;
        pinctrl-15 = <&camera_pins_cam_ldo_vcamd_0>;
        pinctrl-16 = <&camera_pins_cam_ldo_vcamd_1>;
        pinctrl-17 = <&camera_pins_cam_ldo_e2_vcamd_0>;
        pinctrl-18 = <&camera_pins_cam_ldo_e2_vcamd_1>;
        status = "okay";
};
```

Path ： \kernel-3.18\drivers\misc\mediatek\imgsensor\src\mt6797\camera_hw

```
cam_ldo_vcamd_1 = pinctrl_lookup_state(camctrl, "cam_ldo_vcamd_0");


if (pinSetIdx == 2) {
                if (PowerCustList.PowerCustInfo[CUST_MAIN2_AVDD].Gpio_Pin ==
GPIO_UNSUPPORTED) {
                        if (TRUE != _hwPowerOn(pwInfo.PowerType, pwInfo.Voltage)) {
                                PK_ERR("[CAMERA SENSOR] Fail to enable digital power\n");
                                return FALSE;
                        }
                } else {
                        if (mtkcam_gpio_set(pinSetIdx, MAIN2_AVDD,
PowerCustList.PowerCustInfo[CUST_MAIN2_AVDD].Voltage)) {
                                        PK_INFO("[CAMERA CUST_AVDD] set gpio failed!!\n");
                        }
                }
        }



case AVDD:
        case MAIN2_AVDD:
                /*Main & Main2 use same cotrol GPIO */
                PK_DBG("mAVDD_usercounter(%d)\n",mAVDD_usercounter);
                if (Val == 0 && !IS_ERR(cam_ldo_vcama_l)){
                        mAVDD_usercounter --;
                        if(mAVDD_usercounter <= 0)
        {
                        if(mAVDD_usercounter < 0)
                                PK_ERR("Please check AVDD pin control\n");

                        mAVDD_usercounter = 0;
                        pinctrl_select_state(camctrl, cam_ldo_vcama_l);
                }
```

```
        }
        else if (Val == 1 && !IS_ERR(cam_ldo_vcama_h)){
            mAVDD_usercounter ++;
            pinctrl_select_state(camctrl, cam_ldo_vcama_h);
        }
        break;


PowerUp PowerOnList = {
    {
     {SENSOR_DRVNAME_OV23850_MIPI_RAW,
      {
       {SensorMCLK, Vol_High, 0},
       {DOVDD, Vol_1800, 0},
       {AVDD, Vol_2800, 0},
       {DVDD, Vol_1200, 0},
       {AFVDD, Vol_2800, 2},
       {PDN, Vol_Low, 0},
       {PDN, Vol_High, 0},
       {RST, Vol_Low, 0},
       {RST, Vol_High, 5},
       },
      },
```

5.1. PMIC

- download partition error. code 0x10004 when run ubuntu xflash
  sudo apt-get purge modemmanager

5.1.1. Use dct tools for customized

5.1.2. cust.dtsi was generated during

- Compiling
  out/target/product/amt6797_64_open/obj/KERNEL_OBJ/arch/arm64/boot/dts/cust.dtsi

5.1.3. Kernel standard

Regulator

• regulator_get()

- ❖ Get instance of regulator
- ❖ Called @ probe in kd_sensorlist.c

• regulator_set_voltage() & regulator_enable()

- ❖ Set voltage and enable among different regulator
- ❖ Called @_hwPowerOn() in kd_sensorlist.c

• regulator_disable()

- ❖ Turn off regulator
- ❖ Called @_hwPowerDown() in kd_sensorlist.c

Path：out/target/product/amt6797_64_open/obj/KERNEL_OBJ/arch/arm64/boot/dts/cust.dtsi

```
&kd_camera_hw1 {
	vcama-supply = <&mt_pmic_vcama_ldo_reg>;
	vcama_main2-supply = <&mt_pmic_vcama_ldo_reg>;
	vcama_sub-supply = <&mt_pmic_vcama_ldo_reg>;

	vcamaf-supply = <&mt_pmic_vldo28_ldo_reg>;
	vcamaf_main2-supply = <&mt_pmic_vldo28_ldo_reg>;
	vcamaf_sub-supply = <&mt_pmic_vldo28_ldo_reg>;

	vcamd-supply = <&mt_pmic_vcamd_ldo_reg>;
	vcamd_main2-supply = <&mt_pmic_vcamd_ldo_reg>;
	vcamd_sub-supply = <&mt_pmic_vcamd_ldo_reg>;

	vcamio-supply = <&mt_pmic_vcamio_ldo_reg>;
	vcamio_main2-supply = <&mt_pmic_vcamio_ldo_reg>;
	vcamio_sub-supply = <&mt_pmic_vcamio_ldo_reg>;

	status = "okay";
};
```

6. I 2 C BUS

6.1. Also use dct tools for customized and located at cust.dtsi

6.2. &i2cX

Path：out/target/product/amt6797_64_open/obj/KERNEL_OBJ/arch/arm64/boot/dts/cust.dtsi

```
&i2c2 {
	#address-cells = <1>;
	#size-cells = <0>;
	clock-frequency = <400000>;
	mediatek,use-open-drain;
	camera_main@36 {
		compatible = "mediatek,camera_main";
		reg = <0x36>;
		status = "okay";
	};
	camera_main_af@72 {
		compatible = "mediatek,camera_main_af";
		reg = <0x72>;
		status = "okay";
	};
};
```

7. ADD A NEW CAMERA FLOW

7.1. Configure camera sensor hal driver in ProjectConfig.mk

file path device\mediatek\$project\ProjectConfig.mk

Example：

- CUSTOM_HAL_IMGSENSOR = ov23850_mipi_raw
- CUSTOM_HAL_MAIN_IMGSENSOR = ov23850_mipi_raw

7.2. Add camera sensor kernel driver project config (<project>_defconfig)

file path : alps\<kernel>\arch\arm64\configs\<project>_debug_defconfig

Example:

- CONFIG_CUSTOM_KERNEL_IMGSENSOR=" ov23850_mipi_raw s5k3m2_mipi_raw s5k5e2ya_mipi_raw imx258_mipi_raw imx377_mipi_raw s5k2x8_mipi_raw CUSTOM_HAL_MAIN_IMGSENSOR = ov23850_mipi"
- use menuconfig with console to modify xxx_defconfig

7.3. Add camera sensor kernel driver code to the corresponding path

[By project]

- kernel-3.18/drivers/misc/mediatek/mach/mt6755/<project>/ imgsensor/

[By Platform]

- kernel-3.18/drivers/misc/mediatek/imgsensor/src/<platform>/
- Must modify the corresponding makefile

7.3.1 Add image sensor kernel driver folder

[By project]

- kernel-3.18/drivers/misc/mediatek/mach/mt6797/<project>/ imgsensor/Makefile (if it exsit)

Example:

Add the following items

- obj-y += imgsensor/ov23850_mipi_raw/

[By platform]

If the image sensor driver doesn't exist in the ,Project- directory , build system will proceed to

search the driver in {platform} directory. Do not need to modify the makefile.

- kernel-3.18\drivers\misc\mediatek\imgsensor\src\<platform>\Makefile .

7.3.2 Add camera sensor kernel driver code to the corresponding path

[By project]

- kernel-3.18/drivers/misc/mediatek/mach/mt6797/<project>/ imgsensor/

[By Platform]

- kernel-3.18/drivers/misc/mediatek/imgsensor/src/<platform>/

Must modify the corresponding makefile

link you image sensor compiled object

Example:

- obj-y += ov23850_mipi_raw.o

[By project]

- alps/kernel-3.18/drivers/misc/mediatek/mach/mt6755/<project>/ imgsensor/xxxx/Makefile (if it exsit)

[By platform]

- alps\<kernel>\drivers\misc\mediatek\imgsensor\src\<platform>\Makefile

7.3.3 Modify other related files

1 .Makefile

- ProjectConfig.mk xxx_defconfig
- Kernel xxx_defconfig

2 .imgsensor

- CUSTOM_HAL_IMGSENSOR = xxxxxx_xxx
- CUSTOM_KERNEL_IMGSENSOR = xxxxxx_xxx
- CUSTOM_HAL_SUB_IMGSENSOR = xxxxxx_xxx
- CUSTOM_HAL_MAIN_IMGSENSOR = xxxxxx_xxx
- CUSTOM_KERNEL_MAIN_IMGSENSOR = xxxxxx_xxx=
- CUSTOM_KERNEL_SUB_IMGSENSOR = xxxxxx_xxx
- CUSTOM_HAL_IMGSENSOR& CUSTOM_KERNEL_IMGSENSOR mainsub sensor
- CUSTOM_HAL_MAIN_IMGSENSOR& CUSTOM_KERNEL_MAIN_IMGSENSOR main sensor
- CUSTOM_HAL_SUB_IMGSENSOR& CUSTOM_KERNEL_SUB_IMGSENSOR sub sensor

3 .camera

- # lens port sensor dummy AF sensormain or sub module AF IC
- CUSTOM_HAL_LENS = dummy_lens
- CUSTOM_KERNEL_LENS = dummy_lens
- CUSTOM_HAL_MAIN_LENS = dummy_lens

- CUSTOM_HAL_SUB_LENS =
- CUSTOM_KERNEL_MAIN_LENS = dummy_lens
- CUSTOM_KERNEL_SUB_LENS =

- #Flashlight port sensor dummy constant_flashlight
- CUSTOM_KERNEL_FLASHLIGHT=dummy_flashlight
- CUSTOM_HAL_FLASHLIGHT=dummy_flashlight

- #OTPRAW sensorport sensor dummy, OTP driver
- CUSTOM_KERNEL_CAM_CAL=dummy_eeprom
- CUSTOM_HAL_CAM_CAL=dummy_eeprom
- CUSTOM_KERNEL_EEPROM=dummy_eeprom
- CUSTOM_HAL_EEPROM=dummy_eeprom

4.Image Sensor

– kd_imgsensor.h

• #define XXXXXX_SENSOR_ID 0xFFFF

– sensor ID

- #define SENSOR_DRVNAME_XXXXXX_YUV “xxxxxxxxx“ //YUV sensor
- #define SENSOR_DRVNAME_XXXXXX_RAW “xxxxxxxxx“ //RAW sensor

– sensor part no YUV or raw

» xxxxxxmipiraw

– kd_sensorlist.h

- RAW sensor
- UINT32 XXXXXX_RAW_SensorInit(PSENSOR_FUNCTION_STRUCT *pfFunc);
- kdSensorList[]

– #if defined(XXXXXX_RAW)

– {XXXXXX_SENSOR_ID, SENSOR_DRVNAME_XXXXXX_RAW, XXXXXX_RAW_SensorInit},

– #endif

- YUV sensor
- UINT32 XXXXXX_YUV_SensorInit(PSENSOR_FUNCTION_STRUCT *pfFunc);
- kdSensorList[]

– #if defined(XXXXXX_YUV)

– {XXXXXX_SENSOR_ID, SENSOR_DRVNAME_XXXXXX_YUV, XXXXXX_YUV_SensorInit},

– #endif

– sensorlist.cpp

- SensorList[]

– #if defined(XXXXXX_RAW)

– RAW_INFO(XXXXXX_SENSOR_ID, SENSOR_DRVNAME_XXXXXX_RAW, NULL),

– #endif

– #if defined(XXXXXX_YUV)

– YUV_INFO(XXXXXX_SENSOR_ID, SENSOR_DRVNAME_XXXXXX_YUV, NULL),
– #endif
– sensorkdSensorList[]SensorList[]resolution