RK3399 IO Domain Configuration Developer Guide

ID: RK-SM-YF-906

Release Version: V1.0.0

Release Date: 2021-05-15

Security Level: □Top-Secret □Secret □Internal ■Public

Rockchip Electronics Co., Ltd.

No.18 Building, A District, No.89, software Boulevard Fuzhou, Fujian,PRC

Website: www.rock-chips.com

Customer service Tel: +86-4007-700-590

Customer service Fax: +86-591-83951833

Customer service e-Mail: fae@rock-chips.com

Preface

Overview

The IO level of the controller's power domain must be matched with the IO level of the connected peripheral chip, and the voltage configuration of the software must be consistent with the voltage of the hardware. Otherwise, it may cause IO damage at worst.

This document mainly introduce the ways to configure IO power domain of RK3399 Linux SDK platform, aiming to help developers to configure IO power domain correctly.

Product Version

Chipset	System Version
RK3399	Linux 4.4, Linux 4.19

Intended Audience

This document (this guide) is mainly intended for:

- Technical support engineers
- Software development engineers
- Hardware development engineers

Revision History

Version	Author	Date	Change Description
V1.0.0	Caesar Wang	2021-05-15	Initial version

Contents

1. Step 1: Obtain the Hardware Schematic Diagram and Check the Design of the Hardware Power Supply

It will take RK_IND_EVB_RK3399_LP4D200P232SD8_V13_20200615 EVB as an example to introduce in this document.

Hardware schematic diagram is: RK_IND_EVB_RK3399_LP4D200P232SD8_V13_20200615.pdf.

Power solution: checking from the hardware schematic, the power solution of the **EVB RK_IND_EVB_RK3399_LP4D200P232SD8_V13_20200615** is with a PMU (RK809-3).

2. Step 2: Find the Corresponding Kernel dts Configuration File

From the first step, it can be seen that the hardware power supply design of the EVB is with a PMU , and the corresponding kernel dts configuration file is located in:

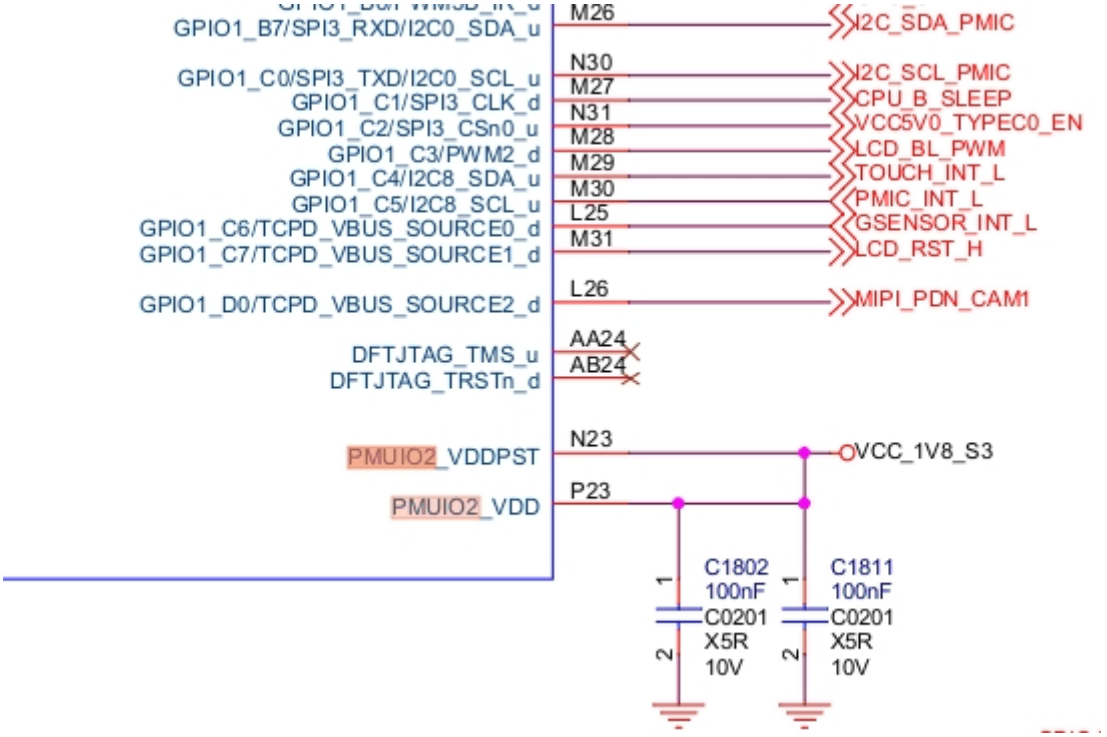
arch/arm64/boot/dts/rockchip/rk3399-evb-ind.dtsi (The solution discussed in this document)

3. Step 3: Modify the Power Domain Configuration Node pmu_io_domains of the Kernel dts

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

        bt656-supply = <&vcc_3v0>;              /* bt656_gpio2ab_ms */
        audio-supply = <&vcca_1v8>;     /* audio_gpio3d4a_ms */
        sdmmc-supply = <&vccio_sd>;             /* sdmmc_gpio4b_ms */
        gpio1830-supply = <&vcc_3v0>;           /* gpio1833_gpio4cd_ms */
};

&pmu_io_domains {
        status = "okay";
        pmu1830-supply = <&vcc_1v8>;
};

```

Take **pmuio2-supply** for example, firstly, check the hardware schematic diagram to confirm the configuration of the pmuio2 power domain (PMUIO2) as shown in the below figure.

From the figure, you will find that the power supply of **PMUIO2** is VCC_1V8_S3 (that is 1.8V)



In the same way, you can find from the hardware schematic diagram that bt656-supply is connected to APIO2_VDD, audio-supply is connected to APIO5_VDD, and gpio1830-supply is connected to APIO4_VDD.

4. Step 4: Check the Current Firmware IO Domain Configuration from SDK

Command: `./build.sh info`

```
请再次确认板级的电源域配置！！！！！！
<<< 特别是Wi-Fi, FLASH, 以太网这几路IO电源的配置 >>> ！！！！！
检查内核文件 /home/wxt/linux-develop/rk3399/kernel/arch/arm64/boot/dts/rockchip/rk3399-evb-ind-lpddr4-linux.dts 的节点 [pmu_io_domains]

pmu1830-supply
regulator-min-microvolt =  1800mV
regulator-max-microvolt =  1800mV

bt656-supply
regulator-min-microvolt =  3000mV
regulator-max-microvolt =  3000mV

audio-supply
regulator-min-microvolt =  1800mV
regulator-max-microvolt =  1800mV

sdmmc-supply
regulator-min-microvolt =  1800mV
regulator-max-microvolt =  3300mV

gpio1830-supply
regulator-min-microvolt =  3000mV
regulator-max-microvolt =  3000mV
```

5. Step 5: Confirm Whether the Register Value is Correct after Flashing the Firmware

Take **RK3399** as an example, get PMU_SOC_CON0 register (0xFF320180) and GRF_IO_VSEL register (0xFF77E640) from the manual, they are shown as follows:

PMUGRF_SOC_CON0
Address: Operational Base + offset (0x00180)
SoC control register 0

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:10	RO	0x0	reserved
9	RW	0x1	pmu1830_vol pmu IO 1.8v/3.0v select. 0: 3.0v ; 1: 1.8v ;
8	RW	0x1	pmu1830_volsel pmu GPIO1 1.8v/3.0v control source select. 0: controlled by IO_GPIO0B1 ; 1: controlled by PMUGRF.SOC_CON0.pmu1830_vol
7	RO	0x0	reserved

GRF_IO_VSEL
Address: Operational Base + offset (0x0e640)

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:4	RO	0x0	reserved
3	RW	0x0	gpio1833_gpio4cd_ms
2	RW	0x0	sdmmc_gpio4b_ms
1	RW	0x0	audio_gpio3d4a_ms
0	RW	0x0	bt656_gpio2ab_ms

```
# io -r -4 0xff320180
ff320180:  00000300

# io -r -4 0xff77e640
ff77e640:  00000002

```