Linux 快充
开发指南

版本号: 1.5
发布日期: 2025.02.11

版本历史

版本号	日期	制/修订人	内容描述
1.0	2023.08.28	AWA1691	初始版本。
1.1	2023.09.14	AWA1691	1. 调整了部分内容的顺序。 2. 修正图片和一些描述。
1.2	2023.11.15	AWA1691	1. 更新了部分配置的说明。 2. 新增关于 Android 配置的内容。
1.5	2025.02.11	AWA1691	1. 刷新文档结构，以适配多种快充方案。 2. 新增 A733 双节快充的内容。

目　录

插　图

表　格

1 前言

1.1 文档简介

介绍快充模块配置和调试方法。

1.2 目标读者

快充模块开发、维护人员。

1.3 适用范围

表 1-1: 适用产品列表

产品名称	内核版本	驱动文件
A523	Linux-5.15	bsp/drivers/power/
A733	Linux-6.6	bsp/drivers/power/

表 1-2: 不同平台的快充方案一览表

相关平台	快充类型	方案名称	芯片组合 (Type-C PD + 高压充电 + 电量计)
A523	单节快充	a523_evb_qc_single	HUSB311 + ETA6973/ETA6974 + AXP717
A523	双节快充	a523_evb_qc_double	HUSB311 + AXP519 + AXP2601
A733	双节快充	a733_evb1_qc_double	HUSB311 + AXP519 + AXP2602

说明

例如 “ETA6973/ETA6974” 代表该方案下 ETA6973 与 ETA6974 可兼容使用。

1.4 相关术语介绍

表 1-3: 术语介绍

术语	说明
PMU	电源管理单元。
BMU	电池控制器。
PMIC	电源管理芯片。
AXP	全志 PMU 的名称，如 AXP2202 等。
LDO	文档里指低压差线性稳压器经过调节的输出。
DC-DC	文档里把直流变直流由开关方式实现的器件和对应输出叫 DCDC。
regulator	linux 内核对 LDO、DC-DC 的控制核心。
powerkey	电源按键和其控制核心。
TWI	同等于 I2C，AXP 的使用需要用到 I2C 的通讯进行读写寄存器。
GPIO	通用型输入和输出，其引脚可以作为输入、输出或其他特殊功能，如中断。
VBUS	文档内指从 USB 线输入 PMU 的供电或供电源。
NTC	热敏电阻器，文档内指电池过温停充的策略。
JETIA	一种锂电池使用和充电规范，文档内指电池过温限流的策略。
TS	在这里指用于测量电池温度用的电阻中通过的电流大小。
单节快充	指用于单节电池的快充方案，通常输入上限功率是 18w（9V2A）。
双节快充	指用于双节电池的快充方案，最高输入上限功率是 45w（15V3A）。

表 1-4: 相同驱动的命名和编号对照表

驱动命名	对应编号
ETA6973	ETA6973、ETA6974
AXP2202	AXP2202、AXP717
AXP519	AXP519
AXP2601	AXP2601
AXP2602	AXP2602

2 模块介绍

如无特殊说明，以下介绍均适用于所有版本的快充配置。

2.1 模块功能介绍

快充管理模块，负责电池快速充电和放电的管理，由 Type-C PD 控制器驱动、充电驱动和电量计驱动组成。

- Type-C PD 控制器驱动负责管理快速充电协议。它根据连接的快充适配器和支持的快充协议类型，向充电器发送特定的控制信号，并通过接口通知充电驱动控制快速充电的电压和电流，以实现快速充电的效果。
- 充电驱动负责管理充电限流限压、boost 5V 输出和相关保护功能。它通过接收 Type-C PD 控制器驱动下发的指令，并综合其他充电保护条件来调整电源的电压、电流和其他参数，以实现高压充电和 boost 5V 输出功能。
- 电量计驱动负责监测电池的各种状态信息，例如电池电量、电压、温度等。它通过接口和充电驱动进行信息交流，完成自身初始化的同时协助充电驱动实现充电保护功能。

此外，完整系统的电源管理（PMIC）分为电池管理（BMU）和电源管理（PMU），后者详细配置参考《Linux_PMIC_ 开发指南》。

快充管理模块在完整系统中相当于 BMU，PMU 相关内容在本文档里不会展开太多。

这四者的各阶段对应详细功能如下表所示：

说明

Type-C PD 控制器芯片控制仅在内核阶段生效。

表 2-1: kernel 芯片驱动功能对应表

对应芯片	涉及对应驱动类型	负责功能
高压充电芯片	usb_power、regulator	上报 vbus 接入状态、开关 boost 输出
电量计芯片	battery	上报电池状态
Type-C PD 控制器芯片	tcpci	处理快充协议并进行协议握手
电源管理芯片	regulator、powerkey	给其他驱动供电、控制按键

表 2-2: uboot 芯片驱动功能对应表

对应芯片	涉及对应驱动类型	负责功能
高压充电芯片	bmu_ext	通过 bmu 驱动提供充电信息
电量计芯片	bmu	处理开机源、提供电池信息
电源管理芯片	pmu	各路电的电压和开关

2.2 方案设计原理

说明

以下内容以 A523-单节快充方案为例，其他方案如果有差异，会在后续《xx 方案-差异化》中详细说明。

如图所示绿色框内 ETA6973 SYS 输出作为电源管理芯片（PMU）PS 的输入。红色框内 ETA6973 VBAT 输出给单节电池充电，与电源管理芯片（PMU）的 VBAT 连接，使用电源管理芯片 (PMU) 的电量计功能。

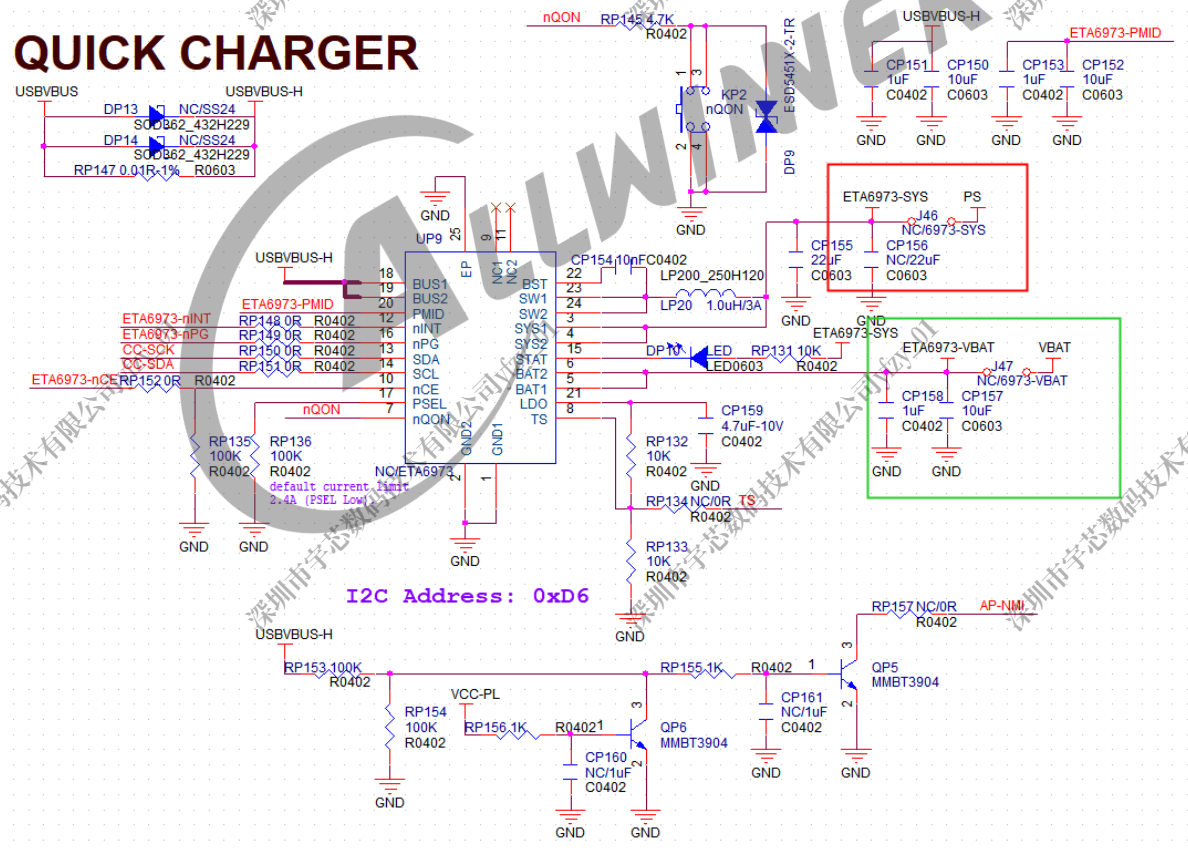

图 2-1: 快充硬件原理图

2.3 模块框架介绍

充电驱动和电量计驱动均采用 i2c 总线跟主控进行交互，使用 regmap 方式注册访问接口，共同使用父设备 mfd 的资源（bus、irq）。

说明

充电驱动和电量计驱动相比，多一个 regulator 子系统。

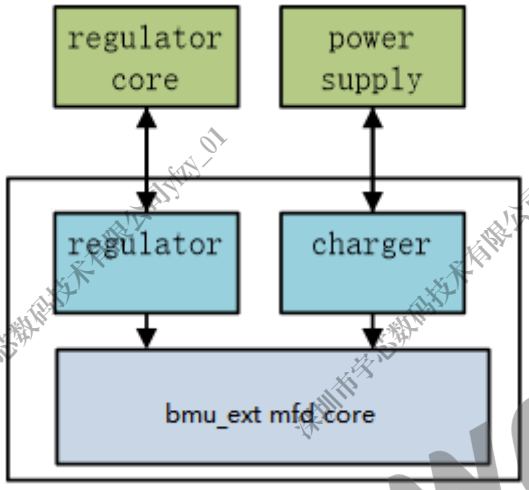

图 2-2: 快充框图

其中：

1. mfd 部分代码负责管理整体配置；
2. regulator 部分代码负责对外输出的 boost 5V 的控制；
4. power/supply 为 charger 部分代码，分别负责管理电池状态信息和充电控制等事项。

2.3.1 驱动限流通路框架

说明

以下内容以 A523-单节快充方案为例，其他方案如果有差异，会在后续《xx 方案-差异化》中详细说明。

其中黄色线为 otg 输出时的路径，蓝色线为充电时的路径，橙色线则是充电/电池数据的流动路径：

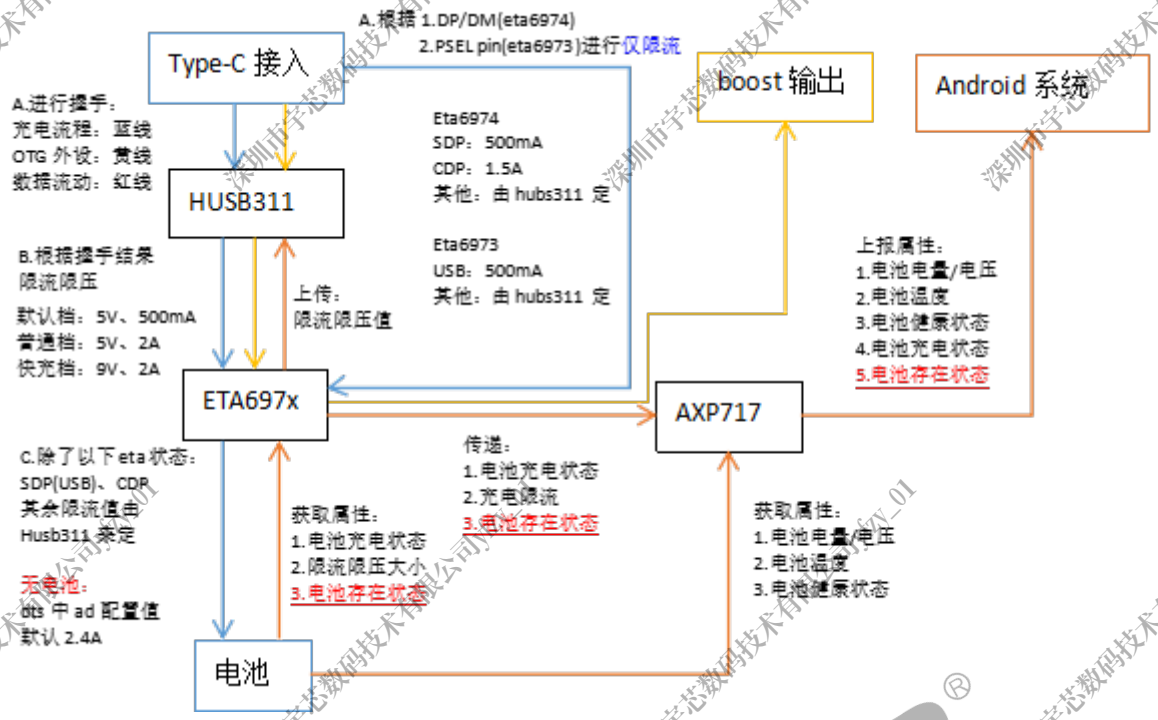

图 2-3: 驱动限流通路框架

2.3.2 驱动唤醒通路框架

说明

以下内容以 A523-单节快充方案为例，其他方案如果有差异，会在后续《xx 方案-差异化》中详细说明。

其中如果中断引脚使用了 gpio 口，就需要放在非掉电区域内，如图中的 PL 域的 gpio 口

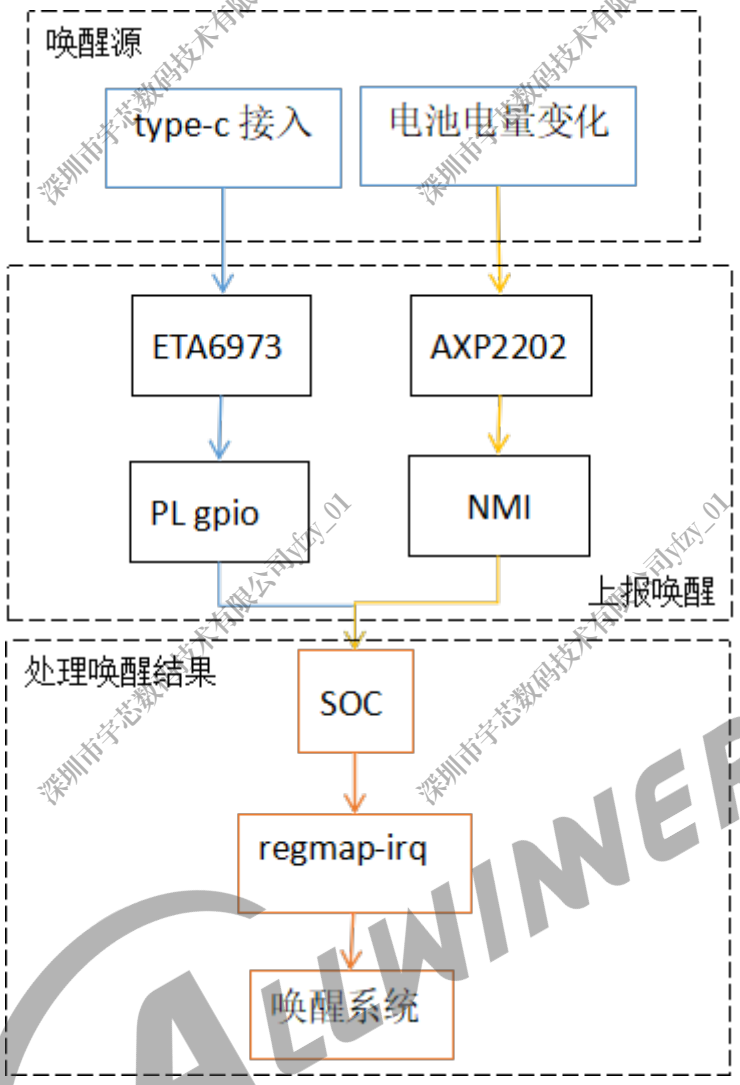

图 2-4: 驱动唤醒通路框架

2.4 源码结构介绍

以下源码结构的介绍顺序为开机流程的顺序。

说明

以下内容以 A523-单节快充方案为例，其他方案如果有差异，会在后续《xx 方案-差异化》中详细说明。

2.4.1 uboot

```
brandy/brandy-2.0/u-boot-2018
|-- board/sunxi/power_manage.c
|-- drivers/sunxi_power/
   |-- axp.c
   |-- bmu.c
   |-- pmu.c
```

```
   |-- bmu_axp2202.c
   |-- pmu_axp2202.c
   |-- bmu_ext.c
   |-- bmu_eta6973.c
...
```

2.4.2 kernel

```
bsp/drivers/power
|-- mfd/
   |-- axp2101.c
   |-- axp2101-i2c.c
   |-- bmu-ext-core.c
   |-- bmu-ext-i2c.c
|-- supply/
   |-- axp2202_battery.c
   |-- eta6973_charger_power.c
|-- regulator/
   |-- axp2101-regulator.c
   |-- bmu-ext-regulator.c
|-- power_key/axp2101-pek.c

bsp/drivers/usb/typec/tcpm/tcpci_husb311.c
```

2.5 模块配置介绍

这部分将讲述一个正常运行的快充模块需要的基础配置。

要实现完整快充功能，需要同时正确配置 Type-C PD 控制器驱动和充电驱动。

其中，device tree 相关的所有配置项基本都在 device 下，并根据配置生效时间分布在不同的配置文件中。

表 2-3: 不同版本的配置文件一览

生效时间	配置文件
uboot	uboot-board.dts
kernel	board.dts

说明

以下内容以 A523-单节快充方案为例，其他方案如果有差异，会在后续《xx 方案-差异化》中详细说明。

2.5.1 Device Tree 配置说明

2.5.1.1 uboot

uboot 阶段没有必要的基础配置，其相关的扩展配置详见 FAQ 章节内的常见功能配置。

2.5.1.2 kernel

2.5.1.2.1 Type-C PD 控制器驱动

```
&twi5 {
  husb311: husb311@4e {
    compatible = "hynetek,husb311";
    reg = <0x4e>;
    interrupt-parent = <&r_pio>;
    husb311,intr_gpio = <&r_pio PL 13 GPIO_ACTIVE_LOW>;
    vbus-supply = <&reg_qc_drivevbus>;
    det_usb_supply = <&charger_power_supply>;
    /* aw,vbus-wakeup-quirk; */
    /* aw,port-reset-quirk; */
    status = "okay";

    usb_con: connector {
      compatible = "usb-c-connector";
      label = "USB-C";
      data-role = "dual";
      power-role = "dual";
      try-power-role = "sink";
      op-sink-microwatt = <1000000>;
      slow-charger-loop;
      sink-pdos = < PDO_FIXED(5000, 500, (0x1<<29)|(0x1<<25))
                PDO_VAR(5000, 5000, 2000)
                PDO_VAR(9000, 9000, 2000) >;
      source-pdos = < PDO_FIXED(5000, 500, (0x1<<29)|(0x1<<25))
                  PDO_VAR(5000, 5000, 1200) >;
    };
  };
};
```

```
husb311: husb311@4e {

compatible <char>
    设备匹配名，对应Type-C PD控制器的型号

reg <u32>
    i2c寄存器地址

interrupt-parent <args>
husb311,intr_gpio <args>
    中断配置，例子中husb311使用PL13作为中断脚

vbus-supply = <&reg_qc_drivevbus>;
```

```
    引用boost 5V输出，根据CC状态设置Vbus的开启和关闭

det_usb_supply = <&charger_power_supply>;
    引用充电驱动，通过PowerSupply框架接口回调快充驱动的函数

aw,vbus-wakeup-quirk <bool>
    husb311是否能唤醒soc

aw,port-reset-quirk <bool>
    husb311唤醒时是否重新初始化
}

usb_con: connector {

data-role <char>
    数据传输方向：
    dual：双向
    host/device：主机/从机
power-role <char>
    充电/供电方向：
    dual：双向
    sink/source：充电/放电
try-power-role <char>
    优先匹配充电/供电方向
    sink/source：充电/放电
op-sink-microwatt <char>
    最小充电功率
slow-charger-loop <bool>
    匹配档位时，是否从最低默认档位开始
sink-pdos <args>
    配置充电档位，例子中配置的档位如下：
    默认档：5V 0.5A
    档1：5V 2A
    档2：9V 2A
source-pdos <args>
    配置放电档位，例子中配置的档位如下：
    默认档：5V 0.5A
    档1：5V 1.2A
}
```

2.5.1.2.2 充电驱动

- mfd

充电驱动在内核中的使用 mfd 框架管理其中断和通讯总线配置，考虑到需兼顾 boost 5V 输出的功能，因此驱动包含 regulator 和 power_supply（usb）两个子系统。

因此，在内核要使用充电驱动时需要先配置、打开主设备，其次才能去配置和使用两个子系统。

```
充电驱动-主设备(MFD)
|
+------> regulator device
|
+------> power supply device
```

主设备的节点为 qc0，放置在对应所使用的 i2c 节点下。
后面两个子系统的节点均放在 qc0 下。

说明

不同的快充方案，其板级配置不兼容。

```
qc0: qc@6b{
   compatible = "eta,eta6973";
   reg = <0x6b>;
   /*
   interrupts = <0 IRQ_TYPE_LEVEL_LOW>;
   interrupt-parent = <&nmi_intc>;
   */
   status = "okay";
   x-powers,drive-vbus-en;
   wakeup-source;
.....
}
```

```
compatible <char>
    设备匹配名，对应快充的型号

reg <u32>
    i2c寄存器地址

interrupts <args>
    中断配置，参考内核中断配置文档
    **部分快充驱动不支持，如当前的ETA6973**
    **ETA6973的中断配置放在了power_supply下**

interrupt-parent <phandler>
    上级中断控制器结点
    **部分快充驱动不支持，如当前的ETA6973**
    **ETA6973的中断配置放在了power_supply下**

wakeup-source <bool>
    是否作为唤醒源

x-powers,drive-vbus-en <bool>
    是否将 N_VBUSEN 作为输出以对外供电
```

- regulator

regulator 为系统 regulator_dev 设备，每个 regulator_dev 代表一路电源。

快充设备通过对 regulator_dev 的引用建立 regulator，用来实现对 boost 5V 升压输出的功能。

regulator 属性配置和具体含义可参考内核原生 regulator 使用文档：

> Documentation/devicetree/bindings/regulator/regulator.yaml

```
regulator4: regulators@4 {
    reg_qc_drivevbus: drivevbus {
        regulator-name = "eta6973-drivevbus";
        regulator-enable-ramp-delay = <1000>;
    };
```

```
};
```

```
reg_qc_drivevbus: drivevbus {
    regulator-name = "eta6973-drivevbus";
        为电源设备的名称

    regulator-enable-ramp-delay = <1000>;
        电源从关闭到开启的使能延时，单位：us

    regulator-boot-on;
        电源从启动时开启

    regulator-always-on;
        电源保持常开
};
```

- power_supply

power_supply 属性配置，因充电驱动管理充电行为，因此以 charger 子节点来配置。

```
charger_power_supply: charger_power_supply{
    compatible = "eta,eta6973-charger-power-supply";
    status = "okay";
    wakeup-source;
    det_battery_supply = <&bat_power_supply>;
    ……
    qc_det_gpio = <&r_pio PL 4 IRQ_TYPE_LEVEL_LOW>;
    qc_nce_gpio = <&r_pio PL 5 GPIO_PULL_DOWN>;
    bat_state_gpio = <&r_pio PL 7 GPIO_ACTIVE_LOW>;
};
```

```
compatible <char>
    驱动匹配名

wakeup-source <bool>
    是否作为唤醒源

det_battery_supply = <&bat_power_supply>;
    引用电量计驱动驱动，用于获取电池状态信息

qc_det_gpio
    检测高压充电设备中断的gpio口。
    **部分快充驱动支持，如ETA6973**

qc_nce_gpio
    配置高压充电设备充电使能的gpio口。
    **部分快充驱动支持，如ETA6973**

bat_state_gpio
    检测电池存在判断的gpio口。
    **部分快充驱动支持，如ETA6973**
```

2.5.1.2.3 电量计驱动

说明

AXP2202 虽然有充电功能，但在 A523-单节快充方案中，AXP2202 是作为内置电量计 + 电源管理芯片使用

- mfd

电量计驱动在内核中也使用 mfd 框架进行管理其中断和通讯总线配置，仅 power_supply（battery）子系统。

因此也需要先配置、打开主设备，才能去配置和使用子系统。

```
电量计驱动-主设备(MFD)
|
+------> power supply device
```

其主设备的节点不固定，由具体电量计驱动决定，放置在对应所使用的 i2c 节点下。
后面子系统的节点均放在主设备节点下。

```
pmu0: pmu@0{
   compatible = "x-powers,axp2202";
   reg = <0x34>;
   interrupts = <0 IRQ_TYPE_LEVEL_LOW>;
   interrupt-parent = <&nmi_intc>;
   status = "okay";
   wakeup-source;
……
}
```

```
compatible <char>
      设备匹配名，对应电量计的型号

reg <u32>
      i2c寄存器地址

interrupts <args>
      中断配置，参考内核中断配置文档

interrupt-parent <phandler>
      上级中断控制器结点

wakeup-source <bool>
      是否作为唤醒源
```

- power_supply

power_supply 属性配置，因电量计驱动管理电池状态，因此以 battery 子节点来配置。

```
bat_power_supply: bat-power-supply {
        compatible = "x-powers,axp2202-bat-power-supply";
        status = "okay";
        param = <&axp2202_parameter>;
        det_qc_supply = <&charger_power_supply>;
```

```
    ......
}
```

```
compatible <char>
    驱动匹配名

param = <&axp2202_parameter>;
    引用电池参数，用于电量计算
    AXP2202：128个参数
    AXP2601：80个参数

det_battery_supply = <&bat_power_supply>;
    引用充电驱动驱动，用于获取和控制充电状态
```

2.5.2 kernel menuconfig 配置说明

进入内核根目录，执行 make ARCH=arm menuconfig（64 位平台为 make ARCH=arm64 menuconfig），或是在 longan 目录下输入:

```
./build.sh menuconfig
```

以 A523-单节快充方案为例，其余方案仅驱动名不同，其他参考相应的 defconfig 文件。进入配置主界面，并按以下步骤操作：

- Type-C PD 控制器驱动

```
Allwinner BSP  --->  Device Drivers  --->
  I2C Drivers  --->  <*> I2C Support for Allwinner SoCs
Allwinner BSP  --->  Device Drivers  --->
  USB Drivers  --->  USB Type-C Support --->
    <*> Allwinner USB Type-C support
    <*> Hynetek HUSB311 Type-C chip driver
```

- 充电驱动

```
Allwinner BSP  --->  Device Drivers  --->
  I2C Drivers  --->  <*> I2C Support for Allwinner SoCs
  PMIC Drivers  --->  <*> BMU_EXT PMICs with I2C
        --->  <*> ETA6973 Power Supply Driver
```

- 电量计驱动

```
Allwinner BSP  --->  Device Drivers  --->
  I2C Drivers  --->  <*> I2C Support for Allwinner SoCs
  PMIC Drivers  --->  <*> X-POWERS AXP2101 PMICs with I2C
        --->  <*> AXP2202 Power Supply Driver
```

2.5.3 Android 配置说明

进入 android 的 device 目录，在 device-common.mk 文件中增加以下属性。

以 A523-单节快充方案为例，其余方案仅驱动名不同：

打开快充配置：

```
device/softwinner/saturn/a523-evb/device-common.mk:
……
CONFIG_AW_QUICK_CHARGER := true
……
```

```
CONFIG_AW_QUICK_CHARGER <bool>
    打开Android层适配快充的功能
```

```
device/softwinner/saturn/AndroidProducts.mk:
PRODUCT_MAKEFILES := \
  $(LOCAL_DIR)/a523_evb_qc_single_arm_go.mk \
……
COMMON_LUNCH_CHOICES := \
  a523_evb_qc_single_arm_go-user \
  a523_evb_qc_single_arm_go-userdebug \

……
```

```
PRODUCT_MAKEFILES 与 COMMON_LUNCH_CHOICES
    增加方案配置
```

```
device/softwinner/saturn/a523_evb_qc_single_arm_go.mk:
……
PRODUCT_NAME := a523_evb_qc_single_arm_go
PRODUCT_DEVICE := a523-evb
PRODUCT_BOARD := evb_qc_single
……
```

```
a523_evb_qc_single_arm_go.mk
    复制已有板型配置，修改NAME/DEVICE/BOARD配置信息，以新增方案配置文件

PRODUCT_NAME
    当前方案名称

PRODUCT_DEVICE
    Android-device配置

PRODUCT_BOARD
    linux-device配置
```

3 模块使用范例

说明

以下内容以 A523-单节快充方案为例，其他方案如果有差异，会在后续《xx 方案-差异化》中详细说明。

3.1 regulator 使用 demo

其他设备在使用boost 5V输出时，采用标准regulator_dev设备的引用配置。
<name>-supply = <&reg_qc_drivevbus>;
设备中通过对name的获取，可以获取reg_qc_drivevbus的regulator_dev设备，然后对此路电源进行电源开关等控制。
具体设备引用参考内核的regulator使用文档。

3.2 power_supply 使用 demo

- dts 配置示例

其他设备对 power supply 设备的引用通过设备树配置。= <&usb_power_supply>; 设备中通过对 name 的获取，可以获取 usb_power_supply 的 power_supply 设备，然后读取或设置此供电状态。
具体设备引用参考内核的 power supply 使用文档。

> Documentation/power/power_supply_class.rst

```
qc0: qc@6b{
...
  charger_power_supply: charger_power_supply{
    compatible = "eta,eta6973-charger-power-supply";
    status = "okay";

  };
...
}

...
udc:udc-controller@0x05100000 {
  det_vbus_supply = <&charger_power_supply>;
};
...
```

- 驱动代码示例

模块驱动通过 devm_power_supply_get_by_phandle 获取 usb_power_supply 的 power_supply 设备，然后使用 power_supply_set_property 等接口，读取或设置此供电状态。

```
if (of_find_property(g_udc_pdev->dev.of_node, "det_vbus_supply", NULL))
    psy = devm_power_supply_get_by_phandle(&g_udc_pdev->dev,
                                           "det_vbus_supply");

if (! psy || IS_ERR(psy)) {
    DMSG_PANIC("%s()%d WARN: get power supply failed\n",
               __func__, __LINE__);
} else {
    temp.intval = 500;

    power_supply_set_property(psy,
                    POWER_SUPPLY_PROP_INPUT_CURRENT_LIMIT, &temp);
}
```

4 调试方法

说明

以下内容以 A523-单节快充方案为例，其他方案如果有差异，会在后续《xx 方案-差异化》中详细说明。

4.1 调试节点

4.1.1 获取 regulator 引用设备

获取有哪几路电源引用了 boost 5V，可以和普通 regulators 一样进入需要查看的电源，

以普通 regulator 为例：

```
进入/sys/class/regulator/regulator.1
通过sl查看当前目录下的目录，即可确定有哪几路引用设备。
```

也可以进入 regulator 的 debugfs 结点，用来查看 regulator 的 map 信息，详见下章。

4.1.2 查询 regulator 状态

kernel 提供调试结点供电源进行调试进行，我们可以通过 kernel 的调试结点获取各路电源的各个详细状态。

```
要求内核已打开DEBUG_FS的宏
```

首先需要 mount debugfs 文件系统。

然后就可以很直观的看出有哪几路设备挂在了 boost 5v 输出上。

```
mount -t debugfs none /sys/kernel/debug
cat /sys/kernel/debug/regulator/regulator_summary

regulator                use  open bypass opmode   voltage current   min    max
---------------------------------------------------------------------------------
regulator-dummy           24  34    0   unknown    0mV    0mA    0mV    0mV
  eta6973-drivevbus        0   1    0   unknown    0mV    0mA    0mV    0mV
    6-004e-vbus            0                       0mA    0mV    0mV
```

4.1.3 手动读写 PMIC 寄存器

4.1.3.1 使用标准 regmap registers 节点

寄存器调试是指直接对 PMIC 的寄存器进行读写操作，此操作应该对寄存器有了解的情况下进行操作，不正确的操作方式将会导致芯片烧毁。
在终端中，对抛出的调试结点进行读写操作，即可对寄存器进行读写操作。
无论是读还是写寄存器，都应该首先挂载 debugfs 文件系统。

```
mount -t debugfs none /sys/kernel/debug
```

由于 PMIC 是通过 regmap 进行读写操作，应该可以使用 regmap 的调试结点进行对 PMIC 的读写访问操作。
regmap 的调试结点在 debugfs 文件系统下面，通过对 regmap 调试结点的操作可以对 PMIC 的寄存器进行读写访问操作。

说明

需要注意，原生内核默认不支持写操作，需要增加宏定义 REGMAP_ALLOW_WRITE_DEBUGFS
对应驱动：drivers/base/regmap/regmap-debugfs.c

- 写操作

```
寄存器调试挂载在debugfs文件系统。
mount -t debugfs none /sys/kernel/debug
echo ${reg} ${value} > /sys/kernel/debug/regmap/${dev-name}/registers

实例：
echo 0xff 0x01 > /sys/kernel/debug/regmap/4-0034/registers
写0xff寄存器值为0x01
```

- 读操作

```
寄存器调试挂载在debugfs文件系统。
mount -t debugfs none /sys/kernel/debug
cat /sys/kernel/debug/regmap/${dev-name}/registers

实例：
cat /sys/kernel/debug/regmap/4-0034/registers
读取pmic所有寄存器
```

4.1.3.2 i2c 工具读写

另外，i2c 工具还支持单独读写特定外设的寄存器。

以 eta6973（基地址 0x6b，挂载在 i2c-6 上）为例：

```
往寄存器0x05写入值0x55：
i2cset -f -y 6 0x6b 0x05 0x55 b
读出寄存器0x05的值：
i2cget -f -y 6 0x6b 0x05
```

4.1.4 查看供电和电池状态

根据供电的种类，读取/sys/class/power_supply/{battery,usb} 下面状态，判断一致性。

- Type-C PD 控制器驱动

说明

以下内容以 A523-单节快充方案为例，其他方案如果有差异，会在后续《xx 方案-差异化》中详细说明。

```
/ # cat sys/class/power_supply/tcpm-source-psy-5-004e/uevent
POWER_SUPPLY_NAME=tcpm-source-psy-5-004e
POWER_SUPPLY_TYPE=USB
POWER_SUPPLY_USB_TYPE=[C] PD PD_PPS
POWER_SUPPLY_ONLINE=0
POWER_SUPPLY_VOLTAGE_MIN=0
POWER_SUPPLY_VOLTAGE_MAX=0
POWER_SUPPLY_VOLTAGE_NOW=0
POWER_SUPPLY_CURRENT_MAX=0
POWER_SUPPLY_CURRENT_NOW=0
```

表 4-1: Type-C PD 控制器驱动状态信息一览

节点名称	含义
POWER_SUPPLY_ONLINE	是否有接入 cc
POWER_SUPPLY_VOLTAGE_MIN	当前 vbus 输入限压最小值
POWER_SUPPLY_VOLTAGE_MAX	当前 vbus 输入限压最大值
POWER_SUPPLY_VOLTAGE_NOW	当前 vbus 输入限压值
POWER_SUPPLY_CURRENT_MAX	当前 vbus 输入限流最大值
POWER_SUPPLY_CURRENT_NOW	当前 vbus 输入限流值

- 充电驱动

说明

以下内容以 A523-单节快充方案为例，其他方案如果有差异，会在后续《xx 方案-差异化》中详细说明。

可通过读取/sys/class/power_supply/eta6973-charger 下面状态，来获取充电相关的信息

```
/ # cat sys/class/power_supply/eta6973-charger/uevent
POWER_SUPPLY_NAME=eta6973-charger
POWER_SUPPLY_TYPE=USB
POWER_SUPPLY_PRESENT=1
POWER_SUPPLY_ONLINE=1
POWER_SUPPLY_STATUS=Charging
```

```
POWER_SUPPLY_VOLTAGE_NOW=10500
POWER_SUPPLY_INPUT_CURRENT_LIMIT=1500
POWER_SUPPLY_MANUFACTURER=eta,eta6973
POWER_SUPPLY_TIME_TO_FULL_NOW=10
POWER_SUPPLY_SCOPE=System
POWER_SUPPLY_CALIBRATE=1
POWER_SUPPLY_CONSTANT_CHARGE_CURRENT=0
```

表 4-2: 充电驱动状态信息一览

节点名称	含义
POWER_SUPPLY_PRESENT、POWER_SUPPLY_ONLINE	是否有接入充电器
POWER_SUPPLY_VOLTAGE_NOW	当前 vbus 输入限压
POWER_SUPPLY_INPUT_CURRENT_LIMIT	当前 vbus 输入限流
POWER_SUPPLY_PROP_STATUS	当前电池充电状态
POWER_SUPPLY_PROP_TIME_TO_FULL_NOW	当前电池满充充电时间限制
POWER_SUPPLY_PROP_SCOPE	当前开启的模式（host、default）
POWER_SUPPLY_PROP_CALIBRATE	当前电池的存在状态（有电池、无电池）
POWER_SUPPLY_PROP_CONSTANT_CHARGE_CURRENT	当前给电池充电的限流情况

- 电量计驱动

```
/ # cat sys/class/power_supply/axp2202-battery/uevent
POWER_SUPPLY_NAME=axp2202-battery
POWER_SUPPLY_TYPE=Battery
POWER_SUPPLY_PRESENT=1
POWER_SUPPLY_VOLTAGE_NOW=4165000
POWER_SUPPLY_ENERGY_FULL_DESIGN=1771
POWER_SUPPLY_STATUS=Charging
POWER_SUPPLY_CAPACITY_ALERT_MIN=15
POWER_SUPPLY_TEMP=300
POWER_SUPPLY_CAPACITY=90
POWER_SUPPLY_TEMP_ALERT_MIN=-200000
POWER_SUPPLY_TEMP_ALERT_MAX=200000
POWER_SUPPLY_TEMP_AMBIENT_ALERT_MIN=-200000
POWER_SUPPLY_TEMP_AMBIENT_ALERT_MAX=200000
POWER_SUPPLY_TIME_TO_EMPTY_NOW=52440
POWER_SUPPLY_TIME_TO_FULL_NOW=1260
POWER_SUPPLY_MANUFACTURER=xpower,axp2202
POWER_SUPPLY_CAPACITY_LEVEL=High
POWER_SUPPLY_CONSTANT_CHARGE_CURRENT=1044
POWER_SUPPLY_HEALTH=Good
POWER_SUPPLY_CHARGE_COUNTER=200000
POWER_SUPPLY_CHARGE_FULL=5000000
```

表 4-3: 电量计驱动信息一览

参数名称	对应含义
POWER_SUPPLY_NAME	驱动名称
POWER_SUPPLY_TYPE	驱动类型（电池）
POWER_SUPPLY_PRESENT	电池存在状态

参数名称	对应含义
POWER_SUPPLY_VOLTAGE_NOW	电池电压
POWER_SUPPLY_ENERGY_FULL_DESIGN	满充电量
POWER_SUPPLY_STATUS	充电状态
POWER_SUPPLY_CAPACITY_ALERT_MIN	警告电量
POWER_SUPPLY_TEMP	电池温度/芯片温度（根据配置变化）
POWER_SUPPLY_CAPACITY	电池电量百分比
POWER_SUPPLY_TEMP_ALERT_MIN	电池欠温关机温度
POWER_SUPPLY_TEMP_ALERT_MAX	电池过温关机温度
POWER_SUPPLY_TEMP_AMBIENT_ALERT_MIN	电池欠温停充温度
POWER_SUPPLY_TEMP_AMBIENT_ALERT_MAX	电池过温停充温度
POWER_SUPPLY_TIME_TO_EMPTY_NOW	放电放光时间
POWER_SUPPLY_TIME_TO_FULL_NOW	电池充满时间
POWER_SUPPLY_MANUFACTURER	电池驱动名字
POWER_SUPPLY_CAPACITY_LEVEL	电池电量情况
POWER_SUPPLY_CONSTANT_CHARGE_CURRENT	充电电流上限
POWER_SUPPLY_HEALTH	电池状态
POWER_SUPPLY_CHARGE_COUNTER	当前电流
POWER_SUPPLY_CHARGE_FULL	满充电量

5 FAQ

5.1 常见功能配置

这部分将根据实现的不同功能进行配置的讲解，可根据功能检索对应配置项。

说明

以下内容以 A523-单节快充方案为例，其他方案如果有差异，会在后续《xx 方案-差异化》中详细说明。

说明

大部分配置和《Linux_PMIC_ 开发指南》中描述一致。
如有冲突，则以当前文档为准。

5.1.1 唤醒源配置

```
device/·····/board.dts:

pmu0: pmu@34 {
  ·····
  pmu_irq_wakeup = <1>;
  wakeup-source;
  ·····
  battery_power_supply: battery-power-supply {
    ·····
    wakeup_bat_out;
    /* wakeup_new_soc; */
    /* wakeup_bat_in; */
    /* wakeup_bat_charging; */
    /* wakeup_bat_charge_over; */
    /* wakeup_low_warning1; */
    /* wakeup_low_warning2; */
    /* wakeup_bat_untemp_work; */
    /* wakeup_bat_ovtemp_work; */
    /* wakeup_bat_untemp_chg; */
    /* wakeup_bat_ovtemp_chg; */
    ·····
  };
  ·····
};
```

```
**以下通用配置，所有方案均支持**
wakeup-source <bool>
    是否作为唤醒源，在使用PMU任何中断时必须配置

pmu_irq_wakeup
    trigger irq wakeup or not when sleep or power down，default 0
    0: not wakeup
```

1: wakeup

wakeup_new_soc <bool>
电池电量更新唤醒使能

以下配置视具体的board.dts属性节点而定，部分方案暂未支持该功能
当前支持方案：A523-单节快充方案

wakeup_bat_in <bool>
电池插入唤醒使能

wakeup_bat_out <bool>
电池拔出唤醒使能

wakeup_bat_charging <bool>
电池充电唤醒使能

wakeup_bat_charge_over <bool>
电池充电结束唤醒使能

wakeup_low_warning1 <bool>
电池低电量告警唤醒使能

wakeup_low_warning2 <bool>
电池低电量告警2唤醒使能

wakeup_bat_untemp_chg <bool>
电池低温充电唤醒使能

wakeup_bat_ovtemp_chg <bool>
电池超温充电唤醒使能

wakeup_bat_untemp_work <bool>
电池低温工作唤醒使能，欠温关机必须配置

wakeup_bat_ovtemp_work <bool>
电池高温工作唤醒使能，过温关机必须配置

5.1.2 无电池方案属性

```
device/……/board.dts:
qc0: qc@6b{
  ……
  charger_power_supply: charger_power_supply{
    ……
    battery_exist_sets = <0>;
    ……
  };
}

device/……/uboot-board.dts:
&power_sply {
……
  battery_exist = <0>;
……
};
```

```
**以下通用配置，所有方案均支持**
battery_exist <bool>
    该属性不配置默认为1，仅uboot生效。
    1：根据实际寄存器情况来判断
    0：强制判断为不使用电池

注：在配置无电池方案的同时，建议关闭NTC温控属性功能，详见《模块功能配置-NTC温控属性配置》

**以下配置视具体的board.dts属性节点而定，部分方案暂未支持该功能**
**当前支持方案：A523-单节快充方案**

battery_exist_sets <bool>
    该属性不配置默认为0，仅内核生效。
    2：强制判断为有电池
    1：强制判断为无电池
    0：根据实际寄存器情况来判断
```

5.1.3 开机判断配置

以下几项内容跟开机判断相关：

1. 强制电池不存在判断
2. 机器的开机模式
3. 开机最低电压和电量

```
device/·····/uboot-board.dts:
&power_sply {
  charge_mode = <1>;
  battery_exist = <1>;
};
&charger0 {
  pmu_safe_vol = <3400>;
  pmu_safe_ratio = <1>;
};
```

```
battery_exist <u32>
    强制电池存在状态，uboot阶段根据该属性决定是否做电池状态的相关判断。
    适用于部分无电持方案或factory_mode的无电池场景的调试。
    如果该属性不进行配置，默认为1。
    0： 强制认为无电池存在，uboot阶段不做电池相关状态判断
    1： 认为电池存在，uboot阶段正常进行电池状态的相关判断

charge_mode <u32>
    配置开机方式，根据该属性决定接适配器开机的方式：进入充电页面、需等待按键摁下开机和直接开机。
    适用于不需要充电页面，或者适配器唤醒直接开机的需求。
    又或者是需要等待按键摁下才能开机的需求。
    如果该属性不进行配置，默认为1。
    0： 不进入充电页面，接适配器开机直接进入开机流程
      此情况下在接适配器时关机后会重启
    1： 接适配器开机后进入关机充电页面
      此情况下在接适配器时关机后会进入关机充电页面
    2： 接适配器开机进入等待状态，摁下按键后直接进入开机流程
      此情况下在接适配器时关机后会重新进入等待状态，摁下按键后会直接进入开机流程

pmu_safe_vol <u32>
```

```
  uboot阶段允许开机到内核的最低电压，默认为3.4v

pmu_safe_radio <u32>
  uboot阶段允许开机到内核的最低电量，默认为1%
```

5.1.4 基础电池属性

基础电池属性包含以下几个方面：

1. 电池满充电压、电池容量
2. 低电警告电量
3. 电池曲线参数

5.1.4.1 电池满充电压、电池容量

```
device/……/board.dts:
bat_power_supply: bat-power-supply {
  ……
  pmu_init_chgvol = <4200>;
  pmu_battery_cap = <1771>;
  ……
};
```

```
pmu_init_chgvol <u32>
    电池满充电压，默认4.2v
    单位为mV

pmu_battery_cap <u32>
    电池容量，默认4000mAh
    单位为mAh
```

5.1.4.2 低电警告电量

低电警告电量如果有两个等级，一般默认为 15% 和 0%，

如果仅有一个等级，一般默认为 15%。

```
device/……/board.dts:
bat_power_supply: bat-power-supply {
  ……
  pmu_battery_warning_level1 = <15>;
  pmu_battery_warning_level2 = <0>;
  ……
};
```

```
**以下配置视具体的board.dts属性节点而定，部分方案暂未支持该功能**
**当前支持方案：A523-单节快充方案**
pmu_battery_warning_level1 <u32>
  5-20 5% - 20% 警告等级1，默认15%
```

```
pmu_battery_warning_level2 <u32>
    0-15 0% - 15% 警告等级2，默认0%

**以下配置视具体的board.dts属性节点而定，部分方案暂未支持该功能**
**当前支持方案：A523-双节快充方案、A733-双节快充方案**
pmu_battery_warning_level <u32>
    5-20 5% - 20% 警告等级，默认15%
```

- 电池曲线参数

说明

若要更新该参数，可参考 1 号通上的文档《X-POWERS 电池参数测试系统使用说明 _V1.4.pdf》来进行电池参数测试

```
device/……/board.dts:
/{
  axp2202_parameter:axp2202-parameter {
      select = "battery-model";

      battery-model {
          parameter = /bits/ 8 <0x01 0xf5 0x00 0x00 0xfb 0x00 0x00 0xfb
              0x00 0x1e 0x32 0x02 0x14 0x05 0x0a 0x04
              0x74 0xfd 0xbc 0x0c 0xdf 0x10 0xcc 0xfb
              0x80 0x00 0xdc 0x18 0xe4 0x06 0xcc 0x06
              0x94 0x0b 0x4f 0x0f 0xcf 0x0f 0x6d 0x0a
              0x05 0x0e 0xcb 0x0e 0xc8 0x04 0xbb 0x04
              0xaa 0x09 0x98 0x0e 0x79 0x0e 0x67 0x09
              0x62 0x0e 0x08 0x0c 0x3e 0x02 0x0c 0x01
              0xda 0x06 0xa8 0x0b 0x71 0x0b 0x44 0x06
              0x12 0x72 0x3c 0x33 0x27 0x14 0x1a 0x0e>;
      };
  };
};
```

```
param <phandler>
    指定电池充电结点，节点名称一般统一为“axp2202_parameter”
    电池参数由多个字节组成,在param指定的phandler结点里面添加电池结点，然后通过select选择参数结点名字用来指定哪个结点。
    参考实例：根据前面的pmic参考设备树节点的pmic-parameter结点。
    **电池参数根据使用的电池不同，电量计不同而不同，使用前请仔细确认当前方案的电量计芯片型号**
```

5.1.4.3 输入限流/限压默认值

限流分为输入限流和电池充电限流，前者决定适配器/pc 能输入到机器的电流上限，后者决定充到电池的电流上限。

限压指的是输入到机器的电压上限。

```
输入电流 = 电池充电电流 + SOC功耗
```

说明

快充模块中，输入限流/限压值最终由 Type-C PD 控制器驱动来决定。
这里的配置值仅用于 Type-C PD 控制器驱动未进行限流设置时的默认值。

```
device/……/board.dts:
charger_power_supply: charger_power_supply{
        ……
        pmu_chargerpc_vol = <5000>;
        pmu_chargerpc_cur = <500>;
        pmu_chargerad_vol = <5000>;
        pmu_chargerad_cur = <2400>;
        ……
};
```

```
pmu_chargerpc_vol <u32>
    usb pc输入电压限制值
    单位为mV

pmu_chargerpc_cur <u32>
    usb pc输入电流限制值
    单位为mA

pmu_chargerad_vol <u32>
    usb adaptor输入电压限制值(vimdpm)
    单位为mV

pmu_chargerad_cur <u32>
    usb adaptor输入电流限制值
    单位为mA
```

5.1.5 充电属性配置

充电相关的内容如下所示：

1. 电池充电限流

5.1.5.1 电池充电限流

```
device/……/board.dts:
charger_power_supply: charger_power_supply{
  ……
  pmu_runtime_chgcur = <1000>;
  pmu_suspend_chgcur = <2000>;
  pmu_shutdown_chgcur = <2000>;
  ……
};
```

```
pmu_runtime_chgcur <u32>
    运行时constant充电电流限制，默认2000mA
    单位为mA

pmu_suspend_chgcur <u32>
    休眠时constant充电电流限制，默认3000mA
    单位为mA

pmu_shutdown_chgcur <u32>
    关机时constant充电电流限制，默认3000mA
```

```
    单位为mA
```

5.1.6 drivevbus 属性配置

完整的 boost 5V 输出需要同时配置好 Type-C PD 控制器驱动和充电驱动的属性。

```
qc0: qc@6b{
   regulator4: regulators@4 {
        reg_qc_drivevbus: drivevbus {
          regulator-name = "eta6973-drivevbus";
          regulator-enable-ramp-delay = <1000>;
        };
   };
};
husb311: husb311@4e {
      ......
      det_usb_supply = <&charger_power_supply>;
      ......
}
```

```
reg_qc_drivevbus: drivevbus {
   regulator-name = "eta6973-drivevbus";
        为电源设备的名称

   regulator-enable-ramp-delay = <1000>;
        电源从关闭到开启的使能延时，单位：us
};
```

5.1.7 NTC 温控属性配置

NTC 温控相关属性如下所示：

1. ntc、jeita 使能、jeita 限流参数
2. 电池温度参数、ts 电流配置
3. 开机温度限制 (uboot)
4. 触发限流、停充、关机的电压阈值

说明

需内核过温关机功能，则必须配置过温关机的唤醒源，相关配置归类于唤醒源配置中

不同快充方案的 NTC 流程有差异，其中差异关系如下表所示：

表 5-1: NTC 功能-快充方案对照表

相关功能	单节快充	双节快充
阈值/参数配置	取 ts 的电压值，单位 mV	取 NTC 阻值，单位 Ω
NTC 温控相关唤醒中断配置	有	无
过温关机功能	有	无

相关功能	单节快充	双节快充
boost 放电过温/欠温	无	有
NTC 温控实现方式	硬件中断 + 软件控制	纯硬件控制

除外的功能各方案基本一致，因此以下内容以 A523-单节快充方案为例，其他方案如果有差异，会在后续《xx 方案-差异化》中详细说明：

5.1.7.1 软件流程设计

电池温控的软件控制流程如下，其中的温度为 ts 电压转化过来的电池温度：

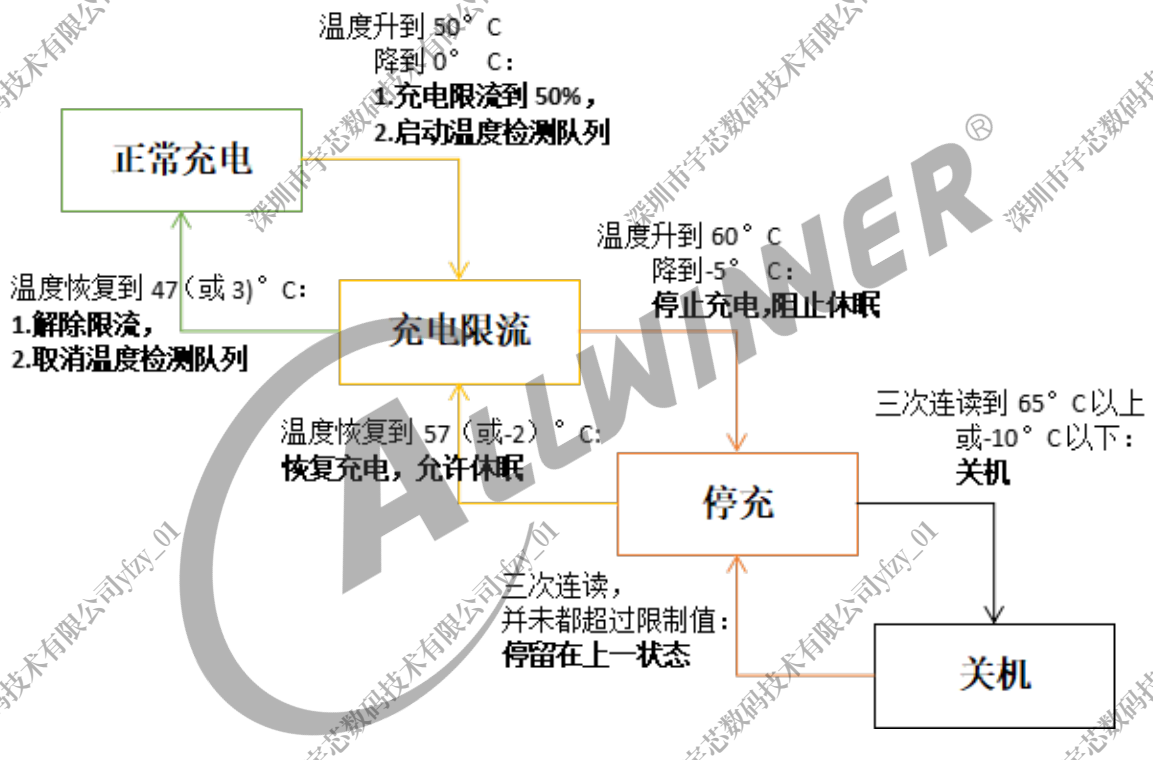

图 5-1: 单节快充充电限流流程设计

5.1.7.2 ntc、jeita 使能、jeita 限流参数

```
device/……/uboot-board.dts:
&power_sply {
    ntc_status = <2>;
};

device/……/board.dts:
bat_power_supply: bat-power-supply {
    ……
    pmu_bat_temp_enable = <0>;
    pmu_jetia_en      = <0>;
```

```
    pmu_jcool_ifall = <1>;
    pmu_jwarm_ifall = <1>;
    ……
};
```

```
ntc_status <u32>
    配置ts引脚电流开关，根据该属性决定是否开启，该属性不配置默认为0，仅在uboot生效。
    如果不使用ntc功能，则配置为0。
    如果使用ntc功能，则必须配置成1或2。
    0：关闭ts引脚的电，ntc功能失效
    1：开启ts引脚的电，ntc功能启用,在过温/欠温时直接关机
    2：开启ts引脚的电，ntc功能启用,在过温/欠温时进入等待状态，如果没接适配器则关机

pmu_bat_temp_enable <u32>
    设置电池温度检测、ntc是否使能，该设置必须与ntc_status保持一致。
    该属性不配置默认为0，仅在内核生效。
    0: disalbe
    1: enable

pmu_jetia_en <u32>
    设置jeita功能是否使能，需配置pmu_bat_temp_enable为1才生效。
    该属性不配置默认为0，仅在内核生效。
    0: disalbe
    1: enable

pmu_jcool_ifall <u32>
    低温电流限流降，需配置pmu_jetia_en为1才生效
    该属性不配置默认为50%，仅在内核生效。
    00：100%
    01：50%，电流降低到一半
    10：75%，电流降低到75%
    00：0%

pmu_jwarm_ifall <u32>
    高温电流限流降，需配置pmu_jetia_en为1才生效
    该属性不配置默认为50%，仅在内核生效。
    00：100%
    01：50%，电流降低到一半
    10：75%，电流降低到75%
    00：0%
```

5.1.7.3 电池温度参数相关配置

说明

由于 TS pin 电压寄存器存在上下限，因而电池温度参数尽量配置在一定范围内。
AXP2202：0 ~ 4,095 mV。

```
device/……/board.dts:
bat_power_supply: bat-power-supply {
    ……
    pmu_bat_ts_current = <40>;
    ……
    pmu_bat_temp_para1 = <2814>;
    pmu_bat_temp_para2 = <2202>;
    pmu_bat_temp_para3 = <1737>;
    pmu_bat_temp_para4 = <1381>;
```

```
    pmu_bat_temp_para5 = <1105>;
    pmu_bat_temp_para6 = <890>;
    pmu_bat_temp_para7 = <722>;
    pmu_bat_temp_para8 = <484>;
    pmu_bat_temp_para9 = <332>;
    pmu_bat_temp_para10 = <233>;
    pmu_bat_temp_para11 = <196>;
    pmu_bat_temp_para12 = <166>;
    pmu_bat_temp_para13 = <141>;
    pmu_bat_temp_para14 = <121>;
    pmu_bat_temp_para15 = <89>;
    pmu_bat_temp_para16 = <66>;
    ……
};

device/……/uboot-board.dts:
&charger0 {
    ntc_cur = <40>;
    pmu_bat_temp_para1 = <2814>;
    ……
    pmu_bat_temp_para16 = <66>;
};
```

```
pmu_bat_ts_current <u32>
    TS pin的电流大小配置，电流大小跟随AXP
    AXP2202/AXP2101：20uA/40uA/50uA/60uA 四档可配，默认50uA
    如若修改，下面的pmu_bat_temp_para计算时也要换成对应的TS电流

pmu_bat_temp_para1 <u32>
    电池包-25度对应的TS pin电压，单位：mV

pmu_bat_temp_para2 <u32>
    电池包-15度对应的TS pin电压，单位：mV

pmu_bat_temp_para3 <u32>
    电池包-10度对应的TS pin电压，单位：mV

pmu_bat_temp_para4 <u32>
    电池包-5度对应的TS pin电压，单位：mV

pmu_bat_temp_para5 <u32>
    电池包0度对应的TS pin电压，单位：mV

pmu_bat_temp_para6 <u32>
    电池包5度对应的TS pin电压，单位：mV

pmu_bat_temp_para7 <u32>
    电池包10度对应的TS pin电压，单位：mV

pmu_bat_temp_para8 <u32>
    电池包20度对应的TS pin电压，单位：mV

pmu_bat_temp_para9 <u32>
    电池包30度对应的TS pin电压，单位：mV

pmu_bat_temp_para10 <u32>
    电池包40度对应的TS pin电压，单位：mV

pmu_bat_temp_para11 <u32>
    电池包45度对应的TS pin电压，单位：mV
```

```
pmu_bat_temp_para12 <u32>
    电池包50度对应的TS pin电压，单位：mV

pmu_bat_temp_para13 <u32>
    电池包55度对应的TS pin电压，单位：mV

pmu_bat_temp_para14 <u32>
    电池包60度对应的TS pin电压，单位：mV

pmu_bat_temp_para15 <u32>
    电池包70度对应的TS pin电压，单位：mV

pmu_bat_temp_para16 <u32>
    电池包80度对应的TS pin电压，单位：mV
    **不同电池包的温敏电阻特性不一样，根据电池包的TS温敏电阻手册，找到pmu_bat_temp_para[1-16]对应温度点的电
      阻阻值，将阻值乘以pmu_bat_ts_current中的值得到的电压数值（单位：mV），将电压数值填进
      pmu_bat_temp_para[1-16]的节点中即可**

ntc_cur <u32>
    配置ts引脚电流，默认为0，该配置必须跟内核dts的一致

pmu_bat_temp_para[16] <u32>
    配置16个ntc参数，默认为0，该配置必须跟内核dts的一致
```

5.1.7.4 开关机温度限制（uboot）

开关机温度限制即只允许一定温度范围内的机器开机：

```
device/·····/uboot-board.dts:
&charger0 {
  safe_temp_H = <600>;
  safe_temp_L = <0xFFFFFFCE>;
};
```

```
safe_temp_H\safe_temp_L <u32>
    配置uboot阶段允许开机的温度范围，该范围为小于safe_temp_H，大于safe_temp_L
    该属性不配置默认为0，仅在uboot生效。
    注：此处填入的值为: 温度值 * 10，负数需要转化成32进制负数
    例：在-5°~60°才能开机
    safe_temp_H = <600>;
    safe_temp_L = <0xFFFFFFCE>;
```

5.1.7.5 触发限流、停充、关机的电压阈值

说明

由于寄存器存在上下限，因此 TS pin 电压阈值必须配置在一定范围内。
AXP2202 可配范围：
停充/关机过温：0 ~ 8160mV。
停充/关机欠温：0 ~ 510mV。
限流过温：0-4080mV。
限流欠温：0-2040mV。

说明

对于限流、停充、关机的阈值

过温情况下：关机 <= 停充 <= 限流欠温情况下：关机 >= 停充 >= 限流

```
device/·····/board.dts:
bat_power_supply: bat-power-supply {
  ·····
  pmu_bat_charge_ltf = <1105>;
  pmu_bat_charge_htf = <141>;
  pmu_bat_shutdown_ltf = <1381>;
  pmu_bat_shutdown_htf = <89>;
  pmu_jetia_cool = <722>;
  pmu_jetia_warm = <196>;
  ·····
};
```

```
pmu_bat_charge_ltf <u32>
    触发电池低温停充的TS pin电压阈值，单位：mV
    默认：1312mV
    范围：0-8160mV

pmu_bat_charge_htf <u32>
    触发电池高温停充的TS pin电压阈值，单位：mV
    默认：176mV
    范围：0-510mV

pmu_bat_shutdown_ltf <u32>
    非充电模式下，触发电池低温中断的TS pin电压阈值，单位：mV
    默认：1984mV
    范围：0-8160mV

pmu_bat_shutdown_htf <u32>
    非充电模式下，触发电池高温中断的TS pin电压阈值，单位：mV
    默认：152mV
    范围：0-510mV

pmu_jetia_cool <u32>
    触发电池低温限流/限压的TS pin电压阈值，单位：mV
    默认：880mV(10°C)
    范围：0-4080mV

pmu_jetia_warm <u32>
    触发电池高温限流/限压的TS pin电压阈值，单位：mV
    默认：240mV(45°C)
    范围：0-2040mV
```

5.1.8 其他配置

说明

视具体的 board.dts 属性节点而定，部分型号 PMU 驱动暂未支持该功能，为无效参数。

```
device/·····/board.dts:
bat_power_supply: bat-power-supply {
  ·····
  bmu_ext_save_charge_time = <0>;
```

```
    battery_exist_sets = <0>;
    bmu_ext_max_power = <0>;
    ……
};
```

```
**以下配置视具体的board.dts属性节点而定，部分方案暂未支持该功能**
**当前支持方案：A523-单节快充方案**
bmu_ext_save_charge_time <u32>
    充电多长时间后触发充电保护
    ETA6973/6974：10/5(单位h，是指完整充电流程)
    AXP519：12/24/48/72(单位h，是指恒流充电时间)

battery_exist_sets <u32>
    强制判断电池存在状态
    0：按照实际情况检测
    1：强制判断无电池
    2：强制判断有电池

bmu_ext_max_power <u32>
    高压充电芯片最高输入功率
```

5.1.9 双节快充方案-差异化

双节快充是指有两节电池的快充方案，以下介绍它和单节快充在配置和流程上的差异点。

目前双节快充方案包含：A523-双节快充方案、A733-双节快充方案

说明

以下如无特殊说明，则均以 A523-双节快充方案为例。

5.1.9.1 方案设计差异

方案上外挂独立电量计，并且外挂电量计由于存在上报电量的功能，其 IRQ 不能和系统的 IRQ 直连，需要单独接 GPIO.

电量计芯片，高压充电芯片，Type-C PD 控制器驱动和电源管理芯片四者的方案设计如下：

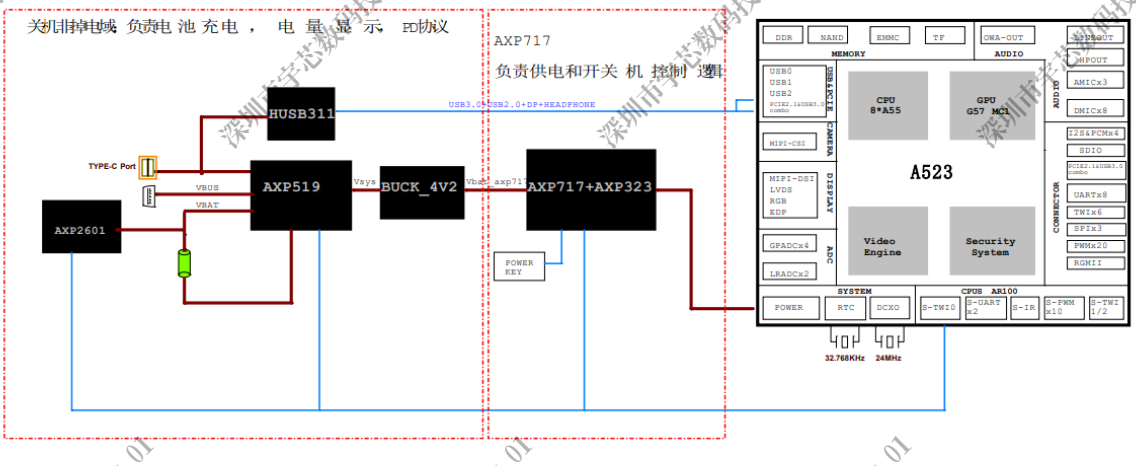

图 5-2: 双节快充硬件设计

5.1.9.2 驱动限流通路差异

其驱动限流通路如下所示，其中黄色线为 otg 输出时的路径，蓝色线为充电时的路径，橙色线则是充电/电池数据的流动路径

和单节快充相比，差异点在限流值的设置、电池状态信息的获取和传递上。

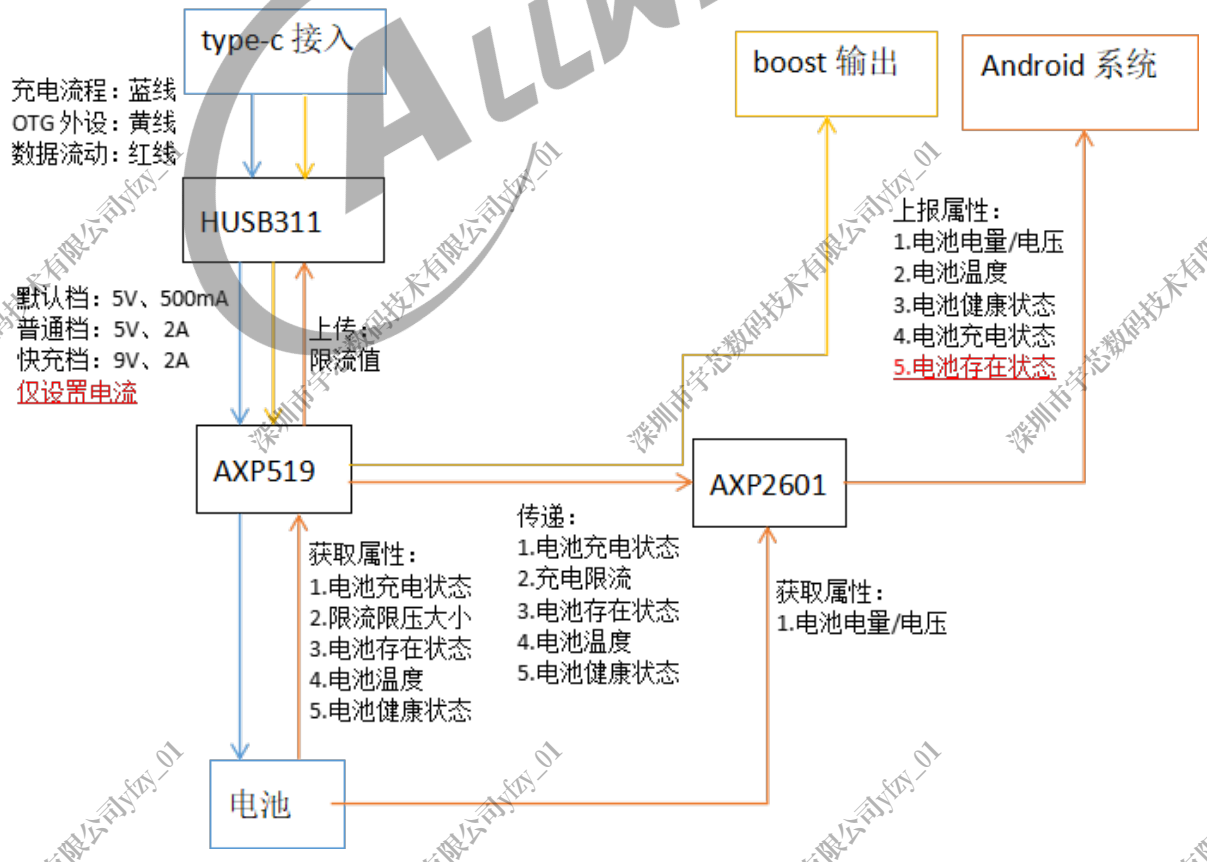

图 5-3: 驱动限流通路框架

5.1.9.3 驱动唤醒通路框架差异

其驱动唤醒通路如下所示，差异点在 type-c 中断变更为从 NMI 上报，而电池电量变化中断从非掉电域 gpio 上报，如图中的 pl 域的 gpio 口。

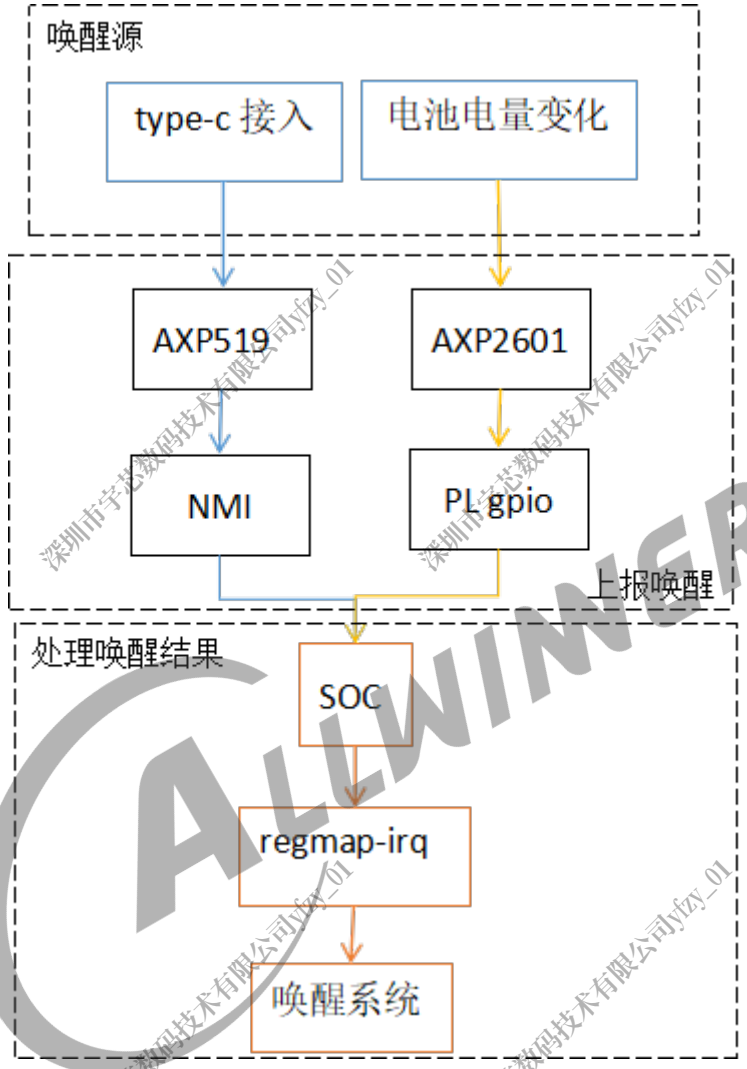

图 5-4: 驱动唤醒通路框架

5.1.9.4 menuconfig 配置差异

和单节快充相比，双节快充在高压充电芯片和电量计芯片配置上存在差异

```
Allwinner BSP  --->  Device Drivers  --->
   I2C Drivers   --->  <*> I2C Support for Allwinner SoCs
   PMIC Drivers  --->  <*> BMU_EXT PMICs with I2C
            --->  <*> AXP519 Power Supply Driver
            --->  <*> AXP2601 Power Supply Driver
```

5.1.9.5 Device Tree 基础配置差异

和单节快充相比，双节快充在高压充电芯片和电量计芯片配置上存在差异。

其中的差异仅为使用芯片型号和中断配置差异，因此不再单独列出。

5.1.9.6 Android 配置差异

除了在 device-common.mk 文件中增加属性外，以版型 a523-evb 为例，还需要另外增加以下修改:

```
device/softwinner/saturn/a523-evb/system/vendor_ramdisk.modules:
……
+axp519_charger_power.ko
+axp2601_battery.ko
……
```

```
system/vendor_ramdisk.modules:
    加载两个双节快充的ko
```

5.1.9.7 NTC 温控属性配置差异

和单节快充相比，双节快充没有过温/欠温关机，且由高压充电芯片（axp519）硬件控制停充和限流：

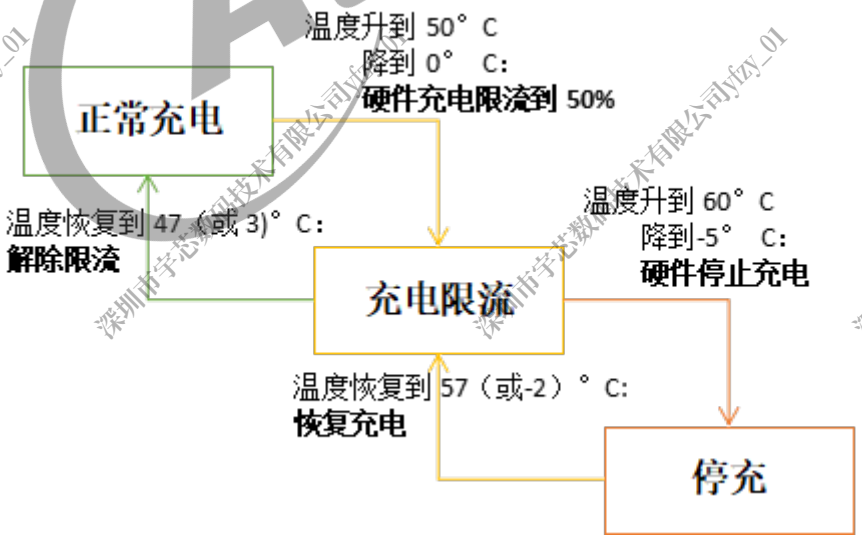

图 5-5: 双节快充过温欠温限流流程设计

其相关配置差异如下：

1. 双节快充的阈值和参数的配置单位为 Ω，可配置范围是 0 ~ 112,612 Ω。

2. 双节快充没有触发限流、关机的配置
3. 双节快充多一个 boost 放电过温/欠温的配置值

温度/℃	NTC 阻值/k	NTC 电压/V	NTC 电流/uA
-25	86.43	1.728	20
-20	67.77	1.355	20
-15	53.41	1.068	20
-10	42.47	1.698	40
-5	33.9	1.356	40
0	27.28	1.091	40
5	22.05	1.764	80
10	17.96	1.436	80
15	14.69	1.175	80
20	12.09	0.967	80
25	10.00	0.800	80
30	8.313	0.665	80
35	6.940	0.555	80
40	5.827	0.466	80
45	4.911	0.392	80
50	4.160	0.332	80
55	3.536	0.282	80
60	3.020	0.241	80
65	2.588	0.207	80

图 5-6: 双节快充 NTC 阻值/电压/电流和温度值的对应表

其完整配置如下：

```
device/……/board.dts:
qc_bat_power_supply: qc_bat_power_supply {
    ……
    pmu_bat_temp_enable = <1>;
    pmu_bat_charge_ltf = <27280>;//0
    pmu_bat_charge_htf = <3536>;//55
    pmu_bat_work_ltf = <27280>;//0
    pmu_bat_work_htf = <3536>;//55
    pmu_bat_temp_para1 = <86430>;//-25
    pmu_bat_temp_para2 = <53410>;//-15
    pmu_bat_temp_para3 = <42470>;//-10
    pmu_bat_temp_para4 = <33900>;//-5
    pmu_bat_temp_para5 = <27280>;//0
    pmu_bat_temp_para6 = <22050>;//5
    pmu_bat_temp_para7 = <17960>;//10
    pmu_bat_temp_para8 = <14690>;//15
    pmu_bat_temp_para9 = <12090>;//20
    pmu_bat_temp_para10 = <10000>;//25
    pmu_bat_temp_para11 = <8313>;//30
    pmu_bat_temp_para12 = <5827>;//40
    pmu_bat_temp_para13 = <4911>;//45
    pmu_bat_temp_para14 = <4160>;//50
```

```
    pmu_bat_temp_para15 = <3536>;//55
    pmu_bat_temp_para16 = <2588>;//65
    ......
};
```

pmu_bat_charge_ltf <u32>
触发电池低温停充的TS pin电阻阈值，单位：Ω
由实际停充温度值换算而来，其门限范围为：
-5°C、0°C、5°C、10°C

pmu_bat_charge_htf <u32>
触发电池高温停充的TS pin电阻阈值，单位：Ω
由实际停充温度值换算而来，其门限范围为：
40°C、45°C、50°C、55°C

pmu_bat_work_ltf <u32>
在boost输出模式下，触发电池低温中断的TS pin电阻阈值，单位：Ω
由实际停充温度值换算而来，其门限范围为：
40°C、45°C、50°C、55°C

pmu_bat_work_htf <u32>
在boost输出模式下，触发电池高温中断的TS pin电阻阈值，单位：Ω
由实际停充温度值换算而来，其门限范围为：
40°C、45°C、50°C、55°C

pmu_bat_temp_para1 ~ pmu_bat_temp_para16 <u32>; //<u32>°C
电池在各对应温度下的ntc阻值，单位Ω

著作权声明

本文档及内容受著作权法保护，其著作权由珠海全志科技股份有限公司（“全志”）拥有并保留一切权利。

本文档是全志的原创作品和版权财产，未经全志书面许可，任何单位和个人不得擅自摘抄、复制、修改、发表或传播本文档内容的部分或全部，且不得以任何形式传播。

商标声明





（不完全列举）均为珠海全志科技股份有限公司的商标或者注册商标。在本文档描述的产品中出现的其它商标，产品名称，和服务名称，均由其各自所有人拥有。

免责声明

您购买的产品、服务或特性应受您与珠海全志科技股份有限公司（“全志”）之间签署的商业合同和条款的约束。本文档中描述的全部或部分产品、服务或特性可能不在您所购买或使用的范围内。使用前请认真阅读合同条款和相关说明，并严格遵循本文档的使用说明。您将自行承担任何不当使用行为（包括但不限于如超压，超频，超温使用）造成的不利后果，全志概不负责。

本文档作为使用指导仅供参考。由于产品版本升级或其他原因，本文档内容有可能修改，如有变更，恕不另行通知。全志尽全力在本文档中提供准确的信息，但并不确保内容完全没有错误，因使用本文档而发生损害（包括但不限于间接的、偶然的、特殊的损失）或发生侵犯第三方权利事件，全志概不负责。本文档中的所有陈述、信息和建议并不构成任何明示或暗示的保证或承诺。

本文档未以明示或暗示或其他方式授予全志的任何专利或知识产权。在您实施方案或使用产品的过程中，可能需要获得第三方的权利许可。请您自行向第三方权利人获取相关的许可。全志不承担也不代为支付任何关于获取第三方许可的许可费或版税（专利税）。全志不对您所使用的第三方许可技术做出任何保证、赔偿或承担其他义务。