Smart Phone Battery Charging FAQ

Version:	1.0
Release date:	2012-05-0821

Specifications are subject to change without notice.

Document Revision History

Revision	Date	Author	Description
1.0	2012-05-03	Sangmei Liu	Initial Draft

Table of Contents

1 Introduction

This document provides basic battery charging introduction and customization guidelines based on Mediatek Smartphone platform MT6575/MT6515/MT6577/MT6517. We list the common codebase notes according to source code and related advanced customer training slide (Battery Charging module). The details will be described in below sections.

If you have any question about corresponding chapters or any items you think should be added, please contact us at any time you are convenient. This document will be updated continuously under your suggestion and great support.

2 Codebase Notes

The chapter provides more detailed notes of battery customization files. Common definition is for customization items both in uboot and kernel cust_battery.h. Separate parts are also listed followed that.

2.1 Common definition

2.1.1 cust_charging_current_enum

这里定义 MT6575/MT6577 charging current 枚举变量。USB charging current 默认使用 450mA 电流档位，参考 USBIF 规范（小于 500mA）,不建议修改. AC(Adapter Charger)充电电流默认 650mA, 可灵活克制化，大于 800mA 充电考虑系统散热问题，建议使用 switching charger IC, switching charger IC 最新资讯，请咨询我司 HW 同仁。

```
typedef enum
{
        Cust_CC_1600MA = 0x0,
        Cust_CC_1500MA = 0x1,
        Cust_CC_1400MA = 0x2,
        Cust_CC_1300MA = 0x3,
        Cust_CC_1200MA = 0x4,
        Cust_CC_1100MA = 0x5,
        Cust_CC_1000MA = 0x6,
        Cust_CC_900MA  = 0x7,
        Cust_CC_800MA  = 0x8,
        Cust_CC_700MA  = 0x9,
        Cust_CC_650MA  = 0xA,
        Cust_CC_550MA  = 0xB,
        Cust_CC_450MA  = 0xC,
        Cust_CC_400MA  = 0xD,
        Cust_CC_200MA  = 0xE,
        Cust_CC_70MA   = 0xF,
        Cust_CC_0MA    = 0xDD
}cust_charging_current_enum;
```

2.1.2 Battery Voltage and Percentage Mapping Table

结构体 VBAT_TO_PERCENT 定电池电压到电量的转换。Batt_VoltToPercent_Table 用于 uboot 关机充电获取电池电量，即传统电压 mapping 电量方法。而在 kernel 开机充电 flow, default 开启 fuel guage 采用混合算法及库仑法获取较精确的电池电量。

```
typedef struct{
        unsigned int BattVolt;
        unsigned int BattPercent;
}VBAT_TO_PERCENT;
/* Battery Voltage and Percentage Mapping Table */
VBAT_TO_PERCENT Batt_VoltToPercent_Table[] = {
        /*BattVolt,BattPercent*/
        {3400,0},
        {3641,10},
        {3708,20},
        {3741,30},
        {3765,40},
        {3793,50},
        {3836,60},
        {3891,70},
        {3960,80},
        {4044,90},
        {4183,100},   //{4200,100},
};
```

Notes: 我们不建议该 table 做较大改动，for specific scenario, 您可以微调最后一组 mapping item from {4183,100} to {4200,100}. 例如，添加 logo show charging full screen, 可能会出现开机后电量较关机 show 100%偏差较大问题。

2.1.3 Precise Tunning

为获取平滑充电曲线及稳定的电池电压等讯息，导入移动平均算法。也就是，当前值会取前 X 分钟/秒平均值，目前方案中，电池电压，充电电流，及采用传统电压 mapping 电量方式得到的电量百分比有使用到它。 X = BATTERY_AVERAGE_SIZE*10/60

```
#define BATTERY_AVERAGE_SIZE   12  //default 2 分钟平均值
```

Notes: 在 uboot/cust_battery.h 该宏的定义应是 kernel/cust_battery.h 值的 10 倍，比如当前 design：

```
#define BATTERY_AVERAGE_SIZE   120
```

2.1.4 Charger Related Setting

下面定义与 HW design 关系紧密：

```
#define R_CURRENT_SENSE 2                         // 0.2 Ohm
#define R_BAT_SENSE 4                             // times of voltage
#define R_I_SENSE 4                               // times of voltage
#define R_CHARGER_1 330
#define R_CHARGER_2 39
#define R_CHARGER_SENSE ((R_CHARGER_1+R_CHARGER_2)/R_CHARGER_2)
```

Note: 以上宏不需要改动，R_CHARGER_SENSE 目前没有使用到。

```
#define V_CHARGER_MAX 6000                        // 6 V
#define V_CHARGER_MIN 4400                        // 4.4 V
#define V_CHARGER_ENABLE 0                        // 1:ON , 0:OFF
```

Note:当 charger 电压大于 V_CHARGER_MAX, SW 关充电；当 V_CHARGER_ENABLE 置为 1, 且 charger 电压低于 V_CHARGER_MIN, SW 关充电。考虑线路阻抗及电池内阻，我们不建议开启宏 V_CHARGER_ENABLE

2.1.5 Temperature Related Setting

下面是电池温度克制化部分：

```
#define RBAT_PULL_UP_R          24000   //24K 上拉电阻
//#define RBAT_PULL_UP_VOLT        2500   //MT6573 平台定义 2.5v 上拉电压
#define RBAT_PULL_UP_VOLT        1200  // MT6575/77 平台定义 1.2v 上拉电压，不可修改
//#define TBAT_OVER_CRITICAL_LOW     68237
#define TBAT_OVER_CRITICAL_LOW     483954
#define BAT_TEMP_PROTECT_ENABLE    0
#define BAT_NTC_10 0
#define BAT_NTC_47 1
```

Notes: 1. TBAT_OVER_CRITICAL 是 Batt_Temperature_Table 中-20 度对应的电阻值，单位为欧姆。因此，使能 BAT_NTC_47---→ define TBAT_OVER_CRITICAL_LOW 483954

使能 BAT_NTC_10 --→ define TBAT_OVER_CRITICAL_LOW 68237

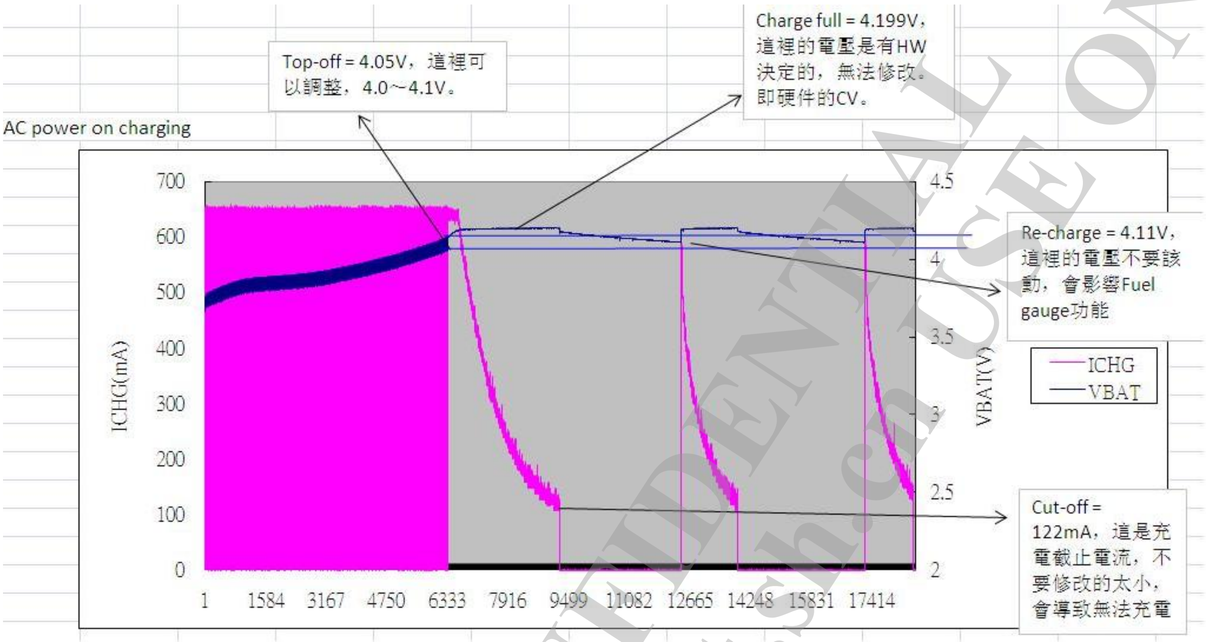
2. 我们无法从这里得到主板温度讯息，FYR

3. BAT_TEMP_PROTECT_ENABLE 用于使能低温电池充电功能。高温充电会对系统造成永久性损害，SW 会做关充电的行为，详细软体控制请参考 API--- BAT_CheckBatteryStatus().

2.1.6 Recharging Issue

当恒压充电电流低于 CHARGING_FULL_CURRENT, SW 判断充电已满 dis-charging。当 VBAT 电压低于 RECHARGING_VOLTAGE，会进入 Pre-CC state 开始回充，user 在回充过程会 always 看到 battery SOC 为 100%。

Note: 回充电压不建议做大调整，会影响 fuel gauge update 电量。若 recharging 电压调过低，极端情况会出现 charger 拔出，电量从 100%跳变到某一较低百分比，用户体验效果较差。



Top-off = 4.05V，這裡可以調整，4.0~4.1V。

Charge full = 4.199V，這裡的電壓是有HW決定的，無法修改。即硬件的CV。

Re-charge = 4.11V，這裡的電壓不要該動，會影響Fuel gauge功能

Cut-off = 122mA，這是充電截止電流，不要修改的太小，會導致無法充電

Figure 1. example of charging behavior and notes

2.2 Specific definition

2.2.1 Battery warning mechanism

在 kernel battery driver 定义有 charging error warning 机制，宏开关定义在 kernel/cust_battery.h 文档中(uboot 暂无该 mechanism)

```
/* Battery Notify */
#define BATTERY_NOTIFY_CASE_0001
#define BATTERY_NOTIFY_CASE_0002
#define BATTERY_NOTIFY_CASE_0003
#define BATTERY_NOTIFY_CASE_0004
#define BATTERY_NOTIFY_CASE_0005
```

各宏定义对应 case 如表：

CASE	Description	Condition
CASE_0000	Default	Nothing
CASE_0001	Charger OVP	V_CHR>6V
CASE_0002	High battery temperature	>50degree
CASE_0003	High charging current	I_SENSE>1A
CASE_0004	High battery voltage	V_BATSNS>4.35V
CASE_0005	Safety timer Timeout	>24hr does not charge full

其中 CASE_0003 当充电电流大于 1A，会 pop out warning window for end user, SW 不会关充电的行为，因此若出現误报高温问题，以下建议供您参考：(1)在 kernel/cust_battery.h 文档中 mark 掉 BATTERY_NOTIFY_CASE_0003; (2)修改 over charging current 阈值，default 1A, SW control API---mt_battery_notify_check, and test again.

2.2.2 Fuel Gauge Method

上面有提及，kernel battery driver 使用 hybrid method 和 coulomb counting method 来 update battery SOC 给 battery service. 若 HW 上没有支持，也请 SW disable 该 feature, 方法如下：

If you do not want to use fuel gauge function on MT657* solution,

Firstly, Please set gForceADCsolution from 0 to 1 in file

mediatek/platform/mt657*/kernel/drivers/power/mt657*_battery.c

Secondly, Please set gFG_booting_counter_I_FLAG from 0 to 2 in file

mediatek/platform/mt657*/kernel/drivers/power/mt657*_fuel_gauge.c

详细原理及克制化部分，请参考 MT6575_15 Fuel gauge Application Notes.pptx 和 Advanced Customer Training Battery Charging MT6575 4.0.pptx

2.2.3 Uboot charging logo

Uboot charging logo 存放 path: mediatek/custom/common/uboot/logo/{resolution}，最后打包到 logo.bin 档。调用接口请参考 mt657*_bat.c 文件中 uboot_charging_display.
若需要添加一张新 logo，请 follow 下面 steps:

1. 在 mediatek/config/${project}/ProjectConfig.mk 查看 resolution ，参考文件中的 BOOT_LOGO = *****(cmcc_wvga)
2. 在 mediatek/custom/common/uboot/logo/${resolution}/ 目录下添加对应的图片
3. 打开 在 mediatek/custom/common/uboot/logo 目录下的 update 文件,按格式轉換 bmp_to_raw, 壓縮 zpipe，mkimage 的順序
   仿照格式(图片命名及大小写必须一致)添加新的图片的打包行为，如
   ./tool/bmp_to_raw ./temp2.raw ./$p/"${p}_charging_warning".bmp
   最后要在 zpipe 那一行，添加打包出来的 temp 文件的名称(请 check logo.bin 是否有更新)
4. 程序里可以调用 show_logo(index)的方式来显示图片,具体：

   在 mediatek\platform\mt657*\uboot\mt657*_logo.c 和 mediatek\source\external\ipod\bootlogo.c 这两支文件中参考 mt65xx_disp_show_boot_logo 添加对应 show logo API，暂定添加的 API 为 mt65xx_disp_show_charger_warning_logo
5. 在 mt657*_bat.c 程序中适当位置调用 mt65xx_disp_show_charging_warning_logo

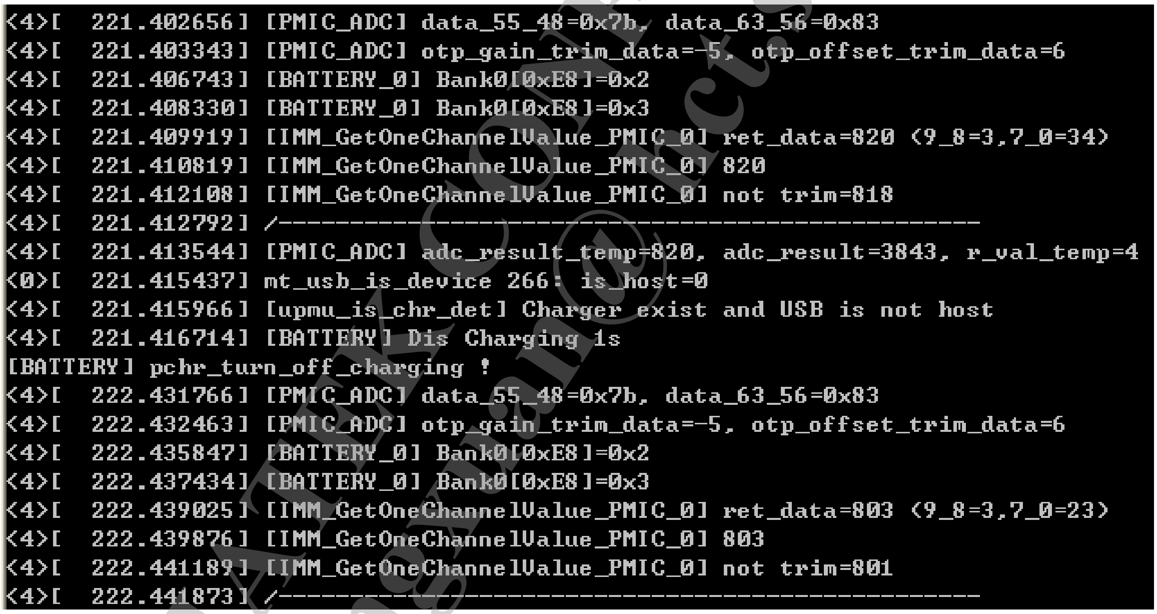
3 Debugging and Configuration

The chapter includes runtime check the PMIC register values & log sample

3.1 Runtime battery logging entry

若有需要在程式執行時得知 charging flow，可以在 shell 下底下的 command，就可以获取整个 charging state 资讯：

Path : cd /proc
Enable logging : echo 2 > batdrv_log or echo 1 > batdrv_log
echo 1 > fgadc_log
Disable logging : echo 0 > batdrv_log
Demo:



```
<4>[  221.402656] [PMIC_ADC] data_55_48=0x7b, data_63_56=0x83
<4>[  221.403343] [PMIC_ADC] otp_gain_trim_data=-5, otp_offset_trim_data=6
<4>[  221.406743] [BATTERY_0] Bank0[0xE8]=0x2
<4>[  221.408330] [BATTERY_0] Bank0[0xE8]=0x3
<4>[  221.409919] [IMM_GetOneChannelValue_PMIC_0] ret_data=820 (9_8=3,7_0=34)
<4>[  221.410819] [IMM_GetOneChannelValue_PMIC_0] 820
<4>[  221.412108] [IMM_GetOneChannelValue_PMIC_0] not trim=818
<4>[  221.412792] /-------------------------------------------------
<4>[  221.413544] [PMIC_ADC] adc_result_temp=820, adc_result=3843, r_val_temp=4
<0>[  221.415437] mt_usb_is_device 266: is_host=0
<4>[  221.415966] [upmu_is_chr_det] Charger exist and USB is not host
<4>[  221.416714] [BATTERY] Dis Charging 1s
[BATTERY] pchr_turn_off_charging !
<4>[  222.431766] [PMIC_ADC] data_55_48=0x7b, data_63_56=0x83
<4>[  222.432463] [PMIC_ADC] otp_gain_trim_data=-5, otp_offset_trim_data=6
<4>[  222.435847] [BATTERY_0] Bank0[0xE8]=0x2
<4>[  222.437434] [BATTERY_0] Bank0[0xE8]=0x3
<4>[  222.439025] [IMM_GetOneChannelValue_PMIC_0] ret_data=803 (9_8=3,7_0=23)
<4>[  222.439876] [IMM_GetOneChannelValue_PMIC_0] 803
<4>[  222.441189] [IMM_GetOneChannelValue_PMIC_0] not trim=801
<4>[  222.441873] /-------------------------------------------------
```

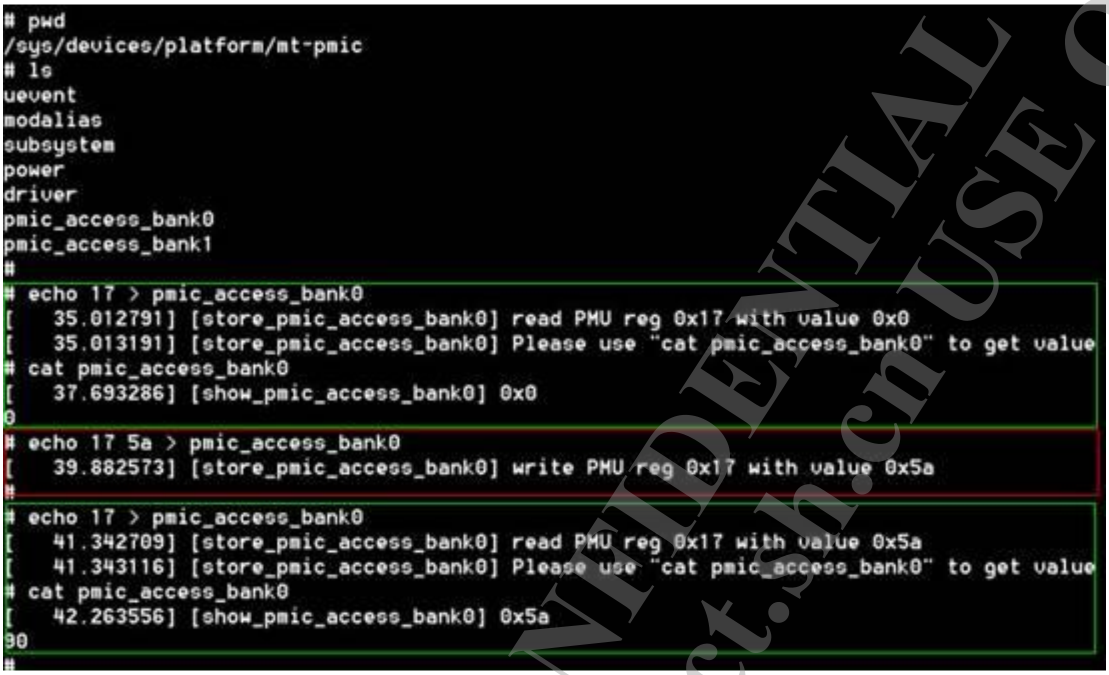
3.2 Runtime check PMIC register values

若有需要在程式執行時得知 PMIC register 的 value，可以在 shell 下底下的 command，就可以對 PMIC 進行 read / write：

Path : /sys/devices/platform/mt-pmic
Read CMD : echo Address[HEX based] > [pmic_access_bank0|pmic_access_bank1]
Write CMD : echo Address[HEX based] Value[HEX based] > [pmic_access_bank0|pmic_access_bank1]

Example : read 0x17 of bank0 ➔ echo 17 > pmic_access_bank0
cat pmic_access_bank0

Example : write 0x5a to 0x17 of bank0 ➔ echo 17 5a > pmic_access_bank0 cat pmic_access_bank0



```
# pwd
/sys/devices/platform/mt-pmic
# ls
uevent
modalias
subsystem
power
driver
pmic_access_bank0
pmic_access_bank1
#
# echo 17 > pmic_access_bank0
[   35.012791] [store_pmic_access_bank0] read PMU reg 0x17 with value 0x0
[   35.013191] [store_pmic_access_bank0] Please use "cat pmic_access_bank0" to get value
# cat pmic_access_bank0
[   37.693286] [show_pmic_access_bank0] 0x0
0
# echo 17 5a > pmic_access_bank0
[   39.882573] [store_pmic_access_bank0] write PMU reg 0x17 with value 0x5a
#
# echo 17 > pmic_access_bank0
[   41.342709] [store_pmic_access_bank0] read PMU reg 0x17 with value 0x5a
[   41.343116] [store_pmic_access_bank0] Please use "cat pmic_access_bank0" to get value
# cat pmic_access_bank0
[   42.263556] [show_pmic_access_bank0] 0x5a
90
#
```

3.3 Log sample

3.3.1 boot up check battery exist

MT6575 之后引入 Download w/o battery, 此时插入 USB， HW default 开启充电功能，timeout=8s. 除此，另外一种可能出现的 scenario 是 插入 USB/Charger, 外接电源（假电池，BATON pin 不接）模拟电池，则开机过程中，程式卡在如下无穷 loop 中

```
[hw_check_battery+] Bank0[0x38]=0x34
[hw_check_battery+] Bank0[0xE9]=0x1A
[hw_check_battery] No Battery
[hw_check_battery-] Bank0[0x38]=0x34
[hw_check_battery-] Bank0[0xE9]=0x1A
[pl_kick_chr_wdt] done
[hw_check_battery+] Bank0[0x38]=0x34
[hw_check_battery+] Bank0[0xE9]=0x1A
[hw_check_battery] No Battery
[hw_check_battery-] Bank0[0x38]=0x34
[hw_check_battery-] Bank0[0xE9]=0x1A
[pl_kick_chr_wdt] done
```

BATON Pin(有 detect battery exist and battery temperature 两种功能)不能悬空的限制，主要是系统的 power supply 是 battery, 电池不在必须断电，这部分 SW control 是在 platform_post_init 中，真实环境中并不会出现，且电量显示必须依照真实电池进行克制化，因此该 scenario 下，建议您使用真实电池 or 下拉 BATON pin 到 GND 便于开机调试，属测试手法 issue.

3.3.2 Low power boot up

底下是 low battery check 的地方：

```
In  mediatek\platform\mt657*\uboot\mt657*_bat.c

#define BATTERY_LOWVOL_THRESOLD            3450

        if ( (rtc_boot_check(false)||mtk_wdt_boot_check()==WDT_BY_PASS_PWK_REBOOT) && BMT_status.bat_vol >=
BATTERY_LOWVOL_THRESOLD)
      //if (BMT_status.bat_vol >= BATTERY_LOWVOL_THRESOLD)
      {
         printf("[BATTERY] battery voltage(%dmV) >= CLV ! Boot Linux Kernel !! \n\r",BMT_status.bat_vol);
         sc_mod_exit();
         return;
      }
      else
      {
         printf("[BATTERY] battery voltage(%dmV) <= CLV ! Can not Boot Linux Kernel !! \n\r",BMT_status.bat_vol);
         pchr_turn_off_charging();
         #ifndef NO_POWER_OFF
         mt6575_power_off();
         #endif
         while(1)
         {
            printf("If you see the log, please check with RTC power off API\n\r");
         }
```

3.3.3 Normal mode log analysis

请在 log 中 search “battery” keyword, 通过 enable logging,可以从 log 中获取更多讯息

For example:

```
[   40.206678] [BATTERY:AVG:fan5405] BatTemp:29 Vbat:3758 VBatSen:3792 SOC:51 ChrDet:0
Vchrin:66 Icharging:0 ChrType:0 USBstate:1
```

29：代表 SW 读到的电池温度

3758：代表 SW 平均算法得到的电池电压，单位 mV

3792：SW 实时从 ADC 读到的 BATSENSE 电压

51:电量百分比(%) 0: 代表 charger exist(1)/not exist(0)

66:SW 读取 charger ADC channel value

0: Icharging,SW 平均算法得到的充电电流值

0: charger type, NONSTANDARD_CHARGER/ STANDARD_HOST/ CHARGING_HOST/ STANDARD_CHARGER   1: USB state, not used USB config now

```
[ 532.865557] [FGADC] 1,5353,0,16,3718,15,14,17,1309,-119,0,0,1000,0,3842,1000,
85,84,5695
```

1/0：代表电池充電/放電動作

5353：代表充電或者放電的電流大小，這裡表示 535.3mA

0：設計之初定義的開關值，表示 HW 的 counter 是否有使用，0 表示 disable

16：表示 coulomb counter 統計到的電量，單位是 mAh，這裡 “+”表示充電， “-”表示放電

3718：這裡 ZCV 回推電壓，這個是平均值，10s 採一次，會進行 20 次然後平均電壓。

3.3.4 Not charging current issue debug

若在开发过程中出现无法充电的行为，以下分析方向供您参考：

（1） 请您先用真实电池测试充电功能是否正常，用程控电源模拟电池（即假电池）测试，由于 BATON pin 悬空，会出现不能充电的现象

（2） 请参考以上 debug 方法，double check log 中 battery temperature 是否在正常范围内（MIN_CHARGE_TEMPERATURE~MAX_CHARGE_TEMPERATURE）

（3） Log 中会有提示“BAD Battery status... Charging Stop”，请您根据 log 提示，看是否卡住 BAT_CheckBatteryStatus 中 charging protection 的某一条件

（4） 请您 debug 时，用 AC and USB 都做下测试，若以上均不能得到无法充电的原因，请 eservice 联系我们

3.3.5 High power consumption issue debug

以下是我司内部 debug，在 MT6575 & MT6577 目前耗电优化数据：

MTK Internal Tracking Status					
ALPS.ICS.MP.p49	AC charging and system on	USB charging and system on	AC charging and system off	USB charging and system off	Note
eagle75v1_2_ori	29mA	68mA	51mA	51mA	Orginal
eagle75v1_2_led	17mA	56mA	40mA	40mA	Optimize led
eagle75v1_2_led_26M	17mA	56mA	19mA	19mA	Optimize led + Dynamic turn on/off 26Mhz

其中 LED 采用 register mode control, CHR_Step=4mA, 若贵司 project HW 量测数据偏差较大，请 eservice 联系我们(maybe SW or HW issue，较难直接归类)，thanks

另外，我司在 Android 4.0 上有针对 USB power on charging 开发 USB charger only mode，勾选此项，耗电数据跟 AC power on 是一致的，知悉!

3.3.6 AC/USB charger type detection issue debug

我司提供的 solution，采用 BC1.1 circuit 来识别不同的 charger type: STANDARD_CHARGER/NONSTANDARD_CHARGER/CHARGING_HOST/STANDARD_HOST, 其中前两种归为 Adapter Charger(标准充电器和非标准充电器)，后两种为 USB cable。Charger type detection 动作在 preloader/uboot/kernel 中都有，参考 API: hw_charger_type_detection().

若您有遇到 AC/USB 误判问题，也请提供相关 log，并说明测试方法，联系我司，thanks.

3.4 Source Code & File Description

Describe your source code structure along with its file description as follows:

File	Description(sample as MT6575)
alps\mediatek\platform\mt6575\kernel\drivers\power	
mt6575_battery.c	The implementation of battery charging related APIs.
mt6575_fuel_gauge.c	The implementation of fuel gauge related APIs.
alps\mediatek\platform\mt6575\kernel\drivers\power	
mt6575_battery.h	The battery charging related settings (internal).
alps\mediatek\custom\${*project_name*}\kernel\battery\battery	
cust_battery.h	The battery charging related settings (customer).
cust_fuel_gauge.h	The fuel gauge related settings (customer).

3.5 Switching Charger Parts

Mediatek is preparing for it, and we will provide some training documents for you ASAP.