Panchip Microelectronics Co., Ltd.

PAN211x 芯片版本升级软件修改点说明

当前版本: 1.0

发布日期: 2025.3

上海磐启微电子有限公司

地址：上海张江高科技园区盛夏路 666 号 D 栋 3 楼

联系电话：021-50802371

网址：http://www.panchip.com

文档说明

由于版本升级或存在其他原因，本文档内容会不定期进行更新。除非另有约定，本文档内容仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

商标

免责声明

本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，磐启微电子有限公司对本文档内容不做任何明示或暗示的声明或保证。

修订历史

版本	修订时间	描述
V1.0	2025.3	首次发布

目录

1 芯片版本更新概述 3
2 配置参数更改 4
2.1 更改目标 4
2.2 配置参数导出方法 4
3 增加工厂校准函数 5
3.1 修改 PAN211_Err_t PAN211_FactoryCalibration(void)函数 5
3.2 增加 PAN211_Err_t PAN211_FactoryCalibration(void)函数 6
3.3 增加 uint8_t Factory_Read(uint8_t addr)函数 6
4 初始化函数更改 8
4.1 增加 cons_reg_en 使能配置 8
4.2 增加晶振频率配置 8
4.3 增加相关宏定义配置 9

1 芯片版本更新概述

为了进一步优化完善产品特性，我司对 PAN211x(SOP8)芯片做了性能升级，增加支持最大 9dBm 的输出功率配置，其他功能不变，后续以 B 版本出货为主。

改版后，芯片的软件功率档位配置有更新，软件 SDK 有所不同，

PAN211x SDK V1.2.0 正式版本，适用于 **A 版本**（仅 SOP8 封装），

PAN211x SDK V2.1.0 或者以上正式版本，适用于 **B 版本**（仅 SOP8 封装）。

注意：

1、 为了和之前旧版货物区分，改版后的芯片丝印做了区分。

第二行丝印区分方式如下：

A 版本丝印对应 2110BPA1A，支持最大输出功率 7dBm；

B 版本丝印对应 2110CPA1B，支持最大输出功率 9dBm；

2、 PAN211x-SDK 的下载地址：

https://docs.panchip.com/pan211xdk-doc/05_resources/index.html

2 配置参数更改

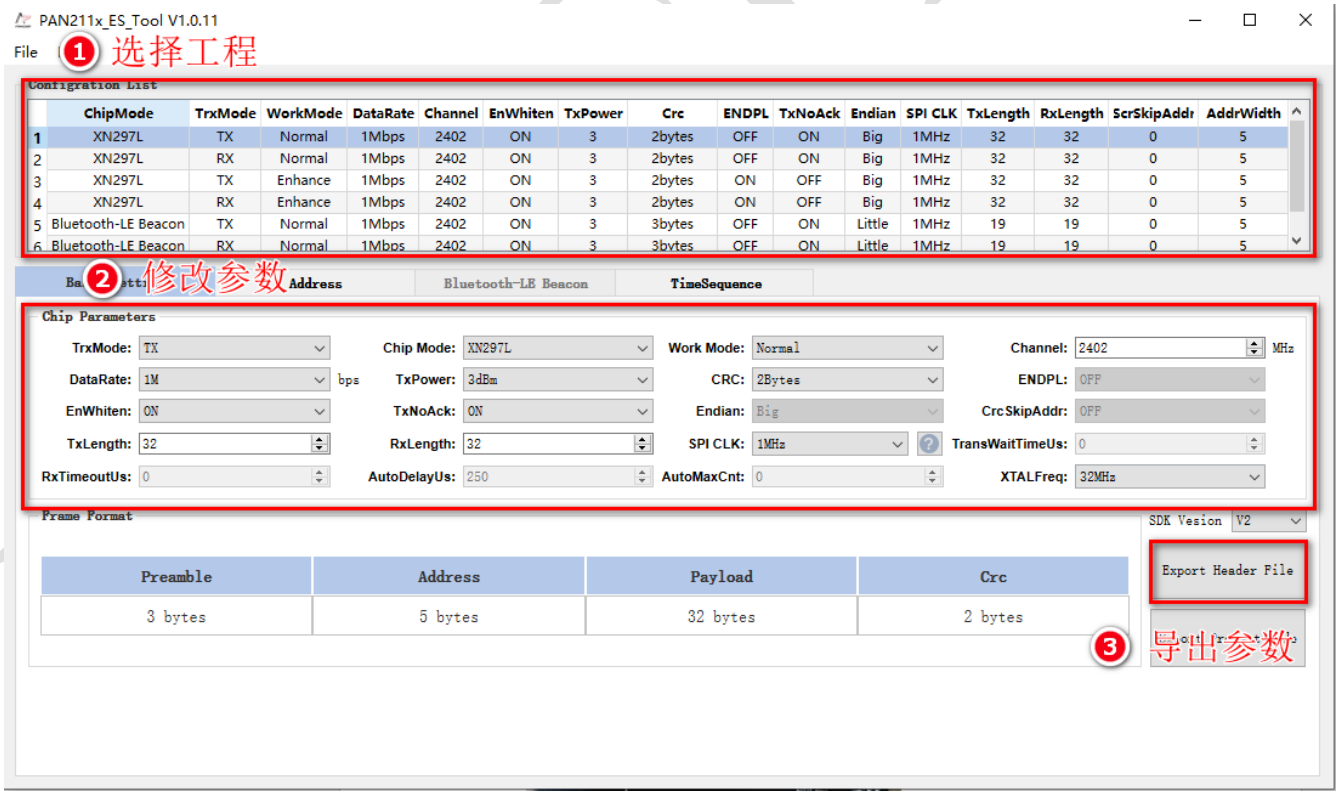
2.1 更改目标

从 MPA（SDK_V1.0.0 或 SDK_V1.1.0 或 SDK_V1.2.0）升级到 MPB（SDK_V2.1.x）需要修改 PAN211_Err_t PAN211_Init(void)中的 gPN211Page0Table 和 gPN211Page1Table。

gPN211Page0Table 和 gPN211Page1Table 从代码导出工具中导出。

2.2 配置参数导出方法

使用最新 PAN211x_ES_TOOL 代码导出工具，导出 easy_rf.h，替换目标例程 src 文件夹中的 easy_rf.h。



3 增加工厂校准函数

3.1 修改 PAN211_Err_t PAN211_FactoryCalibration(void) 函数

从 MPA(SDK_V1.2.0)升级到 MPB(SDK_V2.1.x)需要修改 MPA(SDK_V1.2.0)原有 PAN211_Err_t PAN211_FactoryCalibration(void)函数。

SDK_V2.1.x 的校准函数如下：

```
PAN211_Err_t PAN211_FactoryCalibration(void)
{
    uint8_t val[6] = {0};

    PAN211_WriteRegUnchecked(0x73, 0x01);
    val[2] = Factory_Read(0x02);
    val[5] = Factory_Read(0x05);

    if ((val[5] >> 6) != 2 || (val[2] & 0x0F) != 1)
    {
        PAN211_SetPage(1);
        PAN211_WriteReg(0x05, 0x01);
        PAN211_SetPage(0);
        return PAN211_ERR;
    }

    val[1] = Factory_Read(0x01);
    val[3] = Factory_Read(0x03);
    val[4] = Factory_Read(0x04);

    PAN211_SetPage(1);
    /* Recover cons_reg_en here */
    PAN211_WriteReg(0x05, 0x01);
    /* LDO_ANA_TRIM: 0x01[7:4]->P1 0x48[7:4] */
    /* LDO_RFFE_TRIM: 0x01[4:0]->P1 0x48[4:0] */
    PAN211_WriteReg(0x48, val[1]);
    /* IPOLY_TRIM: 0x02[7:5]->P1 0x47[6:4] */
    PAN211_WriteRegBits(0x47, (val[2] >> 5) & 0x07, BITMASK_6_4);
    /* LDO_HP_TRIM[3:0]: 0x03[3:0] -> P1 0x4C[3:0] */
    PAN211_WriteRegBits(0x4c, val[3] & 0x0f, BITMASK_4_0);
    /* FSYNVCO[4]*/
```

```
    if ((val[2] & BIT4) == 0)
    {
        PAN211_WriteReg(0x43, 0x11);
    }
    else
    {
        PAN211_WriteReg(0x43, 0x10);
    }
    PAN211_SetPage(0);
    /* VBG_TRIM_3V: 0x04[7:4]->P0 0x05[3:0] */
    /* LP_VREF_TRIM[3:0]: 0x03[7:4] -> P0 0x05[7:4] */
    PAN211_WriteReg(0x05, (val[3] & 0xF0) | (val[4] >> 4) & 0x0F);
    return PAN211_OK;
}
```

该函数定义于 pan211.c 文件中，由 PAN211_Err_t PAN211_Init(void)调用生效。

3.2 增加 PAN211_Err_t PAN211_FactoryCalibration(void)函数

从 MPA(SDK_V1.0.0 或 SDK_V1.1.0)升级到 MPB(SDK_V2.1.x)需要增加 PAN211_Err_t PAN211_FactoryCalibration(void)函数。该函数需要在 pan211.c 文件中的 PAN211_Err_t PAN211_Init(void)前声明，且在 PAN211_Err_t PAN211_Init(void)中写完配置参数（gPN211Page0Table 和 gPN211Page1Table）后调用。

具体调用位置如下所示：

```
    PAN211_SetPage(0);
    /* Write preconfigured registers on Page 0 */
    for (int i = 0; i < (sizeof(gPAN211Page0Table) / sizeof(gPAN211Page0Table[0])); i++)
    {
        PAN211_WriteReg(gPAN211Page0Table[i][0], gPAN211Page0Table[i][1]);
    }

    PAN211_FactoryCalibration();//调用工厂校准函数

#if EASY_RF
    PAN211_Calibration();
```

3.3 增加 uint8_t Factory_Read(uint8_t addr)函数

为使 PAN211_Err_t PAN211_FactoryCalibration(void)正常使用，需要增加 uint8_t Factory_Read(uint8_t addr)函数，如下所示：

```
uint8_t Factory_Read(uint8_t addr)
```

```
{
    uint8_t val;
    PAN211_SetPage(1);
    PAN211_WriteRegUnchecked(0x04, addr << 1);
    val = PAN211_ReadReg(0x04);
    PAN211_SetPage(0);
    Pan211_Funs.delayus(100);
    PAN211_WriteRegUnchecked(0x73, 0x01);
    return val;
}
```

该函数定义于 pan211.c 文件中，由 PAN211_Err_t PAN211_FactoryCalibration(void)调用生效。

4 初始化函数更改

4.1 增加 cons_reg_en 使能配置

从 MPA(SDK_V1.0.0 或 SDK_V1.1.0)升级到 MPB(SDK_V2.1.x)需要在 PAN211_Err_t PAN211_Init(void)函数增加 cons_reg_en 配置。

```
    PAN211_WriteReg(0x05, 0x00);    // Reset cons_reg_en
```

具体位置如下所示：

```
    /* Wait for FSYNXO_CLKRDY, then turn off FSYNXO_STARTUP_FAST */
    while ((PAN211_ReadReg(0x6f) & BIT7) != BIT7)
        ;
    PAN211_WriteReg(0x4c, 0x68);

    /* Reset cons_reg_en, recover it later in PAN211_FactoryCalibration() */
    PAN211_WriteReg(0x05, 0x00);

    /* Enter STB3 Mode */
    PAN211_WriteReg(0x02, 0x74);
```

4.2 增加晶振频率配置

从 MPA(SDK_V1.0.0 或 SDK_V1.1.0 或 SDK_V1.2.0)升级到 MPB(SDK_V2.1.x)需要增加晶振频率配置，如果使用 32M 晶振，则该配置可以忽视；16M 晶振需要增加该配置。
晶振频率配置代码如下所示：

```
#if (XTAL_FREQ == XTAL_FREQ_16M)
    /* Set 16M crystal */
    PAN211_WriteReg(0x37, 0xE0);
    PAN211_SetPage(1);
    PAN211_WriteReg(0x41, 0xA6); // bit2 -> 1
#else
    /* Set 32M crystal by default */
    PAN211_SetPage(1);
    PAN211_WriteReg(0x3f, 0xD2);
    PAN211_WriteReg(0x40, 0x20);
    PAN211_WriteReg(0x41, 0xA2);
#endif
```

上述代码需在 PAN211_Err_t PAN211_Init(void)中 LDO 稳定后配置。具体位置如下所示：

```
    /* Wait for LDO to be stable*/
    Pan211_Funs.delayus(200);

#if (XTAL_FREQ == XTAL_FREQ_16M)
    /* Set 16M crystal */
    PAN211_WriteReg(0x37, 0xE0);
    PAN211_SetPage(1);
    PAN211_WriteReg(0x41, 0xA6); // bit2 -> 1
#else
    /* Set 32M crystal by default */
    PAN211_SetPage(1);
    PAN211_WriteReg(0x3f, 0xD2);
    PAN211_WriteReg(0x40, 0x20);
    PAN211_WriteReg(0x41, 0xA2);
#endif

    /* Enter STB2 Mode */
     PAN211_WriteReg(0x02, 0x73);
```

4.3 增加相关宏定义配置

为使上述代码正确运行，需要在 pan211_port.h 中增加如下所示的宏定义。

```
#define XTAL_FREQ_16M        1
#define XTAL_FREQ_32M        2
#define XTAL_FREQ            XTAL_FREQ_32M
```