Panchip Microelectronics Co., Ltd.

PAN3028 SDK 用户指南

当前版本: 2.2

发布日期: 2022.06

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 302 室

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

磐启是磐启微电子有限公司的商标。本文档中提及的其他名称是其各自所有者的商标/注册商标。

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	2021.01.18	初始版本创建
V1.1	2021.03.02	更新公司信息
V1.2	2021.03.25	添加部分接口函数
V1.3	2021.06.03	修改 AGC 描述
V1.4	2021.06.18	增加 DCDC 描述
V1.5	2021.07.15	底板更新为华大芯片 HC32F460
V1.6	2021.08.25	修改部分接口描述
V1.7	2021.09.14	修改 3028 相关描述
V1.8	2021.11.08	增加智能搜索相关接口描述
V1.9	2022.03.29	修改发射功率函数及功率表
V2.0	2022.05.18	应用说明增加 RF 参数说明
V2.1	2022.05.20	修改接口描述和 SDK 示例
V2.2	2022.06.24	增加 3.17，7.6 章节，修改 7.2 描述

目录

1 概述

本文档主要介绍 PAN3028 SDK 接口函数的使用。包括芯片的初始化过程以及数据的收发过程。

SDK 相关的文件有《pan3028.c》、《pan3028.h》、《pan3028_port.c》、《pan3028_port.h》、《radio.c》、《radio.h》。

2 SDK使用流程

本章节详细描述 SDK 使用的基本流程，包括初始化，参数配置和数据收发。

2.1 初始化流程

2.1.1 接口函数

```
rf_init();
```

2.1.2 实现功能

1、唤醒 PAN3028 芯片，从 deep sleep 状态至 standby3 状态

```
PAN3028_deepsleep_wakeup
```

2、初始化芯片

```
PAN3028_init
```

3、配置芯片 AGC

```
rf_set_agc
```

4、初始化芯片 TX_IO/RX_IO

```
rf_port.antenna_init
```

2.1.3 使用方法

芯片初次上电时调用接口函数。

2.2 参数配置流程

2.2.1 接口函数

```
rf_set_default_para();
```

2.2.2 实现功能

1、设置频率、Code_Rate、带宽、扩频因子、发射功率和 CRC

```
PAN3028_set_freq
PAN3028_set_code_rate
PAN3028_set_bw
PAN3028_set_sf
PAN3028_set_tx_power
PAN3028_set_crc
PAN3028_set_ldr
```

2、刷新参数

```
PAN3028_rst
```

2.2.3 使用方法

芯片初始化完成后调用接口函数。

2.3 发送流程

2.3.1 接口函数

```
rf_enter_continous_tx();
rf_continous_tx_send_data();
rf_single_tx_data();
```

2.3.2 实现功能

1、切换状态

```
PAN3028_set_mode
```

2、打开 TX_IO

```
rf_antenna_tx
```

3、设置发射模式

```
PAN3028_set_tx_mode
```

4、进入 TX 状态并发送数据

```
PAN3028_send_packet
```

5、获取本包数据发射时间（可选）

```
rf_get_tx_time
```

2.3.3 使用方法

芯片初始化并且配置参数完成后调用接口函数。

调用接口函数，芯片发送完成后，IRQ 的引脚会被置高，并产生中断。

完整应用流程参照第五章节 SDK 示例。

2.4 接收流程

2.4.1 接口函数

```
rf_enter_continous_rx();
rf_enter_single_timeout_rx();
rf_enter_single_rx();
```

2.4.2 实现功能

1、切换状态

```
PAN3028_set_mode
```

2、打开 RX_IO

```
rf_antenna_rx
```

3、设置接收模式

```
PAN3028_set_rx_mode
```

4、设置进入接收状态

```
PAN3028_set_mode
```

2.4.3 使用方法

芯片初始化并且配置参数完成后调用接口函数。

接收数据，必须是在数据被收到的情况下才能进行。数据收到后，IRQ 引脚会被置高。可使用轮询或中断的方式来确定数据包是否被接收到，接收到后调用收包函数将数据包从芯片中取出。

完整应用流程参照第五章节 SDK 示例。

3 SDK接口描述

本章节详细描述 SDK《radio.c》中提供的用户接口函数，用户可以根据需要自行修改 radio 下的相关函数接口（如要修改建议在充分熟悉芯片后进行）。

大多参数配置都需要在芯片初始化完成后进行（standby3 状态下）。

3.1 rf_flag 接口

句法

```
uint32_t rf_get_recv_flag(void)
void rf_set_recv_flag(int status)
uint32_t rf_get_transmit_flag(void)
void rf_set_transmit_flag(int status)
```

目的

芯片设置发送或接收后，产生的 IRQ 中断，通过 flag 标志将事件传递给用户。用户可使用轮询方式来确定芯片的发送或接收结果。

参数

```
Status    当前的事件状态
RADIO_FLAG_IDLE
RADIO_FLAG_TXDONE
RADIO_FLAG_RXDONE
RADIO_FLAG_RXTIMEOUT
RADIO_FLAG_RXERR
RADIO_FLAG_PLHDRXDONE
```

3.2 rf_init

句法

```
uint32_t rf_init(void)
```

目的

初始化芯片，包含 wakeup，init，agc，antenna_init 过程。

参数

无

返回值

FAIL　操作失败

OK　　操作成功

3.3 rf_deepsleep_wakeup

句法

```
uint32_t rf_deepsleep_wakeup(void)
```

目的

设置芯片从 deepsleep 状态唤醒，包含 wakeup，init，agc，antenna_init 过程。

当芯片处于 deepsleep 状态，需要唤醒芯片时使用。当芯片处于 deepsleep 状态时，芯片的寄存器配置不会保存，因此唤醒后需要重新配置初始化和 agc 参数，其他收发配置参数（SF，BW 等）也需要重新配置。

参数

无

返回值

FAIL　操作失败

OK　　操作成功

3.4 rf_sleep_wakeup

句法

```
uint32_t rf_sleep_wakeup(void)
```

目的

设置芯片从 sleep 状态唤醒，包含 wakeup，antenna_init 过程。

当芯片处于 sleep 状态，需要唤醒芯片时使用。当芯片处于 sleep 状态时，芯片的寄存器配置会保存。

参数

无

返回值

FAIL　操作失败

OK　　操作成功

3.5 rf_deepsleep

句法

```
uint32_t rf_deepsleep(void)
```

目的

将芯片从 standby3 状态切换到 deep sleep 状态，从而降低功耗。

参数

无

返回值

FAIL　操作失败

OK　　操作成功

3.6 rf_sleep

句法

```
uint32_t rf_sleep(void)
```

目的

将芯片从 standby3 状态切换到 sleep 状态，从而降低功耗。

参数

无

返回值

FAIL　操作失败

OK　　操作成功

3.7 rf_get_tx_time

句法

```
uint32_t rf_get_tx_time(void)
```

目的

获取本次数据包发射所需时间，rf_single_tx_data()内提供该函数用法演示。

参数

无

返回值

本次数据包发射所需时间，单位 ms。

3.8 rf_set_mode

句法

```
uint32_t rf_set_mode(uint8_t mode)
```

目的

设置芯片工作的状态，包括 deepsleep/sleep/standby1/standby2/standby3/tx/rx，参数配置通常在 standby3 状态下进行。

参数

Mode	芯片工作的状态
PAN3028_MODE_DEEP_SLEEP	
PAN3028_MODE_SLEEP	
PAN3028_MODE_STB1	
PAN3028_MODE_STB2	
PAN3028_MODE_STB3	
PAN3028_MODE_TX	
PAN3028_MODE_RX	

返回值

FAIL 操作失败

OK 操作成功

3.9 rf_get_mode

句法

```
uint8_t rf_get_mode(void)
```

目的

读取芯片当前的工作状态，包括 deepsleep/sleep/standby1/standby2/standby3/tx/rx。

参数

无

返回值

芯片状态值

3.10 rf_set_tx_mode

句法

```
uint32_t rf_set_tx_mode(uint8_t mode)
```

目的

设置发射模式，可以设置为单次发射和连续发射。

参数

Mode	发射数据模式
PAN3028_TX_SINGLE	
PAN3028_TX_CONTINOUS	

返回值

FAIL 操作失败

OK 操作成功

3.11 rf_set_rx_mode

句法

```
uint32_t rf_set_rx_mode(uint8_t mode)
```

目的

设置接收模式，可以设置为单次接收、单次超时接收、连续接收。

参数

Mode	接收模式

```
PAN3028_RX_SINGLE
PAN3028_RX_SINGLE_TIMEOUT
PAN3028_RX_CONTINOUS
```

返回值

FAIL　操作失败

OK　　操作成功

3.12　rf_set_rx_single_timeout

句法

```
uint32_t rf_set_rx_single_timeout(uint32_t timeout)
```

目的

设置单次超时接收的超时时间，仅在单次超时接收模式下有效。

参数

```
Timeout　1~65535　超时时间，单位为 ms
```

返回值

FAIL　操作失败

OK　　操作成功

3.13　rf_get_snr

句法

```
float rf_get_snr(void)
```

目的

读取信噪比 SNR 的值，这个接口需要在接收到数据包的时候调用，且在清除 RxDone 中断之前。如果清除中断，这个值就会失效。RSSI 也一样。PAN3028_irq_handler()函数内提供了该函数的用法演示。

示例：

```
// RxDone IRQ 已收到
snr = rf_get_snr();
rssi = rf_get_rssi();
rf_receive(buf);
```

参数

无

返回值

SNR 值

3.14 rf_get_rssi

句法

```
float rf_get_rssi(void)
```

目的

读取接收信号强度的值，这个接口需要在接收到数据包的时候调用，且在清除 RxDone 中断之前。如果清除中断，这个值就会失效。SNR 也一样。PAN3028_irq_handler()函数内提供了该函数的用法演示。

示例：

```
// RxDone IRQ 已收到
snr = rf_get_snr();
rssi = rf_get_rssi();
rf_receive(buf);
```

参数

无

返回值

RSSI 值

3.15 rf_set_preamble

句法

```
uint32_t rf_set_preamble(uint16_t pream)
```

目的

设置芯片发射前导码，需要在发射数据前配置。

参数

Pream	1~65535	发射前导码长度

返回值

FAIL 操作失败

OK 操作成功

3.16 rf_set_cad

句法

```
uint32_t rf_set_cad(void)
```

目的

设置芯片打开 CAD 功能，需要在芯片进入接收模式前配置，该功能修改了接收阈值。

参数

无

返回值

FAIL 操作失败

OK 操作成功

3.17 rf_set_cad_off

句法

```
uint32_t rf_set_cad_off(void)
```

目的

设置芯片关闭 CAD 功能，并将接收阈值恢复。

参数

无

返回值

FAIL 操作失败

OK 操作成功

3.18 rf_set_syncword

句法

```
uint32_t rf_set_syncword(uint8_t sync)
```

目的

设置芯片的同步字。

参数

Sync 同步字

返回值

FAIL 操作失败

OK 操作成功

3.19 rf_get_syncword

句法

```
uint8_t rf_get_syncword(void)
```

目的

读取芯片的同步字。

参数

无

返回值

同步字值

3.20 rf_irq_handler

句法

```
void rf_irq_handler(void)
```

目的

IRQ 中断服务程序，在外部中断产生时调用。

参数

无

返回值

无

3.21 rf 提前中断接口

句法

```
void rf_set_plhd_rx_on(uint8_t addr,uint8_t len)
```

```
void rf_set_plhd_rx_off(void)
uint32_t rf_plhd_receive(uint8_t *buf,uint8_t len)
```

目的

提前中断功能是在芯片读取一帧数据的过程中，查看已经解出来的数据，判断是不是自己想要的，再决定继续读取还是放弃这帧数据。更多功能说明请参考《PAN3028 提前中断应用参考文档》。

3.22 rf_receive

句法

```
uint32_t rf_receive(uint8_t *buf)
```

目的

接收一包数据

参数

```
Buf  用于接收数据包
```

返回值

接收到的数据包长度

3.23 rf_enter_continous_rx

句法

```
uint32_t rf_enter_continous_rx(void)
```

目的

设置芯片进入连续接收模式接收。

该模式下，芯片会一直处于接收状态，当接收到数据后，芯片会产生 rxdone IRQ 中断，并继续进行接收。

参数

无

返回值

FAIL　操作失败

OK　　操作成功

3.24　rf_enter_single_timeout_rx

句法

```
uint32_t rf_enter_single_timeout_rx(uint32_t timeout)
```

目的

设置芯片进入单次超时接收模式接收。

该模式下，芯片会进行一次接收，当接收到数据后，芯片会产生 rxdone IRQ 中断，并等待用户后续操作。当超时时间内都未接收到数据时，芯片会产生 rxtimeout IRQ 中断，并停止接收，等待用户后续操作。

参数

```
Timeout　超时时间
```

返回值

FAIL　操作失败

OK　　操作成功

3.25　rf_enter_single_rx

句法

```
uint32_t rf_enter_single_rx(void)
```

目的

设置芯片进入单次接收模式接收。

该模式下，芯片会进行一次接收，当接收到数据后，芯片会产生 rxdone IRQ 中断，并等待用户后续操作。

参数

无

返回值

FAIL　操作失败

OK　　操作成功

3.26　rf_single_tx_data

句法

```
uint32_t rf_single_tx_data(uint8_t *buf, uint8_t size, uint32_t *tx_time)
```

目的

设置芯片进入单次发射模式并发射数据。

该模式下，芯片会进行一次发射，当发射完成后，芯片会产生 txdone IRQ 中断，并退出发射状态（工作电流降低），等待用户后续操作。

参数

Buf　发送 buff

Size　发送包长

Tx_time 获取本次数据包的传输时间，tx_time 参数非必须，用户可根据需要删减

返回值

FAIL　操作失败

OK　　操作成功

3.27　rf_enter_continous_tx/rf_continous_tx_send_data

句法

```
uint32_t rf_enter_continous_tx(void)
uint32_t rf_continous_tx_send_data(uint8_t *buf, uint8_t size)
```

目的

设置芯片进入连续发射模式并发射数据。

该模式下，芯片一次发射完成后，芯片会产生 txdone IRQ 中断，并保持发射状态（工作电流仍为发射电流，但无数据发射），并等待用户后续操作。

随后，用户可以选择停止发射，将芯片状态切换至 standby3；或者选择继续发射，直接调用 rf_continous_tx_send_data()即可。不需要重复 rf_enter_continous_tx()操作。

参数

无

返回值

FAIL　操作失败

OK　　操作成功

3.28　rf_set_agc

句法

```
uint32_t rf_set_agc(uint32_t state)
```

目的

配置芯片 AGC。默认初始化芯片时配置并打开 AGC。

参数

State　关闭 AGC_OFF/ 打开 AGC_ON

返回值

FAIL　操作失败

OK　　操作成功

3.29 rf_set_para

句法

```
uint32_t rf_set_para(rf_para_type_t para_type, uint32_t para_val)
```

目的

配置芯片参数，包括频率、Code_Rate、带宽、扩频因子、发射功率和 CRC。

参数

Para_type 准备设置参数的类型

RF_PARA_TYPE_FREQ

RF_PARA_TYPE_CR

RF_PARA_TYPE_BW

RF_PARA_TYPE_SF

RF_PARA_TYPE_TXPOWER

RF_PARA_TYPE_CRC

Para_val 准备设置参数的值

CODE_RATE_45/CODE_RATE_46/CODE_RATE_47/CODE_RATE_48

BW_62_5K/BW_125K/BW_250K/BW_500K

SF_7/SF_8/SF_9/SF_10/SF_11/SF_12

TXPOWER 参照附录

CRC_OFF/CRC_ON

返回值

FAIL 操作失败

OK 操作成功

3.30 rf_get_para

句法

```
uint32_t rf_get_para(rf_para_type_t para_type, uint32_t *para_val)
```

目的

读取芯片参数，包括频率、Code_Rate、带宽、扩频因子、发射功率和 CRC。

参数

```
Para_type   准备读取参数的类型
    RF_PARA_TYPE_FREQ
    RF_PARA_TYPE_CR
    RF_PARA_TYPE_BW
    RF_PARA_TYPE_SF
    RF_PARA_TYPE_TXPOWER
    RF_PARA_TYPE_CRC
Para_val    读取参数的值
```

返回值

FAIL　操作失败

OK　　操作成功

3.31　rf_set_dcdc_mode

句法

```
uint32_t rf_set_dcdc_mode(uint32_t dcdc_val)
```

目的

设置芯片 DCDC 模式，默认关闭。

开启本功能前，请联系并确认模组是否支持该功能开启，否则易对模组造成损害。

参数

```
Dcdc_val    模式类型
    DCDC_ON
    DCDC_OFF
```

返回值

FAIL　操作失败

OK　　操作成功

3.32　rf_set_ldr

句法

```
uint32_t rf_set_ldr(uint32_t mode)
```

目的

设置芯片 LDR 模式，默认关闭。

参数

Mode　模式类型

LDR_ON

LDR_OFF

返回值

FAIL　操作失败

OK　　操作成功

3.33　rf 智能搜索接口

句法

```
uint32_t rf_set_all_sf_preamble(uint32_t sf)
        sf 参数配置：SF_7/SF_8/SF_9/SF_10/SF_11/SF_12
uint32_t rf_set_all_sf_search(void)
uint32_t rf_set_all_sf_search_off(void)
```

目的

智能搜索功能，可实现在接收时智能化识别信道中的 SF 参数，达到接收不同 SF 信号数据的目的。更多功能说明请参考《PAN3028 智能搜索应用参考文档》。

4 SDK移植

本章节详细描述 SDK 移植过程及注意事项。

SDK《pan3028_port.c》中提供了用户需要移植的接口函数。

要正确运行 SDK，需要实现以下接口：

- spi_cs_set_high()，SPI 片选拉高。
- spi_cs_set_low()，SPI 片选拉低。
- spi_readwritebyte()，SPI 数据传输。
- rf_delay_ms()，rf_delay_us()，delay 函数。
- RX_IO/TX_IO 实现，包括 rf_antenna_init，rf_antenna_tx，rf_antenna_rx，rf_antenna_close。

RX_IO/TX_IO 实现，SDK 中通过 SPI 读写寄存器实现该部分，通过 SPI 控制 3028 内部 GPIO1 和 GPIO3 实现 RX_IO/TX_IO。

用户也可以使用外部 MCU IO 去控制 RX_IO/TX_IO，具体实现取决于底板设计情况。

- TCXO_IO 实现，包括 rf_tcxo_init，rf_tcxo_close。

TCXO_IO 实现，SDK 中通过 SPI 读写寄存器实现该部分，通过 SPI 控制 3028 内部 GPIO5 实现 TCXO_IO。

用户也可以使用外部 MCU IO 去控制 TCXO_IO，具体实现取决于 PAN3028 模组版本。

- rf_irq_handler()，IRQ 中断处理函数。

PAN3028 的 IRQ 引脚用于产生中断给 MCU，连接 IRQ 的 MCU 引脚需要配置成外部中断触发模式，它的中断服务程序中要调用 PAN3028 的中断处理函数。

```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    rf_irq_handler();
}
```

5 SDK示例

本章节详细描述 SDK 示例的实现过程。

5.1 Tx Demo

5.1.1 代码流程图

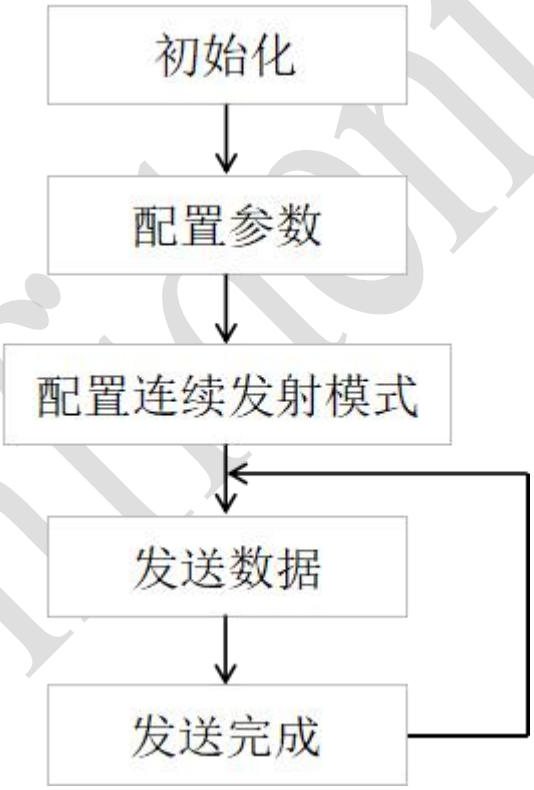

图 5-1 Tx 代码流程图

5.1.2 代码实现

```
ret = rf_init();                                                    //初始化
if(ret != OK)
{
```

```
        DDL_Printf("    RF Init Fail");
        while(1);
    }
    rf_set_default_para();                                          //配置参数
    rf_enter_continous_tx();                                        //配置发射模式
    if(rf_continous_tx_send_data(tx_test_buf, TX_LEN) != OK)        //发送数据
    {
        DDL_Printf("tx fail \r\n");
    }
    else
    {
        cnt ++;
        DDL_Printf("Tx cnt %d\r\n", cnt );
    }
    while (1)
    {
        rf_irq_process();                                           //轮询中断标志
        if(rf_get_transmit_flag() == RADIO_FLAG_TXDONE)             //检查中断事件
        {
            BSP_LED_Toggle();
            rf_set_transmit_flag(RADIO_FLAG_IDLE);                  //清除中断事件
            SysTick_Delay(1000);
            if(rf_continous_tx_send_data(tx_test_buf, TX_LEN) != OK)    //再次发送
            {
                DDL_Printf("tx fail \r\n");
            }
            else
            {
                cnt ++;
```

```
                DDL_Printf("Tx cnt %d\r\n", cnt );
            }
        }
    }
```

5.1.3 注意事项

当芯片配置进入连续发射模式后，就可以连续进行数据发射。

连续发射模式中，发射完成后，如果不退出发射状态（切换至 standby3 模式可退出发射状态），那么芯片的工作电流会一直保持为发射电流。

5.2 Rx Demo

5.2.1 代码流程图

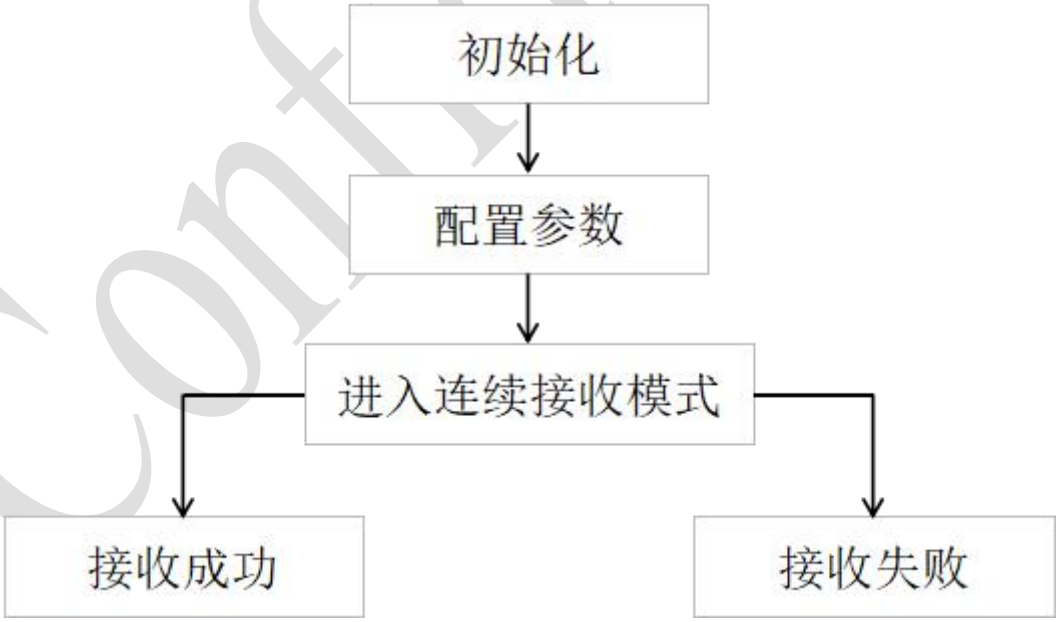

图 5-2 Rx 代码流程图

5.2.2 代码实现

```
    ret = rf_init();                                                    //初始化
```

```
        if(ret != OK)
        {
            DDL_Printf("    RF Init Fail");
            while(1);
        }

        rf_set_default_para();                                  //配置参数
        rf_enter_continous_rx();                                //进入连续接收模式
        while (1)
        {
            rf_irq_process();                                   //轮询中断标志
            if(rf_get_recv_flag() == RADIO_FLAG_RXDONE)         //接收成功
            {
                BSP_LED_Toggle();
                rf_set_recv_flag(RADIO_FLAG_IDLE);
                 DDL_Printf("Rx  :  SNR:  %f  ,RSSI:  %f  \r\n",  RxDoneParams.Snr,
RxDoneParams.Rssi);
                 for(i = 0; i < RxDoneParams.Size; i++)
                 {
                     DDL_Printf("0x%02x ", RxDoneParams.Payload[i]);
                 }
                 DDL_Printf("\r\n");
                 cnt ++;
                 DDL_Printf("###Rx cnt %d##\r\n", cnt);
            }
            if((rf_get_recv_flag()  ==  RADIO_FLAG_RXTIMEOUT)  ||  (rf_get_recv_flag()  ==
RADIO_FLAG_RXERR))                                              //接收失败
            {
                rf_set_recv_flag(RADIO_FLAG_IDLE);
                DDL_Printf("Rxerr\r\n");
```

```
        }
    }
```

5.2.3 注意事项

当芯片配置进入连续接收模式后，就可以连续进行数据接收。不论接收成功还是接收失败，芯片都会在接收结束后继续保持接收状态，除非用户主动退出该状态（切换至 standby3 模式可退出接收状态）。

5.3 Tx-rx Demo

5.3.1 代码流程图

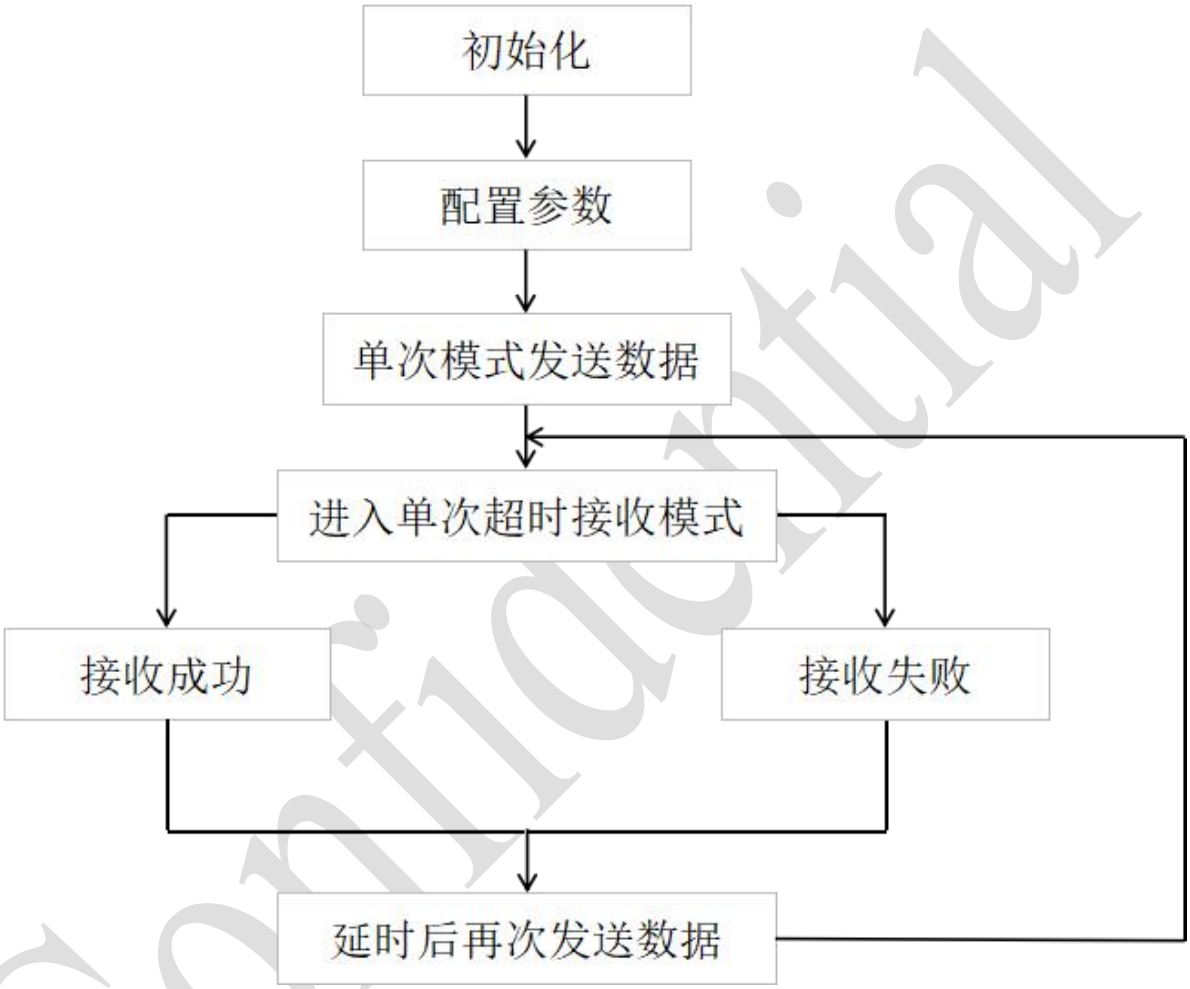

图 5- 3 Tx-Rx 代码流程图

5.3.2 代码实现

```
ret = rf_init();                                                    //初始化
if(ret != OK)
{
    DDL_Printf("    RF Init Fail");
    while(1);
```

```
        }
        rf_set_default_para();                                  //配置参数
        if(rf_single_tx_data(tx_test_buf, TX_LEN, &tx_time) != OK)          //发送数据
        {
            DDL_Printf("tx fail \r\n");
        }
        while (1)
        {
            rf_irq_process();                                   //轮询中断标志
            if(rf_get_recv_flag() == RADIO_FLAG_RXDONE)         //接收成功
            {
                BSP_LED_Toggle();
                rf_set_recv_flag(RADIO_FLAG_IDLE);
                DDL_Printf("Rx : SNR: %f ,RSSI: %f \r\n", RxDoneParams.Snr, RxDoneParams.Rssi);
                for(i = 0; i < RxDoneParams.Size; i++)
                {
                    DDL_Printf("0x%02x ", RxDoneParams.Payload[i]);
                }
                DDL_Printf("\r\n");
                rxcnt ++;
                DDL_Printf("###Rx cnt %d##\r\n", rxcnt);
                rf_sleep();
                rf_sleep_wakeup();
                SysTick_Delay(3000);
                if(rf_single_tx_data(tx_test_buf, TX_LEN, &tx_time) != OK)  //成功后再次发送数据
                {
                    DDL_Printf("tx fail \r\n");
```

```
                }
            }
            if((rf_get_recv_flag() == RADIO_FLAG_RXTIMEOUT) || (rf_get_recv_flag() ==
RADIO_FLAG_RXERR))                                          //接收失败
            {
                rf_set_recv_flag(RADIO_FLAG_IDLE);
                DDL_Printf("Rxerr\r\n");
                rf_sleep();
                rf_sleep_wakeup();
                HAL_Delay(10000);
                if(rf_single_tx_data(tx_test_buf, TX_LEN, &tx_time) != OK)    //失败后再次发
送数据
                {
                    DDL_Printf("tx fail \r\n");
                }
            }
            if(rf_get_transmit_flag() == RADIO_FLAG_TXDONE)                 //发送成功
            {
                rf_set_transmit_flag(RADIO_FLAG_IDLE);
                txcnt ++;
                DDL_Printf("Tx cnt %d\r\n", txcnt );
                //txdone, set sleep and wakeup
                rf_sleep();
                rf_sleep_wakeup();
                //single_timeout_rx
                rf_enter_single_timeout_rx(15000)                           ;//进入接收
            }
        }
```

5.3.3 注意事项

将两个模组先后上电，即可实现两个模组互相收发的功能。

模组初次上电发送后，会进入单次超时接收模式。如果超时时间内未收到数据，则延时后，重新发送一包数据；如果收到了数据，则延时后，再次发送一包数据。

注意，PAN3028 芯片在单次模式下（包含单次发射模式，单次超时接收模式和单次接收模式），发送或接收完成后需要切换至 sleep 模式。如果需要再次进行收发，需要将芯片从 sleep 模式再次唤醒，方可继续使用。

6 PAN3028状态图

本章节描述的是 PAN3028 芯片的状态切换图，方便用户深入了解芯片工作原理。其中，DeepSleep 到 STB3(standby3)的切换过程对应 SDK 中的 rf_deepsleep_wakeup 函数；STB3 到 DeepSleep 对应 SDK 中的 rf_deepsleep 函数。Sleep 到 STB3 的切换过程对应 SDK 中的 rf_sleep_wakeup 函数；STB3 到 Sleep 对应 SDK 中的 rf_sleep 函数。

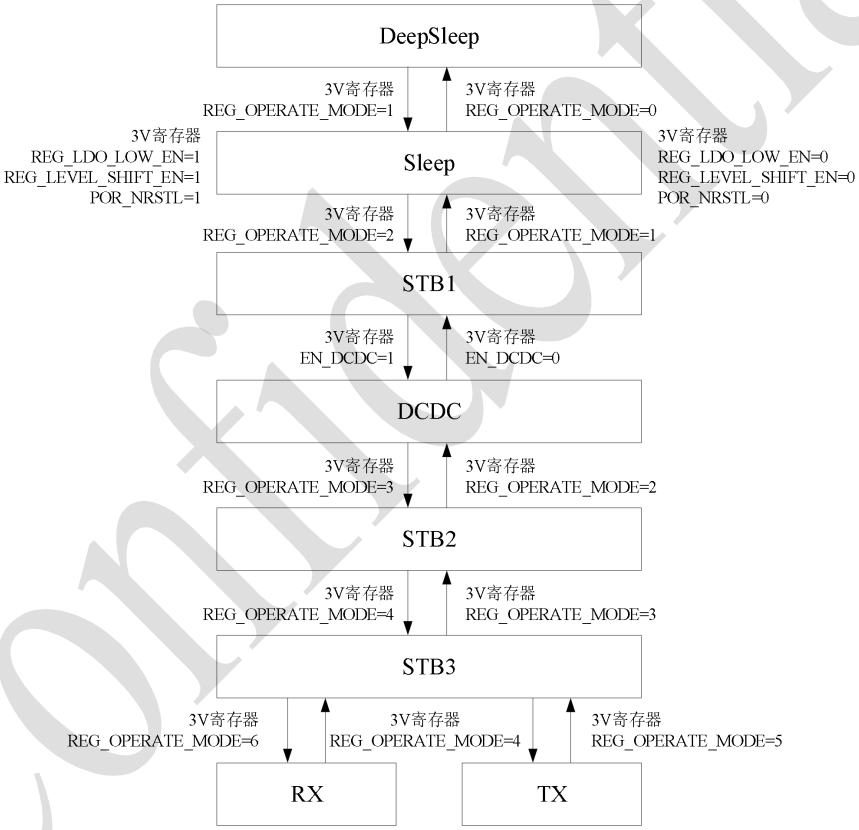

图 6-1 PAN3028 上电流程

PAN3028 上电后有六种状态，包括深度睡眠模式(DeepSleep)、睡眠模式(Sleep)、OSC 工作模式(STB1)、LDO 工作模式(STB2)、OSC 输出模式(STB3)、Tx 模式以及 Rx 模式。所有状态可通过配置寄存器进行切换。在 STB1 状态下可通过给 EN_DCDC 寄存器写入高电平来打开芯片的 DCDC 功能。

1. 上电后芯片处于 DeepSleep 模式，首先将寄存器切换至第三页，给地址 0x00 的 reg_page_sel 写入 0x03。在此状态下可以对诸如 SF,BW 等参数进行相应的配置。配置完成后延时再进行后续操作。
2. 将芯片切换至 Sleep 模式，给地址 0x02 的 reg_operate_mode 写入 0x01。在此模式下打开 1.5V 逻辑的复位，low LDO 以及 3V 电压域至 1.5V 电压域的 level shift，在地址 0x04 写入 0x76。另外，在此模式之后，都可以对 1.5V 寄存器进行读写，如果有需求，可以在此处修改相应的寄存器。配置完成后延时再进行后续操作。
3. 将芯片切换至 STB1 模式，给地址 0x02 的 reg_operate_mode 写入 0x02。此模式下打开 DCDC 功能，在地址 0x1e 写入 0x6d。配置完成后延时再进行后续操作。
4. 切换模式至 STB2 模式，给地址 0x02 的 reg_operate_mode 写入 0x03。配置完成后延时再进行后续操作。
5. 切换模式至 STB3 模式，给地址 0x02 的 reg_operate_mode 写入 0x04。配置完成后延时再进行后续操作。
6. 根据需求选择 TX 或者 RX 模式，给地址 0x02 的 reg_operate_mode 写入 0x05(TX)或者 0x06(RX)。
7. 切换模式至 STB3 模式，给第三页地址 0x02 的 reg_operate_mode 写入 0x04。配置完成后延时再进行后续操作。
8. 切换模式至 STB2 模式，给第三页地址 0x02 的 reg_operate_mode 写入 0x03。配置完成后延时再进行后续操作。
9. 切换模式至 STB1 模式，给第三页地址 0x02 的 reg_operate_mode 写入 0x02。关闭 DCDC 功能，在地址 0x1e 写入 0x6c。配置完成后延时再进行后续操作。
10. 切换模式至 Sleep 模式，给第三页地址 0x02 的 reg_operate_mode 写入 0x01。配置完成后延时再进行后续操作。
11. 切换模式至 DeepSleep 模式，给第三页地址 0x02 的 reg_operate_mode 写入 0x00。

7 应用说明

7.1 低功耗休眠

关于低功耗休眠，SDK 中提供了四个接口函数。

一种是睡眠 Sleep 模式。包含从 STB3 到 Sleep 的 rf_sleep 休眠函数，从 Sleep 到 STB3 的 rf_sleep_wakeup 唤醒函数。

一种是深度休眠 DeepSleep 模式。包含从 STB3 到 DeepSleep 的 rf_deepsleep 休眠函数，从 DeepSleep 到 STB3 的 rf_deepsleep_wakeup 唤醒函数。注意，PAN3028 芯片进入深度休眠并再次唤醒后，原有的配置参数需要重新配置，收发模式也需要重新选择。

```
//deepsleep 休眠示例
rf_deepsleep();                          //进入深度休眠
rf_deepsleep_wakeup();                   //唤醒
rf_set_default_para();                   //重新配置参数
rf_enter_continous_rx();                 //进入连续接收模式
```

7.2 CAD 功能

CAD 功能，修改了芯片的接收阈值，可能影响芯片的接收灵敏度。

使用完成后，需要关闭 CAD 功能。

更多功能说明请参考《PAN3028 CAD 应用参考文档》。

7.3 灵敏度 RSSI

读取信号强度值需要在接收到数据包的时候调用，且在清除 rxdone 中断之前。如果清除中断，这个值就会失效。RSSI 的测量范围是-60 到-140，不同参数（SF、BW）模式下，测量范围略有不同。

当需要读取芯片 RSSI 时，建议打开 AGC，可以提高芯片 RSSI 准确性。更多功能说明请参考《PAN3028 RSSI 应用参考文档》。

7.4 DCDC 功能

DCDC 功能开启后，可以使芯片的接收电流降低至 12.5mA，默认该功能关闭。

开启 DCDC 功能前，请联系并确认模组是否支持该功能开启，否则易对模组造成损害。

开启 DCDC 功能后，当芯片需要休眠时，需要提前手动关闭 DCDC 功能。

7.5 RF 参数说明

- **信道带宽（BW）**

信道带宽是限定允许通过该信道的信号下限频率和上限频率，可以理解为一个频率通带，其取值范围为 62.5KHz、125KHz、250KHz、500KHz。比如一个信道允许的通带为 1.5kHz 至 15kHz，则其带宽为 13.5kHz。增加 BW，可以提高有效数据速率以缩短传输时间，但是以牺牲部分接受灵敏度为代价，从而影响通信距离。

- **扩频因子（SF）**

SF 取值范围 SF7-12，值越小传输速率越高，传输距离越短，相反 SF 值越大，传输速率越慢，传输距离越远。在同等数量的数据传输时，SF 越大传输时间越长。

- **编码率（CR）**

CR 是数据流中有用部分（非冗余）的比例，取值范围 1-4 对应 1=4/5，2=4/6，3=4/7，4=4/8。也就是说，如果编码率是 k/n，则对每 k 位有用信息，编码器总共产生 n 位的数据，其中 n-k 是多余的。CR 越大单次传输的多余数据越多，有效数据速率降低，传输时间变长。

- **低速率模式**

取值范围 1 或者 0，为 1 时为开启低速率模式，默认为关闭，开启后通信速率会降低，接收灵敏度会提升 1-2dBm，通信距离会稍增加。

- **前导码符号数量**

取值范围 8—65535，默认值为 8。在实现空中唤醒应用场景时可以将前导码符号数量增大，

同时发射时间也会增大，从而实现接收端在低功耗模式下被长的前导码唤醒的功能。

备注：上述 RF 参数需要根据实际应用场景进行选择配置，且收发端的 RF 配置需要一致时才能正常通信。RF 数据发送时间及参数的选择可参考 SDK 中工具目录下《磐启微电子 PAN3028 计算器.xlsx》

7.6 SPI 和 FIFO

外部设备可通过四线 SPI 方式对芯片中的寄存器和 FIFO 进行配置访问。

PAN3028 芯片实现了 SPI 总线的从机 Slave，用于读写寄存器和 FIFO。SPI 总线为四线制，分别为：

- SCK（时钟）
- CSN（片选信号，低电平有效）
- MOSI（数据输入）
- MISO（数据输出）

其中 SCK、CSN、MOSI 由主机 Master 控制，MISO 由 Slave 控制。

在通信过程中，以 CSN 电平拉低起始，直至 CSN 电平拉高时结束本次传输过程。主机 Master 通过 MOSI 发送数据，MISO 接收数据。SCK 下降沿时产生数据，上升沿时进行数据采样。

Master 传输的信息由 Address Byte 和 Data Byte 两部分组成。其中 Address Byte 前 7bit 为地址位 addr；最后 1bit 为读写位 wr，写操作时该 bit 置 1，读操作时该 bit 置 0。

SPI 有三种传输模式：

- Single：单字节传输模式。信息仅为 2 byte，Master 通过 MOSI 发送 Address Byte。若为写操作，Master 继续通过 MOSI 发送 Data Byte；若为读操作，则 Master 读取 MISO 上 Slave 回复的 Data Byte。
- Burst：突发连续传输模式。信息大于 2byte，Address Byte 后跟若干个 Data Byte，Data Byte 之间无需增加 Address Byte，从机 Slave 内部会自动在每个 Data Byte 之间递增地址。CSN 信号在最后一个 Data Byte 后拉高，其余传输信息过程均维持低电平。
- FIFO：FIFO 读写模式。该模式下单字节或连续传输均可实现，传输规则同 Signle 模式和 Burst 模式，不同点在于 Address Byte 中的地址位 addr 只能配置为 7'h1，且 Slave 在 Data Byte 之间不做地址递增操作。

SPI 写时序如下：

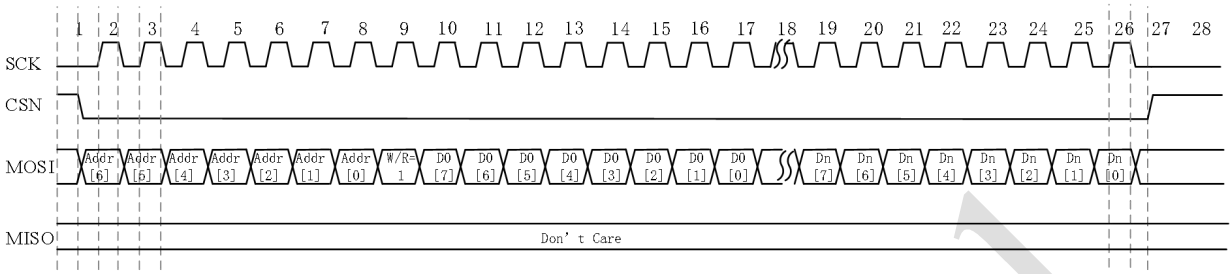

图 7-1 SPI 写时序

SPI 读时序如下：

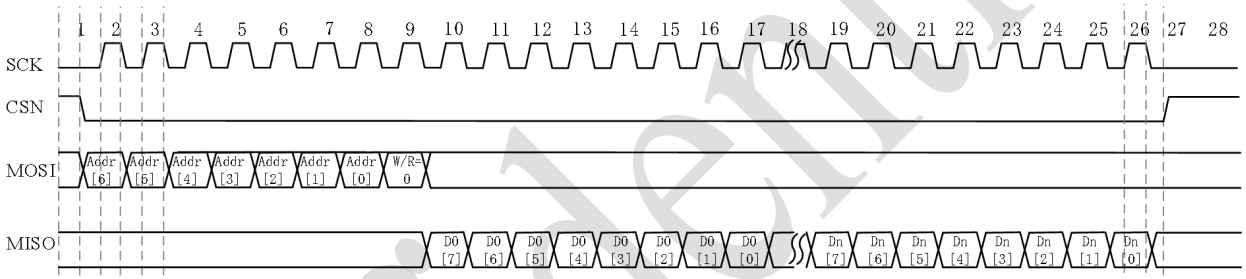

图 7-2 SPI 读时序

PAN3028 具有 256bytes 的 FIFO 用以存储 TX 模块发送数据和 RX 模块解码数据。

FIFO 由单口 RAM 组成，只能实现单包数据信息的存储和读取，在 FIFO 已存有一包数据的情况下，应先读取完此包数据后再写入，否则 FIFO 中前一包数据将被覆盖。

FIFO 在 STB3 及之后的工作模式中，可以由 Modem 和 SPI 完成读写操作。

8 附件

8.1 发射功率表（433MHz）

功率挡位	发射功率 (dbm)	发射电流 (mA)
0	-23.5	11
1	-18	13
2	-11	15
3	-8	16
4	-7	17
5	-6	18
6	-4	20
7	-2	23
8	0	25
9	0.6	29
10	1.1	30
11	4.6	34
12	7.1	38
13	9	42
14	10	46
15	11.7	50
16	12.7	53
17	13.7	57
18	14.4	60
19	15.1	64
20	16.1	69
21	17	74
22	17.6	79
23	18.6	87
24	19.7	100
25	22	135
26	22	136
27	22	137
28	22	138
29	22	139

注：功率挡位为 PAN3028_set_tx_power 函数的参数，即 rf_set_para 函数的功率参数。
其它频段的发射功率和发射电流会略有不同，其它频段 26-29 挡位的功率会有一定变化。