



Panchip Microelectronics Co., Ltd.

## PAN3029 SDK 用户指南

当前版本: 0.5

发布日期: 2023.12

## 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
V0.1	2023.06	初始版本创建
V0.2	2023.08	1.修改单包模式使用之后的操作流程 2.增加获取当前信道能量检测接口
V0.3	2023.09	1.更新发射功率表 2.更新 MAPM 接口
V0.4	2023.12	1.更新发射功率表 1-23 档位
V0.5	2023.12	1.更新 SDK 移植说明

## 目录

1 概述.....	1
2 SDK 使用流程.....	2
2.1 初始化流程.....	2
2.1.1 接口函数.....	2
2.1.2 实现功能.....	2
2.1.3 使用方法.....	2
2.2 参数配置流程.....	2
2.2.1 接口函数.....	2
2.2.2 实现功能.....	3
2.2.3 使用方法.....	3
2.3 发送流程.....	3
2.3.1 接口函数.....	3
2.3.2 实现功能.....	3
2.3.3 使用方法.....	4
2.4 接收流程.....	4
2.4.1 接口函数.....	4
2.4.2 实现功能.....	4
2.4.3 使用方法.....	5
3 SDK 接口描述.....	6
3.1 rf_flag 接口.....	6
3.2 rf_init.....	7
3.3 rf_deepsleep_wakeup.....	7
3.4 rf_sleep_wakeup.....	8
3.5 rf_deepsleep.....	8
3.6 rf_sleep.....	9
3.7 rf_get_tx_time.....	9
3.8 rf_set_mode.....	10
3.9 rf_get_mode.....	10
3.10 rf_set_tx_mode.....	11
3.11 rf_set_rx_mode.....	11
3.12 rf_set_rx_single_timeout.....	12
3.13 rf_get_snr.....	12
3.14 rf_get_rssi.....	13
3.15 rf_get_channel_rssi.....	14
3.16 rf_get_irq.....	14
3.17 rf_clr_irq.....	15
3.18 rf_set_refresh.....	15
3.19 rf_set_preamble.....	16
3.20 rf_set_cad.....	16

3.21 rf_set_cad_off.....	17
3.22 rf_set_syncword.....	17
3.23 rf_get_syncword.....	18
3.24 rf_irq_handler.....	18
3.25 rf 提前中断接口.....	18
3.26 rf_set_mapm_rx_on.....	19
3.27 rf_set_mapm_rx_off.....	19
3.28 rf_set_mapm_para.....	20
3.29 rf_receive.....	20
3.30 rf_enter_continuous_rx.....	21
3.31 rf_enter_single_timeout_rx.....	21
3.32 rf_enter_single_rx.....	22
3.33 rf_single_tx_data.....	22
3.34 rf_enter_continuous_tx/rf_continuous_tx_send_data.....	23
3.35 rf_set_agc.....	24
3.36 rf_set_para.....	24
3.37 rf_get_para.....	25
3.38 rf_set_dcdc_mode.....	26
3.39 rf_set_ldr.....	26
3.40 rf 智能搜索接口.....	27
3.41 get_chirp_time.....	27
3.42 check_cad_rx_inactive.....	28
3.43 check_cad_tx_inactive.....	28
4 SDK 移植.....	30
5 SDK 示例.....	31
5.1 Tx Demo.....	31
5.1.1 代码流程图.....	31
5.1.2 代码实现.....	31
5.1.3 注意事项.....	33
5.2 Rx Demo.....	33
5.2.1 代码流程图.....	33
5.2.2 代码实现.....	33
5.2.3 注意事项.....	35
5.3 Frequency-Hopping/Tx Demo.....	35
5.3.1 代码流程图.....	35
5.3.2 注意事项.....	35
5.4 Frequency-Hopping/Rx Demo.....	36
5.4.1 代码流程图.....	36
5.4.2 注意事项.....	37
5.5 Tx-rx Demo.....	38
5.5.1 代码流程图.....	38
5.5.2 代码实现.....	38
5.5.3 注意事项.....	40

6 PAN3029 状态图 .....	41
7 应用说明 .....	43
7.1 低功耗休眠 .....	43
7.2 CAD 功能 .....	43
7.3 灵敏度 RSSI .....	43
7.4 DCDC 功能 .....	44
7.5 EFUSE 功能 .....	44
7.6 RF 参数说明 .....	46
7.7 四线 SPI 和 FIFO .....	47
8 附件 .....	49
8.1 发射功率表 .....	49

## 1 概述

---

本文档主要介绍 PAN3029 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《pan3029.c》、《pan3029.h》、《pan3029\_port.c》、《pan3029\_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 PAN3029 芯片，从 deep sleep 状态至 standby3 状态

```
PAN3029_deepsleep_wakeup
```

- 2、 初始化芯片

```
PAN3029_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

芯片初次上电或复位时调用接口函数。

### 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

```
PAN3029_set_freq  
PAN3029_set_code_rate  
PAN3029_set_bw  
PAN3029_set_sf  
PAN3029_set_tx_power  
PAN3029_set_crc  
PAN3029_set_ldr
```

- 2、刷新参数

```
PAN3029_refresh
```

### 2.2.3 使用方法

芯片初始化完成后，在 standby3 状态下，调用接口函数。

## 2.3 发送流程

### 2.3.1 接口函数

```
rf_enter_continuous_tx();  
rf_continuous_tx_send_data();  
rf_single_tx_data();
```

### 2.3.2 实现功能

- 1、切换状态

```
PAN3029_set_mode
```

- 2、打开 TX\_IO



```
rf_port.set_tx
```

3、设置发射模式

```
PAN3029_set_tx_mode
```

4、进入 TX 状态并发送数据

```
PAN3029_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 SDK 示例。

## 2.4 接收流程

### 2.4.1 接口函数

```
rf_enter_continuous_rx();
```

```
rf_enter_single_timeout_rx();
```

```
rf_enter_single_rx();
```

### 2.4.2 实现功能

1、切换状态

```
PAN3029_set_mode
```

2、打开 RX\_IO

```
rf_port.set_rx
```

3、设置接收模式

```
PAN3029_set_rx_mode
```

4、设置进入接收状态

## 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，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 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。

芯片初次上电或复位时，需要唤醒芯片时使用。

接口函数会设置芯片进入 deepsleep 状态，deepsleep 状态下芯片的寄存器值会被复位。因此使用接口函数唤醒后，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，芯片的寄存器值不会被复位。sleep 唤醒后仍是休眠前的参数配置值。

### 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

PAN3029\_MODE\_DEEP\_SLEEP

PAN3029\_MODE\_SLEEP

PAN3029\_MODE\_STB1

PAN3029\_MODE\_STB2

PAN3029\_MODE\_STB3

PAN3029\_MODE\_TX

PAN3029\_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

PAN3029\_TX\_SINGLE

PAN3029\_TX\_CONTINUOUS

### 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、超时接收、连续接收。

### 参数

Mode 接收模式



PAN3029\_RX\_SINGLE

PAN3029\_RX\_SINGLE\_TIMEOUT

PAN3029\_RX\_CONTINUOUS

## 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

设置超时接收的超时时间，仅在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，SNR 越高代表信号质量越好。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。rf\_irq\_processr() 函数内提供了该函数的用法演示。

伪码示例:

```
// 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

```
int8_t 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。rf\_irq\_process()函数内提供了该函数的用法演示。更多用法可以参考 RSSI 应用参考文档。

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\_get\_channel\_rssi

#### 句法

```
int8_t rf_get_channel_rssi(void)
```

#### 目的

PAN3029 支持信道能量检测功能，可用 channel\_rssi 值来指示该信道下的能量大小。芯片在进入 RX 状态后，可以调用该接口获取信道检测能量，读取信道能量强度的值。

#### 示例：

```
//芯片进入 RX 状态后  
channel_rssi = rf_get_channel_rssi();
```

#### 参数

无

#### 返回值

RSSI 值

### 3.16 rf\_get\_irq

#### 句法

```
uint8_t rf_get_irq(void)
```

#### 目的

读取中断寄存器值。在 rf\_irq\_process 中使用。

#### 参数

无

返回值

中断结果

## 3.17 rf\_clr\_irq

句法

```
uint8_t rf_clr_irq(void)
```

目的

清除当前所有中断。

参数

无

返回值

FAIL 操作失败

OK 操作成功

## 3.18 rf\_set\_refresh

句法

```
uint32_t rf_set_refresh(void)
```

目的

寄存器参数刷新。不会修改寄存器值，参考 SDK 在设置参数后使用。

参数

无

返回值

FAIL 操作失败

OK 操作成功

### 3.19 rf\_set\_preamble

#### 句法

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

#### 目的

设置芯片发射前导码，仅对发射端有效，需要在发射数据前配置。数据发射时，发射内容由“前导码+数据”组成，前导码越长代表整个发射内容的发包时间越长。

#### 参数

参数	范围	描述
Pream	8~65535	发射前导码长度

#### 返回值

FAIL 操作失败

OK 操作成功

### 3.20 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

### 3.21 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

### 3.22 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

## 3.23 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

## 3.24 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

## 3.25 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)

uint8_t rf_get_plhd_len(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《PAN3029 提前中断应用参考文档》。

### 3.26 rf\_set\_mapm\_rx\_on

#### 句法

```
void rf_set_mapm_on(void)
```

#### 目的

使能 mapm 模式，开 mapm 中断允许

#### 参数

无

#### 返回值

无

### 3.27 rf\_set\_mapm\_rx\_off

#### 句法

```
void rf_set_mapm_off(void)
```

#### 目的

关 mapm 模式，关 mapm 中断允许



参数

无

返回值

无

### 3.28 rf\_set\_mapm\_para

句法

```
void rf_set_mapm_para(stc_mapm_cfg_t *p_mapm_cfg)
```

目的

设置 mapm 模式相关参数

参数

p_mapm_cfg->fn	用于配置 mapm preamble 中 Field 数量
p_mapm_cfg->fnm	用于配置 mapm preamble 中每个 Field 重复次数
p_mapm_cfg->gfs	用于配置 Field 中最后一个 Group，其 ADDR 位置功能选择
p_mapm_cfg->gn	用于配置一个 Field 中 Group 个数
p_mapm_cfg->pgl	用于配置第一个 Group 内 Preamble 数量
p_mapm_cfg->pgn	用于配置其他 Group 内 Preamble 数量
p_mapm_cfg->gn	所有的 Field 都发送完毕后 syncword 前 chirp 的数量

返回值

无

### 3.29 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

## 3.30 rf\_enter\_continuous\_rx

### 句法

```
uint32_t rf_enter_continuous_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 rx IRQ 相关中断，并继续进行接收。

### 参数

无

### 返回值

FAIL 操作失败

OK 操作成功

## 3.31 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，等待用户后续操作，如果放弃接收需要主动退出接收状态，否则芯片会再次启动一次超时接收。使用后，用户软件需先切换至 STB3 退出接收状态，再进行后续操作。

### 参数

Timeout	超时时间
---------	------

### 返回值

FAIL 操作失败

OK 操作成功

## 3.32 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 rx IRQ 相关中断，并退出接收状态，用户软件需先切换至 STB3 退出接收状态，再进行后续操作。

### 参数

无

### 返回值

FAIL 操作失败

OK 操作成功

## 3.33 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），用户软件需先切换至 STB3 退出发射状态，再进行后续操作。

## 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

### 3.34 rf\_enter\_continuous\_tx/rf\_continuous\_tx\_send\_data

## 句法

```
uint32_t rf_enter_continuous_tx(void)
```

```
uint32_t rf_continuous_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\_continuous\_tx\_send\_data() 即可，继续发射不需要重复 rf\_enter\_continuous\_tx() 操作。

## 参数

无

## 返回值

FAIL 操作失败

OK 操作成功

## 3.35 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

## 3.36 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\_5/SF\_6/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

## 3.37 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

## 3.38 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。DCDC 模式可以获得更低的接收电流。

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

## 3.39 rf\_set\_ldr

### 句法

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

### 目的

设置芯片 LDR 低速率模式，默认关闭。低速率模式可以进一步降低传输速率，获得更好的传输效果。

### 参数

Mode 模式类型

LDR\_ON

LDR\_OFF

## 返回值

FAIL 操作失败

OK 操作成功

## 3.40 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\_5/SF\_6/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《PAN3029 智能搜索应用参考文档》。

## 3.41 get\_chirp\_time

### 句法

```
uint32_t get_chirp_time(uint32_t bw,uint32_t sf)
```

### 目的

计算单个 chirp 持续时间。

### 参数

SF: SF\_5/SF\_6/SF\_7/SF\_8/SF\_9/SF\_10/SF\_11/SF\_12

BW: BW\_62\_5K/BW\_125K/BW\_250K/BW\_500K

### 返回值

单个 chirp 持续时间, 单位: us。



### 3.42 check\_cad\_rx\_inactive

#### 句法

```
uint32_t check_cad_rx_inactive(uint32_t one_chirp_time)
```

#### 目的

使用 CAD 检测当前信道状态，用于 CAD 接收前的信道检测。

#### 参数

one\_chirp\_time: 单个 chirp 持续时间

#### 返回值

LEVEL\_ACTIVE 信道活跃，应当继续接收

LEVEL\_INACTIVE 信道无信号，应当放弃接收

### 3.43 check\_cad\_tx\_inactive

#### 句法

```
uint32_t check_cad_tx_inactive(void)
```

#### 目的

使用 CAD 检测当前信道状态，用于 CAD 发射前的信道检测。

#### 参数

无

#### 返回值

FAIL 操作失败

OK 操作成功

check\_cad\_tx\_inactive()操作成功后，需要检测 SDK 定义的 cad\_tx\_timeout\_flag 事件判断 CAD 检测结果。开启检测前 cad\_tx\_timeout\_flag 默认为 MAC\_EVT\_TX\_CAD\_NONE，如果检测到信道活跃，不适宜发射，cad\_tx\_timeout\_flag 会变为 MAC\_EVT\_TX\_CAD\_ACTIVE，如果检测到

信道不活跃，适宜发射，`cad_tx_timeout_flag` 会变为 `MAC_EVT_TX_CAD_TIMEOUT`。

Confidential

## 4 SDK移植

本章节详细描述 SDK 移植过程及注意事项。keil 配置 floating point hardware 为 “Not Used”，配置 “One ELF Section per Function” 不勾选，优化等级设置为 “default”。编译代码后，统计 SDK 所需的 Flash 大小为 9125 (8.91kB)，所需 SRAM 大小为 1485 (1.45kB)。

SDK 《PAN3029\_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，芯片收发 IO 切换接口，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 3029 内部 GPIO4 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，芯片有源晶振 IO 控制接口，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 3029 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 PAN3029 模组版本和底板设计情况。

- rf\_irq\_handler(), IRQ 中断处理函数。

PAN3029 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 PAN3029 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

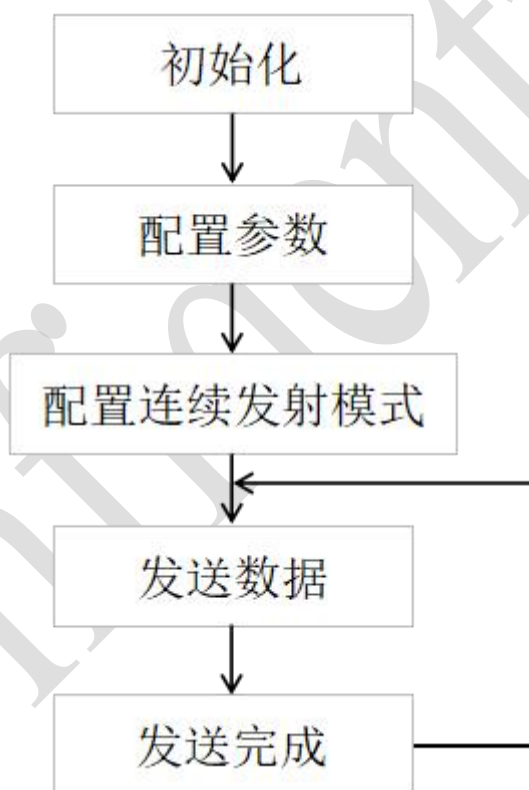


图 5-1 Tx 代码流程图

#### 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_continuous_tx();                             //配置发射模式
    if(rf_continuous_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_continuous_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

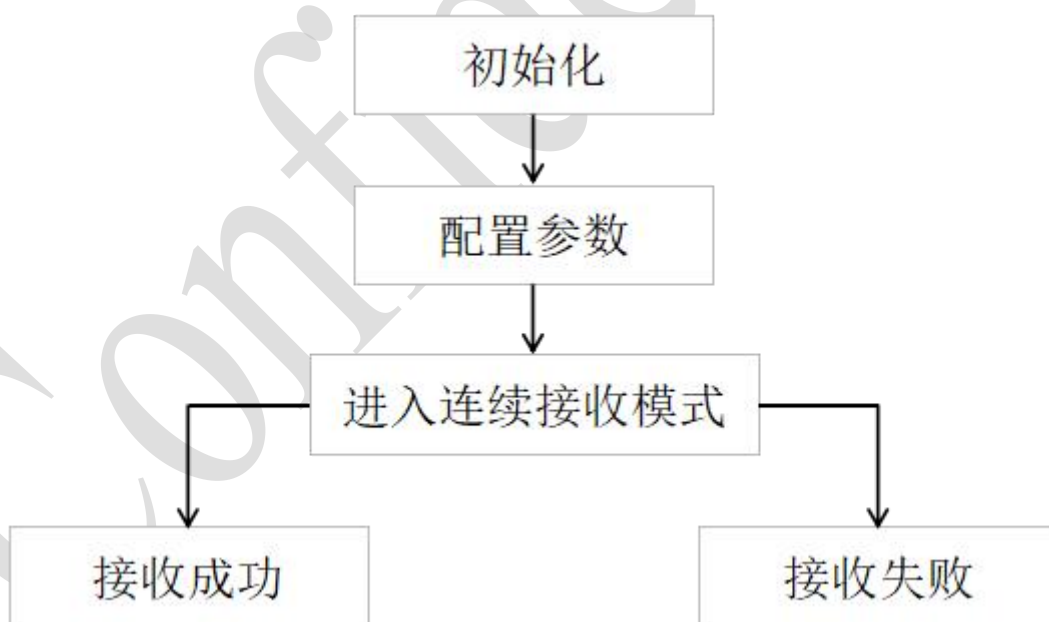


图 5-2 Rx 代码流程图

### 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_continuous_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 Frequency-Hopping/Tx Demo

### 5.3.1 代码流程图

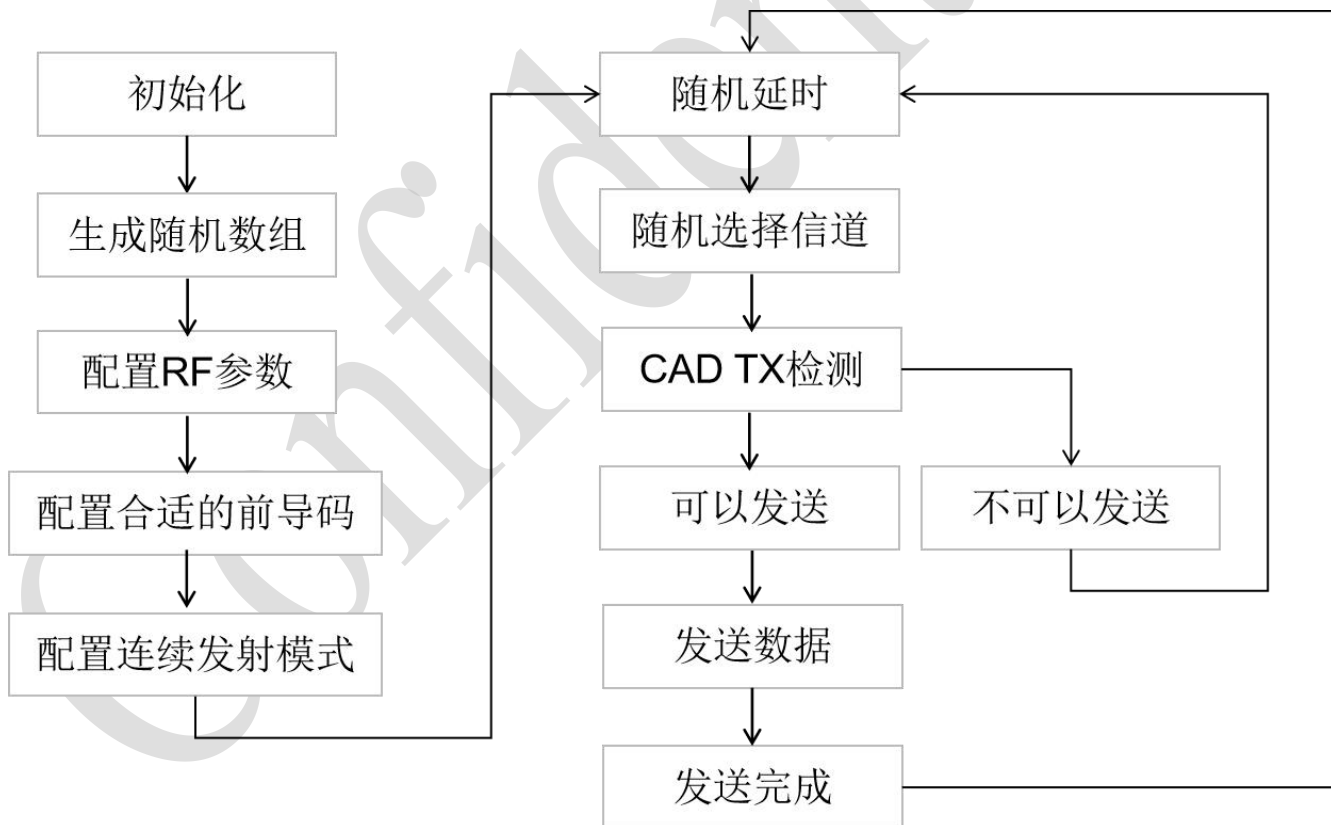


图 5-1 Tx 代码流程图

### 5.3.2 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。

本 Demo 示例中发射端的发射间隔和发射包长是随机的。发射包长 1-255 字节，发射间隔时间大于 150ms，随机数来自初始生成的随机数组。

多频点切换收发时，发射端需要根据不同的 MCU 和代码逻辑实测一下 RX 程序切换耗时，然后根据这个耗时计算 TX 发射端应当配置的前导码个数。例如，对于双频点切换的场景，测得程序切换耗时为 T1，根据 SF 和 BW 计算得 one\_chirp\_time，RF 启动 RX 时有一个固定的启动耗时约 360us。双频点场景极限情况下所需前导码示例表如下：

表 5-1 双频点场景极限情况下所需前导码示例表

RX (双频点扫描)	FREQ1（未检测到，至多 7 个 chirp 时间）	切换（程序执行时间，实测）	FREQ2	切换	FREQ1	FREQ1 接收
TX (FREQ1)	额外前导码					前导码 8    数据

在 TX DEMO 中，TX 端根据计算结果  $((one\_chirp\_time * 7 + 360) * 3 + T1 * 2) / one\_chirp\_time + 8$  配置前导码个数。详情可参考<PAN3029\_SDK 测试结果>表。

## 5.4 Frequency-Hopping/Rx Demo

### 5.4.1 代码流程图

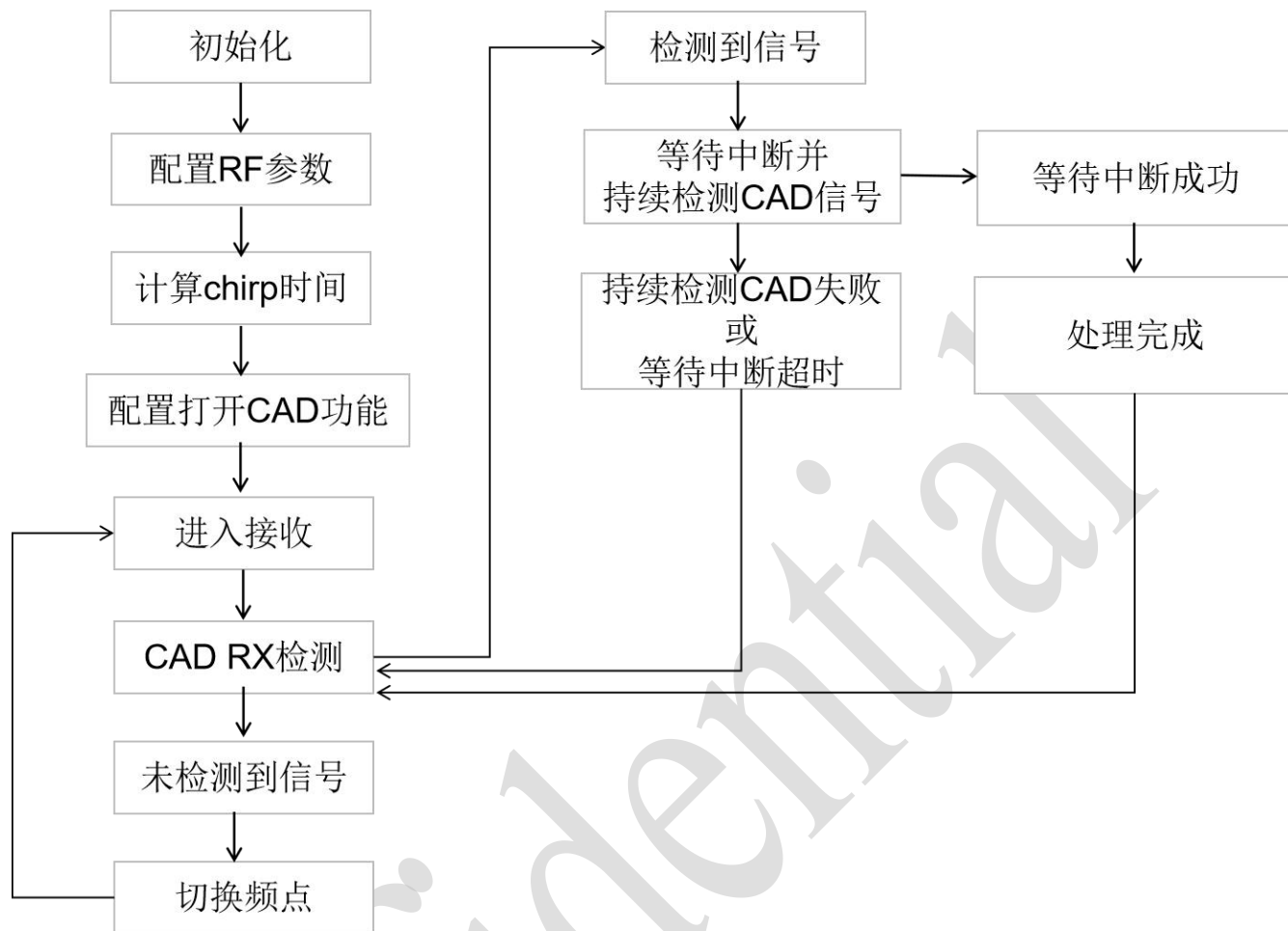


图 5-2 Rx 代码流程图

## 5.4.2 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）。

在 RX DEMO 中，检测到 CAD 信号后，进入等待接收结果状态。这里在等待接收结果期间，建议多读取两次 CAD 信号，确保更高的 CAD 准确性。同是，建议设置一个 RX 端最大接收超时时间，超时时间根据 TX 端最大发包时间确定，可以选择 RX DEMO 中的计时方法，也可以选用 `rf_enter_single_timeout_rx(timeout)` 来接收。

更多 CAD 应用相关说明请参考《CAD 应用参考文档》。

## 5.5 Tx-rx Demo

### 5.5.1 代码流程图

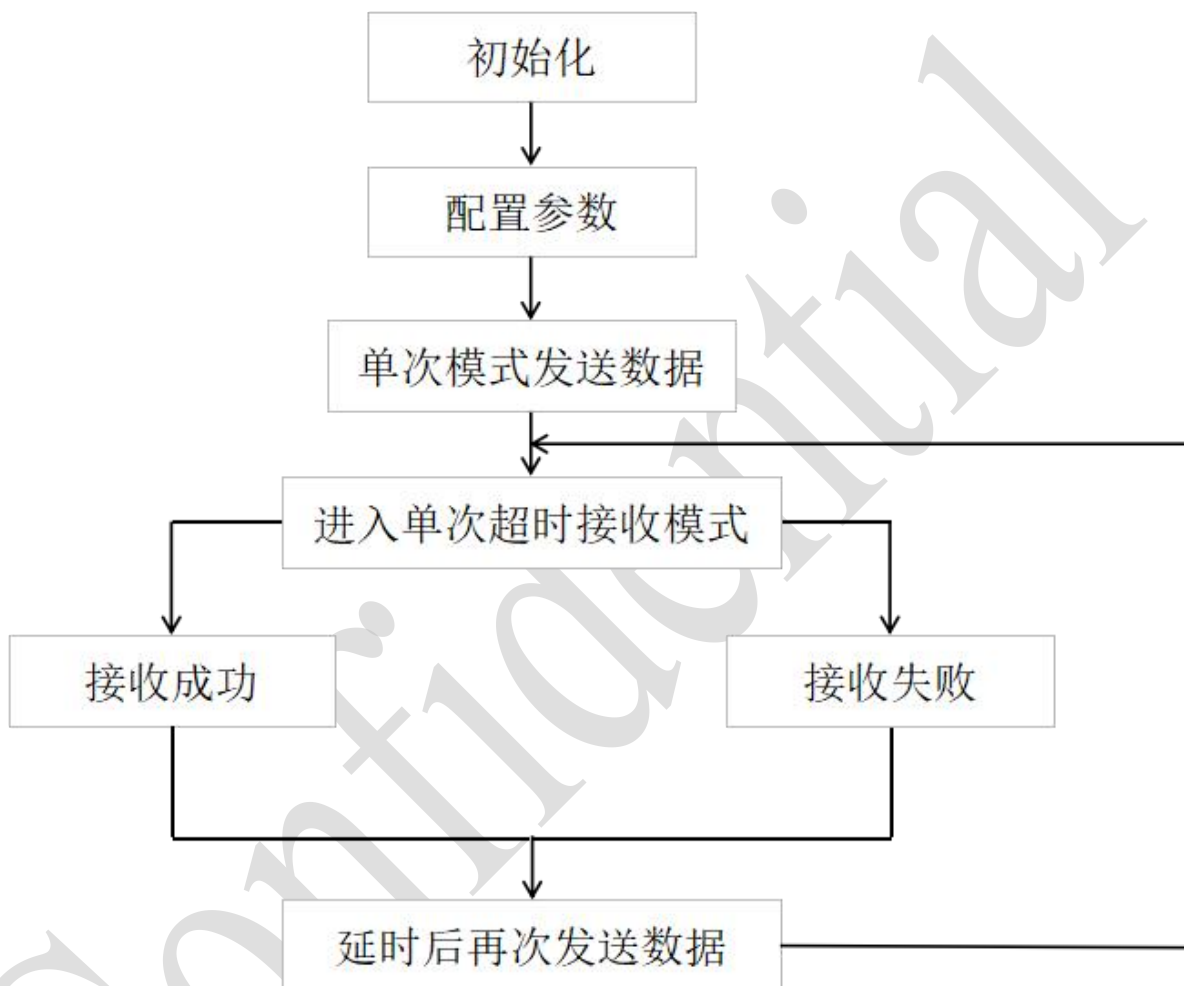


图 5-3 Tx-Rx 代码流程图

### 5.5.2 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);

            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_rcv_flag() == RADIO_FLAG_RXTIMEOUT) || (rf_get_rcv_flag() ==
RADIO_FLAG_RXERR))                                     //接收失败
        {
            rf_set_rcv_flag(RADIO_FLAG_IDLE);
            DDL_Printf("Rxerr\r\n");
            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
            //single_timeout_rx
            rf_enter_single_timeout_rx(15000)                        ;//进入接收
        }
    }
}

```

## 5.5.3 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，PAN3029 芯片在单次模式下（包含单次发射模式，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 stb3 模式，然后再执行后续操作。

## 6 PAN3029状态图

本章节描述的是 PAN3029 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。

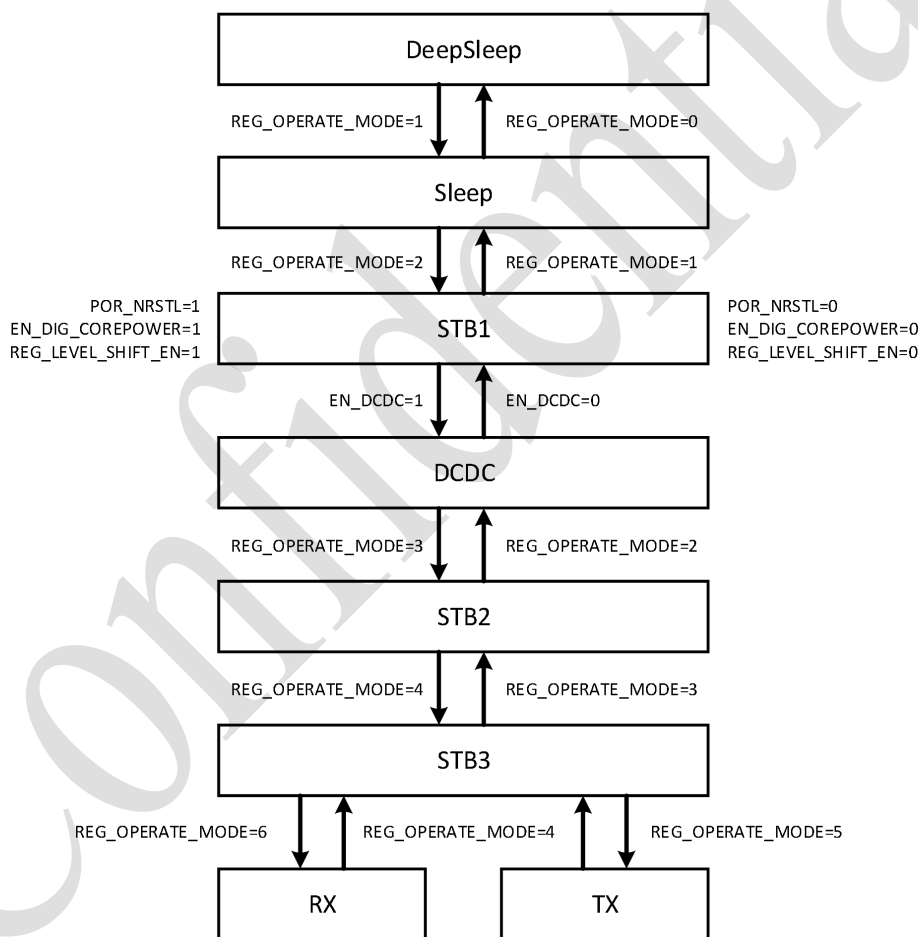


图 6-1 PAN3029 上电流程

PAN3029 上电后有七种状态，包括深度睡眠模式(DeepSleep)、睡眠模式(Sleep)、LDO 工作模式(STB1)、OSC 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。

注：如果芯片外围电路不支持 DCDC，可不配置 EN\_DCDC，直接从 STB1 进入 STB2。SDK 默认不使用 DCDC 功能，如果需要使用 DCDC 功能，请参考对应的应用文档。

Confidential

## 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，PAN3029 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配置。

//deepsleep 休眠示例

<code>rf_deepsleep();</code>	//进入深度休眠
<code>rf_deepsleep_wakeup();</code>	//唤醒
<code>rf_set_default_para();</code>	//重新配置参数
<code>rf_enter_continous_rx();</code>	//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《PAN3029 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-12 到-110，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《PAN3029 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 4.77mA，默认该功能关闭。SDK 中提供了一个接口函数开启或关闭 DCDC 功能。

```
uint32_t PAN3029_set_dcdc_mode(uint32_t dcdc_val)
```

参数 dcdc\_val

DCDC\_ON     开启

DCDC\_OFF    关闭

DCDC 功能可在 STB1~STB3 及 RX 模式开启。

开启 DCDC 功能后，当芯片需要发送时，需要提前手动关闭 DCDC 功能。

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。

## 7.5 EFUSE 功能

PAN3029 支持 82Bytes 的 eFuse 空间区域供客户使用，可用于存储掉电不丢失的数据。关于 eFuse 功能的使用，SDK 中提供了四个接口函数。

```
uint32_t PAN3029_efuse_on(void)
```

目的

设置芯片打开 eFuse 功能

参数

无

返回值

FAIL    操作失败

OK     操作成功

```
uint32_t PAN3029_efuse_off(void)
```

目的

设置芯片关闭 eFuse 功能

参数

无

返回值

FAIL 操作失败

OK 操作成功

`uint8_t PAN3029_efuse_read_byte(uint8_t reg_addr, uint8_t efuse_addr)`

目的

读取指定 efuse 地址存储数据

参数

`reg_addr` efuse 进入地址，客户区默认 0x3c

`efuse_addr` efuse 内部地址，客户区 0x2d~0x7f

返回值

`value` 读取指定 efuse 地址返回的数据

`void PAN3029_efuse_write_byte(uint8_t reg_addr, uint8_t efuse_addr, uint8_t value)`

目的

往指定 efuse 地址写入想存储的数据

参数

`reg_addr` efuse 进入地址，客户区默认 0x3c

`efuse_addr` efuse 内部地址，客户区 0x2d~0x7f

`value` 欲写入 efuse 的数据

返回值

无

eFuse 读写需要特别注意：

1.eFuse 只能烧写一次，一旦进行烧写，无法修改。

- 2.不能对 eFuse 同一地址重复写，否则容易导致 eFuse 器件损坏。
- 3.写 eFuse 的电压范围是 2.25V 到 2.75V，典型值为 2.5V，因此在配置 eFuse 时，建议外部 VDD1 使用 2.5V 电压。读电压可正常使用 3V 电压。
- 4.eFuse 仅支持在 STB3 读写操作，请勿在其他模式下读写。
- 5.eFuse 区域不支持 SPI 和 I2C 的连续读写操作。
- 6.eFuse 读写的最高时钟速率为 8MHz。

## 7.6 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 SF5-12，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 是数据流中有效部分（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。

### ➤ 校验 (CRC)

CRC 是 TX 端设置，RX 端不设置。RX 端接收信号中的信息，在帧结构中读取 CRC，来决定要不要进行 CRC 检查。

### ➤ 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 PAN3029 计算器.xlsx》

## 7.7 四线 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。

PAN3029 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 Single 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：

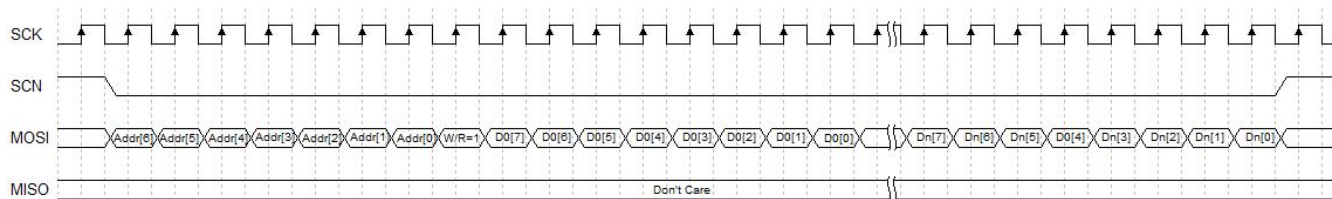


图 7-1 SPI 写时序

SPI 读时序如下：

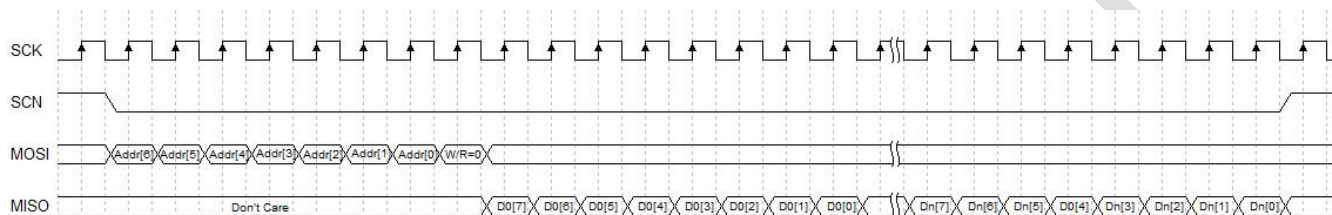


图 7-2 SPI 读时序

PAN3029 具有 255bytes 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。

## 8 附件

### 8.1 发射功率表

档位 \ 频率	410MHz		430MHz		450MHz		460MHz	
	功率(dBm)	电流(mA)	功率(dBm)	电流(mA)	功率(dBm)	电流(mA)	功率(dBm)	电流(mA)
1	-18.733	7.267	-18.165	7.418	-18.767	7.546	-19.679	7.607
2	-8.276	9.42	-7.936	9.563	-8.596	9.671	-9.548	9.758
3	1.599	17.935	1.748	18.349	0.947	18.725	-0.111	18.932
4	3.925	20.301	4.513	20.661	4.114	20.967	3.348	21.439
5	4.921	21.76	5.329	22.183	4.79	22.662	3.865	23.076
6	5.413	22.933	5.501	23.385	4.776	23.811	3.841	24.136
7	7.104	26.605	7.611	26.904	7.345	27.468	6.805	28.54
8	7.655	28.062	8.152	28.282	7.951	28.995	7.491	30.298
9	8.638	30.074	9.185	30.663	8.755	31.328	8.117	32.387
10	8.966	32.23	9.29	31.92	9.106	32.679	8.897	34.62
11	10.274	35.676	10.746	36.083	10.571	37.273	10.298	39.578
12	10.816	39.495	10.988	38.391	10.753	38.94	10.56	41.211
13	11.567	41.972	11.815	41.256	11.612	42.165	11.468	44.97
14	12.83	46.616	13.205	46.748	13.077	48.484	12.919	52.149
15	13.782	50.643	14.252	51.595	14.182	54.016	14.001	58.272
16	14.871	55.72	15.404	57.538	15.36	60.671	15.015	64.89
17	15.404	58.355	15.957	60.647	15.797	63.599	15.317	67.146
18	16.052	63.254	16.602	65.475	16.522	69.186	16.267	74.527
19	16.493	66.892	16.956	68.616	16.899	72.499	16.738	78.608
20	17.414	73.954	17.868	75.84	17.8	80.261	17.632	86.882
21	18.229	87.976	18.297	85.286	17.994	86.519	17.694	90.391
22	19.436	102.128	19.414	98.008	19.07	98.768	18.736	102.895
23	19.754	105.378	19.771	101.687	19.433	102.835	19.105	107.278

注：功率档位为 PAN3029\_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。