武汉芯源半导体有限公司
WUHAN XINYUAN SEMICONDUCTOR CO., LTD

CW32W031 类 Mesh 通信应用参考手册

应用笔记

版本号：Rev 1.0

声明

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。

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，武汉芯源半导体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。

版权声明

技术支持

若您在使用过程中有任何意见或建议，请随时与我们联系。

网址：www.whxy.com

通信地址：湖北省武汉市东湖新技术开发区光谷大道武大园三路 5 号

邮编：430070

目录

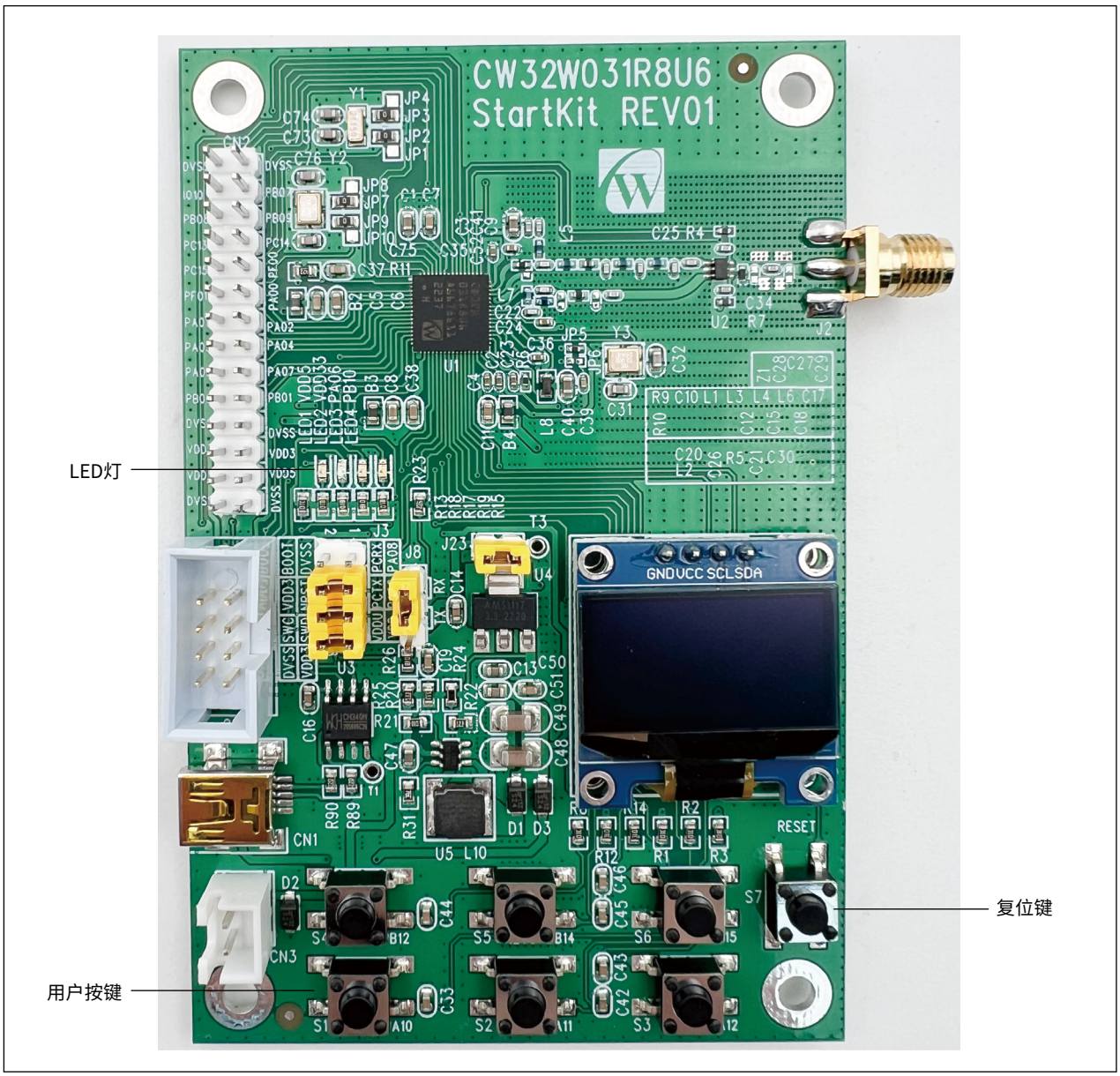
1 概述

CW32W031 ad hoc Network SDK 通信主要实现基于 CW32W031 的组网功能。每个终端节点都可以成为一个中继设备，接收终端可以把数据包中继到发送终端通信范围外的设备（前提是通信范围外设备在接收终端的通信范围内），使得通信的范围得到了扩大。

它实现了类似 Mesh 的多跳网络。每个终端节点接收到数据后会将 TTL（数据包跳跃限制值）减 1，如果 TTL 大于 0，则还会转发收到的数据包。以此实现数据中继功能。

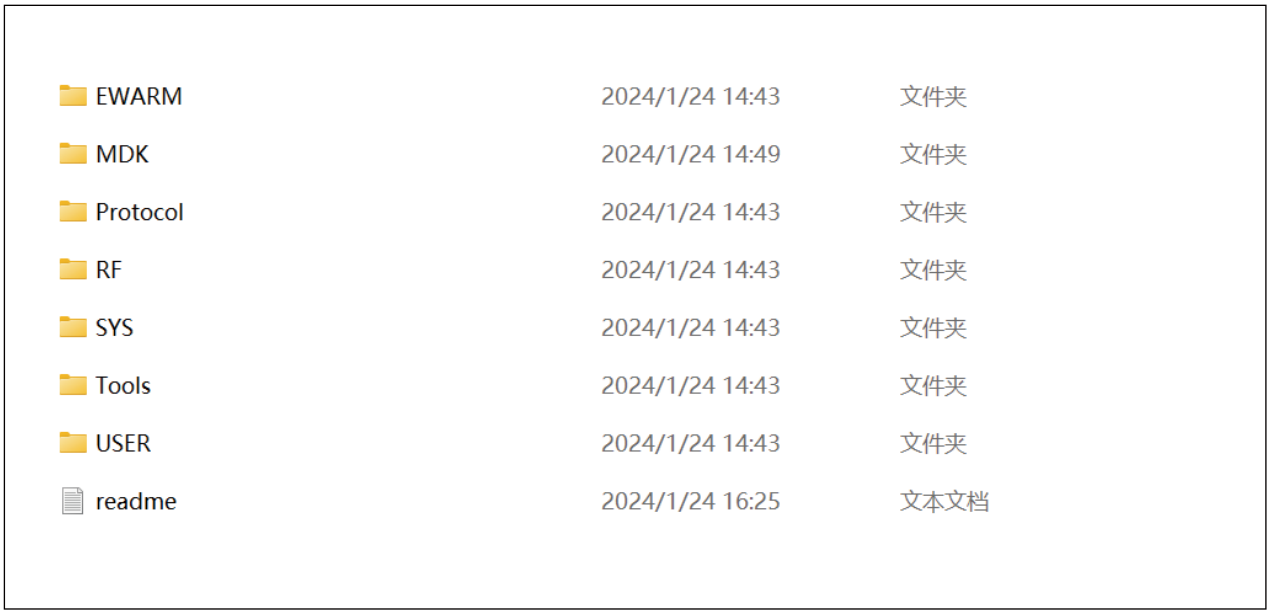
值得注意的是，此协议不支持低功耗应用。

下图是该例程适配的 DEMO 板：



2 工程简介

2.1 目录说明



- EWARM: 目录存放 Iar 的工程文件
- MDK：目录存放 keil 的工程文件
- Tools/Protocol：目录存放例程的一些组件
- RF：目录存放 RF 相关驱动代码文件
- SYS：目录存放 MCU 的一些系统和配置文件
- USER：目录存放应用代码文件
- readme：例程简要说明文件

2.2 使用说明

2.2.1 资源使用情况

	FLASH	RAM
协议栈	5.5KB	0.2KB
组件（可删减）	5.4KB	0.8KB

2.2.2 工程配置说明

1. 配置文件名称：pro_config.h
2. 配置项说明：
   a. PAN_CFG_RF_BW RF 带宽配置
   b. PAN_CFG_RF_SF RF 扩频因子配置
   c. PAN_CFG_RF_FREQ RF 通信频率配置
   d. PAN_CFG_RF_CR RF 码率配置
   e. PAN_CFG_NET_SEND_TIMEOUT 发送超时时间配置，单位 ms
   f. PAN_CFG_NET_CONFLICT_TIMEOUT 发送时检测信号冲突的次数
   g. PAN_CFG_NET_TTL_MAX 数据包存活节点数
   h. PAN_CFG_MEM_UART_CACHE Debug 串口接收缓存区大小
   i. PAN_CFG_LOG_LEVEL 系统日志打印级别配置

2.2.3 工程开发说明

1. 命令行工具使用
   a. 例程自带了命令行工具，如需使用命令行工具，需要调用 xcmd_init 函数，并且将输出字符和输入字符的函数作为参数传递。
   b. 在程序中需要调用持续调用 xcmd_task 来保证工具的正常执行。

示例代码：

```
#include “xcmd.h”
int main()
{
        xcmd_init(fgetc, fputc);
        while(1)
        xcmd_task();
}
```

2. 组网协议使用

   a. 组网协议需要调用 chirp_init 函数初始化协议栈。并且传入三个参数，分别是发送成功后需要执行的回调函数，接收成功后需要执行的回调函数，毫秒级延时函数。

   b. 文件 chirp_networks.h 中的宏 NET_RANDOM() 需要实现随机数生成函数。

   c. 文件 chirp_networks.h 中的宏 CHECK_CAD() 需要实现 GPIO11(PB07) 的电平检测函数。

   d. 在程序中需要调用持续调用 chirp_task 函数来保证协议栈正常执行。

   e. 发送数据有两种方式，一种是调用 chirp_send 函数直接发送。另一种是调用 chirp_set_send_flag 函数设置标志后，由协议栈自动发送，此方式适合中断的方式。 发送是非阻塞式的，执行完会立即返回，其结果会通过 chirp_task 函数返回。

示例代码：

```
#include “chirp_networks.h”
#include “pan_err.h”
int main()
{
        uint8_t data[10];
        chirp_init(send_cb, recv_cb, delayms);
        // 下面两个发送函数仅作示例演示，实际使用时不能同时调用 chirp_send(data, 10);
        chirp_set_send_flag(1, data, 10)
        while(1)
        pan_err_t ret = chirp_task();
        // 应用时可以根据 ret 的返回值做业务逻辑
}
```

3. 日志组件使用

   a. 日志组件支持打印日志级别设置，级别从高到低分别是 ERROR, WARNING, INFO, DEBUG, TEMP。

   b. 日志级别在 pro_config.h 中设置

示例代码：

```
#include “userlog.h”
int main()
{
        BASLOG(LOG_LEVEL_DEBUG, “hello world\n”);
}
```

2.2.4 操作说明

1. 编译工程并将固件烧录到 Demo 板中 (如已烧录则忽略此步骤)。
2. 将 Demo 板供电并放置到合适的位置，以 3 块 Demo 板通信为例，第 3 块板子需要处于第 1 块发送板的通信范围外、第 2 块中继板通信范围内，才能接收到第 2 块转发来的数据（可以在 radio.h 中修改 DEFAULT_PWR 的值来改变通信范围）。
3. 按压需要发送数据的 Demo 板的用户按键。
4. 观察其他 Demo 板的 LED 灯状态变化。

   接收 Demo 板的 LED 灯显示说明如表 2- 1 所示：第一个中继节点为第一跳，对应的 TTL 值为 7，之后每中继一个节点 TTL 值就会减 1，可通过串口打印信息来验证。

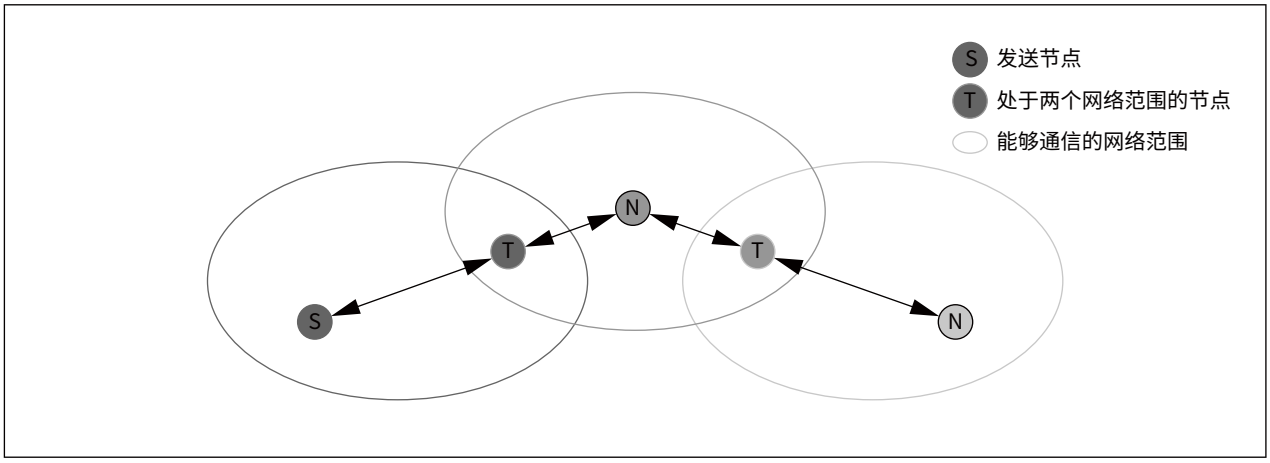
表 2-1 接收 LED 灯状态说明表

中继节点顺序	LED 灯状态
第一跳	两灯慢闪
第二跳	LED3 闪 LED4 亮
第三跳	LED3 暗 LED4 亮
第四跳	LED3 慢闪 LED4 快闪
第五跳	两灯全暗
第六跳	LED3 亮 LED4 闪
第七跳	两灯快闪

3 组网协议

3.1 协议说明

下图来说明各节点的通信范围：



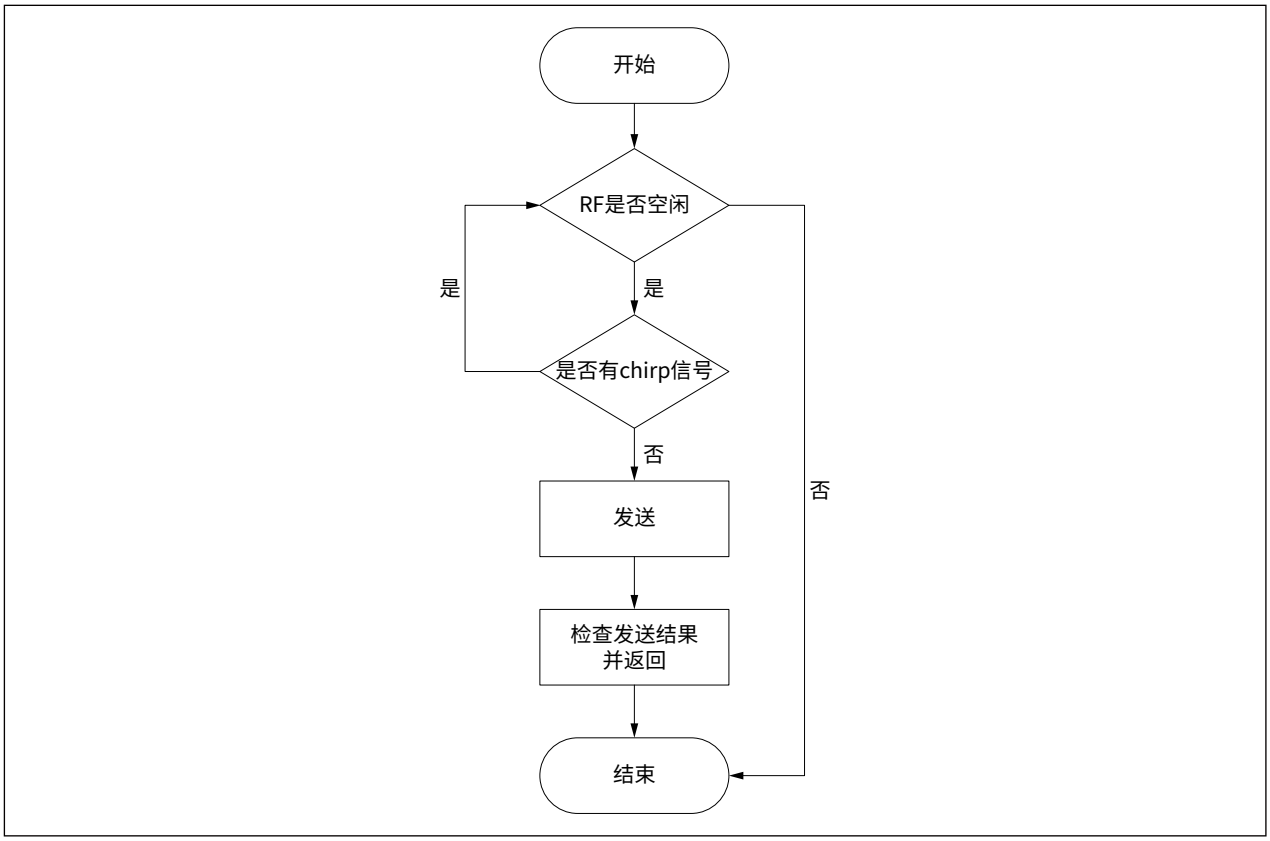
接收节点 N 处于发送节点 S 的通信范围外，S 要发送数据给节点 N，就需要通过中继节点 T 来转发；依此明确每个节点的通信状态：节点 S 发送数据给通信范围内的节点 T，节点 T 收到数据后自动转发，而节点 N 处于节点 T 的通信范围内，它就会接收到转发过来的数据，之后节点 N 再转发数据，依序执行，实现了数据的网络多跳（需要注意的是，每个节点在接收到数据并转发时，其通信范围内若有多个待接收节点，那么这些节点都会接收到转发的数据，因此需要严格注意每个节点的通信范围）。

3.1.1 数据包格式

字节	字段	描述	备注
0	SYNC	同步字	0xEC
1	TTL	数据包生存节点数	节点越多延时越大
2-3	RAND	随机数	/
4	LEN	数据长度	最大值 250
5-N	DATA	数据	最多 250Byte

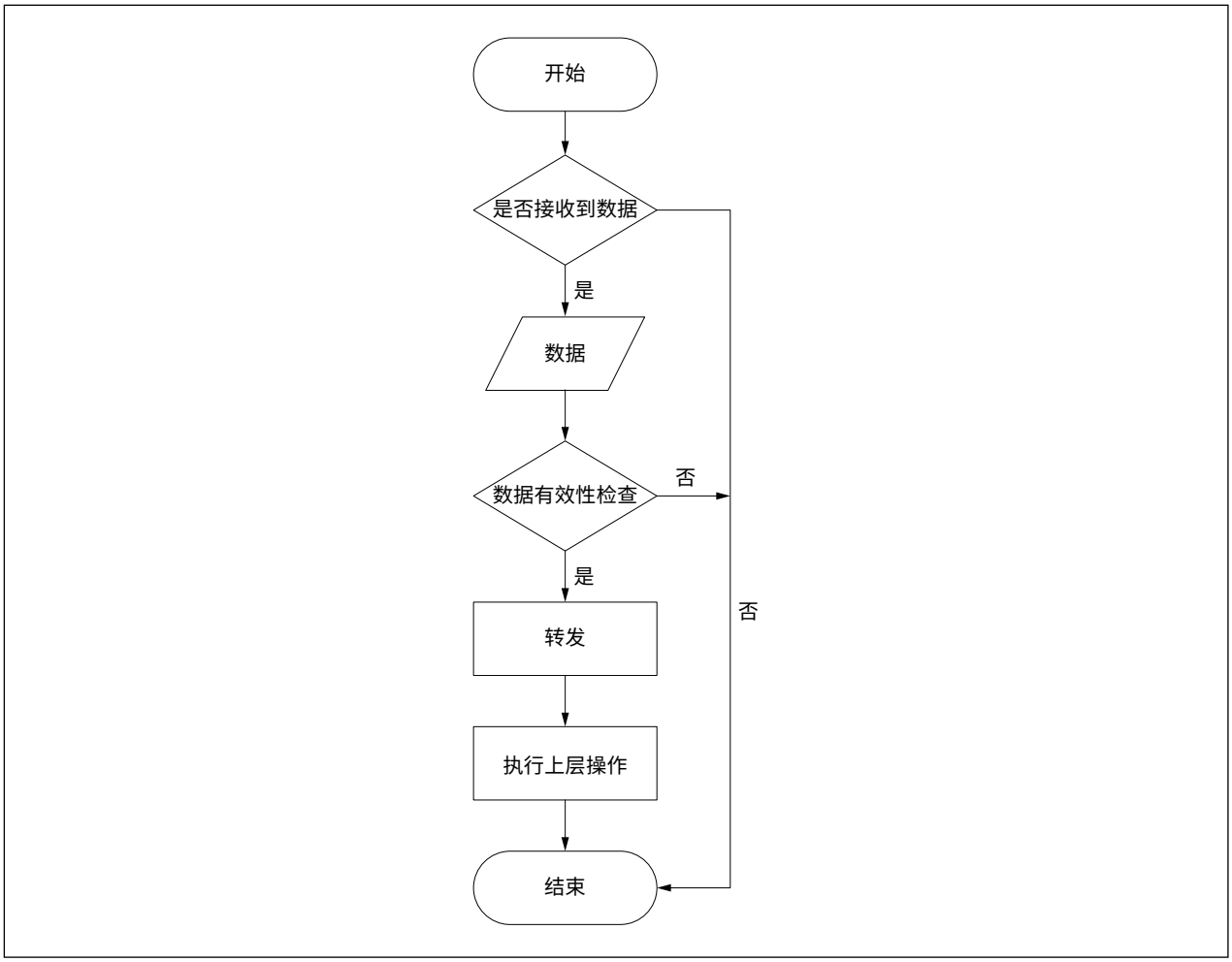
3.1.2 发送流程

如下图所示：



3.1.3 接收流程

如下图所示：



3.2 协议收发包说明

3.2.1 时序说明

TX 代表发送状态，RX 代表接收状态，FW 代表转发数据。

实际应用中，发包的速率需要控制。在一个收发周期（周期大小由数据长度决定）内，由于收发和中继各占用了三分之一的时间，所以两次发送的间隔至少间隔三分之二个周期时间。否则例程会返回 RF 忙碌的错误。

3.2.2 中继和延时

协议最大支持 255 跳，实际使用时需要结合业务需要和最大延时接受程度调整跳跃节点限制。例程中默认的配置（SF：8　　BW：500K　　CR：4/5），数据长度为 10Byte，7 跳总延时约为 154ms。

3.3 协议接口说明

1. 组网协议初始化函数（必须调用）

   pan_err_t chirp_init（chirp_send_callback tx_cb, chirp_recv_callback rx_cb, delayms delay）

   参数：tx_cb 发送数据成功的回调函数

   rx_cb 接收数据成功的回调函数

   delay 毫秒级的延时函数

   返回值：PAN_OK 执行成功 other 执行失败

2. 组网任务函数（必须调用）

   pan_err_t chirp_task（）

   参数：无

   返回值：PAN_OK 无需处理

   TRANS_RECV_SUCCESS 成功接收数据

   TRANS_SNED_SUCCESS 成功发送数据 other 有操作失败，具体错误见错误代码

3. 发送函数

   void chirp_set_send_flag（uint8_t flag, uint8_t *data, uint8_t len）

   参数：flag 发送设置为 1

   data 发送的数据

   len 发送的长度

   返回值：无

4 版本历史

表 4-1 文档版本历史

日期	版本	变更信息
2024-01-30	Rev 1.0	初始发布