PAN1080 FreeRTOS PRF Sample Application Note

PAN-CLT-VER-B0, Rev 0.1

PanchipMicroelectronics

www.panchip.com

修订历史

版本	修订日期	描述
V0.1	2022-12-12	初始版本创建

目录

第 1 章 例程演示内容 4
1.1 例程介绍 4
1.1.1 RTOS 简介 4
1.2 环境准备 5
1.2.1 软件环境 5
1.2.1.1 待测代码 5
1.2.1.2 软件工具 5
1.2.2 硬件环境 5
第 2 章 例程演示流程 6
2.1 环境配置 6
2.1.1 例程编译烧录 6
2.1.2 硬件接线 6
2.2 例程工作流程 6
2.2.1 TX 工程 6
2.2.2 RX 工程 7
2.2.3 Sample 配置说明 10
2.2.4 开发说明 11

第1章 例程演示内容

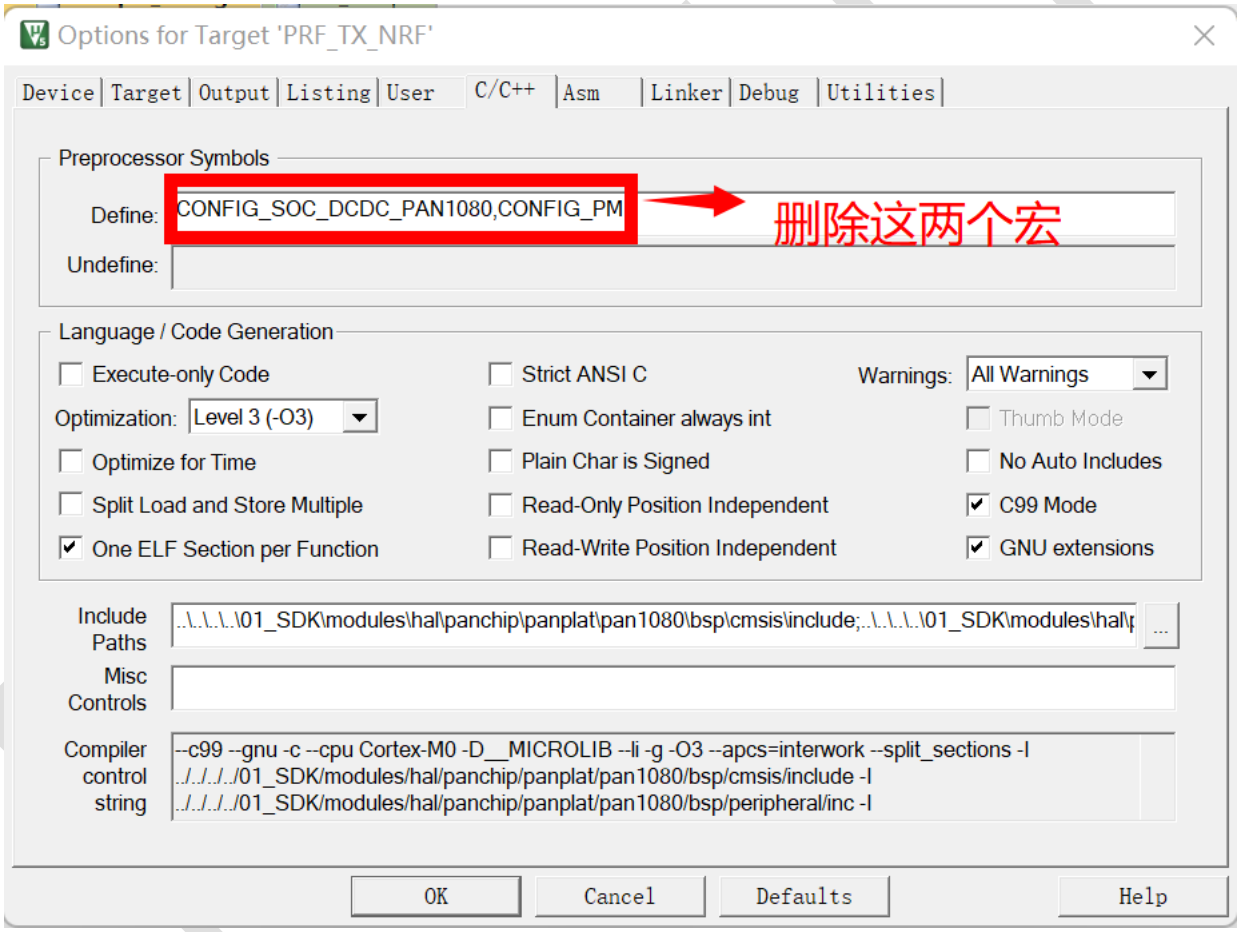
1.1 例程介绍

PRF 例程分为两个工程，一个是 TX 发射工程，一个是 RX 接收工程。

两个例程均支持 FreeRTOS 实时操作系统。其中，TX 工程开启了低功耗功能，RX 工程因为需要一直保持接收状态，故没有开启低功耗。

目前低速时钟默认为 RCL。

例程默认使用 DCDC 模式，如果需要切换到 LDO 模式则做如下修改，但是 LDO 模式的功耗会变大。



1.1.1 RTOS 简介

1. FreeRTOS kernel 的版本为 *202210.01 LTS*

2. FreeRTOS 的系统配置文件 FreeRTOSConfig.h 存放于每个工程的 src 目录：

<PAN1080-DK>\03_MCU\mcu_samples\PRF_RX_FreeRTOS\src\FreeRTOSConfig.h

<PAN1080-DK>\03_MCU\mcu_samples\PRF_TX_FreeRTOS\src\FreeRTOSConfig.h

3. 更多的 FreeRTOS 使用说明请访问官网进行了解。

1.2 环境准备

1.2.1 软件环境

1.2.1.1 待测代码

测试工程文件：

<PAN1080-DK>\03_MCU\mcu_samples\PRF_RX_FreeRTOS\keil\PRF_RX_FreeRTOS.uvprojx

<PAN1080-DK>\03_MCU\mcu_samples\PRF_TX_FreeRTOS\keil\PRF_TX_FreeRTOS.uvprojx

测试源文件目录：

<PAN1080-DK>\03_MCU\mcu_samples \PRF_RX_FreeRTOS\src

<PAN1080-DK>\03_MCU\mcu_samples \PRF_TX_FreeRTOS\src

1.2.1.2 软件工具

1、SecureCRT（用于显示 PC 与 EVB 的交互过程，打印 log 等）

1.2.2 硬件环境

1、PAN1080 EVB 2 块

a) UART0（测试交互接口，TX: P00，RX: P01，波特率: 921600）

b) SWD（用来调试和烧录程序，SWDCLK: P46，SWDIO: P47）

2、JLink（SWD 调试与烧录工具）

第2章 例程演示流程

2.1 环境配置

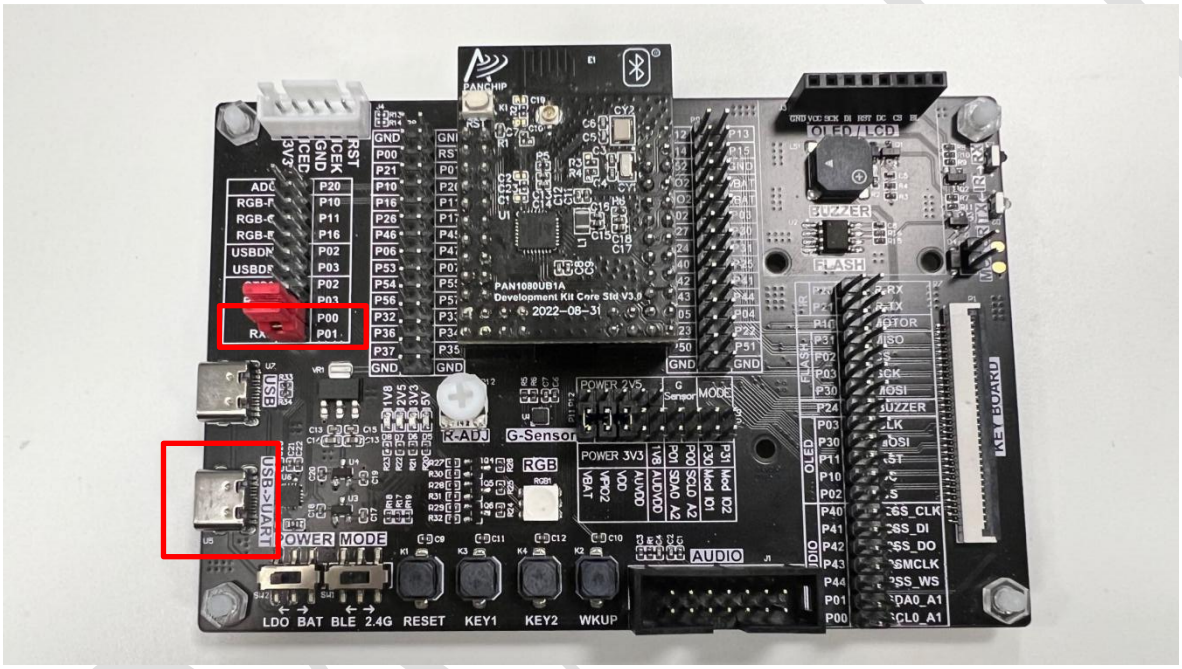
2.1.1 例程编译烧录

准备 2 块 PAN1080 的 EVB 板，分别给 EVB 烧录 TX 工程的程序和 RX 工程的程序。

2.1.2 硬件接线

接线方面，需要：

1. 将 EVB 板的 RX0 和 TX0 进行跳线，然后连接 USB->UART 到 PC。



2.2 例程工作流程

2.2.1 TX 工程

1. 用户的应用程序入口为 app_main 函数。

2. RF 配置如下图所示：

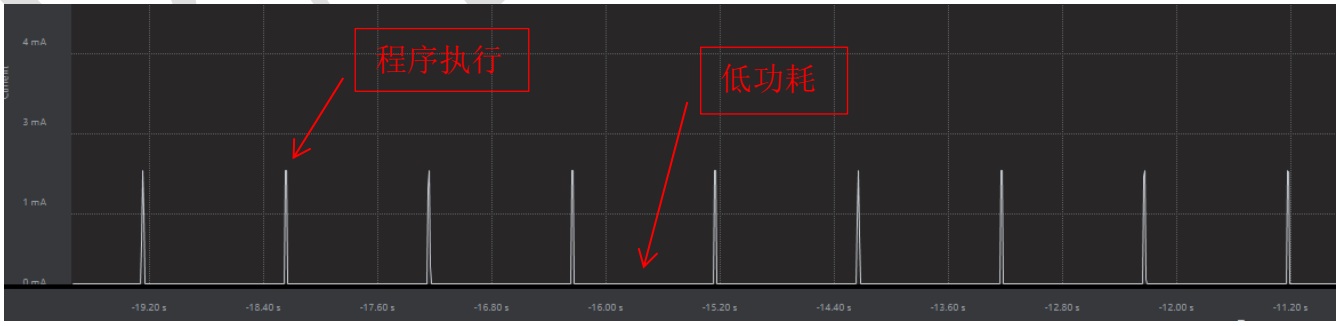
```
pan_prf_config_t tx_config = {
    .work_mode          = PRF_MODE_ENHANCE,
    .chip_mode          = PRF_CHIP_MODE_SEL_NORDIC,
    .trx_mode           = PRF_TX_MODE,
    .phy                = PRF_PHY_1M,
    .crc                = PRF_CRC_SEL_CRC8,
    .src                = PRF_SRC_SEL_NOSRC,
    .dev                = PRF_DEV_BLE,
    .rx_timeout         = 50000,        //us
    .rf_channel         = 2410,
    .tx_no_ack          = DISABLE,
    .nrf52_mode         = ENABLE,
    .rx_length          = 10,
    .addr_length        = 5,
    .crc_include_addr   = DISABLE,
    .tx_trans_time      = 70,           //us
    .rx_trans_time      = 100,          //us
    .addr               = { 0x72, 0x76, 0x41, 0x73, 0x71 },
};
```

3. 程序实现每 1 秒发送一包数据，并接收 RX 端响应的应答包。

4. 日志打印如下：

```
CPU @ 64000000Hz
proprietary rf tx thread start......
tx success
rx data:0x11 0x22 0x33 0x44 0x55 0x66 0x77 0x88 0x99 0xAA
tx success
rx data:0x12 0x22 0x33 0x44 0x55 0x66 0x77 0x88 0x99 0xAA
tx success
rx data:0x13 0x22 0x33 0x44 0x55 0x66 0x77 0x88 0x99 0xAA
tx success
rx data:0x14 0x22 0x33 0x44 0x55 0x66 0x77 0x88 0x99 0xAA
```

5. 电流波形：程序定时 1 秒发送一次，所以线程唤醒会出现电流脉冲，执行完毕后会再进入低功耗。



2.2.2 RX 工程

1. 用户的应用程序入口为 app_main 函数。

2. RF 配置如下图所示：

```
pan_prf_config_t rx_config = {
    .work_mode          = PRF_MODE_ENHANCE,
    .chip_mode          = PRF_CHIP_MODE_SEL_NORDIC,
    .trx_mode           = PRF_RX_MODE,
    .phy                = PRF_PHY_1M,
    .crc                = PRF_CRC_SEL_CRC8,
    .src                = PRF_SRC_SEL_NOSRC,
    .dev                = PRF_DEV_BLE,
    .rx_timeout         = DISABLE,          //us
    .rf_channel         = 2410,
    .tx_no_ack          = DISABLE,
    .nrf52_mode         = ENABLE,
    .rx_length          = 10,
    .addr_length        = 5,
    .crc_include_addr   = DISABLE,
    .tx_trans_time      = 70,               //us
    .rx_trans_time      = 100,              //us
    .addr               = { 0x72, 0x76, 0x41, 0x73, 0x71 },
};
```

3. 程序正常情况是在 TX 中断中释放信号量，线程等待到信号量后打印接收数据。

4. 日志打印如下：

```
CPU @ 64000000Hz
proprietary rf rx thread start......
tx success
rx data:0x11 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A
tx success
rx data:0x12 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A
tx success
rx data:0x13 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A
tx success
rx data:0x14 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A
```

2.2.3 beacon 工程

1. 用户的应用程序入口为 app_main 函数。

2. RF 初始化函数 proprietary_rf_thread，初始化 rf 基本配置，配置信息介绍参考 PAN1080_RADIO_TX 使用说明.docx。

RF 配置如下图所示：

```
pan_prf_config_t tx_config = {
    .work_mode          = PRF_MODE_NORMAL_M1,
    .chip_mode          = PRF_CHIP_MODE_NRF,
    .trx_mode           = PRF_TX_MODE,
    .phy                = PRF_PHY_1M,
    .crc                = PRF_CRC_SEL_CRC24,
    .src                = PRF_SRC_SEL_EN,
    .dev                = PRF_DEV_BLE,
    .rx_timeout         = 1000,      //us
    .rf_channel         = ADV_CHANNEL_1,
    .tx_no_ack          = ENABLE,
    .nrf52_mode         = ENABLE,
    .rx_length          = 0,
    .addr_length        = 4,
    .addr               = { 0xd6, 0xbe, 0x89, 0x8e },
    .tx_power           = 0,
    .pid_manual_flag    = DISABLE,
    .crc_include_addr   = DISABLE,
    .tx_trans_time      = 140,          //us
    .rx_trans_time      = 150,          //us
};
```

3. 设置广播地址和 beacon 数据，如下图所示：

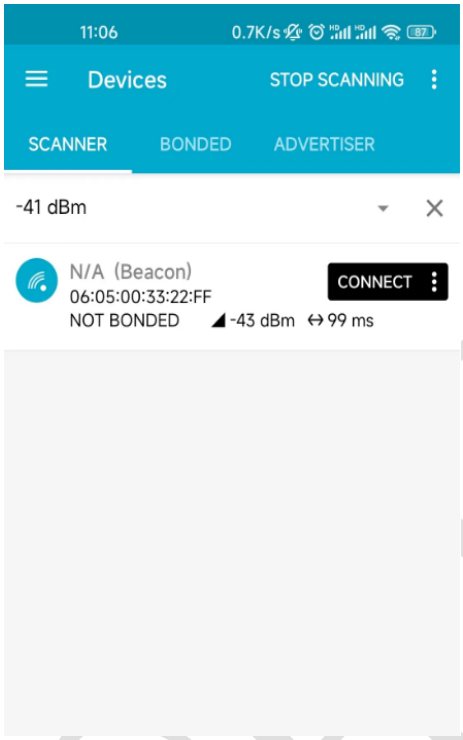
```
panchip_prf_payload_t tx_payload = {
    .data_length        = 36,
    .data               = { 0xff, 0x22, 0x33, 0x00, 0x05, 0x06,
                            0x02, 0x01, 0x06, 0x1A, 0x0ff,
                            0xd1, 0x07, /* Panchip */
                            0x02, 0x15, /* iBeacon */
                            0x18, 0xee, 0x15, 0x16, /* UUID[15..12] */
                            0x01, 0x6b, /* UUID[11..10] */
                            0x4b, 0xec, /* UUID[9..8] */
                            0xad, 0x96, /* UUID[7..6] */
                            0xbc, 0xb9, 0x6d, 0x16, 0x6e, 0x97, /* UUID[5..0] */
                            0x00, 0x00, /* Major */
                            0x00, 0x00, /* Minor */
                            0xc8},      /* RSSI */          //mac addr(06 05 00 33 22 ff) + adv data
};
```

4. 程序实现每 100ms 发送一次广播包，广播频点为 37、38、39。

#define ADV_CHANNEL_NUM 3

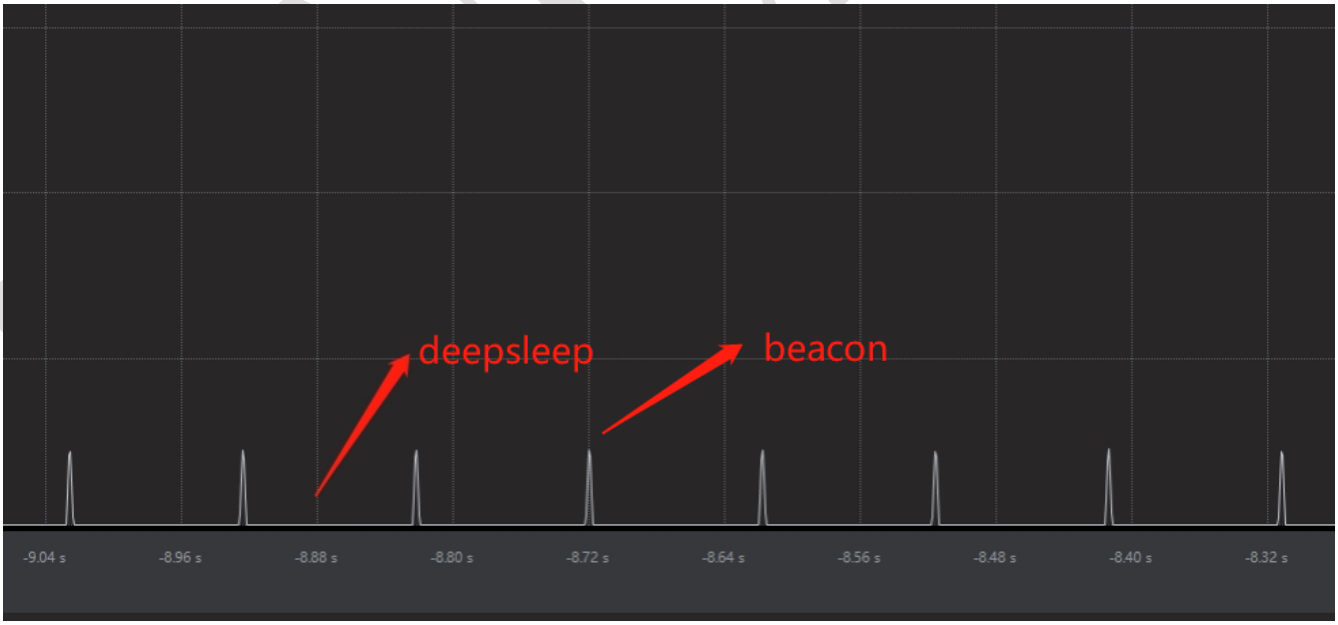
配置广播通道数，默认是 3 个通道。

5. 安卓手机用“nrf connect”app 扫到的 beacon 如下图所示：



广播地址是 0x0605003322FF。

6. 电流波形：程序定时 100ms 发送一次广播包，所以线程唤醒会出现电流脉冲，执行完毕后会再进入低功耗。



2.2.4 Sample 配置说明

每个 sample 的 src 目录下都有一个 sample_config.h 的配置文件，用来配置系统。此文件是必须的头文件，否则会出现编译错误。配置参数说明如下：

① System Clock： 系统主频，可选项为 64M 或者 48M，推荐选择 64M。

② Periph Divide： 外设分频系数，外设的频率 = 主频/外设分频系数。

③ Low-Speed Clock： 低速时钟源选择，默认 RCL。

④ Force Calib RCL： 上电手动校准 RCL，对于测试芯片需要开启。

⑤ Log Enable： 日志使能选项，如不使能则不会初始化日志串口。需要注意的是开启日志会影响系统的功耗，默认是打开的，如果需要测试低功耗电流则需要关闭。

⑥ Low Power Enable： 低功耗使能选项，控制是否开启低功耗。

⑦ 测量低功耗电流需要把 uart 时钟关闭，log disable，如下图所示：

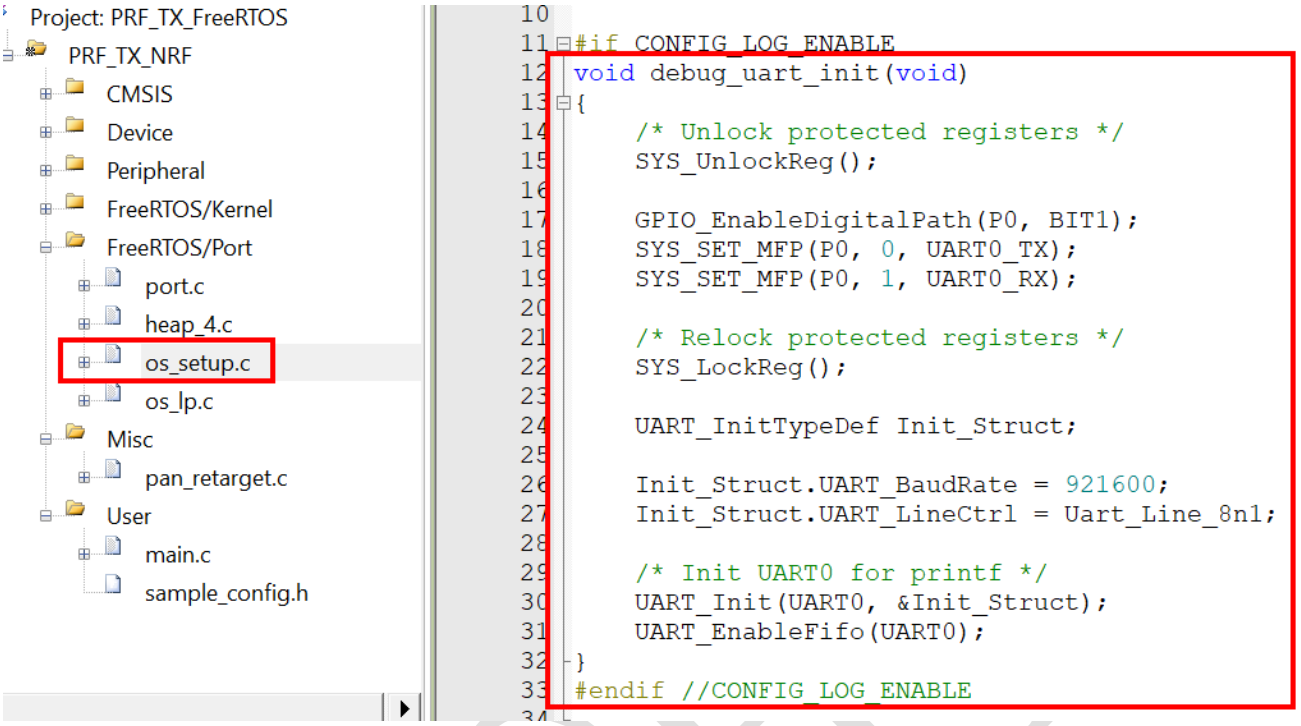
```
#include <stdint.h>

#ifdef __cplusplus
extern "C" {
#endif

#define DBG_ON      0
#define INFO_ON     0
#define WARN_ON     0
#define ERR_ON      0
#define TEST_ON     0
#define ASSERT_ON   0

#define SYS_ABORT()     do { } while (1) //sys_abort()

#define BIT(x)    (0x1UL << (x))

#define LOG(flags, ...)     \
    do {                                \
        if (flags)                      \
            printf(__VA_ARGS__);    \
    } while (0)
```

2.2.5 开发说明

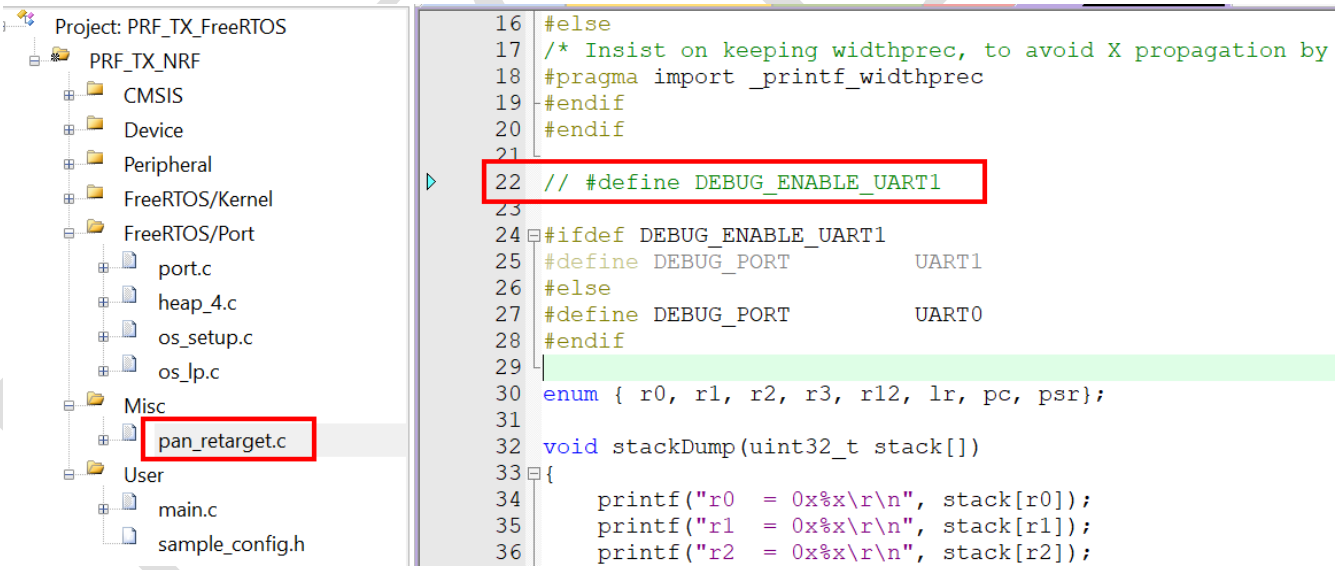
1. 系统时钟目前是全开状态，如果后期为了节省功耗需要关闭没有用到外设模块的时候可以重载如下函数进行外设时钟的关闭操作。

__weak void vPortSystemClockEnable(void);

2. Sample 默认的 debug 串口跟随 EVB 板为 UART0，引脚为 P00 和 P01，如需修改引脚请至如下位置修改：



如果需要使用 UART1 作为 debug 串口，同时要把下图位置的宏打开：



3. Devation 设置

RF 配置的结构体增加了一个成员用于设置 devation，如下所示：

```
24 pan_prf_config_t tx_config = {
25     .work_mode          = PRF_MODE_ENHANCE,
26     .chip_mode          = PRF_CHIP_MODE_SEL_NORDIC,
27     .trx_mode           = PRF_TX_MODE,
28     .phy                = PRF_PHY_2M,
29     .crc                = PRF_CRC_SEL_CRC8,
30     .src                = PRF_SRC_SEL_NOSRC,
31     .dev                = PRF_DEV_BLE,
32     .rx_timeout         = 50000,          //us
33     .rf_channel         = 2410,
34     .tx_no_ack          = DISABLE,
35     .nrf52_mode         = ENABLE,
36     .rx_length          = 10,
37     .addr_length        = 4,
38     .crc_include_addr   = DISABLE,
39     .tx_trans_time      = 110,            //us
40     .rx_trans_time      = 150,            //us
41     .addr               = { 0x71, 0x76, 0x41, 0x76 },
42 };
```

PRF_DEV_BLE 表示 250K

PRF_DEV_NRF 表示 175K

如果需要重新设置 devation 的话，需要重新执行初始化。

4. 注意事项

如果两个 PAN1080 EVB 对测时需要将 crc_include_addr 设置为 DISABLE。

如果 PAN1080 与 NRF52840 对测时看需求设置 crc_include_addr 的属性，如果设置为 ENABLE，RX 的数据应该在 event_crc_err_fun 函数中获取。