



## GR5xx Power Consumption Profile示例手册

版本： 3.1

发布日期： 2023-11-06

版权所有 © 2023 深圳市汇顶科技股份有限公司。保留一切权利。

非经本公司书面许可，任何单位和个人不得对本手册内的任何部分擅自摘抄、复制、修改、翻译、传播，或将其全部或部分用于商业用途。

## 商标声明

**GOODIX** 和其他汇顶商标均为深圳市汇顶科技股份有限公司的商标。本文档提及的其他所有商标或注册商标，由各自的所有人持有。

## 免责声明

本文档中所述的器件应用信息及其他类似内容仅为您提供便利，它们可能由更新之信息所替代。确保应用符合技术规范，是您自身应负的责任。

深圳市汇顶科技股份有限公司（以下简称“GOODIX”）对这些信息不作任何明示或暗示、书面或口头、法定或其他形式的声明或担保，包括但不限于针对其使用情况、质量、性能、适销性或特定用途的适用性的声明或担保。GOODIX对因这些信息及使用这些信息而引起的后果不承担任何责任。

未经GOODIX书面批准，不得将GOODIX的产品用作生命维持系统中的关键组件。在GOODIX知识产权保护下，不得暗或以其他方式转让任何许可证。

深圳市汇顶科技股份有限公司

总部地址：深圳市福田区腾飞工业大厦B座12-13层

电话：+86-755-33338828      邮编：518000

网址：[www.goodix.com](http://www.goodix.com)

# 前言

## 编写目的

本文档介绍如何使用和验证GR5xx SDK中的功耗测试示例，旨在帮助用户快速进行二次开发。

## 读者对象

本文适用于以下读者：

- 芯片用户
- 开发人员
- 测试人员
- 开发爱好者
- 文档工程师

## 版本说明

本文档为第3次发布，对应的产品为低功耗蓝牙GR5xx系列。

## 修订记录

版本	日期	修订内容
1.0	2023-01-10	首次发布
3.0	2023-03-30	新增支持多款芯片的相关描述
3.1	2023-11-06	更新GProgrammer、GRToolbox获取方式。

目录

前言..... I

1 简介..... 1

2 Profile概述..... 2

3 初次运行..... 3

    3.1 准备工作..... 3

    3.2 固件烧录..... 3

    3.3 测试验证..... 3

4 应用详解..... 7

    4.1 运行流程..... 7

    4.2 关键代码..... 7

        4.2.1 电源管理配置..... 7

        4.2.2 指令解析、执行与回应..... 9

        4.2.3 开启通知..... 10

# 1 简介

GR5xx Power Consumption Profile（以下简称“PCP”）示例通过手机端实时设置参数，实现功耗测量场景配置。

本文档将介绍如何使用及验证GR5xx SDK中的Goodix自定义PCP示例。

在进行操作前，可参考以下文档。

表 1-1 文档参考

名称	描述
GR5xx应用及自定义Sample Service	介绍实现自定义Service的相关知识
对应芯片开发者指南	介绍GR5xx SDK以及基于SDK的应用开发和调试
Bluetooth Core Spec	Bluetooth官方标准核心规范
Bluetooth GATT Spec	Bluetooth Profile和服务的详细信息查看地址： <a href="http://www.bluetooth.com/specifications/gatt">www.bluetooth.com/specifications/gatt</a>
J-Link用户指南	J-Link使用说明： <a href="http://www.segger.com/downloads/jlink/UM08001_JLink.pdf">www.segger.com/downloads/jlink/UM08001_JLink.pdf</a>
Keil用户指南	Keil详细操作说明： <a href="http://www.keil.com/support/man/docs/uv4/">www.keil.com/support/man/docs/uv4/</a>

## 2 Profile概述

PCP中定义了功耗服务（Power Consumption Service，PCS）。该服务由Goodix自定义，专属128位UUID为A6ED0501-D344-460A-8075-B9E8EC90D71B，用于数据发送、指令发送以及接收回应。

PCS包含两个特征：

- TX Characteristic：发送数据。
- Setting Characteristic：发送指令定制的功耗测试场景、接收指令的执行回应。


Characteristic的具体描述如表 2-1 所示。

表 2-1 PCS Characteristic

Characteristic	UUID	Type	Support	Security	Properties
TX	A6ED0202-D344-460A-8075-B9E8EC90D71B	128 bits	Mandatory	None	Notify
Setting	A6ED0203-D344-460A-8075-B9E8EC90D71B	128 bits	Mandatory	None	Write, Indicate

## 3 初次运行

本章主要介绍如何运行和验证GR5xx SDK中的PCP示例。

 说明:

SDK\_Folder为对应芯片SDK的根目录。

### 3.1 准备工作

验证并测试PCP示例之前，需要完成以下准备工作。

- 硬件准备

表 3-1 硬件准备

名称	描述
开发板	对应芯片Starter Kit开发板（以下简称“开发板”）
连接线	USB Type-C（GR551x系列使用Micro USB 2.0连接线）
Keysight N6705C	Keysight公司推出的直流功率分析仪

- 软件准备


表 3-2 软件准备

名称	描述
Windows	Windows 7/Windows 10操作系统
J-Link Driver	J-Link驱动程序，下载网址： <a href="http://www.segger.com/downloads/jlink/">www.segger.com/downloads/jlink/</a>
Keil MDK5	IDE工具，支持MDK-ARM 5.20 及以上版本，下载网址： <a href="http://www.keil.com/download/product/">www.keil.com/download/product/</a>
GRToolbox（Android）	Bluetooth LE调试工具，下载网址： <a href="http://www.goodix.com/zh/software_tool/grtoolbox">www.goodix.com/zh/software_tool/grtoolbox</a>
GProgrammer（Windows）	Programming工具，下载网址： <a href="http://www.goodix.com/zh/software_tool/gprogrammer_ble">www.goodix.com/zh/software_tool/gprogrammer_ble</a>
Keysight 14585A	Keysight公司推出的用于电源控制与分析的软件

### 3.2 固件烧录

PCP示例工程的源码位于SDK\_Folder\projects\ble\ble\_peripheral\ble\_app\_pcs。

用户可使用GProgrammer将ble\_app\_pcs.bin下载至开发板。GProgrammer烧录固件的具体操作方法，请参考《GProgrammer用户手册》。

 说明:

ble\_app\_pcs.bin位于SDK\_Folder\projects\ble\ble\_peripheral\ble\_app\_pcs\build。

### 3.3 测试验证

测试验证硬件环境搭建可参考对应芯片功耗模式及功耗测量说明的“环境搭建”章节。

下载有`ble_app_pcs.bin`固件的开发板上电后，将进入Ultra Deep Sleep状态。复位后，长按开发板上的“OK”键3秒以上，系统将发起持续时长为30秒的广播。此后，如果开发板没被其他设备连接，将会广播超时再次进入Sleep状态；如果开发板被其他设备连接，在断连后，也会进入Sleep状态，直至被再次唤醒。按“OK”键可将开发板从Sleep状态唤醒。

#### 说明:

GR5xx SDK PCP示例中，软件上该“OK”键对应的宏定义为APP\_KEY\_OK\_PIN。硬件上GR5xx各芯片对应的开发板电路不同，导致“OK”键对应的实际按键丝印不同（GR551x使用S1，GR5526使用K1，其他芯片系列请参考对应开发板原理图）。为方便描述，本文统一使用“OK”键表示。

具体步骤如下：

#### 1. 建立连接及测量场景设置

利用手机端工具GRToolbox建立连接，具体操作步骤如下：

- (1) 打开GRToolbox APP，选择“应用 > PCS”。
- (2) 连接后开始扫描目标设备。发现广播名为“Goodix\_Power”的设备（广播名可在`user_app.c`文件中进行修改）。

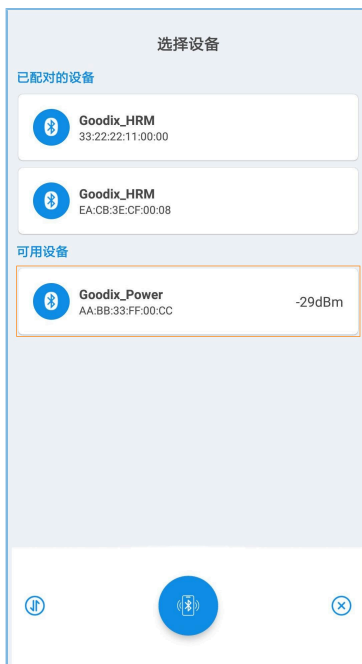


图 3-1 发现“Goodix\_Power”

#### 说明:

本文中GRToolbox的截图仅供用户了解操作步骤，实际界面请参考最新版本GRToolbox。

- (3) 点击“Goodix\_Power”选项并连接，进入功耗测试相关场景设置页面，包括广播间隔、广播数据、连接参数、传输模式、传输功率和开启通知，如图 3-2所示。其中，广播间隔、广播数据和传输功率的设置断连后，重启广播时生效，而连接参数和传输模式的设置仅在当前连



接生效。“最近连接设备”用于在无法通过广播名和服务UUID搜索到设备时根据前一次的设备MAC地址进行搜索。



图 3-2 功耗测试场景设置

(4) 断开连接后，按下开发板上的“OK”键，设备会以设置好的数据长度和连接间隔重新发起广播。

## 2. 观测芯片功耗

测量场景设置完成后，可利用PC端工具Keysight，观测不同场景下芯片功耗，以下截图为GR5526芯片的测量结果以供参考。

场景一：1s间隔的广播态。

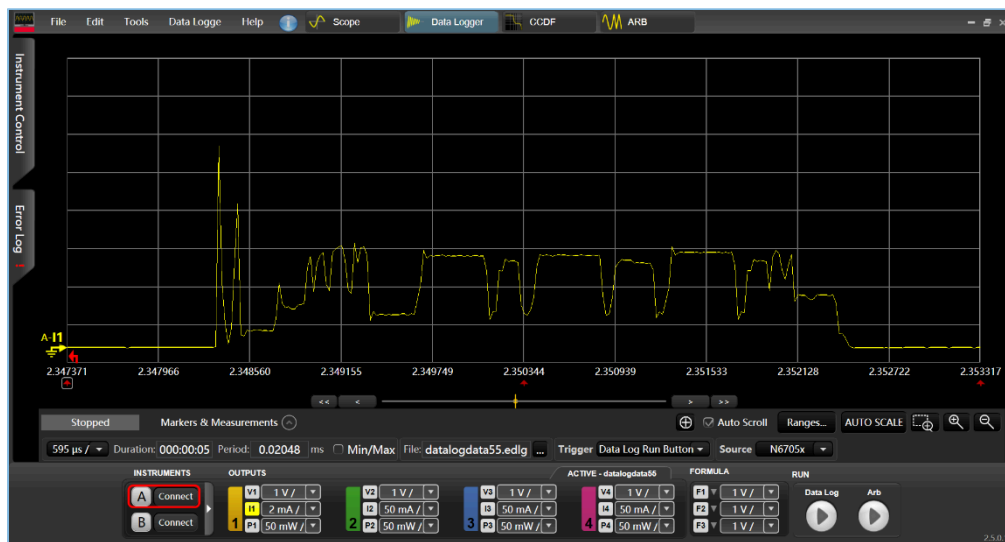


图 3-3 功耗测试场景一

场景二：200 ms间隔的连接态。

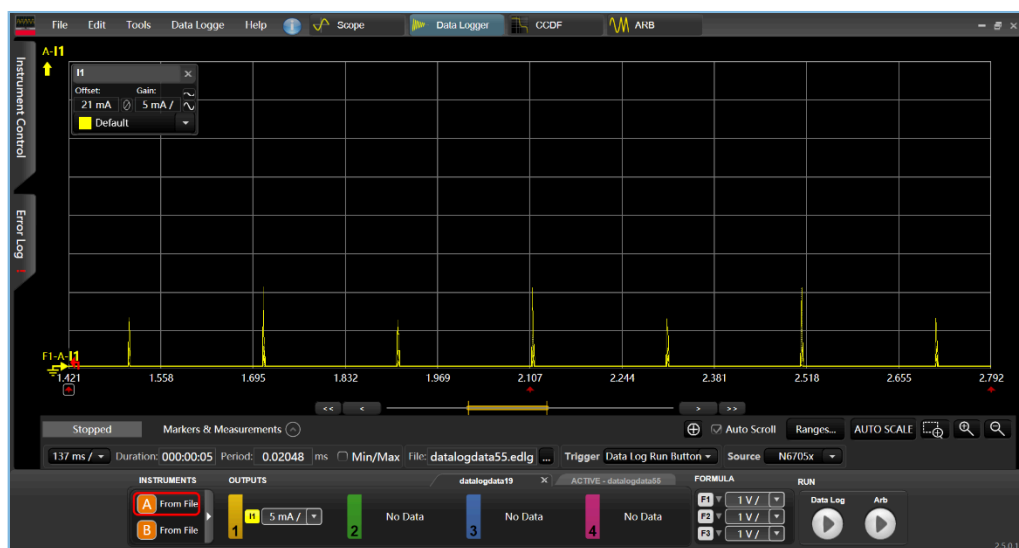


图 3-4 功耗测试场景二

其他场景的功耗测试在此不再赘述，请根据需要进行设置。

## 4 应用详解

本章将介绍PCP示例的运行流程和关键代码。

### 4.1 运行流程

下载PCP示例的开发板上电后，会依次执行外设及电源管理初始化、Bluetooth LE协议栈初始化、PCS初始化等相关操作，主要流程如图 4-1所示：

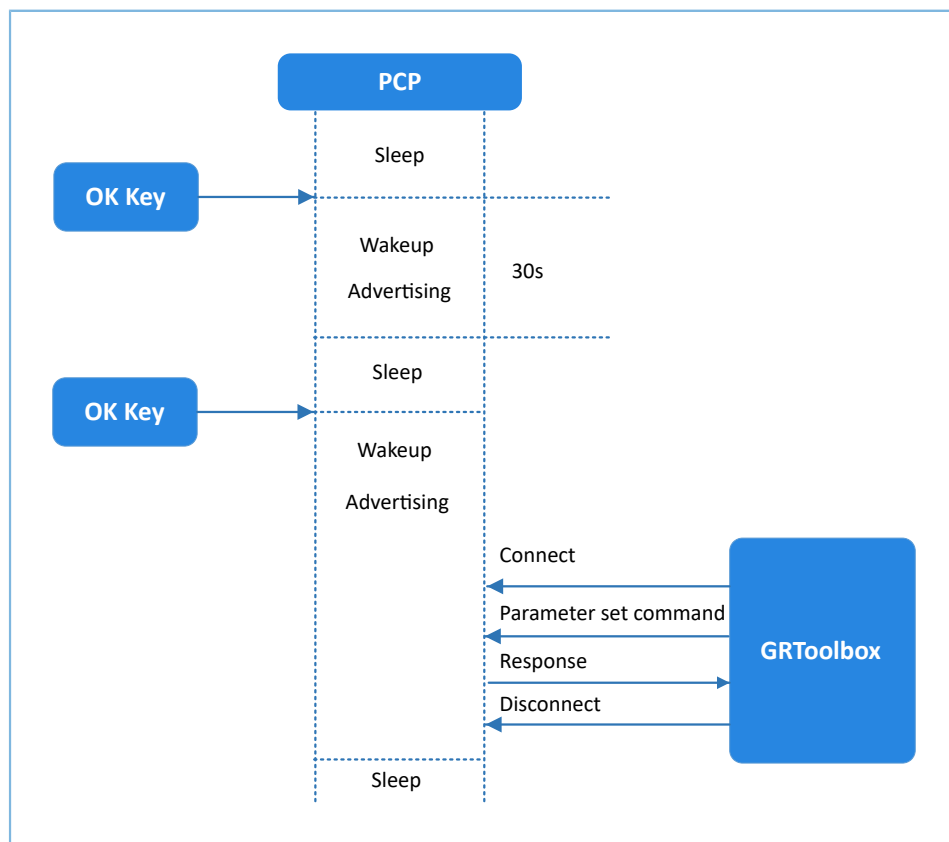


图 4-1 运行流程图

### 4.2 关键代码

下文详细描述了开发板与GRToolbox的交互过程。

#### 4.2.1 电源管理配置

1. 配置芯片电源管理模式为睡眠模式（PMR\_MGMT\_SLEEP\_MODE）及外部唤醒源（板载“OK”键），用于唤醒芯片，开启广播。

路径：工程目录下的user\_platform\user\_periph\_setup.c

名称：wkup\_key\_init(), app\_periph\_init();

```
static void wkup_key_init(void)
{
    ...
}
```

```
s_gpiote_param.pin = APP_KEY_OK_PIN;
...

app_gpiote_init(&s_gpiote_param, 1);
}
```

```
void app_periph_init(void)
{
    ...
    wkup_key_init();
    pwr_mgmt_mode_set(PMR_MGMT_SLEEP_MODE);
}
```

2. 用于判断系统启动后是进入Ultra Deep Sleep，还是进行正常业务逻辑，在main()函数中调用。

路径：工程目录下的user\_platform\user\_periph\_setup.c

名称：is\_enter\_ultra\_deep\_sleep();

```
bool is_enter_ultra_deep_sleep(void)
{
    if (APP_IO_PIN_RESET != app_io_read_pin(APP_IO_TYPE_AON, APP_KEY_OK_PIN))
    {
        return true;
    }

    return false;
}
```

3. main()函数判断系统启动后流程分支，以及低功耗管理。

```
int main(void)
{
    app_periph_init();

    if (is_enter_ultra_deep_sleep())
    {
        pwr_mgmt_ultra_sleep(0);
    }
    // Initialize ble stack.
    ble_stack_init(ble_evt_handler, &heaps_table);
    while (1)
    {
        pwr_mgmt_schedule();
    }
}
```

## 4.2.2 指令解析、执行与回应

当Setting Characteristic Value接收到对端发送的指令数据时，会将事件与相关信息上报至应用层，可利用pcs\_param\_parse()函数对其指令进行解析、执行和回应。本节以设置广播间隔和设置广播数据为例，对功耗测试相关场景设置进行说明，其他设置不再一一赘述。

### 1. 设置广播间隔

设置广播间隔值，然后回应对端设备。该参数值将在下次开启广播时生效。具体代码见pcs\_param\_parse函数中对PCS\_SETTING\_TYPE\_ADV\_INTERVAL事件的处理代码。

路径：工程目录下的user\_app\user\_app.c

名称：pcs\_param\_parse()

```
void pcs_param_parse(uint8_t conn_idx, uint8_t *p_data, uint16_t length)
{
    ...
    switch (p_data[0])
    {
        case PCS_SETTING_TYPE_ADV_INTERVAL:
            s_gap_adv_param.adv_intv_max = BUILD_U16(p_data[1], p_data[2]);
            s_gap_adv_param.adv_intv_min = BUILD_U16(p_data[1], p_data[2]);
            response[0] = PCS_SETTING_TYPE_ADV_INTERVAL;
            response[1] = PCS_SET_PARAM_SUCCESS;
            pcs_setting_reply(0, response, 2);
            break;

            ...

        default:
            break;
    }
}
```

### 2. 设置广播数据

设置广播数据，并根据执行结果回应对端设备。可设置的长度分别为3、10、17、24和31字节。该设置在下次开启广播时生效。具体代码见pcs\_param\_parse函数中对PCS\_SETTING\_TYPE\_ADV\_DATA事件的处理代码。

路径：工程目录下的user\_app\user\_app.c

名称：pcs\_param\_parse()

#### 说明:

通过ble\_gap\_adv\_param\_set()设置广播数据时，由于Advertising Type Flag会占用3个字节的广播数据长度，因此在设置用户广播数据时需减去3个字节有效长度。

```
void pcs_param_parse(uint8_t conn_idx, uint8_t *p_data, uint16_t length)
{
```

```

.....
case PCS_SETTING_TYPE_ADV_DATA:
    response[0] = PCS_SETTING_TYPE_ADV_DATA;
    response[1] = PCS_SET_PARAM_SUCCESS;

    if (PCS_SET_ADV_DATA_3B == p_data[1])
    {
        s_adv_data_set.length = 0; // 3 byte for adv type
    }
    else if (PCS_SET_ADV_DATA_10B == p_data[1])
    {
        memcpy(s_adv_data_set.adv_data, s_adv_data_10b, 7);
        s_adv_data_set.length = 7; // 3 byte for adv type
    }
    .....
    pcs_setting_reply(0, response, 2);
    break;
    .....
}

```

### 4.2.3 开启通知

当对端设备开启通知（即对其CCCD写值“0x0001”），示例应用在收到PCS\_EVT\_TX\_ENABLE事件后开始发送Notify数据；一次数据发送完成后，示例应用收到PCS\_EVT\_DATA\_SENT后再次Notify数据，直到收到PCS\_EVT\_TX\_DISABLE才停止Notify数据。

路径：工程目录下的user\_app\user\_app.c

名称：pcs\_service\_event\_process()

```

static void pcs_service_event_process(pcs_evt_t *p_evt)
{
    switch (p_evt->evt_type)
    {
        case PCS_EVT_TX_ENABLE:
            s_is_notify_enable = true;
            pcs_tx_data_notify();
            break;
        case PCS_EVT_TX_DATA_SENT:
            if (s_is_notify_enable)
            {
                s_notify_counter++;
                pcs_tx_data_notify();
            }
            break;
        .....
        default:
            break;
    }
}

```

}