Simple Central Power Save Application

User Guide

Version 0.2

May 2016

Redpine Signals, Inc.
2107 N. First Street, #540
San Jose, CA 95131.
Tel: (408) 748-3385
Fax: (408) 705-2019
Email: info@redpinesignals.com
Website: www.redpinesignals.com

About this Document

This document describes the process of bringing up the RS9113 based module in BTLE central mode with power save.

Disclaimer:

The information in this document pertains to information related to Redpine Signals, Inc. products. This information is provided as a service to our customers, and may be used for information purposes only. Redpine assumes no liabilities or responsibilities for errors or omissions in this document. This document may be changed at any time at Redpine's sole discretion without any prior notice to anyone. Redpine is not committed to updating this document in the future.

Table of Contents

1 Introduction 5
1.1 Application Overview 5
1.1.1 Overview 5
1.1.2 Sequence of Events 5
1.2 Application Setup 5
1.2.1 SPI based Setup Requirements 5
1.2.2 UART/USB-CDC based Setup Requirements 5
2 Configuration and Execution of the Application 7
2.1 Initializing the Application 7
2.1.1 SPI Interface 7
2.1.2 UART/USB-CDC Interface 7
2.2 Configuring the Application 7
2.3 Executing the Application 9

Table of Figures

Figure 1: Setup diagram 6
Figure 2: Power profile in scan mode 10
Figure 3: Power profile in connected state 11

Table of Tables

No table of figures entries found.

1 Introduction

This project is applicable to all the WiSeConnect variants like WiSeConnect Plus, WiSeMCU and WyzBee. The term WiSeConnect refers to its appropriate variant.

1.1 Application Overview

1.1.1 Overview

This application demonstrates how to configure the device in power save mode when device in scan state and in connected state in BTLE central mode.

1.1.2 Sequence of Events

This Application explains user how to:

- Set local name to the WiSeConnect device
- Configure device in desired power save mode
- Scan for remote BTLE peripheral devices
- Connect to configured remote BTLE slave device
- Analyze power save functionality when WiSeConnect device in scan mode and in connected state using Azilent power analyzer

1.2 Application Setup

The WiSeConnect in its many variants supports SPI and UART interfaces. Depending on the interface used, the required set up is as below:

1.2.1 SPI based Setup Requirements

- Windows PC with CooCox IDE
- Spansion (MB9BF568NBGL) micro controller

Note: If user does not have Spansion (MB9BF568NBGL) host platform, please go through the SPI-Porting guide **\sapis\docs\RS9113-WiSeConnect-SAPI-Porting-Guide-vx.x.pdf** for SAPIs porting to that particular platform.

- WiSeConnect device
- Remote peripheral device (This example uses TI sensor tag for remote peripheral device)

1.2.2 UART/USB-CDC based Setup Requirements

- Windows PC with Dev-C++ IDE
- WiSeConnect device
- Remote peripheral device (This example uses TI sensor tag for remote peripheral device)

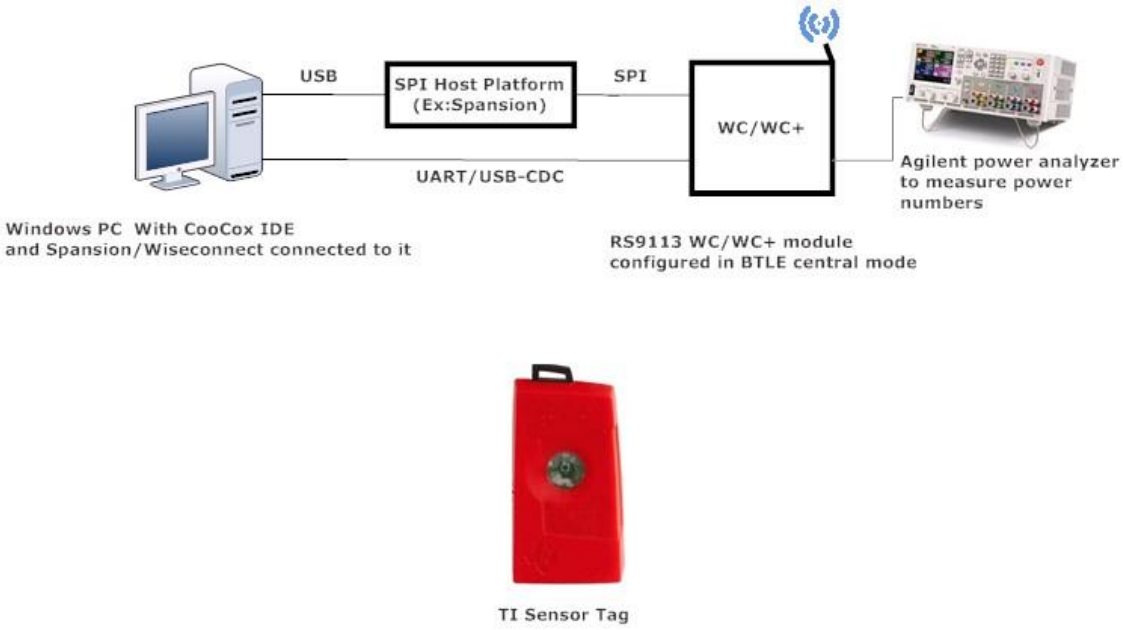


Figure 1: Setup diagram

2 Configuration and Execution of the Application

The example application is available in the Release at **{Release $}/host/*sapis/examples.*** These examples will have to be initialized, configured and executed to test the application. The initialization varies based on the interface but configuration and execution are the common.

2.1 Initializing the Application

2.1.1 SPI Interface

If User using SPI interface, Please refer the document ***sapis/platforms/spansion_MB9BF568NBGL/RS9113-WiSeConnect_SAPIS_Spansion_Project_User_guide.pdf*** for opening the ***simple_central*** example in CooCox IDE.

2.1.2 UART/USB-CDC Interface

If User using UART interface, Please refer the document ***sapis/platforms/windows_uart/RS9113-WiSeConnect_SAPIS_Windows_Project_UserGuide.pdf*** for opening the ***simple_central*** example in Dev-C++ IDE

2.2 Configuring the Application

1. Open ***sapis/examples/ble/ble_power_save/simple_central/rsi_ble_central.c*** file and update/modify following macros :

RSI_BLE_DEV_ADDR_TYPE refers address type of the remote device to connect.

Valid configurations are LE_RANDOM_ADDRESS and LE_PUBLIC_ADDRESS

```
#define RSI_BLE_DEV_ADDR_TYPE          LE_PUBLIC_ADDRESS
```

RSI_BLE_DEV_ADDR refers address of the remote device to connect.

```
#define RSI_BLE_DEV_ADDR_TYPE          "00:1A:7D:DA:71:13"
```

NO_OF_ADV_REPORTS refers Number of advertising reports to be held by application

```
#define NO_OF_ADV_REPORTS          10
```

To Enable Power Save

LE_PS_MODE refers whether power save command issues after scan command or after successful connection with remote peer device.

Update macro with **SCAN_MODE** to issue power save command after scan command or **CONNECTED_MODE** to issue power save command after connectivity success.

```
#define LE_PS_MODE          SCAN_MODE
```

PSP_MODE refers power save profile mode. WiSeConnect device supports following power modes in BTLE,

RSI_ACTIVE (0): In this mode, module is active and power save is disabled.

RSI_SLEEP_MODE_2 (1): This mode is applicable when module is in Advertising state as well as in connected state. In this sleep mode, SoC will go to sleep based on GPIO hand

shake or Message exchange, therefore handshake is required before sending data to the module.

RSI_SLEEP_MODE_8 (8) : In this power mode, module goes to power save when it is in unassociated state with the remote device. In this sleep mode, SoC will go to sleep based on GPIO hand shake or Message exchange, therefore handshake is required before sending the command to the module.

```
#define PSP_MODE                RSI_SLEEP_MODE_2
```

Note1: For **RSI_SLEEP_MODE_2** and **RSI_SLEEP_MODE_8** modes**,** GPIO or Message based hand shake can be selected using **RSI_HAND_SHAKE_TYPE** macro which is defined in ***sapis/include/rsi_wlan_config.h***

Note2: In this example user can verify **RSI_SLEEP_MODE_2** with Message based hand shake. If user wants to verify other power modes, user has to change the application as well as GPIO hand shake signals.

PSP_TYPE refers power save profile type. WiSeConnect device supports following power save profile types in BTLE modedd:

RSI_MAX_PSP (0): In this mode, WiSeConnect device will be in Maximum power save mode. i.e Device will wake up for every DTIM beacon and do data Tx and Rx.

```
#define PSP_TYPE                RSI_MAX_PSP
```

Following are the **non-configurable** macros in the application.

Following are the event numbers for advertising, connection and Disconnection events:

```
#define RSI_APP_EVENT_ADV_REPORT        0
#define RSI_APP_EVENT_CONNECTED         1
#define RSI_APP_EVENT_DISCONNECTED      2
```

BT_GLOBAL_BUFF_LEN refers Number of bytes required by the application and the driver

```
#define BT_GLOBAL_BUFF_LEN          10000
```

2. Open ***sapis/include/rsi_wlan_config.h*** file and update/modify following macros,

```
#define CONCURRENT_MODE             RSI_DISABLE
#define RSI_FEATURE_BIT_MAP         FEAT_SECURITY_OPEN
#define RSI_TCP_IP_BYPASS           RSI_DISABLE
#define RSI_TCP_IP_FEATURE_BIT_MAP TCP_IP_FEAT_DHCPV4_CLIENT

#define  RSI_CUSTOM_FEATURE_BIT_MAP  0
#define  RSI_BAND                     RSI_BAND_2P4GHZ
```

RSI_HAND_SHAKE_TYPE is used to select GPIO or Message based hand shake in **RSI_SLEEP_MODE_2** and **RSI_SLEEP_MODE_8** modes**.**

```
#define RSI_HAND_SHAKE_TYPE          MSG_BASED
```

RSI_SELECT_LP_OR_ULP_MODE is used to select low power mode or ultra low power mode. Valid configurations are**, RSI_LP_MODE** or **RSI_ULP_WITH_RAM_RET** or **RSI_ULP_WITHOUT_RAM_RET**

RSI_LP_MODE: This module will be in Low power mode.

RSI_ULP_WITH_RAM_RET: This module will be in Ultra low power mode and it will remember the previous state after issuing power save mode command.

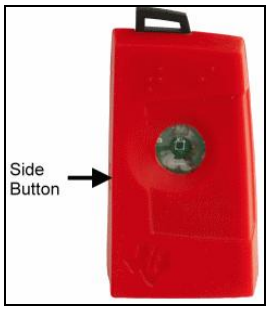
RSI_ULP_WITHOUT_RAM_RET: This module will be in Ultra low power mode and it will not remember the previous state after issuing power save mode command. After wakeup, module will give CARD READY indication and user has to issue commands from wireless initialization.

```
#define RSI_SELECT_LP_OR_ULP_MODE    RSI_ULP_WITH_RAM_RET
```

2.3 Executing the Application

1. Configure the remote TI sensor tag in peripheral mode and put it in advertising mode.

   Press the side button to configure TI sensor tag in advertising mode.



2. **SPI Interface**

   If User using SPI interface, Please refer the document ***sapis/platforms/spansion_MB9BF568NBGL/RS9113-WiSeConnect_SAPIS_Spansion_Project_User_guide.pdf*** for executing the ***simple_central*** example in CooCox IDE.

3. **UART/USB-CDC Interface**

   If User using UART interface, Please refer the document ***sapis/platforms/windows_uart/RS9113-WiSeConnect_SAPIS_Windows_Project_UserGuide.pdf*** for executing the ***simple_central*** example in Dev-C++ IDE

4. After the program gets executed, WiSeConnect module will be in scanning state with configure power save profile. Please refer the given below image for power save cycle in scan mode.

Figure 2: Power profile in scan mode

Note: Default configuration of scanning interval is 160ms. So, WiSeConnect device will wakes up for every 150ms and goes back to sleep after scan complete.

5. After successful scan, WiSeConnect device connects to remote device which is configured in **`RSI_BLE_DEV_ADDR`** macro.
6. After successful connection, device goes to sleep and wakes up for every connection interval, refer the given below image for power cycle after successful connection.

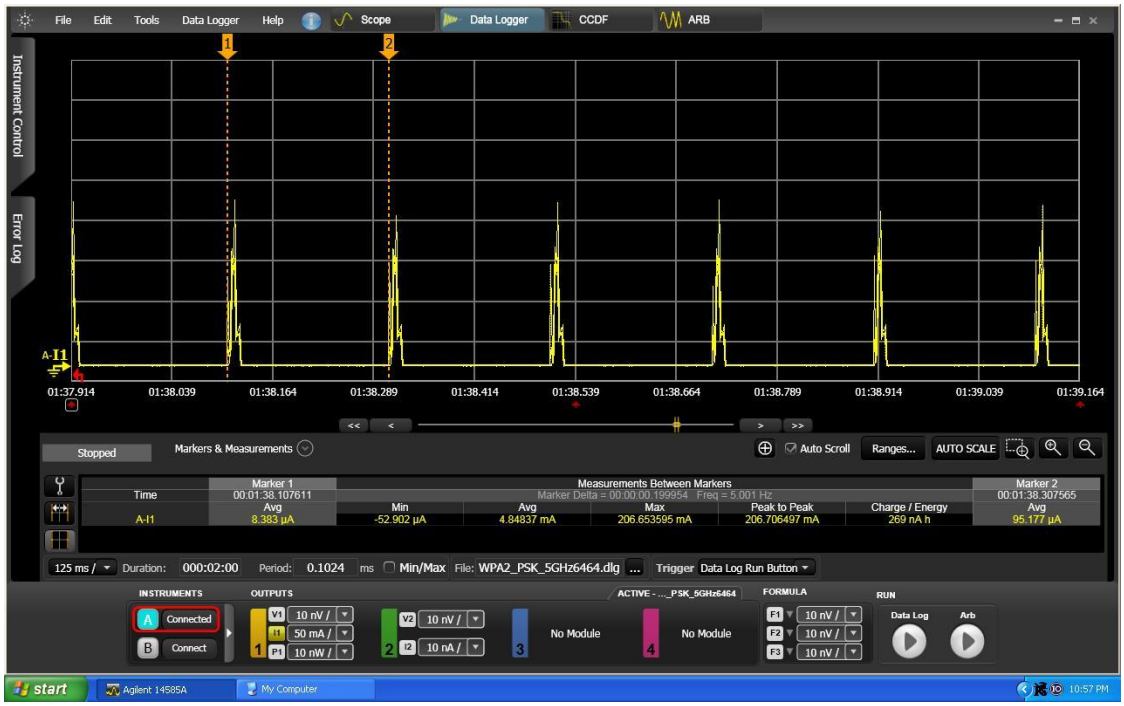


Figure 3: Power profile in connected state

Note1: Default connection interval of WiSeConnect device is 200ms. So, WiSeConnect module wakes up for every 200ms and goes back to sleep.

Note2: Above power save profile image capture when it is in idle state after successful connection. So, user may not get same profile as shown above image. It will vary based on the traffic.