Simple Peripheral 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 peripheral mode with power save.

Disclaimer:

Table of Contents

Table of Figures

Table of Tables

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 that how to configure device in power save in Advertising mode and in connected mode in simple BLE peripheral mode.

1.1.2 Sequence of Events

This Application explains user how to:

- Set a local name to the device
- Configure the module in power save mode
- Configure the device to advertise
- Connect from remote Master device
- Analyze power save functionality when WiSeConnect device in Advertise 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
- Smart phone (Android)/tablet with LE application.

Ex: Install Light blue App for tablet for ipad mini and BLE scanner app for android smart phone**.**

- Agilent power analyzer

1.2.2 UART/USB-CDC based Setup Requirements

- Windows PC with Dev-C++ IDE
- WiSeConnect device
- Smart phone (Android)/tablet with LE application.

Ex: Install Light blue App for tablet for ipad mini and BLE scanner app for android smart phone**.**

- Agilent power analyzer

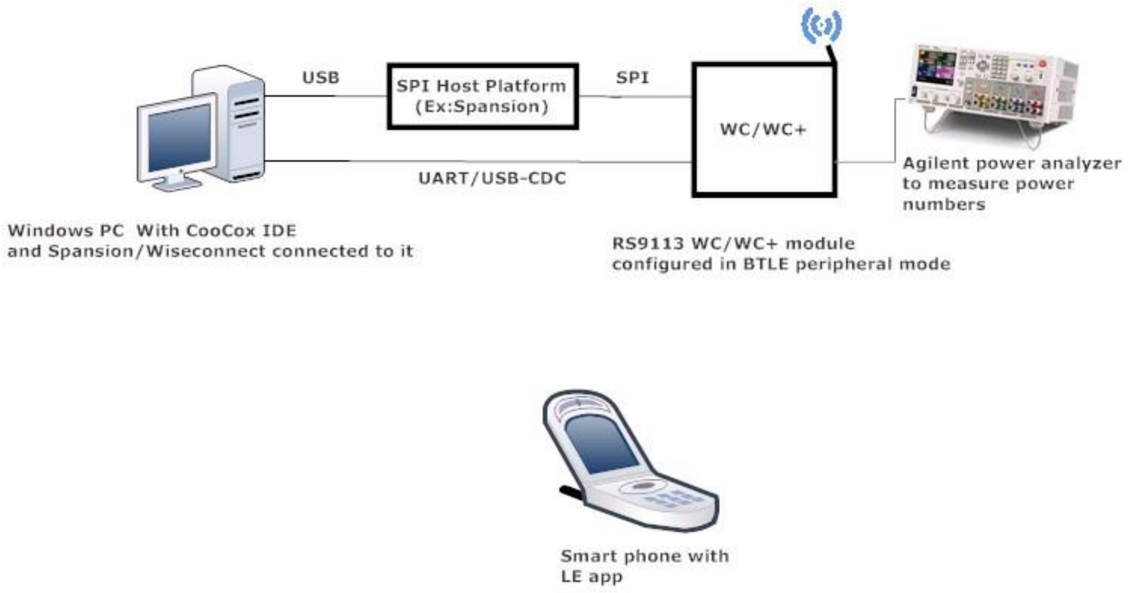

USB
SPI Host Platform (Ex:Spansion)
SPI
WC/WC+
UART/USB-CDC
Agilent power analyzer to measure power numbers
Windows PC With CooCox IDE and Spansion/Wiseconnect connected to it
RS9113 WC/WC+ module configured in BTLE peripheral mode
Smart phone with LE app

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_peripheral*** 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_peripheral*** example in Dev-C++ IDE

2.2 Configuring the Application

1. Open **sapis/examples/ble/ble_power_save/simple_peripheral/rsi_ble_peripheral.c** file and update/modify following macros

RSI_BLE_LOCAL_ANME refers name of the WiSeConnect device to appear during scanning by remote devices.

```
#define RSI_BLE_LOCAL_NAME        "WLAN_BLE_SIMPLE"
```

RSI_SEL_ANTENNA refers antenna to be used by WiSeConnect module.

If user using internal antenna then set,

```
#define RSI_SEL_ANTENNA       RSI_SEL_INTERNAL_ANTENNA
```

If user using external antenna (U.FL connector) then set,

```
#define RSI_SEL_ANTENNA       RSI_SEL_EXTERNAL_ANTENNA
```

To Enable Power Save

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, the 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 mode,

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_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`: In this, module will be in Low power mode.

`RSI_ULP_WITH_RAM_RET:` In 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:` In 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

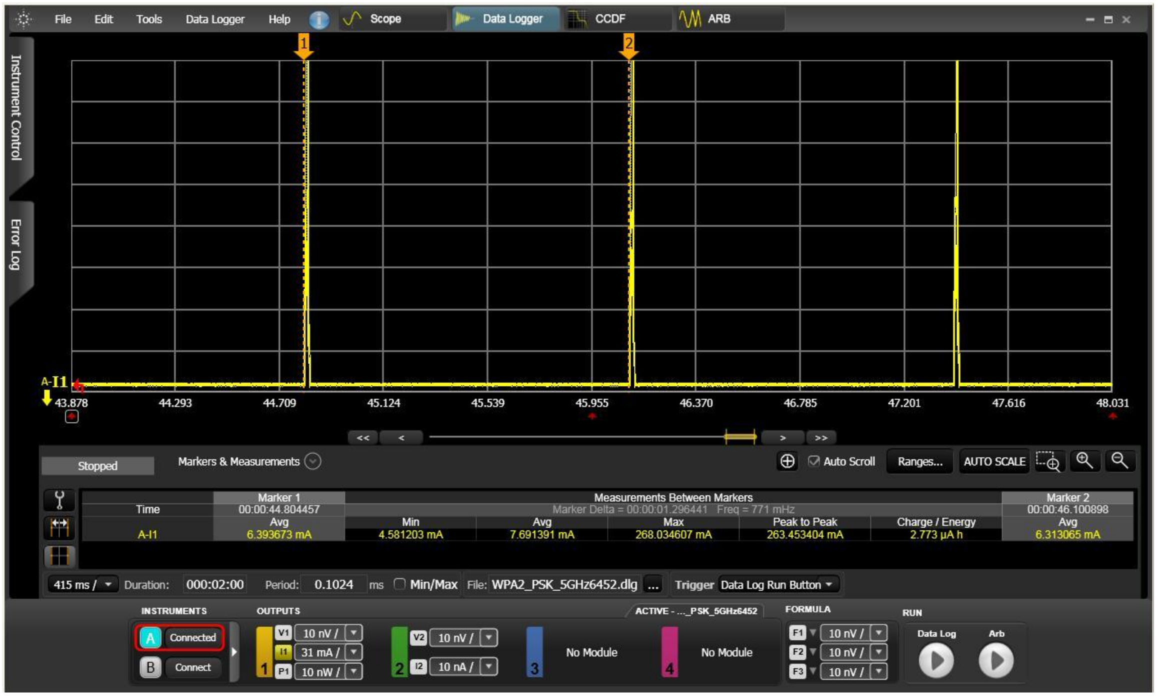
1. **SPI Interface**

   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_peripheral*** example in CooCox IDE.

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 executing the ***simple_peripheral*** example in Dev-C++ IDE

3. After the program gets executed, WiSeConnect module would be in Advertising state with configured power save profile.
4. WiSeConnect device will go to sleep and wakes up for every advertising interval and goes back to sleep after advertising. Please refer the given below image for power save cycle in advertising mode.



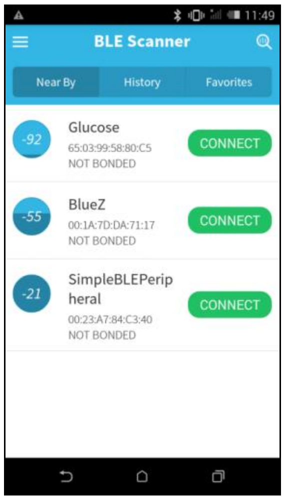
Time	Marker 1	Measurements Between Markers					Marker 2
	00:00:44.804457	Marker Delta = 00:00:01.296441 Freq = 771 mHz					00:00:46.100898
	Avg	Min	Avg	Max	Peak to Peak	Charge / Energy	Avg
A-I1	6.393673 mA	4.581203 mA	7.691391 mA	268.034607 mA	263.453404 mA	2.773 µA h	6.313065 mA

415 ms / Duration: 000:02:00 Period: 0.1024 ms Min/Max File: WPA2_PSK_5GHz6452.dlg ... Trigger Data Log Run Button

Figure 2: Power profile in advertising mode

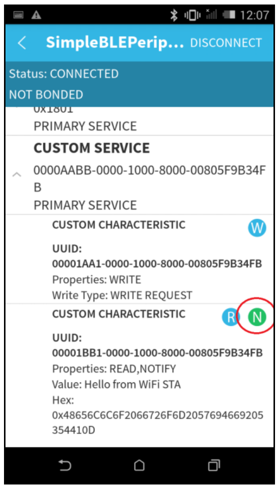
> **Note:** Default configuration of advertising interval is 1.2 sec. So, WiSeConnect device will wakes up for every 1.2 sec and goes back to sleep after advertise.

5. Open a LE App in the Smartphone and do Scan.
6. In the App, WiSeConnect module device will appear with the name configured in the macro **`RSI_BLE_LOCAL_NAME (Ex: "WLAN_BLE_SIMPLE")`** or sometimes observed as WiSeConenct device as internal name "**`SimpleBLEPeripheral`**".



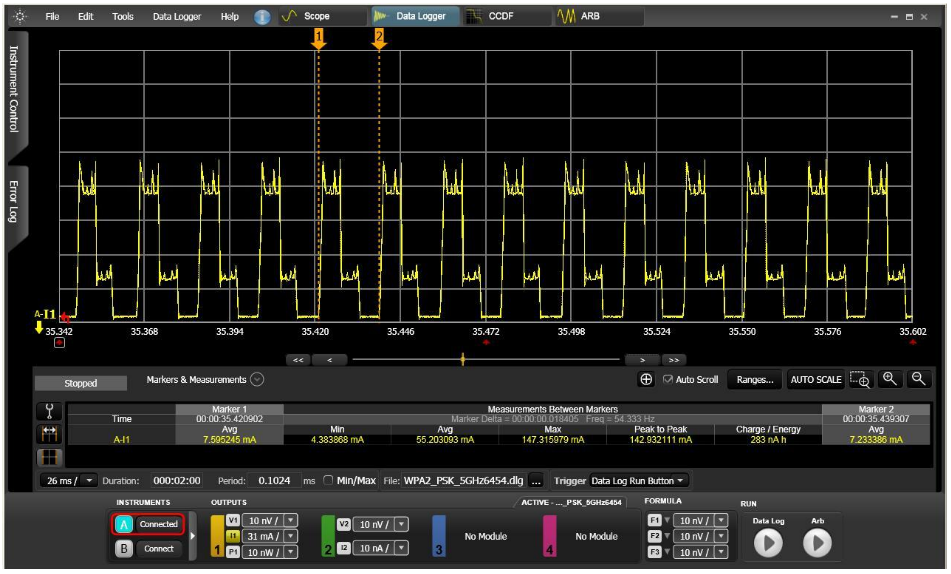
-92	Glucose 65:03:99:58:80:C5 NOT BONDED	CONNECT
-55	BlueZ 00:1A:7D:DA:71:17 NOT BONDED	CONNECT
-21	SimpleBLEPeripheral 00:23:A7:84:C3:40 NOT BONDED	CONNECT

7. Initiate connection from the mobile App.
8. After successful connection, User can see the connected state in BLE scanner app and also check the supported services by WiSeConnect device.



Status: CONNECTED
NOT BONDED
0x1801
PRIMARY SERVICE
CUSTOM SERVICE
0000AABB-0000-1000-8000-00805F9B34FB
PRIMARY SERVICE
CUSTOM CHARACTERISTIC
UUID:
00001AA1-0000-1000-8000-00805F9B34FB
Properties: WRITE
Write Type: WRITE REQUEST
CUSTOM CHARACTERISTIC
UUID:
00001BB1-0000-1000-8000-00805F9B34FB
Properties: READ,NOTIFY
Value: Hello from WiFi STA
Hex:
0x48656C6C6F2066726F6D2057694669205354410D

9. After successful connection, Module goes to sleep and wakes up for every connection interval. Please check the given below image for power save cycle after connection.



	Marker 1	Measurements Between Markers					Marker 2
Time	00:00:35.420902	Marker Delta = 00:00:00.018405 Freq = 54.333 Hz					00:00:35.439307
	Avg	Min	Avg	Max	Peak to Peak	Charge / Energy	Avg
A-I1	7.595245 mA	4.383868 mA	55.203093 mA	147.315979 mA	142.932111 mA	283 nA h	7.233386 mA

26 ms / Duration: 000:02:00 Period: 0.1024 ms Min/Max File: WPA2_PSK_5GHz6454.dlg Trigger Data Log Run Button

Figure 3: Power profile in connected state

Note1: Default configuration of connection interval of Master device (smart phone) is 18ms. So, WiSeConnect device will wakes up for every 18ms sec and goes back to sleep after advertise.

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.