iBeacon Application

User guide

Version 0.1

September 2015

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 Advertising mode with iBeacon data.

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

Table of Figures

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

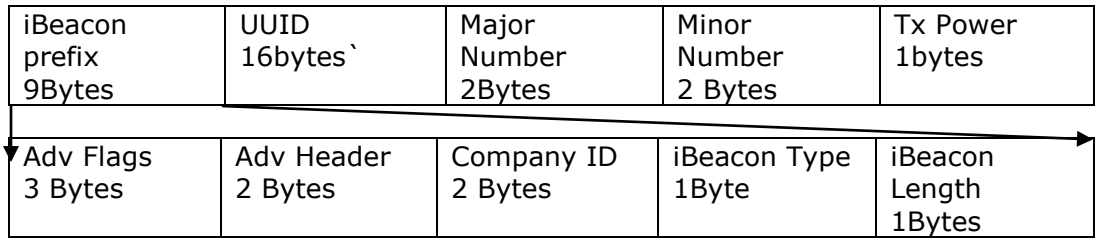
This application demonstrates how to set the iBeacon data format in advertising parameters in simple BLE peripheral mode.

1.1.2 Sequence of Events

This Application explains user how to:

- Set a local name to the device
- Set the iBeacon data to be advertised in WiSeConnect module
- Configure the device in Advertise mode
- Scan from remote Master device

1.1.3 iBeacon Advertise data format



iBeacon Prefix:

Vendor specific fixed value.

Default iBeacon prefix values setting by application is,

Prefix = {0x02, 0x01, 0x02, 0x1A, 0xFF, 0x4C, 0x00, 0x02, 0x15}

UUID:

User generated proximity UUID.

Remote devices recognize which beacon they approach on the basis of UUID, major and minor numbers.

Default UUID, Major and Minor values setting by application is,

UUID = {0xFB , 0x0B , 0x57 , 0xA2 , 0x82 , 0x28 , 0x44 , 0xCD , 0x91 ,

0x3A , 0x94 , 0xA1 , 0x22 , 0xBA , 0x12 , 0x06}

major_num = {0x11, 0x22}

minor_num = {0x33, 0x44}

Tx power is used to calculate distance from iBeacon.

Default Tx power value setting by application is,

Tx Power = 0x33

Note: If the user wants to change the prefix, UUID, Major number, Minor number and Tx Power values, Please change the following values in ***sapis/examples/ble/ibeacon/rsi_ble_ibeacon.c*** file.

For Prefix:

uint8_t adv[31] = {0x02, 0x01, 0x02, 0x1A, 0xFF, 0x4C, 0x00, 0x02, 0x15}; //prefix(9bytes)

For UUID:

uint8_t uuid[16] = {0xFB , 0x0B , 0x57 , 0xA2 , 0x82 , 0x28 , 0x44 , 0xCD , 0x91 , 0x3A , 0x94 , 0xA1 , 0x22 , 0xBA , 0x12 , 0x06};

For Major Number:

uint8_t major_num[2] = {0x11, 0x22};

For Major Number:

uint8_t minor_num[2] = {0x33, 0x44};

For Tx Power:

uint8_t tx_power = 0x33;

```
86  * This function is used to test the BLE peripheral role and simple GAP API's.
87  */
88 int32_t rsi_ble_ibeacon(void)
89 {
90     int32_t status = 0;
91     int32_t temp_event_map = 0;
92     uint8_t remote_dev_addr[18] = {0};
93     uint8_t adv[31] =  {0x02, 0x01, 0x02, 0x1A, 0xFF, 0x4C, 0x00, 0x02, 0x15}; //prefix(9
94     uint8_t uuid[16] = {0xFB , 0x0B , 0x57 , 0xA2 , 0x82 , 0x28 , 0x44 , 0xCD , 0x91 , 0x
95   uint8_t major_num[2] = {0x11, 0x22};
96   uint8_t minor_num[2] = {0x33, 0x44};
97   uint8_t tx_power = 0x33;
98
```

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 with iBeaconDetector application.

Note: Install iBeaconDetector app for android smart phone.

1.2.2 UART/USB-CDC based Setup Requirements

- Windows PC with Dev-C++ IDE
- WiSeConnect device
- Smart phone with iBeaconDetector application

Note: Install iBeaconDetector app for android smart phone.

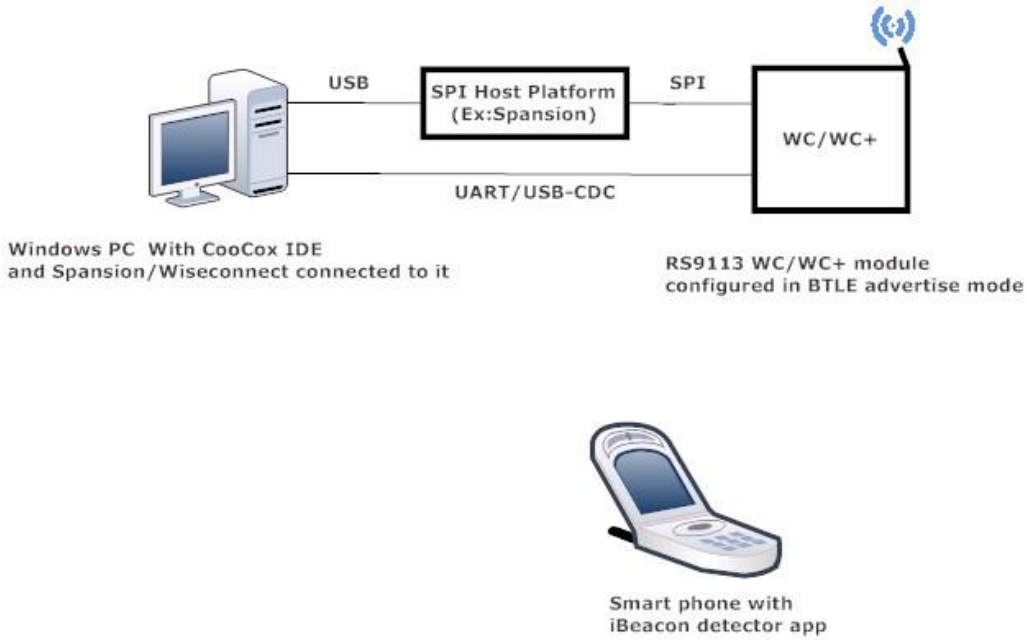


Figure 1: Setup diagram

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

2.2 Configuring the Application

1. Open *sapis/examples/ble/ibeacon/ rsi_ble_ibeacon.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        "ibeacon"
```

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

```
#define RSI_APP_EVENT_CONNECTED           1
#define RSI_APP_EVENT_DISCONNECTED        2
```

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

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
```

2.3 Executing the Application

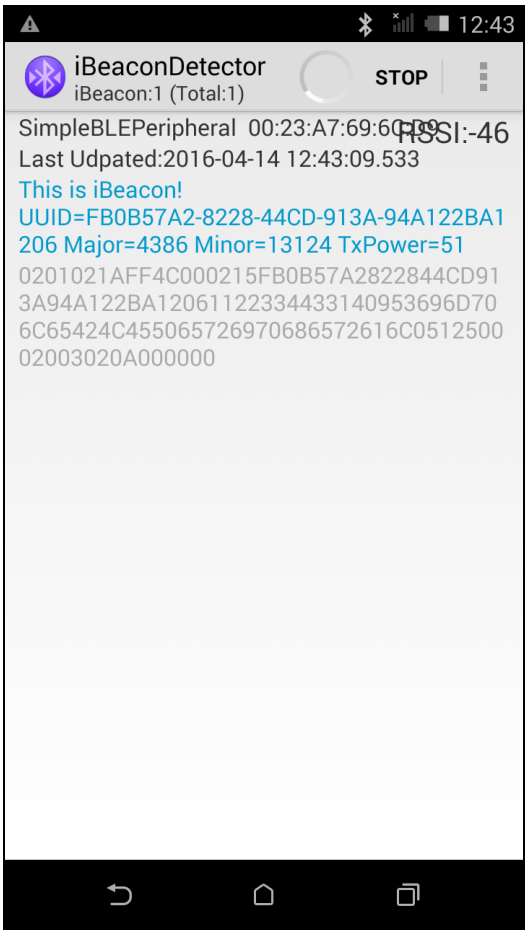
1. **SPI Interface**

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

3. After the program gets executed, WiSeConnect module would be in Advertising state.
4. Open iBeaconDetector app in the Smartphone and do Scan.
5. In the App, WiSeConnect module device would appear with the name configured in the macro **`RSI_BLE_LOCAL_NAME (Ex: "ibeacon")`** or sometimes observed as "**`SimpleBLEPeripheral`**".
6. After successful scan, User can see the WiSeConnect device advertised data i.e UUID, Maximum Number, Minimu Number and Tx Power in iBeaconDetector application.


12:43
iBeaconDetector
iBeacon:1 (Total:1)
STOP
SimpleBLEPeripheral 00:23:A7:69:6C:89
RSSI:-46
Last Udpated:2016-04-14 12:43:09.533
This is iBeacon!
UUID=FB0B57A2-8228-44CD-913A-94A122BA1
206 Major=4386 Minor=13124 TxPower=51
0201021AFF4C000215FB0B57A2822844CD91
3A94A122BA120611223344331409536 96D70
6C65424C455065726970686572616C0512500
02003020A000000