Provisioning 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 as a provisioning application.

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

The provisioning application demonstrates how to provide provisioning to connect desired Access Point using the http server, mdns functionality of WiSeConnect device.

In this application, WiSeConnect device starts as an Access point. After successful creation of AP, application initializes and registers the MDNS service. And application waits for the input to connect with the desired Access Point in station mode. Now, User can open provisioning page by connecting to HTTP server running in device and do scan and select the required Access point to connect and submit the configuration. Once configuration submits, Application receives the configuration set by user and restarts the device in station mode and connects to the desired Access Point.

1.1.2 Sequence of Events

This Application explains user how to:

- Start WiSeConnect device as Access Point after power up.
- Connect a station to the device and get IP address through DHCP.
- Open provisioning page of the Device from the browser of the connected station (STA).
- Click on scan button, this updates scan results in scan results table.
- Select the required Access point (AP) settings in the scan results table and submit the provisioning page.
- WiSeConnect Device would restart as WiFi station and join to the configured Access point (AP).

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
- Wireless Access Point

1.2.2 UART/USB-CDC based Setup Requirements

- Windows PC with Dev-C++ IDE

- WiSeConnect device
- Smart Phone
- Wireless Access Point

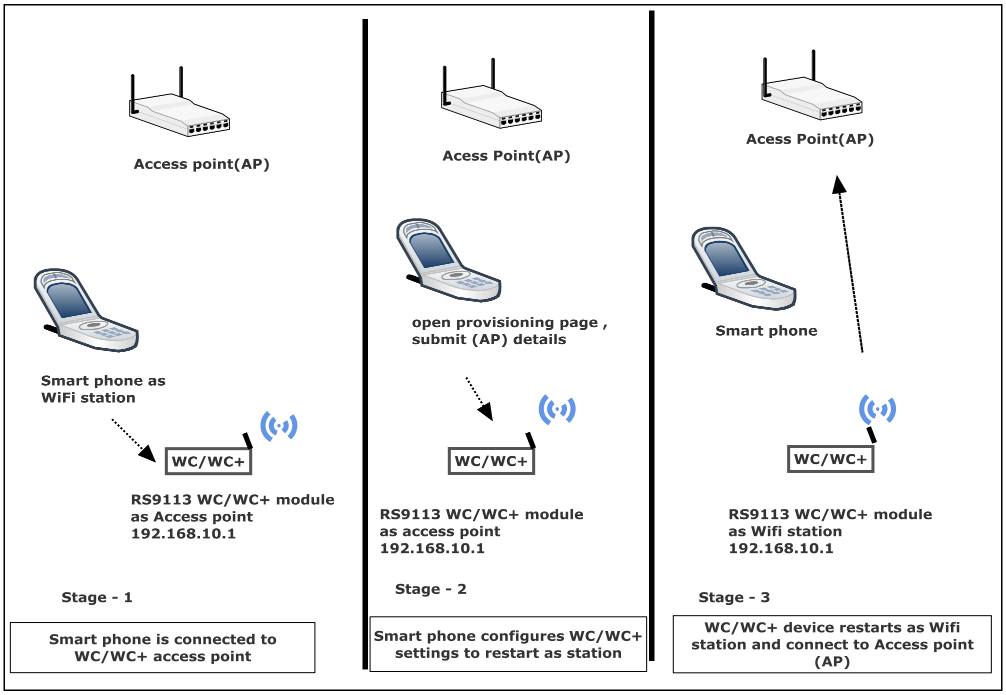


Figure 1:Setup Diagram

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

2.2 Configuring the Application

1. Open ***sapis/examples/provisioning/rsi_ provisioning_app.c*** file and update/modify following macros,

SSID refers to the name of the Access point.

```
      #define SSID                   "<ap name>"
```

CHANNEL_NO refers to the channel in which AP would be started

```
      #define CHANNEL_NO             11
```

Note:
Valid values for **CHANNEL_NO** are 1 to 11 in 2.4GHz band and 36 to 48 & 149 to 165 in 5GHz band. In this example default configured band is 2.4GHz. So, if user wants to use 5GHz band then the user has to set **RSI_BAND** macro to 5GHz band in **sapis/include/rsi_wlan_config.h** file

SECURITY_TYPE refers to the type of security .Access point supports Open, WPA, WPA2 securities.

Valid configuration is:

RSI_OPEN - For OPEN security mode

RSI_WPA - For WPA security mode

RSI_WPA2 - For WPA2 security mode

```
      #define SECURITY_TYPE          RSI_WPA2
```

ENCRYPTION_TYPE refers to the type of Encryption method .Access point supports OPEN, TKIP, CCMP methods.

Valid configuration is:

RSI_CCMP - For CCMP encryption

RSI_TKIP - For TKIP encryption

RSI_NONE - For open encryption

```
      #define ENCRYPTION_TYPE     RSI_CCMP
```

PSK refers to the secret key if the Access point to be configured in WPA/WPA2 security modes.

```
      #define PSK                       “1234567890”
```

BEACON_INTERVAL refers to the time delay between two consecutive beacons in milliseconds. Allowed values are integers from 100 to 1000 which are multiples of 100.

```
      #define BEACON_INTERVAL          100
```

DTIM_INTERVAL refers DTIM interval of the Access Point. Allowed values are from 1 to 255.

```
      #define DTIM_INTERVAL            4
```

To configure MDNSD parameters with desired values

To configure the MDNS IP version

Configure **MDNSD_IP_VERSION** to 4 for IPv4 or 6 for IPv6

```
      #define MDNSD_IP_VERSION      4
```

To configure the MDNS time to live in seconds

```
            #define MDNSD_INIT_TTL       300
```

To configure the time of the added service to live in seconds

```
            #define MDNSD_SERVICE_TTL    300
```

MDNS service port number to be used

```
            #define MDNSD_SERVICE_PORT    80
```

User can add multiple services, but this example is designed to add only one service (HTTP). So more services is set to zero.

To add multiple services user should call rsi_mdnsd_register_service with multiple times, each time setting service more parameter to 1, and setting service_more parameter to 0 for the last service.

```
            #define MDNSD_SERVICE_MORE    0
```

Configure MDNS host name

```
            #define MDNSD_HOST_NAME      "wiseconnect.local."
```

Configure MDNS service pointer name

```
            #define MDNSD_POINTER_NAME   “_http._tcp.local.”
```

Configure MDNS service name

```
#define MDNSD_SERVICE_NAME      "wiseconnect._http._tcp.local"
```

Configure MDNS service text filed

```
            #define MDNSD_SERVICE_TEXT   "text_field"
```

FILE_NAME refers File name of the webpage and Json object.

```
            #define FILE_NAME       "provisioning.html"
```

To configure IP address

IP address to be configured in the device should be in long format and in little endian byte order.

Example: To configure "192.168.10.1" as IP address, update the macro **DEVICE_IP** as 0x010AA8C0.

```
      #define  DEVICE_IP          0X010AA8C0
```

IP address of the gateway should also be in long format and in little endian byte order

Example: To configure "192.168.10.1" as Gateway, update the macro GATEWAY as 0x010AA8C0

```
      #define  GATEWAY            0x010AA8C0
```

IP address of the network mask should also be in long format and in little endian byte order

Example: To configure "255.255.255.0" as network mask, update the macro **NETMASK** as 0x00FFFFFF

```
      #define   NETMASK            0x00FFFFFF
```

Note: In AP mode, configure same IP address for both **DEVICE_IP** and **GATEWAY** macros

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

```
#define  CONCURRENT_MODE             DISABLE
#define  RSI_FEATURE_BIT_MAP         FEAT_SECURITY_PSK
#define  RSI_TCP_IP_BYPASS           DISABLE
#define RSI_TCP_IP_FEATURE_BIT_MAP  (TCP_IP_FEAT_DHCPV4_SERVER
                                    | TCP_IP_FEAT_DHCPV4_CLIENT
                                    | TCP_IP_FEAT_MDNSD
                                    | TCP_IP_FEAT_HTTP_SERVER)
 #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 ***provisioning*** 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 ***provisioning*** example in Dev-C++ IDE

3. After the program gets executed, WiSeConnect Device will be started as Access point having the configuration same as that of in the application.
4. Now connect a Smart phone (STA) to Device and get IP address.
5. After successful connection open the provisioning page from STA browser by giving the following URL:

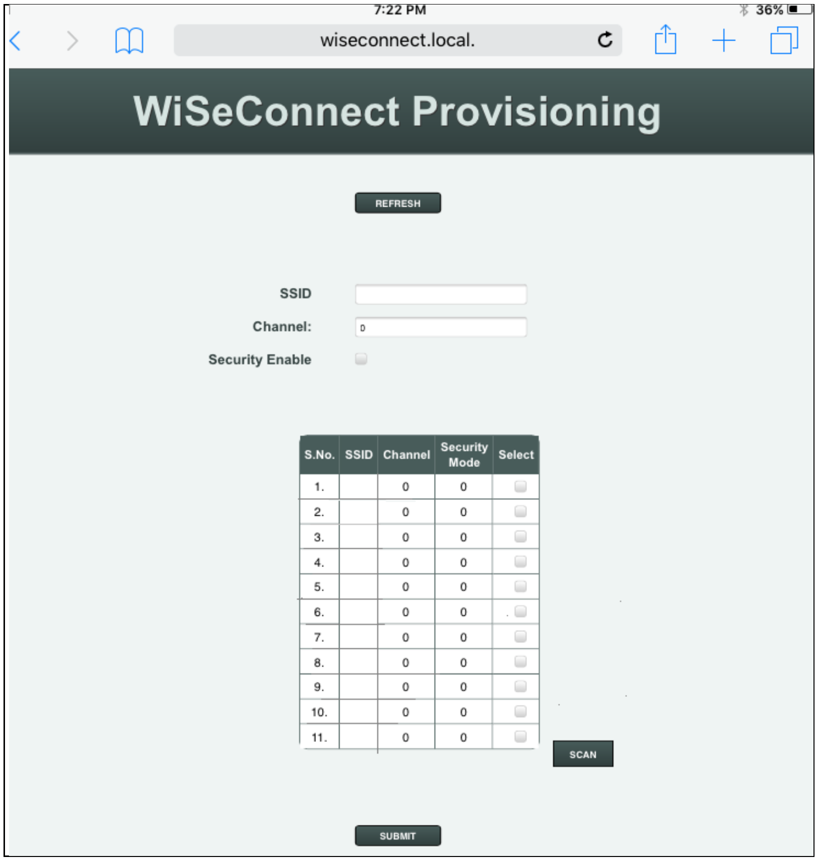
URl: wiseconnect.local/provisioning.html

> **Note:** WiSeConnect device responds to the mdns requests destined to 5353 port only. So ensure that the WiFi station do request for the URL to the Device on 5353 port.

If webpage not opened with the above URL, Use following URL to open provisioning page.

URl: DEVICE_IP/provisioning.html (i.e 192.168.10.1/ provisioning.html)

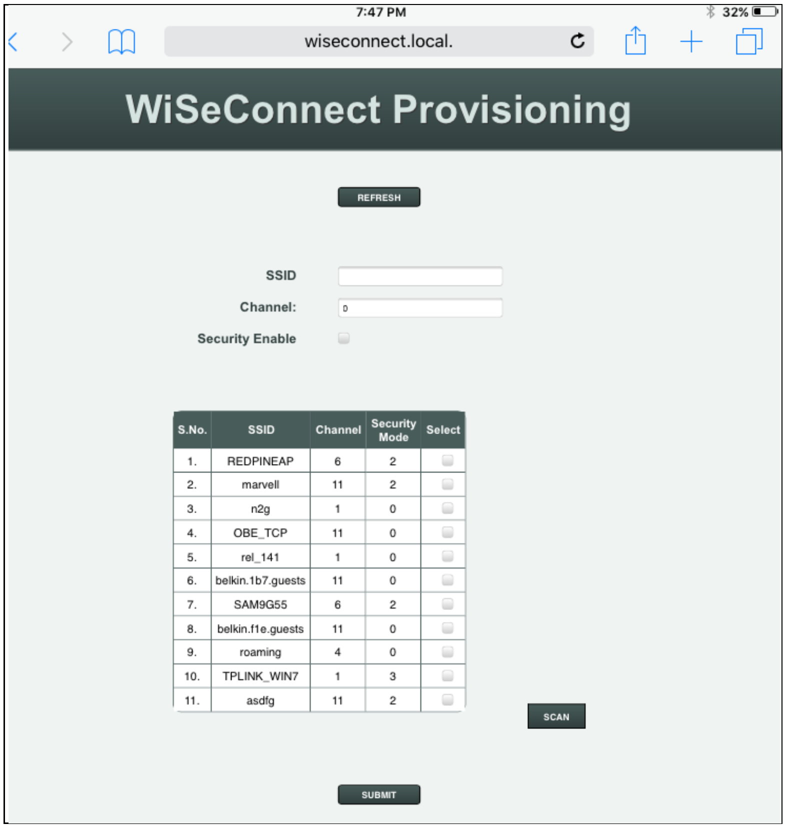
6. Provisioning page is displayed in the browser. Please refer the given below image for provisioning page,



SSID
Channel: 0
Security Enable □

S.No.	SSID	Channel	Security Mode	Select
1.		0	0	□
2.		0	0	□
3.		0	0	□
4.		0	0	□
5.		0	0	□
6.		0	0	□
7.		0	0	□
8.		0	0	□
9.		0	0	□
10.		0	0	□
11.		0	0	□

7. Click on scan button, scan list table will be updated after a **10 seconds** timeout.



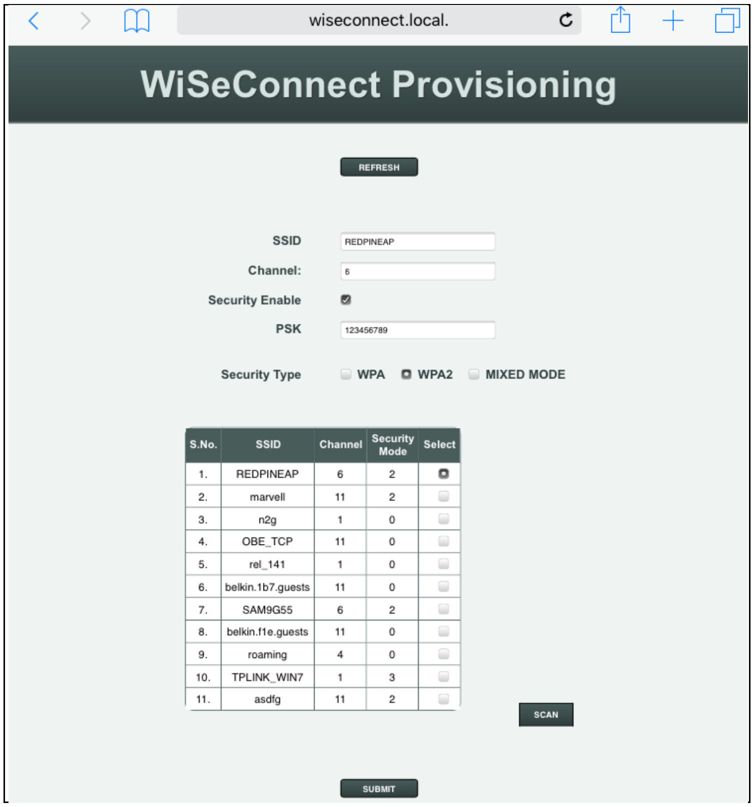
SSID

Channel: 0

Security Enable □

S.No.	SSID	Channel	Security Mode	Select
1.	REDPINEAP	6	2	□
2.	marvell	11	2	□
3.	n2g	1	0	□
4.	OBE_TCP	11	0	□
5.	rel_141	1	0	□
6.	belkin.1b7.guests	11	0	□
7.	SAM9G55	6	2	□
8.	belkin.f1e.guests	11	0	□
9.	roaming	4	0	□
10.	TPLINK_WIN7	1	3	□
11.	asdfg	11	2	□

8. Select the required Access point (AP) to which user wants to connect and provide the required PSK in case of AP secured and submit the configuration.



SSID: REDPINEAP

Channel: 6

Security Enable ☑

PSK: 123456789

Security Type ○ WPA ● WPA2 ○ MIXED MODE

S.No.	SSID	Channel	Security Mode	Select
1.	REDPINEAP	6	2	●
2.	marvell	11	2	○
3.	n2g	1	0	○
4.	OBE_TCP	11	0	○
5.	rel_141	1	0	○
6.	belkin.1b7.guests	11	0	○
7.	SAM9G55	6	2	○
8.	belkin.f1e.guests	11	0	○
9.	roaming	4	0	○
10.	TPLINK_WIN7	1	3	○
11.	asdfg	11	2	○

9. WiSeConnect device will restart as WiFi Client and automatically will connect to the selected Access Point.