

Multicast 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 WiFi station and used for Multicast data traffic.

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.

Copyright © 2015 Redpine Signals, Inc. All rights reserved.

Table of Contents

1	Introduction	4
1.1	Protocol Overview	4
1.2	Application Overview	4
1.2.1	Overview	4
1.2.2	Sequence of Events	4
1.3	Application Setup	4
1.3.1	SPI based Setup Requirements	4
1.3.2	UART/USB-CDC based Setup Requirements	5
2	Configuration and Execution of the Application	6
2.1	Initializing the Application	6
2.1.1	SPI Interface	6
2.1.2	UART/USB-CDC Interface	6
2.2	Configuring the Application	6
2.3	Executing the Application	8

Table of Figures

Figure 1: Setup Diagram	5
-------------------------------	---

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 Protocol Overview

In Networking, Multicast IP Routing protocols are used to distribute data (for example, audio/video streaming broadcasts) to multiple recipients. Using multicast, a source can send a single copy of data to a single multicast address, which is then distributed to an entire group of recipients.

A multicast group identifies a set of recipients that are interested in a particular data stream, and is represented by an IP address from a well-defined range. Data sent to this IP address is forwarded to all members of the multicast group.

Routers between the source and recipients duplicate data packets and forward multiple copies wherever the path to recipients diverges. Group membership information is used to calculate the best routers at which to duplicate the packets in the data stream to optimize the use of the network.

1.2 Application Overview

1.2.1 Overview

This application demonstrates how to add WiSeConnect device to a multicast group and how to send and receive multicast data on a UDP socket.

In this application, WiSeConnect device connects to WiFi Access point and opens UDP socket and joins to a Multicast group ID. After successful join, application sends data to multicast group ID and receives data from Multicast group ID using opened UDP socket.

1.2.2 Sequence of Events

This Application explains user how to:

- Configure WiSeConnect device as a WiFi station
- Connect to WiFi Access Point
- Open UDP socket
- Join multicast group ID
- Send UDP data to multicast group ID
- Receive UDP data coming from Multicast group

1.3 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:

1.3.1 SPI based Setup Requirements

- Windows PC1 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
- Wlan Access point
- Windows PC2 with iperf to send and receive Multicast data

1.3.2 UART/USB-CDC based Setup Requirements

- Windows PC1 with Dev-C++ IDE
- WiSeConnect device
- Wlan Access point
- Windows PC2 with iperf to send and receive Multicast data

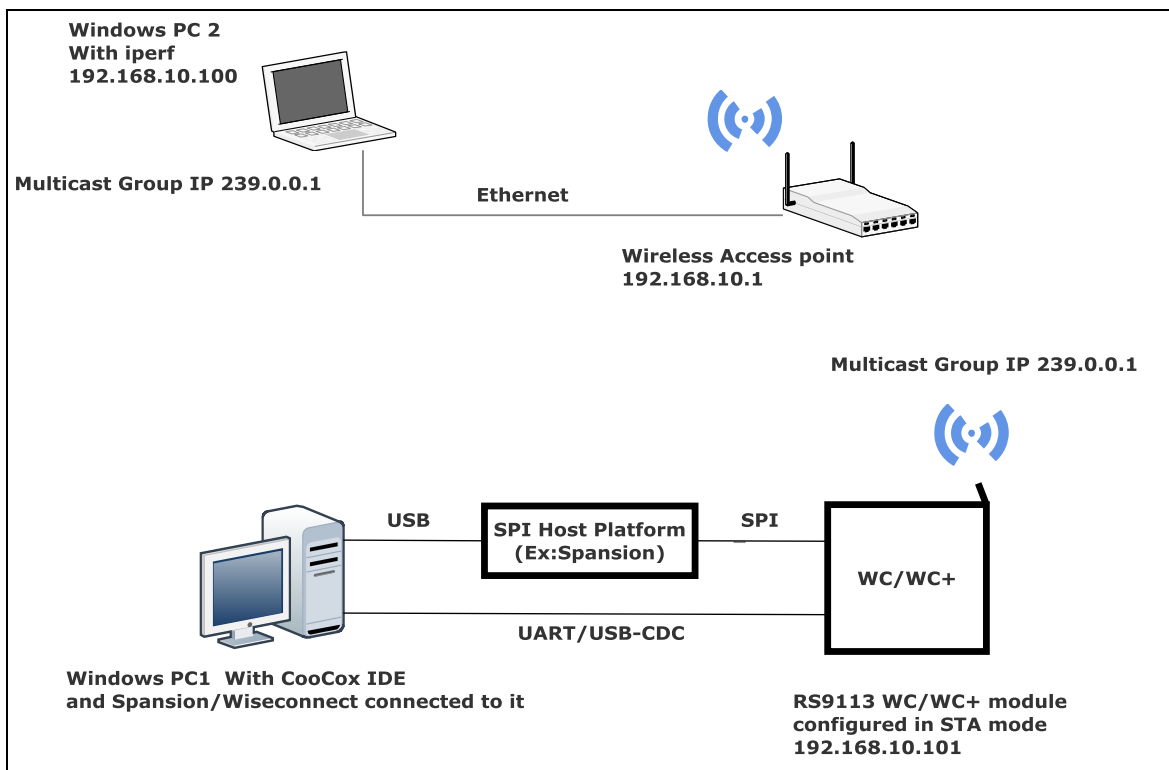


Figure 1: Setup Diagram

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 **multicast** example in CoCoX 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 **multicast** example in Dev-C++ IDE

2.2 Configuring the Application

1. Open *sapis/examples/wlan/multicast/rsi_multicast_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 to be scanned, If it is 0, device will scan all the channels

```
#define CHANNEL_NO          0
```

SECURITY_TYPE refers to the type of security. In this application STA supports Open, WPA-PSK, WPA2-PSK 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_OPEN
```

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

```
#define PSK                  "<psk>"
```

DEVICE_PORT port refers internal UDP port number

```
#define DEVICE_PORT          5001
```

SERVER_PORT port refers remote UDP server port number

```
#define SERVER_PORT          5002
```

MULTICAST_GROUP_ADDRESS refers the device to which multicast group address has to join. **MULTICAST_GROUP_ADDRESS** address should be configured in long format and in little endian byte order.

Example: To configure “239.0.0.1” as multicast group IP address, update the macro **MULTICAST_GROUP_ADDRESS** as **0x010000EF**.

```
#define MULTICAST_GROUP_ADDRESS 0x010000EF
```

NUMEBR_OF_PACKETS refers how many packets to send/receive to/from multicast group before leaving multicast group.

```
#define NUMBER_OF_PACKETS <no of packets>
```

RECV_BUFFER_SIZE is expected size of data in each packet. If packet is half the size of receive buffer, then Device will read for the data again.

```
#define RECV_BUFFER_SIZE <receive buf size>
```

To configure IP address

DHCP_MODE refers whether IP address configured through DHCP or STATIC

```
#define DHCP_MODE 1
```

Note: If user wants to configure STA IP address through DHCP then set **DHCP_MODE** to 1 and skip configuring the following **DEVICE_IP**, **GATEWAY** and **NETMASK** macros.

(Or)

If user wants to configure STA IP address through STATIC then set **DHCP_MODE** macro to “0” and configure following **DEVICE_IP**, **GATEWAY** and **NETMASK** macros.

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

Example: To configure “192.168.10.10” as IP address, update the macro **DEVICE_IP** as **0x0A0AA8C0**.

```
#define DEVICE_IP 0X0A0AA8C0
```

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
```

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

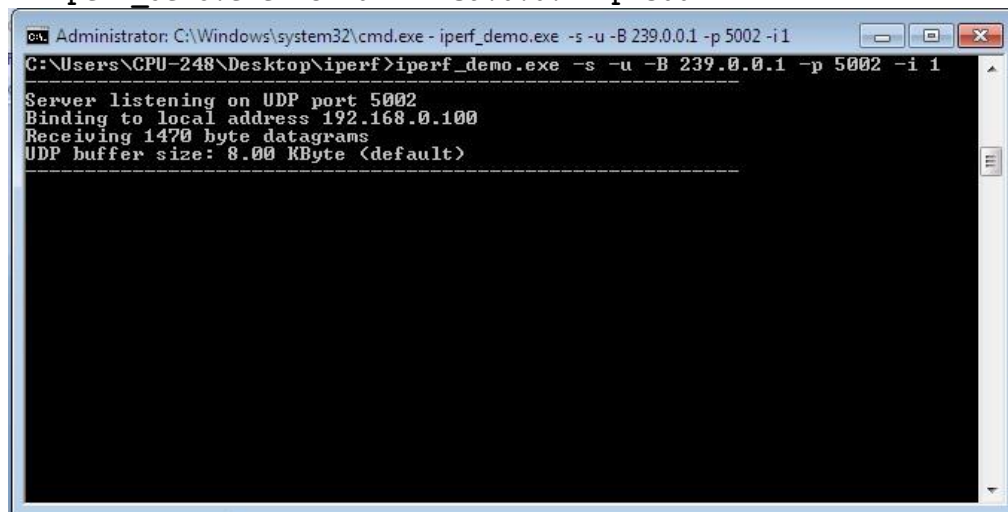
```
#define CONCURRENT_MODE RSI_DISABLE
#define RSI_FEATURE_BIT_MAP FEAT_SECURITY_OPEN
#define RSI_TCP_IP_BYPASS RSI_DISABLE
```

```
#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. Configure the Access point in OPEN/WPA-PSK/WPA2-PSK mode to connect WiSeConnect device in STA mode.
2. Open multicast UDP server socket on port number **SERVER_PORT** (Ex: 5002) by binding to **MULTICAST_GROUP_ADDRESS** (Ex: 239.0.0.1) using iperf application in Windows PC2 which is connected to Access point through LAN.

iperf_demo.exe -s -u -B 239.0.0.1 -p 5002 -i 1



```
Administrator: C:\Windows\system32\cmd.exe - iperf_demo.exe -s -u -B 239.0.0.1 -p 5002 -i 1
C:\Users\CPU-248\Desktop\iperf>iperf_demo.exe -s -u -B 239.0.0.1 -p 5002 -i 1
Server listening on UDP port 5002
Binding to local address 192.168.0.100
Receiving 1470 byte datagrams
UDP buffer size: 8.00 KByte (default)
```

3. 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 **multicast** example in CooCox IDE.

4. 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 **multicast** example in Dev-C++ IDE

- 5.
6. After the program gets executed, WiSeConnect Device will be connected to Access point and get IP.
7. After successful connection with Access point, Device will join to the multicast group and sends configured number of UDP packets to multicast group address on port number **SERVER_PORT**. After Device starts sending multicast data, user can see UDP receiving data on opened UDP socket on port number **SERVER_PORT**.


```
Administrator: C:\Windows\system32\cmd.exe - iperf_demo.exe -s -u -B 239.0.0.1 -p 5002 -i 1
[1240] Sent 309 datagrams
C:\Users\CPU-248\Desktop>iperf_demo.exe -s -u -B 239.0.0.1 -p 5002 -i 1
Server listening on UDP port 5002
Binding to local address 192.168.0.100
Receiving 1470 byte datagrams
UDP buffer size: 8.00 KByte (default)
-----
[1224] local 192.168.0.100 port 5002 connected with 192.168.0.101 port 5001
[ ID] Interval      Transfer      Bandwidth      Jitter      Lost/Total Datagram
[1224] 0.0- 1.0 sec  1.15 KBytes   9.41 Kbits/sec  21.160 ms   -1614698882/12146
444 (-1.3e+002%)
[1224] 1.0- 2.0 sec  1.99 KBytes   16.3 Kbits/sec  11.905 ms    -85/    0 (-1.5%
[1224] 1.0- 2.0 sec  85 datagrams received out-of-order
[1224] 2.0- 3.0 sec  1.01 KBytes   8.26 Kbits/sec  18.892 ms    -43/    0 (-1.5%
[1224] 2.0- 3.0 sec  43 datagrams received out-of-order
[1224] 3.0- 4.0 sec  1.57 KBytes   12.9 Kbits/sec  22.440 ms    -67/    0 (-1.5%
[1224] 3.0- 4.0 sec  67 datagrams received out-of-order
[1224] 4.0- 5.0 sec  1.90 KBytes   15.6 Kbits/sec  11.004 ms    -81/    0 (-1.5%
[1224] 4.0- 5.0 sec  81 datagrams received out-of-order
[1224] 5.0- 6.0 sec  1.83 KBytes   15.0 Kbits/sec  14.264 ms    -78/    0 (-1.5%
[1224] 5.0- 6.0 sec  78 datagrams received out-of-order
[1224] 6.0- 7.0 sec  1.27 KBytes   10.4 Kbits/sec  14.391 ms    -54/    0 (-1.5%
[1224] 6.0- 7.0 sec  54 datagrams received out-of-order
[1224] 7.0- 8.0 sec  1.24 KBytes   10.2 Kbits/sec  13.357 ms    -53/    0 (-1.5%
[1224] 7.0- 8.0 sec  53 datagrams received out-of-order
[1224] 8.0- 9.0 sec  1.66 KBytes   13.6 Kbits/sec  9.550 ms     -71/    0 (-1.5%
[1224] 8.0- 9.0 sec  71 datagrams received out-of-order
[1224] 9.0-10.0 sec  816 Bytes    6.53 Kbits/sec  18.346 ms    -34/    0 (-1.5%
[1224] 9.0-10.0 sec  34 datagrams received out-of-order
[1224] 10.0-11.0 sec  1.08 KBytes   8.83 Kbits/sec  20.584 ms    -46/    0 (-1.5%
[1224] 10.0-11.0 sec  46 datagrams received out-of-order
```

8. After sending configured NUMBER of UDP packets from device, remote Windows PC2 stops receiving data on SERVER_PORT and WiSeConnect device waits for receiving multicast data on UDP port number **DEVICE_PORT**.
9. From Windows PC2, after UDP data reception stops, open UDP client socket and send UDP data to multicast IP address with port number **DEVICE_PORT** by giving following command in **iperf**,

iperf_demo.exe -c 239.0.0.1 -p <DEVICE_PORT> -u -i 1 -t 100 -T32

```
Administrator: Command Prompt - iperf_demo.exe -c 239.0.0.1 -p 5001 -u -i 1 -t 20
C:\Users\test>cd Desktop
C:\Users\test\Desktop>cd iperf
C:\Users\test\Desktop\iperf>iperf_demo.exe -c 239.0.0.1 -p 5001 -u -i 1 -t 20
-----
Client connecting to 239.0.0.1, UDP port 5001
Sending 1470 byte datagrams
Setting multicast TTL to 1
UDP buffer size: 8.00 KByte (default)
-----
[132] local 192.168.10.5 port 57212 connected with 239.0.0.1 port 5001
[ ID] Interval      Transfer      Bandwidth
[132] 0.0- 1.0 sec  129 KBytes    1.06 Mbits/sec
[132] 1.0- 2.0 sec  128 KBytes    1.05 Mbits/sec
[132] 2.0- 3.0 sec  128 KBytes    1.05 Mbits/sec
[132] 3.0- 4.0 sec  128 KBytes    1.05 Mbits/sec
[132] 4.0- 5.0 sec  128 KBytes    1.05 Mbits/sec
[132] 5.0- 6.0 sec  128 KBytes    1.05 Mbits/sec
[132] 6.0- 7.0 sec  128 KBytes    1.05 Mbits/sec
[132] 7.0- 8.0 sec  128 KBytes    1.05 Mbits/sec
[132] 8.0- 9.0 sec  128 KBytes    1.05 Mbits/sec
[132] 9.0-10.0 sec  129 KBytes    1.06 Mbits/sec
[132] 10.0-11.0 sec  128 KBytes    1.05 Mbits/sec
```

10. Device will read configured number of packets which are coming from joined multicast group address and leave from that joined multicast group.