WIFI Direct Client Application

User guide
Version 0.1

September 2015

Redpine Signals, Inc.

2107 N. First Street, #680

San Jose, CA 95131.

Tel: (408) 748-3385

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

Application Overview:

Wi-Fi Direct is a Wi-Fi standard enabling devices to easily connect with each other without requiring a wireless access point. It is usable for everything from internet browsing to file transfer and to communicate with more than one device simultaneously at typical Wi-Fi speeds. One advantage of Wi-Fi Direct is the ability to connect devices even if they are from different manufacturers.

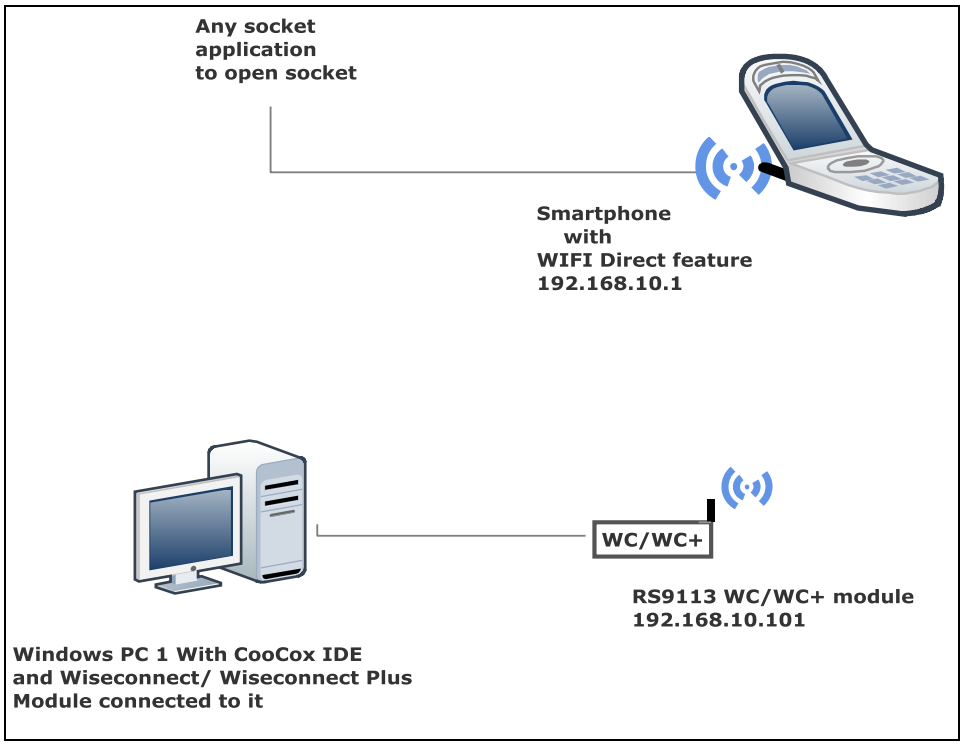
Wi-Fi Direct may not only replace the need for routers, but may also replace the need of Bluetooth for applications that do not rely on low energy.

The application demonstrates how to configure device in WIFI Direct mode and how to open socket to transmit or receive data. Following are steps:

- Configure in WIFI Direct mode
- Connect to peer device
- Open TCP socket

Setup required:

1. Windows PC with Coocox IDE.
2. WiSeConnect device.
3. Remote device or peer device(Smart Phone) configured in WIFI Direct mode to which device will connect.



Description:

WiSeConnect/WiSeConnectPlus device act as WIFI Direct client mode.

Device wait for remote device to join or send join request based on **CONNECTION_TYPE** macro set.

If **CONNECTION_TYPE** is set to **ACCEPT_CONNECTION_REQ** device wait for remote device join request after that device will send join request.

If **CONNECTION_TYPE** is set to **SEND_CONNECTION_REQ** device first send join request to remote device to join.

After joining device may act as GO(Group Owner) or client, based on GO Intent value, negotiation will happen between device and remote side device.

Get the IP address for device. Device open TCP socket to data transmit and receive. At the remote side device also need to open TCP socket.

Configuring the application:

Edit the **rsi_wfd_client.c** file in the following path to establish connection with the remote WIFI Direct device.

sapis/examples/wlan/wifi_direct/

1. From given configuration,

CONNECTION_TYPE is set to send join request or

wait for join request from remote device.

```
#define   CONNECTION_TYPE              SEND_CONNECTION_REQ
```

REMOTE_DEVICE is WIFI-Direct device to which wants to connect.

```
#define   REMOTE_DEVICE                "<device_name>"
```

RSI_DEVICE_NAME refers to device name for module. Maximum

length of this field is 32 characters.

```
#define   RSI_DEVICE_NAME              "<ap_name>"
```

POST_FIX_SSID refers to post fix SSID for device

```
#define   POST_FIX_SSID                "<post_ssid>"
```

GO_INTENT this determines whether the device is intended to form

a GO(Group Owner) or work as a WIFI-Direct peer node. GO Intent value is between 0 to 16

```
#define   GO_INTENT                    0
```

CHANNEL refers to channel in which device would operate

```
#define   CHANNEL                      0
```

PSK refers to password, this PSK is used if the module becomes a GO owner

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

2. Enable/Disable DHCP mode

   1 – Enables DHCP mode (gets the IP from DHCP server)

   0 – Disables DHCP mode

```
#define DHCP_MODE 1
```

3. If DHCP mode is disabled, then change the following macros to configure static IP address

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

   Example: To configure "192.168.10.101" as IP address, update the macro **DEVICE_IP** as **0x650AA8C0**.

```
#define   DEVICE_IP            0x650AA8C0
```

   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
```

4. To open TCP server socket and receive TCP data configure the below macros.

Internal socket port number.

```
#define DEVICE_PORT 5001
```

Number of packets to send

```
#define NUMBER_OF_PACKETS 1000
```

Application memory length which is required by the driver

```
#define  GLOBAL_BUFF_LEN  8000
```

Edit the Wlan configuration file:

sapis/include/rsi_wlan_config.h

CONCURRENT_MODE	DISABLE
RSI_FEATURE_BIT_MAP	FEAT_SECURITY_OPEN
RSI_TCP_IP_BYPASS	DISABLE
RSI_TCP_IP_FEATURE_BIT_MAP	(TCP_IP_FEAT_DHCPV4_CLIENT | TCP_IP_FEAT_DHCPV4_SERVER)
RSI_CUSTOM_FEATURE_BIT_MAP	0
RSI_BAND	RSI_BAND_2P4GHZ

Note: Supported security modes list are mentioned in rsi_wlan_apis.h which is at follow path: /sapis/include/rsi_wlan_apis.h

Executing the Application:

1. Connect WiSeConnect device to the PC open IDE.

2. Configure the macros in the files located at

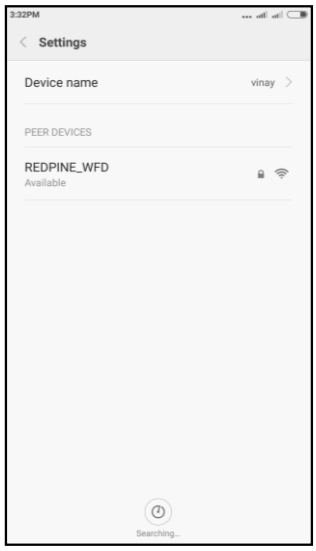
   sapis/examples/wlan_ble/wlan/wifi_direct/rsi_wfd_client.c

   sapis/include/rsi_wlan_config.h

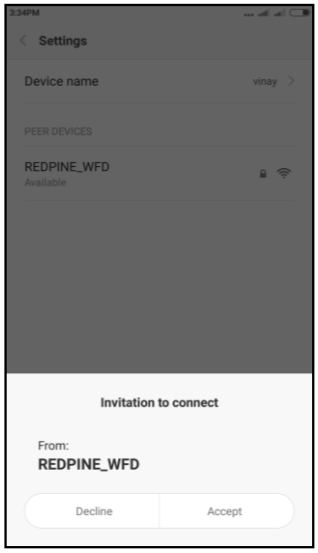
3. Build and launch the application.

4. Configure remote device in WIFI-Direct to connect to module.

5. After the program gets executed, WiSeConnect module configured as WIFI-Direct client and send join request or wait for join request.



WiFi Direct Device



Connection Request

Device Connected

6. After sending connect request GO negotiation start.

7. After negotiation, IP assign to module and TCP server socket opened in module and waits for TCP connection from remote peer.

8. Open the TCP client socket and send data from remote peer using any TCP application like iperf.

9. WiSeConnect device receives TCP data configured number of packets sent by remote peer and exits.