RS9113 Module Evaluation Kit

User Guide

Version 2.7

December 2017

About this Document

This document covers the RS9113 Module's Evaluation Board(EVB) and its usage for evaluating Silicon Labs' RS9113 based ultra-low-power, single spatial stream, dual-band 802.11n + BT4.0 + ZigBee Convergence modules in n-Link® and WiSeConnect® modes.

Table Of Contents

Table of Figures

Table of Tables

1 Overview

The RS9113 Module Evaluation Kit (EVK) is a platform for evaluating the RS9113 modules with multiple Host Processors/MCUs over interfaces like SDIO, USB, USB-CDC, SPI and UART. The EVK includes sample driver, supplicant, applications to test the following:

- Wireless Functionality for Wi-Fi, BT 4.0 and ZigBee
- Security modes
- Throughputs
- Power Consumption
- Firmware Upgrade

The RS9113 n-Link®, WiSeConnect® module families are based on Silicon Labs' RS9113 ultra-low-power, single spatial stream, dual-band 802.11n + BT4.0 + ZigBee Convergence SoC. The RS9113 module integrates a multi-threaded MAC processor with integrated analog peripherals and support for digital peripherals, baseband digital signal processor, analog front-end, crystal oscillator, calibration OTP memory, dual-band RF transceiver, dual-band high-power amplifiers, baluns, diplexers, diversity switch and Quad-SPI Flash thus providing a fully-integrated solution for embedded wireless applications.

Note: All the latest user level Documents, Firmware Release packages, certifications of the module and other material related to the RS9113 based Modules is available on our Documentation and Software server.

http://www.redpinenetworks.com/OpenKM/com.openkm.frontend.Main/index.jsp [1]

1.1 The RS9113 WiSeConnect® and Connect-io-n®

The WiSeConnect® and Connect-io-n® modules offer WLAN, ZigBee and Bluetooth protocols along with Wi-Fi Direct™( ), WPA/WPA2-PSK, WPA/WPA2-Enterprise (EAP-TLS, EAP-FAST, EAP-TTLS, PEAP-MS-CHAP-V2) and a feature-rich networking stack embedded in the device, thus providing a fully-integrated solution for embedded wireless applications. These modules can be interfaced to 8/16/32-bit host processors through SPI, UART, USB and USB-CDC interfaces.

To evaluate The WiSeConnect® and Connect-io-n®module Family based EVB refer to Section 5: Evaluation of Connect-io-n®/WiSeConnect®

1.2 The RS9113 n-Link®

The n-Link™ module Family is suitable for high throughput applications where software stack runs on the host processor utilizing the networking stack of operating systems such as Linux, Windows, Embedded OS like WinCE and RTOS. The OneBox ™ software package developed by Silicon Labs allows for the same hardware design to operate in both Wi-Fi station and access point modes. The solution also supports Wi-Fi Direct. Both SDIO and SPI interfaces are supported in this solution.

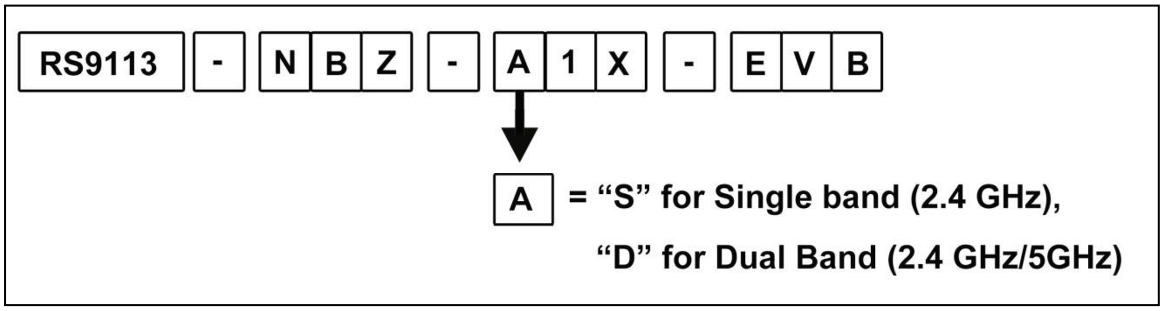
To evaluate the RS9113 n-Link™ module Family based EVB refer to Section 4: Evaluation of n-Link

[1] This server requires log in credentials. Please contact your Account manager if you require these details.

2 Evaluation Kit Details

2.1 Evaluation Kit Part Numbers

2.1.1 Ordering Information for Evaluation Kits


RS9113 - N B Z - A 1 X - E V B
A = "S" for Single band (2.4 GHz),
"D" for Dual Band (2.4 GHz/5GHz)

2.1.2 Related Links

- RS9113 n-Link Module Product Brief
- RS9113 Connect-io-n Module Product Brief
- RS9113 WiSeConnect Module Product Brief

2.2 Evaluation Kit Contents

The RS9113 Module Evaluation Kit comes with the following components:

1) RS9113 Module Evaluation Board
2) USB Pen drive
3) Micro A/B-type USB cable
4) SDIO Adaptor Cable
5) SPI Adaptor Cable

It is highly recommended to use the Micro A/B type USB cable that comes with the kit. If a longer cable is needed ensure that you use a USB-IF certified cable which can supply peak current of at least 500mA.

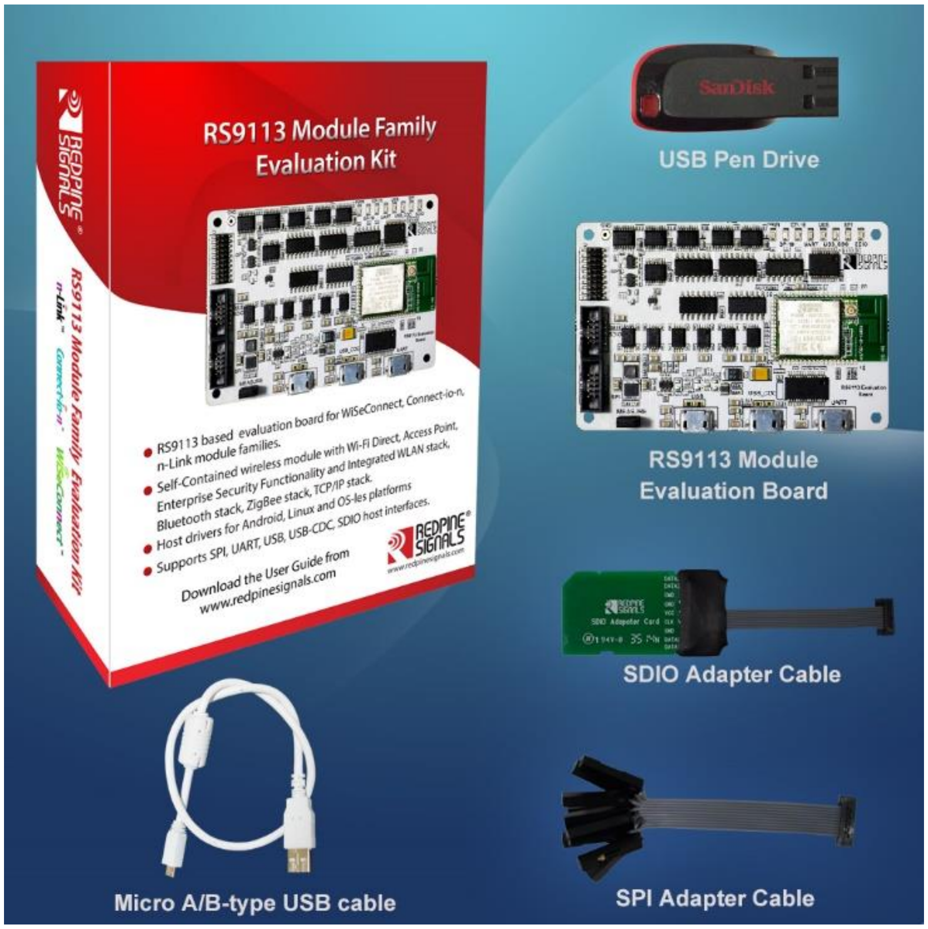


Figure 1: Evaluation Kit Contents

The USB drive is bootable. It is loaded with Fedora Core 18 OS with the OneBox-Mobile driver binaries included for evaluation of n-Link® modules on a standard PC/Laptop platform.

The USB drive also has the release package for the Connect-io-n®/WiSeConnect® modules which includes the firmware, documentation and reference projects for multiple platforms.

Redpine provides drivers for multiple OS's and MCU platforms for the n-Link® modules and also OS-less MCU platforms for WiSeConnect® modules. The software provided in this kit is to enable easy and quick evaluation on a PC. Please contact Silicon Labs' Sales for availability of drivers for an OS and MCU of your choice.

3 Hardware Details

This section describes the RS9113 EVB's various components and headers.

The OneBox-Mobile software for the n-Link® modules supports SDIO and USB interfaces to connect to the Host MCU. The OneBox-Embedded software for the WiSeConnect® modules supports UART, SPI, USB and USB-CDC interfaces to connect to the Host MCU.

As shown in the image below, the RS9113 EVB has three USB connectors for the USB, USB-CDC and UART connections. The UART signals of the module are converted to USB using on-board circuit. The board also has SDIO and SPI Headers.

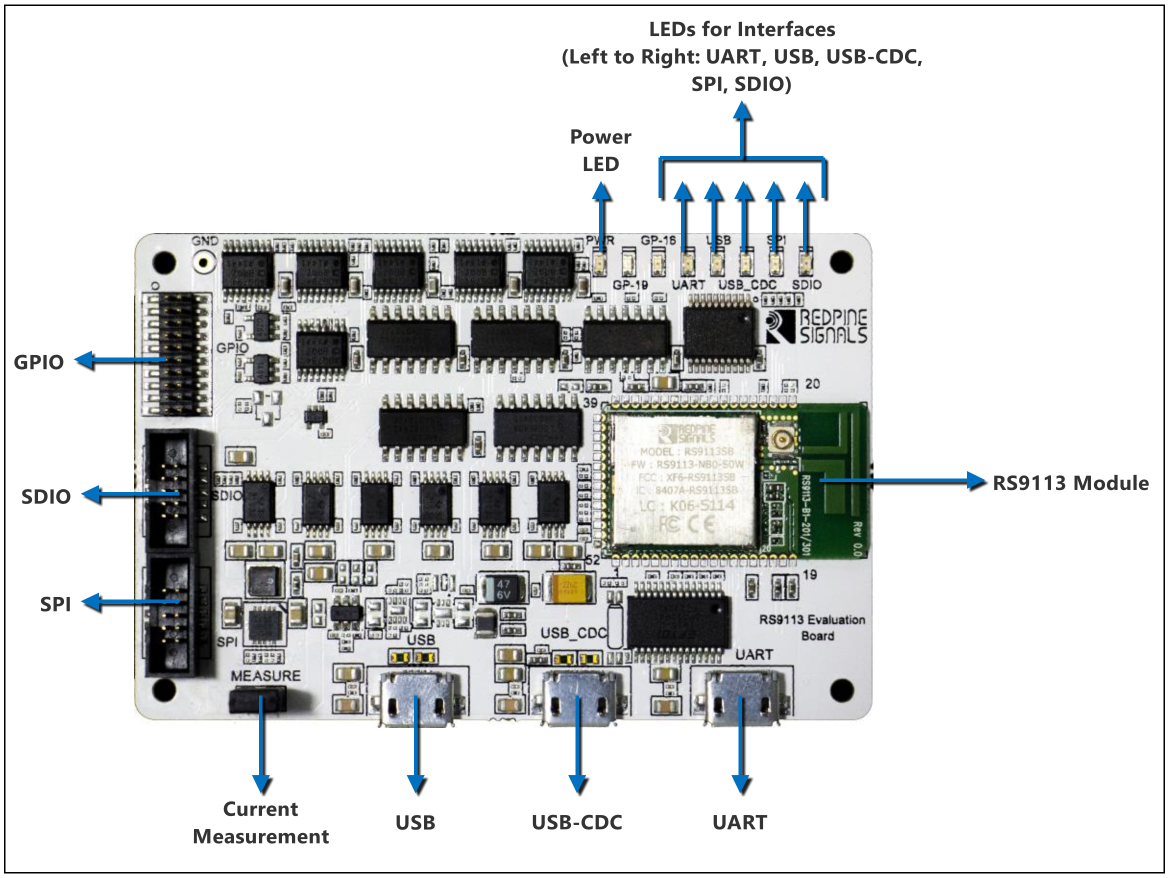


Figure 2: RS9113 EVB

The board is designed to configure the module to use the interface on which power supply is detected. This is indicated through the LEDs mounted on the board. The SDIO and SPI interfaces require power supply to be provided over the USB connector. Hence, for these interfaces, it is required that the USB connection be provided first followed by the SDIO or SPI connection.

Follow the steps below to use the EVB with different interfaces:

1) USB, UART, USB-CDC Modes

   a. Connect the Micro A/B-type USB cable between a USB port of a PC/Laptop and the micro-USB port labeled USB, UART or USB-CDC on the EVB.
   b. Verify that the LED labeled "USB", "UART" or "USB-CDC" lights up as per the port which is connected.

2) SDIO Mode
   a. Connect the Micro A/B-type USB cable between a USB port of a PC/Laptop and the micro-USB port labeled USB on the EVB – this connection will be used only as a power supply in this mode.
   b. Connect the 10-pin header of the SDIO Adaptor Cable to the EVB. Insert the SDIO Adaptor into the SDIO slot of the PC/Laptop.
   c. Verify that the LED labeled "SDIO" lights up.

3) SPI Mode
   a. Connect the Micro A/B-type USB cable between a USB port of a PC/Laptop and the micro-USB port labeled USB on the EVB – this connection will be used only as a power supply in this mode.
   b. Connect the 10-pin header of the SPI Adaptor Cable to the EVB. Connect the other wires of this connector to the SPI signals of a Host MCU platform. The details of the Header are given in Appendix A.

There is a 2-pin inline jumper available for measuring the current being sourced by the module during different stages of operation. This is labeled as "MEASURE" on the PCB. The user may connect a power meter or an ammeter to this jumper to measure the current.

To proceed further, click on the links below as per your requirement.

1) Evaluation of n-Link® modules
2) Evaluation of WiSeConnect® modules

4 Evaluation of n-Link®

This section covers the evaluation of the n-Link® mode of the EVK using the OneBox-Mobile drivers in the following modes:

1) Wi-Fi
    a. Client
    b. Access Point
    c. Wi-Fi Direct
2) Wi-Fi Client and Bluetooth Classic
3) Wi-Fi Client and Bluetooth Low Energy
4) Wi-Fi Client and ZigBee

The n-Link modules support SDIO and USB as the host interfaces.

4.1 Required Setup

- RS9113 EVK
- PC/Laptop to program the RS9113 EVB over SDIO or USB
- For evaluation of the Wi-Fi Client mode, you will require:
    - Wi-Fi Access Point
    - A second PC/Laptop with Ethernet
- For evaluation of the Wi-Fi Access Point mode, you will require:
    - A second PC/Laptop with Wi-Fi
- For evaluation of the Wi-Fi Direct mode, you will require:
    - A device with Wi-Fi Direct capability like an Android 4.0 Smartphone or a Laptop
- For evaluation of the Bluetooth Classic mode, you will require a device like a Smartphone or Laptop with Bluetooth capability
- For evaluation of the ZigBee mode, you will require a ZigBee Coordinator and ZigBee-enabled Light bulb which support the Home Automation Profile.

NOTE: The procedure explained below is common across the USB and SDIO interfaces.

4.2 Getting Started

The USB flash drive provided with the EVK is to be used for evaluation of n-link mode only. Please follow the steps below to get started with the evaluation process.

1) Plug the USB drive provided in the EVK and turn on the host PC. Interrupt the PC from booting in normal mode (default) by pressing '**F12**' key you can see the boot menu option, from there, select the boot device as USB storage device and click Enter.

NOTE: If you have a Windows 8 or later PC, the booting process might fail because of the new UEFI mode (instead of BIOS) being used in such PCs. Disable the UEFI Boot mode (enable CSM boot mode) and Secure Bootup mode to boot from the USB drive on such PCs. Please follow the steps listed in Appendix E: Booting USB Drive on Windows 8 and Later PCs

Once these changes are done, the original Windows 8 OS will not be bootable till the changes are reverted.

2) Once these changes are done, the original Windows 8 OS will not be bootable till the changes are reverted.

3) You will be presented a menu with the following options:

   1. Start RS9113 X.X.X (debug)
   2. Start RS9113 X.X.X (PAE)
   3. Troubleshooting

   Here X.X.X represents the Live USB image version. Make sure that “Start RS9113 X.X.X (PAE)” is selected from the above options

NOTE: To update/upgrade to latest version of Live USB image or Flashing procedure of Live USB image, Please follow the steps listed in Appendix D under section “Steps to update Live USB image”

4) Select “Start RS9113-X.X.X (PAE)” and click Enter.

   NOTE: In some PCs the bootup might fail because of Graphics hardware incompatibility with the Fedora Core 18 OS loaded on the USB drive. The screen will go blank rather than showing the booting process. In such cases, select “Troubleshooting” then attempt to boot in the Basic Graphics Mode.

5) Wait until the booting up completes and you get the Linux screen.

6) Press “Alt+F2” and enter “gnome-terminal” in the pop-up window to open a terminal window.

7) Enter into super user mode using the “su” command in the terminal. Enter “redpine” as the password when prompted.

8) Follow the steps below to use the EVB with different interfaces:

   a. USB Mode

      i. Connect the Micro A/B-type USB cable between a USB port of a PC/Laptop and the micro-USB port labeled USB, UART or USB-CDC on the EVB.

      ii. Verify that the LED labeled “USB”, “UART” or “USB-CDC” lights up as per the port which is connected.

   b. SDIO Mode

      i. Connect the Micro A/B-type USB cable between a USB port of a PC/Laptop and the micro-USB port labeled USB on the EVB – this connection will be used only as a power supply in this mode.

      ii. Connect the 10-pin header of the SDIO Adaptor Cable to the EVB. Insert the SDIO Adaptor into the SDIO slot of the PC/Laptop.

iii. Verify that the LED labeled “SDIO” lights up.

9) Go to the /home/OneBox-Mobile folder and follow the instructions listed in the sub-sections below.

10) To proceed further, click on one of the links below as per your choice.

   a. OneBox-Mobile in Wi-Fi Only Mode
   b. OneBox-Mobile in Wi-Fi + Bluetooth Classic Mode
   c. OneBox-Mobile in Wi-Fi + Bluetooth LE Mode
   d. OneBox-Mobile in Wi-Fi + Bluetooth ZigBee Mode

4.3 OneBox-Mobile in Wi-Fi Only Mode

1) Open the common_insert.sh file present in “/home/OneBox-Mobile/” using the gvim editor.

2) Ensure that the DRIVER_MODE and COEX_MODE are set as below

   a. DRIVER_MODE = 1
   b. COEX_MODE = 1 (For Station Mode only/WIFI-Direct)

      COEX_MODE = 2 (For ACCESS POINT)

      COEX_MODE = 3 (For Both ACCESS POINT and Station Mode)

NOTE: For SDIO mode, ensure that the SDIO stack related modules are already inserted in the kernel. Follow the steps below for this:

```
# cd /home/OneBox-Mobile/
# sh load_stack.sh
# lsmod
```

Verify that the output of the “lsmod” command lists the sdhci.ko, sdhci_pci.ko, mmc_block.ko and mmc_core.ko modules. This is a one-time process and need not be repeated unless the modules are explicitly removed by the user.

4.3.1 Installation in Wi-Fi Client Mode

1) Edit the “sta_settings.conf” file in the /home/OneBox-Mobile/ folder using gvim and enter the parameters of the Wi-Fi network as below.

   a. For Open (non-Secure) mode

```
network={
ssid=”<SSID of Access Point>”
key_mgmt=NONE
}
```

   b. For Open (non-Secure) mode connection to a Hidden SSID

```
network={
ssid=”<SSID of Access Point>”
scan_ssid=1
key_mgmt=NONE
}
```

c. For WEP-64 mode

```
network={
ssid=”<SSID of Access Point>”
key_mgmt=NONE
wep_key0=XXXXXXXXXX
wep_tx_keyidx=X
}
```

The key can be input either in ASCII or Hexadecimal formats:

ASCII Format: `wep_key0=”12345”`

Hexadecimal Format: `wep_key0=1234567890`

d. For WEP-128 mode

```
network={
ssid=”<SSID of Access Point>”
key_mgmt=NONE
wep_key0=XXXXXXXXXXXXXXXXXXXXXXXXXX
wep_tx_keyidx=X
}
```

The key can be input either in ASCII or Hexadecimal formats:

ASCII Format: `wep_key0=”1234567890123”`

Hexadecimal Format: `wep_key0=12345678901234567890123456`

e. For WPA-PSK (TKIP) mode

```
network={
ssid=”<SSID of Access Point>”
key_mgmt=WPA-PSK
psk=<passphrase specified in the Access Point>
proto=WPA
pairwise=TKIP
group=TKIP
}
```

f. For WPA2-PSK (CCMP) mode

```
network={
ssid="<SSID of Access Point>"
key_mgmt=WPA-PSK
psk=<passphrase specified in the Access Point>
proto=WPA2
pairwise=CCMP
group=CCMP
}
```

2) Next, run the "start_sta.sh" script in the /home/OneBox-Mobile/ folder to load the driver modules and the supplicant and connect to the Access Point specified in the "sta_settings.conf" file.

```
# sh start_sta.sh
```

3) After issuing the above command a virtual interface with the name "wifi0" will be created. You can view the list of interfaces using the following command:

```
# ifconfig –a
```

4) You can check whether the connection to the Access Point is successful by running the following command:

```
# iwconfig wifi0
```

This command gives the status of the device. If the connection is successful, then the connected Access point SSID along with the MAC address is displayed. If it is not connected to an Access point a message "Not Associated" appears.

5) To view the list of Access Points scanned in each channel, you can run the following command in the /home/OneBox-Mobile/ folder.

```
# ./wpa_cli –i wifi0 scan_results
```

6) To obtain an IP address using DHCP, start the DHCP client using the command below.

```
# dhclient wifi0
```

4.3.2 Installation in Access Point Mode

The "start_ap.sh" script present in the /home/OneBox-Mobile/ folder needs to be run with the different configuration files provided in the same folder to install an Access Point in different security modes.

```
# cd /home/OneBox-Mobile/
# sh start_ap.sh <conf_file>
```

The different configuration files (conf_file) provided are as follows:

1) Access Point in Open Mode
   a. Configuration File: wpa_supplicant_open.conf
   b. This starts an Access Point with the following parameters:
      i. SSID: REDPINE_AP

ii. Channel 1 of 2.4GHz Band (2412 MHz)

iii. Open (non-Secure) mode

2) Access Point in WEP-64 Mode

  a. Configuration File: wpa_supplicant_wep64.conf

  b. This starts an Access Point with the following parameters:

    i. SSID: onebox_wep

    ii. Channel 1 of 2.4GHz Band (2412 MHz)

    iii. Security Mode: WEP-64

    iv. WEP Key: 1234567890

    v. Key Index: 0

3) Access Point in WEP-128 Mode

  a. Configuration File: wpa_supplicant_wep128.conf

  b. This starts an Access Point with the following parameters:

    i. SSID: onebox_wep

    ii. Channel 1 of 2.4GHz Band (2412 MHz)

    iii. Security Mode: WEP-128

    iv. WEP Key: 12345678901234567890123456

    v. Key Index: 0

4) Access Point in WPA-PSK (TKIP) Mode

  a. Configuration File: wpa_supplicant_tkip.conf

  b. This starts an Access Point with the following parameters:

    i. SSID: onebox_tkip

    ii. Channel 1 of 2.4GHz Band (2412 MHz)

    iii. Security Mode: WPA-PSK (TKIP)

    iv. Passphrase: “12345678”

5) Access Point in WPA2-PSK (CCMP) Mode

  a. Configuration File: wpa_supplicant_ccmp.conf

  b. This starts an Access Point with the following parameters:

    i. SSID: onebox_ccmp

    ii. Channel 1 of 2.4GHz Band (2412 MHz)

    iii. Security Mode: WPA2-PSK (CCMP)

    iv. Passphrase: “12345678”

NOTE: All the above parameters can be modified in the respective configuration files by the user. The values provided are only for reference.

After running the “start_ap.sh”script a virtual interface with the name “wifi0” will be created. You can view the list of interfaces using the following command:

```
# ifconfig -a
```

You can check whether the Access Point has been started successfully by running the following command:

```
# iwconfig wifi1
```

This command gives the status of the device. It displays the Access Point’s SSID along with the MAC address and channel frequency. If the Access Point doesn’t start, a message saying “Exiting: Driver Initialization not completed even after waiting for xxms” is displayed.

To start a DHCP server, use the commands below.

```
# sh dhcp_server.sh wifi1
```

4.3.3 Installation in Wi-Fi Direct Mode

The “start_p2p.sh” script present in the /home/OneBox-Mobile/ folder needs to be run to start the supplicant and install the Wi-Fi Direct mode. The configurable parameters in the p2p.conf file are the listen channel, operating channel and GO Intent. After starting the supplicant the p2p_commands mentioned below should be executed.

1) To find other P2P networks

```
# ./wpa_cli -i wifi0 p2p_find
```

2) To find other P2P devices in range

```
# ./wpa_cli -i wifi0 p2p_peers
```

3) To connect to a P2P network

```
# ./wpa_cli -i wifi0 p2p_connect <BSS ID> pbc
go_intent=<intent value>
```

4) To start device in Autonomous GO Mode

```
# ./wpa_cli -i wifi0 p2p_group_add freq=<channel_freq>
```

The “channel_freq” input above is the center frequency of the Wi-Fi channel in which the GO needs to start1. If this parameter is not provided, then the GO starts in the channel specified in the p2p.conf file. Legacy Wi-Fi clients (non P2P clients) need a passphrase to connect to the P2P group. The command below generates the passphrase for legacy Wi-Fi clients.

4.3.4 Checking Throughputs

You can check the Transmit and Receive throughputs in UDP and TCP modes by using the iperf application. If you wish to evaluate the throughputs in Wi-Fi Client mode, you will need to connect a second PC/Laptop to the Access Point. Download and install the iperf application fromhttps://iperf.fron the second PC/Laptop.

Please note the following points for this evaluation:

1) To evaluate the module for throughput performance, it’s recommended that you connect the second PC/Laptop to the Access Point over Ethernet.
2) It is recommended that you choose a Wi-Fi channel which has less interference, preferably in the 5GHz band if the EVK is for a Dual band module, to observe optimal throughputs of the module.

3) Wireless throughputs vary with the environment of the test setup – distance, obstacles, type of obstacles, etc.

4) The throughputs are also dependent on whether all the Wi-Fi components in the test support 40MHz and 20MHz or only 20MHz bandwidths.

To check Transmit throughputs, first run the command below on the 2nd PC/Laptop:

```
# iperf –s <-u> -i 1
```

This command starts an iperf Server which is listening for data. It prints the received throughput at an interval of 1 second. The <-u>option in the above command is for UDP traffic. If it is not mentioned, then the traffic is TCP.

Next, run the command below on the PC/Laptop connected to the EVB.

```
# iperf –c <IP address of 2nd PC><-u> -i 1 <-b 100m> –t 120
```

This command starts an iperf Client which starts transmitting data to the Server. The <-u> and <-b 100m> options in the above command are for UDP traffic. If they are not mentioned, then the traffic is TCP.

To check Receive throughputs, run the Server command first on the PC/Laptop connected to the EVB, followed by the client command on the 2nd PC/Laptop.

4.4 OneBox-Mobile in Wi-Fi + Bluetooth Classic Coexistence Mode

This section describes the installation of WiFi and BT Classic modes. Note that to use the Coexistence mode, each protocol should be loaded individually one after the other.

1) Open the common_insert.sh file present in "/home/OneBox-Mobile/" using the gvim editor.

2) Ensure that the DRIVER_MODE and COEX_MODE as set as below

    a. DRIVER_MODE = 1

    b. COEX_MODE = 5 (For WLAN Station and BT Classic Mode)

       COEX_MODE = 6 (For WLAN Access Point and BT Classic Mode)

NOTE: For SDIO mode, ensure that the SDIO stack related modules are already inserted in the kernel. Follow the steps below for this:

```
# cd /home/OneBox-Mobile/
# sh load_stack.sh
# lsmod
```

Verify that the output of the "lsmod" command lists the sdhci.ko, sdhci_pci.ko, mmc_block.ko and mmc_core.ko modules. This is a one-time process and need not be repeated unless the modules are explicitly removed by the user.

3) Follow the instructions in Section 4.3.1, to install the Wi-Fi Client mode.

4) Run the "bt_enable.sh" script present in the /home/OneBox-Mobile/ folder to start the Bluetooth Classic mode. This script inserts Bluetooth modules and common HAL modules if not already inserted.

5) You can check whether the BT Classic mode has been started successfully by running the following command:

```
# hciconfig
```

If the driver is loaded correctly, the above command displays a network adaptor named “hciX”. An example output if given below:

```
hci0:    Type: BR/EDR  Bus: SDIO

BD Address: 00:23:A7:00:05:68  ACL MTU: 1021:8  SCO MTU:
30:8

UP RUNNING PSCAN

RX bytes:478 acl:0 sco:0 events:20 errors:0

TX bytes:331 acl:0 sco:0 commands:19 errors:0
```

6) After the device is up, we can pair with other devices or from other devices using the Bluetooth Manager application. Files can also be sent and received using Bluetooth Manager. Instead of Bluetooth Manager, the device can be configured using “hcitool” or “hciconfig”. The procedure for using Bluetooth Manager is explained in Appendix B.

4.5 OneBox-Mobile in Wi-Fi + Bluetooth LE Coexistence Mode

This section describes the installation of WiFi and Bluetooth LE (BLE) modes. Note that to use the Coexistence mode, each protocol should be loaded individually one after the other.

1) Open the common_insert.sh file present in “/home/OneBox-Mobile/” using the gvim editor.
2) Ensure that the DRIVER_MODE and COEX_MODE as set as below
   a. DRIVER_MODE = 1
   b. COEX_MODE = 9 (For WLAN Station and BT LE)

NOTE: For SDIO mode, ensure that the SDIO stack related modules are already inserted in the kernel. Follow the steps below for this:

```
# cd /home/OneBox-Mobile/

# sh load_stack.sh

# lsmod
```

Verify that the output of the “lsmod” command lists the sdhci.ko, sdhci_pci.ko, mmc_block.ko and mmc_core.ko modules. This is a one-time process and need not be repeated unless the modules are explicitly removed by the user.

3) Follow the instructions in Section 4.3.1, to install the Wi-Fi Client mode.
4) Run the “bt_enable.sh” script present in the /home/OneBox-Mobile/ folder to start the Bluetooth Classic mode. This script inserts Bluetooth modules and common HAL modules if not already inserted.
5) You can check whether the BLE mode has been started successfully by running the following command:

```
# hciconfig
```

If the driver is loaded correctly, the above command displays a network adaptor named “hciX”. An example output if given below:

```
hci0:    Type: BR/EDR  Bus: SDIO
```

```
BD Address: 00:23:A7:00:05:68  ACL MTU: 1021:8  SCO MTU:
30:8

UP RUNNING PSCAN

RX bytes:478 acl:0 sco:0 events:20 errors:0

TX bytes:331 acl:0 sco:0 commands:19 errors:0
```

6) After the device is up, we can Advertise, Scan and Connect with other BLE devices. The device can be configured using hcitool or hciconfig.

   Advertise, Scan, Connect Commands

   a. Enable Advertise

      # hciconfig –a <hciX> leadv

   b. Disable Advertise

      # hciconfig –a <hciX> noleadv

   c. Initiate Scan

      # hcitool -i <hciX> lescan

      The above command displays the scan responses and advertising information.

   d. Master Mode Connected State

      Ensure that the remote device is in Advertise mode and then issue the command below.

      # hcitool –i <hciX> lecc <remote_MAC_Addr>

      The “remote_MAC_Addr” parameter above is the MAC address of the remote device, e.g.: 00:23:AC:01:02:03.

   e. Slave Mode Connected State

      Ensure that our device is in Advertise mode and then issue command below.

      # hcitool –i <hciX> lecc <device_MAC_Addr>

      The “device_MAC_Addr” parameter above is the MAC address of the Redpine module,   e.g., 00:23:AC:01:02:03

4.6 OneBox - Mobile in Wi-Fi + ZigBee Coexistence Mode

This section describes the installation of WiFi and ZigBee (ZB) modes. Note that to use the Coexistence mode, each protocol should be loaded individually one after the other.

1) Open the common_insert.sh file present in “/home/OneBox-Mobile/” using the gvim editor.

2) Ensure that the DRIVER_MODE and COEX_MODE as set as below

   a. DRIVER_MODE = 1

   b. COEX_MODE = 17 (For WLan Station and ZigBee)

NOTE: For SDIO mode, ensure that the SDIO stack related modules are already inserted in the kernel. Follow the steps below for this:

```
# cd /home/OneBox-Mobile/
# sh load_stack.sh
# lsmod
```

Verify that the output of the "lsmod" command lists the sdhci.ko, sdhci_pci.ko, mmc_block.ko and mmc_core.ko modules. This is a one-time process and need not be repeated unless the modules are explicitly removed by the user.

3) Follow the instructions in Section 4.3.1, to install the Wi-Fi Client mode.
4) Run the "zigb_enable.sh" script present in the /home/OneBox-Mobile/ folder to start the ZigBee mode. This script inserts ZigBee modules and common HAL modules if not already inserted.
5) You can check whether the ZigBee mode has been started successfully by running the following command:

   ```
   # ifconfig -a
   ```

   If the driver is loaded correctly, the above command displays a network adapter named "zigb0".

4.6.1 Building and Running the Sample Home Automation Switch Application

To help evaluate the ZigBee mode, a sample Home Automation switch application is made available with the release. You will need a 3rd party ZigBee Coordinator and ZigBee-enabled Light bulb which support the Home Automation Profile. Ensure that the Coordinator and Light bulb are switched on and connected before proceeding further.

4.6.1.1 About the Sample Application

This is the ZigBee Home Automation-defined switch application using Host APIs. This application connects to the light parent and tries to match the simple descriptors by using MatchDescriptor command.

After exchanging the simple descriptors, it will send the toggle command to the light continuously.

4.6.1.2 Host API Folder Structure

The folder structure for host API along with sample applications has been given below. This folder structure is available in the "release/utils_ZigBee" folder.

The folders in the "utils_ZigBee" folder are as follows:

1) apis – contains the core APIs and sample application
   a) core – contains the host mode API implementation.
   b) ref_apps – contains the reference HA switch application.
   c) build - contains Makefile to compile core and ref_apps irrespective of reference project.
2) reference_projects – contains code related to netlink sockets which is used to communicate with driver.

4.6.1.3 Building and Running the Home Automation Sample Application

1) Goto the folder /home/OneBox-Mobile/<interface>/utils_ZigBee/reference_projects/src
2) Clean the existing builds by giving the following command

```
#make clean
```

3) Build the Home Automation Switch application using the following command

```
#make switch
```

4) run the switch app by giving following command

```
#./rsi_wsc_zigb_app
```

4.7 Driver Uninstallation Procedure

The driver can be uninstalled along with the different modules using the scripts provided in the /home/OneBox-Mobile/<interface> folder.

1) remove_all.sh: Uninstalls the complete driver and all modules, including the common HAL modules.
2) wlan_remove.sh: Uninstalls the Wi-Fi specific modules only.
3) bt_remove.sh: Uninstalls the Bluetooth specific modules only.
4) zigb_remove.sh: Uninstalls the ZigBee specific modules only.

4.8 Driver Information

4.8.1 Driver Statistics

Use the command below to view driver statistics:

```
# cat /proc/onebox-halX/stats (Here X can be 0,1,2..)
```

This command prints statistics related to the total management packets, total data packets with respect to a given access category sent to/from the driver, buffer full status as well as semi buffer full status, FSM states etc.

4.8.2 Disabling Driver Debug Prints

You may opt to disable the debug prints of the driver appearing on the console using the command below. Ensure that the driver is installed correctly before using this command.

```
# echo 0x0 > /proc/onebox-halX/debug_zone
```

5 Evaluation of Connect-io-n®/WiSeConnect®

5.1 Introduction

The following table describes the available test methods for all supported interfaces on the RS9113 Evaluation board. It also explains the available software mechanisms to evaluate these.

INTERFACE	UART	SPI	USB	USB-CDC
Accepted Commands	• AT Commands • Binary Commands	• Binary Commands	• Binary Commands	• AT Commands • Binary Commands
Evaluation Options	1. Using Terminal Applications on Windows/Linux using AT commands or Binary commands. 2. Using ready to run Dev C++ IDE based projects on Windows using Binary Commands. 3. Using ready to run Reference projects on Linux.(Available as part of the release package)	1. Using Reference projects on the Renesas RX62N platform. (Available as part of the release package). 2. Using ready to run Reference projects on the Spansion Cortex M4 platform. (Available as part of the release package)	1. Ready to run Reference projects on Linux.(Available as part of the release package)	1. Using Terminal Application on Windows/Linux using AT commands or Binary Commands. 4. Using ready to run Dev C++ IDE based projects on Windows using Binary Commands. 2. Using ready to run Reference projects on Linux.(Available as part of the release package) 3. Performing ABRD in host interaction mode is must for USB CDC mode.
NOTE: For a complete list of examples available as part of the package, refer to Appendix B.				

5.2 Required Setup

The required setup is illustrated in figures in the sub-sections below for evaluation in the following modes:

1. Wi-Fi Client in Personal Security Mode
2. Wi-Fi Client in Enterprise Security Mode
3. Wi-Fi Direct Mode

4. Wi-Fi Access Point Mode
5. BLE Peripheral Mode
6. BLE Central Mode
7. BT Master Mode
8. BT Slave Mode
9. Wi-Fi Client + BLE Peripheral Mode
10. Wi-Fi Client + BLE Central Mode
11. Wi-Fi AP + BLE Peripheral Mode
12. Wi-Fi AP + BLE Central Mode

NOTE: This section describes the evaluation process of the RS9113 modules using the UART interface. To learn about the APIs for SPI, USB, and USB-CDC, please view the reference projects and associated readmes included in the release package.

5.2.1 Third Party Tools

The following third party tools maybe be required during your evaluation.

1. Docklight
2. SPP Pro Mobile Application
3. LightBlue Mobile Application
4. Bluesoleil Windows Bluetooth Application

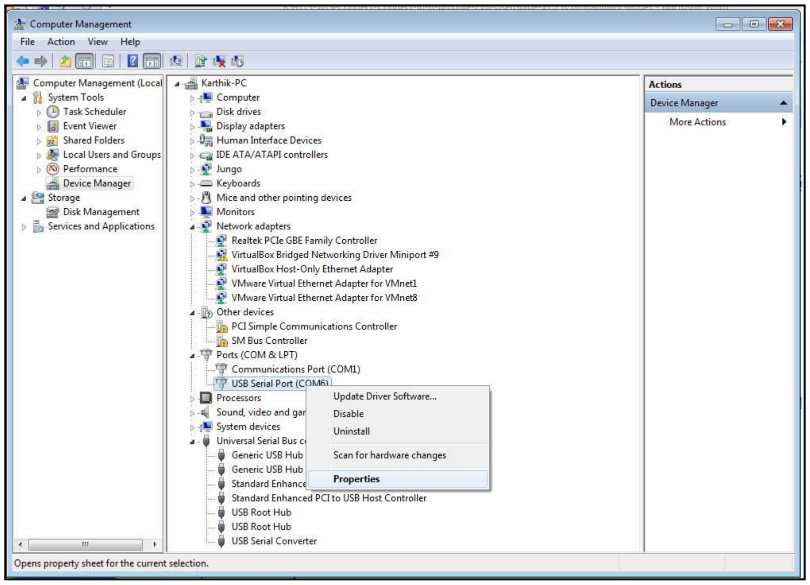
5.3 Getting Started

5.3.1 Installing Virtual COM Port Drivers

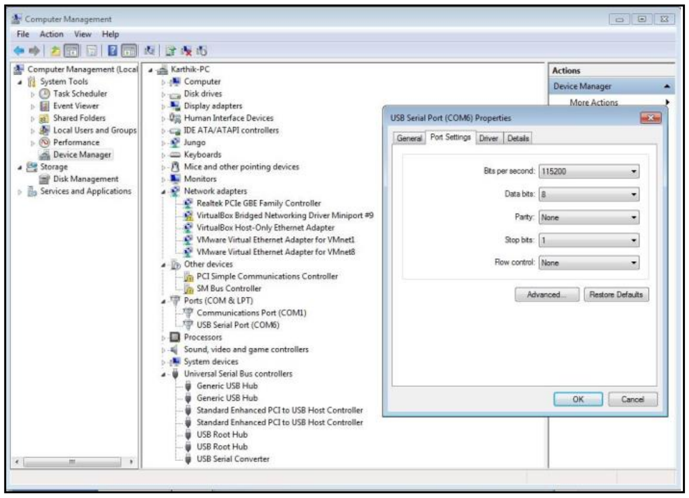
For UART communication, an FT232R USB-UART bridge is used on the EVB. This will enable a PC host to interface with the device using a USB cable.

Before working with the board, install virtual COM port drivers on the PC host. Detailed installation instructions are available for download from FTDI. The following link directs to the installation guides specific to PC operating systems.

http://www.ftdichip.com/Support/Documents/InstallGuides.htm After driver installation is done as per the guide, ensure the FT232R chip is configured to match with the default baud settings of the EVB. Baud rate setting and serial port settings can be configure from device manager as shown in the following screen-shots. Navigate to the Device manager-> USB Serial Port(COMx) -> right-click and select “Properties.”.



In the “USB Serial Port (COMx)” window, select “Port Settings” tab as shown in the screen-shot below.



Bits per second: 115200
Data bits: 8
Parity: None
Stop bits: 1
Flow control: None

Configure the “Port Settings” as mentioned below.

Baud rate: 115200 baud

Data bits: 8 bits

Parity: None

Stop bits: 1

Flow Control: None

5.3.2 Installing Tools to evaluate over the UART interface

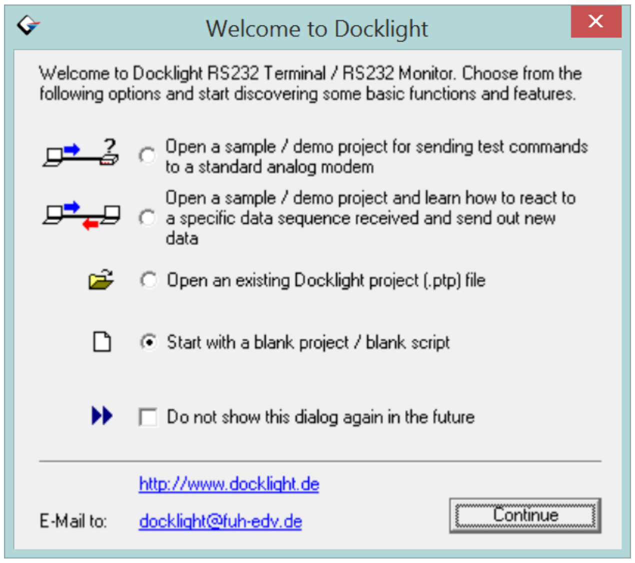
AT Mode over UART interface: Installing and Using Docklight on Windows

Binary Mode over UART interface: Installing and Using Development C++ on Windows

5.3.2.1 Installing and Using Docklight on Windows

Please follow the steps below to get started with the evaluation process. This process is explained for evaluation on a Windows PC.

1) Download and install a Serial Emulation program like Docklight, Teraterm, etc. This document uses Docklight for explaining the process. This software is relatively easy to use and allows the user to monitor the serial port data while also enabling sending and receiving data. However, since it does not support sending of files using Kermit, it cannot be used to upgrade the firmware of the module. For firmware upgradation, we suggest Teraterm. The process for firmware upgradation is explained in Appendix C. Docklight can be downloaded from http://www.docklight.de/download_en.htm
2) Open Docklight and click OK when asked for Registration, to use it in Evaluation mode.
3) In the dialog box that opens, select “Start with a blank project / blank script” and click Continue.


Welcome to Docklight

Welcome to Docklight RS232 Terminal / RS232 Monitor. Choose from the following options and start discovering some basic functions and features.

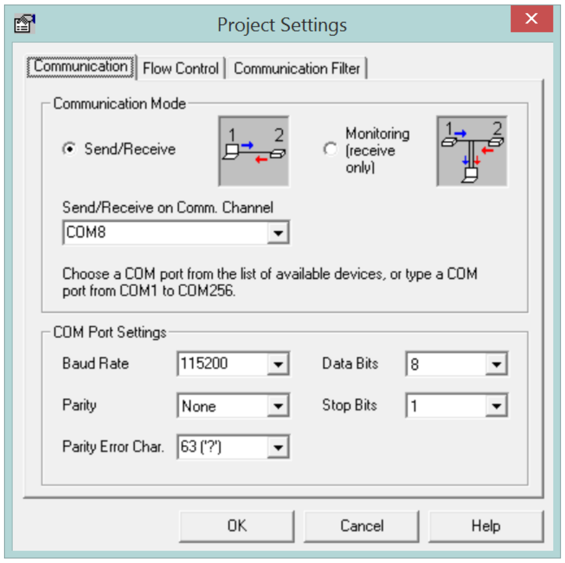
- ( ) Open a sample / demo project for sending test commands to a standard analog modem
- ( ) Open a sample / demo project and learn how to react to a specific data sequence received and send out new data
- ( ) Open an existing Docklight project (.ptp) file
- [✓] Start with a blank project / blank script
- [ ] Do not show this dialog again in the future

http://www.docklight.de

E-Mail to: docklight@fuh-edv.de

Figure 3: Docklight Startup Dialog Box

4) Next, connect the EVB to the PC using the Micro A/B-type USB cable. Plug in the cable into the micro-USB port labeled “UART” on the EVB.
5) Wait for the drivers to be installed the first time. Disconnect the EVB and connect it again.
6) In Docklight, click on Tools -> Project Settings to open the dialog box shown below.



Project Settings

Communication | Flow Control | Communication Filter

Communication Mode

(●) Send/Receive

( ) Monitoring (receive only)

Send/Receive on Comm. Channel: COM8

Choose a COM port from the list of available devices, or type a COM port from COM1 to COM256.

COM Port Settings

Baud Rate	115200	Data Bits	8
Parity	None	Stop Bits	1
Parity Error Char.	63 ('?')		

Figure 4: Docklight Project Settings Dialog Box

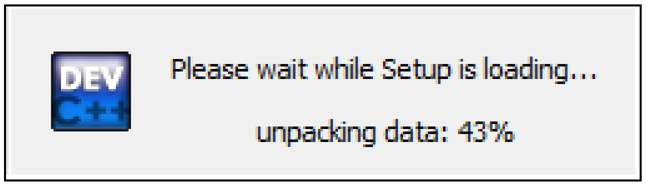
7) Select the following options for the Serial port settings:
   a. Select the COM port in the drop-down menu under "Send/Receive on Comm. Channel". In the above figure, it is COM8.
   b. Select the Baud Rate as 115200.
   c. Click OK.
8) Disconnect the EVB and follow the steps in the sub-sections below.

5.3.2.2 Installing and Using Development C++ on Windows

Recommended Version of Dev C++: **5.6.3**

Recommended Tool chain: **dev-cpp_5.6.3_mingw_4.8.1.exe**

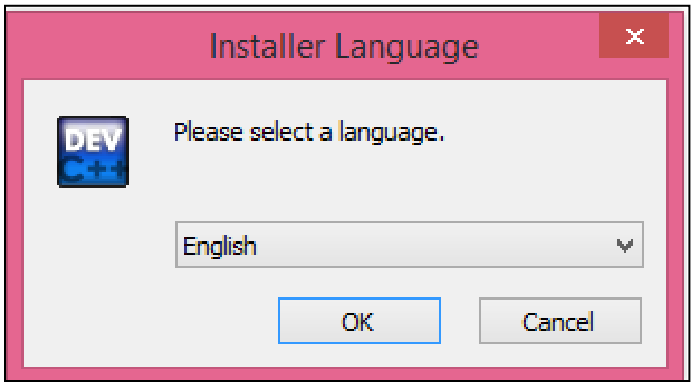
1. Download and extract the Dev-C++ package.
2. Run the set up file and wait until it loads

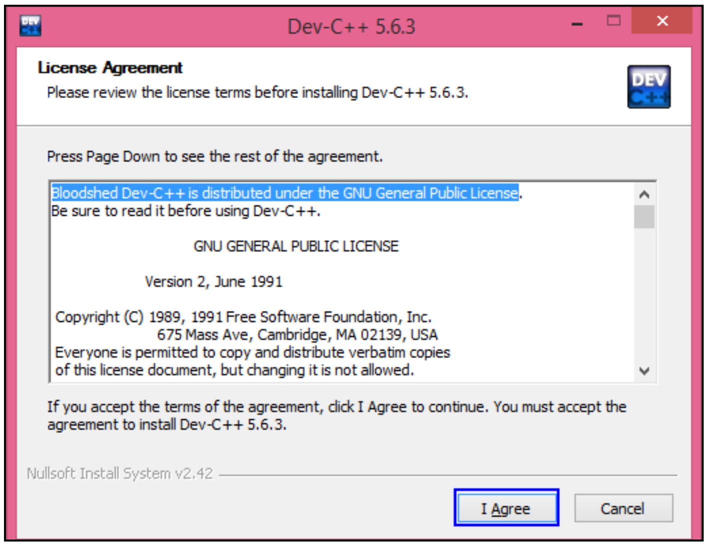


Please wait while Setup is loading...

unpacking data: 43%

3. Select the language and accept the terms and conditions.





License Agreement
Please review the license terms before installing Dev-C++ 5.6.3.

Press Page Down to see the rest of the agreement.

Bloodshed Dev-C++ is distributed under the GNU General Public License.
Be sure to read it before using Dev-C++.

GNU GENERAL PUBLIC LICENSE

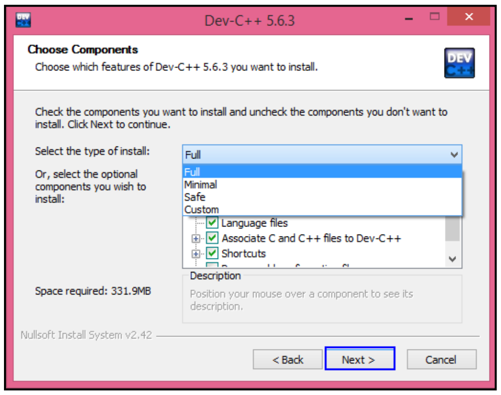
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
675 Mass Ave, Cambridge, MA 02139, USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Dev-C++ 5.6.3.

Nullsoft Install System v2.42

4. Choose the type of install as 'full' and proceed



Choose Components
Choose which features of Dev-C++ 5.6.3 you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select the type of install: Full

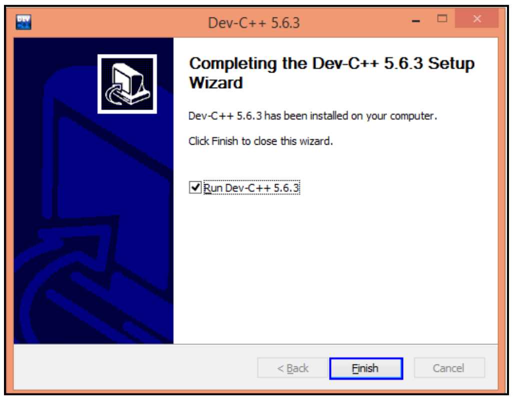
Or, select the optional components you wish to install:

Space required: 331.9MB

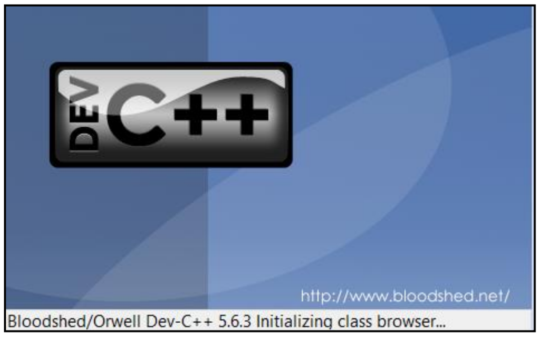
Description
Position your mouse over a component to see its description.

Nullsoft Install System v2.42

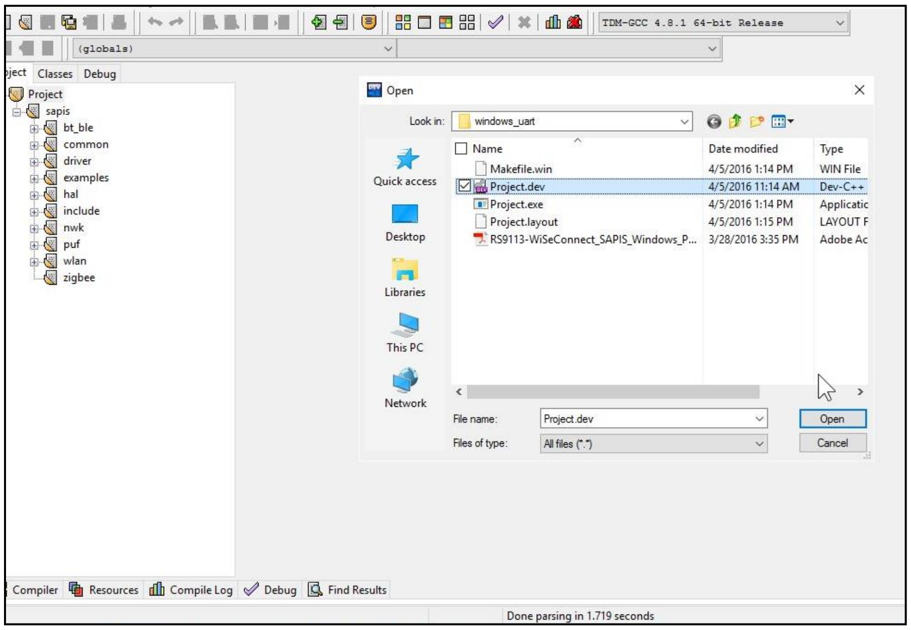
5. Wait until all components are installed to click on Finish


Dev-C++ 5.6.3
Completing the Dev-C++ 5.6.3 Setup Wizard
Dev-C++ 5.6.3 has been installed on your computer.
Click Finish to close this wizard.
Run Dev-C++ 5.6.3
< Back
Finish
Cancel

6. Once installation is complete, the IDE loads automatically.


DEV C++
http://www.bloodshed.net/
Bloodshed/Orwell Dev-C++ 5.6.3 Initializing class browser...

Configuring the IDE: After successful installation of Dev-C++ IDE. Open configuration settings file 'Project' from the sapis directory located in the RS9113.NBZ.WC.GEN.OSI.1.4.1\host\sapis\platforms\windows_uart\ directory and select the project.dev file. The following screenshot shows selecting the correct project.dev file.



Project: sapis — bt_ble, common, driver, examples, hal, include, nwk, puf, wlan, zigbee

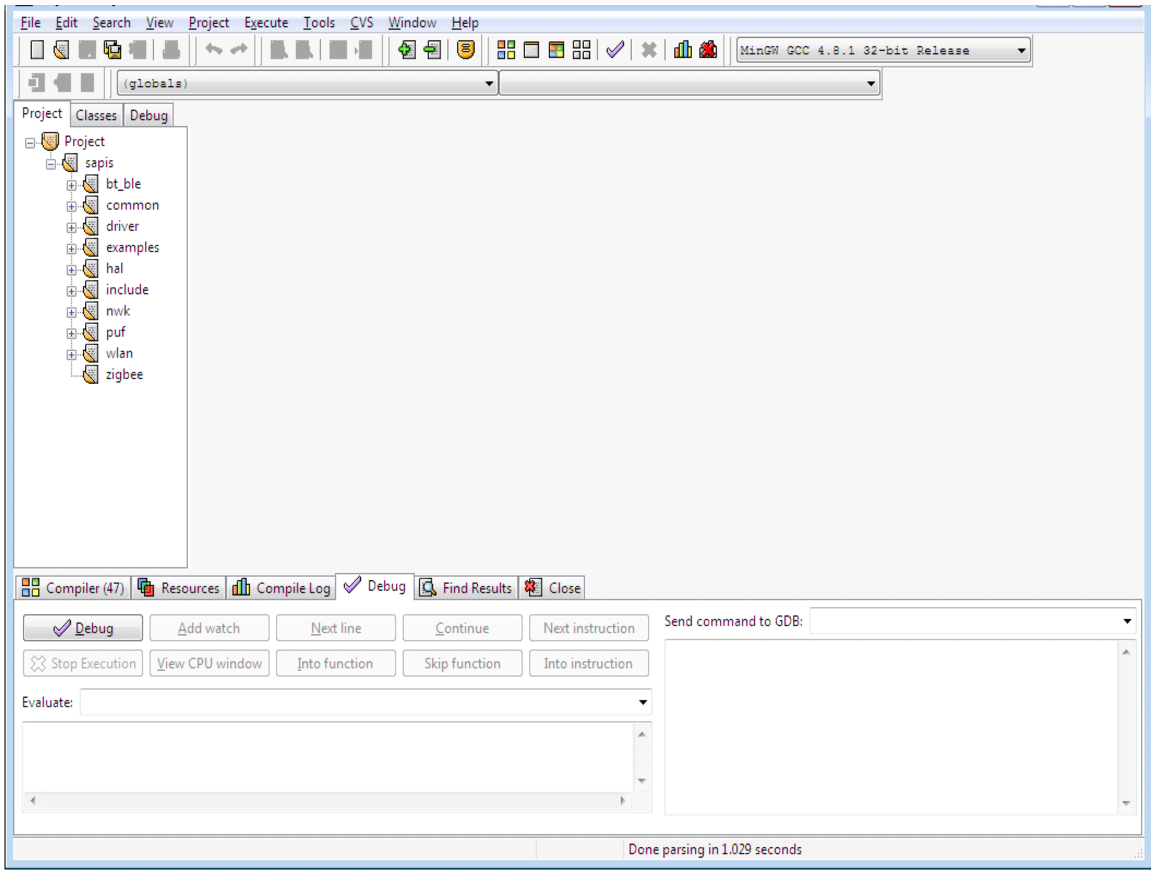
Open — Look in: windows_uart

Name	Date modified	Type
Makefile.win	4/5/2016 1:14 PM	WIN File
Project.dev	4/5/2016 11:14 AM	Dev-C++
Project.exe	4/5/2016 1:14 PM	Applicatic
Project.layout	4/5/2016 1:15 PM	LAYOUT F
RS9113-WiSeConnect_SAPIS_Windows_P...	3/28/2016 3:35 PM	Adobe Ac

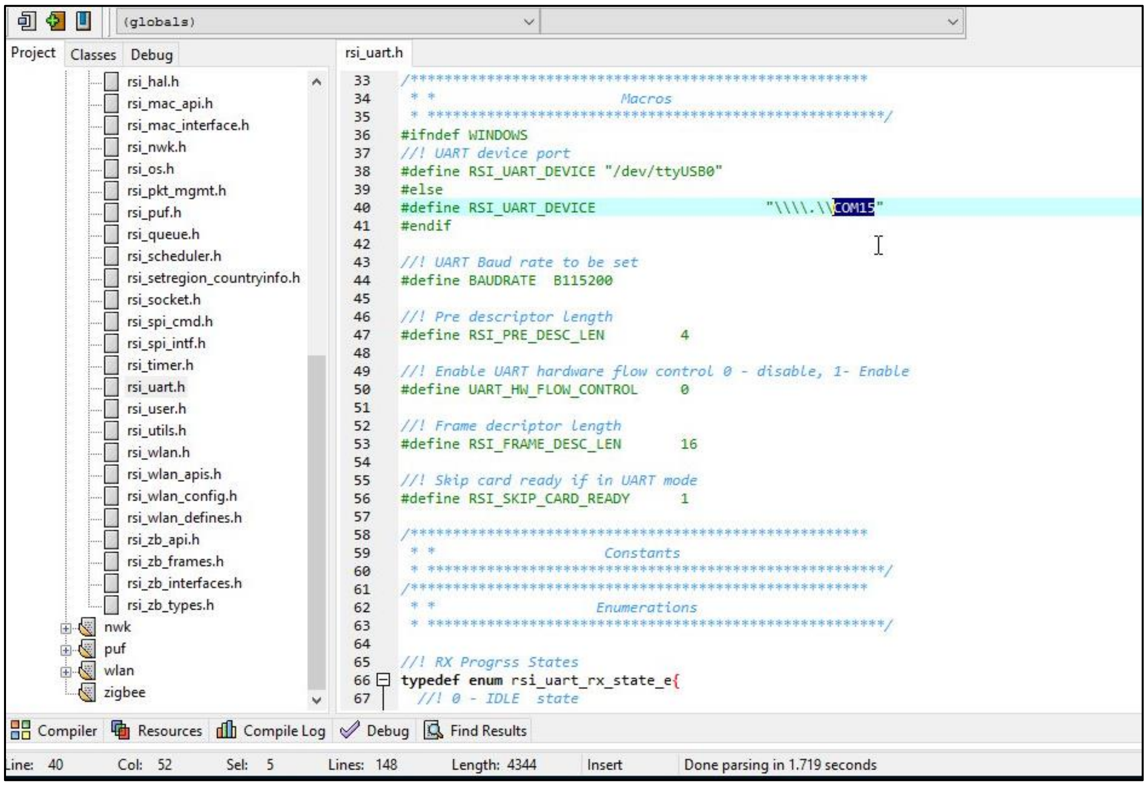
File name: Project.dev
Files of type: All files (*.*)

Done parsing in 1.719 seconds

Once the project.dev file is opened you will see the project file “sapis” with all of its sub directories.


File Edit Search View Project Execute Tools CVS Window Help
MinGW GCC 4.8.1 32-bit Release
(globals)
Project Classes Debug
Project
sapis
bt_ble
common
driver
examples
hal
include
nwk
puf
wlan
zigbee
Compiler (47) Resources Compile Log Debug Find Results Close
Debug Add watch Next line Continue Next instruction
Stop Execution View CPU window Into function Skip function Into instruction
Evaluate:
Send command to GDB:
Done parsing in 1.029 seconds

Configure UART: Change the macro definition of ‘RSI_UART_DEVICE’ available in sapis/include/rsi_uart.h, replacing ‘COM97’ with com-port number which the device got registered on the PC in system device manager. Macro is available in sapis/include/rsi_uart.h

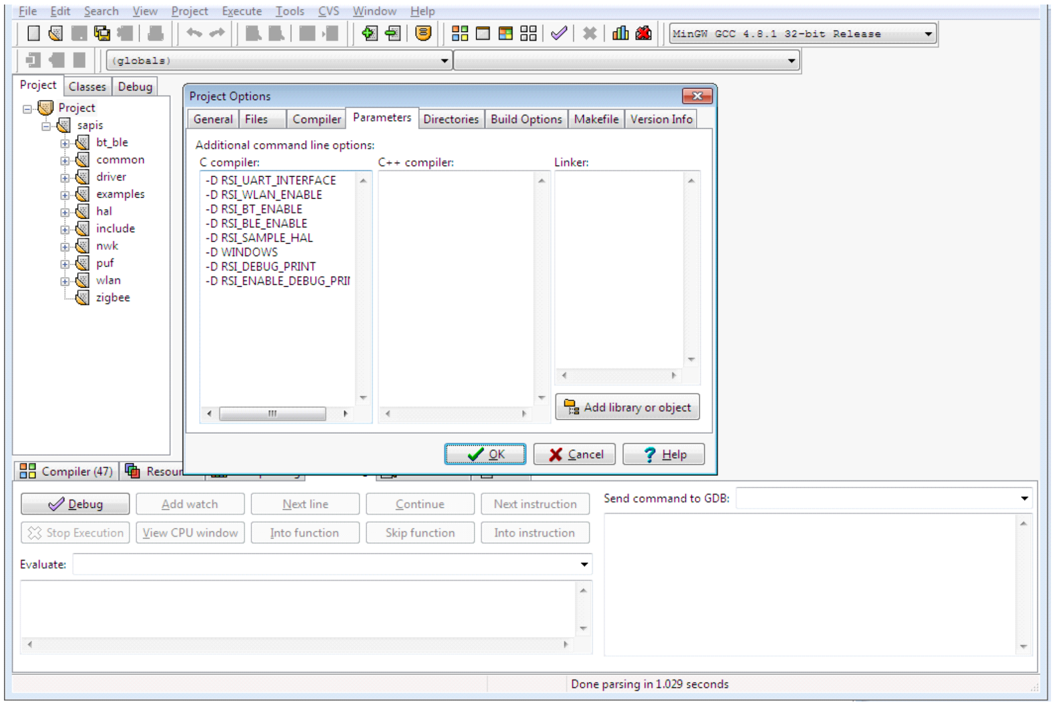


```
/*********************************************************
 * *                     Macros
 * ********************************************************/
#ifndef WINDOWS
//! UART device port
#define RSI_UART_DEVICE "/dev/ttyUSB0"
#else
#define RSI_UART_DEVICE                    "\\\\.\\COM15"
#endif

//! UART Baud rate to be set
#define BAUDRATE  B115200

//! Pre descriptor length
#define RSI_PRE_DESC_LEN          4

//! Enable UART hardware flow control 0 - disable, 1- Enable
#define UART_HW_FLOW_CONTROL      0

//! Frame decriptor length
#define RSI_FRAME_DESC_LEN        16

//! Skip card ready if in UART mode
#define RSI_SKIP_CARD_READY       1

/*********************************************************
 * *                   Constants
 * ********************************************************/
/*********************************************************
 * *                  Enumerations
 * ********************************************************/

//! RX Progrss States
typedef enum rsi_uart_rx_state_e{
  //! 0 - IDLE  state
```

Line: 40 Col: 52 Sel: 5 Lines: 148 Length: 4344 Insert Done parsing in 1.719 seconds

You can verify which Serial port by opening the system device manager from the control panel

Configure Macros: To add macros for compilation open ‘Project Settings’ from ‘Project’ tab. Then add Macros in ‘parameters’ sub tab under ‘C Complier’ Options, also shown in following screenshot.



Additional command line options:

C compiler:

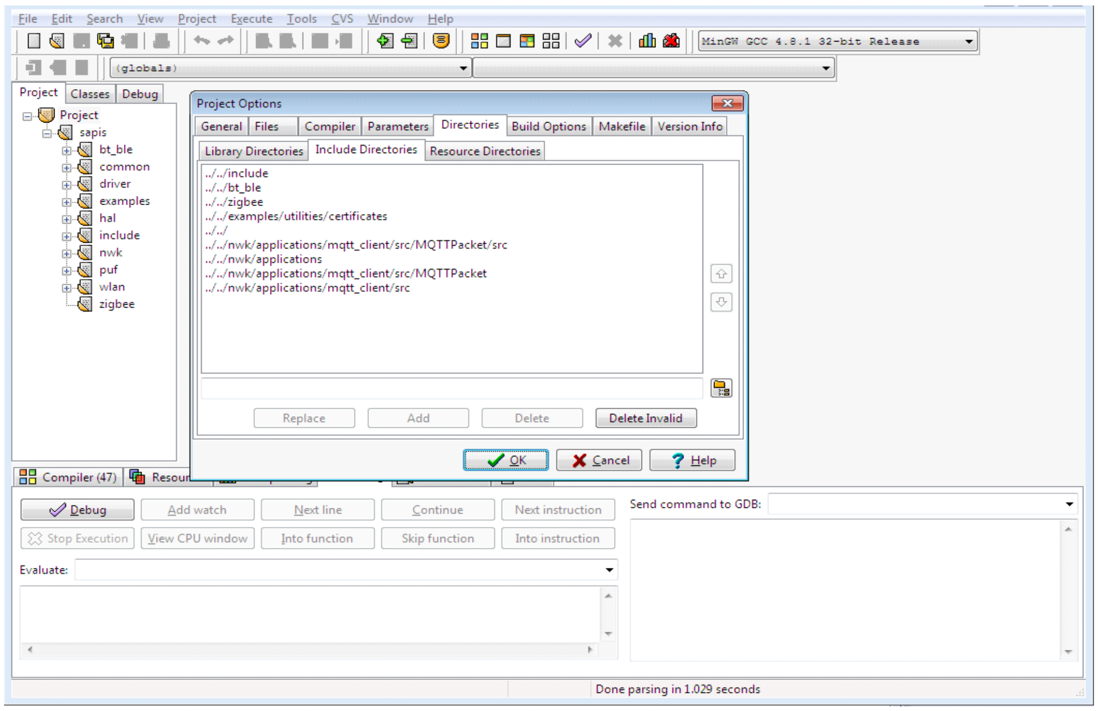
```
-D RSI_UART_INTERFACE
-D RSI_WLAN_ENABLE
-D RSI_BT_ENABLE
-D RSI_BLE_ENABLE
-D RSI_SAMPLE_HAL
-D WINDOWS
-D RSI_DEBUG_PRINT
-D RSI_ENABLE_DEBUG_PRI
```

C++ compiler:

Linker:

Done parsing in 1.029 seconds

Configure Include paths: In order to add include paths, move to 'Directories' sub tab in 'Project Options' window, add include paths as shown below.



```
../../include
../../bt_ble
../../zigbee
../../examples/utilities/certificates
../../
../../nwk/applications/mqtt_client/src/MQTTPacket/src
../../nwk/applications
../../nwk/applications/mqtt_client/src/MQTTPacket
../../nwk/applications/mqtt_client/src
```

5.3.3 Check the Firmware version running on your Evaluation Board

Once you have the drivers installed and are ready to use the Evaluation board, we recommend you to check the firmware version running on the board. Ensure to verify that this is the latest and the greatest firmware released by Silicon Labs.

AT Command: `at+rsi_fwversion?/r/n`

Binary API: `rsi_wlan_get(RSI_FW_VERSION,*response, sizeof(response));`

5.3.4 Introduction to Command Types

5.3.4.1 AT+ Commands

The Wi-Fi AT command set represents the frames that are sent from the Host to operate the RS9113-WiSeConnect Module. The command set resembles the standard AT command interface used for modems.
All AT commands start with "at" and are terminated with a carriage return('\r') and a new line('\n') character.

The AT command set for the RS9113-WiSeConnect Module starts with "at+rsi_" followed by the name of the command and any relevant parameters.

In some commands, a '?' character is used after the command to query data from the module.

[2] Contact your Account Manager for login credentials to the server.

5.3.4.2 Binary Commands

The Wi-Fi configuration and operation commands are sent to the module and the responses are read from the module using frame write/frame read (as mentioned in the preceding sections) so these configuration and operation commands are called as command frames. The command frame is categorized as management or data frames. The management frames are used to configure the Wi-Fi module to access Wi-Fi connectivity, TCP/IP stack and operate the module. Data frames are used to send the data. Management and data frames are exchanged between host and module. Management frame is sent from Host to the module to configure the module, and also is sent from module to host to send responses to these commands.

5.4 Wi-Fi Evaluation in UART Mode

In UART Mode, the supported test methods for Wi-Fi are

- Using AT Commands via a terminal Utility
- Using binary commands via Dev C++ IDE

The following sub-sections describe configuring and using the module in different security and operational modes of Wi-Fi.

5.4.1 Wi-Fi Evaluation using AT Commands

5.4.1.1 Wi-Fi Client in Personal Security Mode

The figure below shows the setup required for this process.

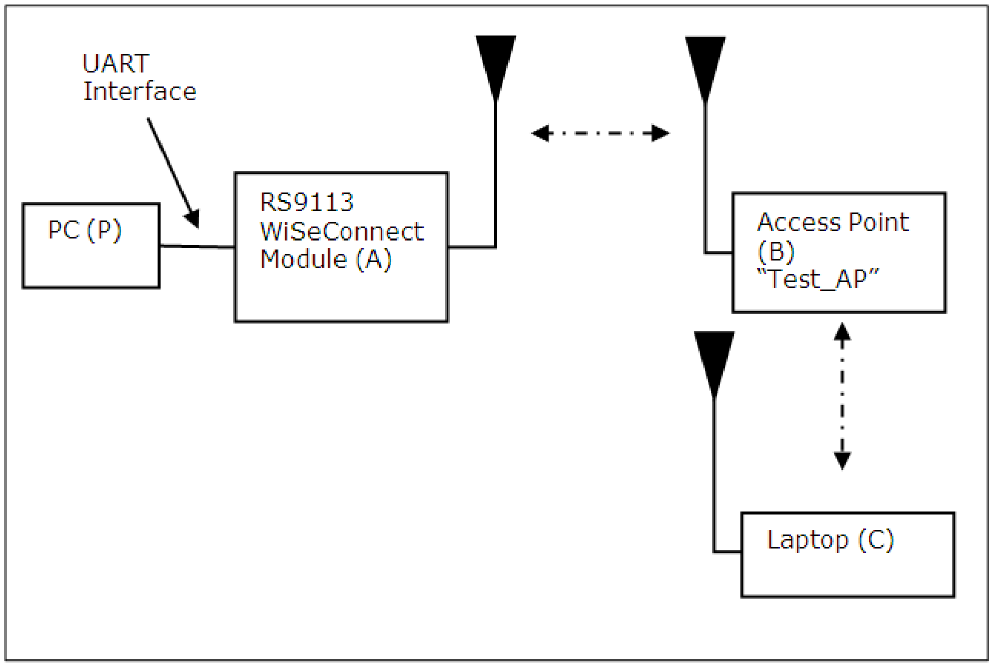
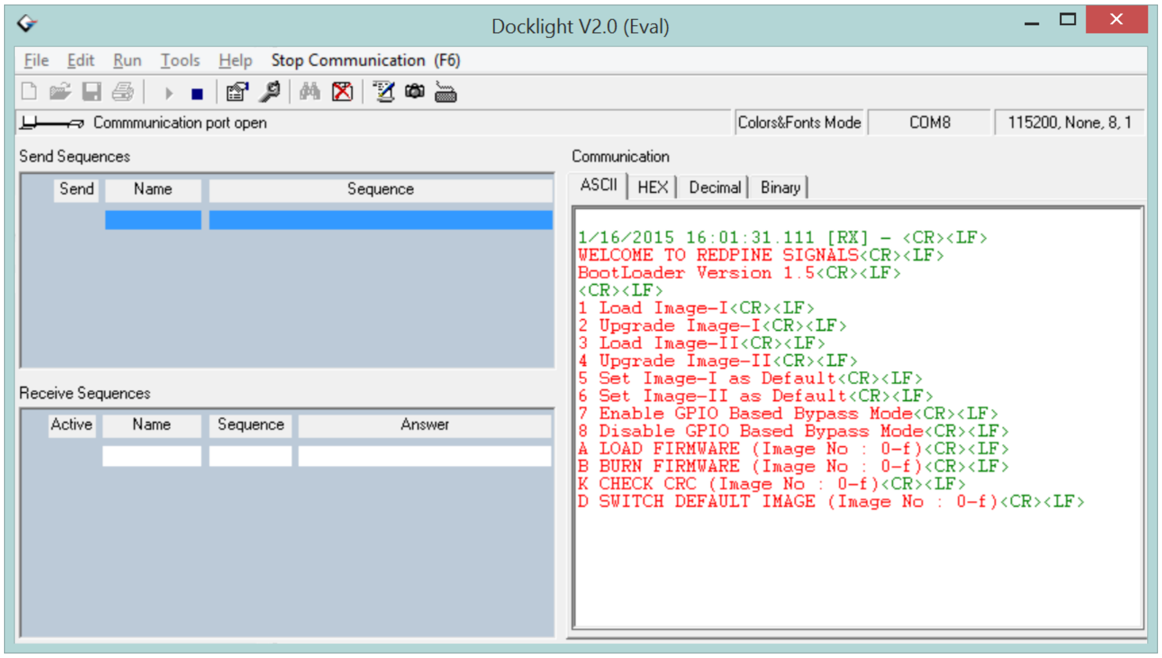


Figure 5: Setup for Wi-Fi Client in Personal Security Mode

In the setup shown in the figure, the RS9113 WiSeConnect/Connect-io-n EVB is a Wi-Fi client. It connects to an Access Point configured in WPA2-PSK security mode. The Access Point is configured with SSID as "Test_AP" and IP address set as 192.168.50.1.The SSID and IP address are for illustration only. The user is required to configure the AP based on network domain in which wireless devices operate.

5.4.1.1.1 Wi-Fi Client Configuration

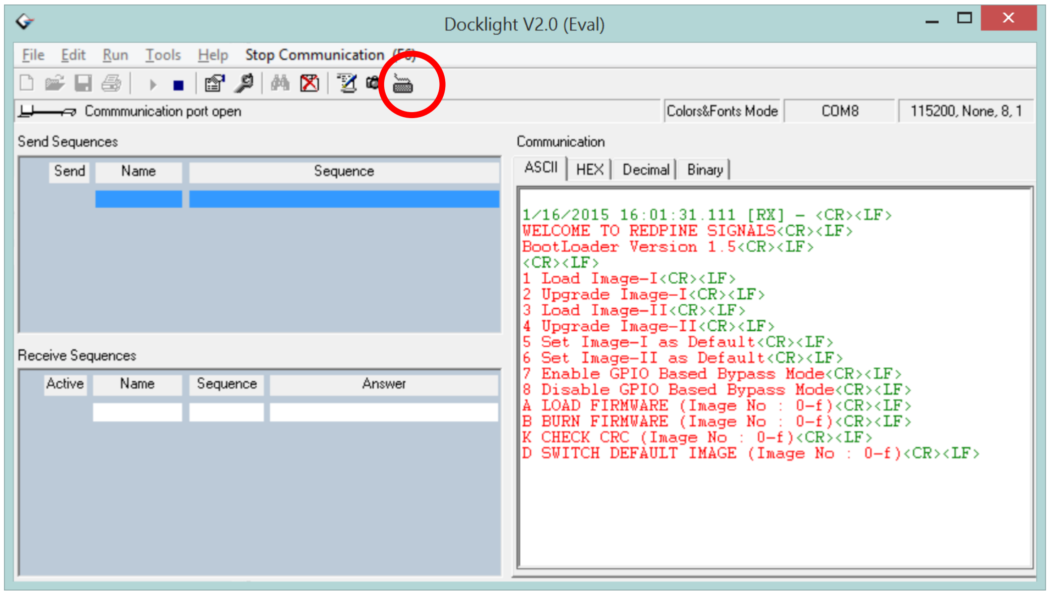
1) Open Docklight.
2) Connect the Micro A/B-type USB cable between the USB port of the PC and the micro-USB port of the EVB labeled "UART".
3) Hit "F5" key when in the Docklight window to start communicating with EVB.
4) At powerup, the module tries to estimate the baud rate of the Host by exchanging data with it. If no data is sent by the Host, the module defaults to the 115200 bps baud rate in approximately 18 seconds. Please refer to the Software PRM document to understand the Automatic Buad Rate Detection Process for faster bootup. At the end of the timeout, you will see a message like, "WELCOME TO REDPINE SIGNALS", followed by multiple options. Please see the figure below.



```
1/16/2015 16:01:31.111 [RX] - <CR><LF>
WELCOME TO REDPINE SIGNALS<CR><LF>
BootLoader Version 1.5<CR><LF>
<CR><LF>
1 Load Image-I<CR><LF>
2 Upgrade Image-I<CR><LF>
3 Load Image-II<CR><LF>
4 Upgrade Image-II<CR><LF>
5 Set Image-I as Default<CR><LF>
6 Set Image-II as Default<CR><LF>
7 Enable GPIO Based Bypass Mode<CR><LF>
8 Disable GPIO Based Bypass Mode<CR><LF>
A LOAD FIRMWARE (Image No : 0-f)<CR><LF>
B BURN FIRMWARE (Image No : 0-f)<CR><LF>
K CHECK CRC (Image No : 0-f)<CR><LF>
D SWITCH DEFAULT IMAGE (Image No : 0-f)<CR><LF>
```

Figure 6: Module Startup Messages

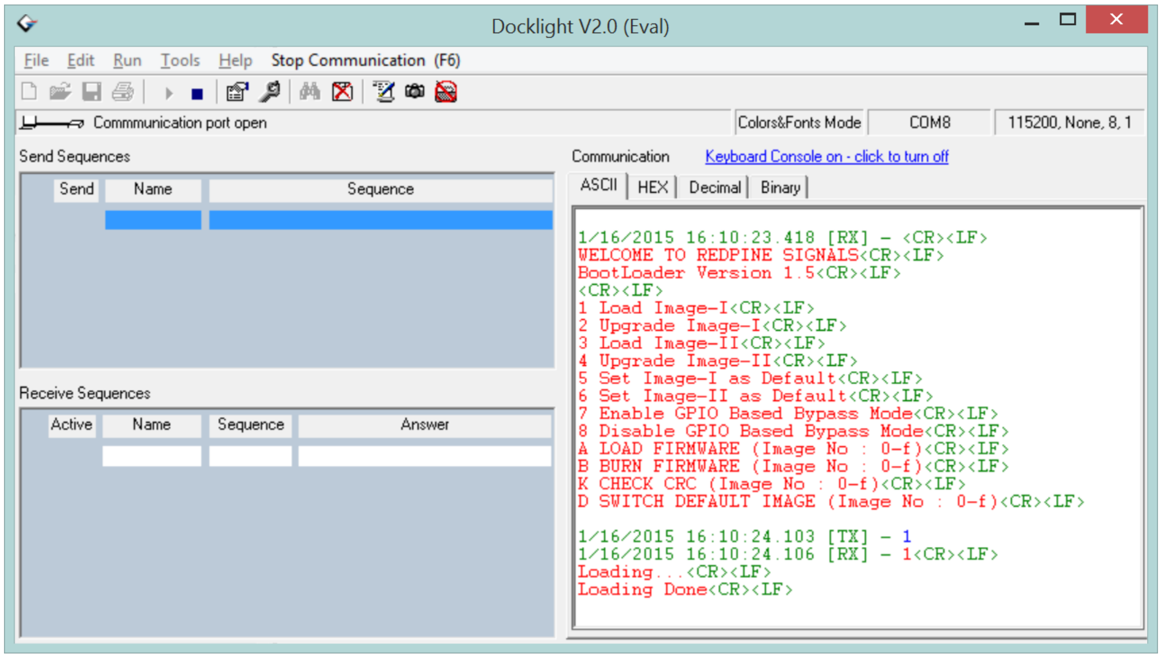
5) Click on the "Keyboard Console On/Off" button as shown in the figure below to enable keyboard inputs to Docklight.



```
1/16/2015 16:01:31.111 [RX] - <CR><LF>
WELCOME TO REDPINE SIGNALS<CR><LF>
BootLoader Version 1.5<CR><LF>
<CR><LF>
1 Load Image-I<CR><LF>
2 Upgrade Image-I<CR><LF>
3 Load Image-II<CR><LF>
4 Upgrade Image-II<CR><LF>
5 Set Image-I as Default<CR><LF>
6 Set Image-II as Default<CR><LF>
7 Enable GPIO Based Bypass Mode<CR><LF>
8 Disable GPIO Based Bypass Mode<CR><LF>
A LOAD FIRMWARE (Image No : 0-f)<CR><LF>
B BURN FIRMWARE (Image No : 0-f)<CR><LF>
K CHECK CRC (Image No : 0-f)<CR><LF>
D SWITCH DEFAULT IMAGE (Image No : 0-f)<CR><LF>
```

Figure 7: Keyboard Console On

6) Hit ‘1’ and press Enter. You will see a message that says, “Loading…” followed by “Loading Done”.



```
1/16/2015 16:10:23.418 [RX] - <CR><LF>
WELCOME TO REDPINE SIGNALS<CR><LF>
BootLoader Version 1.5<CR><LF>
<CR><LF>
1 Load Image-I<CR><LF>
2 Upgrade Image-I<CR><LF>
3 Load Image-II<CR><LF>
4 Upgrade Image-II<CR><LF>
5 Set Image-I as Default<CR><LF>
6 Set Image-II as Default<CR><LF>
7 Enable GPIO Based Bypass Mode<CR><LF>
8 Disable GPIO Based Bypass Mode<CR><LF>
A LOAD FIRMWARE (Image No : 0-f)<CR><LF>
B BURN FIRMWARE (Image No : 0-f)<CR><LF>
K CHECK CRC (Image No : 0-f)<CR><LF>
D SWITCH DEFAULT IMAGE (Image No : 0-f)<CR><LF>

1/16/2015 16:10:24.103 [TX] - 1
1/16/2015 16:10:24.106 [RX] - 1<CR><LF>
Loading...<CR><LF>
Loading Done<CR><LF>
```

Figure 8: Firmware Loading Messages

7) The following commands can now be issued to the module. Please refer to the Software PRM document for detailed description of each command and the expected responses. A command should not be sent until the response for the previous command is received.

NOTE: Each AT command should have a suffix of Carriage Return <CR> (Keyboard keys are Ctrl+Enter) and Line Feed <LF> (Keyboard keys are Ctrl+Shift+Enter). This is enabled by default in Docklight.

a. `at+rsi_opermode=0,1,4,0`

This command configures the module as Wi-Fi client. The module responds with "OK"

b. `at+rsi_band=0`

This command configures the operating band of Wi-Fi client to 2.4GHz. The module responds with "OK"

c. `at+rsi_init`

This command initializes the Wi-Fi module in the EVB. The module responds with OK<MAC_Address>.

d. `at+rsi_fwversion?`

This command displays current firmware version in use.

e. `at+rsi_scan=0`

This command scans for available Access Points operating in the 2.4 GHz band. The module responds with information of the Access Points scanned. The data received might have some unreadable characters because of ASCII conversion. You can use the HEX tab of Docklight to see the bytes sent by the module.

f. `at+rsi_psk=1,12345678`

This command configures the PSK of the Wi-Fi client with the key entered by user. The User can enter any other key as PSK.

g. `at+rsi_join=Test_AP,0,2,2`

This command connects the Wi-Fi client to the Access Point with SSID "Test_AP". On successful association, the module responds with OK.

h.
```
at+rsi_ipconf=0,192.168.50.10,255.255.255.0,192.168.50.1
at+rsi_ipconf=1,0,0,0
```

This command configures the IP address of the module. The IP address configured in the above command is for illustration only. The user has to configure the IP address as per the Access Point's settings using either Manual (first command above) or DHCP mode (second command above).

For the Manual mode, ensure that the desired IP is in the same subnet as the Access Point's subnet. The module responds to this command by sending the configured IP address to the Host. In the terminal, this response might appear as unreadable characters because of ASCII conversion. You can use the HEX tab of Docklight to see the bytes sent by the module.

5.4.1.1.2 Test Procedure

The IP addresses configured in this process are meant for illustration only. It is assumed that Wi-Fi client of the EVB is assigned an IP address of 192.168.50.10 and Laptop C, connected to the Access Point is assigned an IP address of 192.168.50.20.

The applications provided in the USB drive (as part of the release package) send and receive TCP and UDP packets.

TCP and UDP applications are provided along with release package for execution on the Laptop. These applications are located in the path RS9113.NBZ.WC.GEN.OSI.x.x.x\utils\peer_applications\Windows where RS9113.WSC.GENR.x.x.x is software package directory.

1) Open a TCP Server socket on the Wi-Fi Client (EVB) side using the following AT command:

   ```
   at+rsi_ltcp=5001,1,0
   ```

2) The module's response will look as follows:

   ```
   OK<ip_version><socket_type><socket_handle><Lport><module_ip
   addr>\r\n
   ```

   The `socket_handle` in the response above is used for subsequent commands.

3) Open a TCP client socket on Laptop C by running the TCP.exe application as follows in the Windows Command Prompt:

   ```
   TCP.exe c 2001 192.168.50.10 5001
   ```

   Ensure that you run the Windows Command Prompt program as an Administrator and any firewalls which block creation of sockets are disabled before running the application.

4) The above command opens a new window with the following options:
   a. Send
   b. Receive
   c. Exit

5) Observe that the Docklight on the Wi-Fi Client side (EVB) prints the following message, once the TCP connection is setup with Laptop C.

   ```
   AT+RSI_LTCP_CONNECT=<ip_version><socket_descriptor><dest_po
   rt_no><dest_ipaddr><mss><window_size><src_port_no>\r\n
   ```

6) For testing the Receive mode of the Wi-Fi Client (EVB), follow the steps below:
   a. Type 1 in the TCP.exe window on Laptop C.
   b. On being prompted to enter a data string to be transmitted, type any string and hit Enter to transmit the typed data.
   c. When the data is received on the Wi-Fi Client (EVB) side, you will see a response (asynchronous) from the module as follows:

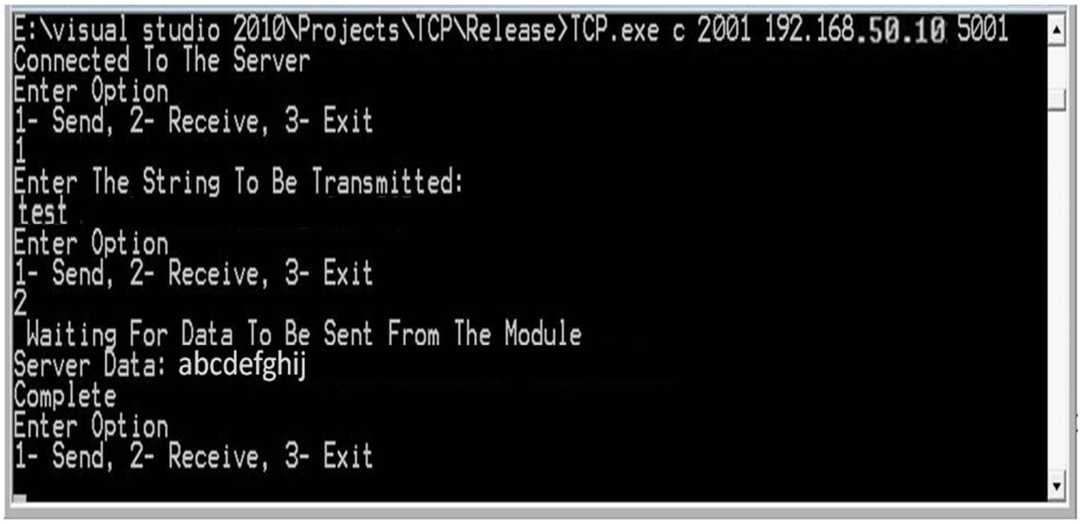
      ```
      AT+RSI_READ<ip_version><socket_handle><payload_len><s
      ource_ip_addr><source_port><payload>
      ```

7) For testing the Transmit mode of the Wi-Fi Client (EVB), follow the steps below:
   a. Type 2 in the TCP.exe window on Laptop C.
   b. On the Wi-Fi Client (EVB) side, send the command below to transmit data to the Laptop C.

```
at+rsi_snd=<socket_handle>,<payload_len>,<dest_ipaddr
>,<dest_port>,<payload >
```

c. The transmitted data is displayed on the TCP.exe window of Laptop C.

8) The figure below is a snapshot of the TCP.exe application window.



```
E:\visual studio 2010\Projects\TCP\Release>TCP.exe c 2001 192.168.50.10 5001
Connected To The Server
Enter Option
1- Send, 2- Receive, 3- Exit
1
Enter The String To Be Transmitted:
test
Enter Option
1- Send, 2- Receive, 3- Exit
2
 Waiting For Data To Be Sent From The Module
Server Data: abcdefghij
Complete
Enter Option
1- Send, 2- Receive, 3- Exit
```

Figure 9: TCP Application Window

5.4.1.2 Wi-Fi Client in Enterprise Security Mode

The figure below shows the setup required for this process.

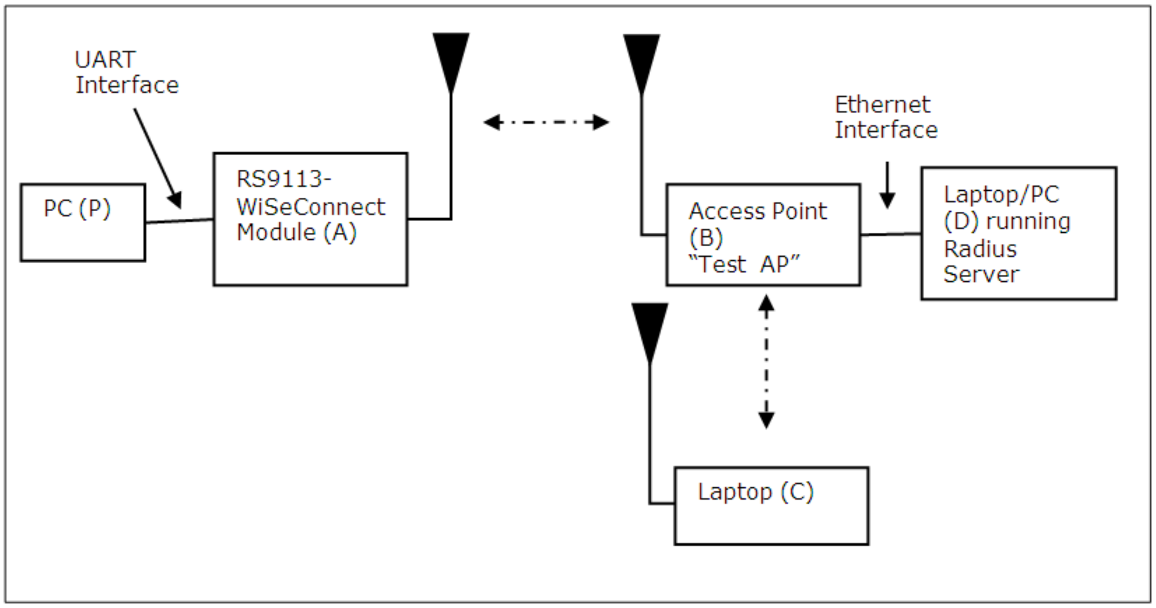

UART Interface
PC (P)
RS9113-WiSeConnect Module (A)
Ethernet Interface
Access Point (B) "Test AP"
Laptop/PC (D) running Radius Server
Laptop (C)

Figure 10: Setup for Wi-Fi Client in Enterprise Security Mode

In the setup shown in the above figure, the WiSeConnect® module on the RS9113 EVB acts as Wi-Fi client. It connects to an Enterprise Security enabled Access Point.The WiSeConnect® module supports four Enterprise Security modes:

1) EAP-TLS
2) EAP-TTLS
3) EAP-PEAP
4) EAP-FAST

5.4.1.2.1 Radius Server Configuration

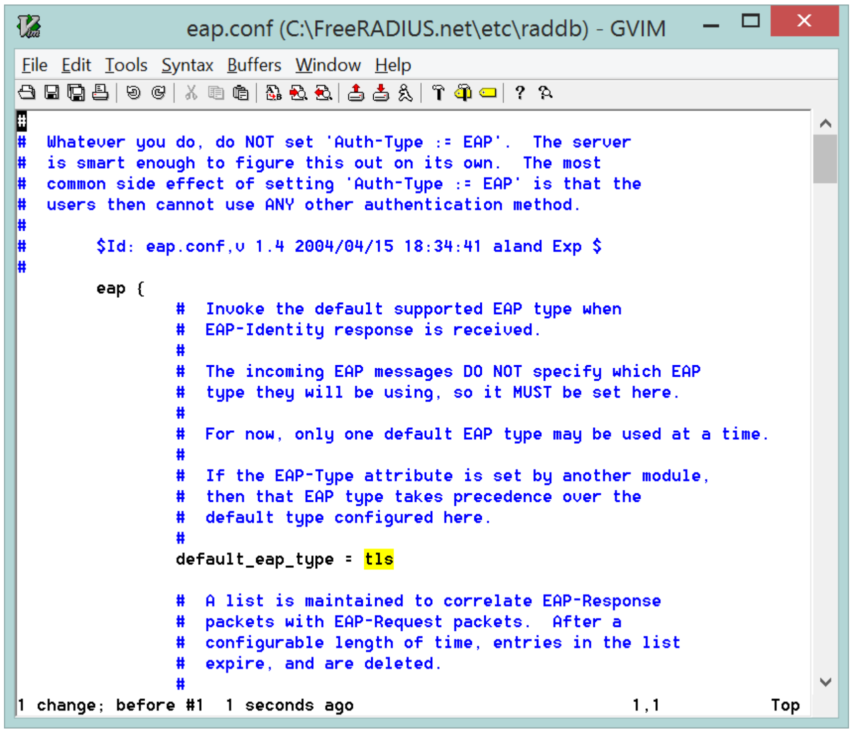
Follow the steps below to setup a Radius Server on Laptop D as per the setup shown in Figure 10. The process explained here is for a Windows Laptop. A similar process may be followed for other OS.

1) Download and install the FreeRADIUS server version 2.2.3 or above. The FreeRADIUS server software is available at http://freeradius.org/download.html .
2) Once installed, go to the C:\FreeRADIUS.net\etc\raddb folder and make the following modifications.
3) Open the clients.conf file and add the following lines at the end of the file.

```
client 192.168.50.1/24 {
    secret      =     12345678
    shortname   =     private-network-1
}
```

The IP address in the above lines (192.168.50.1) is the IP address of the Access Point in this example setup. The “12345678” input is the key to be entered in the Access Point’s configuration to authenticate it with the Radius Server.

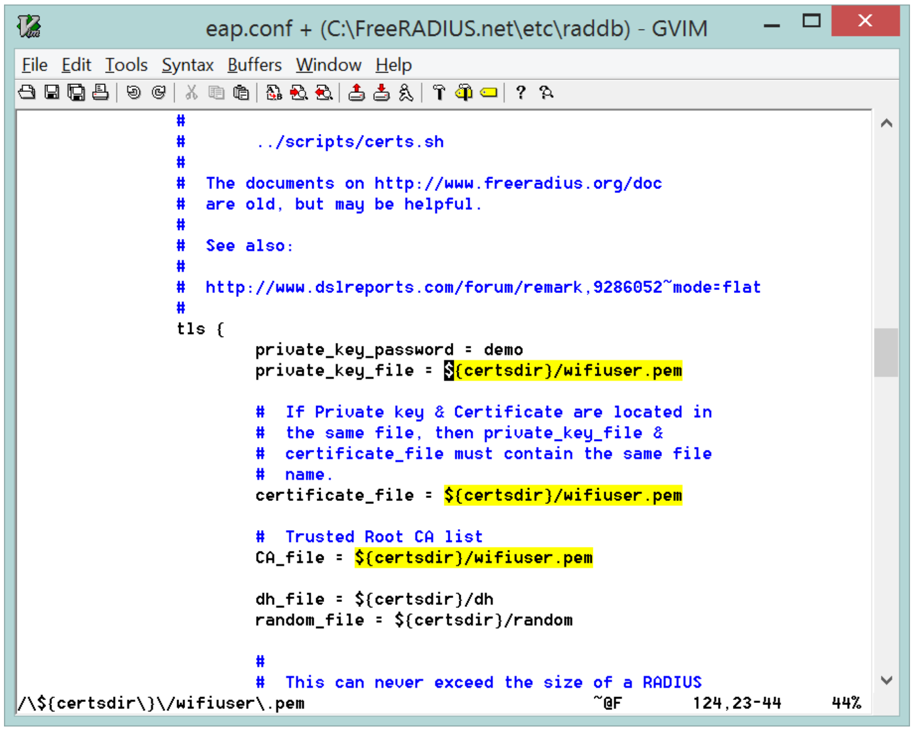
4) Open the eap.conf file and make the following changes:
    a. Change the input for the “default_eap_type” field under the “eap” section to “tls”, as shown in the figure below.


eap.conf (C:\FreeRADIUS.net\etc\raddb) - GVIM
File Edit Tools Syntax Buffers Window Help

```
#
#  Whatever you do, do NOT set 'Auth-Type := EAP'.  The server
#  is smart enough to figure this out on its own.  The most
#  common side effect of setting 'Auth-Type := EAP' is that the
#  users then cannot use ANY other authentication method.
#
#       $Id: eap.conf,v 1.4 2004/04/15 18:34:41 aland Exp $
#
        eap {
                #  Invoke the default supported EAP type when
                #  EAP-Identity response is received.
                #
                #  The incoming EAP messages DO NOT specify which EAP
                #  type they will be using, so it MUST be set here.
                #
                #  For now, only one default EAP type may be used at a time.
                #
                #  If the EAP-Type attribute is set by another module,
                #  then that EAP type takes precedence over the
                #  default type configured here.
                #
                default_eap_type = tls

                #  A list is maintained to correlate EAP-Response
                #  packets with EAP-Request packets.  After a
                #  configurable length of time, entries in the list
                #  expire, and are deleted.
                #
1 change; before #1  1 seconds ago                          1,1           Top
```

Figure 11: Default EAP Type

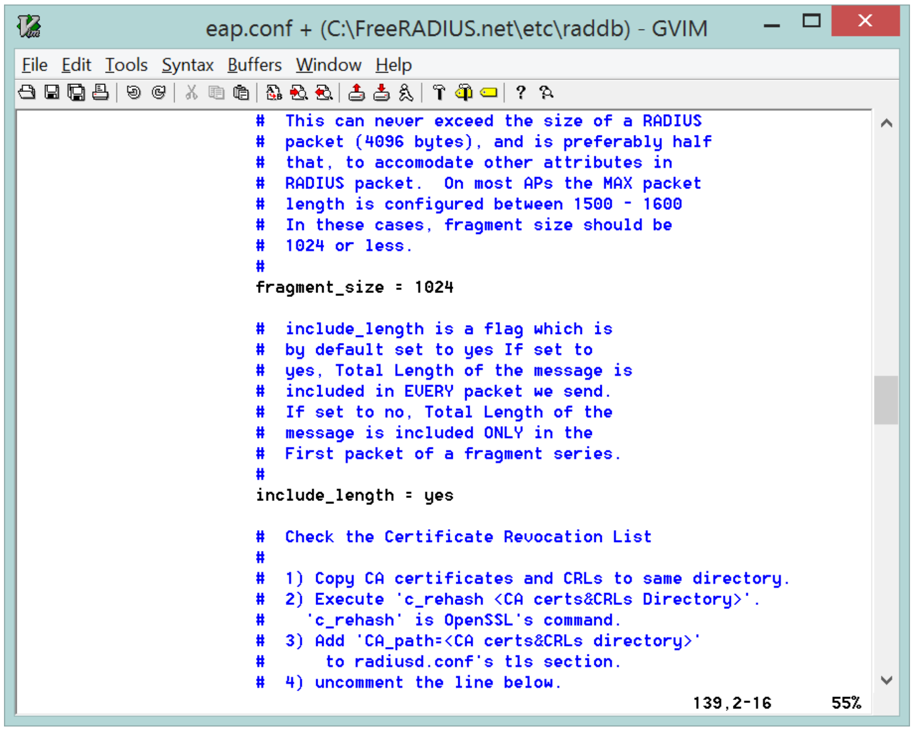
b. Change the inputs for “private_key_file”, “certificate_file” and “CA_file” fields under the “tls” section to “${certsdir}/wifiuser.pem”, as show in the figure below.


eap.conf + (C:\FreeRADIUS.net\etc\raddb) - GVIM
File Edit Tools Syntax Buffers Window Help

```
                #
                #       ../scripts/certs.sh
                #
                #  The documents on http://www.freeradius.org/doc
                #  are old, but may be helpful.
                #
                #  See also:
                #
                #  http://www.dslreports.com/forum/remark,9286052~mode=flat
                #
                tls {
                        private_key_password = demo
                        private_key_file = ${certsdir}/wifiuser.pem

                        #  If Private key & Certificate are located in
                        #  the same file, then private_key_file &
                        #  certificate_file must contain the same file
                        #  name.
                        certificate_file = ${certsdir}/wifiuser.pem

                        #  Trusted Root CA list
                        CA_file = ${certsdir}/wifiuser.pem

                        dh_file = ${certsdir}/dh
                        random_file = ${certsdir}/random

                        #
                        #  This can never exceed the size of a RADIUS
/\${certsdir\}\/wifiuser\.pem                          ^@F       124,23-44      44%
```

Figure 12: TLS Section - I

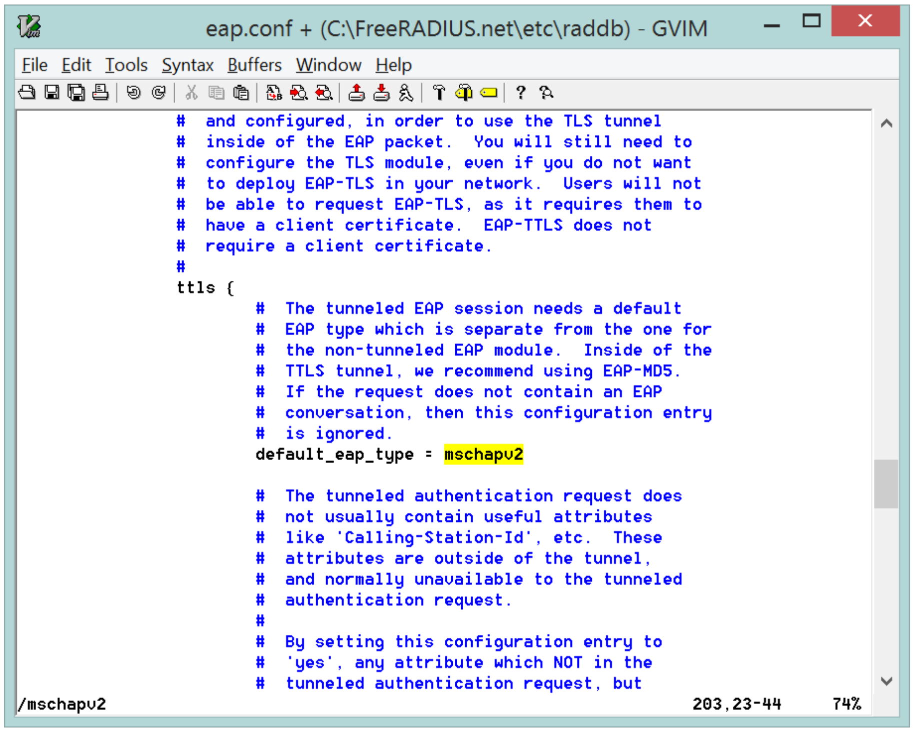
c. Uncomment the “fragment_size” and “include_length” lines under the “tls” section, as shown in the figure below.


eap.conf + (C:\FreeRADIUS.net\etc\raddb) - GVIM
File Edit Tools Syntax Buffers Window Help

```
                        #  This can never exceed the size of a RADIUS
                        #  packet (4096 bytes), and is preferably half
                        #  that, to accomodate other attributes in
                        #  RADIUS packet.  On most APs the MAX packet
                        #  length is configured between 1500 - 1600
                        #  In these cases, fragment size should be
                        #  1024 or less.
                        #
                        fragment_size = 1024

                        #  include_length is a flag which is
                        #  by default set to yes If set to
                        #  yes, Total Length of the message is
                        #  included in EVERY packet we send.
                        #  If set to no, Total Length of the
                        #  message is included ONLY in the
                        #  First packet of a fragment series.
                        #
                        include_length = yes

                        #  Check the Certificate Revocation List
                        #
                        #  1) Copy CA certificates and CRLs to same directory.
                        #  2) Execute 'c_rehash <CA certs&CRLs Directory>'.
                        #    'c_rehash' is OpenSSL's command.
                        #  3) Add 'CA_path=<CA certs&CRLs directory>'
                        #      to radiusd.conf's tls section.
                        #  4) uncomment the line below.
                                                              139,2-16       55%
```

Figure 13: TLS Section – II

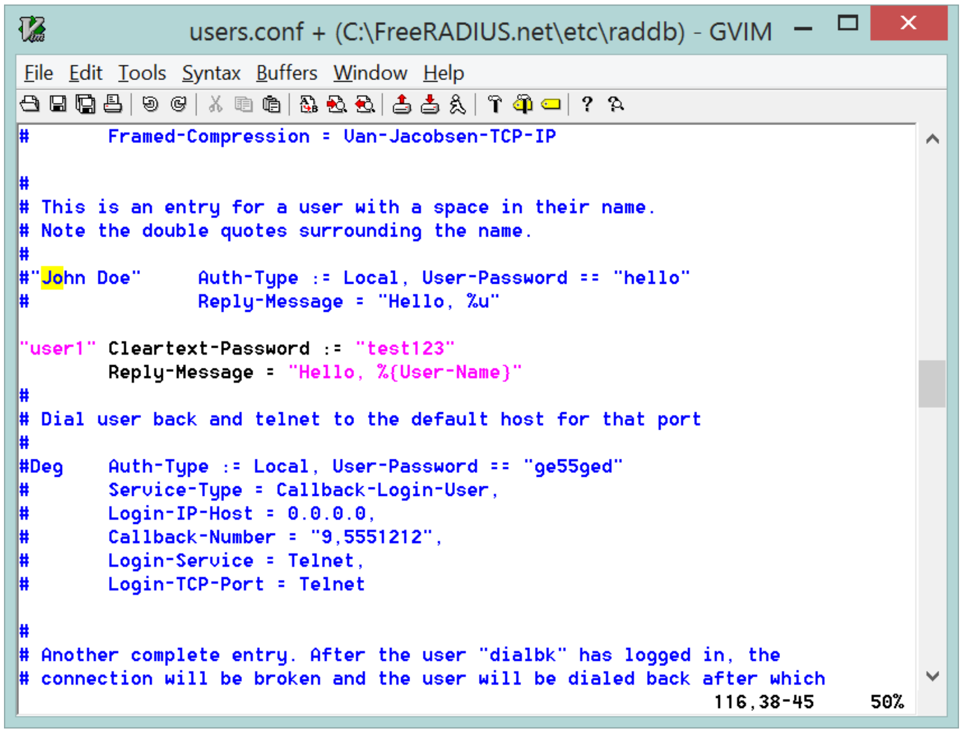
d. Change the input for the “default_eap_type” field under the “ttls” section to “mschapv2”, as shown in the figure below.


eap.conf + (C:\FreeRADIUS.net\etc\raddb) - GVIM
File Edit Tools Syntax Buffers Window Help

```
                #  and configured, in order to use the TLS tunnel
                #  inside of the EAP packet.  You will still need to
                #  configure the TLS module, even if you do not want
                #  to deploy EAP-TLS in your network.  Users will not
                #  be able to request EAP-TLS, as it requires them to
                #  have a client certificate.  EAP-TTLS does not
                #  require a client certificate.
                #
                ttls {
                        #  The tunneled EAP session needs a default
                        #  EAP type which is separate from the one for
                        #  the non-tunneled EAP module.  Inside of the
                        #  TTLS tunnel, we recommend using EAP-MD5.
                        #  If the request does not contain an EAP
                        #  conversation, then this configuration entry
                        #  is ignored.
                        default_eap_type = mschapv2

                        #  The tunneled authentication request does
                        #  not usually contain useful attributes
                        #  like 'Calling-Station-Id', etc.  These
                        #  attributes are outside of the tunnel,
                        #  and normally unavailable to the tunneled
                        #  authentication request.
                        #
                        #  By setting this configuration entry to
                        #  'yes', any attribute which NOT in the
                        #  tunneled authentication request, but
/mschapv2                                                 203,23-44     74%
```

Figure 14: TTLS Section

5) Open the users.conf file and add the lines shown in the figure below starting with “user1”. This adds a user with username “user1” and password “test123”.


users.conf + (C:\FreeRADIUS.net\etc\raddb) - GVIM
File Edit Tools Syntax Buffers Window Help

```
#       Framed-Compression = Van-Jacobsen-TCP-IP

#
# This is an entry for a user with a space in their name.
# Note the double quotes surrounding the name.
#
#"John Doe"      Auth-Type := Local, User-Password == "hello"
#                Reply-Message = "Hello, %u"

"user1" Cleartext-Password := "test123"
        Reply-Message = "Hello, %{User-Name}"
#
# Dial user back and telnet to the default host for that port
#
#Deg    Auth-Type := Local, User-Password == "ge55ged"
#       Service-Type = Callback-Login-User,
#       Login-IP-Host = 0.0.0.0,
#       Callback-Number = "9,5551212",
#       Login-Service = Telnet,
#       Login-TCP-Port = Telnet

#
# Another complete entry. After the user "dialbk" has logged in, the
# connection will be broken and the user will be dialed back after which
                                                    116,38-45     50%
```

Figure 15: User Addition

6) Copy the wifiuser.pem file from RS9113.NBZ.WC.GEN.OSI.x.x.x\utils\Radius_Server\raddb\certs folder to C:\FreeRADIUS.net\etc\raddb\certs\FreeRADIUS.net\DemoCerts folder.

7) Ensure that the FreeRADIUS server is not running. Open the Windows Command Prompt with Administrator privileges and navigate to the C:\FreeRADIUS.net folder.

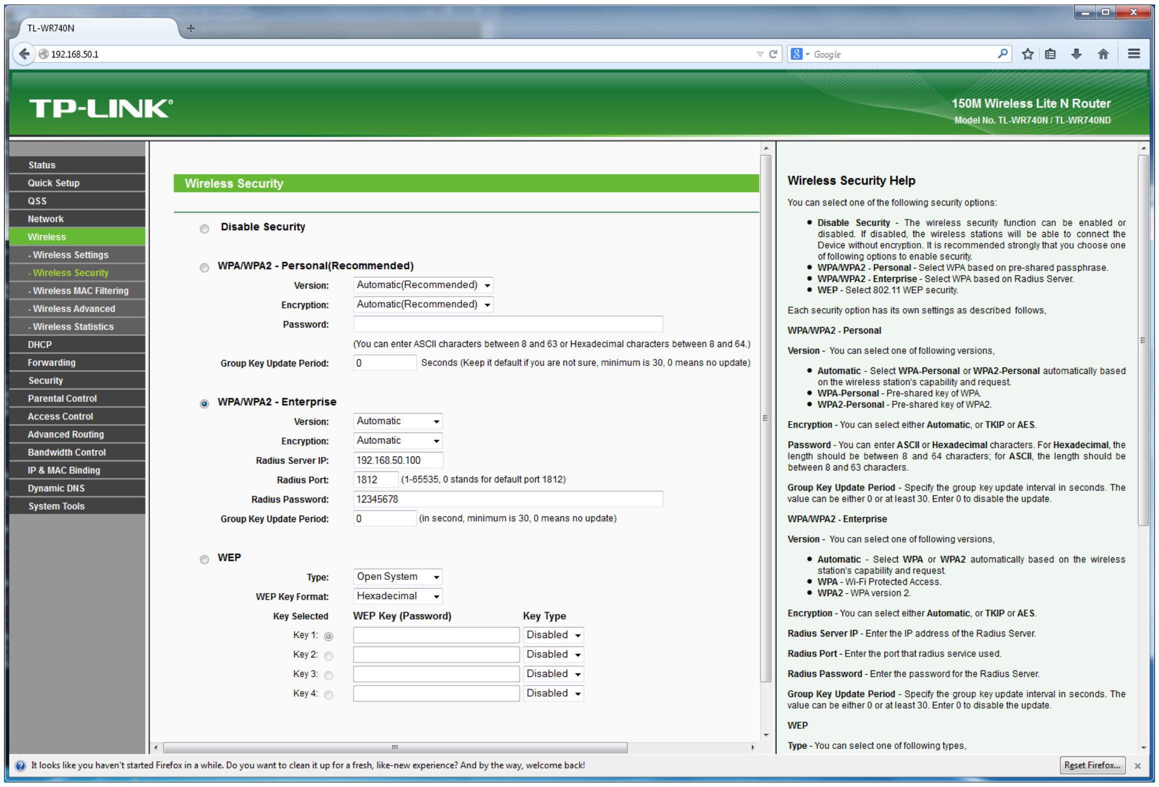
8) Run the “start_radiusd_debug.bat” batch file. You will see a series of prints on the screen. Monitor the prints to ensure that all changes are done correctly. The Radius server has started successfully if you see a print at the end which says, “Ready to process requests.”

5.4.1.2.2 Access Point Configuration

Follow the steps below to configure the Access Point in the setup shown in Figure 10 to work with the Radius server started on Laptop D.

1) Connect the Access Point to Laptop D over Ethernet and open the Access Point pages in a browser by typing the IP address of the Access Point.

2) Navigate to the Wireless Security section and enable the “WPA/WPA2 – Enterprise” option, as shown in the figure below. The page below is for a TP-Link Access Point.



150M Wireless Lite N Router
Model No. TL-WR740N / TL-WR740ND

Wireless Security

Disable Security

WPA/WPA2 - Personal(Recommended)

Version: Automatic(Recommended)
Encryption: Automatic(Recommended)
Password:
(You can enter ASCII characters between 8 and 63 or Hexadecimal characters between 8 and 64.)
Group Key Update Period: 0 Seconds (Keep it default if you are not sure, minimum is 30, 0 means no update)

[✓ WPA/WPA2 - Enterprise]

Version: Automatic
Encryption: Automatic
Radius Server IP: 192.168.50.100
Radius Port: 1812 (1-65535, 0 stands for default port 1812)
Radius Password: 12345678
Group Key Update Period: 0 (in second, minimum is 30, 0 means no update)

WEP

Type: Open System
WEP Key Format: Hexadecimal

Key Selected	WEP Key (Password)	Key Type
Key 1: [✓]		Disabled
Key 2:		Disabled
Key 3:		Disabled
Key 4:		Disabled

Wireless Security Help

You can select one of the following security options:

- **Disable Security** - The wireless security function can be enabled or disabled. If disabled, the wireless stations will be able to connect the Device without encryption. It is recommended strongly that you choose one of following options to enable security.
- **WPA/WPA2 - Personal** - Select WPA based on pre-shared passphrase.
- **WPA/WPA2 - Enterprise** - Select WPA based on Radius Server.
- **WEP** - Select 802.11 WEP security.

Each security option has its own settings as described follows,

WPA/WPA2 - Personal

Version - You can select one of following versions,

- **Automatic** - Select **WPA-Personal** or **WPA2-Personal** automatically based on the wireless station's capability and request.
- **WPA-Personal** - Pre-shared key of WPA.
- **WPA2-Personal** - Pre-shared key of WPA2.

Encryption - You can select either **Automatic**, or **TKIP** or **AES**.

Password - You can enter **ASCII** or **Hexadecimal** characters. For **Hexadecimal**, the length should be between 8 and 64 characters; for **ASCII**, the length should be between 8 and 63 characters.

Group Key Update Period - Specify the group key update interval in seconds. The value can be either 0 or at least 30. Enter 0 to disable the update.

WPA/WPA2 - Enterprise

Version - You can select one of following versions,

- **Automatic** - Select **WPA** or **WPA2** automatically based on the wireless station's capability and request.
- **WPA** - Wi-Fi Protected Access.
- **WPA2** - WPA version 2.

Encryption - You can select either **Automatic**, or **TKIP** or **AES**.

Radius Server IP - Enter the IP address of the Radius Server.

Radius Port - Enter the port that radius service used.

Radius Password - Enter the password for the Radius Server.

Group Key Update Period - Specify the group key update interval in seconds. The value can be either 0 or at least 30. Enter 0 to disable the update.

WEP

Type - You can select one of following types,

It looks like you haven't started Firefox in a while. Do you want to clean it up for a fresh, like-new experience? And by the way, welcome back!

Figure 16: Wireless Security Configuration of Access Point

3) Enter the IP address of the Radius Server in the field labeled, “Radius Server IP”. In the above figure, it is 192.168.50.100.

4) Enter the Radius Password as “12345678”. This is the same as that entered in the clients.conf file of the Radius Server.

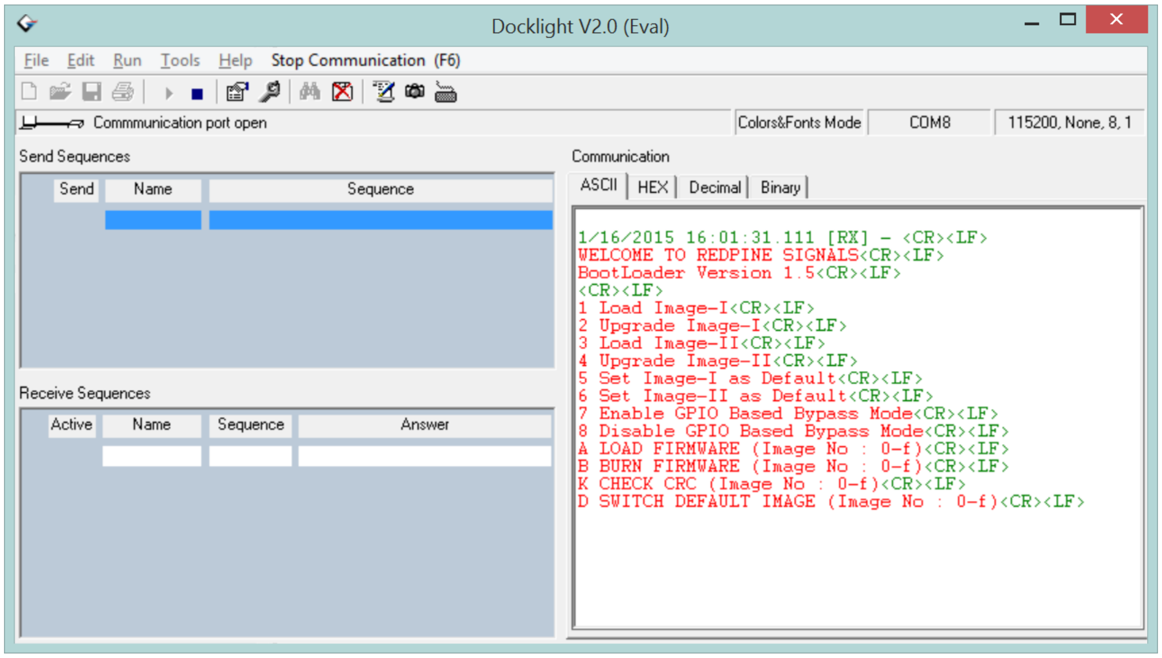
5.4.1.2.3 *Wi-Fi Client Configuration*

1) Open Docklight.

2) Connect the Micro A/B-type USB cablebetween the USB port of the PC and the micro-USB port of the EVB labeled “UART”.

3) Hit “F5” key when in the Docklight window to start communicating with EVB over the serial COM port.

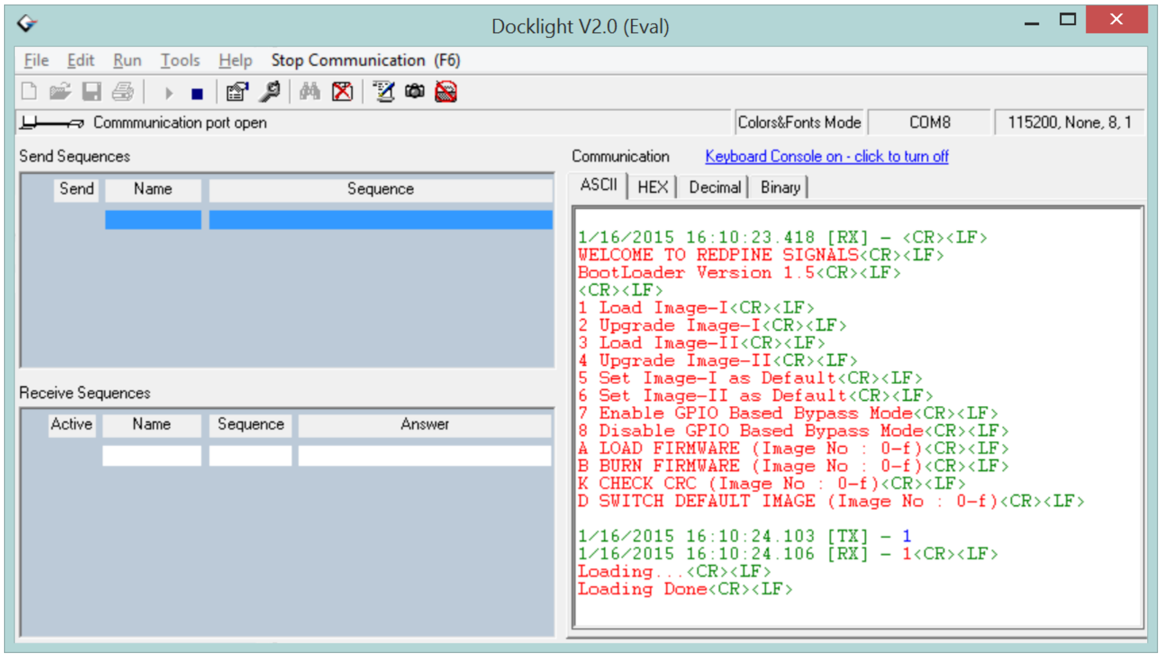
4) At powerup, the module tries to estimate the baud rate of the Host by exchanging data with it. If no data is sent by the Host, the module defaults to the 115200 bps baud rate in approximately 18 seconds. Please refer to the Software PRM document to understand the Automatic Buad Rate Detection Process for faster bootup. At the end of the timeout, you will see a message like, “WELCOME TO REDPINE SIGNALS”, followed by multiple options. Please see the figure below.



```
1/16/2015 16:01:31.111 [RX] - <CR><LF>
WELCOME TO REDPINE SIGNALS<CR><LF>
BootLoader Version 1.5<CR><LF>
<CR><LF>
1 Load Image-I<CR><LF>
2 Upgrade Image-I<CR><LF>
3 Load Image-II<CR><LF>
4 Upgrade Image-II<CR><LF>
5 Set Image-I as Default<CR><LF>
6 Set Image-II as Default<CR><LF>
7 Enable GPIO Based Bypass Mode<CR><LF>
8 Disable GPIO Based Bypass Mode<CR><LF>
A LOAD FIRMWARE (Image No : 0-f)<CR><LF>
B BURN FIRMWARE (Image No : 0-f)<CR><LF>
K CHECK CRC (Image No : 0-f)<CR><LF>
D SWITCH DEFAULT IMAGE (Image No : 0-f)<CR><LF>
```

Figure 17: Module Startup Messages

5) Hit ‘1’ and present Enter. You will see a message that says, “Loading…” followed by “Loading Done”.



```
1/16/2015 16:10:23.418 [RX] - <CR><LF>
WELCOME TO REDPINE SIGNALS<CR><LF>
BootLoader Version 1.5<CR><LF>
<CR><LF>
1 Load Image-I<CR><LF>
2 Upgrade Image-I<CR><LF>
3 Load Image-II<CR><LF>
4 Upgrade Image-II<CR><LF>
5 Set Image-I as Default<CR><LF>
6 Set Image-II as Default<CR><LF>
7 Enable GPIO Based Bypass Mode<CR><LF>
8 Disable GPIO Based Bypass Mode<CR><LF>
A LOAD FIRMWARE (Image No : 0-f)<CR><LF>
B BURN FIRMWARE (Image No : 0-f)<CR><LF>
K CHECK CRC (Image No : 0-f)<CR><LF>
D SWITCH DEFAULT IMAGE (Image No : 0-f)<CR><LF>

1/16/2015 16:10:24.103 [TX] - 1
1/16/2015 16:10:24.106 [RX] - 1<CR><LF>
Loading...<CR><LF>
Loading Done<CR><LF>
```

Figure 18: Firmware Loading Messages

6) The following commands can now be issued to the module. Please refer to the Software PRM document for detailed description of each command and the expected

responses. A command should not be sent until the response for the previous command is received.

NOTE: Each AT command should have a suffix of Carriage Return <CR> and Line Feed <LF>. This is enabled by default in Docklight.

i. `at+rsi_opermode=2,0,4,0`

This command configures the module as Wi-Fi client. The module responds with "OK"

j. `at+rsi_band=0`

This command configures the operating band of Wi-Fi client to 2.4GHz. The module responds with "OK"

k. `at+rsi_init`

This command initializes the Wi-Fi module in the EVB. The module responds with OK<MAC_Address>.

l.
```
at+rsi_eap=TLS,MSCHAPV2,user1,test123,0
at+rsi_eap=TTLS,MSCHAPV2,user1,test123,0
at+rsi_eap=PEAP,MSCHAPV2,user1,test123,0
at+rsi_eap=FAST,MSCHAPV2,user1,test123,0
```

These commands set the EAP mode for the module and set the authentication credentials (username and password). Issue one of them based on which mode needs to be evaluated.

m. To verify the EAP-TLS mode, a security certificate file should be loaded into the module. A Python based script is provided to do the same since loading of certificate files cannot be done through Docklight. Please download and install Python for Windows from https://www.python.org/downloads/release/python-279/. In addition, please download and install the Pyserial program for accessing the Serial COM port from Python. This is available at https://pypi.python.org/pypi/pyserial.

NOTE: Please download the 32-bit version of Python for Windows from the above link even if your PC is 64-bit since the 64-bit version does not update the registry correctly, which causes a problem during the installation of the Pyserial application.

n. Open load_certificate.py file and change the serial port name to "COM8" as shown in the line below. This is the Serial COM port that was detected in Docklight earlier.

```
sp=serial.Serial(port="COM8",baudrate=115200,timeout=0.01)
```

o. Disable Docklight's serial port connection to the EVB by hitting 'F6' when the Docklight window is open.

p. Open the Windows Command Prompt and navigate to the RS9113.NBZ.WC.GEN.OSI.x.x.x\utils\Python folder and give the following command:

```
C:\Python27\python.exe load_certificate.py 1 ..\Radius_Server\raddb\certs\wifiuser.pem
```

This command loads the TLS Certificate into the module.

q. Go back to the Docklight window and hit "F5" to enable the serial port connection between Docklight and the EVB and continue giving the commands below.

r. For non-TLS modes, issue the command below to clear any certificates that might have been loaded previously.

```
at+rsi_cert=1,0,0,0
```

This command should NOT be given for TLS mode after the at+rsi_eap command.

s. `at+rsi_scan=0`

This command scans for available Access Points operating in the 2.4 GHz band. The module responds with information of the Access Points scanned. The data received might have some unreadable characters because of ASCII conversion. You can use the HEX tab of Docklight to see the bytes sent by the module.

t. `at+rsi_join=Test_AP,0,2,2`

This command connects the Wi-Fi client to the Access Point with SSID "Test_AP". On successful association, the module responds with OK.

u.
```
at+rsi_ipconf=0,192.168.50.10,255.255.255.0,192.168.50.1
at+rsi_ipconf=1,0,0,0
```

This command configures the IP address of the module. The IP address configured in the above command is for illustration only. The user has to configure the IP address as per the Access Point's settings using either Manual (first command above) or DHCP mode (second command above).

For the Manual mode, ensure that the desired IP is in the same subnet as the Access Point's subnet. The module responds to this command by sending the configured IP address to the Host. In the terminal, this response might appear as unreadable characters because of ASCII conversion. You can use the HEX tab of Docklight to see the bytes sent by the module.

v. `at+rsi_ltcp=5001,1,0`

This command opens a Listen TCP socket with port number 5001.

5.4.1.2.4 Test Procedure

1) Connect Laptop C to the Access Point. It should have proper security credentials to connect to the AP.

2) A "ping[3]" can be issued from Laptop C to the Wi-Fi module to verify connectivity through the AP.

3) TCP and UDP applications are provided along with release package for execution on the Laptop C. These applications are located in the pathRS9113.NBZ.WC.GEN.OSI.x.x.x\utils\peer_applications\Windows where RS9113.WSC.GENR.x.x.x is software package directory. Open a client TCP socket on the Laptop C using the following command in a Windows Command Prompt.

[3] The module responds to a ping request sent from a remote terminal. There is no command to send a ping request from the module. This is true in all the modes- Client, AP and Wi-Fi Direct.

```
TCP.exe c 2001 <EVB IP address> 5001
```

4) The above command opens a new window with the following options:
   a. Send
   b. Receive
   c. Exit
5) Observe that the terminal on the Wi-Fi Client side (EVB) prints the following message, once the TCP connection is setup with Laptop C.

```
AT+RSI_LTCP_CONNECT=<ip_version><socket_descriptor><dest_po
rt_no><dest_ipaddr><mss><window_size><src_port_no>\r\n
```

6) For testing the Receive mode of the Wi-Fi Client (EVB), follow the steps below:
   a. Type 1 in the TCP.exe window on Laptop C.
   b. On being prompted to enter a data string to be transmitted, type any string and hit Enter to transmit the typed data.
   c. When the data is received on the Wi-Fi Client (EVB) side, you will see a response (asynchronous) from the module as follows:

```
AT+RSI_READ<ip_version><socket_handle><payload_len><s
ource_ip_addr><source_port><payload>
```

7) For testing the Transmit mode of the Wi-Fi Client (EVB), follow the steps below:
   a. Type 2 in the TCP.exe window on Laptop C.
   b. On the Wi-Fi Client (EVB) side, send the command below to transmit data to the Laptop C.

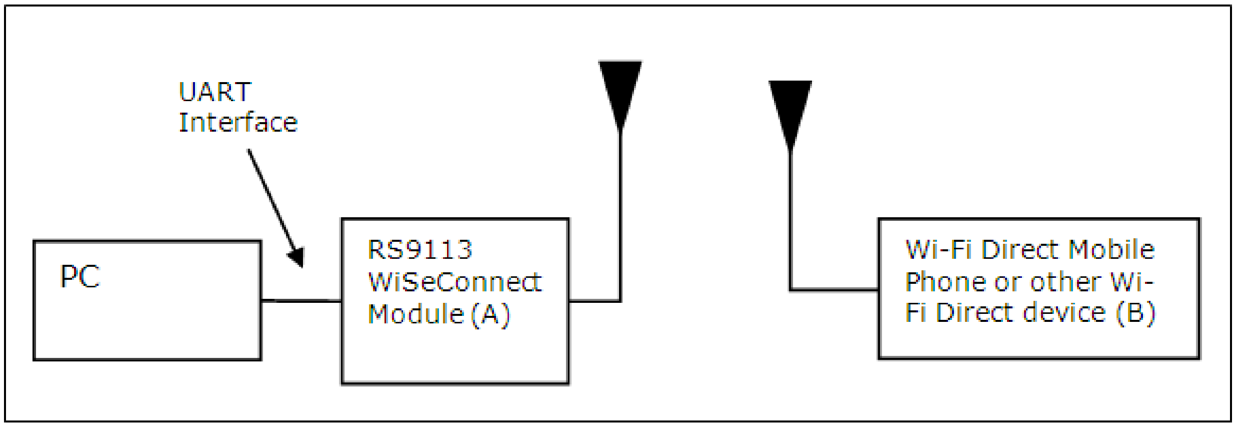
```
at+rsi_snd=<socket_handle>,<payload_len>,<dest_ipaddr
>,<dest_port>,<payload >
```

   c. The transmitted data is displayed on the TCP.exe window of Laptop C.

NOTE: To switch between Enterprise Security and any other security modes (WPA2-PSK, WPA, WEP) Operation mode has to be changed. Enterprise security works in Operation mode 2 and for other security modes the Operation mode is 0. The Change in the Operation mode command can happen just after the Reset (soft reset or hard reset) so Reset is required to toggle between Enterprise Security and other security modes.

5.4.1.3 Wi-Fi Direct Mode

The figure below shows the setup required for this process.


UART Interface
PC
RS9113 WiSeConnect Module (A)
Wi-Fi Direct Mobile Phone or other Wi-Fi Direct device (B)

5.4.1.3.1 *Wi-Fi Configuration*

1) Enable Wi-Fi Direct Mode in the phone. A screenshot of an Android phone's Wi-Fi Direct Settings is shown in the figure below. This is usually available in the Wi-Fi Settings of the phone.

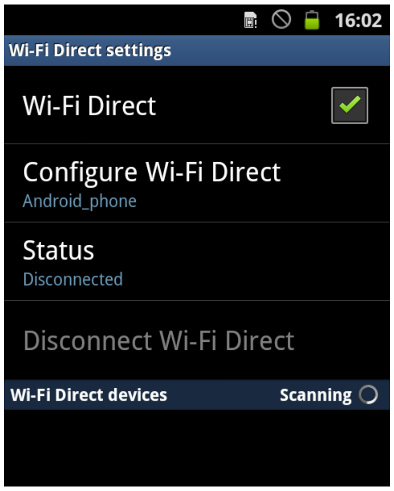


Figure 19: Android Phone's Wi-Fi Direct Settings

The "Configure Wi-Fi Direct" button can be clicked to set the device name accordingly (Android_phone in this case).

2) Follow steps 1 to 6 in Section 5.3.1.1 to power up the EVB and load the firmware.

3) Enter the following commands. A command should be entered only after getting the response of the previous command

   a. `at+rsi_opermode=1,0,20,0`

      This configures the EVB to function in Wi-Fi Direct mode. The module responds with "OK".

   b. `at+rsi_band=0`

This configures the operating band of the EVB. The module responds with “OK”.

c. `at+rsi_init`

This initializes the Wi-Fi module in the EVB. The module responds with OK<MAC_Address>.

d. `at+rsi_fwversion?`

This is an optional command to report the firmware version in use.

e. `at+rsi_wfd=0,RED,1,wiseconnect,12345678`

This starts the Wi-Fi Direct functionality in the module.

The first parameter in this command is called the *Group_Owner_Intent*. It gives the willingness of the module to become a Group Owner. It has been set to the lowest value of 0 in this case. Refer to the Programming Reference Manual for more details. The module responds with “OK”.

4) After issuing the last command, the module starts scanning for Wi-Fi Direct devices, and reports any that are found through the asynchronous message `AT+RSI_WFDDEV`.

5) After the module reports Wi-Fi Direct devices, issue the Join command to connect.

```
at+rsi_join=Android_phone,0,2
```

In this example, it is assumed that the device name of the phone is configured as “Android_phone”. The phone should also display the device name of the module (RED in this example). Within about 10 secs of issuing the Join command in the module, click “Connect” on the phone as well to connect to the module. On successful connection, the module responds with OK and the Phone displays “Connected”

6) If the module has NOT become a GO (Group Owner), issue the below command to get an IP address from the phone over DHCP. If the module has become a GO, this command need not be issued.

```
at+rsi_ipconf=1,0,0,0
```

The acquired IP address is returned to the Host but might not be readable on the Docklight because of ASCII conversion. You can use the HEX tab of Docklight to see the bytes sent by the module.

Please refer to the description for the Join command in the Programming Reference Manual. The parameter *GO_Status* returned with the OK response of the Join command is used to determine whether the module became a GO or not. If the module becomes the GO, it will internally assign itself the IP 192.168.100.76 and will act as a DHCP server. The Wi-Fi Direct phone will acquire an IP address from the module automatically (assuming DHCP client is enabled in the phone). Since the *Group_Owner_Intent* has been set to the lowest value, module will become a Wi-Fi Direct client.

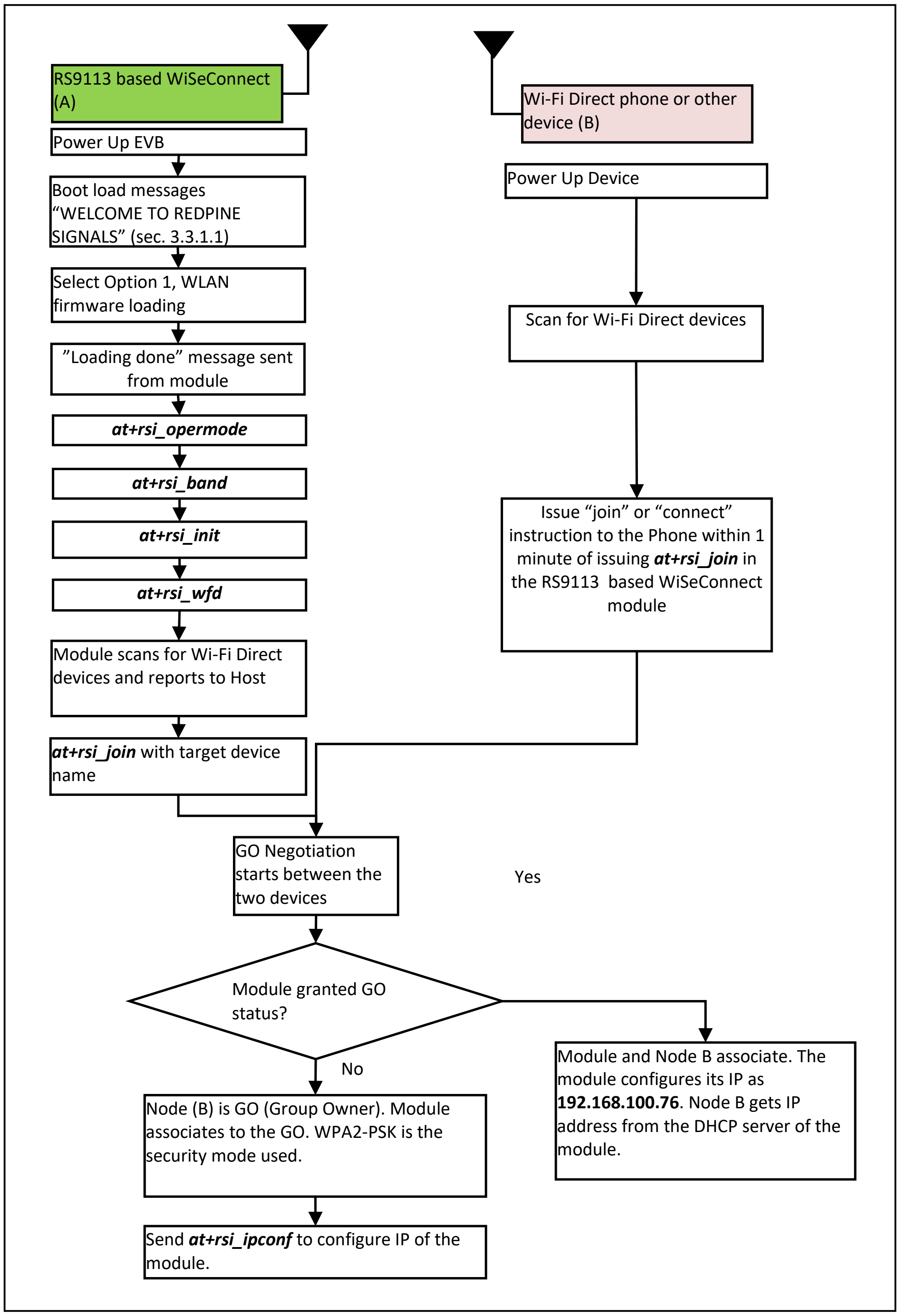

RS9113 based WiSeConnect (A)
Power Up EVB
Boot load messages "WELCOME TO REDPINE SIGNALS" (sec. 3.3.1.1)
Select Option 1, WLAN firmware loading
"Loading done" message sent from module
at+rsi_opermode
at+rsi_band
at+rsi_init
at+rsi_wfd
Module scans for Wi-Fi Direct devices and reports to Host
at+rsi_join with target device name
Wi-Fi Direct phone or other device (B)
Power Up Device
Scan for Wi-Fi Direct devices
Issue "join" or "connect" instruction to the Phone within 1 minute of issuing at+rsi_join in the RS9113 based WiSeConnect module
GO Negotiation starts between the two devices
Yes
Module granted GO status?
No
Node (B) is GO (Group Owner). Module associates to the GO. WPA2-PSK is the security mode used.
Send at+rsi_ipconf to configure IP of the module.
Module and Node B associate. The module configures its IP as 192.168.100.76. Node B gets IP address from the DHCP server of the module.

Figure 20: Command Flow in Wi-Fi Direct Mode

5.4.1.3.2 Test Procedure

In the following descriptions it is assumed for illustrative purposes that the module's IP address is **192.168.100.76** and the remote peer (phone) IP address is **192.168.100.77**.

Running Ping Application

A ping application can be run from the Wi-Fi Direct Phone, with the destination address as 192.168.100.76. The module will send the ping response. Ping based applications are freely available for Android phones.

Exchanging data through sockets

For exchanging data between the module and the Wi-Fi Direct Phone, an application may be written by the user at the mobile phone to open sockets and transmit or receive data. At the module side, sockets can be opened by using the below commands:

1) To open a server TCP socket in the module, issue the below command.

   ```
   at+rsi_ltcp=5001,1,0
   ```

   5001 is the example port number of the socket opened. The module responds with:

   ```
   OK<ip_version><socket_type><socket_handle><Lport><module_ip
   v4addr>\r\n
   ```

   Once the socket is opened, a client socket should be opened at the remote peer (phone) to connect to this socket and establish a TCP connection. When the TCP connection is established in this manner, the module responses with:

   ```
   AT+RSI_LTCP_CONNECT=<ip_version><socket_descriptor><dest_po
   rt_no><dest_ipaddr><mss><window_size><src_port_no>\r\n
   ```

   The `socket_handle` parameter will be used in the subsequent sections to send data.

2) After the TCP connection is established, data can be exchanged between the two nodes. To send data from the module, issue the following command in Docklight:

   ```
   at+rsi_snd=<socket_handle>,8,0,0,abcdefgh
   ```

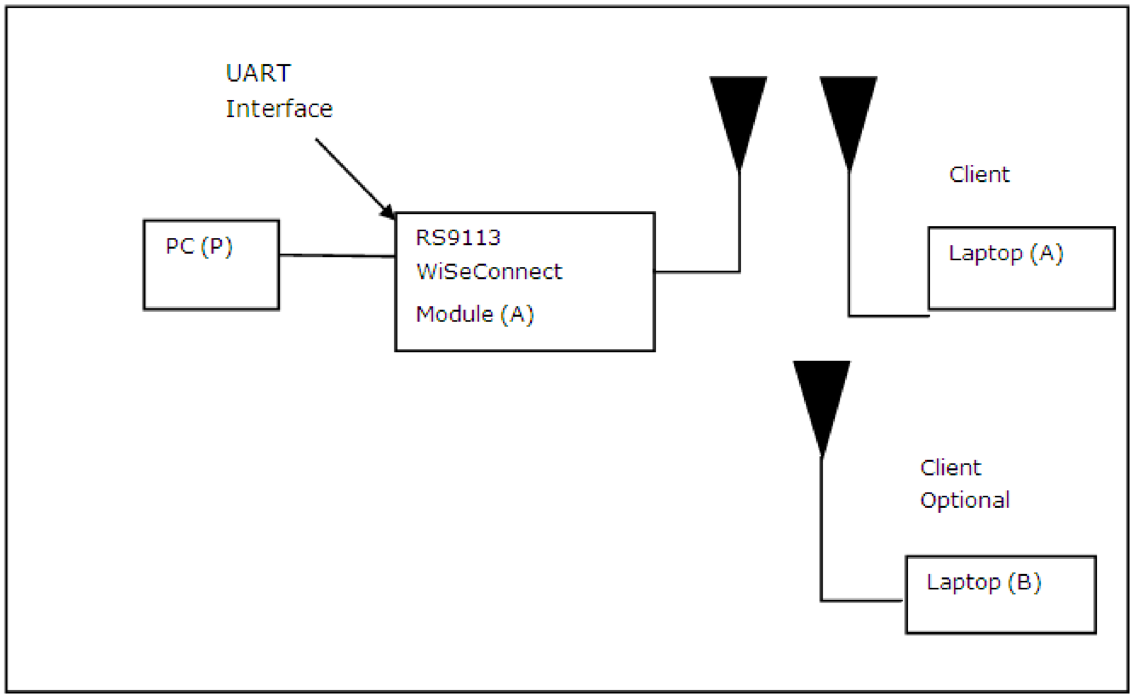
   where abcdefgh is the data stream to be transmitted to the remote Peer and`socket_handle` is the parameter returned when the TCP socket is opened in the module. Refer to the programming reference manual for more details.

   If the remote peer sends data to the module, the module receives it and shows the data in the terminal with the message

   ```
   AT+RSI_READ<ip_version><socket_handle><payload_len><source_
   ip_addr><source_port><payload>
   ```

5.4.1.4 Wi-Fi Access Point Mode

The figure below shows the setup required for this process.



5.4.1.4.1 *Access Point Configuration Through AT Commands*

1) Follow steps 1 to 6 in Section 5.3.1.1 to power up the EVB and load the firmware.

2) Enter the following commands. A command should be entered only after getting the response of the previous command

   a. `at+rsi_opermode=6,1,48,0`

      This configures the EVB to function in AP mode. The module responds with "OK".

   b. `at+rsi_band=0`

      This configures the operating band of the EVB. The module responds with "OK".

   c. `at+rsi_init`

      This initializes the WiFi module in the EVB. The module responds with OK<mac_address>

   d. `at+rsi_fwversion?`

      This is an optional command to report the firmware version in use.

   e. `at+rsi_ipconf=0,192.168.0.30,255.255.255.0,192.168.0.30`

      This command configures the IP (192.168.0.30 in this example) of the AP. If this command is not issued, a default IP of 192.168.100.76 will be used.

   f. `at+rsi_apconf=6,REDPINE_AP,0,0,0,200,3,3`

      The SSID is configured as "redpine", to operate in channel 6. Please refer to the Software PRM for more details on the other parameters of this command.

g. `at+rsi_join=REDPINE_AP,0,2,0`

This starts the Access Point functionality in the module.

The module is now configured as an Access Point. Its IP address is 192.168.0.30. A Laptop can now scan for networks and the SSID of the module, "REDPINE_AP" will be displayed in the Laptop's list of Scanned APs. After the client Laptop (B) connects to the AP, it acquires an IP address over DHCP. The IP address assigned to the laptop can be known by opening the Windows Command Prompt and issuing the command "ipconfig". It is assumed for illustrative purposes that the IP of the Laptop is 192.168.0.32.

5.4.1.4.2 Test Procedure

The applications provided in the USB drive (as part of the release package) send and receive TCP and UDP packets.

TCP and UDP applications are provided along with release package for execution on the Laptop. These applications are located in the path RS9113.NBZ.WC.GEN.OSI.x.x.x\utils\peer_applications\Windows where RS9113.WSC.GENR.x.x.x is software package directory.

1) Open a TCP Server socket on the Wi-Fi Client (EVB) side using the following AT command:

```
at+rsi_ltcp=5001,1,0
```

2) The module's response will look as follows:

```
OK<ip_version><socket_type><socket_handle><Lport><module_ip
addr>\r\n
```

The `socket_handle` in the response above is used for subsequent commands.

3) Open a TCP client socket on Laptop C by running the TCP.exe application as follows in the Windows Command Prompt:

```
TCP.exe c 2001 192.168.0.30 5001
```

Ensure that you run the Windows Command Prompt program as an Administrator and any firewalls which block creation of sockets are disabled before running the application.

4) The above command opens a new window with the following options:

    a. Send

    b. Receive

    c. Exit

5) Observe that the Docklight on the Wi-Fi AP side (EVB) prints the following message, once the TCP connection is setup with Laptop C.

```
AT+RSI_LTCP_CONNECT=<ip_version><socket_descriptor><dest_po
rt_no><dest_ipaddr><mss><window_size><src_port_no>\r\n
```

6) For testing the Receive mode of the Wi-Fi AP (EVB), follow the steps below:

    a. Type 1 in the TCP.exe window on Laptop C.

    b. On being prompted to enter a data string to be transmitted, type any string and hit Enter to transmit the typed data.

c. When the data is received on the Wi-Fi AP (EVB) side, you will see a response (asynchronous) from the module as follows:

```
AT+RSI_READ<ip_version><socket_handle><payload_len><s
ource_ip_addr><source_port><payload>
```

7) For testing the Transmit mode of the Wi-Fi AP (EVB), follow the steps below:

d. Type 2 in the TCP.exe window on Laptop C.

e. On the Wi-Fi AP (EVB) side, send the command below to transmit data to the Laptop C.

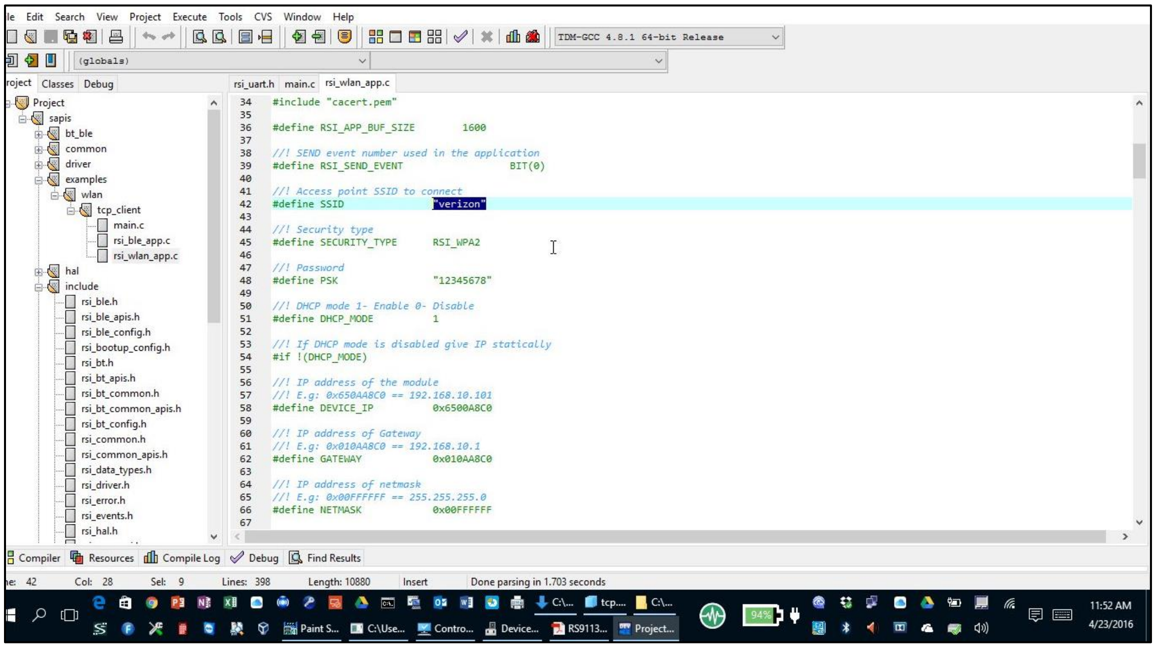
```
at+rsi_snd=<socket_handle>,<payload_len>,<dest_ipaddr
>,<dest_port>,<payload >
```

f. The transmitted data is displayed on the TCP.exe window of Laptop C.

5.4.2 Wi-Fi Evaluation using Binary Commands

5.4.2.1.1 TCP Client Application

Configuring the Application: The first application will be the EVB operating as a Wi-Fi client associated to an access point/router.Edit the following file rsi_tcp_client.c
Below is shown where to modify the Define SSID to match the SSID that is on your access point



```
#include "cacert.pem"

#define RSI_APP_BUF_SIZE       1600

//! SEND event number used in the application
#define RSI_SEND_EVENT                  BIT(0)

//! Access point SSID to connect
#define SSID                   "verizon"

//! Security type
#define SECURITY_TYPE          RSI_WPA2

//! Password
#define PSK                    "12345678"

//! DHCP mode 1- Enable 0- Disable
#define DHCP_MODE              1

//! If DHCP mode is disabled give IP statically
#if !(DHCP_MODE)

//! IP address of the module
//! E.g: 0x650AA8C0 == 192.168.10.101
#define DEVICE_IP              0x6500A8C0

//! IP address of Gateway
//! E.g: 0x010AA8C0 == 192.168.10.1
#define GATEWAY                0x010AA8C0

//! IP address of netmask
//! E.g: 0x00FFFFFF == 255.255.255.0
#define NETMASK                0x00FFFFFF
```

The next step is to modify the Define Security_type to match the one on the access point that you will be connecting to.
You can use either the RSI_WPA2 syntax for WPA2-PSK or the values Shown below for WPA2=2.
Please refer to the SAPI document for the syntax of the various types of security_type available. For this application the WPA2-PSK security_type was shown.

Parameters

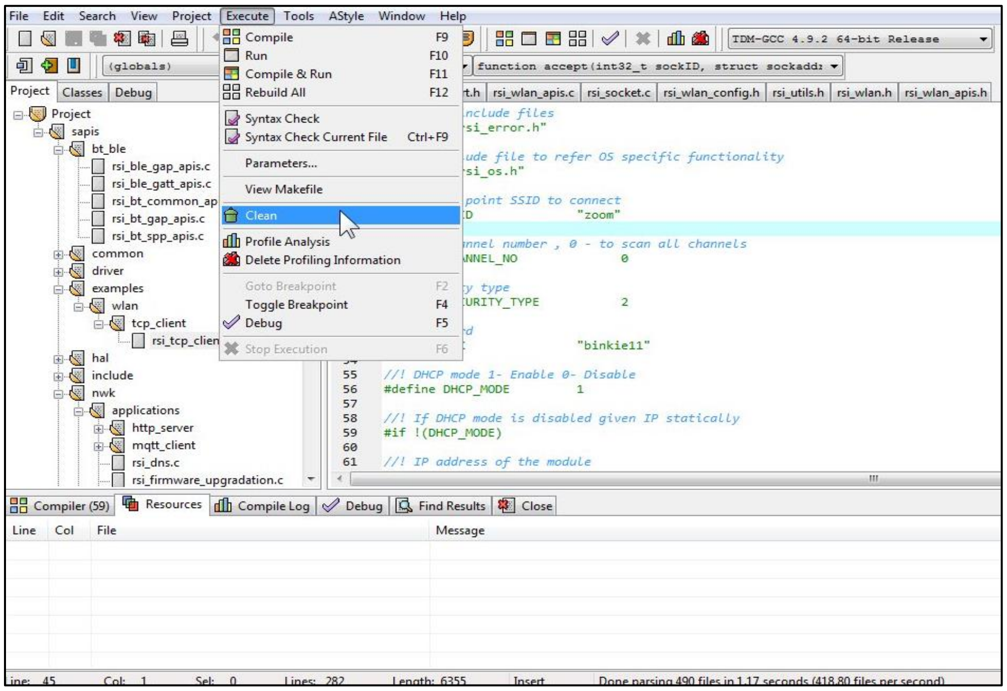
Parameter	Description
`ssid`	SSID of Access point to connect, SSID should be less than or equal to 32 bytes.
`sec_type`	Security type of the Access point to connect 0 : RSI_OPEN, 1 : RSI_WPA, 2 : RSI_WPA2, 3 : RSI_WEP, 4 : RSI_WPA_EAP, 5 : RSI_WPA2_EAP, 6 : RSI_WPA_WPA2_MIXED, 7 : RSI_WPA_PMK, 8 : RSI_WPA2_PMK, 9 : RSI_WPS_PIN, 10 : RSI_USE_GENERATED_WPSPIN, 11 : RSI_WPS_PUSH_BUTTON,
`secret_key`	Point to a buffer contains security information based on `sec_type`.

Also, ensure to change: IP address of the remote server based on the Laptop's IP address where the TCP server is running.

```
#define SERVER_IP_ADDRESS       0x640AA8C0
```

(192.168.10.100 in Little Endean format)

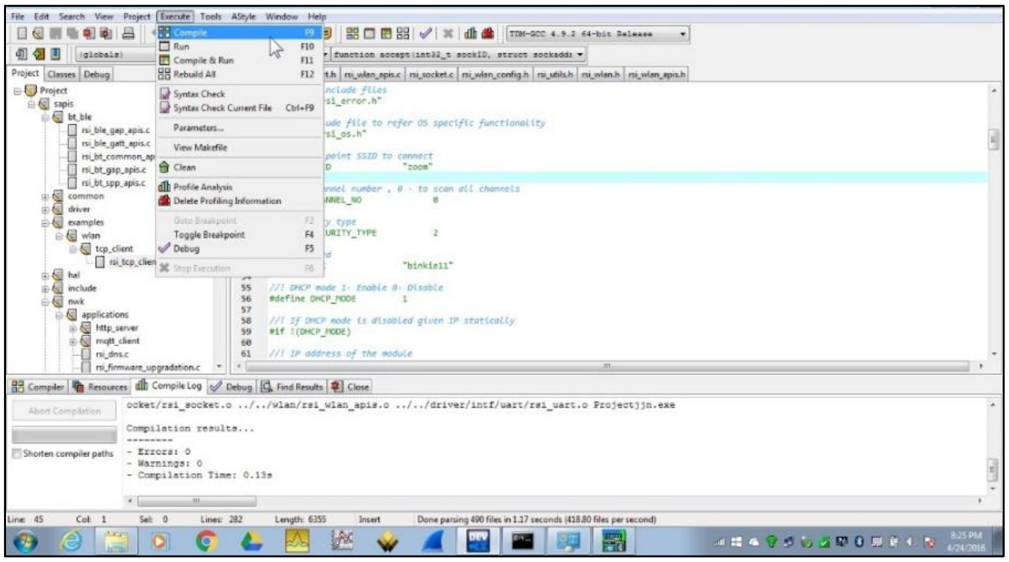
The next step is to select in the Execute menu item the "Clean" item. This will remove the temporary files and any previous compiles executable.



```
nclude files
si_error.h"

ude file to refer OS specific functionality
si_os.h"

point SSID to connect
D                "zoom"

nnel number , 0 - to scan all channels
NNEL_NO                0

y type
URITY_TYPE             2

d
                 "binkie11"

//! DHCP mode 1- Enable 0- Disable
#define DHCP_MODE          1

//! If DHCP mode is disabled given IP statically
#if !(DHCP_MODE)

//! IP address of the module
```

```
Line: 45   Col: 1   Sel: 0   Lines: 282   Length: 6355   Insert   Done parsing 490 files in 1.17 seconds (418.80 files per second)
```

The next step is to compile the code into an executable file. As shown below



```
ocket/rsi_socket.o ../../wlan/rsi_wlan_apis.o ../../driver/intf/uart/rsi_uart.o Projectjjn.exe

Compilation results...
--------
- Errors: 0
- Warnings: 0
- Compilation Time: 0.13s
```

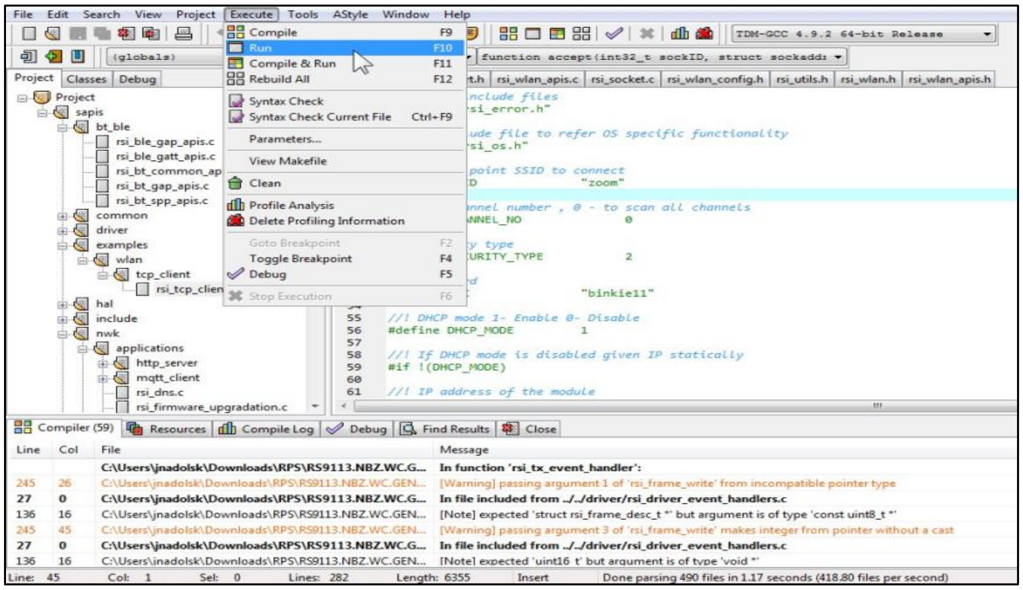
As you can see there was no errors. There were a number of warnings but these are mostly syntax that don't effect the operation of the EVB with the executable.

On another client PC associated to the same access point/router has the EVB you need to run the TCP.exe server using the following command.

1. Make sure that the TCP.exe (Available in the (Package $)\utils\peer_applications\Windows) program is in the windows\system32 directory.
2. Check to see that your PC's firewall allows the TCP port 5001 for this demo
3. Open a command window

4. Enter the following commands;
   a. TCP s 5001
5. You are activating a TCP server using port 5001.

   The next step is to execute the compiled code.



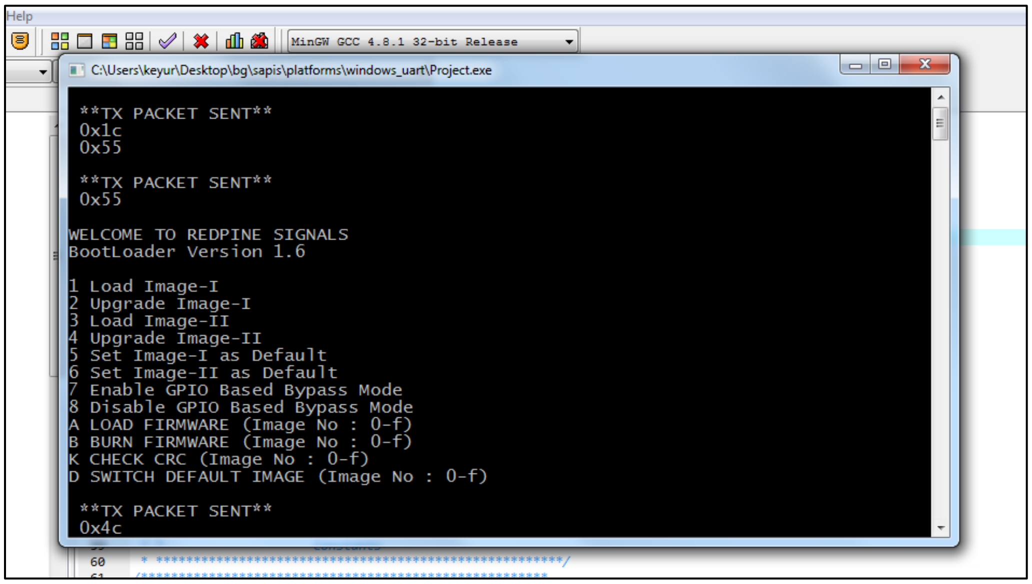
```
nclude files
si_error.h"

ude file to refer OS specific functionality
si_os.h"

point SSID to connect
D                    "zoom"

nnel number , 0 - to scan all channels
NNEL_NO               0

y type
URITY_TYPE            2

d
                     "binkie11"

//! DHCP mode 1- Enable 0- Disable
#define DHCP_MODE         1

//! If DHCP mode is disabled given IP statically
#if !(DHCP_MODE)

//! IP address of the module
```

Line	Col	File	Message
		C:\Users\jnadolsk\Downloads\RPS\RS9113.NBZ.WC.G...	In function 'rsi_tx_event_handler':
245	26	C:\Users\jnadolsk\Downloads\RPS\RS9113.NBZ.WC.GEN...	[Warning] passing argument 1 of 'rsi_frame_write' from incompatible pointer type
27	0	C:\Users\jnadolsk\Downloads\RPS\RS9113.NBZ.WC.G...	In file included from ../../driver/rsi_driver_event_handlers.c
136	16	C:\Users\jnadolsk\Downloads\RPS\RS9113.NBZ.WC.GEN...	[Note] expected 'struct rsi_frame_desc_t *' but argument is of type 'const uint8_t *'
245	45	C:\Users\jnadolsk\Downloads\RPS\RS9113.NBZ.WC.GEN...	[Warning] passing argument 3 of 'rsi_frame_write' makes integer from pointer without a cast
27	0	C:\Users\jnadolsk\Downloads\RPS\RS9113.NBZ.WC.G...	In file included from ../../driver/rsi_driver_event_handlers.c
136	16	C:\Users\jnadolsk\Downloads\RPS\RS9113.NBZ.WC.GEN...	[Note] expected 'uint16_t' but argument is of type 'void *'

Line: 45 Col: 1 Sel: 0 Lines: 282 Length: 6355 Insert Done parsing 490 files in 1.17 seconds (418.80 files per second)

Go to the Execute menu item and select the 'Run" command.

You will notice that a command window has opened showing TCP client/Server data was transmitted and received.

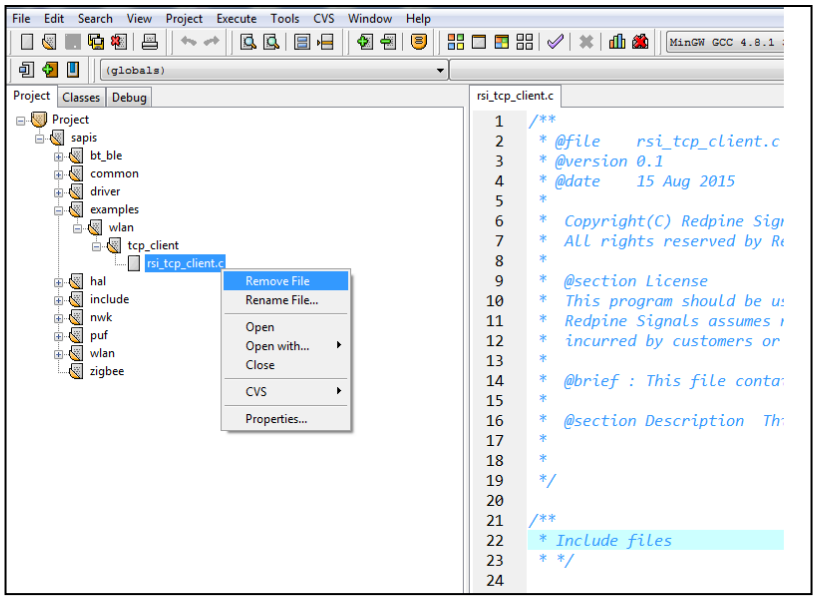

Help
MinGW GCC 4.8.1 32-bit Release
C:\Users\keyur\Desktop\bg\sapis\platforms\windows_uart\Project.exe

```
 **TX PACKET SENT**
 0x1c
 0x55

 **TX PACKET SENT**
 0x55

WELCOME TO REDPINE SIGNALS
BootLoader Version 1.6

1 Load Image-I
2 Upgrade Image-I
3 Load Image-II
4 Upgrade Image-II
5 Set Image-I as Default
6 Set Image-II as Default
7 Enable GPIO Based Bypass Mode
8 Disable GPIO Based Bypass Mode
A LOAD FIRMWARE (Image No : 0-f)
B BURN FIRMWARE (Image No : 0-f)
K CHECK CRC (Image No : 0-f)
D SWITCH DEFAULT IMAGE (Image No : 0-f)

 **TX PACKET SENT**
 0x4c
```

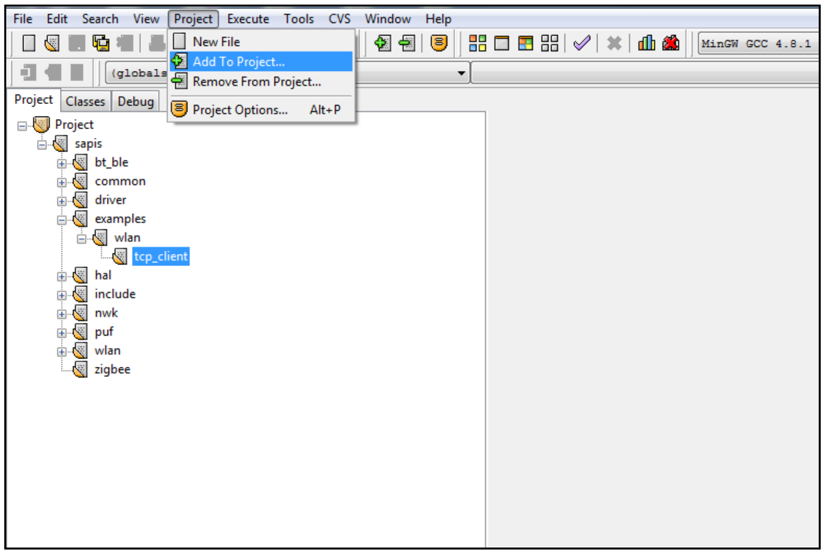
You should then see that there are packets passing from the EVB board through the access point/router to the second client activating as a TCP packet server

Add/Remove/Change Application: To Add another example delete 'tcp_client.c' from project panel at path 'sapis/examples/wlan'. To add new example select 'Add to Project' option from 'Project' tab then explore path of examples and select related files. Then rebuild and run/debug the application.

The below images demonstrate these steps:



```
/**
 * @file    rsi_tcp_client.c
 * @version 0.1
 * @date    15 Aug 2015
 *
 *  Copyright(C) Redpine Sig
 *  All rights reserved by R
 *
 *  @section License
 *  This program should be u
 *  Redpine Signals assumes 
 *  incurred by customers or
 *
 *  @brief : This file conta
 *
 *  @section Description  Th
 *
 *
 */

/**
 * Include files
 * */
```



NOTE:

For a complete list of examples available as part of the package, refer to Appendix B.

Also available is an independent user guide for each example mentioned in the Appendix folder included in the folder

Refer to (Package $)\RS9113.NBZ.WC.GEN.OSI.X.X.X\host\sapis\platforms\windows_uart in the release package for the necessary projects and documents to run a binary project on Windows over the UART interface.

5.5 BLE Evaluation in UART Mode

In UART Mode, the supported test methods for BLE are

- Using AT Commands via a Terminal Utility
- Using binary commands via Dev C++ IDE

The following sub-sections describe configuring and using the module in different security and operational modes of BLE.

5.5.1 BLE Evaluation using AT Commands

5.5.1.1 BLE in Peripheral (Slave) Mode

5.5.1.1.1 BLE Peripheral Mode Configuration through AT commands

```
[TX] - at+rsi_opermode=851968,0,4,0<CR><LF>
[RX] - OK<CR><LF>
bt_loaded<NUL>
```

This opermode enables Wi-Fi+BLE mode of operation. The message bt_loaded indicates successful operation of the command.

```
[TX] - at+rsibt_setlocalname=12,redpines1234<CR><LF>
[RX] - OK <CR><LF>
```

Used to set name to the local device

```
[TX] - at+rsibt_getlocalbdaddr?<CR><LF>
[RX] - OK  00-23-A7-4C-24-95<CR><LF>
```

Used to query the BD address of the local device

```
[TX] - at+rsibt_addservice=2,180A,3,30<CR><LF>
[RX] - OK 1558C,A<CR><LF>
```

used to add the new service Record in BLE GATT record list. The module responds with a service record handle if the service is created successfully. In this example, the service record handle is 1558C. This is used to add attribute records to the service.

```
[TX] -
at+rsibt_addattribute=1558C,B,2,2803,8,6,8,0,0C,00,00,2A<CR><L
F>
[RX] - OK <CR><LF>
```

This is used to add a characteristic attribute record to the above created service using the service record handle 1558C and UUID 2803.

```
[TX]-
at+rsibt_addattribute=1558C,E,2,1AA1,A,a,1,2,3,4,5,6,7,8,9,0\r
\n
[RX] - OK <CR><LF>
```

This is used to add a characteristic attribute record to the above created service using the service record handle 1558C and UUID 1AA1.

```
at+rsibt_addattribute=1558C,F,2,1BB1,1A,a,1,2,3,4,5,6,7,8,9,0\
r\n
```

```
[RX] – OK <CR><LF>
```

This is used to add a characteristic attribute record to the above created service using the service record handle 1558C and UUID 1BB1.

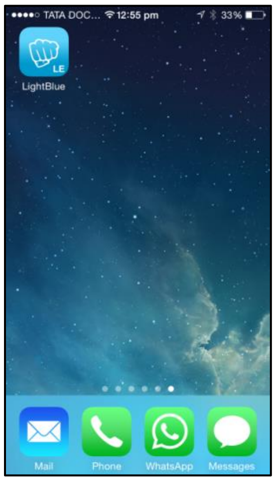
```
[TX] - at+rsibt_advertise=1,0,0,0,0<CR><LF>
[RX] - OK <CR><LF>
```

You will see an asynchronous message to indicate that a B:LE device has connected.

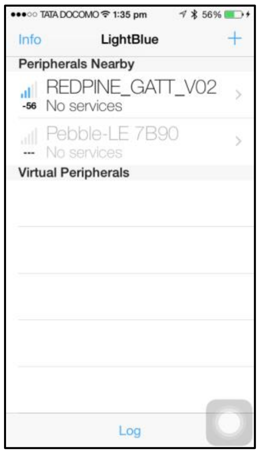
```
AT+RSIBT_LE_DEVICE_CONNECTED=1, 7E-38-0A-91-8C-DF,0<CR><LF>
```

5.5.1.1.2 *Test procedure*

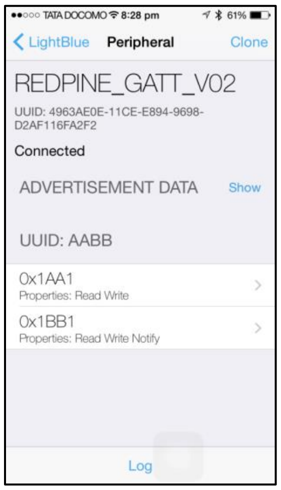
- Go to “LightBlue” iPhone app



- Once you open the APP you can see “REDPINE_GATT_V02”



- Now connect to “REDPINE_GATT_V02”. You will see a BT CONN event on the module side.
- Once connected you will see as below:


●●○○○ TATA DOCOMO 8:28 pm 61%

LightBlue Peripheral Clone

REDPINE_GATT_V02

UUID: 4963AE0E-11CE-E894-9698-D2AF116FA2F2

Connected

ADVERTISEMENT DATA Show

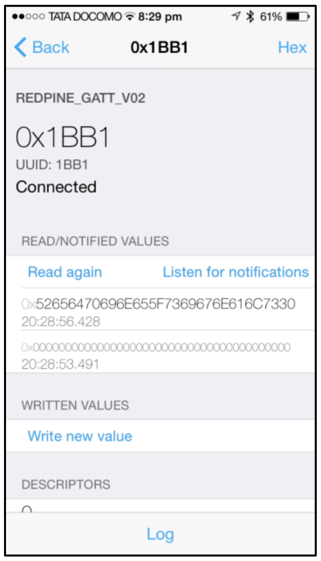
UUID: AABB

0x1AA1
Properties: Read Write

0x1BB1
Properties: Read Write Notify

Log

- Now you need to select 1BB1, which is a Read Write Notify service.
- Enable the "Listen for notification". This will show the hex stream e.g. 0x52656470696E655F7369676E616C7330
- In ASCII → Redpine_signals0
- Last byte will be incremented and seen here every time we send something to module by writing in the 1AA1 service.


●●○○○ TATA DOCOMO 8:29 pm 61%

Back 0x1BB1 Hex

REDPINE_GATT_V02

0x1BB1

UUID: 1BB1

Connected

READ/NOTIFIED VALUES

Read again Listen for notifications

0x52656470696E655F7369676E616C7330
20:28:56.428

0x0000000000000000000000000000000000000000
20:28:53.491

WRITTEN VALUES

Write new value

DESCRIPTORS

Log

5.5.1.2 BLE in Central (Master) Mode

5.5.1.2.1 BLE Central Mode Configuration through AT commands

1) Follow steps 1 to 6 in Section 5.3.1.1 to power up the EVB and load the firmware.
2) Enter the following commands. A command should be entered only after getting the response of the previous command

```
[TX] - at+rsi_opermode=851968,0,4,0<CR><LF>
[RX] – OK <CR><LF>
```

This opermode enables Wi-Fi+BLE mode of operation. The message bt_loaded indicates successful operation of the command.

```
[TX] - at+rsibt_getlocalname?<CR><LF>

[RX] – OK <CR><LF>
```

Used to query the name of the local device

```
[TX] - at+rsibt_getlocalbdaddr?<CR><LF>

[RX] – OK <CR><LF>
```

Used to query the BD address of the local device

```
[TX] - at+rsibt_addservice=2,180A,3,30<CR><LF>

[RX] – OK 1558C,A<CR><LF>
```

used to add the new service Record in BLE GATT record list. The module responds with a service record handle if the service is created successfully. In this example, the service record handle is 1558C. This is used to add attribute records to the service.

```
[TX] -
at+rsibt_addattribute=1558C,B,2,2803,2,6,8,0,0C,00,00,2A<CR
><LF>

[RX] – OK <CR><LF>
```

This is used to add a characteristic attribute record to the above created service using the service record handle 1558C and UUID 2803.

```
[TX] - at+rsibt_scan=1,0,0<CR><LF>

[RX] – OK <CR><LF>
```

Used to scan for remote LE advertise devices

```
[TX] - at+rsibt_connect=0,<BD Address of the peripheral
device><CR><LF>

[RX] – OK <CR><LF>
```

Used to create connection with remote LE device

```
[TX] - at+rsibt_getallprofiles=,<BD Address of the
peripheral device>,1,10<CR><LF>

[RX] - OK 0<LF>

1,5,2,1800<LF>

6,9,2,1801<LF>

A,C,2,180A<LF><CR><LF>
```

Used to query the entire supported profiles list from the connected remote device.

```
[TX] - at+rsibt_getcharservices=,<BD Address of the
peripheral device>,1,10<CR><LF>

[RX] - OK 4, 2,2,3,2,2A00<LF>

4,2,5,2,2A01<LF>

7,10,8,2,2A05<LF>

B,8,C,2,2A00<LF><CR><LF>
```

Used to query characteristic services, with in the particular range, from the connected remote device

```
[TX] - at+rsibt_writevalue=,<BD Address of the peripheral device>,C,A,r,e,d,p,i,n,e,l,e<CR><LF>
[RX] - OK <CR><LF>
```

Used to Set attribute value of the connected remote device.

5.5.1.2.2 Test Procedure

Choose a BLE peripheral device, not its BD address and configure it to advertise. Run the commands explained above on a RS9113 EVB to configure it in the BLE central mode, scan for nearby peripherals and connect to the desired peripheral device. Ensure to replace the `<BD Address of the peripheral device>` with the BD address of the desired peripheral device.

5.5.2 BLE Evaluation using Binary Commands

Refer to *(Package $)\RS9113.NBZ.WC.GEN.OSI.X.X.X\host\sapis\platforms\windows_uart* in the release package for the necessary projects and documents to run a binary project on Windows over the UART interface.

5.6 BT Evaluation in UART Mode

In UART Mode, the supported test methods for BT are

- Using AT Commands via a terminal Utility
- Using binary commands via Dev C++ IDE

The following sub-sections describe configuring and using the module in different security and operational modes of BT.

5.6.1 BT Evaluation using AT Commands

5.6.1.1 BT in Master Mode

5.6.1.1.1 BT Master Mode Configuration through AT commands

Configure the module to UART mode using the steps given in EVB user guide .

For the demo we have used “Bluesoleil” application.

This application works with Bluesoleil Application with Bluetooth dongle .

1. Power up the module and module sends following messages at the boot up.

```
WELCOME TO REDPINE SIGNALS<CR><LF>
BootLoader Version 1.6<CR><LF>
<CR><LF>
1 Load Image-I<CR><LF>
2 Upgrade Image-I<CR><LF>
3 Load Image-II<CR><LF>
4 Upgrade Image-II<CR><LF>
5 Set Image-I as Default<CR><LF>
```

```
6 Set Image-II as Default<CR><LF>
7 Enable GPIO Based Bypass Mode<CR><LF>
8 Disable GPIO Based Bypass Mode<CR><LF>
A LOAD FIRMWARE (Image No : 0-f)<CR><LF>
B BURN FIRMWARE (Image No : 0-f)<CR><LF>
K CHECK CRC (Image No : 0-f)<CR><LF>
D SWITCH DEFAULT IMAGE (Image No : 0-f)<CR><LF>
```

2. Select the appropriate option. As shown above to load firmware option 1(**Load Image-I**) is selected.

```
[TX] - 1
[RX] - 1<CR><LF>
```

3. Once the module loads firmware it will indicate with the message “Loading…” and “Loading Done”as shown below:

```
Loading...<CR><LF>
Loading Done<CR><LF>
```

4. First select the BT profile using the *at+rsi_opermode* command. For BT profile give the first parameter as 327680.

```
[TX] - at+rsi_opermode=327680,1,1,0<CR><LF>
[RX] - OK<CR><LF>
```

5. Now set the BT profile using the *at+rsibt_setprofilemode* command. For the SPP profile select the parameter as 1.

```
[TX] - at+rsibt_setprofilemode=1<CR><LF>
[RX] - OK<CR><LF>
```

6. Set the module name using *at+rsibt_setlocalname* command e.g. “REDPINE_BT”

```
[TX] - at+rsibt_setlocalname=10,REDPINE_BT<CR><LF>
[RX] - OK<CR><LF>
```

7. Set the discovery mode by using *at+rsibt_setdiscvmode* command. Give value 1 to set the mode.

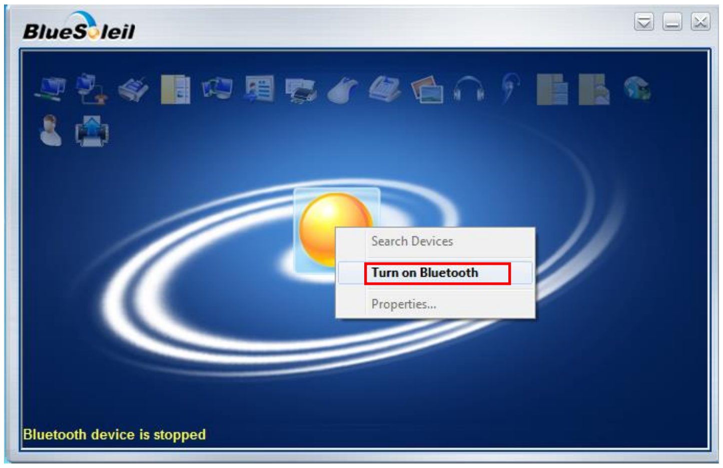
```
[TX] - at+rsibt_setdiscvmode=1<CR><LF>
[RX] - OK <CR><LF>
```

8. Set the connection mode by using *at+rsibt_setconnmode* command. Give value 1 to set the mode.

```
[TX] - at+rsibt_setconnmode=1<CR><LF>
[RX] - OK <CR><LF>
```

5.6.1.1.2 Test Procedure

1. Now open Bluesoleil Application and turn on Bluetooth.



2. Now give inquiry command to scan the Bluetooth devices which are available around the module by giving *at+rsibt_inquiry.*

```
[TX]at+rsibt_inquiry=1,10000,10<CR><LF>
```

```
[RX] - OK <CR><LF>
```

3. It will give the names of the devices which are available by giving the asynchronous messages given below.

```
AT+RSIBT_INQRESP 1, 14-30-C6-3E-BD-
E7,5,XT1033,190,5A020C<CR><LF>
```

```
AT+RSIBT_INQRESP 1,,178,5A020C<CR><LF>
```

```
AT+RSIBT_INQRESP 1,00-1B-DC-06-05-
0F,0,,164,580100<CR><LF>
```

```
AT+RSIBT_INQRESP 1,9C-2A-70-D8-EF-
A2,7,LAPT115,168,2010C<NUL>□<CR><LF>
```

```
AT+RSIBT_INQCOMPLETE<CR><LF>
```

4. Now connect to the device you want to connect by giving *at+rsi_bond* command. In that give the MAC address of device which is also shown with its name in **AT+RSIBT_INQRESP** message.

```
[TX] - at+rsibt_bond=14-30-C6-3E-BD-E7<CR><LF>
```

```
[RX] - OK <CR><LF>
```

5. It will send an asynchronous response to the module as shown below.

```
AT+RSIBT_USRLINKKEYREQ 14-30-C6-3E-BD-E7<CR><LF>
```

6. We need to send link key command *at+rsibt_usrlinkkey.*

```
[TX] - at+rsibt_ usrlinkkey =14-30-C6-3E-BD-
E7,0,1234<CR><LF>
```

```
[RX] - OK<CR><LF>
```

7. Then it will send an asynchronous response to the module as shown below.

```
AT+RSIBT_USRPINCODEREQ 14-30-C6-3E-BD-E7<CR><LF>
```

8. Module need to send an authentication code (Enter the same PIN as entered last time) e.g. 1234 here using *at+rsibt_usrpinocde* command.

```
[TX] - at+rsibt_usrpincode=14-30-C6-3E-BD-
E7,1,1234<CR><LF>
```

```
[RX] - OK<CR><LF>
```

9. As soon as you enter pin and click "OK" on this window, it will send an asynchronous response to the module as shown below.

```
AT+RSIBT_USRLINKKEYSAVE 14-30-C6-3E-BD-
E7,3,C6,4A,73,56,2B,13,47,DB,F7,11,89,55,D0,3D,20<CR><LF
>
```

```
AT+RSIBT_BONDRESP 14-30-C6-3E-BD-E7,1<CR><LF>
```

```
AT+RSIBT_UNBONDRESP 00-1A-7D-DA-71-13,1<CR><LF>
```

10. Now again give bond command

```
[TX] - at+rsibt_bond=14-30-C6-3E-BD-E7<CR><LF>
```

```
[RX] - OK <CR><LF>
```

11. It will send an asynchronous response to the module as shown below.

```
AT+RSIBT_USRLINKKEYREQ 14-30-C6-3E-BD-E7<CR><LF>
```

12. We need to send link key command *at+rsibt_usrlinkkey.*

```
[TX] - at+rsibt_ usrlinkkey =14-30-C6-3E-BD-
E7,0,1234<CR><LF>
```

```
[RX] - OK<CR><LF>
```

13. Then it will send an asynchronous response to the module as shown below.

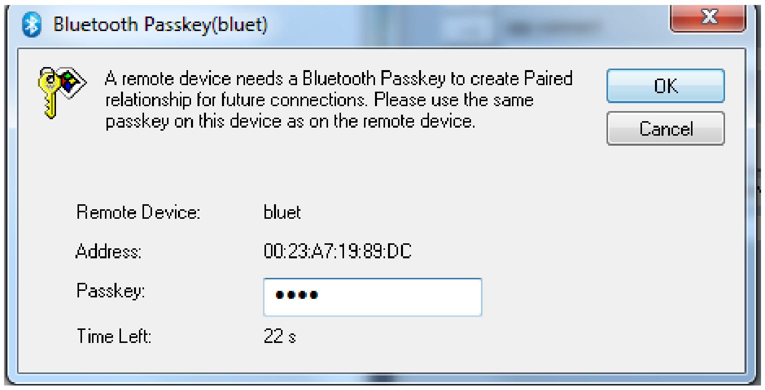
```
AT+RSIBT_USRPINCODEREQ 14-30-C6-3E-BD-E7<CR><LF>
```

14. Module need to send an authentication code (Enter the same PIN as entered last time) e.g. 1234 here using *at+rsibt_usrpinocde* command.

```
[TX] - at+rsibt_usrpincode=14-30-C6-3E-BD-
E7,1,1234<CR><LF>
```

```
[RX] - OK<CR><LF>
```

15. On doing this pop up window given below will come on the window you want to connect. give the same pass key here.


Bluetooth Passkey(bluet)

A remote device needs a Bluetooth Passkey to create Paired relationship for future connections. Please use the same passkey on this device as on the remote device.

Remote Device: bluet
Address: 00:23:A7:19:89:DC
Passkey: ••••
Time Left: 22 s

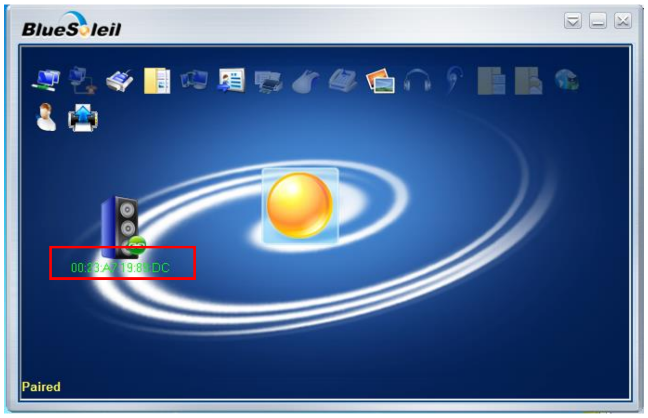
16. After entering pass key module gets bond response as shown below.

```
AT+RSIBT_USRLINKKEYSAVE 00-1A-7D-DA-71-
13,3,31,C9,7A,CB,1C,DD,F8,3A,F,EC,61,78,32,CE,54<CR><LF>
```

17. Module gets bond reponse as follows.

```
AT+RSIBT_BONDRESP 00-1A-7D-DA-71-13,1<CR><LF>
```

18. After bond reponse the module gets paired with the respective device You want to connect.



19. Now give SPP connect command from the module by giving the command.

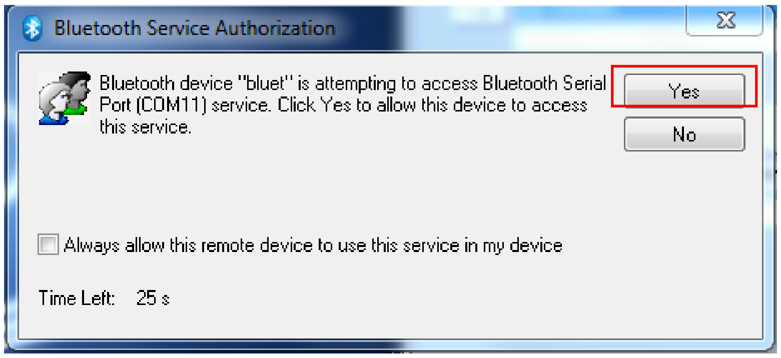
```
at+rsibt_sppconn=00-1A-7D-DA-71-13<CR><LF>
```

20. After successful connection asynchronous message from module given below will come on the terminal.

```
AT+RSIBT_USRLINKKEYSAVE 14-30-C6-3E-BD-
E7,6D,42,6E,4B,6B,E6,CF,C9,7E,48,8C,91,CD,86,F9,AC<CR><L
F>
```

```
AT+RSIBT_SPPCONNECTED 14-30-C6-3E-BD-E7<CR><LF>
```

21. Pop up window comes along with a message at the bottom of the notification window.


Bluetooth Service Authorization

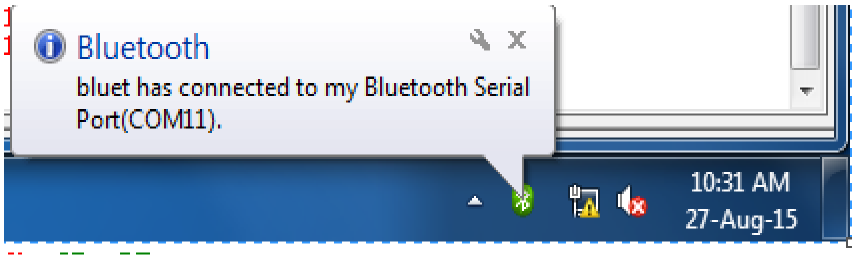
Bluetooth device "bluet" is attempting to access Bluetooth Serial Port (COM11) service. Click Yes to allow this device to access this service.

Yes

No

☐ Always allow this remote device to use this service in my device

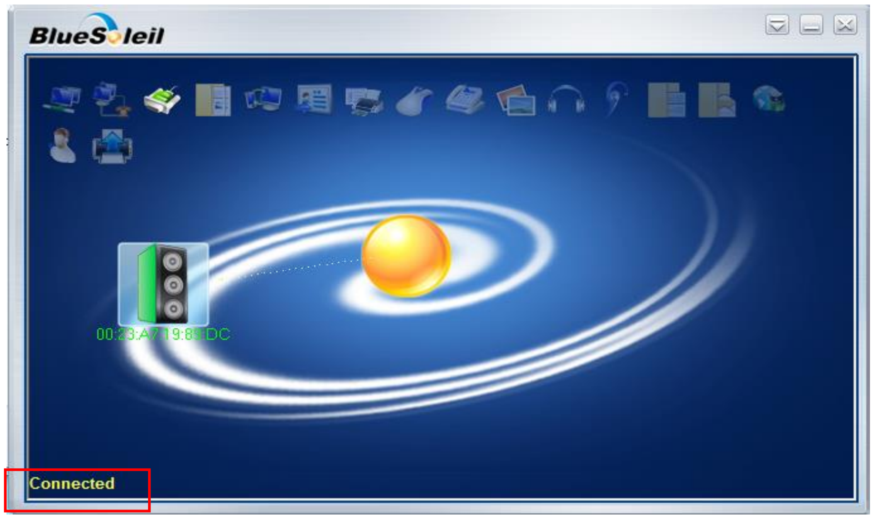
Time Left: 25 s


Bluetooth

bluet has connected to my Bluetooth Serial Port(COM11).

22. Now the Bluesoleil Application will show the status as connected.

    Then communication can be started.



23. Now open another Docklight window with comport number as shown in the above notification window(for eg.COM11).
24. Now start communicating with the other Bluetooth device through these terminals by sending the SPP TRANSFER command.

```
[TX] - at+rsibt_spptx=10,1234567890<CR><LF>
[RX] - OK<CR><LF>
```

    This string will be received on the other Docklight window.

25. In the same way we can send any random data from other terminal to the current terminal.

5.6.1.2 BT in Slave Mode

5.6.1.2.1 BT Slave Mode Configuration through AT commands

Configure the module to UART mode using the steps given in EVB user guide and an Android phone with application for the BT SPP profile support.

For the demo we have used “Bluetooth spp pro” application.

Note: This application works with Android v4.0+.

1. Power up the module and module sends following messages at the boot up.

```
WELCOME TO REDPINE SIGNALS<CR><LF>
BootLoader Version 1.6<CR><LF>
<CR><LF>
1 Load Image-I<CR><LF>
2 Upgrade Image-I<CR><LF>
3 Load Image-II<CR><LF>
4 Upgrade Image-II<CR><LF>
```

```
5 Set Image-I as Default<CR><LF>
6 Set Image-II as Default<CR><LF>
7 Enable GPIO Based Bypass Mode<CR><LF>
8 Disable GPIO Based Bypass Mode<CR><LF>
A LOAD FIRMWARE (Image No : 0-f)<CR><LF>
B BURN FIRMWARE (Image No : 0-f)<CR><LF>
K CHECK CRC (Image No : 0-f)<CR><LF>
D SWITCH DEFAULT IMAGE (Image No : 0-f)<CR><LF>
```

2. Select the appropriate option. As shown above to load firmware option 1(**Load Image-I**) is selected.

```
[TX] - 1
[RX] - 1<CR><LF>
```

3. Once the module loads firmware it will indicate with the message “Loading…” and “Loading Done”as shown below:

```
Loading...<CR><LF>
Loading Done<CR><LF>
```

4. First select the BT profile using the *at+rsi_opermode* command. For BT profile give the first parameter as 327680.

```
[TX] - at+rsi_opermode=327680,1,1,0<CR><LF>
[RX] - OK<CR><LF>
```

5. Now set the BT profile using the *at+rsibt_setprofilemode* command. For the SPP profile select the parameter as 1.

```
[TX] - at+rsibt_setprofilemode=1<CR><LF>
[RX] - OK<CR><LF>
```

6. Set the module name using *at+rsibt_setlocalname* command e.g. “REDPINE_BT”

```
[TX] - at+rsibt_setlocalname=10,REDPINE_BT<CR><LF>
[RX] - OK<CR><LF>
```

7. Set the discovery mode by using *at+rsibt_setdiscvmode* command. Give value 1 to set the mode.

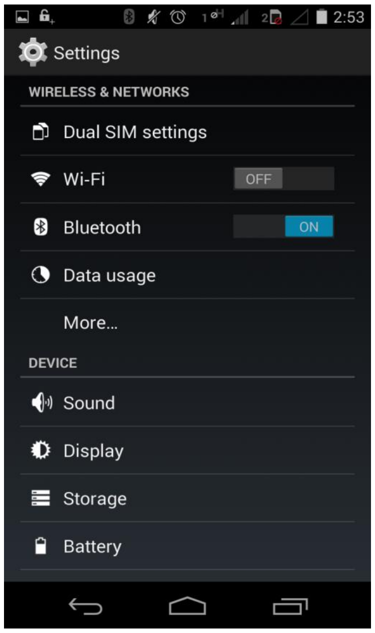
```
[TX] - at+rsibt_setdiscvmode=1<CR><LF>
[RX] - OK <CR><LF>
```

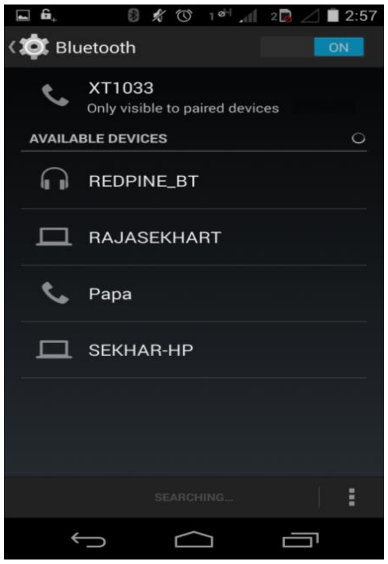
8. Set the connection mode by using *at+rsibt_setconnmode* command. Give value 1 to set the mode.

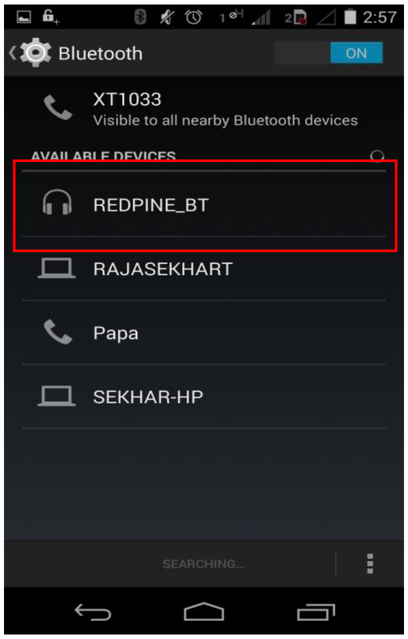
```
[TX] - at+rsibt_setconnmode=1<CR><LF>
[RX] - OK <CR><LF>
```

5.6.1.2.2 Test Procedure

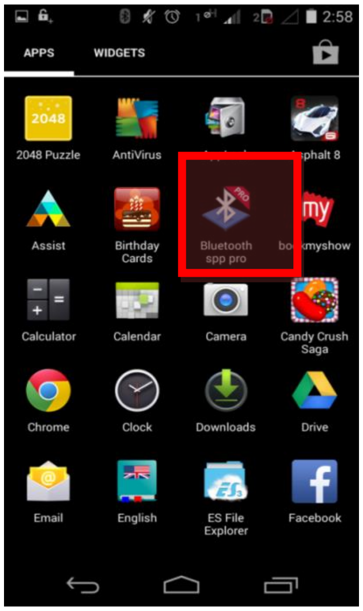
1. In the Android phone go to ***Settings-> Wireless and network-> Bluetooth Settings***. Turn on Bluetooth set device and make it visible.



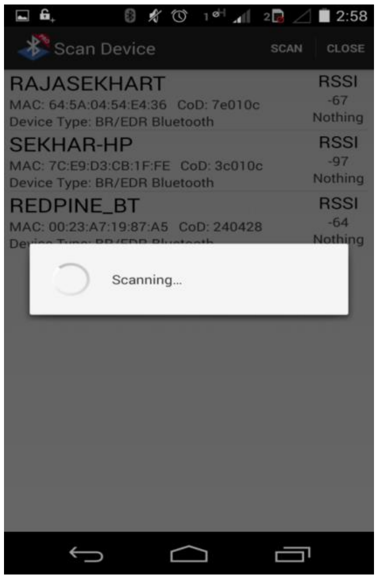




2. Now open the “Bluetooth spp pro” application.



3. In the “Bluetooth spp pro” application, as soon as you open application it starts scanning other Bluetooth devices.



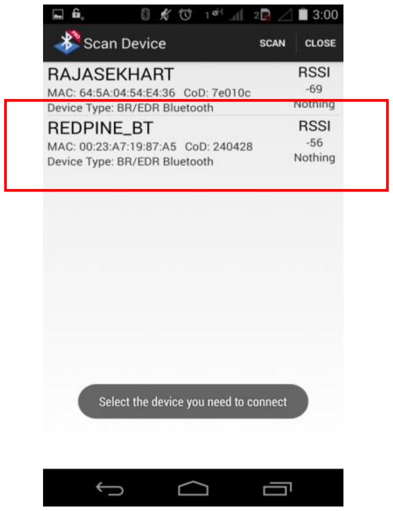
RAJASEKHART — RSSI -67 Nothing
MAC: 64:5A:04:54:E4:36 CoD: 7e010c
Device Type: BR/EDR Bluetooth

SEKHAR-HP — RSSI -97 Nothing
MAC: 7C:E9:D3:CB:1F:FE CoD: 3c010c
Device Type: BR/EDR Bluetooth

REDPINE_BT — RSSI -64 Nothing
MAC: 00:23:A7:19:87:A5 CoD: 240428

Scanning...

4. It will show all the nearby devices as well as option to scan devices again.

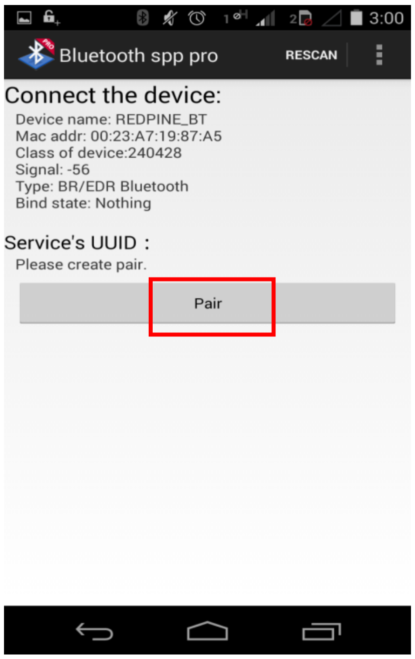


RAJASEKHART — RSSI -69 Nothing
MAC: 64:5A:04:54:E4:36 CoD: 7e010c
Device Type: BR/EDR Bluetooth

REDPINE_BT — RSSI -56 Nothing
MAC: 00:23:A7:19:87:A5 CoD: 240428
Device Type: BR/EDR Bluetooth

Select the device you need to connect

5. If the module name is not appearing then scan again till you find module in the list. Once the module is shown in the list e.g. “REDPINE_BT” select that.
6. After selecting “REDPINE_BT” you will see following screen:


3:00
Bluetooth spp pro RESCAN

Connect the device:

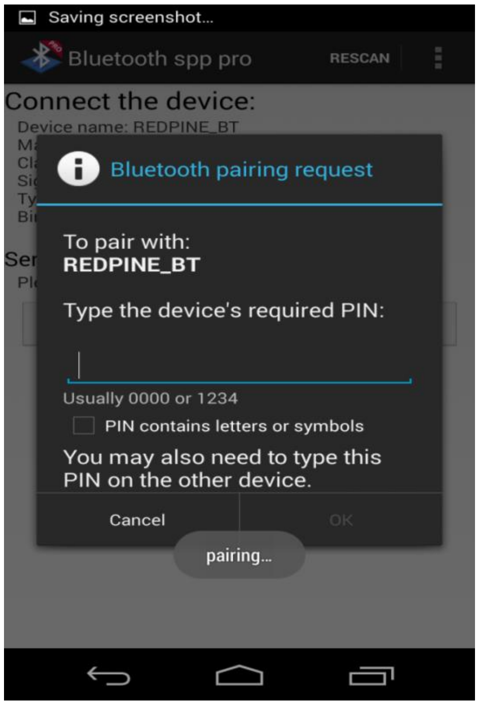
Device name: REDPINE_BT
Mac addr: 00:23:A7:19:87:A5
Class of device:240428
Signal: -56
Type: BR/EDR Bluetooth
Bind state: Nothing

Service's UUID :

Please create pair.

Pair

Now click on “Pair”. It will give a pop-up on phone as shown below:


Saving screenshot...

Bluetooth spp pro RESCAN

Connect the device:

Device name: REDPINE_BT

Bluetooth pairing request

To pair with:
REDPINE_BT

Type the device's required PIN:

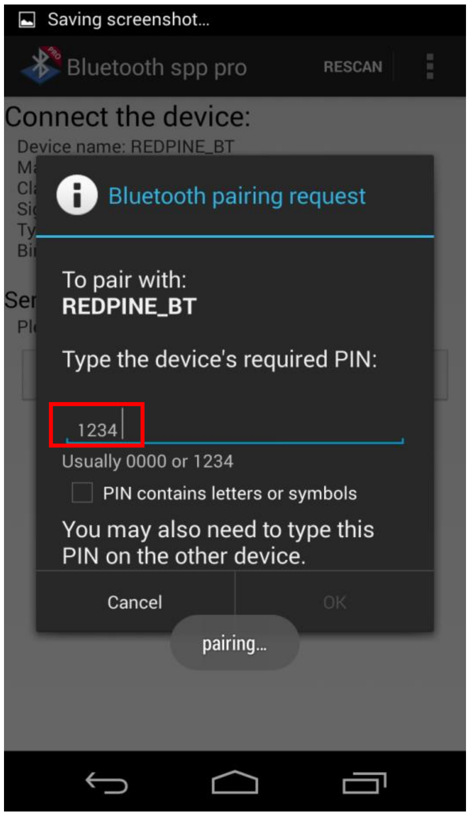
Usually 0000 or 1234

PIN contains letters or symbols

You may also need to type this PIN on the other device.

Cancel OK

pairing...



To pair with:
REDPINE_BT

Type the device's required PIN:

1234

Usually 0000 or 1234

PIN contains letters or symbols

You may also need to type this PIN on the other device.

7. As soon as you enter pin and click "OK" on this window, it will send an asynchronous response to the module as shown below.

```
AT+RSIBT_USRPINCODEREQ 14-30-C6-3E-BD-E7<CR><LF>
```

Module need to send an authentication code (Enter the same PIN) e.g. 1234 here using *at+rsibt_usrpinocde* command.

```
[TX] - at+rsibt_usrpincode=14-30-C6-3E-BD-
E7,1,1234<CR><LF>
```

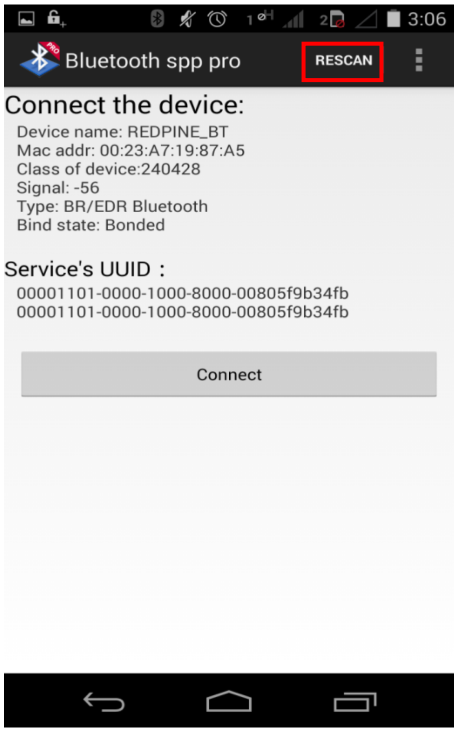
```
[RX] - OK<CR><LF>
```

8. If the authentication is successful it will be connected and below asynchronous response will be received.

```
AT+RSIBT_USRLINKKEYSAVE 14-30-C6-3E-BD-
E7,3,C6,4A,73,56,2B,13,47,DB,F7,11,89,55,D0,3D,20<CR><LF
>
```

```
AT+RSIBT_SPPCONNECTED 14-30-C6-3E-BD-E7<CR><LF>
```

9. Now press the RESCAN button. It will disconnect from REDPINE_BT and an asynchronous disconnect response can be seen on module side.



Connect the device:

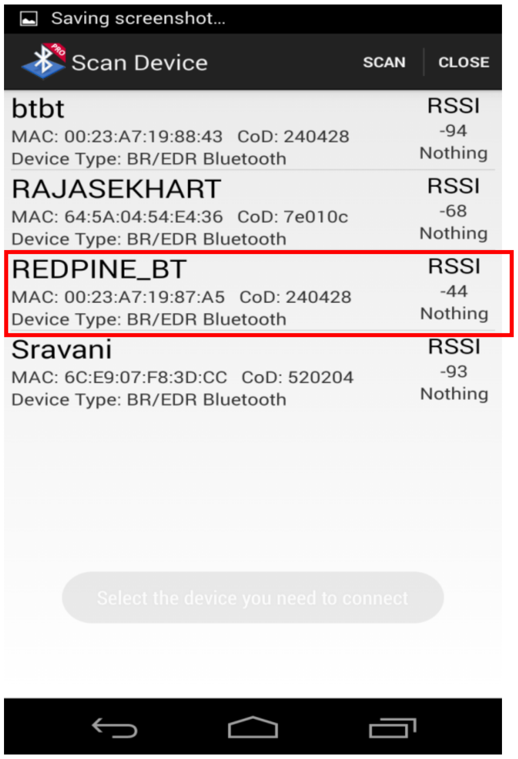
Device name: REDPINE_BT
Mac addr: 00:23:A7:19:87:A5
Class of device:240428
Signal: -56
Type: BR/EDR Bluetooth
Bind state: Bonded

Service's UUID :

00001101-0000-1000-8000-00805f9b34fb
00001101-0000-1000-8000-00805f9b34fb

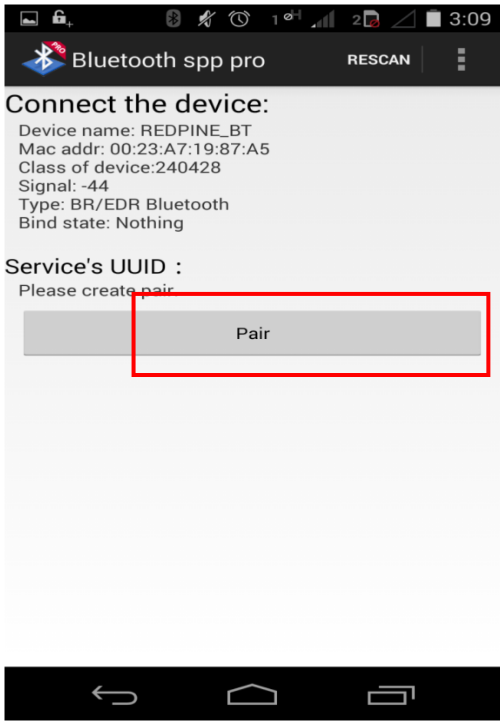
AT+RSIBT_DISCONNECTED 14-30-C6-3E-BD-E7,0 <CR><LF>

10. Phone will again start scanning the BT devices. Look if "REDPINE_BT" is there in the scan list or not. If not then rescan again and again until the device (module) is scanned in the scan results. Select the "REDPINE_BT" device.



btbt MAC: 00:23:A7:19:88:43 CoD: 240428 Device Type: BR/EDR Bluetooth	RSSI -94 Nothing
RAJASEKHART MAC: 64:5A:04:54:E4:36 CoD: 7e010c Device Type: BR/EDR Bluetooth	RSSI -68 Nothing
REDPINE_BT MAC: 00:23:A7:19:87:A5 CoD: 240428 Device Type: BR/EDR Bluetooth	RSSI -44 Nothing
Sravani MAC: 6C:E9:07:F8:3D:CC CoD: 520204 Device Type: BR/EDR Bluetooth	RSSI -93 Nothing

11. Now click on “Pair” button. This time Phone will not ask for PIN again as it has saved the PIN provided last time and will Pair with “REDPINE_BT”.



3:09

Bluetooth spp pro

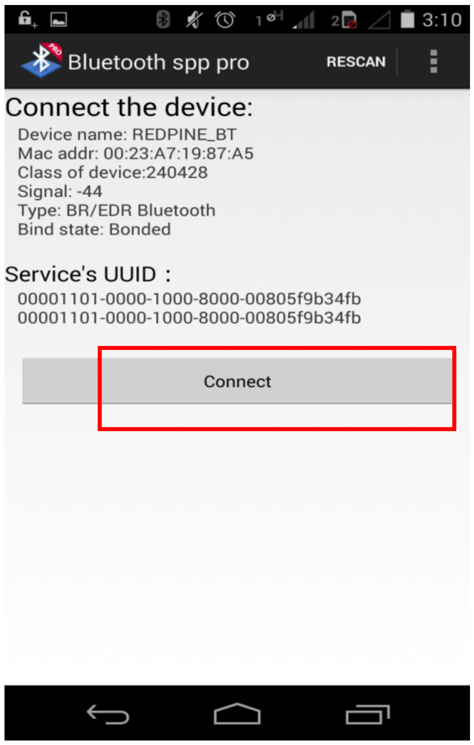
Connect the device:

Device name: REDPINE_BT
Mac addr: 00:23:A7:19:87:A5
Class of device:240428
Signal: -44
Type: BR/EDR Bluetooth
Bind state: Nothing

Service's UUID :

Please create pair.

12. Now click on “Connect” button.



Bluetooth spp pro

RESCAN

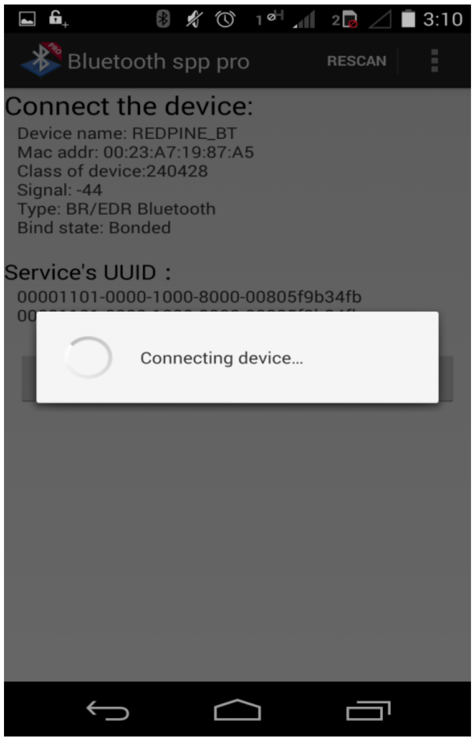
Connect the device:

Device name: REDPINE_BT
Mac addr: 00:23:A7:19:87:A5
Class of device:240428
Signal: -44
Type: BR/EDR Bluetooth
Bind state: Bonded

Service's UUID :

00001101-0000-1000-8000-00805f9b34fb
00001101-0000-1000-8000-00805f9b34fb

Connect



Bluetooth spp pro

RESCAN

Connect the device:

Device name: REDPINE_BT
Mac addr: 00:23:A7:19:87:A5
Class of device:240428
Signal: -44
Type: BR/EDR Bluetooth
Bind state: Bonded

Service's UUID :

00001101-0000-1000-8000-00805f9b34fb

Connecting device...

13. It will send an asynchronous response to the module as shown below.

```
AT+RSIBT_USRLINKKEYREQ 14-30-C6-3E-BD-E7<CR><LF>
```

14. We need to send link key command *at+rsibt_usrlinkkey.*

```
[TX] - at+rsibt_ usrlinkkey =14-30-C6-3E-BD-
E7,0,1234<CR><LF>
```

```
[RX] - OK<CR><LF>
```

15. Then it will send an asynchronous response to the module as shown below.

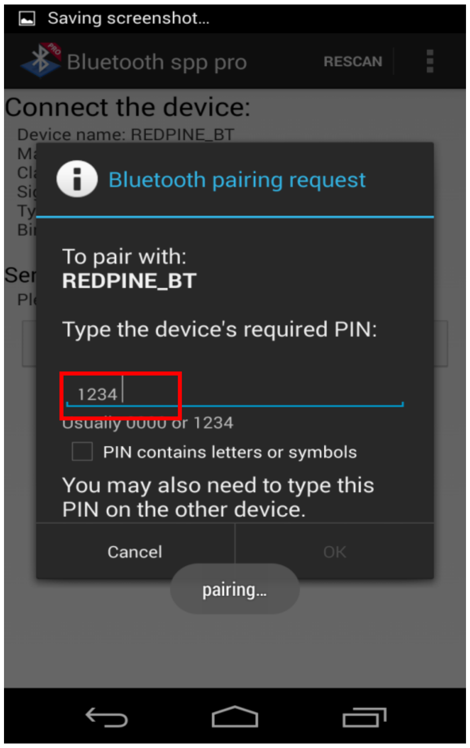
```
AT+RSIBT_USRPINCODEREQ 14-30-C6-3E-BD-E7<CR><LF>
```

16. Module need to send an authentication code (Enter the same PIN as entered last time) e.g. 1234 here using *at+rsibt_usrpinocde* command.

```
[TX] - at+rsibt_usrpincode=14-30-C6-3E-BD-
E7,1,1234<CR><LF>
```

```
[RX] - OK<CR><LF>
```

17. This time Phone will get an incoming Bluetooth request from module. Accept this request and enter the PIN (Enter same PIN as entered in the userpincode command) and then Phone will connect with “REDPINE_BT” as SPP.



To pair with:
REDPINE_BT

Type the device's required PIN:

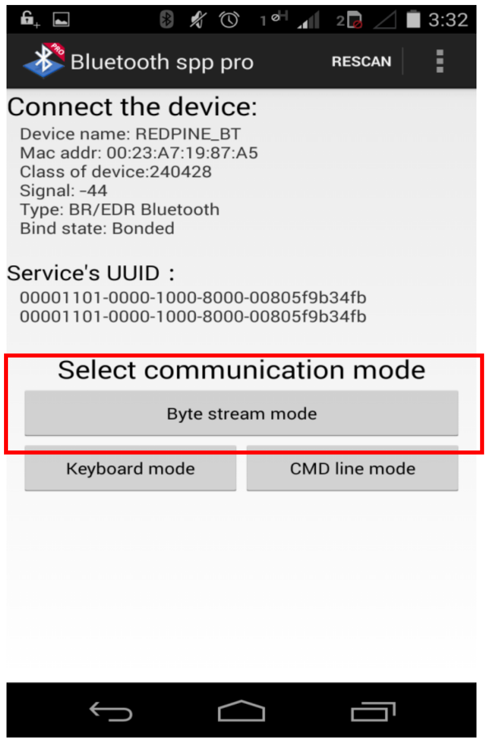
1234

Usually 0000 or 1234

☐ PIN contains letters or symbols

You may also need to type this PIN on the other device.

18. After successful connection a screen for selecting communication mode will come:



Connect the device:

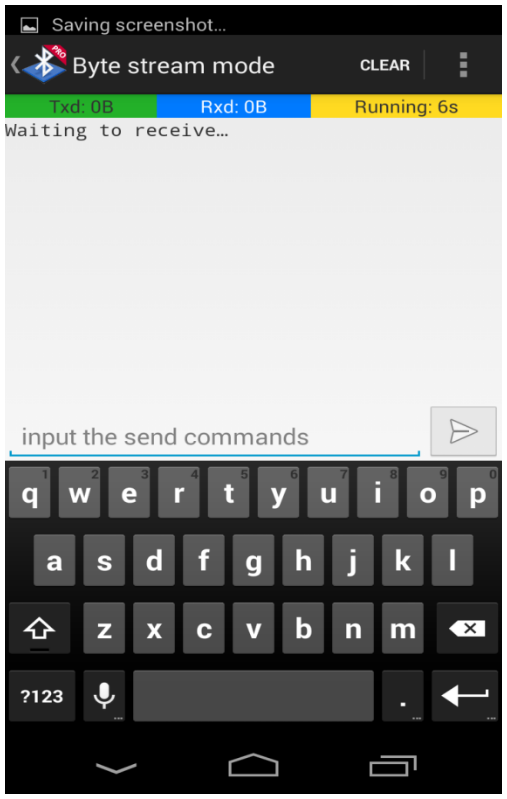
Device name: REDPINE_BT
Mac addr: 00:23:A7:19:87:A5
Class of device:240428
Signal: -44
Type: BR/EDR Bluetooth
Bind state: Bonded

Service's UUID :

00001101-0000-1000-8000-00805f9b34fb
00001101-0000-1000-8000-00805f9b34fb

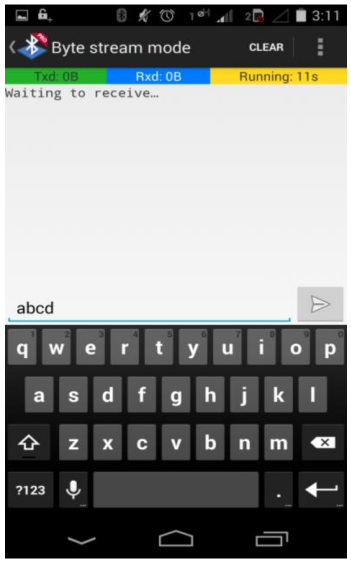
Select "Byte Stream mode" on that screen.

19. Selection of Byte Stream mode will open a communication window as shown below:



```
Waiting to receive…
```

20. Now send some bytes (e.g. string “abcd” here) from Phone to module , by typing abcd and pressing arrow to send:



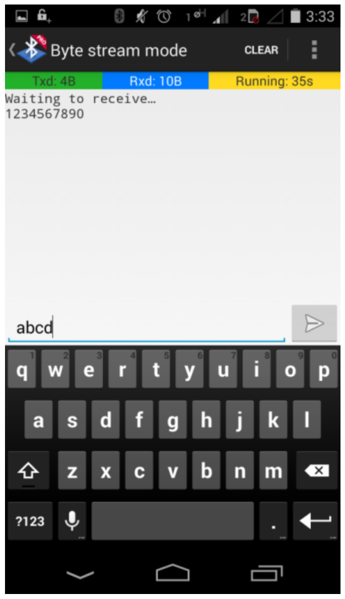
This string will be received on module as *SPPRX* as shown below:

```
AT+RSIBT_SPPRX 4<CR><LF>
abcd<CR><LF>
```

21. Similarly module can also send some bytes to phone by giving *SPPTX* command. Give the data you want to send in the 2nd argument of this command. for ex.,let the data is 1234567890.

```
[TX] – at+rsibt_spptx=10,1234567890<CR><LF>
[RX] – OK<CR><LF>
```

This string will be received on Phone as shown below:



5.6.2 BT Evaluation using Binary Commands

Refer to *(Package $)\RS9113.NBZ.WC.GEN.OSI.X.X.X\host\sapis\platforms\windows_uart* in the release package for the necessary projects and documents to run a binary project on Windows over the UART interface.

5.7 Wi-Fi + BLE Evaluation in UART Mode

In UART Mode, the supported test methods for Wi-Fi + BLE are

- Using AT Commands via a terminal Utility
- Using binary commands via Dev C++ IDE

The following sub-sections describe configuring and using the module in different operational modes of Wi-Fi + BLE.

5.7.1 Wi-Fi + BLE Evaluation using AT Commands

5.7.1.1 Wi-Fi Client + BLE Peripheral Mode

5.7.1.1.1 Wi-Fi Client + BLE Peripheral Mode Configuration

In order to run the Wi-Fi client and BT-LE coexistence mode, issue the operating mode as the first command with the relevant coexistence parameters. After the operating mode command the module will operate in both WiFi and BT LE mode and will respond to WiFi commands as well as BT LE commands in parallel .

Common Command: Set the operating mode command with below parameters to run in Wi-Fi + BT-LE Coex Mode.
Oper_mode = ((wifi_oper_mode) | (coex_mode << 16))
Wifi_oper_mode = 0 (to operate wifi in STA mode)
Coex_mode=13(to operate in WiFi+BT LE coex mode)
Feature_bit_map = 1(to operate WiFi in open security mode)
Tcp_ip_feature_bit_map = As required
Custom_feature_bit_map=0

```
at+rsi_opermode=851968,0,4,0
```

Wi-Fi Command Sequence to Associate with Access Point:

- **Band :** This command sets the operating mode of the module
- **Init :**This command initializes the module
- **Scan:** This command scans for Access points in the vicinity and reports them.
- **Join :** This command associates the module to the AP

Refer to Sections Wi-Fi Client in Personal Security Mode for the full list of commands and corresponding descriptions.

BLE Command Sequence:

- **Advertise:** This command advertises the local name of the module in peripheral role.

Refer to Sections BLE in Peripheral (Slave) Mode for the full list of commands and corresponding descriptions.

5.7.1.1.2 Test Procedure

Refer to Sections Wi-Fi Client in Personal Security Mode and BLE in Peripheral (Slave) Mode for the respective test procedures.

5.7.1.2 Wi-Fi Client + BLE Central Mode

5.7.1.2.1 Wi-Fi Client + BLE Central Mode Configuration

In order to run the Wi-Fi client and BT-LE coexistence mode, issue the operating mode as the first command with the relevant coexistence parameters. After the operating mode command the module will operate in both WiFi and BT LE mode and will respond to WiFi commands as well as BT LE commands in parallel .

Common Command: Set the operating mode command with below parameters to run in Wi-Fi + BT-LE Coex Mode.
Oper_mode = ((wifi_oper_mode) | (coex_mode << 16))
Wifi_oper_mode = 0 (to operate wifi in STA mode)
Coex_mode=13(to operate in WiFi+BT LE coex mode)
Feature_bit_map = 1(to operate WiFi in open security mode)
Tcp_ip_feature_bit_map =As required
Custom_feature_bit_map=0

```
at+rsi_opermode=851968,0,4,0
```

Wi-Fi Command Sequence to Associate with Access Point:

- **Band :** This command sets the operating mode of the module
- **Init :**This command initializes the module
- **Scan:** This command scans for Access points in the vicinity and reports them.
- **Join :** This command associates the module to the AP

Refer to Sections Wi-Fi Client in Personal Security Mode for the full list of commands and corresponding descriptions.

BLE Command Sequence:

- **Scan:** This command scans for BT LE devices and reports the devices found
- **Connect:** This command initializes a connection to a remote peripheral device..

Refer to Sections BLE in Central (Master) Mode for the full list of commands and corresponding descriptions.

5.7.1.2.2 Test Procedure

Refer to Sections Wi-Fi Client in Personal Security Mode and BLE in Central (Master) Mode for the respective test procedures.

5.7.1.3 Wi-Fi AP + BLE Peripheral Mode

5.7.1.3.1 Wi-Fi AP + BLE Peripheral Mode Configuration

In order to run the Wi-Fi AP and BT-LE coexistence mode, issue the operating mode as the first command with the relevant coexistence parameters. After the operating mode command the module will operate in both WiFi and BT LE mode and will respond to WiFi commands as well as BT LE commands in parallel.

Common Command: Set the operating mode command with below parameters to run in Wi-Fi + BT-LE Coex Mode.
Oper_mode = ((wifi_oper_mode) | (coex_mode << 16))
Wifi_oper_mode = 6 (to operate wifi in STA mode)
Coex_mode=13(to operate in WiFi+BT LE coex mode)
Feature_bit_map = 1(to operate WiFi in open security mode)
Tcp_ip_feature_bit_map =1(Bypass the TCp/IP stack on the module)
Custom_feature_bit_map=0

```
at+rsi_opermode=851974,0,18,0
```

Wi-Fi Command Sequence to Associate with Access Point:

- **Band :** This command sets the operating mode of the module
- **Init :**This command initializes the module
- **IPConfig:** This command configures the DHCP server on the module.
- **APConfig:** This command configures the AP related parameters on the module.

Refer to Sections Wi-Fi Access Point Mode for the full list of commands and corresponding descriptions.

BLE Command Sequence:

- **Advertise:** This command advertises the local name of the module in peripheral role.

Refer to Sections BLE in Peripheral (Slave) Mode for the full list of commands and corresponding descriptions.

5.7.1.3.2 Test Procedure

Refer to Sections Wi-Fi Access Point Mode for and BLE in Peripheral (Slave) Mode for the respective test procedures.

5.7.1.4 Wi-Fi AP + BLE Central Mode

5.7.1.4.1 Wi-Fi AP + BLE Peripheral Mode Configuration

In order to run the Wi-Fi AP and BT-LE coexistence mode, issue the operating mode as the first command with the relevant coexistence parameters. After the operating mode command the module will operate in both WiFi and BT LE mode and will respond to WiFi commands as well as BT LE commands in parallel.

Common Command: Set the operating mode command with below parameters to run in Wi-Fi + BT-LE Coex Mode.
Oper_mode = ((wifi_oper_mode) | (coex_mode << 16))
Wifi_oper_mode = 6 (to operate wifi in STA mode)
Coex_mode=13(to operate in WiFi+BT LE coex mode)
Feature_bit_map = 1(to operate WiFi in open security mode)
Tcp_ip_feature_bit_map =1(Bypass the TCP/IP stack on the module)
Custom_feature_bit_map=0

```
at+rsi_opermode=851974,0,18,0
```

Wi-Fi Command Sequence to Associate with Access Point:

- **Band :** This command sets the operating mode of the module
- **Init :**This command initializes the module
- **IPConfig:** This command configures the DHCP server on the module.
- **APConfig:** This command configures the AP related parameters on the module.

Refer to Sections Wi-Fi Access Point Mode for the full list of commands and corresponding descriptions.

BLE Command Sequence:

- **Scan:** This command scans for BT LE devices and reports the devices found
- **Connect:** This command initializes a connection to a remote peripheral device.

Refer to Sections BLE in Central (Master) Mode for the full list of commands and corresponding descriptions.

5.7.1.4.2 Test Procedure

Refer to Sections Wi-Fi Access Point Mode for and BLE in Central (Master) Mode for the respective test procedures.

5.7.2 Wi-Fi + BLE Evaluation using Binary Commands

Refer to *(Package $)\RS9113.NBZ.WC.GEN.OSI.X.X.X\host\sapis\platforms\windows_uart* in the release package for the necessary projects and documents to run a binary project on Windows over the UART interface.

6 Configuration over Wi-Fi

The module can be configured wirelessly to join a specific AP (referred to as "auto-connect") or create an Access Point (referred to as "auto-create").

6.1 Configuration to Connect to an Access Point

Flow 1: In this flow, an AP is first created in the module, to which a remote device connects and configures the module.

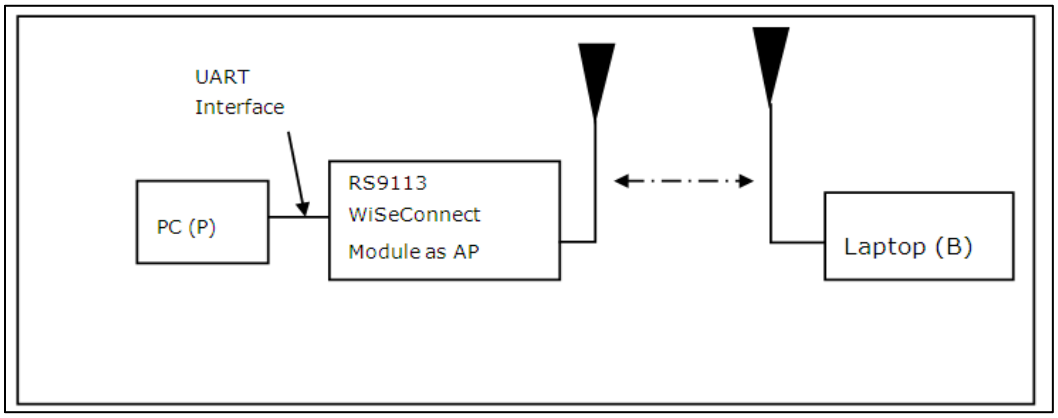
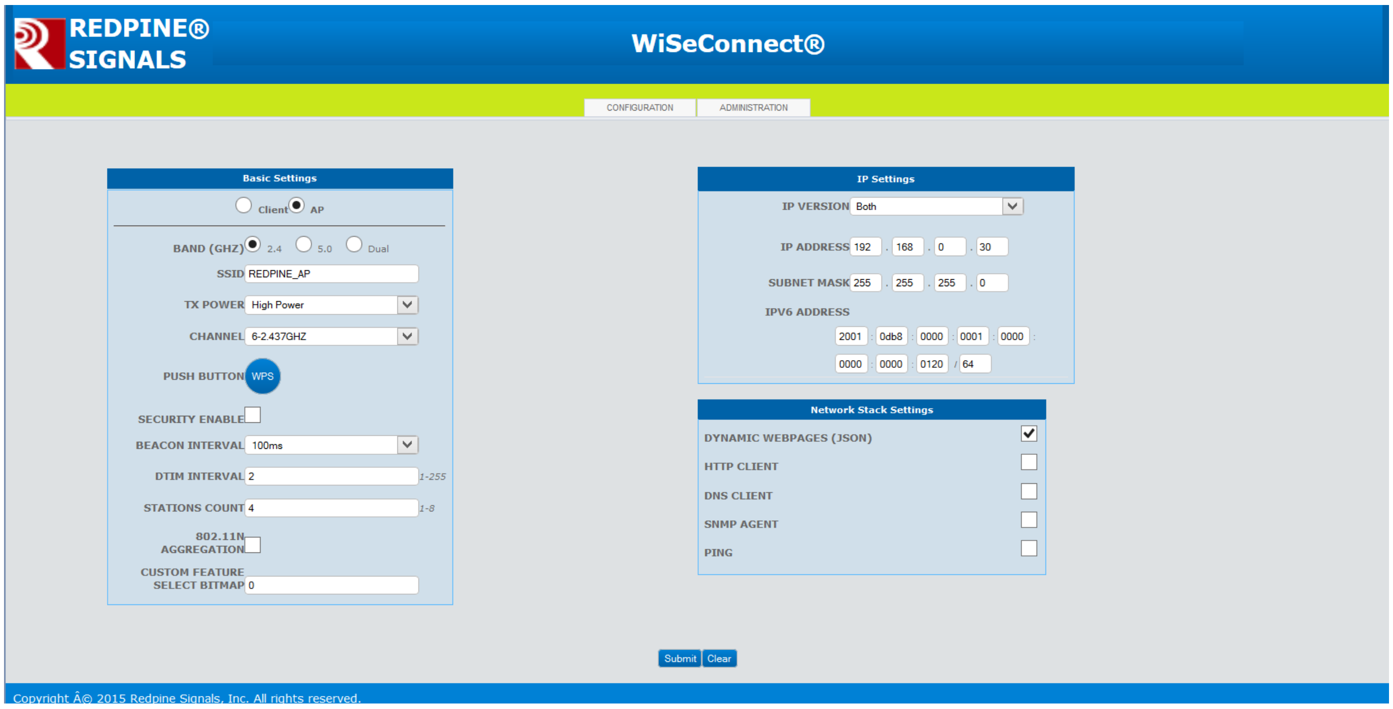


Figure 21: Module Configured as Access Point

1) Connect a PC or Host to the module through the USB and power up the module.
2) Configure the module to become an AP and DHCP Server by issuing commands from PC (P) as in section Evaluation of Access Point Mode. The sequence of commands is given below.
   - `at+rsi_opermode=6,1,18,0`
   - `at+rsi_band=0`
   - `at+rsi_init`
   - `at+rsi_fwversion?`
   - `at+rsi_ipconf=0,192.168.0.30,255.255.255.0,192.168.0.30`
   - `at+rsi_apconf=6,REDPINE_AP,0,0,0,100,3,4`
   - `at+rsi_join=REDPINE_AP,0,2,0`

   The module is now configured as an Access Point. Its IP address is **192.168.0.30.**
3) Connect a Laptop (B) to the created AP. Open the URL **http://<Module's IP address>**in the Laptop. In this case, the URL is http://192.168.0.30. Make sure the browser in the laptop does not have any proxies enabled.
4) The module's webpage opens as shown in the figure below.



Basic Settings

[✓ Client] AP (first screenshot: Client [✓ AP])

- BAND (GHZ): [✓ 2.4] 5.0 Dual
- SSID: REDPINE_AP
- TX POWER: High Power
- CHANNEL: 6-2.437GHZ
- PUSH BUTTON: WPS
- SECURITY ENABLE: ☐
- BEACON INTERVAL: 100ms
- DTIM INTERVAL: 2 (1-255)
- STATIONS COUNT: 4 (1-8)
- 802.11N AGGREGATION: ☐
- CUSTOM FEATURE SELECT BITMAP: 0

IP Settings

- IP VERSION: Both
- IP ADDRESS: 192 . 168 . 0 . 30
- SUBNET MASK: 255 . 255 . 255 . 0
- IPV6 ADDRESS: 2001 : 0db8 : 0000 : 0001 : 0000 : 0000 : 0000 : 0120 / 64

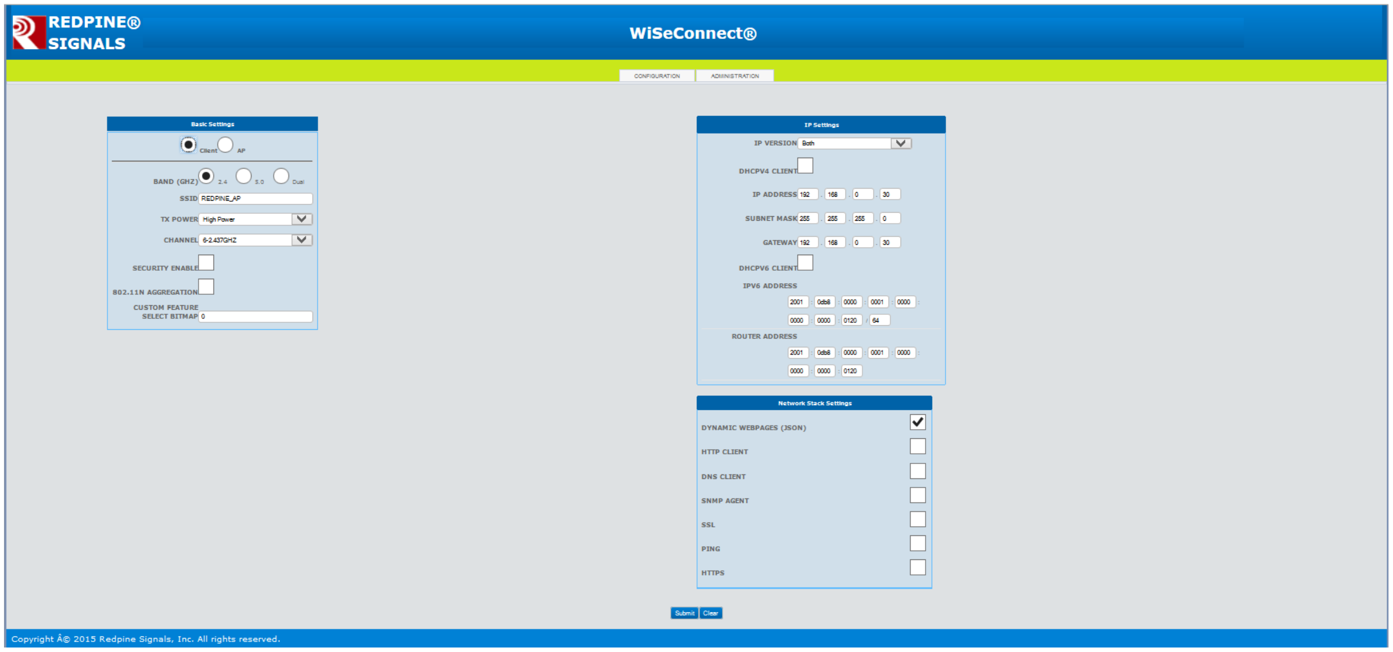
Network Stack Settings

- DYNAMIC WEBPAGES (JSON): ✓
- HTTP CLIENT: ☐
- DNS CLIENT: ☐
- SNMP AGENT: ☐
- PING: ☐

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

Figure 22: Module Webpage

5) In the page, select "Client" under Basic Settings. The page changes to show the settings required to be configured for the module in Client mode, as shown in the figure below.



Basic Settings

[✓ Client] AP

- BAND (GHZ): [✓ 2.4] 5.0 Dual
- SSID: REDPINE_AP
- TX POWER: High Power
- CHANNEL: 6-2.437GHZ
- SECURITY ENABLE: ☐
- 802.11N AGGREGATION: ☐
- CUSTOM FEATURE SELECT BITMAP: 0

IP Settings

- IP VERSION: Both
- DHCPV4 CLIENT: ☐
- IP ADDRESS: 192 . 168 . 0 . 30
- SUBNET MASK: 255 . 255 . 255 . 0
- GATEWAY: 192 . 168 . 0 . 30
- DHCPV6 CLIENT: ☐
- IPV6 ADDRESS: 2001 : 0db8 : 0000 : 0001 : 0000 : 0000 : 0000 : 0120 / 64
- ROUTER ADDRESS: 2001 : 0db8 : 0000 : 0001 : 0000 : 0000 : 0000 : 0120

Network Stack Settings

- DYNAMIC WEBPAGES (JSON): ✓
- HTTP CLIENT: ☐
- DNS CLIENT: ☐
- SNMP AGENT: ☐
- SSL: ☐
- PING: ☐
- HTTPS: ☐

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

Figure 23: Client Mode Webpage

6) Select the Band and enter SSID of the Access Point to which the module needs to be connected to.

7) Set the "TX POWER" to "High Power".

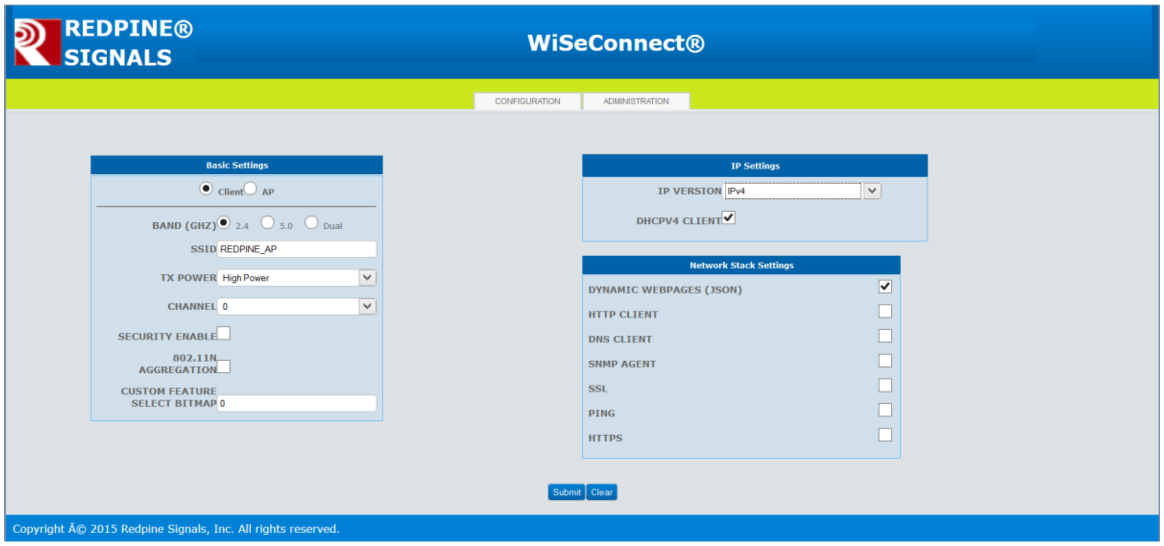
8) Select "0" in the drop down menu for Channel – this ensures that the module will scan in all the channels for the Access Point.

9) If the Access Point is in Secure mode, click the radio button next to "SECURITY ENABLE" and then input the security parameters as per the Access Point's settings. In this example, the Access Point is in Open (non-Secure) mode.

10) Click the radio button next to "802.11N AGGREGATION" to enable A-MPDU aggregation.

11) For details on the "CUSTOM FEATURE BITMAP", refer to the Software PRM.

12) Under IP Settings, select between IPv4, IPv6 and Both and enter the IP Settings (DHCP Enable/Disable) as per the settings in the Access Point. In this example, we have selected IPv4 and enabled DHCP, as shown in the figure below.



Basic Settings

[● Client] ○ AP

BAND (GHZ)	[● 2.4] ○ 5.0 ○ Dual
SSID	REDPINE_AP
TX POWER	High Power
CHANNEL	0
SECURITY ENABLE	☐
802.11N AGGREGATION	☐
CUSTOM FEATURE SELECT BITMAP	0

IP Settings

IP VERSION	IPv4
DHCPV4 CLIENT	✓

Network Stack Settings

DYNAMIC WEBPAGES (JSON)	✓
HTTP CLIENT	☐
DNS CLIENT	☐
SNMP AGENT	☐
SSL	☐
PING	☐
HTTPS	☐

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

Figure 24: Example Client Mode Configuration

13) Click on "Submit" button. The information is sent to the module and stored in its internal flash.

14) The module should now be power cycled or hard reset. It boots up and then automatically scans channels for the target AP and connects to it and gets an IP address. The module will send out two responses to the Host (PC), the first corresponds to the internally given "Join" command and the second to the "Set IP Parameters" command. Note that once the module is restarted, no commands need to be given. The module automatically scans and joins the target AP, after which the stored configuration parameters can be retrieved using the command "at+rsi_cfgget". If the auto-connect feature needs to be disabled, issue the command "at+rsi_cfgenable=0". Refer to the PRM for more details on these commands.

Flow 2: In this flow, the module is connected to an AP. A remote device connects to the same AP and configures the module.

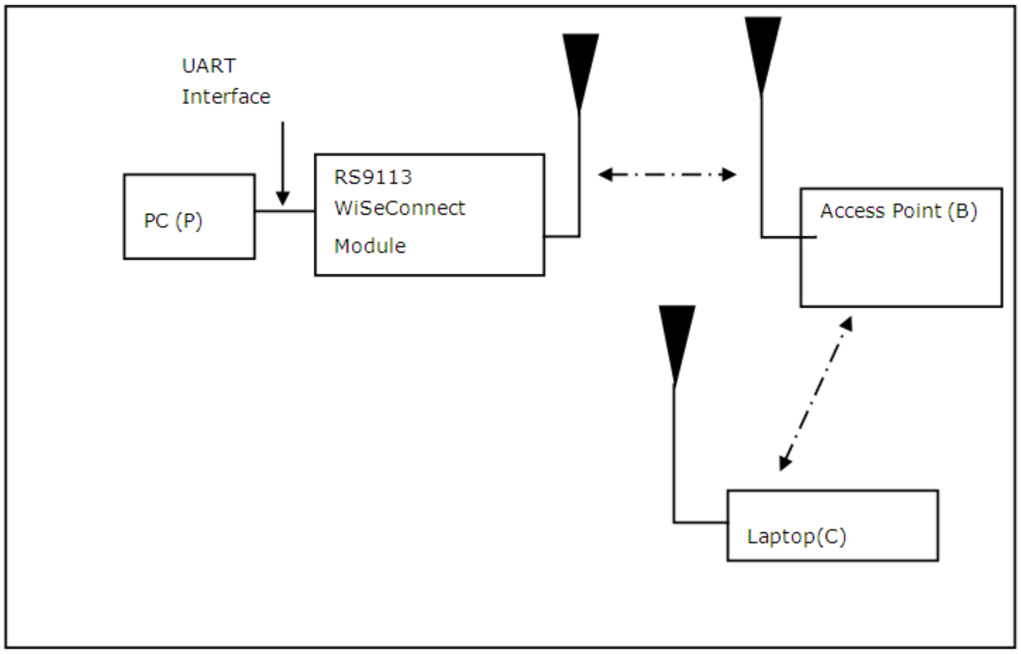


Figure 25: Module Connected to Access Point

1) Connect a PC or Host to the module through the USB and power up the module.
2) Configure the module to become a client and connect to an AP as described in section Wi-Fi Client in Personal Security Mode.
3) Connect a Laptop (C) to the same AP. Open the URL http://<Module's IP address>in the Laptop. For example, if the module was configured to have an IP of 192.168.100.20, then the URL is http://192.168.100.20. Make sure the browser in the laptop does not have any proxies enabled.
4) Follow the instructions in Flow 1 from step 4 to reconfigure the module to connect to a different Access Point.

6.1.1 Configuration to Create an Access Point

Flow 1: In this flow, an AP is first created in the module, to which a remote device connects and configures the module.

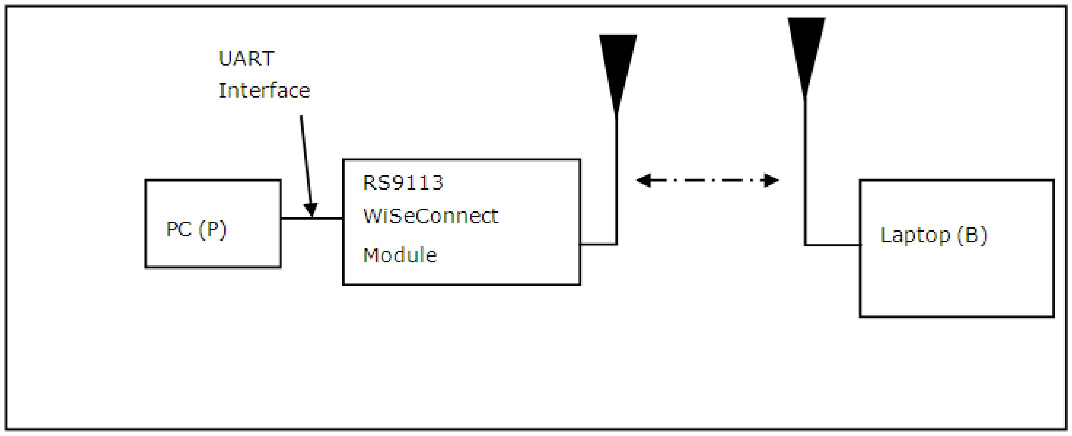


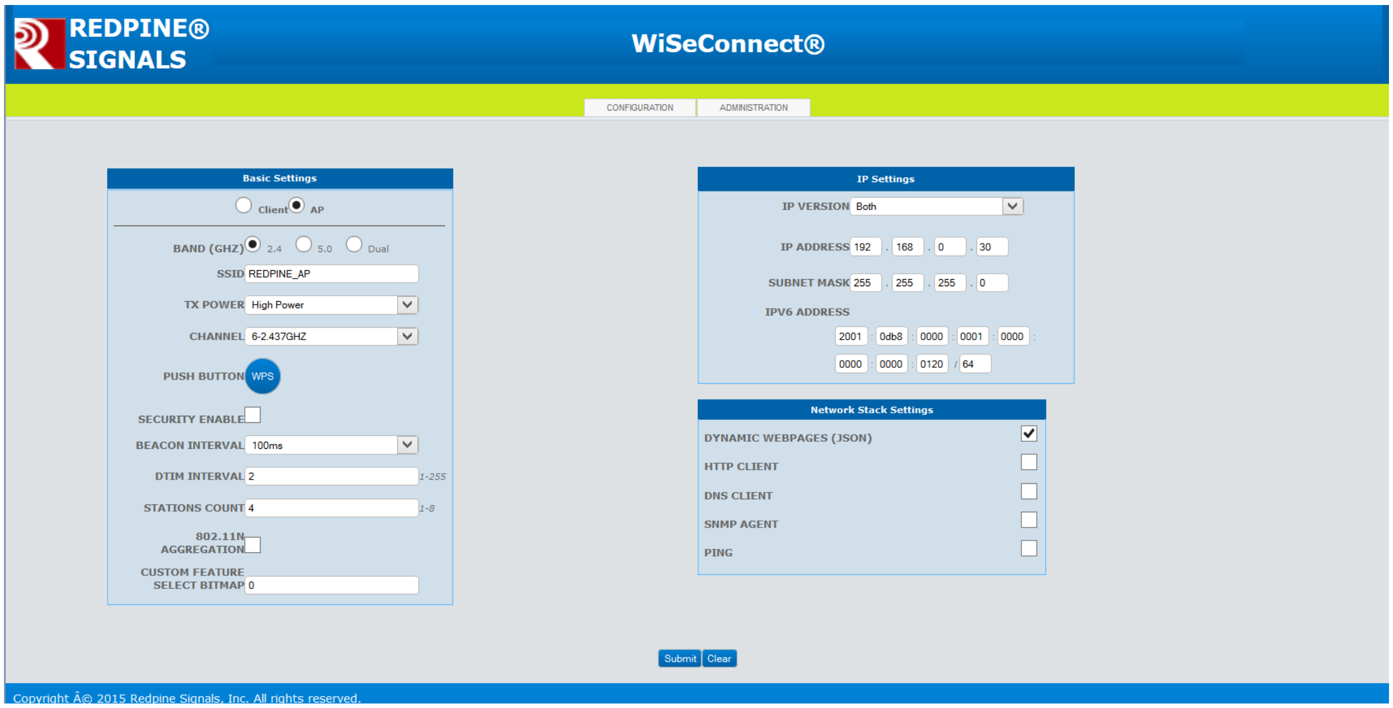
Figure 26: Module Configured as Access Point

1) Connect a PC or Host to the module through the USB and power up the module.
2) Configure the module to become an AP and DHCP Server by issuing commands from PC (P) as in section Evaluation of Access Point Mode. The sequence of commands is given below.

```
a. at+rsi_opermode=6,1,18,0
b. at+rsi_band=0
c. at+rsi_init
d. at+rsi_fwversion?
e. at+rsi_ipconf=0,192.168.0.30,255.255.255.0,192.168.0.
   30
f. at+rsi_apconf=6,REDPINE_AP,0,0,0,100,3,4
g. at+rsi_join=REDPINE_AP,0,2,0
```

The module is now configured as an Access Point. Its IP address is **192.168.0.30.**

3) Connect a Laptop (B) to the created AP. Open the URL **http://<Module's IP address>**in the Laptop. For example, if the module was configured to have an IP of 192.168.0.30, then the URL is http://192.168.0.30. Make sure the browser in the laptop does not have any proxies enabled.
4) The module's webpage opens as shown in the figure below.



Basic Settings

Client [✓ AP]

BAND (GHZ)	[✓ 2.4] 5.0 Dual
SSID	REDPINE_AP
TX POWER	High Power
CHANNEL	6-2.437GHZ
PUSH BUTTON	WPS
SECURITY ENABLE	☐
BEACON INTERVAL	100ms
DTIM INTERVAL	2 (1-255)
STATIONS COUNT	4 (1-8)
802.11N AGGREGATION	☐
CUSTOM FEATURE SELECT BITMAP	0

IP Settings

IP VERSION	Both
IP ADDRESS	192 . 168 . 0 . 30
SUBNET MASK	255 . 255 . 255 . 0
IPV6 ADDRESS	2001 : 0db8 : 0000 : 0001 : 0000 : 0000 : 0000 : 0120 / 64

Network Stack Settings

DYNAMIC WEBPAGES (JSON)	✓
HTTP CLIENT	☐
DNS CLIENT	☐
SNMP AGENT	☐
PING	☐

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

Figure 27: Module Webpage

5) In the page, select “AP” under Basic Settings.

6) Next, Select the Band and enter the new SSID of the Access Point.

7) Set the Tx Power, Channel, Security mode, Beacon Interval, DTIM Interval settings as per your requirements.

8) Enter the number of Stations you want to allow to connect to the Access Point. If this number is greater than 4, a bit needs to be set in the Custom Feature Bitmap, as per the details given in the Software PRM. In this example, we will select 4.

9) Click the radio button next to “802.11N AGGREGATION” to enable A-MPDU aggregation.

10) For details on the “CUSTOM FEATURE BITMAP”, refer to the Software PRM.

11) Under IP Settings, select between IPv4, IPv6 and Both and enter the IP address and Subnet Mask of the Access Point. In this example, we have selected IPv4. The DHCP Server is enabled by default in the Access Point mode.

12) Click on “Submit” button. The information is sent to the module and stored in its internal flash.

13) The module should now be power cycled or hard reset. It boots up and then automatically creates an AP with the configured parameters. The module will send out two responses to the Host, the first corresponds to the internally given “Set IP Parameters” command and the second to the “Join” command. Note that once the module is restarted, no commands need to be given. The module automatically and internally executes the commands to create an AP. The stored configuration parameters can be retrieved using the command “at+rsi_cfgget”. If the auto-connect

feature needs to be disabled, issue the command "at+rsi_cfgenable=0"to the module. Refer to the PRM for more details on these commands.

Flow 2: In this flow, the module is connected to an AP. A remote device connects to the same AP and configures the module.

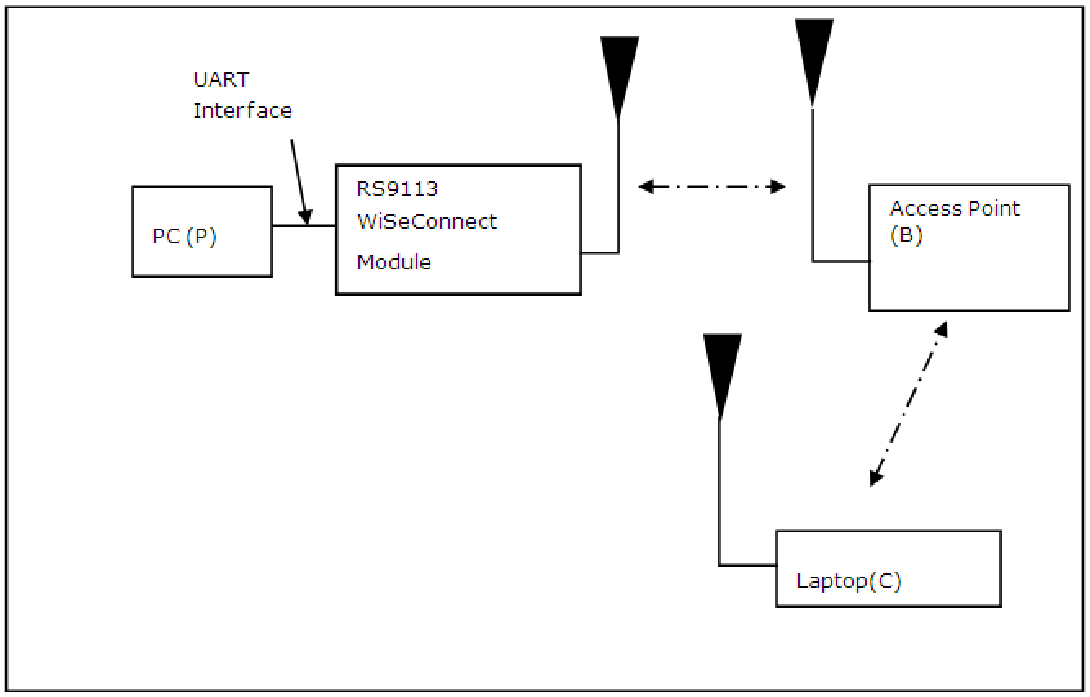


Figure 28: Module Connected to Access Point

1) Connect a PC or Host to the module through the USB and power up the module.
2) Configure the module to become client and connected to an AP by issuing commands through PC (P).
3) Connect a Laptop (c) to the created AP. Open the URL **http://<Module's IP address>**in the Laptop. For example, if the module was configured to have an IP of 192.168.100.1, then the URL is http://192.168.100.1. Make sure the browser in the laptop does not have any proxies enabled.
4) Follow the instructions in **Flow 1** from step 4 to reconfigure the module as an Access Point.

7 Appendix A: Headers on the EVB

7.1 Headers' Pin Orientations

The figure below shows the pin orientations for the SDIO, SPI and GPIO headers.

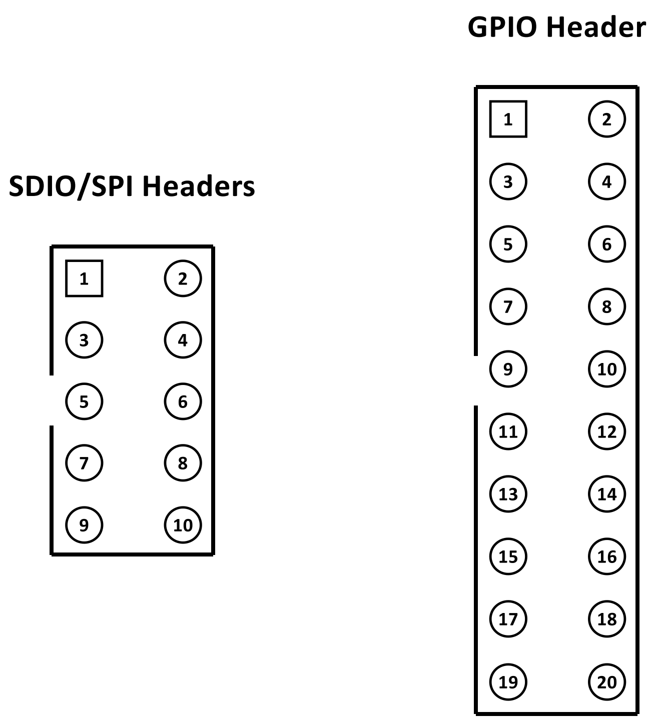


Figure 29: Headers' Pin Orientations

7.2 SPI Header Pin Description

The following table describes the pins of SPI header.

Pin Number	Pin Name	Direction	Description
1	NC	-	This pin must be left open.
2	SPI_CS	Input	SPI slave select from host(active low)
3	GND	-	Ground
4	VDD	-	Supply voltage.
5	SPI_CLK	Input	Serial clock in from the host. The clock can be up to80 MHz
6	GND	-	Ground
7	SPI_MOSI	Input	SPI data input
8	SPI_MISO	Output	SPI data output
9	SPI_INTR	Output	Active high, level triggered interrupt, used in SPI mode. The interrupt is raised by the EVB to indicate there is data to be read by the Host.
10	RESET_N	Input	Reset generated on the EVB or taken from the

			GPIO header – this is an input to the EVB.

Table 1: SPI Header Pins

7.3 SDIO Header Pin Description

The following table describes the pins of SDIO header.

Pin Number	Pin Name	Direction	Description
1	SDIO_DATA3	Input/Output	Data3 of SDIO interface.
2	SDIO_CMD	Input/Output	SDIO Mode: SDIO interface command signal.
3	GND	-	Ground
4	VDD	-	Supply voltage.
5	SDIO_CLK	Input	This signal is SDIO clock.
6	GND	-	Ground
7	SDIO_DATA0	Input/Output	Data0 of SDIO interface.
8	SDIO_DATA1	Input/Output	Data1 of SDIO interface.
9	SDIO_DATA2	Input/Output	Data2 of SDIO interface.
10	RESET_N	Output	Reset generated on the EVB or taken from the GPIO header – this is an output from the EVB.

Table 2: SDIO Header Pins

7.4 GPIO Header Pin Description

The following table describes the pins of SDIO header.

Pin Number	Pin Name	Direction	Description
1	GND	The GPIOs are generally not available for usage from the Host with the standard firmware. Please refer to the Software PRM for the pins that are supported or contact Silicon Labs in case you need to use them. GPIOs 9 to 12 are UART interface signals. The pins on the GPIO header corresponding to these signals can be used only to monitor these signals and not for actively driving them.	
2	GPIO_15		
3	GPIO_19		
4	GPIO_16		
5	GPIO_7		
6	GND		
7	GPIO_2		
8	GPIO_17		
9	GPIO_8		
10	GPIO_18		
11	GPIO_21		
12	GPIO_12		
13	GPIO_13		
14	GPIO_14		
15	GPIO_10		
16	GPIO_9		

17	NC	
18	GPIO_11	
19	3.3V Supply Output	
20	RESET_EXT	Active low reset input to the module.

Table 3: GPIO Header Pins

Note:

Signal Integrity Guidelines for SPI/SDIO interface: Glitches in the SPI/SDIO clock can potentially take the SPI/SDIO interface out of synchronization. The quality and integrity of the clock line needs to be maintained. In case a cable is used for board to board connection, the following steps are recommended (please note that this is not an exhaustive list of guidelines and depending on individual cases additional steps may be needed.):

a. Minimize the length of the SPI/SDIO bus cable to as small as possible, preferably to within an inch or two.

b. Increase the number of ground connections between the EVB and the Host processor PCB.

8 Appendix B: Example Applications

8.1 WLAN Example Applications

Example	Description	Application path
UDP server	This example demonstrates how to configure module in station mode and receive the data from the remote side using UDP socket	sapis/examples/wlan/udp_server/rsi_udp_server.c
UDP client	This example demonstrates how to configure module in station mode and send data to the remote side using UDP client socket	sapis/examples/wlan/udp_client/rsi_udp_client.c
TCP server	This example demonstrates how to configure module in station mode and receive the data from the remote side using TCP socket	sapis/examples/wlan/tcp_server/rsi_tcp_server.c
TCP client	This example demonstrates how to configure module in station mode and send data to the remote side using TCP client socket	sapis/examples/wlan/tcp_client/rsi_tcp_client.c
Enterprise mode connectivity	This example demonstrates how to connect the module to enterprise security enabled access point	sapis/examples/wlan/eap/rsi_eap_connectivity.c
ssl client	This example application demonstrates how to send data using TCP client socket over SSL in station mode.	sapis/examples/wlan/ssl_client/rsi_ssl_client.c
Access point creation	This example demonstrates how to Configure module in access point mode and receive the data from the remote side using TCP server socket.	sapis/examples/wlan/access_point/rsi_ap_start.c
ap udp echo	This example demonstrates how to Configure module in access point mode and echo the udp data sent by the remote side connected client device	sapis/examples/wlan/ap_udp_echo/rsi_ap_udp_echo.c
http client	This example application demonstrates how HTTP client is able to request a page and post the data to simple HTTP server	sapis/examples/wlan/http_client/rsi_http_client_app.c
smtp client	This example application demonstrates how SMTP client is able to send a mail to simple SMTP mail server	sapis/examples/wlan/smtp_client/rsi_smtp_client_app.c
ftp client	This example application demonstrates how file content is read from the ftp	sapis/examples/wlan/smtp_client/rsi_ftp_client.c

Example	Description	Application path
	server and copy this to a new file created	
Concurrent mode	This example application demonstrates how concurrent mode is used in allowing Device to act as Access point and Wifi station simultaneously	sapis/examples/wlan/concurrent_mode/ rsi_concurrent_mode.c
Connected sleep	This example application demonstrates how	sapis/examples/wlan/connected_sleep/rsi_wlan_connected_sleep_app.c
Connection using asynchronous apis app	This example application demonstrates how asynchronous apis are to to connect the device to access point	sapis/examples/wlan/connection_using_asynchronous_apis_app/ rsi_connection_using_asynchronous_apis_app.c
Firmware upgradation	This example application demonstrates how firmware is upgraded to the device remote TCP server socket	sapis/examples/wlan/fwup/rsi_fwup_app.c
multicast	This example application demonstrates how to data is receive from multicast group	sapis/examples/wlan/multicast/rsi_multicast_app.c
Power save	This example application demonstrates how power save feature is implemented in the device	sapis/examples/wlan/power_save/rsi_wlan_powersave_profile.c
provisioning	This example application demonstrates how provisioning is used to configure the device to join to an AP.	sapis/examples/wlan/provisioning/rsi_provisioning_app.c
Station ping	This example application demonstrates how ping to ping the remote peer	sapis/examples/wlan/station_ping/rsi_station_ping.c
Transmit test	This example application demonstrates how FCC certification test is run	sapis/examples/wlan/transmit_test/rsi_transmit_test_app.c
Websocket client	This example application demonstrates how to create a web socket client and send data	sapis/examples/wlan/websocket_client/rsi_websocket_client_app.c
Wifi direct	This example application demonstrates how to create a wifi direct client and send data	sapis/examples/wlan/wifi_direct/rsi_wfd_client.c
Wps station	This example application demonstrates how to connect to a WPS supported AP using push button method	sapis/examples/wlan/wps_station/rsi_wps_station.c

8.2 BLE Example Applications

Example	Description	Application path
simple_peripheral	This example demonstrates simple BLE peripheral mode	sapis/examples/ble/ simple_peripheral/rsi_ble_peripheral.c
simple_central	This example demonstrates	sapis/examples/ble/simple_central/rsi

Example	Description	Application path
	simple BLE central mode	_ble_central.c
simple_chat	This example demonstrates simple data exchange (loopback) b/w module and peer device.	sapis/examples/ble/simple_chat/rsi_ble_simple_chat.c
immediate_alert_client	This example demonstrates GATT client role for immediate alert service.	sapis/examples/ble/immediate_alert_client/rsi_ble_immediate_alert_client.c

8.3 ZigBee Example Applications

Example	Description	Application path
switch	This example demonstrates how to send on/off/toggle command to a Light coordinator.	sapis/examples/ZigBee/switch/rsi_zb_app.c

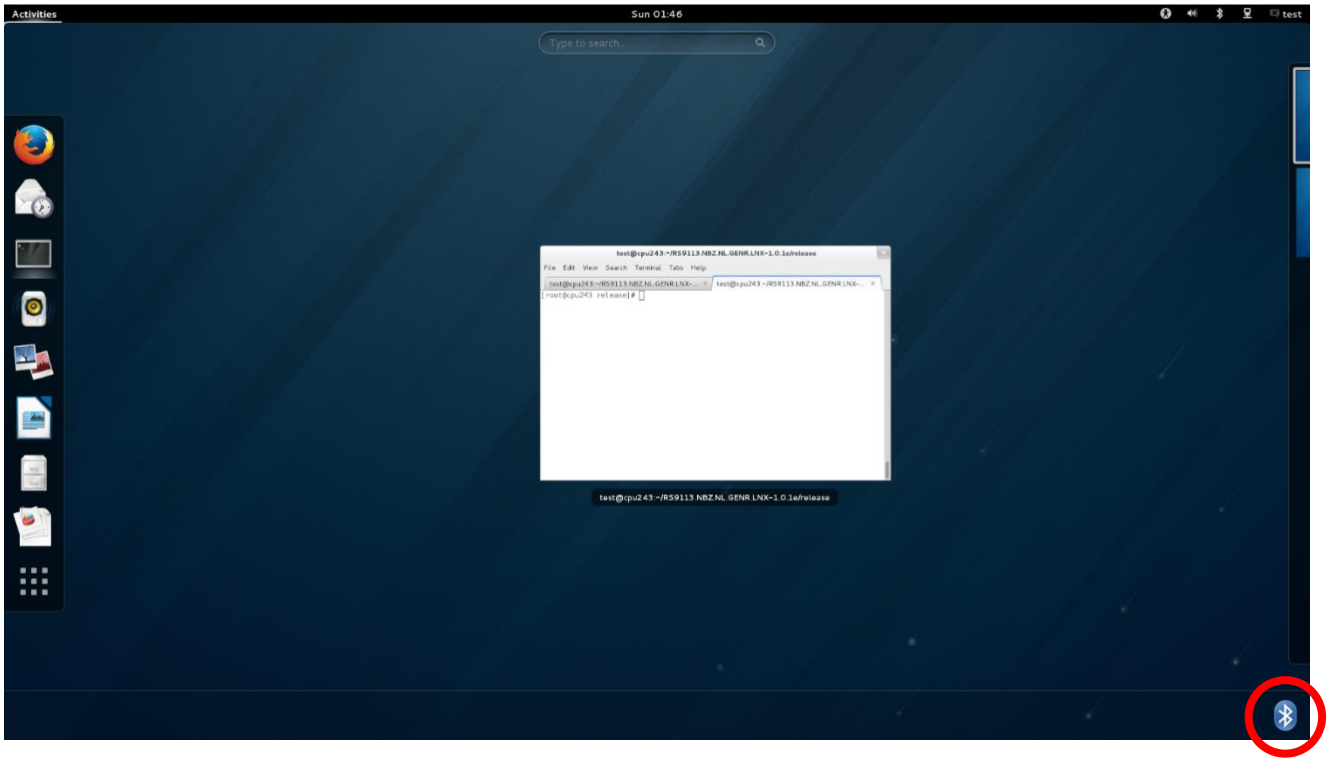
8.4 Coexistence Example Applications

Example	Description	Application path
Controlling switch using TCP IP socket on remote side	This example demonstrates how to send on/off/toggle command to a Light coordinator using TCP client socket. On/off commands are triggered by the TCP server.	sapis/examples/wlan_ZigBee/wlan_ZigBee_switch/
Simple chat using BLE and WLAN station applications	This example demonstrates the data exchanges between BLE and WLAN applications.	sapis/examples/wlan_ble/wlan_station_ble_bridge/
Simple chat using BLE and WLAN access point applications	This example demonstrates the data exchanges between BLE and WLAN applications.	sapis/examples/wlan_ble/wlan_ap_ble_bridge/
Simple chat using BLE and WLAN access point applications in tcpip bypass mode	This example demonstrates the data exchanges between BLE and WLAN applications	sapis/examples/wlan_ble/wlan_ap_ble_bridge_tcpipbypass/
Simple chat using BT and WLAN station applications	This example demonstrates the data exchanges between BT and WLAN applications	sapis/examples/wlan_bt/wlan_bt_bridge/
Simple chat using BT and WLAN station applications in tcpip bypass mode	This example demonstrates the data exchanges between BT and WLAN applications	sapis/examples/wlan_bt/wlan_bt_bridge_tcpipbypass/
Simple chat using BT and WLAN access point applications in	This example demonstrates the data exchanges between BT and WLAN applications	sapis/examples/wlan_bt/wlan_ap_bt_bridge_tcpipbypass/

Example	Description	Application path
tcpip bypass mode		
Coex power save and Simple chat using BT and WLAN station applications in tcpip bypass mode	This example demonstrates the data exchanges between BT and WLAN applications with power save	sapis/examples/wlan_bt/power_save/

9 Appendix C: Using the Bluetooth Manager

1) Once the Bluetooth modules have been installed using the bt_enable.sh script as per the instructions in Section 4.4, hit the "Windows" button on the keyboard. You will see Bluetooth symbol at the bottom-right corner of the screen, as show in the image below.


Activities
Sun 01:46
test
Type to search...
File Edit View Search Terminal Tabs Help
test@cpu243:~/RS9113.NBZ.NL.GENR.LNX-...
test@cpu243:~/RS9113.NBZ.NL.GENR.LNX-...
test@cpu243:~/RS9113.NBZ.NL.GENR.LNX-1.0.1e/release

```
test@cpu243:~/RS9113.NBZ.NL.GENR.LNX-1.0.1e/release
[root@cpu243 release]# 
```

Figure 30: Invoking Bluetooth Manager

2) This will open the Bluetooth Manager as shown in the image below.

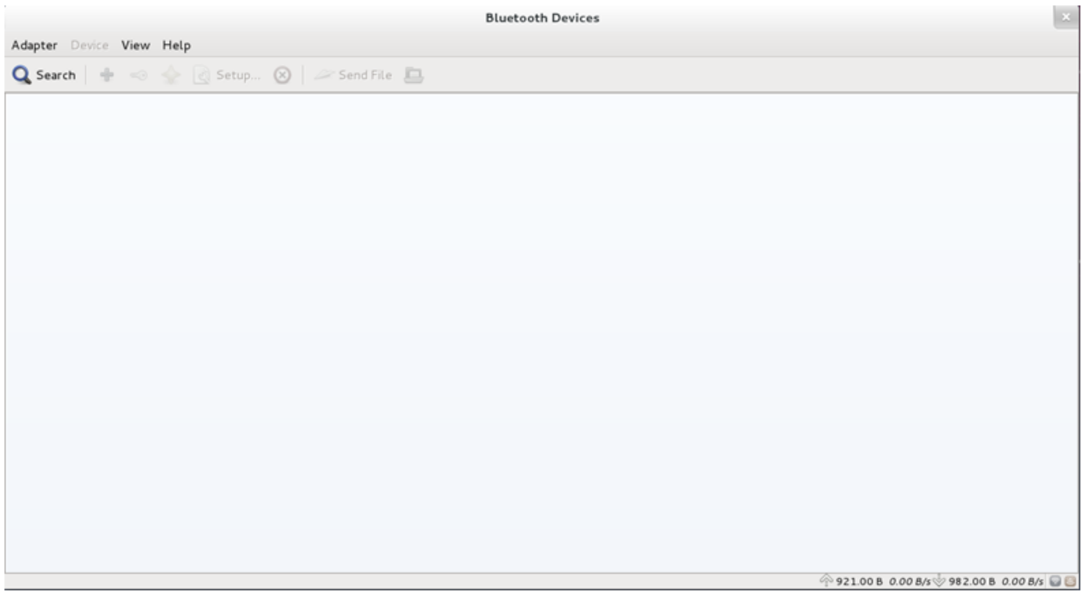
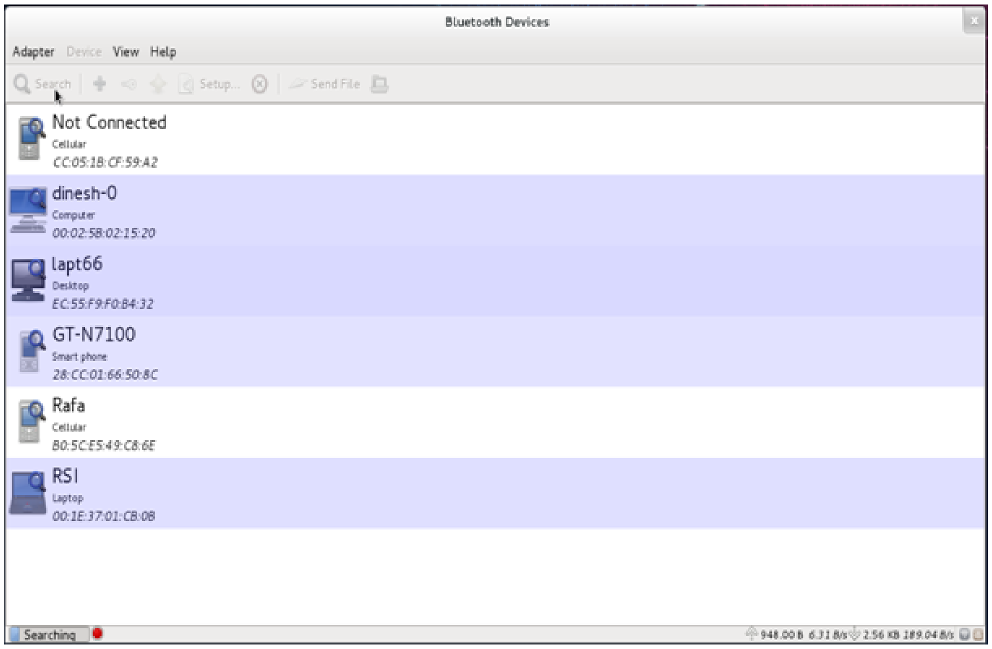


Figure 31: Bluetooth Manager basic window

3) Click on Search to start inquiry.



Not Connected
Cellular
CC:05:1B:CF:59:A2

dinesh-0
Computer
00:02:58:02:15:20

lapt66
Desktop
EC:55:F9:F0:B4:32

GT-N7100
Smart phone
28:CC:01:66:50:8C

Rafa
Cellular
B0:5C:E5:49:C8:6E

RSI
Laptop
00:1E:37:01:CB:0B

Figure 32: Click on Search to inquire

4) Select particular device, like your smartphone, right click and select Pair to pair with that device.

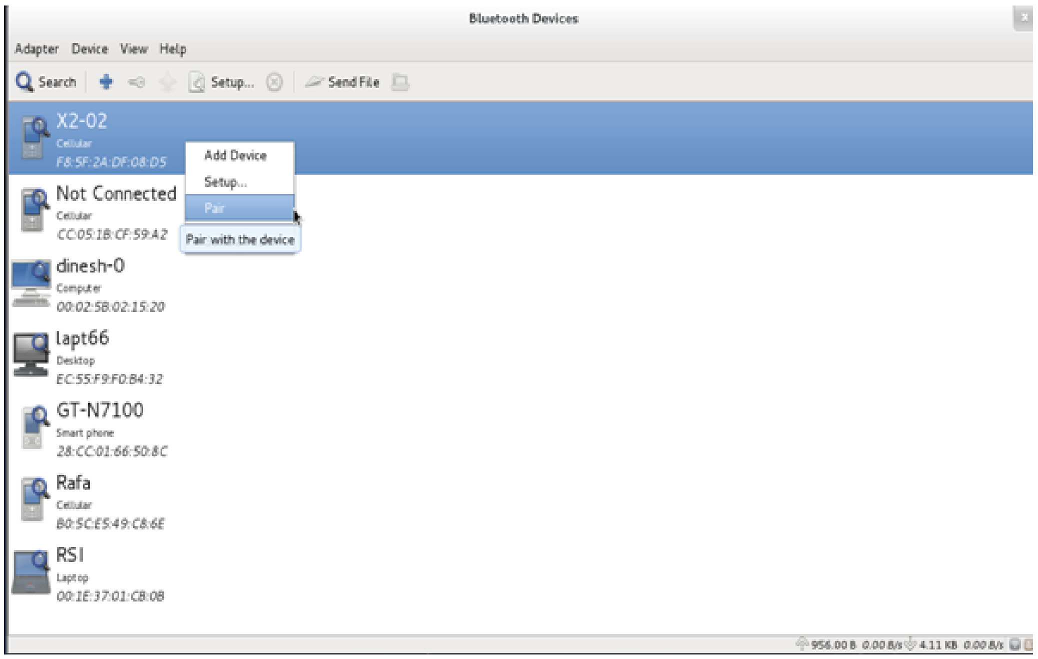


Figure 33: Pairing with a Device

5) After successfully pairing with the device, right-click on the device and select “Send a file” to send data to the device. You will be presented with a dialog box to select the file that you wish to send.

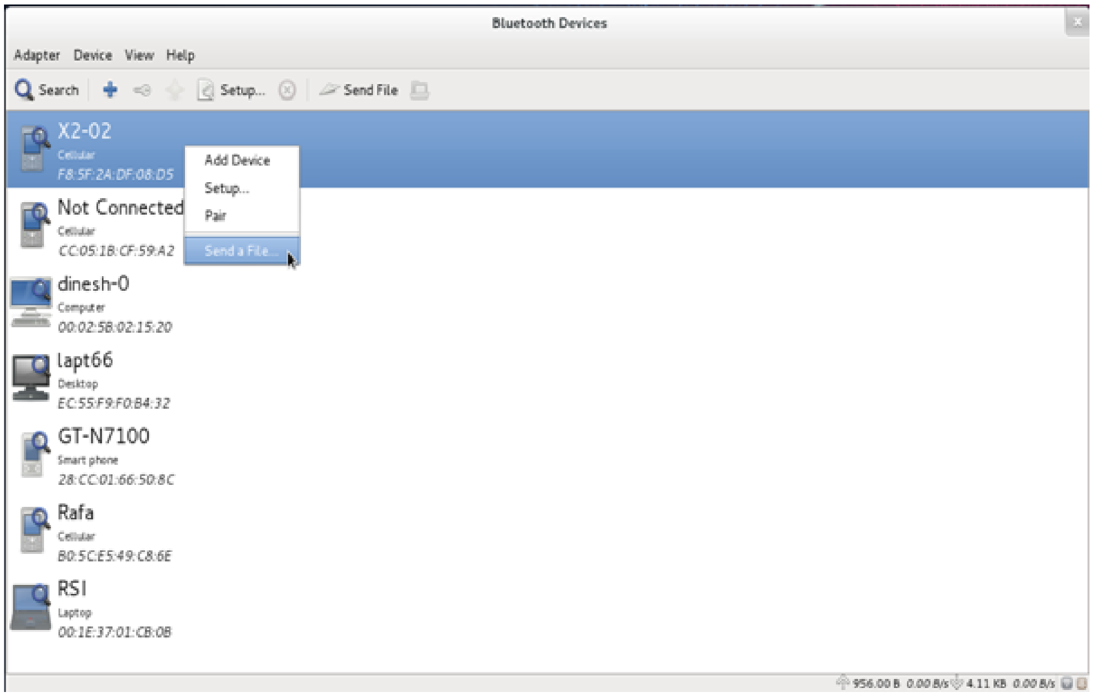


Figure 34: Send a File to a Device

10 Appendix D: WiSeConnect® Firmware Upgrade

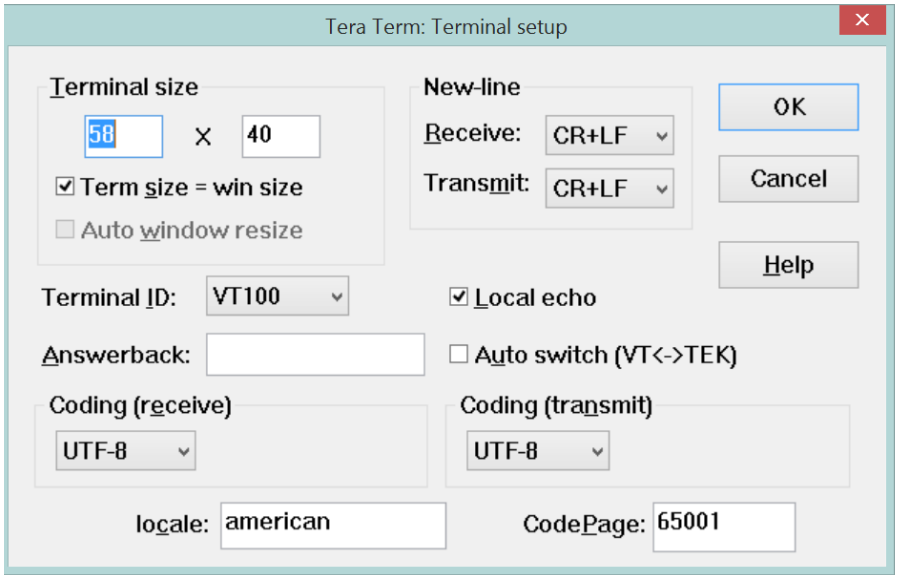
The WiSeConnect® firmware of the module can be upgraded either from the Host or over the Wi-Fi connection. This section describes both these options, with UART being the Host interface. For details on upgrading over other interfaces (SPI, USB and USB-CDC), refer to the Software PRM.

NOTE:

1) The firmware upgrade for n-Link mode is done automatically by the driver that is provided as part of the release package. This is done by comparing the version of the firmware present with the driver and the one which is loaded.
2) The EVK is shipped with the WiSeConnect® firmware loaded. So a firmware upgrade is not necessary unless there is a newer version available or the EVB has been used in n-Link® mode and the user wants to evaluate the EVB in WiSeConnect® mode after that.
3) The process explained below is for upgrading the WiSeConnect® firmware over UART from a Windows PC using Teraterm. The Software PRM gives details related to the process and commands to be used to upgrade over other interfaces and MCU platforms.

10.1 Firmware Upgrade over UART

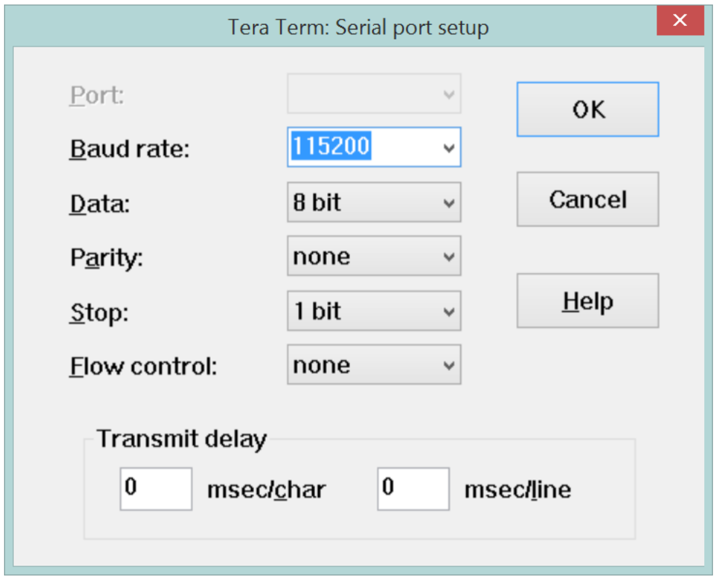
1) Download and install Teraterm from http://en.sourceforge.jp/projects/ttssh2/releases/
2) Open Teraterm. You will be asked to setup a "New connection". Click Cancel.
3) In the Teraterm window, click on Setup -> Terminal…In the dialog box that opens, change the following settings:
   a. Under "New-line", select "CR+LF" for Receive and Transmit
   b. Enable "Local echo".
   c. Click OK



Terminal size: 58 x 40
☑ Term size = win size
☐ Auto window resize
New-line
Receive: CR+LF
Transmit: CR+LF
Terminal ID: VT100
☑ Local echo
Answerback:
☐ Auto switch (VT<->TEK)
Coding (receive): UTF-8
Coding (transmit): UTF-8
locale: american
CodePage: 65001

Figure 35: Terminal Settings for Teraterm

4) Next, click on Setup -> Serial port… In the dialog box that opens, select the Baud Rate as 115200 and Click OK.


Tera Term: Serial port setup

Port:
Baud rate: 115200
Data: 8 bit
Parity: none
Stop: 1 bit
Flow control: none

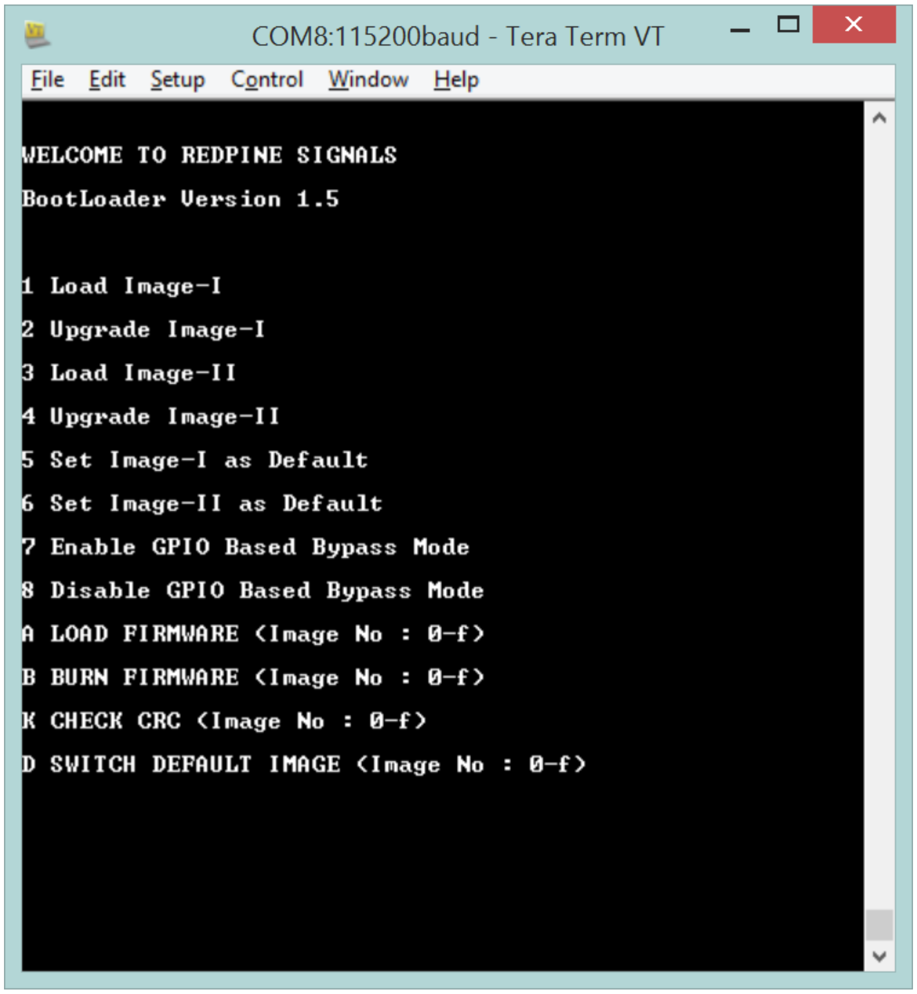
Transmit delay
0 msec/char 0 msec/line

Figure 36: Serial Port Settings for Teraterm

5) Connect the EVB to the PC using the Micro A/B-type USB cable. Plug in the cable into the micro-USB port labeled “UART” on the EVB.

6) Click on File -> New connection… In the dialog box that opens, select the Serial option and select the COM Port from the drop-down menu. It is COM8 in the figure below. Click OK.

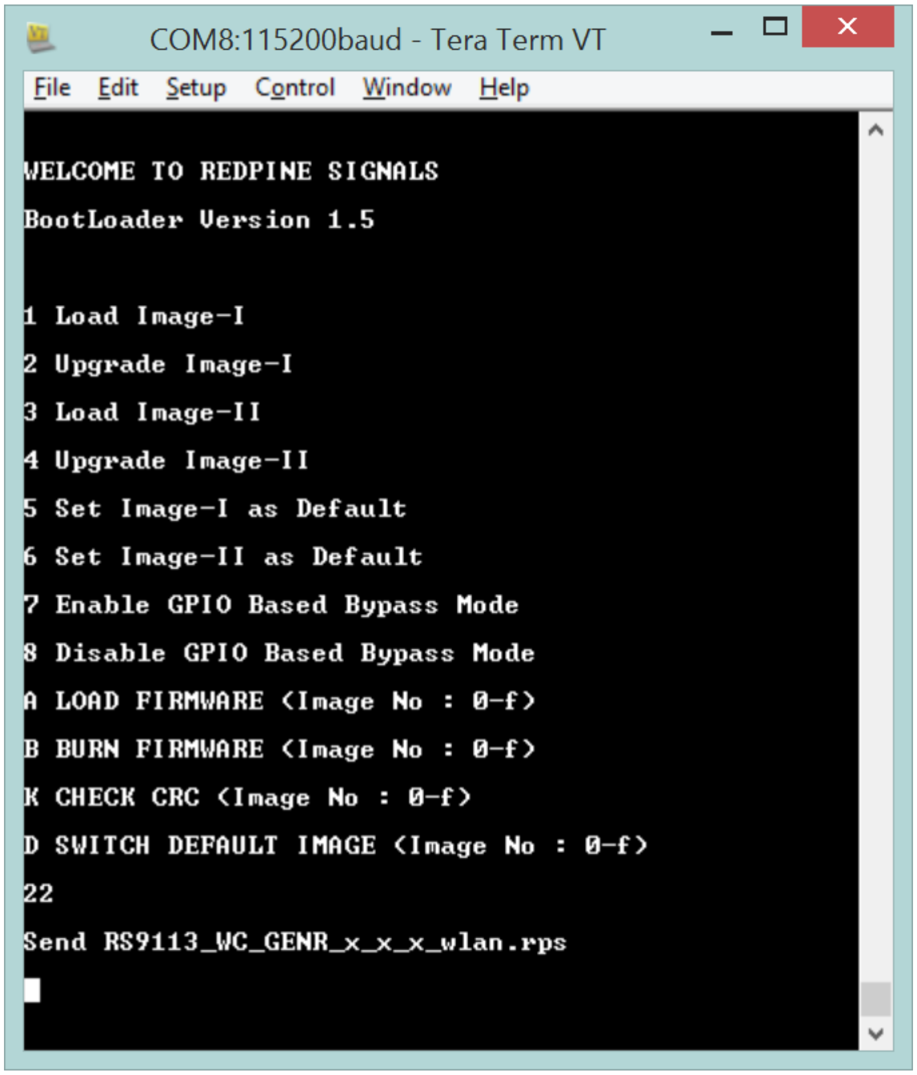
7) After the Automatic Baud Rate Detection timeout, you will see the Welcome message and available options on the Teraterm screen.


COM8:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
WELCOME TO REDPINE SIGNALS
BootLoader Version 1.5

1 Load Image-I
2 Upgrade Image-I
3 Load Image-II
4 Upgrade Image-II
5 Set Image-I as Default
6 Set Image-II as Default
7 Enable GPIO Based Bypass Mode
8 Disable GPIO Based Bypass Mode
A LOAD FIRMWARE (Image No : 0-f)
B BURN FIRMWARE (Image No : 0-f)
K CHECK CRC (Image No : 0-f)
D SWITCH DEFAULT IMAGE (Image No : 0-f)
```

Figure 37: Module Startup Messages

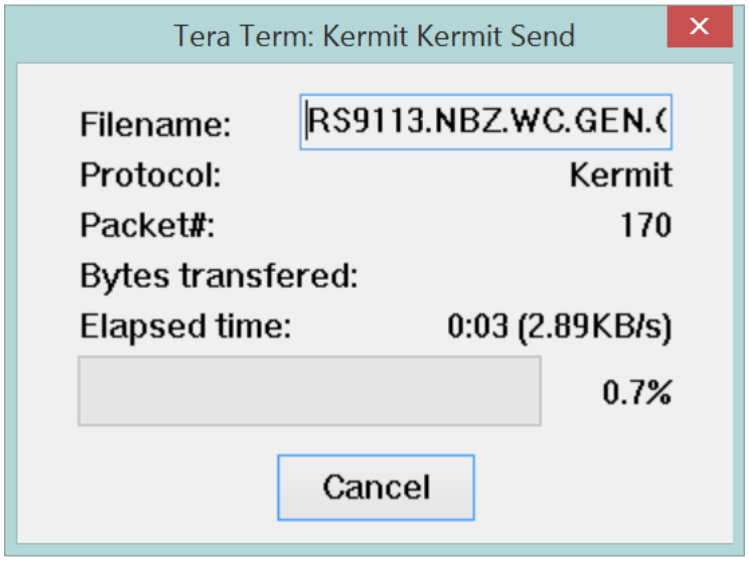
8) Hit ‘2’ to select Firmware Upgradation mode. You will be requested to send the Firmware file. For the EVK, the firmware file is present in the USB drive in the RS9113.NBZ.WC.GEN.OSI.x.x.x\Firmware folder. Here, x.x.x is the release version.


COM8:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
WELCOME TO REDPINE SIGNALS
BootLoader Version 1.5

1 Load Image-I
2 Upgrade Image-I
3 Load Image-II
4 Upgrade Image-II
5 Set Image-I as Default
6 Set Image-II as Default
7 Enable GPIO Based Bypass Mode
8 Disable GPIO Based Bypass Mode
A LOAD FIRMWARE (Image No : 0-f)
B BURN FIRMWARE (Image No : 0-f)
K CHECK CRC (Image No : 0-f)
D SWITCH DEFAULT IMAGE (Image No : 0-f)
22
Send RS9113_WC_GENR_x_x_x_wlan.rps
```

Figure 38: Request for Firmware File

9) Click on File -> Transfer -> Kermit -> Send…
10) In the dialog box that opens, navigate to the RS9113.NBZ.WC.GEN.OSI.x.x.x\Firmware folder and select the RS9113.NBZ.WC.GEN.OSI.x.x.x.rps file.
11) A dialog box will open, showing the transfer of the firmware file.


Tera Term: Kermit Kermit Send

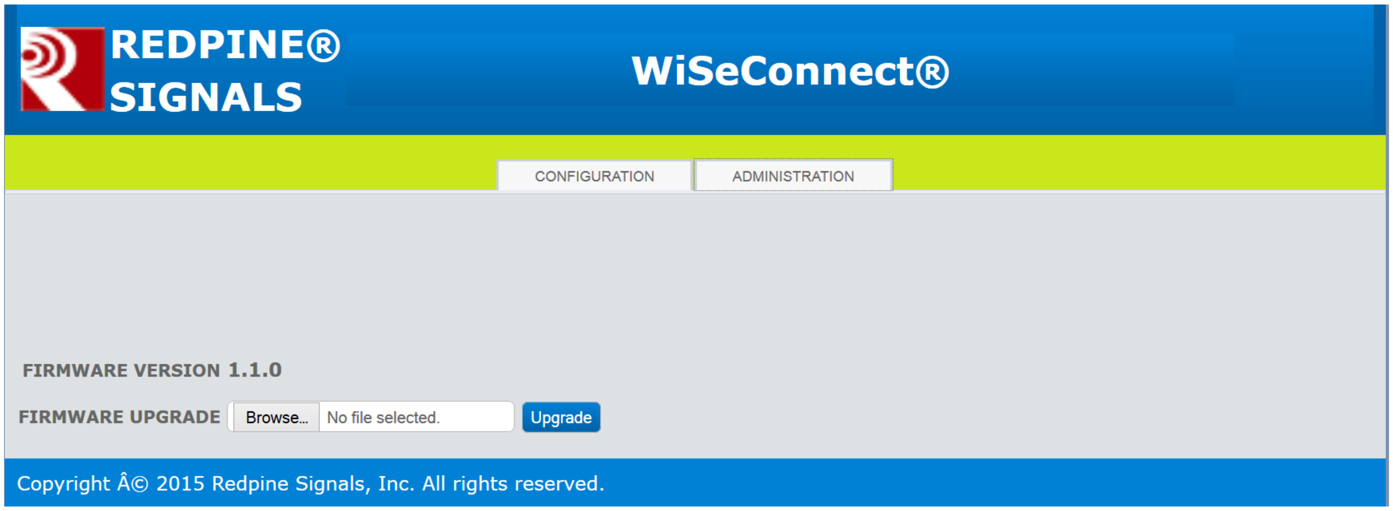
Filename: RS9113.NBZ.WC.GEN.(
Protocol: Kermit
Packet#: 170
Bytes transfered:
Elapsed time: 0:03 (2.89KB/s)
0.7%

Figure 39: Firmware Transfer Progress

12) Once the firmware is transmitted to the module successfully, the module boots up with the updated firmware and requests the user to select the option.
13) You may continue to use the EVB without disconnecting it. If you wish to switch to Docklight from Teraterm, you can click on File -> Disconnect in Teraterm, open Docklight and hit ‘F5’ to start the communication from Docklight.

10.2 Firmware Upgrade over Wi-Fi

1) Module configured as an Access Point: Connect a Laptop which has the new firmware file (.rps extension) to the module.
2) Module configured as a Client and connected to an Access Point: Connect a Laptop which has the new firmware file (.rps extension) to the Access Point.
3) Ensure that the Laptop has an IP assigned.
4) Open the URL **http://<Module’s IP address>**in the Laptop. For example, if the module was configured to have an IP of 192.168.0.30, then the URL is http://192.168.0.30. Make sure the browser in the laptop does not have any proxies enabled.
5) In the webpage that opens, click on the “ADMINISTRATION” tab.


REDPINE® SIGNALS

WiSeConnect®

FIRMWARE VERSION 1.1.0

FIRMWARE UPGRADE Browse... No file selected. Upgrade

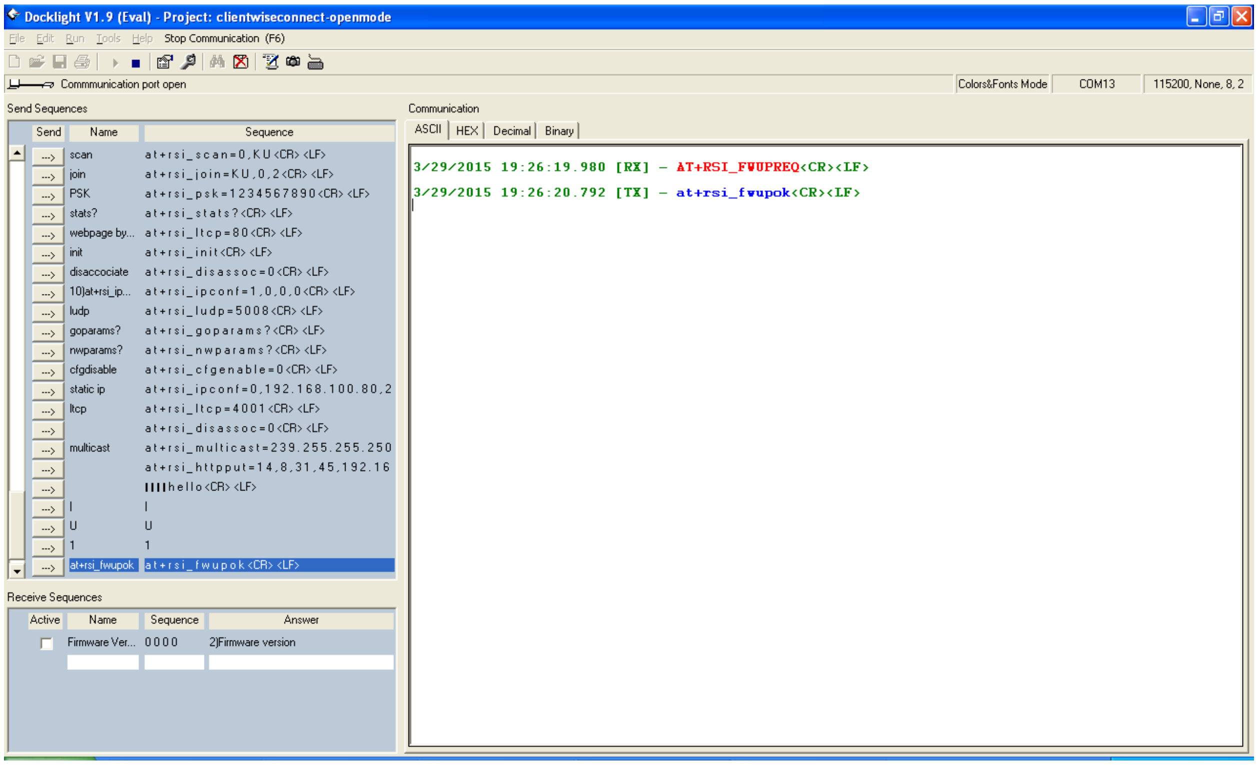
Copyright Â© 2015 Redpine Signals, Inc. All rights reserved.

Figure 40: ADMINISTRATION Tab

6) Click on “Browse” to navigate to the location of the firmware file (.rps extension), select the file and click “Open".

7) Next, click on “Upgrade”.

8) The module sends a response (AT+RSI_FWUPREQ) to the Host PC to confirm the Firmware Upgrade, as shown in the figure below. The Host has to send the confirmation command (AT+RSI_FWUPOK) to start upgrading the firmware.


Docklight V1.9 (Eval) - Project: clientwiseconnect-openmode

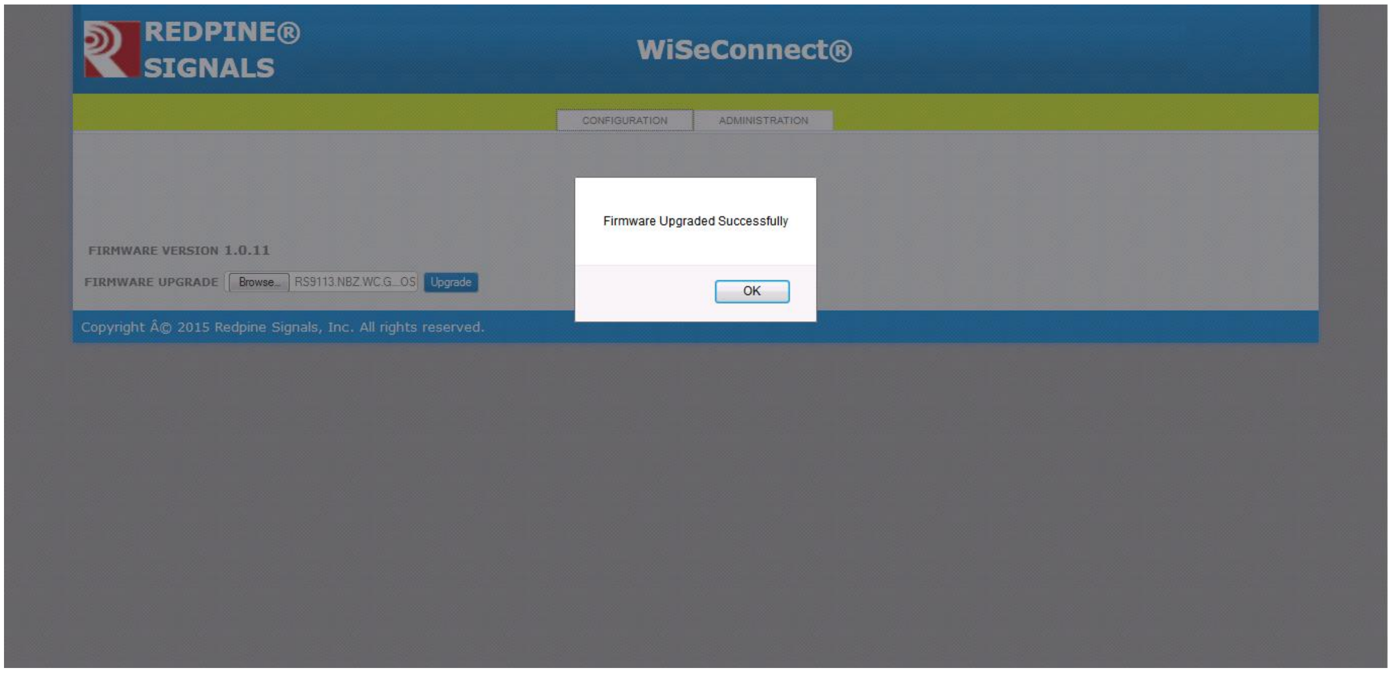
Send	Name	Sequence
--->	scan	at+rsi_scan=0,KU<CR><LF>
--->	join	at+rsi_join=KU,0,2<CR><LF>
--->	PSK	at+rsi_psk=1234567890<CR><LF>
--->	stats?	at+rsi_stats?<CR><LF>
--->	webpage by...	at+rsi_ltcp=80<CR><LF>
--->	init	at+rsi_init<CR><LF>
--->	disaccociate	at+rsi_disassoc=0<CR><LF>
--->	10)at+rsi_ip...	at+rsi_ipconf=1,0,0,0<CR><LF>
--->	ludp	at+rsi_ludp=5008<CR><LF>
--->	goparams?	at+rsi_goparams?<CR><LF>
--->	nwparams?	at+rsi_nwparams?<CR><LF>
--->	cfgdisable	at+rsi_cfgenable=0<CR><LF>
--->	static ip	at+rsi_ipconf=0,192.168.100.80,2
--->	ltcp	at+rsi_ltcp=4001<CR><LF>
--->		at+rsi_disassoc=0<CR><LF>
--->	multicast	at+rsi_multicast=239.255.255.250
--->		at+rsi_httpput=14,8,31,45,192.16
--->		IIIIhello<CR><LF>
--->	I	I
--->	U	U
--->	1	1
--->	at+rsi_fwupok	at+rsi_fwupok<CR><LF>

Active	Name	Sequence	Answer
☐	Firmware Ver...	0000	2)Firmware version

```
3/29/2015 19:26:19.980 [RX] - AT+RSI_FWUPREQ<CR><LF>
3/29/2015 19:26:20.792 [TX] - at+rsi_fwupok<CR><LF>
```

Figure 41: Firmware Upgrade Confirmation

9) Once the uprgadation is done, a “Firmware Upgraded Successfully” message is displayed, as shown in the figure below.



Firmware Upgraded Successfully

Figure 42: Firmware Upgraded Successfully

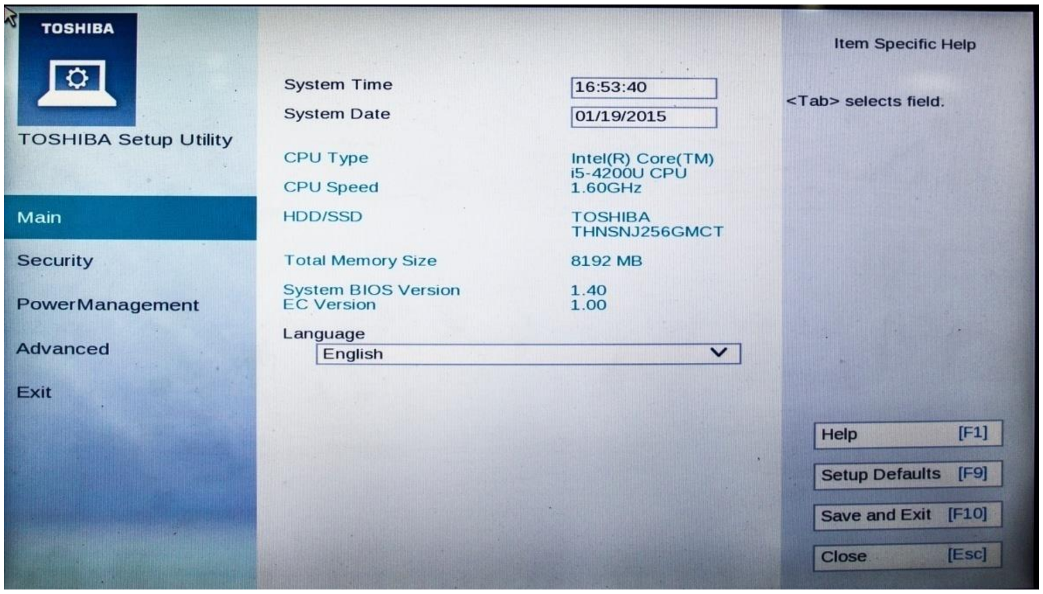
11 Appendix E: Booting USB Drive on Windows 8 and Later PCs

Starting with Windows 8, the BIOS uses thenew UEFI system. This can cause failures in booting from the USB drive provided as part of the EVK. To get around this problem, please follow the steps below. The steps in this section are explained for a Toshiba Kirabook. The process will be similar for other Windows 8 PCs.

NOTE: It is recommended that the evaluation be done on PCs which have the Legacy BIOS system installed. Please continue with the steps below only if you are aware of what is being done because incorrect settings might corrupt the Laptop's settings and render it useless.

Once these changes are done, the original Windows 8 OS will not be bootable until the changes are reverted.

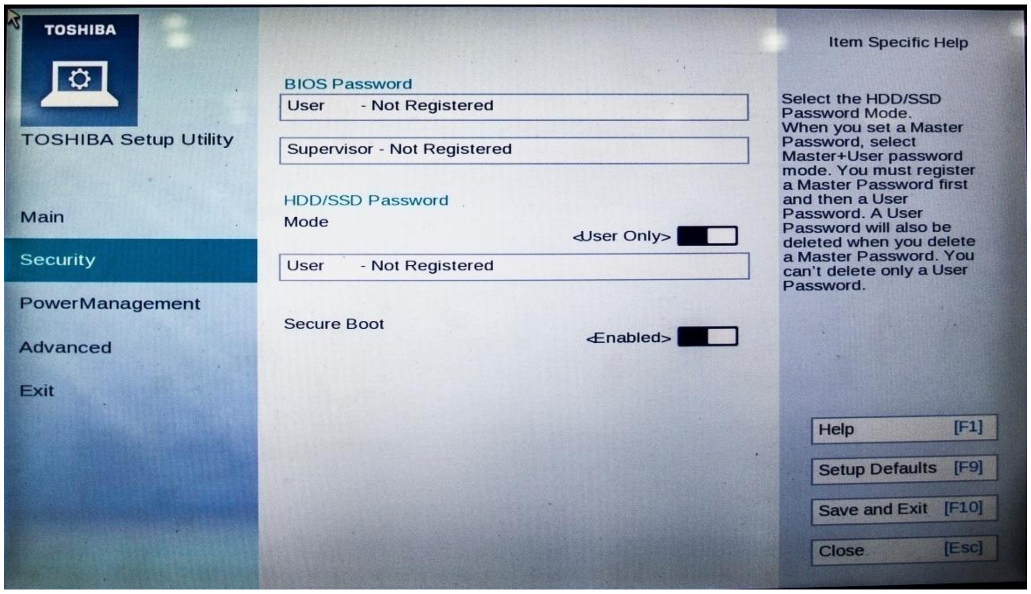
1) While booting up the PC, press F2 to enter the BIOS setup screen.



System Time: 16:53:40
System Date: 01/19/2015
CPU Type: Intel(R) Core(TM) i5-4200U CPU
CPU Speed: 1.60GHz
HDD/SSD: TOSHIBA THNSNJ256GMCT
Total Memory Size: 8192 MB
System BIOS Version: 1.40
EC Version: 1.00
Language: English

Figure 43: BIOS Setup Screen

2) Scroll down to the "Security" section on the left part of the screen. You will be shown an option called "Secure Boot".



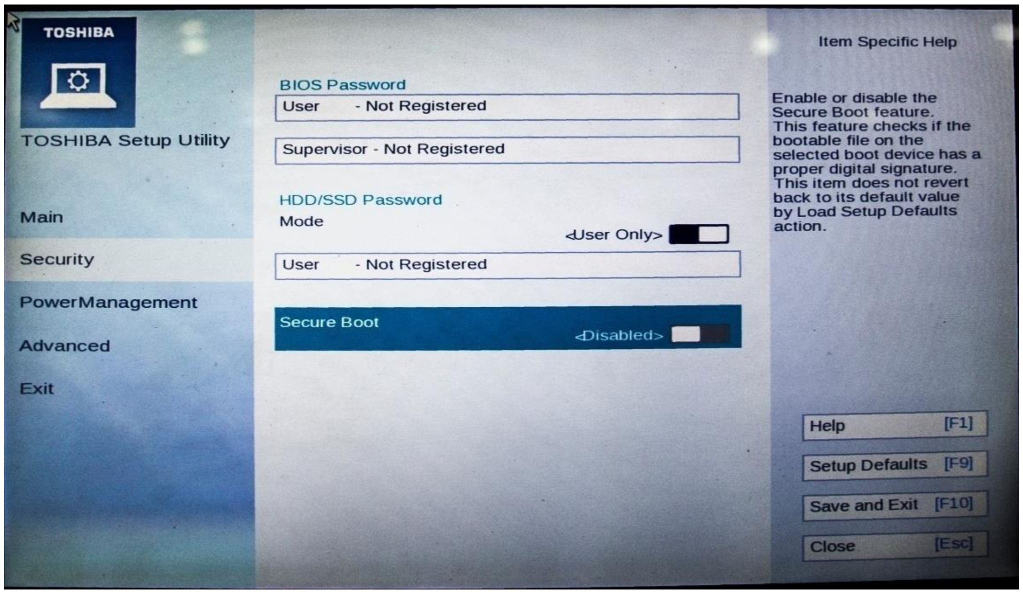
BIOS Password
User - Not Registered
Supervisor - Not Registered
HDD/SSD Password
Mode <User Only>
User - Not Registered
Secure Boot <Enabled>

Item Specific Help

Select the HDD/SSD Password Mode. When you set a Master Password, select Master+User password mode. You must register a Master Password first and then a User Password. A User Password will also be deleted when you delete a Master Password. You can't delete only a User Password.

Figure 44: Security Screen of BIOS

3) Disable the “Secure Boot” option.



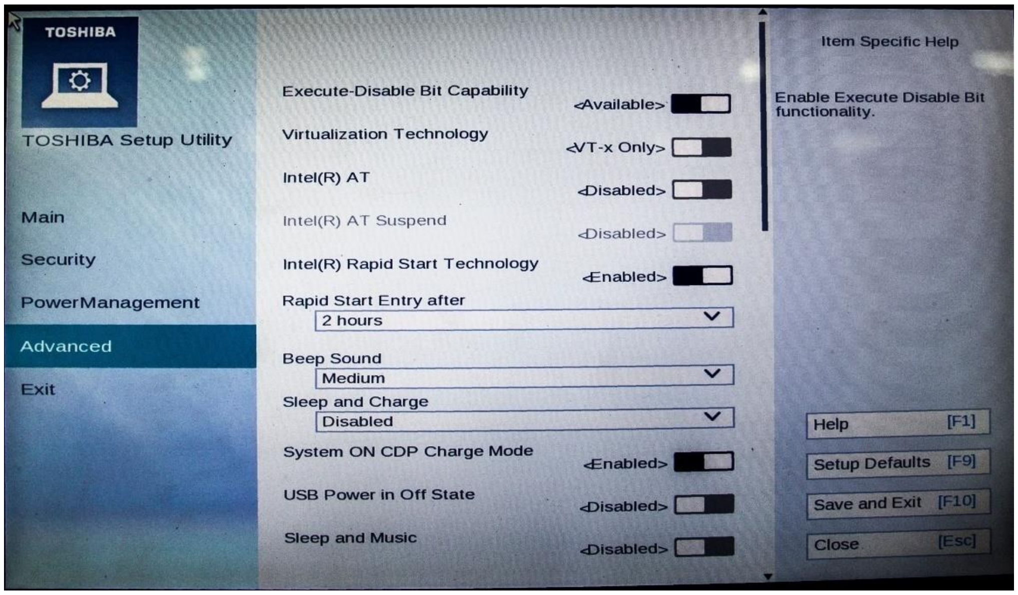
BIOS Password
User - Not Registered
Supervisor - Not Registered
HDD/SSD Password
Mode <User Only>
User - Not Registered
Secure Boot <Disabled>

Item Specific Help

Enable or disable the Secure Boot feature. This feature checks if the bootable file on the selected boot device has a proper digital signature. This item does not revert back to its default value by Load Setup Defaults action.

Figure 45: Disable “Secure Boot”

4) Next, scroll down to the “Advanced” section on the left part of the screen.

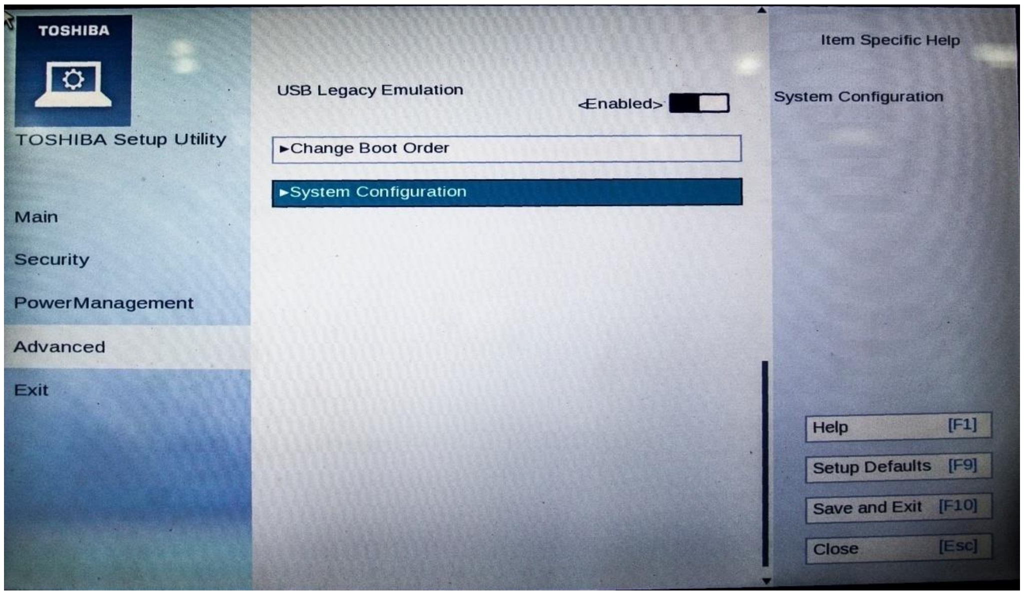


Execute-Disable Bit Capability <Available>
Virtualization Technology <VT-x Only>
Intel(R) AT <Disabled>
Intel(R) AT Suspend <Disabled>
Intel(R) Rapid Start Technology <Enabled>
Rapid Start Entry after: 2 hours
Beep Sound: Medium
Sleep and Charge: Disabled
System ON CDP Charge Mode <Enabled>
USB Power in Off State <Disabled>
Sleep and Music <Disabled>

Enable Execute Disable Bit functionality.

Figure 46: Advanced Screen of BIOS

5) Scroll down to the “System Configuration” option under “Advanced” and hit Enter.

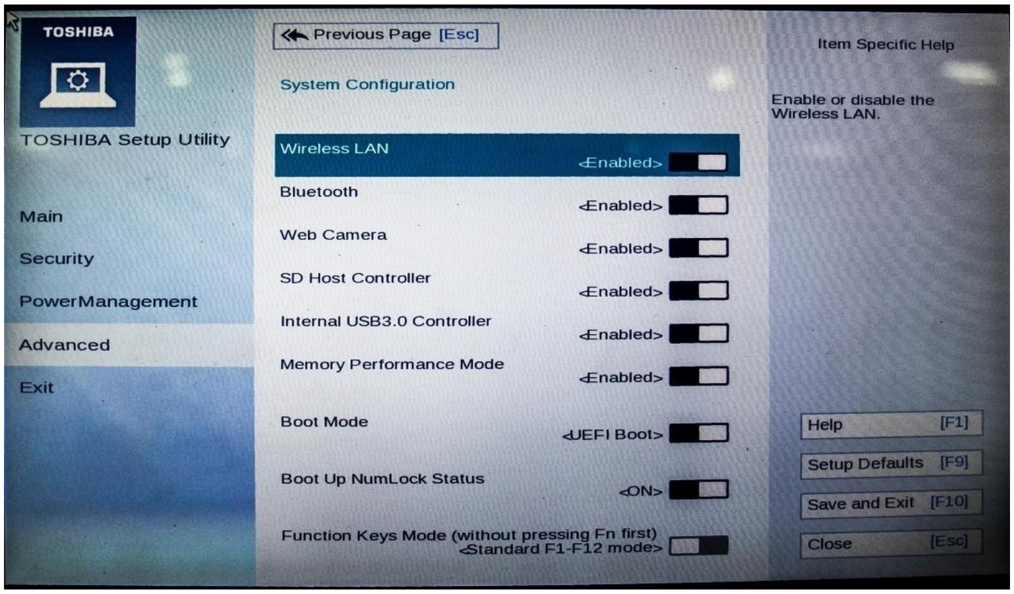


USB Legacy Emulation <Enabled>
►Change Boot Order
►System Configuration

System Configuration

Figure 47: System Configuration under Advanced

6) In the new screen, you will see an option labeled, “Boot Mode” and it will be in “UEFI Boot” mode.

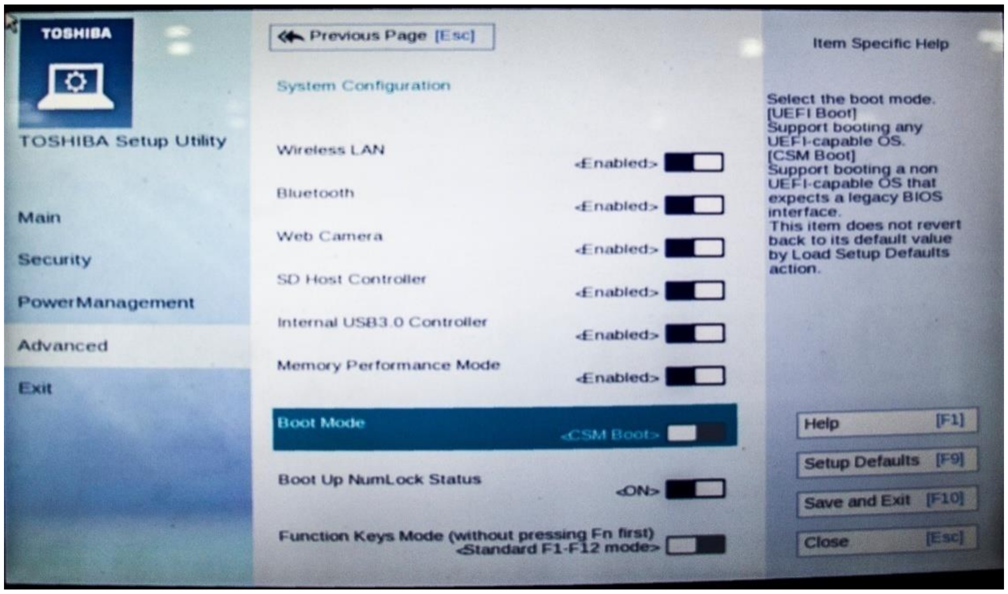


Wireless LAN <Enabled>
Bluetooth <Enabled>
Web Camera <Enabled>
SD Host Controller <Enabled>
Internal USB3.0 Controller <Enabled>
Memory Performance Mode <Enabled>
Boot Mode <UEFI Boot>
Boot Up NumLock Status <ON>
Function Keys Mode (without pressing Fn first) <Standard F1-F12 mode>

Enable or disable the Wireless LAN.

Figure 48: System Configuration

7) Select the "Boot Mode" option and hit Enter to disable UEFI Boot and enable CSM Boot – this is the legacy booting option of BIOS.



Wireless LAN <Enabled>
Bluetooth <Enabled>
Web Camera <Enabled>
SD Host Controller <Enabled>
Internal USB3.0 Controller <Enabled>
Memory Performance Mode <Enabled>
Boot Mode <CSM Boot>
Boot Up NumLock Status <ON>
Function Keys Mode (without pressing Fn first) <Standard F1-F12 mode>

Select the boot mode.
[UEFI Boot]
Support booting any UEFI-capable OS.
[CSM Boot]
Support booting a non UEFI-capable OS that expects a legacy BIOS interface.
This item does not revert back to its default value by Load Setup Defaults action.

Figure 49: Disable UEFI Boot

8) Select the Exit option and save the changes before exiting.
9) Once these changes are done, the original Windows 8 OS will not be bootable until the changes are reverted. To revert changes, enable UEFI Boot first and then enable Secure Boot (in that order).

11.1 Steps to update Live USB Image

Please use the steps mentioned below to flash the new Live USB image:

11.1.1 On Windows PC:

1. Copy the generated .iso image to pen drive. From the USB pen drive, copy it to a windows PC.
2. Install liveusb-creator-3.11.1-setup.exe (Search for fedora live usb creator for windows in google)
3. After installation run liveusb-creator in windows. Browse the .iso image location and set target as pen drive location, then click on "Create Live USB" button.
4. This will take few minutes to flash, and then pendrive will be ready for testing.

11.1.2 On Linux PC:

1. Install liveusb-creator by using the command "yum insall liveusb-creator"
2. After installation move to the folder where your iso image is present and launch liveusb-creator to flash it to pendrive.
3. Click on 'Browse' to select iso image and then click on 'Create Live USB' button.
4. This will take few minutes to flash, and then pendrive will be ready for testing.

12 Appendix F: n-Link® Binary Files for Embedded Platforms

Starting from from the n-Link® modules' OneBox-Mobile software release version 1.1.0, Redpine offers pre-compiled binary files for n-Link® modules to enable customers to evaluate the software on specific embedded processor platforms. The platforms supported for the current release are listed below:

1) Freescale i.MX6 and i.MX53
2) Atmel ATSAM9G45and AT91SAM9M10

The sections below explain the usage of the binaries on these platforms and also how to generate the binaries in case the OneBox-Mobile software source is available. These binaries are available in the "release/Binary_files" folder.

12.1 Freescale i.MX6

12.1.1 Hardware Requirements

- RS9113 Evaluation Kit. The contents are as follows:
  - RS9113 Module Evaluation Board
  - Micro A/B-type USB cable
  - SDIO Adaptor Cable
  - SPI Adaptor Cable
  - USB Pen Drive
- i.MX 6SoloLite Evaluation Kit. The kit contents are as follows:
  - Board: MCIMX6SLEVK
  - Cables: Micro USB-B-2-USB-Type A male, V2.0
  - Power supply: 100/240 V input, 5 V, 2.4 A output W/AC adaptor
  - Two SD cards: Programmed Android™
- Linux PC with Serial-to-USB drivers installed – this will be used to communicate with the i.MX6 platform.

12.1.2 Software Requirements

- Toolchain, BSP and Ubuntu Linux OS package for i.MX6 - Kernel version 3.0.35.
- OneBox-Mobile Software Release package

12.1.3 Hardware Setup

1) Connect the i.MX6 board to the Linux PC using the Micro A/B-type USB cable – the cable has to be connected to port J26 (microUSB) of the board.
2) Connect the Redpine Evaluation Board (EVB) to the i.MX6 board using the SDIO adaptor or Micro A/B-type USB cable (both included in the Redpine Evaluation Kit), depending on which Host Interface is needed.
   a. i.MX6 + Redpine EVB with USB: Connect USB cable to J10 (USB) port of i.MX6
   b. i.MX6 + Redpine EVB with SDIO: Connect SDIO Adapter to SD3 port of i.MX6

3) Preparing the MMC Card: An SD/MMC memory card is required to transfer the bootloader and kernel images for initializing the partition table and copy the root file system. This is included in the i.MX6 Evaluation Kit, but is programmed for Android OS. Refer to the i.MX_6SoloLite_EVK_Linux_User's_Guide.pdf document provided by Freescale as part of the L3.0.35_4.1.0_LINUX_MMDOCS documentation package, to prepare the SD/MMC card for Linux OS with kernel version 3.0.35.

12.1.4 Cross Compile and Copy OneBox-Mobile Software

If the OneBox-Mobile software's source is available, cross compile the OneBox-Mobile software as per the instructions below for i.MX6.

Assign the DEF_KERNEL_DIR variable in the Make file (present in the "release" folder) as follows (assuming the kernel source is available in the "/lib/modules" folder):

```
DEF_KERNEL_DIR := /lib/modules/linux-3.0.35_SOLOLITE_hw
```

The "make" command for the i.MX6 is as follows, assuming the toolchain is present in the "/toolchain/opt/freescale" folder:

```
# make ARCH=arm
CROSS_COMPILE=/toolchain/opt/freescale/FWIOCUA0R1M1P1/TOOLS/cr
oss/bin/arm-mv5sft-linux-gnueabi-
```

Next, plugin the SD/MMC card to the PC and execute the commands below to copy the pre-compiled binaries or the binaries generated above to the SD/MMC card.

```
# sudo mount /dev/sdb1 /mnt

# mkdir -p /mnt/home/rsi

# cp -r <path to OneBox-Mobile package>/host/release
/mnt/home/rsi

# umount /mnt
```

Plugin the SD/MMC card into the i.MX6 board and follow the boot procedure. Once the bootup and login are completed, go to the /home/rsi/release folder and follow the procedure explained in Section 4.

12.2 Freescale i.MX53

12.2.1 Hardware Requirements

- RS9113 Evaluation Kit. The contents are as follows:
    - RS9113 Module Evaluation Board
    - Micro A/B-type USB cable
    - SDIO Adaptor Cable
    - SPI Adaptor Cable
    - USB Pen Drive
- IMX53QSB: i.MX53 Quick Start Board. The kit contents are as follows:
    - i.MX53-QUICK START Board
    - microSD Card preloaded with Ubuntu Demonstration Software
    - USB Cable (Standard-A to Micro-B connectors)

- 5V/2.0A Power Supply
- Quick Start Guide
- Documentation DVD

- Linux PC with Serial port – this will be used to communicate with the processor platform.
- Serial RS232 Cable

12.2.2 Software Requirements

- Toolchain, BSP and Linux OS package for i.MX6 - Kernel version 2.6.35.
- OneBox-Mobile Software Release package
- minicom/GTKTerm on the Linux PC

12.2.3 Hardware Setup

1) Connect the i.MX53 board to the Linux PC using the Serial RS232 cable.
2) Connect the Redpine Evaluation Board (EVB) to the i.MX53 board using the SDIO adaptor or Micro A/B-type USB cable(both included in the Redpine Evaluation Kit), depending on which Host Interface is needed.
3) Open a serial terminal program like minicom or GTKTerm and configure it with the following settings:
   a. Baud Rate: 115200
   b. Data bits: 8
   c. Stop bits: 1
   d. Parity: None
   e. Flow Control:
4) Preparing the MMC Card: An SD/MMC memory card is required to transfer the bootloader and kernel images for initializing the partition table and copy the root file system. This is included in the i.MX53 Evaluation Kit. Refer to the i.MX53_EVK_Linux_BSP_UserGuide.pdf document provided by Freescale as part of the IMX53_1109_LINUXDOCS_BUNDLEdocumentation package, to prepare the SD/MMC card for Linux OS with kernel version 2.6.35.

12.2.4 Cross Compile and Copy OneBox-Mobile Software

If the OneBox-Mobile software's source is available, cross compile the OneBox-Mobile software as per the instructions belowfor i.MX53.

Assign the DEF_KERNEL_DIR variable in the Makefile as follows (assuming the kernel source is available in the "/lib/modules" folder):

```
DEF_KERNEL_DIR := /lib/modules/linux-2.6.35.3
```

The "make" command for the i.MX53 is as follows:

```
# make ARCH=arm
CROSS_COMPILE=/toolchain/opt/freescale/usr/local/gcc-4.4.4-
glibc-2.11.1-multilib-1.0/arm-fsl-linux-gnueabi/bin/arm-none-
linux-gnueabi-
```

Next, plugin the SD/MMC card to the PC and execute the commands below to copy the pre-compiled binaries or the binaries generated above to the SD/MMC card.

```
# sudo mount /dev/sdb1 /mnt

# mkdir -p /mnt/home/rsi

# cp -r <path to OneBox-Mobile package>/host/release
/mnt/home/rsi

# umount /mnt
```

Plugin the SD/MMC card into the i.MX53 board and follow the boot procedure. Once the bootup and login are completed, go to the /home/rsi/release folder and follow the procedure explained in the Section 4.

12.3 Atmel AT91SAM9G45 and AT91SAM9M10[4]

12.3.1 Hardware Requirements

- RS9113 Evaluation Kit. The contents are as follows:
  - RS9113 Module Evaluation Board
  - Micro A/B-type USB cable
  - SDIO Adaptor Cable
  - SPI Adaptor Cable
  - USB Pen Drive
  - SAM9M10-G45-EK - ARM926-based eMPU Eval Kit. The kit contents are as follows:
    - Board: SAM9M10-G45-EK
    - Cables: One micro A/B-type USB cable, One serial RS232 cable, One RJ45 crossed cable
    - Power supply: Universal input AC/DC power supply, One 3V Lithium Battery type CR1225
- Linux PC with Serial port – this will be used to communicate with the processor platform.

12.3.2 Software Requirements

- Toolchain, BSP and Ubuntu Linux OS package for AT91SAM9G45 and AT91SAM9M10 - Kernel version 2.6.30
- OneBox-Mobile Software Release package
- minicom/GTKTerm on the Linux PC

12.3.3 Hardware Setup

1) Connect the Atmel board to the Linux PC using the Serial RS232 cable.
2) Connect the Redpine Evaluation Board (EVB) to the processor board using the SDIO adaptor or Micro A/B-type USB cable cable (both included in the Redpine Evaluation Kit), depending on which Host Interface is needed.

[4] The Linux kernel version used on the Atmel AT91SAM9G45/M10 is 2.6.30. This has been used to verify only the Wi-Fi mode. Bluetooth and ZigBee drivers are not compatible with this kernel version.

3) Power up the processor board.
4) Open a serial terminal program like minicom or GTKTerm and configure it with the following settings:
    a. Baud Rate: 115200
    b. Data bits: 8
    c. Stop bits: 1
    d. Parity: None
    e. Flow Control:
5) Connect the RJ45 cable between the PC and the board.
6) Follow the instructions given at http://www.at91.com/linux4sam/bin/view/Linux4SAM/GettingStarted to setup the board with the Linux OS kernel version 2.6.30.

12.3.4 Cross Compile and Copy OneBox-Mobile Software

If the OneBox-Mobile software's source is available, cross compile the OneBox-Mobile software as per the instructions belowfor the Atmel processor.

Assign the DEF_KERNEL_DIR variable in the Makefile as follows (assuming the kernel source is available in the "/lib/modules" folder):

```
DEF_KERNEL_DIR := /lib/modules/linux-2.6.30
```

The "make" command for the AT91SAM9G45/M10 is as follows, assuming the toolchain is present in the "/toolchain/opt/atmel" folder:

```
# make ARCH=arm CROSS_COMPILE=/toolchain/opt/atmel/arm-
2007q1/bin/arm-none-linux-gnueabi-
```

Follow the steps below to copy the pre-compiled binaries or the binaries generate above to the Atmel processor platform.

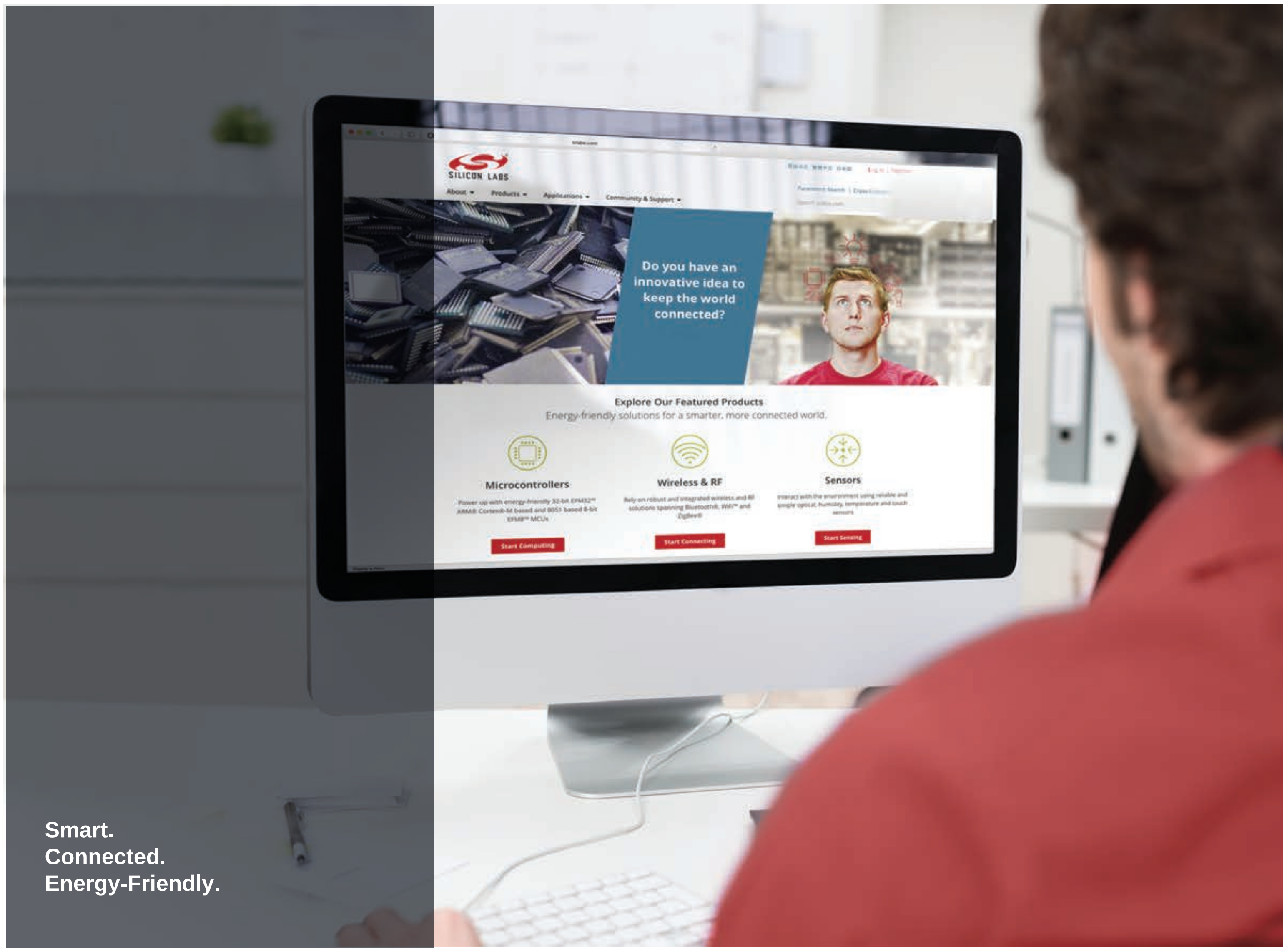
1) Ensure that the Linux PC and the Atmel platform are in the same subnet. The IP of the processor platform can be assigned using the minicom/GTKTerm terminal.

    ```
    # ifconfig <vap_name><ip_address>
    ```

    Example: `ifconfig eth0 192.168.1.24`
2) Power cycle the board.
3) Login as "root". There is not password required for the default credentials unless there has been a change done by the user.
4) Create a folder called "rsi" in the "/home" folder.
5) Copy the OneBox-Mobile binaries using the command below.

    ```
    # scp -r release/ root@192.168.1.24:/home/rsi
    ```
6) Follow the procedure explained in the Section 4to start using the OneBox-Mobile software.

*****


Smart.
Connected.
Energy-Friendly.
Do you have an innovative idea to keep the world connected?
Explore Our Featured Products
Energy-friendly solutions for a smarter, more connected world.
Microcontrollers
Wireless & RF
Sensors
Start Computing
Start Connecting
Start Sensing

Products
www.silabs.com/products

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com