Open Source Driver

Technical Reference Manual

Version 1.4

April 2020

Redpine Signals, Inc.
2107 N. First Street, #540
San Jose, CA 95131.
Tel: (408) 748-3385
Fax: (408) 705-2019
Email: info@redpinesignals.com
Website: www.redpinesignals.com

Disclaimer:

The information in this document pertains to information related to Redpine Signals, Inc. products. This information is provided as a service to our customers, and may be used for information purposes only. Redpine assumes no liabilities or responsibilities for errors or omissions in this document. This document may be changed at any time at Redpine's sole discretion without any prior notice to anyone. Redpine is not committed to updating this document in the future.

Table of Contents

1 Overview

RSI open source driver is a SoftMAC driver which interacts with the Linux wireless MAC layer i.e.mac80211. This driver currently supports Redpine's 9113 and 9116 chipsets. RSI open source driver is a group of simple and efficient kernel modules which can be ported to any embedded platform in-addition to X-86 platform. This has been tested on IMX and Caracalla boards.

1.1 Feature set

RSI open source driver supports all the features required for a user to use the module effectively. It also supports most demanding features Wi-Fi BT coexistence to offer single module for Wi-Fi and BT applications.

The feature set of RSI open source driver is outlined below.

- Wi-Fi station mode
- Wi-Fi Access Point mode
- QoS support (802.11e)
- wireless security modes i.e. WEP / WPA / WPA2 (802.11i)
- Regulatory support (802.11d)
- Basic and high throughput modes(802.11b/g/n)
- Background scanning and roaming
- Management frame protection (802.11w)
- Wi-Fi direct mode (P2P)
- Wi-Fi Protected Setup (WPS)
- HCI support for Bluetooth applications
- Wi-Fi station + Bluetooth classic coexistence mode
- Wi-Fi station + Bluetooth LE coexistence mode
- BT classic + BLE coexistence mode
- Wi-Fi station + Bluetooth classic + Bluetooth BLE Coexistence mode
- Wi-Fi AP + Bluetooth classic coexistence mode
- Wi-Fi AP + BT classic + Bluetooth LE coexistence mode

The subsequent sections explain the usage of the open source driver package and its features.

1.2 Host interface

Redpine 911x chipset supports two host interfaces: USB and SDIO. Host interface can be selected while building the modules.

2 Package

The RSI open source driver package contains the following files / folders.

- **Driver:** rsi

  Driver source is present in 'rsi' directory. Same driver can be seen in Linux kernel @path 'drivers/net/wireless/rsi'.

- **Firmware:**

  This folder contains the firmware files needed to load into the device. Any one or all of the files need to be moved to the Linux firmware path i.e. '/lib/firmware' or /lib/firmware/updates' based on the application of the module. The files which are present in the Firmware folder are outlined below:

  - RS9113_WLAN_QSPI.rps [Only Wi-Fi mode]
  - RS9113_WLAN_BT_DUAL_MODE.rps [Wi-Fi station + BT/BLE mode or BT alone modes]
  - RS9113_WLAN_AP_BT_DUAL_MODE.rps [Wi-Fi AP + BT/BLE mode or BT alone modes]
  - RS9116_NLINK_WLAN_IMAGE.rps [Only Wi-Fi alone for RS9116 Flash mode]
  - RS9116_NLINK_WLAN_BT_IMAGE.rps [Wi-Fi station + BT/BLE mode or BT alone modes for RS9116 Flash mode]
  - pmemdata [Wi-Fi station + AP for RS9116-RAM mode]
  - pmemdata_wlan_bt_classic [Wi-fi(STA + AP) + BT(DUAL) for RS9116-RAM mode]
  -

- **Release Notes.txt:**

  This file contains the information about the current release like release version, release date, Driver and Firmware versions, new features added in the current release, any bug fixes done in the current release and so on.

- **Readme.txt**

  It is a quick guide to build and install the driver. it also contain how to enable debug zone in case of troubleshoot, how to get driver and firmware version.

- **Scripts:**

  This folder contains useful shortcut scripts to start the driver in different modes i.e. station, AP, Wi-Fi direct and the required configuration files.

- **Documents**

  This folder contains useful documents needed for using the device and the driver.

3 Compilation & Installation / Un-installation instructions

3.1 Extract

Extract the package using the following command.

- # tar -xzvf RS911x.NB0.NL.XXX.LNX.<version>.tgz

3.2 Build

- Go to the package.
  # cd RS911x.NB0.NL.XXX.LNX.<version>
- Copy all the firmware files present in Firmware folder to /lib/firmware.
  # cp Firmware/* /lib/firmware
- There are two way to build the driver.
  i. Build the driver from the local path.
  ii. Build the driver from kernel source.

3.2.1 To build driver from local path:

Configure build flags in driver source.

cd rsi

Build Flags.

Given below are the build flags to be set based on the usage of driver. Selecting the required options shall reduce the binary size which is important for kernel modules particularly on embedded platforms.

a. KERNELDIR

Provide the kernel source path here. For example on X-86 below path is used.

KERNELRELEASE=$(Shell uname -r)

KERNELDIR=/lib/modules/$(KERNELRELEASE)/build

b. CONFIG_RSI_COEX_MODE

Enable this flag when Wi-Fi and BT or Wi-Fi and ZigB coexistence mode is used.

c. CONFIG_RSI_P2P

Enable this flag when Wi-Fi direct mode is to be used.

d. CONFIG_RSI_DEBUGFS

Enable this flag when RSI driver debug file system options are to be used.

e. CONFIG_HW_SCAN_OFFLOAD

Enable this flag to use Hardware scan offload feature. By default this flag is set and it is recommended to keep it enabled (particularly when using back ground scan and roaming feature).

f. CONFIG_RSI_NO_SDIO_MULTIBLOCK

Enable this flag if the platform's SDIO host controller does not support multi block mode.

g. CONFIG_RS9116_FLASH_MODE

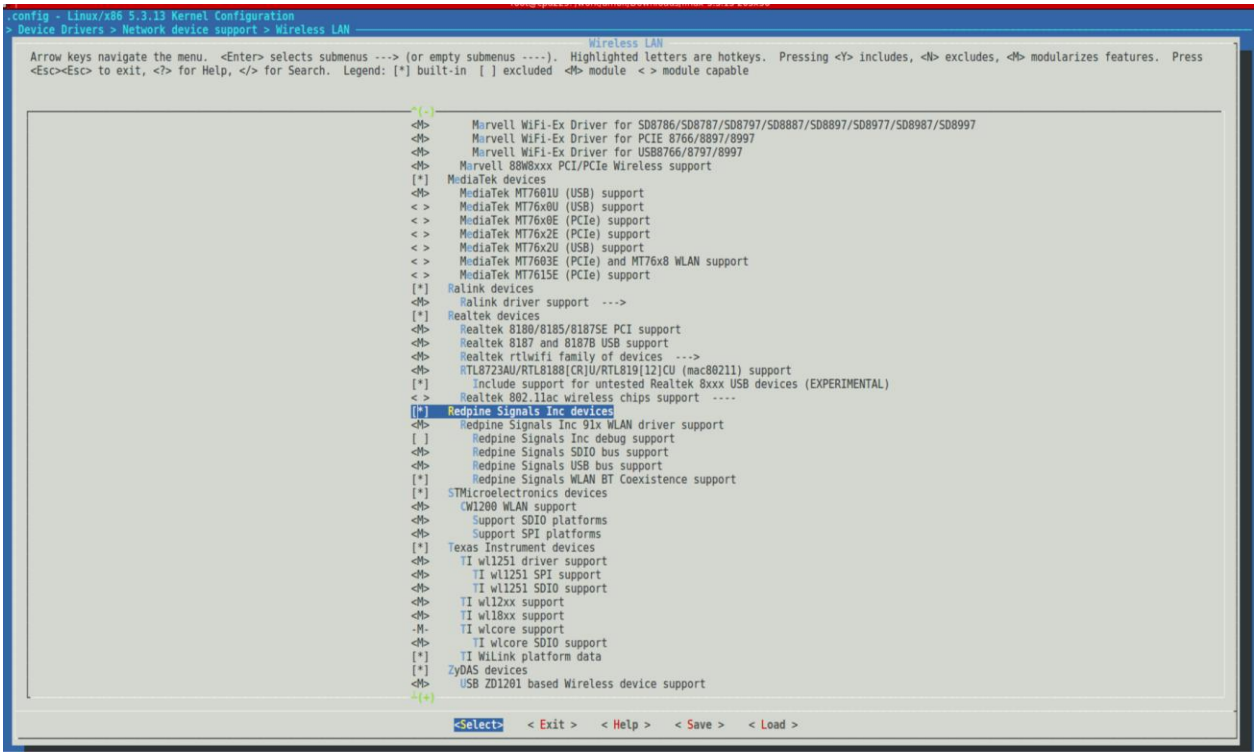
Enable this flag for RS9116 chip Flash mode FW load support.

- Build the driver using make command.
  # make

3.2.2 To build from kernel source:

- Copy the driver 'rsi' to <kernel_source_path> /drivers/net/wireless.
- Move Makefile to Makefile_local.
- Move Makefile_ker to Makefile.
- Give 'make menuconfig' from kernel source directory.
  - **$ make menuconfig**
- Go to 'Device Drivers->Network device support->Wireless LAN'.
- Select 'Redpine Signals Inc' devices.
- Select the SDIO/USB bus support depending on requirement.

  You will see the below screen with all the build options mentioned above. Select the required options.



- Build driver by using the below commands:
  - **$ make SUBDIRS=drivers/net/wireless/rsi**

3.3 Install

Build step shall generate 3 binaries i.e. rsi_91x.ko, rsi_usb.ko and rsi_sdio.ko

In order to install the driver, use the following commands:

1. To use single or multiple modules with single dev_oper_mode, insert rsi_91x.ko as below

    # insmod rsi_91x.ko rsi_zone_enabled=<val>dev_oper_mode=<mode>

    # insmod rsi_usb.ko

    # insmod rsi_sdio.ko sdio_clock=<clk_val>

    Note: Here “clk_val” is 1 to 50 (in MHz’s).

2. To use multiple modules with multiple dev_oper_modes, insert rsi_91x.ko as below

    # insmod rsi_91x.ko rsi_zone_enabled=<val>
    dev_oper_mode=<mode1>,<mode2>, ... , <mode5>

    # insmod rsi_usb.ko

    #insmod rsi_sdio.ko

    Note: Max support for mulltiple dev_oper_modes for multiple modules in RSI driver are 5(<mode1>,....., <mode5>).You can install either USB or SDIO or both depending on the application.

dev_oper_mode

Device operating mode indicates the operating mode to be used with the device. Below table provides the operating mode details with its constraints.

Seq No	Operating mode	Protocol support				No. of Clients
		STA	AP	BT EDR	BT LE	
1	1	√	X	X	X	N/A
2	1	√	√	X	X	N/A
2	1	X	√	X	X	32 Clients
3	4	X	X	√	X	N/A
4	5	√	X	√	X	N/A
5	6	X	√	√	X	32 Clients
6	8	X	X	X	√	N/A
7	9	√	X	X	√	N/A
8	12	X	X	√	√	N/A
9	13	√	X	√	√	N/A
10	14	X	√	√	√	4 Clients

If any invalid mode is passed to the module, driver returns error and exit. You can check the error message debug logs.

For modes 4, 8 and 12 build flag CONFIG_RSI_BT_ALONE should be enabled.
For modes 5, 9, 13, 6, and 14, build flag CONFIG_RSI_COEX_MODE should be enabled.

Both SDIO and USB modules can be installed depending on the requirement. After successful installation, a new wireless interface shall be created. The name of the interface depends on the operating system network subsystem’s naming conventions. For example on Ubuntu machine it creates wlan0, wlan1 and so on.

3.4 Uninstall the driver

In order to un-install the driver, use the following commands:

rmmod rsi_usb

```
# rmmod rsi_sdo
# rmmod rsi_91x
```

After un-installing the driver, the created wireless interface is no more seen.

4 Device Configuration commands

Open wireless configuration tools like **'iw'** and **'iwconfig'** can be used to configure the some of the standard wireless parameters. Some of the configuration options are provided in the below subsections.

4.1 Requirements

For using these tools, user may need to know the interface name and phy device number.

4.1.1 Interface name

Name of the interface is created after successful installation of the driver. This can be seen using 'ifconfig' command. The interface name associated with Redpine MAC **"00:23:a7:XX:XX:XX"** is the interface name of the driver.

Example:

ifconfig

wlan0 Link encap: Ethernet HWaddr 00:23:a7:b9:ab:44

- inet addr: 192.168.0.7 Bcast:192.168.0.255 Mask:255.255.255.0
- UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
- RX packets: 358 errors:0 dropped:0 overruns:0 frame:0
- TX packets: 300 errors:0 dropped:0 overruns:0 carrier:0
- collisions:0 txqueuelen:1000
- RX bytes:39470 (39.4 KB) TX bytes:28673 (28.6 KB)

4.1.2 Phy device number

This is the physical device number of the system for the device. It can be obtained by using the below command.

iw dev <intf_name> info

Interface wlan0

- ifindex 5
- wdev 0x100000001
- addr 00:23:a7:b9:ab:44
- type managed
- wiphy 1
- channel 6 (2437 MHz), width: 20 MHz (no HT), center1: 2437 MHz

As can be seen, in this case, phy<X> is termed as phy5.

4.2 Background Scanning and roaming

For roaming requirement, enable CONFIG_HW_SCAN_OFFLOAD in Makefile.

Background scanning and roaming can be verified using wpa_supplicant. To use this facility, user needs to ensure the flag CONFIG_BGSCAN_SIMPLE' is enabled in the supplicant build configuration file (.config). This will enable building BGSCAN SIMPLE module which is responsible for requesting background scans for the purpose of roaming within ESS. If this option is not enabled, rebuild wpa_supplicant binary with this option.

'bgscan' parameters use the following format:

bgscan="simple:<short_bgscan_intrvl_in_secs>:<signal_strength_thrshld>:<long_bgscan_intrvl_in_secs>

This line should be present either inside a network block or outside of all network blocks based on the requirement.

Eg: bgscan="simplel: 30:-45:300"

4.3 Antenna Selection

To select external antenna, follow the steps given below:

ifconfig wlan0 down

iw phy <phyX> set antenna 1 0

ifconfig wlan0 up

By default internal antenna is configured.

To select internal antenna, use the command given below:

ifconfig wlan0 down

iw phy <phyX> set antenna 0 0

ifconfig wlan0 up

4.4 Country Setting

To set a country, use the command given below:

iw reg set <country_code>

Example:

iw reg set IN (For India)

iw reg set JP (For Japan)

iw reg set GE (For Germany)

To check the current country / regulatory domain, use the command given below:

iw reg get

To understand the list of channels allowed in the current regulatory domain, please check the below link.

https://en.wikipedia.org/wiki/List_of_WLAN_channels

4.4.1 Regulatory mapping

Mapping of country code to regulatory region code for Caracalla

S.No	Country	Country code	Region code	S.No	Country	Country code	Region code
1	Australia	AU	ETSI	2	Austria	AT	ETSI
3	Belgium	BE	ETSI	4	Brazil	BR	WORLD
5	Canada	CA	FCC	6	Chile	CL	WORLD
7	China	CN	WORLD	8	Colombia	CO	FCC
9	Czech Republic	CZ	ETSI	10	Denmark	DK	ETSI
11	Finland	FI	ETSI	12	France	FR	ETSI
13	Germany	DE	ETSI	14	Hong Kong	HK	WORLD
15	India	IN	WORLD	16	Indonesia	ID	WORLD
17	Ireland	IE	ETSI	18	Israel	IL	ETSI
19	Italy	IT	ETSI	20	Japan	JP	TELEC
21	Republic of Korea	KR	WORLD	22	Luxembourg	LU	ETSI
23	Malaysia	MY	WORLD	24	Mexico	MX	FCC
25	Morocco	MA	WORLD	26	Netherlands	NL	ETSI

4.5 Software RFKill

The open source driver has support for RFKill command. As a pre-requisite, please install the rfkill package.

- In order to list out the wireless interfaces in the system, use the command given below.

  # rfkill list

- To block the 911x Wi-Fi interface, use the command given below:

  # rfkill block <interface_number_listed_in_rfkill_list>

- To unblock the 911x Wi-Fi interface, use the command given below:

  # rfkill unblock <interface_number_listed_in_rfkill_list>

4.6 Wake on WLAN (WoWLAN) – WLAN Connectivity

- WoWLAN is a feature where device can go to sleep until a specific external trigger is received through WLAN.
- WoWLAN works based on Station connectivity with an AP. Hence, please make sure station is connected to an AP and IP is obtained before issuing WoWLAN suspend/hibernate.

- Open source driver has support for WoWLAN in station mode. This feature is tested by using the following applications:
  https://packages.debian.org/stable/net/wakeonlan
  https://packages.debian.org/stable/net/etherwake
- For eatherwake application, please edit ether-wake.c and go to main() function, update the ifname with the interface name of our device. Compile the application using below command.
  # gcc ether-wake.c –o etherwake
- The steps to be followed in order to configure & test WoWLAN are outlined below:
  - Enable CONFIG_RSI_WOW in the rsi driver Makefile.
  - Initiate connection to an AP.
  - Using dhclient, get the IP address or assign an IP address statically.
  - Configure device for WoWLAN by using the command given below:
    # iw phy <phyX> wowlan enable magic-packet
  - Issue system sleep (For example in S3)
    # systemctl suspend
  - Using the etherwake application, send the magic packet from a desktop connected to the AP to the device.
    # ./etherwake 00:23:a7:xx:xx:xx
    Device would now wakeup the system.
    or
  - Using the wakeonlan application, send the magic packet from a desktop connected to the AP to the device.
    # ./wakeonlan 00:23:a7:

- To check WoWLAN in S4 state use below command for sleep
  # systemctl hibernate
- To check WoWLAN in S5 state use below command for sleep
  # shutdown

4.7 Wake on WLAN (WOWLAN) – BT Connectivity

The steps to be followed in order to make WoWLAN work in coex mode are outlined below:

- Insert Co-Ex driver.
- Connect to an AP (Access Point).
- Bring up the Bluetooth.
- Enable WoWLAN mode.
- Test BT functionality - Connectivity / data transfer.
- Give HCI reset command in BT. - hcitool -i hci0 cmd 0x03 0x0003 or hciconfig hci0 down.
- Suspend the board (in S3 mode).
- Wake up the board by providing keyboard interrupt or with an on-air pattern.
- Test BT functionality - Connectivity / data transfer.

- Basically, BT interface should be reset before suspending the board so that there will not be any activity on BT interface. Otherwise, the Rx BT packets would wake up the board.

5 Wi-Fi station mode

The Wi-Fi station mode settings can be configured using wpa_supplicant or the Network Manager CLI. Below we have captured the steps for both of them.

5.1 Configure station using WPA_supplicant

After having configured the device by using the commands in section 4, the user would need to start the supplicant. To start the supplicant on the command line, please enter:

- #wpa_supplicant -i wlan0 -Dnl80211 –c sta.conf –ddddt > supp.log &

Where –i option specifies the Wi-Fi interface name

-D specifies the driver interface to be used. In open source driver it is net80211.

-c specifies the supplicant configuration file

-d specifies the log level of supplicant. You can apend more d's to it for more detailed logs.

-f specifies the output file to redirect wpa_supplicant logs

Below are sample network blocks for various security modes that can be put up in the sta.conf file. Please fill in the information (ssid, psk etc) corresponding to the AP you intend to connect to in this file.

Open:

```
network={
    ssid=”open”
   key_mgmt=NONE
   priority=3
}
```

WEP:

```
network={
     ssid="static-wep-test"
     key_mgmt=NONE
     wep_key0="abcde"
     wep_key1=0102030405
     wep_key2="1234567890123"
     wep_key3=”1236666666666”
     wep_tx_keyidx=0
     priority=2
}
```

WPA/2: To enable WPA/2 with all valid ciphers:

```
network={
    ssid="wpa"
    psk="12345678"
    priority=1
}
```

5.2 Configure station using the Network-Manager CLI (nmcli)

Below are the specific commands that can be used for connection using the Network Manager CLI(nmcli):

- To view the currently available network connections, enter the following on command prompt:

```
# nmcli con show
NAME    UUID                                  TYPE             DEVICE
eth0    96a5deb0-5eb0-41e1-a7ed-38fea413f9c8  802-3-ethernet   eth0
wlan0   91451385-4eb8-4080-8b82               802-11-wireless  wlan0
```

- To view the list of access points, issue the below command:

```
#nmcli dev wifi list
```

Below is a sample output:

```
$ nmcli dev wifi list
 SSID            MODE   CHAN RATE      SIGNAL  BARS          SECURITY
 FedoraTest      Infra  11   54 MB/s   98      â–‚â–„â–†â–ˆ  WPA1
 Red Hat Guest   Infra  6    54 MB/s   97      â–‚â–„â–†â–ˆ  WPA2
 Red Hat         Infra  6    54 MB/s   77      â–‚â–„â–†_   WPA2 802.1X
```

- To connect to an AP with WPA/2 security, issue the below command:

```
#nmcli dev wifi connect hello password 12345678 wlan0
```

(hello is the AP's SSID and password is 12345678, interface name is wlan0)

- To connect to an AP without security, issue the below command:

```
#nmcli dev wifi connect open_source wlan0
```

open_source is the SSID

- To connect to an AP with WEP security, issue the below command:

```
#nmcli dev wifi0 connect wep_ap password
```

- To review the status of the devices and the connections, issue the below command:

```
#nmcli dev status
```

Sample output after connection is as follows

```
nmcli dev status
```

DEVICE TYPE STATE CONNECTION

wlan0 wifi connected my-ssid

eth0 ethernet unavailable --

As can be seen, the STATE corresponding to wlan0 interface shows connected.

- To up an interface, using nmcli, issue the below command:

#nmcli con up id wlan0

- To down an interface using nmcli, issue the below command:

#nmcli dev disconnect wlan0

6 Power Save

The module broadly supports two types of power save modes. They are outlined below:

- Low Power (LP) Mode: The PHY (RF and Baseband) and LMAC sections are powered off but the UMAC and Host Interface sections of the module are powered on and fed a low frequency clock. The module responds to commands/requests from the Host processor immediately in this mode.
- Ultra-low Power (ULP) Mode: A majority of the module is powered off except for a small section which has a timer and interrupts logic for waking up the module. The module cannot respond to the Host processor's commands/requests unless and until it gets wake up because of timeout or because of an interrupt asserted by Host processor. The sleep entry/exit procedures in this mode are indicated to the Host processor either through a packet based or signal based handshake. This mode is supported only for SDIO host interface.

6.1 LP power save mode:

- Install the driver as follow to enable LP power save

  #insmod rsi_91x.ko rsi_zone_enabled=1 dev_oper_mode=<value> ps_sleep_type=1

  Here value is dev_oper_mode mentioned in section 3.3

  # insmod rsi_usb.ko (OR) # insmod rsi_sdio.ko

6.2 ULP Power Save mode

- Install the driver as follow to enable ULP power save

  # insmod rsi_91x.ko rsi_zone_enabled=1 dev_oper_mode=<value> ps_sleep_type=2 ulp_handshake_mode=<value1>

  Here <value> is dev_oper_mode mentioned in section 3.3

  Here <value1=1> is for GPIO based handshake, 2 for packet based handshake

  # insmod rsi_sdio.ko

6.2.1 GPIO Handshake

- For GPIO handshake driver requires two GPIO pins. These need to be configured in code.
- Also, In order to use GPIO handshake, enable USE_GPIO_HANDSHAKE flag in Make file.

6.3 Power save configuration to device

Power save can be enabled or disabled through command line using iw commands. By default 802.11 default power save is enabled if Coex mode is enabled. UAPSD is enabled based on AP's UAPSD configuration.

Following are the commands used in power save configuration.

- Enable the power save:

iw dev <interface_name> set power_save on

- Disable power save:

  # iw dev < interface_name> set power_save off

- Check the power save status:

  # iw dev <interface_name> get power_save

  Here interface_name will vary from host to host we can get that interface name with below command's

  # iw dev

- In case of BT coexistence with WiFi, we need to give bt power save command.

  # hcitool -i hci0 cmd 0x3f 0x0003 0x01 0x02 0xff

7 Bluetooth commands

The hcitool and hciconfig commands are used to control and configure parameters for the Bluetooth interface. The HCI commands explained here are the most frequently used commands.

For other HCI commands please refer the Bluetooth specification, Volume 2 Part E, Chapter7 from www.bluetooth.org

Reset	
Description	This command is used to issue a soft reset to the Bluetooth module
Default Value	-
Input Parameters	None
Output Parameter	None
Reset Required	No.
Usage	`hcitool -i <hciX> cmd 0x03 0x03`
Read Local Version Information	
Description	This command is used to read the local version information
Default Value	-
Input Parameters	None
Output Parameter	HCI version HCI revision LMP version Manufacturer name LMP subversion
Reset Required	No.
Usage	`hcitool -i <hciX> cmd 0x04 0x01`
Read Local Supported Commands	
Description	This command is used to read the local controller supported HCI commands.
Default Value	-
Input Parameters	None
Output Parameter	List of supported commands (64 bytes of bit field)

Reset Required	No.
Usage	hcitool -i <hciX> cmd 0x04 0x02
Get Local BD Address	
Description	This command is used to get the local BD Address
Default Value	-
Input Parameters	None
Output Parameter	6 Byte BD Address
Reset Required	No.
Usage	hcitool -i <hciX> cmd 0x04 0x09
Start Inquiry	
Description	This command is used to start the Inquiry process
Default Value	
Input Parameters	LAP (3 Bytes): (0x9E8B00 – 0x9E8B3F) Inquiry duration: (0x01 to 0x30 -> 1.28 to 61.44 Seconds) Number of responses: (0x01 – 0xFF)
Output Parameter	None.
Reset Required	No.
Usage	hcitool -i <hciX> cmd 0x01 0x01 <LAP> <duration> <no_of_responses>
Write Local Name	
Description	This command is used to set the local device name
Default Value	
Input Parameters	Name of the device.
Output Parameter	None.
Reset Required	No.
Usage	hcitool -i <hciX> cmd 0x03 0x13 <name>

Sniff Mode command	
Description	This command is use to keep RSI BT device in Sniff Mode.
Default value	None
Input Parameter	Connection Handle -0x1 Sniff max interval -0x0190 (250 msec) Sniff min interval – 0x0190(250 msec) Sniff attempt – 0x0005(6.25 msec) Sniff timeout – 0x0002 (2.50 msec)
Output parameter	None
Reset Required	No.
Usage	hcitool –i hci<x> cmd 0x02 0x0003 <Connetion handle > < Sniff Max Interval> <Sniff Min Interval> <sniff attempt> <sniff timeout>
Example	hcitool -i hci0 cmd 0x02 0x0003 0x01 0x00 0x90 0x01 0x90 0x01 0x05 0x00 0x02 0x00

Below set of commands, list the steps to be followed to successfully pair 911x BT device with a third-party BT dongle/Mobile phone. Please make sure bluez-tools are installed on the system where the 911x driver is installed.

#bluetoothctl -a

[bluetooth]# power on

[bluetooth]# scan on

[bluetooth]# pair 64:CC:2E:9C:23:BA

[bluetooth]# connect 64:CC:2E:9C:23:BA

Attempting to connect to 64:CC:2E:9C:23:BA

[bluetooth]# disconnect CC:2E:9C:23:BA

Attempting to connect to 64:CC:2E:9C:23:BA

[CHG] Device 64:CC:2E:9C:23:BA Connected: yes

[CHG] Device 64:CC:2E:9C:23:BA Modalias: bluetooth:v001Dp1200d1436

[CHG] Device 64:CC:2E:9C:23:BA UUIDs: 00001103-0000-1000-8000-00805f9b34fb

[CHG] Device 64:CC:2E:9C:23:BA UUIDs: 00001105-0000-1000-8000-00805f9b34fb

[CHG] Device 64:CC:2E:9C:23:BA UUIDs: 00001106-0000-1000-8000-00805f9b34fb

[CHG] Device 64:CC:2E:9C:23:BA UUIDs: 0000110a-0000-1000-8000-00805f9b34fb

[CHG] Device 64:CC:2E:9C:23:BA UUIDs: 0000110c-0000-1000-8000-00805f9b34fb

8 AP mode

We have tested AP mode configuration with the hostapd application. Following are the steps to be followed in order to configure AP mode in the device:

- Install hostapd.

  # apt-get install hostapd

- Configure hostapd

  Create a hostapd configuration file (for eg: ap.conf) and add below:

  Set interface name:

  ### Wireless network name ###

  interface=wlan0

  Set driver name:

  driver=nl80211

  Set country name code in ISO/IEC 3166-1 format. This is used to set regulatory domain. Set as needed to indicate country in which device is operating. This can limit available channels and transmit power.

  ### (IN == INDIA, UK == United Kingdom, US == United Stats and so on) ###

  country_code=IN

  Set your SSID:

  ssid=Redpine

  Set operation mode (a = IEEE 802.11a, b = IEEE 802.11b, g = IEEE 802.11g)

  hw_mode=g

  Set channel number (some driver will only use 0 as value)

  channel=6

  Set wpa mode to 2:

  wpa=2

  Set your passphrase (WiFi password):

  wpa_passphrase=MyWiFiPassword

  Set key and auth options for WPA2:

  ## Key management algorithms ##

  wpa_key_mgmt=WPA-PSK

  ## Set cipher suites (encryption algorithms) ##

  ## TKIP = Temporal Key Integrity Protocol

  ## CCMP = AES in Counter mode with CBC-MAC

  wpa_pairwise=TKIP

  rsn_pairwise=CCMP

Shared Key Authentication

auth_algs=1

Save and close the file.

- Start the hostapd application:

  # ./hostapd ap.conf –dddt > log_file &

- Run dhcp server script (In scripts folder) to assign IPs to client.

  # sh start_dhcp_server.sh

In the scripts folder, several hostapd config files are provided to start the AP in various modes like open (ap_open.conf), WPA/2-PSK (ap_wpa.conf). User could use these conf files instead of creating one

8.1 ACS with Hostapd

Following steps should be followed for Auto Channel Selection using Hostapd:

- Compile hostapd by Enabling CONFIG_ACS in hostapd .config file
- Configure hostapd configuration file (for ex ap_wpa.conf) with following

  #channel= 0

  # acs_num_scans=5

- Insert the driver and run hostapd with below commands

#./hostapd ap_wpa.conf –dddt > log_file_name &

9 Sniffer mode

Following are the steps which need to follow to use RS9116 and RS9113 modules as sniffer with Open Source Driver.

- Install the driver using the below commands
  # insmod rsi_91x.ko rsi_zone_enabled=1 dev_oper_mode=1 driver_mode_value=7
  # insmod rsi_usb.ko (OR) # insmod rsi_sdio.ko
- Make sure that interface must be Down if not use below command to down the interface
  # ifconfig <interface name> down
- Change the default interface to monitor using below command
  # iwconfig < interface name> mode monitor
  OR
  # iw dev <interface name> set type monitor
- Make interface up
  # ifconfig <interface name> up
- Set the channel in which you want to capture the on air packets
  # iwconfig <interface name> channel < channel no>
  OR
  # iw dev <interface name> set channel <channel no>
- To change the bandwidth use below command
  # iw dev <interface name> set channel <channel no> <HT40+/HT40-/HT20>
  →HT40+/HT40- for 40 MHz bandwidth
  →HT20 for 20 MHz bandwidth
- Use any network packet analysis tool to see captured packets
  # wireshark &
  OR
  # tcpdump &

10 Steps to configure 802.11W

10.1 Configuring and Compiling Driver for PMF in client mode:

- Enable CONFIG_IEEE80211W=y in wpa_supplicant .config
- Enable WPA-PSK-SHA256 as key_mgmt in network block in supplicant sta_settings.conf
  pmf=1/2, PMF is enabled/required correspondingly .

```
pmf=2
network = {
ssid="REDPINE_AP_MFP"
pairwise=CCMP
group=CCMP
key_mgmt=WPA-PSK-SHA256
psk="12345678"
proto=WPA2
priority=1
}
```

- Configure AP as MFP Capable/Required

10.2 Configuring and Compiling Driver for PMF in AP mode:

- Enable CONFIG_IEEE80211W=y in hostapd .config
- Enable WPA-PSK-SHA256 as key_mgmt in hostapd_ccmp.conf
  pmf=1/2, PMF is enabled/required correspondingly .
- Make sure below options are enabled apart from your configuration.

```
# This field is a bit field that can be used to enable WPA (IEEE 802.11i/D3.0)
and/or WPA2 (full IEEE 802.11i/RSN):
# bit0 = WPA
# bit1 = IEEE 802.11i/RSN (WPA2) (dot11RSNAEnabled)
wpa=2
# ieee80211w: Whether management frame protection (MFP) is enabled
# 0 = disabled (default)
# 1 = optional
# 2 = required
ieee80211w=2
wpa_key_mgmt =WPA-PSK-SHA256
group_mgmt_cipher=AES-128-CMAC
```

11 WPS

11.1 PBC method

- Start the supplicant in station mode as mentioned in section 5.1 without any network details in wpa_supplicant configuration file.
- Connect a PC to AP and get IP. Open the AP's WPS page.
- Give the wpa_cli command to initiate connection.

  # wpa_cli –i wlan0 wps_pbc [MAC address of AP]
- In AP's WPS page, a button presets for pbc. Click on the button.
- Connection will progress and you will see connection done in some time.

11.2 PIN method

- Start the supplicant in station mode as mentioned in section 5.1 without any network details in wpa_supplicant configuration file.
- Connect a PC to AP and get IP. Open the AP's WPS page.
- Give the wpa_cli command to initiate connection.

  # wpa_cli –i wlan0 wps_pin [Mac address of AP]

  This will generate a random pin and display below.
- In AP's WPS page, a Text box presets for pin. Enter the pin in the text box and press start connection.
- Connection will progress and you will see connection done in some time.

12 Wi-Fi direct (p2p) mode

This mode is used for peer to peer communication between Wi-Fi devices. Wi-Fi devices which want to communicate with each other can form a group. One of the device acts as group owner who acts as AP.

Wpa_supplicant can be used to verify this mode. This section presents the p2p connections in nl80211 interface only. Follow below steps:

- Enable CONFIG_RSI_P2P in driver Makefile and build the driver.
- Install the driver
- Like in station mode, wpa_supplicant configuration file need to be created and p2p specific parameters should be configured. Below is the sample p2p configuration file.

  ctrl_interface=/tmp/p2p

  update_config=1

  device_name=rsi_wfd

  manufacturer=Redpine Signals Inc

  model_name=RSI

  model_number=911x

  device_type=1-0050F204-1

  os_version=01020300

  config_methods=display push_button keypad

  p2p_listen_reg_class=81

  p2p_listen_channel=1

  p2p_oper_channel=11

  p2p_go_intent=0

  p2p_ssid_postfix=RS

- Run the supplicant

  # wpa_supplicant -i wlan0 -Dnl80211 -c p2p.conf -d -f p2p.log

  This step will create a separate p2p interface with name “p2p-dev-wlan0”.

P2P connection can be done using two methods namely group negotiation and adhoc group creation and joining clients.

12.1 Group negotiation method

In this method two modes can be verified i.e. group owner (GO) mode or group client mode.

12.1.1 P2P GO mode

- To verify this mode, in p2p configuration file, set the p2p_go_intent value to high number (14 or 15) running the wpa_supplicant. Maximum go_intent value is 15.

  p2p_go_intent=14

- Issue p2p device discovery command from wpa_cli.

  # wpa_cli -i p2p-dev-wlan0 -p/tmp/p2p

  > p2p_find

- This step will display near by p2p devices. Check whether target device you wish to connect is discovered or not. You can also check this using below command in cli.

  > p2p_peers

- You can either directly start the connection or if you wish to stop discovery issue below command

  > p2p_stop_find

- Issue connect command

  > p2p_connect <MAC_of_target> pbc

  Or if you wish to use pin method

  > p2p_connect <MAC_of_target> pin any

- This step will proceed to connection, you can see connection message. It also creates a new interface for the current session as “p2p-wlan0-1”.

- Start the dhcp server with the new interface

  # sh dhcp_server p2p-wlan0-1

12.1.2 P2P Client mode

- To verify this mode, set the p2p_go_intent value to a small number (1 or 2) before running the wpa_supplicant. Maximum go_intent value is 15.

  p2p_go_intent=1

- Issue p2p device discovery command from wpa_cli.

  # wpa_cli -i p2p-dev-wlan0 -p/tmp/p2p

  > p2p_find

- This step will display near by p2p devices. Check whether target device you wish to connect is discovered or not. You can also check this using below command in cli.

  > p2p_peers

- You can either directly start the connection or if you wish to stop discovery issue below command

  > p2p_stop_find

- Issue connect command

  > p2p_connect <MAC_of_target> pbc

  Or if you wish to use pin method

  > p2p_connect <MAC_of_target> pin any

- This step will proceed to connection, you can see connection message. It also creates a new interface for the current session as “p2p-wlan0-1”.

- Get the IP using dhclient.

  # dhclient –r;

  # dhclient –v p2p-wlan0-1

12.2 Autonomous GO creation method

In this method, group is created first and clients are joined.

- To verify this mode, use below command to start the group from wpa_cli after starting the wpa_supplicant.

  > p2p_group_add freq=<channel_freq>

  Passing channel_freq is optional. If not passed, some random channel shall be selected. This command will form a new group with interface name “p2p-wlan0-1” in the given channel. Device becomes group owner (AP kind) and starts beaconing in the selected channel with random ssid derived. SSID name looks like “DIRECT-XXXX”. This can be seen in the message displayed under the command.

- Start the dhcp server from command terminal

  # sh dhcp_server p2p-wlan0-1

- You can discover the p2p devices or directly attempt to join the client using below command.

  > wps_pbc

  Or

  > wps_pin any [This will generate a random key and displays down].

- From other p2p device, initiate connection. If pin method is used, you have to enter the above generate key when asked. This step will proceed for connection.

13 Revision History

Revision No.	Version No.	Date	Author	Changes
1	0.1	15th Feb	Fariya	Initial version
2	0.2	18th Mar	Prameela Rani	• Modified chapter 1 -Modified the content as well as added section **1.1** and **1.2**. • Modified chapter 2-Package • Modified the whole chapter 3-Compilation & Installation / Un-installation instructions • Modified the content of chapter 4-Device Configuration commands as well as content of section 4.1 and **4.2** • Removed chapter 8
			Venkanna	• Added section 4.7-Wake on WLAN (WOWLAN) – BT Connectivity under chapter 4
3	0.5	3rd Oct, 2017	Prameela Rani	• ZiGB basic details (build, install) added
4	0.6	5th Oct, 2017	Prameela Rani	• ZiGB usage details added
5	0.7	26th Oct, 2017	Prameela Rani	• WPS, P2P sections added • Regulatory mapping table added for Caracalla.
6	0.8	10th May 2018	Siva Rebbagondla	• Removed rsi_zigb.ko insertion and deletetion sections, as zigb modules added to rsi_91x
7	1.0	25th Oct 2018	Siva Rebbagondla	• Added sdio_clock module_param and RS9116_flash mode details.
8	1.1	13th Mar 2019	Ganapathi Raju	• Added Antenna Diversity in RSI_STA mode.
9	1.3	24th Dec 2019	Amol Hanwate	• Added Sniffer mode • Added ACS with Hostapd • Added dev_oper_mode 12 details • Removed Concurrent mode • Added Power save section
10	1.4	3rd April 2020	Amol Hanwate	• Added how to configure 11W • Added BT sniff mode command