Wi-Fi

User Guide

Issue 08

Date 2019-05-30

HiSilicon (Shanghai) Technologies Co., Ltd.

Address: New R&D Center, 49 Wuhe Road, Bantian,
Longgang District,
Shenzhen 518129 P. R. China

Website: http://www.hisilicon.com/en/

Email: support@hisilicon.com

About This Document

Purpose

This document describes the configurations, basic operations, and debugging methods of the Wi-Fi drivers. It also describes the precautions to be taken and solutions to common problems.

NOTE

- Unless otherwise stated, Hi3516D and Hi3516A contents are consistent.
- Unless otherwise stated, Hi3518E V201, Hi3516C V200, and Hi3518E V200 contents are consistent.
- Unless otherwise stated, Hi3516E V200, Hi3516E V300, and Hi3518E V300 contents are consistent.

Related Versions

The following table lists the product versions related to this document.

Product Name	Version
Hi3516A	V100
Hi3516D	V100
Hi3518E	V200
Hi3518E	V201
Hi3516C	V200
Hi3559A	V100
Hi3559C	V100
Hi3519A	V100
Hi3516E	V200
Hi3516E	V300
Hi3518E	V300
Hi3559	V200
Hi3556	V200
Hi3516D	V200

Intended Audience

This document is intended for:

- Technical support engineers
- Software development engineers

Change History

Changes between document issues are cumulative. The latest document issue contains all changes made in previous issues.

Issue 08 (2019-05-30)

This issue is the eighth official release, which incorporates the following changes:

The description about unsupported Wi-Fi modules is deleted.

Issue 07 (2019-05-15)

This issue is the seventh official release, which incorporates the following changes:

The RTL8188FTV section is added to the Linux version.

Issue 06 (2019-02-28)

This issue is the sixth official release, which incorporates the following changes:

The content about wifi_project is deleted.

In section 2.2, the content about boot configuration is added.

In sections 3.1 and 3.2, the descriptions about the porting of the RTL8189FTV and AP6181 drivers are added.

In chapter 4, the method of obtaining Wi-Fi tools is added.

Issue 05 (2018-02-10)

This issue is the fifth official release, which incorporates the following changes:

The descriptions of the Hi3559A V100 and Hi3559CV100 are added.

Section 1.2 is modified.

Sections 3.1.1, 3.1.2, 3.2.4, and 3.3.1 are modified.

Issue 04 (2018-01-15)

This issue is the fourth official release, which incorporates the following changes:

The description of the Hi3518E V20X is removed.

The contents in the entire documents are updated.

Chapter 4 is added.

Issue 03 (2015-09-25)

This issue is the third official release, which incorporates the following changes:

The contents related to the Hi3518EV 200, Hi3518E V201, and Hi3516C V200 are added.

Issue 02 (2015-02-10)

This issue is the second official release, which incorporates the following changes:

In section 3.3.3, the original step 3 and step 4 are combined.

Issue 01 (2014-12-30)

This issue is the first official release, which incorporates the following changes:

Chapter 2 Configurations

In section 2.1, the position of the section "Configuring Wireless Extension" is changed, the section "Configuring IPv6" is added, and the section "Configuring Netlink" is deleted.

In section 2.2, the sections "Configuring Wi-Fi Drivers" and "Wireless Manager" are deleted.

Chapter 3 Basic Operations

Section 3.1 and section 3.4 are added.

Issue 00B01 (2014-11-09)

This issue is the first draft release.

Contents

Figures

1 Introduction

1.1 Background

In the Wi-Fi industry, there are many chip vendors who produce various types of Wi-Fi chips. Each Wi-Fi chip has its own driver, which is not universal. Because Linux applications are widely used, Wi-Fi chips on the market have drivers which support the Linux platform. Therefore, the corresponding drivers must be ported to the platform when these Wi-Fi chips are used. The HiSilicon platform uses the standard Linux and can adapt to all the Linux-based Wi-Fi drivers. Therefore, the platform does not need to be modified. However, due to the difference in the Linux version, the corresponding Wi-Fi driver varies. You need to obtain the Wi-Fi driver of the corresponding Linux version from the Wi-Fi chip vendor or module vendor.

For different Wi-Fi chips, the methods of porting a Wi-Fi driver and performing an operation on the Wi-Fi chip are universal. This document uses several commonly used Wi-Fi chips as an example to describe the driver porting and debugging method, which provides reference for other Wi-Fi chips.

1.2 Scope

The HiSilicon and Realtek Wi-Fi chips are widely used in the camera field. For details about Hi1131S usage, see the Hi1131S V100 related documents. This document uses Realtek RTL8189FTV and RTL8188FTV as examples. It also uses only one driver version of the chip as an example. In actual use, you need to obtain the latest driver version from the Wi-Fi vendor.

Typically, a Wi-Fi device supports one or more working modes. There are four modes in total:

- SoftAP: access point, that is, a device that is used to connect a wireless device to the network. It can be considered as a wireless route.
- STA: station, that is, a wireless device client. It is available only after an AP is connected.
- DIRECT: Wi-Fi direct mode, which is also called P2P mode
- CONCURRENT: a mode in which AP and STA modes are supported simultaneously

In DIRECT mode, the compatibility is bad and the user experience is poor. The CONCURRENT mode is seldom used. Therefore, this document describes the basic operations of the STA and SoftAP modes.

In addition to testing of basic functions, Wi-Fi performance is an important indicator. This document also describes how to test the throughput and radio frequency (RF) indicators.

2 Configurations

2.1 Configuring the Kernel

2.1.1 Configuring the CFG80211

CFG80211 is the standard interface for the Wi-Fi drivers in the kernel and user-mode processes. It becomes more popular than WEXT. Only CFG80211 supports the Wi-Fi direct function.

Choose **Network support** > **Wireless**, and set **cfg80211** and **mac80211** to **M**, as shown in Figure 2-1.

Figure 2-1 CFG80211 configuration

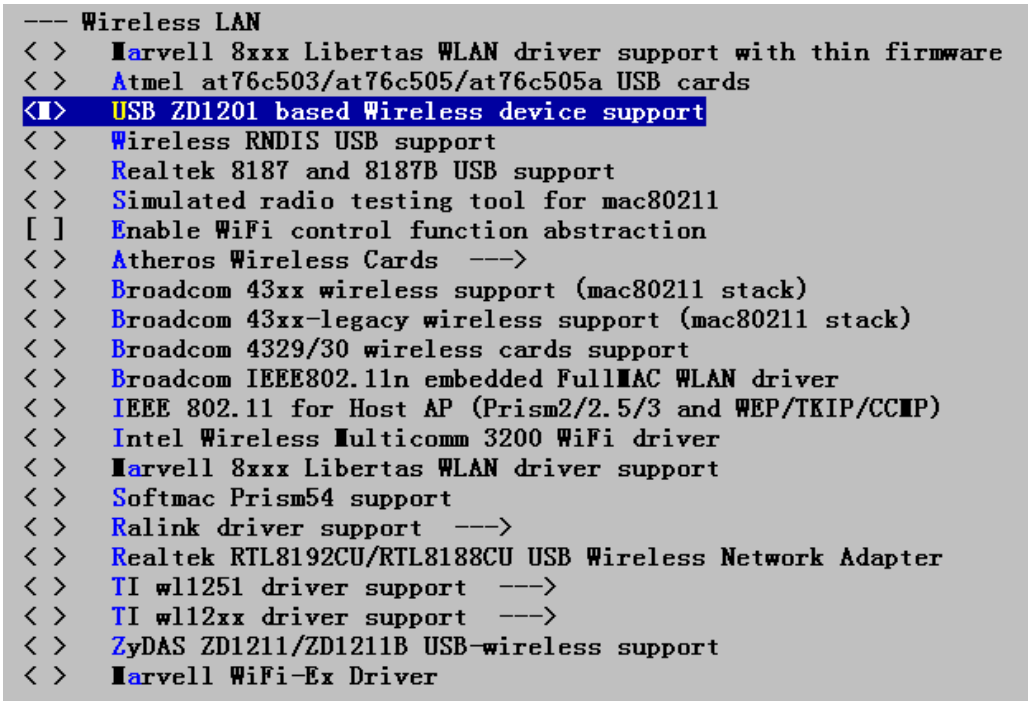
```
--- Wireless
<M>   cfg80211 - wireless configuration API
[ ]     nl80211 testmode command (NEW)
[ ]     enable developer warnings (NEW)
[ ]     cfg80211 regulatory debugging (NEW)
[*]     enable powersave by default (NEW)
[ ]     cfg80211 wireless extensions compatibility (NEW)
<M>   Generic IEEE 802.11 Networking Stack (mac80211)
      Default rate control algorithm (Minstrel)  --->
[ ]   Enable mac80211 mesh networking (pre-802.11s) support (NEW)
[ ]   Trace all mac80211 debug messages (NEW)
[ ]   Select mac80211 debugging features (NEW)  ----
```

2.1.2 Configuring Wireless Extension

- Wireless extension (WEXT) is the standard interface for the Wi-Fi drivers in the kernel and user-mode processes. The debugging tools iwconfig, iwlist, and iwpriv need to use this interface. If this interface is not configured, errors occur when some drivers are compiled.
- Because there is no separate configuration item for the WEXT in the kernel configuration, you can configure the WEXT only by configuring the item that is

dependent on the WEXT. To be specific, choose **Device Drivers** > **Network device support** > **Wireless LAN**, and set **USB ZD1201 based Wireless device support** to **M**, as shown in Figure 2-2.

Figure 2-2 Configuring wireless extension



If the configuration item cannot be found, configure the USB first.

2.1.3 Configuring the USB and SDIO

For details about the USB and SDIO operations, see the *Peripheral Driver Operation Guide*.

The I/O voltage of the SDIO can be 1.8 V or 3.3 V. Ensure that the I/O voltage of the Wi-Fi module is the same as that of the SDIO of the master chip.

2.1.4 Configuring the SDIO Interrupt

The SDIO interrupt is disabled by default in the kernel. The SDIO interrupt needs to be enabled if the Wi-Fi driver used does not support out-of-band (OOB) management. For Hi3516E V200, enable the SDIO interrupt by adding **cap-sdio-irq** to the attribute of the SDIO port which is connected to the Wi-Fi in the **arch/arm/boot/dts/hi3516ev200.dtsi** configuration file.

NOTICE

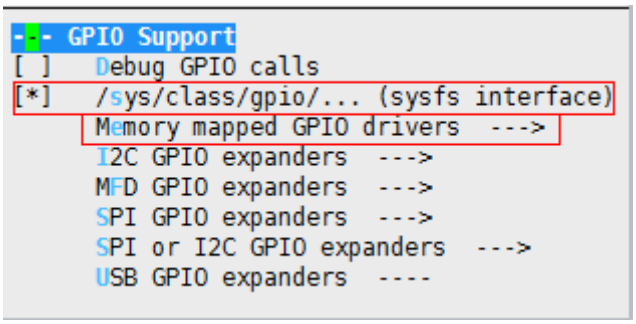
After the kernel is configured and compiled, the Wi-Fi driver must be recompiled based on the new kernel; otherwise, the pointer is null or the kernel symbol cannot be found when the driver is running.

2.1.5 Configuring the GPIO

Step 1 Open the GPIO configuration window.

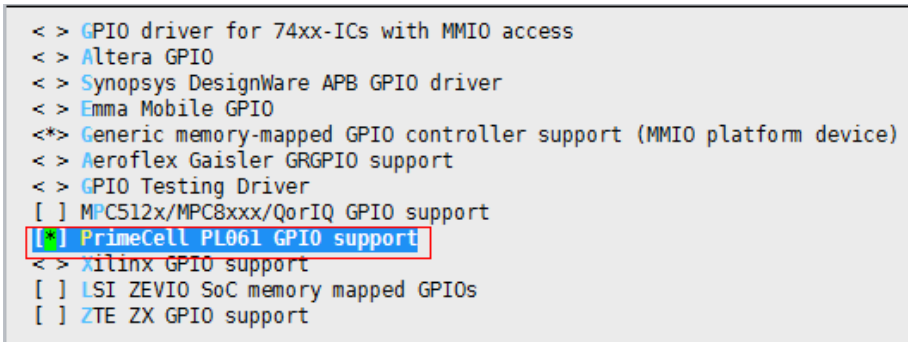
Step 2 Go to **Device Drivers**, enable **GPIO Support**, and then go to **GPIO Support**, as shown in Figure 2-3.

Figure 2-3 Configuring the GPIO A



Step 3 Go to the memory-mapped GPIO drivers and configure the GPIO as shown in Figure 2-4.

Figure 2-4 Configuring the GPIO B



After the kernel is configured, compile the kernel and burn it to the board.

----End

2.2 Boot Configuration

The multiplexing and drive current of the SDIO and USB pins must be configured in the boot table. For details about the configuration method, see the boot table configuration document. After the configuration is complete, compile and generate a new boot file and burn it to the board. The configuration table is stored in the **tools/pc/uboot_tools** directory in the **package/osdrv.tgz** file of the SDK.

3 Porting Drivers

3.1 Porting the RTL8189FTV Driver

Step 1 Obtain the driver from the Realtek or module vendor, for example, **rtl8189FS_linux_v4.3.24.11_26052.20180108.tar.gz**.

Step 2 Decompress the package to obtain the **rtl8189FS_linux_v4.3.24.11_26052.20180108** code directory.

```
tar zxvf rtl8189FS_linux_v4.3.24.11_26052.20180108.tar.gz
```

Step 3 Modify the **rtl8189FS_linux_v4.3.24.11_26052.20180108/include/autoconf.h** file.

Remove the driver printing and comment out the following line:

```
    //#define CONFIG_DEBUG /* DBG_871X, etc... */
```

If the printing information is required during debugging, you do not need to modify the line.

Step 4 Modify the file **rtl8189FS_linux_v4.3.24.11_26052.20180108/Makefile**.

Perform the following setting:

```
CONFIG_REDUCE_TX_CPU_LOADING = y
```

Step 5 Modify the file **rtl8189FS_linux_v4.3.24.11_26052.20180108/hal/rtl8188f/sdio/rtl8189fs_xmit.c**.

Change both **rtw_msleep_os(1)** in the rtl8188fs_xmit_handler function to **rtw_usleep_os(1)** to reduce the CPU usage.

```
    ret = xmit_xmitframes(padapter, pxmitpriv);
    if (ret == -2) {
        /* here sleep 1ms will cause big TP loss of TX */
        /* from 50+ to 40+ */
        if (padapter->registrypriv.wifi_spec)
            rtw_msleep_os(1);
        else
#ifdef CONFIG_REDUCE_TX_CPU_LOADING
        //  rtw_msleep_os(1);
            rtw_usleep_os(1);
```



```
#else
            rtw_yield_os();
#endif
        goto next;
    }

    _enter_critical_bh(&pxmitpriv->lock, &irql);
    ret = rtw_txframes_pending(padapter);
    _exit_critical_bh(&pxmitpriv->lock, &irql);
    if (ret == 1) {
#ifdef CONFIG_REDUCE_TX_CPU_LOADING
        //rtw_msleep_os(1);
#endif
        goto next;
    }
```

Step 6 Compile the code.

Run the following command in the directory where **rtl8189FS_linux_v4.3.24.11_26052.20180108** is located:

```
make -C rtl8189FS_linux_v4.3.24.11_26052.20180108 ARCH=arm
CROSS_COMPILE=arm-himix100-linux-  KSRC=/home/kernel/linux-4.9.y
```

In the preceding command, replace **arm-himix100-linux-** with the compiler used for compiling the kernel, and replace **/home/kernel/linux-4.9.y** with the directory of the compiled kernel.

After the compilation is complete, the driver **8189fs.ko** is generated in the **rtl8189FS_linux_v4.3.24.11_26052.20180108** directory.

----End

3.2 Porting the RTL8188FTV Driver

Step 1 Obtain the driver from the Realtek or module vendor, for example, **rtl8188FU_linux_v5.2.11.1_22924.20170703.tar.gz**.

Step 2 Decompress the package to obtain the **rtl8188FU_linux_v5.2.11.1_22924.20170703** code directory.

```
tar zxvf rtl8188FU_linux_v5.2.11.1_22924.20170703.tar.gz
```

Step 3 Modify the file **rtl8188FU_linux_v5.2.11.1_22924.20170703/include/autoconf.h**.

1. Enable **CONFIG_IOCTL_CFG80211**.

   Comment out the following line:

   ```
   #define CONFIG_IOCTL_CFG80211
   ```

2. Enable **RTW_USE_CFG80211_STA_EVENT**.

   Comment out the following line:


```
#define RTW_USE_CFG80211_STA_EVENT
```

Step 4 Modify the file **rtl8188FU_linux_v5.2.11.1_22924.20170703/Makefile**.

Disable the driver printing function as follows:

```
CONFIG_RTW_DEBUG = n
```

Step 5 Compile the driver.

Run the following command in the directory where **rtl8188FU_linux_v5.2.11.1_22924.20170703** is located:

```
make -C rtl8188FU_linux_v5.2.11.1_22924.20170703 ARCH=arm
CROSS_COMPILE=arm-himix100-linux-  KSRC=/home/kernel/linux-4.9.y
```

In the preceding command, replace **arm-himix100-linux-** with the compiler used for compiling the kernel, and replace **/home/kernel/linux-4.9.y** with the directory of the compiled kernel.

After the compilation is complete, the **8188fu.ko** driver is generated in the **rtl8188FU_linux_v5.2.11.1_22924.20170703** directory.

----End

4 Wi-Fi Tools

When perform Wi-Fi operations, the wpa_supplicant, libnl, iwconfig, iwlist, iwpriv, and iperf tools are required. These tools are open source software and can be downloaded from the Internet.

Table 4-1 Paths for obtaining the tools

Tool Name	Download Path	Compilation Result
wpa_supplicant	http://w1.fi/releases	wpa_supplicant, wpa_cli, and hostapd
libnl	http://www.linuxfromscratch.org/blfs/view/svn/basicnet/libnl.html	libnl-genl.so.2.0.0 and libnl.so.2.0.0
Wireless tools	http://www.linuxfromscratch.org/blfs/view/svn/basicnet/wireless_tools.html	iwconfig, iwlist, and iwpriv
iperf	https://iperf.fr/iperf-download.php	iperf

For details about how to compile the tools, see the description in the website. The compilation of the wpa_supplicant depends on the **libnl** library. You need to compile the **libnl** library before compiling the wpa_supplicant.

5 Basic Operations

5.1 Loading Files

5.1.1 Loading Driver Files

Copy the **net/wireless/cfg80211.ko** file in the kernel and the compiled driver to the board.

You can copy them to any directory on the board, for example, **/kmod**.

5.1.2 Loading Tools

- Copy the **libnl-genl.so.2.0.0** and **libnl.so.2.0.0** files to the **/lib** directory of the board, and create the following soft links for the two files:

```
ln –s libnl-genl.so.2.0.0 libnl-genl.so.2
ln –s libnl.so.2.0.0 libnl.so.2
```

- Copy iwconfig, iwlist, iwpriv, and iperf to the **/sbin** directory of the board. These are debugging tools. You do not have to copy them.
- In STA mode, you need to copy wpa_supplicant and wpa_cli to the **/sbin** directory of the board.
- In AP mode, you need to copy hostapd to the **/sbin** directory of the board.

After the tool is copied to the board, you need to modify the executable permission of the tool. For example:

```
chmod a+x wpa_supplicant
```

5.1.3 wpa_supplicant.conf File

wpa_supplicant.conf is a configuration file required when the wpa_supplicant process is started. You can create **wpa_supplicant.conf** on the board and store it in any directory (such as **/etc/Wireless**). The file content is as follows:

```
ctrl_interface=/var/wpa_supplicant
```

5.1.4 hostapd.conf File

hostapd.conf is a configuration file required when the hostapd process is started. You can create **hostapd.conf** on the board and store it in any directory (such as **/etc/Wireless**). For

details about the file content, see section 5.4.2 "Configuring and Enabling the SoftAP by Using the hostapd Process."

5.1.5 udhcpd.conf File

udhcpd.conf is a configuration file required by the DHCP server in SoftAP mode. You can create the **udhcpd.conf** file by referring to online materials, and then copy it to any directory of the board (such as **/etc/Wireless**).

5.2 Detecting Wi-Fi Devices

Before using the SDIO Wi-Fi, you need to detect SDIO devices. Before using the USB Wi-Fi, you need to detect USB devices. After the SDIO is configured in boot, the Realtek SDIO Wi-Fi can detect SDIO devices during startup. If the information "mmc1: new SDIO card at address 0001" is displayed, the card detection is successful.

After startup, you can run the **cat /proc/mci/mci_info** command to check whether an SDIO device is detected.

Figure 5-1 Viewing SDIO devices

```
~ #
~ # cat /proc/mci/mci_info
MCI0: unplugged_disconnected
MCI1: pluged_connected
        Type: SDIO card Mode: HS
        Speed Class: Class 0
        Uhs Speed Grade: Less than 10MB/sec(0h)
        Host work clock: 50MHz
        Card support clock: 50MHz
        Card work clock: 50MHz
        Card error count: 0
MCI2: invalid
~ #
~ #
```

The USB Wi-Fi can detect USB devices upon startup. Run the **lsusb** command to check whether any USB device is detected.

Figure 5-2 Viewing USB devices

```
~ # lsusb
Bus 001 Device 001: ID 1d6b:0002
Bus 001 Device 002: ID 148f:7601
Bus 002 Device 001: ID 1d6b:0003
~ #
```

If the USB ID of any Wi-Fi device is displayed, the detection is successful. For details about the USB ID, contact the Wi-Fi vendor.

5.3 Operation Examples for the STA Mode

5.3.1 Loading Drivers

Perform the following steps:

Step 1 Load driver files by running the following commands based on the Wi-Fi drivers used:

- rtl8189ftv
  insmod cfg80211.ko
  insmod 8189fs.ko
- rtl8188ftv
  insmod cfg80211.ko
  insmod 8188fu.ko

Step 2 Check whether the driver is successfully loaded by running the shell command **iwconfig**.

If the wlan0 network port exists, the driver is successfully initialized, and the Wi-Fi device is available.

Figure 5-3 Execution result of iwconfig

```
# iwconfig
lo        no wireless extensions.

eth0      no wireless extensions.

wlan0     unassociated  Nickname:"<WIFI@REALTEK>"
          Mode:Auto  Frequency=2.412 GHz  Access Point: Not-Associated
          Sensitivity:0/0
          Retry:off   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:off
          Link Quality:0  Signal level:0  Noise level:0
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

Step 3 Enable the Wi-Fi network port by running the shell command **ifconfig wlan0 up**.

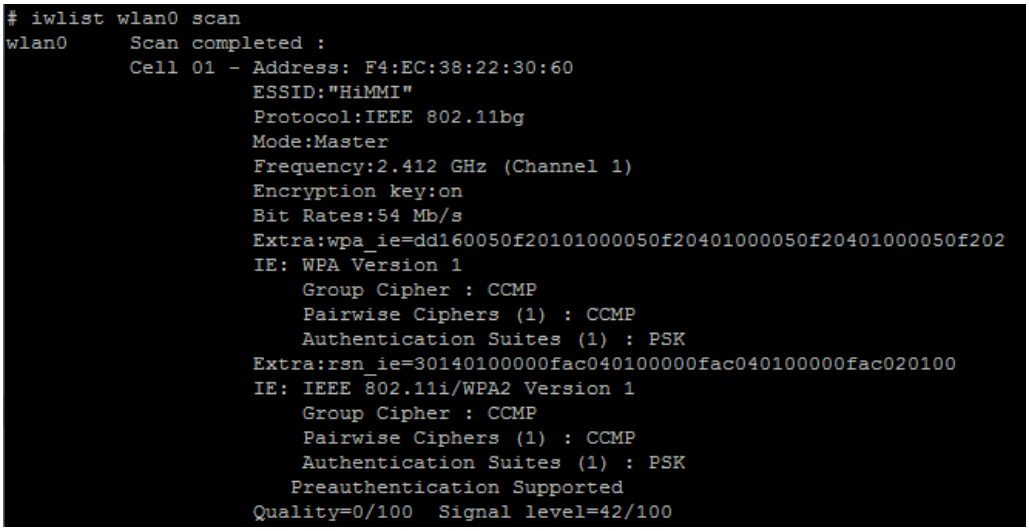
After the preceding command is executed, the Wi-Fi is available, and you can perform the scan and connect operations.

-----End

5.3.2 Scanning for APs

To scan for APs, run the shell command **iwlist wlan0 scan**.

Figure 5-4 AP scanning result



The detected APs are displayed in the format of Cell *xx*, and each AP corresponds to a Cell *xx*.

The AP information includes:

- **Address**: MAC address
- **ESSID**: AP name, that is, SSID
- **Protocol**: IEEE80211 protocol, 11b/g/n
- **Frequency**: frequency
- **Encryption key** (authentication encryption information): WEP, WPA-PSK, WPA2-PSK, WPA, and WPA2
- **Quality**: signal quality. This data is sometimes inaccurate and can be ignored.
- **Signal Level**: signal strength. A larger value indicates greater signal strength. The display mode of the signal level varies with the Wi-Fi driver, for example, *xx*/100, or *xx* dBm.

The display format of the preceding information varies with the Wi-Fi driver.

NOTICE

When you scan for APs by running the **iwlist** command, the scan result is returned not necessarily after all frequencies are scanned. Therefore, some APs cannot be detected, especially for MT7601U. MT7601U scans each frequency for a long time period, and therefore only APs at one or two frequencies can be detected during the first scan.

5.3.3 Connecting to an AP

The wpa_supplicant process is used to connect the Wi-Fi device to an AP. wpa_supplicant is an open-source code which is used on Linux and Android to implement the Wi-Fi connection process. It includes protocols such as WEP, WPA/WPA2, WPA-PSK/WPA2-PSK, WAPI, WPS, P2P, and EAP.

Step 1 Start the wpa_supplicant process by running the following shell command:

wpa_supplicant –iwlan0 –Dnl80211 –c/etc/Wireless/wpa_supplicant.conf&

- **-iwlan0** indicates that the wlan0 network port is used.
- **-Dnl80211** indicates that the cfg80211 interface is used (libnl for user-mode interfaces and cfg80211 for kernel-mode interfaces).
- /**etc/Wireless/wpa_supplicant.conf** indicates the configuration file of wpa_supplicant. Ensure that the file exists.

After the command is executed, run the **ps** command to check whether the wpa_supplicant process exists. If yes, it works properly. If no, raise the wpa_supplicant print level and find out the cause from the logs.

wpa_supplicant –iwlan0 –Dnl80211 –c/etc/Wireless/wpa_supplicant.conf –ddd &

Step 2 Start the wpa_cli process by running the following shell command:

wpa_cli –iwlan0 –p/var/wpa_supplicant

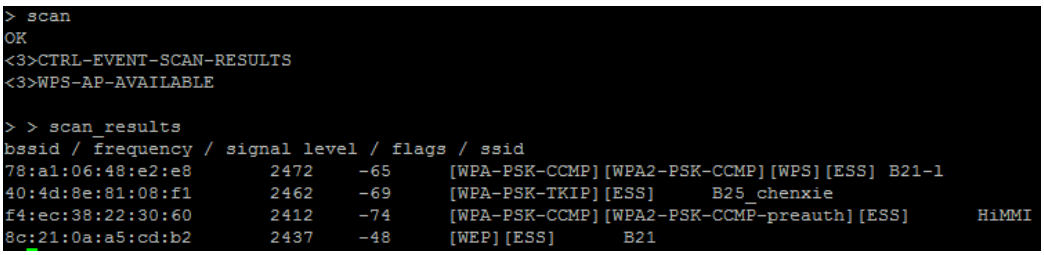
If the preceding command is successfully executed, the symbol ">" is displayed.

If "Could not connect to wpa_supplicant -re-trying" is displayed, the socket connection cannot be set up between wpa_cli and wpa_supplicant. In this case, check whether the wpa_supplicant process, **/var/wpa_supplicant/wlan0**, and **ctrl_interface=/var/wpa_supplicant** in the **wpa_supplicant.conf** file exists.

Step 3 Scan for APs.

Run the **scan** command after the symbol ">", and run **scan_results** after **CTRL-EVENT-SCAN-RESULTS** is received. The scan result is displayed.

Figure 5-5 AP scanning result of wpa_cli



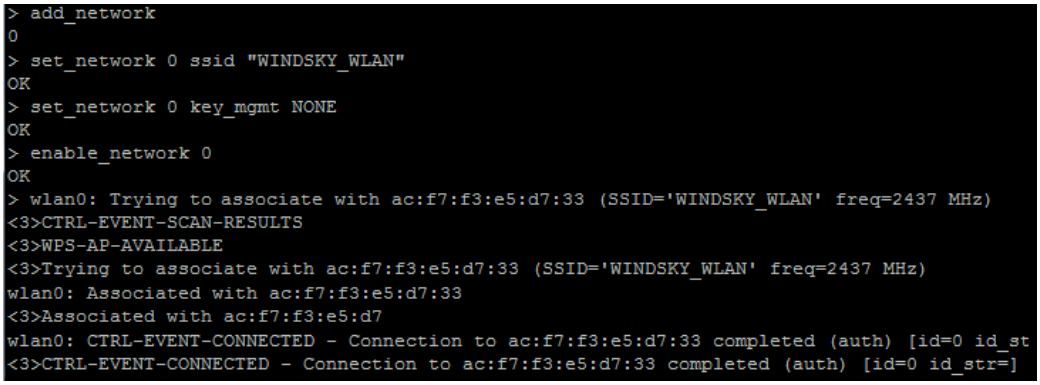
Step 4 Connect an AP in any of the following ways as required:

- Connect an AP in open mode.

1. Run **add_network** after the symbol ">". Assume that the returned network ID is 0.
2. Configure the network SSID by running **set_network 0 ssid** (SSID of the AP).
3. Configure the network encryption mode by running **set_network 0 key_mgmt NONE**.
4. Enable the network by running **select_network 0**.

   If **CTRL-EVENT-CONNECTED** is received, the AP is successfully connected.

Figure 5-6 Connecting to an AP



- Connect an AP in WPA-PSK/WPA2-PSK mode.

1. Run **add_network** after the symbol ">". Assume that the returned network ID is 0.
2. Configure the network SSID by running **set_network 0 ssid** (SSID of the AP).
3. Configure the network encryption mode by running **set_network 0 psk** "*AP password*".
4. Enable the network by running **select_network 0**.
5. If **CTRL-EVENT-CONNECTED** is received, the AP is successfully connected.

Step 5 Obtain the IP address.

Enter **q** to exit wpa_cli, and run the shell command **udhcpc –i wlan0**.

After the IP address is configured, run the **ping** command to check whether it can be pinged.

----End

5.3.4 Uninstalling Drivers

Uninstall drivers by running the following commands based on the used Wi-Fi drivers:

- rtl8189ftv
  ifconfig wlan0 down
  rmmod 8189fs.ko
  rmmod cfg80211.ko
- rtl8188ftv
  ifconfig wlan0 down
  rmmod 8188fu.ko
  rmmod cfg80211.ko

5.4 Operation Examples for the SoftAP Mode

5.4.1 Checking Wi-Fi Devices and Loading Drivers

The RTL8189FTV/RTL8188FTV driver is loaded the same way as in STA mode.

5.4.2 Configuring and Enabling the SoftAP by Using the hostapd Process

The hostapd process is used to configure the SoftAP. The hostapd process is similar to wpa_supplicant. It contains various authentication protocols and connection processes of the AP end, whereas wpa_supplicant belongs to the STA end.

Step 1 Modify **hostapd.conf**.

The hostapd process requires the **hostapd.conf** configuration file. You can set the SSID, frequency, and encryption mode in the configuration file. The following shows the examples of configuration files:

- OPEN

```
interface=wlan0
driver=nl80211
ctrl_interface=/var/hostapd
ssid=HisiAP
channel=6
hw_mode=g
ieee80211n=1
ht_capab=[SHORT-GI-20][SHORT-GI-40][HT40-]
```

- WPA2-PSK

```
interface=wlan0
driver=nl80211
ctrl_interface=/var/hostapd
ssid=HisiAP
channel=6
hw_mode=g
ieee80211n=1
ht_capab=[SHORT-GI-20][SHORT-GI-40][HT40-]
wpa=3
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP CCMP
wpa_passphrase=12345678
```

The hostapd code is open source. For details about parameters in the configuration file, search for network resources.

ht_capab can be configured to support or not support the 40 MHz bandwidth. When **[SHORT-GI-40][HT40-]** or **[SHORT-GI-40][HT40+]** is added after the **ht_capab=[SHORT-GI-20]**, the 40 MHz bandwidth is supported. When the value of the **channel** is less than 6, **[SHORT-GI-40][HT40+]** is added after the **ht_capab=[SHORT-GI-20]**. When the value of the **channel** is greater than or equal to 6, **[SHORT-GI-40][HT40-]** is added after the **ht_capab=[SHORT-GI-20]**.

Step 2 Start the hostapd process by running the following shell command:

hostapd /etc/Wireless/hostapd.conf &

After the command is executed, run the **ps** command to check whether the hostapd process exists. If yes, it works properly, and the SoftAP can be detected by the STA device. If no, raise the hostapd print level and find out the cause from the logs. For example:

hostapd –ddd /etc/Wireless/hostapd.conf &

----End

5.4.3 Enabling udhcpd

Enable udhcpd by running the following shell commands:

ifconfig wlan0 192.168.1.1

udhcpd –fS /etc/Wireless/udhcpd.conf

Ensure that **/etc/Wireless/udhcpd.conf** exists, and the configured network segment is 192.168.1.*x*. After the preceding commands are executed, the SoftAP can be scanned and connected by using the STA device. If the SoftAP is successfully connected and network gateway can be pinged, the AP is successfully configured.

5.4.4 Uninstalling Drivers

The driver uninstalling process is the same as in STA mode.

5.5 Configuring the Country or Region

The frequency range varies according to the country or region. For example, for the 2.4 GHz frequency band, the USA supports channels 1 to 11, China and Europe support channels 1 to 13, and Japan supports channels 1 to 14. The situation is similar for the 5 GHz frequency band. The Wi-Fi device needs to be configured based on the country or region in which the device is to be launched.

The configuration method varies according to the Wi-Fi device. For example, to use RTL8188ftv in the USA, add the parameter **rtw_channel_plan=0x22** as follows when loading the driver:

insmod 8188fu.ko rtw_channel_plan=0x22

This document does not provide the configurations of all Wi-Fi drivers for various countries or regions. For details, consult module vendors or Wi-Fi driver vendors.

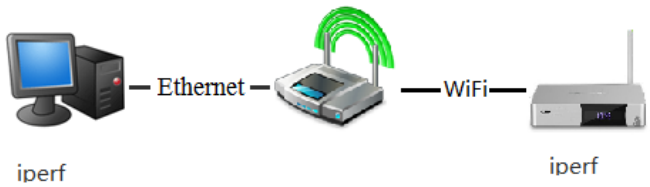
6 Tests

6.1 Throughput Test

The throughput tests show the Wi-Fi performance and are widely used and proved by chip vendors, module vendors, and Wi-Fi device vendors. The most frequently used throughput testing tool is iperf.

The test environment is that a PC connects to the AP with a cable, a board connects to the AP by using the Wi-Fi, and the PC and the board can ping each other successfully. The iperf tool is installed on both the PC and the board. Assume that the IP address for the PC is 192.168.1.100, and that for the board is 192.168.1.101.

Figure 6-1 Networking for the throughput test



6.1.1 TCP TX Throughput Test

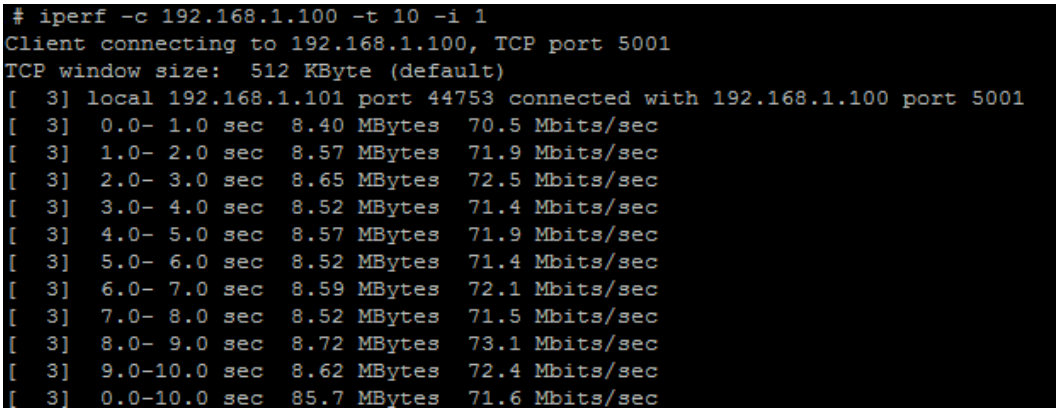
To test the (transmit) TX throughput, perform the following steps:

Step 1 Run the **iperf –s** command to o to the directory of the iperf tool on the PC.

Step 2 Go to the directory of the iperf tool using shell on the board by the following command:

iperf –c 192.168.1.100 –t 10 –i 1

Figure 6-2 TX throughput test example



iperf -s indicates starting the server. **iperf -c 192.168.1.100** indicates starting the client and connecting to 192.168.1.100. **-t 10** indicates 10-second testing period. **-i 1** indicates that the result will be printed once every 1 second.

The displayed test result "0.0-10.0 sec 85.7 MBytes 71.6 Mbit/sec" in the last row indicates that the average throughput in 10 seconds is 71.6 Mbit/s.

----End

6.1.2 TCP RX Throughput Test

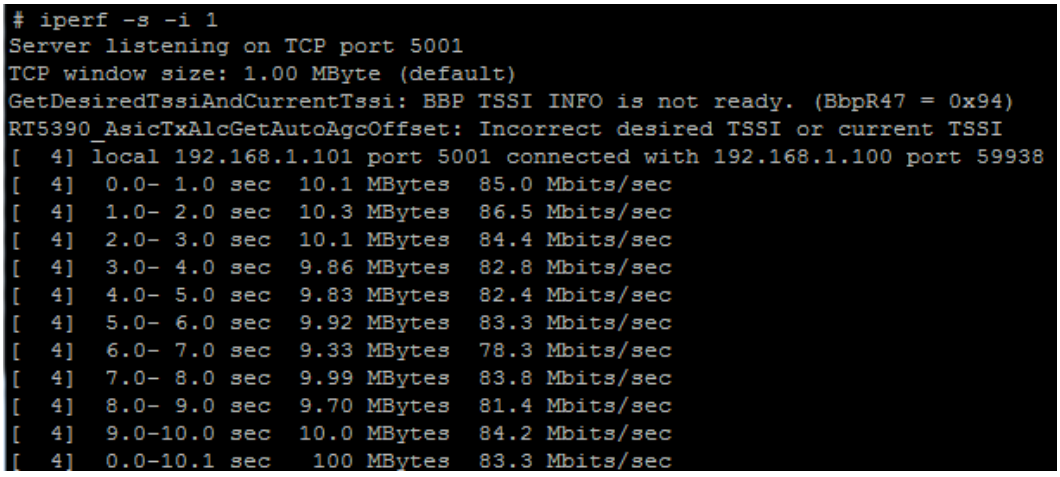
To test the receive (RX) throughput, perform the following steps:

Step 1 Run the **iperf –s** command to go to the directory of the iperf tool using shell on the board.

Step 2 Go to the directory of the iperf tool on the PC by the following command:

iperf –c 192.168.1.101 –t 10 –i 1 –w 1M

Figure 6-3 RX throughput test example



The iperf tool can also be used to perform the User Datagram Protocol (UDP) test. The speed of a single UDP thread is limited on some PCs, and therefore multiple threads are required.

The throughput tests for the SoftAP are similar.

NOTICE

The speed of some PCs is affected by the installed software. Ensure that the PC speed is not affected. The 802.11n protocol cannot be used in WEP safe mode, and therefore the speed is low, typically over 20 Mbit/s.

----End

6.1.3 UDP TX Throughput Test

To test the TX throughput, perform the following steps:

Step 1 Run the **iperf –s -u –l 32k** command to go to the directory of the iperf tool on the PC.

Step 2 Go to the directory of the iperf tool using shell on the board by the following command:

iperf –c 192.168.1.100 –u –t 10 –i 1 –l 32k –b 100M

----End

6.1.4 UDP RX Throughput Test

To test the RX throughput, perform the following steps:

Step 1 Run the **iperf –s -u** command to go to the directory of the iperf tool using shell on the board.

Step 2 Go to the directory of the iperf tool on the PC by the following command:

iperf –c 192.168.1.101 –u –t 10 –i 1 –l 32k –b 100M

----End

6.2 RF Specification Test

The throughput tests reflect the Wi-Fi performance and are mandatory during product development. Some companies also conduct the radio frequency (RF) specifications test, which can accurately verify whether the Wi-Fi RF meets specifications. The RF specifications test is mandatory during module production. Therefore, if a Wi-Fi module is used, this test is optional. However, the Wi-Fi RF performance may be affected due to board interference and unclean GND traces during hardware design. Therefore, you are advised to conduct this test if possible.

The RF specifications include the following: receive sensitivity, transmit power, error tolerance of the transmit carrier frequency, packet loss rate, error vector magnitude (EVM), transmit adjacent channel power ratio (ACPR), receive ACPR, and so on.

The test tools include spectrum analyzer, power measurer, network analyzer and so on.

For details about the test methods, see the instructions of the test tools.

