

# Realtek Wi-Fi SDK for Android N 7.0

## ver. 1.0.1

### Contents

Release History .....	2
Introduction .....	3
1. Copy Necessary Files into SDK .....	4
2. Platform Related Files .....	4
2.1. BoardConfig.mk .....	4
2.2. init.xxx.rc .....	6
2.3. Others .....	8
3. System Resource Configurations .....	9
4. wpa_supplicant_8 .....	11
5. Driver Configurations for Android 7.0 .....	11
5.1. CONFIG_RTW_ANDROID .....	13
6. FAQ .....	14
6.1. Wi-Fi (STA mode) .....	14
6.1.1. Why Wi-Fi can't enable? .....	14
6.2. Portable Wi-Fi hotspot (AP mode) .....	15
6.2.1. Why Portable Wi-Fi hotspot can't enable? .....	15
6.3. Wi-Fi Direct (P2P mode) .....	15
6.3.1. There is no Wi-Fi Direct UI shown? .....	15
6.3.2. Wi-Fi Direct can't scan any peer? .....	15

## Release History

1.0.0	2016/10/14	1. First formal release
1.0.1	2019/8/21	1. CONFIG_RTW_ANDROID in Makefile

Realtek

## SDK packages

- hardware/realtek/\*

Folder to store config files, private code from Realtek.

## Introduction

This document provides a simple guide to help engineers to apply Realtek Wi-Fi solution onto their Android N 7.0 system. For now, we have supported the following two scenarios:

- **STA/AP** – Switch between STA mode and AP mode
- **(STA+P2P)/AP** – Switch between STA+P2P(Wi-Fi Direct) concurrent mode and AP mode

To port Realtek Wi-Fi driver onto Android 7.0 platform, you can go through the following guide with reference codes within our driver package's [realtek\\_wifi\\_SDK\\_for\\_android\\_N\\_7.0\\_20161024.zip](#).

Because Android's SDK may differ from platform to platform, our reference codes may not be applied on every platform without modifications. You should check if our reference code is suitable for you to use.

In this document, ANDROID\_SDK is the path of top folder of the target Android SDK; this term is used in the following paragraphs.

## 1. Copy Necessary Files into SDK

After unzipping `realtek_wifi_SDK_for_android_N_7.0_20161024.zip`, copy the following folder into `ANDROID_SDK/hardware/folder`:

- `hardware/realtek`

## 2. Platform Related Files

### 2.1. BoardConfig.mk

To apply Realtek Wi-Fi solution onto your Android 7.0 system, you need to define some compile-time variables in `BoardConfig.mk` of your platform. In general, the `BoardConfig.mk` file is located in:

`ANDROID_SDK /device/<soc_vendor_name>/<board_name>/`

Take TI panda board for example:

`ANDROID_SDK /device/ ti/panda/ BoardConfig.mk`

Please define the following compile-time variables below:

```

BOARD_WIFI_VENDOR := realtek
ifeq ($(BOARD_WIFI_VENDOR), realtek)
    WPA_SUPPLICANT_VERSION := VER_0_8_X
    BOARD_WPA_SUPPLICANT_DRIVER := NL80211
    BOARD_WPA_SUPPLICANT_PRIVATE_LIB := lib_driver_cmd_rtl
    BOARD_HOSTAPD_DRIVER      := NL80211
    BOARD_HOSTAPD_PRIVATE_LIB := lib_driver_cmd_rtl

    BOARD_WLAN_DEVICE := rtl8822bu
    #BOARD_WLAN_DEVICE := rtl8192cu
    #BOARD_WLAN_DEVICE := rtl8192du
    #BOARD_WLAN_DEVICE := rtl8192ce
    #BOARD_WLAN_DEVICE := rtl8192de
    #BOARD_WLAN_DEVICE := rtl8723as
    #BOARD_WLAN_DEVICE := rtl8723au
    #BOARD_WLAN_DEVICE := rtl8189es
    #BOARD_WLAN_DEVICE := rtl8723bs
    #BOARD_WLAN_DEVICE := rtl8723bu

    WIFI_DRIVER_MODULE_NAME   := "wlan"
    WIFI_DRIVER_MODULE_PATH   := "/vendor/modules/wlan.ko"
    WIFI_DRIVER_MODULE_ARG    := "ifname=wlan0 if2name=p2p0"

    WIFI_FIRMWARE_LOADER     := "rtw_fwloader"
    WIFI_DRIVER_FW_PATH_STA  := "STA"
    WIFI_DRIVER_FW_PATH_AP   := "AP"
    WIFI_DRIVER_FW_PATH_P2P  := "P2P"
    WIFI_DRIVER_FW_PATH_PARAM := "/dev/null"
endif

```

- **BOARD\_WIFI\_VENDOR := realtek**

To distinguish the platform Wi-Fi device from products of other vendors, we define variable BOARD\_WIFI\_VENDOR as realtek. This is for compile-time choices to be applied for Realtek Wi-Fi solutions.

- **WPA\_SUPPLICANT\_VERSION := VER\_0\_8\_X**

For Android L, please set WPA\_SUPPLICANT\_VERSION as VER\_0\_8\_X to

use wpa\_supplicant\_8.

- **BOARD\_WPA\_SUPPLICANT\_DRIVER := NL80211**
- **BOARD\_WPA\_SUPPLICANT\_PRIVATE\_LIB := lib\_driver\_cmd\_rtl**
- **BOARD\_HOSTAPD\_DRIVER := NL80211**
- **BOARD\_HOSTAPD\_PRIVATE\_LIB := lib\_driver\_cmd\_rtl**

We use NL80211 as the driver interface for wpa\_supplicant and hostapd to communicate with driver and provide lib\_driver\_cmd\_rtl as the private library.

- **BOARD\_WLAN\_DEVICE**

Realtek provide a variety of Wi-Fi solutions to choose. For now, BOARD\_WLAN\_DEVICE is not used for any purpose but we suggest setting this variable for your Wi-Fi solution you used.

- **WIFI\_DRIVER\_MODULE\_NAME**
- **WIFI\_DRIVER\_MODULE\_PATH**
- **WIFI\_DRIVER\_MODULE\_ARG**

These three variables will be used in libhardware\_legacy (wifi.c) to do insmod and rmmod. The value of WIFI\_DRIVER\_MODULE\_NAME should match the value of MODULE\_NAME specified in our driver's Makefile at compile-time. Please refer to "Platform Setting Section in Detail" of:

document/Quick\_Start\_Guide\_for\_Driver\_Compilation\_and\_Installation.pdf

- **WIFI\_FIRMWARE\_LOADER := "rtw\_fwloader"**

This variable will be used in libhardware\_legacy (wifi.c) as the name of Wi-Fi firmware loader, which will be executed after driver's insmod and before the executing of wpa\_supplicant and hostapd. Setting it to "rtw\_fwloader" for calling service rtw\_fwloader which provided by Realtek.

- **WIFI\_DRIVER\_FW\_PATH\_STA := "STA"**
- **WIFI\_DRIVER\_FW\_PATH\_AP := "AP"**
- **WIFI\_DRIVER\_FW\_PATH\_P2P := "P2P"**
- **WIFI\_DRIVER\_FW\_PATH\_PARAM := "/dev/null"**

Realtek driver has FW embedded inside, and will automatically load FW at NIC initialization process. Setting these four variables is just to fit the requirement of the libhardware\_legacy (wifi.c).

## 2.2. init.xxx.rc

For Wi-Fi to operate properly, we need some daemons to be defined as service inside init.xxx.rc. In general, the init.xxx.rc file is located in:

ANDROID\_SDK/device/<soc\_vendor\_name>/<board\_name>/

Take TI panda board for example:

ANDROID\_SDK/device/ti/panda/init.omap4pandaboard.rc.

Please add the service definitions below:

- **rtw\_fwloader**

```
service rtw_fwloader /system/bin/rtw_fwloader
    class main
    disabled
    oneshot
```

- **wpa\_supplicant**

```
service p2p_supplicant /system/bin/wpa_supplicant \
    -ip2p0 -Dnl80211 -c/data/misc/wifi/p2p_supplicant.conf \
    -e/data/misc/wifi/entropy.bin -N \
    -iwlan0 -Dnl80211 -c/data/misc/wifi/wpa_supplicant.conf \
    -O/data/misc/wifi/sockets \
    -g@android:wpa_wlan0
    class main
    socket wpa_wlan0 dgram 660 wifi wifi
    disabled
    oneshot

service wpa_supplicant /system/bin/wpa_supplicant \
    -iwlan0 -Dnl80211 -c/data/misc/wifi/wpa_supplicant.conf \
    -O/data/misc/wifi/sockets \
    -e/data/misc/wifi/entropy.bin \
    -g@android:wpa_wlan0
    class main
    socket wpa_wlan0 dgram 660 wifi wifi
    disabled
    oneshot
```

- **dhcpcd**

```
service dhcpcd_wlan0 /system/bin/dhcpcd -aABDKL
    class main
    disabled
    oneshot

service dhcpcd_p2p /system/bin/dhcpcd -aABKL
    class main
    disabled
    oneshot

service iprenw_wlan0 /system/bin/dhcpcd -n
    class main
    disabled
    oneshot

service iprenw_p2p /system/bin/dhcpcd -n
    class main
    disabled
    oneshot
```

### 2.3. Others

For topics mentioned here, you can add the following code segments in any .mk file which your platform will use. Take TI panda board for example:

ANDROID\_SDK /device/ ti/panda/device.mk.

- **Add android.hardware.wifi.xml**

To claim Wi-Fi support for your device, please add the rule in the PRODUCT\_COPY\_FILES variable to copy the permission definition file of Wi-Fi to the /system/etc/permissions/ folder of your system image.

```
PRODUCT_COPY_FILES += \
frameworks/native/data/etc/android.hardware.wifi.xml:system/etc/permissions/android.hardware.
wifi.xml
```

- **Add android.hardware.wifi.direct.xml**

To claim Wi-Fi Direct (P2P) support for your device, please add the rule in the



PRODUCT\_COPY\_FILES variable to copy the permission definition file of Wi-Fi Direct to the /system/etc/permissions/ folder of your system image.

```
PRODUCT_COPY_FILES += \
frameworks/native/data/etc/android.hardware.wifi.direct.xml:system/etc/permissions/android.hardware.wifi.direct.xml
```

Make sure your driver is configured for STA+P2P concurrent mode or you may encounter error when you open the Wi-Fi. Please refer to “**5. Driver Configurations for Android 7.0**”

- **Set wifi.interface**

To specify the wifi interface name in Android, a system property named “wifi.interface” is used. For Realtek Wi-Fi driver, Wi-Fi interface name is assigned with “wlan%d”. In general, you should set wifi.interface as “wlan0”.

```
PRODUCT_PROPERTY_OVERRIDES += \
    wifi.interface=wlan0
```

- **Include rtw\_fwloader**

To include rtw\_fwloader in the system image, add rtw\_fwloader into the PRODUCT\_PACKAGES variable.

```
ifeq ($(BOARD_WIFI_VENDOR), realtek)
PRODUCT_PACKAGES += rtw_fwloader
#endif
```

### 3. System Resource Configurations

You should set the following four resource configurations for your platform to configure the network function and enable the corresponding UI interface. In general, you can set the following configurations in your platform dependent config.xml file. Take TI panda board for example:

ANDROID\_SDK/device/ti/panda/overlay/frameworks/base/core/res/res/values/config.xml

Or the global config.xml file:

ANDROID\_SDK/frameworks/base/core/res/res/values/config.xml

- **networkAttributes**

To define the system’s available network interfaces, make sure the wifi interface

items is defined in the networkAttributes resource configuration in the config.xml. For example:

```
<string-array translatable="false" name="networkAttributes">
    <item>"wifi,1,1,1,-1,true"</item>
    <item>"bluetooth,7,7,0,-1,true"</item>
    <item>"ethernet,9,9,2,-1,true"</item>
</string-array>
```

- **radioAttributes**

To define the system's available network interfaces, we need to define interface items for wifi in the radioAttributes resource configuration. For example:

```
<string-array translatable="false" name="radioAttributes">
    <item>"1,1"</item>
    <item>"7,1"</item>
    <item>"9,1"</item>
</string-array>
```

- **config\_tether\_wifi\_regexs**

The interfaces set here are tetherable Wi-Fi interfaces which will be used as interfaces for Wi-Fi LAN port. We use 'wlan0' by default when our Wi-Fi is set as softap mode. So it needs to set 'wlan0' here. For example:

```
<string-array translatable="false" name="config_tether_wifi_regexs">
    <item>"wlan0"</item>
</string-array>
```

- **config\_tether\_upstream\_types**

The connection types set here are used as the interfaces for WAN port to connect to internet. For example, adding Wi-Fi and Ethernet:

```
<integer-array translatable="false" name="config_tether_upstream_types">
    <item>1</item>
    <item>9</item>
</integer-array>
```

At least one item should be declared here to enable the “Tethering & portable hotspot” option of WirelessSettings in Settings.apk.

To know the definition and set other upstream connection types, please refer to ANDROID\_SDK/frameworks/base/core/java/android/net/ConnectivityManager.java.

- **config\_enableWifiDisplay**

To enable Wi-Fi Display(Miracast) function, set config\_enableWifiDisplay to true:

```
<bool name="config_enableWifiDisplay">true</bool>
```

#### 4. wpa\_supplicant\_8

We provide [wpa\\_supplicant\\_8\\_N\\_7.x\\_rtw\\_r20140.20161115.tar.gz](#) or newer version in the wpa\_supplicant\_hostapd/ of our SW release package. You can:

- **Use the [wpa\\_supplicant\\_8\\_N\\_7.0\\_rtw](#) instead of the original**
1. Backup and remove the original external/wpa\_supplicant\_8/ folder
  2. Extract and copy the [wpa\\_supplicant\\_8\\_N\\_7.0\\_rtw](#) tar file to the external/ folder of your Android SDK.
  3. Rename [wpa\\_supplicant\\_8\\_N\\_7.0](#) as [wpa\\_supplicant\\_8](#).

#### 5. Driver Configurations for Android 7.0

Android 7.0 support two scenarios for Wi-Fi solution:

- **STA/AP – Switch between STA and AP mode**
- **(STA+P2P)/AP – Switch between STA+P2P concurrent and AP mode**

The configuration of driver to fit the requirement of each scenario, see the following table:

MACRO	STA /AP	(STA+P2P)/AP	Kernel ver.
CONFIG_IOCTL_CFG80211	Defined	Defined	ver. >= 2.6.35
RTW_USE_CFG80211_STA_EVENT	Defined	Defined	ver. >= 3.2.0
<a href="#">CONFIG_RADIO_WORK</a>	<a href="#">Defined</a>	<a href="#">Defined</a>	-
CONFIG_CONCURRENT_MODE	Undefined	Defined	-
RTW_ENABLE_WIFI_CONTROL_FUNC	Defined for platform device/driver mechanism		
<a href="#">CONFIG_RTW_ANDROID</a>	<a href="#">Must set this when rtk driver ver &gt;= v5.9, please refer to section 5.1</a>		

- **CONFIG\_IOCTL\_CFG80211** is used for driver to enable cfg80211 ioctl interface, which is required by Realtek Wi-Fi to operate on Android 7.0 system.

- **RTW\_USE\_CFG80211\_STA\_EVENT** is used for driver to indicate new cfg80211 STA event, which is required by wpa\_supplicant\_8 of Android 7.0 Linux kernel supports this feature after kernel 3.2. For kernel version between 3.0 and 3.2, please refer to the patch file:

linux-3.0.42\_STATION\_INFO\_ASSOC\_REQ\_IES.diff

- **CONFIG\_RADIO\_WORK** is used for driver to fit 'radio work' mechanism of Android 7's wpa\_supplicant\_8. If this MACRO doesn't exist in driver's source code, please contact with Realtek technical windows for suitable driver.

- **CONFIG\_CONCURRENT\_MODE** is used for driver to enable concurrent mode, which is required by STA+P2P concurrent mode of Android 7.0.

- **RTW\_ENABLE\_WIFI\_CONTROL\_FUNC** is used to register platform driver callbacks. If your platform needs those callbacks, please define this macro to register platform driver callback functions. For example, these functions include:

```
static struct platform_driver wifi_device = {  
    .probe          = wifi_probe,  
    .remove         = wifi_remove,
```

By default, the probe callback is used to set up Wi-Fi power and remove callback is used to close Wi-Fi power.

To compile Realtek Wi-Fi driver with the above setting, please refer to the following document:

document/Quick\_Start\_Guide\_for\_Driver\_Compilation\_and\_Installation.pdf

Adding platform selection and setting sections for compilation settings of your platform.

For example, if you want to configure Realtek Wi-Fi driver for the (STA+P2P)/AP scenario, make sure the macros: CONFIG\_IOCTL\_CFG80211, RTW\_USE\_CFG80211\_STA\_EVENT, **CONFIG\_RADIO\_WORK** and CONFIG\_CONCURRENT\_MODE are defined into the EXTRA\_CFLAGS settings as following:

```

CONFIG_PLATFORM_ANDROID_M60_SAMPLE = y
...
...
...
ifeq ($(CONFIG_PLATFORM_ANDROID_ML0_SAMPLE), y)
EXTRA_CFLAGS += -DCONFIG_LITTLE_ENDIAN
EXTRA_CFLAGS += -DCONFIG_CONCURRENT_MODE
EXTRA_CFLAGS += -DCONFIG_IOCTL_CFG80211 -DRTW_USE_CFG80211_STA_EVENT
EXTRA_CFLAGS += -DCONFIG_RADIO_WORK
ARCH := arm
CROSS_COMPILE := /toolchain/bin/arm-none-linux-gnueabi-
KSRC := / android_sdk/android_1/ kernel
endif

```

## 5.1. CONFIG\_RTW\_ANDROID

From Wifi driver version 5.9, a new setting CONFIG\_RTW\_ANDROID is added in Makefile, We can set CONFIG\_RTW\_ANDROID with the Android version in Makefile. e.g. CONFIG\_RTW\_ANDROID = 8

Please note that we must set CONFIG\_RTW\_ANDROID with correct Android version from wifi driver version 5.9, otherwise there will be problem in wifi driver for Android. And the default value of CONFIG\_RTW\_ANDROID is 0, which means the driver is for pure linux, not Android.

( CONFIG\_RTW\_ANDROID=4 means Android 4.4 )

Example in Makefile:

```

##### Android #####
# CONFIG_RTW_ANDROID - 0: no Android, 4/5/6/7/8/9/10 : Android version
CONFIG_RTW_ANDROID = 8

ifeq ($(shell test $(CONFIG_RTW_ANDROID) -gt 0; echo $$?), 0)
EXTRA_CFLAGS += -DCONFIG_RTW_ANDROID=$(CONFIG_RTW_ANDROID)
endif

```

Then most of the settings mentioned above are set automatically by Android version (CONFIG\_RTW\_ANDROID) in drv\_conf.h, and we don't need to write these setting in Makefile; Except CONFIG\_CONCURRENT\_MODE and RTW\_ENABLE\_WIFI\_CONTROL\_FUNC still need to be set manually, depends on the platforms in Makefile as above as before.

Drv\_conf.h

```
#define CONFIG_IOCTL_CFG80211

#define RTW_USE_CFG80211_STA_EVENT

#if (CONFIG_RTW_ANDROID > 4)
#ifndef CONFIG_RADIO_WORK
#define CONFIG_RADIO_WORK
#endif
#endif

#if (CONFIG_RTW_ANDROID >= 8)
    #if (LINUX_VERSION_CODE >= KERNEL_VERSION(3,18,0))
```

## 6. FAQ

### 6.1. Wi-Fi (STA mode)

#### 6.1.1. Why Wi-Fi can't enable?

The whole Wi-Fi enabling procedure includes the following three main check points. Please check in sequence:

- **Is network interface(s) created?**
  - insmod driver success
  - Wi-Fi device is recognized
  - wlan0 is created
- **Does wpa\_supplicant run successfully?**
  - wpa\_supplicant.conf (and p2p\_supplicant.conf) exists and is correct
  - Service definition of wpa\_supplicant exists and is correct
  - Binary file wpa\_supplicant exists and is executable

- **Do connections of communication socket setup?**

- Make sure the communication socket settings is matched below:
  - ◆ ctrl\_interface in:  
/data/misc/wifi/wpa\_supplicant.conf  
(and /data/misc/wifi/p2p\_supplicant.conf)
  - ◆ Service definition of wpa\_supplicant
  - ◆ Paths of communication socket in wifi.c

## **6.2. Portable Wi-Fi hotspot (AP mode)**

### **6.2.1. Why Portable Wi-Fi hotspot can't enable?**

The whole Portable Wi-Fi hotspot enabling procedure includes the following three main check points. Please check in sequence:

- **Is network interface created?**
  - insmod driver success
  - Wi-Fi device is recognized
  - wlan0 is created
- **Does netd and hostapd run successfully?**
  - /data/misc/wifi/hostapd.conf exists and is correct
  - Binary file netd and hostapd exist and are executable
- **Does dnsmasq run successfully?**
  - Binary file dnsmasq exist and are executable

## **6.3. Wi-Fi Direct (P2P mode)**

### **6.3.1. There is no Wi-Fi Direct UI shown?**

Please refer to “**Add android.hardware.wifi.direct.xml**” in chapter 2.3. Others to enable Wi-Fi Direct functionality of Android N.

### **6.3.2. Wi-Fi Direct can't scan any peer?**

First, make sure you have workable Wi-Fi Direct device nearby. Make them into Wi-Fi Direct scanning state. Push “SEARCH FOR DEVICES” button also in our device and wait for a while.

If there is still no peer shown the problem is usually caused by wrong service definition of wpa\_supplicant services. Please refer to “**wpa\_supplicant**” in chapter 2.2. **init.xxx.rc** to check your service definition of wpa\_supplicant.