

# Realtek Wi-Fi SDK for Android ICS

## ver. 2.2.0

### Contents

Release History .....	1
SDK packages .....	2
Introduction.....	2
1. Copy Necessary Files into SDK .....	3
2. Platform Related Files.....	3
2.1. BoardConfig.mk.....	3
2.2. init.xxx.rc .....	5
2.3. device/ti/panda/device.mk .....	6
3. System Resource Configurations .....	6
4. libhardware_legacy .....	8
4.1. Apply wifi_realtek.c .....	8
5. netd.....	8
5.1. Apply SoftapController_realtek.cpp .....	8
6. wpa_supplicant_8 .....	8
6.1. Patch for wpa_supplicant_8.....	8
7. Apply Realtek patch Android's frameworks/base/ .....	8
7.1. Patch for SupplicantStateTracker.java .....	9
7.2. Patch for WifiStateMachine.java .....	9
7.3. Patch for WifiP2pService.java .....	9
7.4. Patch for WifiP2pDevice.java.....	11
8. Modify Settings to show Wi-Fi Direct UI.....	12
8.1. Patch for WirelessSettings.java.....	12
9. Driver Config for NL80211 interface .....	12

### Release History

ver. 1.0.0	1. Support STA and SoftAP mode for Android 4.0 (ICS)
ver. 2.0.0	1. Support NL80211 interface 2. Support WiFi-Direct
ver. 2.1.0	1. Update patch for wpa_supplicant_8 2. Update patch for WifiP2pService.java 3. Update patch for WifiP2pDevice.java
ver. 2.2.0	1. Update patch for wpa_supplicant_8 2. Patch for SupplicantStateTracker.java 3. Patch for WifiStateMachine.java

## SDK packages

- hardware/realtek/\*  
Folder to store config file, private code from Realtek.
- device/ti/panda/BoardConfig.mk
- device/ti/panda/init.omap4pandaboard.rc
- device/ti/panda/device.mk
- device/ti/panda/overlay/frameworks/base/core/res/res/values/config.xml  
Reference codes for platform related files, which is retrieved from panda board and has been patched by Realtek
- hardware/libhardware\_legacy/wifi/Android.mk  
Reference codes for applying wifi\_realtek.c
- system/netd/Android.mk  
Reference codes for applying SoftapController\_realtek.c
- external/wpa\_supplicant\_8/\*  
Contain reference codes patched by Realtek for wpa\_supplicant\_8
- frameworks/base/wifi/java/android/net/wifi/SupplicantStateTracker.java
- frameworks/base/wifi/java/android/net/wifi/WifiStateMachine.java  
Reference codes patched by Realtek for WiFi STA mode
- frameworks/base/wifi/java/android/net/wifi/p2p/WifiP2pDevice.java
- frameworks/base/wifi/java/android/net/wifi/p2p/WifiP2pService.java  
Reference codes patched by Realtek for WiFi-Direct
- packages/apps/Settings/src/com/android/settings/WirelessSettings.java  
Reference code to enable WiFi-Direct Settings UI

## Introduction

This document provides a simple guide to help engineers to apply Realtek Wi-Fi solution onto their Android 4.0 (ICS) system. For now, we have supported the following Wi-Fi functionality:

- Standard STA mode

- Portable Wi-Fi Hotspot(SoftAP mode)
- WiFi-Direct

To port Realtek Wi-Fi driver onto Android platform, you can go through the following guide with reference codes within our driver package's `realtek_wifi_SDK_for_android_ICS_20120404.tar.gz`.

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.

## **1. Copy Necessary Files into SDK**

You need to copy the following folder into your target Android ICS SDK folder:

- `hardware/realtek/`

## **2. Platform Related Files**

### **2.1. BoardConfig.mk**

To apply Realtek Wi-Fi solution onto your Android ICS system, define the following compile-time variables in `BoardConfig.mk` of your platform:

```

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 := rtl8192cu
    #BOARD_WLAN_DEVICE := rtl8192du
    #BOARD_WLAN_DEVICE := rtl8192ce
    #BOARD_WLAN_DEVICE := rtl8192de
    #BOARD_WLAN_DEVICE := rtl8723as
    #BOARD_WLAN_DEVICE := rtl8723au
    #BOARD_WLAN_DEVICE := rtl8188es

    WIFI_DRIVER_MODULE_NAME  := wlan
    WIFI_DRIVER_MODULE_PATH  := "/system/lib/modules/wlan.ko"

    WIFI_DRIVER_MODULE_ARG   := ""
    WIFI_FIRMWARE_LOADER    := ""
    WIFI_DRIVER_FW_PATH_STA  := ""
    WIFI_DRIVER_FW_PATH_AP   := ""
    WIFI_DRIVER_FW_PATH_P2P  := ""
    WIFI_DRIVER_FW_PATH_PARAM := ""
endif

```

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

To distinguish and group the platform Wi-Fi device from products of other companies, 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 ICS, 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 processing 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 remmod. 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.doc

- **WIFI\_FIRMWARE\_LOADER**
- **WIFI\_DRIVER\_FW\_PATH\_STA**
- **WIFI\_DRIVER\_FW\_PATH\_AP**
- **WIFI\_DRIVER\_FW\_PATH\_P2P**
- **WIFI\_DRIVER\_FW\_PATH\_PARAM**

Because our driver has FW embedded inside, and will automatically load FW at NIC initialization process, there is no need to set these 5 variables, just keep them empty.

## 2.2. init.xxx.rc

For WiFi to operate properly, we need wpa\_supplicant daemon to be defined as service inside init.xxx.rc. Please refer to the service definitions below:

```
service wpa_supplicant /system/bin/wpa_supplicant -Dnl80211 -iwlan0 -c/data/misc/wifi/wpa_supplicant.conf
    socket wpa_wlan0 dgram 660 wifi wifi
    group wifi inet
    disabled
    oneshot
```

### 2.3. device/ti/panda/device.mk

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

If you want to use WiFi-Direct functionality, please add the rule in the PRODUCT\_COPY\_FILES variable in your device platform related file to copy the permission definition file, frameworks/base/data/etc/android.hardware.wifi.direct.xml to the system/etc/permissions/ folder of your system image. For example:

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

- **wifi.interface**

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

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

### 3. System Resource Configurations

We should set the following three resource configurations of 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 file such as:

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

Or the global config file:

frameworks/base/core/res/res/values/config.xml

- **networkAttributes**

To define the system’s available network interfaces, we need to define interface items in the networkAttributes resource configuration. 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>
    <item>"wifi_p2p,13,1,0,-1,true"</item>
</string-array>
```

- **radioAttributes**

To define the system's available network interfaces, we need to define interface items in the networkAttributes 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 used as the 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 system to recognized 'wlan0' as Wi-Fi LAN port. For example:

```
<string-array translatable="false" name="config_tether_wifi_regexs">
    <item>"wlan\\d"</item>
</string-array>
```

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

The connection types set here are used as the interfaces for WAN port to connect to internet. Please mask the item 4 (TYPE\_MOBILE\_DUN) in global config file or the Portable Wi-Fi Hostapd UI will not be shown. For example:

```
<integer-array translatable="false" name="config_tether_upstream_types">
    <item>1</item>
    <!-- <item>4</item> -->
</integer-array>
```

Or you could define another entry in your platform dependent config file to override the global definition. For example in our reference code:

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

To know the definition and set other upstream connection types, please refer to

frameworks/base/core/java/android/net/ConnectivityManager.java.

#### **4. libhardware\_legacy**

The libhardware\_legacy library includes functionality for Wi-Fi to operate. We have made modifications and extensions for our Wi-Fi solutions. To apply this, please go through the following instructions:

##### **4.1. Apply wifi\_realtek.c**

Modify hardware/libhardware\_legacy/wifi/Android.mk to include wifi\_realtek.c instead of wifi.c into LOCAL\_SRC\_FILES. For example:

```
ifeq ($(BOARD_WIFI_VENDOR), realtek)
LOCAL_SRC_FILES += ../realtek/wlan/libhardware_legacy/wifi/wifi_realtek.c
else
LOCAL_SRC_FILES += wifi/wifi.c
endif
```

#### **5. netd**

The Portable Wi-Fi hotspot functionality is controlled by the SoftapController subfunction of netd. We provide our own SoftapController implementation to run hostapd. To apply this, please go through the following instructions:

##### **5.1. Apply SoftapController\_realtek.cpp**

Modify system/netd/Android.mk to apply SoftapController\_realtek.cpp instead of SoftapController.cpp into LOCAL\_SRC\_FILES. For example:

```
ifeq ($(BOARD_WIFI_VENDOR), realtek)
LOCAL_SRC_FILES += ../../hardware/realtek/wlan/netd/SoftapController_realtek.cpp
else
LOCAL_SRC_FILES += SoftapController.cpp
endif
```

#### **6. wpa\_supplicant\_8**

##### **6.1. Patch for wpa\_supplicant\_8**

Replace the files in external/wpa\_supplicant\_8 folder from our reference codes to your SDK.

#### **7. Apply Realtek patch Android's frameworks/base/**



### 7.1. Patch for SupplicantStateTracker.java

Add or modify the following code segment in SupplicantStateTracker.java. For the specific line number and code segments, please reference our reference code:

frameworks/base/wifi/java/android/net/wifi/SupplicantStateTracker.java

- **Line 159:**

```
public boolean processMessage(Message message) {  
    if (DBG) Log.d(TAG, getName() + message.toString() + "\n");  
    switch (message.what) {  
        case WifiMonitor.AUTHENTICATION_FAILURE_EVENT:  
            if(!mWifiStateMachine.isDoingWps())  
                mAuthenticationFailuresCount++;  
            mAuthFailureInSupplicantBroadcast = true;  
            break;
```

### 7.2. Patch for WifiStateMachine.java

Add or modify the following code segment in WifiStateMachine.java. For the specific line number and code segments, please reference our reference code:

frameworks/base/wifi/java/android/net/wifi/WifiStateMachine.java

- **Line 951:**

```
public boolean isDoingWps() {  
    return getCurrentState() == mWaitForWpsCompletionState;  
}
```

### 7.3. Patch for WifiP2pService.java

Add or modify the following code segments of WifiP2pService.java. For the specific line number and code segments, please reference our reference code:

frameworks/base/wifi/java/android/net/wifi/p2p/WifiP2pService.java

- **Line 487:**

```

class P2pDisabledState extends State {
    @Override
    public void enter() {
        if (DBG) logd(getName());
        try {
            mNwService.wifiFirmwareReload(mInterface, "STA");
        } catch (Exception e) {
            loge("Failed to reload sta firmware " + e);
        }
    }
    ...
    ...
    ...
}

```

- **Line 973:**

```

...
...
    break;
case WifiMonitor.NETWORK_DISCONNECTION_EVENT:
    if (DBG) logd("Network connection lost");
    sendMessage(WifiP2pManager.REMOVE_GROUP);
    break;
case DhcpStateMachine.CMD_POST_DHCP_ACTION:
    ...
    ...

```

- **Line 1017:**

```

...
...
try {
    mNwService.clearInterfaceAddresses(mGroup.getInterface());
    mNwService.disableIpv6(mGroup.getInterface());
} catch (Exception e) {
    loge("Failed to clear addresses or disable ipv6" + e);
}
...
...

```

- **Line 1060:**

```

...
...
case WifiMonitor.P2P_PROV_DISC_PBC_REQ_EVENT:
    if (mGroup.isGroupOwner())
        notifyP2pProvDiscPbcRequest((WifiP2pDevice) message.obj);
    break;
case WifiMonitor.P2P_PROV_DISC_ENTER_PIN_EVENT:
    if (mGroup.isGroupOwner())
        notifyP2pProvDiscPinRequest((WifiP2pDevice) message.obj);
    break;
...
...

```

#### 7.4. Patch for WifiP2pDevice.java

Add or modify the following code segment in WifiP2pDevice.java to fix the string parsing bug. For the specific line number and code segments, please reference our reference code:

frameworks/base/wifi/java/android/net/wifi/p2p/WifiP2pDevice.java

- **Line 307:**

```

private String trimQuotes(String str) {
    str = str.trim();
    if (str.startsWith("\"") && str.endsWith("\"")) {
        if(str.length()==1)
            return new String();
        return str.substring(1, str.length()-1);
    }
    return str;
}

```

## 8. Modify Settings to show Wi-Fi Direct UI

### 8.1. Patch for WirelessSettings.java

Add or modify the following code segment in WifiP2pService.java to show Wifi-Direct Setting UI. For the specific line number and code segments, please reference our reference code:

packages/apps/Settings/src/com/android/settings/WirelessSettings.java

#### ● Line 158:

```

if (!getPackageManager().hasSystemFeature(PackageManager.FEATURE_WIFI_DIRECT)) {
    getPreferenceScreen().removePreference(wifiP2p);
} else {
    mWifiP2pEnabler = new WifiP2pEnabler(activity, wifiP2p);
}
//getPreferenceScreen().removePreference(findPreference(KEY_WIFI_P2P_SETTINGS));

```

## 9. Driver Config for NL80211 interface

For compatibility issue with older kernel, we disable driver's NL80211 interface by default. To enable driver's NL80211 interface, please modify the specific autoconf file for your Wi-Fi product.

Chip type	Autoconf file to modify
RTL8192CU-series	autoconf_rtl8192c_usb_linux.h
RTL8192CE-series	autoconf_rtl8192c_pci_linux.h
RTL8192DU-series	autoconf_rtl8192d_usb_linux.h
RTL8192DE-series	autoconf_rtl8192d_pci_linux.h
RTL8723AS-series	autoconf_rtl8723a_sdio_linux.h
RTL8723AU-series	autoconf_rtl8723a_usb_linux.h
RTL8188ES-series	autoconf_rtl8188e_sdio_linux.h
RTL8188EU-series	autoconf_rtl8188e_usb_linux.h

For example, if you want to enable NL80211 interface for RTL8192CU-series driver (ex: RTL8188CUS, RTL8192CU), remove the comment of #define CONFIG\_IOCTL\_CFG80211 in autoconf\_rtl8192c\_usb\_linux.h as following:

```
#define CONFIG_IOCTL_CFG80211 1
#ifdef CONFIG_IOCTL_CFG80211
    #define CONFIG_CFG80211_FORCE_COMPATIBLE_2_6_37_UNDER
    // #define CONFIG_DEBUG_CFG80211 1
#endif
```