Linux Network Config 介绍

文件标识：RK-KF-YF-378

发布版本：V1.0.1

日期：2020-08-13

文件密级：□绝密 □秘密 □内部资料 ■公开

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd.

地址： 福建省福州市铜盘路软件园A区18号

网址： www.rock-chips.com

客户服务电话： +86-4007-700-590

客户服务传真： +86-591-83951833

客户服务邮箱： fae@rock-chips.com

前言

概述

本文档主要介绍基于Rockchip Linux 平台的配网方式。

产品版本

芯片名称	内核版本
RK3308/RK3326/RK3288/RK3399/RK1808/RV1108	4.4

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

软件开发工程师

修订记录

版本号	作者	修改日期	修改说明
V1.0.0	CTF/XY	2019-06-16	初始版本
V1.0.1	Ruby Zhang	2020-08-13	更新公司名称和文档格式

目录

1. Wi-Fi/BT配置

1.1 kernel配置

请参考 /docs/Linux reference documents 目录下的 Rockchip Linux WIFI BT 开发指南 V6.0.pdf 文档，第一章节'WIFI/BT 配置'。

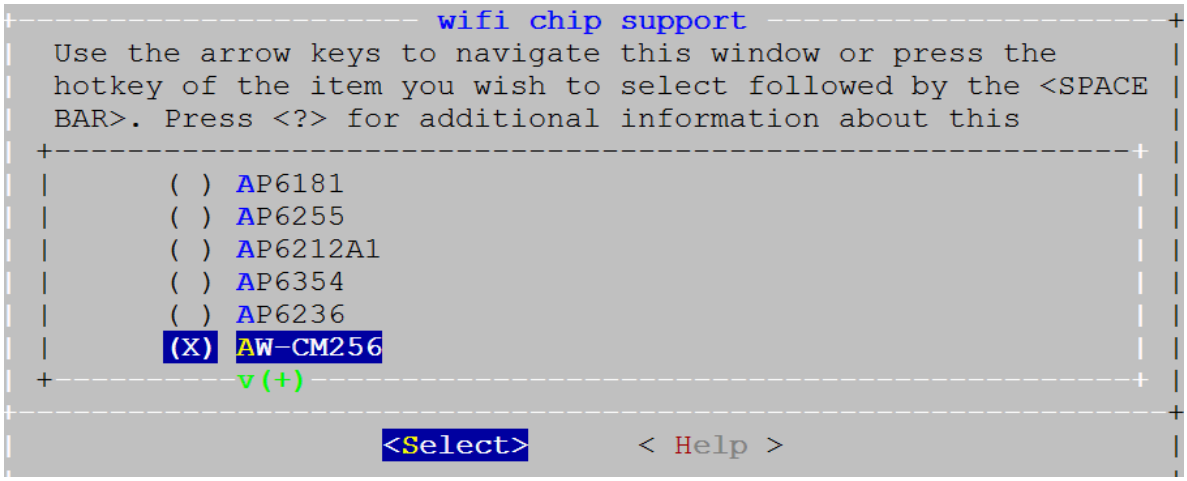
1.2 Buildroot配置

根目录下执行：`make menuconfig`。

1. Wi-Fi 配置：

rkwifibt配置，根据实际使用Wi-Fi选择对应配置，且必须跟kernel配置一致。

```
Symbol: BR2_PACKAGE_RKWIFIBT [=y]
Type  : boolean
Prompt: rkwifibt
  Location:
    -> Target packages
(1)   -> rockchip BSP packages (BR2_PACKAGE_ROCKCHIP [=y])
  Defined at package/rockchip/rkwifibt/Config.in:1
  Depends on: BR2_PACKAGE_ROCKCHIP [=y]
```



2. 蓝牙配置

realtek模组建议使用BlueZ 协议，正基/海华模组建议使用BSA协议。以下配置，根据模组类型三选一：

- Realtek模组选择：bluez-utils 5.x，使用BlueZ需要同时开启: bluez-alsa，readline

```
Symbol: BR2_PACKAGE_BLUEZ5_UTILS [=y]
Type  : boolean
Prompt: bluez-utils 5.x
  Location:
    -> Target packages
(2)   -> Networking applications
  Defined at package/bluez5_utils/Config.in:1
  Depends on: BR2_USE_WCHAR [=y] && BR2_TOOLCHAIN_HAS_THREADS [=y] && BR2_U
  Selects: BR2_PACKAGE_DBUS [=y] && BR2_PACKAGE_LIBGLIB2 [=y]
  Selected by: BR2_PACKAGE_BLUEZ_ALSA [=y] && !BR2_STATIC_LIBS [=n] && !BR2
```

```
    [*] alsa-utils  --->
    [*] alsa-plugins  ----
    [ ] atest
    [ ] aumix
    [ ] bellagio
    [*] bluez-alsa
    [*]   hcitop
    [ ] dvblast
    [ ] dvdauthor
    [ ] dvdrw-tools
    [ ] espeak
    -*- faad2
```

```
Symbol: BR2_PACKAGE_BLUEZ_ALSA [=y]
Type  : boolean
Prompt: bluez-alsa
  Location:
    -> Target packages
(9)   -> Audio and video applications
  Defined at package/rockchip/bluez-alsa/Config.in:1
  Depends on: !BR2_STATIC_LIBS [=n] && !BR2_PACKAGE_BLUEZ_UTILS [=n] && BR2
  Selects: BR2_PACKAGE_ALSA_LIB [=y] && BR2_PACKAGE_BLUEZ5_UTILS [=y] && BR
```

```
    [*] alsa-utils  --->
    [*] alsa-plugins  ----
    [ ] atest
    [ ] aumix
    [ ] bellagio
    [*] bluez-alsa
    [*]   hcitop
    [ ] dvblast
    [ ] dvdauthor
    [ ] dvdrw-tools
    [ ] espeak
    -*- faad2
```

```
Symbol: BR2_PACKAGE_READLINE [=y]
Type  : boolean
Prompt: readline
  Location:
    -> Target packages
      -> Libraries
(7)     -> Text and terminal handling
  Defined at package/readline/Config.in:1
  Selects: BR2_PACKAGE_NCURSES [=y]
  Selected by: BR2_PACKAGE_BLE_WIFICONFIG [=n] && BR2_PACKAGE_ROCKCHIP [=y]
```

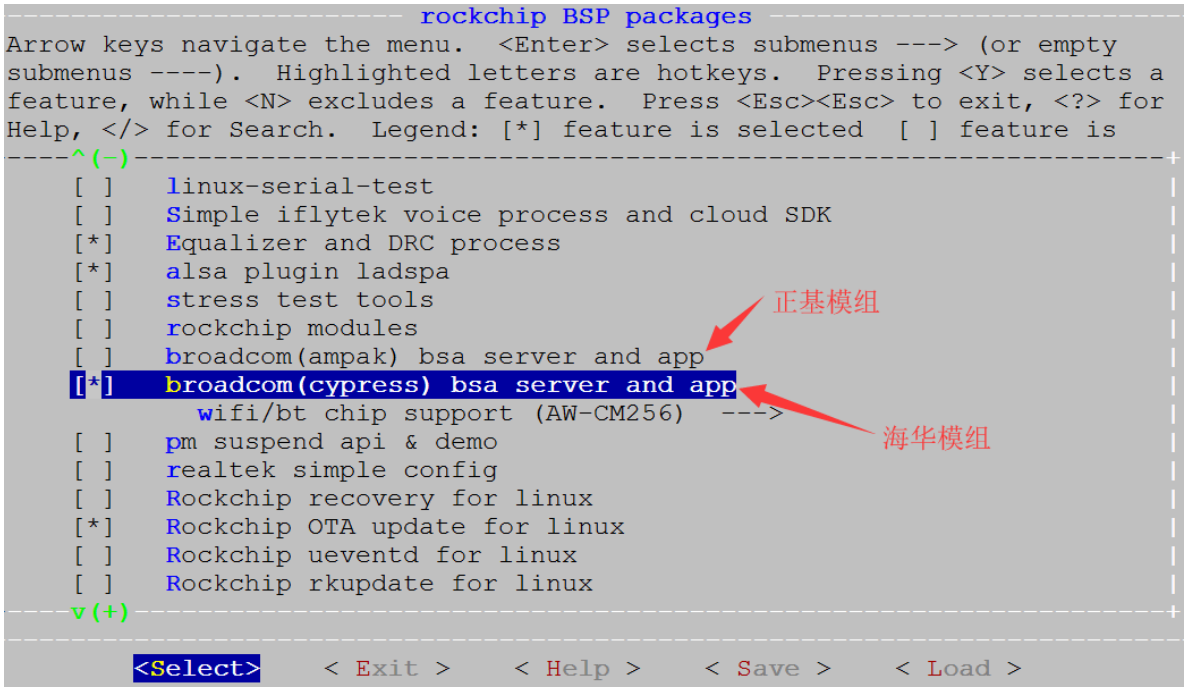
```
    -*-   UTF-8/16/32 support in pcre
    -*-   Unicode properties support in pcre
    [ ] pcre2
    -*- popt
    -*- readline
    [ ] slang
    [ ] tclap
    [ ] ustr
```

- 正基模组选择：broadcom(ampak) bsa server and app

进入 wifi/bt chip support(XXX)---> 选择实际的芯片型号，必须跟rkwifibt配置一致。

- 海华模组选择：broadcom(cypress) bsa server and app

进入 wifi/bt chip support(XXX)---> 选择实际的芯片型号，必须跟rkwifibt配置一致。



3. 退出配置框，`make savedefconfig`保存配置

1.3 编译说明

1. 编译rkwifibt，根目录下执行：

```
make rkwifibt-dirclean && make rkwifibt-rebuild
```

2. 编译蓝牙模块，以下编译选项，根据模组类型三选一

- realtek模组编译：

```
make bluez5_utils-rebuild
make bluez-alsa-rebuild
```

- 正基模组编译：

```
make broadcom_bsa-rebuild
```

- 海华模组编译：

```
make cypress_bsa-rebuild
```

3. 编译deviceio，根目录下执行：

```
make deviceio-dirclean && make deviceio-rebuild
```

4. 打包固件，根目录下执行：

```
./mkfirmware.sh  #也可以./build.sh，全局编译，会自动打包固件
```

2. 命令行配网

1. 首先确保Wi-Fi的服务进程启动，串口输入：`ps | grep wpa_supplicant`

```
# ps | grep wpa_supplicant
  532 root      3380 S    wpa_supplicant -B -i wlan0 -c /data/cfg/wpa_supplica
  618 root      1836 R    grep wpa_supplicant
```

2. 如果没启动，请手动启动：

```
wpa_supplicant -B -i wlan0 -c /data/cfg/wpa_supplicant.conf &
```

3. 修改 /data/cfg/wpa_supplicant.conf 文件，添加配置项

```
network={
    ssid="WiFi-AP"       // Wi-Fi名字
    psk="12345678"       // Wi-Fi密码
    key_mgmt=WPA-PSK     // 选填加密方式，不填的话可以自动识别
    #key_mgmt=NONE       // 不加密
}
```

4. 重新读取上述配置：`wpa_cli reconfigure`
5. 重新连接：`wpa_cli reconnect`

3. 手机配网

3.1 BLE 配网

3.1.1 简介

BLE配网同时支持BlueZ BLE配网和BSA BLE配网，配置参照本文档的第一章节‘WIFI/BT 配置’。并且BLE配网已集成到deviceio，接口位于RkBle.h。

3.1.2 接口说明

请参考/docs/Develop reference documents/DeviceIo目录下
《Rockchip_Developer_Guide_Rk3308_DeviceIo_Bluetooth_CN.pdf》文档，第二章节‘BLE接口介绍（RkBle.h）’。

3.1.3 示例程序

示例程序的路径为：`external/deviceio/test/rk_ble_app.c`。

3.1.4 APP

APP路径：/external/app/RockHome.apk

APP源码路径：/external/app/src/RockHome

该APP仅作为手机端开发Demo，我们适配了Hornor 8，Remi6, 小米6，一加6，OPPO A5型号、iphone6s(plus)、三星S6、VIVO X9等手机。其他型号的手机没有测试，APP兼容性可能存在风险。

3.1.5 配网步骤

该配网步骤以BSA BLE配网为例进行说明，所有板端log均为BSA的配网log。BlueZ操作步骤相同，板端log不同。

1. 确保Wi-Fi的服务进程启动，串口输入：`ps | grep wpa_supplicant`

```
# ps | grep wpa_supplicant
  532 root      3380 S    wpa_supplicant -B -i wlan0 -c /data/cfg/wpa_supplica
  618 root      1836 R    grep wpa_supplicant
```

2. 如果没启动，请手动启动：

```
wpa_supplicant -B -i wlan0 -c /data/cfg/wpa_supplicant.conf &
```

3. 板端命令行执行：`deviceio_test wificonfig`，输入1回车， 启动BLE 配网

```
# deviceio_test wificonfig
version:V1.2.1
#### Please Input Your Test Command Index ####
01.  ble_wifi_config_start
02.  ble_wifi_config_stop
03.  airkiss_wifi_config_start
04.  airkiss_wifi_config_stop
05.  softap_wifi_config_start
06.  softap_wifi_config_stop
07.  voiceprint_wifi_config_start
08.  voiceprint_wifi_config_stop
Which would you like: 1
===== rk_ble_wifi_init =====
hcd_file = /system/etc/firmware/BCM4345C0.hcd
killall: bsa_server: no process killed
bsa_server died.
[   24.786822] [BT_RFKILL]: ENABLE UART_RTS
[   24.889285] [BT_RFKILL]: DISABLE UART_RTS
[   24.889432] [BT_RFKILL]: bt turn on power
start broadcom bluetooth server bsa_sever
|----- bluetooth bsa server is open ------|
[   25.072066] dw-apb-uart ff0e0000.serial: got rx and tx dma channels
DEBUG: check_bsa_server: wait bsa_server open.
DEBUG: check_bsa_server: bsa_server has been opened.
```

4. 设置的BLE广播设备名必须以**RockChip**为前缀，否则APK无法检索到设备：

```
DEBUG: app_ble_rk_server_open: app_ble_rk_server_open
[RK] ble status: RK_BLE_STATE_IDLE
INFO: app_ble_start: app_ble_start
BSA_trace 1029@ 01/01 09h:56m:09s:326ms: BSA_BleEnableInit
BSA_trace 1030@ 01/01 09h:56m:09s:326ms: BSA_BleEnable
DEBUG: app_ble_rk_server_set_device_name: app_ble_device_name: RockChipBle
INFO: app_ble_rk_server_gatt_server_init: wifi_introducer_gatt_server_init
BSA_trace 1031@ 01/01 09h:56m:09s:328ms: BSA_BleSeAppRegisterInit
BSA_trace 1032@ 01/01 09h:56m:09s:329ms: BSA_BleSeAppRegister
INFO: app_ble_rk_server_register: server_if:4
```

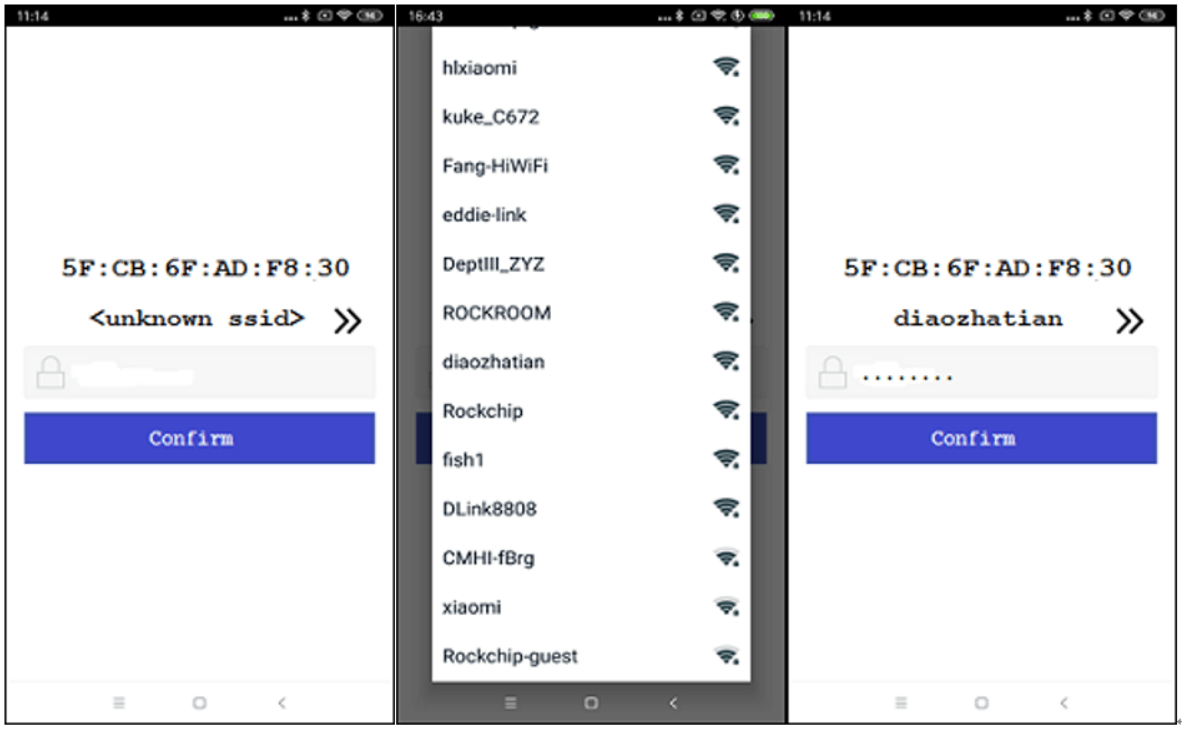
5. 手机端打开APK

点击CONTINUE -> START SCAN，扫描以RockChip为前缀命名的BLE设备：

6. 点击想要连接的BLE设备，开始连接设备，设备连接成功，板端log如下

```
INFO: app_ble_rk_server_profile_cback: BSA_BLE_SE_OPEN_EVT status:0
INFO: app_ble_rk_server_profile_cback: app_ble_rk_server_conn_up conn_id:0x4
INFO: app_ble_rk_server_profile_cback: app_ble_rk_server_conn_up connected to [40:BD
:ED:F8:9A:1D]
DEBUG: app_dm_set_ble_visibility: Set BLE Visibility Discoverable:0 Connectable:0
BSA_trace 1049@ 01/01 09h:57m:56s:262ms: BSA_DmSetConfigInit
BSA_trace 1050@ 01/01 09h:57m:56s:263ms: BSA_DmSetConfig
[RK] ble status: RK_BLE_STATE_CONNECT
INFO: app_ble_rk_server_profile_cback: Stopping Advertisements
BSA_trace 1051@ 01/01 09h:57m:56s:267ms: bsa_sec_event_hdlr event:0
DEBUG: app_mgr_security_callback: event:0
DEBUG: app_mgr_security_callback: BSA_SEC_LINK_UP_EVT bd_addr: 40:bd:ed:f8:9a:1d
DEBUG: app_mgr_security_callback: ClassOfDevice:00:00:00 => Misc device
DEBUG: app_mgr_security_callback: LinkType: 2
DEBUG: bt_mgr_notify_callback: BT_LINK_UP_EVT
```

7. 设备连接成功，APK进入配网界面，点击 >> 按钮 获取Wi-Fi 列表，选择想要连接的Wi-Fi ，输入密码，点击Confirm开始配网：



8. 板端接收到ssid和psk后，开始连接网络

```
[RK] ble_data.cmd: wifisetup, ble_data.start: 1, ble_data.end: 4
01-01 09:59:30.161   954   995 D [RK] wifi ssid is diaozhatian
01-01 09:59:30.162   954   995 D [RK] wifi psk is 7788123456
[RK] rk_config_wifi_thread
[RK] controlWifi connect ...
[RKWIFI] exec1: wpa_cli -iwlan0 disable_network all
[ 7170.184932] CFG80211-ERROR) wl_cfg80211_disconnect : Reason 3
[ 7170.191679] CFG80211-ERROR) wl_is_linkdown : Link down Reason : WLC_E_LINK
[ 7170.191800] link down if wlan0 may call cfg80211_disconnected. event : 16, reason
=2 from 64:09:80:0a:13:b0
[ 7170.216075] CFG80211-ERROR) wl_is_linkdown : Link down Reason : WLC_E_DEAUTH
[ 7170.219478] CFG80211-ERROR) wl_is_linkdown : Link down Reason : WLC_E_DEAUTH
[RKWIFI] exec1: wpa_cli -iwlan0 add_network
format_wifiinfo ssid: 6469616f7a68617469616e
[RKWIFI] exec1: wpa_cli -iwlan0 set_network 2 ssid 6469616f7a68617469616e
format_wifiinfo password: \7\7\8\8\1\2\3\4\5\6
[RKWIFI] exec1: wpa_cli -iwlan0 set_network 2 psk \"\7\7\8\8\1\2\3\4\5\6\"
01-01 09:59:31.301   954  3769 I RK_wifi_connect ssid:"diaozhatian" strlen(ssid):11;
 ori:"diaozhatian" strlen(ori):11; psk:"7788123456"
```

9. 连接成功，板端发送通知给手机APK

```
wifi is connected.
OK
OK
[RK] rk_blewifi_state_callback state: 4
DEBUG: app_ble_rk_server_send_message: conn id : 0x4
INFO: app_ble_rk_server_send_message: Sending Notification
INFO: app_ble_rk_server_send_notification: app_ble_rk_server_send_notification
BSA_trace 1220@ 01/01 09h:59m:41s:219ms: BSA_BleSeSendIndInit
DEBUG: app_ble_rk_server_send_notification: uuid: 00009999-0000-1000-8000-00805F9B34
FB
DEBUG: app_ble_rk_server_send_notification: uuid_string: 0000180A-0000-1000-8000-008
05F9B34FB
DEBUG: app_ble_rk_server_send_notification: uuid_string: 00009999-0000-1000-8000-008
05F9B34FB
DEBUG: app_ble_rk_server_send_notification: attr_index_notify: 1
BSA_trace 1221@ 01/01 09h:59m:41s:222ms: send notification:
BSA_trace 1222@ 01/01 09h:59m:41s:223ms:     0000: 01
```

10. APK端收到配网成功的通知后，断开BLE连接，返回设备搜索界面，板端log如下

```
DEBUG: app_ble_rk_server_profile_cback: event = 23
INFO: app_ble_rk_server_profile_cback: BSA_BLE_SE_CLOSE_EVT status:19
INFO: app_ble_rk_server_profile_cback: conn_id:0x4
INFO: app_ble_rk_server_profile_cback: app_ble_rk_server_connection_down  conn_id:4
reason:19
DEBUG: app_dm_set_ble_adv_param: BDA:00:00:00:00:00:00
DEBUG: app_dm_set_ble_adv_param: adv_int_min:2056 adv_int_max:2056 inst_id:0
BSA_trace 224@ 01/01 08h:17m:48s:918ms: BSA_DmSetConfigInit
BSA_trace 225@ 01/01 08h:17m:48s:919ms: BSA_DmSetConfig
DEBUG: app_dm_set_ble_visibility: Set BLE Visibility Discoverable:1 Connectable:1
BSA_trace 226@ 01/01 08h:17m:48s:923ms: BSA_DmSetConfigInit
BSA_trace 227@ 01/01 08h:17m:48s:923ms: BSA_DmSetConfig
[RK] ble status: RK_BLE_STATE_DISCONNECT
BSA_trace 228@ 01/01 08h:17m:48s:928ms: bsa_sec_event_hdlr event:1
DEBUG: app_mgr_security_callback: event:1
DEBUG: app_mgr_security_callback: BSA_SEC_LINK_DOWN_EVT bd_addr: 51:59:51:a1:1d:03
DEBUG: app_mgr_security_callback: Reason: 19
DEBUG: app_mgr_security_callback: LinkType: 2
DEBUG: bt_mgr_notify_callback: BT_LINK_DOWN_EVT
```

11. 再次启动配网，需要先输入2，关闭BLE配网；再输入1重新启动BLE，重复上述配网流程。

3.2 AirKiss 配网

3.2.1 简介

目前AirKiss配网只支持rtl8723ds，请参照本文档第一章节 'Wi-Fi/BT 配置'进行相应配置；ap模组请参考external/wifiAutoSetup目录下的说明。

AirKiss兼容性很差，不建议作为唯一的配网方式使用，需要增加其他的配套配网方案，原因请参考《/docs/Develop reference documents/WIFIBT/RK平台RTL8723DS AIRKISS配网说明.pdf》。

目前AirKiss配网已集成到deviceio中，接口位于Rk_wifi.h。

3.2.2 kernel 修改

修改 /drivers/net/wireless/rockchip_wlan/rtl8723ds/Makefile 文件：

```
-CONFIG_WIFI_MONITOR = n
+CONFIG_WIFI_MONITOR = y
```

3.2.3 接口说明

启动AirKiss配网，成功返回0，失败返回-1:

```
int RK_wifi_airkiss_start(char *ssid, char *password)
```

- ssid：手机端发送的Wi-Fi名称
- password：手机端发送的Wi-Fi密码

关闭AirKiss配网

```
void RK_wifi_airkiss_stop()
```

3.2.4 示例程序

示例程序的路径为：external/deviceio/test/rk_wifi_test.c

该测试用例调用RK_wifi_airkiss_start()启动AirKiss，获取ssid和password并启动Wi-Fi配网。主要接口：void rk_wifi_airkiss_start(void *data)， DeviceIOTest.cpp中调用。

```
void rk_wifi_airkiss_start(void *data)
{
    int err = 0;
    struct wifi_info info;
    pthread_t tid = 0;

    memset(&info, 0, sizeof(struct wifi_info));

    printf("===== %s =====\n", __func__);

    if(RK_wifi_airkiss_start(info.ssid, info.psk) < 0)
        return;

    err = pthread_create(&tid, NULL, rk_wifi_config_thread, &info);
    if (err) {
        printf("Error - pthread_create() return code: %d\n", err);
        return;
    }

    while (!wifi_state)
        sleep(1);
} « end rk_wifi_airkiss_start »
```

3.2.5 微信配网方式

可以使用手机APP 或者 扫描微信二维码的方式配置网络.

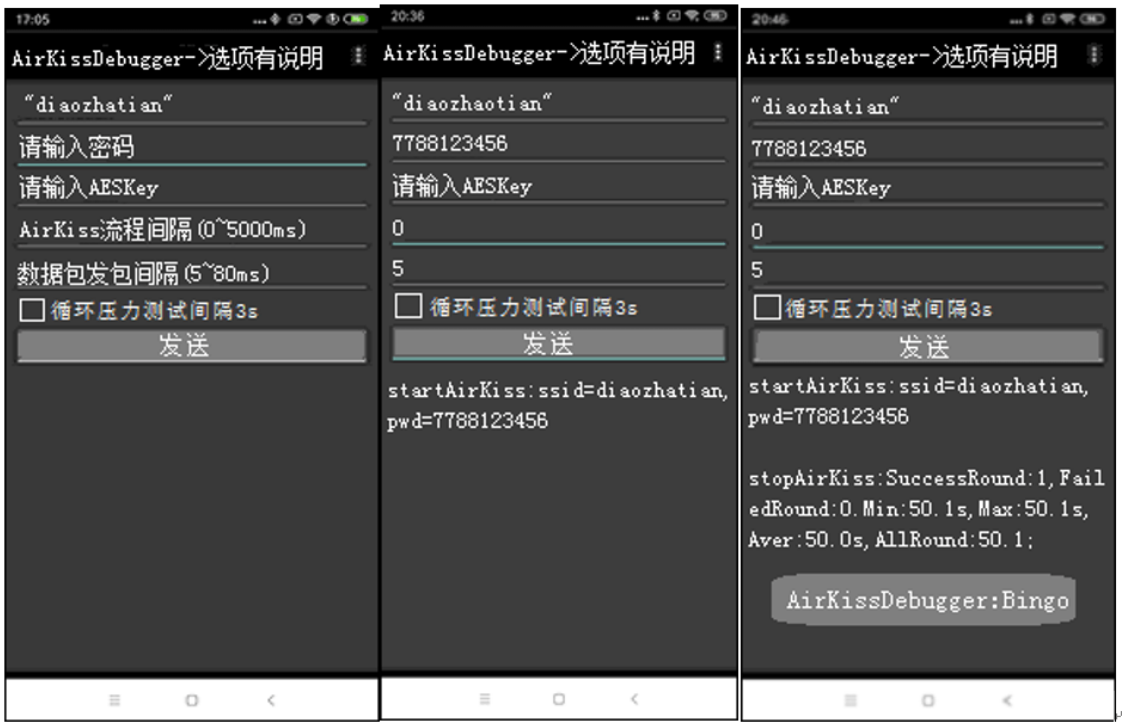
1. 手机APP下载地址：https://iot.weixin.qq.com/wiki/document-download.html ，进入下载中心 -> WiFi 设备 -> AirKiss 调试工具，下载AirKissDebugger.apk

2. 微信扫描如下二维码，二维码配网时，手机必须先连接Wi-Fi ，否则会提示：未能搜索设备，请开启手机Wi-Fi 后重试

微信扫描二维码配置网络

3.2.6 操作示例

1. 手机端操作以APP为例进行说明，打开AirKissDebugger.apk，输入ssid和password，AESKey为空、不输入。点击发送按钮，配网成功会弹窗提示“AirKissDebugger：Bingo”



2. 板端命令行执行：`deviceio_test wificonfig`，输入3回车，启动airkiss 配网

```
# deviceio_test wificonfig
version:V1.2.1
#### Please Input Your Test Command Index ####
01.  ble_wifi_config_start
02.  ble_wifi_config_stop
03.  airkiss_wifi_config_start
04.  airkiss_wifi_config_stop
05.  softap_wifi_config_start
06.  softap_wifi_config_stop
07.  voiceprint_wifi_config_start
08.  voiceprint_wifi_config_stop
Which would you like: 3
===== rk_wifi_airkiss_start =====
```

3. AirKiss 启动成功

```
scan_ap_cnt: 42
use channel: 1 2 3 4 5 6 7 8 9 10 11 13
Start airkiss!
Airkiss init succeed!
```

4. 成功接收ssid和password，并开始配网

```
AirKiss complete: ssid "diaozhatian", pwd "7788123456", random 0xa5
AIRKISS_STATUS_COMPLETE
airkiss_get_result() ok!
ssid = "diaozhatian", pwd = "7788123456", ssid_length = 11, "pwd_le
 = 0xa5
killall: wpa_supplicant: no process killed
```

5. 配网成功

```
wpa_cli -iwlan0 status | grep wpa_state: wpa_state=COMPLETED

wpa_cli -iwlan0 status | grep ip_address: ip_address=192.168.31.164

Congratulation: wifi connected.
Selected interface 'wlan0'
OK
Selected interface 'wlan0'
OK
```

6. 再次启动配网，需要先输入4，关闭AirKiss配网；再输入3重新启动AirKiss，重复上述配网流程

3.3 SoftAP 配网

3.3.1 简介

首先，用SDK板的Wi-Fi创建一个AP热点，在手机端连接该AP热点；其次，通过手机端APK获取SDK板的当前扫描到的热点列表，在手机端填入要连接AP的密码，APK会把AP的ssid和密码发到SDK板端；最后，SDK板端会根据收到的信息连Wi-Fi。

SoftAP配网已集成到deviceio中，接口位于Rk_softap.h。

3.3.2 APP

app路径：/external/app/RockHome.apk

app源码路径： /external/app/src/RockHome

3.3.3 Buildroot配置

```
Type  : boolean
Prompt: softap mode to setup wifi
  Location:
    -> Target packages
(1)   -> rockchip BSP packages (BR2_PACKAGE_ROCKCHIP [=y])
  Defined at package/rockchip/softap/Config.in:1
  Depends on: BR2_PACKAGE_ROCKCHIP [=y]
  Selected by: BR2_PACKAGE_SOFTAPSERVER [=y] && BR2_PACKAGE_ROCKCHIP [=y]


Symbol: BR2_PACKAGE_SOFTAPSERVER [=y]
Type  : boolean
Prompt: socket server based on softap
  Location:
    -> Target packages
(2)   -> rockchip BSP packages (BR2_PACKAGE_ROCKCHIP [=y])
  Defined at package/rockchip/softapServer/Config.in:1
  Depends on: BR2_PACKAGE_ROCKCHIP [=y]
  Selects: BR2_PACKAGE_SOFTAP [=y]
```

```
Symbol: BR2_PACKAGE_IW [=y]
Type  : boolean
Prompt: iw
  Location:
    -> Target packages
(2)   -> Networking applications
  Defined at package/iw/Config.in:1
  Depends on: BR2_TOOLCHAIN_HAS_THREADS [=y]
  Selects: BR2_PACKAGE_LIBNL [=y]
```

3.3.4 接口说明

1. 启动softap配网：

```
RK_softap_start(char* name, RK_SOFTAP_SERVER_TYPE server_type)
```

- name：Wi-Fi热点的名字，前缀必须为Rockchip-SoftAp
- server_type：网络协议类型，目前只支持TCP协议

2. 结束softap配网

```
int RK_softap_stop(void)
```

3. 注册状态回调

```
RK_softap_register_callback(RK_SOFTAP_STATE_CALLBACK cb)
```

- 正在连接网络：RK_SOFTAP_STATE_CONNECTTING
- 网络连接成功：RK_SOFTAP_STATE_SUCCESS
- 网络连接失败：RK_SOFTAP_STATE_FAIL

3.3.5 示例程序

示例程序的路径为：external/deviceio/test/rk_wifi_test.c

主要接口：

```
void rk_wifi_softap_start(void *data)
rk_wifi_softap_stop(void *data)
```

在DeviceIOTest.cpp中调用。

3.3.6 配网步骤

1. 首先确保Wi-Fi的服务进程启动，串口输入：`ps | grep wpa_supplicant`，如果没启动，请手动启动：

```
wpa_supplicant -B -i wlan0 -c /data/cfg/wpa_supplicant.conf &
```

2. 板端命令行执行`deviceio_test wificonfig`，输入5 回车，启动SoftAP配网

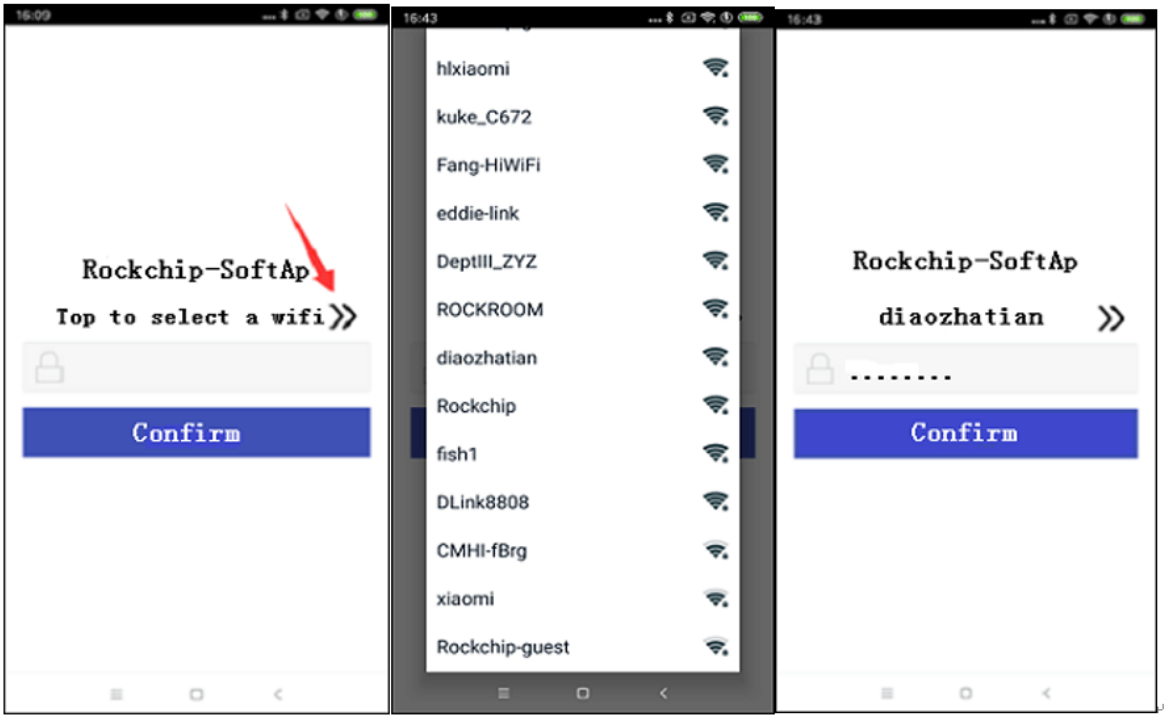
```
# deviceio_test wificonfig
version:V1.2.1
#### Please Input Your Test Command Index ####
01.  ble_wifi_config_start
02.  ble_wifi_config_stop
03.  airkiss_wifi_config_start
04.  airkiss_wifi_config_stop
05.  softap_wifi_config_start
06.  softap_wifi_config_stop
07.  voiceprint_wifi_config_start
08.  voiceprint_wifi_config_stop
Which would you like: 5
[ 4794.018629] Current WiFi chip is AP6255.
Hostapd 143: wifi type: AP6255
Hostapd 19: cmdline = killall dnsmasq
Hostapd 19: cmdline = killall hostapd
killall: hostapd: no process killed
Hostapd 19: cmdline = ifconfig wlan1 down
ifconfig: SIOCGIFFLAGS: No such device
Hostapd 19: cmdline = rm -rf /userdata/bin/wlan1
Hostapd 19: cmdline = iw dev wlan1 del
command failed: No such device (-19)
Hostapd 19: cmdline = ifconfig wlan0 up
Hostapd 19: cmdline = iw phy0 interface add wlan1 type managed
[ 4794.189854] CFG80211-ERROR) wl_cfg80211_event : ignore event 54, not interested
[ 4794.191298] Register interface [wlan1]  MAC: 82:c5:f2:2e:e7:65
[ 4794.191298]
```

3. 打开RockHome.apk，左侧滑选择第三个选项，进入SoftAP配网方式，点击 SEARCH DEVICES，扫描以Rockchip-SoftAp为前缀命名的SoftAP设备

4. 点击想要连接的SoftAP设备，开始连接设备，设备连接成功，板端log如下

```
wlan1: STA 94:87:e0:34:e6:fd IEEE 802.11: associated
wlan1: AP-STA-CONNECTED 94:87:e0:34:e6:fd
[ 5955.601561] CFG80211-ERROR) wl_cfg80211_change_station : WLC_SCB_AUTHORIZE sta_fl
ags_mask not set
```

5. 设备连接成功，APK进入配网界面，点击 >> 获取Wi-Fi 列表，选择想要连接的Wi-Fi，输入密码，点击Confirm开始配网



6. 板子收到ssid和psk，开始连接网络

```
TcpServer recv buf:
POST /provision/wifiSetup HTTP/1.1
Content-Type: application/json
User-Agent: Dalvik/2.1.0 (Linux; U; Android 8.1.0; MI 6X MIUI/V10.2.2.0.ODCCNXM)
Host: 10.201.126.1:8443
Connection: Keep-Alive
Accept-Encoding: gzip
Content-Length: 41

{"ssid":"diaozhatian","pwd":"7788123456"}
do connect ssid:"diaozhatian", psk:"7788123456", isConnecting:0
RK_SOFTAP_STATE_CONNECTTING
```

7. 网络连接成功

```
GET /provision/wifiState HTTP/1.1
Content-Type: application/json
User-Agent: Dalvik/2.1.0 (Linux; U; Android 8.1.0; MI 6X MIUI/V10.2.2.0.ODCCNXM)
Host: 10.201.126.1:8443
Connection: Keep-Alive
Accept-Encoding: gzip


[    64.288035] CFG80211-ERROR) wl_cfg80211_connect : Connecting with64:09:80:0a:13:b
0 ssid "diaozhatian", len (11) channel=4

[    64.613264] wl_bss_connect_done succeeded with 64:09:80:0a:13:b0
[    64.618258] CFG80211-ERROR) wl_cfg80211_determine_vsdb_mode : Same Channel concur
rency is enabled
[    64.696452] wl_bss_connect_done succeeded with 64:09:80:0a:13:b0
```

8. 配网成功后，板端disableWifiAp，手机APK返回设备搜索界面，板端log如下

```
POST /provision/connectResult HTTP/1.1
Content-Type: application/json
User-Agent: Dalvik/2.1.0 (Linux; U; Android 8.1.0; MI 6X MIUI/V10.2.2.0.ODCCNXM)
Host: 10.201.126.1:8443
Connection: Keep-Alive
Accept-Encoding: gzip
Content-Length: 14

{"result":"1"}
RK_SOFTAP_STATE_SUCCESS
Hostapd 19: cmdline = killall hostapd
wlan1: interface state ENABLED->DISABLED
wlan1: AP-STA-DISCONNECTED 94:87:e0:34:e6:fd
[    67.201146] CFG80211-ERROR) wl_cfg80211_del_station : Disconnect STA : ff:ff:ff:f
f:ff:ff scb_val.val 3
[    67.201644] Current WiFi Hostapd 19: cmdline = killall dnschip is AP6255.
masq
[    67.205086] CFG80211-ERROR) wl_notify_connect_status_ap : event WLC_E_DEAUTH(5) s
tatus 0 reason 3
Hostapd 19: cmdline = ifconfig wlan1 down
wlan1: AP-DISABLED
nl80211: deinit ifname=wlan1 disabled_11b_rates=0
```

9. 想要再次启动SoftAP配网，需要先输入6，回车反初始化SoftAP，再输入5重新初始化SoftAP，重复上述配网步骤

3.4 Softap Web UI 配网

3.4.1 简介

Softap Web UI配网原理和上面的SoftAP配网一样，只是手机端无需安装任何APK，直接连上热点，然后在浏览器里面进行进行配网。

3.4.2 代码目录

external/rk_webui/ (包含源码、启动脚本)

buildroot/package/boa/

buildroot/package/rockchip/rk_webui/ （包含编译脚本）

3.4.3 Buildroot配置

首先Buildroot选择`BR2_PACKAGE_RK_WEBUI = y`，然后保存配置重新编译`make rk_webui`，重新生新固件。

```
There is no help available for this option.
Symbol: BR2_PACKAGE_RK_WEBUI [=y]
Type  : boolean
Prompt: Rockchip web ui
  Location:
    -> Target packages
      -> rockchip BSP packages (BR2_PACKAGE_ROCKCHIP [=y])
  Defined at package/rockchip/rk_webui/Config.in:1
  Depends on: BR2_PACKAGE_ROCKCHIP [=y]
```

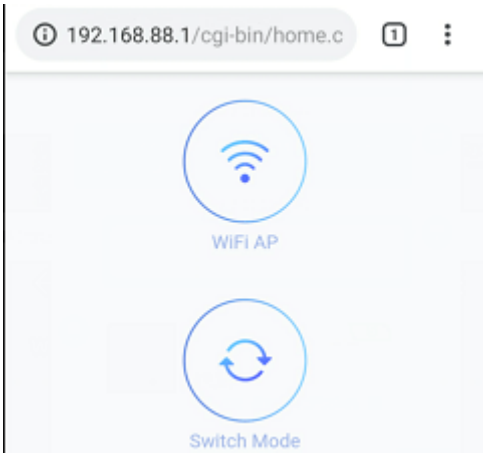
3.4.4 配网

1. 正常启动后执行ps查看，确保有如下4个进程启动：

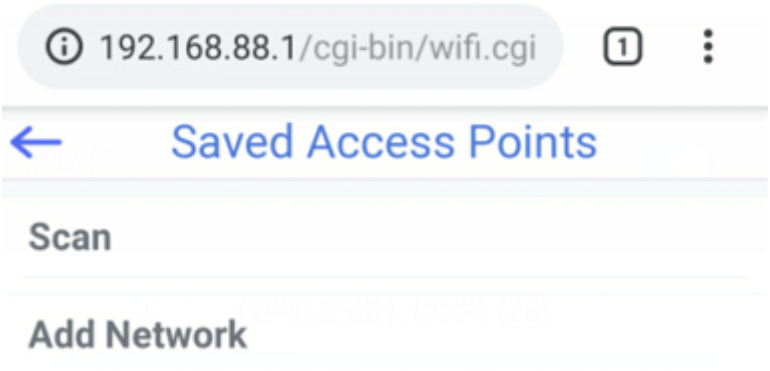
```
394 root      3380 S    wpa_supplicant -B -i wlan0 -c /data/cfg/wpa_supplica
420 root      2004 S    dnsmasq -C /userdata/bin/dnsmasq.conf --interface=p2
422 root      3728 S    hostapd /userdata/bin/hostapd.conf
427 root      1532 S    boa
```

2. 打开手机设置界面搜索Rockchip_WebUI_前缀的AP，比如Rockchip_WebUI_9604(后面的4位数字表示本机Wi-Fi的MAC地址的后4位，方便区分)，点击连接：

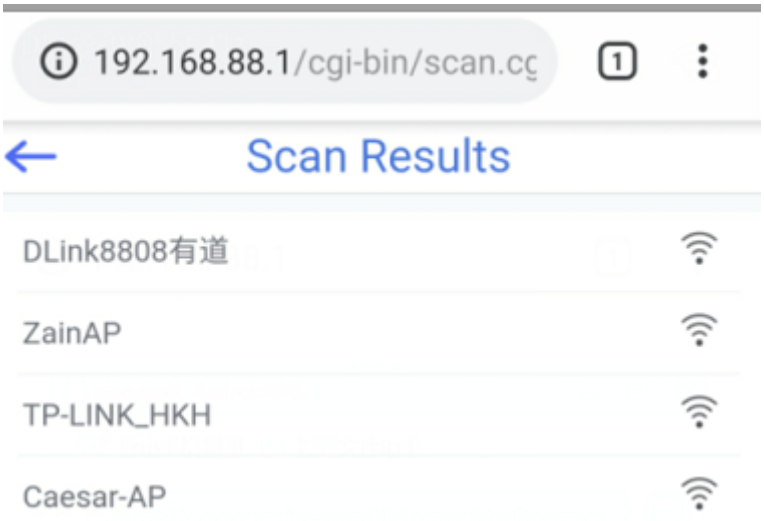
3. 打开手机浏览器，输入：192.168.88.1（浏览器会自动跳转到/cgi-bin/home.c），然后回车出现如下界面：



4. 点击WiFi AP：



5. 点击Scan扫描：



6. 点击要连接的Wi-Fi，然后输入密码并点击Connect（注意：由于Wi-Fi芯片的硬件限制：当连接目前Wi-Fi比如TP-LINK_HKH 和 本身热点Rockchip_WebUI_XXXX不在同一个信道，会导致手机和热点断开，请重新连接热点获取配网状态）

7. 手机重新连接热点，点击刷新，可以看到已经连接Connected（且支持忘记和断开操作）