Rockchip

WIFI/BT 开发指南

发布版本:**1.0**

日期:**2018.05**

前言

概述

本文档主要介绍基于 Rockchip 平台的 WIFI、BT 的内核配置、相关功能的开发等等；

产品版本

芯片名称	内核版本
RK3308	4.0

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

日期	版本	作者	修改说明
2018/05/02	0.01	XY	初始版本
2018/05/16	1.0	XY	正式版本

目录

目录

1 WIFI/BT 内核配置

1.1 DTS

注意如下 pinctrl 的配置，其中 sdio-pwrseq 是 WIFI_REG_ON 管脚

```
wireless-wlan {
    compatible = "wlan-platdata";
    rockchip,grf = <&grf>;
    wifi_chip_type = "ap6255"; //海华模组可需要不用修改此名称，realtek 需要按实际填写
    WIFI,host_wake_irq = <&gpio0 RK_PA0 GPIO_ACTIVE_HIGH>; // WIFI_WAKE_HOST
    // GPIO_ACTIVE_HIGH 特别注意：确认下这个 pin 脚跟 wifi 直接的连接关系，如果中间加了一个反向管就要改成低电平触发
    status = "okay";
};

wireless-bluetooth {
    compatible = "bluetooth-platdata";
    uart_rts_gpios = <&gpio4 RK_PA7 GPIO_ACTIVE_LOW>;
    pinctrl-names = "default", "rts_gpio";
    pinctrl-0 = <&uart4_rts>;
    pinctrl-1 = <&uart4_rts_gpio>;
    BT,power_gpio     = <&gpio4 RK_PB3 GPIO_ACTIVE_HIGH>; // BT_REG_ON
    BT,wake_host_irq = <&gpio4 RK_PB4 GPIO_ACTIVE_HIGH>; // BT_WAKE_HOST
    status = "okay";
};

&pinctrl {
    sdio-pwrseq {
        wifi_enable_h: wifi-enable-h {
            rockchip,pins =
                <0 RK_PA2 RK_FUNC_GPIO &pcfg_pull_none>; // WIFI_REG_ON
        };
    };
};
```

1.2 内核

```
CONFIG_WL_ROCKCHIP:

Enable compatible Wifi drivers for Rockchip platform.

Symbol: WL_ROCKCHIP [=y]
Type  : boolean
Prompt: Rockchip Wireless LAN support
  Location:
    -> Device Drivers
      -> Network device support (NETDEVICES [=y])
        -> Wireless LAN (WLAN [=y])
  Defined at drivers/net/wireless/rockchip_wlan/Kconfig:2
  Depends on: NETDEVICES [=y] && WLAN [=y]
  Selects: WIRELESS_EXT [=y] && WEXT_PRIV [=y] && CFG80211 [=y] && MAC80211 [=y]
```



```
-*- Rockchip Wireless LAN support
[ ]   build wifi ko modules
[*]   Wifi load driver when kernel bootup
< >   ap6xxx wireless sdio cards support
<*>     Cypress wireless sdio cards support
[ ]   Realtek Wireless Device Driver Support  ----
< >   Realtek 8723B SDIO or SPI WiFi
< >   Realtek 8723C SDIO or SPI WiFi
< >   Realtek 8723D SDIO or SPI WiFi
< >   Marvell 88W8977 SDIO WiFi
```

2 配网开发

2.1 命令行配网：

根据对应 WiFi 选择相应配置：

```
There is no help available for this option.
Prompt: wifi chip support
  Location:
    -> Target packages
      -> rockchip BSP packages (BR2_PACKAGE_ROCKCHIP [=y])
        -> rkwifibt (BR2_PACKAGE_RKWIFIBT [=y])
  Defined at package/rockchip/rkwifibt/Config.in:5
  Depends on: BR2_PACKAGE_ROCKCHIP [=y] && BR2_PACKAGE_RKWIFIBT [=y]
  Selected by: BR2_PACKAGE_ROCKCHIP [=y] && BR2_PACKAGE_RKWIFIBT [=y] && m
```

```
--------------------- wifi chip support ----------------------
  Use the arrow keys to navigate this window or press the
  hotkey of the item you wish to select followed by the <SPACE
  BAR>. Press <?> for additional information about this
 +-------------------------------------------------------------+
 |                        ( ) AP6255                           |
 |                        ( ) AP6212A1                         |
 |                        ( ) AW-CM256                         |
 |                        ( ) AW-NAB197                        |
 |                        (X) RTL8723DS                        |
 |                        ( ) RTL8189FS                        |
 +-------------------------------------------------------------+
-----------------------------------------------------------------
                    <Select>      < Help >
-----------------------------------------------------------------
```

首先确保 WiFi 的服务进程启动： 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

修改如下文件：

/ # vi /data/cfg/wpa_supplicant.conf

```
ctrl_interface=/var/run/wpa_supplicant
ap_scan=1
#添加如下配置项
network={
        ssid="WiFi-AP"          // WiFi 名字
        psk="12345678"          // WiFi 密码
        key_mgmt=WPA-PSK    // 加密方式
        # key_mgmt=NONE     // 不加密
}
```

重新读取上述配置：wpa_cli reconfigure

并重新连接：wpa_cli reconnect

2.2 手机配网：

2.2.1 Softap 配网

APP: /external/app/RkEcho.apk

简介：SDK 板的 WiFi 起一个 AP 热点，然后手机端去连接该 AP 热点，然后通过手机端 apk 会获取 SDK 板的当前扫描到的热点列表，然后手机端填入要连接 AP 的密码，apk 会把 AP 的 ssid 和密码发到 SDK 端，最后 SDK 端会根据收到的信息去连接 WiFi。

Buildroot 配置：

```
There is no help available for this option.
Symbol: BR2_PACKAGE_SOFTAPSERVER [=y]
Type  : boolean
Prompt: socket server based on softap
  Location:
    -> Target packages
      -> 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]
```

源码开发目录：

/external/softapServer/ -- WIFI 与 APK 端相关操作

/external/softapDemo/ -- WiFi 相关操作

准备手机安装 apk：

确保 wifi server 进程启动

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

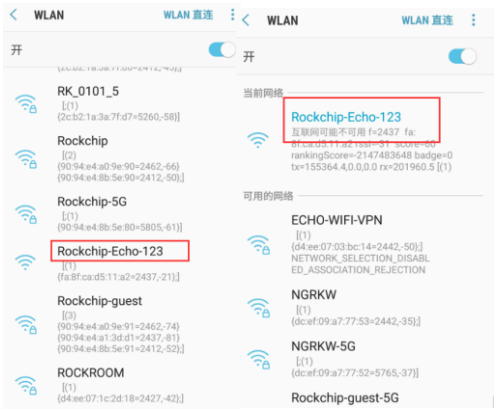
第一步：板子的命令行执行：

softapServer Rockchip-Echo-123 (wifi 热点的名字，前缀必须为 Rockchip-Echo-xxx)

```
/ # softapServer Rockchip-Echo-123
DEBUG 263: check_wifi_chip_type_string: AP6255DEBUG 274:
wifi type: AP6255
DEBUG 297: start softap with name: Rockchip-Echo-123---DEBUG 30: cmdline = killall dnsmasq
killall: dnsmasq: no process killed
DEBUG 30: cmdline = killall hostapd
killall: hostapd: no process killed
DEBUG 30: cmdline = ifconfig wlan1 down
DEBUG 30: cmdline = rm -rf /data/bin/wlan1
DEBUG 30: cmdline = iw dev wlan1 del
DEBUG 30: cmdline = ifconfig wlan0 up
DEBUG 30: cmdline = iw phy0 interface add wlan1 type managed
```

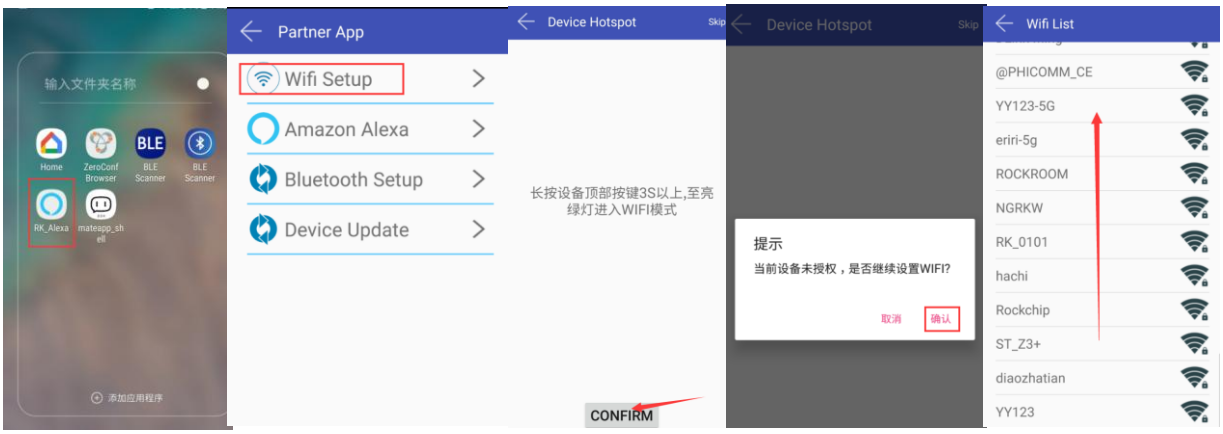
第二步：打开手机的 wifi setting 界面：

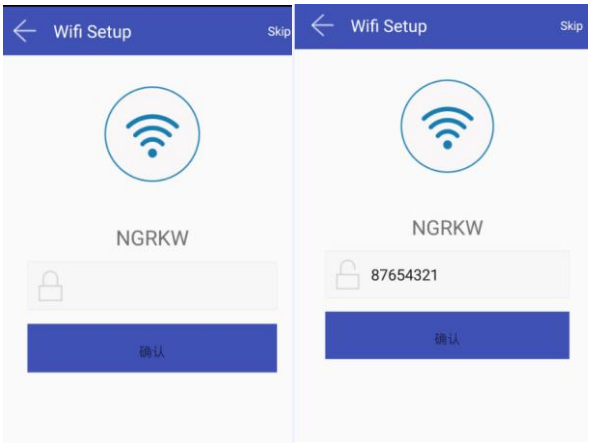
找到 Rockchip-Echo-123 ，点击连接；



第三步：打开手机 apk：

打开 apk，点击 wifi setup->CONFIRM->确认->wifi 列表->点击你要连接的网络名字->输入密码->点击确认





板子串口端显示：

检查网络是否连通：

/ # echo nameserver 8.8.8.8 > etc/resolv.conf // 添加 dns 域名解析

/ # ping www.baidu.com //看下是否 ping 通

注意要点：

1、softspServer Rockchip-Echo-123 执行后命令行是无法退出的，直到配网完成；

2、名字千万不要写错，否则 apk 无法进入确认界面（Rockchip-Echo-xxx）

2.2.2 蓝牙配网

仅支持海华模组，下个版本提供 demo；

2.2.3 Simple config 配网

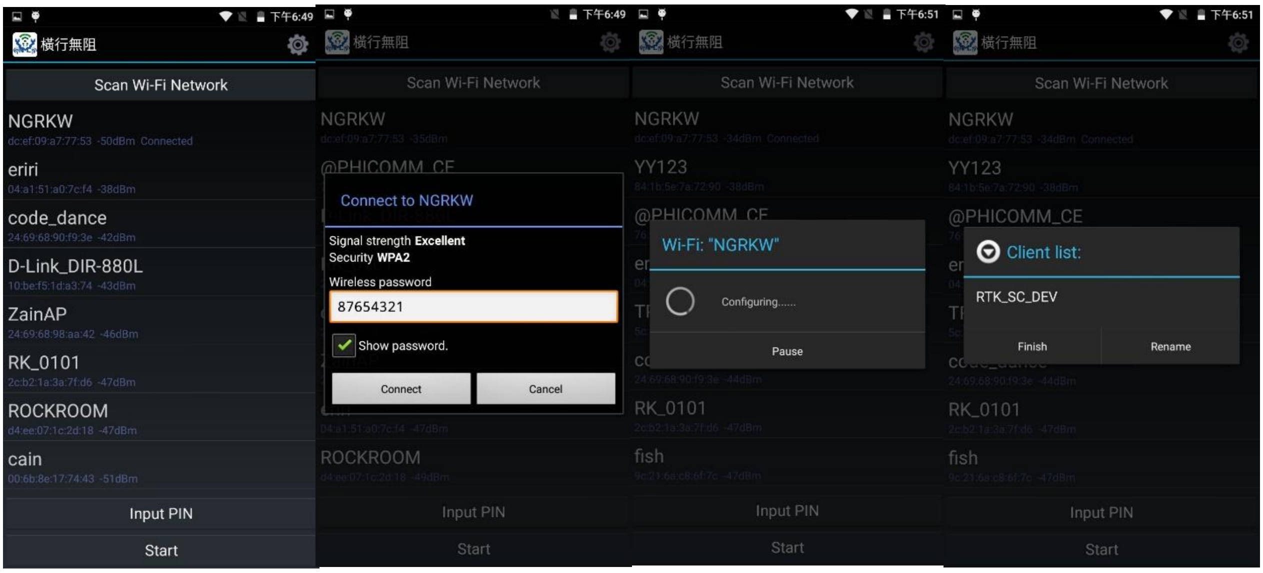
```
There is no help available for this option.
Symbol: BR2_PACKAGE_RTW_SIMPLE_CONFIG [=y]
Type  : boolean
Prompt: realtek simple config
  Location:
    -> Target packages
      -> rockchip BSP packages (BR2_PACKAGE_ROCKCHIP [=y])
  Defined at package/rockchip/rtw_simple_config/Config.in:1
  Depends on: BR2_PACKAGE_ROCKCHIP [=y]
```

仅支持 realtek 模组

external/app/SimpleConfigApp.apk

命令行执行 rtw_simple_config –D & (rtw_simple_config –h 查看帮助)

手机端按照 app：选择网络->输入密码->点击 start 发送->配置完成



板子端显示如下：

```
get the profile
shell:  iwconfig wlan0 mode managed

collect_scanres() target_bssid=[dc:ef:09:a7:77:53], ssid=[NGRKW]
shell:  echo 1 > /proc/net/rtl8723ds/wlan0/survey_info

bssid = [dc:ef:09:a7:77:53], ssid = [NGRKW], :channel=[10] flag=002E
ap_scan=1
network={
        ssid="NGRKW"
        scan_ssid=1
        psk="87654321"
}

shell:  killall wpa_supplicant
killall: wpa_supplicant: no process killed
shell:  ifconfig wlan0 up

shell:  wpa_supplicant -i wlan0 -c /data/wpa_conf -Dnl80211 &
Successfully initialized wpa_supplicant
wlan0: Trying to associate with dc:ef:09:a7:77:53 (SSID='NGRKW' freq=2457 MHz)
wlan0: Associated with dc:ef:09:a7:77:53
wlan0: WPA: Key negotiation completed with dc:ef:09:a7:77:53 [PTK=CCMP GTK=CCMP]
wlan0: CTRL-EVENT-CONNECTED - Connection to dc:ef:09:a7:77:53 completed [id=0 id_str=]
shell:  dhcpcd wlan0
sending commands to master dhcpcd process
the ack from application is:
 20004dc0210db9a21000000000000000052544b5f53435f4445560000000000000000000000000000000000000000000000000000000
the ack from application is:
 20004dc0210db9a21000000000000000052544b5f53435f4445560000000000000000000000000000000000000000000000000000000
the ack from application is:
 20004dc0210db9a21000000000000000052544b5f53435f4445560000000000000000000000000000000000000000000000000000000
sockfd_scan !! pMsg->flag=SC_SUCCESS_ACK
receive config success ack
sockfd_scan !! pMsg->flag=SC_SUCCESS_ACK
receive config success ack
sockfd_scan !! pMsg->flag=SC_SUCCESS_ACK
receive config success ack
sockfd_scan !! pMsg->flag=SC_SUCCESS_ACK
receive config success ack
sockfd_scan !! pMsg->flag=SC_SUCCESS_ACK
receive config success ack
sockfd_scan !! pMsg->flag=SC_SUCCESS_ACK
receive config success ack
sockfd_scan !! pMsg->flag=SC_SUCCESS_ACK
receive config success ack
sockfd_scan !! pMsg->flag=SC_SUCCESS_ACK
receive config success ack
```

3 蓝牙开发

3.1 海华模组

Buildroot 配置：

```
BR2_PACKAGE_CYPRESS_BSA:

broadcom bsa server and app

Symbol: BR2_PACKAGE_CYPRESS_BSA [=y]
Type  : boolean
Prompt: broadcom(cypress) bsa server and app
  Location:
    -> Target packages
      -> rockchip BSP packages (BR2_PACKAGE_ROCKCHIP [=y])
  Defined at package/rockchip/cypress_bsa/Config.in:1
  Depends on: BR2_PACKAGE_ROCKCHIP [=y]
```

相关开发文件源码目录：external/bluetooth_bsa
App 介绍目录：external /release_notes/bsa_examples

基于 broadcom 的海华模组支持 BSA 协议栈，而 BSA 协议栈是 broadcom 公司开发的蓝牙协议栈，类似 BLUEZ，开发人员可以基于它开发各种蓝牙 APP，并且提供丰富的 app demo：

*Application Demo List

release_notes/bsa_examples/Release_app_xx.txt
app_hh -- HH (HID Host): Used to connect to HID Devices (Mouse, Keyboard, Remote Control,)
app_hd -- HD(HID Device): To act HID device
app_av -- AV (Audio/Video): Used to stream audio to stereo headset
app_avk -- AVK (Audio/Video Sink): To act like a stereo headset
app_ag -- HS/HF –AG (Audio Gateway): Used in a phone or device connected to network
app_hs -- HS/HF –HS (HeadSet/HandsFree): To act like a mono headset (used by cellular)
app_fts -- FTP Server -- FTS (File Transfer Server): Used by remote devices (cellular, PC) to access files/folders.
app_ftc -- FTP Client
app_ops -- OPP Server -- OPS Object Push Server : Used by remote devices to push/pull files (e.g. business card)
app_opc -- OPP Client
app_pbs -- PBS (Phone Book Server): Used by remote devices to access local phone book.
app_pbc -- Phone Book Profile Client
app_pam -- Personal Area Networking Profile (PAN)
app_hl -- HDP (Health Device Profile): Used for exchange of medical device data
app_mce – MAP(Message Access Profile) client
app_3d – 3D Synchronization Profile
app_tm – Test Mode, for RF test
app_dg -- SPP (Serial Port Profile): Used for wireless replacement of serial cable
app_ble -- GATT
app_ble_cscc – BLE CSC(Cycling speed and cadence) controller
app_ble_hrc -- BLE Heart Rate Controller

app_ble_pm -- BLE Proximity Monitor
app_ble_rscc -- BLE RSC(Running speed and cadence) controller
app_hogp -- HOGP host

首先上电：
echo 0 > /sys/class/rfkill/rfkill0/state
echo 1 > /sys/class/rfkill/rfkill0/state
启动 server 进程：注意所有的蓝牙相关的进程都要在同一个可写目录执行（cd rw-dir/）
bsa_server -r 12 -p /system/etc/firmware/BCM4345C0.hcd -d /dev/ttyS4 -b /data/btsnoop.log > /data/bsa_log &

启动管理进程：
app_manager &

启动你想要运行的客户端：
app_xxxx

3.1.1 A2DP SINK

执行：bsa_bt_sink.sh start，打开手机蓝牙会显示出 My BSA Bluetooth Device，点击连接即可实现播放音乐的功能；
蓝牙设备的名字可以修改如下代码实现：

/external/bluetooth_bsa /3rdparty/embedded/bsa_examples/linux/app_manager/source/app_manager.c

```
/* Default local Name */
#define APP_DEFAULT_BT_NAME             "My BSA Bluetooth Device"
```

注意：如果连接之后没有声音，请检查声卡的配置。

3.1.2 A2DP SRC

执行：bsa_bt_source.sh start
cd /data/bsa/config //注意 app_av 一定要在该目录执行
把想要播放的文件 xx.wav 文件 push 到/data/bsa/config/test_files/av
./app_av
app_av menu:
a. input 2 (Start Discovery)

```
2
Start Regular Discovery
BSA_trace 23@ 01/01 10h:11m:24s:460ms: BSA_DiscStartInit
BSA_trace 24@ 01/01 10h:11m:24s:460ms: BSA_DiscStart

Bluetooth AV Main menu
  AV Point To Point menu:
    1 => Abort Discovery
    2 => Start Discovery
    3 => Display local source points
    4 => AV Register (Create local source point)
    5 => AV DeRegister (Remove local source point)
    6 => AV Open (Connect)
    7 => AV Close (Disconnect)
    8 => AV Play Tone
    9 => AV Toggle Tone
    10 => AV Play File
    11 => AV Start Playlist
    12 => AV Play Microphone
    13 => AV Stop
    14 => AV Pause
    15 => AV Resume
    16 => AV Send RC Command (Inc Volume)
    17 => AV Send RC Command (Dec Volume)
    18 => AV Close RC
    19 => AV Send Absolute Vol RC Command
    20 => AV Configure UIPC
    21 => AV Change Content Protection (Currently:NONE)
    98 => AV Set Tone sampling frequency
    22 => AV Test SEC codec
    23 => AV Change busy level(1-5)
    99 => Quit
Select action => New Discovered device:0
        Bdaddr:00:24:1c:d9:e0:28
        Name:Motorola Elite Flip
        ClassOfDevice:24:04:04 => Audio/Video
        Services:0x00000000 ()
        Rssi:-68
BSA_trace 25@ 01/01 10h:11m:29s:587ms: COMPLETE_LOCAL_NAME_TYPE:
BSA_trace 26@ 01/01 10h:11m:29s:587ms:     0000: 4d 6f 74 6f 72 6f 6c 61 20 45 6c 69 74 65 20 46   Motorola Elite F
BSA_trace 27@ 01/01 10h:11m:29s:587ms:     0010: 6c 69 70                                          lip
BSA_trace 28@ 01/01 10h:11m:29s:587ms: COMPLETE_16BITS_UUID_TYPE:
BSA_trace 29@ 01/01 10h:11m:29s:587ms:     0000: 08 11 1e 11 0b 11 0e 11 01 11                     ..........
        inq_result_type:1 ble_addr_type:0 device_type:1
Discovery complete
```

发现一个 Audio Sink 的设备

```
Select action => New Discovered device:0
        Bdaddr:f0:13:c3:50:ff:26
        Name:HUAWEI AM08
        ClassOfDevice:24:04:04 => Audio/Video
        Services:0x00000000 ()
        Rssi:-48
        DeviceType:BR/EDR InquiryType:BR AddressType:Public
        Extended Information:
            FullName: HUAWEI AM08
            TxPower:4 dB
            Incomplete Service [UUID16]:
                0x110D [Advanced Audio Distribution]
                0x110B [Audio Sink]
                0x110E [A/V Remote Control]
                0x110F [A/V Remote Control Controller]
                0x111E [Handsfree]
                0x1108 [Headset]
                0x1131 [Headset HS]
```

b. input **6** (start connect)

```
Select action => 6
Bluetooth AV Open menu:
    0 Device from XML database (already paired)
    1 Device found in last discovery
```

c. input **1**

```
Select source => 1
Dev:0
        Bdaddr:f0:13:c3:50:ff:26
        Name:HUAWEI AM08
        ClassOfDevice:24:04:04 => Audio/Video
        Rssi:-48
```

d. input **0**

```
Select device => 0
Connecting to AV device
BSA_trace 35@ 12/31 19h:14m:24s:556ms: BSA_AvOpenInit
BSA_trace 36@ 12/31 19h:14m:24s:556ms: BSA_AvOpen
```

```
Select action => BSA_trace 37@ 12/31 19h:14m:26s:325ms: bsa_cl_av_event_hdlr event:0
BSA_AV_OPEN_EVT status:0 handle:65 cp:1 aptx:0 sec:0
DEBUG: app_read_xml_remote_devices: read(./bt_devices.xml): OK
Added trusted services
Update name with HUAWEI AM08
        ClassOfDevice:24:04:04 => Audio/Video
Update class-of-device [0x24-0x04-0x04]
DEBUG: app_write_xml_remote_devices: write(./bt_devices.xml): OK
BSA_trace 38@ 12/31 19h:14m:28s:303ms: bsa_cl_av_event_hdlr event:4
DEBUG: app_av_cback: BSA_AV_RC_OPEN_EVT status:0 handle:0
BSA_trace 39@ 12/31 19h:14m:28s:343ms: bsa_cl_av_event_hdlr event:19
DEBUG: app_av_cback: BSA_AV_FEAT_EVT Peer feature:24b
```

e. Play File

input **11** to play specified music , and then select specified music, as below:

```
11
Play list:
    0 : ./test_files/av/8k16bpsStereo.wav
        codec(PCM) ch(2) bits(16) rate(8000)
    1 : ./test_files/av/8k8bpsMono.wav
        codec(PCM) ch(1) bits(8) rate(8000)
```

播放 0

```
Select file => 0
0 :./test_files/av/8k16bpsStereo.wav
```

要播放 1 时，需要先输入 AV Stop 停止播放，然后再重复 e 步骤；

3.2 Realtek 模组

使用开源的 bluez + plusaudio

注意由于 realtek 模组的 hci_uart 驱动和官方不兼容，所以先在内核中去掉相关配置：

```
CONFIG_BT_HCIUART:

Bluetooth HCI UART driver.
This driver is required if you want to use Bluetooth devices with
serial port interface. You will also need this driver if you have
UART based Bluetooth PCMCIA and CF devices like Xircom Credit Card
adapter and BrainBoxes Bluetooth PC Card.

Say Y here to compile support for Bluetooth UART devices into the
kernel or say M to compile it as module (hci_uart).

Symbol: BT_HCIUART [=n]
Type  : tristate
Prompt: HCI UART driver
  Location:
    -> Networking support (NET [=y])
      -> Bluetooth subsystem support (BT [=y])
        -> Bluetooth device drivers
  Defined at drivers/bluetooth/Kconfig:72
  Depends on: NET [=y] && BT [=y] && TTY [=y]
```

模组初始化命令：bt_load_rtk_firmware（会执行加载 hci_uart.ko, 然后进行初始化操作），执行完命令后会生成 hci0 节点，可以使用 hciconfig、hcitool 工具操作蓝牙；

板子作为 A2DP SINK 音乐播放：

/usr/libexec/bluetooth/bluetoothd --compat -n &

sdptool add A2SNK

hciconfig hci0 up

```
hciconfig hci0 piscan
hciconfig hci0 name 'rk-bt-12345'
hciconfig hci0 down
hciconfig hci0 up
pulseaudio --start
```

板子作为 A2DP SRC 连接蓝牙播放设备：

```
bt_load_rtk_firmware //重新初始化
sdptool add A2SRC
hciconfig hci0 up
hciconfig hci0 piscan
pulseaudio --start
```