Realtek Bluetooth Porting Guide for Android6.x

v0.3

2017/03/01

修订历史（Revision History）

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 版本 | 修改 | 作者 |
| 2015/12/1 | v0.1 | 初稿 | 毛为锋 |
| 2016/04/20 | v0.2 | v0.2 | 毛为锋 |
| 2017/03/02 | V0.3 | Move patch/code/linux to linux | 毛为锋 |

# 目 录

[修订历史（Revision History） 2](#_Toc3205219)

[目 录 4](#_Toc3205220)

[表格目录 6](#_Toc3205221)

[图表目录 7](#_Toc3205222)

[第1章 文档适用范围 8](#_Toc3205223)

[第2章 Release说明 9](#_Toc3205224)

[2.1 Release文件说明 9](#_Toc3205225)

[2.2 Porting Guide说明 9](#_Toc3205226)

[第3章 Porting方法 11](#_Toc3205227)

[3.1 Android部分 11](#_Toc3205228)

[3.1.1 Changelist 11](#_Toc3205229)

[3.1.2 build 11](#_Toc3205230)

[3.1.3 device/{vendor}/{product}/ 12](#_Toc3205231)

[3.1.4 frameworks/base 19](#_Toc3205232)

[3.1.5 packages/app/Bluetooth 19](#_Toc3205233)

[3.1.6 其他部分 19](#_Toc3205234)

[3.2 Kernel部分 20](#_Toc3205235)

[3.2.1 Change list 20](#_Toc3205236)

[3.2.2 rtk\_btusb driver 21](#_Toc3205237)

[3.2.3 rtk\_rfkill driver 21](#_Toc3205238)

[3.2.4 TUN driver 22](#_Toc3205239)

[3.2.5 UINPUT driver 22](#_Toc3205240)

[3.2.6 HID driver 22](#_Toc3205241)

[第4章 BT功能配置 23](#_Toc3205242)

[4.1 支持的Profile配置 23](#_Toc3205243)

[4.2 配置本地设备名称 23](#_Toc3205244)

[4.3 配置本地设备COD 23](#_Toc3205245)

[4.4 配置Extra Config文件 24](#_Toc3205246)

[第5章 Porting结束后的基本测试 25](#_Toc3205247)

[5.1 配置检查 25](#_Toc3205248)

[5.2 BT测试 25](#_Toc3205249)

[5.2.1 基本功能测试 25](#_Toc3205250)

[第6章 蓝牙问题报告 26](#_Toc3205251)

[6.1 蓝牙Log相关的配置文件 26](#_Toc3205252)

[6.2 Logcat –v time 27](#_Toc3205253)

[6.3 BtSnoop 27](#_Toc3205254)

[6.4 注意事项 27](#_Toc3205255)

[第7章 常见问题分析 28](#_Toc3205256)

[7.1 BT 打开失败(UART) 28](#_Toc3205257)

[7.2 BT 打开失败(USB) 28](#_Toc3205258)

表格目录

**未找到图形项目表。**

图表目录

**未找到图形项目表。**

# 文档适用范围

本文档适用于Realtek Bluetooth Solution for Android 6.x产品。

# Release说明

## Release文件说明

Release由linux,rtkbt,patch目录和Porting Guide，ReleaseNote 文件组成。

* Linux是Realtek提供的Linux Kernel驱动文件，仅用于USB接口的蓝牙卡片。
* rtkbt 是Realtek发布的主要驱动文件,请直接放入device/**{vendor}/{product}/**rtkbt目录即可
* patch 是Android补丁文件.里面包括code和diff两部分,code包含修改后的文件,diff目录是修改的差异部分.由于基于的原始文件会有差异,请根据实际情况合并,切勿直接覆盖.
* Porting Guide也就是本文件，主要说明Realtek驱动包的使用方法。
* ReleaseNote列出驱动包的主要修改。

## Porting Guide说明

* 本文中所有code的修改都遵循如下格式：

1. 所有修改的code都使用方框括起来。
2. 方框中的code会使用SDK中没有颜色的原有code给出修改位置的信息。
3. 所有修改或添加的code均使用灰色突出显示。

对code格式的来举例说明：

如下code是SDK中原有的code：

|  |
| --- |
| ifeq ($(BLUETOOTH\_HCI\_USE\_MCT),true)  LOCAL\_CFLAGS := -DHCI\_USE\_MCT  LOCAL\_SRC\_FILES += \  src/hci\_mct.c \  src/userial\_mct.c  else  LOCAL\_SRC\_FILES += \  src/hci\_h4.c \  src/userial.c  endif |

Realtek需要添加H5的支援，则在porting guide中会给出如下code：

|  |
| --- |
| ifeq ($(BLUETOOTH\_HCI\_USE\_MCT),true)  LOCAL\_CFLAGS := -DHCI\_USE\_MCT  LOCAL\_SRC\_FILES += \  src/hci\_mct.c \  src/userial\_mct.c  else  ifeq ($(BLUETOOTH\_HCI\_USE\_RTK\_H5),true)  LOCAL\_CFLAGS := -DHCI\_USE\_RTK\_H5  LOCAL\_SRC\_FILES += \  src/hci\_h5.c \  src/userial.c \  src/bt\_skbuff.c \  src/bt\_list.c  else  LOCAL\_SRC\_FILES += \  src/hci\_h4.c \  src/userial.c  endif  endif |

灰色部分code是Realtek所添加以及修改的code。

* 该文档中对应的厂商以及平台分别标识为**{vendor}**和**{product}**，不同的客户会对应到不同的厂商以及平台，请客户在porting时注意修改到对应平台的文件. 例如在1195平台该文档{vendor}对应realtek，{product}对应phoenix。

# Porting方法

为了方便客户整合Realtek的WIFI/BT Combo Chip到自己的平台中，Realtek对于如何整合BT的Driver到客户平台分成Android和Linux两部分说明.

## Android部分

### Changelist

1. **build**
2. **device**/**{vendor}/{product}/**
3. **frameworks/base**
4. **packages/app/Bluetooth**
5. **其他部分**

### build

1. build\core\product.mk

|  |
| --- |
| \_product\_stash\_var\_list += \  BOARD\_WPA\_SUPPLICANT\_DRIVER \  BOARD\_WLAN\_DEVICE \  BOARD\_USES\_GENERIC\_AUDIO \  BOARD\_KERNEL\_CMDLINE \  BOARD\_KERNEL\_BASE \  BOARD\_HAVE\_BLUETOOTH \  BOARD\_HAVE\_BLUETOOTH\_BCM \  BOARD\_HAVE\_BLUETOOTH\_QCOM \  BOARD\_HAVE\_BLUETOOTH\_RTK \  BOARD\_VENDOR\_QCOM\_AMSS\_VERSION \  BOARD\_VENDOR\_USE\_AKMD \  BOARD\_EGL\_CFG \  BOARD\_BOOTIMAGE\_PARTITION\_SIZE \  … |

添加Realtek BT Chip支持的宏定义。

### device/{vendor}/{product}/

该目录主要用于设定不同硬件平台board配置信息。不同的子目录对应不同的硬件平台，需要根据具体硬件平台进行修改。

1. 将Release包中rtkbt目录拷贝到device/{vendor}/{product}/下。

如果是UART接口蓝牙芯片,请修改device/{vendor}/{product}/rtkbt/system/etc/bluetooth/rtkbt.conf

将BtDeviceNode设置为正确蓝牙串口设备.例如BtDeviceNode=/dev/ttyS1

**注意:Realtek驱动默认会将UART波特率切换为1.5M.如果使用的是Amlogic平台,建议使用2M波特率.使用方法:**

**请将rtkbt/system/etc/firmware/UART\_2M/下的文件覆盖到rtkbt/system/etc/firmware/目录下同名文件.**

如果是USB接口蓝牙芯片,请保持BtDeviceNode=/dev/rtk\_btusb不变.

1. 添加设备权限.

以使用的是USB蓝牙, BtDeviceNode设置为/dev/rtk\_btusb举例.

* 在device目录下有个file\_contexts文件，在该文件中添加

/dev/rtk\_btusb              u:object\_r:rtk\_bt\_device:s0

* 在device.te中添加

type rtk\_bt\_device, dev\_type;

* device目录下如果有bluetooth.te的话（没有就去externel/sepolicy目录下去找），就在该文件中添加

allow bluetooth rtk\_bt\_device:chr\_file rw\_file\_perms;

1. PRODUCT\_MAKEFILES

首先请查看**device/{vendor}/{product}/****AndroidProducts.mk**，查找到正确的PRODUCT\_MAKEFILES，例如:

|  |
| --- |
| ............  PRODUCT\_MAKEFILES := \  $(LOCAL\_DIR)/ rtk\_phoenix.mk  ............ |

这样可以找到PRODUCT\_MAKEFILES为rtk\_phoenix.mk，然后在rtk\_phoenix.mk中进行下面的修改：

|  |
| --- |
| ............  $(call inherit-product, $(SRC\_TARGET\_DIR)/product/aosp\_base.mk)  $(call inherit-product, device/realtek/phoenix/device.mk)  #rtkbt  $(call inherit-product, device/{vendor}/{product}/rtkbt/rtkbt.mk)  PRODUCT\_BRAND := realtek  PRODUCT\_NAME := rtk\_phoenix  PRODUCT\_DEVICE := phoenix  ............ |

注意: 请根据实际情况替换{vendor}与{product}。

1. 修改device/{vendor}/{product}/rtkbt/Android.mk

由于device/{vendor}/可能存在多个product使用realtek蓝牙芯片,为了降低不同产品之间的耦合性,我们建议在Android.mk增加对产品的判断.例如在编译Android时执行命令lunch命令结果如下:

|  |
| --- |
| $ lunch rtk\_phoenix-eng  …… <省略部分>  PLATFORM\_VERSION=5.1.1  TARGET\_PRODUCT=rtk\_phoenix  TARGET\_BUILD\_VARIANT=eng  ……<省略部分> |

我们建议在Android.mk中增加下面的修改.

|  |
| --- |
| LOCAL\_PATH := $(call my-dir)  ifeq ($(TARGET\_PRODUCT), rtk\_phoenix)  ifeq ($(BOARD\_HAVE\_BLUETOOTH\_RTK),true)  include $(call all-subdir-makefiles)  endif  endif |

1. device/**{vendor}/{product}/**init.**{product}**.rc

注意: 在添加过程中，请务必注意不要有重复项，如果原始文件中有相同内容，请删除相关内容后再添加。

注意: 请根据实际情况替换{vendor}与{product}。

* + 如果该产品采用的是USB蓝牙,请将下列的所有项目添加到device/**{vendor}/{product}/**init.**{product}**.rc。

on boot

………..

# bluetooth

# change back to bluetooth from system

chown bluetooth net\_bt\_stack /data/misc/bluetooth

mkdir /data/misc/bluedroid 0770 bluetooth net\_bt\_stack

# bluetooth LPM

chown bluetooth net\_bt\_stack /proc/bluetooth/sleep/lpm

chown bluetooth net\_bt\_stack /proc/bluetooth/sleep/btwrite

#USB device

insmod /system/lib/modules/rtk\_btusb.ko

chmod 0660 /dev/rtk\_btusb

chown bluetooth net\_bt\_stack /dev/rtk\_btusb

# rfkill

chmod 0660 /sys/class/rfkill/rfkill0/state

chmod 0660 /sys/class/rfkill/rfkill0/type

chown bluetooth net\_bt\_stack /sys/class/rfkill/rfkill0/state

chown bluetooth net\_bt\_stack /sys/class/rfkill/rfkill0/type

# bluetooth MAC address programming

chown bluetooth net\_bt\_stack ro.bt.bdaddr\_path

chown bluetooth net\_bt\_stack /system/etc/bluetooth

chown bluetooth net\_bt\_stack /data/misc/bluetooth

setprop ro.bt.bdaddr\_path "/data/misc/bluetooth/bdaddr"

service dhcpcd\_bnep0 /system/bin/dhcpcd -BKLG

disabled

oneshot

service dhcpcd\_bnep1 /system/bin/dhcpcd -BKLG

disabled

oneshot

service dhcpcd\_bnep2 /system/bin/dhcpcd -BKLG

disabled

oneshot

service dhcpcd\_bnep3 /system/bin/dhcpcd -BKLG

disabled

oneshot

service dhcpcd\_bnep4 /system/bin/dhcpcd -BKLG

disabled

oneshot

service dhcpcd\_bt-pan /system/bin/dhcpcd -ABKL

class main

disabled

oneshot

service iprenew\_bt-pan /system/bin/dhcpcd -n

class main

disabled

oneshot

* + 如果该产品采用的是UART蓝牙，将下面的所有项目添加到device/**{vendor}/{product}/**init.**{product}**.rc。

注意: {UART设备节点} 代表着蓝牙接口的设备节点,例如/dev/ttyS1, /dev/ttyS2…等

chmod 0660 {UART设备节点} 真正应用场合可以修改为 chmod 0660 /dev/ttyS1

on boot

…………..

# bluetooth

# change back to bluetooth from system

chown bluetooth net\_bt\_stack /data/misc/bluetooth

mkdir /data/misc/bluedroid 0770 bluetooth net\_bt\_stack

# bluetooth LPM

chown bluetooth net\_bt\_stack /proc/bluetooth/sleep/lpm

chown bluetooth net\_bt\_stack /proc/bluetooth/sleep/btwrite

#UART device

chmod 0660 {UART设备节点}

chown bluetooth net\_bt\_stack {UART设备节点}

# rfkill

chmod 0660 /sys/class/rfkill/rfkill0/state

chmod 0660 /sys/class/rfkill/rfkill0/type

chown bluetooth net\_bt\_stack /sys/class/rfkill/rfkill0/state

chown bluetooth net\_bt\_stack /sys/class/rfkill/rfkill0/type

write /sys/class/rfkill/rfkill0/state 0

# bluetooth MAC address programming

chown bluetooth net\_bt\_stack ro.bt.bdaddr\_path

chown bluetooth net\_bt\_stack /system/etc/bluetooth

chown bluetooth net\_bt\_stack /data/misc/bluetooth

# setprop ro.bt.bdaddr\_path "/data/misc/bluetooth/bdaddr"

service dhcpcd\_bnep0 /system/bin/dhcpcd -BKLG

disabled

oneshot

service dhcpcd\_bnep1 /system/bin/dhcpcd -BKLG

disabled

oneshot

service dhcpcd\_bnep2 /system/bin/dhcpcd -BKLG

disabled

oneshot

service dhcpcd\_bnep3 /system/bin/dhcpcd -BKLG

disabled

oneshot

service dhcpcd\_bnep4 /system/bin/dhcpcd -BKLG

disabled

oneshot

service dhcpcd\_bt-pan /system/bin/dhcpcd -ABKL

class main

disabled

oneshot

service iprenew\_bt-pan /system/bin/dhcpcd -n

class main

disabled

oneshot

1. **device/{vendor}/{product}/ueventd.{product}.rc**

注意：这个文件只会在采用USB蓝牙时才会使用到。请在文件末尾添加下列行：

/dev/rtk\_btusb 0660 bluetooth net\_bt\_stack

### frameworks/base

### packages/app/Bluetooth

### 其他部分

由于很多平台已经集成了其他的蓝牙方案,所以移植的第一件事可能是先将其他方案的蓝牙相关文件去除,最简单的方法如下:

1. 将BoardConfig中的BOARD\_HAVE\_BLUETOOTH\_XXX全部设置为false或者删除.

注意:正常情况下这样配置后,其他厂家相关的蓝牙代码都不应该继续编译,但是不排除某些蓝牙厂家代码处理不干净,于是关闭这个配置还是照样会编译该厂家独有的文件,所以请仔细检查后面编译生成的文件是否存在冗余.

1. 将通过命令” rm –rf out/target/product/{product}/obj/\*/\*bt\* out/target/product/{product}/obj/\*/\*luetooth\* out/target/product/{product}/obj/\*/\*hci\* out/target/product/{product}/obj/\*/\*a2dp\* out/target/product/{product}/system/\* ”删除掉之前编译出来的文件.
2. 再次编译Android,编译通过后,确定out/target/product/{product}/system/lib/hw/没有bluetooth.default.so.如果要是存在bluetooth.default.so,需要找到对应的bluedroid源码所在位置,然后按照下面的方法修改bluedroid的Android.mk.

ifneq ($(BOARD\_HAVE\_BLUETOOTH\_RTK),true)

…………

#orignal Makefile

…………

endif

1. 然后回头执行2,3步骤.直到编译后不再产生bluetooth.default.so
2. 完成第4步后开始移植我们的蓝牙驱动.
3. 在移植完后请留意一下编译出来的/system/app目录是否有BCM或者其他厂家的蓝牙应用.如果在运行中没有问题,则不予处理,如果运行有问题,请将这些APK从产品中移除.

注意: 上述这些动作根据平台需要灵活处理,请移植工作负责人留意处理.

## Kernel部分

### Change list

Chg linux /driver/bluetooth/Kconfig

Chg linux /driver/bluetooth/Makefile

New linux /driver/bluetooth/rtk\_btusb.c

New linux /driver/bluetooth/tk\_btusb.h

### rtk\_btusb driver

1. 将Realtek提供的文件rtk\_btusb.h和rtk\_btusb.c拷贝到linux kernel的/drivers/bluetooth/目录下；
2. 修改linux kernel的/kernel/drivers/bluetooth/目录下的文件“Kconfig” and “Makefile”

在Kconfig文件中增加BT\_RTKBTUSB的选项：

config BT\_RTKBTUSB

tristate "RTK HCI USB driver"

depends on USB

help

RTK Bluetooth HCI USB driver

在Makefile文件中添加目标文件rtk\_btusb.o:

obj-$(CONFIG\_BT\_RTKBTUSB) += rtk\_btusb.o

1. 在kernel中make menuconfig选中rtk\_btusb driver；

### rtk\_rfkill driver

在需要用到BT\_DIS pin来控制蓝牙芯片的场合, 则需要一个rfkill驱动.

rfkill驱动的具体实现与平台严重相关,所以如果有需要,请根据实际情况进行开发.RTK可以提供reference code.

注意:大多数情况下,客户平台之前搭配过其他家的蓝牙,所以一般都已有rfkill驱动,只是需要将GPIO pin脚接到之前蓝牙使用过的pin就可以,如果不一样,修改一下rfkill驱动中GPIO脚即可.

确认当前是否有支持蓝牙的rfkill驱动方法如下:

cat /sys/class/rfkill/rfkill\*/type

检查是否有bluetooth字样输出,如果有,则已有rfkill驱动,不需要重新开发.

### TUN driver

如果需要支持Bluetooth PAN， 确保TUN Driver已经编译到kernel，

|  |
| --- |
| CONFIG\_TUN=y |

### UINPUT driver

如果需要支持AVRCP功能，确认以下打开配置，

|  |
| --- |
| CONFIG\_INPUT\_UINPUT=y # User level driver support  CONFIG\_INPUT\_MISC=y |

### HID driver

如果需要使用Bluetooth HID，必须支持uhid driver；确认打开需要支持的HID配置

Kernel对一些HID有支持，请尽可能全部打开。

|  |
| --- |
| CONFIG\_UHID=y  CONFIG\_HID\_xxx=y |

注意:经常碰到某些HID设备可以连接,但是不能正常使用,大多都是因为这个配置不全导致的兼容性问题，请确保所有的CONFIG\_HID\_xxx都被设置为y。

# BT功能配置

## 支持的Profile配置

对于有些平板应用不需要支持PBAP，HFP以及HSP，可以按照下面的配置来关闭这几个Profile，如果用户需要支持，只需把对应值设置为true即可。

packages/apps/Bluetooth/res/values/config.xml文件如下

|  |
| --- |
| <resources>  <bool name="profile\_supported\_a2dp">true</bool>  <bool name="profile\_supported\_a2dp\_sink">fasle</bool>  <bool name="profile\_supported\_hdp"> false </bool>  <bool name="profile\_supported\_hs\_hfp"> false </bool>  <bool name="profile\_supported\_hfpclient">fasle</bool>  <bool name="profile\_supported\_hid">true</bool>  <bool name="profile\_supported\_opp">true</bool>  <bool name="profile\_supported\_pan"> false </bool>  <bool name="profile\_supported\_pbap"> false </bool>  <bool name="profile\_supported\_gatt">true</bool>  <bool name="pbap\_include\_photos\_in\_vcard"> false </bool>  <bool name="pbap\_use\_profile\_for\_owner\_vcard"> false </bool>  <bool name="profile\_supported\_map"> false </bool>  <bool name="profile\_supported\_avrcp\_controller">false</bool>  </resources> |

## 配置本地设备名称

修改device/**{vendor}**/**{product}**/rtkbt/bluetooth/bdroid\_buildcfg.h

BTM\_DEF\_LOCAL\_NAME配置项用于配置设备的名字.如果没有这个配置项,将采用ro.product.model property设置的名字作为设备名.

## 配置本地设备COD

修改device/**{vendor}**/**{product}**/rtkbt/bluetooth/bdroid\_buildcfg.h

#define BTA\_DM\_COD {0x1A, 0x01, 0x1C}

三个直接分别表示DevClassServiceClass, DevClassMajorClass, DevClassMinorClass

默认配置如下:

DevClassServiceClass=0x1A

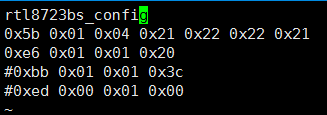
DevClassMajorClass=0x01

DevClassMinorClass=0x1C

表示设备COD为{0x1A,0x01,0x1C}

## 配置Extra Config文件

对于新卡片支持Extra Config文件配置config，如需使用Extra Config文件配置config，可在/data/misc/bluetooth/目录下新建rtk\_btconfig.txt文档，修改文件权限为644。具体配置方式以8723bs为例说明：



1. 文件首行必须为需要配置的蓝牙芯片的config文件名，如这里对8723bs config进行配置，则首行内容必须与8723bs config文件名一致，即“rtl8723bs\_config”
2. 从文件次行开始进行config offset及value值配置。格式为两字节offset + 一字节length + length字节value值，小端模式，以16进制设置，以空格分隔每个字节，每行设置一个offest及value。如图中次行设置即为：offst:0x015b、length:0x04 value:0x21222221
3. 支持用“#”进行单行注释
4. 不支持对MAC address配置

注：请不要随意使用Extra Config文件进行config内容配置，如有需要，请一定请FAE对配置内容进行review，否则容易由于异常配置导致各种问题。

# Porting结束后的基本测试

## 配置检查

为了进一步确保porting没有问题，在测试之前先确认fw以及config文件是否存在。

adb shell到测试平台的根目录，检查测试平台的system/etc/firmware/目录中rtlxxxx\_fw以及rtlxxxx\_config文件是否存在(xxxx为BT Chip型号)。检查Release包中的rtkbt/下的文件是否都已经安装到对应的目录.

## BT测试

***注意：本测试是porting结束后对BT基本和常用功能的一个快速测试，旨在快速验证一些基本问题，不代表BT完整的测试，测试结果也非正式test report。如果使用的是非Realtek BT chip，该项测试可能没有意义。***

### 基本功能测试

1. 打开/关闭BT无失败现象。
2. 能够搜索到近处BT设备。
3. 和搜索到的蓝牙耳机或其他设备配对。
4. 连接上蓝牙耳机，使用BT A2DP听音乐(sdcard确保存在)。
5. 连接上蓝牙耳机，使用BT HFP/HSP打电话(确保用蓝牙时能够正常通话)。
6. 传输文件到远端支持蓝牙OPP Server的设备，从远端支持蓝牙OPP client的设备传送文件到本地(sdcard确保存在)。
7. 连接上蓝牙键盘，打开需要输入的应用，通过蓝牙键盘输入。

# 蓝牙问题报告

当发现蓝牙有问题的时候,需要同时提供下面的log,否则可能会因为log信息不足无法定位问题.所以请务必学会抓取下列蓝牙log的方法.

## 蓝牙Log相关的配置文件

蓝牙相关的配置文件存放在设备的/system/etc/bluetooth/bt\_stack.conf .可以通过adb方式进行修改.默认的配置文件如下:

|  |
| --- |
| # Enable BtSnoop logging function  # valid value : true, false  BtSnoopLogOutput=false  # BtSnoop log output file  BtSnoopFileName=/sdcard/btsnoop\_hci.log  # Preserve existing BtSnoop log before overwriting  BtSnoopSaveLog=false  # Enable trace level reconfiguration function  # Must be present before any TRC\_ trace level settings  TraceConf=true  # configuration for uart card to save HCI log for slave  H5LogOutput=1  # Enable Coex log  BtCoexLogOutput=0  # Trace level configuration  # BT\_TRACE\_LEVEL\_NONE 0 ( No trace messages to be generated )  # BT\_TRACE\_LEVEL\_ERROR 1 ( Error condition trace messages )  # BT\_TRACE\_LEVEL\_WARNING 2 ( Warning condition trace messages )  # BT\_TRACE\_LEVEL\_API 3 ( API traces )  # BT\_TRACE\_LEVEL\_EVENT 4 ( Debug messages for events )  # BT\_TRACE\_LEVEL\_DEBUG 5 ( Full debug messages )  # BT\_TRACE\_LEVEL\_VERBOSE 6 ( Verbose messages ) - Currently supported for TRC\_BTAPP only.  TRC\_BTM=2  TRC\_HCI=2  TRC\_L2CAP=2  TRC\_RFCOMM=2  TRC\_OBEX=2  TRC\_AVCT=2  TRC\_AVDT=2  TRC\_AVRC=2  TRC\_AVDT\_SCB=2  TRC\_AVDT\_CCB=2  TRC\_A2D=2  TRC\_SDP=2  TRC\_GATT=2  TRC\_SMP=2  TRC\_BTAPP=2  TRC\_BTIF=3  TRC\_GAP=2  TRC\_HID=2 |

## Logcat –v time

在抓取log的时候,必须添加”-v time”选项.否则没法将log和出现问题的时间点对应起来,给分析log会带来很大的麻烦.

测试前,可以打开log并将log导入到设备的某个分区,待测试完成后再通过adb, sdcard等方式导出log.

## BtSnoop

BtSnoopLogOutput 设置为 TRUE,打开之后,btsnoop会存放在BtSnoopFileName指明的路径下.在蓝牙打开时,这个文件会自动被覆盖一次,所以重现到问题后,需要尽快采用adb或者sdcard将btsoop文件导出.切勿重新开关蓝牙或者重启设备.

## 注意事项

**报告蓝牙问题至少需要提供下列信息:**

* **logcat –v time**
* **btsnoop**
* **出现问题的时间点(以设备时间为准)**
* **重现步骤.**

# 常见问题分析

## BT 打开失败(UART)

打开H5 UART Driver Log， 使用logcat抓取log，看H5 SYNC过程时候成功，如果H5 SYNC失败，那么需要首先检查硬件电路是否正确（Power Supply， BT Reset PIN， UART TX/RX，CTS/RTS），然后检查卡片efuse，用示波器量测UART波形，看Host是否把数据正确的发送到Controller。

如果H5 SYNC成功，那么下一步就是Change Baudrate，判断Change Baudrate是否成功。如果Change Baudrate失败，那么需要确定Host是否支持该波特率，config文件是否正确设定了波特率。

如果Change Baudrate 成功，下一步是下载fw以及config文件，如果下载完毕之后，收不到Controller回复的Command Complete Event，那么需要检查fw以及config文件是否正确，BT Reset PIN是否为高电平。

如果下载fw以及config文件成功，那么下一步就是根据config文件的设定修改HW Flowcontrol的设置。设置成功之后，bluedroid stack会下第一个HCI Comamnd。

如果第一个HCI Command一直H5 重传，那么说明可能HW flowcontrol有问题，需要检查Host的UART driver是否支持HW Flowcontrol。

## BT 打开失败(USB)

用logcat抓取打开蓝牙打开的log，搜索“dev/bus/usb”字样看是否有这样的log：Added device UsbDevice[mName=/dev/bus/usb/002/002,mVendorId=3034,mProductId=46880,mClass=239,mSubclass=2. 如果有，检查下mVendorId和mProductId是不是对应当前使用的蓝牙芯片。如果没有则是没有识别蓝牙卡片，需要首先检查硬件电路是否正确.

查看USB的驱动是否正常加载。登陆到平台里(adb shell)，然后使用命令lsmod来查看是否有 rtk\_btusb.ko的存在。