RK 平台android系统下蓝牙框架，配置与调试说明

目录

[一、 概述 2](#_Toc525292163)

[1. 框架结构 2](#_Toc525292164)

[2. 蓝牙服务启动流程 3](#_Toc525292165)

[二、 配置 3](#_Toc525292166)

[1. 内核层配置 3](#_Toc525292167)

[2. Android层配置 5](#_Toc525292168)

[3. 其他配置 9](#_Toc525292169)

[三、 调测 9](#_Toc525292170)

[1. 功能性调测 9](#_Toc525292171)

[2. 沟通反馈问题注意事项 18](#_Toc525292172)

[四、 新模块移植与开发 19](#_Toc525292173)

[选择要加载协议栈 19](#_Toc525292174)

[选择调用对应厂商的硬件初始化层 19](#_Toc525292175)

[加入新厂商的协议栈的注意事项 20](#_Toc525292176)

1. 概述

### 框架结构

Android4.2之后采用bluedroid作为协议；整体由bluetooth.apk,bluedroid ,libbt-vendor三个部分组成；bluetooth.apk作为蓝牙服务进程，通过libblueooth\_jni层获取bluetooth\*.default.so提供的bt\_interface\_t控制接口并状态与数据回调注册到bluetooth\*.default.so；完成对蓝牙模块的控制与数据接收；libbt-vendor完成对蓝牙模块硬件初始化与控制（物理通讯接口初始化，固件下载，供电控制）；

Bluetooth.APK

libblueooth\_jni

Libbt-vendor.so

UART或USB

蓝牙模块

控制命令或数据

控制命令或数据

控制命令或数据

bluedroid

Libbt-hci.so

其中RK平台通过判断模块的type，在libbuetooth\_jni层选择加载不同适配厂商协议栈；并在libbt-hci中根据模块类型加载不同厂家模块的libbt-vendor\*.so，完成动态适配模块。

【Android8.0以后底层使用同一套协议栈】

【附注】

1 Jni层根据模块类型加载不同的厂家协议栈【Android8.0以后底层使用同一套协议栈】

packages/apps/Bluetooth/jni/com\_android\_bluetooth\_btservice\_AdapterService.cpp

classInitNative

1. 蓝牙服务启动流程

RK平台下通过BOARD\_HAVE\_BLUETOOTH 控制是否编译生成蓝牙相关的服务和协议栈的so，以及settings页面是否有蓝牙选项；如果所使用的sdk中已经添加动态适配机制的，sdk默认的配置中已经定义该宏，对于没有动态适配机制的，需要在平台做相关配置；

SystemServer注册BluetoothManagerService服务

BluetoothManagerService收到BOOTCOMPLETEED系统广播启动服务（bluetooth.apk开始运行）

Bluetooth.apk进行初始化流程

## 配置

### 内核层配置

3.10 之后的内核 通过 dts 文件 进行配置，蓝牙相关配置节点

wireless-bluetooth {

compatible = "bluetooth-platdata";

/\* wifi-bt-power-toggle; \*/

uart\_rts\_gpios = <&gpio3 GPIO\_A6 GPIO\_ACTIVE\_LOW>;

//正基的部分模块需要对模块上下电时主控的rts脚输出电平切换信号

pinctrl-names = "default","rts\_gpio";

pinctrl-0 = <&uart11\_rts>;

pinctrl-1 = <&uart11\_rts\_gpio>;

BT,power\_gpio = <&gpio2 GPIO\_D5 GPIO\_ACTIVE\_HIGH>;

/\* BT,reset\_gpio = <&gpio2 GPIO\_D5 GPIO\_ACTIVE\_HIGH>; \*/

//power\_gpio 与reset\_gpio 为同样的功能均为 BT的功能使能脚，配置其中一个即可

BT,wake\_gpio = <&gpio3 GPIO\_D3 GPIO\_ACTIVE\_HIGH>;

// 让模块一直处于唤醒状态，当模块处于休眠态时 模块会不响应 协议栈层下发的指令，如果模块无相关功能脚 不配置即可

BT,wake\_host\_irq = <&gpio3 GPIO\_D2 GPIO\_ACTIVE\_HIGH>;

//需要使用 bt唤醒功能（例如 连接hid设备时 设备进入休眠时 此时 hid 设备能够通过bt唤醒主控），如果模块无相关功能脚 不配置即可

status = "okay";

};

物理通讯接口配置

2 串口配置

以 uart1 为例

//带流控

&uart1{

status = "okay";

dma-names = "!tx", "!rx";

pinctrl-0 = <&uart11\_xfer &uart11\_cts>;

};

//不带硬件流控

&uart1{

status = "okay";

dma-names = "!tx", "!rx";

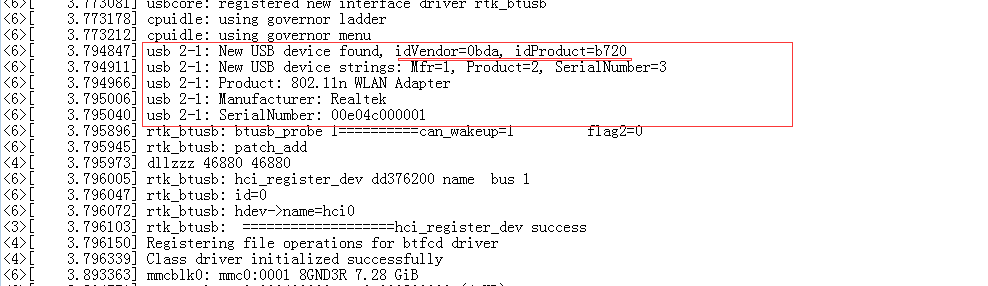
pinctrl-0 = <&uart11\_xfer >;

};

配置生效的情况下 /dev/ 下可以看到对应的ttyS1设备，并且此时向该串口设备上写入数据时 能够测量到相关波形；

USB接口模块配置

usb接口的模块需要确保CONFIG\_BT\_RTKBTUSB 配置为y，生效的情况下 在枚举设备正常时可以看到 /dev/rtkbt\_usb

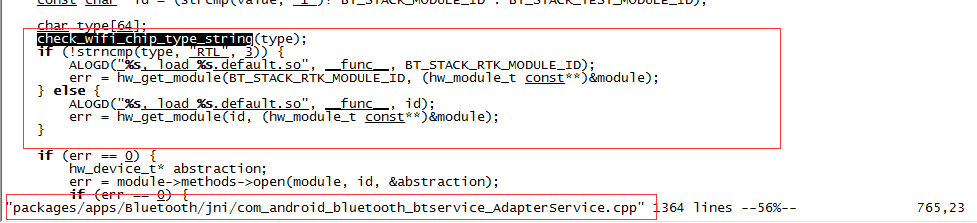


### Android层配置

#### 基础配置

Android4.4 部分平台采用动态适配机制，部分采用静态适配（需要修改wifi\_bt.mk 中的BOARD\_CONNECTIVITY\_VENDOR 与BOARD\_CONNECTIVITY\_MODULE），部分采用动态适配机制，如何判断采用静态适配还是动态适配机制

检查packages/apps/Bluetooth/jni/com\_android\_bluetooth\_btservice\_AdapterService.cpp函数classInitNative是否有调用check\_wifi\_chip\_type\_string 获取wifi\_type 加载不同协议栈的代码；如下截图



确认修改配置是否生效的方法

1 source build/envsetup.sh

2 lunch 选择你编译要使用 lunch

3 get\_build\_var BOARD\_CONNECTIVITY\_VENDOR

get\_build\_var BOARD\_CONNECTIVITY\_MODULE

// 检查 读取出来的内容与配置的内容是否一致，如果不一致检查相关mk脚本

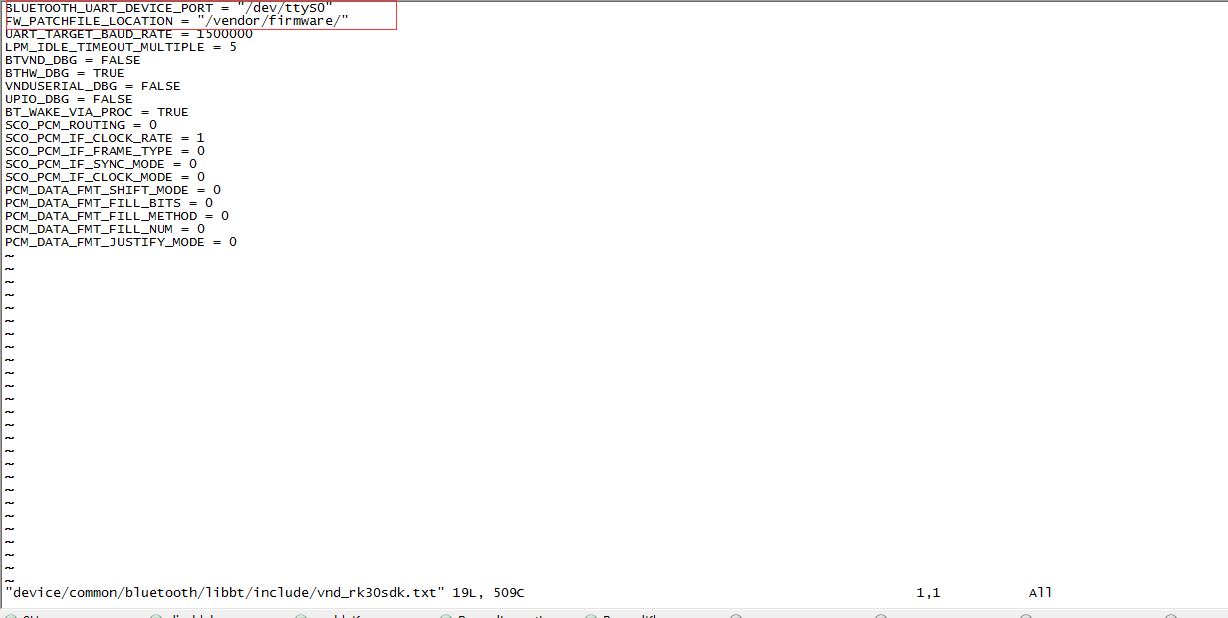
Android5.1 ~Android7.1 采用动态适配机制，android层 保持默认配置即可；

Android8.0 之后的系统由于不同模块厂使用同一个协议栈

#### 通讯接口配置

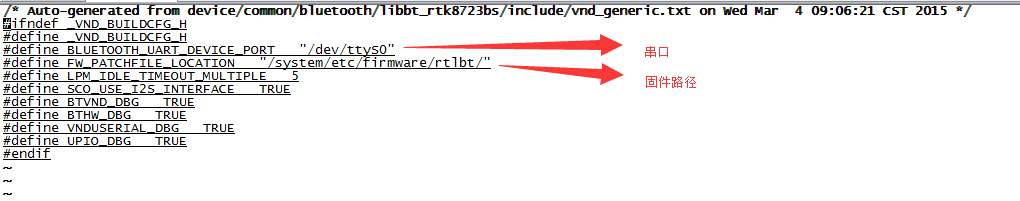
##### 正基系列模块

正基系列模块通过配置文件libbt/include/vnd\_rk30sdk.txt配置串口；



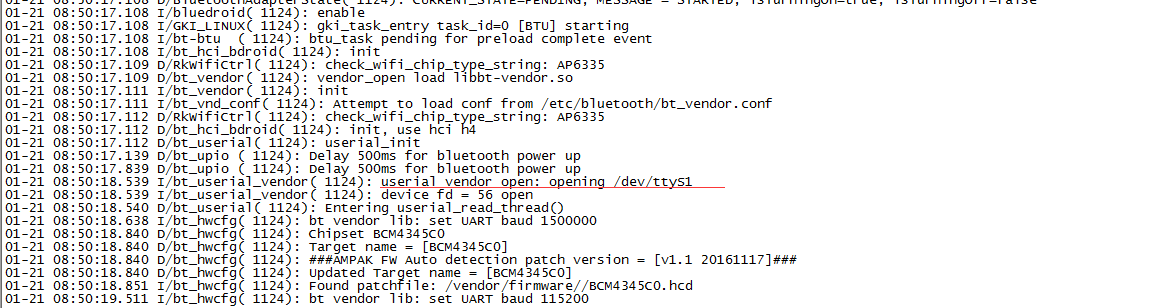
##### Realtek系列模块

Realtek系列模块通过libbt-vendor 层的include/vnd\_buildcfg.h；（libbt层的代码实现可能会发生变动请以实际代码为准）



Realtek系列模块的 波特率在 模块的 config 文件中的进行配置，具体请咨询realtek

通过logcat 日志可以确认自己的配置是否生效，以博通的模块为例



【注】该文档示例性说明，请结合自己所使用的sdk灵活变通

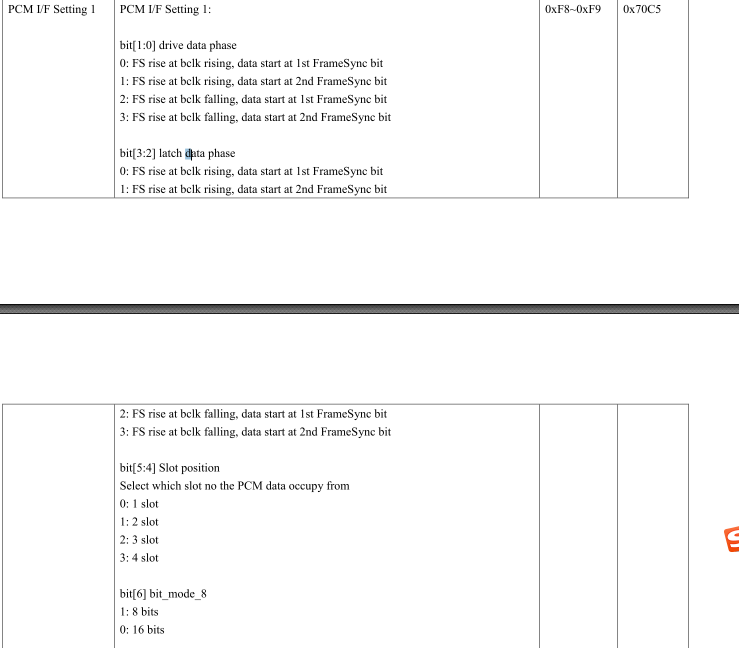
#### PCM相关配置

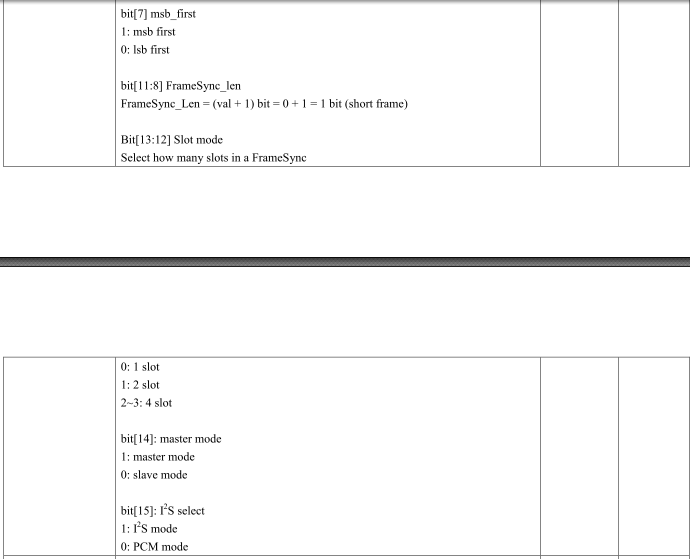
##### 正基系列模块

正基系列模块配置参数位于libbt/include/vnd\_rk30sdk.txt，相关参数说明参看 Bluetooth User Guide for Android 4.2\_4.3\_4.4-v05.pdf

##### Realtek系列模块

Realtek系列PCM配置参数位于对应的config文件中，详细信息见RealtekBluetooth User Guide for Android 4.2\_4.3\_4.4-v05.pdf







上图以 8723bs\_config 为例 f8 f9 的配置内容 对应 BT 作为 master

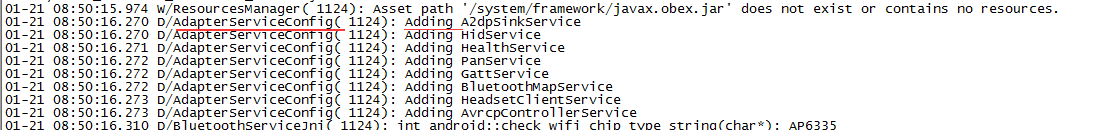
#### Profile配置

下面两个文件中 将对应的 profile项配置为true 即可

device/rockchip/common/overlay/packages/apps/Bluetooth/res/values/config.xml

packages/apps/Bluetooth/res/values/config.xml

配置生效的情况下 可以看到 相关profile service的 adding 日志



### 其他配置

#### 外围模式

处于外围模式的设备可以发布广播，并建立属性服务，中心模式的设备可以搜寻广播；使用场景：ibeacon，外围模式的设备通过发布属性服务，处于中心模式的设备可以搜寻外围模式下的设备与之连接并进行数据交互

Android平台下开启外围 模式请参照 外围模式.7z 压缩包

#### Iphone settings中搜索蓝牙看不到我的蓝牙设备

ios系统的设置的蓝牙搜索界面只显示能够被搜索并且含有ios感兴趣的profile的设备（当前确认的音频类，ble的含有时间服务）；5.1之后下通过启动hfp\_client profile，a2dpsink等 此时能够被ios搜索到；

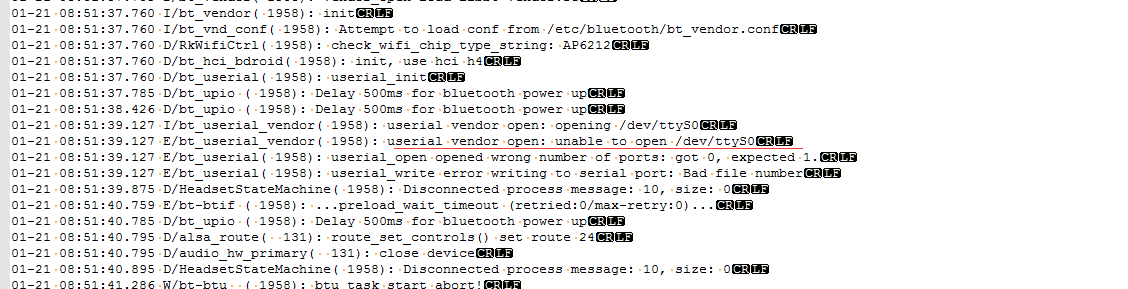
## 调测

### 功能性调测

**蓝牙无法打开类，以android5.1 的 logcat 日志为例**

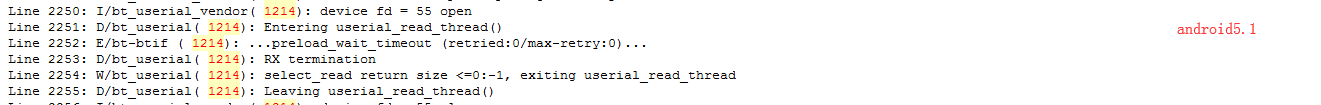
#### 确认串口是否正常被打开

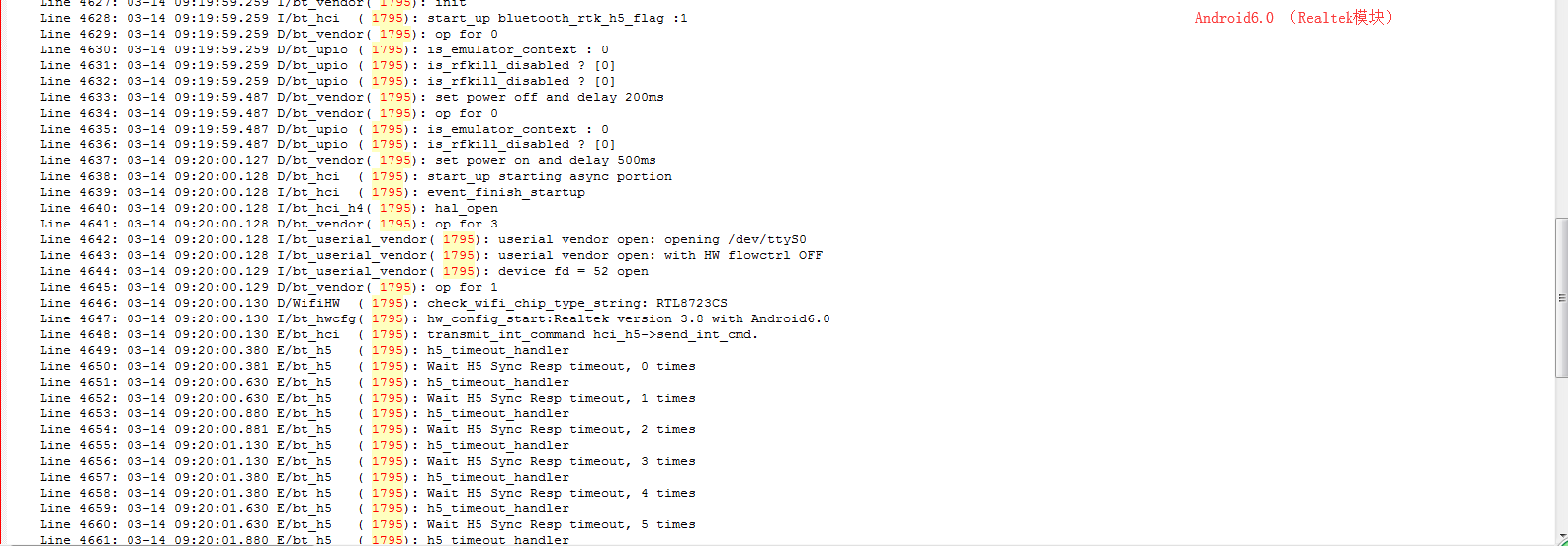
如果看到下方截图的错误信息，请确认内核中是否有正确将对应串口打开（以ttyS0为例），并且可以根据下面的信息确认 实际打开的串口与自己的硬件是否一致

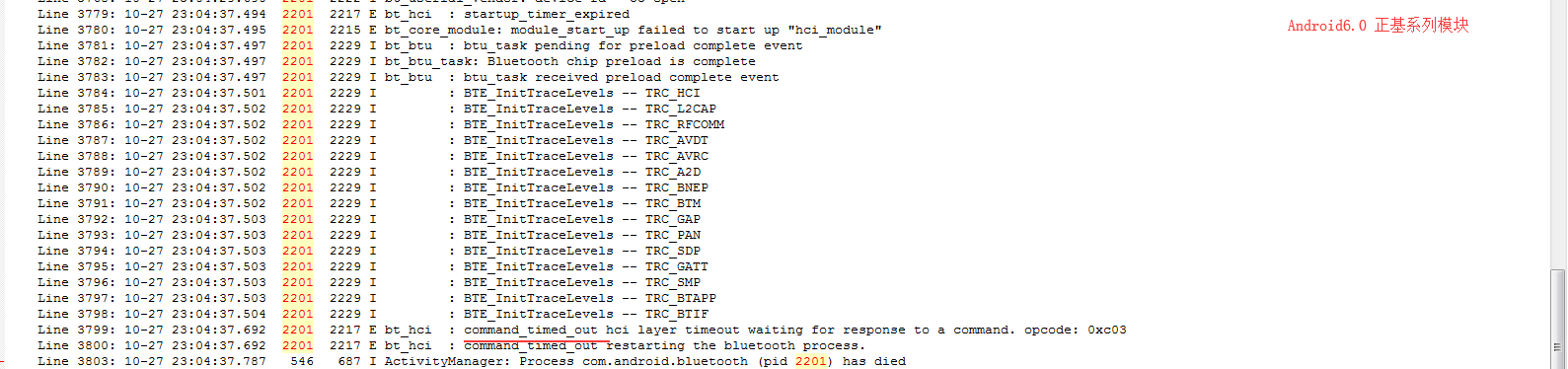


#### 确认模块是否无响应

在logcat日志中看到如下几种情形的日志都是代表模块无响应，







#### 如何调试模块无响应情形

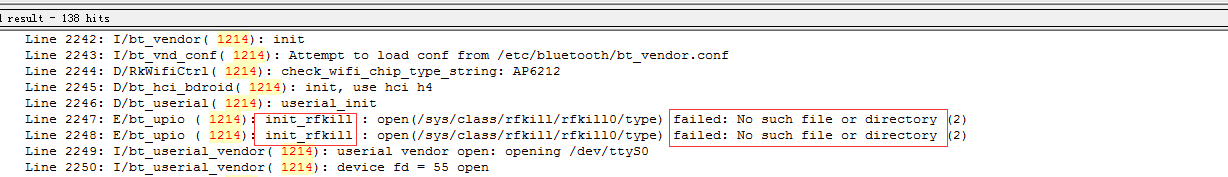
主要做三个方面： 供电控制 串口功能 外部时钟信号排查

##### 供电控制：

logcat 中 看到如下 红框的类似日志，代表 协议栈层 枚举/sys/class/rfkill 下子目录没有找到type 为bluetooth的 rfkill 控制节点

Android下供电控制通过遍历/sys/class/rfkill/目录，查找 type 为bluetooth 的 节点 进行供电控制；

logcat 中日志中看到截图中的日志表示没有找到rfkill 控制节点；



没有控制蓝牙供电控制的rfkill节点，请检查 dts中 wireless-bluetooth 节点是否配置okay，如果已经配置 ok，检查wireless-bluetooth中用到的管脚和哪些地方产生了冲突，通常是由于管脚冲突 导致初始化失败而没有注册；可以通过跟踪net/rfkill-bt.c 中初始化部分的代码确认哪部分管脚申请失败；

蓝牙供电控制rfkill节点存在的情况下，检查确认供电控制是否正常；需要确认 上电时 模块主供电、VDDIO、bt使能脚、bt\_wake脚（bt\_wake输出高电平用于控制让模块始终处于唤醒状态，bt\_wake脚输出的电平为低电平 此时 模块可以自行决定切换到低功耗模式 此时会出现 模块不响应 主控下发的hci命令）

echo 1 > sys/class/rfkill/rfkill0/state //上电

echo 0 > /sys/class/rfkill/rfkill0/state //下电

##### 串口功能确认(以ttyS1为例进行说明)

测试工具 bt\_debug//见bt\_debug.rar 使用方法见 该压缩包中的readme

###### 数据发送测试

执行bt\_debug ，此时 串口的 tx脚示波器可以测量到方波；

如果测试不到方波首先确认管脚服用关系是否正常具体参看Rockchip Pin-Ctrl 开发指南

###### 自收自发测试

2.1 将 tx 与 rx 进行短接

2.2 cat /dev/ttyS1 &

2.3 执行bt\_debug 程序（adb 另外开个终端）此时再 执行cat 命令的窗口看到 大量的字母U，此时说明自收自发正常

###### rts cts 脚信号确认

rts cts 为流控控制脚，在启用流控时 串口通讯是 示波器可以测量到 rts cts 脚 电平变化（高低的变化）；

如果rts cts 信号输出高电平信号时 无法正常拉高 输出低电平信号时 无法拉低到 0V，确认下修改下 该管脚的内部上下拉配置

###### 串口接口上的电压确认

在没有数据通讯是 uart上的 电平为高电平 电压应该和 模块的VDDIO电压一致

【注】 在信号测量异常时 可以通过将对应的管脚通过 将该管脚配置为 gpio ，通过 gpio输出高低电平 来确认硬件上的导通性

###### 波特率确认

波特率方面的确认可以参照rockchip\_uart\_调试指南.pdf

【之前在3188 平台有遇到过 客户板子重启测试过程中会出现蓝牙无法打开，后面测定为串口的时钟源出现异常 导致波特率不准确引发问题，所以在某些情况下有必要确认波特率的准确性】

###### 其他电气信号确认

确认在关机状态 模块的电（主供电 VDDIO ）电压应该是0V（如果有残留电会有问题：有遇到过因为主供电和 VDDIO残留1V多的电导致 模块工作异常），在上电过程中 主供电和 VDDIO会切换到工作电平(检查确认这个上电过程不应该 出现很缓慢的爬坡现象)；

###### 外部其他信号

确保 32k 和外部晶振正常起振；以正基系列模块为例 32k 没有供给的情况下 模块是不响应 主控下发的任何命令的；

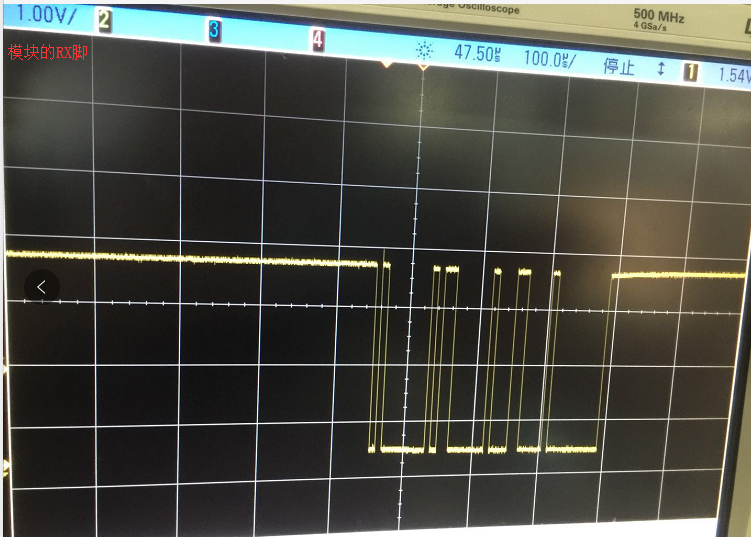
其他请结合所使用模块的规格书检查确认外部信号

模块无响应情形下 上述排查后所有信号正常情况下 logcat 日志依然报告模块无响应的，可以确认下 是否所有板子均如此 如果均如此 可以联系模块厂一起看下；

###### 蓝牙模块初始化的主控下发的第一条指令的波形

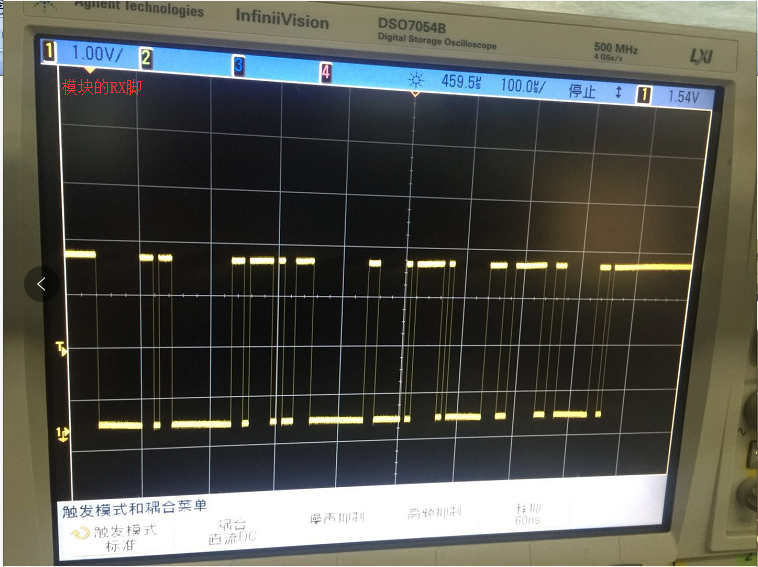
正基系列模块

正基系列在协议栈打开蓝牙后 首先下发reset指令（01 03 0c 00）复位指令，Realtek UART 接口的模块下发的第一条指令为 Sync指令，模块工作正常情况下收到主控下发的指令就会有回应 无回回应我们再logcat上就会看到示波器抓到到的主控下发到串口上的第一条命令数据波形如下图（有遇到过因为蓝牙和调试串口公用，在启动阶段trust.img 会往串口上写入数据，客户只测试到串口上有波形 就误判主控端数据下发正常，确认波形可以进一步判定）



Realtek系列模块

Realtek系列模块打开 蓝牙过程下发的一条指令是 sync 指令 ，波形如下图



##### 其他情形

###### bt\_wake信号

打开蓝牙过程中 bt\_wake 脚输出的 为 低电平，导致模块切换到低功耗模式下，此时也会出现不响应模块命令的，进而看到命令超时的情形；只要确认bt\_wake在打开蓝牙过程中一直是高电平状态即可

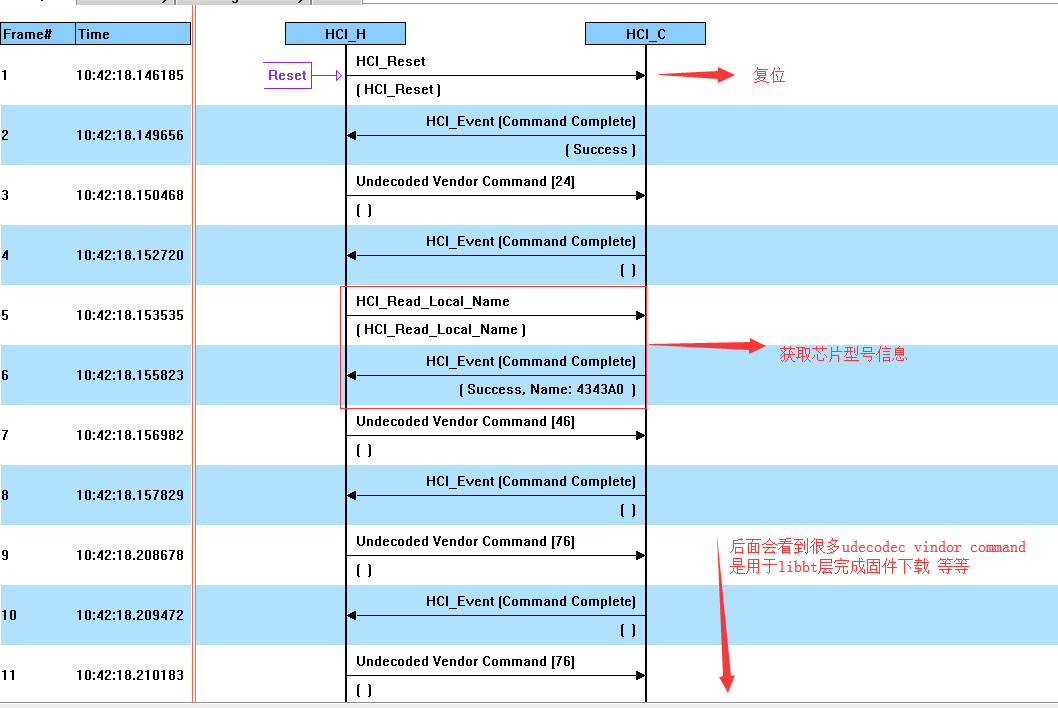
###### Rts信号

正基系列模块中部分模块在bt上电时需要在主控的rts脚输出电平切变信号，如果没有切变信号此时模块也会出现不响应主控下发的命令，确保内核中打开对应的宏控制，控制rts脚输出电平切变信号的代码位于

net/rfkill/rfkill-bt.c中rfkill\_rk\_set\_power 函数

###### 等待蓝牙firmware复位超时

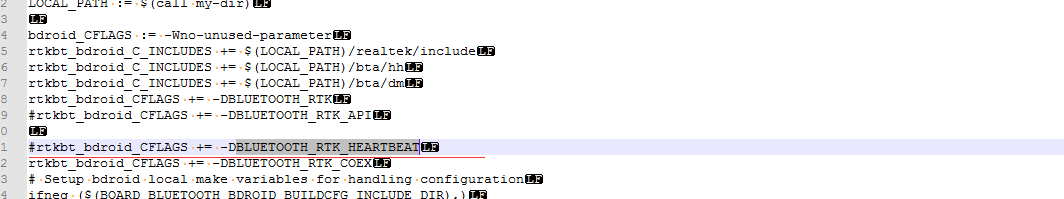
正基系列模块在对模块下载完固件后 libbt层会等待一段时间等待下载到模块中的 firmware运行起来，等待的时间在 libbt/src/hardware.c 中的 fw\_settlement\_table配置；当处于下载完firmware后模块无响应的情形时：此时删除掉要下载的firmware文件，此时蓝牙正常打开（功能可能会存在异常，模块内部自带的firmware功能可能不完善），那么可以通过调节fw\_settlement\_table中的时间 尝试调大或者调小，如果调大调小均无收效建议与模块厂或者RK工程师确认下下载的firmware是否正确；



###### Heartbeat机制

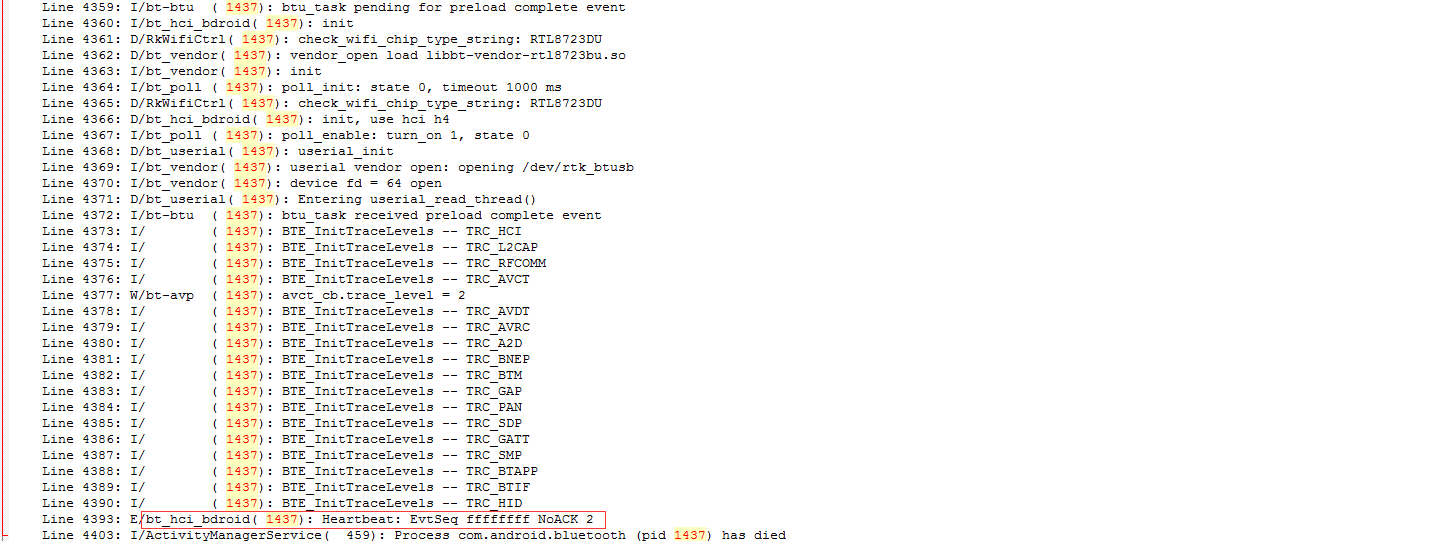
Realtek 系列模块在新版本的协议栈中存在heartbeat机制,其中Android4.4 Android5.1 下是通过 协议栈mk文件中BLUETOOTH\_RTK\_HEARTBEAT 宏进行控制，Android6.0 Android7.1 是通过profile\_supported\_rtkbt来控制；

Android4.4 Android5.1 下确认目前的协议栈中是否带heartbeat可以通过检查协议栈（external/Bluetooth\_rtk/bluedroid/Android.mk文件）是否包含如下截图的内容



R如果蓝牙的firmware与协议栈配置 heartbeat配置情况不一致也会出现打不开的情形：

如下截图的日志（android5.1 下为例 模块firmware层不支持hearbeat 而 协议栈层开启hearbeat打开蓝牙过程中 因为心跳计数异常异常退出）



#### 连接断开类问题

连接类问题，首先确认板子RF部分是否有问题，RF无问题情况下 可以参看下面的说明进行调测

##### 遥控器回连失败类

由于启用随机mac地址（BLE）引发的回连失败

现象：打开蓝牙后与对测的遥控器进行配对，此时对遥控器进行掉电再上电 能够重连成功，如果重新开关蓝牙 此时无法重连成功；查看协议栈日志可以看到LE Set\_Random\_Address

![D:\Documents\Tencent Files\2076130866\Image\C2C\Image5\EJ01T6S0]HWU%4]((OTIOBA.png](data:image/png;base64,)

此种情形问题由于启动随机mac地址，遥控器发出的回连广播中的mac与目前设备的mac不一致导致回连失败，在bt\_target.h 中BLE\_PRIVACY\_SPT配置为FALSE，即可解决;

##### Realtek 系列模块与设备连接后立马断开

Realtek 系列模块因为Hearbeat机制，在Android6.0 Android7.1 下通过

packages/apps/Bluetooth/res/values/config.xml

或

device/rockchip/common/overlay/packages/apps/Bluetooth/res/values/config.xml

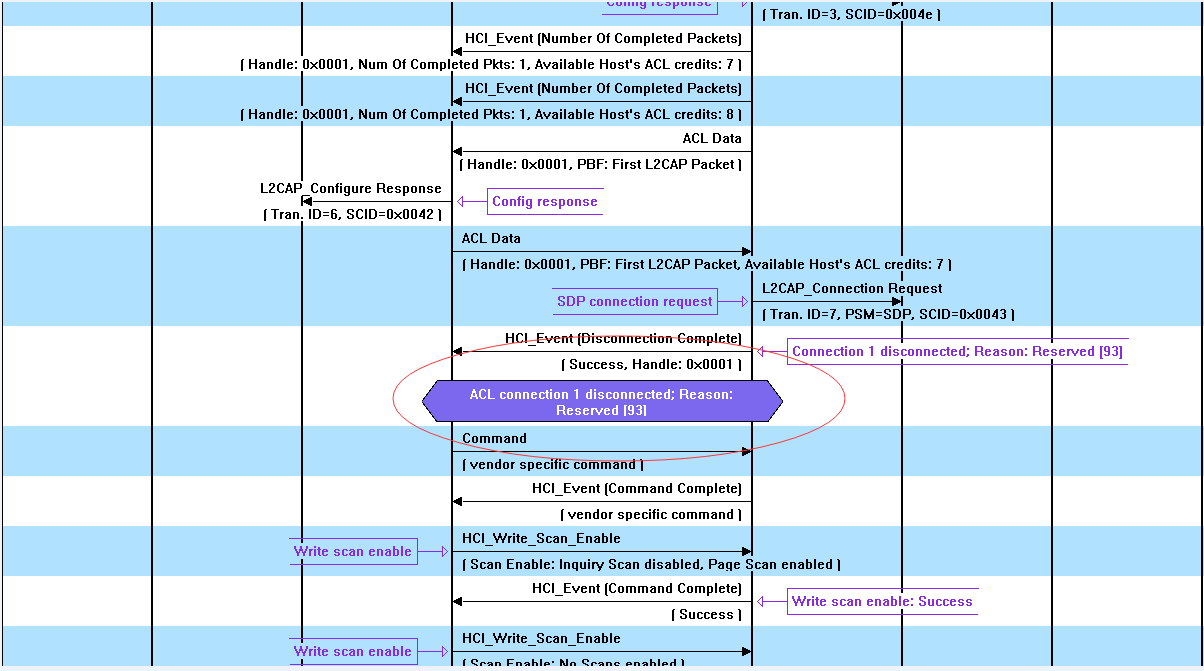
文件中profile\_supported\_rtkbt 进行控制，当配置为true时打开 当配置为false时 关闭

当模块的firmware固件支持hearbeat 而 软件配置上没有打开hearbeat 机制时，此时会出现如下情形：

1：与对测设备配对成功后 会不停的断开 然后再次尝试连接 ；

2 ：与手机配对时，在弹出配对确认窗口时 如果 RK设备端或者手机端点击确认 此时另外一端的窗口会迅速消失

此类问题 在 协议栈日志看断线的错误原因为 Reserved[93] 该错误号为realtek内部保留错误号



#### 蓝牙唤醒问题

主控休眠时 蓝牙可以通过bt\_host\_wake 口（具体的pin脚参看 对应模块的datasheet）唤醒主控，软件配置上确保内核wireless-bluetooth节点中 BT,wake\_host\_irq 配置为与硬件上一致的io口，当蓝牙唤醒功能异常时首先确认蓝牙是否有输出唤醒信号（Realtek系列的 大部分模块的 wifi\_host\_wake bt\_host\_wake 为共用口 需要与 模块厂确认清楚目前firmware和config的状态配置的相关口是bt的host\_wake 口还是 wifi的 host\_wake口）

#### 卡顿类问题

##### 作为source端连接音箱卡顿

如果打开wifi情况下 会出现卡顿 关闭不会那么为 wifibt共存问题 联系模块厂处理即可

当看到如下图截图中的类似打印 排查确认是否wifi bt共存问题，如果非共存问题确认RF 部分是否正常 如果RF正常请联系模块厂即可；

下面的日志表示协议栈通过模块底层向外送数据但是无法及时将数据送出去导致缓存buffer队列一直处于满的状态



其他：连接音箱播放声音卡顿的 需要确认首先本地播放音乐是否存在问题，本地音乐没问题再检查是否系由于其他原因导致协议栈无法及时将数据送出去而出现的卡顿

##### 作为A2dp Sink端做蓝牙音箱功能时卡顿

客户基于RK平台开发智能音箱产品或者类音箱的产品，此时RK平台作为A2dp Sink 端，此类卡顿问题可以做如下确认：

1 关闭wifi是否有同样问题：如果关闭wifi 不存在卡顿 那么 系wifi bt共存问题 联系模块厂即可

2 RF指标是否有做过确认

3 抓取btsnoop 日志 通过frontline 工具 将协议栈中的音频数据export 出来确认 协议栈层的数据是否卡顿，如果协议栈层的数据存在卡顿，可以联系模块厂解决

4 串口部分

可以尝试更改串口波特率，避免因为串口通讯速率过低引发

4 如果部分手机测试会存在卡顿 可以考虑调大 录放线程的 缓存buffer队列的大小（5.1 6.0 下的 录放线程 在 A2dpSinkStateMachine .java 中实现 Android7.1 则 直接在协议栈层调用audiotrack实现）其中协议栈中对收到的数据也有缓存队列（android5.1 下的协议栈是通过下面两个宏控制向上层送数据的，第一宏为 协议栈中缓存队列的大小，第二个宏是表示至少缓存多少数据才会上层送数据，第二个宏调节过大的会带来声音延时的副作用）

MAX\_OUTPUT\_A2DP\_FRAME\_QUEUE\_SZ  
MAX\_A2DP\_DELAYED\_START\_FRAME\_COUNT

#### 蓝牙mac问题

博通系列模初始化过程中在libbt层下HCI\_VSC\_WRITE\_BD\_ADDR命令写入自定义mac地址，realtek系列模块通过替换蓝牙firmware中的config文件中的btmac地址项的内容写入自定义mac地址；libbt-vendor库中如果配置USE\_CONTROLLER\_BDADDR为TRUE 则使用模块自身的mac地址（正基系列模块内部包含mac，realtek系列模块则直接使用config文件带的默认地址）

代码流程如下

协议栈中的 btif\_fetch\_local\_bdaddr 函数获取mac地址（从vlflash或者系统property），将获取到的mac地址传递给libbt-vendor层，libbt-vendor层代码完成对mac地址的初始化

无论使用自定义mac还是模块自带的mac，请确保在mac地址不要重复，如果同一环境下两台设备的蓝牙mac是相同的，此时同时打开蓝牙 和其他设备进行通讯 会出现 错乱！

#### PCBA蓝牙测试问题

Android4.4~Android 7.1 蓝牙部分的pcba测试是通过bdt程序来做，Android8.1 后开始变为通过串口给模块下发复位或者Sync信号检查 模块是否有回应来作为判断依据

Android4.4 到Android7.1 pcba测试失败，首先确认在大固件下执行 bdt enable 是否能够成功，如果大固件下成功，pcba测试固件中失败，做如下检查

1 检查是否有合并bluetooth\_pcba测试补丁（rk\_pcbatest.patch ===》1 解决 pcba 调用的 \_\_popen 函数执行失败问题）

2 bdt依赖的库：协议栈库（bluetooth.default.so bluetooth\_rtk.default.so）以libbt-vendor 层库 （libbt-vendor.so libbt-vendor-rtl8723bs libbt-vendor-rtl8723bu.so）是否有正确集成在pcba测试固件，

3蓝牙的firmware是否有正确集成在pcba测试固件中（集成进的路径要与libbt-vendor层访问的路径一致）

4路径问题（libbt-vendor层有时配置的firmware路径为 /vendor/firmware 该路径实际是/system/etc/firmware下的软链接 但是 在pcba测试固件中 没有该软链接 直接访问 /vendor/firmware 是没有该路径的）；

5 权限问题 检查 要访问的 串口或者/dev/rtkbtusb 权限

Bt\_wake

6 可以单独保留蓝牙的测试项是否依然测试失败，如果单独跑蓝牙测试项是成功的，可以考虑将run\_type 都配置为0 （逐个测试并且等待前一项测试完再测下一项）

如果按照上述调整后依然存在问题 readmine上反馈日志的要点：测试项只保留蓝牙测试项 进行复测抓取日志

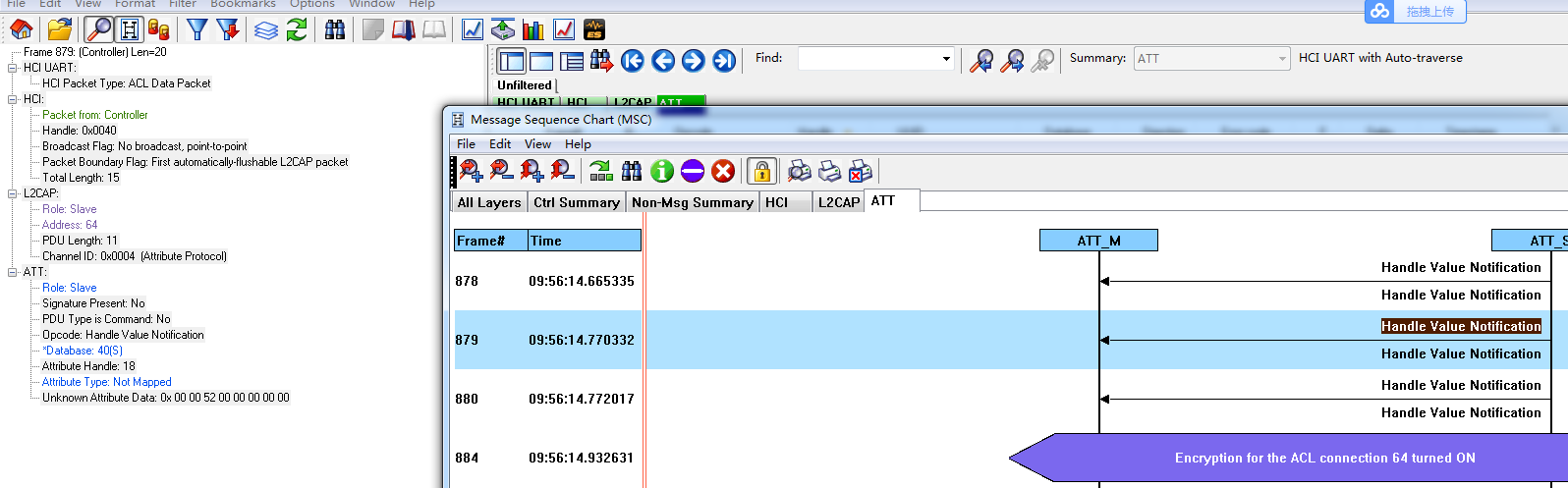
#### 遥控器键值上报丢键值类 debug方法

参看 针对丢键值添加日志方法，测试要求与日志要求说明.txt 添加对应的调试日志，（该调试补丁中会记录ATT Notification 的个数并dump出内容）

调试要点：对比协议栈日志中的notification个数，bluedroid中dump出的notification个数 ，getevent 统计的上报的键值 ，这三个数值 保证 协议栈日志中按键个数与bluedroid中dump出的notification个数一致，并能够与getevent 统计的按键数对应上；因此调试时我们需要记录回连成功后按了多少次按键（按键顺序 次数），完整的logcat日志（需要放开协议栈日志）以及对应的btsnoop日志

之前处理过的案例 3399 Android 6.0 +正基或realtek模块 遥控器在加密完成前上报的键值是是无法被协议栈正确处理的

在协议栈日志中可以看到在加密完成前是有键值上报。



在该案例问题中系 从开始加密链路到链路加密完成太慢 ，遥控器提前发送了键值，最终通过调小连接间隔（调小间隔的优势在于 数据速度通讯更快）；该案例问题中 realtek系列模块单纯通过修改 connect\_interval的数值 无收效 后通过更新协议栈解决问题

BTM\_BLE\_CONN\_INT\_MIN\_DEF

BTM\_BLE\_CONN\_INT\_MAX\_DEF

其他遥控器键值确认是否有丢失的同样借助getevent 来分析确认

#### 蓝牙电话与车载免提功能

蓝牙电话功能：RK平台作为AG端 耳机作为 HF 端：需要打开profile\_supported\_hs\_hfp ，logcat日志中留意HeadsetStateMachine 状态机的变化 确认耳机和RK平台处于正常的连接状态，并确认PCM部分的参数配置符合音频部分的要求

车载免提功能：RK平台作为HF端 手机作为 AG 端：需要打开profile\_supported\_hs\_hfp ，logcat日志中留意HeadsetClientStateMachine状态机的变化 确认手机和RK平台处于正常的连接状态，并确认PCM部分的参数配置符合音频部分的要求；

1. 文件传输类

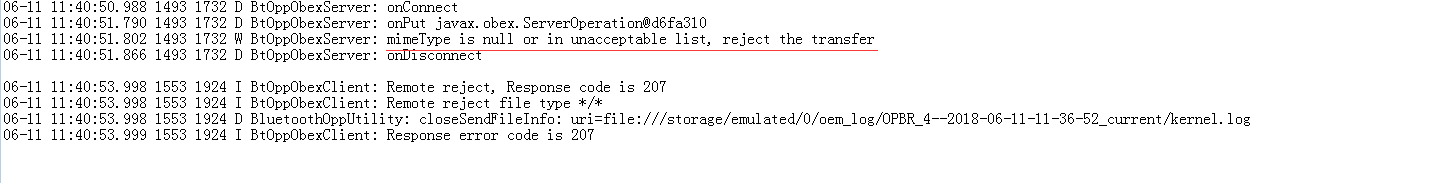
##### 文件格式不支持而传输失败的

支持的传输格式配置见packages/apps/Bluetooth/AndroidManifest.xml

文件中



如果传输的文件不在支持列表导致传输失败 logcat 中可以看到如下日志

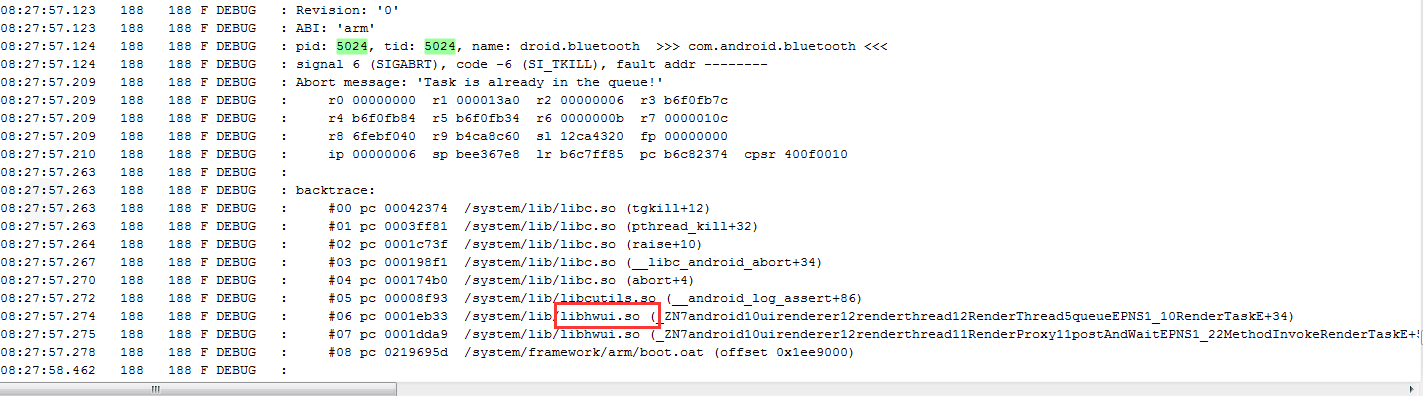


##### 传输文件出现蓝牙服务crashed的

目前处理过两种情形的

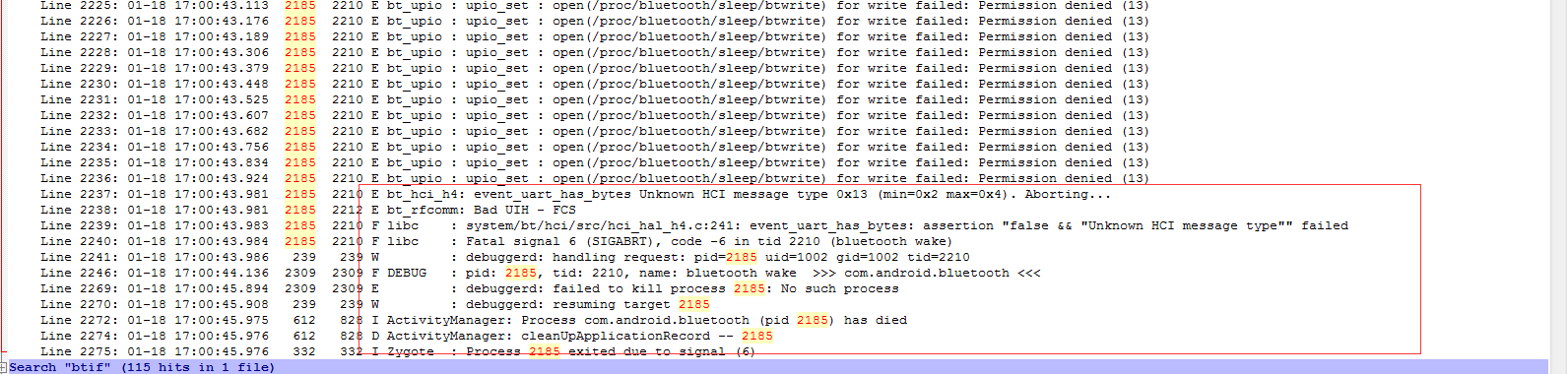
* 情形1 libhwwui引发的

crashed信息中可以看到libhwui的异常点 如果有该部分异常点 请图形部分的工程协助分析



* + 情形2 由于串口上有丢数据引发的crashed

串口上发生丢数据，导致协议栈对收到的数据进行解包 出问题 进而crashed（目前问题发现在 内核版本为4.4版本的平台）如遇到此类文件建议更新内核（该历史问题 福州更新uart部分的驱动解决）



其他类似问题的分析方法

同步抓取btsnoop日志以及 使用协议分析仪抓取整个测试的日志 对比确认btsnoop中的包与协议分析仪中抓取的包

1. 沟通反馈问题注意事项

#### 放开btsnoop日志与协议栈日志的方法

btsnoop 日志开启方法

打开btsnoop 日志的方法：修改 /system/etc/bluetooth/bt\_stack.conf 中将 配置项为false的修改为true 重启即可生效(或者将 开发者选项中将 收集蓝牙hcibtsnoop 日志) 在/sdcard/下 可以看到 btsnoop开头的相关文件

开发所有协议栈日志的方法

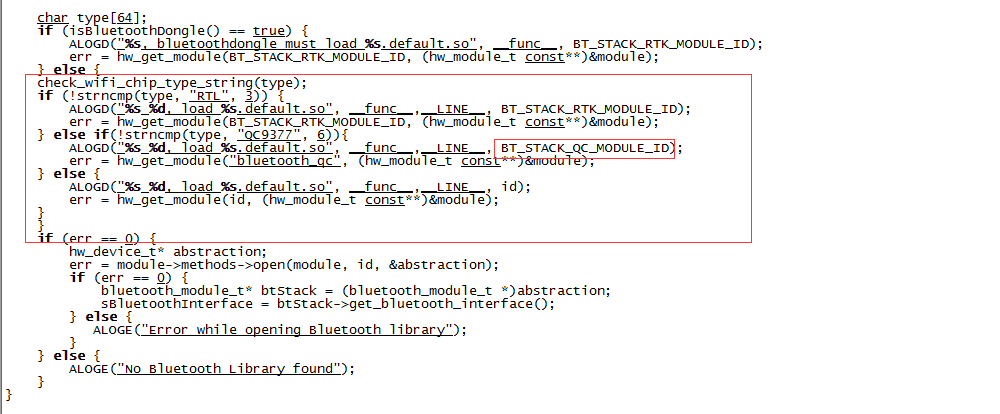
修改 /system/etc/bluetooth/bt\_stack.conf 中将 日志级别为2的修改为7 重启即可生效

## 新模块移植与开发

以android5.1 的代码为示例讲解 其他android 灵活调整

选择要加载协议栈

通常情况下建议使用原生的蓝牙协议栈，如果有厂商的协议栈对原生协议栈做过较大的修正而必须使用厂商自己的协议栈的情况下在jni层完成对不同厂商协议栈的加载



Jni中的mk文件需要增加libhardware\_leacgy的依赖（因为要使用check\_wifi\_chip\_type\_string函数）



选择调用对应厂商的硬件初始化层

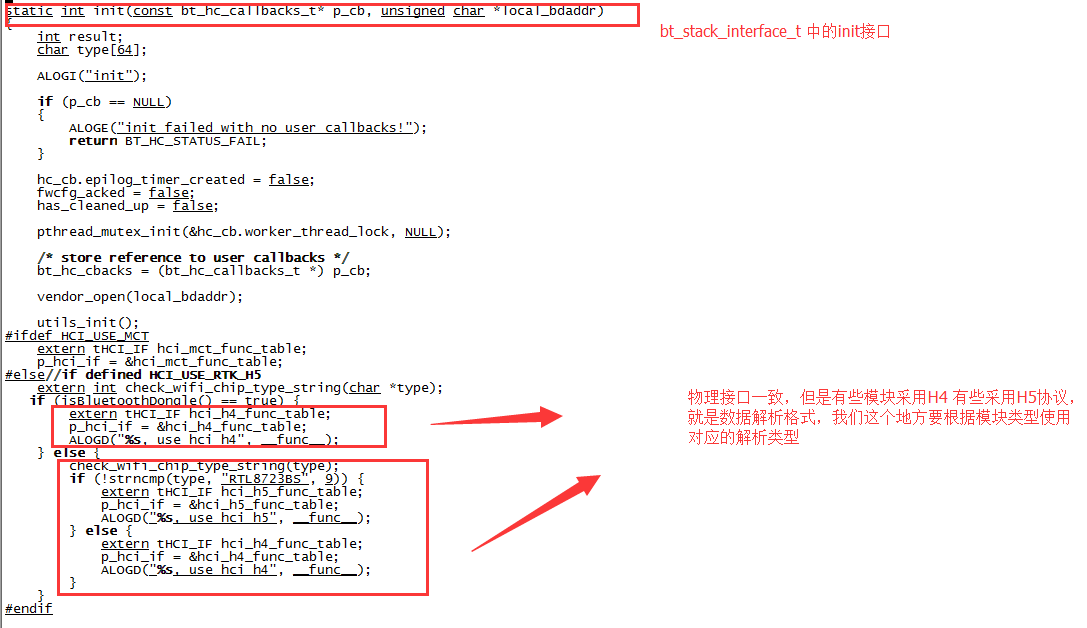
#### 根据模块类型调用厂商的硬件初始化层；

Bluedroid/hci 目录下，由于android版本不同该对应的文件名会有差异，但路径基本都在hci目录下，可以通过搜搜libbt-vendor 相关关键字找到



选择调用对应的hci协议层

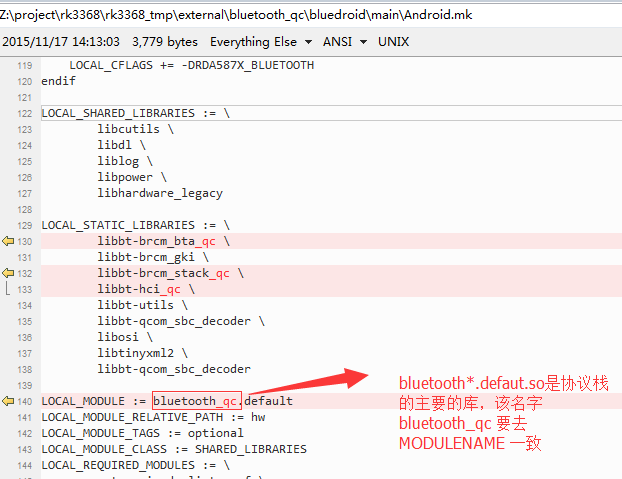
Hci层协议栈层调用；Bluedroid/hci 目录下，由于android版本不同该对应的文件名会有差异，但路径基本都在hci目录下，可以通过搜搜hci\_h4，或者hci\_h5 类似关键字找到

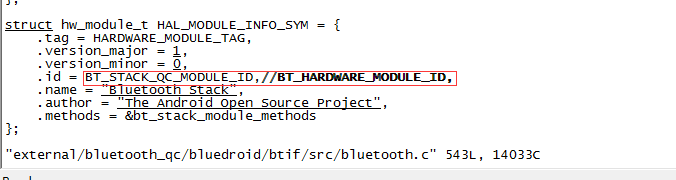


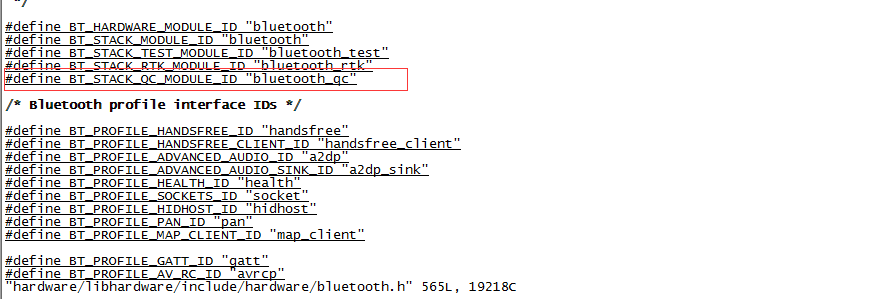
加入新厂商的协议栈的注意事项

更改模块名

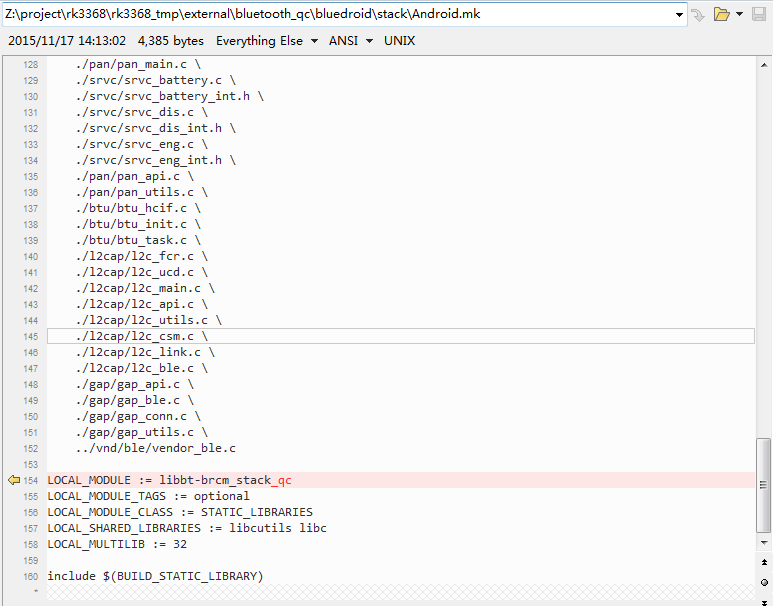
更改协议栈模块的名字同时记得模块厂协议栈依赖的库（模块厂自己生成的）也要改名

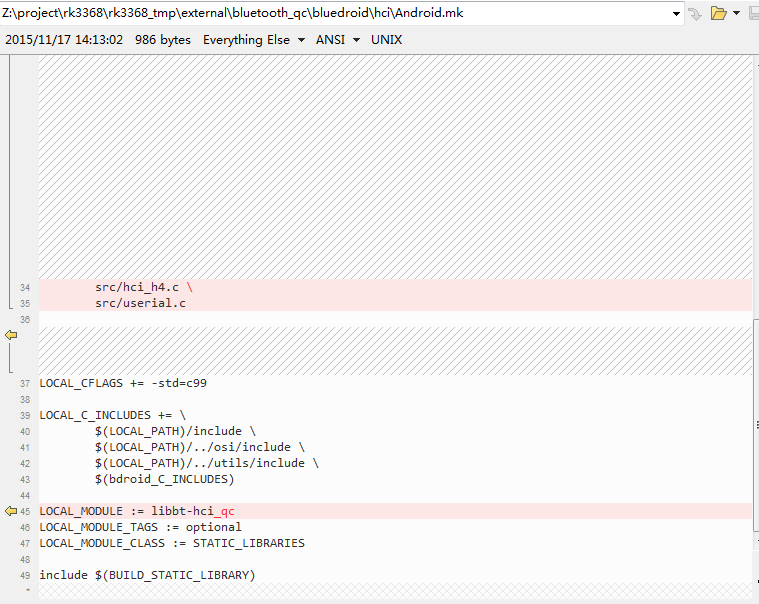


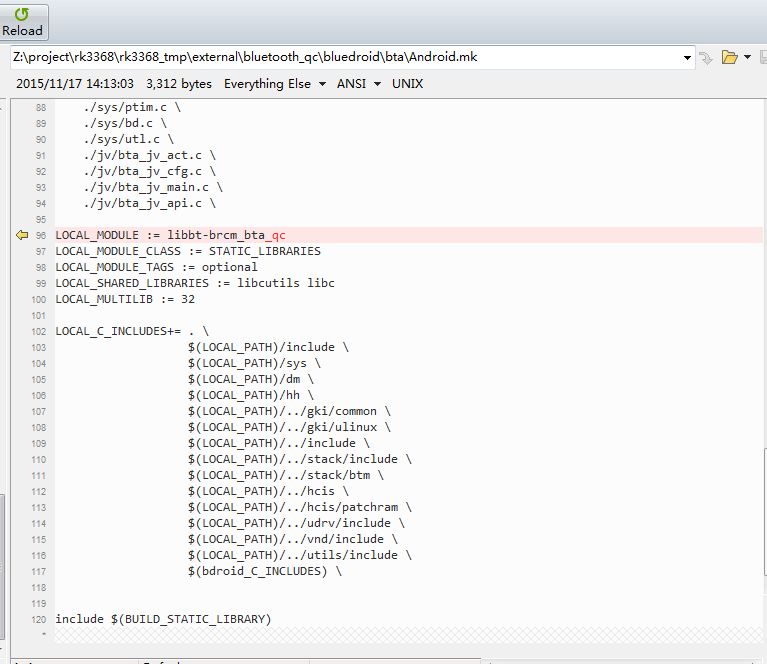




#### 相关其他库改名，比较简单的方法，







其他

其他部分由于不同厂商的模块会有自己厂商的特殊要求，请按照厂商要求进行处理