Realtek Bluetooth Porting Guide for Android6.x

v0.3

2017/03/01

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| data | version | Changelist | Author |
| 2015/12/1 | v0.1 | Draft | Jason\_mao |
| 2016/04/20 | v0.2 | v0.2 | Jason\_mao |
| 2017/03/01 | v0.3 | Move patch/code/linux to linux | Jason\_mao |

Contents

[Revision History 2](#_Toc476239380)

[Contents 3](#_Toc476239381)

[Tables 5](#_Toc476239382)

[Charts 6](#_Toc476239383)

[1. Brief 7](#_Toc476239384)

[2. Release Note 8](#_Toc476239385)

[2.1 Release Note 8](#_Toc476239386)

[2.2 Porting Guide Note 8](#_Toc476239387)

[3. SDK Porting Instruction 10](#_Toc476239388)

[3.1 Android 10](#_Toc476239389)

[3.1.1 Changelist 10](#_Toc476239390)

[3.1.2 build 10](#_Toc476239391)

[3.1.3 device/{vendor}/{product}/ 11](#_Toc476239392)

[3.1.4 Others 18](#_Toc476239393)

[3.1.5 Attentions 18](#_Toc476239394)

[3.2 Kernel 19](#_Toc476239395)

[3.2.1 Change list 19](#_Toc476239396)

[3.2.2 rtk\_btusb driver 19](#_Toc476239397)

[3.2.3 rtk\_rfkill driver 20](#_Toc476239398)

[3.2.4 TUN driver 20](#_Toc476239399)

[3.2.5 UINPUT driver 21](#_Toc476239400)

[3.2.6 HID driver 21](#_Toc476239401)

[4. BT Function Configuration 22](#_Toc476239402)

[4.1 Supported Profiles 22](#_Toc476239403)

[4.2 Configure Local Device Name 22](#_Toc476239404)

[4.3 Configure COD 22](#_Toc476239405)

[5. Basic Test After Porting 24](#_Toc476239406)

[5.1 Config Test 24](#_Toc476239407)

[5.2 BT Test 24](#_Toc476239408)

[5.2.1 Basic Function Test 24](#_Toc476239409)

[6. Debug And Log config 25](#_Toc476239410)

[6.1 Bluetooth Log Configuration 25](#_Toc476239411)

[6.2 Logcat –v time 26](#_Toc476239412)

[6.3 BtSnoop 26](#_Toc476239413)

[6.4 Attentions 26](#_Toc476239414)

[7. Common Problem Analysis 27](#_Toc476239415)

[7.1 BT Open Failure(UART) 27](#_Toc476239416)

[7.2 BT Open Failure (USB) 27](#_Toc476239417)

Tables

Charts

# Brief

This document applies to Realtek Bluetooth Solution for Android 6.x products.

# Release Note

## Release Note

Release package is divided into linux, rtkbt, patch directories and Porting Guide, ReleaseNotes document.

* Linux is Linux Kernel driver provided by Realtek, only applied for USB interface Bluetooth solution
* Rtkbt is the main Realtek released driver file, please directly copy it to device/**{vendor}**/ **{product}**/rtkbt directory.
* Patch is the patch file for Android, which includes code part and diff part. Code directory contains modified files and diff directory contains modified differences. Pay attention to the file differences when porting. Because there are exists differences in original files, please do not directly replace it.
* Porting Guide is this document, mainly describes the use of Realtek driver package.
* ReleaseNotes lists main modification of driver package.

## Porting Guide Note

* Conventions used in this document：

1. All code modified or added by realtek are marked in boxes.
2. All code modified or added by realtek are highlighted in gray.
3. All code in original SDK use normal color in boxes.

Code format example：

Original code in SDK：

|  |
| --- |
| 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 |

Code modified for support Realtek UART H5：

|  |
| --- |
| 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 |

Code modified by realtek is marked in gray.

* In this guide, we take **{product}** as phoenix, **{vendor}** as realtek for example, you may modify files according to your platform and vendor to support Realtek BT Chip. Different customers have different vendor and platform name.

# SDK Porting Instruction

In order to integrate Realtek Wifi/BT combo Chip into your platform, we provide guides for customers to merge Realtek BT driver into their SDK in Android part and Linux part.

## Android

### Changelist

1. **build**
2. **device**/**{vendor}/{product}/**
3. **others**
4. **attentions**

### 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 \  … |

Add macro BOARD\_HAVE\_BLUETOOTH\_RTK to support Realtek BT Chip. And [delete](http://cn.bing.com/dict/search?q=Delete&FORM=BDVSP6&mkt=zh-cn) [or](http://cn.bing.com/dict/search?q=or&FORM=BDVSP6&mkt=zh-cn) [comment](http://cn.bing.com/dict/search?q=comment&FORM=BDVSP6&mkt=zh-cn) out other Macros such as BOARD\_HAVE\_BLUETOOTH\_BCM, BLUETOOTH\_HCI\_USE\_MCT in case of compiling conflict.

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

This directory is mainly aimed to configure different hardware platform board. Different subdirectories correspond to different hardware platforms, which need to be modified according to the specific hardware platform.

1. Copy rtkbt directory to device/{vendor}/{product}/

If BT interface is UART, then modify device/{vendor}/{product}/rtkbt/system/etc/bluetooth/rtkbt.conf

, which sets BtDeviceNode as the related Uart device node such as /dev/ttyS1.

If BT interface is USB, then keep BtDeviceNode as /dev/rtk\_btusb.

1. Add device permissions

Take USB interface as example, when BtDeviceNode is /dev/rtk\_btusb.

* Add the item in file\_contexts under device dir:

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

* Add the item in device.te:

type rtk\_bt\_device, dev\_type;

* Add the item in bluetooth.te under device dir (if not exist, check in external/sepolicy dir):

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

1. PRODUCT\_MAKEFILES

In **device/{vendor}/{product}/****AndroidProducts.mk** to look for correct PRODUCT\_MAKEFILES:

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

rtk\_phoenix.mk：

Noted: it should replace correct {vendor} and {product} based on actual scenario

|  |
| --- |
| ............  $(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  ............ |

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

In order to reduce the coupling between different products under vendor directory，we recommend to add judgement for product in Android.mk, for example, when compile the SDK with lunch command, the results are as follows:

|  |
| --- |
| $ lunch rtk\_phoenix-eng  ……  PLATFORM\_VERSION=5.1.1  TARGET\_PRODUCT=rtk\_phoenix  TARGET\_BUILD\_VARIANT=eng  …… |

We recommend add modification in Android.mk:

|  |
| --- |
| 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

Noted: make sure no repeat in the process of addition, if there exist the same content in the original file, delete it then add.

* + For BT Usb interface, modify device/**{vendor}/{product}/**init.**{product}**.rc as :

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

* + For BT UART interface, modify device/**{vendor}/{product}/**init.**{product}**.rc as :

Attention: {UART Device Node} should be replaced by the specific platform Uart device node such as /dev/ttyS1, /dev/ttyS2.

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 Device Node}

chown bluetooth net\_bt\_stack {UART Device Node }

# 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. **Just modify device/{vendor}/{product}/ueventd.{product}.rc only for BT USB interfcace.**

Add the following in the end of file

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

### Others

Refer to the patch directory of the Release package.

Patch is the patch file for Android, which includes code part and diff part. Code directory contains modified files and diff directory contains modified differences. Pay attention to the file differences when porting. Because there are exists differences in original files, please do not directly replace it.

### Attentions

As many platforms have been integrated with other Bluetooth solutions, the first thing of porting driver mayremove Bluetooth solutions in the SDK, the simplest way of which is as follows:

1. Set BOARD\_HAVE\_BLUETOOTH\_XXX as false or just delete it in BoardConfig.
2. Use command: 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/\* ” . for the purpose of deleting related files before start compiling.
3. Before start porting, compile Android and check there is no bluetooth.default.so under out/target/product/{product} /system/lib/hw/ . If bluetooth.default.so exists, you need to find the corresponding bluedroid source code location , and modify the Android.mk under bluedroid as below.

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

…………

#orignal Makefile

…………

endif

1. And then go back to step 2,3 until the bluetooth.default.so is not generated any more after compile.
2. After step 4, start porting Realtek BT driver.
3. After porting, please pay attention to the compiled /system/app directory to check whether BCM or other manufacturers of Bluetooth applications exsit or not. And move them if there exists problem in normal operation.

## Kernel

### Change list

Chg kernel/driver/bluetooth/Kconfig

Chg kernel/driver/bluetooth/Makefile

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

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

### rtk\_btusb driver

1. Copy Realtek rtk\_btusb.h,rtk\_btusb.c to linux kernel’s /drivers/bluetooth/ directory.
2. Modify “Kconfig” and “Makefile” in linux kernel’s /drivers/bluetooth/ directory.

Add BT\_RTKBTUSB items as below in the Kconfig：

config BT\_RTKBTUSB

tristate "RTK HCI USB driver"

depends on USB

help

RTK Bluetooth HCI USB driver

Add rtk\_btusb.o into the Makefile:

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

1. Select rtk\_btusb driver in kernel make menuconfig.

### rtk\_rfkill driver

**Notes：**

**It needs rfkill driver while you need BT\_DIS pin to control BT chip.**

**Generally ,the platform manufactures have already fulfilled the rfkill dirver, You may just apply for the RFKILL\_TYPE\_BLUETOOTH rfkill node for Bluetooth to configure the related rfkill GPIO as BT RESET\_PIN. Please kindly contact us if any question.**

How to check if rfkill driver support BT exist:

cat /sys/class/rfkill/rfkill\*/type, if return “bluetooth”, it means rfkill driver existed.

### TUN driver

If it need to support Bluetooth PAN, make sure TUN Driver is compiled into kernel

kernel\arch\arm\configs\XXX\_defconfig

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

### UINPUT driver

When AVRCP is needed, config UINPUT driver as below:

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

### HID driver

When Bluetooth HID is needed, config uhid driver.

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

Note: some HID devices can be connected, but can not be used normally, mostly because of this configuration does not result in compatibility issues,so Please Make sure all the CONFIG\_HID\_xxx settings are set to y.

# BT Function Configuration

## Supported Profiles

If customer product do not support PBAP HFP and HSP, use the following configuration, set true when need support, set false when need not support.

packages/apps/Bluetooth/res/values/config.xml as follows:

|  |
| --- |
| <resources>  <bool name="profile\_supported\_rtkbt">true</bool>  <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> |

## Configure Local Device Name

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

set BTM\_DEF\_LOCAL\_NAME as default device name when rtkbt.conf exists, if not, Set ro.product.model property as device name.

## Configure COD

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

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

Three parts means DevClassServiceClass, DevClassMajorClass, DevClassMinorClass separately.

Default configuration as:

DevClassServiceClass=0x1A

DevClassMajorClass=0x01

DevClassMinorClass=0x1C

COD as {0x1A,0x01,0x1C}

# Basic Test After Porting

## Config Test

To ensure that there is no problem after porting, Please kindly verify that the fw and config files are present before testing:

* adb shell.
* Check system/etc/firmware/rtlxxxx\_fw rtlxxxx\_config (xxxx as BT Chip type).
* Check if the files under rtkbt/ of release package already merged into correct directory in SDK.

## BT Test

***Notes: This is a fast Bluetooth function test to verify that Realtek driver has been portded into your platform successfully. The test is only to verify some basic function. You should not take the test result as a formal test report. And if you don’t use Realtek BT chip, the test procedure will be not needed.***

### Basic Function Test

1. Turn On/Off BT success.
2. Search nearby devices which are discoverable.
3. Pair and unpair with device successfully.
4. Connect to Bluetooth headset, listen music with A2DP profile.
5. Connect to Bluetooth headset, make a call and talk with Bluetooth HFP/HSP.
6. Transfer files to remote device which supports OPP server, and transfer files from remote device which supports OPP client to local device.
7. Connect Bluetooth HID device (Mouse or Keyboard), Mouse and keyboard can work successfully.

# Debug And Log config

When Bluetooth problem occur, the related logs need to be captured as below to analyze and fix it.

## Bluetooth Log Configuration

Bluetooth related configuration files are stored in the device's /system/etc/bluetooth/bt\_stack.conf which can be modified by adb. The default configuration file is as follows:

|  |
| --- |
| # 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  #begin:modified for BQB  #BQB Certification  CertConf=FFFFFFFF  #end:modified for BQB  # open Heartbeat, RtkbtLogFilter = 1,don't show heartbeat packet in btsnoop  RtkbtLogFilter=1  # 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

add the "-v time" option in logcat to address the problem with the corresponding timepoint.

And you may config the log output file in the device to be exported to the PC before the test.

## BtSnoop

BtSnoopLogOutput set as TRUE, btsnoop will exist in the BtSnoopFileName dir.

## Attentions

**Please kindly provide at least the following information :**

* **logcat –v time.**
* **btsnoop.**
* **problem timestamp.**
* **reproduce steps.**

# Common Problem Analysis

## BT Open Failure(UART)

If the H5 SYNC fails, you need to check whether the hardware circuit is correct (Power Supply, BT Reset PIN, UART TX / RX, CTS / RTS). Then check the card efuse, measured with the oscilloscope UART waveform to see whether the Host to send the correct data to the Controller.

If H5 SYNC succeeds, then the next step is to check whether the Baudrate modification is successful. If the Baudrate modification fails, it is necessary to check whether the Host supports the baud rate and the baud rate is correctly set in the config file.

If the Baudrate modification succeeds, the next step is to download the fw and config files. If the Command Complete Event is not received by the Controller after the download is complete, check whether the fw and config file are correct and the BT Reset PIN is high.

If the download of fw and config file is successful, the next step is to config the hw flow control according to the config file. Then bluedroid stack will send the first HCI Command.

If the first HCI Command has always been retransmitted , you need to check whether the Host UART driver supports HW Flow control.

## BT Open Failure (USB)

* Check the device node :

Check whether "dev / bus / usb" exists in the Bluetooth logcat.

* Check the Kernel log :

UsbDevice [mName = / dev / bus / usb / 002/002, mVendorId = 3034, mProductId = 46880 , MClass = 239, mSubclass = 2.

mVendorId and mProductId is not corresponding to the current use of Bluetooth chips. If no Bluetooth Usb card is not recognized, you need to first check the hardware circuit.

* Check USB driver：

Login to the platform (adb shell), and then use the command lsmod to see whether rtk\_btusb.ko exists.