

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

Realtek

Contents

Revision History	2
Contents	3
Tables	5
Charts	6
1. Brief	7
2. Release Note	8
2.1 Release Note	8
2.2 Porting Guide Note	8
3. SDK Porting Instruction	10
3.1 Android	10
3.1.1 Changelist	10
3.1.2 build	10
3.1.3 device/{vendor}/{product}/	11
3.1.4 Others	18
3.1.5 Attentions	18
3.2 Kernel	19
3.2.1 Change list	19
3.2.2 rtk_btusb driver	19
3.2.3 rtk_rfkill driver	20
3.2.4 TUN driver	20
3.2.5 UINPUT driver	21
3.2.6 HID driver	21
4. BT Function Configuration	22
4.1 Supported Profiles	22
4.2 Configure Local Device Name	22
4.3 Configure COD	22
4.4 Configure Extra Config	23
5. Basic Test After Porting	24

5.1 Config Test	24
5.2 BT Test.....	24
5.2.1 Basic Function Test	24
6. Debug And Log config.....	25
6.1 Bluetooth Log Configuration.....	25
6.2 Logcat –v time	26
6.3 BtSnoop.....	26
6.4 Attentions	26
7. Common Problem Analysis.....	27
7.1 BT Open Failure(UART).....	27
7.2 BT Open Failure (USB).....	27

Tables

Realtek

Charts

Realtek

1. Brief

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

Realtek

2. Release Note

2.1 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.

2.2 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/serial_mct.c
else
LOCAL_SRC_FILES += \
```



```
src/hci_h4.c \  
src/serial.c  
endif
```

Code modified for support Realtek UART H5:

```
ifeq ($(BLUETOOTH_HCI_USE_MCT),true)  
LOCAL_CFLAGS := -DHCI_USE_MCT  
LOCAL_SRC_FILES += \  
    src/hci_mct.c \  
    src/serial_mct.c  
else  
ifeq ($(BLUETOOTH_HCI_USE_RTK_H5),true)  
LOCAL_CFLAGS := -DHCI_USE_RTK_H5  
LOCAL_SRC_FILES += \  
    src/hci_h5.c \  
    src/serial.c \  
    src/bt_skbuff.c \  
    src/bt_list.c  
else  
LOCAL_SRC_FILES += \  
    src/hci_h4.c \  
    src/serial.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.

3. 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.

3.1 Android

3.1.1 Changelist

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

3.1.2 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 or comment out other Macros such as `BOARD_HAVE_BLUETOOTH_BCM`, `BLUETOOTH_HCI_USE_MCT` in case of compiling conflict.

3.1.3 `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`.

Note: The Realtek driver will change the UART baud rate to 1.5M by default. If you use the Amlogic platform, it is recommended to use the 2M baud rate. Use method:

Please overwrite the file under `rtkbt/system/etc/firmware/UART 2M/` into the `rtkbt/system/etc/firmware/` directory.

If BT interface is USB, then keep `BtDeviceNode` as `/dev/rtk_btusb`.

- 2) 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;
```

3) 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
```

.....

4) 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
```

5) 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/dhccpd -n
```

```
class main
```

```
disabled
```

```
oneshot
```

6) Just modify device/{vendor}/{product}/ueventd.{product}.rc only for BT USB interface.

Add the following in the end of file

```
/dev/rtk_btusb          0660    bluetooth    net_bt_stack
```

3.1.4 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.

3.1.5 Attentions

As many platforms have been integrated with other Bluetooth solutions, the first thing of porting driver may remove 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/*/*lueetooth* 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 bluebird source code location , and modify the Android.mk under bluebird as below.

```
ifneq ($(BOARD_HAVE_BLUETOOTH_RTK),true)

.....

#original Makefile

4) .....

5) endif
```

- 6) After porting, please pay attention to the compiled /system/app directory to check whether BCM or other manufacturers of Bluetooth applications exist or not. And move them if there exists problem in normal operation.

3.2 Kernel

3.2.1 Change list

Chg	kernel/driver/bluetooth/Kconfig
Chg	kernel/driver/bluetooth/Makefile
New	kernel/driver/bluetooth/rtk_btusb.c
New	kernel/driver/bluetooth/rtk_btusb.h

3.2.2 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
```

- 3) Select rtk_btusb driver in kernel make menuconfig.

3.2.3 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.

3.2.4 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
```

3.2.5 UINPUT driver

When AVRCP is needed, config UINPUT driver as below:

```
CONFIG_INPUT_UINPUT=y    # User level driver support  
CONFIG_INPUT_MISC=y
```

3.2.6 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.

4. BT Function Configuration

4.1 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_a2dp">true</bool>
    <bool name="profile_supported_a2dp_sink">false</bool>
    <bool name="profile_supported_hdp"> false </bool>
    <bool name="profile_supported_hs_hfp"> false </bool>
    <bool name="profile_supported_hfpclient">false</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>
```

4.2 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.

4.3 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}

4.4 Configure Extra Config

If you need configure extra config, you need new a file named “rtk_btconfig.txt” in /data/misc/bluetooth/, and modify file permissions to 644. **Please be sure to find FAE to review.** Take rtl8723bs as an example:

```
rtl8723bs_conf
0x5b 0x01 0x04 0x21 0x22 0x22 0x21
0xe6 0x01 0x01 0x20
#0xbb 0x01 0x01 0x3c
#0xed 0x00 0x01 0x00
~
```

- 1) The first line must be the name of the config file to be configured.
- 2) Starting from the second line, each line configure one offset and values. Format is offset(2Bytes) + length(1byte) + value(lengthBytes); Little-endian, Separate each byte(Hex) with a space. The setting in the second line of the figure is: offset:0x015b length:0x04 value:0x21222221
- 3) Support for single-line comments with “#”
- 4) Not support configure MAC

5. Basic Test After Porting

5.1 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.

5.2 BT Test

Notes: This is a fast Bluetooth function test to verify that Realtek driver has been ported 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.

5.2.1 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.

6. Debug And Log config

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

6.1 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

# 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
```

6.2 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.

6.3 BtSnoop

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

6.4 Attentions

Please kindly provide at least the following information :

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

7. Common Problem Analysis

7.1 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 bluebird 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.

7.2 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.