FC2x&FC06E&FC6xE&FC900E&FGE576Q
Third-Party Android Bluetooth User Guide

Wi-Fi/Bluetooth Module Series

Version: 1.0.0

Date: 2023-08-12

Status: Preliminary

At Quectel, our aim is to provide timely and comprehensive services to our customers. If you require any assistance, please contact our headquarters:

Quectel Wireless Solutions Co., Ltd.
Building 5, Shanghai Business Park Phase III (Area B), No.1016 Tianlin Road, Minhang District, Shanghai 200233, China
Tel: +86 21 5108 6236
Email: info@quectel.com

Or our local offices. For more information, please visit:
http://www.quectel.com/support/sales.htm.

For technical support, or to report documentation errors, please visit:
http://www.quectel.com/support/technical.htm.
Or email us at: support@quectel.com.

Legal Notices

We offer information as a service to you. The provided information is based on your requirements and we make every effort to ensure its quality. You agree that you are responsible for using independent analysis and evaluation in designing intended products, and we provide reference designs for illustrative purposes only. Before using any hardware, software or service guided by this document, please read this notice carefully. Even though we employ commercially reasonable efforts to provide the best possible experience, you hereby acknowledge and agree that this document and related services hereunder are provided to you on an “as available” basis. We may revise or restate this document from time to time at our sole discretion without any prior notice to you.

Use and Disclosure Restrictions

License Agreements

Documents and information provided by us shall be kept confidential, unless specific permission is granted. They shall not be accessed or used for any purpose except as expressly provided herein.

Copyright

Our and third-party products hereunder may contain copyrighted material. Such copyrighted material shall not be copied, reproduced, distributed, merged, published, translated, or modified without prior written consent. We and the third party have exclusive rights over copyrighted material. No license shall be granted or conveyed under any patents, copyrights, trademarks, or service mark rights. To avoid ambiguities, purchasing in any form cannot be deemed as granting a license other than the normal non-exclusive, royalty-free license to use the material. We reserve the right to take legal action for noncompliance with abovementioned requirements, unauthorized use, or other illegal or malicious use of the material.

Trademarks

Except as otherwise set forth herein, nothing in this document shall be construed as conferring any rights to use any trademark, trade name or name, abbreviation, or counterfeit product thereof owned by Quectel or any third party in advertising, publicity, or other aspects.

Third-Party Rights

This document may refer to hardware, software and/or documentation owned by one or more third parties ("third-party materials"). Use of such third-party materials shall be governed by all restrictions and obligations applicable thereto.

We make no warranty or representation, either express or implied, regarding the third-party materials, including but not limited to any implied or statutory, warranties of merchantability or fitness for a particular purpose, quiet enjoyment, system integration, information accuracy, and non-infringement of any third-party intellectual property rights with regard to the licensed technology or use thereof. Nothing herein constitutes a representation or warranty by us to either develop, enhance, modify, distribute, market, sell, offer for sale, or otherwise maintain production of any our products or any other hardware, software, device, tool, information, or product. We moreover disclaim any and all warranties arising from the course of dealing or usage of trade.

Privacy Policy

To implement module functionality, certain device data are uploaded to Quectel's or third-party's servers, including carriers, chipset suppliers or customer-designated servers. Quectel, strictly abiding by the relevant laws and regulations, shall retain, use, disclose or otherwise process relevant data for the purpose of performing the service only or as permitted by applicable laws. Before data interaction with third parties, please be informed of their privacy and data security policy.

Disclaimer

a) We acknowledge no liability for any injury or damage arising from the reliance upon the information.
b) We shall bear no liability resulting from any inaccuracies or omissions, or from the use of the information contained herein.
c) While we have made every effort to ensure that the functions and features under development are free from errors, it is possible that they could contain errors, inaccuracies, and omissions. Unless otherwise provided by valid agreement, we make no warranties of any kind, either implied or express, and exclude all liability for any loss or damage suffered in connection with the use of features and functions under development, to the maximum extent permitted by law, regardless of whether such loss or damage may have been foreseeable.
d) We are not responsible for the accessibility, safety, accuracy, availability, legality, or completeness of information, advertising, commercial offers, products, services, and materials on third-party websites and third-party resources.

About the Document

Revision History

Version	Date	Author	Description
-	2023-08-12	Lidong ZHOU	Creation of the document
1.0.0	2023-08-12	Lidong ZHOU	Preliminary

Contents

Table Index

Figure Index

1 Introduction

This document takes Quectel SG368Z ARM development board as an example to describe how to enable and verify Bluetooth feature with Quectel FC2x, FC06E, FC6xE, FC900E, and FGE576Q modules on the third-party Android platform.

1.1. Applicable Modules

Table 1: Applicable Modules

Module Family	Mudule	Git Library
FC2x	FC20 series	https://git-master.quectel.com/wifi.bt/fc2x
	FC21	https://git-master.quectel.com/wifi.bt/fc2x
-	FC06E	https://git-master.quectel.com/wifi.bt/fc6xe
FC6xE	FC64E	https://git-master.quectel.com/wifi.bt/fc6xe
	FC65E	https://git-master.quectel.com/wifi.bt/fc6xe
	FC66E series	https://git-master.quectel.com/wifi.bt/fc6xe
-	FC900E	https://git-master.quectel.com/wifi.bt/fc2x
-	FGE576Q	https://git-master.quectel.com/wifi.bt/qcc207x

2 Environment Preparations

2.1. Hardware Environment

Table 2: Hardware Environment

Hardware	Quantity
Quectel SG368Z EVB	1
Quectel Bluetooth module	1
Antenna	1
USB Type C Cable	1
Power Cable	1

NOTE

The development board model is for reference only. If you use other models, please consult the corresponding manufacturer for details.

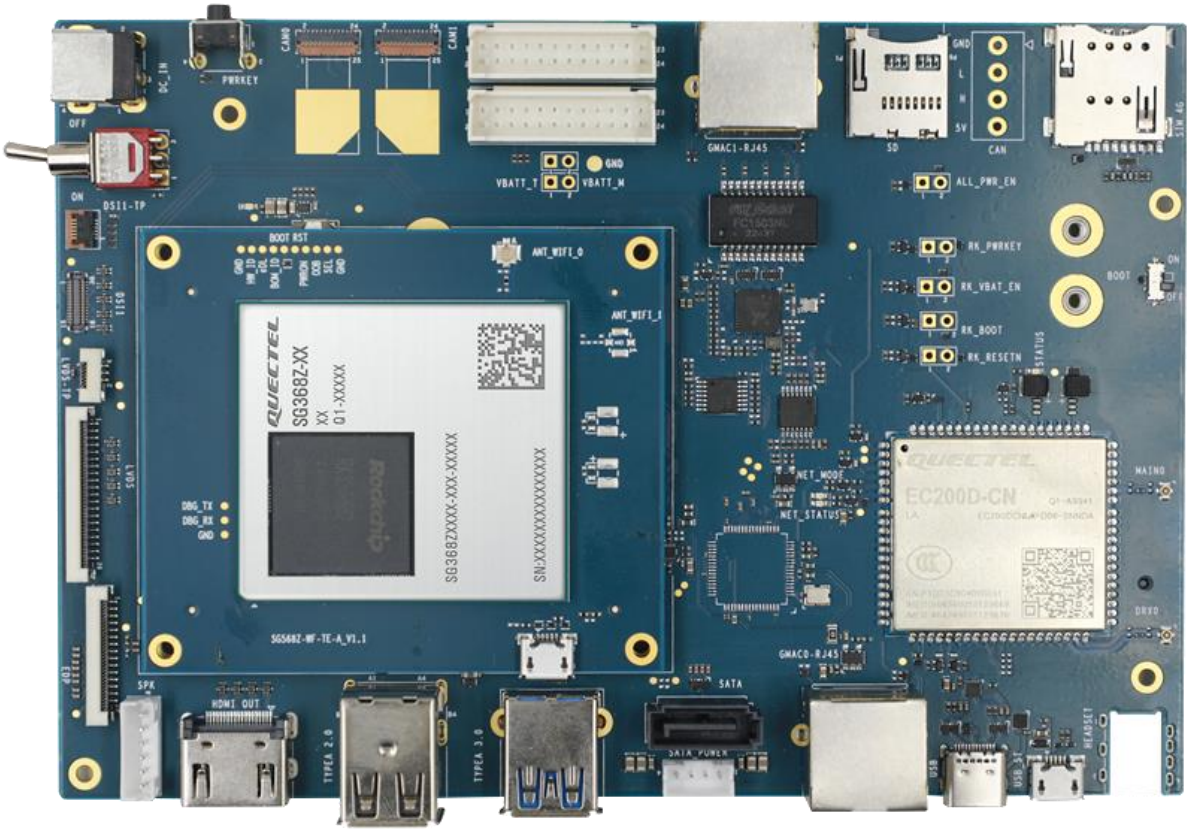

Figure 1: SG368Z EVB

2.2. Software Environment

Table 3: Software Environment

Type	Description
Code environment	SG368Z Android 11.0
Driver package	Quectel Bluetooth driver package
Compilation environment	Ubuntu 20.04
Tool	1. qlbt and Btdiag tool 2. QRCT test tool 3. Install Ellisys Bluetooth Analyzer software on your PC to use with Ellisys Bluetooth Vanguard

NOTE

Please contact Quectel Technical Support for SG368Z Android 11.0 source code if needed. If you use other development board models, please consult the corresponding manufacturer for details.

Execute the following command to obtain the source code for the Android Bluetooth driver:

Table 4: Command for Obtaining Source Code for the Android Bluetooth Driver

Module	Command
FC20 series	git clone https://git-master.quectel.com/wifi.bt/fc2x
FC21	git clone https://git-master.quectel.com/wifi.bt/fc2x
FC900E	git clone https://git-master.quectel.com/wifi.bt/fc2x
FC06E	git clone https://git-master.quectel.com/wifi.bt/fc6xe
FC64E	git clone https://git-master.quectel.com/wifi.bt/fc6xe
FC65E	git clone https://git-master.quectel.com/wifi.bt/fc6xe
FC66E series	git clone https://git-master.quectel.com/wifi.bt/fc6xe
FGE576Q	git clone https://git-master.quectel.com/wifi.bt/qcc207x

Quectel Bluetooth driver package includes the following files:

Table 5: Bluetooth Files

File	Description
BT/Linux/patch	Includes patch files for the BlueZ source code package to provide support for the Bluetooth modules (not covered in this document)
BT/FW	Bluetooth firmware (users need to obtain the firmware from the subdirectory with the same name as the module model under BT/FW directory)
BT/Android	Bringup software for Android platform
Doc/EN	Quectel_FC2x&FC0xE&FC6xE&FC900E&FGE576Q_第三方_Android 平台_蓝牙用户指导
Doc/CN	Quectel_FC2x&FC06E&FC6xE&FC900E&FGE576Q_Third-Party_Android_Bluetooth_User_Guide
WiFi	Wi-Fi Bringup driver package (not covered in this document)

3 Integration and Compilation

3.1. Code Integration

3.1.1. Adding libbt-vendor Code

Step 1: Enter the *vendor/rockchip/common/bluetooth/qcom/* directory of SG368Z Android source code.
Step 2: Copt *libbt-vendor* folder in the *BT/Android/* directory of Bluetooth driver package.

3.1.2. Adding Bluetooth Firmware

Step 1: Enter the *vendor/rockchip/common/bluetooth/lib/firmware/* directory of SG368Z Android source code.
Step 2: Copy the files in the Bluetooth firmware package in the BT/FW directory of the Bluetooth driver package to *vendor/rockchip/common/bluetooth/lib/firmware/.*

```
        new file:   vendor/rockchip/common/bluetooth/lib/firmware/hpbtfw21.tlv
        new file:   vendor/rockchip/common/bluetooth/lib/firmware/hpnv21.bin
        new file:   vendor/rockchip/common/bluetooth/lib/firmware/hpnv21g.bin
        new file:   vendor/rockchip/common/bluetooth/qcom/Android.mk
        new file:   vendor/rockchip/common/bluetooth/qcom/CleanSpec.mk
        new file:   vendor/rockchip/common/bluetooth/qcom/METADATA
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/Android.mk
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/NOTICE
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/include/bt_vendor_persist.h
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/include/bt_vendor_qcom.h
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/include/hci_smd.h
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/include/hci_uart.h
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/include/hw_ar3k.h
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/include/hw_rome.h
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/include/vnd_generic.txt
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/include/vnd_mako.txt
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/src/bt_vendor_persist.cpp
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/src/bt_vendor_qcom.c
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/src/hardware.c
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/src/hci_smd.c
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/src/hci_uart.c
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/src/hw_ar3k.c
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/src/hw_rome.c
        new file:   vendor/rockchip/common/bluetooth/qcom/libbt-vendor/vnd_buildcfg.mk
```

3.1.3. Configuring PCM

You can configure the PCM roles on the Android platform and the role of Bluetooth module according to your specific requirements.

NOTE

For assistance with PCM configuration, please contact Quectel Technical Support.

3.1.4. Configuring rfkill

Step 1: Configure the dts related to the rfkill command-line tool according to the Android platform you are using.

Step 2: Open the deconfig configuration file of the Android platform and set CONFIG_RFKILL=y to enable rfkill.

3.1.5. Modifying Android Configuring

Step 1: Open the configuration file *device/rockchip/common/wifi_bt_common.mk* in Android souce code.

Step 2: Remove BOARD_HAVE_BLUETOOTH_BCM := true. You can remove other Bluetooth configurations based on youyr specific needs.

Step 3: Add BOARD_HAVE_BLUETOOTH_QCOM := true.

```
--- a/device/rockchip/common/wifi_bt_common.mk
+++ b/device/rockchip/common/wifi_bt_common.mk
@@ -20,7 +20,7 @@ endif
 # bluetooth support
 ifeq ($(strip $(BOARD_CONNECTIVITY_VENDOR)), Broadcom)
 BOARD_HAVE_BLUETOOTH := true
-BOARD_HAVE_BLUETOOTH_BCM := true
+#BOARD_HAVE_BLUETOOTH_BCM := true
 BOARD_BLUETOOTH_BDROID_BUILDCFG_INCLUDE_DIR ?= device/rockchip/$(TARGET_BOARD_PLATFORM)/bluetooth

 ifeq ($(strip $(PRODUCT_BUILD_MODULE)), px5car)
@@ -39,3 +39,4 @@ endif

 BOARD_HAVE_BLUETOOTH_RTK := true

+BOARD_HAVE_BLUETOOTH_QCOM := true
```

3.2. Compilation and Replacement of libbt-vendor

3.2.1. Compilling libbt-vendor

Step 1: Open a new Ubuntu 20.04 terminal window.

Step 2: Enter the root directory of the Android source code and configure environment variables according to your specific needs.

Step 3: Execute **make libbt-vendor** to compile libbt-vendor code. After the compilation is completed, *libbt-vendor.so is generated in* the*out/target/product/xxx/vendor/lib64* directory and *out/target/product/xxx/vendor/lib* directory of Android source code.

3.2.2. Replaceing libbt-vendor

Execute the following commands to replace libbt-vendor.

```
adb root
adb push vendor/lib64/libbt-vendor.so /vendor/lib64/
adb push vendor/lib/libbt-vendor.so /vendor/lib/
```

NOTE

You can choose to either compile the libbt-vendor code or compile the entire Android source code, depending on the actual situation. For assistance with compiling the entire Android source code, please contact Quectel Technical Support.

3.3. Controlling Bluetooth Enabling Pin Through rfkill

Step 1: After completing the operations in ***Chapter 3.2***, check whether the /sys/class/rfkill/rfkill0 node is generated in the dts of rfkill.

Step 2: Execute **cat /sys/class/rfkill/rfkill0/type** to check the type of /sys/class/rfkill/rfkill0. If “bluetooth” is returned, it indicates that the type of the node is Bluetooth.

Step 3: Perform the following steps to check if you can control the Bluetooth enabling pin through rfkill.

1) Execute **echo 0 > /sys/class/rfkill/rfkill0/state** to pull the Bluetooth enabling pin low.
2) Execute **echo 1 > /sys/class/rfkill/rfkill0/state** to pull the Bluetooth enabling pin high.

If the Bluetooth enabling pin is successfully pulled low or high using the above commands, it indicates that the Bluetooth enabling pin can be controlled through rfkill.

```
rk3568_r:/ # cat /sys/class/rfkill/rfkill0/type
bluetooth
rk3568_r:/ # echo 0 > /sys/class/rfkill/rfkill0/state
rk3568_r:/ # echo 1 > /sys/class/rfkill/rfkill0/state
```

4 Bluetooth Verification

This chapter uses SG368Z Android 11 installed with Quectel Bluetooth module as an example to describe how to verify the Bluetooth function on Android 11.

NOTE

The function verification process for other versions of Android is similar to the Bluetooth function verification described in this chapter. You can contact Quectel Technical Support if needed.

4.1. Bluetooth Switch

Step 1: Open the Bluetooth settings interface on the SG368Z Android system.

Step 2: Click the Bluetooth switch to turn on Bluetooth. The Bluetooth switch displays as "On." If the Bluetooth switch remains "On" and does not automatically turn off, it indicates that the Bluetooth swicth is normal.

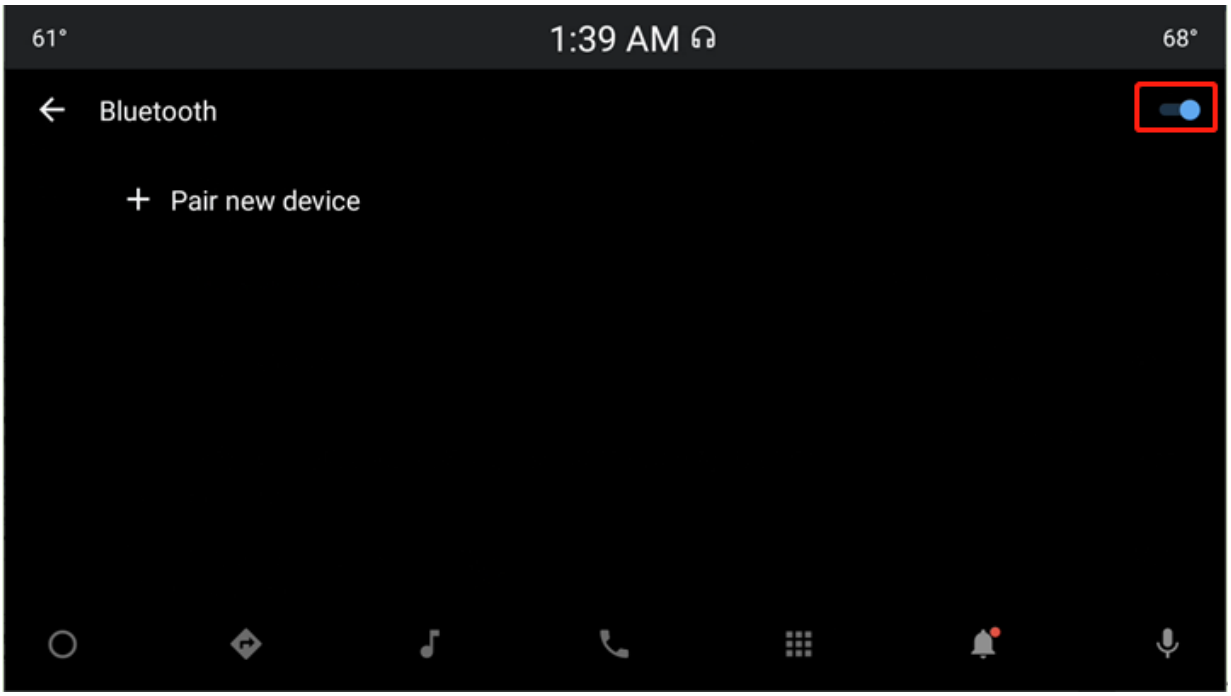

Figure 2: Bluetooth Switch

4.2. Scanning Function

Step 1: Turn on your mobile device's Bluetooth settings and set your phone's Bluetooth to discoverable mode.

Step 2: In the Bluetooth settings interface of the SG368Z Android platform, click "**Pair new device**". If the Bluetooth name of your phone is displayed in the "Available devices" section of the interface, it indicates that the Bluetooth scanning function is normal.

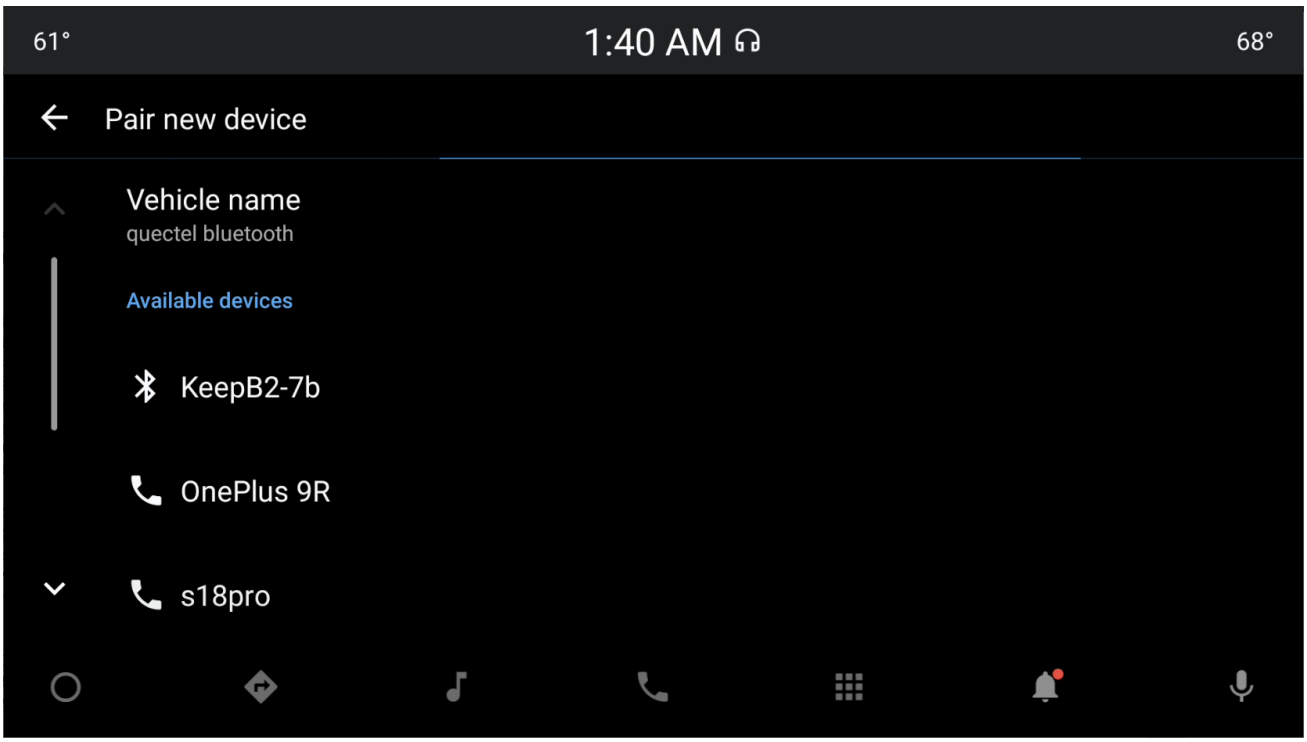

Figure 3: Scan Device

4.3. BLE Function

Step 1: Install and open the nRF Connect tool on the SG368Z Android platform.

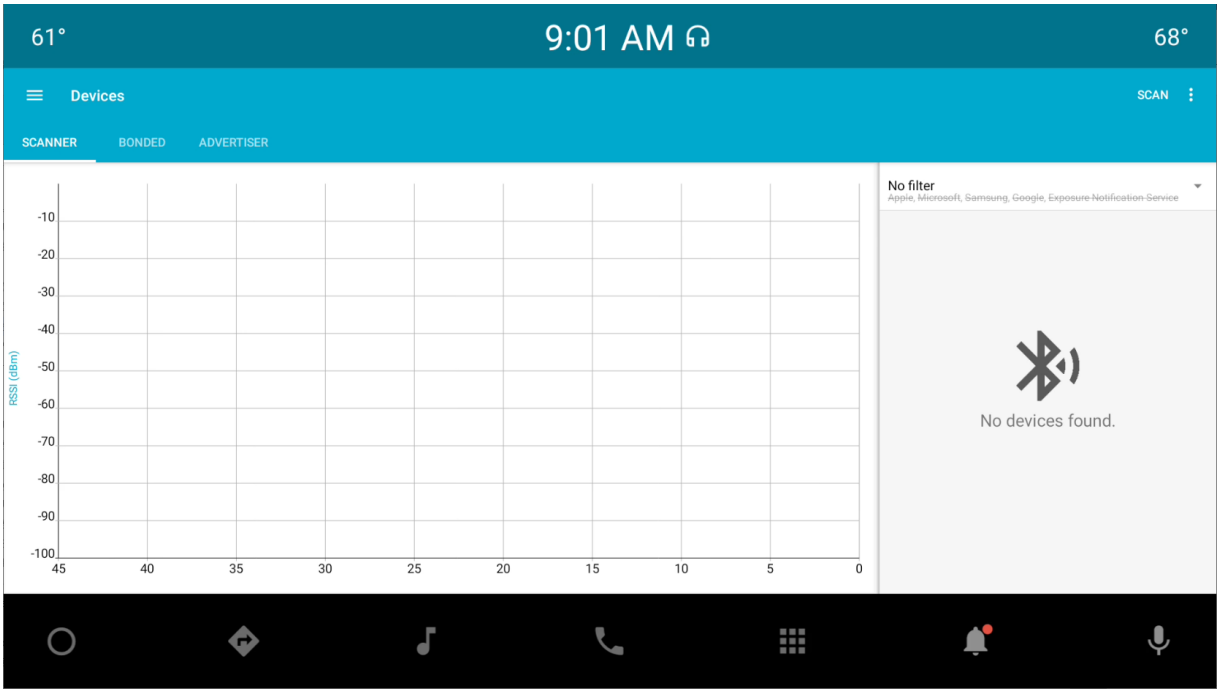

Figure 4: nRF Connect Tool Interface

Step 2: Install the nRF Connect tool on your mobile phone and configure it as a GATT server.

Step 3: Select "**SCANNER**" on the interface of the nRF Connect tool on SG368Z Android system. Then, click the "**SCAN**" button located at the top-right corner of the tool. A scan starts.The devices that are scanned will be displayed in the device below the "SCANNER" tab.

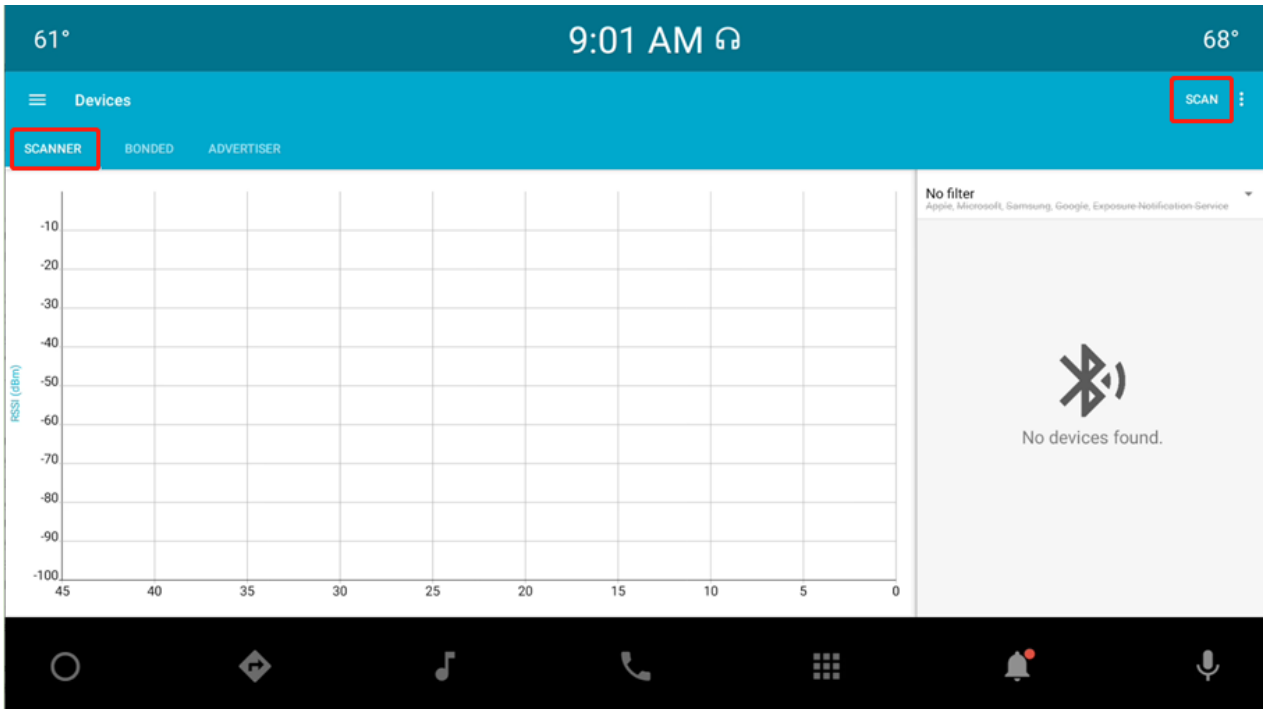

Figure 5: Scan Device

Step 4: Locate the your mobile phone in the device list and click "**CONNECT**" button next to the mobile phone to connect it.

Figure 6: Connect Device

Step 5: After the mobile phone is connected successfully, send data through the nRF Connect on your phone to send data as shown in ***Figure 7***.

Step 6: Check if the nRF Connect on the SG368Z Android platform has received the data. If the nRF Connect interface displays the data as shown in ***Figure 7***, it indicates that the BLE function is normal.

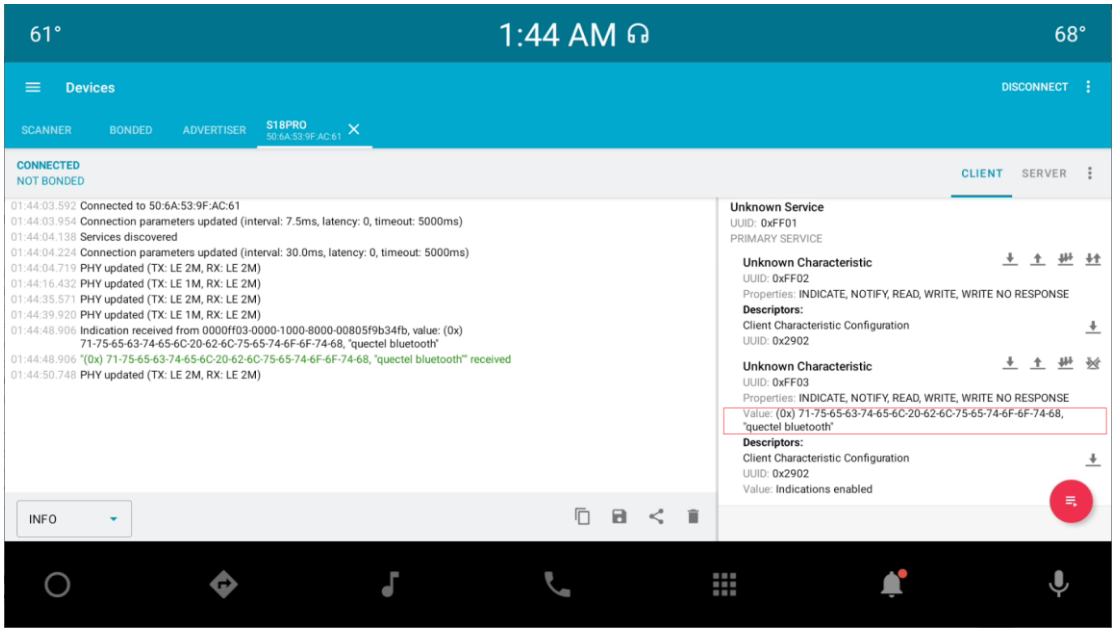

Figure 7: BLE Verification

NOTE

For the downloading and using of nRF Connect tool, please refer to the official website: https://www.nordicsemi.com/Products/Development-tools/nRF-Connect-for-mobile.

4.4. A2DP Sink

Step 1: Enter the Bluetooth settings interface on the SG368Z Android platform and click "**Pair new device**".

Step 2: After your mobile phone is scanned, click the mobile phone in the Available devices area to pair with and connect it.

Step 3: After the mobile phone is connected, check the "Profiles" section to see if Media Audio is enabled. If the Media Audio switch is displayed as "on," it indicates that Media Audio is enabled. If the switch is not turned on, you can simply click on the "Media Audio" switch to enable Media Audio.

Step 4: Play music on your mobile phone. If you hear the music playing from the speaker of the SG368Z Android platform, it indicates that the A2DP Sink is normal.

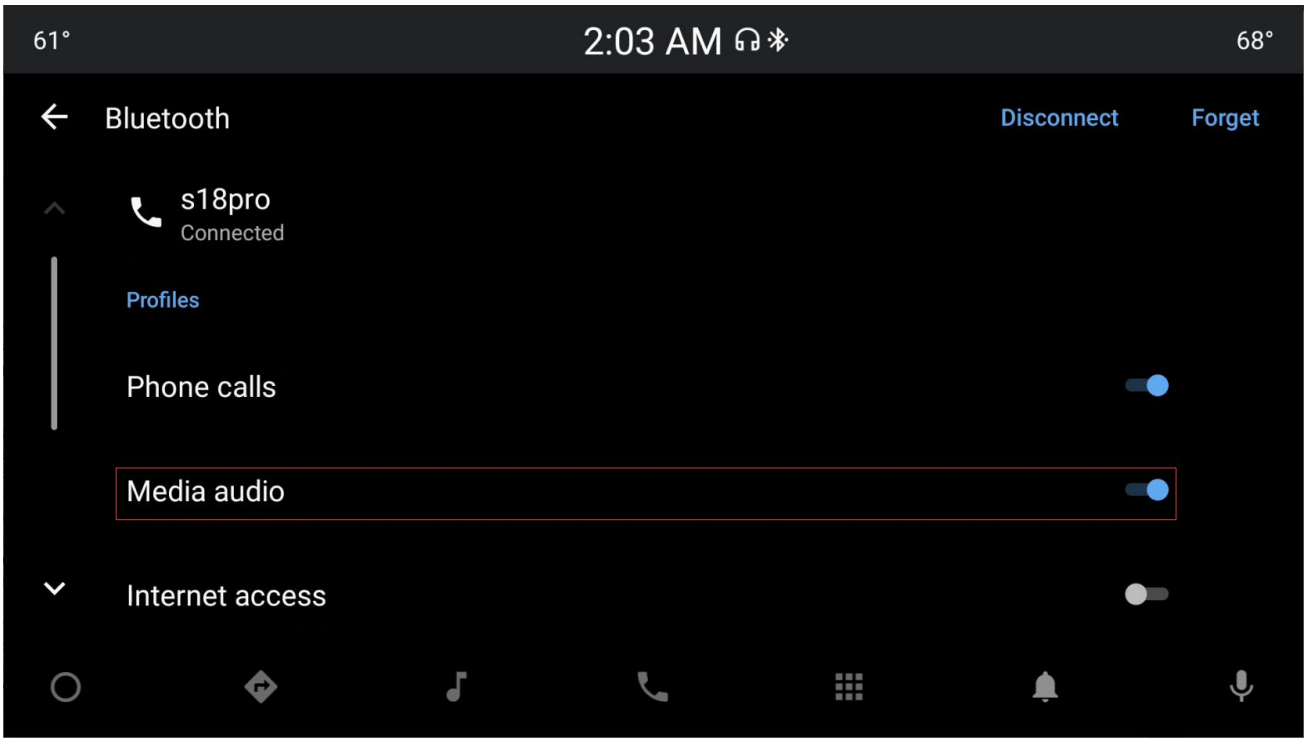

Figure 8: Enable Media Audio

4.5. HFP HF

Step 1: Enter the Bluetooth settings interface on the SG368Z Android platform and click "**Pair new device**".

Step 2: After your mobile phone is scanned, click the mobile phone in the Available devices area to pair with and connect it.

Step 3: After the mobile phone is connected, check the "Profiles" section to see if Phone Calls is enabled. If the Phone Calls switch is displayed as "on," it indicates that Phone Calls is enabled. If the switch is not turned on, you can simply click the Phone Calls switch to enable Phone Calls.

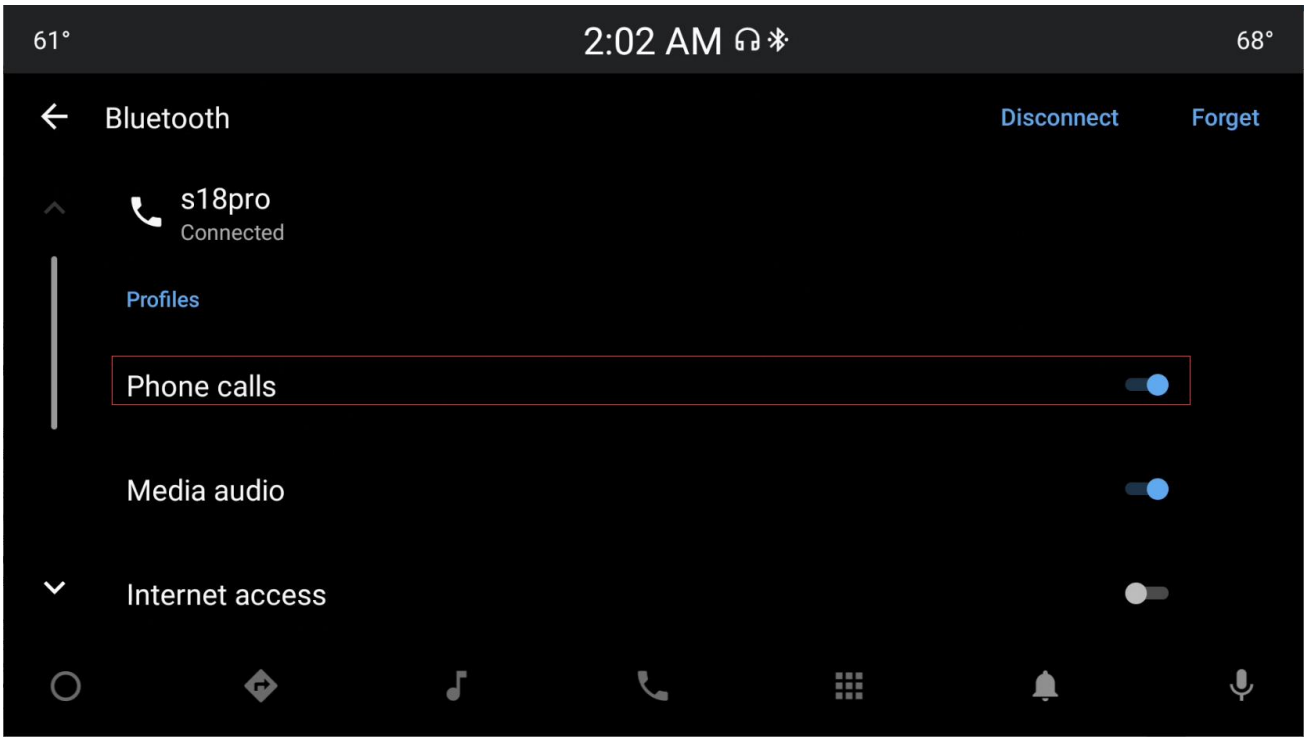

Figure 9: Enbale Phone calls

Step 5: Make a phone call from your mobile phone.

Step 6: Execute **tinycap test.wav -D 1 -d 0 -c 2 -r 8000 -b 16 -p 1024 -n 16** to record HFP audio.

Step 7: Execute **tinyplay test.wav -D 1** to play HFP audio. If the HFP audio is played successfully, it indicates that HFP HF is normal.

NOTE

You can adjust the parameters of **tinycap** and **tinyplay** based on the actual configuration of your development board.
The meaning of the parameters are as follows:

- *-D* Specify the sound card

- *-d* Specify the device
- *-c* The number of channels. *2* indicates two channels.
- *-r* Sample rate. Unit: kHz.
- *-b* The bit width of PCM. Unit:bit.
- *-p* The number of frames that are processed in a single interrupt
- *-n* The number of times to collect samples.

5 Usage of Tool

This chapter takes Quectel SG368Z EVB as an example to introduce the use of Bluetooth related tools in the library.

5.1. qlbt

This tool is used to check whether the main controller of the Android platform can communicate with the Bluetooth module through the UART when Bluetooth initialization fails.

5.1.1. Compilling qlbt

Step 1: Open a new Ubuntu 20.04 terminal window.
Step 2: Enter the root directory of the Android source code and configure environment variables according to your specific needs.
Step 3: Copy the *qlbt* folder in the *BT/Android* directory to the *BT/Android/vendor* directory.
Step 4: Execute **make qlbt** to compile the tool.
Step 5: After the compilation is completed, *out/target/product/xxx/vendor/bin/qlbt* is generated in the root directory of the Android source code. The qlbt-related files are stored in *out/target/product/xxx/vendor/bin/qlbt*.

5.1.2. Enabling Bluetooth

Step 1: Execute **svc bluetooth disable** to disable the Bluetooth function of the Android system.
Step 2: Execute **echo 0 > /sys/class/rfkill/rfkill0/state** to pull the Bluetooth enabling pin low.
Step 3: Execute **echo 1 > /sys/class/rfkill/rfkill0/state** to pull the Bluetooth enabling pin high.

5.1.3. Checking Bluetooth Communication via UART

Step 1: Execute **adb root** to grant root permissions.
Step 2: Execute **adb shell** to access the Android command line.
Step 3: Execute **adb push qlbt /vendor/** to push *qlbt* folder to the *vendor/ directory* of Android system.
Step 4: Execute **#chmod 777 /vendor/qlbt** to change the permissions of qlbt to readable, writable and executable.
Step 5: Taking the currently used UART port ttyS1 as an example, execute # **./vendor/qlbt -v**

/dev/ttyS1 to read the version information of the Bluetooth chip. If the information as shown in the red box in the figure below, it indicates that the Android platform can communicate with the Bluetooth module normally through the UART.

```
rk3568_car:/ # ./vendor/qlbt -v /dev/ttyS1
send   :  0x01 0x00 0xfc 0x01 0x19
recv   :  0x04 0x0e 0x12 0x01 0x00 0xfc 0x00 0x19 0x0c 0x13 0x00 0x00 0x00 0xe6 0x38 0x01 0x02 0x11 0x12 0x0c 0x40
OK     :
```

NOTE

/dev/ttyS1 indicates the device ID corresponding to UART port of the Bluetooth module. It is only an example. Please choose the device ID corresponding to the UART port used on your development board.

5.2. Btdiag

This tool is used to verify the RF function of Bluetooth with QRCT tool.

5.2.1. Compilling Btdiag

Step 1: Open a new Ubuntu 20.04 terminal window.
Step 2: Enter the root directory of the Android source code and configure environment variables according to your specific needs.
Step 3: Copy the *Btdiag* folder in the *BT/Android* directory to the *BT/Android/vendor* directory.
Step 4: Execute **make Btdiag** to compile the tool.
Step 5: After the compilation is completed, *out/target/product/xxx/vendor/bin/Btdiag* is generated in the root directory of the Android source code. The Btdiag-related files are stored in *out/target/product/xxx/vendor/bin/Btdiag*.

5.2.2. Enabling Bluetooth

Step 1: Execute **svc bluetooth disable** to disable the Bluetooth function of the Android system.
Step 2: Execute **echo 0 > /sys/class/rfkill/rfkill0/state** to pull the Bluetooth enabling pin low.
Step 3: Execute **echo 1 > /sys/class/rfkill/rfkill0/state** to pull the Bluetooth enabling pin high.

5.2.3. Using Btdiag with QRCT

Step 1: Execute **adb root** to grant root permissions.
Step 2: Execute **adb shell** to access the Android command line.
Step 3: Execute **adb push Btdiag /vendor/** to push *Btdiag* folder to the *vendor/ directory* of Android system.

Step 4: Copy the currently used *libbt-vendor.so* to the *vendor/ directory* of Android system.

Step 5: Execute **#chmod 777 /vendor/Btdiag** to change the permissions of Btdiag to readable, writable and executable.

Step 6: Execute the following command to run Btdiag tool.

```
./vendor/Btdiag UDT=yes PORT=2390 IOType=SERIAL QDARTIOType=ethernet BT-DEVICE=/dev/ttyUSB0 BT-BAUDRATE=115200
```

```
rk3568_car:/ # ./vendor/Btdiag  UDT=yes PORT=2390 IOType=SERIAL QDARTIOType=ethernet BT-DEVICE=/dev/ttyUSB0 BT-BAUDRATE=115200
Run in UDT mode
bc_download_fw: dlopen(libbt-vendor.so, ...) = 0x2c7385de7e71fb19
bc_download_fw: dlsym() = 0x74675a2008
bc_download_fw: lib_interface->init() = 0
bc_download_fw: BT_VND_OP_POWER_CTRL: 0
bc_download_fw: BT_VND_OP_USERIAL_OPEN: 1
bc_download_fw: ret = 5
Download Patch firmware and NVM file completed
Connected to SoC DUT!

Thread created successfully
Socket created
bind done
Waiting for incoming connections...
```

Step 7: Execute **ifconfig** to check the Ethernet IP address.

Step 8: Open the QRCT tool and enter the TCP port set in ***Step 6*** into the input box next to “TCP Port”. Enter the IP address obtained in ***Step 7*** into the input box next to “IP Address”. Then, click the "**Connect**" button to connect the Btdiag tool. As shown in the figure below:

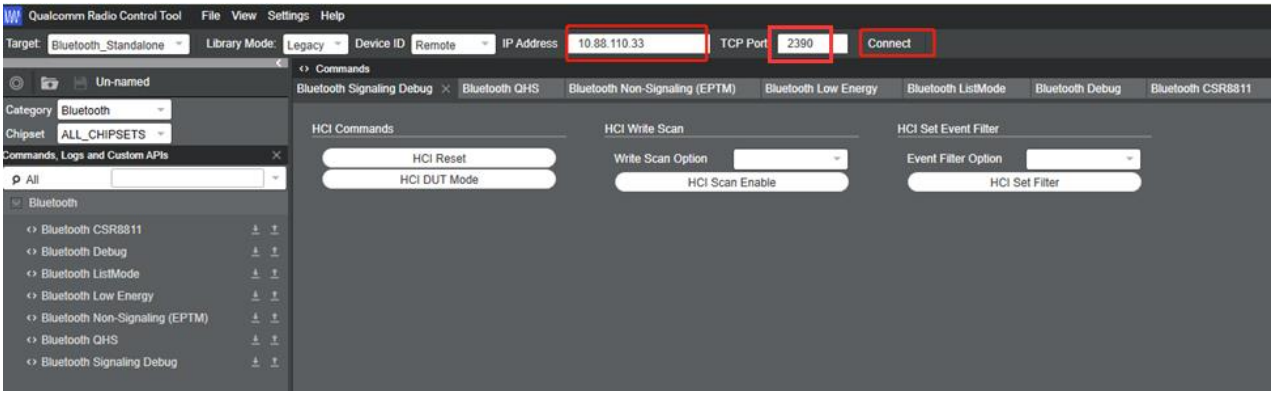

Figure 10: QRCT Configuration

After completing the above steps, you can proceed with the subsequent RF testing process. For specific RF testing procedures, please contact Quectel Technical Support.

NOTE

1. Using QRCT test tool requires authorization from Qualcomm.
2. The PC on which QRCT is used must be on the same network segment as the development board on which Btdiag runs.

6 Bluetooth Log Capture

6.1. Device Log

Step 1: Power on the Android device.

Step 2: Connect the Android device to the USB port of the PC. Open ADB tool on the PC and then execute **adb shell logcat > logcat.txt** to export device log as *logcat.txt*.

6.2. Kernel Log

Step 1: Power on the Android device.

Step 2: Connect the Android device to the USB port of the PC. Open ADB tool on the PC and then execute **adb shell dmesg >dmesg.txt** to export kernel log as *dmesg.txt*.

6.3. HCI Snoop Log

Step 1: Power on the Android device.

Step 2: In the "Settings" interface of the SG368Z Android 11 system, enter “Developer options” and click "**Enable Bluetooth HCI snoop log**" option. Then select “**Enabled**” in the pop-up window as shown in the following figure (the settings may be different for different Android devices).

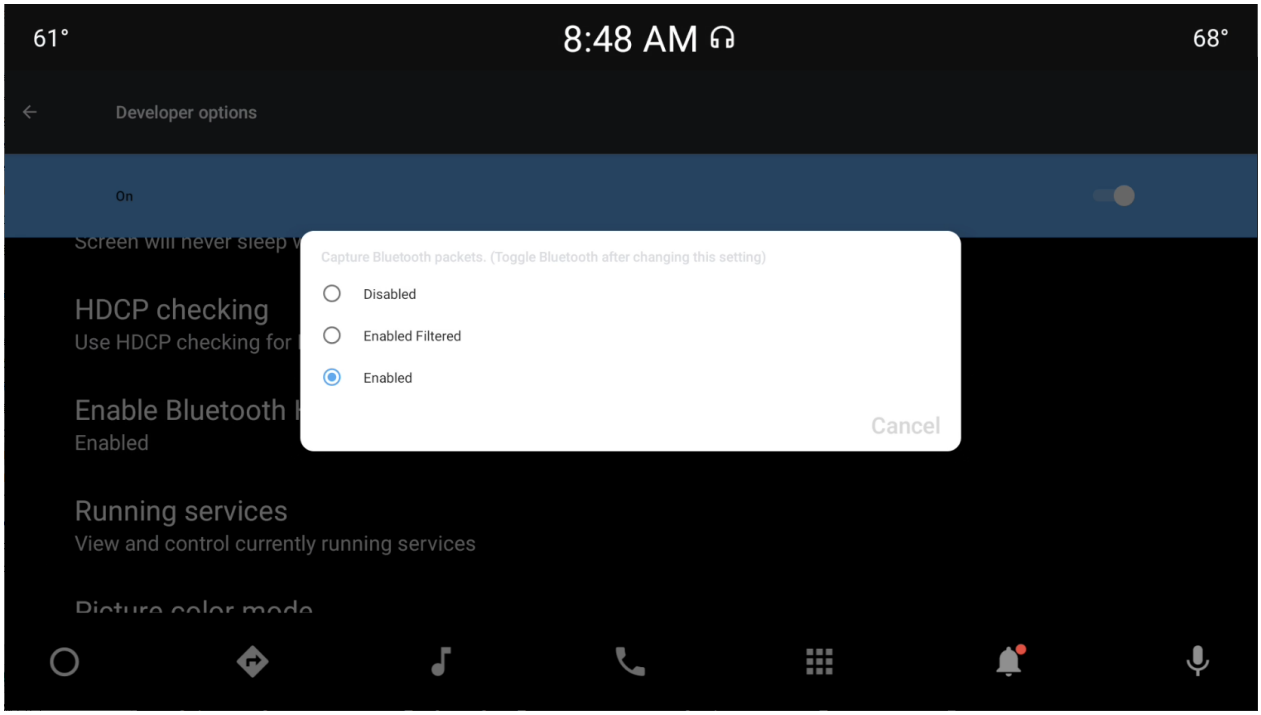

Figure 11: Enable "Capture Bluetooth Packets"

Step 3: Connect the Android device to the USB port of the PC. Open ADB tool on thes PC and then execute **adb pull /data/misc/bluedroid/btsnoop_hci.cfa** to export the HCI log (the file name is *hci.cfa* by default) from */data/misc/bluedroid/btsnoop_hci.cfa* directory of Android device.

6.4. Btsnoop Air Log

Step 1: Power on the Android device and connect it to the USB port of the PC.
Step 2: Connect Ellisys Bluetooth Vanguard to PC.
Step 3: Open Ellisys Bluetooth Analyzer on PC.
Step 4: Click "**Record**" on the interface of Ellisys Bluetooth Analyzer to capture air log.
Step 5: After the Bluetooth function test is completed, click "**Stop**", and the air log file (the file name is *Untitled.btt* by default) and *bt_config.conf* file are automatically generated.
Step 6: Click "**Save&Continue**" to save the air log to a custom path.
Step 7: Open ADB tool on the PC and then execute **adb pull /data/misc/bluedroid/bt_config.conf** to export *bt_config.conf* file which is in the */data/misc/bluedroid/ directoru of Android device*.

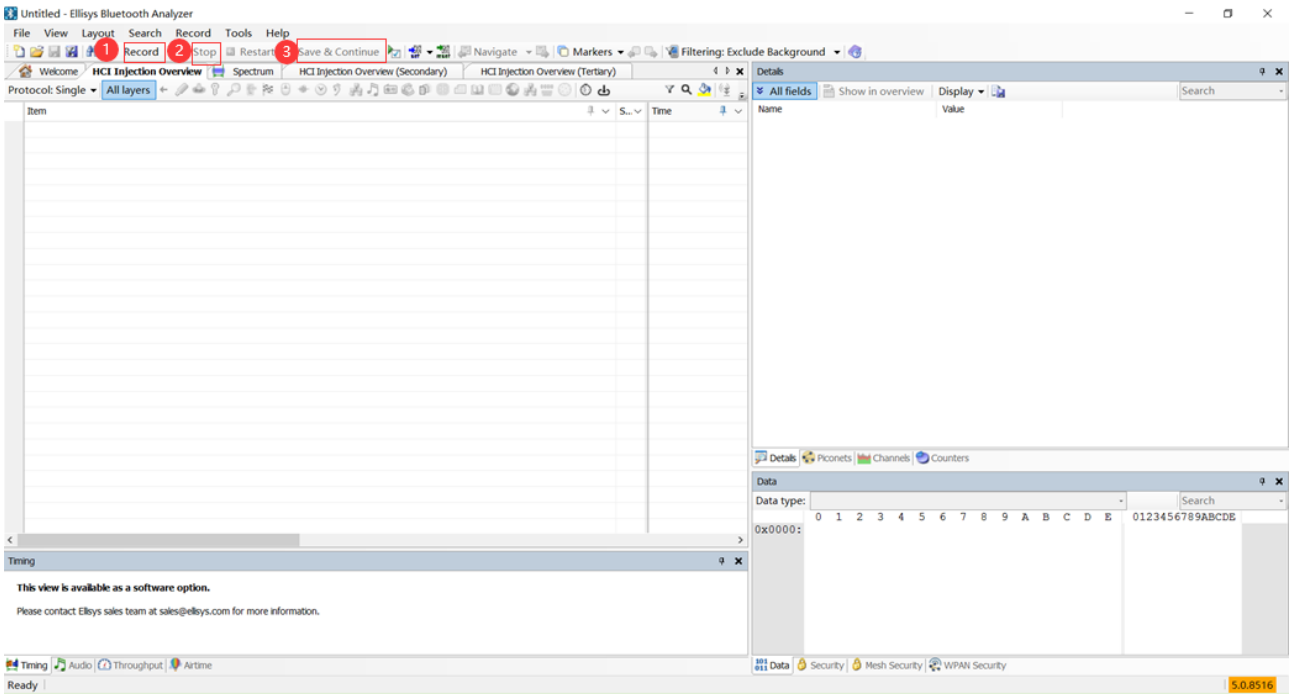

Figure 12: Ellisys Bluetooth Analyzer Interface

7 Appendix Terms and Abbreviations

Table 6: Terms and Abbreviations

Abbreviation	Description
A2DP	Advanced Audio Distribution Profile
ARM	Advanced RISC Machine
AVRCP	Audio/Video Remote Control Profile
App	Application
AVRCP	Audio/Video Remote Control Profile
BLE	Bluetooth Low Energy
EVB	Evaluation Board
GATT	Generic Attribute Profile
HAL	Hardware Abstraction Layer
HCI	Host Controller Interface
HF	Hands-Free unit
HFP	Hands-free Profile
ID	Mostly refers to Identifier in terms of software
LCC	Leadless Chip Carrier (package)
MAC	Medium Access Control
PC	Personal Computer
PCM	Pulse Code Modulation
TCP	Transmission Control Protocol
UART	Universal Asynchronous Receiver/Transmitter

USB	Universal Serial Bus