XR829 Bluetooth RFTest CLI Tool User Guide

Adroid8.1 Linux4.9

Revision 1.0

November 13, 2018

Declaration

THE INFORMATION FURNISHED BY XRADIO IS BELIEVED TO BE ACCURATE AND RELIABLE. XRADIO RESERVES THE RIGHT TO MAKE CHANGES IN CIRCUIT DESIGN AND/OR SPECIFICATIONS AT ANY TIME WITHOUT NOTICE. XRADIO DOES NOT ASSUME ANY RESPONSIBILITY AND LIABILITY FOR ITS USE. NOR FOR ANY INFRINGEMENTS OF PATENTS OR OTHER RIGHTS OF THE THIRD PARTIES WHICH MAY RESULT FROM ITS USE. NO LICENSE IS GRANTED BY IMPLICATION OR OTHERWISE UNDER ANY PATENT OR PATENT RIGHTS OF XRADIO. THIS DATASHEET NEITHER STATES NOR IMPLIES WARRANTY OF ANY KIND, INCLUDING FITNESS FOR ANY PARTICULAR APPLICATION.

THIRD PARTY LICENCES MAY BE REQUIRED TO IMPLEMENT THE SOLUTION/PRODUCT. CUSTOMERS SHALL BE SOLELY RESPONSIBLE TO OBTAIN ALL APPROPRIATELY REQUIRED THIRD PARTY LICENCES. XRADIO SHALL NOT BE LIABLE FOR ANY LICENCE FEE OR ROYALTY DUE IN RESPECT OF ANY REQUIRED THIRD PARTY LICENCE. XRADIO SHALL HAVE NO WARRANTY, INDEMNITY OR OTHER OBLIGATIONS WITH RESPECT TO MATTERS COVERED UNDER ANY REQUIRED THIRD PARTY LICENCE.

Revision History

Version	Data	Summary of Changes
1.0.0	2018-11-13	Initial Version
1.0.1	2018-11-14	Add BLE RSSI test item
1.0.2	2019-04-18	Supplementary test environment configuration success description, confirm the existence of key nodes, etc
1.0.3	2019-04-28	Supplemental tool checkpoints, etc.
1.0.4	2019-07-10	Modify the parameters of TX (add hopping_mode)
1.0.5	2019-11-28	Add Sectio 3.2 Parameter Description, Modify example of tests
1.0.6	2020-01-19	Added single_tone test item
1.0.7	2020-03-11	Added description of hciattach tools for different platforms

Table 0-1 Revision History

Contents

Tables and Figures

1 Overview

1.1 Purpose of writing

Introduce Bluetooth RF Test command line tool (btetf) for XR829.

1.2 Scope of application

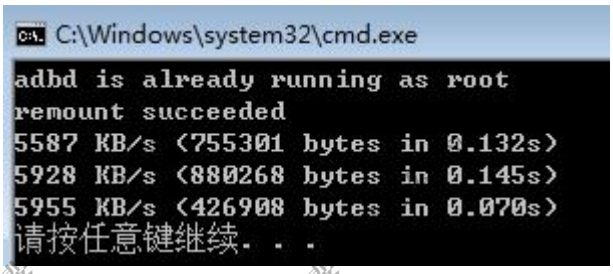
Software Environment：Linux & Android.

2 Bluetooth RF Test Environment Setup

2.1 Prepare the test platform

The test tools required are hciattach, hciconfig, btetf

Run script “ **init_tools_android.bat** ” or “ **init_tools_linux.bat**” to install test tools.



Note:

（1）When we use Anritsu MT8850A tester for testing, the TEST PAUSE fuction needs to be set the off state for testing.

（2）When testing with the btetf tool, the hciattach tool is used to load the Bluetooth firmware. The Linux platform comes with hciattach, so there is no need to push this file. The hciattach tool in the toolkit is provided for Android.

2.2 Configuring the test environment

There are two ways to configure the test environment:

A) Run script “init_test.bat” .

B) Configure manually.

2.2.1 Run the script to configure

The command set for configuring the test environment is integrated in the script "**init_test.bat**". Double-click to run the script, and if "**init_success!**" appears it means the test environment has been configured successfully:

```
[userial_sync] read buf: 00 00.
[userial_sync] uart sync count: 2.
[userial_sync] read buf: 00 00.
[userial_sync] uart sync count: 3.
[userial_sync] read buf: 4f 4b.
[userial_sync] Receive OK, uart sync done.
[load_btfirmware] start loading firmware...
[load_btfirmware] open firmware file success. loading...
load firmware done.
jump:
set pc 0, val 0
Now the system will jump to 00000000
Set HW FlowControl On
userial_vendor_set_hw_fctrl set hw flowcontrol on
[xradio_init] send reset cmd...
writing
01 03 0c 00
received 7
04 0e 04 05 03 0c 00
[xradio_init] update hci baudrate...
writing
01 18 fc 04 60 e3 16 00
received 7
04 0e 04 05 18 fc 00
Done setting baudrate
[xradio_init] set bdaddr...
generating random bdaddr...
writing
01 0a fc 09 02 00 06 0b b9 f9 fe 22 22
received 7
04 0e 04 05 0a fc 00
writing
01 03 0c 00
received 7
04 0e 04 05 03 0c 00
[xradio_init] bring up hci...
Done setting line discpline
Device setup complete
brom_done
hci0:   Type: Primary  Bus: UART
        BD Address: 22:22:FE:F9:B9:0B  ACL MTU: 1021:8  SCO MTU: 255:4
        UP RUNNING
        RX bytes:1168 acl:0 sco:0 events:56 errors:0
        TX bytes:752 acl:0 sco:0 commands:56 errors:0
        Features: 0xbf 0xfe 0xcd 0xfe 0xdb 0xfd 0x7b 0x87
        Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
        Link policy: RSWITCH SNIFF
        Link mode: SLAVE ACCEPT
        Name: 'XR829_BT'
        Class: 0x000000
        Service Classes: Unspecified
        Device Class: Miscellaneous,
        HCI Version: 4.1 (0x7)  Revision: 0xa68
        LMP Version: 4.1 (0x7)  Subversion: 0xa68
        Manufacturer: not assigned (1597)

init success!
```

Figure 2-1 Run script "**init_test.bat**" to configure the test environment

2.2.2 Configure manually

If you choose to configure the test environment manually, follow the steps below:

1) Load XR829 BT firmware (Please use the bluetooth's tty number instead of the yellow part)

```
venus-a3:/# hciattach -n ttyS1 xradio &
```

```
venus-a3:/ # hciattach -n ttyS1 xradio &
hciattach -n ttyS1 xradio &
[1] 4182
venus-a3:/ # xradio_init
set LPM mode:disabled[userial_sync] uart sync count: 1.
[userial_sync] read buf: 00 00.
[userial_sync] uart sync count: 2.
[userial_sync] read buf: 4f 4b.
[userial_sync] Receive OK, uart sync done.
Set uart mode done
[userial_sync] uart sync count: 1.
[userial_sync] read buf: 00 00.
[userial_sync] uart sync count: 2.
[userial_sync] read buf: 4f 4b.
[userial_sync] Receive OK, uart sync done.
[load_btfirmware] start loading firmware...
[load_btfirmware] open firmware file success. loading...
load firmware done.
jump:
set pc 0, val 0
Now the system will jump to 00000000
Set HW FlowControl On
userial_vendor_set_hw_fctrl set hw flowcontrol on
[xradio_init] send reset cmd...
writing
01 03 0c 00
received 7
04 0e 04 05 03 0c 00
[xradio_init] update hci baudrate...
writing
01 18 fc 04 60 e3 16 00
received 7
04 0e 04 05 18 fc 00
Done setting baudrate
[xradio_init] set bdaddr...
writing
01 0a fc 09 02 00 06 6b 7b eb 13 22 22
received 7
04 0e 04 05 0a fc 00
writing
01 03 0c 00
received 7
04 0e 04 05 03 0c 00
[xradio_init] bring up hci...
Done setting line discpline
Device setup complete
```

Figure 2-2 Manually configure - Load bt fw

2) Start up device:

```
venus-a3:/# hciconfig hci0 up
```

```
venus-a3:/ # hciconfig hci0 up
hciconfig hci0 up
venus-a3:/ #
```

Figure 2-3 Manually configure - Start up device

3) Check device status:

```
venus-a3:/# hciconfig -a
```

Check the device status, and if the result is same as the figure shown below, it means the test configuration has been initialized successfully:

```
venus-a3:/ # hciconfig -a
hciconfig -a
hci0:   Type: Primary  Bus: UART
        BD Address: 22:22:13:EB:7B:6B  ACL MTU: 1021:8  SCO MTU: 255:4
        UP RUNNING
        RX bytes:1168 acl:0 sco:0 events:56 errors:0
        TX bytes:752 acl:0 sco:0 commands:56 errors:0
        Features: 0xbf 0xfe 0xcd 0xfe 0xdb 0xfd 0x7b 0x87
        Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
        Link policy: RSWITCH SNIFF
        Link mode: SLAVE ACCEPT
        Name: 'XR829_BT'
        Class: 0x000000
        Service Classes: Unspecified
        Device Class: Miscellaneous,
        HCI Version: 4.1 (0x7)  Revision: 0xa64
        LMP Version: 4.1 (0x7)  Subversion: 0xa64
        Manufacturer: not assigned (1597)
```

Figure 2-4 Manually configure - Check device status

2.3 Checkpoints for the noraml use of the tool

2.3.1 Confirm that the relevant node exists

When configuring the Bluetooth test environment, we need the access of certain key nodes of the operating system.

Please confirm that the system has the following key nodes:

```
venus-a3:/proc/bluetooth/sleep # ls -l
ls -l
total 0
--w--w---- 1 bluetooth net_bt_admin 0 2019-04-18 09:52 btwake
--w--w---- 1 bluetooth net_bt_admin 0 2019-04-18 09:52 btwrite
--w--w---- 1 bluetooth net_bt_admin 0 2019-04-18 09:52 lpm
venus-a3:/proc/bluetooth/sleep #
```

Figure 2-5 Ensure that the essential file nodes exist

If those key nodes do not exist, there may be something wrong with Bluetooth Porting. Please refer to ***XR829 Bluetooth Porting Guide(Android8.1) V1.0.pdf*** chapter 2.1 “Add Sleep Wakeup and FDI Module”.

2.3.2 Ensure the firmware path of bluetooth

The hciattach command line tool on Android will look for firmware in the following path :

/system/etc/firmware/fw_xr829_bt.bin or /system/vendor/etc/firmware/fw_xr829_bt.bin

The hciattach command line tool on Linux will look for firmware in the following path :

/lib/firmware/fw_xr829_bt.bin

Please confirm that the path to the firmware is one of the above paths.

2.3.3 ensure

Bluetooth RF Test command line tool (btetf) rely on the Bluetooth drive module that comes with kernel. Please confirm that the following kernel compilation options are selected on.

```
< > HCI USB driver
< > HCI SDIO driver
<*> HCI UART driver
[*]   UART (H4) protocol support
[ ]   BCSP protocol support (NEW)
[ ]   Atheros AR300x serial support (NEW)
[ ]   HCILL protocol support (NEW)
[ ]   Three-wire UART (H5) protocol support (NEW)
< > HCI BCM203x USB driver
< > HCI BPA10x USB driver
< > HCI BlueFRITZ! USB driver
< > HCI VHCI (Virtual HCI device) driver
< > Marvell Bluetooth driver support
```

Figure 2-6 Ensure relevant kernel compilation options

3 Software introductions

3.1 Help information

Btetf tool is mainly used to perform Bt Tx/Rx Test and Ble Tx/Rx Test.

You can use “btetf -h” command to see the usage of it:

```
venus-a3:/# btetf -h
```

```
venus-a3:/system/bin # btetf -h
btetf -h
BT-ETF Tool Version 1.0.0
Usage:
        btetf [options] <command> [command parameters]
Options:
        --help  Display help
        --debug Dump Debug Info
        -i dev  HCI device
Commands:
        hci_reset       Hci Reset
        get_fwv         Hci Read Local Version Info
        ble_tx          Ble Tx Mode Start
        ble_rx          Ble Rx Mode Start
        ble_close       Ble Test Mode End
        bt_test_mode    Bt Test Mode Open
        bt_tx           Bt Tx Mode Start
        bt_close_tx     Bt Tx Mode End
        bt_rx           Bt Rx Mode Start
        bt_close_rx     Bt Rx Mode End
        auto_test       Auto Test Mode

For more information on the usage of each command use:
        btetf <command> --help
```

Figure 3-1 Use “btetf -h” to get tool usage help

3.2 Parameter Description

In some test items, you may need to set the Link Type and Packet Type. Data Packet Type are related to the logical transmission links that use them. Four different Link Type are defined:ACL/SCO(Basic Rate)、eSCO(Basic Rate)、ACL(EDR) and eSCO(EDR).

4 Operation instructions for Test

4.1 BT Test

4.1.1 Start BT RX Test

You can use “ btetf bt_rx -h ” to see the usage of “ bt_rx ” command:

```
venus-a3:/ # btetf bt_rx -h
btetf bt_rx -h
bt_rx: unrecognized option: h
Usage:
        bt_rx [option] [parameters]
Options:
       [--bdaddr=N]
       [--channel_num=N]  Range:0~79 default=1
       [--link_type=N] Range:0~3 default=0
                     0 ACL/SCO (Basic Rate)
                     1 eSCO (Basic Rate)
                     2 ACL (EDR)
                     3 eSCO (EDR)
       [--packet_type=N] Range:0~15 default=3
              ACL/SCO (Basic Rate):
                     0 NULL
                     1 POLL
                     2 FHS
                     3 DM1
                     4 DH1
                     5 HV1
                     6 HV2
                     7 HV3
                     8 DV
                     9 AUX1
                     10 DM3
                     11 DH3
                     14 DM5
                     15 DH5
              eSCO (Basic Rate):
                     0 NULL
                     1 POLL
                     7 EV3
                     12 EV4
                     13 EV5
              ACL (EDR):
                     0 NULL
                     1 POLL
                     2 FHS
                     3 DM1
                     4 2-DH1
                     5 HV1
                     6 HV2
                     7 HV3
                     8 3-DH1
                     9 AUX1
                     10 2-DH3
                     11 3-DH3
                     14 2-DH5
                     15 3-DH5
              eSCO (EDR):
                     0 NULL
                     1 POLL
                     6 2-EV3
                     7 3-EV3
                     12 2-EV5
                     13 3-EV5
Example:
tetf bt_rx --bdaddr 11:22:33:44:22:22 --channel_num 8 --link_type 1 --packet_type 3
```

Figure 4-1 Use “btetf bt_rx -h” to get BT RX test help

Parameters needed to be specified for BT RX test are as follows:

Default parameter list	Sample Value	Remarks
bdaddr	11:22:33:44:22:22	Mac Address
channel_num	1	Channel
link_type	0	Link Type
packet_type	4	Packet Type

Example: Specify BT RX test to receive DH1 packet on channel 1: (Notice:At this point, there should be an auxiliary device to send packet to the test equipment.)

```
venus-a3:/# btetf -i hci0 -d    bt_rx --bdaddr 11:22:33:44:22:22 --channel_num 1 --link_type 0 --packet_type 4
```

```
venus-a3:/ # btetf -i hci0 -d  bt_rx --bdaddr 11:22:33:44:22:22 --channel_num 8 --link_type 0 --packet
_type 3
2:33:44:22:22 --channel_num 8 --link_type 0 --packet_type 3                    <
bt_rx:channel_num(8) link_type/packet_type(0-3) lpt:(0x 3)
< HCI Command: opcode:(0xfc50) ogf-ocf:(0x3f-0x0050) plen:(13)
   10 11 22 33 44 22 22 00 00 00 08 00 03
> HCI Event: 0x0e plen 17
   05 50 FC 00 10 11 22 33 44 22 22 00 00 00 08 00 03
venus-a3:/ #
```

Figure 4-2 Start BT RX test example

Example: Specify BT RX test to receive 3-DH5 packet on channel 1: (Notice:At this point, there should be an auxiliary device to send packet to the test equipment.)

```
venus-a3:/# btetf -i hci0 -d    bt_rx --bdaddr 11:22:33:44:22:22 --channel_num 1 --link_type 2 --packet_type 15
```

4.1.2 Stop BT RX Test

Use the following command to stop BT RX test:

```
venus-a3:/# btetf -i hci0 -d    bt_close_rx
```

When BT RX test is stopped, BT RX test results will be displayed (The red box of the following picture shows that device has received 90 packages):

```
venus-a3:/ # btetf -i hci0 -d  bt_close_rx
btetf -i hci0 -d  bt_close_rx
< HCI Command: opcode:(0xfc50) ogf-ocf:(0x3f-0x0050) plen:(1)
   F0
> HCI Event: 0x0e plen 21
   05 50 FC 00 F0 5A 00 00 00 00 00 00 00 00 00 00 00 00 00 00
   00
event: 04 0E 15 05 50 FC 00 F0 5A 00 00 00 00 00 00 00 00 00 00 00
event: 00 00 00 00
status:0x 0, mode_status:0xf0
receive_packet:90, hec_err_packet:0
crc err packet:0, type err packet:0
venus-a3:/ #
```

Figure 4-3 Stop BT RX test

4.1.3 Start BT TX Test

You can use “ btetf bt_tx -h ” to see the usage of “ bt_tx ” command:

```
root@TinaLinux:/# btetf  bt_tx  -h
btetf  bt_tx  -h
bt_tx: unrecognized option: h
Usage:
        bt_tx [option] [parameters]
Options:
       [--bdaddr=N]
       [--pattern=N] Range:0~7 Data pattern
                     0 Transmitter test - 0 pattern
                     1 Transmitter test - 1 pattern
                     2 Transmitter test - 1010 pattern
                     3 Transmitter test - 0101 pattern
                     4 Transmitter test 1111 0000 pattern
                     5 Transmitter test-0000 1111 pattern
                     6 Pseudorandom 9 bit sequence
                     7 Pseudorandom 15 bit sequence
                     8-254 reserved
       [--packet_len=N]   Range:0~65535 default=10
       [--channel_num=N]  Range:0~79 default=1
       [--power_level=N]  Range:0~6 default=1
       [--link_type=N] Range:0~3 default=0
                     0 ACL/SCO (Basic Rate)
                     1 eSCO (Basic Rate)
                     2 ACL (EDR)
                     3 eSCO (EDR)
```

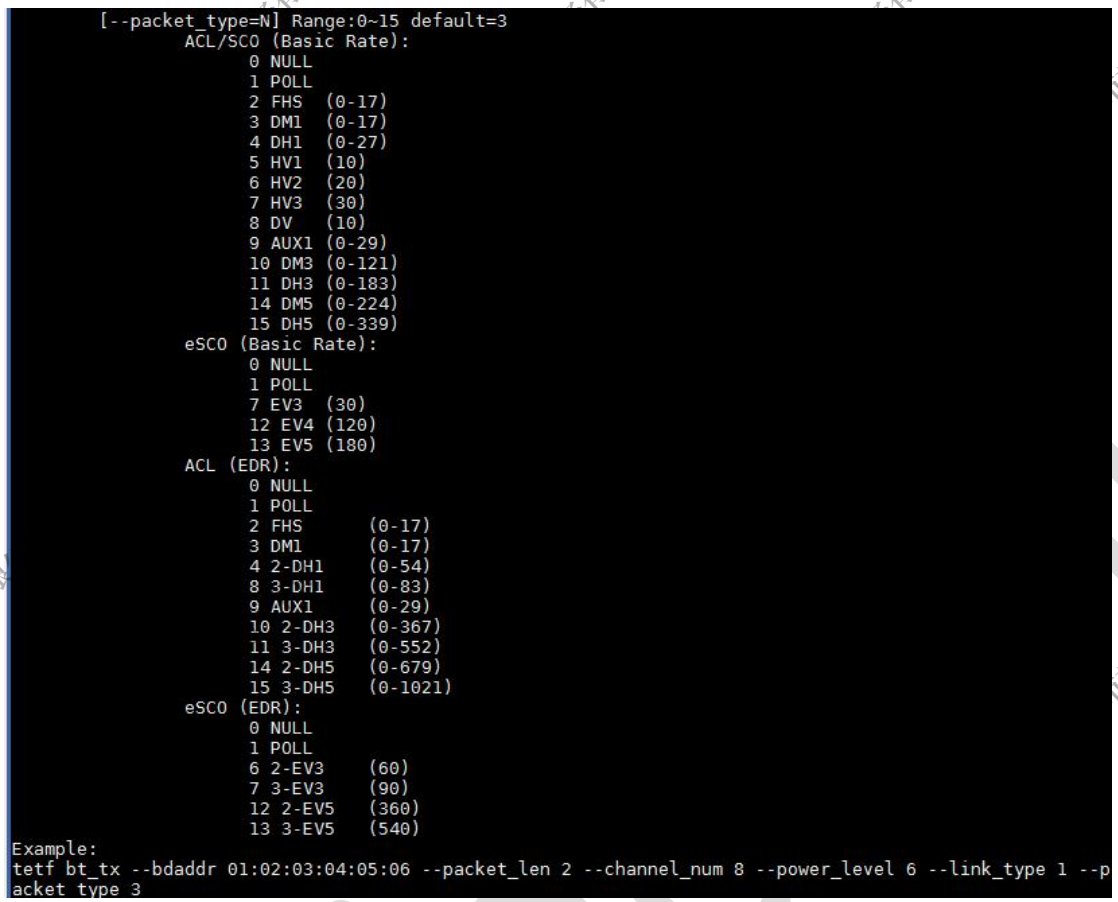


Figure 4-4 Use "btetf bt_tx -h" to get BT TX test help

Parameters needed to be specified for BT TX test are as follows. Channel, link type and packet type should be consistent with RX Test:

Default parameter list	Sample Value	Remarks
bdaddr	11:22:33:44:22:22	Mac Address
pattern	6	Data Type
packet_len	37	Packet Length
channel_num	1	Channel
power_level	6	Power
link_type	0	Link Type
packet_type	4	Packet Type
hopping_mode	0	Close Hopping mode

Example: Specify BT TX test to transmit DH1 packet on channel 1 with packet length of 37 bytes.

```
venus-a3:/# btetf -i hci0 -d  bt_tx --bdaddr 11:22:33:44:22:22 --pattern 6 --packet_len 37 --channel_num 1
--power_level 6 --link_type 0 --packet_type 4 --hopping_mode 0
```

```
venus-a3:/ # btetf -i hci0 -d  bt_tx --bdaddr 11:22:33:44:22:22 --packet_len 10 --channel_num 8 --powe
r_level 6 --link_type 0 --packet_type 3
n 10 --channel_num 8 --power_level 6 --link_type 0 --packet_type 3            <
bt_tx: pattern(0)-packet_len(10)-channel_num(8)-power_level(6) link_type/packet_type(0-3) lpt:(0x 3)
< HCI Command: opcode:(0xfc50) ogf-ocf:(0x3f-0x0050) plen:(13)
   00 11 22 33 44 22 22 00 0A 00 08 06 03
> HCI Event: 0x0e plen 17
   05 50 FC 00 00 11 22 33 44 22 22 00 0A 00 08 06 03
```

Figure 4-5 Start BT TX test example

Example: Specify BT TX test to transmit 3-DH5 packet on channel 1 with packet length of 37 bytes.

```
venus-a3:/# btetf -i hci0 -d  bt_tx --bdaddr 11:22:33:44:22:22 --pattern 6 --packet_len 37 --channel_num 1
--power_level 6 --link_type 2 --packet_type 15 --hopping_mode 0
```

4.1.4 Stop BT TX Test

Use the following command to stop BT TX test:

```
venus-a3:/# btetf -i hci0 -d  bt_close_tx
```

```
venus-a3:/ # btetf -i hci0 -d  bt_close_tx
btetf -i hci0 -d  bt_close_tx
< HCI Command: opcode:(0xfc50) ogf-ocf:(0x3f-0x0050) plen:(1)
   F0
> HCI Event: 0x0e plen 5
   05 50 FC 00 F0
```

Figure 4-6 Stop BT TX test

4.2 BLE Test

4.2.1 Start BLE RX Test

You can use " btetf ble_rx -h " to see the usage of " ble_rx " command:

```
venus-a3:/# btetf ble_rx -h
```

```
venus-a3:/ # btetf ble_rx -h
btetf ble_rx -h
ble_rx: unrecognized option: h
Usage:
        ble_rx [option] [parameters]
Options:
       [--rx_channel=N] Range:0~27 Frequency Range:2402 MHz to 2480 MHz
       [--auto] Auto test mode, must connect measuring device first
venus-a3:/ #
```

Figure 4-7 Use "btetf ble_rx -h" to get BLE RX test help

Parameters needed to be specified for BLE RX test are as follows. Channel should be consistent with RX Test:

Default parameter list	Sample Value	Remarks
channel	1	Channel

Example: Listening Ble packets on channel 5:

```
venus-a3:/# btetf -d -i hci0 ble_rx --rx_channel 5
```

```
venus-a3:/ # btetf -d -i hci0 ble_rx --rx_channel 5
btetf -d -i hci0 ble_rx --rx_channel 5
ble_rx:rx_channel:5
< HCI Command: opcode:(0x201d) ogf-ocf:(0x08-0x001d) plen:(1)
   05
> HCI Event: 0x0e plen 4
   05 1D 20 00
```

Figure 4-8 Start BLE RX test example

4.2.2 Stop BLE RX Test

Use the following command to stop BLE RX test:

```
venus-a3:/# btetf -d -i hci0 ble_close
```

When BLE RX test is stopped, BLE RX test results will be displayed (The red box of the following picture shows that device has received 3744 BLE packages):

```
venus-a3:/ # btetf -d -i hci0 ble_close
btetf -d -i hci0 ble_close
< HCI Command: opcode:(0x201f) ogf-ocf:(0x08-0x001f) plen:(0)
> HCI Event: 0x0e plen 6
   05 1F 20 00 A0 0E
event: 04 0E 06 05 1F 20 00 A0 0E
status:0x 0
send_or_receive_packet:3744
venus-a3:/ #
```

Figure 4-9 Stop BLE RX test, showing the number of packets received

4.2.3 Read RSSI After BLE RX Test finished

Use the following command to Read RSSI after BLE RX Test is finished:

```
venus-a3:/# btetf -i hci0 -d    ble_rssi
```

```
root@TinaLinux:/# btetf -d -i hci0 ble_rssi
btetf -d -i hci0 ble_rssi
< HCI Command: opcode:(0xfc15) ogf-ocf:(0x3f-0x0015) plen:(1)
   00
> HCI Event: 0x0e plen 8
   05 15 FC 00 00 00 00 DF
event: 04 0E 08 05 15 FC 00 00 00 00 DF
status:0x 0
ble_rssi_result:-32
root@TinaLinux:/#
```

Figure 4-10 Read RSSI after BLE RX test stopped

4.2.4 Start BLE TX Test

You can use " btetf ble_tx -h " to see the usage of " ble_tx " command:

```
venus-a3:/# btetf ble_tx -h
```

```
venus-a3:/ # btetf ble_tx -h
btetf ble_tx -h
ble_tx: unrecognized option: h
Usage:
        ble_tx [option] [parameters]
Options:
       [--tx_channel=N] Range:0~39 Frequency Range:2402 MHz to 2480 MHz
       [--len=N] Range:0~37 Length in bytes of payload data in each packet
       [--payload=N] Range:0~7 Type of payload
                     0x00 Pseudo-Random bit sequence 9
                     0x01 Pattern of alternating bits '11110000'
                     0x02 Pattern of alternating bits '10101010'
                     0x03 Pseudo-Random bit sequence 15
                     0x04 Pattern of All '1' bits
                     0x05 Pattern of All '0' bits
                     0x06 Pattern of alternating bits '00001111'
                     0x07 Pattern of alternating bits '0101'
       [--auto] Auto test mode, must connect measuring device first
venus-a3:/ #
```

Figure 4-11 Use "btetf ble_tx -h" to get BLE TX test help

Parameters needed to be specified for BLE TX test are as follows. Channel ,length, payloadshould be consistent with RX Test:

Default parameter list	Sample Value	Remarks
channel	1	Channel
len	37	Packet Length
payload	0	Data Type

Example: Specify to send BLE packets on channel 1 with packet length of 37 bytes and random data padding:

```
venus-a3:/# btetf -d -i hci0 ble_tx --tx_channel 1 --len 37 --payload 0
```

```
root@TinaLinux:/# btetf -d -i hci0 ble_tx --tx_channel 5 --len 10 --payload 0
btetf -d -i hci0 ble_tx --tx_channel 5 --len 10 --payload 0
ble_tx:tx_channel:5, tx_data_len:10, packet_payload:0
< HCI Command: opcode:(0x201e) ogf-ocf:(0x08-0x001e) plen:(3)
   05 0A 00
> HCI Event: 0x0e plen 4
   05 1E 20 00
```

Figure 4-8 Start BLE RX test example

4.2.5 Stop BLE TX Test

Use the following command to stop BLE TX test:

```
venus-a3:/# btetf -d -i hci0 ble_close
```

```
root@TinaLinux:/# btetf -d -i hci0 ble_close
btetf -d -i hci0 ble_close
< HCI Command: opcode:(0x201f) ogf-ocf:(0x08-0x001f) plen:(0)
> HCI Event: 0x0e plen 6
   05 1F 20 00 00 00
event: 04 0E 06 05 1F 20 00 00 00
status:0x 0
send_or_receive_packet:0
root@TinaLinux:/#
```

Figure 4-13 Stop BLE RX test

4.3 BT Single Tone Test

You can use " btetf ble_tx -h " to see the usage of " ble_tx " command:

```
venus-a3:/# btetf single_tone --help
```

```
venus-a3:/ # btetf single_tone --help
btetf single_tone --help
Usage:
        single_carrier [option]
Options:
        [--open channel_num]
        [--power_level level]
        [--close]
Example:
tetf single_tone --opne 0 --power_level 6
```

Figure 4-14 Use "btetf single_tone --help" to get BT single Tone Test help

Parameters needed to be specified for BT Single Tone test are as follows.

Default parameter list	Sample Value	Remarks
open	1	channel
power_level	6	Power

close	NULL	close

Example: Open Single Tone Test on channel 1:

```
venus-a3:/# btetf -d -i hci0 single_tone --open 1
```

Example: Close Single Tone Test:

```
venus-a3:/# btetf -d -i hci0 single_tone --close
```

5 FAQ

5.1 Fail To configure the test environment

If the configuration of the test environment fails, first follow chapter 2.3 “Checkpoints for the noraml use of the tool” to check and then follow the steps below.

5.1.1 Can’t Open Serial Port

When Run script “init_test.bat” to configuring the test environment, problem point “Can’t open serial port” appear:

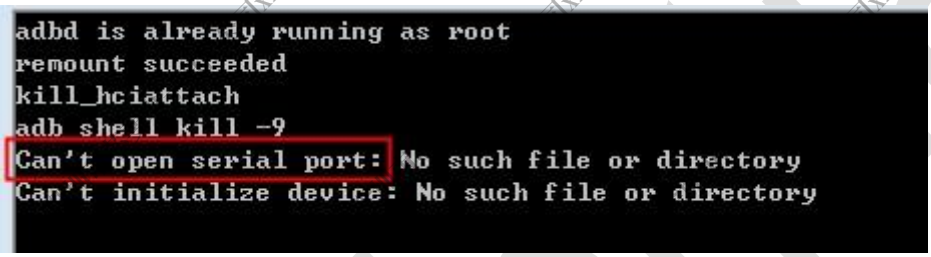


Figure 5-1 Problem “Can’t open serial port”

Please confirm whether the tty number in the script "init_test.bat" is the tty number corresponding to Bluetooth :

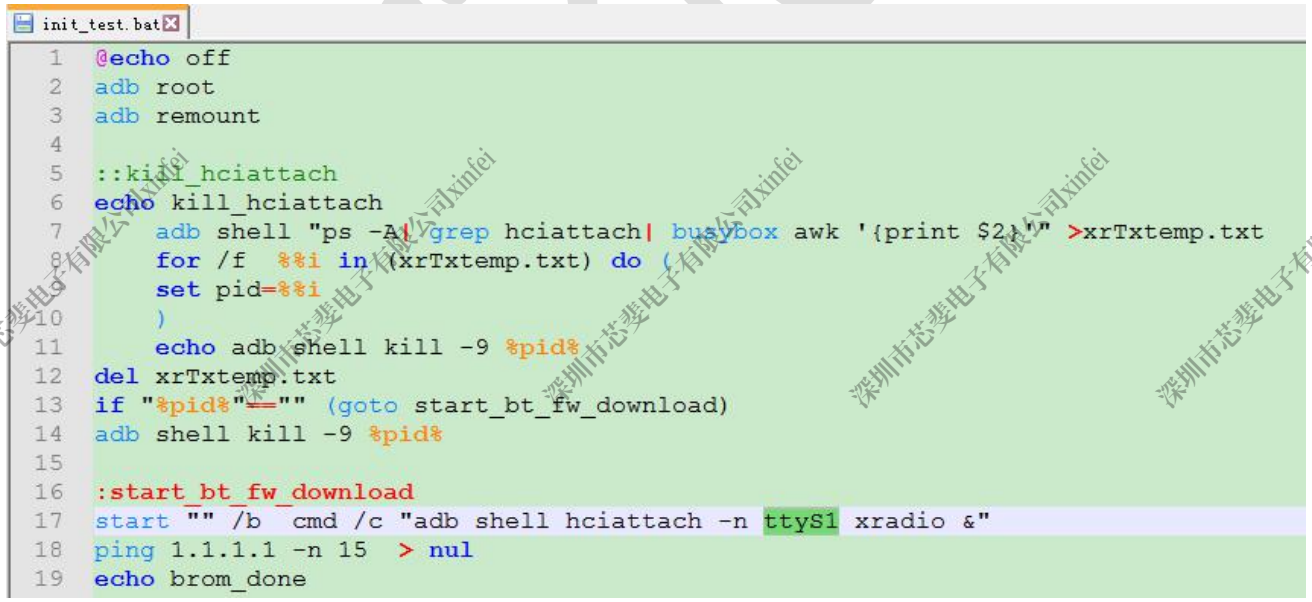


Figure 5-2 Confirm tty number in the script “init_test.bat”

Among them, the tty number corresponding to Bluetooth can be obtained by viewing the /dev/ directory.

```
crw-------  1 root      root            10,  59 2019-04-28 10:48 sunxi-wlan
crw-------  1 root      root            10,  61 2019-04-28 10:48 sunxi_soc_info
crwxrwxrw-  1 media     mediadrm       250,   0 2019-04-28 10:48 tee0
crwxrwxrw-  1 media     mediadrm       250,  16 2019-04-28 10:48 teepriv0
crw-rw-rw-  1 root      root             5,   0 2019-04-28 10:48 tty
crw-------  1 root      root           247,   0 2019-04-28 10:48 ttyS0
crw-rw----  1 bluetooth net_bt_admin   247,   1 2019-04-28 10:50 ttyS1
crw-rw----  1 system    vpn            10, 200 2019-04-28 10:48 tun
crw-rw----  1 uhid      uhid           10, 239 2019-04-28 10:48 uhid
crw-rw----  1 system    bluetooth      10, 223 2019-04-28 10:48 uinput
crw-rw-rw-  1 root      root             1,   9 2019-04-28 10:48 urandom
drwxrwx---  3 shell     shell                60 2019-04-28 10:48 usb-ffs
crw-rw----  1 root      usb            10,  46 2019-04-28 10:48 usb_accessory
crwxrwx---  1 camera    camera         81,   1 2019-04-28 10:48 v4l-subdev0
crwxrwx---  1 camera    camera         81,   2 2019-04-28 10:48 v4l-subdev1
crwxrwx---  1 camera    camera         81,   3 2019-04-28 10:48 v4l-subdev2
crwxrwx---  1 camera    camera         81,   4 2019-04-28 10:48 v4l-subdev3
crwxrwx---  1 camera    camera         81,   5 2019-04-28 10:48 v4l-subdev4
crwxrwx---  1 camera    camera         81,   6 2019-04-28 10:48 v4l-subdev5
crwxrwx---  1 camera    camera         81,   7 2019-04-28 10:48 v4l-subdev6
crwxrwx---  1 camera    camera         81,   8 2019-04-28 10:48 v4l-subdev7
crwxrwx---  1 camera    camera         81,   9 2019-04-28 10:48 v4l-subdev8
crwxrwx---  1 camera    camera         81,   0 2019-04-28 10:48 video0
crw-rw-rw-  1 root      root           10,  53 2019-04-28 10:48 vndbinder
crw-r--r--  1 root      root           10,  52 2019-04-28 10:48 xt_qtaguid
crw-rw-rw-  1 root      root             1,   5 2019-04-28 10:48 zero
venus-a3:/dev #
```

Figure 5-3 View the tty number corresponding to the Bluetooth

5.1.2 Fail To Load FW

Brom sync will be performed first during Bluetooth boot, if Brom sync failed, load firmware exception will occur. If the console keep printing "uart sync count: x" as shown below, it means Brom sync failed:

```
venus-a3:/ # hciattach -n ttyS1 xradio &
hciattach -n ttyS1 xradio &
[1] 11395
venus-a3:/ # xradio_init
set LPM mode:disabled[userial_sync] uart sync count: 1.
[userial_sync] read buf: 00 00.
[userial_sync] uart sync count: 2.
[userial_sync] read buf: 00 00.
[userial_sync] uart sync count: 3.
[userial_sync] read buf: 00 00.
[userial_sync] uart sync count: 4.
[userial_sync] read buf: 00 00.
[userial_sync] uart sync count: 5.
[userial_sync] read buf: 00 00.
```

Figure 5-4 Brom sync failed

The possible reasons are as follows:

- Bluetooth firmware has been loaded

First confirm that Bluetooth is turn off in the Android Settings interface (or confirm that Linux systerm Bluetooth is turned off). Then check if there is a Hciattch process, if there is, do not repeat loading.

```
venus-a3:/ # ps -A | grep hciattach

venus-a3:/ # kill -9 PID
```

```
venus-a3:/ # ps -A | grep hciattach
ps -A | grep hciattach
root          3398 27197     900    644 poll_schedule_timeout 0 S hciattach
venus-a3:/ #

venus-a3:/ #

venus-a3:/ # kill -9 3398
kill -9 3398
[1] + Killed                 \hciattach -n ttyS1 xradio
venus-a3:/ #
```

Figure 5-5 Check if Bluetooth firmware has been loaded

- Chip core is not reset

Use “cat /sys/class/rfkill/rfkill0/state” command to see whether the power level of “ BT_RST ” is 1:

```
venus-a3:/sys/class/rfkill/rfkill0 # cat state
cat state
1
```

Figure 5-6 Check if Chip core has been reset

- Chip is not wake up

Use “ echo 1 > /proc/bluetooth/sleep/btwake ” command to check.

```
venus-a3:/proc/bluetooth/sleep # cat btwake
cat btwake
bt wake state:1
```

Figure 5-7 Check if chip has been wake up