Configuring the BT800 in Linux and Android

Quick Start Guide *v1.0*

Note: This Quick Start Guide is relevant to Laird's BT800, BT810 and BT820 Bluetooth modules.

INTRODUCTION

The Linux operating system, and Android by extension, has excellent USB support for a variety of devices. This makes it very easy to connect and configure Laird's BT800-series Bluetooth modules in Linux or Android.

This guide illustrates how to connect and configure a BT800-series radio to a host machine which is running Linux or Android, as well as how to test Bluetooth and Bluetooth Low Energy connectivity.

This guide assumes that you have already physically connected the BT800 to the host PC. For information on hardware integration, see the BT800 Hardware Integration Guide.

REQUIREMENTS

This guide uses the BT800 module as an example. You will need the following items:

- Linux PC or Android device
- BlueZ Linux Bluetooth protocol stack v4.101 or later
- Laird BT800 Bluetooth module

CONNECT THE BT800 VIA COMMAND LINE

Verify BT800 Connection

In Linux, you may configure and test the BT800 via terminal. The Linux utility to configure and identify Bluetooth is hcitool. A full list of hcitool commands is found in Table 1 of Appendix I: HCITOOL and GATTTOOL Commands. To verify that the BT800 is recognized by the operating system, run hcitool and check for devices as follows:

- Open the command terminal.
- Enter the following command:

  ```
  hcitool dev
  ```

This command displays local devices. If it finds one, it returns the following:

```
Devices: hci0 [MAC Address]
```

This MAC address should correspond to the MAC address of the BT800 module.

Note: The hcitool command uses the first available Bluetooth device for its operations. If multiple Bluetooth devices are found, all hcitool commands must specify which device to use, as follows:

```
hcitool [-i <hciX>] [command [command parameters]]
```

In this example, <hciX> must correspond to the HCI device number found using `hcitool dev`.

Scan and Connect with Classic Bluetooth

With the device initialized, you may test Bluetooth functionality from the command prompt. To test scanning, you must have a nearby device (such as a tablet or smartphone) set to be discoverable.

The command to initialize a scan is:

```
hcitool scan
```

When a scan is initialized, the terminal returns found devices in the following format:

```
Scanning ...
          [MAC Address]   Friendly_Name
```

If there are discoverable devices nearby, they appear in this list as they are discovered.

To connect to a device, use the hidd command as follows:

```
hidd --connect xx:xx:xx:xx:xx:xx
```

In this example, xx:xx:xx:xx:xx:xx is the MAC address of the device you wish to connect to.

Scan and Connect with Bluetooth Low Energy

The hcitool commands to scan and connect with Bluetooth Low Energy are distinct from those used in classic Bluetooth connections. They are listed in Table 2 of Appendix I: HCITOOL and GATTTOOL Commands.

To initiate a BLE scan from the terminal, issue the following command:

```
#hcitool lescan
```

The terminal returns the following:

```
LE Scan ...
[MAC Address] – [BLE device]
```

To connect to a found device, use the lecc command as follows:

```
hcitool lecc <MAC Address>
```

Connecting with Gatttool

The BlueZ software stack includes a utility called Gatttool that allows you to connect, gather advertised data, and other GATT-specific BLE operations. A full list of gatttool commands is found in Table 3 shows commands for gatttool.

Table 3 of Appendix I: HCITOOL and GATTTOOL Commands.

To connect with the gatttool command, you must first discover another device by running the lescan as follows:

```
#hcitool lescan
```

Note the MAC address of the returned device that you wish to connect to. Then, use the gatttool to connect to the device with the following command:

```
gatttool -b <MAC Address> -I
```

This command connects the BT800 to the device with the specified MAC address.

Note: Some BLE devices, including the BL600, come programmed with a random MAC address. To connect to them for the first time, you must use the option *–t random* in the gatttool command. The command is written as follows:

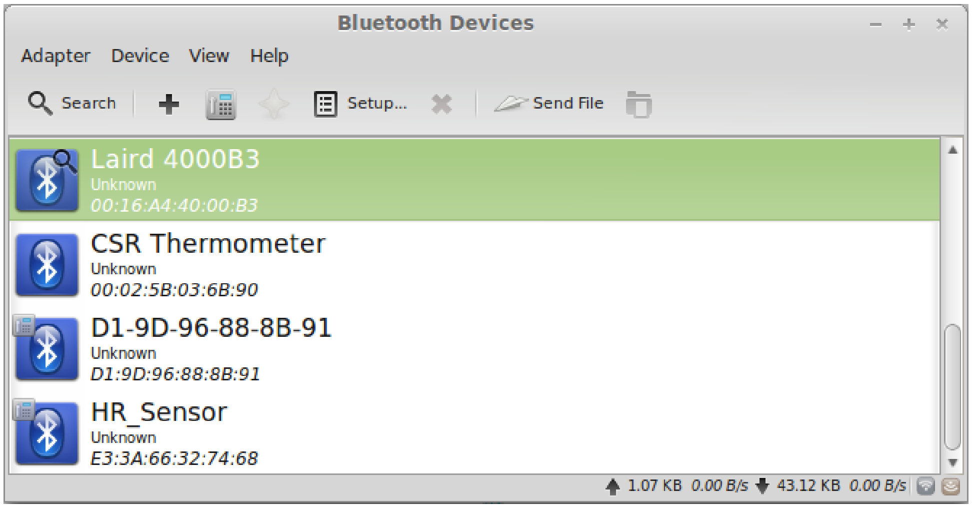
```
Gatttool –t random –b <MAC ADDRESS> -I
```

CONNECT THE BT800 VIA GUI

Using the BT800 from the Linux GUI

If the BT800 is recognized by hcitool, you may use the Linux GUI to establish connections with the BT800.

Note: The example shown in Figure 1 is captured from Linux Mint 14 Nadia. The procedure may vary slightly between different desktop environments.



Laird 4000B3
Unknown
00:16:A4:40:00:B3

CSR Thermometer
Unknown
00:02:5B:03:6B:90

D1-9D-96-88-8B-91
Unknown
D1:9D:96:88:8B:91

HR_Sensor
Unknown
E3:3A:66:32:74:68

Figure 1: Bluetooth applet in Linux Mint 14 Nadia - Kernel version 3.5.0-17-generic

The built-in Bluetooth management software of your OS allows you to connect to remote devices with the BT800. From the Bluetooth manager, you may scan, pair, connect, and disconnect with Bluetooth devices.

Using the BT800 from the Android GUI

The Android user interface has a built-in Bluetooth device management menu for discovering, pairing, and connecting to remote devices. As of Android 4.0 ICS, this is found in the System Settings menu (Figure 2) by tapping **Bluetooth**. This opens the Bluetooth Devices menu (Figure 3). From here, you can search for devices, pair or unpair with found devices, change Bluetooth settings, and enable or disable the Bluetooth radio.

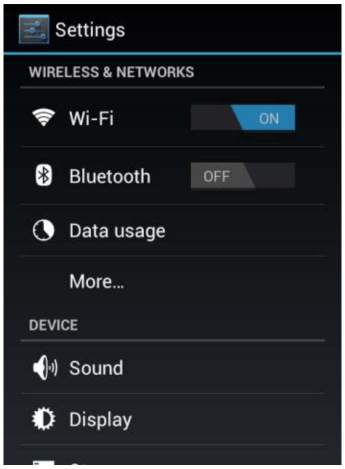


Figure 2: Android Settings Menu in Android v4.2 Jellybean

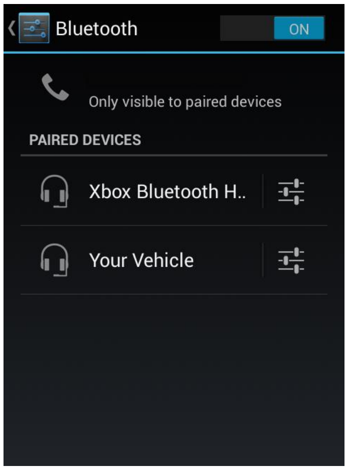


Figure 3: Bluetooth Devices Menu in Android v4.2 Jellybean

APPENDIX I: HCITOOL AND GATTTOOL COMMANDS

The Linux utility to configure and identify Bluetooth is hcitool. The usage for hcitool is:

```
hcitool [-i <hciX>] [command [command parameters]]
```

If –i option is specified, device <hciX> must A list of commands for *hcitool* is shown in Table 1: General and Classic BT Commands in hcitool Additionally, Bluetooth Low Energy commands are shown in Table 2.

Table 1: General and Classic BT Commands in hcitool

Command	Description	Usage
dev	Display local devices	*hcitool dev*
inq	Inquire remote devices	*hcitool inq [--length=N] maximum inquiry duration in 1.28 s units* [--numrsp=N] specify maximum number of inquiry responses [--iac=lap] specify the inquiry access code [--flush] flush the inquiry cache
scan	Scan for remote devices	*hcitool scan [--length=N] [--numrsp=N] [--iac=lap] [--flush] [--class] [--info] [--oui] [--refresh]*
name	Get name from remote device	*hcitool name <bdaddr>*
info	Get information from remote device	*hcitool info <bdaddr>*
spinq	Start periodic inquiry	*hcitool spinq*
epinq	Exit periodic inquiry	*hcitool epinq*
cmd	Submit arbitrary HCI commands	*hcitool cmd <ogf> <ocf> [parameters]* *e.g.* - cmd 0x03 0x0013 0x41 0x42 0x43 0x44
con	Display active connections	*hcitool con*
cc	Create connection to remote device	*hcitool cc [--role=m|s] [--ptype=pkt_types] <bdaddr>* *Example:* cc --ptype=dm1,dh3,dh5 01:02:03:04:05:06 cc --role=m 01:02:03:04:05:06
dc	Disconnect from remote device	hcitool dc <bdaddr> [reason]
sr	Switch master/slave role	hcitool sr <bdaddr> <role>
cpt	Change connection packet type	hcitool cpt <bdaddr> <packet_types>
rssi	Display connection RSSI	hcitool rssi <bdaddr>
lq	Display link quality	hcitool lq <bdaddr>
tpl	Display transmit power level	*hcitool tpl <bdaddr> [type]*
afh	Display AFH channel map	hcitool afh <bdaddr>
lp	Set/display link policy settings	hcitool lp <bdaddr> [link policy]
lst	Set/display link supervision timeout	hcitool lst <bdaddr> [new value in slots]
auth	Request authentication	hcitool auth <bdaddr>
enc	Set connection encryption	hcitool enc <bdaddr> [encrypt enable]
key	Change connection link key	hcitool key <bdaddr>
clkoff	Read clock offset	hcitool clkoff <bdaddr>
clock	Read local or remote clock	hcitool clock [bdaddr] [which clock]

Table 2: BLE commands in hcitool

Command	Description	Usage
lescan	Start LE scan	*hcitool lescan [--privacy] enable privacy* *hcitool lescan [--passive] set scan type passive (default)* *hcitool lescan [--discovery=g|l] enable general or limited discoveryprocedure* *hcitool lescan [--duplicates] don't filter duplicates*
lewladd	Add device to LE White List	*hcitool lewladd [--random] <bdaddr>*
lewlrm	Remove device from LE White List	hcitool lewlrm <bdaddr>
lewlsz	Read size of LE White List	hcitool lewlsz
lewlclr	Clear LE White list	hcitool lewlclr
lecc	Create a LE Connection	*hcitool lecc [--random] <bdaddr>* *hcitool lecc –whitelist*
ledc	Disconnect a LE Connection	hcitool ledc <handle> [reason]
lecup	LE Connection Update	hcitool lecup <handle> <min> <max> <latency> <timeout> Options: *-H, --handle <0xXXXX> LE connection handle* *-m, --min <interval> Range: 0x0006 to 0x0C80* *-M, --max <interval> Range: 0x0006 to 0x0C80* *-l, --latency <range> Slave latency. Range: 0x0000 to 0x03E8* *-t, --timeout <time> N * 10ms. Range: 0x000A to 0x0C80* min/max range: 7.5ms to 4s. Multiply factor: 1.25ms *timeout range: 100ms to 32.0s. Larger than max interval*

Table 3 shows commands for gatttool.

Table 3: Commands for gatttool

Command	Description
gatttool -b <MAC Address> --primary	Discover available services.
gatttool -b <MAC Address> --characteristics	Discover characteristics.
gatttool –b <MAC Address> --char-desc	Discover characteristic descriptors
gatttool -b <MAC Address> --char-read --uuid=XXXX	Read characteristic values.
gatttool -b <MAC Address> --char-write --uuid=XXXX --value=XXXX	Write to characteristic values.
gatttool -b <MAC Address> --char-write-req --handle=0xXXXX --value=0100 --listen	Enable notifications.
gatttool -b <MAC Address> --char-write-req --handle=0xXXXX --value=0200 --listen	Enable indications.
gatttool -b <MAC Address> --interactive	Enter interactive mode.

Revision History

Revision	Date	Description	Approved By

1.0	4 Sep 2014	Initial Release	Jonathan Kaye