AT Interface *smart*BASIC Application

BL652/BL654

Quick Start Guide *v1.0*

1 Introduction

Laird's BL652 and BL654 modules are deployed with embedded *smart*BASIC. *smart*BASIC is an event-driven programming language that does not require complex toolchains and allows the quick creation of custom applications. The language provides an easy to use API to the underlying Bluetooth, NFC and other hardware and it in turn will generate events that the developer can service through handlers written in smart*BASIC*.

This guide assumes that you are familiar with loading smart*BASIC* applications into the module. If not, please refer to the application note Loading smartBASIC Applications which is available from the BL654 product page: https://www.lairdtech.com/products/bl654-ble-thread-nfc-modules

This quick start guide explains how to perform the following:

- Load the AT Interface application into two devices
- Make a BLE Virtual Serial Port connection
- Send the text *hello* from one device to the second
- Send the text *world* in the reverse direction
- Disconnection

2 Requirements

To perform the steps in this document, you need the following:

- Two devices which could be any one of a DVK-BL652 or DVK-BL654 development board or BL654 USB dongles.
- USB-A to USB-micro cable (included with development board) which is not needed if you use a BL654 USB dongle.
- UwTerminalX (v1.10a or later recommended), available at https://github.com/LairdCP/UwTerminalX/releases
- BL652 *smart*BASIC sample applications, available at https://github.com/LairdCP/BL652-Applications or https://github.com/LairdCP/BL654-Applications in which you will find a folder called 'ATinterface' which contains the application that will need to be loaded.
- (Recommended) The latest firmware for your module which can be found on the BL652 page: http://www.lairdtech.com/products/bl652-ble-module or BL654 page: https://www.lairdtech.com/products/bl654-ble-thread-nfc-modules under the 'Software' section

3 Hardware and Terminal Setup

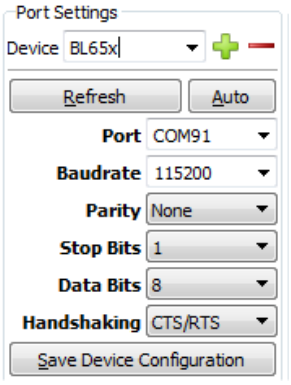
To begin, you must physically connect the two devices to your PC and connect to them via UwTerminalX.
To do so, complete the following:

1. Connect the USB dongle or development board to your PC via the included USB micro cable.
2. Power your development board (SW4).
3. Launch UwTerminalX.
4. From the Update tab within the UwTerminalX pane, click **Check for Updates** to ensure you're using the latest version of UwTerminalX with support for the BL652 or BL654.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

5. From the Config tab in the Device drop-down menu, select **BL65X**.

Note: The BL65X option may not appear in the drop-down menu in older versions of UwTerminalX. In these earlier versions, you must manually enter the correct settings (115200, N, 1, 8) as shown below:



6. Select the correct port to which your development board or USB dongle is connected.
7. Click **OK** to advance to the Terminal tab and check that the CTS solid circle is green and not red.
8. Press **Enter** on your keyboard. If you see the return *00*, you are successfully communicating over UART.

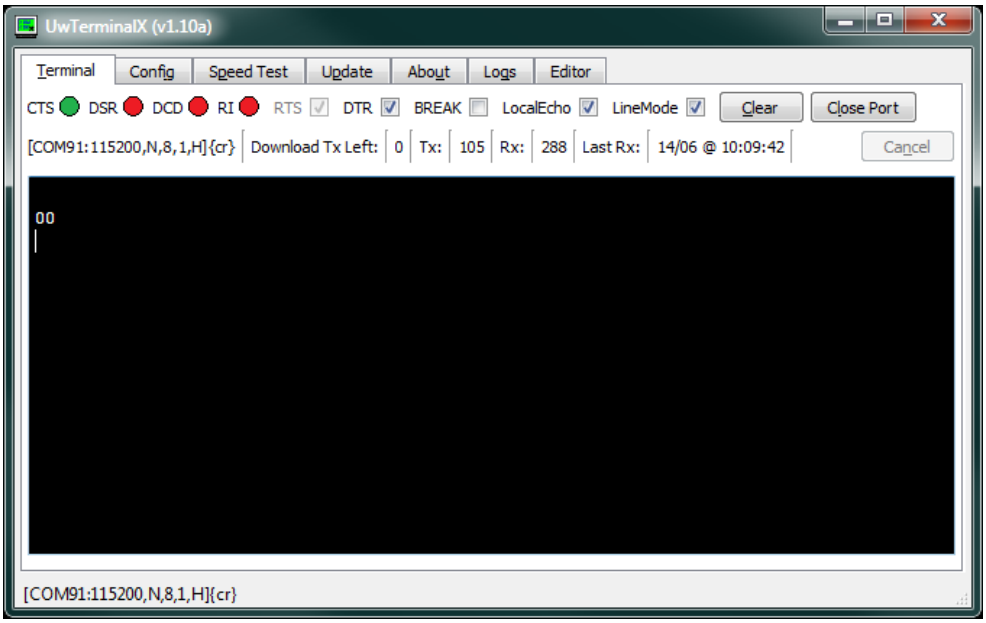


Figure 1: Terminal returns 00 if connected successfully

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

4 Compiling, Loading, and Running The AT Application

Once you are successfully connected to the DVK-BL652, DVK-BL654, or BL654 USB dongle over UwTerminalX, you may compile, load, and run the ATinterface application.

Note: For cross-compiling *smart*BASIC applications, UwTerminalX must use a version-specific XCompiler that matches the firmware of your module. UwTerminalX, however, provides the option for using an online XCompiler. For supported firmware versions, this means that UwTerminalX finds the correct XCompiler over the Internet so that you don't need to maintain the XCompiler on your machine. This is enabled by default and can be disabled in the *Config* tab.

For this quick start guide, we use the *ATinterface* application in the BL652 or BL654 applications GitHub library. To compile, load, and run this application, complete the steps below.

If you are using the BL652, then the application to download is called **$autorun$.AT.interface.BL652.sb** and if you are using the BL654, then the application to download is called **$autorun$.AT.interface.BL654.sb**. For convenience, given that the behavior is the same for both module types, we will just refer to the application as the *ATinterface autorun* application.

1. Download the ATinterface autorun application from the appropriate applications library (https://github.com/LairdCP/BL652-Applications/tree/master/ATinterface or https://github.com/LairdCP/BL654-Applications/tree/master/ATinterface) to a directory on your PC. The application is located in the ATinterface subfolder of your local repository folder.
2. If you are not already, connect to the BL652 or BL654 in UwTerminalX.
3. Right-click in the terminal window and, in the context menu, click **XCompile + Load**.
4. Depending on the type of module to which you are downloading, in the file selector window, select **$autorun$.AT.interface.BL652.sb** or **$autorun$.AT.interface.BL654.sb** and click **Open**.
5. When the terminal displays 00, the compiler has finished successfully.
6. Type **at+dir** and press **Enter**. You should see $autorun$ in the file list.
7. Perform steps 1 to 6 again but on the second device so that you have both modules loaded with the ATinterface application.
8. To run the script, type **$autorun$** and press **Enter** or reset the module by ticking and unticking the BREAK checkbox. The latter which results in a reset of the module will mean that on startup it will find the specially named $autorun$ application and automatically run it.
9. Press **Enter** in the terminal. If the terminal returns OK, the module is now running the ATinterface application and will process AT commands sent to it.
   If the terminal returns **00**, if you started the application by using the BREAK checkbox, ensure that the DTR checkbox is ticked and then tick/untick BREAK again.
10. In UWTerminalX, type **AT**, press **Enter**, and confirm that you see the following:

```
AT

OK
```

11. Type **ATI0**, press **Enter** and confirm you see the following:

```
ATI0

BL654
OK
```

    The returned module type could be the BL652 or the BL654.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

12. On the second module, type **ATI4**, press **Enter**, and confirm that you see the following:

```
ati4

01EA06E7EF3E5F
OK
```

Note: The string (01EA06E7EF3E5F, in the example above) is the 14-hex character Bluetooth address of the second device. Your string will be a different hex string value.

Going forward, we refer to this 14-character hex string as **<mac_address>**. For your purposes, substitute this with the appropriate values for your devices.

At this point both devices are configured to accept an incoming Virtual Serial Port (VSP) connection and are advertising.

13. To check this on the first module, type **AT+LSCN** and press **Enter**. The module scans for adverts for about ten seconds and displays any adverts received.

```
AD1:0 01EA06E7EF3E5F -23 "LAIRD BL654-EF3E5F"

AD1:0 01EA06E7EF3E5F -23 "LAIRD BL654-EF3E5F"

AD1:0 01EA06E7EF3E5F -26 "LAIRD BL654-EF3E5F"

OK
```

01EA06E7EF3E5F and LAIRD BL654-EF3E5F are specific to this example. Your numbers will be different.

14. Enter **AT+LSCN** on the second module. The first module is also advertising similarly.

The next step is to initiate a BLE connection from the first to the second module.

Note: Because of the ATinterface app design, this procedure is symmetrical and works either way.

15. On the first module, enter **ATD<mac_address>.**
16. After a few seconds on the first device's UwTerminalX screen, confirm that you see the following:

```
ATD01EA06E7EF3E5F
CONNECT 0,01EA06E7EF3E5F,15000,6000000,0
```

17. On the second device's UwTerminalX screen, confirm that you see the following:

```
RING 01C7231E370E06,U
CONNECT 0,01C7231E370E06,15000,6000000,0
```

On both sides, the **CONNECT** response is as follows:

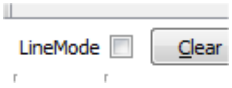
`CONNECT 0,01EA06E7EF3E5F,15000,6000000,0`

Where…

01EA06E7EF3E5F	The address of the peer device
15000	The connection interval
6000000	The link supervision timeout
0	The slave latency

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

18. On the first device, type **hello**, press **Enter**, and confirm you see that string appear at the second device.
19. On the second device, type **world**, press **Enter**, and confirm you see that string appear at the first device.
20. Disconnect. Again, you can initiate this from either end – it's up to you to choose.
21. On the side you choose, ensure that the LineMode checkbox is not ticked (this is essential, otherwise the escape sequence to enter AT command mode will not be recognised as it looks for idle delays)



22. Type the ^ character four times (ensure there is at least half a second delay between each ^ character being typed).

    The following displays when it is successfully disconnected:

```
NOCARRIER 90
```

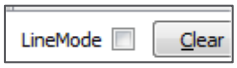
    After a disconnection, both devices automatically return to advertising mode so that another connection can either be accepted or initiated.

5 Notes

1. In the previous section (steps 18 and 19) where **hello<enter>** and **world<enter>** were exchanged, the messages only appeared at the remote end when you pressed Enter.

   That occurred because UwTerminalX was configured for Line Mode and not because the module behaves like that inherently.

   In UwTerminalX, if you have LineMode unticked, each key is immediately transmitted as you type them.



2. When a connection is established, both ends also automatically stop advertising.

3. A complete AT Interface user guide is available from the BL652 product page:

   https://www.lairdtech.com/products/bl654-ble-thread-nfc-modules

6 Conclusion

Laird's smart*BASIC* ATinterface application is available for use in your design. This can accelerate and simplify your development of wireless by enabling an existing system that sends serial data over a wired connection. You are free and encouraged to modify the source of that application to suit your needs.
UwTerminalX, sample BL652/BL654 applications, and more are available from the following links:

https://github.com/LairdCP/UwTerminalX

http://www.lairdtech.com/products/bl652-ble-module

https://github.com/LairdCP/BL652-Applications

https://www.lairdtech.com/products/bl654-ble-thread-nfc-modules

https://github.com/LairdCP/BL654-Applications

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

7 Revision History

Version	Date	Notes	Contributor(s)	Approver
1.0	20 June 2018	Initial Release	Youssif M. Saeed	Jonathan Kaye