APPENDIX E

Using a Host Terminal Emulator

E.1 Introducing Host Terminal Applications

A terminal emulator allows a host computer to send or receive data from another computer, through a variety of links. Terminal emulators simply take the form of a software package running on the host computer which creates a screen image on the computer screen, through which settings can be selected. It then provides a context for data transfer, using the host computer keyboard for data input. Such a terminal is particularly useful for displaying messages from the mbed to a computer screen. It can become a very useful tool for user interfacing and software debugging. For example, status or error messages can be embedded into a program running on the mbed, and transferred to the terminal emulator when certain points in the program are reached.

While a number of terminal emulators are available, Tera Term is recommended by the mbed team for developers using the Windows operating system. This is a free and open source terminal emulator which allows host PC communication with the mbed. In order to use Tera Term with a Windows PC, you will first have to install the Windows serial driver; this can be installed from the Handbook section of the mbed website (Ref. [1]). Note that you need to run the installer for every mbed, as Windows loads the driver based on the mbed serial number. Tera Term can be downloaded for free from the developers LogMeTT at Ref. [2]. At the time of writing the latest version is Tera Term 4.90.

Tera Term is only a Windows compatible program, so Apple Mac and Linux users are advised to use another free program called CoolTerm. This can be downloaded directly from developer Roger Meier's freeware webpage at Ref. [3]. At the time of writing the latest version of CoolTerm is 1.4.6.

E.2 Windows Users—Setting Up and Testing Tera Term

Open the Tera Term application. You will need to perform the following configuration:

- Select File -> New Connection (or just press Alt+N)
- Select the *Serial* radio button and select the *mbed Serial Port* from the drop down menu, as seen in Fig. E.1.
- Click *OK*

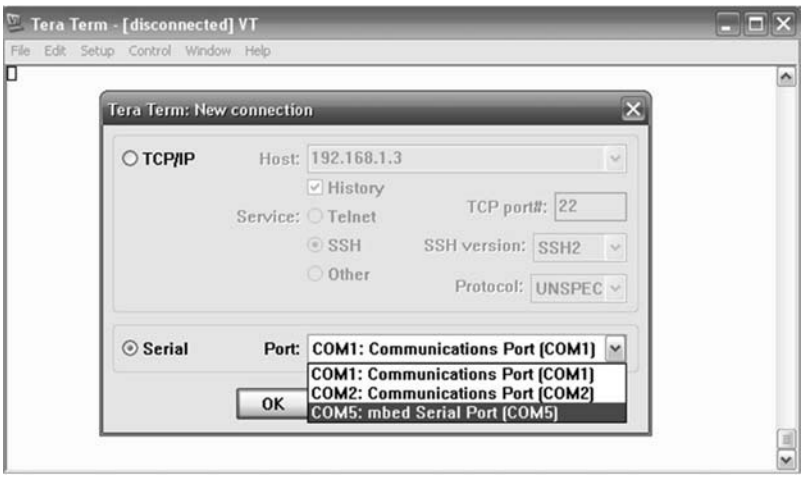

Figure E.1
Setting up the Tera Term connection.

If *mbed Serial Port* is not in the drop-down menu, the Windows serial driver may not be installed, or the mbed may not be connected to your computer's USB port.

Set up new-line format (to print out new-line characters correctly) by following:

- Setup -> Terminal...
- Under "New-line," set Receive to "LF"

To test Tera Term running with an mbed, create a new project of suitable name in the compiler, and enter the code of Program Example E.1. This code simply greets the world, and then reads your keyboard input and displays it to Tera Term. To show that the character value is being seen by the mbed, the program adds one to the ASCII code, meaning that every letter is echoed back to the terminal as the letter following in the alphabet, b instead of a, c instead of b, and so on.

```
/* Program Example E.1: Print to the PC, then pass back characters (slightly
modified!)
                                                                          */
#include "mbed.h"
Serial pc(USBTX, USBRX);    // define transmitter and receiver

int main() {
  pc.printf("Hello World!");
  while(1) {
    pc.putc(pc.getc() + 1);   //adds 1 to the ASCII code, and returns it
  }
}
```

Program Example E.1: Transferring keyboard characters to screen

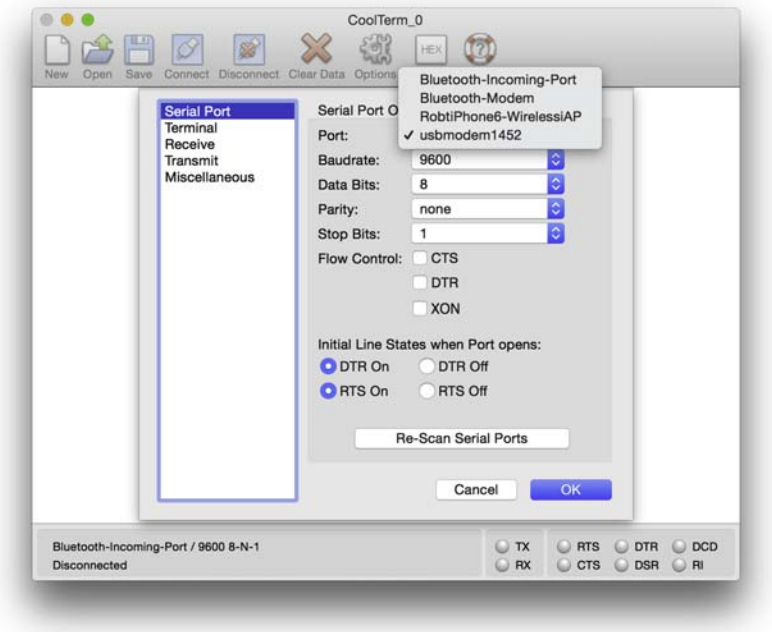

Figure E.2
CoolTerm options.

E.3 Apple Mac and Linux Users—Setting Up and Testing CoolTerm

Ensure the mbed is connected to the host PC's USB port and then open CoolTerm. To initiate the terminal connection to the mbed go to

- Options -> Serial Port
- View the pull-down options in the "Port" filed as shown in Fig. E.2
- Select the "usbmodem1452" option, which refers to the mbed and press "OK"
- Select "Connect" to initiate the connection

Note that the number after the name "usbmodem" is arbitrary and may differ for your mbed.

You can now run Program Example E.1 on your mbed and verify that the "Hello World" and any typed characters are displayed on the CoolTerm screen, with the offset described.

E.4 A More Advanced Host Terminal Example

Regardless of which host terminal application you are using, all Program Examples in this book will work identically, so once you have the emulator set up and working, you no

longer need to worry about the difference between Windows, Apple Mac, and Linux operating systems.

A more advanced program using terminal access is given in Program Example E.2. This is taken from the mbed site, and uses a pulse-width modulated (PWM) signal to increase and decrease the brightness of an LED. The PWM is controlled from the keyboard and displayed on the host terminal screen. To test the program, connect an LED between pin 21 and ground, and of course enable your host terminal application: Tera Term or CoolTerm. If you're at an early stage of reading this book, you're unlikely to be familiar with all the C features of this program, don't worry, they will become clearer as your reading progresses.

```
/* Program Example E.2: Connects to the mbed with a Terminal program and uses the
'u' and 'd' keys to make LED1 brighter or dimmer
                                                                              */
#include "mbed.h"
Serial pc(USBTX, USBRX);            // define transmitter and receiver
PwmOut led(p21);
float brightness=0.0;

int main() {
    pc.printf("Press 'u' to turn LED1 brightness up, 'd' for down\n\r");
    while(1) {
        char c = pc.getc();
        wait(0.001);
        if((c == 'u') && (brightness < 0.1)) {
            brightness += 0.001;
            led = brightness;
        }
        if((c == 'd') && (brightness > 0.0)) {
            brightness -= 0.001;
            led = brightness;
        }
    pc.printf("%c %1.3f \n \r",c,brightness);
    }
}
```

Program Example E.2: Controlling PWM setting from the keyboard

E.5 Shorthand Terminal Communication

In Program Examples E.1 and E.2, we can see that the USB terminal communication object is defined by the following line:

```
Serial pc(USBTX, USBRX);            // define transmitter and receiver
```

This code sets up a serial link called **pc** and explains that it should use the mbed's USB transmit (**USBTX**) and receive (**USBTX**) connection wires. Then in the program we use

the **pc** object to invoke the terminal screen commands such as **pc.printf( )**, **pc.putc( )**, and **pc.getc( )**.

Because almost every detailed mbed program will benefit from exchanging print messages with a host terminal application, the mbed team have also introduced a shorthand programming method for passing formatted print data. Where programs only rely on host terminals for displaying **printf( )** statements, it is possible to omit the **Serial** declaration and remove the **pc** object from **printf( )** statements. For example, Program Example E.3. shows the shorthand approach for simply displaying data to a host terminal.

```
/* Program Example E.3: Shorthand approach to displaying text data on a host
terminal application
                                                                          */
#include "mbed.h"

int main()
{
    printf("Hello World - This shows the shorthand printf approach in action!");
}
```

Program Example E.3.: Shorthand approach to displaying data on host terminal

Note that the shorthand approach works well for **printf( )** commands, but it is not configured for every **Serial** library feature. In this book, you will see us use both the shorthand and longhand versions for communicating with host terminal applications. Where more advanced **Serial** library features are utilized, the longhand version will always be used.

References

[1] Windows Serial Configuration. https://developer.mbed.org/handbook/Windows-serial-configuration.
[2] Tera Term 4.90. http://logmett.com/tera-term-the-latest-version.
[3] CoolTerm download page. http://freeware.the-meiers.org.