SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

SUPPLEMENTAL INFORMATION

To: CENG 447/547 Students

From: Dr. Randy Hoover

Subject: HC-08 Bluetooth Module and Python Quick-Start Guide

Date: April 16, 2019

Note: This guide is written for Windows 10. The provided code can work under Linux systems as well, but more work is required to get bleak running, particularly on older distributions.

1 Python

To get started, you will need **Python 3.6** specifically. Python 3.7 has some incompatibility with the libraries needed for Bluetooth, at least on Windows 10. It's suggested that you use the 64-bit, executable installer. Importantly, when you run the installer, make sure you add python to the PATH environment variable (it's one of the installer options). If you're on Linux, all of the aforementioned instructions may not be an option for you, and you may be also fine with whatever version of Python you try.

Once installed, make sure it works by opening a command prompt and typing "python". You should get an interactive prompt, which you can quit by typing `exit()`. Make sure you get this working before you move on.

Next, you'll need some package(s) from the python package repo, which you can obtain using PIP:

- `bleak` a BLE (Bluetooth Low Energy) library needed to connect to the Bluetooth module.
- `readchar` (*optional*) used in the controller script to immediately forward keystrokes from the keyboard to your Bluetooth module.

Install them both using a command prompt *in Administrator mode* (right-click cmd.exe, "Run as Administrator"):

```
python -m pip install bleak readchar
```

If you installed python in the default location (i.e. `C:/Users/.../AppData/...`) then you may need to run `pip` with the `--user` flag to install the package for the user-local version of Python you installed:

```
python -m pip install --user bleak readchar
```

It is critical at this point that bleak works on your system. Some Linux users may have a hard time getting this working due to an older version of Bluetooth firmware or other backend Bluetooth tools, and versions of Windows older than Windows 10 don't even support the necessary backend. You can test out the discovery script to sanity check that bleak doesn't throw any errors:

```
python findbt.py
```

2 Microcontroller

The next step is to look to the microcontroller base firmware: `microctrl_base.c`: you'll want to change the macro definition `MY_BT_NAME` at the top to something short and distinct enough to pick it out of a list of Bluetooth devices, potentially in a crowded room. You'll probably want to stick with alphanumeric names. The firmware gives an "AT" command to the Bluetooth module on startup which tells it to advertise itself under this name.

Compile and upload the firmware to your Uno **with the Bluetooth module unplugged** and then disconnect the Uno from your computer. With the USB cable disconnected, plug in the Bluetooth module, and power the Uno using the batteries (or some other external power that isn't the USB cable).

3 Connecting

Run the discovery script with `python findbt.py` and wait for a list of devices to populate, along with their MAC addresses. Search the list for your device, and save that MAC address: This is yours: it will not change! You can hard-code it into any script you use, including the provided one: `controller.py`. You don't even need to know the name of the device anymore; the mac address is what you'll actually use to connect.

Update the `controller.py` script with your MAC address (`mac_addr`) and run the script. After 10-15 seconds, you should see the light on the Bluetooth module glow constant, and a message on your command prompt letting you know that it's connected. Every keypress will send a character to the Bluetooth module, and get piped into the UART on your machine.

The supplied firmware file simply sets the onboard LED high when it gets an 'h' and pulls it low when it sees an 'l'. This should give you an idea of how to extend this functionality into some serious remote control. Be creative with the python script if you want to implement a "press and hold" mechanism for remote control. It's up to you to modify the firmware to actually cooperate with the other devices you'll need to manage the robot car challenge. This may include dramatically rewriting the UART interface to the Bluetooth module.

Good luck!