ULab 7

Explore more BLE on the Adafruit Feather Bluefruit Sense

Part 1 - BLE UART with Bluefruit App

UART stands for Universal Asynchronous Read Transmit. We used a UART when we used the serial connection in CP Lab 2. Now we are going to use a BLE connection as a UART. Once connected, it should work similar to the serial UART we used previously.

We will need CP code on the Feather sense (peripheral code) and will connect to the Bluefruit app.

Save the code below to the Feather sense as code.py. Make sure to give the ble object a name so you can connect to your specific device more easily with the application.

****

```
# feather ble uart code
# put on feather sense board

from adafruit_ble import BLERadio
from adafruit_ble.advertising.standard import ProvideServicesAdvertisement
from adafruit_ble.services.nordic import UARTService

ble = BLERadio()
# this will be the advertisement name when advertising begins
ble.name = "LAB&_FEATHER"

uart = UARTService()
advertisement = ProvideServicesAdvertisement(uart)

print("Advertising as: "+ble.name)
while True:
    ble.start_advertising(advertisement)
    print("Waiting to connect")
    while not ble.connected:
        pass
    print("Connected")
    while ble.connected:
        data_string = uart.readline().decode("utf-8")
        if data_string:
            print(ble.name+ " received: " +data_string)
            uart.write(ble.name+ " received: " +data_string)
```

With the code above running on the Feather Sense, connect with the Bluefruit app. After the connection has been established, select UART from the list of modules. From the UART window, send a string to the Feather Sense. In the Mu editor you should see the Feather acknowledge it received something and the Feather should respond to the connected Bluefruit app as well.

With this simple UART connection, we can send any “command” desired to the Feather Sense. Modify your code running on the Feather Sense by adding commands to turn the neopixel on and off with the commands: “on” and “off”. Choose a default “on” color for your neopixel.

Add more “commands” to change the neopixel color with the commands “red”, “blue”, “green” and “default”.

Show your code working on the Feather to your instructor.

Add header information and inline comments to your code and save it as “lastname_lab7_part1.py”

Possible solution code lab7_part1.py

Part 2 - BLE UART with the computer connection

Server code -- run in the Mu editor in Python3 mode. Before we can run this code on the pc, we need to install BLE libraries made by Adafruit which can be used on the desktop computer too. Follow your instructor's instructions to properly install the necessary modules on the desktop computer.

Need to install:

```
adafruit-blinka-bleio

adafruit-circuitpython-ble
```

To add these to the Mu, have students click on the gear icon at the bottom right of the Mu editor. This will bring up a new window. Click on the tab - Third Party Packages. Here students will see a list of available packages that can be imported into the Mu Python. Scroll to the bottom of the list and add the two packages listed above and click the OK button. Mu should install the packages. A check on Third Party Packages should now list the recently installed packages.

When you have successfully installed the modules, you are ready to proceed. First copy the following code onto the Feather Sense and save it code.py

****

```
# feather ble uart code
# put on feather sense board

from adafruit_ble import BLERadio
from adafruit_ble.advertising.standard import ProvideServicesAdvertisement
from adafruit_ble.services.nordic import UARTService

ble = BLERadio()
# this will be the advertisement name when advertising begins
ble.name = "LAB&_FEATHER"

uart = UARTService()
```

```
advertisement = ProvideServicesAdvertisement(uart)

print("Advertising as: "+ble.name)
while True:
    ble.start_advertising(advertisement)
    print("Waiting to connect")
    while not ble.connected:
        pass
    print("Connected")
    while ble.connected:
        data_string = uart.readline().decode("utf-8")
        if data_string:
            print(ble.name+ " received: " +data_string)
            uart.write(ble.name+ " received: " +data_string)
```

This is the same code we started part 1 of the lab with on the Feather Sense. Next, switch Mu to Python mode and copy the following code to an editor window. Make sure to change the name in line 40 (or so) to the name your Feather Sense is advertising. Save this file as “desktop_ble_uart_echo.py”

```
**********************************************************************************************

# SPDX-FileCopyrightText: 2020 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

"""
Used with ble_uart_echo_test.py. Transmits word2send to the UARTService and
receives a response from Feather
"""

import time

from adafruit_ble import BLERadio
from adafruit_ble.advertising.standard import ProvideServicesAdvertisement
from adafruit_ble.services.nordic import UARTService

ble = BLERadio()

word2send = "hello"

while True:
    while ble.connected and any(
        UARTService in connection for connection in ble.connections
    ):
        for connection in ble.connections:
            if UARTService not in connection:
                continue
            print("sending: "+word2send)
            uart = connection[UARTService]
            uart.write(word2send.encode("utf-8"))
            # Returns b'' if nothing was read.
            one_byte = uart.readline()
            if one_byte:
                # convert bytearray to string
                data_string = ''.join([chr(b) for b in one_byte])
                #print(data_string,end='')
                print(data_string,end='')
            print()
```

```
        time.sleep(1)

    print("disconnected, scanning")
    for advertisement in ble.start_scan(ProvideServicesAdvertisement,
timeout=1):
        #print(advertisement.address, advertisement.complete_name)
        if UARTService not in advertisement.services:
            continue
        if advertisement.complete_name == "LAB7_FEATHER":
            ble.connect(advertisement)
            print("connected to:",advertisement.complete_name)
        break
    ble.stop_scan()
```

Run the desktop_ble_uart_echo program. What happens?

The desktop program looks for ble advertisements with an exact name. If the name matches and the advertisement.services includes a UART, the computer connects to the Feather. As soon as a connection is established, the computer begins to send “word2send” to the Feather. The Feather receives the transmission and echoes a response back to the computer.

Document your code. Change the “word” sent from the desktop computer. Change the response sent from the server Feather sense.

Show your modified code working on both the Feather and desktop computer to your instructor.

Add header information and inline comments to all your Feather code and save it as “lastname_lab7_part2.py”

Add header information and inline comments to your desktop_ble_uart_echo.py program.

Part 3 - BLE UART led control from the PC

With a BLE UART between the desktop computer and the Feather Sense you can again start to make up your own commands to have the computer tell the Feather what to do.

Load “lastname_lab7_part1.py” into Mu and save it to your Feather Sense again as code.py. This is the code that allows UART control of the neopixel.

Now modify the “desktop_ble_uart_echo.py” file to send the proper commands and control the neopixel.

Show your code working on both the Feather and computer to your instructor.

Add header information and inline comments to your code and save it as “desktop_ble_uart_neopixel_control.py”

This should take very little time to modify the Python3 desktop ble file to accept an input from the user and then send it over ble UART to the Feather.

Possible solution code: desktop_ble_uart_neopixel_control.py

Part 4 - BLE UART stream sensor data to PC

IOT is also about collecting data, not just control. Starting with copies of your code from Part 3, you are going to make changes to both. The goal is to allow the Feather Sense to send temperature data to the desktop computer over the BLE UART.

Create a new "command" that tells the Feather Sense to transmit the current temperature to the computer every fifteen seconds. The desktop computer should receive the temperature value and display it with a simple print statement. Blink the neopixel red everytime a temperature is sent from the Feather Sense as an indication that the Feather is still active. Also add a command 'delay ##' that will modify the delay time on the Feather Sense.

Show your code working on both the Feather and computer to your instructor. ☐

Modify header information and inline comments in your code and save.
"desktop_ble_uart_temp.py" for the desktop code.

"lastname_lab7_part4.py" for the Feather Sense code.

Possible solution code: desktop_ble_uart_temp.py
lab7_part4.py

Part 5 - BLE UART stream data to pc and save to file

Modify your desktop code from Part 4 so it now saves the data sent from the Feather Sense to a data file on the PC. When saving the temperature data to the file, make sure to include a timestamp for each data point. Refer back to Lab 5 Part 3. Add a try and except to your desktop code to catch Ctrl-C to exit data logging.

Show your code working on both the Feather and computer to your instructor. ☐

Modify header information and inline comments in your code and save.

"desktop_ble_uart_temp_logging.py"

Possible solution code: desktop_ble_uart_temp_logging.py