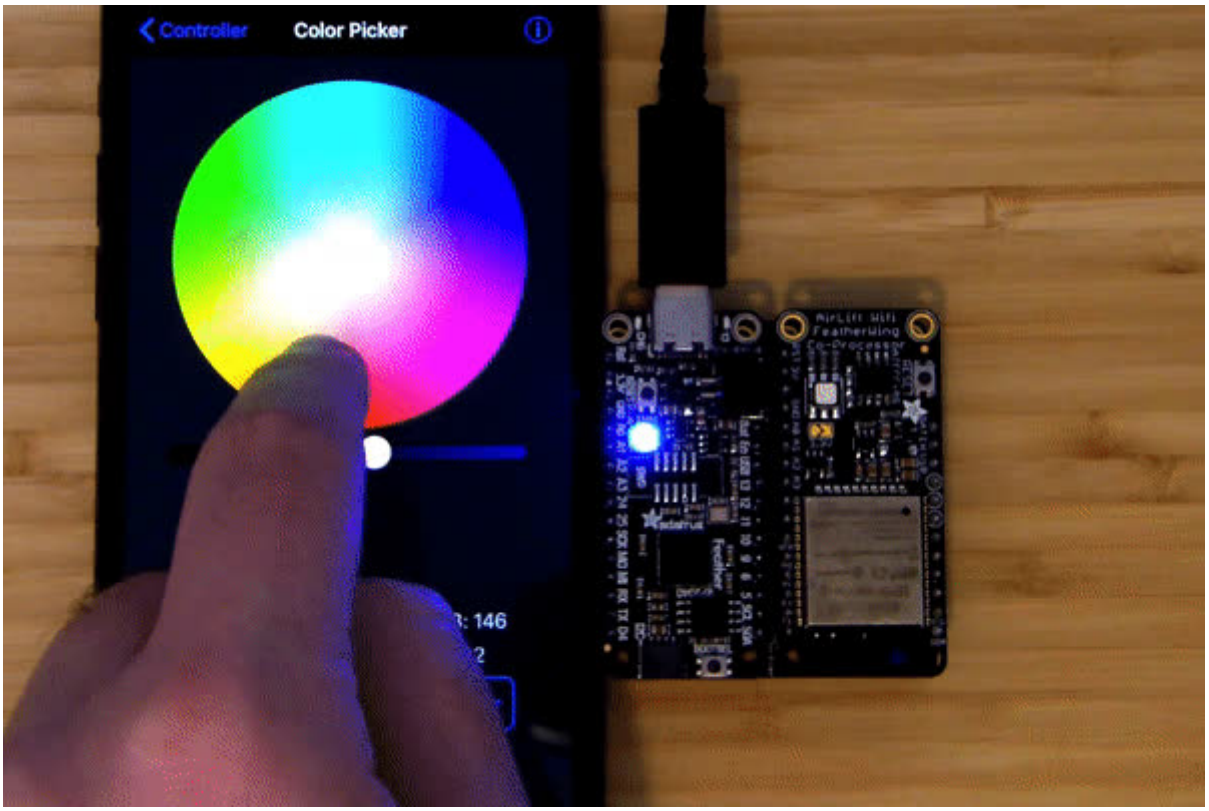
Quickstart - Raspberry Pi RP2040 with BLE and CircuitPython

Created by Brent Rubell

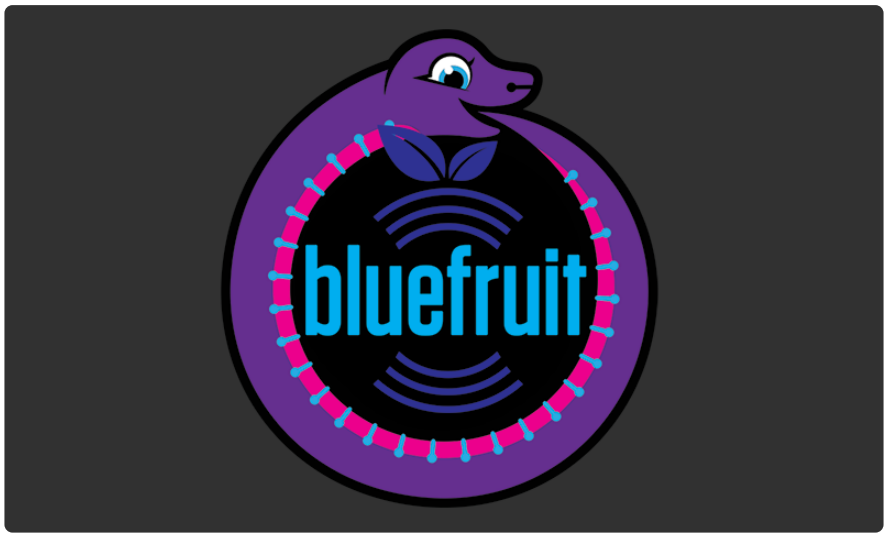


https://learn.adafruit.com/quickstart-raspberry-pi-rp2040-with-ble-and-circuitpython

Last updated on 2024-06-03 03:21:52 PM EDT

Table of Contents

Overview



Add Bluetooth Low Energy (BLE) connectivity to your Raspberry Pi RP2040 project by adding an Adafruit AirLift ESP32 co-processor. The AirLift's ESP32 module can be placed into either a WiFi mode (we've published a guide about using the AirLift's WiFi mode with Pico here (https://adafru.it/RIC)) or BLE mode, giving you the option of adding two different methods of wireless connectivity to your RP2040 project.

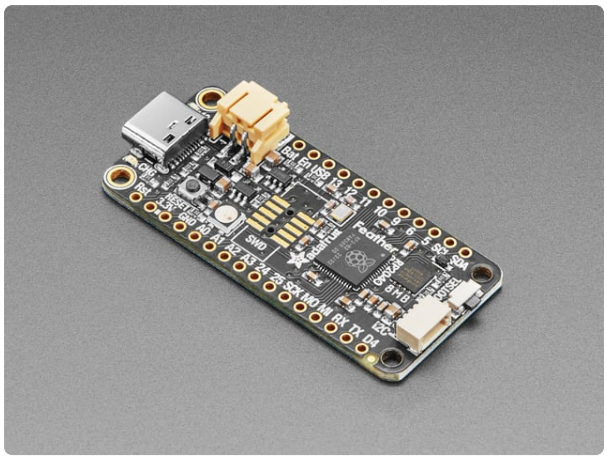
This guide will cover getting started using BLE to connect an RP2040 board with the Bluefruit LE Connect App along with examples for a UART echo server and wirelessly changing the color of a NeoPixel RGB LED.

A few notes before we begin:

- **You can not use the WiFi and BLE mode on the Adafruit AirLift simultaneously**, so select your transport wisely (https://adafru.it/Cio).
- **CircuitPython's AirLift support only provides BLE peripheral support**, BLE central support is under development. This means you can not connect the RP2040 to BLE devices such as heart rate monitors or thermometers, but you can make the RP2040 act as a BLE peripheral.

This tutorial uses the mobile Adafruit Bluefruit Connect app on iOS and Android. To run the examples, your device must be compatible with one of these two apps - see the Apple app store or Google Play for compatibility.

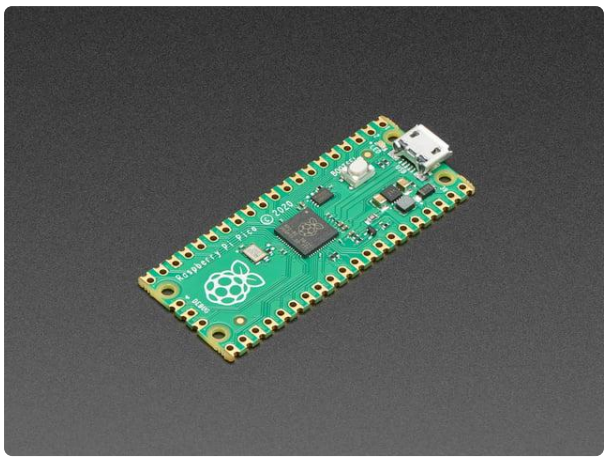
Parts



Adafruit Feather RP2040

A new chip means a new Feather, and the Raspberry Pi RP2040 is no exception. When we saw this chip we thought "this chip is going to be awesome when we give it the Feather...

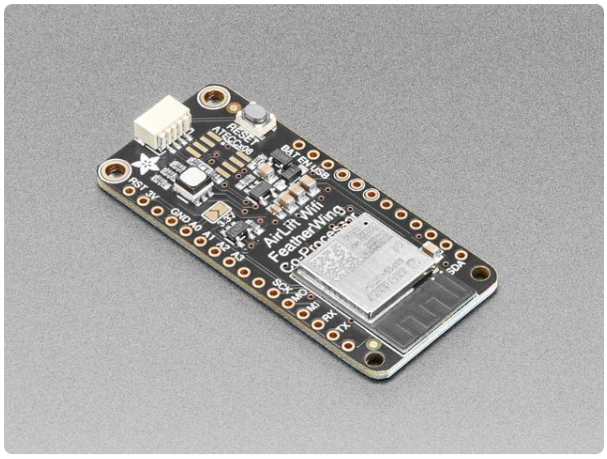
https://www.adafruit.com/product/4884



Raspberry Pi Pico RP2040

The Raspberry Pi foundation changed single-board computing when they released the Raspberry Pi computer, now they're ready to...

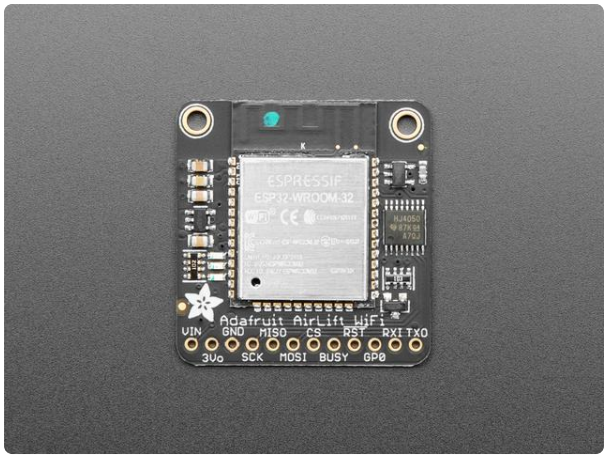
https://www.adafruit.com/product/4864



Adafruit AirLift FeatherWing – ESP32 WiFi Co-Processor

Give your Feather project a lift with the Adafruit AirLift FeatherWing - a FeatherWing that lets you use the powerful ESP32 as a WiFi co-processor. You probably have your...

https://www.adafruit.com/product/4264



Adafruit AirLift – ESP32 WiFi Co-Processor Breakout Board

Give your plain ol' microcontroller project a lift with the Adafruit AirLift - a breakout board that lets you use the powerful ESP32 as a WiFi co-processor. You probably...

https://www.adafruit.com/product/4201

1 x FeatherWing Doubler — https://www.adafruit.com/product/2890

FeatherWing Doubler - Prototyping Add-on For All Feather Boards

---

1 x USB-C Cable — https://www.adafruit.com/product/4199

USB-C Cable for RP2040 Feather

---

1 x USB Cable — https://www.adafruit.com/product/592

USB cable - USB A to Micro-B - 3 foot long

---

1 x Breadboard — https://www.adafruit.com/product/239

Full sized breadboard

---

1 x Breadboarding Wires — https://www.adafruit.com/product/153

Breadboarding wire bundle

Bluetooth Low Energy Basics

The nRF52840 uses Bluetooth Low Energy, or BLE. BLE is a wireless communication protocol used by many devices, including mobile devices. You'll be able to communicate with your nRF52840 board using your mobile phone!

There's a few terms and concepts commonly used in BLE with which you may want to familiarise yourself. This will help you understand what your code is doing when you're using CircuitPython and BLE.

The major concepts can be broken down into two categories: connection set up and communication. The first deals with setting up connections between devices, such as between your mobile phone and the nRF52840 board. The second deals with communication between the devices once they are connected.

Bluetooth Terms

- **Central** - The host computer. This is often a mobile device such as a phone or tablet, or it could be a desktop or laptop.
- **Peripheral** - The connected device. Examples of peripherals are: heart rate monitor, smart watch, or fitness tracker. The CircuitPython code we have so far is designed to make the Adafruit nRF52840 devices work as peripherals.
- **Advertising** - Information sent by the peripheral during connection set up. When a device advertises, it is transmitting the name of the device and describing its capabilities. The central looks for an advertising peripheral to connect to, and uses that information to determine what the peripheral is capable of.
- **Service** - A function the peripheral provides. The peripheral advertises its services. A really common service that we use is the UART service, which acts like a hardware UART and is a way of bidirectionally sending information to and from devices.
- **Packet** - Data transmitted by a device. BLE devices and host computers transmit and receive data in small bursts called packets.

Making a Bluetooth Connection

To use these terms in the context of connecting to your Adafruit nRF52840:

- You run CircuitPython code that makes your board act as a peripheral by advertising its name and the services it's capable of.
- You start up Adafruit's **Bluefruit LE Connect app** on an Android or iOS device in central mode, that device becomes the central, and begins listening for the peripheral.
- You set up the connection between the nRF52840 peripheral and the Bluefruit LE Connect app, and the app discovers the details about the services that the peripheral is capable of.
- Once this connection is made, you can use CircuitPython code to read packets sent from the Bluefruit LE Connect app to your nRF52840 board. For example, you can receive data describing screen button presses or RGB color values.

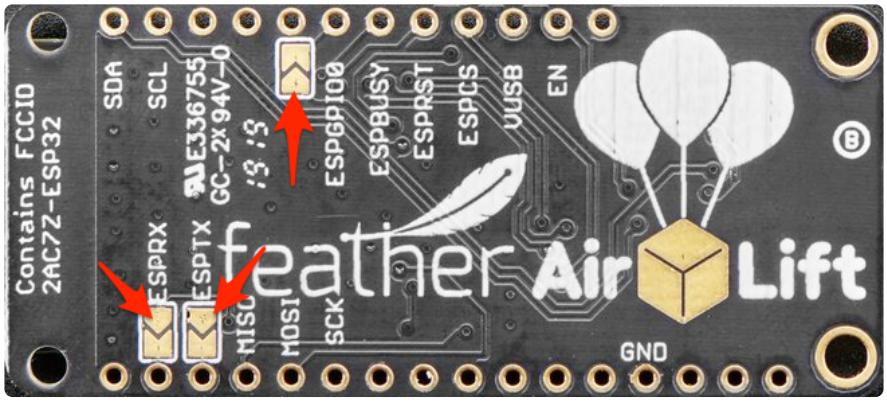
Now that you have a general idea of basic BLE terms and concepts, it's time to install the Bluefruit LE Connect application, and run some CircuitPython demos!

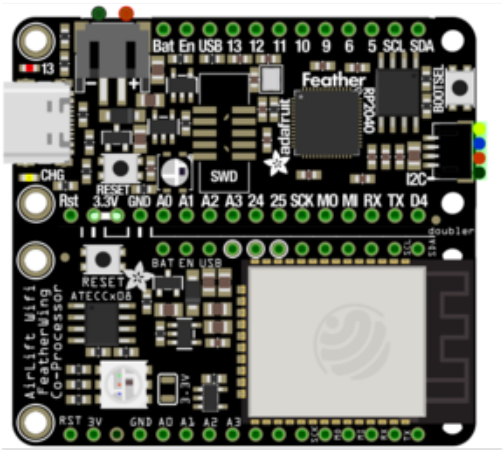
Assembly

Adafruit sells Airlift hardware in multiple (http://adafru.it/4264) form (http://adafru.it/4201) factors (http://adafru.it/4363) in order to be compatible with many microcontroller boards. While each microcontroller board and airlift hardware combination has its own wiring scheme, we suggest either using the wiring below or referencing the AirLift product guide.

Adafruit Feather RP2040 Wiring

To use the AirLift FeatherWing with BLE, you will need to **solder the three jumpers closed on the bottom of the AirLift FeatherWing**:



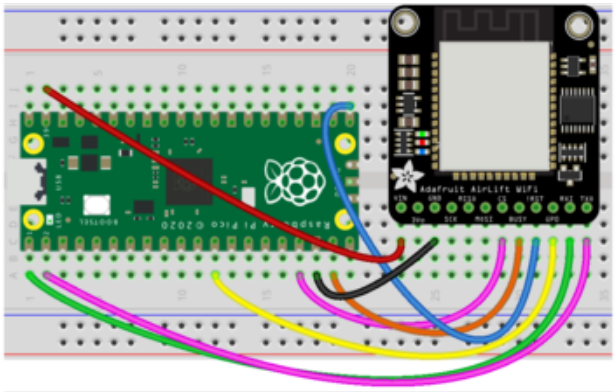


After soldering the jumpers closed, we suggest using the FeatherWing Doubler (http://adafru.it/2890) to connect a Feather RP2040 and an AirLift FeatherWing side-by-side.

Raspberry Pi Pico RP2040 Wiring

If you're using a Raspberry Pi Pico RP2040, make the following connections between the Pico RP2040 and an Adafruit AirLift breakout.

You MUST use the Pico's VSYS pin for powering the AirLift Breakout.



Pico **VSYS** to **AirLift Vin**
Pico GND to **AirLift GND**
Pico GP13 (SPI1 CSn) to **AirLift CS**
Pico GP14 to **AirLift BUSY**
Pico GP16 to **AirLift !RST**
Pico GP9 to **AirLift GPIO0**
Pico GP0 to **AirLift RXI**
Pico GP1 to **AirLift TX0**
For more information about the UART peripheral and pinouts of the Pico, check out this guide (https://adafru.it/Qsd).

Code Usage

Before continuing, ensure your AirLift's firmware is version 1.7.1 or higher for BLE to work.

Install CircuitPython Libraries

Make sure you are running the latest version of Adafruit CircuitPython (https://adafru.it/Amd) for your board; you'll need 6.0.0 or later.

Next, you'll need to install the necessary libraries to use the hardware and BLE. Carefully follow the steps to find and install these libraries from Adafruit's CircuitPython library bundle (https://adafru.it/uap). Our CircuitPython starter guide has a great page on how to use the library bundle (https://adafru.it/ABU).

Install these libraries from the bundle:

- **adafruit_airlift**
- **adafruit_ble**
- **adafruit_esp32spi**
- **adafruit_bus_device**
- **adafruit_bluefruit_connect**
- **adafruit_requests.mpy**
- **neopixel.mpy**

Before continuing make sure your board's **lib** folder or root filesystem has the files and folders listed above copied over.

Install Adafruit Bluefruit LE Connect App

The Adafruit Bluefruit LE Connect iOS and Android apps allow you to connect to BLE peripherals. Follow the instructions in the Bluefruit LE Connect Guide (https://adafru.it/Eg5) to download and install the app on your phone or tablet.

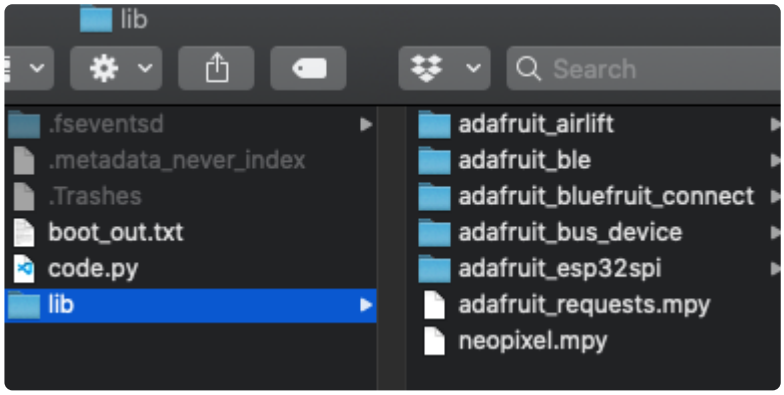
Code Setup

Copy the code below to the **code.py** file on your **CIRCUITPY** drive. If you're using an AirLift Breakout, you will need to modify the code by commenting out the default pin settings and uncommenting the code for the AirLift breakout.

```
# SPDX-FileCopyrightText: 2021 Brent Rubell for Adafruit Industries
#
# SPDX-License-Identifier: MIT

import board
from adafruit_ble import BLERadio
from adafruit_ble.advertising.standard import ProvideServicesAdvertisement
from adafruit_ble.services.nordic import UARTService
from adafruit_airlift.esp32 import ESP32

# If you are using an AirLift FeatherWing or AirLift Bitsy Add-On,
# use the pin settings below:
esp32 = ESP32(
    reset=board.D12,
    gpio0=board.D10,
    busy=board.D11,
    chip_select=board.D13,
    tx=board.TX,
    rx=board.RX,
)

# If you are using an AirLift Breakout, comment out the DEFAULT lines
# above and uncomment the lines below:
# esp32 = ESP32(
#     reset=board.GP16,
#     gpio0=board.GP9,
#     busy=board.GP14,
#     chip_select=board.GP13,
#     tx=board.GP0,
#     rx=board.GP1,
# )


adapter = esp32.start_bluetooth()

ble = BLERadio(adapter)
uart = UARTService()
advertisement = ProvideServicesAdvertisement(uart)

while True:
    ble.start_advertising(advertisement)
    print("waiting to connect")
```

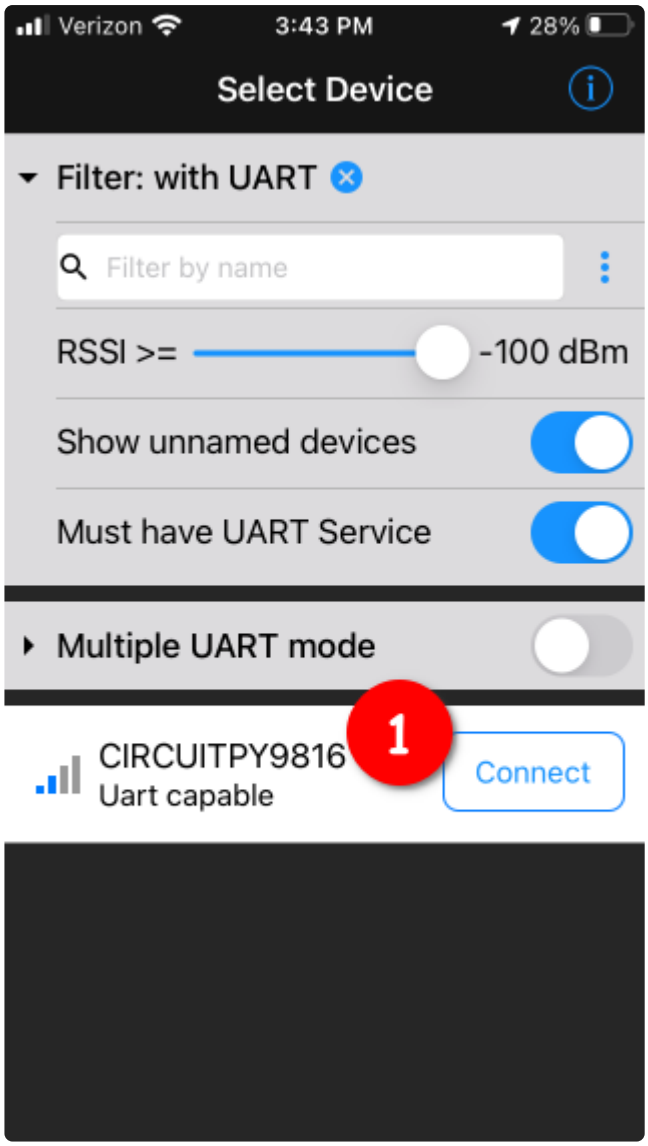
```
while not ble.connected:
    pass
print("connected: trying to read input")
while ble.connected:
    # Returns b'' if nothing was read.
    one_byte = uart.read(1)
    if one_byte:
        print(one_byte)
        uart.write(one_byte)
```

Your CIRCUITPY file-system should look like the following:

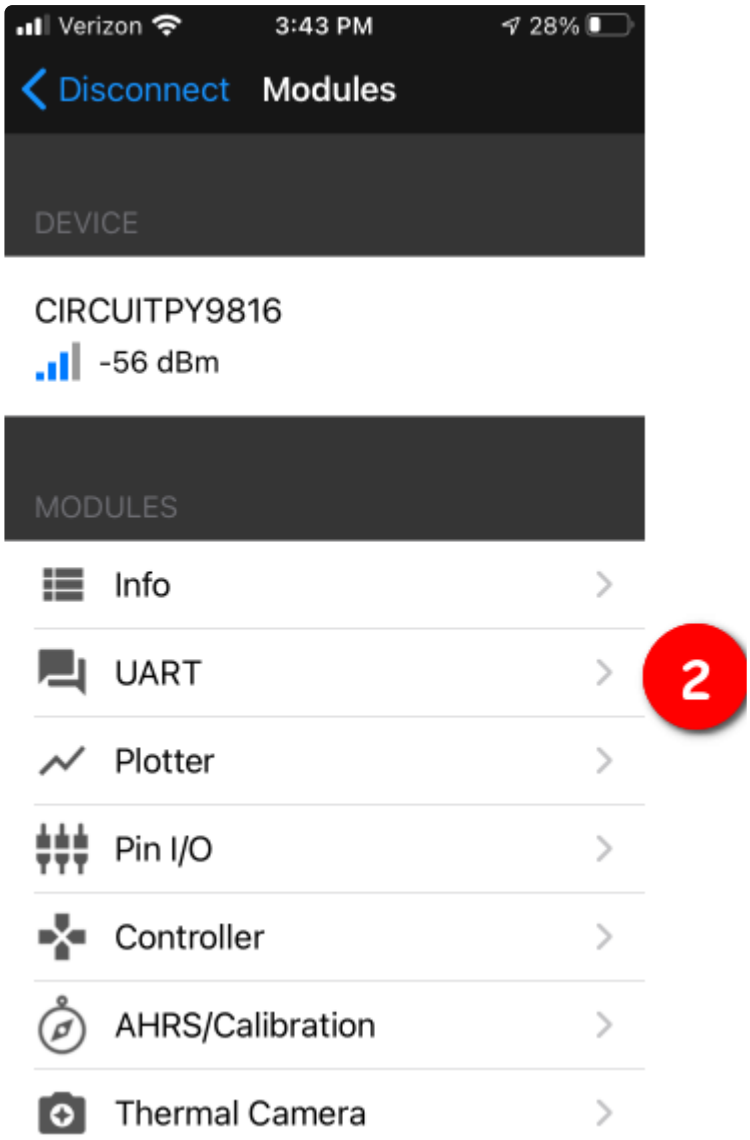


Code Usage

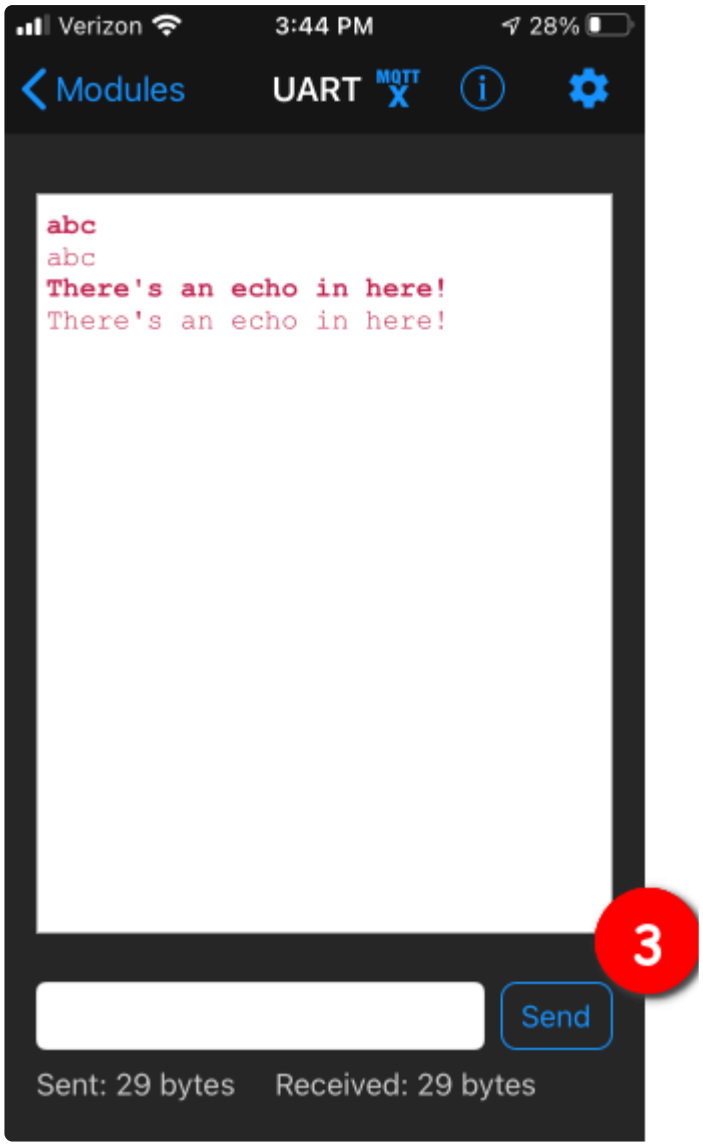
Start the Bluefruit LE Connect App on your phone or tablet. You should see a CIRCUITPY device available to connect to. Tap the Connect button.



You'll then see a list of Bluefruit Connect functions ("modules"). Choose the UART module.



On the UART module page, you can type a string and press Send. You'll see that string entered, and then see it echoed back (echoing is in gray).



That's it!

NeoPixel Color

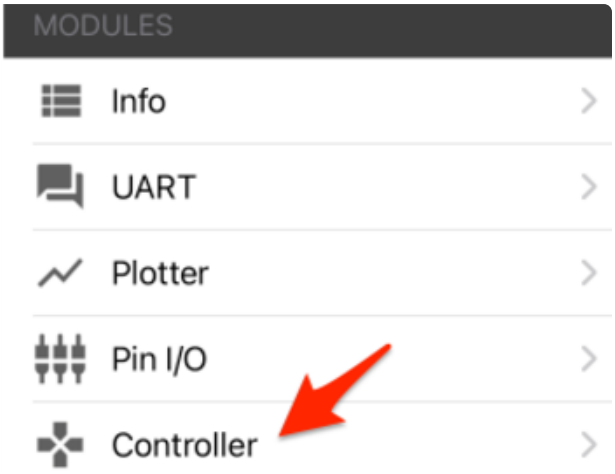
> This page is designed for use with the Adafruit Feather RP2040. The Raspberry Pi Pico RP2040 does not include an onboard NeoPixel LED.

The Bluefruit LE Connect app has a Color Picker that allows you to easily set the color of NeoPixels connected to your Adafruit Feather RP2040. In this example, we'll change the color of the onboard NeoPIxel using CircuitPython and the Bluefruit LE Connect app.

Save the following as **code.py** on your **CIRCUITPY** drive and then connect to the board using the Bluefruit LE Connect app.

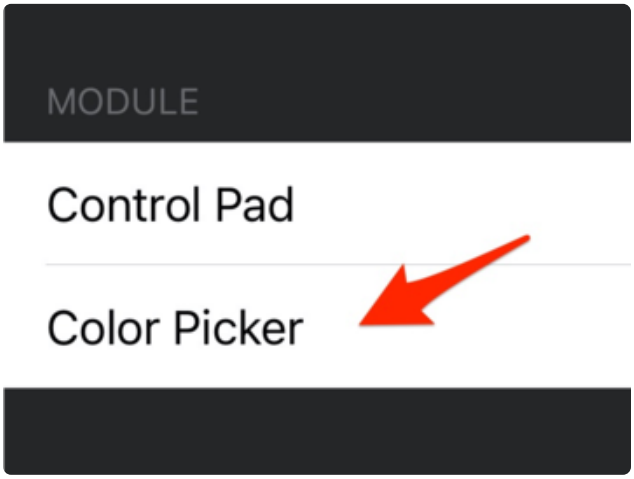
```
# SPDX-FileCopyrightText: 2021 Brent Rubell for Adafruit Industries
#
# SPDX-License-Identifier: MIT

import board
import neopixel

from adafruit_bluefruit_connect.packet import Packet
from adafruit_bluefruit_connect.color_packet import ColorPacket


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

from adafruit_airlift.esp32 import ESP32


# If you are using an AirLift FeatherWing or AirLift Bitsy Add-On,
# use the pin settings below.
# If you are using an AirLift Breakout, check that these
# choices match the wiring to your microcontroller board,
# or change them as appropriate.
esp32 = ESP32(
    reset=board.D12,
    gpio0=board.D10,
    busy=board.D11,
    chip_select=board.D13,
    tx=board.TX,
    rx=board.RX,
)

adapter = esp32.start_bluetooth()

ble = BLERadio(adapter)
uart_service = UARTService()
advertisement = ProvideServicesAdvertisement(uart_service)

pixels = neopixel.NeoPixel(board.NEOPIXEL, 10, brightness=0.1)

while True:
    # Advertise when not connected.
    ble.start_advertising(advertisement)
    while not ble.connected:
        pass

    while ble.connected:
        if uart_service.in_waiting:
            packet = Packet.from_stream(uart_service)
            if isinstance(packet, ColorPacket):
                print(packet.color)
                pixels.fill(packet.color)
```

Then, connect to your board using the Bluefruit LE Connect application. From the controller page, navigate to the Color Picker page.



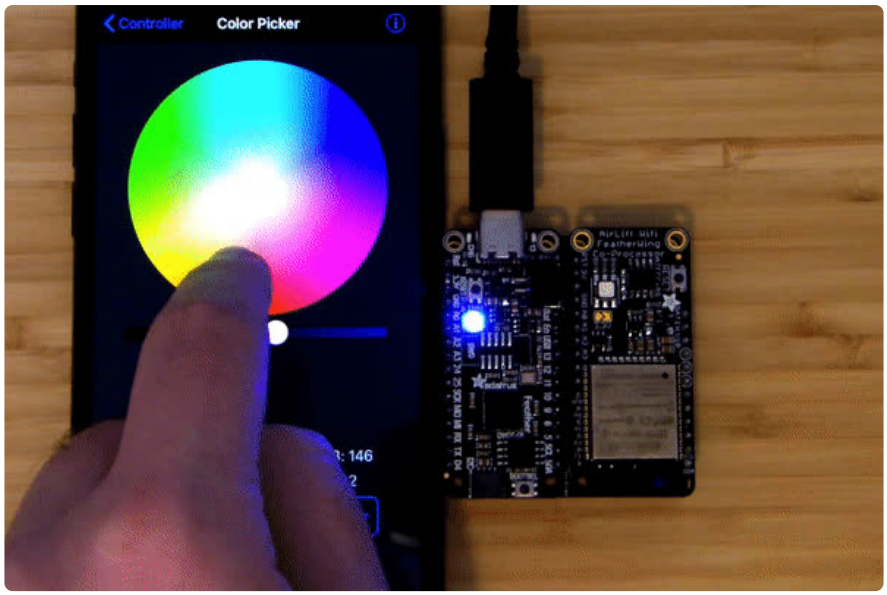
Connect to your board using the Bluefruit LE Connect application.

From the Device page, navigate to the Controller page.



On the controller page, tap the Color Picker.

Selecting a color and tapping the "Send Selected Color" button should send the red, green, and blue values from the Bluefruit Connect App to the RP2040.



Adapting Existing CircuitPython BLE Code for AirLift BLE

CircuitPython's AirLift support only provides BLE peripheral support. This means you can not use CircuitPython BLE code to BLE devices such as thermometers to the RP2040.

We have written lots of examples (https://adafru.it/RIE) and projects for using CircuitPython's BLE peripheral mode. However, these examples are designed to work with CircuitPython hardware that includes a built-in BLE module such as the Nordic NRF52840.

Adapting code that uses the built-in BLE module to use the AirLift ESP32's BLE mode is simple. Add the following line to your code to import the `adafruit_airlift` module:

```
from adafruit_airlift.esp32 import ESP32
```

Code that utilizes native `adafruit_ble` will have the following line which initializes an onboard BLE module:

```
ble = BLERadio()
```

Replace this line with the following lines below to initialize the AirLift and set up the `BLERadio` module for use with the AirLift. Make sure to match the `esp32`'s pinout with your wiring.

```
esp32 = ESP32(
    reset=board.D12,
    gpio0=board.D10,
    busy=board.D11,
    chip_select=board.D13,
    tx=board.TX,
    rx=board.RX,
)

adapter = esp32.start_bluetooth()
ble = BLERadio(adapter)
```

That's it - you do not need to modify the rest of the code, it should work as-is! Visit the Adafruit Learning System (https://adafru.it/dlu) for more guides on using Bluetooth Low Energy with CircuitPython.

BLE Technical Details

BLE Technical Details (https://adafru.it/DN8)