# Table of Contents

## Overview
- Parts 3
- Materials & Tools 4

## Build the Rover
- Feather to Crickit 6
- Standoffs 7
- Roller Bearing 9
- Motor Wiring 11
- NeoPixel Prep 12
- Motor Mounts 13
- Wheels 15
- Connect Motor Wires to Crickit 16
- Underbody Lighting Effects 17
- Battery Power 18

## Code the Rover with CircuitPython
- Feather and Crickit Prep 23
- Libraries 23
- CircuitPython Code 25
- How the Code Works 27
- Motor Movement 27
- Control the Rover with the Bluefruit LE Connect App 28
- Connect to Feather 28
- Troubleshooting 31
- Device Menu 31
- Controller Menu 32
- Change Color 33
Overview

It's now easier than ever to build your own rover bot with Bluetooth Low Energy (BLE) control! Use the Adafruit Bluefruit app on an Android or iOS device to drive it and change the underlighting colors!

All thanks to CircuitPython, the coding is simple and straightforward. You can even add accessories or assign the buttons to different functions. There are lots of opportunities for modifications on the project, too, including accessing your phone's accelerometer for tilt control!

You'll use the powerful Feather nRF52840 Express, the Crickit robotics platform, a beautiful purple aluminum chassis, motors, wheels, a 24 NeoPixel ring and a few other parts to build this one-of-a-kind rover.

Parts

1 x Feather nRF52840 Express
Bluetooth Low Energy and native USB support  https://www.adafruit.com/product/4062

1 x Adafruit CRICKIT FeatherWing
for any Feather  https://www.adafruit.com/product/3343

1 x Purple Aluminum Chassis
for TT Motors - 2WD  https://www.adafruit.com/product/3796
1 x NeoPixel Ring - 24 x 5050 RGB LED with Integrated Drivers

https://www.adafruit.com/product/1586

2 x TT Motor All-Metal Gearbox 1:90 Gear Ratio

https://www.adafruit.com/product/3802

2 x DC Gearbox Motor "TT Motor" 200RPM - 3 to 6VDC

https://www.adafruit.com/product/3777

2 x Skinny Wheel for TT DC Gearbox Motors

https://www.adafruit.com/product/3757

1 x Ball Caster 3/4" Metal Ball

https://www.adafruit.com/product/1200

1 x 2x2 AA Battery Holder with Premium Jumper Header Wires

https://www.adafruit.com/product/3905

1 x Male DC Power adapter 2.1mm plug to screw terminal block

https://www.adafruit.com/product/369

1 x Circuit Playground Bolt-On Kit 6 M3 standoffs and screws

https://www.adafruit.com/product/3816

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

https://www.adafruit.com/product/592

Alternative motors:

Materials & Tools

In addition to the parts above, you'll also need:

- 4 Rechargeable NiMH AA batteries (such as Eneloop)
- Eight small zip ties
- Four M3 x 25mm long screws with nuts
- Stranded hookup wire
- Soldering iron and solder
- Small screwdriver
- Wire cutter and wire stripper

©Adafruit Industries
Build the Rover

Here's our collection of parts, let's build it!
Feather to Crickit

First, solder the male header pins onto the Feather as shown here (https://adafruit.it/DRi), then connect it to the Crickit.
Standoffs
We'll use four brass standoffs and seven of the provided M3 screws to connect the Crickit to the chassis.

First, fasten one brass standoff to the indicated mounting hole next to the Crickit's reset button. This one won't go through a chassis hole, it will just provide support.

Fasten the remaining three standoffs to the chassis as shown, screwing them in from the bottom. You can leave them a little bit loose for now to get the proper spacing when we connect the Crickit on top, then tighten them.
Roller Bearing
The roller bearing will allow the rover to turn without the need for a complicated differential or steering mechanism.

You will need to remove the bearing from the housing to get to the screw heads.

Screw it into place as shown, using the nuts to secure from the underside.
Motor Wiring

The standard, yellow TT motors come pre-wired, but if you use the blue metal or bi-metal motors for lower speed/higher torque, you'll need to solder on your own wires as shown here.

You don't need to worry too much about which wire is connected to which tab, as everything can be adjusted in software. These motors can be driven in either direction by telling the Crickit motor drivers to flip the polarity in software.

Any of the TT Motors from Adafruit will work well in this project. The blue ones are slower but have more torque, so can deal better with obstacles, the yellow ones are zippier!
NeoPixel Prep

While you've got the soldering iron out, you'll also want to connect three wires to the NeoPixel ring's DATA IN, PWR, and GND pads.
Motor Mounts

It's very easy to mount the motors, as the chassis is designed for their exact dimensions!

Place each motor into its spot and secure with one or two of the long screws and nuts.
Wheels
Stretch the tires onto the wheels and then press the wheels on to the motor shafts.
Connect Motor Wires to Crickit

Now, push the motor wires through an opening in the chassis so they can be connected to the Crickit.

The pair of wires from one motor will go into the two positions marked '1' and the pair of wires from the other motor will go into the the two positions marked '2'.

Push each wire into place and then screw down the terminal securely.
We're using the Motor 1 and Motor 2 ports, each can act as bi-polar motor drivers. There is no need to plug a wire into the center position that is marked "GND".

Underbody Lighting Effects

Next we'll connect the NeoPixel ring under the chassis for sweet lighting effects, especially at night, in the rain, on the wet asphalt at an illegal street race!
Push the three NeoPixel wires from the bottom through a hole in the chassis so they emerge up top near the Crickit NeoPixel port.

Use four zip ties to connect the ring to the chassis as shown.

Screw the DATA IN, PWR, and GND wires into their respective ports on the Crickit NeoPixel terminal block.

Battery Power

Now, we'll give the rover a power source!

First, connect the battery pack’s jumper wires to the plug adapter. Red to +, Black to -
Feed a zip tie through the two screw holes on the back of the chassis as shown, from out to in and back out.

If your zip tie is too short to make it all the way around the battery pack, extend its length with a second zip tie as shown.

Form a loose ring with the zip ties, then repeat these steps for the second position.

Place four NiMH batteries (not Alkaline, they have too high a voltage and won't work with the Crickit!) into the holder, then slide the plug through one side and secure the pack in place.

Now, tighten the zip ties, but not too tight. You want to still be able to remove the pack to change batteries.
Plug in the power plug into the Crickit's power jack and now we're ready to program it!

This arrangement works well with the blue TT motors, but may be a bit back heavy for the yellow motors. In this case, you can put the battery pack closer to the wheels by attaching it at the other end of the chassis, or underneath.
Code the Rover with CircuitPython

Feather and Crickit Prep

We’ll be using CircuitPython for this project. Are you new to using CircuitPython? No worries, there is a full getting started guide here (https://adafruit.it/cpy-welcome).

For more info on the Feather nRF52840 Express, check out this guide (https://adafruit.it/DLI).

Adafruit suggests using the Mu editor to edit your code and have an interactive REPL in CircuitPython. You can learn about Mu and its installation in this tutorial (https://adafruit.it/ANO).

Follow this guide for instructions on installing the latest release version of CircuitPython for the Feather nRF52840 Express:

Installing CircuitPython
https://adafruit.it/Amd

Libraries

You’ll also need to add the following libraries for this project. Follow this guide (https://adafruit.it/Cqa) on adding libraries.
Plug your nRF Feather board into your computer via a USB cable. Please be sure the cable is a good power+data cable so the computer can talk to the board.

A new disk should appear in your computer's file explorer/finder called CIRCUITPY. This is the place we'll copy the code and code library. If you can only get a drive named FTHR840BOOT, load CircuitPython per the guide above.

Make a folder on the CIRCUITPY drive named lib.

Now, download the latest CircuitPython libraries to your computer using the green button below. Match the library you get to the version of CircuitPython you are using. Save to your computer's hard drive where you can find it.

Note: This project uses the Bluetooth functionality available in CircuitPython 4.0.0-beta.1 and later, on nRF52840 boards such as the Feather nRF52840 Express. Make sure you are using that version or later.

With your file explorer/finder, browse to the bundle and open it up. Copy the following folders and files from the library bundle to your CIRCUITPY lib directory you made earlier:

The ones you'll need are:

- adafruit_ble (folder)
- adafruit_bluefruit_connect (folder)
- adafruit_bus_device (folder)
- adafruit_crickit.mpy (file)
- adafruit_motor (folder)
- adafruit_seesaw (folder)

All of the other necessary code is baked into CircuitPython!
CircuitPython Code

Copy the program below and paste it into a new document in Mu. Then, save it from Mu onto your CIRCUITPY flash drive as code.py. As soon as CircuitPython restarts, the NeoPixel ring will light up purple-ish and await your Bluetooth connection!

```python
# CircuitPython BLE Rover
# Use with the Adafruit BlueFruit LE Connect app
# Works with CircuitPython 4.0.0-beta.1 and later
# running on an nRF52840 Feather board and Crickit FeatherWing

import time
import board
import digitalio
from adafruit_crickit import crickit
from adafruit_ble import BLERadio
from adafruit_ble.advertising.standard import ProvideServicesAdvertisement
from adafruit_ble.services.nordic import UARTService

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

# Prep the status LEDs on the Feather
blue_led = digitalio.DigitalInOut(board.BLUE_LED)
red_led = digitalio.DigitalInOut(board.RED_LED)
blue_led.direction = digitalio.Direction.OUTPUT
red_led.direction = digitalio.Direction.OUTPUT

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

# motor setup
motor_1 = crickit.dc_motor_1
motor_2 = crickit.dc_motor_2

FWD = -1.0
REV = 0.7
```

© Adafruit Industries
crickit.init_neopixel(24, brightness = 0.2)  # create Crickit neopixel object
RED = (200, 0, 0)
GREEN = (0, 200, 0)
BLUE = (0, 0, 255)
PURPLE = (120, 0, 160)
YELLOW = (100, 100, 0)
AQUA = (0, 100, 100)
color = PURPLE  # current NeoPixel color
prior_color = PURPLE  # to store state of previous color when changing them
crickit.neopixel.fill(color)

print("BLE Rover")
print("Use Adafruit Bluefruit app to connect")
while True:
    blue_led.value = False
    ble.start_advertising(advertisement)
    while not ble.connected:
        # Wait for a connection.
        pass
    blue_led.value = True  # turn on blue LED when connected
while ble.connected:
    if uart_service.in_waiting:
        red_led.value = False  # turn off red LED
        packet = Packet.from_stream(uart_service)
        if isinstance(packet, ColorPacket):
            # Change the color.
            color = packet.color
            crickit.neopixel.fill(color)

    # do this when buttons are pressed
    if isinstance(packet, ButtonPacket) and packet.pressed:
        if packet.button == ButtonPacket.UP:  # UP button pressed
            red_led.value = True  # blink to show a packet has been received
            if packet.button == ButtonPacket.UP:
                # UP button pressed
                crickit.neopixel.fill(color)
                motor_1.throttle = FWD
                motor_2.throttle = FWD
            elif packet.button == ButtonPacket.DOWN:  # DOWN button
                crickit.neopixel.fill(color)
                motor_1.throttle = REV
                motor_2.throttle = REV
            elif packet.button == ButtonPacket.RIGHT:
                prior_color = color
                color = YELLOW
                crickit.neopixel.fill(color)
                motor_1.throttle = FWD
                motor_2.throttle = FWD * 0.5
        elif packet.button == ButtonPacket.LEFT:
            prior_color = color
            color = YELLOW
            crickit.neopixel.fill(color)
            motor_1.throttle = FWD * 0.5
            motor_2.throttle = FWD
        elif packet.button == ButtonPacket.BUTTON_1:
            crickit.neopixel.fill(RED)
            motor_1.throttle = 0.0
            motor_2.throttle = 0.0
            time.sleep(0.5)
            crickit.neopixel.fill(color)
        elif packet.button == ButtonPacket.BUTTON_2:
            color = GREEN
            crickit.neopixel.fill(color)
        elif packet.button == ButtonPacket.BUTTON_3:
            color = BLUE
            crickit.neopixel.fill(color)
        elif packet.button == ButtonPacket.BUTTON_4:
            color = PURPLE
            crickit.neopixel.fill(color)
# do this when some buttons are released

```python
elif isinstance(packet, ButtonPacket) and not packet.pressed:
    if packet.button == ButtonPacket.RIGHT:
        print("released right")
        color = prior_color
        crickit.neopixel.fill(color)
        motor_1.throttle = FWD
        motor_2.throttle = FWD
    if packet.button == ButtonPacket.LEFT:
        print("released left")
        color = prior_color
        crickit.neopixel.fill(color)
        motor_1.throttle = FWD
        motor_2.throttle = FWD
```

## How the Code Works

The first part of the program sets up a `UARTServer`, which is a BLE service that wirelessly sends streams of characters between a BLE device and a client computer (which could be a phone or tablet). The program then starts advertising its UART service to anyone listening. The client computer receives the advertisement, and tells the user it's available. The user can then choose to connect as a client of the service.

One a connection is established, the program code waits for incoming command packets from the client, either a `ColorPacket` or a `ButtonPacket`. When one arrives, the program waits to see which button has been pressed or which color value is sent and then either runs the motors or changes the colors on the NeoPixels.

## Motor Movement

We set up a couple of variables -- `FWD` and `REV` -- that tell the program which base values to send for forward and reverse. Depending on how you wired the motors, and which way you want to consider "forward" on the cart, you can adjust these values anywhere from `-1.0` to `1.0`.

When you test the rover, if it goes backwards when you press the up button, you can switch the `FWD` variable from `-1.0` to `1.0` and flip the `REV` value from `0.7` to `-0.7`.

If you find the rover spinning in circle when you press up or down on the remote app, then one of the motors is wired "backwards". Simply swap the two wires of that motor into each others' ports on the Crickit!
Control the Rover with the Bluefruit LE Connect App

Download the App

To control the motors and NeoPixels in from a client device (phone or tablet), you use the Adafruit Bluefruit LE Connect App (https://adafruit.it/iCi). Install it from the Apple App Store (https://adafruit.it/BYj) or Google Play App Store (https://adafruit.it/f4G).

Connect to Feather

With the Crickit plugged into DC power and turned on with its on-board power switch, the Feather will also get power and automatically start running your code. Then start up the Bluefruit LE Connect app, and make sure it's in Central Mode (left button on the bottom). When you start the app, you should see a device named CIRCU or CIRC UITPY. If there's a long list of devices, you can shorten it by turning on the "Must have UART Service" switch.

To connect to the Light Switch, touch the Connect button. You should see Connecting and then Discovering Services.
Once connected, you can use the Controllers > Control Pad to drive with the arrow buttons, stop with the 1 button, and change to some preset colors on 2, 3, & 4.
Troubleshooting

If you don't see CIRCU or CIRCUITPY right away, try pulling down to refresh. If that doesn't work, try turning Bluetooth off and back on on your phone or tablet and restarting the app.

Device Menu

After you connect, you'll see a menu of how you can interact with the device. Choose Controller.
Controller Menu

After you choose Controller, you'll see another screen with more choices. Control Pad is used for this project.
Change Color

To change the color of the NeoPixels, use the Color Picker screen. Choose a color and press Send Selected Color.