# Table of Contents

**Overview**

**Pinouts**

**Python & CircuitPython**
- Additional Necessary Hardware
- CircuitPython Microcontroller Wiring
- Encoder and NeoPixel Example

**Arduino**
- Additional Necessary Hardware
- Wiring
- Library Installation
- Load Example

**Downloads**
- Files
- Schematic and Fab Print
Overview

This funky user interface element is reminiscent of the original clicking scroll wheel interface on the first iPods. It's a fancy mechanical kit but has an intuitiveness that is hard to argue with - everyone knows how to use this kind of rotary encoder to scroll and select.

This product is just the encoder/button wheel element. The pin-out is a little odd, but, you could solder to the pins if you like.
There are 5 buttons (up down left right center) and a rotary encoder wheel in the center. Use with any microcontroller that can read pulse-code rotary encoders!

To deal with the odd pinout we made a handy breakout board that converts the funky pin set into a straightforward, breadboard-friendly header strip.
There's no pull-up or pull-down resistors on this PCB, and of course you'll need to solder the encoder onto the breakout. Then use your microcontroller's button and rotary encoder library/hardware support to interface with the pins.

You'll need 7 GPIO total: 5 buttons and 2 rotary encoder pins. There's also two COMmon pins, which you can set to ground or VCC - usually ground so that you can use the microcontroller internal pull-ups for the button/encoders. Note: to make our wiring simple, our example code uses GPIO to the COM's and then sets then to outputs, but you can just wire them directly.
Pinouts

The ANO Rotary Navigation Encoder breakout has 9 pins. Here are the details.

- ENCA - This is the rotary encoder A pin.
- ENCB - This is the rotary encoder B pin.
- SW1 - This pin is the center button.
- SW2 - This pin is the down button.
- SW3 - This pin is the right button.
- SW4 - This pin is the up button.
- SW5 - This pin is the left button.
- COMA - Common pin A. Set to ground or VCC - usually ground - so that you can use the microcontroller internal pull-ups for the button/encoders. You can also use a GPIO pin and pull it low.
- COMB - Common pin B. Set to ground or VCC - usually ground - so that you can use the microcontroller internal pull-ups for the button/encoders. You can also use a GPIO pin and pull it low.

Python & CircuitPython

The ANO Directional Navigation and Scroll Wheel Rotary Encoder is made up of 5 buttons and a two-direction rotary encoder. CircuitPython has two built in modules - `digitalio` and `rotaryio` - designed to work with buttons and rotary encoders, making it super simple to get started with this encoder! These modules make it easy
to write Python code that allows you to read button presses and encoder rotation from the encoder.

This example uses a 12-pixel NeoPixel ring, along with the rotary encoder, to use the encoder to light up various pixels.

### Additional Necessary Hardware

**NeoPixel Ring - 12 x 5050 RGB LED with Integrated Drivers**

Round and round and round they go! 12 ultra bright smart LED NeoPixels are arranged in a circle with 1.5" (37mm) outer diameter. The rings are 'chainable' - connect the...


**Hook-up Wire Spool Set - 22AWG Solid Core - 6 x 25 ft**

Perfect for bread-boarding, free wiring, etc. This box contains 6 spools of solid-core wire. The wire is easy to solder to and when bent it keeps its shape pretty well. We like to have...

[https://www.adafruit.com/product/1311](https://www.adafruit.com/product/1311)

### CircuitPython Microcontroller Wiring

Solder up the encoder to the breakout, and wire up the NeoPixel ring. Then, wire up the breakout and ring as shown below. This is an example of wiring the rotary encoder breakout and a NeoPixel ring to a Feather M4 Express.

This method of wiring the breakout takes advantage of the fact that the short side of a Feather has, in a row, exactly the 7 GPIO pins needed for the breakout. You can also wire this up to ANY microcontroller (with 7+ GPIO available) using jumper wires!
NeoPixel ring data in to Feather A0
NeoPixel ring ground to Feather GND
NeoPixel ring power to Feather 3.3v
Breakout ENCA to Feather D13
Breakout ENCB to Feather D12
Breakout COMA to Feather D11
Breakout SW1 to Feather D10
Breakout SW2 to Feather D9
Breakout SW3 to Feather D6
Breakout SW4 to Feather D5
Breakout SW5 to Feather SCL
Breakout COMB to Feather SDA

The wiring shown here ties COMA/B to digital outputs set LOW in code (instead of
connected to GND) to allow for the direct-to-Feather connection.

Encoder and NeoPixel Example

To use with CircuitPython, you need to first install the NeoPixel library, into the lib
folder on your CIRCUITPY drive. Then you need to update code.py with the example
script.

Thankfully, you can do this in one go. In the example below, click the Download
Project Bundle button below to download the necessary libraries and the code.py file
in a zip file. Extract the contents of the zip file, and copy the entire lib folder and the c
ode.py file to your CIRCUITPY drive.

Your CIRCUITPY/lib folder should contain the following folder and file:

• neopixel.mpy

    # SPDX-FileCopyrightText: 2021 Kattni Rembor for Adafruit Industries
    #
    # SPDX-License-Identifier: MIT
    
    """
    CircuitPython ANO Rotary Encoder and NeoPixel Ring example.
    """
    import board
    import digitalio
    import rotaryio
    import neopixel

    # The pin assignments for the breakout pins. Update this if you are not using a
    Feather.
    ENCA = board.D13
    ENCB = board.D12
    COMA = board.D11
SW1 = board.D10
SW2 = board.D9
SW3 = board.D6
SW4 = board.D5
SW5 = board.SCL
COMB = board.SDA

# Rotary encoder setup
encoder = rotaryio.IncrementalEncoder(ENCA, ENCB)
last_position = None

# NeoPixel ring setup. Update num_pixels if using a different ring.
num_pixels = 12
pixels = neopixel.NeoPixel(board.A0, num_pixels, auto_write=False)

# Set the COMA and COMB pins LOW. This is only necessary when using the direct-to-
# Feather or other
# GPIO-based wiring method. If connecting COMA and COMB to ground, you do not need
to include this.
com_a = digitalio.DigitalInOut(COMA)
com_a.switch_to_output()
com_a = False
com_b = digitalio.DigitalInOut(COMB)
com_b.switch_to_output()
com_b = False

# Button pin setup
button_pins = (SW1, SW2, SW3, SW4, SW5)
buttons = []
for button_pin in button_pins:
    pin = digitalio.DigitalInOut(button_pin)
    pin.switch_to_input(digitalio.Pull.UP)
    buttons.append(pin)

while True:
    position = encoder.position
    if last_position is None or position != last_position:
        print("Position: {}\n        last_position = position

        pixels.fill((0, 0, 0))
        pixels[position % num_pixels] = (0, 150, 0)
        if not buttons[0].value:
            print("Center button!")
            pixels.fill((100, 100, 100))
        if not buttons[1].value:
            print("Up button!")
            pixels[0] = (150, 0 ,0)
        if not buttons[2].value:
            print("Left button!")
            pixels[3] = (150, 0, 0)
        if not buttons[3].value:
            print("Down button!")
            pixels[6] = (150, 0, 0)
        if not buttons[4].value:
            print("Right button!")
            pixels[9] = (150, 0, 0)

        pixels.show()
Rotate the encoder to see the green NeoPixel move around the NeoPixel ring. Press the outer directional buttons to see a red LED light up in a corresponding location on the ring. Finally, click the center button to see the entire ring light up white.

First you import the necessary modules and library.

Next, you provide the pin assignments. The variables are in order of the breakout, left to right, to make it easier to provide the pins you used to connect the breakout to your microcontroller. Update this if you connected the breakout in a different way than the wiring diagram above. Remember, the COMA and COMB pins are not needed in the code, if you connected them to ground.

Then, you setup the hardware. First up, you setup the rotary encoder, and create the `last_position` variable and set it to `False`. Second, you create the `num_pixels` variable and set it to the number of pixels on whatever NeoPixel hardware you're using. In the case of this ring, there are 12 pixels. Then you setup the NeoPixel ring, and set `auto_write=False` so that you must call `show()` to send any updates to the pixels. This eliminates flickering that occurs when updates are otherwise spammed throughout the loop.

Next, you setup the two COM pins as outputs and set them low by making them `False`.

Then, you setup the buttons. There are 5 buttons, and setup for all of them is the same. So, instead of doing all of them individually, you do them all at the same time. First, you create a Python tuple of the button pins. Then you create an empty list to hold your setup buttons. Finally, you setup each button, and append it to the list until all setup has been completed. Now, `buttons` contains a list of button objects available for you to use in your code.

Inside the loop, you first create the `position` variable and set it equal to the initial encoder position. Then, you check it against the `last_position`, and if it's changed, print the position to the serial console.

Next, you clear the pixels so the rest of the pixel updates show up as expected with encoder rotation and button presses.

Then, you turn on one pixel green, that is equal to the rotary encoder position, mapped out to the possible values of 0 to 11 (12 total values, 12 total pixels). This means as you rotate the encoder past 12 or -12, the pixel continues to loop around the ring.
Next, you check the button states. First, you check to see if the center button is pressed, and if so, light up the LEDs white, and print to the serial console. Then you check each of the outside buttons, light up a red LED on the ring that corresponds to the location of the button, and print to the serial console.

Finally, you call `pixels.show()` to update the NeoPixels.

That's all there is to using the ANO rotary encoder with a NeoPixel ring using CircuitPython!

---

**Arduino**

Using the ANO Directional Navigation and Scroll Wheel Rotary Encoder with Arduino involves wiring it up, installing the necessary libraries, and running the example code. This example also uses a NeoPixel ring. Therefore, you'll need to install the RotaryEncoder and Adafruit NeoPixel libraries.
Additional Necessary Hardware

**NeoPixel Ring - 12 x 5050 RGB LED with Integrated Drivers**
Round and round and round they go! 12 ultra bright smart LED NeoPixels are arranged in a circle with 1.5" (37mm) outer diameter. The rings are 'chainable' - connect the...
https://www.adafruit.com/product/1643

**Hook-up Wire Spool Set - 22AWG Solid Core - 6 x 25 ft**
Perfect for bread-boarding, free wiring, etc. This box contains 6 spools of solid-core wire. The wire is easy to solder to and when bent it keeps its shape pretty well. We like to have...
https://www.adafruit.com/product/1311

Wiring

First, solder the rotary encode to the breakout, and wires to the NeoPixel ring. Then, follow the example below to connect both to a Feather M4.

This method of wiring the breakout takes advantage of the fact that the short side of a Feather has, in a row, exactly the 7 GPIO pins needed for the breakout. You can also wire this up to ANY Arduino board (with 7+ GPIO available) using jumper wires!
NeoPixel ring data in to Feather A0
NeoPixel ring ground to Feather GND
NeoPixel ring power to Feather 3.3v
Breakout ENCA to Feather D13
Breakout ENCB to Feather D12
Breakout COMA to Feather D11
Breakout SW1 to Feather D10
Breakout SW2 to Feather D9
Breakout SW3 to Feather D6
Breakout SW4 to Feather D5
Breakout SW5 to Feather SCL
Breakout COMB to Feather SDA

The wiring shown here ties COMA/B to digital outputs set LOW in code (instead of connected to GND) to allow for the direct-to-Feather connection.

**Library Installation**

You can install the RotaryEncoder and NeoPixel libraries for Arduino using the Library Manager in the Arduino IDE.

Click the **Manage Libraries ...** menu item, search for RotaryEncoder, and select the RotaryEncoder library (by Matthias Hertel):

Then search for NeoPixel and select the Adafruit NeoPixel library:
Load Example

Click Download Project Bundle to download the example to your computer. Open the ANO_Rotary_Encoder_NeoPixel_Arduino_Example.ino file in the Arduino IDE.

```c
#include <Adafruit_NeoPixel.h>
#include <RotaryEncoder.h>

#define PIN_ENCODER_A 13
#define PIN_ENCODER_B 12
#define COM_A    11
#define COM_B    SDA
#define BUTTON_UP 5
#define BUTTON_LEFT SCL
#define BUTTON_DOWN 9
#define BUTTON_RIGHT 6
#define BUTTON_IN 10

RotaryEncoder encoder(PIN_ENCODER_A, PIN_ENCODER_B,
    RotaryEncoder::LatchMode::TWO03);
// This interrupt will do our encoder reading/checking!
void checkPosition() {
    encoder.tick(); // just call tick() to check the state.
}
int last_rotary = 0;

#define NUMPIXELS 12
Adafruit_NeoPixel pixels(NUMPIXELS, A0, NEO_GRB + NEO_KHZ800);

void setup(void) {
    Serial.begin(115200);
    while (!Serial);
    Serial.println("ANO Rotary Encoder Demo");

    pinMode(COM_A, OUTPUT);
    digitalWrite(COM_A, LOW);
    pinMode(COM_B, OUTPUT);
    digitalWrite(COM_B, LOW);

    attachInterrupt(digitalPinToInterrupt(PIN_ENCODER_A), checkPosition, CHANGE);
    attachInterrupt(digitalPinToInterrupt(PIN_ENCODER_B), checkPosition, CHANGE);

    pinMode(BUTTON_UP, INPUT_PULLUP);
    pinMode(BUTTON_DOWN, INPUT_PULLUP);
    pinMode(BUTTON_LEFT, INPUT_PULLUP);
    pinMode(BUTTON_RIGHT, INPUT_PULLUP);
    pinMode(BUTTON_IN, INPUT_PULLUP);
    pixels.begin();
    pixels.setBrightness(30);
    pixels.show();
}

void loop(void) {
    // read encoder
    int curr_rotary = encoder.getPosition();
    RotaryEncoder::Direction direction = encoder.getDirection();

    pixels.clear();
    pixels.show();
}
```

©Adafruit Industries
if (curr_rotary != last_rotary) {
    Serial.print("Encoder value: ");
    Serial.print(curr_rotary);
    Serial.print(" direction: ");
    Serial.println((int)direction);
}
last_rotary = curr_rotary;

pixels.setPixelColor((curr_rotary + (1000*NUMPIXELS)) % NUMPIXELS, pixels.Color(0, 150, 0));
if (! digitalRead(BUTTON_UP)) {
    pixels.setPixelColor(0, pixels.Color(150, 0, 0));
} if (! digitalRead(BUTTON_LEFT)) {
    pixels.setPixelColor(NUMPIXELS/4, pixels.Color(150, 0, 0));
} if (! digitalRead(BUTTON_DOWN)) {
    pixels.setPixelColor(NUMPIXELS/2, pixels.Color(150, 0, 0));
} if (! digitalRead(BUTTON_RIGHT)) {
    pixels.setPixelColor(NUMPIXELS*3/4, pixels.Color(150, 0, 0));
} if (! digitalRead(BUTTON_IN)) {
    pixels.fill(pixels.Color(50, 50, 50));
} pixels.show();
delay(20);

After opening the demo file, upload to your Arduino wired up to the encoder. Once you upload the code, you will see the rotary encoder values when you rotate it being printed when you open the Serial Monitor (Tools->Serial Monitor) at 115200 baud, similar to this:

![Serial Monitor Output](image)

Rotate the encoder to see the green NeoPixel move around the NeoPixel ring. Press the outer directional buttons to see a red LED light up in a corresponding location on the ring. Finally, click the center button to see the entire ring light up white.
Downloads

Files

- Rotary Encoder Datasheet ()
- Rotary Encoder Breakout Fritzing Object in the Adafruit Fritzing Library ()
- Rotary Encoder Breakout EagleCAD PCB files on GitHub ()
- 3D Models on GitHub ()

Schematic and Fab Print