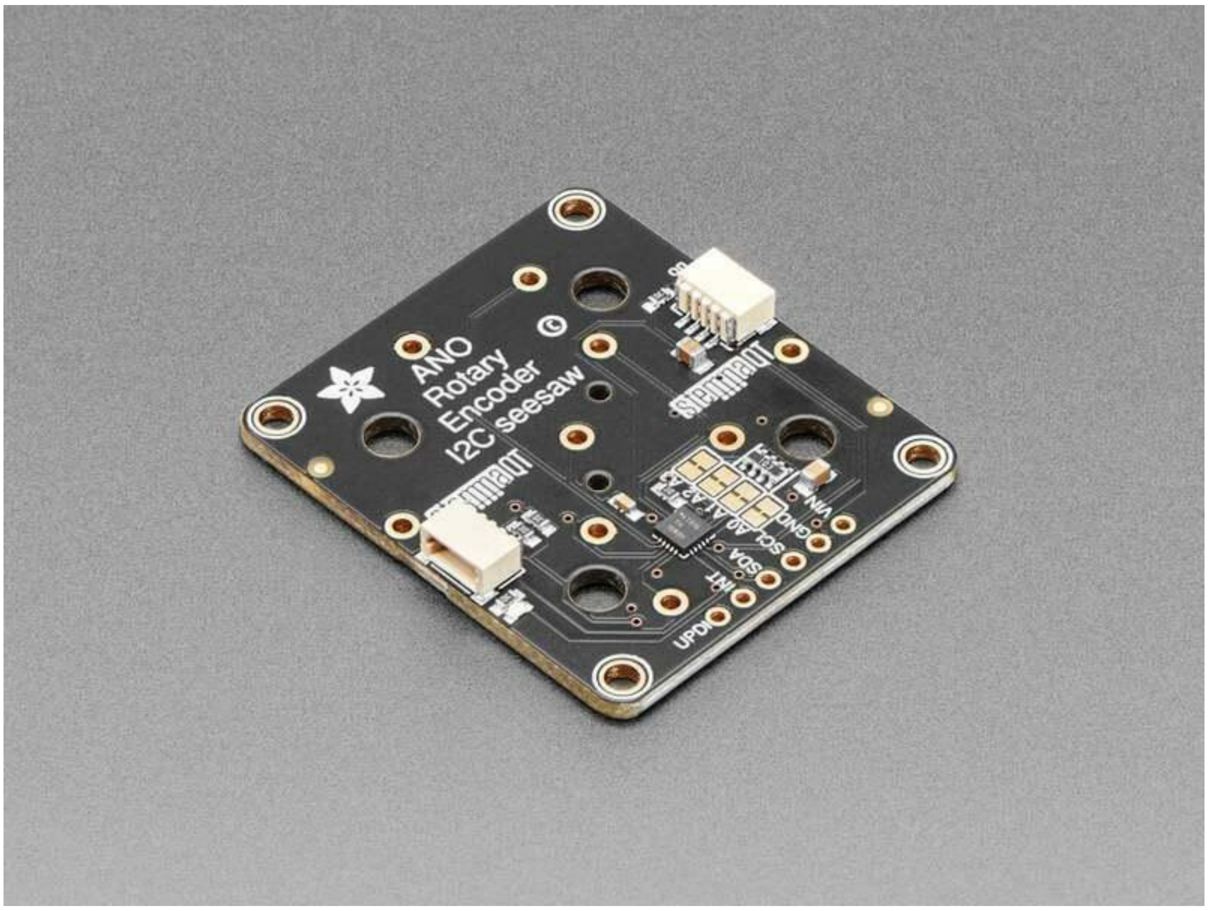
Adafruit ANO Rotary Encoder to I2C Adapter

Created by Liz Clark

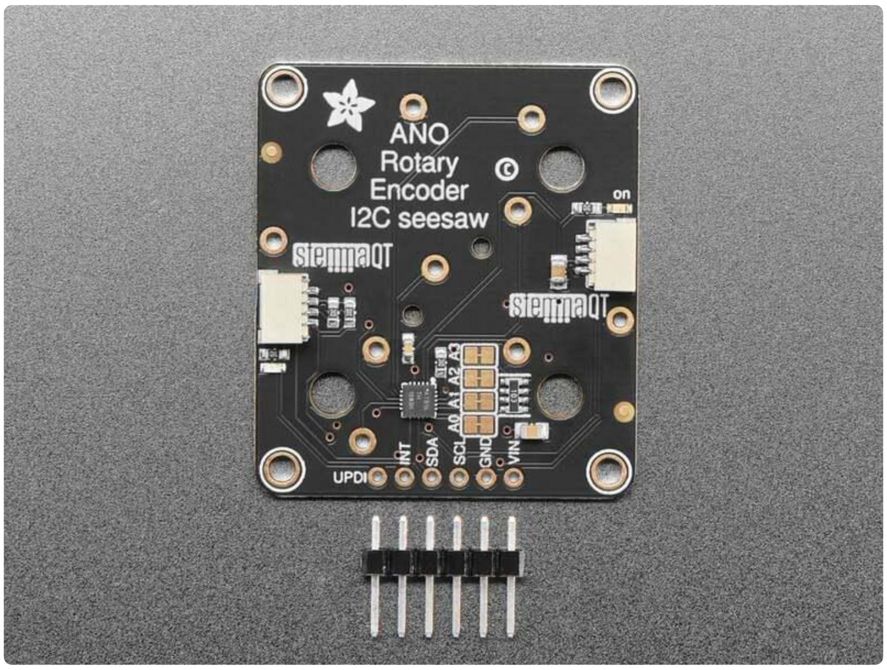


https://learn.adafruit.com/adafruit-ano-rotary-navigation-encoder-to-i2c-stemma-qt-adapter

Last updated on 2023-08-29 04:55:58 PM EDT

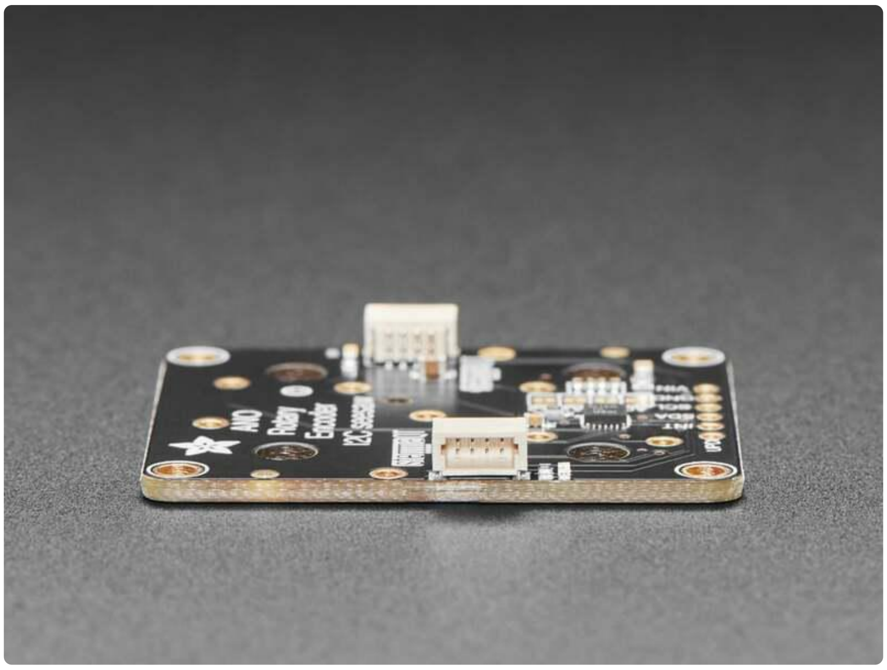
Table of Contents

Overview



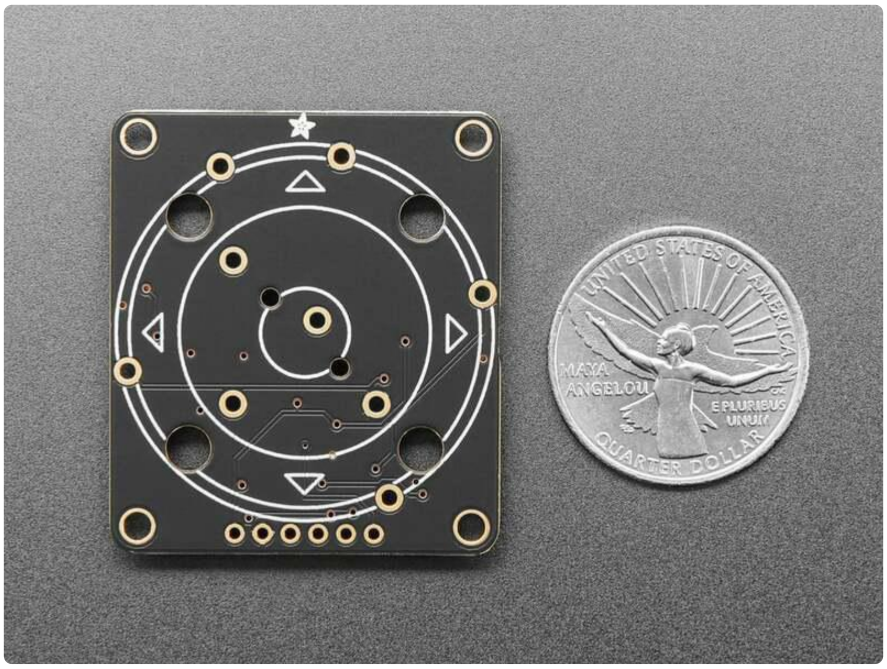
The ANO rotary encoder wheel is a funky user interface element, reminiscent of the original clicking scroll wheel interface on the first iPods (). It's a nifty kit, but the pin-out is a little odd - and there are a ton of pins needed to connect to the rotary encoder and 5 button switches.

This Stemma QT breakout makes all that frustration go away - solder in the ANO Directional Navigation and Scroll Wheel Rotary Encoder (not included) (). The onboard microcontroller is programmed with our seesaw firmware and will track all pulses and pins for you and then save the incremental value for querying at any time over I2C. Plug it in with a Stemma QT cable for instant rotary goodness, with any kind of microcontroller from an Arduino UNO up to a Raspberry Pi to one of our QT Py's.



You can use our Arduino library to control and read data () with any compatible microcontroller. We also have CircuitPython/Python code () for use with computers or single-board Linux boards.

Power with 3 to 5V DC, and then use 3 or 5V logic I2C data. The INT pin can be configured to pulse low whenever rotation or push-buttoning is detected so you do not have to spam-read the I2C port to detect motion. It's also easy to add this breakout to a breadboard - with six 0.1"-spaced breakout pads - if you so choose.

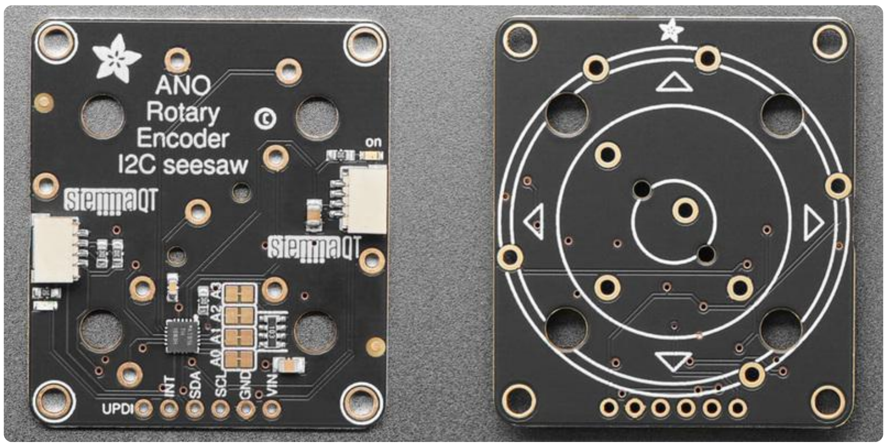


Four solder jumpers can be used to change the I2C address - up to 16 can share one I2C bus! If you happen to need more, it's possible to set the I2C address with a special address-change command that is saved to the onboard non-volatile EEPROM memory.

This is just the assembled and programmed 'seesaw' PCB for the ANO encoder, the encoder is NOT included! Pick one up here (). Some soldering is required to attach the encoder to the backing.

ANO Directional Navigation and Scroll Wheel Rotary Encoder
This funky user interface element is reminiscent of the original clicking scroll wheel interface...
https://www.adafruit.com/product/5001

Pinouts



The default I2C address is 0x49.

Power Pins

- VIN - This is the power pin. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 3V microcontroller like a Feather RP2040, use 3V, or for a 5V microcontroller like Arduino, use 5V.
- GND - This is common ground for power and logic.

I2C Logic Pins

The default I2C address is 0x49.

- SCL - I2C clock pin, connect to your microcontroller I2C clock line. There's a 10K pullup on this pin.
- SDA - I2C data pin, connect to your microcontroller I2C data line. There's a 10K pullup on this pin.
- STEMMA QT () - These connectors allow you to connect to development boards with STEMMA QT / Qwiic connectors or to other things with various associated accessories ().

Address Jumpers

On the back of the board are four address jumpers, labeled A0, A1, A2 and A3, above the breakout pads along the bottom of the board. These jumpers allow you to chain up to 16 of these boards on the same pair of I2C clock and data pins. To do so, you cut the jumpers "open" by separating the two pads.

If you happen to need more than 16, it's possible to set the I2C address with a special address-change command that is saved to the onboard non-volatile EEPROM memory.

The default I2C address is 0x49. The other address options can be calculated by "adding" the A0/A1/A2/A3 to the base of 0x49.

A0 sets the lowest bit with a value of 1, A1 sets the next bit with a value of 2, A2 sets the next bit with a value of 4 and A3 sets the next bit with a value of 8. The final address is 0x49 + A3 + A2 + A1 + A0 which would be 0x58.

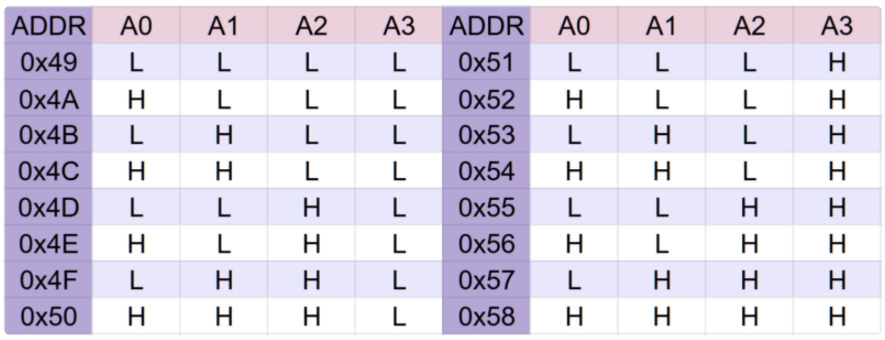
If only A0 is cut, the address is 0x49 + 1 = 0x4A

If only A1 is cut, the address is 0x49 + 2 = 0x4B

If only A2 is cut, the address is 0x49 + 4 = 0x4D

If only A3 is cut, the address is 0x49 + 8 = 0x51

The table below shows all possible addresses, and whether the pin(s) should be high (closed) or low (open).



ADDR	A0	A1	A2	A3	ADDR	A0	A1	A2	A3
0x49	L	L	L	L	0x51	L	L	L	H
0x4A	H	L	L	L	0x52	H	L	L	H
0x4B	L	H	L	L	0x53	L	H	L	H
0x4C	H	H	L	L	0x54	H	H	L	H
0x4D	L	L	H	L	0x55	L	L	H	H
0x4E	H	L	H	L	0x56	H	L	H	H
0x4F	L	H	H	L	0x57	L	H	H	H
0x50	H	H	H	L	0x58	H	H	H	H

Interrupt Pin and LED

- INT - This is the interrupt output pin. It can be configured to pulse low whenever rotation or push-buttoning is detected so you do not have to spam-read the I2C port to detect motion.
- Interrupt LED - On the back of the board below the STEMMA QT port on the left is the interrupt LED. It is the red LED and turns on whenever an interrupt is detected.

UPDI Pin

- UPDI - This is the single-pin Unified Program and Debug Interface. This pin is for external programming or on-chip-debugging for the ATtiny816 running the seesaw firmware (). We have a page in the ATtiny Breakouts with seesaw Learn Guide () detailing how to reprogram these chips with your own firmware (at your own risk). We don't provide any support for custom builds of seesaw - we think this is cool and useful for the Maker community.

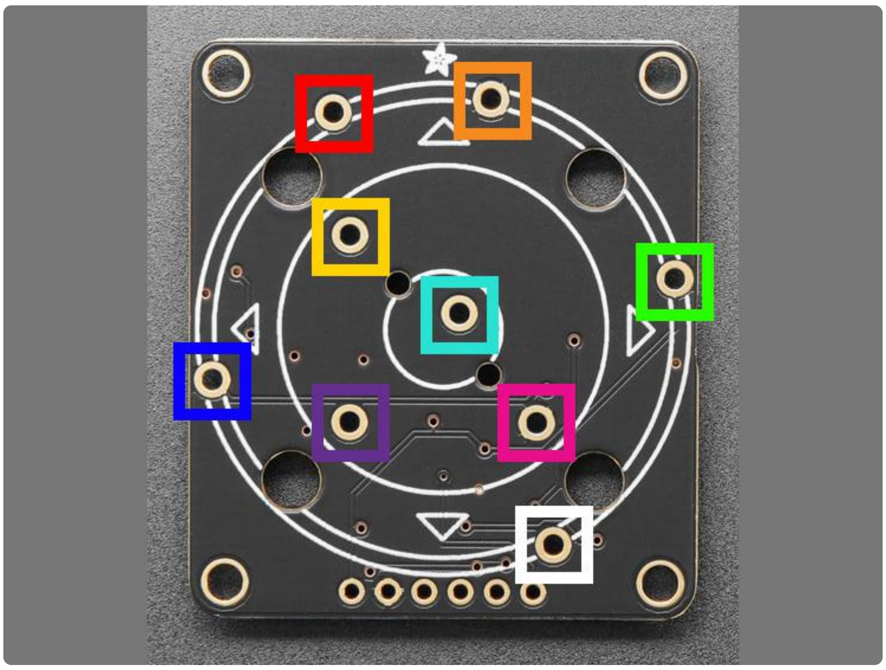
Power LED

- Power LED - On the back of the board, above the STEMMA connector on the right, is the power LED, labeled on. It is the green LED.

ANO Encoder Pins

On the front of the board is the outline of the ANO encoder on the board silk. This lets you know how you should place the ANO encoder on the board for soldering. The pins on the encoder are keyed so it only fits onto the breakout one way. Please note that the encoder is NOT included! You'll need to pick one up from the shop () and solder it to the board.

ANO Directional Navigation and Scroll Wheel Rotary Encoder
This funky user interface element is reminiscent of the original clicking scroll wheel interface...
https://www.adafruit.com/product/5001



- ENCA - This is the rotary encoder A pin (yellow square)
- ENCB - This is the rotary encoder B pin (purple square)
- SW1 - This pin is the select button, pin 1 in the seesaw firmware (aqua square)
- SW2 - This pin is the up button, pin 2 in the seesaw firmware (red square)
- SW3 - This pin is the left button, pin 3 in the seesaw firmware (blue square)
- SW4 - This pin is the down button, pin 4 in the seesaw firmware (white square)
- SW5 - This pin is the right button, pin 5 in the seesaw firmware (green square)

- COMA - Common pin A (pink square)
- COMB - Common pin B (orange square).

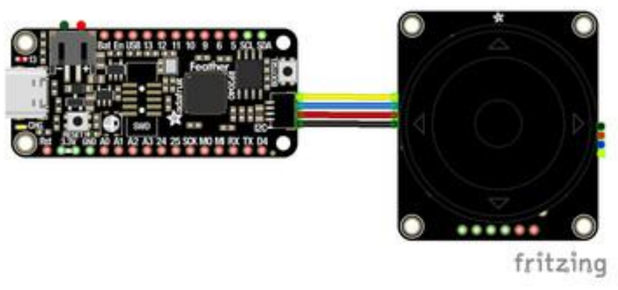
CircuitPython & Python

It's easy to use the ANO Rotary Encoder to I2C adapter with Python or CircuitPython, and the Adafruit_CircuitPython_seesaw () module. This module allows you to easily write Python code that reads encoder position (relative to the starting position) and the five button presses (up, down, left, right and select).

You can use this adapter with any CircuitPython microcontroller board or with a computer that has GPIO and Python thanks to Adafruit_Blinka, our CircuitPython-for-Python compatibility library ().

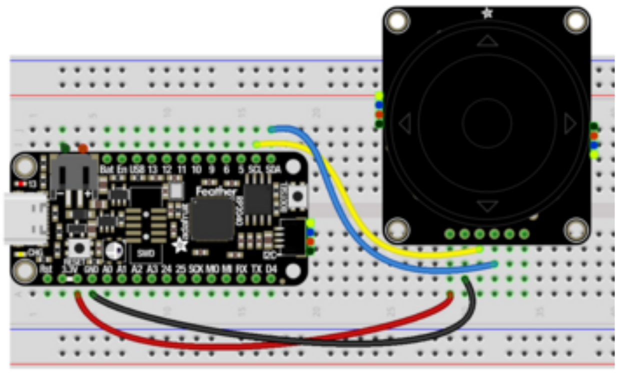
CircuitPython Microcontroller Wiring

First wire up an I2C adapter to your board exactly as follows. The following is the adapter wired to a Feather RP2040 using the STEMMA connector:



Board STEMMA 3V to adapter VIN (red wire)
Board STEMMA GND to adapter GND (black wire)
Board STEMMA SCL to adapter SCL (yellow wire)
Board STEMMA SDA to adapter SDA (blue wire)

The following is the adapter wired to a Feather RP2040 using a solderless breadboard:

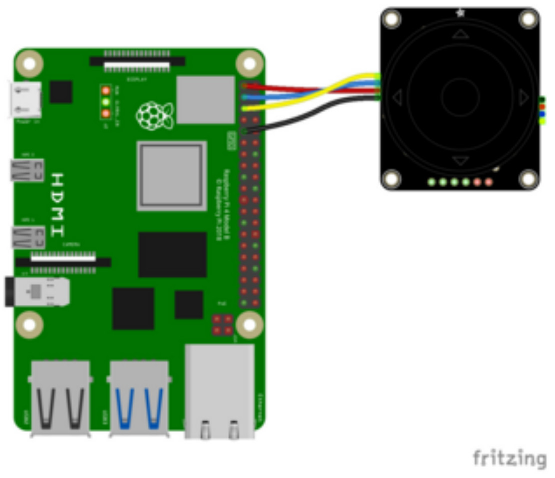


Board 3V to adapter VIN (red wire)
Board GND to adapter GND (black wire)
Board SCL to adapter SCL (yellow wire)
Board SDA to adapter SDA (blue wire)

Python Computer Wiring

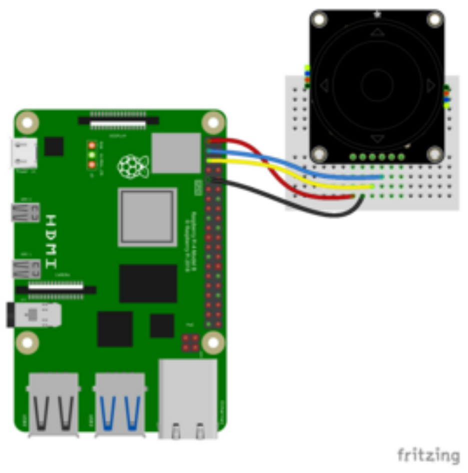
Since there's dozens of Linux computers/boards you can use we will show wiring for Raspberry Pi. For other platforms, please visit the guide for CircuitPython on Linux to see whether your platform is supported ().

Here's the Raspberry Pi wired with I2C using the STEMMA connector:



Pi 3V to adapter VIN (red wire)
Pi GND to adapter GND (black wire)
Pi SCL to adapter SCL (yellow wire)
Pi SDA to adapter SDA (blue wire)

Here's the Raspberry Pi wired with I2C using a solderless breadboard:



Pi 3V to adapter VIN (red wire)
Pi GND to adapter GND (black wire)
Pi SCL to adapter SCL (yellow wire)
Pi SDA to adapter SDA (blue wire)

Python Installation of seesaw Library

You'll need to install the Adafruit_Blinka library that provides the CircuitPython support in Python. This may also require enabling I2C on your platform and verifying you are running Python 3. Since each platform is a little different, and Linux changes often, please visit the CircuitPython on Linux guide to get your computer ready ()!

Once that's done, from your command line run the following command:

- `pip3 install adafruit-circuitpython-seesaw`

If your default Python is version 3 you may need to run 'pip' instead. Just make sure you aren't trying to use CircuitPython on Python 2.x, it isn't supported!

CircuitPython Usage

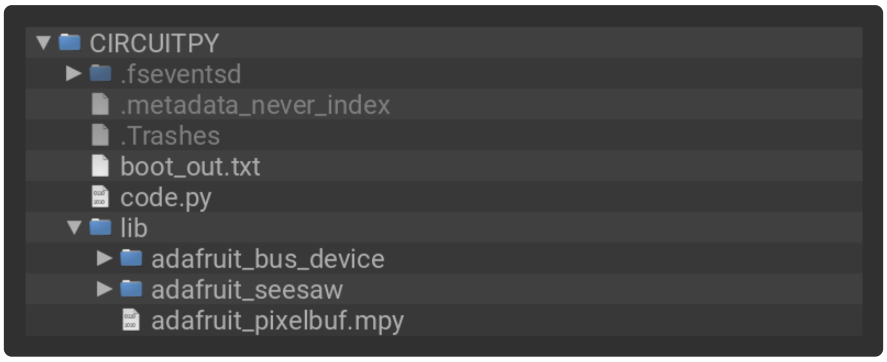
To use with CircuitPython, you need to first install the Adafruit_CircuitPython_seesaw library, and its dependencies, into the lib folder on your CIRCUITPY drive. Then you need to update code.py with the example script.

Thankfully, we 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 folders and file:

- adafruit_bus_device/
- adafruit_seesaw/

- adafruit_pixelbuf.mpy



```
▼ CIRCUITPY
    ▶ .fseventsd
      .metadata_never_index
      .Trashes
      boot_out.txt
      code.py
    ▼ lib
        ▶ adafruit_bus_device
        ▶ adafruit_seesaw
          adafruit_pixelbuf.mpy
```

Python Usage

Once you have the library `pip3` installed on your computer, copy or download the following example to your computer, and run the following, replacing code.py with whatever you named the file:

`python3 code.py`

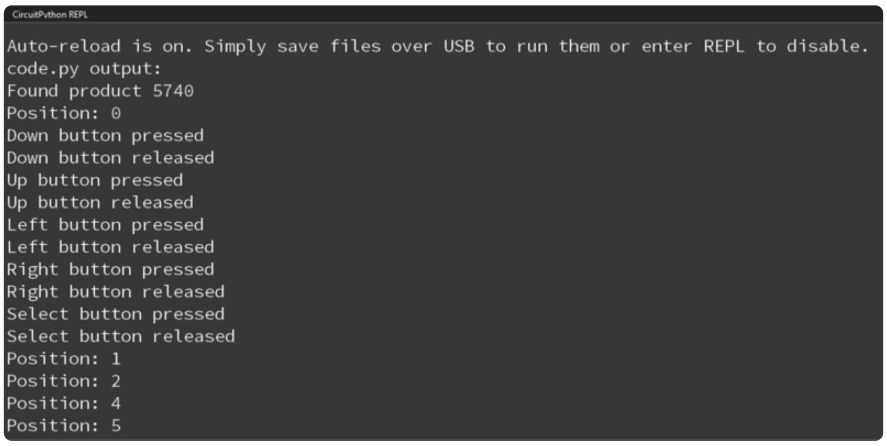
Simple Test Example Code

If running CircuitPython: Once everything is saved to the CIRCUITPY drive, connect to the serial console () to see the data printed out!

If running Python: The console output will appear wherever you are running Python.

```
# SPDX-FileCopyrightText: 2021 John Furcean
# SPDX-License-Identifier: MIT

"""I2C ANO rotary encoder simple test example."""

import board
from adafruit_seesaw import seesaw, rotaryio, digitalio

# For use with the STEMMA connector on QT Py RP2040
# import busio
# i2c = busio.I2C(board.SCL1, board.SDA1)
# seesaw = seesaw.Seesaw(i2c, 0x49)

i2c = board.I2C()  # uses board.SCL and board.SDA
# i2c = board.STEMMA_I2C()  # For using the built-in STEMMA QT connector on a
microcontroller
seesaw = seesaw.Seesaw(i2c, addr=0x49)

seesaw_product = (seesaw.get_version() >> 16) & 0xFFFF
print(f"Found product {seesaw_product}")
if seesaw_product != 5740:
    print("Wrong firmware loaded?  Expected 5740")

seesaw.pin_mode(1, seesaw.INPUT_PULLUP)
seesaw.pin_mode(2, seesaw.INPUT_PULLUP)
seesaw.pin_mode(3, seesaw.INPUT_PULLUP)
seesaw.pin_mode(4, seesaw.INPUT_PULLUP)
seesaw.pin_mode(5, seesaw.INPUT_PULLUP)

select = digitalio.DigitalIO(seesaw, 1)
select_held = False
up = digitalio.DigitalIO(seesaw, 2)
up_held = False
left = digitalio.DigitalIO(seesaw, 3)
left_held = False
down = digitalio.DigitalIO(seesaw, 4)
down_held = False
right = digitalio.DigitalIO(seesaw, 5)
right_held = False

encoder = rotaryio.IncrementalEncoder(seesaw)
last_position = None

buttons = [select, up, left, down, right]
button_names = ["Select", "Up", "Left", "Down", "Right"]
button_states = [select_held, up_held, left_held, down_held, right_held]

while True:
    position = encoder.position

    if position != last_position:
        last_position = position
        print(f"Position: {position}")

    for b in range(5):
        if not buttons[b].value and button_states[b] is False:
            button_states[b] = True
            print(f"{button_names[b]} button pressed")

        if buttons[b].value and button_states[b] is True:
            button_states[b] = False
            print(f"{button_names[b]} button released")
```

In the simple test example, the code confirms that you have connected the ANO rotary encoder to I2C adapter by checking the seesaw firmware for the product ID number. After that, in the loop, the rotary encoder position is printed to the serial console. Additionally, if a button is pressed or released that is also printed to the serial console.



```
Auto-reload is on. Simply save files over USB to run them or enter REPL to disable.
code.py output:
Found product 5740
Position: 0
Down button pressed
Down button released
Up button pressed
Up button released
Left button pressed
Left button released
Right button pressed
Right button released
Select button pressed
Select button released
Position: 1
Position: 2
Position: 4
Position: 5
```

Python Docs

Python Docs ()

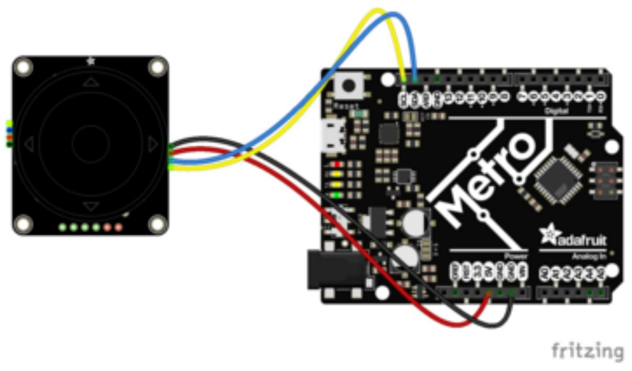
Arduino

Using the ANO rotary encoder to I2C adapter with Arduino involves wiring up the I2C adapter to your Arduino-compatible microcontroller, installing the Adafruit_Seesaw () library and running the provided example code.

Wiring

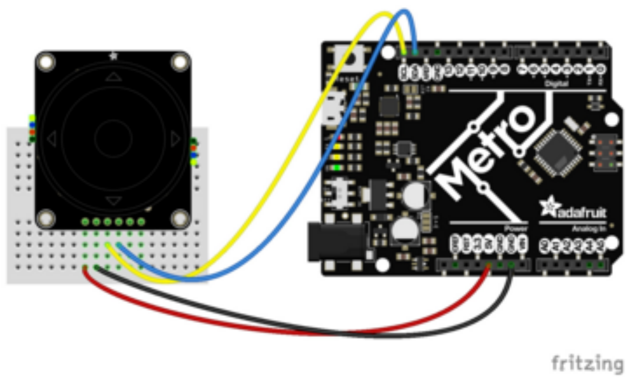
Wire as shown for a 5V board like an Uno. If you are using a 3V board, like an Adafruit Feather, wire the board's 3V pin to the adapter VIN.

Here is an Adafruit Metro wired up to the adapter using the STEMMA QT connector:



Board 5V to adapter VIN (red wire)
Board GND to adapter GND (black wire)
Board SCL to adapter SCL (yellow wire)
Board SDA to adapter SDA (blue wire)

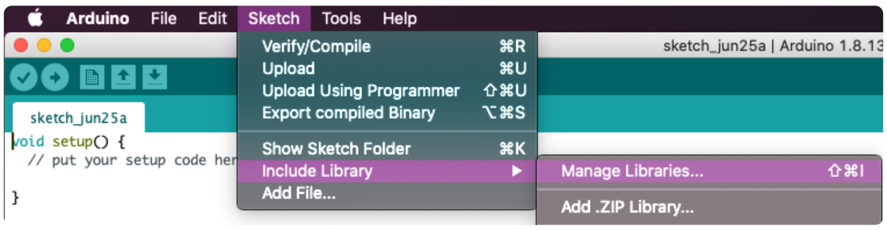
Here is an Adafruit Metro wired up using a solderless breadboard:



Board 5V to adapter VIN (red wire)
Board GND to adapter GND (black wire)
Board SCL to adapter SCL (yellow wire)
Board SDA to adapter SDA (blue wire)

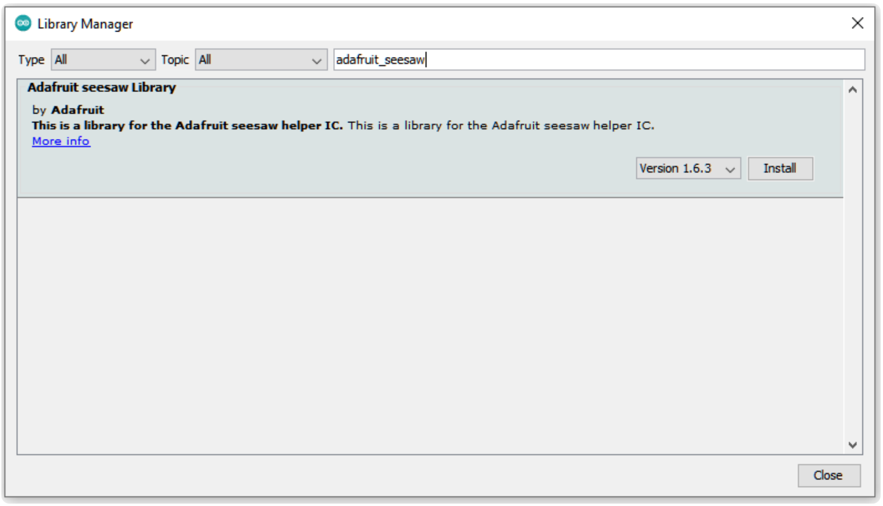
Library Installation

You can install the Adafruit_Seesaw library for Arduino using the Library Manager in the Arduino IDE.



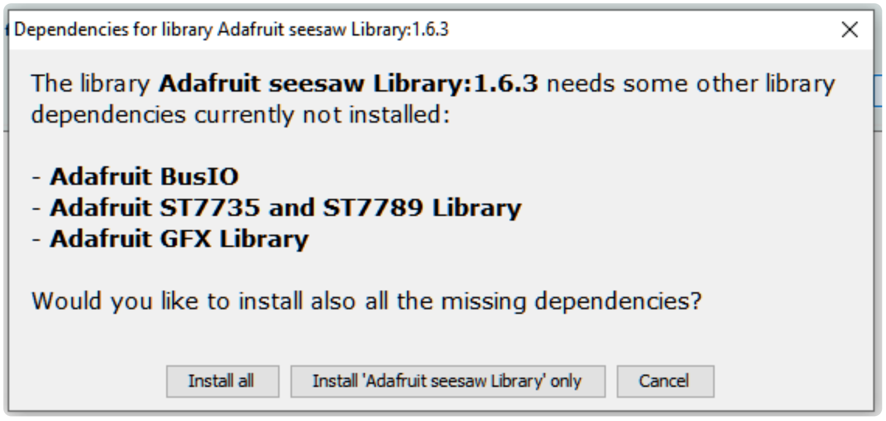
```
void setup() {
  // put your setup code her
}
```

Click the Manage Libraries ... menu item, search for Adafruit_Seesaw, and select the Adafruit seesaw Library library:



Adafruit seesaw Library
by **Adafruit**
This is a library for the Adafruit seesaw helper IC. This is a library for the Adafruit seesaw helper IC.
More info

If asked about dependencies, click "Install all".


Dependencies for library Adafruit seesaw Library:1.6.3

The library **Adafruit seesaw Library:1.6.3** needs some other library dependencies currently not installed:

- **Adafruit BusIO**
- **Adafruit ST7735 and ST7789 Library**
- **Adafruit GFX Library**

Would you like to install also all the missing dependencies?

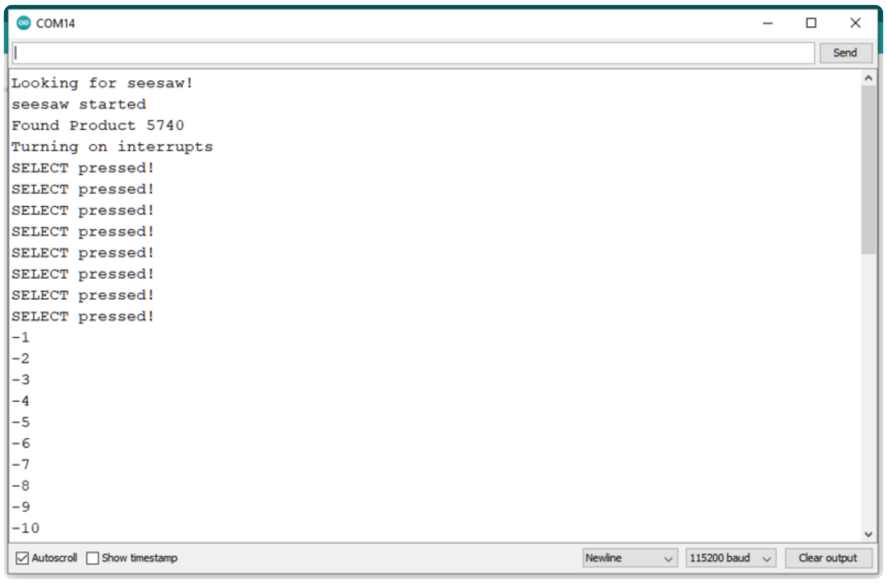
If the "Dependencies" window does not come up, then you already have the dependencies installed.

If the dependencies are already installed, you must make sure you update them through the Arduino Library Manager before loading the example!

Example Code

```
/*
 * This example shows how to read from a seesaw encoder module.
 * The available encoder API is:
 *      int32_t getEncoderPosition();
        int32_t getEncoderDelta();
        void enableEncoderInterrupt();
        void disableEncoderInterrupt();
 */
#include "Adafruit_seesaw.h"

#define SS_SWITCH_SELECT 1
#define SS_SWITCH_UP     2
#define SS_SWITCH_LEFT   3
#define SS_SWITCH_DOWN   4
#define SS_SWITCH_RIGHT  5

#define SEESAW_ADDR      0x49

Adafruit_seesaw ss;
int32_t encoder_position;

void setup() {
  Serial.begin(115200);
  while (!Serial) delay(10);

  Serial.println("Looking for seesaw!");

  if (! ss.begin(SEESAW_ADDR)) {
    Serial.println("Couldn't find seesaw on default address");
    while(1) delay(10);
  }
  Serial.println("seesaw started");
  uint32_t version = ((ss.getVersion() >> 16) & 0xFFFF);
  if (version  != 5740){
    Serial.print("Wrong firmware loaded? ");
    Serial.println(version);
    while(1) delay(10);
  }
  Serial.println("Found Product 5740");

  ss.pinMode(SS_SWITCH_UP, INPUT_PULLUP);
  ss.pinMode(SS_SWITCH_DOWN, INPUT_PULLUP);
  ss.pinMode(SS_SWITCH_LEFT, INPUT_PULLUP);
  ss.pinMode(SS_SWITCH_RIGHT, INPUT_PULLUP);
  ss.pinMode(SS_SWITCH_SELECT, INPUT_PULLUP);

  // get starting position
  encoder_position = ss.getEncoderPosition();

  Serial.println("Turning on interrupts");
  ss.enableEncoderInterrupt();
  ss.setGPIOInterrupts((uint32_t)1 << SS_SWITCH_UP, 1);

}

void loop() {
  if (! ss.digitalRead(SS_SWITCH_UP)) {
    Serial.println("UP pressed!");
  }
  if (! ss.digitalRead(SS_SWITCH_DOWN)) {
    Serial.println("DOWN pressed!");
  }
  if (! ss.digitalRead(SS_SWITCH_SELECT)) {
    Serial.println("SELECT pressed!");
  }
  if (! ss.digitalRead(SS_SWITCH_LEFT)) {
    Serial.println("LEFT pressed!");
  }
  if (! ss.digitalRead(SS_SWITCH_RIGHT)) {
    Serial.println("RIGHT pressed!");
  }

  int32_t new_position = ss.getEncoderPosition();
  // did we move around?
  if (encoder_position != new_position) {
    Serial.println(new_position);         // display new position

    encoder_position = new_position;      // and save for next round
  }

  // don't overwhelm serial port
  delay(10);
}
```

Upload the sketch to your board and open up the Serial Monitor (Tools -> Serial Monitor) at 115200 baud. You'll see the seesaw firmware recognized by the code. Then, when you press any of the buttons or turn the encoder it will print to the Serial Monitor. You'll also see the interrupt LED light up with each encoder turn.



```
Looking for seesaw!
seesaw started
Found Product 5740
Turning on interrupts
SELECT pressed!
SELECT pressed!
SELECT pressed!
SELECT pressed!
SELECT pressed!
SELECT pressed!
SELECT pressed!
SELECT pressed!
-1
-2
-3
-4
-5
-6
-7
-8
-9
-10
```

Arduino Docs

Arduino Docs ()

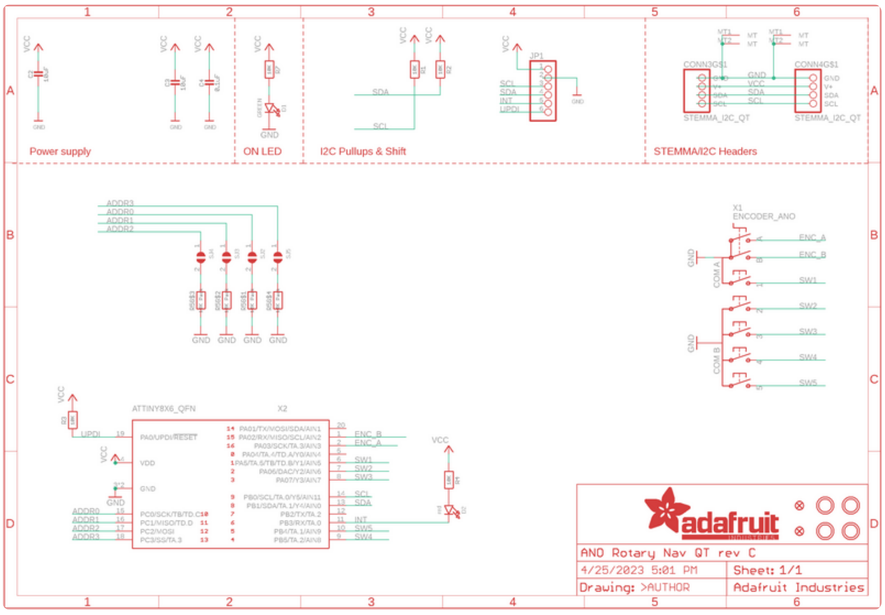
Downloads

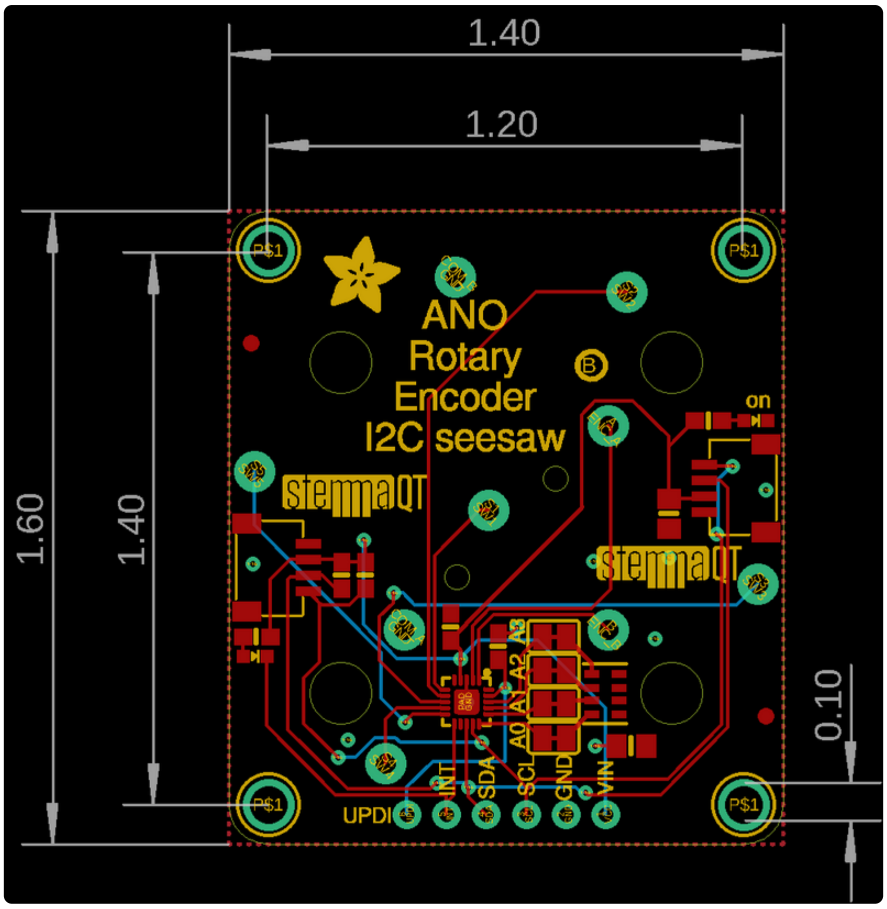
Files

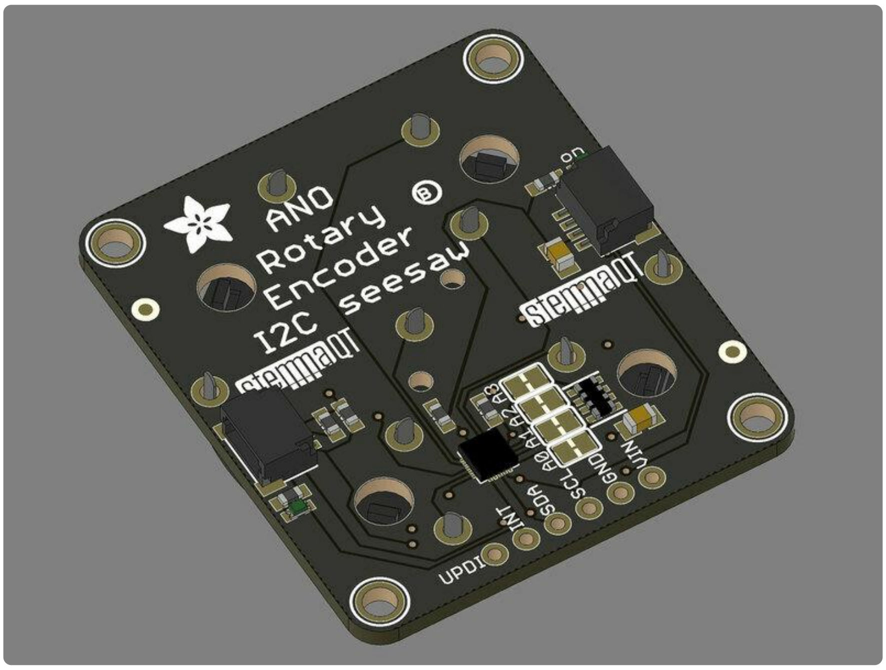
- ATtiny816 Datasheet ()
- ANO Rotary Encoder Datasheet ()
- EagleCAD PCB files on GitHub ()
- 3D models on GitHub ()
- Fritzing object in the Adafruit Fritzing Library - back view, no encoder ()
- Fritzing object in the Adafruit Fritzing Library - front view, with encoder ()

Schematic and Fab Print

Dimensions in inches.




1.40
1.20
1.60
1.40
0.10
ANO Rotary Encoder I2C seesaw
stemmaQT
on
UPDI
INT
SDA
SCL
GND
VIN
A0
A1
A2
A3


ANO Rotary Encoder I2C seesaw
stemmaQT
UPDI
INT
SDA
SCL
GND
VIN
A0
A1
A2
A3