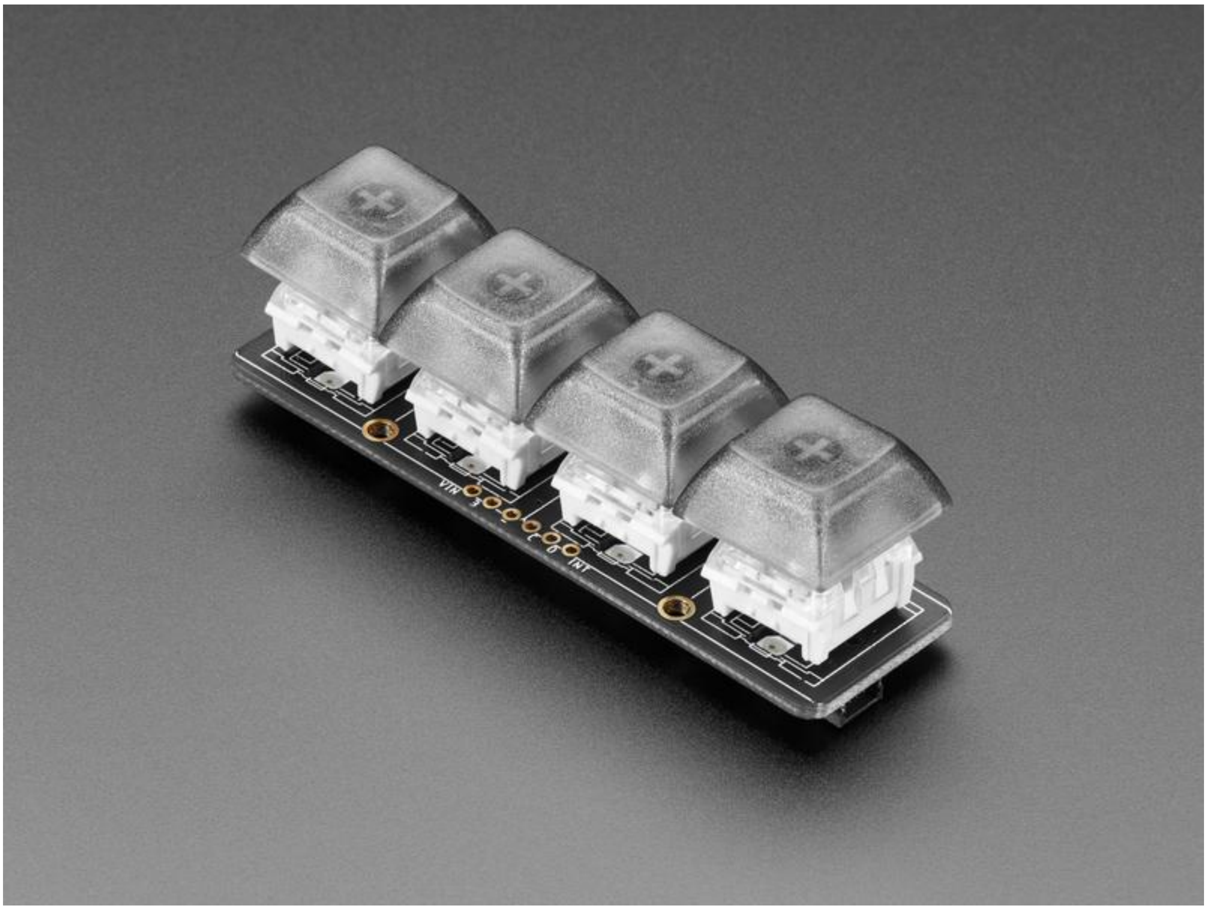
Adafruit NeoKey 1x4 QT I2C Breakout

Created by Kattni Rembor

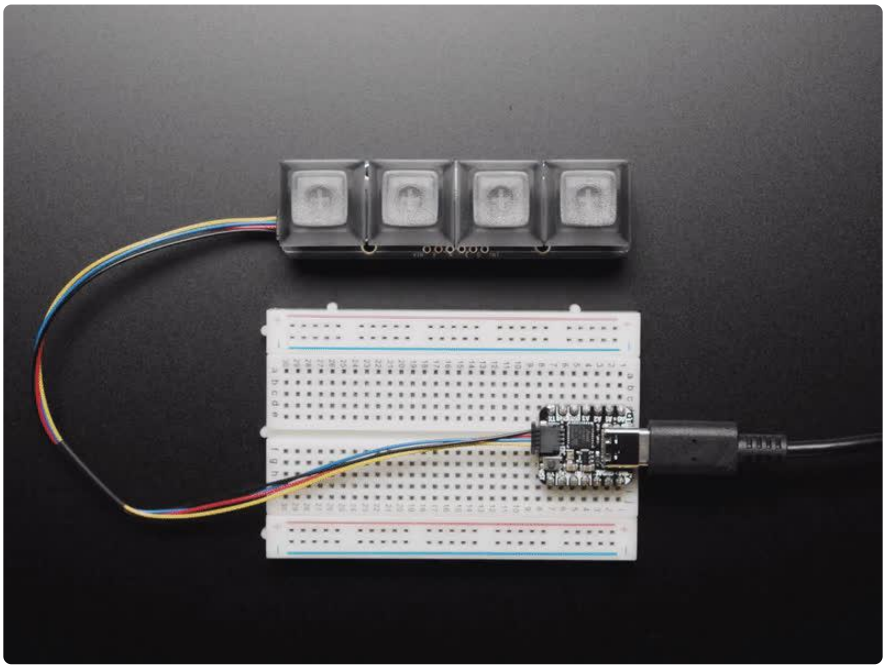


https://learn.adafruit.com/neokey-1x4-qt-i2c

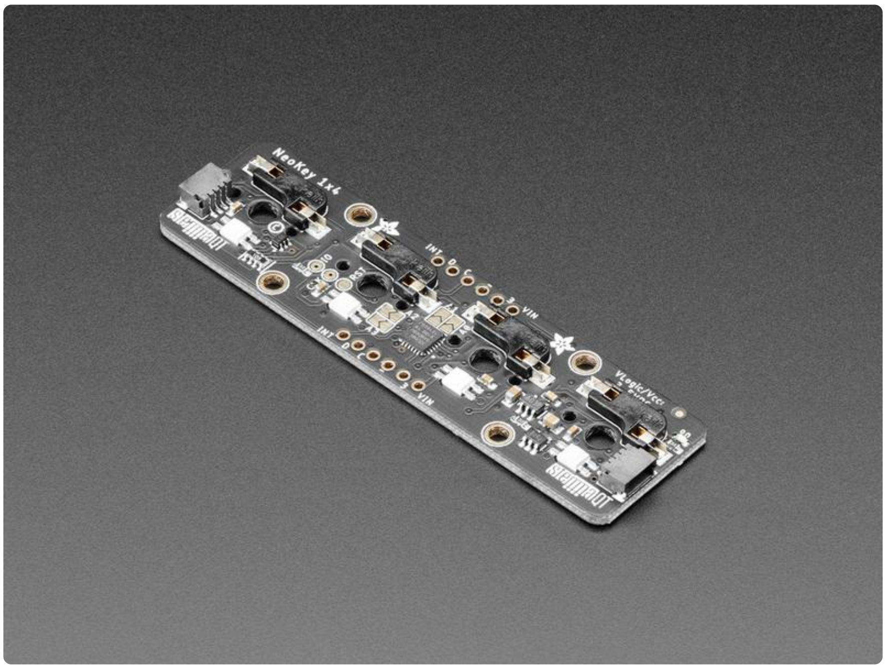
Last updated on 2021-11-15 08:25:35 PM EST

Table of Contents

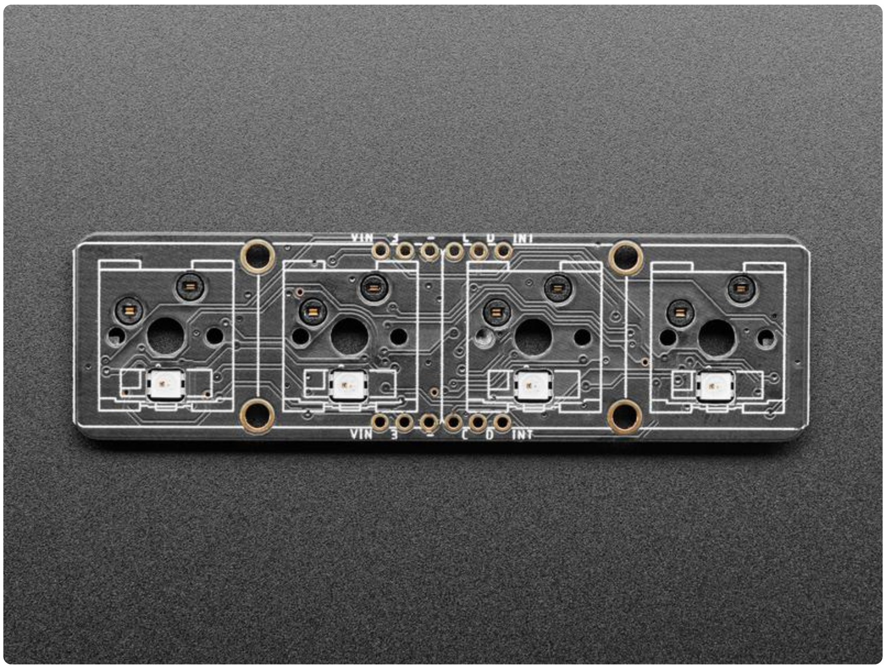
Overview



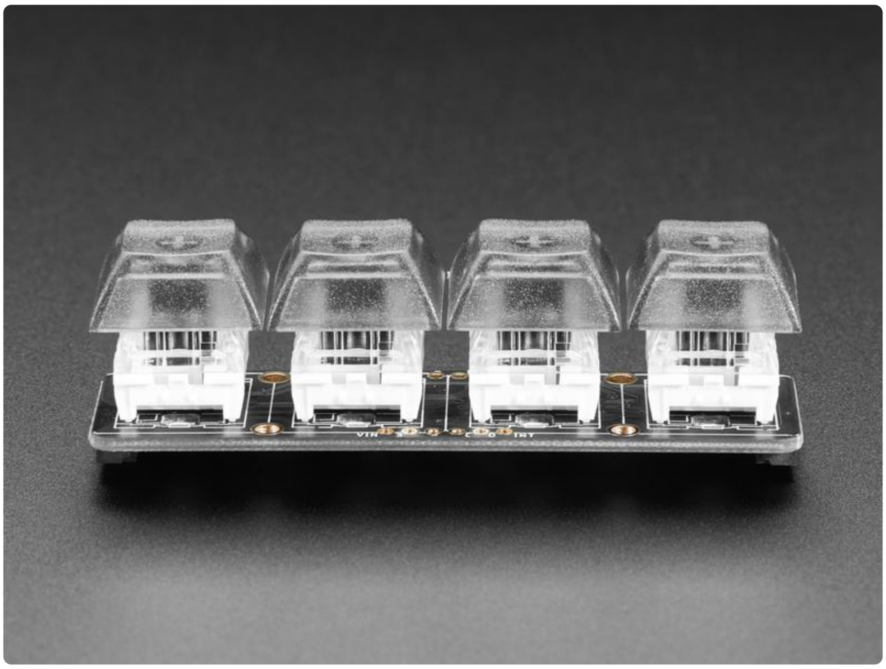
The only thing better than a nice mechanical key is, perhaps, FOUR mechanical keys that also can glow any color of the rainbow - and that's what the Adafruit NeoKey 1x4 QT I2C Breakout will let you do! This long 3" x 0.8" PCB fits four Cherry MX or compatible switches and make it easy to use with a breadboard/perfboard or with a STEMMA QT (Qwiic) connector for instant I2C connectivity on any platform.



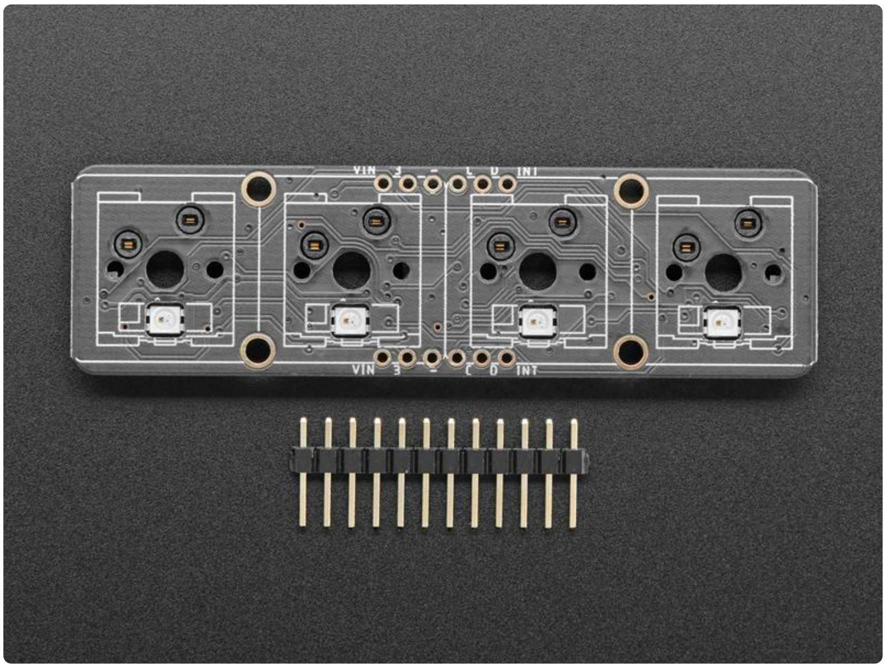
Please note, each order comes with one assembled and programmed PCB but no switches or keycaps or microcontroller. Most folks have specific key switches and keycaps they want to include, this is just the controller board that plugs into a microcontroller.



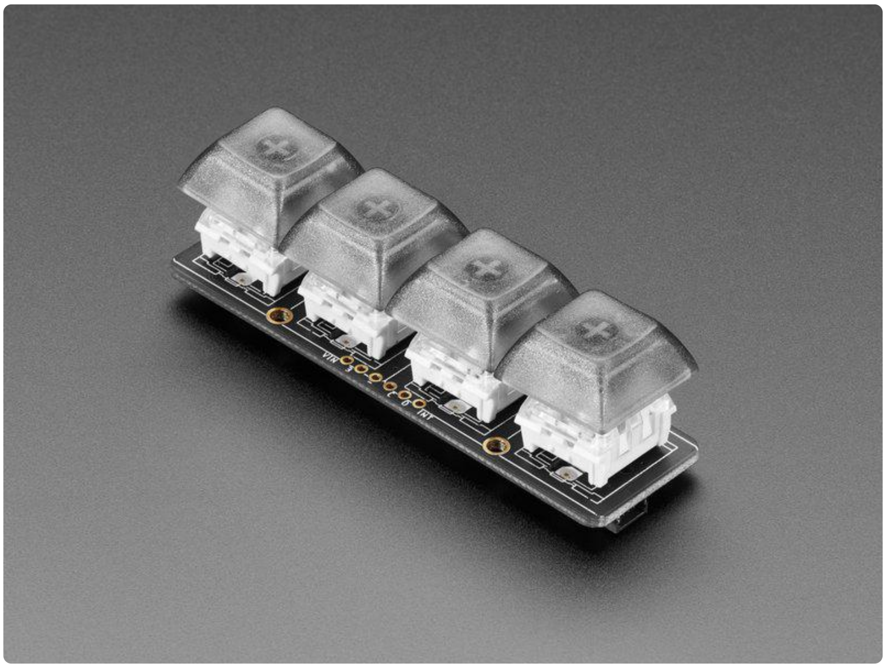
The breakout has four Kailh sockets, which means you can plug in any MX-compatible switch instead of soldering it in. You may need a little glue to keep the switch in place: hot glue or a dot of epoxy worked fine for us. Each key also has a reverse-mount NeoPixel (https://adafru.it/Txa) pointing up through the spot where many switches would have an LED to shine through.



A microcontroller is pre-programmed with our Seesaw firmware so button presses and NeoPixel-controlling is done all over I2C. You can even connect multiple board by chaining the I2C and soldering closed the I2C address jumpers - with four jumpers you can have up to 16 of these boards on a single I2C bus. We have Arduino (https://adafru.it/BrV) and CircuitPython/Python libraries (https://adafru.it/BrW) for controlling the NeoKey 1x4's so you can use any microcontroller/computer for quick creation of a custom macropad.

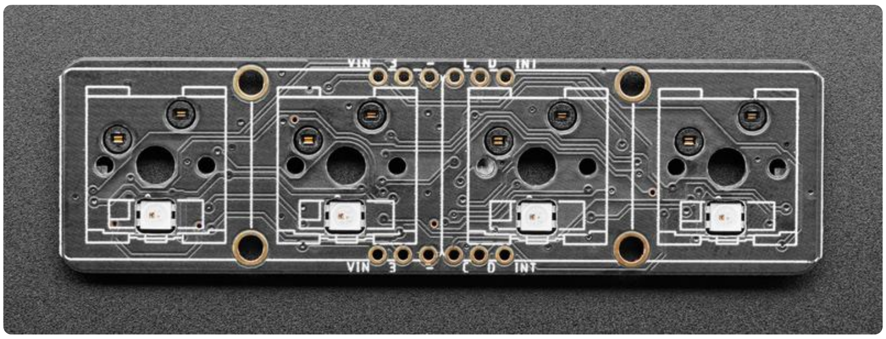


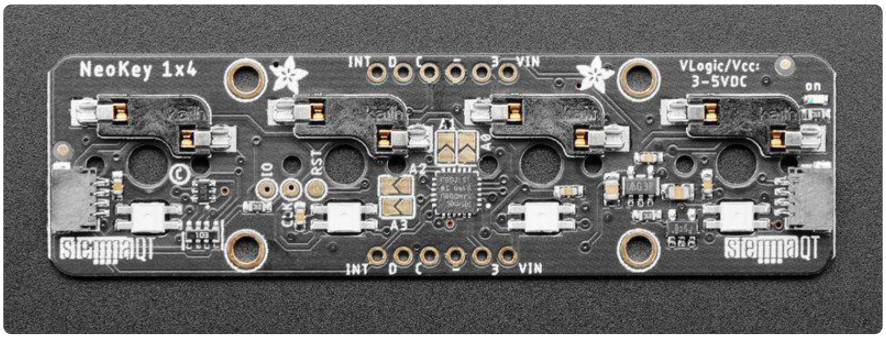
You can also fit the breakouts onto a breadboard if you like - with two sets of breakout pads, there's plenty of flexibility for any kind of use. There are two rows of 6-pin contacts on a 0.1" grid on both sides. Solder in both sides for mechanical stability.



Soldering is required to attach the header for breadboard use, you may also need to solder jumpers closed to connect multiple boards together. A microcontroller is required to drive this board, it isn't stand-alone. Keys & keycaps are not included: Use any MX-compatible switch: Kailh, Gateron, etc all work!

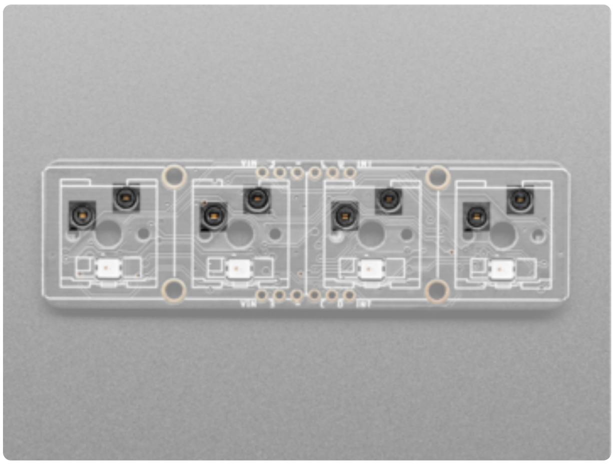
Pinouts

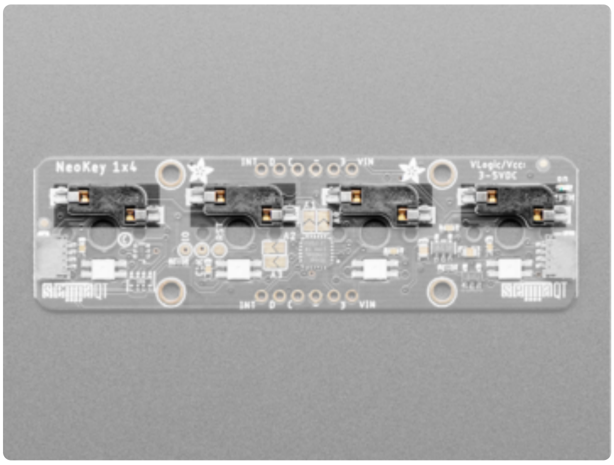




The NeoKey 1x4 QT I2C breakout is super simple to use but has a lot going on! This page covers all of the pins on and features of the board. Time for a tour!

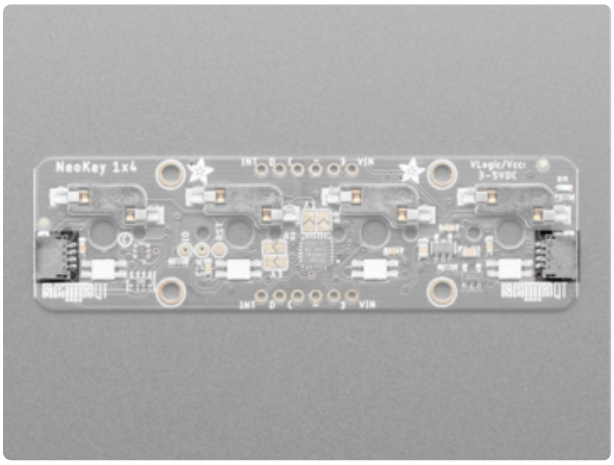
Key Sockets





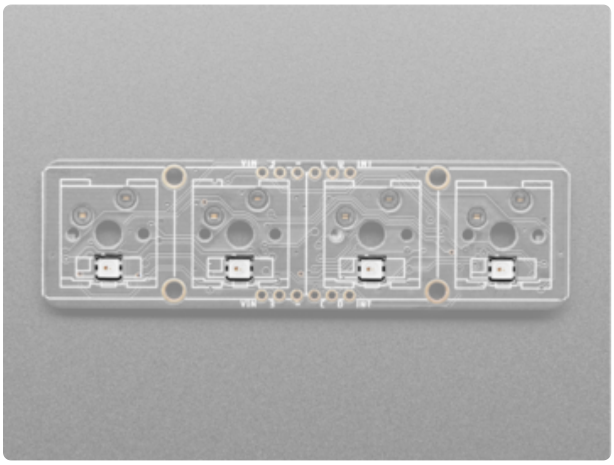
There are four Cherry MX or compatible key switch sockets, distributed evenly across the board. Simply press any compatible key switch into the socket from the top of the board. You can add a dab of glue to keep the switch in place; hot glue or a dot of epoxy will work.

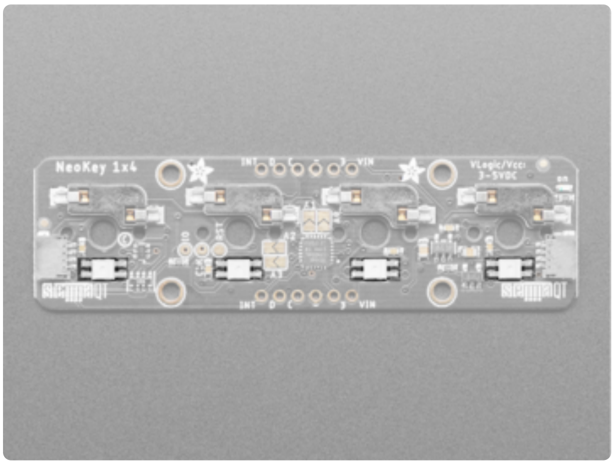
STEMMA QT Connectors



On the bottom of the board, on both ends, are STEMMA QT (https://adafru.it/Ft4) connectors. These connectors allow you to connect to development boards with STEMMA QT connectors or to other devices with various associated accessories (https://adafru.it/Ft6).

NeoPixel LEDs

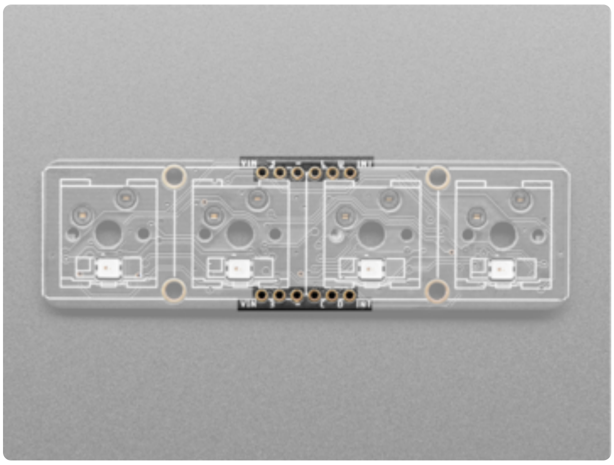


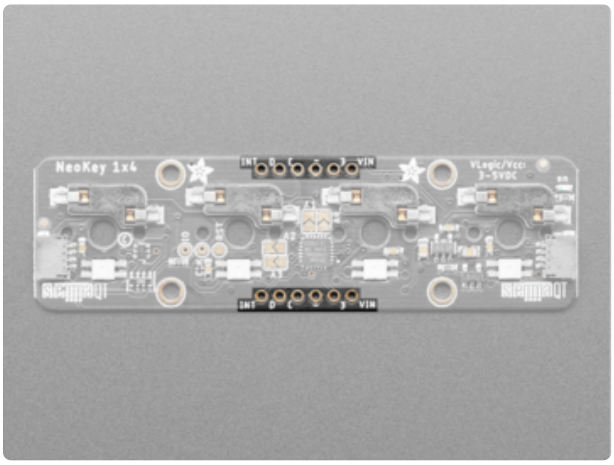


Along the edge of the board, below the key sockets, are four NeoPixel LEDs. These LEDs are reverse mount, (https://adafru.it/Txa) meaning they're mounted on the back of the board to shine through to the top. This allows for the key switches to sit flat while still providing rainbow goodness!

The NeoPixels are controlled over I2C using the seesaw library. They are connected to Seesaw pin 3.

Header Pins



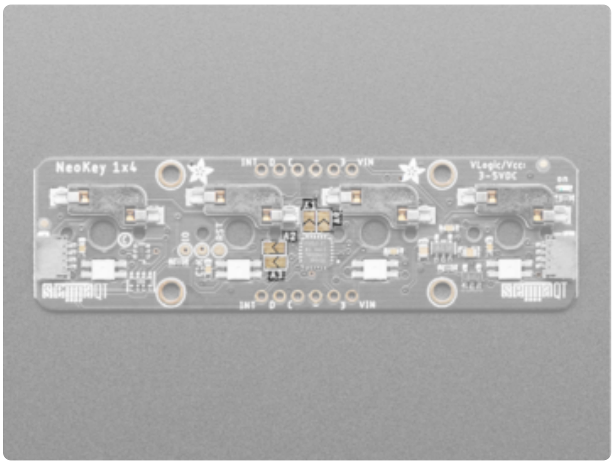


Along both long edges, in the middle of the board, are a set of through-hole headers. Solder the provided header pins to both sets of through-holes for stability in a breadboard/perfboard setup.

When viewed from the top of the board they are:

- VIN - This is the power pin. Since the chip uses 3 VDC, we have included a voltage regulator on board that will take 3-5VDC and safely convert it down. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 5V microcontroller like Arduino, use 5V.
- 3 - This is the 3.3V output from the voltage regulator. You can grab up to 100mA from this if you like.
- - - This is the ground pin. It is the common ground for power and logic.
- C - This is the I2C clock pin. Connect to your microcontroller I2C clock line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.
- D - This is the I2C data pin. Connect to your microcontroller I2C data line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.
- INT - This is the interrupt pin. The Seesaw allows for setting GPIO interrupts. This pin will go low when a button is pressed.

Address Jumpers



On the back of the board are four address jumpers. 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 solder the jumpers "closed" by connecting the two pads.

The default I2C address is 0x30. The other address options can be calculated by “adding” the A0/A1/A2/A3 to the base of 0x30.

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 high bit with a value of 8. The final address is 0x30 + A3 + A2 + A1 + A0 which would be 0x3F.

So for example if A2 is soldered closed and A0 is soldered closed, the address is 0x 30 + 4 + 1 = 0x35.

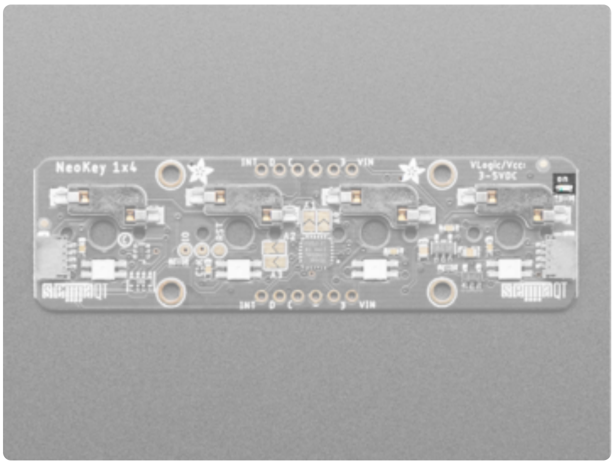
If only A0 is soldered closed, the address is 0x30 + 1 = 0x31

If only A1 is soldered closed, the address is 0x30 + 2 = 0x32

If only A2 is soldered closed, the address is 0x30 + 4 = 0x34

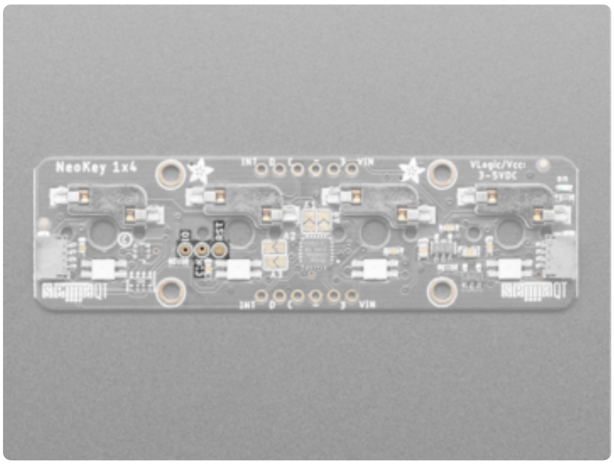
If only A3 is soldered closed, the address is 0x30 + 8 = 0x38

Power LED



In the top-right corner of the board, when viewed from the back, is a green on LED that is lit up when the board is powered.

Debug Pins



On the back of the board are three, round debug pads. They are, from left to right, IO, clock, and reset.

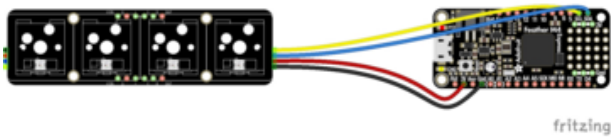
Python & CircuitPython

It's easy to use the NeoKey 1x4 with CircuitPython using the Adafruit CircuitPython NeoKey (https://adafru.it/TxC) library. It allows you to write Python code to read the key presses and control the NeoPixel LEDs.

You can use this sensor with any CircuitPython microcontroller board or with a computer that has GPIO and Python thanks to Adafruit_Blinka, our CircuitPython-for-Python compatibility library (https://adafru.it/BSN).

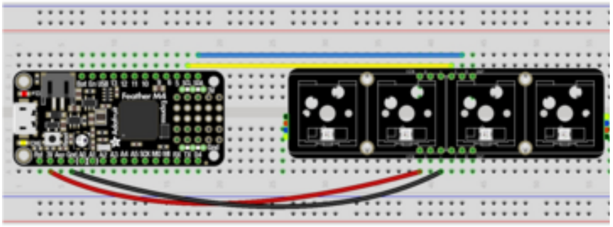
CircuitPython Microcontroller Wiring

First wire up a NeoKey 1x4 breakout to your board exactly as follows. The following is the breakout wired to a Feather using the STEMMA connector:



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

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

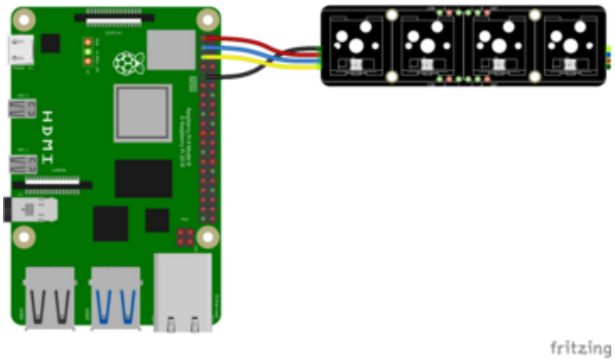


- Board 3V to breakout VIN (red wire)
- Board GND to breakout GND (black wire)
- Board SCL to breakout SCL (yellow wire)
- Board SDA to breakout SDA (blue wire)

Python Computer Wiring

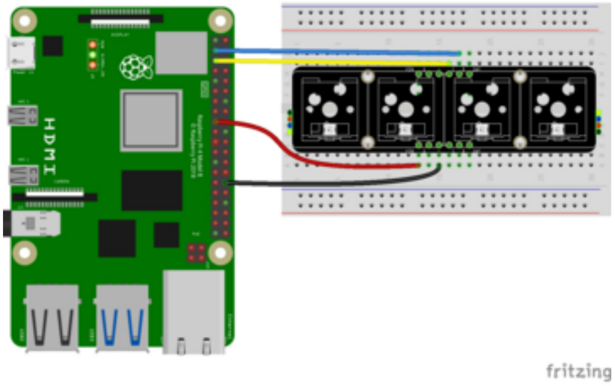
Since there's dozens of Linux computers/boards you can use we will show wiring for Raspberry Pi. For other platforms, please visit the guide for CircuitPython on Linux to see whether your platform is supported (https://adafru.it/BSN).

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



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

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



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

Python Installation of NeoKey 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 (https://adafru.it/BSN)!

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

- `pip3 install adafruit-circuitpython-neokey`

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 & Python Usage

To demonstrate using this breakout with CircuitPython, you'll install the necessary libraries, update your code, and then connect to the serial console (https://adafru.it/Bec) to see the information printed out.

To use the NeoKey 1x4 breakout with CircuitPython, you need to first install the NeoKey library, and its dependencies, into the lib folder on your CIRCUITPY drive.

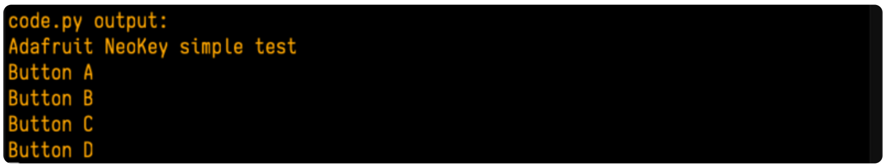
Then you need to update code.py.

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 code.py file to your CIRCUITPY drive.

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT
"""NeoKey simpletest."""
import board
from adafruit_neokey.neokey1x4 import NeoKey1x4

# use default I2C bus
i2c_bus = board.I2C()

# Create a NeoKey object
neokey = NeoKey1x4(i2c_bus, addr=0x30)

print("Adafruit NeoKey simple test")

# Check each button, if pressed, light up the matching neopixel!
while True:
    if neokey[0]:
        print("Button A")
        neokey.pixels[0] = 0xFF0000
    else:
        neokey.pixels[0] = 0x0

    if neokey[1]:
        print("Button B")
        neokey.pixels[1] = 0xFFFF00
    else:
        neokey.pixels[1] = 0x0

    if neokey[2]:
        print("Button C")
        neokey.pixels[2] = 0x00FF00
    else:
        neokey.pixels[2] = 0x0

    if neokey[3]:
        print("Button D")
        neokey.pixels[3] = 0x00FFFF
    else:
        neokey.pixels[3] = 0x0
```

Now press the buttons to see the associated NeoPixel LED light up!

Connect to the serial console to see the messages printed out.



```
code.py output:
Adafruit NeoKey simple test
Button A
Button B
Button C
Button D
```

That's all there is to using the NeoKey 1x4 breakout with CircuitPython!

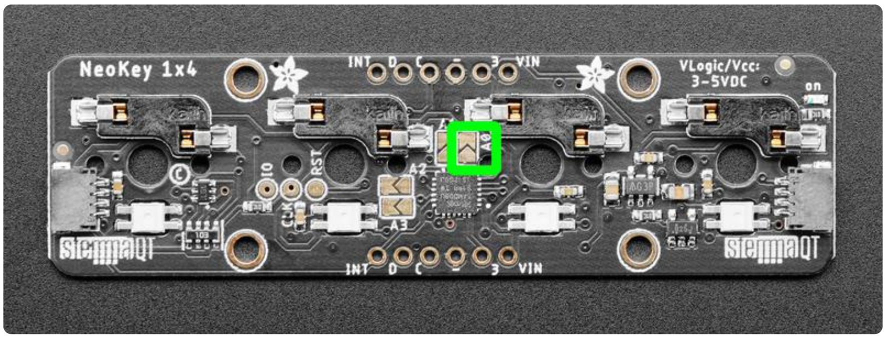
Multi-NeoKey 1x4 Usage

The address jumpers on the back of the NeoKey 1x4 breakout enable you to chain together up to 16 of these breakouts on a single I2C bus. This example shows how to connect two and use them together.

This example requires minimal soldering.

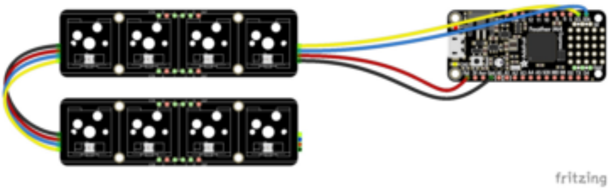
Address Jumper Soldering

Use solder to bridge the A0 jumper (highlighted in green below) on the back of the board.



NeoKey Wiring

Use a STEMMA QT cable to connect a second NeoKey 1x4 breakout to your current wiring setup. The example below is a Feather M4, but it will work the same on any CircuitPython-compatible board or Python computer.



- Board one 3V to breakout VIN (red wire)
- Board one GND to breakout GND (black wire)
- Board one SCL to breakout SCL (yellow wire)
- Board one SDA to breakout SDA (blue wire)
- Connect board one STEMMA QT to board two STEMMA QT using a STEMMA QT cable such as this (https://adafru.it/Tff) or this (https://adafru.it/FNS).

CircuitPython and Python Multi-NeoKey 1x4 Usage

To demonstrate using this breakout with CircuitPython, you'll install the necessary libraries, update your code, and then connect to the serial console (https://adafru.it/Bec) to see the information printed out.

To use two NeoKey 1x4 breakouts with CircuitPython, you need to first install the NeoKey library, and its dependencies, into the lib folder on your CIRCUITPY drive.

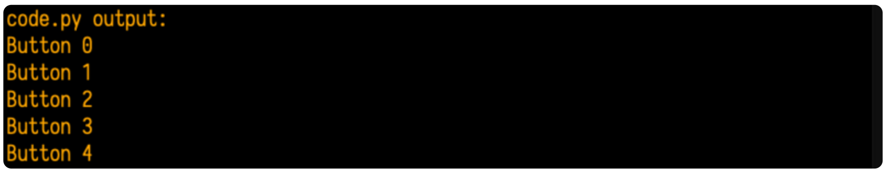
Then you need to update code.py.

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 entir e lib folder and the code.py file to your CIRCUITPY drive.

```
# SPDX-FileCopyrightText: 2021 Kattni Rembor for Adafruit Industries
# SPDX-License-Identifier: MIT
"""Example for connecting two NeoKey 1x4 breakouts. Requires bridging the A0 jumper
on one board."""
import board
from rainbowio import colorwheel
from adafruit_neokey.neokey1x4 import NeoKey1x4

# Create a NeoKey object
neokey1 = NeoKey1x4(board.I2C())
neokey2 = NeoKey1x4(board.I2C(), addr=0x31)

keys = [
    (neokey1, 0, colorwheel(0)),
    (neokey1, 1, colorwheel(32)),
    (neokey1, 2, colorwheel(64)),
    (neokey1, 3, colorwheel(96)),
    (neokey2, 0, colorwheel(128)),
    (neokey2, 1, colorwheel(160)),
    (neokey2, 2, colorwheel(192)),
    (neokey2, 3, colorwheel(224)),
]

off = (0, 0, 0)

# Check each button, if pressed, light up the matching NeoPixel!
while True:
    for i in range(8):
        neokey, key_number, color = keys[i]
        if neokey[key_number]:
            print("Button", i)
            neokey.pixels[key_number] = color
        else:
            neokey.pixels[key_number] = off
```

Now press the buttons to see the associated NeoPixel LED light up!

Connect to the serial console to see the messages printed out.



```
code.py output:
Button 0
Button 1
Button 2
Button 3
Button 4
```

That's all there is to using two NeoKey 1x4 breakouts with CircuitPython!

Python Docs

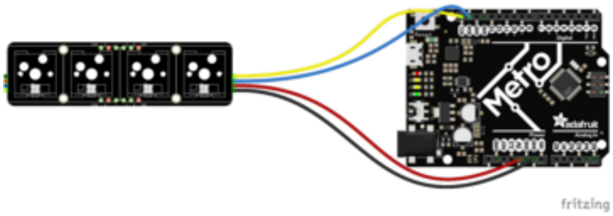
Python Docs (https://adafru.it/Tud)

Arduino

The Adafruit NeoKey 1x4 QT I2C uses the Seesaw chip. To use the NeoKey 1x4 with Arduino, you'll use the Adafruit Seesaw library. With the key sockets and the STEMMA QT connectors, you can easily get started with no soldering necessary!

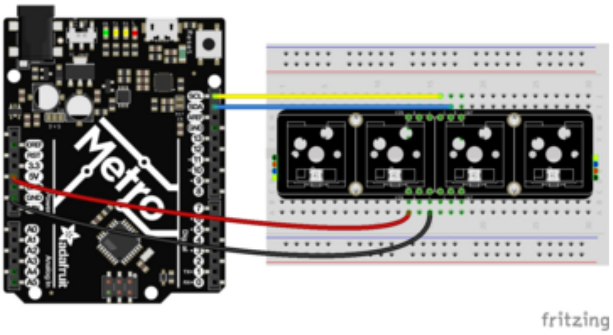
I2C Wiring

Here is how to wire up the breakout using one of the STEMMA QT (https://adafru.it/Ft4) connectors. The examples show a Metro but wiring will work the same for an Arduino or other compatible board.



- Connect board VIN (red wire) to Arduino 5V if you are running a 5V board Arduino (Uno, etc.). If your board is 3V, connect to that instead.
- Connect board GND (black wire) to Arduino GND
- Connect board SCL (yellow wire) to Arduino SCL
- Connect board SDA (blue wire) to Arduino SDA

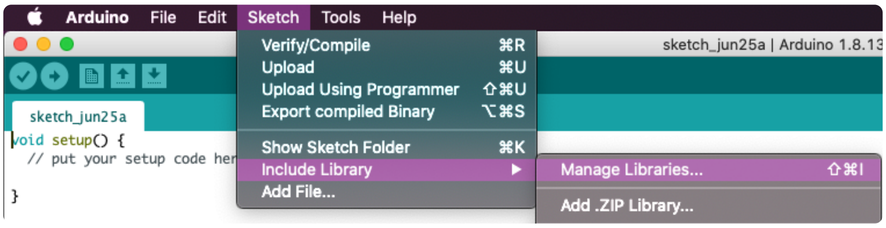
Here is how to wire the breakout to a board using a solderless breadboard. To do this, you must solder the header pins provided to the breakout.



- Connect board VIN (red wire) to Arduino 5V if you are running a 5V board Arduino (Uno, etc.). If your board is 3V, connect to that instead.
- Connect board GND (black wire) to Arduino GND
- Connect board SCL (yellow wire) to Arduino SCL
- Connect board SDA (blue wire) to Arduino SDA

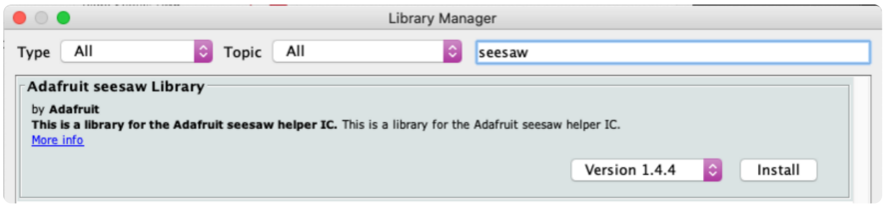
Library Installation

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 seesaw , and select the Adafruit s eesaw library:


Library Manager

Type: All Topic: All seesaw

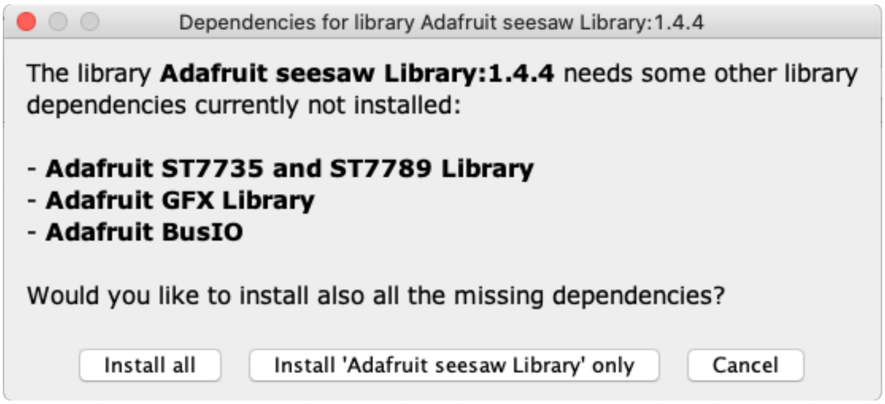
Adafruit seesaw Library

by **Adafruit**

This is a library for the Adafruit seesaw helper IC. This is a library for the Adafruit seesaw helper IC.

More info

When asked to install the Adafruit Seesaw library dependencies, click Install all.


Dependencies for library Adafruit seesaw Library:1.4.4

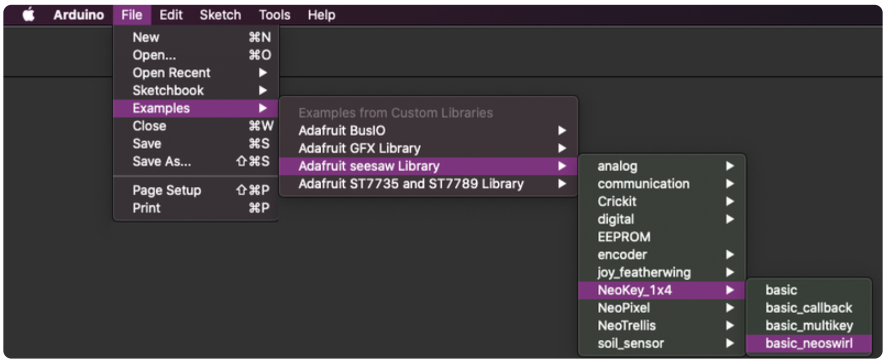
The library **Adafruit seesaw Library:1.4.4** needs some other library dependencies currently not installed:

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

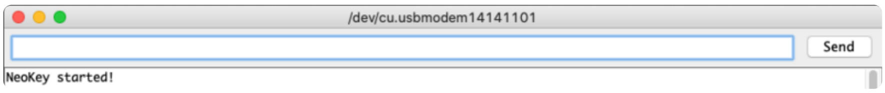
Would you like to install also all the missing dependencies?

Load Example

Open up File -> Examples -> Adafruit seesaw Library -> NeoKey_1x4 -> basic_neoswirl

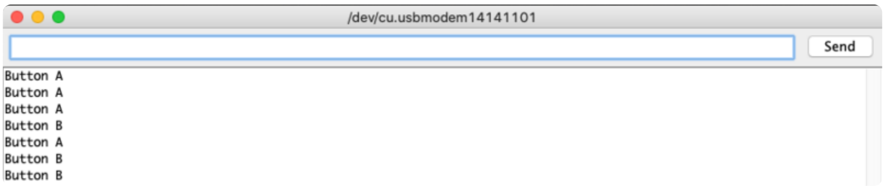


After opening the basic_neoswirl file, upload it to the Arduino wired to your NeoKey 1x4. Open the Serial Monitor at 115200 baud. You should see the following as the sketch starts up.



```
NeoKey started!
```

Now try pressing the keys to see the corresponding NeoPixel light up, and a message printed to the serial output!



```
Button A
Button A
Button A
Button B
Button A
Button B
Button B
```

That's all there is to using the NeoKey 1x4 breakout with Arduino!

Example Code

```
#include "Adafruit_NeoKey_1x4.h"
#include "seesaw_neopixel.h"

Adafruit_NeoKey_1x4 neokey;

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

  if (! neokey.begin(0x30)) {
    Serial.println("Could not start NeoKey, check wiring?");
    while(1) delay(10);
  }

  Serial.println("NeoKey started!");

  for (uint16_t i=0; i<neokey.pixels.numPixels(); i++) {
    neokey.pixels.setPixelColor(i, Wheel(map(i, 0, neokey.pixels.numPixels(), 0, 
255)));
    neokey.pixels.show();
    delay(50);
  }
  for (uint16_t i=0; i<neokey.pixels.numPixels(); i++) {
    neokey.pixels.setPixelColor(i, 0x000000);
    neokey.pixels.show();
    delay(50);
  }
}

uint8_t j=0;  // this variable tracks the colors of the LEDs cycle.

void loop() {
  uint8_t buttons = neokey.read();


  for (int i=0; i< neokey.pixels.numPixels(); i++) {
    neokey.pixels.setPixelColor(i, Wheel(((i * 256 / neokey.pixels.numPixels()) + j) 
& 255));
  }

  if (buttons & (1<<0)) {
    Serial.println("Button A");
  } else {
    neokey.pixels.setPixelColor(0, 0);
  }

  if (buttons & (1<<1)) {
    Serial.println("Button B");
  } else {
    neokey.pixels.setPixelColor(1, 0);
  }

  if (buttons & (1<<2)) {
    Serial.println("Button C");
  } else {
    neokey.pixels.setPixelColor(2, 0);
  }

  if (buttons & (1<<3)) {
    Serial.println("Button D");
  } else {
    neokey.pixels.setPixelColor(3, 0);
  }

  neokey.pixels.show();

  delay(10);    // don't print too fast
  j++;          // make colors cycle
}



/******************************************/

// Input a value 0 to 255 to get a color value.
// The colors are a transition r - g - b - back to r.
uint32_t Wheel(byte WheelPos) {
  if(WheelPos < 85) {
   return seesaw_NeoPixel::Color(WheelPos * 3, 255 - WheelPos * 3, 0);
  } else if(WheelPos < 170) {
   WheelPos -= 85;
   return seesaw_NeoPixel::Color(255 - WheelPos * 3, 0, WheelPos * 3);
  } else {
   WheelPos -= 170;
   return seesaw_NeoPixel::Color(0, WheelPos * 3, 255 - WheelPos * 3);
  }
  return 0;
}
```

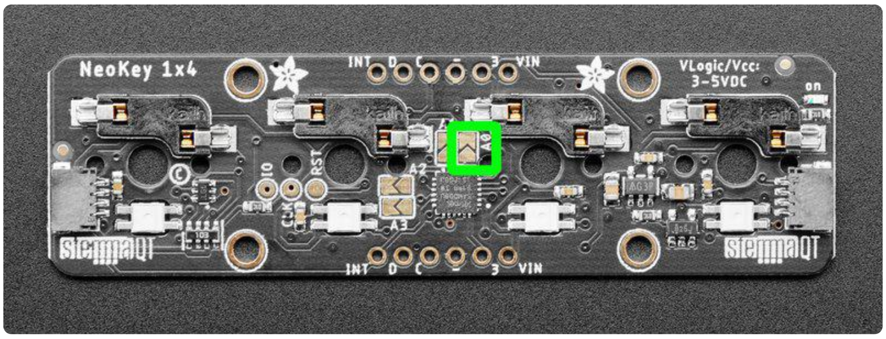
Using Multiple NeoKey 1x4 Breakouts

The address jumpers on the back of the NeoKey 1x4 breakout enable you to chain together up to 16 of these breakouts on a single I2C bus. This example shows how to connect two and use them together.

This example requires minimal soldering.

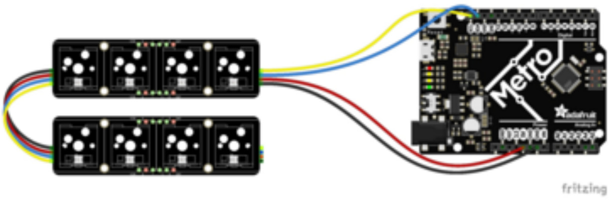
Address Jumper Soldering

Use solder to bridge the A0 jumper (highlighted in green below) on the back of the board.



I2C Wiring

Here is how to wire up two breakouts using the STEMMA QT (https://adafru.it/Ft4) connectors. The examples show a Metro but wiring will work the same for an Arduino or other compatible board.



- Connect board one VIN (red wire) to Arduino 5V if you are running a 5V board Arduino (Uno, etc.). If your board is 3V, connect to that instead.
- Connect board one GND (black wire) to Arduino GND
- Connect board one SCL (yellow wire) to Arduino SCL
- Connect board one SDA (blue wire) to Arduino SDA
- Connect board one STEMMA QT to board two STEMMA QT using a STEMMA QT cable such as this (https://adafru.it/Tff) or this (https://adafru.it/FNS).

Library Installation

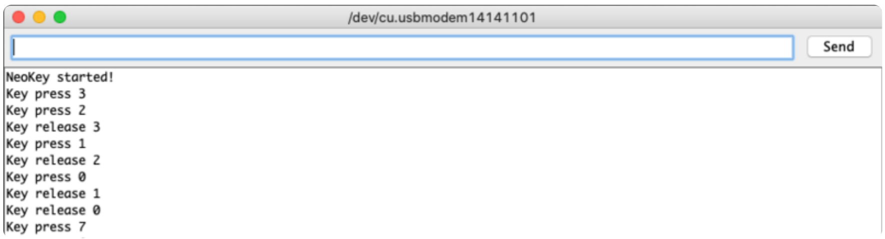
The library installation is the same as above. If you have not already, follow the library installation instructions above before continuing.

Load Multi-Key Example

Open up File -> Examples -> Adafruit seesaw Library -> NeoKey_1x4 -> basic_multikey.

After opening the basic_multikey file, upload it to the Arduino wired to your pair of NeoKey 1x4 breakouts. Open the Serial Monitor at 115200 baud.

Try pressing the keys to see the corresponding NeoPixel LEDs light up, and messages printed to the serial monitor on key press and release.



```
NeoKey started!
Key press 3
Key press 2
Key release 3
Key press 1
Key release 2
Key press 0
Key release 1
Key release 0
Key press 7
```

Multi-Key Example Code

```
#include "Adafruit_NeoKey_1x4.h"
#include "seesaw_neopixel.h"

#define Y_DIM 2 //number of rows of keys
#define X_DIM 4 //number of columns of keys

// create a matrix of NeoKey 1x4's
// this example is just two, one on top of another to make a 2x4 grid
Adafruit_NeoKey_1x4 nk_array[Y_DIM][X_DIM/4] = {
  { Adafruit_NeoKey_1x4(0x30) },
  { Adafruit_NeoKey_1x4(0x31) },
};

// pass this matrix to the multi-neokey object
Adafruit_MultiNeoKey1x4 neokey((Adafruit_NeoKey_1x4 *)nk_array, Y_DIM, X_DIM/4);


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

  if (! neokey.begin()) {  // start matrix
    Serial.println("Could not start NeoKeys, check wiring?");
    while(1) delay(10);
  }

  Serial.println("NeoKey started!");

  // Pulse all the LEDs on to show we're working
  for (uint16_t i=0; i< X_DIM*Y_DIM; i++) {
    neokey.setPixelColor(i, 0x808080); // make each LED white
    neokey.show();
    delay(50);
  }
  for (uint16_t i=0; i< X_DIM*Y_DIM; i++) {
    neokey.setPixelColor(i, 0x000000);
    neokey.show();
    delay(50);
  }

  // activate all keys and set callbacks
  for(int y=0; y<Y_DIM; y++){
    for(int x=0; x<X_DIM; x++){
      neokey.registerCallback(x, y, blink);
    }
  }
}

void loop() {
  neokey.read();

  delay(10);    // don't print too fast
}


//define a callback for key presses
NeoKey1x4Callback blink(keyEvent evt) {
  uint8_t key = evt.bit.NUM;

  if (evt.bit.EDGE == SEESAW_KEYPAD_EDGE_RISING) {
    Serial.print("Key press ");
    Serial.println(key);
    neokey.setPixelColor(key, Wheel(map(key, 0, X_DIM*Y_DIM, 0, 255)));

  } else if (evt.bit.EDGE == SEESAW_KEYPAD_EDGE_FALLING) {
    Serial.print("Key release ");
    Serial.println(key);

    neokey.setPixelColor(key, 0);
  }

  // Turn on/off the neopixels!
  neokey.show();
  return 0;
}

/******************************************/

// Input a value 0 to 255 to get a color value.
// The colors are a transition r - g - b - back to r.
uint32_t Wheel(byte WheelPos) {
  if(WheelPos < 85) {
   return seesaw_NeoPixel::Color(WheelPos * 3, 255 - WheelPos * 3, 0);
  } else if(WheelPos < 170) {
   WheelPos -= 85;
   return seesaw_NeoPixel::Color(255 - WheelPos * 3, 0, WheelPos * 3);
  } else {
   WheelPos -= 170;
   return seesaw_NeoPixel::Color(0, WheelPos * 3, 255 - WheelPos * 3);
  }
  return 0;
}
```

Arduino Docs

Arduino Docs (https://adafru.it/TuA)

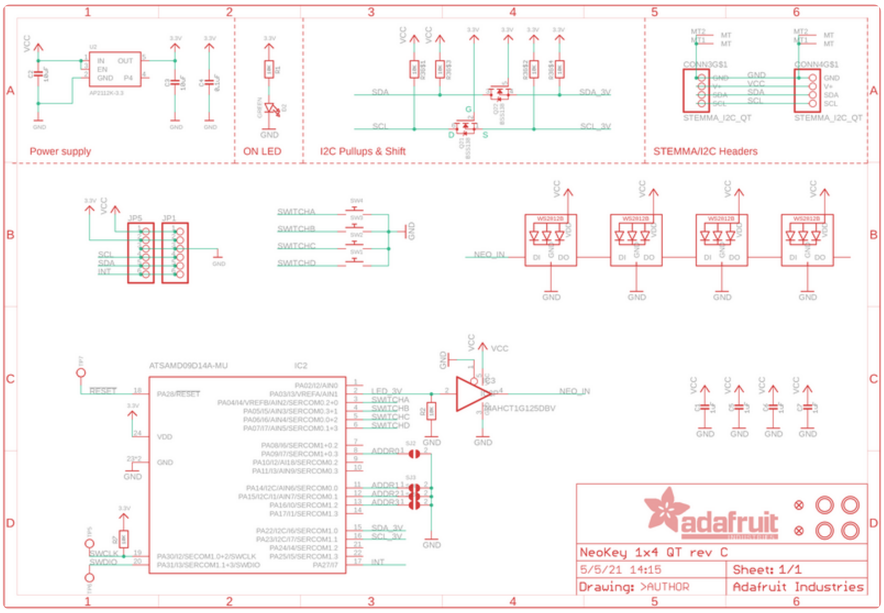
Downloads

Files

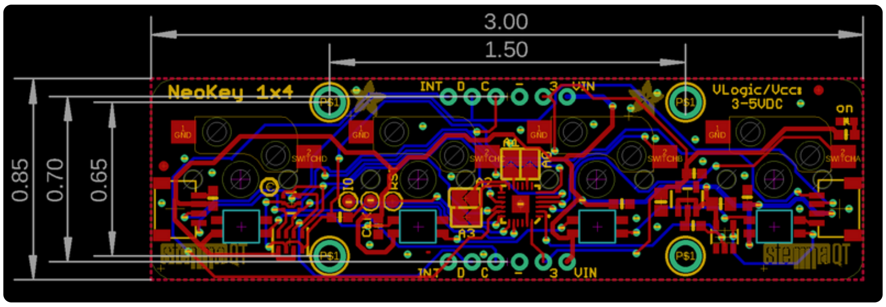
- SAMD09 datasheet (https://adafru.it/BrY)
- EagleCAD PCB files on GitHub (https://adafru.it/Txb)

- [3D Models on GitHub](https://adafru.it/TAn) (https://adafru.it/TAn)
- [Fritzing object in the Adafruit Fritzing Library](https://adafru.it/Txc) (https://adafru.it/Txc)

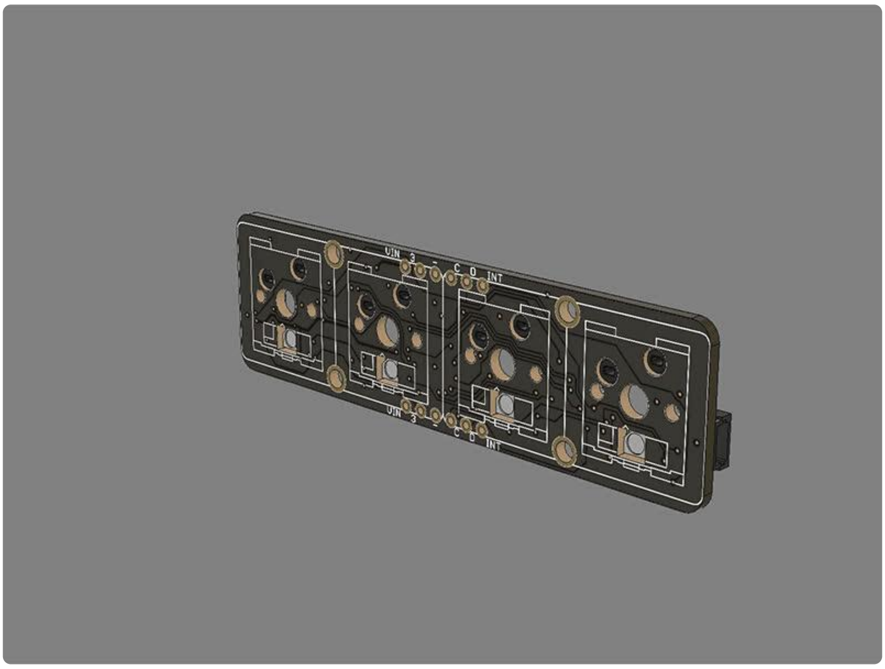
Schematic



Fab Print



3D Model


VIN 3 - C D INT
VIN 3 - C D INT