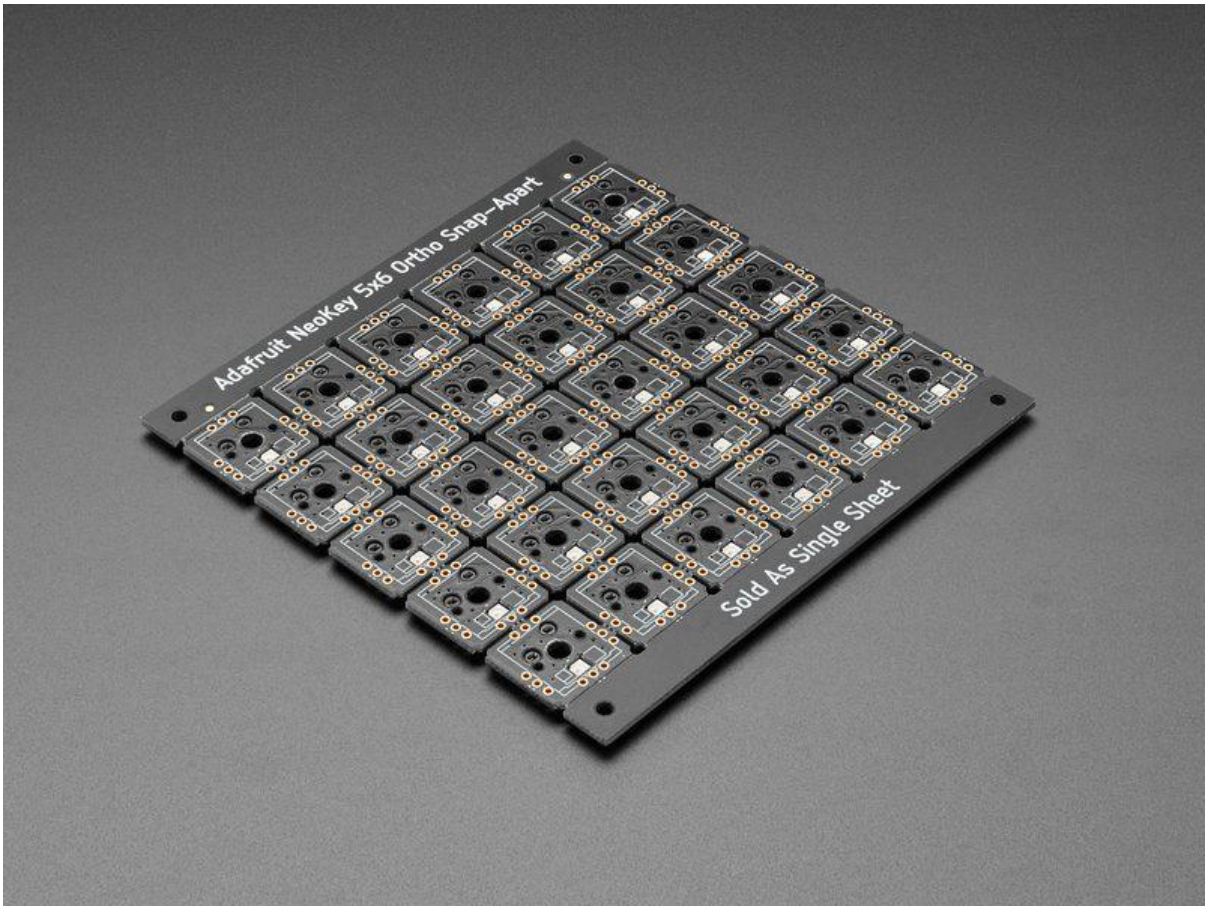
Adafruit NeoKey 5x6 Ortho Snap-Apart

Created by Kattni Rembor

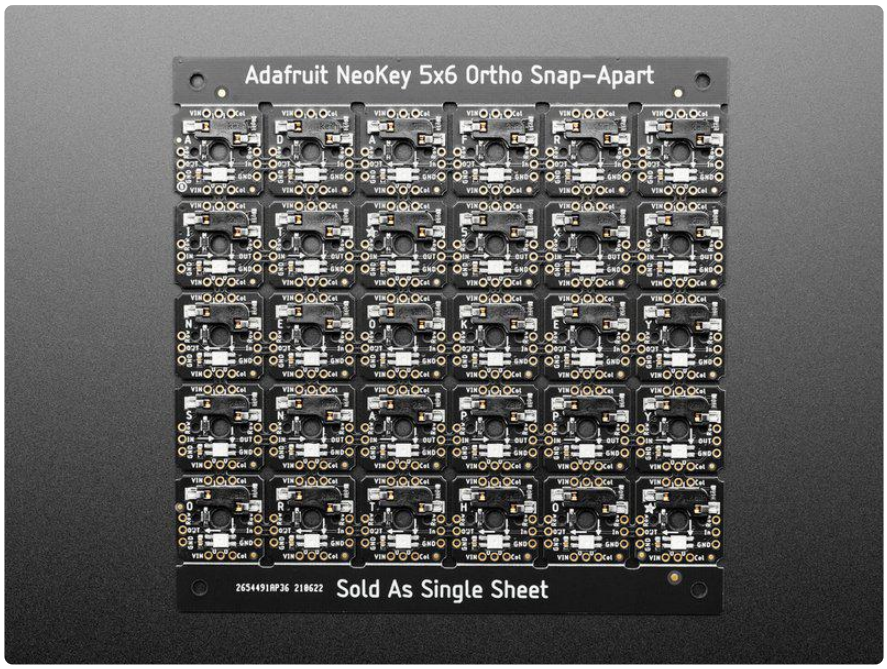


https://learn.adafruit.com/adafruit-neokey-5x6-ortho-snap-apart

Last updated on 2021-11-15 08:26:12 PM EST

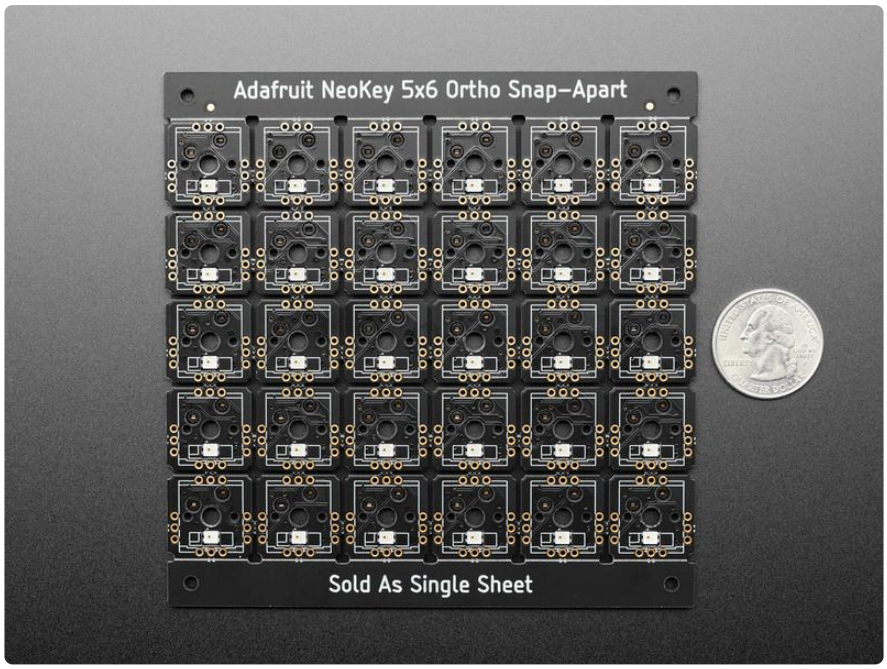
Table of Contents

Overview



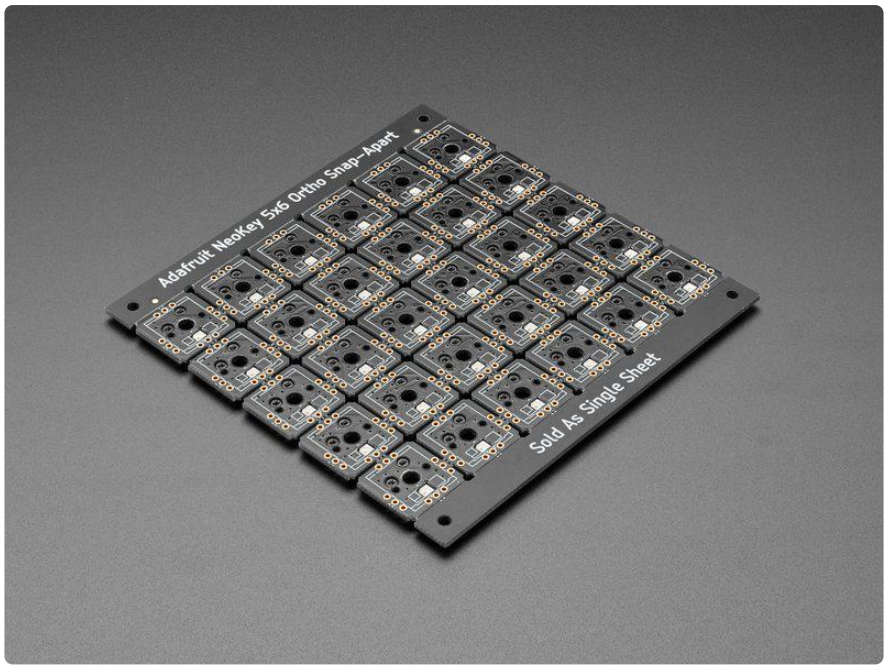
For folks who want ready-to-go keeb action, we've got the lovely Adafruit Macropad with a 3x4 grid of MX+NeoPixel key switches (https://adafru.it/TJB) - but for those who like to forge their own path, we now present the easiest way of creating custom ortholinear (that's a fancy word for gridded) key matrices. Instead of hand soldering together a custom matrix of NeoKey socket breakouts, the Adafruit NeoKey 5x6 Ortho Snap-Apart Mechanical Key Switches with NeoPixel has 30 key switches that can be snapped apart to make any size grid.

That's right! Even though it comes as 5x6, you can turn it into 2x3, 1x6, 4x3...whatever you wish, and it will work just the way you expect. The little break-apart tabs have traces running through them that pass the power, NeoPixel, and keymatrix lines up and down.



Each 'sub-key' is the same - with a column and row line that is diode protected. For whatever grid size you eventually choose, connect a digital IO pin from your microcontroller to each of the outer rows and columns. With the diode, you don't have to worry about key ghosting.

If you want to add glowy goodness, that's even easier. Each key also has a single reverse-mount NeoPixel pointing up through the spot where many switches would have an LED to shine through. Connect 3 to 5V power to any of the + and - pins on the grid, and connect a single digital IO line to the beginning of the 'NeoPixel snake' that winds its way through the matrix. A weak 2.2K pass-through resistor will shunt data through at the end of each row, so you don't have to do any jumpering unless the matrix is not-rectangular.

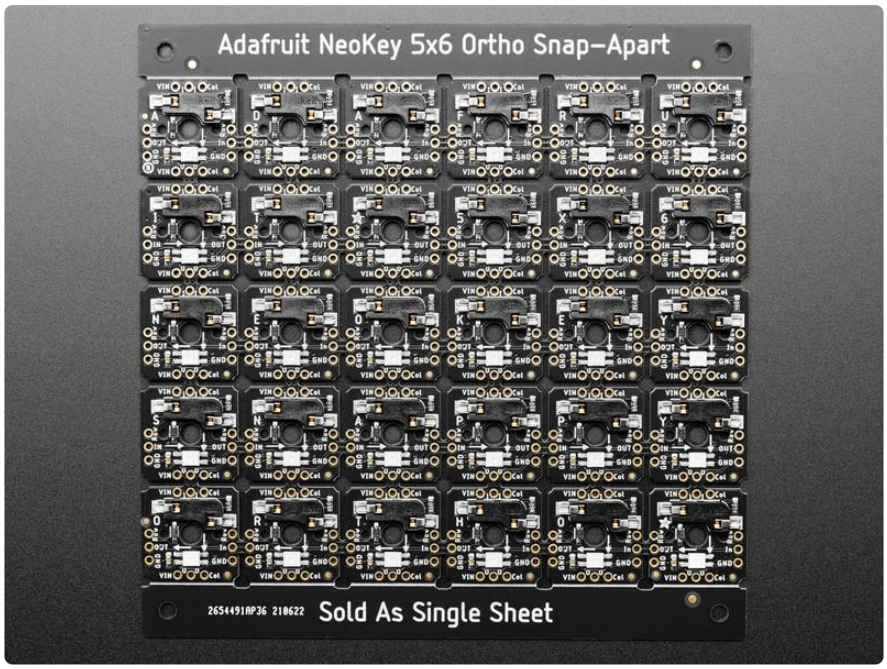


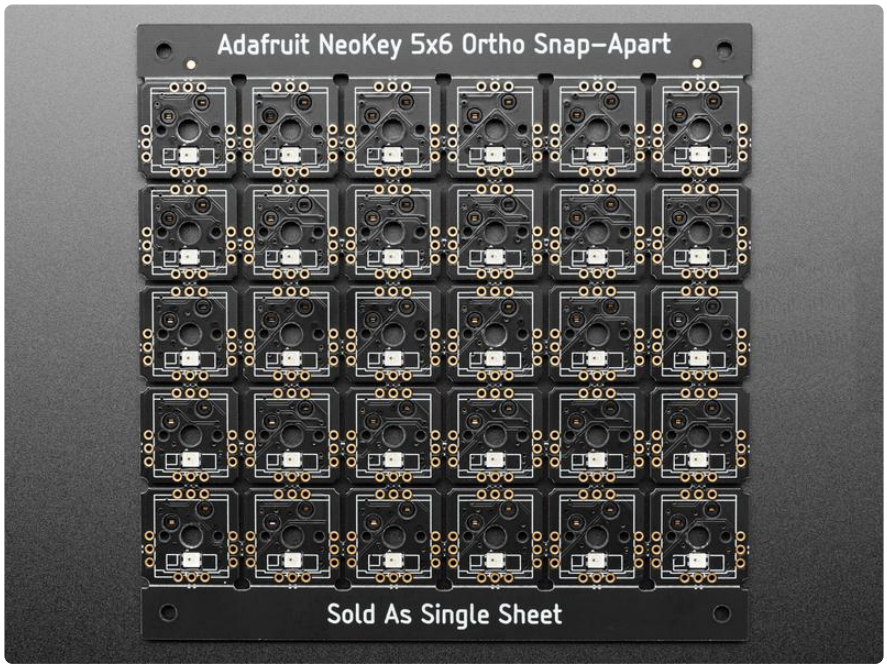
Every breakout has a Kailh socket, which means you can plug in any MX-compatible switch instead of soldering it in. You may need a little glue to keep the switch in place: hot glue or a dot of epoxy worked fine for us. The sub-keys are spaced 0.75" apart for compact keebing. There's large 0.125" mounting holes at the corners of each key for easy mounting to a backplane.

Please note, each order comes with one sheet of assembled PCBs that can be broken apart - we include top and bottom railing bits to help keep it nice and flat during shipment. Once the grid is broken, you could jumper the breakout pads together to re-connect but it won't be very fun. Mechanical switches and keycaps are not included! Use any MX-compatible switch: Kailh, Gateron, etc all work!

Note: The Kailh Low Profile switches will not work! They have a different footprint.

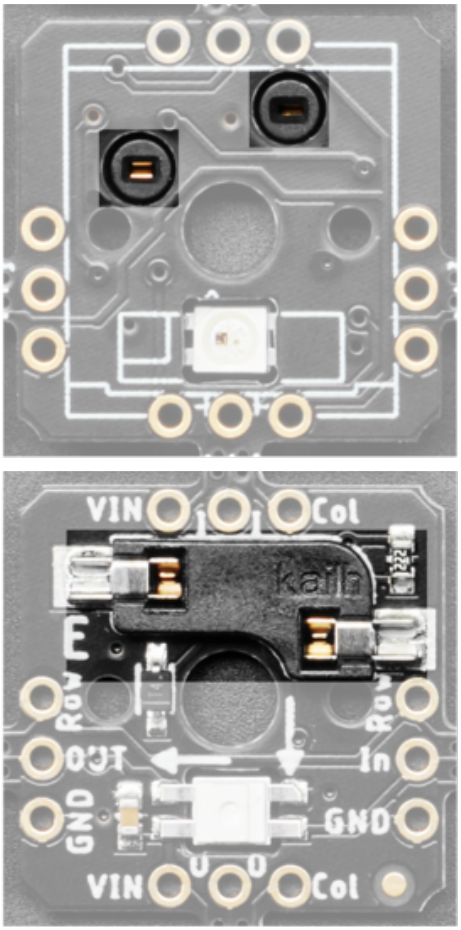
Pinouts





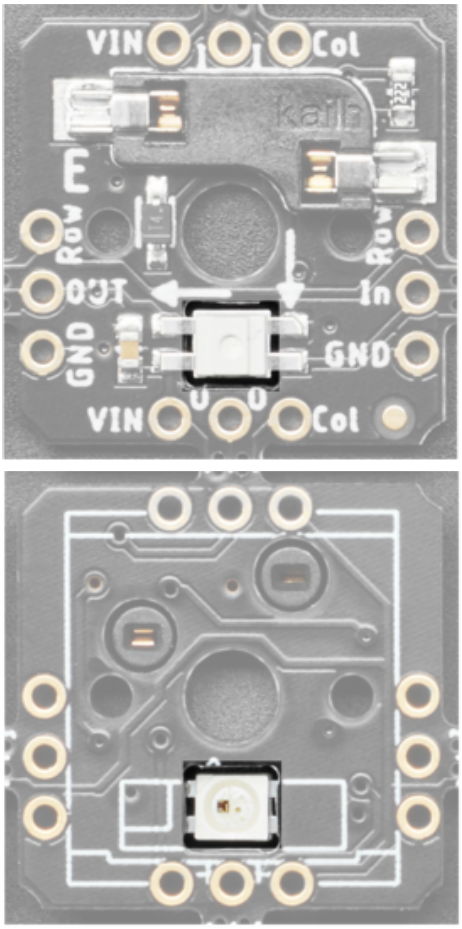
Each of the keys on the NeoKey 5x6 Snap-Apart are the same. The board can be snapped apart to make a matrix of any size 5x6 or smaller. The traces run through the snap-apart bits and pass the power, NeoPixel, and keymatrix lines up and down. Below are the pinouts and features of each individual key.

Key Sockets



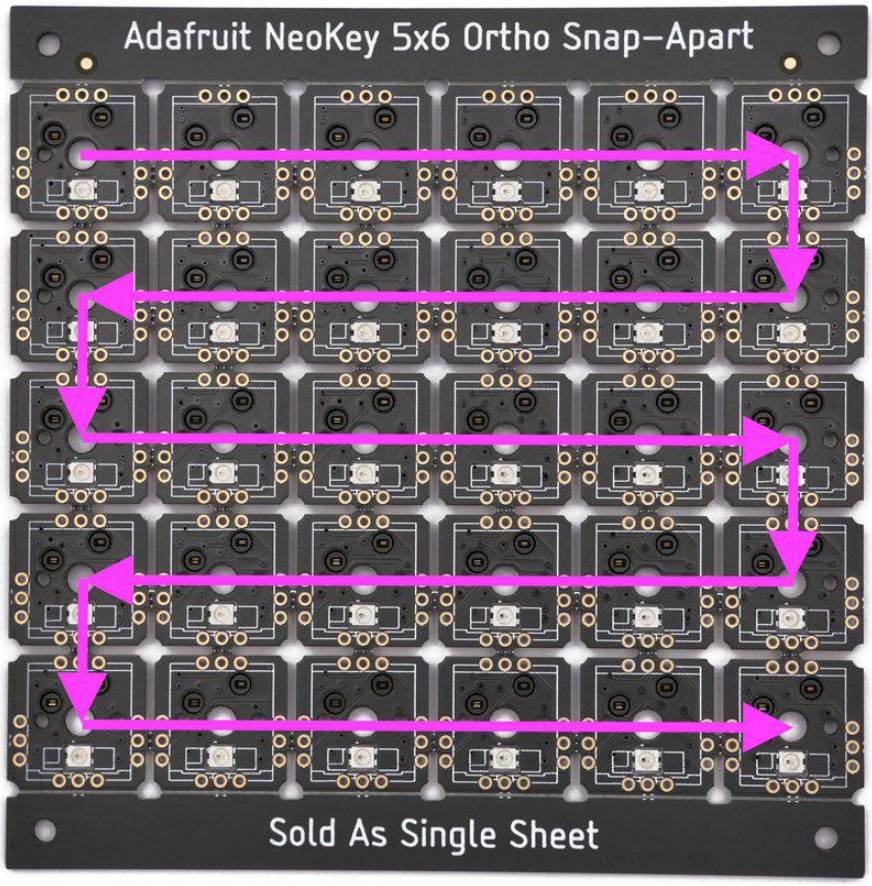
Each key has a Cherry MX or compatible key switch socket. Simply press any compatible key switch into the socket from the top of the board. You can add a dab of glue to keep the switch in place; hot glue or a dot of epoxy will work.

NeoPixel LEDs

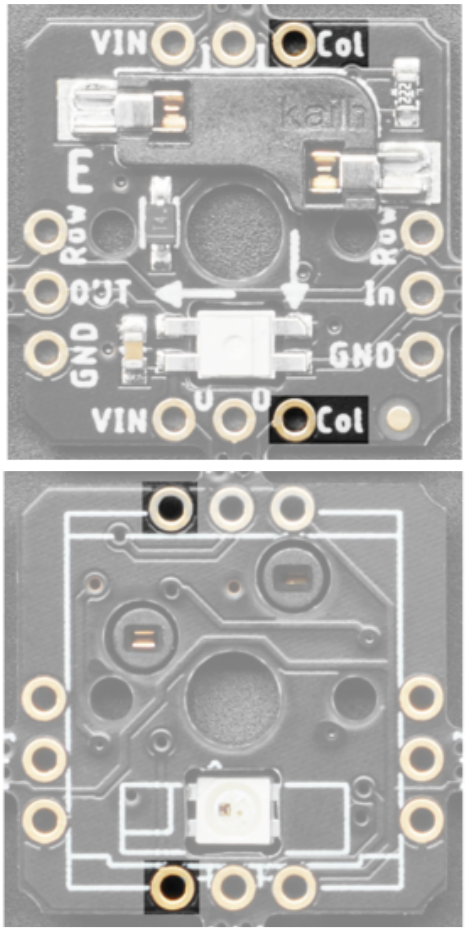


At the bottom of each key is a NeoPixel LED. These LEDs are reverse mount, (https://adafru.it/Txa) meaning they're mounted on the back of the board to shine through to the top. This allows for the key switches to sit flat while still providing rainbow goodness! The pins labeled I or IN are for sending the NeoPixels data. The pins labeled O or OUT are for wiring to another IN pin (to connect a separated matrix, or a second 5x6 sheet).

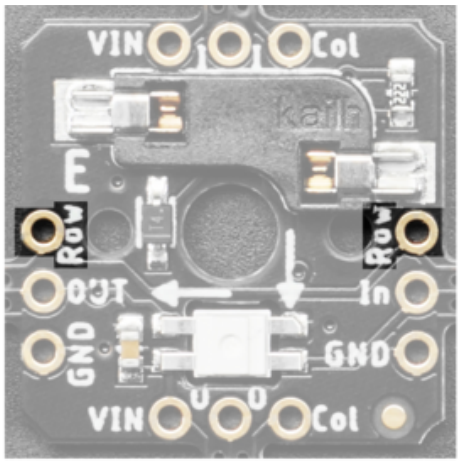
The NeoPixel LEDs are in a zigzag order across the board, beginning at the upper left corner and ending in the lower right corner. If using the full sheet, the "strip" follows the arrows in the image below.


Adafruit NeoKey 5x6 Ortho Snap-Apart
Sold As Single Sheet

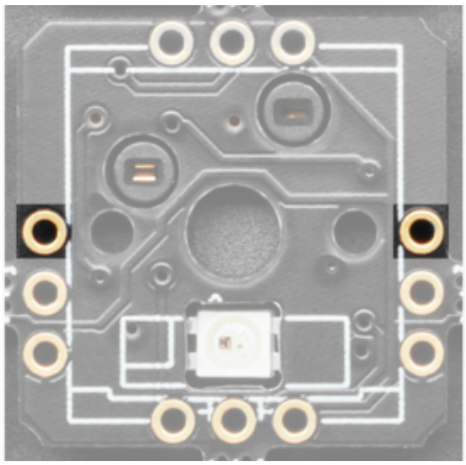
Pins

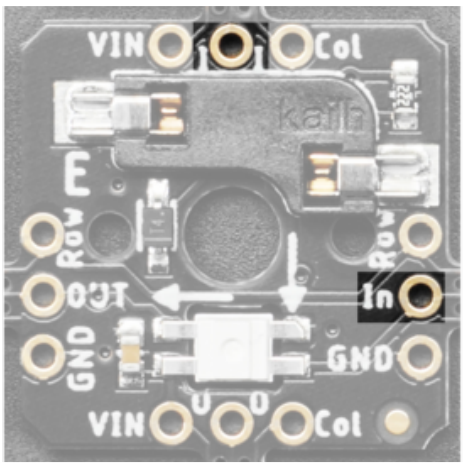


On the top and bottom of each key are the COL (column) pins.

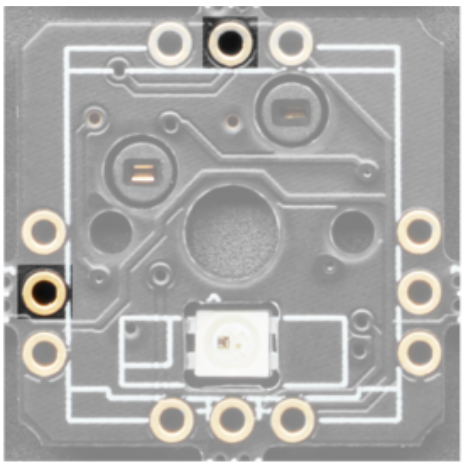


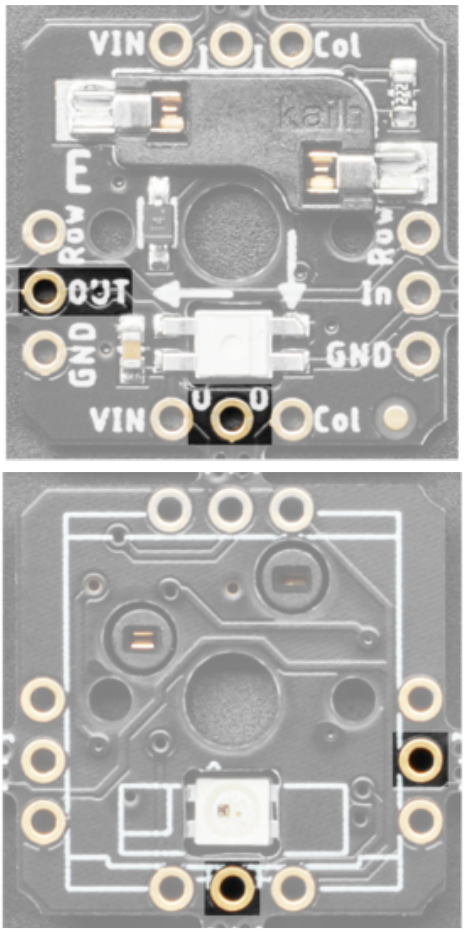
On the left and right of each key are the ROW (row) pins.



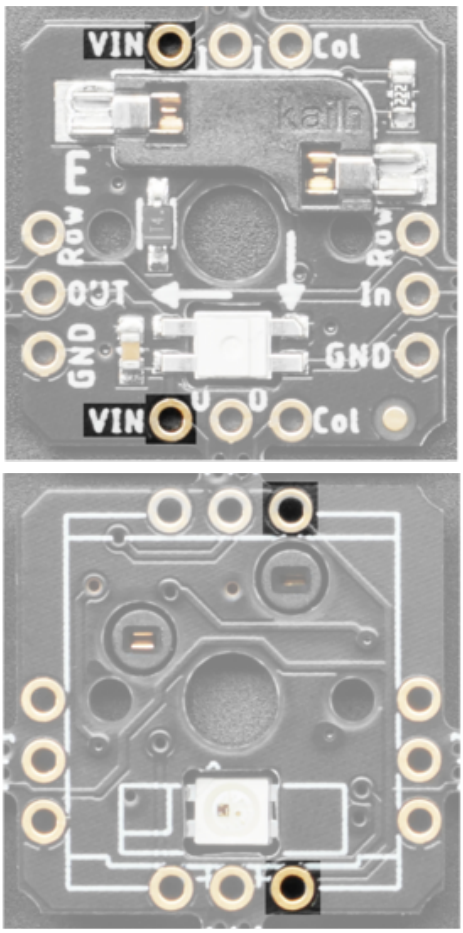


The I and IN pins are the NeoPixel data in pins for sending data to the NeoPixels from a microcontroller, etc.

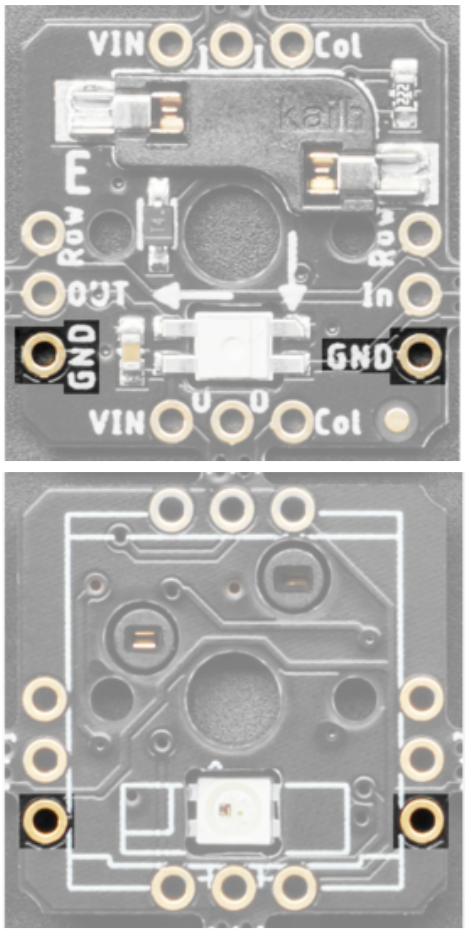




The O and OUT pins are the NeoPixel data out pins, for connecting to the data in pin on another set of NeoPixels.



On the top and bottom of each key are the VIN (power) pins.



The bottom pin on each side of the key are the GND (ground) pins.

CircuitPython

The `keypad` module makes using the NeoKey 5x6 Ortho Snap-Apart with CircuitPython super simple. Combined with the Adafruit CircuitPython NeoPixel (https://adafru.it/yew) library, you can easily write Python code to read key presses and light up the NeoPixel LEDs.

The following example is written for the Adafruit Feather, but you can use the NeoKey Snap with any CircuitPython compatible microcontroller as long as it has enough pins available. Depending on the configuration you choose, you'll need a maximum of twelve IO pins (eleven pins for the keys and one pin for the NeoPixels), a ground pin, and a power pin.

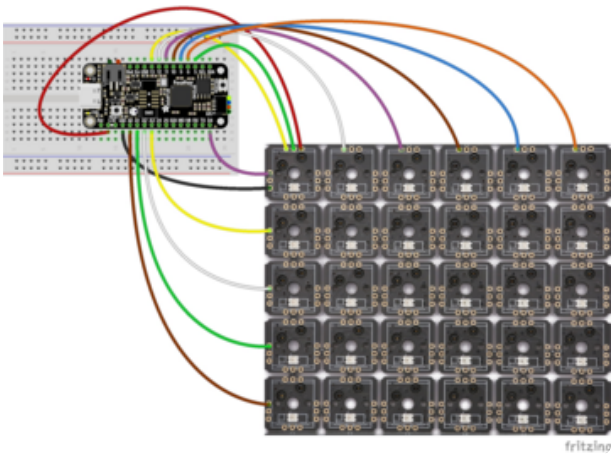
CircuitPython Wiring

The pins are explained on the Pinouts page, but here is a quick summary. The COL pins across the tops of the keys are the column pins. The ROW pins across the sides of the keys are the row pins. The I or IN pins are input for the NeoPixels. The O or OUT pins are NeoPIxel output (for connecting to the input on another set of NeoPixels).

You'll want to solder wires (leave them long enough to connect to a breadboard or solder to a microcontroller directly!) to the following pins, viewing the board from the front: all of the COL pins along the top, all of the ROW pins along the left side, the I pin in the top left, and any VIN and GND.

Connect the NeoKey 5x6 Ortho Snap-Apart sheet to your board exactly as follows. The following shows it wired to a Feather RP2040, but this exact setup will work with any Adafruit Feather. For the purposes of this diagram, references to column pin numbers are 1 to 6, left to right, and references to row pin numbers are 1 to 5, top to bottom.

The diagram shows the NeoKey Snap from the front, e.g. the side to which the keys are socketed. The pin labels are on the back of the NeoKey Snap sheet.



- Board 3V to NeoKey Snap VIN
- Board GND to NeoKey Snap GND
- Board D5 to NeoKey Snap IN (I) top left
- Board D13 to NeoKey Snap COL 1
- Board D12 to NeoKey Snap COL 2
- Board D11 to NeoKey Snap COL 3
- Board D10 to NeoKey Snap COL 4
- Board D9 to NeoKey Snap COL 5
- Board D6 to NeoKey Snap COL 6
- Board D4 to NeoKey Snap ROW 1
- Board A3 to NeoKey Snap ROW 2
- Board A2 to NeoKey Snap ROW 3
- Board A1 to NeoKey Snap ROW 4
- Board A0 to NeoKey Snap ROW 5

The number of available GPIO pins on your microcontroller determines how many keys you can support. The RP2040 has 23 GPIO pins, so in theory, the biggest key matrix it could support would be an 11x11 matrix and still utilise the NeoPixels. That's

121 keys in an 11x11 matrix, in which you would need 22 GPIO pins to handle 11 rows and 11 columns.

CircuitPython Usage

To demonstrate using NeoKey 5x6 Ortho Snap-Apart sheet with CircuitPython, you'll install the necessary library, update your code, and then optionally connect to the serial console (https://adafru.it/Bec) to see information printed out.

To use a 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.

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.

```
"""
NeoKey 5x6 Ortho Snap-Apart simple key press NeoPixel demo.
"""
import board
import keypad
import neopixel

COLUMNS = 6
ROWS = 5

pixels = neopixel.NeoPixel(board.D5, 30, brightness=0.3)

keys = keypad.KeyMatrix(
    row_pins=(board.D4, board.A3, board.A2, board.A1, board.A0),
    column_pins=(board.D13, board.D12, board.D11, board.D10, board.D9, board.D6),
    columns_to_anodes=False,
)


def key_to_pixel_map(key_number):
    row = key_number // COLUMNS
    column = (key_number % COLUMNS)
    if row % 2 == 1:
        column = COLUMNS - column - 1
    return row * COLUMNS + column


pixels.fill((0, 0, 0))  # Begin with pixels off.
while True:
    key_event = keys.events.get()
    if key_event:
        print(key_event)
        if key_event.pressed:
            pixels[key_to_pixel_map(key_event.key_number)] = (255, 0, 0)
        else:
            pixels.fill((0, 0, 0))
```

Now, press a key to see the associated LED light up! If you connect to the serial console, you'll see the key event printed out when you press or release a key.

After importing the necessary library and modules, you configure the number of columns and rows. For the full sheet, there are 6 columns and 5 rows.

Then, you setup the NeoPixels and the keys. There are 30 NeoPixels, and the brightness is set to 30%. The keys use the `KeyMatrix` feature of the `keypad` module. `KeyMatrix` expects a tuple of `row_pins`, and a tuple of `column_pins`. For the NeoKey Snap, you'll also want to set `columns_to_anodes=False`.

Next, is a helper function to map the NeoPixels to the keys. The key numbers read from left to right across each row from top to bottom. The NeoPixels begin in the upper left, and then alternate direction across each row. Simply using the raw NeoPixel number won't match the key number! Therefore, you include a helper function to map the NeoPixels to the appropriate key.

Before the loop, you turn off the NeoPixels by setting them to `(0, 0, 0)` so the program begins with the NeoPixels off.

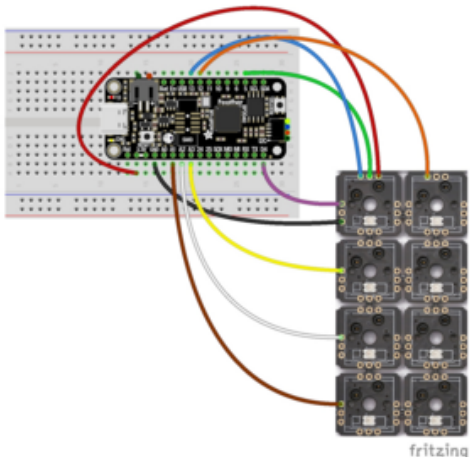
Inside the loop, we begin by getting the key events. Next, we check to see if there has been a key event, e.g. a key has been pressed or released. If so, we print the key event to the serial console, and then check to see if the key event is a key press event. If it is a press event, you light up the associated pixel red using the helper function to identify the pixel number based on key number. Otherwise, you turn all the pixels off.

That's all there is to reading key events and lighting up the associated NeoPixel LED using CircuitPython!

Snap it!

The clever design of the NeoKey Snap is that it is made to be snapped apart! This example shows you how to make a 2x4 keypad. The wiring is essentially the same, except with less wires. Carefully break off the top and bottom bar of PCB (the bits with the product name and "Sold As Single Sheet" written on them). Then, snap off a full row. Finally, snap off two columns. Ta-da! 2x4 keypad.

Wire it up as follows.



- Board 3V to NeoKey Snap VIN
- Board GND to NeoKey Snap GND
- Board D5 to NeoKey Snap IN (I) top left
- Board D13 to NeoKey Snap COL 1
- Board D12 to NeoKey Snap COL 2
- Board D4 to NeoKey Snap ROW 1
- Board A3 to NeoKey Snap ROW 2
- Board A2 to NeoKey Snap ROW 3
- Board A1 to NeoKey Snap ROW 4

Then, simply change the number of `COLUMNS` and `ROWS` at the top of the file so `COLUMNS = 2` and `ROWS = 4`.

```
"""
NeoKey 4x2 Ortho Snap-Apart simple key press NeoPixel demo.
"""
import board
import keypad
import neopixel

COLUMNS = 2
ROWS = 4

pixels = neopixel.NeoPixel(board.D5, 30, brightness=0.3)

keys = keypad.KeyMatrix(
    row_pins=(board.D4, board.A3, board.A2, board.A1),
    column_pins=(board.D13, board.D12),
    columns_to_anodes=False,
)


def key_to_pixel_map(key_number):
    row = key_number // COLUMNS
    column = (key_number % COLUMNS)
    if row % 2 == 1:
        column = COLUMNS - column - 1
    return row * COLUMNS + column


pixels.fill((0, 0, 0))  # Begin with pixels off.
while True:
    key_event = keys.events.get()
    if key_event:
        print(key_event)
        if key_event.pressed:
            pixels[key_to_pixel_map(key_event.key_number)] = (255, 0, 0)
        else:
            pixels.fill((0, 0, 0))
```

Now, press a key to see the associated LED light up! If you connect to the serial console, you'll see the key event printed out when you press or release a key.

That's all there is to customising the size of your NeoKey Ortho Snap-Apart key matrix!

---

keypad docs

keypad docs (https://adafru.it/TXA)

NeoPixel Docs

NeoPixel Docs (https://adafru.it/C5m)

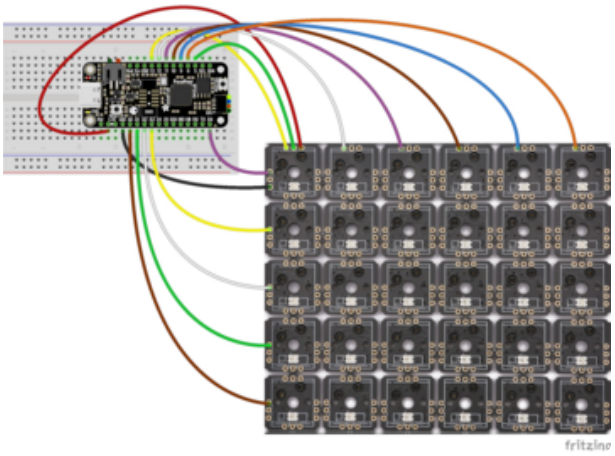
Arduino

Using the Adafruit NeoKey 5x6 Ortho Snap-Apart with Arduino requires the Adafruit Keypad library and the Adafruit NeoPixel library.

First, you'll need to wire it up to your Arduino-compatible microcontroller.

Wiring

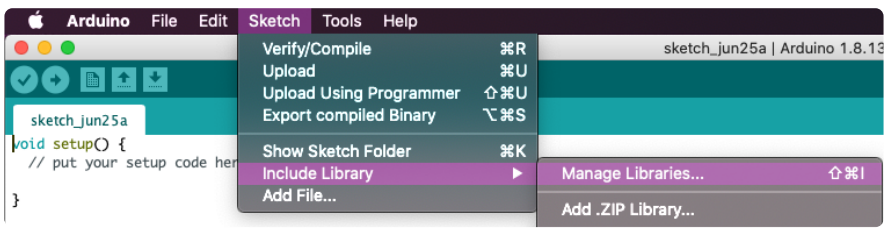
The following example uses the Adafruit Feather RP2040, but you can use any Arduino-compatible microcontroller. Simply update the pins in the example to match your wiring. Pins below are listed to both match the Feather top silk and the Arduino pin numbering.



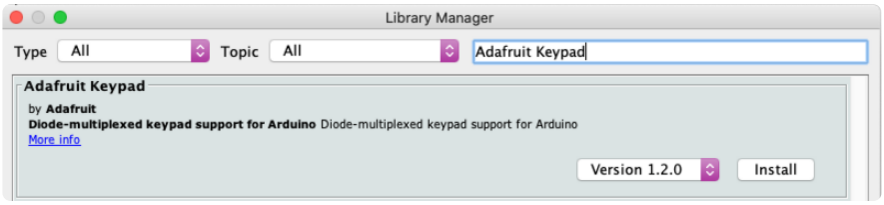
- Feather 3V to NeoKey Snap VIN. You should connect VIN to the Arduino 5V if you are running a 5V board (Arduino Uno, etc.)
- Feather GND to NeoKey Snap GND
- Feather D5 (GP7) to NeoKey Snap IN (I) top left
- Feather D13 (GP13) to NeoKey Snap COL 1
- Feather D12 (GP12) to NeoKey Snap COL 2
- Feather D11 (GP11) to NeoKey Snap COL 3
- Feather D10 (GP10) to NeoKey Snap COL 4
- Feather D9 (GP9) to NeoKey Snap COL 5
- Feather D6 (GP8) to NeoKey Snap COL 6
- Feather D4 (GP6) to NeoKey Snap ROW 1
- Feather A3 (GP29) to NeoKey Snap ROW 2
- Feather A2 (GP28) to NeoKey Snap ROW 3
- Feather A1 (GP27) to NeoKey Snap ROW 4
- Feather A0 (GP26) to NeoKey Snap ROW 5

Library Installation

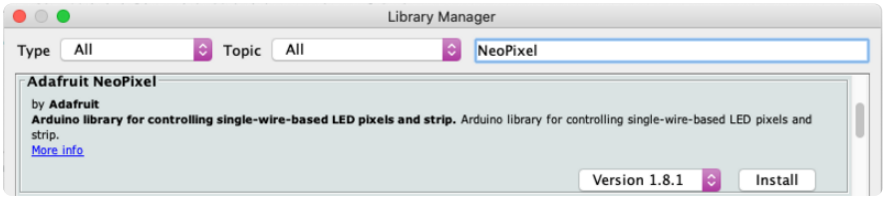
You can install the Adafruit Keypad and NeoPixel libraries for Arduino using the Library Manager in the Arduino IDE.



Click the Manage Libraries ... menu item, search for Adafruit Keypad , and select the A dafruit Keypasd library:



Search for Adafruit NeoPixel , and select the Adafruit NeoPixel library:



Example Code

Save the following file to your computer, then upload it to your Arduino wired to your NeoKey Snap.

```
#include <Adafruit_NeoPixel.h>
#include "Adafruit_Keypad.h"

#define ROWS 5 // rows
#define COLS 6 // columns

#define NEOPIXEL_PIN 7
#define NUM_PIXELS (ROWS * COLS)

Adafruit_NeoPixel strip = Adafruit_NeoPixel(NUM_PIXELS, NEOPIXEL_PIN, NEO_GRB +
NEO_KHZ800);


//define the symbols on the buttons of the keypads
char keys[ROWS][COLS] = {
  {'1','2','3','4','5','6'},
  {'7','8','9','A','B','C'},
  {'D','E','F','G','H','I'},
  {'J','K','L','M','N','O'},
  {'P','Q','R','S','T','U'}
};

uint8_t rowPins[ROWS] = {6, 29, 28, 27, 26}; //connect to the row pinouts of the
keypad
uint8_t colPins[COLS] = {13, 12, 11, 10, 9, 8}; //connect to the column pinouts of
the keypad

//initialize an instance of class NewKeypad
Adafruit_Keypad customKeypad = Adafruit_Keypad( makeKeymap(keys), rowPins, colPins,
ROWS, COLS);

bool lit[ROWS*COLS] = {0};

void setup() {
```

```
  Serial.begin(115200);
  //while (!Serial);
  Serial.println("Ortho 5x6 keypad demo");
  strip.begin();
  strip.setBrightness(40);
  strip.show(); // Initialize all pixels to 'off'

  customKeypad.begin();
  for (int i=0; i<ROWS*COLS; i++) {
    lit[i] = false;
  }
}

uint8_t j=0; // color ticker

void loop() {
  //Serial.println("Test NeoPixels");
  customKeypad.tick();

  while(customKeypad.available()){
    keypadEvent e = customKeypad.read();
    Serial.print((char)e.bit.KEY);
    if (e.bit.EVENT == KEY_JUST_PRESSED) {
      Serial.println(" pressed");
      uint8_t row = e.bit.ROW;
      uint8_t col = e.bit.COL;
      Serial.print("Row: "); Serial.print(row);
      Serial.print(" col: "); Serial.print(col);
      Serial.print(" -> ");
      uint16_t keynum;
      if (row % 2 == 0) { // even row
        keynum = row * COLS + col;
      } else { // odd row the neopixels go BACKWARDS!
        keynum = row * COLS + (5 - col);
      }
      Serial.println(keynum);
      lit[keynum] = !lit[keynum]; // invert neopixel status
    }
    else if(e.bit.EVENT == KEY_JUST_RELEASED) {
      Serial.println(" released");
    }
  }

  for(int i=0; i< strip.numPixels(); i++) {
    if (lit[i]) {
      strip.setPixelColor(i, Wheel(((i * 256 / strip.numPixels()) + j) & 255));
    } else {
      strip.setPixelColor(i, 0);
    }
  }
  strip.show();
  j++;

  delay(10);
}


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

```
  }
}
```

Press any key to see the associated NeoPixel LED light up in a rainbow swirl!

Open the Serial Monitor at 115200 baud. Press the keys to see information printed for each key press.

That's all there is to using the NeyKeEy 5x6 Ortho Snap-Apart with Arduino!

Keypad Docs

Keypad Docs (https://adafru.it/U0e)

NeoPixel Docs

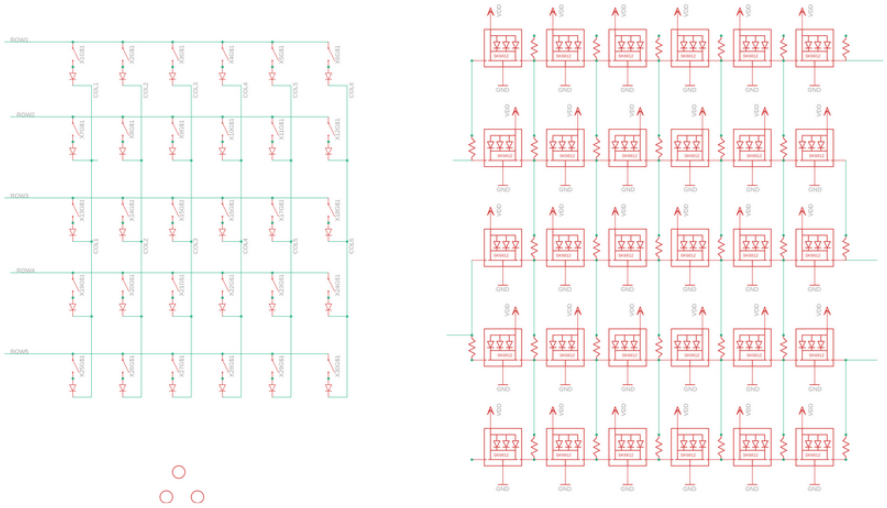
NeoPixel Docs (https://adafru.it/Etk)

Downloads

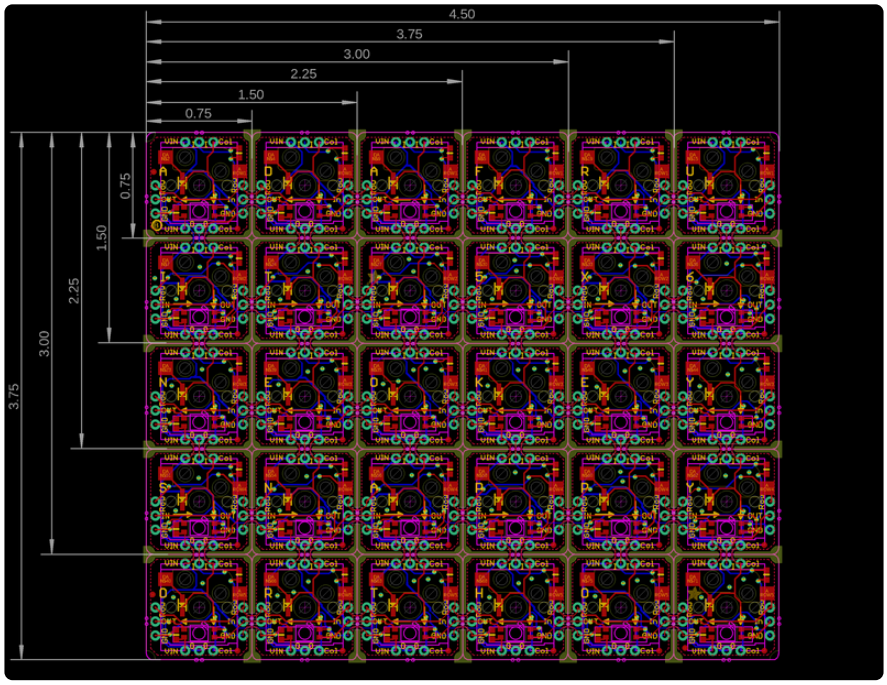
Files

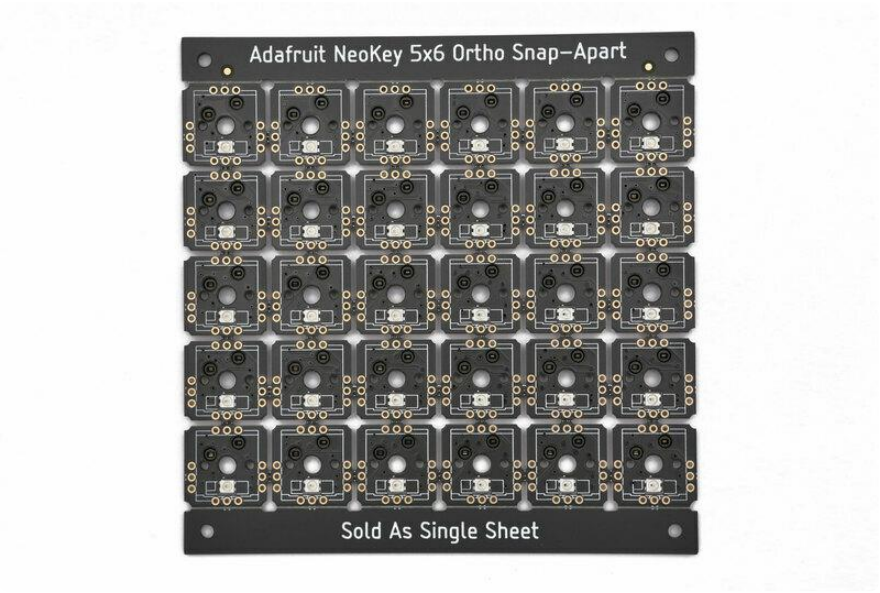
- EagleCAD PCB files on GitHub (https://adafru.it/TKF)

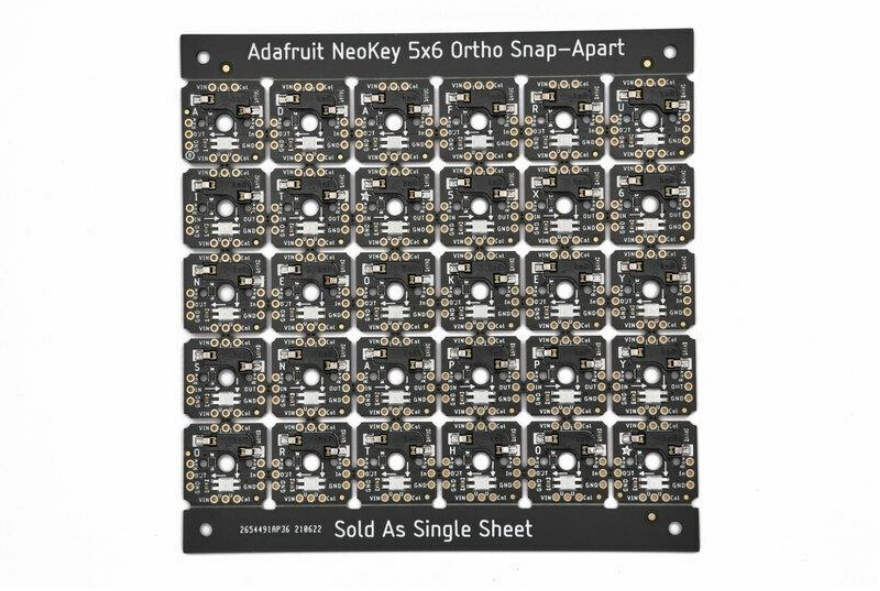
Schematic



Fab Print




Adafruit NeoKey 5x6 Ortho Snap-Apart
Sold As Single Sheet


Adafruit NeoKey 5x6 Ortho Snap-Apart
Sold As Single Sheet