

MultiWingSpan

[Home](#) [Programming](#) [Web Design](#) [Computer Science](#) [Twisting Puzzles](#) [Arduino](#) [BBC micro:bit](#)

BBC micro:bit Bit:Commander Potentiometer

Introduction

The rotary potentiometer in middle of the board is connected to pin 0. It shares this connection with the buzzer. You have to be a little careful when using them both in the same project but, as long as you separate their use in your programs, you can do it.

The potentiometer centres with a little click and reads from 1 to 851 for me.



Programming

This first, simple program shows how to read from the potentiometer. You can view the output in the REPL window.

```
from microbit import *

while True:
    reading = pin0.read_analog()
    print(reading)
    sleep(50)
```

There are marks on the silkscreen of the board. If you sample the readings at each of these marks, you can work out a program for a pretty exact mapping. If you allow some tolerance, ± 5 or 10, you can make it easy for your user.

Rotating Clock

A quick way to map the potentiometer reading to a smaller range is to divide. This one lets you pick out all of the different positions on the clock. That makes for a selection of 0 to 11.

```
from microbit import *

clocks = [
    Image.CLOCK12, Image.CLOCK1, Image.CLOCK2,
    Image.CLOCK3, Image.CLOCK4, Image.CLOCK5,
    Image.CLOCK6, Image.CLOCK7, Image.CLOCK8,
    Image.CLOCK9, Image.CLOCK10, Image.CLOCK11]

# set pin8 to LOW for potentiometer
pin8.write_digital(0)

while True:
    reading = pin0.read_analog()//72
    display.show(clocks[reading])
    sleep(50)
```

Lighting Up

This program uses the same principle as the last, except it lights up a number of the Neopixels according to the dial position.

BBC Microbit

- + [Block Editor - The Basics](#)
- + [Block Editor - Components](#)
- + [Kodu - micro:bit Worlds](#)
- + [JavaScript Blocks](#)
- + [JavaScript Blocks - Exercises](#)
- + [Blocks - Bit:Bot](#)
- + [Blocks - Bit:Commander](#)
- + [MicroPython - Starting Off](#)
- + [MicroPython - Examples](#)
- + [MicroPython - Components](#)
- + [MicroPython - Breakout Boards](#)
- + [MicroPython - Exercises](#)
- + [MicroPython - Pi Accessories](#)
- + [MicroPython - Bit:Bot](#)
- [MicroPython - Bit:Commander](#)
 - ✱ [Bit:Commander](#)
 - ✱ [The Joystick](#)
 - ✱ [The Neopixels](#)
 - ✱ [The Potentiometer](#)
 - ✱ [The Pushbuttons](#)
 - ✱ [The Buzzer](#)
 - ✱ [Evasion Game](#)
 - ✱ [Light's Out Game](#)
 - ✱ [Simon Game](#)
 - ✱ [Bit:Bot/Robot Controller](#)
 - ✱ [Text Entry](#)
 - ✱ [Unicorn Commander](#)
- + [MicroPython - Projects](#)
- + [MicroPython - Visual Basic](#)
- + [Other - Odds & Ends](#)



```
from microbit import *
import neopixel

# Initialise neopixels
npix = neopixel.NeoPixel(pin13, 6)

red = (64,0,0)
off = (0,0,0)
# set pin8 to LOW for potentiometer
pin8.write_digital(0)

while True:
    reading = pin0.read_analog()//123
    for pix in range(0, len(npix)):
        if pix<reading:
            npix[pix] = red
        else:
            npix[pix] = off
    npix.show()
    sleep(20)
```

Next Steps

A useful next task would be to take readings at the positions indicated on the board and work out some base codes to convert each one to a number from 0 to 6 (using zero for the min and 6 for the max positions). That would make a much improved version of the last program and a basis for using the potentiometer in future projects.