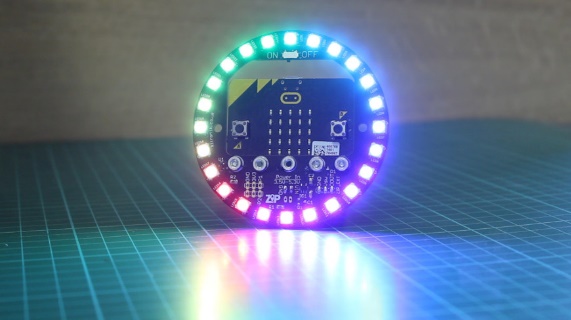
## Lesson 26 – Activity Sheet

## Setting the Scene



The Kitronick Halo board for the BBC micro:bit is features 24 ZIP LEDs, which are individually addressable full colour LEDs. These LEDs are often referred to as NeoPixels or pixels, as they emulate the small pixels that are used on screen, television and other digital displays. Kitronick say this means that each LED can display a huge spectrum of colours, allowing amazing colourful effects to be achieved. The Halo has been designed to bolt directly onto the BBC micro:bit using five bolts which are secure and robust. Set up your Halo with your micro:bit.

## Getting Started

Let’s turn the pixels on. Which pixel do you think the program below will turn on? What colour will the pixel be?

from microbit import \*

import neopixel

num\_of\_pix = 24

np = neopixel.NeoPixel(pin0, num\_of\_pix)

np[0] = (255, 0, 0)

np.show()

Copy up the program code and download it to your micro:bit. Were you correct in guessing which pixel would light up? Can you now turn pixel number 8 on and turn it green?

## **Turning on More Than One Pixel**

There are 24 pixels which can be turned ON or OFF. Can you turn pixels on at the same time?

num\_of\_pix = 24

np = neopixel.NeoPixel(pin0, num\_of\_pix)

np[8] = (0, 255, 0)

**np[16] = (0, 255, 0)**

np.show()

Simply add in an additional line for each pixel that you want to turn on show in the highlighted line. Can you turn on four of the pixels?

## **Light Chaser**

Now that you can turn any of the pixels ON or OFF you can build a light chaser. This simple program lights the first pixel, then the second, then the third and so on. Look at the example program code below

from microbit import \*

import neopixel

num\_of\_pix = 24

np = neopixel.NeoPixel(pin0, num\_of\_pix)

np[0] = (0, 255, 0)

np.show()

sleep(1000)

np[1] = (0, 255, 0)

np.show()

sleep(1000)

np[2] = (0, 255, 0)

np.show()

sleep(1000)

Adapt the program show above to:

* Adjust the colour of each pixel
* Adjust the time delay between each pixel
* Make each light chase the other by turning off the previous pixel, use the code example below

np[0] = (0, 255, 0)

np.show()

sleep(1000)

np.clear()

The np.clear() turns of the previous pixel so it creates the impression that one single pixel is moving around the circle.

## Success Criteria

* Light up one pixel
* Light up two or more pixel
* Change the colour of the pixels

## Pro-tip

Use a loop and iterate over the pixels. The original code to make the light chaser has a lot of lines of code. This will work but is not the most appropriate programming style. Iteration is a programming technique where a section of code is looped over or repeated a number of times. The code is applied to each of the separate pixels.

from microbit import \*

import neopixel

from random import randint

np = neopixel.NeoPixel(pin0, 24)

while True:

for pixel\_id in range(0, len(np)):

red = randint(0, 255)

green = randint(0, 255)

blue = randint(0, 255)

np[pixel\_id] = (red, green, blue)

np.show()

sleep(100)

Copy up the program and add the np.clear() to complete your chaser.

## Test Time

Ensure that the Halo is attached and then download the program code to your micro:bit. Try not to look directly at the LEDs as they are bright. You can always place a sheet of paper over the Halo to reduce the brightness.

## Stretch Tasks

Create a program that displays the colours of a rainbow, this could be a chaser or a simple static display. To calculate the required RGB colour values you can either experiment with the values or search for a website that lists the RGB values required.

## Final Thoughts

In the next lesson you will learn how to use the Halo as a real time temperature display