Driving WS2812B LEDs (aka NeoPixels) with Picon Zero

Requirements:

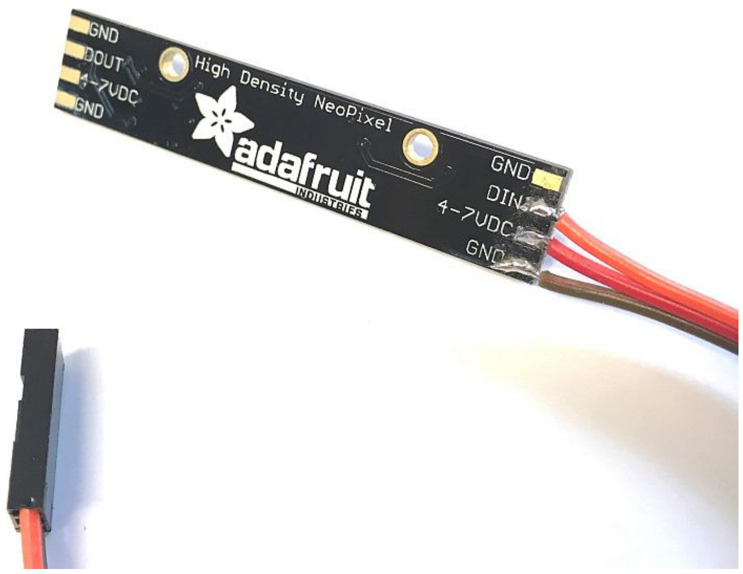
- Picon Zero
- Neopixel strip or matrix up to 64 pixels
- Raspberry Pi Connected to Internet, keyboard and screen
- Raspberry Pi already setup following Worksheet 01

Wire up the NeoPixels

The easiest way to connect sensors, switches and other input devices to Picon Zero is to use a 3-pin “dupont” female header. You can make these up yourself, or easier to get a ready-made cable and attach your input device to the other end. These are the same cables sold as servo cables:

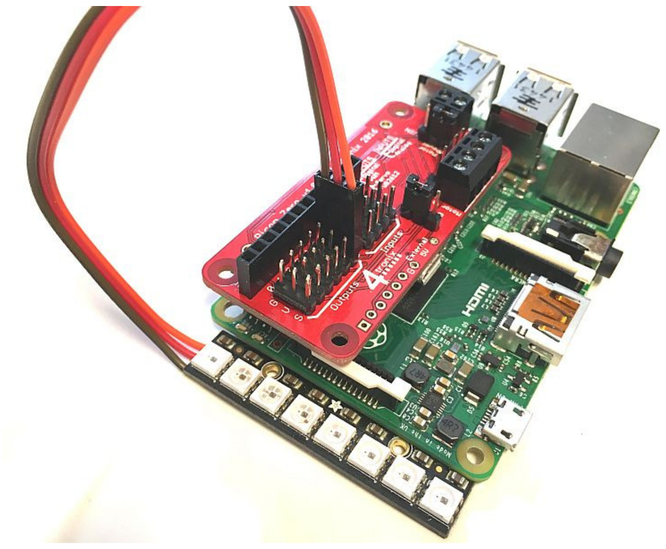
We stock these at 4tronix here:
http://4tronix.co.uk/store/index.php?rt=product/product&product_id=553

On the WS2812B (neopixel) Strip/Array there will be three connections marked for Ground, Vcc (or 5V) and Din (for Data In). There may also be a connection for Dout (Data Out) – do not use this unless adding a second or subsequent strip/matrix. The total number of LEDs cannot exceed 64 as the Picon Zero will not control any more than that due to memory limitations.



Above is an 8-pixel strip from Adafruit. You can see that we’ve connected Orange to Data In, Red to 5V and Brown to Ground.

The other end of the GVS cable plugs directly into Output 5. Note that Picon Zero only supports neopixels on Output 5 due to internal memory limitations.



How do NeoPixels Work?

WS2812B chips contain 3 LEDS: Red, Green and Blue. They also contain a controller chip which takes the colour data for each LED through a single input, Din.

Each chip reads in all the data for its own LEDS, then passes the remaining data directly through to the next LED in the chain.

Every time you update even a single LED, all the data is sent for each LED

As this is a serial process, with 64 chips in the chain a significant time is spent updating the LEDs.

Programming the Test

Set the output configuration of Output 5 to WS2812B:

pz.setOutputConfig (5, 3)

Then you can send pixel data directly, either individual pixels, or changing all pixels to the same colour. Each pixel data command takes 3 values (arguments) for Red, Green and Blue values. Each value can be from 0 (fully off) to 255 (fully on). So setting pixel 4 to brightest white would be:

pz.setPixel (4, 255, 255, 255)

and setting all pixels to a dim Blue would be:

pz.setAllPixels (0, 0, 20)

Here is a complete (but trivial) program to flash all pixels on and off

```
import piconzero as pz, time
pz.init()
pz.setOutputConfig(5, 3)    # Set output to WS2812B
while True:
    pz.setAllPixels(255,255,255) # all Pixels to White
    time.sleep(1)
    pz.setAllPixels(0, 0, 0) # all Pixels Off
    time.sleep(1)
```

Other Useful Functions

As well as the basic functions shown above, there are a few additional functions that can improve the experience of working with neopixels.

setBrightness ( ): This function sets the overall brightness of the display to a value between 0 (off) and 255 (brightest). All subsequent updates to the pixels will be scaled so that the maximum brightness is not exceeded. The default setting for this is 40 so the display will be quite dim initially (and therefore easier on the eyes)

setPixel (r, g, b, False) and setAll Pixels (r, g, b, False): Adding the "False" argument after the RGB values, stops the Picon Zero updating the pixels. If you have a lot of changes to make, it is best not to update the LEDs until all your changes are ready to be shown. Each update takes quite a bit of time (see the panel "How do neopixels Work?") so it is better not to spend this time unless the display needs updating

updatePixels ( ): If you have updated the pixel data, but not updated the actual pixels, then using this command will send all the new data to the pixel array immediately

Note:

"NeoPixels" is a name given to WS2812B and compatible devices by Adafruit. We use it here for convenience and because the name has moved into common parlance – rather like using "Hoover" to mean a vacuum cleaner.

The WS2812B is a 5mm square device that is most common at the time of writing. 4tronix also use another chip (SK6812) which is software and hardware compatible, but is only 3.5mm square so many more can be fitted into the same space