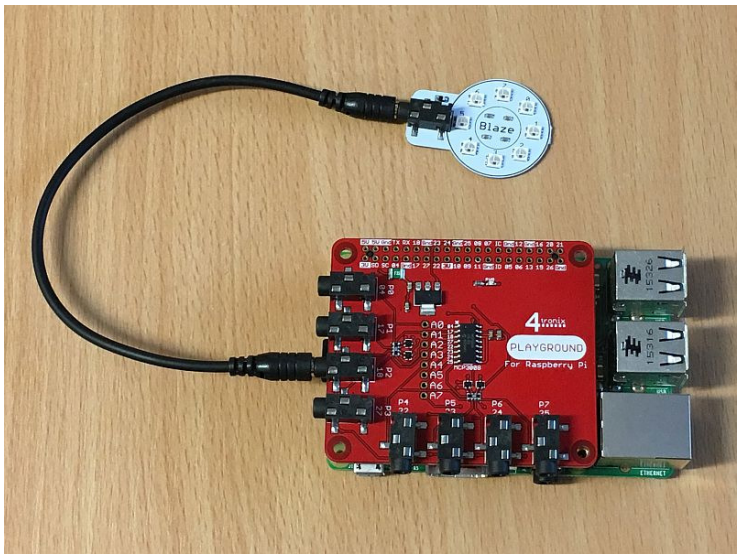
Using the Blaze Gizmo with Playground for Raspberry Pi

Requirements:

- Playground for Raspberry Pi
- Blaze Gizmo
- Connection cable
- Installed playground libraries with neopixel support (see below)

Connections:

1. Plug connection cable from **Port 2** on Playground to Blaze Gizmo



Installing Playground Software with NeoPixel Support:

From the command line (eg. using LXTerminal) after the $ type:

```
$ wget www.4tronix.co.uk/playground/pg.sh -O pg.sh
$ bash pg.sh
```

Once these commands have finished processing, you can enter the playground folder using:

```
$cd playground
```

Python Example Software:

The example software below sends a Blue light around the Blaze

- We need to include the playground library file to handle the pixels
- Then we initialise the pixel data with 8 pixels
- Then we set pixel 0 to Red and pixel 1 to Green
- Wait a bit, then swap over the colours
- Wait a bit then repeat. Finish by typing Ctrl-C

```
import playground as gp
from time import sleep

gp.init(8)

    while True:
    gp.setPixel(0, 0, 0, 255)
    gp.setPixel(7, 0, 0, 0)
    sleep(0.1)
    for i in range(1, 8):
        gp.setPixel(i, 0, 0, 255)
        gp.setPixel(i-1, 0, 0, 0)
        sleep(0.1)
```

Other commands we could use:
gp.init(8, Brightness=100). The default brightness is 70. It can go up to 255
gp.clear(). Clears all the pixels in the chain
gp.setAll(r, g, b). Sets all the pixels to the same colour
gp.update(). Updates all pixels with latest data set

The commands for setting pixels, all take a default update of True. If you include “Update=False” as an argument, then the data will be updated as requested, but it won’t show on the pixels. This is useful if you want to make a number of changes then show them all at once. It makes for a cleaner, snappier update process. For example, we could change the demo program as follows:

```
import playground as gp
from time import sleep

gp.init(8, Brightness=150)

    while True:
    gp.setPixel(0, 0, 0, 255, Update=False)
    gp.setPixel(7, 0, 0, 0, Update=False)
    gp.update()
    sleep(0.1)
    for i in range(1, 8):
        gp.setPixel(i, 0, 0, 255, Update=False)
        gp.setPixel(i-1, 0, 0, 0, Update=False)
        gp.update()
        sleep(0.1)
```