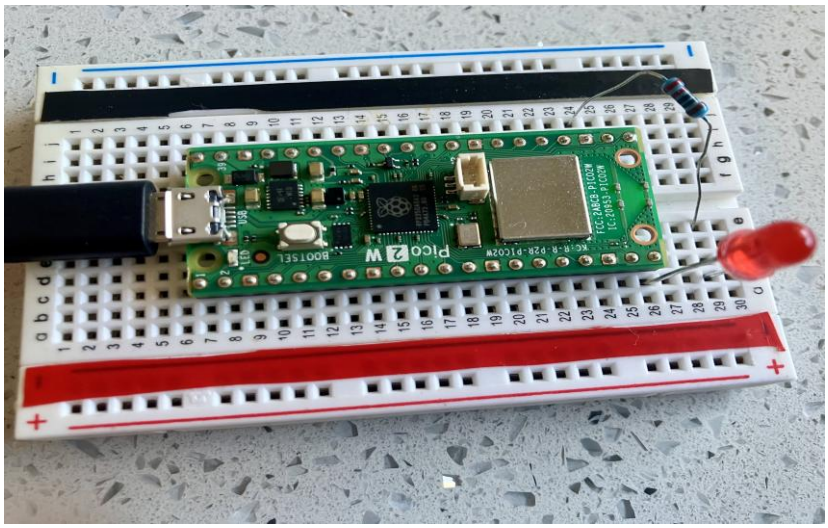
Raspberry Pi Pico
This activity shows how the brightness of an LED can be changed using Pulse Width Modulation. We will program the Raspberry Pi Pico 2W using the Arduino IDE.

The Raspberry Pi Pico 2W is a microcontroller board costing only £7.60 with headers. It is built around the Raspberry chip RP2350 with 2.4 GHz wireless and Bluetooth.

Time: About 60 minutes

Kit list:

- Raspberry Pi Pico 2W
- An LED, any colour
- Resistor, about 220 Ω to protect the LED
- Breadboard
- USB cable
- A computer loaded with Arduino



Instructions:
The Pico has 26 General Purpose Input/Output (GPIO) pins, which provide 3.3 volts when HIGH and O volts when LOW. There are 16 independent Pulse Width Modulation (PWM) pins to choose from.

We use the principle of Pulse Width Modulation to vary the brightness. It provides analogue results from digital values. The PWM pin is switched on and off very fast. We can change the amount of time that the output remains high compared to off. If a value of zero is sent to the pin by the function analogWrite (pin, value), this represents zero percentage Duty Cycle and the LED will be off. A value of 255 represents 100 percent Duty Cycle and full brightness. Sending a value of 64, a quarter of 255, would mean that the LED would be on for 25% of the time and off for 75%. In this way the function analogWrite can control the brightness of the LED.

Connect the positive end of the LED, with the longer pin, to one of the PWM pins on the Pico, for example pin 15, which is in the bottom left corner, as shown in the photo. The other end of the LED can go anywhere on the board. One end of a resistor is inserted in the same row while the other end of the resistor is connected to any of the ground pins, in this case, on the other side of the Pico.

To test that the Pico is connected to the computer properly, we can run a test sketch in Arduino called blink. You will find it in ‘File’ under ‘Examples’ followed by ‘0.1 Basics’. In ‘Tools, you need to find the Raspberry Pi Pico 2W board. Make sure you have the correct port. Click on the arrow pointing to the right to upload the sketch. The built-in LED on the Pico board will blink.

Begin by starting a new sketch under ’File’. In our PWM sketch we need to define which pin we have chosen for the LED output, in this case 15. In the void loop, where any initialisation takes place, we make pin 15 the output for the LED. Make sure you have included the semi-colons!

```
// define the PWM pin connected to the LED
const int ledPin = 15;

void setup() {
pinMode (ledPin,  OUTPUT);

}
```

We create a for loop for a variable called brightness, changing the number from zero to 255, increasing it by one each time. The brightness will vary from zero, off, to a maximum. There is a delay of 50 milliseconds between each change. Having reached maximum brightness after about 12 seconds, we decrease the brightness variable by one, down to zero.

The void loop is repeated over and over.

```
void loop() {
// make the LED go gradually from off to on
for (int brightness = 0; brightness <=255; brightness++)
{analogWrite(ledPin,brightness);
delay(50);
}

// make the LED go gradually from on to off
for (int brightness = 255; brightness >=0; brightness--)
{analogWrite(ledPin,brightness);
delay(50);
}
}
```

Check that you haven't missed out any brackets or semi-colons by clicking on the tick icon to verify the sketch. Upload the sketch and watch the brightness change.

Save the sketch to the Arduino folder.

Further resources:

The Pico features in the book 'Make:Radio' by Fredrik Jansson, OH1HSN and it is the subject of an RSGB presentation '**Raspberry Pi Pico and Outreach**' by John Hislop, G7OHO.

The book 'Raspberry Pi Pico for Radio Amateurs' by Dogan Ibrahim, G7SCU has many projects for amateur radio. For example, the author describes how to make a Pico based FM radio, a Morse decoder and a Frequency counter. He uses the Python programming language while Fredrik Jansson uses Arduino.