IOT Thing Development

Due, Monday 11/11/2024

By Ben Pedler

For my IT project this term, I made a device with micro python and a Pico pi w board. This device consists of two actuators, a potentiometer, and a button. This device also consists of two outputs, a buzzer, and an LED light.

When the button is being held down, it allows the function 'adjust_light_and_buzzer' to run. This function allows the potentiometer's value to be read. When the potentiometer is turned, the LED light will dim / get brighter. When the potentiometer value is greater than 65534, the buzzer will sound. The reason I chose this value is because it is one less than the greatest value. When the buzzer sounds, stop turning the potentiometer because it is at its max.

The components for this device include:

- Raspberry Pi Pico W
- Bread board
- Button
- potentiometer
- Buzzer
- LED Light
- USBC - Micro USB Cable (For Mac)
- 10 Wires
- Resistor

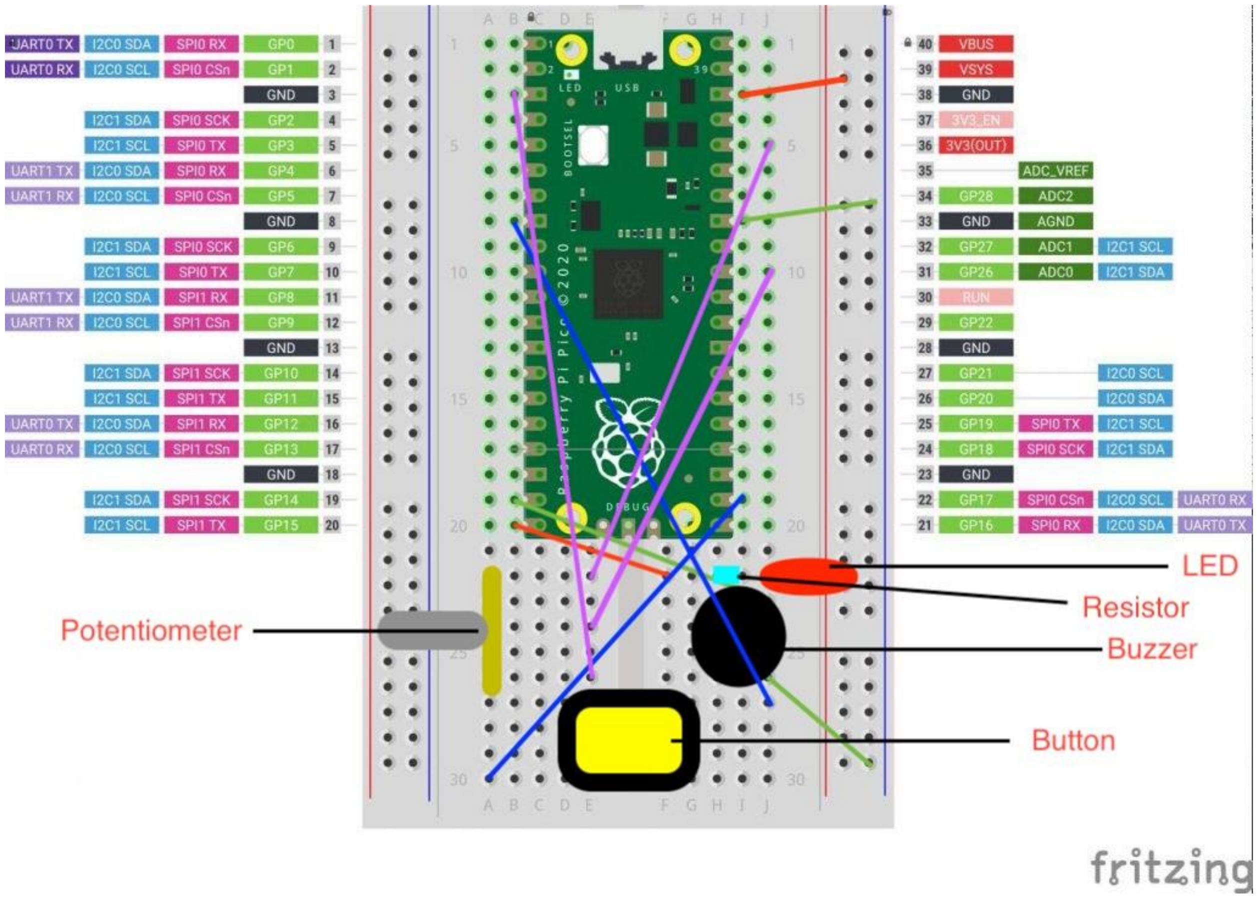
Connecting the components

The LED light is connected to the GPIO Pin 15 and a GND Pin. The ground pin provides a low resistance pathway back to the power source. The GPIO Pin 15 is at the end of the Pico Pi W meaning it is easy to wire.

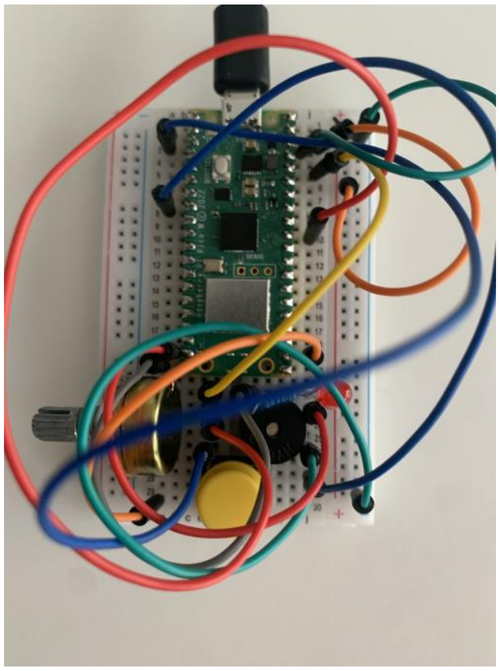
The Buzzer is connected to the GPIO Pin 14 and a GND Pin. The ground pin provides a low resistance pathway back to the power source. The GPIO Pin 14 is near the end of the Pico Pi W meaning it is easy to wire.

The Button is connected to the GPIO Pin 17 and a GND Pin. The ground pin provides a low resistance pathway back to the power source. The GPIO Pin 17 is near the end of the Pico Pi W meaning it is easy to wire.

The potentiometer is connected to the GPIO Pin 26, a GND Pin and the 3V3(OUT) Pin. The significance with Pin 26 is that the GPIO can double as an analog and digital pin, allowing the potentiometer to have different values from 0 - 65,535. The 3V3(OUT) Pin provides 3.3 volts into the potentiometer. The ground pin provides a low resistance pathway back to the power source.



This is an image of my bread board with all actuators, sensors, and wires in it. Edited by me, base picture by Fritzing.



This is an image of my Cables and setup of the Breadboard with the Pico Pi W.

This is an image of the wire that connects the Pico Pi W to my computer.

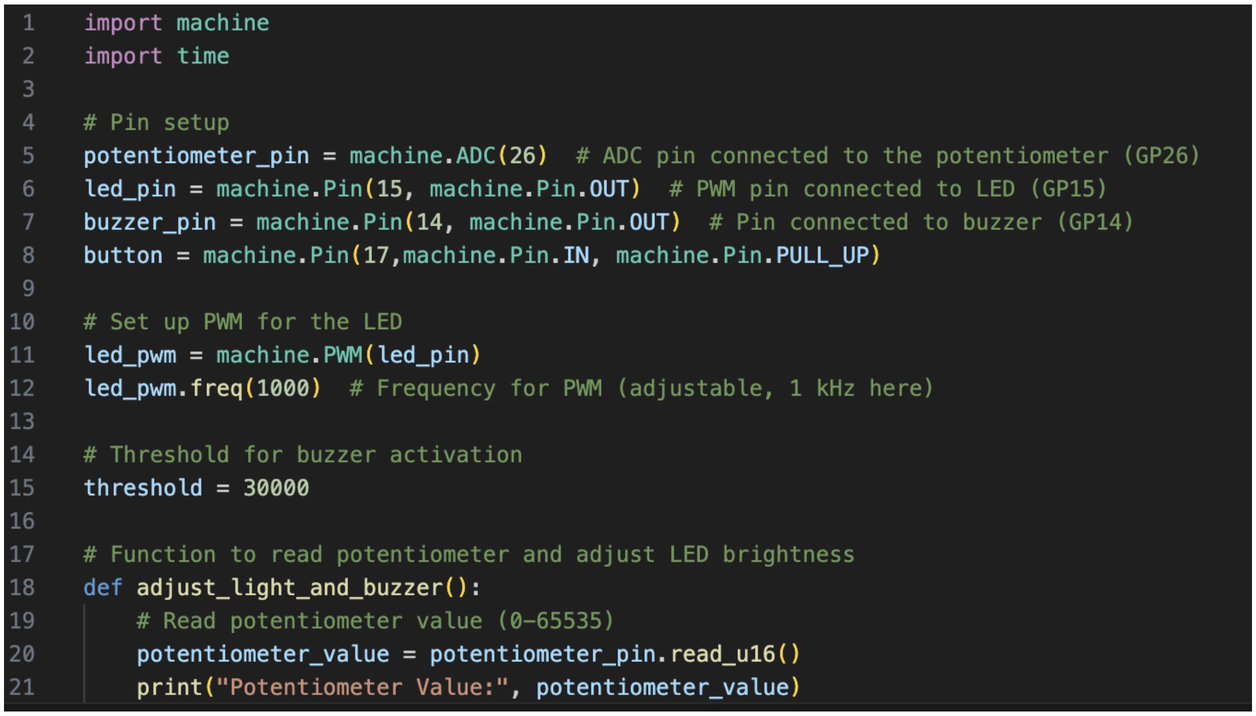
Pulse Width Modulation (PWM)

PWM is a series of signals that can set the frequency of outputs. I am using pulse width modulation in my micro python code so that I can easily set the different LED light brightness levels whilst turning the potentiometer.

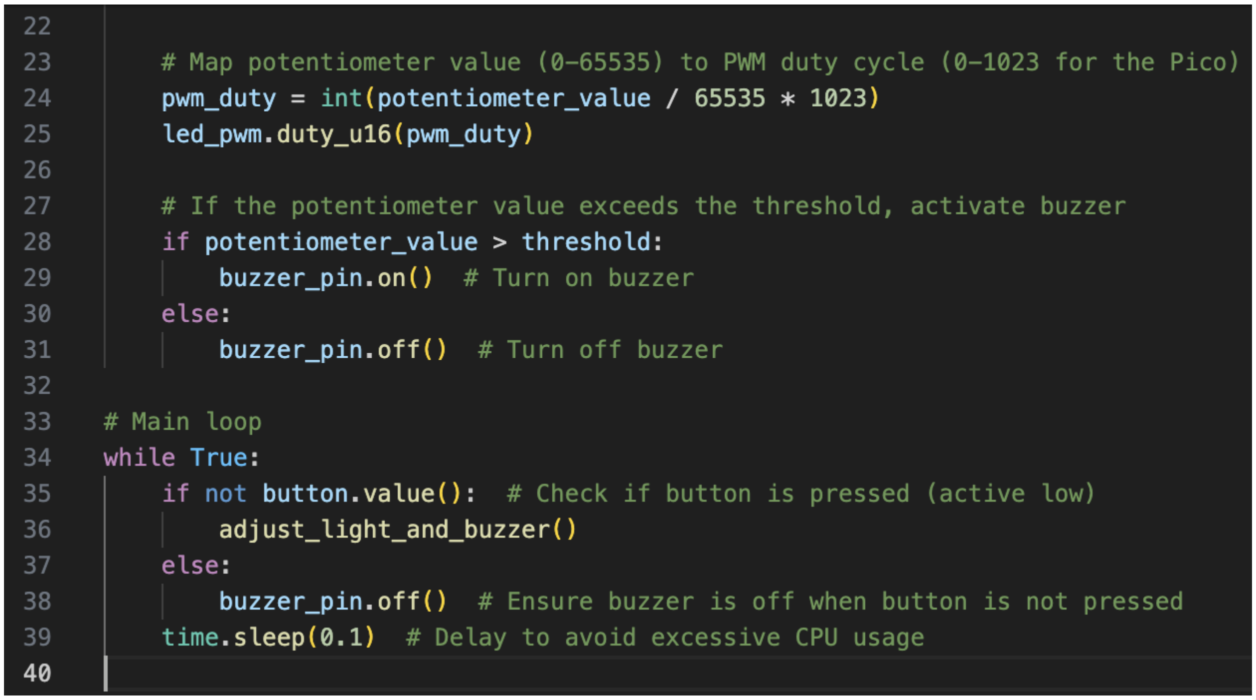
The code for this is:

```
import machine

import time


# Pin setup

potentiometer_pin = machine.ADC(26)  # ADC pin connected to the potentiometer (GP26)

led_pin = machine.Pin(15, machine.Pin.OUT)  # PWM pin connected to LED (GP15)

buzzer_pin = machine.Pin(14, machine.Pin.OUT)  # Pin connected to buzzer (GP14)

button = machine.Pin(17,machine.Pin.IN, machine.Pin.PULL_UP)


# Set up PWM for the LED

led_pwm = machine.PWM(led_pin)

led_pwm.freq(1000)  # Frequency for PWM (adjustable, 1 kHz here)


# Threshold for buzzer activation

threshold = 65534


# Function to read potentiometer and adjust LED brightness

def adjust_light_and_buzzer():

  # Read potentiometer value (0-65535)
```

```
    potentiometer_value = potentiometer_pin.read_u16()
    print("Potentiometer Value:", potentiometer_value)
    # Map potentiometer value (0-65535) to PWM duty cycle (0-1023 for the Pico)
    pwm_duty = int(potentiometer_value / 65535 * 1023)
    led_pwm.duty_u16(pwm_duty)
    # If the potentiometer value exceeds the threshold, activate buzzer
    if potentiometer_value > threshold:
        buzzer_pin.on()  # Turn on buzzer
    else:
        buzzer_pin.off()  # Turn off buzzer

# Main loop
while True:
    if button.value() ==0:  # Check if button is pressed (active low)
        adjust_light_and_buzzer()
    else:
        buzzer_pin.off()  # Ensure buzzer is off when button is not pressed
    time.sleep(0.1)  # Delay to avoid excessive CPU usage
```



```python
import machine
import time

# Pin setup
potentiometer_pin = machine.ADC(26)  # ADC pin connected to the potentiometer (GP26)
led_pin = machine.Pin(15, machine.Pin.OUT)  # PWM pin connected to LED (GP15)
buzzer_pin = machine.Pin(14, machine.Pin.OUT)  # Pin connected to buzzer (GP14)
button = machine.Pin(17,machine.Pin.IN, machine.Pin.PULL_UP)

# Set up PWM for the LED
led_pwm = machine.PWM(led_pin)
led_pwm.freq(1000)  # Frequency for PWM (adjustable, 1 kHz here)

# Threshold for buzzer activation
threshold = 30000

# Function to read potentiometer and adjust LED brightness
def adjust_light_and_buzzer():
    # Read potentiometer value (0-65535)
    potentiometer_value = potentiometer_pin.read_u16()
    print("Potentiometer Value:", potentiometer_value)
```



```python

    # Map potentiometer value (0-65535) to PWM duty cycle (0-1023 for the Pico)
    pwm_duty = int(potentiometer_value / 65535 * 1023)
    led_pwm.duty_u16(pwm_duty)

    # If the potentiometer value exceeds the threshold, activate buzzer
    if potentiometer_value > threshold:
        buzzer_pin.on()  # Turn on buzzer
    else:
        buzzer_pin.off()  # Turn off buzzer

# Main loop
while True:
    if not button.value():  # Check if button is pressed (active low)
        adjust_light_and_buzzer()
    else:
        buzzer_pin.off()  # Ensure buzzer is off when button is not pressed
    time.sleep(0.1)  # Delay to avoid excessive CPU usage

```

References:

SriTu Hobby (2024) *How to control the brightness of the LED bulb using the Raspberry Pi Pico Board*, *SriTu Hobby*. Available at: https://srituhobby.com/how-to-control-the-brightness-of-the-led-bulb-using-the-raspberry-pi-pico-board/?srsltid=AfmBOoqex0hoMkVLnp63dXkhx7D4rlGMIibCBJHvE_acE_Si9_5_h_1- (Accessed: 09 November 2024).

JordanDee (no date) *Pulse width modulation*, *Pulse Width Modulation - SparkFun Learn*. Available at: https://learn.sparkfun.com/tutorials/pulse-width-modulation/all (Accessed: 09 November 2024).

Kattni Rembor and 1 other contributor. Contributors: Trevor Beaton (no date) *Getting started with Raspberry Pi Pico and CircuitPython*, *Adafruit Learning System*. Available at: https://learn.adafruit.com/getting-started-with-raspberry-pi-pico-circuitpython/potentiometer-and-pwm-led (Accessed: 09 November 2024).

Wooter (2018) *Analogread(A2) always at 1023 max value potentiometer arduino pro micro*, *Arduino Forum*. Available at: https://forum.arduino.cc/t/analogread-a2-always-at-1023-max-value-potentiometer-arduino-pro-micro/549802 (Accessed: 09 November 2024).

Geboy, A. *et al.* (2024) *Raspberry pi pico: PWM fading an LED (MicroPython)*, *Random Nerd Tutorials*. Available at: https://randomnerdtutorials.com/raspberry-pi-pico-pwm-micropython/ (Accessed: 09 November 2024).

Help from:

https://Chatgpt.com , helped me error check and check code

Lachlan, my partner.