2023

4. Traffic light controller

Project number: **2021-1-FR01-KA220-SCH-000031617**

Co-funded by the European Union

SCRAPY Partnership

31/05/2023

Table of Contents

Experiment 4: Traffic light controller

Short Description

With this experiment, students will be able to control LED lights that light up in the same order as a traffic light.

Extended Description

In this activity we will use 3 LED in different colors (red, yellow and green) and students will use a push button to restart the circuit.

The Raspberry Pi Pico detects the level change of the button to determine whether the button was pressed. Press the button to turn on the LED light for the first time and press the button to get the red LED to light again, so as to realize the function of turning on and off the LED light as well as timing the lights so they light up one after another.

Objectives:

With this activity, students will experiment with a push button, LED of different colors and timing so that the traffic light runs smoothly.

In terms of knowledge, students will:

1. Understand what a circuit is.
2. Be able to identify the hardware used in a circuit.
3. Build three LEDs (red, yellow, and green) and learn how to code them to light up sequentially.
4. Add a push button to the system and understand how it can work with it.

Materials to be used:

- 1 x Raspberry Pi Pico
- 1 x Pico breadboard kit
- 1 x Full-size breadboard
- 1 x OLED I2C ICC
- 1 x push button
- 3 x LED (red, green, and yellow)
- 1 x Buzzer
- 3 x 220-ohm resistor
- Jumper wires

Steps to be followed:

Let's create a traffic light that moves sequentially and use a push button to restart the circuit.

For this, let's star by

1. Connect the **OLED I2C ICC** to the Raspberry Pi Pico board using connection wires.
2. Connect the push button to the Raspberry Pi Pico board.
3. Connect the buzzer and LEDs to the Raspberry Pi Pico board using connection wires and the 220-ohm resistors to limit the current flow.
4. Write a Python program to control the Raspberry Pi Pico board and use the push button to start/restart the traffic light.
5. Test the push button to see what happens with the LEDs and the OLED I2C ICC.

Raspberry Pi Pico Board:

- GP26: SDA pin of the OLED I2C ICC
- GP27: SCL pin of the OLED I2C ICC
- GP7: Pin 1 of the push button
- GP39: Pin 3 of the push button
- GP13: Positive pin of the red LED
- GP12: Positive pin of the yellow LED
- GP11: Positive pin of the green LED
- GP16: Positive pin of the buzzer
- GND: Ground pin of the board

OLED I2C ICC:

- VCC: Connect to 3V3/5V of Raspberry Pi Pico board
- GND: Connect to GND of Raspberry Pi Pico board
- SCL: Connect to GP27 of Raspberry Pi Pico board
- SDA: Connect to GP26 of Raspberry Pi Pico board

Push button:

- Pin 1: Connect to GP7 of Raspberry Pi Pico board via a 220-ohm resistor
- Pin 3: Connect to 3V3 of Raspberry Pi Pico board

Red LED:

- Positive leg: Connect to GP13 of Raspberry Pi Pico board via a 220-ohm resistor
- Negative leg: Connect to GND of Raspberry Pi Pico board

Yellow LED:

- Positive leg: Connect to GP12 of Raspberry Pi Pico board via a 220-ohm resistor
- Negative leg: Connect to GND of Raspberry Pi Pico board

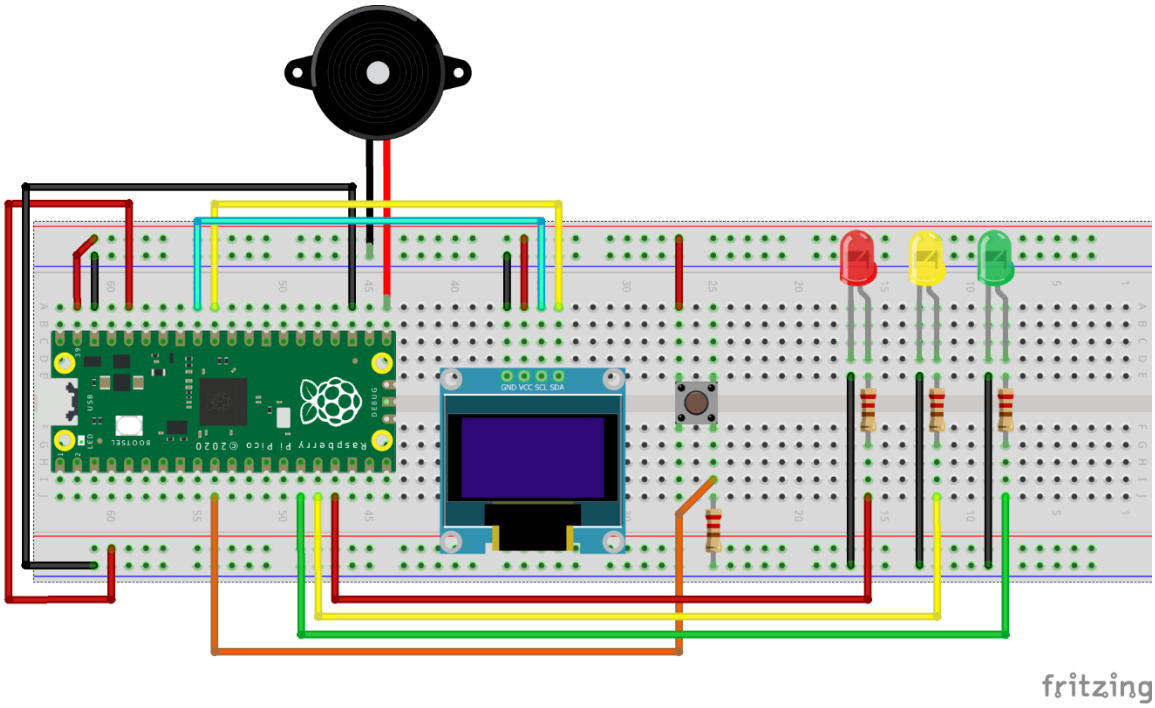
Green LED:

- Positive leg: Connect to GP11 of Raspberry Pi Pico board via a 220-ohm resistor
- Negative leg: Connect to GND of Raspberry Pi Pico board

Buzzer:

- Positive leg: Connect to GP16 of Raspberry Pi Pico board
- Negative leg: Connect to GND of Raspberry Pi Pico board

Wiring diagram



Code

```
import machine
import ssd1306
import utime



# Pin assignments
button_pin = machine.Pin(7, machine.Pin.IN,
machine.Pin.PULL_DOWN)
red_led_pin = machine.Pin(13, machine.Pin.OUT)
yellow_led_pin = machine.Pin(12, machine.Pin.OUT)
green_led_pin = machine.Pin(11, machine.Pin.OUT)
buzzer_pin = machine.Pin(16, machine.Pin.OUT)


# Initialize OLED display
i2c = machine.I2C(0, sda=machine.Pin(0), scl=machine.Pin(1))
oled = ssd1306.SSD1306_I2C(128, 32, i2c)

# Set initial state
is_crossing_allowed = False

def button_interrupt_handler(pin):
    global is_crossing_allowed
    if pin.value() == 1:
        is_crossing_allowed = not is_crossing_allowed

# Register button interrupt
button_pin.irq(trigger=machine.Pin.IRQ_RISING,
handler=button_interrupt_handler)

# Function to update the display
def update_display():
    oled.text("TRAFFIC LIGHT", 0, 0)
    if is_crossing_allowed:
        oled.text("CROSSING:", 0, 12)
        oled.text("ALLOWED", 0, 22)
    else:
        oled.text("PLEASE", 0, 12)
        oled.text("WAIT", 0, 22)
        oled.show()
```

```
# Function to control the traffic light
def control_traffic_light():
    red_led_pin.value(1)
    yellow_led_pin.value(0)
    green_led_pin.value(0)
    buzzer_pin.value(1)
    utime.sleep(4) #control time
    buzzer_pin.value(0)
    red_led_pin.value(0)
    yellow_led_pin.value(0)
    green_led_pin.value(1)
    utime.sleep(4) #control time
    red_led_pin.value(0)
    yellow_led_pin.value(1)
    green_led_pin.value(0)
    utime.sleep(0.5)

# Main loop
while True:
    control_traffic_light()
    update_display()
```

Conclusion

If needed, the teacher can challenge the students to create the code so that the push button will control the color of the LED that lights up and use this program as an assessment tool for the application of the framework created.

This way, the teacher presents a question, and the students respond by lighting up either a red, yellow, or green sticky according to what they think is the answer. With this resource, the teacher can immediately see the results of the student's choice.