COMP6043 Physical Computing Lab 1

Note: if a task asks you to demonstrate your work to your lecturer, you must demonstrate and get your work signed off. Otherwise, no marks will be awarded for your work.

Once completed, upload your report to Canvas.

This lab is based on the "Spaceship Interface" project in the Arduino Starter Kit Projects Book (p.32 – p.41), which is available on Canvas. However, you will be building the circuit using the lab PCBs (Printed Circuit Boards) rather than using your own components and breadboard. The PCB schematic diagram is also available on Canvas in the Labs unit.

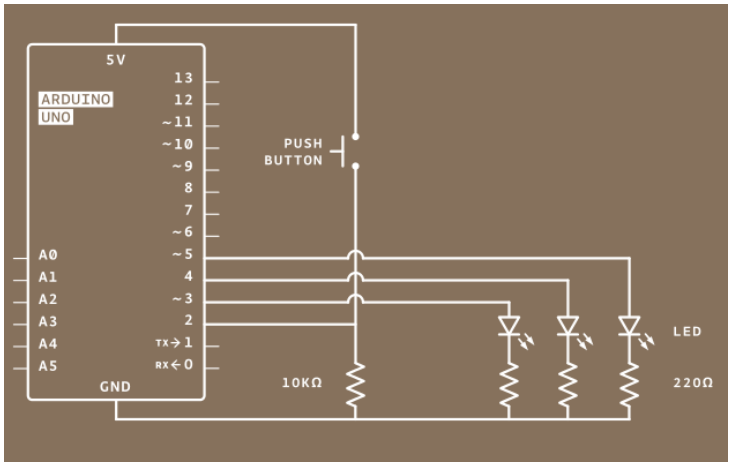
Pay particular attention when connecting the PCB to the Arduino power pins. It is very important that you connect with the correct polarity, i.e. 5 V on the Arduino to 5 V on the PCB, and GND on the Arduino to GND on the PCB. If you wire up with the incorrect polarity, you will burn out the on-board temperature sensor and damage the board.

It is a good idea to colour-code your connections here – if available, use red wires for Vcc (5V) and black for GND.

For your first lab at least, wire up your circuit (Arduino ↔ PCB) **before** connecting the USB cable from the Arduino to the PC. Then call your lecturer to check your circuit before proceeding.

Task 1 [60%]

Wire up your circuit as follows, with a green LED connected to pin 3 of the Arduino and red LEDs connected to pins 4 and 5:



Note: you do not need to wire up the pull-down resistors (for the LEDs and the push-

button switch) yourself, they are already connected on the PCB*.

Open the Arduino IDE and enter the following code:

```
int switchstate = 0;

void setup(){
  // declare the LED pins as outputs
  pinMode(3,OUTPUT);
  pinMode(4,OUTPUT);
  pinMode(5,OUTPUT);

  // declare the switch pin as an input
  pinMode(2,INPUT);
}

void loop(){
  // read the value of the switch
  switchstate = digitalRead(2);

  if (switchstate == LOW) { // button is not pressed
    digitalWrite(3, HIGH); // green LED
    digitalWrite(4, LOW);  // red LED
    digitalWrite(5, LOW);  // red LED
  }
  else {  // button is pressed
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    digitalWrite(5, HIGH);

    delay(250); // wait for a quarter second

    // toggle red LEDs
    digitalWrite(4, HIGH);
    digitalWrite(5, LOW);

    delay(250); // wait for a quarter second
  }
} // go back to the beginning of the loop
```

Upload your code to the Arduino and verify that your project works correctly. When the button is not pressed, the green LED should be on and the red LEDs off. When the

*The circuit diagram here specifies 220 Ω for the LED pull-down resistors but the PCB uses 1 kΩ. This is due to a difference in the LED types but you can ignore this difference for now.

button is pressed, the green LED should be off and the red LEDs should be flashing.

Demonstrate your project to your lecturer once completed.

Task 2 [10%]

When the button is pressed, the red LEDs flash on and off at a frequency of 2 Hz, i.e. two cycles per second or, in other words, on for 0.25 seconds, off for 0.25 seconds (1 cycle per 0.5 seconds).

Modify your project such that, when the button is pressed, the red LEDs flash on and off at a frequency of 1 Hz, i.e. on for 0.5 seconds, off for 0.5 seconds.

Copy and paste the changes made to your code into your report.

Task 3 [10%]

Modify your project such that pressing the button stops the LEDs flashing (all three LEDs should be flashing when the button is not pressed), i.e.

- Button off: all three LEDs flashing.
- Button on: all LEDs off.

Copy and paste your code into your report.

Task 4 [20%]

Modify your project to include an extra push-button switch such that both buttons need to be pressed simultaneously to stop the LEDs flashing. i.e.

- At least one button off: all three LEDs flashing.
- Both buttons on: all LEDs off.

Note that && and || are logical `AND` and `OR` in C++, respectively.

Copy and paste your code into your report.

Demonstrate your project to your lecturer once completed.