Group Number: 07

Name	ID
Safwat Sufia Raida	19101293
Shoeb Islam Hamim	20101337
Shoeb Islam Hamim	20101327
MD Fuad Hasan	20101345

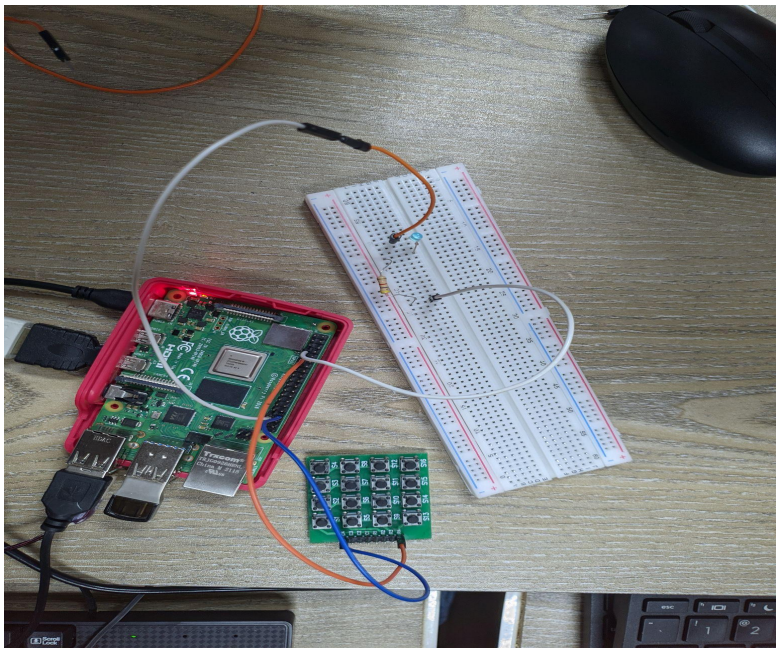
Name of experiment: Introduction to the Raspberry Pi GPIO pins, and using push buttons to control LEDs.

Objective: The objective of this experiment is to get the basic idea on the General Purpose Input-Output, or GPIO, pins of Raspberry Pi, how they can be used to interface with external devices like LEDs and push buttons. Additionally, to learn how to configure and control the GPIO pins on the Raspberry Pi using software tools and Python programming language. Also exploring the use of GPIO pins in unison with push buttons and LEDs to create an elementary user-controlled circuit.

Equipment: The following components were used for this experiment,

- Raspberry Pi 4
- LED
- A resistor of 220 ohms
- Push-button
- Breadboard
- Connecting wires (female to male)

Experimental Setup:



Code:

```
from gpiozero import LED
#imports LED functions from gpiozero library
from gpiozero import Button
#imports Button functions from gpiozero library
led = LED(4)
#declare the GPIO pin 4 for LED output and store it in led variable
button = Button(17)
#declare the GPIO pin 17 for Button output and store it in button variable
while True:
    #initiated an infinite while loop
    button.wait_for_press()
    #use the built-in function of the button to wait till press
    led.on()
    #turn on the led
    button.wait_for_release()
    #use the built-in function of button to wait till release
    led.off()
    #turn off the led
```

Results: After connecting the wires, the resistors, the breadboard, the switch and the Raspberry Pi, when the code is executed, an LED is turned on if we push and hold the switch but after we let go of the switch, the LED turns off. It demonstrates that the test was successful.

Lab Task:

1. We used a 220 ohm resistor in series with the LED to limit the current flow and prevent damage to the LED.
2. The push button is connected to a GPIO pin on the RPI and the ground gnd pin of the RPI, rather than directly to the LED and resistor combination, because this allows us to control the LED using the push button as a switch. When the button is not pressed, the GPIO pin is connected to the 3.3V supply on the RPI and when the button is pressed, the GPIO pin is connected to gnd, which triggers the LED to turn on. If the push button were connected directly to the LED and resistor combination, pressing the button would connect the LED directly to ground, bypassing the resistor and potentially damaging the LED due to the high current flow. Thus, to protect the LED and control it we connect the push button from a GPIO pin on the RPI to the gnd pin of the RPI instead of being connected directly to the LED and the resistor combination.

3. Firstly, the LED would be dimmer, due to the higher resistance lower current will pass through. Additionally, the LED will take longer to turn on and off due to high resistance slowing down the rate at which the LED charges and discharges. Thus, the LED would still turn on and off with the push button, but the brightness and response time would be affected by the change in resistor value.

Discussion: In this experiment, we faced some difficulties such as even after completing the circuit, our LED was not turning on or off. After spending some time troubleshooting it, we discovered that a malfunctioning switch and a faulty wire were to blame. We were able to get the intended outcome after replacing these components.

In conclusion, this experiment shows that RPI's GPIO pins can control external components such as LEDs and switches, a push button switch can be used to control the state of an LED using the RPI's GPIO pins, we can write a python code to control the behavior of the LED as well as the importance of using a current-limiting resistor in the LED circuit to prevent the LED from being damaged due to excessive current flow.