A picture containing text, tableware, plate, dishware

Description automatically generated

**Internet of Things Fundamentals**

**Name** Kanza Kashaf

**Registration NO:** 22-NTU-CS-1350

**Program**  BSAI

**Semester**  6th

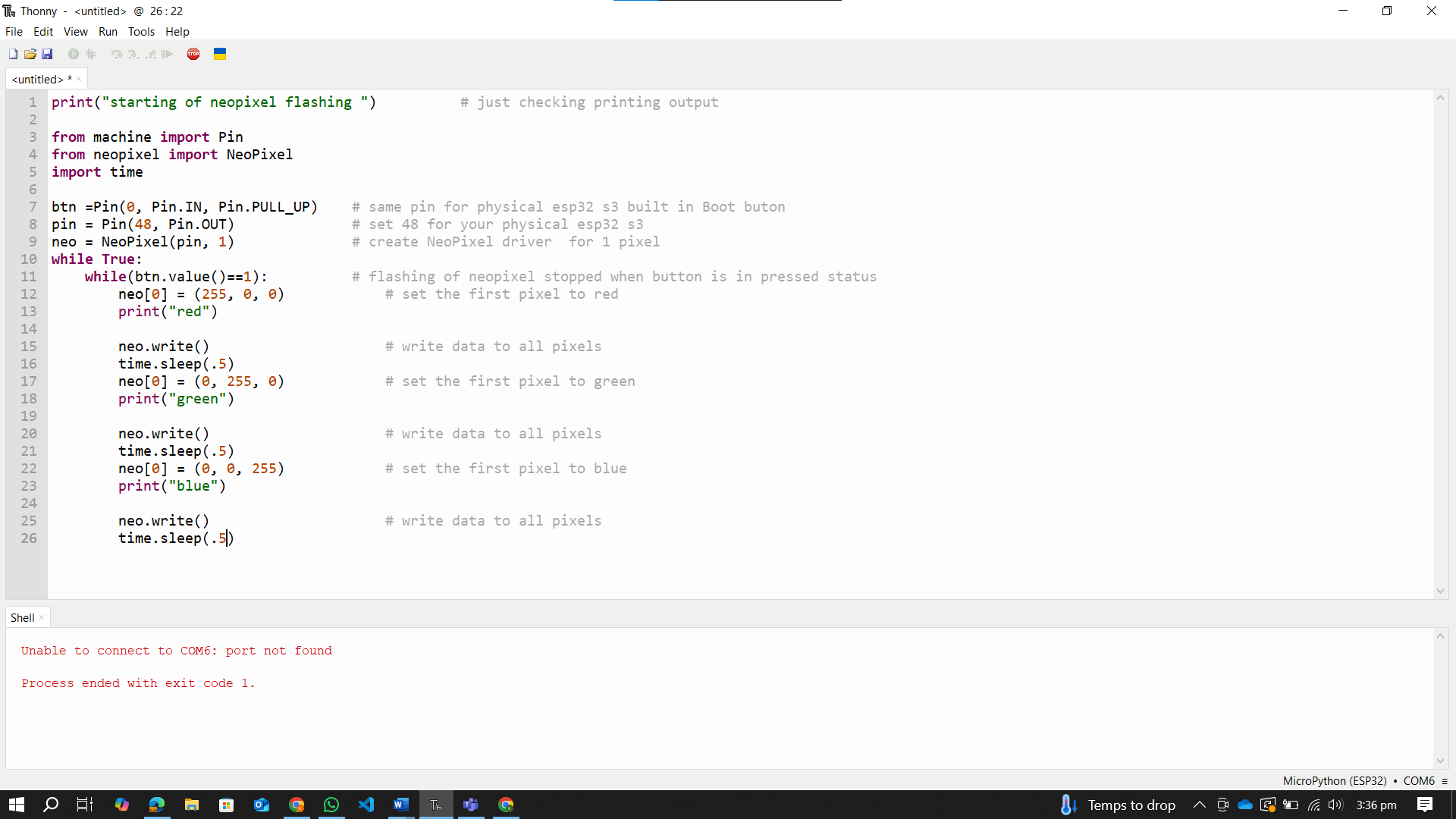
**Department**  Computer Science

**Submitted to** Sir Nasir Mehmood

**Title:**

Lab 2 Home Task

**Task 1:**



I don’t have a physical device (ESP32 S3).

In this code, we simply executing a nested loop, in which we display colors on Neopixel. The loop terminates when the button is pressed. And then the Neopixel shows the blue color as when loops completely it ends on blue color.

**Task 2:**

**2-A:**

The condition in the loop is that when the value of button is 1 then loops execute. And when the value of button is 0 then loops terminate. The value of the button becomes 0 when we press the button. Thus, when we press the button, the loops terminate but the condition checks after the current iteration is completed and the last color that we are showing in the loop is blue. That’s why the Neopixel always turns blue.

**2-B:**

**Code:**

print("starting of neopixel flashing ")          # just checking printing output

from machine import Pin

from neopixel import NeoPixel

import time

btn =Pin(0, Pin.IN, Pin.PULL\_UP)    # same pin for physical esp32 s3 built in Boot buton

pin = Pin(33, Pin.OUT)              # set 48 for your physical esp32 s3

neo = NeoPixel(pin, 1)              # create NeoPixel driver  for 1 pixel

while True:

    while(btn.value()==1):          # flashing of neopixel stopped when button is in pressed status

        neo[0] = (255, 0, 0)            # set the first pixel to red

        neo.write()                     # write data to all pixels

        print("red")

        if btn.value()==0:  break       # if button pressed it will break

        time.sleep(.5)

        neo[0] = (0, 255, 0)            # set the first pixel to green

        neo.write()                     # write data to all pixels

        print("green")

        if btn.value()==0:  break       # if button pressed it will break

        time.sleep(.5)

        neo[0] = (0, 0, 255)            # set the first pixel to blue

        neo.write()                     # write data to all pixels

        print("blue")

        if btn.value()==0:  break       # if button pressed it will break

        time.sleep(.5)

        neo[0] = (255, 0, 0)            # set the first pixel to red

        neo.write()                     # write data to all pixels

        print("red")

I used a break statement to achieve the requirement. After every pixel change, I check the condition, if button is pressed, I break the loop. In this way, we get different colors on pressing button in real time.

**Task 3:**

**3-A:**

**Code:**

print("starting of neopixel flashing ")          # just checking printing output

from machine import Pin

from neopixel import NeoPixel

import time

btn =Pin(0, Pin.IN, Pin.PULL\_UP)    # same pin for physical esp32 s3 built in Boot buton

pin = Pin(33, Pin.OUT)              # set 48 for your physical esp32 s3

neo = NeoPixel(pin, 1)              # create NeoPixel driver  for 1 pixel

press = 0

neo[0] = (255, 0, 0)            # set the first pixel to red

print("red")

neo.write()                     # write data to all pixels

while True:

    while(btn.value()==1):          # flashing of neopixel stopped when button is in pressed status

        if btn.value()==0:

            press += 1

            print(f"Button Press Count: {press}")

            time.sleep(0.3)  # Debounce: Wait to prevent multiple counts

            while btn.value() == 0:

                pass  # Wait until button is released

        if press == 5:

            if neo[0] == (255, 0, 0):

                neo[0] = (0, 255, 0)            # set the first pixel to green

                print("green")

            elif neo[0] == (0, 255, 0):

                neo[0] = (0, 0, 255)            # set the first pixel to blue

                print("blue")

            elif neo[0] == (0, 0, 255):

                neo[0] = (255, 0, 0)            # set the first pixel to red

                print("red")

            neo.write()                     # write data to all pixels

            press = 0

I set the press value to 0 and Neopixel color to red outside the loops. Then in the nested loop, I count the number of presses. When the number of presses is equal to 5, I change the color of Neopixel by using if-else condition. If the color is red change it into green,if green change into blue and if blue change into red. After changing the color, I reset the value of press with 0. Thus, for the next time press counts start with zero. I also used debounce time to prevent the multiple counting of single press. In this way, my code is running and changing color after 5 presses.

**3-B:**

Yes, this code changes the color exactly after 5 time presses if u deal properly with button bonce and press detection time.

**3-C:**

The abnormal behaviour is due to:

* Button bonce. So, we must increase sleep time to avoid counting multiple time in one press.
* Counting another press before realising it. So, we must wait until it released.

**Task 4:**

**Code:**

print("starting of neopixel flashing ")          # just checking printing output

from machine import Pin

from neopixel import NeoPixel

import time

btn =Pin(0, Pin.IN, Pin.PULL\_UP)    # same pin for physical esp32 s3 built in Boot buton

pin = Pin(33, Pin.OUT)              # set 48 for your physical esp32 s3

neo = NeoPixel(pin, 1)              # create NeoPixel driver  for 1 pixel

press = 0                           # press variable to count presses

reverse = False                     # using flag value to reverse the colors

# Interrupt Service Routine (ISR) for button press

def button\_isr(pin):

    while(btn.value()==0):

        global press

        if press == 5:

            press = 0

        press += 1

        print(f"Button Press Count: {press}")

        time.sleep(0.3)  # Debounce: Wait to prevent multiple counts

# Attach the ISR to the button pin

btn.irq(trigger=Pin.IRQ\_FALLING, handler=button\_isr)

while True:

    if press == 5:

        reverse = not reverse           # reversing the flag value

press = 0

    if reverse:                         # checking flag value

        neo[0] = (255, 0, 0)            # set the first pixel to red

        print("red")

        neo.write()                     # write data to all pixels

        time.sleep(0.5)

        neo[0] = (0, 255, 0)            # set the first pixel to green

        print("green")

        neo.write()                     # write data to all pixels

        time.sleep(0.5)

        neo[0] = (0, 0, 255)            # set the first pixel to blue

        print("blue")

        neo.write()                     # write data to all pixels

        time.sleep(0.5)

    else:

        neo[0] = (0, 0, 255)            # set the first pixel to blue

        print("blue")

        neo.write()                     # write data to all pixels

        time.sleep(0.5)

        neo[0] = (0, 255, 0)            # set the first pixel to green

        print("green")

        neo.write()                     # write data to all pixels

        time.sleep(0.5)

        neo[0] = (255, 0, 0)            # set the first pixel to red

        print("red")

        neo.write()                     # write data to all pixels

        time.sleep(0.5)

In this code, I reverse the sequence of colors when the button is pressed for the 5 times. I used interrupt to track the presses of button and the condition that checks the button is presses for the 5 times or not. If pressed for 5 times, reverse the sequence of colors. For this purpose, I used reverse flag variable. In if-else statements, I gave the sequence of colors.