

# MICROCONTROLLER

Milestone 1

Task 1-5

Documentation

<https://github.com/MoezMufti/Microcontroller--Moez-Mufti>

Moez Mufti

moez.mufti@stud.hshl.de

## Table of Contents

1. GitHub: .....	2
Link:.....	2
2. Task 1.....	2
GitHub: .....	2
TinkerCAD Simulation:.....	2
State Machine Diagram.....	3
Code:.....	3
3. Task 2.....	5
GitHub: .....	5
TinkerCAD Simulation:.....	5
Physical Simulation: .....	6
Code:.....	7
Class Machine Diagram:.....	9
State Machine: .....	9
Sequence Diagram: .....	10
4. Task 4:.....	11
Github: .....	11
Polling-Based:.....	11
Interrupt-based: .....	11
5. Task 5.....	12
GitHub: .....	12
TinkerCAD Simulation:.....	12
Physical Simulation: .....	13
Code:.....	14
Master Code (Pedestrian Button):.....	14
Slave Code (Traffic Lights): .....	14

GitHub:

Link:

<https://github.com/MoezMufti/Microcontroller--Moez-Mufti>

The screenshot shows the GitHub repository page for 'Microcontroller--Moez-Mufti'. The repository is public and has 1 branch (main) and 0 tags. The user 'MoezMufti' has 687ac8 - now and 34 commits. The repository contains several files and folders:

File/Folder	Commit Message	Time Ago
Task1	Add files via upload	3 hours ago
Task2	Rename Pedestrian Traffic Light System to Pedestrian Traffic ...	30 minutes ago
Task4	Add files via upload	1 hour ago
Task5	Add files via upload	3 hours ago
Microcontroller Milestone 1 - Documentation....	Add files via upload	now

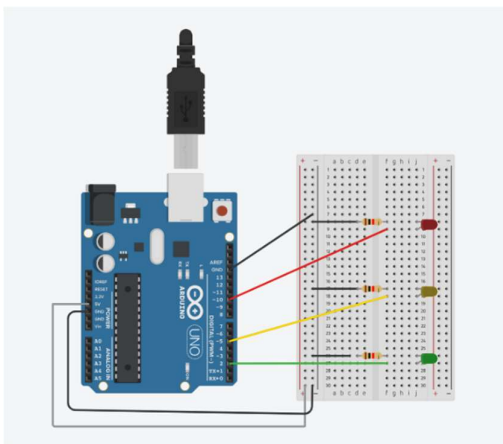
Task 1

GitHub:

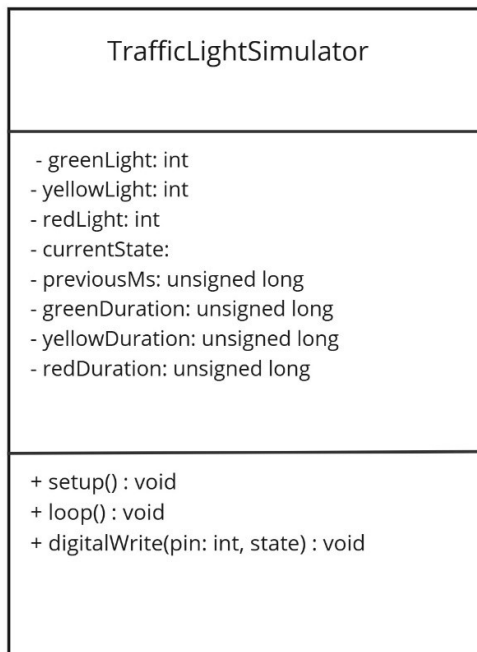
The screenshot shows the GitHub repository page for 'Microcontroller--Moez-Mufti / Task1'. The repository is public and has 1 branch (main) and 0 tags. The user 'MoezMufti' has 6738885 - 1 hour ago and 1 commit. The repository contains several files and folders:

Name	Last commit message	Last commit date
..		
Class Diagram - Task 1.pdf	Add files via upload	1 hour ago
Lab Task 01 - Moez Mufti (1).brd	Add files via upload	1 hour ago
Lab Task 01 - Moez Mufti (1).png	Add files via upload	2 hours ago
Traffic Light Simulator	Update Traffic Light Simulator	1 hour ago

TinkerCAD Simulation:



## State Machine Diagram:



## Code:

```
// Pin delegations
const int greenLight = 2;
const int yellowLight = 3;
const int redLight = 4;

// State definitions
enum TrafficState { Green, Yellow, Red };
TrafficState currentState = Green;

/*setting durations as constant so that they
don't change with with other condition*/
unsigned long previousMs = 0;
const unsigned long greenDuration = 10000; //in ms
const unsigned long yellowDuration = 3000;
const unsigned long redDuration = 5000;

void setup() {
    // Set variables to their modes
    pinMode(greenLight, OUTPUT);
    pinMode(yellowLight, OUTPUT);
    pinMode(redLight, OUTPUT);

    // System always comes back to Green Light
    digitalWrite(greenLight, HIGH);
```

```

    previousMs = millis(); //tells how much ms has passed since beginning
}
void loop() {
    unsigned long currentMs = millis();

    switch (currentState) {
        case Green:
            // Green light is on for 10s
            if (currentMs - previousMs >= greenDuration) {
                //how much time has passed from beginning to current
                digitalWrite(greenLight, LOW);
                digitalWrite(yellowLight, HIGH);
                currentState = Yellow;
                previousMs = currentMs;
            }
            break;

        case Yellow:
            // Yellow light is on for 3s
            if (currentMs - previousMs >= yellowDuration) {
                digitalWrite(yellowLight, LOW);
                digitalWrite(redLight, HIGH);
                currentState = Red;
                previousMs = currentMs;
            }
            break;

        case Red:
            // Red light is on for 5s
            if (currentMs - previousMs >= redDuration) {
                digitalWrite(redLight, LOW);
                digitalWrite(greenLight, HIGH);
                currentState = Green;
                previousMs = currentMs;
            }
            break;
    }
}
}

```

# Task 2

GitHub:

Microcontroller--Moez-Mufti / Task2 /

Add file

MoezMufti

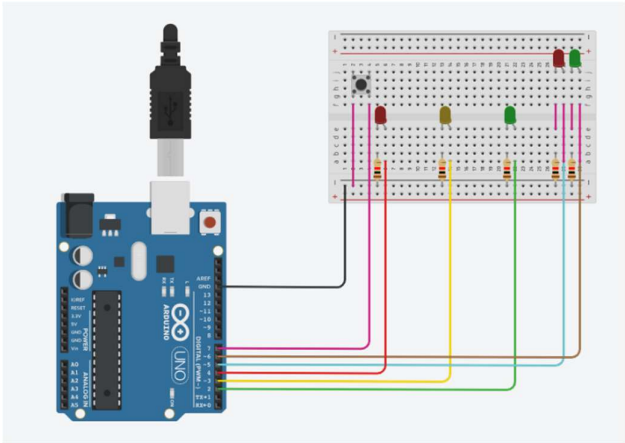
Rename Pedestrian Traffic Light System to Pedestrian Traffic Light Sy...

517254b · now

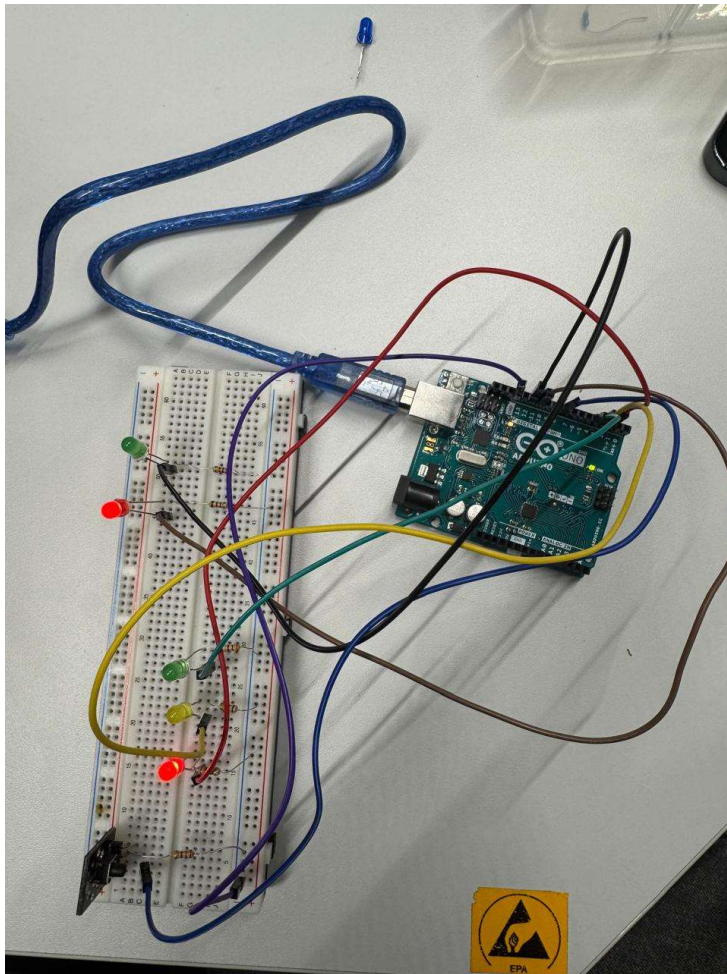
History

Name	Last commit message	Last commit date
..		
Original Traffic Light Simulation (1).brd	Add files via upload	2 hours ago
Original Traffic Light Simulation.png	Add files via upload	2 hours ago
Pedestrian Traffic Light System Code	Rename Pedestrian Traffic Light System to Pedestrian Traffic Light Sy...	now
Pedestrian Traffic Lights Simulator.jpeg	Add files via upload	2 hours ago
T2 - Class Diagram.pdf	Add files via upload	2 hours ago
T2 - Sequence Diagram.pdf	Add files via upload	2 hours ago
T2 - State Machine Diagram.pdf.pdf	Add files via upload	1 hour ago

TinkerCAD Simulation:



Physical Simulation:



Code:

```
// Pin definitions
const int greenLight = 2;
const int yellowLight = 3;
const int redLight = 4;
const int pedRed = 5;
const int pedGreen = 6;
const int button = 7;

// State definitions
enum TrafficState { GREEN, YELLOW, RED, RESET };
TrafficState currentState = GREEN;

//setting timer to zero for initi setup and button to false
unsigned long greenTimer = 0;
bool buttonPressed = false;

void setup() {
    // Set pin AND modes
    pinMode(greenLight, OUTPUT);
    pinMode(yellowLight, OUTPUT);
    pinMode(redLight, OUTPUT);
    pinMode(pedRed, OUTPUT);
    pinMode(pedGreen, OUTPUT);
    pinMode(button, INPUT_PULLUP);

    // Always start and loop back to Green
    digitalWrite(greenLight, HIGH);
    digitalWrite(pedRed, HIGH);
    greenTimer = millis();
}

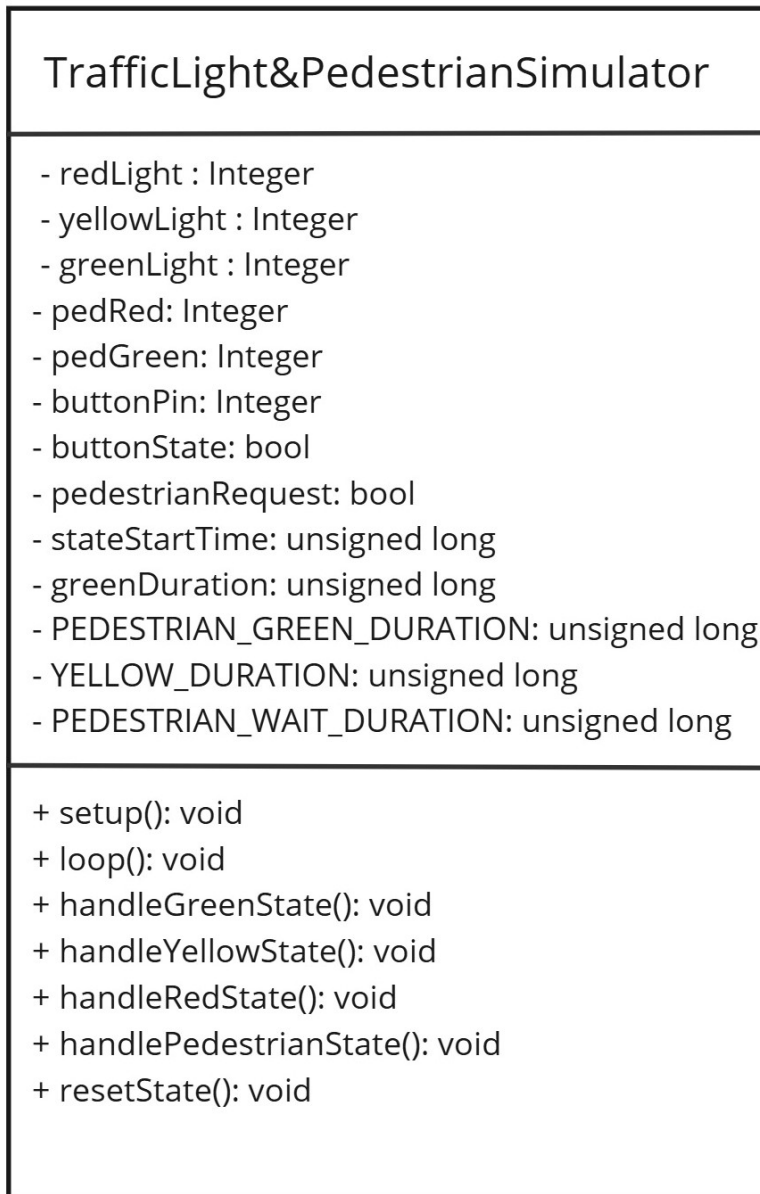
void loop() {
    // Check whether or not the button is pressed
    if (digitalRead(button) == LOW) {
        buttonPressed = true;
    }

    // State machine for traffic light control
    switch (currentState) {
        case GREEN:
            // Stay in GREEN state until 10 seconds pass or button is pressed
            if ((buttonPressed && digitalRead(greenLight) == HIGH) ||
                (millis() - greenTimer >= 10000)) {
                if (buttonPressed) {
                    delay(3000); // Green light will stay on for 3s after button press
                }
                currentState = YELLOW;
            }
        }
    }
}
```

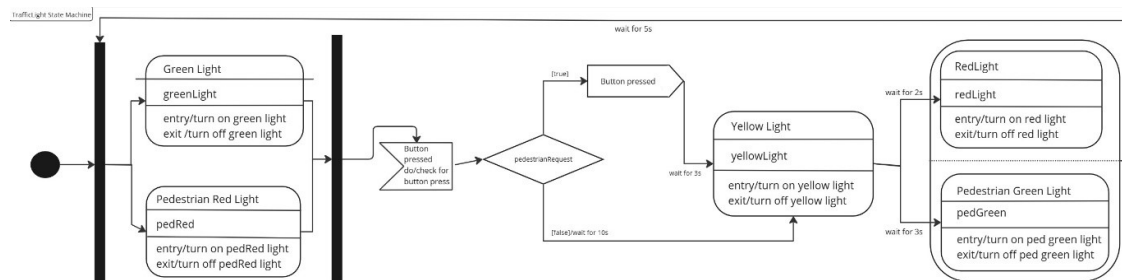


```
    }  
    break;  
  
    case YELLOW:  
        digitalWrite(greenLight, LOW);  
        digitalWrite(yellowLight, HIGH);  
        delay(2000);  
        digitalWrite(yellowLight, LOW);  
        currentState = RED;  
        break;  
  
    case RED:  
        // Red light and pedestrian crossing sequence  
        digitalWrite(redLight, HIGH);  
        digitalWrite(pedRed, LOW);  
        digitalWrite(pedGreen, HIGH);  
        delay(5000);  
        currentState = RESET;  
        break;  
  
    case RESET:  
        // Reset to initial state  
        digitalWrite(pedGreen, LOW);  
        digitalWrite(pedRed, HIGH);  
        digitalWrite(redLight, LOW);  
        digitalWrite(greenLight, HIGH);  
        buttonPressed = false;  
        greenTimer = millis();  
        currentState = GREEN;  
        break;  
    }  
}
```

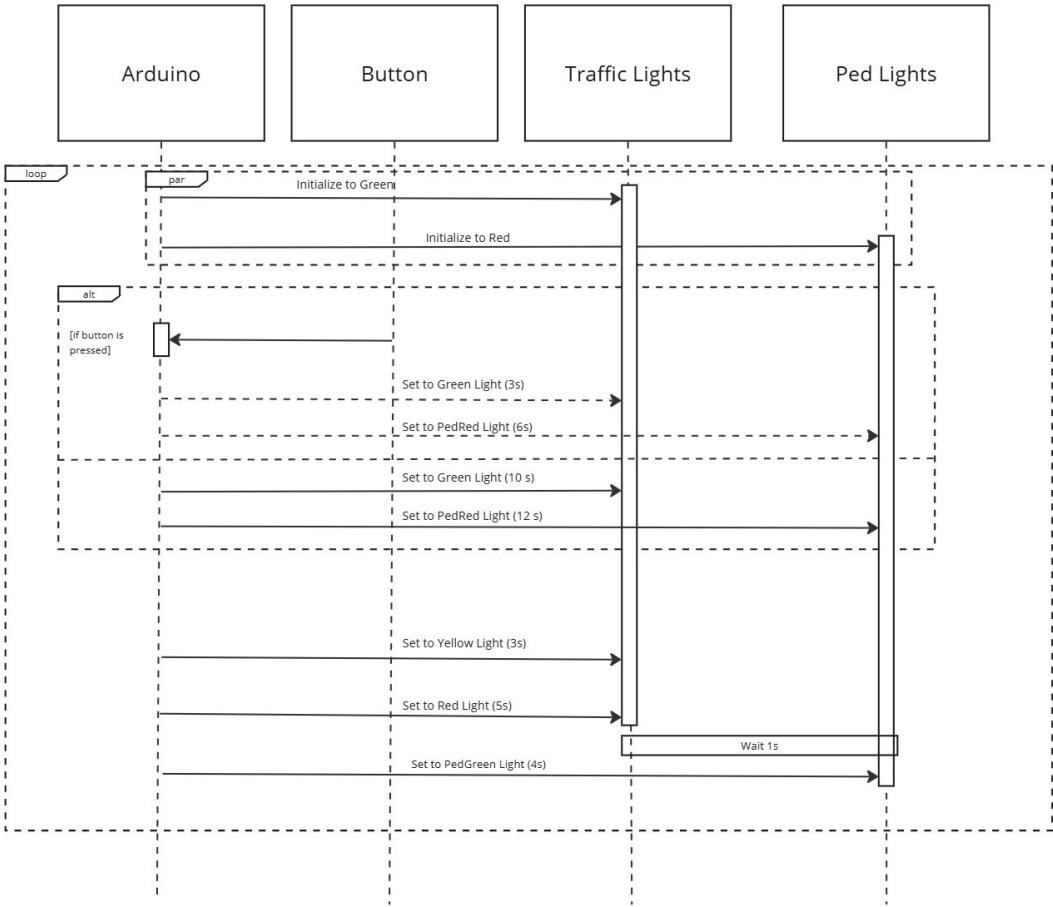
## Class Machine Diagram:



## State Machine:



Sequence Diagram:



# Task 4:

## Github:

Microcontroller--Moez-Mufti / Task4 /

MoezMufti Delete Task4/DDDDD

a44b88f · 1 hour ago History

Name	Last commit message	Last commit date
..		
Difference Based Implemetation.pdf	Add files via upload	1 hour ago
Interrupt of Original Traffic Light Simulation.brd	Add files via upload	1 hour ago
Interrupt of Original Traffic Light Simulation.png	Add files via upload	1 hour ago
Original Traffic Light Simulation (1).brd	Add files via upload	1 hour ago
Original Traffic Light Simulation.png	Add files via upload	1 hour ago

## Polling-Based:

```
13 //setting timer to zero for initi setup and button to raise
14 unsigned long greenTimer = 0;
15 bool buttonPressed = false;
16
17 void setup() {
18   // Set pin AND modes
19   pinMode(greenLight, OUTPUT);
20   pinMode(yellowLight, OUTPUT);
21   pinMode(redLight, OUTPUT);
22   pinMode(pedRed, OUTPUT);
23   pinMode(pedGreen, OUTPUT);
24   pinMode(button, INPUT_PULLUP);
25
26   // Always start and loop back to Green
27   digitalWrite(greenLight, HIGH);
28   digitalWrite(pedRed, HIGH);
29   greenTimer = millis();
30 }
31
32 void loop() {
33   // Check whether or not the button is pressed
34   if (digitalRead(button) == LOW) {
35     buttonPressed = true;
36   }
37
38   // State machine for traffic light control
39   switch (currentState) {
40     case GREEN:
41       // Stay in GREEN state until 10 seconds pass or button is
42       if ((buttonPressed && digitalRead(greenLight) == HIGH) ||
43           (millis() - greenTimer >= 10000)) {
44         if (buttonPressed) {
45           // ...
46         }
47       }
48     }
49 }
```

## Interrupt-based:

```
41 // Yellow light sequence
42 digitalWrite(greenLight, LOW);
43 digitalWrite(yellowLight, HIGH);
44 delay(2000);
45
46 // Red light and pedestrian crossing sequence
47 digitalWrite(yellowLight, LOW);
48 digitalWrite(redLight, HIGH);
49 digitalWrite(pedRed, LOW);
50 delay(2000); // Wait before pedestrian green
51 digitalWrite(pedGreen, HIGH);
52 delay(5000); // Pedestrian crossing duration
53
54 // Reset to initial state
55 digitalWrite(pedGreen, LOW);
56 digitalWrite(pedRed, HIGH);
57 digitalWrite(redLight, LOW);
58 digitalWrite(greenLight, HIGH);
59
60 // Reset state
61 buttonPressed = false;
62 greenTimer = millis(); // Restart green light timer
63 }
64
65 // Interrupt Service Routine (ISR) for button press
66 void buttonISR() {
67   buttonPressed = true; // Set flag to true on button press
68 }
69
70
71
```

# Task 5

GitHub:

Microcontroller--Moez-Mufti / Task5 /

Add file

MoezMufti

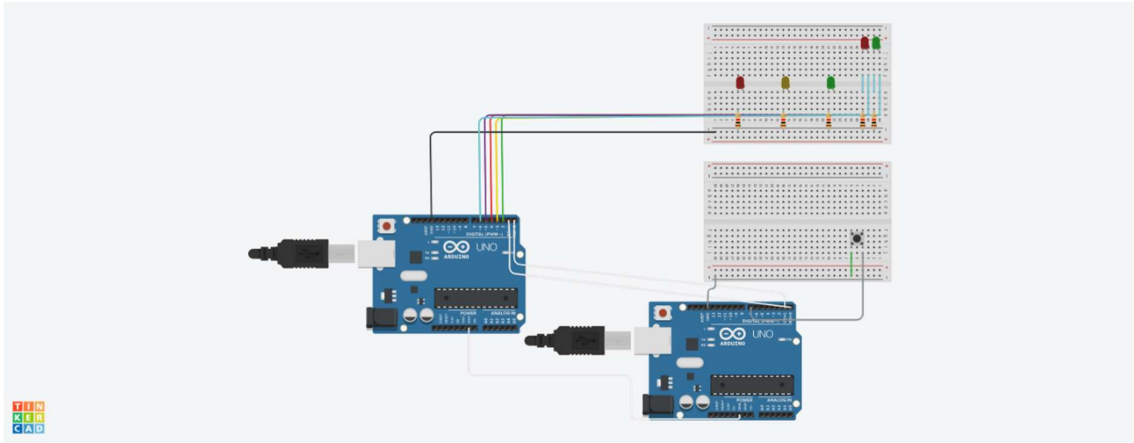
Add files via upload

1a90c1f · 1 hour ago

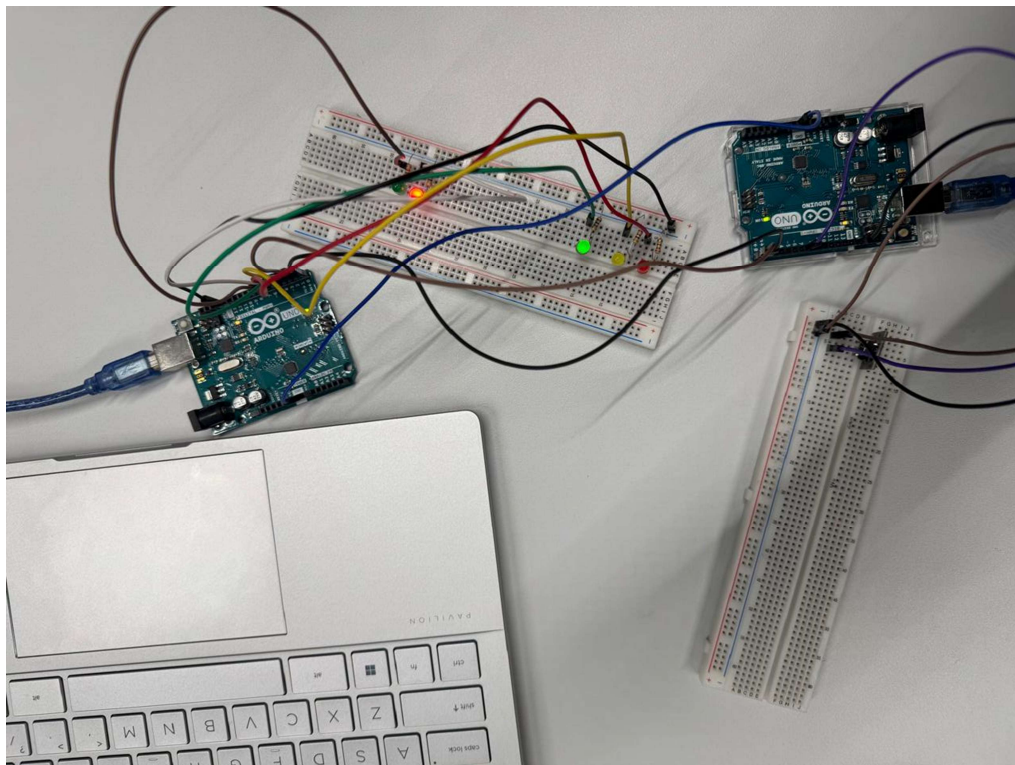
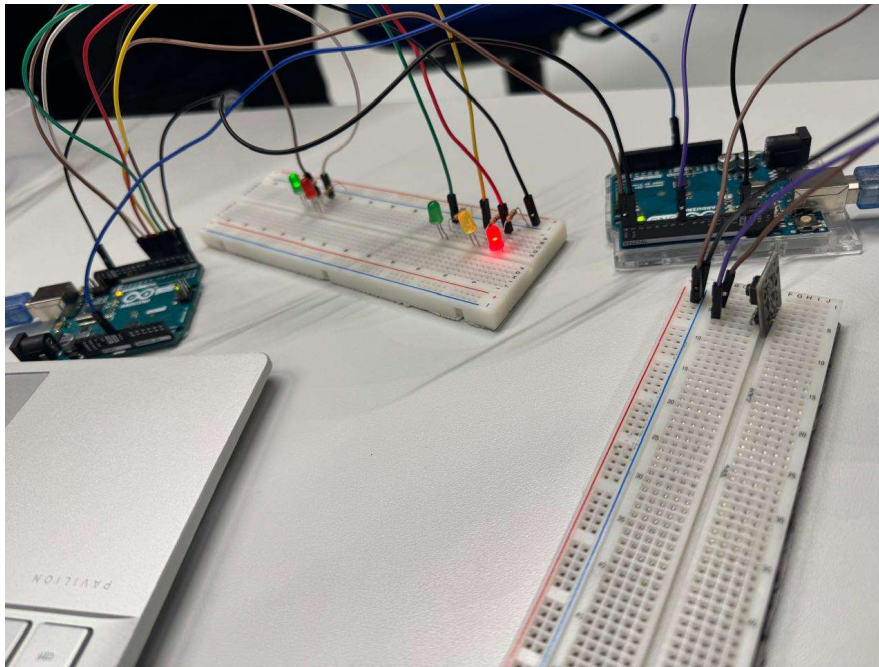
History

Name	Last commit message	Last commit date
..		
Code for Pedestrian Button	Create Code for Pedestrian Button	1 hour ago
Code for Traffic Lights	Create Code for Traffic Lights	1 hour ago
P2P Communication BW Traffic and Pedestrian - Physical Simula...	Add files via upload	1 hour ago
P2P Communication BW Traffic and Pedestrian - Physical Simula...	Add files via upload	1 hour ago
P2P Communication BW Traffic and Pedestrian - Physical Simula...	Add files via upload	1 hour ago
P2P Communication BW Traffic and Pedestrian.brd	Add files via upload	1 hour ago
P2P Communication BW Traffic and Pedestrian.png	Add files via upload	1 hour ago

TinkerCAD Simulation:



## Physical Simulation:



Code:

Master Code (Pedestrian Button):

```
const int buttonPin = 7;
int buttonState = 0;

void setup() {
  Serial.begin(9600); // Start serial communication
  pinMode(buttonPin, INPUT_PULLUP); // Use internal pull-up resistor for the
  button
}

void loop() {
  buttonState = digitalRead(buttonPin);

  if (buttonState == LOW) { // if Button is pressed
    Serial.println(1); // Send signal "1" to the Slave to communicate
    delay(500); // Debounce delay
  }
}
```

Slave Code (Traffic Lights):

```
const int greenLight = 2;
const int yellowLight = 3;
const int redLight = 4;
const int pedRed = 5;
const int pedGreen = 6;

enum TrafficState { GREEN, YELLOW, RED, PEDESTRIAN, RESET };
TrafficState currentState = GREEN;

bool pedestrianRequest = false;
unsigned long stateStartTime = 0;
unsigned long greenDuration = 10000; // Default green light duration
const unsigned long PEDESTRIAN_GREEN_DURATION = 5000; //ms
const unsigned long YELLOW_DURATION = 3000;
const unsigned long PEDESTRIAN_WAIT_DURATION = 1000;

void setup() {
  Serial.begin(9600);
  pinMode(greenLight, OUTPUT);
  pinMode(yellowLight, OUTPUT);
  pinMode(redLight, OUTPUT);
  pinMode(pedRed, OUTPUT);
  pinMode(pedGreen, OUTPUT);

  // Start with green light and pedestrian red
  digitalWrite(greenLight, HIGH);
```



```

    digitalWrite(pedRed, HIGH);
    stateStartTime = millis();
}

void loop() {
    // Check for pedestrian request via serial input
    if (Serial.available() > 0) {
        char signal = Serial.read();
        if (signal == '1') {
            pedestrianRequest = true;
            // Adjust green duration to extend by 3 seconds from the moment of
button press
            unsigned long currentTime = millis();
            if (currentTime - stateStartTime < greenDuration - 3000) {
                greenDuration = (currentTime - stateStartTime) + 3000;
            }
        }
    }

    unsigned long currentTime = millis();

    switch (currentState) {
        case GREEN:
            if (currentTime - stateStartTime >= greenDuration) {
                currentState = YELLOW;
                stateStartTime = currentTime;
            }
            break;

        case YELLOW:
            digitalWrite(greenLight, LOW);
            digitalWrite(yellowLight, HIGH);
            if (currentTime - stateStartTime >= YELLOW_DURATION) {
                digitalWrite(yellowLight, LOW);
                digitalWrite(redLight, HIGH);
                currentState = RED;
                stateStartTime = currentTime;
            }
            break;

        case RED:
            // Handle pedestrian request in RED state
            if (pedestrianRequest) {
                if (currentTime - stateStartTime >= PEDESTRIAN_WAIT_DURATION) {
                    digitalWrite(pedRed, LOW);
                    digitalWrite(pedGreen, HIGH);
                    currentState = PEDESTRIAN;
                    stateStartTime = currentTime;
                }
            }
        }
    }
}

```



```

    }
} else {
    currentState = RESET;
    stateStartTime = currentTime;
}
break;

case PEDESTRIAN:
    if (currentTime - stateStartTime >= PEDESTRIAN_GREEN_DURATION) {
        digitalWrite(pedGreen, LOW);
        digitalWrite(pedRed, HIGH);
        pedestrianRequest = false; // Reset pedestrian request
        currentState = RESET;
        stateStartTime = currentTime;
    }
    break;

case RESET:
    digitalWrite(redLight, LOW);
    digitalWrite(greenLight, HIGH);
    greenDuration = 10000; // Reset green duration to default timing
    currentState = GREEN;
    stateStartTime = currentTime;
    break;
}
}

```