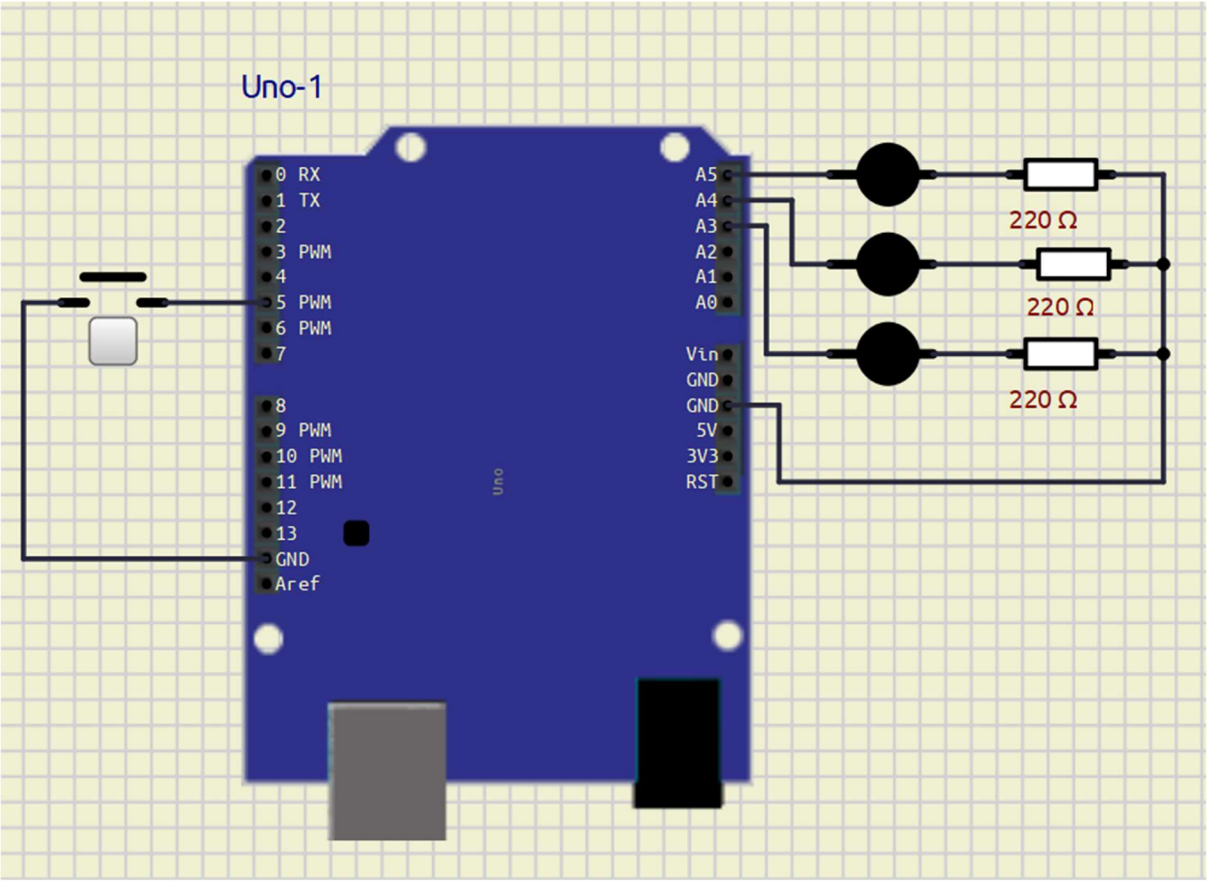
روان سطوحي محمد سطوحي
CS2



# Traffic Light Controller

## Project Overview

This project implements an intelligent traffic light controller with a pedestrian crossing request system. The system maintains a standard traffic sequence (Green → Yellow → Red) but extends the red light duration when a pedestrian presses the request button.

## System Requirements

- **Normal Sequence:**
  - Green Light: 8 seconds
  - Yellow Light: 2 seconds
  - Red Light: 5 seconds
- **With Pedestrian Request:**
  - If button pressed during Green phase
  - System completes Yellow phase
  - Red light extends to 9 seconds (5 + 4 extra seconds)

## Circuit Components

| Component | Quantity | Specification |
|---|---|---|
| Arduino Uno | 1 | Microcontroller |
| Red LED | 1 | Traffic signal |
| Yellow LED | 1 | Traffic signal |
| Green LED | 1 | Traffic signal |
| Resistor | 3 | 220Ω (for LEDs) |
| Push Button | 1 | Pedestrian request |
| Ground | 4 | Common ground |

## Circuit Diagram

**Connections:**

Pin 13 (Arduino) → 220Ω Resistor → Red LED (+) → Red LED (-) → GND

Pin 12 (Arduino) → 220Ω Resistor → Yellow LED (+) → Yellow LED (-) → GND

Pin 11 (Arduino) → 220Ω Resistor → Green LED (+) → Green LED (-) → GND

Pin 2 (Arduino) → Push Button → GND

## Code Explanation

The system uses a **state machine** approach with three states:

- **State 0 (Green):** Green LED is ON for 8 seconds
- **State 1 (Yellow):** Yellow LED is ON for 2 seconds
- **State 2 (Red):** Red LED is ON for 5 or 9 seconds (depending on pedestrian request)

**Key Variables:**

- state: Tracks current traffic light phase (0, 1, or 2)
- pedestrianRequest: Boolean flag set to true when button is pressed
- previousMillis: Stores timestamp for timing control
- millis(): Returns milliseconds since Arduino started (used for non-blocking timing)

**Logic Flow:**

1. Button press sets pedestrianRequest = true
2. System continues normal sequence (doesn't interrupt immediately)
3. When Red phase is reached, duration becomes 9000ms instead of 5000ms
4. After extended red phase, pedestrianRequest resets to false

## Arduino Code

```
// Pin Definitions
const int redLight = 13;
const int yellowLight = 12;
const int greenLight = 11;
const int buttonPin = 2;

// Variables
bool pedestrianRequest = false;
unsigned long previousMillis = 0;
int state = 0; // 0=Green, 1=Yellow, 2=Red

void setup() {
  pinMode(redLight, OUTPUT);
  pinMode(yellowLight, OUTPUT);
  pinMode(greenLight, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);

  Serial.begin(9600);
}

void loop() {
  // Read button state
  if (digitalRead(buttonPin) == LOW) {
    pedestrianRequest = true;
    Serial.println("Pedestrian request received!");
  }

  unsigned long currentMillis = millis();
```

```
switch(state) {
  case 0: // Green Phase (8 seconds)
    digitalWrite(greenLight, HIGH);
    digitalWrite(yellowLight, LOW);
    digitalWrite(redLight, LOW);

    if (currentMillis - previousMillis >= 8000) {
      state = 1;
      previousMillis = currentMillis;
      Serial.println("Switching to Yellow");
    }
    break;

  case 1: // Yellow Phase (2 seconds)
    digitalWrite(greenLight, LOW);
    digitalWrite(yellowLight, HIGH);
    digitalWrite(redLight, LOW);

    if (currentMillis - previousMillis >= 2000) {
      state = 2;
      previousMillis = currentMillis;
      Serial.println("Switching to Red");
    }
    break;

  case 2: // Red Phase (5 or 9 seconds)
    digitalWrite(greenLight, LOW);
    digitalWrite(yellowLight, LOW);
    digitalWrite(redLight, HIGH);

    int redDuration = pedestrianRequest ? 9000 : 5000;
```

```
    if (currentMillis - previousMillis >= redDuration) {
      state = 0;
      previousMillis = currentMillis;
      pedestrianRequest = false;
      Serial.println("Switching to Green");
    }
    break;
  }
}
```

## Testing Results

The system was tested using SimulIDE simulation software. Test results:

- Normal sequence operates correctly (Green 8s → Yellow 2s → Red 5s)
- Pedestrian button press is detected successfully
- Red light extends to 9 seconds when button is pressed during green phase
- System returns to normal operation after extended red phase
- Timing accuracy verified using stopwatch

## Conclusion

The traffic light controller successfully implements all required features. The system maintains precise timing while safely accommodating pedestrian crossing requests. The state machine approach ensures smooth transitions and prevents unsafe interruptions of the traffic sequence.