JSS MAHAVIDYAPEETA

JSS SCIENCE AND TECHNOLOGY UNIVERSITY

SRI JAYACHAMARAJENDRA COLLEGE OF ENGINEERING

MYSURU-570006

Subject name: Internet of Things (20CS630)
Event – 4

A REPORT ON

"FOUR-WAY TRAFFIC CONTROL SYSTEM".

Submitted by

Roll No.	USN	NAME
17	01JCE21CS082	SANCHANA SHEKAR
29	01JST21CS024	AVIN SKANDA B N
13	01JCE21CS058	M V NAGAVARDHAN VASIST

Submitted to

Prof. SHWETHASHREE G C
ASSISTANT PROFESSOR
Department of CSE
JSS S&TU, SJCE, Mysuru

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SRI JAYACHAMARAJENDRA COLLEGE OF ENGINEERING
JSS SCIENCE AND TECHNOLOGY UNIVERSITY

MYSURU-570006

2024-2025

CONTENTS

INTRODUCTION

Traffic lights are an integral part of the world's transportation systems. Over the years a number of different algorithms regarding traffic lights have been developed. The algorithm being used at any place for the purpose of controlling traffic takes into account of various factors, such as number of lanes, people that cross a certain road, etc. The most common usage of traffic lights is to control the flow of traffic, which means providing a steady flow for people to go about their daily business on the road. Traffic lights help reduce accidents by a large margin since they allow the flow of vehicles in only one direction at a time. Traffic lights also help in avoiding traffic jams. The most common traffic light pattern being used in the world today is a 4-way traffic control that accounts for pedestrians as well. This sort of pattern is used in main city blocks and squares since these possess both vehicular traffic as well as pedestrian traffic. Traffic lights have a universal color understanding that red light signals for the traffic to stop, yellow light serves as a transition light from going to stop and vice versa.

Importance of 4-Way Traffic Lights

- Safety: 4-way traffic lights reduce intersection accidents by organizing traffic streams.
- Efficient Flow: Traffic lights prevent gridlocks and maintain smooth traffic transitions.
- Pedestrian Management: Integrated pedestrian signals ensure safe crossing in urban areas.
- Economic Impact: 4-way lights decrease congestion costs by enhancing traffic efficiency.

Objectives of 4-Way Traffic Lights

- Clear Communication: Traffic lights use colors and symbols to give straightforward instructions to drivers, cyclists, and pedestrians.
- Adaptive Control: Advanced traffic lights adjust their timings using sensors and data to best manage the flow of vehicles and pedestrians in real-time.
- Integration with Smart Systems: 4-way lights connect with modern city technologies, ensuring efficient coordination with public transport and urban planning tools.
- Sustainability: By minimizing vehicle waiting times and promoting eco-friendly transit options, 4-way lights support environmental conservation in urban areas.

The 4-way traffic lights are more than mere signaling devices, they are intricate systems designed to enhance safety, optimize traffic flow, & foster sustainable urban development. As cities continue to grow & evolve, the importance of efficient and adaptive trafficmanagement becomes increasingly pronounced, underscoring the enduring relevance of 4-waytraffic.

LITREATURE REVIEW

1. Assessment of Arduino-Based Traffic Signal Simulations (2023):

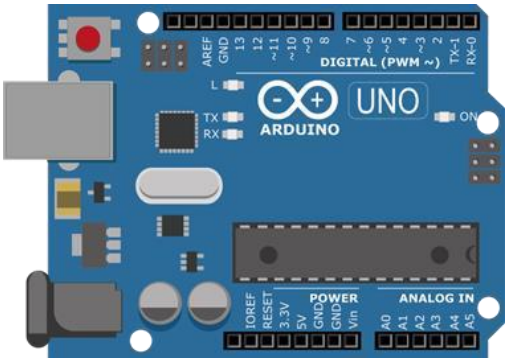
This paper evaluates Arduino-based traffic signal simulations for basic traffic scenarios. It primarily focuses on simulating single intersections with limited traffic light configurations. While it demonstrates the feasibility of using Arduino for traffic signal control, it lacks sophistication in simulating complex traffic interactions.

2.Exploring Basic IoT Applications in Traffic Management (2022):

This paper introduces simple IoT applications in traffic management, focusing on basic sensor deployments and data collection. It discusses the use of IoT sensors at intersections to monitor vehicle presence and traffic flow. In contrast, 4-way traffic scenario using Arduino microcontrollers, offering a more comprehensive approach to traffic light control and simulation.

PROPOSED METHOD

ARDUINO UNO:



Arduino Uno, is built around the ATmega2560 microcontroller. This board, larger in size than its predecessor, offers an impressive 54 digital I/O pins, 16 analog inputs, and 4 UARTs. Operating at a clock speed of 16 MHz, it houses 256 KB of flash memory, 8 KB of SRAM, and 4 KB of EEPROM. With a 5V operating voltage, the Mega is a powerhouse suitable for intricate projects, especially in robotics, automation, and data acquisition. Its compatibility with a vast range of shields and the Arduino IDE further amplifies its versatility and appeal to electronics enthusiasts and professionals alike.

TRAFFIC LIGHTS:

The traffic lights are pivotal components emulating real-world intersections. Each intersection comprises a set of vertically arranged red, yellow, and green LEDs, representing stop, prepare, and go signals, respectively, for both horizontal and vertical traffic directions. Controlled by the Arduino code, these LEDs illuminate in sequences to simulate the flow and coordination of traffic.

1. Declaration Section:

The declaration section in Arduino code is where variables and constants are defined. These variables and constants store data that can be used and manipulated throughout the program. Declarations give the programmer a structured way to specify what kind of data a variable will hold and how it will be used.

For example:

```
int ledPin = 13;      // Declare a variable 'ledPin' of type integer and assign it the value 13.
const int buttonPin = 2;  // Declare a constant variable 'buttonPin' that cannot be changed later.
```

2. Setup Section:

The setup() function is called when the Arduino is powered on or reset. It is used to initialize variables, pin modes, and other settings that only need to be done once at the beginning of the program. This function runs once and prepares the Arduino for the main program execution.

For example:

```
void setup() {
  pinMode(ledPin, OUTPUT); // Set the ledPin as an output.
  pinMode(buttonPin, INPUT); // Set the buttonPin as an input.
}
```

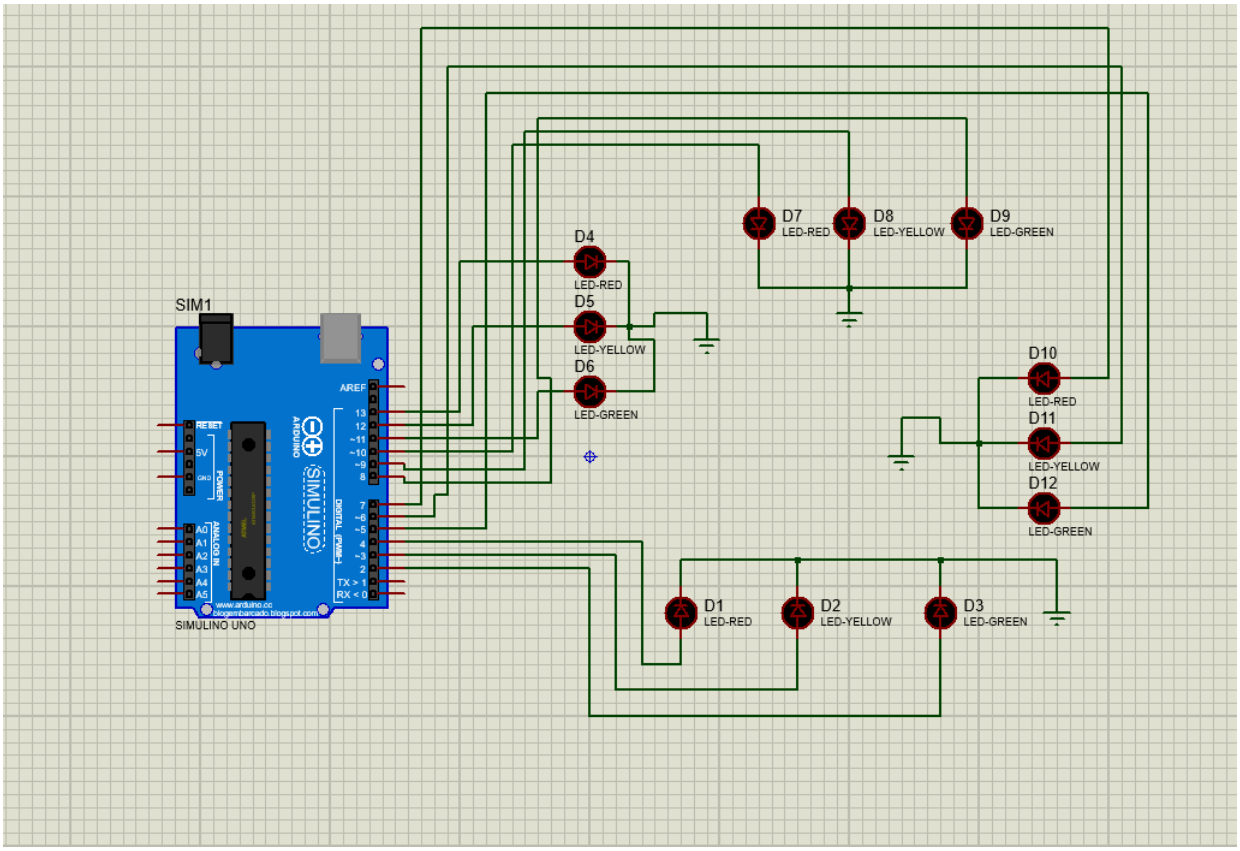
3. Loop Section:

The loop() function is the core of most Arduino sketches. Once the `setup()` function has completed its tasks, the `loop()` function continuously executes, running the main logic of the program in a loop indefinitely until the Arduino is powered off or reset.

For example:

```
void loop() {
  int buttonState = digitalRead(buttonPin); // Read the state of the button.
  if (buttonState == HIGH) { // Check if the button is pressed.
  digitalWrite(ledPin, HIGH); // Turn on the LED.
  } else {
    digitalWrite(ledPin, LOW);  // Turn off the LED.
  }
  delay(100);  // Delay for 100 milliseconds. }
```

DESIGN



To set up a basic 4-way traffic system using an Arduino Uno, you'll need three LEDs for each traffic signal (red, yellow, green) and the Arduino board.By connecting each LED to the Arduino's digital pins, specifying which pins control which colors.

Then, write a simple Arduino sketch to manage the traffic flow. Your code should include timed sequences for each signal, ensuring smooth transitions between red, yellow, and green lights for each direction of traffic.

And to account for delays and synchronization between signals to prevent collisions. Testing and iterating on your code will refine the timing and functionality of your traffic system.

IMPLEMENTATION

STEP 1: Assemble all the components of the project

STEP 2: Place the components and make connections as given below

STEP 3: Make the connection as follows:

- Connect 0,1 and 2 digital pins of Arduino to red, yellow and green of traffic light 1 respectively.
- Connect 3,4 and 5 digital pins of Arduino to red, yellow and green of traffic light 2 respectively.
- Connect 6,7 and 8 digital pins of Arduino to red, yellow and green of traffic light 3 respectively.
- Connect 9,10 and 11 digital pins of Arduino to red, yellow and green of traffic light 4 respectively.
- Ground the negative terminals of all LED.

STEP 4: Upload the Arduino code to Arduino UNO for the 4-way traffic light.

WORKING OF CODE:

Declaration Section:

The first step in the code is the declaration of variables that we will utilize in our program. At first is the declaration of traffic lights and setting them up with their respective pins of Arduino board.

Void Setup:

This part of the code along with the declaration part is run only once, we have to define output and input pins. This helps Arduino to understand which pins to take data from and which pins to write data on. Since there is no input, we have to define traffic lights and pedestrian lights as output pins.

Void loop():

This part of the code runs in a loop consistently and is used to write the main section of the code.

In the first section, we have to turn on the green light of signal 1 while all other signals are red.

After a delay of 2000ms, we have to turn on the yellow light for signal 1 and signal 2 to indicate that a transition from signal 1 to signal 2 will be made shortly.

After a delay of 1000ms, all traffic lights will turn off for 100ms.

The second signal, we have to turn on the green light of signal 2 while all other signals are red.

After a delay of 2000ms, we have to turn on the yellow light for signal 2 and signal 3 to indicate that a transition from signal 2 to signal 3 will be made shortly.

After a delay of 1000ms, all traffic lights will turn off for 100ms.

In signal 3, we have to turn on the green light of signal 3 while all other signals are red.

After a delay of 2000ms, we have to turn on the yellow light for signal 3 and signal 4 to indicate that a transition from signal 3 to signal 4 will be made shortly.

After a delay of 1000ms, all traffic lights will turn off for 100ms.

In the final signal, we have to turn on the green light of signal 4 while all other signals are red.

After a delay of 2000ms, we have to turn on the yellow light for signal 4 and signal 1 to indicate that a transition from signal 4 to signal 1 will be made shortly. This will complete the loop and the sequence will keep running on its own.

After a delay of 1000ms, all traffic lights will turn off for 100ms.

CODE:

//DECLARATION SECTION

```
int red1=0;

int yellow1=1;

int green1=2;

int red2=3;

int yellow2=4;

int green2=5;

int red3=6;

int yellow3=7;

int green3=8;

int red4=9;

int yellow4=10;

int green4=11;
```

//VOID SETUP() SECTION

```
void setup () {

pinMode (red1, OUTPUT);

pinMode(green1, OUTPUT);

pinMode(yellow1, OUTPUT);

pinMode (red2, OUTPUT);

pinMode(green2, OUTPUT);

pinMode(yellow2, OUTPUT);

pinMode (red3, OUTPUT);
```

```
pinMode(green3, OUTPUT);
pinMode(yellow3, OUTPUT);
pinMode (red4, OUTPUT);
pinMode(green4, OUTPUT);
pinMode(yellow4, OUTPUT);
}
```

//VOID LOOP() SECTION

```
void loop() {
digitalWrite(green1, HIGH);
digitalWrite(red2, HIGH);
digitalWrite(red3, HIGH);
digitalWrite(red4, HIGH);
delay(2000);
digitalWrite(yellow1, HIGH);
digitalWrite(yellow2, HIGH);
digitalWrite(green1, HIGH);
digitalWrite(red2, HIGH);
digitalWrite(red3,HIGH);
digitalWrite(red4, HIGH);
delay(1000);
digitalWrite(yellow1,LOW);
digitalWrite(yellow2, LOW);
digitalWrite(green1, LOW);
digitalWrite(red2, LOW);
```

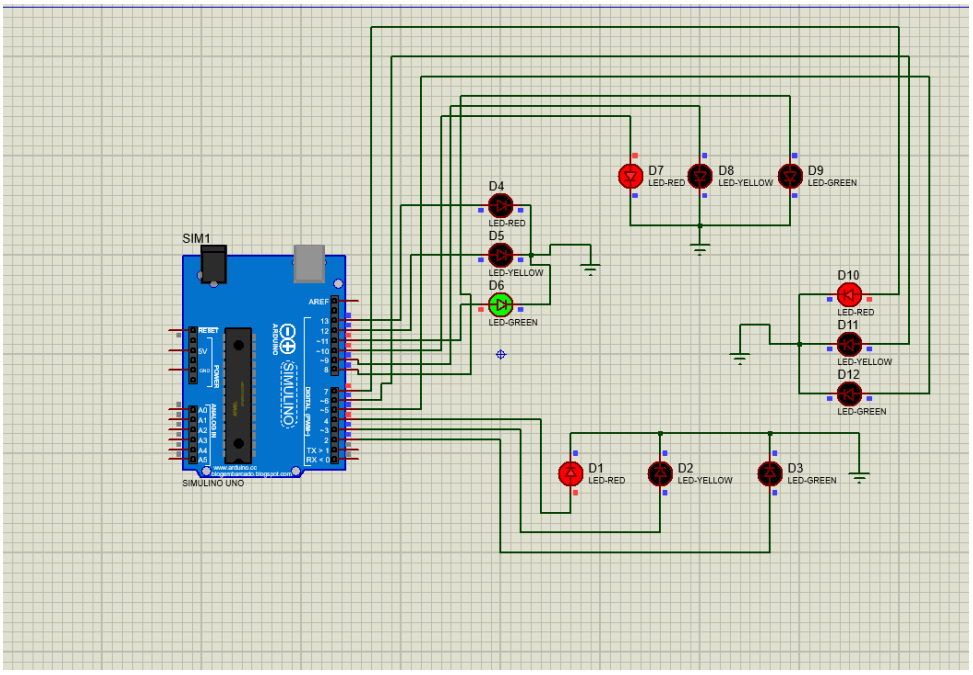
```
digitalWrite(red3,LOW);
digitalWrite(red4, LOW);

//SIGNAL2
digitalWrite(green2, HIGH);
digitalWrite(red1, HIGH);
digitalWrite(red3, HIGH);
digitalWrite(red4, HIGH);
delay(2000);
digitalWrite(yellow3,HIGH);
digitalWrite(yellow2, HIGH);
digitalWrite(green2, HIGH);
digitalWrite(red1, HIGH);
digitalWrite(red3, HIGH);
digitalWrite(red4, HIGH);
delay(1000);
digitalWrite(yellow2, LOW);
digitalWrite(yellow3,LOW);
digitalWrite(green2, LOW);
digitalWrite(red1, LOW);
digitalWrite(red3,LOW);
digitalWrite(red4, LOW);
delay(100);

//SIGNAL3
```

```
digitalWrite(green3, HIGH);
digitalWrite(red1, HIGH);
digitalWrite(red2, HIGH);
digitalWrite(red4,HIGH);
delay(2000);
digitalWrite(yellow3,HIGH);
digitalWrite(yellow4,HIGH);
digitalWrite(green3, HIGH);
digitalWrite(red1, HIGH);
digitalWrite(red3,HIGH);
digitalWrite(red4,HIGH);
delay(1000);
digitalWrite(yellow3,LOW);
digitalWrite(yellow4,LOW);
digitalWrite(green3, LOW);
digitalWrite(red1, LOW);
digitalWrite(red2, LOW);
digitalWrite(red4, LOW);
delay(100);
digitalWrite(green4, HIGH);
digitalWrite(red1,HIGH);
digitalWrite(red2, HIGH);
digitalWrite(red3, HIGH);
delay(2000);
digitalWrite(yellow4, HIGH);
```

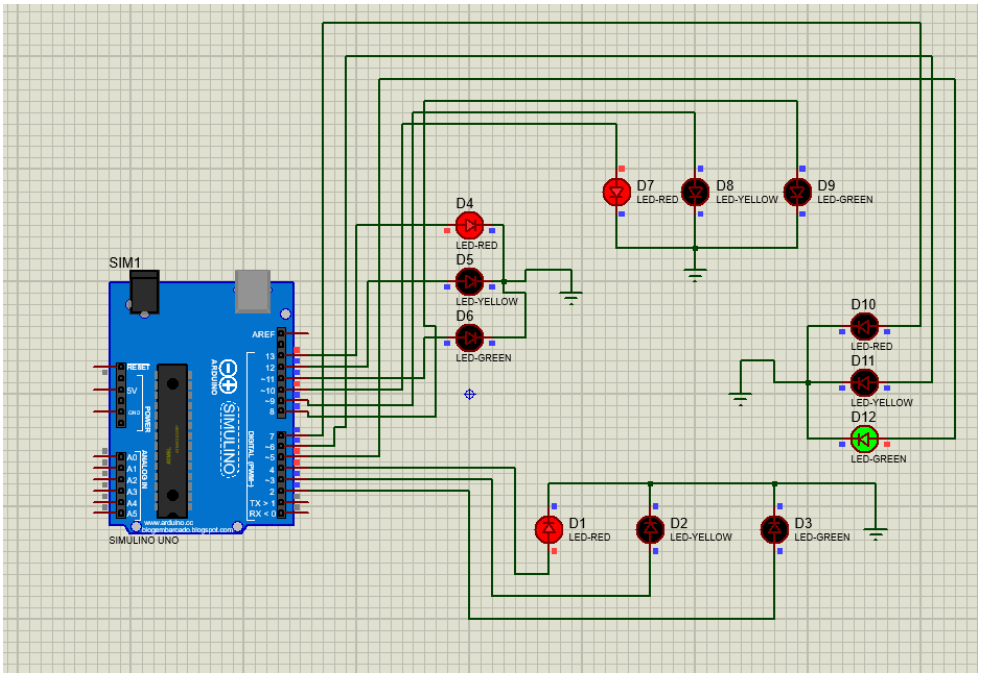
```
digitalWrite(yellow1,HIGH);
digitalWrite(green4, HIGH);
digitalWrite(red1,HIGH);
digitalWrite(red3, HIGH);
digitalWrite(red4,HIGH);
delay(1000);
digitalWrite(yellow4,LOW);
digitalWrite(yellow1,LOW);
digitalWrite(green4, LOW);
digitalWrite(red1,LOW);
digitalWrite(red2, LOW);
digitalWrite(red3, LOW);
delay(100);
}
```

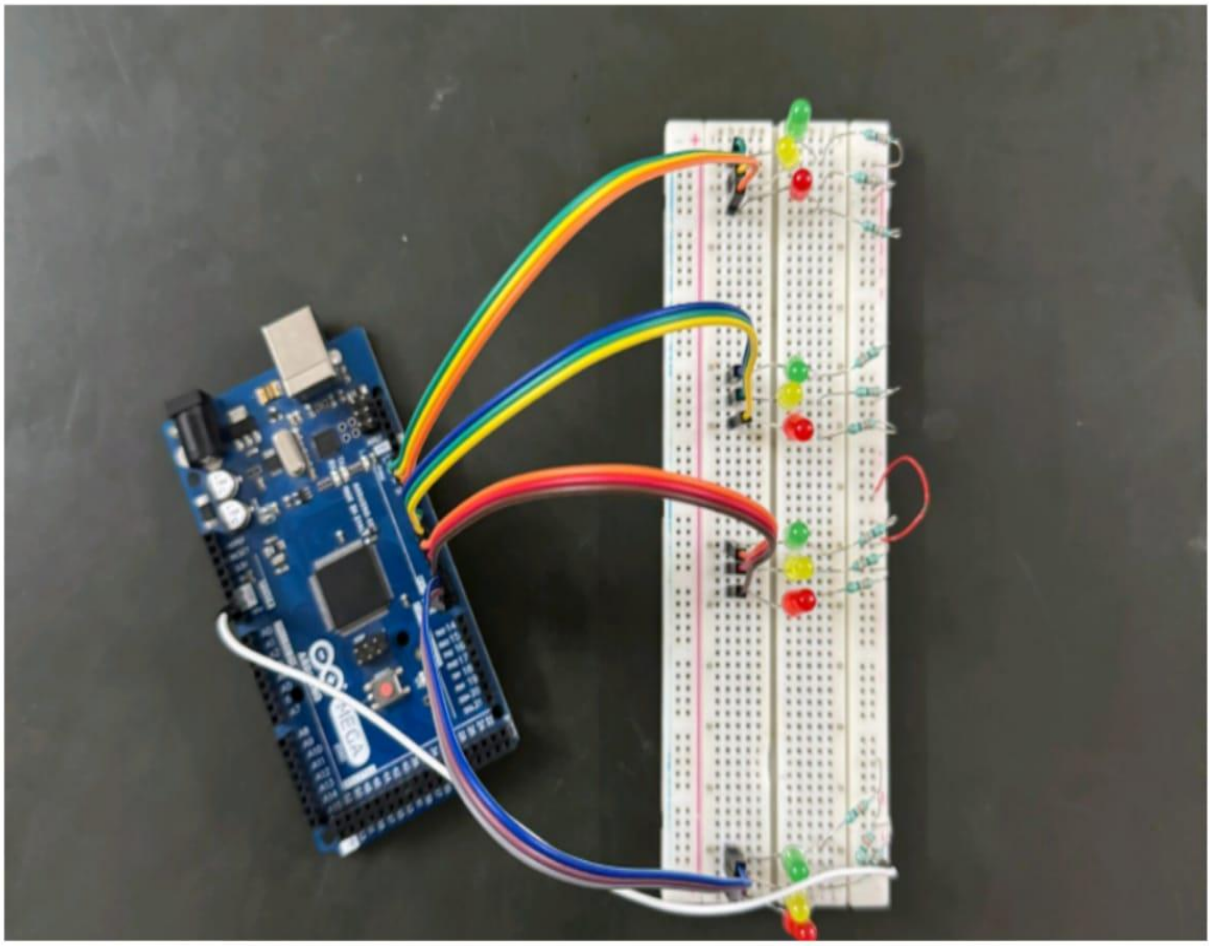
RESULTS

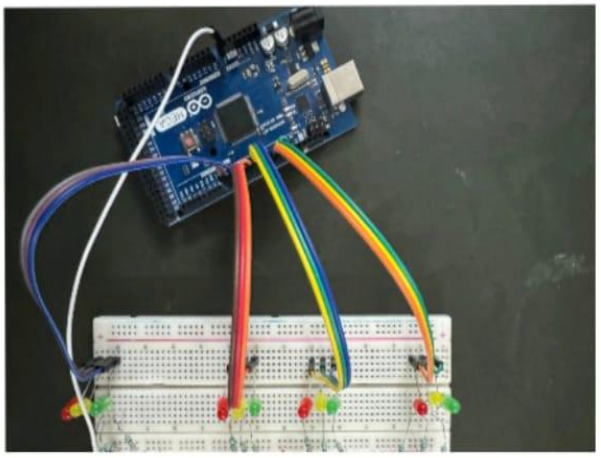
At first, traffic signal 1 is turned ON and the green light is displayed for 2000ms

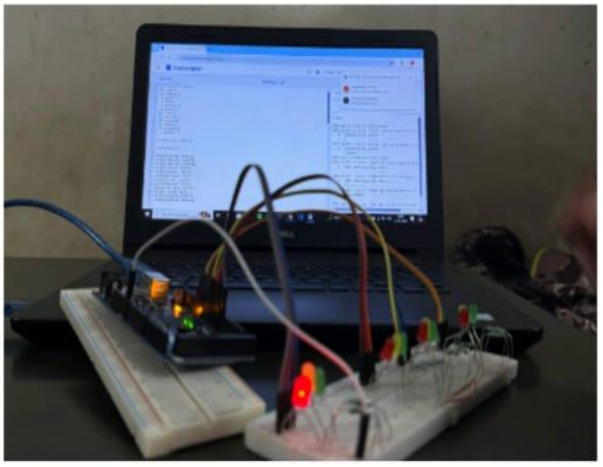


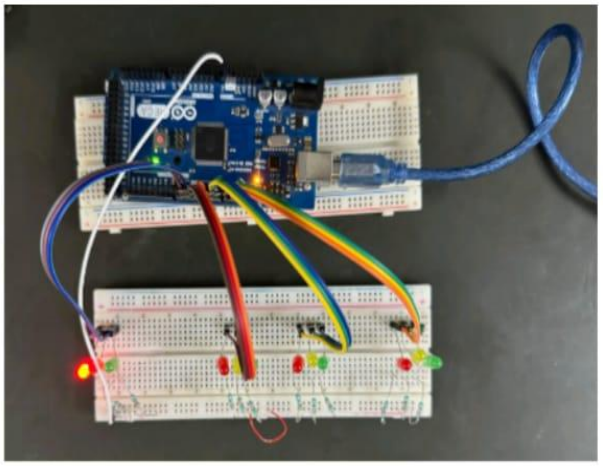
Then traffic signal 2 is turned ON and the green light is displayed for 2000ms.

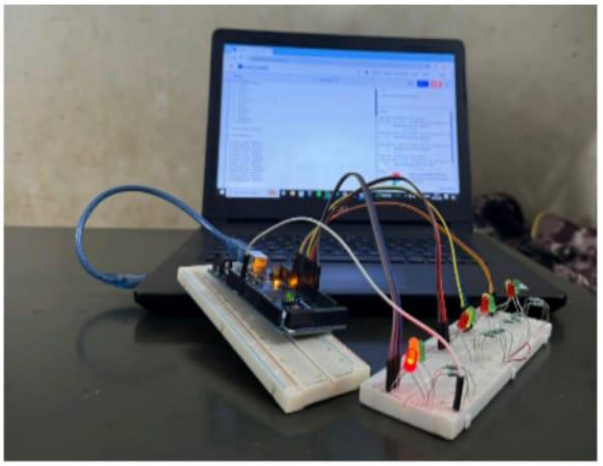


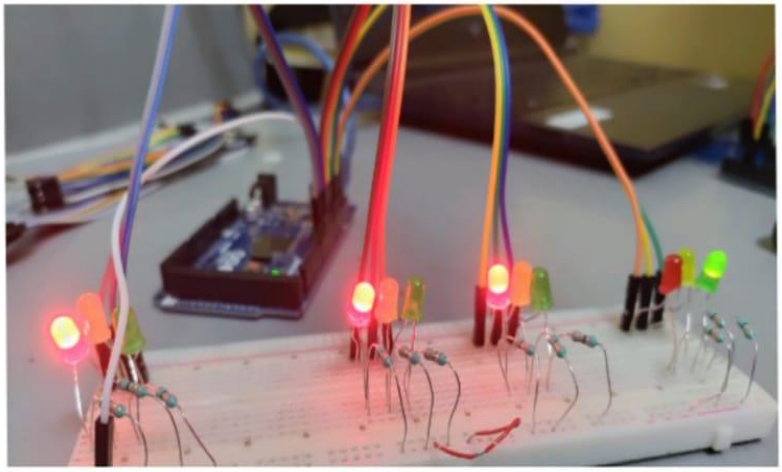


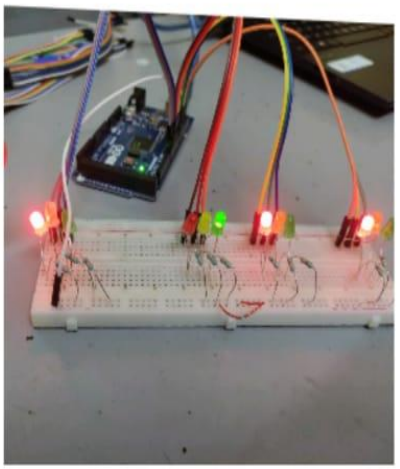


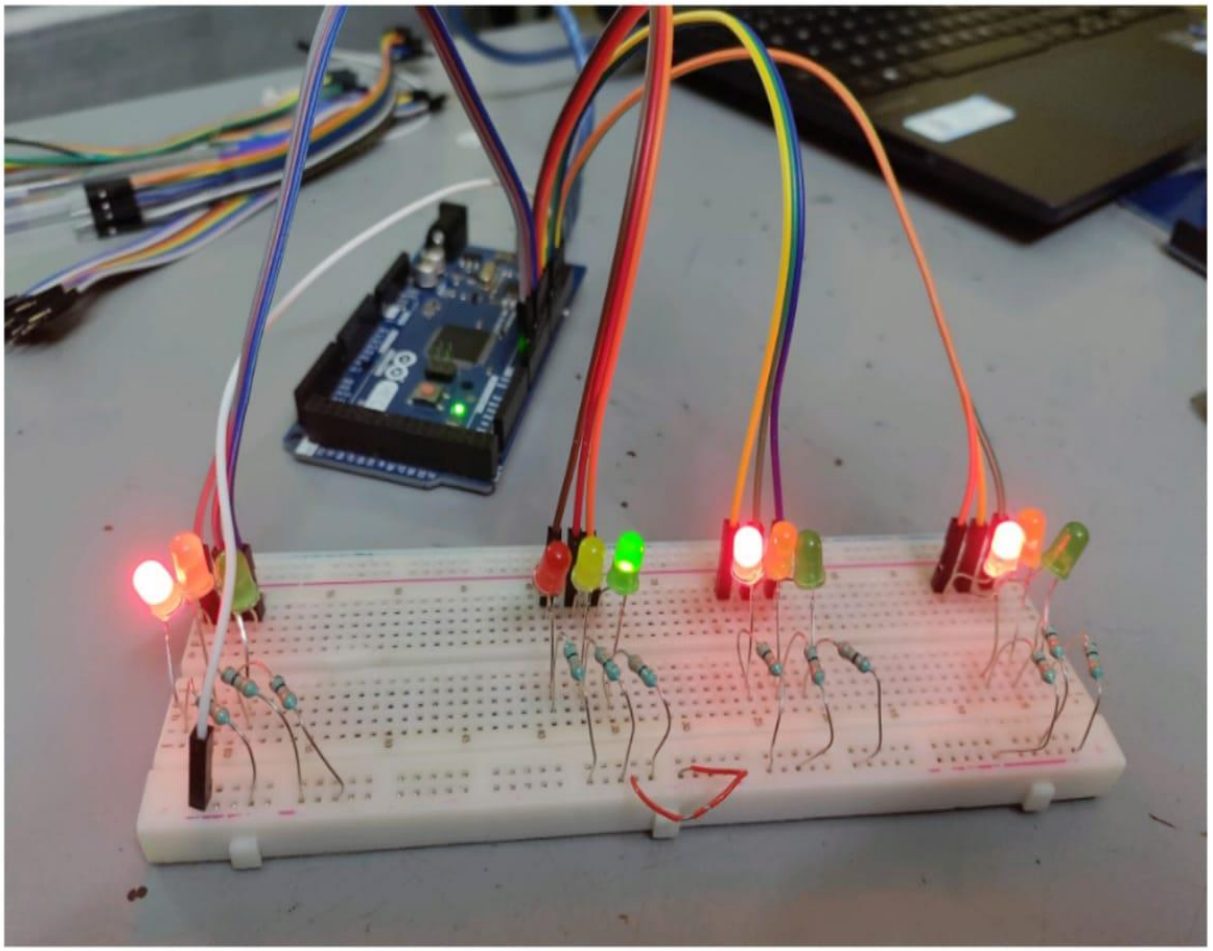


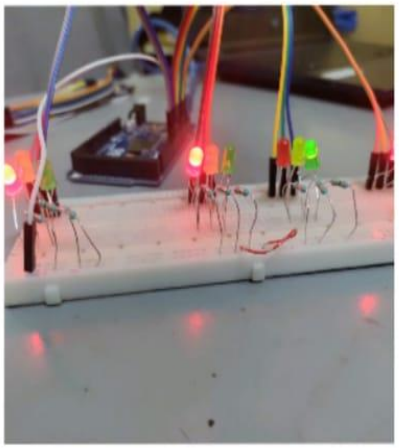


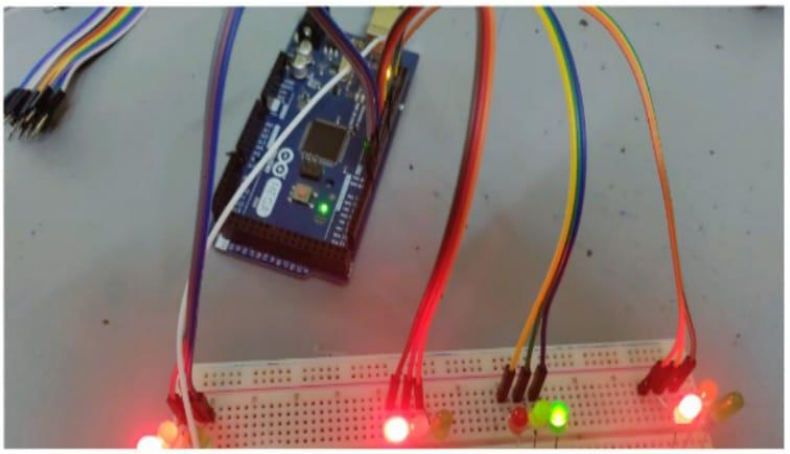












CONCLUSION

The 4-way traffic simulation utilizing the Arduino Mega represents a groundbreaking application of microcontroller technology in urban traffic management. Demonstrating the intricate interplay of safety protocols, efficient traffic flow, and advanced pedestrian management, this project underscores the Arduino Mega's versatility and adaptability. By emulating the complexities of real-world intersections, the simulation showcases how technology can mitigate challenges at critical traffic nodes. Furthermore, the successful implementation underscores the potential of Arduino platforms in shaping the future of smart cities, where connectivity, efficiency, and safety converge seamlessly. This initiative not only highlights the tangible benefits of integrating microcontroller solutions in urban planning but also sets a precedent for innovative approaches to addressing contemporary traffic management challenges. As urban landscapes evolve, such endeavors illuminate the path forward, championing technology's pivotal role in crafting sustainable, efficient, and safe urban environments.

REFERENCES

- https://www.theengineeringprojects.com/2021/12/4-way-traffic-light-control-using-arduino.html
- https://www.arduino.cc/reference/en/
- Exploring Basic IoT Applications in Traffic Management (2022)
- Assessment of Arduino-Based Traffic Signal Simulations (2023)