Department of CSE

Course name: Computer Architecture

Course Code: CSE360

Section: 01

Group: 07

Mini Project Report

Project Title: Simulate modern traffic control system

Submitted By:

Student's Name	Student's ID
Rahma Mahbub	2020-2-60-026
Mim Bin Hossain	2021-1-60-071
Md Rakibul Hasan	2021-1-50-006
Meherun Nessa Nowrin	2020-2-60-011

Submitted To:

Dr. Md. Nawab Yousuf Ali

Professor

Department of Computer Science and Engineering

Introduction:

Roadway network congestion is a common occurrence in contemporary metropolitan contexts due to the increasing use of engine cars and trip needs. In addition to harming economic output, this congestion aggravates environmental issues. It becomes clear that traffic control is an essential tool for controlling activity flow, reducing traffic, and minimizing its negative effects. The progress of traffic control systems has been fueled by the ongoing advancements in infrastructure science and data and computer technology. The most notable of these innovations is the self-adaptive control system, which can dynamically modify the timing parameters of the signals in real-time to match the main control goal-that is, to reduce junction delays and adapt to the entrance characteristics of traffic streams.Compared to traditional timing control and activated control methods, the self-adaptive control system demonstrates superior efficiency in maximizing overall traffic capacity and optimizing the performance of roadway networks. By adapting to changing traffic conditions on the fly, it represents a promising approach to address the challenges posed by urban congestion and enhance the effectiveness of traffic management strategies.

Objective:

In urban areas, traffic congestion arises when the volume of vehicles exceeds the road capacity, resulting in delays and frustration for commuters. This congestion can stem from various factors such as:-

- High volume of vehicles
- Accidents or incidents
- Improper routing
- VIP movements
- Illegal parking or deficiencies in road design
- Adherence to traffic regulations
- Special events or peak travel times

The consequences of traffic congestion are wide-ranging and can significantly impact individuals, communities, and economies. The consequences of traffic congestion are numerous and impactful:-

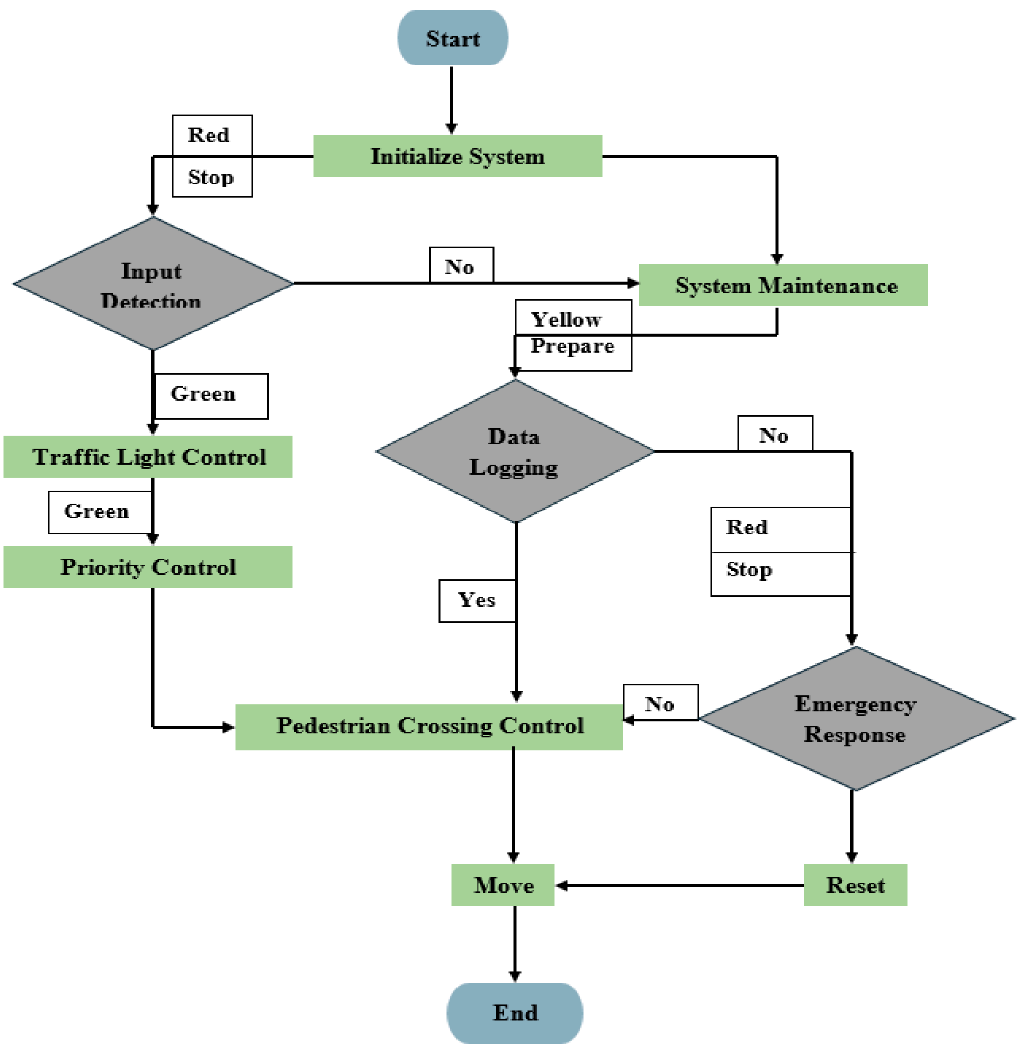
- It consumes significant amounts of time, impacting individuals' daily schedules
- Delays in reaching schools, workplaces, and other destinations become commonplace
- The heightened risk of accidents poses a threat to road safety

Recognizing this, the development of a modern traffic control system emerges as a critical solution for metropolitan areas. Such a system aims to minimize economic costs associated with congestion and optimize travel time for commuters, thereby enhancing overall efficiency and quality of life.

Theory:.

The theory section provides an overview of the system architecture, including input data sources, control units, and communication protocols. It also outlines the mathematical modeling of traffic flow dynamics and describes various control algorithms used in modern systems.

Flowchart:



The flowchart is used to simulate a modern traffic control system. Here's a breakdown of how it works:-

Initialize System: The system begins by initializing itself. This might involve loading up settings or calibrating sensors

Input Detection: The system then looks for input from traffic light sensors. These sensors could be inductive loop detectors embedded in the road that sense the presence of vehicles or cameras that can detect stopped vehicles

System Maintenance: If there is no input detected, the system might check to see if maintenance is required.

Yellow: If there is input detected, the light might turn yellow to warn drivers that the light is about to turn red

Red: The light then turns red, bringing traffic to a stop

Green: While the light is red, a timer counts down. Once the timer expires, the light may turn green

Data Logging: Data on traffic flow might be logged during this time

Priority Control: When the light is green, the system might check for any priority controls.

These controls could include:-

Pedestrian crossing control: If a pedestrian button is pressed, the light might give pedestrians the right of way

Emergency response: If an emergency vehicle is approaching with its lights and sirens on, the light might change to allow it to pass through the intersection

Move to End: If there are no priority controls, the system loops back to step 2 (Input Detection) to begin detecting traffic again.

Design:

We are used for two software:-

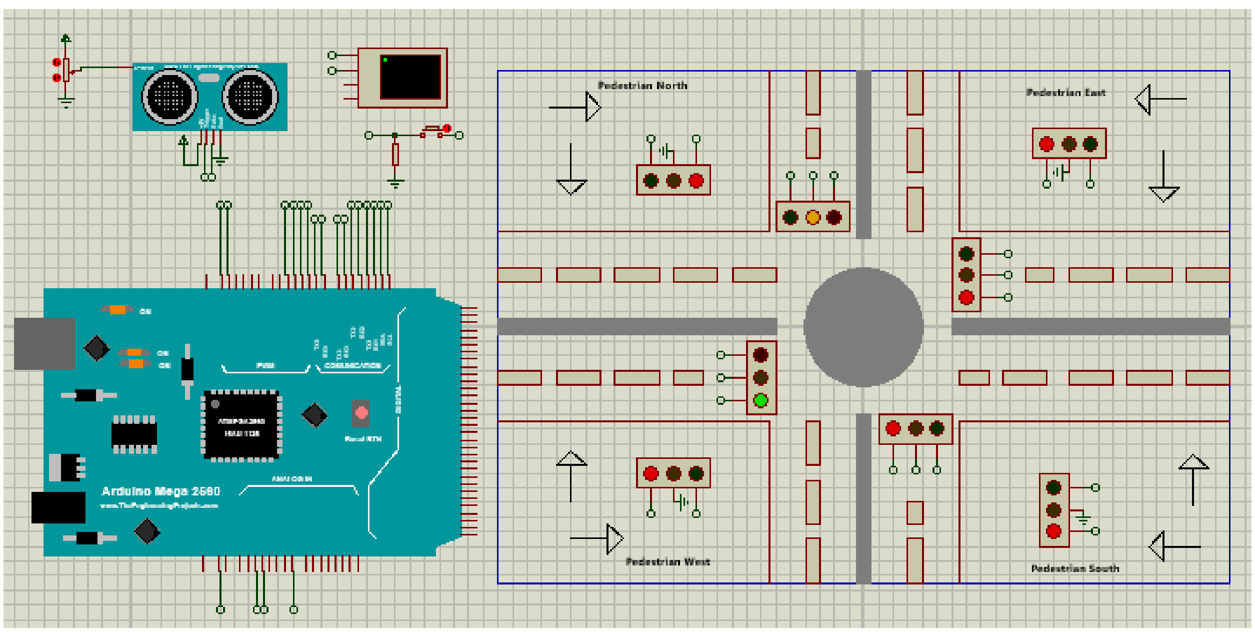
- Proteus
- Arduino IDE

Component:

We are used for component in this project:-

- Arduino Mega 2560
- Ultrasonic Sensor HC SR-04
- Traffic Light
- Virtual Thermal
- Wire
- Resistor
- POT -HG
- Button.

The components are shown in the following picture:-



Implementation (Code):

```
const int TrigPin = 14;
const int echoPin = 15;
const int g1 = 21;
const int y1 = 20;
const int r1 = 19;
const int g2 = 18;
const int y2 = 17;
const int r2 = 16;
const int g3 = A4;
const int y3 = 3;
const int r3 = 4;
const int g4 = 5;
const int y4 = 12;
const int r4 = 13;
const int pRed = A2;
const int pGreen = A3;
const int pIN = 2;

long microsecondToInches(long microsecond) {
  return microsecond / 74 / 2;
}

long microsecondToCentimeter(long microsecond) {
  return microsecond / 29 / 2;
}

void changeLights(int g, int y, int r, int nextGreenDelay) {
  digitalWrite(r, LOW);
  digitalWrite(g, HIGH);
```

```
  delay(nextGreenDelay);
  digitalWrite(g, LOW);
  digitalWrite(y, HIGH);
  delay(2000);
  digitalWrite(y, LOW);
  digitalWrite(r, HIGH);
}

void setup() {
  pinMode (r1, OUTPUT);
  pinMode (y1, OUTPUT);
  pinMode (g1, OUTPUT);
  pinMode (r2, OUTPUT);
  pinMode (y2, OUTPUT);
  pinMode (g2, OUTPUT);
  pinMode (r3, OUTPUT);
  pinMode (y3, OUTPUT);
  pinMode (g3, OUTPUT);
  pinMode (r4, OUTPUT);
  pinMode (y4, OUTPUT);
  pinMode (g4, OUTPUT);
  pinMode (pRed, OUTPUT);
  pinMode (pGreen, OUTPUT);
  pinMode (pIN, INPUT);
  digitalWrite (r1, HIGH);
  digitalWrite (r2, HIGH);
  digitalWrite (r3, HIGH);
  digitalWrite (r4, HIGH);
  digitalWrite (pRed, HIGH);
  Serial.begin (9600);
}
```

```
void loop() {
  long duration = 0;
  long inches = 0;
  long cm = 0;
  pinMode (TrigPin, OUTPUT);
  digitalWrite (TrigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(TrigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(TrigPin, LOW);
  pinMode(echoPin, INPUT);
  duration = pulseIn (echoPin, HIGH);
  inches = microsecondToInches (duration);
  cm = microsecondToCentimeter(duration);
  Serial.print ("Distance = ");
  Serial.print(inches);
  Serial.print("in, ");

  int crossIn = digitalRead(pIN);
  digitalWrite(g1, HIGH);
  digitalWrite(y1, LOW);
  digitalWrite(r1, LOW);
  digitalWrite(g2, LOW);
  digitalWrite(y2, LOW);
  digitalWrite(r2, HIGH);
  digitalWrite(g3, LOW);
  digitalWrite(y3, LOW);
  digitalWrite(r3, HIGH);
  digitalWrite(g4, LOW);
  digitalWrite(y4, LOW);
```

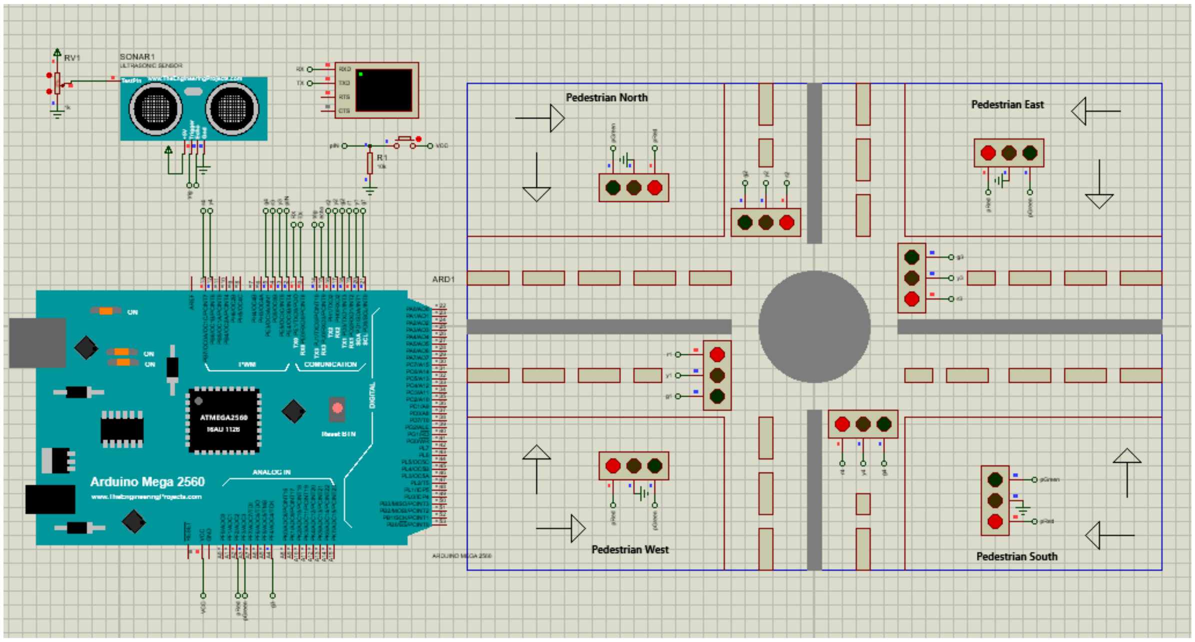
```
digitalWrite(r4, HIGH);
delay(1500);

digitalWrite(g1, HIGH);
digitalWrite(y1, LOW);
digitalWrite(r1, LOW);
digitalWrite(g2, LOW);
digitalWrite(y2, HIGH);
digitalWrite(r2, LOW);
digitalWrite(g3, LOW);
digitalWrite(y3, LOW);
digitalWrite(r3, HIGH);
digitalWrite(g4, LOW);
digitalWrite(y4, LOW);
digitalWrite(r4, HIGH);
delay(1500);

digitalWrite(g1, LOW);
digitalWrite(y1, LOW);
digitalWrite(r1, HIGH);
digitalWrite(g2, HIGH);
digitalWrite(y2, LOW);
digitalWrite(r2, LOW);
digitalWrite(g3, LOW);
digitalWrite(y3, LOW);
digitalWrite(r3, HIGH);
digitalWrite(g4, LOW);
digitalWrite(y4, LOW);
digitalWrite(r4, HIGH);
delay(1500);

digitalWrite(g1, LOW);
```

```
digitalWrite(y1, LOW);
digitalWrite(r1, HIGH);
digitalWrite(g2, HIGH);
digitalWrite(y2, LOW);
digitalWrite(r2, LOW);
digitalWrite(g3, LOW);
digitalWrite(y3, HIGH);
digitalWrite(r3, LOW);
digitalWrite(g4, LOW);
digitalWrite(y4, LOW);
digitalWrite(r4, HIGH);
delay(1500);

digitalWrite(g1, LOW);
digitalWrite(y1, LOW);
digitalWrite(r1, HIGH);
digitalWrite(g2, LOW);
digitalWrite(y2, LOW);
digitalWrite(r2, HIGH);
digitalWrite(g3, HIGH);
digitalWrite(y3, LOW);
digitalWrite(r3, LOW);
digitalWrite(g4, LOW);
digitalWrite(y4, LOW);
digitalWrite(r4, HIGH);
delay(1500);

digitalWrite(g1, LOW);
digitalWrite(y1, LOW);
digitalWrite(r1, HIGH);
digitalWrite(g2, LOW);
digitalWrite(y2, LOW);
```

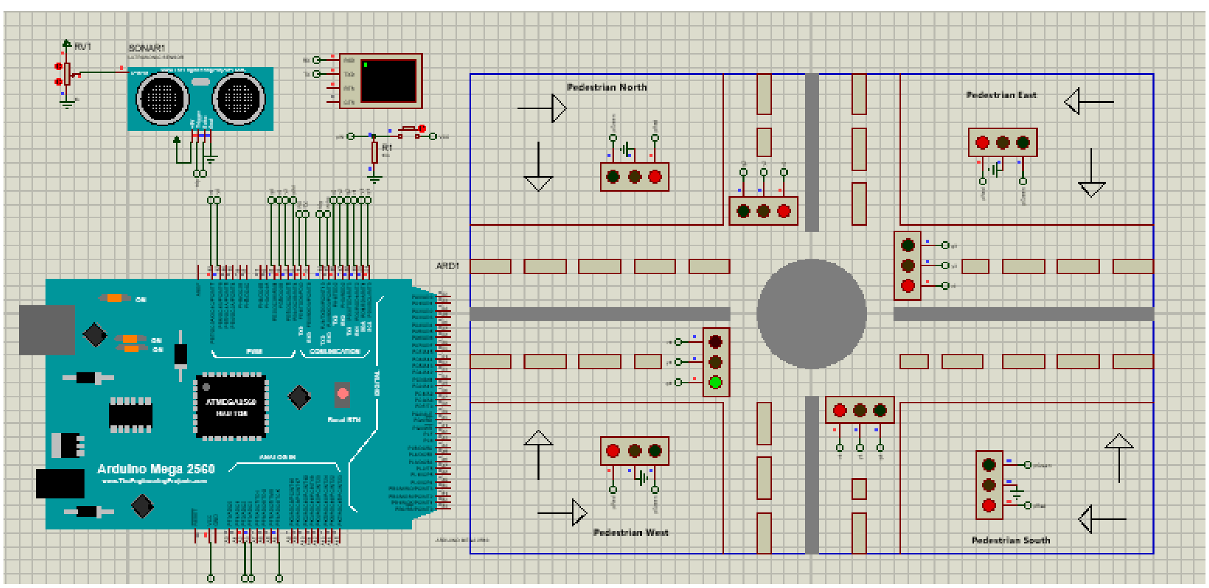
```
digitalWrite(r2, HIGH);
digitalWrite(g3, HIGH);
digitalWrite(y3, LOW);
digitalWrite(r3, LOW);
digitalWrite(g4, LOW);
digitalWrite(y4, HIGH);
digitalWrite(r4, LOW);
delay(1500);

digitalWrite(g1, LOW);
digitalWrite(y1, LOW);
digitalWrite(r1, HIGH);
digitalWrite(g2, LOW);
digitalWrite(y2, LOW);
digitalWrite(r2, HIGH);
digitalWrite(g3, LOW);
digitalWrite(y3, LOW);
digitalWrite(r3, HIGH);
digitalWrite(g4, HIGH);
digitalWrite(y4, LOW);
digitalWrite(r4, LOW);
delay(1500);

digitalWrite (g1, LOW);
digitalWrite (g2, LOW);
digitalWrite (g3, LOW);
digitalWrite (g4, LOW);
digitalWrite (y1, LOW);
digitalWrite (y2, LOW);
digitalWrite (y3, LOW);
digitalWrite (y4, LOW);
digitalWrite (r1, HIGH);
```

```
  digitalWrite (r2, HIGH);
  digitalWrite (r3, HIGH);
  digitalWrite (r4, HIGH);

  digitalWrite (pGreen, HIGH);
  digitalWrite (pRed, LOW);
  delay (800);
  digitalWrite (pGreen, LOW);
  digitalWrite(pRed, HIGH);

  digitalWrite(g1, LOW);
  digitalWrite(y1, HIGH);
  digitalWrite(r1, LOW);
  digitalWrite(g2, LOW);
  digitalWrite(y2, LOW);
  digitalWrite(r2, HIGH);
  digitalWrite(g3, LOW);
  digitalWrite(y3, LOW);
  digitalWrite(r3, HIGH);
  digitalWrite(g4, HIGH);
  digitalWrite(y4, LOW);
  digitalWrite(r4, LOW);
  delay(1500);

}
```

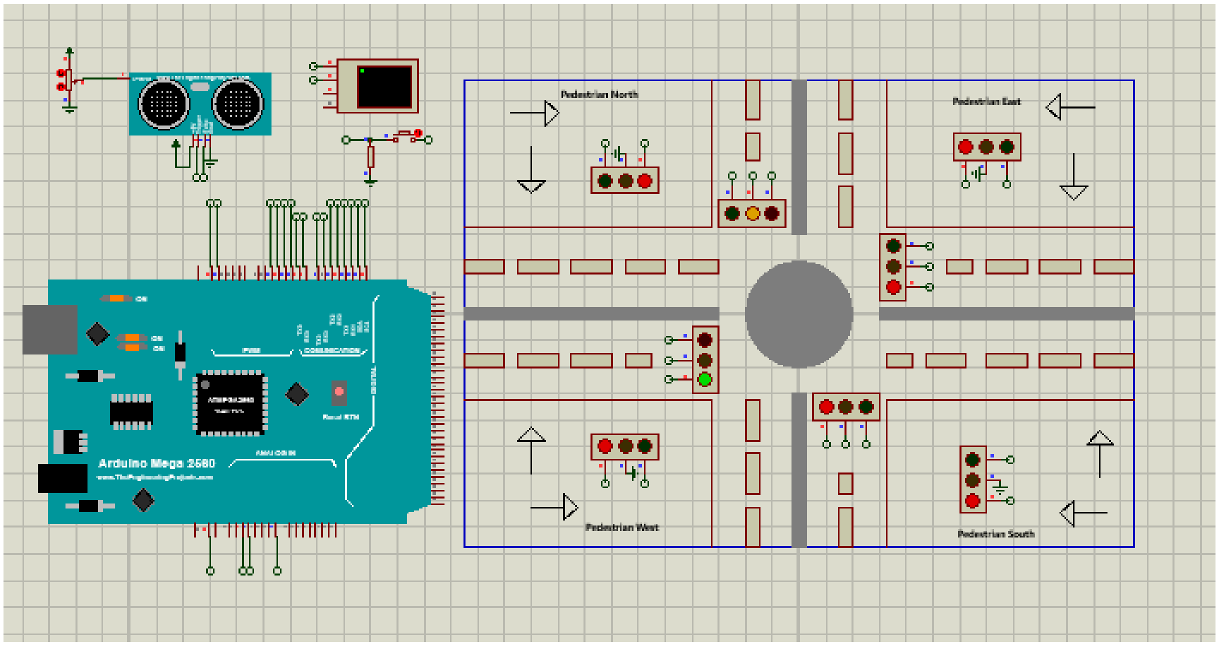
Debugging-Test-run:



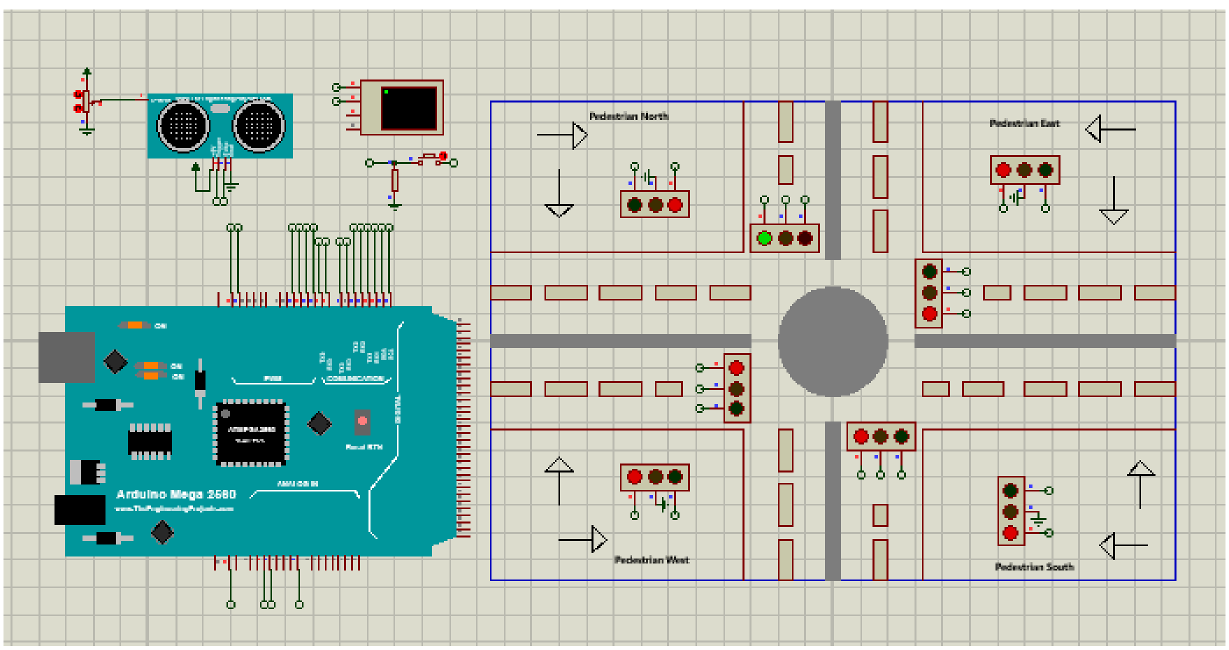
Simulation starts: when g1 is high, all other lights are turned off. Wait for 1.5 seconds.



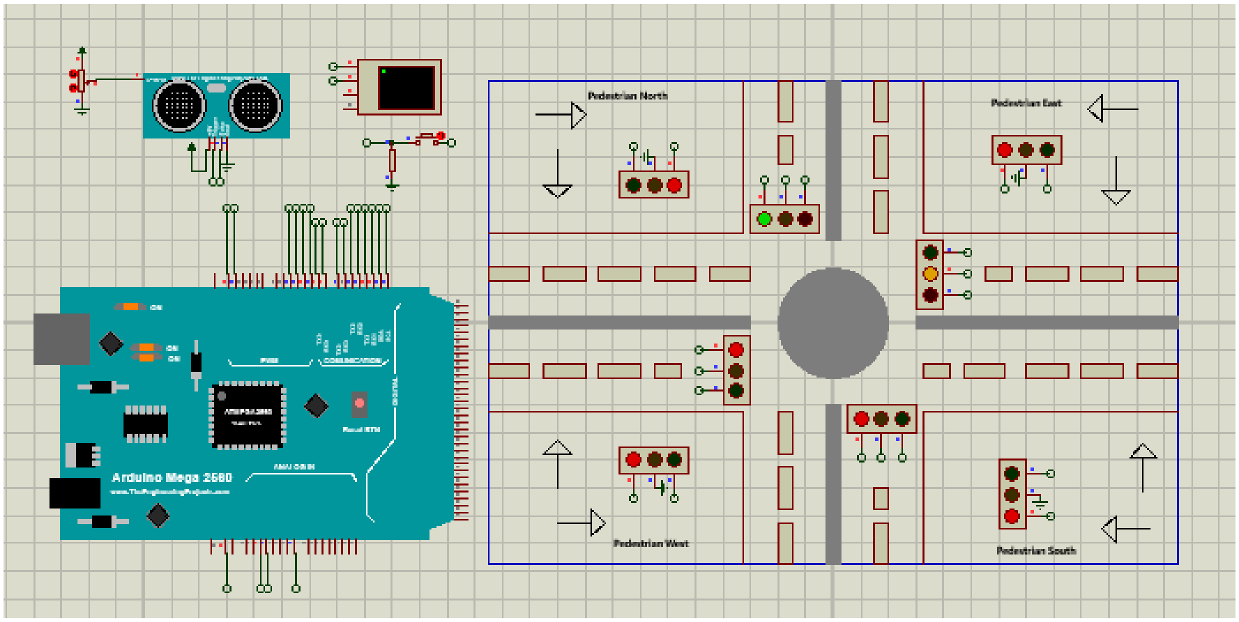
When g1 is high, y2 is turned on, because the vehicle of lane 2 is ready to go. Wait for 1.5 seconds.



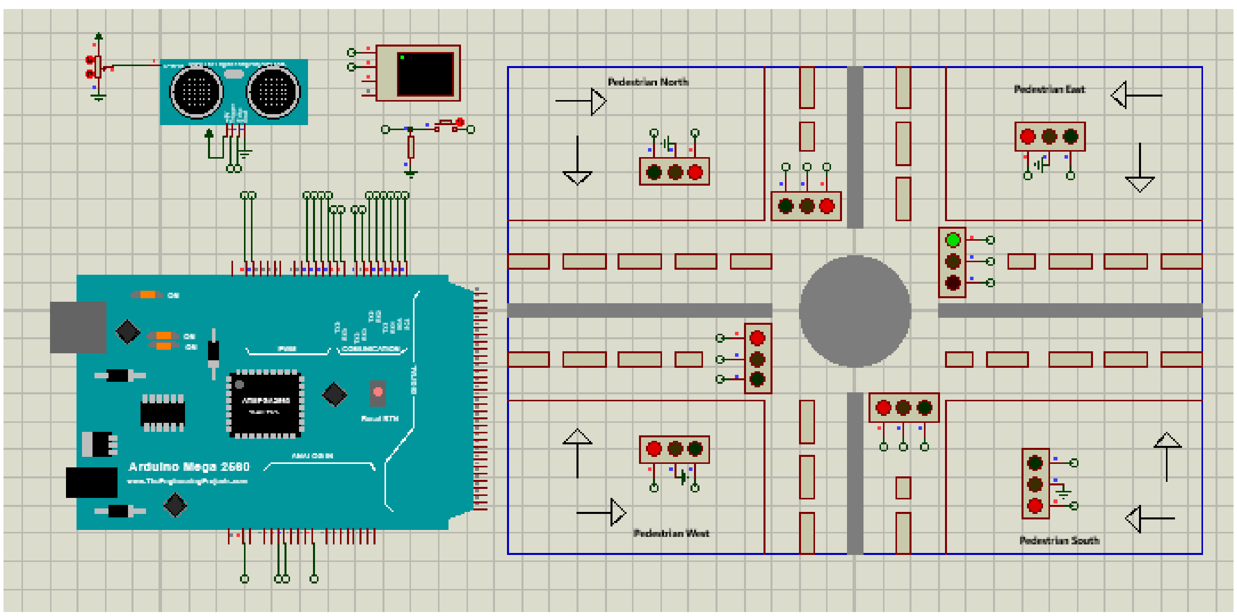
When g2 is high, all other lights are turned off. Wait for 1.5 seconds.



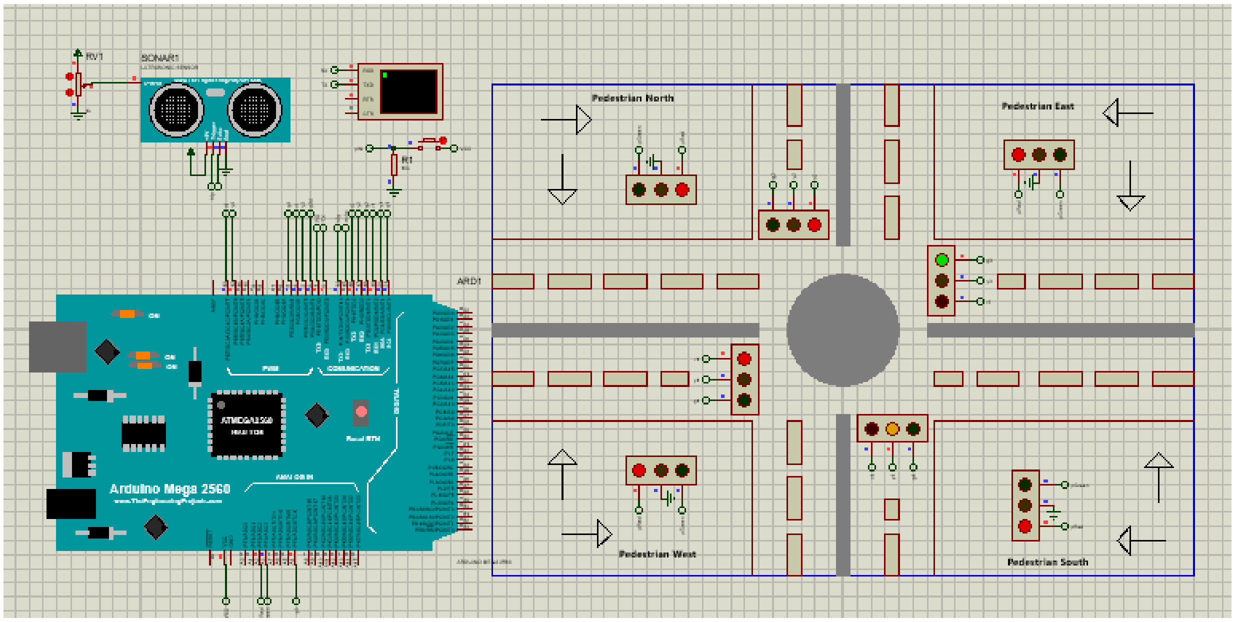
When g2 is high, y3 is high because the vehicles of lane 3 are ready to go. Wait for 1.5 seconds.



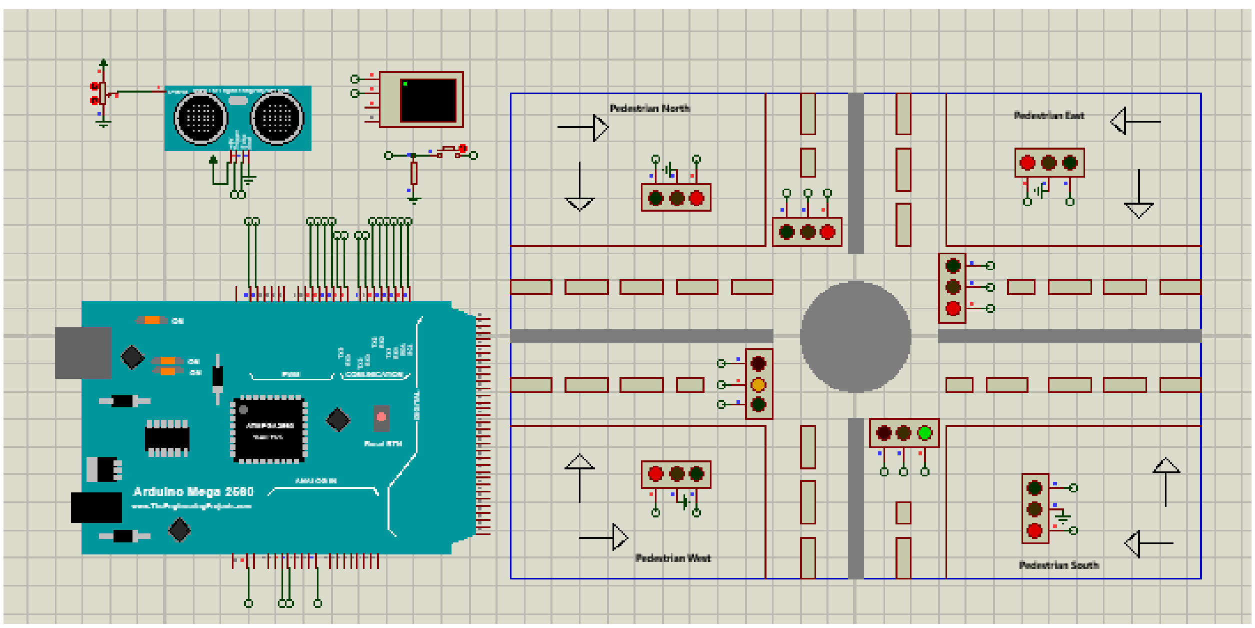
When g3 is high, all other lights are turned off and wait for 1.5 seconds.



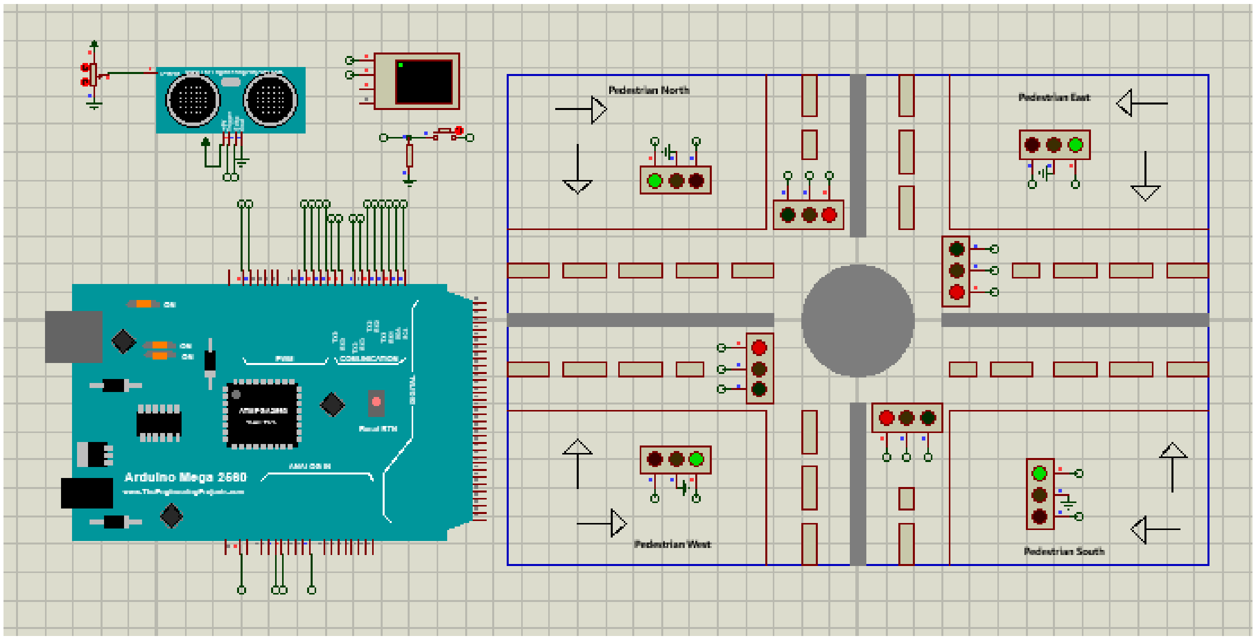
When g3 is high, y4 is high, because the vehicles of lane 4 are ready to go. Wait for 1.5 seconds.



When g4 is high, y1 is high, because the vehicles of lane 1 are ready to go. Wait for 1.5 seconds.



Now all of the pedestrian lights are turned on and other red lights are turned on. Wait for 0.8 second.



So, our code is fool-proof and fully debugged.

Limitations:

- Could not simulate a density based traffic control system.
- There is no Emergency vehicle detection
- No Hardware Simulation

Conclusion and Future Improvement:

We have controlled all the traffic lights following a sequence and there is no deadlock. This paper proposes the concept of modern traffic control management without involving many changes in the existing design. Since time is very precious, this system will save you a lot of unnecessary wasted time due to improper operation of traffic lights. This will not only save time but fuel is also burned unnecessarily and thus saves the environment and an individual's money. This will make transportation by road more convenient and easier. The modern traffic management system helps to detect congested areas and, accordingly, reduce congestion.

Bibliography :

- https://www.researchgate.net/publication/341871721_Traffic_Control_System_and_Technologies_A_Survey
- https://www.sciencedirect.com/topics/computer-science/traffic-control-system
- https://www.cdc.gov/injury/features/global-road-safety/index.html
- https://iopscience.iop.org/article/10.1088/1757-899X/377/1/012201/pdf

THE END...THANK YOU FOR READING