Density Based Traffic Control System

Requirements:

Hardware:

1. Arduino Uno Board x 1
2. LEDs=12(Redx3+Yellowx3+Greenx3)
3. Jumper wires (as much as possible)
4. Breadboardx1
5. IR Sensors(optional)

Software:

Arduino IDE

Description:

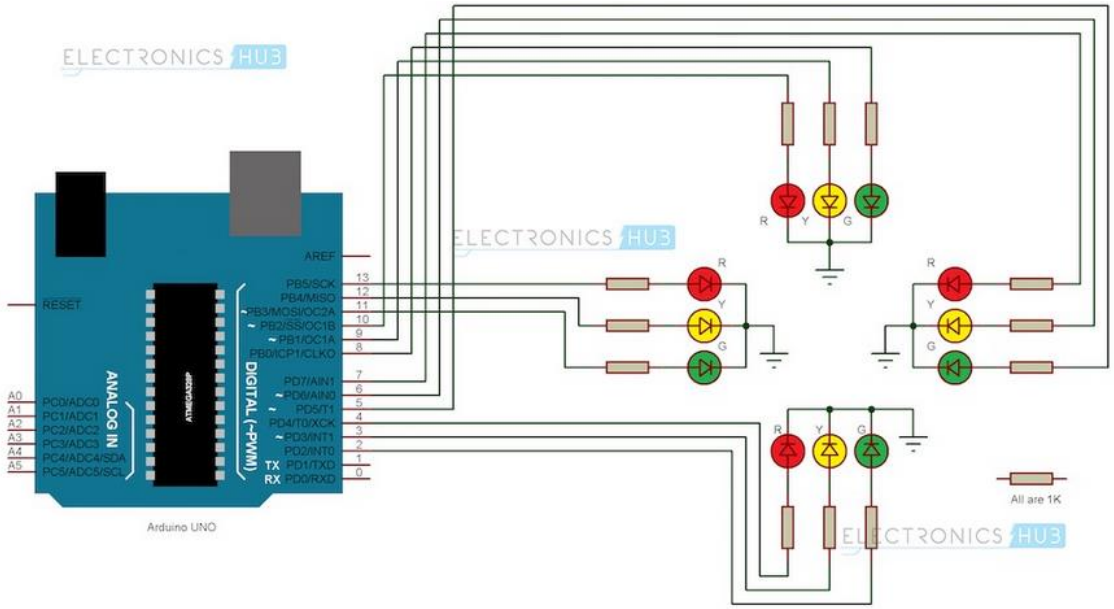
This project is just a mini-world implementation of a 4 lane automated traffic control system.

In this project, every lane should have only 1 light on at one time while others should be off. Let the green light of any one lane gets green then it is necessary for the red lights on the other 3 lanes to turn on. After a delay of certain time say 5 seconds, the green light of any one of the adjacent lane will be turned on and the red lights of the other lanes should be on and other lights off. As a warning indicator, the yellow light of that adjacent lane should be on giving a green signal afterwards. Similar happens for all the lanes and then it comes to the first lane again and hence the loop continues.

For the purpose of traffic control via vehicle density calculation, we can make use of IR sensors to detect the presence of vehicles and accordingly coding the requirement.

Another alternative is by using Google API.

Circuit:

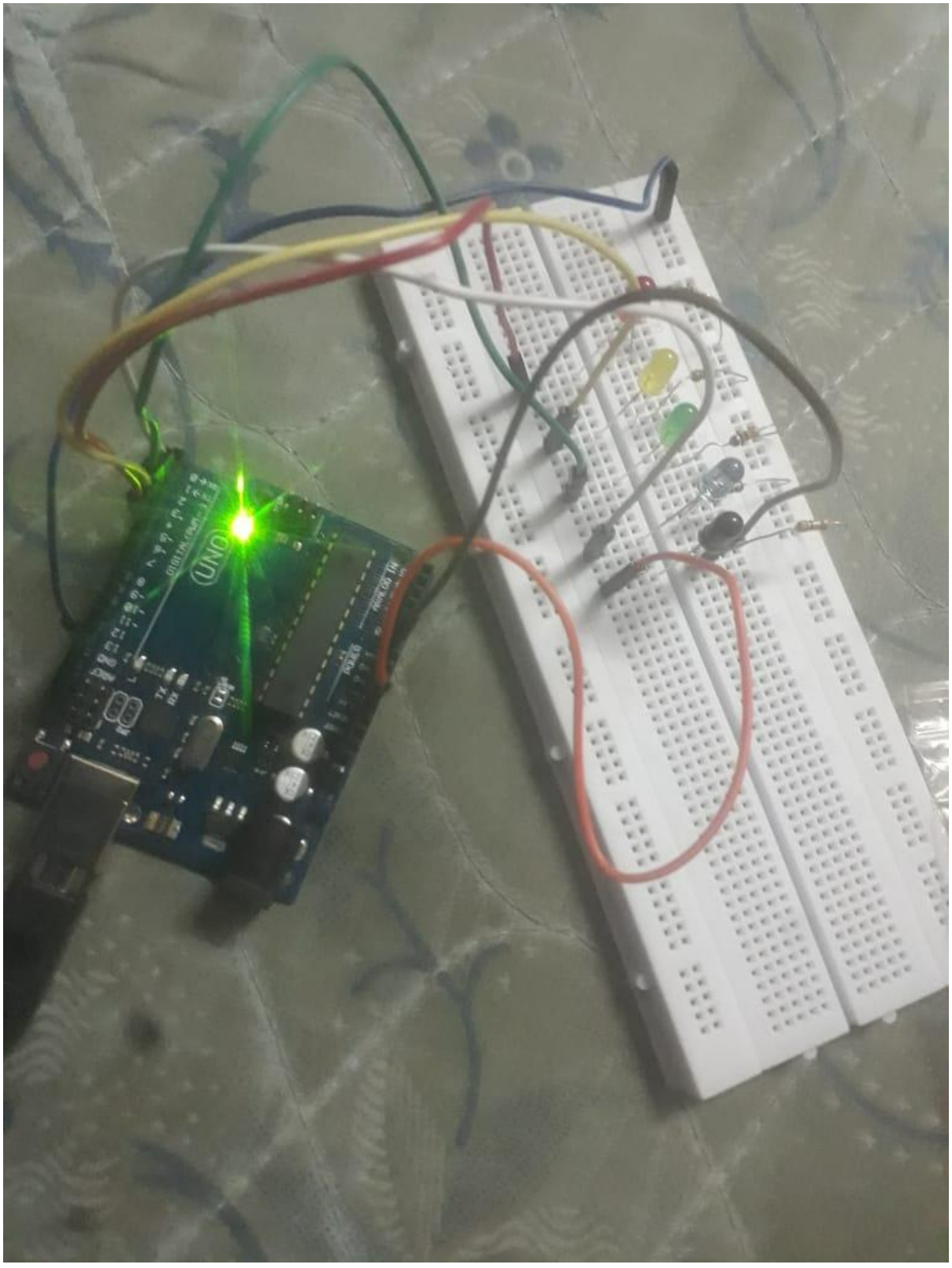


Code:

```
int Lane1[] = {13,12,11}; // Lane 1 Red, Yellow and Green
int Lane2[] = {10,9,8};// Lane 2 Red, Yellow and Green
int Lane3[] = {7,6,5};// Lane 3 Red, Yellow and Green
int Lane4[] = {4,3,2};// Lane 4 Red, Yellow and Green

void setup()
{
for (int i = 0; i < 3; i++)
{
```

```
pinMode(Lane1[i], OUTPUT);
pinMode(Lane2[i], OUTPUT);
pinMode(Lane3[i], OUTPUT);
pinMode(Lane4[i], OUTPUT);
}
for (int i = 0; i < 3; i++)
{
digitalWrite(Lane1[i], LOW);
digitalWrite(Lane2[i], LOW);
digitalWrite(Lane3[i], LOW);
digitalWrite(Lane4[i], LOW);
}

}

void loop()
{
digitalWrite(Lane1[2], HIGH);
digitalWrite(Lane3[0], HIGH);
digitalWrite(Lane4[0], HIGH);
digitalWrite(Lane2[0], HIGH);
delay(7000);
digitalWrite(Lane1[2], LOW);
digitalWrite(Lane3[0], LOW);
digitalWrite(Lane1[1], HIGH);
digitalWrite(Lane3[1], HIGH);
delay(3000);
digitalWrite(Lane1[1], LOW);
digitalWrite(Lane3[1], LOW);
digitalWrite(Lane1[0], HIGH);
digitalWrite(Lane3[2], HIGH);
delay(7000);
digitalWrite(Lane3[2], LOW);
digitalWrite(Lane4[0], LOW);
```

```
digitalWrite(Lane3[1], HIGH);
digitalWrite(Lane4[1], HIGH);
delay(3000);
digitalWrite(Lane3[1], LOW);
digitalWrite(Lane4[1], LOW);
digitalWrite(Lane3[0], HIGH);
digitalWrite(Lane4[2], HIGH);
delay(7000);
digitalWrite(Lane4[2], LOW);
digitalWrite(Lane2[0], LOW);
digitalWrite(Lane4[1], HIGH);
digitalWrite(Lane2[1], HIGH);
delay(3000);
digitalWrite(Lane4[1], LOW);
digitalWrite(Lane2[1], LOW);
digitalWrite(Lane4[0], HIGH);
digitalWrite(Lane2[2], HIGH);
delay(7000);
digitalWrite(Lane1[0], LOW);
digitalWrite(Lane2[2], LOW);
digitalWrite(Lane1[1], HIGH);
digitalWrite(Lane2[1], HIGH);
delay(3000);
digitalWrite(Lane2[1], LOW);
digitalWrite(Lane1[1], LOW);
}
```



Code for the above setup:

```
int green=3;

int yellow=4;
```

```
int red=5;
int val=0;
const int IR=A0;
const int delay_g=5000;
const int delay_y=2000;
const int delay_r=7500;
void setup() {

  Serial.begin(9600);
  pinMode(green,OUTPUT);
  pinMode(yellow,OUTPUT);
  pinMode(red,OUTPUT);

}

void loop() {
  greenlight();
  delay(delay_g);
  yellowlight();
  delay(delay_y);
  redlight();
  delay(delay_r);
  val=analogRead(IR);
  Serial.print("Sensor Value = ");
```

```
  Serial.print(val);
  if(val<100)
  {
    redlight();
    delay(delay_r);
    Serial.print("\nCar approaching");
  }
  else
  {
    yellowlight();
    delay(delay_y);
    greenlight();
    delay(delay_g);
  }

}
void greenlight()
{
  digitalWrite(green,HIGH);
  digitalWrite(yellow,LOW);
  digitalWrite(red,LOW);
}
void yellowlight()
{
```

```
  digitalWrite(yellow,HIGH);
  digitalWrite(red,LOW);
  digitalWrite(green,LOW);
}
void redlight()
{
  digitalWrite(red,HIGH);
  digitalWrite(yellow,LOW);
  digitalWrite(green,LOW);
}
```