Density Based Traffic Control System

# Problem statement

Majority of the traffic signals deployed on Pakistan’s roads are controlled through set periods of time. This can cause a jam on roads where the amount of traffic is heavy. The solution to this problem is to move towards traffic signals being governed by a constraint other than time. One of the constraints can be “Traffic Density” on a particular side of the road. The signal would turn green for the side where the traffic density is comparatively higher.

# Design requirements

Basic materials required for the project.

● Arduino Uno

● 2 X Ultrasonic Sensors

● 2 X Red LEDs

● 2 X Green LEDs

● 2 X Yellow LEDs

● 6 X 220-ohm resistors

● Jumper cables

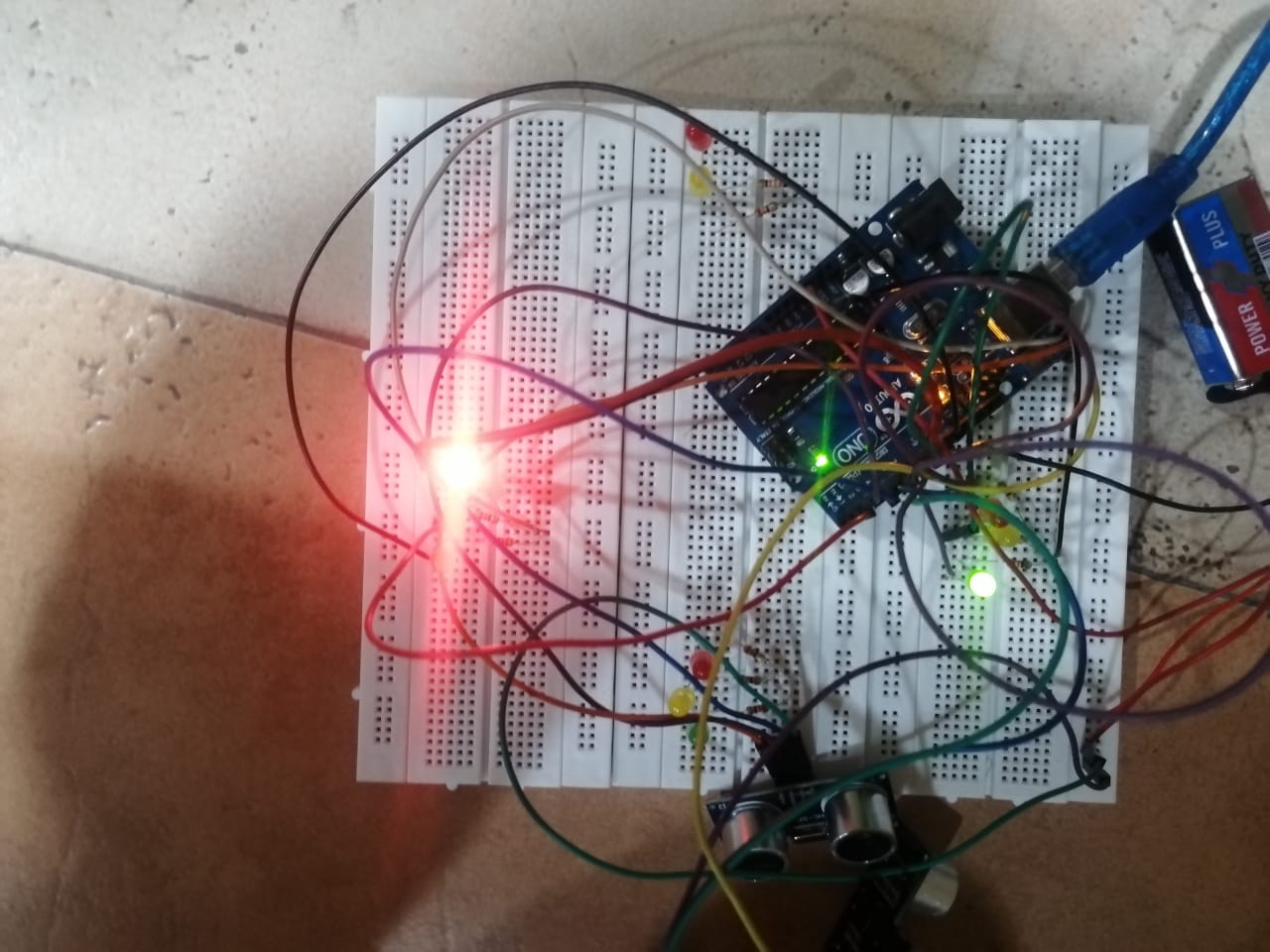
● Breadboards

# Circuit Diagram

# How to Build an Arduino Traffic Light Controller | 4-Way - Arduino Project Hub

# Real World Implementation

A circuit board with wires

Description automatically generated with low confidence

# Working and Flow Diagram

The working of the project is divided into three steps

* Normal Working of a traffic signal if there are cars present on all 4 signals or 2 in our case.
* If there is only traffic at one signal and no traffic at the others, this loop skips all the non-traffic regions and turns green on for the traffic one leading to better road navigation.
* If there is no traffic at any signal, the signals stop at their concurrent position until further presence.

Diagram

Description automatically generated

# Design

We were tasked with a traffic control system which required 4 ultrasonic sensors adding to the density-based charm/novelty of our project.

Firstly, we used the TimerOne.h library which is used to repetitively measure time in microseconds and repeatedly iterate over normal signal functioning until or unless interrupt is called.

The main purpose of this project is, if there is no traffic at a signal, one should skip over that signal and move on to the next one. This is where the Interrupt signal comes into play. It’s called after every 1/10th of a second and is constantly detecting “cars” / “traffic” at the signals.

# Limitations and Improvements

1. We were monetarily limited to an Arduino UNO, so we sorted to a highway (2-way) traffic system. This can be rendered into a 4-way traffic system as earlier intended using an Arduino MEGA.
2. Instead of detecting presence and absence of a car, this can detect the number of cars at a station. This can be improved by adding more sensors to the circuit.
3. Use better ultrasonic sensors or use Infrared technology i.e., Infrared sensors for better detection and analysis.
4. Program can be further coded to prevent the signals to stop if there is no traffic at all.
5. LEDS can be programmed to work repeatedly on loop (normally) without subdue traffic.

# Programs Used

* Arduino IDE

# Code

#include<TimerOne.h>

int signal1[] = {23, 25, 27};

int signal2[] = {46, 48, 50};

int signal3[] = {13, 12, 11};

int signal4[] = {10, 9, 8};

int redDelay = 5000;

int yellowDelay = 2000;

volatile int triggerpin1 = 31;

volatile int echopin1 = 29;

volatile int triggerpin2 = 44;

volatile int echopin2 = 42;

volatile int triggerpin3 = 7;

volatile int echopin3 = 6;

volatile int triggerpin4 = 5;

volatile int echopin4 = 4;

volatile long time; // Variable for storing the time traveled

volatile int S1, S2, S3, S4; // Variables for storing the distance covered

int t = 5; // distance under which it will look for vehicles.

void setup(){

Serial.begin(115200);

Timer1.initialize(100000); //Begin using the timer. This function must be called first. "microseconds" is the period of time the timer takes.

Timer1.attachInterrupt(softInterr); //Run a function each time the timer period finishes.

// Declaring LED pins as output

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

pinMode(signal1[i], OUTPUT);

pinMode(signal2[i], OUTPUT);

pinMode(signal3[i], OUTPUT);

pinMode(signal4[i], OUTPUT);

}

// Declaring ultrasonic sensor pins as output

pinMode(triggerpin1, OUTPUT);

pinMode(echopin1, INPUT);

pinMode(triggerpin2, OUTPUT);

pinMode(echopin2, INPUT);

pinMode(triggerpin3, OUTPUT);

pinMode(echopin3, INPUT);

pinMode(triggerpin4, OUTPUT);

pinMode(echopin4, INPUT);

}

void loop()

{

// If there are vehicles at signal 1

if(S1<t)

{

signal1Function();

}

// If there are vehicles at signal 2

if(S2<t)

{

signal2Function();

}

// If there are vehicles at signal 3

if(S3<t)

{

signal3Function();

}

// If there are vehicles at signal 4

if(S4<t)

{

signal4Function();

}

}

// This is interrupt function and it will run each time the timer period finishes. The timer period is set at 100 milli seconds.

void softInterr()

{

// Reading from first ultrasonic sensor

digitalWrite(triggerpin1, LOW);

delayMicroseconds(2);

digitalWrite(triggerpin1, HIGH);

delayMicroseconds(10);

digitalWrite(triggerpin1, LOW);

time = pulseIn(echopin1, HIGH);

S1= time\*0.034/2;

// Reading from second ultrasonic sensor

digitalWrite(triggerpin2, LOW);

delayMicroseconds(2);

digitalWrite(triggerpin2, HIGH);

delayMicroseconds(10);

digitalWrite(triggerpin2, LOW);

time = pulseIn(echopin2, HIGH);

S2= time\*0.034/2;

// Reading from third ultrasonic sensor

digitalWrite(triggerpin3, LOW);

delayMicroseconds(2);

digitalWrite(triggerpin3, HIGH);

delayMicroseconds(10);

digitalWrite(triggerpin3, LOW);

time = pulseIn(echopin3, HIGH);

S3= time\*0.034/2;

// Reading from fourth ultrasonic sensor

digitalWrite(triggerpin4, LOW);

delayMicroseconds(2);

digitalWrite(triggerpin4, HIGH);

delayMicroseconds(10);

digitalWrite(triggerpin4, LOW);

time = pulseIn(echopin4, HIGH);

S4= time\*0.034/2;

// Print distance values on serial monitor for debugging

Serial.print("S1: ");

Serial.print(S1);

Serial.print(" S2: ");

Serial.print(S2);

Serial.print(" S3: ");

Serial.print(S3);

Serial.print(" S4: ");

Serial.println(S4);

}

void signal1Function()

{

Serial.println("1");

low();

// Make RED LED LOW and make Green HIGH for 5 seconds

digitalWrite(signal1[0], LOW);

digitalWrite(signal1[2], HIGH);

delay(redDelay);

// if there are vehicels at other signals

if(S2<t || S3<t || S4<t)

{

// Make Green LED LOW and make yellow LED HIGH for 2 seconds

digitalWrite(signal1[2], LOW);

digitalWrite(signal1[1], HIGH);

delay(yellowDelay);

}

}

void signal2Function()

{

Serial.println("2");

low();

digitalWrite(signal2[0], LOW);

digitalWrite(signal2[2], HIGH);

delay(redDelay);

if(S1<t || S3<t || S4<t)

{

digitalWrite(signal2[2], LOW);

digitalWrite(signal2[1], HIGH);

delay(yellowDelay);

}

}

void signal3Function()

{

Serial.println("3");

low();

digitalWrite(signal3[0], LOW);

digitalWrite(signal3[2], HIGH);

delay(redDelay);

if(S1<t || S2<t || S4<t)

{

digitalWrite(signal3[2], LOW);

digitalWrite(signal3[1], HIGH);

delay(yellowDelay);

}

}

void signal4Function()

{

Serial.println("4");

low();

digitalWrite(signal4[0], LOW);

digitalWrite(signal4[2], HIGH);

delay(redDelay);

if(S1<t || S2<t || S3<t)

{

digitalWrite(signal4[2], LOW);

digitalWrite(signal4[1], HIGH);

delay(yellowDelay);

}

}

// Function to make all LED's LOW except RED one's.

void low()

{

for(int i=1; i<3; i++)

{

digitalWrite(signal1[i], LOW);

digitalWrite(signal2[i], LOW);

digitalWrite(signal3[i], LOW);

digitalWrite(signal4[i], LOW);

}

for(int i=0; i<1; i++)

{

digitalWrite(signal1[i], HIGH);

digitalWrite(signal2[i], HIGH);

digitalWrite(signal3[i], HIGH);

digitalWrite(signal4[i], HIGH);

}

}

# References

Density Based Traffic Control System using Arduino [https://create.arduino.cc/projecthub/muhammad-aqib/density-based-traffic-light-controller-using-arduino-8636ad]