TRAFFIC MANAGEMENT SYSTEM

PHASE-3 Project using **WOKWI** simulator.

*-GOKUL B*

*111421106015*

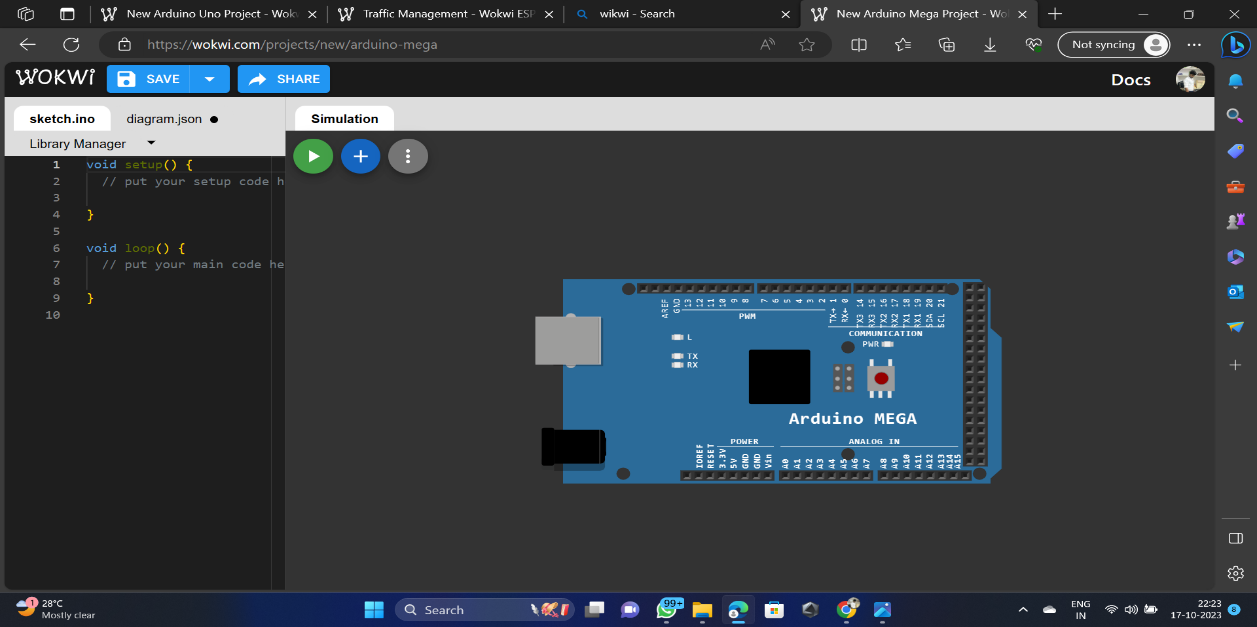
The smart Traffic management system is implemented by using the simulator WOKWI simulator.

This simulation is designed for a two-sides traffic road.

Step1:

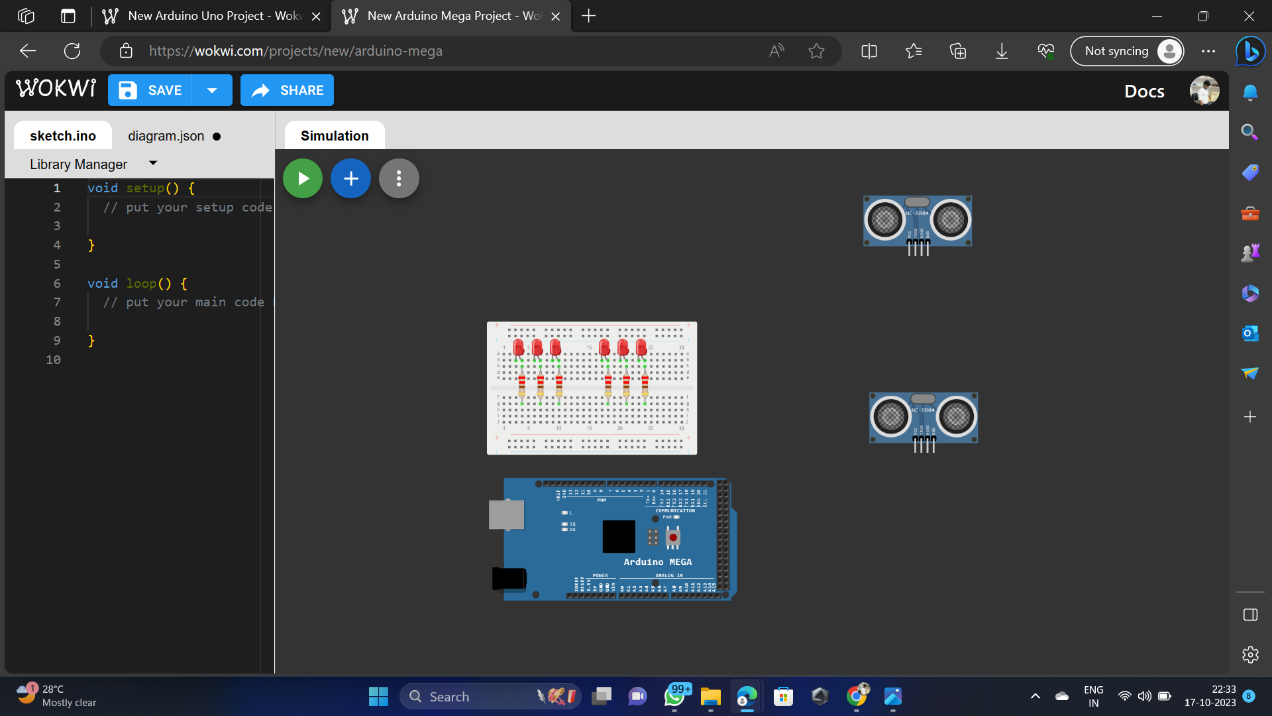
* + Start the WOKWI simulator. (<https://wokwi.com/>).
  + Log in to the side to create a new project, by clicking the “Create a New Project”.

Step2:



On the simulation area place an “ARDUINO MEGA” as controller of our system

Step3:

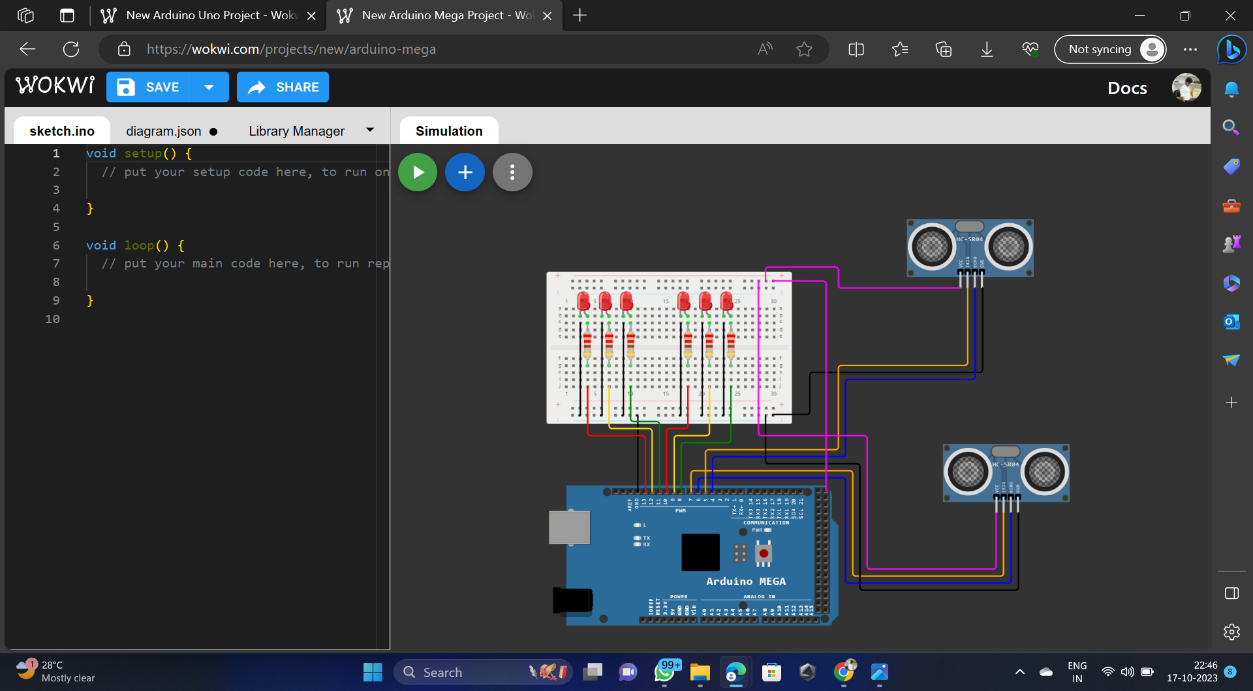


In step3 place the components required for the system from the component provider.

The required components are used in this system are:

* Bread board
* Ultra sonic sensors (HC-SR04)
* Resistors
* LED’s

Step4:



In step4, connections are completed according to the components requirements and our convinences.

Aurdunio mega pin connections:

* The LED to digital pin 2 and 3 will be the traffic lights of main road.
* The LED to digital pin 4 and 5 will be the traffic lights of side road.
* The uktra sonic sensor s connected to the pins of 5.2 v input terminal, ground pins to GND, the trx and rx are connected to the pins of 4,5,6,7 pins.
* The resistors are connected to the LED’s positive terminal and the negative terminalas are connected to the GND pin with help of bread board.

Step5:

In this step the coading is created according to the connections given to the aurduino mega with the pin configurations.

The 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);

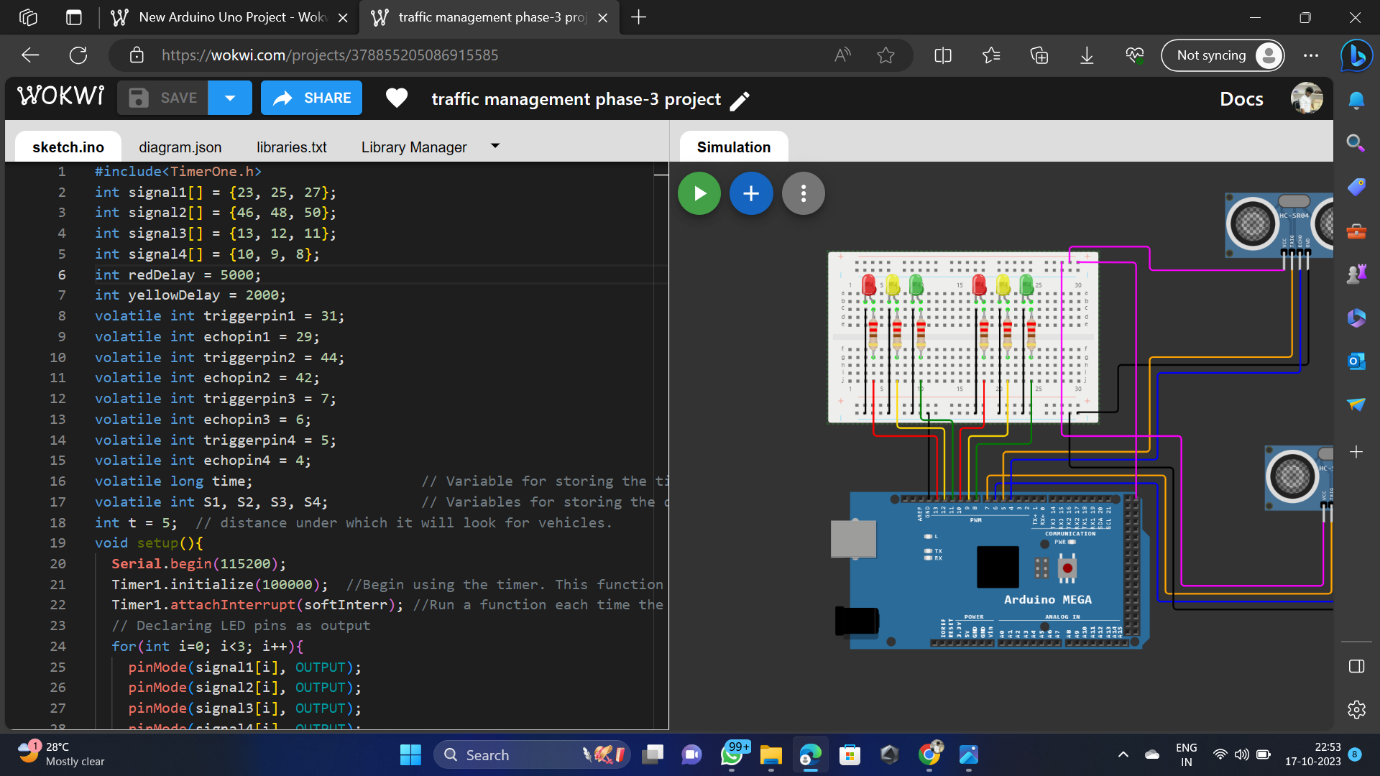
}

}

Then the code is created the simulation is ready to perform.

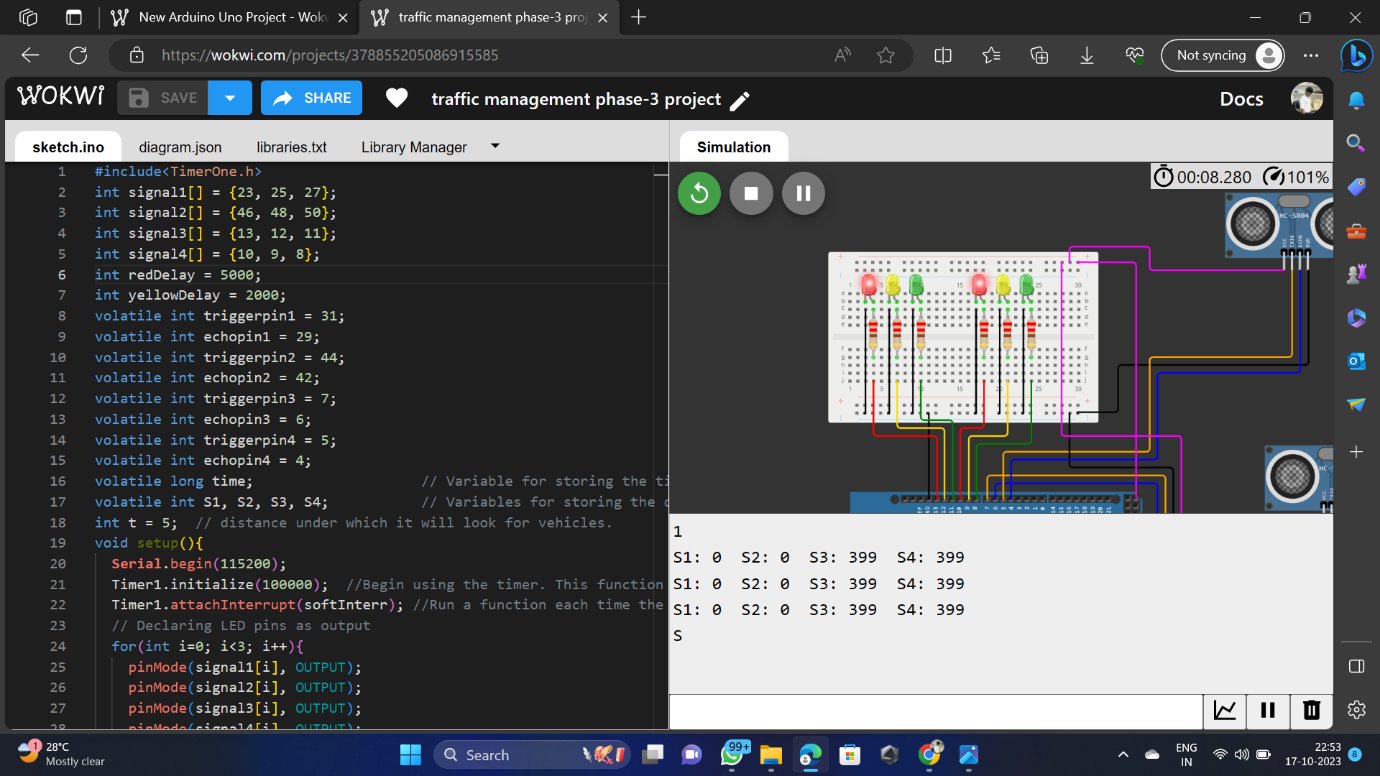
Step6:

In this step the simulaation is executed by cliking the run button



Then the last step is output progress for our system.

Step7:



Here the simulation is implemented successfully without any errors and simmulation issues.