School of Engineering And Applied Sciences

Analog And Digital Communications Lab - ECE211

Street and Traffic light Management using Zigbee

Group - 12 and 14

Guided By : Prof. Ashok Ranade

Abstract

This report focuses on issues of street light and traffic mangement. We have solved the problems of high usage of electricity of street lights and traffic density control using Zigbee and Arduino

Keywords: Zigbee, Arduino, Ultrasonic sensor, Photo register, Traffic lights (LEDs), Street Lights (LEDs)

1. Introduction

1.1. Background and motivation

The use of personal vehicles is getting very common nowadays and as a result increased traffic is becoming a severe problem in many cities. Vehicles on the road without any supervision or guidance can lead to traffic congestion and accidents. To monitor the flow of traffic we use traffic lights or traffic signals. Generally, a conventional traffic system is based on a fixed time concept allotted to each side of the junction which cannot be varied as per varying traffic density. In this case we cannot change the priority based on the greater number of vehicles waiting in a lane.
Apart from this traffic density problem, there is one more matter of concern and that is the street light system. Current street lights consume more power and energy and hence increases the maintenance cost.
Hence, here we also tried to propose a combined design and develop a density based traffic signal system with Automatic Street light system. The density based traffic signal system would prioritise the flow of traffic according to density (number) of vehicles and Automatic Street light system would take decisions for switching ON/OFF considering movement of vehicle or pedestrian and also surrounding light intensity.

1.2. Problem Statement

To overcome the current existing problem of more power consumption in the street light system, we try to develop a model which provides a safe night time environment for all road users including pedestrians [5]. The task of this model would be to function the lights only at night time and also to dim the street lights when no activity is detected as well as to brighten when any movement of vehicle is detected during night time. This can be achieved by using a device named ZIGBEE. Apart from ZigBee, sensors like photoresistor, microcontroller, ultrasonic sensor are also used. Here the photoresistor senses whether it is day time or night. The task of dimming and brightening lights is controlled by a microcontroller. If the photoresistor senses no light then the ultrasonic sensor will calculate the distance of the vehicle and and switch the light ON.

For the second problem of traffic signals, we try to design an arduino based Traffic Light Controller system where traffic signals would work on density of vehicles [3]. It is a simple implementation of a traffic lights system but can be extended to a real time system with programmable timings, pedestrian lighting etc. The real time traffic light controller is a complex piece of equipment which consists of a power cabinet, main controller or processor, relays, control panel with switches or keys, communication ports and many more. Hence, here in this project we design a simple traffic light system for a 4 way intersection using a microcontroller Arduino UNO where traffic is controlled in a predefined timing system. Also, here we use Zigbee to store and manage the data related to peak hours of high density traffic in backend. Although it is not the ideal implementation for real life scenarios, it gives an idea of the process behind the traffic light control system.

1.3. What is the need of ZIGBEE and how does it work?

ZigBee is an IEEE 802.15.4-based specification for a suite of high-level communication protocols used to create personal area networks with small, low -power digital radios. The technology defined by the ZigBee specification is intended to be simpler and less expensive than other wireless personal area networks (WPANs), such as Bluetooth or Wi-Fi. ZigBee is typically used in low data rate applications that require long battery life and secure networking. ZigBee devices can transmit data over long distances by passing data through a mesh network of intermediate devices to reach more distant ones. ZigBee has a defined rate of 250 kbit/s and has a range of 10 to 100m best suited for intermittent data transmissions from a sensor or input device. Here, in the Automatic Street Light system, the key principle of Zigbee technology is to control and monitor applications.
At the beginning of the development of the street lighting system, the lights were turned ON manually at night and turned OFF manually at morning. Sooner after that, a timer was used to turn ON and OFF the lights based on a pre-set time within the street light. Evolution in street light took place after the invention of light detecting sensors such as photodiode, photo resistor and photo relays. These sensors were mounted on the street light. Although, conventional street lights are powered by underground cable line that connects to the nearest distribution line. Zigbee based street lights are mostly battery powered, hence there is no need for laying underground cable lines.

Zigbee based street light consists of wireless sensor network application that utilizes Zigbee wireless communication protocol to enhance the technology of street lighting systems by providing communication capabilities. Zigbee based street lightning consists of 3 types of circuitry- sensor circuit, zigbee circuit and microcontroller circuit. The street lights would be controlled by a microcontroller and with the help of Zigbee microcontroller reports every action and status of the street light to the control panel wirelessly from the transmitter side to receiver . The host at the control station is able to monitor and control the street light all the time. Hence, Zigbee transfers all the information point by point to the control terminal. This also helps us to check the state of the street lamps and to take appropriate measures in case of failure.
Similary, Zigbee is also used in traffic light density to manage the data in beckend. It stores all the relevant data of peak hours of all traffic density and also time delay of green lights signals are stored using Zigbee.

2. Block Diagram

2.1. Automatic Street Lights

Block Diagram of Automatic Street Light

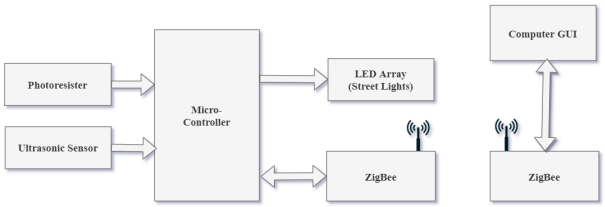

Figure 1: Block Diagram of Street Light management

2.2. Traffic Density Management

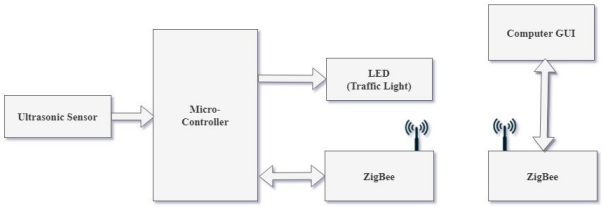

Figure 2: Block Diagram of Traffic Density management

3. Method of Testing and demonstrations

3.1. Flowchart

Automatic Street Light Management
Below flowchart shows the process in which the system will work in management of street and light thus reducing the power usage.

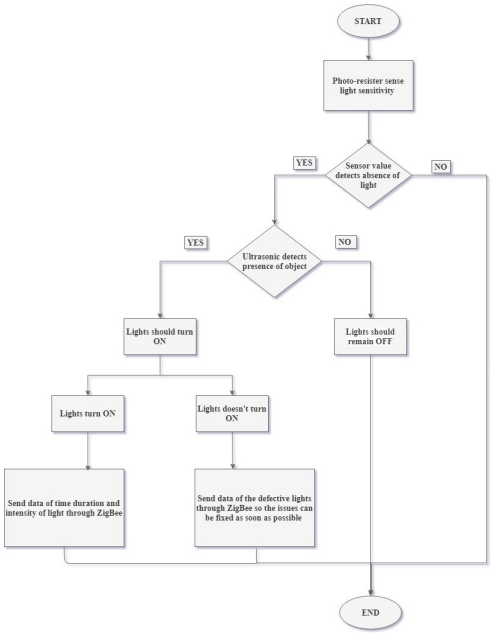

Figure 3: Flowchart of Street Light Management

Traffic Density Management
Below given flowchart shows the process in which the system will work in management of traffic density.

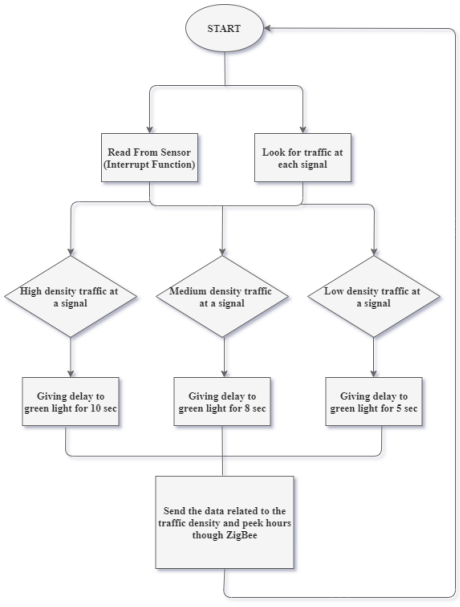

Figure 4: Flowchart of Traffic Density Management

3.2. Algorithm

3.2.1. Automatic Street Lights

Street lights used in this system are powered by solar energy instead of underground cable connecting to the nearest distributor line. The microcontroller is the brain of the overall system. It controls the lights and manages the data flow. The signals from sensors and Zigbee are sent to the microcontroller to perform immediate actions. The threshold from which microcontroller decides to switch ON/OFF lights is obtained from ADC value (Analog to digital converter value) [4]. The ADC value is actually the voltage that depends on the change in resistance of photoresistor sensor. Hence, the values obtained from ADC are directly proportional to the voltage divider value of both the photoresistor and the potentiometer in series with each other. If the surrounding light is brighter, then the resistance will be low and hence the value of ADC. The microcontroller used here is of 8 bit, hence the value of ADC obtained is between 0-255. The working procedure of the system is as follows-

ADC value	Lamp State	Environment
0-127	OFF	Bright
128-255	ON	Dark

1. The photoresistor gets activated if the ADC value is more than a particular threshold and sends signal to the microcontroller.
2. As soon as the photoresistor gets activated, ultrasonic sensor gets activated and sends signal to the microcontroller if any object (vehicle or pedestrian) enters into the detection range. The microcontroller then switches the lights ON.
3. Now, ZigBee device at transmission side is ready to receive information from streetlights and communicate with ZigBee device at receiver side, then sends to the terminal via USB cable.
4. ZigBee device communicates point-to-point to detect the faulty lights (if any) in the system.
5. Through GUI technicians can identify the faults and can easily maintain the system.

3.2.2. Traffic Density Management

The ultrasonic sensors used here can measure distance from 2 to 400 cm and hence we cannot use these sensors for an ideal implementation in real life scenarios but it gives an idea of the process behind the traffic light control system.
The working procedure is as follows-

1. Ultrasonic sensor measures the distance of the object (vehicles) and sends the signal to the microcontroller.
2. The microcontroller classifies the traffic into high density, medium density and low density and controls traffic light accordingly. There are

three possible cases here-

Case 1: Low traffic density
If the distance measured by an ultrasonic sensor is more than 200 cm then it is classified as low traffic at 4 signals and the system will stop at the current signal and will only move on the next signal if there will be traffic at any other signal. Also the time delay provided for green light signal is of 5 seconds.

Case 2: Medium traffic density
If the distance measured by an ultrasonic sensor is between than 150 - 200 cm then it is classified as medium traffic density and the system will skip the signal with less traffic and will move on to the next one. The time delay here provided for green light signal is of 8 seconds.

Case 3: High traffic density
If the distance measured by an ultrasonic sensor is less than 150 cm then it is classified as high traffic density and the system will work normally by controlling the signals one by one. Time delay of 10 seconds is provided for green light signal.

3. All the data related to traffic density, its peak hours and the individual time delay gets stored in PC through ZigBee in backend. Through this we can detect any fault in the time delay and can manage it.

3.3. Circuit Diagram and Working

3.3.1. Automatic Street Lights

ZigBee Interfacing with Arduino

We are not able to show Zigbee simulation as it was not available on an online simulator but we have tried to show the interface and connection of arduino with Zigbee manually [1 2].
Zigbee modules are capable of two types of communication – wireless communication and serial communication. A microcontroller can send data through the serial interface to the Zigbee module (transmitter) and the Zigbee module wirelessly transmits the data to another Zigbee module (Receiver). The receiver Xbee module transmits the data through the serial interface to controller, processor or PC to which it is interfaced. The controller interfaced to the Xbee module processes the information received by the Xbee devices. This way, controllers can monitor and control remote devices by sending messages through the local Xbee modules.

Circuit Diagram
We have worked on simulation of an Automatic Street Light System which uses an arduino uno (microcontroller). Photoresistor used here senses surrounding light, if there light is not detected then the ultrasonic sensor used here will calculate the distance of the object . If the object distance is less than 150 cm than arduino will turn on the lights . Else it will not turn the lights on.

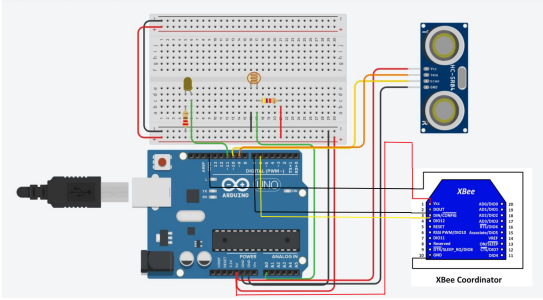

Figure 5: Circuit Diagram of Automatic Street light Manangement

3.3.2. Traffic Density System

As arduino mega was not available on online simulator, we implemented the circuit on arduino uno. Since arduino Uno had less number of connections, instead of 4 signals we were able to test and implement for 2 signals only. Two ultrasonic sensors are interfaced with the Arduino. Arduino will read from these sensors and will calculate the distance of the object and decide the density of the traffic and act accordingly. LED's are connected to the Arduino through the 220 ohm resistors. Here for each signal 3 LED'S are used representing yellow, green and red light. Also ZigBee is connected to transmitter side in order to transmit all the relevant data and store in the database at reciever side.

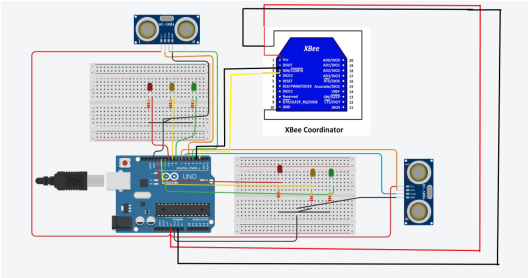

Figure 6: Circuit Diagram of Traffic Density System

3.4. Components

3.4.1. For Automatic Street Light System

1. Arduino Uno
2. Photoresistor
3. Ultrasonic Sensor HC-SR04
4. Resisitors
5. XBee Coordinator (Transmitter)

3.4.2. For Traffic Density System

1. Arduino Uno
2. Ultrasonic Sensor HC-SR04
3. LEDs(Taffic Lights)
4. Resisitors
5. XBee Coordinator (Transmitter)

4. Coding and Simulation

4.1. Automatic Street Light System

Implementation of the above circuit diagram of Automatic Street Light System is represented in the form of code. Here ZigBee is included in the implementation of code, but since there was no online simulator avalable that can simulate ZigBee, simulations do not connect ZigBee.

```
#include<SoftwareSerial.h>//include software serial to communicate with xbee

// Create an instance for software serial
SoftwareSerial xbee(7,6); // Digital pin 7-Xbee Rx, Digital pin 6-Xbee Tx
int photoresistorsensor = 0;
int ultrasonicsensor = 0;
long readUltrasonicDistance(int triggerPin, int echoPin)
{
pinMode(triggerPin, OUTPUT); // Clear the trigger
digitalWrite(triggerPin, LOW);
delayMicroseconds(2);
// Sets the trigger pin to HIGH state for 10 microseconds
digitalWrite(triggerPin, HIGH);
delayMicroseconds(10);
digitalWrite(triggerPin, LOW);
pinMode(echoPin, INPUT);
// Reads the echo pin, and returns the sound wave travel time in microseconds
return pulseIn(echoPin, HIGH);
}
void setup()
{
pinMode(A0, INPUT);
Serial.begin(9600);
pinMode(11, OUTPUT);
}
void loop()
{
photoresistorsensor = analogRead(A0); ultrasonicsensor = 0.01723 * readUltrasonicDistance(9, 10);
Serial.println("Distance:");
xbee.write("Distance:");
Serial.println(ultrasonicsensor);
xbee.write(ultrasonicsensor); // To send data to xbee reciever
delay(10); // Wait for 10 millisecond(s)
if (ultrasonicsensor < 150) {
analogWrite(11, map(photoresistorsensor, 0, 1023, 0, 255));
} else {
analogWrite(11, 0);
}
}
```

4.2. Traffic Density System

Implementation of the above circuit diagram of Traffic light density System is represented in the form of code.

```
#include<SoftwareSerial.h>//include software serial to communicate with xbee

// Create an instance for software serial
SoftwareSerial xbee(3,2); // Digital pin 3-Xbee Rx, Digital pin 2-Xbee Tx

// Declaring variables
int signal3[] = {13, 12, 11};// assigning pins for leds of signal S3
int signal4[] = {10, 9, 8};// assigning pins for leds of signal S4

int redDelay = 5000;// Red light Delay for low denisty traffic
int yellowDelay = 2000;// Standard yellow light delay

int redDelay2 = 8000;// Red light Delay for medium denisty traffic
```

```
14
15 int redDelay3 = 10000; // Red light Delay for high denisty traffic
16
17 String v; // string to assign density type for signal S3
18 String v2; // string to assign density type for signal S4
19
20 // assigning pins connected to ultrasonic sensor S3
21  volatile int triggerpin3 = 7;
22  volatile int echopin3 = 6;
23 // assigning pins connected to ultrasonic sensor S4
24  volatile int triggerpin4 = 5;
25  volatile int echopin4 = 4;
26
27  volatile long time; // Variable for storing the time traveled
28  volatile int S3, S4; // Variables for storing the distance covered
29
30 int t = 150; // distance under which it will look for vehicles .
31 int t2 = 200; // distance under which it will look for vehicles .
32
33 void setup(){
34   Serial.begin(115200);
35
36   // Declaring LED pins as output
37   for(int i=0; i<3; i++){
38     pinMode(signal3[i], OUTPUT);
39     pinMode(signal4[i], OUTPUT);
40   }
41
42   // Declaring ultrasonic sensor pins as output
43   pinMode(triggerpin3, OUTPUT);
44   pinMode(echopin3, INPUT);
45   pinMode(triggerpin4, OUTPUT);
46   pinMode(echopin4, INPUT);
47 }
48
49 void loop()
50 {
51    softInterr();
52
53
54   // If there are vehicles at signal 3
55   if(S3<t)
56   {
57     signal3Function_high(); // if at signal 3 traffic is at high density
58   }
59   else if(S3>t && S3<t2){
60     signal3Function_medium(); // if at signal 3 traffic is at medium density
61   }
62   else{
63     signal3Function_low(); // if at signal 3 traffic is at low density
64   }
65
66   // If there are vehicles at signal 4
67   if(S4<t)
68   {
69     signal4Function_high(); // if at signal 4 traffic is at high density
70   }
71   else if(S4>t && S4<t2){
72     signal4Function_medium(); // if at signal 4 traffic is at medium density
73   }
74   else{
75     signal4Function_low(); // if at signal 4 traffic is at low density
76   }
77
78 }
79
80 // This is function and it will run each time the timer period finishes .
81 void softInterr()
82 {
83   // Reading from third ultrasonic sensor
84   digitalWrite(triggerpin3, LOW);
85   delayMicroseconds(2);
86   digitalWrite(triggerpin3, HIGH);
87   delayMicroseconds(10);
88   digitalWrite(triggerpin3, LOW);
89   time = pulseIn(echopin3, HIGH);
90   S3= time*0.034/2;
91
92   // assigning type of traffic at signal 3
93   if(S3<150){
94   v="High";
95   }
96   else if(S3>150 && S3<200){
97   v="Medium";
98   }
99   else{
100  v="Low";
101  }
102
103  // Reading from fourth ultrasonic sensor
```

```
104     digitalWrite ( triggerpin4 , LOW);
105    delayMicroseconds(2);
106     digitalWrite ( triggerpin4 , HIGH);
107    delayMicroseconds(10);
108     digitalWrite ( triggerpin4 , LOW);
109    time = pulseIn(echopin4, HIGH);
110    S4= time*0.034/2;
111
112
113    // assigning type of traffic at signal 4
114    if(S4<150){
115    v2="High";
116    }
117    else if (S4>150 && S4<200){
118    v2="Medium";
119    }
120    else{
121    v2="Low";
122    }
123    // Print distance values on serial monitor
         for debugging
124     Serial . print ("\n Density on signal S3: ");
125     Serial . print (S3);
126     Serial . print (" S4: ");
127     Serial . println (S4);
128    // To send the data through xbee to computer
129      xbee.write("\n Density on signal S3: ");
130    xbee.write(S3);
131    xbee.write(" S4: ");
132    xbee.write(S4);
133 }
134
135
136 void signal3Function_low ()
137 {
138     Serial . println ("\n Signal 3: low");
139    low();
140     digitalWrite ( signal3 [0], LOW);
141     digitalWrite ( signal3 [2], HIGH);
142    delay(redDelay); // delay red light for low
         density
143     Serial . print ("\n Time taken for redlight :");
144  Serial . print (redDelay/1000);
145 xbee.write("\n Time taken for redlight :");
146 xbee.write(redDelay/1000);
147     Serial . print (" seconds");
148    xbee.write(" seconds");
149     if (S4<t)
150     {
151        digitalWrite ( signal3 [2], LOW);
152        digitalWrite ( signal3 [1], HIGH);
153       delay(yellowDelay); // delay yellow light
154     }
155 }
156
157 void signal3Function_medium()
158 {
159     Serial . println ("\n Signal 3: Medium");
160    xbee.write("\n Signal 3: Medium");
161    low();
162     digitalWrite ( signal3 [0], LOW);
163     digitalWrite ( signal3 [2], HIGH);
164    delay(redDelay2); // delay red light for
         medium density
165     Serial . print ("\n Time taken for redlight :");
166  Serial . print (redDelay2/1000);
167     Serial . print (" seconds");
168
169 xbee.write("\n Time taken for redlight :");
170 xbee.write(redDelay2/1000);
171    xbee.write(" seconds");
172
173     if (S4<t)
174     {
175        digitalWrite ( signal3 [2], LOW);
176        digitalWrite ( signal3 [1], HIGH);
177       delay(yellowDelay); // delay yellow light
178     }
179 }
180
181
182
183 void signal3Function_high ()
184 {
185     Serial . println ("\n Signal 3: high");
186    xbee.write("\n Signal 3: high");
187    low();
188     digitalWrite ( signal3 [0], LOW);
189     digitalWrite ( signal3 [2], HIGH);
190    delay(redDelay3); // delay red light for high
         density
191     Serial . print ("\n Time taken for redlight :");
192  Serial . print (redDelay3/1000);
193     Serial . print (" seconds");
194
195 xbee.write("\n Time taken for redlight :");
196 xbee.write(redDelay3/1000);
197    xbee.write(" seconds");
198
199     if (S4<t)
200     {
201        digitalWrite ( signal3 [2], LOW);
202        digitalWrite ( signal3 [1], HIGH);
203       delay(yellowDelay); // delaying yellow light
204     }
205 }
```

```
206
207 void signal4Function_low ()
208 {
209    Serial . println ("\n Signal 4: Low");
210   xbee. write ("\n Signal 4: Low");
211   low();
212    digitalWrite ( signal4 [0], LOW);
213    digitalWrite ( signal4 [2], HIGH);
214   delay(redDelay); // delay red light for low
        density
215    Serial . print ("\n Time taken for redlight :");
216 Serial . print (redDelay/1000);
217    Serial . print (" seconds");
218
219 xbee. write ("\n Time taken for redlight :");
220 xbee. write (redDelay/1000);
221   xbee. write (" seconds");
222
223   if (S3<t)
224   {
225      digitalWrite ( signal4 [2], LOW);
226      digitalWrite ( signal4 [1], HIGH);
227     delay(yellowDelay); // delaying yellow light
228   }
229 }
230
231
232 void signal4Function_medium()
233 {
234    Serial . println ("\n Signal 4: Medium");
235
236   xbee. write ("\n Signal 4: Medium");
237   low();
238    digitalWrite ( signal4 [0], LOW);
239    digitalWrite ( signal4 [2], HIGH);
240   delay(redDelay2); // delay red light for
        medium density
241    Serial . print ("\n Time taken for redlight :");
242 Serial . print (redDelay2/1000);
243    Serial . print (" seconds");
244
245 xbee. write ("\n Time taken for redlight :");
246 xbee. write (redDelay2/1000);
247   xbee. write (" seconds");
248
249   if (S3<t)
250   {
251      digitalWrite ( signal4 [2], LOW);
252      digitalWrite ( signal4 [1], HIGH);
253     delay(yellowDelay); // delaying yellow light
254   }
255 }
256
```

```
257
258 void signal4Function_high ()
259 {
260    Serial . println ("\n Signal 4: High");
261
262   xbee. write ("\n Signal 4: High");
263   low();
264    digitalWrite ( signal4 [0], LOW);
265    digitalWrite ( signal4 [2], HIGH);
266   delay(redDelay3); // delay red light for high
        density
267    Serial . print ("\n Time taken for redlight :");
268 Serial . print (redDelay3/1000);
269    Serial . print (" seconds");
270
271   xbee. write ("\n Time taken for redlight :");
272 xbee. write (redDelay3/1000);
273   xbee. write (" seconds");
274
275   if (S3<t)
276   {
277      digitalWrite ( signal4 [2], LOW);
278      digitalWrite ( signal4 [1], HIGH);
279     delay(yellowDelay); // delaying yellow light
280   }
281 }
282
283 // Function to make all LED's LOW except RED
      one's.
284 void low()
285 {
286   for( int i=1; i<3; i++)
287   {
288      digitalWrite ( signal3 [i ], LOW);
289      digitalWrite ( signal4 [i ], LOW);
290   }
291   for( int i=0; i<1; i++)
292   {
293      digitalWrite ( signal3 [i ], HIGH);
294      digitalWrite ( signal4 [i ], HIGH);
295   }
296 }
```

Note: The video of all the above two simulation is attached in the link

5. References

1. “XBee S2 (ZigBee) Interfacing with Arduino UNO” ElectronicWings, https://www.electronicwings.com/arduino/xbee-s2-zigbee-interfacing-with-arduino-uno

2. Priya.”IoT Communication between two devices over Zigbee Protocol : IOT Part 37” ENGINEERSGARAGE,WTWH Media LLC, 7 March 2018, https://www.engineersgarage.com/contributions/iot-communication-between-two-devices-over-zigbee-protocol-iot-part-37/

3. Muhammad,Aqib.”Density Based Traffic Light Controller Using Arduino” Project Hub,23 Dec 2018, https://create.arduino.cc/projecthub/muhammad-aqib/density-based-traffic-light-controller-using-arduino-8636adtoc-circuit-diagram-density-based-traffic-light-controller-using-arduino-2

4. Ansari,Faiz , et al. “Zigbee Based Smart Street Lighting System .” International Journal of Scientific Engineering Research, Volume 7, Issue 2, February-2016, https://www.ijser.org/researchpaper/Zigbee-Based-Smart-Street-Lighting-System.pdf

5. M,Srikanth,et al. ”ZigBee Based Remote Control Automatic Street Light System”, IJESC, June 2014

Contributors

Priyanshi Shah
AU1841009

Hardi Kadia
AU1841059

Nancy Radadia
AU1841070

Varshil Shah
AU1841095

Kahaan Patel
AU1841110

Suhanee Patel
AU1841113

Vidit Vaywala
AU1841128

Dhatri Kapuriya
AU1841129

Hemil Shah
AU1841135