Smart Intervehicular Communication

With applications in auto light adjustments, information network formation, accident prevention etc.

ECE3501- IoT Fundamentals

Report

by

Harisha S - 20BRS1127

Sachit Nair - 20BRS1140

Alan Shaji - 20BRS1024

Abstract:

As of the present moment, vehicles have very little means of communicating with each other besides the use of horns or visual and auditory indicators. This, however only allows for a very basic and limited form of information transmission. We think with the integration of smart technologies, a whole lot more can be achieved.

There would be many applications and use cases like automatically dimming the lights when a signal from a vehicle that's speeding opposite to you is received, or much faster transmission of critical information from one vehicle to another when accidents or heavy roadblocks occur. We may also use this communication to reduce the number of deadly vehicular collisions by having the vehicles automatically slow down when a collision seems unavoidable.

We would be building on existing technologies like VANETs to come up with a solution to implement inter-vehicular communication. Vehicles are undoubtedly a very important part of our lives, and innovating them will surely lead us closer to achieving smarter and safer cities on a wider scale.

Concepts Involved

The aim of the project is to develop a system that can automatically dim the lights of a car speeding down the opposite side when its headlights are too bright, we simulate this situation using 2 Arduionos to represent the cars.

The crux of the project involves the measurement of data on one node, processing that information, drawing inferences, and then finally transmitting data to the other node, where some action will be taken.

Sensing:

The measured attribute in this case was the light intensity. An LDR (Light Dependent Resistor) was used to accomplish this.

Communication:

The major task involved in this project was the transfer of information from one node to another. In order to accomplish this, we made use of the NRF24L01 transceiver module. The NRF24L01 module is a wireless module that works in the 2.4 GHz band, it communicates by using the 4 SPI with a maximum data rate of 10Mbps. 2 nodes on the same address may communicate with each other using this module.

Processing:

The node classifies the detected light intensity as either normal or too bright. If it is too bright, a packet is sent to the other node containing the measured details along with a request to reduce the intensity of the emitted light.

Upon receiving such a packet, the node checks the intensity of the light emitted by it to know if it is the source of the light, if it is found that it is indeed emitting high-intensity light when it is not required, the intensity of the emitted light is reduced.

Thus, based on measured sensor values, inferences are made and this information is transmitted using which a decision is taken after validation.

Code

Transmitter Code:

```
#include <SPI.h> // to handle the communication interface with the
modem
#include "RF24.h" // the library which helps us to control the radio
modem

RF24 myRadio (9, 10); //creating radio instance (CE, CSN) pins
byte addresses[][6] = {"0"}; //address to which data is transmitted

//for the collision avoidance communication
#define echoPin 2
#define trigPin 3

//creating a struct package to send the data
//contents include data packet id, light intensity detected by the
//car, and whether the car is stopping due to collision avoidance
//mechanism
struct package
{
  int id=1;
  float lightIntensity = 350;
  bool amStop = false;
  char  text[100] = "Light intensity HIGH!";
};


typedef struct package Package;
Package data;

//ldr pin connected to receive light intensity is connected to the
analog pin A0
int ldr = A0;

//variables required for collision avoidance
long duration;
int distance;

void setup()
{
  //sensors
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(ldr, INPUT);

  //Serial Monitor initialized with baud rate as 115200
  Serial.begin(115200);

  //Radio
  myRadio.begin(); //Activate the modem
  myRadio.setChannel(115); Setting the communication channel
  myRadio.setPALevel(RF24_PA_MAX); //Power Amplification Level set
max
  myRadio.setDataRate( RF24_250KBPS ) ; //Data rate of communication
  myRadio.openWritingPipe( addresses[0]); //reading pipe opened with
the receiver’s address

}

void loop()
{
    //Calculating the distance between car and obstacle
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);

    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    duration = pulseIn(echoPin, HIGH);
    distance = duration * 0.034 / 2;

    //If the distance is less than 10cm, The car stops and sends a
    //message to the nearby car that it has stopped due to an
obstacle
    if(distance <= 10){
      Serial.println("Obstacle detected, I do be stopping");
      data.amStop = true;
      data.lightIntensity = analogRead(ldr);
      //changing the package text according to the scenario
      strncpy(data.text,"Nearby Vehicle Stopping",
sizeof(data.text));


      //transmitting the package to the receiver’s address.
Parameters
      //given are the data to be sent, and the size of the data
packet
      myRadio.write(&data, sizeof(data));
      data.id = data.id + 1;
    }
    Else{
      //In case no obstacles, then this mechanism isn’t activated
      data.amStop = false;
    }

    Serial.print("\nPackage:");
    Serial.print(data.id);
    Serial.print("\n");
    Serial.println(data.lightIntensity);
    Serial.println(data.text);

    //ldr sensor reading
    data.lightIntensity = analogRead(ldr);

    //low ldr sensor output -> Bright
    //high ldr sensor output -> Dull
    // Brighter the light, smaller the sensor value
    //if ldr value less than 300, then the car transmits a message
    if(ldr <= 300){
      strncpy(data.text, "Light intensity HIGH!", sizeof(data.text));
      myRadio.write(&data, sizeof(data));
      data.id = data.id + 1;
    }

    delay(5000);

}
```

Receiver Code:

```
#include <SPI.h>
#include "RF24.h"

int ledPin= 5; //pin where the led is connected for control

RF24 myRadio (9, 10); //Radio initializing (CE, CSN) pin
struct package
{
  int id=0;
  float intensity = 0;
  bool carStop=false;
  char  text[100] ="empty";
};


byte addresses[][6] = {"0"}; //Address of itself (Receiver)
```

```
typedef struct package Package;
Package data;

void setup()
{
  Serial.begin(115200);
  delay(1000);

  pinMode(ledPin,OUTPUT);

  myRadio.begin();
  myRadio.setChannel(115);
  myRadio.setPALevel(RF24_PA_MAX);
  myRadio.setDataRate( RF24_250KBPS ) ;
  //the radio module is actively listening for messages
  myRadio.openReadingPipe(1, addresses[0]);
  myRadio.startListening();
}

int flag = 1;

int myIntensity = 255;


void loop()
{
  //LED is ON when there is no message from other cars that the light
  //brightness is high
  if(flag==1)
  analogWrite(ledPin, myIntensity);

  // myRadio.available() checks whether there is data to be received
  if ( myRadio.available())
  {
    while  (myRadio.available())
    {
      //read the data and store it in the package variable data
      myRadio.read( &data, sizeof(data) );
    }

    //print the message received
    Serial.println();
    Serial.println("My Intensity: " + String(myIntensity));
    Serial.print("Message recieved : " + String(data.text));
    Serial.print("\nReceiving Package:");
    Serial.print(data.id);
    Serial.print("\n");
    Serial.print("Intensity: ");
    Serial.println(data.intensity);
    //if the nearby car is experiencing high intense light, and the
    //intensity of self is high, the light is dimmed

    if(data.intensity <= 300 and myIntensity > 125){
      analogWrite(ledPin, 0);
      Serial.println("----DIMMING LIGHTS-------");
      flag=0; //light dimmed till no "high intensity" message
received
                  //from the surrounding
  }else{
    flag=1;
    Serial.println("----NO LONGER BRIGHT, SO LED ON-------");
  }

  //if the nearby car is stopping, then self is slowing down
  if(data.carStop==true){
        Serial.println("I am slowing down");
  }

}

  //generating varied intensity of self to demonstrate working of the
  //concept
  myIntensity -= 50;
  if(myIntensity < 50){
    myIntensity  = 255;
  }
}
```

Hardware

Components

Arduino Uno (x2)

Breadboard (x2)

NRF24L01+ modules (x2)

Ultrasonic Sensor (x2)

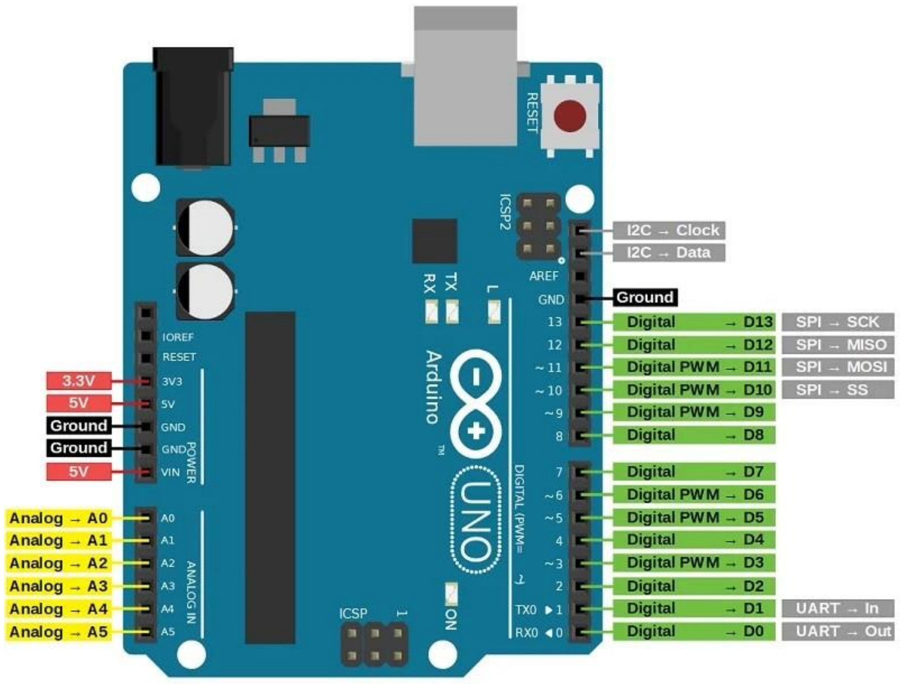
LDR sensor (x2)

LED (x2)
Resistor (10k Ohm) (x2)

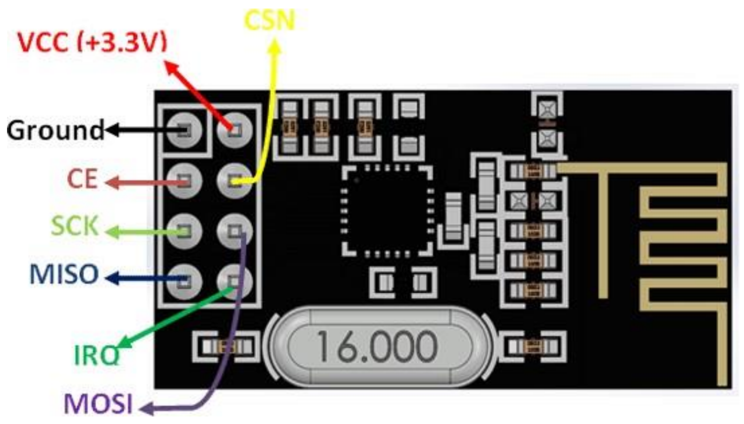
Connecting wires

Pin Diagrams

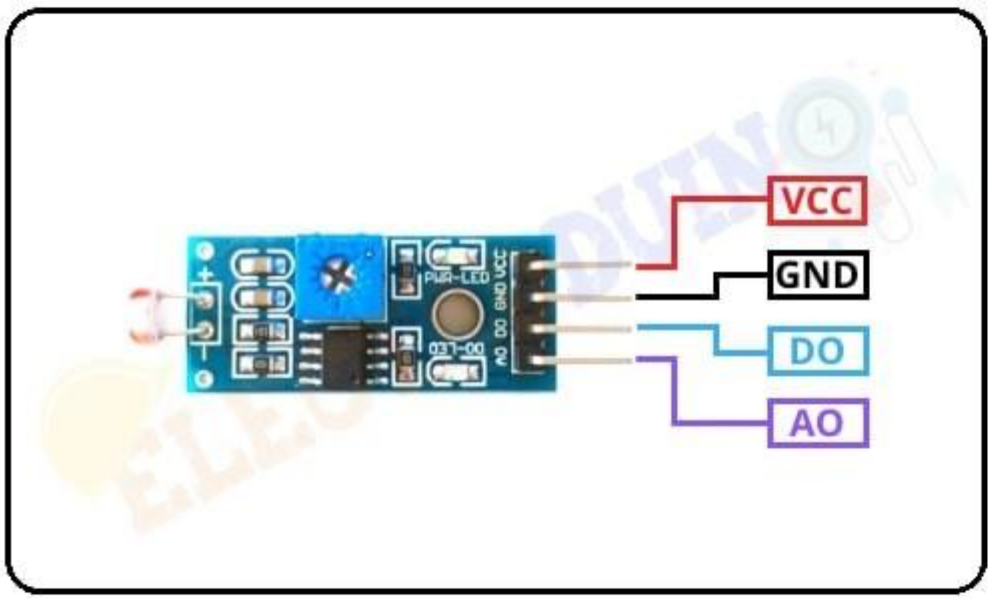
Arduino Uno:



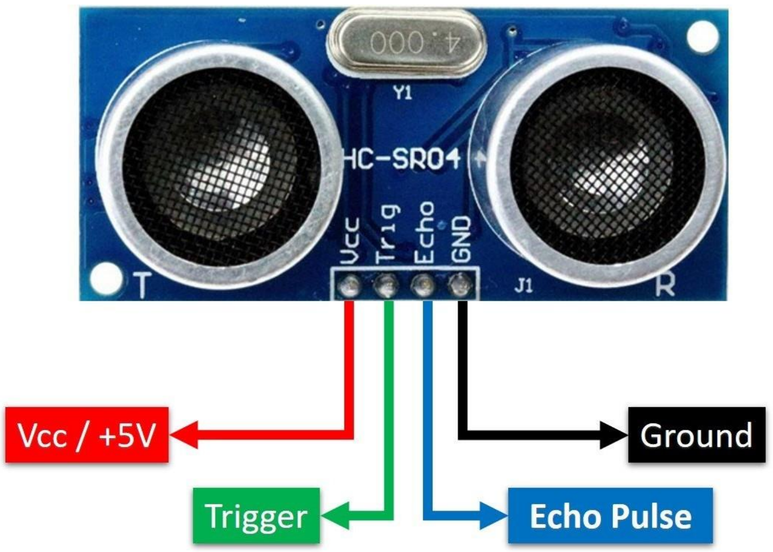
NRF24L01:



LDR sensor:



Ultrasonic sensor:



Connections:

NRF24L01 to Arduino UNO:

VCC	3.3V through breadboard
GND	GND through breadboard
SCK	D13
MISO	D12
MOSI	D11
CSN	D9
CE	D10

LDR sensor to Arduino UNO:

VCC	+3.3 v through breadboard
GND	Ground (-) through breadboard
DO	A0

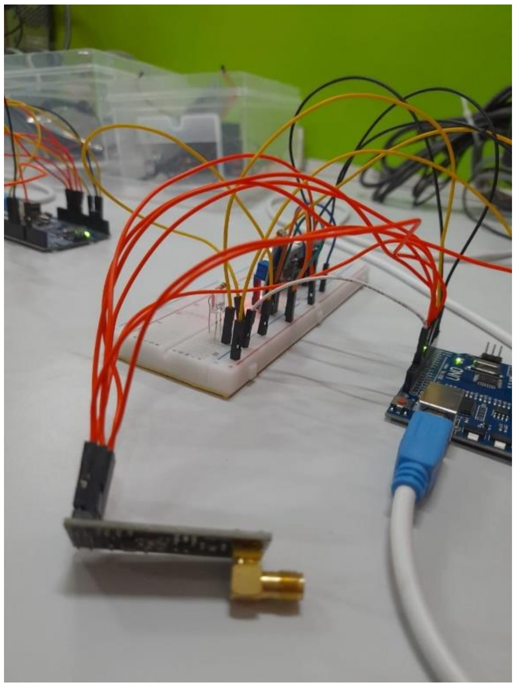
LED for receiver side:

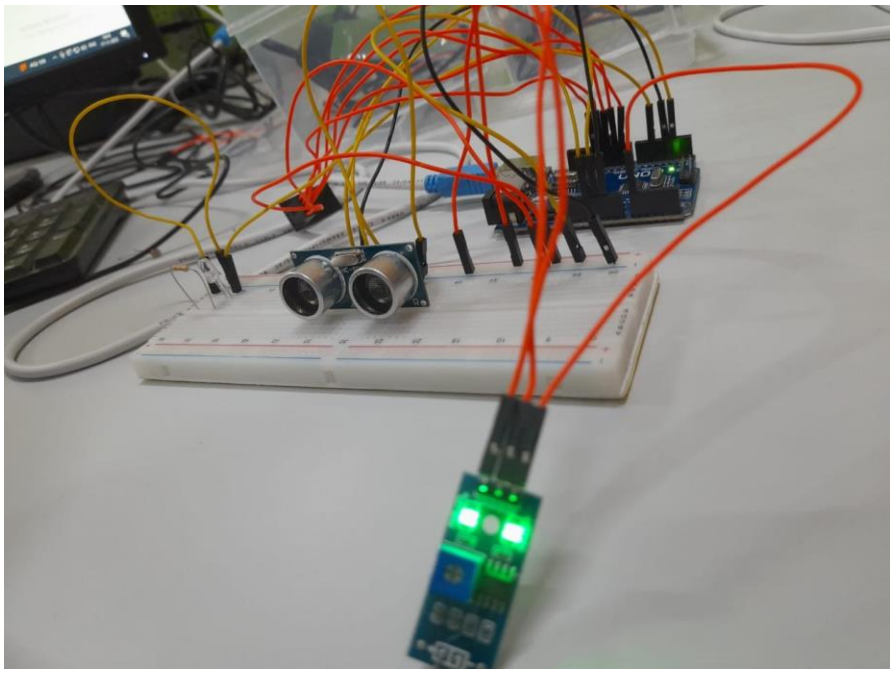
+ve	D5
GND	Ground (-) through breadboard

Ultrasonic sensor for transmitter side:

VCC	+3.3 v through breadboard
GND	Ground (-) through breadboard
Trig	D3
Echo	D2

Results





Demo Link:

https://drive.google.com/file/d/1fbAPWyvmL_zlNSXxN1oS5BVY7o-UGlyf/view?usp=sharing

Future Scope:

With this project we have established communication between two vehicles and the NRF modules we have used have a range of 800-1000m.This prototype alone opens up a huge field for innovation and is the first step to Internet of Vehicles(IoV). Currently our application is to dim the headlights of oncoming vehicles if the intensity of the oncoming vehicle is above a threshold value. We can extend the application to traffic guidance systems, safe navigation, intelligent vehicle control, crash prevention, electronic toll collection, traffic flow monitoring, and even vehicle autonomy.