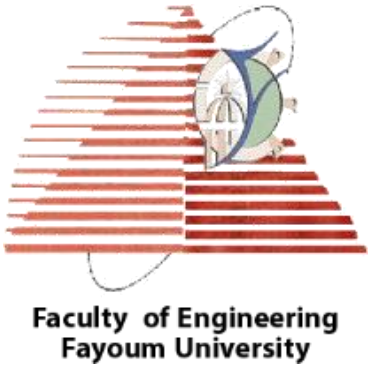
Fayoum University

Engineering Faculty

Electrical Engineering Department



B. Eng. Final Year Project

Vehicle-to-Everything Communication

By:

Diaa Abdeltoab Ramdan

Mohamed Rageb Mohamed

Mohamed Alaa Rushdi

Mustafa Mohamed Shaban

Ahmed Mohamed Rabia

Khaled Atef Mahmoud

Supervised By:

Dr. Mohamed Hamdy

Supervisor(s)	*Date of examination*
DR. Mohamed Hamdy	*2, July 2024*

J u l y – 2 0 2 4

DEDICATION

We dedicate this project to our families, who are our first and last supporters, and this support includes moral support, psychological support, and financial support, we thank you for everything you have provided us.

We also dedicate it to our brothers, sisters, and friends who did not hesitate to provide us with any information or assistance, thank you all.

ACKNOWLEDGMENT

The success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

The special thanks go to our helpful supervisor **Dr. Mohamed Hamdy** the supervision and support he gave truly helped in the progression of our graduation project. The cooperation is much indeed appreciated. We also take this opportunity to express our gratitude to the board of the Faculty of Computers and Information for their Support and efforts in providing us with all useful resources.

We express a special thanks to the Faculty of Computers and Information stuff for the valuable information provided by them in their respective fields.
Lastly, we thank Almighty, Our Parents and all the people who helped, supported and encouraged us to successfully finish the graduation project whether they were in the university or in the industry.

DECLARATION

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Bachelor of Science in *Electrical Engineering* is entirely my own work, that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: *Diaa Abdeltoab Ramdan*

Registration No.: 4820

Date: 2 , July 2024

Signed: *Mohamed Rageb Mohamed*

Registration No.: 4829

Date: 2 , July 2024

Signed: *Mohamed Alaa Rushdi*

Registration No.: 4831

Date: 2 , July 2024

Signed: *Mustafa Mohamed Shaban*

Registration No.: 4837

Date: 2 , July 2024

Signed: *Ahmed Mohamed Rabia*

Registration No.: 4904

Date: 2 , July 2024

Signed: *Khaled Atef Mahmoud Mouhamed*

Registration No.: 4910

Date: 2 , July 2024

ABSTRACT

At this modern age, wireless communication is affecting every aspect of life and its current advancement is incorporating the vehicular operations. This graduation project aims to explore and implement Vehicle-to-Everything (V2X) communication technology to enhance road safety and traffic efficiency. V2X is a groundbreaking technology that allows vehicles to communicate with each other, infrastructure, and pedestrians, enabling the exchange of critical information for improved traffic management and safety. This project will involve the development and testing of a V2X system prototype to demonstrate its potential benefits in real-world scenarios and in brief we will talk about features in V2X

- V2P (Vehicle-to- pedestrian)
    - Vehicles can warn pedestrians themselves of the car
    - a pedestrian perspective can contain comprehensive information about the vehicle.
- V2I(Vehicle-to-Infrastructure)
    - Vehicles receive real-time data about traffic signal timings,
      enabling them to adjust their speed for better traffic flow and fuel efficiency .
    - Vehicles receive warnings about approaching red-light violations or pedestrians in crosswalks.
- V2G(Vehicle-to-Grid)
    - Electric vehicles (EVs) can communicate with the power grid to manage energy consumption and support grid stability through vehicle-to-grid (V2G) technology

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS/ABBREVIATIONS

ADC	*Analog to Digital Converter*
CPU	*Central Processing Unit*
DAC	*digital to analog converter*
ECU	*Electronic Control Unit*
IEEE	*Institute of Electrical and Electr*onics Engineers
IOT	*Internet of Things*
I2C	*Inter-Integrated Circuit*
LCD	*liquid-crystal display*
MC	*Microcontroller*
PWM	*Pulse-width modulation*
SPI	*Serial Peripheral Interface*
UART	*Universal Asynchronous Receiv*er Transmitter
GPS	*Global Positioning System*
V2I	v*ehicle-to-infrastructure*
V2P	v*ehicle-to-pedestrian*
V2G	v*ehicle-to-Grid*
EV	E*lectric Vehicle*
SECC	*Charging Station*
VIN	*Vehicle Identification Number*
TFT	thin-film transistor

Chapter One

1 INTRODUCTION

1.1 STATEMENT OF THE PROBLEM

The Problems Like:

- Traffic Congestion.
- Road unsafety and Accidents.
- Pedestrian and Cyclist Safety.
- Environmental Impact and Emissions.
- Emergency Response Efficiency.
- Limited Connectivity in Remote Areas communication system.

We Founded the following regardless the safety, mobility and environment Safety: 5 million Crashes/year, Million injuries, 30,000 deaths/year, Leading cause of death in ages 4 to 34.

Mobility: 5 billion hours of travel delay, 100 billion cost of urban congestion.

Environment: 3 billion gallons of wasted fuel.

1.2 PROJECT MOTIVATIONS AND GOALS

1.2.1 Motivation

The motivation of this project is to reduce injuries and fatalities due to collision by increase drive range of visibility and alertness on the road.

1.2.2 Project Goals

The main focus of this project is to maintain reliable communication between vehicle, infrastructure, Grid and Pedestrian provide accurate positioning information, calculate condition for possible collusion, and alert the driver and pedestrian early enough for effective reaction time.

The goals of this project include but are not limited to the following:

- Reducing traffic congestion by providing real-time traffic information and optimizing traffic flow.
- Improving road safety.
- Enhancing the energy efficiency of vehicles.
- Maintaining a time-efficient system in responding to road incidents.

1.3 PROJECT POTENTIAL

V2X which stands for (vehicle to everything), is the umbrella term for the car's communication system. There are several components of V2X vehicle-to-infrastructure (V2I), vehicle-to-pedestrian (V2P), vehicle-to-Grid (V2G).

In this multifaceted ecosystem, cars will talk to other cars, to infrastructure such as traffic lights or parking spaces and to smartphone-toting pedestrians. Different use cases will have different sets of requirements, which the communications system must handle efficiently and cost-effectively and with infrastructure.

V2P communication enables direct communication between a vehicle and a pedestrian. The scope of V2P may also apply to other vulnerable road users, such as cyclists. Signals are transmitted when pedestrians are near a motor vehicle.

- Warnings warn drivers of approaching pedestrians or warn pedestrians themselves of the car, for this we need (sensors) to establish a stable V2P connection.
- Regular warnings or safety instructions from a pedestrian perspective can contain comprehensive information about the vehicle.

It informs you about the speed, position and direction of an approaching car. potential pedestrians use a smartphone to establish communication either by Wi-Fi network or communicate to the cloud via cellular network (3G, LTE) and vehicles utilize either a cellular module (with a dedicated SIM-card) or driver’s smartphone.

V2I is a communication model that allows vehicles to share information with the components that support a country's highway system. Such components include overhead RFID (radio frequency identification) readers and cameras, traffic lights, lane markers, streetlights, signage and parking meters. V2I communication is typically wireless and bi-directional: data from infrastructure components can be delivered to the vehicle. Similar to vehicle-to-vehicle (V2V) communication, V2I uses dedicated short-range communication (DSRC) frequencies to transfer data.

V2G communication, People see an electric car’s battery as an extension of the electrical grid, with vehicle-to-grid, an electric vehicle user can therefore decide to store electricity when rates are the lowest, then use it when the price goes up. EV batteries are by far the most cost-efficient.

With V2G, we can utilise the battery capacity up to 10x more efficiently than with regular smart charging. Vehicle-to-grid technology enables us to make the best use of the existing population of vehicles.

The main components of V2G:

1- Electric Vehicle (EV): EV is the core components of V2G.
2- Communication Infrastructure: A communication network is essential for exchanging data between EVs and V2G chargers.
3- V2G Management System: This software platform manages V2G operations.
4- BMS (Battery Management System) & Dashboard.
5- SECC (Charging Station).

Chapter Two

2 HOW PROJECT WORK

In our project we are creating a smart vehicle system that uses V2X (Vehicle-to-Everything) communication to improve safety and traffic efficiency. Here's how each component of the system works together

Each car is equipped with an STM microcontroller and a TFT display. The STM controls the operations, and the TFT display shows important information to the driver. The car also has an ESP module that connects to Wi-Fi. This module helps the car communicate with the Firebase cloud. Every car has a GPS module that provides real-time location data. Electric vehicles (EVs) have a system to monitor battery levels and manage charging and discharging.

2.1 CAR TO FIREBASE COMMUNICATION:

- The car's GPS module constantly updates its location.
- The ESP module connects to Wi-Fi and sends this location data to the Firebase cloud.
- Other important vehicle data, like speed and direction, are also sent to Firebase.

- **Interaction with Infrastructure:**

  When a car enters the range of a traffic light or road sign, the infrastructure sends updates to the car's TFT display. For example:

  - **Stop Signs:** The display shows a stop sign when approaching one.
  - **Speed Limits:** The display shows the current speed limit.
  - **Traffic Lights:** The display indicates red, yellow, or green lights.

  If a traffic light detects an accident nearby, it sends a warning to the emergency services and updates nearby car

- **Battery management system:**

    - When an EV enters a charging station, the station communicates with the car's battery management system.
    - The TFT display shows the current battery percentage.
    - The system checks the price for charging and displays it to the driver.
    - The driver can choose to charge the battery, and the station begins charging.
    - Payment information is handled through the mobile application, with updates sent to Firebase.

- **V2X Mobile Application:**

    - Allows the driver to share their vehicle's data with Firebase.
    - Receives updates from Firebase about traffic conditions and alerts.
    - Pedestrians can receive alerts about approaching vehicles, making it safer to cross roads.
    - The app provides real-time traffic updates, helping pedestrians navigate safely.

- **Car movement:**

    - Vehicle in our project is controlled by mobile application using Bluetooth communication between mobile application and Bluetooth module connected to vehicle with UART protocol, and interrupt algorithm is used to detect change in direction of vehicle.
    - Mobile application has four buttons for four directions of vehicle, each one has specific capital letter is sent to vehicle by Bluetooth, F for forward direction, B for backward direction, R for right direction and L for left direction.
    - GUI in vehicle is updated automatically based on current direction of vehicle, if no button is pushed vehicle stopped.

- **The Controlling Mobile application:**

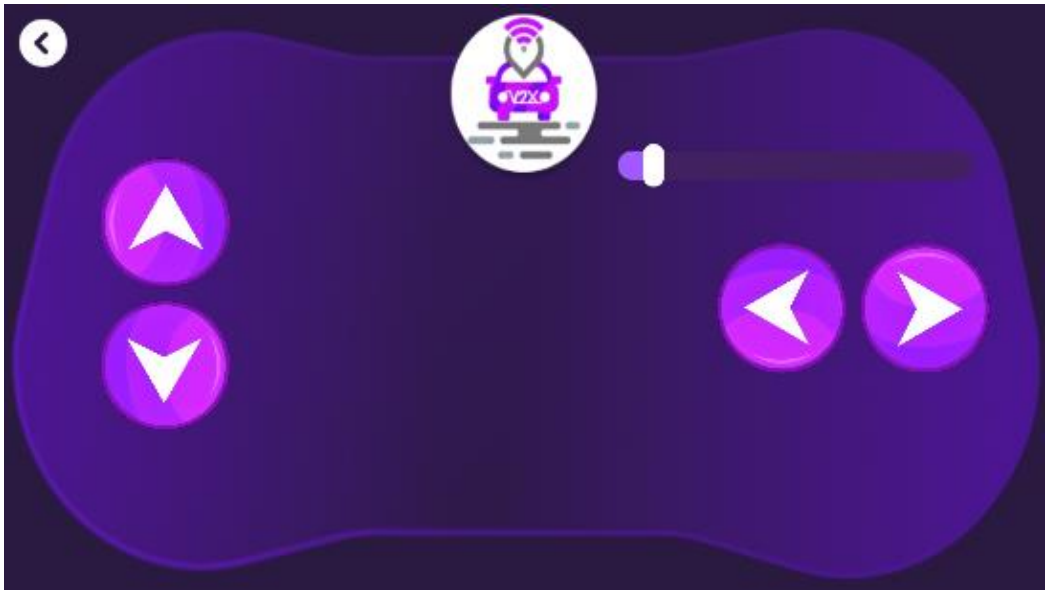


figure 1control vehicle

- **The GUI in vehicle :**

figure 4GUI-vehicle move to forward

figure 5GUI-vehicle stop

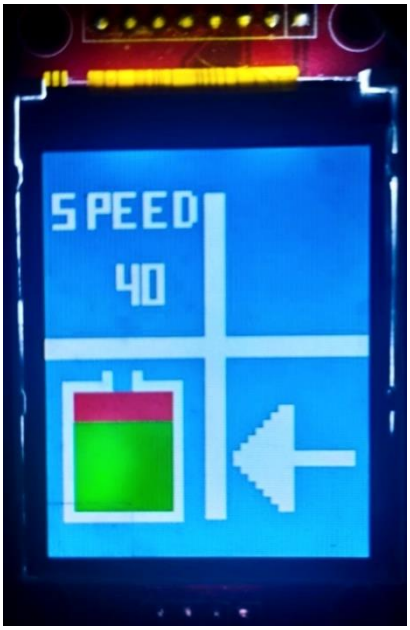


figure 6GUI vehicle move to left

Example Scenario

1. **Driving Near a Traffic Light:**

    - A car approaches a traffic light with a red signal.
    - The traffic light sends a red-light signal to the car's TFT display.
    - The driver stops the car.

2. **Accident Detection:**

    - A traffic light detects an accident through its sensors.
    - It sends an alert to emergency services and updates nearby vehicles to be cautious.
    - The information is also sent to Firebase, updating all connected apps and infrastructure.

3. **Charging Battery at Station:**

    - An EV enters a charging station.
    - The station detects the EV, retrieves its VIN, and communicates with its battery management system.
    - The TFT display shows the battery's current charge level and the cost to charge.
    - The driver confirms the charge, and the station begins charging the EV.
    - Payment is processed through the mobile app, linked to the car's VIN, with details sent to Firebase.

4. **Pedestrian Safety:**

    - A pedestrian opens the app and receives an alert about a fast-approaching vehicle.
    - The pedestrian waits until it is safe to cross the road.

Chapter Three

3 V2X MOBILE APPLICATION

3.1 INTRODUCTION

The Vehicle-to-Everything (V2X) App is an innovative mobile application developed using Android Studio and Java, designed to enhance safety and communication between vehicles and pedestrians. This application plays a crucial role in our graduation project, focusing on improving road safety through real-time location sharing and alert systems. The V2X App consists of two primary modes: **Street Mode (V2P - Vehicle to Pedestrian)** and **Car Mode**. the V2X App leverages modern technologies such as GPS, Firebase Realtime Database, and sensors to provide timely alerts and data sharing, thereby reducing the risk of accidents. Each mode serves distinct purposes and functionalities, facilitating seamless interaction between pedestrians and vehicles.

3.2 OBJECTIVES

- **Enhance Pedestrian Safety:** Provide real-time alerts to pedestrians and vehicles to prevent collisions.
- **Improve Traffic Management:** Facilitate better communication between road users.
- **Leverage Modern Technology:** Use GPS, Firebase, and sensors to create a reliable and efficient system.

3.3 SYSTEM ARCHITECTURE

3.3.1 Overview

The V2X App consists of two primary modes: Street Mode and Car Mode. Each mode serves distinct functionalities tailored for pedestrians and drivers, respectively. The system architecture is designed to ensure seamless communication between these two modes.

3.3.2 Components

- **Mobile Application:** Developed using Android Studio and Java, available for both Street Mode and Car Mode.
- **Firebase Realtime Database:** Used for storing and retrieving real-time data.
- **Sensors:** Utilized in Car Mode for obtaining compass data.
- **GPS:** Used in both modes for location tracking.
- **Hardware Component (ESP-32):** Used in Car Mode for additional functionalities.

3.3.3 Data Flow

- **Pedestrian shares location data via Street Mode.**
- **Vehicle subscribes to pedestrian location data and compares it with its own location.**
- **If a potential collision is detected, an alert is sent to the pedestrian.**
- **In Car Mode, the driver inputs the VIN number, and the vehicle's location, compass degree, monetary information, and timestamp are shared.**

3.4 STREET MODE (V2P - VEHICLE TO PEDESTRIAN)

3.4.1 Purpose

Street Mode is designed for pedestrians, enabling them to share their real-time location with nearby vehicles. This mode enhances pedestrian safety by providing timely alerts to both pedestrians and drivers in case of potential collisions.

3.4.2 Key Features

1. **Location Sharing:**
    - Uses GPS to obtain the pedestrian's current location.
    - Publishes location data to the Firebase Realtime Database.
2. **Location Subscription:**
    - Vehicles subscribe to the pedestrian's location data from Firebase.
    - The vehicle's application continuously compares its own location with the pedestrian's location.
3. **Proximity Alerts:**
    - If the vehicle detects that it is in close proximity to the pedestrian, it sends a warning alert to the pedestrian via Firebase.
    - The pedestrian's app reads this warning flag and displays a warning screen.

3.5 TECHNICAL IMPLEMENTATION

3.5.1 GPS Integration

The pedestrian's mobile device uses GPS to obtain the current location, which is then published to Firebase.

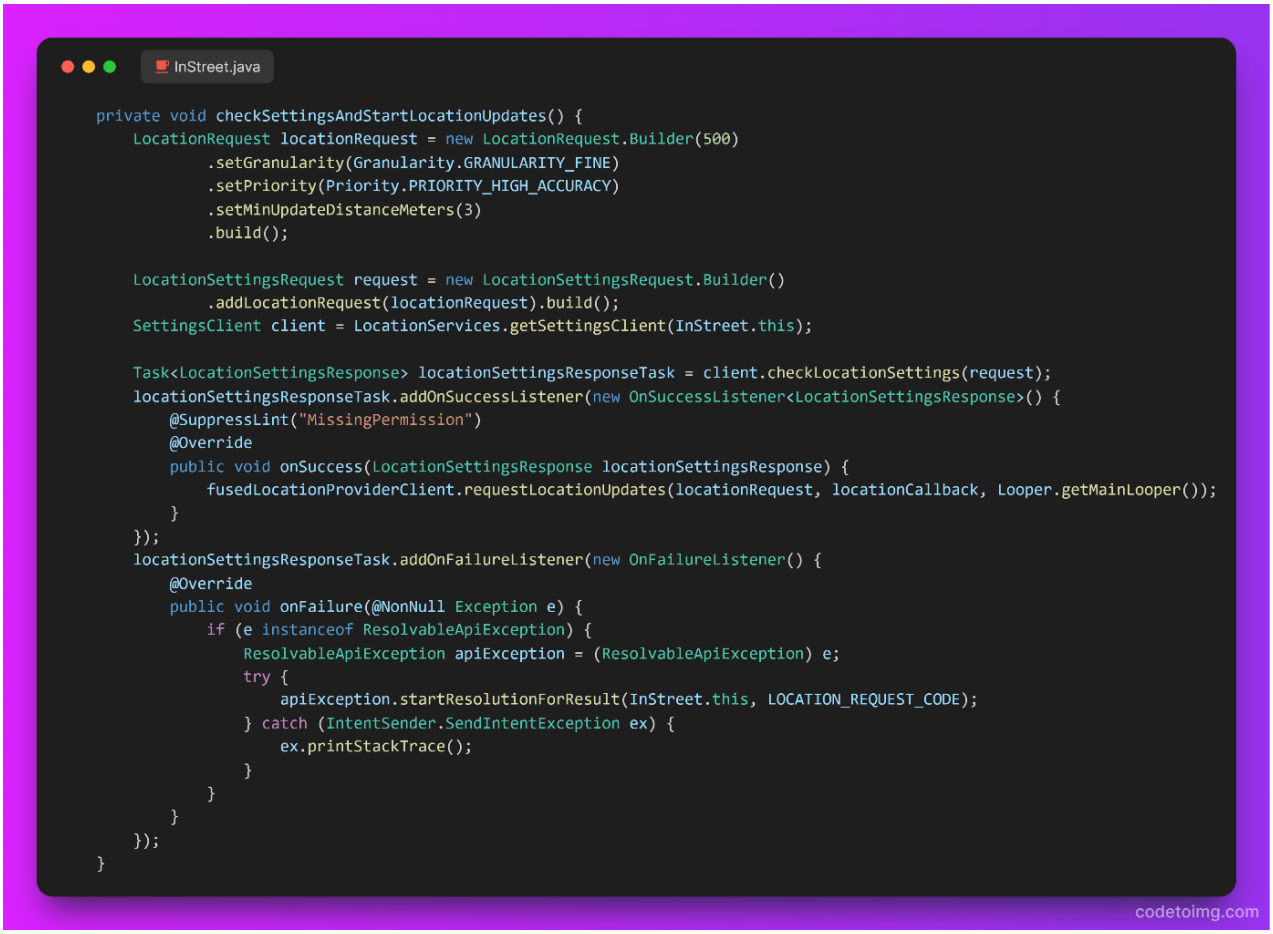


figure 7GPS integration

3.5.2 Location Formatting and publish Location to Firebase.

The vehicle's application subscribes to the pedestrian's location data and compares it with its own location to determine proximity. If a potential collision is detected, an alert is sent to the pedestrian.

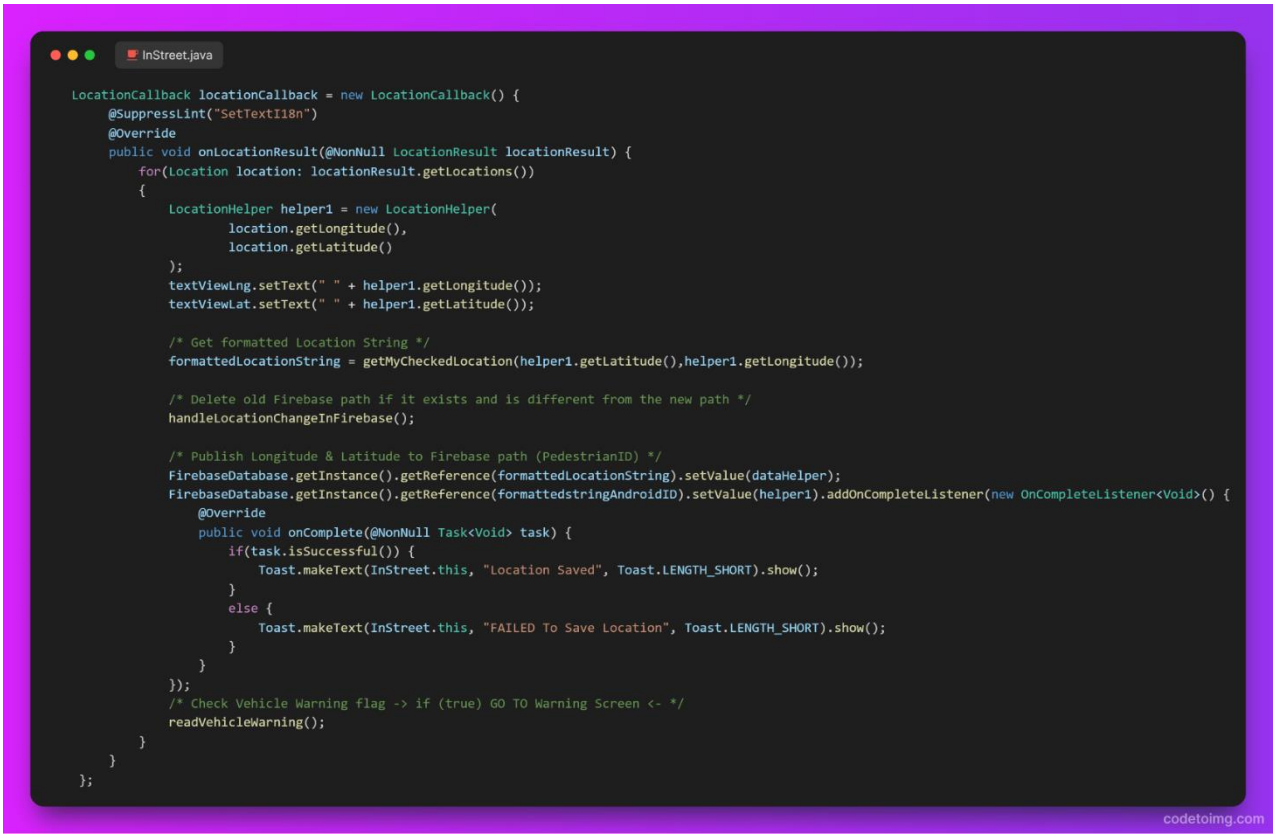


figure 8Location Formatting and publish Location to Firebase

3.5.3 Receiving Warnings

The pedestrian's application listens for warning flags in Firebase and displays a warning screen if an alert is received.

InStreet.java

```
public void readVehicleWarning() {
    /* Subscribe warning From Firebase path (formattedLocation/warning) */
    DatabaseReference databaseReference = FirebaseDatabase.getInstance().getReference(formattedLocationString);
    databaseReference.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot snapshot) {
            warningFlag = Boolean.TRUE.equals(snapshot.child("warning").getValue(Boolean.class));
            if(warningFlag){
                /* Go to warning screen */
                Intent intent = new Intent(InStreet.this, WarningScreen.class);
                startActivity(intent);
            }
        }
    });
}
```


figure 9: Receiving Warnings

3.6 CAR MODE

3.6.1 Purpose

Car Mode is designed for drivers, allowing them to share their vehicle's location, compass degree, monetary information, and timestamp with Firebase. This data is also used by the vehicle's onboard hardware component (ESP-32) for various functionalities.

3.6.2 Key Features

1. **VIN Number Input:**
    - At the start, the driver inputs their vehicle's VIN number.
    - This VIN number is used as a unique identifier to publish all related data to Firebase.
2. **Location Sharing:**
    - Uses GPS to obtain the vehicle's current location.
    - Publishes location data to Firebase under the VIN number.
3. **Compass Degree Sharing:**
    - Retrieves the compass degree using the Android Sensor API.
    - Publishes the compass degree to Firebase under the VIN number.
4. **Monetary Information:**
    - Captures or calculates monetary values related to V2G.
    - Publishes monetary data to Firebase under the VIN number.
5. **Timestamp Sharing:**
    - Retrieves the current date and time using Java's date and time API.
    - Publishes the timestamp to Firebase under the VIN number.

3.6.3 Technical Implementation

- **In Car Activity and publish all related data to Firebase**

InCar.java

```
LocationCallback locationCallback = new LocationCallback() {
    @SuppressLint("SetTextI18n")
    @Override
    public void onLocationResult(@NonNull LocationResult locationResult) {

        for (Location location : locationResult.getLocations()) {

            LocationHelper helper = new LocationHelper(
                    location.getLongitude(),
                    location.getLatitude(),
                    Money,
                    compassDegree,
                    DateTimeFormat
            );

            textViewLng.setText(" " + helper.getLongitude());
            textViewLat.setText(" " + helper.getLatitude());
            textViewMoney.setText(" " + Money);

            /* Publish Longitude & Latitude to Firebase path (CarVIN) */
            FirebaseDatabase.getInstance().getReference(Car_VIN)
                    .setValue(helper).addOnCompleteListener(new OnCompleteListener<Void>() {
                        @Override
                        public void onComplete(@NonNull Task<Void> task) {
                            if (task.isSuccessful()) {
                                Toast.makeText(InCar.this, "Location Saved", Toast.LENGTH_SHORT).show();
                            } else {
                                Toast.makeText(InCar.this, "FAILED To Save Location", Toast.LENGTH_SHORT).show();
                            }
                        }
                    });

        }
    }
};
```

codetoimg.com

figure 10publish all related data to Firebase

- **Showing VIN**

The VIN (Vehicle Identification Number) is stored using SharedPreferences. This allows the application to save the VIN so that it can be retrieved and displayed even after the app is closed and reopened.

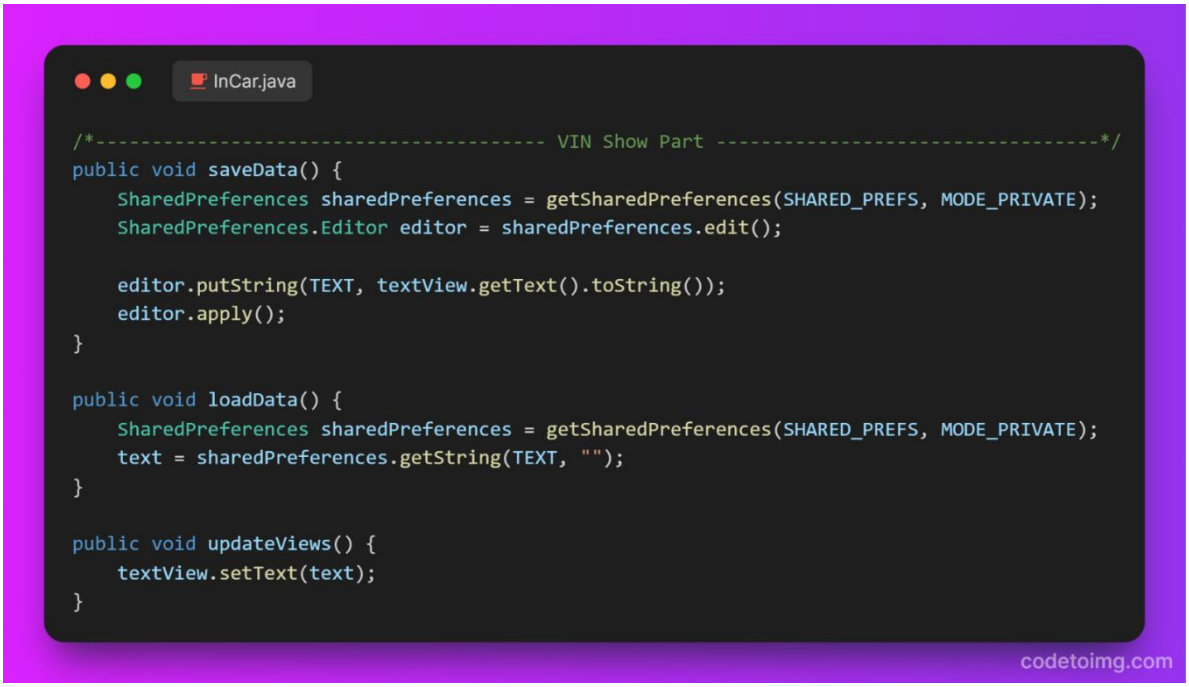


figure 11Showing VIN In App

- **Detailed Explanation:**

  saveData():

  - This method saves the VIN entered by the user into SharedPreferences.
  - SharedPreferences sharedPreferences = getSharedPreferences(SHARED_PREFS, MODE_PRIVATE); gets a reference to the shared preferences file.
  - SharedPreferences.Editor editor = sharedPreferences.edit(); creates an editor for making changes.
  - editor.putString(TEXT, textView.getText().toString()); saves the text from the textView (which contains the VIN) into the shared preferences.
  - editor.apply(); commits the changes asynchronously.

loadData(): This method loads the saved VIN from SharedPreferences.

- SharedPreferences sharedPreferences = getSharedPreferences(SHARED_PREFS, MODE_PRIVATE); gets a reference to the shared preferences file.

- text = sharedPreferences.getString(TEXT, ""); retrieves the saved VIN (stored under the key TEXT) and assigns it to the text variable.

updateViews(): This method updates the textView with the loaded VIN.

- textView.setText(text); sets the text of the textView to the value stored in the text variable.

3.6.4 Compass Padding Sensor

This part of the code handles the compass sensor data to show the vehicle's heading direction in degrees.

InCar.java

```
/*----------------------------- compass padding -----------------------------*/
@Override
protected void onResume() {
    super.onResume();
    sensorManager.registerListener(this, sensorManager
                                    .getDefaultSensor(Sensor.TYPE_ORIENTATION),
                                     SensorManager.SENSOR_DELAY_UI);
}

@Override
protected void onPause() {
    super.onPause();
    sensorManager.unregisterListener(this);
}

@SuppressLint("SetTextI18n")
@Override
public void onSensorChanged(SensorEvent event) {
    /* For Compass Degree Sensor */
    compassDegree = Math.round(event.values[0]);
    textViewDg.setText(compassDegree + "°");
}
```

codetoimg.com

figure 12Compass Padding Sensor

- **Detailed Explanation**

onResume(): This method is called when the activity enters the Resumed state.

- sensorManager.registerListener(this, sensorManager.getDefaultSensor(Sensor.TYPE_ORIENTATION), SensorManager.SENSOR_DELAY_UI); registers a listener for orientation sensor updates. The listener (this) will receive updates at a UI-friendly rate.

onPause(): This method is called when the activity enters the Paused state.

- sensorManager.unregisterListener(this); unregisters the listener to stop receiving sensor updates, which conserves battery.

onSensorChanged(SensorEvent event): This method is called whenever the sensor's data changes.

- compassDegree = Math.round(event.values[0]); gets the compass degree from the sensor event and rounds it.
- textViewDg.setText(compassDegree + "°"); updates the textViewDg to display the compass degree followed by a degree symbol.

3.6.5 Showing Date & Time

This part of the code continuously updates and displays the current date and time.

InCar.java

```
/*-------------------------------------- Date & Time ----------------------------------*/
public void getDateAndTime() {
    Thread thread = new Thread() {
        @Override
        public void run() {
            try {
                while (!isInterrupted()) {
                    Thread.sleep(1000);
                    runOnUiThread(new Runnable() {
                        @SuppressLint("SetTextI18n")
                        @Override
                        public void run() {
                            Date calendar = Calendar.getInstance().getTime();
                            String dateFormat = DateFormat.getDateInstance().format(calendar);
                            String timeFormat = DateFormat.getTimeInstance().format(calendar);
                            DateTimeFormat = String.format("%s - %s", dateFormat, timeFormat);
                            textViewDate.setText(DateTimeFormat);
                        }
                    });
                }
            } catch (Exception e) {
                textViewDate.setText(R.string.app_name);
            }
        }
    };
    thread.start();
}
```

codetoimg.com

figure 13Showing Date & Time

- **Detailed Explanation**

  getDateAndTime(): This method creates and starts a new thread that updates the date and time every second.

  - Thread thread = new Thread() { ... }; creates a new thread.
  - while (!isInterrupted()) { ... } keeps the thread running until it is interrupted.
  - Thread.sleep(1000); pauses the thread for 1 second between updates.
  - runOnUiThread(new Runnable() { ... }); ensures that UI updates (setting the text in textViewDate) are performed on the main thread.

3.7 FIREBASE INTEGRATION

3.7.1 Configuration

Setting Up Firebase To integrate Firebase with the V2X App, the following steps were undertaken:

1. **Firebase Project Creation:** A new project was created in the Firebase console.
2. **Database Configuration:** The Firebase Realtime Database was configured to store location data, compass degrees, monetary values, and timestamps.
3. **Security Rules:** Security rules were implemented to ensure data security and integrity, allowing only authorized users to read and write data.

3.7.2 Data Structure

The data in Firebase is organized as follows:

3.7.3 Street Firebase Database

figure 14Street Firebase Database

3.7.4 Car Firebase Database

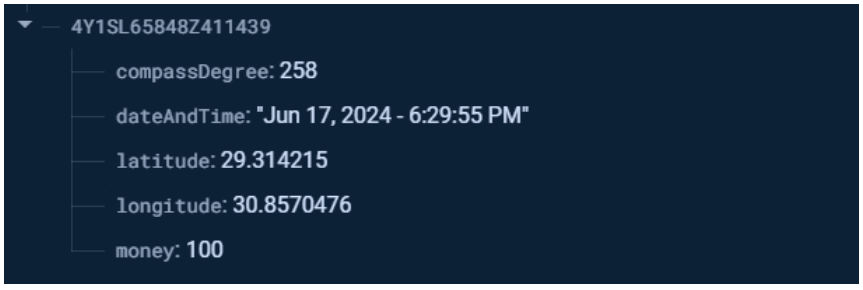


figure 15Car Firebase Database

3.8 USER INTERFACE DESIGN

3.8.1 Get Started Screen and Info Screen

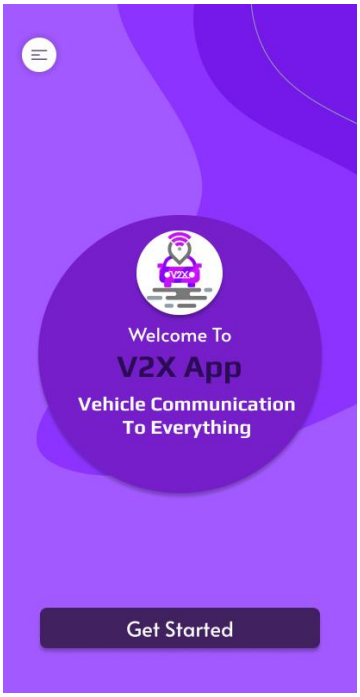


figure 16Info Screen in mobile app

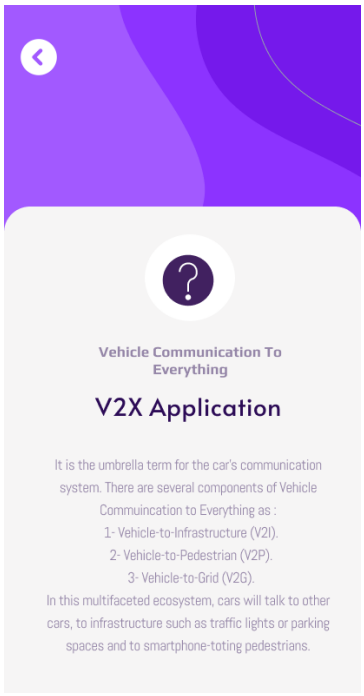


figure 17Info Screen in mobile app

3.8.2 Choice screen

The user interface for Choice screen is designed to user choose he in street or in car.

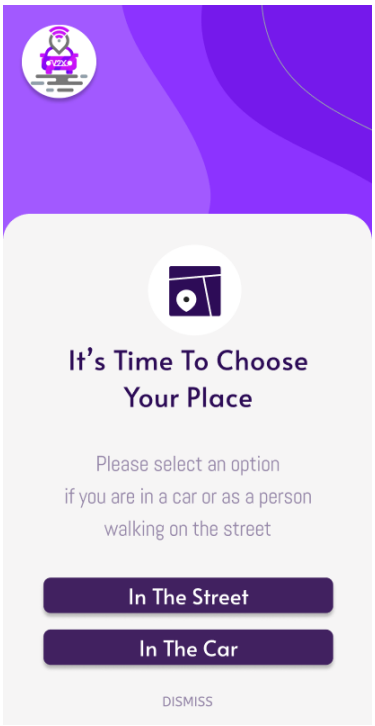


figure 18Choice screen

3.8.3 Street Mode

The user interface for Street Mode is designed to be simple and intuitive, allowing pedestrians to easily share their location and receive warning.

- **Main Screen:**
  - A map view showing the pedestrian's current location.
  - A buttons to start and stop sharing location data.

- **Warning Screen:**
  - A prominent warning message displayed when a potential collision is detected.

- And phone will vibrate strongly and a warning sound will play.

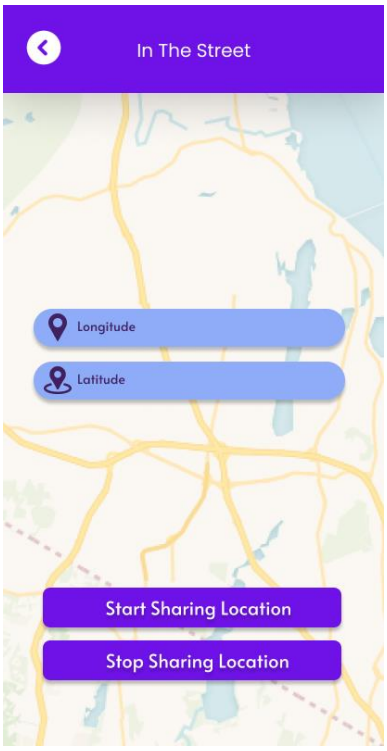


figure 20Main Screen

figure 19Warning Screen

3.8.4 Car Mode

The user interface for Car Mode allows drivers to input their VIN number and start sharing vehicle data.

- **Main Screen:**

  - An input field for entering the VIN number.
  - A text view to submit the VIN number.
  - A buttons to start and stop data sharing.

figure 21Main Screen mobile app

3.9 CAR CONTROLLER APP

3.9.1 Overview

The Bluetooth RC Car Controller app allows users to control an RC car using their mobile device. By leveraging Bluetooth technology and a microcontroller equipped with Bluetooth, the app provides a seamless and interactive way to operate RC cars. Users can navigate the car using virtual buttons and adjust its speed with a slider bar, enhancing the overall control experience.

3.9.2 Features

1. **Bluetooth Connectivity:**
    - The app pairs with a Bluetooth-equipped microcontroller installed in the RC car, enabling wireless control.

2. **Virtual Buttons:**
    - Users can control the car's movement (forward, backward, left, right) using intuitive virtual buttons on the app's interface.
3. **Speed Adjustment:**
    - A slider bar allows users to adjust the car’s speed, provided the car's control circuit supports variable speed functionality.
4. **User-Friendly Interface:**
    - The app is designed with a clean and intuitive user interface, making it easy for users of all ages to control their RC cars.

3.9.3 Technical Details

- **Bluetooth Integration**

The app uses the Bluetooth API to connect to the RC car’s microcontroller. Upon launching, the app scans for available Bluetooth devices, allowing the user to select and pair with their RC car.

MainActivity.java

```
BluetoothAdapter bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
if (!bluetoothAdapter.isEnabled()) {
    Intent enableBtIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
    startActivityForResult(enableBtIntent, REQUEST_ENABLE_BT);
}

// Scan for devices
Set<BluetoothDevice> pairedDevices = bluetoothAdapter.getBondedDevices();
if (pairedDevices.size() > 0) {
    for (BluetoothDevice device : pairedDevices) {
        // Add the name and address to an array adapter to show in a ListView
        mArrayAdapter.add(device.getName() + "\n" + device.getAddress());
    }
}
```

codetoimg.com

figure 22Bluetooth Integration

- **Virtual Button Controls**

The app's interface includes buttons for directional control. These buttons send corresponding commands to the RC car via Bluetooth.

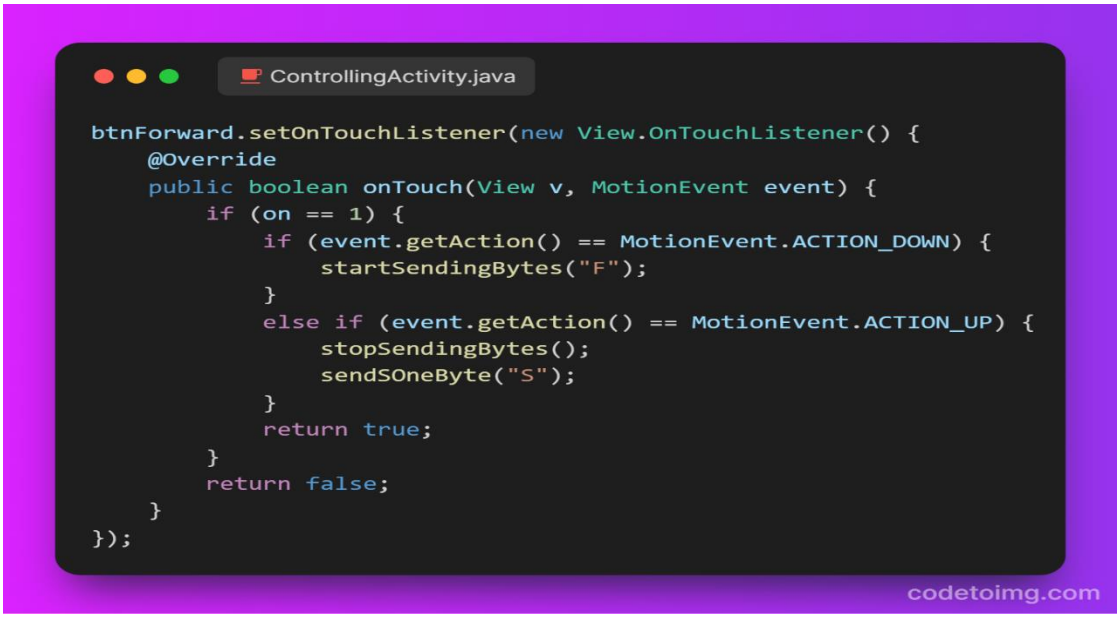


Figure 21: Virtual Button Controls

Forward button sends (“F”).

Backward button sends (“B”).

Right button sends (“R”).

Left button sends (“L”).

- **Speed Control**

A slider bar allows for dynamic speed adjustments, enhancing the driving experience. The app sends speed values to the RC car, adjusting its speed in real-time.

- For Speed 0 sends (“0”).
- For Speed 10 sends (“1”).
- For Speed 20 sends (“2”).
- For Speed 30 sends (“3”).

- For Speed 40 sends (“4”).
- For Speed 50 sends (“5”).
- For Speed 60 sends (“6”).
- For Speed 70 sends (“7”).
- For Speed 80 sends (“8”).
- For Speed 90 sends (“9”).
- For Speed 100 sends (“q”).

3.9.4 User Interface Design

- **Bluetooth Search and Connect Screen**

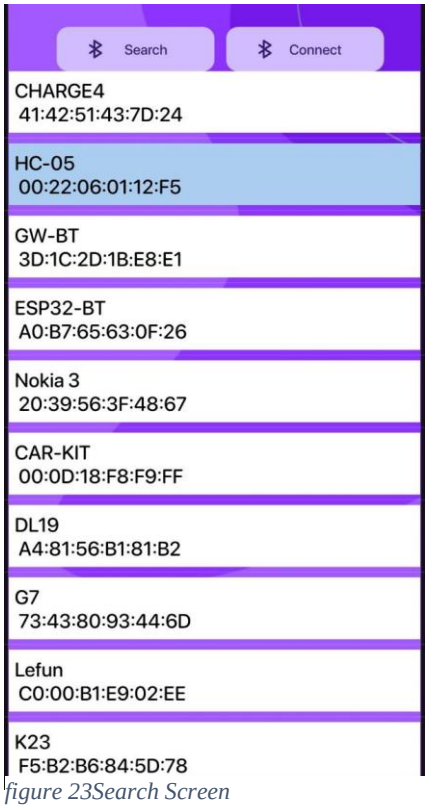


figure 23Search Screen

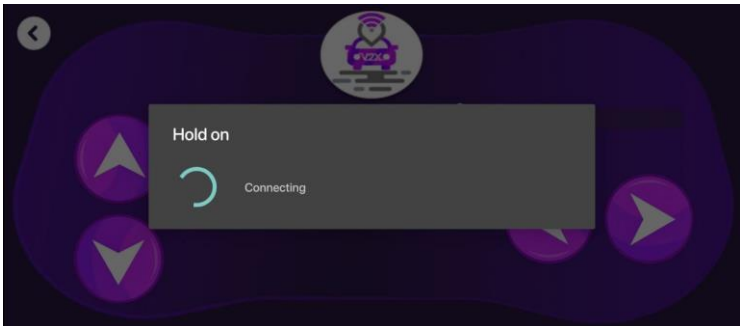


figure 24Connect screen

- **Controlling Screen**

figure 25Controlling Screen

3.10 CHALLENGES AND SOLUTIONS

Challenge: Real-time Data Accuracy

Solution:

- Utilized the FusedLocationProviderClient for accurate and efficient location tracking.
- Implemented frequent data updates and checks to ensure real-time accuracy.

Challenge: Ensuring Data Security

Solution:

- Configured Firebase security rules to restrict unauthorized access.
- Used Firebase Authentication to ensure that only authenticated users can read and write data.

Challenge: Handling Sensor Data

Solution:

- Used the Android Sensor API to obtain accurate compass readings.
- Implemented sensor listeners to continuously update the compass degree in real-time.

3.11 CONCLUSION

The V2X App represents a significant advancement in vehicle-to-pedestrian communication, leveraging modern technology to enhance road safety. By facilitating real-time location sharing and alerts, the application aims to reduce accidents and improve traffic management. This project demonstrates the potential of innovative solutions in addressing real-world problems and highlights the importance of integrating technology with everyday activities.

3.12 FUTURE WORK

- **Integration with More Vehicles:** Expand the application to support more vehicle types and models.
- **Enhanced Alerts:** Develop more sophisticated alert systems using machine learning to predict potential collisions.
- **User Interface Improvements:** Continuously improve the user interface based on user feedback.
- **Data Analytics:** Implement data analytics to provide insights on traffic patterns and pedestrian behavior.

Chapter Four

4 ALGORITHMS COMMUNICATION

V2X communication is a technology that allows vehicles to communicate with different parts of the transportation system.

V2X includes several types of communication: Vehicle-to-Infrastructure (V2I), Vehicle-to-Pedestrian (V2P), and Vehicle-to-Grid (V2G).

4.1 V2I COMMUNICATION

4.1.1 Overview

V2I Communication refers to the real-time exchange of information between vehicles and infrastructure elements such as traffic lights, road signs, and intelligent transportation systems. This bidirectional communication enables vehicles to receive critical data from infrastructure components and, in turn, share relevant information, contributing to a more connected and responsive transportation ecosystem.

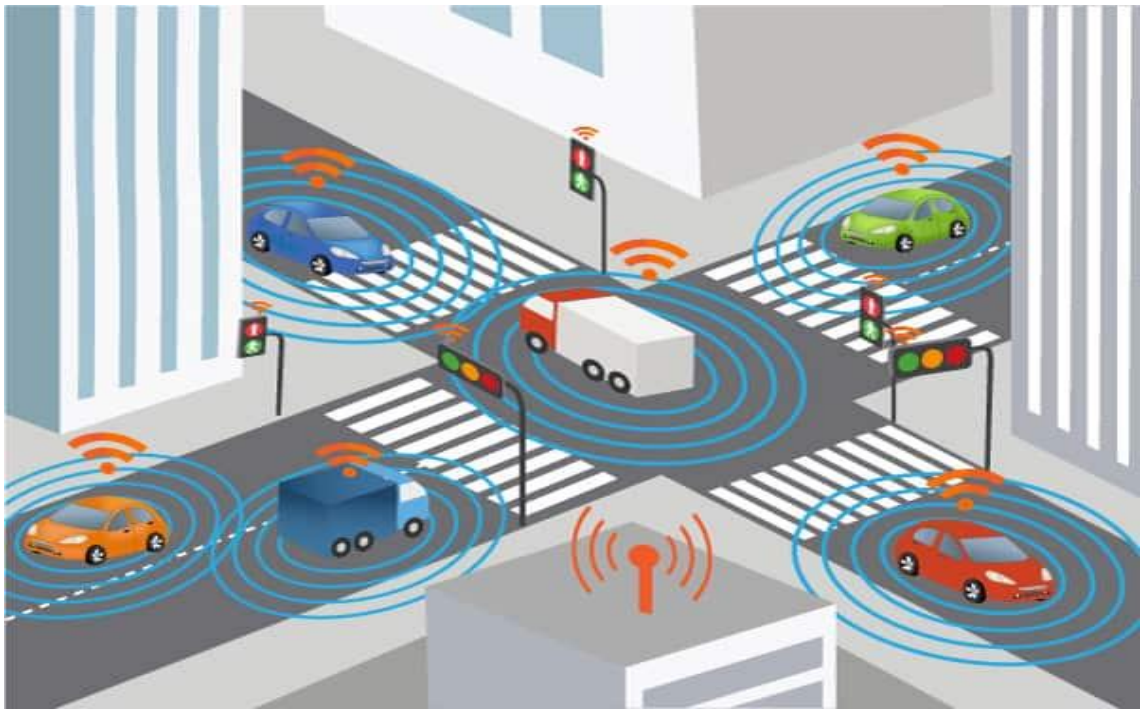


figure 26V2I COMMUNICATION

4.1.2 V2I COMMUNICATION

Data Exchange: V2I Communication relies on a network of sensors and communication devices embedded in both vehicles and infrastructure. These devices facilitate the exchange of data, ranging from traffic conditions and road hazards to signal timing and navigation information.

Real-Time Decision-Making: The exchanged data enables vehicles to make informed, real-time decisions based on the current state of the road and surrounding conditions. For example, a vehicle approaching an intersection can receive information about the optimal speed to catch a green light, enhancing traffic flow and reducing congestion.

Safety Applications: V2I Communication plays a crucial role in safety applications such as collision avoidance. Vehicles can receive alerts about potential hazards or obstacles on the road, allowing for immediate response and mitigating the risk of accidents.

- **EXAMPLES OF V2I COMMUNICATION USE**
  - **Traffic Signal Optimization**: In smart cities like Singapore and Barcelona, V2I Communication is employed to optimize traffic signal timings based on real-time traffic flow. This not only reduces congestion but also improves fuel efficiency by minimizing unnecessary stops and starts.
  - **Emergency Vehicle Preemption**: V2I Communication allows emergency vehicles to communicate with traffic signals, preempting the signal to turn green in their favor. This ensures swift passage for emergency responders, optimizing response times during critical situations.
  - **Smart Parking Solutions**: V2I Communication aids in providing real-time information about available parking spaces. Drivers can access this information through in-vehicle displays or mobile apps, reducing the time spent searching for parking and minimizing traffic congestion.

4.1.3 BENEFITS OF V2I COMMUNICATION

For urban planners, transportation authorities, and professionals in the automotive and infrastructure sectors, V2I Communication offers a range of benefits:

- **Improved Traffic Management**: V2I Communication enables more effective traffic management, reducing congestion and enhancing the overall efficiency of transportation networks.
- **Enhanced Safety**: Real-time information exchange contributes to improved safety by providing drivers with timely alerts about potential hazards and critical road conditions.
- **Efficient Resource Allocation**: V2I Communication allows for the optimization of infrastructure resources, ensuring that traffic signals, road signs, and other elements operate in sync to meet the demands of varying traffic conditions.

As the world moves towards a future of connected mobility, V2I Communication stands as a linchpin in creating intelligent transportation ecosystems. Its application not only transforms the driving experience but also lays the foundation for more sustainable, efficient, and safe transportation networks.

4.1.4 How V2I works

- At first, v2i communication is working using firebase to load data of location and all other needed data to it.
  For car, we replaced GPS module by mobile application –developed by us- that can upload location, time and date to firebase.

figure 27V2I firebase

Also, stm32f103 of vehicle is connected to esp32 by UART1 communication protocol and esp32 is connected to firebase.

There is static topic in firebase named with VIN of vehicle including location, each two seconds vehicle updates its location in firebase, also traffic light, speed limit sign and stop sign in road created a static topic with its special ID including its location.

then vehicle checks if there is another topic equal to its or not, if there is one equal, this means that two topics are in the same location.

Next step is that vehicle check ID in the another topic to detect if it is a traffic light or stop sign or speed limit sign or pedestrian.

We will postpone talk about pedestrian to next chapter, So now we have three cases of id.

- If id was “R1-1”, this mean that is a stop sign, then esp32 will transmit specific id number to stm32 which will show stop sign in TFT in vehicle.

Next figure show stop sign communication between firebase and esp32.

```
XP7SA1DG9SFC14705/latitude <<---29.301060
XP7SA1DG9SFC14705/longitude <<---30.708948
XP7SA1DG9SFC14705/compassDegree <<---290
29,3010-30,7089 /ID <<---R1-1-001
Found
R1-1-001/latitude <<---29.301073
R1-1-001/longitude <<---30.711433
271 - 290 = -19
XP7SA1DG9SFC14705/dateAndTime <<---Jun 14, 2024 - 8:42:27 AM
R1-1-001/Jun 14, 2024 - 8:42:27 AM--->> XP7SA1DG9SFC14705
STOP
```

figure 29stop sign communication between firebase and esp32

figure 28: stop sign in GUI in vehicle

- If id was “W13-1”, this mean that is a speed limit 35 km sign, then esp32 will transmit specific id number to stm32 which will show speed limit 35 km in TFT in vehicle. if id was “W13-2”, this mean that is a speed limit 25 km sign, then esp32 will transmit specific id number to Stm32 which will show a speed limit 25 km sign in TFT in vehicle.

```
XP7SA1DG9SFC14705/latitude <<---29.301060
XP7SA1DG9SFC14705/longitude <<---30.708948
XP7SA1DG9SFC14705/compassDegree <<---290
29,3010-30,7089 /ID <<---W13-2-001
Found
W13-2-001/latitude <<---29.301045
W13-2-001/longitude <<---30.711453
270 - 290 = -20
XP7SA1DG9SFC14705/dateAndTime <<---Jun 14, 2024 - 8:42:27 AM
W13-2-001/Jun 14, 2024 - 8:42:27 AM--->> XP7SA1DG9SFC14705
Limit Speed : 25
```

figure 30limit speed sign communicates with vehicle

Figure 29: limit speed 25 GUI

- If Id was “W3-3”, this mean that is traffic light -Red- sign.

```
XP7SA1DG9SFC14705/latitude <<---29.301060
XP7SA1DG9SFC14705/longitude <<---30.708948
XP7SA1DG9SFC14705/compassDegree <<---290
29,3010-30,7089 /ID <<---W3-3-001
Found
W3-3-001/latitude <<---29.301073
W3-3-001/longitude <<---30.711433
271 - 290 = -19
XP7SA1DG9SFC14705/dateAndTime <<---Jun 14, 2024 - 8:42:27 AM
W3-3-001/Jun 14, 2024 - 8:42:27 AM--->> XP7SA1DG9SFC14705
W3-3-001/Status <<---180
Traffic Light : 180
```

figure 32: traffic light red communication on serial of Arduino IDE

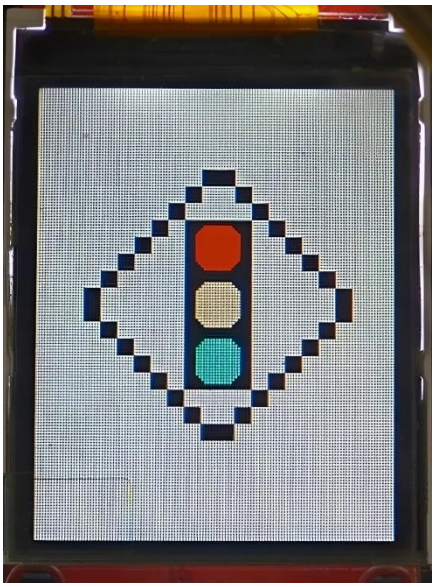


figure 33: traffic light red in GUI

Upper figure show traffic light red communication on serial of Arduino IDE.

Esp32 will transmit specific id number to stm32 which will show a is traffic light -Red- sign in TFT in vehicle.

there is another feature in v2i is that special vehicle like ambulance can control traffic light by pushing button in vehicle connected to stm32 which sends id to esp32 using UART1 and esp32 upload warning id to firebase, and next figure show traffic light red for special vehicle.

```
XP7SA1DG9SFC14705/latitude <<---29.301060
XP7SA1DG9SFC14705/longitude <<---30.708948
XP7SA1DG9SFC14705/compassDegree <<---290
29,3010-30,7089 /ID <<---W3-3-001
Found
W3-3-001/latitude <<---29.301073
W3-3-001/longitude <<---30.711433
271 - 290 = -19
XP7SA1DG9SFC14705/dateAndTime <<---Jun 14, 2024 - 8:42:27 AM
W3-3-001/Jun 14, 2024 - 8:42:27 AM--->> XP7SA1DG9SFC14705
W3-3-001/Status <<---180
Traffic Light : 180
W3-3-001/Special Vehicle <<---0
W3-3-001/Special Vehicle--->> 196
```

figure 34special vehicle control traffic light

traffic light periodically reads this flag if it is equal to zero ,none happens but if it changed to another value, traffic light will change from red to green.

figure 35V2I before pushing button of special vehicle

Next figure show V2I after pushing button of special vehicle

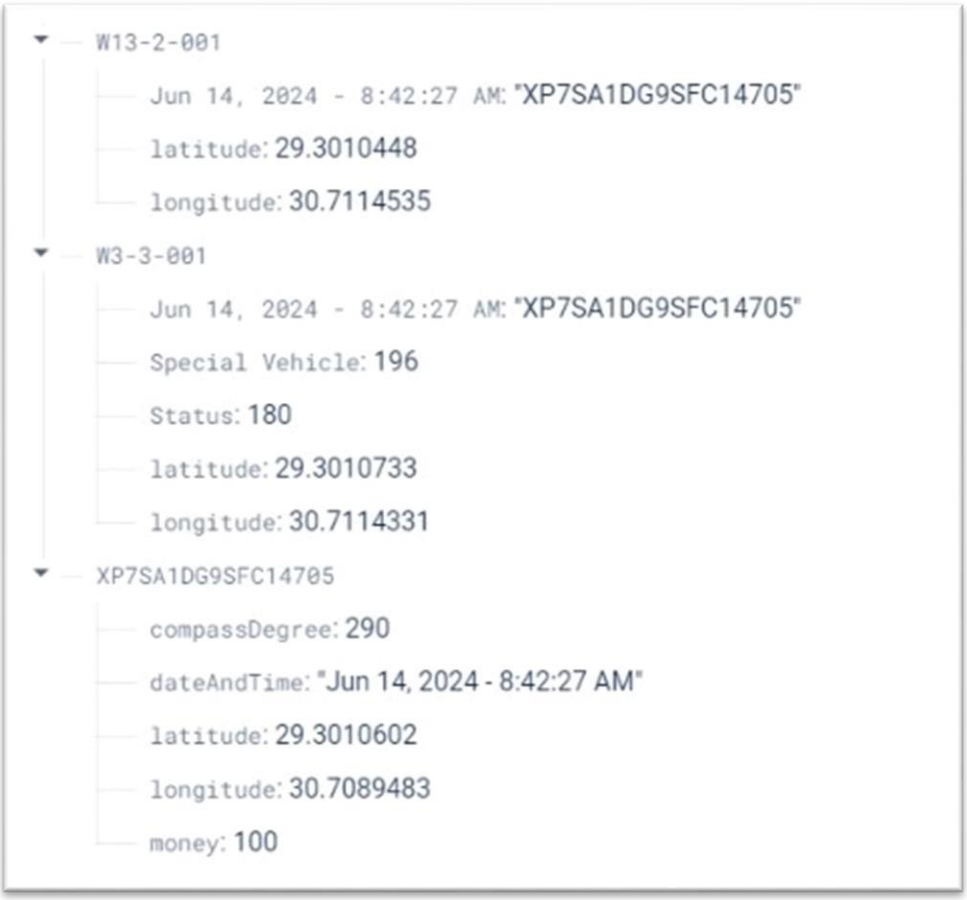


figure 36V2I after pushing button of special vehicle

4.1.5 Accident Detection:

AI Accident Detection System integrated with V2I communication and Cloud connectivity. This system aims to revolutionize road safety by leveraging traffic light sensors to detect accidents and transmit real-time GPS location information to the Cloud. Emergency response teams can access this data promptly for rapid intervention.

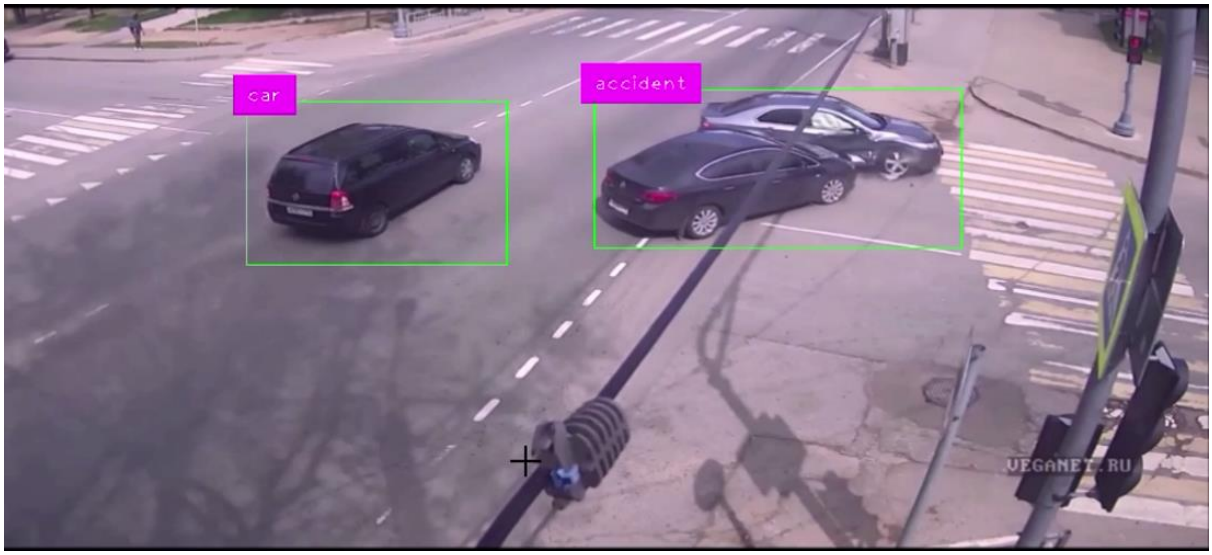


figure 37Accident Detection

- **Problem Statement:**

The timely detection and response to road accidents are critical for minimizing casualties and property damage. However, traditional accident reporting mechanisms often suffer from delays and inaccuracies. By integrating traffic light sensors with V2I communication and Cloud connectivity, we address these challenges and enable swift emergency responses.

- **Objectives:**

  - Develop an AI-based accident detection system using traffic light sensors.

- Integrate V2I communication capabilities to transmit accident data to the Cloud.
- Enable real-time GPS location tracking of accidents for emergency responders.
- Ensure high accuracy and reliability in accident detection and data transmission.
- Facilitate seamless connectivity between traffic infrastructure and emergency services.

- **Dataset Description:**

The dataset description details the data used for training and testing the AI models, including annotated images and videos of accident scenarios captured by traffic light cameras. Additionally, simulated V2I messages containing GPS location information are generated to evaluate the system's performance under various conditions.

- **Implementation:**

1 - Model Architecture (YOLOv8)

YOLOv8 is the latest iteration of the YOLO family of object detection models, known for its high performance in terms of speed and accuracy. YOLOv8 is designed to be more efficient and effective, making it a suitable choice for real-time applications such as accident detection in various environments like traffic monitoring, industrial safety, and more.

- **Why use Yolo in Accident Detect**

1. **Real-Time Processing**

   Accident detection often requires real-time analysis of video streams to promptly identify incidents. YOLO's architecture enables it to process images at high frame rates, making it suitable for real-time applications.

2. **High Accuracy**

   YOLO models provide high accuracy in detecting objects. This is crucial for accident detection, where false positives (incorrectly detecting accidents) or false negatives (missing actual accidents) can have serious consequences.

3. **Single-Pass Detection**

   YOLO's single-pass detection means that it processes the entire image in one go, as opposed to traditional methods that require multiple passes. This efficiency reduces the computational load and speeds up the detection process.
   Versatility

   YOLO can be trained to detect a wide range of objects, including different types of accidents. Whether it's a car crash, a person falling, or an industrial mishap, YOLO can be adapted to recognize these events.

4. **Scalability**

   YOLO's architecture can be scaled up or down based on the computational resources available. This flexibility allows it to be deployed on various devices, from powerful servers to edge devices like drones and CCTV cameras.

2 - integration firebase

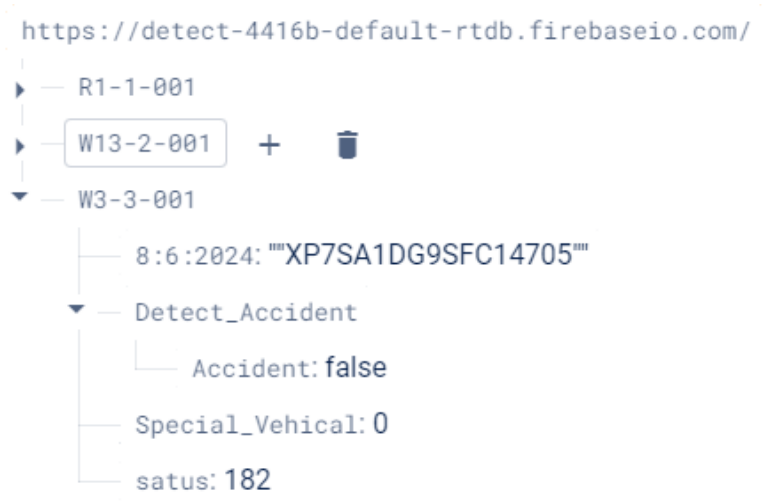
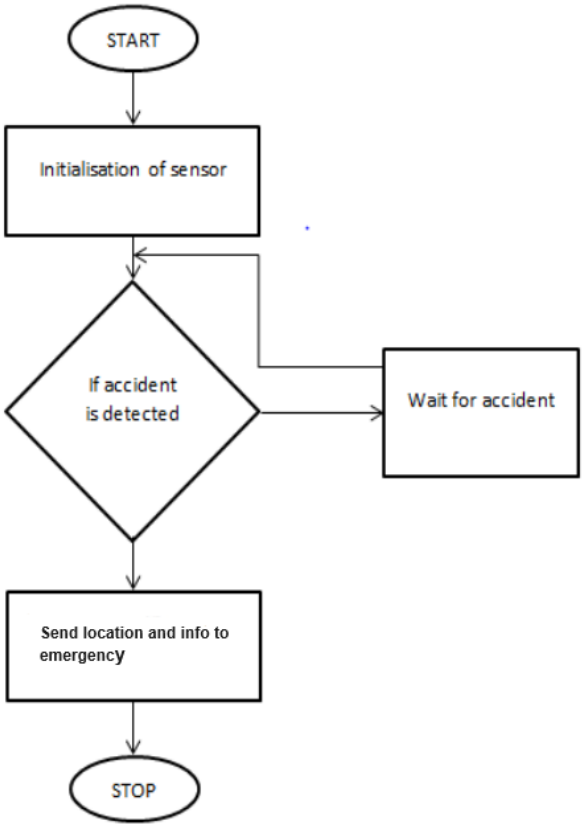
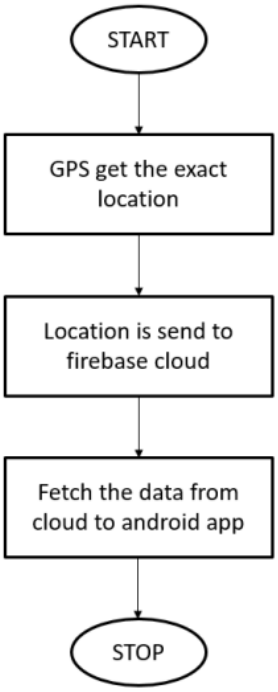


figure 38: integration firebase

- **Flow chart**


START
Initialisation of sensor
If accident is detected
Wait for accident
Send location and info to emergency
STOP


START
GPS get the exact location
Location is send to firebase cloud
Fetch the data from cloud to android app
STOP

4.2 V2P: VEHICLE-TO-PEDESTRIAN TECHNOLOGY

4.2.1 Overview

The emergence of smart technologies in transportation is important how we approach road safety and urban mobility. Among these innovations, Vehicle-to-Pedestrian (V2P) communication stands out as a critical development aimed at protecting the most vulnerable road users: pedestrians. V2P communication is part of the broader Vehicle-to-Everything (V2X) ecosystem. V2P focuses on the interaction between vehicles and pedestrians to prevent accidents and ensure safety for all road users. This chapter delves into the concept of V2P communication, its significance, underlying technologies, and practical applications, including the mobile application developed as part of this project.

figure 39: Vehicle-to-Pedestrian

4.2.2 BENEFITS OF V2I COMMUNICATION

- Preventing Accidents: Real-time alerts can significantly reduce the chances of collisions between vehicles and pedestrians.
- Enhancing Situational Awareness: Both drivers and pedestrians receive timely information, improving their ability to make right decisions for their safety.
- Facilitating Quick Reactions: Immediate warnings allow for rapid response, which is important in preventing accidents in fast-moving urban settings

4.2.3 How V2P works

At first, V2P communication is working as same as V2I communication However, in V2P, the mobile application is used entirely and relies on it completely.

The driver uses the mobile application in in-car mode, while pedestrians use the mobile application in in-street mode."

Firstly, for the car, the driver uses the mobile application to upload information about the vehicle such as the Vehicle Identification Number (VIN), the location of the car, direction, time, and date to cloud database – Firebase.

Next figures show how the driver uses the mobile application to upload vehicle data.

1- The first step is to open the application and press the **Get Stated** button.
2- The second step is to press the **In Car** button.
3- The third step is to **Enter** Vehicle Identification Number (**VIN**).
4- The last step is to press the **In Car** button to begin sending vehicle data to Firebase.

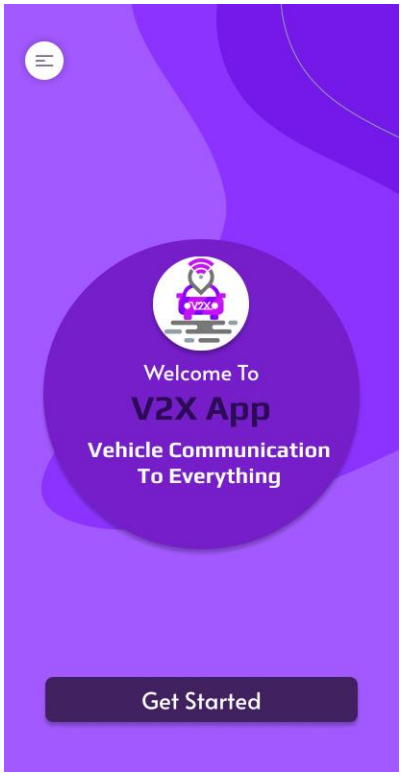


figure 40: press the Get Stated button

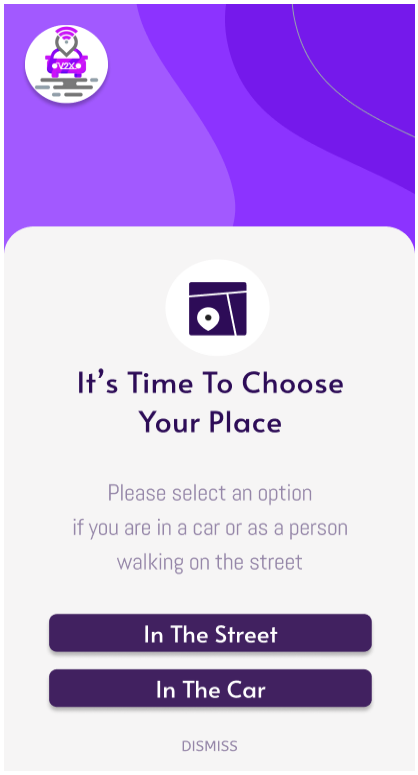


figure 41: press the In Car button

figure 42start button

After that, the data will be uploaded to Firebase.

Next figure shows the format of the vehicle data in Firebase:

figure 43format of the vehicle data in Firebase

Also, STM32f103 of vehicle is connected to ESP-32 by UART1 communication protocol and ESP32 is connected to firebase.

Figure 39: press the Get Stated button *Figure 40: press the In Car button* *Figure 41: enter VIN and press the In Car button*

There is static topic in firebase named with VIN of vehicle including location, every two seconds vehicle updates its location in firebase, and Pedestrians on the road also create a topics with their location.

figure 44vehicle and pedestrian updates its location in firebase

Next figures show how Pedestrians uses the mobile application to upload their location.

1- The first step is to open the application and press the **Get Stated** button.
2- The second step is to press the **In Street** button.
3- The last step is to press the **Start Sharing Location** button to begin sending their location and formatted string of their location.

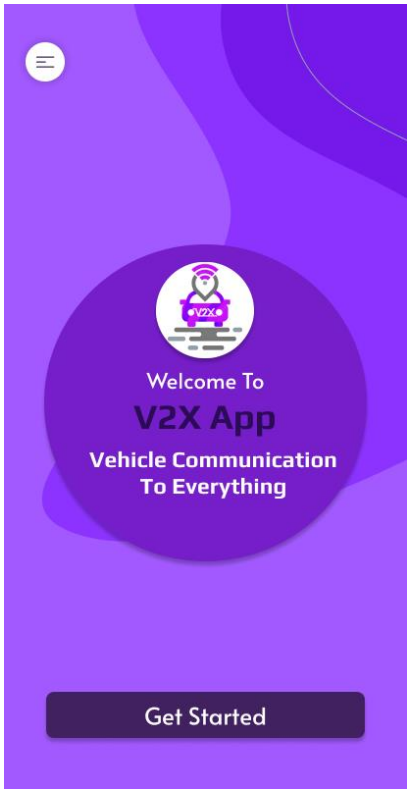


figure 45how Pedestrians uses the mobile application to upload their location

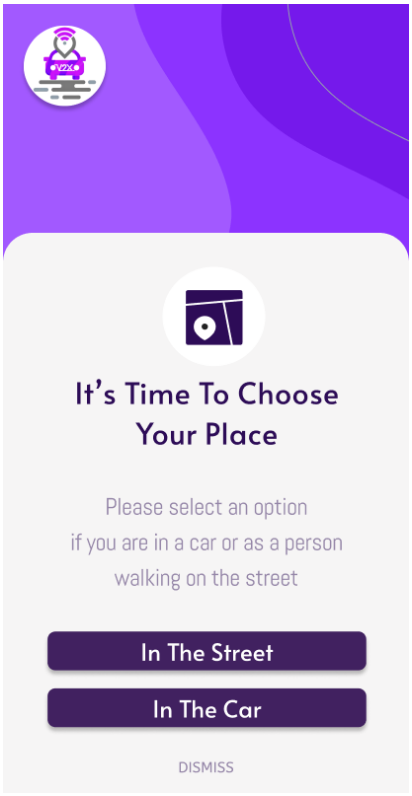


Figure 462nd :press the In Street button

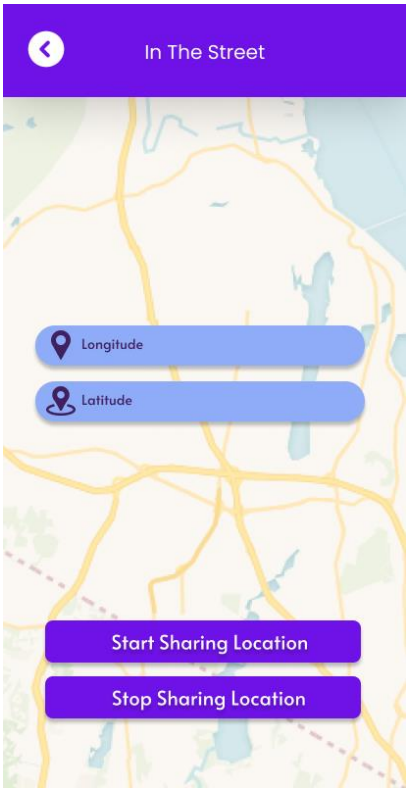


figure 47Start Sharing Location button

And Pedestrians on the road also create two topics

- The first one is static topic with their location, Named by their unique ID.
- The second one is dynamic topic with formatted string of their location to detect the range of their location. including their unique ID and Warning Flag.

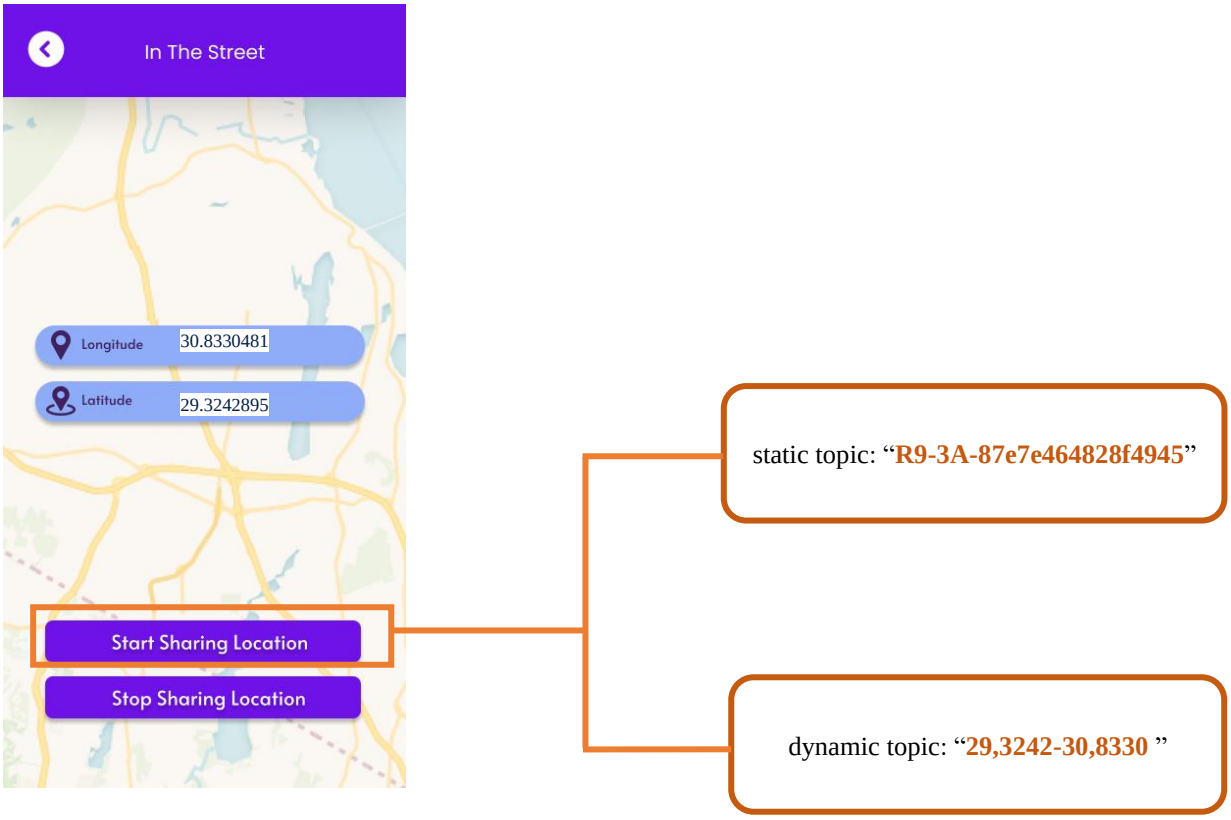
Next figures show the two topics in Firebase:

- The static topic:

figure 48The static topic

- The dynamic topic:

figure 49dynamic topic



4.2.4 Dynamics topic:

Since a person's location is not static and changes over time and movement, with each update to the person's location, we create a new range for this location. As soon as the person moves out of the old range and enters a new range, we delete the old range and create the new range. This way, we have a dynamic topic in Firebase.

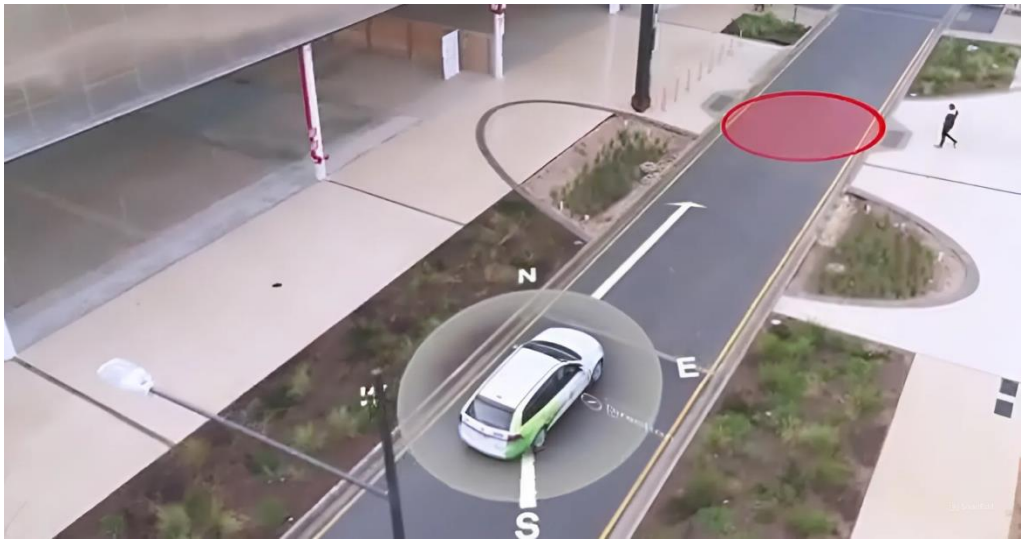


figure 50dynamics topic in real

How we created a dynamic topic in the mobile application?

- Function to get formatted location string

InStreet.java

```
public static String getMyCheckedLocation(double latitude, double longitude) {
    // Format the latitude and longitude to 5 decimal places
    @SuppressLint("DefaultLocale") String checkedAreaLatitude = String.format("%.5f", latitude);
    @SuppressLint("DefaultLocale") String checkedAreaLongitude = String.format("%.5f", longitude);

    // Convert to char arrays for manipulation
    char[] latArray = checkedAreaLatitude.toCharArray();
    char[] lonArray = checkedAreaLongitude.toCharArray();

    // Manipulate specific indices
    latArray[2] = ',';
    lonArray[2] = ',';
    lonArray[7] = ' ';
    latArray[7] = '-';

    // Convert back to strings
    checkedAreaLatitude = new String(latArray);
    checkedAreaLongitude = new String(lonArray);

    // Concatenate the modified strings
    return (checkedAreaLatitude + checkedAreaLongitude);
}
```

codetoimg.com

figure 52function to get formatted location string

- Function to delete old formatted location string in firebase if there is new location.

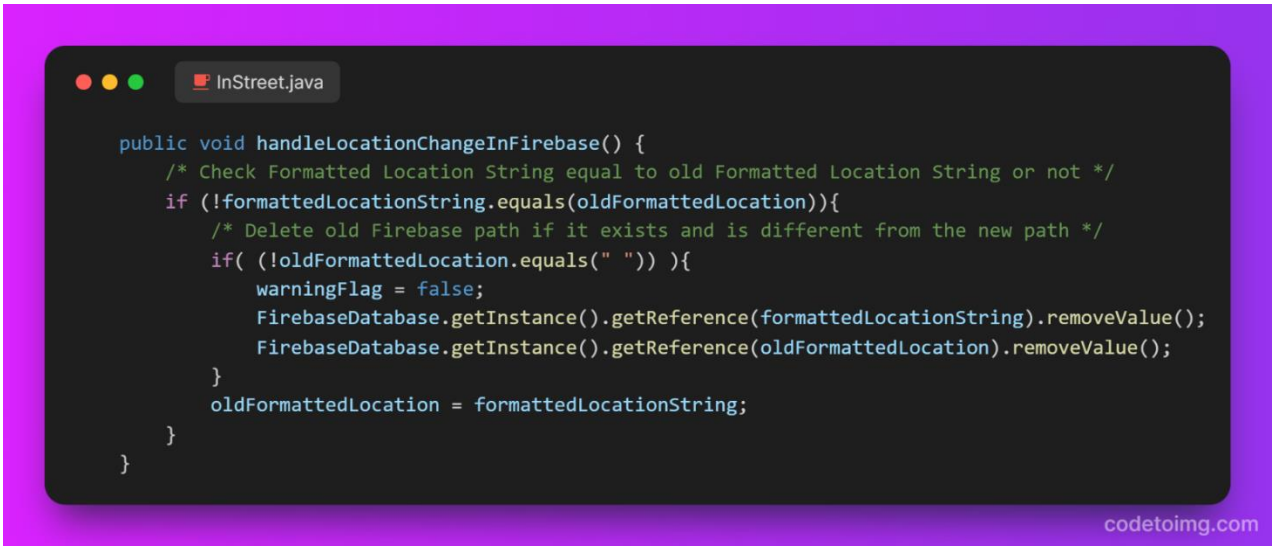


figure 53Function to delete old formatted location

- Function to read the Warning from Firebase.

figure 54Function to read the Warning from Firebase

This function uses a real-time listener that checks for updates in the data stored in Firebase.

When a warning is detected in the database, the application reads the warning flag. If the warning flag is set to true, indicating a potential collision or danger, the application immediately triggers a visual alert for the pedestrian. This is done by transitioning the user to a dedicated warning screen within the app, ensuring they are promptly informed about the imminent danger. This real-time warning system is crucial for enhancing

pedestrian safety by providing immediate alerts about nearby vehicles or other potential hazards.

4.2.5 Action of Vehicle

The vehicle periodically checks the static topic to get its long and lat ,then checks if there another dynamic topic equals to its and this means they are in the same range to send a warning to the driver.

Next figure shows the ESP32 check:

```
02:28:19.830 -> XP7SA1DG9SFC14705/latitude <<---29.324368
02:28:21.012 -> XP7SA1DG9SFC14705/longitude <<---30.833130
02:28:21.247 -> XP7SA1DG9SFC14705/compassDegree <<---26
02:28:21.511 -> 29,3243-30,8331 /id <<---ERROR!
02:28:21.778 -> Not Found
```

figure 55ESP32 check

if there is one equal, this means that two topics are in the same location.

Next step is that vehicle check ID in another topic to detect if it is a pedestrian.

if ID was “R9-3A”, this mean that is a pedestrian, then ESP32 will set warning flag in this topic which its check and transmit specific id number to STM32f103 which will show stop in TFT in vehicle.

Next figure shows communication between firebase and ESP32

```
02:36:44.235 -> XP7SA1DG9SFC14705/latitude <<---29.324217
02:36:44.516 -> XP7SA1DG9SFC14705/longitude <<---30.833076
02:36:44.846 -> XP7SA1DG9SFC14705/compassDegree <<---55
02:36:45.077 -> 29,3242-30,8330 /id <<---R9-3A-87e7e464828f4945
02:36:45.391 -> Found
02:36:45.391 -> XP7SA1DG9SFC14705/dateAndTime <<---Jun 23, 2024 - 3:29:42 AM
02:36:45.733 -> R9-3A-87e7e464828f4945/Jun 23, 2024 - 3:29:42 AM--->> XP7SA1DG9SFC14705
02:36:46.000 -> 29,3242-30,8330 /warning <<---1
02:36:46.302 -> Pedestrian
```

Figure 56communication between firebase and ESP32

Next figures show firebase after warning:

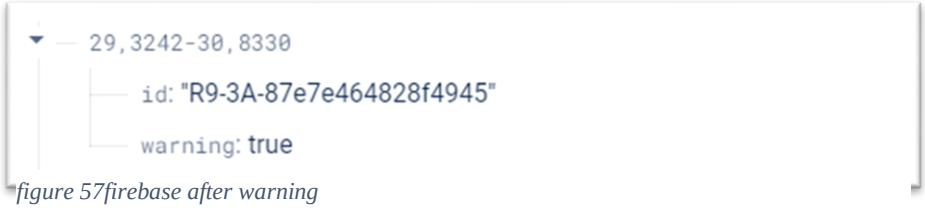


figure 57firebase after warning

When the warning topic in firebase became **True** the warning screen will appear in mobile app and phone will vibrate strongly and a warning sound will play

Next figures shows warning in TFT in vehicle and warning screen in mobile application:

figure 60warning in TFT in vehicle

figure 59warning screen in mobile application

figure 58warning screen in mobile application in street

4.3 V2G: VEHICLE-TO-GRID TECHNOLOGY

4.3.1 Overview

Vehicle-to-grid, or V2G for short, is a technology that enables energy to be pushed back to the power grid from the battery of an electric vehicle (EV). With V2G technology, an EV battery can be discharged based on different signals – such as energy production or consumption nearby.

V2G technology powers bi-directional charging, which makes it possible to charge the EV battery and take the energy stored in the car's battery and push it back to the power grid. While bi-directional charging and V2G are often used synonymously, there is a slight difference between the two.

While bi-directional charging means two-way charging (charging and discharging), V2G technology only enables the flow of the energy from the car's battery back to the grid.

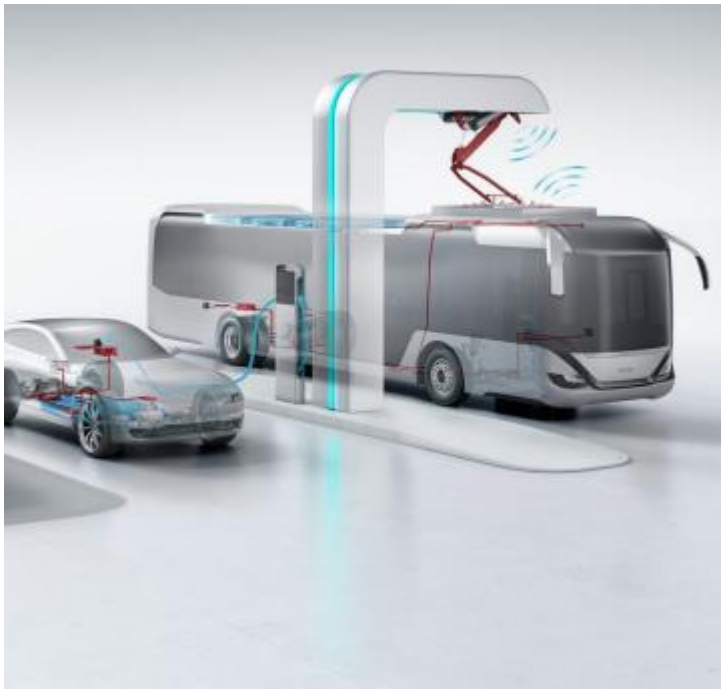


figure 61: vehicle-to-grid

4.3.2 Why should you care about V2G?

Long story short, V2G helps mitigate climate change by allowing our energy system to balance more and more renewable energy. However, to succeed in tackling the climate

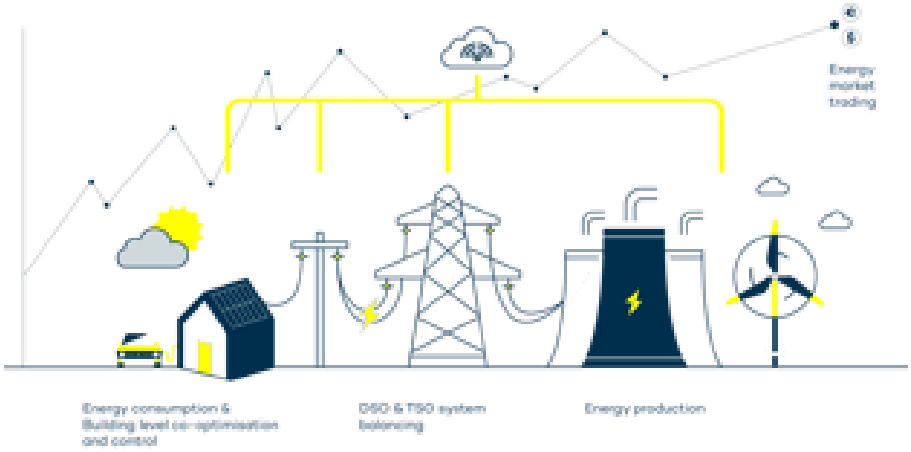


figure 62V2G energy consumption

crisis, three things need to happen in the energy and mobility sectors: Decarbonisation, energy efficiency, and electrification.

In the context of energy production, decarbonisation refers to the deployment of renewable energy sources, such as wind and solar. This introduces the problem of energy storage. While fossil fuels can be seen as a form of energy storage as they release energy when burned, wind and solar power function differently.

This energy should be either used when it's produced or then stored for later usage. As renewable energy production increasingly makes its way into our energy system, it creates more volatility and a need for new ways of balancing and storing renewable energy.

Simultaneously, the transportation sector is doing its fair share of carbon reduction. A notable proof of that is the number of EVs on our roads, which is steadily increasing. In 2022, 14% of all cars sold were electric, while that number was only 5% in 2020.

EV batteries are by far the most cost-efficient form of energy storage since they require no additional investments in hardware. With V2G, we can utilise the battery capacity up to 10x more efficiently than with regular smart charging. Vehicle-to-grid technology enables us to make the best use of the existing population of vehicles. And by 2030, there could be up to 250 million EVs globally. That means that we'll have around 250 million tiny energy storages on wheels. Research actually shows that by the end of this decade, EV batteries should be able to meet the demand for short-term energy storage.

4.3.3 Benefits of V2G

Certainly! Vehicle-to-Grid (V2G) technology offers several benefits:

- **Supporting the Electrical Grid**: V2G helps reduce concerns about grid overload by directing the charging and discharging of EV batteries based on users' needs and the grid's electricity supply
- **Maximizing Business Opportunities**: EV owners can maximize the business case for their vehicles by participating in V2G programs.
- **Cheap and Fast Energy Storage**: V2G allows EVs to serve as cost-effective energy storage solutions, benefiting both the grid and EV owners.
- **Optimizing Existing Resources**: By using existing EV batteries, V2G optimizes the supply of local renewable energy and reduces infrastructure costs.
- **Reducing Environmental Impact**: V2G contributes to mitigating climate change by balancing more renewable energy in our energy system

4.3.4 How V2G works

figure 63How V2G work

Firstly, cables of electric station should be plugged to vehicle.

To start communication between them user should push button 1.

After that lcd of station starts with *"welcome to V2G "*then if cable plugged to vehicle.

Another message is sent to Lcd *"plugged cable success"* then by communication vehicle automatically sends its identification number (VIN) - is a unique code assigned to every motor vehicle when it's manufactured. The VIN is a 17-character.

Each section of the VIN provides a specific piece of information about the vehicle, including the year, country, and factory of manufacture; the make and model; and the serial number.

VINs are usually printed in a single line- and station takes VIN and check manufacturer and serial number in data base.

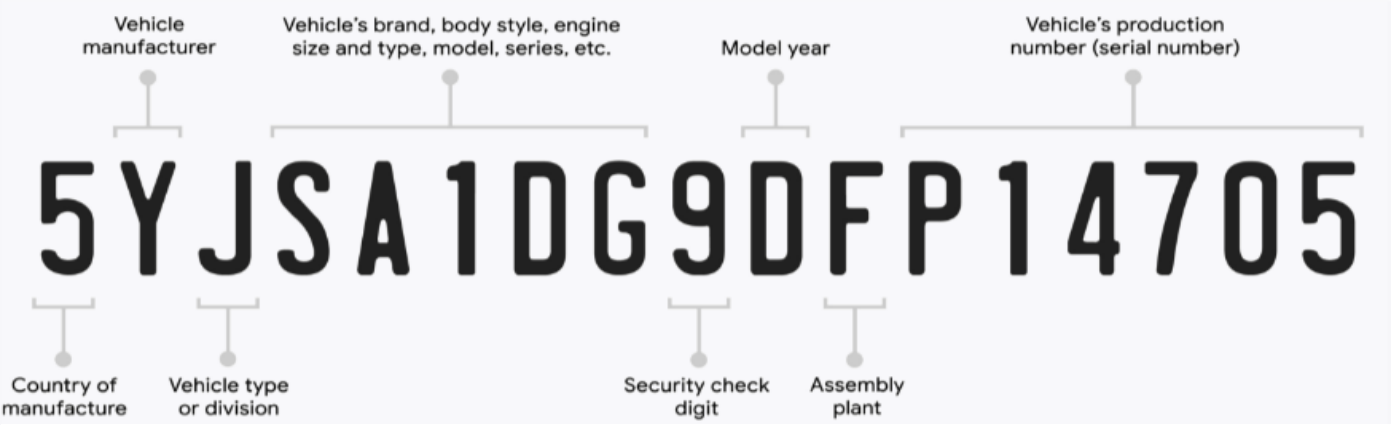


figure 64VIN

If station supports manufacturer of this vehicle, it then checks if it supports this serial number or not –has an account or not.

If serial is exists in database and has balance in database it accepts vehicle and sends message to lcd-on station- with "-VIN of vehicle - , we find you ".

Station after that checks if percentage of voltage of vehicle less than 70% it accepts charging.

if not, it refuses then sends a message to Lcd.

If station accepts charging, user can stop it manually by pushing a button 2.

In parallel, voltage sensor calculates battery's voltage using ADC-Analog to digital converter- and show percentage value on TFT maintained on car.

figure 65LCD energy amount value

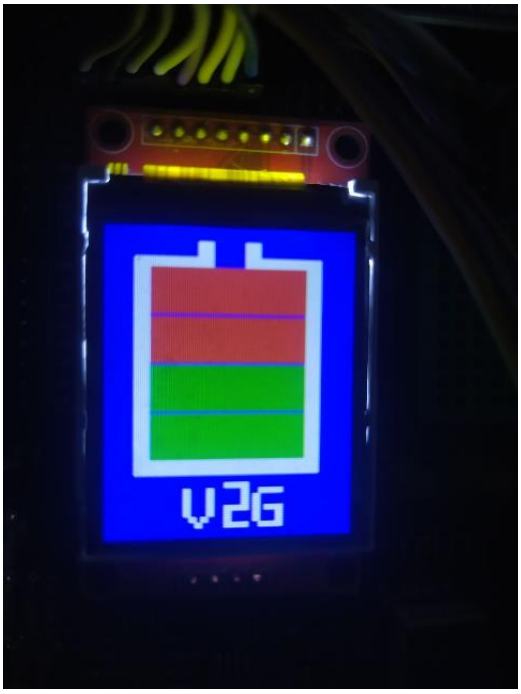


figure 66GUI charging

figure 67start charge GUI

Finally, station calculate how much money of charging, then adding it to vehicle's account in fire base.

figure 68end charge GUI

4.3.5 Code Illustrate:

Communication during UART protocol, frame consists of length, code id and data.

frame from vehicle to station:

Length (code &data)	**Code Id**	**Data**
1 byte	**1 byte**	**Variable length**

Table 1: frame from veicle to station

frame from station to vehicle:

ACK ID	**Length of data**	**data**
1 Byte	**1byte**	**Variable**

Table 2:frame from station to vehicle

uint8_t EVCC_u8PlugCable (void)

void SE_c (uint8_t * Copy_pu8CmdPacket)

In this function, it checks if cable plugged well and there are no problems in the connection.

void SE_voidIdentification (uint8_t *Copy_pu8CmdPacket)

uint8_t c_u8Identification (void)

This function sends VIN to station and station extracts manufacturer and serial number to check it by comparing it with VINs in database to accept charging or not.

void SE_voidChargeParameter (uint8_t * Copy_pu8CmdPacket)

uint8_t EVCC_u8ChargeParameter (void)

This function read voltage from voltage sensor and send percentage to station. Station checks percentage, if it exceeds 70% it refuses charging.

Next figure show V2G station before charging on firebase.

figure 69V2G station before charging on firebase

uint8_t EVCC_u8ChargingProfiles (void)

void SE_voidChargingCost (uint8_t * Copy_pu8CmdPacket)

Cost is calculated in this function by subtracting volt before charging from voltage after charging then multiply result with constant number. After that it sends result to station to subtract it from your balance in data base.

Next figure calculate money on firebase.

figure 70calculate money on firebase

void SE_voidStartCharging (uint8_t * Copy_pu8CmdPacket)

void SE_voidEndCharging (uint8_t * Copy_pu8CmdPacket)

In vehicle it just sends code id to station to start charging, Station receives this code and then starts charging, also relay turns on to turn led on. And vice versa, End charging turns relay and subsequently led off.

void EVCC_voidSendEnergyAmount (void)

void SE_voidSendEnergyAmount (uint8_t * Copy_pu8CmdPacket)

It is called periodically by timer to measure voltage to send to station, then showing that percentage during charging on lcd.

Next figure show firebase after charging.

figure 71: show firebase after charging

4.4 SELF-DRIVING-CAR

4.4.1 Introduction

- self-driving navigation system for a vehicle, leveraging the capabilities of the ESP32 microcontroller, GPS data, and Firebase Realtime Database. This system shows how a microcontroller can be utilized to create an autonomous vehicle that navigates through predefined waypoints using real-time GPS coordinates and compass direction
- Background on self-driving technology
- Importance and applications of autonomous vehicles

figure 72self driving

Autonomous vehicles have a wide range of applications, including:

Transportation: Autonomous cars can provide safe and efficient transportation for individuals and goods.

Logistics: Self-driving trucks and delivery drones can optimize supply chains and reduce delivery times.

Accessibility: Autonomous vehicles can offer mobility solutions for people with disabilities or the elderly.

Public Safety: Self-driving technology can reduce accidents caused by human error, improving overall road safety

- Overview of the project

  This project demonstrates the implementation of a basic self-driving car using the ESP32 microcontroller, GPS data, and Firebase Realtime Database. The vehicle is designed to navigate through a series of predefined waypoints autonomously. The ESP32 handles the control logic, while Firebase provides real-time data storage and retrieval. This project showcases the integration of hardware and software to achieve autonomous navigation.

4.4.2 System Architecture

- Hardware components
  - ESP32 microcontroller
  - GPS module
  - Compass sensor
  - Motors and actuators
- Software components
  - Arduino IDE
  - Firebase Realtime Database
  - Wi-Fi communication

4.4.3 How Implementation

- Setting up the hardware
- Configuring Wi-Fi and Firebase
- Programming the ESP32
- Key functions and their roles
- Initialization
  - Data retrieval and processing
  - Navigation logic
  - Movement control

1. Algorithm and Navigation

- Distance calculation using the Haversine formula
- Waypoint navigation strategy
- Direction determination and correction

2. **Testing and Results**
    - Testing environment and setup
    - Simulated results and observations
    - Challenges and troubleshooting
3. **Conclusion**
    - Summary of achievements
    - Potential improvements
    - Future work and applications

4.5 IMPLEMENTATION

4.5.1 Setting up the Hardware

The self-driving car project is built around the ESP32 microcontroller, which serves as the brain of the system. The essential hardware components include:

- **ESP32 Microcontroller**: This microcontroller is chosen for its powerful processing capabilities and built-in Wi-Fi module.
- **GPS Module**: Provides real-time latitude and longitude coordinates, essential for navigation.
- **Compass Sensor**: Gives the current heading direction of the vehicle.

4.5.2 Software Components

- **Arduino IDE**: The Arduino Integrated Development Environment (IDE) is used to program the ESP32. It provides a user-friendly interface and a vast library of functions for hardware control and communication.
- **Firebase Realtime Database**: Firebase is used to store and retrieve the vehicle's GPS coordinates and compass data. It provides a scalable and reliable cloud-based database solution.
- **Wi-Fi Communication**: The ESP32 connects to a Wi-Fi network to communicate with Firebase, enabling real-time data exchange.

The hardware is assembled on a chassis, with the GPS module and compass sensor mounted securely to ensure accurate readings which allows for precise control over the vehicle's movement.

4.5.3 Configuring Wi-Fi and Firebase

The ESP32 connects to a Wi-Fi network to communicate with Firebase, where the real-time database stores critical data like GPS coordinates and the compass heading.

- **Key Functions and Their Roles**

1- Data Retrieval and Processing:

- Fetches the current GPS coordinates and compass heading from Firebase.

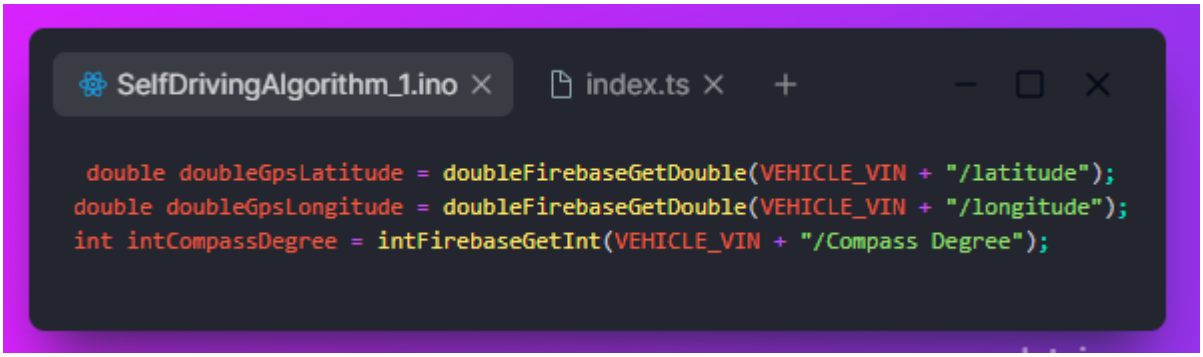


figure 73Fetches the current GPS coordinates and compass heading from Firebase

2- Getting compass Direction

intGetDirection: The direction to the next waypoint is determined by calculating the bearing between the current location and the waypoint. This bearing is compared to the current compass heading to decide the necessary direction of movement

- Determines the direction to turn based on the current heading and the bearing to the waypoint

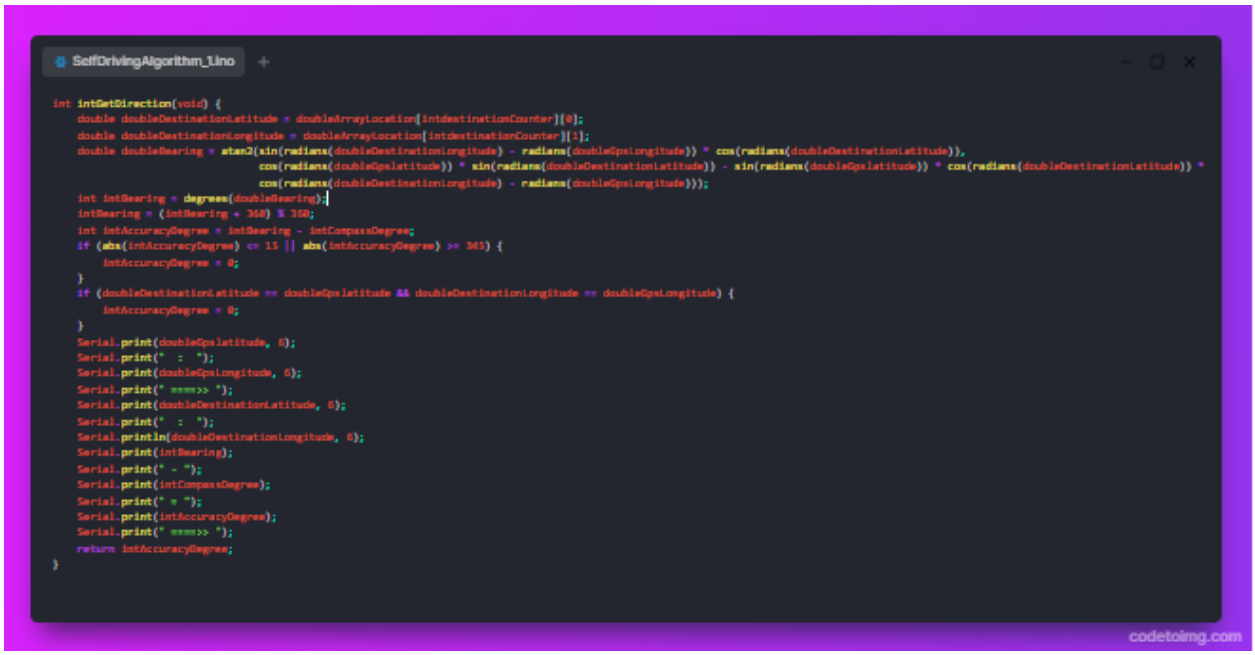


figure 74Getting compass Direction

3- Update Destination

Calculates the distance to the next waypoint using the Haversine formula.

The Haversine formula is used to calculate the great-circle distance between two points on the Earth's surface. This formula accounts for the spherical shape of the Earth and provides accurate distance measurements.

The vehicle follows a series of predefined GPS waypoints. Each waypoint is a latitude and longitude coordinate that the vehicle aims to reach sequentially. The navigation strategy involves:

1-Calculating the distance to the next waypoint.

2-Checking if the vehicle is within a 3-meter radius of the waypoint.

3-Updating to the next waypoint if the current one is reached.

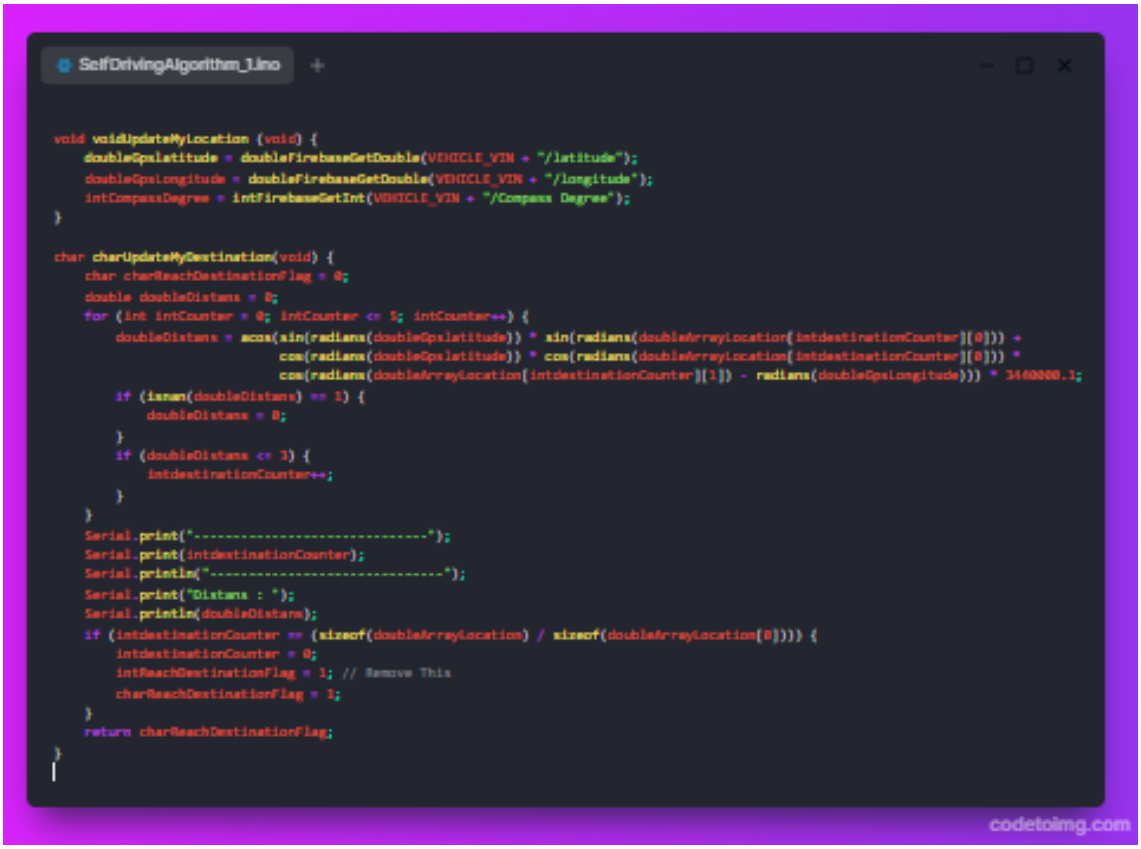


figure 75Update Destination

4-Move Function

- Determines the movement action based on the direction and updates the action ID
- This function we can select the direction of movement whether to move forward, Right, Left or stope when reach the destination Depending on the Calculations from the functions "**charUpdateMyDestination"** and "**intGetDirection**"

4.5.4 Testing and Results

- **Testing Environment and Setup**

To validate the self-driving car system, the following testing environment was set up:

- A controlled outdoor area with clear GPS signal reception.
- A series of predefined waypoints marked with GPS coordinates.
- A laptop or smartphone to monitor Firebase and ensure real-time data updates.

- **Simulated Results and Observations**

During testing, the vehicle successfully navigated through the waypoints with the following observations:

- The GPS module provided accurate location data, allowing precise navigation.
- The compass sensor effectively determined the vehicle's heading, facilitating correct directional adjustments.
- The Firebase Realtime Database efficiently handles data storage and retrieval, ensuring timely updates.

Challenges encountered included occasional GPS signal loss and minor deviations due to sensor inaccuracies. These were corrected by trying Many attempts .
GPS Signal Loss In areas with poor GPS reception, the vehicle struggled to maintain accurate positioning. Solutions included using a higher-quality GPS module and ensuring a clear line of sight to the sky.

4.5.5 Conclusion

- **Summary of Achievements**

This self-driving car project successfully demonstrated the integration of ESP32 microcontroller, and Firebase Realtime Database to achieve autonomous navigation. Key achievements include:

- Accurate distance calculation using the Haversine formula.
- Effective waypoint navigation and directional control.
- Real-time data management with Firebase.

- **Potential Improvements**

Future iterations of this project could include:

- Enhancing GPS accuracy with advanced modules or differential GPS techniques.

- Incorporating additional sensors like ultrasonic or LiDAR for obstacle detection and avoidance.
- Implementing machine learning algorithms for improved decision-making and route optimization.

- **Future Work and Applications**

This project lays the groundwork for more advanced autonomous vehicle systems. Future work could explore:

- Integration with traffic management systems for coordinated vehicle movements.
- Development of autonomous delivery systems for urban logistics.
- Expansion to larger, more complex environments with multiple vehicles operating simultaneously.

Chapter Five

5 MATERIAL AND METHODOLOGY

5.1 MAIN HARDWARE COMPONENTS

Stm32f103

GPS

Character LCD

Relay

TFT (thin film transistor)

Voltage sensor

ESP32 module

Mobile app

Ultrasonic sensor

L298N H-bridge

Dc motor

5.2 HARDWARE

5.2.1 Stm32f103

The STM32F103x4 and STM32F103x6 performance line family incorporates the high-performance ARM® Cortex™-M3 32-bit RISC core operating at a 72 MHz frequency, high-speed embedded memories (Flash memory up to 32 Kbytes and SRAM up to 6 Kbytes), and an extensive range of enhanced I/Os and peripherals connected to two APB buses. All devices offer two 12-bit ADCs, three general purpose 16-bit timers plus one PWM timer, as well as standard and advanced communication interfaces: up to two I^2Cs and SPIs, three USARTs, an USB and a CAN.

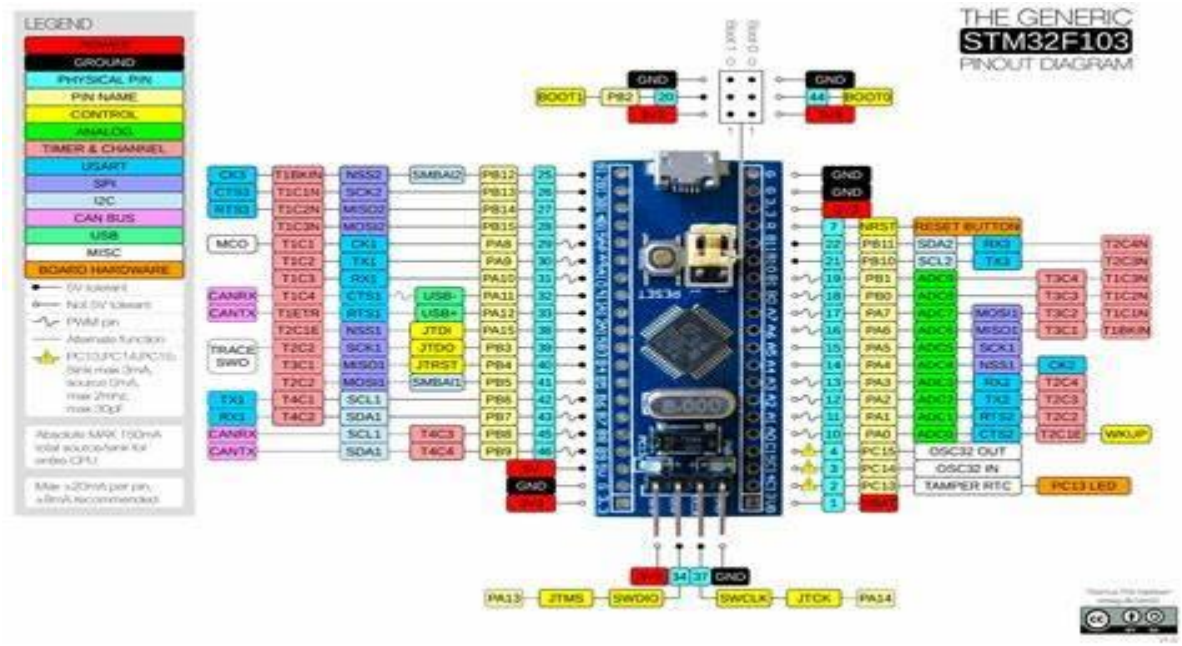


figure 76STMF103 pinout

The STM32F103xx low-density performance line family operates from a 2.0 to 3.6 V power supply. It is available in both the –40 to +85 °C temperature range and the –40 to +105 °C extended temperature range. A comprehensive set of power-saving mode allows the design of low-power applications.

The STM32F103xx low-density performance line family includes devices in four different package types: from 36 pins to 64 pins. Depending on the device chosen, different sets of peripherals are included, the description below gives an overview of the complete range of peripherals proposed in this family.

These features make the STM32F103xx low-density performance line microcontroller family suitable for a wide range of applications such as motor drives, application control, medical and handheld equipment, PC and gaming peripherals, GPS platforms, industrial applications, PLCs, inverters, printers, scanners, alarm systems, video intercoms, and HVACs

5.2.2 Circuit diagram

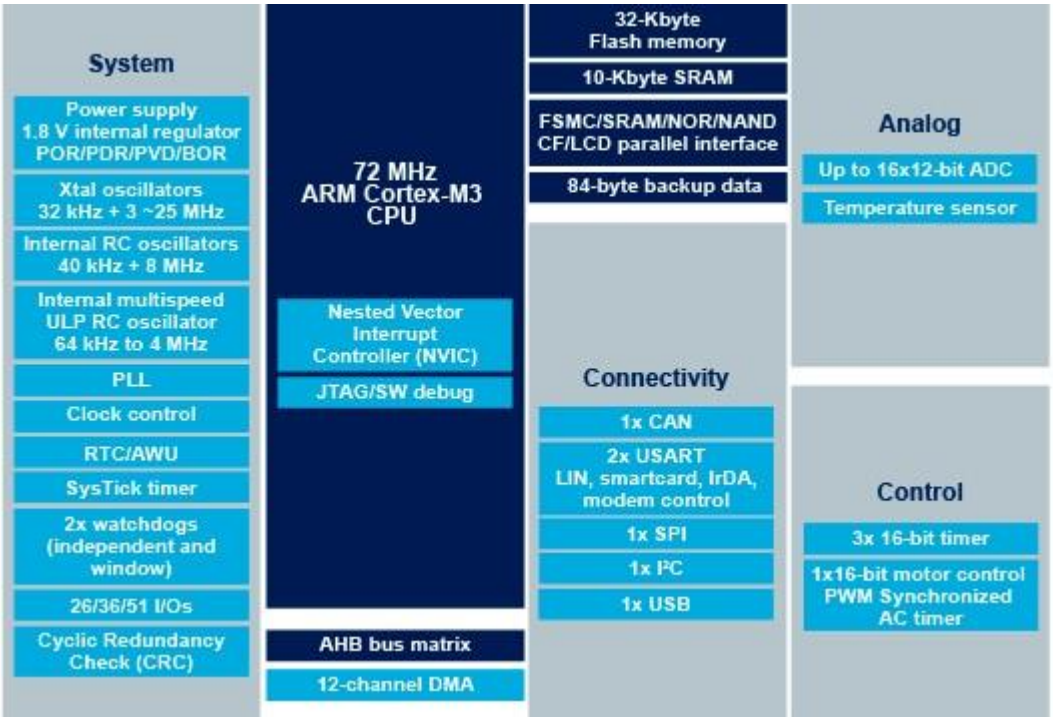


figure 77Circuit diagram

5.2.3 FEATURE

ARM 32-bit Cortex™-M3 CPU Core

72 MHz maximum frequency,1.25 DMIPS/MHz (Dhrystone 2.1) performance at 0 wait state memory access

Single-cycle multiplication and hardware division

Memories

16 or 32 Kbytes of Flash memory

6 or 10 Kbytes of SRAM

Clock, reset and supply management

2.0 to 3.6 V application supply and I/Os

POR, PDR, and programmable voltage detector (PVD)

4-to-16 MHz crystal oscillator

Internal 8 MHz factory-trimmed RC

Internal 40 kHz RC

PLL for CPU clock

32 kHz oscillator for RTC with calibration

Low power

Sleep, Stop and Standby modes

VBAT supply for RTC and backup registers

2 x 12-bit, 1 μs A/D converters (up to 16 channels)

Conversion range: 0 to 3.6 V

Dual-sample and hold capability

Temperature sensor

DMA

7-channel DMA controller

Peripherals supported: timers, ADC, SPIs, I^2Cs and USARTs

Up to 51 fast I/O ports

26/37/51 I/Os, all mappable on 16 external interrupt vectors and almost all 5 V-tolerant

Debug mode

Serial wire debug (SWD) & JTAG interfaces

6 timers

Two 16-bit timers, each with up to 4 IC/OC/PWM or pulse counter and quadrature (incremental) encoder input

16-bit, motor control PWM timer with dead-time generation and emergency stop

2 watchdog timers (Independent and Window)

SysTick timer 24-bit downcounter

6 communication interfaces

1 x I^2C interface (SMBus/PMBus)

2 × USARTs (ISO 7816 interface, LIN, IrDA capability, modem control)

1 × SPI (18 Mbit/s)

CAN interface (2.0B Active)

USB 2.0 full-speed interface

CRC calculation unit, 96-bit unique ID

5.1 GLOBAL POSITION SYSTEM (GPS):

For the vehicle to get a full view of its surroundings whether these surroundings are other vehicles, pedestrians, road pumps, etc., that car needs to know its coordinates and its position relative to them. And to do so, we need to use a GPS module in each car to receive its coordinates using GPS satellites and also to receive the coordinates of other vehicles embedded in their frames through the WI-FI network.

5.1.2 GPS Theory

GPS works by measuring time. Specifically, how long it takes a signal from the GPS satellites to reach the receiver. Since the speed of light is known, we can use the exact travel time to calculate the distances between the satellite and the receiver. And if we can do this with multiple satellites, all with different distances to the receiver, we can calculate a unique position for the receiver using nothing but a very good clock, knowing the speed of light, and a hardcore geometrical technique called trilateration. All this happens with a coded signal from the GPS satellites, plus some atomic clocks aboard each satellite. This signal is packed with information about the time (from the onboard atomic clocks), the orbital information for that specific GPS satellite, and the current GPS almanac: the ID of the satellite, the state of the GPS constellation, and error correction. And we know that the signal comes to your receiver at the speed of light, allowing a bit of the effect of Special Relativity.

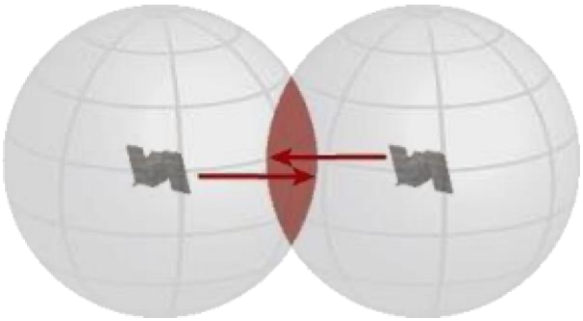
This is what happens, albeit a bit simplified. You are standing on the surface of the Earth, and there are at least three GPS satellites in view in the sky above you. Since we know the speed of light and the location of each satellite in space, in effect we can use

figure 78GPS Theory

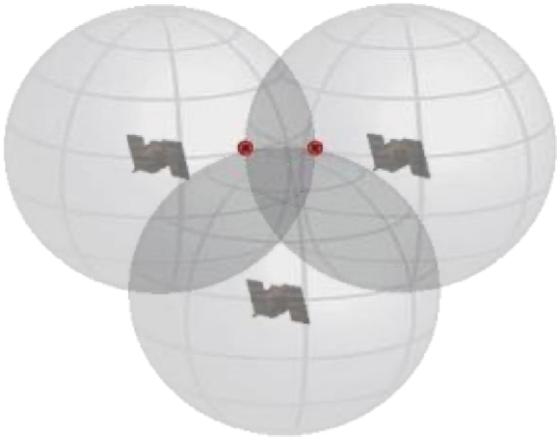
the GPS satellites as a reference point to triangulate our location on the surface of the Earth.

Step One: The location of the GPS receiver is somewhere on the surface of a sphere with a radius of 20,200 kilometers. This is the orbital altitude of the constellation

Step Two: Since all GPS satellites orbit at the same altitude (but orbit in one of six different paths), if we add a second satellite that is in view of the point, we can narrow down the area where the true location can be to the intersection of the areas on the surface of the earth that are visible to both.



Step Three: Adding a third satellite narrows things even further, to the two points where the third satellite's sphere intersects the first two



Typically, one of the two points used for calculating the distance between a GPS receiver and a satellite is meaningless and can be disregarded. The "true point" refers to a specific location on the Earth's surface, and the actual distance between the GPS receiver and the satellite may not always be precisely 20,200 kilometers due to variations in the receiver's location and the satellite's orbital path. To achieve better accuracy, we can calculate the precise distance between the receiver and each satellite by using the coded signal transmitted by the GPS satellites and the knowledge that the

speed of light is roughly 300,000 kilometers/second. This is accomplished by computing the pseudo-range to each satellite, which involves using the PRN code embedded in the GPS signal along with the GPS information described earlier. Each satellite has a unique PRN code, and the receiver uses this code to calculate the pseudo range to each satellite

To determine the travel time of a signal, the GPS receiver compares the PRN it receives from a satellite to the same PRN it generates locally. Initially, the signals are out of phase and don't match. The receiver then shifts its signal until it matches the one from the satellite. The distance to the satellite is equal to the product of the amount of signal shift and the speed of light. This technique is effective because each satellite broadcasts a GPS signal with consistent time signatures from its onboard atomic clock, which is synchronized by the U.S.

Naval Observatory.

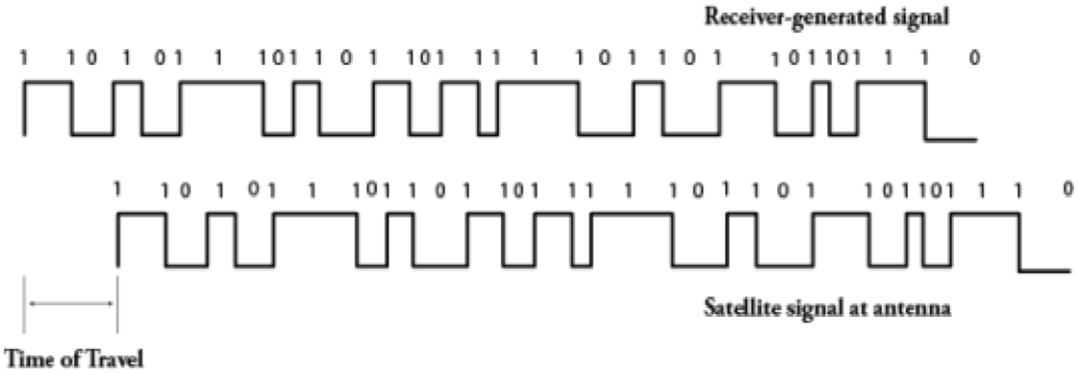


figure 79GPS signal with consistent time signatures from its onboard atomic clock

5.2 ESP8266 module

The ESP8266 is a highly popular Wi-Fi module developed by Espressif Systems. Here's a comprehensive overview:

figure 80esp8266 module

5.2.1 Overview:

- Manufacturer: Espressif Systems
- Release Date: The ESP8266 module was first released in 2014.
- Main Features: It's known for its low cost, low power consumption, and ease of integration into IoT (Internet of Things) projects.
- Microcontroller: The ESP8266 integrates a Tensilica L106 32-bit microcontroller unit (MCU), which handles the Wi-Fi connectivity and general-purpose processing tasks.
- Wi-Fi: It supports IEEE 802.11 b/g/n Wi-Fi standards, allowing devices to connect to Wi-Fi networks and communicate over the internet.
- Flash Memory: Typically, ESP8266 modules come with varying amounts of flash memory, commonly ranging from 512KB to 4MB. This memory is used to store the firmware, program code, and other data.
- GPIO Pins: The ESP8266 modules feature a number of GPIO (General Purpose Input/Output) pins, which can be used for interfacing with sensors, actuators, and other external components.
- Operating Voltage: Most ESP8266 modules operate at 3.3V, which means they require level shifting or voltage dividers to interface with 5V systems.

5.2.2 Variants:

1. ESP-01: The ESP-01 is one of the earliest and simplest variants of the ESP8266. It features a small form factor and limited GPIO pins.
2. ESP-12: The ESP-12 is a more feature-rich variant with more GPIO pins and often more flash memory.
3. NodeMCU: NodeMCU is a development board based on the ESP8266, featuring USB connectivity for easy programming and debugging. It's popular among hobbyists and developers for its ease of use.
4. Wemos D1 Mini: Similar to NodeMCU, the Wemos D1 Mini is another development board based on the ESP8266. It's compact and offers a variety of shields for easy expansion.
5. ESP32: While not a variant of the ESP8266, the ESP32 is worth mentioning. It's a successor to the ESP8266, offering more GPIO pins, Bluetooth connectivity, and additional features.

5.2.3 Programming:

- The ESP8266 can be programmed using the Arduino IDE, making it accessible to a wide range of developers.
- Additionally, it supports programming using Lua scripting language through platforms like Node MCU.

5.2.4 Applications:

- Home automation
- IoT devices
- Sensor networks
- Wi-Fi-enabled gadgets
- Smart appliances

5.2.5 Community and Resources:

- The ESP8266 has a large and active community of developers and enthusiasts, providing extensive documentation, tutorials, and libraries.

- Platforms like GitHub, Hackster.io, and Instructables host numerous ESP8266 projects and resources.

Overall, the ESP8266 has revolutionized the DIY electronics and IoT landscape due to its affordability, versatility, and ease of use.

5.3 TFT

The ST7735S is a popular display controller chip manufactured by Sitronix. It's commonly used in small color TFT LCD displays found in various electronic devices such as Arduino projects, Raspberry Pi accessories, and other embedded systems. The ST7735S supports SPI (Serial Peripheral Interface) communication, making it easy to interface with microcontrollers and other devices. It's known for its low power consumption, decent refresh rates, and ease of use, making it a popular choice for hobbyists and DIY electronics enthusiasts.

figure 81TFT screen

5.4 BLUETOOTH

The **HC-05** module is a popular Bluetooth module that enables wireless communication between electronic devices over short distances. Here are some key details about the HC-05:

5.4.1 Functionality: The HC-05 module allows two-way (full-duplex) wireless communication. It uses Bluetooth technology to establish a serial communication link between a microcontroller (such as Arduino) or other embedded systems and a host device (e.g., a smartphone, tablet, or computer).

5.4.2 Pinout and Specifications:

- **Enable / Key (Pin 1)**: Toggles between Data Mode (low) and AT command mode (high). By default, it operates in Data mode.
- **Vcc (Pin 2)**: Powers the module (connect to +5V supply voltage).
- **Ground (Pin 3)**: Connects to system ground.
- **TX (Transmitter, Pin 4)**: Transmits serial data received via Bluetooth.
- **RX (Receiver, Pin 5)**: Receives serial data to be broadcasted via Bluetooth.
- **State (Pin 6)**: Connected to an onboard LED for feedback on Bluetooth operation.
- **LED (Pin 7)**: Indicates module status (e.g., Command Mode, successful connection in Data Mode).
- **Button (Pin 8)**: Used to control the Key/Enable pin for mode toggling.

5.4.3 Default Settings:

- **Bluetooth Name**: “HC-05”
- **Default Password**: 1234 or 0000
- **Default Communication**: Slave
- **Default Mode**: Data Mode
- **Data Mode Baud Rate**: 9600, 8, N, 1
- **Command Mode Baud Rate**: 38400, 8, N, 1
- **Default Firmware**: LINVOR

5.4.4 Technical Specifications:

- Operating Voltage: 4V to 6V (typically +5V)
- Operating Current: 30mA
- Range: <100m
- Supports serial communication (USART) and TTL compatibility
- Follows IEEE 802.15.1 standardized protocol
- Uses Frequency-Hopping Spread Spectrum (FHSS)
- Can operate in Master, Slave, or Master/Slave mode
- Easily interfaced with laptops or mobile phones via Bluetooth
- Supported baud rates: 9600, 19200, 38400, 57600, 115200, 230400, 460800

5.4.5 Applications:

- Communicate between microcontrollers (e.g., Arduino) or devices with Bluetooth functionality (phones, laptops).
- Android applications simplify the process.

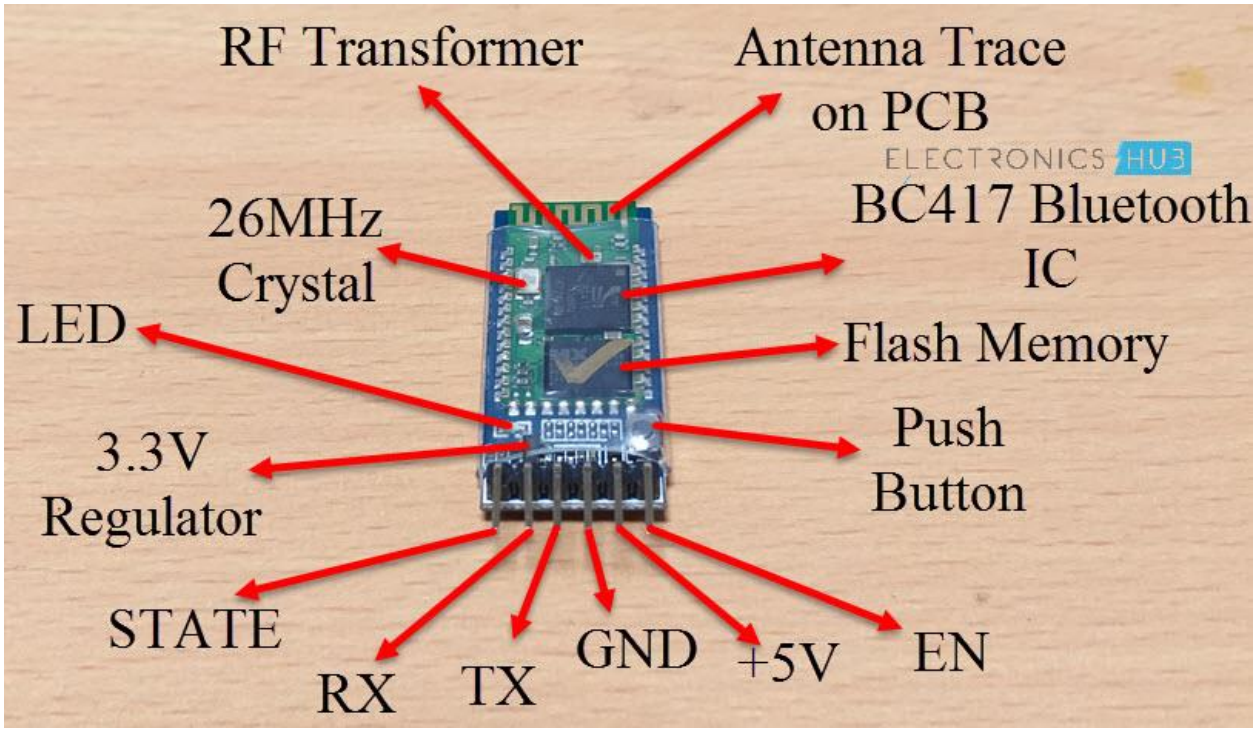


figure 82bluetooth module

5.5 VOLTAGE SENSOR MODULE

The voltage sensor module is a small size 0-25 DC voltage sensing device. The design of the module is based on a resistive voltage divider circuit. It is a voltage sensor module that reduces the input voltage signal by a factor of 5 and generates a corresponding analog output voltage concerning step down voltage factor. This voltage measurement circuit is small and portable and can be used to detect under and over-voltage faults in electrical circuits.

Voltage Sensor Module Pin out Configuration

Pin Name	Description
Vin	The positive terminal of the External voltage source (0-25V)
GND	The negative terminal of the External voltage source

Vout	Analog pin connected to Analog pin of MICROCONTRLER
NC	Not Connected
–	Ground Pin connected to GND of MICROCONTRLER

Table 3: Voltage Sensor Module Pin out Configuration

Voltage Detection Sensor Module Features & Specifications

- Input Voltage: 0 to 25V
- Voltage Detection Range: 0.02445 to 25
- Analog Voltage Resolution: 0.00489V
- Needs no external components
- Easy to use with Microcontrollers
- Small, cheap, and easily available
- Dimensions: 25 × 13 × 13 mm

Brief about Voltage Sensor Module

The voltage Detection Sensor Module is a simple and very useful module that uses a potential divider to reduce any input voltage by a factor of 5. This allows us to use the Analog input pin of a microcontroller to monitor voltages higher than it capable of sensing. For example, with a 0V – 5V Analog input range, you can measure a voltage up to 25V. This module also includes convenient screw terminals for easy and secure connections of a wire.

The internal **circuit diagram of the Voltage Sensor Module** is given below.

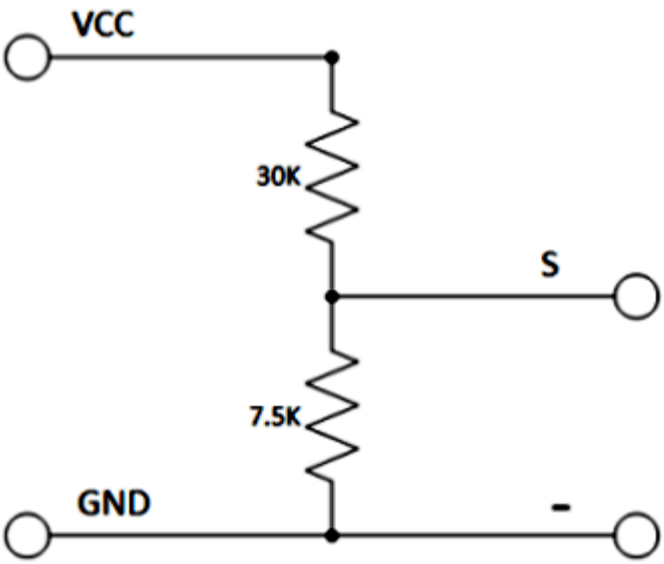


figure 83circuit diagram of the Voltage Sensor Module

The voltage circuit consists of a voltage divider circuit of two resistors in which R1 is 30K and R2 is 7.5K.

How to Use Voltage Sensor Module with Arduino

Interfacing a voltage sensor with Arduino or any other **microcontroller** is pretty straightforward. Connect the V_{CC} and GND of the voltage source whose voltage is to be measured to the screw terminals of the voltage sensor. Connect the S and – (GND) pins of the voltage sensor to the Analog pin and GND of Arduino respectively.
Input and output voltage can be calculated using:

Vin = Vout * (R2/(R1+R2)) Here R1 = 30K ohm and R2 = 7.5K ohm Vout = (analogvalue * 5 / 1024).

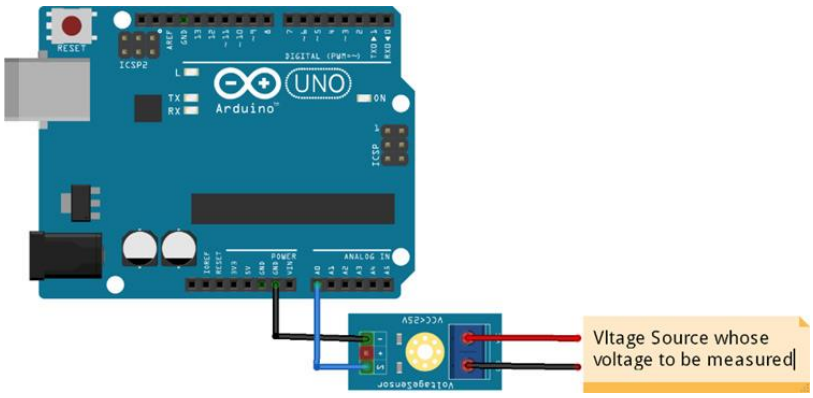


figure 84Use Voltage Sensor Module with Arduino

5.6 CHARACTER LCD MODULE

A **16x2 LCD** (Liquid Crystal Display) is a common type of display module used in various electronic devices. Let me provide you with some details:

- **Display Size**: A 16x2 LCD can show **16 characters** in each of its **two rows**, providing a total of **32 characters** of information.
- **Pixel Dots**: Each character is made up of **5x8 (40)** pixel dots.
- **Working Principle**: Instead of electrons diffraction at a glass display, a liquid crystal display has a backlight that provides light to each pixel arranged in a rectangular network. Every pixel includes a blue, red, and green sub-pixel that can be switched ON/OFF.
- **Specifications**:
    - Operating voltage: **4.7V to 5.3V**
    - Display bezel size: **72 x 25mm**
    - Operating current (without backlight): **1mA**
    - HD47780 controller
    - LED color for backlight: **green or blue**
    - Number of columns: **16**
    - Number of rows: **2**
    - Number of LCD pins: **16**
    - **Font size of character: 0.125 width x 0.200 height2**

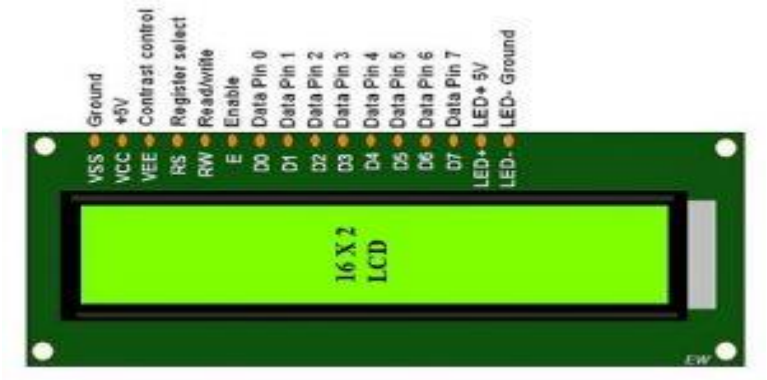


figure 85LCD screen

5.6.1 16×2 LCD pinout

The LCD has 16 connection pins, numbered 1-16 from left to right.

The pins at the top of the display are numbered 1-16 from left to right. The pinout of a standard HD44780 LCD is given in the table below:

Pin no.	Symbol	Connection	Function
1	VSS	GND	Signal ground
2	VDD	5 V	Logic power for LCD
3	V0	10 kΩ potentiometer	Contrast adjustment
4	RS	Pin 2	Register select signal
5	R/W	GND	Read/write select signal
6	E	Pin 3	Operation enable signal
7 – 14	D0 – D7	–	Data bus lines used for 8-bit mode
11 – 14	D4 – D7	Pin 4 – 7	Data bus lines used for 4-bit mode
15	A (LED+)	5 V	Anode for LCD backlight
16	K (LED-)	GND	Cathode for LCD backlight

Table 4: The pinout of a standard HD44780 LCD

5.6.2 Working Principle

The basic **working principle of LCD** is passing the light from layer to layer through modules. These modules will vibrate & line up their position on 90^{o} that permits the polarized sheet to allow the light to pass through it.

These molecules are accountable for viewing the data on every pixel. Every pixel utilizes the method of absorbing light to illustrate the digit. To display the value, the position of molecules must be changed to the angle of light.

So this light deflection will make the human eye notice the data that will be the ingredient wherever the light gets absorbed. Here, this data will supply to the molecules & will be

there till they get changed
At present, LCDs are used frequently in CD/DVD players, digital watches, computers, etc. In screen industries, LCDs have replaced the CRTs (Cathode Ray Tubes) because these displays use more power as compared to LCD, heavier & larger.

The displays of LCDs are thinner as compared to CRTs. As compared to LED screens, LCD has less power consumption because it functions on the fundamental principle of blocking light instead of dissipating.

5.6.3 Registers of LCD

The registers used in LCD are two types like data register & command register. The register can be changed by using the RS pinout. If we set '0' then it is command register and if it is '1' then it is data register.

Command Register

The main function of the command register is to save instructions illustrated on LCD. That assists in data clearing & changes the cursor location & controls the display.

Data Register

The data register is used to save the date to exhibit on the LCD. Once we transmit data to LCD, then it shifts to the data register to process the data. If we fix the register value at one that the data register will start working.

5.7 RELAY MODULE

5.7.1 What is a Relay Module?

Relay modules are simply circuit boards that house one or more relays. They come in a variety of shapes and sizes, but are most commonly rectangular with 2, 4, or 8 relays mounted on them, sometimes even up to a 16 relays.

Relay modules contain other components than the relay unit. These include indicator LEDs, protection diodes, transistors, resistors, and other parts. But what is the module relay, which makes the bulk of the device? You may ask. Here are facts to note about it:

- A relay is an electrical switch that can be used to control devices and systems that use higher voltages. In the case of module relay, the mechanism is typically an electromagnet.
- The relay module input voltage is usually DC. However, the electrical load that a relay will control can be either AC or DC, but essentially within the limit levels that the relay is designed for.
- A relay module is available in an array of input voltage ratings: It can be a 3.2V or 5V relay module for low power switching, or it can be a 12 or 24V relay module for heavy-duty systems.
- The relay module information is normally printed on the surface of the device for ready reference. This includes the input voltage rating, switch voltage, and current limit.

5.7.2 Relay Module Function

What does a relay module do? The relay module function is mainly to switch electrical devices and systems on or off. It also serves to isolate the control circuit from the device or system being controlled.

This is important because it allows you the use a microcontroller or other low-power device to control devices with much higher voltages and currents.

Another relay module purpose is to amplify the control signal so that it can switch the higher currents using only a small out of power from a microcontroller.

GEYA 2NO2NC Omron Plug in Relay Module FY-2NG2R Technical Parameters：

Input operating voltage	Rated DC±20% & AC±10%
Relay Mounting	Pluggable
Rated Load Current	5A 250V AC/30V DC
Max Transient Load Current	5A
Coil Power	approx.0.53W
Operate Time(at normal volt.)	≤15ms
Release Time(at normal volt.)	≤10ms
Input/Output Connections	Screw terminal block
Wire Range	0.2～2.5mm²
Stripping Length	6-7mm
DIN Rail Mountable	15/28/35mm

Table 5: Relay Technical Parameters

5.7.3Relay Module Working

How does a relay module work? The relay module working principle is actually quite simple. It uses an electromagnet to open and close a set of electrical contacts. Here is the sequential working of relay module devices for easier understanding:

- The typical relay module connection points include an input side that consists of 3 or 4 jumper pins, and an output side that has 3 screw terminals.
- When the control signal is applied to the input side of the relay, it activates the electromagnet, which attracts an armature.
- This in turn closes the switch contacts on the output (high voltage) side, allowing electricity to flow and power the device or system that is connected to it.
- To prevent flyback voltage from damaging the relay module circuit and the input device, a diode is often placed in parallel with the electromagnet coil. This diode is known as a flyback diode. It allows current to flow in only one direction.
- When a higher level of isolation is required, an optocoupler is used. An opto-isolated relay module has the photoelectric device on the input side, which is used to control the electromagnet's switching action.

Relay modules are available with either normally open (NO) or normally closed (NC) switch configurations.

- A NO switch is open when the electromagnet is not activated, and closed when it is activated.
- An NC relay switch, on the other hand, remains closed by default, and only opens when the relay is activated.

5.7.3 Relay Module Uses

Relay modules are used in all sorts of applications, from controlling lights and motors to more complex systems such as automation processes and safety or security systems. Here are just a few examples of the relay module uses in different applications.

5.7.4 Home Automation Relay Module

Relay modules are also used in home automation systems to control lights, appliances, and other devices. The home automation relay module is often used with mains electricity. So it's mostly rated for 10A or less for maximum current, and up to 250V AC.

5.7.5 Industrial Relay Module

In industrial applications, the industrial relay module is used to control machinery, process controls, and other industrial equipment. Other relay module uses in industrial settings include lighting control and the control of alarm or security systems.

5.7.6 Automotive Relay Module

Relay modules are widely used in automotive applications. The automotive relay module controls things like headlights, turn signals, and even the starter motor. A car relay module is also found in many other automotive circuits such as those that operate remote starters or theft alarms.

5.7.7 Ardu Relay Module

Hobbyists often use a relay module with Arduino in their projects. An Arduino is a microcontroller board that is widely popular in DIY electronics projects. The relay module, when paired with an Arduino, can control various appliances and devices.

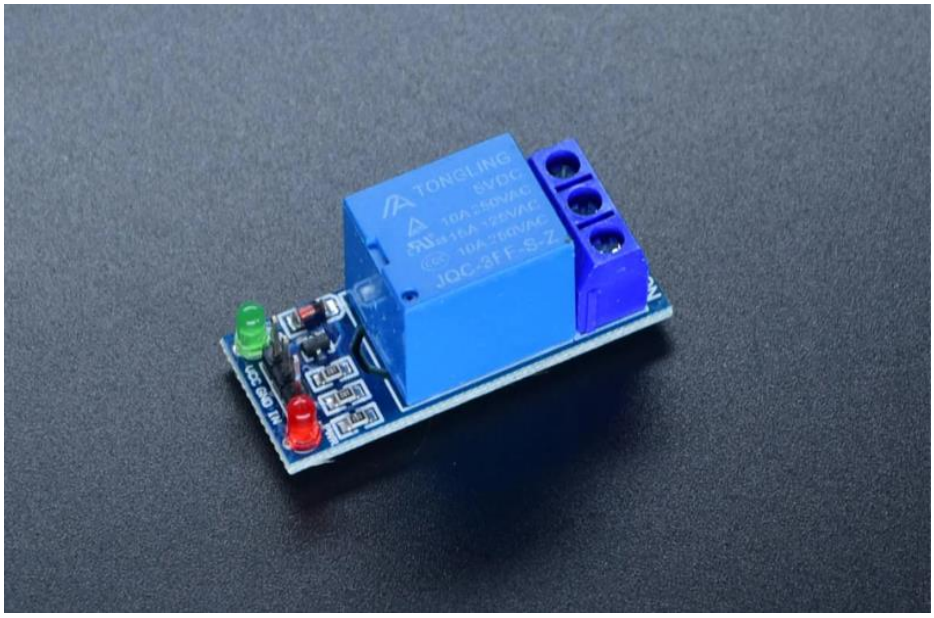


figure 86relay module

5.8 THE L298N H-BRIDGE

5.8.1 Overveiw

The L298N H-bridge is a popular integrated circuit (IC) used for controlling the speed and direction of DC motors. It is a dual full-bridge driver that can drive up to two DC motors simultaneously, and it is widely used in robotics, automation, and other applications that require precise motor control. The L298N H-bridge has a maximum input voltage of 35V and a maximum output current of 2A per channel. It is designed to work with a wide range of DC motors and can handle a variety of different motor types, including bipolar stepper motors, unipolar stepper motors, and DC brush motors. Figure 54(H-Bridge) The IC features four inputs, two for each bridge, which are used to control the direction and speed of the motors. The direction of the motors can be controlled by sending a high or low signal to the inputs, while the speed of the motors can be controlled using pulse-width modulation (PWM).

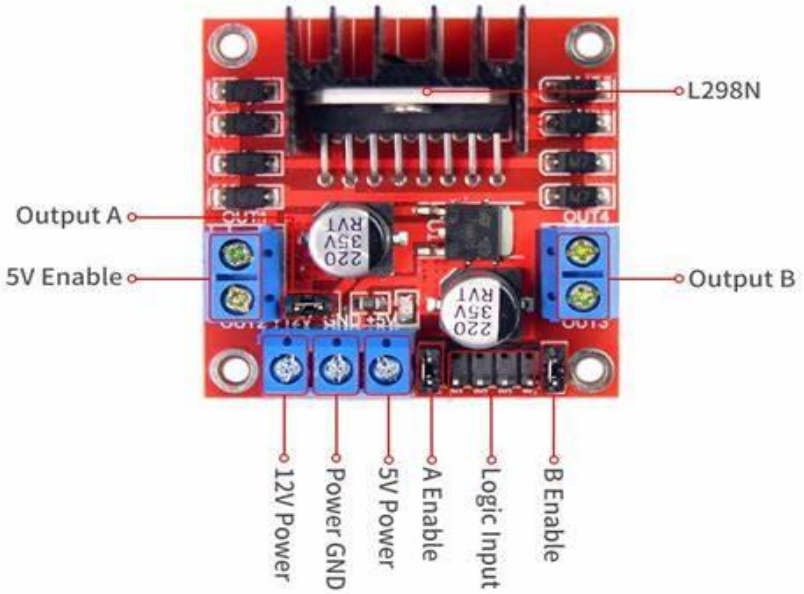


figure 87L298N H-BRIDGE

The L298N H-bridge also features built-in protection features, such as thermal shutdown and overcurrent protection, which help to prevent damage to the IC or the connected motors in the event of a fault.

5.8.2 L298N Motor Driver Module pinout

2 DC motor output pins, 12-volt external motor power supply, motor direction control pins (IN1, IN2, IN3, IN4), motor output enable pins (ENA, ENB), and a heat sink.

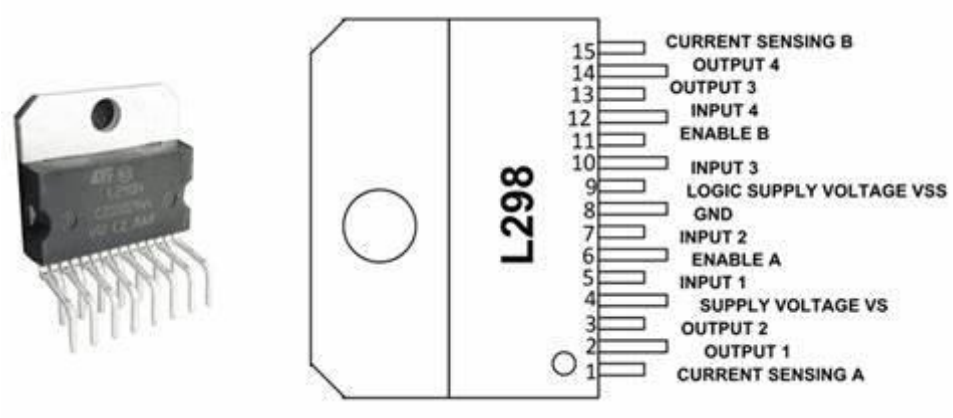


figure 88L298N Motor Driver Module pinout

VCC pin supplies power to the motor. Voltage anywhere between 5 to 35V can be applied. Remember, if the 5V-EN jumper is in place, you need to supply 2 extra volts than the motor's actual voltage requirement, to run the motor at its maximum speed. GND is the common ground pin. 5V pin supplies power to the switching logic circuitry inside the L298N IC. If the 5V-EN jumper is in place, this pin acts as output and can be used to power up the Arduino. If the 5V EN jumper is removed, you need to connect it to the 5V pin on Arduino. ENA pins are utilized to control the speed of Motor A. Supplying this pin with HIGH logic makes Motor A rotate, supplying it with LOW logic causes the motor to stop. Removing the jumper and connecting this pin to the PWM input let us control the speed of the Motor A. IN1 & IN2 pins are used to control the direction of Motor A. If IN1 is HIGH and IN2 is LOW, Motor A spins in a certain direction. To change the direction, make IN1 LOW and IN2 HIGH. If both the inputs are either HIGH or LOW, Motor A stops. IN3 & IN4 pins are used to control the direction of the Motor B. If IN3 is HIGH and IN4 is LOW, Motor B spins in a certain direction. To change the direction, make IN3 LOW and IN4 HIGH. If both the inputs are either HIGH or LOW, the Motor B stops. ENB pin can be used to control the speed of Motor B. Supplying this pin with the HIGH signal makes the Motor B turn, supplying it LOW causing the motor to stop. Eliminating the jumper and interfacing this pin to PWM

information let us control the speed of Motor B. OUT1 & OUT2 pins are connected to Motor A. OUT3 & OUT4 pins are connected to Motor B

5.9 DC MOTOR

Figure 87: dc motor

A DC motor is one of a class of rotary electrical machines that converts direct current electrical energy into mechanical energy. The most common types rely on the forces produced by magnetic fields. Nearly all types of DC motors have some internal mechanism, either electromechanical or electronic, to periodically change the direction of current flow in part of the motor. Motor's direction of rotation can be changed by reversing the Figure 56(DC-motor module) polarity of the applied voltage. On the other hand, to control speed is using a timer in PWM mode. The pulse width modulation method is very efficient and is the most used method. The PWM method drives the motor by applying a series of ON and OFF pulses. It varies the duty cycle by applying the ON or OFF time greater or lower while keeping the frequency constant. The pulse width modulation signal is a square signal which has a high frequency (more than 1 kHz). If we want to vary the power supplied, then we have to vary the duty cycle of this square wave. The duty cycle is the fraction of time in which the signal was high. The duty cycle can be varied from 0% to 100%.

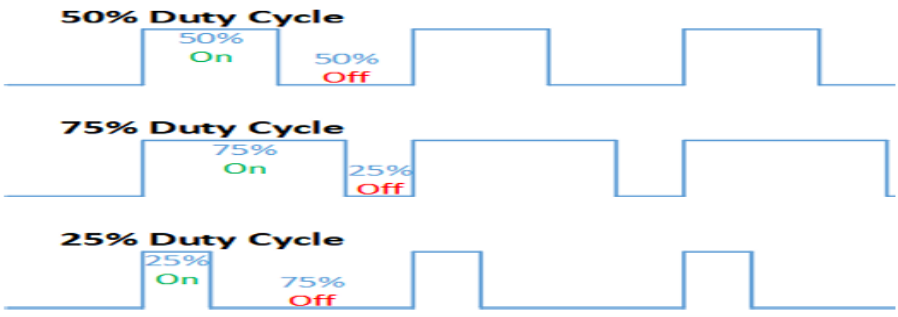


Figure 57(different kinds of duty cycle)

figure 89duty cycle

The formula for calculating the duty cycle of square wave:

%Duty Cycle = (Ton / (Ton + To)) * 100

Ton is the time in which the square wave is on.

Toff is the time in which the square wave is off.

Chapter Six

6 TRANSMISSION COMMUNICATIONS PROTOCOLS AND METHODS

6.1 SERIAL COMMUNICATIONS

This is the most commonly used in small embedded application and they are the first ones we learn when we get on the embedded system field. Used when using short-distance communication between devices, protocols such as UART, I2C, and SPI are commonly used in different applications domains.

Some characteristics which make the use of such communication type on embedded systems so popular are :

1. Simple and eficient : You usually don't need lots of wires. In a context where cost and space are limited as the embedded systems one, this communication system makes a widely popular choice for projects.

2. Versatile : It is versatile because you can connect a different variety of devices. From simple sensors, memory devices until other processors, which opens possibility to a rich varieties of devices, for complex embedded applications.

3. Error handling : in general such communications have methods which allows the software to recognize errors from different natures. One good example is the use of the parity bit from the UART protocol.

The UART (Universal Asynchronous Receiver Transmitter) communication, is one example of serial communication, commonly used for serial debugging when testing application.

The SPI (Serial Peripheral Interface) is a commonly master-slave clock driven system usually used in embedded applications. With a slightly more important dara rate, it is used in applications as audio/communication ones. It is more commnly used when fewer devices are present, due to lack of space (more the devices, more wires recquired).

The I2C (Inter Integrated circuits) is another master-slave, clock-driven communication method widely used in embedded system development. Sharing the same bus, this protocol is a good fit when you are dealing with more devices, and you don't need a higher data rate.

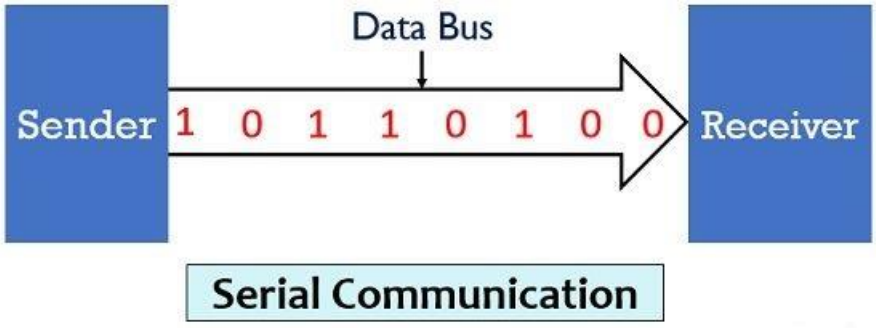


figure 90serial communication

6.2 PARALLEL COMMUNICATIONS

This communication type is not as common as the serial communication in embedded systems, firstly because it's faster, it recquires more resources than the serial communication ones.

The most common use is the AHB (Advanced High-performance Bus), widely used in applications such as the aerospace field and the industrial automation, applications where high perfomance and low latency is recquired, due to their critical-real-time nature.

It's architecure allows multiple devices to communicate simultaneously, providing a high data transfer rate. Being a multilayer bus protocol, it also supports multiple master and slave devices.

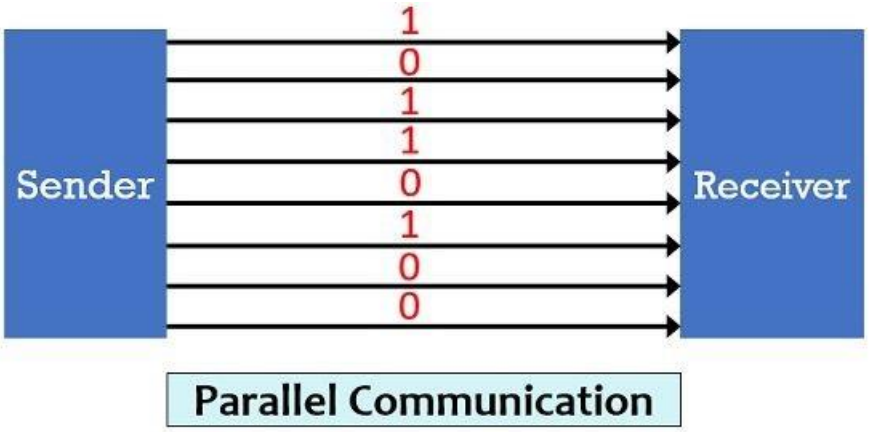


figure 91: parallrl communication

6.3 WIRELESS COMMUNICATIONS

Another use case of communication in embedded systems is the wireless communications, widely used to connect systems or connect systems to a network to manage data transfer without using wires.

It provides you flexibility, which can be really interesting if you are in an application context which recquires mobility freedom, a case where the wired communication is very limited.

It also has some limitations. The advantage of providing more freedom and mobility, it comes with a tradeoff : lack of security. In comparison to wired communications, wireless ones can be less secure in data transfer.

Some common wireless communication are : Bluetooth Low Energy (BLE), Wi-Fi and Zigbee.

The BLE is a wireless communication making use of UHV (Ultra High Waves) radio waves for exchanging data in short distances on a PAN (Personal Area Network), having some application target such as smart home and IoT. It’s designed for devices that need to communicate frequently but with low data rates.

Zigbee is a high level, wireless mesh networking technology. It allows devices on a PAN to connect each other with small, low power and low cost technology, which fits very well for applications as smart home devices, because it supports a large number of devices.

The Wi-fi protocol, also known by the IEEE standard as 802.11, is the most daily used communication protocol. It's the main bridge between you and the internet. Probably it's because you are using Wi-fi that you are able to read this article. It allows for high-speed data transfer and can connect devices over a wide range, making it suitable for applications such as streaming video, online gaming, and file transfers.

6.4 USART COMMUNICATION PROTOCOL

6.4.1 Overview of UART (Universal Asynchronous Receiver-Transmitter)

UART is a computer hardware device used for asynchronous serial communication. It enables the transmission of data between devices without the need for a clock signal. Here are some key aspects of UART:

6.4.2 Key Components:

1. Transmitter (TX): Converts parallel data from the CPU into a serial format.
2. Receiver (RX): Converts serial data back into parallel format for the CPU.
3. Baud Rate Generator: Determines the rate at which bits are sent or received.
4. Buffers: Temporary storage areas for data waiting to be transmitted or received.

6.4.3 Operation:

- Asynchronous Communication: No clock signal is shared between sender and receiver. Both must agree on the baud rate beforehand.
- Start and Stop Bits: Indicate the beginning and end of a data packet.
- Parity Bit (optional): Used for error checking.

6.4.4 Data Frame Format:

- Start Bit: Typically 1 bit (low level).
- Data Bits: 5 to 9 bits (commonly 8).
- Parity Bit: Optional for error checking (odd, even, or none).
- Stop Bits: 1, 1.5, or 2 bits (high level).

6.4.5 Advantages of UART:

- Simplicity: Easy to set up and use.
- Cost-effective: Commonly found in many microcontrollers and peripheral devices.
- No Clock Signal: Reduces complexity in communication.

6.4.6 Disadvantages of UART:

- Limited Speed: Generally slower than synchronous communication methods.
- Short Distance: Not suitable for long-distance communication without additional hardware.
- No Multi-Drop Capability: Typically supports only point-to-point communication.

6.4.7 Common Uses:

- Microcontroller Communication: Interfacing microcontrollers with sensors, displays, or other peripherals.
- Serial Ports: Traditional RS-232 serial ports use UART.
- Embedded Systems: Widely used in various embedded systems for device communication.

Chapter Seven

7 HOW WE BUILT SOFTWARE

7.1 STM CUBE IDE

7.1.1 Description

STM32CubeIDE is an all-in-one multi-OS development tool, which is part of the STM32Cube software ecosystem.

STM32CubeIDE is an advanced C/C++ development platform with peripheral configuration, code generation, code compilation, and debug features for STM32 microcontrollers and microprocessors. It is based on the Eclipse®/CDT™ framework and GCC toolchain for the development, and GDB for the debugging. It allows the integration of the hundreds of existing plugins that complete the features of the Eclipse IDE.
STM32CubeIDE integrates STM32 configuration and project creation functionalities from STM32CubeMX to offer all-in-one tool experience and save installation and development time. After the selection of an empty STM32 MCU or MPU, or preconfigured microcontroller or microprocessor from the selection of a board or the selection of an example, the project is created and initialization code generated. At any time during the development, the user can return to the initialization and configuration of the peripherals or middleware and regenerate the initialization code with no impact on the user code.
STM32CubeIDE includes build and stack analyzers that provide the user with useful information about project status and memory requirements.
STM32CubeIDE also includes standard and advanced debugging features including views of CPU core registers, memories, and peripheral registers, as well as live variable watch, Serial Wire Viewer interface, or fault analyzer.

7.1.2 All features

Integration of services from STM32CubeMX:STM32 microcontroller, microprocessor, development platform and example project selection Pinout, clock, peripheral, and middleware configuration Project creation and generation of the initialization code Software and middleware completed with enhanced STM32Cube Expansion Packages Based on Eclipse®/CDT™, with support for Eclipse® add-ons, GNU C/C++ for Arm® toolchain and GDB debugger
STM32MP1 Series: Support for Open ST Linux projects: Linux Support for Linux Additional advanced debug features including: CPU core, peripheral register, and memory views Live variable watch view System analysis and real-time tracing (SWV)CPU fault analysis tool RTOS-aware debug support including Azure Support for ST-LINK (STMicroelectronics) and J-Link (SEGGER) debug probes Import project from Atollic True STUDIO® and AC6 System Workbench for STM32 (SW4STM32)
Multi-OS support: Windows®, Linux®, and macOS, 64-bit versions only

7.2 AUTOSAR layered architecture

The AUTOSAR (AUTomotive Open System ARchitecture) layered architecture is a standardized software architecture for automotive electronic control units (ECUs). Here's a description of the key layers and components in the AUTOSAR layered architecture:

1- Microcontroller Abstraction Layer (MCAL): This layer abstracts the microcontroller-specific functionalities. It provides standard interfaces to the higher software layers, facilitating portability of the software across different microcontrollers.

2- Hardware Abstraction Layer (HAL): HAL abstracts the hardware-specific components of the ECU. It enables the software to be independent of the underlying hardware, thus enhancing portability and reusability.

3- Service Layer: This layer provides basic services for the application software and other software layers. It includes services like communication, diagnostic, memory, and operating system services.

4- Application Layer: This layer contains the application software components that implement the functionality required by the specific ECU.

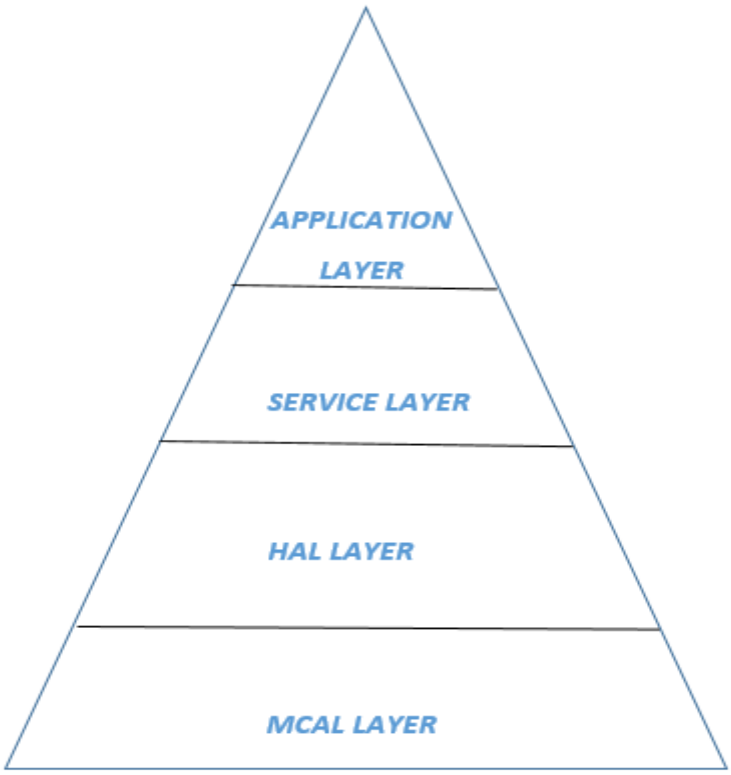


figure 92AUTOSAR layered architecture

7.3 Developed drivers

We have developed all needed drivers from scratch by our self, Then we used these drivers to develop our project.

MCAL layer:

- **RCC**
- **GPIO**
- **AFIO**
- **EXTI**
- **USART**
- **SPI**
- **I2C**
- **GPTIM**
- **DMA**
- **ADC**

HAL Layer:

- **7SEG**
- **KPD**
- **LCD**
- **IR**
- **LEDMRX**
- **STP**
- **DAC**
- **DAC**
- **TFT**
- **rgb565**
- **MPU6050**
- **RTC**
- **ULTRASO**
- **INA219**

SERVICE layer:

- **TIMERS**
- **CARKIT**
- **GUI**
- **V2G**

7 CONCLUSION AND FUTURE WORK

7.1 OVERVIEW

In this chapter, we will summarize the hole of project , In addition discussing the future works that will improve the availability, performance, security features.

7.2 CONCLUSION

In conclusion, the proposed system that utilizes V2X communication technology to remotely
stop stolen cars shows promise in addressing the issue of car theft and preventing stolen cars
from causing harm on the streets. By avoiding high-speed chases and other dangerous situations, this system can provide a safer and more efficient approach to stopping stolen cars.
While there are some challenges and limitations, such as the need for widespread adoption of
V2X technology and potential security risks, future developments could address these concerns and improve the effectiveness of the system. Additionally, integrating AI technology could further enhance the system's capabilities and improve its ability to predict
and prevent car theft. Overall, the proposed system has the potential to make a significant
impact on public safety and law enforcement efforts.

7.3 BENEFITS

1- Increased safety: The use of V2X communication technology and the ability to remotely stop a stolen car can prevent dangerous situations from arising, such as high-speed chases, which can put the lives of police officers, the driver of the car, and innocent bystanders at risk.

2- More efficient and effective: The conventional methods of dealing with car theft, such as police chases or car owners attempting to track their stolen vehicles, can be time-consuming and sometimes ineffective. The use of V2X communication technology can help law enforcement stop a stolen car quickly and prevent it from causing further harm.

3- Enhanced security: The system's use of RSA encryption helps ensure that the signal sent to stop the stolen car is secure, reducing the risk of the signal being intercepted or tampered with by unauthorized parties.

4- Concealed and embedded: The system being concealed and embedded in the car can make it difficult for thieves to detect and disable the system, increasing the chances of stopping the car and recovering it safely.

5- Accuracy: The proposed system can send targeted commands to specific cars, which can increase its accuracy and reduce the chances of unintended consequences. By using a unique car ID to send the stop command, the system can ensure that only the intended car is affected, even in situations where multiple cars are present.

Overall, the proposed V2X communication system provides a safer and more efficient approach to stop stolen cars and prevent them from causing harm on the streets.

7.4 CHALLENGES AND LIMITATIONS

1- Implementation and integration: Developing and integrating the necessary hardware

and software components for the system could be a complex and time-consuming process.

2- Cost: Depending on the complexity and scope of the system, it could require a significant investment in resources and infrastructure, which may be a barrier for some law enforcement agencies or municipalities.

3- False positives or negatives: Depending on how the system is designed, there is a possibility of false positives or false negatives, where the system may incorrectly identify a car as stolen or fail to identify a stolen car, respectively. This could potentially result in unnecessary stops or missed opportunities to stop stolen vehicles.

4- Security: While the use of encryption is a step towards ensuring security, there is still the potential for the system to be hacked or compromised by malicious actors, which could have serious consequences.

5- Privacy concerns: The use of V2X technology raises questions about data privacy and how the system will handle and protect sensitive information, such as the location and movements of vehicles. It will be important to establish clear protocols and safeguards to protect individual privacy and prevent misuse of the system

7.5 FUTURE DEVELOPMENT

Incorporating artificial intelligence (AI) and machine learning (ML) algorithms to improve the accuracy of identifying stolen cars and reducing false positives.
Developing a mobile application for police officers to have more control over the system and allow for easier communication and tracking.
Implementing real-time tracking of stolen cars to provide more accurate location data and better monitoring of the situation.
Integrating with other technologies such as autonomous vehicles to enable remote control of the car in certain situations.
Expanding the use of V2X technology beyond just stolen cars, such as for emergency response vehicles to clear the way during urgent situations.
AI can be used in several ways to enhance the functionality of this app. Here are some potential ways:

1- Automatic license plate recognition (ALPR): By using computer vision and deep learning algorithms, the app can detect and recognize license plates of the cars around, which can help identify stolen cars and alert law enforcement.
2- Predictive analytics: By analyzing historical data on car theft and police chase incidents, the app can use machine learning algorithms to predict the likelihood of a car being stolen or involved in a police chase. This can help law enforcement take proactive measures to prevent such incidents.
3- Natural language processing (NLP): The app can be equipped with NLP capabilities to process voice commands from the police officers and respond accordingly, such as sending stopping commands to the stolen car.
4- Automated decision-making: The app can be designed to make automated decisions based on real-time data, such as whether to send a stopping command to a particular car based on its speed, location, and other factors.
5- Autonomous driving: In the future, the app can integrate with autonomous driving technologies to remotely take control of a stolen car and bring it to a stop in a safe manner, without risking the lives of police officers or bystanders.

8 REFERENCE

[1] Shanzhi Chen , Li Zhao , Rui Zhao Jinling Hu and Jiayi FangLi . Cellular Vehicle-to-Everything (C-V2X)

[2] Fei Hu, Vehicle-to-Vehicle and Vehicle-to-Infrastructure Communications

[3] Schoettle, B., & Sivak, M. (2018). Enabling Safe and Efficient Connected and Automated Vehicles: Emerging Research and Insights. University of Michigan Transportation Research Institute.

[4] Zhang, W., & Liu, Y. (2021). Vehicle-to-Everything (V2X) Communication: Emerging Trends and Practical Applications. Springer.Atmel Corporation. "ATMEGA32 Datasheet." Atmel Corporation, 2013.

[5] Chisalita, I., & Shahmehri, N. (2002). A peer-to-peer approach to vehicular communication for the support of traffic safety applications. Proceedings of the IEEE Conference on Intelligent Transportation Systems, 336-341.

[6] Ali, F., Saad, W., & Bennis, M. (2018). Vehicle-to-Pedestrian (V2P) Communication in Cellular Networks: Stochastic Geometry Modeling and Analysis. IEEE Transactions on Wireless Communications, 17(8), 5221-5236.

[7] Feng, L., Qin, Z., Li, W., Wang, Y., & Zhao, W. (2018). V2P communication in smart cities: A mobile pedestrian safety system for low-speed urban areas. Sensors.

[8] Valvano, J. W. (2016). Embedded Systems: Introduction to Arm Cortex-M Microcontrollers (Volume 1). CreateSpace Independent Publishing Platform.

[9] Paz, I., & Barzilai, G. (2020). Hands-On Embedded Programming with STM32: Build Industry-Grade Embedded Applications with the Powerful STM32F3 Series of Microcontrollers. Packt Publishing.

[10] Pham, L. (2020). Programming with STM32: Getting Started with the Nucleo Board and C/C++. CreateSpace Independent Publishing Platform.

[11] Paz, I., & Barzilai, G. (2020). Hands-On Embedded Programming with STM32: Build Industry-Grade Embedded Applications with the Powerful STM32F3 Series of Microcontrollers. Packt Publishing.

[12] Rauch, L. (2020). Programming ESP32 in MicroPython: A Practical Guide to MicroPython for ESP32. Independently Published.

[13] Barkakati, N. (2018). ESP32 Programming for the Internet of Things: Using the Arduino IDE. CreateSpace Independent Publishing Platform.

[14] Android Developer Documentation: https://developer.android.com/docs

[15] Firebase Documentation: https://firebase.google.com/docs

[16] GPS,Location Services: https://developer.android.com/training/locatio

[17] Sensor API: https://developer.android.com/guide/topics/sensors