

# **INTELLIGENT COLLISION AVOIDANCE FOR RAILWAYS USING LORAWAN TECHNOLOGY**

**A PROJECT REPORT**

*Submitted by*

<b>SUDHARSAN S</b>	<b>110521106026</b>
<b>SATHISH S</b>	<b>110521106022</b>
<b>DAKSHINA MOORTHY N</b>	<b>110521106305</b>

*In partial fulfillment for the award of the degree  
of*

**BACHELOR OF ENGINEERING**

**IN**

**ELECTRONICS AND COMMUNICATION**

**ENGINEERING**

**GOJAN SCHOOL OF BUSINESS AND TECHNOLOGY  
EDAPALAYAM, REDHILLS, CH-52**

**ANNA UNIVERSITY:: CHENNAI 600 025**

**APRIL/MAY 2025**

## **BONAFIDE CERTIFICATE**

Certified that this project report “**INTELLIGENT COLLISION AVOIDANCE FOR RAILWAYS USING LORAWAN TECHNOLOGY**” is the Bonafide work of **SUDHARSAN S(110521106026), SATHISH S(110521106022), DAKSHINA MOORTHY N(110521106305)**, who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported here is not a part of any other project work.

**SIGNATURE**

**Mrs.S.KOKILA, B.E,  
M.TECH.,  
HEAD OF DEPARTMENT  
ASSISTANT PROFESSOR  
Department of Electronics and  
Communication Engineering  
Gojan School of Business and  
Technology  
80 Ft Road, Edapalayam,  
Redhills, Ch-52**

**SIGNATURE**

**Mrs.V.SARANYA, M.TECH.,  
ASSISTANT PROFESSOR  
  
Department of Electronics and  
Communication Engineering  
Gojan School of Business and  
Technology  
80 Ft Road, Edapalayam,  
Redhills, Ch-52**

Submitted for the Project Viva Voce Examination held on \_\_\_\_\_

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

**SUDHARSAN S**

Department of Electronics and Communication Engineering

Gojan School of Business and Technology

### **DECLARATION**

I hereby declare that the project entitled **“INTELLIGENT COLLISION AVOIDANCE FOR RAILWAYS USING LORAWAN TECHNOLOGY”** submitted by us to Anna University in May/June 2025 for the award of degree of Bachelor of Engineering in Aeronautical Engineering is the original and independent team work carried out by me under the guidance of **Mrs.V.SARANYA, M.TECH.**, Assistant professor, and that it has not been formed the basis fully or partially for the award of any degree, diploma or other similar titles earlier and no part of above project work has been published or sent for the publication at the time of submission.

Place: - Chennai

Date: -

Signature

**SATHISH S**

Department of Electronics and Communication Engineering

Gojan School of Business and Technology

### **DECLARATION**

I hereby declare that the project entitled **“INTELLIGENT COLLISION AVOIDANCE FOR RAILWAYS USING LORAWAN TECHNOLOGY”** submitted by us to Anna University in May/June 2025 for the award of degree of Bachelor of Engineering in Aeronautical Engineering is the original and independent team work carried out by me under the guidance of **Mrs.V.SARANYA, M.TECH.,** Assistant professor, and that it has not been formed the basis fully or partially for the award of any degree, diploma or other similar titles earlier and no part of above project work has been published or sent for the publication at the time of submission.

Place: - Chennai

Date: -

Signature

**DAKSHINA MOORTHY N**

Department of Electronics and Communication Engineering

Gojan School of Business and Technology

### **DECLARATION**

I hereby declare that the project entitled **“INTELLIGENT COLLISION AVOIDANCE FOR RAILWAYS USING LORAWAN TECHNOLOGY”** submitted us to Anna University in May/June 2025 for the award of degree of Bachelor of Engineering in Aeronautical Engineering is the original and independent team work carried out by me under the guidance of **Mrs.V.SARANYA, M.TECH.**, Assistant professor, and that it has not been formed the basis fully or partially for the award of any degree, diploma or other similar titles earlier and no part of above project work has been published or sent for the publication at the time of submission.

Place: - Chennai

Date: -

Signature

## ACKNOWLEDGEMENT

We express our deepest gratitude to our **Chairman Dr. G. Natarajan, Ph.D.**, and **Chairperson Mrs. Brindha Natarajan, B. Com**, for their valuable guidance and blessings.

We are deeply indebted to our beloved **Principal Dr. C. Selvakumar, Ph.D.**, Gojan School of Business and Technology, for providing us an excellent environment to carry out our course successfully.

We express our sincere thanks to our **Head of the Department Mrs.S.KOKILA, B.E, M.TECH.**, Assistant Professor, who has been a constant source of inspiration and guidance in the course of the project.

We record our thanks to our **Supervisor Mrs.V.SARANYA, M.TECH.**, Assistant Professor, for being instrumental in the completion of our project with his exemplary guidance.

We thank all the **Staff Members** of our department for their valuable support and assistance at various stages of our project development.

Finally, we take this opportunity to extend our deep sense of gratitude and appreciation to our family and friends for all that they meant to us during the crucial times of the completion of our project.

## **ABSTRACT**

This project proposes an intelligent and cost-effective collision avoidance system for railway applications using LoRaWAN (Long Range Wide Area Network) technology. With the increasing need for safety in rail transport, especially in rural and single-track areas, this system enables real-time train-to-train communication using GPS and LoRa modules. The system ensures that two locomotives on a potential collision course can detect each other's presence early, communicate their location, and trigger alerts or actions to prevent accidents. Each train continuously transmits its location data via LoRa to nearby trains and a central monitoring station. An algorithm on the microcontroller calculates the risk of collision based on train speeds, directions, and proximity, issuing alerts or automatically initiating braking if necessary. This system enhances safety in rail transport, especially in rural and unmanned tracks, with minimal dependence on centralized infrastructure.

## TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	<b>ABSTRACT</b>	<b>i</b>
	<b>LIST OF FIGURES</b>	<b>iv</b>
	<b>LIST OF ABBREVIATION</b>	<b>vi</b>
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
	1.1 SCOPE OF THE PROJECT	2
	1.2 OBJECTIVE OF THE PROJECT	2
	1.3 PROBLEM STATEMENT	3
	1.4 SOLUTION OVERVIEW	3
	1.5 METHODOLOGY	4
	1.5.1 SYSTEM OVERVIEW	4
	1.5.2 TECHNIQUE TO OVERCOME EXISTING SYSTEM PROBLEM	4
	1.5.3 ROLE OF LORAWAN TECHNOLOGY	4
	1.5.4 IMPLEMENTATION STEPS	6
	1.6 ORGANIZATION OF THE REPORT	8
<b>2</b>	<b>LITERATURE SURVEY</b>	<b>9</b>
	2.1 INTRODUCTION	9
	2.1.1 RAILWAY ACCIDENT PREVENTION AND SAFEGUARDING THE TRACKS USING LORA TECHNOLOGY	9
	2.1.2 LORA BASED RAILWAY SIGNALLING SYSTEM	10
	2.1.3 SMART TRACK CONTINUOUS MONITORING AND TRAIN COLLISION AVOIDANCE SYSTEM USING IOT	11
	2.1.4 AUTOMATIC RAILWAY LEVEL CROSSING USING LORA	11
	2.1.5 AUTOMATIC RAILWAY CRACK DETECTOR	12



## SYSTEM USING LORA NETWORK

3	<b>EXISTING SYSTEM</b>	13
	3.1 AUTOMATIC RAILWAY LEVEL CROSSING	13
	3.2 RAILWAY TRACK CRACK DETECTION SYSTEM	14
4	<b>PROPOSED DESIGN</b>	16
	4.1 INTRODUCTION	16
	4.2 BLOCK DIAGRAM OF PROPOSED SYSTEM	17
	4.3 CIRCUIT DIAGRAM	20
	4.4 METHODOLOGY	21
5	<b>SYSTEM SPECIFICATION</b>	22
	5.1 HARDWARE COMPONENTS	22
	5.1.1 RASPBERRY PI PICO Microcontroller	22
	5.1.2 TRANSEIVER	28
	5.1.3 NEO 6M GPS	30
	5.1.4 OLED DISPLAY	32
	5.1.5 MOTOR AND MOTOR DRIVER	34
	5.2 SOFTWARE REQUIREMENT	37
	5.2.1 Arduino software (IDE)	37
	5.2.2 Steps to install the arduino software	37
	5.2.3 Programming ESP32 board with arduino IDE	41
	5.2.4 Hardware Information of Raspberry Pi pico	44
	5.3 CLOUD INTEGRATION	46
	5.4 EMBEDDED C++	49
6	<b>IMPLEMENTATION AND RESULTS</b>	51
	6.1 IMPLEMENTATION	51
	6.1.1 Hardware Integration	51
	6.2 SYSTEM OPERATION	52
	6.3 RESULTS AND OBSERVATION	53
7	<b>CONCLUSION ANDE FUTURE SCOPE</b>	53
	7.1 CONCLUTION	56
	7.2 FUTURE SCOPE	57
	REFERENCE	58

## **LIST OF FIGURES**

<b>FIGURE NO</b>	<b>NAME OF THE FIGURES</b>	<b>PAGE NO</b>
1.1	LORAWAN TECHNOLOGY	5
1.2	FEATURES OF LORAWAN TECHNOLOGY	6
3.1	SYSTEM ARCHITECTURE OF EXISTING SYSTEM	13
3.2	EXISTING SYSTEM SIMULATED DIAGRAM	15
4.1	BLOCK DIAGRAM OF PROPOSED SYSTEM(LM-1)	17
4.2	BLOCK DIAGRAM OF PROPOSED SYSTEM(LM-2)	18
4.3	CIRCUIT DIAGRAM OF PROPOSED SYSTEM	20
5.1	RASPBERRY PI PICO	23
5.2	RASPBERRY PI PICO W PINOUT DIAGRAM	25
5.3	LORA 02 SX1278	28
5.4	NEO 6M GPS MODULE	30
5.5	OLED DISPLAY	32
5.6	L293D MOTOR DRIVER	34
5.7	DC MOTOR	35
5.8	BUZZER	36
5.9	THE ARDUINO SOFTWARE IDE	38
5.10	SELECTING A BOARD & PORT	39
5.11	OPENING IN EXAMPLE	39
5.12	BLINKING LED IN ARDUINO BOARD	41

5.13	INSTALL GITHUB LINK IN ARDUINO IDE	42
5.14	INSTALL PICO BOARD	43
5.15	RASPBERRY PI PICO SPECS	45
5.16	CREATE THE TEMPLATE IN BLYNK PLATFORM	47
5.17	INCLUDE YOUR DATASTREAM DATA AND SAVE	47
6.1	HARDWARE OF PROPOSED SYSTEM (LM-1)	51
6.2	SYSTEM OUTPUT	53
6.3	OLED DISPLAY DATA OUTPUT	54
6.4	BLYNK DASHBOARD OUTPUT	55

## **LIST OF ABBREVIATIONS**

<b>S. NO</b>	<b>ABBREVIATION</b>	<b>EXPANSION</b>
1	LM-1	Loco Motive - 1
2	IDE	Integrated Development Environment
3	OTA	Over The Air
4	LORAWAN	Long Range Wide Area Network
5	PWM	Pulse Width Modulation
6	AI	Artificial Intelligence
7	ID	Identification
8	GPS	Global Positioning System
9	OLED	Organic Light Emitting Diode
10	UART	Universal Asynchronous Receiver Transmitter
11	ISM	Industrial Scientific and Medical
12	P2P	Peer to Peer
13	RF	Radio Frequency

# **CHAPTER 1**

## **INTRODUCTION**

The Railway transportation plays a crucial role in the movement of people and goods across vast distances. However, despite its many advantages, railway systems are vulnerable to accidents caused by human error, track obstructions, and communication delays. Collisions between trains or with obstacles on the track can lead to devastating consequences, including loss of life, property damage, and service disruptions. Therefore, there is a pressing need for intelligent systems that can proactively prevent such incidents by enhancing situational awareness and enabling timely decision-making.

This project, “An Intelligent Collision Avoidance for Railways using LoRaWAN Technology”, aims to design and implement a smart, real-time solution for collision detection and avoidance in railway networks. Leveraging the long-range, low-power communication capabilities of LoRaWAN, the system enables continuous monitoring and data exchange between trains and control stations. The integration of GPS modules allows accurate real-time tracking of train positions, while additional components such as buzzers, OLED displays, and motor drivers provide timely alerts and control actions to prevent collisions.

The proposed solution emphasizes affordability, scalability, and efficiency, making it suitable for implementation in both urban and rural railway infrastructures. By combining embedded system design with modern IoT communication protocols, this project contributes to the advancement of intelligent railway safety systems aimed at reducing accidents and enhancing operational reliability

## **1.1 SCOPE OF THE PROJECT**

The scope of this project involves the development of an intelligent railway collision avoidance system using LoRaWAN technology. It focuses on real-time monitoring of train positions through GPS and enables long-range, low-power communication between trains and control units via LoRa modules. The system is designed to detect potential collisions and issue timely alerts using buzzers and display units. This project also includes the integration of hardware components such as Raspberry Pi Pico W, LoRa SX1278, and NEO-6M GPS modules to build a functional prototype. The solution is scalable, cost-effective, and adaptable for deployment across diverse railway networks, aiming to enhance safety and operational efficiency.

## **1.2 OBJECTIVE OF THE PROJECT**

The objective of this project is to design and implement a smart railway collision avoidance system using LoRaWAN technology. It aims to monitor train locations in real-time, detect potential collision risks, and provide timely alerts to prevent accidents. The system focuses on low-cost, long-range communication and efficient integration of GPS and alert modules for enhanced railway safety.

### **1.3 PROBLEM STATEMENT**

Railway accidents, particularly collisions, remain a significant concern due to factors such as limited real-time communication, human error, and inadequate monitoring systems. In many regions, especially in rural or less-developed areas, the lack of intelligent infrastructure and long-range communication makes it difficult to track train positions and prevent potential collisions in a timely manner. Existing systems are often expensive, complex, or unreliable, leading to increased risk to life, property, and operational efficiency..

### **1.4 SOLUTION OVERVIEW**

This project proposes an intelligent, cost-effective collision avoidance system for railways using LoRaWAN technology. By integrating GPS modules for real-time location tracking and LoRa modules for long-range wireless communication, the system enables trains to share their positions with each other and with a central unit. The system detects possible collision scenarios and triggers alerts through buzzers and OLED displays. Built around the Raspberry Pi Pico W and other low-power components, the solution offers a scalable and efficient approach to improving railway safety, especially in areas lacking advanced monitoring infrastructure.

## **1.5 METHODOLOGY**

### **1.5.1 SYSTEM OVERVIEW**

The system consists of multiple train units and a base station. Each train is equipped with GPS to track its location and a LoRa transceiver for wireless communication. The base station monitors train positions and alerts involved units if a potential collision is detected.

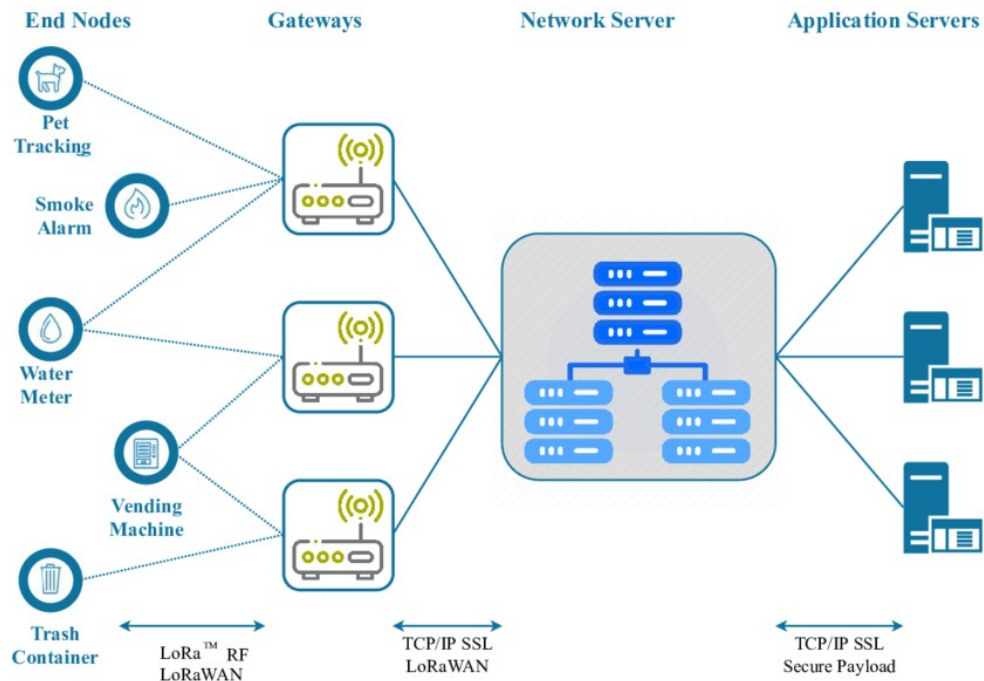
### **1.5.2 TECHNIQUE TO OVERCOME EXISTING SYSTEM PROBLEM**

Instead of relying on static sensors or centralized control rooms, each train in the proposed system functions as an independent node capable of continuously monitoring its location using a GPS module. This location data is transmitted to other nearby trains or to a base station using LoRa (SX1278) modules, which offer long-range and low-power wireless communication. The data packet includes essential details such as train ID, GPS coordinates, direction, and speed.

### **1.5.3 ROLE OF LORAWAN TECHNOLOGY**

In the project titled “Intelligent Collision Avoidance for Railways using LoRaWAN Technology,” LoRaWAN (Long Range Wide Area Network) plays a pivotal role in enabling reliable, long-range, and low-power wireless communication between moving trains and control stations. It serves as the backbone of the communication network that ensures real-time exchange of critical data such as train location, speed, direction, and identity.

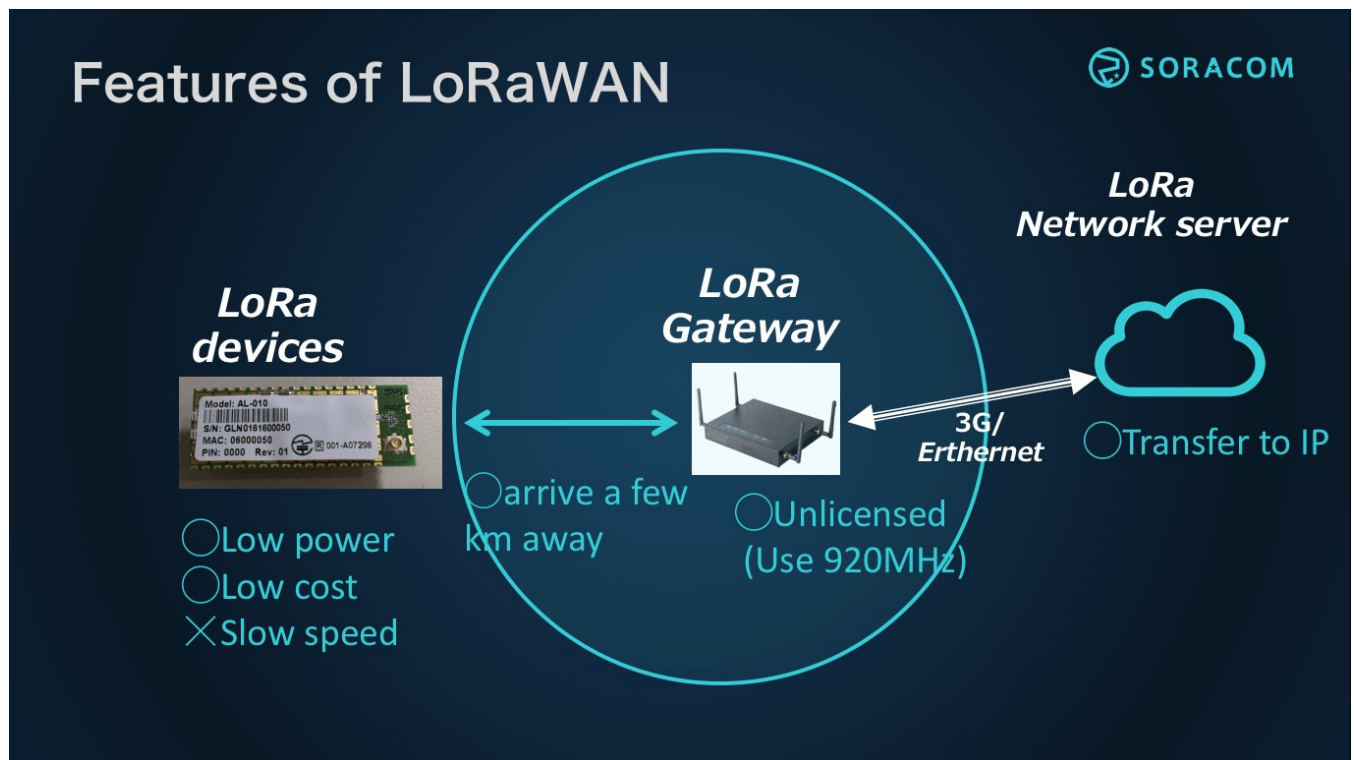




**FIGURE 1.1 LORAWAN ARCHITECTURE**

LoRaWAN can also transmit data to a LoRa gateway connected to the internet, enabling cloud-based monitoring. This opens up possibilities for real-time dashboard visualization, centralized logging, and automated decision-making from a remote control room.

Each train in the system is equipped with a Raspberry Pi Pico W, a NEO-6M GPS module, and an AI Thinker LoRa SX1278 module. The GPS module provides real-time location data (latitude and longitude), which is then processed by the microcontroller. This location data, along with the train's ID and direction of motion, is packaged into a data packet and transmitted wirelessly via the SX1278 module.



**FIGURE 1.2 FEATURES OF LORAWAN TECHNOLOGY**

### 1.5.4 IMPLEMENTATION STEPS

#### Step 1: Node Design and Configuration

Each train unit is designed as an independent LoRa node.

The GPS module continuously provides latitude and longitude.

Data packets with train ID, coordinates, and speed are formed.

These packets are transmitted using the LoRa module.

#### Step 2: Wireless Communication Setup

LoRa modules are configured with the same frequency and spreading factor for consistent communication.

Each node periodically sends and receives location data from nearby nodes or a central base station.

### Step 3: Collision Detection

The received data is compared with the train's own GPS coordinates.

If the system detects:

- Same track ID

- Opposite or same direction

- Distance less than threshold

then a collision warning is triggered.

### Step 4: Alert Generation

On detecting potential collisions, the system:

- Activates a buzzer alarm.

- Displays the warning on the OLED screen.

- Sends the same alert data to the base station (if implemented).

### Step 5: Dashboard and Remote Monitoring (Optional Enhancement)

Data is sent from LoRa to the cloud via a gateway (or via Wi-Fi from Raspberry Pi Pico W).

A web/cloud dashboard Blynk is used to visualize train locations in real time.

## 1.6 ORGANIZATION OF THE REPORT

This thesis is organized into 7 chapters, describing each part of the project with detailed illustrations and system design diagrams. The chapters are as follows:

**Chapter 1** This chapter discusses about the problem statement , an overview about the solution architecture , Methodology with technology explanation and also explains the steps of implementation of our project.

**Chapter 2** This chapter discusses about the information and details of literature survey. This portion represents the problems with solution of the railway accident prevention methods.

**Chapter 3** This chapter discusses about the existing model of Railway collision prevention models that used to implement for improvement of our railway system.

**Chapter 4** This chapter discusses about the implementation of smart railway collision prevention idea and methodologies used in the proposed system. The block diagram and circuit diagram of proposed system is explained.

**Chapter 5** This chapter discusses about the software and hardware requirements of the project i.e, Hardware components used and software used for coding purpose.

**Chapter 6** This chapter discusses about the implementation of the described design and the result obtained by the designed system.

**Chapter 7** This chapter discusses about the conclusion and future scope or idea of our project.

## **CHAPTER 2**

### **LITERATURE SURVEY**

#### **2.1 INTRODUCTION**

The literature survey for the "Intelligent Collision Avoidance for Railways using LoRaWAN Technology" project explores existing research and developments in Railway collision avoidance. It has been a critical area of research due to increasing safety concerns in transportation systems. Traditional systems depend on centralized signaling and human supervision, which often lead to delayed responses and accidents. Recent studies have explored GPS and GSM-based methods, but these face challenges in remote areas. LoRaWAN has emerged as a reliable, low-power alternative for long-range communication in intelligent railway systems.

##### **2.1.1 RAILWAY ACCIDENT PREVENTION AND SAFEGUARDING THE TRACKS USING LORA TECHNOLOGY**

-Pratheeba , Gokulapriya , Mythili , Sindhuvarshni (2024)

In this modern world, railway transportation plays an indispensable role in societal connectivity. Indian railway uses various technologies for transportation schedules and safety factors. Even though there are a lot of accidents and loss of resources. Railway track damages, obstacles, and signal issues are the major reasons for train crashes. After a train accident, significant impact occurs on people, the train's interior, exterior, and the tracks, the recovery process is prolonged and costly. Those unexpected accidents can be prevented by train safeguarding devices using LoRa technology. This technology is used for real-time wide-range communication without any signal issues. This device detects

the obstacle inside the track using an Ultrasonic sensor and detects the crack that appears on the train track using an Infrared sensor. If any obstacles or cracks are present inside the track this system will alert us and the train stops automatically when an accident possibility occurs. In this device, the LoRa transmitter sends the real-time status of the train to the operating station via the Internet of Things. In addition, to avoid accidents in barrier-lifting areas which are connected to a LoRa receiver is used for awareness of the status of the train and automatic operation of the barrier-lifting gate.

### **2.1.2 LORA BASED RAILWAY SIGNALLING SYSTEM**

*-M. Vigneshkumar ,V. Dhanraj ,S. Shrinath ,M. Kesavakumar, G. Sabari Karthik (2024)*

Internet of Things an emerging technology that has great potential to be applied in critical environments. In that regard, IoT is a remarkable solution to the challenge of collecting data from physical environments, flexibility, low cost, and ease of deployment. It has always been a dependent factor for early alert signals for the possibility of unforeseen technical issues that may occur within the mass transport networks. Our proposed infrastructure, IoT, and LoRA are able to detect either rail track damages due to unbearable temperature or determine the fixed object in a far-away distance in a railway network. Thus, an early warning system is deployed to avoid unforeseen fatal accidents. The system comprises LoRa transceivers at both train stations and along the railway tracks, allowing continuous monitoring and control of train movements. Signals related to train speed, location, and track conditions are transmitted using LoRa modules to a centralized control unit, where decisions are made regarding track changes and signal displays. Additionally, the low power consumption of LoRa makes it ideal for remote locations where power supply is limited. This LoRa based signaling system promises to enhance the safety and reliability of railway operations, reduce infrastructure costs and simplify maintenance.

### **2.1.3 SMART TRACK CONTINUOUS MONITORING AND TRAIN COLLISION AVOIDANCE SYSTEM USING IOT**

-Bhavya V, Swetha P , Mr. S. Senthil Kumar(2025)

Railway transportation plays a vital role in the economic and social development of nations, but ensuring safety and efficiency remains a major challenge. The increasing frequency of train accidents due to human errors, mechanical failures, and outdated monitoring systems necessitates the development of intelligent, automated solutions. This paper introduces a Smart Track Continuous Monitoring and Train Collision Avoidance System that integrates multiple advanced technologies, including IoT, infrared (IR) sensors, gyroscope sensors, GPS, and Zigbee communication, to enhance railway safety and prevent collisions. The system operates by detecting the presence of trains using IR sensors, ensuring train alignment through gyroscope sensors, and providing real-time location tracking via GPS modules. Wireless communication is facilitated using Zigbee modules, which enable inter-train and train-to-control center communication, reducing latency and improving response times. The system is also integrated with an IoT-based cloud platform, allowing remote monitoring and data analytics for predictive maintenance and operational optimization

### **2.1.4 AUTOMATIC RAILWAY LEVEL CROSSING USING LORA**

-Prakash Moorthy, Vishnupriya P, Poornima N, Priyadarshini R, Shubhangi Kharche(2020)

Railway level crossing is considered to be one of the most dangerous points in rail networks. In recent research, it is indicated that the risk of accidents due to railway level crossing is increasing. The need for an automatic system arose due to poor management

and responsibility of the railway system. This paper deals with automation in railway system using mechanical and electrical components to control the railway gate. To avoid human intervention at level crossings system need to automate the process of railway gate control. The paper is based on modern wireless technology LoRa (Long Range communication) which mainly indicates the distance of train from the railway booth. The LoRa transmitter mounted with Arduino on train sends the data bits of distance of train concerning railway booth. LoRa receiver placed at railway station receives these bits of data and performs closing and opening of gates and also performs an alarming function to indicate the passenger about the arrival of a train. All this operation makes the railway level crossing automatic without human invention. Thus our paper presents a secure and safe system with low cost and efficiently benefits society.

### **2.1.5 AUTOMATIC RAILWAY CRACK DETECTOR SYSTEM USING LORA NETWORK**

-Haritha P, Sharnya V(2019)

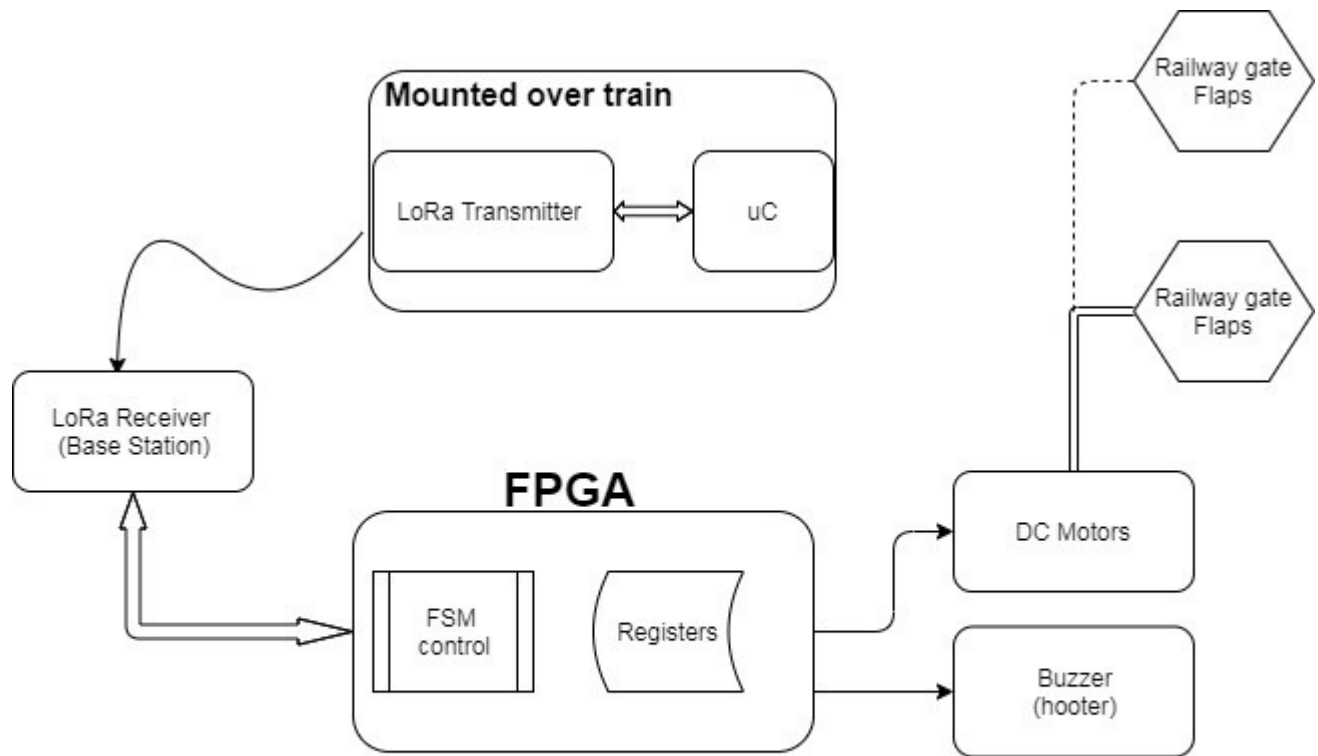
Safety and reliability are considered in one of the main issues at all transport system, particularly in railway. An automatic railway track crack detector system for Railway has been proposed here which aims in building a robot that can detect and analyze any kind of crack on the railway line and send the coordinates of that faulty line to the concerned authority. The proposed system can be networked with multiple robots and central computer system can control all these robots, so that complete track can be scanned for detecting any crack before each time train passes through track. Multiple such systems can be connected by using LoRa network instead of GSM network. This will eliminate the connectivity and network traffic issues. With the networked robotic system can be connected with the railway signaling system to synchronize the signaling with the crack detection. Synchronization with the signaling system can help automatic control of traffic if any crack is detected.



## CHAPTER 3

### EXISTING SYSTEM

#### 3.1 AUTOMATIC RAILWAY LEVEL CROSSING



**Block Diagram of Automatic Railway level Crossing system using LoRa**

**FIGURE 3.1 SYSTEM ARCHITECTURE OF EXISTING SYSTEM**

In this system, the blocks will only allow one train to pass at a time. After that train is passed another train is allowed this is so if any other train is approaching from the opposite direction, then it needs another track. Transmitter system which is mounted over the train consists of an Arduino UNO board connected to wireless communication module SX1278 chip designed by SemtechTechnologies which is based on spread

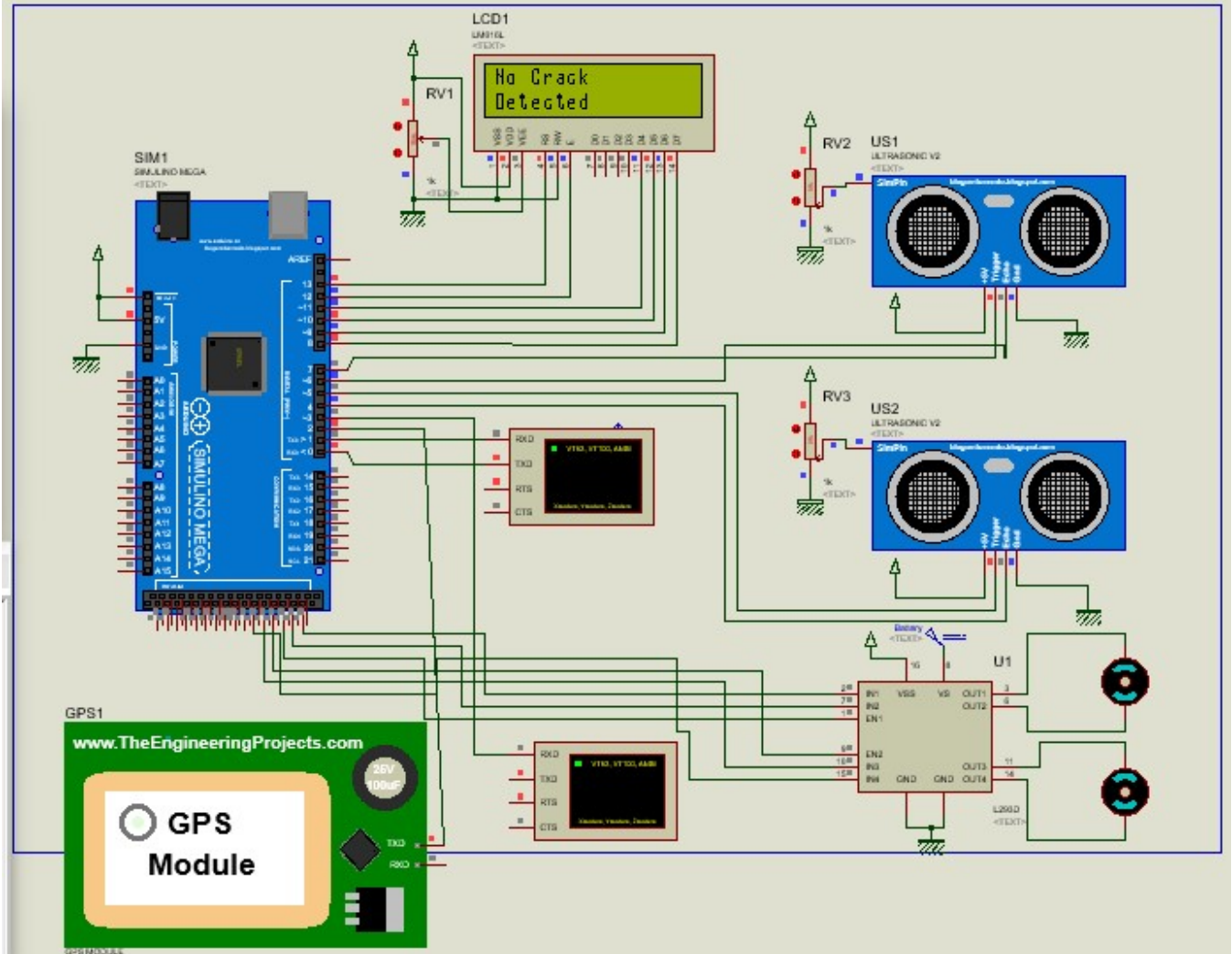
spectrum modulation technique, widely known as LoRa Technology. LoRa has an operating frequency of 433 MHz and operates at 3.3V, providing range up to 5Km. The chip continuously sends data packets.

The base station, Fig 3.1, contains LoRa module SX1278 connected to the main controller of the receiver section, Spartan 6 Numato MIMAS V2 board which has a programmable hardware, this makes coding the system more dynamic and VHDL Hardware description language is used for coding the board. The communication module is interfaced to FPGA by SPI protocol. The receiver chip continuously detects the signals caught by the antenna for data packets. The transmitted signals are in the range of a receiver, this detected signal triggers the controller to command the Servo motor to shift the Flaps downwards, and the Buzzer alarms are set on. These will remain so until the arriving train is in the range of the receiver, once the connection between transceivers is lost the buzzers are signalled off and gate flaps are pulled up.

### **3.2 RAILWAY TRACK CRACK DETECTION SYSTEM**

This robot includes two ultrasonic sensors, GPS, GSM modules, and Arduino Mega based crack detection assembly which is cost effective and robust to facilitate better safety standards in railways. As soon as the robot passed through a crack that might cause the derailment of a train, the ultrasonic sensors sense that and generate a signal. Then this signal is fed into the Arduino Mega. At that point, with the assistance of GSM and GPS modules, an alert SMS consist of the geographic coordinate of that damaged track is sent to the nearby railway authority who can easily take necessary steps to resolve the problem before any major accident occurs. This will save several trains in Bangladesh from an unwanted discontinuity from the rail track. The proposed system can be networked with multiple robots and central computer system can control all these robots, so that complete track can be scanned for detecting any crack before each time train

passes through track. Multiple such systems can be connected by using LoRa network instead of GSM network. This will eliminate the connectivity and network traffic issues.



**FIGURE 3.2 EXISTING SYSTEM SIMULATED DIAGRAM**

## CHAPTER 4

### PROPOSED DESIGN

#### 4.1 INTRODUCTION

The proposed system aims to address one of the critical challenges in railway safety—collision prevention—through the integration of embedded systems, wireless communication, and automated control technologies. This intelligent system is designed to monitor train positions in real-time, communicate location data wirelessly between nearby trains, and initiate preventive actions automatically to avoid collisions. The system operates independently of existing railway infrastructure, making it suitable for deployment in both urban and remote railway tracks.

The communication between trains is established through the LoRa (Long Range) protocol, which is particularly well-suited for railway environments due to its low power consumption and extended range capabilities (up to several kilometers). Each train transmits its GPS coordinates at regular intervals while also listening for location data from nearby trains. This decentralized peer-to-peer communication approach removes the need for centralized infrastructure and makes the system highly scalable. The distance between two points on the Earth's surface given their latitude and longitude. When the computed distance between any two trains falls below a defined safety threshold (e.g., 500 meters), the system raises a warning and activates appropriate countermeasures.

An important innovation in the proposed system is the inclusion of an Automatic Train Control (ATC) mechanism. Once a collision risk is identified, the ATC subsystem takes control of the train's operation and can autonomously reduce the train's speed or bring it to a complete halt, depending on the severity of the risk and proximity to the other train. This automation significantly enhances the system's reliability and reduces dependence on human reaction time, especially in critical scenarios.

For local alerts, an OLED display provides real-time status updates and warning messages to the train operator. Simultaneously, a buzzer is triggered to produce audible alerts, drawing immediate attention. The Wi-Fi connectivity of the Raspberry Pi Pico W is utilized to connect the system to the Blynk IoT cloud platform, which allows remote monitoring and visualization of train status on a mobile app. The Blynk dashboard receives live GPS updates, alert logs, and system status, offering an intuitive interface for control room operators or maintenance staff.

The overall system is compact, cost-effective, and energy-efficient, making it ideal for practical deployment on locomotives, especially in areas ,. Moreover, the integration of cloud-based services ensures scalability and remote observability, which is vital for future smart railway networks.

## 4.2 BLOCK DIAGRAM OF PROPOSED SYSTEM

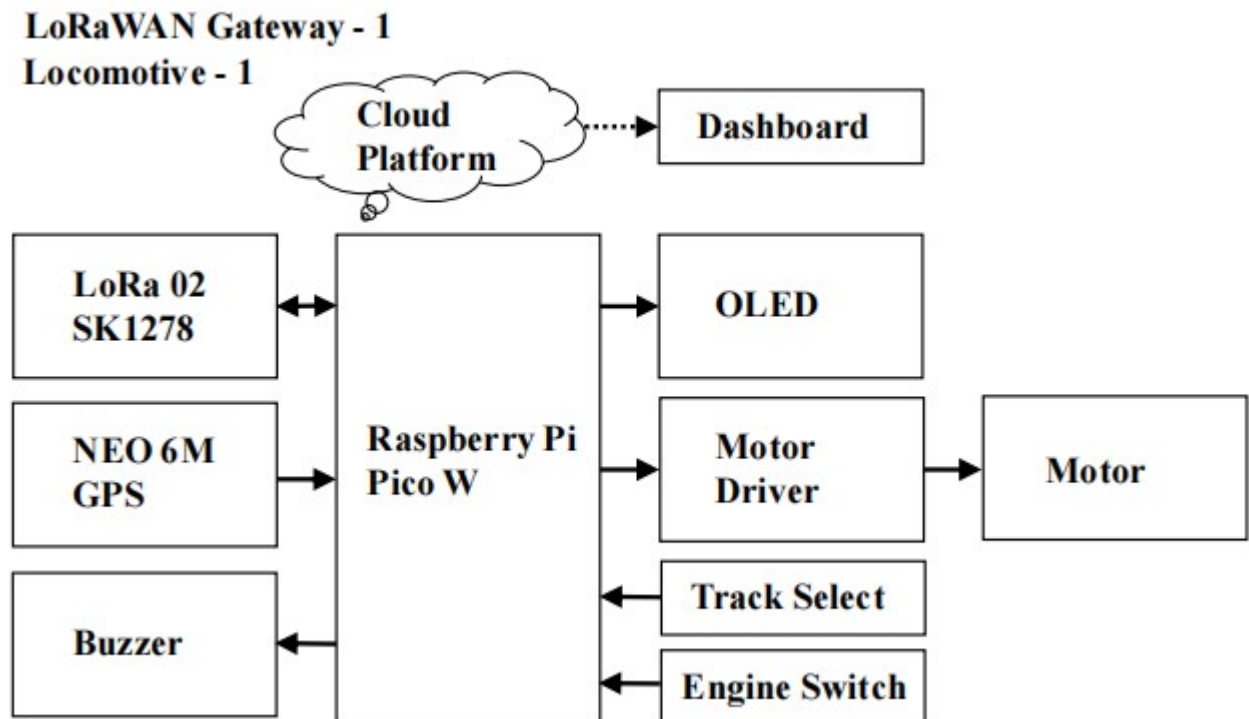
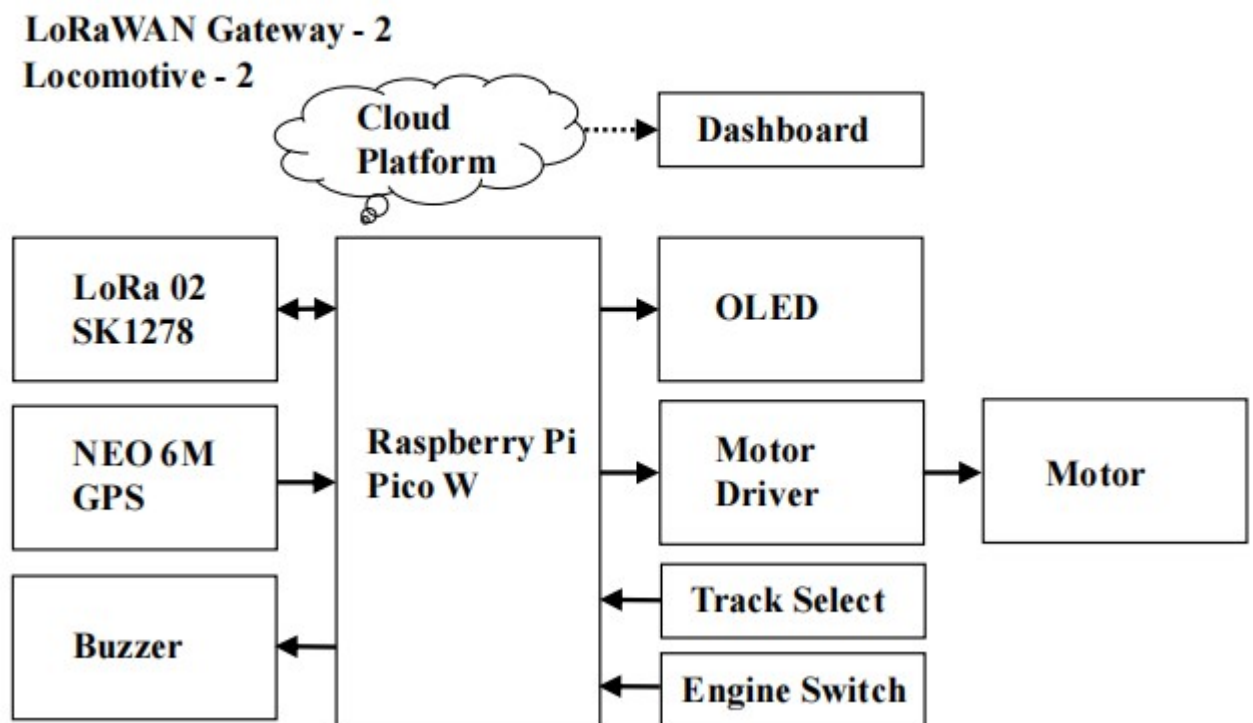


Figure 4.1 Block Diagram of Proposed System(LM-1)

The block diagram of the project titled "An Intelligent Collision Avoidance System for Railways using LoRaWAN Technology with ATC Integration" illustrates the communication and control architecture implemented between two locomotives using embedded and wireless communication components. Each locomotive is equipped with an identical embedded setup, and both are capable of operating independently while communicating wirelessly through a LoRaWAN-based network.

The above block diagram represents the Locomotive 1 system which transmit the required data for the avoidance of railway system which the parameters to detect the speed,distance,track number,Location of the train



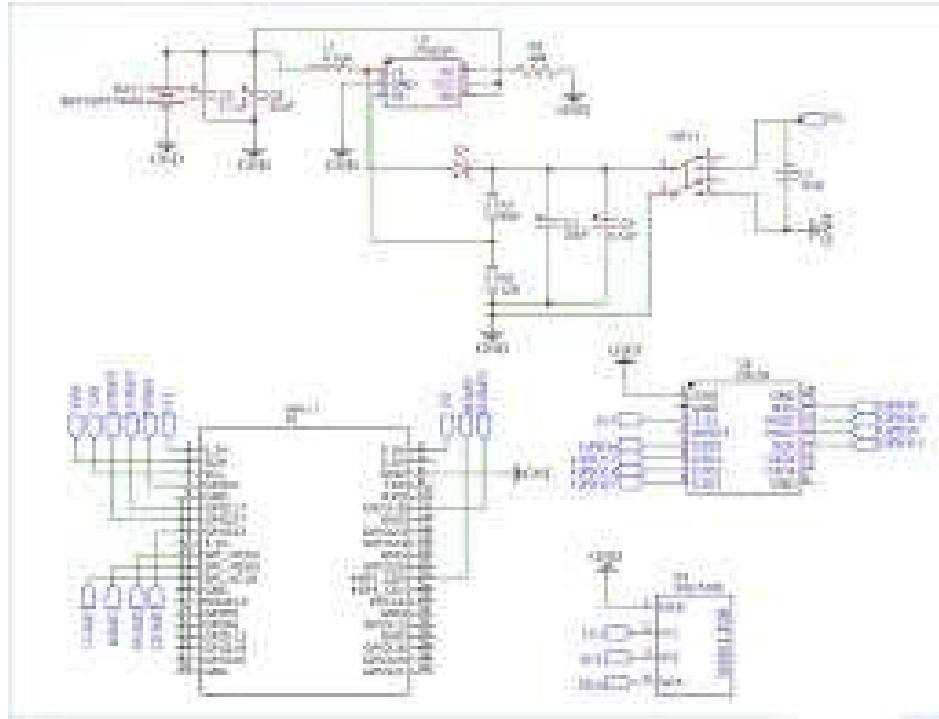
**FIGURE 4.2 BLOCK DIAGRAM OF PROPOSED SYSTEM(LM-2)**

The LM-2 also integrated the Lora 02 SX1278 Module with NEO 6M GPS module to avoid collision in railways using the communication by LORAWAN Technology

The GPS data collected is transmitted through the LoRa SX1278 module (LoRa 02), which allows for long-range, low-power communication between locomotives. The LoRa modules of both locomotives also communicate with a centralized LoRaWAN Gateway, which connects to the internet and transmits collected data to a Cloud Dashboard Platform, such as Blynk. This cloud interface provides real-time monitoring, data logging, and mobile alerts to railway control personnel or operators.

In addition to wireless communication and sensing, each locomotive includes an Automatic Train Control (ATC) mechanism implemented through a motor driver circuit and motor setup. The motor driver receives control signals from the Raspberry Pi Pico W, allowing it to stop or slow down the locomotive automatically when a collision threat is detected. For local alerting and user feedback, each unit is equipped with a buzzer and an OLED display. The buzzer provides an immediate audible warning in the event of a collision risk, while the OLED display shows dynamic messages such as train location, communication status, or warning alerts

### 4.3 CIRCUIT DIAGRAM



**Figure 4.2 Circuit Diagram of Proposed System**

The circuit diagram of this project “Intelligent Collision Avoidance for railways using LoRaWAN technology” uses Raspberry Pi Pico W as the main controller. A NEO-6M GPS module gives location data. LoRa SX1278 modules send and receive GPS data between trains. A buzzer alerts when collision risk is detected. An OLED display shows position and warnings. A motor driver runs small DC motors for the demo. All modules are powered by a 5V source and connected to Pico W using UART, I2C, and GPIO.



## 4.4 METHODOLOGY

The project involves the integration of hardware components with efficient communication and detection algorithms to build a smart collision avoidance system for railways. The central controller of the system is the Raspberry Pi Pico W, which collects real-time GPS data from the NEO-6M GPS module using UART communication. The location data, including latitude and longitude, is processed by the Pico W to determine the train's current position. To enable wireless communication between trains, the system uses the LoRa SX1278 module, connected via SPI, which allows GPS data to be exchanged over long distances with low power consumption.

Each unit broadcasts its GPS location while simultaneously listening for incoming coordinates from other nearby train units. Upon receiving this data, the Raspberry Pi Pico W calculates the distance between its own location and that of other trains using the Haversine formula. If the calculated distance falls below a predefined safety threshold, indicating a potential risk of collision, the system activates a buzzer to audibly alert the train driver. At the same time, a warning message is displayed on the OLED screen, providing visual notification of the threat.

To simulate train movement in the prototype, a motor driver and DC motor are used, controlled by the Pico W through GPIO and PWM signals. Additionally, the system can optionally upload location and alert data to a cloud dashboard via Wi-Fi for remote monitoring and historical analysis. This methodology ensures real-time tracking, efficient communication, and timely warning generation to prevent railway collisions using a low-power, reliable LoRaWAN-based approach.

## **CHAPTER 5**

### **SYSTEM SPECIFICATION**

#### **5.1 HARDWARE COMPONENTS**

##### **5.1.1 Microcontroller (RASPERRY PI PICO W)**

The Raspberry Pi Pico W is a compact yet powerful microcontroller board designed for physical computing and IoT applications. It builds upon the success of the original Raspberry Pi Pico, introducing wireless connectivity through Wi-Fi and Bluetooth. This enhancement makes it an excellent choice for projects requiring remote communication.

The Raspberry Pi Pico W combines powerful real-time processing, wireless communication, and flexible I/O in a compact board. It's ideal for embedded, IoT, robotics, and smart sensing applications. Its rich I/O and PIO capability allow users to emulate complex protocols or create custom peripherals, while Wi-Fi opens the door to cloud-connected projects.

The Raspberry Pi Pico W is powered by the RP2040 microcontroller, designed in-house by Raspberry Pi.

##### **Key Features of RP2040:**

- Processor: Dual-core Arm Cortex-M0+ @ up to 133 MHz
- Memory: 264 KB of SRAM 2 MB of onboard QSPI Flash
- Wi-Fi: 2.4 GHz 802.11n (via Infineon CYW43439)
- DMA Controller: 12-channel Direct Memory Access
- Timer & Watchdog: 4 × hardware timers, real-time clock (RTC), watchdog

- Debug Support: SWD (Serial Wire Debug)
- Floating Point: No hardware FPU (Fixed-point math preferred)



**Figure 5.1 RASPBERRY PI PICO W**

## **Digital I/O**

The Raspberry Pi Pico W features 26 multifunctional GPIO (General Purpose Input/Output) pins. These pins can be programmed as digital inputs or outputs, and many of them can also be configured for alternate functions like SPI, I2C, UART, PWM, and ADC.

**GPIO numbers:** GPIO0 to GPIO28 (but only GPIO0 to GPIO22, and GPIO26–28 are available on the pin header)

**Voltage levels:** 3.3V logic (do not apply 5V to GPIO pins!)

**Drive strength:** Each GPIO can source/sink ~12 mA

## **Analog Pins**

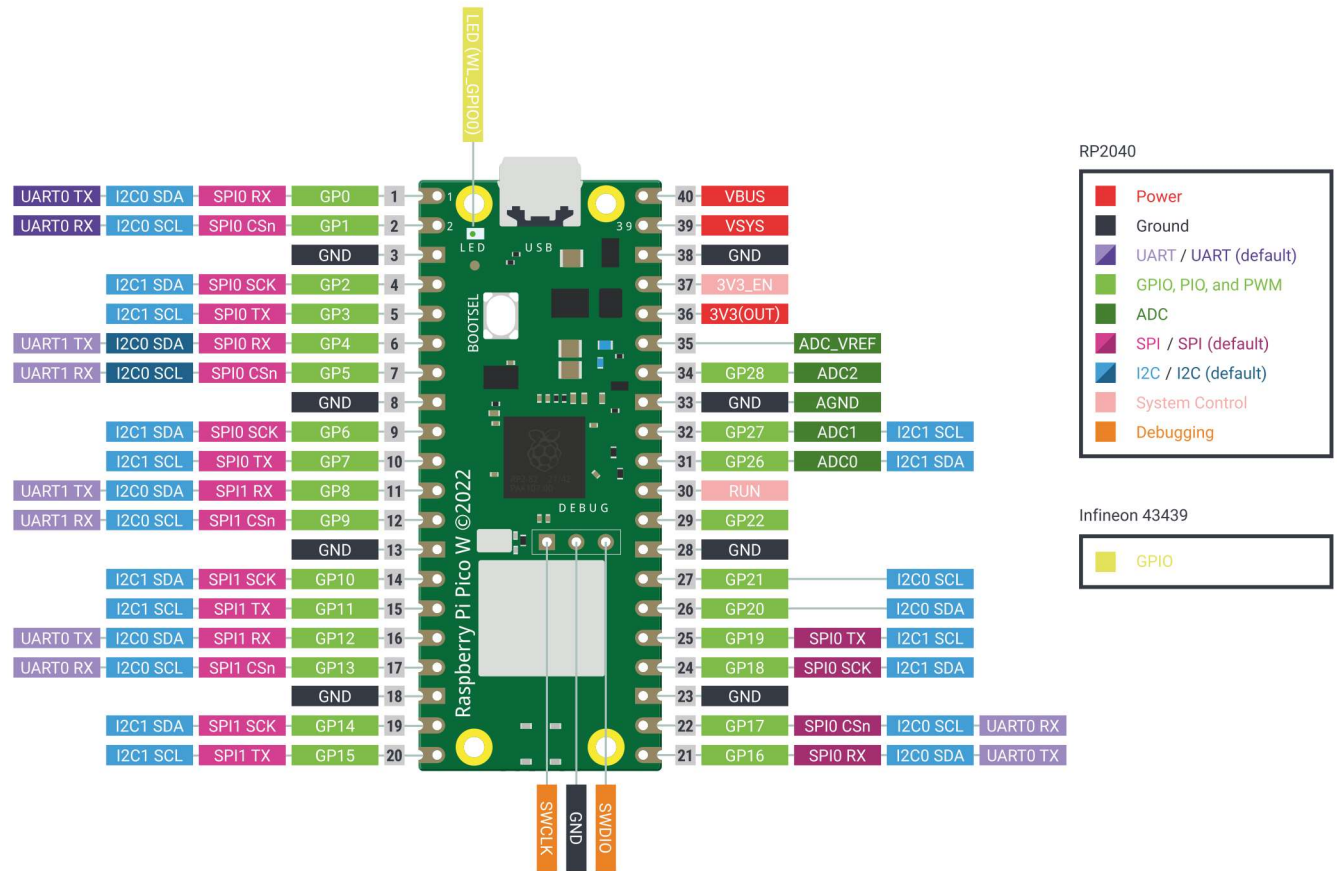
The RP2040 microcontroller on the Raspberry Pi Pico W includes a 12-bit Analog-to-Digital Converter (ADC) with 3 external analog input channels and 1 internal temperature sensor channel.

### **ADC Specifications**

The Raspberry Pi Pico W microcontroller features a built-in 12-bit Analog-to-Digital Converter (ADC) with three external analog input channels (GPIO26, GPIO27, and GPIO28) and one internal channel connected to a temperature sensor. The ADC can convert input voltages ranging from 0 to 3.3 volts into digital values, providing a resolution of 4096 discrete steps (0–4095). This allows for precise analog signal measurement, such as reading sensors or variable voltages. The ADC operates with the internal 3.3V as its reference and can achieve sampling rates up to 500,000 samples per second under optimal conditions, making it suitable for a wide range of real-time analog applications.

- Raspberry Pi Pico W has 3 external analog input pins: GPIO26, GPIO27, GPIO28.
- These connect to a 12-bit ADC and accept voltages from 0V to 3.3V.
- An internal ADC channel is used for temperature sensing (ADC4).
- ADCs are useful for reading sensors, voltages, or analog input signals.

## PINOUT EXPLANATION



### Figure 5.2 Raspberry pi pico w pinout diagram

The Pico W has a **40-pin** header (2×20, 0.1” pitch), out of which **26 are GPIOs**. Here's the breakdown:

## □ Power Pins

**VSYS (Pin 39):** Main input voltage (1.8V to 5.5V)

**VBUS (Pin 40):** USB power (5V from USB port)

**3V3 (Pin 36):** Regulated 3.3V output (up to 300mA)

**GND:** Multiple ground pins available

**3V3\_EN (Pin 37):** Connect to GND to disable 3.3V regulator

#### ☐ **Digital I/O Pins**

**GPIO0 – GPIO22:** Configurable for SPI, I2C, UART, PWM, or GPIO

**GPIO23 – GPIO28:** Not broken out to the header (used internally or for debugging)

#### ☐ **ADC (Analog Inputs)**

**GPIO26 – ADC0**

**GPIO27 – ADC1**

**GPIO28 – ADC2**

#### ☐ **UART (Serial)**

**UART0:** GPIO0 (TX), GPIO1 (RX)

**UART1:** GPIO4 (TX), GPIO5 (RX)

#### ☐ **SPI**

**SPI0 (Default):** GPIO2 (MOSI), GPIO3 (MISO), GPIO4 (SCK)

**SPI1:** Available on multiple pins

#### ☐ **I2C**

**I2C0:** GPIO0 (SDA), GPIO1 (SCL)

## **I2C1: GPIO2 (SDA), GPIO3 (SCL)**

### **□ PWM**

**All GPIOs** can be used for PWM (2 channels per slice, 8 slices)

### **□ Wi-Fi (CYW43439)**

Wi-Fi chip is connected via SPI internally, no user-accessible pins

Shares SPI bus and interrupt

## **Wireless-Specific Pins**

Some GPIOs are dedicated to wireless functions:

**WL\_GPIO0** – Controls the built-in user LED.

**WL\_GPIO1** – Manages the on-board SMPS power save pin.

**WL\_GPIO2** – Used for VBUS sensing (high if VBUS is present, low otherwise).

## **Use Cases**

With its capabilities, the Pico W is perfect for IoT projects, home automation, sensor networks, and remote monitoring systems. Its low-power sleep modes make it suitable for battery-operated applications, while its drag-and-drop programming via USB mass storage simplifies development.

### 5.1.2 TRANSEIVER



**Figure 5.3 LORA 02 SX1278**

Ra-02 can be used for ultra-long distance spread spectrum communication, and compatible FSK remote modulation and demodulation quickly, to solve the traditional wireless design can not take into account the distance, anti-interference, and power consumption.

The Ra-02 wireless module, built on SX1278 transceiver, uses advanced LoRa spread spectrum technology, enabling seamless communication over long distances of up to 10,000 meters. Ra-02 wireless module stands out for its anti-jamming capabilities and includes an air wake-up feature, effectively preserving power.

- Communication distance: 10KM
- Sensitivity: down to -148dBm
- Programmable bit rates: up to 300kbps
- RSSI dynamic range: 127dB



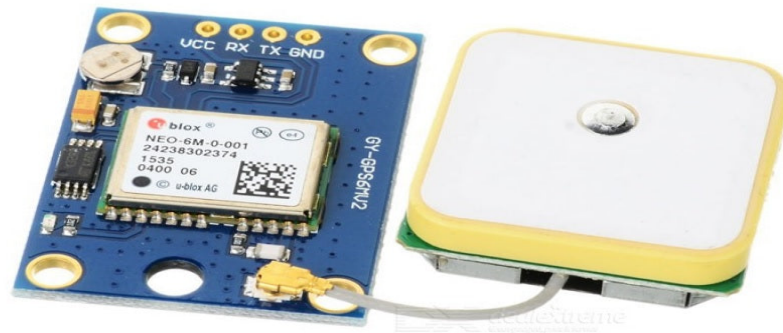
**Features:**

- LoRa Spread Spectrum modulation technology
- Constant RF power output at + 20dBm-100mW voltage change
- Half-duplex SPI communication
- Supports FSK, GFSK, MSK, GMSK, LoRa and OOK modulation modes
- Automatic RF signal detection, CAD mode and very high speed AFC
- Packet engine with CRC up to 256 bytes
- Small footprint dual-row stamp-hole patch package
- Shielded housing
- Spring Antenna

**Specifications:**

- Communication distance: 10Km
- Sensitivity: down to -148dBm
- Programmable bit rates: up to 300kbps
- RSSI dynamic range: 127dB
- Wireless frequency: 433MHz
- Working voltage: 1.8-3.7v
- Working temperature: -40 – +80

### 5.1.3 NEO 6M GPS



**Figure 5.4 NEO 6M GPS MODULE**

The NEO-6MV2 is a GPS (Global Positioning System) module and is used for navigation. The module simply checks its location on earth and provides output data which is longitude and latitude of its position. It is from a family of stand-alone GPS receivers featuring the high performance u-blox 6 positioning engine. These flexible and cost effective receivers offer numerous connectivity options in a miniature (16 x 12.2 x 2.4 mm) package. The compact architecture, power and memory options make NEO-6 modules ideal for battery operated mobile devices with very strict cost and space constraints. Its Innovative design gives NEO-6MV2 excellent navigation performance even in the most challenging environments.

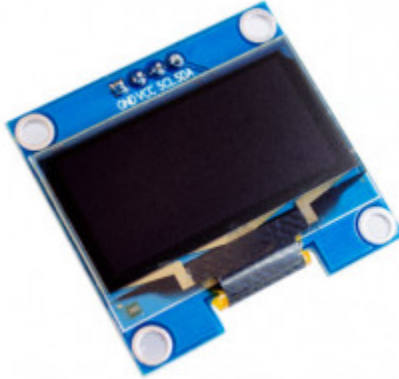
### **Technical Specification:**

- Maximum navigation update rate: 5Hz
- Receiver type: 50 Channels - GPS L1 frequency - SBAS (WAAS, EGNOS, MSAS, GAGAN)
- Time-To-First-fix: For Cold Start 32s, For Warm Start 23s, For Hot Start <1s
- Default baud rate: 9600bps
- EEPROM with battery backup
- Sensitivity: -160dBm
- Supply voltage: 3.6V
- Maximum DC current at any output: 10mA
- Operation limits: Gravity-4g, Altitude-50000m, Velocity-500m/s
- Operating temperature range: -40°C TO 85°C

### **Features:**

- Standalone GPS receiver
- Anti-jamming technology
- UART Interface at the output pins (Can use SPI ,I2C and USB by soldering pins to the chip core)
- Under 1 second time-to-first-fix for hot and aided starts
- Location data provided by it is accurate

### 5.1.4 OLED DISPLAY



**Figure 5.5 OLED DISPLAY**

The SSD1306 embeds with contrast control, display RAM and oscillator, which reduces the number of external components and power consumption. It has 256-step brightness control. Data/Commands are sent from general MCU through the hardware selectable 6800/8000 series compatible Parallel Interface, I<sup>2</sup>C interface or Serial Peripheral Interface. It is suitable for many compact portable applications, such as mobile phone sub-display, MP3 player and calculator, etc.

#### **FEATURES**

- Resolution: 128 x 64 dot matrix panel
- Power supply o  $V_{DD} = 1.65V$  to  $3.3V$ ,
- <  $V_{BAT}$  for IC logic

$V_{BAT} = 3.3V$  to  $4.2V$  for charge pump regulator circuit

VCC = 7V to 15V for Panel driving

- For matrix display
  - o Segment maximum source current: 100uA
  - o Common maximum sink current: 15mA
  - o 256 step contrast brightness current control
- Embedded 128 x 64 bit SRAM display buffer
- Pin selectable MCU Interfaces:
  - o 8-bit 6800/8080-series parallel interface
  - o 3 /4 wire Serial Peripheral Interface
  - o I 2 C Interface
- Screen saving continuous scrolling function in both horizontal and vertical direction
- Internal charge pump regulator
- RAM write synchronization signal
- Programmable Frame Rate and Multiplexing Ratio
- Row Re-mapping and Column Re-mapping
- On-Chip Oscillator
- Chip layout for COG & COF
- Wide range of operating temperature: -40°C to 85°C

### 5.1.5 MOTOR AND MOTOR DRIVER



**Figure 5.6 L293D MOTOR DRIVER**

L293D motor Driver IC is an integrated circuit that can drive two motors simultaneously and is usually used to control the motors in an autonomous system. This motor driver IC enables us to drive a DC motor in either direction and also control the speed of the motor.

L293D is a dual H-bridge motor driver IC. H-bridge is the simplest circuit for controlling a low current-rated motor. One H-bridge is capable to drive a DC motor bidirectional. L293D is a current enhancing IC. It can also act as a switching device.

The L293D is a 16-pin Integrated circuit, with eight pins, on each side, dedicated to the controlling of a motor. There are 2 input pins, 2 output pins and 1 enable pin for each motor. The L293D IC is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. It is designed to drive inductive loads such as relays, solenoids, DC & bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications.

L293D motor Driver IC is one of the most popular drivers in the market. Because of several reasons such as cheap price (compared to other drivers), easy control, proper shape and size, no need for protective circuit and diodes, no need for heat sinks, and good resistance to temperature and high-speed variations, L293D motor driver is mostly preferred driver to the user.



**FIGURE 5.7 DC MOTOR**

The 12V DC motor is used to simulate the movement of a train or railway gate in our prototype. It helps demonstrate how the system reacts to potential collisions by controlling motion (e.g., stopping or redirecting a train, or activating a warning mechanism).

#### Working Principle:

When powered with 12V, the motor converts electrical energy into rotational motion using electromagnetic principles. The current through the armature creates a magnetic field, which interacts with permanent magnets, causing the shaft to spin



**FIGURE 5.8 BUZZER.**

The buzzer is used to give an audible warning when a potential railway collision is detected. This ensures alerting train operators or nearby personnel about the danger in real-time.

### Working Principle

When DC voltage is applied, the buzzer's internal piezoelectric element vibrates, producing sound.

Active buzzer: has an internal oscillator — produces sound when powered.

Controlled by microcontroller (Raspberry Pi Pico W) using GPIO pin (HIGH to activate).



## 5.2 SOFTWARE REQUIREMENTS

### 5.2.1 Arduino Software (IDE)

The Arduino Software (IDE) makes it easy to write code and upload it to the board offline. We recommend it for users with poor or no internet connection. This software can be used with any Arduino board.

There are currently two versions of the Arduino IDE, one is the IDE 1.x.x and the other is IDE 2.x. The IDE 2.x is new major release that is faster and even more powerful to the IDE 1.x.x. In addition to a more modern editor and a more responsive interface it includes advanced features to help users with their coding and debugging.

### 5.2.2 Steps to install Arduino Software

The following steps can guide you with using the offline IDE (you can choose either IDE 1.x.x or IDE 2.x):

1. Download and install the Arduino Software IDE:
2. Connect your Arduino board to your device.
3. Open the Arduino Software (IDE).

**The Arduino Integrated Development Environment** – or Arduino Software (IDE) – connects to the Arduino boards to upload programs and communicate with them. Programs written using Arduino Software (IDE) are called **sketches**. These sketches are written in the text editor and are saved with the file extension .ino.

### Using the offline IDE 2.x

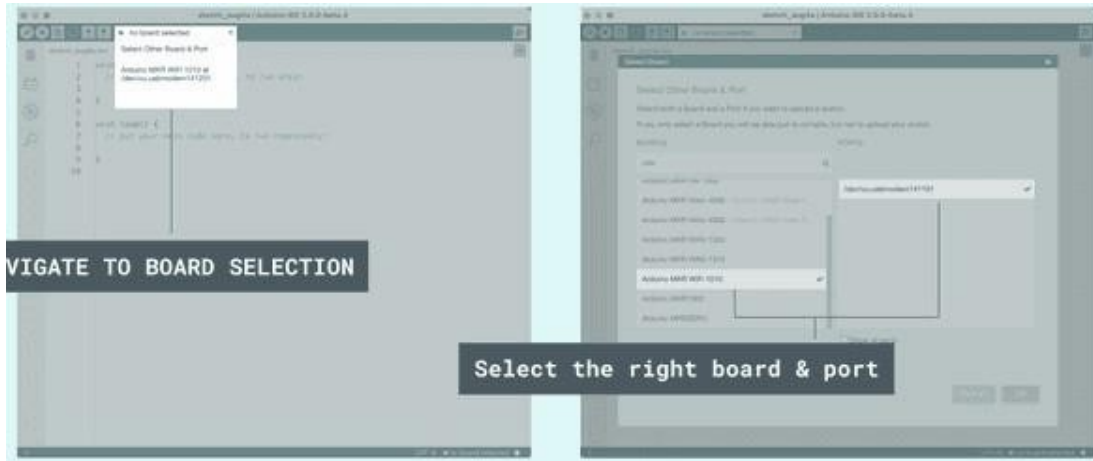
The editor contains the five main areas:



**Figure 5.9 The Arduino Software IDE**

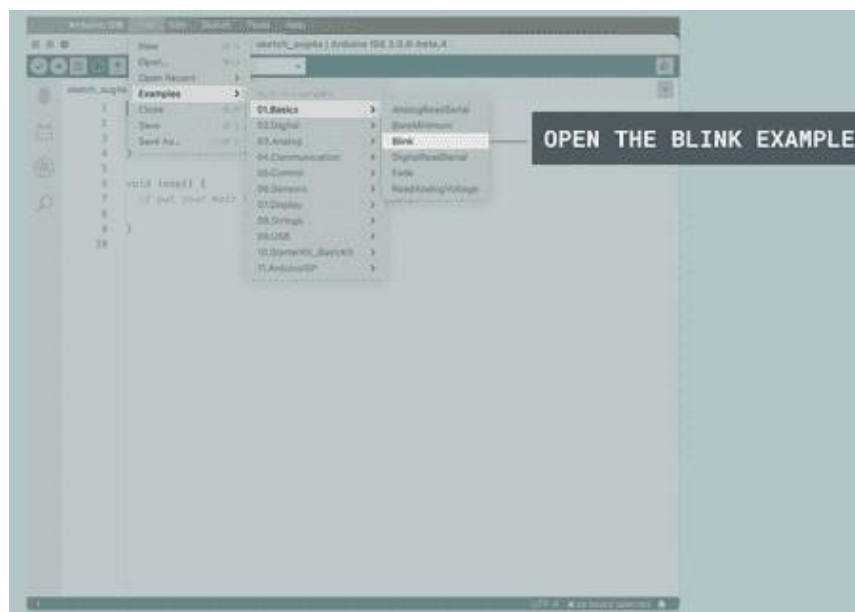
1. A **toolbar with buttons** for common functions and a series of menus. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, choose your board and port and open the serial monitor.
2. The **Sidebar** for regularly used tools. It gives you quick access to board managers, libraries, debugging your board as well as a search and replacement tool.
3. The **text editor** for writing your code.
4. **Console controls** gives control over the output on the console.
5. The **text console** displays text output by the Arduino Software (IDE), including complete error messages and other information.
6. The bottom right-hand corner of the window displays the configured board and serial port.

Now that you are all set up, **let's try to make your board blink**



**Figure 5.10 Selecting a board & port**

2. Now, you need to **select the right board & port**. This is done from the toolbar. Make sure you select the board that you are using. If you cannot find your board, you can add it from the board manager in the sidebar.
3. Let's **try an example**: navigate to **File > Examples > 01.Basics > Blink**.

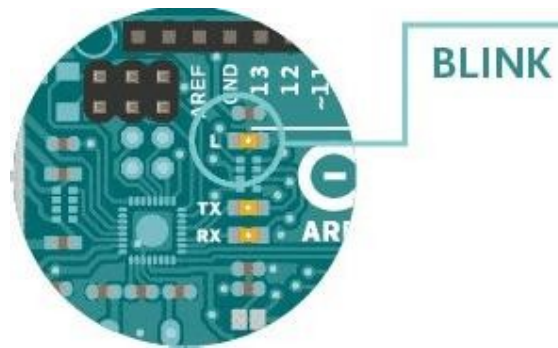


**Figure 5.11 Opening an example**

4. To **upload it to your board**, simply click on the arrow in the top left corner. This process takes a few seconds, and it is important to not disconnect the board during this process. If the upload is successful, the message “Done uploading” will appear in the bottom output area.
5. Once the upload is complete, you should see on your board the yellow LED with the letter **L** next to it, start blinking. You can **adjust the speed of blinking** by changing the delay number in the parenthesis to 100, and upload the Blink sketch again. Now the LED should blink much faster.
6. Now, you need to **select the right board & port**. This is done from the toolbar. Make sure you select the board that you are using. If you cannot find your board, you can add it from the board manager in the sidebar.
7. Let's **try an example**: navigate to **File > Examples > 01.Basics > Blink**.
8. To **upload it to your board**, simply click on the arrow in the top left corner. This process takes a few seconds, and it is important to not disconnect the board during this process. If the upload is successful, the message “Done uploading” will appear in the bottom output area.
9. Once the upload is complete, you should see on your board the yellow LED with the letter **L** next to it, start blinking. You can **adjust the speed of blinking** by changing the delay number in the parenthesis to 100, and upload the Blink sketch again. Now the LED should blink much faster.
10. Now, you need to **select the right board & port**. This is done from the toolbar. Make sure you select the board that you are using. If you cannot find your board, you can add it from the board manager in the sidebar.
11. Let's **try an example**: navigate to **File > Examples > 01.Basics > Blink**.

12. To **upload it to your board**, simply click on the arrow in the top left corner. This process takes a few seconds, and it is important to not disconnect the board during this process. If the upload is successful, the message “Done uploading” will appear in the bottom output area.

13. Once the upload is complete, you should see on your board the yellow LED with the letter **L** next to it, start blinking. You can **adjust the speed of blinking** by changing the delay number in the parenthesis to 100, and upload the Blink sketch again. Now the LED should blink much faster.



**Figure 5.12 Blinking LED in Arduino Board**

**Congratulations!** You have successfully programmed your board to blink its on-board LED.

### **5.2.3 Programming ESP32 Board with Arduino IDE**

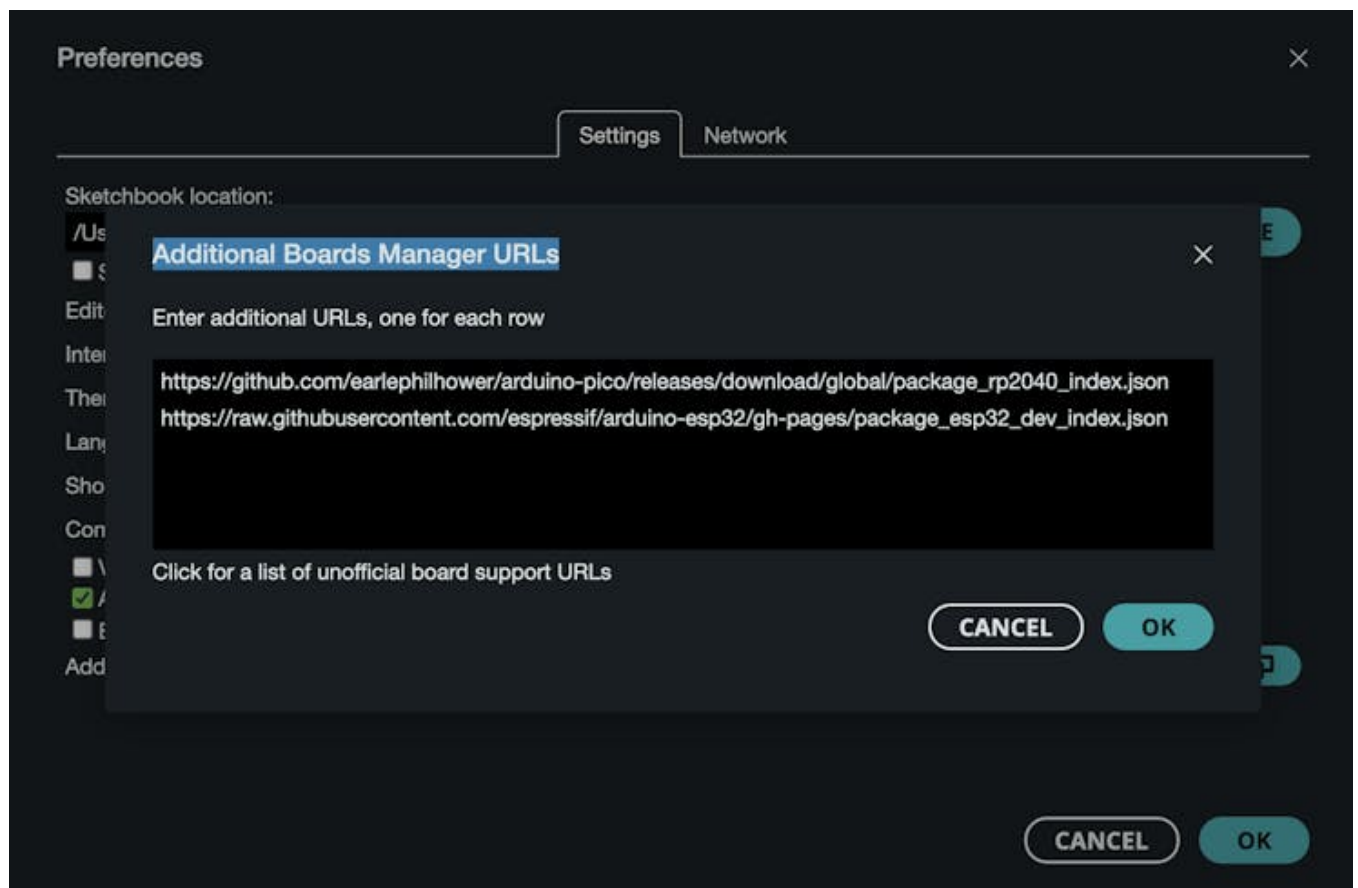
Arduino has officially launched its support for Raspberry boards. So if you have followed the old method of using GIT to install the boards then you would have to follow these steps again (highly recommended) if you need support for new libraries..

You can easily start coding in Arduino on your Raspberry Pi Pico or Pico W. This can be advantageous for people who prefer using the Arduino IDE or the Arduino programming language which is a variant of C/C++. In this tutorial we will go over the quick steps on how to do this, the only thing you will need is a Pico or Pico W and the Arduino IDE installed on your computer

The following steps to program the raspberry pi boards on the Arduino IDE

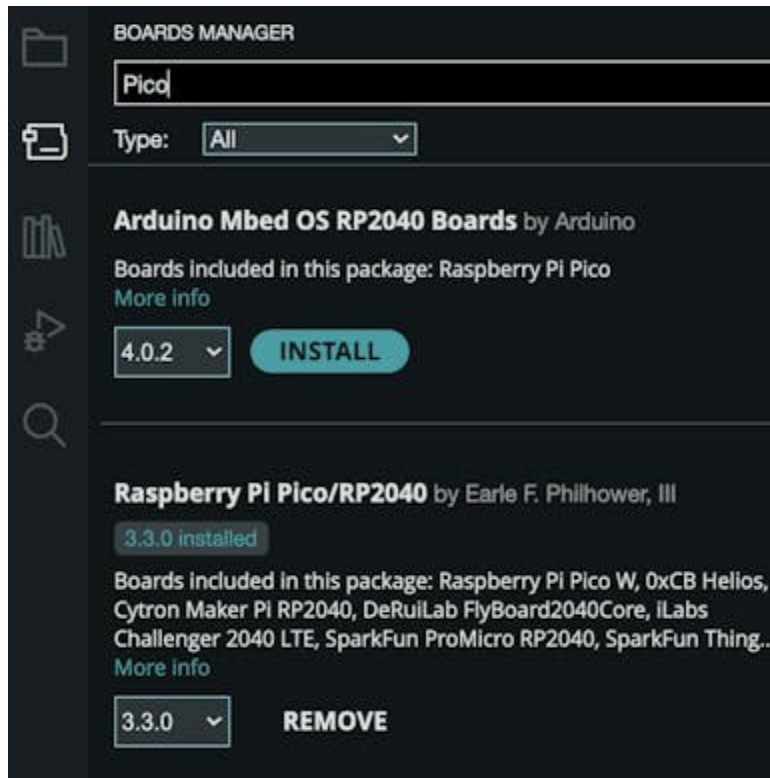
### Step 1: Install the 3rd Party Board Manager.

Because this board is not managed officially by Arduino, you will need to add the board yourself. To do so, add the following link to Additional Boards Manager URLs in the preferences.



**FIGURE 5.13 INSTALL GITHUB LINK IN ARDUINO IDE**

.Once you click okay, you should be able to find the Raspberry Pi Pico board in the Boards Manager. Go ahead and install it.



**FIGURE 5.14 INSTALL PICO BOARD**

## Step 2: Connect Board and Example Code

Now that you installed the Board you need, you can begin connecting to the physical device and write code.

- First, we will be working with the Raspberry Pi Pico W board. To select it go to *Tools > Board* and find the Pico W you are working with.
- Next, select the example we will be working with today. Go to *File > Examples > WiFi > ScanNetworks*. This example only works with the Pico W but it allows us to

scan and see what nearby internet networks are available. You do not need to modify this code

- Next, plug in your Pico W while holding the boot sel button, we need to mount the device so Arduino IDE can generate a COM port for it the first time you are using it

### **Materials Required:**

- Raspberry pi pico w
- Arduino IDE
- Programming cable (micro usb cable)

## **5.2.4 HARDWARE INFORMATION OF RASPBERRY PI PICO**

The Raspberry Pi Pico W is a microcontroller board featuring wireless connectivity. It boasts a dual-core ARM Cortex-M0+ processor (up to 133 MHz) and 264KB of on-chip SRAM. It also includes 2MB of onboard QSPI flash memory. The "W" in Pico W stands for wireless, which in this case includes 2.4GHz 802.11n Wi-Fi and Bluetooth 5.2.

### **Key Hardware Components:**

**Microcontroller:** RP2040 chip, designed by Raspberry Pi.

**Processor:** Dual-core ARM Cortex-M0+.

**Clock Speed:** Up to 133 MHz.

**SRAM:** 264KB.



**Flash Memory:** 2MB QSPI.

**Wireless:** 2.4GHz 802.11n Wi-Fi, Bluetooth 5.2.

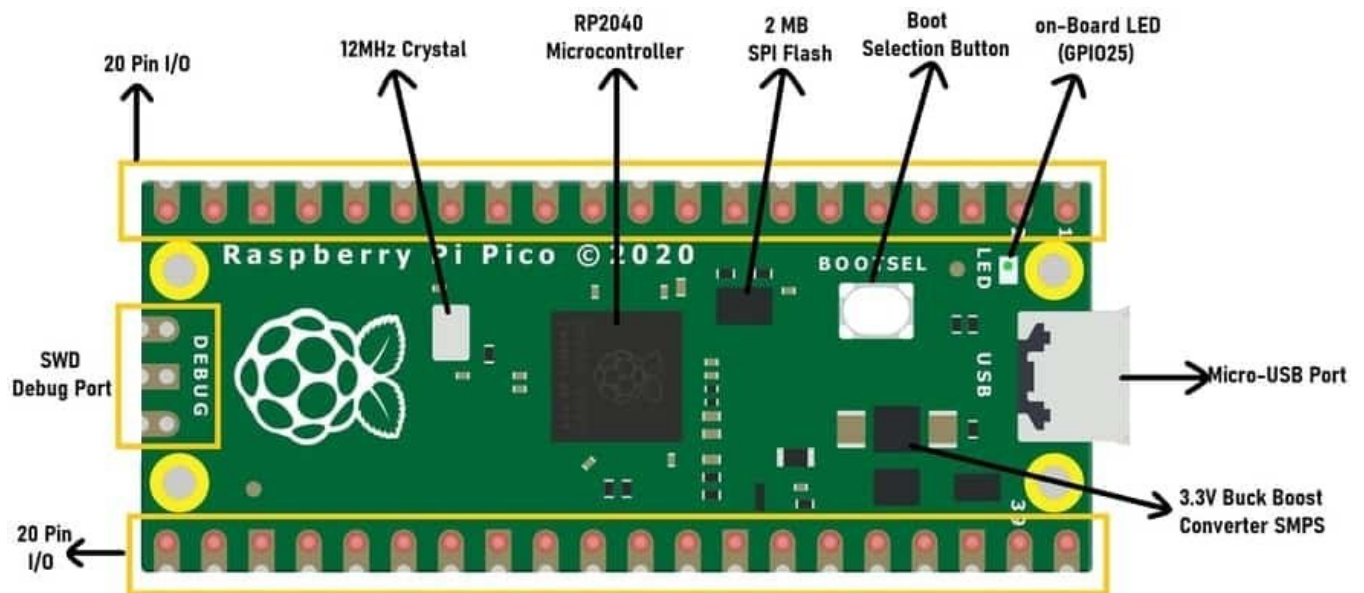
**GPIO Pins:** 26 multifunction GPIO pins, including 3 analog inputs.

**Interfacing:** 2 x UART, 2 x SPI, 2 x I2C controllers.

**PWM Channels:** 16 PWM channels.

**USB:** USB 1.1 controller with host and device support.

**PIO:** 8 x Programmable I/O (PIO) state machines



**FIGURE 5.15 RASPBERRY PI PICO SPECS**

## 5.3 CLOUD INTEGRATION

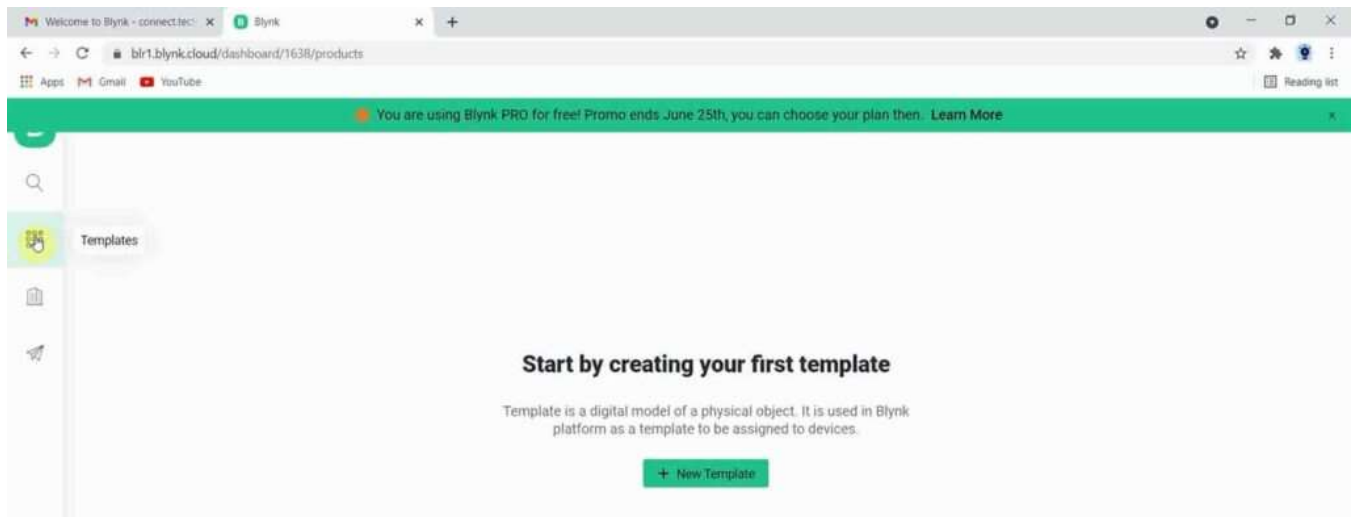
In my project “Intelligent collision avoidance for railways using lorawan technology” that I used Blynk cloud dashboard to view my data parameters continuously upgradable To connect a project with Blynk Iot, you'll need to set up a Blynk project and then configure your device to communicate with it. This involves creating a template, adding datastreams, and then using the Blynk app or web dashboard to interact with your device.

Here's a step-by-step procedure to connect your project with blynk cloud platform

### 1. Set up a Blynk project:

- . Go to the Blynk website and sign up for an account
- . Create a new project and name it.
- . Select the type of device you're using (e.g., Arduino, ESP32, etc.) and the connection type (e.g., Wi-Fi, Ethernet).
- . Choose a project name and select the device you are using.
- . The system will generate an authentication token for your project.

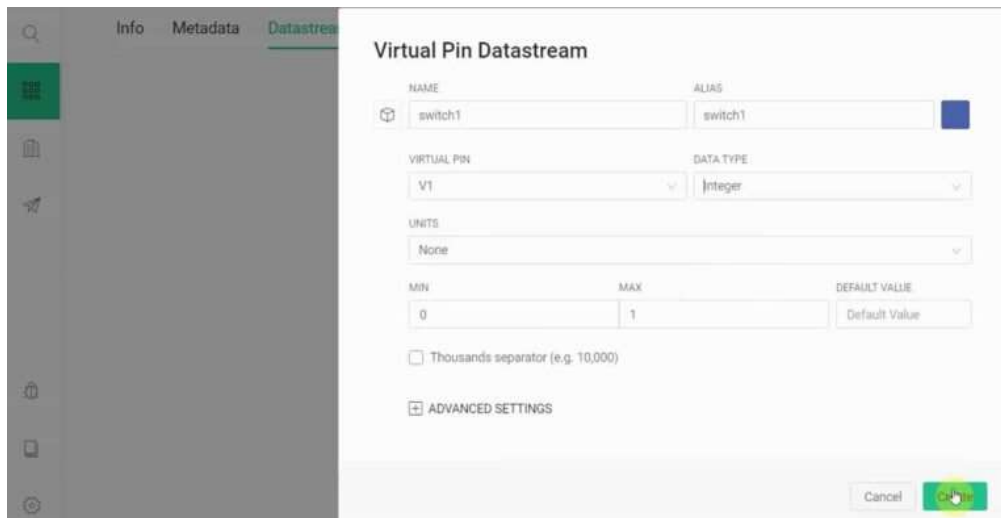
## 2. Create a template:



**FIGURE 5.16 CREATE THE TEMPLATE IN BLYNK PLATFORM**

Click on the "Templates" button and create a new template,  
Name your template and select the device type and connection type.

## 3. Add datastreams:



**FIGURE 5.17 INCLUDE YOUR DATASTREAM DATA AND SAVE**

- . Go to the "Datastreams" tab and add any datastreams you need for your project.
- . These datastreams will be used to transmit data between your device and the Blynk cloud.
- .

#### **4. Configure your device:**

- . **Install Blynk Library:** In your IDE (like Arduino), install the Blynk library.
- . **Use Example Code:** Use the example code provided by Blynk for your device and add your device's Wi-Fi credentials and the authentication token.
- . **Upload Code:** Upload the modified code to your device.
- .

#### **5. Connect your device:**

- . Ensure your device is connected to the Wi-Fi network.
- . Verify that your device is successfully connected to Blynk by checking the Blynk app or web dashboard.
- .

#### **6. Configure your dashboard:**

. Use the Blynk app or web dashboard to add widgets and configure them to interact with your device's datastreams.

## 5.4 EMBEDDED C++

Embedded C++ is a variation of the C++ programming language that is tailored for embedded systems, which are computing devices designed to perform specific functions within a larger system. These systems often have limited resources, such as processing power, memory, and storage, compared to general-purpose computers. Embedded C++ aims to provide the features and flexibility of C++ while being suitable for these resource-constrained environments.

One of the key differences between C++ and Embedded C++ is the approach to memory management. In embedded systems, efficient use of memory is crucial, so Embedded C++ often restricts or modifies certain features of C++ related to memory allocation and deallocation. For example, dynamic memory allocation with `'new'` and `'delete'` operators is typically avoided or used sparingly due to the risk of memory fragmentation and allocation failures. Instead, embedded developers often rely on static memory allocation or custom memory management strategies to ensure predictable memory usage.

Another important consideration in embedded programming is the use of real-time operating systems (RTOS) or bare-metal programming. RTOS provides a higher level of abstraction and can simplify certain aspects of embedded development, such as task scheduling and communication between tasks. However, RTOS introduces overhead and complexity, which may not be suitable for all embedded systems. In contrast, bare-metal programming involves directly programming the hardware without an operating system, which can offer better performance and efficiency but requires more manual effort and expertise.

In terms of language features, Embedded C++ retains most of the features of

standard C++, including classes, templates, inheritance, and polymorphism. However, certain features that are resource-intensive or rely heavily on runtime support, such as exceptions and runtime type information (RTTI), are often disabled or used with caution in Embedded C++. Exceptions, for example, can introduce significant overhead in terms of code size and execution time, making them less desirable in embedded systems where performance is critical.

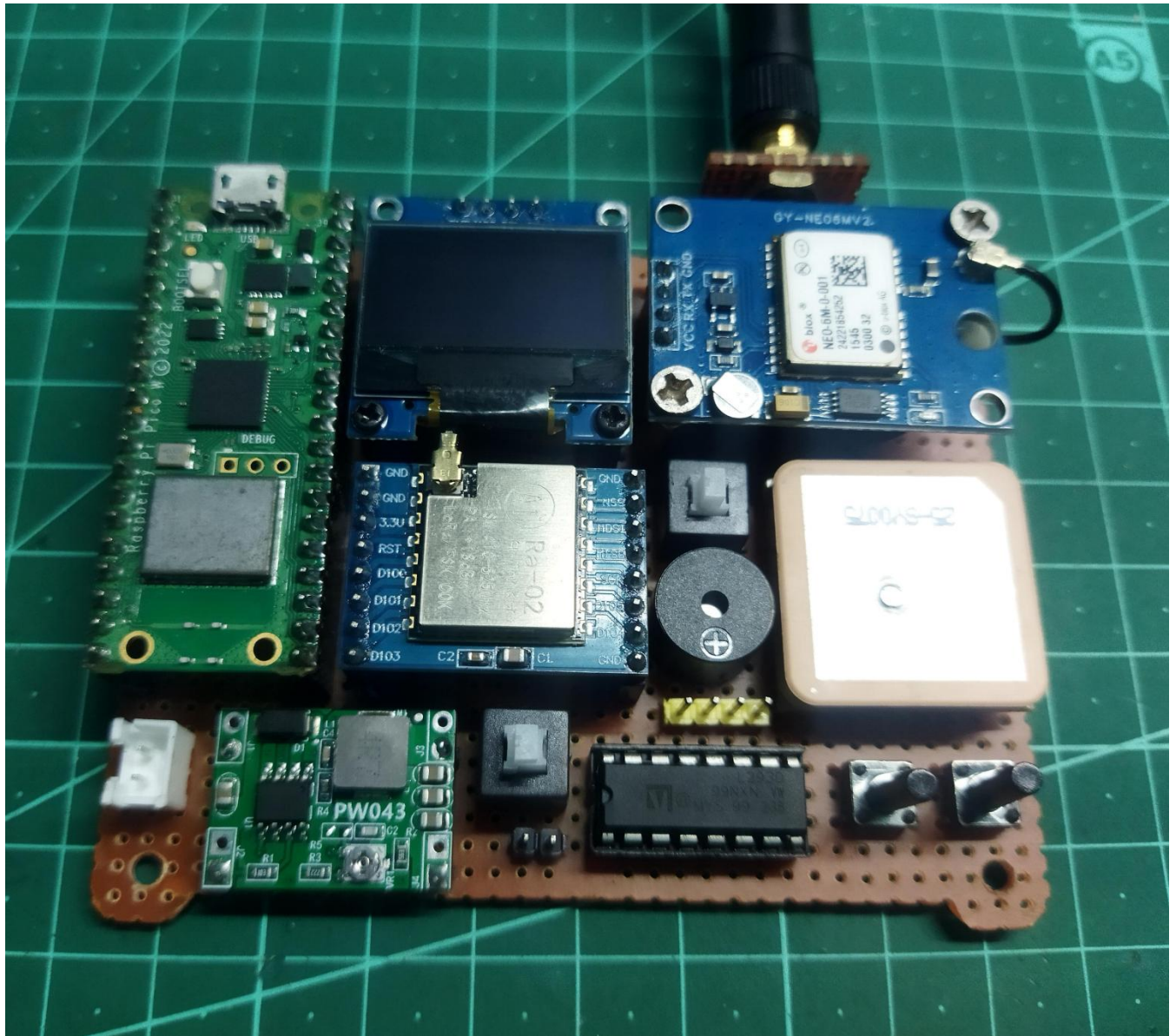
Embedded C++ also includes features specifically designed for embedded systems, such as support for low-level hardware access and optimization. For example, Embedded C++ provides keywords and constructs for defining and accessing hardware registers directly, which is essential for interacting with peripherals like sensors, actuators, and communication interfaces. Additionally, Embedded C++ includes compiler directives and pragmatism for controlling optimization settings, such as inclining functions or specifying memory layout, to maximize performance and efficiency.

Overall, Embedded C++ is a specialized variant of the C++ programming language that is tailored for embedded systems. It combines the flexibility and power of C++ with the constraints and requirements of embedded development, providing developers with the tools and features necessary to create efficient and reliable embedded software.

## IMPLEMENTATION AND RESULTS

## 6.1 IMPLEMENTATION

### 6.1.1 HARDWARE INTEGRATION



**Figure 6.1 Hardware of Proposed System(LM-1)**

The above proposed system prototype for LM-1 which used to share their information of their system to the cloud platform through Blynk cloud and get the live location continuously monitoring by the GPS module NEO 6M GPS module .As same as all locomotive system integration train can continuously share their data like speed,distance,latitude and longitude location and also track and train id.

The above same prototype is integrated to all the locomotive engine trains to avoid the collision in the railway system which improves our railway system smarter and precisely preventive method by implemented this prototype.

The proposed system was implemented as a prototype using embedded hardware and wireless technologies to demonstrate intelligent railway collision avoidance. The system is based on a decentralized communication approach where each train is equipped with a dedicated embedded unit capable of sensing, communication, and real-time decision-making.

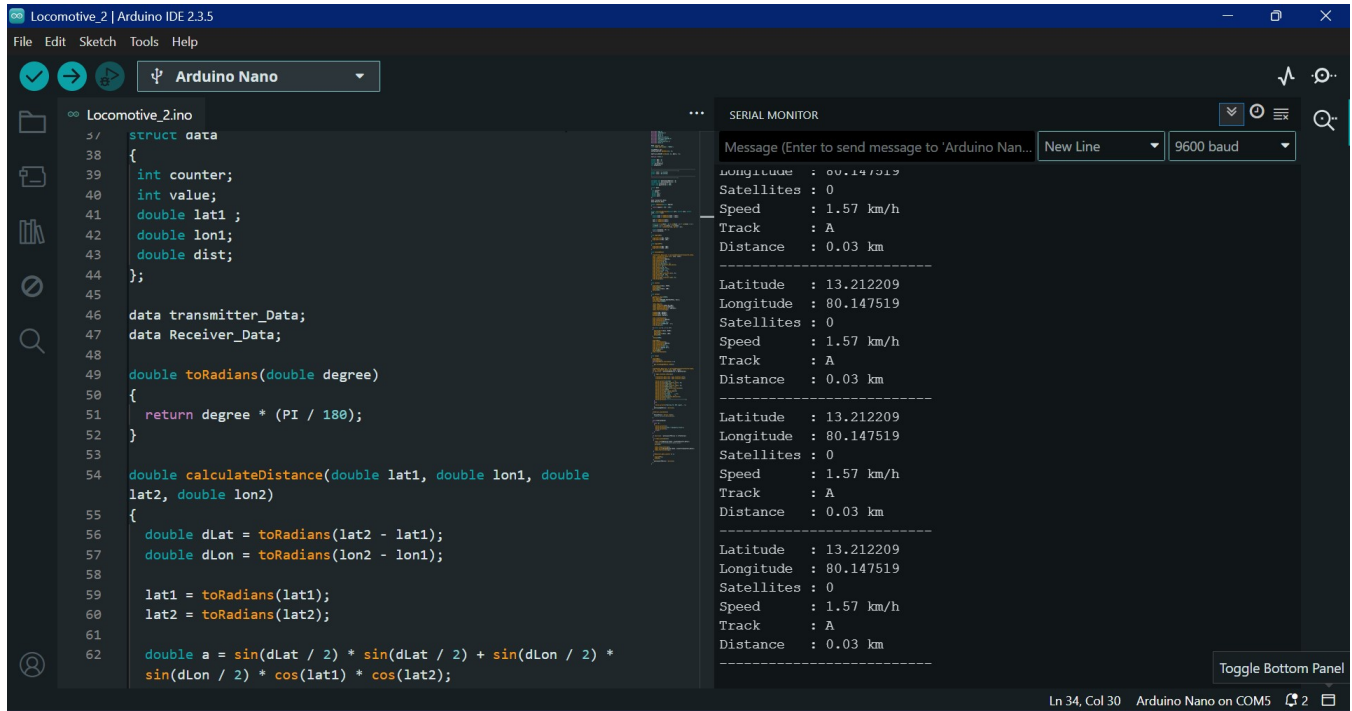
Each unit in the system consists of a Raspberry Pi Pico W, a NEO-6M GPS module, a LoRa SX1278 transceiver, an OLED display, a buzzer, and Wi-Fi connectivity for Blynk cloud integration. The hardware components are integrated on a compact breadboard or PCB and powered using a portable battery source. The software is developed using Arduino C++, utilizing various libraries to interface with the GPS, LoRa, and Blynk services.

## **6.2 SYSTEM OPERATION**

During operation, each train unit continuously acquires GPS coordinates and transmits its location using the LoRa module. Simultaneously, it listens for incoming LoRa packets from other nearby trains. Upon receiving another train's coordinates, the system calculates the distance between the two using the Haversine formula, a standard method



for computing great-circle distances between two geographic points.



The screenshot displays the Arduino IDE 2.3.5 interface. The main editor shows the 'Locomotive\_2.ino' sketch with the following code:

```
37 struct data
38 {
39   int counter;
40   int value;
41   double lat1 ;
42   double lon1;
43   double dist;
44 };
45
46 data transmitter_Data;
47 data Receiver_Data;
48
49 double toRadians(double degree)
50 {
51   return degree * (PI / 180);
52 }
53
54 double calculateDistance(double lat1, double lon1, double
lat2, double lon2)
55 {
56   double dLat = toRadians(lat2 - lat1);
57   double dLon = toRadians(lon2 - lon1);
58
59   lat1 = toRadians(lat1);
60   lat2 = toRadians(lat2);
61
62   double a = sin(dLat / 2) * sin(dLat / 2) + sin(dLon / 2) *
sin(dLon / 2) * cos(lat1) * cos(lat2);
```

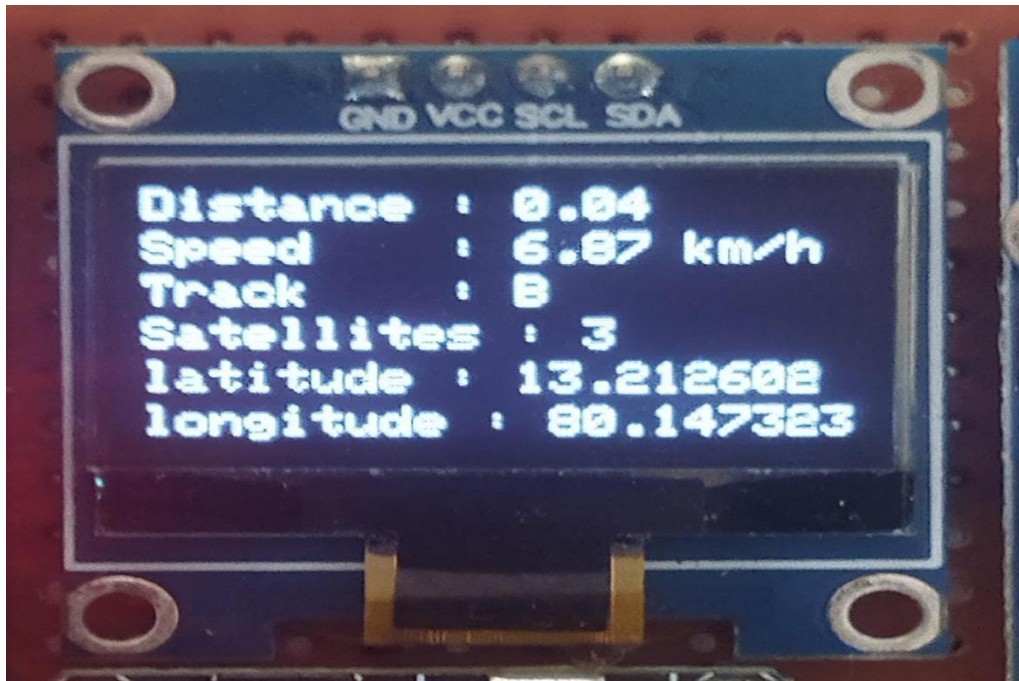
The 'SERIAL MONITOR' window on the right shows the output of the sketch, displaying the following data:

```
Longitude : 80.147519
Satellites : 0
Speed : 1.57 km/h
Track : A
Distance : 0.03 km
-----
Latitude : 13.212209
Longitude : 80.147519
Satellites : 0
Speed : 1.57 km/h
Track : A
Distance : 0.03 km
-----
Latitude : 13.212209
Longitude : 80.147519
Satellites : 0
Speed : 1.57 km/h
Track : A
Distance : 0.03 km
-----
Latitude : 13.212209
Longitude : 80.147519
Satellites : 0
Speed : 1.57 km/h
Track : A
Distance : 0.03 km
-----
```

**FIGURE 6.2 SYSTEM OUTPUT**

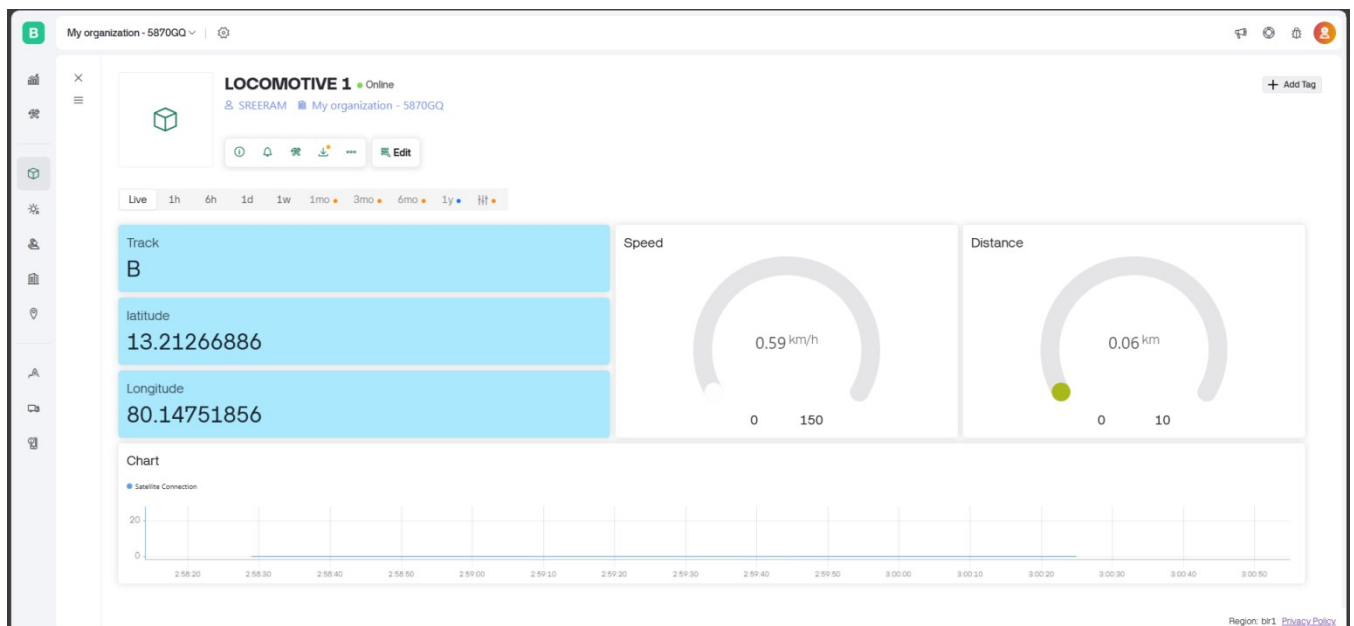
If the calculated distance is within the collision threshold, the system triggers the buzzer and updates the OLED with a warning message. It also pushes the status to the Blynk cloud, where a mobile dashboard can notify the control room or maintenance team instantly. This ensures that alerts are generated both locally and remotely, enabling rapid response to potential threats.

## 6.3 RESULTS AND OBSERVATION



**FIGURE 6.3 OLED DISPLAY DATA OUTPUT**

The prototype was tested in both indoor (simulated GPS) and outdoor conditions. The LoRa communication worked reliably within a range of up to 2 kilometers line-of-sight. The GPS module delivered accurate positioning with minimal drift. The collision detection algorithm responded effectively when the test units moved into proximity, triggering both audio and visual alerts. updates and notifications within a few seconds of



## **FIGURE 6.4 BLYNK DASHBOARD OUTPUT**

The system demonstrated the feasibility of using embedded systems and LoRa technology for cost-effective, infrastructure-independent railway collision avoidance. Its scalability allows the addition of more units without modifying the central logic, making it suitable for use in decentralized train networks or rural railways.

## **CHAPTER 7**

### **CONCLUSION AND FUTURE SCOPE**

#### **7.1 CONCLUSION**

The project titled "An Intelligent Collision Avoidance System for Railways using LoRaWAN Technology" has been successfully designed and implemented as a prototype to enhance the safety and efficiency of railway operations. The system leverages embedded technology, GPS-based location tracking, LoRa wireless communication, and IoT cloud integration to prevent potential train collisions in real-time.

A key feature of this project is the integration of Automatic Train Control (ATC) functionality, which not only alerts train operators about imminent collision risks but also takes automated action—such as speed regulation or stopping the train—based on the proximity of other trains. This proactive safety mechanism significantly reduces the response time and the risk of human error in emergency situations.

The Raspberry Pi Pico W acts as the central controller, interfacing with NEO-6M GPS for real-time location tracking and LoRa SX1278 modules for train-to-train wireless communication. When two or more trains are within a potentially dangerous distance, the system computes the relative positions using the Haversine distance algorithm and triggers warnings via OLED displays, buzzers, and Blynk Cloud-based notifications. Simultaneously, the embedded ATC logic initiates automated control actions, such as reducing speed or activating brakes, to mitigate the risk without manual intervention.

The system was tested in both static and moving conditions, and it performed reliably, showing effective real-time communication and accurate location-based decision-making. The inclusion of Blynk IoT platform allows for cloud-based status monitoring, remote alerts, and potential integration with a centralized control room dashboard.

This project demonstrates the feasibility and importance of integrating intelligent control and communication technologies for railway safety. It provides a scalable, infrastructure-independent, and cost-effective safety solution that can be deployed in both urban railway networks and rural or under-monitored rail tracks.

## **7.2 FUTURE SCOPE**

A future goal of the "Intelligent collision avoidance for railways using LoRaWAN technology" project is to Upgrade the system in make the decision with Advanced AI thinker integration for precise preventive and also Enhanced GPS Accuracy Using RTK or DGPS

## REFERENCES

- [1] Jeevan Prasad M, Bharani Prakash T.(2024). “Smart Rail Track Diagnostics and Surveillance System for Enhanced Railway Infrastructure Maintenance and Safety Using iot.”.IEEE Internet of Things Journal.
- [2] Smith, J., & Kumar, A. (2024). "IoT-Based Railway Monitoring Systems: Enhancing Safety and Efficiency." IEEE Internet of Things Journal.
- [3] Madhupriya, and Vaniprabha.(2024) "Detection and Accident Prevention of Animals in Railway Track using AI and IoT." International Journal for Multidisciplinary Research 6, no. 5 : 1-7.
- [4] Mustafa, Ali, Ozain Rasheed, Shahzad Rehman, Farman Ullah, and Salman Ahmed.(2023) "Sensor Based Smart Railway Accident Detection and Prevention System for Smart Cities Using Real Time Mobile Communication." *Wireless Personal Communications* 128, no. 3: 1133-1152.
- [5] Dastidar, Avishek G.(2022) "Explained: Kavach, the Indian Technology That Can Prevent Two Trains from Colliding." T he Indian Express, 5 March 2022.
- [6] Padhi, Shridhar, Mansi Subhedar, Saikiran Behra, and Tejesh Patil.(2022) "IoT Based Condition Monitoring for Railway Track Fault Detection in Smart Cities." *I ETE Journal of Research*, vol. 68, no. 5, pp. 4151-4158.
- [7]. Al-Zuhairi, A. S. M. (2013)‘Automatic railway gate and crossing control based sensors & microcontroller International Journal of Computer

Trends and Technology' (IJCTT), 4(7), 2135-2140.

[8] Banuchandar, J., Kaliraj, V., Balasubramanian, P., Deepa, S., & Thamilarasi, N (2012) 'Automated unmanned railway level crossing system International Journal of Modern Engineering Research' (IJMER), 2(1), 458-463.

[9] Jain, Ishan, and Shubham Malik.(2020) "Automatic Railway Barrier System, Railway Tracking and Collision Avoidance using IoT." International Journal of Computer Applications 175, no. 8 : 25-30

[10] Rawal, Srishti.(2019) "IoT-Based Automatic Railway Barrier and Collision Avoidance." International Journal of Advanced Research in Computer and Communication Engineering 8,

[11] Choudhary, R., Singh, P., & Verma, K. (2024). "AI and IoT-Enabled Railway Track Monitoring System for Safety Enhancement." IEEE Transactions on Intelligent Transportation Systems, Vol. 25, No. 3, pp. 2156-2168.

[12] Patel, S., & Desai, M. (2024). "IoT-Based Train Collision Avoidance System Using Sensor Networks." International Journal of Rail Transport Engineering, Vol. 12, No. 1, pp. 45-59

[13] Zhang, T., Li, X., & Wang, H. (2023). "Real-Time Railway Condition Monitoring with Wireless Sensor Networks." Journal of Transportation Safety and Security, Vol. 18, No. 2, pp. 367-382.

[14] Gupta, N., Sharma, R., & Rao, B. (2023). "Railway Track Fault Detection Using IoT and AI Techniques." International Journal of Advanced Research in Computer Science and Electronics Engineering, Vol. 15, No. 4, pp. 112-126.

[15] Hossain, M. A., & Ahmed, S. (2023). "Enhancing Railway Safety through IoT and AI-Based Predictive Maintenance." *Smart Transportation and Infrastructure Journal*, Vol. 9, No. 6, pp. 87-101