Development of Underground Mine Monitoring and Communication System Based on Wireless Sensor Networks

BY

Mazimba Clement

A dissertation submitted to the University of Zambia in partial fulfilment of the requirements for the award of the degree of Master's Degree of Mineral Sciences in Mining Engineering

The University of Zambia
Lusaka

September 2023.

Declaration

I, **Clement Mazimba** declare that to the best of my knowledge and belief, this thesis contains no material previously published by any other person except where due acknowledgement has been made. I further declare that I am the sole author of this thesis and that this thesis contains no material that has been accepted for the award of any other degree or diploma in any other University.

Signature: ……. **Date: …….**

CLEMENT MAZIMBA

Acknowledgements

I would like to thank Josh Goveia for the financial assistance that made my pursuing this degree possible.

I would also like to thank Francis Mpolokoso and the entire team at PentaTech Limited for making it easier and possible to build the prototypes that were used to conduct the research and for providing a bountiful amount of other resources that made it easier to conclude most of the preliminary work.

And lastly but not least, my supervisors, Dr. Besa Bunda and Prof. E.K. Chanda, and all the Lecturers in the Department of Mining Engineering at the School of Mines, for their insightful contribution and advice during the development and testing of this project.

Approval

This thesis of Clement Mazimba is approved as fulfilling the requirements of the Degree of Master of Engineering in Mining Engineering at the University of Zambia.

Signature:	Supervisor	……
	1st Examiner	……
	2nd Examiner	……
	External Examiner	……
	Chairperson	……

Dedication

This work is dedicated to my wife and son. Thanks for being the awesome bunch that you are.

Table of Acronyms

Acronym	Description
IoT	Internet of Things
WSN	Wireless Sensor Network
UWB	ultra-wideband
GIS	Geographical Information System
RSSI	Received Signal Strength Indicator
LoRa	Long Range
GPIO	General Purpose Input Out
UART	Universal asynchronous receiver-transmitter
MCU	Microcontroller unit
Wi-Fi	Wireless Fidelity
IDE	Integrated Development Environment
SF	Spreading Factor
SoC	System on Chip
PCB	Printed Circuit Board
SPI	Serial Peripheral Interface
SDIO	Secure Digital Input Output
I2C	Inter-Integrated Circuit
OLED	Organic Light Emitting Diode
LED	Light Emitting Diode
RF	Radio Frequency
Li-Po	Lithium-Polymer
ULP	Ultra Low Power coprocessor

M2M	Machine to Machine
BLE	Bluetooth Low Energy
MB	MegaBytes
MHz	Mega Hertz
kHz	Kilo Hertz
ESD	Electrostatic Discharges
SNR	Signal-to-noise ratio
MAC	Media Access Control Address
FM	Frequency Modulation
AM	Amplitude Modulation
SoftAP	Software-enabled Access Point
ISM	Industrial, Scientific and Medical radio bands
HMI	Human-Machine Interface

Abstract

Historically, mining has been an enormous contributor to the development and sustainability of several societies. The industry despite being a source of wealth is not without risks and challenges, including environmental and operational hazards. These pose a huge challenge to the safety and health of the workers and equipment and result in huge losses in terms of human capital, machinery, operational time, and valuable infrastructure. Accidents that occur in mines can be minor or significant depending on the nature of operations, and safety measures that have been put in place. The underground environment is particularly more challenging than an open pit environment due to the need for a controlled environment for mining operations to occur.

Controlling the environment requires constant monitoring of parameters that make it safe for humans and machinery to work effectively and safely. To enhance safety and mitigate risks in underground mining, this study proposes an integrated, real-time monitoring and communication system that utilizes advanced sensing technologies and robust communication networks based on reliable protocols.

The system leverages a wireless sensor network (WSN) strategically deployed and placed across the mine to track and report key environmental parameters. These sensors continuously collect and transmit real-time data using a combination of Internet of Things(IoT) based sensing devices, a designed prediction algorithm for every sensing node detecting a designated anomaly, and secure communication protocols. The transmitted data is processed at the designed gateway, located at the central hub, and further sent to an IoT platform where deviations from optimal conditions are identified, triggering automated alerts and emergency response mechanisms.

In this study, temperature and humidity were selected as two of the parameters to be used particularly to demonstrate the sensing and communication capabilities of the WSN. A DHT11 sensor was used and combined with an ESP32 microcontroller unit (MCU), which was programmed in C++, to form a single sensing node. This node was programmed to collect and transmit data to the transceiver that relayed it along with other transceivers at an interval of 10 seconds. The system used two communication protocols, ESPNow, and Long Range (LoRa), to transmit the data from the nodes to the transceiver, and from one transceiver to the next respectively. The combination proved effective in that ESPNow provided immediate and real-time

updates in localized areas while LoRa transmitted the data reliably over considerable distances without dropping the data integrity.

The results of preliminary simulations indicate that the proposed system significantly improves hazard detection accuracy and reduces response times to potential threats. Compared to traditional monitoring methods, the system enhances situational awareness, leading to a measurable reduction in mining-related accidents and improved operational efficiency. Additionally, real-time alerts—delivered via mobile applications, email notifications, and on-site alarms—enable rapid response, allowing for immediate evacuation and risk containment.

The findings underscore the importance of integrating digital safety solutions in underground mining operations. This study contributes to broader efforts in improving mining safety standards, aligning with industry best practices and regulatory requirements. Future research should explore further optimization using artificial intelligence for predictive risk assessment, as well as the scalability of the system to incorporate various types of sensors to get a more holistic view of the operational environment.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

1.1 Overview

The mining environment can be a hazardous place to work both for the labour force and the equipment used. Several accidents occur mostly as a result of human error. Others are as a result of a lack of proper mechanisms to monitor the environment and account for rapid changes in the parameters that could pose a threat to miners and operations as a whole.

This research focuses on developing a system for monitoring and reporting critical environmental parameters in the underground mine by employing a wireless sensor network.

An array of sensors operating in real or near-real-time are part of a network to improve mine safety and drive productivity gains in the mining industry. These are linked to the above-ground application that ties all these for monitoring and decision-making.

1.2 Background

In general, underground mining operations tend to experience more safety challenges compared to surface mining operations. The nature of the operational environment dictates that several measures be put in place to make the place as ideal as possible for both the human workforce and the machinery that is operated there. Several parameters need to be actively monitored and maintained at acceptable levels for the safety of the operation. Parameters such as temperature, humidity, gas emissions, and ground stability just to mention a few.

These parameters may rapidly change due to unforeseen events, human error, or machine failure. A failure to have adequate communication and monitoring of the changes in environmental

conditions and potential hazards can lead to safety issues arising which may, in turn, result in preventable fatalities.

Traditional safety measures frequently rely on routine manual inspections, which have a limited capacity to offer real-time insights into dynamic and changing conditions inside mines. Additionally, difficulties with communication in underground environments prevent the quick broadcast of vital information during emergencies, delaying prompt actions and escalating the effects of accidents, should they occur.

The occurrence of accidents in underground mining operations on the Zambian Copperbelt was investigated and documented as illustrated in Figure 1.1 below.

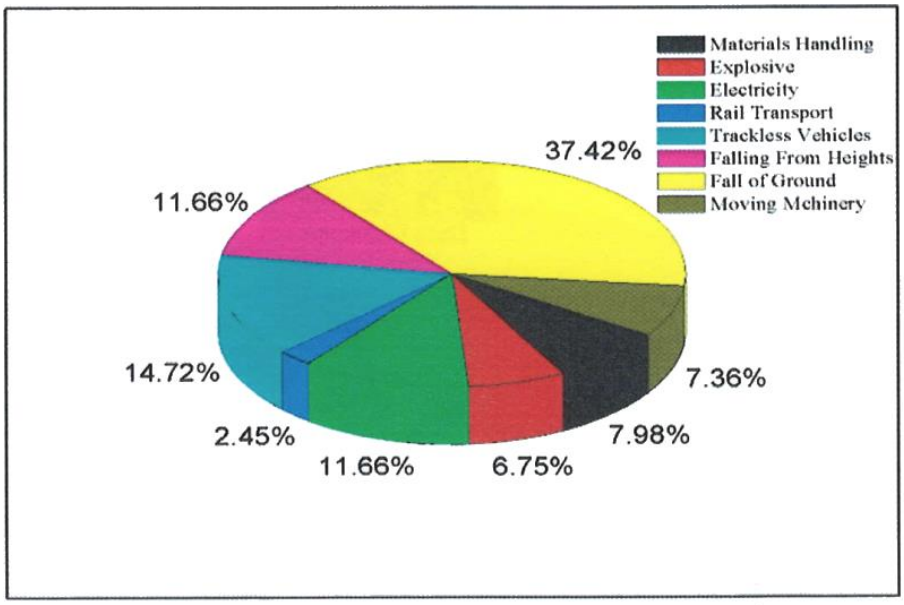

Figure 1.1 Major accidents on the Zambian Copperbelt Mines spanning the period 2006 to 2017, (Besa and Mutambo, 2018)

This study aims to develop a proactive and responsive safety system by utilizing developments in sensor technology, data processing, and communication protocols. The system attempts to detect departures from typical circumstances and issue early warnings to miners and supervisors by continually monitoring crucial factors such as temperature, humidity, and other vital environmental variables and by analyzing the collected data in real time. In order to build trustworthy and dependable communication channels that get beyond the drawbacks of traditional

approaches, the study also investigates reliable communication protocols like LoRa, ESPNow, and Zigbee.

The end result of this study is to have a reliable sensor network system that can contribute to the reduction in the number of mining accidents, the severity of these accidents should they occur, and to have reliable data that can be used in future to improve the well-being of the miners, provide protection to the vast equipment investments that are made, and serve as a template for safety advancements in mining operations.

1.3 Statement of the Problem

Underground mining operations present significant safety and operational challenges due to harsh and unpredictable conditions, (Tubis et. al, 2020). Workers are frequently exposed to hazardous gases, extreme temperatures, humidity fluctuations, and structural instabilities, all of which increase the risk of accidents. Despite advancements in mining safety protocols, existing monitoring systems often rely on periodic manual checks or outdated sensors that provide delayed or incomplete data. This lag in hazard detection compromises the ability of miners and supervisors to respond swiftly, resulting in increased fatalities, injuries, equipment damage, and productivity losses, (Wozniakowski-Zehenter, 2023).

Current monitoring systems also struggle with real-time data transmission, efficient hazard detection, and timely response mechanisms. Traditional safety approaches lack integration with modern digital technologies such as wireless sensor networks (WSNs), Internet of Things (IoT)-based monitoring, and algorithm-driven anomaly detection. Without these advancements, mines continue to face preventable accidents that could be mitigated through improved early warning systems and automated risk assessment.

Several technologies have been developed and designed to help monitor and communicate the potential risks that arise from underground mining operations. Cabled monitoring systems that are connected from end to end such as the hardened phone cable networks used in Zambian mines and the optical fibre sensor(Sato, Honda, and Shibata, 1999.), and Multielectromechanical Systems, (MEMS), (Dasenbrock, 2010.), such as photogrammetry and Wireless Sensor Networks, (WSNs).

This study aims to address the challenges brought about by traditional approaches to safety by developing an integrated real-time monitoring system that enhances hazard detection and emergency response in underground mines. By leveraging advanced sensing technologies, and wireless communication protocols, the proposed system seeks to bridge the gap between traditional safety measures and modern digital solutions. The implementation of this system is expected to significantly reduce mining accidents, improve worker safety, and optimize operational efficiency, thereby contributing to the broader goal of enhancing mining safety standards.

1.4 Purpose of the Study

The purpose of the study is to design and develop a wireless sensor network that can monitor and communicate environmental parameters in underground mines. The system will be the one that combines cutting-edge sensor networks, real-time data processing, and effective communication protocols to reduce the risk of mine accidents.

1.5 Study Objectives

The study was guided by the following objectives:

1. To create a wireless sensor network that will be used to monitor environmental temperatures and humidity in underground mines
2. To test the limitations of transmitting the collected data and sending it over vast distances in near-real time
3. To determine an ideal network topology for reliable data transmission by the system
4. To develop a monitoring and data interpretation interface

1.6 Research Questions

The research questions that guided the study are:

1. Which network topology will best work in the set-up of the wireless network?
2. Which transmission/communication pathway would be ideal to move the sensor data from underground to the surface while maintaining data integrity?
3. What microcontrollers would best be suitable for the type of WSN that can combine several communication protocols?

4. Which web interface would be more cost-effective for a sensor network?
5. How will the sensor nodes be powered to enable remote deployment of the sensor nodes?

1.7 Significance of the study

In this study, therefore, a wireless sensor network and communications software system are to be developed for underground mines to monitor and report tunnel temperature and humidity. The significance of this work is as follows:

- Real-time monitoring of critical variables can aid in evacuating an unsafe zone
- The prototype created in the study can further be expanded upon to monitor additional parameters like gas pockets, roof falls, and subsidence to mention a few
- Identifying cost-effective communication protocols that can be used and deployed underground for more rigorous WSNs
- Improvement of communication methods between the miners underground and the control room on the surface.

This project seeks to greatly improve mine safety, decrease human and economic losses, and give a template for safety enhancement in difficult and dangerous mining environments by developing a holistic solution that can further improve the quality of information made available to key decision-makers.

1.8 Scope of the Study

This study seeks to enhance communication and environmental monitoring in underground environments by the placement of various sensors that will form a simple network. This wireless sensor network will help in monitoring near real-time changes in the mining environment and then communicate those changes to the central hub, which can then aid in decision-making that can save the lives of miners and protect equipment deployed in the areas where the system is being used. The experiments in this study were set to study temperature and humidity variations over a period of time and to successfully transmit the data from the sensor nodes to a central location in real-time.

1.9 Thesis Organisation

This thesis is organized into the following chapters:

1. Introduction:
    - Briefly introduce the research topic and its significance.
    - Outlines the objectives and research questions of the study.
    - Provides an overview of the structure of the thesis.

2. Literature Review:
    - Review existing literature on wireless sensor networks, their benefits, and challenges.
    - Review current systems used on the Zambian Copperbelt
    - Discusses the current state of wireless sensor networks and communication protocols.
    - Identifies gaps in the literature that this study aims to address.

3. Methodology:
    - Describes the research design,
    - Details the design and development of the wireless sensor network,
    - Describes the designed experiments and data collection methods.

4. Communication Protocols:
    - Explores various communication protocols, ZigBee, LoRa, ESPNow, WiFi, and Bluetooth.
    - Presents case studies of real-world installations and their performance metrics.

5. Discussion and Conclusion:
    - Discussed the results of the research, and the designed network's capabilities.
    - Offers insights into the broader implications of underground WSN adoption.
    - Concludes by summarizing the main findings and suggesting areas for future research.

CHAPTER 2

LITERATURE REVIEW

2.1 Review of existing networks and previous studies

2.1.1 Zambian Underground Monitoring and Communication Systems

In Zambian underground mines, the communication system that is used is the hardened telephone system coupled with the Locked-Bell System, according to the Mines Safety Department. (Besa, et.al, 2018)

2.1.1.1 The Locked-Bell System

The Locked-Bell System is a system of signalling to a winding engine driver which cannot be operated unless a special key, known as the “The Key to the Locked-Bell”, remains inserted in the system switch in use at the time. The system comprises different types or tones of signals that are used so as to be easily distinguished without doubt by the winding engine driver between the signals received from the bank and those received from below the bank.

The signal operating mechanism of this system at the bank and at points below the bank is of a type that is securely enclosed in a metal case of substantial construction and is kept locked when not in actual use, and the key retained by the banksman, onsetter, cage tender or other authorized personnel. (Zambian Mining Regulations)

2.1.1.1 Hardened Phone Cable Network

According to the Zambian Mining Regulations, Regulation 1919(1), where there are more than thirty (30) persons employed underground, there shall be provided and maintained an effective

means of telephonic communication or other equivalent means of transmitting speech between a point on the surface and such points underground as may be necessary for safety and within reasonable and easy access from any place which work is being carried out.

These phones are installed in each section, in workshops and at the shaft stations. Despite the installation of these systems in the mines, there is limited access as they are located away from the work areas as development progresses away from the shaft and section stations. (Zambian Mining Regulations)

This study seeks to build on several of the works done by other scholars by proving a much more efficient way of sensor data collection and transmission. The study seeks to use communication protocols that can easily and reliably transmit data between nodes at least 500m apart with little to no data loss or corruption.

2.2 Wireless Sensor Networks

A wireless sensor network, (WSN) is a distributed network that comprises low-powered devices and sensors. It covers self-powered or battery-operated, embedded devices that are networked together with the purpose of collecting data via sensors, processing using a microcontroller or small computing devices, and transmitting data using energy-efficient means.

A WSN is built of nodes whose number varies from a few to thousands. Each node is connected to other sensors and typically has several parts including a transceiver, a microcontroller, an electric circuit for interacting with the sensors and an energy source, as shown in Figure 2.1 below.

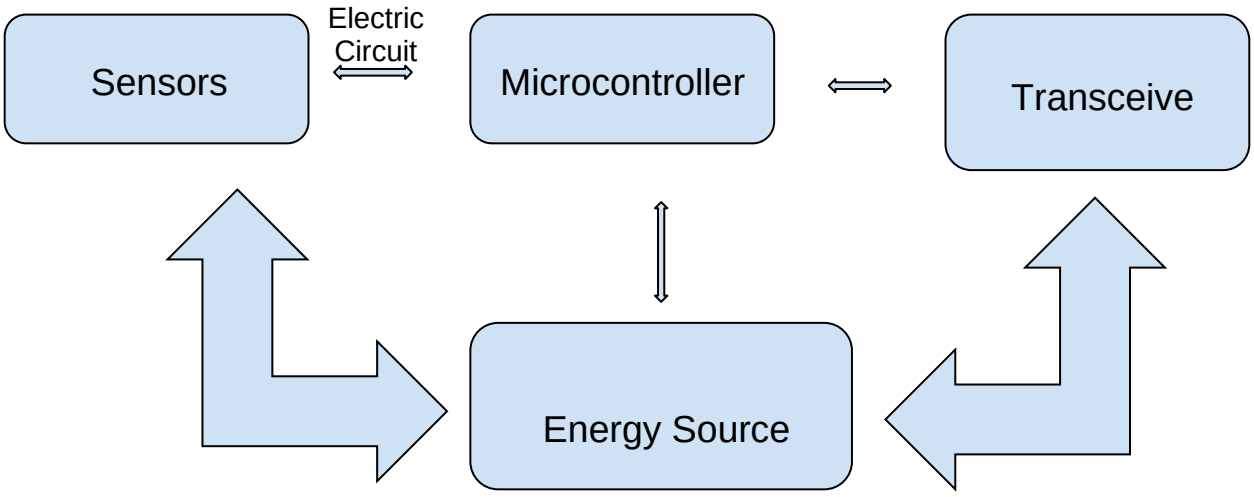

Figure 2.1: Basic Sensor Node

The sensor unit of the node senses the changes in the parameters being monitored, the microcontroller unit computes and processes the confined permutations of the sensed data and the transceiver then communicates the processed information among nodes up to the central hub or gateway. The energy source provides power to all the various components of the node.

In general terms, WSNs are a form of ad hoc wireless networks whose connectivity and set-up heavily rely on the positioning and wireless connectivity of sensors, (Mario, 2011). The network topology of a WSN can vary depending on the complexity of the network and the area being monitored or studied.

2.1.2 An overview of existing systems Wireless Network Systems

Various types of wireless sensor networks exist with different applications and ranges. Some of the most notable include:

2.1.2.1 Habitat/Environmental Monitoring

Wireless sensor networks (WSNs) have emerged as powerful tools for monitoring diverse habitats and understanding environmental changes (Akyildiz et al., 2002). These networks consist of

numerous sensor nodes strategically deployed to collect data on various environmental parameters. The deployment of these nodes can follow two primary approaches: unstructured (ad hoc) and structured (pre-planned). In unstructured (ad hoc) deployment, the sensor nodes are randomly scattered within the target area, often employed in situations where precise placement is challenging or unnecessary, allowing for rapid setup and flexibility (Ullo, 2020). This approach is useful for large-scale deployments where manual placement is impractical. Conversely, in structured (pre-planned) deployment, the nodes are positioned in a grid-like or optimized pattern, ensuring comprehensive coverage and efficient data collection (Yick et al., 2008). This approach is favoured when specific data points or spatial relationships are crucial, such as in precision agriculture.

The real-world applications of WSNs in habitat and environmental monitoring are diverse and impactful. In forest fire detection, WSNs equipped with temperature, smoke, and humidity sensors can identify early signs of fire, enabling swift responses and minimizing damage (Werner-Allen et al., 2005). Early detection is crucial for effective firefighting efforts. In wildlife monitoring, WSNs track animal movement, behaviour, and population dynamics, providing valuable insights into their ecological roles and conservation needs (Mainwaring et al., 2002). This data helps researchers understand migration patterns, habitat use, and the impact of environmental changes on wildlife populations. In precision agriculture, WSNs monitor soil conditions, weather patterns, and crop health, optimizing irrigation, fertilization, and pest control practices to enhance agricultural sustainability (Li et al., 2010). This leads to increased efficiency and reduced resource waste. Additionally, WSNs play a vital role in pollution monitoring by detecting and measuring air and water pollutants, allowing authorities to identify sources, assess risks, and implement mitigation strategies (Kumar et al., 2019). Real-time monitoring enables rapid responses to pollution events, ultimately contributing to environmental protection and public health.

2.1.2.2 Human Health Monitoring

WSNs have revolutionized healthcare by enabling continuous, remote monitoring of patient's physiological data (Pantelopoulos, 2016). This technology has led to the development of various medical applications, including wearable sensors that track vital signs, physical activity, and sleep

patterns, empowering individuals to manage their health proactively. Smartwatches and fitness trackers are prime examples of this technology. Additionally, WSNs facilitate the telemonitoring of patient physiology, allowing remote monitoring of individuals with chronic conditions so that healthcare providers can track progress and intervene when necessary (Rashid et al., 2016). This is particularly beneficial for patients in remote areas or those with limited mobility. Furthermore, WSNs support individuals with disabilities by providing environmental control, fall detection, and other assistive functionalities (Zhu et al., 2010), significantly improving independence and quality of life.

Real-world applications of WSNs in human health monitoring are becoming increasingly sophisticated. Smartwatches continuously monitor heart rate, activity levels, and sleep quality, providing users with personalized health insights. Implantable sensors, such as continuous glucose monitors, allow diabetic patients to track glucose levels in real-time for better insulin management. Remote patient monitoring systems utilize WSNs to enable healthcare providers to track vital signs and activity levels of patients recovering at home, ensuring timely intervention if needed. These advancements demonstrate the potential of WSNs to improve healthcare outcomes, enhance patient autonomy, and reduce the burden on healthcare facilities.

2.1.2.3 Underground WSNs

Underground WSNs are a specialized type of wireless sensor network that mainly focuses on the use of sensors in the subsurface region of the earth's crust. They are envisioned to provide real-time monitoring capabilities in complex underground environments, (Akkas, 2015). According to Wang et al (2020), the common WSNs for monitoring and communication system in underground mining is mainly comprised of Bluetooth technology, ultra-wideband (UWB) technology, Wi-Fi technology and ZigBee technology.

Research in designing and identifying the most reliable and stable wireless communication technology is an ongoing endeavour. Chanda et.al (2014) designed and conducted an investigation on radio wave attenuation using ZigBee at the Angas Zinc Mine in Australia. Their study compared

simulated radio waves and the actual data that was collected in straight and curved underground tunnels by measuring the RSSI between each ZigBee node. The results of the study indicated a stable signal attenuation in straight-line tunnels but a sharp decline in curved tunnels and junctions of branches.

Moridi et al. (2015) designed an underground mine monitoring and communication system that integrated ZigBee and GIS. The system used ZigBee nodes to collect Carbon Dioxide concentrations and ArcGIS to provide geospatial data detailing the map of the mine and areas of both safe and unsafe concentration. The implementation of the design was simulated and proved to improve mine safety significantly.

At China's CoreCast Corporation Limited, a system designed and developed by Tao and Xiaoyang(2011), was tested and implemented. The WSN system which has been adopted by several other firms since combines the low latency of ZigBee technology and high data transfer rates of WiFi to monitor gas, provide wireless communication, personnel management, video surveillance and a disaster prevention system.

Taking advantage of reliable data modelling with adequate data, ByungWan and Khan, (2018), developed an air quality pollution prediction system that was using the Azure Machine Learning Studio. The system comprised of sensor modules, a communication protocol and a base station for data analysis and prediction modelling. The selected protocol of choice was the ZigBee wireless communication protocol while the microcontroller used in the system was the Arduino Uno. Upon installation and testing, this system was able to reliably transmit data, accurately predict mine environment variables and recall readings for over a period of one month.

2.1.3 Microcontroller Unit (MCU) Types and Capabilities

Different types of microcontroller units(MCUs) exit with different capabilities, functions and use cases. MCUs are complex, compact integrated circuits that control specific operations in systems. They interpret digital and analogue data that they receive from input/output peripherals, process it and then provide a controlled output through other peripherals to communicate and enact an appropriate action (Lutkevich, 2019).

Four(4), of the most commonly used ones in projects are the ESP32, ESP8266, Arduino Uno, and the Raspberry Pi.

2.1.3.1 ESP32

The ESP32 is a series of low-cost and low-power System-on-a-chip (SoC) microcontrollers developed by Espressif that include Wi-Fi and Bluetooth wireless capabilities and a dual-core processor.

ESP32 is capable of functioning reliably in industrial environments, with an operating temperature ranging from –40°C to +125°C. Powered by advanced calibration circuitries, ESP32 can dynamically remove external circuit imperfections and adapt to changes in external conditions. The microcontroller achieves ultra-low power consumption with a combination of several types of proprietary software. The ESP32 supports a wide variety of input (read data from the outside world) and output (to send commands/signals to the outside world) peripherals making it an ideal MCU for IoT and WSN projects.

The ESP32 board which was used in this study is the Heltec LoRa 32 Board. Heltec WiFi LoRa 32 V2 has Wi-Fi, BLE, LoRa functions, and also a Li-Po battery management system. Its features include the following:

- Microprocessor: ESP32 (dual-core 32-bit MCU + ULP core)
- LoRa node chip SX1276/SX1278
- Micro USB interface with a complete voltage regulator, ESD protection, short circuit protection, RF shielding, and other protection measures
- Onboard SH1.25-2 battery interface, integrated lithium battery management system
- Integrated WiFi, LoRa, Bluetooth three network connections, onboard Wi-Fi, Bluetooth dedicated 2.4GHz metal 3D antenna, reserved IPEX (U.FL) interface for LoRa use
- Onboard 0.96-inch 128*64 dot matrix OLED display
- Integrated CP2102 USB to serial port chip

Figure 2.2 shows the Heltech LoRa 32 v2 Board

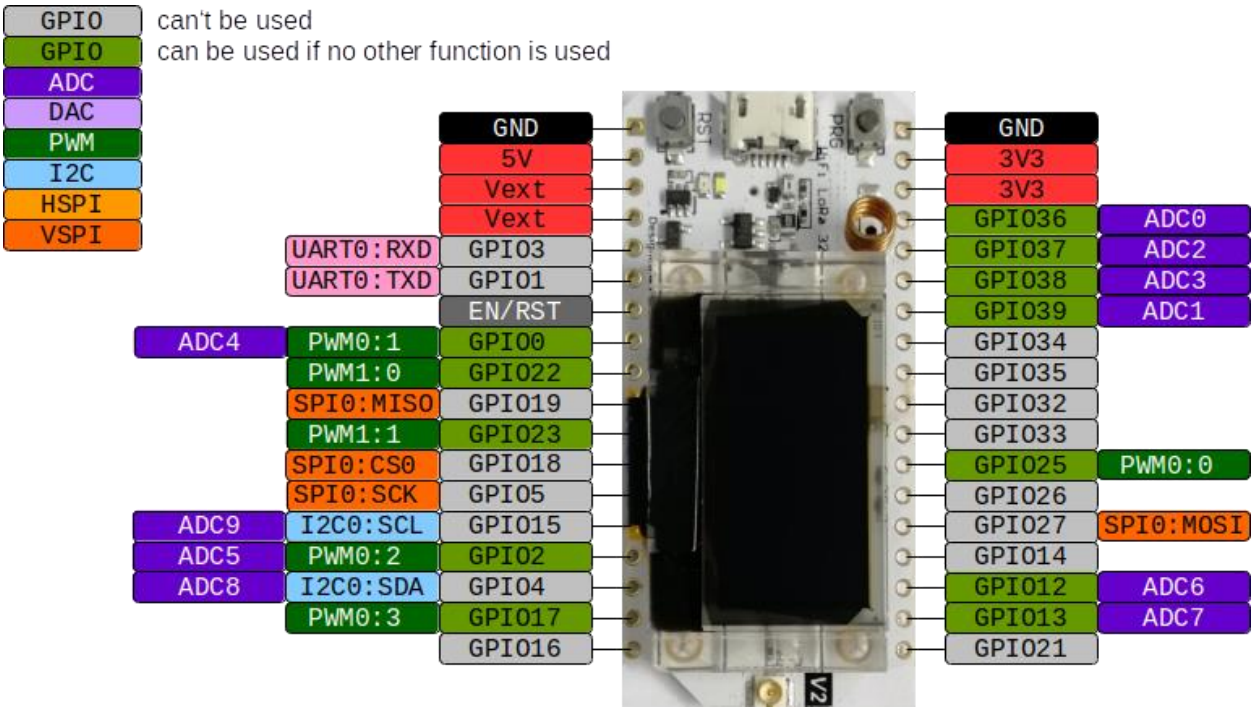

Figure 2.2: The Heltech LoRa 32 v2 Board. (https://heltec.org)

2.1.3.2 Arduino Uno

The Arduino Uno is a ATmega328 based MCU board that is used in prototyping and system development. It is comprised of 14 input and output pins that can be used for communication, powering, and sensor data collection and attenuator connections. (Cameron, 2021).

Figure 2.3 shows the Arduino Uno development board

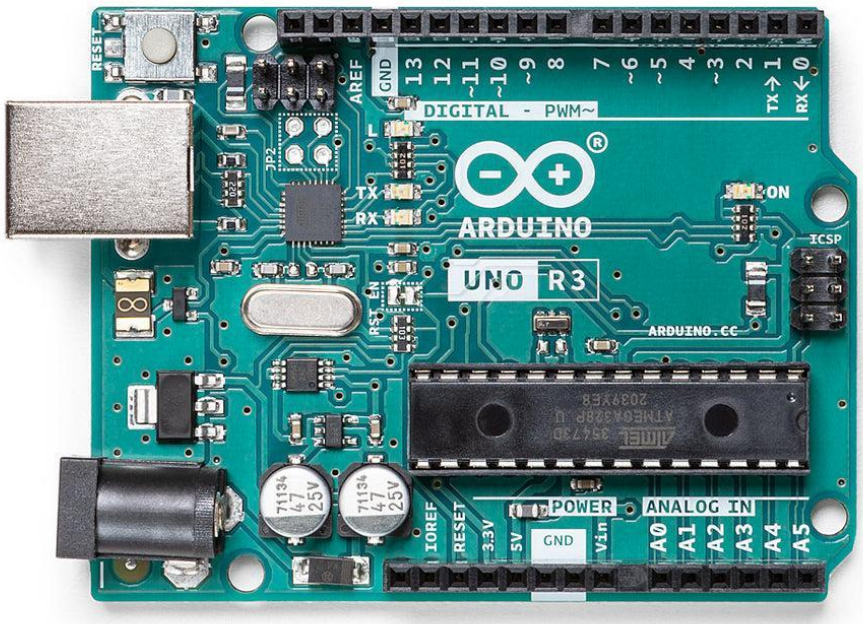

Figure 2.3: Arduino Uno Board. (https://www.arduino.cc/)

The Arduino Uno board can be powered via a USB connection to a computer or with an external power source. The board is designed to be capable of automatically selecting which among the two sources can supply it with power.(Arduino SKU, 2022)

The following are the features of the Arduino Uno:

- Power on Reset
- 2x 8-bit Timer/Counter with a dedicated period register and compare channels
- 1x 16-bit Timer/Counter with a dedicated period register, input capture and compare channels
- 1x USART with fractional baud rate generator and start-of-frame detection
- 1x controller/peripheral Serial Peripheral Interface (SPI)
- 1x Dual mode controller/peripheral I2C
- 1x Analog Comparator (AC) with a scalable reference input
- Watchdog Timer with separate on-chip oscillator
- Six PWM channels
- Interrupt and wake-up on pin change

2.1.3.3 Raspberry Pi

The Raspberry Pi is a family of single-board computers developed by the Raspberry Pi Foundation (www.raspberrypi.org). They are low-cost, low-power systems that run on ARM processors and have general-purpose input and output pins that can be connected to external devices like sensors, actuators and other MCUs like the Arduino. Figure 2.4 below shows a Raspberry Pi 4 model MCU.

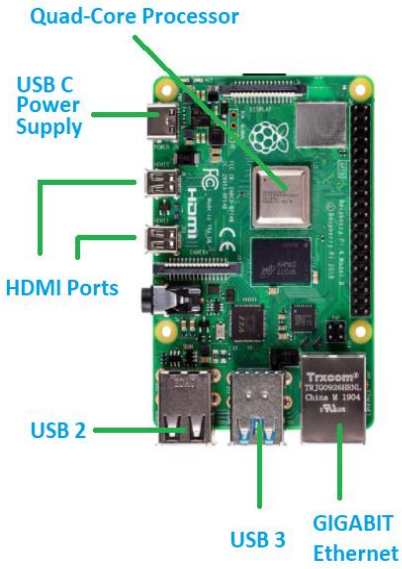

Figure 2.4 Raspberry Pi 4. (The Engineering Projects)

2.1.3.2 The DHT11 Sensor

The DHT11 is a sensor which measures relative humidity and temperature sensor. It provides a calibrated digital output with a 1-wire protocol. DHT sensors are calibrated out-of-the-box to be able to sense a wide range of temperature and humidity values from the environment. It directly connects them with ESP32/ESP8266 to obtain sensor output readings. They are internally composed of a capacitive humidity sensing sensor and a thermistor. These two components measure humidity and temperature. Figure 2.6 shows a schematic view of the DHT11 sensor.

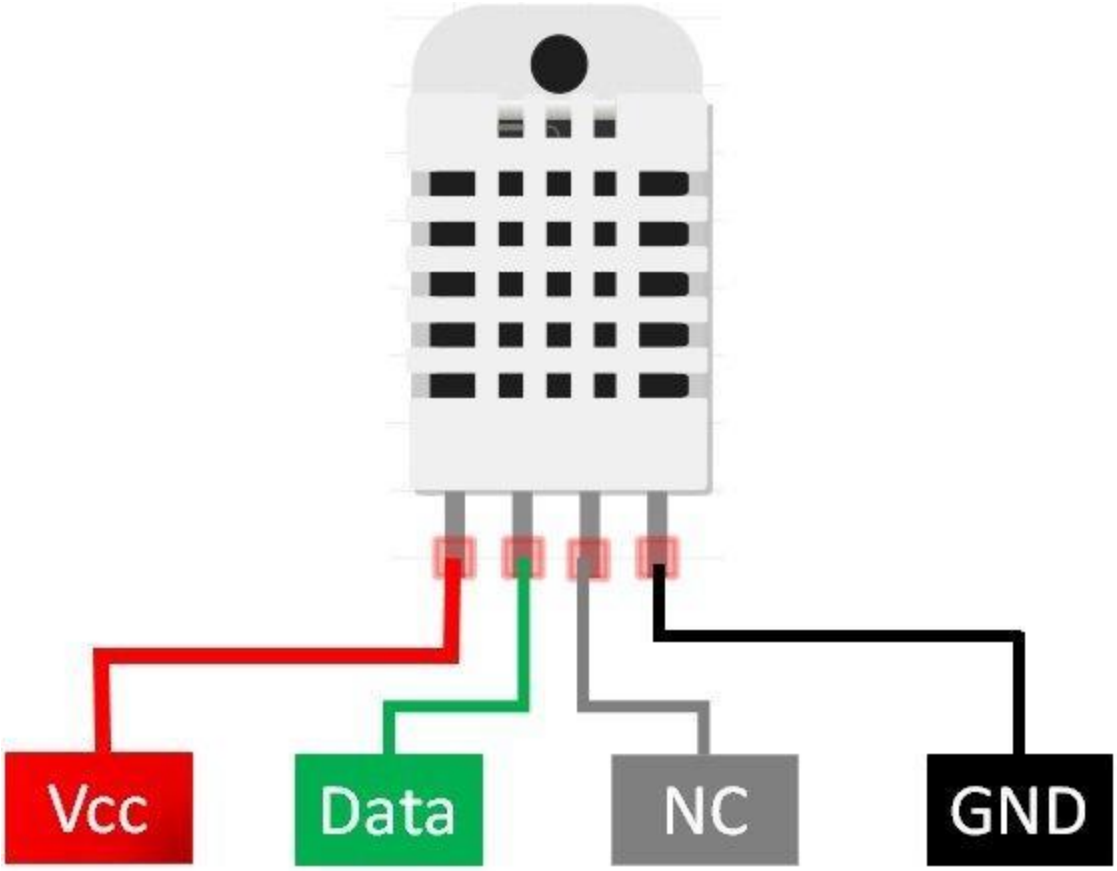

Figure 2.5: A DHT11 Sensor

The GND terminal connects to the ground of the circuit, NC is not connected to anything and in some models is completely left out. The data connection connects to the MCU for data transmission and feeds the MCU with environmental readings. The Vcc connection is connected to the power supply of the circuit, (D-robotics, 2010).

Technical Details

The DHT11 sensor operates within a voltage range of 3 to 5 volts, making it compatible with a variety of microcontrollers and electronic circuits. During data conversion, it consumes a maximum current of 2.5mA, ensuring minimal power usage. The sensor is designed to measure humidity levels between 20% and 80%, with an accuracy of ±5%, making it suitable for general environmental monitoring. Additionally, it can detect temperatures ranging from 0°C to 50°C, with a measurement accuracy of ±2°C. The sensor has a sampling rate of up to 1 Hz, meaning it can provide a new reading once every second at most.

2.3 Blynk IoT Platform

Blynk is an IoT platform for iOS or Android smartphones that is used to control microcontrollers via the Internet. This application is used to create a graphical interface or human-machine interface (HMI) by compiling and providing the appropriate address on the available widgets. It is a full suite of software required to prototype, deploy, and remotely manage connected electronic devices at any scale (Media, 2019).

Blynk was designed for the Internet of Things. It can control hardware remotely, it can display sensor data, can store data, visualize it and do many other things.
There are three major components in the platform:

- Blynk App: – It allows you to create graphical interfaces for your projects using various widgets which are provided.
- Blynk Server: – It is responsible for all the communications between the internet source and hardware.
- You can use the Blynk Cloud or run your private Blynk server locally. It's open-source, can easily handle thousands of devices and can even be launched on some MCUs.
- Blynk Libraries: – It enables communication, for all the popular hardware platforms, with the server and processes all the incoming and outgoing commands.

Figure 2.6 below shows the web-based set-up for the Blynk IoT platform with readings from sensor nodes 1 and 2.

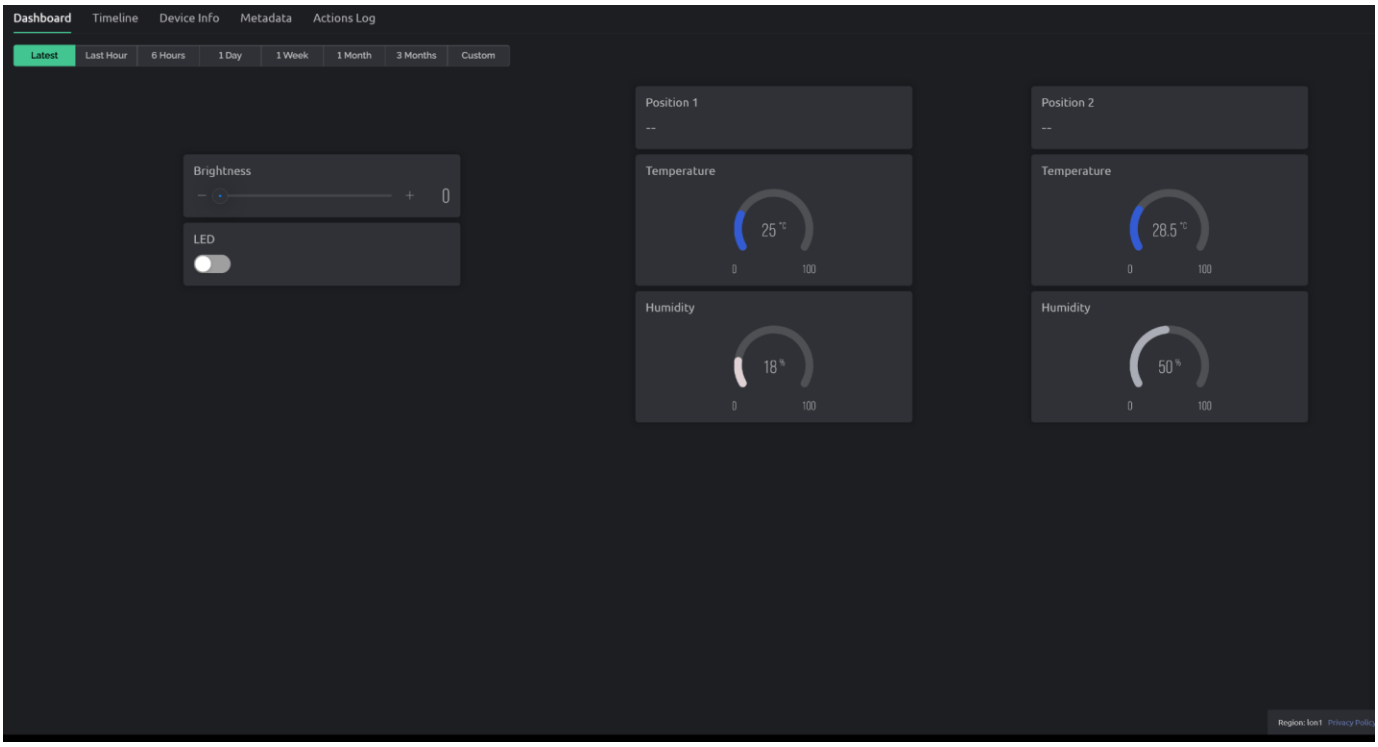

Figure 2.6: Blynk Server setup for the research work

Figure 2.7 below shows the mobile app-based set up of the Blynk IoT platform with readings from sensor node 1.

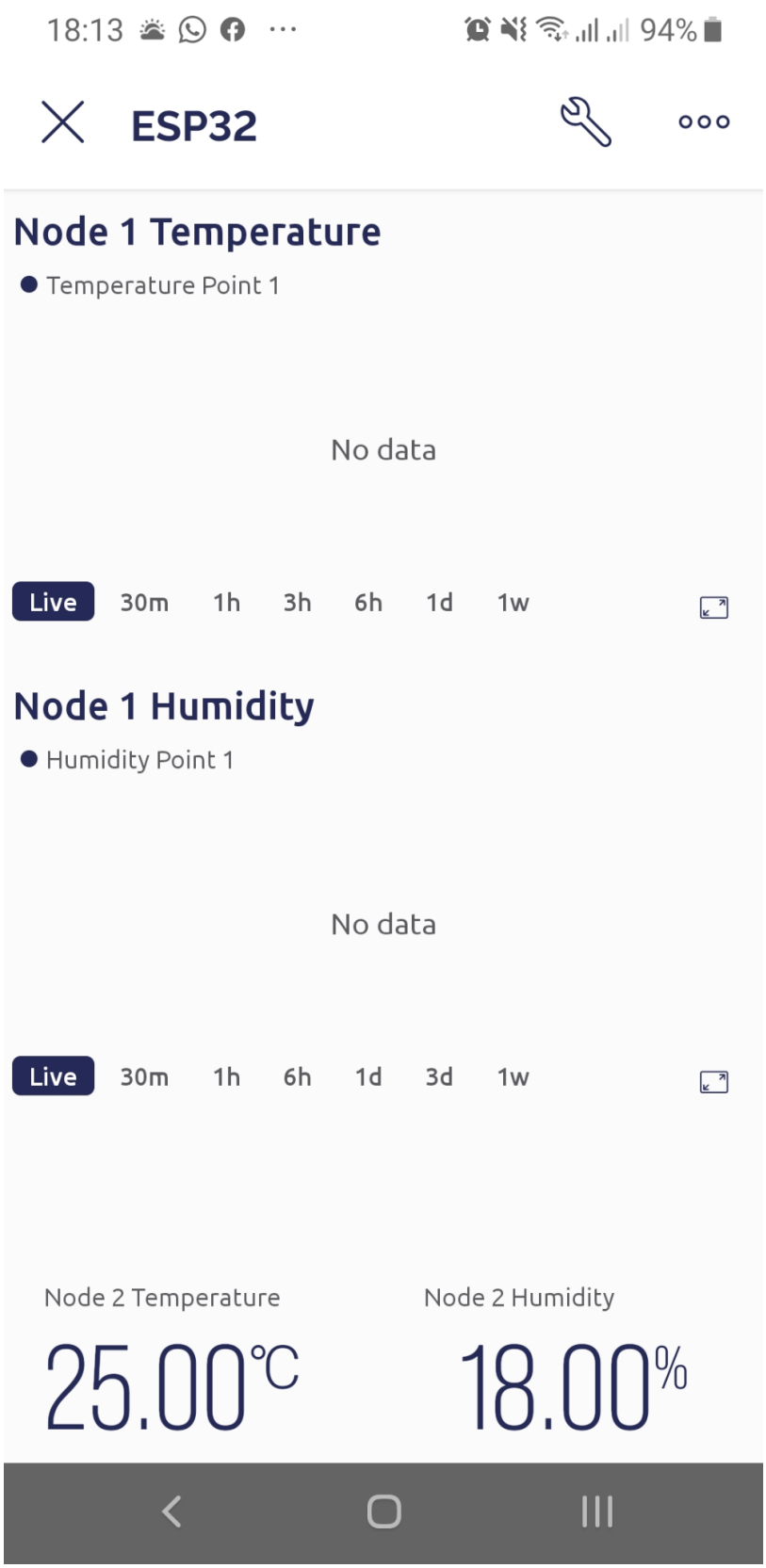

Figure 2.7: Blynk mobile App setup for the research work

2.4 Communication Protocols

Several communication protocols exist that can be used when designing a WSN. As was earlier alluded to, the most commonly used communication protocols are Bluetooth technology, Wi-Fi technology and ZigBee technology. Two additional protocols that we explore in this study are ESPNow, a newer technology developed by Espressif and was officially released in September 2016, and Long Range (LoRa) technology developed by Semtech in 2009.

2.4.1 ESPNow

ESPNow is a protocol that allows paired devices to communicate directly with one another through the data-link layer. Espressif Systems developed ESP-NOW to enable ESP32 microcontrollers to communicate without requiring a Wi-Fi connection. ESP-NOW operates at 2.4 GHz, the same frequency as Wi-Fi and Bluetooth, with microcontrollers paired prior to communication (ESP-IDF Programming Guide, 2021).

Pairing an ESP32 microcontroller requires the MAC (Media Access Control) address of the microcontroller, which is the address for communication within a network. After the pairing is done, the connection is safe and peer-to-peer, with no handshake required. This means that after pairing a device with each other, the connection is persistent. In other words, if suddenly one of your boards loses power or resets when it restarts, it will automatically connect to its peer to continue the communication (Espressif Systems, 2021).

ESPNow supports encrypted and unencrypted unicast communication, mixed encrypted and unencrypted peer devices, up to 250-byte payload can be carried, and sending callback function that can be set to inform the application layer of transmission success or failure.

ESPNow technology also has the following limitations; the number of encrypted peers is restricted, with 10 encrypted peers at the most supported in Station mode and up to 6 at the most in SoftAP or SoftAP + Station mode. While multiple unencrypted peers are supported, the total number of

peers, including encrypted peers, must be less than 20. In addition to this, the payload size has a limit of 250 bytes.

In simple words, ESPNow is a fast communication protocol that can be used to exchange small messages (up to 250 bytes) between ESP32 microcontrollers. ESPNow is very versatile and allows for one-way or two-way communication in different setups.

2.4.2 Long Range (LoRa)

LoRa is a wireless data communication technology that uses a radio modulation technique that can be generated by Semtech LoRa transceiver chips. LoRa (long range) is a low-power wide area network (LPWAN) technology that operates with a form of frequency modulation (FM), rather than amplitude modulation (AM), uses lower frequencies than the 2.4 GHz of Wi-Fi and Bluetooth communication, and has low power consumption. LoRaWAN (Long Range
Wide Area Network) is the protocol for creating LoRa-based networks, (The Things Network).

This modulation technique allows long-range communication of small amounts of data (which means a low bandwidth), and high immunity to interference while minimizing power consumption. So, it allows long-distance communication with low power requirements, (Cameron, 2021).

LoRa uses unlicensed frequencies that are available worldwide. These are the most widely used frequencies:

- 868 MHz for Europe
- 915 MHz for North America
- 433 MHz band for Asia

According to the Things network, Zambia uses EU863-870 and EU433MHz frequencies for LoRa.

2.4.3 Wi-Fi

Wi-Fi is a wireless technology used to connect computers, tablets, smartphones and other devices to the internet.

Wi-Fi is the radio signal sent from a wireless router to a nearby device, which translates the signal into data that can be seen and used. The device transmits a radio signal back to the router, which connects to the internet by wire or cable.

A Wi-Fi network is simply an internet connection that's shared with multiple devices in a home or business via a wireless router. The router is connected directly to an internet modem and acts as a hub to broadcast the internet signal to available Wi-Fi-enabled devices. This gives one the flexibility to stay connected to the internet as long as they stay within the network coverage area.

Wi-Fi, often referred to as WiFi, wifi, wi-fi or wi-fi, is often thought to be short for Wireless Fidelity and the organization that paid for the marketing firm is sometimes referred to as the Wireless Fidelity Alliance Inc.

Wi-Fi uses radio waves to transmit data from the wireless router to Wi-Fi-enabled devices like a TV, smartphone, tablet and computer. Because they communicate with each other over the airwaves, the devices and personal information can become vulnerable to hackers, cyber-attacks and other threats. This is especially true when connecting to a public Wi-Fi network at places like a coffee shop or airport. When possible, it's best to connect to a wireless network that is password-protected or a personal hotspot.

2.4.4 Protocols Comparison

The table below shows the comparison among the most frequently used communication protocols in underground WSN; ZigBee, WiFi, and Bluetooth and the 2 that were used in this research; LoRa and ESPNow.

Table 2.1: Comparison of communication Protocols (Zourmand, 2019, Heltech Automations Manual, Narmada A., 2012).

	ZigBee	WiFi	Bluetooth	LoRa	EspNow
Range(m)	50-500	50-100	10-100	1000-10000	50-250
Roaming	12 points between coordinators 14 between routers	Can roam between multiple access points	Securely paired to one single base station	Can roam between multiple access points	Can roam between multiple access points
Setup	Simple setup	Complex configurations needed	Communicates out of the box	Simple setup	Simple setup
No. of Connected Devices	Up to 65,536	Unlimited depending on the router	7 per base station	Unlimited depending on configuration	6 to 20 depending on the mode being used
Power usage (mW)	20-40	500-1000	1-100	20-24	10-55
Transmission Rate(Mbps)	$250x10^{-3}$	100-500	1	$3x10^{-3}$ to $22x10^{-3}$	1
Frequency(GHz)	2.4	2.4 or 5	2.4	2.4	2.4

CHAPTER 3

METHODOLOGY

The study will be guided by the three key objectives. By leveraging wireless sensor networks (WSNs), Internet of Things (IoT) platforms, and communication technologies, this research seeks to develop an integrated system capable of real-time environmental monitoring and data transmission. The objectives are:

1. **To create a wireless sensor network that will be used to monitor environmental temperatures and humidity in underground mines**
   One of the primary objectives is to design and implement a wireless sensor network (WSN) capable of monitoring temperature and humidity levels in underground mines. Due to the harsh and dynamic nature of underground environments, it is crucial to maintain stable and safe conditions for both workers and equipment. The proposed system will consist of strategically deployed sensor nodes that continuously measure temperature and humidity. These sensors will be interconnected through a wireless network, allowing seamless data transmission to a centralized monitoring hub. The system aims to provide real-time updates from every sensor node to the gateway through various transceivers, enabling proactive decision-making and rapid responses to hazardous conditions. This objective involves hardware selection, network topology design, and optimizing sensor placement for maximum coverage and efficiency.

2. **To test the limitations of transmitting the collected data using the selected communication protocols**
   Given the underground mining environment's unique challenges, such as signal attenuation, interference from geological structures, and power constraints; the study will assess the effectiveness of selected wireless communication protocols in transmitting data from the sensor nodes all the way to the gateway that is located at the central hub. The objective is to evaluate the performance, range, and reliability of selected protocols, such

as LoRa, and ESP-NOW, under different conditions. Testing will involve measuring data transmission rates, and packet loss under various distances and obstructions. The study will explore how signal strength and network stability change in open areas versus obstructed environments. By understanding these limitations, the system can be optimized to ensure seamless data flow and minimal latency, which are critical for real-time monitoring in underground mining operations.

3. **To adopt and customize a monitoring and data interpretation IoT platform**
   A critical component of the study is to integrate the sensor network with an IoT-based platform for data processing, visualization, and interpretation. The selected IoT platform will be customized to efficiently handle and display real-time environmental data, offering insights through graphical dashboards, trend analysis, and automated alerts. The customization process involves developing software interfaces, integrating cloud-based data storage, and configuring automated notifications for critical thresholds. This objective aims to ensure that the system is not only functional but also user-friendly, accessible, and scalable for future enhancements. Additionally, security and data encryption measures will be incorporated to protect sensitive mining operation data from unauthorized access.

3.1 Desktop Study

Literature review

The literature review served as a foundation for understanding existing research, technologies, and methodologies relevant to the study. It involved analyzing academic papers, technical reports, and industry standards related to sensor-based monitoring systems, underground mining safety, and wireless communication protocols. This review helped identify gaps in current solutions, benchmark best practices, and justify the need for the proposed system. Key areas of focus included wireless sensor networks (WSNs), environmental monitoring technologies, real-time data transmission, and emergency response systems in underground mining. Additionally, existing studies on signal attenuation, environmental interference, and energy-efficient communication protocols were examined to inform the system design and implementation.

Selecting the development language

Choosing the appropriate programming language was crucial for ensuring system efficiency, compatibility, and scalability. The selection process considered factors such as hardware constraints, real-time processing requirements, power efficiency, and ease of integration with microcontrollers (MCUs) and sensors. The C++ language was selected as it is preferred for low-level programming due to its high performance and efficient memory management, making it ideal for microcontrollers. Furthermore, the Arduino IDE was selected as the development environment of choice as it supports C++ out of the box and simplifies development for microcontrollers like ESP32 and Arduino boards due to the availability of libraries, frameworks, and support for the selected MCU and communication protocol.

Research on MCU capabilities

The Microcontroller Unit (MCU) serves as the central processing unit for the monitoring system. Selecting the right MCU involved evaluating its:

- Processing power (clock speed, number of cores)
- Memory (RAM, Flash storage) for handling sensor data
- Power efficiency, especially for battery-operated sensors
- I/O capabilities, ensuring compatibility with selected sensors and communication modules
- Built-in wireless communication (Wi-Fi, Bluetooth, ESPNow, LoRa, ZigBee)

The research phase involved looking at different MCUs' datasheets to determine which one best meets the project's power, performance, and connectivity requirements as was highlighted in 2.1.3.

Selection of the communication protocol

The right communication protocol was critical to ensuring reliable and efficient data transmission between sensors and the central control hub. The selection process considered factors such as range, power consumption, data rate, and environmental interference. Key options included:

- LoRa (Long Range) – Ideal for long-distance, low-power communication in underground mining environments.
- Wi-Fi – Suitable for short-range, high-bandwidth applications but may struggle with underground signal penetration.
- ZigBee – Energy-efficient for mesh networking but has limited range.
- Bluetooth (BLE) – Useful for low-power, short-range applications.
- ESP-NOW – A low-power, direct peer-to-peer communication protocol used in ESP32-based systems.

The final choice depended on factors such as terrain obstacles, power availability, and the need for real-time data updates. A combination of LoRa for long-range communication and ESP-NOW for local device interactions was picked to enhance system performance.

3.2 Experiments

A number of experiments and tests were conducted to determine the most cost-effective design for the wireless sensor network (WSN) that was ultimately developed for this study. The first step involved selecting the most suitable microcontroller unit (MCU) and communication protocols, as highlighted in the literature review. This was followed by the design and configuration of the WSN, ensuring that it met the requirements for reliable environmental monitoring in underground mining conditions. Additionally, distance determination tests were carried out to evaluate the efficiency and range of data transmission. These tests were conducted in two different scenarios: open-air tests, where nodes were placed without any obstructions to assess their maximum range under ideal conditions, and obstructed node tests, where nodes were positioned within a building with several barriers to analyze the impact of physical obstacles on signal strength and data transmission reliability. The insights gained from these experiments played a crucial role in optimizing the final WSN design for both performance and energy efficiency.

3.2.1 The WSN Design

The sensor network that was designed was based on the algorithm shown in Figure 3.1 below. This algorithm guided what sort of components were used, the determination of the programming language and environment, the types of tests to be carried out, and the processing needed in order

to effectively communicate the sensor reading to a central place and also the cloud platform that was used. The assembly process involved the selection and integration of hardware components, circuit design, and network configuration to establish a robust and functional system.

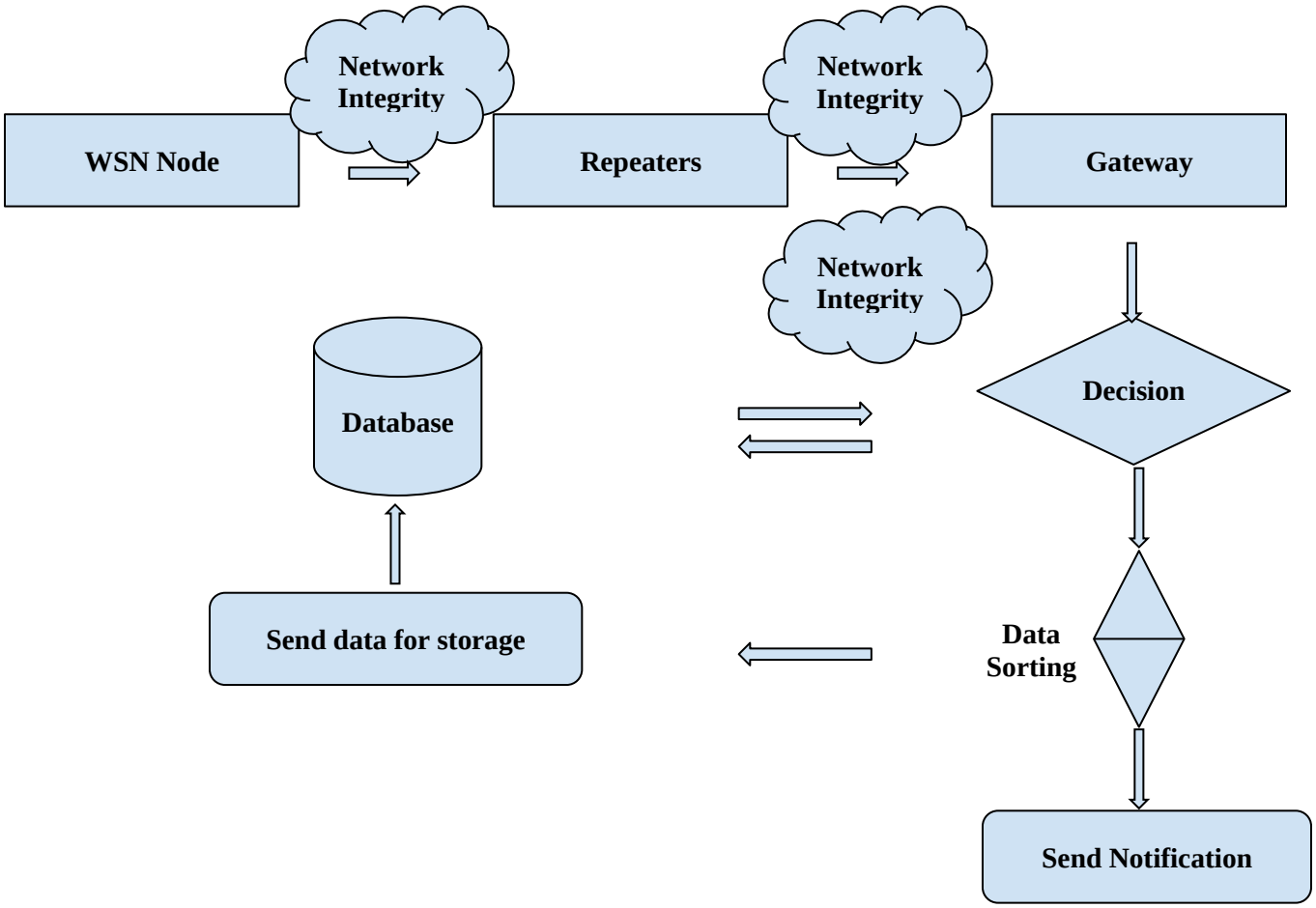

Figure 3.1: System iteration Algorithm

The WSN comprises two sensor nodes, responsible for collecting environmental data, three repeater transceivers, which enhance signal transmission and overcome physical obstructions, one gateway, which consolidates and processes the received data, and one web platform, designed to provide real-time monitoring, data visualization, and analysis.

Each sensor node was built using an ESP32 LoRa V2 board, chosen for its low power consumption and long-range wireless communication capabilities. The DHT11 sensor was integrated into the node to measure temperature and humidity, ensuring accurate environmental readings. To maintain signal stability and proper data transmission, a 10KΩ pull-up resistor was connected to the data pin of the DHT11 to keep the data pin high for proper communication between the microcontroller and sensor. Additionally, a 0.5KΩ resistor was included in the circuit to regulate the operation of an LED indicator by limiting the amount of voltage that is passed through it, which provides visual feedback on the status of the node, such as power-on state, data transmission activity, or error detection.

The circuit was assembled on a breadboard, allowing for flexibility in wiring and easy modifications. Connecting wires were used to link the components securely, ensuring stable electrical connections. A dedicated power source was provided for each node, with considerations for energy efficiency, as underground conditions often limit direct access to power. To optimize wireless communication, an external aerial (antenna) was attached to the ESP32 LoRa V2 board, significantly enhancing signal strength and extending the communication range, which is crucial for maintaining connectivity in underground tunnels and obstructed environments.

Figure 3.2 shows the WSN design and implementation

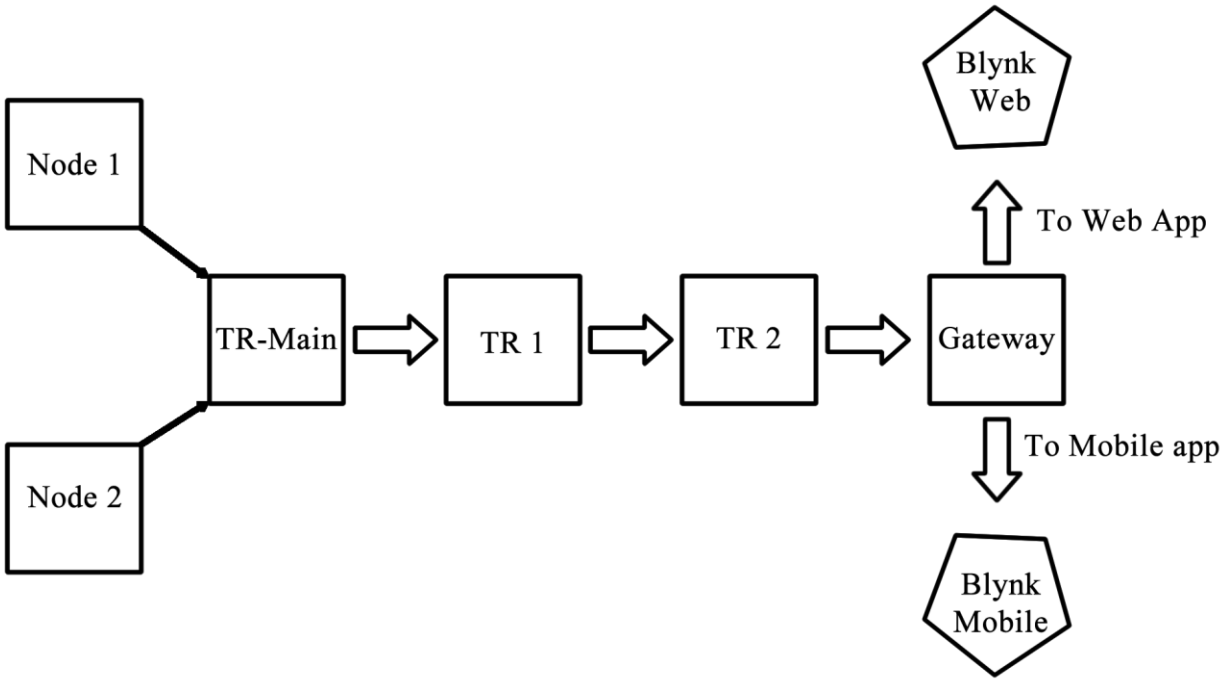

Figure 3.2: The designed WSN

Each node was tested in isolation before integration into the network to verify the functionality of individual components, including the sensor's ability to capture temperature and humidity data accurately, the LED's responsiveness, and the stability of the power supply.

Once individual nodes were validated, they were strategically positioned at selected locations to optimize environmental monitoring coverage. The three repeater transceivers, TR-Main, TR1, and TR2, were placed at intervals to extend the network range and mitigate potential signal losses due to obstacles such as rock formations, tunnel curvature, and various reinforcements that might prevent a clear line of sight. The gateway, which serves as the central data processing hub, was installed at an accessible location to aggregate data from all sensor nodes and relay it to the web platform.

3.2.1.1 The WSN Node

Figure 3.3 shows a photograph of a single WSN node

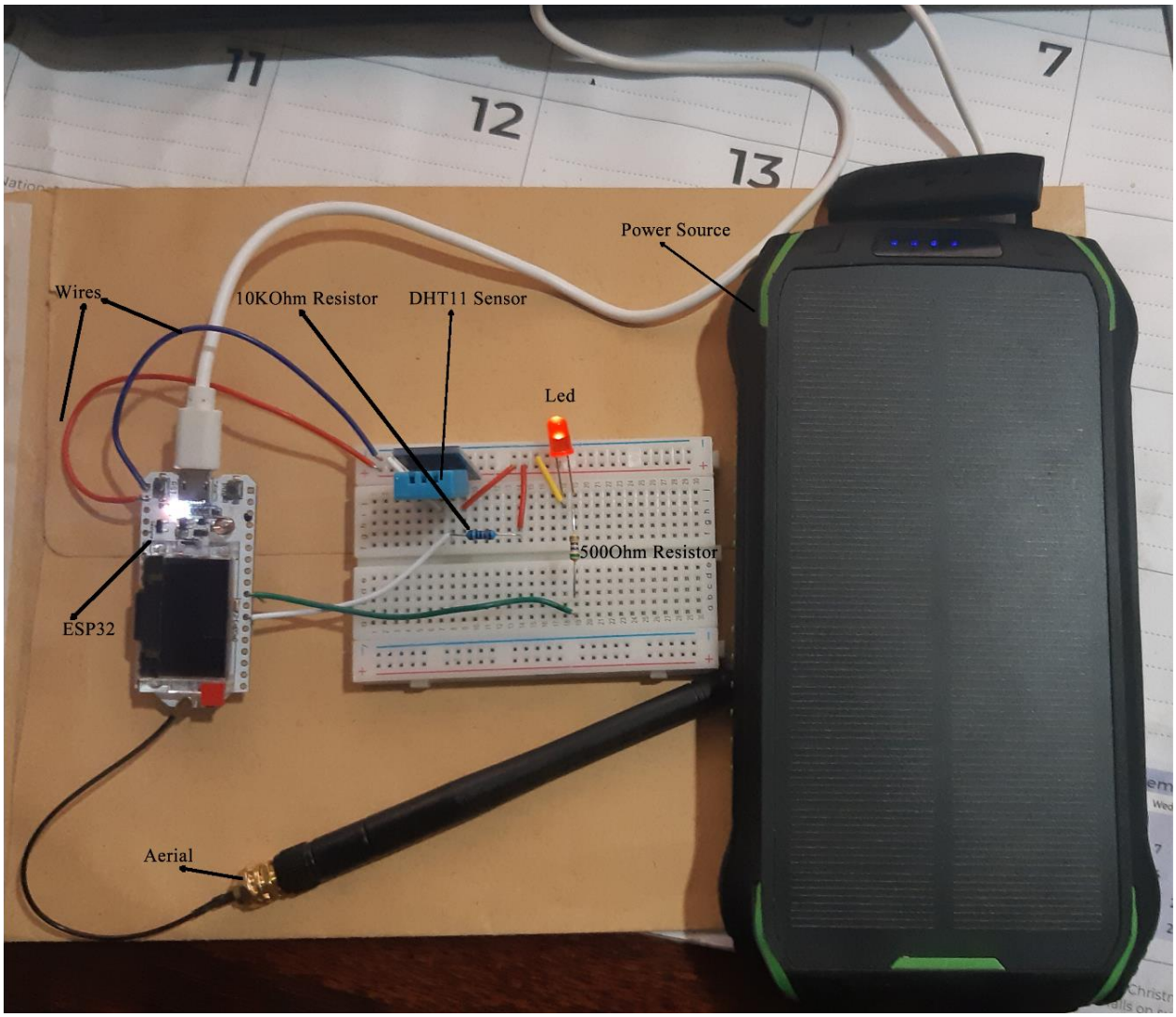

Figure 3.3: Single Node for the WSN

Figure 3.4 shows a schematic of the sensor node

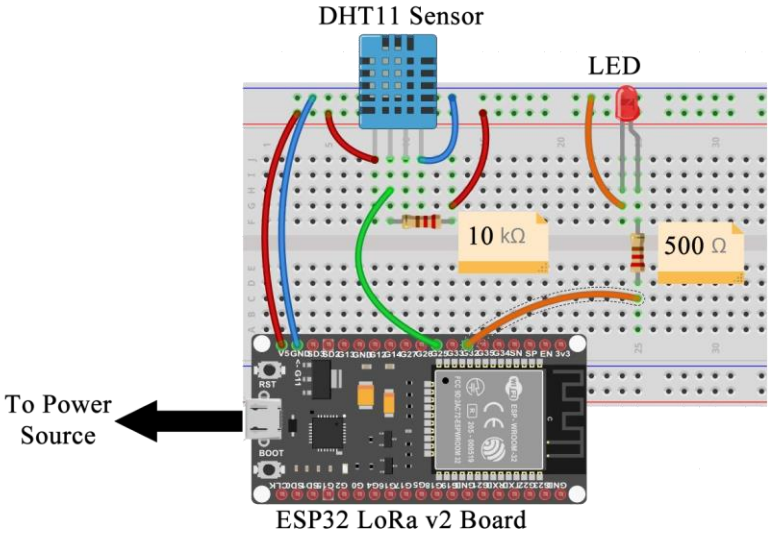

Figure 3.4: Single Node schematic for the WSN

Every node is programmed to collect data at an interval of 10 seconds, after which the sensor data is sent to the ESP32 board, which then transmits the data to the TR-Main transceiver using the ESPNow protocol. The TR-Main board functions as a receiver (slave), while Node 1 and Node 2 act as senders (masters). The nodes are set to receive an acknowledgement message indicating whether the transmission was successful. This acknowledgement is used to adjust the brightness of the LED, which momentarily increases to its highest level before dimming back to the lowest set brightness if the message is successfully delivered. If the message fails to go through, the LED remains at its lowest brightness, indicating that no new information has been received. The TR-Main board is responsible for receiving messages from all sending nodes and identifying the specific board from which each message originates as every board was programmed to have a unique identifier that distinguishes it from other boards and acts as a security encryption mechanism from outside intrusion.

Figure 3.5 shows the set-up for the ESPNow data transmission from nodes to the Main Transceiver

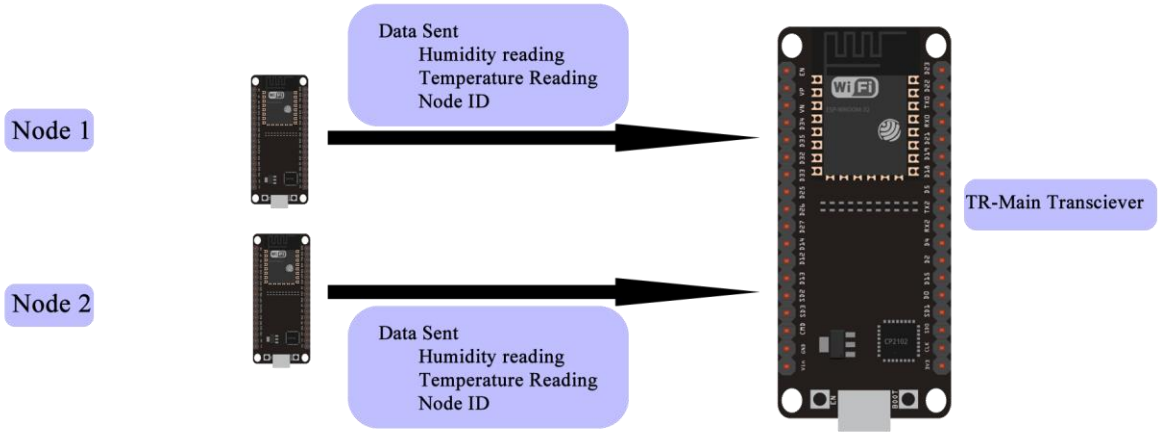

Figure 3.5: ESPNow data transmission from nodes to the Main Transceiver

3.2.1.2 The Transceivers

Each transceiver consists of an ESP32 LoRa V2 board, an external aerial, and a power source, forming a crucial part of the wireless sensor network. The TR-Main transceiver operates using two communication protocols: the ESPNow protocol, which it employs to collect data from the sensor nodes, and the LoRa protocol, which it uses to further transmit data to other transceivers. The data received from the nodes is first serialized and converted into a string, ensuring a structured format for transmission using the LoRa protocol. This serialized data is then sent to TR-1, which utilizes the LoRa protocol to relay the information to TR-2, also operating on the LoRa protocol. Once the data reaches TR-2, it is subsequently forwarded to the gateway, where it undergoes further processing for analysis and transmission.

3.2.1.3 The Gateway

The gateway is comprised of an ESP32 LoRa V2 board, an antenna, and a power source. It can be connected to either a PC or function independently with just an external power source but requires a router or any other source of wireless internet connectivity to enable it to connect to the IoT platform.

The gateway uses two protocols:

- The LoRa Protocol which for collecting data from the transceivers
- And Wifi which it then uses for connecting to the internet and consequently the IoT platform.

Once data from TR-2 reaches the gateway, it is de-serialized and the information is changed from a string to a float for it to be sent to the web and mobile applications, which are sent to
the gateway is connected to a computer and a Wi-Fi connection for information to be made available on the applications that are used for monitoring and logging the sensor readings that are sent from each node. The applications run on the Blynk IoT SaaS platform.

3.2.1.4 The Blynk IoT SaaS Platform Setup

The Blynk setup was configured to receive and show data on mobile and web dashboards.

To properly sync the gateway with the platform, a unique template is created, and the template can be renamed to a more friendly name. The generated template comes with a unique template ID that is combined together with an authentication token to identify and connect the MCU to the cloud platform. This information is collectively used for configuring the firmware of the MCU so that it can be recognized via the API once it makes API calls to the platform.
The platform offers several widgets, which you can place on the dashboard and map to various pins and sensors, the nature of the readings, their units of measure and the interval of readings can then be set as parameters on the cloud platform.

3.2.2 Programming

The language that was used for the development of the backend and data processing of the system was C++.
C++ is a high-level, general-purpose, and cross-platform programming language that is for building high-performance applications. Due to the level of control it grants over systems and

resources, the language is usually used in operating systems, design of graphical user interfaces and frequently used in embedded systems such as the backbone of the WSN that was built for this study.

The programming of the MCUs was done using the Arduino Integrated Development Environment (IDE) which was developed by Davis Mellis in 2005. The IDE comes prepacked with a wide range of compatible MCUs and easily identifies the type of boards that are connected to it.

The Arduino IDE contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the MCU hardware to upload programs and communicate with them.

Figure 3.6 below shows the Arduino IDE which was used for programming.

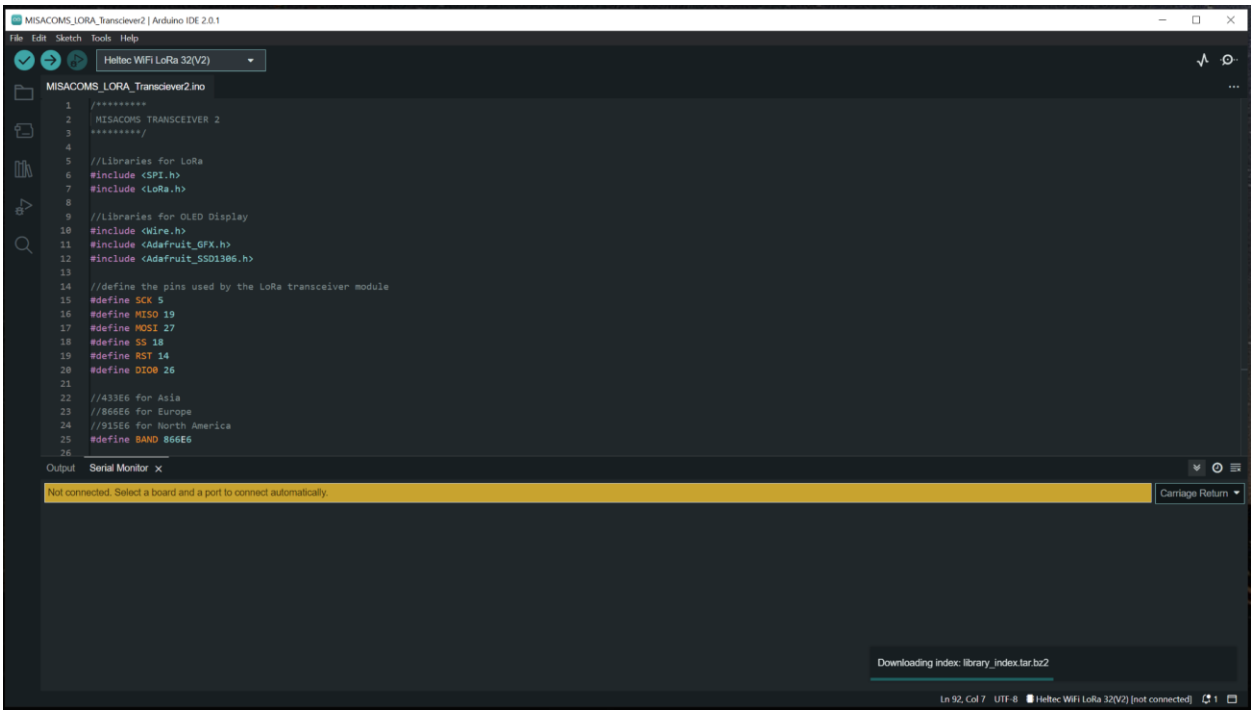

Figure 3.6: The Arduino IDE

3.2.3 Distance Determination Tests

Two on-surface tests were conducted to determine the best placements of the nodes for tests underground. The tests were:

- Open Air Tests
- Obstructed-Node Placement tests

Open Air Tests

One of the nodes was set up in an open field with no obstructions like buildings, vehicles, towers or a large amount of foliage. This was done at the Goma fields and near the TDAU buildings of the University of Zambia

Measurements were taken for the RSSI from each receiver from the respective transmitter for each protocol used at an incremental distance of 50m from the last point. The transmission interval for each protocol was set at 10 seconds. The experiment was designed as shown in Figure 3.7.

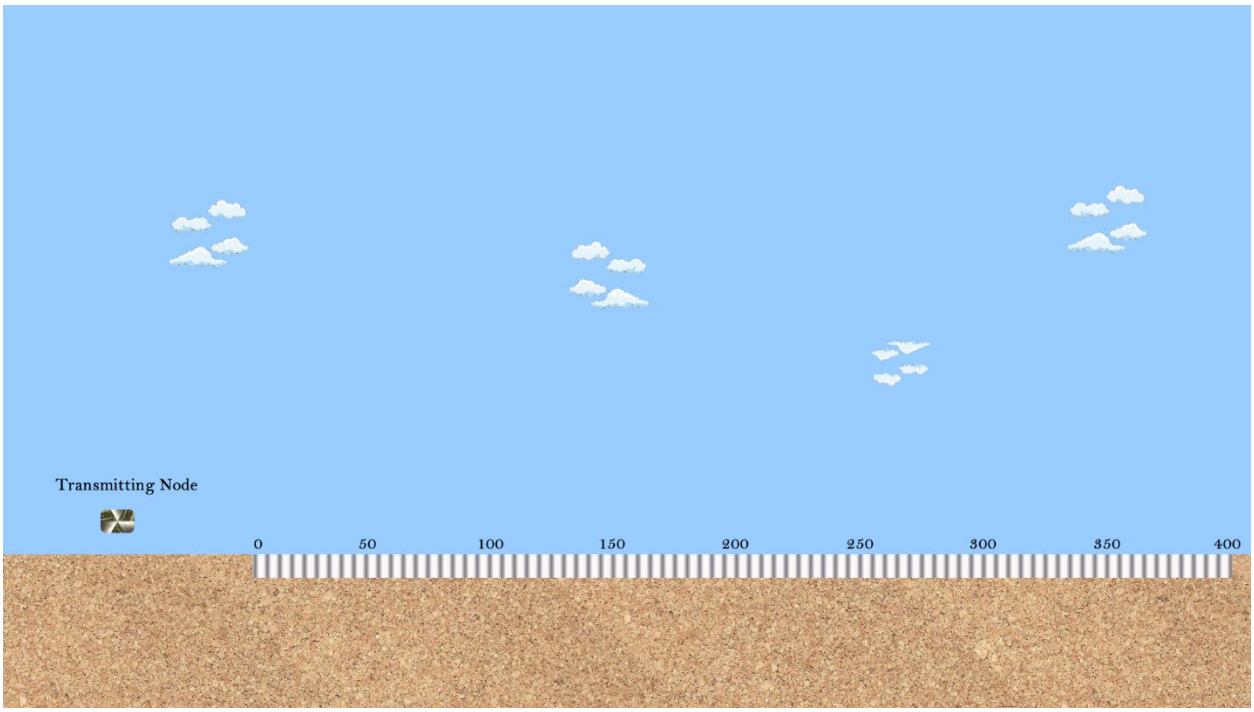

Figure 3.7: Open Air Distance Determination test

Obstructed Node Distance Determination tests

A node was set up in an enclosed concrete building, and the receivers were placed and then later carried, outside the building for the purpose of recording signal transmission readings. To further test the rate of signal reliability, the receivers were blocked by a brick wall fence at the 75m interval. After the 90m mark, another brick wall fence, and several concrete and brick structures, foliage, and vehicles were encountered to further test the signal reliability under obstructions.

Measurements were taken for the RSSI from each receiver from the respective transmitter for each protocol used at an incremental distance of 25m from the last point. The transmission interval for each protocol was set at 10 seconds. The experiment was designed as shown in Figure 3.8.

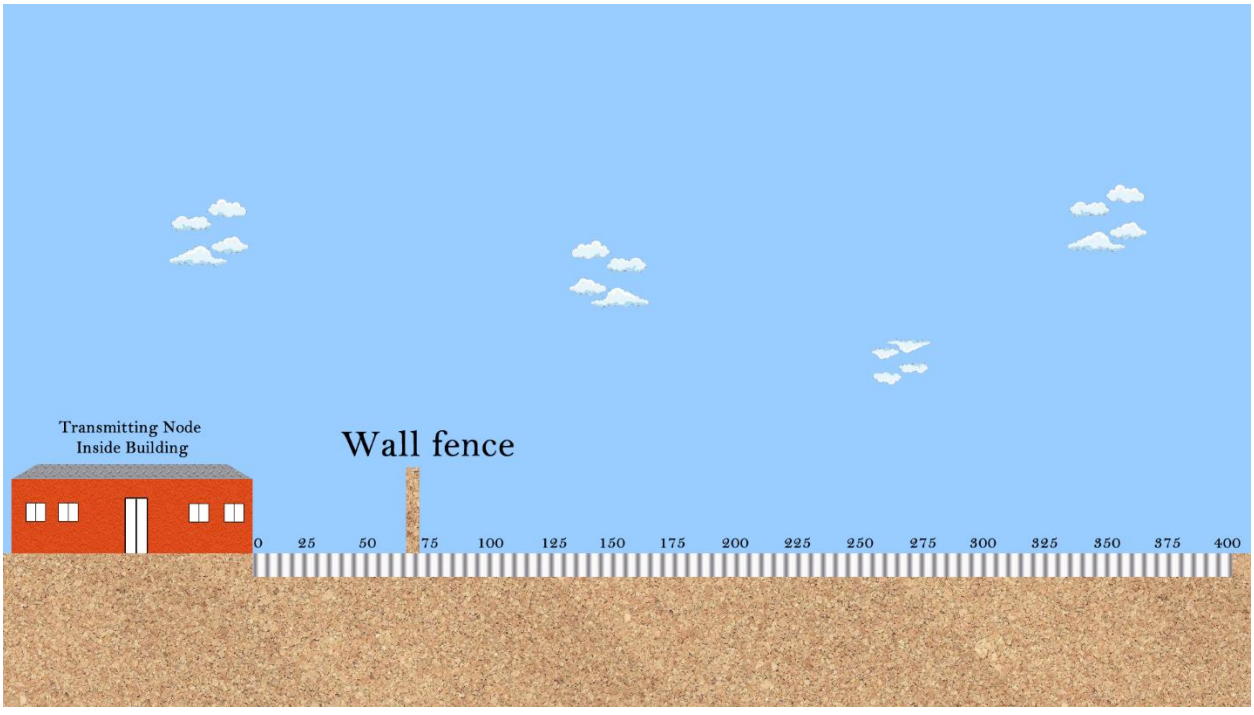

Figure 3.8: Obstructed Node Distance Determination test

3.3 Study Areas

The study areas for the project are the University of Zambia in Lusaka and surrounding areas.
The distance determination tests were conducted at the Goma fields of the University of Zambia, Kalinga linga compound and Kabulonga areas of Lusaka City.
The sample comprised of the readings that were collected from:

- Network reliability at pre-determined distances
- Range of temperatures and humidity recorded
- Open air readings from the sensor nodes to the gateway with a clear line of sight
- Obstructed readings from sensor nodes to the gateway with different types of obstructions
- Varied transmission and reception speeds of the data from the sensors to the gateway.

3.4 Sampling Techniques

For sensor-based data collection that was used in this study, a systematic sampling approach was implemented. Environmental parameters such as temperature and humidity were monitored across various locations using sensor nodes. These readings were verified against established thermometers and humidity sensors to ensure accuracy and reliability.

Additionally, RSSI data was collected from each node using two experimental setups. In the first setup, nodes were placed in open space without obstructions, while in the second setup, nodes were positioned within a building with multiple obstructions. RSSI measurements were recorded at 50-meter intervals in open space and at 25-meter intervals in obstructed environments, with data collected every 10 seconds. This approach enabled a comparative analysis of signal attenuation under varying environmental conditions, ensuring a robust evaluation of the proposed monitoring system’s reliability and efficiency.

CHAPTER 4

RESULTS, AND ANALYSIS

Two protocols were picked and tested in both open air, where there was no tangible obstruction between the sensor nodes, and the gateway, and the obstructed node, where the nodes were left in a building and the gateway was placed outside with several other notable obstructions like concrete and multiple buildings. The protocols were LoRa and ESPNow. The key parameter which was tested was the RSSI that the node was sending to the gateway over a predetermined interval of 10 seconds. The results are shown in tables 4.1 and 4.2, and graphs 4.1 and 4.2 below.

4.1 Distance Determinations Tests Results

Table 4.1 and Figure 4.1 below show the results of the open-air test. These tests were conducted at the UNZA Goma fields and behind the School Engineering's TDAU. The results recorded were the same in both locations for both the LoRa and ESPNow protocols.

As can be seen from the data, ESPNow lost its connection after a distance of 160m while the LoRa device kept transmitting data beyond 1Km which was set as the benchmark distance for these tests.

4.1.1 Open Air Distance Determination Test Results

Table 4.1: Open Air Distance Determination Test Results

Distance(m)	LoRa- RSSI(dBm)	ESPNow- RSSI(dBm)	Comment
0	-35	-40	
50	-81	-81	
100	-97	-85	
150	-103	-90	
200	-107		**Signal for ESP Now lost at 160m**
250	-109		
300	-106		
350	-99		
400	-96		
450	-107		
500	-108		
550	-106		
600	-107		
650	-107		
700	-107		**No Signal loss for Lora until 1Km**

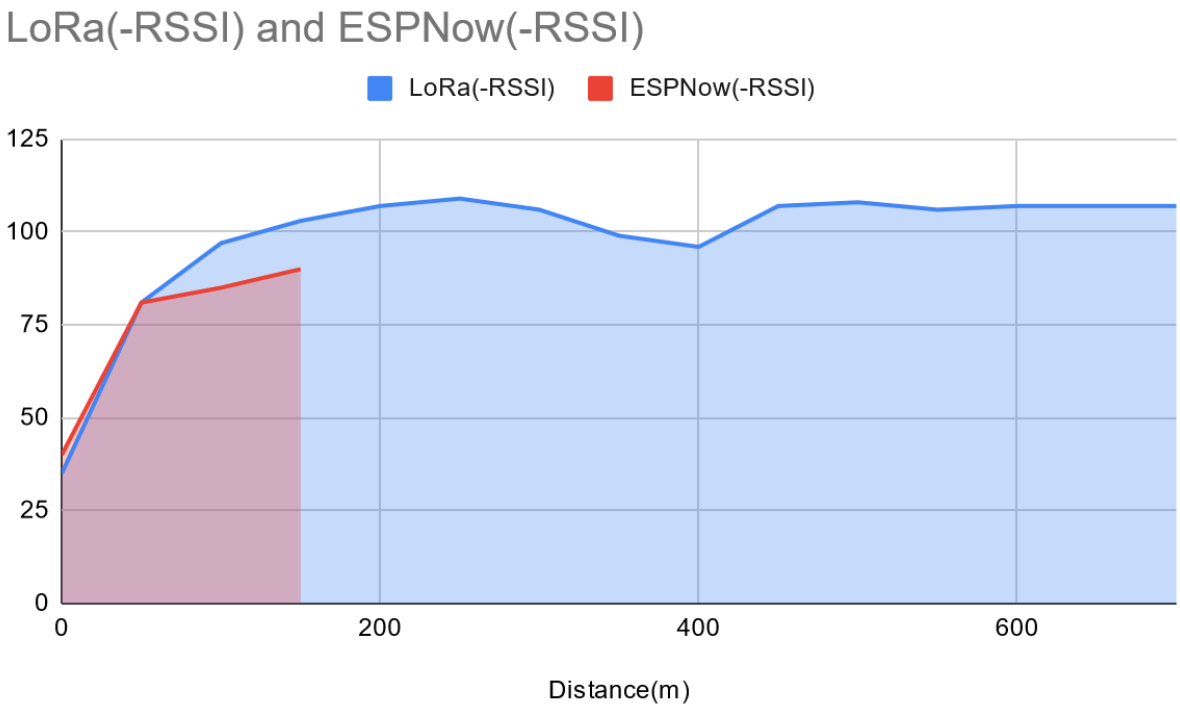

Figure 4.1: Open Air Distance Determination Test Results

4.1.2 Obstructed Node Distance Determination Test Results

Table 4.2 and Figure 4.2 below show the results of the open-air test. These tests were conducted in the residential parts of Kabulonga and Kalinga Linga. The results obtained were not very different in each of the tests for the LoRa and the ESPNow protocols.

As can be seen from the data, ESPNow lost its connection after a distance of 55m while the LoRa device lost transmission after a distance of 330m.

Table 4.2: Obstructed Node Distance Determination Test Results

Distance(m)	LoRa- RSSI(dBm)	ESPNow- RSSI(dBm)	Comment
0	-40	-42	
25	-90	-95	
50	-102	-110	
75	-108		**The signal for ESP Now lost at 55m**
100	-110		
125	-112		
150	-113		
175	-114		
200	-115		
225	-116		
250	-116		
275	-115		
300	-116		
325	-115		
350			**Signal for Lora lost at 330m**

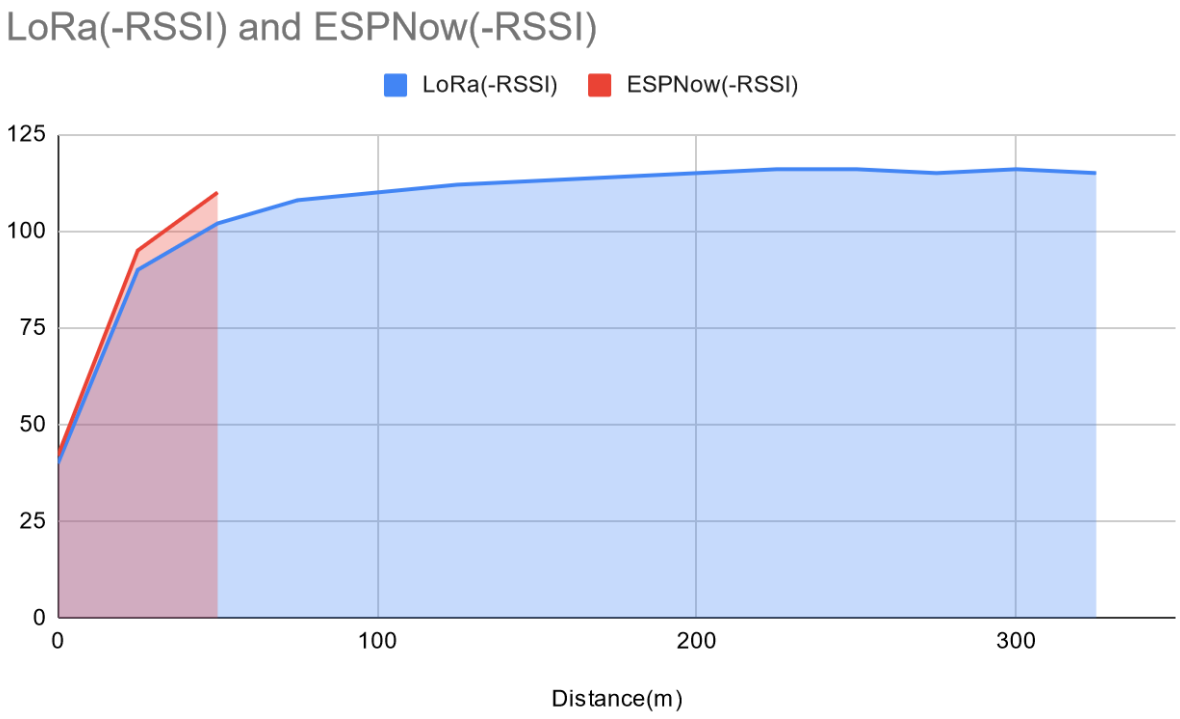

Figure 4.2: Obstructed Node Distance Determination Test Results

In both the open-air and obstructed node tests, one variable that was not recorded or observed was the impact that the elevation change had on the RSSI from each protocol as the terrain changed over certain distances in both test scenarios.

The experiment results revealed a clear contrast between the two communication protocols. ESPNow, while suitable for short-range applications, struggles significantly with distance and obstructions. The rapid degradation in RSSI values for ESPNow indicates that it is more susceptible to environmental factors such as walls and dense structures. On the other hand, LoRa's ability to maintain transmission over significantly longer distances, even in obstructed conditions, makes it an ideal choice for applications requiring extended coverage and reliability. The minimal degradation in LoRa's signal strength, even at 1 km in open-air tests, underscores its efficiency in long-range communication scenarios.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

This study successfully designed and implemented a wireless sensor network (WSN) for environmental monitoring in underground mines by integrating two communication protocols: LoRa and ESPNow. The selection of these protocols was based on their unique strengths, with LoRa providing long-range, obstacle-penetrating communication, while ESPNow enabled low-latency, real-time data transmission. To evaluate the effectiveness of both protocols, extensive tests were conducted in two different environments—open-air conditions with no obstructions and obstructed conditions where sensor nodes were placed inside a building with concrete walls and other physical barriers. The key parameter measured during these tests was the Received Signal Strength Indicator (RSSI), which reflects the quality and reliability of wireless transmissions.

Several considerations were taken into account when choosing the best communication technology for the wireless sensor network that was designed to monitor humidity and temperature levels. LoRa was selected as a viable candidate for scenarios involving large areas and distant locales due to its long-range and obstacle-penetrating capabilities, while the limited range of ESP-NOW's low-latency communication is ideal for situations where real-time updates are essential.

In LoRa, the lower the RSSI the further away the devices are from the gateway, and the poorer the connection gets resulting in poor data transmission rates with the interval between transmitted and received data increasing as the distance increases. LoRa's greater transmission range reduced the need to set up as many gateways or repeaters in remote or large-scale applications where the sensor nodes are dispersed across a wide area. Additionally, its capacity to pass through obstacles made sure that sensor nodes positioned within buildings or other obstructed locations were able to properly connect with the sensor nodes and the central processing system. The performance of

LoRa could further be improved by adjusting parameters such as the spreading factor and selecting more efficient LoRa modules.

In ESPNow, the amount of power needed to transmit a data package increases with the distance between two devices. The need for drawing more power tends to impact the life span of the power source used, and also the quality of the data sent as the transmissions tend to give rise to data loss. Instantaneous updates of temperature and humidity are well-suited to the low-latency communication scenarios for which ESP-NOW is well-suited. When being able to react quickly to changing circumstances is essential, ESP-NOW's direct device-to-device connection provides real-time data without needless delays. The lower transmission distances of ESP-NOW, however, may call for a more densely packed network of nodes for applications that call for a wider coverage area, particularly when involving many buildings or levels.

Considering these findings, the choice of communication technology depends on the specific application requirements. LoRa is more suitable for large-scale deployments where nodes are spread over a wide area, making it ideal for transmitting data from the areas where the sensor nodes are placed to the central hub where the data processing and analysis takes place. Meanwhile, ESPNow is well-suited for scenarios that require rapid, low-latency data transmission within localized areas, ensuring real-time responsiveness. This makes the ESPNow protocol most suitable for collecting sensor data and relaying it to any transceiver or other transmission point that can then transmit this data over great distances. By integrating both protocols, the designed WSN successfully leveraged ESPNow’s speed for immediate updates and LoRa’s range for efficient data transmission to a central processing node. This hybrid approach ensured a responsive and scalable system capable of transmitting environmental data to the Blynk server for further analysis and interpretation.

Overall, the study demonstrated the feasibility of using a dual-protocol WSN to optimize both coverage and efficiency in underground mine monitoring. The insights gained from this research contribute to the growing body of knowledge on WSN deployment in challenging environments and can inform the design of more robust IoT-based monitoring solutions.

5.2 Recommendations

The study conducted an extensive analysis and identified particular issues and limitations that require attention for future work. The recommended actions are as follows:

1. The use of non-industrial grade components in the study could have greatly affected both the range of the signals and the accuracy of the data, hence industry-grade sensors and components need to be tested and compared to off-the-counter grade components in a similar system; and
2. To better provide a more accurate and usable system, the designed and developed system needs to be tested in an underground mine.

3. Future improvements could also focus on optimizing power consumption, refining node placement strategies, and incorporating additional sensors to enhance environmental data collection.

References

1. Akka, M.A. and Sokullu, R. (2015) *Wireless Underground Sensor Networks: Channel Modeling and Operation Analysis in the Terahertz Band*. Available at: http://dx.doi.org/10.1155/2015/780235.
2. Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless sensor networks: a survey. *Computer networks*, *38*(4),[1] 439-476.
3. Akyildiz, I.F. and Stuntebeck, E.P. (2006) *Wireless underground sensor networks: Research challenges*. *Ad Hoc Networks*, 4.
4. Arduino (2022) *Product Reference Manual*.
5. Besa, B. and Mutambo, V. (2018) *Analysis and Management of safety issues on the Copperbelt Mines*.
6. Besa, B., Mulenga, S. and Mazimba, C. (2018) *A safety and accident communications system for underground Mines*.
7. Cameron, N. (2021) *Electronics Projects with the ESP8266 and ESP32 Building Web Pages, Applications, and WiFi Enabled Devices*. Edinburgh: Apress.
8. Dasenbrock, D.D. (2010) *Automated landslide instrumentation programs on US route 2 in Crookston, MN*. In: *Proceedings of the Annual Conference of the Minnesota Geotechnical Society*, pp. 165–185, Minneapolis, MN, USA, February 2010.
9. Di Fransesco, M., Das, S.K., and Anastasi, G. (2011) *Data Collection in Wireless Sensor Networks with Mobile Elements: A Survey*.
10. D-Robotics (2010) *DHT11 Humidity and Temperature sensor datasheet*.
11. Espressif. *ESP32 API Reference*. Available at: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/network/esp_now.html.[1]
12. Heltec Automations. *Wi-Fi LoRa 32(V2) Manual*. Available at: https://heltec.org.
13. Jo, B. and Khan, R.M.A. (2018) *An Internet of Things System for Underground Mine Air Quality Pollutant Prediction Based on Azure Machine Learning*. *Sensors (Basel)*, 18(4), p. 930. doi: 10.3390/s18040930.
14. KCM. *Nampundwe Mine*. Available at: http://kcm.co.zm/our-operations/mining/nampundwe-mine/.

15. Kumar, A., Misra, P., & Gupta, D. (2019). A survey on air pollution monitoring using wireless sensor networks. *Journal of King Saud University-Computer and Information Sciences*, *31*(4), 482-492.
16. Li, Z., Liu, F., & Nie, Z. (2010). Wireless sensor networks for precision agriculture: A review. *Computers and electronics in agriculture*, *71*(2), 127-136.
17. Lutkevich, B. (2019) *Microcontroller*. Available at: https://www.techtarget.com/iotagenda/definition/microcontroller.
18. Mainwaring, A., Welsh, M., Enemark, R., & Agre, J. (2002). A wireless sensor network for habitat monitoring. *ACM international workshop on Wireless sensor networks*, 88-97.
19. Media's, E., S., & Rif'an, M. (2019). *Internet of Things (IoT): BLYNK Framework for Smart Home*. *KnE Social Sciences*, 3(12), 579–586. https://doi.org/10.18502/kss.v3i12.4128
20. Moridi, M.A., Kawamura, Y., Sharifzadeh, M., Chanda, E.K. and Jang, H. (2014) *An investigation of underground monitoring and communication system based on radio waves attenuation using ZigBee*.
21. Moridi, M.A., Kawamura, Y., Sharifzadeh, M., Chanda, E.K., Okawa, H. and Jang, H. (2015) *Development of underground mine monitoring and communication system integrated ZigBee and GIS*. *Int J Min Sci Technol*.
22. Narmada, A. and Rao, P.S. (2012) *Zigbee based WSN with IP connectivity*. In: *2012 Fourth International Conference on Computational Intelligence, Modelling and Simulation*.
23. Pan, T. and Liu, X. (2011) *Hybrid Wireless Communication System Using ZigBee and WiFi Technology in the Coalmine Tunnels*.
24. Pantelopoulos, A. (2016). A survey on wearable sensor-based systems for health monitoring and prognosis. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*,[2] *46*(6), 860-880.
25. Peiris, V. (2013) *Highly integrated wireless sensing for body area network applications*. *SPIE Newsroom*. doi:10.1117/2.1201312.005120.
26. Porter GeoConsultancy Pty Ltd (2010). Available at: https://portergeo.com.au/database/mineinfo.asp?mineid=mn552
27. Rashid, M., Defir, O., & Amini, M. R. (2016). Wireless sensor networks in healthcare: A survey. *International Journal of Wireless & Mobile Networks*, *8*(3), 1-15.

28. Salam, A. and Vuran, M.C. (2017) *Wireless Underground Channel Diversity Reception With Multiple Antennas for Internet of Underground Things*. *CSE Conference and Workshop Papers*, 303.
29. Sato, T., Honda, R. and Shibata, S. (1999) *Ground strain measuring system using optical fibre sensors*. In: *Proceedings of the Smart Structures and Materials 1999: Sensory Phenomena and Measurement Instrumentation for Smart Structures and Materials*, vol. 3670, pp. 470–479, Newport Beach, CA, USA, March 1999.
30. The Things Network. *What is LoRaWAN?*. Available at: https://www.thethingsnetwork.org/docs/lorawan/what-is-lorawan/.
31. Tongco, M.D.C. (2007) *Purposive Sampling as a tool for informant selection*.
32. Tubis, A., & Wroblewski, A. Risk Assessment Methods in Mining Industry—A Systematic Review. *Applied Sciences*, *10*(15), 5172. https://doi.org/10.3390/app10155172
33. Ullo, S.L. and Sinha, G.R. (2020) *Advances in Smart Environment Monitoring Systems Using IoT and Sensors*. *Sensors (Basel)*, 20(11), p. 3113. doi: 10.3390/s20113113.
34. Wang, L., Xu, S., Qiu, J., Wang, K., Ma, E., Li, C. and Guo, C. (2020) *Automatic Monitoring System in Underground Engineering Construction: Review and Prospect*. Available at: https:doi.org/10.1155/2020/3697253.
35. Werner-Allen, G., Lanzisera, S., Pister, K., & Weyer, T. (2005). Environmental monitoring using wireless sensor networks. *Proceedings of the 4th international symposium on information processing in sensor networks*, 407-414.
36. Wozniakowski-Zehenter, M. (2023) *Underground Mining Safety: Equipment Tracking Advancements and Implementation*. Available at: https://www.identecsolutions.com/news/underground-mining-safety-equipment-tracking-advancements-and-implementation.
37. Yick, J., Lee, P., & Hossain, M. A. (2008). Wireless sensor network survey: Technological challenges and applications. *ACM Computing Surveys (CSUR)*, *40*(1), 1-38.
38. Zhou, G. et al. (2015) *Node deployment of band-type wireless sensor network for underground coalmine tunnel*. *Computer Communications*. doi: http://dx.doi.org/10.1016/j.comcom.2015.10.015.
39. Zhou, G. (2018) *Recent advances in wireless underground sensor network*. doi: 10.1177/1550147718822840.

40. Zhu, C., Li, X., & Wang, L. (2010). A survey of key technologies in wireless sensor networks for assisted living. *Mobile networks and applications*, *15*, 615-626.
41. Zourmand, A., Hung, C.W., Hing, A.L.K. and AbdulRehman, M. (2019) *Internet of Things (IoT) using LoRa technology*. In: *2019 IEEE International Conference on Automatic Control and Intelligent^2 Systems (I2CACIS 2019)*, 29 June 2019, Selangor, Malaysia.

Appendices

Code for Node 1

```
/*********
MISACOMS Master's Project | ESP Now Data sender
*********/

#include <esp_now.h>
#include <WiFi.h>
#include <DHT.h>
// REPLACE WITH THE RECEIVER'S MAC Address
uint8_t broadcastAddress[] = {0x78, 0x21, 0x84, 0xA0, 0x58, 0x98};

//LED Controls
int LEDpin = 32;
int brightness = 0;  // how bright the LED is
int fadeAmount = 5;  // how many points to fade the LED by

#define DHTPIN 25
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);
// Structure example to send data
// Must match the receiver structure
typedef struct struct_message {
    int id; // must be unique for each sender board
    float temp;
    float hum;
} struct_message;

// Create a struct_message called myData
struct_message myData;
```

```
//Create the Temp and Humiditty Functions
float readDHTTemperature() {
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit (isFahrenheit = true)
  //float t = dht.readTemperature(true);
  // Check if any reads failed and exit early (to try again).
  if (isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");
    return 0;
  }
  else {
    Serial.println(t);
    return t;
  }
}

float readDHTHumidity() {
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  if (isnan(h)) {
    Serial.println("Failed to read from DHT sensor!");
    return 0;
  }
  else {
    Serial.println(h);
    return h;
  }
}

// Create peer interface
esp_now_peer_info_t peerInfo;
```

```
// callback when data is sent
void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status) {
  Serial.print("\r\nLast Packet Send Status:\t");
  Serial.println(status == ESP_NOW_SEND_SUCCESS ? "Delivery Success" : "Delivery Fail");
}

void setup() {
  // Init Serial Monitor
  Serial.begin(115200);
  dht.begin();

  // Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);

  // Init ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Once ESPNow is successfully Init, we will register for Send CB to
  // get the status of Trasnmitted packet
  esp_now_register_send_cb(OnDataSent);
  pinMode(LEDpin, OUTPUT);

  // Register peer
  memcpy(peerInfo.peer_addr, broadcastAddress, 6);
  peerInfo.channel = 0;
  peerInfo.encrypt = false;

  // Add peer
  if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Failed to add peer");
```

```
    return;
  }
}

void loop() {
  // Set values to send
  myData.id = 1;
  myData.temp =  readDHTTemperature();
  myData.hum = readDHTHumidity();

  // Send message via ESP-NOW
  esp_err_t result = esp_now_send(broadcastAddress, (uint8_t *) &myData, sizeof(myData));

  if (result == ESP_OK) {
    Serial.println("Sent with success");
    analogWrite(LEDpin, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(300);

  }
  else {
    Serial.println("Error sending the data");
  }
  delay(10000);
}
```

Code for Node 2

```
/*********
MISACOMS Master's Project | ESP Now Data sender
*********/

#include <esp_now.h>
#include <WiFi.h>
#include <DHT.h>
// REPLACE WITH THE RECEIVER'S MAC Address
uint8_t broadcastAddress[] = {0x78, 0x21, 0x84, 0xA0, 0x58, 0x98};

//LED Controls
int LEDpin = 32;
int brightness = 0;  // how bright the LED is
int fadeAmount = 5;  // how many points to fade the LED by

#define DHTPIN 25
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);
// Structure example to send data
// Must match the receiver structure
typedef struct struct_message {
    int id; // must be unique for each sender board
    float temp;
    float hum;
} struct_message;

// Create a struct_message called myData
struct_message myData;

//Create the Temp and Humiditty Functions
float readDHTTemperature() {
```

```
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit (isFahrenheit = true)
  //float t = dht.readTemperature(true);
  // Check if any reads failed and exit early (to try again).
  if (isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");
    return 0;
  }
  else {
    Serial.println(t);
    return t;
  }
}

float readDHTHumidity() {
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  if (isnan(h)) {
    Serial.println("Failed to read from DHT sensor!");
    return 0;
  }
  else {
    Serial.println(h);
    return h;
  }
}

// Create peer interface
esp_now_peer_info_t peerInfo;

// callback when data is sent
void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status) {
```

```
  Serial.print("\r\nLast Packet Send Status:\t");
  Serial.println(status == ESP_NOW_SEND_SUCCESS ? "Delivery Success" : "Delivery Fail");
}

void setup() {
  // Init Serial Monitor
  Serial.begin(115200);
  dht.begin();

  // Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);

  // Init ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Once ESPNow is successfully Init, we will register for Send CB to
  // get the status of Trasnmitted packet
  esp_now_register_send_cb(OnDataSent);
  pinMode(LEDpin, OUTPUT);

  // Register peer
  memcpy(peerInfo.peer_addr, broadcastAddress, 6);
  peerInfo.channel = 0;
  peerInfo.encrypt = false;

  // Add peer
  if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Failed to add peer");
    return;
  }
}
```

```
void loop() {
  // Set values to send
  myData.id = 2;
  myData.temp =  readDHTTemperature();
  myData.hum = readDHTHumidity();

  // Send message via ESP-NOW
  esp_err_t result = esp_now_send(broadcastAddress, (uint8_t *) &myData, sizeof(myData));

  if (result == ESP_OK) {
    Serial.println("Sent with success");
    analogWrite(LEDpin, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(300);

  }
  else {
    Serial.println("Error sending the data");
  }
  delay(10000);
}
```

Code for TR-Main

```
#include <esp_now.h>
#include <WiFi.h>
//Libraries for LoRa
#include <SPI.h>
#include <LoRa.h>

//Libraries for OLED Display
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

//define the pins used by the LoRa transceiver module
#define SCK 5
#define MISO 19
#define MOSI 27
#define SS 18
#define RST 14
#define DIO0 26

//433E6 for Asia
//866E6 for Europe
//915E6 for North America
#define BAND 866E6

//OLED pins
#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

// Structure example to receive data
```

```
// Must match the sender structure
typedef struct struct_message {
  int id;
  float temp;
  float hum;
}struct_message;

// Create a struct_message called myData
struct_message myData;

// Create a structure to hold the readings from each board
struct_message board1;
struct_message board2;

// Create an array with all the structures
struct_message boardsStruct[2] = {board1, board2};

// callback function that will be executed when data is received
void OnDataRecv(const uint8_t * mac_addr, const uint8_t *incomingData, int len) {
  char macStr[18];
  Serial.print("Packet received from: ");
  snprintf(macStr, sizeof(macStr), "%02x:%02x:%02x:%02x:%02x:%02x",
           mac_addr[0], mac_addr[1], mac_addr[2], mac_addr[3], mac_addr[4], mac_addr[5]);
  Serial.println(macStr);
  memcpy(&myData, incomingData, sizeof(myData));
  Serial.printf("Board ID %u: %u bytes\n", myData.id, len);
  // Update the structures with the new incoming data
  boardsStruct[myData.id-1].temp = myData.temp;
  boardsStruct[myData.id-1].hum = myData.hum;
  Serial.printf("Temp value: %4.2f \n", boardsStruct[myData.id-1].temp);
  Serial.printf("Humidity value: %4.2f \n", boardsStruct[myData.id-1].hum);
  Serial.println();
}
```

```
//packet counter
int readingID = 0;
int counter =0;

String LoRaMessage = "";

float temperature = 0;
float humidity = 0;
float temperature1 = 0;
float humidity1 = 0;

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST);

//Initialize OLED display
void startOLED(){
  //reset OLED display via software
  pinMode(OLED_RST, OUTPUT);
  digitalWrite(OLED_RST, LOW);
  delay(20);
  digitalWrite(OLED_RST, HIGH);

  //initialize OLED
  Wire.begin(OLED_SDA, OLED_SCL);
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) { // Address 0x3C for 128x32
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever
  }
  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(1);
  display.setCursor(0,0);
  display.print("LORA SENDER");
}
```

```
void getReadings(){
  temperature = boardsStruct[0].temp;
  humidity = boardsStruct[0].hum;
 temperature1 = boardsStruct[1].temp;
  humidity1 = boardsStruct[1].hum;
}


//Initialize LoRa module
void startLoRA(){
  //SPI LoRa pins
  SPI.begin(SCK, MISO, MOSI, SS);
  //setup LoRa transceiver module
  LoRa.setPins(SS, RST, DIO0);
  LoRa.setSyncWord(0xF4);
  while (!LoRa.begin(BAND) && counter < 10) {
    Serial.print(".");
    counter++;
    delay(500);
  }
  if (counter == 10) {
    // Increment readingID on every new reading
    readingID++;
    Serial.println("Starting LoRa failed!");
  }
  Serial.println("LoRa Initialization OK!");
  display.setCursor(0,10);
  display.clearDisplay();
  display.print("LoRa Initializing OK!");
  display.display();
  delay(2000);
}

void sendReadings() {
```

```
  LoRaMessage = String(readingID) + "/" + String(temperature) + "&" + String(humidity) + "#" +
String(temperature1)+ "*" + String(humidity1);
  //Send LoRa packet to receiver
  LoRa.beginPacket();
  LoRa.print(LoRaMessage);
  LoRa.endPacket();

  display.clearDisplay();
  display.setCursor(0,0);
  display.setTextSize(1);
  display.print("MISACOMS R1");
  display.setCursor(0,20);
  display.print("Temp 1:");
  display.setCursor(72,20);
  display.print(temperature);
  display.setCursor(0,30);
  display.print("Humidity 1:");
  display.setCursor(54,30);
  display.print(humidity);
  display.setCursor(0,40);
  display.print("Temp 2:");
  display.setCursor(54,40);
  display.print(temperature1);
  display.setCursor(0,50);
  display.print("Humidity 2:");
  display.setCursor(66,50);
  display.print(humidity1);
  display.display();
  Serial.print("Sending packet: ");
  Serial.println(readingID);
  readingID++;
}

void setup() {
```

```
  //Initialize Serial Monitor
  Serial.begin(115200);

  //Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);

  //Init ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Once ESPNow is successfully Init, we will register for recv CB to
  // get recv packer info
  esp_now_register_recv_cb(OnDataRecv);
  startOLED();
  startLoRA();

}

void loop() {
  // Acess the variables for each board
  /*int board1X = boardsStruct[0].x;
  int board1Y = boardsStruct[0].y;
  int board2X = boardsStruct[1].x;
  int board2Y = boardsStruct[1].y;
  int board3X = boardsStruct[2].x;
  int board3Y = boardsStruct[2].y;*/
  getReadings();
  sendReadings();

  delay(10000);
}
```

Code for TR-1

```
/*********
 MISACOMS TRANSCEIVER 1
*********/

//Libraries for LoRa
#include <SPI.h>
#include <LoRa.h>

//Libraries for OLED Display
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

//define the pins used by the LoRa transceiver module
#define SCK 5
#define MISO 19
#define MOSI 27
#define SS 18
#define RST 14
#define DIO0 26

//433E6 for Asia
//866E6 for Europe
//915E6 for North America
#define BAND 866E6

//OLED pins
#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
```

```
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST);

String LoRaData;

void setup() {
  //initialize Serial Monitor
  Serial.begin(115200);

  //reset OLED display via software
  pinMode(OLED_RST, OUTPUT);
  digitalWrite(OLED_RST, LOW);
  delay(20);
  digitalWrite(OLED_RST, HIGH);

  //initialize OLED
  Wire.begin(OLED_SDA, OLED_SCL);
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) { // Address 0x3C for 128x32
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever
  }

  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(1);
  display.setCursor(0,0);
  display.print("LORA RECEIVER ");
  display.display();

  Serial.println("LoRa Receiver Test");

  //SPI LoRa pins
  SPI.begin(SCK, MISO, MOSI, SS);
  //setup LoRa transceiver module
```

```
  LoRa.setPins(SS, RST, DIO0);

  if (!LoRa.begin(BAND)) {
    Serial.println("Starting LoRa failed!");
    while (1);
  }
  LoRa.setSyncWord(0xF4);
  Serial.println("LoRa Initializing OK!");
  display.setCursor(0,10);
  display.println("LoRa Initializing OK!");
  display.display();
}

void loop() {

  //try to parse packet
  int packetSize = LoRa.parsePacket();
  if (packetSize) {
    //received a packet
    Serial.print("Received packet ");

    //read packet
    while (LoRa.available()) {
      LoRaData = LoRa.readString();
      Serial.print(LoRaData);
      LoRa.setSyncWord(0xF4);
      //Send LoRa packet to receiver
        LoRa.beginPacket();
        LoRa.print(LoRaData);
        LoRa.endPacket();

    }

    //print RSSI of packet
```

```
    int rssi = LoRa.packetRssi();
    Serial.print(" with RSSI ");
    Serial.println(rssi);

   // Dsiplay information
   display.clearDisplay();
   display.setCursor(0,0);
   display.print("MISACOMS TR1");
   display.setCursor(0,20);
   display.print("Received from TR-Main:");
   display.setCursor(0,30);
   display.print(LoRaData);
   display.setCursor(0,40);
   display.print("RSSI:");
   display.setCursor(30,40);
   display.print(rssi);
   display.display();
  }
}
```

Code for TR-2

```
/*********
 MISACOMS TRANSCEIVER 2
*********/

//Libraries for LoRa
#include <SPI.h>
#include <LoRa.h>

//Libraries for OLED Display
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

//define the pins used by the LoRa transceiver module
#define SCK 5
#define MISO 19
#define MOSI 27
#define SS 18
#define RST 14
#define DIO0 26

//433E6 for Asia
//866E6 for Europe
//915E6 for North America
#define BAND 866E6

//OLED pins
#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
```

```
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST);

String LoRaData;

void setup() {
  //initialize Serial Monitor
  Serial.begin(115200);

  //reset OLED display via software
  pinMode(OLED_RST, OUTPUT);
  digitalWrite(OLED_RST, LOW);
  delay(20);
  digitalWrite(OLED_RST, HIGH);

  //initialize OLED
  Wire.begin(OLED_SDA, OLED_SCL);
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) { // Address 0x3C for 128x32
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever
  }

  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(1);
  display.setCursor(0,0);
  display.print("LORA RECEIVER ");
  display.display();

  Serial.println("LoRa Receiver Test");

  //SPI LoRa pins
  SPI.begin(SCK, MISO, MOSI, SS);
  //setup LoRa transceiver module
```

```
  LoRa.setPins(SS, RST, DIO0);

  if (!LoRa.begin(BAND)) {
    Serial.println("Starting LoRa failed!");
    while (1);
  }
  LoRa.setSyncWord(0xF5);
  Serial.println("LoRa Initializing OK!");
  display.setCursor(0,10);
  display.println("LoRa Initializing OK!");
  display.display();
}

void loop() {

  //try to parse packet
  int packetSize = LoRa.parsePacket();
  if (packetSize) {
    //received a packet
    Serial.print("Received packet ");

    //read packet
    while (LoRa.available()) {
      LoRaData = LoRa.readString();
      Serial.print(LoRaData);
      LoRa.setSyncWord(0xF6);
      //Send LoRa packet to receiver
        LoRa.beginPacket();
        LoRa.print(LoRaData);
        LoRa.endPacket();

    }

    //print RSSI of packet
```

```
    int rssi = LoRa.packetRssi();
    Serial.print(" with RSSI ");
    Serial.println(rssi);

   // Dsiplay information
   display.clearDisplay();
   display.setCursor(0,0);
   display.print("MISACOMS TR2");
   display.setCursor(0,20);
   display.print("Received from TR1:");
   display.setCursor(0,30);
   display.print(LoRaData);
   display.setCursor(0,40);
   display.print("RSSI:");
   display.setCursor(30,40);
   display.print(rssi);
   display.display();
  }
}
```

Code for the Gateway

```
#define BLYNK_TEMPLATE_ID "TMPLIRNn9-Et"
#define BLYNK_DEVICE_NAME "Quickstart Template"

#include <WiFi.h>
#include <WiFiClient.h>
#include <analogWrite.h>
#include <BlynkSimpleEsp32.h>

//Libraries for LoRa
#include <SPI.h>
#include <LoRa.h>

//Libraries for OLED Display
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

// Go to the Project Settings (nut icon).
char auth[] = "hjyLub6BDyunOdm0dW9_yY99qXdzmV4G";

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "networkSSID";
char pass[] = "networkPass";
BlynkTimer timer;

//define the pins used by the LoRa transceiver module
#define SCK 5
#define MISO 19
#define MOSI 27
#define SS 18
#define RST 14
```

```
#define DIO0 26

//433E6 for Asia
//866E6 for Europe
//915E6 for North America
#define BAND 866E6

//OLED pins
#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST);
String loRaMessage;
String temp1s;
String temp2s;
String hum1s;
String hum2s;
String readingsID;
float temp1; //Temp from Node 1
float temp2; //Temp from Node 2
float hum1;//Humidity from Node 1
float hum2;//Humidity from Node 2
float rsdata;// RSSI details between TR2 and the gateway
int LEDpin = 32;

//Initialize OLED display
void startOLED(){
  //reset OLED display via software
  pinMode(OLED_RST, OUTPUT);
  digitalWrite(OLED_RST, LOW);
  delay(20);
```

```
  digitalWrite(OLED_RST, HIGH);

  //initialize OLED
  Wire.begin(OLED_SDA, OLED_SCL);
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) { // Address 0x3C for 128x32
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever
  }
  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(1);
  display.setCursor(0,0);
  display.print("MISACOMS Gateway");
}

//Initialize LoRa module
void startLoRA(){
  int counter;
  //SPI LoRa pins
  SPI.begin(SCK, MISO, MOSI, SS);
  //setup LoRa transceiver module
  LoRa.setPins(SS, RST, DIO0);
  //LoRa Unique ID
  LoRa.setSyncWord(0xF6);

  while (!LoRa.begin(BAND) && counter < 10) {
    Serial.print(".");
    counter++;
    delay(500);
  }
  if (counter == 10) {
    // Increment readingID on every new reading
    Serial.println("Starting LoRa failed!");
  }
```

```
  Serial.println("LoRa Initialization OK!");
  display.setCursor(0,10);
  display.clearDisplay();
  display.print("LoRa Initializing OK!");
  display.display();
  delay(2000);
}

// Read LoRa packet and get the sensor readings
void getLoRaData() {
  Serial.print("Lora packet received: ");
  // Read packet
  while (LoRa.available()) {
    String LoRaData = LoRa.readString();
    // LoRaData format: readingID/temperature&humidity#temp2*hum2
    // String example: 1/27.43&654#95.34
    Serial.print(LoRaData);

    // Get readingID, temperature and humidity
    int pos1 = LoRaData.indexOf('/');
    int pos2 = LoRaData.indexOf('&');
    int pos3 = LoRaData.indexOf('#');
    int pos4 = LoRaData.indexOf('*');
    readingsID = LoRaData.substring(0, pos1);
    temp1s = LoRaData.substring(pos1 +1, pos2);
    hum1s = LoRaData.substring(pos2+1, pos3);
    temp2s = LoRaData.substring(pos3+1, pos4);
    hum2s = LoRaData.substring(pos4+1, LoRaData.length());
    temp1 = temp1s.toFloat();
    temp2 = temp2s.toFloat();
    hum1 = hum1s.toFloat();
    hum2 = hum2s.toFloat();
  }
  // Get RSSI
```

```
  rsdata = LoRa.packetRssi();
  Serial.print(" with RSSI ");
  Serial.println(rsdata);
  Serial.println(readingsID);
  Serial.println(temp1);
  Serial.println(hum1);
  Serial.println(temp2);
  Serial.println(hum2);
}

//Blynk Data Sending funcs

BLYNK_WRITE(V1){
  int pinValue = param.asInt();

  //map -> (variable, fromLow, fromHigh, toLow, toHigh)
  int newValue = map(pinValue,0 ,100,0,255);
  analogWrite(LEDpin, newValue);

  Serial.println(pinValue);
}

BLYNK_WRITE(V2){
  int pinValue = param.asInt();
  if(pinValue==1)analogWrite(LEDpin, 255);
  else if(pinValue==0)analogWrite(LEDpin,0);

  Serial.println(pinValue);
}

void sendDataToBlynk(){
  float t = temp1;
  float h = hum1;
```

```
 Blynk.virtualWrite(V3, t);
 Blynk.virtualWrite(V4, h);

  float t2 = temp2;
  float h2 = hum2;

 Blynk.virtualWrite(V5, t2);
 Blynk.virtualWrite(V6, h2);
}
void setup() {
  //Initialize Serial Monitor
  Serial.begin(115200);

  //Start Blynk connection
  Blynk.begin(auth, ssid, pass);

  //initialize sensor data
  startOLED();
  startLoRA();

  timer.setInterval(7000L, sendDataToBlynk);
}

void loop() {

  Blynk.run();
  timer.run();
  int packetSize = LoRa.parsePacket();
  if (packetSize) {
    getLoRaData();
  }
}
```