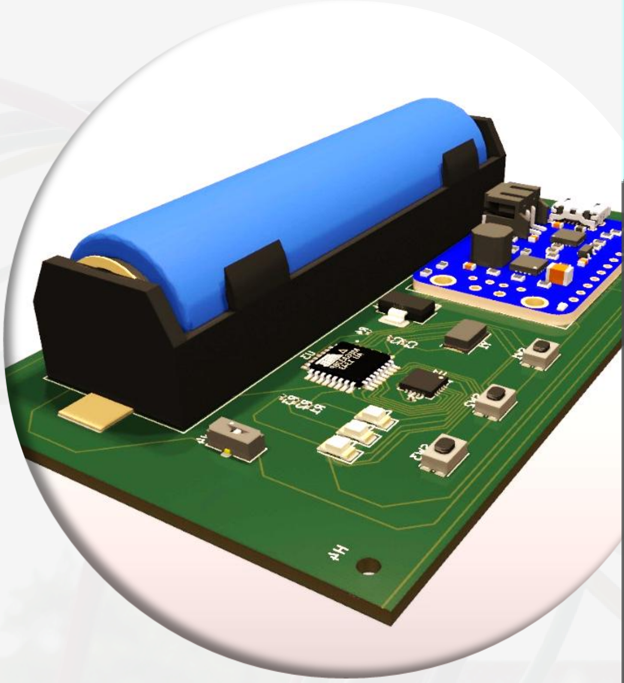
Faculty of Electrical Engineering

UNIVERSITI TEKNOLOGI MALAYSIA



ACADEMIC SESSION

2024/2025

SKEE1103

C PROGRAMMING FOR ENGINEERS

GROUP PROJECT

P2P MESH COMMUNICATION LEARNING KIT

SECTION 02

LECTURER: DR. NUR FAIZAL BIN KASRI

PREPARED BY:

NAME	MATRIC NO.
CHIN YUAN YI	A24KE5081
KOK QI SHENG	A24KE0121
LIM ENG CHIEN	A24KE0138
OOI WEY SHEN	A24KE0270
TONG JIA SHEN	A24KE0313

Team Description

TrekAlert

Project Director & Hardware Implementation

Kok Qi Sheng

"Led the team and constructed the hardware system, ensuring all components were properly assembled and fully functional"

Schematic Drawing & Software Simulation

Lim Eng Chien

"Created the schematic diagrams and implemented software simulations to verify the design prior to physical assembly"

Assembly & Operational Manual

Chin Yuan Yi
Tong Jia Shen

"Carried out the hardware assembly process and contributed to writing the instructions manual to ensure proper setup and usage for users"

Documentation

Ooi Wey Shen

"Conducted research towards the theoretical background and prepared a clear documentation for the report"

Table of Contents

1.0 Introduction

For this project, our main targeted users were particularly the high school students who are able to comprehend most basic **peer-to-peer mesh communication** knowledge which often included in STEM subject syllabus such as **Computer Science** and **Physics** subject.

Other than that, students are also exposed to an exciting involvement in **Arduino kit development**, lies under the **usage of microcontrollers**, which is a foundational skillset for their ***Reka Bentuk Teknologi*** subject that includes hardware and software implementations.

In short, we want student to head the ground up, **build** up a Peer-to-Peer Mesh Communication **Device** on their own with **given instructions manual**, enhance their practical skills on utilise a microcontroller and **strengthen** their **understanding** towards **wireless communication protocols**.

1.1 Problem Statement

In rural or low signal areas such as **jungle, mountain trails, maritime** or disaster zones, the conventional communication device including mobile phones required **cellular network** which might no longer available in service.

This presents a huge challenge in terms of team coordination and emergency communication. To address this problem, we introduced the product, **TrekAlert**, equipped with **Peer-to-Peer (P2P) Mesh** Communication Technology using **Radio Frequency (RF)** Wireless Communication Protocol.

RF technology is being targeted to outperform other wireless communication protocols such as **WiFi, Bluetooth, Cellular Networks, AM/FM** in this specific use case. By utilising this technology as **P2P transceiver** medium, the nodes can communicate between devices without the aid of cellular network.

1.2 Objectives

Develop a hands-on learning kit to demonstrate P2P mesh communication principles
The kit simplifies the concepts of peer-to-peer and mesh communication. This allows students to observe how devices transmit and receive messages across the network without a central hub.

Enable students to design, assemble and program a working device
This kit empowers students to go beyond theory by requiring them to assemble hardware components and getting exposed with firmware that controls the communication nodes. The following will provide them practical experiences with basic microcontroller programming and digital electronic system.

Demonstrate safe, warning and emergency signal propagation without relying on cellular networks
The kit simulates real-world scenarios where conventional communication networks may be unavailable. Students can explore how independent nodes share status updates across the network by sending and receiving predefined signal, in this case, **SAFE**, **WARNING** and **SOS**

1.3 Scope

This project focuses on the **study of mechanism, design, implementation and demonstration** of a peer-to-peer (P2P) mesh communication kit with **Arduino Nano** and **nRF24L01+ module**. The scope includes:

- Developing hardware and software that enable **basic signal transmission** (Safe, Warning, SOS) between multiple nodes.
- Providing hands-on **assembly instructions**, **testing procedures** and a **ready-to-use educational kit**.
- Exploring **educational aspects**, scale down the conventional devices, allowing students to gain **practical experience** with microcontrollers and wireless communication

This scope ensures the kit is **suitable for classroom and beginner use**, while laying the groundwork for future enhancements, such as extended range (LoRa, Zigbee) or graphical displays. For more details, refer future works being discussed in the latter section of the report.

2.0 Literature Review

In a real-world application, this kit can be used in various scenarios that lack of cellular network and internet. This includes but not limited to **jungle trekking**, **island explorations** and **maritime rescue**.

Peer-to-peer and mesh wireless communication using **LoRa, Zigbee** and similar radio technologies has been widely adopted in professional and industry sectors among off-grid applications. For example, military use devices like **goTenna Pro X** [1] to maintain communication without cellular networks during search and rescue operations. **Agricultural** sectors also deploy **LoRa**-based mesh networks, specifically **LoRaWAN (Low Power Wide Area Network)** [2] for remote sensing and automation, such as **monitoring soil moisture** and **water management** across large outdoor areas. For more details, see [3].

Despite these established industrial uses, peer-to-peer mesh radio frequency technology **remains niche** outside specialized domains. Consumer devices rarely feature these protocols and most personal wireless communication **still highly depends on cellular**, **Wi-Fi, or Bluetooth**.

2.1 Overview of Decentralised Mesh Communication Technologies

Mesh networking allows **devices** (which are also known as **nodes**) to communicate with each other **directly** or **indirectly via intermediate nodes**. Meanwhile, **data** are often **encrypted** with AES-256 encryption method and routed through **secure paths** among the mesh topology, hence ensure the **data** integrity when transferring between nodes, prevent intermediate nodes from accessing other nodes' message transfer process [4]

Traditional Wi-Fi technology also includes mesh solution, however from a topper view, it still relies on central ISP (Internet Service Provider) thus it is categorised under centralised mesh solution. In this subsection, mainly **decentralised** mesh solution will be discussed, especially **Radio Frequency** and **Bluetooth**. **Several decentralised mesh communications technologies have been developed** for embedded system and IoT applications:

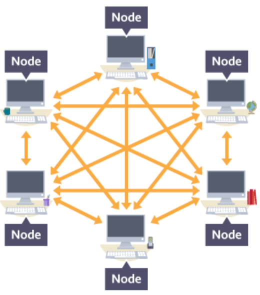


Figure 1: Node in General Mesh Networking. Adapted from [5]

Protocol	Range & Data Rate	Cost & Complexity	Suitability
Zigbee [6]	Short (~10–100 m), moderate data rate	Expensive modules, requires coordinator, complex setup	Common in home automation; too complex for most beginners
LoRa [7]	Long (~10 km), very low data rate	Expensive, low bandwidth, not real-time	Good for long-range, low updates; unsuitable for responsive projects
BLE Mesh [8]	Short (~10–50 m), moderate data rate	Requires significant memory and setup	Large-scale device comms; too advanced

			for simple Arduino projects
nRF24L01+ [9]	Short (~10–100 m), moderate data rate	Very low cost, simple to use, low power	Beginner-friendly P2P; can emulate mesh via custom relay or tree-based routing

Table 2

2.2 Justification for nRF24L01+ Kit

After careful consideration and analysis, the **Arduino + nRF24L01+** is the most ideal solution for this **P2P Mesh Communication Educational Kit** due to several reasons:

- Low cost
- Scale down a conventional use case of **Radio Frequency** Technology, rather than using a Bluetooth technology
- Beginner friendly, enable students' hands-on interaction on both hardware and software for educational purposes
- Energy efficient, only consumes a low power, suitable for **TrekAlert** which focus as a pocket tool

3.0 Methodology

This section describes the systematic approach taken to design, implement and verify the **TrekAlert P2P Mesh Communication Learning Kit** as a hands-on learning tool for high school students. The methodology integrates educational learning outcomes with a practical engineering design process to ensure the kit is beginner friendly and low learning curve towards technical aspects.

3.1 Problem Formulation

The primary objective is to introduce students to RF wireless peer-to-peer (P2P) decentralised mesh communication, a concept often overlooked in traditional STEM curriculum that focus mostly on Wi-Fi, cellular networks or Bluetooth that most of them rely on the central infrastructure.

Hypothetically, if students can successfully build the kit using provided hardware and instruction manual, able to carry out sending and receive messages across a small RF mesh network, then the kit is an effective educational tool for demonstrating decentralised mesh communication.

Educational goals:

- Develop students' practical experiences in hardware assembly, microcontroller programming and basic wireless communication protocol
- Strengthen understanding towards RF technology, decentralised P2P mesh communication and logic level manipulation in microcontroller programming
- Encourage and inspire students to excavate the field of communication, contribute to the STEM environment

3.2 Hardware and Firmware Design

3.2.1 Hardware Design

We selected mainly the low-cost, beginner-friendly components so student can afford them:

- Arduino Nano as control board
- nRF24L01+ transceiver for P2P RF messaging
- LEDs to indicate nodes' status
- Battery powered for a better portability

3.2.2 Firmware Development

Firmware was developed through Arduino code and plugged in with RF24 library that simplifies the wireless messaging between nodes by automating acknowledgement and retransmission process.

Each node carries out a cycle on checking message transmission and reception status, communicate using SAFE/WARNING/SOS states. The firmware source code was well commented and named with proper variables that ensure students can grasp the logic easily and make modifications on top of it for further development. The instruction manual maps the hardware stage to the firmware stage in detail so students can clearly see how each part corresponds to the firmware.

3.3 Instructional Materials

A step-by-step instruction for assembling the kit and microcontroller operation mode was well illustrated and discussed, that includes:

- Wiring the components by following the graphical steps, according to the fritzing schematics
- When assembling the components, understand how active low components were hardware-defined, learn the proper method to setup for different configuration modes
- Installing the firmware source code from GitHub
- Verify and upload firmware to node's microcontroller chip

3.4 Testing and Verification

With two nodes powered on at short range (~15m), students:

- Sent and received predefined messages between the two nodes
- Confirmed successful transceive capability and bidirectional communication
- Verified that communication was achieved without the aid of cellular networks or internet
- Verified nodes' P2P decentralised mesh communication capability

With the open-source access provided through GitHub, students may always revise the Learning Kit details and modify its usage, carry out further study on theoretical concepts such as radio frequency technology and P2P networks independently

3.5 Iteration

The educators can always visit our project repository, follow the same implementation methodology, they could provide the educational session with their targets users, not limited to students only. Researchers or hobbyist can also access the whole project report, to support their own case study or self-guided learning

3.6 Limitations and Future Improvements

While this version only support demonstration for basic two nodes communication, future enhancements may include:

- Adding more nodes to showcase mesh network's relay capabilities
- Expand to a longer ranged RF network using LoRa technology
- Integrating GUI display for more interactive STEM educational demonstration

For more details on future enhancements, refer 8.0 Future Work section.

4.0 System Overview

4.1 Peer-to-Peer and Mesh Communication Principles

Each node in the network can function as both **a transmitter and a receiver** in the **independent** communication model and that refers to peer-to-peer (**P2P**) communication. P2P systems improve reliability and **autonomy** by **getting rid** with the need for a **central infrastructure**, in contrast to conventional client-server architectures. **In simple word, every node is the central hub itself.**

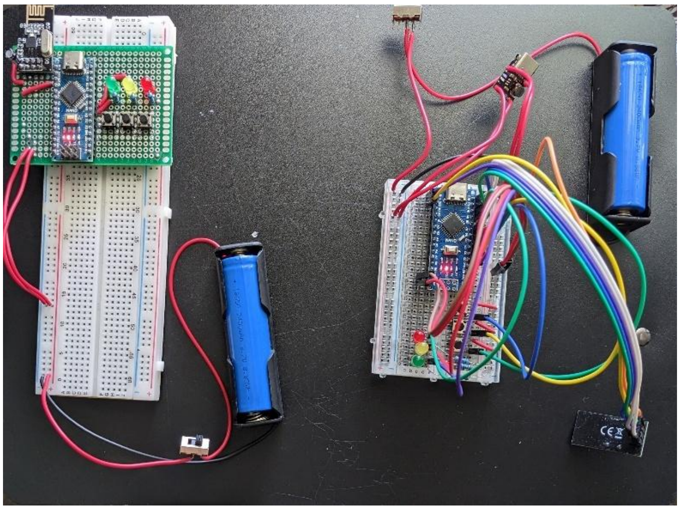


Figure 2: Two isolated node communication devices

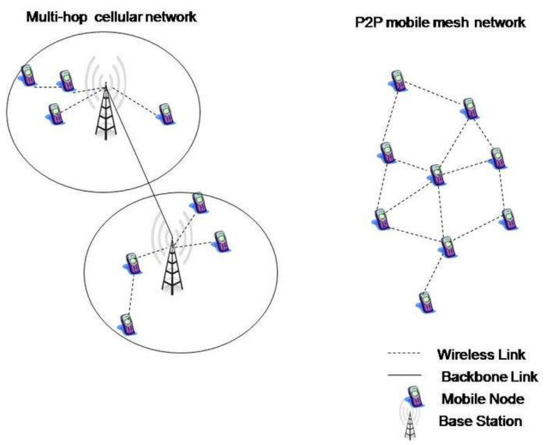


Figure 3: Comparison between traditional cellular network and P2P decentralised mesh network
Adapted from [10]

The P2P idea is extended by mesh communication, which enables several nodes to transmit messages through one another in addition to communicating directly. As a result, data can **travel** via a **variety of routes** to reach its destination, forming a **decentralised** network, enhancing:

- **Reliability: Alternative routes** can be used if one node fails.
- **Scalability:** The network can **dynamically add new nodes** as needed.
- **Coverage: Intermediate nodes** allow signal coverage to **expand wider**, act as a **signal repeater**

Mesh networking can be divided into two categories:

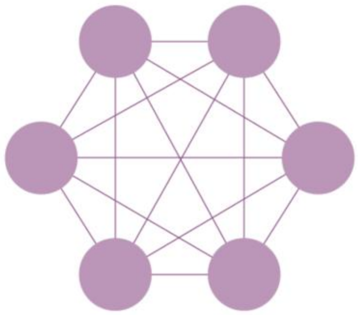
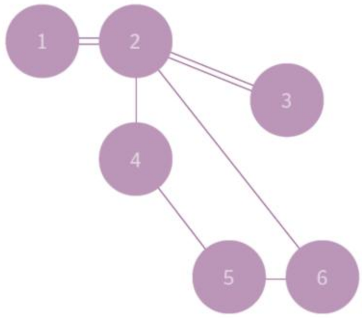
<table>
<tr><th>Full Mesh</th><th>Partial Mesh (Ad Hoc) [11]</th></tr>
<tr><td></td><td>
</td></tr>
<tr><td>This is the best option, but it requires a lot of work and resources because every node is directly connected to every other node.</td><td>This kit uses nodes that connect directly or indirectly, dependent on signal strength or distance. This technology is used by TrekAlert</td></tr>
</table>

Table 3, Mesh Network Topology, Diagrams adapted from [12]

4.2 TrekAlert Working Principle

The **TrekAlert** Communication System is a **portable rechargeable battery-operated Arduino-based** device that allows peer-to-peer (**P2P**) decentralised mesh communication. It enables users to use **push buttons** to **transceive** the three predefined status signals—**Safe**, **Warning** and **SOS** from a node to another **without cellular network**.

The **nRF24L01+** module **at first node** is used to wirelessly **broadcast the signals**, which are then **routed** throughout the **mesh network** and represented on **second transceiver nodes** by **color-coded LEDs**. On the other hand, the functionality as transmitter and receiver can be swapped between nodes, allowing a **bidirectional** communication, which make it outperform the traditional AM/FM transmission and reception.

This kit enables a group of users to communicate basic **safety statuses** through wireless signal propagation using the nRF24L01+ transceiver module in mesh networks, which **every node act as the central device** itself.

Each node in the system is an independent device that can carry out three major functions, which are:

- Sending status signals via push button, act as **transmitter**
- Receiving signals from other nodes, act as **receiver**
- Communicate with each other **bidirectionally** through various colours of LEDs with **real-time** operation, act as **transceiver**

Three predefined signals can be transmitted by triggering the respective push button through this kit (Table 1):

No.	**LED**	**Message**
1	**Green**	SAFE (Clear)
2	**Yellow**	WARNING (Caution)
3	**Red**	SOS (Emergency)

Table 1

5.0 Hardware Description, Designs and Implementation

For hardware integration, we have dissected into a few operational parts, mainly:

- Processing Unit
- RF Transceiver
- I/O Device
- Power Supply

5.1 Processing Unit: Arduino Nano

The **central processing unit (CPU)** in each **TrekAlert** node is the **Arduino Nano**. It controls **push button input detection**, **RF wireless communication** with nRF24l01+ via SPI Protocol and **LED signalling**. Arduino Nano is best suited for **TrekAlert** because of its small size, as compared to Arduino UNO, its USB-C programmability, low power consumption and sufficient amount of digital I/O pins makes it an ideal choice for this project.

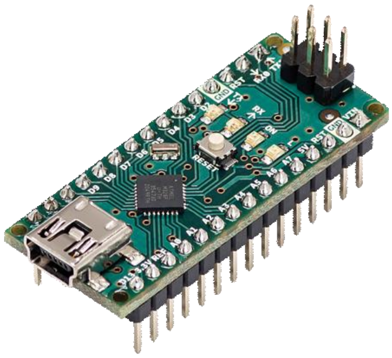


Figure 4: Arduino Nano. Adapted from [13]

Arduino Nano is based on **ATMega328P** 8-bit AVR microprocessor. It runs with a clock speed of 16 MHz and operates at a controlled 5 V in this case. Arduino Nano equipped with **14 digital** input/output **(I/O) pins** (D0 – D13) and **8 analog input pins (A0 – A7).** It also supports communication between nRF24L01+ module via SPI hardware protocol

Main function of **microcontroller** in **Trek**:

- **Input Monitoring**
  Each push button is wired with a pull-up resistor so that the corresponding Nano input pin stays **HIGH** by default. When a button is pressed, the input pin goes **LOW**, signalling the microcontroller to begin processing.

- **Signal Encoding**
  When a button is pressed, the Nano **encodes a message** to SPI protocol, containing:
  - The **signal type** (which button was pressed)
  - The **node ID** (which device is sending)
  - A **message ID** (to help prevent duplicates)

- **Data Transmission and Reception**
  Neighbouring nodes receive the transmitted message through their nRF24L01+ modules over **SPI protocol**. Each node checks if the message is new, then processes the signal, e.g. triggering a button causes another node to light up an LED.

5.2 Transceiver Module: nRF24L01+

The **nRF24L01+** is **a popular wireless transceiver module** that enables communication between microcontrollers. The **compact** and **energy-efficient** characteristics, making it easily integrated into **educational** and **embedded projects** that required **RF wireless communication**. Its **firmware** provided features like **automated handshaking** and **retransmission** [14], hence making its **setup** easier and **beginner friendly**. Moreover, there are also versions with **enhanced antennas** available in the market, providing an **extended range** and **stability** option for more demanding usages. [15]

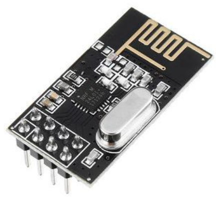


Figure 5: nRF24L01+

The following are the key **specifications** of the nRF24L01+:

Feature	**Description**
Frequency Band	2.4 GHz ISM
Data Rate	Up to 2 Mbps
Transmission Range	50–100 m (open space), varies with environment
Communication Protocol	SPI (Serial Peripheral Interface)
Acknowledgment Support	Auto-ACK and Auto Retransmit
Data Pipes	6 simultaneous data pipes
Payload Size	Up to 32 bytes per message
External Antenna Option	Available for increased range and reliability
Power Consumption	Very low, ideal for battery-powered devices
Common Applications	Educational kits, embedded systems, P2P or mesh-like networks

Table 4: nRF24L01+ specifications. For more details, see [16].

SPI (Serial Peripheral Interface) and Its Role

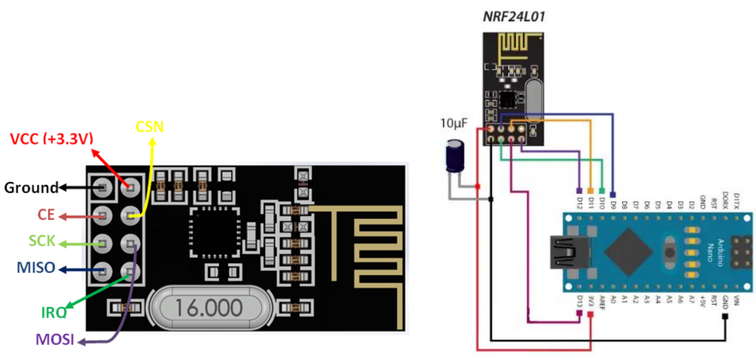


Figure 6: nRF24L01+ Pinout. Adapted from [9]

The **Arduino Nano**, a **microcontroller** act as a **Master** device and the **nRF24L01+** wireless module, a **peripheral** act as a **Slave** device can exchange data via the **Serial Peripheral Interface (SPI)** and it supports the core functionality for **TrekAlert**, as it is a **full-duplex** comfmunication protocol.

Pin	**Full Name**	**Direction**	**Description**
MOSI	Master Out, Slave In	Master → Slave	Sends command bytes and data from the microcontroller to the nRF24L01+
MISO	Master In, Slave Out	Slave → Master	Sends data from the nRF24L01+ back to the microcontroller
SCK	Serial Clock	Master → Slave	Synchronizes data transmission
CSN	Chip Select Not	Active Low	Enables (LOW) or disables (HIGH) communication
CE	Chip Enable	Active High	Enables the transceiver to transmit or receive data

Table 5: nRF24L01+ pinout details (SPI Interfrace). Adapted from [17]

1. **Initialisation**
   When the system is powered on:
   - The Arduino initialises the nRF24L01+ by writing values to its internal configuration registers over SPI.
   - This setup includes:
     - Payload size
     - Transmission power level
     - Communication channel (frequency)
     - CRC (Cyclic Redundancy Check) error validation and automated acknowledgement

2. **Transmission**
   - The **Arduino** generates the data packet and writes it into the nRF24L01+'s TX FIFO buffer using the SPI protocol.
   - The **CE (Chip Enable)** pin is briefly pulled HIGH to begin transmitting after the data is loaded.
   - The signal is subsequently transmitted by the NRF module using 2.4 GHz radio waves.

3. **Reception**
   - On the receiving node, the **nRF24L01+** listens in RX mode and sets an interrupt flag upon receiving a valid packet.
   - The **Arduino** then reads the data from the RX FIFO buffer using SPI commands.
   - After decoding the message, the microcontroller initiates the relevant output action (e.g. turning on the LED to indicate the message).

4. **Role of CSN**
   - The **CSN** pin is pulled LOW to start SPI communication and HIGH to stop it.
   - This guarantees that only one device is active on the SPI bus at a time.

5.3 Power Supply

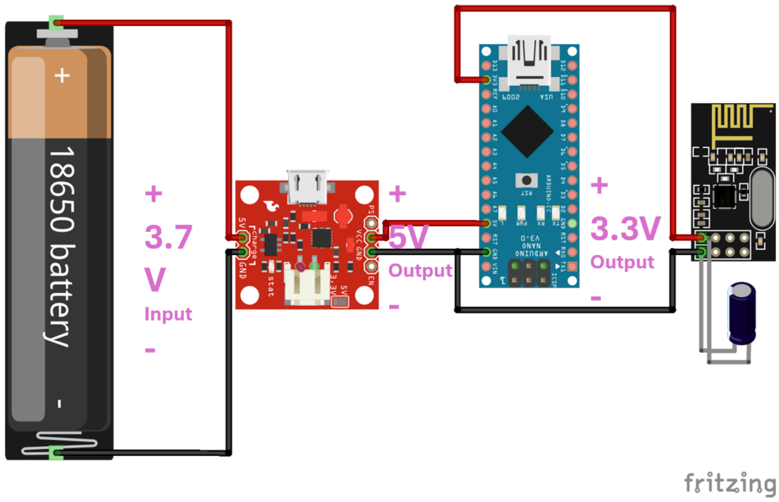


Figure 7: Power Supply Setup

In our **TrekAlert P2P Mesh Communication**, we use a charger/booster module to drive the microcontroller and the nRF24L01+ module. Power is supplied by a single **18650 3.7 V** Li-ion battery, which is connected into the charger module. The charger module then provides a regulated **5 V output** that is fed into the **5V** and **GND** pins of the **Arduino Nano**.

Meanwhile, the **nRF24L01+** module takes its **3.3 V** input from the **3V3** output pin on the Arduino Nano. An additional **decoupling capacitor** is added across the positive and negative terminal to act as a filter, filtering the noise from entering as input.

The **charger module** also provides a **rechargeable solution**, allowing **battery** to be charged without removing it from the **enclosure**. In **off-grid situations** such as **jungle-trekking**, this feature **extends the usage time** of the device when paired with a **power bank**, avoid unnecessary **battery swaps**.

5.4 I/O Device

5.4.1 Push Button Configuration (Microcontroller Input)

In this project, **tactile push buttons** are used as the inputs for the user to control the input signals. To ensure a logic level stability when the button is left unpressed, **pull-up or pull-down** resistors are needed to **avoid** the **floating input**.

- **Pull-up resistor:** Connects the **input GPIO pin** to $\mathbf{V_{cc}}$ through the resistor. **Default** state is pulled to **HIGH**, **pressing** the **button** pulls it **LOW**.
- **Pull-down resistor:** Connects the **input GPIO pin** to **Ground** through the resistor. **Default** state is pulled **LOW**, pressing the **button** pulls it **HIGH**.

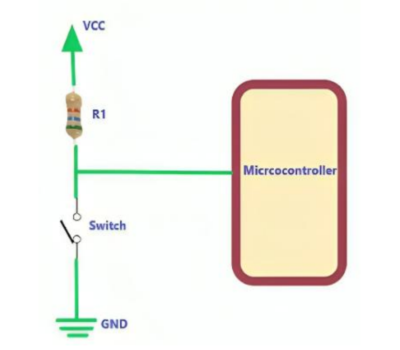
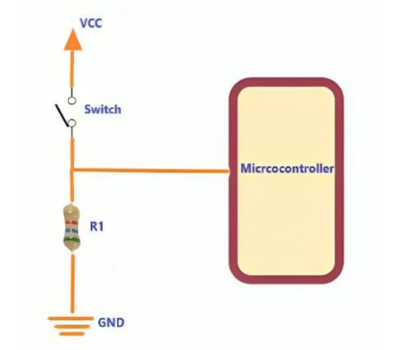
Pull-up resistor	Pull-down resistor
	

Figure 8: Pull-up and pull-down resistor configurations. Adapted from [18].

Additionally, the **resistance** value of $\mathbf{R_1}$ is crucial for ensuring a **stable** pull-up or pull-down configuration:

- If R_1 is **too small**, a large current will flow when the switch is closed, resulting an unnecessary power loses. This is known as using a **strong pull-up (or pull-down)** and less power efficient.

- If R_1 is **too large**, the input may fail to reach a valid logic level when the switch is open due to leakage currents at the microcontroller's pin. This is referred to as using a **weak pull-up (or pull-down)**.

In **TrekAlert,** an **INTERNAL PULLUP configuration** is used to ensure a **HIGH** logic level by **default** when button is **not pressed**. This design results in an **active-low** configuration, where **pressing** a **button** applies a **LOW** signal to the microcontroller's input pin. **Internal pullup** applied the pull-up resistor concept but **without the need of adding an external resistor**, as built-in microcontroller's internal pull-up resistor is used.

Configuration wise, common recommendation for external pull-up or pull-down configuration is to select **a resistor** that is at least **10 times smaller** than the **impedance** of the **input pin**.

By understanding the principles of circuit behaviour in the circuit, students can understand the voltage divider behaviour. Besides, students can explore logic design and the importance of state management in embedded systems.

5.4.2 LED Configuration (Microcontroller Output)

Light-emitting diodes (LEDs) are used in this educational kit as visual indicators for the three statuses, "**SAFE**, **WARNING**, **SOS**". For a complete tutorial towards LEDs operation, this educational includes the explanation for both active-high and active-low configuration for LEDs. However, the final prototype for this educational kit is equipped with active-low configuration only.

- **Active-high:** The **LED** turns **ON** when the LED's pin **receives** a **HIGH** logic level output by **control pin**

- **Active-low:** The **LED** turns **ON** when the LED's pin **receives** a **LOW** logic level output by **control pin**

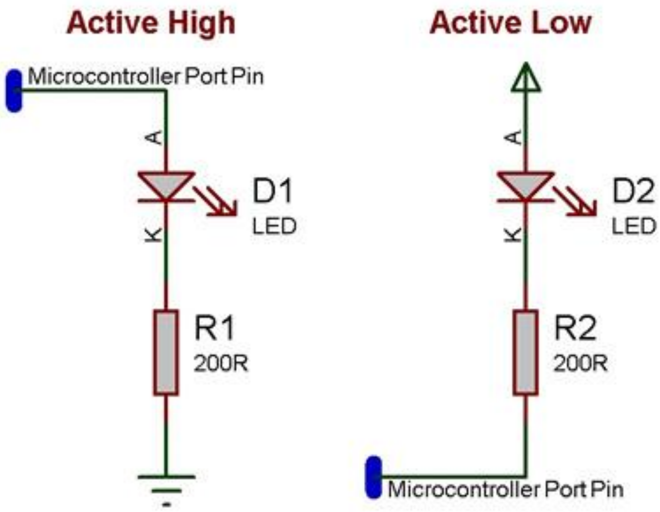


Figure 9: LEDs' configuration types. Adapted from [19]

TrekAlert uses an **active-low design**, meaning that when the microcontroller **wants to turn** the LED **on**, it requires GPIO pin to **output** a **LOW** signal. In this setup, the **cathode of the LED** is connected to the microcontroller's digital output pin through a **current-limiting resistor**, while the **anode** is connected to a steady **5 V supply**.

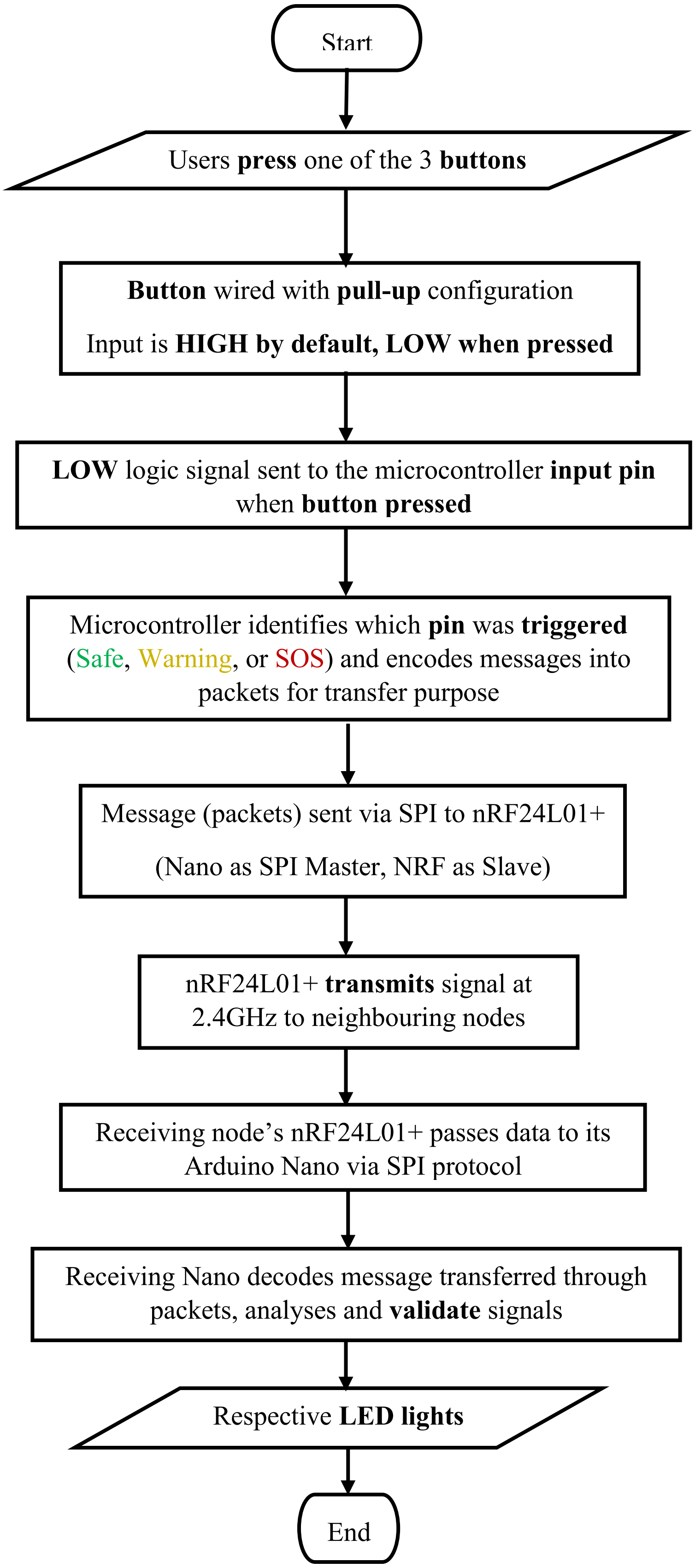
When the microcontroller sets its output pin to **LOW**, it creates a **potential difference**. This allows current to flow from the 5 V source, through the LED and resistor and into the microcontroller pin. The resulting **current flow** causes the LED to light up.

In this configuration, the microcontroller is **sinking current**, which is often more stable and efficient than sourcing current. Thus, the **LED illuminates** whenever the microcontroller GPIO pin output **LOW** signal.

The dual-configuration designs of the LED interface enable students to learn:

- How **logic levels affect circuit behaviour**
- Understand **mode of operation** of components **only defined by hardware**
- Reinforce understanding of **potential difference**, **current flow**

Signal Flowchart: End-to-End Operation between **TrekAlert Node**



5.5 Designs and Implementations

Here comes the section that outlines all the overall system design, circuitry and physical implementation of the **TrekAlert** project. Several design schematics are provided to illustrate the architecture, interconnections between the components for a final assembly.

5.5.1 Block Diagram

Presents a top-level overview of the system, showing the project initiation starting from constructing **power supply flow**, **core components** (Arduino Nano, charger module, NRF24L01+) and **input/output devices** (buttons and LEDs), without the detailed wiring schematics.

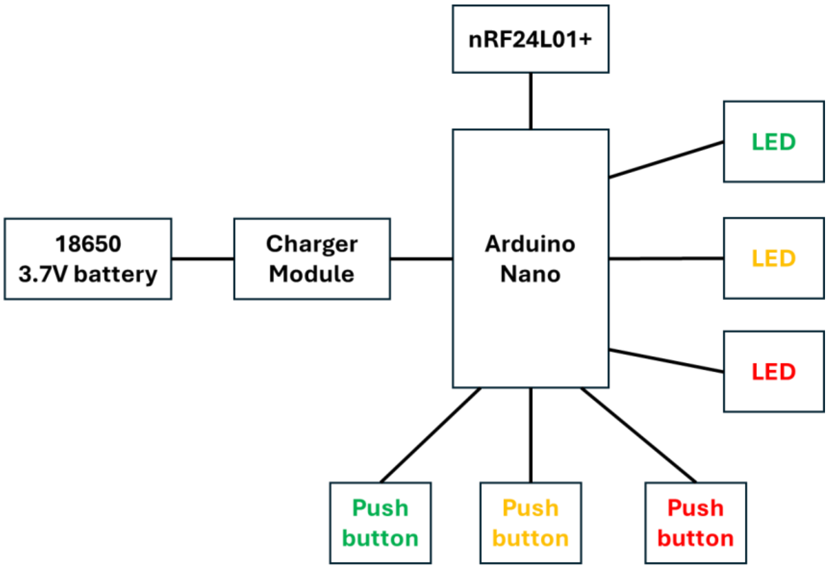


Figure 10: Block Diagram

5.5.2 Fritzing Schematic

Presents a circuit-level wiring of all components on a breadboard, as well as detailed wiring schematics, upgraded from block diagram.

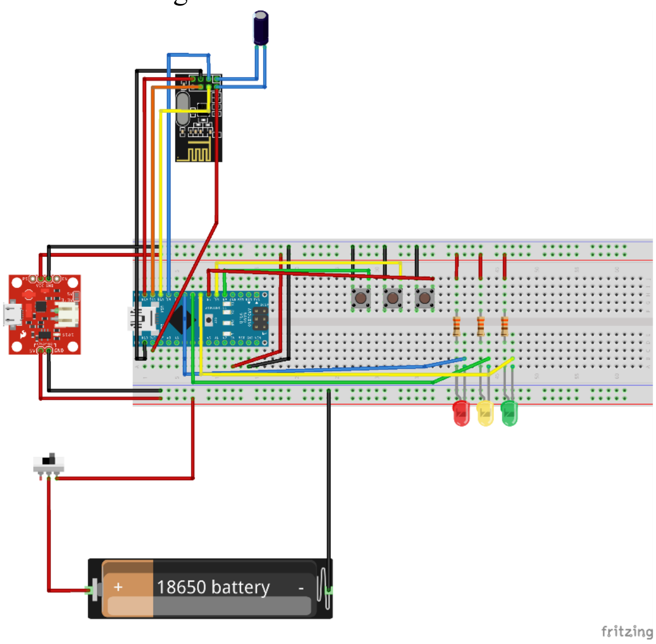


Figure 11: Breadboard Schematic

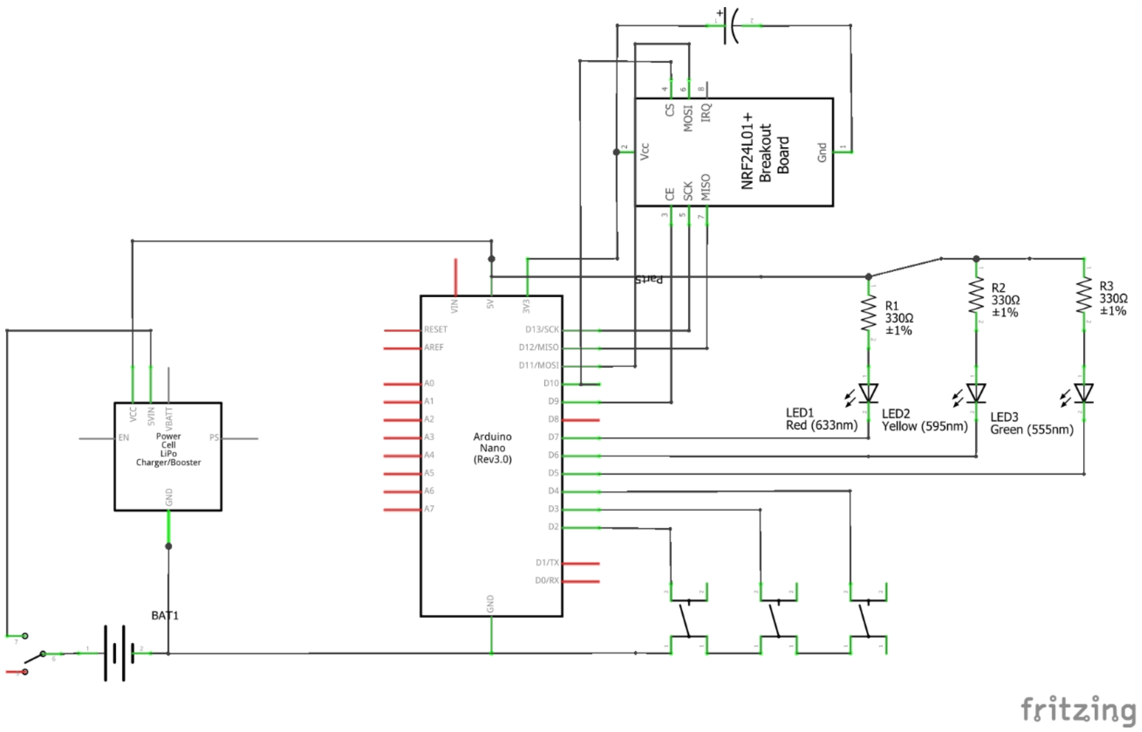


Figure 12: Wiring Schematic

5.5.3 KiCad PCB Design and 3D Model Rendering

This part illustrates the **physical layout** and **component placement** on a custom PCB. The model enables a preview of potential **manufacturing considerations** for publishing the design as a **commercial product** and distributing it worldwide as a **student learning kit**.

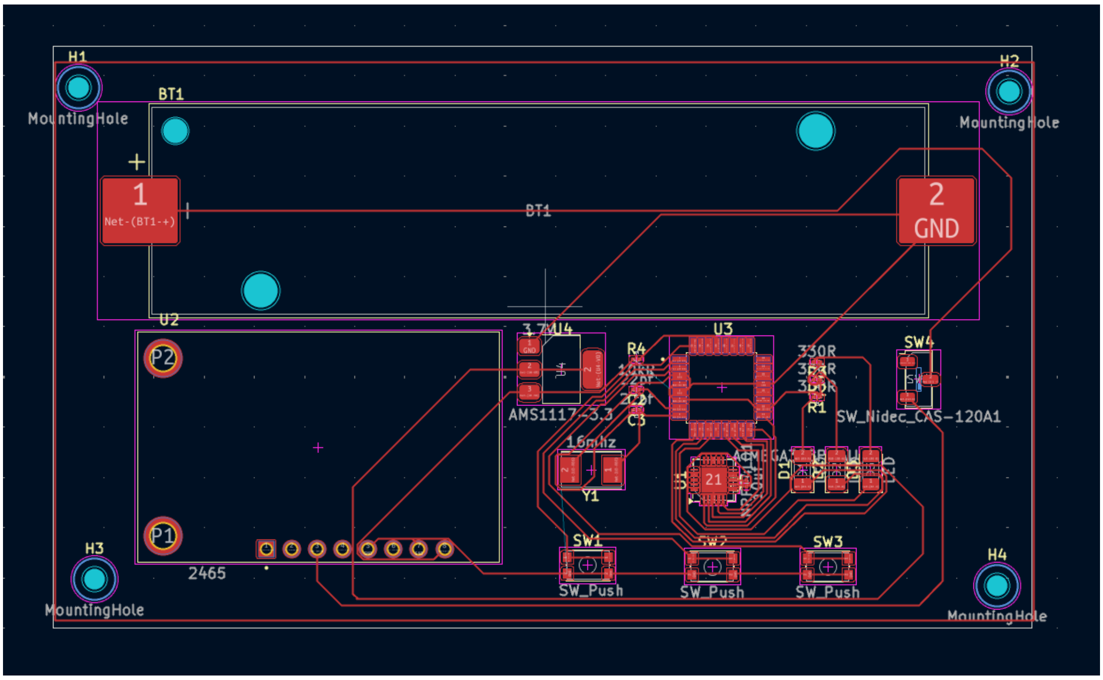


Figure 13: PCB Physical Layout

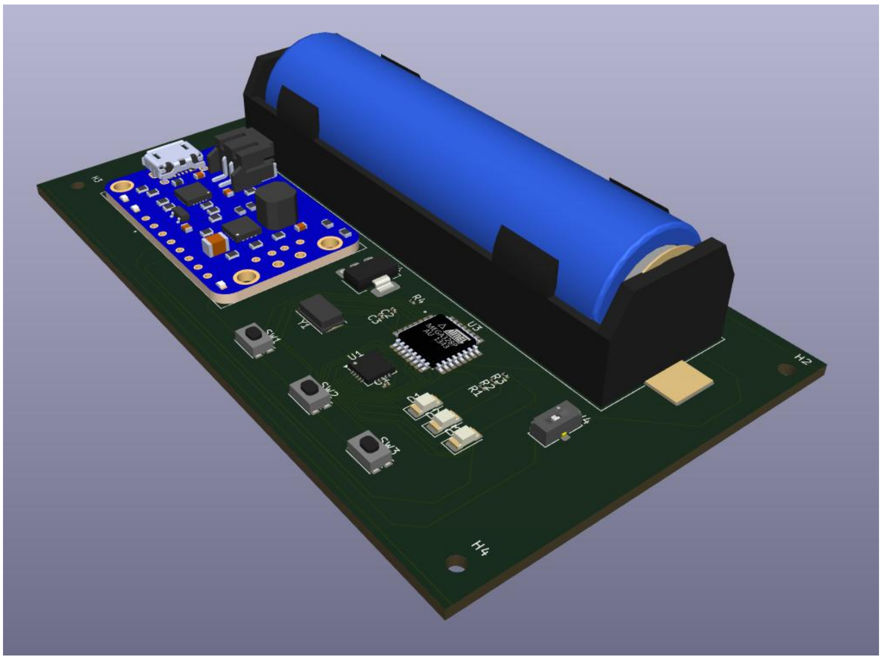
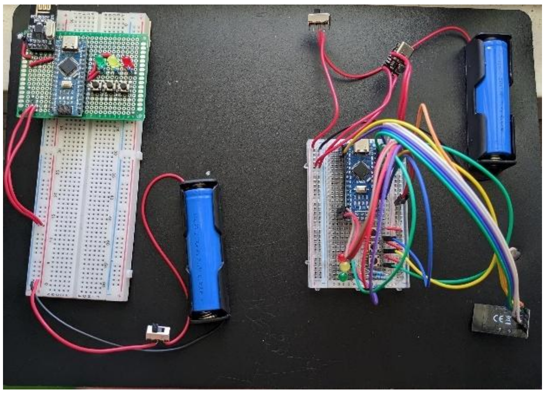


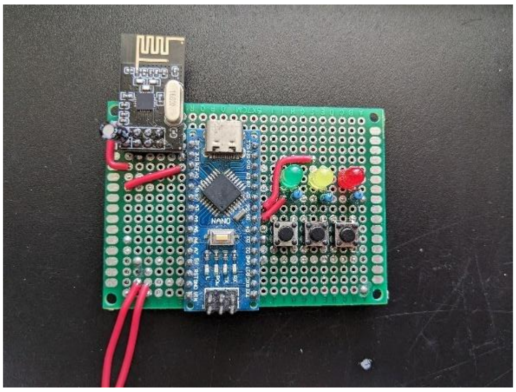
Figure 14: 3D PCB Model Rendering

5.5.4 Final Prototype

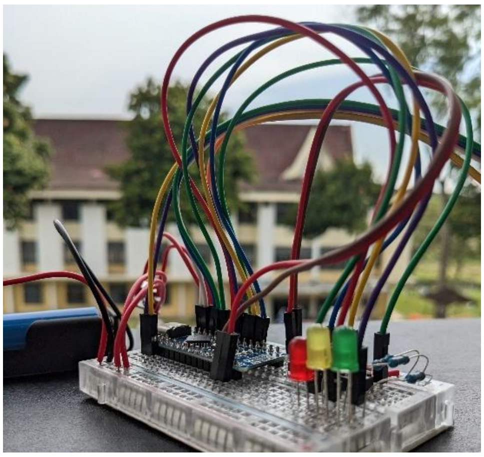
Here compiles all the prototype product, one detachable version on breadboard and one fixed piece on perfboard, with soldered traces



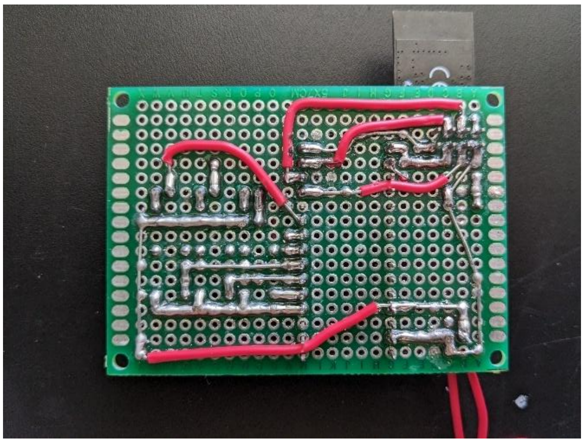
Two isolated nodes



Perfboard model



Breadboard detachable model



Solder traces

6.0 Software Description, Designs and Implementations

Wireless communication of **TrekAlert** using nRF24L01+ transceiver is programmed through Arduino Nano sketches, a firmware. It configures the microcontroller to read the state of three **push buttons**, transceive **packets** between nodes and control **LEDs state**.

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

#define BTN_GREEN 2
#define BTN_YELLOW 3
#define BTN_RED 4

#define LED_GREEN 5
#define LED_YELLOW 6
#define LED_RED 7

RF24 radio(9, 10); // CE, CSN

// Update addresses for each node
const byte thisNode[6] = "NODE2";
const byte otherNode[6] = "NODE1";

// Store button and LED states
bool buttonStates[3] = {0, 0, 0};
bool lastButtonStates[3] = {0, 0, 0};
bool ledStates[3] = {0, 0, 0};

void setup() {
  Serial.begin(9600);

  pinMode(BTN_GREEN, INPUT_PULLUP);
  pinMode(BTN_YELLOW, INPUT_PULLUP);
  pinMode(BTN_RED, INPUT_PULLUP);

  pinMode(LED_GREEN, OUTPUT);
  pinMode(LED_YELLOW, OUTPUT);
  pinMode(LED_RED, OUTPUT);

  digitalWrite(LED_GREEN, HIGH);
  digitalWrite(LED_YELLOW, HIGH);
  digitalWrite(LED_RED, HIGH);

  radio.begin();
  radio.openWritingPipe(otherNode);  // Address to send to
  radio.openReadingPipe(1, thisNode); // Address to receive on
  radio.setPALevel(RF24_PA_LOW);
  radio.stopListening();  // Start in transmit mode
}
```

Included libraries

Button input setup

LEDs output setup

RF communication setup

Define device's address and destination address

Define mode of operation for components

```
void loop() {
  // READ BUTTONS
  buttonStates[0] = !digitalRead(BTN_GREEN);  // Active LOW
  buttonStates[1] = !digitalRead(BTN_YELLOW);
  buttonStates[2] = !digitalRead(BTN_RED);

  // IF BUTTON CHANGED → Send
  bool sendNeeded = false;
  for (int i = 0; i < 3; i++) {
    if (buttonStates[i] != lastButtonStates[i]) {
      sendNeeded = true;
      lastButtonStates[i] = buttonStates[i];
    }
  }

  if (sendNeeded) {
    radio.stopListening();  // Switch to TX mode
    radio.write(&buttonStates, sizeof(buttonStates));
  }

  // RECEIVE if data available
  radio.startListening();
  unsigned long startTime = millis();
  while (!radio.available()) {
    if (millis() - startTime > 5) break;  // Non-blocking wait
  }

  if (radio.available()) {
    radio.read(&ledStates, sizeof(ledStates));
    digitalWrite(LED_GREEN, !ledStates[0]);
    digitalWrite(LED_YELLOW, !ledStates[1]);
    digitalWrite(LED_RED, !ledStates[2]);
  }

  delay(10);  // Small delay to avoid flooding
}
```

Main **loop()** logic:

1. Reads the **buttons** (active-low) and updates the **buttonStates**
2. Compares current state vs. last state — if any button changed, prepares data to send
3. Uses **radio.stopListening()**, change to **transmit** mode, then **radio.write()** to send updated **buttonStates**
4. Switches back to receive mode (**radio.startListening()**), waits for data for up to 5 ms
5. If data is received (**radio.available()**), updates the LED states

This mechanism flows indicates a construction of transceiver mode for each node. It continuously performs checking and switching between transmitter and receiver modes to ensure the transceiver mode go smoothly.

6.1 Feature Highlights

- **Edge-triggered Transmission** — Only sends **new data** when a button **state changes**
- **Active-low Input Logic** — Uses **pull-ups** so **unpressed buttons** read **HIGH**
- **Wireless Duplex** — Operates as both **transmitter and receiver** on the **same device** with a 5ms checking intervals
- **Power Efficiency** — **Low-power mode** for the nRF24L01+ (setPALevel(RF24_PA_LOW))

This **firmware** integrates hardware **inputs (buttons)** and **outputs (LEDs) with radio communication**. Each node continuously **monitors its own inputs and broadcasts any changes**. It also listens for incoming packets to control its LEDs according to the states sent by its peer.

7.0 Results and Discussion

We packaged this **P2P Mesh Communication** system as a **discrete components kit** for educational purposes, allowing students to **build their own communication device from the ground up**.

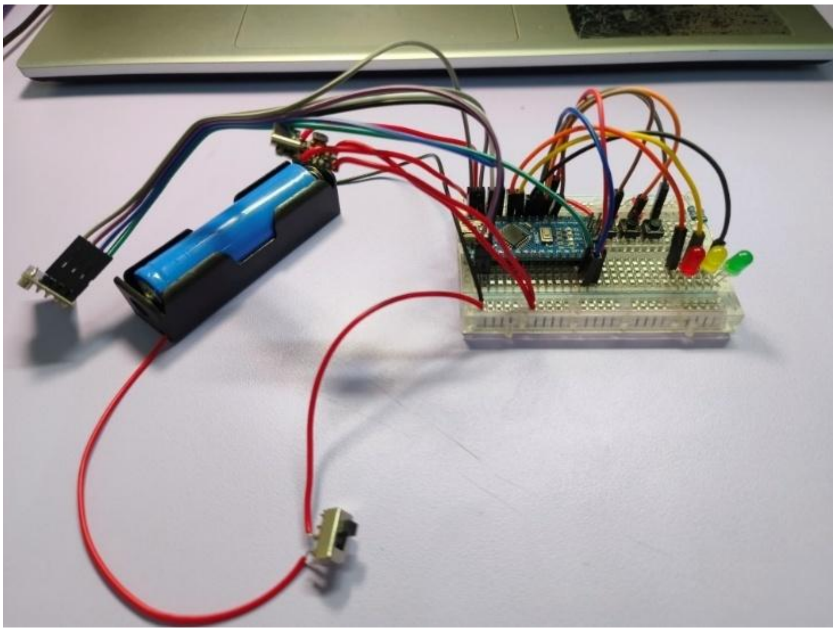
To support this hands-on learning experience, we have published a **detailed instruction manual** that includes **step-by-step assembly guidelines** and **demonstrations** of its operation.
The demonstration verified that:

- Transceiving process successfully carried out between nodes
- LEDs light up as expected
- System works at certain range

When the user receives this **educational kit**, there will be 2 modules, one is the perfboard module and the another is a breadboard module, for educational purpose, user will be required to **assemble the breadboard module themselves**.

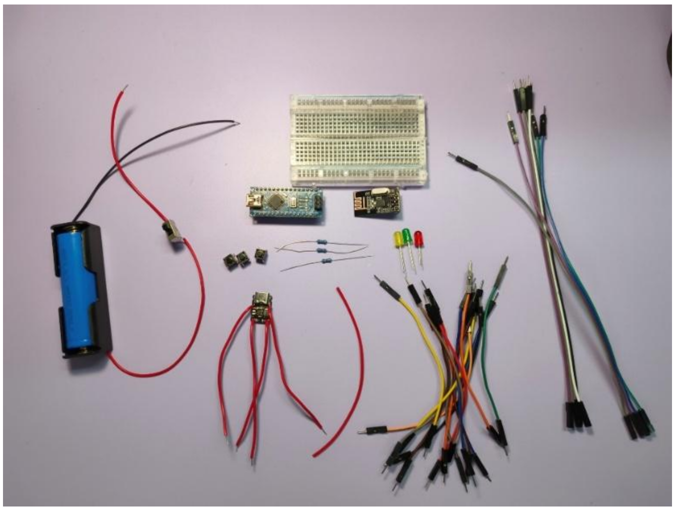
Throughout the **instructions manual** below, students will be able to **comprehend the basic mechanism** of the device as per **discussed**

7.1 Assembly Instructions



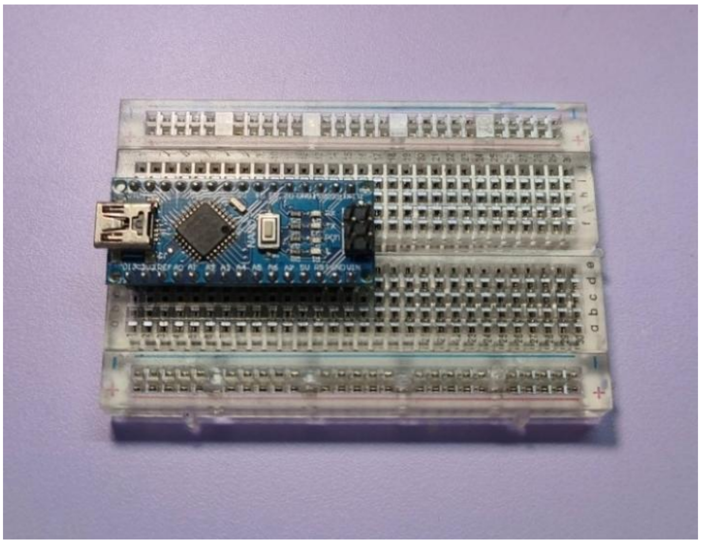
This **Arduino breadboard circuit** is a **jungle trekking peer-to-peer mesh communication system**. By using an **nRF24L01+** module, it allows communication between two **jungle trekking peer-to-peer devices**. Both devices are allowed to send and receive data and trigger the **LED** on the opposite device, giving simple **warnings** or **alerts** to your partner during jungle trekking. It is a great **beginner project** for students to assemble the device from scratch and learn **Arduino** with the following components.

Components required

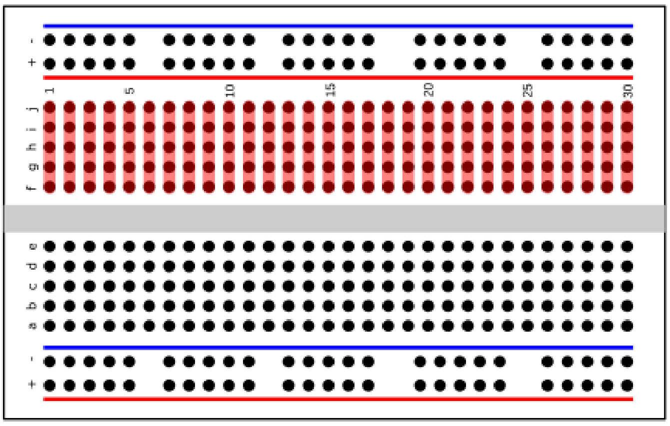


Components	Quantity
Breadboard	1
Arduino nano board	1
nRF24L01+	1
Charger module	1
330Ω resistor	3
Red LED	1
Yellow LED	1
Green LED	1
Push button	3
Slide switch	1
3.7V battery	1
Battery holder	1
Jumper wires (male to male)	8
Jumper wires (male to female)	7

Step 1: Arduino nano

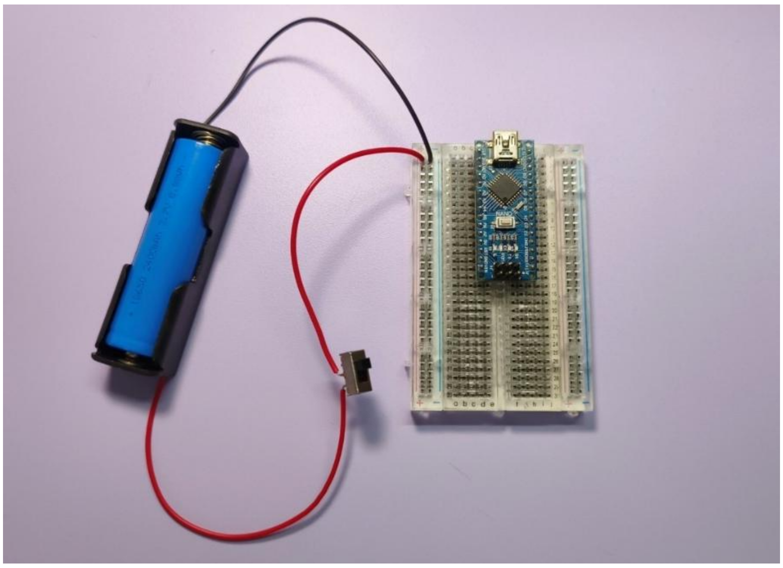


- Place the **Arduino nano** at the breadboard as show in figure above



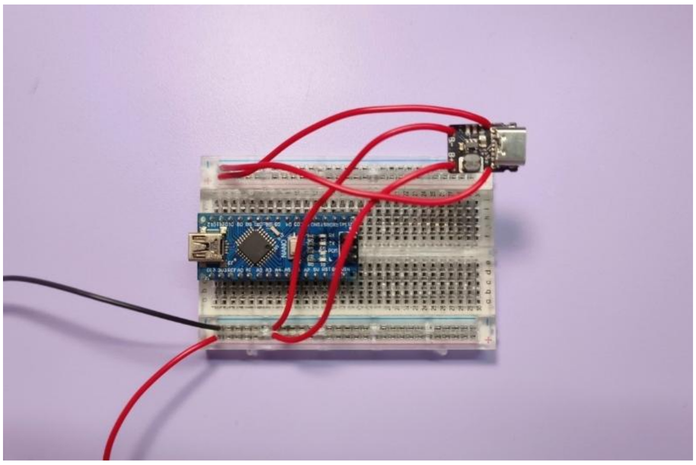
- The figure above shows the connection in a breadboard
- In the main area, there are **5 holes in each vertical row** that are electrically connected. (e.g. a,b,c,d,e holes are connected vertically)
- **Horizontal connection** in the power rails to connect power and ground across the length of the power rail
- The **central gap** to keeps the two halves of the main area separated

Step 2: Battery



- Connect the **battery** to the breadboard as shown in figure above to match with the sign on the breadboard
- Connect the **red wire (positive terminal)** of the battery holder to breadboard power rail marked '+'
- Connect the **black wire (negative terminal)** of the battery holder to breadboard power rail marked '-'

Step 3: Connecting to charger module

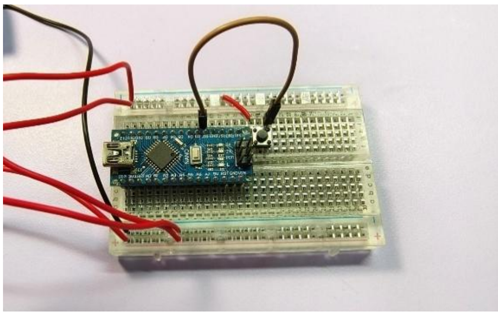


- The **charger module** functions as a **boost converter**, by stepping up **3.7V** from the battery to **5.0V**, to powering the **Arduino nano** since it requires 5V input voltage.
- It is also used to **charge up the battery** with **type-c port**
- Connect the **Charger module** to the breadboard as above

- Connect the **B+ pin** of the charger module to the left + **power rail** (where the positive terminal of the battery holder is connected)
- Connect the **B- pin** of the charger module to the left **- power rail** (where the negative terminal of the battery holder is connected).
-

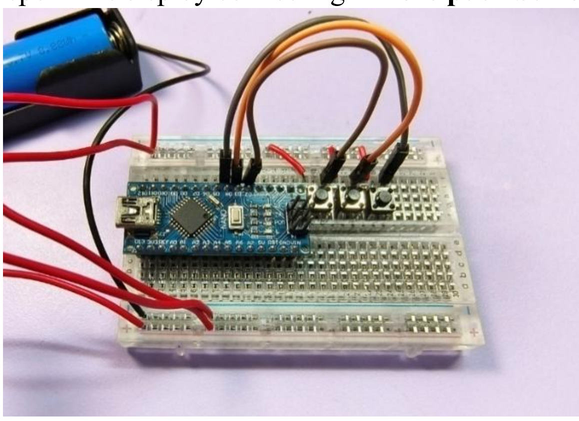
- Connect the + **output pin** of the charger module to the right + **power rail**."
- Connect the **- output pin** of the charger module to the right **- power rail**

Step 4: Connect the push button



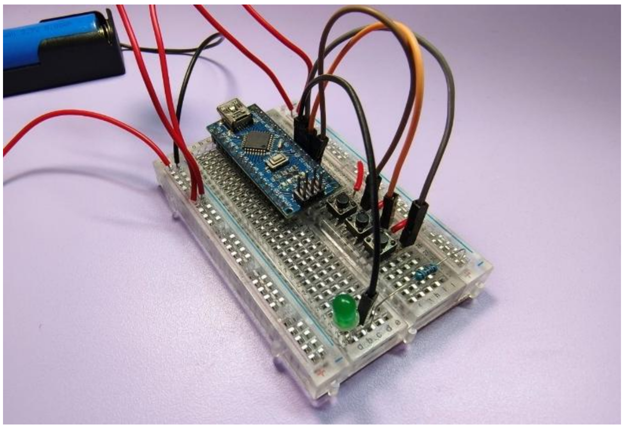
- The configuration of **push button** used is **active low** configuration, meaning the input pin will read low when the button is pressed.
- This setup assume **that Arduino nano internal pull-up resistor** is used on the input pin.
- Place one **push button** onto the breadboard, as shown in the figure above
- Connect one terminal of push button to **Arduino nano Pin D2**
- Connect the other terminal of push button to **power rail GND**

Repeat this step by connecting **2 more push button**

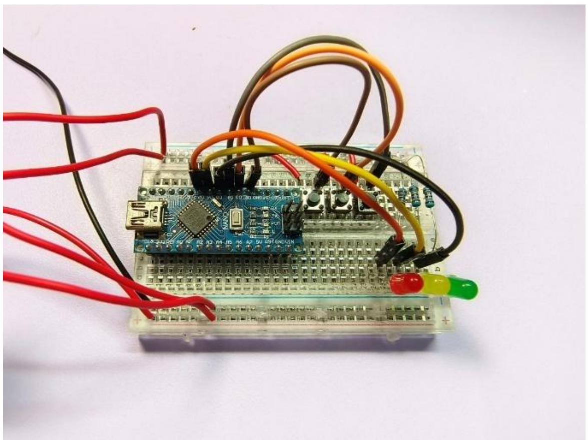


- Place the push buttons as the setup shown above
- Connecting one terminal of the push button to input pins **(D3 and D4)** and the other end to **GND rail**

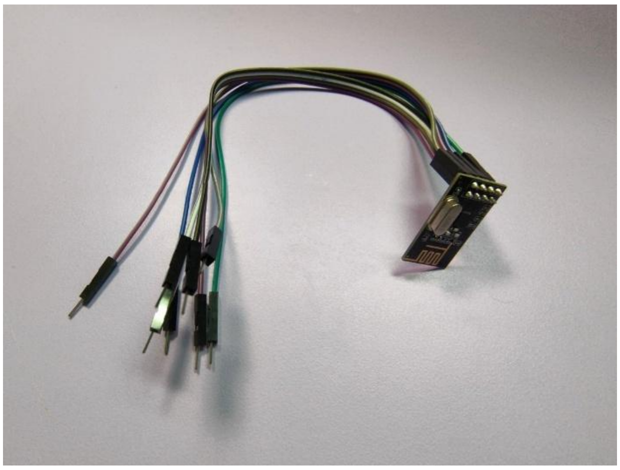
Step 5: Connecting the LEDs



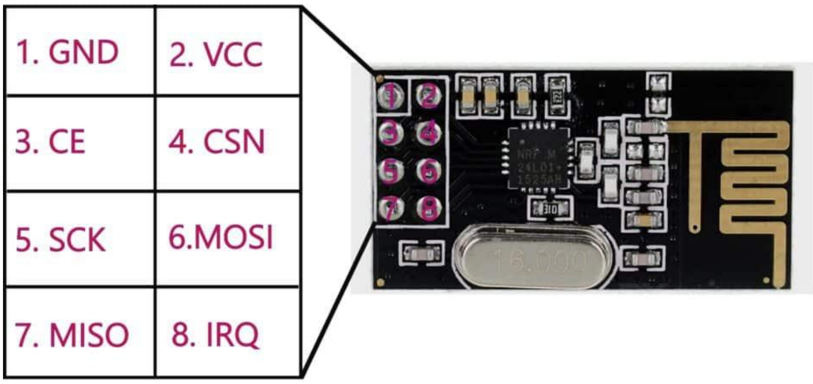
- The **LED** is connected in **active low configuration**
- **LED will illuminate** when its corresponding GPIO pin (**D5,D6,D7**) is set to **low (0V)**
- **NOTE**: The **longer leg** of an LED is the **anode(+),** while the **shorter leg** of an LED is **cathode (-)**
- Connect one end of the **330Ω resistor** to '+' **5V power rail**
- Connect the other end of **330Ω resistor** to **anode (positive leg)** of the LED
- Connect the **cathode (negative leg)** of LED to the **GPIO pin (D7,D6,D5)**
- Repeat these steps for the remaining two LEDs (yellow and green), connecting each to a separate GPIO **pin (Red to D7, Yellow to D6, Green to D5)** as illustrated in the diagram below



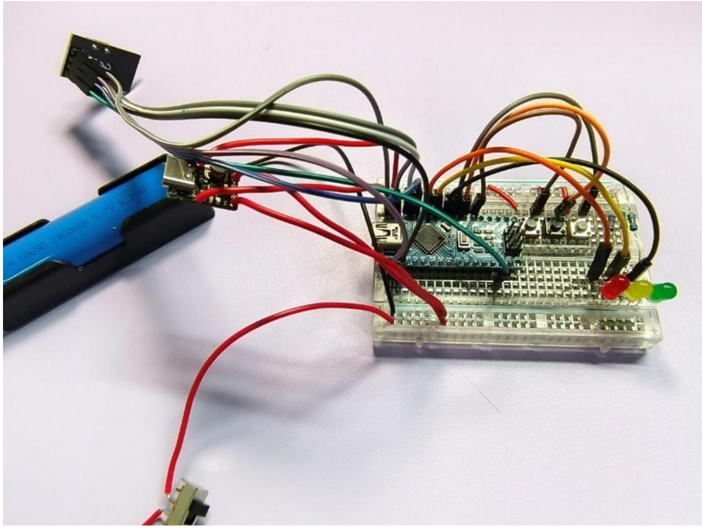
Step 6: nRF24L01+ module



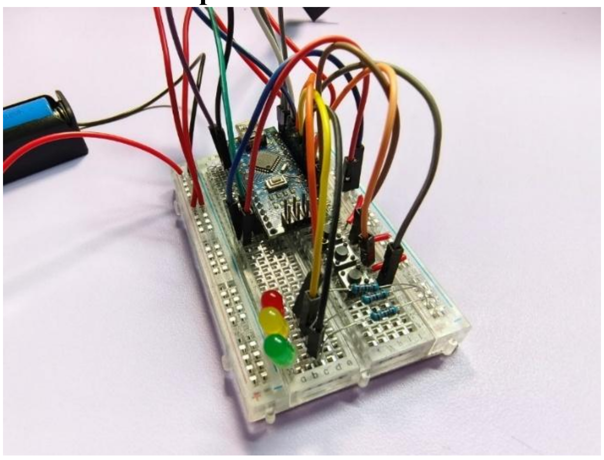
- **Use male to female jumper wire** to connect the **nRF module** to Arduino
- The pinout diagram of nRF module is as below



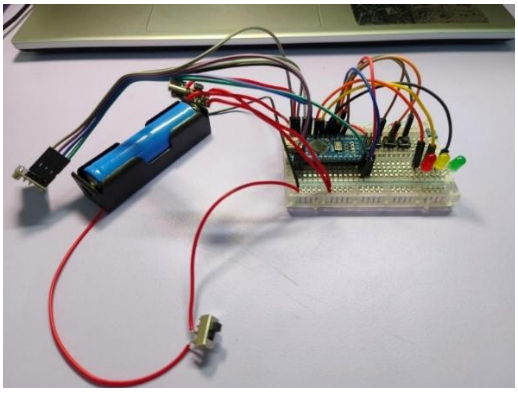
nRF24L01+ pin	Arduino nano pin
GND	Power rail GND
VCC	Power rail 3.3V (Do not connect to 5V)
CE	D9
CSN	D10
SCK	D13
MOSI	D11
MISO	D12
IRQ	Not used



Step 7: VCC and GND



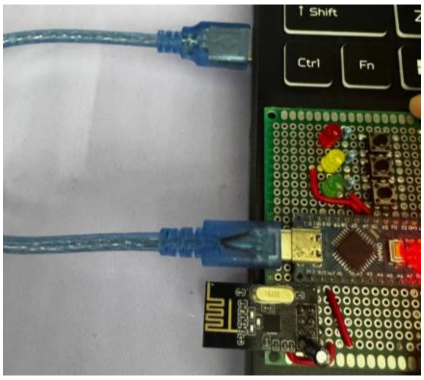
- Supply power to the **Arduino nano** from the breadboard regulated **5V and GND power rail**
- Connect the **Vin pin** on the Arduino nano to the **'+' (5V) power rail**.
- Connect the **GND pin** on the Arduino nano to the **'-' (GND) power rail**.



Final product of a single P2P mesh communication node on breadboard

7.2 Firmware Installation Instructions

Step 1: Connect Arduino and PC using cable



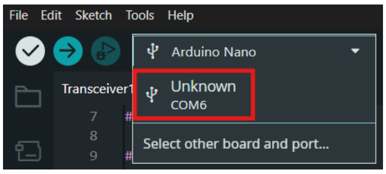
Insert the cable type C side to Arduino and USB A side to PC

Step 2: Configure Arduino Firmware Through Sketches

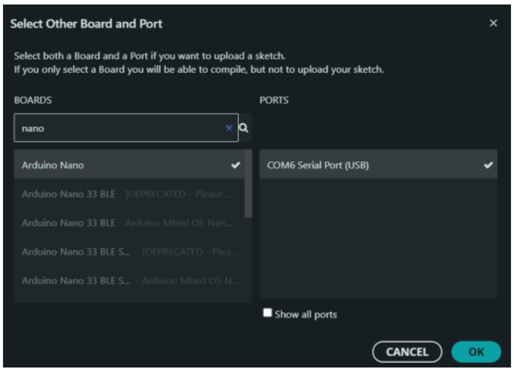
1) Import Code

This project is published to the open-source community on GitHub,
hence students are only required to obtain the firmware from the link below
https://github.com/gr1d-init/P2P_Mesh_Communication/tree/main/Arduino
follow the to install the source code, unzip the file and you will get source code file named transceiver1.ino, open it using Arduino IDE

2) Connect Arduino to PC and Initializing



Connect Arduino to your PC using the cable provided, at the top right select the USB port
NOTE : If occur more than 1 unknown usb devices, make sure to remove other USB devices on your computer and indicate the USB port name of your Arduino (In this case Arduino are connected to COM6 port)



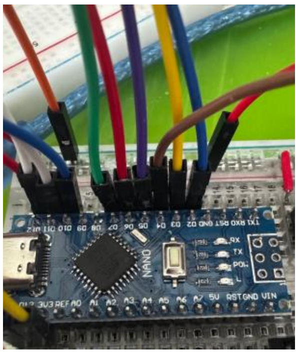
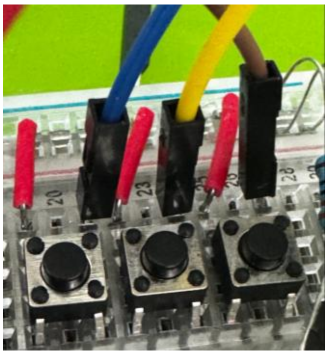
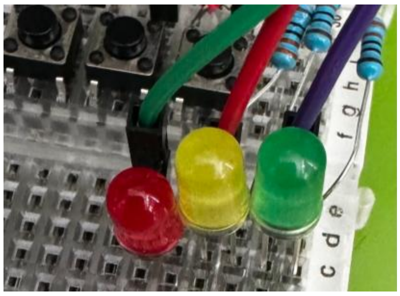
Now, select Arduino Nano as the board and select the proper port. Click OK to save.

Step 3: Declaring Coding based on Arduino I/O Pins

```
1  #include <SPI.h>
2  #include <nRF24L01.h>
3  #include <RF24.h>
4
5  #define BTN_GREEN 4
6  #define BTN_YELLOW 3
7  #define BTN_RED 2
8
9  #define LED_GREEN 5
10 #define LED_YELLOW 6
11 #define LED_RED 7
12
13 RF24 radio(9, 10); // CE, CSN
```

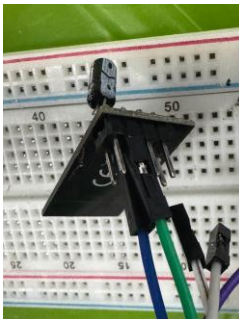
Enter pin number according to the colour of LED and button assembled

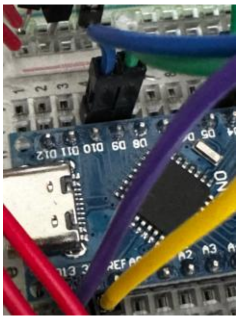
Now we are focusing on the coding.
From line 5 to line 11, declare the Arduino pin number according to the button and LED assembled.

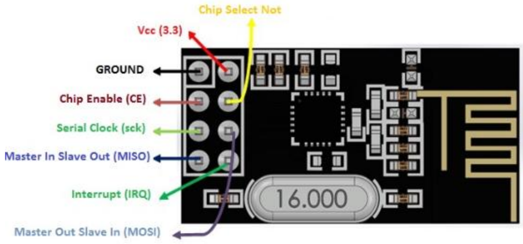




In this case :
D2 → Blue Jumper → BTN_RED
D3 → Yellow Jumper → BTN_YELLOW
D4 → Brown Jumper → BTN_GREEN
D5 → Purple Jumper → LED_GREEN
D6 → Red Jumper → LED_YELLOW
D7 → Green Jumper → LED_RED

From line 13, declare the Arduino pins according to NRF module.

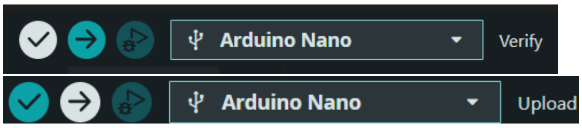






In this case :
D9 → Green Jumper → CE
D10 → Blue Jumper → CSN

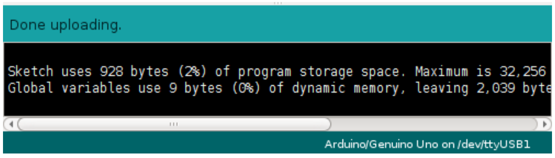
Step 4: Verify and Upload to Arduino



After done all the declaration, what we need is to burn the firmware into Arduino. The first step is clicking the √ button to verify, waiting IDE to done compiling. Next, click → button to upload the firmware.

Please wait patiently and DO NOT disconnect the Arduino from your PC while uploading.

At the end, you are successfully burned in the firmware once the Done Uploading message pop up.

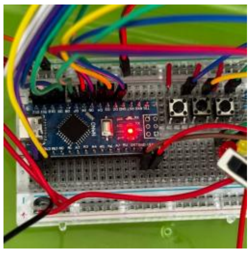


7.3 Operating Instructions

Step 1: Turn on Power Supply

1 Push the button to ON

2 The power indicator light will turn on indicate Arduino are in working mode

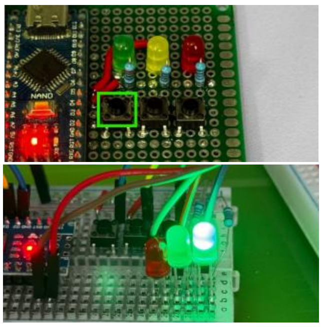


3 The blue light will turn on indicate power booster module are in working mode

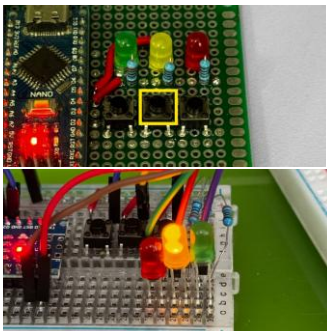
Step 2: Button Mode

Situation : When necessary to communicate with another teammate, push the button according to colour assigned

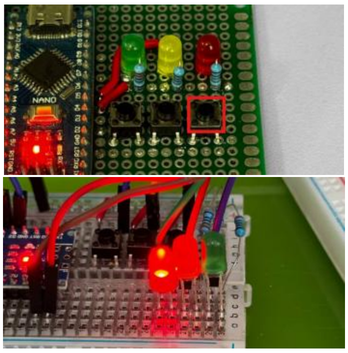
1 Green button on module 1 pressed, green light on module 2 light up



2 Yellow button on module 1 pressed, yellow light on module 2 light up



3 Red button on module 1 pressed, red light on module 2 light up

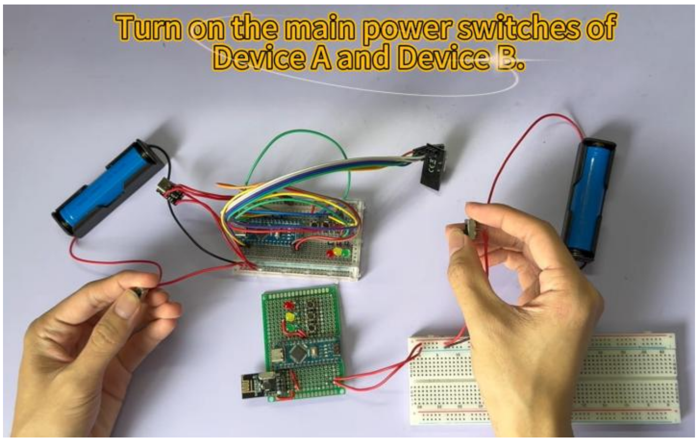


7.4 Result Showcase

In our demonstration video, we carried out some testing and verification:

- Tested switching in **active-low** with **internal pull-up** configuration
- Verified LED logic (in **active-low** configuration)
- Verified **bidirectional** communication between nodes

Follow the link to see the demonstration video: *https://youtu.be/UirkyzqIXWE*



8.0 Future works

Challenges

- The current setup **only demonstrated direct communication between two nodes**
- The **repeater feature of the peer-to-peer mesh** (Node 3 as a relay) was **not fully utilised** in this version
- Given the short range of the nRF24L01+ modules, the system may face **limited operational distance** in larger or more complex terrains
- The demonstration focused only on using **LEDs** to show **communication status**. However, the hardware can support much more, it is capable of **any signal processing task**, unlike traditional **walkie-talkies** that only transmit **audio**
- Security issue due to **absence of encryption** among the intermediate node's communication

Improvements

- **Implement a third node** to fully demonstrate and test the **repeater functionality** for extending the network range
- For larger-scale or long-distance communication, consider using **alternative radio technologies** such as **LoRa** to achieve better range and scalability
- Expand the **user interface** by integrating a **graphical OLED panel** or even a **computer device**, allowing the system to display **messages** and other data beyond simple **LEDs**, implement a feature-rich device that support SMS, document transfers
- Develop a **waterproof enclosure** for the node to ensure it can operate reliably in harsh outdoor conditions, especially in rainy or humid environments
- Implement **AES-256 encryption** for transmitted signals to enhance privacy and security, ensuring that only authorised devices can read or process the messages, prevent intermediate nodes from **intercepting** the message

9.0 Conclusion

Our project focused on two primary objectives:

- **Exposing students to peer-to-peer (P2P) mesh communication**
- **Providing hands-on experience in learning to program a microcontroller under their RBT, Computer Science and Physics syllabus**

With the aid of our **learning kit**, students can gain foundational knowledges in **assembling hardware**, **developing firmware** and **implementing wireless communication** between devices. This hands-on approach equips them with a deeper understanding of **embedded systems**, encourages **problem-solving skills**, synchronise with their Computer Science and Physics syllabus and benefits for further exploration in the fields of **electronics** and **communication engineering** as well.

As time evolve, **P2P Decentralised Mesh Communication Technology** with **RF wireless communication protocol** has become a **rising technology** that increasingly come into our sight. It remains a **great potential** in the **domestic sectors**, which currently only being implemented among **professional** and **industry usages**, such as **military** and **agriculture**. Thus, a **domestic market** of among this technology is still waited to be explored and excavated. For more details, see [20], [21], [22], [23], [24].

In short, we hope more students get exposed to this technology and head the grounds up, build and construct a **P2P Mesh Communication node** on their own with provided **manuals**. As **STEM workers** are highly in demand these days, we sincerely hope students who get accessed to this kit getting **inspired** and start contributing their **efforts** and **research** on this **P2P Mesh Communication Topic** to enhance **networking technology**.

Last but not least, we are driven to deliver the very best in our **P2P Mesh Communication Learning Kit** to capture the true **essence** of an **educational kit**.

10.0 References

[1] goTenna. "goTenna and Everywhere Communications Form Strategic Partnership Transforming Enhanced Connectivity," goTenna Newsroom. Accessed: May 28, 2025. [Online]. Available: https://gotenna.com/blogs/newsroom/gotenna-and-everywhere-communications-form-strategic-partnership-transforming-enhanced-connectivity.

[2] LoRa Alliance. "LoRaWAN for Smart Agriculture," LoRa Alliance Video. (Aug 11, 2023). Accessed: May 29, 2025. [Online Video]. Available: https://youtu.be/l2m9ZJIEZ0g?si=08_inty-NZJ2JIVr.

[3] LoRa Alliance. "LoRaWAN® for Smart Industry," LoRa Alliance Industry Vertical Market. Accessed: May 29, 2025. [Online]. Available: https://lora-alliance.org/industry-vertical-market/.

[4] Zenarmor. "Mesh Topology: Definition, Practices and Importance," Zenarmor Docs. Accessed: Jun. 1, 2025. [Online]. Available: https://www.zenarmor.com/docs/network-basics/what-is-mesh-topology.

[5] BBC Bitesize. "Computer Networks and Topologies – Mesh networks," BBC Bitesize. Accessed: Jun. 1, 2025. [Online]. Available: https://www.bbc.co.uk/bitesize/guides/z7mxh39/revision/6.

[6] YouTube. "What is Zigbee and How it Works | Zigbee Network Explained," (May 22, 2021). Accessed: Jun. 2, 2025. [Online Video]. Available: https://youtu.be/pFDkibWy0yg?si=w_VuCWbKM7w4yXwP.

[7] CDEBYTE. "News," CDEBYTE. Accessed: Jun. 2, 2025. [Online]. Available: https://www.cdebyte.com/news/587.
[8] Wikipedia. "Bluetooth Mesh Networking," Wikipedia. Accessed: Jun. 3, 2025. [Online]. Available: https://en.wikipedia.org/wiki/Bluetooth_mesh_networking.

[9] HowToMechatronics. "Arduino Wireless Communication with NRF24L01 Tutorial," HowToMechatronics. Accessed: Jun. 5, 2025. [Online]. Available: https://howtomechatronics.com/tutorials/arduino/arduino-wireless-communication-nrf24l01-tutorial/.

[10] ResearchGate. "Multi-hop cellular network and P2P mobile mesh network," ResearchGate. Accessed: Jun. 7, 2025. [Online]. Available: https://www.researchgate.net/figure/Multi-hop-cellular-network-and-P2P-mobile-mesh-network_fig1_46476397.

[11] GeeksforGeeks. "Introduction of Mobile Ad-hoc Network (MANET)," GeeksforGeeks. Accessed: Jun. 7, 2025. [Online]. Available: https://www.geeksforgeeks.org/computer-networks/introduction-of-mobile-ad-hoc-network-manet/.

[12] TechTarget. "Definition of Mesh Network Topology," IoT Agenda. Accessed: Jun. 10, 2025. [Online]. Available: https://www.techtarget.com/iotagenda/definition/mesh-network-topology-mesh-network.

[13] Arduino Store. “Arduino Nano,” Arduino Official Store. Accessed: Jun. 14, 2025. [Online]. Available: https://store.arduino.cc/products/arduino-nano.

[14] NRF24 Library Docs. “RF24 Class Reference,” NRF24 GitHub Pages. Accessed: Jun. 15, 2025. [Online]. Available: https://nrf24.github.io/RF24/classRF24.html.

[15] Cytron Technologies. “Wireless NRF24L01 Plus PA/LNA with Antenna,” Cytron. Accessed: Jun. 15, 2025. [Online]. Available: https://my.cytron.io/p-wireless-nrf24l01-plus-pa-lna-with-antenna.

[16] SparkFun. “nRF24L01+ Product Specification,” SparkFun. Accessed: Jun. 17, 2025. [Online]. Available: https://cdn.sparkfun.com/assets/3/d/8/5/1/nRF24L01P_Product_Specification_1_0.pdf.

[17] Last Minute Engineers. “nRF24L01+ Arduino Wireless Communication,” Last Minute Engineers. Accessed: Jun. 18, 2025. [Online]. Available: https://lastminuteengineers.com/nrf24l01-arduino-wireless-communication/.

[18] Circuits-DIY. “Pull-up and Pull-down Resistor Electronics Tutorial,” Circuits-DIY. Accessed: Jun. 22, 2025. [Online]. Available: https://www.circuits-diy.com/pull-up-and-pull-down-resistor-electronics-tutorial/.

[19] Flowcode. “Component: LED Array Template,” Flowcode Wiki. Accessed: Jun. 22, 2025. [Online]. Available: https://www.flowcode.co.uk/wiki_old/wikiv8/index.php?title=Component:_LED_Array_Template_(Outputs:_LEDs).

[20] Chai, Y., Zeng, X.-J., & Liu, Z. “The future of wireless mesh network in next-generation communication: a perspective overview,” *Evolving Systems*, Apr. 28, 2025 (e-pub ahead of print, Aug. 2024). Accessed: Jun. 24, 2025. [Online]. Available: https://www.researchgate.net/publication/380031685_The_future_of_wireless_mesh_network_in_next-generation_communication_a_perspective_overview

[21] Berto, R.; Napoletano, P.; Savi, M. “A LoRa-Based Mesh Network for Peer-to-Peer Long-Range Communication,” *Sensors*, vol. 21, no. 13, p. 4314, Jun. 24 2021. Accessed: Jun. 24, 2025. [Online]. Available: https://doi.org/10.3390/s21134314

[22] Roots Analysis. “Wireless Mesh Network Market, Till 2035,” *Roots Analysis*, published recently. Accessed: Jun. 24, 2025. [Online]. Available: https://www.rootsanalysis.com/wireless-mesh-network-market

[23] MDPI. “Performance Evaluation of a Mesh-Topology LoRa Network,” May 2025. Accessed: Jun. 24, 2025. [Online]. Available: https://www.mdpi.com/1424-8220/25/5/1602

[24] Wikipedia. “Meshtastic,” updated Jun. 2025. Accessed: Jun. 24, 2025. [Online]. Available: https://en.wikipedia.org/wiki/Meshtastic

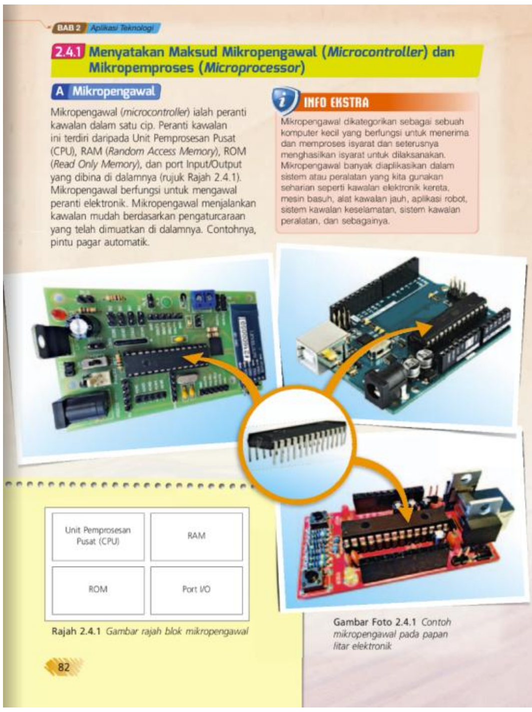
11.0 Appendix

KSSM Form 4 Physics — Relevance to Types of Waves

Table 5.11 Application for each type of electromagnetic wave in daily life

Type of wave	Application
Radio wave	• Long distance radio communication • Local radio and TV broadcasting • Wireless communication (*Bluetooth, Wifi, Zigbee and Z-Wave*) • Millimeter-wave machine to scan body of passengers at airport
Microwave	• International communication through use of satellite • Mobile phone framework • Communication between electronic devices: *Wifi, Bluetooth, Zigbee, Z-Wave* • Detection of plane radar and speed trap • Cooking using microwave oven
Infrared ray	• For cooking (oven, grill and toaster) • For night vision (infrared camera and infrared binoculars) • Drying paint on car • Treatment of muscle pain • Remote control device for television and DVD player
Visible light	• Enables living things to see • Photography • Photosynthesis in green plants • Laser light used in cutting of metal, measurement of land and sending of information through optical fibres
Ultraviolet ray	• Hardens tooth filling material • Determines authenticity of currency notes • Treatment of jaundice in babies • Purification of drinking water • Sterilising surgical instruments and food • Insect traps
X-ray	• Detects fractures or broken bones and examines internal organs • Checking of welding connections • Baggage scanning at airport • Determines authenticity of paintings
Gamma ray	• Kills cancer cells in radiotherapy • Sterilisation of surgical and medical equipment in bulk • Used in food processing industry so that food can last longer

KSSM Form 2 *Reka Bentuk dan Teknologi* — Relevance to Microcontroller Usage



BAB 2 *Aplikasi Teknologi*

2.4.1 Menyatakan Maksud Mikropengawal (*Microcontroller*) dan Mikropemproses (*Microprocessor*)

A Mikropengawal

Mikropengawal (*microcontroller*) ialah peranti kawalan dalam satu cip. Peranti kawalan ini terdiri daripada Unit Pemprosesan Pusat (CPU), RAM (*Random Access Memory*), ROM (*Read Only Memory*), dan port Input/Output yang dibina di dalamnya (rujuk Rajah 2.4.1). Mikropengawal berfungsi untuk mengawal peranti elektronik. Mikropengawal menjalankan kawalan mudah berdasarkan pengaturcaraan yang telah dimuatkan di dalamnya. Contohnya, pintu pagar automatik.

INFO EKSTRA

Mikropengawal dikategorikan sebagai sebuah komputer kecil yang berfungsi untuk menerima dan memproses isyarat dan seterusnya menghasilkan isyarat untuk dilaksanakan. Mikropengawal banyak diaplikasikan dalam sistem atau peralatan yang kita gunakan seharian seperti kawalan elektronik kereta, mesin basuh, alat kawalan jauh, aplikasi robot, sistem kawalan keselamatan, sistem kawalan peralatan, dan sebagainya.

Rajah 2.4.1 *Gambar rajah blok mikropengawal*

Gambar Foto 2.4.1 *Contoh mikropengawal pada papan litar elektronik*