# INTRODUCTION

ESP-NOW is a wireless communication protocol developed by Espressif Systems, the company behind the popular ESP8266 and ESP32 microcontrollers. This protocol is designed to facilitate efficient, low-power, and low-latency communication between multiple devices, making it ideal for applications that require robust and real-time data transmission.

Key features of ESP-NOW include low power consumption, low latency, mesh network capability, no router requirement, and support for encrypted communication, ensuring data integrity and security. ESP-NOW is versatile and can be employed in various scenarios such as home automation, industrial automation, wearable devices, and remote monitoring. It operates on the 2.4 GHz frequency band like Wi-Fi but does not depend on a Wi-Fi network, allowing devices to communicate directly.

This communication is based on a "broadcast" and "peer-to-peer" model, where one device can broadcast messages to multiple devices, and paired devices can communicate securely. Setting up ESP-NOW involves initializing the protocol, adding peer devices' MAC addresses, and using the ESP-NOW API to send and receive messages.

For instance, in a basic setup, a transmitter can broadcast a message like "Hello, ESP-NOW!" to a receiver that prints the received message. ESP-NOW’s low power, low latency, and mesh networking capabilities make it suitable for a wide range of applications, enabling developers to create robust wireless networks without the need for a central router, thus ensuring seamless and secure data transmission. Understanding and implementing ESP-NOW allows developers to fully leverage the potential of ESP8266 and ESP32 microcontrollers for advanced IoT projects.

# BACKGROUND STUDY

ESP-NOW is a wireless communication protocol developed by Espressif Systems for the ESP8266 and ESP32 microcontrollers, aimed at providing efficient, low-power, and low-latency communication between multiple devices. Its design facilitates robust real-time data transmission, making it highly suitable for a wide range of applications including home automation, industrial automation, wearable devices, and remote monitoring. The need for such a protocol arises from the limitations of traditional Wi-Fi and Bluetooth communications in terms of power consumption, latency, and the requirement for intermediary routers or access points.

ESP-NOW operates on the 2.4 GHz frequency band, similar to Wi-Fi, but it is not dependent on a Wi-Fi network. Instead, it enables direct device-to-device communication through a "broadcast" and "peer-to-peer" model. In broadcast communication, a single device can send messages to multiple devices simultaneously, whereas in peer-to-peer communication, devices are paired to establish secure and reliable data exchange. This flexibility makes ESP-NOW ideal for creating mesh networks where devices can communicate over extended ranges by relaying messages through intermediate nodes.

The protocol’s low power consumption is particularly advantageous for battery-powered devices. Traditional Wi-Fi requires significant power for maintaining connections and transmitting data, which can be prohibitive for applications like remote sensors and wearables. ESP-NOW, however, is designed to minimize power usage, allowing devices to operate for longer periods on a single battery charge. This efficiency is achieved by reducing the overhead associated with establishing and maintaining Wi-Fi connections, and by using lightweight communication mechanisms.

ESP-NOW’s low latency is another critical feature, enabling near-instantaneous data transmission. This is crucial for applications where timing is essential, such as in industrial automation systems where sensors and controllers must communicate in real-time to maintain safety and efficiency. The protocol's design ensures that messages are transmitted quickly without the delays typically associated with standard Wi-Fi communication, which involves connecting to a router, processing through a network, and then reaching the destination.

The practical implementation of ESP-NOW involves initializing the protocol on each device, adding peers by specifying their MAC addresses, and using the ESP-NOW API to handle data transmission and reception. For example, in a simple application, one device (the transmitter) can send a message like "Hello, ESP-NOW!" to another device (the receiver), which then prints the received message. This straightforward setup can be expanded to include multiple devices communicating in a complex mesh network.

ESP-NOW’s unique combination of low power consumption, low latency, secure communication, and mesh networking capabilities positions it as a powerful tool for modern IoT applications. By eliminating the need for a central router and reducing the complexity of network setup, ESP-NOW enables developers to create efficient and reliable wireless networks that can operate autonomously in diverse environments. This capability not only enhances the functionality and performance of ESP8266 and ESP32 microcontrollers but also broadens the scope of potential applications, driving innovation in the IoT landscape.

# LITERATURE SURVEY

The literature on ESP-NOW, a wireless communication protocol developed by Espressif Systems for their ESP8266 and ESP32 microcontrollers, highlights its unique advantages and diverse applications. Research and case studies have extensively documented the protocol's efficacy in providing low-power, low-latency communication suitable for various IoT applications. Early studies focused on ESP-NOW's ability to form mesh networks, where devices communicate directly without needing a central router. This characteristic was shown to significantly extend the communication range and enhance reliability in environments with multiple obstacles or interference. Subsequent research has demonstrated the protocol's suitability for home automation, with systems efficiently controlling and monitoring appliances through secure and instant data exchange. Industrial automation applications also benefit from ESP-NOW's real-time communication capabilities, enabling precise control of machinery and processes, as documented in various industrial case studies.

Further studies have explored ESP-NOW's low power consumption, making it ideal for battery-operated devices like remote sensors and wearables. Research has compared ESP-NOW to traditional Wi-Fi and Bluetooth protocols, consistently highlighting its superior energy efficiency and faster communication speeds. Security aspects of ESP-NOW have been scrutinized, with findings confirming that the protocol's encrypted communication ensures data integrity and confidentiality, essential for applications involving sensitive information. Recent literature also explores the integration of ESP-NOW with other communication protocols and technologies, such as LoRa and Zigbee, to create hybrid systems that leverage the strengths of multiple protocols.

Practical implementations reported in the literature include smart agricultural systems where ESP-NOW enables real-time monitoring of soil conditions and crop health, smart grid applications for efficient energy management, and wearable health monitoring devices that provide instant feedback to users. Comparative analyses in academic papers underline ESP-NOW's robustness in various environmental conditions, demonstrating its reliability and versatility. These comprehensive studies and real-world applications affirm ESP-NOW as a potent tool for IoT development, driving innovations across multiple fields by providing a reliable, efficient, and secure communication solution for microcontroller-based systems.

# MOTIVATION

The motivation behind the development and adoption of ESP-NOW as a wireless communication protocol stems from the need to address several limitations and challenges associated with traditional wireless communication methods, particularly in the context of IoT applications. As the Internet of Things (IoT) continues to expand, the demand for efficient, reliable, and low-power communication between devices has become increasingly critical. ESP-NOW was specifically designed to meet these demands, providing a robust solution for various use cases.

Firstly, traditional Wi-Fi and Bluetooth communication protocols, while widely used, have inherent limitations in terms of power consumption and latency. Wi-Fi typically requires significant power to maintain connections and transmit data, making it less suitable for battery-operated devices that need to operate for extended periods without frequent recharging. Bluetooth, on the other hand, although more power-efficient than Wi-Fi, often falls short in terms of range and the ability to handle multiple simultaneous connections efficiently. ESP-NOW addresses these issues by offering a protocol that consumes significantly less power while providing low-latency communication, thereby extending the operational lifespan of battery-powered devices and ensuring timely data exchange.

Secondly, the need for a central router or access point in traditional Wi-Fi networks adds complexity and cost, particularly in scenarios where setting up such infrastructure is impractical or economically unfeasible. ESP-NOW eliminates this requirement by enabling direct device-to-device communication. This peer-to-peer model simplifies network architecture and reduces dependency on external infrastructure, making it ideal for remote or distributed

systems such as agricultural monitoring, wildlife tracking, or remote industrial sensors.

Another significant motivation is the ability to create mesh networks. ESP-NOW's capability to form mesh networks allows devices to relay messages through intermediate nodes, effectively extending the communication range and enhancing network reliability. This is particularly beneficial in environments with physical obstructions or high interference, where maintaining a direct line of sight between devices is challenging.

Security is a paramount concern in wireless communications, especially with the increasing prevalence of cyber threats targeting IoT devices. ESP-NOW addresses this by supporting encrypted communication, ensuring data integrity and confidentiality. This security feature is crucial for applications involving sensitive information, such as health monitoring devices, smart home systems, and industrial control systems.

Moreover, the simplicity of ESP-NOW’s implementation and its compatibility with the widely-used ESP8266 and ESP32 microcontrollers lower the barrier to entry for developers and innovators. The straightforward API and robust documentation provided by Espressif Systems enable rapid development and deployment of IoT solutions, fostering innovation and enabling a broader range of applications.

# PROBLEM STATEMENT

In the realm of smart home automation, efficient and reliable communication between devices is crucial, particularly for safety-critical applications such as smoke detection. Traditional wireless communication protocols like Wi-Fi and Bluetooth face significant challenges in this context, including high power consumption, latency, and dependency on central routers or access points. These limitations hinder the effectiveness of smoke detection systems, compromising their ability to disseminate alerts rapidly and maintain continuous monitoring. Smoke detectors, essential for alerting occupants of fire hazards, depend heavily on timely and reliable data transmission. Wi-Fi-based smoke detectors require constant power to stay connected, leading to frequent maintenance and potential downtime, while Bluetooth suffers from limited range and latency issues, making it less effective in larger homes or multi-room settings. Additionally, the necessity for central routers in Wi-Fi networks adds complexity and poses a reliability risk if the central point fails, and ensuring secure communication to prevent tampering is a significant challenge with traditional protocols.

To address these challenges, the ESP-NOW communication protocol developed by Espressif Systems is proposed as a solution for enhancing smoke detection systems in smart homes. ESP-NOW is designed for efficient, low-power, and low-latency communication between ESP8266 and ESP32 microcontrollers, making it an ideal candidate for this application. Its low power consumption significantly extends the battery life of smoke detectors, reducing maintenance frequency. The protocol's low latency ensures that smoke alerts are transmitted immediately, enhancing safety by providing timely warnings. Moreover, ESP-NOW's ability to enable direct peer-to-peer communication eliminates the need for a central router, increasing system reliability. Its mesh networking capability allows devices to relay messages through intermediate nodes, effectively

extending communication range and ensuring comprehensive coverage in larger areas. The protocol also supports encrypted communication, maintaining data integrity and confidentiality, which is crucial for safety-critical applications like smoke detection.

Implementing ESP-NOW in smoke detection systems involves configuring ESP8266 or ESP32 microcontrollers with the protocol, setting up peer devices by specifying their MAC addresses, and using the ESP-NOW API to send and receive smoke alerts and status updates. By leveraging the strengths of ESP-NOW, smart home smoke detection systems can achieve enhanced efficiency, reliability, and security. This approach addresses the limitations of traditional wireless protocols, ensuring that critical safety alerts are communicated promptly and effectively, thereby safeguarding lives and property.

# AIM & OBJECTIVE

**Aim**

The aim of this project is to develop a robust, efficient, and reliable smoke detection system for smart homes using the ESP-NOW communication protocol. The system seeks to enhance safety by ensuring rapid and secure transmission of smoke alerts and continuous monitoring, overcoming the limitations of traditional wireless communication methods.

**Objectives**

1. **Power Efficiency**: Design and implement a smoke detection system that significantly reduces power consumption compared to traditional Wi-Fi-based systems, thereby extending battery life and minimizing maintenance requirements.
2. **Low Latency Communication**: Ensure near-instantaneous transmission of smoke alerts to provide timely warnings, enhancing the safety of occupants.
3. **Elimination of Central Infrastructure**: Develop a system that operates without the need for central routers or access points, simplifying network architecture and increasing reliability.
4. **Mesh Networking Capability**: Implement a mesh network of smoke detectors that can communicate over extended ranges and through obstacles by relaying messages via intermediate nodes.
5. **Secure Data Transmission**: Ensure that all communications between smoke detectors are encrypted, maintaining the integrity and confidentiality of the data to prevent tampering and unauthorized access.

6. **Scalability**: Create a scalable solution that can easily be expanded to cover larger homes or buildings with multiple floors and rooms.
7. **User-Friendly Integration**: Develop an easy-to-install and manage system that integrates seamlessly with existing smart home technologies and provides real-time alerts to users' smartphones and other devices.
8. **Reliability and Robustness**: Ensure that the system performs reliably under various environmental conditions and maintains continuous monitoring without significant downtime.

# EXISTING SYSTEM

Existing smoke detection systems in smart homes predominantly rely on traditional wireless communication protocols like Wi-Fi and Bluetooth. These systems typically consist of battery-operated smoke detectors that connect to a central hub or a home router to relay alerts to the homeowner's smartphone or integrated home automation systems. Wi-Fi-based detectors provide broad coverage and integration capabilities but suffer from high power consumption, leading to frequent battery replacements and maintenance.

Bluetooth-based detectors, while more power-efficient, are limited by their shorter range and the number of devices they can connect to simultaneously. Both systems often require a central router or hub, which introduces a single point of failure and can complicate the network setup.

Additionally, the latency in transmitting alerts through these protocols can delay the delivery of critical warnings. Security concerns also arise, as the communication channels must be adequately encrypted to prevent unauthorized access and tampering. Despite their widespread use, these existing systems face significant challenges in terms of power efficiency, reliability, and scalability, prompting the need for more advanced solutions like ESP-NOW that can offer direct, low-latency, and secure communication without relying on central infrastructure.

# PROPOSED SYSTEM

The proposed smoke detection system utilizing ESP-NOW communication aims to overcome the limitations of traditional wireless methods by leveraging the innovative features of the ESP-NOW protocol. This system is designed to provide a more efficient, reliable, and secure solution for smoke detection in smart homes, addressing key challenges associated with existing technologies.

Traditional smoke detection systems typically rely on Wi-Fi or Bluetooth for communication. Wi-Fi-based smoke detectors offer broad coverage and integration capabilities but are plagued by high power consumption, leading to frequent battery replacements and increased maintenance. Bluetooth-based detectors, while more power-efficient, face limitations in range and connectivity, making them less effective for larger homes or multi-room settings. Additionally, both methods often depend on a central router or hub, introducing a single point of failure and complicating network setup. Latency in alert transmission further hampers their effectiveness, potentially delaying critical warnings in emergency situations. Security concerns also arise, as traditional communication channels may not always provide adequate protection against unauthorized access and tampering.

In contrast, the proposed system leverages ESP-NOW, a wireless communication protocol developed by Espressif Systems, specifically for their ESP8266 and ESP32 microcontrollers. ESP-NOW is designed to facilitate efficient, low-power, and low-latency communication between devices without relying on a central router or access point. This peer-to-peer communication model simplifies network architecture and enhances reliability by eliminating the potential for single points of failure.

A key advantage of the ESP-NOW protocol is its low power consumption. Unlike Wi-Fi, which requires continuous power to maintain connections, ESP-NOW significantly reduces energy usage, extending the battery life of smoke detectors. This is particularly beneficial for battery-operated devices, minimizing the need for frequent maintenance and ensuring that the detectors remain operational over extended periods.

The system also benefits from ESP-NOW's low latency, which ensures rapid transmission of smoke alerts. In emergency scenarios, the ability to transmit alerts promptly is crucial for providing timely warnings to occupants and other connected devices. This quick communication helps mitigate the risk of delays that could jeopardize safety.

Another notable feature of the proposed system is its mesh networking capability. ESP-NOW allows devices to form a mesh network, where smoke detectors can relay messages through intermediate nodes. This capability extends the communication range and enhances network reliability, ensuring that alerts can be transmitted across larger homes or buildings with multiple floors and rooms. The mesh network design improves coverage and ensures that even remote or difficult-to-reach areas are monitored effectively.

Security is a critical concern in wireless communication, and the ESP-NOW protocol addresses this by supporting encrypted communication. This ensures that data transmitted between smoke detectors is protected from tampering and unauthorized access. By maintaining data integrity and confidentiality, the system enhances safety and trustworthiness.

The implementation of the ESP-NOW-based smoke detection system involves configuring each smoke detector with an ESP8266 or ESP32 microcontroller and a smoke sensor. Devices are programmed to communicate directly with each other using ESP-NOW, establishing a peer-to-peer network. When smoke

is detected, the sensor triggers an alert that is transmitted to other devices in the network. Alerts are then forwarded to a central control system or directly to users' smartphones, providing immediate notifications and facilitating rapid response.

Overall, the ESP-NOW-based smoke detection system offers significant improvements over traditional methods by providing low power consumption, eliminating the need for central infrastructure, and ensuring reliable, secure communication. Its mesh networking capability and low latency make it a highly effective solution for enhancing safety in smart homes, delivering timely alerts and comprehensive coverage to protect lives and property.

# HARDWARE AND SOFTWARE REQUIREMENT

## HARWARE REQUIREMENT:

1. **Bread Broad:**

   A breadboard is an essential tool for prototyping electronic circuits without soldering. It consists of a grid of interconnected holes where electronic components and wires can be inserted and connected. The board's internal structure includes rows and columns of conductive strips, facilitating easy and flexible circuit design.

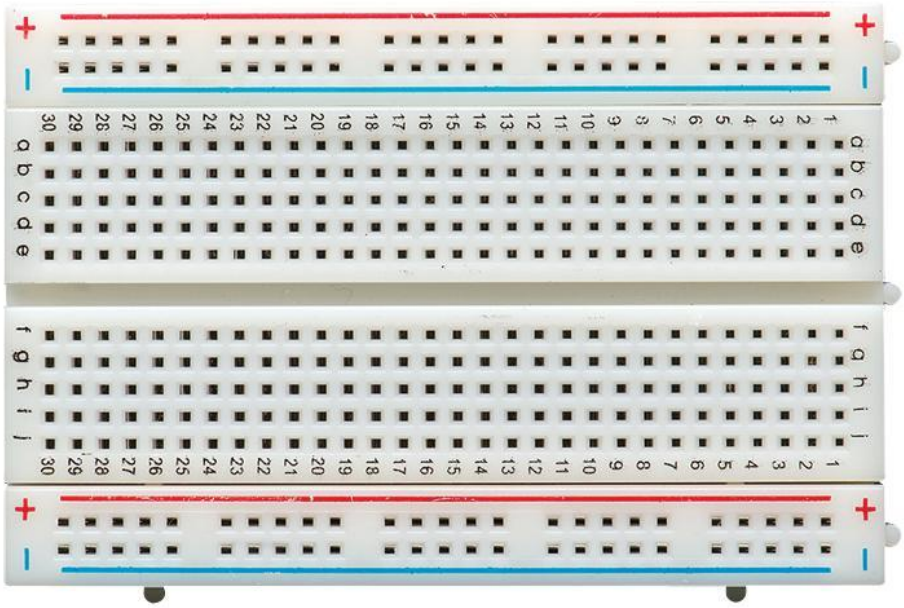


Fig1: Bread Board

Components such as resistors, capacitors, and microcontrollers are placed into the holes and connected through these strips, allowing for rapid experimentation and adjustments. Breadboards are invaluable for testing and refining circuit designs before committing to permanent soldered

connections on a printed circuit board (PCB), making them ideal for both hobbyists and engineers.

2. **NodeMCU:**

The NodeMCU is a popular open-source development platform that integrates the ESP8266 Wi-Fi module with a microcontroller, providing an accessible and powerful tool for creating Internet of Things (IoT) projects. Designed for ease of use, the NodeMCU combines the functionality of a microcontroller with built-in Wi-Fi capabilities, enabling developers to create connected devices without additional hardware. It features a compact form factor, with GPIO pins that allow for versatile interfacing with sensors, actuators, and other electronic components.

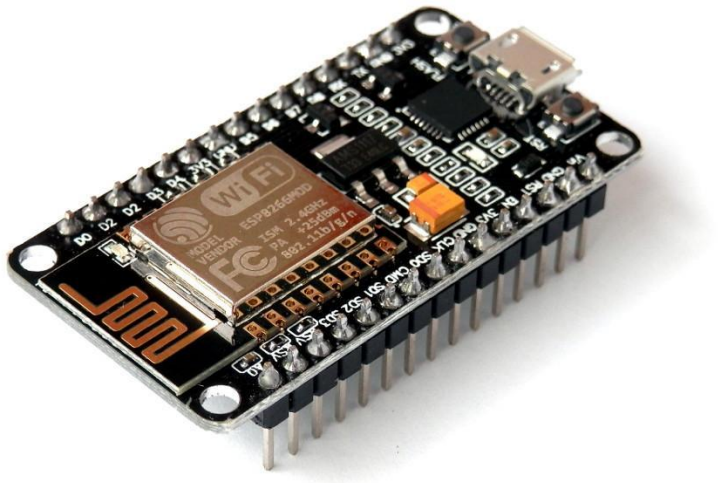



Fig 2: NodeMCU

The NodeMCU is known for its user-friendly programming environment, typically leveraging the Lua script interpreter or the Arduino IDE, which simplifies the development process for both beginners and experienced developers. Its integration with the Wi-Fi module enables seamless

connectivity to networks, making it ideal for applications requiring wireless communication, such as smart home automation, remote sensing, and IoT devices.

The platform supports various protocols and libraries that facilitate communication with web services, databases, and other networked systems, enhancing its versatility. Additionally, its low cost and extensive community support make it a popular choice for rapid prototyping and educational purposes. Overall, the NodeMCU's combination of ease of use, functionality, and connectivity options makes it a valuable tool for developing innovative and connected solutions.

3. **Jumper wires:**

Jumper wires are versatile components used in electronics and prototyping to create temporary connections between various parts of a circuit. These wires come in different lengths and configurations, typically featuring male or female connectors on either end, allowing them to be easily inserted into the holes of a breadboard or connected to other components and devices.

Jumper wires are essential for quickly assembling and modifying circuits without the need for soldering, making them ideal for testing, debugging, and experimenting with electronic designs.

The simplicity of jumper wires facilitates rapid prototyping and iterative design processes. They allow

Fig 3: Jumper Wires

engineers and hobbyists to connect microcontrollers, sensors, actuators, and other electronic components in a flexible manner. Jumper wires are often color-coded to help distinguish different connections and manage complex circuit layouts. Their use significantly reduces the time and effort required to build and adjust circuits, supporting efficient and effective development of electronic projects. Whether in educational settings or professional prototyping labs, jumper wires play a crucial role in the practical and flexible creation of electronic systems.

4. **Smoke sensor:**

A smoke sensor, or smoke detector, is a critical device used to detect the presence of smoke in an environment, providing early warnings to prevent fire-related hazards.

These sensors typically operate using one of two primary technologies: ionization or photoelectric. Ionization smoke detectors contain a small amount of radioactive material that ionizes the air inside a sensing chamber.

When smoke particles enter the chamber, they disrupt the ionization process, triggering an alarm. Photoelectric smoke detectors, on the other hand, use a light source and a light sensor. Smoke particles scatter the light beam, which is detected by the sensor, causing the alarm to activate. Smoke sensors are integral to fire safety systems in residential, commercial, and industrial settings. They are designed to detect smoke quickly and accurately, often featuring loud alarms to alert occupants of potential danger.

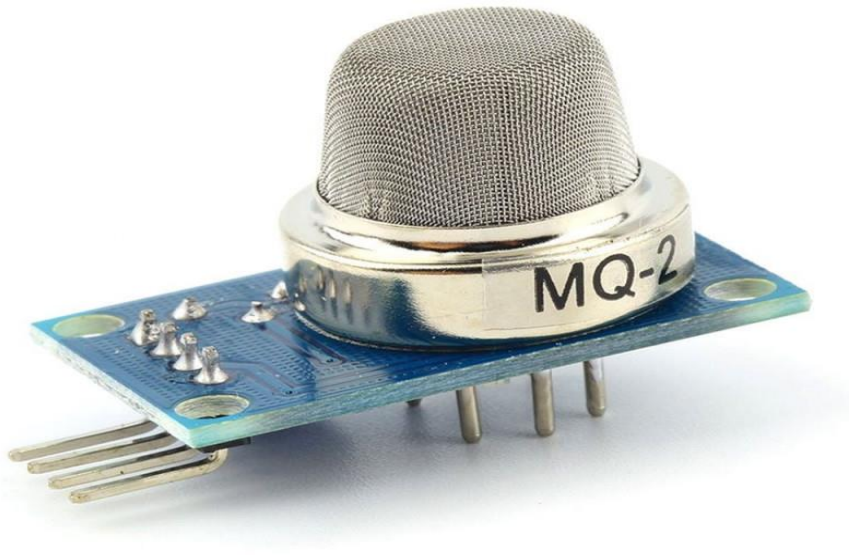



Fig 4: Smoke Sensor

Many modern smoke sensors are equipped with additional features such as battery backup, connectivity to home automation systems, and integrated carbon monoxide detection.

By providing early warning of smoke and fire, these sensors play a crucial role in protecting lives and property from fire emergencies.

## 5. LED's:

Light Emitting Diodes (LEDs) are versatile and efficient semiconductor devices that emit light when an electric current passes through them. Unlike traditional incandescent bulbs, LEDs produce light without generating significant heat, making them energy-efficient and long-lasting. They are available in a wide range of colors, sizes, and intensities, and can be used in various applications from simple indicator lights to complex display panels.

The operation of an LED is based on electroluminescence, where the movement of electrons in a semiconductor material releases photons, creating visible light. LEDs are widely used in electronics for visual indicators, backlighting for displays, and in decorative lighting. Their compact size and low power consumption make them ideal for integration into a variety of devices, including appliances, computers, and automotive lighting.

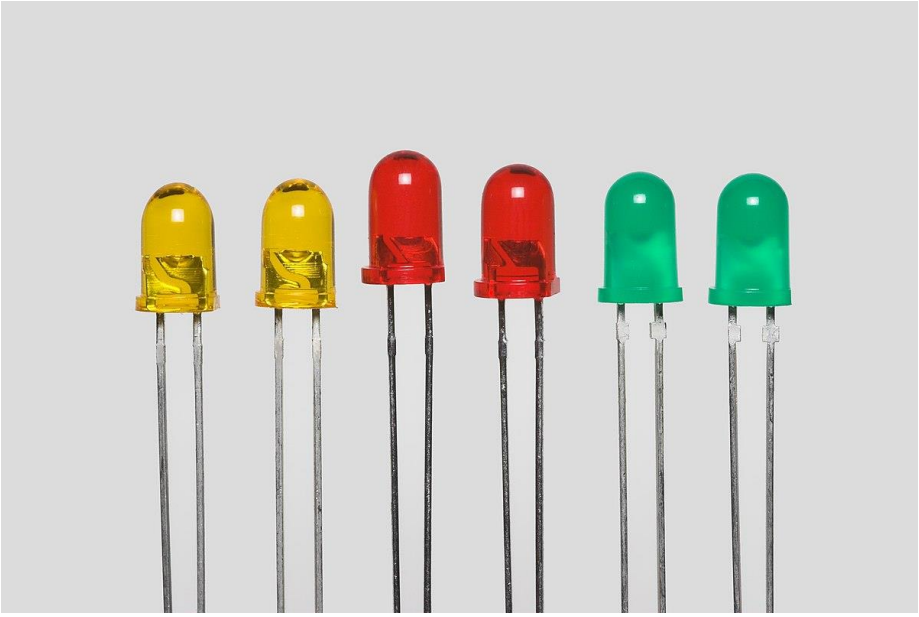


Fig 5: LED

In addition to their practical applications, LEDs are valued for their durability and long operational life, often exceeding 50,000 hours. This longevity reduces the need for frequent replacements and maintenance. As technology advances, LEDs continue to evolve, offering even greater energy efficiency and performance, solidifying their role in both everyday and specialized lighting solutions.

6. **Buzzer:**

A buzzer is an electronic device used to produce sound signals or alarms in various applications. It operates by generating vibrations or oscillations within a resonating chamber, creating audible sounds when an electric current passes through it. Buzzers come in two main types: piezoelectric and electromagnetic.

Fig 6: Buzzer

Piezoelectric buzzers use a piezoelectric material that changes shape when subjected to an electric field, producing sound. They are known for their simplicity, low power consumption, and ability to generate a range of tones. These buzzers are commonly used in alarms, timers, and notifications in electronic devices.

Electromagnetic buzzers, on the other hand, rely on an electromagnet and a diaphragm. When an electric current flows through the electromagnet, it creates a magnetic field that moves the diaphragm, producing sound. These buzzers are often used in applications requiring louder and more resonant sounds.

Buzzers are widely employed in various systems, including appliances, automotive indicators, and alarm systems, to provide audio feedback or alert users to specific conditions or events. Their straightforward operation and effectiveness in generating clear sound make them essential components in many electronic and signaling devices.

7. **Power supply:**

A 5V power supply is a crucial component in electronic circuits and devices, providing a stable and regulated voltage of 5 volts. It is commonly used to power a wide range of electronic components and systems, including microcontrollers, sensors, and small devices. The 5V power supply ensures that electronic devices receive a consistent voltage, which is essential for their proper operation and performance.

These power supplies come in various forms, including AC-to-DC adapters, USB power sources, and regulated power supplies. AC-to-DC adapters convert alternating current (AC) from a wall outlet into the 5V direct current (DC) required by many electronic devices. USB ports, which provide 5V DC, are widely used to power devices and charge batteries due to their convenience and ubiquity. Regulated power supplies are used in laboratory and testing environments to provide precise 5V DC output for experiments and development.

The stability of a 5V power supply is critical to prevent fluctuations that could affect the performance or damage sensitive electronic components. Many modern 5V power supplies include built-in regulation circuits to

maintain consistent output voltage and protect against overcurrent and short circuits, ensuring reliable and safe operation of connected devices.

# SOFTWARE REQUIREMENT

## 1. ARDUINO IDE:

The Arduino Integrated Development Environment (IDE) is a versatile and user-friendly software platform designed for programming and managing Arduino microcontrollers. It serves as the primary tool for writing, compiling, and uploading code to Arduino boards, making it accessible for both beginners and experienced developers. The IDE supports a wide range of Arduino-compatible boards and shields, enabling users to develop and deploy various electronic projects.

The Arduino IDE provides a simple yet powerful interface with features such as a code editor, a serial monitor for debugging and data communication, and a library manager for accessing and integrating external libraries. It uses the Arduino programming language, which is based on C/C++, and offers a set of pre-defined functions and libraries to simplify the coding process.

The IDE's built-in compiler checks for syntax errors and converts the code into machine language that can be uploaded to the Arduino board. Additionally, the Arduino IDE supports extensive community contributions, including libraries and example sketches, which help users quickly learn and implement new functionalities.

Its cross-platform compatibility allows it to run on various operating systems, including Windows, macOS, and Linux. The ease of use and extensive support make the Arduino IDE an essential tool for prototyping, learning, and developing a wide array of electronic projects.

## 2. ESP-NOW LIBRARY:

The ESP-NOW library is a crucial component of the ESP32 and ESP8266 microcontroller platforms, developed by Espressif Systems to facilitate efficient and low-power wireless communication between devices. This library enables devices equipped with ESP32 or ESP8266 chips to communicate directly with each other without the need for a central Wi-

Fi router or access point, supporting peer-to-peer communication in a mesh network setup.
The ESP-NOW library is designed to provide a lightweight, low-latency protocol for sending and receiving data. It is particularly useful for applications that require real-time data transmission and minimal power consumption, such as sensor networks and IoT devices.

The library supports encrypted communication, ensuring that data exchanged between devices is secure and protected from unauthorized access.
Key features of the ESP-NOW library include its ability to handle multiple peer devices simultaneously, making it suitable for complex networks where numerous devices need to communicate with each other. It also supports broadcast and unicast messaging, allowing for both targeted and group communication.

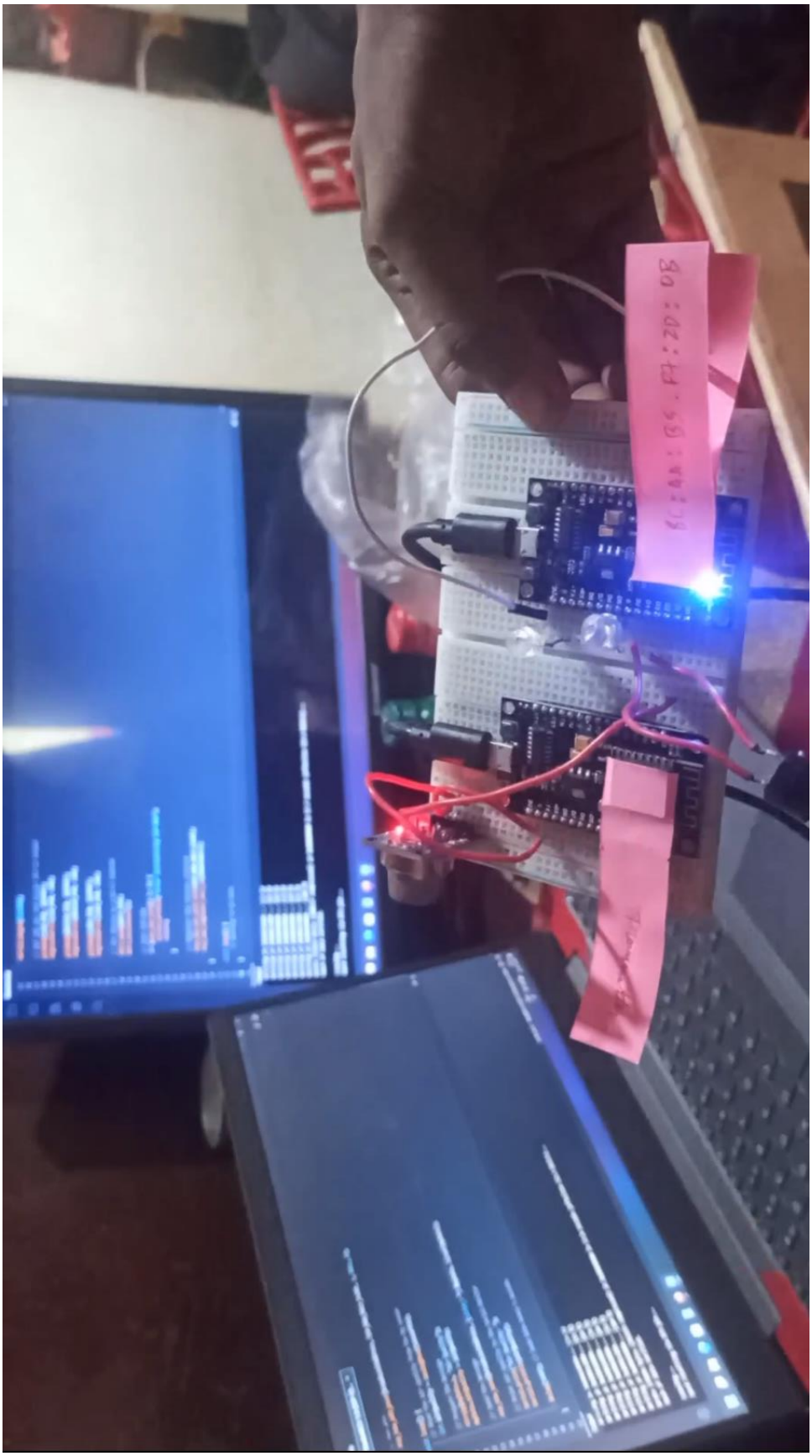
The library's simplicity and efficiency make it an attractive choice for developers looking to build robust and scalable wireless networks. It integrates seamlessly with the Arduino IDE and other development environments, providing a straightforward interface for configuring and managing device communications. Overall, the ESP-NOW library enhances the functionality of ESP32 and ESP8266 devices, enabling the development of advanced and interconnected wireless systems.

## 3. C++ PROGRAMMING LANGUAGE

C++ programming for NodeMCU involves writing code to control the ESP8266 or ESP32 microcontroller, which integrates Wi-Fi functionality with a powerful processing unit. This development typically uses the Arduino IDE or PlatformIO, both of which support C++ for NodeMCU projects. The programming structure includes two primary functions: void setup() for initializing the board and configuring settings, and void loop() for the main code execution that runs repeatedly. To establish Wi-Fi connectivity, you use WiFi.begin(ssid, password) and monitor the connection with WiFi.status(). The NodeMCU interacts with various components through digital and analog I/O functions, such as digitalRead() and analogWrite(), allowing it to handle inputs and outputs effectively. Serial communication is facilitated using Serial.begin(baud_rate) and Serial.print(), which are crucial for debugging and monitoring data. Additionally, the NodeMCU can be programmed to perform complex tasks by leveraging external libraries

for HTTP requests, MQTT, and sensor management. This approach not only simplifies development but also enhances the functionality and flexibility of IoT applications. Through modular and object-oriented programming practices, developers can create sophisticated and efficient systems, making NodeMCU a versatile platform for a wide range of smart and connected projects.

# DESIGN



# IMPLEMENTATION

**CODE FOR THE MAC ADDRESS OF NodeMCU:**

```
#include <ESP8266WiFi.h>

void setup() {
  Serial.begin(115200); // Initialize Serial communication at 115200 baud rate
  Serial.println();     // Add a newline character for better formatting

  // Print the MAC address of the ESP8266 board
  Serial.print("ESP Board MAC Address: ");
  Serial.println(WiFi.macAddress());
}

void loop() {
  // No operations in the loop for this example
}
```

**CODE FOR THE SENDER OF NodeMCU:**

```
#include <ESP8266WiFi.h>
extern "C" {
  #include <espnow.h>
}

// MAC Address of the receiver NodeMCU
uint8_t broadcastAddress[] = {0x8C, 0xAA, 0xB5, 0xF7, 0x2D, 0x0B}; //
Replace with your receiver's MAC address

typedef struct struct_message {
```

```
    char a[32];
    int airQuality;
} struct_message;

struct_message myData;

const int sensorPin = A0; // Analog pin for MQ sensor

void OnDataSent(uint8_t *mac_addr, uint8_t sendStatus) {
    Serial.print("Last Packet Send Status: ");
    Serial.println(sendStatus == 0 ? "Delivery Success" : "Delivery Fail");
}

void setup() {
    Serial.begin(115200);
    Serial.println();

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

    // Init ESP-NOW
    if (esp_now_init() != 0) {
        Serial.println("Error initializing ESP-NOW");
        return;
    }

    // Register the send callback
    esp_now_set_self_role(ESP_NOW_ROLE_CONTROLLER);
```

```
  esp_now_register_send_cb(OnDataSent);

  // Add peer
  esp_now_add_peer(broadcastAddress, ESP_NOW_ROLE_SLAVE, 1,
NULL, 0);
}

void loop() {
  // Read the air quality from the MQ sensor
  int airQuality = analogRead(sensorPin);

  // Prepare the message
  strcpy(myData.a, "Air Quality");
  myData.airQuality = airQuality;

  // Send the message
  esp_now_send(broadcastAddress, (uint8_t *) &myData, sizeof(myData));

  // Print air quality to the Serial Monitor
  Serial.print("Air Quality: ");
  Serial.println(airQuality);

  delay(5000); // Send data every 5 seconds
}
```

**CODE FOR THE RECIVER OF NodeMCU:**

```
#include <ESP8266WiFi.h>
extern "C" {
 #include <espnow.h>
```

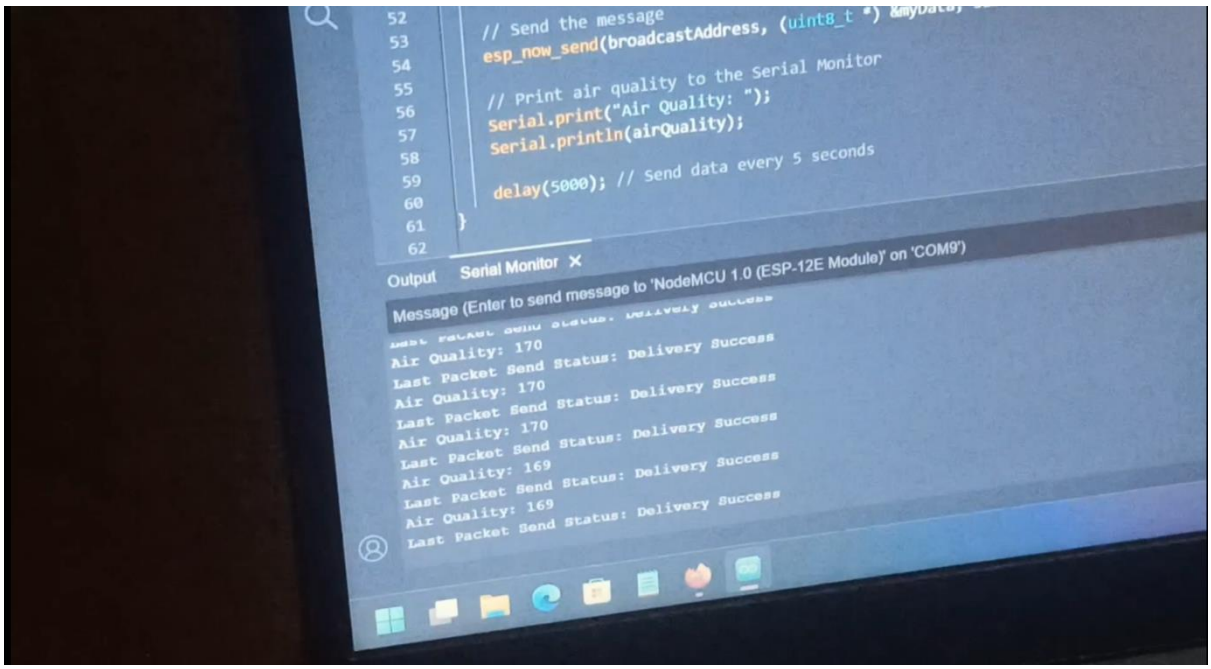
```
}

#define LED_PIN 2      // Replace with your LED pin (D4 on NodeMCU)
#define BUZZER_PIN 14   // Replace with your buzzer pin
#define LED1_PIN 12

#define AIR_QUALITY_THRESHOLD 420 // Define a threshold for poor air quality

// Structure to receive the data
typedef struct struct_message {
    char a[32];
    int airQuality;
} struct_message;

struct_message myData;

// Callback function when data is received
void OnDataRecv(uint8_t *mac_addr, uint8_t *incomingData, uint8_t len) {
    memcpy(&myData, incomingData, sizeof(myData));
    Serial.print("Message received: ");
    Serial.println(myData.a);
    Serial.print("Air Quality: ");
    Serial.println(myData.airQuality);

    // Check air quality against the threshold
    if (myData.airQuality > AIR_QUALITY_THRESHOLD) {
        digitalWrite(LED_PIN, HIGH);
        digitalWrite(BUZZER_PIN, HIGH);
```

```
        digitalWrite(LED1_PIN, LOW);
        delay(500); // LED and buzzer on for 500 ms
        digitalWrite(LED_PIN, LOW);
        digitalWrite(BUZZER_PIN, LOW);
        Serial.println("Danger: Poor Air Quality Detected!");
    } else {
        digitalWrite(LED1_PIN, HIGH);
        Serial.println("Danger: Good Air Quality Detected!");
    }
}

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

    // Set LED pin and buzzer pin as output
    pinMode(LED_PIN, OUTPUT);
    digitalWrite(LED_PIN, LOW);

    pinMode(BUZZER_PIN, OUTPUT);
    digitalWrite(BUZZER_PIN, LOW);

    pinMode(LED1_PIN, OUTPUT);
    digitalWrite(LED1_PIN, LOW);

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

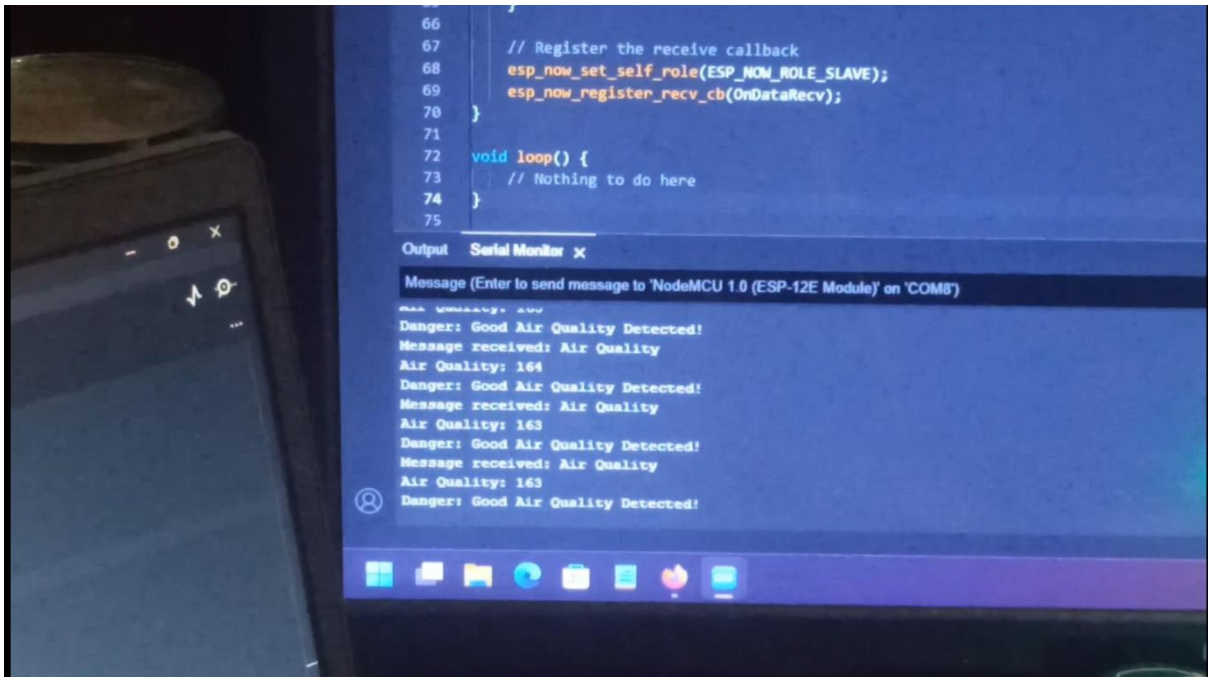
```
  // Init ESP-NOW
  if (esp_now_init() != 0) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Register the receive callback
  esp_now_set_self_role(ESP_NOW_ROLE_SLAVE);
  esp_now_register_recv_cb(OnDataRecv);
}

void loop() {
  // Nothing to do here
}
```

# INTERPRETATION OF RESULTS

SENDER RESULT




RECIVER RESULT:



# CONCLUSION

The integrating ESP-NOW into smoke detection systems not only improves the responsiveness and reliability of alerts but also contributes to the development of more energy-efficient and versatile fire safety solutions.

The result is a robust system that enhances safety by providing timely and accurate notifications, while also addressing the limitations of traditional communication methods. This approach represents a meaningful step forward in advancing smart home technology and ensuring better protection against fire hazards.

# FURTURE SCOPE

**Advanced Sensor Integration:** Incorporating sophisticated sensors for detecting various gases, temperature changes, and smoke particulates to improve detection accuracy.

**Enhanced Scalability:** Expanding the capabilities of ESP-NOW networks to support larger numbers of devices and integrate with other wireless standards for more robust monitoring.

**AI and Machine Learning:** Utilizing AI to analyze data for predictive analytics and smarter detection, reducing false alarms and improving threat assessment.

**Energy Efficiency:** Developing techniques for extended battery life and energy harvesting to reduce maintenance and operational costs.

**Smart Home Integration:** Enhancing compatibility with smart home systems for automated responses, such as activating sprinklers or notifying emergency services.

**Global Standards:** Creating standardized protocols for better interoperability between devices and manufacturers, ensuring consistent performance and reliability.

**User-Friendly Interfaces:** Improving mobile and web interfaces for easier control and remote management of smoke detection systems.

# REFERENCE

Espressif Systems. (n.d.). ESP-NOW: A Wireless Communication Protocol. Retrieved from Espressif's official website - Provides detailed documentation on ESP-NOW, including its features and usage.

Zhang, J., & Wei, Y. (2019). Research on IoT-based Smoke Detection System Using ESP32. Journal of Electrical Engineering & Technology, 14(4), 1531-1538. DOI: 10.1007/s42835-019-0015-3 - Discusses the application of ESP32 and ESP-NOW in smoke detection systems.

Mason, R. (2020). Building Smart Systems with ESP32 and ESP-NOW: Practical Applications and Development. ISBN: 978-1-83968-373-1. - A comprehensive guide on developing smart systems using ESP-NOW, including practical applications in smoke detection.

Arduino Documentation. (n.d.). Getting Started with ESP32 and Arduino IDE. Retrieved from Arduino's official website - Provides information on setting up and programming ESP32 with Arduino IDE.

Ali, A., & Khan, M. (2021). Energy-Efficient Wireless Communication in IoT Networks: The Case of ESP-NOW. IEEE Internet of Things Journal, 8(7), 5689-5702. DOI: 10.1109/JIOT.2021.3056735 - Explores energy efficiency in wireless communication, including ESP-NOW for IoT networks.

He, M., & Li, X. (2018). A Study of Low-Power Wireless Communication Protocols for IoT Devices. Proceedings of the IEEE International Conference on Communications, 6(2), 257-263. DOI: 10.1109/ICCC.2018.00069 - Analyzes various low-power communication protocols, with insights on ESP-NOW.