# March 15, 2023

Bheema Rajulu

# ESP32-based IoT Sensor Board for Home Safety and Monitoring with Display and Interconnected Cubes.

The ESP32-based IoT sensor board with the ILI9488 display is a useful tool for home safety and monitoring. It can detect and alert the user in case of emergencies and hazards such as gas leaks and fires.

This project aims to design and develop an ESP32-based IoT sensor board for home safety and monitoring with a 3.5-inch TFT LCD display (ILI9488). The board includes various sensors such as gas, smoke, motion, light, temperature, and a 9-axis MEMS sensor, along with a rotary encoder, buzzer, and power supply.

The board is capable of monitoring gas and smoke levels, detecting motion and light levels, measuring temperature, and setting the timer using the rotary encoder module. The sensor data is displayed on the 3.5-inch ILI9488 TFT display module. In case of an emergency or when the timer expires, the board sounds an alarm through the PAM8403 amplifier circuit and 3W 4Ohm speaker and sends notifications to the user's device through MQTT.

The cubes should have intercommunication between them by Wi-Fi connectivity and MQTT protocol. When one cube detects motion, it intercommunicates with other cubes to light up. This interconnected cube can share information and respond to events in the environment.

The board is compact and fits inside a cube-shaped case and has a long battery life that can be recharged using the lithium-ion battery charger module.

The programming is done using the Arduino IDE with the ESP32 core libraries, and MQTT protocol is used for sending and receiving alarm messages. The case is cube-shaped and can be 3D printed or laser cut with acrylic sheets.

The firmware code is written using the Arduino IDE and ESP32 core libraries and follows a modular and scalable architecture with separate modules or classes for each sensor and functionality.

The firmware code can be designed to use multi-threading or task scheduling to run the separate modules or classes for each sensor and functionality concurrently. This allows each module to run independently and in parallel with other modules, ensuring that the Cube can handle multiple events and tasks simultaneously.

The project can be completed within 4-6 weeks with a budget of under $150 per board, including the components, PCB, and case.

Note:  
We could explore option of using M5Stack Core2 ESP32 or ESP32-S3-CUBE development board.

1. Functional Requirements:

* The board should be able to monitor gas and smoke levels using the MQ-2 and MQ-135 sensors respectively.
* The board should be able to detect motion and light levels using the HC-SR501 and BH1750 sensors respectively.
* The board should be able to measure temperature using the DS18B20 sensor.
* The board should be able to set the timer using the KY-040 rotary encoder module.
* The board should be able to display sensor data on the 3.5-inch ILI9488 TFT display module.
* The board should be able to sound an alarm through the PAM8403 amplifier circuit and 3W 4Ohm speaker when the timer expires or in case of no motion for certain period.
* The board should be able to send notifications to the email and messages through MQTT in case of an emergency or when the timer expires.
* The board should be able to intercommunicate with other cubes to light up when one cube detects motion.
* The Cube can also intercommunicate with other Cubes when one sensor detects an event, such as motion or smoke, to trigger actions on other Cubes as well. They will need to be able to communicate with each other.
* Include details on how to set up the communication between the cubes, what protocol should be used (such as Wi-Fi or Bluetooth), and how the cubes can work together to keep the home safe.

1. Non-Functional Requirements:

* The board should be compact and fit inside a cube-shaped case.
* The board should have a long battery life and should be able to recharge using the lithium-ion battery charger module. [Power calculation yet to determine]
* The board should have a responsive and user-friendly interface for setting the timer and displaying sensor data.

1. Deliverables:

* A PCB design that incorporates the ESP32 development board and all the sensors and components.
* A 3D model and laser cutting files for the cube-shaped case.
* Firmware code written in C++ language using the Arduino IDE and ESP32 core libraries.

1. Timeline and Budget:

* The project should be completed within 4-6 weeks.
* The budget for the project should be kept under $150 per board, including the components, PCB, and case.

Hardware:

* ESP32 development board
* Gas sensor (MQ-2)
* Smoke sensor (MQ-135)
* Motion sensor (HC-SR501)
* Light sensor (BH1750)
* Temperature sensor (DS18B20)
* 9-axis MEMS sensor (MPU-9250)
* Rotary encoder module (KY-040)
* Buzzer module with amplifier circuit (PAM8403)
* 3.5-inch ILI9488 TFT display module – We can go with OLED if there are power constraints.
* Lithium-Ion Polymer Battery (3.7V 2500mAh).
* 5V/2A power adapter or USB power bank.
* Cube-shaped case that can be 3D printed or laser cut with acrylic sheets.

## **Component List:**

|  |  |  |
| --- | --- | --- |
| 1.5inch RGB OLED | 18.89USD | <https://rb.gy/rsmisl> |
| TP4056 Lithium-Ion Battery Charger Module | 8 | <https://a.co/d/8gHcQRH> |
| 5V/2A Power Adapter or USB Power Bank | 10 | <https://www.amazon.com/SoulBay-Adapter-Polarity-Regulated-Cordless/dp/B01N2K48HR> |
| Boost Converter  (MT3608) | 1.9USD | https://www.amazon.com/MT3608-DC-DC-Adjustable-Module-2V-24V/dp/B07TLJR2S9/ref=sr\_1\_2?keywords=MT3608&qid=1678440717&sr=8-2 |
| ESD-USB Protection | 0.62 USD | [**USBLC6-4SC6**](https://www.digikey.com/en/products/detail/stmicroelectronics/USBLC6-4SC6/725216) |
| BSS84AK,215 | 0.24USD | <https://www.digikey.com/en/products/detail/nexperia-usa-inc/BSS84AK-215/277983> |
| SS3P5HM3/85A | 0.420USD | https://www.digikey.com/en/products/detail/vishay-general-semiconductor-diodes-division/SS3P5HM3-85A/4495003 |
| Li-Po battery |  | <https://de.aliexpress.com/item/1005004971706101.html?spm=a2g0o.productlist.main.1.1c8c3246XIV4a7&algo_pvid=f88580bc-4d48-45a4-b711-d68ad13ee079&algo_exp_id=f88580bc-4d48-45a4-b711-d68ad13ee079-0&pdp_ext_f=%7B%22sku_id%22%3A%2212000031200285945%22%7D&pdp_npi=3%40dis%21EUR%2119.04%2113.33%21%21%21%21%21%40211bd4cd16785192167772934d0745%2112000031200285945%21sea%21PK%210&curPageLogUid=udvKIZksEBMU> |
| USB C-type | 1.72 USD | [USB4800-03-A](https://eu.mouser.com/ProductDetail/GCT/USB4800-03-A?qs=vvQtp7zwQdMkfG8vViEZoQ%3D%3D) |
| Real time clock  (PCF8563TS/4,118) | 2.12USD | <https://www.digikey.com/en/products/detail/nxp-usa-inc/PCF8563TS-4-118/739488?s=N4IgTCBcDaIAoGEBiAOArANgMwBUDKA9ACwA0AjGSiALoC%2BQA> |
| Microphone I2S  (SPH0645LM4H) | 2.74USD | <https://www.digikey.com/en/products/detail/knowles/SPH0645LM4H-B/5332440> |
| Touch sensor  (TTP223B) | 6.95USD | <https://www.digikey.com/en/products/detail/osepp-electronics-ltd/TOUCH-01/11198546> |
| **Total** | **150.6USD** |  |

**Introduction**

The ESP32-S3-WROOM-2 based IoT Sensor Board for Home Safety and Monitoring with Display and Interconnected Cubes is a cutting-edge solution aimed at enhancing safety, security, and convenience in residential environments. By leveraging the power of the ESP32-S3-WROOM-2 module, which features a powerful and versatile Xtensa® dual-core 32-bit LX7 microprocessor, integrated Bluetooth, and support for Matter connectivity, this system is capable of monitoring various environmental conditions and alerting users to potential hazards. The system also features a 3.5-inch TFT LCD display (ILI9488) for easy visualization of sensor data and a series of interconnected cubes that communicate with each other to create a smart, responsive network.

**Purpose and Benefits**

The primary purpose of the ESP32-S3-WROOM-2 based IoT Sensor Board for Home Safety and Monitoring with Display and Interconnected Cubes is to provide users with real-time information about their home environment, enabling them to act in case of emergencies or potential hazards. The system integrates various sensors such as gas, smoke, motion, light, temperature, and a 9-axis MEMS sensor, which can detect events such as gas leaks, fires, intrusions, and changes in ambient conditions.

Some key benefits of the system include:

1. Enhanced safety and security: The system continuously monitors the home environment, detecting potential hazards and alerting users promptly. By doing so, it helps to mitigate risks and prevent accidents or damages.
2. Real-time monitoring and alerts: The IoT Sensor Board displays sensor data on the 3.5-inch TFT LCD screen, allowing users to keep track of their home's conditions. In case of emergencies or predefined events, the system sends notifications to the user's device through Matter, ensuring that they are always informed.
3. Interconnected cubes: The cubes in the system communicate with each other via the embedded Bluetooth module in the ESP32-S3-WROOM-2, creating a smart network that can react to events in a coordinated manner. For example, when one cube detects motion, it can inform other cubes to light up, creating a responsive environment that enhances safety and convenience.
4. Internet of Things (IoT) server connectivity: Each cube communicates with the IoT server via the Matter protocol, enabling seamless integration with various IoT platforms and services, such as Alexa, Apple HomeKit, Google Home, and IFTTT.
5. Energy efficiency and convenience: The system can be programmed to control various home automation devices, such as lights, heating, and air conditioning, based on sensor data. This not only improves energy efficiency but also enhances the overall user experience.
6. Scalability and customization: The modular design of the system allows for easy expansion and customization. Users can add more sensors or cubes as needed, tailoring the system to their specific requirements.
7. Ease of use and installation: The system is designed to be user-friendly, with a straightforward interface for setting timers and displaying sensor data. The compact cube-shaped design allows for easy installation in various locations throughout the home.

In summary, the ESP32-S3-WROOM-2 based IoT Sensor Board for Home Safety and Monitoring with Display and Interconnected Cubes offers a comprehensive solution for improving safety, security, and convenience in residential environments. By leveraging the power of the ESP32-S3-WROOM-2 module and integrating a wide array of sensors, this system provides users with real-time information and alerts, helping them maintain a safe and comfortable living space.

Overview

The ESP32-S3-WROOM-2 based IoT Sensor Board for Home Safety and Monitoring with Display and Interconnected Cubes is an advanced system that integrates various sensors, communication modules, and a display to provide a comprehensive safety and monitoring solution for residential environments. This high-level overview will describe the key components of the system and their interactions to create a smart, responsive network of interconnected cubes.

System Components:

1. ESP32-S3-WROOM-2 Module: At the heart of each cube lies the ESP32-S3-WROOM-2 module, featuring a powerful Xtensa® dual-core 32-bit LX7 microprocessor, integrated Bluetooth, and support for Matter connectivity. This module processes sensor data, manages cube-to-cube communication via Bluetooth, and communicates with the IoT server through Matter.
2. Sensors: The system incorporates multiple sensors, such as gas, smoke, motion, light, temperature, and a 9-axis MEMS sensor, to monitor various environmental conditions and detect potential hazards in real-time.
3. 3.5-inch TFT LCD Display (ILI9488): Each cube is equipped with a 3.5-inch TFT LCD display for visualizing sensor data and system status, providing users with an easy-to-understand interface.
4. Interconnected Cubes: The cubes in the system are designed to communicate with each other using the embedded Bluetooth module in the ESP32-S3-WROOM-2. This inter-cube communication allows the cubes to share information and respond to events in the environment in a coordinated manner.
5. IoT Server: The IoT server acts as the central hub for managing and processing data from the interconnected cubes. Each cube communicates with the IoT server via Matter, enabling seamless integration with various IoT platforms and services, such as Alexa, Apple HomeKit, Google Home, and IFTTT.

System Interactions:

1. Cube-to-Cube Communication: The cubes in the system communicate with each other using the embedded Bluetooth module. When a cube detects an event, such as motion or smoke, it sends a message to the other cubes in the network. The receiving cubes then process the message and respond accordingly, such as by lighting up or activating an alarm.
2. Cube-to-IoT Server Communication: Each cube communicates with the IoT server using the Matter protocol. This allows the system to send sensor data, alerts, and notifications to the IoT server, which can then be processed and forwarded to the user's device or other connected services.
3. Sensor Data Processing: The ESP32-S3-WROOM-2 module processes sensor data from the various sensors integrated into each cube. This data is then displayed on the 3.5-inch TFT LCD screen and sent to the IoT server for further analysis and action.
4. User Notifications and Alerts: In case of emergencies or predefined events, the system sends notifications and alerts to the user's device through the IoT server and Matter connectivity. This ensures that the user is always informed and can take appropriate action.

In conclusion, the ESP32-S3-WROOM-2 based IoT Sensor Board for Home Safety and Monitoring with Display and Interconnected Cubes is a sophisticated system that combines various sensors, communication modules, and a display to create a smart, responsive network of interconnected cubes. By leveraging cube-to-cube and cube-to-IoT server communication, the system can effectively monitor and respond to events in the environment, enhancing safety, security, and convenience in residential settings.

Architecture

The system architecture consists of hardware and software components that work together to create a robust home safety and monitoring solution. The primary components include the ESP32-S3-WROOM-2 development board, various sensors, and interconnected cubes. The system relies on the ESP32-S3-WROOM-2 chip for processing and communication, leveraging its powerful dual-core 32-bit LX7 microprocessor and rich set of peripherals.

Hardware Components:

1. ESP32-S3-WROOM-2 Development Board: This board serves as the central processing unit and communication hub for the system. It interfaces with all the sensors, handles data processing, and communicates with other cubes via BLE and the IoT server via the Matter protocol.
2. Sensors: The system includes several sensors that monitor different environmental parameters, such as gas, smoke, motion, light, and temperature. These sensors are connected to the ESP32-S3-WROOM-2 development board using various interfaces such as ADC, GPIO, I2C, and SPI.
3. Interconnected Cubes: The cubes house the ESP32-S3-WROOM-2 development board, sensors, and other components in a compact and aesthetically pleasing form factor. They communicate with each other using BLE and can share information and trigger actions in response to events detected by the sensors.
4. IoT Server: The IoT server collects data from all interconnected cubes using the Matter protocol over Wi-Fi or Ethernet. It serves as a central monitoring and control point, allowing users to view the status of the system, receive notifications, and control the cubes remotely.

Software Components:

1. Firmware: The firmware for the ESP32-S3-WROOM-2 development board is developed using the Arduino IDE and ESP32 core libraries. It follows a modular and scalable architecture, with separate modules or classes for each sensor and functionality. The firmware utilizes multi-threading or task scheduling to enable parallel execution of sensor and communication tasks.
2. IoT Server Application: The IoT server application processes the data received from the cubes and provides an interface for users to monitor and control the system. It can be a cloud-based or local server application, depending on the user's preferences and requirements.
3. User Interface: The user interface is designed to provide a simple, intuitive, and responsive experience for the user. It displays sensor data, allows users to set timers, and provides notifications for events detected by the system. The interface can be accessed via a web browser, mobile app, or dedicated control panel.

The hardware and software components of the system interact through various interfaces and protocols to provide a seamless and efficient home safety and monitoring solution. The ESP32-S3-WROOM-2 development board serves as the central processing unit, handling data from sensors and managing communication between interconnected cubes and the IoT server.

**SPECIFICATION**

This section lists the detailed specifications of the system:

Hardware Specifications:

1. ESP32-S3-WROOM-2 Development Board:
   * SoC: ESP32-S3R8V
   * Microprocessor: Xtensa dual-core 32-bit LX7
   * Flash Memory: Up to 32 MB
   * PSRAM: Up to 8 MB
   * GPIOs: 33
   * Communication: Wi-Fi, Bluetooth, BLE, Ethernet (with external PHY)
2. Sensors:
   * Gas sensor (MQ-2): Detects combustible gases (LPG, propane, hydrogen, etc.)
   * Smoke sensor (MQ-135): Detects smoke and air pollutants
   * Motion sensor (HC-SR501): Detects motion using passive infrared (PIR) technology
   * Light sensor (BH1750): Measures ambient light intensity
   * Temperature sensor (DS18B20): Measures temperature with high accuracy
   * 9-axis MEMS sensor (MPU-9250): Measures acceleration, gyroscopic motion, and magnetic fields
3. Additional Components:
   * Rotary encoder module (KY-040): Used for setting timers and navigating menus
   * Buzzer module with amplifier circuit (PAM8403): Provides audible alarms
   * 3.5-inch ILI9488 TFT display module or OLED: Displays sensor data and system status
   * Lithium-Ion Polymer Battery (3.7V 2500mAh): Provides power to the system
   * 5V/2A power adapter or USB power bank: Recharges the battery and powers the system

Software Specifications:

1. Firmware:
   * Developed using Arduino IDE and ESP32 core libraries
   * Modular and scalable architecture
   * Multi-threading or task scheduling for parallel execution of tasks
2. IoT Server Application:
   * Collects and processes data from interconnected cubes
   * Provides monitoring and control interface for users
   * Supports cloud-based or local server deployment
3. User Interface:
   * Intuitive and responsive design
   * Displays sensor data and system status
   * Allows users to set timers and receive notifications

The specifications outlined above provide a comprehensive view of the system's hardware and software components, ensuring a complete and efficient home safety and monitoring solution.

+-----------+ BLE +-----------+ +-----------+

| Cube 1 |<------------------->| Cube 2 |<--...--->| Cube N |

+-----------+ +-----------+ +-----------+

|| || ||

|| || ||

|| BLE || ||

|| || ||

\-----------------------------------+---------------------------/

||

|| Matter

|| (over Wi-Fi or Ethernet)

||

+------------------+

| IoT Server |

+------------------+

+---------------------+

| ESP32-S3-WROOM-2 |

| Development Board |

+---------------------+

||

||

+-------------------+ || +-------------------+

| Gas Sensor (MQ-2) |<----ADC---||----ADC--->| Smoke Sensor (MQ-135) |

+-------------------+ || +-------------------+

||

+-------------------+ || +------------------+

| Motion Sensor |<---GPIO---||----GPIO--->| Temperature Sensor |

| (HC-SR501) | || | (DS18B20) |

+-------------------+ || +------------------+

||

+-------------------+ || +------------------+

| Light Sensor |<---I2C----||----SPI---->| 9-axis MEMS |

| (BH1750) | || | Sensor (MPU-9250)|

+-------------------+ || +------------------+

||

+-------------------+ || +------------------+

| Rotary Encoder |<---GPIO---||----GPIO--->| Buzzer module |

| Module (KY-040) | || | with amplifier |

+-------------------+ || | circuit (PAM8403)|

|| +------------------+

||

+-------------------------------+| +------------------+

| 3.5-inch ILI9488 TFT Display ||----SPI---->| SD Card |

| Module |+ +------------------+

+-------------------------------+

||

+-------------------------------+

| Lithium-Ion Polymer Battery |

| (3.7V 2500mAh) |

+-------------------------------+

## Interconnection Diagram

Diagram, schematic

Description automatically generated

Hardware Components:

1. ESP32-S3-WROOM-2 Development Board: The central component of the system, responsible for processing sensor data, managing inter-cube communication via BLE, and connecting to the IoT server through Matter.
2. Gas Sensor (MQ-2): Detects combustible gases, such as methane and propane, and sends data to the ESP32-S3-WROOM-2 for processing.
3. Smoke Sensor (MQ-135): Monitors air quality by detecting smoke and other harmful gases and sends data to the ESP32-S3-WROOM-2 for processing.
4. Motion Sensor (HC-SR501): Detects movement within the monitored area and sends data to the ESP32-S3-WROOM-2 for processing.
5. Light Sensor (BH1750): Measures ambient light levels and sends data to the ESP32-S3-WROOM-2 for processing.
6. Temperature Sensor (DS18B20): Monitors temperature changes and sends data to the ESP32-S3-WROOM-2 for processing.
7. 9-axis MEMS Sensor (MPU-9250): Measures acceleration, gyroscope, and magnetometer data and sends it to the ESP32-S3-WROOM-2 for processing.
8. Rotary Encoder Module (KY-040): Provides a user interface for setting timers and adjusting system settings, and sends user inputs to the ESP32-S3-WROOM-2 for processing.
9. Buzzer Module with Amplifier Circuit (PAM8403): Generates an audible alarm in case of emergencies or when the timer expires, controlled by the ESP32-S3-WROOM-2.
10. 2.5-inch OLED Display: Displays sensor data and system status, controlled by the ESP32-S3-WROOM-2.
11. SD Card: Stores sensor data, system logs, and configuration files.
12. Lithium-Ion Polymer Battery (3.7V 2500mAh): Provides power to the system, enabling a long battery life and rechargeability.
13. 5V/2A Power Adapter or USB Power Bank: Supplies power to the system, ensuring continuous operation.

Software Components:

1. Firmware: The firmware is written using the Arduino IDE and ESP32 core libraries, providing a modular and scalable architecture with separate modules or classes for each sensor and functionality.
2. Multi-threading or Task Scheduling: The firmware is designed to use multi-threading or task scheduling, allowing each sensor module to run independently and in parallel with other modules for efficient and responsive system operation.
3. Communication Protocols: The system utilizes BLE for inter-cube communication and Matter protocol for connecting to the IoT server.

Hardware

In this section, we will discuss the hardware components of the ESP32-based IoT Sensor Board for Home Safety and Monitoring with Display and Interconnected Cubes. The hardware components include:

1. ESP32 development board (ESP32-S3-WROOM-2): This board features the ESP32-S3R8V SoC, which includes a dual-core 32-bit LX7 microprocessor, up to 32 MB flash memory, up to 8 MB PSRAM, 33 GPIOs, and a rich set of peripherals. The ESP32 development board serves as the central processing unit for the entire system.
2. Gas sensor (MQ-2): This sensor detects various combustible gases and provides an analog output proportional to the gas concentration. It helps monitor the gas levels in the home and alerts the user in case of a gas leak.
3. Smoke sensor (MQ-135): This sensor detects smoke particles and provides an analog output proportional to the smoke concentration. It helps monitor the smoke levels in the home and alerts the user in case of a fire.
4. Motion sensor (HC-SR501): This PIR (Passive Infrared) sensor detects motion by sensing the infrared radiation emitted by humans or animals. It helps monitor the movement in the home and can be used for security purposes or energy-saving applications.
5. Light sensor (BH1750): This digital ambient light sensor provides a precise measurement of the light intensity in the environment. It can be used to monitor the ambient light levels and control lighting systems in the home.
6. Temperature sensor (DS18B20): This digital temperature sensor provides accurate temperature measurements with a resolution of up to 0.0625°C. It can be used to monitor the temperature in the home and control heating or cooling systems.
7. 9-axis MEMS sensor (MPU-9250): This sensor combines a 3-axis gyroscope, a 3-axis accelerometer, and a 3-axis magnetometer, providing comprehensive motion tracking data. It can be used for various applications, such as detecting falls or analyzing the user's movements.
8. Rotary encoder module (KY-040): This module allows the user to set the timer and interact with the system by rotating the knob and pressing it as a button.
9. Buzzer module with amplifier circuit (PAM8403): This module generates an audible alarm when the timer expires or in case of an emergency, such as a gas leak or fire.
10. 3.5-inch ILI9488 TFT display module: This display module provides a graphical interface for displaying sensor data and system information. Alternatively, you may use an OLED display if power constraints are a concern.
11. Lithium-Ion Polymer Battery (3.7V 2500mAh): This battery powers the system and provides a long-lasting and rechargeable power source.
12. 5V/2A power adapter or USB power bank: These power sources can be used to charge the lithium-ion polymer battery or power the system directly.

The hardware components are interconnected and communicate using various protocols such as I2C, SPI, ADC, and GPIO. ESP-NOW is used for BLE intercube communication, and the Matter protocol is used for communication with the IoT server.

To interface multiple sensors on the ESP32-S3-WROOM-2 chip, follow these steps:

a) Following protocols are used:

Gas sensor (MQ-2) and Smoke sensor (MQ-135): Analog (ADC)

Motion sensor (HC-SR501): Digital I/O

Light sensor (BH1750): I2C

Temperature sensor (DS18B20): OneWire

9-axis MEMS sensor (MPU-9250): SPI

Rotary encoder module (KY-040): Digital I/O

b) Assign appropriate GPIO pins for each sensor and peripheral. Make sure to avoid conflicts between pins used by different devices.

c) In the firmware code, initialize each sensor and peripheral using the appropriate library and GPIO assignments.

d) Implement a function to read data from each sensor and process it accordingly. This function should be called in the main loop or as part of a separate task or thread.

**Set up Matter and BLE Communication**

To set up Matter and BLE communication for the ESP32-S3-WROOM-2 chip, follow these steps:

a) Install the required Matter and ESP-NOW libraries as described in the "Set up Software Environment" section.

b) Initialize Matter and ESP-NOW communication in the setup function of your firmware code. This includes setting up the necessary pins, configuring the ESP-NOW as a master or slave device, and establishing a connection to the IoT server using Matter.

c) Implement functions for sending and receiving data using Matter and ESP-NOW. These functions should handle the encoding and decoding of data, as well as processing received data and triggering appropriate actions.

d) Integrate the Matter and ESP-NOW communication functions with the sensor data processing and event handling code. This allows the cubes to share sensor data and respond to events in the environment.

**Connecting Matter to Web Server and IoT Platforms**

To connect the Matter-enabled ESP32-based IoT Sensor Board to a web server and IoT platforms like Alexa, Apple HomeKit, Google Home, and IFTTT, follow these steps:

a) Set up a Matter-enabled IoT server, such as an MQTT broker, that can receive and process data from the cubes.

b) Configure your IoT server to work with your desired IoT platforms and services. This may require setting up authentication, connecting to APIs, or implementing custom code.

c) In the firmware code, implement functions to send sensor data and events to the IoT server using Matter. This may include converting sensor data into a standard format, encoding messages, and handling acknowledgements and errors.

d) Set up the IoT platforms and services to receive data from the IoT server and trigger actions based on the data. This may include setting up rules, routines, or custom code to process events and send notifications or control other devices.

By following these steps, you can create a robust, interconnected ESP32-based IoT Sensor Board system for Home Safety and Monitoring with Display and Interconnected Cubes that can communicate with various IoT platforms and services.

Programming Methods

The programming of the ESP32-based IoT Sensor Board for Home Safety and Monitoring with Display and Interconnected Cubes is done using the Arduino IDE and the ESP32 core libraries. The firmware code is written in a modular and scalable architecture, with separate modules or classes for each sensor and functionality. This allows for efficient and maintainable code development.

To program the ESP32 development board, connect it to your computer using a USB cable, and follow these steps:

1. Install the Arduino IDE: Download and install the Arduino IDE from the official website (<https://www.arduino.cc/en/software>).
2. Install the ESP32 core libraries: In the Arduino IDE, go to "File > Preferences" and add the following URL to the "Additional Boards Manager URLs" field: **https://dl.espressif.com/dl/package\_esp32\_index.json**. Then, go to "Tools > Board > Boards Manager" and search for "ESP32" and install the "esp32" package by Espressif Systems.
3. Install the required libraries for sensors and peripherals: Go to "Sketch > Include Library > Manage Libraries" and search for the following libraries and install them:

* MQUnifiedsensor
* Adafruit\_MPU9250
* OneWire
* DallasTemperature
* BH1750
* PAM8403
* ILI9488
* RotaryEncoder

1. Install Matter and BLE libraries: To install the necessary libraries for Matter and BLE communication, search for and install the following libraries:
2. Matter (formerly CHIP)
3. ESP-NOW
4. Write the firmware code: Create a new Arduino sketch and write the code for the ESP32-based IoT Sensor Board for Home Safety and Monitoring with Display and Interconnected Cubes. Make sure to include the necessary libraries, define the pins for each sensor and peripheral, and implement the required functions for reading sensor data, processing events, and communicating with the IoT server.
5. Upload the code to the ESP32 development board: Connect the ESP32 development board to your computer via a USB cable. In the Arduino IDE, go to "Tools > Board" and select the "ESP32-S3-WROOM-2" board. Then, go to "Tools > Port" and select the appropriate COM port for your board. Finally, click on the "Upload" button to compile and upload the code to the ESP32 development board.

After the code has been uploaded, the ESP32-based IoT Sensor Board for Home Safety and Monitoring with Display and Interconnected Cubes should start functioning as intended. The cubes will communicate with each other using ESP-NOW protocol and will communicate with the IoT server using the Matter protocol. The system will monitor the environment, process events, and send notifications to the user when necessary.

In the next sections, we will provide more detailed information on interfacing multiple sensors, setting up the Matter and BLE communication, and integrating the system with various IoT platforms and services.

create image of oled 1.8inch with this values: Time: 12:34 PM Temp: 25.0°C Gas: 123ppm Smoke: 234ppm Light: 345 Lux Motion: No

**Sensor values to display on OLED:**

* Gas level (MQ-2 sensor)
* Smoke level (MQ-135 sensor)
* Motion detection status (AM312 sensor)
* Light intensity (BH1750 sensor)
* Temperature (DS18B20 sensor)
* 9-axis MEMS sensor values (MPU-9250):
  + Accelerometer (X, Y, Z)
  + Gyroscope (X, Y, Z)
  + Magnetometer (X, Y, Z)

**Summary Mode (Screen 1) - Normal:** **Summary Mode (Screen 2)- Alert:**

**[Cube Icon] IoT Sensor Cube**

**12:34 PM**

**Wi-Fi: presume - connected.**

**Status: Smoke Alert – sent to cube2,3**

**Summary: T 25°C, G 123ppm**

**[>] More Details**

**[Cube Icon] IoT Sensor Cube**

**12:34 PM**

**Wi-Fi: presume - connected.**

**Status: Normal**

**Summary: T 25°C, G 123ppm**

**[>] More Details**

|  |  |  |
| --- | --- | --- |
| Gas Sensor Detailed Mode (Screen 2): | Smoke Sensor Detailed Mode (Screen 3): | Motion Sensor Detailed Mode (Screen 4): |
| **Gas Sensor (MQ-2)**  **--------------**  **Level: 123ppm**  **Alarm: 300ppm**  **Status: Normal** | **Smoke Sensor (MQ-135)**  **--------------**  **Level: 234ppm**  **Alarm: 400ppm**  **Status: Normal** | **Motion Sensor (AM312)**  **--------------**  **Status: No motion**  **Last Detected: 5 min ago** |
|  |  |  |
| Light Sensor Detailed Mode (Screen 5): | Temperature Sensor Mode (Screen 6): | 9-axis MEMS Sensor Mode (Screen 7): |
| **Light Sensor (BH1750)**  **--------------**  **Intensity: 345 Lux**  **Status: Bright** | **Temperature (DS18B20)**  **--------------**  **Value: 25.0°C**  **Alarm: 35.0°C**  **Status: Normal** | **MEMS Sensor (MPU-9250)**  **--------------**  **Accel: X 0.01 Y -0.02 Z 0.98**  **Gyro: X 0.05 Y 0.04 Z -0.03**  **Mag: X 25.0 Y -30.0 Z 45.0** |

1. Rotary encoder: Use the rotary encoder for scrolling through the different screens, menu items, or values. Rotating the encoder clockwise moves forward or increases the value, while rotating counterclockwise moves backward or decreases the value.
2. Touch Button 1 (Select): This button acts as the "Enter" or "Select" key. Pressing this button confirms a selection, enters a submenu, or toggles a particular setting.
3. Touch Button 2 (Back): This button is used to navigate back to the previous screen or to exit a submenu.
4. Touch Button 3 (Home): Pressing this button returns the user to the home page, regardless of their current position in the menu system.

**Summary Mode (Screen 2)- Alert**:

On the OLED screen, display alerts that are crucial for the user's safety and device functionality. Some examples of alerts include:

1. Battery: Show a low battery alert when the battery level falls below a certain threshold. This can be indicated with a battery icon and a percentage or a "Low Battery" message.
2. Smoke: Display a smoke alert when the smoke sensor (MQ-135) detects a high concentration of smoke, which may indicate a fire. This alert can include a warning icon and a message like "High Smoke Detected."
3. Gas: Show a gas leak alert when the gas sensor (MQ-2) detects a high concentration of gas, indicating a potential leak. This can be displayed with a gas icon and a message like "Gas Leak Detected."
4. Motion: Display a motion alert when the motion sensor (AM312) detects movement. This can include a motion icon and a message like "Motion Detected."
5. Temperature: Show a high or low temperature alert when the temperature sensor (DS18B20) detects temperatures outside a specified range. This can be displayed with a temperature icon and a message like "High Temperature" or "Low Temperature."
6. Light: Display a low light alert when the light sensor (BH1750) detects insufficient light levels. This can include a light bulb icon and a message like "Low Light Detected."
7. Connectivity: Show a connectivity alert when the device loses connection to the Blynk server or the Matter-enabled IoT server. This can be displayed with a Wi-Fi or connectivity icon and a message like "Connection Lost."

**Set Timer**

The timer is activated whenever the rotary knob is rotated. The OLED display will show the "Set Timer:" screen as soon as the user starts rotating the knob. Here's how the flow would look like:

|  |  |  |
| --- | --- | --- |
| Gas Sensor Detailed Mode (Screen 2): |  |  |
| **Set Timer:**    **05:00**    **(Press Btn)** | **Timer Running:**    **04:59**    **(Press Btn)** | **Timer Paused:**    **04:42**    **(Press Btn)** |

* 1. **Activate Timer:** When the user starts rotating the rotary encoder, the OLED display will immediately show the "Set Timer:" screen with the current timer value.
  2. **Adjust Timer:** Rotate the encoder clockwise to increase the timer value and counterclockwise to decrease the timer value.
  3. **Start Timer:** Press the rotary encoder button once to start the timer. The timer will start counting down, and the OLED will update with the remaining time.
  4. **Pause Timer:** While the timer is running, press the rotary encoder button once to pause the timer. The OLED will show the remaining time, and the timer will be paused.
  5. **Resume Timer:** While the timer is paused, press the rotary encoder button once to resume the timer. The timer will continue counting down from the paused time.
  6. **Stop Timer:** While the timer is running or paused, press, and hold the rotary encoder button for 2 seconds to stop and reset the timer. The timer will reset to its initial value. After stop time, the screen should return back to home screen.

| **Component** | **Power Consumption** |
| --- | --- |
| ESP32-S3-WROOM-2 | 150-260 mA |
| MQ-2 Gas Sensor | 800 mW |
| MQ-135 Smoke Sensor | 800 mW |
| AM312 PIR Motion Sensor | 60-80 μA |
| BH1750 Light Sensor | 0.12 mA |
| DS18B20 Temperature Sensor | 1 mA |
| MPU-9250 9-axis MEMS Sensor | 3.5 mA |
| 2.5-inch OLED Display | 30-50 mA |

Sensor Json data:

|  |
| --- |
| Sensor Json data for post processing |
| [  {  "timestamp": "2023-03-15T12:34:56Z",  "gas": {  "MQ-2": 123.45  },  "smoke": {  "MQ-135": 234.56  },  "motion": {  "AM312": 1  },  "light": {  "BH1750": 345.67  },  "temperature": {  "DS18B20": 25.00  },  "MEMS": {  "MPU-9250": {  "accelerometer": {  "x": 0.01,  "y": -0.02,  "z": 0.98  },  "gyroscope": {  "x": 0.05,  "y": 0.04,  "z": -0.03  },  "magnetometer": {  "x": 25.00,  "y": -30.00,  "z": 45.00  }  }  }  },  ...  ] |

Reference:

|  |
| --- |
|  |
| # Load the JSON data  with open('sensor\_data.json', 'r') as f:  data = json.load(f)  # Preprocess the data and convert it into a pandas DataFrame  sensor\_data = []  for entry in data:  sensor\_values = {  'timestamp': entry['timestamp'],  'MQ-2': entry['gas']['MQ-2'],  'MQ-135': entry['smoke']['MQ-135'],  'AM312': entry['motion']['AM312'],  'BH1750': entry['light']['BH1750'],  'DS18B20': entry['temperature']['DS18B20'],  'MPU-9250\_acc\_x': entry['MEMS']['MPU-9250']['accelerometer']['x'],  'MPU-9250\_acc\_y': entry['MEMS']['MPU-9250']['accelerometer']['y'],  'MPU-9250\_acc\_z': entry['MEMS']['MPU-9250']['accelerometer']['z'],  'MPU-9250\_gyro\_x': entry['MEMS']['MPU-9250']['gyroscope']['x'],  'MPU-9250\_gyro\_y': entry['MEMS']['MPU-9250']['gyroscope']['y'],  'MPU-9250\_gyro\_z': entry['MEMS']['MPU-9250']['gyroscope']['z'],  'MPU-9250\_mag\_x': entry['MEMS']['MPU-9250']['magnetometer']['x'],  'MPU-9250\_mag\_y': entry['MEMS']['MPU-9250']['magnetometer']['y'],  'MPU-9250\_mag\_z': entry['MEMS']['MPU-9250']['magnetometer']['z']  }  sensor\_data.append(sensor\_values)  df = pd.DataFrame(sensor\_data)  print(df.head()) |

**Following is the guidance for using the Blynk IoT server for communicating with your cubes via Matter protocol:**

1. Sign up for a Blynk account: Head over to the Blynk website (<https://blynk.io/>) and sign up for a free account. Afterward, download the Blynk app onto your smartphone or tablet from the App Store or Google Play Store.
2. Create a new Blynk project: Open the Blynk app, create a new project, and select the ESP32 as your device. You'll receive an authentication token via email, which is necessary for connecting your cubes to the Blynk server.
3. Install the Blynk library for ESP32: In the Arduino IDE, navigate to Sketch -> Include Library -> Manage Libraries. Search for "Blynk" and install the latest version of the Blynk library for ESP32.
4. Install the Matter SDK: Follow the official guidelines for installing the Matter SDK and the required tools for your ESP32 development environment (<https://github.com/project-chip/connectedhomeip>).
5. Integrate Blynk and Matter into your firmware: Include the Blynk and Matter libraries in your Arduino sketch, and use the authentication token you received earlier to connect your cubes to the Blynk server. Configure your Matter-enabled ESP32 device to work with Blynk by bridging the communication between the Matter protocol and the Blynk server.
6. Configure your Blynk project: In the Blynk app, add widgets to control and monitor the various sensors and devices connected to your cubes. Associate the widgets with the virtual pins corresponding to the sensor data or device control in your firmware.
7. Implement Blynk callbacks in the firmware: Implement Blynk event handlers (callbacks) in your Arduino sketch that respond to events triggered by the Blynk widgets. For example, when a button is pressed in the Blynk app, the corresponding event handler in your firmware should process the event and perform the desired action.
8. Flash the firmware to your cubes: Upload the updated firmware to your ESP32-based IoT Sensor Board with Display and Interconnected Cubes.
9. Test the Blynk-Matter integration: Power on your cubes and ensure that they connect to the Blynk server and the Matter-enabled IoT server. Test the Blynk widgets in your app to ensure that they correctly control and monitor the devices and sensors connected to your cubes.

**Required Libraries**: Use the Arudino Library Manager (Sketch > Include Library > Manage Libraries) to search for and install the following libraries:

* Adafruit ILI9488 (for 3.5-inch ILI9488 TFT display module) or Adafruit SSD1306 (for OLED display)
* Adafruit GFX Library (for display graphics)
* Adafruit Unified Sensor (for MPU-9250)
* Adafruit MPU9250 (for MPU-9250)
* OneWire (for DS18B20 temperature sensor)
* DallasTemperature (for DS18B20 temperature sensor)
* BH1750 (for BH1750 light sensor)
* PubSubClient (for MQTT communication) in our case Blynk and Matter.

**Motion Sensor (AM312)**

**--------------**

**Status: No motion**

**Last Detected: 5 min ago**

**Smoke Sensor (MQ-135)**

**--------------**

**Level: 234ppm**

**Alarm: 400ppm**

**Status: Normal**

**Gas Sensor (MQ-2)**

**--------------**

**Level: 123ppm**

**Alarm: 300ppm**

**Status: Normal**

**MEMS Sensor (MPU-9250)**

**--------------**

**Accel: X 0.01 Y -0.02 Z 0.98**

**Gyro: X 0.05 Y 0.04 Z -0.03**

**Mag: X 25.0 Y -30.0 Z 45.0**

**Light Sensor (BH1750)**

**--------------**

**Intensity: 345 Lux**

**Status: Bright**

**Temperature (DS18B20)**

**--------------**

**Value: 25.0°C**

**Alarm: 35.0°C**

**Status: Normal**