HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING

BIG PROJECT

Digital System Design II

Project Name:

Fire Detection System Using ESP32 Microcontroller

Students: Đỗ Văn Đoàn (20213607)
Nguyễn Việt Hùng (20213614)
Nguyễn Song Nam (20210621)
Class: CTTT Điện tử 01 - K66
Instructor: Dr. Ngô Vũ Đức

Table of Contents

Table of Figures

I. Introduction

In recent years, Vietnam has seen an increase in serious fires in multi-storey buildings, making fire safety more important than ever. To help address this, we have created a modern fire detection system using the ESP32 microcontroller. This system is designed to keep multi-storey buildings safe by detecting fires early and responding quickly.

Our system uses the ESP32 microcontroller, which is powerful and reliable. It connects to three types of sensors: smoke sensors, temperature sensors, and fire sensors. These sensors work together to detect any signs of a fire as soon as possible.

When the system detects smoke, a sudden rise in temperature, or flames, the ESP32 sends signals to sound alarms and activate an automatic water spraying system. This quick response helps alert everyone in the building and starts fighting the fire immediately, reducing damage and preventing the fire from spreading.

Figure 1. Need of fire alarm nowadays

II. Overall Design Of Fire Detection System

1. Device modules used in modeling

- ESP32 microcontroller
- Arduino Uno R3 board
- Module RF Zigbee CC2530
- Temperature and Humidity sensor DHT11
- Smoke sensor MQ-3
- Flame sensor
- Fire alarm
- Leds present for water injection system

2. Schematic

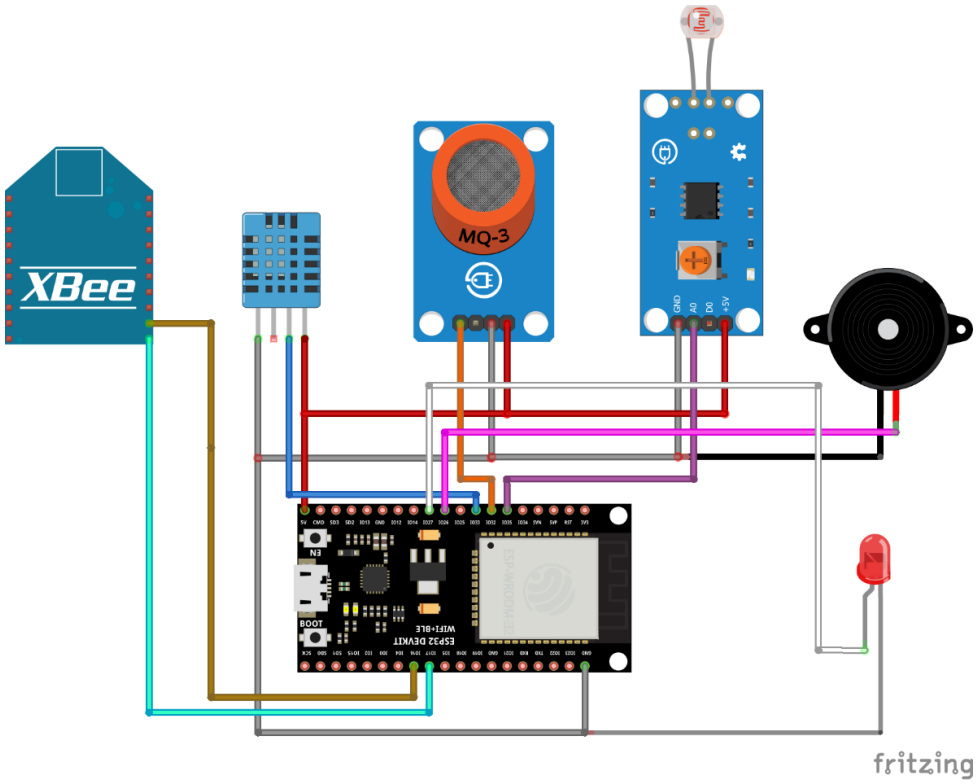

Figure 2. Schematic of Floor 1

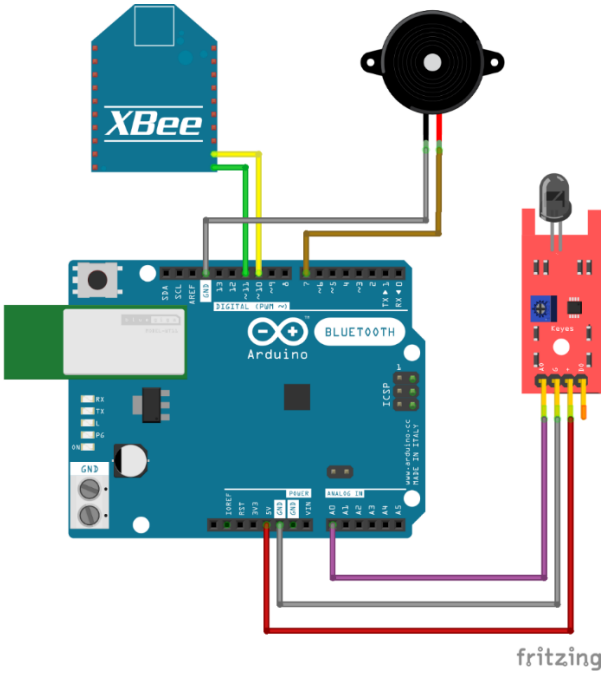

Figure 3. Schematic of Floor 2,3

3. Introduction of the ESP32 and Arduino module

- ESP32 Microcontroller:

- ESP32 is a system on a chip (SoC) microcontroller unit (MCU) designed and manufactured by Espressif Systems. It is a popular choice for IoT applications due to its low cost, low power consumption, and integrated Wi-Fi and Bluetooth capabilities.
- ESP32 modules are typically small in size and can be easily integrated into IoT devices. They are also relatively inexpensive, making them a good choice for budget-conscious projects.
- Some of the key features of ESP32 modules that make them well-suited for IoT applications include:
    - Low power consumption: ESP32 modules have a number of features that help to reduce power consumption, such as deep sleep mode and ultra low power mode. This makes them ideal for battery-powered IoT devices.
    - Integrated Wi-Fi and Bluetooth: ESP32 modules have built-in Wi-Fi and Bluetooth capabilities, which eliminates the need for external modules. This simplifies the design of IoT devices and reduces the overall cost
    - Powerful processing: ESP32 modules have a dual-core processor that provides ample processing power for even the most demanding IoT applications.
    - Rich peripheral set: ESP32 modules have a wide range of peripherals, including GPIO pins, ADCs, DACs, and I2C/UART interfaces. This makes them easy to connect to a variety of sensors and actuators.

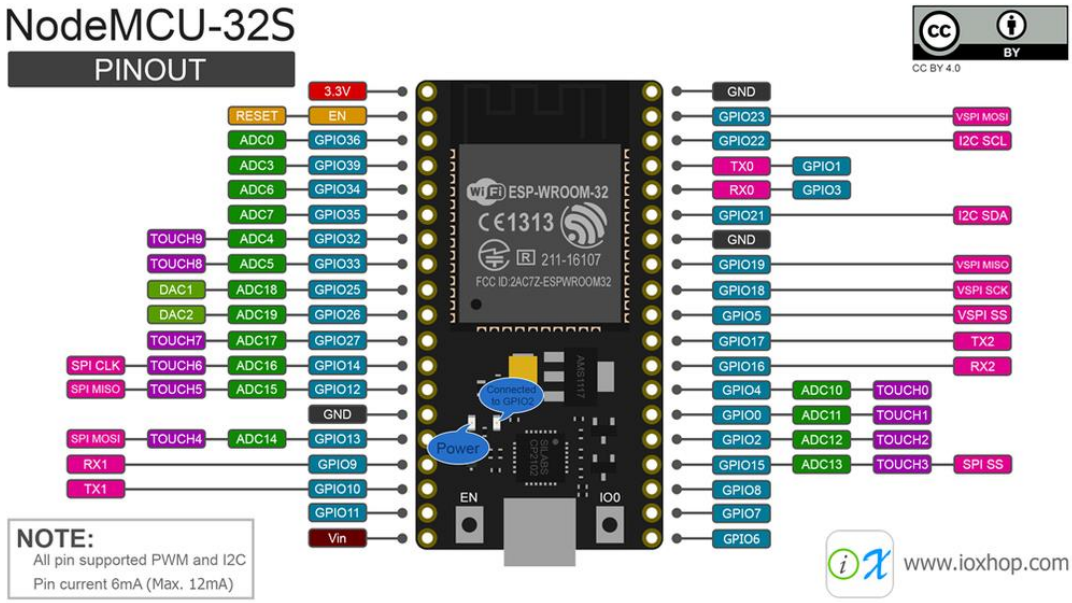

Figure 4. ESP32-DevKit Pin Layout

- Arduino Uno R3

- The Arduino Uno is a microcontroller board based on the Atmega328 chip. It shrinks 14 pins in and out by digital signal, of which 6 pins can be used for pulse width modulation. There are 6 pins of analog signal input that allow us to connect to external sensors to collect data, using a 16MHz oscillating frequency quartz oscillator, there is a USB standard connector for us to load the program into the board and a pin to power the circuit, an ICSP header, a reset button. It contains everything needed to support microcontrollers, power supply for the Arduino can be from a computer via USB or from a dedicated power supply converted from AC to DC or power taken from a battery.

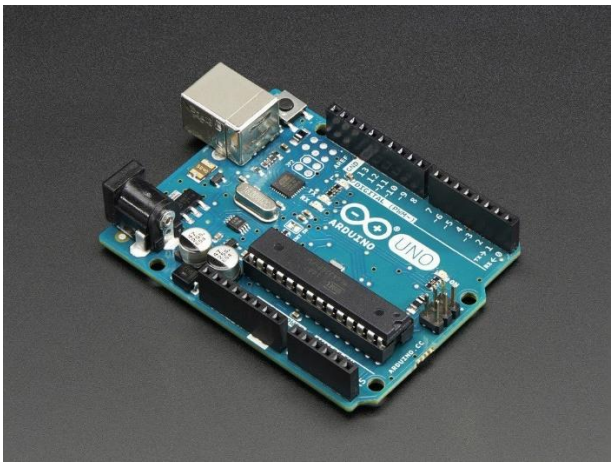

Figure 5. The hardware structure of Arduino Uno R3

III. Finite State Machine of Fire Dectection System

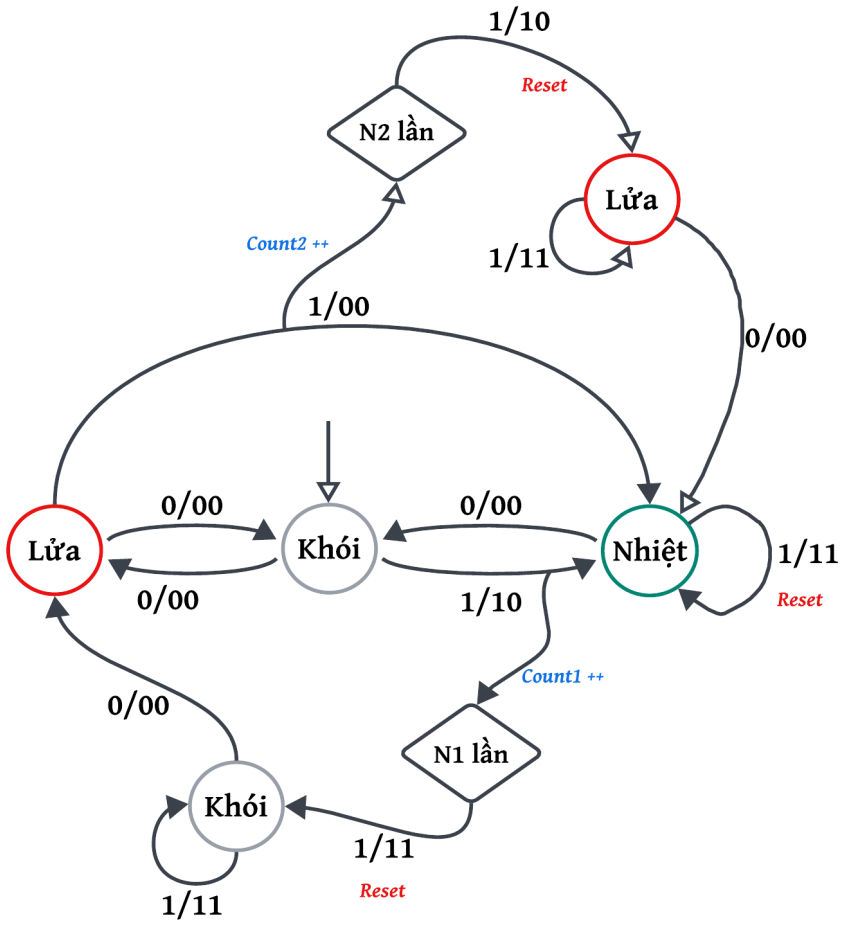

Figure 6. Finite State Machine

- Consider 3 main fire situations

- Burning flame:
  - The fire was small at first
  - No smoke detection capability
  - The temperature is not high enough for the system to detect
- Fire in cold room:
  - The temperature is not high enough for the system to detect in cold environments
  - Fire is difficult to detect if it is a case of component fire
- Gas tank explodes:
  - In this case the temperature increases

IV. Design Zigbee Network to Tranfer Data

1. Zigbee Network Overview

What is Zigbee?

- As the Internet of Things (IoT) industry develops, more and more wireless technologies come out.

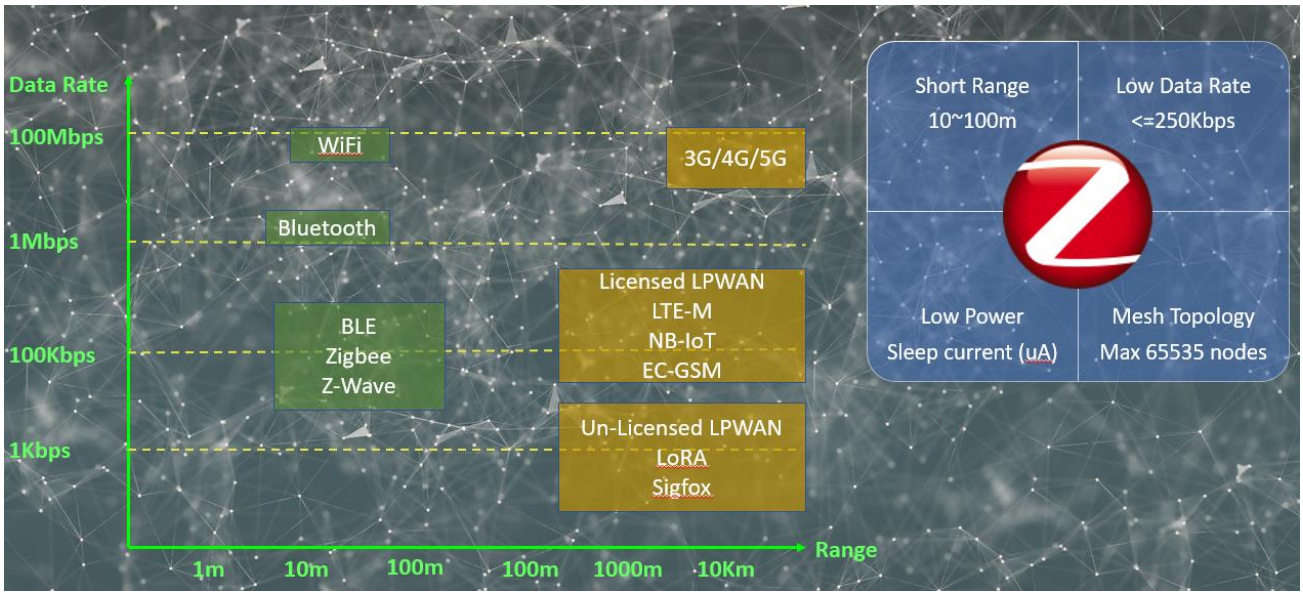

Figure 7. Zigbee

- As we know, in IoT industry, we have two typical networks. One is WAN (Wide Area Network), the other is PAN (Personal Area Network).

- For the wireless technologies like LoRa, NB-IoT, 2G/3G/4G, etc, normally the transmit distance more than 1 km, so they are mainly used in wide area network (WAN).
- For the wireless technologies like WiFi, Bluetooth, BLE, Zigbee and Zwave, normally the transmit distance is less than 1 km, so they are mainly used in personal area network (PAN).

- Zigbee is one of the most popular wireless technologies used in IoT networks, especially in home automation industry. Its characteristics includes:

- Short range – Normally the radio can conver from 10 to 100 meters.
- Low data rate – the maximum data rate is 250 Kbps.
- Low Power – a sleepy end device can use less than 5uA at sleep mode.
- It's a mesh technology – the network can be easily extended to very large. Theoretically maximum nodes number is 65535.

2. Protocol Overview

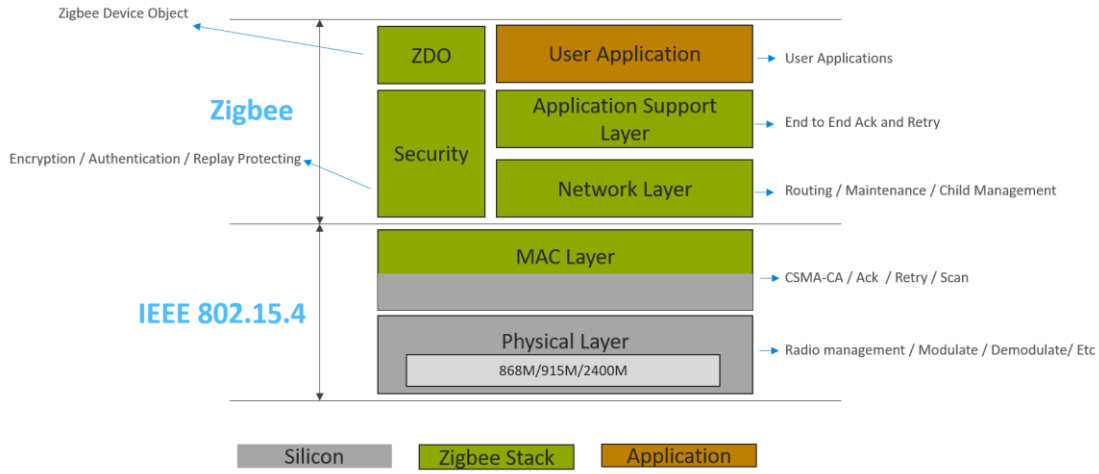

Figure 8. Protocol Overview

- Physical layer and MAC layer are defined by IEEE-802.15.4. The physical layer is responsible for radio management, including functions like modulating/ demodulating, signal strength detecting, etc. MAC layer is in charge of one-hop communication.
- The network layer is responsible for message transmitting and receiving, device mantainance, routing, etc.
- Application Support Layer (APS) is in charge of end to end message transmitting.
- Application layer is left for user design. Each application instance is called an endpoint. A special endpoint, which is endpoint 0, is reserved for management functionalities. We also call this management functionalities model as Zigbee Device Object (ZDO).
- In APS layer and network layer, there are some security features which are used to protect the network from being hacked.

3. Overview of Zigbee CC2530

- The Zigbee UART CC2530 V1 RF transceiver circuit uses the CC2530 IC from TI, the circuit is pre-programmed with firmware so that it can be easily used as a Zigbee standard wireless data transmission module with UART communication that is easy to connect to the microcontroller or computer (via USB-UART cable) with just a few configurable steps with the push of a button.

- Zigbee UART CC2530 V1 RF transceiver circuit has a long transmission distance, industrial standard Zigbee 2.4Ghz wave transmission standard is very stable and configurable

to form a wireless transmission network with many nodes and network points via Zigbee protocol.

Figure 9. Zigbee UART CC2530 V1 RF

- Specifications:

- Zigbee UART CC2530 V1 RF transceiver circuit.
- RF Zigbee SoC CC2530 main IC from TI.
- Usage voltage: 3 - 5.5VDC
- Current consumption: < 30mA
- Zigbee 2.4Ghz transmission standard.
- Maximum wave transmission rate 3300bps.
- Transmission Power: 4.5dbm
- Ideal transmission distance: 250m.
- UART TTL connection protocol (3.3VDC or 5VDC), Baudrate up to 115200.
- Size: 15.5x31.5mm.

4. Zigbee network in Fire Dectection System

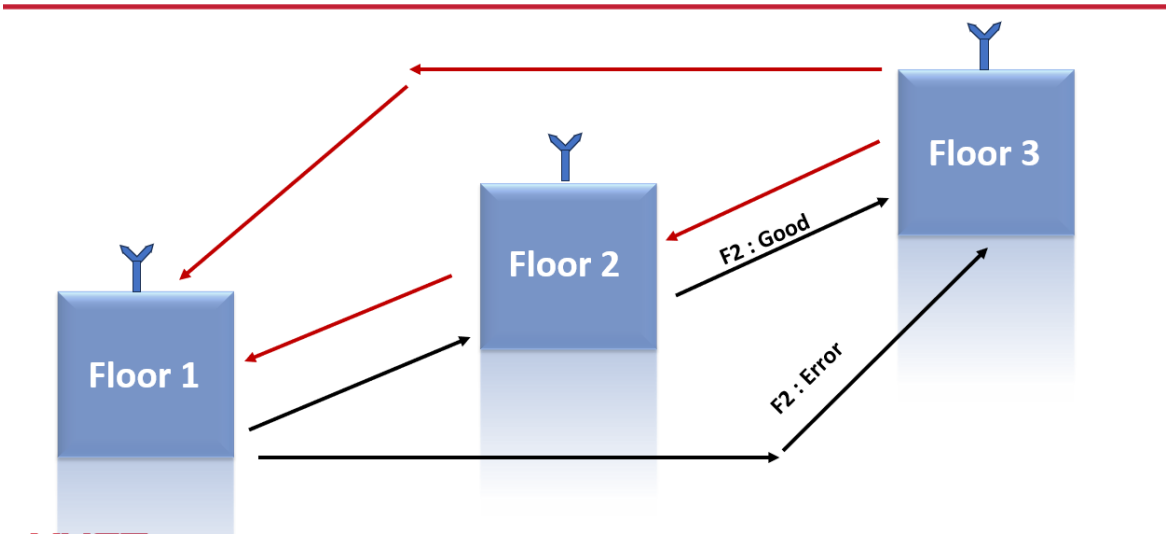

Figure 10. Zigbee Network

```
void TruyenThongBaoChay(){

  // gửi dữ liệu đến tầng 2
  String send_data = data + "*";
  ZigbeeSerial.print(send_data);
  Serial.println("Message sent to floor 2");

  // Đợi phản hồi từ tầng 2
  unsigned long startTime = millis();
  bool receiverAck =  false ;

  while((millis() - startTime)< 5000 ){ // wate 5 seconds
    if (ZigbeeSerial.available() > 0) {
      String Ack = ZigbeeSerial.readStringUntil('*');
      if( Ack == "Ack"){
        receiverAck = true;
        break;
      }
    }
  }


  if (!receiverAck){
    // Nếu không nhận được ACK từ tầng 2 , gửi trực tiếp đến tầng 3
    Serial.println ("No Ack from floor , sending to floor 3");
    ZigbeeSerial.print(send_data);
  }
  delay(100);
}


void NhanThongBaoChay() {
  String i;
  i = ZigbeeSerial.readStringUntil('*'); //  i="From_2_To_1" or "From_3_To_1";
  if((i == "From_2")||(i == "From_3")){
    count=0;
    Serial.println(i);
    Serial.println(" Co chay nhe  ");
    baochuong();
  }
  delay(100);
```

Figure 11. Floor 1

```
void TruyenThongBaoChay_For_From2() {
    String send_data = data_2 + "*";
    ZigbeeSerial.print(send_data);
}

void NhanThongBaoChay() {
    if (ZigbeeSerial.available() > 0) {
        // Đọc dữ liệu từ tầng 1
        String data = ZigbeeSerial.readStringUntil('*');
        Serial.print("Co_chay ");
        Serial.println(data);

        // Gửi ACk tới
        if (data == "From_1") {
            ZigbeeSerial.print("Ack");
            delay(3000);
            // Gửi dữ liệu đi
            ZigbeeSerial.print(data);
            batchuong();
        }
        if (data == "From_3") {
            ZigbeeSerial.print("Ack");
            delay(3000);
            // Gửi dữ liệu đi
            ZigbeeSerial.print(data);
            batchuong();
        }
    }else{
      offdevice();
    }
    delay(100);
}
```

Figure 12. Floor 2

V. Optimizing energy for ESP32 in Fire Dectection System

1. Deep sleep power mode in ESP32

- The ESP32 can switch between different power modes:

  - Active mode
  - Modem Sleep mode
  - Light Sleep mode
  - Deep Sleep mode
  - Hibernation mode

Power mode	Active	Modem-sleep	Light-sleep	Deep-sleep	Hibernation
Sleep pattern	Association sleep pattern			ULP sensor-monitored pattern	-
CPU	ON	ON	PAUSE	OFF	OFF
Wi-Fi/BT baseband and radio	ON	OFF	OFF	OFF	OFF
RTC memory and RTC peripherals	ON	ON	ON	ON	OFF
ULP co-processor	ON	ON	ON	ON/OFF	OFF

Figure 13. Power modes of ESP32

Power mode	Description	Power consumption
Active (RF working)	Wi-Fi Tx packet 14 dBm ~ 19.5 dBm	Please refer to Table 10 for details.
	Wi-Fi / BT Tx packet 0 dBm	
	Wi-Fi / BT Rx and listening	
Modem-sleep	The CPU is powered on.	Max speed 240 MHz: 30 mA ~ 50 mA
		Normal speed 80 MHz: 20 mA ~ 25 mA
		Slow speed 2 MHz: 2 mA ~ 4 mA
Light-sleep	-	0.8 mA
Deep-sleep	The ULP co-processor is powered on.	150 μA
	ULP sensor-monitored pattern	100 μA @1% duty
	RTC timer + RTC memory	10 μA
Hibernation	RTC timer only	5 μA
Power off	CHIP_PU is set to low level, the chip is powered off	0.1 μA

Figure 14. Power consumption in each power mode

Mode	Min	Typ	Max	Unit
Transmit 802.11b, DSSS 1 Mbps, POUT = +19.5 dBm	-	240	-	mA
Transmit 802.11b, OFDM 54 Mbps, POUT = +16 dBm	-	190	-	mA
Transmit 802.11g, OFDM MCS7, POUT = +14 dBm	-	180	-	mA
Receive 802.11b/g/n	-	95 ~ 100	-	mA
Transmit BT/BLE, POUT = 0 dBm	-	130	-	mA
Receive BT/BLE	-	95 ~ 100	-	mA

Figure 15. RF Power Consumption Specifications

Why choose Deep Sleep mode for this Fire Detection System?

- Having your ESP32 running on active mode with batteries it's not ideal, since the power from batteries will drain very quickly.

- If you put your ESP32 in deep sleep mode, it will reduce the power consumption and your batteries will last longer.

- While the ESP32 is in deep sleep mode the RTC memory also remains powered on, so we can write a program for the ULP co-processor and store it in the RTC memory to access peripheral devices, internal timers, and internal sensors.

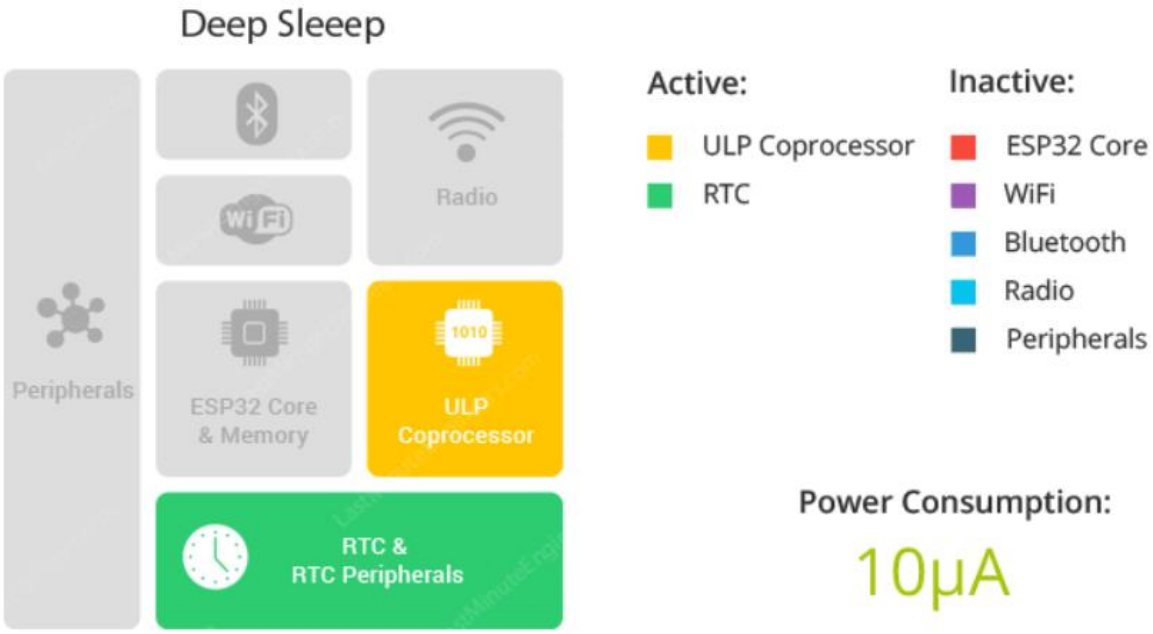

Figure 16. Deep sleep's active components

- With the ESP32, you can save data on the RTC memories. The ESP32 has 8kB SRAM on the RTC part, called RTC fast memory. The data saved here is not erased during deep sleep. However, it is erased when you press the reset button

- ESP32 Deep Sleep Wake-up sources:

  - Timer
  - Touch pad
  - External wakeup(ext0 & ext1)

  => In this Fire Detection System, Timer wake-up source is suitable for ESP32 Deep Sleep mode.

2. Deep Sleep with Timer Wake-up source in Fire Detection System

- The ESP32 can go into deep sleep mode, and then wake up at predefined periods of time. This feature is specially useful if you are running projects that require time stamping or daily tasks, while maintaining low power consumption.

- In this system, ESP32 starts **going to sleep and wake up after 5 seconds**, then if none of the smoke, temperature and fire values reach the dangerous threshold (for example, smoke<3500, temperature>33.5, fire<3200) within an amount of time then ESP32 will continue going to sleep.

- But if there is any of 3 values reach the threshold, or there is fire alarms from floor 2 and floor 3, the system will immediately jump into loop for checking fire possibilities!

```
17:00:50.382 -> □1□"□□z□"H□ A□ESP32 Awake!
17:00:53.947 -> Gia tri Khoi: 4095
17:00:55.118 -> Gia tri khoi: 4095
17:00:55.118 -> Giá trị bộ đếm khói: 0
17:00:57.223 -> Gia tri lua: 4095
17:00:59.313 -> Gia tri khoi: 4095
17:00:59.346 -> Giá trị bộ đếm khói: 0
17:01:01.397 -> Gia tri lua: 4095
17:01:03.533 -> Gia tri khoi: 4095
17:01:03.533 -> Giá trị bộ đếm khói: 0
17:01:05.632 -> Gia tri lua: 4095
17:01:07.707 -> Gia tri khoi: 4095
17:01:07.739 -> Giá trị bộ đếm khói: 0
17:01:09.813 -> Gia tri lua: 4095
17:01:11.914 -> Gia tri khoi: 4095
17:01:11.945 -> Giá trị bộ đếm khói: 0
17:01:11.977 -> Going to sleep now
17:01:16.974 ->         DdOW�ql�□���□ �ESP32 Awake!
17:01:20.570 -> Gia tri Khoi: 4095
17:01:21.744 -> Gia tri khoi: 4095
```

Figure 17. System in deep sleep power mode

3. Using watchdog timer of ESP32 in Fire Detection System

- A watchdog timer (WDT) is a timer that can be used to automatically detect software anomalies and automatically generates a system reset if the main program neglects to periodically service it.

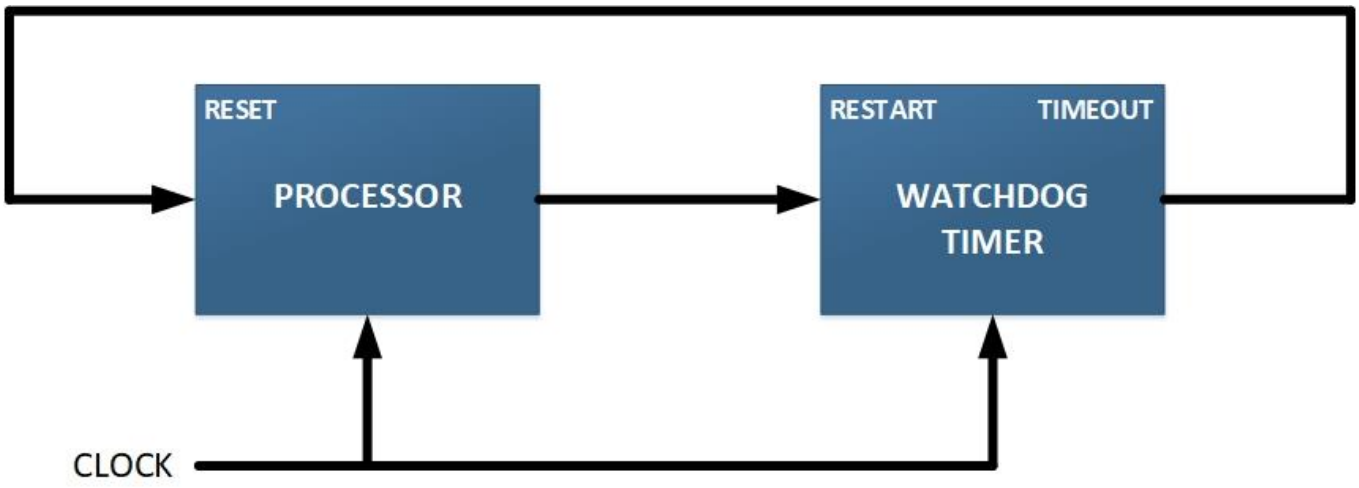

Figure 18. Typical Watchdog setup

- In our system, we use watchdog timer to prevent hang event may causes when the system has worked for a long time.

```
23:51:41.969 -> Gia tri lua: 1188 - Bộ đếm lửa: 0
23:51:46.969 -> E (22513) task_wdt: Task watchdog got triggered. The following tasks/users did not reset the watchdog in time:
23:51:47.095 -> E (22513) task_wdt:  - loopTask (CPU 1)
23:51:47.127 -> E (22513) task_wdt: Tasks currently running:
23:51:47.192 -> E (22513) task_wdt: CPU 0: IDLE0
23:51:47.223 -> E (22513) task_wdt: CPU 1: loopTask
23:51:47.256 -> E (22513) task_wdt: Aborting.
23:51:47.288 -> E (22513) task_wdt: Print CPU 1 backtrace
23:51:47.351 ->
23:51:47.351 ->
23:51:47.351 ->
23:51:47.351 ->
23:51:47.351 -> Backtrace: 0x400d3b95:0x3ffb2200 0x400d39e4:0x3ffb2220 0x400d3c86:0x3ffb2240 0x400db4e8:0x3ffb2270 0x4008fd96:0x3ffb2290
23:51:47.481 ->
23:51:47.481 ->
23:51:47.481 ->
23:51:47.481 ->
23:51:47.481 -> ELF file SHA256: 1e47eb1e48b53e3d
```

Figure 19. System with watchdog timer

VI. Hardware Model, Testing cases and Displaying Results

1. Hardware Model

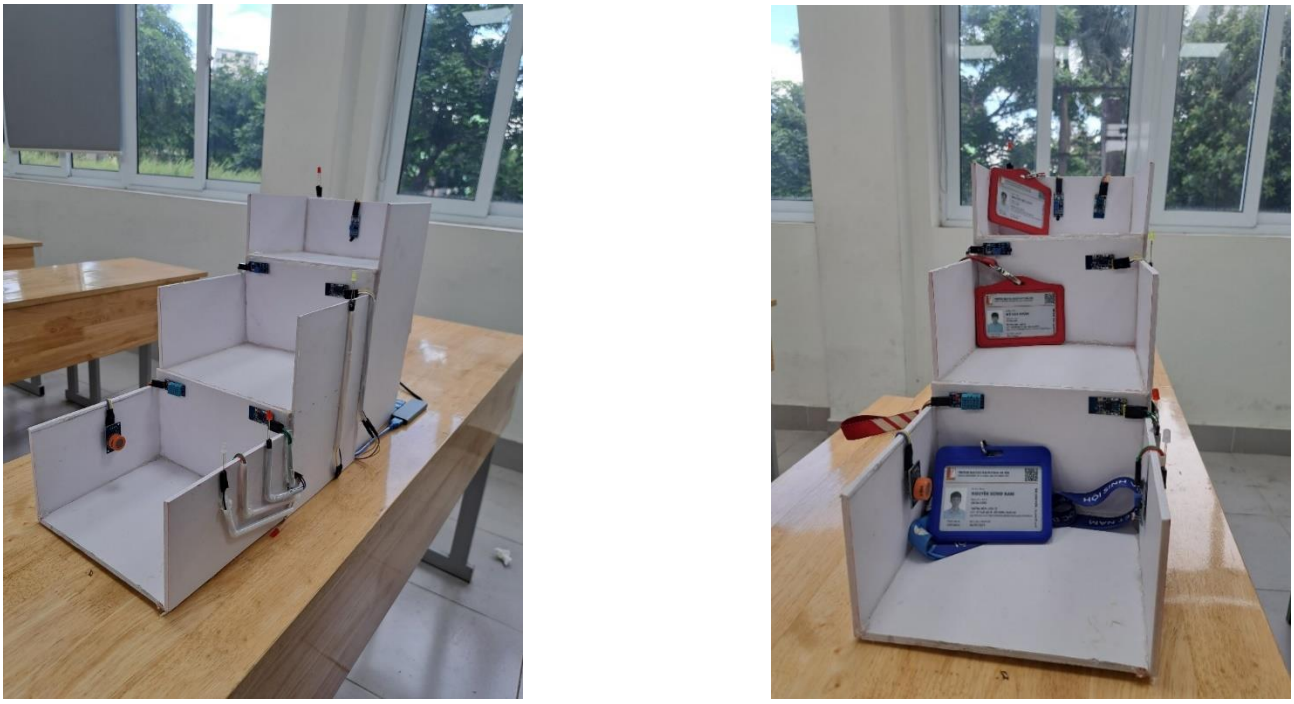

Figure 20. Hardware Model

2. Testing cases

- Case 1: There is fire in floor 1

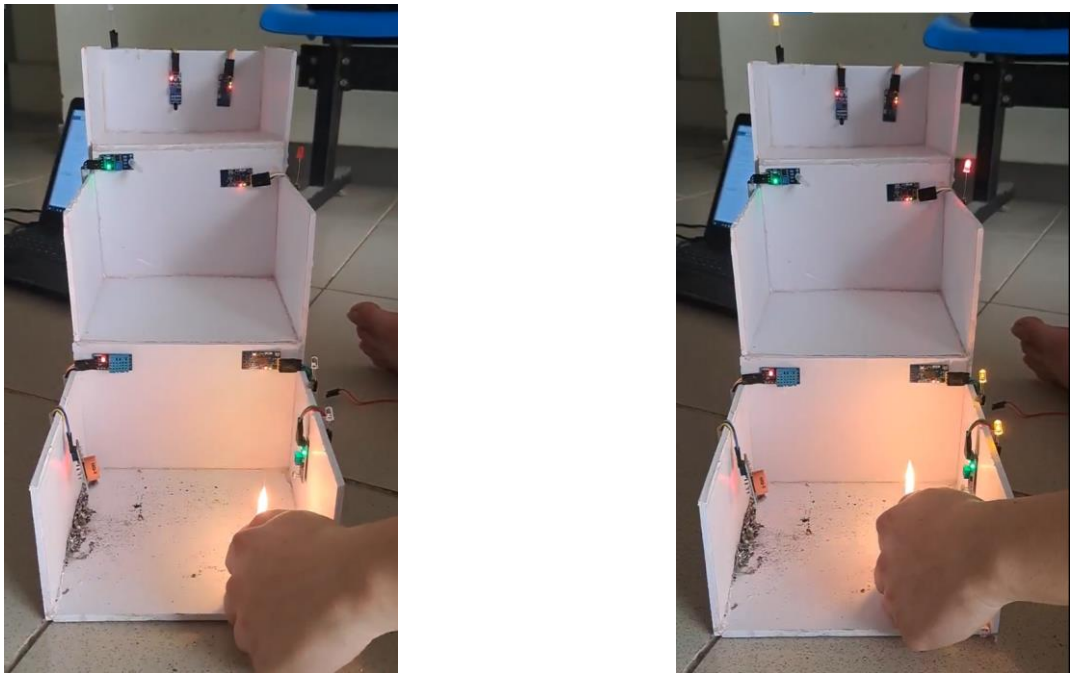

Figure 21. Fire in floor 1

- Case 2: There is fire in floor 2

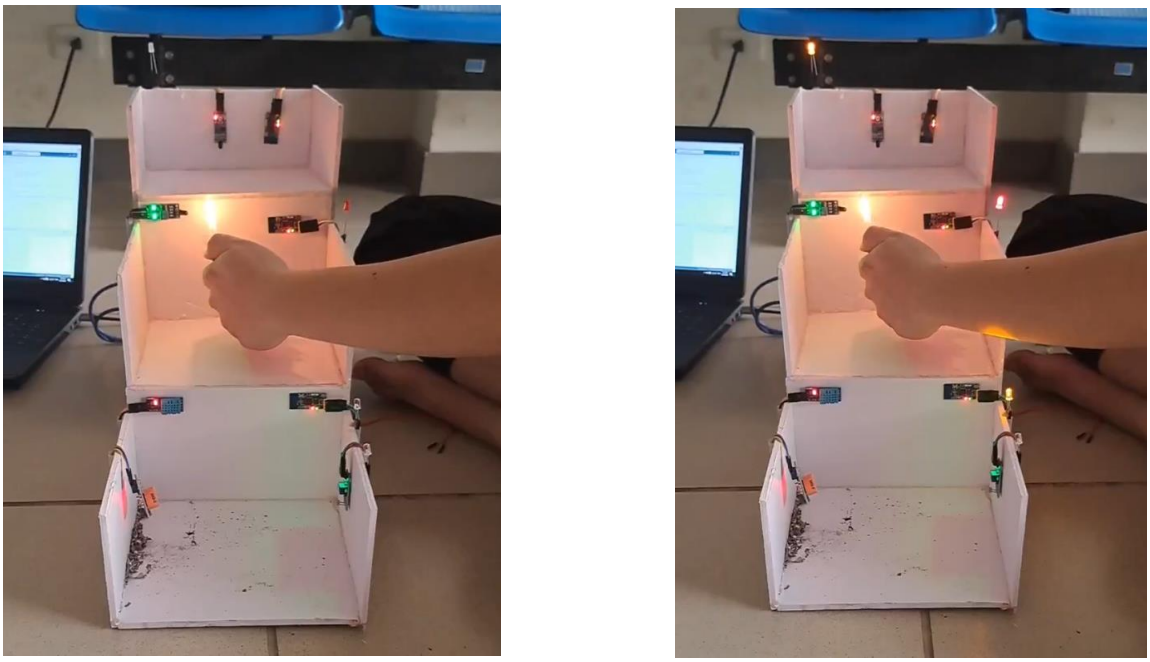

Figure 22. Fire in floor 2

- Case 3: There is fire in floor 3

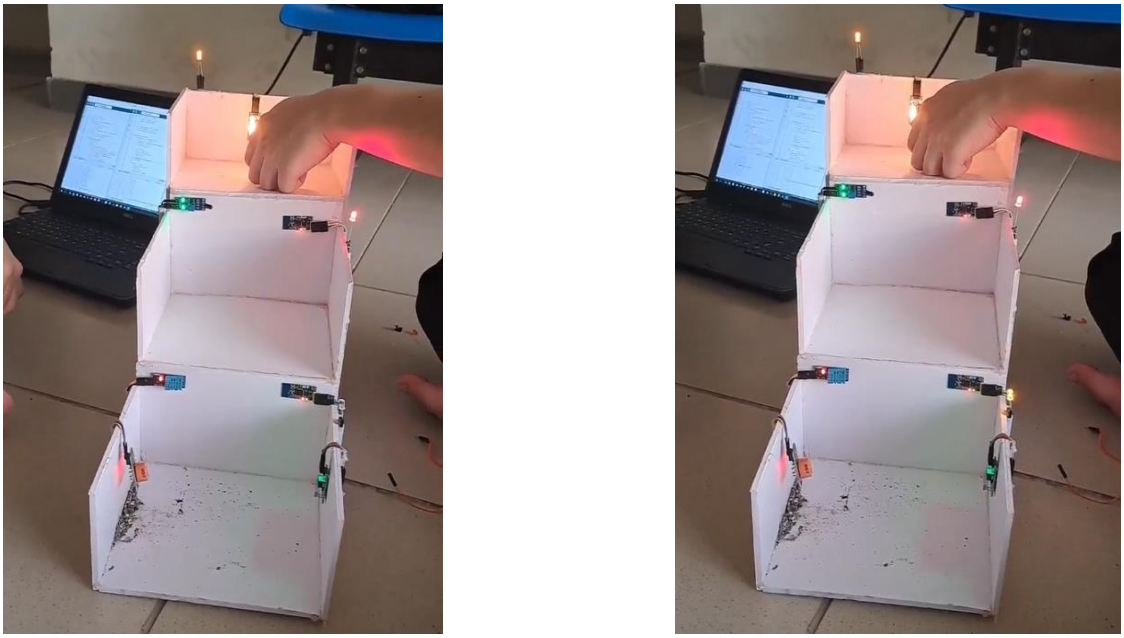

Figure 23. Fire in floor 3

3. Display data on web

- When the system meets nothing unusual, or when the system is in sleeping period:

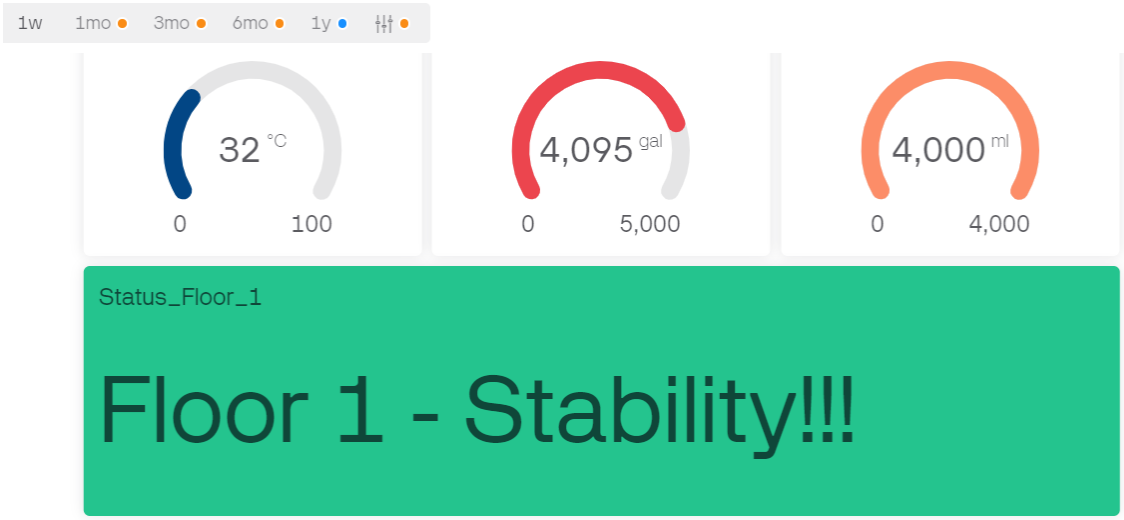

Figure 24. Data display when the system is stable

- Floor 1 encounters unusual data from one of 3 sensors and start checking:

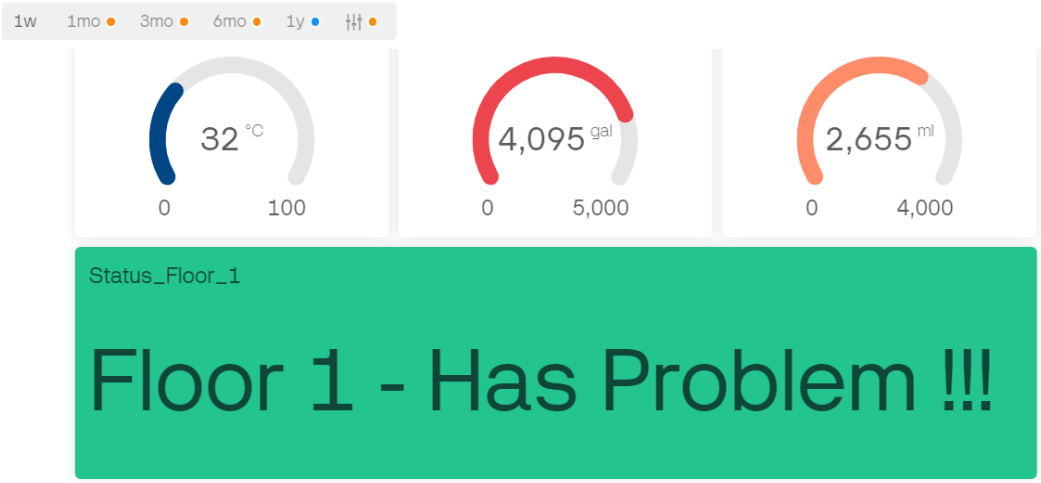

Figure 25. Data display when the system encounters problems

- The floor 1 confirms there is fire in the building, then sends fire alarm to other floors and spray water:

Figure 26. Data display when there is fire

VII. Conclusion

In conclusion, the development and implementation of our fire detection system using the ESP32 microcontroller demonstrate a significant advancement in ensuring the safety of multi-storey buildings in Vietnam. The system's integration of smoke, temperature, and fire sensors allows for prompt detection and response, effectively minimizing the risks associated with catastrophic fires. The use of the ESP32 microcontroller not only enhances the reliability and efficiency of the system but also ensures low power consumption, making it a sustainable solution. By utilizing this technology, we aim to provide a robust and proactive approach to fire safety, potentially saving lives and reducing property damage in the event of a fire. This project underscores the importance of modern technology in addressing critical safety challenges and highlights the potential for further innovations in this field.