AUTOMATIC FIRE EXTINGUISHER

MINI PROJECT REPORT

Submitted by

ABISHEAK A 9517202109004

MOHAMED ASLAM K 9517202109034

ROHIT K 9517202109044

in

19AD752 – INTELLIGENT SYSTEMS FOR IOT LABORATORY

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

MEPCO SCHLENK ENGINEERING COLLEGE

AUTONOMOUS

SIVAKASI

OCTOBER 2024

MEPCO SCHLENK ENGINEERING COLLEGE, SIVAKASI
AUTONOMOUS

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

BONAFIDE CERTIFICATE

This is to certify that it is the bonafide work of **"Abisheak A 9517202109004, Mohamed Aslam K 9517202109034, Rohit K 9517202109044"** for the Mini project titled **"Automatic Fire Exinguisher"** in 19AD752–Intelligent Systems for IOT Laboratory during theseventh semester June 2024–November 2024 under my supervision.

SIGNATURE

Dr.A.Shenbagarajan,

M.E.,Ph.D Associate Professor,

Artificial Intelligence and Data Science,

Mepco Schlenk Engineering College,

Autonomous,

Sivakasi – 626 005.

SIGNATURE

Dr.J.Angela Jennifa Sujana,

M.Tech.,Ph.D Professor & Head,

Artificial Intelligence and Data Science,

Mepco Schlenk Engineering College,

Autonomous,

Sivakasi – 626 005.

ACKNOWLEDGEMENT

First and foremost, we **praise and thank "The Almighty",** the lord of all creations, who by his abundant grace has sustained us and helped us to work on this project successfully.

We really find unique pleasure and immense gratitude in thanking our respected management members**,** who are the backbone of our college.

A deep bouquet of thanks to respected Principal **Dr.S.Arivazhagan M.E.,Ph.D.,** for having provided the facilities required for our mini project.

We sincerely thank our Head of the Department **Dr. J. Angela Jennifa Sujana M.E.,Ph.D.,** Professor & Head, Department of Artificial Intelligence and Data Science, for herguidance and support throughout the mini project .

We extremely thank our project coordinator **Dr.A.Shenbagarajan M.E.,Ph.D.,** Associate Professor and **Dr.E.Emerson Nithiyaraj M.E.,Ph.D.,** Assistant Professor, Department of Artificial Intelligence and Data Science, who inspired us andsupported us throughout the mini project.

We extend our heartfelt thanks and profound gratitude to all the faculty members of Artificial Intelligence and Data Science department for their kind help during our mini projectwork.

We also thank our parents and our friends who have provided us with constant support during the mini project work.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Fire safety is a critical concern in various environments, from residential spaces to industrial facilities. The Automatic Fire Extinguisher system provides an innovative solution that enhances safety through early detection and automated response to fire hazards. This system is built around an Arduino Uno microcontroller, which coordinates the actions of a flame sensor and a smoke sensor to monitor the environment continuously. When fire is detected, the flame sensor triggers the activation of a water pump to extinguish the fire, ensuring a swift response to potentially dangerous situations. Simultaneously, the system uses the smoke sensor to detect smoke, activating an exhaust fan to ventilate the area and clear the air. An audible alarm in the form of a buzzer is also triggered to alert nearby individuals. By automating the fire extinguishing and smoke control process, this system minimizes human intervention, reducing the time taken to respond to a fire emergency and potentially limiting the damage caused. The integration of the L298 motor module allows precise control of the exhaust fan and water pump, ensuring energy-efficient operation. This solution is not only effective in detecting and controlling fire-related hazards, but it is also cost-effective and easy to implement in various settings. The Automatic Fire Extinguisher system offers a scalable and reliable approach to fire safety, capable of being adapted for use in homes, offices, or industrial environments. It provides a proactive method of safeguarding lives and property by ensuring that fire hazards are promptly mitigated, reducing the likelihood of catastrophic damage.

1. INTRODUCTION

1.1 INTRODUCTION

Fire hazards pose a significant risk in both residential and industrial settings, often leading to property damage, injuries, and even loss of life. The rapid detection and timely response to fire-related incidents are crucial in minimizing the impact of such hazards. Traditional fire extinguishing systems often rely on manual intervention, which can be delayed or hindered, especially in high-risk environments or situations where human response is insufficient. To address this challenge, automation in fire detection and suppression systems has emerged as a practical solution. The Automatic Fire Extinguisher system is designed to provide a reliable, autonomous method of detecting and mitigating fire and smoke hazards. This project utilizes an Arduino Uno microcontroller as the central control unit, interfacing with a flame sensor and a smoke sensor to continuously monitor the environment. Upon detecting a fire, the flame sensor triggers the activation of a water pump to release water and extinguish the fire. In the event of smoke detection, the smoke sensor activates an exhaust fan to ventilate the area, reducing smoke buildup, while a buzzer serves as an audible alert to warn nearby individuals. One of the unique aspects of this system is its dual functionality: it handles both fire suppression and smoke control. The integration of the L298 motor module allows efficient control of the water pump and exhaust fan, ensuring that the system operates effectively without unnecessary power consumption. The 12V power supply supports the motor and other connected components, making the system robust and capable of functioning for extended periods.

1.2 OBJECTIVES

The aim of this project is to develop an automated fire extinguishing and smoke ventilation system using Arduino Uno and sensors to enhance fire safety in various environments. To achieve this objective, we consider the following goals:

1. **Develop a Real-Time Fire and Smoke Detection System**: Utilize a flame sensor to detect fire and a smoke sensor to identify the presence of smoke, ensuring immediate response through automated actions.

2. **Activate Fire Suppression System**: Upon detecting a fire, automatically trigger the water pump to extinguish the fire, minimizing the risk of the fire spreading and reducing potential damage.

3. **Implement Smoke Control Mechanism**: When smoke is detected, activate the exhaust fan to ventilate the area and clear smoke, improving air quality and visibility while reducing the risk of smoke inhalation.

4. **Provide an Audible Alert System**: Trigger a buzzer upon smoke detection to alert nearby individuals, ensuring timely evacuation or further manual intervention if necessary.

5. **Energy-Efficient Operation**: Optimize the system by using the L298 motor module to control the water pump and exhaust fan, ensuring efficient use of power and resources.

6. **Cost-Effective and Scalable Design**: Develop a low-cost solution using readily available components that can be easily implemented in a variety of settings, including residential, commercial, and industrial spaces.

1.3 SCOPE OF THE PROJECT

The Automatic Fire Extinguisher project focuses on developing a microcontroller-based fire safety system that can detect fire and smoke and automatically activate suppression and ventilation mechanisms. The scope of the project includes the following:

1. **Hardware Design**:

    - Integration of a flame sensor for detecting fire and a smoke sensor for monitoring smoke levels.

    - Use of an Arduino Uno to process sensor data and control the activation of the water pump, exhaust fan, and buzzer.

    - Implementation of the L298 motor driver module to control the exhaust fan and water pump.

    - Utilization of a 12V power supply to power the system components.

2. **Software Development**:

    - Writing a program that continuously reads data from the flame and smoke sensors and dynamically triggers appropriate actions (water pump for fire suppression, exhaust fan and buzzer for smoke control).

    - Implementing threshold-based control to distinguish between normal conditions and fire/smoke hazards.

    - Developing code to handle simultaneous sensor input and efficient motor control for real-time response.

3. **Testing and Validation**:

    - Testing the system under different fire and smoke scenarios to ensure reliable operation.

    - Validating the responsiveness and effectiveness of the water pump in extinguishing fire and the exhaust fan in clearing smoke.

    - Ensuring that the system's buzzer provides timely alerts in the presence of smoke.

4. **Real-World Applications**:

    - Integration into various environments, such as homes, offices, industrial facilities, or warehouses, providing automated fire detection and suppression capabilities.

    - Suitable for areas where rapid response to fire and smoke hazards is critical for safety and property protection.

5. **Future Enhancements**:

    - Incorporating additional sensors, such as temperature or humidity sensors, to further improve the system's adaptability to different environments.

    - Expanding the system to include remote monitoring and control through IoT platforms, allowing real-time alerts and responses via smartphones.

2. PROPOSED SOLUTION

2.1 BLOCK DIAGRAM

This block diagram illustrates the operation of a fire and smoke detection system. The system first initializes, checks sensors, and then responds to fire or smoke by activating the water pump or exhaust fan, respectively. A buzzer sounds to alert, and the system will continue monitoring until the fire or smoke is no longer detected **(Figure 2.1).**

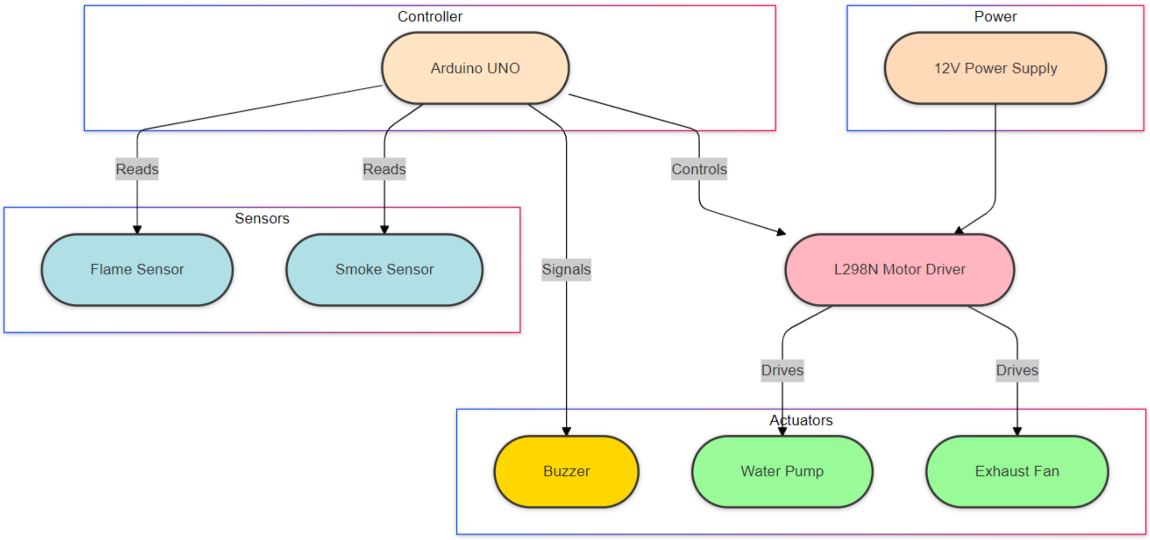

Figure 2.1 Block Diagram

2.2 CIRCUIT DIAGRAM

This diagram outlines the hardware connections for a fire and smoke detection system using an Arduino Uno. It includes a flame sensor, smoke sensor, water pump, buzzer, and exhaust fan, controlled through an L298N motor module and powered by a 12V supply. The system responds to sensor inputs by activating the appropriate devices **(Figure 2.2).**

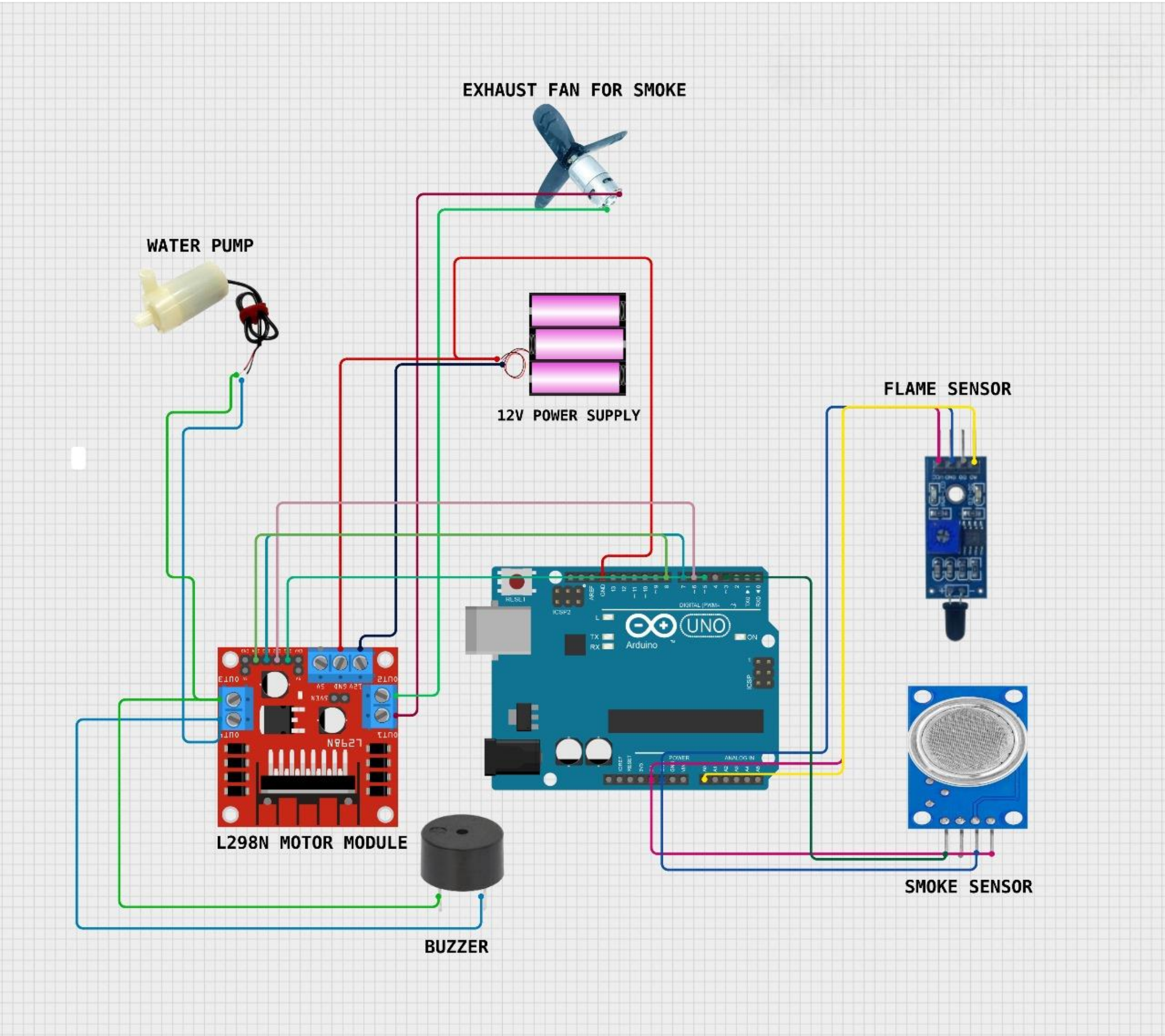

Figure 2.2 Circuit Diagram

2.3 HARDWARE AND SOFTWARE REQUIREMENTS

- Arduino Uno
- Flame Sensor
- Smoke Sensor
- Water Pump
- Exhaust Fan
- Buzzer
- L298 Motor Driver Module
- 12V Power Supply
- Jumper Wires
- Breadboard
- USB Cable

SOFTWARE REQUIRMENTS

Arduino IDE Software

2.4 ARDUINO UNO

1. Microcontroller: The Arduino Uno is based on the ATmega328P microcontroller, which is an 8-bit microcontroller that runs at 16 MHz. It can execute various tasks such as reading inputs from sensors and controlling outputs like motors and LEDs.

2. Digital and Analog I/O Pins: The board features 14 digital input/output pins, of which 6 can be used as PWM outputs. Additionally, it has 6 analog input pins, allowing it to read signals from various sensors.

3.USB Interface: The Arduino Uno includes a USB interface for programming and powering the board. It can be connected to a computer via a USB cable, making it easy to upload code and communicate with the Arduino IDE.

4.Extensive Community Support: The Arduino platform has a vast and active community, providing numerous libraries, tutorials, and resources. This makes it accessible for both beginners and experienced developers to create a wide range of projects.

5.Versatility and Compatibility: The Arduino Uno is compatible with a variety of shields and modules, allowing users to expand its functionality easily. It can be used in diverse applications, including robotics, IoT projects, home automation, and educational purposes.

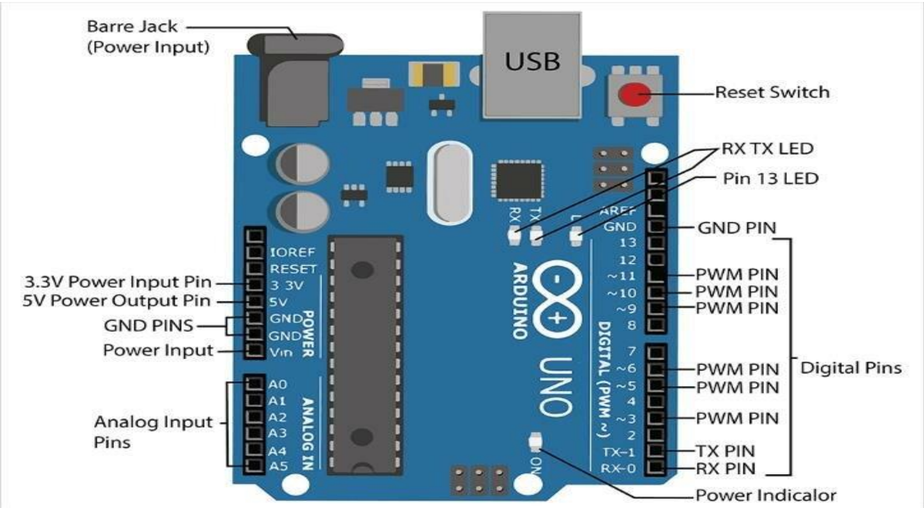

Figure 2.3 Arduino Uno

- **ATmega328 Microcontroller-** It is a single chip Microcontroller of the ATmel family. Theprocessor code inside it is 8-bit. It combines Memory (SRAM, EEPROM, and Flash), Analog to Digital Converter, SPI serial ports, I/O lines, registers, timer, external and internal interrupts, and oscillator.
- **ICSP pin-** The In-Circuit Serial Programming pin allows the user to program using thefirmware of the Arduino board.
- **Power LED Indicator-** The ON status of LED shows the power is activated. When thepower is OFF, the LED will not light up.
- **Digital I/O pins-** The digital pins have the value HIGH or LOW. The

pins numbered fromD0 to D13 are digital pins.

- **TX and RX LED's-** The successful flow of data is represented by the lighting of theseLED's.
- **AREF-** The Analog Reference (AREF) pin is used to feed a reference voltage to the ArduinoUNO board from the external power supply.
- **Reset button-** It is used to add a Reset button to the connection.
- **USB-** It allows the board to connect to the computer. It is essential for the programming ofthe Arduino UNO board.
- **Crystal Oscillator-** The Crystal oscillator has a frequency of 16MHz, which makes theArduino UNO a powerful board.
- **Voltage Regulator-** The voltage regulator converts the input voltage to 5V.
- **GND-** Ground pins. The ground pin acts as a pin with zero voltage.
- **Vin-** It is the input voltage.

Analog Pins- The pins numbered from A0 to A5 are analog pins. The function of Analog pins is to read the analog sensor used in the connection. It can also act as GPIO (General Purpose Input Output) pins.

2.5 FLAME SENSOR

- **Detection Range**: Flame sensors can typically detect flames from distances of about **1 to 3 meters** (3 to 10 feet) depending on the sensor model and environmental conditions.

- **Temperature Sensitivity**: Most flame sensors can detect flames at temperatures ranging from **300°C to 1,000°C** (572°F to 1,832°F), depending on the type of sensor used (e.g., infrared or ultraviolet).

- **Detection Angle**: The effective detection angle for flame sensors is usually around **60 to 90 degrees**, allowing them to monitor a wide area for flames.

- **Response Time**: Flame sensors have a fast response time, typically between **2 to 10 seconds**, enabling them to react quickly to the presence of flames.

- **Wavelength Sensitivity**: Infrared flame sensors are sensitive to specific wavelengths, usually around **3 to 5 micrometers** for infrared detection, which corresponds to the emission spectrum of most flames.

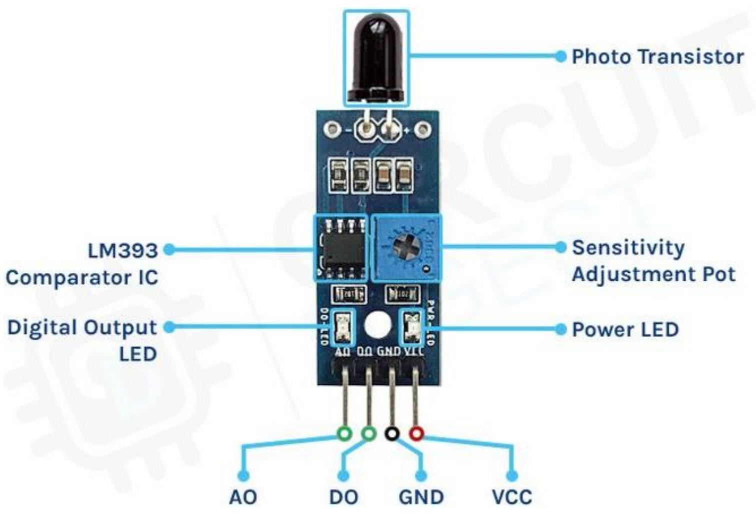

Figure 2.4 Flame Sensor

1. VCC Pin

- Description: This pin is used to supply power to the flame sensor.
- Typical Voltage: Usually requires a voltage of 5V for operation.

2. GND Pin

- Description: This pin is the ground connection.
- Usage: Connects to the ground (0V) of the power supply or microcontroller.

3. AO (Analog Output) Pin

- Description: This pin provides an analog signal that varies based on the intensity of the flame detected.
- Usage: Used when you want to read the strength of the flame. The output voltage typically ranges from 0V to the supply voltage (5V) based on the detected flame intensity.

4. DO (Digital Output) Pin

- Description: This pin provides a digital signal indicating whether a flame has been detected.
- Usage: It outputs a HIGH signal (usually 5V) when a flame is detected and a LOW signal (0V) when no flame is present. This is suitable for binary (yes/no) detection.

2.6 SMOKE SENSOR

1. Detection Method: Smoke sensors, such as the MQ series, detect the presence of smoke particles in the air, indicating potential fire hazards.

2. Analog and Digital Outputs: They provide both analog output (varying voltage based on smoke concentration) and digital output (HIGH/LOW signal for smoke detection).

3. Operating Voltage: Most smoke sensors operate on a 5V power supply, making them compatible with microcontrollers like Arduino.

4. Response Time: Smoke sensors typically have a fast response time, allowing for quick detection of smoke and potential fire emergencies.

Figure 2.5 Smoke Sensor

VCC Pin

- Description: This pin is used to supply power to the smoke sensor.
- Typical Voltage: Usually operates at 5V.

GND Pin

- Description: This pin is the ground connection for the sensor.
- Usage: Connects to the ground (0V) of the power supply or microcontroller.

AO (Analog Output) Pin

- Description: This pin provides an analog voltage output that varies based on the concentration of smoke detected.
- Usage: You can connect this pin to an analog input pin on a microcontroller (like Arduino) to read varying levels of smoke concentration. The output voltage typically ranges from 0V to the supply voltage (5V).

DO (Digital Output) Pin

- Description: This pin outputs a digital signal indicating the presence of smoke.
- Usage: It will output a HIGH signal (typically 5V) when smoke is detected and a LOW signal (0V) when no smoke is present. This is suitable for binary detection.

2.7 WATER PUMP

1. **Functionality**: Water pumps are devices used to move water from one location to another. They can be used for various applications, including irrigation, drainage, water supply, and firefighting systems.
2. **Types**: Common types of water pumps include centrifugal pumps (which use rotational energy) and positive displacement pumps (which move water by trapping a fixed amount and forcing it into the discharge). Submersible pumps and diaphragm pumps are also widely used.
3. **Flow Rate**: The flow rate of a water pump is typically measured in liters per minute (L/min) or gallons per minute (GPM). The flow rate indicates how much water the pump can move in a given time and varies based on the pump's design and operating conditions.
4. **Speed and Power:** Water pumps are often rated by their motor speed (in RPM) and power consumption (in Watts or HP). A typical small water pump may operate at speeds around 1,500 to 3,600 RPM and consume

anywhere from 50 Watts to several hundred Watts, depending on the pump size and application.

5. **Pressure Rating**: Water pumps are also rated for pressure, measured in pounds per square inch (PSI) or bar. This rating indicates the maximum pressure the pump can produce, which is crucial for applications requiring high water delivery pressure, such as in firefighting systems.

Figure 2.5 Water Pump

1. **Positive (+) Terminal**

- **Description**: This terminal is used to connect the positive power supply to the pump motor.
- **Usage**: When connecting to a power source, this pin typically connects to the positive terminal of a battery or power supply (e.g., 6V to 12V for small DC pumps).

2. **Negative (-) Terminal**

- **Description**: This terminal connects to the ground or negative side of the power supply.
- **Usage**: This pin connects to the negative terminal of the power supply or ground in your circuit. Proper grounding is essential for the pump to operate correctly.

2.8 EXHAUST FAN

1. Speed and Power: Exhaust fans typically operate at speeds ranging from 1,000 to 3,000 RPM, with power consumption varying from 20 Watts to over 100 Watts, depending on the size and design.

2. **Airflow Rate**: The performance of an exhaust fan is often measured in CFM (Cubic Feet per Minute**)**, indicating how much air the fan can move. Typical residential exhaust fans may have airflow rates between 50 to 300 CFM**.**

3. Noise Level: Exhaust fans are rated for noise levels measured in decibels (dB)**.** Quiet models operate around 30 to 50 dB**,** while more powerful fans may reach 60 dB or higher**,** impacting comfort levels in living spaces.

Figure 2.6 Exhaust Fan

• Positive **(+) Terminal**

- **Description**: This terminal is used to connect to the positive side of the power supply.
- **Usage**: In a DC exhaust fan, this pin connects to the positive terminal of a DC power source (e.g., a battery or DC adapter).

• Negative **(-) Terminal**

- **Description**: This terminal connects to the negative side of the power supply or ground.
- **Usage**: This pin connects to the negative terminal of the power supply or the ground in your circuit.

- **Control Input Pin (if applicable)**

  - **Description**: This pin allows for variable control over the fan speed.
  - **Usage**: In fans with PWM (Pulse Width Modulation) capability, this pin can be connected to a microcontroller (like Arduino) to adjust the speed of the fan based on the application requirements.

- **Speed Control Terminal (for multi-speed fans)**

  - **Description**: Some exhaust fans come with multiple speed settings, and this pin may be used to switch between those speeds.
  - **Usage**: It can connect to a switch or a control module that selects different speeds (low, medium, high) for the fan.

- **AC Connection Terminals (for AC fans)**

  - **Description**: For AC exhaust fans, there are typically two terminals for AC power connections.

2.9 L298 MOTOR MODULE

1. Dual H-Bridge Configuration

- The L298 module can control two DC motors or one stepper motor, allowing for bidirectional control of motor speed and direction using H-Bridge circuitry.

2. Power Supply Pins

- VCC: This pin connects to the power supply for the motors, typically rated between 5V and 35V. It powers the motors connected to the driver.
- GND: Connects to the ground of the power supply and the microcontroller, ensuring a common reference voltage.

3. Motor Control Pins

- Input Pins (IN1, IN2, IN3, IN4): These four pins are used to control the direction of the motors. By setting these pins HIGH or LOW, you can control the rotation direction of the connected motors.
  - Example:
    - To rotate Motor A forward: IN1 = HIGH, IN2 = LOW
    - To rotate Motor A backward: IN1 = LOW, IN2 = HIGH

4. PWM Control Pins

- Enable Pins (ENA, ENB): These pins allow for speed control of the motors using PWM signals. By sending a PWM signal to these pins, you can vary the speed of the motors.
  - Example: ENA controls Motor A speed, and ENB controls Motor B speed.

5. Current Rating

- The L298 motor driver can handle a continuous current of up to 2A per channel, with a maximum of 4A for short periods. It also features thermal shutdown protection to prevent overheating during operation.

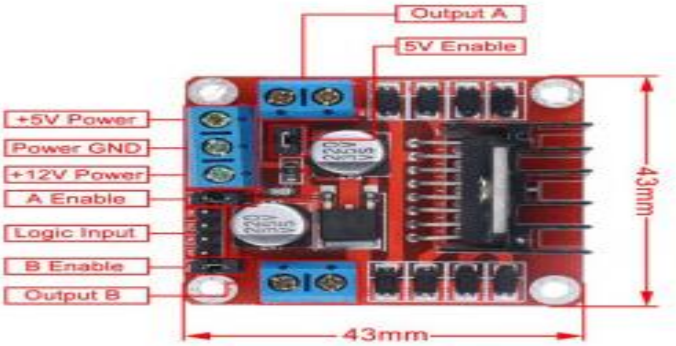

Figure 2.7 L298 Motor Module

1. Power Pins
2. VCC (Motor Supply Voltage):
    a. Description: This pin connects to the power supply for the motors.
    b. Voltage Range: Typically accepts voltage between 5V and 35V.
    c. Usage: Supplies power to the motors connected to the module.
3. GND (Ground):
    a. Description: This pin is the ground connection for the power supply and the microcontroller.
    b. Usage: Must be connected to the ground of the power supply and the microcontroller to ensure proper functioning.
4. Motor Control Pins
5. IN1:
    a. Description: This pin controls the direction of Motor A.
    b. Usage: Set HIGH to rotate Motor A in one direction; set LOW to stop or reverse.
6. IN2:
    a. Description: This pin also controls Motor A’s direction.
    b. Usage: Set HIGH to reverse Motor A; set LOW to stop or rotate in the opposite direction compared to IN1.

7. IN3:
    a. Description: This pin controls the direction of Motor B.
    b. Usage: Set HIGH to rotate Motor B in one direction; set LOW to stop or reverse.
8. IN4:
    a. Description: This pin also controls Motor B's direction.
    b. Usage: Set HIGH to reverse Motor B; set LOW to stop or rotate in the opposite direction compared to IN3.
9. Enable Pins
10. ENA:
    a. Description: This pin enables Motor A and can control its speed using PWM signals.
    b. Usage: Connect to a PWM-capable pin on the microcontroller. A HIGH signal enables the motor; varying the PWM duty cycle adjusts the speed.
11. ENB:
    a. Description: This pin enables Motor B and can also control its speed using PWM.
    b. Usage: Connect to a PWM-capable pin on the microcontroller for speed control. A HIGH signal enables the motor; varying the PWM duty cycle adjusts the speed.
12. Additional Pins
13. 12V Pin (if present):
    a. Description: Some L298 modules have an additional pin for connecting a higher voltage supply for the motors (e.g., 12V).
    b. Usage: Provides extra voltage for motors requiring more power, depending on the design of the module.

2.10 ARDUINO SOFTWARE

1. Main Components

- Menu Bar: Located at the top, this includes options for File (new, open, save), Edit (cut, copy, paste), Sketch (verify, upload), Tools (board selection, port selection), and Help (documentation, examples).
- Toolbar: This provides quick access to frequently used actions such as verifying code, uploading to the board, and opening the serial monitor.
- Code Editor: The central area where users write their Arduino sketches (programs). It supports syntax highlighting, which helps differentiate between keywords, variables, functions, and comments for better readability.

2. Code Editor Features

- Line Numbers: Displayed on the left side of the editor to help keep track of code lines.
- Auto-Completion: Offers suggestions as you type, helping speed up coding and reducing errors.
- Error Highlighting: Compiles code and highlights any syntax errors or issues, providing feedback before uploading.

3. Output and Serial Monitor

- Output Window: Located at the bottom, this shows messages related to the compilation process, including errors and warnings. It provides feedback after attempting to upload the code to the board.
- Serial Monitor: Accessible via the toolbar, this allows you to view

and send data between the Arduino and your computer. It is useful for debugging and monitoring real-time data.

4. Sketch Area

- Setup Function: This function runs once when the program starts. It's used for initializing variables, pin modes, etc.
- Loop Function: This function runs continuously after the setup function. It contains the main code that controls the Arduino's behaviour.

5. Library Management

- Library Manager: Accessible through the Sketch menu, it allows users to include and manage libraries, which are collections of code that simplify complex tasks and expand the Arduino's capabilities.

6. Board and Port Selection

- Board Selector: Located in the Tools menu, this lets you select the specific Arduino board you are using, ensuring that the code is compiled for the correct hardware.
- Port Selector: This allows you to select the correct port for uploading the code to your Arduino board, which is essential for communication between the computer and the board.

3. IMPLEMENTATION

3.1 SOURCE CODE

```
// Pin Definitions
const int smokePin = 2;          // Digital pin for smoke sensor
const int flamePin = 3;          // Digital pin for flame sensor
const int fanIN1 = 5;            // Pin connected to L298N IN1
   for fan
const int fanIN2 = 6;            // Pin connected to L298N IN2
   for fan
const int pumpIN3 = 7;           // Pin connected to L298N IN3
   for water pump
const int pumpIN4 = 8;           // Pin connected to L298N IN4
   for water pump
const int buzzerPin = 9;         // Pin connected to buzzer

void setup() {
  pinMode(smokePin, INPUT);      // Smoke sensor input
  pinMode(flamePin, INPUT);      // Flame sensor input
  pinMode(fanIN1, OUTPUT);       // Fan IN1 pin
  pinMode(fanIN2, OUTPUT);       // Fan IN2 pin
  pinMode(pumpIN3, OUTPUT);      // Pump IN3 pin
  pinMode(pumpIN4, OUTPUT);      // Pump IN4 pin
  pinMode(buzzerPin, OUTPUT);    // Buzzer pin

  // Initially turn everything off
  deactivateFan();
  deactivatePump();
  deactivateBuzzer();

  Serial.begin(9600);            // Begin serial communication
   for debugging
}

void loop() {
  int smokeDetected = digitalRead(smokePin);   // Read smoke
   sensor
  int flameDetected = digitalRead(flamePin);   // Read flame
   sensor

  Serial.print("Flame Detected: ");
```

```
  Serial.println(flameDetected);  // Print flame detection
   state

  if (smokeDetected == HIGH && flameDetected == LOW) {
    // Both smoke and flame detected, run fan, pump, and buzzer
    Serial.println("Both smoke and flame detected, activating
   all systems.");
    activateFan();
    activatePump();
    activateBuzzer();
  }
  else if (flameDetected == LOW) {
    // Flame detected, run water pump and sound buzzer
    Serial.println("Flame detected, activating water pump and
   buzzer.");
    activatePump();
    activateBuzzer();
    deactivateFan();
  }
  else if (smokeDetected == HIGH) {
    // Smoke detected, run fan as exhaust
    Serial.println("Smoke detected, activating exhaust fan.");
    activateFan();
    deactivatePump();
    deactivateBuzzer();
  }
  else {
    // No smoke or flame detected, turn everything off
    deactivateFan();
    deactivatePump();
    deactivateBuzzer();
  }

  delay(500);  // Delay to prevent sensor bouncing
}

// Function to activate fan using L298N motor driver
void activatePump() {
  digitalWrite(fanIN1, HIGH);  // Fan ON
  digitalWrite(fanIN2, LOW);   // Set direction (can be
   adjusted if needed)
}
```

```
// Function to deactivate fan
void deactivatePump() {
  digitalWrite(fanIN1, LOW);   // Fan OFF
  digitalWrite(fanIN2, LOW);
}

// Function to activate water pump using L298N motor driver
void activateFan() {
  digitalWrite(pumpIN3, HIGH);  // Pump ON
  digitalWrite(pumpIN4, LOW);   // Set direction (can be
   adjusted if needed)
}

// Function to deactivate water pump
void deactivateFan() {
  digitalWrite(pumpIN3, LOW);   // Pump OFF
  digitalWrite(pumpIN4, LOW);
}

// Function to activate buzzer directly through Arduino
void activateBuzzer() {
  digitalWrite(buzzerPin, HIGH);  // Buzzer ON
}

// Function to deactivate buzzer
void deactivateBuzzer() {
  digitalWrite(buzzerPin, LOW);   // Buzzer OFF
}
```

3.2 RESULTS

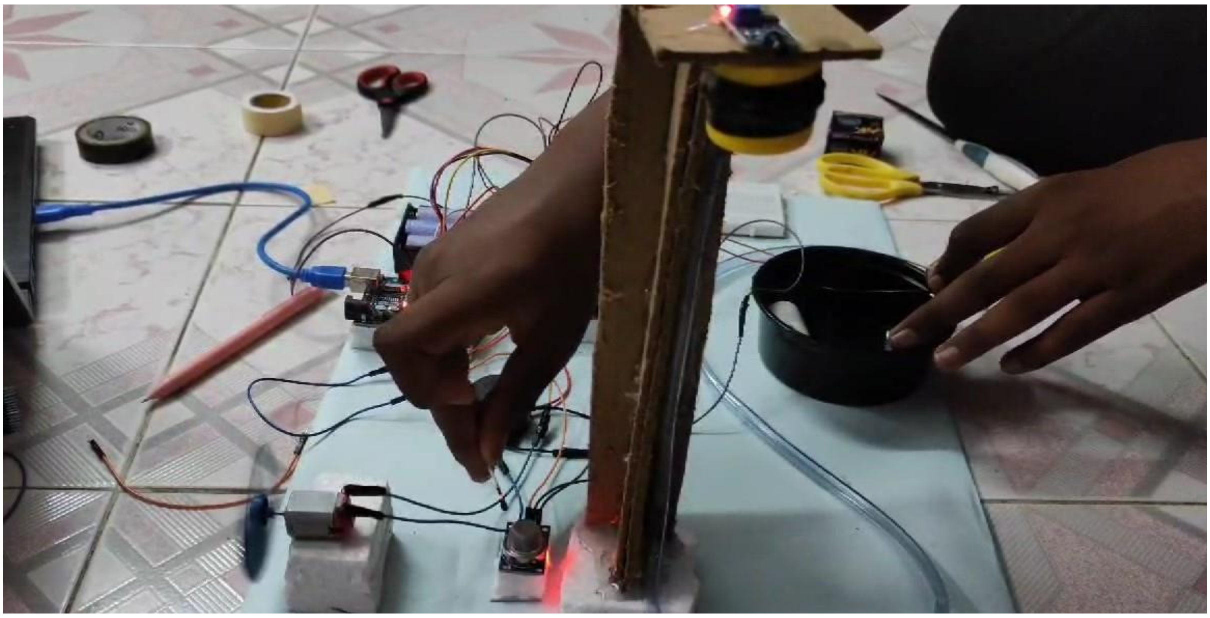

Fig 3.1 Smoke Sensor Detection

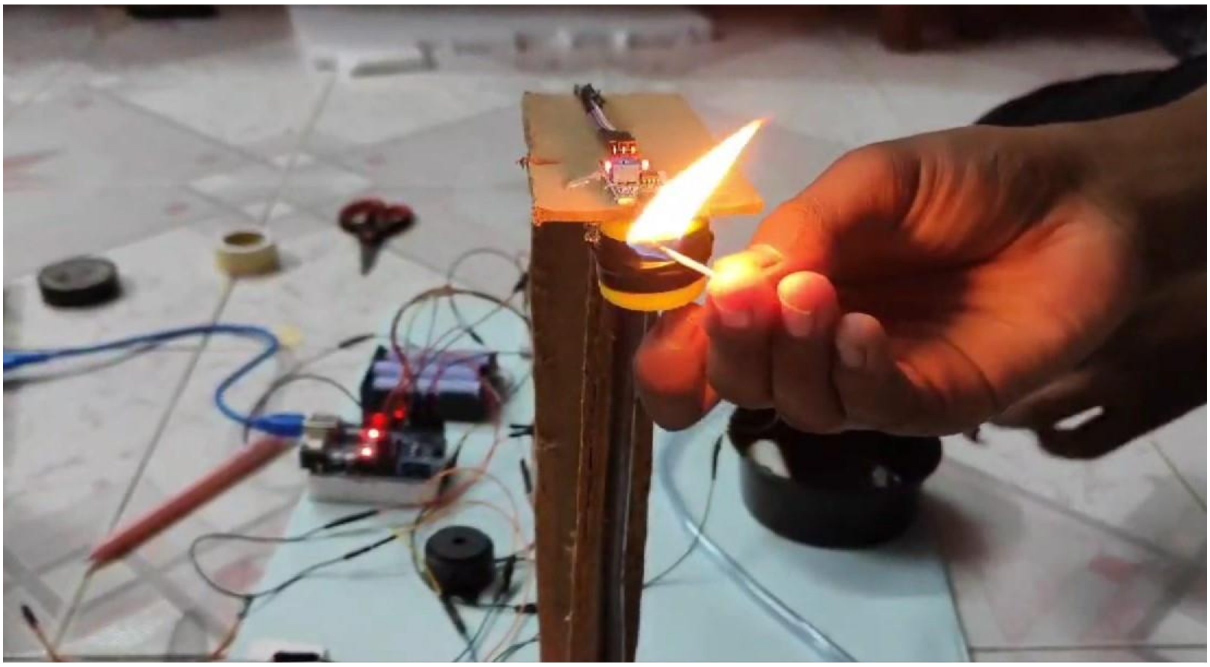

Fig 3.2 Flame Sensor Detection

4.CONCLUSION

The Automatic Fire Extinguisher project presents a vital innovation in fire safety and prevention, utilizing an efficient and responsive system to combat fire hazards in real-time. By integrating essential components such as smoke sensors, flame sensors, and a water pump, this system can swiftly detect both smoke and flames, activating appropriate countermeasures immediately. When fire is detected, the water pump is triggered to extinguish flames, while smoke detection activates an exhaust fan and buzzer, ensuring a prompt alert to potential dangers.

This project not only enhances safety in residential and commercial spaces but also serves as a cost-effective solution that can be easily implemented and adapted to various environments. The use of the Arduino Uno as the central controller allows for straightforward prototyping and further enhancements, ensuring the system remains versatile and user-friendly.

The implications of this project extend to numerous real-world applications, from homes to industrial settings, highlighting its potential to mitigate the devastating effects of fire incidents. Future enhancements could include the integration of wireless notifications, advanced sensor technologies, or connectivity with smart home systems for improved monitoring and control.

In summary, the Automatic Fire Extinguisher project is a proactive and forward-thinking approach to fire safety, promising to enhance protection and peace of mind for individuals and communities alike while reducing the risk of property damage and loss of life.

5.REFERENCE

5.1 REFERENCE

https://www.arduino.cc/en/Reference/HomePage

https://create.arduino.cc/projecthub/projects/tags/l298

Wani, A. R., & Kumar, A. (2017). "Design and Implementation of Smoke Detection and Alarm System." *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, 6(4).

Sabine, H. J. (2010). "Fire Protection Systems." *Fire Safety Journal*, 45(8), 568-577.