*A Project Report*

*on*

# ADVANCED LPG GAS LEAKAGE DETECTION & ALERT SYSTEM

*Submitted in partial fulfillment of the requirements for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

*In*

**ELECTRICAL AND ELECTRONICS ENGINEERING**


Submitted by

| | |
|---|---|
| **K. SANJANA MOULYA** | **21B91A0274** |
| P.SAI GOVARDHAN VARMA | 21B91A02C2 |
| M.PRASAD | 21B91A0297 |
| K.V.S. BHAVANI PRASAD | 21B91A0283 |
| K. VIVEK MOHAN | 21B91A0280 |

Under the esteemed guidance of

**Sri. B.ESWAR RAO**
Assistant Professor, EEE Department

**DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING**

**S.R.K.R. ENGINEERING COLLEGE (A)**
**(Affiliated to JNTU KAKINADA)(Approved by A.I.C.T.E, New Delhi)**
**(Accredited by N.B.A., NAAC with 'A+' grade, New Delhi)**
**SRKR MARG, CHINNA AMIRAM, BHIMAVARAM-534204.**


**(2025)**

hgp

# S.R.K.R. ENGINEERING COLLEGE (A)

**(Affiliated to JNTU KAKINADA) (Recognized by A.I.C.T.E, New Delhi)**
**(Accredited by N.B.A., NAAC with 'A+' grade, New Delhi)**
**CHINNA AMIRAM, BHIMAVARAM-534204**

## DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

# Certificate

This is to certify that Mr./Ms. K.SANJANA MOULYA of final year B. Tech., EEE, have carried out the project work on "ADVANCED LPG GAS LEAKAGE DETECTION & ALERT SYSTEM" in partial fulfillment of the requirements for the award of the Degree of 'Bachelor of Technology' with specialization of Electrical and Electronics Engineering in S.R.K.R. Engineering College (A), Bhimavaram.This is a bonafide record of the work done by us during the academic year 2024-2025.The results of this project work have not been submitted to any other university or Institute for the award of any degree.

| Guide | Head of the Department |
|---|---|
| **B.ESWAR RAO** | **Dr. B.R.K. VARMA, M.E., Ph.D.** |
| Assistant Professor | Professor & HOD |

# CERTIFICATE OF EXAMINATION

This is to certify that we had examined the project report and here by accord our approval of it as a study carried out and presented in a manner required for its acceptance in partial fulfillment of the award of the degree of **BACHELOR OF TECHNOLOGY** in **ELECTRICAL AND ELECTRONICS ENGINEERING** for which it has been submitted.

This approval does not necessarily endorse or accept every statement made, opinion expressed or conclusions drawn as recorded in the report. It only signifies the acceptance of the report for the purpose of which it is submitted.

**INTERNAL EXAMINER** **EXTERNAL EXAMINER**

# DECLARATION

This thesis entitled "ADVANCED LPG GAS LEAKAGE DETECTION & ALERT SYSTEM" has been carried out by us in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in the Department of Electrical and Electronics Engineering, Sagi Rama Krishnam Raju Engineering College (A), Bhimavaram. We hereby declare that this work has not been submitted to any other university/institute for the award of any other degree/diploma.

| **Roll. No.** | **Name** | **Signature** |
|---|---|---|
| 21B91A0274 | K.Sanjana Moulya | _____ |

# ACKNOWLEDGEMENT

Words are inadequate to express the overwhelming sense of gratitude and humble regards to my supervisor Sri. B.Eswar Rao, Assistant Professor in Department of Electrical and Electronics Engineering for his constant motivation, support, expert guidance, constant supervision and constructive suggestion for the submission of my progress report of thesis work “ADVANCED LPG GAS LEAKAGE DETECTION & ALERT SYSTEM”.We would like to express our deep sense of gratitude and sincere regards to Dr.B.R.K. VARMA, Head of Electrical and Electronics Engineering Department for his intense support throughout the project work. We are over whelmed to thank our principal Dr. K. V. MURALI KRISHNAM RAJU and management of S.R.K.R Engineering College who has given all the facilities and support for the successful completion of our project.We take this opportunity to express our acknowledgement to all the faculty members of Electrical and Electronics Engineering Department for giving valuable advises and support which paved the way for successful completion of our project.We take this opportunity to convey our sincere thanks to all those who have directly and indirectly contributed for the successful completion of our project work

**ProjectAssociated by**

21B91A0274

# ABSTRACT

The Gas Leakage Detection and Alert System is a safety solution designed for industrial and home environments to detect and respond to hazardous gas leaks. Using a gas sensor, it continuously monitors for dangerous concentrations of gases like LPG, methane, and fire. When a leak is detected, the system provides visual alerts on an LCD, sounds a buzzer, shuts off the gas valve to prevent further leakage, and sends real-time notifications to a mobile app via IoT. This ensures prompt action to minimize risks and enhance safety. The system is suitable for various applications, including industrial settings, home environments, and commercial establishments, providing an added layer of safety and security. Its advanced features and benefits make it a reliable and efficient solution for detecting and responding to gas leaks.

These protocols include activating a visual alert on an integrated LCD display, sounding a loud buzzer to draw immediate attention, and automatically shutting off the gas supply via an electronically controlled valve—thereby preventing further leakage and potential disasters. Simultaneously, the system employs IoT technology to send real-time alerts and status updates directly to a connected mobile application, ensuring that users can take prompt action, even remotely.

# TABLE OF CONTENTS

# LIST OF FIGURES

# CHAPTER 1

# INTRODUCTION

Gas leaks pose a significant threat to life, property, and the environment in both domestic and industrial settings. Even a minor leakage of combustible gases like LPG (Liquefied Petroleum Gas), methane, or natural gas can lead to devastating consequences, including fire outbreaks, explosions, and serious health complications such as respiratory issues or suffocation. In densely populated areas or confined spaces, the risks are even more pronounced due to the potential for rapid gas accumulation and limited ventilation.

Traditional gas leak detection methods primarily depend on manual monitoring or basic standalone alarms, which often lack the ability to deliver timely alerts or initiate automatic safety responses. Such limitations make them unreliable, especially when no one is present to notice the warning signals. Furthermore, manual systems are prone to human error, delayed reaction times, and false alarms triggered by non-critical environmental changes.

This project aims to overcome these shortcomings by developing an intelligent, automated Gas Leakage Detection and Alert System. By integrating advanced gas sensors, micro-controllers, and IoT communication modules, the system offers real-time detection of hazardous gas concentrations. Upon detecting a leak, the system performs a series of immediate actions: it activates a loud buzzer and visual alert on an LCD screen, automatically shuts off the gas supply using a servo-controlled valve, and sends real-time alerts to users via a mobile app using GSM/IoT connectivity.

Such automation significantly reduces human dependency and enables instant action, even in the user's absence. The system enhances safety by ensuring early detection and prevention, making it an ideal solution for homes, restaurants, factories, laboratories, and other environments where gas is in regular use. Moreover, the project supports future scalability to include multi-gas detection, smart home integration, and sustainable power backup solutions, thus reinforcing a proactive and comprehensive approach to gas safety management.

## 1.1 Overview of Gas Leakage Hazards and Safety Concerns

Gas leakage poses a significant threat to both human life and property, especially in domestic, commercial, and industrial environments where flammable gases such as LPG (Liquefied Petroleum Gas), methane, and natural gas are commonly used. A minor leak, if undetected, can lead to catastrophic consequences including fire outbreaks, explosions, and asphyxiation due to the displacement of oxygen in confined spaces.

One of the most dangerous aspects of gas leaks is their silent and often undetectable nature without proper sensors. Odorless or slow leaks can go unnoticed until they reach a concentration level that becomes highly flammable or toxic. In residential settings, leaks often occur due to faulty appliances, damaged pipelines, or loose connections. In industrial areas, high-pressure gas lines and storage tanks introduce even greater risks if not continuously monitored.

## 1.2 Importance of Early Detection System

Early detection of gas leakage is a critical component in ensuring safety and preventing hazardous incidents. Flammable and toxic gases like LPG, methane, and carbon monoxide can accumulate rapidly, especially in enclosed or poorly ventilated spaces. Without timely intervention, these gases can ignite with a small spark, leading to fires, explosions, or severe health hazards such as suffocation and poisoning.

An early detection system provides a proactive approach to managing gas-related risks. By continuously monitoring gas concentration levels and identifying abnormal increases at the very beginning, these systems can trigger alerts before the situation escalates to a dangerous level. This timely response window is vital in giving occupants or operators the chance to evacuate, shut down sources, or call for emergency services.

## 1.3 Objectives of the project

1. Continuously monitor gas concentration levels in the environment.
2. Provide immediate alerts via an LCD display, buzzer, and IoT notifications.
3. Automatically shut off the gas supply to prevent further leakage.
4. Ensure remote monitoring and notification through a mobile app.
5. Design a cost-effective and reliable system for domestic and industrial use.

## 1.4 Scope and applications

## Scope of the Project

The scope of this project encompasses the development of an intelligent, sensor-based gas leakage detection and alert system that integrates modern technologies such as IoT, GSM communication, and automation. The system is designed to detect the presence of flammable gases like LPG and methane, trigger immediate alerts, and take automated safety actions such as shutting off the gas supply.

This system is not limited to a single-use scenario but is designed to be modular, cost-effective, and scalable. It can be integrated into existing infrastructures or deployed as a standalone unit. Its functionality can be further enhanced with mobile applications, smart home systems, or cloud platforms for data visualization and historical tracking.

## Applications of the System

- **Residential Homes:**
  Monitors kitchens and utility areas to detect LPG leaks and alert homeowners, especially useful when residents are not present.
- **Commercial Establishments:**
  Ideal for restaurants, hotels, and catering facilities where LPG is used regularly, helping ensure compliance with safety standards.
- **Industrial Plants:**
  Used in manufacturing units, storage facilities, and chemical plants where large volumes of flammable gases are handled.
- **Hospitals and Healthcare Centers:**
  Helps maintain environmental safety standards in sensitive areas where gas leaks could have critical consequences.

# CHAPTER 2

# LITERATURE REVIEW

## Literature Survey 1:

In this paper, the authors proposed a wireless sensor network-based LPG gas leak detection and automatic gas regulator system using Arduino. The system employs gas sensors to detect leaks and activates an automatic regulator to control gas flow, enhancing safety. Additionally, it uses wireless communication to send alerts, providing a timely response to potential hazards and improving reliability in gas usage environments.

## Literature Survey 2:

In this paper, the authors developed a gas leakage detection system using Arduino and GSM technology. The system uses gas sensors to detect LPG leaks and immediately sends SMS alerts to users via the GSM module. This ensures quick awareness and response to gas leakage incidents, enhancing safety in both residential and commercial environments.

## Literature Survey 3:

In this paper, the author implemented a gas leakage detection system using the MQ-6 sensor, which is capable of detecting LPG, butane, and propane gases. The system utilizes an Arduino microcontroller to monitor gas concentration levels and trigger alerts when thresholds are exceeded. The paper emphasizes the importance of real-time monitoring and presents a low-cost solution suitable for residential and industrial safety applications.

## Literature Survey 4:

In this paper, the authors developed a gas leakage detection system designed for both domestic and industrial applications. The system employs the MQ-series gas sensor for detecting combustible gases and integrates a microcontroller for signal processing. It features both visual and audible alarms to alert users and can be connected to external devices for enhanced safety automation. The paper highlights the system's reliability, cost-effectiveness, and adaptability to various environments requiring gas monitoring.

## Literature Survey 5:

In this paper, the authors proposed an LPG gas leakage detection system featuring an automatic cutoff regulator integrated with an Arduino microcontroller. The system utilizes the MQ gas sensor to detect LPG presence and automatically actuates a servo motor to shut off the gas supply when a leak is detected. The solution also includes buzzer alerts and LED indicators for immediate local warning. The study emphasizes automation and safety, presenting a practical and cost-effective solution for preventing gas-related accidents in households.

# CHAPTER-3
# DESIGN AND IMPLEMENTATION OF THE PROPOSED GAS LEAKAGE DETECTION SYSTEM

## 3.1 Introduction to the Proposed System

The proposed system integrates a gas sensor with IoT technology to enhance safety through real-time monitoring and automated response. It combines local alerts with remote notifications to ensure timely action.

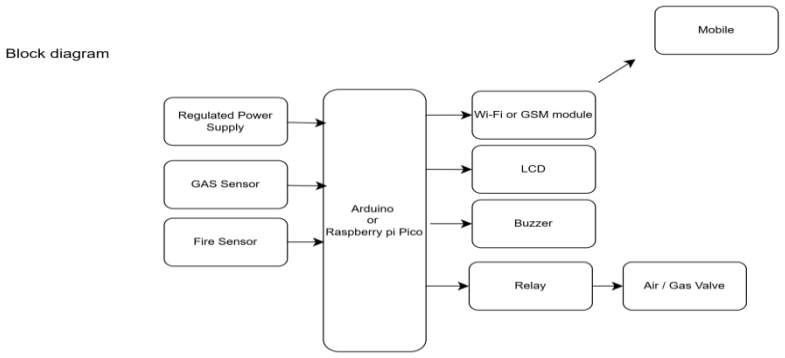



**Fig 3.1:Block diagram for LPG Gas Leakage Detection**

## 3.2 Working Steps of the System

1. **Gas Detection:** The gas sensor continuously monitors the environment for gas concentrations.
2. **Threshold Check:** When gas levels exceed a predefined threshold, the system triggers an alert.
3. **Automatic Valve Shutoff:** A servo motor-operated valve cuts off the gas supply to prevent further leakage
4. **Visual and Audible Alerts:** The LCD displays the gas levels, and a buzzer sounds to warn users.
5. **Emergency Response:** Authorities and responders are informed for immediate action.

## 3.3 Methodology

The methods employed in the design of the Gas Leakage Detection and Alert System include

- **Gas Sensor Selection**: Sensors are chosen based on the types of gases to be detected, such as methane, LPG, or carbon monoxide. Common sensors used include MQ series sensors (e.g., MQ-2 for methane) or electrochemical sensors that provide real-time gas concentration readings.
- **System Architecture**: The system uses an integrated architecture comprising a microcontroller (e.g., Arduino or Raspberry Pi), a gas sensor, an LCD display, a buzzer, a gas valve controller, and a mobile application via IoT. The microcontroller processes data from the sensor and triggers alerts.
- **Data Processing and Alerting Mechanism:** The sensor continuously monitors the concentration of gases. When the gas concentration surpasses a predefined threshold, the system activates the buzzer, displays a warning on the LCD, and sends a push notification to the mobile app using IoT protocols (e.g., MQTT or HTTP).
- **Valve Control**: The system is equipped with a solenoid valve to shut off the gas supply automatically when a leak is detected. This prevents further leakage and minimizes the risk of fire or explosion.

## 3.4 System Implementation

The implementation follows these steps:

- **Hardware Setup**: The sensor is connected to the microcontroller, with an LCD, buzzer, and solenoid valve as output devices. The mobile app communicates with the microcontroller through the internet (using Wi-Fi or GSM).
- **Software Setup**: Code is written to read the sensor data, check for gas leak thresholds, control the buzzer and valve, and send notifications. Platforms like Arduino IDE or Python (for Raspberry Pi) are typically used for the programming.
- **GSM-Based Alert System Integration:** A GSM-based communication system is implemented to ensure that users receive real-time alerts in case of a gas leak. Instead of using a mobile application, the system utilizes the SIM800L GSM module to send SMS notifications and automated phone calls directly to the user’s mobile number.

This allows the user to be informed immediately, even without internet connectivity or a dedicated app, ensuring quick response and enhanced safety from anywhere**.**

- **Testing and Calibration:** The system is tested in controlled environments to ensure accurate gas detection and proper alerting. Calibration is done to set the correct threshold levels for gas concentrations.

## 3.5 Tools and Technologies Used

- **Hardware Tools**

1. Arduino Board
2. Gas Sensor (e.g., MQ-5 or MQ-6)
3. Servo Motor (for valve control)
4. 16x2 LCD Display
5. Buzzer
6. Wi-Fi Module (e.g., ESP8266)
7. Power Supply
8. Gas Valve

- **Software Tools**

1. Arduino IDE

## 3.6 Advantages of the Proposed System

1. **Real-Time Monitoring:** Continuous detection of gas leaks.
2. **Automated Response:** Immediate valve shutoff to prevent further leakage.
3. **Remote Accessibility:** IoT integration allows real-time monitoring and notifications.
4. **Cost-Effective:** Affordable and scalable solution.
5. **Enhanced Safety:** Minimizes the risk of fire or explosion by taking proactive measures

## 3.7 Applications of the System

1. **Home Safety:** Ensures safe use of gas in residential kitchens.
2. **Industrial Safety:** Prevents accidents in gas storage and processing units.
3. **Hospitality Sector:** Monitors gas safety in hotels and restaurants.
4. **Laboratories:** Detects gas leaks in chemical and research labs

# CHAPTER 4

# HARDWARE DESCRIPTION

## 4.1 Overview of the Hardware System

The hardware system for the Gas Leakage Detection and Alert System is designed to provide a reliable, real-time safety solution by integrating multiple electronic components. At its core, the system utilizes a microcontroller (Arduino Uno) to interface with sensors and control modules that detect gas leaks and initiate immediate responses.The MQ-5 gas sensor continuously monitors the environment for the presence of combustible gases such as LPG and methane. When the gas concentration exceeds a predefined threshold, the microcontroller activates a buzzer for an audible alert and displays the warning on a 16x2 LCD screen.To prevent further leakage, the system includes a servo motor-controlled gas valve, which is triggered automatically to shut off the gas supply. Additionally, an exhaust fan can be activated using the MX1508 motor driver to ventilate the area and disperse the gas.A critical safety feature is the inclusion of the SIM800L GSM module, which sends SMS alerts and automated calls to the user's mobile number, ensuring that the user is informed immediately—even if they are not near the system.Each component is powered by a regulated power supply, ensuring consistent performance and reliability. The system is modular, easy to maintain, and scalable, making it suitable for a variety of applications, from household kitchens to industrial plants.

## 4.2 Arduino Uno Microcontroller

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter. "Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The

Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno **is** the latest in a series of USB Arduino boards, and the reference model for the Arduino platform;

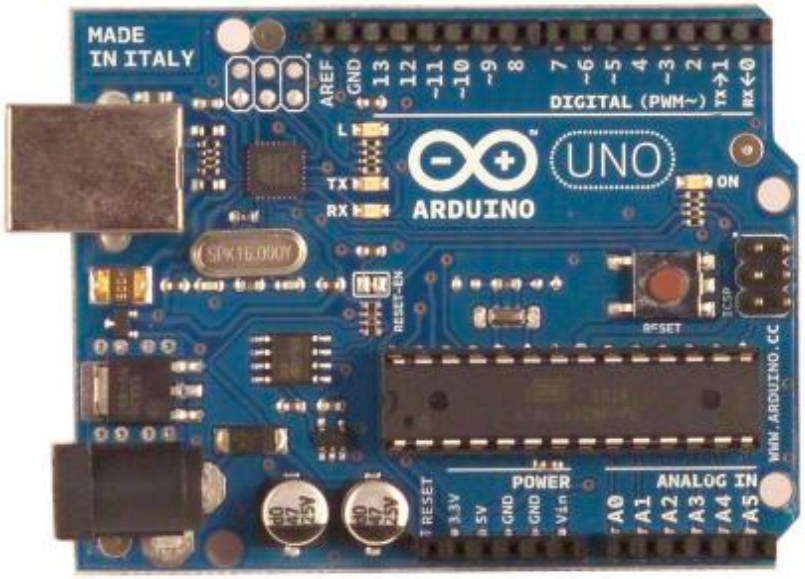



**Fig 4.2: Arduino Uno**

## 4.2.1 Why choosing Arduino Uno

There are numerous different microcontrollers and microcontroller platforms accessible for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and numerous others offer comparative usefulness. These apparatuses take the chaotic subtle elements of microcontroller programming and wrap it up in a simple to-utilize bundle. Arduino additionally rearranges the methodology of working with microcontrollers; moreover it offers some advantages for instructors, students, and intrigued individuals:

- **Inexpensive:** Arduino boards are moderately cheap compared with other microcontroller boards. The cheapest version of the Arduino module can be amassed by hand, and even the preassembled Arduino modules cost short of what $50.
- **Cross-platform:** The Arduino programming runs multiple operating systems Windows, Macintosh OSX, and Linux working frameworks. So we conclude that Arduino has an advantage as most microcontroller frameworks are constrained to Windows.
- **Clear programming method:** The Arduino programming environment is easy to use for novices, yet sufficiently versatile for cutting edge customers to adventure as well. For educators, its favorably engaged around the Processing programming

environment, so understudies finding ways to understand how to program in that environment will be familiar with the nature of arduino.

- **Open source and extensible programming:** The Arduino program language is available as open source, available for development by experienced engineers. The lingo can be reached out through C++ libraries, and people expecting to understand the specific purposes of different interests can make the leap from Arduino to the AVR C programming language on which it is based. Basically, you can incorporate AVR-C code clearly into your Arduino programs if you have to.
- **Open source and extensible hardware:** The Arduino is concentrated around Atmel's Atmega8 and Atmega168 microcontrollers. The plans for the modules are circulated under a Creative Commons license, so experienced circuit designers can make their own particular interpretation of the module, extending it and improving it. slightly inexperienced customers can build the breadboard variation of the module remembering the finished objective to perceive how it capacities and save money.

### 4.2.2 Technical specifications of arduino:

| **Specification** | **Details** |
|---|---|
| Microcontroller | ATmega328P |
| Operating Voltage | 5V |
| Input Voltage (Recommended) | 7V – 12V |
| Input Voltage (Limits) | 6V – 20V |
| Digital I/O Pins | 14 (6 provide PWM output) |
| Analog Input Pins | 6 |
| DC Current per I/O Pin | 40 mA |
| DC Current for 3.3V Pin | 50 mA |
| Flash Memory | 32 KB (0.5 KB used by bootloader) |
| SRAM | 2 KB |
| EEPROM | 1 KB |
| Clock Speed | 16 MHz |
| USB Interface | Atmega8U2 (USB-to-Serial Converter) |
| Communication Protocols | UART, I2C (TWI), SPI |

| | |
|---|---|
| PWM Pins | 3, 5, 6, 9, 10, 11 |
| LED Built-in | Digital Pin 13 |
| Board Dimensions | 68.6 mm x 53.4 mm |
| Weight | Approx. 25 g |

### 4.2.3 Pin Configuration:

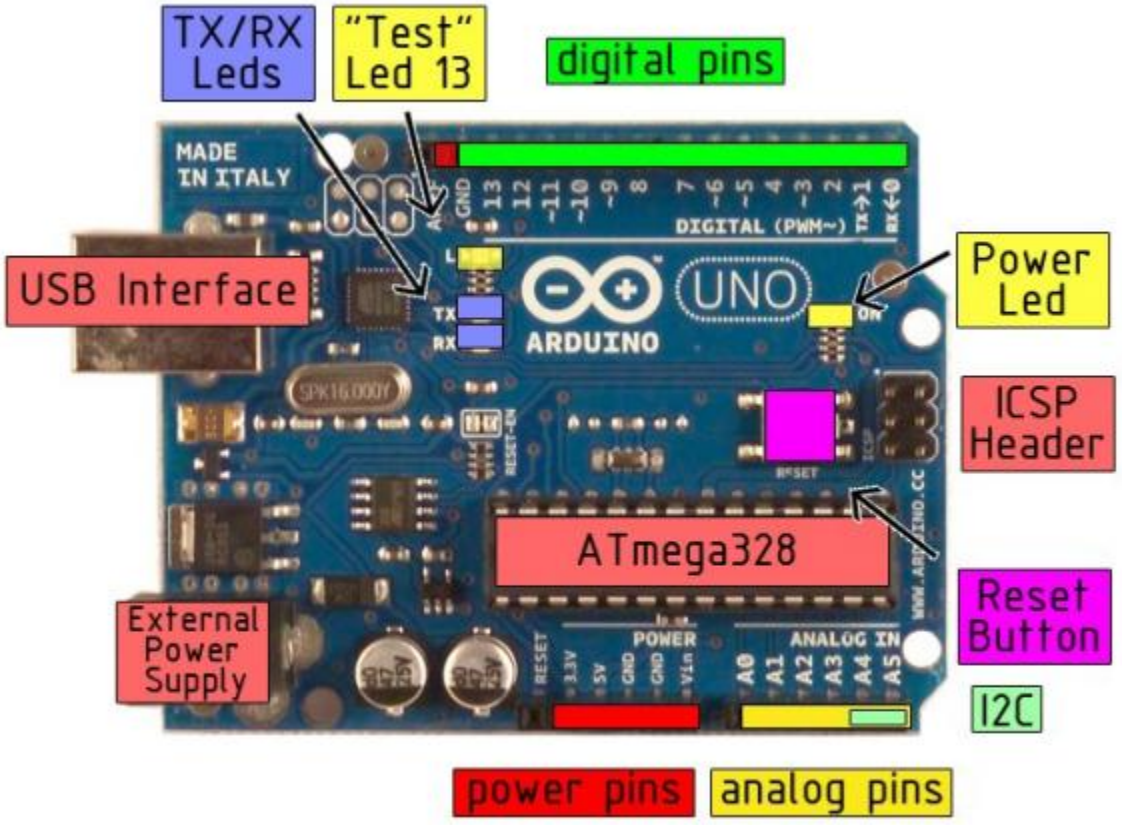



**Fig 4.2.3:Pin Configuration of Arduino Uno**

### Power Supply Options:

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector. The board can operate on an

external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

- **VIN:** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V:** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3.3V:** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND:** Ground pins.
- **MEMORY:** The Atmega328 has 32 KB of flash memory for storing code (of which 0,5 KB is used for the bootloader); It has also 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).
- **INPUT/OUTPUT:** Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:
- **Serial:** 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. TThese pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip .
- **External Interrupts-2 and 3:** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM:** 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.

- **SPI:** 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.
- **LED:** 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off. The Uno has 6 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:
- **I2C:** 4 (SDA) and 5 (SCL). Support I2C (TWI) communication using the Wire library. There are a couple of other pins on the board:
- **AREF:** Reference voltage for the analog inputs. Used with analogReference().
- **Reset:** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

## COMMUNICATION:

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega8U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '8U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, an *.inf file is required.

## Ease of Programming:

The Arduino Uno can be programmed with the Arduino software. The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C headerfiles). You can also bypass the bootloader and program the microcontroller through the ICSP (InCircuit Serial Programming) header; see these instructions for details. The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available . The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by: On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. 10 On Rev2 or later boards:

there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode. You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorial for more information.

## Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of theATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload. This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data. The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it.

## 4.3 Gas Sensor (MQ-5)

The MQ-5 is a widely used gas sensor for detecting flammable gases such as LPG, propane, methane, hydrogen, alcohol, and smoke. It has a high sensitivity and fast response time, making it ideal for gas leakage detection applications.

Operating Voltage: 5V DC

Sensing Range: 200 – 10,000 ppm (gas concentration)

Analog Output: Provides variable voltage depending on gas concentration

Digital Output (optional): Can be used via onboard comparator

Preheat Time: ~20 seconds for accurate readings

The sensor's output is read by a microcontroller (e.g., Arduino), which triggers alerts and actions when gas concentration exceeds a predefined threshold.

### 4.3.1 Pin Configuration:

| Pin No. | Label | Description |
|---|---|---|
| 1 | VCC | Connect to 5V DC power supply (Arduino 5V pin). |
| 2 | GND | Connect to ground. |
| 3 | AOUT | Analog Output – Outputs a variable voltage based on the gas concentration. |
| 4 | DOUT | Digital Output – Goes HIGH or LOW depending on a set threshold. |

## Features:

- Operating Voltage is +5V
- Can be used to Measure or detect LPG, Alcohol, Propane, Hydrogen, CO and even methane
- Analog output voltage: 0V to 5V
- Digital Output Voltage: 0V or 5V (TTL Logic)
- Preheat duration 20 seconds
- Can be used as a Digital or analog sensor

### 4.3.1 Where to use MQ-5 Gas sensor:

The MQ-5 Gas sensor can detect or measure gasses like LPG, Alcohol, Propane, Hydrogen, CO and even methane. The module version of this sensor comes with a Digital Pin which makes this sensor to operate even without a microcontroller and that comes in handy when you are only trying to detect one particular gas. When it comes to measuring the gas in ppm the analog pin has to be used, the analog pin also TTL driven and works on 5V and hence can be used with most common microcontrollers.

So if you are looking for a sensor to detect or measure gasses like LPG, Alcohol, Propane, Hydrogen, CO and even methane with or without a microcontroller then this sensor might be the right choice for you.

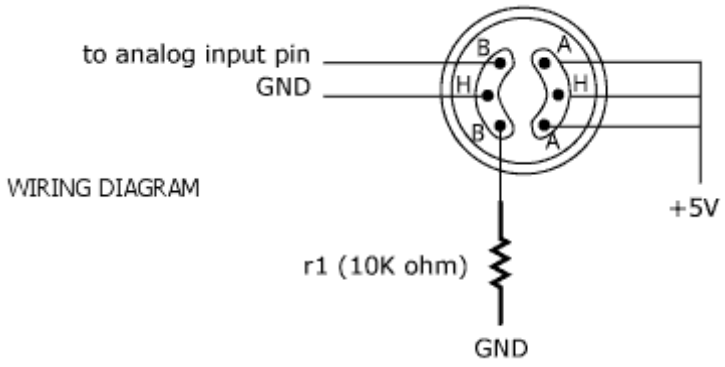
### 4.3.2 How to use MQ-2 Sensors to detect gas:

Using an MQ sensor it detects a gas is very easy. You can either use the digital pin or the analog pin to accomplish this. Simply power the module with 5V and you should notice the power LED on the module to glow and when no gas it detected the output LED will remain turned off meaning the digital output pin will be 0V. Remember that these sensors have to be kept on for pre-heating time (mentioned in features above) before you can actually work with it. Now, introduce the sensor to the gas you want to detect and you should see the output LED to go high along with the digital pin, if not use the potentiometer until the output gets high. Now every time your sensor gets introduced to this gas at this particular concentration the digital pin will go high (5V) else will remain low (0V).

You can also use the analog pin to achieve the same thing. Read the analog values (0-5V) using a microcontroller, this value will be directly proportional to the concentration of the gas to which the sensor detects. You can experiment with this values and check how the sensor reacts to different concentration of gas and develop your program accordingly.

### 4.3.3 How to use the MQ-5 sensor to measure PPM:

If you are looking for some accuracy with your readings then measuring the PPM would be the best way to go with it. It can also help you to distinguish one gas from another. So to measure PPM you can directly use a module. A basic wiring for the sensor from datasheet is shown below.




The procedure to measure PPM using MQ sensor is the same but few constant values will vary based on the type of MQ sensor used. Basically, we need to look into the (Rs/Ro) VS PPM graph given in the datasheet (also shown below).


**Fig 4.3:MQ-5 Gas Sensor**

The value of Ro is the value of resistance in fresh air and the value of Rs is the value of resistance in Gas concentration. First, you should calibrate the sensor by finding the values of Ro in fresh air and then use that value to find Rs using the formulae

Resistance of sensor(Rs): Rs=(Vc/VRL-1)×RL

Once we calculate Rs and Ro we can find the ratio and then use the graph shown above we can calculate the equivalent value of PPM for that particular gas.

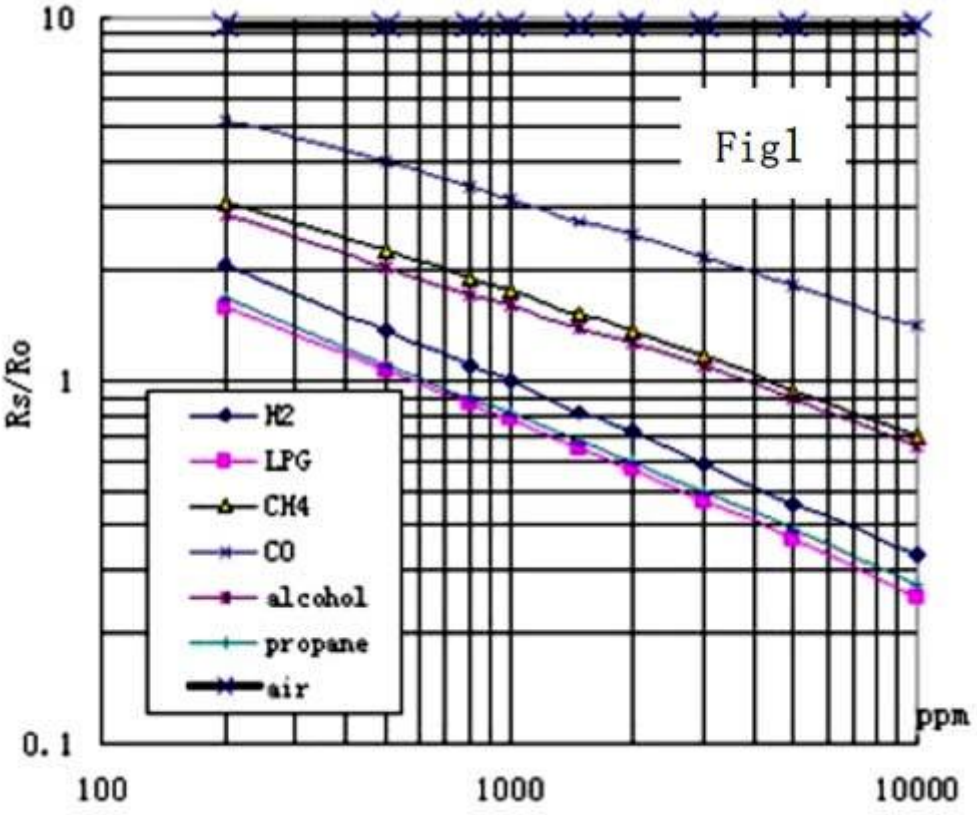



**Fig 4.3.3:(Rs/Ro) VS PPM graph for MQ-5 Sensor**

### 4.3.4 Applications:

- Detects or measure Gases like LPG, Alcohol, Propane, Hydrogen, CO and even methane
- Air quality monitor
- Gas leak alarm
- Safety standard maintenance

## 4.4 16x2 LCD Display

### 4.4.1 What is the LCD 16×2:

The term LCD stands for liquid crystal display. It is one kind of electronic display module used in an extensive range of applications like various circuits & devices like mobile phones, calculators, computers, TV sets, etc. These displays are mainly preferred for multi-segment light-emitting diodes and seven segments. The main benefits of using this module

are inexpensive; simply programmable, animations, and there are no limitations for displaying custom characters, special and even animations, etc.


**Fig 4.4:LCD Display**

## 4.4.2 LCD 16×2 Pin Diagram

The 16×2 LCD pinout is shown below.

- Pin1 (Ground/Source Pin): This is a GND pin of display, used to connect the GND terminal of the microcontroller unit or power source.
- Pin2 (VCC/Source Pin): This is the voltage supply pin of the display, used to connect the supply pin of the power source.
- Pin3 (V0/VEE/Control Pin): This pin regulates the difference of the display, used to connect a changeable POT that can supply 0 to 5V.
- Pin4 (Register Select/Control Pin): This pin toggles among command or data register, used to connect a microcontroller unit pin and obtains either 0 or 1(0 = data mode, and 1 = command mode).
- Pin5 (Read/Write/Control Pin): This pin toggles the display among the read or writes operation, and it is connected to a microcontroller unit pin to get either 0 or 1 (0 = Write Operation, and 1 = Read Operation).
- Pin 6 (Enable/Control Pin): This pin should be held high to execute Read/Write process, and it is connected to the microcontroller unit & constantly held high.
- Pins 7-14 (Data Pins): These pins are used to send data to the display. These pins are connected in two-wire modes like 4-wire mode and 8-wire mode. In 4-wire mode, only four pins are connected to the microcontroller unit like 0 to 3, whereas in 8-wire mode, 8-pins are connected to microcontroller unit like 0 to 7.
- Pin15 (+ve pin of the LED): This pin is connected to +5V

- Pin 16 (-ve pin of the LED): This pin is connected to GND.

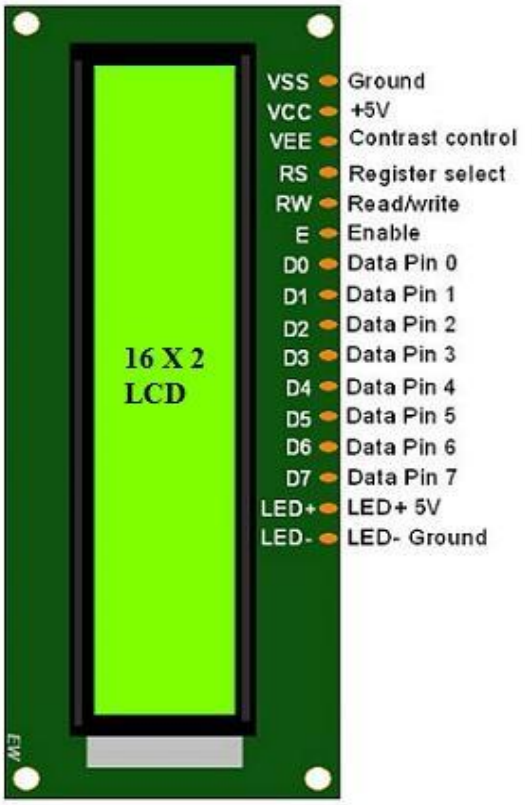



**Fig 4.4.2: LCD Pin-Diagram**

## 4.4.3 Features of LCD 16x2

The features of this LCD mainly include the following.

- The operating voltage of this LCD is 4.7V-5.3V
- It includes two rows where each row can produce 16-characters.
- The utilization of current is 1mA with no backlight
- Every character can be built with a 5×8 pixel box
- The alphanumeric LCDs alphabets & numbers
- Is display can work on two modes like 4-bit & 8-bit
- These are obtainable in Blue & Green Backlight
- It displays a few custom generated characters

## 4.4.4 Registers of LCD

A 16×2 LCD has two registers like data register and command register. The RS (register select) is mainly used to change from one register to another. When the register set is '0', then it is known as command register. Similarly, when the register set is '1', then it is known as data register.

### 4.4.5 Command Register

The main function of the command register is to store the instructions of command which are given to the display. So that predefined tasks can be performed such as clearing the display, initializing, set the cursor place, and display control. Here commands processing can occur within the register.

### 4.4.6 Data Register

The main function of the data register is to store the information which is to be exhibited on the LCD screen. Here, the ASCII value of the character is the information which is to be exhibited on the screen of LCD. Whenever we send the information to LCD, it transmits to the data register, and then the process will be starting there. When register set =1, then the data register will be selected.

### 4.4.7 16×2 LCD Commands

The commands of LCD 16X2 include the following.

- For Hex Code-01, the LCD command will be the clear LCD screen
- For Hex Code-02, the LCD command will be returning home
- For Hex Code-04, the LCD command will be decrement cursor
- For Hex Code-06, the LCD command will be Increment cursor
- For Hex Code-05, the LCD command will be Shift display right
- For Hex Code-07, the LCD command will be Shift display left
- For Hex Code-08, the LCD command will be Display off, cursor off
- For Hex Code-0A, the LCD command will be cursor on and display off
- For Hex Code-0C, the LCD command will be cursor off, display on
- For Hex Code-0E, the LCD command will be cursor blinking, Display on
- For Hex Code-0F, the LCD command will be cursor blinking, Display on
- For Hex Code-10, the LCD command will be Shift cursor position to left

- For Hex Code-14, the LCD command will be Shift cursor position to the right
- For Hex Code-18, the LCD command will be Shift the entire display to the left
- For Hex Code-1C, the LCD command will be Shift the entire display to the right
- For Hex Code-80, the LCD command will be Force cursor to the beginning ( 1st line)
- For Hex Code-C0, the LCD command will be Force cursor to the beginning ( 2nd line)
- For Hex Code-38, the LCD command will be 2 lines and 5×7 matrix.

## 4.5 Buzzer

A buzzer is an audio signaling device used in electronic systems to provide sound alerts. In this project, the buzzer serves as a warning alarm that is activated when a gas leak is detected.It operates on DC voltage (typically 5V–6V) and produces a continuous beep sound to alert users of danger. The buzzer is controlled by the microcontroller (Arduino), which turns it ON or OFF based on the sensor's output.There are two common types of buzzers: active and passive. For most embedded safety systems, including this project, an active buzzer is used because it can produce sound with just a DC voltage input, without needing an oscillating signal.

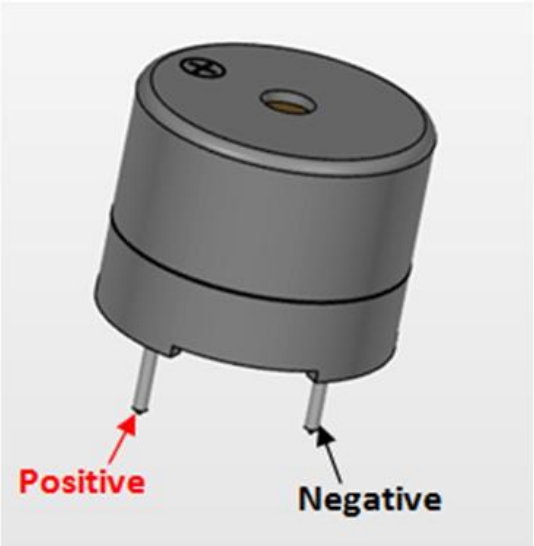



**Fig 4.5:Active Passive Buzzer** **fig4.5:Active Passive Buzzer Pinout**

## Pin Configuration:

| Pin Number | Pin Name | Description |
|---|---|---|
| 1 | Positive | Identified by (+) symbol or longer terminal lead. Can be powered by 6V DC |
| 2 | Negative | Identified by short terminal lead. Typically connected to the ground of the circuit |

## Buzzer Features and Specifications

- Rated Voltage: 6V DC
- Operating Voltage: 4-8V DC
- Rated current: <30mA
- Sound Type: Continuous Beep
- Resonant Frequency: ~2300 Hz
- Small and neat sealed package
- Breadboard and Perf board friendly

### 4.5.1 How to use a Buzzer

A buzzer is a small yet efficient component to add sound features to our project/system. It is very small and compact 2-pin structure hence can be easily used on breadboard, Perf Board and even on PCBs which makes this a widely used component in most electronic applications.

There are two types are buzzers that are commonly available. The one shown here is a simple buzzer which when powered will make a Continuous Beeeeeeppp.... sound, the other type is called a readymade buzzer which will look bulkier than this and will produce a Beep. Beep. Beep. Sound due to the internal oscillating circuit present inside it. But, the one shown here is most widely used because it can be customised with help of other circuits to fit easily in our application.

This buzzer can be used by simply powering it using a DC power supply ranging from 4V to 9V. A simple 9V battery can also be used, but it is recommended to use a regulated +5V

or +6V DC supply. The buzzer is normally associated with a switching circuit to turn ON or turn OFF the buzzer at required time and require interval.

### 4.5.2 Applications of Buzzer

- Alarming Circuits, where the user has to be alarmed about something
- Communication equipments
- Automobile electronics
- Portable equipments, due to its compact size

## 4.6 Servo Motor SG-90


**Fig4.6:SG-90 Servo Motor**


**Fig4.6:Servo Motor Pinout (Wires)**

## Wire Configuration

| Wire Number | Wire Colour | Description |
|---|---|---|
| 1 | Brown | Ground wire connected to the ground of system |
| 2 | Red | Powers the motor typically +5V is used |
| 3 | Orange | PWM signal is given in through this wire to drive the motor |

## TowerPro SG-90 Features

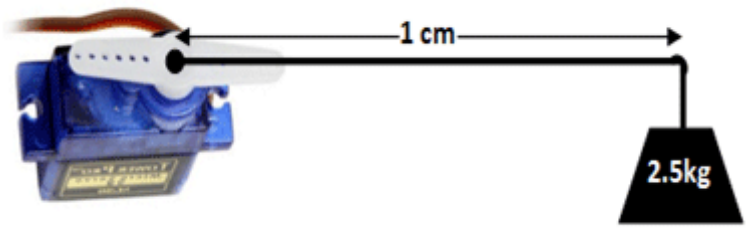
- Operating Voltage is +5V typically
- Torque: 2.5kg/cm
- Operating speed is 0.1s/60°
- Gear Type: Plastic
- Rotation : 0°-180°
- Weight of motor : 9gm
- Package includes gear horns and screws

### 4.6.1 Selecting your Servo Motor

There are lots of servo motors available in the market and each one has its own speciality and applications. The following two paragraphs will help you identify the right type of servo motor for your project/system.

Most of the hobby Servo motors operates from 4.8V to 6.5V, the higher the voltage higher the torque we can achieve, but most commonly they are operated at +5V. Almost all hobby servo motors can rotate only from 0° to 180° due to their gear arrangement so make sure you project can live with the half circle if no, you can prefer for a 0° to 360° motor or modify the motor to make a full circle. The gears in the motors are easily subjected to wear and tear, so if your application requires stronger and long running motors you can go with metal gears or just stick with normal plastic gear.

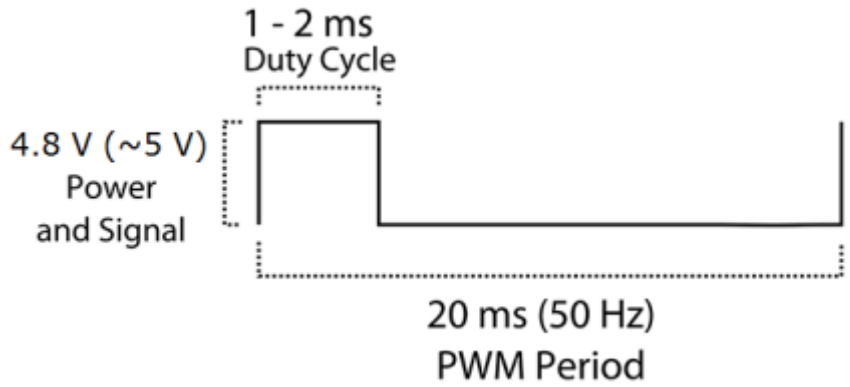
Next comes the most important parameter, which is the torque at which the motor operates. Again there are many choices here but the commonly available one is the 2.5kg/cm torque which comes with the Towerpro SG90 Motor. This 2.5kg/cm torque means that the motor can pull a weight of 2.5kg when it is suspended at a distance of 1cm. So if you suspend the load at 0.5cm then the motor can pull a load of 5kg similarly if you suspend the load at 2cm then can pull only 1.25. Based on the load which you use in the project you can select the motor with proper torque. The below picture will illustrate the same.




### 4.6.2 How to use a Servo Motor

After selecting the right Servo motor for the project, comes the question how to use it. As we know there are three wires coming out of this motor. The description of the same is given on top of this page. To make this motor rotate, we have to power the motor with +5V using the Red and Brown wire and send PWM signals to the Orange colour wire. Hence we need

something that could generate PWM signals to make this motor work, this something could be anything like a 555 Timer or other Microcontroller platforms like Arduino, PIC, ARM or even a microprocessor like Raspberry Pie. Now, how to control the direction of the motor? To understand that let us a look at the picture given in the datasheet.




From the picture we can understand that the PWM signal produced should have a frequency of 50Hz that is the PWM period should be 20ms. Out of which the On-Time can vary from 1ms to 2ms. So when the on-time is 1ms the motor will be in 0° and when 1.5ms the motor will be 90°, similarly when it is 2ms it will be 180°. So, by varying the on-time from 1ms to 2ms the motor can be controlled from 0° to 180°

### 4.6.3 Applications

- Used as actuators in many robots like Biped Robot, Hexapod, robotic arm etc..
- Commonly used for steering system in RC toys
- Robots where position control is required without feedback
- Less weight hence used in multi DOF robots like humanoid robots

## 4.7 Gas Regulator

A gas regulator is a crucial safety and control component used in gas-powered systems to manage the flow and pressure of LPG (Liquefied Petroleum Gas) from the cylinder to the appliance. In this project, the gas regulator works in combination with a servo motor to automatically shut off the gas supply during leakage, ensuring maximum safety.

## Working Principle:

The gas regulator maintains a steady and safe pressure of gas as it flows from the high-pressure cylinder to the low-pressure outlet that feeds appliances. If uncontrolled, gas pressure can be hazardous and lead to equipment damage or leaks.

In this system, the regulator is mechanically coupled with a servo motor. Upon gas leak detection by the MQ-2 sensor, the microcontroller activates the servo motor, which rotates and closes the gas valve, cutting off the gas supply and preventing further leakage.

**Fig4.7: Regulator**

## Key Features:

- **Flame Control:** Enables users to adjust the flame intensity easily.
- **Safety Lock:** Includes a child lock feature to prevent accidental tampering.
- **Secure Connection:** Multipoint grip system ensures tight, leak-free attachment to the gas cylinder.
- **Manual Override Option:** Allows manual operation in case of system failure.

## 4.8 SIM800L GPRS GSM Module

The SIM800L GSM module is a compact and powerful cellular communication device that allows microcontrollers like Arduino to send SMS messages, make phone calls, and access the internet via GPRS. In this project, it plays a crucial role by sending real-time alerts (SMS

and calls) when gas leakage is detected, ensuring the user is notified even when away from the site.

## Working Principle:

The SIM800L module communicates with the Arduino using AT commands over a serial interface (UART). When the system detects gas beyond a set threshold, the Arduino sends commands to the SIM800L to:

1.Send a text message (SMS) to a predefined mobile number.

2.Initiate a call as an emergency alert.

This ensures real-time remote communication, even without internet access.

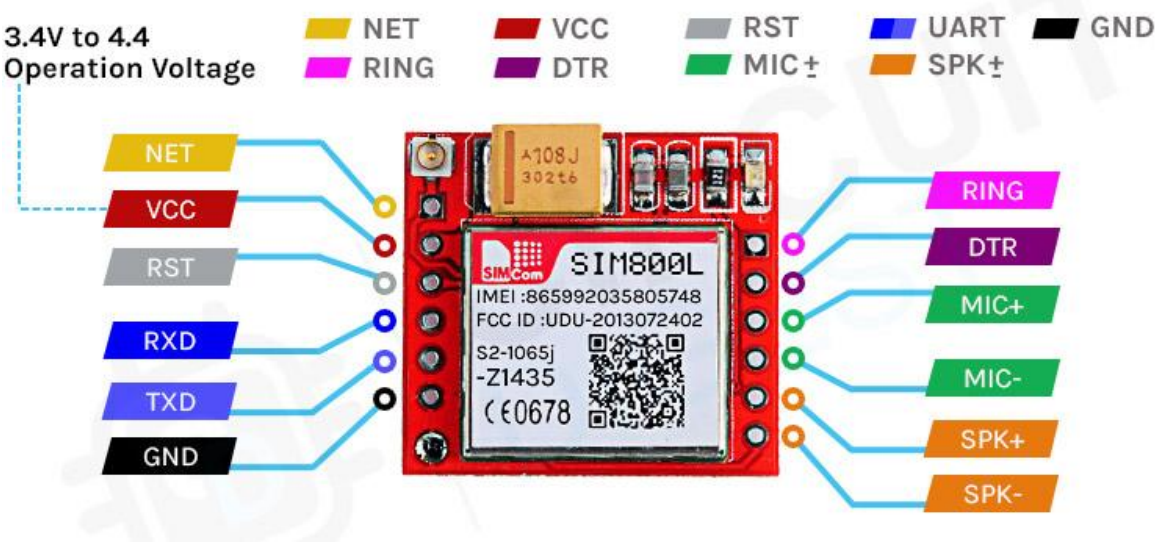



**Fig 4.8: SIM800L GPRS GSM Module**

## Pin Configuration:

| pin | Description |
|---|---|
| NET | is a pin where you can solder the helical antenna that comes with the module. |

| | |
|---|---|
| VCC | is the Power supply pin of the module and it needs to be powered anywhere from 3.4V to 4.4 volts. Connecting this module to a 5V supply will most likely destroy it and if you connect it to 3.3V it won't even run. A lithium battery or a buck converter with 2A current capacity is recommended for this module. |
| RST | is the hard reset pin of the sim800L module. If you are having trouble communicating with this, pull the pin low for 100ms. |
| RXD | is the RX pin for the module used in serial communication. |
| TXD | is the TX pin for this module used in Serial communication. |
| GND | is the Ground pin for this module; connect this pin to the Ground pin of the ESP32. |
| RING | is the ring indicator pin of the module. This pin generally is active high. It will go low for 120ms to indicate incoming calls and can also be configured to pulse when an SMS is received. |
| DTR | this pin can be used to put the module in sleep mode. Pulling the pin high puts the module in sleep mode and disables the serial**.** Pulling it low will wake the module up |
| MIC+- | These two pins can be used to connect an external microphone to the module. |
| SPK+- | these two pins can be used to connect an external speaker to the module. |

## 4.9 MX1508 Dual H-Bridge DC PWM Stepper Motor Driver

The MX1508 Motor Driver Module is a compact and efficient dual H-Bridge driver used for controlling DC motors and small stepper motors. In this project, it is used to drive the exhaust fan or any other small DC motor to help ventilate the area in case of a gas leak.

It enables the Arduino to control the direction and speed of a DC motor using PWM (Pulse Width Modulation) signals, making it a vital component for real-time safety response in the gas detection system.

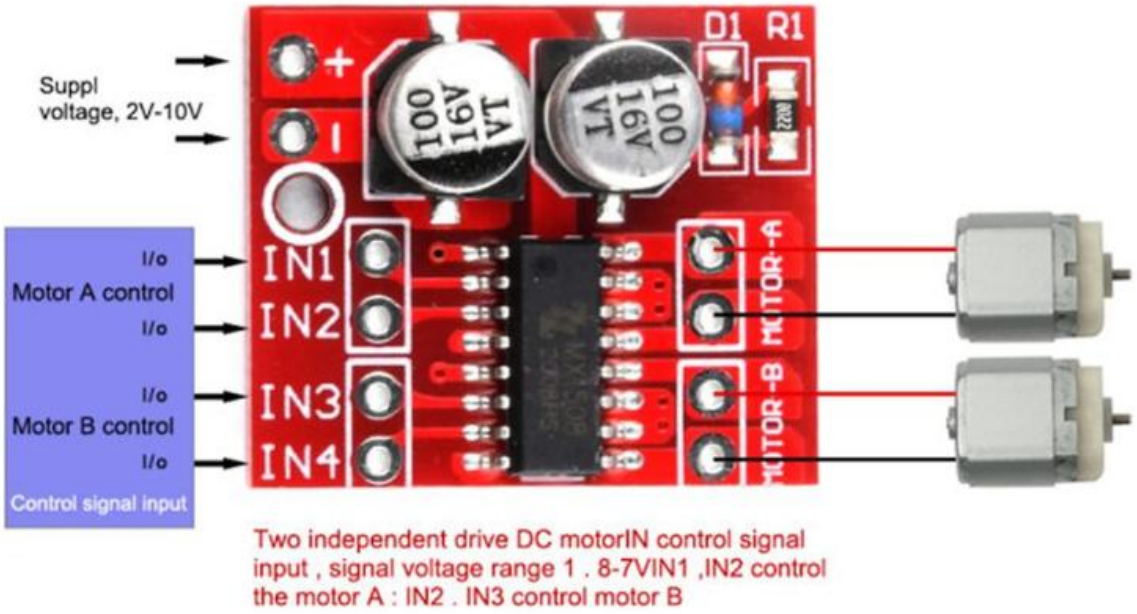



**Fig 4.9:MX1508 Dual H-Bridge DC PWM Stepper Motor Driver**

## Working Princ

The MX1508 receives logic signals from the Arduino and uses an H-Bridge configuration to control the flow of current through the motor. This allows for:

- Bidirectional control (clockwise and counterclockwise rotation)
- Speed variation via PWM
- Simultaneous control of two motors (dual-channel)

## Key Features:

- **Input Voltage:** 2V – 10V DC
- **Logic Voltage:** 2V – 6V
- **Output Current:** 1.5A continuous per channel (up to 2.5A peak)
- **Control Inputs:** PWM and direction control pins for each motor
- **Two Motor Channels (A & B):** Can drive two DC motors or one stepper motor Compact and lightweight, ideal for embedded applications

## Pin Configuration:

| Pin | Function |
|---|---|
| IN1, IN2 | Control motor A (direction & speed) |

| | |
|---|---|
| IN3, IN4 | Control motor B (direction & speed) |
| VCC | Motor power supply input (2V–10V) |
| GND | Ground |
| OUT1, OUT2 | Motor A output terminals |
| OUT3, OUT4 | Motor B output terminals |

## Applications in the Project:

- **Exhaust Fan Control:** Automatically activates to disperse gas during a leak
- **Ventilation System Integration:** Ensures airflow in hazardous conditions
- **Safe Shutdown or Response Mechanism:** Controlled by Arduino based on sensor input

## 4.10 Exhaust Fan:

The exhaust fan is an essential component in the gas leakage detection system, designed to ventilate the area and expel accumulated gas in the event of a leak. It acts as a secondary safety measure alongside the gas shutoff mechanism, helping to reduce the risk of fire or explosion by removing flammable gases from enclosed spaces.

**Fig 4.10:Exhaust Fan**

## Working Principle:

The fan is controlled by the Arduino Uno through the MX1508 Dual H-Bridge Motor Driver, which allows the microcontroller to switch the fan ON or OFF and control its speed via PWM signals. When the gas concentration exceeds a defined threshold (detected by the MQ-2 sensor), the fan is automatically activated to clear the air.

## Key Features:

- **Voltage Rating:** Typically operates at 5V or 12V DC (based on fan type)
- Low Power Consumption
- Compact and easy to mount
- Provides rapid air circulation to reduce gas buildup
- Controlled via Arduino through motor driver (MX1508)

## Applications in the Project:

- **Automatic Ventilation:** Removes leaked gases to lower concentration quickly
- **Fire Hazard Prevention:** Reduces flammable environment inside confined areas
- **Improved Safety:** Supports the gas valve shutoff by reducing risk of ignition
- **Triggered Response:** Operates only when gas leak is detected, conserving energy

# CHAPTER 5

# SOFTWARE DESCRIPTION AND HARDWARE RESULTS

## 5.1 Arduino IDE:

The Arduino Integrated Development Environment (IDE) is an open-source software application that facilitates the writing, editing, compiling, and uploading of code to Arduino-compatible microcontroller boards. It plays a critical role in the development of embedded systems, allowing users to program physical computing platforms with ease, whether for hobbyist projects, educational purposes, research prototypes, or industrial automation.

### 5.1.1.Overview and Purpose

The primary purpose of the Arduino IDE is to provide a user-friendly and accessible programming environment for interacting with Arduino hardware. It abstracts many of the complexities traditionally associated with embedded system development and provides a simplified interface for writing C/C++ based code. The IDE is designed to support rapid prototyping and iterative testing, essential for electronic system design and implementation.

### 5.1.2. Key Features

- **Cross-platform support**: Compatible with Windows, macOS, and Linux.
- **Built-in code editor**: Includes features such as syntax highlighting, auto-formatting, and error highlighting to improve coding efficiency.
- **One-click compilation and uploading**: The IDE simplifies the compilation of code and its uploading to the connected Arduino board via a USB interface.
- **Serial Monitor**: A built-in terminal that enables real-time communication between the computer and the microcontroller, useful for debugging and data monitoring.
- **Library Manager**: Easily import and manage libraries that extend functionality (e.g., for sensors, communication protocols, displays).
- **Board and Port Selector**: Automatically detects connected boards and available COM ports for seamless interaction.

### 5.1.3. Programming Language

The Arduino IDE primarily uses a dialect of **C/C++**, made accessible through a simplified structure. Every Arduino program (known as a "sketch") includes at least two essential functions:

- setup(): Initializes variables, pins, and configurations. It runs once at the beginning of the program.
- loop(): Contains the core logic and runs continuously after setup().

The simplicity of this structure makes Arduino an excellent platform for beginners, while still offering advanced capabilities for experienced developers.

### 5.1.4. Architecture and Workflow

The typical workflow in the Arduino IDE follows these steps:

1. **Write Code**: Using the built-in editor, users write or modify a sketch.
2. **Verify/Compile**: The IDE compiles the sketch into binary format using the **avr-g++** compiler and related toolchain.
3. **Upload**: The binary is uploaded to the microcontroller via a USB-to-serial interface using the **avrdude** uploader.
4. **Monitor**: Optional use of the serial monitor to view sensor outputs, debug messages, or interactive input/output.

### 5.1.5.Board Support and Compatibility

The IDE supports a wide range of official and third-party boards, including but not limited to:

- Arduino Uno, Mega, Nano, Leonardo,Arduino Due, Zero
- ESP8266, ESP32 (via additional board manager URLs)
- ATtiny series
- STM32-based boards (via third-party support)

The flexibility to include additional boards and cores makes the Arduino IDE versatile for various microcontroller platforms beyond the default Arduino boards.

### 5.1.6. Extensions and Integrations

- **Arduino CLI**: For command-line interface and advanced automation or CI/CD integration.
- **PlatformIO and Visual Studio Code**: For more advanced development needs, developers may use Arduino libraries within these more feature-rich environments.
- **IoT Integration**: Support for IoT platforms like Blynk, ThingSpeak, and Arduino IoT Cloud.

### 5.1.7. Use in Academic and Industry Projects

The Arduino IDE is widely used in academic institutions for teaching embedded systems, electronics, and IoT. Its straightforward learning curve, extensive documentation, and global community support make it a preferred tool for students and researchers. In industry, Arduino is often used for prototyping, proof-of-concept development, and automation applications in fields like robotics, smart agriculture, wearable technology, and home automation.

## 5.2 Component Interfacing

The Advanced LPG Gas Leakage Detection System depends on the proper and efficient interfacing of all hardware components with the Arduino UNO. The microcontroller acts as the brain of the system, interpreting sensor data and controlling various output devices based on real-time readings. Each component is connected to a specific pin on the Arduino, and their seamless interaction ensures the system's quick response to any LPG leakage.

- **MQ-5 Gas Sensor:** The MQ-5 gas sensor, responsible for detecting LPG gas concentration in the environment, is interfaced with the Arduino UNO through its analog output pin connected to the analog pin A0 of the Arduino. The sensor continuously outputs a voltage level that corresponds to the amount of gas present. The Arduino reads this voltage from pin A0 and compares it to a calibrated threshold. This analog interfacing allows for real-time monitoring of gas levels, and when the concentration exceeds the set threshold, it triggers the Arduino to initiate the alert and response process.

- **SIM900A GSM Module:** The SIM900A GSM module is interfaced with the Arduino UNO using software serial communication to send SMS alerts in the event of gas leakage. The module's TX pin is connected to digital pin 7 of the Arduino, and its RX pin is connected to digital pin 8. These pins are defined in the code using the SoftwareSerial library, which enables serial communication on digital pins other than the default hardware serial. The Arduino sends AT commands via pin 7 (TX) to the GSM module, and receives responses through pin 8 (RX). This setup ensures reliable message delivery to a registered mobile number, even when the user is not physically near the device.
- **Servo Motor:** The servo motor, used to automatically turn off the gas supply during leakage, is interfaced to digital PWM pin 9 of the Arduino UNO. Using the Servo library in the Arduino IDE, the microcontroller sends PWM signals to this pin to control the angle of rotation. Upon gas detection, the Arduino instructs the servo motor connected to pin 9 to rotate to a predefined angle, closing the valve that controls gas flow. This safety feature adds a mechanical layer of protection to the system, preventing further leakage and minimizing risk.
- **Buzzer:** The buzzer, which provides an immediate audible alert when gas is detected, is connected to digital pin 10 of the Arduino UNO. The Arduino sends a HIGH signal to pin 10 to activate the buzzer when the gas concentration exceeds the threshold. This pin remains LOW during normal conditions and switches to HIGH in emergency situations. The buzzer emits a loud sound that serves as a warning to individuals in the area, alerting them to evacuate or take safety measures immediately.
- **Exhaust Fan:** The exhaust fan is controlled through a relay module that enables the Arduino to switch the fan on or off indirectly. The signal pin of the relay module is connected to digital pin 11 of the Arduino UNO. When gas leakage is detected, the Arduino sends a HIGH signal to pin 11, triggering the relay to complete the fan's power circuit, which is connected to an external power source. This causes the fan to activate and ventilate the room, dispersing the accumulated gas and helping to reduce the concentration of LPG in the air.
- **Power Supply:**All components are powered through a 5V DC adapter or a USB to DC jack cable, supplying power to the Arduino UNO through its power input port. The Arduino then provides regulated 5V and ground connections to various components via its 5V and GND pins. Components such as the MQ-6 gas sensor,

GSM module, buzzer, and servo motor draw power from these regulated output pins. Special care is taken to provide adequate current to components like the GSM module and exhaust fan to ensure smooth operation, avoiding voltage fluctuations and system resets.

## 5.3 Interfacing the MQ-5 Gas Sensor

The MQ-5 gas sensor is widely used for detecting gases such as LPG, natural gas, and coal gas. It operates by varying its resistance in response to different gas concentrations, providing an analog output proportional to the detected gas level. To interface the MQ-5 sensor with a microcontroller like Arduino, the sensor's VCC and GND pins are connected to the 5V and GND of the microcontroller, respectively. The analog output (AOUT) pin is connected to an analog input pin of the microcontroller, allowing continuous monitoring of gas concentration levels.

Additionally, the sensor module provides a digital output (DOUT) that can be used for threshold-based detection, where the onboard potentiometer is adjusted to set the desired gas concentration limit. In the system setup, the microcontroller reads the analog values from the sensor, processes the data, and can trigger actions such as alarms or exhaust fans when gas levels exceed safe limits. The sensor requires a preheating period of around 20–30 seconds after powering on to ensure stable and accurate readings. Proper calibration in clean air is recommended to achieve precise measurements.

## 5.4 Interfacing the SIM800L GSM Module

The SIM800L GSM module is a compact and low-power device used for enabling GSM communication in embedded systems. It supports functionalities such as sending and receiving SMS, making and receiving calls, and connecting to GPRS for internet data transmission. To interface the SIM800L with a microcontroller like Arduino, the module's VCC pin is connected to a stable 3.7V–4.2V power supply (often provided via a separate regulator, as the module is sensitive to higher voltages), and the GND is connected to the common ground.

The communication between the microcontroller and the SIM800L module is established through UART (Universal Asynchronous Receiver/Transmitter) using the TX and RX pins.

The TX pin of the module is connected to the RX pin of the microcontroller and vice versa. Proper voltage level shifting may be necessary to ensure reliable data exchange if the microcontroller operates at 5V logic levels. AT commands are used to control the SIM800L module for operations like sending SMS, making calls, and checking network status. Stable power and a proper antenna connection are crucial for maintaining a reliable GSM network connection.

## 5.5 Interfacing the 16x2 LCD Display

The 16x2 LCD display is a widely used module in embedded systems for displaying alphanumeric characters. It consists of two rows with 16 characters each and operates based on the HD44780 controller. To interface the 16x2 LCD with a microcontroller such as Arduino, either a 4-bit or 8-bit data communication mode can be used. In 4-bit mode, only four data lines (D4–D7) are used along with control pins (RS, RW, and EN), which reduces the number of GPIO pins required.

The RS (Register Select) pin is used to select between command and data registers, RW (Read/Write) pin determines the operation mode (usually grounded for write-only operations), and the EN (Enable) pin is used to latch the data into the display. A potentiometer is typically connected to the V0 pin to adjust the display contrast. The VCC and GND pins provide the necessary power supply, usually 5V. Libraries like "LiquidCrystal" in Arduino IDE simplify the coding process by providing built-in functions for initializing the display and printing characters. Proper initialization and timing delays are essential for correct display operation.

## 5.6 Interfacing the Servo Motor

A servo motor is a rotary actuator that allows precise control of angular position, speed, and acceleration. It is widely used in embedded systems for applications requiring controlled movement, such as robotics, automation, and positioning systems. A standard servo motor typically has three connections: power (VCC), ground (GND), and a control signal (PWM input). To interface a servo motor with a microcontroller like Arduino, the VCC is connected to a 5V power supply, GND to the common ground, and the control wire to a PWM-capable digital pin.

The position of the servo motor shaft is controlled by sending Pulse Width Modulated (PWM) signals, where the width of the pulse determines the angle of rotation. Standard servos generally operate between 0° and 180°, with a pulse width range typically between 1 ms (0°) and 2 ms (180°) at a 50 Hz frequency. In Arduino, libraries like "Servo.h" simplify the process of generating PWM signals, allowing easy control over the motor's angle with simple commands. It is important to ensure that the servo receives adequate and stable power, especially for larger servos that may require external power sources.

## 5.7 Interfacing the Buzzer

A buzzer is a simple audio signaling device commonly used in embedded systems to provide audible alerts, alarms, or notifications. It typically has two pins: one for power (positive) and one for ground (negative). To interface a buzzer with a microcontroller like Arduino, the positive pin is connected to a digital output pin, and the negative pin is connected to the ground. The buzzer can be controlled by setting the output pin HIGH to turn it ON and LOW to turn it OFF.Buzzers can be of two types: active and passive. An active buzzer produces sound when powered with a DC voltage, while a passive buzzer requires a PWM signal to generate different tones. Libraries or direct digitalWrite() commands can be used to manage buzzer operations. Proper current-limiting resistors may be used if required to prevent overcurrent damage to the microcontroller or buzzer.

## 5.8 Interfacing the MX1508 Dual H-Bridge Motor Driver

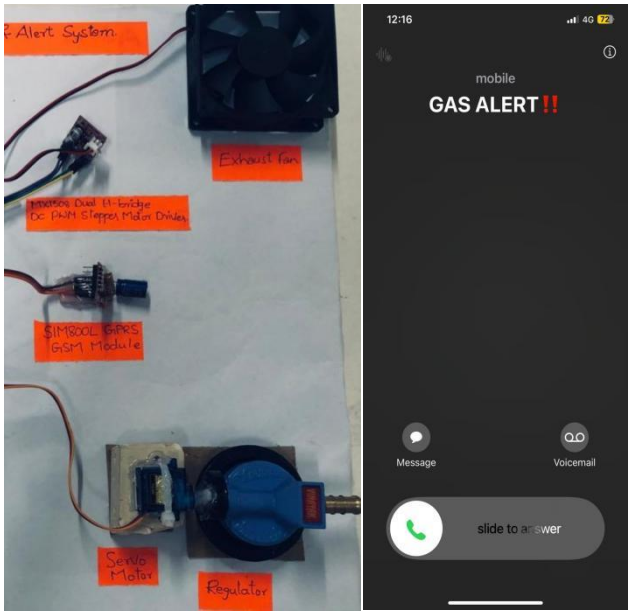
The MX1508 dual H-bridge motor driver module is designed to control the speed and direction of two DC motors independently. It allows bidirectional control by using an H-bridge circuit arrangement and can drive motors with operating voltages typically between 2V and 10V. To interface the MX1508 module with a microcontroller such as Arduino, each motor requires two input pins: IN1 and IN2 for Motor A, and IN3 and IN4 for Motor B.By setting the input pins HIGH or LOW, the motor can be made to rotate forward, reverse, or stop. PWM signals can be applied to the input pins to control the motor speed effectively. The motor power supply (VCC) and ground (GND) are connected to an external power source, while the logic control inputs are connected to the digital output pins of the microcontroller. The MX1508 driver provides a simple, efficient, and compact solution for small to medium-sized motor control applications.

## 5.9 HARDWARE RESULTS

The developed Gas Leakage Detection System was successfully implemented and tested under various simulated conditions. The system demonstrated accurate and reliable detection of gas leakage by the MQ-2 sensor. Upon detection, the following actions were observed:

- **Immediate Alerts**:The system triggered an audible alarm through the buzzer as soon as the gas concentration exceeded the preset threshold. This ensured immediate attention from nearby individuals.
- **Automatic Safety Measures:**The servo motor-controlled valve automatically closed the gas supply when a leak was detected, preventing further gas flow and minimizing potential hazards.
- **GSM Notifications:**The SIM800L GSM module successfully sent alert messages to the pre-registered mobile numbers. It also initiated automated emergency calls, effectively ensuring remote awareness and quicker response.
- **Exhaust Fan Activation:**The MX1508 motor driver module activated the exhaust fan, which helped in dispersing the leaked gas and improving the surrounding air quality.

Overall, the system met its primary objectives by ensuring early detection, timely notification, and effective preventive measures against gas leakage. It proved to be a cost-effective, efficient, and scalable solution for enhancing household and industrial safety.



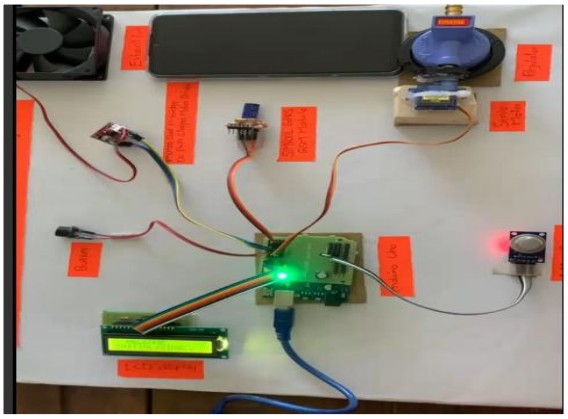



**Fig 5.9: Phone Call ALERT**

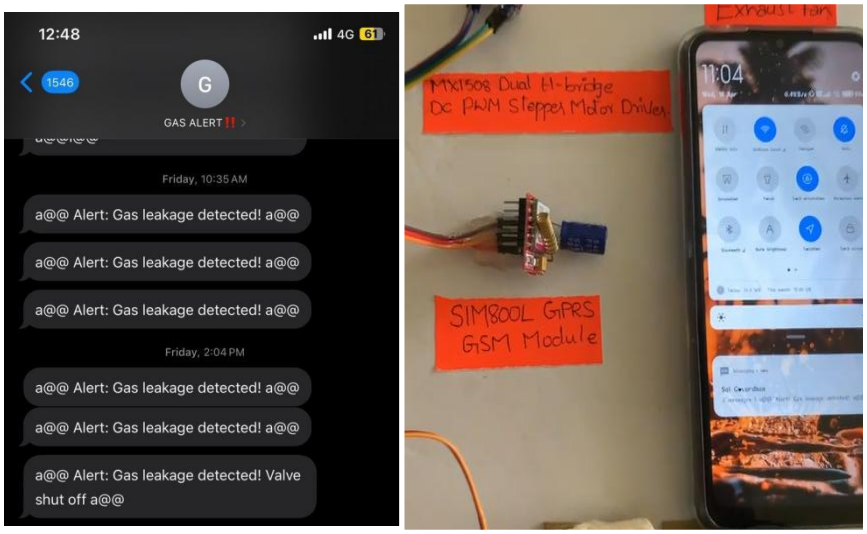



**Fig5.9:SMS Alert**

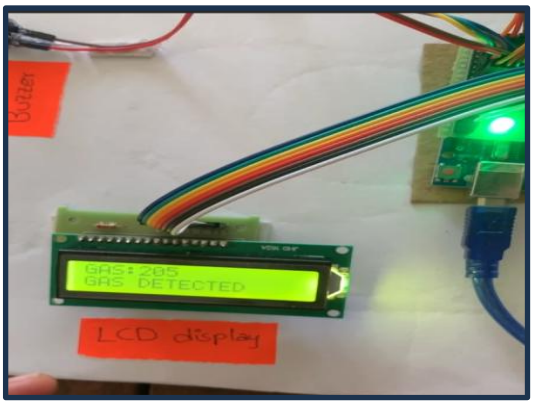



**Fig5.9: LCD Display Showing Gas Detection Alert"**

# CHAPTER-6

# CONCLUSIONS AND FUTURE SCOPE

## Conclusion:

- The Gas Leakage Detection System was successfully designed, developed, and tested to address safety concerns related to gas leaks. By integrating key components such as the MQ-2 gas sensor, 16x2 LCD display, SIM800L GSM module, buzzer, servo motor, and exhaust fan, the system provided an effective, real-time solution for detecting gas leaks and alerting users promptly.
- The automatic closure of the gas valve and the activation of the exhaust fan further minimized risks associated with gas accumulation.
- The project demonstrated reliability, quick response time, low cost, and ease of implementation, making it a viable safety solution for homes, industries, and commercial spaces.

Thus, the objectives of early detection, quick notification, and preventive action were effectively achieved.

## Future Scope:

- **Integration with IoT Platforms:** Future versions can integrate with IoT platforms to monitor gas levels remotely through a mobile app or cloud dashboard.
- **Multiple Gas Detection:**The system can be enhanced to detect multiple gases such as carbon monoxide (CO), methane ($CH_4$), and smoke for broader safety coverage.
- **Advanced AI-Based Analysis:** AI and Machine Learning algorithms can be added to predict leak patterns and detect early signs of faults in gas pipelines.
- **Compact PCB Design:** Future work could involve designing a compact PCB to reduce the overall size of the system, making it more suitable for commercial production.
- **Battery Backup Support:** Adding a rechargeable battery backup would ensure the system remains functional during power outages.

# APPENDICES

## CODE:

```
#include <Servo.h>

#include <LiquidCrystal.h>

Servo myservo;

const int rs = 8, en = 9, d4 = 10, d5 = 11, d6 = 12, d7 = 13;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

// Pin Declarations

int buz = 5;

int gs = A0;

int rly = 3;

int rly1 = 4;

int sts = 0;

int threshold = 0;

bool valveClosed = false;

String phoneNumbers[] = {

  "8309141036",

  "9346778366",

  "9959827390"

};
```

```
int totalNumbers = sizeof(phoneNumbers) / sizeof(phoneNumbers[0]);

void setup() {
  Serial.begin(9600);
  lcd.begin(16, 2);
  myservo.attach(6);
  lcd.print("WELCOME");
  lcd.setCursor(0, 1);
  lcd.print("INITIALIZING...");
  delay(3000);
  pinMode(buz, OUTPUT);
  pinMode(rly, OUTPUT);
  pinMode(rly1, OUTPUT);
  digitalWrite(rly, LOW);
  digitalWrite(rly1, LOW);
  myservo.write(0);
  // Sensor calibration
  lcd.clear();
  lcd.print("Calibrating...");
  int sum = 0;
  for (int i = 0; i < 100; i++) {
    sum += analogRead(gs) / 2;
```

```
    delay(50);
  }
  int avg = sum / 100;
  threshold = avg + 100;
  lcd.setCursor(0, 1);
  lcd.print("Thresh: " + String(threshold));
  delay(3000);
}
void loop() {
  int gval = analogRead(gs) / 2;
  lcd.clear();
  lcd.print("GAS: " + String(gval));
  // Check for incoming SMS command
  if (Serial.available()) {
    String incoming = Serial.readStringUntil('\n');
    incoming.trim();
    incoming.toUpperCase();
    if (incoming.indexOf("ON") >= 0 && !valveClosed) {
      close_valve(); // Close valve if not already closed
      send_sms();     // Alert after manual shutdown
    } }
```

```
if (gval > threshold) {
  digitalWrite(buz, HIGH);
  lcd.setCursor(0, 1);
  lcd.print("GAS DETECTED");
  if (sts == 0 && !valveClosed) {
    digitalWrite(rly, HIGH);
    close_valve();  // Valve closed
    send_sms();     // Alert sent AFTER valve close
    send_call();
    digitalWrite(buz, LOW);
  }
  sts = 1;
}
else if (gval < threshold - 50) {
  digitalWrite(rly, LOW);
  digitalWrite(buz, LOW);
  lcd.setCursor(0, 1);
  lcd.print("SAFE LEVEL");
  sts = 0;
  valveClosed = false; // Reset for next detection
}
```

```
 delay(500);
}
void close_valve() {
  for (int i = 0; i <= 90; i++) {
    myservo.write(i);
    delay(15);
  }
  valveClosed = true;
}
void send_sms() {
  for (int i = 0; i < totalNumbers; i++) {
    Serial.println("AT");
    delay(1000);
    Serial.println("ATE0");
    delay(1000);
    Serial.println("AT+CMGF=1");
    delay(1000);
    Serial.print("AT+CMGS=\"");
    Serial.print(phoneNumbers[i]);
    Serial.println("\"");
    delay(1000);
```

```
    Serial.println("⚠ Alert: Gas leakage detected! Valve shut off ⚠");
    delay(500);
    Serial.write(26);
    delay(5000);
  }
}
void send_call() {
  for (int i = 0; i < totalNumbers; i++) {
    Serial.print("ATD");
    Serial.print(phoneNumbers[i]);
    Serial.println(";");
    delay(20000);
    Serial.println("ATH");
    delay(1000);
  }
}
```

# REFERENCES

**[1] Dewi, L., & Somantri, Y. (2018). "Wireless Sensor Network on LPG Gas Leak Detection and Automatic Gas Regulator System Using Arduino". *IOP Conference Series: Materials Science and Engineering*, 384(1), 012064.**

**[2] Ubale, G., Dere, G., Derkar, M., Deore, V., Depale, N., & Deore, K. (2023). "Gas Leakage Detector Using GSM and Arduino". *International Journal of Scientific Development and Research (IJSDR)*, 8(11), 641-643.**

**[3] Tommy, A. (2022). "Implementation of a Gas Leakage Detection System Using the MQ-6 Sensor". *Brilliance: Research of Artificial Intelligence*, 2(1), 17-20.**

**[4] Alao, P.O., Raji, A.A., Olajide, M.B., Jagun, Z.O., Okubanjo, A.A., & Oyedeji, A.O. (2024). "Gas Leakage Detection System for Domestic and Industrial Applications". *Scientia Africana*, 23(3), 231-240.**

**[5] Thomas, W.T.H.M., Doraiswamy, A.S., Nanda, C.B., Reddy, G.B., & Kumar, H.P. (2024). "LPG Gas Leakage Detection System with Auto Cutoff Regulator using Arduino". *International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)*, 4(6), 321-327.**