# DEVELOPMENT OF IOT ENABLED FRAMEWORK FOR LPG LEAKAGE DETECTION AND WEIGHT MONITORING SYSTEM

*Submitted to Jawaharlal Nehru Technological University, Kakinada in partial fulfillment of requirement for the degree of*

### Bachelor of Technology

in

### ELECTRONICS AND COMMUNICATION ENGINEERING

*Submitted by*

### K.Harika

(20KN1A0452)

|  |  |  |
| --- | --- | --- |
| **B.Pranitha** | **Ch.Nagalakshmi** | **K.Lakshmi Aparna** |
| (20KN1A0414) | (20KN1A0433) | (20KN1A0454) |

Under the guidance of

**Mr. SYED USMAN,** M.Tech

Assistant Professor



**DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING**

### NRI INSTITUTE OF TECHNOLOGY

**Autonomous**

##### Approved by AICTE, New Delhi, Permanently Affiliated to JNTUK, Kakinada Accredited by NBA (ECE, CSE & EEE), Accredited by NAAC with ‘A’ Grade ISO 9001: 2015 Certified Institution

**Pothavarappadu (V), Agiripalli (M), Eluru (Dt)-521212 2020-2024**

# Department of

**ELECTRONICS & COMMUNICATION ENGINEERING**



**CERTIFICATE**

This is to certify that the project report entitled "**DEVELOPMENT OF IOT ENABLED FRAMEWORK FOR LPG LEAKAGE DETECTION AND WEIGHT MONITORING**

**SYSTEM"** that is being submitted by

|  |  |
| --- | --- |
| K.Harika | -20KN1A0452 |
| B.Pranitha | -20KN1A0414 |
| Ch.Nagalakshmi | -20KN1A0433 |
| K.Lakshmi Aparna | -20KN1A0454 |

in partial fulfillment for the award of **Bachelor of Technology** in **Electronics &Communication Engineering** to the **Jawaharlal Nehru Technological University, Kakinada** is a record of bonafide work carried out by them under my guidance and supervision. The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma.

|  |  |
| --- | --- |
| **Mr.SYED USMAN** | **Dr. R. SUNITHA** |
| **Assistant Professor** | **Professor&Head** |

**External Examiner**

# ACKNOWLEDGEMENT

We are very much grateful to **Mr.SYED USMAN**, **Assistant Professor**, Electronics & Communication Engineering, for his inspiring guidance and for providing the background knowledge to deal with the problem at every phase of our project in a systematic manner.

We have immense pleasure in expressing our thanks and deep sense of gratitude to **Dr. R.Sunitha M.Tech, Ph.D**, Head of the Department, Electronics & Communication Engineering, for providing necessary facilities for the completion of project.

We are very much indebted to **Dr. V. Srinivasarao, M.Tech, Ph.D, Dean Academics of ECE Department** for his critical analysis during the reviews and useful suggestions towards successful completion of the project work.

We are grateful to our beloved **PRINCIPAL, Dr. NAGABHASKAR, M. Tech., Ph.D., FIE, CE, MISTE, MCMSI** for his consistent encouragement and guidance.

We express our heartfelt gratitude to **Dr. R. VENKATARAO**, Chairman and

**Sri K. SREEDHAR,** Secretary, of **NRI INSTITUTE OF TECHNOLOGY** for their patronage and parental hood in achieving excellence in our academic pursuit.

We would like to express our sincere and heartfelt thanks to the faculty of our department for their continuous cooperation, which has given us the ultimate confidence in successful completion of this project.

**Project Associates**

|  |  |
| --- | --- |
| **K.Harika** | - **20KN1A0452** |
| **B.Pranitha** | - **20KN1A0414** |
| **Ch.Nagalakshmi** | - **20KN1A0433** |
| **K.Lakshmi Aparna** | - **2OKN1A0454** |

# DECLARATION

We the members of the project"**DEVELOPMENT OF IOT ENABLED FRAMEWORK FOR LPG LEAKAGE DETECTION AND WEIGHT**

**MONITORING SYSTEM** hereby declare that the matter embodied in this project is the genuine work done by us and has not been submitted either to this University or to any other University/Institute for the fulfillment of the requirement of any other course of study.

**Project Associates**

|  |  |
| --- | --- |
| **K.Harika** | - **20KN1A0452** |
| **B.Pranitha** | - **20KN1A0414** |
| **Ch.Nagalakshmi** | - **20KN1A0433** |
| **K.Lakshmi Aparna** | - **20KN1A0454** |

# TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT** | **i** |
|  | **LIST OF FIGURES** | **ii-iv** |
|  | **LIST OF TABLES** | **v** |
| **CHAPTER 1** | **INTRODUCTION** | **1** |
|  | 1.1 INTRODUCTION | 1 |
|  | 1.2 EMBEDDED SYSTEM IMPLEMENTATION | 2 |
|  | 1.2.1 EMBEDDED SYSTEM HARDWARE | 4 |
|  | 1.2.2 EMBEDDED SYSTEM SOFTWARE | 5 |
|  | 1.2.3 APPLICATIONS | 6 |
|  | 1.2.4 IMPLEMENTATION FLOW | 7 |
| **CHAPTER 2** | **LITERATURE SURVEY** | **8** |
|  | 2.1 LITERATURE SURVEY | 8 |
|  | 2.2 EXISTING SYSTEM | 11 |
|  | 2.3 DRAWBACKS IN EXISTING SYSTEM | 11 |
| **CHAPTER 3** | **PROPOSED SYSTEM** | **12** |
|  | 3.1 PROPOSED METHOD | 12 |
|  | 3.2 BLOCK DIAGRAM | 13 |
|  | 3.3ADVANTAGES | 13 |
|  |  | 14 |
|  | 3.4 APPLICATIONS |  |

[CHAPTER 4 HARDWARE REQUIREMENTS 15](#_bookmark0)

* 1. [ARDUINO 15](#_bookmark1)
     1. [INTRODUCTION TO ARDUINO 16](#_bookmark2)
     2. [FEATURES OF ARDUINO 17](#_bookmark3)
     3. [ARDUINO PINOUT 18](#_bookmark4)
     4. [PIN DESCRIPTION 19](#_bookmark5)
     5. ARDUINO UNO TECHNICAL 20

SPECIFICATIONS

* + 1. COMMUNICATION & PROGRAMMING 21
    2. [APPLICATIONS 21](#_bookmark6)
  1. [LCD 22](#_bookmark7)
     1. EXECUTION OF LCD **22**
     2. [IMAGES OF LCD DISPLAY 23](#_bookmark8)

24

* + 1. PIN DIAGRAM

24

* + 1. PIN DESCRIPTION

26

* + 1. LCD COMMANDS

27

* + 1. BLOCK DIAGRAM OF LCD DISPLAY 27
    2. CONTROL AND DISPLAY COMMAND 27

|  |  |
| --- | --- |
| 4.3 RELAY | 30 |
| 4.3.1 PIN DIAGRAM | 30 |
| 4.3.2 RELAY DESIGN | 31 |
| 4.3.3 RELAY BASICS | 32 |
| 4.3.4 POLE AND THROW | 33 |
| 4.3.5 APPLICATIONS | 34 |
| 4.3.6 APPLICATION OF OVERLOAD RELAY | 34 |
| 4.3.7 RELAY SELECTION | 34 |
| 4.4 POWER SUPPLY | **35** |
| 4.4.1 RECTIFIER | 35 |
| 4.4.2 CAPACITOR | 36 |
| 4.4.3 VOLTAGE REGULATORS | 36 |
| 4.4.4 BUZZER | 37 |
| 4.4.4.1 PIN CONFIGURATION | 37 |
| 4.4.4.2 FEATURES AND SPECIFICATIONS | 37 |
| 4.4.4.3 APPLICATIONS | 38 |
| 4.4.5 MQ2 SENSOR | 38 |
| 4.4.5.1 INTERNAL FEATURES | 39 |
| 4.4.6 GSM/GPRS | 41 |
| 4.4.6.1 GPRS MODULE | 42 |

|  |  |  |
| --- | --- | --- |
|  | 4.4.6.2 SYNTAX OF AT COMMANDS | 43 |
|  | 4.4.7 CPU FAN | 47 |
|  | 4.4.8 LOAD CELL | 47 |
|  | 4.4.8.1 HX711 LOAD CELL AMPLIFIER | 48 |
| **CHAPTER 5** | **SOFTWARE REQUIREMENTS** | **49** |
|  | 5.1 ARDUINO IDE | 49 |
|  | 5.2 INTRODUCTION TO ARDUINO IDE | 49 |
|  | 5.2.1 HOW TO INSTALL ARDUINO IDE | 49 |
|  | 5.3 LIBRARIES | 53 |
|  | 5.4 MAKING PINS INPUT & OUTPUT | 54 |
|  | 5.4.1 HOW TO SELECT THE BOARD | 54 |
|  | 5.4.2 BOOT LOADER | 55 |
| **CHAPTER 6** | **CODE AND RESULTS** | **56** |
|  | 6.1 CODE | 56 |
|  | 6.2 RESULTS | 62 |
| **CHAPTER 7** | **CONCLUSION** | **65** |
|  | **REFERENCES** | **66** |

# ABSTRACT

### DEVELOPMENT OF IOT ENAMBLED FRAMEWORK FOR LPG LEAKAGE DETECTION AND WEIGHT MONITORING SYSTEM

This project outlines the development of an Internet of Things (IoT)-enabled framework for an LPG leakage detection and weight monitoring system, designed to enhance the safety and efficiency of LPG usage in various environments. The core components of this framework include an Arduino controller, gas sensor, LED indicators, a buzzer for audible alerts, a GSM module for real-time notifications, and GPRS for sensor value uploading to the Thingspeak server. The primary goal of this project is to detect LPG gas leaks and monitor the weight of LPG cylinders, ensuring timely warnings and remote monitoring. The system collects data from the gas sensor and load cells, processes it using the Arduino controller, and communicates real-time information to users via SMS notifications and cloud-based data visualization on Thingspeak.

The IoT-enabled framework leverages the power of Arduino as the central processing unit, interfacing with a gas sensor capable of detecting LPG gas leaks with high precision. When a gas leak is detected, the system activates LED indicators and a buzzer, providing immediate visual and audible alerts to users in the vicinity. Moreover, the integration of a GSM module ensures that users receive SMS notifications, allowing for timely responses and safety measures. Additionally, the GPRS connectivity facilitates the seamless uploading of sensor values to the Thingspeak cloud server, where users can access real-time data and historical trends through web or mobile applications. This comprehensive project addresses the critical need for LPG safety by combining hardware and software components into a robust and user- friendly solution, ultimately minimizing the risks associated with gas leakage and ensuring the efficient management of LPG cylinder weight.

**Keywords: Arduino, GSM/GPRS, LPG leakage, sensors, Load cell**

# LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **FIG.NO** | **NAME OF THE FIGURE** | **PAGE NO** |
| 1.2.1 | Overview of Embedded system | 2 |
| 1.2.2 | Block diagram of Embedded system | 3 |
| 1.2.2.1 | Flow of burning source code to processor | 5 |
| 3.2.1 | Block Diagram of Proposed system | 13 |
| 4.1a | Types of Arduinos | 15 |
| 4.1b | Arduino Board | 16 |
| 4.1.c | Micro controller | 16 |
| 4.1.1.1 | Arduino UNO | 16 |
| 4.1.2.1 | Atmega328 Microcontroller | 17 |
| 4.1.3.1 | Arduino Uno Pinout | 18 |
| 4.2.2.1  4.2.2.2 | LCD Front view  LCD Back view | 23  23 |
| 4.2.3 | Pin Diagram of LCD | 24 |
| 4.2.6 | Block Diagram of LCD | 27 |
| 4.3 | Relay | 30 |
| 4.3.1 | Pin Diagram of Relay | 30 |
| 4.3.2.1 | Relay Construction | 31 |
| 4.3.2.2 | Relay Design | 31 |
| 4.3.3.1 | Relay Operation | 32 |
| 4.3.3.2 | Energized Relay | 32 |

|  |  |  |
| --- | --- | --- |
| 4.3.3.3 | De-Energized Relay | 33 |
| 4.4 | Power Supply | 35 |
| 4.4.1a | Circuit of Rectifier | 35 |
| 4.4.1b | Rectifier | 35 |
| 4.4.2 | Capacitor | 36 |
| 4.4.3a | 7805 Voltage Regulator With Pinout | 36 |
| 4.4.3b | 7812 Voltage Regulator With Pinout | 36 |
| 4.4.4 | Buzzer | 37 |
| 4.4.5a | Gas Sensor | 38 |
| 4.4.5b | Parts of a Gas Sensor | 39 |
| 4.4.5c | Steel Mash in Gas Sensor | 39 |
| 4.4.5.1a | Inside View of Gas Sensor | 39 |
| 4.4.5.1b | Hexapod Structure | 40 |
| 4.4.5.1c | Ceramic Sensing Element | 40 |
| 4.4.5.1d | Closr Look of Ceramic Element | 40 |
| 4.4.6 | GSM/GPRS | 41 |
| 4.4.7 | CPU Fan | 47 |
| 4.4.8.1 | HX711 Cell Amplifier | 48 |
| 5.2.1.1 | Arduino IDE | 50 |
| 5.2.1.2 | File Option | 50 |
| 5.2.1.3 | Prefernces | 51 |
| 5.2.1.4 | Hex File | 51 |

|  |  |
| --- | --- |
| 5.2.1.5  5.2.1.6  5.2.1.7  5.2.1.8  5.3  5.4.1.1  5.4.2  6.2.1  6.2.2  6.2.3  6.2.4  6.2.5  6.2.6  6.2.7  6.2.8  6.2.9 | Menu Tab 51  Output in Serial Monitor 52  Text Editor 52  Output Pane 53  Sketch Button 53  Port Section 54  Boot Loader 55  Working Model of The Project 62  Display of No Gas Leak And Normal Weight 63  Display of NO Gas Leak And Weight Decreases 63  Display of Gas Leak And Weight Decreases 63  Display of Gas Leak And Normal Weight 63  Display of Data Uploading 63  Display of Message Sent 63  GSM Result 63  Things Speak Result 64 |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **NAME OF THE TABLE** | **PAGE NO** |
| 4.1.5.1  4.2.4  4.2.4  4.2.7  4.4.4.1 | Arduino UNO technical specifications Pin Discription of LCD  LCD Commands  Control And Display Commands Pin Configuration of Buzzer | 20  24  26  27  37 |

# CHAPTER 1 INTRODUCTION

### INTRODUCTION

The project "DEVELOPMENT OF IOT ENABLED FRAMEWORK FOR LPG LEAKAGE

DETECTION AND WEIGHT MONITORING SYSTEM" focuses on enhancing safety and efficiency in managing liquefied petroleum gas (LPG) usage. LPG is widely used in households and industries for cooking, heating, and other purposes, but it poses potential risks due to leakage and improper monitoring. This project addresses these concerns by leveraging Internet of Things (IoT) technology to develop an advanced framework for detecting LPG leaks and monitoring the weight of LPG cylinders in real-time.

One of the key components of this framework is the integration of IoT devices with LPG cylinders. These devices, equipped with sensors, are capable of detecting even minor leaks of LPG gas, which can be crucial in preventing accidents and ensuring safety. Additionally, weight sensors are incorporated into the system to monitor the quantity of LPG remaining in the cylinder accurately. By continuously monitoring the weight, users can anticipate when a refill is required, thereby minimizing the risk of running out of gas unexpectedly.

Furthermore, the IoT-enabled framework facilitates remote monitoring and management of LPG cylinders. Through a centralized platform, users can access real-time data regarding LPG consumption, cylinder weight, and leak detection from anywhere, using their smartphones or computers. This remote accessibility not only enhances convenience but also enables prompt response to any detected leaks or anomalies. Overall, the project aims to provide a comprehensive solution for LPG management, promoting safety, efficiency, and peace of mind for users and stakeholders alike

### EMBEDDED SYSTEM IMPLEMENTATION Introduction:

An embedded system is one kind of a computer system mainly designed to perform several tasks like to access, process, and store and also control the data in various electronics-based systems. Embedded systems are a combination of hardware and software where software is usually known as firmware that is embedded into the hardware. One of its most important characteristics of these systems is, it gives the o/p within the time limits. Embedded systems support to make the work more perfect and convenient. So, we frequently use embedded systems in simple and complex devices too. The applications of embedded systems mainly involve in our real life for several devices like microwave, calculators, TV remote control, home security and neighborhood traffic controlsystems, etc.

User interface

Inputs

Software

Embedded system

Hardware

Output

Link to other systems

**Fig:1.2a Overview of Embedded system**

### Embedded system:

Embedded system includes mainly two sections, they are

1. Hardware
2. Software

Output devices interfacing

Interrupt controller

Parallel ports

Serial communication ports

Processor

Application specific circuits

Power supply and oscillator circuits

Memory

Timers

Input devices interfacing

and driver circuits

**Fig:1.2b Block diagram of Embedded system**

### EMBEDDED SYSTEM HARDWARE:

As with any electronic system, an embedded system requires a hardware platform on which it performs the operation. Embedded system hardware is built with a microprocessor or microcontroller. The embedded system hardware has elements like input output (I/O) interfaces, user interface, memory and the display. Usually, an embedded system consists of:

* + - * Power Supply
      * Processor
      * Memory
      * Timers
      * Serial communication ports
      * Output/Output circuits
      * System application specific circuits

Embedded systems use different processors for its desired operation. Some of the processors used are

1. Microprocessor
2. Microcontroller
3. Digital signal processor **Microprocessor vs. Microcontroller Microprocessor**
   * CPU on a chip.
   * We can attach required amount of ROM, RAM and I/O ports.
   * Expensive due to external peripherals.
   * Large in size
   * general-purpose

**Microcontroller**

* Computer on a chip
* fixed amount of on-chip ROM, RAM, I/O ports
* Low cost.
* Compact in size.
* Specific –purpose

### EMBEDDED SYSTEM SOFTWARE:

The embedded system software is written to perform a specific function. It is typically written in a high level format and then compiled down to provide code that can be lodged within a non- volatile memory within the hardware. An embedded system software is designed to keep in view of the three limits:

* + - * Availability of system memory
      * Availability of processor’s speed
      * When the system runs continuously, there is a need to limit power dissipation for events like stop, run and wake up.

**Bringing software and hardware together for embedded system:**

To make software to work with embedded systems we need to bring software and hardware together .for this purpose we need to burn our source code into microprocessor or microcontroller which is a hardware component and which takes care of all operations to be done by embedded system according to our code.

1. Each of the source files must be compiled or assembled into an object file.
2. All of the object files that result from the first step must be linked together to produce a single object file, called the re-locatable program.
3. Physical memory addresses must be assigned to the relative offsets within the re- locatable program in a process called relocation.

The result of the final step is a file containing an executable binary image that is ready to run on the embedded system.

Source code

Assembler

Linker

Locator

Executable file

Processor

**Fig 1.2.2.1:Flow of burning source code to processor**

### Applications:

Embedded systems have different applications. A few select [applications of embedded](https://www.elprocus.com/embedded-systems-real-time-applications/) [systems](https://www.elprocus.com/embedded-systems-real-time-applications/) are smart cards, telecommunications, satellites, missiles, digital consumer electronics, computer networking, etc.

[Embedded Systems in Automobiles](http://www.edgefx.in/importance-of-embedded-systems-in-automobiles-with-applications/)

* + - * Motor Control System
      * Engine or Body Safety
      * [Robotics](http://www.edgefx.in/top-list-robotics-projects-for-engineering-beginners/) in Assembly Line
      * Mobile and E-Com Access Embedded systems in Telecommunications
      * Mobile computing
      * Networking
      * [Wireless Communications](http://www.edgefx.in/multiple-input-and-multiple-output-mimo-wireless-communications/) Embedded Systems in Smart Cards
      * Banking
      * Telephone
      * [Security Systems](http://www.edgefx.in/microcontroller-based-projects-on-car-security-systems-using-gsm/)

### Implementation flow:

##### Stage 1:

* Considering the problems of existing methods and giving solution to that problem by considering the basic requirements for our proposed system

##### Stage 2:

* Considering the hardware requirement for the proposed system
* For this we need to select the below components:
* 1. Microcontroller
* 2. Inputs for the proposed system (ex: sensors, drivers etc..,)
* 3. Outputs (ex: relays, loads)

##### Stage 3:

* After considering hardware requirements, now we need to check out the software requirements. Based on the microcontroller we select there exists different software for coding, compiling, debugging. we need to write source code for that proposed system based on our requirements and compile, debug the code in that software .
* After completing all the requirements of software and hardware we need to bring both together to work our system. For this we need to burn our source code into microcontroller, after burning our source code to microcontroller then connect all input and output modules as per our requirement.

# CHAPTER 2 LITERATURE SURVEY

### LITERATURE SURVEY

##### James Doorhy, “Real-Time Pipeline Leak Detection and Location Using Volume Balancing”, Pipeline & Gas Journal, February 2011.

The paper[1] deals with "Real-Time Pipeline Leak Detection and Location Using Volume Balancing" by James Doorhy, published in the February 2011 issue of Pipeline & Gas Journal, likely provides an overview of a method for detecting and locating pipeline leaks in real-time using volume balancing techniques. It may discuss the principles behind this approach, its advantages, potential applications, and possibly case studies or examples illustrating its effectiveness.

##### Pal-Stefan Murvay, Ioan Silea, “A Survey on gas leak detection and localization techniques,” Journal of Loss Prevention in the Process Industries, vol. 25, no. 6, pp. 966- 973, Nov. 2012.

The paper[2] represents "A Survey on Gas Leak Detection and Localization Techniques" by Pal-Stefan Murvay and Ioan Silea, published in the Journal of Loss Prevention in the Process Industries in November 2012, likely provides an extensive overview of various methods and technologies used for gas leak detection and localization. It likely covers sensor-based approaches, infrared imaging, acoustic methods, and possibly other innovative techniques, discussing their principles, advantages, limitations, and real-world applications.

##### Roy, Aashis S. Anilkumar, Koppalkar R.Sasikala, M.Machappa, T.Prasad, M.V. N. Ambika, “Sensitivity Enhancement for LPG Detection by Employing Cadmium Oxide Doped in Nanocrystalline Polyaniline”, Volume 9, Number 4, August 2011, pp. 1342-1348

The article[3] "Sensitivity Enhancement for LPG Detection by Employing Cadmium Oxide Doped in Nanocrystalline Polyaniline" by Aashis S. Anilkumar Roy, Koppalkar R. Sasikala,

M. Machappa, T. Prasad, and M.V.N. Ambika, published in Volume 9, Number 4, in August 2011, likely discusses a study on enhancing the sensitivity of LPG (liquefied petroleum gas) detection. The study probably focuses on employing cadmium oxide doped in nanocrystalline polyaniline as a sensing material. It may delve into the fabrication process, characterization

techniques, experimental results demonstrating the enhanced sensitivity, and potential applications of this novel sensing approach.

##### Falkiner, RJ, “Liquefied Petroleum Gas”, Chapter 2, Jun 2003.

The citation "Liquefied Petroleum Gas" by RJ Falkiner, Chapter 2, June 2003, likely provides an in-depth exploration of liquefied petroleum gas (LPG). The chapter may cover various aspects such as the composition of LPG, its properties, production processes, storage methods, transportation, safety considerations, environmental impacts, and industrial applications. It serves as a comprehensive resource for understanding the characteristics and uses of LPG in various sectors.

##### Tai-Yih Chen, Isobel J.Simpson, Donald R.Blake, F.Sherwood Rowland, “Impact of the leakage of liquefied petroleum gas (LPG) on Santiago Air Quality”, 2001

The article "Impact of the Leakage of Liquefied Petroleum Gas (LPG) on Santiago Air Quality" by Tai-Yih Chen, Isobel J. Simpson, Donald R. Blake, and F. Sherwood Rowland, likely explores the consequences of LPG leakage on air quality in Santiago. It probably investigates the chemical composition of LPG emissions, their dispersion patterns, and their effects on atmospheric pollutants such as ozone, particulate matter, and volatile organic compounds. The study may involve field measurements, atmospheric modeling, and data analysis to assess the extent of LPG-related air pollution and its implications for public health and environmental quality in the Santiago region**.**

##### M.M. Sirdah, N.A. Al Laham and R.A. El Madhoun (2013), “Possible health effects of liquefied petroleum gas on workers at filling and distribution stations of Gaza governorates”, EMHJ, Vol. 19.

The article "Possible Health Effects of Liquefied Petroleum Gas on Workers at Filling and Distribution Stations of Gaza Governorates" by M.M. Sirdah, N.A. Al Laham, and R.A. El Madhoun, published in the Eastern Mediterranean Health Journal (EMHJ) in 2013, likely investigates the potential health risks associated with exposure to liquefied petroleum gas (LPG) among workers at filling and distribution stations in the Gaza governorates. The study may involve assessments of respiratory symptoms, occupational hazards, exposure levels to LPG, and possible long-term health effects through surveys, medical examinations, and

environmental monitoring. The findings aim to raise awareness of occupational health hazards and inform strategies for minimizing risks among workers in the LPG industry.

##### Erick D. Gamas, Moises Magdaleno , Luis Diaz, Isaac Schifter, Luis Ontiveros & G. Alvarez-Cansino (2000) Contribution of Liquefied Petroleum Gas to Air Pollution in the Metropolitan Area of Mexico City, Journal of the Air & Waste Management Association, 50:2, 188-198.

The article "Contribution of Liquefied Petroleum Gas to Air Pollution in the Metropolitan Area of Mexico City" by Erick D. Gamas, Moises Magdaleno, Luis Diaz, Isaac Schifter, Luis Ontiveros, and G. Alvarez-Cansino, published in the Journal of the Air & Waste Management Association in 2000, likely examines the impact of liquefied petroleum gas (LPG) on air pollution levels in the Metropolitan Area of Mexico City. The study may involve field measurements, atmospheric modeling, and data analysis to assess the emissions of LPG, their dispersion patterns, and their contribution to key pollutants such as ozone, particulate matter, and volatile organic compounds. The findings aim to provide insights into the role of LPG in urban air quality degradation and inform strategies for air pollution control and management in Mexico City**.**

##### Kirk R Smitha, Jonathan M Sametb, Isabelle Romieuc, Nigel Bruced, “Indoor air pollution in developing countries and acute lower respiratory infections in children”, 2000, 55:518-532

The article "Indoor Air Pollution in Developing Countries and Acute Lower Respiratory Infections in Children" by Kirk R. Smith, Jonathan M. Samet, Isabelle Romieu, and Nigel Bruce, likely explores the relationship between indoor air pollution and acute lower respiratory infections (ALRI) in children in developing countries. Published in 2000, the study probably investigates the sources of indoor air pollution, such as biomass fuel combustion for cooking and heating, and their impacts on respiratory health outcomes among children. It may involve epidemiological studies, risk assessments, and intervention evaluations aimed at understanding and mitigating the adverse effects of indoor air pollution on child health in resource-limited settings.

### EXISTING SYSTEM

The existing method for LPG leakage detection and weight monitoring relies on outdated concepts, with a system that lacks the capability to upload sensor data to a central server. This conventional approach typically involves standalone gas sensors and load cells connected to LED indicators and audible alarms, which trigger locally when a gas leak is detected or when the cylinder weight exceeds certain limits. However, this method lacks the critical feature of remote monitoring and data analysis, making it challenging to provide real-time alerts and insights to users. Without the ability to upload sensor data to a server, users are left without access to historical trends or notifications beyond their immediate vicinity, rendering it less effective in ensuring the safety and efficiency of LPG usage.

### DRAWBACKS:

* + 1. Limited Data Accessibility:
    2. Lack of Remote Monitoring:
    3. Inefficient Alerting System:
    4. Absence of Historical Data:
    5. Reduced Safety and Efficiency

# CHAPTER-3 PROPOSED SYSTEM

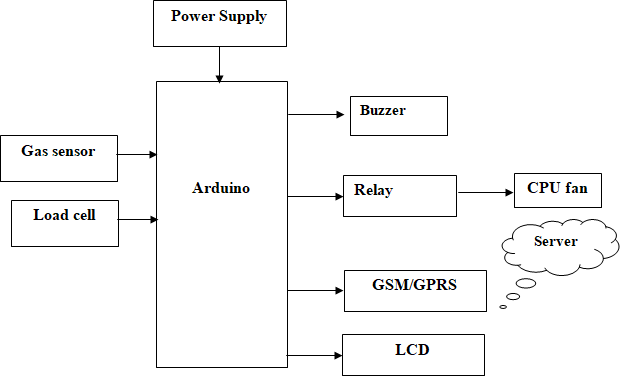
### Proposed Method:

The proposed method introduces an innovative Internet of Things (IoT)-enabled framework for LPG leakage detection and weight monitoring, addressing the limitations of the existing approach. This method leverages modern technology, such as an Arduino controller, gas sensor, LED indicators, a buzzer for audible alerts, a GSM module, and GPRS connectivity for uploading sensor data to the Thingspeak server. The primary objective of this approach is to enhance user safety and operational efficiency by allowing remote access to real-time sensor data and historical trends. The system not only detects gas leaks and weight fluctuations but also provides immediate visual, audible, and SMS alerts to users, ensuring swift responses to potential hazards. This proposed method transforms the conventional approach by enabling centralized data monitoring and analysis, ultimately ensuring safer and more efficient LPG usage.

At the heart of this proposed method is the Arduino controller, which efficiently processes data from the gas sensor and load cells. When a gas leak or weight anomaly is detected, the system activates LED indicators, a buzzer for immediate local alerts, and a GSM module for remote SMS notifications. Moreover, the inclusion of GPRS connectivity facilitates the seamless uploading of sensor values to the Thingspeak cloud server, allowing users to access real-time data and historical trends through user-friendly web or mobile applications. This comprehensive approach revolutionizes LPG safety by providing a robust and user-friendly solution that empowers users with the tools needed to mitigate risks associated with gas leakage and manage LPG cylinder weight efficiently.

The system not only detects gas leaks and weight fluctuations but also provides immediate visual, audible, and SMS alerts to users, ensuring swift responses to potential hazards. This proposed method transforms the conventional approach by enabling centralized data monitoring and analysis, ultimately ensuring safer and more efficient LPG usage.

### BLOCK DIAGRAM:



**Fig 3.1:Block Diagram of Proposed system**

### ADVANTAGES:

* + 1. Real-Time Energy Monitoring
    2. Interactive User Experience
    3. Efficient Energy Management
    4. Immediate theft Alerts

### APPLICATIONS

* + 1. **Residential Safety:** In residential settings, the system ensures enhanced safety by promptly detecting any LPG leaks, mitigating the risk of fire or explosions. Residents can receive immediate alerts on their smartphones, allowing them to take swift action to address the issue, such as shutting off the gas supply or contacting emergency services.
    2. **Commercial and Industrial Use:** Industries and commercial establishments handling large volumes of LPG can benefit from the system's real-time monitoring capabilities. It enables facility managers to maintain optimal gas levels, prevent wastage, and ensure compliance with safety regulations. Timely detection of leaks also minimizes production disruptions and protects valuable assets.
    3. **Hospitality Sector:** Hotels, restaurants, and catering services rely heavily on LPG for cooking and heating.. The ability to monitor LPG consumption and cylinder weights remotely facilitates proactive maintenance scheduling and resource planning, optimizing kitchen operations
    4. **Transportation and Distribution:** LPG distribution companies can utilize the framework to streamline their operations and enhance customer service. By monitoring cylinder weights and tracking deliveries in real-time, logistics teams can optimize route planning, manage inventory effectively, and ensure timely replenishment of cylinders at customer locations
    5. **Healthcare Facilities:** Hospitals and clinics utilize LPG for sterilization processes and backup power generation. Implementing the IoT-enabled framework enables healthcare facilities to maintain uninterrupted services while adhering to stringent safetyprotocols.
    6. **Educational Institutions:** Schools, colleges, and universities that use LPG for cooking and heating purposes can benefit from enhanced safety measures offered by the system. Real-time monitoring ensures a secure environment for students, faculty, and staff, while also optimizing operational efficiency by preventing potential disruptions due to gas leaks.
    7. **Remote Monitoring Stations:** Off-grid installations and remote monitoring stations, such as weather stations or research outposts, often rely on LPG for power generation and heating. Implementing the IoT framework allows operators to monitor LPG levels and detect leaks remotely, minimizing the risk of environmental contamination and ensuring the safety of personnel working in isolated locations.

# CHAPTER 4 HARDWARE REQUIREMENTS

### ARDUINO:

Arduino Uno is a very valuable addition in the electronics that consists of USB interface, 14 digital I/O pins, 6 analog pins, and Atmega328 microcontroller. It also supports serial communication using Tx and Rx pins.

There are many versions of Arduino boards introduced in the market like Arduino Uno, Arduino Due, Arduino Leonardo, Arduino Mega, however, most common versions are Arduino Uno and Arduino Mega. If you are planning to create a project relating to digital electronics, embedded system, robotics, or IoT, then using Arduino Uno would be the best, easy and most economical option.

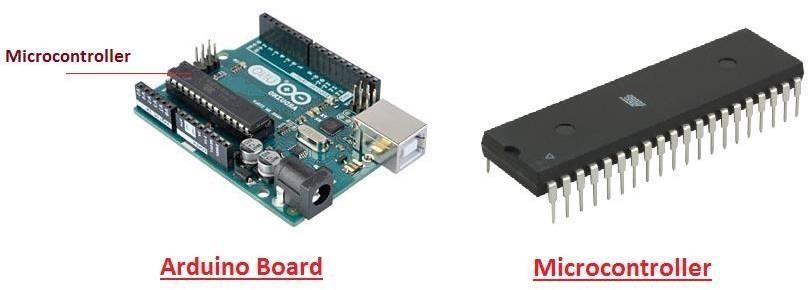


##### Fig :4.1a Types of Arduinos

The software used for Arduino devices is called IDE (Integrated Development Environment) which is free to use and required some basic skills to learn it. It can be programmed using C and C++ language.

Some people get confused between **Microcontroller and Arduino**. While former is just an on system 40 pin chip that comes with a built-in microprocessor and later is a board that comes

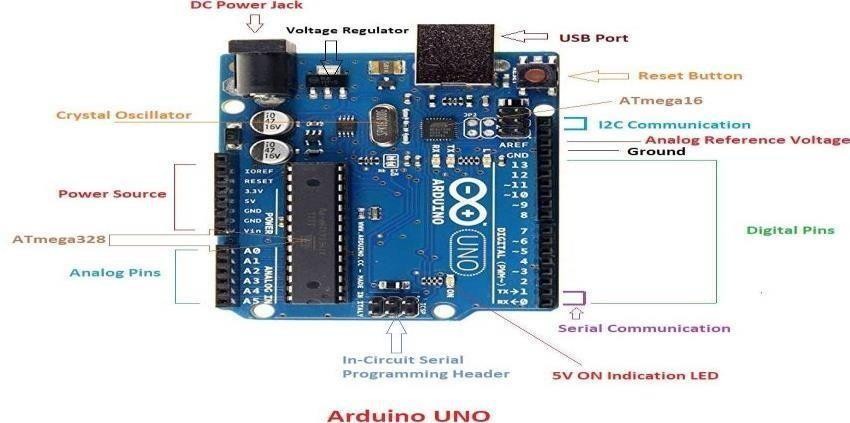
with the microcontroller in the base of the board, bootloader and allows easy access to input- output pins and makes uploading or burning of the program very easy.



**Fig :4.1b Arduino Board Fig :4.1c Microcontroller**

### INTRODUCTION TO ARDUINO

* Arduino Uno is a microcontroller board developed by Arduino.cc which is an open-source electronics platform mainly based on AVR microcontroller Atmega328.
* First Arduino project was started in Interaction Design Institute Ivrea in 2003 by David Cuartielles and Massimo Banzi with the intention of providing a cheap and flexible way to students and professional for controlling a number of devices in the real world.
  + The current version of Arduino Uno comes with USB interface, 6 analog input pins, 14 I/O digital ports that are used to connect with external electronic circuits.



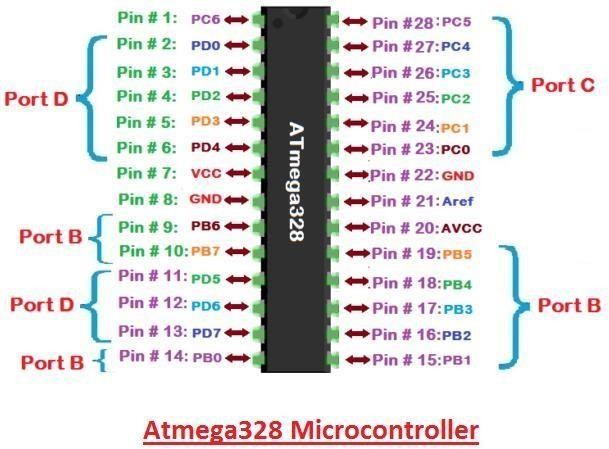
**Fig :4.1.1.1 Arduino UNO**

This board comes with all the features required to run the controller and can be directly connected to the computer through USB cable that is used to transfer the code to the controller using IDE (Integrated Development Environment) software, mainly developed to program Arduino. IDE is equally compatible with Windows, MAC or Linux Systems, however, Windows is preferable to use. Programming languages like C and C++ are used in IDE.

* Apart from USB, battery or AC to DC adopter can also be used to power the board.
* Arduino Uno boards are quite similar to other boards in Arduino family in terms of use and functionality, however, Uno boards don’t come with FTDI USB to Serial driver chip.
* There are many versions of Uno boards available, however, Arduino Nano V3 and Arduino Uno are the most official versions that come with Atmega328 8-bit AVR Atmel microcontroller where RAM memory is 32KB.

### FEATURES OF ARDUINO

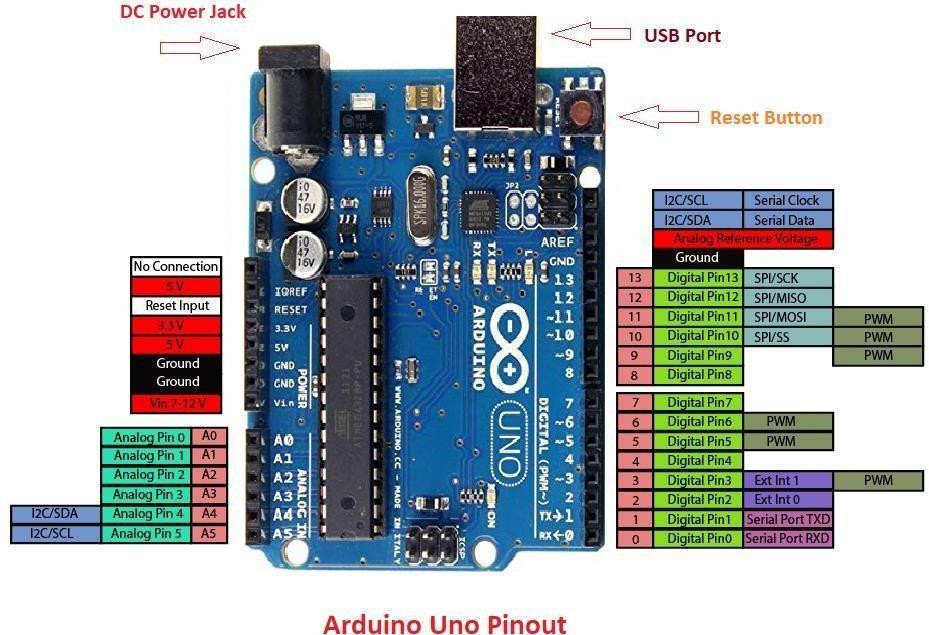
* + - * Arduino Uno comes with USB interface i.e. USB port is added on the board to develop serial communication with the computer.
      * [Atmega328](https://www.theengineeringprojects.com/2017/08/introduction-to-atmega328.html) microcontroller is placed on the board that comes with a number of features like timers, counters, interrupts, PWM, CPU, I/O pins and based on a 16MHz clock that helps in producing more frequency and number of instructions per cycle.



##### Fig 4.2.1.2:Atmega328 Microcontroller

* It is an open source platform where anyone can modify and optimize the board based on the number of instructions and task they want to achieve.
* Reset pin is added in the board that reset the whole board and takes the running program in the initial stage.
* There are 14 I/O digital and 6 analog pins incorporated in the board that allows the external connection with any circuit with the board. These pins provide the flexibility and ease of use to the external devices that can be connected through these pins
* The 6 analog pins are marked as A0 to A5 and come with a resolution of 10bits. These pins measure from 0 to 5V, however, they can be configured to the high range using analogReference() function and AREF pin.
* 13KB of flash memory is used to store the number of instructions in the form of code.
* Only 5 V is required to turn the board on, which can be achieved directly using USB port or external adopter, however, it can support external power source up to 12 V which can be regulated and limit to 5 V or 3.3 V based on the requirement of the project.

### ARDUINO PINOUT

* + - * Arduino Uno is based on AVR microcontroller called Atmega328. This controller comes with 2KB SRAM, 32KB of flash memory, 1KB of EEPROM. Arduino Board comes with 14 digital pins and 6 analog pins. A 16 MHz frequency crystal oscillator is equipped on the board. Following figure shows the pinout of the Arduino Uno Board

**Fig :4.1.3.1 Arduino UNO Pinout**

### PIN DESCRIPTION:

There are several I/O digital and analog pins placed on the board which operates at 5V. These pins come with standard operating ratings ranging between 20mA to 40mA. Internal pull-up resistors are used in the board that limits the current exceeding from the given operating conditions. However, too much increase in current makes these resisters useless and damages the device.

**LED.** Arduino Uno comes with built-in LED which is connected through pin 13. Providing HIGH value to the pin will turn it ON and LOW will turn it OFF.

**Vin.** It is the input voltage provided to the Arduino Board. It is different than 5 V supplied through a USB port. This pin is used to supply voltage. If a voltage is provided through power jack, it can be accessed through this pin.

**5V.** This board comes with the ability to provide voltage regulation. 5V pin is used to provide output regulated voltage. The board is powered up using three ways i.e. USB, Vin pin of the board or DC power jack.

USB supports voltage around 5V while Vin and Power Jack support a voltage ranges between 7V to 20V. It is recommended to operate the board on 5V. It is important to note that, if a voltage is supplied through 5V or 3.3V pins, they result in bypassing the voltage regulation that can damage the board if voltage surpasses from its limit.

**GND.** These are ground pins. More than one ground pins are provided on the board which can be used as per requirement.

**RESET.** This pin is incorporated on the board which resets the program running on the board. Instead of physical reset on the board, IDE comes with a feature of resetting the board throughprogramming.

**IOREF.** This pin is very useful for providing voltage reference to the board. A shield is used to read the voltage across this pin which then select the proper power source.

**PWM.** PWM is provided by 3, 5, 6,9,10, 11pins. These pins are configured to provide 8-bit output PWM.

**SPI.** It is known as Serial Peripheral Interface. Four pins 10(SS), 11(MOSI), 12(MISO), 13(SCK) provide SPI communication with the help of SPI library.

**AREF.** It is called Analog Reference. This pin is used for providing a reference voltage to the analog inputs.

**TWI.** It is called Two-wire Interface. TWI communication is accessed through Wire Library. A4 and A5 pins are used for this purpose.

**Serial Communication.** Serial communication is carried out through two pins called Pin 0 (Rx) and Pin 1 (Tx).

Rx pin is used to receive data while Tx pin is used to transmit data.

**External Interrupts.** Pin 2 and 3 are used for providing external interrupts. An interrupt is called by providing LOW or changing value.

### ARDUINO UNO TECHNICAL SPECIFICATIONS

|  |  |
| --- | --- |
| Microcontroller | [ATmega328P](https://components101.com/microcontrollers/atmega328p-pinout-features-datasheet) – 8 bit AVR family microcontroller |
| Operating Voltage | 5V |
| Recommended InputVoltage | 7-12V |
| Input Voltage Limits | 6-20V |
| Analog Input Pins | 6 (A0 – A5) |
| Digital I/O Pins | 14 (Out of which 6 provide PWM output) |
| DC Current on I/O Pins | 40 mA |
| DC Current on 3.3V Pin | 50 mA |
| Flash Memory | 32 KB (0.5 KB is used for Bootloader) |
| SRAM | 2 KB |
| EEPROM | 1 KB |
| Frequency (Clock Speed) | 16 MHz |

**Table :4.1.5.1 Arduino Uno Technical Specifications**

### COMMUNICATION AND PROGRAMMING:

Arduino Uno comes with an ability of interfacing with other other Arduino boards, microcontrollers and computer. The Atmega328 placed on the board provides serial communication using pins like Rx and Tx.

The Atmega16U2 incorporated on the board provides a pathway for serial communication using USB com drivers. Serial monitor is provided on the IDE software which is used to send or receive text data from the board. If LEDs placed on the Rx and Tx pins will flash, they indicate the transmission of data.



### APPLICATIONS:

Arduino Uno comes with a wide range of applications. A larger number of people are using Arduino boards for developing sensors and instruments that are used in scientific research. Following are some main applications of the board.

* [Embedded System](https://www.theengineeringprojects.com/2016/10/what-is-embedded-systems.html)
* Security and Defense System
* Digital Electronics and Robotics
* Parking Lot Counter
* Weighing Machines
* Traffic Light Count Down Timer
* Medical Instrument

There are a lot of other microcontrollers available in the market that are more powerful and cheap as compared to Arduino board. So, why you prefer Arduino Uno?

Actually, Arduino comes with a big community that is developing and sharing the knowledge with a wide range of audience. Quick support is available pertaining to technical aspects of any electronic project. When you decide Arduino board over other controllers, you don’t need to arrange extra peripherals and devices as most of the functions are readily available on the board that makes your project economical in nature and free from a lot of technical expertise.

### LCD:

LCD (Liquid Crystal Display) is the innovation utilized in scratch pad shows and other littler PCs. Like innovation for light-producing diode (LED) and gas-plasma, LCDs permit presentations to be a lot more slender than innovation for cathode beam tube (CRT). LCDs expend considerably less power than LED shows and gas shows since they work as opposed to emanating it on the guideline of blocking light.

A 16x2 LCD show is an essential module that is generally utilized in various gadgets and circuits. These modules more than seven sections and other multi fragment LEDs are liked. The reasons being: LCDs are affordable; effectively programmable; have no restriction of showing exceptional and even custom characters (not at all like in seven fragments), movements, etc.

A 16x2 LCD implies 16 characters can be shown per line and 2 such lines exist. Each character is shown in a lattice of 5x7 pixels in this LCD. There are two registers in this LCD, in particular Command and Data.

### DATA/SIGNALS/EXECUTION OF LCD

Now that was all about the signals and the hardware. Let us come to data, signals and execution.

Two types of signals are accepted by LCD, one is data and one is control. The LCD module recognizes these signals from the RS pin status. By pulling the R / W pin high, data can now also be read from the LCD display. Once the E pin has been pulsed, the LCD display reads and executes data at the falling edge of the pulse, the same for the transmission case.

Any attempt to send data before this interval may result in failure in some devices to read data or execute the current data. Some devices compensate for the speed by storing some temporary registers with incoming data.

There are two RAMs for LCD displays, namely DDRAM and CGRAM. DDRAM registers the position in which the character would be displayed in the ASCII chart. Each DDRAM byte represents every single position on the display of the LCD.

The DDRAM information is read by the LCD controller and displayed on the LCD screen. CGRAM enables users to define their personalized characters. Address space is reserved for users for the first 16 ASCII characters.

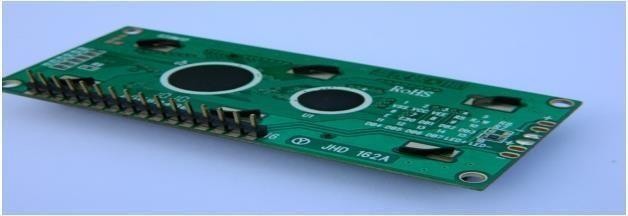
### IMAGES OF LCD DISPLAY:-

LCD stands for Liquid Crystal Display. It's a type of flat-panel display technology commonly used in TVs, computer monitors, smartphones, and other electronic devices. They offer benefits such as thinness, lightweight, and low power consumption compared to older display technologies like CRT (Cathode Ray Tube).



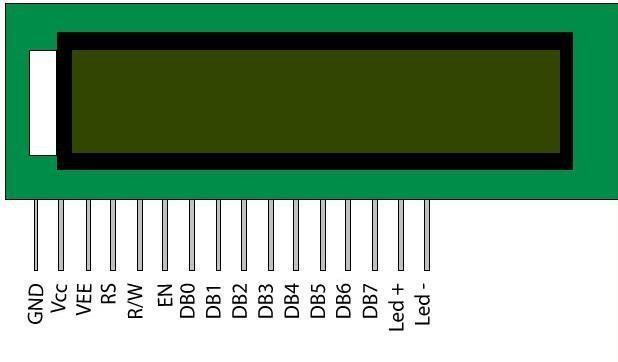
##### Fig :4.2.2.1 LCD – Front View

The back view of an LCD typically includes ports for power, video input, and sometimes audio output. It may also have mounting holes for attaching the display to a stand or wall mount. Additionally, you might find ventilation slots to dissipate heat generated by the electronics inside the display.



**Fig :4.2.2.2 LCD-Back View**

### PIN DIAGRAM:



**Fig :4.2.3 Pin Diagram**

### PIN DESCRIPTION

|  |  |  |
| --- | --- | --- |
| **Pin**  **No** | **Function** | **Name** |
| 1 | Ground (0V) | Ground |
| 2 | Supply voltage; 5V (4.7V – 5.3V) | Vcc |
| 3 | Contrast adjustment; through a variable resistor | VEE |
| 4 | Selects command register when low; and data register when high | Register Select |
| 5 | Low to write to the register; High to read from the register | Read/write |
| 6 | Sends data to data pins when a high to low pulse is given | Enable |
| 7 | 8-bit data pins | DB0 |
| 8 | DB1 |
| 9 | DB2 |
| 10 | DB3 |
| 11 | DB4 |
| 12 | DB5 |
| 13 | DB6 |
| 14 | DB7 |
| 15 | Backlight VCC (5V) | Led+ |
| 16 | Backlight Ground (0V) | Led- |

**Table :4.2.4 Pin Description of LCD**

**RS (Register select)**

A 16X2 LCD has two order and information registers. The determination of the register is utilized to change starting with one register then onto the next. RS=0 for the register of directions, while RS=1 for the register of information.

#### Command Register

The guidelines given to the LCD are put away by the direction register. An order is a direction given to LCD to play out a predefined assignment, for example, instating it.

#### Data Register:

The information register will store the information that will be shown on the LCD. The information is the character's ASCII incentive to show on the LCD. It goes to the information register and is prepared there when we send information to the LCD. While choosing RS=1, the information register.

***Read and Write Mode of LCD:***

As stated, the LCD itself comprises of an interface IC. This interface IC can be perused or composed by the MCU. A large portion of the occasions we're simply going to keep in touch with the IC since perusing will make it increasingly perplexing and situations like that are exceptionally uncommon.Information such as cursor position, status completion interrupts, etc. can be read if necessary.

The two primary modes of operation for an LCD (Liquid Crystal Display) are "Read Mode" and "Write Mode":

1. \*Read Mode:\* In this mode, the LCD is being observed or read. It's not being manipulated or altered. Read mode is when you're viewing the information displayed on the LCD screen.
2. \*Write Mode:\* This mode involves altering the information displayed on the LCD. It includes actions like updating the display with new data, changing the position of characters or pixels, or modifying the appearance of the screen in some way.

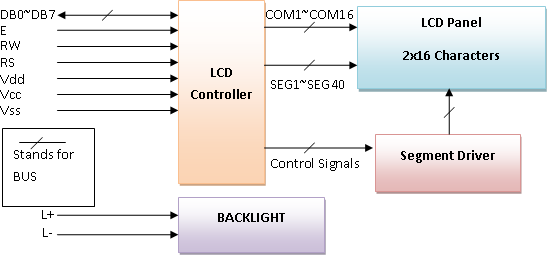
These modes are essential for understanding how to interact with an LCD, whether you're designing circuits to control it or writing software to update its content.

### LCD COMMANDS:

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Hex Code** | **Command to LCD instruction Register** |
| 1 | 01 | Clear display screen |
| 2 | 02 | Return home |
| 3 | 04 | Decrement cursor (shift cursor to left) |
| 4 | 06 | Increment cursor (shift cursor to right) |
| 5 | 05 | Shift display right |
| 6 | 07 | Shift display left |
| 7 | 08 | Display off, cursor off |
| 8 | 0A | Display off, cursor on |
| 9 | 0C | Display on, cursor off |
| 10 | 0E | Display on, cursor blinking |
| 11 | 0F | Display on, cursor blinking |
| 12 | 10 | Shift cursor position to left |
| 13 | 14 | Shift cursor position to right |
| 14 | 18 | Shift the entire display to the left |
| 15 | 1C | Shift the entire display to the right |
| 16 | 80 | Force cursor to beginning ( 1st line) |
| 17 | C0 | Force cursor to beginning ( 2nd line) |

**Table :4.2.5 LCD Commands**

### BLOCK DIAGRAM OF LCD DISPLAY:-



**Fig :4.2.6 Block Diagram**

### CONTROL AND DISPLAY COMMANDS

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Instruc tion** | **Instruction Code** | | | | | | | | | | **Instruction Code Description** | **Execut ion time** |
| R S | R/ W | D B7 | D B6 | D B5 | D B4 | D B3 | D B2 | D B1 | D B0 |
| Read Data From RAM | 1 | 1 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Read data from internal RAM | 1.53-  1.64ms |
| Write data to RAM | 1 | 0 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Write data into internal RAM (DDRAM/CG RAM) | 1.53-  1.64ms |
| Busy flag & Address | 0 | 1 | BF | A C6 | A C5 | A C4 | A C3 | A C2 | A C1 |  | Busy flag (BF: 1→ LCD Busy) and contents of  address counter | 39 µs |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  | in bits AC6-  AC0. |  |
| Set  DDRA M  Address | 0 | 0 | 1 | A  C6 | A  C5 | A  C4 | A  C3 | A  C2 | A  C1 | A  C0 | Set DDRAM  address in address counter. | 39 µs |
| Set  CGRA M  Address | 0 | 0 | 0 | 1 | A  C5 | A  C4 | A  C3 | A  C2 | A  C1 | A  C0 | Set CGRAM  Address in address counter. | 39 µs |
| Functio n Set | 0 | 0 | 0 | 0 | 1 | DL | N | F | X | X | Set interface  data length (DL: 4bit/8bit), Numbers of display line (N: 1-line/2-line) display font type (F:0→ 5×8 dots, F:1→  5×11 dots) | 39 µs |
| Cursor  or Display Shift | 0 | 0 | 0 | 0 | 0 | 1 | S/  C | R/  L | X | X | Set cursor  moving and  display shift control bit, and the direction without changing DDRAM data | 39 µs |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Display  & Cursor On/Off | 0 | 0 | 0 | 0 | 0 | 0 | 1 | D | C | B | Set  Display(D),Cur sor(C) and cursor blink(b) on/off control | 39 µs |
| Entry Mode Set | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | I/D | SH | Assign cursor moving direction and  enable shift entire display. | 0µs |
| Return  Home | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | X | Set DDRAM  Address to  “00H” from AC and return cursor to its original position if shifted. | 43µs |
| Clear  Display | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | Write “20H” to  DDRAM and set DDRAM Address to “00H” from AC | 43µs |

**Table :4.2.7 Control and Display Commands**

### RELAY:

**What is a relay?**

A relay is an electromagnetic switch that is used to turn on and turn off a circuit by a low power signal, or where several circuits must be controlled by one signal.

Most of the high end industrial application devices have relays for their effective working. Relays are simple switches which are operated both electrically and mechanically. Relays consist of an electromagnet and also a set of contacts. There are also other operating principles for its working. But they differ according to their applications. Most of the devices have the application of relays.



**Fig :4.3 Relay**

### PIN DIAGRAM:



**Fig :4.3.1 Pin Diagram of relay**

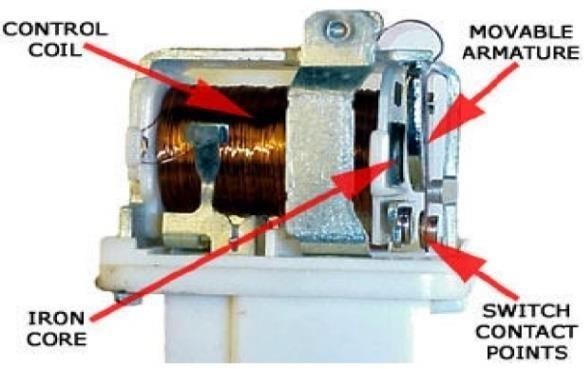
### Why is a relay used?

The main operation of a relay comes in places where only a low-power signal can be used to control a circuit. It is also used in places where only one signal can be used to control a lot of circuits. The application of relays started during the invention of telephones.

### RELAY DESIGN

* + - * There are only four main parts in a relay. They are
      * Electromagnet
      * Movable Armature
      * Switch point contacts
      * Spring

The figures given below show the actual design of a simple relay.

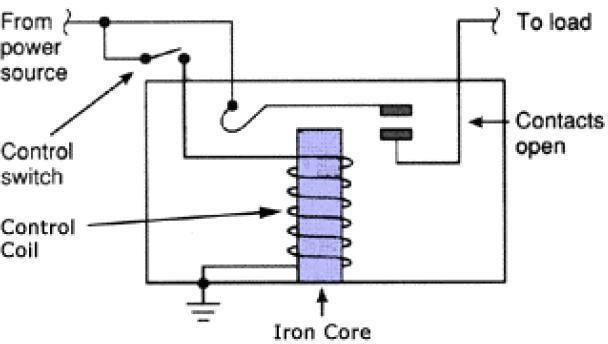


**Fig :4.3.2.1 Relay Construction**

It is an electro-magnetic relay with a wire coil, surrounded by an iron core. A path of very low reluctance for the magnetic flux is provided for the movable armature and also the switch point contacts.

### How relay works?

The relay function can be better understood by explaining the following diagram given below.



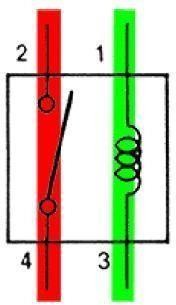
##### Fig :4.3.2.2 Relay Design

The diagram shows an inner section diagram of a relay. An iron core is surrounded by a control coil. When current starts flowing through the control coil, the electromagnet starts energizing and thus intensifies the magnetic field. Thus the upper contact arm starts to be attracted to the lower fixed arm and thus closes the contacts causing a short circuit for the power to the load.

Relays are mainly made for two basic operations. One is low voltage application and the other is high voltage. For low voltage applications, more preference will be given to reduce the noise of the whole circuit. For high voltage applications, they are mainly designed to reduce a phenomenon called arcing.

### RELAY BASICS

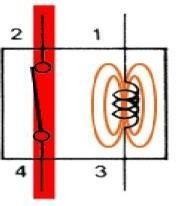
The basics for all the relays are the same. Take a look at a 4 pin relay shown below. There are two colors shown. The green color represents the control circuit and the red color represents the load circuit. Now let us take the different steps that occur in a relay.



##### Fig :4.3.3.1 Relay operation

* **Energized Relay (ON)**

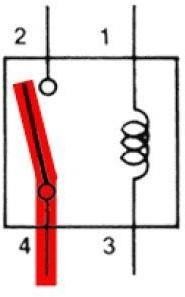
As shown in the circuit, the current flowing through the coils represented bypins 1 and 3 causes a magnetic field to be aroused. This magnetic field causes the closing of the pins 2 and 4. Thus the switch plays an important role in the relay working.



##### Fig :4.3.3.2 Energized Relay (ON)

**De – Energized Relay (OFF)**

As soon as the current flow stops through pins 1 and 3, the relay switch opens and thus the open circuit prevents the current flow through pins 2 and 4. Thus the relay becomes de- energized and thus in off position.



**Fig :4.3.3.3 De-Energized Relay (OFF)**

### POLE AND THROW

Relays have the exact working of a switch. So, the same concept is also applied. A relay is said to switch one or more poles. Each pole has contacts that can be thrown in mainly three ways. They are

* **Normally Open Contact (NO):** NO contact is also called a make contact. It closes the circuit when the relay is activated. It disconnects the circuit when the relay is inactive.
* **Normally Closed Contact (NC):** NC contact is also known as break contact. This is opposite to the NO contact. When the relay is activated, the circuit disconnects. When the relay is deactivated, the circuit connects.
* **Change-over (CO) / Double-throw (DT) Contacts:** This type of contacts are used to control two types of circuits. According to their type they are called by the names **break before make** and **make before break** contacts.
* **Single Pole Single Throw (SPST)**: The SPST relay has a total of four terminals. Out of these two terminals can be connected or disconnected. The other two terminals are needed for the coil to be connected.
* **Single Pole Double Throw (SPDT):** The SPDT relay has a total of five terminals. Out of these two are the coil terminals. A common terminal is also included which connects to either of two others.
* **Double Pole Single Throw (DPST):** The DPST relay has a total of six terminals. These terminals are further divided into two pairs. Thus they can act as two SPST which are actuated by a single coil. Out of the six terminals two of them are coil terminals.
* **Double Pole Double Throw (DPDT)**: The DPDT relay is the biggest of all. It has mainly eight relay terminals. Out of these two rows are designed to be change over terminals. They are designed to act as two SPDT relays which are actuated by a single coil.

### Relay Applications

* A relay circuit is used to realize logic functions. They play a very important role in providing safety critical logic.
* Relays are used to provide time delay functions. Theyare used to time the delay open and delay close of contacts.
* They are also used as protective relays. By this function all the faults during transmission and reception can be detected and isolated.

### APPLICATION OF OVERLOAD RELAY

Overload relay is an electro-mechanical device that is used to safeguard motors from overloads and power failures. Overload relays are installed in motors to safeguard against sudden current spikes that may damage the motor. An overload relay switch works in characteristics with current over time and is different from circuit breakers and fuses, where a sudden trip is made to turn off the motor.

All overload relays available to buy comes in different specifications, the most important of them being the current ranges and response time. Most of them are designed to automatically reset to work after the motor is turned back on.

### Relay Selection

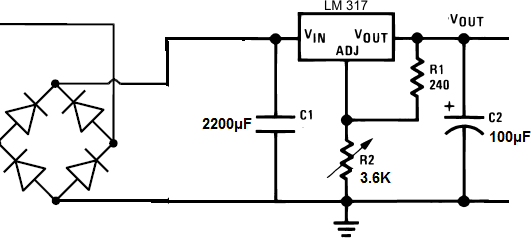
You must note some factors while selecting a particular relay. They are

* Protection Different protections like contact protection and coil protection must be noted. Contact protection helps in reducing arcing in circuits using inductors. Â Coil protection helps in reducing surge voltage produced during switching.
* Look for a standard relay with all regulatory approvals.
* Switching time Ask for high speed switching relays if you want one.
* Ratings There are current as well as voltage ratings. The current ratings vary from a few amperes to about 3000 amperes. Â In case of voltage ratings, they vary from 300 Volt AC to 600 Volt AC. There are also high voltage relays of about 15,000 Volts.
* Type of contact used whether it is a NC or NO or closed contact.

### Power supply:

A power supply is a component that provides at least one electrical charge with power. It typically converts one type of electrical power to another, but it can also convert a different Energy form in electrical energy, such as solar, mechanical, or chemical.

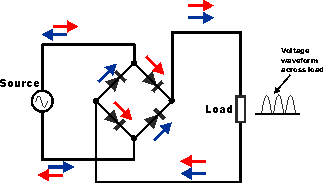
A power supply provides electrical power to components. Usually, the term refers to devices built into the powered component.Computer power supplies, for example, convert AC current to DC current and are generally located along with at least one fan at the back of the computer case.



**Fig :4.4 Power Supply**

### Rectifier:

A **rectifier** is an electrical device that [convertsalternating current](https://en.wikipedia.org/wiki/Electric_power_conversion) (AC), which periodically reverses direction, to [direct current](https://en.wikipedia.org/wiki/Direct_current) (DC), which flows in only one direction. The process is known as *rectification*, since it "straightens" the direction of current.



**Fig :4.4.1a Circuit of rectifier fig :4.4.1b Rectifier**

### CAPACITORS:

Capacitors are used to attain from the connector the immaculate and smoothest DC voltage in which the rectifier is used to obtain throbbing DC voltage which is used as part of the light of the present identity. Capacitors are used to acquire square DC from the current AC experience of the current channels so that they can be used as a touch of parallel yield.

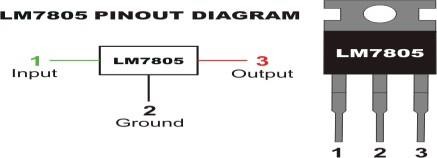


**Fig :4.4.2 Capacitor**

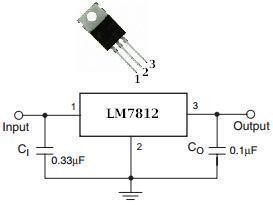
### VOLTAGE REGULATORS:

The 78XX voltage controller is mainly used for voltage controllers as a whole. The XX speaks to the voltage delivered to the specific gadget by the voltage controller as the yield. 7805 will supply and control 5v yield voltage and 12v yield voltage will be created by 7812.

The voltage controllers are that their yield voltage as information requires no less than 2 volts. For example, 7805 as sources of information will require no less than 7V, and 7812, no less than 14 volts. This voltage is called Dropout Voltage, which should be given to voltage controllers.



##### Fig :4.4.3a 7805 voltage regulator with pinout



**Fig :4.4.3b7812 voltage regulator with pinout**

### BUZZER:

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers and confirmation of user input such as a mouse click or keystroke. Buzzer is an integrated structure of electronic transducers, DC power supply, widely used in computers, printers, copiers, alarms, electronic toys, automotive electronic equipment, telephones, timers and other electronic products for sound devices.



**Fig :4.4.4 Buzzer**

### BUZZER PIN CONFIGURATION

|  |  |  |
| --- | --- | --- |
| **Pin Number** | **Pin Name** | **Description** |
| 1 | Positive | Identified by (+) symbol or longer terminal lead. Can be powered by 5V DC |
| 2 | Negative | Identified by short terminal lead. Typically connected to the ground of the circuit |

**Table :4.4.4.1 Pin Configuration of Buzzer**

### 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

### 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](https://components101.com/misc/breadboard-connections-uses-guide), 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.

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.

### APPLICATIONS OF BUZZER

* + Alarming Circuits, where the user has to be alarmed about something
  + Communication equipment’s
  + Automobile electronics
  + Portable equipment’s, due to its compact size

### MQ2 SENSOR:

In current technology scenario, monitoring of gases produced is very important. From home appliances such as air conditioners to electric chimneys and safety systems at industries monitoring of gases is very crucial. Gas sensors are very important part of such systems. Small like a nose, gas sensors spontaneously react to the gas present, thus keeping the system updated about any alterations that occur in the concentration of molecules at gaseous state.



##### Fig :4.4.5a Gas Sensor

The **gas sensor module** consists of a steel exoskeleton under which a sensing element is housed. This current is known as heating current through it, the gases coming close to the sensing element get ionized and are absorbed by the sensing element.



**Fig :4.4.5b Parts of a Gas Sensor**

Image 01 shows externals of a standard gas sensor module: a steel mesh, copper clamping ring and connecting leads. The top part is a stainless steel mesh which takes care of the following:

1. Filtering out the suspended particles so that only gaseous elements are able to pass to insides of the sensor.
2. Protecting the insides of the sensor.



**Fig :4.4.5c Steel Mash Used In Gas Sensor**

The connecting leads of the sensor are thick so that sensor can be connected firmly to the circuit and sufficient amount of heat gets conducted to the inside part. Four of the six leads (A, B, C, D) are for signal fetching while two (1, 2) are used to provide sufficient heat to the sensing element.

### 4.4.5.1 INTERNAL FEATURES



**Fig :4.4.5.1a Inside View of Gas Sensor**

The top of the gas sensor is removed off to see the internals parts of the sensor: sensing element and connection wiring.

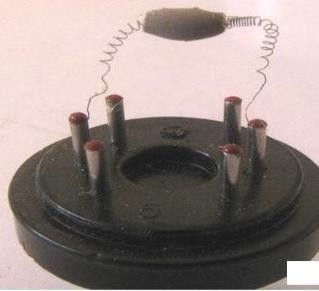
The hexapod structure is constituted by the sensing element and six connecting legs that extend beyond the Bakelite base.



##### Fig :4.4.5.1b Hexapod Structure

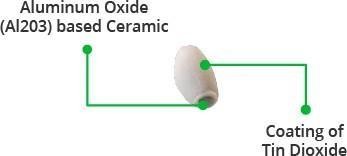
Image4 shows the hollow sensing element which is made up from Aluminum Oxide based ceramic and has a coating of tin oxide. Using a ceramic substrate increases the heating efficiency and tin oxide, being sensitive towards adsorbing desired gas’ components (in this case methane and its products) suffices as sensing coating.

The platinum wires are connected to the body of the sensing element while Nickel-Chromium wires pass through its hollow structure.

**Ceramic Sensing Element**

##### Fig :4.4.5.1c Ceramic Sensing Element

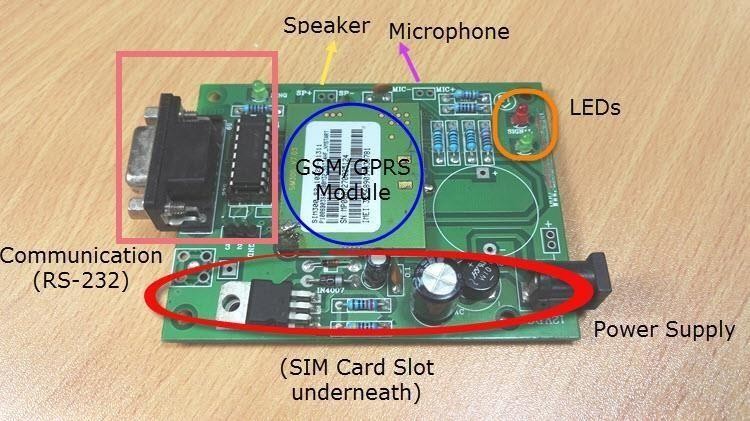
While other wires are attached to the outer body of the element, Nickel-Chromium wires are placed inside the element in a spring shaped. Image 5 shows coiled part of the wire which is placed on the inside of the hollow ceramic.



##### Fig :4.4.5.1d Closer Look at the Ceramic Element

Image06 shows the ceramic with tin dioxide on the top coating that has good adsorbing property. Any gas to be monitored has specific temperature at which it ionizes. The task of the sensor is to work at the desired temperature so that gas molecules get ionized. Adsorbed molecules change the resistance of the tin dioxide layer. This changes the current flowing through the sensing element and is conveyed through the output leads to the unit that controls the **working of the gas sensor**.

### 4.4.5 GSM/GPRS MODULE :



##### Fig :4.4.6 GSM/GPRS

GPRS Modules are one of the commonly used communication modules in embedded systems. A GPRS Module is used to enable communication between a microcontroller (or a microprocessor) and the GPRS Network. Here, GSM stands for Global System for Mobile Communication and GPRS stands for General Packet Radio Service.

GPRS Modules allow microcontrollers to have a wireless communication with other devices and instruments. Such wireless connectivity of microcontroller opens up to wide range of applications like Home Automation, Home Security Systems, Disaster Management, Medical Assistance, Vehicle Tracking, Online Banking, E – Commerce etc. to name some.

What is GPRS?

GPRS or General Packet Radio Service is an extension of the GSM Network. GPRS is an integrated part of the GSM Network which provides an efficient way to transfer data with the same resources as GSM Network.

Difference between a Module, MODEM and Mobile (System)

A GSM/GPRS Module is a device or chip that is actually responsible for the wireless communication with the GSM Network.

A GSM/GPRS MODEM is device that modulates and demodulates the signals from the Wireless Network and allows internet connectivity. A GSM MODEM generally consists of a GSM Module along with some other components like a SIM Card, a device to modulate and demodulate the signals and power supply.

A System, like a mobile phone for example, is a complete device that has a GSM Module (might be integrated in the processor), a GSM MODEM (even this might be integrated) and other components like processor, screen, keypad, speakers, microphone etc.

### GPRS MODULE

A GPRS Module is an IC or chip that connects to the GSM Network using a SIM (Subscriber Identity Module) and Radio Waves. The common radio frequencies in which a typical GSM Module operates are 850MHz, 900MHz, 1800MHz and 1900MHz.

It consists of the GPRS Module, slot for inserting a SIM Card, RS-232 Interface for connecting with computer or a microcontroller, signal status LED, power supply and a provision for connecting microphone and speaker.

Each GPRS Module is unique and it can be differentiated by its IMEI Number. IMEI or International Mobile Equipment Identity Number is a 15 – digit unique number associated with mobile phone, satellite phones and other GSM Network devices.

With the help of this GPRS Module, we can do the following tasks.

* + - * + Make, receive or reject voice calls
        + Send, receive or delete SMS messages in the SIM Card
        + Add, read and search the contacts in the SIM Card

The processor or controller, to which the GSM/GPRS Module is connected to, is responsible for sending the AT Commands to the module. In response, the GSM Module performs command specific tasks like answering a phone call; send an SMS Message, etc.

Even through the AT Commands may seem generic; it is advisable to refer with the data provided by the manufacturer of the GSM Module for correct and complete list of AT Commands.

### SYNTAX OF AT COMMANDS

Generally, AT Commands consists of three parts: the Prefix, the Body or Command and the Termination. The Prefix part of the command consists of either “AT” or “at”. The body or command is the actual command given to the GSM Module. The Termination character is by default Carriage Return <CR>.

Before seeing an example, we should know that AT Commands are categorized in to three types: Basic AT Commands, S Parameter AT Commands and Extended AT Commands.

Example for the Basic Commands: ATCMD1<CR>, where AT is the prefix of the command line, CMD1 is the body of the command and <CR> command terminator character.

***NOTE***: Basic commands never begin with +. Example for Extended Commands: AT+CMD1<CR>.

**NOTE**: Extended AT Commands always begin with +.

**NOTE**: The Carriage Return <CR> will be omitted from the next command example or syntax. You have to assume them as a part of the command even though they are not inserted in the text here.

Types of AT Command Operations

Based on the operation performed by the AT Commands, they are again divided in to four types: Test Commands, Read Commands, Write Commands and Execution Commands. We will now see the definition and syntax of all these command types.

**Test Commands**: These commands are used to test whether the command exists (supported by the GSM GPRS Module) or not and also checks for the range of a command’s

subparameters. When a Test Command is given to the GSM GPRS Module, it returns the list of all the parameters and also the range set of the parameters.

The syntax of a Test Command is ATCMD1=?<CR>

Example for Test Command: AT+CGMI=? (Request manufacturer identification)

**Read Commands**: Read Commands will return the current value of the parameter. Using these commands, we can read the current settings of the GSM GPRS Module.

The syntax for Read Commands is: ATCMD1?<CR>

Example for Read Command: AT+CSCA? (Query for Service Center)

**Write or Set Commands**: Set Commands will attempt to change or modify the settings of the GSM GPRS Module by setting a new parameter for the particular command.

The syntax for Set Commands is: ATCMD1=value1,value2,value3…valuen<CR> An example for Set Command: AT+CMGF=1 (Set Message format to TEXT Mode).

**Execution Command**: These commands perform an operation like send an SMS, retrieving information about battery charging status etc. They read the non – variable subparameters that are affected by the GSM Module.

Syntax of Execution Commands: ATCMD1<CR>

Example for Execution Commands: AT+CMGS=<number><CR><text message><CTRL-Z> (Sends text message to the number).

Information Responses and Final Codes

After sending the AT Commands to the GSM GPRS Module, we have look for the response. For example, if we send the command as AT+CGMI<CR> to the GSM Module, then the response would be as follows.

<CR><LF>Apple<CR><LF>

<CR><LF>OK<CR><LF>

Here, <CR> is Carriage Return and <LF>is Line Feed.

In a HyperTerminal, if you entered AT+CGMI<CR>, the response will look something like this.

AT+CGMI <– Command entered Apple <– Information Response OK <– Final Code

The syntax of the information response and final command is as follows:

<Carriage Return><Line Feed><Information Response / Final Result Code><Carriage Return><Line Feed>

<CR><LF><Response><CR><LF>

***NOTE***: The sequence of execution of commands will be first commands, then second command, followed by the rest i.e. a sequential execution of commands.

If there is an error anywhere in the execution, an error code is returned by the GSM Module and the execution of further commands is terminated.

Frequently used AT Commands

In this list, you can find out some of the most commonly used AT Commands. For a complete list of AT Commands and their definitions, it is advised to refer the manufacturer data. The

<Carriage Return> or <CR> is denoted by this symbol ↲.

###### To check the communication between the GSM Module and the host (Computer)

AT ↲

OK

###### To make a voice call

ATD9848032919; ↲

###### To answer or receive an incoming call

ATA ↲

###### To redial the last number

ATDL ↲

###### To disconnect a call

ATH ↲

###### To set the message mode to text mode

AT+CMGF=1 ↲

OK

***To send a text message*** AT+CMGS=”9848032919” ↲ CTRL+Z

### CPU FAN

The CPU fan is used to cool the CPU (central processing unit) heat sink. Effective cooling of concentrated heat sources such as large integrated circuits requires a heat sink, which can be cooled by a fan. However, using a fan alone does not prevent the small chip from overheating.

A CPU fan is a crucial component in your computer that keeps the central processing unit cool, preventing overheating and ensuring smooth performance. It's the unsung hero that works tirelessly to maintain optimal temperatures, much like a radiator for your car's engine



##### Fig :4.4.7 CPU Fan

* + - * Perfectly fit on the extruder
      * Noiseless Performance
      * Very low Current consumption
      * Easy to install and connect

The 12VDC ZY-922512SM 2.16W Cooling Fan Power Supply Cabinet powerful exhaust cooling is designed for the power supply or CPU cabinet. It has a two-pin connector to connect in a controller 12V output pin.

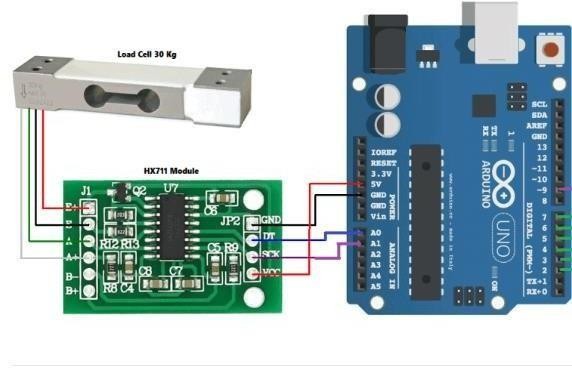
### LOAD CELL:

A load cell is a transducer that is used to create an electrical signal whose magnitude is directly proportional to the force being measured. It is basically a device that measures strain and then converts force into electric energy which serves as a measurement for scientists and workers. The strain measurement by load cells helps in maintaining the integrity of the unit under pressure and protects people and equipment nearby.

### 4.4.8.1 HX711 Load Cell Amplifier

The Load Cell Amplifier is a small breakout board for the HX711 IC that allows you to easily read load cells to measure weight. By connecting the amplifier to your microcontroller you will be able to read the changes in the resistance of the load cell, and with some calibration, you’ll be able to get very accurate weight measurements.

The HX711 uses a two-wire interface (Clock and Data) for communication. Any microcontroller’s GPIO pins should work, and numerous libraries have been written, making it easy to read data from the HX711. These are commonly colored RED, BLK, WHT, GRN, and YLW. Each color corresponds to the conventional color coding of load cells:



##### Fig :4.4.8.1 HX711 Cell Amplifier

Before making the whole circuit to work it is necessary to “calibrate load cell with HX711 with Arduino“. For “calibrating Load Cell with HX711 with Arduino” we need to put 100g weight at starting when the LCD displays put 100g weight. Once 100gm weight is kept over the load cell, calibration is done. Now simply you can put any weight for measuring with 99.9% accuracy.

Load cell which is an amplifier senses the weight and supplies an electrical analog voltage to HX711 Load Amplifier Module. The output result is displayed on the 16\*2 LCD.

he HX711 is a precision 24-bit analog-to-digital converter (ADC) specifically designed for weigh scales and industrial control applications to interface directly with a bridge sensor. It provides two differential inputs for the bridge sensor and performs low-noise amplification and 24-bit ADC conversion. It's commonly used in load cell amplifier applications to accurately measure weight or force.

# CHAPTER 5 SOFTWARE REQUIREMENTS

### ARDUINO IDE:

**Arduino IDE** where IDE stands for Integrated Development Environment – An official software introduced by Arduino.cc, that is mainly used for writing, compiling and uploading the code in the Arduino Device. Almost all Arduino modules are compatible with this software that is an open source and is readily available to install and start compiling the code on the go.

### INTRODUCTION TO ARDUINO IDE:

* + Arduino IDE is an open source software that is mainly used for writing and compiling the code into the Arduino Module.
  + It is an official Arduino software, making code compilation too easy that even a common person with no prior technical knowledge can get their feet wet with the learning process.
  + It is easily available for operating systems like MAC, Windows, and Linux and runs on the Java Platform that comes with inbuilt functions and commands that play a vital role for debugging, editing and compiling the code in the environment.
  + A range of Arduino modules available including Arduino Uno, Arduino Mega, Arduino Leonardo, [Arduino Micro](https://www.theengineeringprojects.com/2018/09/introduction-to-arduino-micro.html) and many more.
  + This environment supports both C and C++ languages.

### HOW TO INSTALL ARDUINO IDE:

You can download the Software from [Arduino](https://www.arduino.cc/en/Main/Software) main website. As I said earlier, the software is available for common operating systems like Linux, Windows, and MAX, so make sure you are downloading the correct software version that is easily compatible with your operating system.

* + If you aim to download Windows app version, make sure you have Windows 8.1 or Windows 10, as app version is not compatible with Windows 7 or older version of this operating system.

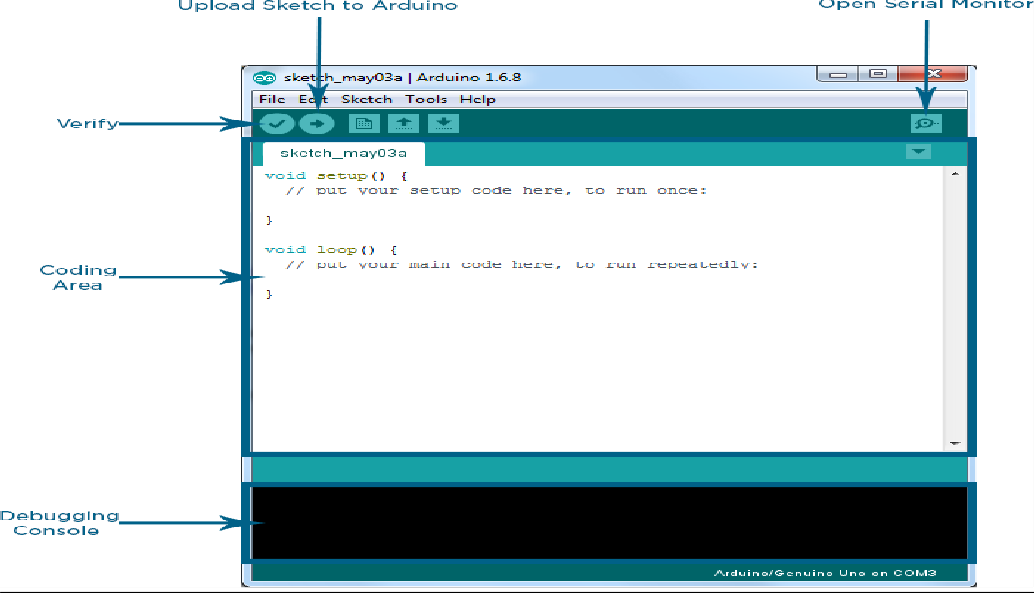
The IDE environment is mainly distributed into three sections

##### 1. Menu Bar

* + **2. Text Editor**

##### 3. Output Pane

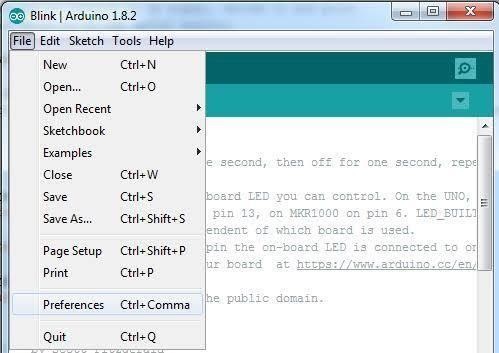
As you download and open the IDE software, it will appear like an image below.



##### Fig :5.2.1.1 Arduino IDE

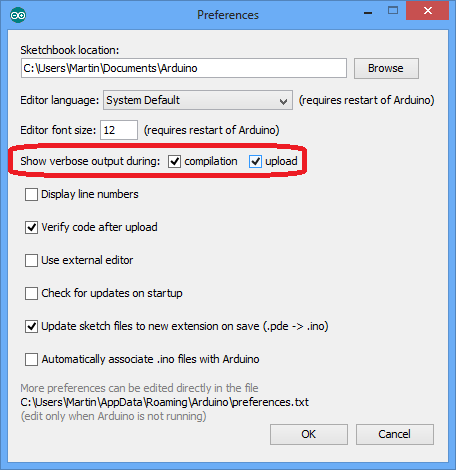
The bar appearing on the top is called **Menu Bar** that comes with five different options as follow

* + **File** – You can open a new window for writing the code or open an existing one. Following table shows the number of further subdivisions the file option is categorized into.



##### Fig :5.2.1.2 File Option

As you go to the preference section and check the compilation section, the Output Pane will show the code compilation as you click the upload button.



##### Fig :5.2.1.3 Preferences

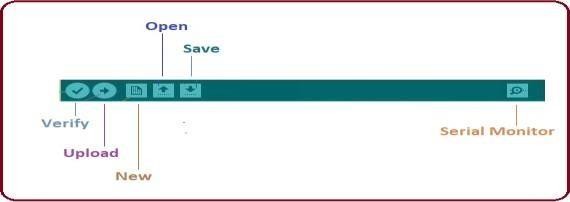
And at the end of compilation, it will show you the hex file it has generated for the recent sketch that will send to the Arduino Board for the specific task you aim to achieve.



##### Fig :5.2.1.4 Hex File

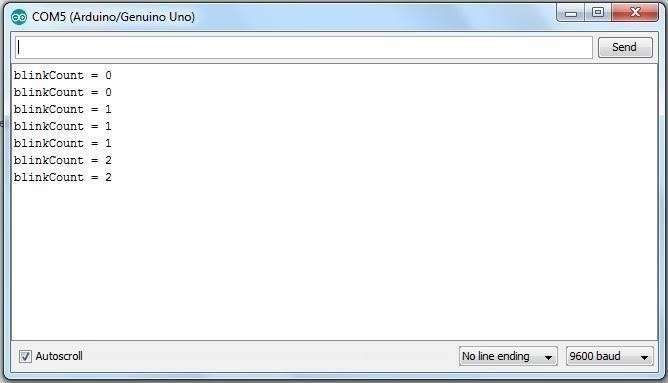
* + **Edit** – Used for copying and pasting the code with further modification for font
  + **Sketch** – For compiling and programming
  + **Tools** – Mainly used for testing projects
  + **Help** – In case you are feeling skeptical about software, complete help is available from getting started to troubleshooting.

The **Six Buttons** appearing under the Menu tab are connected with the running program as follow.



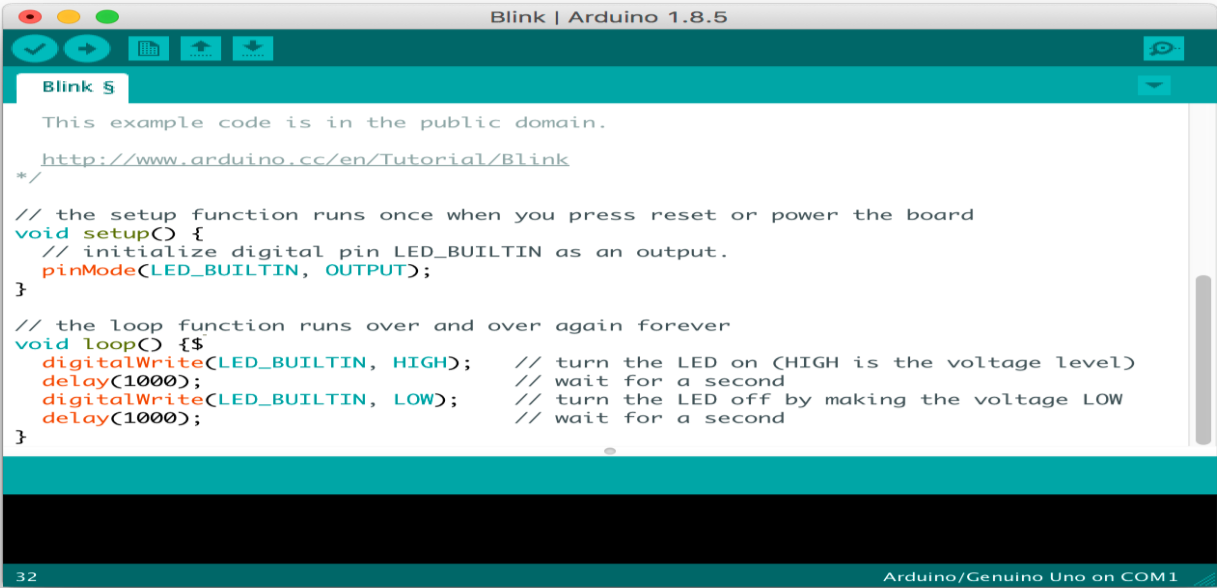
##### Fig :5.2.1.5 Menu Tab

* + The check mark appearing in the circular button is used to verify the code. Click this once you have written your code.
  + The arrow key will upload and transfer the required code to the Arduino board.
  + The dotted paper is used for creating a new file.
  + The upward arrow is reserved for opening an existing Arduino project.
  + The downward arrow is used to save the current running code.



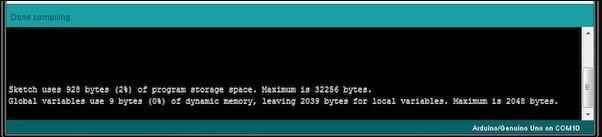
##### Fig :5.2.1.6 Output In Serial Monitor

The main screen below the Menu bard is known as a simple text editor used for writing the required code.



##### Fig:5.2.1.7 Text Editor

The bottom of the main screen is described as an Output Pane that mainly highlights the compilation status of the running code: the memory used by the code, and errors occurred in the program. You need to fix those errors before you intend to upload the hex file into your Arduino Module.

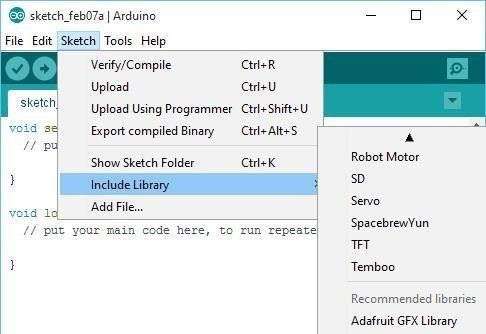


##### Fig:5.2.1.8 Output Pane

More or less, Arduino C language works similar to the regular C language used for any embedded system microcontroller, however, there are some dedicated libraries used for calling and executing specific functions on the board.

### LIBRARIES:

Libraries are very useful for adding the extra functionality into the Arduino Module. There is a list of libraries you can add by clicking the Sketch button in the menu bar and going to Include Library.



##### Fig :5.3 Sketch Button

As you click the Include Library and Add the respective library it will on the top of the sketch with a #include sign. Suppose, I Include the EEPROM library, it will appear on the text editor as

#include <EEPROM.h>.

### MAKING PINS INPUT AND OUTPUT:

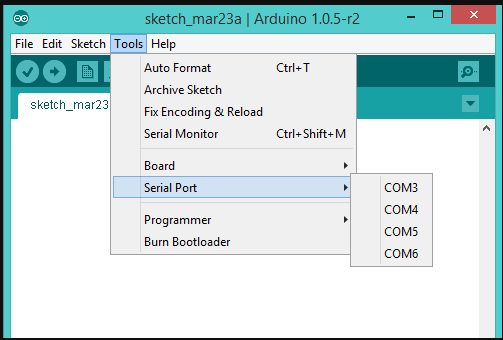
The digitalRead and [digitalWrite](https://www.theengineeringprojects.com/2018/09/how-to-use-digitalwrite-arduino-command.html) commands are used for addressing and making the Arduino pins as an input and output respectively.

These commands are text sensitive i.e. you need to write them down the exact way they are given like digitalWrite starting with small “d” and write with capital “W”. Writing it down with Digitalwrite or digitalwrite won’t be calling or addressing any function.

### HOW TO SELECT THE BOARD:

In order to upload the sketch, you need to select the relevant board you are using and the ports for that operating system. As you click the Tools on the Menu, it will open like the figure below.

* + Just go to the “Board” section and select the board you aim to work on. Similarly, COM1, COM2, COM4, COM5, COM7 or higher are reserved for the serial and USB boardFollowing figure shows the COM4 that I have used for my project, indicating the Arduino Uno with COM4 port at the right bottom corner of the screen.



##### Fig :5.4.1.1 Port Selection In Tools Option

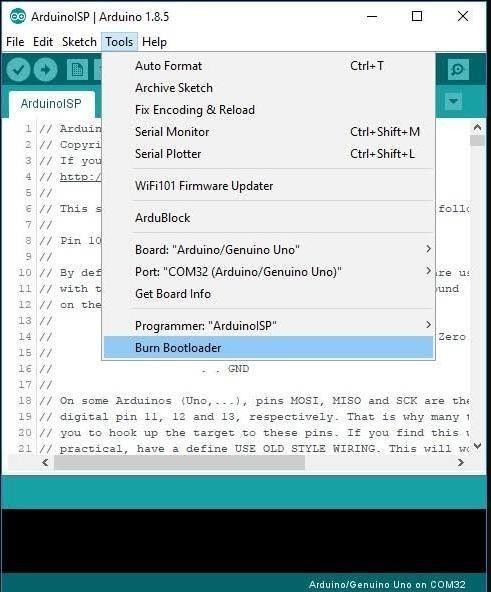
* + After correct selection of both Board and Serial Port, click the verify and then upload button appearing in the upper left corner of the six button section or you can go to the Sketch section and press verify/compile and then upload.
  + The sketch is written in the text editor and is then saved with the file extension .ino..
  + Once you upload the code, TX and RX LEDs will blink on the board, indicating the desired program is running successfully.

**Note**: The port selection criteria mentioned above is dedicated for Windows operating system only, you can check this [Guide](https://www.arduino.cc/en/Guide/Environment) if you are using MAC or Linux.

* + The amazing thing about this software is that no prior arrangement or bulk of mess is required to install this software, you will be writing your first program within 2 minutes after the installation of the IDE environment.

### BOOTLOADER:

As you go to the Tools section, you will find a bootloader at the end. It is very helpful to burn the code directly into the controller, setting you free from buying the external burner to burn the required code.



##### Fig :5.4.2 Boot Loader

When you buy the new Arduino Module, the bootloader is already installed inside the controller. However, if you intend to buy a controller and put in the Arduino module, you need to burn the bootloader again inside the controller by going to the Tools section and selecting the burn bootloader.

# CODE:

**CHAPTER 6 CODE AND RESULTS**

#include<LiquidCrystal.h> LiquidCrystal lcd(A5,A4,A3,A2,A1,A0);

#include "HX711.h"

#define calibration\_factor 244.70 //This value is obtained using the SparkFun\_HX711\_Calibration sketch

#define LOADCELL\_DOUT\_PIN 3

#define LOADCELL\_SCK\_PIN 2 int gas=4;

int fan=5; int buz=6;

int gas\_value; HX711 scale; float val; void setup()

{

Serial.begin(9600); pinMode(gas,INPUT); pinMode(fan,OUTPUT); pinMode(buz,OUTPUT); digitalWrite(buz,LOW); digitalWrite(fan,HIGH); Serial.println("HX711 scale demo");

scale.begin(LOADCELL\_DOUT\_PIN, LOADCELL\_SCK\_PIN);

scale.set\_scale(calibration\_factor); //This value is obtained by using the SparkFun\_HX711\_Calibration sketch

scale.tare(); //Assuming there is no weight on the scale at start up, reset the scale to 0

Serial.println("Readings:"); lcd.begin(16,2);

lcd.clear(); lcd.setCursor(0,0);

lcd.print("LPG GAS LEAK"); lcd.setCursor(0,1); lcd.print("WEIGHT MONITOR");

}

void loop()

{

Serial.print("Reading: "); val=scale.get\_units();

// val=(val,2);

Serial.print(val); //scale.get\_units() returns a float

Serial.print(" gm"); //You can change this to kg but you'll need to refactor the calibration\_factor

Serial.println(); gas\_value=digitalRead(gas); lcd.clear(); lcd.setCursor(0,0); lcd.print("LPG WEIGHT: "); lcd.setCursor(0,1); lcd.print(val); lcd.setCursor(8,1); lcd.print("gms"); delay(1000);

lcd.clear(); lcd.setCursor(0,0); lcd.print("LPG GAS:"); lcd.setCursor(0,1); lcd.print(gas\_value);

if(gas\_value==0&&val<100)

{

digitalWrite(buz,HIGH); digitalWrite(fan,LOW); lcd.clear(); lcd.setCursor(0,0);

lcd.print("GAS LEAKING.."); lcd.setCursor(0,1); lcd.print("WEIGHT DECRESE..");

delay(1000); sendsms(); gprs();

}

else if(gas\_value==0&&val>100)

{

digitalWrite(buz,HIGH); digitalWrite(fan,LOW); lcd.clear(); lcd.setCursor(0,0);

lcd.print("GAS LEAKING.."); lcd.setCursor(0,1); lcd.print("NORMAL WEIGHT"); delay(1000);

}

else if(gas\_value==1&&val<100)

{

digitalWrite(buz,HIGH); digitalWrite(fan,HIGH); lcd.clear(); lcd.setCursor(0,0); lcd.print("NO GAS LEAK"); lcd.setCursor(0,1);

lcd.print("WEIGHT DECREASE..");

}

else if(gas\_value==1&&val>100)

{

digitalWrite(buz,LOW); digitalWrite(fan,HIGH); lcd.clear(); lcd.setCursor(0,0); lcd.print("NO GAS LEAK"); lcd.setCursor(0,1);

lcd.print("NORMAL WEIGHT"); delay(1000);

}

else

{

digitalWrite(buz,LOW); digitalWrite(fan,HIGH);

}

}

void sendsms()

{

delay(1000); Serial.println("AT"); delay(1000); Serial.println("ATE0"); delay(1000); Serial.println("AT+CMGF=1"); delay(1000);

Serial.println("AT+CMGS=\"6305406071\""); delay(1000);

Serial.println("LPG GAS LEAKED & WEIGHT DECREASED");

delay(1000); Serial.write(26); lcd.clear();

lcd.setCursor(0, 0); lcd.print("msg sent"); delay(1000); Serial.println("msg sent");

}

void gprs()

{

lcd.clear(); lcd.setCursor(0, 1);

lcd.print("DATA UPLOADING");

Serial.println("AT");//at test delay(1000);

Serial.println("AT+CPIN?");//This is to check if SIM is unlocked delay(1000);

Serial.println("AT+CREG?"); //This checks if SIM is registered or not delay(1000);

Serial.println("AT+CGATT?");//Check if GPRS is attached or not delay(1000);

Serial.println("AT+CIPSHUT");//Reset the IP session if any delay(1000);

Serial.println("AT+CIPSTATUS");//Check if the IP stack is initialized delay(2000);

Serial.println("AT+CIPMUX=0");//To keep things simple, I’m setting up a single connection mode

delay(2000);

Serial.println("AT+CSTT=\"Airtel Internet\"");//start task and setting the APN, delay(1000);

Serial.println("AT+CIICR");//Now bring up the wireless. Please note, the response to this might take some time

delay(6000);

Serial.println("AT+CIFSR");//get local IP adress delay(1000);

Serial.println("AT+CIPSPRT=0");

delay(3000); Serial.println("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",\"80\"");//start up the connection

delay(10000);

Serial.println("AT+CIPSEND");//begin send data to remote server delay(10000);

lcd.clear(); lcd.setCursor(0,0); lcd.print("Data uploding...");

String strs="GET

https://api.thingspeak.com/update?api\_key=S817TOAMVKXKSUSV&field1=" + String(val)+ "&field2="+ String(gas\_value);

Serial.println(strs);//begin send data to remote server delay(6000);

Serial.write(26);//sending

delay(6000);//waiting for reply, important! the time is base on the condition of internet Serial.println();

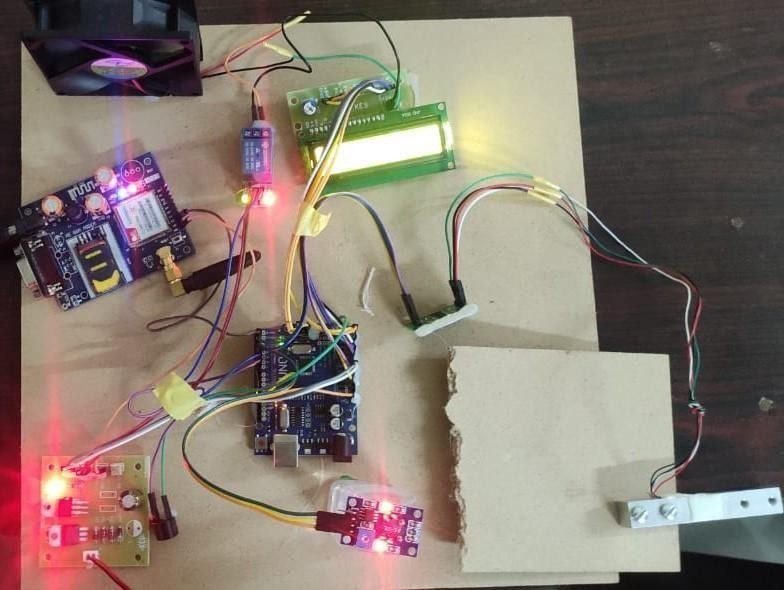
Serial.println("AT+CIPSHUT");//close the connection delay(1000);

Serial.println("Data uploded in server"); lcd.clear();

lcd.setCursor(0,0); lcd.print("data uploded");

## }

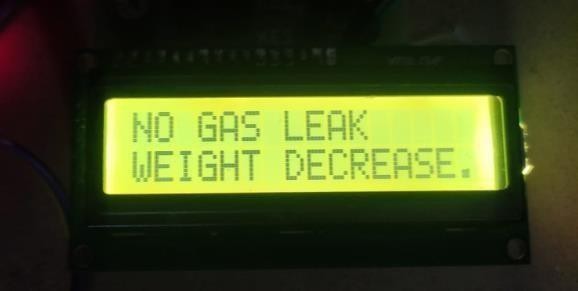
### RESULTS :



##### Fig 6.2.1:Working Model of The Project

The circuit is powered with a 12V battery. Arduino Uno microcontroller which is the main component of this work is programmed using embedded C language.LPG gas leakage is continuously measured with MQ2 gas sensor.When the sensor detects the gas leakage and then automatically the buzzer and cpu fan will ON,and the data will be uploaded to server and message will be sent to the respective number.

The load cell will monitor the weight of the cylinder,When it was decreased in weight then automatically the buzzer will ON.The following figures shows the operating setup for different conditions.

##### Fig :6.2.2 Display of No Gas Leak and Fig:6.2.3 Display of No Gas Leak and Normal Weight Weight Decreases

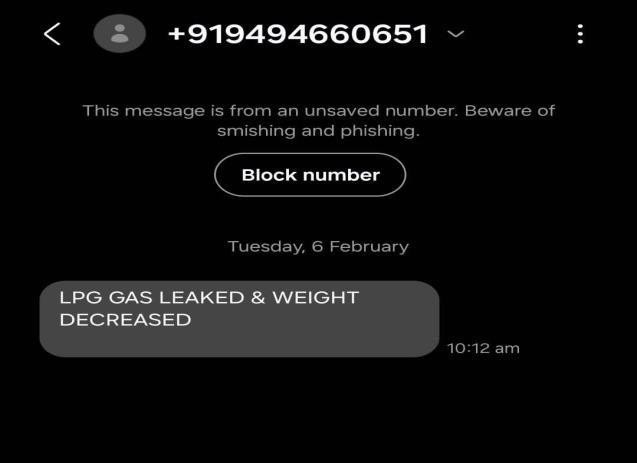


**Fig :6.2.4 Display of Gas Leak and Fig:6.2.5 Display of Gas Leak and**

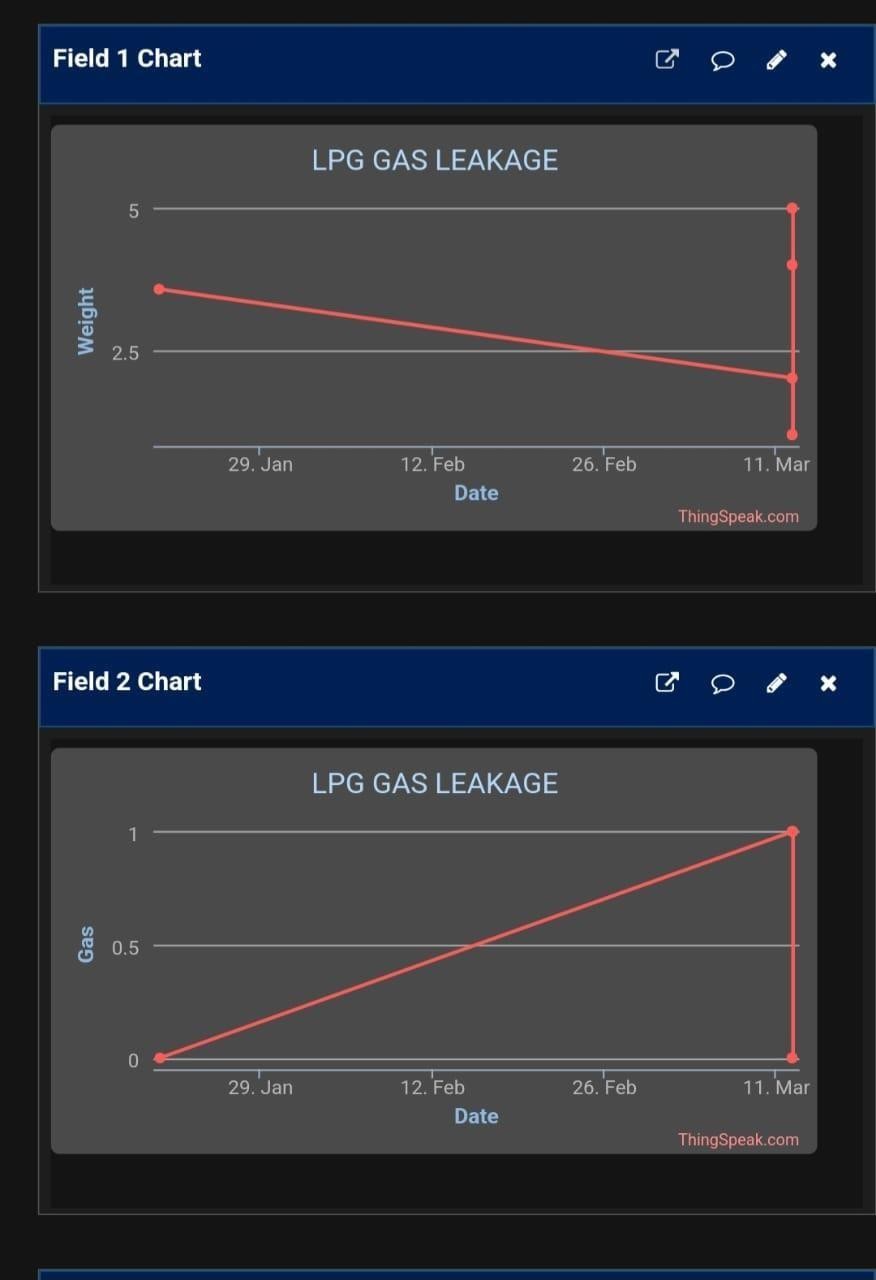
##### Weight Decreases Normal Weight



**Fig:6.2.6 Display of Data Uploading Fig:6.2.7 Display of Message Sent**



##### Fig :6.2.8 GSM Result



**Fig :6.2.9 Things Speak Result**

##### Condition 1:gas\_value==0&&weight>100

BUZ HIGH FAN LOW GAS LEAK

NORMAL WEIGHT

##### Condition 2:gas value==0&&weight<100

BUZ HIGH FAN LOW GAS LEAK

WEIGHT DECREASES

##### Condition 3:gas value==1&&weight>100

BUZ LOW FAN HIGH

NO GAS LEAK NORMAL WEIGHT

##### Condition 4:gas value==1&&weight<100

BUZ HIGH FAN HIGH

NO GAS LEAK WEIGHT DECREASES

# CHAPTER 7 CONCLUSION

In conclusion, the development of the IoT-enabled framework for LPG leakage detection and weight monitoring represents a significant advancement in safety, efficiency, and resource management across various sectors. Through the integration of smart sensors, communication technologies, and data analytics, the system offers real-time monitoring capabilities, proactive alerting mechanisms, and remote management features, ensuring enhanced safety and operational effectiveness.

One key takeaway from this project is the critical role of technology in mitigating risks associated with LPG usage. By leveraging IoT capabilities, stakeholders can gain unprecedented visibility into LPG systems, enabling early detection of leaks, timely response to anomalies, and proactive maintenance interventions. This proactive approach not only enhances safety for individuals and communities but also minimizes the potential for environmental damage and property loss.

Furthermore, the successful implementation of the IoT-enabled framework underscores the importance of collaboration between technology developers, industry stakeholders, and regulatory authorities. Continued research and innovation in this area can lead to further enhancements in LPG safety, efficiency, and sustainability. Moving forward, it is essential to prioritize the adoption of such advanced systems to safeguard lives, protect assets, and promote responsible use of LPG resources in both urban and rural settings. Overall, the project marks a significant step forward in leveraging IoT technology for enhancing safety and efficiency in LPG management, setting a precedent for future developments in the field.

### REFERENCES:

1. James Doorhy, “Real-Time Pipeline Leak Detection and Location Using Volume Balancing”, Pipeline & Gas Journal, February 2011.
2. Pal-Stefan Murvay, Ioan Silea, “A Survey on gas leak detection and localization techniques,” Journal of Loss Prevention in the Process Industries, vol. 25, no. 6, pp. 966-973, Nov. 2012.
3. Roy, Aashis S. Anilkumar, Koppalkar R.Sasikala, M.Machappa, T.Prasad, M.V. N. Ambika, “Sensitivity Enhancement for LPG Detection by Employing Cadmium Oxide Doped in Nanocrystalline Polyaniline”, Volume 9, Number 4, August 2011, pp. 1342-1348
4. Falkiner, RJ, “Liquefied Petroleum Gas”, Chapter 2, Jun 2003.
5. Tai-Yih Chen, Isobel J.Simpson, Donald R.Blake, F.Sherwood Rowland, “Impact of the leakage of liquefied petroleum gas (LPG) on Santiago Air Quality”, 2001
6. M.M. Sirdah, N.A. Al Laham and R.A. El Madhoun (2013), “Possible health effects of liquefied petroleum gas on workers at filling and distribution stations of Gaza governorates”, EMHJ, Vol. 19.
7. Erick D. Gamas, Moises Magdaleno , Luis Diaz, Isaac Schifter, Luis Ontiveros & G. Alvarez-Cansino (2000) Contribution of Liquefied Petroleum Gas to Air Pollution in the Metropolitan Area of Mexico City, Journal of the Air & Waste Management Association, 50:2, 188-198.
8. Kirk R Smitha, Jonathan M Sametb, Isabelle Romieuc, Nigel Bruced, “Indoor air pollution in developing countries and acute lower respiratory infections in children”, 2000, 55:518-532
9. Shivalingesh B.M, Ramesh C, Mahesh S.R, Pooja R, Preethi K. Mane, Kumuda S., “LPG Detection, Measurement and Booking System”, 2014
10. A. Mahalingam, R. T. Naayagi, N. E. Mastorakis. Design and Implementation of an Economic Gas Leakage Detector. 2012; ISBN: 978-1-61804-074-9 443.