IoT Based Air Quality Monitoring System

Submitted by

Md. Fatuham Mobin Rabby
ID: 191001185

Md. Ashikur Rahman
ID:191001025

Sujit Roy
ID:191001070

Department of Electrical and Electronic Engineering

GREEN UNIVERSITY OF BANGLADESH

February, 2023

IoT-Based Air Quality Monitoring System

Submitted by

Md.Fatuham Mobin Rabby

ID: 191001185

Md.Ashikur Rahman

ID:191001025

Sujit Roy

ID:191001070

Department of Electrical and Electronic Engineering

GREEN UNIVERSITY OF BANGLADESH

Supervised by

Md. Ahsanul Alam

Associate Professor

Department of Electrical and Electronic Engineering

A thesis submitted to the Department of EEE for the partial fulfillment of the degree of Bachelor of Science

February, 2023

Certificate

This is to certify that this thesis entitled "**IoT Based Air Quality Monitoring System**" is done by the following students under my direct supervision and this work has been carried out by them in the laboratories of the Department of Electrical and Electronic Engineering under the Faculty of Science and Engineering of the Green University of Bangladesh in partial fulfillment of the requirements for the degree of Bachelor of Science in Electrical and Electronic Engineering. The presentation of the work was held on February, 2023.

Signature of the Candidates

____

Md.Fatuham Mobin Rabby
ID: 191001185

____

Md.Ashikur Rahman
ID:191001025

____

Sujit Roy
ID:191001070

Signature of Supervisor

Md. Ahsanul Alam
Associate Professor
Department of EEE
Green University of Bangladesh

Dedication To

Our Parents

Abstract

Our project is made of physically challenged and disabled people using an "**IoT Based Air Quality Monitoring System**" IoT Technology is proposed to improve the quality of air. The use of IOT technology enhances the process of monitoring various aspects of the environment such as the air quality monitoring issue proposed in this paper. Here, using the MQ135 gas sensor gives a sense of the different types of dangerous gas and Arduino is the heart of this project, which controls the entire process. The Wi-Fi module connects the whole process to the internet users using the visual Output. It supports the new technology and effectively supports the healthy life concept. This system has features for people to monitor the amount of pollution on their mobile phones. The proposed outline includes a set of gas sensors (Mythen gas, Dust, temperature, and humidity) that are positioned on masses and structure of an IOT (Internet of things) and a dominant server to support both short-range real-time incident management and continuing deliberate planning. This provides a real-time low-rate monitoring system over the use of low rate, low information rate, and little control of wireless communication technology. The projected monitoring system can be transferred to or shared by different applications. Through IOT we can able to visualize the values from the globe.

Acknowledgment

We work for the **IoT Based Air Quality Monitoring System**. At the time of our study and research, we take help from several thesis and research papers from several journals which we mention and listed at the end of our thesis paper. Our honorable supervisor and teacher Md. Ahsanul Alam, Associate Professor, Department of EEE of the Green University of Bangladesh inspire us to study and research the topic of **IoT Based Air Quality Monitoring Systems.** He gives us vital instructions and guidelines for the whole period of our thesis. We the member of this group also try our best to do well as possible. Sir senior lecturer department of EEE of the Green University of Bangladesh inspire us to study and research the topic of floating power plants. He gives us vital instructions and guidelines for the whole period of our thesis. We the member of this group also try our best to do well as possible.

Contents

List of Figures

List of Tables

List of Acronyms

IoT		Internet of Things
PCB		Printed Circuit Board
MCU		Micro Controller Unit
IC		Integrated Circuit
Wi-Fi		Wireless Fidelity
AC		Alternating Current
DC		Direct Current
LCD		Liquid Crystal Display
LED		Light Emitting Diode
I/O		Input Output
USB		Universal Serial Bus
GND		Ground

Chapter 1
Introduction

1.1 Introduction

Air is the given name for the atmospheric gases used for respiration and photosynthesis. Dry air contains 70.09% nitrogen, 20.95% oxygen, 0.93% argon, 0.03% carbon dioxide, and a small number of other gases. [1] We have determined the amount of Ozone(O3), carbon-dai-oxide (CO2), Dust, Temperature, and humidity in the air in different places in Dhaka city, as well as observed from anywhere in the IOT. The Internet of Things (IoT) is nowadays finding profound use in each and every sector and plays a key role in our air quality monitoring system too. Our setup will show the air quality in PPM (Parts per Million) on a web page so that we can monitor it very easily. In this IoT project, one can monitor the pollution level from anywhere using a computer or a mobile. The global average atmospheric carbon dioxide in 2019 was 409.8 ppm. The 2008 ozone standard is set at a level of 0.075 ppm averaged over an 8-hour period. This standard is met at an air quality monitor when the 3-year average of the annual fourth-highest daily maximum 8-hour average ozone concentration is less than or equal to 0.075 ppm. The national ambient Dust Standard list is 0.053 ppm. Measuring Air Quality is an important element for bringing awareness to take care of future generations and for a healthier life. We are trying to implement a system using IoT platforms like Blynk in order to bring awareness to every individual about the harm we are doing to our environment.

1.2 Purpose of the Study

The project is an implementation of an IoT (Internet of Things) Based Air Pollution Monitoring System Using Arduino. Air pollution is a growing issue and it is necessary to monitor air quality for a better future and healthy living for all. IoT is getting more popular day by day and standards are on their way. Therefore, the collection of air quality information is easier. Analysis of monitoring data allows us to assess how bad air pollution is from day to day. According to a recent survey, Dhaka, the capital of Bangladesh is the third on the list of most air-polluted cities. Thus because of this expansion in the quantity of vehicles contamination is developing quickly and it influencing people groups well-being too. This air contamination makes disease and harms safe, neurological, regenerative, and respiratory framework. In extraordinary cases, it can likewise cause passing. As indicated by the overview 50000 to 100000 unexpected losses occurred to us only because of air contamination [2]. Along these lines, there is a requirement for checking air quality and monitoring it. IoT is the system of physical gadgets, vehicles, home apparatuses, and different things implanted with hardware, programming, sensors, and availability which empowers these articles to associate and trade information. IoT permits articles to be noticed or controlled. In this paper, I am proposing and going to pilot a model which IoT to screen air contamination.

1.3 Statement of the Problem

An increase in vehicle use gives rise to an increase in traffic-related pollutant emissions. According to science, the six common air pollutants are particulate matter, ground-level ozone, carbon monoxide, sulfur oxides, nitrogen oxides, and lead. These are called the criteria pollutants and thus are required to be measured to tell us how healthy the air is to breathe. Vehicular emissions contribute carbon monoxide, carbon dioxide, and nitrogen oxides to air pollution.

The current pollution measurement methodology uses expensive equipment at fixed locations or dedicated mobile equipment. The raw data obtained in this manner is used to further extrapolate the extent and concentration of pollution through dispersion models. This is a coarse-grained system where the pollution measurements are few and far in between. Widespread deployment of this measurement paradigm is constrained by its prohibitive cost. In addition, it is desirable to have access to real-time measurements to be able to quickly analyze and identify alarming levels of pollutants. Currently, access to such data is limited if not absent. It is available to and discernable by only a few who are well-informed on the subject of pollution.

.

1.4 Objectives

Our objectives are pointed out below:

- Design & Construct **IoT Based Air Quality Monitoring System**.
- Design & Implementation of air quality monitoring, automatic temperature measuring, and IOT monitoring.
- Collect various data for future modification.

1.5 Research Methodology

The methodologies we have used to build this project are given below:

- Creating an idea for the Design and construction of **IoT Based Air Quality Monitoring System.**
- Designing a block diagram & circuit diagram to know which components need to construct it.
- Collecting the all components and programming for the microcontroller to control the system.
- Setting all components in a PCB board & soldering. Then assemble the whole block in a board and finally run the system & check.

1.6 Project Outline

This Project is organized as follows:

Chapter 1 Introduction: The first chapter contains the statement of the introduction, our background study for the project, the objectives of the study, the methodology used in the project, and the project outline.

Chapter 2 Background and Motivation: Chapter two contains our literature review part.

Chapter 3 Methodology: Chapter three describes the theoretical model. Here we mainly discuss the proposed system architecture in details with having block diagram, circuit diagram, project working principle, complete project image, project instrument cost analysis, and discuss Hardware and software development of our project, etc.

Chapter 4 Performance Evaluation: Chapter four deals with the result and discussion and discuss our project's stages and application.

Chapter 5 Conclusion: Chapter five is all about our project conclusion, limitati,ons and future scope.

Chapter 2
Background and Motivation

2.1 Introduction

This chapter reviews some of the past works in processing and understanding IoT-based air pollution detection monitoring systems. Air pollution is not only a natural medical matter that impacts creating nations alike. This section is devoted to the literature used and studied in the preparation of this project. A literature review demonstrates an understanding of information processing solutions while addressing various findings from the literature.

2.2 Literature Review

Air is the most fragile ingredient of the ecosystem, and it has been degraded by the pollutants thrown in and out on a continuous basis. This proposed methodology is a wireless sensor network that works exclusively to monitor pollution in an indoor place in order to measure the amount of quality in the air. It's a minimal monitoring system with modest but functional sensors. Some preceding works, such as Air Quality Monitoring System Based on ISO/IEC/IEEE 21451 Standards [2], became introduced in 2016. It effectively evaluated the rate of toxin gas, which are responsible for air pollution.

To comprehend the toxic gasses and their influence, Air quality monitoring using Raspberry Pi based on IoT [3] was proposed. The design of a Sensor Network for Urban Micro-Climate Monitoring [4] was proposed which can monitor the air quality of the urban environment. IoT-based urban climate monitoring using Raspberry Pi [5] which is able to monitor the city environment using a raspberry pi system. Looking at recent data, we may assume that poor air quality has reached unprecedented levels. If it is not eradicated immediately, the entire region will be vulnerable to severe events in the future.

There are various other types of pollution, such as water contamination, excessive noise, industrial effluents, and soil contamination, but future studies may show that air pollution is the most serious issue that needs to be addressed in order to breathe safe air. According to the World Health Organization: WHO, air pollution presents a serious threat to human health and the environment, from pollutants hanging over cities to indoor smoking. Each year, the combined effects of environmental (outside) and domestic air pollution result in approximately 7 million premature deaths, the majority of which are caused by increased mortality from stroke, heart disease, chronic obstructive pulmonary disease, lung cancer, and acute respiratory infections. Over 80% of people who live in urban areas where air pollution is measured are exposed to levels of pollution that exceed the WHO's recommended limit of 10g/m3, with the largest exposures occurring in low- and middle-income countries.

Around the world, 9 out of 10 people breathe harmful pollutants. According to WHO, air pollution alone caused approximately 4.2 million deaths in 2016, while domestic air pollution from cooking with polluting fuels and technology caused approximately 3.8 million casualties during the same time period. So, the aim was to create a device that would alert individuals to the quantity of harmful air they were inhaling. This method incorporates past research studies that demonstrate how critical it is to work on such a topic. The basic objective was to create a gadget that is portable and easy to set up. The number of Android devices and internet users has skyrocketed. It would be a big help for humanity if they always are alert about the environmental quality they consume in every second. For detecting air pollution indoors, a detection device can be placed in the center of an indoor environment. The primary target is to determine the number of harmful gases emitted in a closed environment. These sensors will capture real data in real-time on various gases (air and water) existing in the environment - Carbon Monoxide, Carbon Dioxide, Ammonia, LPG, smoke, Nitrous Oxide, etc. Our proposed monitoring system will allow us to monitor the air quality through a smartphone application conveniently and also from a dedicated web server.

A wireless distributed mobile air pollution monitoring system using General Packet Radio Service (GPRS) sensors were reported. Advancements in wireless communication and sensor technology are rapidly changing the air pollution monitoring paradigm. The internet of things (IoT) also allows the creation of smart environments in which objects interact and cooperate with each other. A lot of improvements have been made to existing air pollution monitoring systems. For example, proposed a system for monitoring air quality at home. The system transmits sensor data wirelessly by making use of the “request and respond” protocol along with a combination of address and data-centric protocols. The system monitors the indoor air quality of a home and displays the sensor reading on a screen. The researchers employed an Unmanned Aerial Vehicle (UAV) based system to monitor the air pollution in areas with poor accessibility. The system was equipped with a Pixhawk autopilot for UAV control and a Raspberry Pi for sensing and collating air pollution data. An adaptive algorithm was used to analyze the gathered pollution data. A participatory sensor system for monitoring air pollution in Sydney was proposed by scientists.

In the proposed system, sensors were mounted on vehicles, and a mobile application was used to upload data to a centralized repository. The system gave information to the user about their private vulnerability to pollution in the air. Recursive Converging Quartiles (RCQ) algorithm was utilized to improve the efficiency of the wireless air pollution monitoring system. Recursive Converging Quartiles (RCQ) algorithm aggregates and eliminates data duplicates by removing invalid readings. This saves energy. The system consisted of sensor nodes and wireless communication links to a server. The sensor nodes collated data automatically and passed it on through the network to the server. The sensor nodes automatically forwarded data measured to the server the moment they received instruction from the system to do so. The means clustering algorithm for analyzing air pollution was proposed by the authors.

A comparative study was made between the proposed k-means algorithm and the probabilistic fuzzy c-means (PFCM) clustering algorithm in terms of exactness and process period. The authors submitted that the proposed K-means clustering algorithm yielded exact values within a fewer process period in comparison to other existing techniques. The authors proposed a model that showed the concentration of air pollutants in real-time. An optimal Wireless Sensor Network (WSN) was proposed for monitoring the level of contaminants in the air. The system was enhanced by utilizing a flow concept that gave a combined formulation of coverage and connectivity. The probabilistic sensors taken into consideration could handle multiple weather scenarios. There was also a need for the flow constraints to be formulated to ensure that the network remained conservative.

A sensor-based system to monitor air pollution employed the use of the Internet of Things (IoT) to enable data about air pollutants to be monitored online.

When the pollutants' level in a locality exceeded the standard air quality index, the sensor-based system shared the information via SMS with the public. An ambient real-time air quality monitoring system that consisted of numerous distributed monitoring stations that were connected wirelessly to a backend server using machine-to-machine communication was explained. The backend server assembled real-time data from the stations and reproduced them as information that can be delivered to users through web portals and mobile applications. However, cooperative, distributed, and energy-efficient communication protocols are required. Geographical search on social networks was used by the researchers to gauge the level of pollutants in the air. The assumptions made were evaluated on three continents of the planet. A minimum increment in the number of air pollution-related posts meant a rise in air pollution in that environment. Measured data were acquired online while processing and statistical analysis were performed offline.

2.3 Summary

The above has been discussed in detail in the past few kinds of literature which have given us a lot of motivation to do this project.

Chapter 3

Methodology

3.1 Introduction

In this chapter, we describe our project system description, block diagram, circuit diagram, flow chart, project working principle, final project view, and project instrument cost analysis.

3.2 Block Diagram

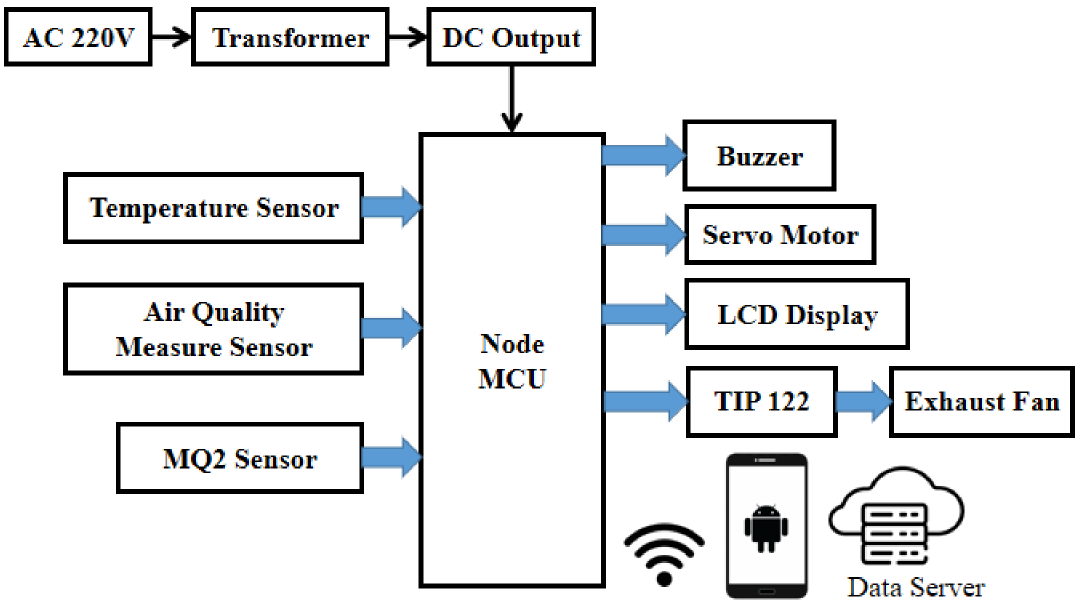


Figure 3.1: Block Diagram of our System

3.3 System Description

The complete circuit is built with Node MCU ESP8266 module, air quality monitoring system, MQ2 sensor, temperature and humidity sensor, relay, buck converter, loads & power supply. The ESP8266 microcontroller is the CPU of the system. It needs a 5V DC to power up. Constant 5V DC from the power supply is connected to VIN pin of the

ESP8266 board. A 16*2 LCD display is connected to a I/O of Node MCU and the transformer is the voltage provided to the microcontroller.

3.4 Circuit Diagram

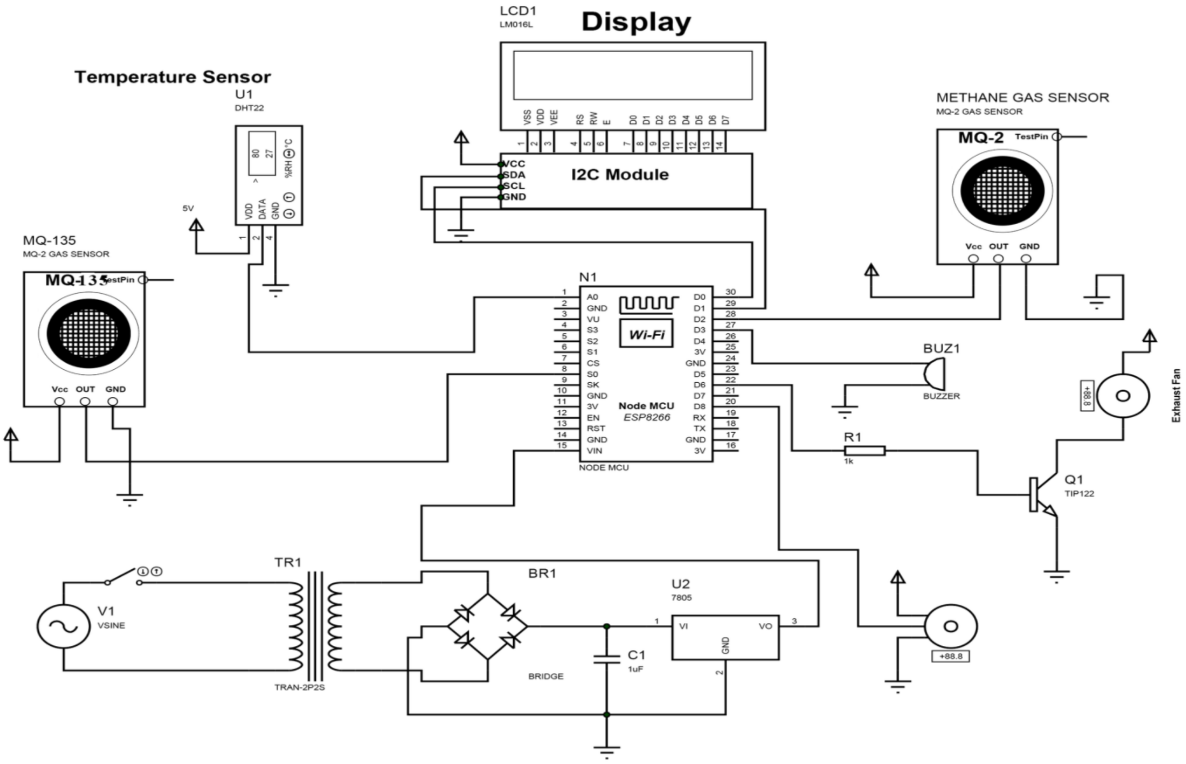
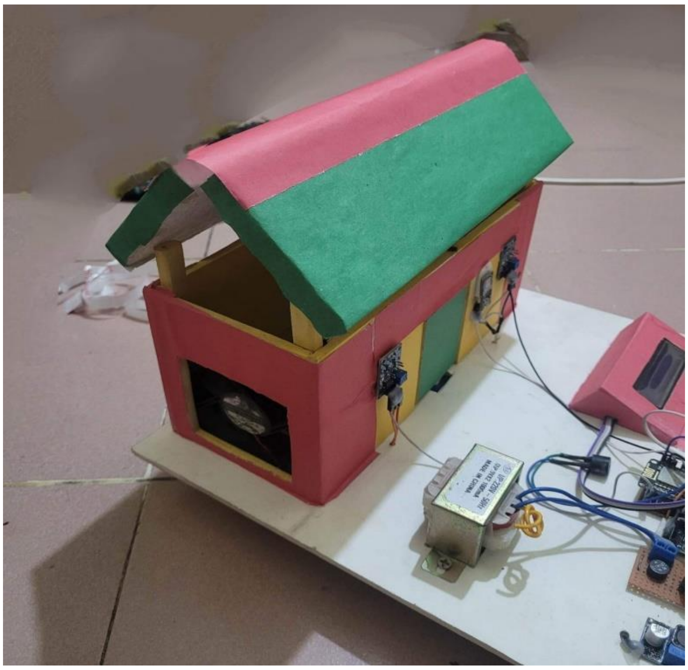


Figure 3.2: Circuit Diagram of our System.

3.5 Working Principle

The main brain of our system is the Node MCU. The way of whole project works is that we take 220V (rms) AC power from the supply voltage and then feed it to a transformer. The transformer simply converts the 220V AC to a pure DC of 12V. After that buck converter output is 5V DC. We will use this 5V DC output from the power supply system to run our microcontroller, sensor, and other units. Here Node MCU is connected to the transformer through Voltage Input (Vin) pin. The sensor collects the reading data and sends it to the controller for the next step. A temperature sensor detects the room's inside temperature and humidity, the MQ2 sensor detects the gas and the air quality sensor measures the air density level. If there create any abnormal situation inside the room Then it will ring a buzzer for alerting system. All kinds of operational output will be displayed in a LCD Display and a Mobile Apps Display. This is the main process of our system.

3.6 Complete Project Prototype



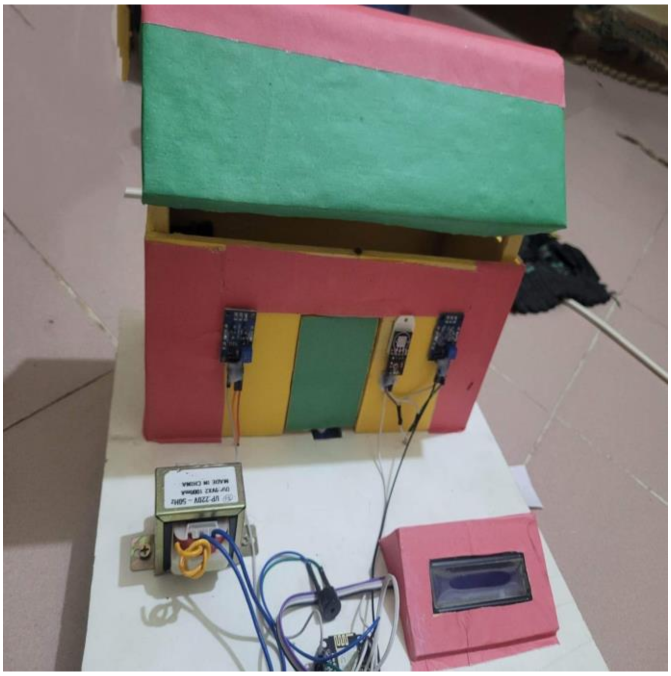


Figure 3.3: Our Final Project Image

3.7 About Hardware and Software

In this project we need some hardware and software for run this project smoothly.

- **Hardware**
    1. Node MCU
    2. Transformer
    3. Buck Converter
    4. Relay
    5. Temperature Sensor
    6. Smoke Sensor
    7. Air Quality Monitoring Sensor

8. Exhaust Fan
9. Servo Motor
10. LCD Display

- **Software**
  1. Arduino IDE
  2. Proteus
  3. Blynk app

3.8 Hardware

Electronic hardware consists of interconnected electronic components which perform analog or logic operations on received and locally stored information to produce as output or store resulting new information. Electronic hardware can range from individual chips/circuits to distributed information processing systems. Well-designed electronic hardware is composed of hierarchies of functional modules which inter-communicate via precisely defined interfaces.

3.8.1 Node MCU

Node MCU is an open-source firmware for which open-source prototyping board designs are available. The name "Node MCU" combines "node" and "MCU" (micro-controller unit). The term "Node MCU" strictly speaking refers to the firmware rather than the associated development kits. Both the firmware and prototyping board designs are open source. The firmware uses the Lua scripting language. The firmware is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open-source projects, such as lua-cjson and SPIFFS. Due to resource constraints, users need to select the modules relevant for their project and build a firmware tailored to their needs. Support for the 32-bit ESP32 has also been implemented.

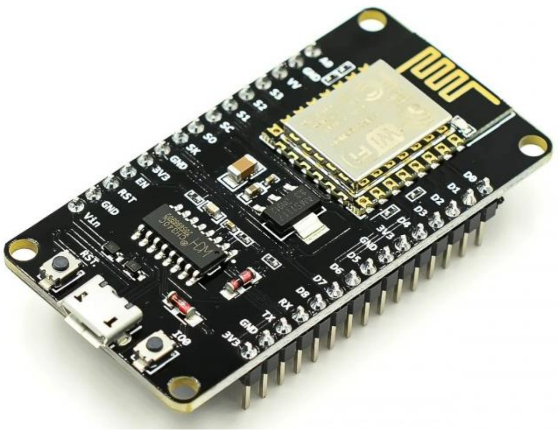


Figure 3.4: Node MCU

The prototyping hardware typically used is a circuit board functioning as a dual in-line package (DIP) which integrates a USB controller with a smaller surface-mounted board containing the MCU and antenna. The choice of the DIP format allows for easy prototyping on breadboards. The design was initially was based on the ESP-12 module of the ESP8266, which is a Wi-Fi SoC integrated with a Ten silica Xtensa LX106 core, widely used in IoT applications.

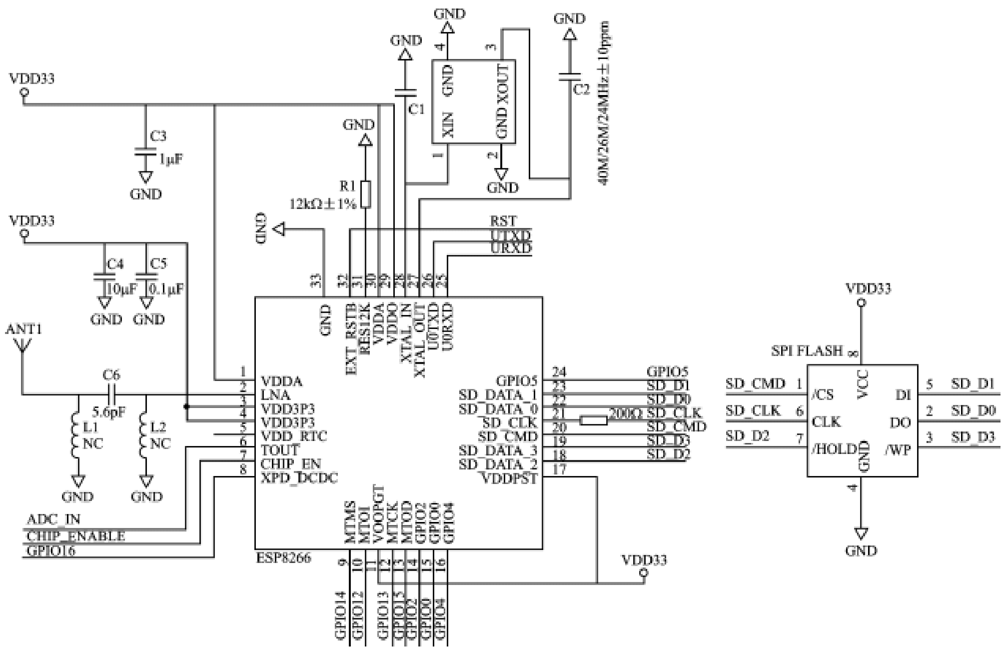


Figure 3.5: Node MCU Schematic Diagram

This an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Express if Systems, and hardware which is based on the ESP-12 module. The term "Node MCU" by default refers to the firmware rather than the development kits. The firmware uses the Luascripting language. It is based on the eLua project, and built on the Espress if Non-OS SDK for ESP8266. Node MCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems began production of the ESP8266. The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications (see related projects).

Node MCU started on 13 Oct 2014, when Hong committed the first file of node mcu firmware to GitHub. Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the gerber file of an ESP8266 board, named devkit v0.9. Later that month, Tuan PM ported MQTT client library from Contiki to the ESP8266 SoC platform, and committed to Node MCU project, then Node MCU was able to support the MQTT IoT protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus ported the u8glibto Node MCU project, enabling Node MCU to easily drive LCD, Screen, OLED, even VGA displays.

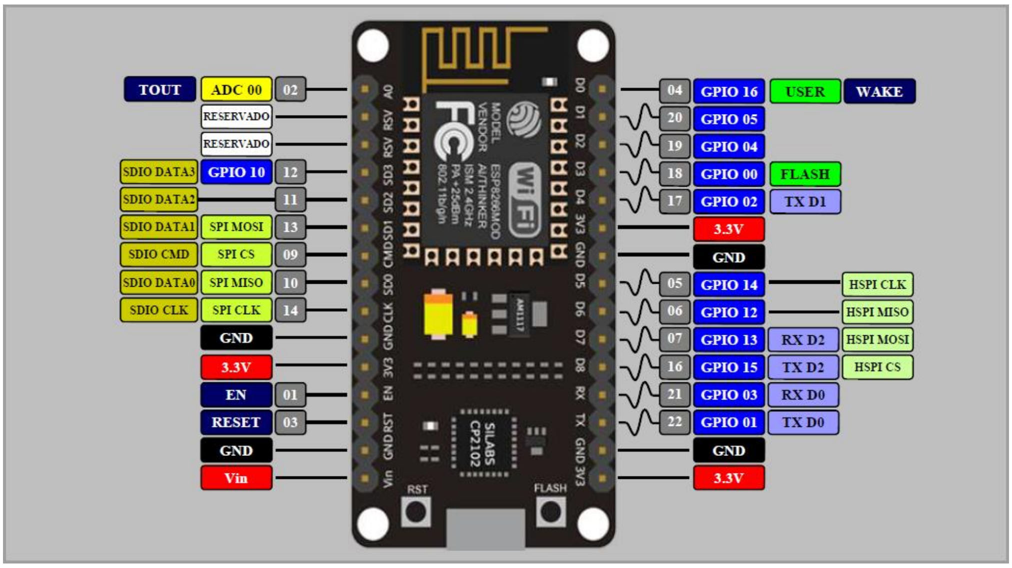


Figure 3.6: Node MCU Pin Out

Node MCU V3 ESP8266 ESP-12E is Wi-Fi development board that helps you to prototype your IoT product with few Lua script lines, or through Arduino IDE. The board is based on ESP8266 ESP-12E variant, unlike other ESP-12E, you won't need to buy a separate breakout board, USB to serial adapter, or even solder it to a PCB to get started, you will only need a USB cable (Micro USB).

Features

1. Communication interface voltage: 3.3V.
2. Antenna type: Built-in PCB antenna is available.
3. Wireless 802.11 b/g/n standard
4. WIFI at 2.4GHz, support WPA / WPA2 security mode
5. Support STA/AP/STA + AP three operating modes
6. Built-in TCP/IP protocol stack to support multiple TCP Client connections (5 MAX)
7. D0 ~ D8, SD1 ~ SD3: used as GPIO, PWM, IIC, etc., port driver capability 15mA
8. AD0: 1 channel ADC
9. Power input: 4.5V ~ 9V (10VMAX), USB-powered
10. Current: continuous transmission: ≈70mA (200mA MAX), Standby: <200uA
11. Flash size: 4MByte.

3.8.2 Relay

A relay is an electrically operated switch. Many relays use electromagnets to mechanically operate a switch, but other operating principles are also used, such as solid-state relays. Relays are used where it is necessary to control a circuit with a separate low-power signal, or where several circuits must be controlled by a single signal. Early relays were used as amplifiers in long-distance telegraphic circuits: they repeated signals from one circuit and retransmitted them onto another circuit. Interchange relays were widely used in early telephones and computers to perform logic functions. In below figure 3.7 we shown a relay physical overview and normal pin diagram of a relay.

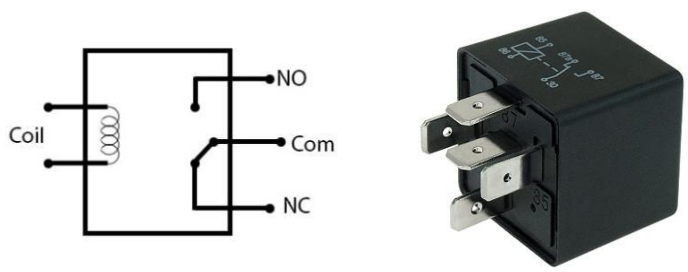


Figure 3.7: Relay

A type of relay capable of handling the maximum power required to directly control an electric motor or other load is called a contactor. Solid-state relays control power circuits with no moving parts, instead using a semiconductor device to perform switching. Relays with measured operating characteristics and sometimes multiple operating coils are used to protect electrical circuits from overload or hazards; in modern electrical power systems these operations are performed by digital devices still called "protective relays".

Magnetic latching relays require one coil of force to move their contacts in one direction, and another, to move the directed pulse back. Repeated pulses from the same input have no effect. Magnetic latching relays are useful in applications where interrupted power cannot permeate the contacts. Magnetic latching relays may have one or two coils. On a single coil device, the relay will operate in one direction when power is applied in one polarity and reset when the polarity is reversed. On a two-coil device, the contacts transition when the bias voltage is applied to the reset coil. AC controlled magnetic latch relays have a single coil that uses steering diodes to distinguish between operating and reset commands.

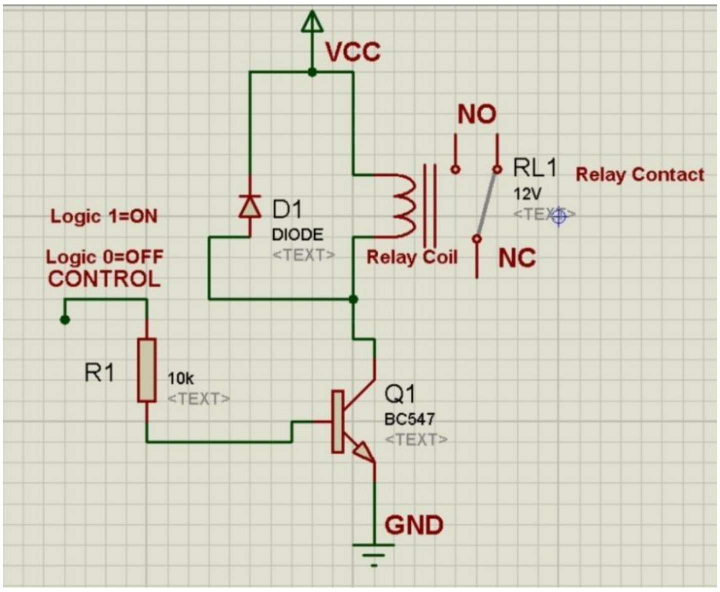


Figure 3.8: Transistor Switching Circuit.

Figure 3.12 the above circuit is called a low side switch, because the switch – our transistor – is on the low (ground) side of the circuit. Alternatively, we can create a high-side switch using a PNP transistor: Similar to the NPN circuit, the base is our input, and the emitter is connected to a constant voltage. A relay is an electrically operated high voltage switch. Whether or not current flows through it means it can turn on or off. Controlling a relay with an Arduino is as easy as controlling an output like an LED. The relay module is that shown in the figure below.

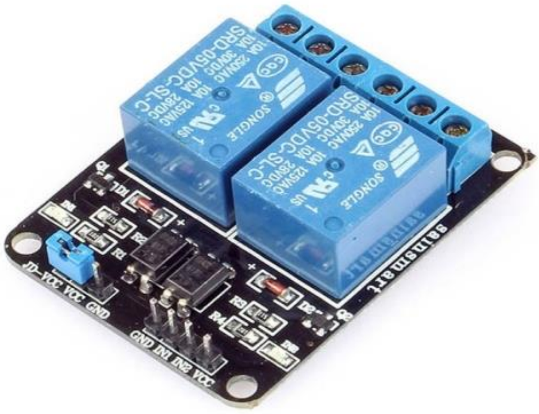


Figure 3.9: Relay Module.

Figure 3.13 this module has two channels (those blue cubes). There are other varieties with one, four and eight channels.

Mains voltage connections:

In relation to mains voltage, relays have 3 possible connections:

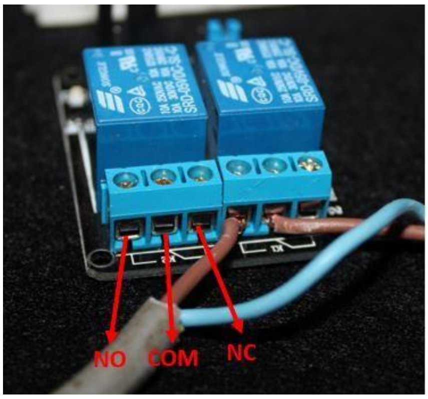


Figure 3.10: Pin diagram of Relay Module

COM: common pin

NO (Normally Open): there is no contact between the common pin and the normally open pin. So, when you trigger the relay, it connects to the COM pin and supply is provided to a load

NC (Normally Closed): there is contact between the common pin and the normally closed pin. There is always connection between the COM and NC pins, even when the relay is turned off. When you trigger the relay, the circuit is opened and there is no supply provided to a load.

Pin wiring:

Figure 3.15 the connections between the relay module and the Arduino are really simple:

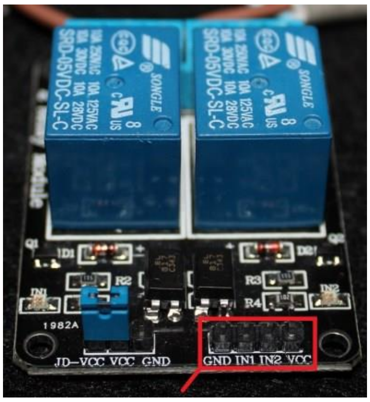


Figure 3.11: Main Voltage Connection

- **GND**: goes to ground
- **IN1**: controls the first relay (it will be connected to an Arduino digital pin)
- **IN2**: controls the second relay (it should be connected to an Arduino digital pin if you are using this second relay. Otherwise, you don’t need to connect it)
- **VCC**: goes to 5V

3.8.3 LCD Display

LCD (liquid crystal display) is an electronic display module and search applications. The 16x2 LCD display is a very basic module and is widely used in many devices and circuits. These sections are usually more than seven sections and many more. Here the physical overview of 16*2 LDC Display in below Figure 3.16.

Figure 3.12: 16*2 LCD Display

The reasons for having an LCD are economic; The characters can be easily learned, unique and even custom (seven different categories), there are no restrictions to display animations. A 16x2 LCD means that it can display 16 characters per line and consists of 2 lines. Each character on this LCD is displayed in a 5x7 pixel matrix. Which is shown in the below figure 3.17.

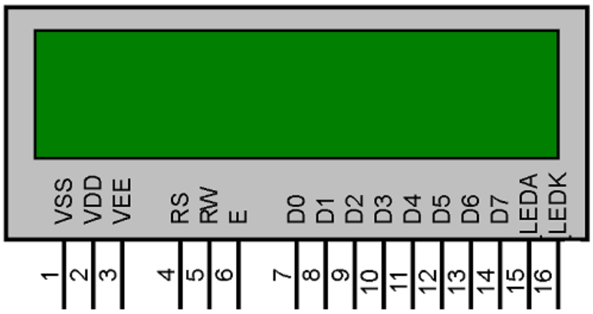


Figure 3.13: Pin out of 16*2 LCD Display

Pin No	Symbol	Level	Description
1	Vss	0 V	Ground
2	V_{DD}	5.0V	Supply voltage for logic
3	V o	variable	Operating voltagefor LCD
4	RS	H/L	H data/L instruction code
5	R/W	H/L	H/Read (MPU-module) L/write (MPU-module)
6	E	H, H-L	Chip enable signal
7	DB0	H/L	Data bit 0
8	DB1	H/L	Data bit 1
9	DB2	H/L	Data bit 2
10	DB3	H/L	Data bit 3
11	DB4	H/L	Data bit 4
12	DB5	H/L	Data bit 5
13	DB6	H/L	Data bit 6
14	DB 7	H/L	Data bit 7
15	A		Power supply for LED backlight (+)
16	K		Power supply for LED backlight (-)

Figure 3.14: 16*2 LCD Display Pin Description

Figure 3.18 describe the display pin, level, symbol description in details. This LCD consists of two parts namely Command and Data. The command register stores the commands for the

LCD. The command is given to the LCD to perform a predefined task such as initialization, clearing its screen, cursor search, control display, and more.

Features of LCD 16x2

The features of this LCD mainly include the following.

- □LCD power supply is 4.7V-5.3V
- □ It consists of two lines where each line can produce 16 letters.
- □ Current consumption is 1mA without backlight
- □ Every character can be built with a 5 × 8-pixel box
- □The alphanumeric LCDs alphabets & numbers
- □Is display can work on two modes like 4-bit & 8-bit
- □ These are available in Blue & Green Backlight
- □ Displays a few created colors

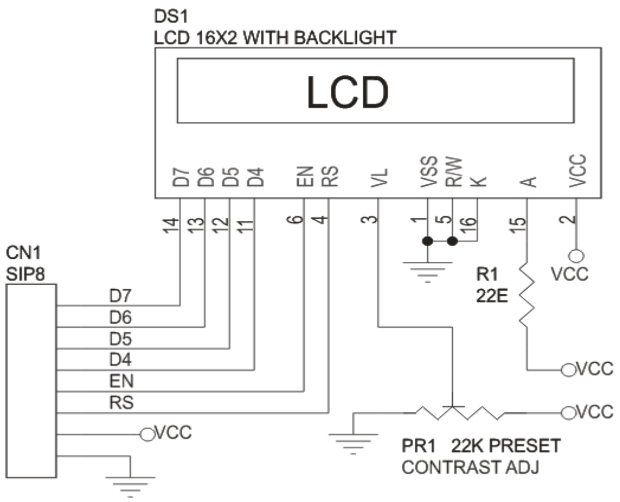


Figure 3.15: 16*2 LCD Display interfacing with Arduino

3.8.4 MQ 2 Gas Sensor

The utility model can be used for gas leakage monitoring devices in families and factories, and is suitable for the detection of liquefied petroleum gas, butane, propane, methane, Hydrogen, smoke, etc. This is a very easy to use low-cost semiconductor Gas sensor Module with analog and digital output.

Figure 3.16: MQ 2 Gas Sensor

Features:

- Adopt high quality double panel design, with power indication and TTL signal output indication.
- It has DO switch signal (TTL) output and AO analog signal output.
- TTL output valid signal is low level. When the output is low, the signal light is on, and the micro controller or relay module can be directly connected.
- The analog output voltage increases with the concentration, the higher the voltage.
- It has better sensitivity to liquefied petroleum gas, natural gas, urban gas and smoke.
- MQ-2 MQ2 Smoke Gas LPG Butane Methane Sensor Detector Module
- With four screw holes, easy to locate.
- Product size: 32 (L), *20 (W), *22 (H)
- With long service life and reliable stability.
- Fast response recovery features

Specifications:

- Input voltage: DC5V
- Power dissipation (current): 150mA
- DO output: TTL, numeric quantities 0 and 1 (0.1 and 5V)
- AO output: 0.1-0.3V (relatively pollution-free), the highest concentration of about 4V voltage

- Special reminder: after the sensor is energized, you need to preheat 20S or so, the data to be stable, sensor heating is a normal phenomenon, because the internal heating wire, if hot, it is not normal.

Connection mode:

- VCC: power supply positive (5V)
- GND: power supply negative pole
- DO: TTL switch signal output
- AO: analog signal output
- Functions: This version supporting test procedures
- Using chips: AT89S52
- Crystal oscillator: 11.0592MHZ

3.8.5 Temperature Sensor

The DHT11 is a basic, ultra-low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin (no analog input pins needed). It's fairly simple to use, but requires careful timing to grab data.

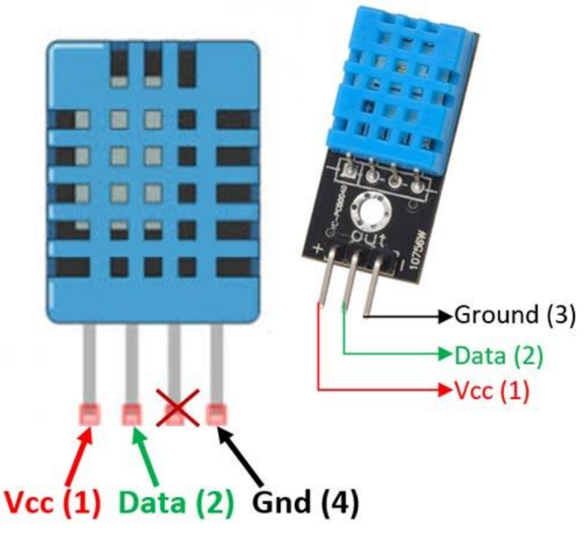


Figure 3.17: DHT11 Temperature Sensor

DHT11 Specifications:

- Operating Voltage: 3.5V to 5.5V
- Operating current: 0.3mA (measuring) 60uA (standby)
- Output: Serial data
- Temperature Range: 0°C to 50°C
- Humidity Range: 20% to 90%
- Resolution: Temperature and Humidity both are 16-bit
- Accuracy: ±1°C and ±1%

3.8.6 Buzzer

Piezo elements are actually one of the coolest crystals you'll play with. Flex them, they generate a voltage (pretty high, actually). Pass power through them, and they'll flex. Do it fast enough, and they'll start to make a sound! So besides making annoying little beep noises, you could use it as a sensor to detect a knock or other similar vibration.

Figure 3.18: Buzzer

Specifications

- Operating Voltage: 1.0 ~ 20.0V
- Rated Voltage: 3.0V
- Diameter: Ø12.5mm
- Total Height: 6.3mm

- Capacitance: 15,000pF
- Current Consumption: ≤2mA
- Resonance Frequency: 4,000Hz
- Sound Pressure Level at 10cm: ≥75db
- Self-Drive: No

3.8.7 MQ 135 Air Quality Monitoring System

MQ-135 is a gas sensor that has lower conductivity in clean air. It is low cost and suitable for different applications. This module operates at 5V, has 33Ω±5% resistance, and consumes around 150mA. This sensor has four pins from four pins, Digital and Analog output pins, which are used to approximate these gasses levels in the atmosphere. When these gasses go beyond a threshold limit in the air, the digital pin rises. This threshold value can be set by using the onboard potentiometer.

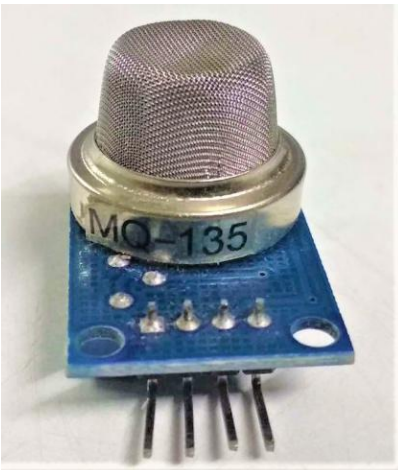


Figure 3.19: MQ 135 Sensor

Features of MQ-135 Sensor

1. Wide detecting scope, Fast response, and High sensitivity
2. Long lifespan
3. Heater Voltage: 5.0V
4. It contains analog output and high/low digital output
5. The TTL output signal is a low level
6. The operating Voltage is +5V

7. Detected/Measure NH_3, NO_x, alcohol, Benzene, smoke, CO_2, etc.
8. **Detection Range:** 10 – 300 ppm NH_3, 10 – 1000 ppm Benzene, 10 – 300 Alcohol

Working Mechanism

The MQ-135 Gas sensor consists of Tin Dioxide (SnO_2). When target pollution gas exists, the sensor's conductivity increases along with the gas concentration. Users can convert the change of conductivity to correspond to the output signal of gas concentration through a simple circuit. The MQ-135 gas sensor has a high sensitivity to NH_3, S_2, C_6H_6 series steam and can monitor smoke and other toxic gasses. It can detect kinds of toxic gasses.

Pin out of MQ135

The MQ-135 sensor module has four pins, with the most important part being an adjustable potentiometer.

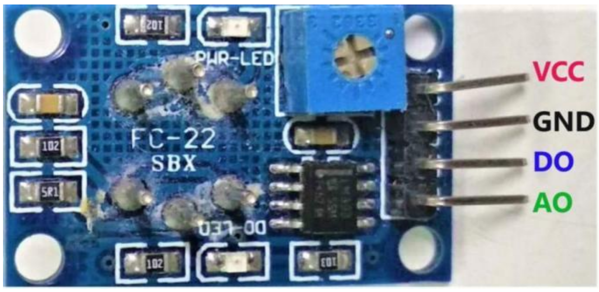


Figure 3.20: MQ 135 Pin Out

The MQ-135 Gas sensor can detect gases like Ammonia (NH_3), sulfur (S), Benzene (C_6H_6), CO_2, and other harmful gases and smoke. Similar to other MQ series gas sensor, this sensor also has a digital and analog output pin.

Table 01: MQ135 Pin Out

Pin Number	Pin name	Details
1	VCC Pin	The Pin requires 5V to power the module.
2	Ground Pin	To connect the module to the system's common Ground
3	Digital Output Pin	This pin sets the threshold value by using a potentiometer.
4	Analog Output Pin	The analog voltage pin is based on the concentration of the gas

3.8.8 Transformer

12-0-12 3Amp Center Tapped Step Down Transformer is a general-purpose chassis mounting mains transformer. Transformer has 230V primary winding and center tapped secondary winding. The transformer has flying colored insulated connecting leads (Approx. 100 mm long). The Transformer act as step down transformer reducing AC - 230V to AC - 12V. The Transformer gives outputs of 12V, 12V and 0V. The Transformer's construction is written below with details of Solid Core and Winding.

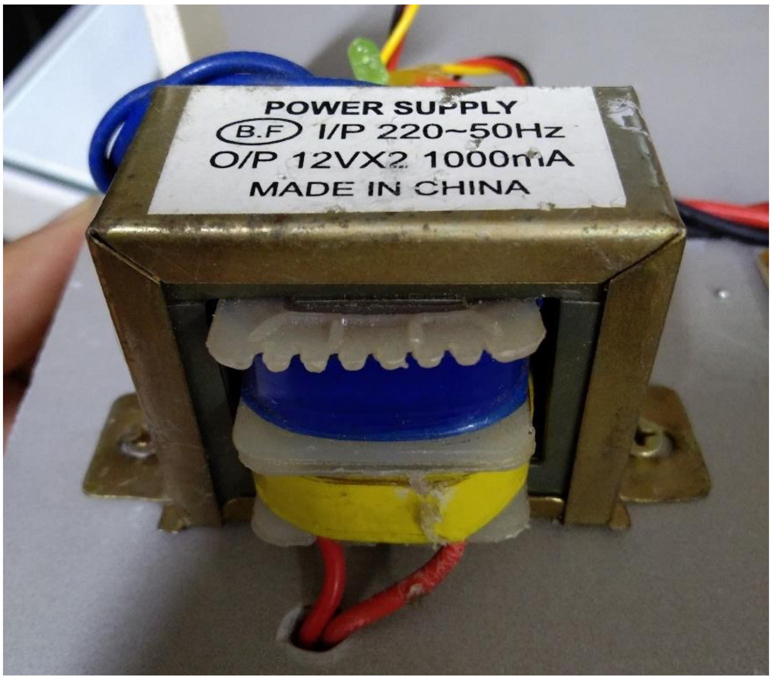


Figure 3.21: Transformer

The transformer is a static electrical device that transfers energy by inductive coupling between its winding circuits. A varying current in the primary winding creates a varying magnetic flux in the transformer's core and thus a varying magnetic flux through the secondary winding. This varying magnetic flux induces a varying electromotive force (E.M.F) or voltage in the secondary winding. The transformer has cores made of high permeability silicon steel. The steel has a permeability many times that of free space and the core thus serves to greatly reduce the magnetizing current and confine the flux to a path which closely couples the winding.

Specifications of 12-0-12 3 Ampere Center Tapped Transformer: -

- Input Voltage: 230V AC
- Output Voltage: 12V, 12V or 0V
- Output Current: 3 Amp
- Mounting: Vertical mount type
- Winding: Copper

Features of 12-0-12 3 Ampere Center Tapped Transformer: -

- Soft Iron Core.
- 3 Amp Current Drain.
- 100% Copper Winding

Applications of 12-0-12 3 Ampere Center Tapped Transformer: -

- DIY projects Requiring In-Application High current drain.
- On chassis AC/AC converter.
- Designing a battery, Charger.

3.8.9 The Full Wave Bridge Rectifier

Another type of circuit that produces the same output waveform as the full wave rectifier circuit above, is that of the **Full Wave Bridge Rectifier**. This type of single-phase rectifier uses four individual rectifying diodes connected in a closed loop "bridge" configuration to produce the desired output. The main advantage of this bridge circuit is that it does not require a special center tapped transformer, thereby reducing its size and cost. The single secondary winding is connected to one side of the diode bridge network and the load to the other side as shown below.

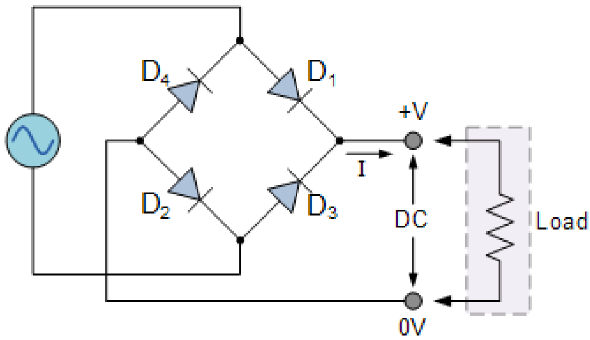


Figure 3.22: The Diode Bridge Rectifier

The four diodes labelled D_1 to D_4 are arranged in "series pairs" with only two diodes conducting current during each half cycle. During the positive half cycle of the supply, diodes D1 and D2 conduct in series while diodes D3 and D4 are reverse biased and the current flows through the load as shown below.

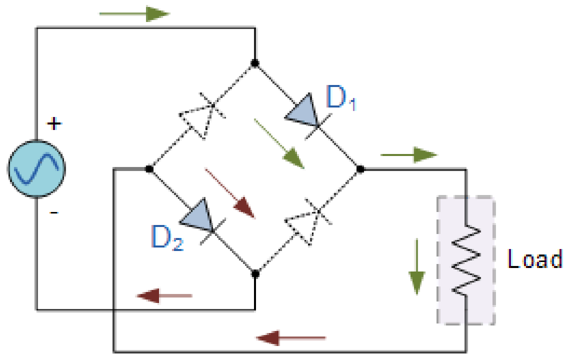


Figure 3.23: The Positive Half-cycle

During the negative half cycle of the supply, diodes D3 and D4 conduct in series, but diodes D1 and D2 switch "OFF" as they are now reverse biased. The current flowing through the load is the same direction as before. As the current flowing through the load is unidirectional, so the voltage developed across the load is also unidirectional the same as for the previous two diode full-wave rectifier, therefore the average DC voltage across the load is $0.637V_{max}$. However in reality, during each half cycle, the current flows through two diodes instead of just one so the amplitude of the output voltage is two voltage drops (2*0.7 = 1.4V) less than the input V_{MAX} amplitude. The ripple frequency is now twice the supply frequency (e.g., 100Hz for a 50Hz supply or 120Hz for a 60Hz supply.)

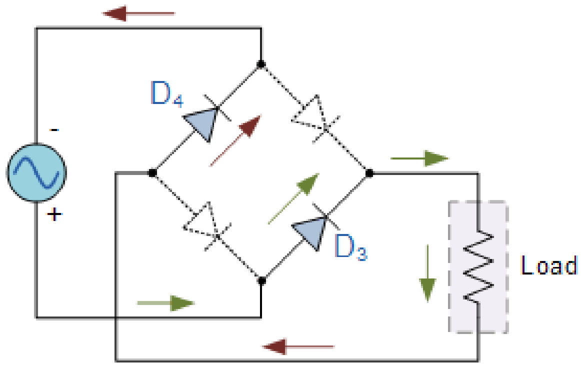


Figure 3.24: The Negative Half-cycle

Although we can use four individual power diodes to make a full wave bridge rectifier, pre-made bridge rectifier components are available "off-the-shelf" in a range of different voltage and current sizes that can be soldered directly into a PCB circuit board or be connected by spade connectors. The image to the right shows a typical single phase bridge rectifier with one corner cut off. This cut-off corner indicates that the terminal nearest to the corner is the positive or +ve output terminal or lead with the opposite (diagonal) lead being the negative or -ve output lead. The other two connecting leads are for the input alternating voltage from a transformer secondary winding.

3.8.10 The Smoothing Capacitor

We saw in the previous section that the single-phase half-wave rectifier produces an output wave every half cycle and that it was not practical to use this type of circuit to produce a steady DC supply. The full-wave bridge rectifier however, gives us a greater mean DC value (0.637 Vmax) with less superimposed ripple while the output waveform is twice that of the frequency of the input supply frequency. We can improve the average DC output of the rectifier while at the same time reducing the AC variation of the rectified output by using smoothing capacitors to filter the output waveform. Smoothing or reservoir capacitors connected in parallel with the load across the output of the full wave bridge rectifier circuit increases the average DC output level even higher as the capacitor acts like a storage device as shown below.

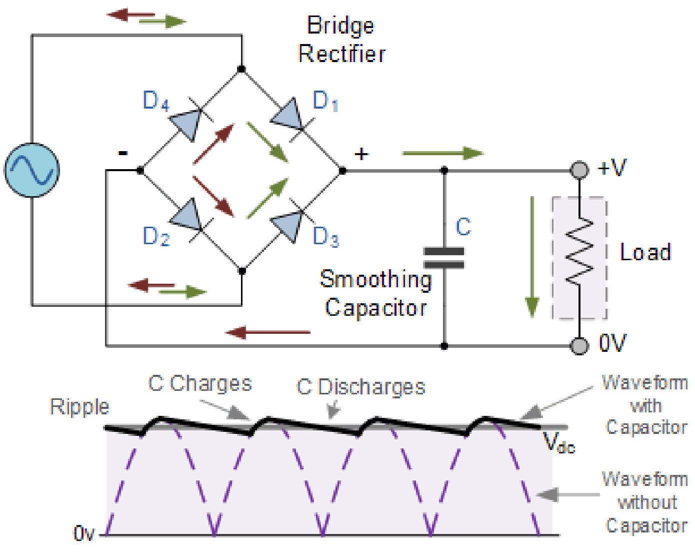


Figure 3.25: Full Wave Bridge Rectifier and output Wave form

The smoothing capacitor converts the full-wave rippled output of the rectifier into a smoother DC output voltage. If we now run the Partsim Simulator Circuit with different values of smoothing capacitor installed, we can see the effect it has on the rectified output waveform as shown.

3.8.11 Buck Converter

The converter (step-down converter) is a DC-to-DC converter that reduces the voltage from the input (supply) to the output (load). It is an intermittent power supply (SMPS) project that usually involves at least two semiconductors (including a diode and a transistor), a capacitor, an inductor or two. To reduce the voltage, filters made of capacitors (sometimes combined with inductors) are usually added to the output (single-pass filter) and input (supply-side filter) of such a converter.

Figure 3.26: DC -DC Buck Converter

DC-DC Buck Step Down Module LM2596 Power Supply is a 3-A load converter, capable of carrying a 3-A with a beautiful line and load pattern. These devices are available at an output voltage of 3.3 V, 5 V, 12 V, as well as a variable output voltage. The LM2596 system operates at a frequency change of 150kHz, thus allowing more precision filtering than would be required with high frequency converters.

Specifications of DC-DC Buck Converter Step Down Module LM2596 Power Supply :

- Conversion efficiency: 92% (highest)
- Switching frequency: 150KHz

- Output ripple: 30mA9max)
- Load adjustment: ± 0.5%
- Voltage adjustment: ± 0.5%
- Dynamic response rate: 5% 200 us
- Input voltage: 4.75-35V
- Output voltage: 1.25-26 V (adjustable)
- Output current: rated current 2A, maximum 3A (additional heat sink required)
- Conversion efficiency: up to 92% (higher output voltage, higher efficiency)
- Switching frequency: 150KHz
- Rectifier: Asynchronous straightening
- Module properties: non-isolated top-down module (fee)
- Short circuit protection: current limitation since recovery
- Operating temperature: industrial grade (-40 to +85) (output power 10 W or less)

3.8.12 Resistor

A resistor is a two-phase transmitter that acts as an electrical circuit as a circuit. Opponents are taking steps to reduce current, and, in some cases, actions to reduce voltage levels within the circuit. Resistors can have stability or resistance, such as thermostats, heaters, trimmers, photo resistors, hamsters and potentiometers. The current from the resistor drops to the voltage across the resistor terminals. Ohm's law represents this relationship.

Figure 3.27: Resistor

Theory of operation

The motion of an ideal resistor is determined by the relationship defined by Ohm's law:

$$V = I.R$$

According to Ohm’s law, the voltage (V) in a resistor is proportional to the current (I), where the proportionality constant is the resistance (R).

Similarly, Ohm's law can also be defined as:

$$I = V / R$$

This word indicates that current (I) is proportional to voltage (V) and inversely proportional to resistance (R). It is used directly in practical calculations. For example, if a 300-ohm resistor is connected across a 12-volt battery terminal, 12/300 = 0.04 amps will flow through this resistor.

3.9 Software

Software is a set of instructions, data or programs used to operate computers and execute specific tasks. It is the opposite of hardware, which describes the physical aspects of a computer.

3.9.1 Arduino IDE

A digital micro controller unit called the Arduino Nano can be programmed with the Arduino software IDE. No need to install other software than Arduino. First, Select "Arduino Nano from the Tools, Board menu (according to the micro controller on our board). The IC used called ATmega328 in the Arduino Nano comes pre-burned with a boot loader that allows us to load new code into it. without using hardware external programmer.

Communication uses the original STK500 protocol (reference, C header files). We can also skip the boot loader and program the microcontroller via the ICSP (In-Circuit Serial Programming) header. ATmega16U2 (or 8U2 for rev1 and rev2 boards) firmware source code is available. The ATmega16U2/8U2 is loaded with the DFU boot loader, which can be activated by: For Rev1 boards: connecting the solder jumper on the back of the board (next to the Italy map) and reset 8U2. On Rev2 or later boards: there is a resistor that pulls the 8U2/16U2 HWB line down, making it easier to put it in DFU mode.

The Arduino Nano is one of the latest digital microcontroller units and has a number of tools for communicating with a computer, another Arduino, or other microcontrollers. ATmega328 provides UART TTL at (5V) for serial communication, available on digital pin 0 -(RX) for data and pin no.1 (TX) for data transfer. The ATmega16U2 on board channels this serial communication via USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses standard USB COM drivers, and no external driver is required. However, on Windows, the .in file is required. The Arduino software includes a serial monitor that allows simple textual data to be sent to and from the Arduino board.

The RX and TX LEDs on the board will light up when data is being transferred via the USB-to-serial chip and the USB connection to the computer (but not via serial communication on pins 0 and 1). The Software Serial library allows serial communication to any of Nano's digital pins. The ATmega328 also supports I2C (TWI) and SPI interfaces. The Arduino software includes a Wire library to facilitate the use of the I2C bus. Arduino programs are written in C or C++ and the program code written for the Arduino is called a sketch. The Arduino IDE uses a series of GNU tools and the AVR lab to compile programs, and loading programs using them is controversial. Since the Arduino platform uses Atmel microcontrollers, Atmel's development environment, AVR Studio or the new Atmel Studio, can be used to develop Arduino software figure below 3.24.

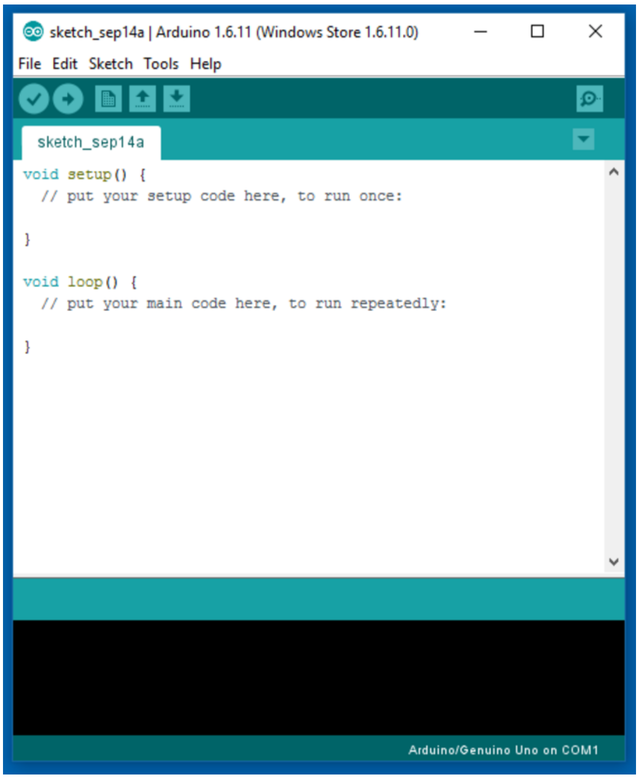


Figure 3.28: Arduino Software Interface IDE

The Arduino Integrated Development Environment - or Arduino Software (IDE) - includes a text editor, a message field, a text console, a button and a toolbar for common functions to write code. It connects to Arduino and Real Devices to download and communicate with applications.

Writing Sketches

A program written using the Arduino Software (IDE) is called a sketch. These diagrams are written in a text editor and saved with the file extension. ino. The editor has features for cut/paste and for searching/replacing text. The message area provides feedback when saving and exporting and also displays errors. The controller displays text output from the Arduino Software (IDE), including detailed error messages and other information. The lower right corner of the window displays a custom dashboard and ports. The Tools button allows you to browse and load programs, create, open, and save sketches, and open the viewer.

Sketchbook

Arduino Software (IDE) uses the sketchbook concept: a common place to store your programs (or sketches). Sketches in your sketchbook can be opened from the File > Sketchbook menu or from the Open button on the toolbar. The first time you run the Arduino software, it will automatically create a directory for your sketchbook. You can view or change the location of the sketchbook location from via the Options dialog. Starting with version 1.0, files are stored with the .ino file extension. Previous versions use the .pde extension. You can still open .pde named files in version 1.0 and later, the software will automatically rename the extension to .ino.

Tabs, Multiple Files, and Compilation

Allows you to manage sketches with more than one file (each of which appears in its own tab). These can be normal Arduino code files (no visible extension), C files (.c extension), C++ files (.cpp), or header files (.h).

Uploading

Before you load your sketch, you will need to select the correct items from the Tools > Board and Tools > Port menus. The boards are illustrated below. On a Mac, the serial port is probably something like /dev/tty.usbmodem241 (for a Uno or Mega2560 or Leonardo) or /dev/tty.usbserial-1B1 (for a Duemilanove or earlier USB board), or /dev/tty .USA19QW1b1P1.1 (for the serial board connected to the Keyspan USB-to-Serial adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, look for the USB serial device in the ports section of Windows Device Manager. On Linux, this should be /dev/ttyACMx , /dev/ttyUSBx or similar. Once you have selected the correct serial port and board, press the upload button on the toolbar or select the Upload item from the Sketch menu. Existing Arduino boards will automatically reset and start uploading. On older boards (pre-Diecimila) without auto-reset, you need to press the reset button on the board just before starting the upload.

Libraries

The library provides additional functions for use in sketches, e.g. working on hardware or manipulating data. To use an existing library in a sketch, select it from the Sketch menu > Import Library. This will place one or more #include statements above the drawing and compile the library with your drawing. Because your drawings are uploaded to the board, they increase the space it takes up. If the library is no longer needed by Graphics, delete its #include statement from the top of your code.

There is a library list and references. Some libraries are included in the Arduino software. Others can be downloaded from various sources or from the Library Manager. Starting with version 1.0.5 of the IDE, you can import libraries from zip files and use them in open graphics. See these instructions for installing third-party libraries.

Third-Party Hardware

Support for other hardware can be added to the hardware directory of your directory. The installed platform may include board definitions (which appear in the board menu), core libraries, bootloaders, and program definitions. To install, create a hardware directory, and unzip the third-party platform into its own directory. (Do not use "Arduino" as a sub-directory or you will disturb the installation on the Arduino platform.) To remove, simply delete the registry. For details on creating packages for 3rd party hardware, see Arduino IDE 1.5 3rd party Hardware details.

Serial Monitor

This indicates the series sent from the Arduino or USB or serial connector from the original board. Enter text to send information to the board and press the "Send" button or press Enter. Select the price level from the drop-down menu that matches the price for the serial. Start your sketch. Keep in mind that the board will restart (restart your thumbnail) when connected to a serial monitor on Windows, Mac, or Linux. Keep in mind that the Serial Monitor does not operate the control icons; If your sketch needs full control over the regular communication with the control symbols, you can connect it to the COM port assigned to the Arduino board using the external terminal program.

3.9.2 Proteus Software

The Proteus Design Suite is primarily a suite of proprietary software tools used to automate electronic design. The software is mainly used to create schematics and electronic tracks for the production of electronic boards published by electronic design engineers and technician.

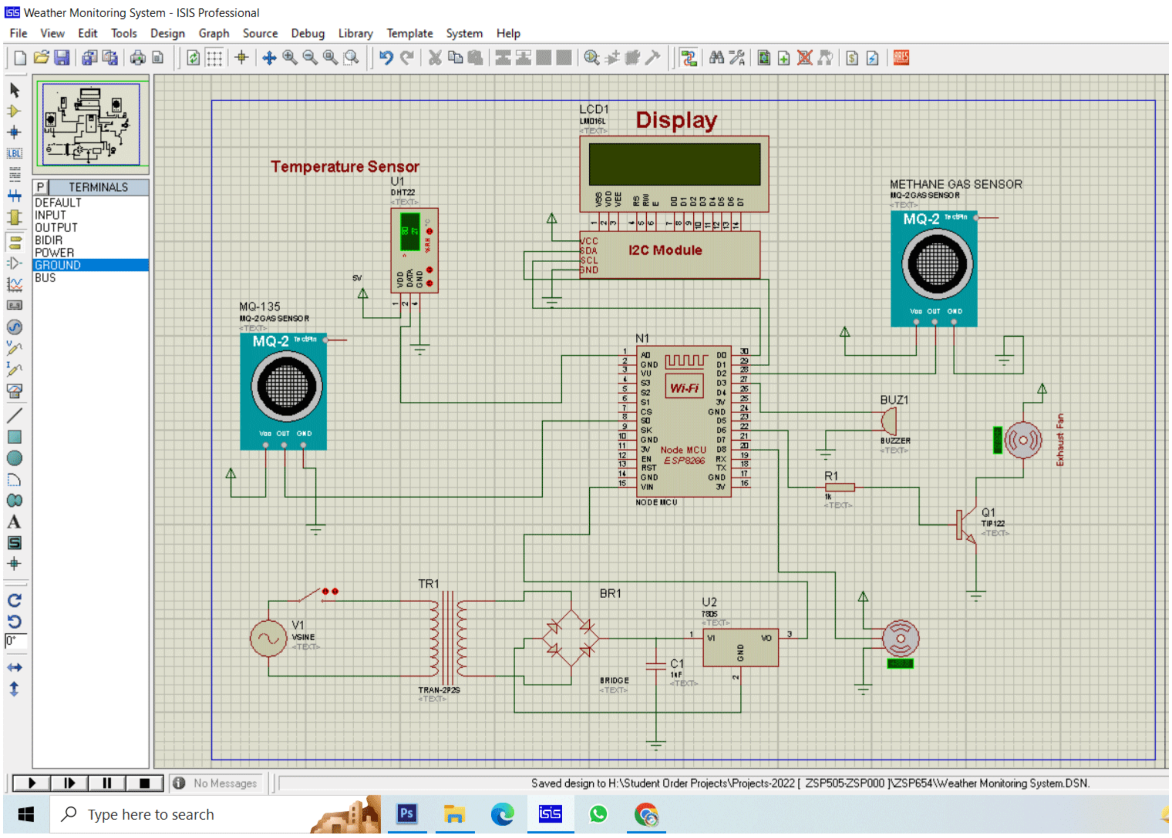


Figure 3.29: Proteus Software Interface

The first version of the so-called Proteus Design Suite is now called PC-B, and was written in 1988 for DOS by John Jameson, the company's CEO. Schematic Capture support was ported to the Windows environment in a short time in 1990. The mixed re SPim SPICE simulation was first integrated into Proteus in 1996, and the microcontroller simulation came to Proteus in 1998. In 2002 and 2006, the iconic car route was added and there was a product upgrade with 3D Board Visualization. Recently a special IDE for simulation was recently added in 2011 and in 2015 MCAD import / export was added.

3.9.3 Blynk Apps

Blink is an Internet of Things platform designed to make smart IoT devices faster and easier. Can be used to read, store, and visualize sensor data, and to remotely control devices.

Figure 3.30: Blynk Apps

In recent times, the internet has become more and more popular, and more and more devices are talking to the Internet every day. With the advancement of such a great technology, the security risks have also increased significantly. Some of the key concerns in IoT:

- If IoT devices are sending your data to the internet, the communication needs to be closed and encrypted which cannot be possible without using a dedicated and closed server which is really hard to manage.

- The IoT devices also need to be responsive and again, that is not possible without a server with low latency and high responsiveness.
- In IoT, the platform needs to be compatible with many different types of hardware architecture and devices, so that it doesn’t restrict its users with single type of hardware with limited capabilities.

Keeping in view the problems mentioned above figure 3.26, Blynk is the perfect solution for all these problems. Blynk consists of the following three major components:

Blink Software - A portable software developed by Blink that acts as a control panel to view and control your device. Also available for Android and iOS. The program offers a very productive interface and a variety of widgets for different purposes. Blink operates in its so-called energy currency. New users receive 2,000 free Blink energy with a free Blink account, and this energy is used to buy and place widgets on projects.

Blink server - Blink server is the most important part of the Blink platform that makes it all possible. Blynk provides a reliable, efficient and centralized cloud service that allows this communication between devices through the server. The Blynk server is also available as an open source, so you can make your server more complete by building your word in the full sense of the word.

Blink Library - The main feature of the Blink platform that can be expanded and enhanced is the Blink library. The Blink library allows you to connect your device, get up, and perform flashing. Support for several devices, such as the Arduino, ESP8266, and Raspberry Pi, is housed in the library and allows you to connect to the device through various communication methods such as Wi-Fi, Bluetooth, BLE, USB, and GSM.

Features

- Similar API & UI for all supported hardware & devices
- Connection to the cloud using:
    - Wi-Fi
    - Bluetooth and BLE
    - Ethernet
    - USB (Serial)

- GSM
- Set of easy-to-use Widgets

- Direct pin manipulation with no code writing
- Easy to integrate and add new functionality using virtual pins
- History data monitoring via Super Chart widget
- Device-to-Device communication using Bridge Widget
- Sending emails, tweets, push notifications, etc.

Chapter 4

Performance Evaluation

4.1 Introduction

In this chapter we will discuss the important part of our project. Here is a detailed discussion of the results, advantages, applications and discussions of our project.

4.2 Result

Now, it’s time to talk about the results. We have written our commands using the Arduino IDE and the following things can happen: Our project is **IoT Based Air Quality Monitoring System**. In our project making we used PVC boards for total hardware making. After finally completing this project, we run it & we observed the output of this project. We can see that it is working well as expected.

- After power up the project will be able to work its function.
- The system measures the air quality, tries to detect smoke, and measures the temperature and humidity.
- Our system control load with the help of IoT.
- The load can control from mobile apps through IoT.
- All data will show on LCD Display.

4.3 Advantages

There are certainly many advantages of our project and some of the major ones have been given below:

- It is use for measure air quality, temperature, humidity and smoke.
- For use load control far away from house .
- IoT based system for long distance monitoring facility.
- The whole system consumes very little energy.
- The system can be implemented anywhere with very little effort.
- Requires low maintenance.
- Cost effective.

4.4 Disadvantages

Every project has some limitation. So, some limitations are given below:

- This system must be connected with internet for control with IoT system.
- This project is a demo project, so it takes a few times to view the various data on the display.

4.5 Applications

Our project has many applications. These are –given bellow:

- To use in residential area.
- Hospital
- Shopping Mall
- Industries
- Commercial Buildings

4.6 Discussion

While working on our project, we did face some difficulties as it is a very complex system but the end results, we came up with were quite satisfactory. We have put the whole system through several tasks to validate our work and also have taken necessary notes for future improvements. Some future recommendations that we have involve improvement in system design and wiring, adding features for more efficiency.

Chapter 5
Conclusion

5.1 Conclusion

Detection of temperature, humidity, air quality is a major role in home, industry and air quality monitoring. The system we are implementing is rather self-same humble as likened to previous and existing air excellence intensive automobile schemes. This design has the advantages of stability and low-power consumption and self-sustained. The users can monitor real-time information and observe the changes in the information. This design will also play a significant role when the atmospheric conditions of a given area need to be checked which is not convenient for humans to measure.

5.2 Future Scope of Work

We are thinking about adding many features to our project in the future to get more desirable outcomes. Some of the steps that we are thinking about taking are given below:

- In the future, we are thinking about solving the delay data show the problem.
- In the future, we are thinking about adding more sensors for more data monitoring.
- In the future, we are thinking about controlling the Mythen gas Open and Off handle when the gas leakage, so that we sensors can sensing the gas and transform into electrical signal and can controlled.

References

[1] Ch.V.Saikumar, M.Reji, P.C.Kishoreraja, “IOT Based Air Quality Monitoring System”, International Journal on Information Theory (IJIT), Vol-117, No.-9, 2017;

[2] Riteeka Nayak, Malaya Ranjan Panigrahy , Vivek Kumar Rai and T Appa Rao “IOT based air pollution monitoring system”, International Journal on Information Theory (IJIT) Vol-3, Issue-4, 2017;

[3] Poonam Pal, Ritik Gupta, Sanjana Tiwari, Ashutosh Sharma, “Air Pollution System Using Arduino”, International Journal on Information Theory (IJIT), Vol-04, Issue-10, 2017;

[4] D.Arunkumar, K.Ajaykanth, M.Ajithkannan, M.Sivasubramanian, “Smart Air Pollution Detection And Monitoring Using IoT”, International Journal on Information Theory (IJIT) Vol-119, No.-15, 2018;

[5] Shanzhi Chen, Hui Xu, Dake Liu, Bo Hu, and Hucheng Wang, “A Vision of IoT: Applications, Challenges, and Opportunities with China Perspective”, IEEE INTERNET OF THINGS JOURNAL, VOL.-1, NO.-4, August 2014;

[6] S. Chen, H. Xu, D. Liu, B. Hu and H. Wang, "A Vision of IoT: Applications, Challenges, and Opportunities with China Perspective," in IEEE Internet of Things Journal, Vol-1, No.-4, 2014;

[7] Ms. Sarika Deshmukh, Mr.Saurabh surendran and Prof.M.P. Sardey, “Air and Sound Pollution Monitoring System using IoT” International Journal on Information Theory (IJIT), Vol-5, Issue-6, 2017;

[8] Navreetinder Kaur,Rita Mahajan and Deepak Bagai, “Air Quality Monitoring System based on Arduino Microcontroller”, International Journal on Information Theory (IJIT), Vol.-5, Issue-6, June 2016;

[9]. Palaghat Yaswanth Sai, “An IoT Based Automated Noise and Air Pollution Monitoring System”, International Journal on Information Theory (IJIT), Vol.-6, Issue-3, March 2017;

[10] L.Ezhilarasi, 2 K.Sripriya, 3 A .Suganya , 4 K.Vinodhini, “A System for Monitoring Air and Sound Pollution using Arduino Controller with IOT Technology”, International Journal on Information Theory (IJIT), Vol.-3 Issue-2, 2017;

[11] Devahema, P.V. Sai Surya Vamsi, Archit Garg, Abhinav Anand, Desu Rajasekhar Gupta, “IOT based Air Pollution Monitoring System”, Journal of Network Communications and Emerging Technologies (JNCET), Vol-8, Issue-4, 2018;

Appendix

Program Code:

Program is the brain of our project. Arduino is an open-source computer hardware which works on the program's instructions.

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27 ,16,2);
#define BLYNK_TEMPLATE_ID "TMPLXdQsIJ9V"
#define BLYNK_DEVICE_NAME "project"
#define BLYNK_AUTH_TOKEN "xv90VIpXc9F_MB_Y56dYHHim7i9ZGmat"
#define BLYNK_PRINT Serial
char auth[] = BLYNK_AUTH_TOKEN;

#include "DHTesp.h"
DHTesp dht;

#include <Servo.h>
Servo myservo;  // create servo object to control a servo

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

int smoke = D5;
int relay = D7;
int buzzar = D0;
int mq135 = A0;
int mq1355;

char ssid[] = "abcde";
char pass[] = "12345678";

void setup()
{
  Serial.begin(115200);

lcd.init();
lcd.backlight();
lcd.begin(16,2);
lcd.clear();
```

```
  myservo.attach(D6);  // attaches the servo on GIO2 to the servo object


pinMode(smoke, INPUT);
pinMode(mq135,INPUT);
pinMode(relay, OUTPUT);
pinMode(buzzar, OUTPUT);
myservo.write(0);
 dht.setup(2, DHTesp::DHT22);
 //Blynk.begin(auth, ssid, pass);
  // You can also specify server:
  Blynk.begin(auth, ssid, pass, "blynk.cloud", 80);
  //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8080);
}
WidgetLCD lcd1(V0);
WidgetLCD lcd2(V3);
void loop()
{
Blynk.run();

int smoke_sensor = digitalRead(smoke);
mq1355 = analogRead(mq135);
delay(dht.getMinimumSamplingPeriod());

  float humidity = dht.getHumidity();
  float temperature = dht.getTemperature();
  Blynk.virtualWrite(V1,temperature);
  Blynk.virtualWrite(V2,humidity);

lcd.clear();
lcd.setCursor(0,0);
lcd.print(" Temp   Humidity");
lcd.setCursor(0,1);
lcd.print(temperature);
lcd.print("    ");
lcd.print(humidity);
lcd.print("%");

delay(1500);
lcd.clear();
lcd.setCursor(0,0);
lcd.print(" The amount of  ");
lcd.setCursor(0,1);
lcd.print(" Air Qulity: ");
lcd.setCursor(12,1);
lcd.print(mq1355);
   lcd2.print(0,0," The amount of  ");
   lcd2.print(0,1," Air Qulity: ");
   lcd2.print(12,1, mq1355);
delay(2000);
```

```
if(smoke_sensor == 0){
  lcd1.print(0,0," Gas Detected");
lcd.clear();
lcd.setCursor(0,0);
lcd.print(" Gas Detected");
digitalWrite(relay,HIGH);
digitalWrite(buzzar,HIGH);
myservo.write(160);
delay(5000);
digitalWrite(relay,LOW);
digitalWrite(buzzar,LOW);
myservo.write(0);
delay(1000);
}
if(smoke_sensor == 1){
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Normal Condition");
lcd1.print(0,0,"Normal Condition");
}

delay(2000);
  }
```