

SMART WAREHOUSE MANAGEMENT SYSTEM USING IOT

Submitted in partial fulfilment of the requirements for the award of
Bachelor of Engineering Degree in Electronics and Communication Engineering

By

SASIDHAR REDDY.R (Reg. No.- 39130402)

MANOJ SAI.K (Reg. No. - 39130283)



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION
ENGINEERING**

SCHOOL OF ELECTRONICS

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119**

APRIL-2023



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION
ENGINEERING**

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **SASIDHAR REDDY.R(Reg. No.- 39130402)** and **MANOJ SAI.K (Reg. No. - 39130283)** who carried out the Project Phase-II entitled "**SMART WAREHOUSE MANAGEMENT USING IOT**" under my supervision from January 2023 to April 2023.

INTERNAL GUIDE

Ms. Mary Sajin Sanju I, M.E., (Ph.D.)

Head of Department

Dr. T. RAVI, M.E., Ph.D.

SUBMITTED FOR VIVA-VOCE EXAMINATION HELD ON 19-04-23

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION

We **SASIDHAR REDDY.R** and **MANOJ SAI.K** hereby declare that the project report entitled **Smart Warehouse Management using IOT** done by us under the guidance of **Ms. Mary Sajin Sanju I, M.E., (Ph.D.)** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Electronics and Communication Engineering.

DATE: 19-04-23

PLACE: Chennai

A handwritten signature in orange ink, appearing to be 'K. Manoj Sai', written over a yellow rectangular background.A handwritten signature in blue ink, appearing to be 'R. Sasidhar Reddy', written below the orange signature.

SIGNATURE OF CANDIDATE

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to the **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr. N. M. NANDHITHA, M.E., Ph.D. Dean, School of Electrical and Electronics Engineering** and **Dr. T. RAVI, M.E., Ph.D. Head of the Department of Electronics and Communication Engineering** for providing us necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to our Project Guide **Ms. Mary Sajin Sanju I, M.E., (Ph.D.)** Dept. of Electronics and Communication Engineering, for his valuable guidance, suggestions and constant encouragement for the successful completion of our project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Electronics and Communication Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

In general, warehouses are used to store goods or products. In the Warehouses, if the user wants to locate any product it is very difficult, because user must do a detailed search manually in all the available stockrooms this requires a lot of effort. So, to avoid this problem the warehouse inventory management system is very helpful because it maintains the detailed product information and tells us in which stockroom the product is present. The warehouse inventory management system is playing a significant aspect in many productions and goods-based methodology. Though there are many wireless communication technologies the RFID suits the best for the warehouse inventory management system. The tag information is transferred from the transmitter section to open-source hardware via a wireless link with the aid of the internet.

TABLE OF CONTENTS

Chapter No	TITLE	Page No.
	ABSTRACT	v
	LIST OF FIGURES	viii
	LIST OF TABLES	ix
	LIST OF ABBREVIATIONS	x
1	INTRODUCTION	1
	1.1 Overview	1
	1.2 Embedded System Software	5
	1.3 Implementation flow	7
2	LITERATURE SURVEY	8
	2.1 Inferences from Literature Survey	9
3	REQUIREMENTS AND SYSTEM ANALYSIS	10
	3.1 Drawbacks Of Existing System	11
	3.2 Proposed System	11
4	HARDWARE DESCRIPTION	12
	4.1 Arduino	12
	4.2 DHT11 SENSOR	21
	4.3 LM35 Sensor	22
	4.4 MQ2 sensor	25
	4.5 RFID reader and Tag	29
	4.6 GSM	32
	4.7 Node MCU	35
5	SOFTWARE DESCRIPTION	55

	5.1	Arduino IDE	55
6		CONCLUSION	64
7		REFERENCE	65
		APPENDIX	
	A.	SOURCE CODE	66
	B.	SCREENSHOTS	71
	C.	RESEARCH PAPER	73

LIST OF FIGURES

Figure No.	Figure Name	Page No.
1.1	Overview of embedded system	1
1.2	Block diagram of embedded system	2
1.3	Flow of burning source code to processor	5
1.4	Block Architecture	11
1.5	Arduino	12
1.6	Arduino Board and Microcontroller	13
1.8	Atmega	15
1.9	Arduino Pinout	16
2.0	DHT11	20
2.1	LM35	22
2.5	MQ2 Sensor	25
2.8	RFID Tags	30
2.9	GSM Modem	34
3.1	Node MCU	36
3.2	LCD	42
3.3	switch	50
3.4	Power Supply	51

3.5	Transformer	52
3.6	Rectifier	53
3.7	Capacitor	53
3.8	Voltage Regulator	54

LIST OF TABLES

Table No.	Table Name	Page No.
4.1	Arduino	21
4.2	Power supply	21
4.3	LCD	43
4.4	Command codes for LCD	45
4.5	Control and Display Commands	48

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Nowadays, the warehouse acts as a competitive factor in any supply chain as it has a main role in linking all the partners in it. Hence, it has become very necessary to allocate its resources efficiently and manage it effectively. A sound warehouse management system can contribute to cost reduction and improve customer satisfaction. However, traditional warehouse management system models have become unsuitable and inefficient for today's market requirements. So, companies have started to adopt innovative approaches and technologies to be used for such applications; one of these technologies is the Internet of Things that enables the connection between several physical objects and produces a massive amount of real-time data that can be transferred to useful information that helps in managing and decision making. In this paper, the architecture of this application is illustrated, its potential benefits are overviewed, and a framework for implementing this technology in a warehouse is proposed.

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 control systems, etc.

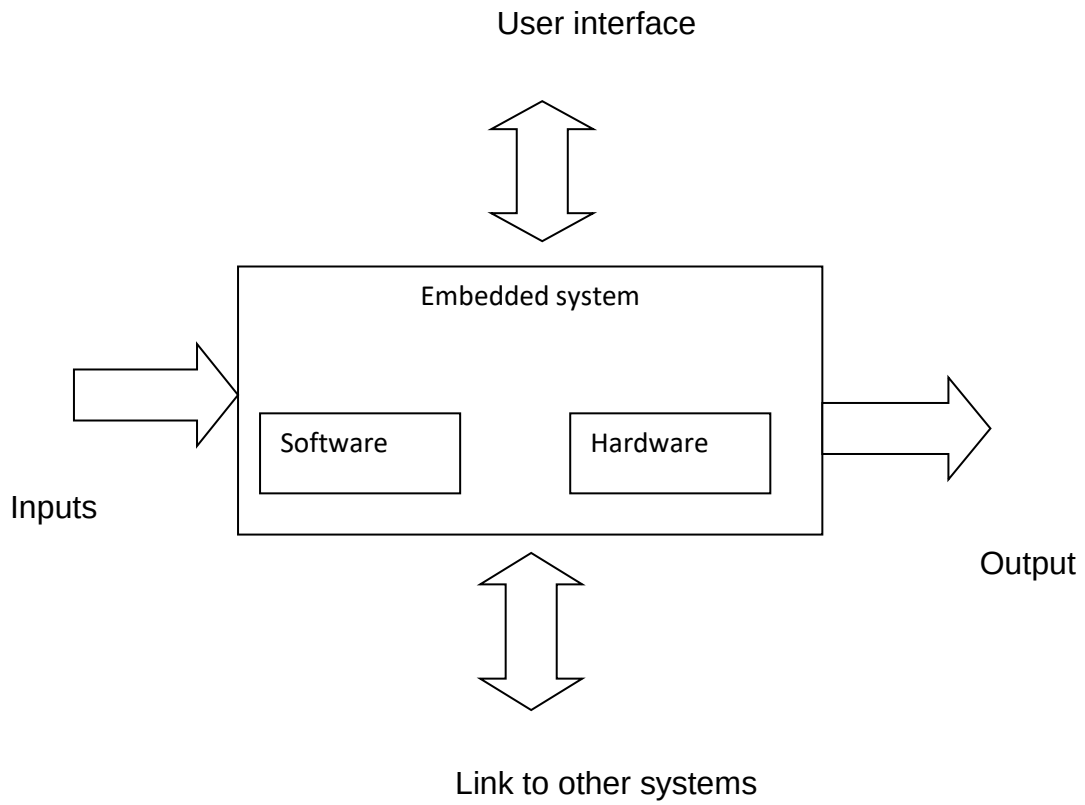


Fig 1.1 Overview of embedded system

Embedded system

Embedded system includes mainly two sections, they are

1. Hardware
2. Software

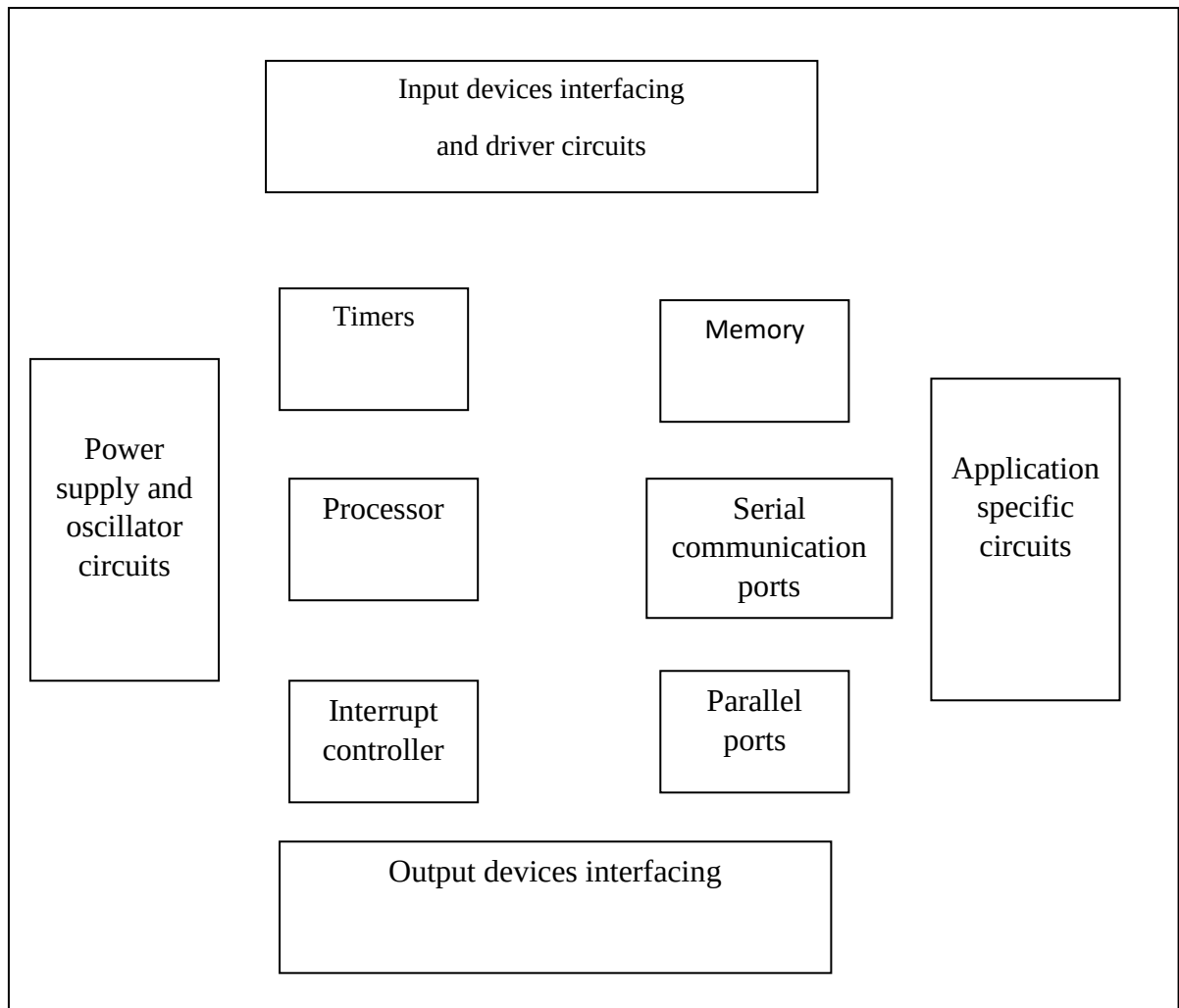


Fig 1.2 Block diagram of embedded system

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

1.2 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.

Generally we write source codes for embedded systems in assembly language, but the processors run only executable files. The process of converting the source code representation of your embedded software into an executable binary image involves three distinct steps:

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.

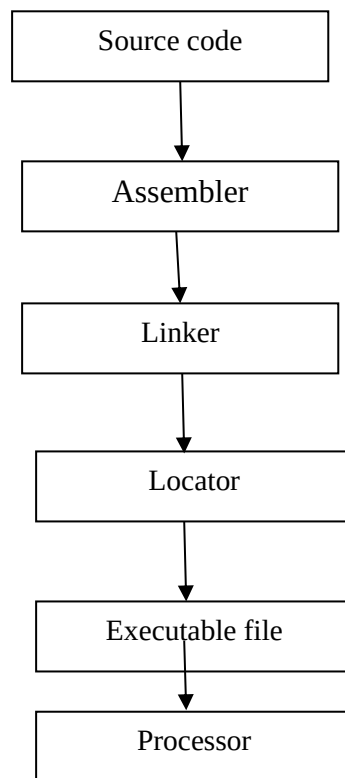


Fig 1.3 Flow of burning source code to processor

Applications

Embedded systems have different applications. A few select applications of embedded systems are smart cards, telecommunications, satellites, missiles, digital consumer electronics, computer networking, etc.

Embedded Systems in Automobiles

- Motor Control System
- Engine or Body Safety

- Robotics in Assembly Line
- Mobile and E-Com Access

Embedded systems in Telecommunications

- Mobile computing
- Networking
- Wireless Communications

Embedded Systems in Smart Cards

- Banking
- Telephone
- Security Systems

1.3 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

1. An Intelligent Warehouse Management System Using the Internet of Things **Walaa hamdy**

Nowadays, the warehouse acts as a competitive factor in any supply chain as it has a main role in linking all the partners in it. Hence, it has become very necessary to allocate its resources efficiently and manage it effectively. A sound warehouse management system can contribute to cost reduction and improve customer satisfaction. However, traditional warehouse management system models have become unsuitable and inefficient for today's market requirements. So, companies have started to adopt innovative approaches and technologies to be used for such applications; one of these technologies is the Internet of Things that enables the connection between several physical objects and produces a massive amount of real-time data that can be transferred to useful information that helps in managing and decision making. In this paper, the architecture of this application is illustrated, its potential benefits are overviewed, and a framework for implementing this technology in a warehouse is proposed.

2. STUDY OF SMART INVENTORY MANAGEMENT SYSTEM BASED ON THE INTERNET OF THINGS (IOT) Souvik Paul*, Atrayee Chatterjee, Digbijay Guha

In developing enterprises and the constant demands of the product diversity, traditional Inventory Management model can't achieve that, due to its heavy workload and low efficiency. This paper presents a new type of intelligent Inventory Management System based on the IoT and explains the principles and structure of it. This system has great advantages compared to the traditional mode, and we expect good prospects for its development. Inventory Management is a key area for

customer service and cost optimization in any manufacturing setup. As companies turn global and have thousands of components and hundreds of warehouses the inventory becomes a nightmare, and a lot of time is spend in tracking inventory and ensuring right shipments. Traditional systems of robotic arms for inventory pick and drop have been based on premises of marking areas of the warehouse and tracking it.

3.Study of warehouse management system using IOT

With the development of enterprises and the constant demands of product diversity, traditional warehouse management model cannot meet that, due to its heavy workload and low efficiency. This paper presents a new type of intelligent warehouse management system - Smart Warehouse Management System based on the IOT and expounded the principles and structure of it. This system has great advantages compared with the traditional mode, and we expect good prospects for its development. Given that smart warehouses have emerged in recent years, the earlier published reviews discussed less about current warehouse development and smart technologies. But the earlier reviews can still bring insights into the basic concepts related to warehouse operations management. In Rouwenhorst et al. (2000)'s review, warehouse design problems and models are presented at strategic, tactical, and operational levels. Gu et al. (2007) present a review of warehouse operation planning based on the classification according to the basic warehouse functions, including receiving, storage, order picking, and shipping. Gu et al. (2010) provide some comments on warehouse design based on the following warehouse design problems: Overall structure, sizing and dimensioning, equipment selection, operation strategy selection, and department layout. de Koster et al. (2007) stress the importance of the order picking process. Optimization problems related to layout design, storage assignment, routing, order batching, and zoning are reviewed.

Automated Storage and Retrieval System (AS/RS) has attracted much attention from earlier reviews. Roodbergen and Vis (2009) provide an overview of the AS/RS literature for a range of issues, including physical design, storage assignment,

batching, dwell-point location sequencing, and performance evaluation. Various models for AS/RS performance evaluation are surveyed in Gagliardi et al. (2012)'s work. Analytical models and simulations are the two types of models commonly used in research.

Recent reviews focus more on the emerging warehouse technologies and new developing trends. IoT, Industry 4.0, and other emerging technologies influence the way of information exchange in warehouses. Ben-Daya et al. (2017) reveal the role of the IoT and its impact on supply chain management. Radio frequency identification (RFID) tags are the major IoT technology studied in previous warehousing research. Manavalan and Jayakrishna (2019) review IoT's role in achieving sustainable goals in supply chain management. The importance of digitalization and the influence of IoT in the overall supply chain management are analyzed. The criteria for achieving business readiness for Industry 4.0 transformation are concluded. Winkelhaus and Grosse (2020) propose a systematic literature review for Logistics 4.0. The application of IoT, CPS, and other technologies related to Logistics 4.0 are summarized in the review.

The rapid development of automation technology has dramatically improved the automation level of warehouse systems. In this development process, different concepts related to automated warehouses have emerged. Boysen and Stephan (2016) classify single crane scheduling problems in AS/RS in three perspectives: Layout, order characteristics, and objectives, and then survey crane scheduling problems. Jaghbeer et al. (2020) present an analysis of the literature related to automated picking systems.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 EXISTING METHOD

Previously they needed to search the stored goods manually and they cannot monitor the each good at a time by manually. So that the goods may damage. To monitor the goods and find the good we are implementing the proposed method by using RFID technology.

Drawbacks of existing method:

- Takes more time to find the goods.
- Need a specific person to monitor the goods

3.2 PROPOSED METHOD

In the proposed method we are using RFID reader and tags which is connected to Arduino to find the Products or goods. To monitor the weather and status of goods we are using Temperature sensor and gas sensor.

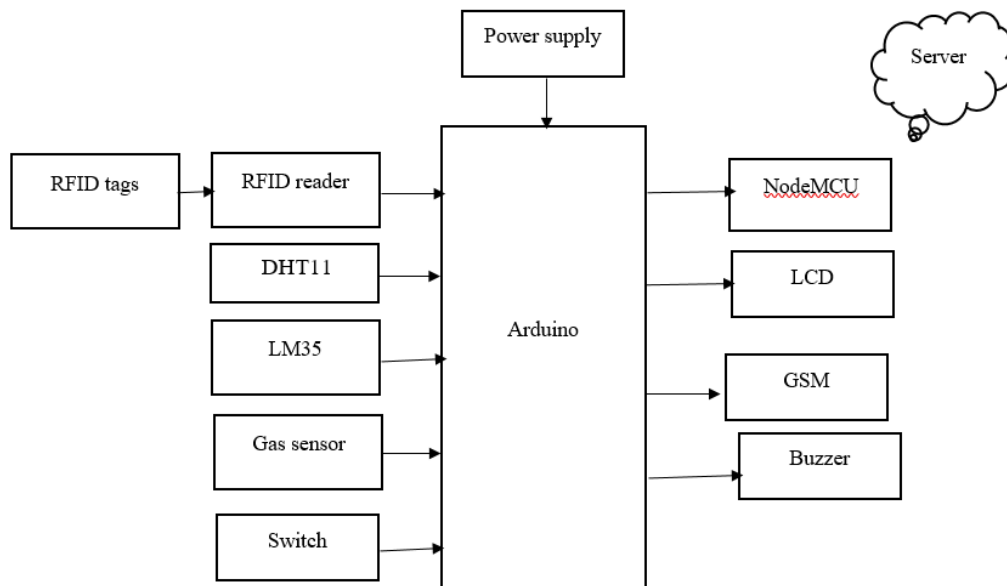


Fig 1.4 Block Architecture

CHAPTER 4

HARDWARE DESCRIPTION

4.1 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.

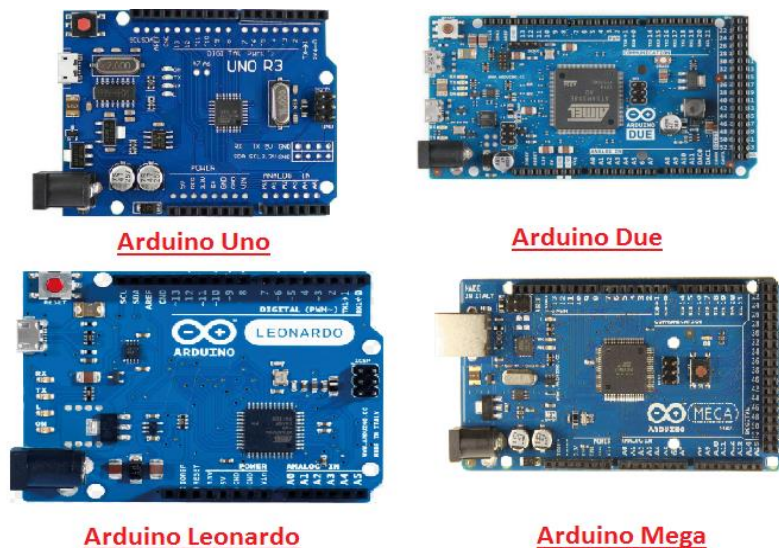


Fig 1.5 Arduino

It is an open-source platform, means the boards and software are readily available and anyone can modify and optimize the boards for better functionality.

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.

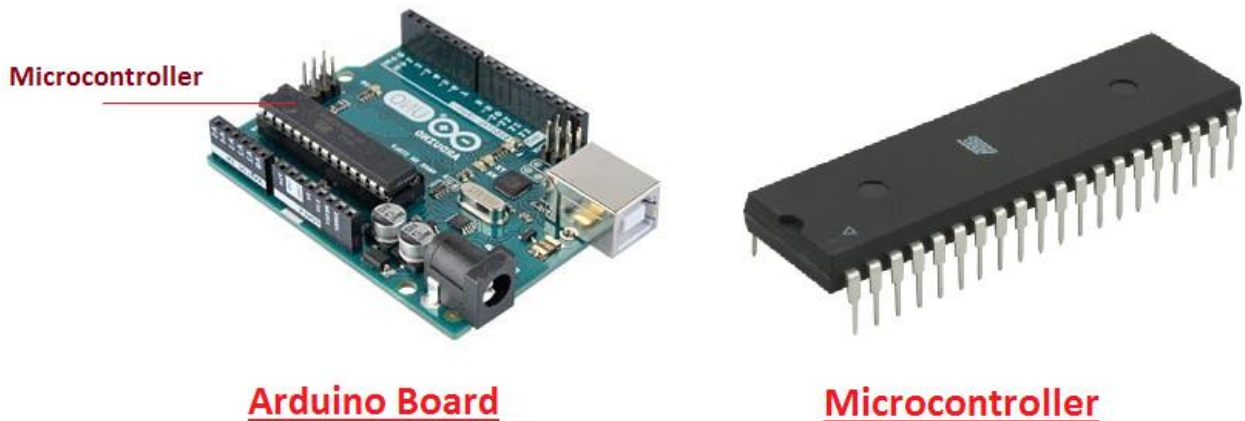


Fig 1.6 Arduino Board and Microcontroller

While learning microcontroller requires some expertise and skills.

Nevertheless, we can say every Arduino is basically a microcontroller but not every microcontroller is an Arduino.

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. Out of 14 I/O ports, 6 pins can be used for PWM output.
- It allows the designers to control and sense the external electronic devices in the real world

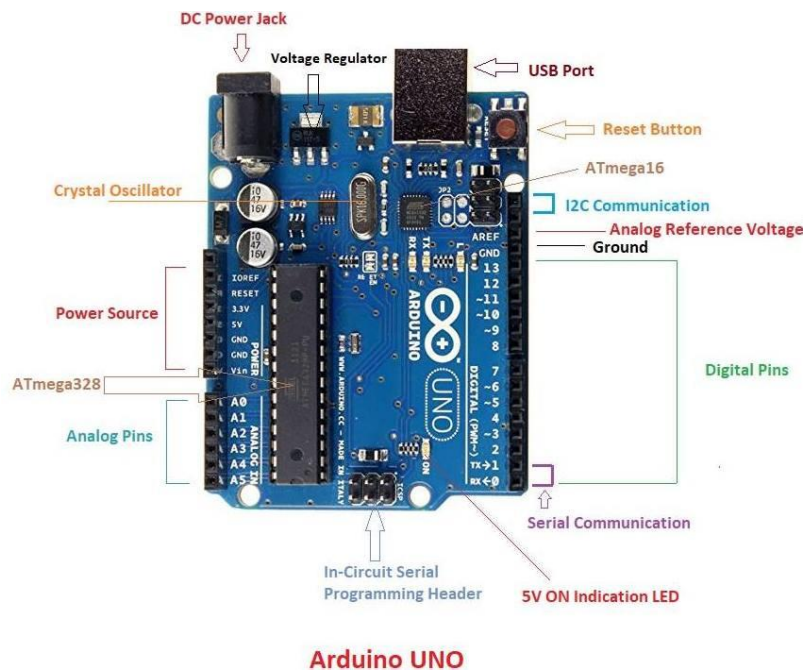


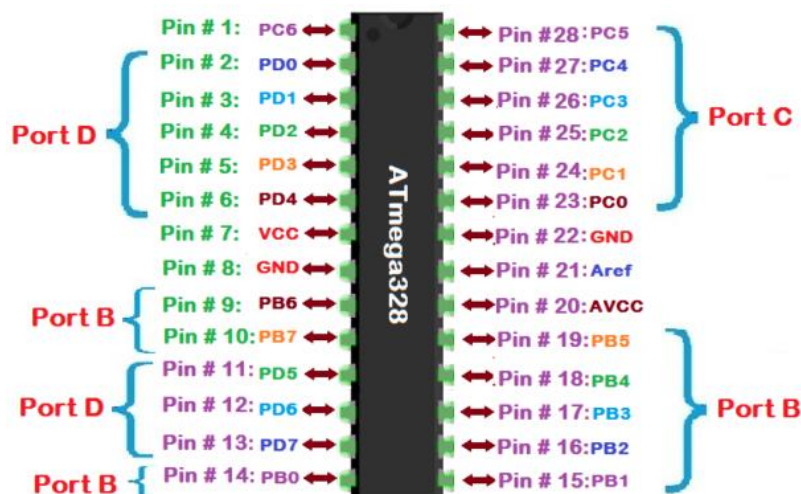
Fig 1.7 Arduino UNO

- 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.
- When nature and functionality of the task go complex, Micro SD card can be added in the boards to make them store more information.

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 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.



Atmega328 Microcontroller

Fig 1.8 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.
- This board comes with a built-in regulation feature which keeps the voltage under control when the device is connected to the external device.

- Reset pin is added in the board that reset the whole board and takes the running program in the initial stage. This pin is useful when board hangs up in the middle of the running program; pushing this pin will clear everything up in the program and starts the program right from the beginning.
- 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. There is no hard and fast interface required to connect the devices to the board. Simply plug the external device into the pins of the board that are laid out on the board in the form of the header.
- 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. ON-chip ADC is used to sample these pins. A 16 MHz frequency crystal oscillator is equipped on the board. Following figure shows the pinout of the Arduino Uno Board

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 through programming.

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

Table 4.1 ARDUINO

Microcontroller	<u>ATmega328P</u> – 8-bit AVR family microcontroller
Operating Voltage	5V
Recommended Input Voltage	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

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.

Arduino Uno is programmed using Arduino Software which a cross-platform application called IDE is written in Java. The AVR microcontroller Atmega328 laid out on the base comes with built-in boot loader that sets you free from using a separate burner to upload the program on the board.

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
- Security and Defense System
- Digital Electronics and Robotics
- Parking Lot Counter
- Weighing Machines
- Traffic Light Count Down Timer
- Medical Instrument
- Emergency Light for Railways
- Home Automation
- Industrial Automation

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.

4.2 DHT11 SENSOR (TEMPERATURE/HUMIDITY)

The DHT11 is a basic, 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. The only real downside of this sensor is you can only get new data from it once every 2 seconds.

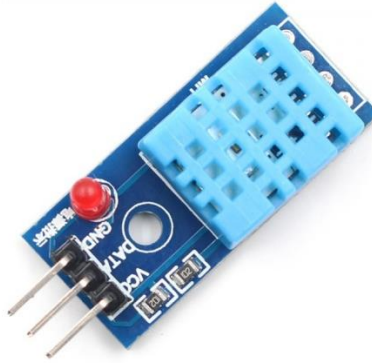


Fig 2.0 DHT11

Technical Specifications:

Table 4.1 DHT11

Item	Measurement Range	Humidity Accuracy	Temperature Accuracy	Resolution	Package
DHT11	20-90%RH 0-50°C	±5%RH	±2°C	1	4 Pin Single Row

Typical Application

Note: 3Pin – Null; MCU = Micro-computer Unite or single chip Computer

When the connecting cable is shorter than 20 metres, a 5K pull-up resistor is recommended;

When the connecting cable is longer than 20 metres, choose an appropriate pull-up resistor as

Needed.

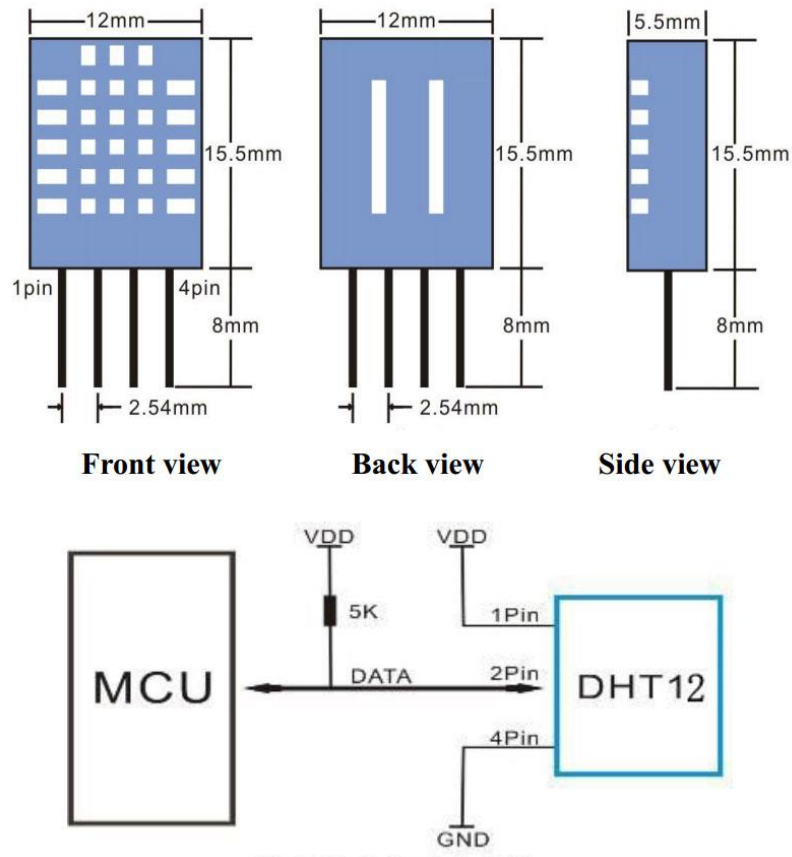


Fig 2.1 DHT11

Power and Pin

DHT11's power supply is 3-5.5V DC. When power is supplied to the sensor, do not send any instruction to the sensor in within one second in order to pass the unstable status. One Capacitor valued 100nF can be added between VDD and GND for power filtering.

4.3 LM35 Sensor

LM35 is a temperature sensor that outputs an analog signal which is proportional to the instantaneous temperature. The output voltage can easily be interpreted to obtain a temperature reading in Celsius. The advantage of LM35 over thermistor is it does not require any external calibration. The coating also protects it from self-heating. Many low-end products take advantage of low cost, greater accuracy and used LM35 in their products.

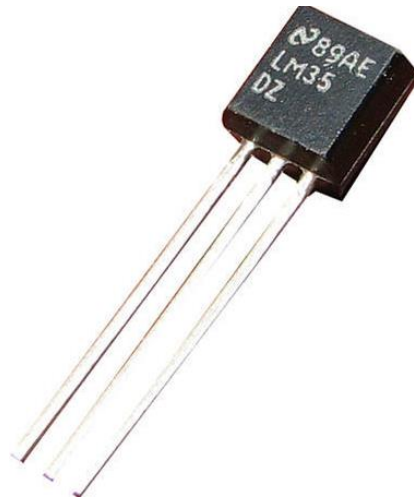


Fig 2.2 LM35 Sensor

Pin Description

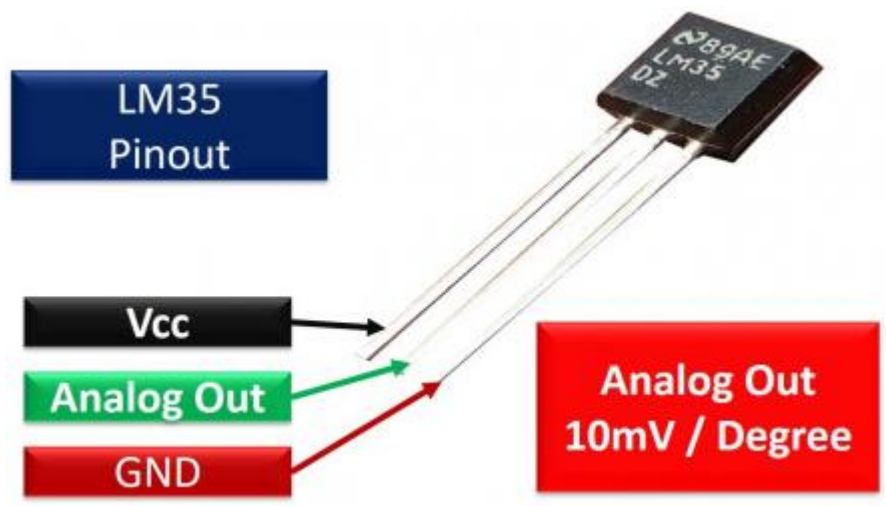


Fig 2.3 LM35

VCC: Supply Voltage (4V – 30V)

Out: It gives analog output voltage which is proportional to the temperature (in degree Celsius).

GND: Ground

LM35 Working Principle (Understanding LM35 Linear Scale Factor)

- Linear + 10-mV/°C Scale Factor

In order to understand the working principle of LM35 temperature sensor we have to understand the linear scale factor. In the features of LM35 it is given to be **+10 mills volt per degree centigrade**. It means that with increase in output of 10 mills volt by the sensor vout pin the temperature value increases by one. For example, if the sensor is outputting 100 mills volt at vout pin the temperature in centigrade will be 10-degree centigrade. The same goes for the negative temperature reading. If the sensor is outputting -100 mills volt the temperature will be -10 degrees Celsius.

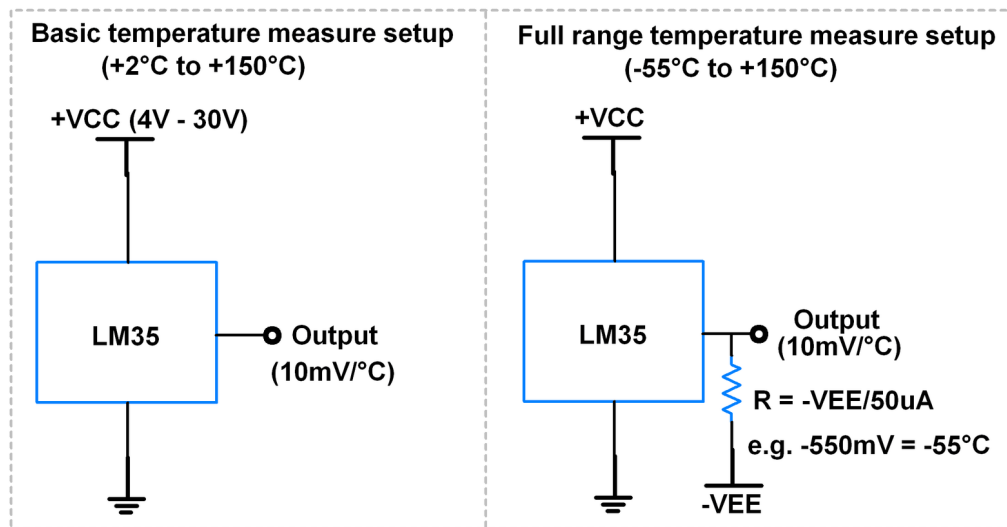


Fig 2.4 LM35

LM35 can be used in two circuit configurations. Both yield different results. In the first configuration, you can only measure the positive temperature from 2 degrees Celsius to 150 degrees Celsius. In this first configuration, we simply power Im35 and connect the output directly to analog to digital converters. In the second configuration, we can utilize all the sensor resources and can measure the full range temperature from -55 degree centigrade to 150-degree centigrade. This configuration is a little complex but yields high results. We have to connect an external resistor, in this case, to switch the level of negative voltage upwards. The external resistor value can be calculated from the formula given below the configuration circuit. The second configuration circuit can be made in various ways. The accuracy parameters for both configurations are different. The average accuracy level is ± 1 degree Celsius for both configurations. But the accuracy level decreases for temperature between 2 degrees to 25-degree centigrade.

4.4 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.

Gas sensors are available in wide specifications depending on the sensitivity levels, type of gas to be sensed, physical dimensions and numerous other factors. This Insight covers a methane gas sensor that can sense gases such as ammonia which might get produced from methane. When a gas interacts with this sensor, it is first ionized into its constituents and is then adsorbed by the sensing element. This adsorption creates a potential difference on the element which is conveyed to the processor unit through output pins in form of current.



Fig 2.5 MQ2 Sensor

The **gas sensor module** consists of a steel exoskeleton under which a sensing element is housed. This sensing element is subjected to current through connecting leads. 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. This changes the resistance of the sensing element which alters the value of the current going out of it.

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.

3. Exhibits an anti-explosion network that keeps the sensor module intact at high temperatures and gas pressures.

In order to manage above-listed functions efficiently, the steel mesh is made into two layers. The mesh is bound to rest of the body via a copper plated clamping ring. 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. They are casted from copper and have tin plating over them. 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.

The pins are placed on a Bakelite base which is a good insulator and provides firm gripping to the connecting leads of the sensor.

Internal Features

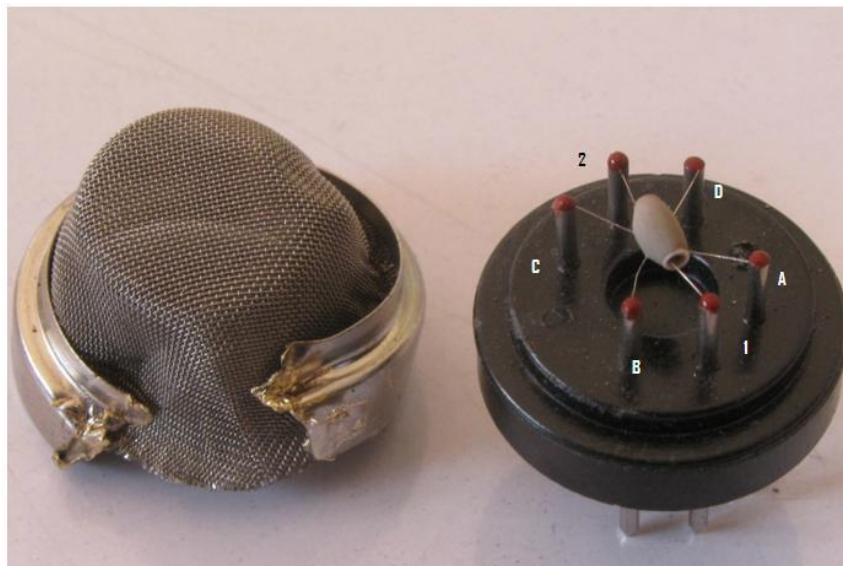


Fig. 2.6 Inside View of Gas Sensor after Removal of Steel Mash

The top of the gas sensor is removed off to see the internal 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. Image 4 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 leads responsible for heating the sensing element are connected through Nickel-Chromium, well known conductive alloy. Leads responsible for output signals are connected using platinum wires which convey small changes in the current that passes through the sensing element. 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 2.7 Ceramic Sensing Element Present Inside a Gas Sensor

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.

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. Through Nickel-chromium wire, the ceramic region of the sensing element is subjected to heating current. The heat is radiated by the element

in the nearby region where gases interact with it and get ionized. Once, ionized, they are absorbed by the tin dioxide. 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.

4.5 RFID READER AND TAG

RFID represents ID of radio frequencies. RFID labels are little chips (more often than not as a keen card or a meeting card) that are utilized in our day by day lives to open lodgings, enter autos, and so on. These little chips structure the RFID framework together with a RFID reader.

During the Second World War, RFID technology was first used to identify enemy aircraft. RFID technology has evolved since then and is now being used in many different industries. A clean example is a smart warehouse where RFID technology is used to automate the warehousing process.

Two parts of an RFID system are 1) RFID Reader and 2) RFID Tag. Data is stored electronically in the RFID tag. The reader collects this data using electromagnetic waves. Tags can only store a few kilograms of data bytes.

What is an RFID Reader?



Fig 2.8 RFID Tags

RFID module

RFID module The RFID reader has a radio transmitter and recipient inside. It is likewise called as an investigative specialist. The reader transmits radio recurrence flags persistently after controlling. At the point when a RFID tag is set inside the range region of a reader, it invigorates the tag through electromagnetic enlistment and gathers the data from it.

RFID TAG

RFID labels are accessible in various sorts of size and shapes. The Tag contains an IC for putting away the information, a reception apparatus for transmitting and accepting, and furthermore a modulator. Tags are very small in size, and they can hold only few bits of data.

Working of RFID

RFID has a place with the Automatic Identification and Data Capture (AIDC) innovation gathering. AIDC strategies consequently distinguish objects, gather information on them, and straightforwardly enter this information into PC frameworks with next to zero human mediation. To accomplish this, RFID techniques utilize radio waves.

RFID frameworks are comprised of three segments at a straightforward dimension: a RFID tag or brilliant mark, a RFID reader, and a radio wire. RFID labels contain a coordinated circuit and receiving wire for transmitting information to the RFID reader (otherwise called the examiner). The reader then changes the radio waves into an information structure that is progressively usable. Data assembled from the labels is then exchanged to a host PC framework by means of a correspondence interface, where information can be put away in a database and broke down at a later date.

GSM

GSM is a mobile communication modem; it stands for global system for mobile communication (GSM). The idea of GSM was developed at Bell Laboratories in 1970. It is widely used mobile communication system in the world. GSM is an open and digital cellular technology used for transmitting mobile voice and data services operates at the 850MHz, 900MHz, 1800MHz and 1900MHz frequency bands.

GSM system was developed as a digital system using time division multiple access (TDMA) technique for communication purpose. A GSM digitizes and reduces the data, then sends it down through a channel with two different streams of client data, each in its own particular time slot. The digital system has an ability to carry 64 kbps to 120 Mbps of data rates.

There are various cell sizes in a GSM system such as macro, micro, pico and umbrella cells. Each cell varies as per the implementation domain. There are five different cell sizes in a GSM network macro, micro, pico and umbrella cells. The coverage area of each cell varies according to the implementation environment.

Time Division Multiple Access

TDMA technique relies on assigning different time slots to each user on the same frequency. It can easily adapt to data transmission and voice communication and can carry 64kbps to 120Mbps of data rate.

4.6 GSM ARCHITECTURE

A GSM network consists of the following components:

- **A Mobile Station:** It is the mobile phone which consists of the transceiver, the display and the processor and is controlled by a SIM card operating over the network.
- **Base Station Subsystem:** It acts as an interface between the mobile station and the network subsystem. It consists of the Base Transceiver Station which contains the radio transceivers and handles the protocols for communication with mobiles. It also consists of the Base Station Controller which controls the Base Transceiver station and acts as a interface between the mobile station and mobile switching centre.
- **Network Subsystem:** It provides the basic network connection to the mobile stations. The basic part of the Network Subsystem is the Mobile Service Switching Centre which provides access to different networks like ISDN, PSTN etc. It also consists of the Home Location Register and the Visitor Location Register which provides the call routing and roaming capabilities of GSM. It also contains the Equipment Identity Register which maintains an

account of all the mobile equipments wherein each mobile is identified by its own IMEI number. IMEI stands for International Mobile Equipment Identity.

Features of GSM Module

- Improved spectrum efficiency
- International roaming
- Compatibility with integrated services digital network (ISDN)
- Support for new services.
- SIM phonebook management
- Fixed dialing number (FDN)
- Real time clock with alarm management
- High-quality speech
- Uses encryption to make phone calls more secure
- Short message service (SMS)

The security strategies standardized for the GSM system make it the most secure telecommunications standard currently accessible. Although the confidentiality of a call and secrecy of the GSM subscriber is just ensured on the radio channel, this is a major step in achieving end-to- end security.

GSM Modem

A GSM modem is a device which can be either a mobile phone or a modem device which can be used to make a computer or any other processor communicate over a network. A GSM modem requires a SIM card to be operated and operates over a network range subscribed by the network operator. It can be connected to a computer through serial, USB or Bluetooth connection.

A GSM modem can also be a standard GSM mobile phone with the appropriate cable and software driver to connect to a serial port or USB port on your computer. GSM modem is usually preferable to a GSM mobile phone. The GSM modem has wide range of applications in transaction terminals, supply chain management, security applications, weather stations and GPRS mode remote data logging.

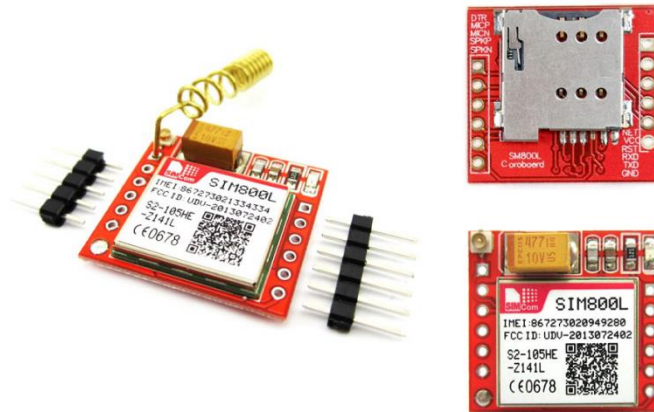


Fig 2.9 GSM Modem

It requires a SIM (Subscriber Identity Module) card just like mobile phones to activate communication with the network. Also, they have IMEI (International Mobile Equipment Identity) number similar to mobile phones for their identification. A GSM/GPRS MODEM can perform the following operations:

1. Receive, send or delete SMS messages in a SIM.
2. Read, add, search phonebook entries of the SIM.
3. Make, Receive, or reject a voice call.

The MODEM needs AT commands, for interacting with processor or controller, which are communicated through serial communication. These commands are sent by the controller/processor. The MODEM sends back a result after it receives a command. Different AT commands supported by the MODEM can be sent by the processor/controller/computer to interact with the GSM and GPRS cellular network.

GSM Architecture

The GSM architecture is divided into Radio Subsystem, Network and Switching Subsystem and the Operation Subsystem. The radio sub system consists of the Mobile Station and Base Station Subsystem.

The mobile station is generally the mobile phone which consists of a transceiver, display and a processor. Each handheld or portable mobile station consists of a unique identity stored in a module known as SIM (Subscriber Identity Chip). It is a

small microchip which is inserted in the mobile phone and contains the database regarding the mobile station.

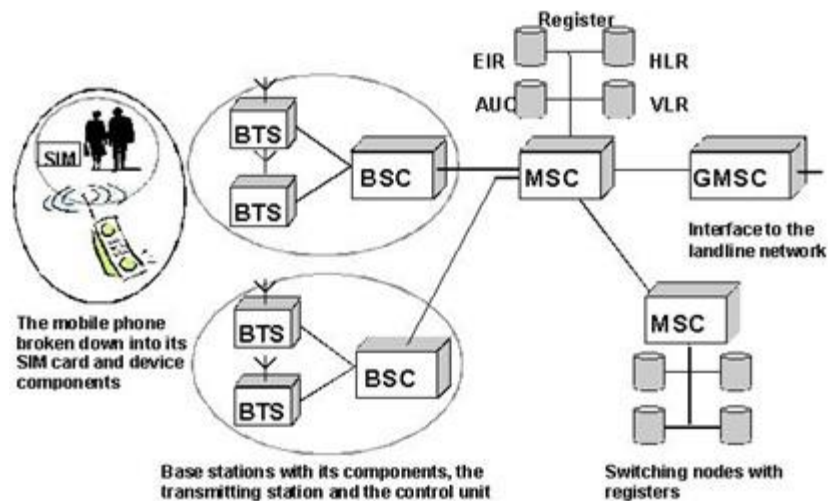


Fig 3.0 GSM

4.7 NODEMCU

Introduction to Node MCU

Node MCU is an open-source firmware and development kit that plays a vital role in designing your own IoT product using a few Lua script lines.

Multiple GPIO pins on the board allow you to connect the board with other peripherals and are capable of generating PWM, I2C, SPI, and UART serial communications.

The interface of the module is mainly divided into two parts including both Firmware and Hardware where former runs on the ESP8266 Wi-Fi SoC and later is based on the ESP-12 module.

The firmware is based on Lua – A scripting language that is easy to learn, giving a simple programming environment layered with a fast scripting language that connects you with a well-known developer community.

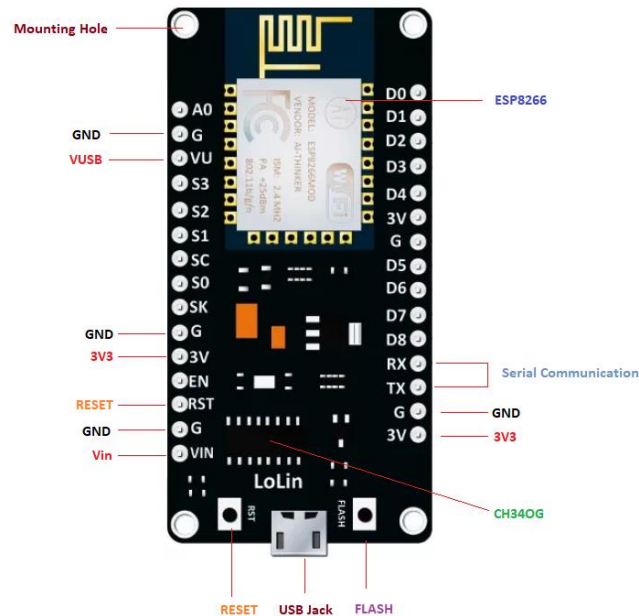


Fig 3.1 Node MCU

And open-source firmware gives you the flexibility to edit, modify and rebuilt the existing module and keep changing the entire interface until you succeed in optimizing the module as per your requirements.

USB to UART converter is added on the module that helps in converting USB data to UART data which mainly understands the language of serial communication. Instead of the regular USB port, MicroUSB port is included in the module that connects it with the computer for dual purposes: programming and powering up the board.

The board incorporates status LED that blinks and turns off immediately, giving you the current status of the module if it is running properly when connected with the computer.

The ability of module to establish a flawless WiFi connection between two channels makes it an ideal choice for incorporating it with other embedded devices like Raspberry Pi.

NodeMCU Pinout

NodeMCU comes with a number of GPIO Pins. Following figure shows the Pinout of the board.

There is a candid difference between VIN and VU where former is the regulated voltage that may stand somewhere between 7 to 12 V while later is the power voltage for USB that must be kept around 5 V.

Features

- Open-source
- Arduino-like hardware
- Status LED
- MicroUSB port
- Reset/Flash buttons
- Interactive and Programmable
- Low cost
- ESP8266 with inbuilt wifi
- USB to UART converter
- GPIO pins

As mentioned above, a cable supporting micro USB port is used to connect the board. As you connect the board with a computer, LED will flash. You may need some drivers to be installed on your computer if it fails to detect the NodeMCU board. You can download the driver from [this](#) page.

Note: We use [Arduino IDE](#) software for programming this module. It is important to note that the pin configuration appearing on the board is different from the

configuration we use to program the board on the software i.e. when we write code for targeting pin 16 on the Arduino IDE, it will actually help in laying out the communication with the D0 pin on the module.

Powering NodeMCU

You can see from the pinout image above, there are five ground pins and three 3V3 pins on the board. The board can be powered up using the following three ways.

USB Power. It proves to be an ideal choice for loading programs unless the project you aim to design requires separate interface i.e. disconnected from the computer.

Provide 3.3V. This is another great option to power up the module. If you have your own off-board regulator, you can generate an instant power source for your development kit.

Power Vin. This is a voltage regulator that comes with the ability to support up to 800 mA. It can handle somewhere between 7 to 12 V. You cannot power the devices operating at 3.3 V, as this regulator is unable to generate as low as 3.3V.

Applications

NodeMCU V3 is mainly used in the WiFi Applications which most of the other embedded modules fail to process unless incorporated with some external WiFi protocol. Following are some major applications used for NodeMCU V3.

- Internet Smoked Alarm
- VR Tracker
- Octopod
- Serial Port Monitor
- ESP Lamp
- Incubator Controller

- IoT home automation
- Security Alarms

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 LCD is either made with a uninvolved lattice or a showcase network for dynamic framework show. Likewise alluded to as a meager film transistor (TFT) show is the dynamic framework LCD. The uninvolved LCD lattice has a matrix of conductors at every crossing point of the network with pixels. Two conductors on the lattice send a current to control the light for any pixel. A functioning framework has a transistor situated at every pixel crossing point, requiring less current to control the luminance of a pixel.

Some aloof network LCD's have double filtering, which implies they examine the matrix twice with current in the meantime as the first innovation took one sweep. Dynamic lattice, be that as it may, is as yet a higher innovation.

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.

The directions given to the LCD are put away by the order register. An order is a direction given to LCD to play out a predefined assignment, for example, introducing it, clearing its screen, setting the situation of the cursor, controlling presentation, and so forth. 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.

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.

It takes 39-43 μ S for the LCD display to place a character or execute a command. It takes 1.53ms to 1.64ms except for clearing display and searching for cursor to the home position.

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.

Users can easily display their custom characters on the LCD screen after CGRAM has been set up to display characters.

Pin Diagram:

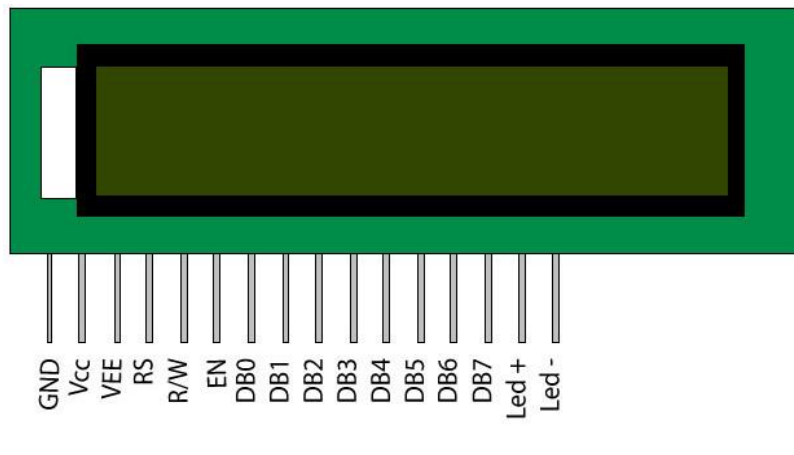


Fig 3.2 LCD

Table 4.3 LCD

Pin No	Function	Name
1	Ground (0V)	Ground
2	Supply voltage; 5V (4.7V – 5.3V)	V _{cc}
3	Contrast adjustment; through a variable resistor	V _{EE}
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 V _{cc} (5V)	Led+
16	Backlight Ground (0V)	Led-

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, clearing its screen, setting the situation of the cursor, controlling showcase, and so on. Order preparing happens in the direction register.

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.

LCD Commands

There are some preset commands in the LCD that we need to send to the LCD via some microcontroller. The following are some important command instructions:

Table 4.4 Command codes for LCD

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)
18	38	2 lines and 5×7 matrix

Block Diagram of LCD Display: -

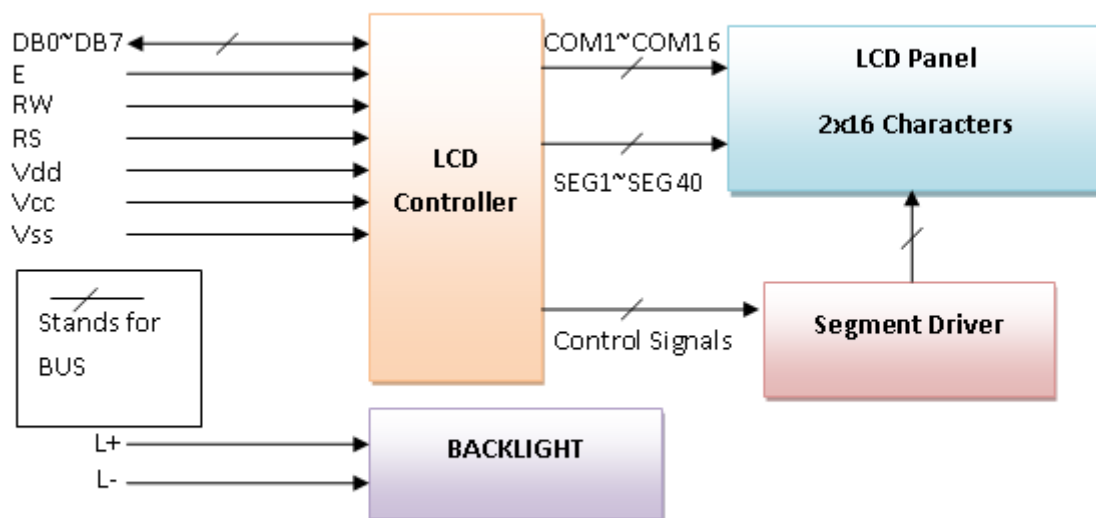


Fig 4.4 LCD

Control and display commands

Table 4.5 Control and Display Commands

Instruc tion	Instruction Code										Instruction Code Description	Execu tion 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	D 7	D 6	D 5	D 4	D 3	D 2	D 1	D 0	Read data from internal RAM	1.53- 1.64m s
Write data to RAM	1	0	D 7	D 6	D 5	D 4	D 3	D 2	D 1	D 0	Write data into internal RAM (DDRAM/CG RAM)	1.53- 1.64m s
Busy flag & Addres s	0	1	B F	A C 6	A C 5	A C 4	A C 3	A C 2	A C 1	A C 0	Busy flag (BF: 1→ LCD Busy) and contents of address	39 μs

											counter in bits AC6-AC0.	
Set DDRAM Address s	0	0	1	A C 6	A C 5	A C 4	A C 3	A C 2	A C 1	A C 0	Set DDRAM address in address counter.	39 μ s
Set CGRAM Address s	0	0	0	1	A C 5	A C 4	A C 3	A C 2	A C 1	A C 0	Set CGRAM Address in address counter.	39 μ s
Function Set	0	0	0	0	1	D L	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 μ s
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 μ s
Display & Cursor On/Off	0	0	0	0	0	0	1	D	C	B	Set Display(D),C ursor(C) and cursor blink(b) on/off control	39 μ s

Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S/H	Assign cursor moving direction and enable shift entire display.	0 μ s
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.	43 μ s
Clear Display	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM and set DDRAM Address to "00H" from AC	43 μ s

4-bit and 8-bit Mode of LCD:

The LCD can work in two striking modes, the 4-bit mode and the 8-bit mode. We send the information snack through snack in 4 bit mode, first upper chomp, by then lower snack. For those of you who don't have the foggiest idea what a goody is: a chomp is a four-piece gathering, so a byte's lower four bits (D0-D3) are the lower snack, while a byte's upper four bits (D4-D7) are the higher snack. This enables us to send 8 bit data. This connects with us to send 8 bit data. Whereas in 8 bit mode we can send the 8-bit information truly in one stroke since we utilize all the 8 information lines. You need to get it now; yes 8-bit mode is quicker and immaculate than 4-bit mode. In any case, the fundamental shortcoming is that it needs 8 microcontroller-related information lines. This will result in our MCU coming up short on I/O pins, so 4-bit mode is extensively utilized. To set these modes, no control pins are used.

Switch

A Push Button switch is a type of switch which consists of a simple electric mechanism or air switch mechanism to turn something on or off.

Depending on model they could operate with momentary or latching action function.

The button itself is usually constructed of a strong durable material such as metal or plastic. Push Button Switches come in a range of shapes and sizes. We have a selection of push button switches here at Herga.

Push button switches are used throughout industrial and medical applications and are also recognisable in everyday life.

For uses within the Industrial sector, push buttons are often part of a bigger system and are connected through a mechanical linkage. This means that when a button is pressed it can cause another button to release.

Push button switches are present in so many areas across different industries and for different uses here are some examples;

- Calculator buttons – a hand held calculator has lots of small push button switches
- Reset switches – these are usually small and require a tool to press to avoid accidental operation
- Stopping machinery – often around industrial machinery there will be an emergency stop button, sometimes these are located on the wall
- Arcade gaming – these are usually brightly coloured to encourage people to play

Features

Prevent flux rise by the insert-molded terminal

Snap-in mount terminal

Contact Bounce: MAX 5mS

Crisp clicking by tactile feedback

Dielectric Withstanding Voltage 250V AC for 1 minute

Technical Specifications

Mode of Operation: Tactile feedback

Power Rating: MAX 50mA 24V DC

Insulation Resistance: 100Mohm at 100v

Operating Force: 2.55 ± 0.69 N

Contact Resistance: MAX 100mOhm

Operating Temperature Range: -20 to +70 °C

Storage Temperature Range: -20 to +70 °C

Push-Buttons are normally-open **tactile switches**. Push buttons allow us to power the circuit or make any particular connection only when we press the button. Simply, it makes the circuit connected when pressed and breaks when released. A push button is also used for triggering of the SCR by gate terminal. These are the most common buttons which we see in our daily life electronic equipment's.

When connecting in between of supply and the circuit we should only connect the wires with both the legs of the Push-Button as shown in the circuit below:

A Push-Button can also be used for the triggering purpose like of SCR. An SCR is a gate controlled Switch which needs a triggering pulse. So, for this we can add a Push button in the circuit to give a triggering pulse, as shown in the circuit below:

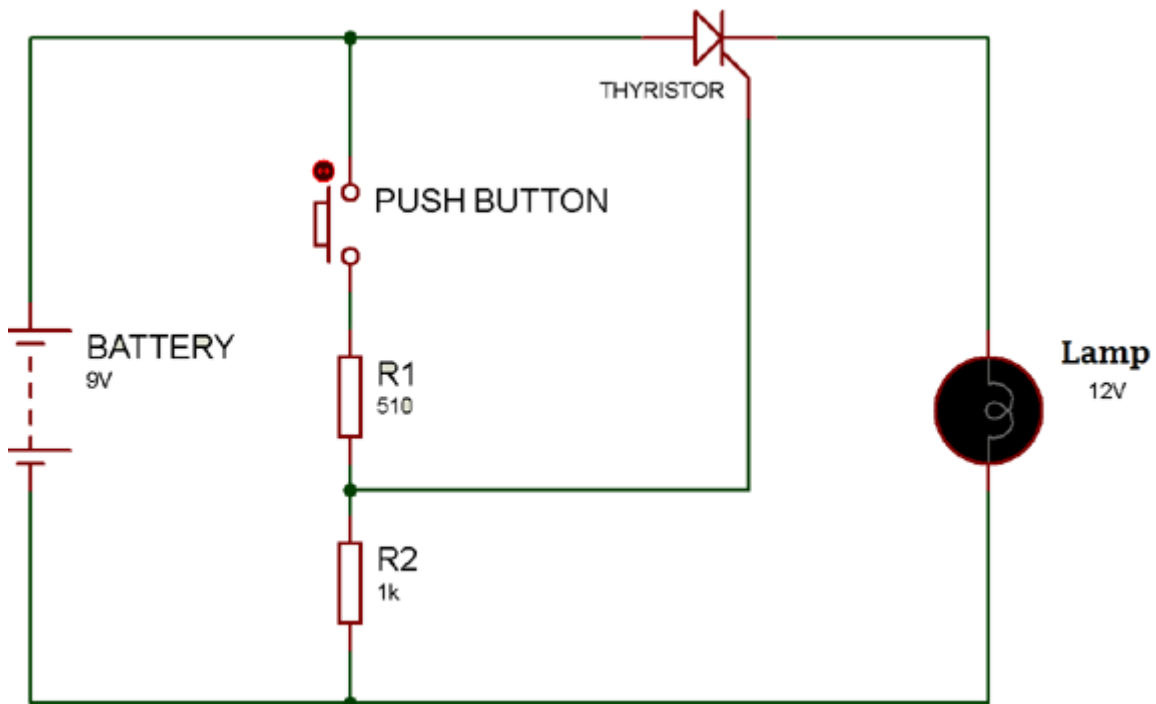


Fig 3.3 switch

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.

Most computer power supplies also have an input voltage switch that, depending on the geographic location, can be set to 110v/115v or 220v/240v. Due to the different power voltages supplied by power outlets in different countries, this switch position is crucial.

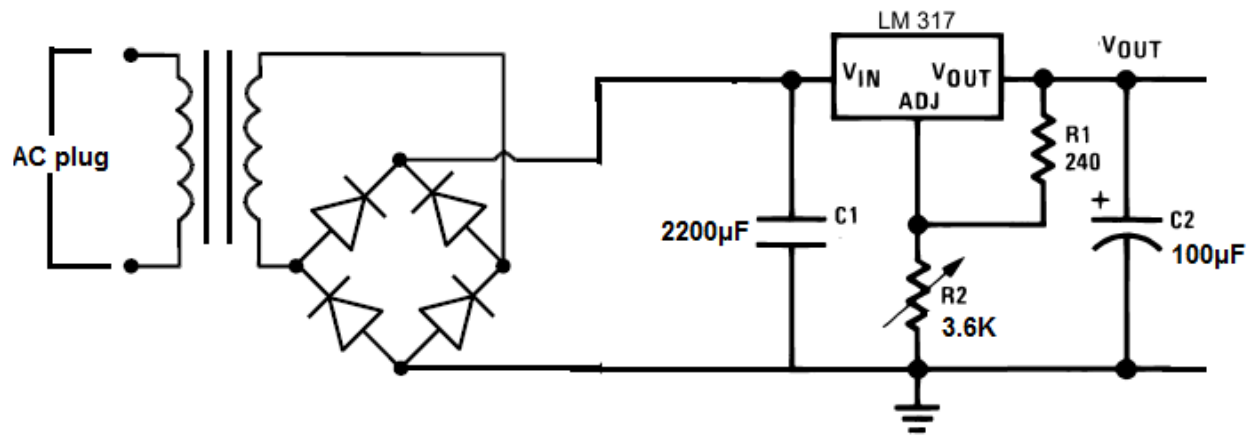


Fig 3.4 Power Supply

Some basic components used in the supply of power:

Transformer

A transformer is a static electrical gadget that exchanges control between at least two circuits. A fluctuating current creates a changing attractive motion in one transformer curl, which thus actuates a differing electromotive power over a second loop twisted around a similar center.

Without a metallic association between the two circuits, electrical vitality can be exchanged between the two loops. The enlistment law of Faraday found in 1831 portrayed the impact of prompted voltage in any curl because of the changing attractive flux surrounded by the coil.

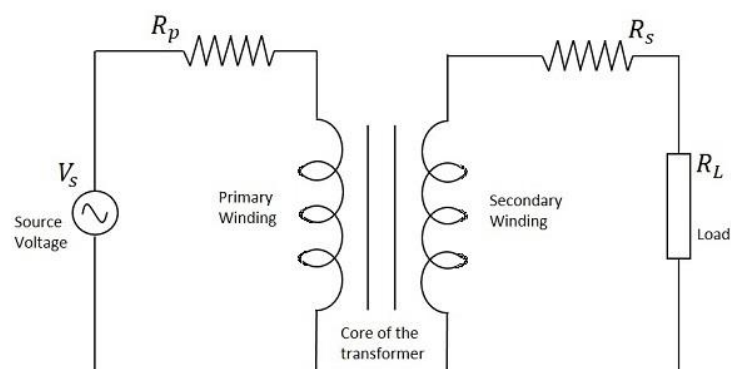


Fig 3.5 Circuit of transformer

Rectifier

A rectifier is an electrical device that converts alternating current (AC), which periodically reverses direction, to direct current (DC), which flows in only one direction. The process is known as *rectification*, since it "straightens" the direction of current.

Rectifiers have many uses, but are often found to serve as components of DC power supplies and direct power transmission systems with high voltage. Rectification can be used in roles other than direct current generation for use as a power source.

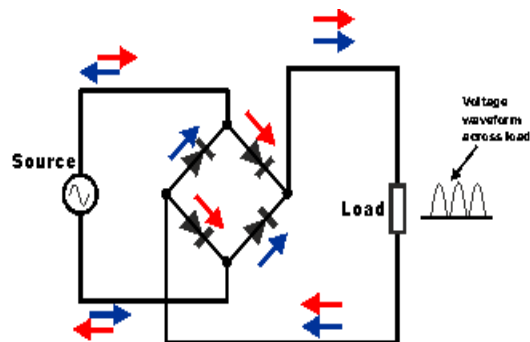


Fig 3.6 Circuit of rectifier

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 3.7 Capacitor

Voltage regulator

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.

7805 voltage regulator with pinout

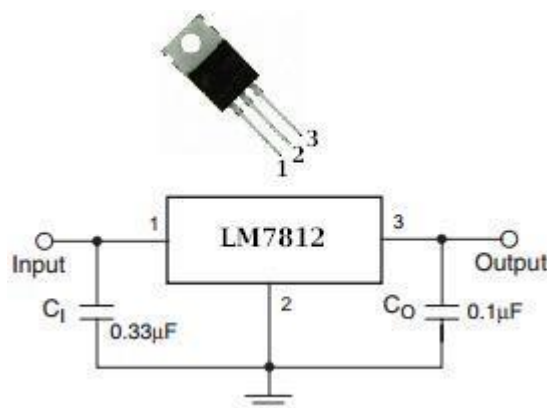


Fig 3.8 7812 voltage regulator with pinout

CHAPTER 5

SOFTWARE REQUIREMENTS

5.1 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 and many more.
- Each of them contains a microcontroller on the board that is actually programmed and accepts the information in the form of code.
- The main code, also known as a sketch, created on the IDE platform will ultimately generate a Hex File which is then transferred and uploaded in the controller on the board.

- The IDE environment mainly contains two basic parts: Editor and Compiler where former is used for writing the required code and later is used for compiling and uploading the code into the given Arduino Module.
- This environment supports both C and C++ languages.

How to install Arduino IDE

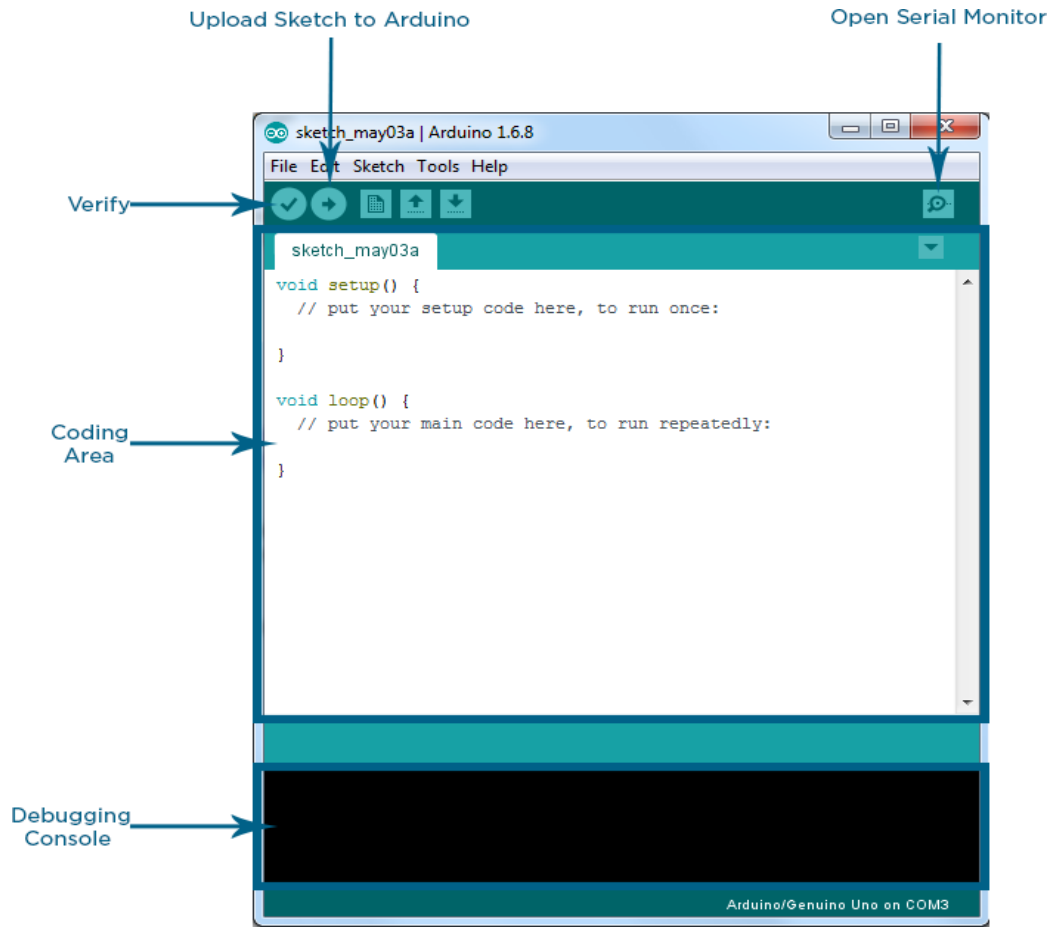
You can download the Software from [Arduino](https://www.arduino.cc/en/software) main website. As I said earlier, the software is available for common operating systems like Linux, Windows, and MAX, so make sure you are downloading the correct software version that is easily compatible with your operating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

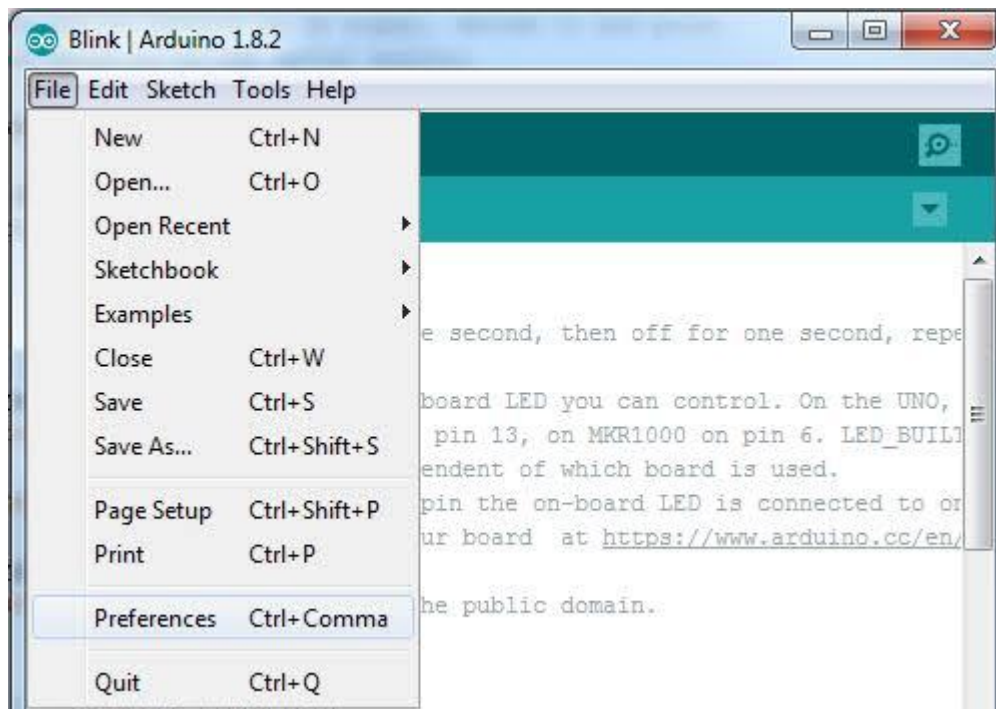
- **1. Menu Bar**
- **2. Text Editor**
- **3. Output Pane**

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

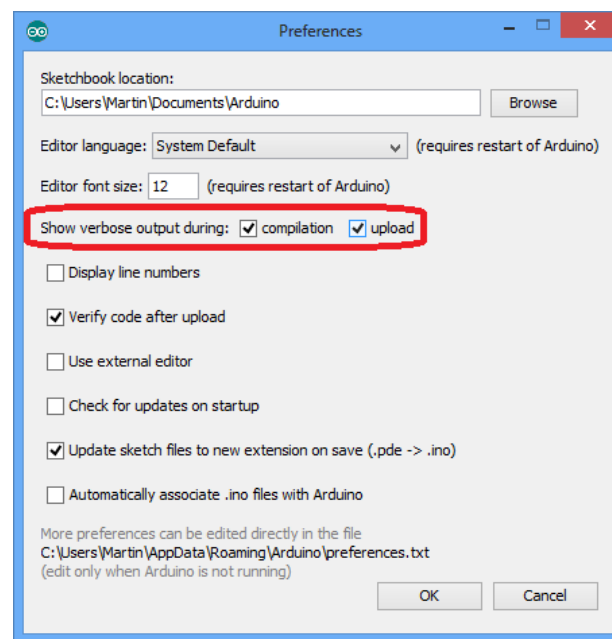


The bar appearing on the top is called **Menu Bar** that comes with five different options as follow

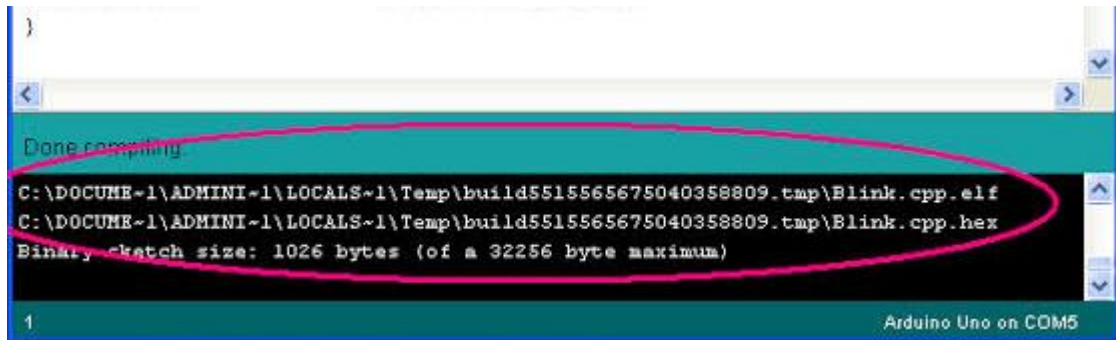
- **File** – You can open a new window for writing the code or open an existing one. Following table shows the number of further subdivisions the file option is categorized into.



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.

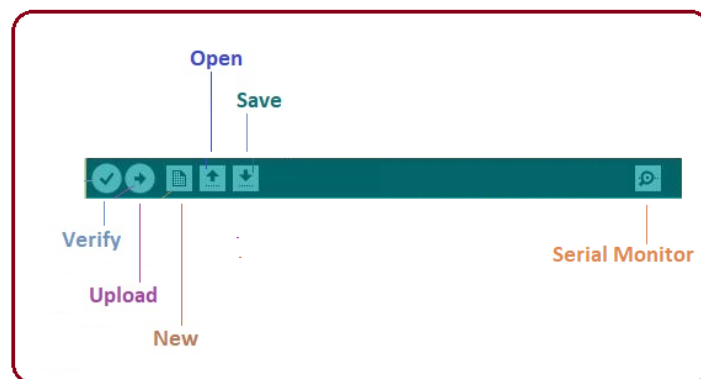


And at the end of compilation, it will show you the hex file it has generated for the recent sketch that will send to the Arduino Board for the specific task you aim to achieve.



- **Edit** – Used for copying and pasting the code with further modification for font
- **Sketch** – For compiling and programming
- **Tools** – Mainly used for testing projects. The Programmer section in this panel is used for burning a bootloader to the new microcontroller.
- **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.



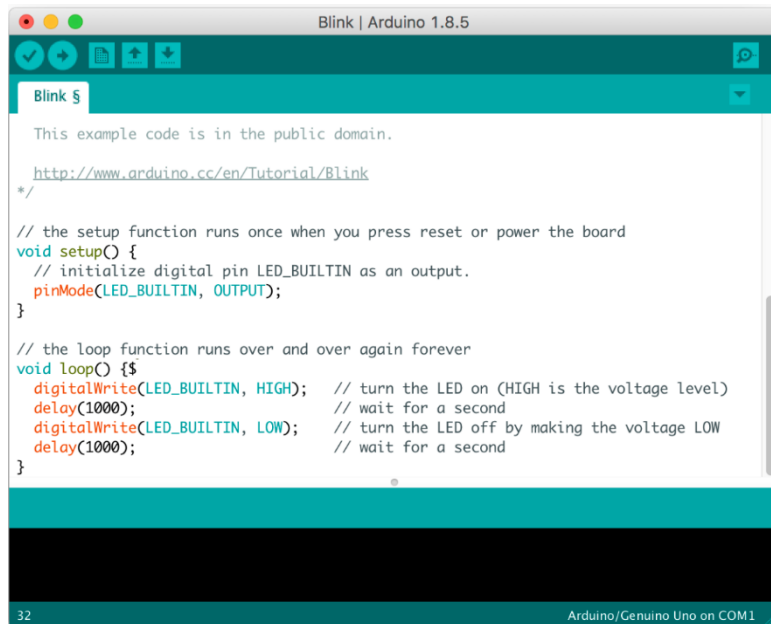
- The check mark appearing in the circular button is used to verify the code. Click this once you have written your code.
- The arrow key will upload and transfer the required code to the Arduino board.
- The dotted paper is used for creating a new file.
- The upward arrow is reserved for opening an existing Arduino project.
- The downward arrow is used to save the current running code.
- The button appearing on the top right corner is a Serial Monitor – A separate pop-up window that acts as an independent terminal and plays a vital role for sending and receiving the Serial Data. You can also go to the Tools panel and select

Serial Monitor, or pressing Ctrl+Shift+M all at once will open it instantly. The Serial Monitor will actually help to debug the written Sketches where you can get a hold of how your program is operating. Your Arduino Module should be connected to your computer by USB cable in order to activate the Serial Monitor.

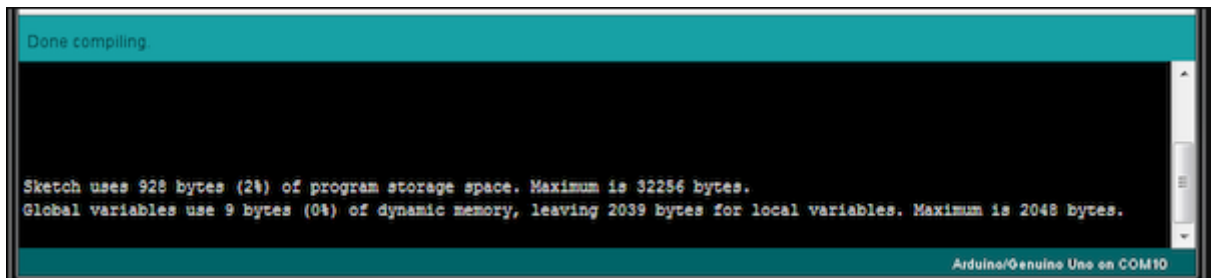
- You need to select the baud rate of the Arduino Board you are using right now. For my Arduino Uno Baud Rate is 9600, as you write the following code and click the Serial Monitor, the output will show as the image below.



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



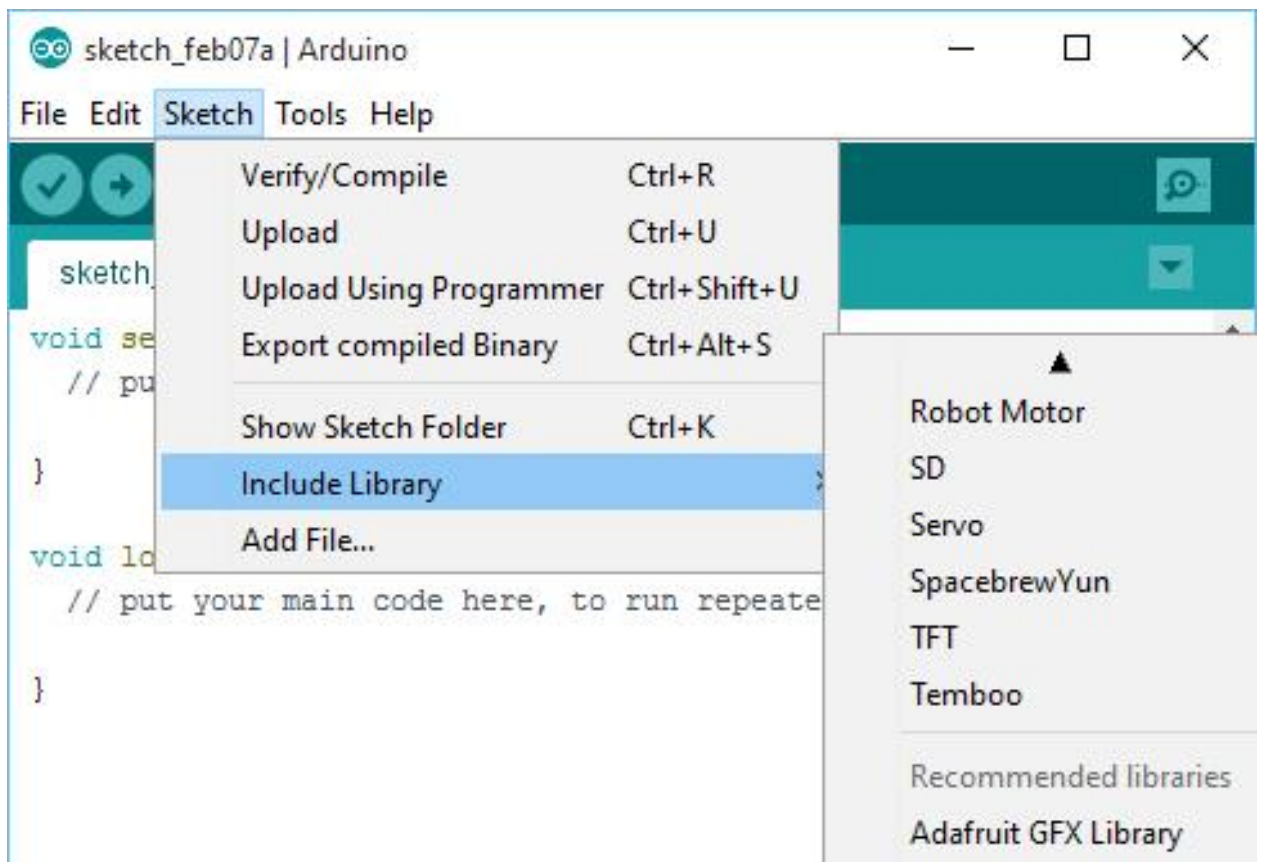
The bottom of the main screen is described as an Output Pane that mainly highlights the compilation status of the running code: the memory used by the code, and errors occurred in the program. You need to fix those errors before you intend to upload the hex file into your Arduino Module.



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.



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>.
```

Most of the libraries are preinstalled and come with the Arduino software. However, you can also download them from the external sources.

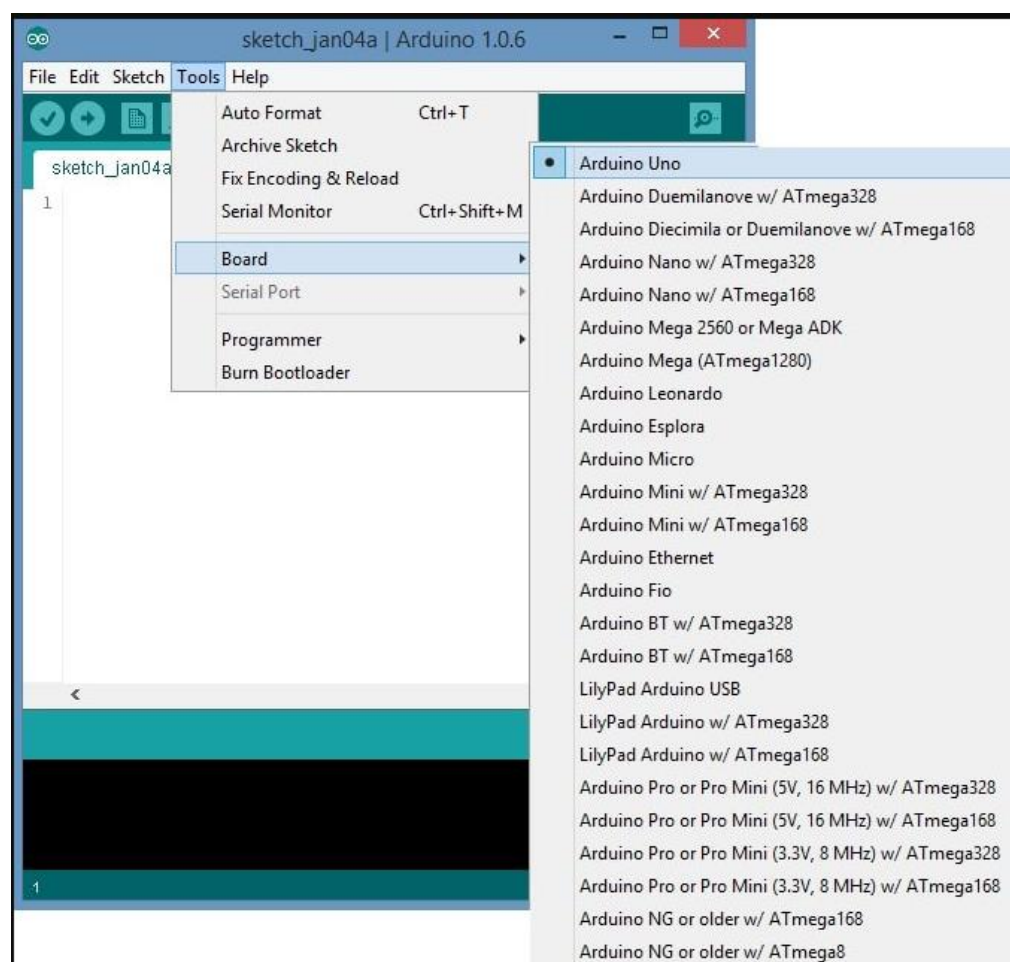
Making pins Input and output

The `digitalRead` and `digitalWrite` 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

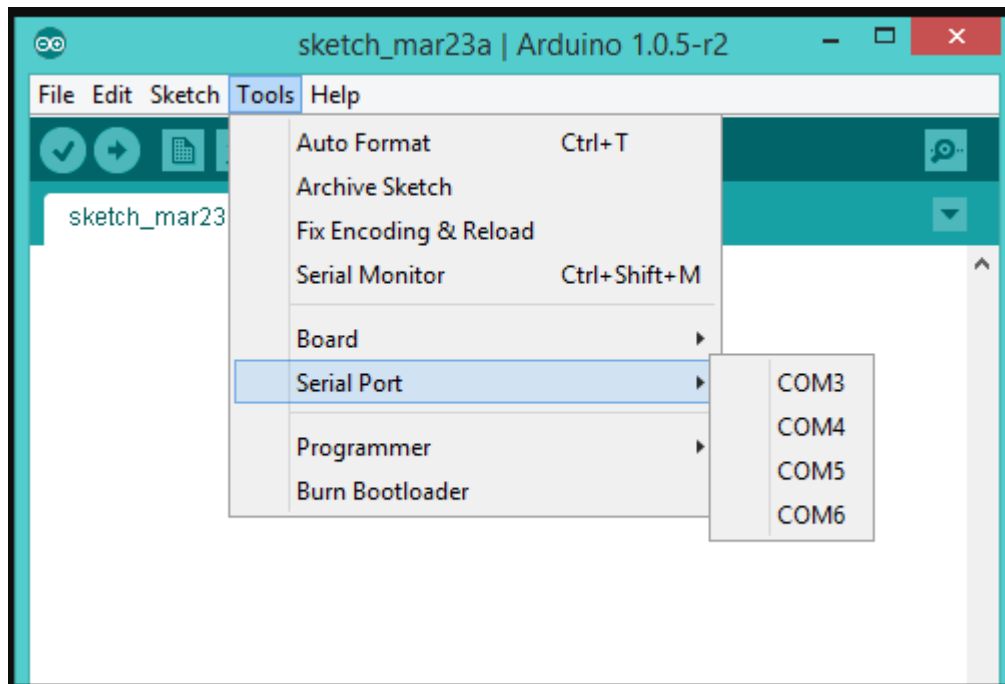
In order to upload the sketch, you need to select the relevant board you are using and the ports for that operating system. As you click the Tools on the Menu, it will open like the figure below.



- 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 board. You can look for the USB serial device in the ports section of the Windows Device Manager.

Following 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.



- 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.

It is important to note that the recent Arduino Modules will reset automatically as you compile and press the upload button the IDE software, however, older version may require the physical reset on the board.

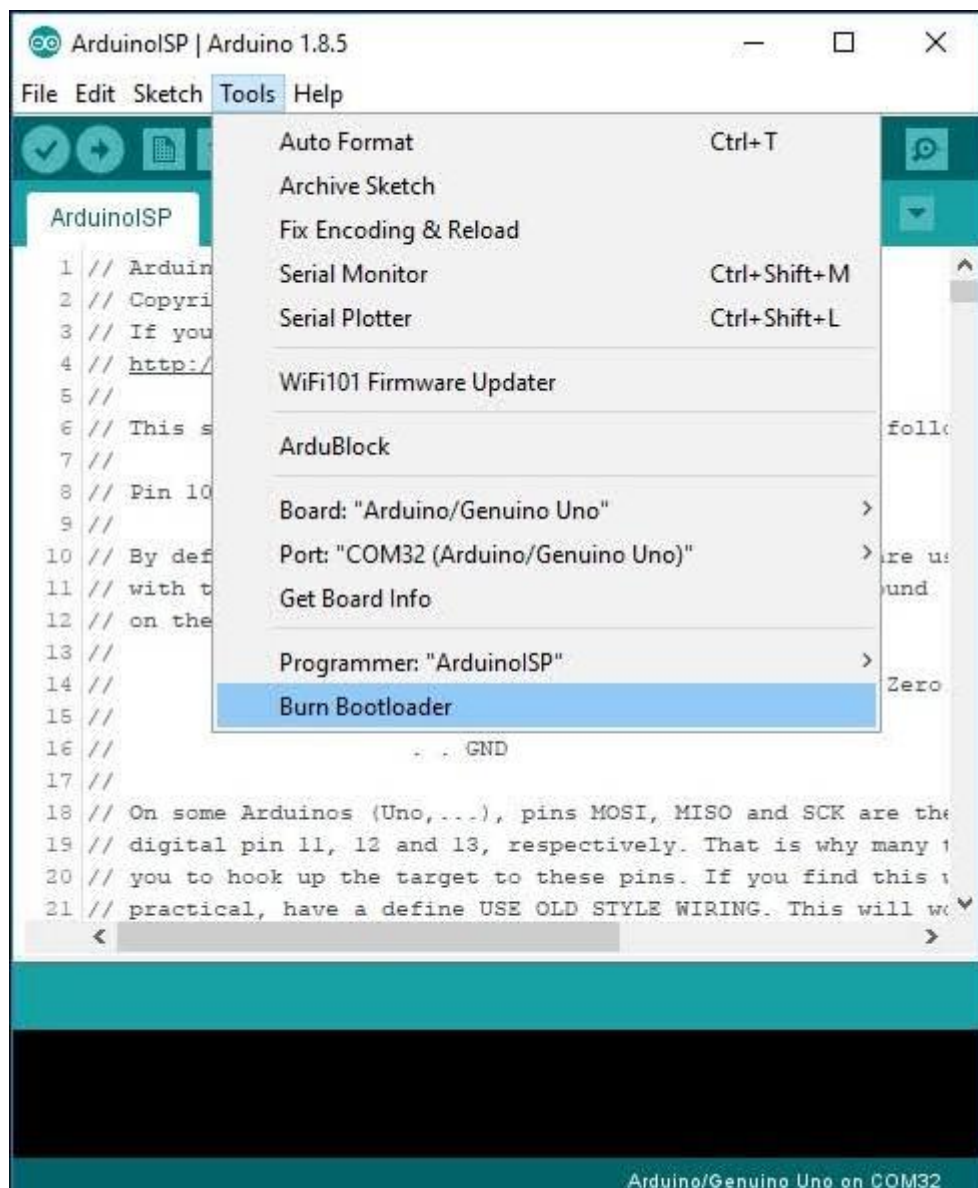
- Once you upload the code, TX and RX LEDs will blink on the board, indicating the desired program is running successfully.

Note: The port selection criteria mentioned above is dedicated for Windows operating system only, you can check this [Guide](#) 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.



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.

ADVANTAGES AND APPLICATIONS

Advantages:

- Manpower will decrease
- Takes less time
- Cost effective

Applications:

- Food storage places
- Vaccine storage places

CONCLUSION

From proposed method we can monitor the goods from remote places. This paper built a supply chain simulation platform system based on IOT technology, analyzed the application of key technology in the system, and designed smart shelves subsystem, smart shopping cart subsystem and so on. The operation process is simulated according to the particularity of the supply chain enterprise.

REFERENCES

- [1] S. Chopra and P. Meindl, Supply Chain Management: Strategy, Planning, and Operation, 5th ed. London: Pearson Education, 2013.
- [2] G. Kovács and S. Kot, "New Logistics and Production Trends as The Effect of Global Economy Changes", Polish Journal of Management Studies, Vol. 14, No.2, pp. 115–126, 2016.
- [3] A. Nettsträter, T. Geißen, M. Witthaut, D. Ebel, J. Schoneboom, "Logistics Software Systems and Functions: An Overview of ERP, WMS, TMS and SCM Systems", In: ten Hompel M., Rehof J., Wolf O. (eds) Cloud Computing for Logistics. Lecture Notes in Logistics. Springer, Cham, pp. 1-11, 2015.
- [4] P. J. Readdy, A. Gunasekaran, and A. Spalanzani, "Bottomup approach based on Internet of Things for order fulfillment in a collaborative warehousing environment", International Journal of Production Economics, Vol. 159, pp. 29–40, 2015.
- [5] S. Ibrahim, H. A. Elayat, M. M. Khater, and N. Mostafa, "Data analysis for inventory management in a multi-echelon supply chain" International Journal of Economic Resources, Vol. 2, No. 5, pp. 138–150, 2011.
- [6] A. Y. H. Nee, "Warehouse Management System and Business Performance: Case Study of a Regional Distribution Centre", Conference: International Conference on Computing and Informatics (ICOCI), 2009. [7] M. Oner, A. Budak, and A. Ustundag, "RFID-based warehouse management system in wool yarn industry", International Journal of RF Technologies, Vol. 8, No. 4, pp. 165-189, 2017.
- [8] W. Qin, R. Y. Zhong, H. Y. Dai, and Z L. Zhuang, "An assessment model for RFID impacts on prevention and visibility of inventory inaccuracy presence", Advanced Engineering Informatics, Vol. 34, pp. 70–79, 2017.

APPENDIX

A . SOURCE CODE

* Typical pin layout used:

* -----						
* MFRC522	Arduino	Arduino	Arduino	Arduino	Arduino	Arduino
* Reader/PCD	Uno/101	Mega	Nano v3	Leonardo/Micro	Pro	Micro
* Signal	Pin	Pin	Pin	Pin	Pin	Pin
* -----						
* RST/Reset	RST	9	5	D9	RESET/ICSP-5	RST
* SPI SS	SDA(SS)	10	53	D10	10	10
* SPI MOSI	MOSI	11 / ICSP-4	51	D11	ICSP-4	16
* SPI MISO	MISO	12 / ICSP-1	50	D12	ICSP-1	14
* SPI SCK	SCK	13 / ICSP-3	52	D13	ICSP-3	15
*						

* More pin layouts for other boards can be found here:

<https://github.com/miguelbalboa/rfid#pin-layout>

*

*/

```
#include <SPI.h>
```

```
#include <MFRC522.h>
```

```
#define RST_PIN 9 // Configurable, see typical pin layout above
```

```
#define SS_PIN 10 // Configurable, see typical pin layout above
```

```
MFRC522 mfrc522(SS_PIN, RST_PIN); // Create MFRC522 instance.
```

```
// Number of known default keys (hard-coded)
```

```
// NOTE: Synchronize the NR_KNOWN_KEYS define with the defaultKeys[] array
```

```

#define NR_KNOWN_KEYS 8

// Known keys, see:
https://code.google.com/p/mfcuk/wiki/MifareClassicDefaultKeys
byte knownKeys[NR_KNOWN_KEYS][MFRC522::MF_KEY_SIZE] = {
    {0xff, 0xff, 0xff, 0xff, 0xff, 0xff}, // FF FF FF FF FF FF = factory default
    {0xa0, 0xa1, 0xa2, 0xa3, 0xa4, 0xa5}, // A0 A1 A2 A3 A4 A5
    {0xb0, 0xb1, 0xb2, 0xb3, 0xb4, 0xb5}, // B0 B1 B2 B3 B4 B5
    {0x4d, 0x3a, 0x99, 0xc3, 0x51, 0xdd}, // 4D 3A 99 C3 51 DD
    {0x1a, 0x98, 0x2c, 0x7e, 0x45, 0x9a}, // 1A 98 2C 7E 45 9A
    {0xd3, 0xf7, 0xd3, 0xf7, 0xd3, 0xf7}, // D3 F7 D3 F7 D3 F7
    {0xaa, 0xbb, 0xcc, 0xdd, 0xee, 0xff}, // AA BB CC DD EE FF
    {0x00, 0x00, 0x00, 0x00, 0x00, 0x00} // 00 00 00 00 00 00
};

/*
 * Initialize.
 */
void setup() {
    Serial.begin(9600);    // Initialize serial communications with the PC
    while (!Serial);      // Do nothing if no serial port is opened (added for
                          // Arduinos based on ATMEGA32U4)
    SPI.begin();          // Init SPI bus
    mfrc522.PCD_Init();    // Init MFRC522 card
    Serial.println(F("Try the most used default keys to print block 0 of a MIFARE
PICC."));
}

/*
 * Helper routine to dump a byte array as hex values to Serial.
 */
void dump_byte_array(byte *buffer, byte bufferSize) {

```

```

    for (byte i = 0; i < bufferSize; i++) {
        Serial.print(buffer[i] < 0x10 ? " 0" : " ");
        Serial.print(buffer[i], HEX);
    }
}

/*
 * Try using the PICC (the tag/card) with the given key to access block 0.
 * On success, it will show the key details, and dump the block data on Serial.
 *
 * @return true when the given key worked, false otherwise.
 */
bool try_key(MFRC522::MIFARE_Key *key)
{
    bool result = false;
    byte buffer[18];
    byte block = 0;
    MFRC522::StatusCode status;

    // Serial.println(F("Authenticating using key A..."));

    status =
mfr522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A, block,
key, &(mfr522.uid));

    if (status != MFRC522::STATUS_OK) {
        // Serial.print(F("PCD_Authenticate() failed: "));
        // Serial.println(mfr522.GetStatusCodeName(status));
        return false;
    }

    // Read block

    byte byteCount = sizeof(buffer);

```

```

status = mfrc522.MIFARE_Read(block, buffer, &byteCount);
if (status != MFRC522::STATUS_OK) {
    // Serial.print(F("MIFARE_Read() failed: "));
    // Serial.println(mfrc522.GetStatusCodeName(status));
}
else {
    // Successful read
    result = true;
    Serial.print(F("Success with key:"));
    dump_byte_array((*key).keyByte, MFRC522::MF_KEY_SIZE);
    Serial.println();
    // Dump block data
    Serial.print(F("Block ")); Serial.print(block); Serial.print(F(":"));
    dump_byte_array(buffer, 16);
    Serial.println();
}
Serial.println();

mfrc522.PICC_HaltA();    // Halt PICC
mfrc522.PCD_StopCrypto1(); // Stop encryption on PCD
return result;
}

/*
 * Main loop.
 */
void loop() {
    // Reset the loop if no new card present on the sensor/reader. This saves the
    // entire process when idle.
    if ( ! mfrc522.PICC_IsNewCardPresent())

```

```

    return;

// Select one of the cards
if ( ! mfrc522.PICC_ReadCardSerial())
    return;

// Show some details of the PICC (that is: the tag/card)
Serial.print(F("Card UID:"));
dump_byte_array(mfrc522.uid.uidByte, mfrc522.uid.size);
Serial.println();
Serial.print(F("PICC type: "));
MFRC522::PICC_Type piccType = mfrc522.PICC_GetType(mfrc522.uid.sak);
Serial.println(mfrc522.PICC_GetTypeName(piccType));

// Try the known default keys
MFRC522::MIFARE_Key key;
for (byte k = 0; k < NR_KNOWN_KEYS; k++) {
    // Copy the known key into the MIFARE_Key structure
    for (byte i = 0; i < MFRC522::MF_KEY_SIZE; i++) {
        key.keyByte[i] = knownKeys[k][i];
    }
    // Try the key
    if (try_key(&key)) {
        // Found and reported on the key and block,
        // no need to try other keys for this PICC
        break;
    }

    // http://arduino.stackexchange.com/a/14316
    if ( ! mfrc522.PICC_IsNewCardPresent())

```

```
        break;
    if ( ! mfrc522.PICC_ReadCardSerial())
        break;
}
}
```

B. SCREENSHOTS

