WATER QUALITY MONITORING SYSTEM USING IOT

A Project report submitted in partial fulfillment of requirements
for the award of the Degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

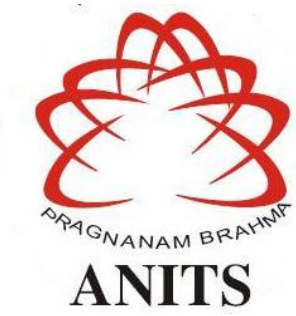
Submitted By:

G.JYOTHSNA (318126512137) **D.VISHNU VARDHAN (318126512133)**

K.ANJINI ROHINI LAKSHMI (318126512149) **I.MANOJ (319126512L15)**

Under the Esteemed Guidance

Dr.P.Murugapandiyan M.E, Ph.D
Associate Professor
Department of ECE



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES

(UGC AUTONOMOUS)

(***Permanently Affiliated to AU, Approved by AICTE and Accredited by NBA & NAAC with 'A' Grade***)

SANGIVALASA, BHEEMILI, VISAKHAPATNAM-531162, A.P

(2021-22)

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our project guide **Dr.P.Murugapandiyan**, Associate Professor, Department of Electronics and Communication Engineering, ANITS, for his guidance with unsurpassed knowledge and immense encouragement. We are grateful to **Dr. V. Rajya Lakshmi,** Professor, Head of the Department, Electronics and Communication Engineering, for providing us with the required facilities for the completion of the project work.

We are very much thankful to **Principal and Management, ANITS, Sangivalasa,** for their encouragement and cooperation to carry out this work.

We express our thanks to all **teaching faculty** of Department of ECE, whose suggestions during reviews helped us in accomplishment of our project. We would like to thank all **non-teaching staff** of the Department of ECE, ANITS for providing great assistance in accomplishment of our project.

We would like to thank our parents, friends, and classmates for their encouragement throughout the project period. At last but not least, we thank everyone for supporting us directly or indirectly in completing this project successfully.

PROJECT STUDENTS

G.JYOTHSNA (318126512137)

D.VISHNU VARDHAN (318126512133)

K.ANJINI ROHINI LAKSHMI (318126512149)

I.MANOJ (319126512L15)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES

(UGC AUTONOMOUS)

(Permanently Affiliated to AU, Approved by AICTE and Accredited by NBA & NAAC with 'A' Grade)

SANGIVALASA, BHEEMILI, VISAKHAPATNAM-531162, A.P

PRAGNANAM BRAHMA

ANITS

CERTIFICATE

This is to certify that the project report entitled **"WATER QUALITY MONITORING SYSTEM USING IOT"** submitted by **G.JYOTHSNA(318126512137),D.VISHNU VARD HAN(318126512133), K.ANJINI ROHINI LAKSHMI(318126512149), I.MANOJ(3191 26512L15)** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Engineering** in **Electronics and Communication Engineering** of Andhra University, Visakhapatnam is a bonafied work carried out under my guidance and supervision.

Project Guide

Dr.P.MurugaPandiyan
M.E,Ph.D
Associate Professor
Department of ECE
ANITS

Associate Professor
Department of E.C.E.
Anil Neerukonda
Institute of Technology & Sciences
Sangivalasa, Visakhapatnam-531 162

Head of the Department

Dr. V. Rajya Lakshmi
M.E, Ph.D, MHRM, MIEEE, MIE, MIETE
Professor
Department of ECE
ANITS

Head of the Department
Department of E C E
Anil Neerukonda Institute of Technology & Sciences
Sangivalasa-531 162

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Water is a precious resource that is already scarce in many places and has become a non-renewable resource due to the rapid development of the social economy, and excessive discharge of industrial sewage. This made an urge to know the water quality of different water bodies present around us and make them clean for domestic use as well as for the life of the aquatic animals. We should prevent the water from water pollution, to do that we need to acquire multiple parameter data that affect water quality, and timely and accurately monitor water quality information. This data can give an insight into any potential problems that may arise so that repairs or adjustments can be made before it is too late.

CHAPTER -1

INTERNET OF THINGS

1.1 INTRODUCTION

The Internet of Things is a novel paradigm that has received considerable attention from both academia and industry. The basic idea of IoT is the pervasive presence around us of a variety of things or objects such as sensors, actuators, radio frequency identification tags, mobile phones, etc. which through unique addressing schemes, are able to interact with each other and cooperate with others to reach common goals.

With the rapid development of science and technology, the world is becoming "smart". Living in such a smart world, people will be automatically and collaboratively served by the smart devices (like watches, mobile phones, computers), smart transportation (like cars, buses, trains), smart environments (like home, factories) etc. Our world is consisted of various "things". As one of the enablers of smart world, Internet of Things targets to connect various objects with unique addressees, to enable them are interacting with each other and with the world.

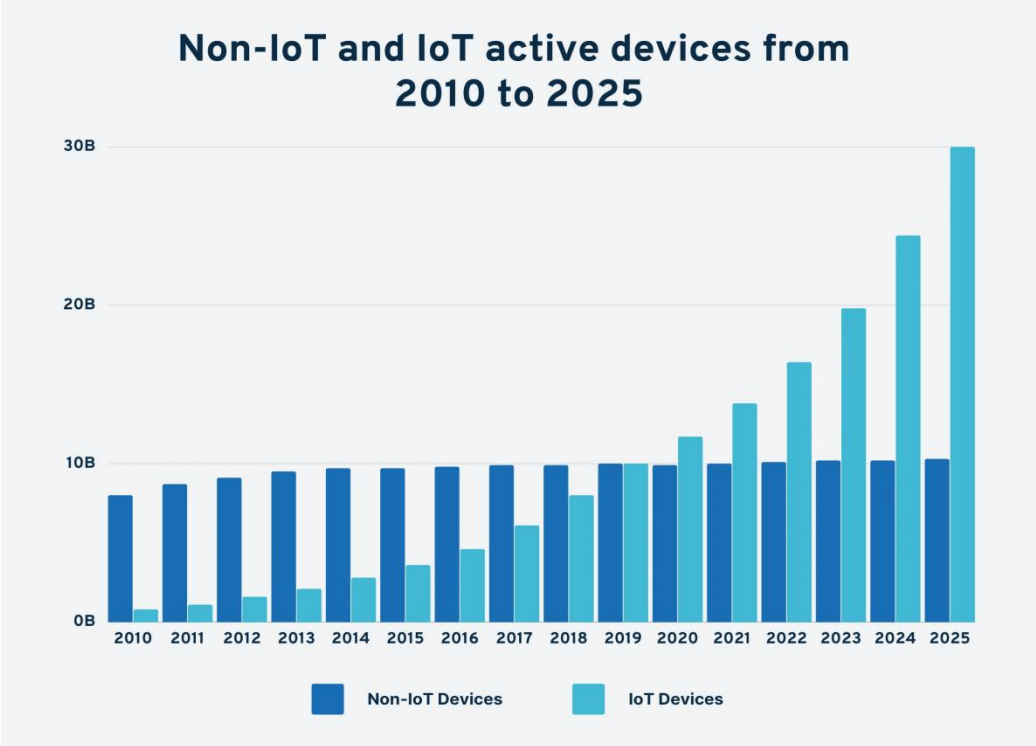


Figure 1.1:IoT users by 2025 (Source:Google)

Internet of Things is used to connect all things in the world to the internet. It is expected that things can be identified automatically, can communicate with each other and can even make decisions by them.

The research groups, academics, and government departments have paid attention to revolutionizing the internet, by constructing a more convenient environment that is composed of various intelligent systems, such as smart home, intelligent transportation and health care.

1.2 DEFINITION AND EXAMPLE OF IOT

1.2.1 DEFINITION BY ITU-T

"In a broad perspective, the loT can be perceived as a vision with technological and societal implications. From the perspective of technical standardization, IoT can be viewed as a global infrastructure for the information society, enabling advanced services by interconnecting things based on, existing and evolving, interoperable information and communication technologies. Through the exploitation of identification data capture, processing and communication capabilities, the IoT makes full use of things to offer services to all kinds of applications, while maintaining the required privacy."

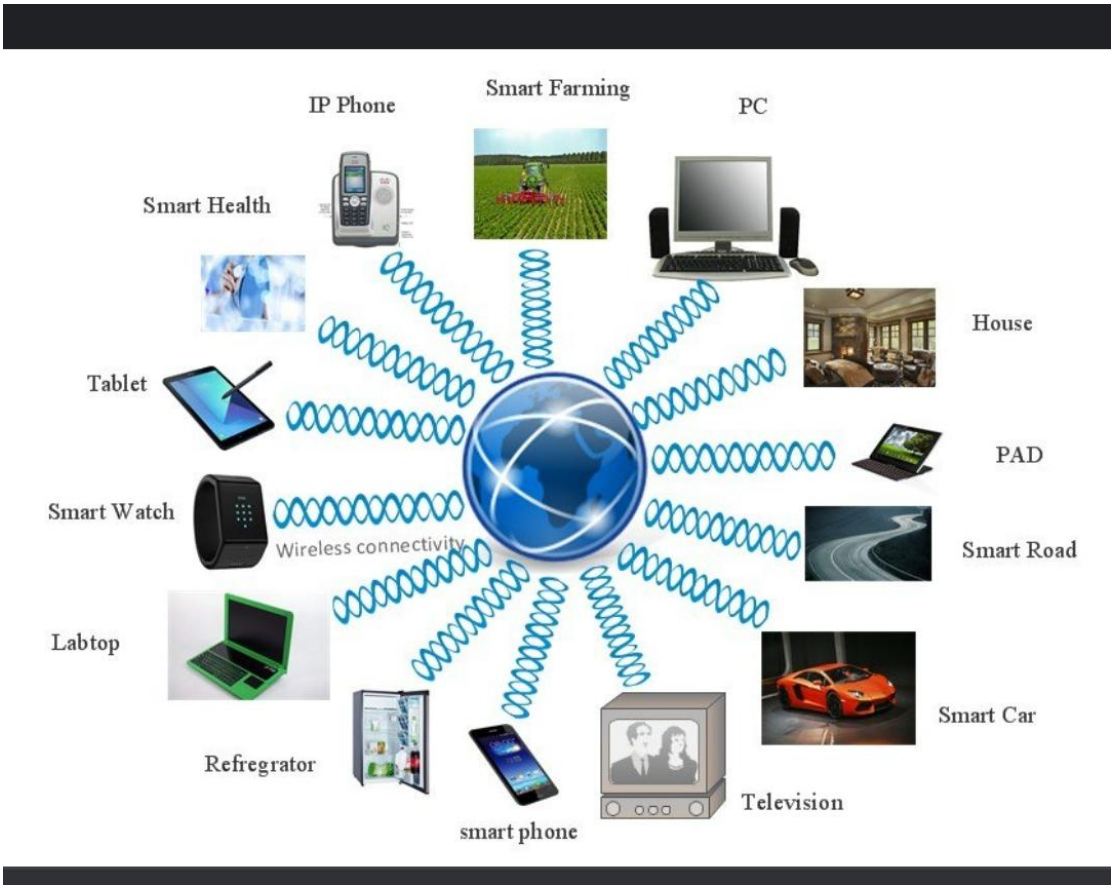


Figure 1.2: An exmaple of IoT (Source:Google)

1.2.2 ELEMENTS OF IOT

There are six elements in IoT. They are identification, sensing, communication technologies, computation, services and semantic.Identification plays a crucial role in naming and matching services with their demand. Ex: Electronic Product Code (EPC)Sensing is for collecting various data from related objects and sending it to a database.

Ex: temperature sensors.Communication technologies connect heterogeneous objects together to offer.special services. Ex. Wi-Fi, Bluetooth Computation, the hardware processing units and software applications perform.this task. The services in IoT are identity related services, information aggregation services, collaborative aware services and ubiquitous services.Semantic means the ability to extract knowledge intelligently so as to provide the required services.

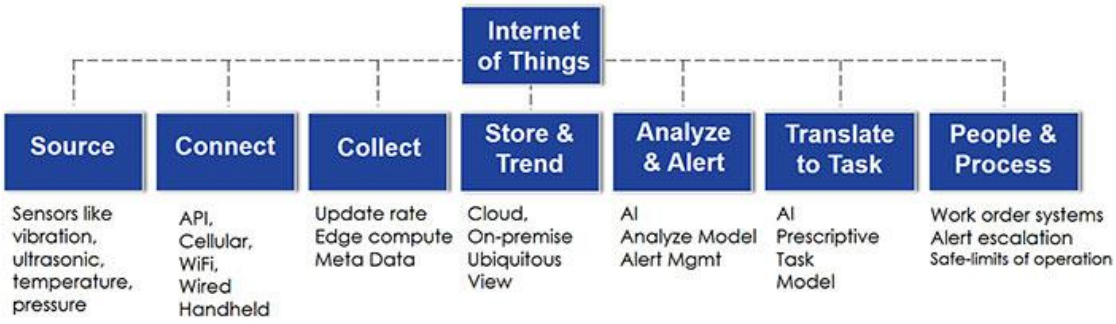


Figure 1.3: Elements of IoT (Source:Google)

1.3 APPLICATIONS OF IOT

There are a lot of applications for loT. We list some application scenarios as follows.

1. Smart Home: Personal life style at home is enhanced, by aking it more i convenient and easier to monitor and operate home appliances and systems remotely.

2. Indusrial Automation: With the minimum involvement, robotic evices are computerized to finish manufacturing tasks. In addition the productivity is improved.

3. Smart Healthcare: Performance of healthcare applications is improved, by embedding sensors and actuators in patients and their medicine for monitoring and tracking patients. For example, by gathering and analyzing patients body data with sensors and further delivering analyzed data to processing center

4. Smart Grid: Power suppliers are assisted to control and manage, resources so that power can be offered proportionally to the population growth. Therefore, the energy consumption of houses and buildings could be enhanced. For example, the meters are connected to the network of energy providers, then the energy providers could enhance their services, by collecting, analyzing, controlling, monitoring, and managing energy consumption.

5. Smart City: Quality of life in the city is ameliorated, by making it more convenient and easier for the residents to obtain information of interest. For instance, according to people's needs, various interconnected systems intelligently offer he desirable services (e.g., transportation, utilities, health, etc) to people.

CHAPTER -2

PROJECT DESCRIPTION

2.1 ABOUT THE PROBLEM:

Water quality plays a very important part in the health of animals and human beings. Lakes and reservoirs, and canals are one of the major sources of drinking water. The first step toward water pollution control is to be able to monitor the actual level of water pollution. This challenge occurred because of limited water resources growing population, and aging infrastructure. So there is a need for better methodologies for monitoring the water quality and traditional methods of water to character the water quality.

Although the current methodologies analyze the physical, chemical, and biological agents, it has several drawbacks:

1. Poor spatiotemporal coverage

2. It is labor-intensive and high cost (labor, operation; and equipment)

3. The lack of real-time water quality information to enable critical decisions for public health protection.

Therefore, there is a need for continuous water quality monitoring.

2.2 INTRODUCTION:

Water is the source of life, a direct impact on human health. It is a substance that can be seen everywhere in our daily life and is an indispensable resource for human life and the development of industry and agriculture. With the rapid development of the social economy, the water quality problem has been exposed gradually. The discharge of industrial sewage and urban and rural domestic sewage far exceeds the self-purification capacity of ecosystems. Whether for drinking, household, food production, or recreational activities, safe and readily available water is needed for public health. Once the water source is polluted, people's life and health will be seriously threatened.

Traditional aquaculture relies on farmers to monitor the water quality and surrounding for a long time. The most important work in the culturing process is the mastery of water quality parameters of the fish pond, such as water temperature, PH, and level. These water quality parameters affect the survival rate of cultured species

directly, so farmers must monitor the water quality for a long time frequently. The water quality measured by an automated system is generally transmitted to the back end via a wireless network. Therefore, the transmission of water quality information consuming too much power will increase the culturing costs. In rural areas, due to the lack of safety awareness and professional knowledge, the lack of attention to the safety of drinking water, agricultural great significance to the accelerating of the agricultural production process using pesticides, fertilizers, garbage, human and animal manure, and another improper handling, often cause drinking water Or non-point source pollution of the soil in the water source. And the geographic distribution information is possessed by water, it is necessary to monitor the water quality. The water quality monitoring system consists of water quality measuring equipment, data acquisition equipment, and a data transmission system. Water quality measurement equipment is mainly based on all kinds of sensors. Data acquisition equipment is usually controlled by an MCU. The data transmission system is mainly divided into the wired transmission and wireless transmission. For a general scenario such as the household environment, pH, turbidity, and water temperature are usually monitored to assess the water quality comprehensively, which can reflect the water condition. The current main monitoring method is regular manual sampling and analysis. But it takes more manual effort and costs much. So we are trying to make an efficient model.

2.3 BLOCK DIAGRAM AND ITS OPERATION:

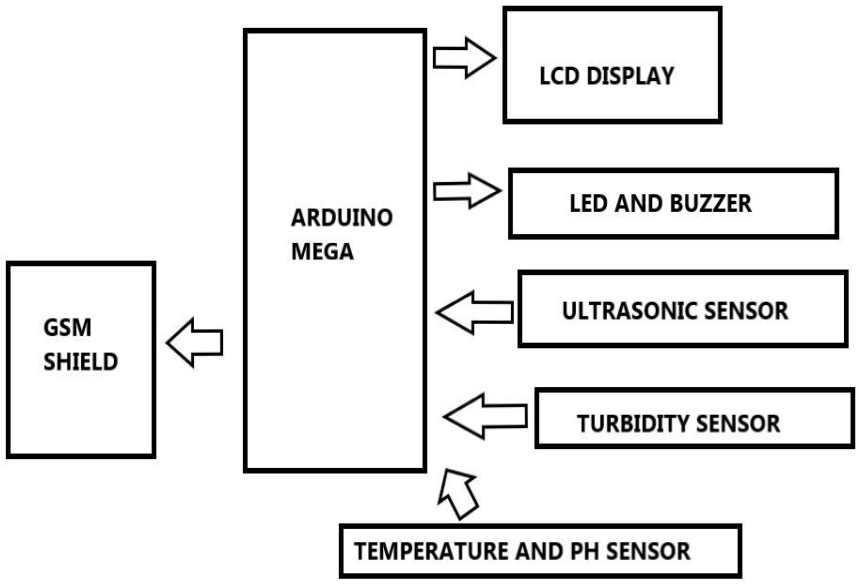


Figure 2.1: Block diagram of water quality monitoring system using IoT (Google)

2.4 HARDWARE COMPONENTS USED:

1.Arduino Mega Board.

2. (16 * 2) LCD display.

3. The 2 in one Temperature and PH sensor.

4. The Turbidity Sensor.

5. A GSM shield.

6. An ultrasonic Sensor.

7. 4 RGB led.

8. A buzzer.

2.5 SOFTWARE USED:

Arduino IDE (Integrated Development Environment).

CHAPTER-3

ARDUINO MEGA

3.1 WHAT IS ARDUINO:

Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a Microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

The Arduino platform has become quite popular with people just starting with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) to load new code onto the board we can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the microcontroller into a more accessible package.

It simplifies the process of creating any control system by providing the standard board that can be programmed and connected to the system without the need for any sophisticated PCB design and implementation.

3.2 WHY ARDUINO:

- It is an open-source project, software/hardware is extremely accessible and very flexible to be customized and extended.
- It is flexible, offers a variety of digital and analog inputs, *SPI* (serial interface), digital and PWM outputs.
- It is easy to use, connects to computer via USB and communicates using standard serial protocol, runs in standalone mode and as interface connected to PC/Macintosh computers.
- It is inexpensive board and comes with free authoring software.
- It have own MAC address to identify in network.

3.3 WHAT DOES ARDUINO DO?

The arduino hardware and software was designed for artists, designers, hobbyists, hackers, newbie's, and anyone interested in creating interactive objects or environments. Arduino can interact with motors, GPS units, cameras, the internet, and even smart phone or TV.

- It works with a sensors.
- Push buttons touch pads, tilt switches.
- Variable resistors (eg. volume knob / sliders)
- Photoresistors (sensing light levels)
- Thermistors (temperature)
- Ultrasound (proximity range finder)
- It works with Actuators
- Lights, LEDs
- Motors
- Speakers
- Displays (LCD)

3.4 ARDUINO USES

- Physical Computing projects / research.
- Interactive Installations. Rapid prototyping.

3.5 TYPES OF ARDUINO

There have been many revisions of the USB Arduino. Some of them are,

- Arduino Uno

• Arduino Leonardo

• Arduino LilyPad

• Arduino Mega

• Arduino Nano

• Arduino Mini

• Arduino Mini Pro

• Arduino BT

Boards	Microcontroller	Operating Voltage/s (V)	Digital I/O Pins	PWM Enabled Pins	Analog I/O Pins	DC per I/O (mA)	Flash Memory (KB)	SRAM (KB)	EEPROM (KB)	Clock (MHz)	Length (mm)	Width (mm)	Cable	Native Network Support
Uno	ATmega328	5	14	6	6	20	32	2	1	16	68.6	53.4	USB A-B	None
Leonardo	ATmega32u4	5	20	7	12	40	32	2.5	1	16	68.6	53.3	micro-USB	None
Micro	ATmega32u4	5	20	7	12	40	32	2.5	1	16	48	18	micro-USB	None
Nano	ATmega328	5	22	6	8	40	32	2	0.51	16	45	18	mini-B USB	None
Mini	ATmega328	5	14		6	20	32	2	1	16	30	18	USB-Serial	None
Due	Atmel SAM3X8E ARM Cortex-M3 CPU	3.3	54	12	12	800	512	96	X	84	102	53.3	micro-USB	None
Mega	ATmega2560	5	54	15	16	20	256	8	4	16	102	53.3	USB A-B	None
M0	Atmel SAMD21	3.3	20	12	6	7	256	32	X	48	68.6	53.3	micro-USB	None
Yun Mini	ATmega32u4	3.3	20	7	12	40	32	2.5	1	400	71.1	23	micro-USB	Ethernet/Wifi
Uno Ethernet	ATmega328p	5	20	4	6	20	32	2	1	16	68.6	53.4	Ethernet	Ethernet
Tian	Atmel SAMD21	5	20	12	0	7	16000	64000	X	560	68.5	53	micro-USB	Ethernet/Wifi
Mega ADK	ATmega2560	5	54	15	16	40	256	8	4	16	102	53.3	USB A-B	None
M0 Pro	Atmel SAMD21	3.3	20	12	6	7	256	32	X	48	68.6	53.3	micro-USB	None
Industrial 101	ATmega32u4	5	7	2	4	40	16000	64000	1	400	51	42	micro-USB	Ethernet/Wifi
Uno Wifi	ATmega328	5	20	6	6	20	32	2	1	16	68.6	53.4	USB A-B	Wifi
Leonardo Ethernet	ATmega32u4	5	20	7	12	40	32	2.5	1	16	68.6	53.3	USB A-B	Ethernet
MKR1000	Atmel SAMD21	3.3	8	12	7	7	256	32	X	48	64.6	25	micro-USB	Wifi

Table 3.5: Comparision of different Arduino boards (Google)

3.6 ARDUINO MEGA :

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins (of which 15 can be used as PWM outputs), 16

analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, and a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started. The Mega 2560. It can be powered via USB or an AC-to-DC adapter and is compatible with most shields designed for the Arduino Uno.

3.6.1 DIFFERENCES WITH OTHER BOARDS:

The Mega Board runs on the ATmega2560 MCU, a step up from the 328p & 32u4. The Mega provides everything these other boards do, but adds a ton of extra pins to make bigger, more ambitious projects possible! CNC machinery, 3D Printing & Home automation can all require quite a lot of IO pins to get running. The Mega is the board for this very purpose.

- 16MHz Clock
- 256 KB Flash Memory
- 8 KB SRAM
- 54 Digital I/O Pins, 15 of these can be used with PWM.
- 16 Analog Input Pins
- Can be run with anything from 5-12V DC power.

While the Mega 2560 is slightly heavier on the pocket than the smaller boards, the ridiculous amount of IO pins and SHIELD compatibility it comes with make it worth the money. A beefier brother to the Uno, this board is the go-to choice for larger projects.

3.6.2 OPERATIONAL SPECIFICATIONS:

MICROCONTROLLER	ATmega2560
OPERATING VOLTAGE	5V
INPUT VOLTAGE (RECOMMENDED)	7-12V
INPUT VOLTAGE (LIMIT)	6-20V
DIGITAL I/O PINS	54 (of which 15 provide PWM output)
ANALOG INPUT PINS	16
DC CURRENT PER I/O PIN	20 mA
DC CURRENT FOR 3.3V PIN	50 mA
FLASH MEMORY	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
CLOCK SPEED	16 MHz
LED_BUILTIN	13
LENGTH	101.52 mm
WIDTH	53.3 mm

Table 3.6: Operational Specifications (Google)

3.7 ARDUINO FUNCTIONAL BLOCKS:

3.7.1 Programming

The Mega 2560 board can be programmed with Arduino Software (IDE). For details, see the reference and tutorials.

The ATmega2560 on the Mega 2560 comes preprogrammed with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using Arduino ISP or similar; see these instructions for details.

The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available in the Arduino repository. The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

- On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2.
- On Rev2 or later boards: there is a resistor that pulls the 8U2/16U2 HWB line to the ground, making it easier to put into DFU mode. You can then use Atmel's FLIP software(Windows) or the DFU programmer(Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorial for more information.

3.7.2 Warnings

The Mega 2560 has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

3.7.3 Power

The Mega 2560 can be powered via a USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector.

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

The power pins are as follows:

- Vin. The input voltage to the board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- 5V. This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
- 3V3. A 3.3 volt supply is generated by the onboard regulator. The maximum current draw is 50 mA.
- GND. Ground pins.
- IOREF. This pin on the board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs for working with the 5V or 3.3V.

3.7.4 Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

3.7.5 INPUT AND OUTPUT

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode(),digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50 k ohm. A maximum of 40mA is the value that must not be exceeded to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX). Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega16U2 USB-to-TTL Serial chip.
- External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2). These pins can be configured to trigger an interrupt on a low level, a rising or falling edge, or a change in level. See the attachInterrupt() function for details.
- PWM: 2 to 13 and 44 to 46. Provide 8-bit PWM output with the analogWrite() function.
- SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). These pins support SPI communication using theSPI library. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Arduino /Genuino Uno and the old Duemilanove and Diecimila Arduino boards.
- LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- TWI: 20 (SDA) and 21 (SCL). Support TWI communication using the Wire library. Note that these pins are not in the same location as the TWI pins on the old Duemilanove or Diecimila Arduino boards.

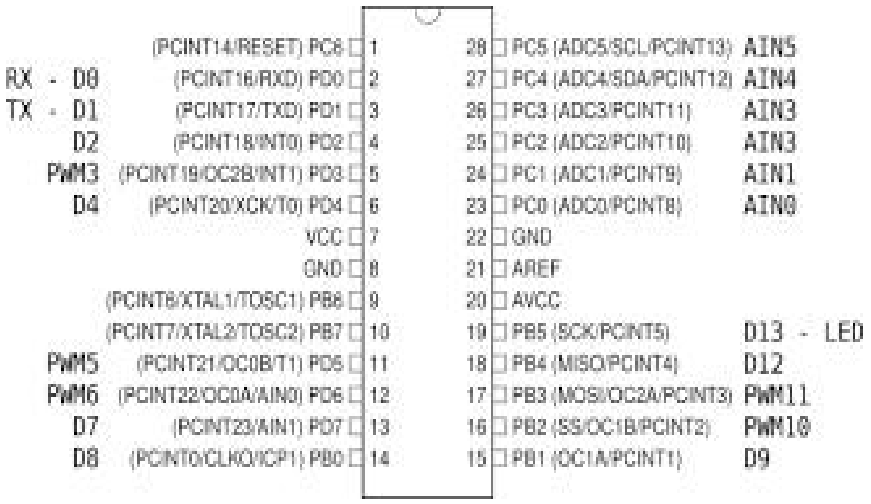


Figure 3.7:Arduino Mega Pin Diagram (Google)

The Mega 2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function.

There are a couple of other pins on the board:

- AREF. Reference voltage for the analog inputs. Used with analogReference().
- Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

3.7.6 COMMUNICATION

The Mega 2560 board has a number of facilities for communicating with a computer, another board, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega16U2 (ATmega 8U2 on the revision 1 and revision 2 boards) on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2/ATmega16U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Mega 2560's digital pins.

The Mega 2560 also supports TWI and SPI communication. The Arduino Software (IDE) includes a Wire library to simplify use of the TWI bus; see the documentation for details. For SPI communication, use the SPI library.

3.7.7 RESET BUTTON

Rather then requiring a physical press of the reset button before an upload, the Mega 2560 is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega2560 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino Software (IDE) uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Mega 2560 board is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the ATMega2560. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Mega 2560 board contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

3.7.8 VOLTAGE REGULATOR

The voltage regulator converts the input voltage to 5V. The primary use of a voltage regulator is to control the voltage level in the Arduino board. Even if there are any changes in the input voltage of the regulator, the output voltage is constant and steady.

3.7.9 APPLICATIONS

Arduino was basically designed to make the process of using electronics in multidisciplinary projects more accessible. It is intended for artists, designers and anyone 19interested in creating interactive objects or environments. Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. Because of these features, Arduino finds extensive application in various fields. Arduino projects can be stand-alone or they can communicate with software running on a computer.

CHAPTER-4

ARDUINO PROGRAMMING

Arduino programming can be (variables and constants), and functions.divided into three parts. They are structure, values

4.1 STRUCTURE

The basic structure of Arduino programming language is simple and runs in two parts. These two parts, or functions, enclose blocks of statements.

```
Void setup()
{
Statements;
}
Void loop()
```

} where setup() is the preparation, loop() is the execution. Both functions are required for the program to work.

The setup function should follow the declaration of any variables at the beginning of the program. It is the first function to run the program, is run only once and is used to set the pin Mode or initialize the serial communication.

The loop function follows next and includes the code to be executed continuously byreading inputs, triggering outputs. This function is the core of arduino programs and does a lot of work.

4.1.1 SETUP

The setup function is called once when the program starts. We use it to initialize pin modes or to begin serial communication. It must be included in the program even if there are no statements to run.

Syntax: Void setup()

```
{
pinMode(pin, OUTPUT); // sets the 'pin' as output
}
```

4.1.2 LOOP

The loop() function does precisely what its name suggests and loops consecutively allowing the program to change, respond, and control the Arduino board.

Syntax: Void loop()

{

digitalWrite(pin, HIGH) ;turns pin delay(1000);// pauses for one second

4.1.3 OTHER STRUCTURES

1. CONTROL STRUCTURES

There are many control structures which are used in the programming language like if, if else, while, do while, for, switch case, break, continue, return, goto. They are mainly used for controlling after a certain condition is reached. Some control structures used in this project are listed below.

IF

If statements test whether a certain condition has been reached, such as an value being above a certain value, and executes the statements inside the brackets if it is true. If false the program skips over the statement

Syntax: If (some variable ?? value)

Statements;

}

While

While loops will loop continuously, and infinitely, until the expression inside the parenthesis becomes false. Something must change the tested variable, or the while loop will never exit. This could be in the code such as an incremented variable, or an external condition, such as testing a sensor.

Syntax: While (variable<200) {

Control statement;

Variable++

}

Do..while

The do loop is a bottom driven loop that works in the same manner as the while loop. with the exception that the condition is tested at the end of the loop, so the do loop will always run at least once.

Syntax: Do

Control Statements;

} while (condition);

If..else

If..else allows for 'either-or' decisions. For example, if we wanted to test a digital input, and do one thing if the input went high or instead do another thing if the input was low.

Syntax: if (input pin=high)

Control statementl;

}

Else

Control statement2;

}

It is even possible to have unlimited number of these else branches but only one set of statements will be run depending on the test conditions.

2. FURTHER SYNTAX

Further syntax like semicolon, curly braces, single line comments, multiline comments,define, #include are used in the programming.

1) A semicolon must be used to end a statement and separate elements of the program. For ex: int x = 11;

2)Curly braces define the beginning and end of function blocks and statement blocks such as void loop function and the for and if statements.

3) #include is used to include any library functions.

4)Single line comments begins with // and end with the next line of code. They are ignored by the program and take no memory space.

5)Multiple line comments are also ignored by the program and are used for large

6)text description of code. Syntax:/they are balanced so place closing comment/

3. ARITHMETIC OPERATORS

Arithmetic operators include addition, subtraction, multiplication, division. They return the sum, difference, product or quotient (respectively) of the two operands.

Example: y=y+3

x= x-3

i=j*6

r=r/5

4. COMPARISON OPERATORS

Comparisons of one variable or constant against another are often used in if statements to test if a specified condition is true.

Examples:

1 X=Y // X is equal to Y

2 X =y //X is equal to y

3 X <- Y // X is less than or equal to Y

4 X >= Y // X is Greater than than or equal to Y

5 X>Y //X is Greater than Y

6 X<Y // X is Less than Y

5. LOGICAL OPERATORS

Logical operators are usually a way to compare two expressions and return a TRUE or FALSE depending on the operator. There are three logical operators AND, OR, NOT that are often used in if statements.

* Logical AND: if (x>3 && x<7) is true only if both expressions are true.

* Logical OR: if (x>3 || x<7) is true if either of the expressions is true.

* Logical NOT: if (!x>3) is true only the expression is false.

6. COMPOUND OPERATORS

Compound operators combine an arithmetic operation with a variable assaignment. These are commonly found in for loops. The most common compound operators are

1 X ++ //same as X=X+1

2 X -- //same as X=X-1

3 X+=Y //same as X=X+Y

4 X-+Y //same as X=X-Y

5 X*=Y //same as X=X*Y

6 X/=Y //same as X=X/Y

4.2 VARIABLES

A variable is a way of naming and storing a numerical value for later use by the program. As its name suggests variables are numbers that can be continually changed as opposed to constants whose value never changes. A variable needs to be declared and optionally assigned to the value need to be changed.

4.2.1 VARIABLE SCOPE

A variable can be declared at the beginning of program before void setup(), locally inside the functions, and sometimes with in statement block such as for loops. Where the variable is declared determines the variable scope, or the ability to certain parts of the program to make use of the variable.

A global variable is the one that can be seen and used by every function and statement in a program. This variable is declared at the beginning of program, before the setup() function.

A local variable is the one that is defined inside a function or as a part of loop. It is only visible and can only be used inside the function in which it was declared. It is therefore possible to have two or more variables of the same name in different parts of the same program contain different values.

4.2.2 CONSTANTS

The Arduino language has a pre defined values, which are called constants. They are used to make the program easier to read. Constants are classified in groups:

True/False

These are boolean constants that define logic levels. FALSE is easily defined 0 (m) while TRUE is often defined as one, but can also be anything else expect zero, So in a Boolean sense, -1,2, and -200 are all also defined as TRUE.

high/Low

These constants define pin levels as HIGH or LOW and are used when reading or writing to digital pins, HIGH is defined as logic level 1, ON, or 5 volts while LOW is logic level 0, OFF, or 0 volts.

Example: digitalWrite(13, HIGH);

Input/Output

Constants used with the pinMode() function to define the mode of a digital pin as either input or output.

Example: pinMode(13,OUTPUT);

4.3 FUNCTIONS

Segmenting code into functions allows a programmer to create modular pieces of code that perform a defined task and then return to the area of code from which the function was "called". The typical case for creating a function is when one needs to perform the same action multiple times in a program.

4.3.1 DIGITAL I/O

pinMode(pin, mode)

pinMode is used in void setup() to configure a specified pin to behave either as an INPUT or OUTPUT.

Example: pinMode(pin, OUTPUT); // set 'pin' to output mode

Arduino digital pins default to inputs, so they do not need to be explicitly declared as inputs with pinMode(). Pin configured as INPUT are said to be in a high

impedance state and pins configured as OUTPUT are said to be in a low impedance state

Short circuits on Arduino pins and excessive current can damage or destroy the output pin, or damage the entire Atmega chip.

digitalWrite(pin, value)

Outputs either high or low at(turn off or on) a specified digital pin. The pint can be specified as either a variable or constant (0-13). Example: digitalWrite(pin, HIGH), //sets 'pin' to high

digitalRead(pin)

It reads the value from a specified digital pin with the result either HIGH or LOW. The pin can be specified as either a variable or constant (0-13).

Example: Value-digitalWrite(pin); //sets 'value' equal to the input pin

4.3.2 ANALOG I/O analogRead(pin)

It reads the value from a specified analog pin with a 10-bit resolution. This function only works on the analog in pins (0-5). The resulting integer values range from 0 to 1023. Analog pins unlike digital ones do not need to be first declared as INPUT nor OUTPUT Value-analogRead(pin):

Example: Value = analogRead(pin): sets 'value' equal to pin

analogWrite(pin, value)

It writes a pseudo analog value using hardware enabled pulse width modulation (PWM) to an output pin marked PWM. This is a hardware function, the pin will generate steady wave after a call to analog Write in the background until the next call to analogWrite (or a call to digitalRead or digitalWrite on the same pin).

a value of 0 generates a steady 0 volts output at the specified pin, a value of 255 generates a steady 5 volts output at the specified pin.

4.3.3 OTHER FUNCTIONS

1.TIME FUNCTIONS

delay (ms)

It pauses the program for the amount of time as specified in milliseconds, where 1000

equals 1 second.

Example: delay(1000); // waits for one second

millis()

Returns the number of milliseconds since the Aruino board began running the current program as an unsigned long value. This number will overflow (reset back to zero), after approximately 9 hours.

Example: value = millis(); // sets 'value' equal to millis())

2. MATH FUNCTIONS

min(x,y)

It calculates the minimum of two numbers of any data type and returns the smaller number.

number. min(value, 100); sets 'value the smaller 'value' 100.

max(x,y)

It calculates the maximum of two numbers of any data type and returns the biggest number.

Example: 100.

3. SERIAL COMMUNICATION

Serial.begin(rate)

Opens serial and sets the baud for serial data transmission. The baud rate for communicating with the computer is 9600 althrough other speeds are supported.When

using serial communications,digital pins 0(RX) and 1 (TX) cannot be used at the same time.

Example :

```
Void setup(){
Serial.begin(9600);
}
```

Serial.println(data)

It prints data to the serial port ,followed by an automatic carriage return and line feed.This command takes the same form as serial.print(),but is easier for reading data on the serial monitor.

Example: Serial.println(analogValue); //sends the value of 'analog value'

4.4 LIBRARIES

The Arduino environment be extended through use libraries, just like most programming platforms. Libraries provide functionality working with hardware manipulating data.

Libraries a collection code that makes it easy for you connect sensor, display, module, etc. There several library functions standard library functions, Ethernet library functions, library functions make the Arduino to connect internet.

With the Arduino Ethernet Shield, this library allows an Arduino board to connect to the internet. It can serve as either a server accepting incoming connections or a client making outgoing ones. The library supports up to four concurrent connection (incoming or outgoing or a combination).

4.4.1 ONE WIRE

OneWire communicates with 1-wire devices. The OneWireSlave library can be used to act as a 1-wire device.

OneWire requires a single 4.7K pullup resistor, connected between the pin and your power supply. When using very long wires, or with counterfeit DS18B20 chips and 3.3V power, a resistor in the 1K to 2.7K range may be required.

Then just connect each 1-wire device to the pin and ground. Some 1-wire devices can also connect to power, or get their power from the signal wire. Please refer to the specifications for the 1-wire devices you are using.

4.4.2 DALLAS TEMPERATURE

The DS18B20 is a one-wire digital temperature sensor which uses the Dallas 1-Wire protocol. This means that it only requires one data line and GND to communicate with the Arduino. The DS18B20 has a resolution of 9 bits, so it can measure temperatures between 0 and 255 degrees Celsius with a precision of 0.5 degrees Celsius. It also has an alarm function which can be used to trigger an event when a particular temperature is reached..

It can be a power supply for electronic devices that can be powered by an external power supply or by data line power (called "parasite mode"), which eliminates the need for an external power supply..

The DS18B20 is a 1-wire digital temperature sensor. It requires only a single wire to send data to a microcontroller, making it perfect for embedded applications. The sensor can measure temperatures from -55°C to +125°C with a resolution of 0.5°C. It also has an alarm function that can be used to signal when a certain temperature has been exceeded.

The DS18B20 can be operated in two different modes: normal mode and parasite mode. In normal mode, the sensor connects to 5V and GND. In parasite mode, the VDD pin should be connected to GND.

Wire multiple sensors to the same data wire using the Arduino platform. It also explains that each sensor has a unique serial code, which allows them to be individually identified. This is beneficial for reading temperature from multiple sensors using a single Arduino digital pin.

4.4.3 SOFTWARE SERIAL

The native serial support happens via a piece of hardware (built into the chip) called a UART. This hardware allows the Atmega chip to receive serial communication even while working on other tasks, as long as there room in the 64 byte serial buffer.The SoftwareSerial library has been developed to allow serial communication on other digital pins of the Arduino, using software to replicate the functionality (hence the name "SoftwareSerial"). It is possible to have multiple software serial ports with speeds up to 115200 bps. A parameter enables inverted signaling for devices which require that protocol.

4.4.4 NEW PING

A library that makes working with ultrasonic sensors easy. The author states that when he first received an ultrasonic sensor, he was not happy with how poorly it performed. He soon realized the problem was not the sensor, it was the available ping and ultrasonic libraries causing the problem. The NewPing library totally fixes these problems, adds many new features, and breathes new life into these very affordable distance sensors. This library is compatible with the avr, arm, megaavr, esp32 architectures.

CHAPTER-5

HARDWARE COMPONENTS USED

5.1 ARDUINO MEGA 2560 BOARD

The ATmega2560 is the basis for the Arduino Mega 2560 microcontroller board. It has 54 digital i/o pins, 4 UARTs (hardware serial ports), 16 analog ip's and a 16 MHz crystal oscillator.We need not do anything more to setup ,we just need to power it.The Mega 2560 is a supercomputer. It may be powered by USB or an AC-to-DC adapter and is compatible with the majority of Arduino Uno shields.

This also consists of 32KB of flash memory for the requirement of storage of code in which bootloader uses 20 of it. The Mega 2560 also has a number of built-in peripherals, including UARTs, ADCs, DACs, I2C host controller, SPI host controller.The arduino mega along with esp can be used to IOT.

Figure 5.1: Arduino Mega 2560(Source:Google)

5.2 LIQUID CRYSTAL DISPLAY

An LCD (Liquid Crystal Display) consists of liquid crystals between electrodes. The arrangement consists of polarization filters which are aligned perpendicular to each other. This arrangement doesn't allow any visible light if there was no liquid crystal between the filters. This arrangement is aligned in between transparent conductors. When sufficient voltage is applied to a certain pixel, the crystal at that pixel aligns such that no light passes through it. Therefore that particular pixel appears dark. If such an electric field is applied for a longer period,

the alignment of the crystal change and the quality of LCD degrades. In a bigger LCD display, to provide voltage sources to each pixel, the rows and column lines are multiplexed.

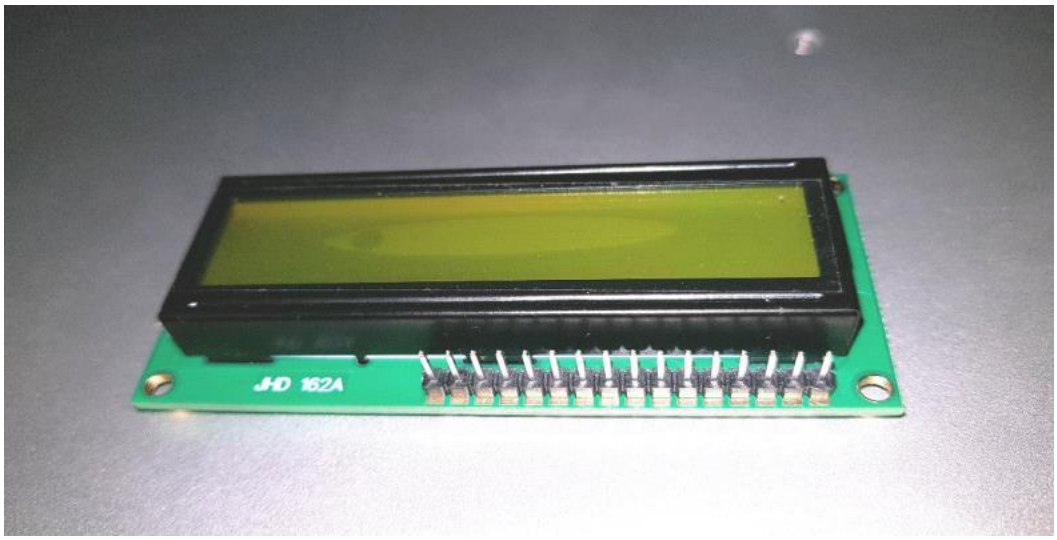


Figure 5.2.1 : LCD Display(Source:Google)

5.2.1 LCD Interfacing with Arduino:

To wire the LCD screen to your Arduino board connect the following pins

LCD RS pin to digital pin 12

Lcd enable pin to pin 11

D4 pin to digital pin 5

D5 pin to digital pin 4

D6 pin to digital pin 3

D7 pin to digital pin 2

A wire , 10k pot to 5V and GND, with its output to LCD screens V0 pin.A 220ohm resistor is used to power the backlight of the display, usually on pin 15 and 16 to LCD connector.

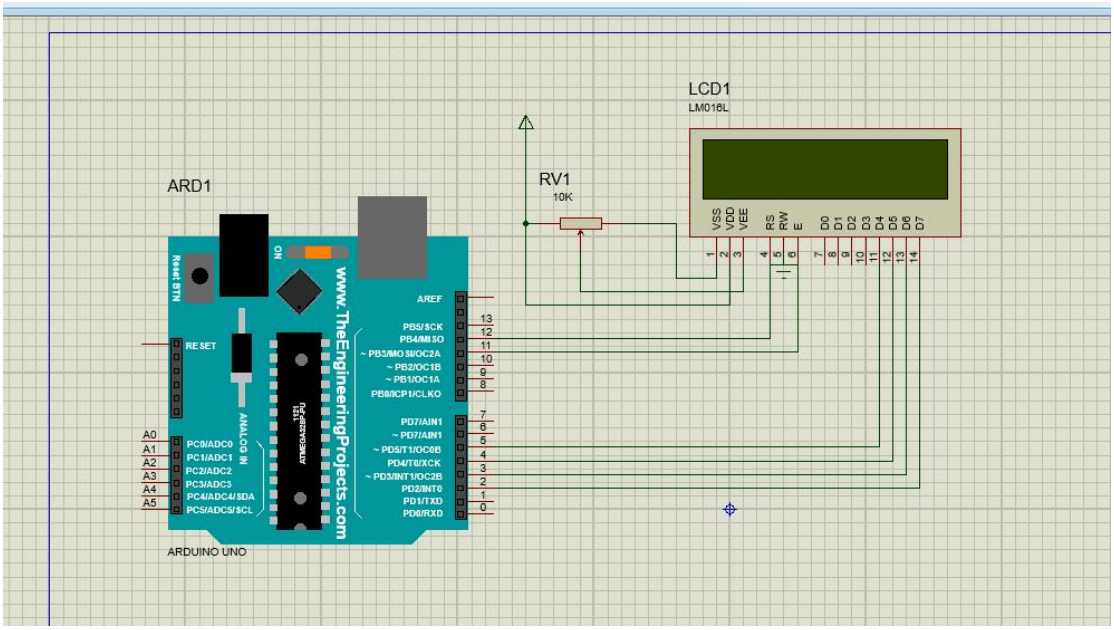


Figure 5.2.2:LCD interfacing with Arduino(Source:Google)

5.3 THE 2 IN ONE TEMPERATURE AND PH SENSOR

The Analog pH Sensor Kit is specially designed for Arduino controllers and has a built-in simple, convenient, and practical connection and features. It has an LED that works as the Power Indicator, a BNC connector, and a PH2.0 sensor interface.

To use it, just connect the pH sensor with the BND connector, and plug the PH2.0 interface into the analog input port of any Arduino controller. If pre-programmed, you will get the pH value easily. Comes in a compact plastic box with foams for better mobile storage.

Figure 5.3: 2 in one temprature and ph sensor(Source:Google)

5.4 THE TURBIDITY SENSOR

Turbidity is the cloudiness or haziness of a fluid caused by large numbers of individual particles that are generally invisible to the naked eye, similar to smoke in the air. The measurement of turbidity is a key test of water quality. Turbidity is caused by particles suspended or dissolved in water that scatter light making the water appear cloudy or murky. Particulate matter can include sediment, especially clay and silt, fine organic and inorganic matter, soluble colored organic compounds, algae, and other microscopic organisms.

Impact of Turbidity:

High turbidity can significantly reduce the aesthetic quality of lakes and streams. It can increase the cost of water treatment for drinking and food processing. It can harm fish and other aquatic life by reducing food supplies, degrading spawning beds, and affecting gill function.

Figure 5.4: Turbidity Sensor(Source:Google)

5.5 A GSM SHIELD

Using the GPRS wireless network, the Arduino GSM Shield allows an Arduino board to connect to the internet. It's simple to use: just plug it into an Arduino board, insert a SIM card from a GPRS-enabled carrier, and follow the on-screen instructions. You can also use the on-board audio/mic jack to make and receive voice calls, as well as send and receive SMS messages and notifications.

Using an Arduino and the AT command, you may communicate with the SIM900A module. To control the module, the Arduino should send an AT instruction to it. The AT command of the SIM900A communicates via the serial port. Serial requires two pins: the transmitter (Tx) and the receiver (Rx) (Rx). A command that starts with the letter "AT" is known as the AT command.

Example of an AT Command:

AT+CMGF=1;/ It was traditionally used to put the GSM Module into Text Mode.

/ AT+CMGS

It was traditionally used to send a message.

Figure 5.5: GSM SHIELD(Source:Google)

5.6 AN ULTRASONIC SENSOR

Ultrasonic transducers and **ultrasonic sensors** are devices that generate or sense ultrasound energy. They can be divided into three broad categories: transmitters, receivers and transceivers. Transmitters convert electrical signals into ultrasound, receivers convert ultrasound into electrical signals, and transceivers can both transmit and receive ultrasound.

Ultrasound can be used for measuring wind speed and direction (anemometer), tank or channel fluid level, and speed through air or water. For measuring speed or direction, a device uses multiple detectors and calculates the speed from the relative distances to particulates in the air or water. To measure tank or channel liquid level, and also sea level (tide gauge), the sensor measures the distance (ranging) to the surface of the fluid. Further applications include: humidifiers, sonar, medical ultrasonography, burglar alarms, non-destructive testing and wireless charging.

Systems typically use a transducer that generates sound waves in the ultrasonic range, above 18 kHz, by turning electrical energy into sound, then upon

receiving the echo turn the sound waves into electrical energy which can be measured and displayed.

This technology, as well, can detect approaching objects and track their positions.

Figure 5.6:Ultrasonic sensor(Source:Google)

5.7 4 RGB LED

An RGB LED is a combination of 3 LEDs in just one package:

- 1x Red LED
- 1x Green LED
- 1x Blue LED

You can produce almost any color by combining those three colors. An RGB LED is shown in the following figure:

Figure 5.7:RGB LED(Source:Google)

To produce other colors, you can combine the three colors in different intensities. To adjust the intensity of each LED you can use a PWM signal.

Because the LEDs are very close to each other, our eyes see the result of the combination of colors, rather than the three colors individually.

To have an idea on how to combine the colors, take a look at the following chart. This is the simplest color mixing chart, but gives you an idea how it works and how to produce different colors.

5.8 NODE MCU 2296

The NodeMCU (Node MicroController Unit) is an open source software and hardware development environment that is built around a very inexpensive System-on-a-Chip (SoC) called the ESP8266. The ESP8266, designed and manufactured by Espressif Systems, contains all crucial elements of the modern computer: CPU, RAM, networking (wifi), and even a modern operating system and SDK. When purchased at bulk, the ESP8266 chip costs less.That makes it an excellent choice for IoT projects of all kinds.

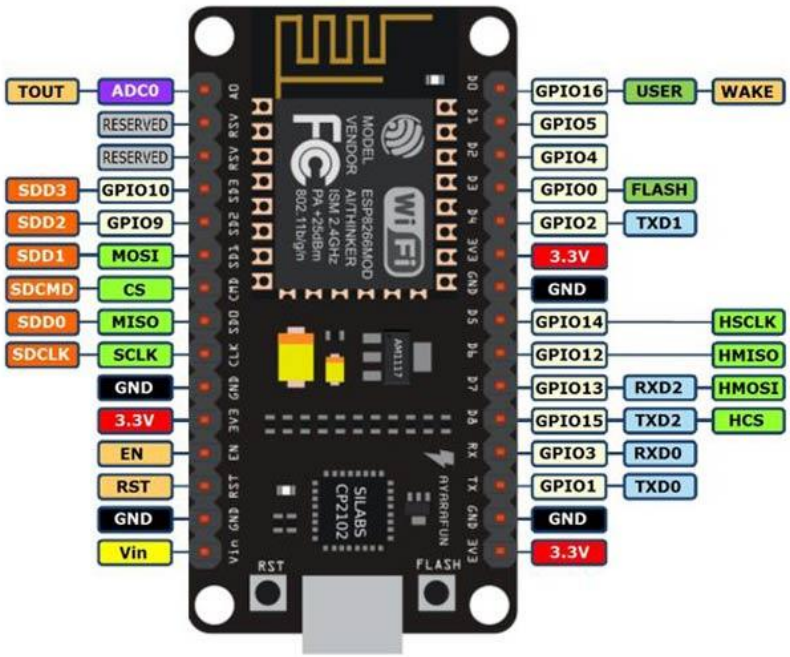


Figure 5.8 :Node MCU(Source:Google)

Through its pins we can read inputs - light on a sensor, a finger on a button, or a Twitter message -and turn them into an output - activating a motor, turning on an

LED, publishing something online. It has also WiFi capabilities, so we can control it wirelessly and make it work on a remote installation easily! We can tell our board what to do by sending a set of instructions to the microcontroller on the board. To do so we can use the the Arduino Software (IDE).

5.9 BUZZER

The piezo buzzer produces sound based on reverse of the piezoelectric effect. The generation of pressure variation or strain by the application of electric potential across a piezoelectric material is the underlying principle. These buzzers can be used to alert a user of an event corresponding to a switching action, counter signal or sensor input. They are also used in alarm circuits. The buzzer produces a same noisy sound irrespective of the voltage variation applied to it. It consists of piezo crystals between two conductors. When a potential is applied across these crystals, they push on one conductor and pull on the other. This, push and pull action, results in a sound wave. Most buzzers produce sound in the range of 2 to 4 kHz.

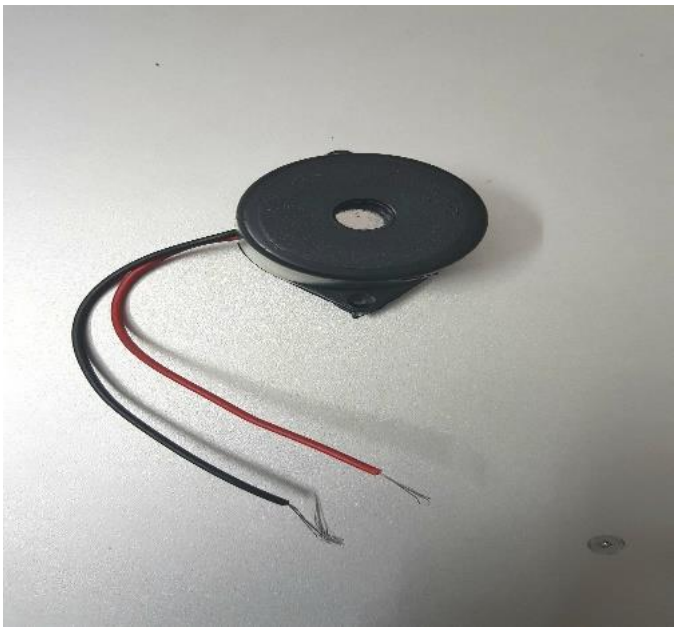


Figure 5.9:Buzzer(Source:Google)

CHAPTER-6

ARDUINO IDE SOFTWARE

6.1 ARDUINO IDE

The Arduino Integrated Development Environment or Arduino Software (IDE) contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the arduino hardware to upload programs and communicate with them.

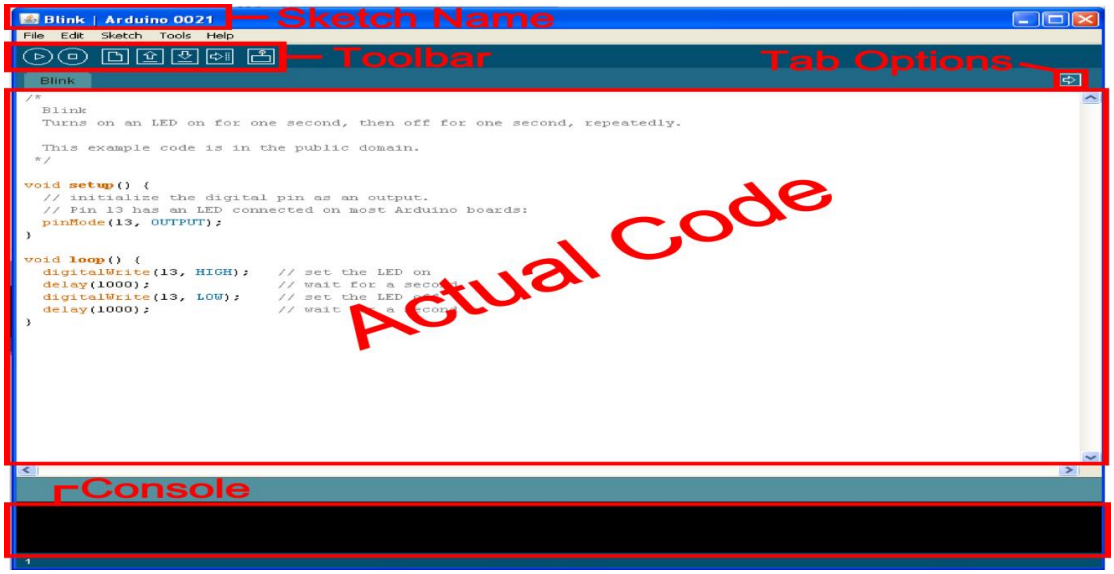


Figure 6.1:Arduino Environment

6.2 WRITING SKETCHES

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in text editor and the editor has features for cutting, pasting, searching, replacing text. The message area gives feedback while saving and uploading and also displays errors. The console displays text output by Arduino Software (IDE), including complete error messages and other information. The toolbar buttons allows to verify and upload programs, create, open, and save sketches, and open the serial monitor.

6.3 STEPS TO DUMP THE PROGRAM INTO THE ARDUINO UNO BOARD

1. Select the Arduino UNO board from **tools>board>Arduino UNO**.

2. Select the serial port to communicate **tools>port>serial**.

3. The sketches(code) in your sketchbook can be opened from the **File>sketchbook** or create a new file and write the program**.**

4. Checks your code for errors compiling it using **verify**.

5. Save your program/sketch using **save**.

6. Upload the sketch to Arduino using **upload**.

7. Open the serial monitor incase of any outputs to be displayed on it .

Serial monitor.

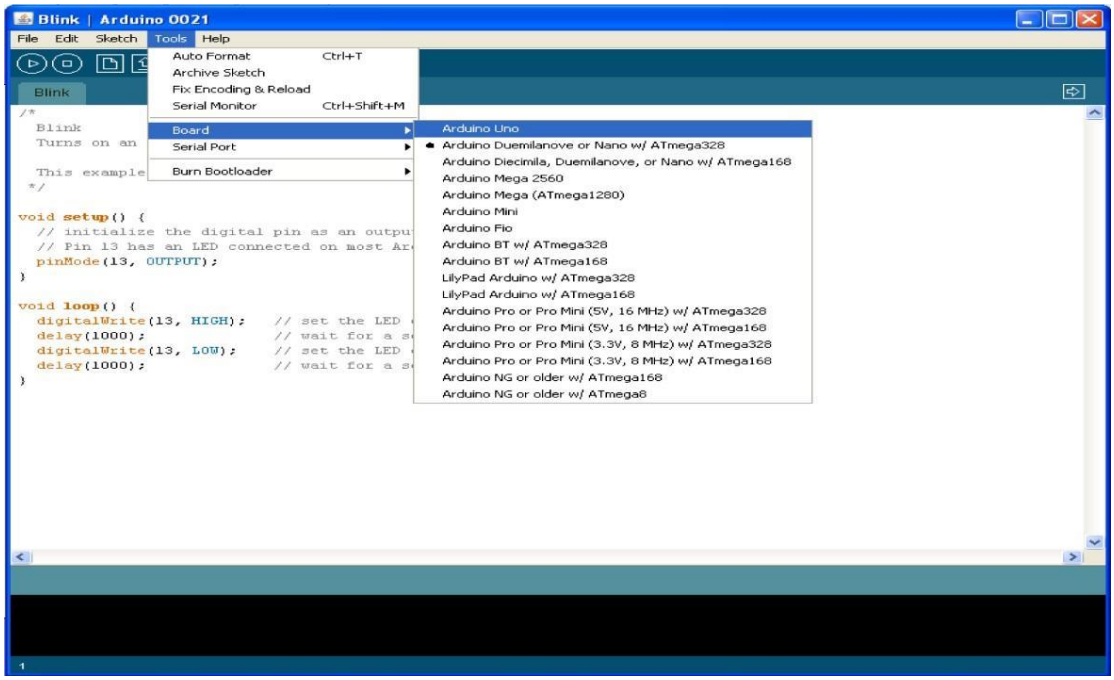


Figure 6.3.1: Selecting The Required Boards

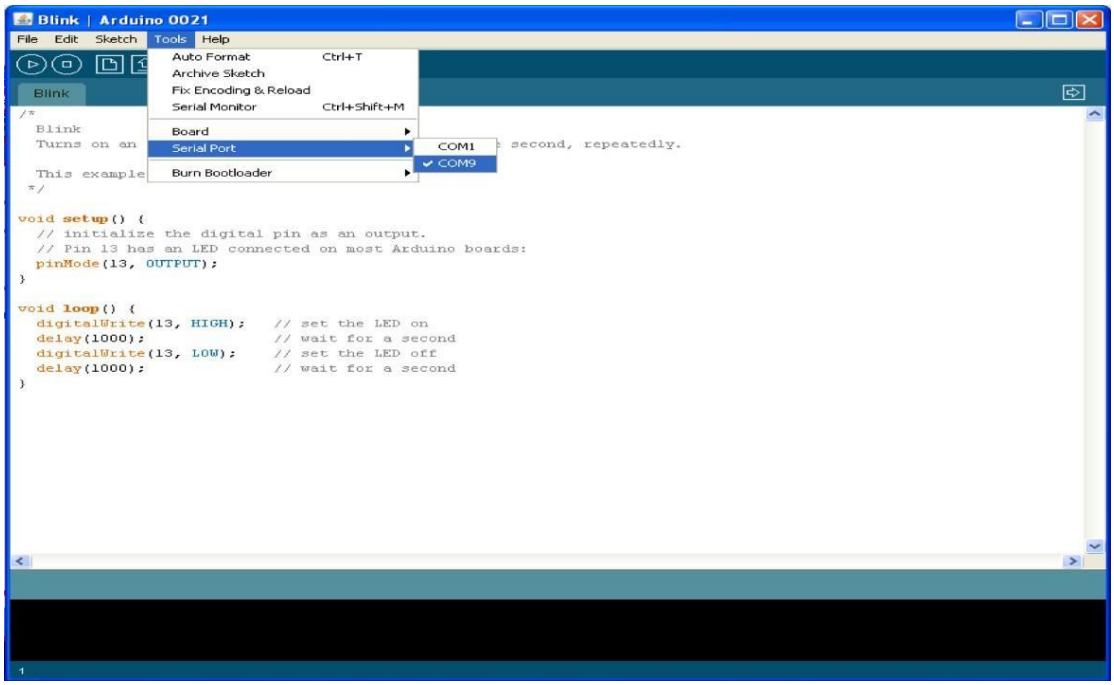


Figure 6.3.2:Selecting the serial port

Figure 6.3.3:Uploading the sketch to the Arduino board

6.4 ADVANTAGES

1. Additional commands are found within the five menus: File, Edit, Sketch, Tools, Help.
2. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.
3. Uses high speed serial communication.
4. Allows to manage sketches with more than one file. These can be normal Arduino code files(no visible extension), c files(.c extension), c++ files(.cpp extension), or header files (.h)
5. Library functions can be uploaded to board
6. Displays an error message in case of any mistakes in the sketch

CHAPTER-7

RESULTS

projecct_code | Arduino 1.8.16

File Edit Sketch Tools Help

projecct_code

```
#include <OneWire.h>
#include <DallasTemperature.h>
#include <SoftwareSerial.h>
#include <NewPing.h>
#define SensorPin A2            //pH meter Analog output to Arduino Analog Input 0
#define Offset 0.00            //deviation compensate
unsigned long int avgValue;     //Store the average value of the sensor feedback

#define TRIGGER_PIN  23  // Arduino pin tied to trigger pin on ping sensor.
#define ECHO_PIN     22  // Arduino pin tied to echo pin on ping sensor.
#define MAX_DISTANCE 200 // Maximum distance we want to ping for (in centimeters). Maximum sensor distance is rated at 400-500cm.

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE); // NewPing setup of pins and maximum distance.

unsigned int pingSpeed = 50; // How frequently are we going to send out a ping (in milliseconds). 50ms would be 20 times a second.
unsigned long pingTimer;     // Holds the next ping time.

// Data wire is plugged into pin 2 on the Arduino
#define ONE_WIRE_BUS 6

SoftwareSerial mySerial(7, 8);

// Setup a oneWire instance to communicate with any OneWire devices (not just Maxim/Dallas temperature ICs)
OneWire oneWire(ONE_WIRE_BUS);

// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);
```

Done compiling.

```
Sketch uses 20346 bytes (8%) of program storage space. Maximum is 253952 bytes.
Global variables use 1638 bytes (19%) of dynamic memory, leaving 6554 bytes for local variables. Maximum is 8192 bytes.
```

1 Arduino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM7

Figure:7.1 Arduino Code

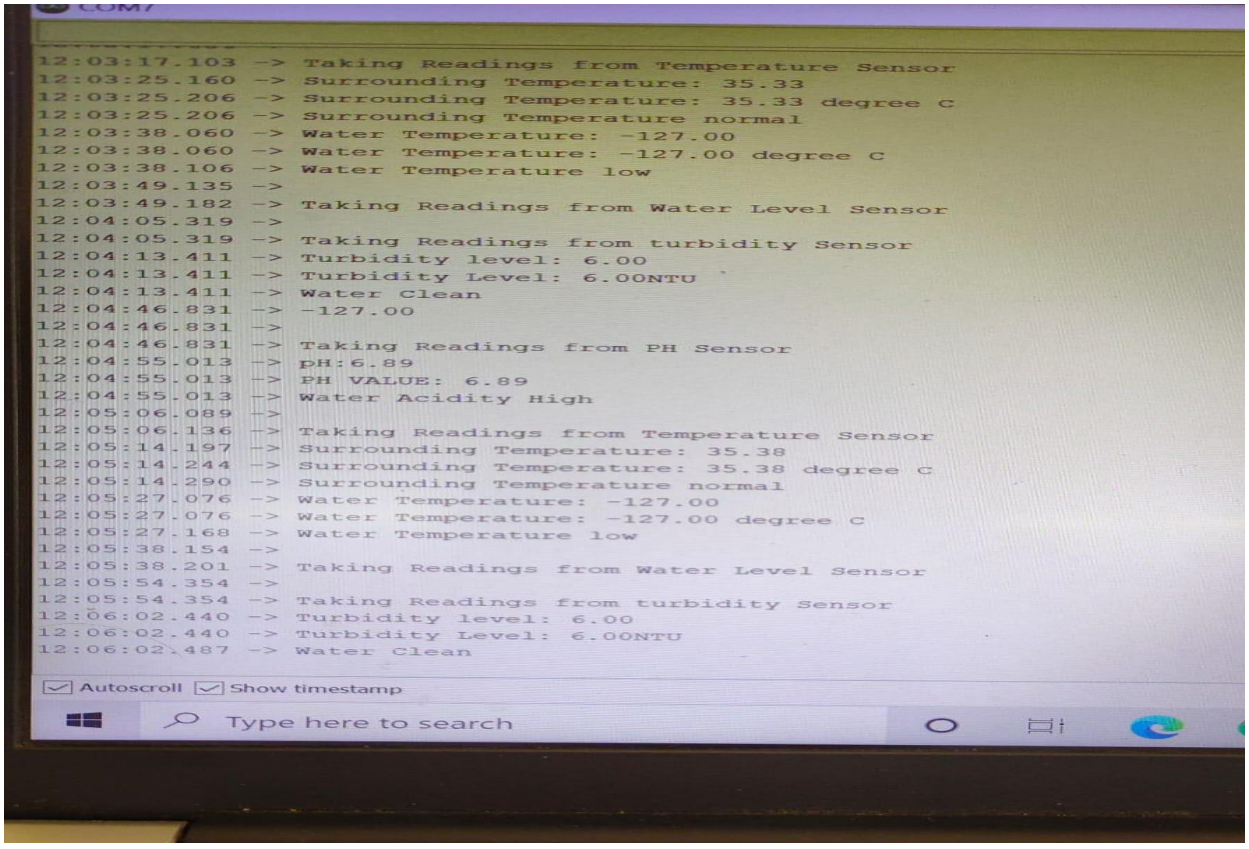


Figure: 7.2 Serial Monitor Output

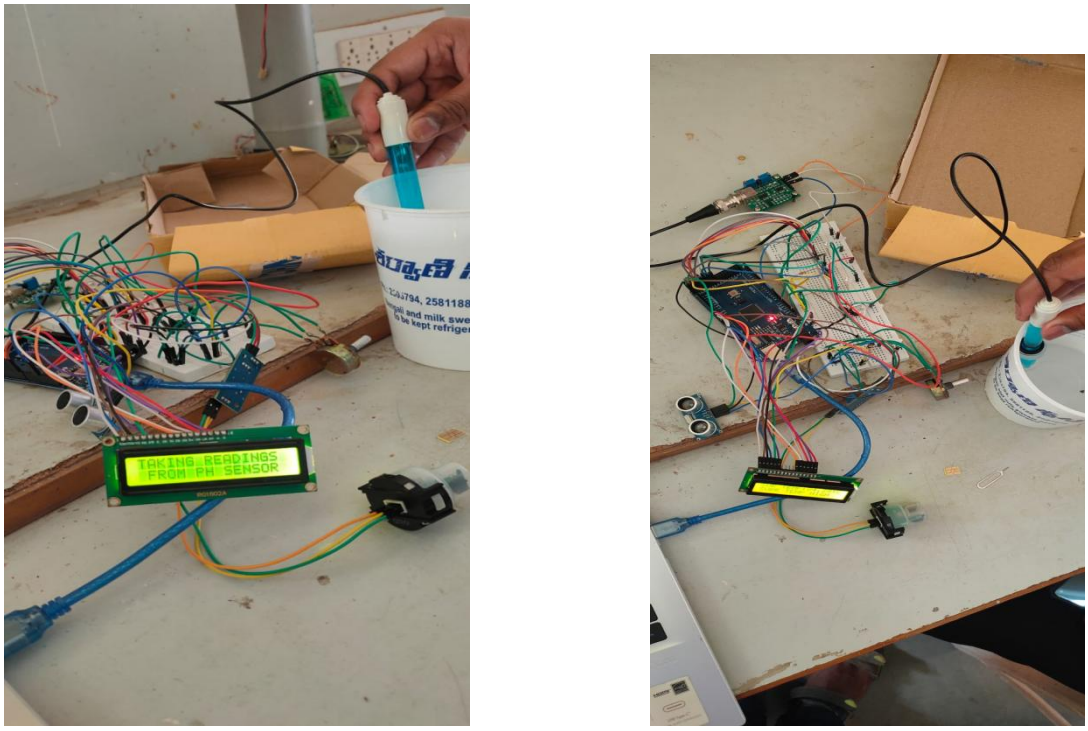


Figure: 7.3 Working Model

Figure:7.4 Outputs

Figure:7.5 Blink outputs

CHAPTER-8

CONCLUSION AND FUTURE SCOPE

Conclusion:

Because water is such a valuable resource, it is critical to keep track of it in order to maintain a healthy lifestyle.

This problem arose due to the scarce water resources, increasing population, and aging infrastructure. As a result, better strategies for observing of state of water and traditional methods of water characterization are required.

Although current approaches analyze physical, chemical, and biological agents, they have significant flaws, including inadequate spatiotemporal coverage, labor-intensive and high costs (people, operation, and equipment), and a lack of real-time water quality data to enable crucial public health choices.

As a result, constant water quality monitoring is required, necessitating the use of this model.

Future Scope :

- We can provide the world with safe water through constant monitoring of its quality.
- We can always keep on eye on level water is present in lakes and rivers which are main sources of our drinking water.
- We can also use “Internet of Things” to let everyone acknowledge about the water quality and level.

REFERENCES

[1] Dongling Ma and Jian Cui, "Design and realization of water quality information management system based on GIS," *2011 International Symposium on Water Resource and Environmental Protection*, 2011, pp. 775-778, doi: 10.1109/ISWREP.2011.5893122.

[2] Q. BAI, J. Wu and C. JIN, "The Water Quality Online Monitoring System Based on Wireless Sensor Network," *2020 13th International Symposium on Computational Intelligence and Design (ISCID)*, 2020, pp. 234-237, doi: 10.1109/ISCID51228.2020.00059..

[3] C. Zhang, J. Wu and J. Liu, "Water quality monitoring system based on Internet of Things," *2020 3rd International Conference on Electron Device and Mechanical Engineering (ICEDME)*, 2020, pp. 727-730, doi: 10.1109/ICEDME50972.2020.00171.

[4] Z. Lin, W. Wang, H. Yin, S. Jiang, G. Jiao and J. Yu, "Design of Monitoring System for Rural Drinking Water Source Based on WSN," *2017 International Conference on Computer Network, Electronic and Automation (ICCNEA)*, 2017, pp. 289-293, doi: 10.1109/ICCNEA.2017.106.

[5] R. P. N. Budiarti, A. Tjahjono, M. Hariadi and M. H. Purnomo, "Development of IoT for Automated Water Quality Monitoring System," *2019 International Conference on Computer Science, Information Technology, and Electrical Engineering (ICOMITEE)*, 2019, pp. 211-216, doi: 10.1109/ICOMITEE.2019.8920900.

[6] D. Sekhwela, P. A. Owolawi, T. Mapayi and K. Odeyemi, "Water Quality Monitoring with Notifications System," *2020 2nd International Multidisciplinary Information Technology and Engineering Conference (IMITEC)*, 2020, pp. 1-6, doi: 10.1109/IMITEC50163.2020.933409