CHAPTER 1

1. INTRODUCTION

The growing industry advancement and world population industry advancement, environmental pollution became big concern. Systems for air and water quality monitoring are required for activity analysis and their impact on nature of the power plants, mining sector, oil and gas industry, etc. Basically, determination of water quality relies on estimation of values of some important and indicative parameters. For example, the water quality monitoring demands the determination of parameters like PH, dissolved oxygen, content of ammonia, conductivity, turbidity, temperature, dissolved metal ions, etc. Although there are well known and widely used methods for measurement of these parameters with appropriate Sensors, design of electronic systems for environmental monitoring is not often straightforward. The engineering challenges are various: (a) sensor nodes are usually deployed in remote places, (b) long-term deployments require sensor nodes to be robust and systems to be easily reconfigurable, (c) sensor nodes have to be able to operate autonomously in the required environment, etc. Moreover, such applications require highly reliable and accurate sensors with the reduced level of maintenance, long lifetime, fast response times, high sensitivity and high selectivity. With the introduction of IoT in the modern world, many problems have been solved. With the use of IoT in monitoring water and air quality, various issues such as data collection, communication, data analysis and early warnings are worked on. But in order to get this into picture, technologies and protocols are combined to get the desired output.

1.1 EMBEDDED SYSTEM

As the name signifies, an embedded system is embedded or builds into something else. Embedded systems encompass a variety of hardware and software components, which perform specific functions in host system, for example satellites, washing machine, hand held telephones and automobiles.

A few years ago, embedded technology existed in standalone devices such as vending machines and copiers that did their jobs with little regards for what wanton around them. But as technology advance to connect devices to the internet and to each other, the potential of embedded technology has increased. Home appliances, mobile phones, cars, avionics etc.., are all using embedded technology. With the continuous economic growth, the water demand of enterprise sisal so increasing. The monitoring of water resource for these enterprises can prevent the occurrence of stealing water and leaking water effectively. Therefore, the monitoring system of urban water supply has aroused extensive attention in recent years. Urban water supply networks form the link between drinking water supply and drinking water consumers. These large-scale networks are vital for the survival of urban life, for maintaining a healthy level of economic development, and for them continuous operation of factories and hospitals.

The With rapid development of global system mobile infrastructure and information communication technology in the past few decades has made the communication is reliable for transmitting and receiving information efficiently. So here we used GSM modem for efficient communication purpose. A design is based on RF and Zigbee which has some shortcomings, such as high-power consumption, near distance and network's size is small. So, GSM is chosen in this project.

Embedded systems have become increasing -digital peripheral (analog power) and therefore both Hardware and software coding signs are relevant. Most Embedded control product must special requirement, cost effectiveness, low power, small-foot print and a high level of system integration.

Typically, most embedded control systems are control system, that are designed around an MCU, which integrates on-chip program memory, data memory (RAM) and various peripheral functions, such as timer and serial communication. In addition, these systems usually require analog/interface devices, serial EEPROM display driver's keypads or small displays.

1.2 EMBEDDED SOFTWARE

Software in the embedded system is implanted with either assembly language or any high-level language. Now-a-days C and C++ has been the choice but language for the embedded software for the following reasons and C++ are machine independent language, so the programmer can concentrate only on the algorithms has the ability for direct hardware control and it can be interfaced to run any mechanical machine.

Any source code written in can C++ or assembly must be converted into an executable image that can be loaded onto an EEPROM chip. The process of converting the source code representation of embedded software into an executable image involves three distinct steps and the system or computer on which these processes are executed is called host computer.

There are some differences between conventional programming and embedded programming. Even if the processor architecture is the same, the I/O interfaces or sensors or activators may differ.

The Embedded system software comprise of building program that will run on the host. These tools are called Native tools. Some of them are:

1.2.1 CROSS-COMPILIERS

The compiler that runs on the host system and produces the binary instructions that will be understood by the target microprocessor is called Cross-Compiler.

1.2.2 CROSS-ASSEMBLER

A cross assembler is the assembler that run on the host and produces binary instructions appropriate for the target system.

1.2.3 LINKER/LOCATOR

Linker/Locator links all the object files produced by the cross compiler and assembler. The loader finds memory into load the program from the disk into the memory and may then do various other processing before starting the program. The basic tool chain for the Embedded Software is given below.

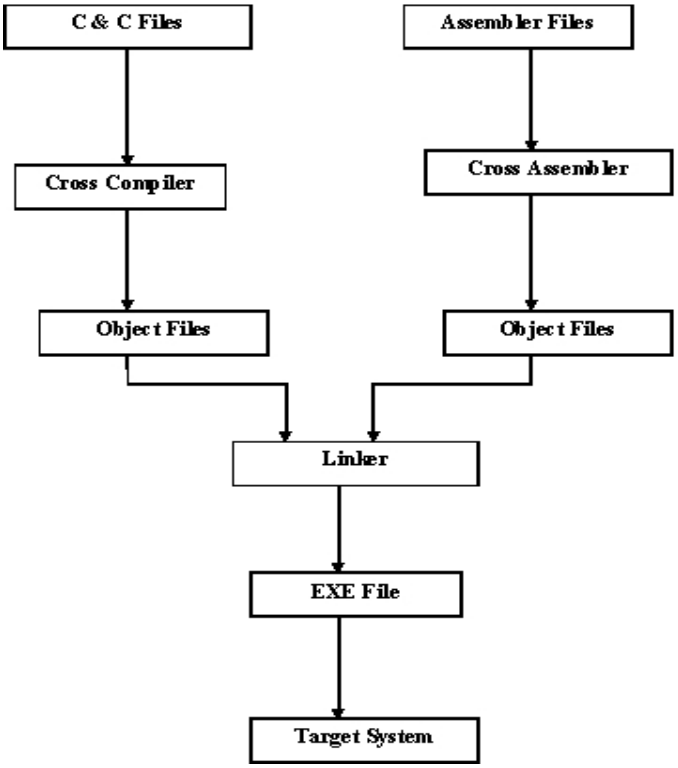


Fig.1.1 Link/Locater

1.3 FEATURES OF EMBEDDED SYSTEM

1. Higher performance
2. Low power consumption
3. Slimmer and more compact
4. Mission-critical functions
5. Reduced design and development time
6. Lower system cost.

1.4 APPLICATION AREAS OF EMBEDDED SYSTEM

Embedded software is present in almost every electronic device we use today. There is embedded software inside the watch, cell phones, automobiles, industries, industrial control equipment and scientific medical equipment. Defence service use embedded software to guide missiles and detect enemy air facts. Communication satellites, medical instruments and deep space probes would have been nearly impossible without these systems. Embedded system covers such a broad range of products, that generalization is difficult. Here are some broad categories:

- Aerospace and Defence electronics
- Automotive
- Broad cast and Entertainment
- Internet appliance
- Data communication

Embedded systems have a vast variety of application domains that varies from low cost to high, consumer electronics to industrial equipment's, entertainment devices to academic equipment's and medical instruments to weapons and aerospace control systems. The applications of embedded systems include home appliances, office automation, security, telecommunication, instrumentation, entertainment, aerospace, banking and finance, automobiles personal and in different embedded systems projects.

1.5 MOTIVATION

Water contamination and scarcity is a global issue, which involves a continuous reform of the concept of directing water supplies to individual wells globally. Water contamination has been listed as the global leader in diseases. More than 10,000 people are killed every day around the world, according to reports. Every day, more than500 people in India die from water contamination problems. Studies have shown that the amount of usable water declines to a minimum level after many years. Dirty or dirty water is used to drink in many developing countries without sufficient use. One explanation is the public and institutional acknowledge and the lack of a monitoring network of water quality, which causes significant global problems. The consistency and ecological status of water often alter environmental impacts such as volcanoes, algae tints and earthquakes.

1.6 OBJECTIVE

To attain our project system, need to use Arduino, controller to monitor the field. In this water level sensor is used to monitor the field environment. If any abnormalities mean indicate the buzzer. Water level sensor used to identify the water level amp; PH sensor and Flow sensor is used to water leakage and quality check. If its level is low means pump motor will be ON & indicated by the single beep sound, otherwise if its level is high means pump motor will be OFF amp; indicated by the long beep sound. Controller status and everything is indicated by LCD and notification will send through IoT. The whole process is controlled by microcontroller.

CHAPTER 2

2. LITERATURE SURVEY

2.1 Automatic Water Level Indicator And Controller Using Arduino

Most of the people in residential areas face the problem of running out of water and overflow of water in water tanks due to excess supply of water. It becomes difficult for users to judge the level of water in water tanks. When the pump is turned ON, users will not realize that the water tank is filled, which may result in overflow. Water level indicator and controller system is used to sort out the issues associated with water tank. It is also possible to check the level of the water using sensor so that whenever the water goes below, pump gets turned ON automatically. Also, when there is overflow of water in water tank it uses sensor to detect the water level so that if the water level goes above, the pump gets turned off automatically. This system prevents wastage of water.

2.2 Iot Based Water Quality Monitoring System By Using Zigbee Protocol

This paper dictates the damages caused by water and what can be done to resolve those issues by involving the Internet of things (IoT). Keeping the quality of water in check is today's ultimate objective. Thereby, to guarantee safe drinking water supply, the quality of water should be observed regularly. The use of IoT based solution, focused mainly on water quality monitoring has therefore been suggested. In order to support the issue, an IoT-based water quality checking network has been introduced that continuously monitors and evaluates the quality of water and tries to distinguish whether it is up to the mark for general use. This paper includes the use of specific sensors that calculates the various parameters of the quality of water which includes conductivity and dissolved oxygen (DO), turbidity, pH, and temperature.

2.3 Online Checking System for Drinking Quality of Drinking Water Vending Machine

Online Checking System for Drinking Quality of Drinking Water Vending Machine aimed to Study and design checking system for drinking quality of drinking water vending machine and Develop drinking water vending machines to have the standard and safety in drinking. The system can be checking with 5 parameters such as Turbidity, Conductivity, Total Dissolved Solids (TDS), pH and temperature of the water. The system was designed in the automation system by a controller included GSM module 3G Shield (UC20-G) in sending data to Firebase for creating a database in Real-time. The system can be noticed about water quality in real-time on the mobile phone, website, station on Google map and Time of maintenance in changing the water filter. The results of this research show that the system and sensors for checking the quality of water are good results in working. It can be easy to use and monitor in the control quality of drinking water.

2.4 Water Quality Monitoring System through IoT

Out of the five-utility water is one important factor. Current edge due to globalization, water pollution has become the threat. The quality of the water should be monitored for the drinking purpose. The article proposes a new approach for water monitoring through Internet of Things (IoT). The IoT system again constitutes several sensors to measure the parameters of drinking water. These parameters can be the PH value, temperature, and density to be measured. These parameters can be processed further by the core controller. Here, Arduino model is used as the core controller. The data can be viewed by the sensors.

2.5 Water quality monitoring and Water pipe Leakage Detection

Water quality monitoring is one of the important aspects which are used to save people from diseases as water is one of the basic resources for human beings. This paper discusses about an IoT application, developed to check quality of water, water level detection and water pipe leakage using IoT technology. Water quality can be measured by using Temperature, pH and Turbidity sensors. In this application these three aspects are considered for quality measurement. Nowadays water is pumped from ground storage to fill the overhead tank. It becomes a hectic task to always go and check the level of water in the tank. This may sometimes lead to wastage of water due to and even consumes more electricity. This issue can be solved by indicating water level in the tank and sending notification to the specified person. Water Pipe Leakage is another important issue to avoid wastage of water. There are many techniques to detect water pipe leakage. In this paper we implemented a new technique based on rate of flow of water. When leakage is detected, a notification is sent to concerned person that helps in saving water and electrical consumption. Results are published in cloud. Data is Analysed using R tool.

CHAPTER 3

3. SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

In this project as per our reference there are no projects with combination of Water quality and water leakage finding, as technical side. For water leakage no sensors are used to find the leakage its has-been identified by manual for water quality checking also GE filtration method is used so for which implemented in RO system maintains cost and manual interpret in required.

3.2 PROPOSED SYSTEM

The proposed system is establishing a technique to avoid drawbacks of existing system. The proposed system also monitors, water quality leakage, contamination at the pipelines between distribution tank and the end user. All application will receive to mobile notification through GSM.

3.3 SYSTEM REQUIREMENTS

3.3.1 HARDWARE REQUIREMENTS

- Power supply
- Ultrasonic sensor
- PH sensor
- Flow sensor
- Arduino microcontroller
- Relay
- Gsm module
- Pump Motor

3.3.2 SOFTWARE REQUIREMENTS

- Arduino IDE
- Embedded c language

3.3.3 BLOCK DIAGRAM

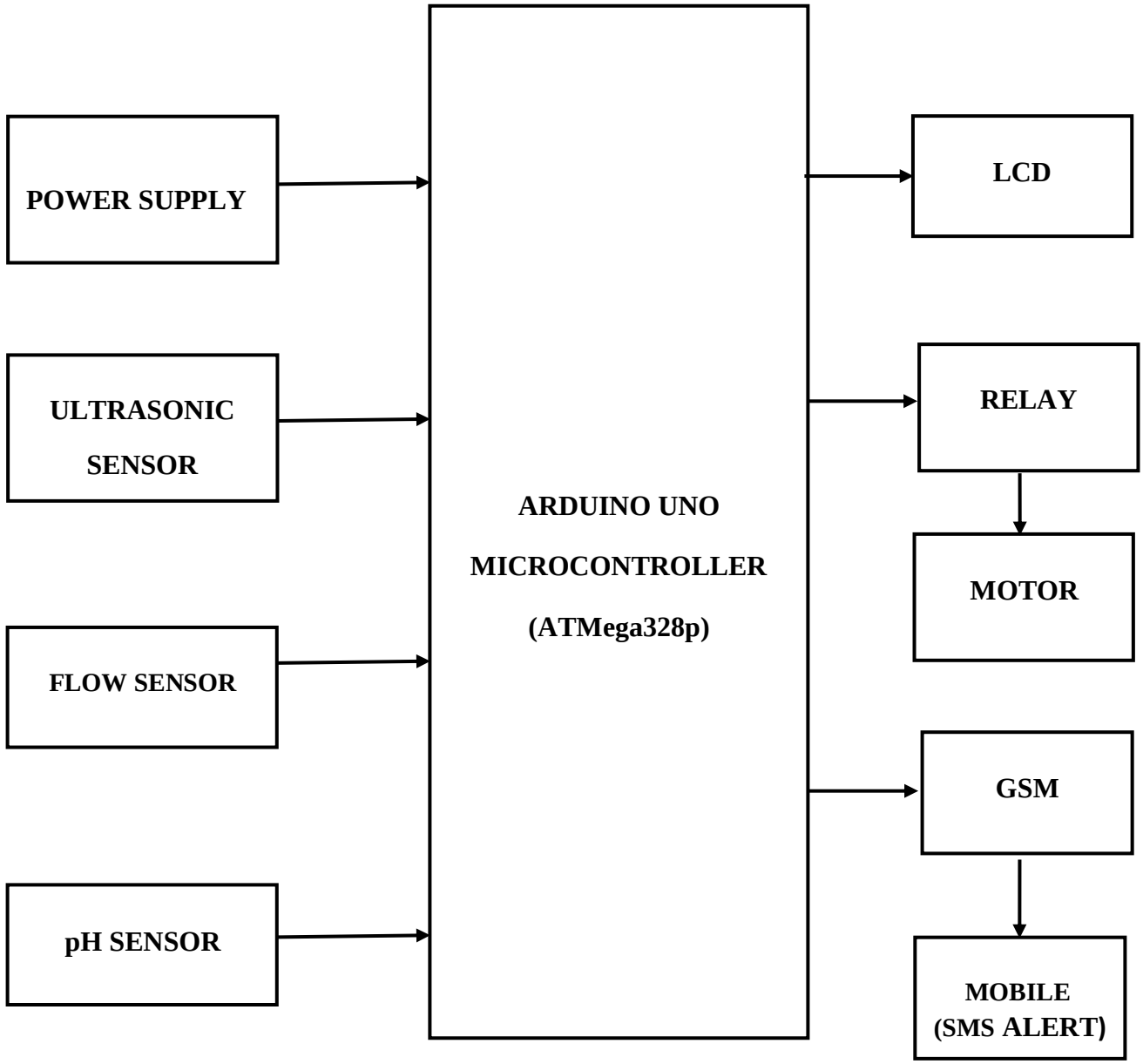


Fig 3.1 Block diagram

3.3.4 CIRCUIT DIAGRAM

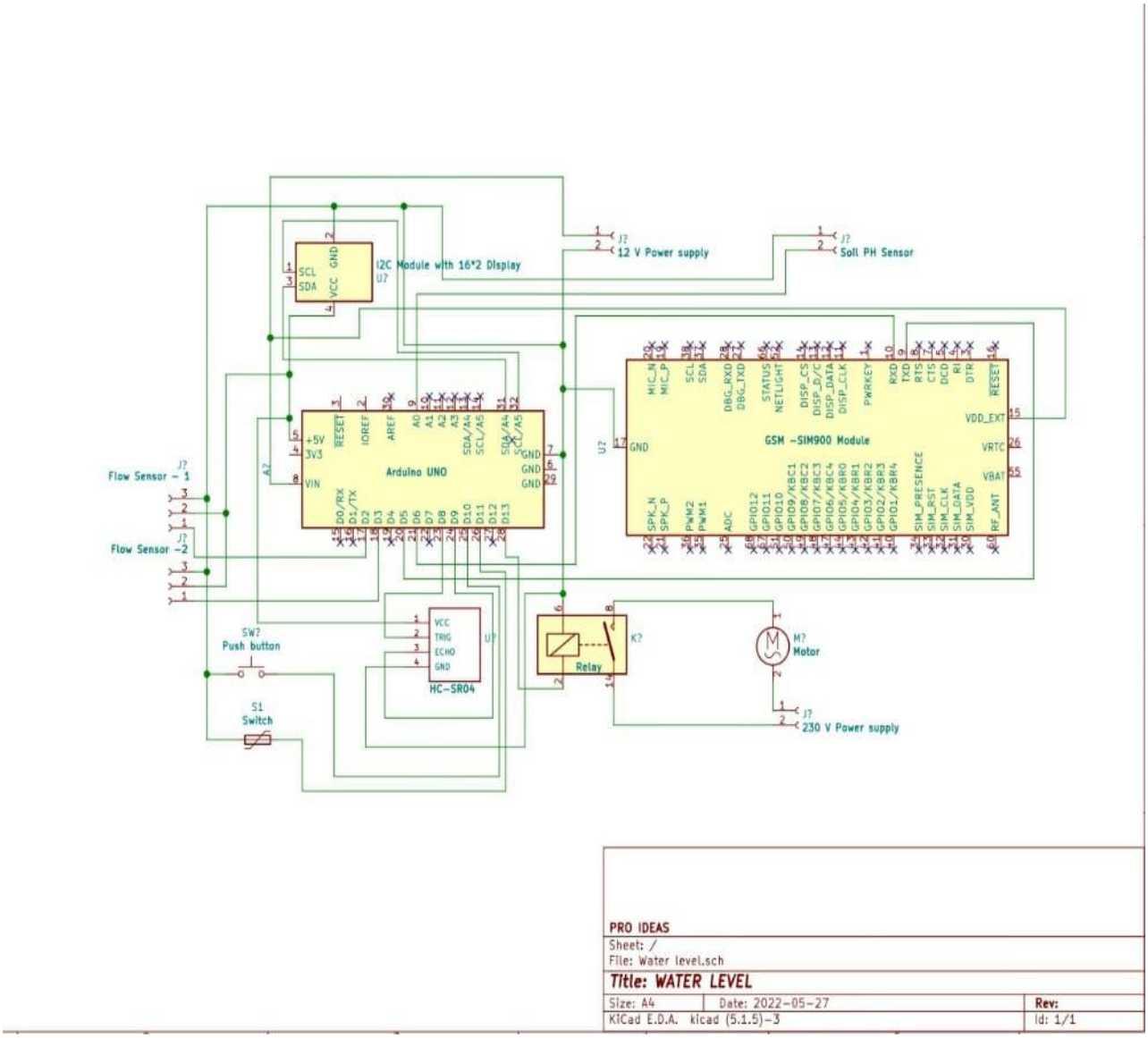


Fig 3.2 Circuit Diagram

CHAPTER 4

4. HARDWARE DESCRIPTION

4.1 ARDUINO UNO CONTROLLER

The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins of which six can be used as PWM outputs, six analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started. The Arduino Uno differs from all preceding boards because it does not use the FTDI USB-to-serial driver chip. Instead, it features the ATmega8U2 programmed as a USB-to-serial converter. Revision 2 of the Arduino Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode.

Uno means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or out-dated boards see the Arduino index of boards.

This is the Arduino Uno R3. In addition to all the features of the previous board, the Uno now uses an ATmega16U2 instead of the 8U2 found on the Uno or the FTDI found on previous generations. This allows for faster transfer rates and more memory.

The Arduino Uno is a microcontroller board based on the ATmega328. Arduino is an open-source, prototyping platform and its simplicity makes it ideal for hobbyists to use as well as professionals. The Arduino Uno has 14 digital input/output 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

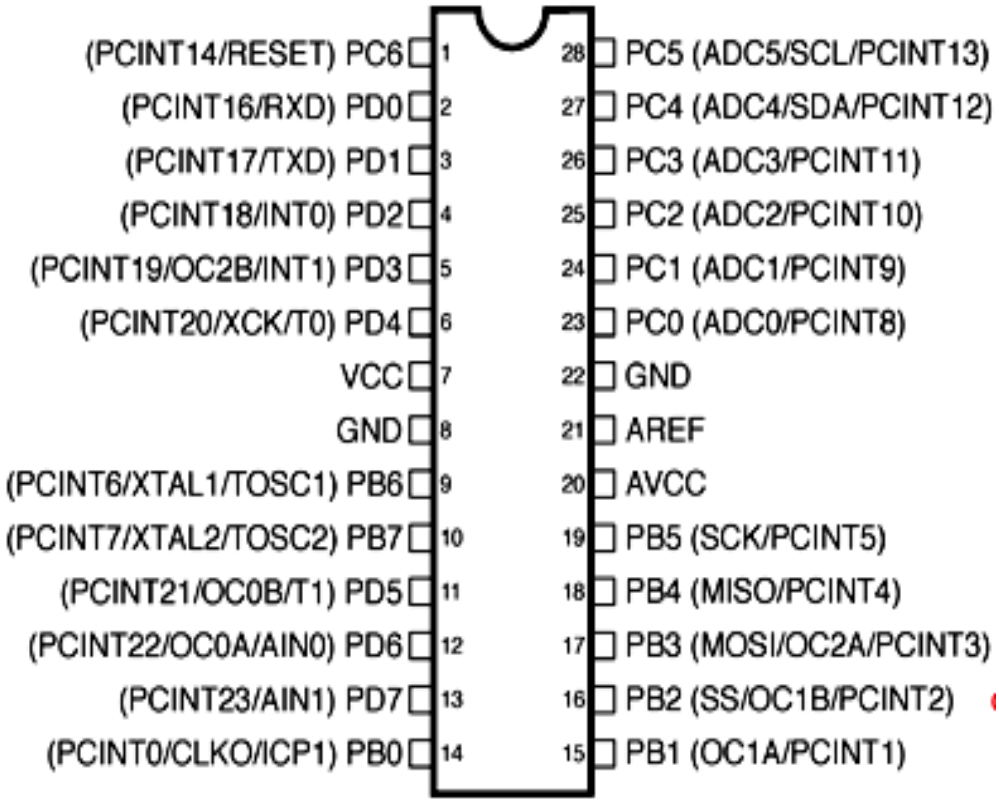


Fig 4.1 Arduino Uno Pin Diagram

Pin Category	**Pin Name**	**Details**
Power	Vin, 3.3V, 5V, GND	Vin: Input voltage to Arduino when using an external power source. 5V: Regulated power supply used to power microcontroller and other components on the board. 3.3V: 3.3V supply generated by on-board voltage regulator. Maximum current draw is 50mA. GND: ground pins.
Reset	Reset	Resets the microcontroller.
Analog Pins	A0 – A5	Used to provide analog input in the range of 0-5V
Input/output Pins	Digital Pins 0 - 13	Can be used as input or output pins.
Serial	0(Rx), 1(Tx)	Used to receive and transmit TTL serial data.
External Interrupts	2, 3	To trigger an interrupt.

PWM	3, 5, 6, 9, 11	Provides 8-bit PWM output.
SPI	10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK)	Used for SPI communication.
Inbuilt LED	13	To turn on the inbuilt LED.
TWI	A4 (SDA), A5 (SCA)	Used for TWI communication.
AREF	AREF	To provide reference voltage for input voltage.

Table 4.1 Arduino Pinout Configuration

4.1.1 COMMUNICATION

Arduino/Genuino Uno has a number of facilities for communicating with a computer, another Arduino/Genuino board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .in file is required.

4.1.2 PROGRAMMING ARDUINO

Capacitors are used as filters in the power supply unit. The action of the system depends upon the fact, that the capacitors stores energy during the conduction period and delivers this energy to the load during the inverse or non-conducting period. In this way, time during which the current passes through the load are prolonged and ripple is considerably reduced.

```
void loop () {

digital Write (LED_BUILTIN, HIGH);

delay (1000);

digital Write (LED_BUILTIN, LOW);

delay (1000);

delay (1000);

digital Write (LED, LOW);

delay (1000);
```

4.1.3 APPLICATIONS

- Prototyping of Electronics Products and Systems
- Easy to use for beginner level DIYers and makers.
- Projects requiring Multiple I/O interfaces and communications.

4.2 pH SENSOR

A pH meter is a scientific instrument that measures the hydrogen-ion activity in water-based solutions, indicating its acidity or alkalinity expressed as pH. The pH meter measures the difference in electrical potential between a pH electrode and a reference electrode, and so the pH meter is sometimes referred to asa"potentiometric pH meter".

Fig 4.2 pH Sensor

4.2.1 pH SENSOR WORKS

The pH amplifier inside the handle is a circuit which allows the standard combination pH electrode to be monitored by a lab interface. The cable from the pH amplifier ends in a BTA plug. The pH Sensor will produce a voltage of approximately 1.75 volts in a pH 7 buffer. The voltage will increase by about 0.25 volts for every pH number decrease. The voltage will decrease by about 0.25 volts/pH number as the pH increases. The Venire gel-filled pH Sensor is designed to make measurements in the pH range of 0 to 14. The gel-filled reference half-cell is sealed; it cannot be refilled.

4.3 ULTRASONIC SENSOR

An ultrasonic sensor is an instrument that measures the distance to an object using ultrasonic sound waves. An ultrasonic sensor uses a transducer to send and receive ultrasonic pulses that relay back information about an object's proximity. High-frequency sound waves reflect from boundaries to produce distinct echo patterns.

4.3.1 WORKING PRINCIPLE OF ULTRASONIC SENSOR

As shown above the HC-SR04 Ultrasonic (US) sensor is a 4-pin module, whose pin names are Vcc, Trigger, Echo and Ground respectively. This sensor is a very popular sensor used in many applications where measuring distance or sensing objects are required. The module has two eyes like projects in the front which forms the Ultrasonic transmitter and Receiver. The sensor works with the simple formula that

Distance = Speed × Time

The Ultrasonic transmitter transmits an ultrasonic wave, this wave travels in air and when it gets objected by any material it gets reflected back toward the sensor this reflected wave is observed by the Ultrasonic receiver module

HC-SR04 distance sensor is commonly used with both microcontroller and microprocessor platforms like Arduino, ARM, PIC, Raspberry Pie etc. The following guide is universally since it has to be followed irrespective of the type of computational device used. Power the Sensor using a regulated +5V through the Vcc ad Ground pins of the sensor. The current consumed by the sensor is less than 15mA and hence can be directly powered by the on board 5V pins (If available). The Trigger and the Echo pins are both I/O pins.

4.3.2 ULTRASONIC SENSORS ARE BEST USED IN THE NON-CONTACT DETECTION

- Presence
- Level

4.3.3 ULTRASONIC ARE INDEPENDENT

- Light
- Smoke
- Dust

Fig 4.3 Ultrasonic Sensor

Pin Number	Pin Name	Description
1	Vcc	The Vcc pin powers the sensor, typically with +5V
2	Trigger	Trigger pin is an Input pin. This pin has to be kept high for 10us to initialize measurement by sending US wave.
3	Echo	Echo pin is an Output pin. This pin goes high for a period of time which will be equal to the time taken for the US wave to return back to the sensor.
4	Ground	This pin is connected to the Ground of the system.

Table 4.2 Ultrasonic Sensor Pin Configuration

4.3.4 HC-SR04 SENSOR FEATURES

- Operating voltage: +5V
- Theoretical Measuring Distance: 2cm to 450cm
- Practical Measuring Distance: 2cm to 80cm
- Accuracy: 3mm
- Operating Current: <15mA

4.4 FLOW SENSOR

Flow sensor more commonly referred to as a “flow meter” is an electronic device that measures or regulates the flow rate of liquids and gasses within pipes and tubes. Flow sensors are generally connected to gauges to render their measurements, but they can also be connected to computers and digital interfaces.

4.4.1 WORKING PRINCIPLE

Water flow sensor consists of a plastic valve from which water can pass. A water rotor along with a Hall Effect sensor is present the sense and measure the water flow. When water flows through the valve it rotates the rotor. By this, the change can be observed in the speed of the motor. This change is calculated as output as a pulse signal by the Hall Effect sensor. Thus, the rate of flow of water can be measured. The main working principle behind the working of this sensor is the Hall effect. According to this principle, in this sensor, a voltage difference is induced in the conductor due to the rotation of the rotor. This induced voltage difference is transverse to the electric current. When the moving fan is rotated due to the flow of water, it rotates the rotor which induces the voltage. This induced voltage is measured by the hall effect sensor and displayed on the LCD display. The water flow sensor can be used with hot waters, cold waters, warm waters, clean water, and dirty water also. These sensors are available in different diameters, with different flow rate ranges. These sensors can be easily interfaced with microcontrollers like Arduino. For this, an Arduino microcontroller board for processing, a Hall Effect water flow sensor, a 16×2 LCD display, and Breadboard connecting wires are required. The sensor is placed at the water source inlet or at the opening of the pipe.

4.4.2 APPLICATIONS OF WATER FLOW SENSOR

Water flow sensors can measure the rate of flow of water either by measuring velocity or displacement. These sensors can also measure the flow of water like fluids such as measuring milk in a dairy industry etc…There are various types of water flow sensors available based on their diameter and method of measuring. A cost-effective and most commonly used water flow sensor is Paddlewheel sensor. It can be used with water-like fluids. For the type of applications where a straight pipe is not available for inlet, Positive displacement flow meter is used. The LCD display is used to display the measurements. The magnetic Hall Effect water flow sensor outputs a pulse of every revolution of the rotor. The Hall Effect sensor present in the device is sealed from water to keep it safe and dry.

4.4.3 FEATURE WATER FLOW SENSOR

- Compact
- Easy to Install High Sealing Performance
- Water Flow Sensor has a High-Quality Hall Effect Sensor

Fig 4.4 Flow Sensor

4.5 LCD

The term LCD stands for liquid crystal display. It is one kind of electronic display module used in an extensive range of applications like various circuits & devices like mobile phones, calculators, computers, TV sets, etc. These displays are mainly preferred for multi-segment light-emitting diodes and seven segments. The main benefits of using this module are inexpensive; simply programmable, animations, and there are no limitations for displaying custom characters, special and even animations, etc.16×2 LCD is named so because; it has 16 Columns and 2 Rows. There are a lot of combinations available like, 8×1, 8×2, 10×2, 16×1, etc. But the most used one is the 16*2 LCD; hence we are using it here.

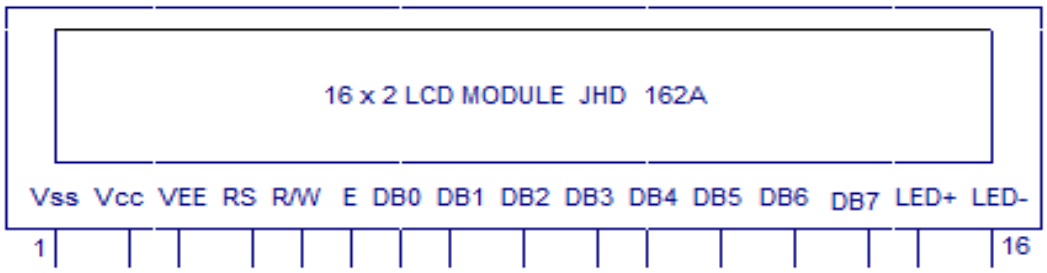


Fig 4.5 16x2 LCD Pin Diagram

Pin1 (Vss): Ground pin of the LCD module.

Pin2 (Vcc): Power to LCD module (+5V supply is given to this pin)

Pin3 (VEE): Contrast adjustment pin. This is done by connecting the ends of a 10K potentiometer to +5V and ground and then connecting the slider pin to the VEE pin. The voltage at the VEE pin defines the contrast. The normal setting is between 0.4 and 0.9V.

Pin4 (RS): Register select pin. The JHD162A has two registers namely command register and data register. Logic HIGH at RS pin selects data register and logic LOW at RS pin selects command register. If we make the RS pin HIGH and feed an input to the data lines (DB0 to DB7), this input will be treated as data to display on LCD screen.

If we make the RS pin LOW and feed an input to the data lines, then this will be treated as a command (a command to be written to LCD controller – like positioning cursor or clear screen or scroll).

Pin5 (R/W): Read/Write modes. This pin is used for selecting between read and write modes. Logic HIGH at this pin activates read mode and logic LOW at this pin activates write mode.

Pin6 (E): This pin is meant for enabling the LCD module. A HIGH to LOW signal at this pin will enable the module.

Pin7 (DB0) to Pin14 (DB7): These are data pins. The commands and data are fed to the LCD module though these pins.

Pin15 (LED+): Anode of the back light LED. When operated on 5V, a 560-ohm resistor should be connected in series to this pin. In Arduino based projects the back light LED can be powered from the 3.3V source on the Arduino board.

Pin16 (LED-): Cathode of the back light LED

4.5.1 CIRCUIT DIAGRAM FOR LCD

The interfacing the LCD screen to the Arduino board, a pin header strip needs to be solder to pin-14 or 16 of the LCD. We can notice this in the following circuit diagram. The following pins need to connect to wire the LCD to an Arduino board.

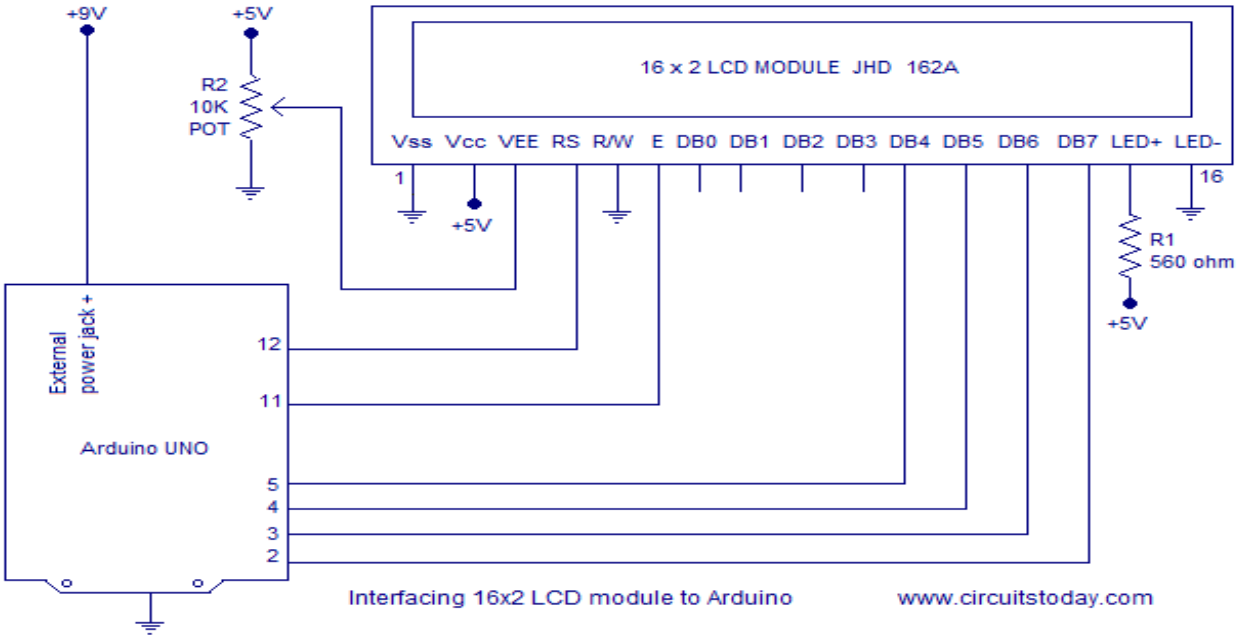


Fig 4.6 LCD Circuit Diagram

- RS pin of LCD to digital pin-12
- Enable pin is connected to digital pin-11
- D4 pin is connected to digital pin -5
- D5 pin is connected to digital pin- 4
- D6 pin is connected to digital pin-3
- D7 pin is connected to digital pin-2
- Read/Write pin is connected to GND
- VSS pin is connected to the GND terminal
- VCC pin is connected to 5V

- A 220-ohm resistor is connected from LED+ to 5V
- LED is connected to the GND terminal
- In addition, connect a 10k pot toward +5V & GND through its wiper to LCD pin3.

4.5.2 SPECIFICATION OF LCD

- The operating voltage of this display ranges from 4.7V to 5.3V
- The display bezel is 72 x 25mm
- The operating current is 1mA without a backlight
- PCB size of the module is 80L x 36W x 10H mm
- HD47780 controller
- LED colour for backlight is green or blue
- Number of columns – 16
- Number of rows – 2
- Number of LCD pins – 16
- Characters – 32
- It works in 4-bit and 8-bit modes
- Pixel box of each character is 5×8 pixel
- Font size of character is 0.125Width x 0.200height

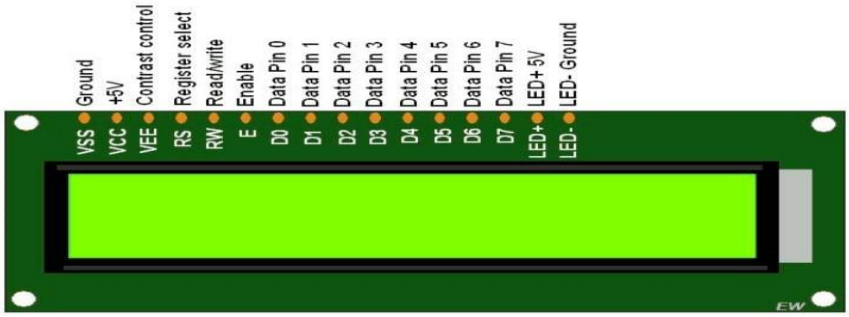


Fig 4.7 LCD

4.6 GSM SIM800C MODULE

SIM800C is a widely used GSM Module with a serial interface modem which runs in between 3.4V-4.4V Voltage level. SIM800C is a Quad-band GSM/GPRS Module which is used in embedded applications where the remote data transfer is required. SIM800C works on 850/900/1800/1900MHz. It can also receive & transmit Voice Call, SMS with low power consumption. The module is controlled by using AT commands. It supports one SIM card interface and has UART (TX & RX) pins along with one RS232 Serial Protocol that can be used to interface with different microcontrollers in embedded applications.

4.6.1 SPECIFICATIONS FOR SMS VIA GSM/GPRS

• Point to point MO and MT

• SMS cell broadcast

• Text and PDU mode

Fig 4.8 Gsm Module

4.7 RELAY

A relay is an electrically operated switch. It consists of a set of input terminals for a single or multiple control signals, and a set of operating contact terminals. The switch may have any number of contacts in multiple contact forms, such as make contacts, break contacts, or combinations thereof.

Relays are used where it is necessary to control a circuit by an independent low-power signal, or where several circuits must be controlled by one signal. Relays were first used in long-distance telegraph circuits as signal repeaters: they refresh the signal coming in from one circuit by transmitting it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

Latching relays require only a single pulse of control power to operate the switch persistently. Another pulse applied to a second set of control terminals, or a pulse with opposite polarity, resets the switch, while repeated pulses of the same kind have no effects. Magnetic latching relays are useful in applications when interrupted power should not affect the circuits that the relay is controlling.

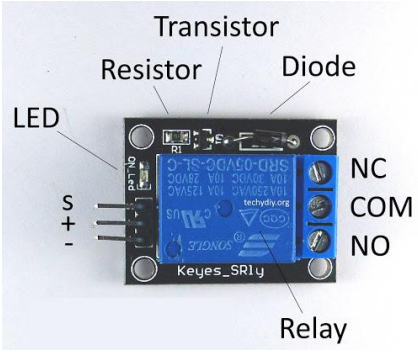


Fig 4.9 Relay

Pin Number	Pin Name	Description
1	Coil End 1	Used to trigger (On/Off) the Relay, normally one end is connected to 5V and the other end to ground
2	Coil End 2	Used to trigger (On/Off) the Relay, normally one end is connected to 5V and the other end to ground
3	Common (COM)	Common is connected to one End of the Load that is to be controlled
4	Normally Close (NC)	The other end of the load is either connected to NO or NC. If connected to NC the load remains connected before trigger
5	Normally Open (NO)	The other end of the load is either connected to NO or NC. If connected to NO the load remains disconnected before trigger

Table 4.3 Relay Pin Configuration

4.7.1 FEATURES OF 5-PIN 5V RELAY

- Trigger Voltage (Voltage across coil) : 5V DC
- Trigger Current (Nominal current) : 70mA
- Maximum AC load current: 10A @ 250/125V AC
- Maximum DC load current: 10A @ 30/28V DC
- Compact 5-pin configuration with plastic molding
- Operating time: 10msec Release time: 5msec
- Maximum switching: 300 operating/minute (mechanically)

4.8 PUMP MOTOR

A pump is a device that moves fluids liquids or gases, or sometimes slurries, [by mechanical action, typically converted from electrical energy into hydraulic energy. Pumps can be classified into three major groups according to the method they use to move the fluid: direct lift, displacement, and gravity pumps Pumps operate by some mechanism typically reciprocating or rotary, and consume energy to perform mechanical work moving the fluid. Pumps operate via many energy sources, including manual operation, electricity, engines, or wind power, and come in many sizes, from microscopic for use in medical applications, to large industrial pumps.

4.8.1 WORKING PRINCIPLE

Electric submersible pumps are multistage centrifugal pumps operating in a vertical position. Liquids, accelerated by the impeller, lose their kinetic energy in the diffuser, where a conversion of kinetic to pressure energy takes place. This is the main operational mechanism of radial and mixed flow pumps. power fluid with the produced fluid downhole, with surface separation.

The pump shaft is connected to the gas separator or the protector by a mechanical coupling at the bottom of the pump. Fluids enter the pump through an intake screen and are lifted by the pump stages. Other parts include the radial bearings (bushings) distributed along the length of the shaft, providing radial support to the pump shaft. An optional thrust bearing takes up part of the axial forces arising in the pump, but most of those forces are absorbed by the protector's thrust bearing.

4.8.2 SPECIFICATION

Model Number: 40 watt cooler pump thermal over load technology submersible Water Pump used for fountain and garden etc.

Type: Water

Power Cord Length: 86cm

Power Consumption: 40watt

Flow Rate: 2200 L/hr

Maximum Head Height: 340 cm

Power Source: Electrical

Suitable For: Salt Water and Fresh Water.

Fig 4.10 Pump Motor

CHAPTER 5

5. SOFTWARE DESCRIPTION

5.1 EMBEDDED C

Embedded C is one of the most popular and most commonly used Programming Languages in the development of Embedded Systems. So, in this article, we will see some of the Basics of Embedded C Program and the Programming Structure of Embedded C.

Embedded C is perhaps the most popular languages among Embedded Programmers for programming Embedded Systems. There are many popular programming languages like Assembly, BASIC, C++ etc. that are often used for developing Embedded Systems but Embedded C remains popular due to its efficiency, less development time and portability. Before digging in to the basics of Embedded C Program, we will first take a look at what an Embedded System is and the importance of Programming Language in Embedded Systems.

5.1.2 PROGRAMMING IN EMBEDDED SYSTEM

As mentioned earlier, Embedded Systems consists of both Hardware and Software. If we consider a simple Embedded System, the main Hardware Module is the Processor. The Processor is the heart of the Embedded System and it can be anything like a Microprocessor, Microcontroller, DSP, CPLD (Complex Programmable Logic Device) and FPGA (Field Programmable Gated Array). All these devices have one thing in common: they are programmable i.e.; we can write a program (which is the software part of the Embedded System) to define how the device actually works.

Embedded Software or Program allow Hardware to monitor external events (Inputs) and control external devices (Outputs) accordingly. During this process, the program for an Embedded System may have to directly manipulate the internal architecture of the Embedded Hardware (usually the processor) such as Timers, Serial Communications Interface, Interrupt Handling, and I/O Ports etc. There are many programming languages that are used for Embedded Systems like Assembly (low-level Programming Language), C, C++, JAVA (high-level programming languages), Visual Basic, JAVA Script (Application level Programming Languages), etc. In the process of making a better embedded system, the programming of the system plays a vital role and hence, the selection of the Programming Language is very important.

5.1.3 INTRODUCTION TO EMBEDDED C PROGRAMMING LANGUAGE

Before going in to the details of Embedded C Programming Language and basics of Embedded C Program, we will first talk about the C Programming Language. The C Programming Language, developed by Dennis Ritchie in the late 60's and early 70's, is the most popular and widely used programming language. The C Programming Language provided low level memory access using an uncomplicated compiler and achieved efficient mapping to machine instructions. The C Programming Language became so popular that it is used in a wide range of applications ranging from Embedded Systems to Super Computers. Embedded C Programming Language, which is widely used in the development of Embedded Systems, is an extension of C Program Language. The Embedded C Programming Language uses the same syntax and semantics of the C Programming Language like main function, declaration of data types, defining variables, loops, functions, statements, etc.

5.1.4 DIFFERENCE BETWEEN C AND EMBEDDED C

There is actually not much difference between C and Embedded C apart from few extensions and the operating environment. Both C and Embedded C are ISO Standards that have almost same syntax, data types, functions, etc. Embedded C is basically an extension to the Standard C Programming Language with additional features like Addressing I/O, multiple memory addressing and fixed-point arithmetic, etc. Programming Language is generally used for developing desktop applications whereas Embedded C is used in the development of Microcontroller based applications.

5.2 PIC C COMPILER:

CCS provides a method to attempt to make sure you can compile code written in older versions of CCS with minimal difficulty by altering the methodology to best match the desired version. Currently, there are 4 levels of compatibility provided: CCS V2.XXX, CCS V3.XXX, CCS V4.XXX and ANSI.

Notice: this only affects the compiler methodology, it does not change any drivers, libraries and include files that may have been available in previous versions.

#device CCS2

- ADC default size is set to the resolution of the device
- Boolean = int8 is compiled as: Boolean = (int8 != 0)
- Overload directive is required if you want to overload functions
- Pointer size was set to only access first bank (PCM *=8, PCB *=5)
- var16 = NegConst8 is compiled as: var16 = NegConst8 & 0xFF (no sign extension)
- Compiler will NOT automatically set certain #fuses based upon certain code conditions.

• rom qualifier is called _rom

#device CCS3

• ADC default is 8 bits (#device ADC=8)

• Boolean = int8 is compiled as: Boolean = (int8 & 1)

• Overload directive is required if you want to overload functions

• Pointer size was set to only access first bank (PCM *=8, PCB *=5)

• var16 = NegConst8 is compiled as: var16 = NegConst8 & 0xFF (no sign extension)

• Compiler will NOT automatically set certain #fuses based upon certain code conditions.

• rom qualifier is called _rom

#device CCS4

• ADC default is 8 bits (#device ADC=8)

• boolean = int8 is compiled as: boolean = (int8 & 1)

• You can overload functions without the overload directive

• If the device has more than one bank of RAM, the default pointer size is now 16

#device *=16

• var16 = NegConst8 is will perform the proper sign extension

• Automatic #fuses configuration (see next section)

#device ANSI

• Same as CCS4, but if there are any discrepancies are found that differ with the ANSI

• Data is signed by default

• const qualifier is read-only RAM, not placed into program memory

• Compilation is case sensitive by default

• Constant strings can be passed to functions

This C compiler, is fully optimised for use with PIC microcontrollers. Built in functions make coding the software very easy. Based on original K&R, the integrated C development environment gives developers a fast method to produce efficient code from an easily Maintainable high-level language.

5.2.1 CAPABILITIES

- Arrays up to 5 subscripts
- Structures and Unions may be nested.
- Custom bit fields (1-8 bits) within structures.
- Enumerated types,
- Constant variables, arrays and strings.

5.2.2 STANDARD C FUNCTIONS

IF, ELSE, WHILE, DO, SWITCH, CASE, FOR,
RETURN, GOTO, BREAK, CONTINUE

· !, ~, ++, --, *, /, %, +, -, <<, >>, <. <=, >, >=,

==, !=, &, ^, |, &&, ||, ?:, =, +=, -=, *=, /=, %=,
>>=, <<=, &=, ^=, |·
TYPEDEF, STATIC, AUTO, CONST, ENUM, STRUCT, UNION

5.2.3 FEATURES

- Built in Libraries for RS232 serial I/O library, I/O, I2C, discrete I/O and precision delays.
- Integrates with MPLAB and other simulators/emulators for source level debugging.
- Standard Hex file and debug files ensure compatibility with all programmers.
- Formatted Printf allows easy formatting and display in Hex or decimal.
- Efficient function implementation allows call trees deeper than the hardware stack.
- Access to hardware from easy-to-use C functions, Timers, A/D, E2, SSP, PSP, I2C & more.
- 1,8-, and 16-bit types.
- Assembly code may be inserted anywhere in source and may reference C variables.
- Automatic linking handles multiple code pages.
- Inline procedures supported; Linker automatically determines optimum architecture or it can be manually specified.
- Compiler directives determine if tri-state registers are refreshed on every I/O
- Constants including strings and arrays are saved in program memory.
- Standard one bit type Short Int permits the compiler to generate efficient Bit oriented code.
- #BIT and #BYTE allow C variables to be placed at absolute addresses to map register to C variables.
- Reference parameters may be used to improve code readability and inline procedure efficiency.
- Both an integrated editor/compiler and a cmd line compiler.

- Special windows show the RAM memory map, C/Assembly listing and the calling tree.
- Interrupt procedures supported on PCM. The compiler generates all start up and clean up code as well as identifying the correct interrupts procedure to be called.
- Updates via modem for 30 days included.

5.2.4 DESCRIPTION

This integrated C development environment gives developers the capability to quickly produce very efficient code from an easily maintainable high level language. The compiler includes built in functions to access the PIC hardware such as READ_ADC to read a value from the A/D converter. Discrete I/O is handled by describing the port characteristics in a PRAGMA. Functions such as INPUT and OUTPUT_HIGH will properly maintain the tri-state registers. Variables including structures may be directly mapped to memory such as I/O ports to best represent the hardware structure in C. The microcontroller clock speed may be specified in a PRAGMA to permit built in functions to delay for a given number of microseconds or milliseconds. Serial I/O functions allow standard functions such as GETC and PRINTF to be used for RS-232 like I/O. The hardware serial transceiver is used for applicable parts when possible. For all other cases a software serial transceiver is generated by the compiler. The standard C operators and the special built in functions are optimised to produce very efficient code for the bit and I/O functions. Functions may be implemented inline or separate.

Function parameters are passed in reusable registers. Inline functions with reference parameters are implemented efficiently with no memory overhead. During the linking process the program structure including the call tree is analysed.

Functions that call one another frequently are grouped together in the same page. Calls across pages are handled automatically by the tool transparent to the user. Functions may be implemented inline or separate. RAM is allocated efficiently by using the call tree to determine how locations can be re-used. Constant strings and tables are saved in the device ROM. The output hex and debug files are selectable and compatible with popular emulators & programmers including MPLAB for source level debugging. The Professional Package (PCW) provides both compilers in a powerful Windows environment.

5.5ARDUINO IDE

Arduino IDE stands for “Integrated Development Environment”: it is an official Software introduced by Arduino.cc, that is mainly used for editing, 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. In this article, we will introduce the Software, how we can install it, and make it ready for developing applications using Arduino modules.

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed.

The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board.

The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer Software Serial library allows for serial communication on any of the Uno's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

ARDUINO IDE DEFINITION

- 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, 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.

5.5 PROGRAMMING

The Arduino Uno can be programmed with the Arduino software download. Select "Arduino Uno from the **Tools** > **Board** menu according to the microcontroller on your board. For details, see the reference and tutorials. The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol reference, C header files. The ATmega16U2 or 8U2 in the rev1 and rev2 boards firmware source code is available The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

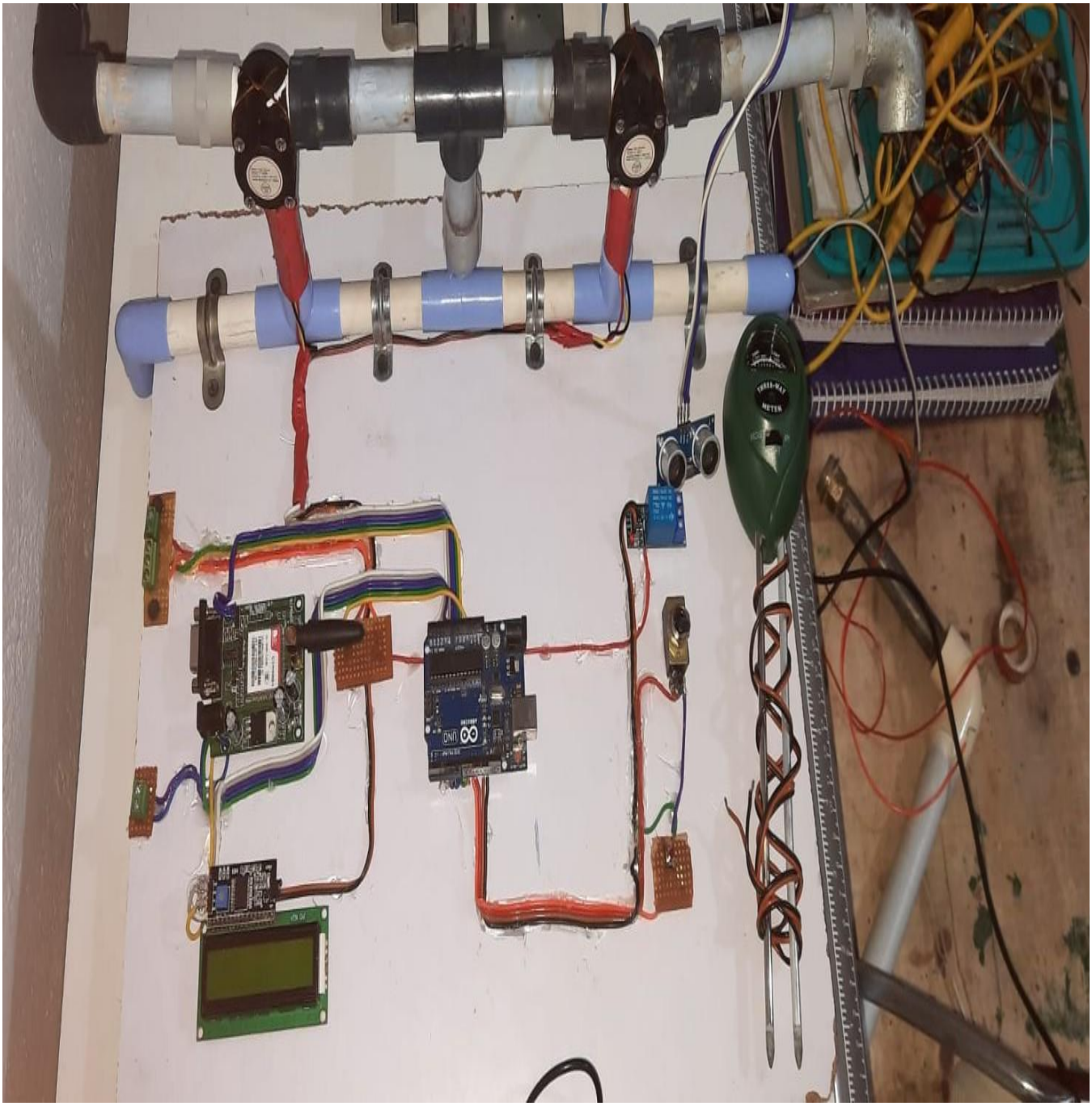
- On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2.
- On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

5.6.1 USB OVERCURRENT PROTECTION

The Arduino Uno has a resettable polypus that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own 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.

CONCLUSION

In this proposed work, the look and deployment of the important time water quality monitoring system for drinkable using wireless sensor network has been presented. The developed system has been field tested at school hostel region, for monitoring of water quality parameters. It's an occasional cost, light weight system and has low power consumption. Moreover, the system is in a position to log bulk data and transfer to remote locations. The contamination detection algorithm and also the fuzzy rules help to spot the contamination within the pipeline and classify water supported the contamination. This deployment provides rich data to the water consumers/public, authorities in municipal office. The sms alert and mobile app ensures the security of beverage. Our future plan is to analyse the performance of designed system against other sorts of contaminants like nitrates, lead etc.



PROJECT IMPLEMENTATION

APPENDIX

```
#include <EEPROM.h>

#include <Wire.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);

#include <SoftwareSerial.h>

Software Serial GSM(5, 6 );

float calibration value = 7.0;

int phval = 0;

unsigned long int avgval;

int buffer_arr[10], temp;

long duration, inches;

int set_val, percentage;

bool state, pump;

int temp_var, cur_pump;

volatile int last_temp, last_pump;

unsigned int checkTime = 10;

unsigned int checkTimeCount = 1;

unsigned int hourTime = 0;
```

```
unsigned int return Value = 0;
unsigned int secondInput = 0;
unsigned int initialValue;
volatile int flow_frequency1;
unsigned char flowsensor1 = 3;
unsigned long currentTime;
unsigned long cloopTime;
{
  flow frequency++;
  secondInput = 1;
}
{
  flow_frequency++;
}
char phone_no[] = "xxxx";
void setup()
{
    Serial. Begin(9600);
```

```
 lcd.begin(16, 2);
 lcd.setCursor(0, 0);
 lcd.print("   WEL COME TO  ");
 lcd.setCursor(0, 1);
 lcd.print("   OUR PROJECT  ");
 delay(2000);
 lcd.clear();
 lcd.init();
 lcd.print("WATER LEVEL:");
 lcd.setCursor(0, 1);
 lcd.print("PUMP:OFF MANUAL");
 pinMode(flowsensor, INPUT);
 pinMode(flowsensor1, INPUT);  Serial.begin(9600);
currentTime = millis();
 cloopTime = currentTime;
 Serial.println("Initializing....");  Serial.println("Initialized Successfully");
 sendSMS(phone_no, "System is Ready");
 pinMode(8, OUTPUT);
```

```
  pinMode(9, INPUT);
  pinMode(10, INPUT_PULLUP);
  pinMode(11, INPUT_PULLUP);
  pinMode(13, OUTPUT);
  set_val = EEPROM.read(0);
  if (set_val > 150)set_val = 150;
  delay(2000);
  lcd.clear();
}
void loop() {
  ph();
  delay(1000);  delay(4000);
  leakage();
  delay(1000);
  lcd.clear();
  water();
  delay(1000);
  delay(5000);
```

```
  lcd.clear();
}
void water() {
 digitalWrite(8, HIGH);
 delay Microseconds(5);
  digitalWrite(8, LOW);
  duration = pulseIn(9, HIGH);
  inches = microsecondsToInches(duration);
  percentage = (set_val - inches) * 100 / set_val;
  lcd.setCursor(0, 0);
  if (percentage < 0)percentage = 0;
  lcd.print("WATER LEVEL: ");
  lcd.print(percentage);
  lcd.print("%");
  if (percentage < 30 & digitalRead(11))pump = 1;
  if (percentage > 95)pump = 0;
  digitalWrite(13, !pump);
  lcd.setCursor(1, 1);
```

```
  if (pump == 1) {
    lcd.print("PUMP:ON ");
    cur_pump = 2;
else if (pump == 0) {
    lcd.print("PUMP:OFF");  cur_pump = 1;
    if (last_pump != cur_pump) {
      sendSMS(phone_no, "PUMP:OFF ");
    }
    last_pump = cur_pump;
  }
  lcd.setCursor(9, 1);
  if (!digitalRead(11))lcd.print("MANUAL");
  else lcd.print("AUTO");
  if (!digitalRead(10) & !state & digitalRead(11))
{
    state = 1;
    set_val = inches;
    EEPROM.write(0, set_val);
```

```
  }
  if (!digital Read(10) & !state & !digital Read(11)) {
    state = 1;
    pump = !pump;
  }
  if (digitalRead(10))state = 0;  delay(500);
}
void ph() {
  for (int i = 0; i < 10; i++)
{
    buffer_arr[i] = analogRead(A0);
    delay(30);
  }
  avgval = 0;
  for (int i = 2; i < 8; i++);
  float ph_act = -5.70 * volt + calibration_value;
  lcd.setCursor(0, 0);
  lcd.print("pH Val:");
```

```
  lcd.setCursor(8, 0);
  lcd.print(ph_act);
  if (ph_act >= 7)
  {
    lcd.setCursor(0, 1);
    lcd.print("Netural");
    temp_var = 2;
    if (last_temp != temp_var) {
      sendSMS(phone_no, "Netural");
void leakage() {
  currentTime = millis();
  if (currentTime >= (cloopTime + 1000))
  {
    lcd.clear();
    hourTime += (millis() / 1000) / 60 / 60;
    cloopTime = currentTime; // Updates cloopTime
    l_hour = (flow_frequency * 60 / 7.5); // (Pulse frequency x 60 min) / 7.5Q =
flowrate in L/hour
```

```
  Serial.print(" L/hour   ");
  Serial.println(initialValue);
  if (l_hour > 0 && initialValue > 0 && secondInput != 1) {
    avarageValue = avarageValue + l_hour;
    avarageCount += 1;
    avarageResult = avarageValue / avarageCount;
  }
  if (secondInput == 1 && l_hour > 0) {
    returnValue = returnValue + 1;
  }
  if ((millis() / 1000) > (checkTime * checkTimeCount)) {
    checkTimeCount += 1;
  }
  if ((millis() / 1000) <= (checkTime * checkTimeCount) && l_hour > estimateAvg && l_hour > 0 && initialValue > 0) {
    lcd.setCursor(0, 0);
    lcd.print("Waterflow"); lcd.setCursor(0, 1);
    if (l_hour < (initialValue - 200) && l_hour > 0 && initialValue > 0) {
```

```
      lcd.setCursor(0, 0);
      lcd.print("LEAKAGE");
      lcd.setCursor(0, 1);
      lcd.print(l_hour);      if (hour == 0 && initial Value > 0) {
      lcd.setCursor(0, 0);
      lcd.print("NO FLOW");
      lcd.setCursor(0, 1);
      lcd.print(l_hour);
     }
   }
   if (hour > 100 && second Input == 1 && return Value == 3) {
     initial Value = hour;
     detach Interrupt(0);
     void sendSMS(char *number, char *msg)
{
   GSM.println("AT+CMGS=\"xxxx\"\r");
  delay(500);
  GSM.println(msg); // Message contents
```

```
  delay(500);
  GSM.write(byte(26)); //Ctrl+Z  send message command (26 in decimal).
  delay(5000);
}
void initModule(String cmd, char *res, int t) {
  Serial.println(cmd);
  GSM.println(cmd);
  delay(100)
  while (GSM.available() > 0) {
    if (GSM.find(res)) {
      Serial.println(res);
      delay(t);
      return;
     } else
 {
      Serial.println("Error");
    }
```

REFERENCES

[1] Haque, Halima & Labeeb, Kashshaf & Riha, Rabea & Khan, Mohammad Nasfikur. (2021). IoT Based Water Quality Monitoring System By Using Zigbee Protocol. 10.1109/ESCI50559.2021.9397031.

[2] S. A. Hamid, A. M. A. Rahim, S. Y. Fadhlullah, S. Abdullah, Z. Muhammad and N. A. M. Leh (2020), IoT based Water Quality Monitoring System and Evaluation, 10th IEEE International Conference on Control System, Computing and Engineering (ICCSCE), 2020, pp. 102-106, doi: 10.1109/ICCSCE50387.2020.9

[3] T. Boonlar, "Online Checking System for Drinking Quality of Drinking Water Vending Machine," 2019 15th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), 2019, pp. 531-537, doi: 10.1109/SITIS.2019.00090.

[4] R. Sugantha Lakshmi, G. Chandra Praba, K. Abhirami, 2021, Automated Water Management and Leakage Detection System using IOT, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) ICRADL – 2021 (Volume 09 – Issue 05).

[5] Th. Thongleam, W. Buangam, H. Dinsakul and B. Jarernpun, 2014, Automatic Water Quality Measurement and Processing for Krachang-Taptim Fish, ECTI-CARD Proceedings,

[6] Anusak, P. 2008, Automatic dissolved oxygen monitoring system using wireless sensor network for shrimp farms, Master of Engineering. King Mongkut's University of Technology North Bangkok.Faculty of Engineering. Electrical Engineering.

[7] Gupta, S., Kohli, M., Kumar, R., & Bandral, S. (2021). IoT Based Underwater Robot for Water Quality Monitoring. IOP Conference Series: Materials Science and Engineering, 1033, 012013. https://doi.org/10.1088/1757-899x/1033/1/0120

[8] M.J.; McKenzie, J.E.; Bossuyt, P.M.; Boutron, I.; Hoffmann, T.C.; Mulrow, C.D.; Shamseer, L.; Tetzlaff, J.M.; Akl, E.A.; Brennan,S.E.; et al. The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. BMJ **2021**, 372, n71.

[9] Singh, R.; Baz, M.; Gehlot, A.; Rashid, M.; Khurana, M.; Akram, S.V.; Alshamrani, S.S.; AlGhamdi, A.S. Water Quality Monitoring and Management of Building Water Tank Using Industrial Internet of Things. Sustainability **2021**, 13, 8452

[10] Yasin, S.; Mohd Yunus, M.F.; Abdul Wahab, N. The development of water quality monitoring system using internet of things. J. Educ. Learn. Stud. 2020.