ELECTRONIC PRODUCT DESIGN PROJECT REPORT

on

SMART POND PISCICULTURE

Submitted by

ABHIJITH P
ADITHYAN V A
AKSHAY M SEBASTIAN
AKSHAY P
ANAKHA SURESH

in partial fulfillment of requirement for the award of the degree

of

BACHELOR OF TECHNOLOGY
In
ELECTRONICS AND COMMUNICATION

DIVISION OF ELECTRONICS ENGINEERING
SCHOOL OF ENGINEERING
COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY
KOCHI-682022

MAY 2024

DIVISION OF ELECTRONICS ENGINEERING
SCHOOL OF ENGINEERING
COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY
KOCHI-682022

CERTIFICATE

*Certified that the seminar report entitled **"SMART POND PISCICULTURE"** is a bonafide work of **ABHIJITH P, ADITHYAN V A, AKSHAY M SEBASTIAN, AKSHAY P, ANAKHA SURESH** towards the partial fulfillment for the award of the degree of B.Tech in Electronics and Communication of Cochin University of Science and Technology, Kochi-682022.*

Project Co-Ordinator

Mrs. Rinu C Varghese

Head of the Division

Dr. Deepa Sankar

ACKNOWLEDGEMENT

We express our overwhelming indebtedness in God Almighty for the successful completion of our project. First and foremost, we express our gratitude to Dr. Deepa Sankar, Head of the Division of Electronics and Engineering. We would like to thank our project guides Mrs. Merin Skariah, Mrs. Rinu C Varghese and Mrs. Lakshmi S Panicker for guiding us in our project and providing us with adequate facilities in order to complete the task we had undertaken. Our sincere gratitude is expressed and extended to our lab staff and other faculty members of the Electronics and Communication Division for their assistance and unwavering support. We proudly thank all our friends who helped us in various stages of this endeavor.

ABHIJITH P (20321003)

ADITHYAN V A (20321006)

AKSHAY M SEBASTIAN (20321012)

AKSHAY P (20321013)

ANAKHA SURESH (20321016)

ABSTRACT

In pisciculture, maintaining optimal water quality is crucial for the health and growth of aquatic organisms. pH level is one of the key parameters influencing water quality. Manual monitoring of pH can be labour intensive and prone to errors. Also feeding food to the fish time to time is also laborious and time consuming. The automatic pH checking system monitors pH level using a pH sensor during specific days and sends the pH level data to the microcontroller. The microcontroller is programmed to control the solution dispenser which is connected to separate tanks containing an acidic and a basic solution. If pH level is below 7 the basic solution is dispensed to counteract the acidity in the water body. If the pH level is above 7 then the acidic solution is dispensed to counteract the basic nature of the waterbody. The time period of on solution dispenser is also programmed into the microcontroller and it also depends on the pH level in the data. Also a storage containing fish food is controlled by the microcontroller which is programmed to dispense food at certain period of time in a day.

CONTENTS

List of Tables

List of Figures

List of Abbreviations

pH - Potential of Hydrogen

DC - Direct Current

RTC – Real Time Clock

V - volts

MHz – Megahertz

SRAM – Static Random Access Memory

EEPROM – Electrically Erasable Programmable Read Only Memory

PWM – Pulse Width Modulation

CHAPTER 1

INTRODUCTION

1.1 Background

With the increasing demand for seafood and the challenges posed by traditional fish farming methods, there's a growing interest in adopting smart technologies to enhance pond pisciculture practices. Smart pond pisciculture systems integrate sensors, automation, and data analytics to monitor and optimize water quality, feeding schedules, and environmental conditions in real-time. By improving efficiency, productivity, and sustainability, these systems offer a promising solution to modernize fish farming operations and meet the evolving demands of the aquaculture industry.

1.2 Purpose

Smart pond pisciculture revolutionizes traditional fish farming by integrating cutting-edge technologies. Through continuous monitoring and precise control, it optimizes water quality parameters, resource utilization, and environmental sustainability. By leveraging real-time data insights, it enhances productivity, mitigates risks, and ensures the long-term viability of fish farming operations.

Questions that will be researched and answered in this report includes:

- How do smart pond pisciculture systems improve water quality management?
- What are the benefits of automated feeding schedules in smart pond pisciculture?
- How do smart pond pisciculture systems contribute to sustainability?
- How can data analytics enhance decision-making in smart pond pisciculture?

1.3 Scope

The scope of smart pond pisciculture encompasses the application of advanced technologies to optimize fish farming practices, including:

1. Improving water quality management
2. Enhancing resource utilization efficiency
3. Real-time monitoring and data analysis
4. Automation of key processes
5. Promoting sustainability and environmental stewardship

Through these initiatives, smart pond pisciculture aims to revolutionize traditional fish farming methods, making them more efficient, productive, and environmentally sustainable.

CHAPTER 2
THEORY

2.1 Components

2.1.1 Arduino UNO:

Arduino is an open-source electronics platform based on easy-to-use hardware and software. It consists of a microcontroller board and a development environment that simplifies the process of programming and creating electronics projects. Here's a detailed explanation covering its various aspects:

- Microcontroller: The Arduino Uno is powered by the Atmega328P microcontroller, which operates at a clock speed of 16 MHz. It has 32 KB of flash memory for storing the program code (of which 0.5 KB is used by the bootloader), 2 KB of SRAM for runtime data storage, and 1 KB of EEPROM for non-volatile data storage.
- Pins and Ports: The Arduino Uno features a total of 14 digital input/output pins (labeled D0 to D13), with 6 of them capable of providing PWM output. Additionally, it has 6 analog input pins (A0 to A5), each of which can measure analog voltage in the range of 0 to 5 volts. The board also provides power and ground pins for connecting external components.
- Voltage Supplies: The Arduino Uno can be powered via the USB connection, which supplies 5 volts from the host computer, or through an external power source connected to the DC power jack. The recommended input voltage range is 7 to 12 volts, although the board can tolerate up to 20 volts. The onboard voltage regulator regulates the input voltage to provide a stable 5-volt supply to the microcontroller and other components.

Features:

- USB Interface: The Arduino Uno features a USB interface (USB-B connector), allowing it to be easily connected to a computer for programming and power.
- Reset Button: A reset button is provided on the board, which restarts the program execution when pressed. This is useful for debugging and uploading new sketches.
- Power LED: There is a power LED (labeled 'ON') that indicates when the board is powered.
- Built-in LEDs: The Uno has built-in LEDs connected to digital pins 13 (labeled 'L') and built-in LED connected to pin 13, which is often used for simple visual feedback in sketches.
- Crystal Oscillator: The Atmega328P microcontroller on the Uno uses an external 16 MHz crystal oscillator for clocking.

Advantages:

- Beginner-friendly: The Arduino Uno is designed with beginners in mind, featuring a simple and intuitive interface that allows users to quickly get started with electronics and programming.
- Versatility: With its ample GPIO pins, analog inputs, and support for various communication protocols (such as SPI, I2C, UART), the Uno can be used for a wide range of projects, from simple blinking LED experiments to complex robotics applications.
- Extensibility: The Uno is compatible with a vast ecosystem of shields and modules, allowing users to easily add functionalities such as Wi-Fi, Bluetooth, GPS, motor control, and more without the need for complex wiring.
- Open-source: Both the hardware design and software environment of the Arduino Uno are open-source, fostering a collaborative community of users and developers who contribute libraries, tutorials, and other resources.

Applications:

- Prototyping: The Arduino Uno is widely used for prototyping electronic projects, enabling rapid iteration and experimentation.
- Education: Many educational institutions utilize the Arduino Uno for teaching electronics, programming, and physical computing concepts due to its simplicity and accessibility.
- DIY Electronics: Hobbyists and makers often use the Uno for creating custom gadgets, interactive installations, home automation systems, and art projects.
- Embedded Systems: The Uno can be employed in embedded systems projects where a compact, low-cost microcontroller is required, such as controlling sensors and actuators in industrial automation or agricultural monitoring systems.

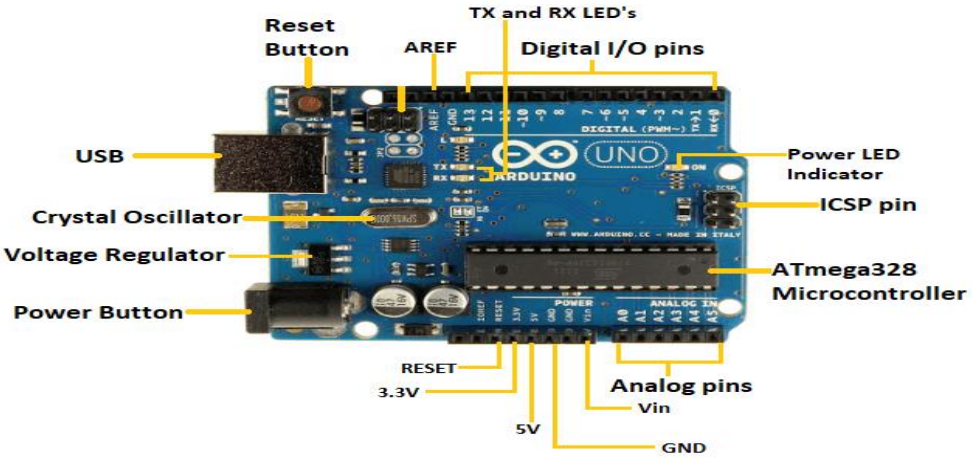


Figure 2.1.1: Arduino UNO

2.1.2 Mini Pumps (x5):

The mini pump is a crucial component of the smart pond pisciculture system, primarily used for pH regulation. It facilitates the dispensing of acidic or basic solutions to adjust pH levels as necessary for optimal fish health. Additionally, the pump is employed for pumping distilled water to the container holding the pH sensor to neutralize the value of pH sensor and remove the contents of the pond water from the container and pH sensor.

Figure 2.1.2: Mini pump

2.1.3 Relay Module:

A relay module is a versatile electronic component used for controlling multiple electrical circuits or devices using low-voltage signals from a microcontroller or other control system. Its operation is based on the principle of electromechanical switching, where a low-power control signal activates an electromagnetic coil, causing the relay contacts to open or close, thereby switching the connected load circuits on or off. One of the key features of relay modules is their ability to isolate the control circuit (input side) from the load circuit (output side), providing electrical protection to the control system and ensuring safety. This isolation is achieved using electromagnetic induction principles, where the control signal energizes the

relay coil, creating a magnetic field that actuates the relay contacts without direct electrical connection between the control and load circuits. Relay modules are available in various configurations, including 4-channel and 2-channel versions, with each channel capable of independently controlling a separate electrical load. This flexibility allows users to switch multiple devices or circuits using a single relay module, simplifying wiring and control in complex systems. Additionally, relay modules often include LED indicators for each channel, providing visual feedback on the status of the relay contacts (e.g., open or closed), aiding in troubleshooting and diagnostics. These modules typically feature low-power consumption, making them suitable for battery-operated or low-power applications. They can be easily interfaced with microcontrollers or other electronic devices using standard communication protocols such as GPIO (General Purpose Input/Output) pins, I2C (Inter-Integrated Circuit), or SPI (Serial Peripheral Interface), allowing for seamless integration into existing control systems. Furthermore, relay modules may include protection features such as flyback diodes or transient voltage suppression to mitigate voltage spikes and protect sensitive electronic components from damage. Relay modules provide a convenient and reliable solution for controlling high-voltage or high-current loads in electronic projects and systems. Their electromechanical switching mechanism, isolation capabilities, multiple channels, low-power consumption, and compatibility with various communication protocols make them indispensable components in a wide range of applications, from home automation to industrial automation and beyond.

Figure 2.1.3: Relay Module

2.1.4 RTC Module:

The Real-Time Clock (RTC) module is a vital component in electronic systems, ensuring accurate time and date tracking over extended periods. At its core, the module relies on a stable clock source, often generated by a crystal oscillator or an integrated circuit, providing a reference for time measurement. To maintain uninterrupted timekeeping, even in the absence of external power, RTC modules typically incorporate a battery backup feature, such as a coin cell battery, to preserve clock operation and data during power outages. Internally, RTC modules contain registers to store time and date information, including seconds, minutes, hours, day, date, month, and year. Continuously updated based on the clock signal, these registers can be accessed and modified by the host system using standard communication protocols like I2C or SPI. This allows for seamless integration with microcontrollers or other electronic devices, enabling precise timekeeping and synchronization across systems. Temperature compensation is a common feature in RTC modules, helping to maintain accuracy over a wide range of operating temperatures. This compensates for temperature-induced variations in the clock signal frequency, ensuring consistent timekeeping performance regardless of environmental conditions. Additionally, RTC modules often include alarm functionality, allowing users to set alarms for specific times or dates, making them invaluable for time-sensitive applications. Designed with efficiency in mind, RTC modules operate with low power consumption, making them suitable for battery-operated devices where power efficiency is critical. Calibration is essential for ensuring accuracy, with some modules offering built-in calibration features or user-adjustable settings to fine-tune clock accuracy. In summary, RTC modules play a crucial role in electronic systems requiring precise timekeeping, offering features such as battery backup, communication interfaces, temperature compensation, alarm functionality, and low power consumption. Understanding their operation and features is essential for integrating RTC modules into electronic projects and ensuring reliable and accurate timekeeping over time.

Figure 2.1.4: RTC Module

2.1.5 pH Sensor:

An analog pH sensor is an electronic device designed to measure the pH level of a solution accurately. It operates based on the principle of potentiometry, utilizing a pH-sensitive electrode and a reference electrode immersed in the solution being tested. The potential difference between these electrodes changes with the pH level of the solution, forming the basis for pH measurement. The pH measurement range typically spans from pH 0 to pH 14, covering the entire pH scale, enabling its use in various applications across different industries.

For accurate and reliable pH measurements, analog pH sensors require calibration using buffer solutions of known pH values. These buffer solutions are comprised of a mixture of a weak acid and its conjugate base (or vice versa) in a specific ratio, which helps maintain a stable pH level even when small amounts of acid or base are added. Common buffer solutions include pH 4.01, pH 7.00, and pH 10.01, each with a known and precisely defined pH value, suitable for calibration purposes. During the calibration process, analog pH sensors are immersed in buffer solutions, and their output signals are adjusted based on the difference between the measured pH value and the known pH value of the buffer solution. This calibration ensures that the sensor provides accurate and reliable pH measurements across its operating range. Buffer solutions also serve as reference points for verifying the accuracy of pH measurements during routine testing and maintenance. In addition to calibration and verification purposes, buffer solutions are used to maintain stable pH

conditions in laboratory experiments, chemical reactions, and industrial processes. Their ability to resist changes in pH makes them invaluable for maintaining consistent pH levels in solutions where precise pH control is critical for the success of the process or experiment. Overall, analog pH sensors are essential tools for measuring pH levels accurately in various solutions across different industries. Their operation relies on potentiometry, calibration with buffer solutions, and versatile configurations, making them invaluable for pH monitoring and control in a wide range of applications.

Figure 2.1.5: pH sensor

2.1.6 Jumper Wires:

Jumper wires are essential components in electronics prototyping, serving as flexible connectors to establish electrical connections between various components on a breadboard or circuit board. They are typically constructed using stranded wire encased in insulating material, providing both electrical insulation and mechanical protection. These wires come with connectors at each end, such as solid pins or male/female headers, facilitating easy insertion into breadboard holes or component terminals. Jumper wires come in various lengths, ranging from a few inches to several inches long, to accommodate different spacing requirements on a breadboard or circuit board. They are often color-coded for organization and identification purposes, with commonly used colors including red, black, blue, green, and yellow. This color-coding aids in distinguishing

between different types of connections, making wiring tasks more efficient. The flexible nature of jumper wires allows them to be bent, twisted, and positioned easily to establish connections between components, even in confined spaces. This flexibility is crucial for prototyping and experimentation, where connections may need to be adjusted frequently. Despite their flexibility, jumper wires are designed for reuse and durability, capable of withstanding multiple insertions and removals without degradation in performance. Jumper wires are versatile and find applications in various electronic projects and experiments. They are commonly used to connect components on a breadboard during prototyping, create temporary connections between circuit boards, or interface with sensors, actuators, and other electronic devices. Additionally, it's possible to customize jumper wires by cutting and stripping wire to the desired length and attaching connectors as needed, providing greater flexibility and adaptation to specific project requirements.

Figure 2.1.6: Jumper Wires

2.1.7 Switch:

A switch is an electrical component used to control the flow of current in a circuit by making or breaking the connection between two or more conducting terminals. It essentially acts as a bridge, allowing or interrupting the flow of electricity in response to manual or automatic operation.

Types of Switches used in the circuit:

- Toggle Switch: A toggle switch has a lever or button that can be flipped or pushed to open or close the circuit.
- Push Button Switch: This type of switch is pressed to make a connection and released to break it. It is often used for momentary actions.

Figure 2.1.7: Switches

CHAPTER 3

METHODOLOGY

3.1 Equipment

The equipment used in this research is specified in the table 3.1

Table 3.1: List of Equipment used for the project

Product Type	**Price (Rs)**
pH sensor	1900
Battery	90
Pump(x5)	234
Arduino UNO	750
Relay Module	220
RTC Module	50
LM2596 DC-DC Buck converter	50
Breadboard	60
Miscellaneous	500
Total	3854/-

3.2 Hardware Setup

Figure 3.1 shows a schematic illustration of the different hardware components used in the system. The arrows in the figure indicates the flow of information between the different hardware components.

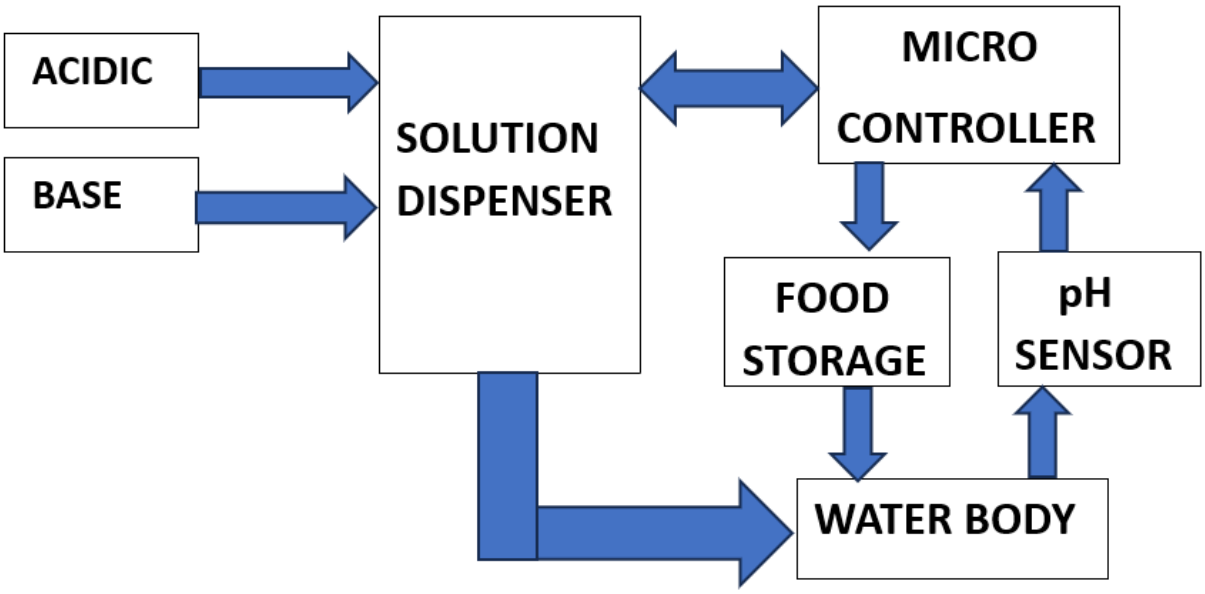


Figure 3.1 Schematic illustration of the hardware

In the next figure 3.2, an image of how each different component were connected to the micro-controller can be viewed.

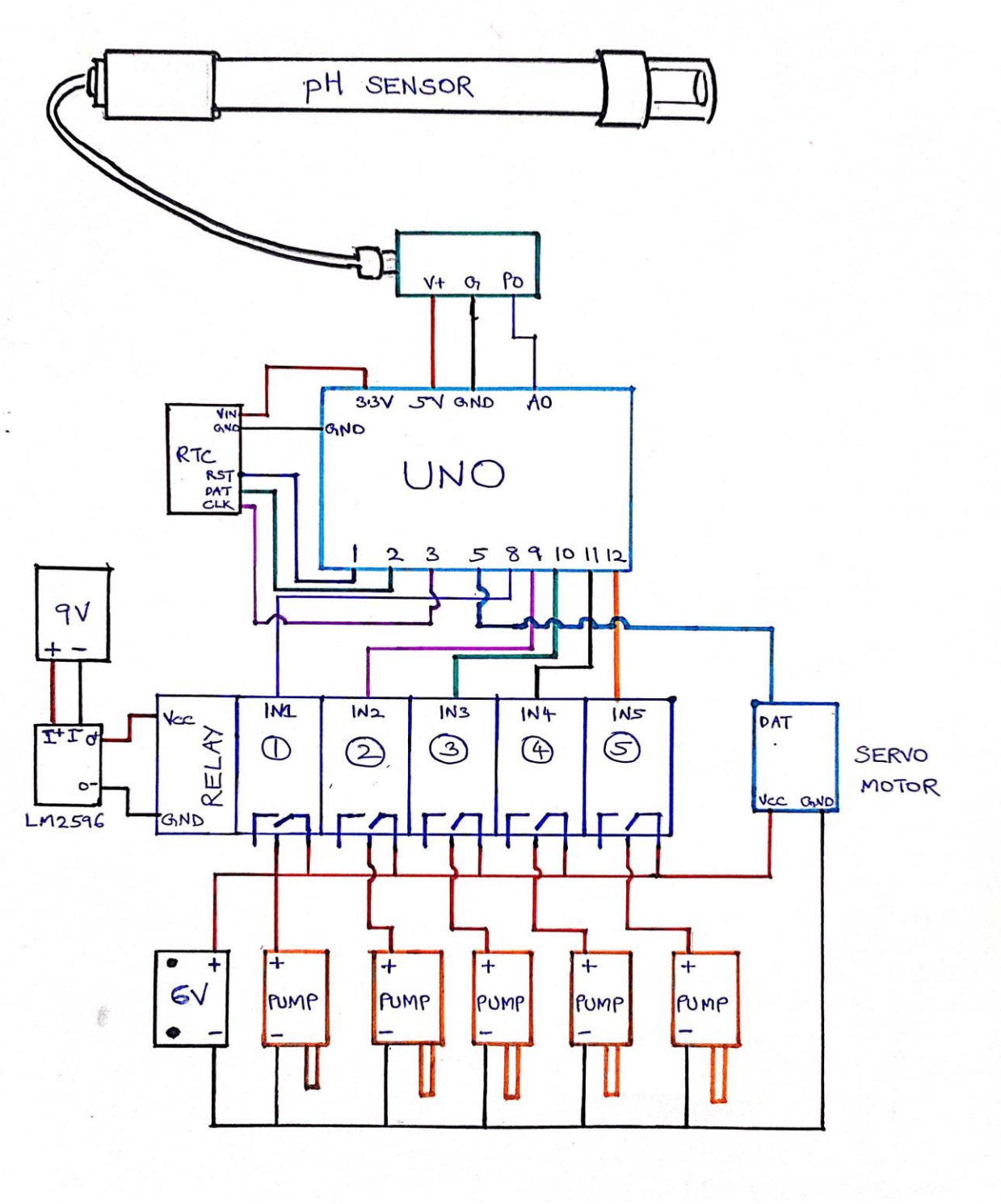


Figure 3.2: Circuit diagram of entire setup

3.2.1 pH sensor

The pH sensor probe was connected to the Arduino chip via a pH meter circuit board that can be seen in Figure above. The pH meter was connected to the Arduino chip through its analog gates along with supplied power at 5V. The specific pH probe used in this research is of the glass-bulb electrode type. The probe was also specifically made for long term submerging. This means that it could monitor the water solvent for longer periods of time without needing frequent calibration.

3.2.2 Fluid Pumps

The fluid pumps were linked in two independent circuits so that each pump could be controlled separately. The only control choices for these pumps were full speed activation, liquid feeding, and total power cutoff. As a result, a relay board was installed in each of the two circuits to allow this capability. The relay boards were linked to the microcontroller's digital pins, as well as 5 V for operational power and a ground connection.

To complete the circuits, the DC motors for the pumps were linked in series with a 5 V power supply and then to the normally closed (NC) port on the relay boards. This meant that until the Arduino delivered a signal to the relays ordering them to open and turn the pumps on, the pumps would always be off.

3.3 Software Setup

1. Libraries:

- Importing necessary libraries for RTC, I2C communication, and servo motor control. For servo motor control, the Servo.h library is commonly used.

2. Initialization:

- Initializing the RTC object with pin configurations for clock, data, and reset, as specified by virtuabotixRTC.

- Declaring variables for time, pH readings, calibration value, buffer array, temporary values, and servo motor control. These variables are essential for various operations within the code.

- Timing constants are defined for motor operations, such as filling water from the pond, spraying base, spraying acid, emptying containers, and filling distilled water.

3. Motor Pin Definitions:

- Assigning pins for controlling different motors responsible for pond, distilled water, acid, base, and drainage.

4. pH Reading Function:

- The ph_read() function reads analog pH sensor data, calculates the average, and converts it to an actual pH value using a calibration formula.

5. pH Adjustment Function:

- The ph_set() function adjusts pH levels based on the current pH reading by controlling the relay module which has different mini pumps connected to it and immersed in different solutions like acid, base and distilled water.

- Initially the pH sensor container has distilled water in it and it is drained and water from the pond is pumped into the container. Then the ph_read() function is called.

- The acid or base solution is dispensed into the pond water based on the value from the pH_read function and the required pH is maintained. The distilled water is pumped and flushed out 2 times for cleaning the pH sensor and the some distilled water is stored in the container of the pH sensor to prevent oxidization.

6. Food dispenser Function:

- The food dispenser function rotates the servo for dispensing the food.

7. Setup Function:

- Initializing serial communication, pin modes for motor control, and attaching the servo motor.

8. Loop Function:

- Contains the main program logic that continuously reads the time and accordingly calls the pH adjustment function and food dispenser function.

3.4 Working of the entire setup

The smart pond system is designed to automate the management of a pond environment by integrating pH level monitoring, pH adjustment, and food dispensing functionalities. The system is built around an Arduino Uno microcontroller board, which serves as the central processing unit. An RTC (Real-Time Clock) module is used to keep track of the current time, allowing users to specify the times for pH testing and food dispensing.

The pH monitoring functionality is implemented using a pH sensor, which measures the pH level of the pond water at a specific period of time specified by the user. The pH readings are displayed prominently on an 7 segment display, providing users with real-time information about the acidity or alkalinity of the water. The pH sensor is calibrated periodically to ensure the accuracy of the readings.

After reading the pH value of the sample pond water, the system initiates the pH adjustment process. Based on the predefined target pH value set by the user, the system calculates the amount of pH adjusting solution required and is automatically dispensed into the pond water through a pump controlled by a relay module, effectively adjusting the pH level to the desired value.

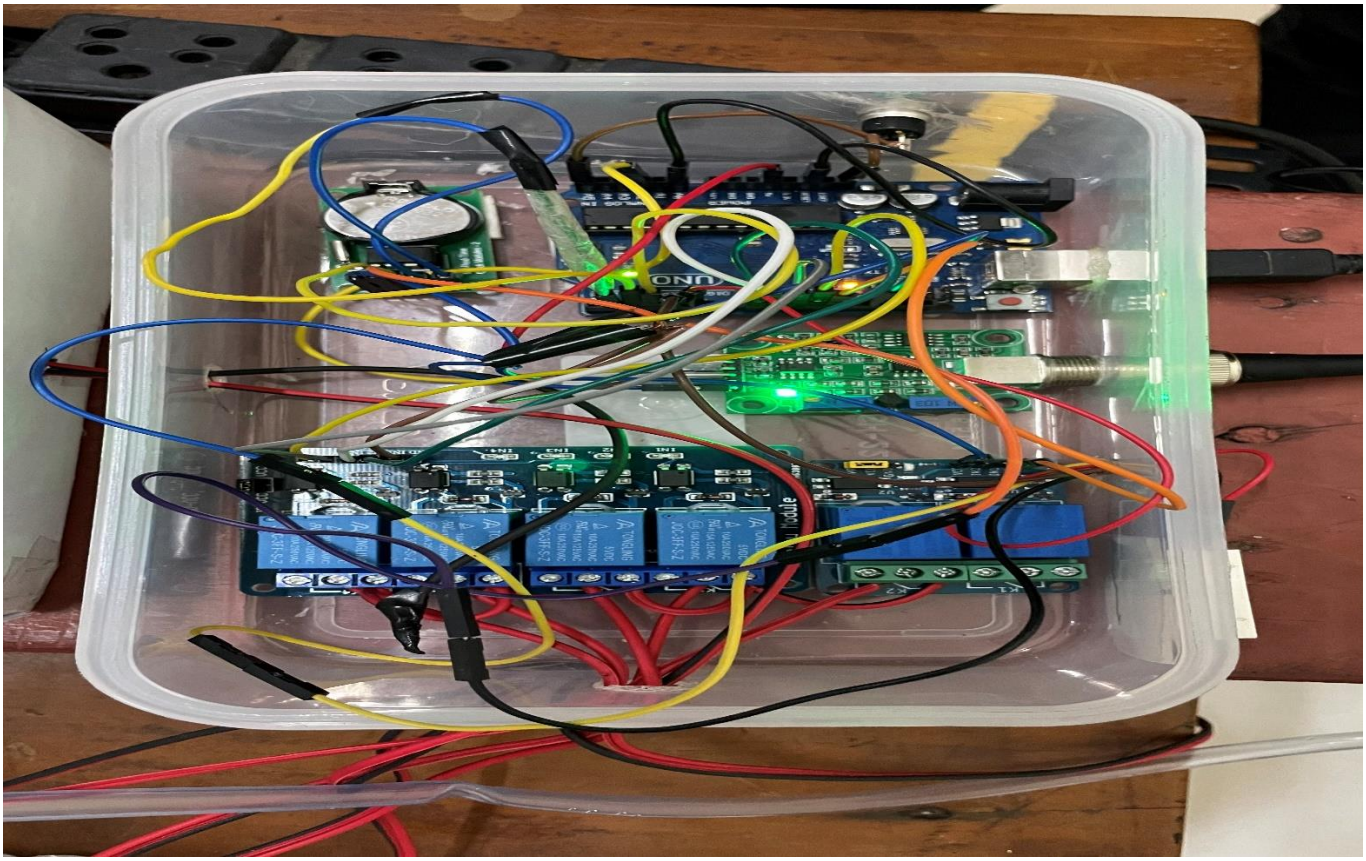
In addition to pH monitoring and adjustment, the smart pond system also includes a food dispensing mechanism. At the user-defined feeding times, a servo motor is activated to dispense the appropriate amount of fish food into the pond. The feeding schedule can be easily customized to accommodate the dietary needs of the pond inhabitants.

The integration of these functionalities into a single system provides users with a convenient and efficient way to maintain a healthy and balanced pond environment. By automating the monitoring and management processes, the smart pond system helps to ensure optimal water quality and nutrition for the pond inhabitants, promoting their health and well-being over the long term.

3.5 Images of the entire setup

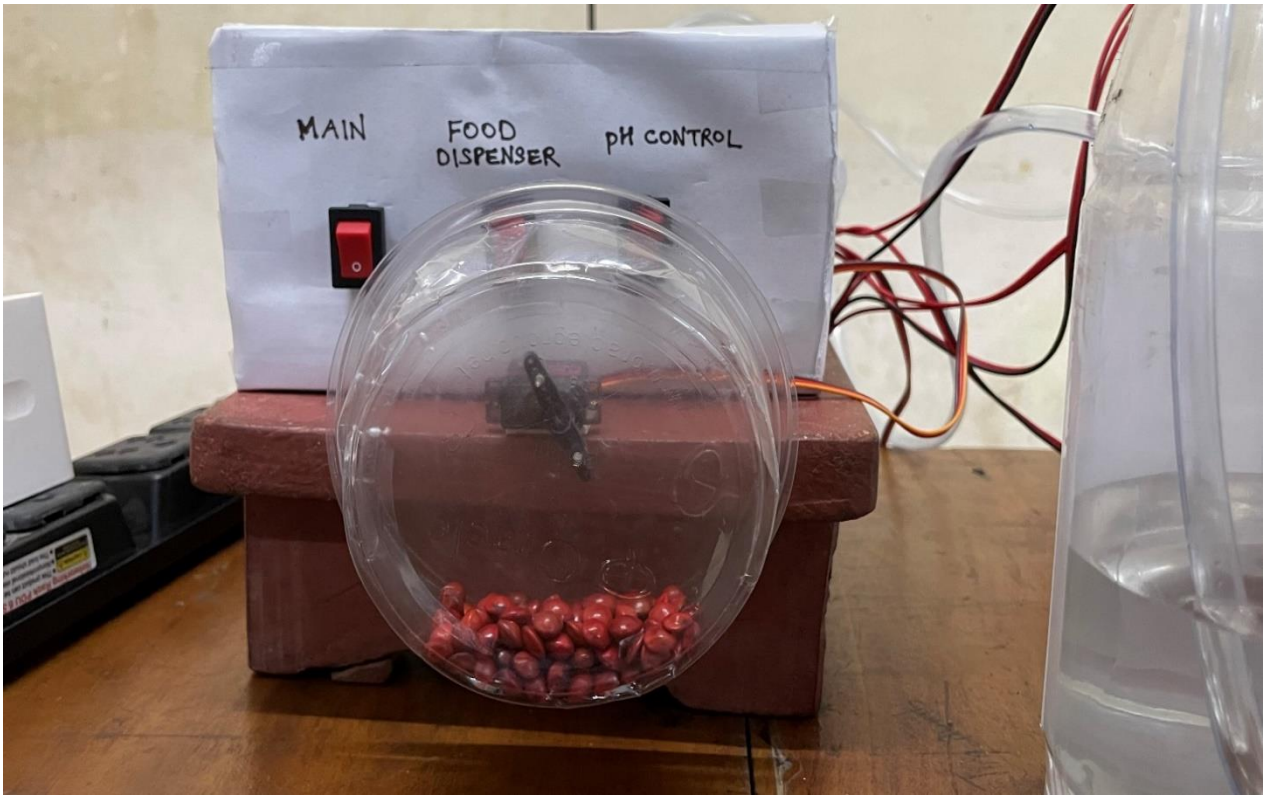
Containers

The containers have base, acid and distilled water. If the pH levels deviate from the optimal range for fish health, the system automatically activates pumps to replace the water with distilled water. This automated process ensures that the water maintains the required pH levels, creating an ideal environment for healthy fish growth and development.



Control box

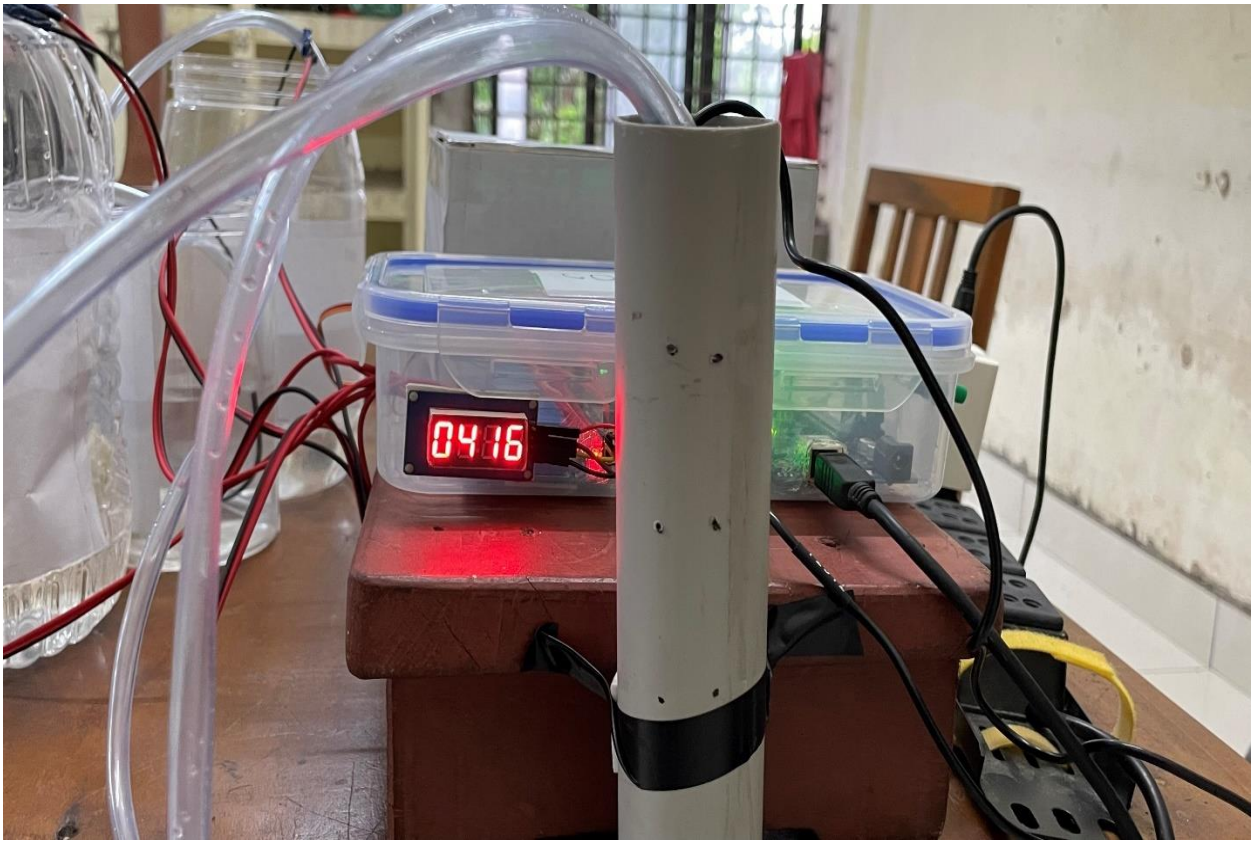
The control box has all the circuit components of the system. It has an Arduino Uno, 4 channel & 2 channel relay. An RTC module and a pH controller which connects the pH sensor and Arduino. All these components are connected using jumper wires accordingly. The circuit integrates an Arduino Uno microcontroller with a Real-Time Clock (RTC) module for precise timekeeping. It also includes a 4-channel and a 2-channel relay for controlling multiple d pumps. A pH sensor is incorporated to monitor the pH levels of the water in real-time. The Arduino Uno processes data from the pH sensor and activates the relays accordingly to maintain the optimal pH levels for the pisciculture system and also to dispense food.



Food dispenser

The food dispenser, integrated with a servo motor and Arduino Uno, is synchronized with an RTC module for precise timing. The system dispenses fish food at specified intervals, ensuring regular feeding cycles for the fish. The Arduino Uno receives time data from the RTC module and activates the servo motor to dispense food accordingly, maintaining a consistent feeding schedule for optimal pisciculture management.

The battery box features three 6V power supplies tailored to efficiently power an Arduino microcontroller, a servo motor for food dispensing, and pumps for water management in a pisciculture system. This setup ensures reliable and independent power sources for seamless operation, providing the necessary energy for the system's electronic and mechanical components.



pH sensor container

The pH sensor container is designed to house the pH sensor securely. It acts as a receptacle where sample water from the pisciculture system is pumped for pH measurement. The sample water from the pond is pumped using a mini pump into the container. After the pH measurement the sample water is drained out and distilled water is pumped into the container to neutralize the pH value and to prevent oxidation of electrode of pH sensor

The 7-segment display serves as a visual indicator, prominently showing the last checked pH value of the water in the pisciculture system.

CHAPTER 4

CONCLUSION

4.1 Addressing the Purpose

4.1.1 How do smart pond pisciculture systems improve water quality management?

Smart pond pisciculture systems employ sensors to monitor key water parameters such as pH, temperature, and oxygen levels in real-time. This data allows for timely adjustments and interventions to maintain optimal water quality conditions for fish health and growth.

4.1.2 What are the benefits of automated feeding schedules in smart pond pisciculture?

Automated feeding schedules ensure consistent and controlled feeding practices, reducing the risk of overfeeding or underfeeding. By optimizing feeding regimes based on fish growth stages and environmental conditions, these systems enhance feed utilization efficiency and minimize waste.

4.1.3 How do smart pond pisciculture systems contribute to sustainability?

Smart pond pisciculture systems promote sustainability through efficient resource utilization and environmental stewardship. By minimizing water and feed wastage, optimizing energy consumption, and reducing environmental impacts such as nutrient runoff, these systems help mitigate the ecological footprint of fish farming operations.

4.1.4 How can data analytics enhance decision-making in smart pond pisciculture?

Data analytics enables fish farmers to derive actionable insights from the vast amounts of data collected by smart pond pisciculture systems. By analyzing trends, patterns, and correlations in water quality data, feeding behavior, and fish health metrics, farmers can make informed decisions to optimize farm management practices and improve overall productivity.

4.2 Challenges Faced

While integrating the project we faced a couple of challenges as listed below:

- Fluctuations were noticed in the value measure by the pH sensor.
- The liquid (acid/base) could not be drained out fully from the pH sensor, it was solved by rinsing the sensor 2-3 times with distilled water.

4.3 Recommendations for Future Improvements

To further automate a ‘Smart Pond Pisciculture’ system there are several features that would be advisable to add a feature is that the system includes a water level sensor to monitor the pond's water level continuously. If the water level exceeds the prescribed upper limit, a drainage mechanism is activated to prevent overflow. Conversely, if the water level drops below the prescribed lower limit, a pumping mechanism is activated to replenish the water level. These actions are automated based on real-time data from the water level sensor, ensuring optimal water levels for fish health and pond stability.

REFERENCES

[1] https://how2electronics.com/

[2]https://www.flyrobo.in/

[3] https://robu.in

[4] https://www.electronicshub.org/

[5] https://www.elprocus.com/

[6] https://www.instructables.com/

[7] https://www.robotique.tech/

[8] https://www.youtube.com/

Appendix A. Program Code

```
 #include <Wire.h>
// #include <virtuabotixRTC.h>
#include <Servo.h>

Servo myservo;
// virtuabotixRTC myRTC(3, 2, 1);//clk,dat,rst
float time_nw,ph_nw;

float calibration_value = 32.64;
int phval = 0;
unsigned long int avgval;
int buffer_arr[10],temp;

int pos = 0;

const int p_motor=8;//pond
const int d_motor=9;//distilled
const int a_motor=10;//acid
const int b_motor=11;//base
const int out_motor=12;//drain
const int servo_pin=5;//servo pin
// connect ph pin to A0
```

```
const int tp=10000; //filling water from pond
const int tb=10000; //spraying base
const int ta=10000; //spraying base
const int tout=10000; //empty container
const int td=10000; //filling distilled water



void food_dispense()
{
  for(int i=0;i<10;i++)
  {
    for (pos = 0; pos <= 180; pos += 1) {
    myservo.write(pos);
    delay(15);
  }
  for (pos = 180; pos >= 0; pos -= 1) {
    myservo.write(pos);
    delay(15);
  }
  }
}
```

```
float ph_read()
{
 for(int i=0;i<10;i++)
 {
 buffer_arr[i]=analogRead(A0);
 delay(30);
 }
 for(int i=0;i<9;i++)
 {
 for(int j=i+1;j<10;j++)
 {
 if(buffer_arr[i]>buffer_arr[j])
 {
 temp=buffer_arr[i];
 buffer_arr[i]=buffer_arr[j];
 buffer_arr[j]=temp;
 }
 }
 }
 avgval=0;
 for(int i=2;i<8;i++)
```

```
 avgval+=buffer_arr[i];
 float volt=(float)avgval*5.0/1024/6;
 float ph_act = -5.70 * volt + calibration_value;

 return ph_act;
}

void ph_set()
{
  digitalWrite(out_motor, HIGH);
  delay(tout);
  digitalWrite(out_motor, LOW);
  delay(5000);
  digitalWrite(p_motor, HIGH);
  delay(tp);
  digitalWrite(p_motor, LOW);
  delay(30000);
  float present_ph=ph_read();
  if(present_ph<6.5)
  {
    digitalWrite(b_motor, HIGH);
    delay(tb);
```

```
  digitalWrite(b_motor, LOW);
 }
 else if(present_ph>8)
 {
  digitalWrite(a_motor, HIGH);
  delay(ta);
  digitalWrite(a_motor, LOW);
 }

 digitalWrite(out_motor, HIGH);
 delay(tout);
 digitalWrite(out_motor, LOW);
 delay(1000);
 digitalWrite(d_motor, HIGH);
 delay(td);
 digitalWrite(d_motor, LOW);
 delay(1000);
 digitalWrite(out_motor, HIGH);
 delay(tout);
 digitalWrite(out_motor, LOW);
 delay(1000);
 digitalWrite(d_motor, HIGH);
```

```
  delay(td);
  digitalWrite(d_motor, LOW);
  delay(5000);
}

void setup()
{
  Serial.begin(9600);

  // myRTC.setDS1302Time(0, 52, 17, 3, 27, 3, 2024);
  pinMode(p_motor, OUTPUT);
  pinMode(d_motor, OUTPUT);
  pinMode(a_motor, OUTPUT);
  pinMode(b_motor, OUTPUT);
  pinMode(out_motor, OUTPUT);

  myservo.attach(servo_pin);

}

void loop()
```

```
{
 // myRTC.updateTime();
 // time_nw=myRTC.hours*100+myRTC.minutes+myRTC.seconds*.01;

 if(time_nw==300.10)
 {
  ph_set();
 }
 else if(time_nw==900.10)
 {
  food_dispense();
 }
 else if(time_nw==1200.20)
 {
  ph_set();
 }
 else if(time_nw==1500.20)
 {
  ph_set();
 }
 else if(time_nw==1800.20)
 {
```

```
   ph_set();
  }
  else if(time_nw==2100.20)
  {
    food_dispense();
  }
  else if(time_nw==0.20)
  {
    ph_set();
  }


  delay(1000);
}
```