AUTOMATIC FISH FEEDER USING IOT DEVICES AND ARDUINO

A MINI PROJECT REPORT

21CSE253T – INTERNET OF THINGS

Submitted by

PREM LOHIA (RA2211029010007)

ANITEJ MISHRA (RA2211029010023)

Under the guidance of

Dr C. Fancy

Assistant Professor, Department of Networking and Communications

In partial satisfaction of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

with specialization in Computer Networking

SRM INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University u/s 3 of UGC Act, 1956)

DEPARTMENT OF NETWORKING AND COMMUNICATIONS

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR-603 203

MAY 2024

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR-603 203

BONAFIDE CERTIFICATE

Certified that this Project Report titled "**AUTOMATIC FISH FEEDER USING IOT DEVICES AND ARDUINO**" is the bonafide work done by:

PREM LOHIA (RA2211029010007)

ANITEJ MISHRA (RA2211029010023)

who completed the project under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE	**SIGNATURE**
Dr C. Fancy	Dr Annapurani Panaiyappan
IOT-Course Faculty	**Head of the Department**
Assistant Professor	Department of Networking and Communications
Department of Networking and Communications	SRMIST
SRMIST	

TABLE OF CONTENTS

ABSTRACT

This project presents the design and implementation of an IoT-based automatic fish feeder system. The system leverages Internet of Things (IoT) technology to automate the feeding process for aquarium fish. By integrating sensors and a microcontroller with cloud-based services, we can even use a mobile app to control the feeder remotely, but our project is a simplified, mainly offline implementation. This paper outlines the methodology used in developing this system, presents experimental results, and discusses the feasibility and benefits of such a solution.

Chapter 1

INTRODUCTION

Aquariums are a popular hobby worldwide, offering a serene and captivating glimpse into aquatic ecosystems. However, maintaining a healthy environment for fish requires careful attention to various factors, including water quality and feeding schedules. One essential aspect of fish care is ensuring they receive adequate and timely nutrition. Traditional methods of feeding, which rely on manual intervention, can be prone to human error and may not provide consistent feeding patterns, especially for busy aquarium owners.

To address these challenges, the integration of Internet of Things (IoT) technology with fish care routines has emerged as a promising solution. IoT enables the development of automated systems that can remotely monitor and control devices over the internet. By applying IoT principles to aquarium management, we can create smart solutions that enhance the efficiency and convenience of fish care tasks.

The focus of this project is to design and implement an IoT-based automatic fish feeder system. The primary goal is to develop a feeder that can dispense fish food automatically based on predefined schedules or on-demand commands from users. This system aims to alleviate the manual effort required for regular fish feeding while ensuring precise and timely nutrition delivery.

The automatic fish feeder will leverage microcontroller technology, sensors, and cloud-based services to achieve its functionality. The microcontroller will serve as the central processing unit, interfacing with sensors to monitor environmental parameters such as water quality and tank conditions. Our project allows the user to control feeding schedules and monitor feeding activity in real-time.

Chapter 2

LITERATURE SURVEY

1. **“Internet of Things”**
   Author: Iresh A. Dhotre
   Edition: First
   Publication Year: 2020
   Publisher: Technical Publications
   We referred to this book to understand the basic concepts of Internet of Things, microcontrollers and how to apply them in our project to make an automatic fish feeder and also use them in our advantage.
2. **“Knowledge Based Real Time Monitoring System for Aquaculture using IoT”**
   Authors: K. Raju, G. Verma
   Publication Year: 2017
   We used this research paper to help us understand why fishes need to fed properly and periodically. It also helped us in identifying the ideal solutions of water content for fishes.

3. **“Design and Construction of Microcontroller-Based Automatic Fish Feeding Device”**
   Authors: E. N. Onwuka, A. O. Adejo, I. U. Joseph
   Publication Year: 2012
   We referred to this research paper to see the various different types of automatic fish feeders and how they are made using a variety of different components.

4. **“Developing Fish Feeder System using Raspberry Pi”**
   Authors: Hidayatul Nur binti Hasim, Mritha Ramalingam, F. Ernawan
   Publication Year: 2017
   We referred to this research paper to see how we can use Raspberry Pi instead of Arduino Uno, in case we couldn’t source or buy one.

Chapter 3

SYSTEM ARCHITECTURE AND DESIGN

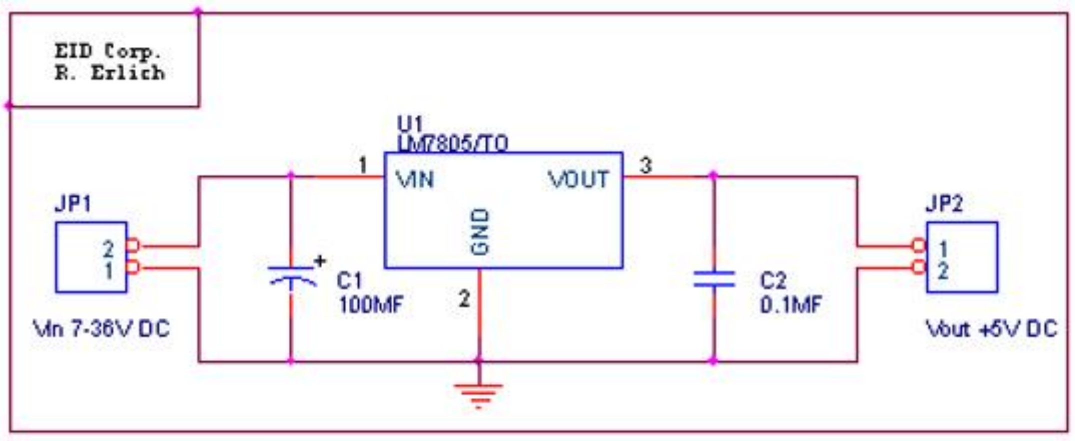
The main components of our Automatic Fish Feeder are given below:

- Power Supply
- Arduino Uno
- ATMEGA328-PU Microcontroller
- Real Time Clock (RTC) Module
- Servo Motor
- Node MCU-ESP8266
- Cables and Jumper Wires for Connections
- Buttons for Controlling

CIRCUIT DESCRIPTION

This design is a modest +5V power supply for digital electronics experiments. All electronics stores and supermarkets sell small, affordable wall transformers with variable output voltage. These transformers are readily available, but their voltage control is weak, making them unsuitable for digital circuit experimenters until they can be improved. This circuit solves the problem.

This design can output +5V at 150 mA, however effective cooling on the 7805 regulator chip can increase it to 1 A. The circuit is overloaded and thermally protected.



Circuit diagram of the power supply

The capacitors must have enough high voltage rating to safely handle the input voltage feed to circuit. The circuit is very easy to build for example into a piece of Vero board.

Pinout of the 7805 regulator IC

- Unregulated voltage in
- Ground
- Regulated voltage out

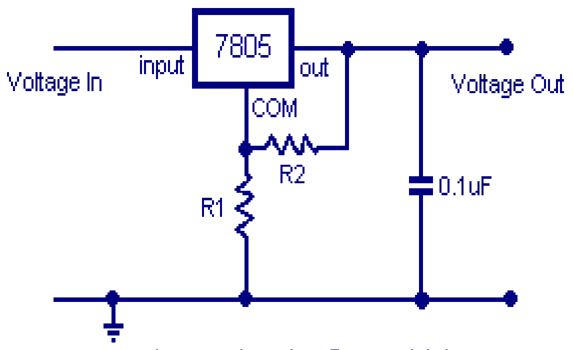
Component list

- 7805 regulator IC
- 100 uF electrolytic capacitor, at least 25V voltage rating
- 10 uF electrolytic capacitor, at least 6V voltage rating
- 100 nF ceramic or polyester capacitor

More output current

If you need more than 150 mA of output current, you can update the output current up to 1A doing the following modifications:

- Change the transformer from where you take the power to the circuit to a model which can give as much current as you need from output
- Put a heat sink to the 7805 regulator (so big that it does not overheat because of the extra losses in the regulator)



Increasing the Output Voltage

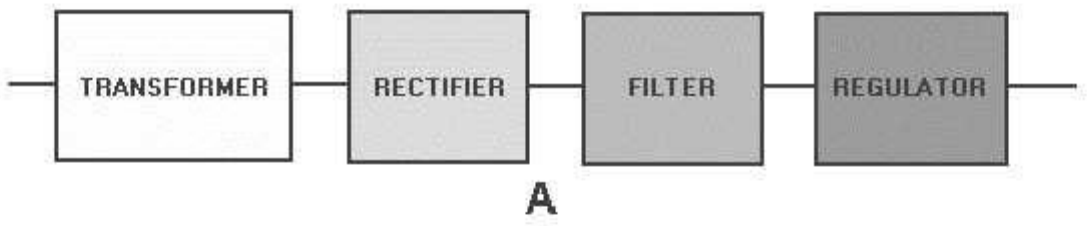
Other output voltages

If you need other voltages than +5V, you can modify the circuit by replacing the 7805 chips with another regulator with different output voltage from regulator 78xx chip family. The last numbers in the chip code tells the output voltage.

Remember that the input voltage must be at least 3V greater than regulator output voltage to otherwise the regulator does not work well.

POWER SUPPLY

A power supply (sometimes known as a power supply unit or PSU) is a device or system that supplies electrical or other types of energy to an output load or group of loads. The term is most commonly applied to electrical energy supplies, less often to mechanical ones, and rarely to others.



Block diagram of a basic power supply

The transformer raises or lowers the input line voltage and isolates the power supply. RECTIFIER converts alternating current input signal to pulsating direct current. However, this chapter will show that pulsing dc is undesirable. Therefore, a FILTER section converts pulsing dc to a purer, better dc voltage.

Final part, REGULATOR, does what it says. It keeps power supply output consistent despite heavy load current or input line voltage variations. Let's trace an ac signal through the power supply now that you know what each part accomplishes. You must now observe how each power supply section changes this signal. See how these modifications occur later in the chapter. In view B of figure 4-1, the transformer primary receives 115 volts ac. The transformer is a 1:3 step-up transformer. To compute the output for this transformer, multiply the input voltage by the ratio of turns in the primary and secondary. For example, 115 volts ac ′ 3 = 345 volts ac (peak-to-peak) at the output. Each rectifier diode carries 180 degrees of the 360-degree input, therefore the output is half, or 173 volts, of pulsing dc. The filter portion, a network of resistors, capacitors, or inductors, controls the signal's rise and fall time, keeping it at a constant dc level. Discussion of filter circuits will clarify the filter process. The filter outputs 110 volts dc with ac ripple. The lower average voltage will be discussed later in this chapter. Electronic equipment (the load) uses the regulator's 110-volt dc output.

Simple 5V power supply for digital circuits

- Brief description of operation: Gives out well regulated +5V output, output current capability of 100 mA
- Circuit protection: Built-in overheating protection shuts down output when regulator IC gets too hot
- Circuit complexity: Very simple and easy to build
- Circuit performance: Very stable +5V output voltage, reliable operation
- Availability of components: Easy to get, uses only very common basic components
- Design testing: Based on datasheet example circuit, I have used this circuit successfully as part of many electronics projects
- Applications: Part of electronics devices, small laboratory power supply
- Power supply voltage: Unregulated DC 8-18V power supply
- Power supply current: Needed output current + 5 mA

ARDUINO UNO

Arduino is an open-source project that created microcontroller-based kits for building digital devices and interactive objects that can sense and control physical devices. The project is based on microcontroller board designs, produced by several vendors, using various microcontrollers. These systems provide sets of digital and analog input/output (I/O) pins that can interface to various expansion boards (termed shields) and other circuits. The boards feature serial communication interfaces, including Universal Serial Bus (USB) on some models, for loading programs from personal computers. For programming the microcontrollers, the Arduino project provides an integrated development environment (IDE) based on a programming language named Processing, which also supports the languages C and C++.

PRODUCT DESCRIPTION

Arduino Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, 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. Arduino Uno has a number of facilities for communicating with a computer, another Arduino board, or other microcontrollers.

Arduino UNO

ATMEGA328P-PU microcontroller

The most important element in Arduino Uno R3 is ATMEGA328P-PU is an 8-bit Microcontroller with flash memory reach to 32k bytes. It's features as follow:

- High Performance, Low Power AVR
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Up to 20 MIPS Throughput at 20 MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 4/8/16/32K Bytes of In-System Self-Programmable Flash program memory
  - 256/512/512/1K Bytes EEPROM

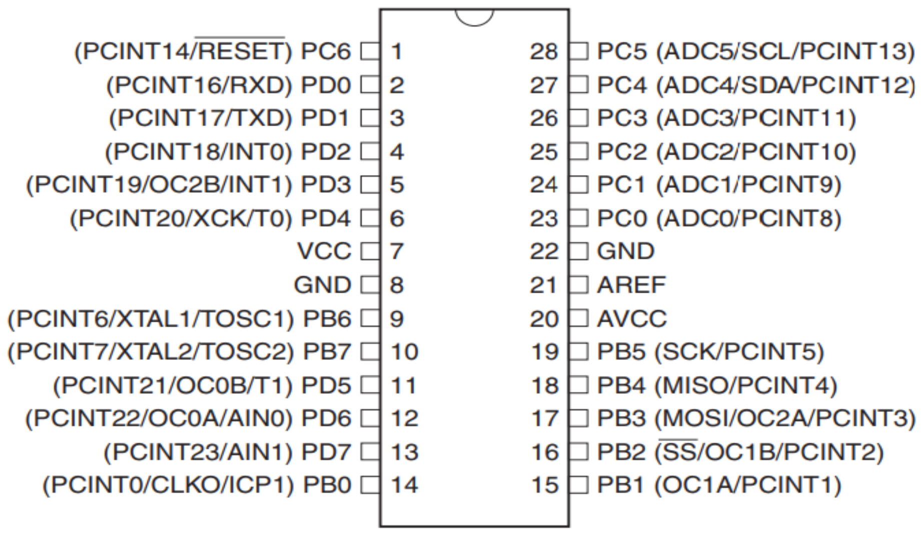
- 512/1K/1K/2K Bytes Internal SRAM
- Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
- Data retention: 20 years at 85°C/100 years at 25°C
- Optional Boot Code Section with Independent Lock Bits
- In-System Programming by On-chip Boot Program
- True Read-While-Write Operation
- Programming Lock for Software Security

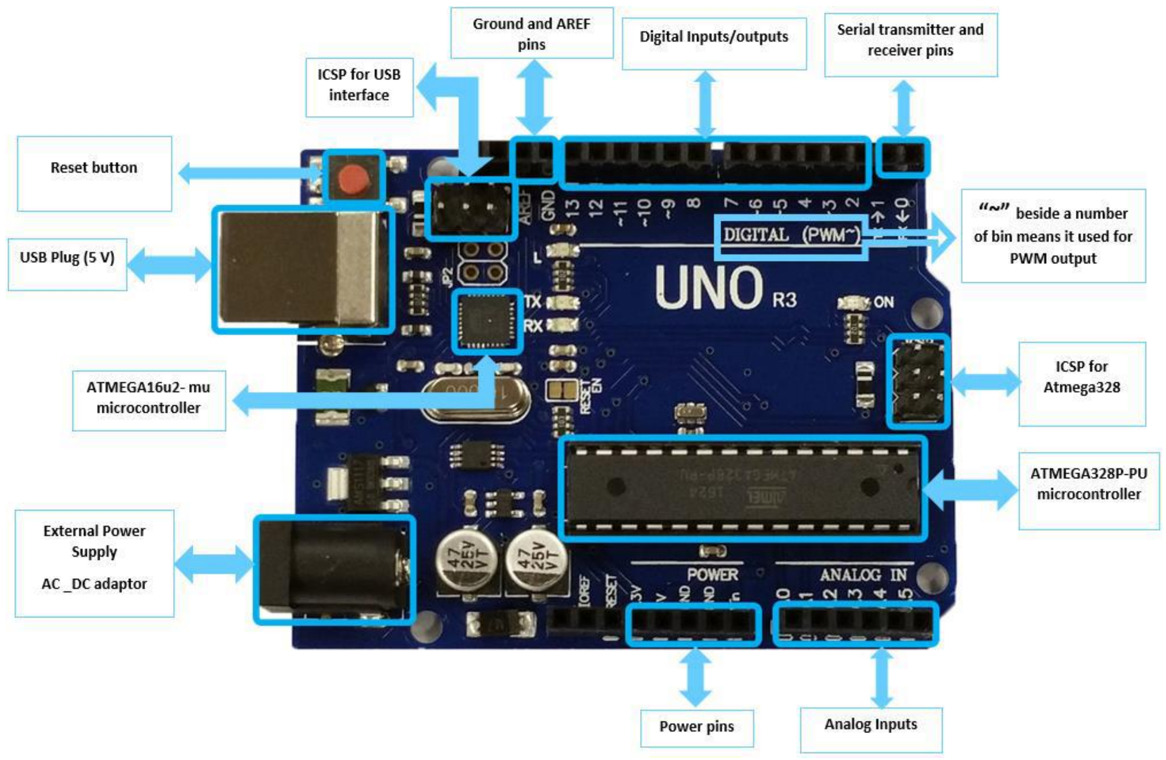
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - 8-channel 10-bit ADC in TQFP and QFN/MLF package
  - Temperature Measurement
  - 6-channel 10-bit ADC in PDIP Package
  - Temperature Measurement
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Byte-oriented 2-wire Serial Interface (Philips I2 C compatible)
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator

- External and Internal Interrupt Sources
- Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby

- I/O and Packages
  - 23 Programmable I/O Lines
  - 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
- Operating Voltage
  - 1.8 - 5.5V
- Temperature Range
  - -40°C to 85°C
- Speed Grade
  - 0 - 4 MHz@1.8 - 5.5V, 0 - 10 MHz@2.7 - 5.5.V, 0 - 20 MHz @ 4.5 - 5.5V
- Power Consumption at 1 MHz, 1.8V, 25°C
  - Active Mode: 0.2 mA
  - Power-down Mode: 0.1 µA
  - Power-save Mode: 0.75 µA (Including 32 kHz RTC)

PIN CONFIGURATION





Ground and AREF pins
Digital Inputs/outputs
Serial transmitter and receiver pins
ICSP for USB interface
Reset button
"~" beside a number of bin means it used for PWM output
USB Plug (5 V)
ICSP for Atmega328
ATMEGA16u2- mu microcontroller
ATMEGA328P-PU microcontroller
External Power Supply AC _DC adaptor
Power pins
Analog Inputs
DIGITAL (PWM~)
UNO R3
POWER
ANALOG IN

FEATURES

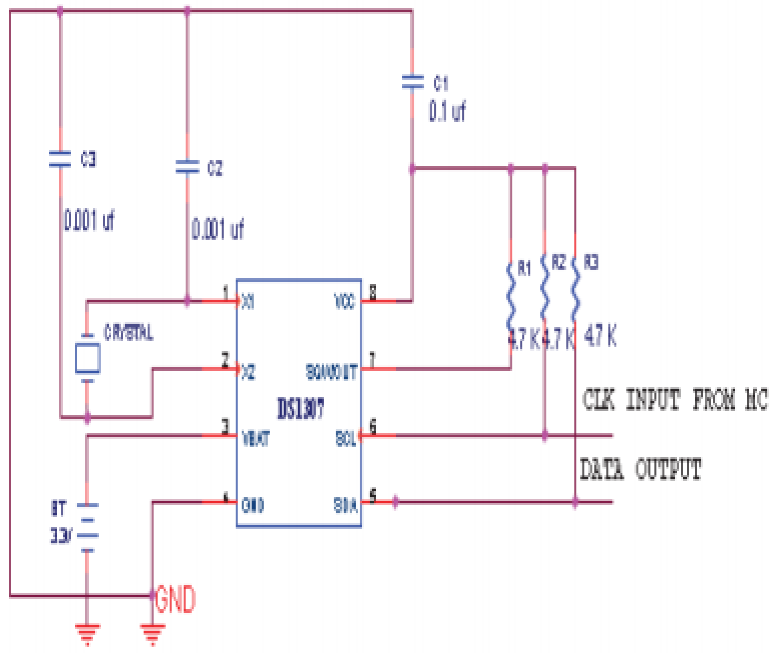
- Microcontroller: ATmega328P
- Operating voltage: 5V
- Input voltage: 7-12V
- Flash memory: 32KB
- SRAM: 2KB
- EEPROM: 1KB

APPLICATIONS

- Real time biometrics
- Robotic applications
- Academic applications

RTC

The Real Time Clock (RTC) is a prime component used to allow digital system to continuously keep track of time relative to human perception. They typically operate at slower speeds and consume much less power (500nA with oscillator running) than a general purpose clock. RTC requires a small portable supply using a battery (Li-ion battery-3.3V) when the rest of the system is switched off. The circuit is shown in figure 3. The other benefits of RTC are Low power consumption (important when running from alternate power), Frees the main system for time-critical tasks and more accurate than other methods.



RTC (DS3231)

The DS3231 I2C precision clock module is suitable for projects that need accurate timekeeping capabilities. Typical applications include clocks and data logging devices.

Overview

DS3231 is a low cost and extremely accurate I2C real-time clock, with an integrated crystal and temperature-compensated crystal oscillator (TCXO). DS3231 can be operated using supply voltages ranging from 2.3 V to 5.5 V and it also features battery backup capabilities.

The DS3231 features an integrated crystal, 2 programmable time-of-day alarms, a temperature sensor, and 32.768 kHz signal output pin.

The AT24C32 EEPROM is a 32K EEPROM that can be used to add non-volatile data storage to your electronic projects and prototypes.

The module also features a battery holder, allowing you to add a backup battery to ensure continuous operation.

Features

- Can be connected directly to the microcontroller IO ports
- Standard 2.54 mm pins for input and output connections
- Two calendars and alarm clock
- Two programmable square-wave outputs
- Real time clock generator for seconds, minutes, hours, day, date, month, and year timing
- Valid until 2100 with leap year compensation
- Can be cascaded with other I2C devices
- The address can be set using the pins A0/A1/A2 (the default address is 0x57)
- Battery socket compatible with LIR2032 batteries
- I2C interface

Specifications

- Operating voltage: 3.3 V to 5.5 V
- Real-time clock chip: DS3231
- Clock accuracy: 2 ppm
- Memory chip: AT24C32 (32 Kb storage capacity)
- On-chip temperature sensor with an accuracy of ±3 °C
- I2C bus interface maximum speed: 400 kHz
- Size: 38 x 22 x 14 mm

SERVO MOTOR

Principle of Working: Servo motor works on the PWM (Pulse Width Modulation) principle, which means its angle of rotation, is controlled by the duration of pulse applied to its control PIN. Basically servo motor is made up of DC motor which is controlled by a variable resistor (potentiometer) and some gears

DESCRIPTION

The Tower-Pro MG-995 metal-gear servo offers exceptional torque and reliability, making it an ideal choice for robotics enthusiasts and beginners alike. With a torque rating of 10 kg*cm and metal gears for added durability, this servo is capable of rotating at least 120 degrees (60 in each direction) using a standard 1.5-2.5ms pulse. For extended rotation up to approximately 170 degrees, pulse lengths can be adjusted, although this may vary slightly between individual servos. Included with the servo are plastic horns for convenient attachment, simplifying the process of integrating it into projects. To control the MG-995 with an Arduino, connect the orange control wire to pin 9 or 10 and utilize the Arduino IDE's Servo library. The servo's middle position corresponds to a 1.5ms pulse, while full right and left positions correspond to ~2ms and ~1ms pulses, respectively. Custom pulse lengths may be necessary to achieve a full 180 degrees of motion, typically ranging from 0.75ms to 2.25ms. Caution is advised when experimenting with pulse lengths beyond the default range to avoid damaging the servo. With its versatility and ease of control, the Tower-Pro MG-995 servo is well-suited for a wide range of robotics and automation applications.

TECHNICAL DETAILS

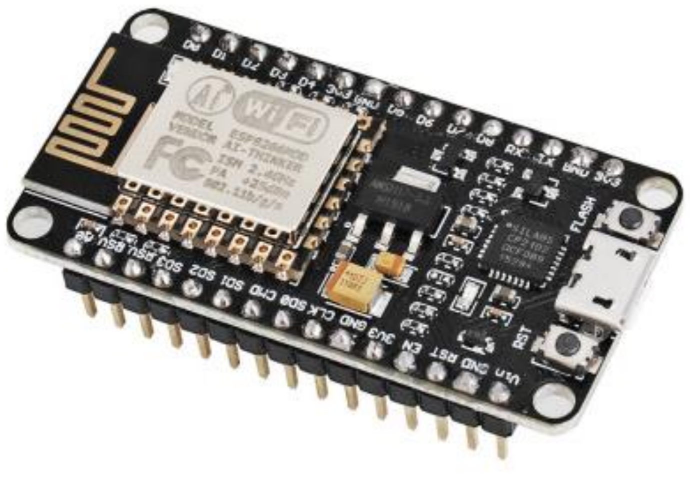
- Power: 4.8V - 6V DC max (5V works well)

- Average Speed: 60 degrees in 0.20 sec (@ 4.8V), 60 degrees in 0.16 sec (@ 6.0V)
- Weight: 62.41g
- Torque: At 4.8V: 8.5 kg-cm / 120 oz-in, and at 6V: 10 kg-cm / 140 oz-in.
- Size mm: (L x W x H) 40.7 x 19.7 x 42.9
- Spline Count: 25

NODEMCU - ESP8266

The ESP8266 is the brains behind the flexible Wi-Fi development board known as the NodeMCU. This open-source platform was created by a group of dedicated individuals with the goal of making IoT (Internet of Things) and Wi-Fi-related applications more affordable and accessible. The NodeMCU, which has Wi-Fi built in, features the CP2102 IC for USB-to-Serial communication and is made to make developing embedded applications simple.

Product Description: The ESP8266 is a microcontroller with built-in Wi-Fi capabilities, and the NodeMCU development board was made to take advantage of this. As a result, it's a great option for everything from straightforward Wi-Fi-controlled gadgets to sophisticated Internet of Things (IoT) implementations. Thanks to the CP2102 IC, the board can communicate with a computer over a USB cable with relative ease. It also includes support for the Arduino IDE and other development environments for programming.



FEATURES

- ESP8266-Based: Powered by the ESP8266 microcontroller known for its low cost and Wi-Fi capabilities.

- CP2102 USB-to-Serial Chip: Enables easy communication with your computer through USB.
- Arduino IDE Support: Program the NodeMCU using the popular Arduino development environment.
- GPIO Pins: Multiple GPIO pins for digital input/output, analog input, and various communication protocols.
- Wi-Fi Connectivity: Built-in Wi-Fi for seamless IoT and wireless applications.
- Power Supply: USB or external power source, with a stable 3.3V supply for the ESP8266.
- Open Source: Extensive documentation and community support for open-source development.
- Compact Design: A compact form factor for various project requirements.

SPECIFICATIONS

- Microcontroller: ESP8266
- USB-to-Serial IC: CP2102
- Programming: Arduino IDE, PlatformIO, etc.
- GPIO Pins: Multiple digital and analog pins
- Wi-Fi: Built-in Wi-Fi for wireless connectivity
- Power Supply: USB or external 3.3V supply
- Dimensions: Compact design for various applications

APPLICATIONS

- IoT Projects: Ideal for a wide range of Internet of Things applications.
- Home Automation: Control and monitor devices remotely over Wi-Fi.
- Sensor Networks: Collect and transmit sensor data wirelessly.
- Wi-Fi-Controlled Devices: Create smart and connected devices.
- Prototyping: Fast and efficient prototyping for Wi-Fi-enabled projects.
- Educational Projects: Learn about Wi-Fi and IoT development.

SOFTWARE DESCRIPTION

ARDUINO IDE

The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It is used to write and upload programs to Arduino board.

The source code for the IDE is released under the GNU General Public License, version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

Arduino is an open-source hardware and software company, project and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices and interactive objects that can sense and control objects in the physical and digital world. Its products are licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially in preassembled form or as do-it-yourself (DIY) kits.

Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards or breadboards (shields) and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs from personal computers. The microcontrollers are typically programmed using a dialect of features from the programming languages C and C++. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) based on the Processing language project.

The Arduino project started in 2003 as a program for students at the Interaction Design Institute Ivrea in Ivrea, Italy, aiming to provide a low-cost and easy way for novices and professionals to create devices that interact with their environment using sensors and actuators. Common examples of such devices intended for beginner hobbyists include simple robots, thermostats and motion detectors.

The name Arduino comes from a bar in Ivrea, Italy, where some of the founders of the project used to meet. The bar was named after Arduin of Ivrea, who was the margrave of the March of Ivrea and King of Italy from 1002 to 1014.

EMBEDDED C

Embedded C is a set of language extensions for the C programming language by the C Standards Committee to address commonality issues that exist between C extensions for different embedded systems.

Chapter 4

METHODOLOGY

The automatic fish feeder system comprises several key components, including a microcontroller (such as Arduino or Raspberry Pi), feeding mechanism, sensors (e.g., moisture sensor, load cell), and connectivity modules (Wi-Fi or Bluetooth). The microcontroller is programmed to interact with the sensors to monitor environmental conditions and trigger feeding actions. A mobile app communicates with the cloud-based server, allowing users to schedule feeding times, monitor feeding activity, and receive notifications. The system's architecture and communication protocols will be elaborated upon in this section.

Chapter 5

CODING

Code in Arduino IDE

```
#include <DS3231.h>
#include <LiquidCrystal.h>
#include <EEPROM.h>
#include <Servo.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
DS3231  rtc(SDA, SCL);
Servo servo_test;
int countToZero = 0;
int angle = 0;
int sw2 = 3;
int sw3 = 2;
int buzz = 10;
int flag = 0;
int val = 0;
int val1 = 0;
int val2 = 0;
int count = 0;
int count1 = 0;
int count2 = 0;
int count3 = 0;
int count4 = 0;
int tempcount = 0;
String submin = "";
int mins = 0;
int flag1 = 0;
void setup()
{
  Serial.begin(9600);
  lcd.init();
  lcd.init();
  lcd.backlight();
  rtc.begin();
  pinMode(sw2, INPUT_PULLUP);
  pinMode(sw3, INPUT_PULLUP);
  pinMode(buzz, OUTPUT);
  lcd.setCursor(0, 0);
  lcd.print("   WELCOME     ");
  delay(1000);
  lcd.setCursor(0, 0);
  lcd.print("    AUTOMATIC    ");
  lcd.setCursor(0, 1);
  lcd.print("  FISH FEEDING  ");
  delay(1000);
  lcd.clear();
  //rtc.setTime(19, 16, 0);
  //rtc.setDate(11, 28, 2019);
  servo_test.attach(6);
}
```

```
void loop()
{
  lcd.setCursor(0, 0);
  lcd.print("Time:");
  lcd.print(rtc.getTimeStr());
  lcd.setCursor(14, 0);
  lcd.print(tempcount);
  lcd.setCursor(0, 1);
  lcd.print(count1);
  lcd.setCursor(4, 1);
  lcd.print(count2);
  lcd.setCursor(8, 1);
  lcd.print(count3);
  lcd.setCursor(12, 1);
  lcd.print(count4);
  submin = rtc.getTimeStr();
  submin = submin.substring(3, 5);
  mins = submin.toInt();
  flag = 0;
  if (mins == count1 || mins == count2 || mins == count3 || mins == count4)
  {
    if (flag1 == 0)
    {
      digitalWrite(buzz, HIGH);
      delay(500);
      digitalWrite(buzz, LOW);
    }
    servo_test.attach(6);
    lcd.setCursor(14, 1);
    lcd.print("FD");
    for (angle = 0; angle < 90; angle += 1)
    {
      servo_test.write(angle);
      delay(5);
    }
    for (angle = 90; angle >= 1; angle -= 1)
    {
      servo_test.write(angle);
      delay(5);
    }
    flag1 = 1;
    countToZero = mins == count1 ? 1 : (mins == count2 ? 2 : (mins == count3 ?
3 : (mins == count4 ? 4 : 0)));
    gsm(countToZero);
  }
  else
  {
    lcd.setCursor(14, 1);
    lcd.print("--");
    if (flag1 == 1)
    {
      digitalWrite(buzz, LOW);
      servo_test.detach();
      flag1 = 0;
      countToZero = mins == count1 ? 1 : (mins == count2 ? 2 : (mins == count3
? 3 : (mins == count4 ? 4 : 0)));
      gsm(countToZero);
      delay(50);
    }
  }
  val = digitalRead(sw3);
  if (val == LOW)
  {
    lcd.clear();
    while (1)
    {
      if (flag == 1)
        break;
      lcd.setCursor(0, 0);
      lcd.print("Select Memory ");
      val = digitalRead(sw3);
      if (val == LOW)
      {
        if (count > 3)
        {
          count = 0;
        }
        count++;
        lcd.setCursor(0, 1);
        lcd.print(count);
        delay(200);
      }
```

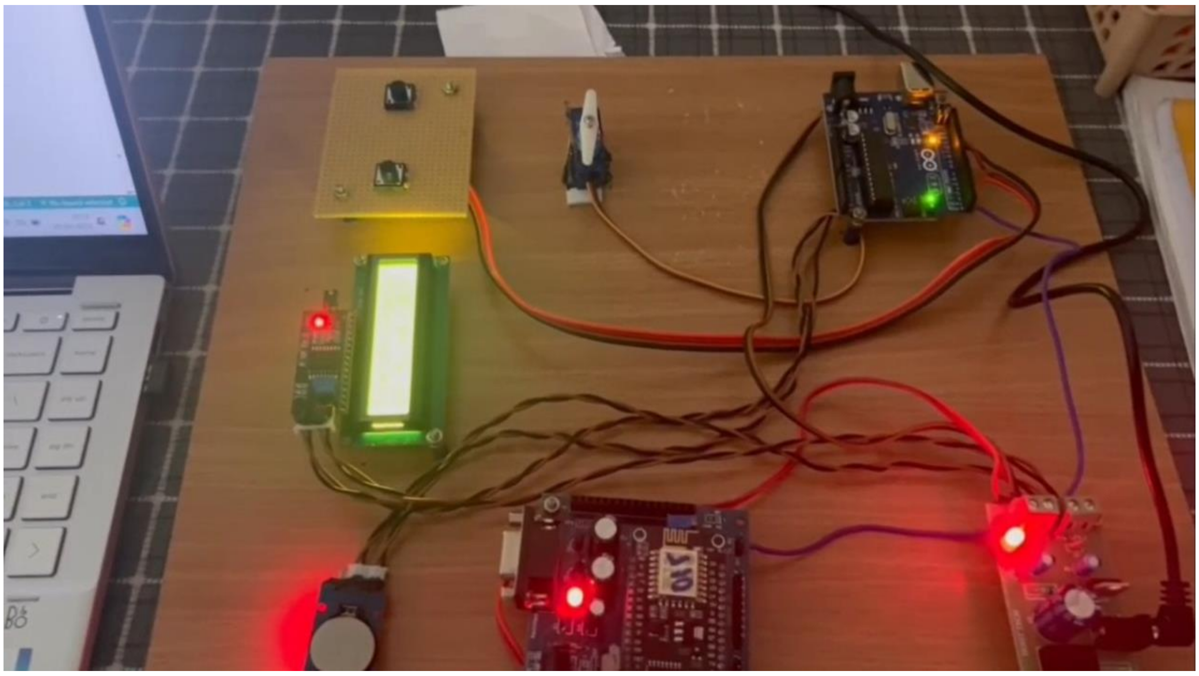
```
val1 = digitalRead(sw2);
if (val1 == LOW)
{
  lcd.clear();
  delay(200);
  while (1)
  {
    lcd.setCursor(0, 0);
    lcd.print("Select Mins ");
    val2 = digitalRead(sw3);
    if (val2 == LOW)
    {
      tempcount++;
      if (tempcount > 59)
      {
        tempcount = 0;
        lcd.setCursor(0, 1);
        lcd.print("  ");
      }
      lcd.setCursor(0, 1);
      lcd.print(tempcount);
      delay(200);
    }
    val1 = digitalRead(sw2);
    if (val1 == LOW)
    {
      delay(200);
```

```
            if (count == 1)
            {
              count1 = tempcount;
            }
            if (count == 2)
            {
              count2 = tempcount;
            }
            if (count == 3)
            {
              count3 = tempcount;
            }
            if (count == 4)
            {
              count4 = tempcount;
            }
            lcd.clear();
            delay(500);
            flag = 1;
            break;
          }
        }
        delay(200);
      }
    }
  }
}
```

```
void gsm(int countToZero)
{
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print(" SENDING    ");
  lcd.setCursor(0, 1);
  lcd.print("    IOT >>> ");
  Serial.print("*");
  Serial.print("Food Feeded");
  Serial.print("#");
  delay(500);
  // Zero out the specified count
  if (countToZero == 1) {
    count1 = 0;
  } else if (countToZero == 2)
{
    count2 = 0;
  } else if (countToZero == 3)
 {
    count3 = 0;
  } else if (countToZero == 4)
{
    count4 = 0;
  }
  lcd.clear();
}
```

Chapter 6

RESULTS



In the experimentation phase, the automatic fish feeder prototype will be constructed and tested under various conditions. The system's performance in accurately dispensing food, its responsiveness to user commands, and its reliability in different environments (e.g., freshwater vs. saltwater tanks) will be evaluated. Data collected from the experiments will be analyzed to assess the system's effectiveness and identify areas for improvement. Results will include quantitative measurements and user feedback.

The working of the Automatic Fish Feeder can be seen in this video here:

https://1drv.ms/v/c/d4fb62a7e0f2659d/EebQANRlL4pMsPIb9J1_ttUB6_DCKvSGMuA2mGSiBrYzmg?e=YoU81c

Scan this QR code to see the video:

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENT

The development of an IoT-based automatic fish feeder system marks a significant step forward in the realm of aquarium automation and fish care technology. This project has demonstrated the practicality and effectiveness of leveraging IoT components, including sensors, microcontrollers, and cloud-based connectivity, to create a smart feeding solution for aquarium enthusiasts. The automated feeding schedule, controlled remotely via a mobile application, ensures that fish receive consistent and appropriate nourishment, irrespective of the owner's presence. This not only simplifies fish care routines but also contributes to maintaining optimal health and well-being within the aquarium environment.

Looking ahead, several potential enhancements could further elevate the capabilities and versatility of such systems. Firstly, integrating water quality sensors into the feeder would enable real-time monitoring of crucial parameters like pH and ammonia levels, allowing for adaptive feeding schedules based on environmental conditions. Additionally, implementing behaviour-based feeding algorithms could optimize feeding patterns to mimic natural feeding behaviours, promoting healthier fish growth and reducing waste. Scaling up the system for use in commercial aquaculture settings could streamline feeding processes in large-scale fish farms, improving efficiency and reducing operational costs.

Furthermore, integrating the fish feeder with popular smart home systems would enhance user experience and provide seamless integration into existing smart home setups. Lastly, leveraging data analytics for health monitoring by analyzing feeding patterns and environmental data could enable proactive disease prevention and early intervention. By embracing these future enhancements, the IoT-based automatic fish feeder system has the potential to revolutionize how we care for aquarium fish, making fish keeping more accessible, efficient, and sustainable for enthusiasts and aquaculture professionals alike. This technology represents a promising pathway towards advancing the field of aquaculture management and fostering healthier aquatic ecosystems.

REFERENCES

1. **IoT Based Smart Fish Feeder and Monitoring System**
   By Ahmed Abu-Khadrah, Ghassan F. Issa, Karamath Ateeq and others
   Saudi Electronic University, Riyadh, Saudi Arabia
   Skyline University College, Sharjah, United Arab Emirates
   Skyline University, Sharjah, United Arab Emirates

2. **Automated Fish Feeder using IoT Module**
   By L. Annie Isabella, S. Aarthi, S. Abirami, R. Beautlin Divya, C. R. Geervani and A. Xavier
   RMK Engineering College, Anna University, Chennai

3. **Fish Feeder using Internet of Things**
   By Sourav Meshram, Gourav Meshram, Bhavika Rokde, Roshan Kapse, Omesh Hedaoo and Chandratiya Mandhata
   S. B. Jain Institute of Technology, Management & Research, Nagpur
   e-ISSN: 2395-0056

4. **Internet of Things**
   By Iresh A. Dhotre
   Published by Technical Publications
   First Edition, June 2020
   ISBN: 978-93-90041-06-0

5. **Hackster**
   https://www.hackster.io/RoboticaDIY/automatic-fish-feeder-mechanism-with-timer-by-using-arduino-3b5350

DEVELOPMENT ENVIRONMENT

- Arduino IDE