

**FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE DEPARTAMENTUL AUTOMATICA**

SINTEZA

proiectului de laborator cu titlul: SISTEM AUTOMATIZAT DE HRANIRE A PESTILOR

Autor: Razvan-Stefan Tinta

# Cerințele temei:

Păstrarea peștilor in acvariu necesita hrănire zilnica la ore fixe, chiar si in absenta proprietarului. Proiectul urmărește dezvoltarea unui sistem automat care sa asigure hrănirea precisa a peștilor in intervale programate.

# Soluții alese:

# Sistemul propus se bazează pe următoarele principii funcționale:

# **Modulul RTC**: Utilizează un ceas real-time pentru a menține ora exacta si a declanșa hrănirea la intervale programate.

# **Servo-motor**: Acționează un mecanism care învârte recipientul in care se afla mâncarea.

# **Algoritm**: Verifica constant ora curenta prin RTC si compara cu ora programata, cand ambele sunt egale, pornește servo-motorul si peștii sunt hrăniți.

# **Structura logica:**

# Inițializare RTC si servo motor.

# Citire ore curente din RTC.

# Comparație cu orele programate.

# Acționare servo motor la coincidenta.

# Rezultate obținute:

Aplicația implementata include:

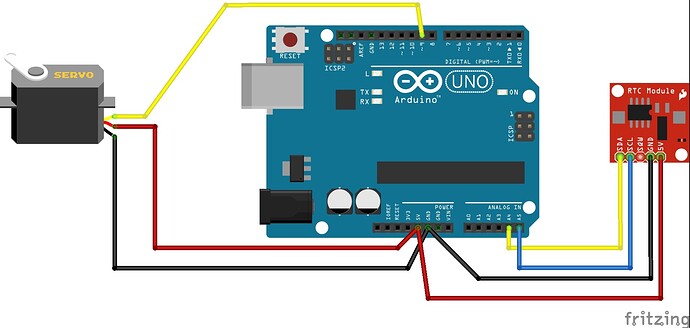
**Schema generala:**

* Arduino Uno conectat la RTC si servo motor.
* Mecanism simplu de eliberare a hranei.

**Descriere componente:**

* Modulul RTC: Păstrează ora exacta si comunica cu Arduino.
* **Servo motor:** Primește comenzi de la Arduino pentru a roti la un unghi specific ca peștii sa fie hrăniți.

**Diagrama:**



# Testări şi verificări:

1. **Programare ore hrănire:** Introduceți orele dorite in codul Arduino.
2. **Instalare:** Montați mecanismul de hrănire deasupra acvariului si conectați componentele.
3. **Funcționare:** Sistemul va elibera hrana automat la orele setate.
4. **Verificare:** Asigurați-va ca RTC afișează ora corecta si servo motorul răspunde la comenzi.

Data:

27.05.2025

ANEXE:

#include <Wire.h> #include <Servo.h>

#define SD2403\_ADDRESS 0x32

Servo myservo;

uint8\_t bcdToDec(uint8\_t val) { return (val / 16 \* 10) + (val % 16); }

void readTime(uint8\_t& hour, uint8\_t& minute, uint8\_t& second) { Wire.beginTransmission(SD2403\_ADDRESS); Wire.write(0x00); Wire.endTransmission();

Wire.requestFrom(SD2403\_ADDRESS, 3); if (Wire.available() >= 3) { second = bcdToDec(Wire.read()); minute = bcdToDec(Wire.read()); hour = bcdToDec(Wire.read()); } }

void setup() { Wire.begin(); Serial.begin(9600); // Servo-ul nu este atașat la pornire }

bool triggered = false;

void loop() { uint8\_t h, m, s; readTime(h, m, s);

Serial.print("Ora RTC: "); Serial.print(h); Serial.print(":"); Serial.print(m); Serial.print(":"); Serial.println(s);

if (h == 18 && m == 37 && s == 0 && !triggered) { myservo.attach(9);

// Deplasare de la 0 la 110°  
for (int pos = 0; pos <= 110; pos++) {  
 myservo.write(pos);  
 delay(15);  
}  
  
delay(3000); // 🔹 Așteaptă 3 secunde la 110°  
  
// Revenire la 0°  
for (int pos = 110; pos >= 0; pos--) {  
 myservo.write(pos);  
 delay(15);  
}  
  
myservo.detach(); // Oprește semnalul PWM  
triggered = true;

}

if (s != 0) { triggered = false; }

delay(500); }