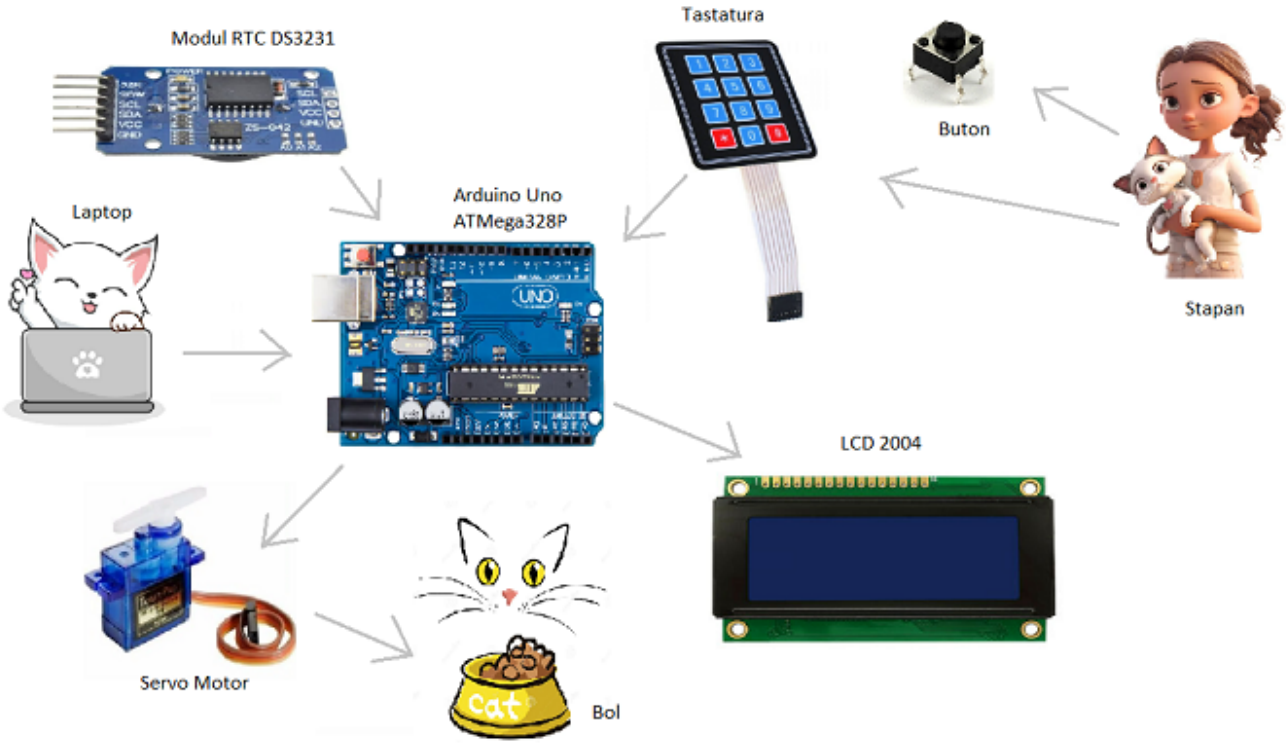
Cat Feeder

- Nume: Mitu Cristina-Stefania
- Grupa: 331CB

Introducere

Proiectul meu îmbină tehnologia modernă cu nevoile zilnice ale stăpânilor de pisici, oferind o soluție simplă și eficientă pentru hrănirea lor: un dispenser automat de hrănire. Cu doar o apăsare de buton și setând ora potrivită, stăpânii de pisici pot asigura hrana la timp pentru prietenii lor pufoși.

Descriere generală



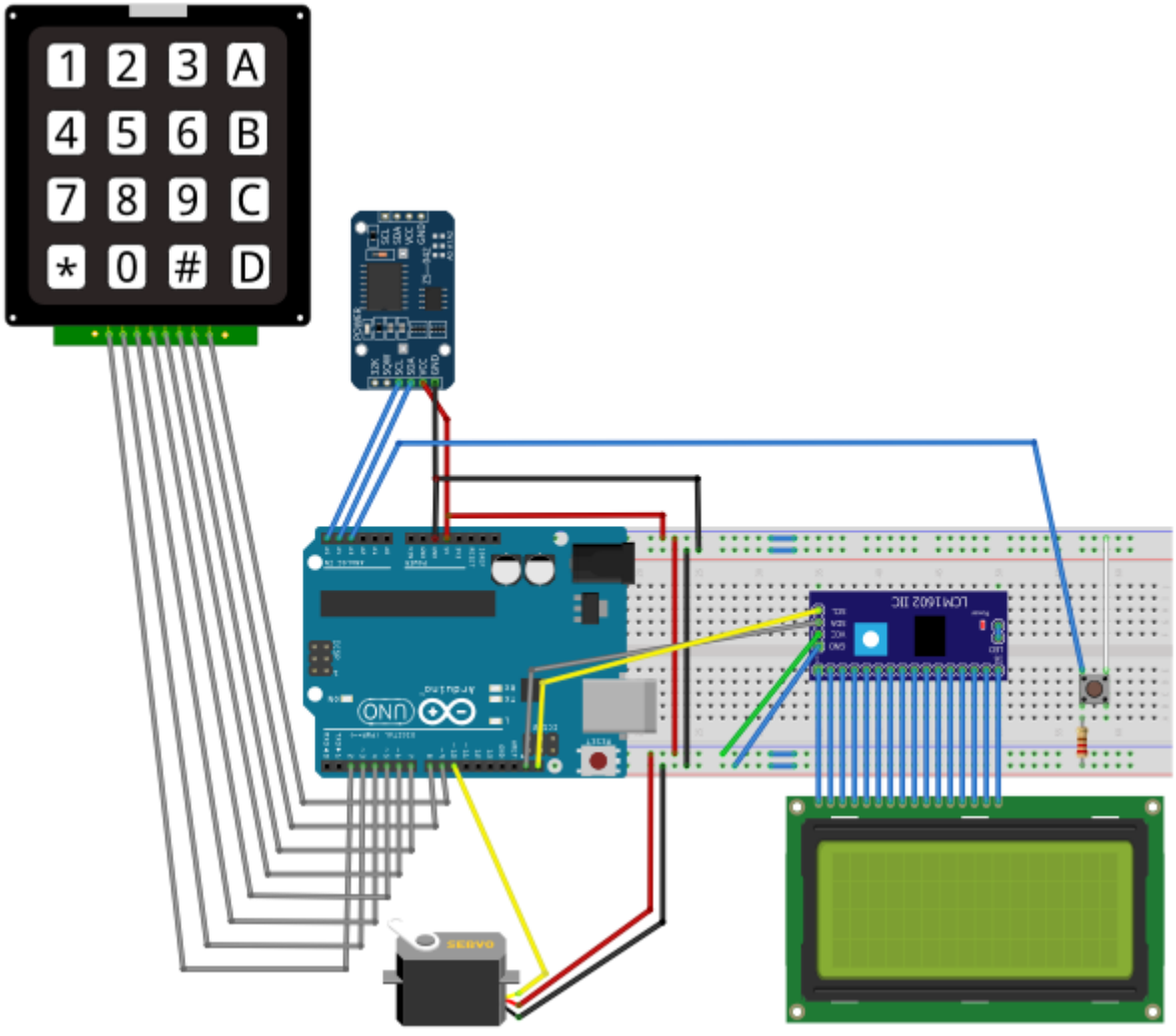
Conectând un cablu USB de la laptop la placa Arduino Uno, aceasta devine centrul de control al întregului sistem. Pe display este afișat numele proiectului pe prima linie, iar ora și data curentă pe a doua și a treia linie. La apăsarea butonului dedicat, se declanșează o întrerupere care va afișa un meniu pe ecran. Utilizatorul are trei opțiuni principale: apăsând tasta 'A' poate introduce manual ora

la care se va elibera hrana, sau apăsând tasta 'B' (meniu dietă) sau 'C' (meniu normal), poate seta hrănirea automată a pisicii de trei ori pe zi la ore prestabilite.

Dacă un meniu a fost deja ales și utilizatorul dorește să selecteze un alt meniu, poate apăsa tasta 'D' pentru a reseta meniul, iar pe ecran va apărea mesajul 'Meniul a fost șters!'. Dacă se încearcă introducerea unei alte taste după ce un meniu a fost selectat, pe ecran va apărea mesajul 'Ai introdus deja un meniu!'. În cazul în care nu a fost ales niciun meniu și se introduce o altă tastă decât cele specificate, se va afișa mesajul 'Tastă invalidă'. De asemenea, dacă utilizatorul a introdus greșit ora după selectarea tastei 'A', poate apăsa tasta 'D' pentru a anula ora.

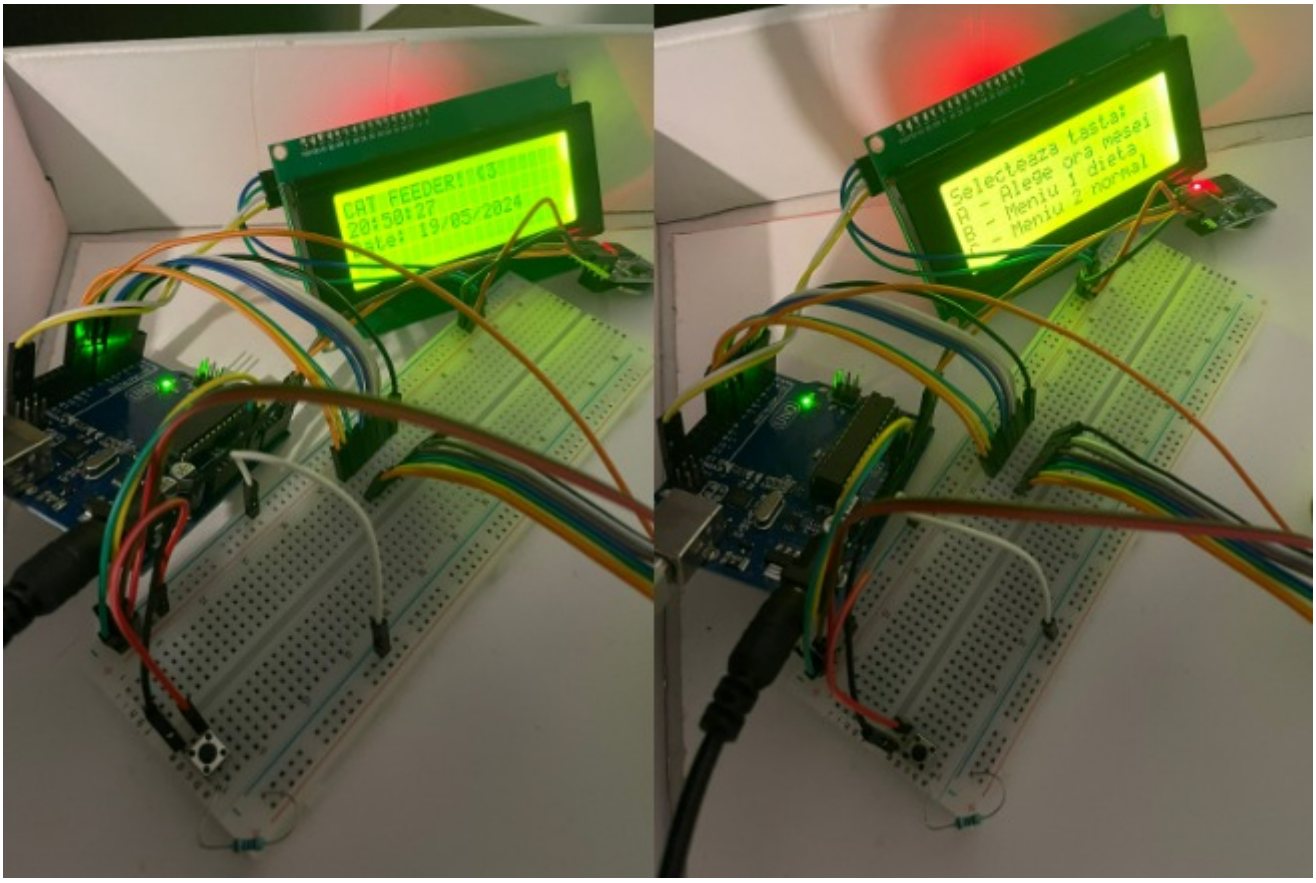
Motorul servo, controlat de Arduino, este esențial în acest sistem. După ce utilizatorul a ales opțiunea dorită, dispozitivul așteaptă până când modulul RTC indică că ora presetată a fost atinsă. În acel moment, Arduino activează motorul servo, care acționează ușița pentru a elibera hrana pentru pisică. Această integrare eficientă a modulelor RTC și servo motor asigură o hrănire precisă și la timp pentru animalul de companie.

Hardware Design



- Arduino Uno ATmega328p
- Tastatura numerica 4*4

- Ecran LCD 2004 Verde
- Buton Tactil
- Modul RTC DS3231 AT24C32
- Servomotor
- Rezistente
- Fire
- Breadboard
- Modul de conversie IIC/I2C



Conexiuni

Tastatura are 8 pini care se conectează la pinurile digitale ale Arduino pentru a detecta apăsările de taste.

- Pin 1 la pinul digital 2 al Arduino
- Pin 2 la pinul digital 3 al Arduino
- Pin 3 la pinul digital 4 al Arduino
- Pin 4 la pinul digital 5 al Arduino
- Pin 5 la pinul digital 6 al Arduino
- Pin 6 la pinul digital 7 al Arduino
- Pin 7 la pinul digital 8 al Arduino
- Pin 8 la pinul digital 9 al Arduino

Modulul RTC are 4 pini: VCC, GND, SDA, SCL, și utilizează comunicarea I2C pentru sincronizarea timpului.

- VCC la 5V al Arduino

- GND la GND al Arduino
- SDA la pinul analogic A4 al Arduino
- SCL la pinul analogic A5 al Arduino

Display-ul LCD simplifică conexiunile datorită modulului I2C integrat, reducând necesitatea de pini suplimentari.

- VCC la 5V al Arduino
- GND la GND al Arduino
- SDA la (I2C) SDA al Arduino
- SCL la (I2C) SCL al Arduino

Butonul tactil detectează apăsările pentru a activa meniul de setări.

- Un pin la pinul analogic A3 al Arduino
- Celălalt pin la GND (prin intermediul unei rezistențe de 4.7kΩ)
- Alt pin la 5V al Arduino

Servo motorul are 3 pini: VCC, GND și Semnal, și controlează mecanismul de eliberare a hranei.

- VCC la 5V al Arduino
- GND la GND al Arduino
- Semnal la pinul digital 10 al Arduino

Software Design

Mediul de dezvoltare utilizat pentru realizarea proiectului a fost: https://www.arduino.cc/en/software/ - versiunea 2.3.2

Alegerea bibliotecilor:

- Keypad.h: pentru interacțiunea cu tastatura.
- Wire.h: pentru comunicarea I2C.
- LiquidCrystal_I2C.h: pentru controlul display-ului LCD prin I2C.
- RTClib.h: pentru interacțiunea cu modulul RTC (DS3231).
- Servo.h: pentru controlul motorului servo.

Element de noutate al proiectului

Elementul de noutate al proiectului constă în integrarea completă a mai multor componente hardware și software pentru a crea un sistem autonom de hrănire a pisicilor. Utilizarea unui meniu interactiv pentru setarea orelor de hrănire și posibilitatea de a selecta diferite programe de hrănire sunt caracteristici inovative care oferă flexibilitate și ușurință în utilizare.

Descrierea codului:

- **global:** Creez un obiect de tipul 'Servo' pentru controlul servomecanismului, specificând si pinul la care este conectat. Configurez o tastatură matricială și inițializez un obiect 'LiquidCrystal_I2C' pentru comunicarea cu display-ul LCD, specificând adresa I2C, dimensiunea și numărul de rânduri, și un obiect 'RTC_DS3231' pentru gestionarea ceasului în timp real. La final, specific pinul la care este conectat butonul și inițializez variabilele de stare pentru gestionarea interacțiunii cu utilizatorul și a procesului de hrănire.

```
Servo servo1;
int servoPin = 10;

const byte ROWS = 4;
const byte COLS = 4;
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
byte rowPins[ROWS] = { 2, 3, 4, 5 };
byte colPins[COLS] = { 6, 7, 8, 9 };
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

LiquidCrystal_I2C lcd(0x27, 20, 4);
RTC_DS3231 rtc;

const int buttonPin = A3; // Pinul la care este conectat butonul
int buttonState = 0; // Variabila pentru a stoca starea butonului (apasat
sau nu)
bool showMenu = false; // Dacă este true se afișează meniul
boolean feed = false; // Dacă a fost ales meniul A true, altfel false
int feedingHour = -1;
int feedingMinute = -1;
int menuSelected = 0;  // 0 = Niciun meniu, 1 = Meniu B dieta, 2 = Meniu C
normal, 3 = Meniu A
int presetFeedingTimes[3][3]; // Ora, minut, secundă  pentru hrănirile
prestabilite
bool butB = false; // Dacă a fost ales meniul B true, altfel false
bool butC = false; // Dacă a fost ales meniul C true, altfel false
```

- **in setup():** Inițializez comunicarea I2C, necesară pentru interacțiunea cu ceasul în timp real (RTC) și display-ul LCD. Apoi, configurez display-ul LCD pentru a afișa informații, iar iluminarea de fundal este activată pentru o vizibilitate mai bună. Atașez servomecanismul la pinul specificat pentru a permite controlul său. De asemenea, stabilesc orele, minutele și secundele pentru hrănirile prestabilite, care sunt stocate într-o matrice. Apoi, verific și inițializez modulul RTC, iar dacă acesta nu poate fi accesat, se afișează un mesaj de eroare pe display. La final, configurez pinul butonului ca intrare, pentru a permite detectarea apăsărilor butonului și interacțiunea cu sistemul în timpul rulării programului.

```
void setup() {
```

```
  Wire.begin();
  lcd.begin(20, 4);
  lcd.backlight();
  servo1.attach(servoPin);

  presetFeedingTimes[0][0] = 20; // Prima ora de hrănire prestabilită
  presetFeedingTimes[0][1] = 44; // Minutul
  presetFeedingTimes[0][2] = 0;  // Secunda
  presetFeedingTimes[1][0] = 20; // A doua ora de hrănire prestabilită
  presetFeedingTimes[1][1] = 44;
  presetFeedingTimes[1][2] = 20;
  presetFeedingTimes[2][0] = 20; // A treia ora de hrănire prestabilită
  presetFeedingTimes[2][1] = 44;
  presetFeedingTimes[2][2] = 35;

  if (!rtc.begin()) {
    lcd.print("Couldn't find RTC");
    while (1);
  }

  if (rtc.lostPower()) {
    rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
  }

  pinMode(buttonPin, INPUT); // Seteaza pinul butonului ca pin de intrare
}
```

- **in loop():** Odată ce detectez o apăsare a butonului, activez afișarea meniului pe ecran, oferind utilizatorului opțiuni clare și ușor de înțeles. Mesajul “Selecteaza tasta:” este afișat pe primul rând al ecranului pentru a indica utilizatorului că trebuie să selecteze o opțiune. Pe rândurile următoare, sunt afișate opțiunile disponibile: “A - Alege ora mesei”, “B - Meniu 1 dieta”, “C - Meniu 2 normal”. În același timp, curăț ecranul pentru a asigura o experiență vizuală plăcută și clară. Dacă utilizatorul selectează una dintre opțiuni, reacționez în consecință, tratând fiecare alegere și oferind feedback adecvat pe ecran. În funcție de alegerea luata, sunt afișate diferite mesaje de confirmare sau de eroare, cum ar fi: “Ai ales tasta 'A'”, “Ai introdus deja un meniu!”, “Tasta invalida”, etc. Dacă utilizatorul selectează opțiunea “D”, care șterge meniul curent, este afișat mesajul “Meniul a fost sters!”. În absența interacțiunii utilizatorului, afișez pe ecran numele proiectului, data și ora curentă. În același timp, monitorizez evenimentele programate, cum ar fi momentele de hrănire prestabilite. Dacă este momentul corespunzător pentru hrănire, activez servomecanismul pentru a asigura hrănirea animalului conform programului. Pentru meniurile A și B, servomecanismul se mișcă la -90 de grade și apoi revine la 90 de grade, simulând deschiderea și închiderea distribuitorului de hrană. În schimb, pentru meniul C, servomecanismul efectuează două astfel de cicluri, mișcându-se de două ori la -90 de grade și înapoi la 90 de grade, pentru a asigura o distribuție mai mare de hrană.

```
void loop() {
  // Citeste starea butonului
  int buttonPress = digitalRead(buttonPin);

  // Dacă butonul este apăsat
  if (buttonPress == 1) {
    showMenu = true;
```

```
    lcd.clear(); // Curăță ecranul pentru a afișa meniul
    lcd.setCursor(0, 0);
    lcd.print("Selecteaza tasta: ");
    lcd.setCursor(0, 1);
    lcd.print("A - Alege ora mesei");
    lcd.setCursor(0, 2);
    lcd.print("B - Meniu 1 dieta");
    lcd.setCursor(0, 3);
    lcd.print("C - Meniu 2 normal");
    delay(500); // Debounce delay
  } else if (showMenu) {
    // Citeste tasta apasata de la keypad
    char key = keypad.getKey();

    if (key) {
      lcd.clear();
      switch (key) {
        case 'A':
          if (menuSelected == 0) {
            lcd.setCursor(0, 0);
            lcd.print("Ai ales tasta 'A'");
            setFeedingTimeA();
            menuSelected = 3;
          } else {
            lcd.setCursor(0, 0);
            lcd.print("Ai introdus deja un meniu!");
          }
          break;
        case 'B':
          if (menuSelected == 0) {
            lcd.setCursor(0, 0);
            lcd.print("Ai ales tasta 'B'");
            menuSelected = 1;
            butB = true;
          } else {
            lcd.setCursor(0, 0);
            lcd.print("Ai introdus deja un meniu!");
          }
          break;
        case 'C':
          if (menuSelected == 0) {
            lcd.setCursor(0, 0);
            lcd.print("Ai ales tasta 'C'");
            menuSelected = 2;
            butC = true;
          } else {
            lcd.setCursor(0, 0);
            lcd.print("Ai introdus deja un meniu!");
          }
          break;
        case 'D':
```

```
          if (menuSelected == 0) {
            lcd.setCursor(0, 0);
            lcd.print("Tasta invalida");
          } else {
            lcd.setCursor(0, 0);
            lcd.print("Meniul a fost sters!");
            menuSelected = 0;
            butB = false;
            butC = false;
            feed = false;
          }
          break;
        default:
          lcd.setCursor(0, 0);
          lcd.print("Tasta invalida");
          break;
      }
      delay(2000); // Afișează mesajul timp de 2 secunde
      showMenu = false; // Revine la afișarea datei și orei după selectarea opțiunii
    }
  } else {
    // Afiseaza data si ora curenta pe ecran
    lcd.clear();
    DateTime now = rtc.now();
    lcd.setCursor(0, 0);
    lcd.print("CAT FEEDER! <3");
    lcd.setCursor(0, 1);
    if (now.hour() < 10) lcd.print('0');
    lcd.print(now.hour());
    lcd.print(':');
    if (now.minute() < 10) lcd.print('0');
    lcd.print(now.minute());
    lcd.print(':');
    if (now.second() < 10) lcd.print('0');
    lcd.print(now.second());

    lcd.setCursor(0, 2);
    lcd.print("Date: ");
    if (now.day() < 10) lcd.print('0');
    lcd.print(now.day());
    lcd.print('/');
    if (now.month() < 10) lcd.print('0');
    lcd.print(now.month());
    lcd.print('/');
    lcd.print(now.year());

    // Verifică dacă este ora de hrănire setată pentru meniul A
    if (now.hour() == feedingHour && now.minute() == feedingMinute && feed) {
      servo1.write(-90);
```

```
      delay(1000);
      servo1.write(90);
      delay(1000);
      menuSelected = 0;
      feed = false; // Dezactivează hrănirea după efectuare
    }
    // Dacă a fost ales meniul B
    if (butB == true) {
      int i;
      for (i = 0; i < 3; i ++) {
        if (now.hour() == presetFeedingTimes[i][0] && now.minute() ==
presetFeedingTimes[i][1] && now.second() == presetFeedingTimes[i][2]) {
          servo1.write(-90);
          delay(1000);
          servo1.write(90);
          delay(2000);
          if (i == 2) {
            butB = false;
            menuSelected = 0;
          }
          break;
        }
      }
    }
    // Dacă a fost ales meniul C
    if (butC == true) {
      int i;
      for (i = 0; i < 3; i ++) {
        if (now.hour() == presetFeedingTimes[i][0] && now.minute() ==
presetFeedingTimes[i][1] && now.second() == presetFeedingTimes[i][2]) {
          servo1.write(-90);
          delay(1000);
          servo1.write(90);
          delay(2000);
          servo1.write(-90);
          delay(1000);
          servo1.write(90);
          delay(2000);
          if (i == 2) {
            butC = false;
            menuSelected = 0;
          }
          break;
        }
      }
    }
  }

  delay(100); // Mică întârziere pentru debounce
}
```

- **functii auxiliare:** În funcția setFeedingTimeA(), inițializez variabilele necesare pentru setarea orei de hrănire și activez flag-ul feed pentru a indica faptul că urmează să fie setată o nouă oră de hrănire. Definind variabilele i și j pentru indexare și r pentru stocarea cifrelor introduse, curăț ecranul și afișez mesajul “Set feeding Time” pe primul rând, urmat de “HH:MM” pe al doilea rând. Într-o buclă infinită, aștept introducerea tastelor de la keypad. Dacă tasta apăsată este un număr între '0' și '9', afișez cifra pe ecran la poziția curentă și o stochez în matricea r după conversia acesteia într-o valoare numerică. Incrementând indicii i și j, mă asigur că după introducerea a două cifre pentru oră, inserez automat un caracter ':' pe ecran pentru separarea orei de minute. Odată ce toate cele patru cifre (două pentru oră și două pentru minute) au fost introduse, calculez valoarea numerică pentru feedingHour și feedingMinute și ies din buclă. Dacă utilizatorul apasă tasta 'D' în timpul procesului de setare, anulez operațiunea și setez flag-ul feed la false.

```
void setFeedingTimeA()
{
  feed = true;
   int i = 0;
   int j = 0;
   char key;
   char r[4];

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Set feeding Time");
  lcd.setCursor(0, 1);
  lcd.print("HH:MM");
  lcd.setCursor(0, 2);
  while(1){
    key = keypad.getKey();
    if(key >= '0' && key <= '9'){ // Acceptă doar cifre
      lcd.setCursor(j, 2);
      lcd.print(key);
      r[i] = key - '0'; // Convertește tasta apăsată la valoare numerică
      i++;
      j++;

      if (j == 2)
      {
        lcd.print(":");
        j++;
      }

      if (i == 4) { // Toate cifrele au fost introduse
        feedingHour = r[0] * 10 + r[1];
        feedingMinute = r[2] * 10 + r[3];
        break;
      }
    }

    if (key == 'D') { // Anulează setarea
      feed = false;
      break;
```

```
    }
  }
}
```

Funcționalități din laborator

Proiectul integrează mai multe noțiuni fundamentale învățate în laboratoare, cum ar fi utilizarea ADC, I2C, Keypad, RTC și Servo.

- *Analog Digital Converter (ADC):* Codul utilizează ADC pentru a citi starea butonului conectat la pinul analogic A3, permițând detectarea interacțiunilor utilizatorului.
- *I2C (Inter-Integrated Circuit):* Protocolul I2C este utilizat pentru comunicarea cu afișajul LCD și modulul RTC (Real-Time Clock). Funcțiile Wire.begin(), lcd.begin(), și rtc.begin() permit afișarea informațiilor în timp real despre starea sistemului și timpul actual.
- *Keypad:* Utilizarea bibliotecii Keypad pentru a citi input-ul de la o tastatură matricială. Funcția keypad.getKey() permite citirea tastelor apăsate, esențială pentru interacțiunea utilizatorului cu sistemul.
- *RTC (Real-Time Clock):* Modulul RTC DS3231 este utilizat pentru a menține și afișa data și ora curente. Funcția rtc.now() este folosită pentru a obține timpul actual, esențial pentru funcționalitatea de hrănire la ore prestabilite.
- *Servo:* Biblioteca Servo este utilizată pentru a controla un servo motor. Funcțiile servo1.attach(servoPin) și servo1.write(angle) permit controlul precis al poziției servo-ului, esențial pentru mecanismul de distribuire a hranei.

Rezultate Obținute

Cu toate că Oscar, motanul meu, nu pare să fie impresionat de proiect, știu că are o latură entuziasmată pe care o păstrează pentru el.

https://youtu.be/IaUBfM3O-8s
Rezumat video:

Inițial, am selectat tasta 'C' pentru a seta meniul normal, cu orele prestabilite la 16:04:00, 16:04:20 și 16:04:35. În timp ce așteptam să se facă orele respective, am vrut să testez și alte funcționalități ale sistemului. Pentru a verifica dacă ecranul afișează mesajele corecte, am selectat un alt meniu la întâmplare. Am vrut să mă asigur că mesajul “Ai introdus deja un meniu!” apare pe ecran atunci când se încearcă setarea unui meniu nou după ce unul este deja activ. Exact așa s-a întâmplat: ecranul a afișat corect mesajul, indicând că nu pot seta un alt meniu până nu îl șterg pe cel actual. Apoi, am ales o tasta la întâmplare, alta decât cele specifice meniului, pentru a testa cum se comportă sistemul. După apăsarea tastei necorespunzătoare, ecranul a afișat mesajul “Tasta invalida!”, confirmând că programul recunoaște și gestionează corect inputurile incorecte. La ora 16:04:00, sistemul a început să execute meniul 'C'. Conform specificațiilor, meniul 'C' face două ture pentru fiecare oră prestabilită, rotind servomecanismul cu -90 de grade și apoi înapoi la 90 de grade. Această secvență s-a repetat și la 16:04:20 și 16:04:35, confirmând că funcția a fost implementată corect. După ce aceste ore au trecut și meniul 'C' și-a încheiat ciclul, am introdus meniul 'A'. Pentru acesta, a trebuit să aleg ora de

hrănire manual, introducând-o prin keypad. Am setat ora la 16:05, iar sistemul a acceptat și afișat noua oră corect. Ulterior, am ales meniul 'B' și am vrut să testez funcția de ștergere a meniului. Am apăsat tasta 'D', iar ecranul a afișat mesajul “Meniul a fost sters!”, indicând că toate meniurile active au fost resetate și sistemul este pregătit pentru o nouă selecție.

Concluzii

Realizarea acestui proiect a fost o experiență extrem de plăcută și interesantă pentru mine. Am avut ocazia să îmbin dragostea mea pentru pisici cu cerințele unei teme academice, ceea ce a dus la crearea unui dispozitiv util și ingenios. Procesul de dezvoltare a fost captivant, deoarece am avut posibilitatea să aplic cunoștințele dobândite la laboratoarele de PM într-un mod practic și relevant. Acest proiect nu este doar o temă de facultate; este un dispozitiv pe care îl voi folosi pentru a-mi hrăni pisica. Faptul că am reușit să creez ceva atât de util și de drag mie îmi aduce o mare bucurie. Pe parcursul realizării acestui proiect, m-am confruntat cu câteva probleme legate de designul ușiței dispozitivului. Inițial, am conceput ușița ca fiind un cerc cu un alt cerc decupat în el, prin care mâncarea să iasă. Cu toate acestea, în timpul testelor, am observat că orezul se bloca între decupatură și peretele pâlniei, ceea ce nu era acceptabil. Am testat diferite tipuri de mâncare, precum alune, cereale, orez, ovăz, zahăr și cafea măcinată. Ușița inițială funcționa fără blocări doar cu cafea măcinată, dar nu era textura ideală a mâncării pentru hrănirea pisicii. Pentru a rezolva această problemă, am modificat designul ușiței, transformând-o într-un semicerc. După această modificare, dispozitivul a funcționat fără blocări indiferent de tipul de mâncare utilizat. Această ajustare a fost esențială pentru a asigura funcționarea fiabilă a dispozitivului și pentru a mă asigura că pisica mea primește hrana necesară fără întreruperi.

Download

cat_feeder.zip

Jurnal

- 1 mai: Achiziționare piese
- 8 mai: Au ajuns piesele
- 12 mai: Milestone - Hardware
- 19 mai: Realizarea codului
- 24 mai: Milestone - Software

Bibliografie/Resurse

- https://docs.arduino.cc/learn/electronics/lcd-displays/
- https://docs.arduino.cc/learn/electronics/servo-motors/
- https://docs.arduino.cc/resources/datasheets/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf
- https://fritzing.org
- https://github.com/SolderedElectronics/e-radionica.com-Fritzing-Library-parts-/blob/master/DS3231%20RTC.fzpz
- https://fritzing.org/projects/arduino-i2c-lcd-display
- https://www.arduino.cc/en/software/
- https://www.geeksforgeeks.org/how-to-interface-i2c-lcd-display-with-arduino/