Garduino

Nume: Crăciun Flavia Maria

Grupa: 336CA

Introducere

Garduino este un proiect ce are ca scop îmbinarea tehnologiei cu natura. În zilele noastre, oamenii petrec mai mult timp acasă și plantele de interior au devenit o modalitate populară de a aduce un pic de natură în locuințe. Plantele nu sunt, însă, doar elemente decorative, ele oferind și multe beneficii pentru sănătatea și bunăstarea noastră. Cu toate acestea, lipsa timpului și neatenția sunt cei mai des întâlniți factori care duc la dispariția prematură a plantelor de interior.

Garduino își propune să rezolve această problemă. Proiectul constă într-un sistem automatizat de întreținere a plantelor și de monitorizare a mediului ambiant în care acestea se află, urmărind parametri precum temperatura și intensitatea luminoasă. Astfel, acesta reprezintă o soluție ideală pentru îngrijirea plantelor de interior, reducând riscul neglijenței și creând un mediu optim pentru creșterea și dezvoltarea lor.

Descriere generală

Sistemul va folosi date primite de la diverși senzori pentru a monitoriza parametrii mediului ambiant și a satisface nevoile plantei. Cele două funcționalități principale constau în automatizarea irigării plantei și afișarea pe ecranul LCD a unor informații legate de condițiile din încăperea în care se află planta.

Sistemul va utiliza date primite de la senzorii de umiditate a solului, de luminozitate și de temperatură și umiditate, iar la nevoie va activa automat o pompă pentru udarea plantei. În momentul în care parametrii detectați de senzor vor fi reglați, sistemul va înceta acționarea pompei.

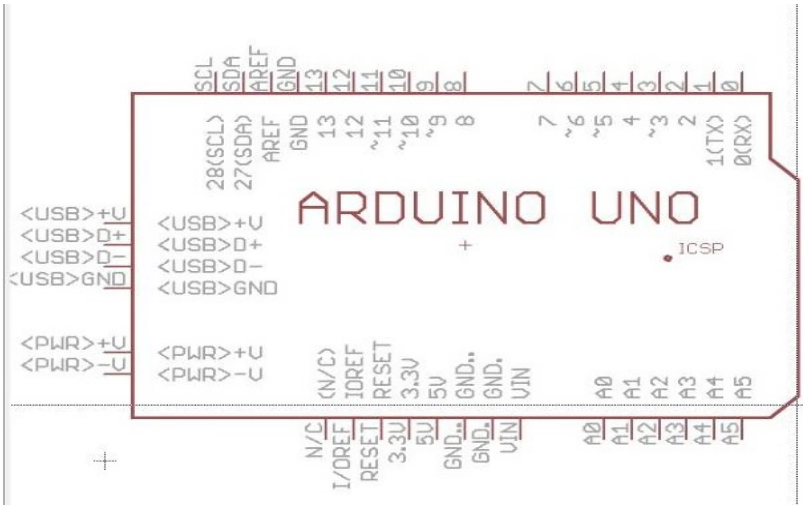
Cei doi senzori de intensitate luminoasă și temperatură și umiditate vor înregistra constant parametrii mediului înconjurător, temperatura și umiditatea fiind afișate constant pe LCD pentru a facilita monitorizarea mediului în care se află planta.

Schema bloc

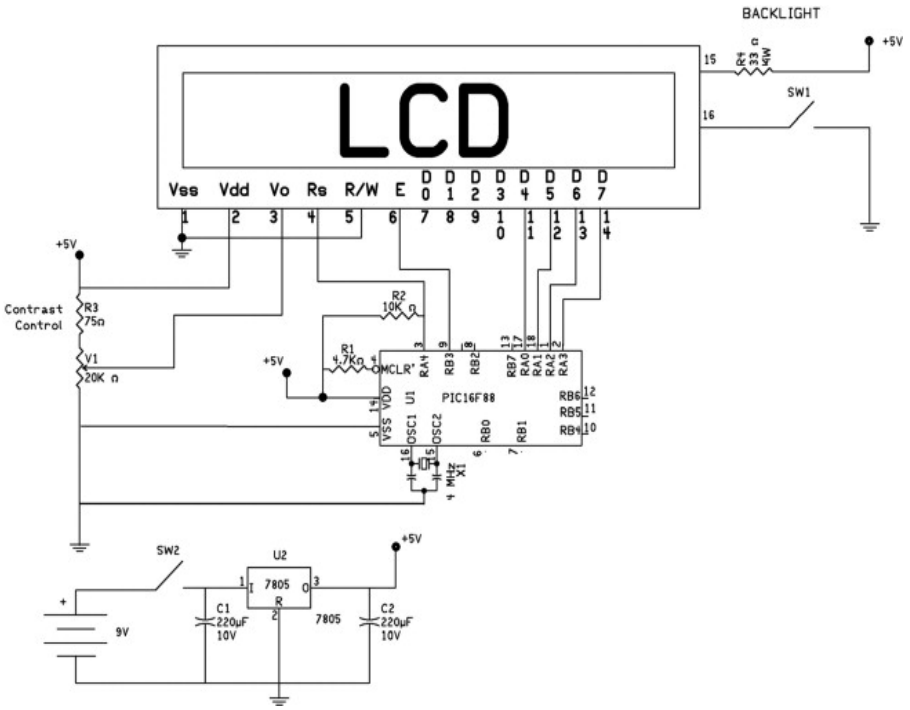
Hardware Design

Listă de piese

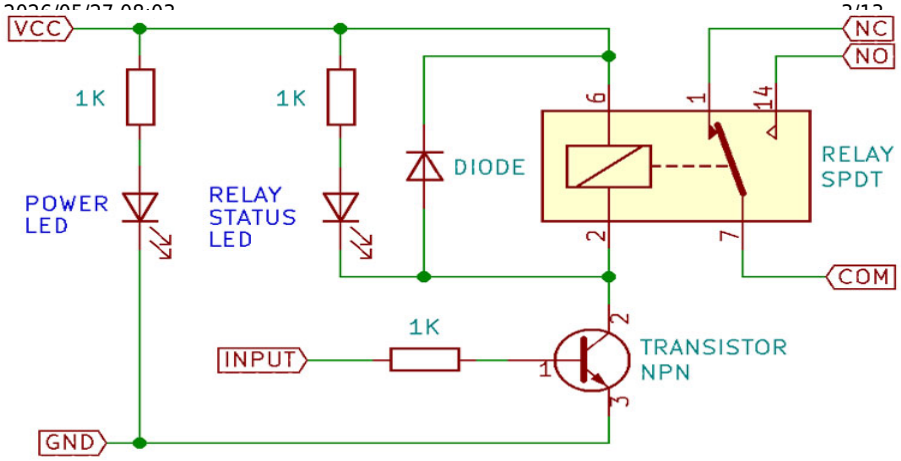
- Arduino UNO R3



- Breadboard
- Ecran LCD 1602

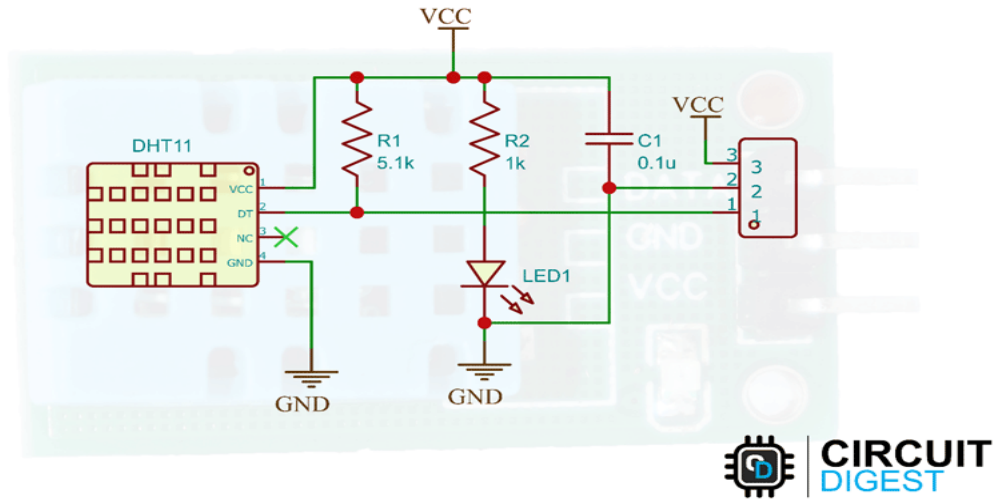


- Pompă de apă
- Modul releu

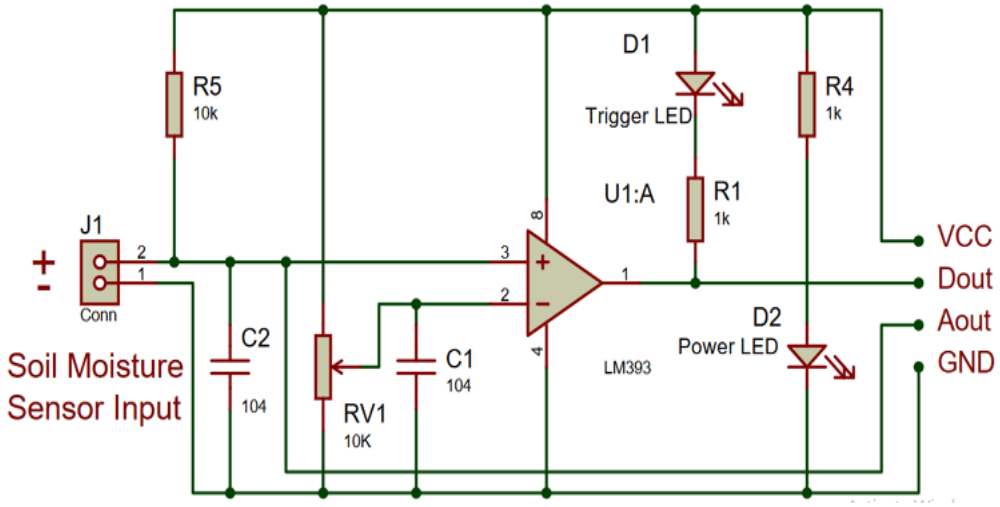


Relay Module Basic Schematic

- Senzor de temperatura si umiditate DHT11

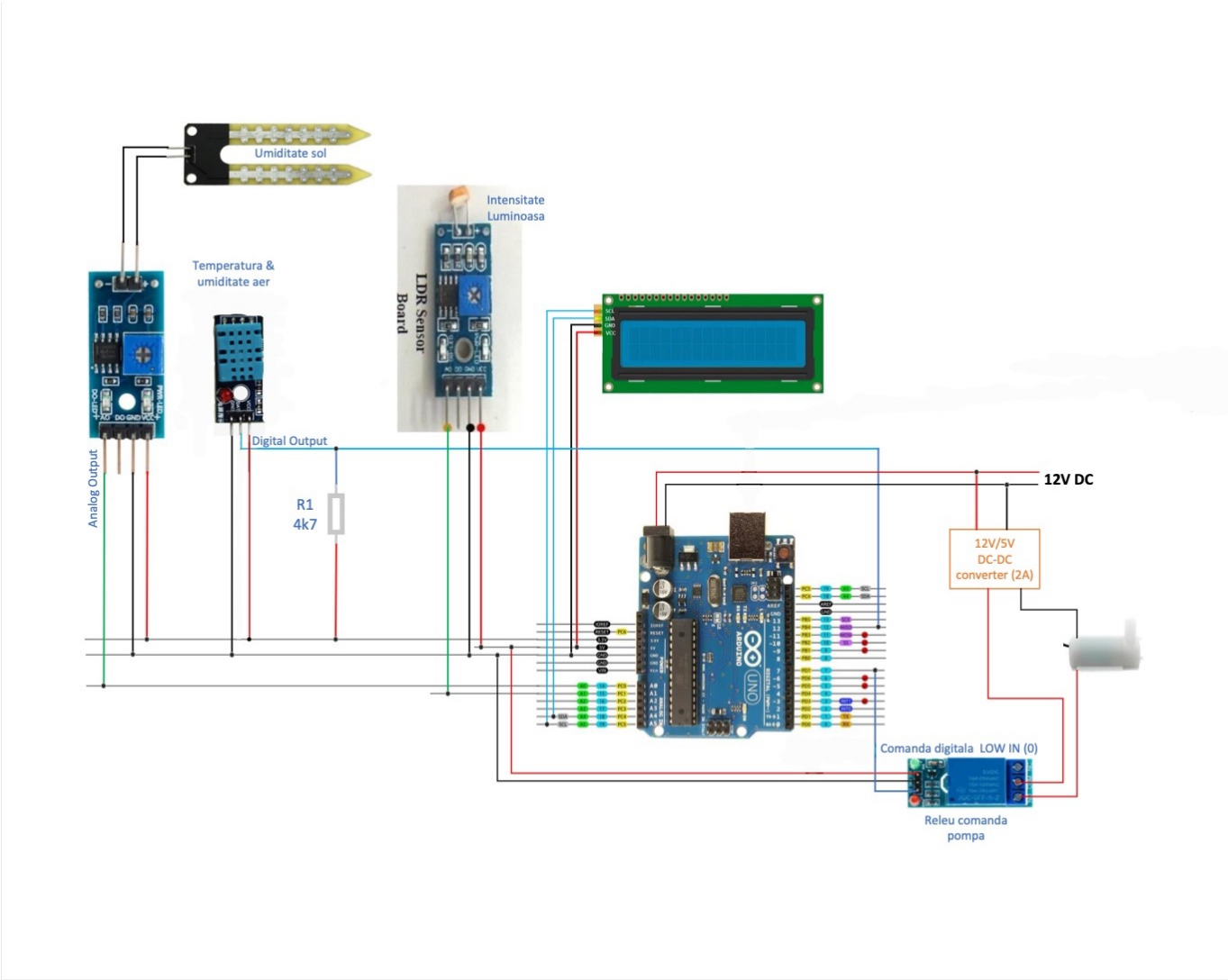


- Modul cu senzor umiditate sol



- Modul intensitate luminoasă

Schema circuitului



Rezultatele simulării

În timpul simulării, circuitul a prezentat comportamentul așteptat, captând cu precizie condițiile de mediu.

Senzorul de temperatură a înregistrat citiri în intervalul 20-25 de grade Celsius și un nivel constant de umiditate de aproximativ 50%, valorile rămânând relativ stabile pe toată perioada simulării. Senzorul de umiditate a solului și senzorul de intensitate luminoasă au afișat citiri cuprinse în intervalul 0-1023 de unități, care vor fi mapate într-un interval de 0-100 pentru a reprezenta procente.

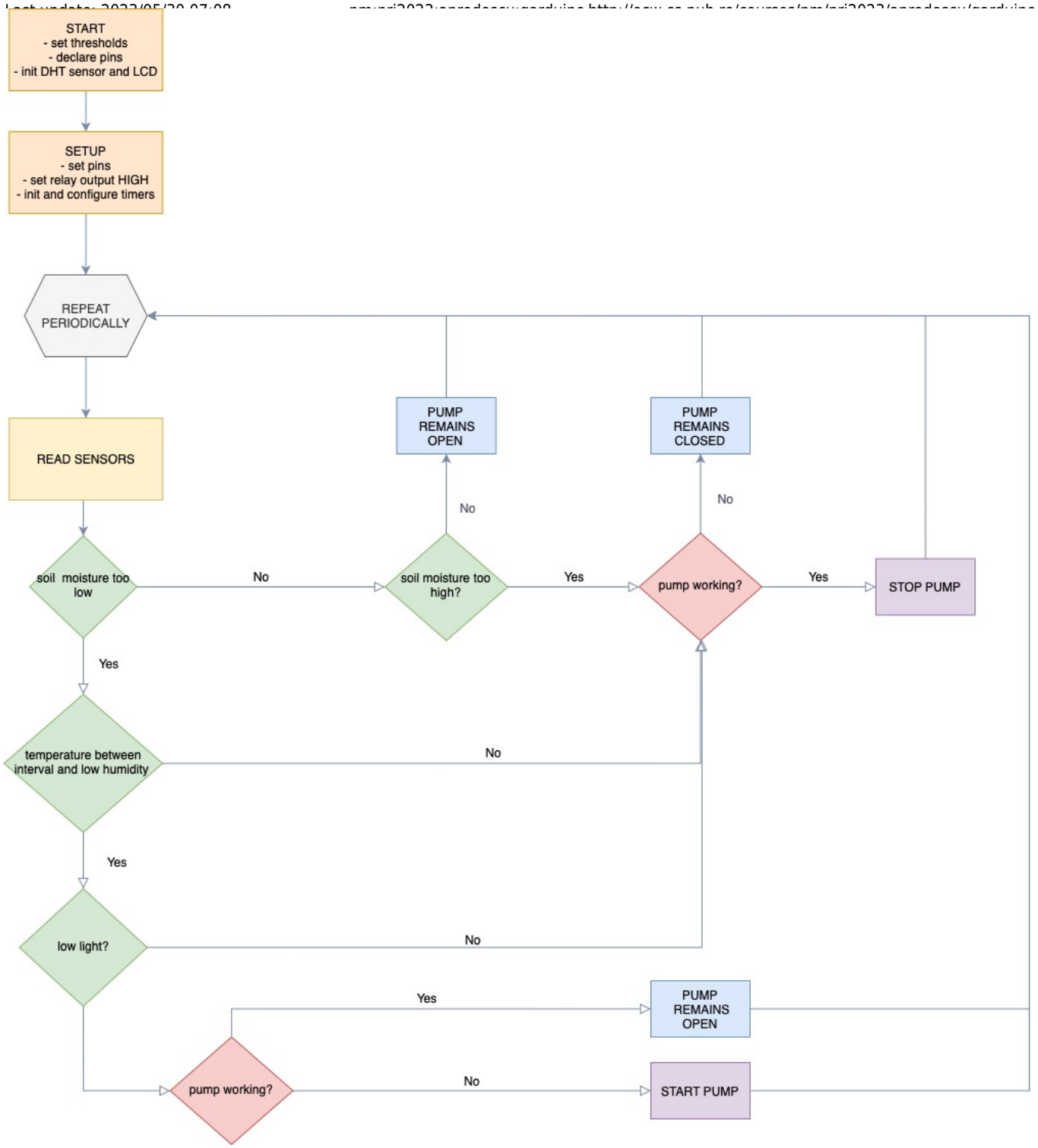
Modulul releu a funcționat conform intenției, activând pompa de apă în funcție de relația dintre datele preluate de senzori și valorile de prag predefinite, asigurând o irigare corespunzătoare. Ecranul LCD a oferit feedback în timp real, prezentând citirile senzorului de temperatură și umiditate, facilitând astfel monitorizarea sistemului.

În general, componentele hardware au demonstrat performanță și interacțiune fiabile, validând funcționalitatea sistemului Garduino.

Software Design

- Mediu de dezvoltare: **Arduino IDE**
- Biblioteci şi surse 3rd-party: **Wire.h**, **LiquidCrystal_I2C.h**, **DHT.h**

Diagrama logică



Workflow

Pentru a avea o modularitate mai bună și a facilita reutilizabilitatea codului, programul a fost organizat în mai multe funcții:

- **setup()**
  Este inițializat senzorul DHT și sunt configurate afișajul LCD-ului și ADC-ul. De asemenea, este definit pinul de ieșire al pompei și se asigură faptul că aceasta este oprită. Întreruperile sunt dezactivate temporar în timpul inițializării (*init_timer()*) și configurării (*configure_timer()*) timere-lor,

apoi sunt reactivate.

Unul dintre timere este folosit de către convertorul analog-digital pentru a citi periodic valorile senzorilor de intensitate luminoasă și umiditate a solului, iar celălalt este folosit pentru a temporiza funcționarea pompei. Acesta va înregistra intervalul de timp de când a fost pornită pompa, iar în cazul în care depășește un anumit prag (raportat la un interval de timp mediu de funcționare în condiții normale), va opri pompa pentru a nu iriga planta mai mult decât este necesar.

- **loop()**
  Controlează fluxul principal de execuție și apelează o serie de funcții care implementează funcționalitățile de bază: citirea senzorilor, afișarea pe LCD, verificarea valorilor citite de senzori pentru a controla starea pompei și monitorizarea intervalului de funcționare a acesteia.

```
void loop() {
  // Automatic pump control based on time elapsed
  if (!pumpBroken)
    checkPump();

  if (pumpBroken) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Broken pump!");
  } else {
    // Read sensor values
    readSensors();

    // Display sensor values on LCD
    printStatus();

    // Check the conditions for pump control
    checkConditions();
  }

  delay(2500); // Delay between iterations
}
```

- **readSensors()**
  Sunt reținute valorile senzorilor de umiditate a solului și intensitate luminoasă determinate prin întreruperi și este calculat timpul scurs de la pornirea pompei. Citirea senzorului de temperatură și umiditate este realizată cu ajutorul bibliotecii DHT.

```
// ADC interrupt service routine for Timer1
ISR(TIMER1_COMPA_vect) {
  // Start a new ADC conversion for soil moisture
  ADMUX = (ADMUX & 0xF0) | (0x00 & 0x0F); // Set ADC input to ADC0 (pin A0)
  ADCSRA |= (1 << ADSC); // Start ADC conversion
  while (ADCSRA & (1 << ADSC)); // Wait for ADC conversion to complete
  uint16_t rawSoilMoisture = ADC;
  interruptSoilMoisture = map(rawSoilMoisture, 0, 1023, 100, 0); // Map ADC reading to soil moisture range

  // Start a new ADC conversion for light sensor
```

```
  ADMUX = (ADMUX & 0xF0) | (0x01 & 0x0F); // Set ADC input to ADC1 (pin A1)
  ADCSRA |= (1 << ADSC); // Start ADC conversion
  while (ADCSRA & (1 << ADSC)); // Wait for ADC conversion to complete
  uint16_t rawLight = ADC;
  interruptLight = map(rawLight, 0, 1023, 100, 0); // Map ADC reading to light range
}
```

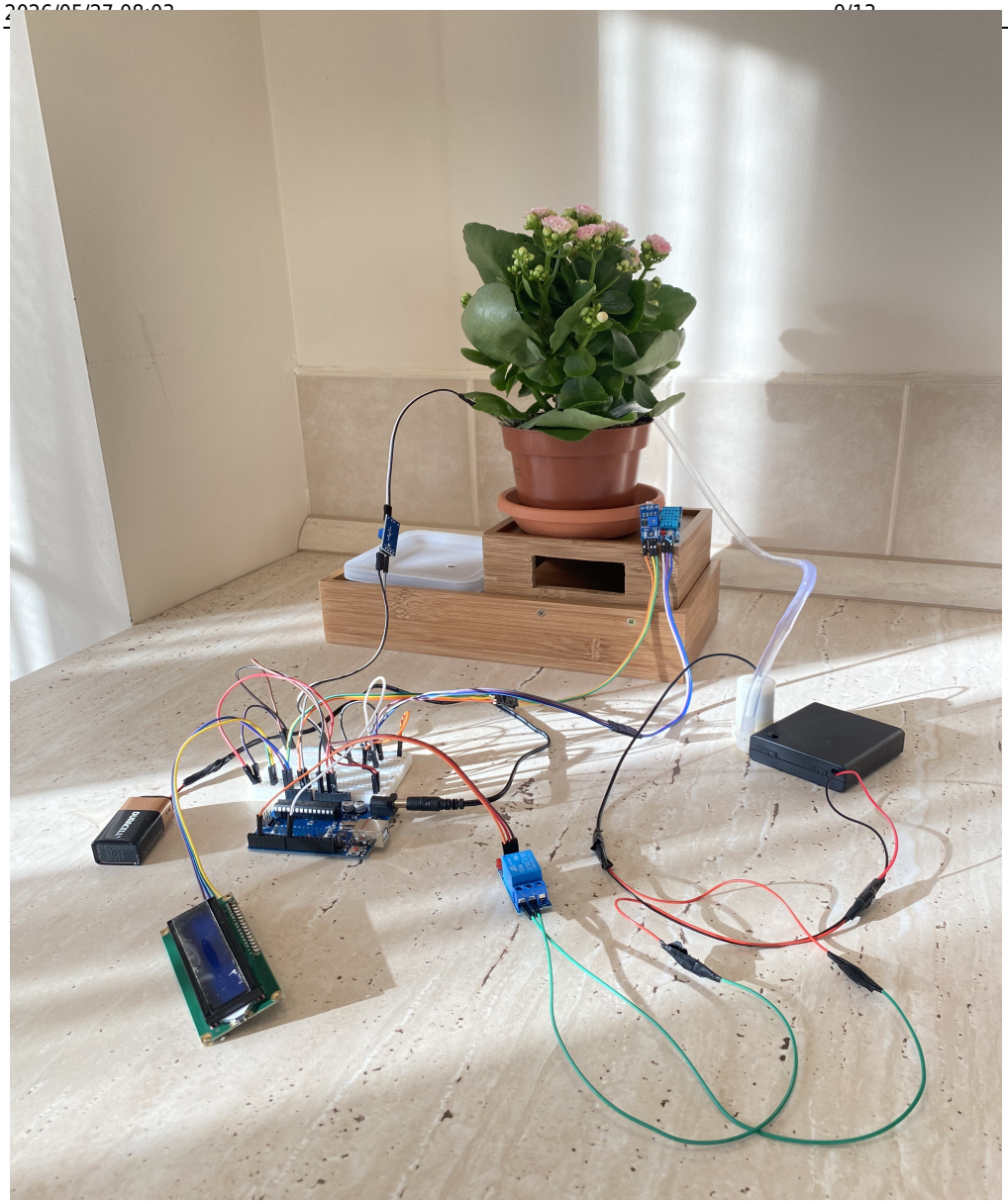
- **printStatus()**
  Utilizată pentru afișarea pe LCD atât a valorilor temperaturii și umidității aerului, cât și a stării pompei și a timpului său de funcționare, după caz.

- **checkConditions()**
  Pe baza unor praguri predefinite, se verifică valorile citite de senzori și este controlată starea pompei în mod corespunzător. Mai întâi este verificată valoarea umidității solului, întrucât aceasta este cea mai importantă, apoi temperatura și umiditatea aerului și ulterior intensitatea luminoasă. Cele două din urmă asigură faptul că irigația are loc într-un moment potrivit al zilei, evitând orele cele mai călduroase, care pot cauza arderea frunzelor plantei.

- **startPump()** și **stopPump()**
  Controlează modul de funcționare a pompei, setând pinul de ieșire al acesteia pe LOW, respectiv HIGH și afișează statusul pompei în momentul pornirii sau opririi acesteia.

- **checkPump()**
  Implementează o verificare suplimentară a funcționării pompei pentrul cazul în care senzorii se defectează. Astfel, dacă timpul curent de funcționare a pompei depășește 3 secunde (timpul mediu de funcționare în condiții normale), aceasta va fi oprită pentru a nu pune o cantitate prea mare de apă plantei.

Rezultate Obţinute



Concluzii

Realizarea acestui proiect mi-a pus la încercare o mulțime de abilități și m-a pus într-o serie de situații pe baza cărora am reușit să trag concluzii importante pentru viitoarele proiecte la care voi lucra.

În primul rând, pentru partea de hardware a fost esențial faptul că am comandat piesele din timp, întrucât unele dintre acestea au ajuns defecte și au trebuit comandate din nou. De asemenea, faptul că am început implementarea suficient de devreme m-a ajutat mult în momentul în care am realizat că aveam câteva componente lipsă (rezistențe, fire). Din procesul de legare a componentelor am învățat cât de importantă e alimentarea la sursa de tensiune potrivită și cum o alimentare necorespunzătoare interferează cu restul componentelor.

Pentru partea de software, cel mai complicat a fost lucrul cu regiștrii și întreruperi și debugging-ul problemelor generate în timpul acestui proces. De asemenea, stabilirea pragurilor pentru valorile senzorilor a fost puțin mai delicată, deoarece senzorii nu au o acuratețe prea bună, deci valorile definite nu corespund neapărat cu valori reale.

Stabilirea design-ului a fost o parte foarte distractivă, trebuind să trec prin cel puțin trei idei diferite de implementare înainte de a alege o formă finală. Până la urmă decizia design-ului final a fost luată punând în balanță atât aspectul practic, cât și existența unui model care să scoată în evidență planta.

În ciuda *numeroaselor* dificultăți pe care le-am întâmpinat (senzori stricați, pompă înfundată, bug-uri în cod, care au provocat ocazional mini inundații etc.), proiectul a fost unul foarte interesant și extrem de satisfăcător odată cu ajungerea la produsul final. Fiind primul proiect la care a trebuit să lucrez individual și la părțile hardware, software și la design, am întâmpinat diverse încercări, pe care fie am reușit să le rezolv, fie pentru care am fost nevoită să caut alternative.

Download

Atât codul sursă, cât și pozele cu schemele proiectului pot fi descărcate aici: garduino.zip

Un demo de funcționare a proiectului poate fi descărcat aici: garduino_demo.zip (nu am putut uploada direct videoclipul)

Export to PDF

Jurnal

1. **7 Mai 2023:** Alegere temă proiect
2. **7 Mai 2023:** M1 - Introducere, Descriere generală, Schemă bloc, Listă componente hardware
3. **13 Mai 2023:** Realizarea legăturilor dintre senzori, LCD, releu și Arduino
4. **18 Mai 2023:** Demo software pentru testarea circuitului
5. **19 Mai 2023:** M2 - Scheme electrice, Schema circuitului, Rezultatele simulării
6. **23 Mai 2023:** Finalizare componentă software
7. **25 Mai 2023:** Legare pompă cu releu și conectarea la alimentator
8. **25 Mai 2023:** Stabilire threshold-uri prin simulare

9. **28 Mai 2023:** M3 - Diagrama logică, Descrierea workflow-ului
10. **28 Mai 2023:** Stabilire design proiect
11. **29 Mai 2023:** Mofidicare caseta pentru prezentare
12. **29 Mai 2023:** Filmare demo
13. **30 Mai 2023:** Asamblare finala a proiectului

Bibliografie/Resurse

Resurse Hardware

- Schemă electrică LCD: https://www.sciencedirect.com/topics/engineering/liquid-crystal-display
- Schemă electrică modul umiditate sol: https://components101.com/modules/soil-moisture-sensor-module
- Schemă electrică senzor intensitate luminoasă: https://www.electroduino.com/ldr-sensor-module-how-ldr-sensor-works/
- Schemă electrică Arduino: https://community.element14.com/products/arduino/b/blog/posts/finally-an-arduino-library-for-eagle
- Schemă electrică senzor temperatură și umiditate: https://circuitdigest.com/microcontroller-projects/interfacing-dht11-sensor-with-arduino
- Schemă electrică modul releu: https://forum.kicad.info/t/arduino-relay-module/32236

Resurse Software

- Citire senzor DHT11: https://www.youtube.com/watch?v=yaOF9mILrRg
- Întreruperi: http://www.gammon.com.au/interrupts
- Arduino UNO R3 datasheet: https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf
- Laborator întreruperi: https://ocw.cs.pub.ro/courses/pm/lab/lab2-2023
- Laborator timere: https://ocw.cs.pub.ro/courses/pm/lab/lab3-2023
- Laborator ADC: https://ocw.cs.pub.ro/courses/pm/lab/lab4-2022
- Wire.h: https://www.arduino.cc/reference/en/libraries/
- DHT.h: https://github.com/adafruit/Adafruit_Sensor
- LiquidCrystal_I2C.h: https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library