Sistem inteligent de monitorizare a umidității solului

Introducere

Proiectul constă în realizarea unui sistem embedded pentru monitorizarea umidității solului unei plante. Sistemul măsoară nivelul de umiditate folosind un senzor higrometru capacitiv, afișează valoarea măsurată și statusul plantei pe un display LCD 16×2, iar atunci când umiditatea scade sub un prag prestabilit, avertizează utilizatorul prin aprinderea unui LED roșu și pornirea unui buzzer activ.

Scopul proiectului este de a ajuta utilizatorul să știe când planta are nevoie de apă și de a urmări evoluția umidității solului în timp. Pentru acest lucru, sistemul salvează măsurătorile pe un card microSD, împreună cu data și ora obținute de la un modul RTC DS3231. De asemenea, valorile măsurate și statisticile calculate pot fi transmise prin Bluetooth către un telefon sau laptop, folosind un modul HC-05.

Care este ideea de la care am pornit și de ce este util? Ei bine, eu toată viața mea, am avut 3 ghivece cu plante: primul eram in gimnaziu, era o floare, a ținut 1 lună, al doilea eram în liceu și am avut un cactus (m-a ținut 3 luni) și acum am unul de o lună și cred că e timpul să accept decesul. Mereu uit să ud plantele și știu sigur că este o problemă pentru multe persoane. Statisticile transmise prin bluetooth mă pot ajuta să-mi dau seama în fazele neoportune ale plantei dacă e din cauza udatului excesiv sau udatului prea rar sau non existent.

Prezentarea pe scurt a proiectului vostru:

- ce face
- care este scopul lui
- care a fost ideea de la care aţi pornit
- de ce credeţi că este util pentru alţii şi pentru voi

Descriere generală

Sistemul are la bază o placă Arduino UNO R3, care coordonează toate componentele proiectului. Senzorul de umiditate este introdus în sol și trimite către placă o valoare analogică, în funcție de cât de umed sau uscat este pământul. Această valoare este citită prin pinul A0, apoi este transformată într-un procent de umiditate și comparată cu un prag stabilit în program.

Dacă umiditatea scade sub pragul ales, sistemul consideră că planta trebuie udată. În această situație, LED-ul roșu se aprinde, buzzerul pornește, iar pe LCD apare un mesaj de avertizare. Dacă nivelul de umiditate este normal, LED-ul și buzzerul rămân oprite, iar pe ecran este afișat statusul plantei.

Pentru a putea urmări măsurătorile în timp, proiectul folosește un modul RTC DS3231, care oferă data și ora fiecărei citiri. Valorile sunt salvate pe un card microSD într-un fișier de tip `.csv`, astfel încât pot fi analizate ulterior. În plus, prin modulul Bluetooth HC-05, datele curente și statisticile calculate pot fi transmise către un dispozitiv.

Sistemul include și un buton Start/Stop, care permite pornirea sau oprirea monitorizării fără a deconecta alimentarea plăcii.

Modul Hardware	Descriere Tehnică	Interacțiune / Protocol
Arduino UNO R3 compatibil	Unitatea centrală de procesare. Gestionează logica sistemului, calculele și comunicarea cu perifericele.	Master-ul sistemului; controlează senzorul, LCD-ul, LED-ul, buzzerul, RTC-ul, microSD-ul și Bluetooth-ul.
Senzor higrometru capacitiv	Măsoară umiditatea solului printr-un semnal analogic variabil.	Analog / ADC: AOUT → A0.
LCD 16×2 cu I2C	Afișează umiditatea curentă și statusul plantei.	I2C: SDA → A4, SCL → A5.
RTC DS3231	Oferă data și ora reală pentru fiecare măsurătoare.	I2C: SDA → A4, SCL → A5.
Modul microSD	Salvează datele într-un fișier CSV.	SPI: CS → D10, MOSI → D11, MISO → D12, SCK → D13.
HC-05 Bluetooth	Transmite valorile și statisticile către telefon/laptop.	UART / SoftwareSerial: TXD → D5, RXD → D6.
LED roșu	Avertizare vizuală pentru sol uscat.	GPIO digital: D7.
Buzzer activ	Avertizare sonoră pentru sol uscat.	GPIO digital: D8.
Buton Start/Stop	Pornește/oprește monitorizarea logică.	Intrare digitală: D2 cu INPUT_PULLUP.

O schemă bloc cu toate modulele proiectului vostru, atât software cât şi hardware însoţită de o descriere a acestora precum şi a modului în care interacţionează.

Exemplu de schemă bloc: http://www.robs-projects.com/mp3proj/newplayer.html

Hardware Design

Descriere componente

- Arduino UNO R3
- Breadboard HQ 830 puncte
- Senzor de umiditate a solului
- LCD 1602 cu interfață I2C
- LED roșu
- Set rezistențe
- Modul cu buzzer activ

- Buton Start/Stop
- Modul RTC DS3231
- Baterie CR2032 3V
- Modul microSD
- Card microSD 4GB
- Modul Bluetooth HC-05
- Fire jumper tată-tată
- Fire jumper mamă-tată
- Cablu USB pentru Arduino UNO

Aici puneţi tot ce ţine de hardware design:

- listă de piese
- scheme electrice (se pot lua şi de pe Internet şi din datasheet-uri, e.g. http://www.captain.at/electronic-atmega16-mmc-schematic.png)
- diagrame de semnal
- rezultatele simulării

==== Stadiul actual al implementării hardware ====

În stadiul actual, partea hardware a proiectului a fost realizată la nivel de proiectare și montaj pe breadboard. Au fost alese componentele principale, au fost stabilite conexiunile dintre acestea și placa Arduino UNO R3, iar schema electrică a fost realizată în Fusion 360 Electronics.

Montajul este realizat fără lipire, folosind breadboard și fire jumper. Alimentarea modulelor se face de la placa Arduino UNO R3, prin pinul de 5V și GND. Toate componentele împart aceeași masă comună, ceea ce este necesar pentru ca semnalele citite și transmise între module să fie interpretate corect.

În schema electrică, modulele comerciale precum senzorul de umiditate, LCD-ul, RTC-ul, modulul microSD și modulul Bluetooth HC-05 sunt reprezentate prin conectori, deoarece acestea sunt deja asamblate și sunt conectate la Arduino prin pini.

==== Componente folosite și rolul lor ====

Componentă	Rol în proiect
Arduino UNO R3 compatibil	Placa principală a proiectului. Coordonează citirea senzorului, afișarea datelor, avertizarea utilizatorului, salvarea datelor și comunicarea Bluetooth.
Breadboard	Permite realizarea montajului fără lipire și conectarea rapidă a componentelor.
Senzor higrometru capacitiv	Măsoară umiditatea solului și transmite către Arduino o valoare analogică.
LCD 16×2 cu I2C	Afișează umiditatea curentă și statusul plantei.
LED roșu	Oferă avertizare vizuală atunci când planta trebuie udată.
Rezistență 220Ω	Limitează curentul prin LED pentru a-l proteja.
Buzzer activ	Oferă avertizare sonoră atunci când umiditatea este sub pragul stabilit.
Buton Start/Stop	Permite pornirea și oprirea logică a monitorizării.
Modul RTC DS3231	Oferă data și ora exactă pentru fiecare măsurătoare.
Baterie CR2032	Menține funcționarea RTC-ului atunci când sistemul nu este alimentat.
Modul microSD	Permite salvarea măsurătorilor într-un fișier CSV.

Card microSD 4GB	Reprezintă mediul de stocare pentru datele salvate.
Modul Bluetooth HC-05	Permite transmiterea datelor și statisticilor către telefon sau laptop.
Rezistențe 1kΩ și 2kΩ	Formează divizorul de tensiune pentru pinul RXD al modulului HC-05.
Fire jumper	Realizează conexiunile dintre Arduino, breadboard și module.
Cablu USB	Alimentează placa Arduino și permite încărcarea codului.

==== Pini folosiți și explicații ====

Componentă	Pin componentă	Pin Arduino / conexiune	Explicație
Senzor higrometru capacitiv	AOUT	A0	Ieșirea senzorului este analogică, deci trebuie citită printr-un pin analogic al plăcii Arduino.
Senzor higrometru capacitiv	VCC	5V	Senzorul este alimentat de la pinul de 5V al plăcii Arduino.
Senzor higrometru capacitiv	GND	GND	Este conectat la masa comună a sistemului.
LCD 16×2 I2C	SDA	A4	Pe Arduino UNO, pinul A4 este folosit pentru linia SDA a comunicației I2C.
LCD 16×2 I2C	SCL	A5	Pe Arduino UNO, pinul A5 este folosit pentru linia SCL a comunicației I2C.
LCD 16×2 I2C	VCC	5V	Display-ul este alimentat la 5V.
LCD 16×2 I2C	GND	GND	Este conectat la masa comună.
RTC DS3231	SDA	A4	RTC-ul folosește tot comunicație I2C și poate împărți linia SDA cu LCD-ul.
RTC DS3231	SCL	A5	RTC-ul împarte linia SCL cu LCD-ul.
RTC DS3231	VCC	5V	Modulul RTC este alimentat la 5V.
RTC DS3231	GND	GND	Este conectat la masa comună.
LED roșu	Anod	D7 prin rezistență 220Ω	Pinul D7 controlează aprinderea LED-ului. Rezistența limitează curentul prin LED.
LED roșu	Catod	GND	Catodul LED-ului este conectat la masă.
Buzzer activ	+	D8	Pinul D8 controlează pornirea și oprirea buzzerului.
Buzzer activ	-	GND	Buzzerul este conectat la masa comună.
Buton Start/Stop	un pin	D2	D2 este folosit ca intrare digitală pentru citirea stării butonului.
Buton Start/Stop	celălalt pin	GND	La apăsare, pinul D2 este conectat la GND. În cod se folosește INPUT_PULLUP.
Modul microSD	CS	D10	Pinul CS selectează modulul microSD pe magistrala SPI.
Modul microSD	MOSI	D11	MOSI transmite date de la Arduino către modulul microSD.
Modul microSD	MISO	D12	MISO transmite date de la modulul microSD către Arduino.
Modul microSD	SCK	D13	SCK este semnalul de ceas pentru comunicația SPI.
Modul microSD	5V	5V	Modulul microSD folosit este alimentat la 5V.
Modul microSD	GND	GND	Este conectat la masa comună.

Modul microSD	3V3	neconectat	Pinul 3V3 nu este folosit, deoarece modulul este alimentat prin pinul de 5V.
HC-05 Bluetooth	TXD	D5	TXD al modulului transmite date către Arduino. D5 este folosit prin SoftwareSerial.
HC-05 Bluetooth	RXD	D6 prin divizor 1kΩ/2kΩ	Arduino transmite date către HC-05 prin D6. Divizorul reduce nivelul logic către RXD.
HC-05 Bluetooth	VCC	5V	Modulul Bluetooth este alimentat la 5V.
HC-05 Bluetooth	GND	GND	Este conectat la masa comună.
HC-05 Bluetooth	STATE, EN	neconectat	Acești pini nu sunt necesari pentru funcționalitatea de bază a proiectului.

==== Schema electrică ====

Schema electrică a fost realizată în Fusion 360 Electronics și prezintă conexiunile dintre placa Arduino UNO R3 și toate modulele folosite în proiect.

În schema electrică, Arduino UNO R3 este componenta centrală. Senzorul de umiditate este conectat la pinul A0, deoarece acesta transmite un semnal analogic. LCD-ul și modulul RTC DS3231 sunt conectate pe aceeași magistrală I2C, folosind pinii A4 pentru SDA și A5 pentru SCL. Acest lucru este posibil deoarece magistrala I2C permite conectarea mai multor module pe aceleași două linii de comunicație.

Modulul microSD folosește protocolul SPI, motiv pentru care este conectat la pinii D10-D13 ai plăcii Arduino. Modulul Bluetooth HC-05 comunică serial cu Arduino prin SoftwareSerial, folosind pinii D5 și D6. Pentru pinul RXD al modulului HC-05 se folosește un divizor de tensiune format din rezistențele de 1kΩ și 2kΩ, deoarece semnalul transmis de Arduino are nivel logic de 5V.

LED-ul roșu este conectat la pinul D7 printr-o rezistență de 220Ω, iar buzzerul activ este conectat la pinul D8. Butonul Start/Stop este conectat între pinul D2 și GND, iar în cod va fi configurat folosind INPUT_PULLUP.

==== Conexiuni principale ====

=== Senzorul de umiditate ===

```
Senzor VCC  -> Arduino 5V
Senzor GND  -> Arduino GND
Senzor AOUT -> Arduino A0
```

Senzorul transmite o valoare analogică, iar Arduino o citește prin pinul A0. Această valoare va fi ulterior transformată într-un procent de umiditate.

=== LCD și RTC pe I2C ===

```
LCD SDA -> A4
LCD SCL -> A5

RTC SDA -> A4
RTC SCL -> A5
```

LCD-ul și RTC-ul folosesc același protocol de comunicație, I2C. De aceea, ambele module pot fi conectate la aceiași pini SDA și SCL ai plăcii Arduino.

=== Modulul microSD ===

```
microSD CS   -> D10
microSD MOSI -> D11
microSD MISO -> D12
microSD SCK  -> D13
```

Modulul microSD folosește protocolul SPI. Pinul CS este folosit pentru selectarea modulului, iar MOSI, MISO și SCK sunt folosite pentru transferul efectiv de date.

=== Modulul Bluetooth HC-05 ===

```
HC-05 TXD -> D5
HC-05 RXD -> D6 prin divizor de tensiune
HC-05 VCC -> 5V
HC-05 GND -> GND
```

Pentru RXD se folosește un divizor de tensiune:

```
D6 Arduino -> rezistență 1kΩ -> nod comun -> HC-05 RXD
nod comun -> rezistență 2kΩ -> GND
```

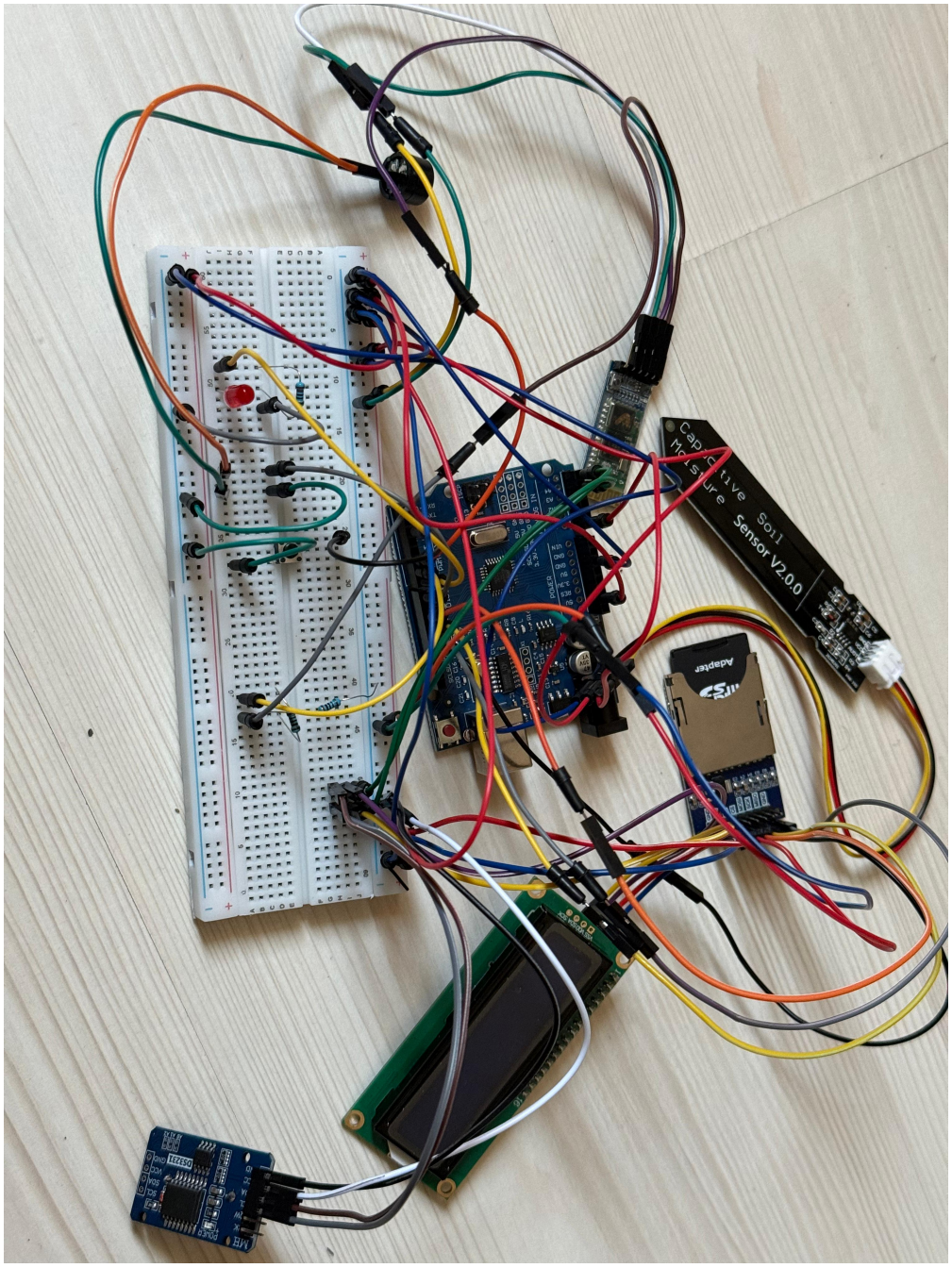
Această conexiune este necesară pentru a proteja intrarea RXD a modulului HC-05.

=== LED, buzzer și buton ===

```
D7 -> rezistență 220Ω -> LED roșu -> GND
D8 -> buzzer activ -> GND
D2 -> buton -> GND
```

LED-ul și buzzerul sunt folosite ca ieșiri digitale. Butonul este folosit ca intrare digitală și va fi configurat în cod cu INPUT_PULLUP.

==== Imagini cu componentele conectate ====



Imaginea de mai sus prezintă montajul hardware realizat pe breadboard. Se observă placa Arduino UNO R3, breadboard-ul și conexiunile dintre module.

Schema electrică atașată prezintă echivalentul logic al montajului fizic. Aceasta include conexiunile către senzor, LCD, RTC, microSD, Bluetooth, LED, buzzer și buton.

==== Verificarea schemei electrice ====

Schema electrică a fost verificată în Fusion 360 Electronics folosind funcția ERC. După conectarea alimentării prin +5V și GND și după lăsarea neconectată a pinilor care nu sunt folosiți, schema nu mai prezintă erori critice.

Pinii nefolosiți sunt:

- STATE și EN de la modulul HC-05;
- 3V3 de la modulul microSD.

Acești pini sunt nefolosiți deoarece nu sunt necesari pentru funcționalitățile planificate ale proiectului.

Software Design

Mediu de dezvoltare

Firmware-ul proiectului este dezvoltat în:

- PlatformIO în Visual Studio Code

Codul este scris în C/C++ pentru Arduino UNO R3.

Proiectul este organizat pe mai multe fișiere, pentru a separa logica aplicației de partea de configurare hardware și de calculul statisticilor.

Structura proiectului

```
soil_moisture_monitor/
│
├── platformio.ini
│
├── include/
│   ├── config.h
│   ├── hardware.h
│   └── statistics.h
│
├── src/
│   ├── main.cpp
│   ├── hardware.cpp
│   └── statistics.cpp
│
├── README.md
└── ChangeLog.md
```

Fișiere implementate

Fișier	Rol
platformio.ini	Conține configurația PlatformIO pentru placa Arduino UNO și librăriile externe folosite.
include/config.h	Conține pinii folosiți, adresa LCD-ului, valorile de calibrare pentru senzor, pragul de umiditate și intervalele de timp pentru măsurare, logare și transmitere Bluetooth.

include/hardware.h	Declară funcțiile folosite pentru interacțiunea cu hardware-ul: LCD, RTC, microSD, Bluetooth, LED, buzzer, buton și senzor.
include/statistics.h	Definește structura de date pentru statistici și funcțiile asociate.
src/main.cpp	Conține logica principală a aplicației și bucla principală de execuție.
src/hardware.cpp	Implementează inițializarea și controlul componentelor hardware.
src/statistics.cpp	Implementează funcțiile pentru calculul statisticilor.
README.md	Conține descrierea proiectului, conexiunile și instrucțiuni de rulare.
ChangeLog.md	Conține istoricul modificărilor proiectului.

Librării folosite

Librărie	Rol
Arduino.h	Funcționalități de bază pentru Arduino.
Wire.h	Comunicare I2C pentru LCD și RTC DS3231.
LiquidCrystal_I2C.h	Controlul display-ului LCD 16×2 cu interfață I2C.
RTClib.h	Citirea datei și orei de la modulul RTC DS3231.
SPI.h	Comunicare SPI pentru modulul microSD.
SD.h	Scrierea datelor pe cardul microSD.
SoftwareSerial.h	Comunicare serială software cu modulul Bluetooth HC-05.

Configurarea pinilor

Pinii proiectului sunt definiți în fișierul `config.h`.

```
const uint8_t PIN_SOIL_SENSOR = A0;
const uint8_t PIN_BUTTON = 2;

const uint8_t PIN_BT_RX = 5;
const uint8_t PIN_BT_TX = 6;

const uint8_t PIN_LED = 7;
const uint8_t PIN_BUZZER = 8;

const uint8_t PIN_SD_CS = 10;
```

Pinii A4 și A5 sunt folosiți automat pentru comunicația I2C:

```
A4 -> SDA
A5 -> SCL
```

Pentru modulul microSD sunt folosiți pinii SPI ai plăcii Arduino UNO:

```
D10 -> CS
D11 -> MOSI
D12 -> MISO
D13 -> SCK
```

Configurarea senzorului de umiditate

Valorile de calibrare ale senzorului sunt definite în `config.h`:

```
const int SENSOR_DRY_ADC = 750;
const int SENSOR_WET_ADC = 350;
const int HUMIDITY_THRESHOLD_PERCENT = 35;
```

Conversia valorii ADC în procent se face în funcția `convertAdcToHumidityPercent()`:

```
int convertAdcToHumidityPercent(int adcValue) {
  long humidity = map(adcValue, SENSOR_DRY_ADC, SENSOR_WET_ADC, 0, 100);
  humidity = constrain(humidity, 0, 100);
  return (int)humidity;
}
```

Algoritmul aplicației

La pornirea sistemului:

1. se inițializează statisticile;
2. se configurează pinii pentru LED, buzzer și buton;
3. se pornește comunicația serială și comunicația Bluetooth;
4. se inițializează magistrala I2C;
5. se pornește LCD-ul;
6. se inițializează modulul RTC DS3231;
7. se inițializează cardul microSD;
8. se afișează mesajul de pornire pe LCD.

În bucla principală:

1. se verifică dacă butonul Start/Stop a fost apăsat;
2. dacă sistemul este oprit logic:
   - LED-ul este stins;
   - buzzerul este oprit;
   - LCD-ul afișează mesajul „Sistem oprit”;
3. dacă sistemul este pornit:
   - se citește valoarea analogică de la senzor;
   - valoarea ADC este convertită în procent de umiditate;
   - se stabilește dacă planta este în status „OK” sau „USCAT”;
   - se actualizează statisticile;
   - se controlează LED-ul și buzzerul;
   - se actualizează LCD-ul;
   - se salvează periodic datele pe cardul microSD;
   - se transmit periodic datele prin Bluetooth.

Funcții principale implementate

Funcție	Rol
initHardware()	Inițializează componentele hardware: pini, LCD, RTC, microSD, Bluetooth și Serial Monitor.
buttonPressedEvent()	Detectează apăsarea butonului Start/Stop folosind debounce.
readSoilAdc()	Citește valoarea analogică de la senzorul de umiditate.
convertAdcToHumidityPercent()	Transformă valoarea ADC într-un procent de umiditate.
setAlerts()	Pornește sau oprește LED-ul și buzzerul în funcție de statusul plantei.
updateLcd()	Actualizează informațiile afișate pe LCD.
getCurrentTime()	Citește data și ora de la modulul RTC DS3231.
logMeasurementToSd()	Salvează măsurătorile și statisticile pe cardul microSD.
sendBluetoothReport()	Trimite prin Bluetooth valorile curente și statisticile.
resetStats()	Resetează valorile statistice.
updateStats()	Actualizează media, minimul, maximul și numărul de măsurători.
getAverageHumidity()	Calculează media umidității.
getTrend()	Determină tendința umidității: creștere, scădere sau stabilă.
getDeltaFromFirst()	Calculează diferența față de prima măsurătoare.

Stabilirea statusului plantei

Statusul plantei este stabilit prin compararea umidității curente cu pragul definit în `config.h`.

```
currentDry = currentHumidity < HUMIDITY_THRESHOLD_PERCENT;
currentStatus = currentDry ? "USCAT" : "OK";
```

Dacă umiditatea este sub prag, LED-ul și buzzerul sunt pornite. Dacă umiditatea este peste prag, acestea sunt oprite.

```
void setAlerts(bool systemActive, bool isDry) {
  if (systemActive && isDry) {
    digitalWrite(PIN_LED, HIGH);
    digitalWrite(PIN_BUZZER, HIGH);
  } else {
    digitalWrite(PIN_LED, LOW);
    digitalWrite(PIN_BUZZER, LOW);
  }
}
```

Structuri de date pentru statistici

Statisticile sunt păstrate într-o structură numită `MoistureStats`.

```
struct MoistureStats {
```

```
  bool initialized;
  unsigned long count;
  long sum;
  int minimum;
  int maximum;
  int first;
  int previous;
  int current;
  unsigned long dryCount;
};
```

Această structură permite calcularea următoarelor informații:

- numărul total de măsurători;
- suma valorilor măsurate;
- media umidității;
- valoarea minimă;
- valoarea maximă;
- prima valoare măsurată;
- valoarea anterioară;
- valoarea curentă;
- numărul de măsurători în care solul a fost uscat.

Calculul statisticilor

La fiecare măsurătoare, funcția `updateStats()` actualizează valorile statistice.

```
void updateStats(MoistureStats &stats, int humidityPercent, bool isDry) {
  if (!stats.initialized) {
    stats.initialized = true;
    stats.first = humidityPercent;
    stats.previous = humidityPercent;
    stats.current = humidityPercent;
    stats.minimum = humidityPercent;
    stats.maximum = humidityPercent;
  } else {
    stats.previous = stats.current;
    stats.current = humidityPercent;

    if (humidityPercent < stats.minimum) {
      stats.minimum = humidityPercent;
    }

    if (humidityPercent > stats.maximum) {
      stats.maximum = humidityPercent;
    }
  }
```

```
  stats.sum += humidityPercent;
  stats.count++;

  if (isDry) {
    stats.dryCount++;
  }
}
```

Media este calculată astfel:

```
int getAverageHumidity(const MoistureStats &stats) {
  if (stats.count == 0) {
    return 0;
  }

  return (int)(stats.sum / (long)stats.count);
}
```

Calculul tendinței

Tendința este calculată prin compararea valorii curente cu valoarea anterioară.

```
const char* getTrend(const MoistureStats &stats) {
  if (!stats.initialized || stats.count < 2) {
    return "STABIL";
  }

  int diff = stats.current - stats.previous;

  if (diff > 2) {
    return "CRESTERE";
  }

  if (diff < -2) {
    return "SCADERE";
  }

  return "STABIL";
}
```

Formatul fișierului CSV

Datele sunt salvate pe cardul microSD în fișierul:

```
umid.csv
```

Header-ul fișierului este:

```
date,time,adc,humidity,status,average,min,max,trend,delta_from_first,dry_cou
nt,count
```

Exemplu de linie salvată:

```
2026-05-08,16:40:00,522,28,USCAT,45,28,72,SCADERE,-14,3,10
```

Mesaj transmis prin Bluetooth

Prin modulul HC-05 se transmite periodic un raport cu valorile curente și statisticile calculate.

```
----------------
Timp: 2026-05-08 16:40:00
ADC: 522
Umiditate: 28%
Status: USCAT
Media: 45%
Minim: 28%
Maxim: 72%
Tendinta: SCADERE
Diferenta fata de start: -14 pct
Masuratori sol uscat: 3
SD: OK
```

Funcționarea butonului Start/Stop

Butonul Start/Stop este configurat folosind rezistența internă de pull-up a microcontrollerului.

```
pinMode(PIN_BUTTON, INPUT_PULLUP);
```

Astfel:

```
buton neapăsat -> HIGH
buton apăsat   -> LOW
```

Funcția `buttonPressedEvent()` detectează apăsarea butonului și schimbă starea sistemului între pornit și oprit.

```
if (buttonPressedEvent()) {
  systemActive = !systemActive;
}
```

Timpi de execuție

Intervalele de timp sunt definite în `config.h`.

```
const unsigned long MEASURE_INTERVAL_MS = 1000;
const unsigned long LCD_INTERVAL_MS = 1000;
const unsigned long SD_LOG_INTERVAL_MS = 10000;
const unsigned long BT_SEND_INTERVAL_MS = 3000;
```

Astfel:

- senzorul este citit la fiecare 1 secundă;
- LCD-ul este actualizat la fiecare 1 secundă;
- datele sunt salvate pe microSD la fiecare 10 secunde;
- raportul Bluetooth este trimis la fiecare 3 secunde.

Concluzii

Download

Codul sursă al proiectului este disponibil pe GitHub: soil_moisture_monitor

Fişierele se încarcă pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul în care se încarcă fişierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (dacă este cazul). **Exemplu:** Dumitru Alin, 331CC → **:pm:prj2009:cc:dumitru_alin**.

Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

Bibliografie/Resurse

Listă cu documente, datasheet-uri, resurse Internet folosite, eventual grupate pe **Resurse Software** şi **Resurse Hardware**.

Export to PDF