House Monitoring System - ŞTEFAN Miruna-Andreea

Nume: Ștefan Miruna Andreea

Grupa: 334CA

Introducere

De cate ori nu s-a întâmplat ca vara sa vii acasă după o zi caniculara si sa îți dorești sa găsești aerul condiționat deja pornit? Sau sa te asiguri că mediul in care dormi respecta toate recomandările legate de nivelul de umiditate, temperatură, etc, pentru un somn odihitor si sănătos? Sau să știi ce temperatură ai în debara / pe balcon / în boxă la subsol pentru a alege unde să depozitezi borcanele cu murături?

Pentru astfel de măsurători vine în ajutor acest house monitoring system, care track-uiește temperatura si umiditatea din locuință, le afișează pe un ecran și stochează toate măsurătorile efectuate pe un card SD pentru a permite vizualizarea si efectuarea analizelor pe aceste date, iar dacă se detectează măsurarea unei temperaturi care nu se încadrează in parametrii normali, se declanșează o alarma.

Descriere generală

Listă componente:

Componentă	**Link către site**
Arduino UNO R3	Link produs
Modul Senzor de Temperatura și Umiditate DHT22	Link produs
Modul MicroSD	Link produs
Modul buzzer	Link produs
LCD 1602 cu Interfata I2C si Backlight Galben-Verde	Link produs
Breadboard 830p MB-102	Link produs
Fire Tata-Tata, 30 cm	Link produs
Fire Mama-Tata, 15 cm	Link produs
Card MicroSD 16GB	Link produs

Rezistor 5 kOhm	

Datele vin de la senzorul de temperatura / umiditate in MCU, de unde merg catre cardul SD pentru stocare, catre ecran pentru afisare si tot din MCU porneste semnal catre buzzer daca valorile citite depasesc anumite praguri, dupa cum sugereaza si schema de mai jos.

Hardware Design

Diagrama desenată in Wokwi:

Observație: La modulul pentru card microSD, pinul DI este echivalent cu MOSI de pe modulul real, iar pinul DO cu MISO.

Descrierea legăturilor făcute și a pinilor utilizați:

- modulul microSD (comunică prin SPI ⇒ am folosit pinii dedicați de pe plăcuță, conform instrucțiunilor de pe site-ul de unde l-am cumpărat (vezi tabel componente din secțiunea anterioară))
  - CS - pinul ~10 (~D10 / PB2 / SS) (firul galben)
  - SCK - pinul 13 (D13 / PB5 / SCK) (firul albastru)
  - MOSI - pinul ~11 (~D11 / PB3 / MOSI) (firul alb)
  - MISO - pinul 12 (D12 / PB4 / MISO) (firul verde)
  - VCC - 5V (firul roșu)
  - GND - GND (firul negru)
- ecranul LCD 1602 (comunică prin I2C ⇒ am folosit pinii dedicați de pe plăcuță, conform instrucțiunilor de pe site-ul de unde l-am cumpărat (vezi tabel componente din secțiunea anterioară))
  - GND - GND (firul gri)
  - VCC - 5V (firul portocaliu)
  - SDA - A4 (A4 / D18 / PC4) (firul maro)
  - SCL - A5 (A5 / D19 / PC5) (firul mov)
- senzorul de temperatură și umiditate DHT22
  - (-) - GND (firul negru)
  - OUT - pinul 8 al plăcuței (D8 / PB0) (am ales dintre pinii digitali nerezervați) (firul alb)
  - (+) - 5V (firul roșu)
  - firul gri = fir de comunicație digitală între Arduino și senzor (permite transmisia de date între cele două)
  - rezistență de pull-up (5 KOhmi)
- buzzer-ul
  - (-) - GND (firul negru)
  - (+) - pinul ~9 al plăcuței (~D9 / PB1) (am ales un pin digital care avea PWM) (firul roșu)

Schema electrică:

Observație: Schema electrică a fost făcută în Tinkercad, unde nu există modul microSD, așa că am folosit în locul acestuia un 8 pin header care să simuleze un modul microSD, unde

pinul 1 = CS

pinul 2 = SCK

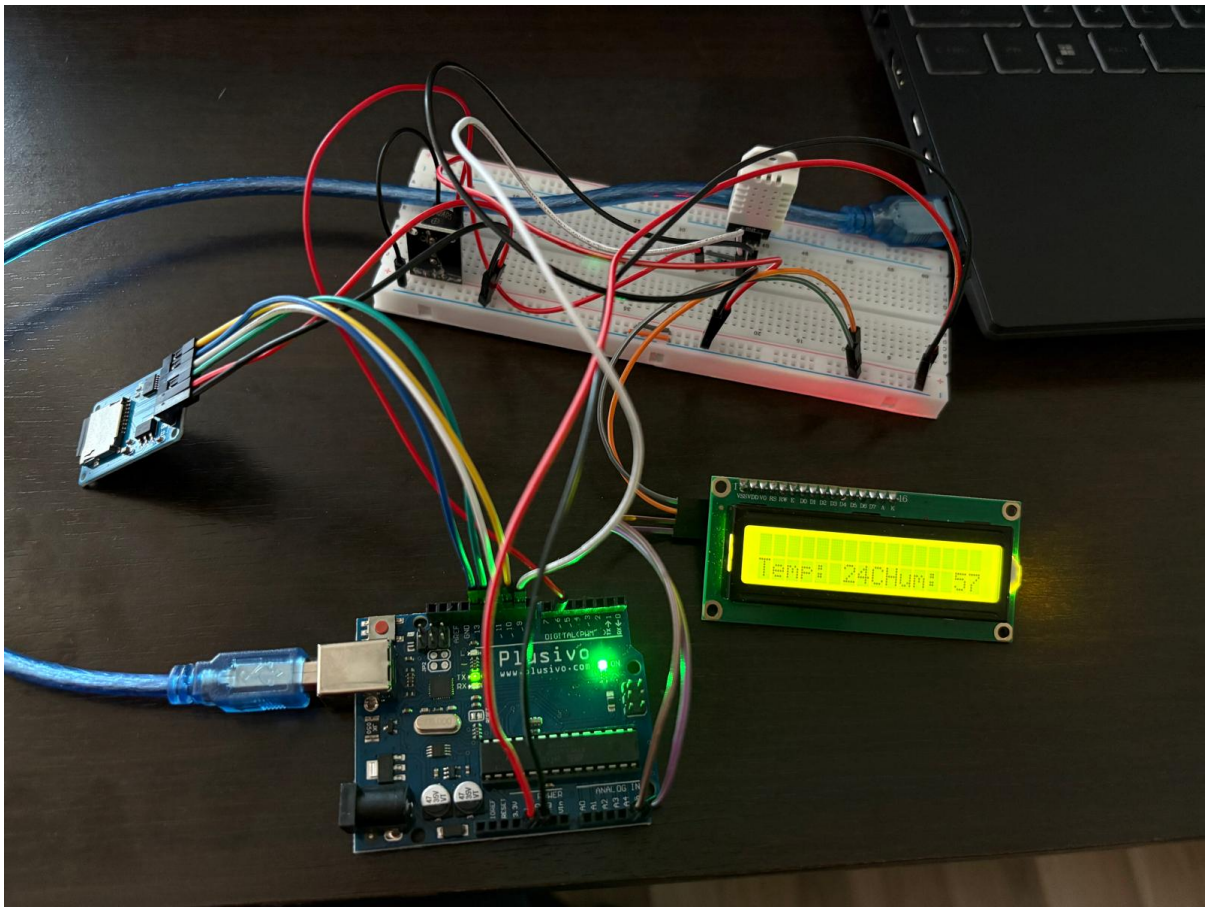
pinul 3 = MOSI

pinul 4 = MISO

pinul 5 = VCC

pinul 6 = GND

Asa arata proiectul in realitate:



Software Design

- mediu de dezvoltare: Arduino IDE

- biblioteci şi surse 3rd-party:

- avr - colectie de headere si functii specifice pentru programarea directa a microcontrollerelor AVR
  - avr/io - definirea registrelor hardware (PORTx, DDRx, TCCRn, UBRRn, etc.
  - avr/interrupt - gestionarea intreruperilor (sei())
- LiquidCrystal_I2C:
  - permite comunicarea cu LCD-ul prin doar 2 fire: SDA (Serial Data) si SCL (Serial Clock), folosind protocolul I2C
  - intern, transformă comenzile LCD tradiționale precum set cursor, write,etc. în semnale transmise pe magistrala I2C
  - extensie a bibliotecii standard LiquidCrystal din Arduino, cu diferenta ca în locul folosirii pinilor paraleli , afisajele LCD sunt controlate prin intermediul interfeței I2C.
- SD:
  - biblioteca din pachetul oficial Arduino IDE utilizata pentru a facilita comunicarea cu un card SD si pentru a scrie si citi fisiere pe / de pe el
  - intern, foloseste interfata SPI pentru comunicarea cu cardul
- TroykaDHT
  - biblioteca dezvoltata de Amperka (Troyka Modules) pentru a facilita lucrul cu senzorii DHT11 si DHT22
  - permite creare unui obiect pentru senzorul DHT22, initializarea sa, citirea datelor de la senzor (temperatura si umiditatea) si verificarea daca citirea a reusit sau nu.

Scheletul proiectului:

- am folosit 2 timere: timer1 pentru pwm si timer2 pentru numararea secundelor (inlocuirea functiei delay());

- la fiecare 2 secunde se citesc temperatura si umiditatea si se afiseaza atat pe seriala, cat si pe ecran si se si stocheaza tot istoricul de masuratori pe microSD;

- daca la un moment dat temperatura masurata depaseste ALERT_TEMP, buzzer-ul incepe sa cante (daca deja canta, nu ia melodia de la inceput, ci continua), iar daca temperatura scade din nou sub ALERT_TEMP, buzzer-ul se opreste din cantat. Am setat ALERT_TEMP la 26 de grade (ceea ce era o temperatura rezonabila pentru mediul in care am facut proiectul, dar aceasta se poate modifica, evident, la un alt threshold.);

- pe masura ce trece timpul si buzzer-ul canta, se actualizeaza nota (se trece la nota urmatoare pentru a forma o melodie completa).

Notiuni din laboratoare:

- Lab 1: USART (pentru scrierea datelor despre temperatura si umiditate pe seriala)

- Lab 2: Intreruperi (numararea secundelor pentru scrierea functiei de wait())

- Lab 3: PWM (pentru a face buzzer-ul sa cante atunci cand este depasita temperatura ALERT_TEMP)

- Lab 5: SPI (protocol folosit pentru scrierea si citirea istoricului de date in fisierul de pe cardul microSD)

- Lab 6: I2C (protocol folosit pentru comunicarea cu ecranul LCD pe care sunt afisate temperatura si

umiditatea).

Codul complet este pe Github, la link-ul urmator: https://github.com/miruna-stefan/Proiect-pm.

Rezultate Obţinute

Demo-ul poate fi urmatit la link-ul urmator: https://www.youtube.com/watch?v=eLbyWtaBuJE

Download

Codul complet este pe Github, la link-ul urmator: https://github.com/miruna-stefan/Proiect-pm.

Jurnal

- 28 aprilie - Alegere temă proiect
- 5 mai - Primire piese de SigmaNortec
- 7 mai - Am ridicat restul de piese de la Optimus Digital
- 8 mai - Am început asamblarea pieselor și am constat că ecranul nu avea backlight galben, cum se specifica pe site, motiv pentru care scrisul de pe ecran nu se vedea prea bine, iar senzorul DHT11 nu funcționa nici el corect (uneori pinii nu făceau contact bine și arăta 0 și la temperatură și la umiditate în loc de datele reale)
- 9 mai - Am înlocuit componentele cu probleme: în loc de DHT11, am cumpărat un DHT22 (doar acesta mai era disponibil pe Optimus) și am schimbat ecranul cu unul identic, dar care funcționează conform specificațiilor de pe site
- 11 mai - finalizare asamblare hardware
- 20 mai - finalizare software

Bibliografie/Resurse

- Datasheet Arduino UNO R3: https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf
- Model pentru legarea firelor pentru LCD: https://www.optimusdigital.ro/ro/optoelectronice-lcd-uri/62-lcd-1602-cu-interfata-i2c-si-backlight-gal

ben-verde.html?search_query=lcd+1602&results=17
- Model pentru legarea firelor pentru modulul MicroSD: https://sigmanortec.ro/Modul-MicroSD-p126079625

Export to PDF