# MINISTERUL EDUCAŢIEI, CULTURII ȘI CERCETĂRII

**Universitatea Tehnică a Moldovei**

# Imagine similarăFacultatea Calculatoare Informatică şi Microelectronică Departamentul Ingineria Software și Automatică

Raport

**la disciplina Internetul lucrurilor IOT**

**Tema: Senzori**

A efectuat: st.gr. TI-204 Agatiev Dumitru A verificat: asist.univ. Litra Dinu

Chișinău 2023

# Definirea problemei 1

Realizarea unei aplicații pentru MCU care va citi datele de la un senzor de umiditate. Aplicația va efectua

1. Citirea de datelor de pe senzor;
2. Prelucrarea datelor în valori reale;
3. Medierea datelor prin 2 tehnici;
4. Afișarea datelor pe LCD și în serial.

# Obiective

* + configurarea aplicației pentru citirea datelor de pe senzor;
  + crearea schemei și codului conform sarcinii propuse;
  + înțelegerea sistemului creat și procesele ce se întâmplă.

# Introducere

Un senzor este un dispozitiv izolat structural care conține unul sau mai mulți traductori de măsurare primari. Senzorul este proiectat pentru a genera un semnal de informație de măsurare într-o formă convenabilă pentru transmisie, conversie ulterioară, procesare și (sau) stocare, dar nu direct perceptibilă de către un observator.

Senzorul poate conține în plus traductoare de măsurare intermediare, precum și o măsură. Senzorul poate fi amplasat la o distanță considerabilă de dispozitivul care primește semnalele acestuia. Cu un raport normalizat al valorii cantității la ieșirea senzorului cu valoarea corespunzătoare a cantității de intrare, senzorul este un instrument de măsură.

# Metode și materiale

Materiale Necesare:

* Arduino Board (de exemplu, Arduino Uno);
* breadboard și fire de conexiune;
* senzor de umiditate;
* editor Arduino IDE instalat pe un calculator.

Metoda de Implementare:

* conexiuni hardware:
  + conectarea senzorului de umiditate la un pin analogic al Arduino;
  + conectare LCD la pinii digitali ai Arduino.
* scrierea codului în Arduino IDE:
  + definirea variabilelor și pinilor corespunzători pentru senzorul de umiditate și LCD;
  + implementare funcții pentru citirea temperaturii de la senzor și aplicarea tehnicilor de "sare și piper" și medie ponderată;
  + afișarea valorilor pe LCD și a umiditații curente în serial.
* testarea și debugging:
  + încărcare cod pe Arduino folosind Arduino IDE;
  + monitorizare comportament senzorului;
  + identificare și rezolvare eventualelor erori în cod.

# Rezultate

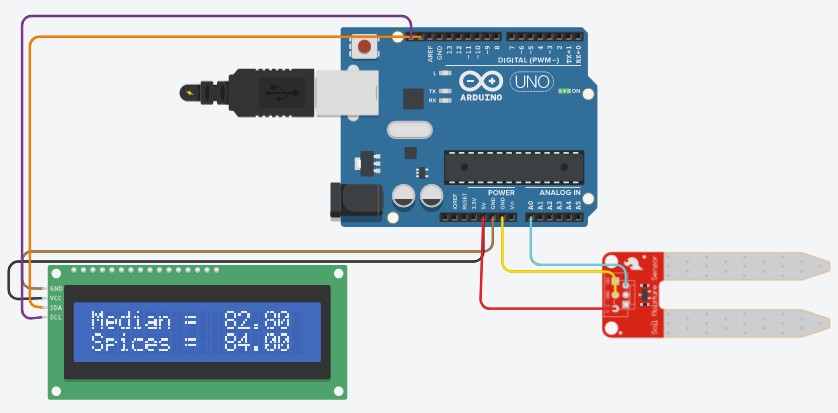


Figura 1 – Asamblarea circuitului pentru senzorul de umiditate

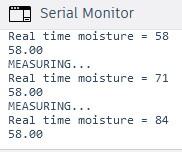


Figura 2 – Datele senzorului de umiditate

# Definirea problemei 2

Realizarea unei aplicații pentru MCU care va citi datele de la un senzor de temperatură. Aplicația va efectua

1. Citirea de datelor de pe senzor;
2. Prelucrarea datelor în valori reale;
3. Medierea datelor prin 2 tehnici;
4. Afișarea datelor pe LCD și în serial.

# Obiective

* + configurarea aplucației pentru pentru citirea datelor de pe senzor;
  + crearea schemei și codului conform sarcinii propuse;
  + înțelegerea sistemului creat și procesele ce se întâmplă.

# Introducere

În prezent, termenii „senzor” și „detector” sunt folosiți interschimbabil pentru a se referi la un traductor de măsurare care îndeplinește funcțiile de a detecta o cantitate de intrare și de a genera un semnal de măsurare, deși termenul de „detector” se concentrează pe percepția cantității de intrare, și termenul

„senzor” privind formarea și emiterea unui semnal de măsurare.

Senzorii multifuncționali pot percepe și converti mai multe cantități de intrare și, în plus față de funcția principală (detectarea unei cantități și generarea unui semnal de măsurare), efectuează o serie de funcții suplimentare, cum ar fi funcții de filtrare, procesare a semnalului etc.

# Metode și materiale

Materiale Necesare:

* Arduino Board (de exemplu, Arduino Uno);
* fire de conexiune;
* senzor de temperatură
* editor Arduino IDE instalat pe un calculator.

Metoda de Implementare:

* conexiuni hardware:
  + conectarea senzorului de temperatură la un pin analogic al Arduino;
  + conectare LCD la pinii digitali ai Arduino.
* scrierea codului în Arduino IDE:
  + definire variabilele și pinii corespunzători pentru senzorul de temperatură și LCD;
  + implementare funcții pentru citirea temperaturii de la senzor și aplicarea tehnicilor de "sare și piper" și medie ponderată;
  + afișarea valorilor pe LCD și a temperaturii curente în serial.
* testarea și debugging:
  + încărcare cod pe Arduino folosind Arduino IDE;
  + monitorizare comportament senzorului;
  + identificare și rezolvare eventualelor erori în cod.

# Rezultate

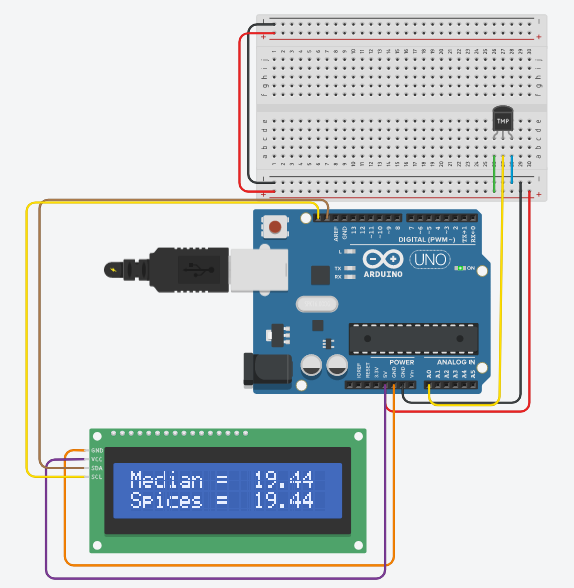


Figura 3 – Asamblarea circuitului pentru senzorul de temperatură

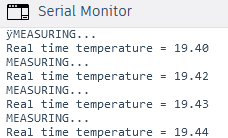


Figura 4 – Datele senzorului de temperatură

## Anexa 1 – Codul sursă sarcina 1

#include <Adafruit\_LiquidCrystal.h>

#define sensor A0 Adafruit\_LiquidCrystal lcd(0); void sw(float a[]) {

for (int i = 0; i < 5; i++) { a[i] = a[i + 1];

}

}

void disp(float t1, float t2) { lcd.setCursor(10, 0); lcd.print(t1); lcd.setCursor(10, 1); lcd.print(t2);

}

float getMoisture() {

float voltage = (analogRead(sensor) \* (5.0 / 1023.0)); int moisture = map(voltage \* 100, 0, 428, 0, 100); Serial.print("Real time moisture = "); Serial.println(moisture);

return moisture;

}

float weighted\_median(float moists[]) { int weight = 2;

float moisture = ((weight \* moists[0] + weight \* moists[1] +

weight \* moists[2] + weight \* moists[3] + weight \* moists[4]) / (weight \* 5));

return moisture;

}

void swap(float \*p, float \*q) { float t;

t = \*p;

\*p = \*q;

\*q = t;

}

float salt\_and\_pepper(float moists[]) { int i, j, n = 5;

float a[5];

for (i = 0; i < 5; i++) { a[i] = moists[i];

}

for (i = 0; i < n - 1; i++) {

for (j = 0; j < n - i - 1; j++) {

if (a[j] > a[j + 1])

swap(&a[j], &a[j + 1]);

}

}

int med\_n = (n + 1) / 2 - 1; float moisture = a[med\_n]; Serial.println(moisture); return moisture;

}

void setup() { pinMode(sensor, INPUT); Serial.begin(9600); lcd.begin(16, 2);

lcd.setCursor(0, 0); lcd.print("Median = "); lcd.setCursor(0, 1); lcd.print("Spices = ");

}

int i = 0;

void loop() { float moists[5];

Serial.println("MEASURING...");

if (i < 4) {

moists[i] = getMoisture();

}

if (i >= 4) { sw(moists);

moists[4] = getMoisture();

}

float val1 = weighted\_median(moists); float val2 = salt\_and\_pepper(moists); disp(val1, val2);

delay(1000); i++;

}

## Anexa 2 – Codul sursă sarcina 2

#include <Adafruit\_LiquidCrystal.h>

// Definirea pinului conectat la senzorul de temperatură #define sensor A0

// Inițializarea bibliotecii pentru LCD Adafruit\_LiquidCrystal lcd(0);

// Funcția pentru deplasarea valorilor în array void sw(float a[]) {

for (int i = 0; i < 5; i++) { a[i] = a[i + 1];

}

}

// Funcția pentru afișarea valorilor pe LCD void disp(float t1, float t2) {

lcd.setCursor(10, 0); lcd.print(t1); lcd.setCursor(10, 1); lcd.print(t2);

}

// Funcția pentru conversia de la voltaj la temperatură float convertToTemperature(float voltage) {

// Specificațiile senzorului float sensorSensitivity = 10.0;

// Conversie de la voltaj la temperatură

float temperature = (voltage - 500) / sensorSensitivity; return temperature;

}

// Funcția pentru achiziționarea temperaturii de la senzor float getTemperature() {

// Citirea valorii analogice de la senzor și conversia la voltaj int sensorValue = analogRead(sensor);

float voltage = (sensorValue \* 5.0) / 1023.0; // 5V este tensiunea de referință

// Apelul funcției pentru conversia la temperatură float temperature = convertToTemperature(voltage); Serial.print("Real time temperature = "); Serial.println(temperature);

return temperature;

}

// Funcția pentru calcularea mediei ponderate float weighted\_median(float temps[]) {

int weight = 2;

float temperature = ((weight \* temps[0] + weight \* temps[1] +

weight \* temps[2] + weight \* temps[3] + weight \* temps[4]) / (weight \* 5));

return temperature;

}

// Funcția pentru interschimbarea a două valori void swap(float \*p, float \*q) {

float t; t = \*p;

\*p = \*q;

\*q = t;

}

// Funcția pentru filtrul "sare și piper" float salt\_and\_pepper(float temps[]) {

int i, j, n = 5; float a[5];

for (i = 0; i < 5; i++) { a[i] = temps[i];

}

for (i = 0; i < n - 1; i++) { // Sortarea array-ului for (j = 0; j < n - i - 1; j++) {

if (a[j] > a[j + 1])

swap(&a[j], &a[j + 1]);

}

}

int med\_n = (n + 1) / 2 - 1; // Alegerea valorii din mijloc float temperature = a[med\_n];

return temperature;

}

void setup() {

// Configurarea modului pinului conectat la senzor

pinMode(sensor, INPUT); Serial.begin(9600); lcd.begin(16, 2);

lcd.setCursor(0, 0); lcd.print("Median = "); lcd.setCursor(0, 1); lcd.print("Spices = ");

}

int i = 0; void loop() {

float temps[5];

Serial.println("MEASURING..."); if (i < 4) {

temps[i] = getTemperature();

}

if (i >= 4) { sw(temps);

temps[4] = getTemperature();

}

float temp1 = weighted\_median(temps); float temp2 = salt\_and\_pepper(temps);

// Afișarea temperaturilor pe LCD disp(temp1, temp2);

delay(2000); i++;

}

## Concluzie

În urma acestui laborator, am avut o ocazie semnificativă de a investiga procesul de colectare și procesare a datelor furnizate de un senzor de temperatură și umiditate. Am acumulat cunoștințe esențiale despre cum să conectăm mai multe senzori la o placa de dezvoltare Arduino, să citim valorile analogice și să le transformăm în date reale. În plus, am experimentat cu două metode de prelucrare a datelor: calculul mediei ponderate și utilizarea unui filtru de tip "sare și piper".

Cu ajutorul unui ecran LCD pentru a afișa rezultatele, am adus o dimensiune practică acestui proiect, permitându-ne să monitorizăm în timp real datele prelucrate. Acest aspect subliniază importanța fundamentală a procesului de achiziție, prelucrare și prezentare a datelor obținute de la senzori, cu

aplicații semnificative în diverse domenii, cum ar fi monitorizarea mediului, automatizarea industrială sau dezvoltarea dispozitivelor inteligente.

În concluzie, acest laborator mi-a furnizat o experiență practică valoroasă în ceea ce privește achiziția și prelucrarea datelor de la senzori, evidențiind aplicabilitatea conceptelor teoretice în dezvoltarea soluțiilor concrete. Acesta a consolidat nu doar înțelegerea mea a tehnologiei, ci și capacitatea mea de a aplica cunoștințele teoretice în proiecte reale.