FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

Sistem multisenzor pentru monitorizarea reacțiilor fiziologice asociate stresului, inspirat din principiile poligrafului

Realizat de:

- Gavriş Alexandru-Cătălin

Grupă:30222, CTI

Anul universitar: 2025 - 2026

Cuprins:

Cuprins

LISTA DE ABREVIERI ȘI SIMBOLURI

În cadrul prezentei lucrări sunt utilizate următoarele abrevieri, acronime și simboluri, pentru a facilita înțelegerea conceptelor tehnice și a implementării practice a sistemului realizat:

ADC (Analog to Digital Converter) – convertor analog-digital utilizat pentru transformarea semnalelor analogice în valori digitale procesabile de microcontroler.

Arduino UNO – placă de dezvoltare bazată pe microcontrolerul ATmega328P, utilizată ca unitate centrală de control a sistemului.

BPM (Beats Per Minute) – numărul de bătăi ale inimii pe minut, parametru utilizat pentru evaluarea ritmului cardiac.

CPU (Central Processing Unit) – unitatea centrală de procesare a microcontrolerului.

DAC (Digital to Analog Converter) – convertor digital-analog (menționat teoretic, neutilizat direct în acest proiect).

DS18B20 – senzor digital de temperatură, utilizat pentru măsurarea temperaturii corporale periferice.

FSM (Finite State Machine) – mașină cu stări finite, concept utilizat implicit în logica de funcționare a aplicației.

GND (Ground) – masă electrică, referință de tensiune 0 V în circuit.

GSR (Galvanic Skin Response) – răspuns galvanic al pielii, parametru care reflectă variațiile conductanței electrice ale pielii în funcție de activitatea sistemului nervos simpatic.

HR (Heart Rate) – ritm cardiac, exprimat în BPM.

I2C (Inter-Integrated Circuit) – protocol de comunicație serială utilizat pentru conectarea modulului LCD și a senzorului MAX30102.

IR (Infrared) – radiație infraroșie utilizată de senzorul MAX30102 pentru detectarea variațiilor de volum sanguin.

LCD (Liquid Crystal Display) – afișaj cu cristale lichide utilizat pentru prezentarea informațiilor către utilizator.

MCU (Microcontroller Unit) – microcontrolerul care coordonează funcționarea sistemului.

MAX30102 – senzor optic utilizat pentru măsurarea pulsului și a variațiilor de oxigenare a sângelui.

PWM (Pulse Width Modulation) – modulație în lățime de puls, tehnică utilizată pentru controlul semnalelor digitale.

RESP (Respiration) – parametru asociat respirației, monitorizat indirect prin intermediul senzorului piezoelectric.

SCL (Serial Clock Line) – linie de ceas utilizată în comunicația I2C.

SDA (Serial Data Line) – linie de date utilizată în comunicația I2C.

SPI (Serial Peripheral Interface) – protocol de comunicație serială (menționat teoretic).

TEMP (Temperature) – temperatura corporală măsurată cu ajutorul senzorului DS18B20.

USB (Universal Serial Bus) – interfață utilizată pentru alimentare și comunicație cu calculatorul.

VCC – tensiune de alimentare pozitivă a circuitului.

Piezo / Senzor piezoelectric – element utilizat pentru detectarea variațiilor mecanice produse de mișcările respiratorii.

Scor de stres – indicator numeric compozit (0–100) rezultat din corelarea parametrilor fiziologici măsurați, utilizat pentru estimarea nivelului de stres al utilizatorului.

1.INTRODUCERE

1.1 Justificarea alegerii temei

Acest proiect a fost ales pentru a propune o soluție experimentală, accesibilă și ușor de implementat pentru monitorizarea reacțiilor fiziologice asociate stresului. Stresul reprezintă un fenomen frecvent în viața cotidiană, influențând atât performanța, cât și starea de sănătate a individului. Reacțiile fiziologice generate de stres pot fi observate prin modificări ale unor parametri biologici precum ritmul cardiac, conductanța pielii, respirația și temperatura corporală periferică.

Alegerea acestei teme este justificată de interesul crescut pentru analiza obiectivă a reacțiilor fiziologice și de posibilitatea implementării unui sistem funcțional utilizând componente accesibile și ușor de integrat într-o platformă embedded.

1.2 Scopul lucrării

Scopul principal al proiectului este realizarea unui **sistem multisenzor pentru monitorizarea reacțiilor fiziologice asociate stresului, inspirat din principiile poligrafului**, utilizând o platformă hardware accesibilă, respectiv Arduino UNO.

Lucrarea urmărește reducerea interpretărilor subiective și oferirea unei metode obiective de observare și analiză a modificărilor fiziologice, prin intermediul senzorilor și al prelucrării software a datelor achiziționate.

1.3 Beneficiile și motivația soluției propuse

Principalele beneficii și motive care au stat la baza alegerii acestei soluții sunt următoarele:

- **Monitorizare multisenzor** – Sistemul utilizează mai mulți senzori (puls, GSR, respirație și temperatură), oferind o imagine mai completă asupra reacțiilor fiziologice asociate stresului.
- **Calibrare individuală** – Valorile măsurate sunt raportate la un nivel de referință specific fiecărui utilizator, ceea ce crește relevanța și corectitudinea rezultatelor.

- **Automatizare** – Achiziția, procesarea și afișarea datelor sunt realizate automat, fără intervenție manuală.
- **Afișare intuitivă** – Informațiile sunt prezentate în timp real pe un ecran LCD, fiind ușor de interpretat de către utilizator.
- **Cost redus** – Sistemul este bazat pe platforma Arduino, fiind o soluție accesibilă, low-cost și ușor de prototipat.

1.4 Importanța și actualitatea temei

Importanța acestui proiect este susținută de interesul tot mai mare pentru tehnologiile de monitorizare biometrică și pentru aplicațiile embedded din domeniul sănătății și al analizei comportamentale. În prezent, dispozitivele wearable și sistemele de monitorizare a stării fiziologice sunt utilizate pe scară largă pentru evaluarea stresului, a oboselii și a nivelului de activare al organismului.

Proiectul se aliniază tendințelor actuale din domeniul sistemelor inteligente și al Internetului Lucrurilor (IoT), unde monitorizarea continuă a parametrilor fiziologici joacă un rol important. Soluția propusă oferă un exemplu practic de integrare a senzorilor biomedicali într-un sistem embedded funcțional, cu aplicații educaționale și experimentale.

1.5 Originalitatea lucrării

Deși principiile de măsurare ale parametrilor fiziologici sunt cunoscute și utilizate în sistemele de tip poligraf, originalitatea lucrării constă în integrarea acestora într-un sistem experimental compact, realizat cu componente accesibile și adaptat scopurilor educaționale.

Abordarea software utilizată pentru calibrarea individuală, filtrarea semnalelor și calcularea unui scor de stres compozit contribuie la creșterea relevanței și stabilității rezultatelor obținute.

1.6 Aplicabilitatea practică și limitările abordării

Din punct de vedere al aplicabilității, sistemul poate fi utilizat ca platformă de studiu pentru înțelegerea funcționării senzorilor biomedicali și a principiilor de bază ale sistemelor de tip poligraf. De asemenea, proiectul poate reprezenta un

punct de plecare pentru dezvoltări ulterioare, cum ar fi îmbunătățirea algoritmilor de analiză sau adaptarea sistemului pentru aplicații portabile.

Este important de menționat că lucrarea nu urmărește detectarea directă a minciunii, ci doar monitorizarea reacțiilor fiziologice asociate stresului, iar rezultatele obținute trebuie interpretate într-un mod orientativ.

2.CONTEXT TEORETIC PRIVIND STRESUL FIZIOLOGIC

Acest capitol prezintă noțiunile teoretice necesare pentru înțelegerea reacțiilor fiziologice asociate stresului, precum și rolul principalilor parametri biologici utilizați în sistemul realizat. Informațiile prezentate constituie baza teoretică pe care se sprijină alegerea senzorilor și implementarea practică a sistemului multisenzor.

▪ *2.1 Noțiuni generale despre stres*

Stresul poate fi definit ca răspunsul organismului la stimuli interni sau externi care perturbă starea de echilibru fiziologic, numită homeostazie. Acești stimuli, denumiți factori stresori, pot fi de natură fizică, psihologică sau emoțională și pot apărea atât în situații reale, cât și anticipate.

Din punct de vedere biologic, stresul reprezintă un mecanism adaptativ esențial, care permite organismului să reacționeze rapid în situații de pericol sau solicitare intensă. Cu toate acestea, expunerea prelungită la stres poate conduce la efecte negative asupra sănătății, afectând funcționarea sistemelor cardiovasculare, endocrine și nervoase.

▪ *2.2 Sistemul nervos autonom și răspunsul la stres*

Sistemul nervos autonom este responsabil de controlul funcțiilor involuntare ale organismului, precum ritmul cardiac, respirația, secreția glandelor sudoripare și reglarea temperaturii corporale. Acesta este alcătuit din două componente principale: sistemul nervos simpatic și sistemul nervos parasimpatic.

În situații de stres, sistemul nervos simpatic este predominant activat, determinând creșterea ritmului cardiac, accelerarea respirației, creșterea transpirației și modificări ale circulației periferice. Aceste reacții pregătesc organismul pentru un răspuns rapid, cunoscut sub denumirea de reacție de tip „luptă sau fugi”.

Sistemul nervos parasimpatic are rolul de a readuce organismul la starea de repaus după încetarea stimulului stresor. Dezechilibrul între cele două componente ale sistemului nervos autonom poate conduce la apariția stresului cronic.

▪ *2.3 Ritmul cardiac ca indicator al stresului*

Ritmul cardiac este unul dintre cei mai utilizați parametri fiziologici pentru evaluarea nivelului de stres. Activarea sistemului nervos simpatic determină creșterea frecvenței cardiace, fenomen observabil în situații de stres emoțional sau cognitiv.

Măsurarea ritmului cardiac se realizează, în mod uzual, în bătăi pe minut (BPM) și poate fi efectuată prin metode optice sau electrice. În sistemele experimentale, senzorii optici sunt preferați datorită caracterului lor neinvaziv și ușurinței în utilizare.

Este important de menționat faptul că valorile ritmului cardiac diferă de la un individ la altul, motiv pentru care interpretarea acestui parametru necesită utilizarea unei etape de calibrare individuală.

▪ *2.4 Conductanța electrică a pielii (GSR)*

Conductanța electrică a pielii, cunoscută sub denumirea de răspuns galvanic al pielii (GSR – *Galvanic Skin Response*), reprezintă un indicator direct al activității sistemului nervos simpatic. Acest parametru este influențat de activitatea glandelor sudoripare, care determină variații ale rezistenței electrice a pielii.

În condiții de stres, activitatea glandelor sudoripare crește, chiar și în absența transpirației vizibile, ceea ce conduce la o creștere a conductanței pielii. Din acest motiv, GSR este unul dintre cei mai sensibili indicatori ai stresului emoțional și este utilizat frecvent în sistemele de tip poligraf.

Semnalul GSR necesită filtrare și procesare software pentru separarea variațiilor lente (baseline) de cele rapide (phasic), asociate reacțiilor emoționale.

2.5 Respirația și modificările asociate stresului

Respirația este un parametru fiziologic puternic influențat de starea emoțională a individului. În situații de stres, respirația devine mai rapidă și mai superficială, iar frecvența mișcărilor respiratorii poate crește semnificativ.

Monitorizarea respirației poate fi realizată indirect prin utilizarea senzorilor piezoelectrici, care detectează variațiile mecanice produse de mișcările toracice sau abdominale. Analiza acestui semnal permite identificarea episoadelor de respirație accelerată sau neregulată, caracteristice stărilor de stres.

2.6 Temperatura corporală periferică și stresul

Temperatura corporală periferică reprezintă un parametru complementar în evaluarea stresului. Activarea sistemului nervos simpatic determină vasoconstricție la nivel periferic, ceea ce poate conduce la o scădere ușoară a temperaturii măsurate la extremități.

Deși variațiile de temperatură sunt mai lente comparativ cu alți parametri fiziologici, acestea oferă informații suplimentare privind starea generală a organismului și pot fi utilizate în corelație cu ceilalți indicatori de stres.

2.7 Necesitatea abordării multisenzor

Evaluarea nivelului de stres pe baza unui singur parametru fiziologic poate conduce la interpretări incomplete sau eronate. Din acest motiv, abordarea multisenzor este preferată, deoarece permite corelarea mai multor semnale biologice și obținerea unei estimări mai robuste a stării fiziologice.

Sistemul prezentat în această lucrare utilizează o abordare multisenzor inspirată din principiile poligrafului, integrând mai mulți parametri fiziologici într-un scor de stres orientativ. Această metodă oferă o perspectivă mai completă asupra reacțiilor organismului, fără a formula concluzii absolute.

3.SENZORI UTILIZAȚI ÎN SISTEM

În cadrul sistemului multisenzor pentru monitorizarea reacțiilor fiziologice asociate stresului, inspirat din principiile poligrafului, au fost utilizați mai mulți senzori biomedicali, fiecare având rolul de a măsura un parametru fiziologic relevant pentru evaluarea nivelului de stres. Alegerea senzorilor a fost realizată astfel încât sistemul să permită corelarea mai multor reacții fiziologice, obținându-se o estimare mai robustă și mai relevantă a stării organismului.

- ***3.1 Senzorul de puls – MAX30102***

3.1.1 Rolul senzorului în sistem

Senzorul MAX30102 este utilizat pentru măsurarea ritmului cardiac, exprimat în bătăi pe minut (BPM). Ritmul cardiac reprezintă unul dintre cei mai importanți indicatori fiziologici ai stresului, fiind direct influențat de activarea sistemului nervos simpatic. În situații de stres, frecvența cardiacă crește, fenomen ușor de observat și cuantificat.

În sistemul realizat, senzorul de puls contribuie la identificarea variațiilor ritmului cardiac față de o valoare de referință stabilită prin calibrare individuală.

3.1.2 Principiul de funcționare

MAX30102 este un senzor optic care funcționează pe baza principiului fotopletismografiei (PPG). Acesta utilizează o sursă de lumină infraroșie și un fotodetector pentru a măsura variațiile volumului sanguin la nivel periferic, cauzate de contracțiile inimii.

Semnalul optic obținut este prelucrat digital pentru a detecta bătăile inimii și pentru a calcula ritmul cardiac.

3.1.3 Caracteristici tehnice relevante

Principalele caracteristici ale senzorului MAX30102 sunt:

- măsurare neinvazivă a pulsului;
- comunicație digitală prin interfața I2C;
- consum redus de energie;

- integrare facilă cu platforme embedded precum Arduino.

3.1.4 Motivul alegerii senzorului

MAX30102 a fost ales datorită preciziei sale, ușurinței în utilizare și compatibilității cu platforma Arduino UNO. De asemenea, caracterul său neinvaziv îl face potrivit pentru aplicații experimentale și educaționale.

3.1.5 Limitări și observații

Citirile senzorului pot fi influențate de mișcările utilizatorului sau de poziționarea incorectă a degetului. Din acest motiv, sistemul utilizează filtrare software și o etapă de calibrare pentru îmbunătățirea stabilității măsurătorilor.

3.2 Senzorul GSR (Galvanic Skin Response)

3.2.1 Rolul senzorului în sistem

Senzorul GSR este utilizat pentru măsurarea conductanței electrice a pielii, un parametru extrem de sensibil la activitatea sistemului nervos simpatic. Acesta reprezintă unul dintre cei mai importanți indicatori utilizați în sistemele de tip poligraf.

În cadrul sistemului realizat, senzorul GSR are un rol central în evaluarea reacțiilor emoționale asociate stresului.

3.2.2 Principiul de funcționare

Măsurarea GSR se realizează prin plasarea a doi electrozi pe suprafața pielii, între care se aplică o tensiune de referință. Variațiile de conductanță electrică sunt cauzate de activitatea glandelor sudoripare, care crește în condiții de stres.

Semnalul obținut conține atât variații lente (baseline), cât și variații rapide (phasic), asociate reacțiilor emoționale.

3.2.3 Caracteristici tehnice relevante

Caracteristicile principale ale senzorului GSR includ:

- ieșire analogică;

- sensibilitate ridicată la stres emoțional;
- integrare ușoară cu ADC-ul microcontrolerului;
- cost redus.

3.2.4 Motivul alegerii senzorului

Senzorul GSR a fost ales datorită relevanței sale ridicate în evaluarea stresului și utilizării sale frecvente în sistemele de tip poligraf. Acesta oferă informații valoroase despre reacțiile emoționale ale utilizatorului.

3.2.5 Limitări și observații

Valorile GSR pot varia semnificativ de la un individ la altul și pot fi influențate de condițiile de mediu sau de poziționarea electrozilor. Din acest motiv, este necesară calibrarea individuală și filtrarea semnalului.

3.3 Senzorul de temperatură – DS18B20

3.3.1 Rolul senzorului în sistem

Senzorul DS18B20 este utilizat pentru măsurarea temperaturii corporale periferice. Acest parametru oferă informații suplimentare privind reacțiile fiziologice asociate stresului, în special în urma vasoconstricției periferice.

3.3.2 Principiul de funcționare

DS18B20 este un senzor digital de temperatură care utilizează protocolul OneWire pentru transmiterea datelor. Acesta oferă valori digitale precise, eliminând necesitatea conversiei analog-digitale.

3.3.3 Caracteristici tehnice relevante

- rezoluție configurabilă;
- precizie ridicată;
- comunicație digitală OneWire;
- stabilitate bună a măsurătorilor.

3.3.4 Motivul alegerii senzorului

Senzorul DS18B20 a fost ales datorită preciziei, stabilității și ușurinței în integrare. Acesta este potrivit pentru aplicații embedded care necesită măsurători fiabile pe termen lung.

3.3.5 Limitări și observații

Temperatura periferică variază lent și este influențată de mediul ambiant, motiv pentru care acest parametru este utilizat complementar, în corelație cu ceilalți senzori.

▪ *3.4 Senzorul piezoelectric pentru monitorizarea respirației*

3.4.1 Rolul senzorului în sistem

Senzorul piezoelectric este utilizat pentru monitorizarea respirației prin detectarea variațiilor mecanice produse de mișcările toracice sau abdominale. Respirația este un parametru puternic influențat de stres, devenind mai rapidă și mai superficială în situații tensionate.

3.4.2 Principiul de funcționare

Elementul piezoelectric generează un semnal electric proporțional cu deformarea mecanică aplicată. Mișcările respiratorii produc variații mecanice detectabile, care sunt convertite în semnale electrice.

3.4.3 Caracteristici tehnice relevante

- ieșire analogică;
- sensibilitate ridicată la vibrații mecanice;
- cost redus;
- implementare simplă.

3.4.4 Motivul alegerii senzorului

Senzorul piezoelectric a fost ales datorită simplității și eficienței sale în monitorizarea respirației, fiind o soluție accesibilă pentru aplicații experimentale.

3.4.5 Limitări și observații

Semnalul piezoelectric poate fi influențat de mișcările corpului, motiv pentru care este necesară filtrarea software și detectarea artefactelor de mișcare.

3.5 Concluzii privind alegerea senzorilor

Utilizarea unui ansamblu de senzori pentru monitorizarea ritmului cardiac, conductanței pielii, respirației și temperaturii corporale permite realizarea unei evaluări multisenzor a reacțiilor fiziologice asociate stresului. Această abordare, inspirată din principiile poligrafului, oferă o estimare mai robustă a stării organismului comparativ cu utilizarea unui singur parametru fiziologic.

4.ARHITECTURA GENERALĂ A SISTEMULUI

Acest capitol descrie arhitectura generală a sistemului multisenzor realizat, evidențiind structura hardware și software, precum și modul de interacțiune dintre componente. Arhitectura a fost concepută astfel încât să permită achiziția, procesarea și afișarea în timp real a parametrilor fiziologici, într-un mod modular și extensibil.

- ***4.1 Prezentare generală a arhitecturii sistemului***

Sistemul realizat este organizat pe o arhitectură de tip **sistem embedded centralizat**, având ca element principal de control microcontrolerul Arduino UNO. Acesta coordonează funcționarea tuturor senzorilor, realizează prelucrarea software a datelor achiziționate și gestionează afișarea informațiilor către utilizator.

Din punct de vedere conceptual, arhitectura sistemului poate fi împărțită în următoarele blocuri funcționale:

- bloc de achiziție a datelor fiziologice;
- bloc de procesare și analiză;
- bloc de afișare și interacțiune cu utilizatorul;
- bloc de alimentare.

Această structurare permite o înțelegere clară a fluxului de date și facilitează extinderea ulterioară a sistemului.

4.2 Arhitectura hardware a sistemului

4.2.1 Unitatea centrală de control

Unitatea centrală de control este reprezentată de placa Arduino UNO, bazată pe microcontrolerul ATmega328P. Aceasta are rolul de a coordona întregul sistem, fiind responsabilă de:

- citirea semnalelor analogice și digitale provenite de la senzori;
- execuția algoritmilor de filtrare, calibrare și corelare;
- comunicarea cu modulele externe;
- afișarea rezultatelor către utilizator.

Arduino UNO a fost ales datorită popularității, ușurinței în programare și disponibilității resurselor hardware necesare aplicației.

4.2.2 Blocul de senzori

Blocul de senzori este compus din:

- senzorul de puls MAX30102;
- senzorul GSR;
- senzorul de temperatură DS18B20;
- senzorul piezoelectric pentru monitorizarea respirației.

Acești senzori sunt conectați la microcontroler prin interfețe analogice și digitale (I2C, OneWire, ADC), în funcție de tipul semnalului furnizat. Fiecare senzor este dedicat măsurării unui parametru fiziologic specific, iar datele colectate sunt ulterior corelate pentru evaluarea nivelului de stres.

4.2.3 Blocul de afișare

Afișarea informațiilor către utilizator este realizată cu ajutorul unui ecran LCD cu interfață I2C. Acesta permite prezentarea în timp real a valorilor relevante, precum ritmul cardiac și scorul de stres, într-o formă ușor de interpretat.

Utilizarea interfeței I2C reduce numărul de pini utilizați pe microcontroler și simplifică schema electrică a sistemului.

4.2.4 Blocul de alimentare

Alimentarea sistemului este realizată prin intermediul interfeței USB a plăcii Arduino UNO, care furnizează tensiunea necesară funcționării microcontrolerului și a senzorilor conectați. Această soluție asigură o alimentare stabilă și ușor de implementat în cadrul unui sistem experimental.

▪ *4.3 Arhitectura software a sistemului*

4.3.1 Structura generală a aplicației software

Aplicația software a fost dezvoltată în mediul Arduino IDE și este structurată modular, fiecare componentă a sistemului având un rol bine definit. Codul este organizat astfel încât să permită:

- inițializarea componentelor hardware;
- achiziția periodică a datelor de la senzori;
- prelucrarea și filtrarea semnalelor;
- calcularea scorului de stres;
- afișarea rezultatelor.

Această abordare contribuie la lizibilitatea codului și facilitează întreținerea și extinderea acestuia.

4.3.2 Achiziția și prelucrarea datelor

Datele provenite de la senzori sunt achiziționate periodic, folosind o abordare non-blocantă, bazată pe funcția millis(). Această metodă permite citirea simultană a mai multor senzori fără a bloca execuția programului.

Semnalele brute sunt supuse unor etape de filtrare și calibrare, pentru a reduce zgomotul și pentru a raporta valorile la un nivel de referință specific fiecărui utilizator.

4.3.3 Corelarea parametrilor și calculul scorului de stres

După prelucrarea individuală a fiecărui semnal fiziologic, datele sunt corelate într-un scor numeric compozit, denumit scor de stres. Acesta reflectă variațiile relative ale parametrilor fiziologici față de valorile de bază stabilite prin calibrare.

Abordarea utilizată este inspirată din principiile poligrafului clasic, dar adaptată scopului experimental al lucrării, fără a formula concluzii absolute.

4.3.4 Interfața cu utilizatorul

Interfața cu utilizatorul este realizată prin afișarea informațiilor pe ecranul LCD. Mesajele afișate permit utilizatorului să urmărească starea sistemului, procesul de calibrare și valoarea scorului de stres, într-un mod intuitiv și accesibil.

4.4 Fluxul de date în cadrul sistemului

Fluxul de date în cadrul sistemului poate fi descris prin următoarele etape:

1. achiziția semnalelor fiziologice de la senzori;
2. filtrarea și calibrarea datelor;
3. analiza individuală a fiecărui parametru;
4. corelarea parametrilor și calculul scorului de stres;
5. afișarea rezultatelor către utilizator.

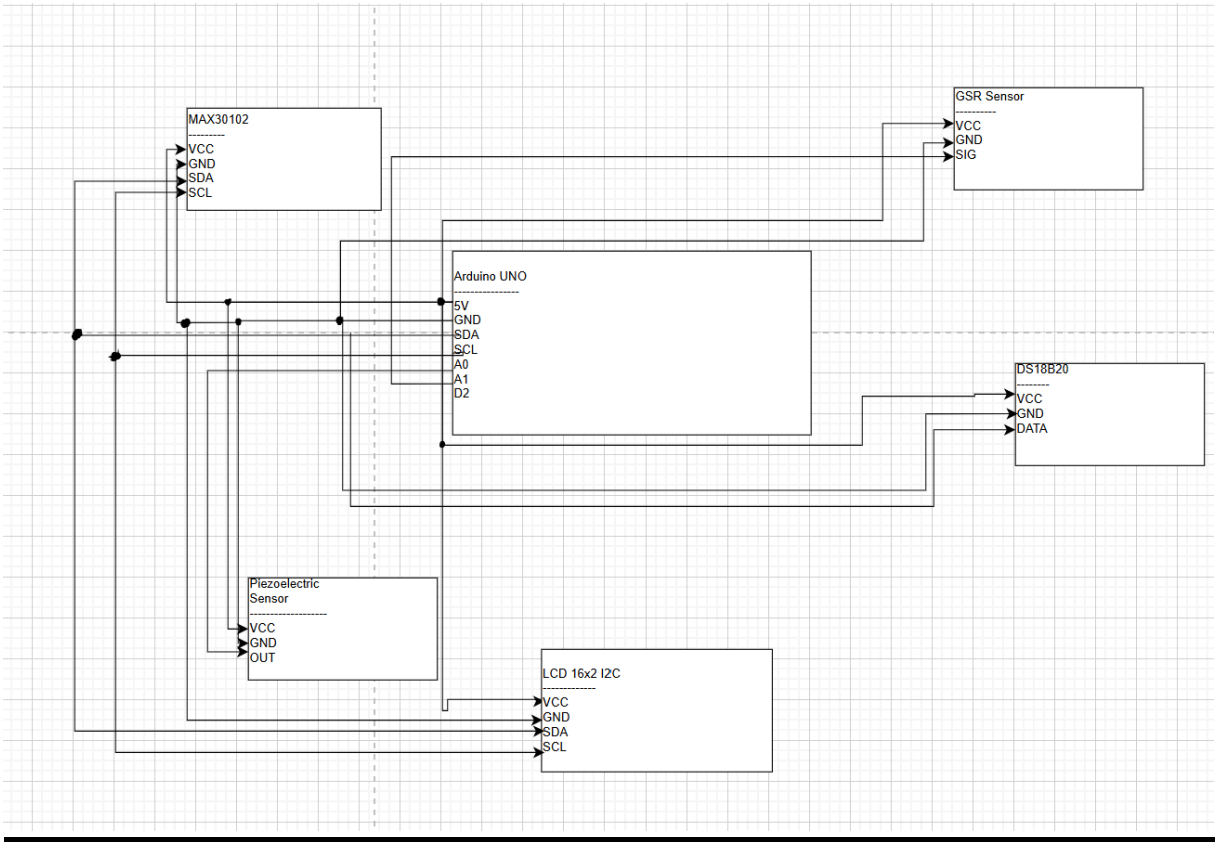
Această succesiune de etape asigură o funcționare coerentă și eficientă a sistemului.

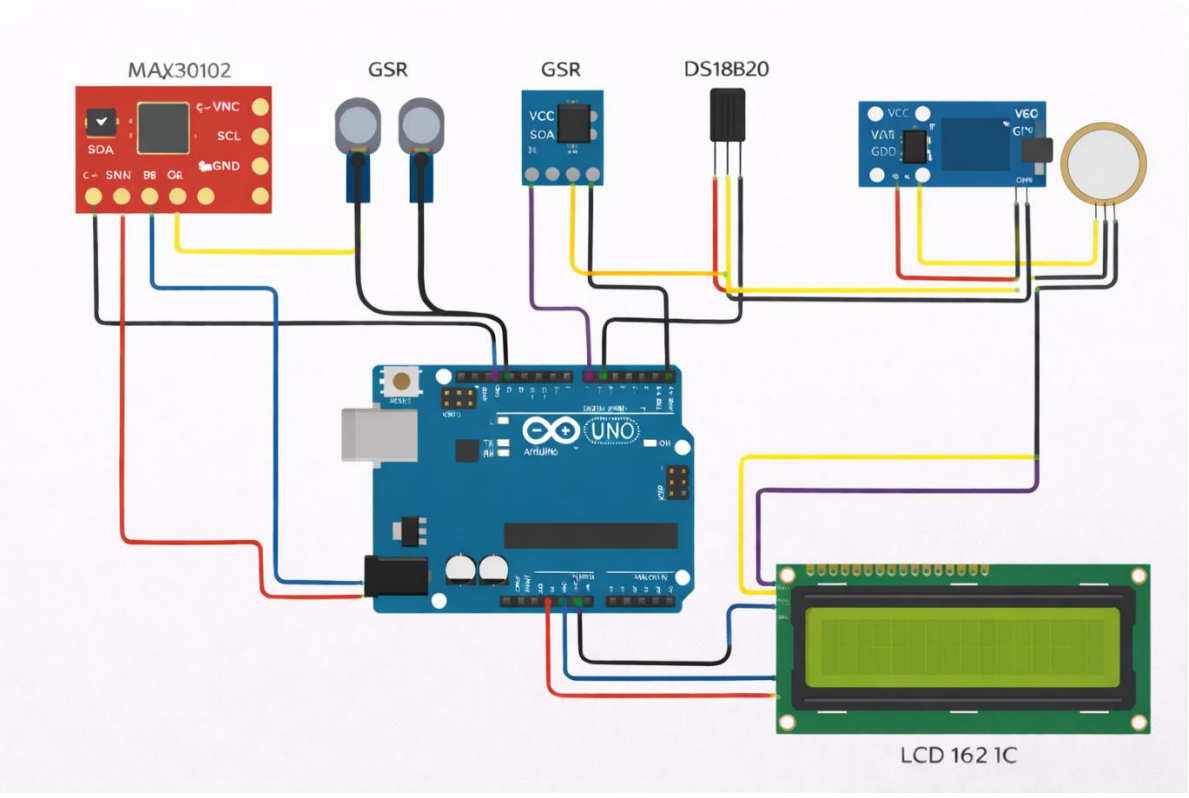
4.5 Concluzii privind arhitectura sistemului

Arhitectura generală a sistemului a fost concepută pentru a îmbina simplitatea implementării cu funcționalitatea necesară monitorizării reacțiilor fiziologice

asociate stresului. Structura modulară permite extinderea ulterioară a sistemului și adaptarea acestuia pentru aplicații mai complexe.

5.SCHEMA ELECTRICĂ ȘI CONECTAREA COMPONENTELOR





Acest capitol prezintă schema electrică a sistemului multisenzor realizat, precum și modul de conectare a componentelor hardware la unitatea centrală de control. Scopul acestui capitol este de a evidenția clar interconectarea senzorilor, tipurile de semnale utilizate și modul în care acestea sunt gestionate de microcontrolerul

Arduino UNO.

▪ *5.1 Considerații generale privind conectarea componentelor*

Schema electrică a sistemului a fost concepută astfel încât să asigure:

- compatibilitatea electrică între senzori și microcontroler;
- stabilitatea semnalelor măsurate;
- simplitatea și claritatea conexiunilor;
- ușurința în realizarea și depanarea montajului.

Toate componentele sunt alimentate la tensiunea de 5 V furnizată de placa Arduino UNO, iar masa comună (GND) este partajată între toate modulele pentru asigurarea unui punct de referință electric stabil.

5.2 Schema electrică a sistemului

Schema electrică a sistemului include următoarele blocuri principale:

- microcontrolerul Arduino UNO;
- blocul de senzori (MAX30102, GSR, DS18B20, piezoelectric);
- modulul de afișare LCD;
- sursa de alimentare.

Fiecare senzor este conectat la Arduino UNO prin interfața specifică tipului de semnal furnizat (analogic sau digital), conform specificațiilor tehnice ale componentelor utilizate.

5.3 Conectarea senzorului de puls – MAX30102

Senzorul MAX30102 este conectat la microcontroler prin intermediul interfeței I2C, utilizând următoarele conexiuni:

- VCC → 5 V;
- GND → GND;
- SDA → pinul SDA al Arduino UNO;
- SCL → pinul SCL al Arduino UNO.

Interfața I2C permite comunicarea bidirecțională între microcontroler și senzor, utilizând un număr redus de pini și asigurând o transmisie stabilă a datelor.

5.4 Conectarea senzorului GSR

Senzorul GSR furnizează un semnal analogic proporțional cu conductanța electrică a pielii. Conectarea acestuia se realizează astfel:

- VCC → 5 V;
- GND → GND;
- SIG → pin analogic A1 al Arduino UNO.

Semnalul analogic este preluat de convertorul analog-digital (ADC) al microcontrolerului și prelucrat software pentru obținerea valorilor relevante.

5.5 Conectarea senzorului de temperatură – DS18B20

Senzorul DS18B20 este conectat la Arduino UNO utilizând protocolul OneWire, după cum urmează:

- VCC → 5 V;
- GND → GND;
- DATA → pin digital D2 al Arduino UNO.

Utilizarea protocolului OneWire permite conectarea mai multor senzori pe același pin de date, însă în cadrul acestui sistem este utilizat un singur senzor de temperatură.

5.6 Conectarea senzorului piezoelectric pentru respirație

Senzorul piezoelectric furnizează un semnal analogic proporțional cu variațiile mecanice produse de mișcările respiratorii. Conectarea acestuia se realizează astfel:

- VCC → 5 V;
- GND → GND;
- OUT → pin analogic A0 al Arduino UNO.

Semnalul analogic este supus ulterior unor etape de filtrare software pentru eliminarea zgomotului și a artefactelor produse de mișcările involuntare ale utilizatorului.

5.7 Conectarea modulului de afișare LCD

Modulul de afișare LCD este conectat la Arduino UNO prin interfața I2C, folosind următoarele conexiuni:

- VCC → 5 V;
- GND → GND;
- SDA → pinul SDA al Arduino UNO;
- SCL → pinul SCL al Arduino UNO.

Această soluție reduce numărul de pini utilizați și simplifică schema electrică a sistemului.

- ***5.8 Tabel de conexiuni ale componentelor***

Pentru o mai bună claritate, conexiunile principale ale componentelor sunt sintetizate în tabelul următor:

MAX30102:	SDA, SCL (I2C), 5 V, GND
GSR:	A1 (analog), 5 V, GND
DS18B20:	D2 (OneWire), 5 V, GND
Piezoelectric:	A0 (analog), 5 V, GND
LCD I2C:	SDA, SCL (I2C), 5 V, GND

- ***5.9 Observații privind realizarea practică a montajului***

Pentru realizarea practică a montajului, componentele au fost conectate pe o placă de test (breadboard), utilizând fire de legătură. Poziționarea senzorilor a fost realizată astfel încât să asigure un contact corespunzător cu utilizatorul și citiri stabile ale semnalelor fiziologice.

De asemenea, au fost respectate bunele practici de cablare, precum utilizarea firelor scurte și evitarea buclelor inutile, pentru reducerea zgomotului electric.

- ***5.10 Concluzii privind schema electrică***

Schema electrică a sistemului este simplă, clară și ușor de reprodus, fiind adaptată scopului experimental al lucrării. Conectarea componentelor asigură compatibilitatea electrică și permite funcționarea stabilă a sistemului multisenzor pentru monitorizarea reacțiilor fiziologice asociate stresului.

6.IMPLEMENTAREA SOFTWARE A SISTEMULUI

Acest capitol descrie implementarea software a sistemului multisenzor pentru monitorizarea reacțiilor fiziologice asociate stresului. Sunt prezentate structura generală a aplicației, modul de achiziție și prelucrare a datelor, algoritmii de

calibrare și filtrare, precum și metoda de calcul a scorului de stres și afișarea rezultatelor către utilizator.

Aplicația software a fost dezvoltată în mediul Arduino IDE și rulează pe microcontrolerul Arduino UNO, fiind concepută pentru funcționare în timp real.

6.1 Structura generală a aplicației software

Programul este organizat modular, fiecare componentă având un rol bine definit în funcționarea sistemului. Structura generală a aplicației include următoarele etape principale:

- inițializarea componentelor hardware;
- calibrarea parametrilor fiziologici;
- achiziția periodică a datelor de la senzori;
- filtrarea și prelucrarea semnalelor;
- corelarea parametrilor și calculul scorului de stres;
- afișarea rezultatelor pe ecranul LCD.

Această organizare permite o bună lizibilitate a codului și facilitează modificarea sau extinderea ulterioară a sistemului.

6.2 Inițializarea sistemului și a componentelor hardware

În etapa de inițializare sunt configurate toate modulele utilizate în sistem. Aceasta include:

- inițializarea comunicației seriale pentru depanare;
- inițializarea ecranului LCD;
- configurarea senzorului de puls MAX30102;
- inițializarea senzorului de temperatură DS18B20;
- configurarea pinilor analogici pentru senzorii GSR și piezoelectric.

Această etapă asigură faptul că toate componentele sunt pregătite pentru funcționare înainte de intrarea în bucla principală a aplicației.

6.3 Achiziția datelor de la senzori

Achiziția datelor se realizează periodic, folosind o abordare **non-blocantă**, bazată pe funcția millis(). Această metodă permite citirea simultană a mai multor senzori fără a întrerupe execuția programului.

Fiecare senzor este citit la un interval de timp specific, adaptat caracteristicilor sale:

- senzorul de puls este citit continuu pentru detectarea bătăilor inimii;
- senzorul GSR este citit la intervale regulate pentru evaluarea conductanței pielii;
- senzorul de temperatură este citit periodic, având variații lente;
- senzorul piezoelectric este citit frecvent pentru detectarea mișcărilor respiratorii.

6.4 Calibrarea individuală a parametrilor fiziologici

Pentru a asigura relevanța rezultatelor, sistemul utilizează o etapă de calibrare individuală. În această etapă sunt determinate valorile de referință (baseline) pentru fiecare utilizator, corespunzătoare unei stări de repaus.

Calibrarea include:

- stabilirea ritmului cardiac de bază;
- determinarea temperaturii corporale periferice de referință;
- calcularea nivelului mediu al semnalului GSR;
- estimarea nivelului de referință pentru respirație.

Valorile obținute în această etapă sunt utilizate ulterior pentru raportarea variațiilor fiziologice asociate stresului.

6.5 Filtrarea și prelucrarea semnalelor

Semnalele achiziționate de la senzori pot conține zgomot sau variații nedorite, cauzate de mișcări involuntare sau interferențe electrice. Din acest motiv, în aplicație sunt implementate mecanisme de filtrare software.

Prelucrarea semnalelor include:

- filtrarea variațiilor rapide și a zgomotului de pe semnalele analogice;
- separarea componentelor lente și rapide ale semnalului GSR;
- eliminarea valorilor nevalide ale ritmului cardiac;
- detectarea și ignorarea artefactelor de mișcare pentru senzorul piezoelectric.

Aceste etape contribuie la creșterea stabilității și fiabilității măsurătorilor.

6.6 Corelarea parametrilor și calculul scorului de stres

După prelucrarea individuală a fiecărui parametru fiziologic, datele sunt corelate într-un **scor numeric compozit**, denumit scor de stres. Acesta este exprimat procentual, într-un interval de la 0 la 100, și reflectă nivelul de activare fiziologică al organismului.

Calculul scorului de stres se bazează pe:

- variația ritmului cardiac față de valoarea de referință;
- modificările conductanței pielii;
- schimbările în tiparul respirației;
- variațiile temperaturii corporale periferice.

Fiecărui parametru i se atribuie o contribuție ponderată, astfel încât scorul final să reflecte influența cumulată a tuturor semnalelor monitorizate. Abordarea este inspirată din principiile poligrafului clasic, dar adaptată scopului experimental al lucrării.

6.7 Afișarea rezultatelor și interfața cu utilizatorul

Rezultatele obținute sunt afișate în timp real pe ecranul LCD. Interfața este concepută pentru a fi simplă și intuitivă, permițând utilizatorului să urmărească:

- starea sistemului;
- procesul de calibrare;
- valorile parametrilor principali;
- scorul de stres calculat.

Mesajele afișate sunt actualizate periodic, fără a afecta funcționarea generală a aplicației.

6.8 Gestionarea erorilor și stabilitatea sistemului

Aplicația software include mecanisme de gestionare a situațiilor neprevăzute, precum:

- lipsa contactului cu senzorul de puls;
- valori nevalide sau instabile ale semnalelor;
- mișcări excesive ale utilizatorului.

În aceste cazuri, sistemul afișează mesaje corespunzătoare și evită calcularea unor rezultate eronate.

6.9 Concluzii privind implementarea software

Implementarea software a sistemului permite funcționarea stabilă și eficientă a aplicației, asigurând achiziția, prelucrarea și afișarea corectă a parametrilor fiziologici. Structura modulară și abordarea non-blocantă contribuie la fiabilitatea sistemului și facilitează extinderea ulterioară a aplicației.

7.TESTARE ȘI REZULTATE EXPERIMENTALE

Acest capitol prezintă metodologia de testare a sistemului multisenzor realizat, scenariile experimentale utilizate, precum și interpretarea rezultatelor obținute. Scopul testării este de a verifica funcționarea corectă a sistemului, stabilitatea măsurătorilor și capacitatea acestuia de a evidenția variații fiziologice asociate stresului.

7.1 Metodologia de testare

Testarea sistemului a fost realizată în condiții controlate, utilizând un singur utilizator, pentru a elimina variațiile interpersonale. Procedura de testare a inclus următoarele etape:

- inițializarea sistemului și verificarea funcționării tuturor componentelor;

- realizarea etapei de calibrare individuală;
- monitorizarea parametrilor fiziologici în diferite stări;
- observarea și înregistrarea variațiilor scorului de stres.

Testele au fost realizate într-un mediu interior, la temperatură constantă, pentru a reduce influența factorilor externi asupra măsurătorilor.

7.2 Scenarii de testare

Pentru evaluarea comportamentului sistemului, au fost definite mai multe scenarii de testare, corespunzătoare unor stări fiziologice diferite.

7.2.1 Scenariul de repaus (stare neutră)

În acest scenariu, utilizatorul a fost într-o stare de repaus, fără stimuli stresori. A fost realizată calibrarea inițială a sistemului, iar valorile fiziologice obținute au fost utilizate ca referință (baseline).

Rezultatele au indicat:

- ritm cardiac stabil;
- valori GSR constante;
- respirație regulată;
- temperatură periferică relativ constantă;
- scor de stres scăzut.

Acest scenariu a permis verificarea stabilității sistemului în condiții normale.

7.2.2 Scenariul de stres cognitiv

În acest scenariu, utilizatorul a fost supus unui stimul de tip stres cognitiv, constând în rezolvarea rapidă a unor probleme aritmetice sub presiunea timpului.

În urma aplicării acestui stimul, au fost observate următoarele modificări:

- creșterea ritmului cardiac față de valoarea de referință;
- creșterea conductanței pielii;
- modificarea tiparului respirator;

- creșterea scorului de stres afișat pe ecranul LCD.

Aceste rezultate confirmă capacitatea sistemului de a detecta variații fiziologice asociate stresului.

7.2.3 Scenariul de stres emoțional

Scenariul de stres emoțional a constat în expunerea utilizatorului la stimuli auditivi sau vizuali cu impact emoțional. În acest caz, reacțiile fiziologice au fost mai subtile, dar totuși detectabile.

Sistemul a evidențiat:

- variații rapide ale semnalului GSR;
- fluctuații ale ritmului cardiac;
- modificări ușoare ale respirației;
- o creștere moderată a scorului de stres.

▪ *7.3 Rezultate experimentale obținute*

Rezultatele obținute în urma testării au arătat că sistemul funcționează conform așteptărilor, fiind capabil să monitorizeze și să coreleze mai mulți parametri fiziologici.

Observațiile principale includ:

- scorul de stres a crescut în mod consistent în scenariile de stres;
- parametrii fiziologici au revenit gradual la valorile de referință după încetarea stimulilor;
- sistemul a reacționat în timp real la modificările fiziologice.

Aceste rezultate demonstrează funcționalitatea sistemului și relevanța abordării multisenzor.

▪ *7.4 Analiza și interpretarea rezultatelor*

Analiza rezultatelor indică faptul că utilizarea mai multor senzori contribuie la o evaluare mai robustă a reacțiilor fiziologice asociate stresului. Corelarea

parametrilor permite reducerea influenței variațiilor izolate și oferă o imagine de ansamblu asupra stării organismului.

De asemenea, etapa de calibrare individuală s-a dovedit esențială pentru interpretarea corectă a rezultatelor, întrucât valorile absolute ale parametrilor diferă semnificativ de la un utilizator la altul.

▪ *7.5 Limitări observate în timpul testării*

În timpul testării au fost identificate următoarele limitări:

- influența mișcărilor involuntare asupra semnalului piezoelectric;
- sensibilitatea semnalului GSR la condițiile de mediu;
- variații individuale semnificative ale parametrilor fiziologici;
- imposibilitatea corelării directe a scorului de stres cu veridicitatea răspunsurilor.

Aceste limitări subliniază caracterul experimental al sistemului și necesitatea interpretării orientative a rezultatelor.

▪ *7.6 Concluzii privind testarea sistemului*

Testarea sistemului multisenzor a demonstrat că acesta este capabil să monitorizeze și să evidențieze variații fiziologice asociate stresului, în timp real. Rezultatele obținute confirmă corectitudinea implementării hardware și software, precum și utilitatea abordării multisenzor inspirate din principiile poligrafului.

8.LIMITĂRI ȘI PROBLEME NEREZOLVATE

Acest capitol prezintă principalele limitări identificate în cadrul sistemului multisenzor realizat, precum și problemele care nu au putut fi rezolvate în totalitate în cadrul acestui proiect. Analiza acestor aspecte este esențială pentru înțelegerea corectă a rezultatelor obținute și pentru evidențierea caracterului experimental al lucrării.

8.1 Limitări legate de variațiile individuale

Una dintre principalele limitări ale sistemului este reprezentată de variațiile fiziologice semnificative dintre utilizatori. Parametri precum ritmul cardiac, conductanța pielii sau tiparul respirator diferă considerabil în funcție de vârstă, condiția fizică, nivelul de stres cronic și starea emoțională generală.

Deși sistemul utilizează o etapă de calibrare individuală, aceasta nu poate elimina complet diferențele interpersonale, motiv pentru care rezultatele obținute sunt relevante doar în raport cu utilizatorul testat și nu pot fi generalizate.

8.2 Limitări ale senzorilor utilizați

Fiecare senzor integrat în sistem prezintă limitări specifice:

- **Senzorul de puls** poate fi influențat de mișcările utilizatorului sau de poziționarea incorectă a degetului.
- **Senzorul GSR** este sensibil la condițiile de mediu, precum temperatura sau umiditatea, și la poziționarea electrozilor.
- **Senzorul piezoelectric** poate detecta și alte mișcări ale corpului, nu doar respirația, ceea ce poate introduce artefacte în semnal.
- **Senzorul de temperatură** reacționează lent la modificări fiziologice, fiind influențat de mediul ambiant.

Aceste limitări impun utilizarea filtrării software și interpretarea orientativă a rezultatelor.

8.3 Limitări ale algoritmilor de procesare

Algoritmii de procesare utilizați pentru filtrarea și corelarea semnalelor sunt concepuți pentru a oferi un compromis între complexitate și performanță. Din acest motiv, aceștia nu pot elimina complet zgomotul de semnal sau artefactele produse de mișcările involuntare.

De asemenea, ponderile utilizate în calculul scorului de stres sunt stabilite empiric și nu se bazează pe un model clinic validat, ceea ce limitează acuratețea absolută a scorului obținut.

8.4 Imposibilitatea detectării directe a minciunii

O limitare fundamentală a sistemului este faptul că acesta **nu poate detecta direct minciuna**. Sistemul monitorizează exclusiv reacțiile fiziologice asociate stresului, iar aceste reacții pot apărea în numeroase contexte care nu implică comportamente neadevărate.

Astfel, scorul de stres obținut nu trebuie interpretat ca un indicator al veridicității răspunsurilor, ci doar ca o estimare a nivelului de activare fiziologică al utilizatorului.

8.5 Probleme nerezolvate și direcții de îmbunătățire

În cadrul acestui proiect nu au fost abordate următoarele aspecte:

- validarea clinică a sistemului pe un număr mare de utilizatori;
- adaptarea automată a ponderilor scorului de stres în funcție de utilizator;
- integrarea unor algoritmi avansați de analiză a semnalelor;
- utilizarea unor senzori suplimentari pentru creșterea acurateței.

Aceste probleme reprezintă posibile direcții de dezvoltare pentru lucrări viitoare.

8.6 Concluzii privind limitările sistemului

Analiza limitărilor și a problemelor nerezolvate evidențiază caracterul experimental al sistemului realizat. În ciuda acestor limitări, proiectul își atinge obiectivele propuse, oferind o platformă funcțională pentru monitorizarea reacțiilor fiziologice asociate stresului și pentru studierea principiilor de bază ale sistemelor de tip poligraf.

9.CONCLUZII ȘI ASPECTE ECONOMICE

Acest capitol sintetizează rezultatele obținute în urma realizării sistemului multisenzor pentru monitorizarea reacțiilor fiziologice asociate stresului și evidențiază principalele concluzii ale lucrării. De asemenea, sunt prezentate

aspectele economice ale soluției propuse, subliniind caracterul accesibil și aplicabil al sistemului.

9.1 Concluzii generale

În cadrul acestei lucrări a fost realizat un **sistem multisenzor pentru monitorizarea reacțiilor fiziologice asociate stresului, inspirat din principiile poligrafului**, bazat pe platforma Arduino UNO. Sistemul integrează mai mulți senzori biomedicali pentru măsurarea ritmului cardiac, conductanței pielii, respirației și temperaturii corporale periferice, oferind o abordare multisenzor pentru evaluarea nivelului de stres.

Rezultatele obținute în urma testării demonstrează că sistemul este capabil să monitorizeze în timp real variațiile fiziologice asociate stresului și să coreleze aceste informații într-un scor numeric orientativ. Implementarea etapei de calibrare individuală s-a dovedit esențială pentru obținerea unor rezultate relevante și stabile.

Lucrarea evidențiază faptul că utilizarea mai multor parametri fiziologici permite o evaluare mai robustă a reacțiilor organismului comparativ cu analiza unui singur semnal biologic. Abordarea adoptată confirmă utilitatea principiilor utilizate în sistemele de tip poligraf, adaptate însă unui scop experimental și educațional.

9.2 Îndeplinirea obiectivelor propuse

Obiectivele stabilite în cadrul lucrării au fost atinse în mod corespunzător:

- a fost realizată integrarea hardware a senzorilor într-un sistem funcțional;
- a fost implementată o arhitectură software modulară și stabilă;
- a fost dezvoltat un mecanism de calibrare individuală;
- a fost calculat un scor de stres bazat pe corelarea parametrilor fiziologici;
- rezultatele au fost afișate într-un mod intuitiv pentru utilizator.

Prin atingerea acestor obiective, lucrarea își îndeplinește scopul declarat și demonstrează aplicabilitatea practică a soluției propuse.

- ***9.3 Aspecte economice ale sistemului***

Un avantaj important al sistemului realizat îl reprezintă **costul redus al implementării**, comparativ cu echipamentele comerciale sau profesionale de tip poligraf. Utilizarea platformei Arduino UNO și a senzorilor accesibili permite realizarea unei soluții funcționale cu resurse financiare limitate.

Costurile principale sunt asociate componentelor hardware utilizate:

- placa Arduino UNO;
- senzorul de puls MAX30102;
- senzorul GSR;
- senzorul de temperatură DS18B20;
- senzorul piezoelectric;
- modulul de afișare LCD;
- elemente auxiliare (fire, breadboard).

Costul total al sistemului este redus(~400lei), ceea ce îl face accesibil pentru proiecte educaționale, demonstrații didactice sau prototipuri experimentale.

9.4 Avantajele economice și practice ale soluției propuse

Din punct de vedere economic și practic, soluția propusă prezintă următoarele avantaje:

- **cost scăzut** de realizare;
- **ușurință în implementare** și întreținere;
- **flexibilitate** în extinderea sistemului;
- **accesibilitate** pentru utilizatori fără experiență avansată;
- **potențial educațional ridicat**.

Aceste caracteristici fac ca sistemul să fie potrivit pentru utilizarea în medii academice, laboratoare didactice sau proiecte de cercetare la nivel introductiv.

9.5 Perspective de dezvoltare ulterioară

Deși sistemul realizat își atinge obiectivele propuse, acesta poate fi extins și îmbunătățit prin:

- integrarea unor algoritmi avansați de procesare a semnalelor;
- adaptarea automată a ponderilor scorului de stres;
- utilizarea unor senzori suplimentari;
- dezvoltarea unei interfețe grafice mai avansate;
- portarea sistemului pe o platformă portabilă.

Aceste direcții pot constitui baza unor lucrări viitoare sau a unor proiecte de cercetare mai complexe.

9.6 Concluzie finală

În concluzie, lucrarea demonstrează că este posibilă realizarea unui sistem multisenzor funcțional pentru monitorizarea reacțiilor fiziologice asociate stresului, folosind o platformă embedded accesibilă și componente low-cost. Proiectul evidențiază atât importanța abordării multisenzor, cât și necesitatea unei interpretări responsabile a datelor fiziologice, subliniind caracterul experimental al soluției propuse.

10.BIBLIOGRAFIE

1. • Arduino,
   Arduino UNO Rev3 – Technical Specifications,
   disponibil online la:
   https://www.arduino.cc/en/Main/ArduinoBoardUno

2. • Maxim Integrated,
   MAX30102 High-Sensitivity Pulse Oximeter and Heart-Rate Sensor Datasheet,
   disponibil online la:
   https://www.analog.com/media/en/technical-documentation/data-sheets/MAX30102.pdf

3. • Dallas Semiconductor,
   DS18B20 Programmable Resolution 1-Wire Digital Thermometer Datasheet,
   disponibil online la:
   https://www.analog.com/media/en/technical-documentation/data-sheets/ds18b20.pdf

4. • OpenAI,
   ChatGPT – instrument de asistență pentru redactarea, structurarea și clarificarea conținutului documentației tehnice,
   https://chat.openai.com

11.ANEXE

- ***Anexa A – Codul sursă al aplicației Arduino***

În această anexă este prezentat codul sursă complet utilizat pentru implementarea sistemului multisenzor de monitorizare a reacțiilor fiziologice asociate stresului. Codul a fost dezvoltat pe platforma Arduino și include funcționalități pentru citirea, filtrarea și procesarea semnalelor provenite de la senzorii utilizați, precum și pentru afișarea rezultatelor pe modulul LCD.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include "MAX30105.h"
#include "heartRate.h"
#include <OneWire.h>
#include <DallasTemperature.h>

// ================== LCD ==================
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

```
// ================== MAX30102 ==================
MAX30105 particleSensor;

// ================== DS18B20 ==================
#define ONE_WIRE_BUS 2
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

// ================== PIEZO ==================
// Piezo module: +5V, GND, S -> A0
#define PIEZO_PIN A0

// ================== GSR ==================
// GSR module: VCC->5V, GND->GND, SIG->A1
#define GSR_PIN A1

// GSR sampling
const int GSR_SAMPLE_MS = 120;
unsigned long lastGsrSample = 0;

// GSR calibration (after baselineDone)
bool gsrCalibrating = false;
bool gsrCalDone = false;
unsigned long gsrCalStart = 0;
long gsrCalAcc = 0;
int gsrCalCnt = 0;

// GSR filters
float gsrBase = 0;   // very slow baseline
float gsrFast = 0;   // faster tracking
float gsrScoreSmooth = 0;
int gsrLastRaw = 0;
int gsrLastPhasic = 0;
int gsrLastScore = 0;

// GSR parameters (tunable)
const unsigned long GSR_CAL_MS = 60000; // 60 sec
const int GSR_PHASIC_CLAMP = 40;        // map 0..40 -> 0..100
// ================== PIEZO sampling ==================
const int PIEZO_SAMPLE_MS = 10;    // 100 Hz
unsigned long lastPiezoSample = 0;

// piezo processing
int piezoBaselineRaw = 0;
int piezoEnvelope = 0;

// piezo baseline calibration
long piezoBaseSum = 0;
int  piezoBaseN = 0;
int  piezoBaseEnv = 1;             // baseline envelope (can be 1-3 for calm breathers)
int  piezoSpikeTH = 80;

// piezo window stats (5 sec)
const int PIEZO_WIN_MS = 5000;
unsigned long piezoWinStart = 0;
int piezoCurMax = 0;
int piezoActiveSamples = 0;
int piezoSpikeCount = 0;

// latest piezo classification
enum PiezoState { PZ_NORMAL = 0, PZ_MED = 1, PZ_HIGH = 2, PZ_MOVE = 3 };
PiezoState piezoState = PZ_NORMAL;
```

```
// debug values (for adding score / printing)
int piezoLastRatioMax = 0;
int piezoLastActiveMs = 0;
int piezoLastCurMax = 0;

// ================== BPM ==================
long lastBeat = 0;
float BPM = 0;
float bpmBuffer[5] = {0};
int bpmIndex = 0;
bool bpmFilled = false;

// ================== TEMP ==================
unsigned long lastTempMillis = 0;
float tempC = 0;

// ================== BASELINE ==================
float bpmBaseline = 0;
float tempBaseline = 0;
int baselineCount = 0;
const int baselineSamples = 30;
bool baselineDone = false;

// ================== LCD timing ==================
unsigned long lastLCD = 0;
const unsigned long lcdInterval = 400;

// ================== SCOR ==================
int lieScore = 0;

float getBpmAverage() {
  float sum = 0;
  int count = bpmFilled ? 5 : bpmIndex;
  if (count == 0) return BPM;
  for (int i = 0; i < count; i++) sum += bpmBuffer[i];
  return sum / count;
}

// -------- PIEZO HELPERS --------
void piezoInitBaselineRaw() {
  long s = 0;
  for (int i = 0; i < 300; i++) {
    s += analogRead(PIEZO_PIN);
    delay(5);
  }
  piezoBaselineRaw = (int)(s / 300);
  piezoWinStart = millis();
}

void piezoUpdateOneSample() {
  int v = analogRead(PIEZO_PIN);

  // diff from slow baseline
  int diff = v - piezoBaselineRaw;
  if (diff < 0) diff = -diff;

  // very slow baseline tracking (so it doesn't "eat" breathing)
  piezoBaselineRaw = (piezoBaselineRaw * 9995 + v * 5) / 10000;

  // envelope: rise faster, decay slower
  if (diff > piezoEnvelope) piezoEnvelope = (piezoEnvelope * 7 + diff * 3) / 10;
```

```
  else                  piezoEnvelope = (piezoEnvelope * 97 + diff * 3) / 100;

  // spike detection (movement/noise)
  static int prevEnv = 0;
  if (piezoEnvelope - prevEnv > piezoSpikeTH) piezoSpikeCount++;
  prevEnv = piezoEnvelope;

  // window stats
  if (piezoEnvelope > piezoCurMax) piezoCurMax = piezoEnvelope;

  // "active" if above 2x baseline (works even when baseline is small)
  if (piezoEnvelope > piezoBaseEnv * 2) piezoActiveSamples++;
}

void piezoFinalizeWindowIfNeeded() {
  unsigned long now = millis();
  if (now - piezoWinStart < PIEZO_WIN_MS) return;

  // compute metrics
  int ratioMax = (int)((long)piezoCurMax * 100 / piezoBaseEnv);
  int activeMs = piezoActiveSamples * PIEZO_SAMPLE_MS;

  piezoLastRatioMax = ratioMax;
  piezoLastActiveMs = activeMs;
  piezoLastCurMax = piezoCurMax;

  // thresholds (tune these if needed)
  const int ABS_MED  = 15;
  const int ABS_HIGH = 25;

  const int R_MED  = 200;   // 2.0x
  const int R_HIGH = 300;   // 3.0x

  const int ACTIVE_MED_MS  = 800;
  const int ACTIVE_HIGH_MS = 1500;

  const int MOVE_SPIKES = 8;

  if (piezoSpikeCount >= MOVE_SPIKES) {
    piezoState = PZ_MOVE;
  } else if ((piezoCurMax >= ABS_HIGH && ratioMax >= R_HIGH) ||
             (activeMs >= ACTIVE_HIGH_MS && ratioMax >= R_MED)) {
    piezoState = PZ_HIGH;
  } else if ((piezoCurMax >= ABS_MED && ratioMax >= R_MED) ||
             (activeMs >= ACTIVE_MED_MS)) {
    piezoState = PZ_MED;
  } else {
    piezoState = PZ_NORMAL;
  }

  // reset window
  piezoCurMax = 0;
  piezoActiveSamples = 0;
  piezoSpikeCount = 0;
  piezoWinStart = now;
}

int piezoScoreContribution() {
  // If movement/noise, ignore (score 0)
  if (piezoState == PZ_MOVE) return 0;

  int score = 0;
```

```
  // base contribution by state
  if (piezoState == PZ_MED) score += 15;
  else if (piezoState == PZ_HIGH) score += 30;

  // extra small bonus by intensity (kept safe)
  int extra = piezoLastCurMax;          // typical: 0..80+ in your setup
  if (extra > 60) extra = 60;
  extra = extra / 3;                    // 0..20
  score += extra;

  if (score > 40) score = 40;           // piezo max contribution
  return score;
}

const char* piezoStateText() {
  switch (piezoState) {
    case PZ_NORMAL: return "N";
    case PZ_MED:    return "M";
    case PZ_HIGH:   return "H";
    case PZ_MOVE:   return "X"; // movement/noise
  }
  return "?";
}

// -------- GSR HELPERS --------
int gsrReadAvg() {
  long sum = 0;
  for (int i = 0; i < 15; i++) {
    sum += analogRead(GSR_PIN);
    delay(3);
  }
  return (int)(sum / 15);
}

// returns 0..30
int gsrScoreContribution() {
  if (!gsrCalDone) return 0;

  // map phasic 0..GSR_PHASIC_CLAMP -> 0..100
  int p = gsrLastPhasic;
  if (p < 0) p = 0;
  if (p > GSR_PHASIC_CLAMP) p = GSR_PHASIC_CLAMP;
  int pct = (p * 100) / GSR_PHASIC_CLAMP;

  // smooth score already 0..100
  int s = gsrLastScore;

  // convert to 0..30 contribution
  int contrib = (s * 30) / 100;
  if (contrib < 0) contrib = 0;
  if (contrib > 30) contrib = 30;
  return contrib;
}

void gsrStartCalibration() {
  gsrCalibrating = true;
  gsrCalDone = false;

  gsrCalStart = millis();
  gsrCalAcc = 0;
  gsrCalCnt = 0;
```

```
  // init filters with current reading
  int r = gsrReadAvg();
  gsrBase = r;
  gsrFast = r;
  gsrScoreSmooth = 0;
  gsrLastRaw = r;
  gsrLastPhasic = 0;
  gsrLastScore = 0;
}

void gsrUpdate() {
  unsigned long now = millis();
  if (now - lastGsrSample < GSR_SAMPLE_MS) return;
  lastGsrSample = now;

  int raw = gsrReadAvg();
  gsrLastRaw = raw;

  // If calibration hasn't started, do nothing
  if (!gsrCalibrating && !gsrCalDone) return;

  // During calibration: accumulate raw mean for 60 sec
  if (gsrCalibrating) {
    gsrCalAcc += raw;
    gsrCalCnt++;

    if (now - gsrCalStart >= GSR_CAL_MS) {
      float b = (gsrCalCnt > 0) ? ((float)gsrCalAcc / gsrCalCnt) : raw;
      gsrBase = b;
      gsrFast = b;          // IMPORTANT: align fast/base after calibration
      gsrCalibrating = false;
      gsrCalDone = true;
      gsrScoreSmooth = 0;
    }
    return;
  }

  // After calibration:
  // slow baseline follows drift very slowly
  gsrBase = 0.998f * gsrBase + 0.002f * raw;
  // fast follows quicker
  gsrFast = 0.85f * gsrFast + 0.15f * raw;

  int phasic = (int)gsrFast - (int)gsrBase;
  gsrLastPhasic = phasic;

  int p = phasic;
  if (p < 0) p = 0;
  if (p > GSR_PHASIC_CLAMP) p = GSR_PHASIC_CLAMP;

  float target = (p * 100.0f) / (float)GSR_PHASIC_CLAMP;

  // smooth score (realistic)
  gsrScoreSmooth = 0.95f * gsrScoreSmooth + 0.05f * target;
  gsrLastScore = (int)gsrScoreSmooth;
  if (gsrLastScore < 0) gsrLastScore = 0;
  if (gsrLastScore > 100) gsrLastScore = 100;
}

void setup() {
  Serial.begin(115200);
```

```
  while (!Serial);

  lcd.init();
  lcd.backlight();
  lcd.print("Detector minciuni");
  delay(1500);
  lcd.clear();

  if (!particleSensor.begin(Wire)) {
    lcd.print("MAX30102 ERR");
    while (1);
  }

  particleSensor.setup();
  particleSensor.setPulseAmplitudeIR(0x1F);
  particleSensor.setPulseAmplitudeRed(0x1F);
  particleSensor.setPulseAmplitudeGreen(0);

  sensors.begin();
  sensors.setResolution(9);
  sensors.setWaitForConversion(false);

  // piezo init
  piezoInitBaselineRaw();

  lcd.print("PUNE DEGETUL");
}

void loop() {
  // ===== BPM =====
  long irValue = particleSensor.getIR();
  bool fingerDetected = irValue > 20000;

  if (fingerDetected && checkForBeat(irValue)) {
    long delta = millis() - lastBeat;
    lastBeat = millis();

    if (delta > 0) {
      float newBPM = 60000.0 / delta;
      if (newBPM >= 45 && newBPM <= 120) {
        BPM = newBPM;
        bpmBuffer[bpmIndex++] = BPM;
        if (bpmIndex >= 5) {
          bpmIndex = 0;
          bpmFilled = true;
        }
      }
    }
  }

  float bpmAvg = getBpmAverage();

  // ===== TEMP =====
  if (millis() - lastTempMillis >= 1000) {
    sensors.requestTemperatures();
    lastTempMillis = millis();
  }
  tempC = sensors.getTempCByIndex(0);

  // ===== PIEZO sampling (non-blocking) =====
  if (millis() - lastPiezoSample >= PIEZO_SAMPLE_MS) {
    lastPiezoSample = millis();
```

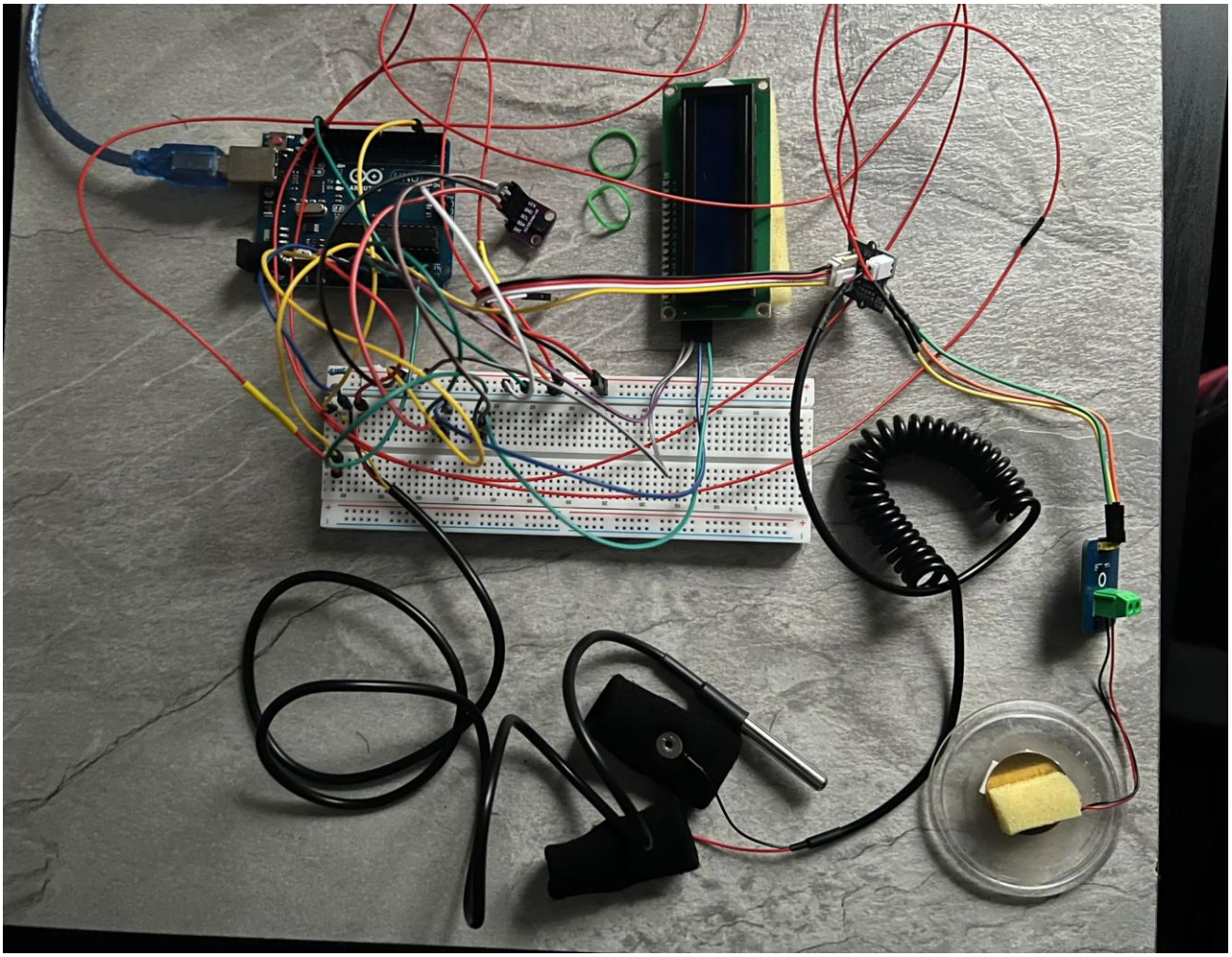
```
  piezoUpdateOneSample();
  piezoFinalizeWindowIfNeeded();
}

// ===== BASELINE (BPM/TEMP + PIEZO baseline) =====
// We only build baseline when finger is detected and bpmAvg valid (so user is "ready")
if (!baselineDone && fingerDetected && bpmAvg > 0) {
  bpmBaseline += bpmAvg;
  tempBaseline += tempC;
  baselineCount++;

  // piezo baseline during the same period
  piezoBaseSum += piezoEnvelope;
  piezoBaseN++;

  if (baselineCount >= baselineSamples) {
    bpmBaseline /= baselineSamples;
    tempBaseline /= baselineSamples;
    baselineDone = true;

    // finalize piezo baseline
    piezoBaseEnv = (piezoBaseN > 0) ? (int)(piezoBaseSum / piezoBaseN) : 1;
    if (piezoBaseEnv < 1) piezoBaseEnv = 1;

    // set adaptive spike threshold
    piezoSpikeTH = piezoBaseEnv * 10;
    if (piezoSpikeTH < 20) piezoSpikeTH = 20;

    // reset window after baseline
    piezoCurMax = 0;
    piezoActiveSamples = 0;
    piezoSpikeCount = 0;
    piezoWinStart = millis();

    // start GSR calibration NOW (user already has finger on MAX + electrodes on)
    gsrStartCalibration();

    // debug
    Serial.print("BASELINES DONE. bpmBase=");
    Serial.print(bpmBaseline, 1);
    Serial.print(" tempBase=");
    Serial.print(tempBaseline, 2);
    Serial.print(" piezoBaseEnv=");
    Serial.print(piezoBaseEnv);
    Serial.print(" piezoSpikeTH=");
    Serial.print(piezoSpikeTH);
    Serial.println();
    Serial.println("GSR calibration started (60 sec)...");
  }
}

// ===== GSR update (non-blocking) =====
gsrUpdate();

// ===== SCOR "minciuna" (de fapt: stres) =====
if (baselineDone) {
  lieScore = 0;

  // BPM contribution
  if (bpmAvg > bpmBaseline)
    lieScore += (int)((bpmAvg - bpmBaseline) * 2);
```

```
  // TEMP contribution (temp drop -> stress)
  if (tempC < tempBaseline)
    lieScore += (int)((tempBaseline - tempC) * 20);

  // PIEZO contribution
  lieScore += piezoScoreContribution();

  // GSR contribution (max 30)
  lieScore += gsrScoreContribution();

  lieScore = constrain(lieScore, 0, 100);

  // Optional debug to Serial
  static unsigned long lastDbg = 0;
  if (millis() - lastDbg > 1000) {
    lastDbg = millis();
    Serial.print("bpm=");
    Serial.print((int)bpmAvg);
    Serial.print(" base=");
    Serial.print((int)bpmBaseline);

    Serial.print(" temp=");
    Serial.print(tempC, 2);
    Serial.print(" base=");
    Serial.print(tempBaseline, 2);

    Serial.print(" piezoEnv=");
    Serial.print(piezoEnvelope);
    Serial.print(" piezoBase=");
    Serial.print(piezoBaseEnv);
    Serial.print(" pzState=");
    Serial.print(piezoStateText());

    Serial.print(" GSRraw=");
    Serial.print(gsrLastRaw);
    Serial.print(" GSRbase=");
    Serial.print((int)gsrBase);
    Serial.print(" GSRfast=");
    Serial.print((int)gsrFast);
    Serial.print(" GSRph=");
    Serial.print(gsrLastPhasic);
    Serial.print(" GSRs=");
    Serial.print(gsrLastScore);
    Serial.print(" gsrContrib=");
    Serial.print(gsrScoreContribution());

    Serial.print(" score=");
    Serial.println(lieScore);
  }
}

// ===== LCD =====
if (millis() - lastLCD >= lcdInterval) {
  lastLCD = millis();

  lcd.setCursor(0, 0);
  lcd.print("               ");
  lcd.setCursor(0, 1);
  lcd.print("               ");

  if (!fingerDetected) {
```

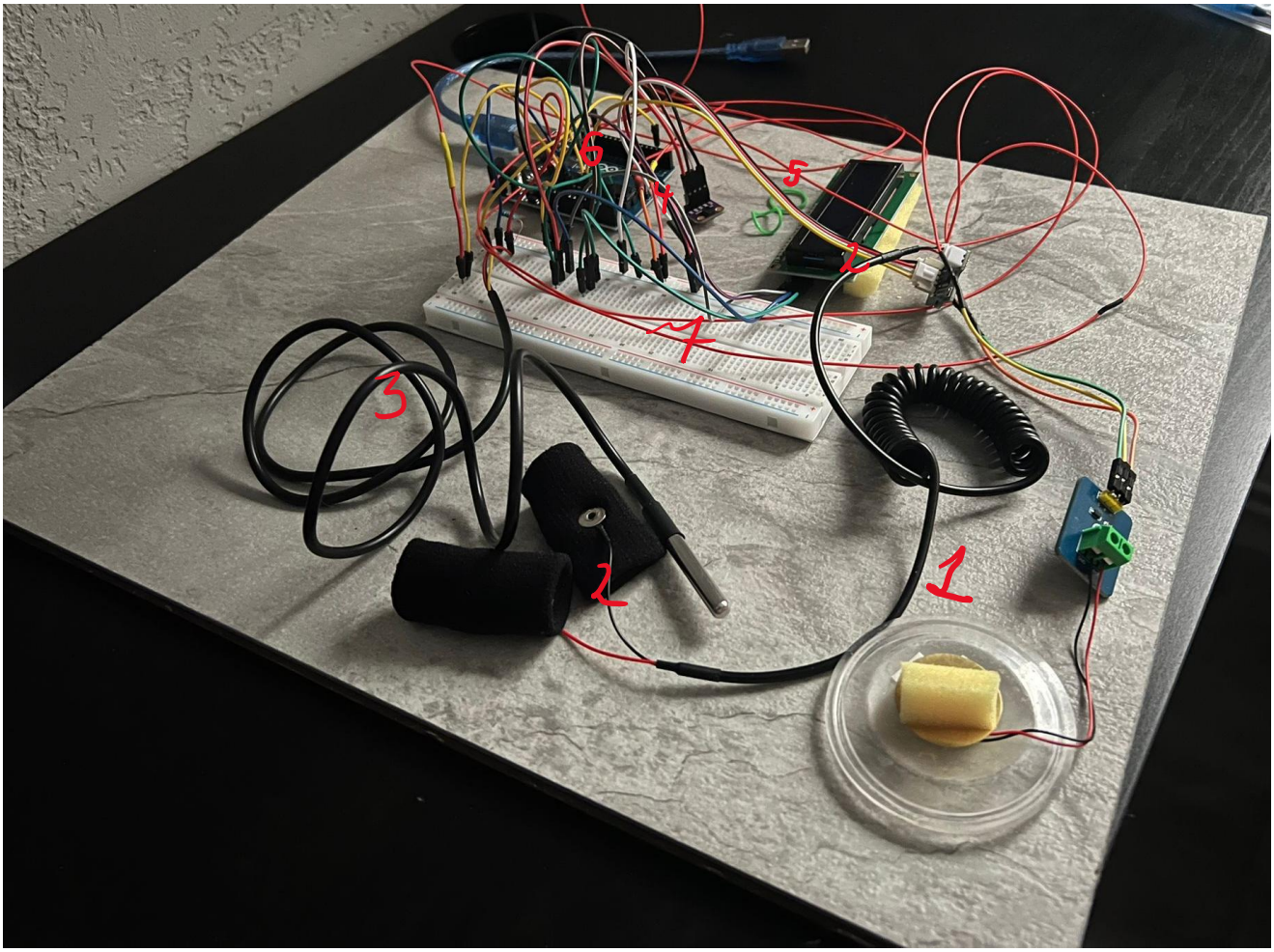
```
    lcd.setCursor(0, 0);
    lcd.print("PUNE DEGETUL");
    lcd.setCursor(0, 1);
    lcd.print("pe senzor");
   }
   else if (!baselineDone) {
    lcd.setCursor(0, 0);
    lcd.print("CALIBRARE");
    lcd.setCursor(0, 1);
    lcd.print(baselineCount);
    lcd.print("/");
    lcd.print(baselineSamples);
   }
   else if (!gsrCalDone) {
    // show GSR calibration progress
    lcd.setCursor(0, 0);
    lcd.print("GSR CALIBRARE");
    lcd.setCursor(0, 1);
    unsigned long passed = millis() - gsrCalStart;
    int pct = (passed >= GSR_CAL_MS) ? 100 : (int)(passed * 100 / GSR_CAL_MS);
    lcd.print(pct);
    lcd.print("%");
   }
   else {
    lcd.setCursor(0, 0);
    lcd.print("BPM:");
    lcd.print((int)bpmAvg);
    lcd.print(" PZ:");
    lcd.print(piezoStateText()); // N/M/H/X

    lcd.setCursor(0, 1);
    lcd.print("SCOR:");
    lcd.print(lieScore);
    lcd.print("%");
   }
  }
}
```

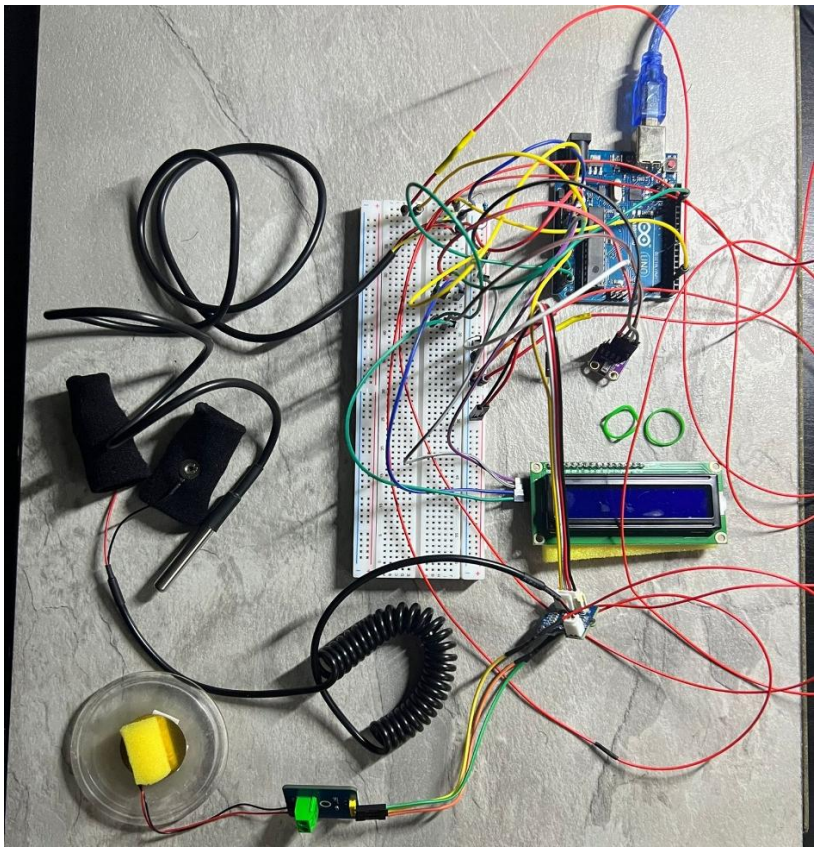
- ***Anexa B – Fotografii ale montajului experimental***

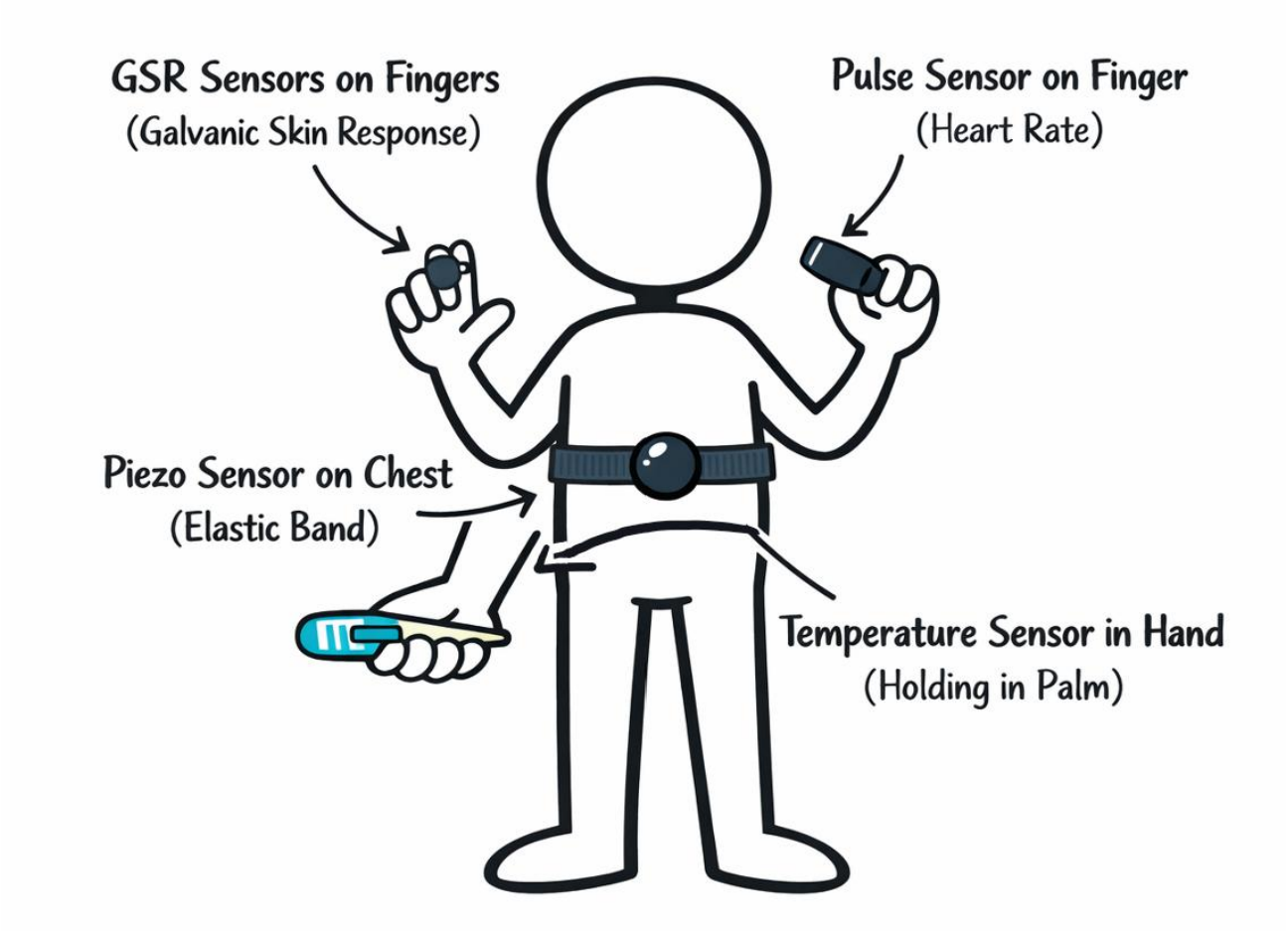
Această anexă conține imagini reprezentative ale montajului experimental realizat pe baza plăcii Arduino UNO. Fotografiile evidențiază poziționarea modulelor de senzori, utilizarea breadboard-ului pentru distribuția alimentării și modul de conectare a componentelor.



1. Piezo-electric
2. GSR
3. Temperatura
4. Puls
5. LCD
6. Arduino UNO
7. Breadboard


6
5
4
3
2
1
1
2

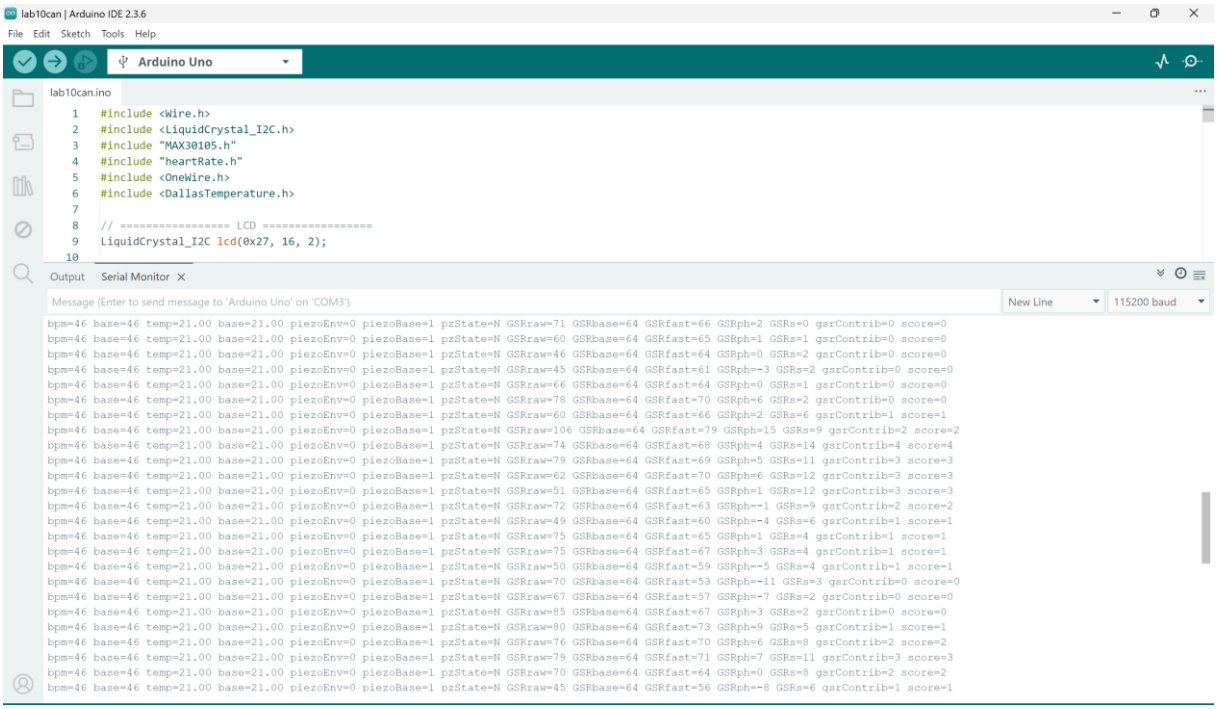


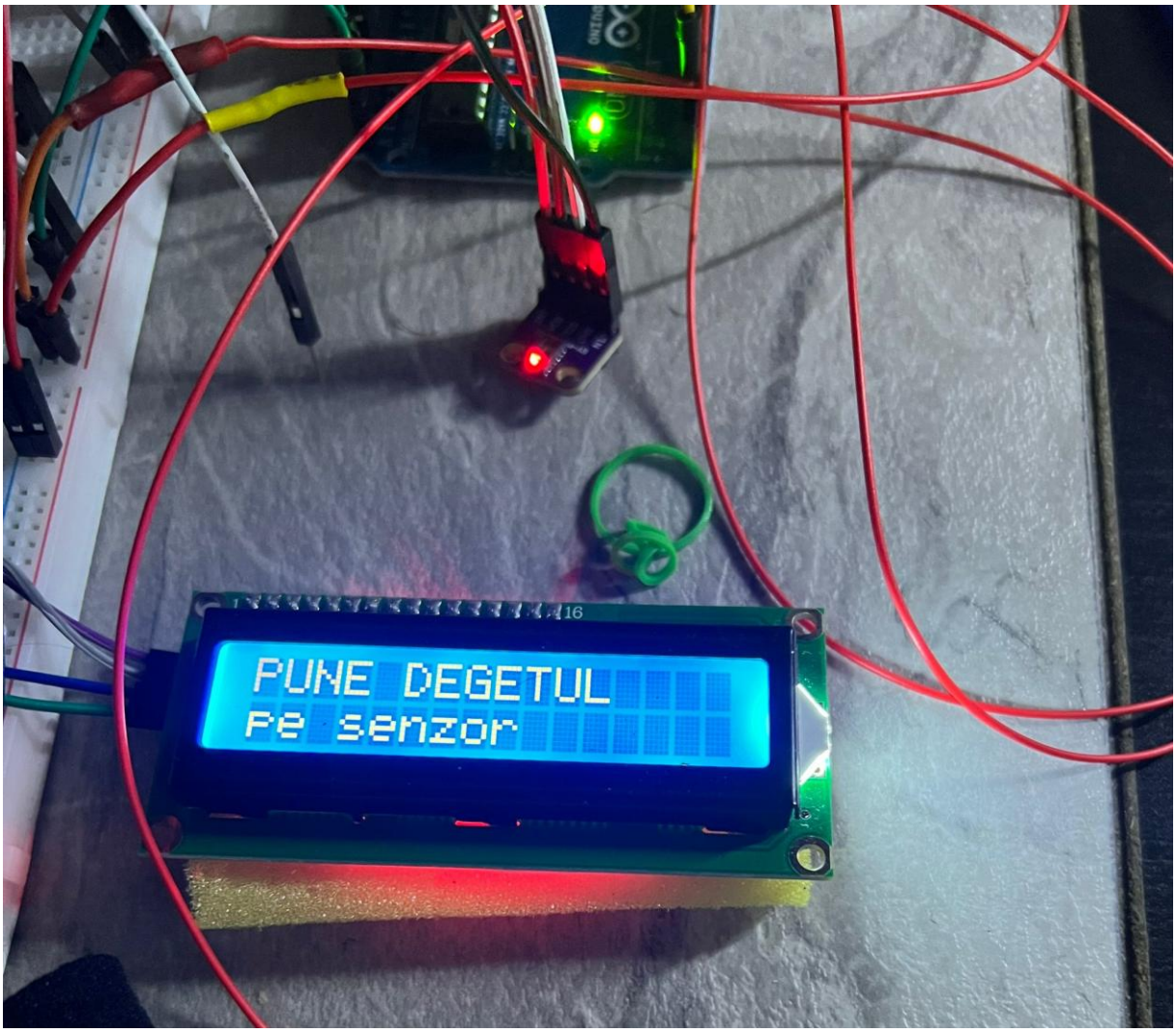


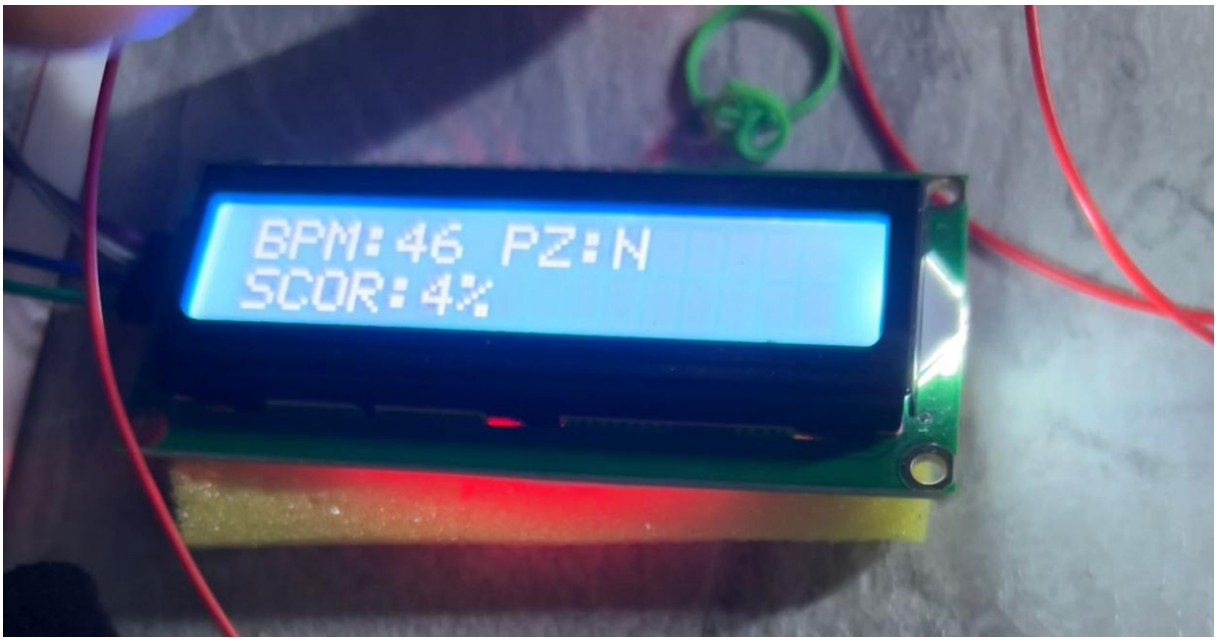
- ***Anexa C – Capturi de ecran ale funcționării sistemului***

În această anexă sunt prezentate capturi de ecran relevante pentru funcționarea sistemului, incluzând:

- afișarea valorilor pe ecranul LCD;
- mesajele de depanare și monitorizare afișate în Serial Monitor;
- exemple de scor de stres obținute în timpul testării.


lab10can | Arduino IDE 2.3.6
File Edit Sketch Tools Help
Arduino Uno
lab10can.ino
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include "MAX30105.h"
#include "heartRate.h"
#include <OneWire.h>
#include <DallasTemperature.h>
// ================ LCD ================
LiquidCrystal_I2C lcd(0x27, 16, 2);
Output Serial Monitor
Message (Enter to send message to 'Arduino Uno' on 'COM3')
New Line
115200 baud
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=71 GSRbase=64 GSRfast=66 GSRph=2 GSRs=0 gsrContrib=0 score=0
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=60 GSRbase=64 GSRfast=65 GSRph=1 GSRs=1 gsrContrib=0 score=0
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=46 GSRbase=64 GSRfast=64 GSRph=0 GSRs=2 gsrContrib=0 score=0
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=45 GSRbase=64 GSRfast=61 GSRph=-3 GSRs=2 gsrContrib=0 score=0
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=66 GSRbase=64 GSRfast=64 GSRph=0 GSRs=1 gsrContrib=0 score=0
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=78 GSRbase=64 GSRfast=70 GSRph=6 GSRs=2 gsrContrib=0 score=0
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=60 GSRbase=64 GSRfast=66 GSRph=2 GSRs=6 gsrContrib=1 score=1
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=106 GSRbase=64 GSRfast=79 GSRph=15 GSRs=9 gsrContrib=2 score=2
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=74 GSRbase=64 GSRfast=68 GSRph=4 GSRs=14 gsrContrib=4 score=4
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=79 GSRbase=64 GSRfast=69 GSRph=5 GSRs=11 gsrContrib=3 score=3
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=62 GSRbase=64 GSRfast=70 GSRph=6 GSRs=12 gsrContrib=3 score=3
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=51 GSRbase=64 GSRfast=65 GSRph=1 GSRs=12 gsrContrib=3 score=3
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=72 GSRbase=64 GSRfast=63 GSRph=-1 GSRs=9 gsrContrib=2 score=2
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=49 GSRbase=64 GSRfast=60 GSRph=-4 GSRs=6 gsrContrib=1 score=1
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=75 GSRbase=64 GSRfast=65 GSRph=1 GSRs=4 gsrContrib=1 score=1
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=75 GSRbase=64 GSRfast=67 GSRph=3 GSRs=4 gsrContrib=1 score=1
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=50 GSRbase=64 GSRfast=59 GSRph=-5 GSRs=4 gsrContrib=1 score=1
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=70 GSRbase=64 GSRfast=53 GSRph=-11 GSRs=3 gsrContrib=0 score=0
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=67 GSRbase=64 GSRfast=57 GSRph=-7 GSRs=2 gsrContrib=0 score=0
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=85 GSRbase=64 GSRfast=67 GSRph=3 GSRs=2 gsrContrib=0 score=0
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=80 GSRbase=64 GSRfast=73 GSRph=9 GSRs=5 gsrContrib=1 score=1
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=76 GSRbase=64 GSRfast=70 GSRph=6 GSRs=8 gsrContrib=2 score=2
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=79 GSRbase=64 GSRfast=71 GSRph=7 GSRs=11 gsrContrib=3 score=3
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=70 GSRbase=64 GSRfast=64 GSRph=0 GSRs=8 gsrContrib=2 score=2
bpm=46 base=46 temp=21.00 base=21.00 piezoEnv=0 piezoBase=1 pzState=N GSRraw=45 GSRbase=64 GSRfast=56 GSRph=-8 GSRs=6 gsrContrib=1 score=1


PUNE DEGETUL
pe senzor


BPM:46 PZ:N
SCOR:4%


GSR CALIBRARE
70%