UNIVERSITATEA TEHNICĂ „Gheorghe Asachi” din IAȘI

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

DOMENIUL: Calculatoare și Tehnologia Informației

SPECIALIZAREA: Calculatoare

MONITORIZAREA ACTIVITĂȚILOR FIZICE

LUCRARE DE LICENȚĂ

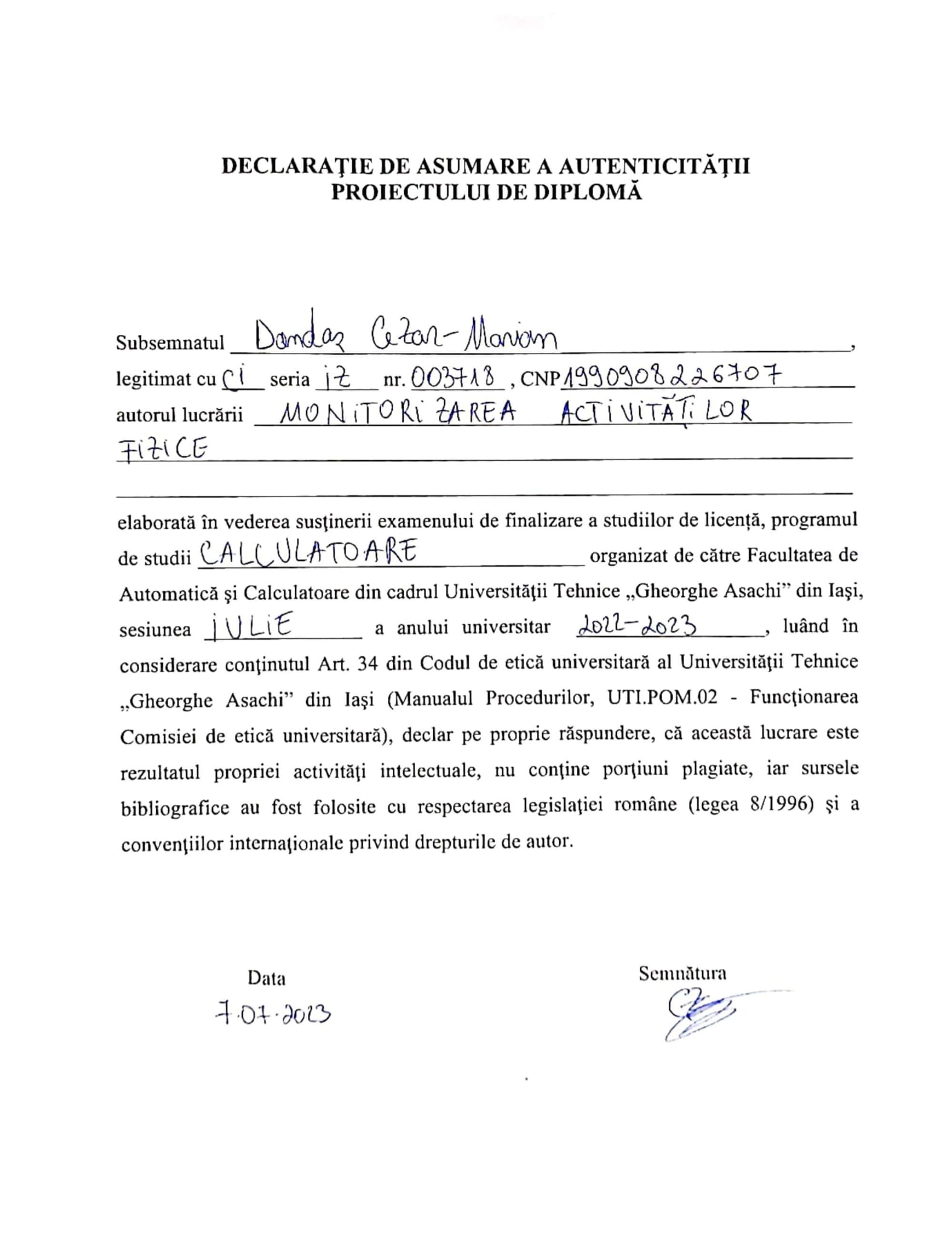
Coordonator științific

Conf.dr. ing. Andrei Stan

Absolvent

Cezar-Marian Dondaș

Iași, 2023



Monitorizarea activităților fizice

Rezumat

Cezar-Marian Dondaș

Scopul acestei teme este de a supraveghea în detaliu anumiți parametri care ne ajută la constatarea aspectelor legate de sănătatea umană și de cunoaștere a frecvenței realizării de activități fizice de-a lungul timpului. De asemenea, se vor avea în vedere aspecte care pot evidenția sedentarismul din cauza efortului fizic scăzut sau deloc, ori un somn agitat de-a lungul nopții. Aceste informații pot fi decisive pentru luarea unei hotărâri în ceea ce privește sănătatea.

Am ales această temă din dorința de aprofundare a cunoștințelor dobândite pe parcursul anilor de studiu și de a le aplica într-un proiect de interes în zilele noastre, observând faptul că, pe piața actuală, există din ce în ce mai multe smartwatch-uri care au aceste funcții ce oferă informații despre activitatea fizică. Specialiștii în domeniul medicinei sugerează că o rutină zilnică de 10 000 de pași reduce depresia și anxietatea, îmbunătățește calitatea somnului, previne apariția unor afecțiuni precum boli de inimă, accident vascular cerebral și hipertensiune arterială.

Monitorizarea pașilor efectuați în timpul zilei este relevantă în diagnosticarea bolilor sau afecțiunilor de care suferim. Un senzor incorporat în propriul telefon mobil sau în smartwatch este capabil să măsoare numărul de pași și chiar să se tragă o concluzie despre calitatea somnului bazată pe citirea valorilor accelerometrului și a giroscopului.

În tema propusă se vor detalia soluțiile abordate, se vor menționa problemele întâlnite și rezolvarea acestora pe parcursul realizării și documentării proiectului.

Cuprins

[Introducere 2](#_Toc139539190)

[Capitolul 1. Prezentare generală 3](#_Toc139539191)

[1.1 Caracteristici 3](#_Toc139539192)

[1.1.1 Caracteristici giroscop 3](#_Toc139539193)

[1.1.2 Caracteristici accelerometru 3](#_Toc139539194)

[1.1.3 Caracteristici magnetometru 3](#_Toc139539195)

[1.1.4 Alte caracteristici ale plăcii de dezvoltare MPU-9250 3](#_Toc139539196)

[1.2 Convertori analog-digitali pe 16 biți 4](#_Toc139539197)

[1.2.1 Giroscop 4](#_Toc139539198)

[1.2.2 Accelerometru 4](#_Toc139539199)

[1.2.3 Magnetometru 4](#_Toc139539200)

[1.3 Digital Motion Processor(DMP) 4](#_Toc139539201)

[1.4 Interfețe de comunicare serială 5](#_Toc139539202)

[1.4.1 Comunicarea serială I2C 5](#_Toc139539203)

[1.4.1.1 Protocolul I2C 6](#_Toc139539204)

[1.4.2 Comunicarea serială SPI 8](#_Toc139539205)

[1.5 Registrele de date ale senzorilor 8](#_Toc139539206)

[1.6 FIFO 8](#_Toc139539207)

[1.7 Întreruperi 8](#_Toc139539208)

[1.8 Clocking 9](#_Toc139539209)

[1.9 Descrierea registrelor 9](#_Toc139539210)

[1.9.1 Divizorul de rată de eșantionare 9](#_Toc139539211)

[1.9.2 Configurație giroscop și accelerometru 10](#_Toc139539212)

[1.9.3 FIFO Enable 10](#_Toc139539213)

[1.9.4 MPU Configuration 11](#_Toc139539214)

[1.9.5 External sensor data 11](#_Toc139539215)

[1.9.6 Power management 1 12](#_Toc139539216)

[Capitolul 2. Proiectarea aplicației 13](#_Toc139539217)

[2.1 Configurare Raspberry Pi 13](#_Toc139539218)

[2.2 Proiectarea hardwarea a aplicației 13](#_Toc139539219)

[2.3 Protocolul Secure Shell 15](#_Toc139539220)

[2.4 Arhitecturi utilizate 16](#_Toc139539221)

[2.4.1 Arhitectura master-slave 16](#_Toc139539222)

[2.4.2 Arhitectura client-server 17](#_Toc139539223)

[Capitolul 3. Implementarea aplicației 19](#_Toc139539224)

[3.1 Citirea valorilor de pe axele accelerometrului și giroscopului 19](#_Toc139539225)

[3.2 Obținerea și interpretarea datelor citite 20](#_Toc139539226)

[3.3 Algoritmi propuși 22](#_Toc139539227)

[3.3.1 Detecție pași 22](#_Toc139539228)

[3.3.2 Determinarea calitatății somnului 22](#_Toc139539229)

[3.3.3 Testarea aplicației 23](#_Toc139539230)

[Capitolul 4. Utilizarea proiectului 26](#_Toc139539231)

[4.1 Funcționalitate 26](#_Toc139539232)

[4.2 Îmbunătățiri viitoare 27](#_Toc139539233)

[Concluzie 28](#_Toc139539234)

[Bibliografie 30](#_Toc139539235)

# **Introducere**

În acest document vor exista multiple referințe din (InvenSense, 2015) și (InvenSense, 2016).

În proiectul ce va fi prezentat, se vor folosi dispozitivul de urmărire a mișcării MPU-9250 pentru achiziția valorilor de pe fiecare axă a fiecărui senzor și Raspberry PI 2 , un computer de mici dimensiuni, de marimea unui card bancar având un procesor ARM cu sistem de operare Raspian, având o distribuție linux.

MPU-9250 este un modul multicip format din două blocuri integrate într-un singur pachet numit QUAD FLAT NO-LEAD [[1]](#footnote-1). Un bloc este alcătuit din accelerometru și giroscop fiecare senzor cu câte trei axe. Celălalt bloc reprezintă AK8963, magnetometru de la Asahi Kasei Microdevices Corporation [[2]](#footnote-2). MPU-9250 este un dispozitiv de urmărire a mișcării pe nouă axe care combină un giroscop cu trei axe, un accelerometru cu trei axe, un magnetometru cu trei axe și un procesor digital de mișcare. Acesta este conceput, de asemenea, pentru a se interfața cu o gamă variată de senzori digitali care nu sunt inerțiali prin intermediul conexiunii auxiliare I2C [[3]](#footnote-3). Dispozitivul de monitorizare a mișcării cu cele nouă axe integrate oferă consumatorilor o performanță optimă mișcării, iar calibrările efectuate acestuia permit producătorilor să elimine eventualele costuri suplimentare.

MPU-9250 dispune de trei convertoare analog-digitale pe 16 biți pentru digitizarea ieșirilor giroscopului, accelerometrului și magnetometrului. Pentru urmărirea cu precizie a mișcărilor rapide și lente, componentele dispun de un sistem programabil de către utilizator. Giroscopul are o gamă completă de ±250, ±500, ±1000 și ±2000°/sec (dps), accelerometrul ±2g, ±4g, ±8g și ±16g și magnetometrul cu o gamă completă de ±4800µT.[[4]](#footnote-4)

Alte caracteristici din industrie includ filtre digitale programabile, un ceas de precizie de 1% de la -40°C până la 85°C, un senzor de temperatură încorporat și întreruperi programabile.

Comunicarea cu toate registrele dispozitivului se desfășoară folosind fie I2C la 400kHz, fie SPI la 1MHz.

Raspberry PI 2 este un computer care conține un procesor cu patru nuclee ARM Cortex-A7 de 900Mhz, 1 GIGABYTE memorie RAM, patru porturi USB, ieșire HDMI, port pentru cablu Ethernet, placă video integrată, interfață pentru cameră și afișare. Cu ajutorul unui card microSD vom adăuga și distribuția linux Raspian.

Comunicarea celor două se va realiza prin interfața serială I2C, unde senzorul mereu va avea un comportament de tip “slave”. Plăcuța va prelua controlul și va juca rol de “master”.

# **Capitolul 1. Prezentare generală**

## ***Caracteristici***

### *Caracteristici giroscop*

* Senzori de viteza unghiulara cu ieșire digitală pe axele X, Y și Z cu o gamă completă programabilă și convertoare integrate pe 16 biți;
* Filtru trece-jos programabil digital;
* Curentul de funcționare 3.2mA;
* Curent în modul „sleep” : 8 µA;
* Autotestare.

### *Caracteristici accelerometru*

* Curent normal de funcționare de 450µA;
* Trei axe cu ieșire digitala cu o gamă completa programabilă și convertoare pe 16 biți;
* Curent în modul de consum redus de 8,4µA la 0,98Hz, 19,8µA la 31,25Hz;
* Curent în modul „sleep” de 8µA;
* Întreruperi programabile de către utilizator.

### *Caracteristici magnetometru*

* Senzor magnetic cu efect Hall monolitic pe trei axe din siliciu;
* Rezoluție a datelor de ieșire de 14 biți (0.6µT/LSB);
* Limita maximă de măsurare este de ±4800µT;
* Curentul de funcționare normală a magnetometrului 280µA la o rată de 8Hz;
* Autotestare.

### *Alte caracteristici ale plăcii de dezvoltare MPU-9250*

* Magistrala I2C pentru citirea datelor de la senzori externi;
* 3.5 mA curent de funcționare atunci cand toate cele nouă axe de detectare a mișcării și DMP sunt activate;
* Tensiune de alimentare de la 2.4V la 3.6V;
* Buffer FIFO de 512 octeti care asigură procesorului de aplicații să citească mai multe date simultan;
* Senzor de temperatură;
* Tolerantă la șocuri de 10.000 g;
* Mod rapid pentru comunicarea I2C cu toate registrele de 400kHz;
* Interfață seriala SPI de 1MHz pentru comunicarea cu toate registrele.

## ***Convertori analog-digitali pe 16 biți***

### *Giroscop*

Atunci când giroscopul este rotit în jurul oricărei dintre axele X,Y, Z efectul Coriolis provoacă o vibrație care este detectată de un dispozitiv de preluare capacitivă. Semnalul obținut este amplificat, demodulat și filtrat pentru a produce o tensiune care este proporțională cu viteza unghiulară. Această tensiune este transformată prin intermediul convertorului analog-digital care eșantioneaza fiecare axă. Intervalul maxim al senzorului pentru fiecare axă poate fi programat digital la ±250, ±500, ±1000 sau la ±2000 grade pe secundă (dps). Rata de eșantionare a convertorului analog-digital este programabilă de la 8000 de eșantioane pe secundă până la 3.9 eșantioane pe secundă.

### *Accelerometru*

Accelerometrul utilizează mase de probă separate pentru fiecare axă. Accelerația induce o deplasare pe masa de probă corespunzatoare, iar senzorii capacitivi detectează deplasarea diferențiat. Când dispozitivul este plasat pe o suprafață plană, acesta măsoară 0g pe axele X și Y și +1g pe axa Z. Factorul de scalare al acestuia este calibrat în fabrică și este independent de tensiunea de alimentare. Fiecare senzor are un convertor analog-digital dedicat pentru furnizarea ieșirilor digitale. Intervalul complet pentru acestea poate fi ajustat la ±2g, ±4g, ±8g sau ±16g.

### *Magnetometru*

Magnetometrul cu trei axe utilizează tehnologia efectului Hall. Efectul Hall poate fi aplicat într-un magnetometru pentru a identifica și măsura câmpul magnetic. Partea magnetometrică a circuitului conține senzori magnetici pentru detectarea magnetismului terestru pe axele X, Y, Z, un senzor de acționare, un lanț de amplificare a semnalului și un circuit aritmetic pentru procesarea semnalului de la fiecare senzor. Fiecare convertor analog-digital are 16 biți și o gamă completă de ±4800 µT.

## ***1.3 Digital Motion Processor(DMP)***

Procesorul încorporat este integrat MPU-9250 și transferă calculul algoritmilor de la procesorul gazdă. DMP-ul achiziționează datele de la accelerometru, giroscop, magnetometru și senzori externi procesându-le. Acesta are posibilitatea de a accesa unul dintre pinii externi ai MPU-9250 care poate fi utilizat în generarea de întreruperi. Pinul 12 trebuie conectat pe un altul de pe procesorul gazdă care o și trezește din modul „suspend”.

Scopul DMP este de a descărca puterea de procesare de la procesorul gazdă, cât și cerințele de sincronizare. Acesta poate fi folosit ca un instrument de minimizare a puterii, pentru simplificarea sincronizarii și arhitecturii software fiind lucruri utile într-o aplicație.

## ***Interfețe de comunicare serială***

MPU-9250 comunică cu un procesor al unui sistem electronic , unde vom vedea în următoarele capitole că ne referim la sistemul de calcul echipat cu procesor ARM, fie folosind SPI [[5]](#footnote-5) sau I2C [[6]](#footnote-6). MPU funcționează întotdeauna ca „slave” în comunicarea cu acesta. Prin serial înțelegem că transmiterea informației se va efectua bit cu bit.

### *Comunicarea serială I2C*

I2C este o interfață care utilizează doar două fire pentru transmiterea datelor între dispositive. SDA reprezintă linia cu datele seriale care vor fi trimise și primite pentru „master” și „slave”. SCL este linia care transportă semnalul de ceas. Acestea sunt bidirecționale. Dispozitivul „master” [[7]](#footnote-7) trimite adresa „slave-ului” pe magistrală, iar dispozitivul „slave” [[8]](#footnote-8) cu adresa respectiva recunoaște „master-ul”. Viteza maximă a magistralei este 400kHz.

MPU-9250 are la dispoziție un circuit auxiliar I2C pentru comunicarea cu senzorii externi. Magistrala are două moduri de funcționare:

* „I2C Master Mode”: Placa de dezvoltare se comportă ca și „master” pentru orice senzor extern. Modul permite acesteia să acceseze direct registrele senzorilor externi astfel obținând date de la ei fără ca procesorul de sistem electronic să intervină;
* „Pass-Through Mode”: În acest caz, senzorii externi comunică direct cu procesorul sistemului. Modul permite acestuia să joace rol de „master” comunicând direct cu senzorii care sunt conectați la pinii auxiliari ai magistralei I2C „AUX\_DA” și „AUX\_CL”. Așadar, logica de control a magistralei pentru modul anterior va fi dezactivată. Configurarea prezentată este utilă pentru a accesa magnetometrul AK8963 direct de la gazdă. El va primi adresa „slave” 0x0C sau 12 zecimal. Este important de reținut faptul că AK8963 se va folosi ca un senzor extern sau cum este denumit în documentație, senzor terț.

MPU-9250 are abilități reduse ca „master” I2C și depinde de procesorul de sistem în vederea gestionării configurării inițiale a oricarui senzor extern. Are un multiplexor de „bypass” care asigură conectarea magistralei I2C al procesorului prin pinii „SDA” și „SCL” direct la magistrala I2C a senzorilor auxiliari. Dupa configurarea acestora de către procesor, trebuie dezactivat multiplexorul interfeței deoarece auxiliarul „I2C master” al MPU-9250 să preia controlul magistralei I2C a senzorilor și să colecteze date de la senzorii auxiliari.

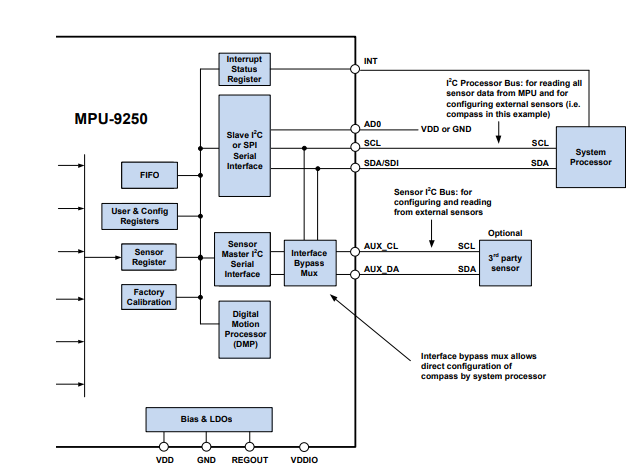


Figura 1.1 Soluție utilizând interfața I2C

#### 1.4.1.1 Protocolul I2C

Comunicarea pe magistrala I2C începe când master-ul trece condiția START(S) pe magistrală care este caracterizată ca o tranziție de la high la low a liniei SDA în timp ce linia SCL este pe high. Putem spune că magistrala este ocupată până când master-ul pune condiția de oprire(P) pe magistrală fiind definită ca o tranziție de la low la high pe linia SDA, SCL fiind pe high. Magistrala continuă să fie ocupată dacă se genereaza un START repetat(Sr) în loc de o condiție STOP.

Octeții de date I2C sunt definiți având lungimea de 8 biți. Nu există vreo restricție în ceea ce privește numărul de octeți transmiși per date de transfer. Fiecare octet transferat este urmat de un semnal de confirmare numit „acknowledge”, găsit și ca ACK în documentație. Acesta este produs de master, în timp ce receptorul generează semnalul de confirmare propriu-zis trăgând jos SDA și păstrându-l pe low pe porțiunea de high a impulsului de ceas al ACK.

Dacă un slave este ocupat și nu poate transmite sau primi un alt octet cât timp operațiunea nu este îndeplinită, acesta poate menține SCL pe low, determinând master-ul să intre în stare de așteptare. Transferul normal de date se restabilește în momentul în care slave-ul este gata și eliberează linia de ceas.

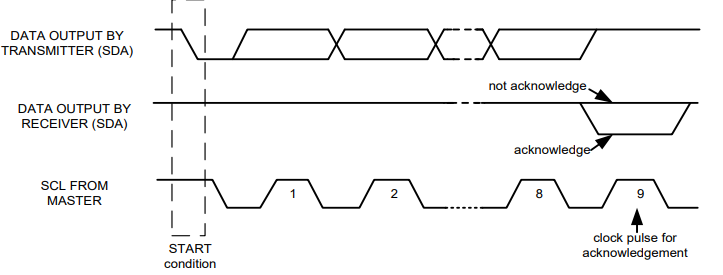


Figura 1.2 Asertare magistrală I2C

După începerea cu condiția de START(S), master-ul trimite o adresă slave de 7 biți urmată de un al 8-lea bit, cel de citire/scriere. Acesta specific dacă master-ul date de la slave sau dacă scrie date pe acesta. Master-ul eliberează linia SDA și așteaptă semnalul de confirmare ACK de la slave. Fiecare octet trimis este urmat de un bit de confirmare. Pentru aceasta, dispozitivul slave păstrează linia SDA pe low cât timp linia SCL este pe high. Transmisia de date este considerată încheiată atunci când master-ul pune condiția de STOP(P), astfel eliberând linia de comunicare. O tranziție de la low la high pe linia SDA, în timp ce linia SCL este pe high stabilește condiția de stop. Toate schimbările SDA au loc cât timp SCL este pe low, cu excepția condițiilor de STOP și START.

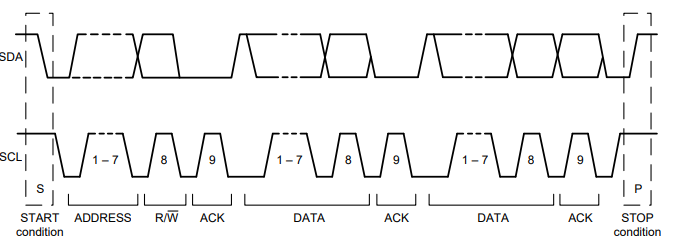


Figura 1.3 Transfer complet de date I2C

### *1.4.2 Comunicarea serială SPI*

SPI este o interfață serială sincronă cu patru fire care folosește două linii de control și două linii de date. Ca și in cazul de mai sus, plăcuța de dezvoltare mereu va fi de tip „slave”. Semnalele CS, SDO, SCLK și SDI sunt utilizate pentru comunicarea SPI. Pinii acestora sunt partajați cu pinii „slave-ului” I2C, procesorul de sistem nu are acces la magistrala auxiliară I2C prin intermediul mutiplexorului de „bypass” care leagă pinii de interfața I2C ai procesorului cu cei de pe senzor. CS reprezintă linia pentru „master” care selectează „slave-ul” cui să trimită informații. MISO este linia care trimite date de la „slave” la „master”, iar MOSI reprezintă opusul, de la „master” la „slave”. Ieșirea de date seriale(SDO), ieșirea de ceas serial(SCLK) și ieșirea de date de intrare(SDO) sunt partajate între dispozitive „slave”. Fiecare dispozitiv „slave” SPI necesita propria linie de selectare a cipului(CS) de la „master”.

Când se folosește SPI ca și metoda de comunicare între MPU-9250 și celălalt dispozitiv care il va controla, configurearea acestora pe magistrala auxiliară I2C pentru senzori poate fi posibilă prin intermediul „I2C slaves 0-4” în vederea efectuării de tranzacții pentru citire și scriere pe orice dispozitiv. Imediat ce senzorii externi au fost configurați, se pot efectua citiri pe unul sau mai muți octeți.

Atunci când CS-ul este pe „low” la începutul transmiterii linia este activa, iar la final revine pe „high” și devine inactivă. Doar o singură linie CS este activă la un moment dat, ceea ce garantează selectarea unui singur „slave” la un moment dat. Liniile CS ale dispozitivelor „slave” neselectate sunt ținute pe „high”, determinând ca liniile SDO ale acestora să fie menținute într-o stare de impedanță „high-z” astfel încât să nu interfereze cu alt dispozitiv activ.

## ***Registrele de date ale senzorilor***

Registrele de date ale acestora cuprind cele mai recente date de măsurare a accelerometrului, giroscopului, magnetometrului, senzorului auxiliar și a temperaturii. Acestea sunt registre doar pentru citire și sunt obținute prin intermediul interfeței seriale. Datele pot fi citite oricând.

Senzorul de temperatura de pe cip și un convertor analog-digital sunt utilizate pentru a măsura temperatura MPU-9250. Citirile sunt citite din FIFO sau din registrele de date ale senzorului.

## ***FIFO***

MPU-9250 conține un registru FIFO de 512 octeți care este disponibil prin interfața serială. Registrul FIFO determină ce date sunt scrise. Printre opțiuni se regăsesc datele de la giroscop și accelerometru, citiri de temperatură, citiri ale senzorului auxiliar și intrarea FSYNC. Un contor FIFO reține câți octeți de date valide sunt conținute în FIFO. Registrul FIFO suportă mai multe citiri simultan. Funcția de întrerupere poate fi folosită pentru a identifica momentul în care sunt disponibile date noi.

## ***Întreruperi***

Funcționalitatea întreruperilor este configurată cu ajutorul registrului INT. Elementele care pot fi configurate includ configurația pinului INT, metoda de blocare și compensare a întreruperii și declanșatoarelor pentru întrerupere. Componentele care declanșează o întrerupere sunt: generatorul de ceas blocat la noul oscilator de referință, noi date sunt pregătite pentru a fi citite(din registrele FIFO și registrele de date), întreruperi ale accelerometrului și MPU-9250 nu a primit confirmare de la un senzor auxiliar pe magistrala I2C secundară. Starea întreruperii este citită din registrul Interrupt Status. Pinul INT ar trebui conectat la un pin de pe procesorul gazdă cu posibilitatea de a-l trezi din starea „suspend”.

## ***Clocking***

MPU-9250 are o schemă flexibilă de sincronizare, facilitând folosirea unei varietăți de surse de ceas intern pentru circuitele interne sincrone. Acest circuit sincron cuprinde circuitele de condiționare a semnalului si convertoare analog-digitale, DMP-ul împreuna cu diverse circuite și registre de control.

Selectarea sursei pentru generarea ceasului intern sincron depinde de necesitățile privind consumul de energie și precizia ceasului. Aceste cerințe vor varia adesea în funcție de modul de operare. De exemplu, într-un mod, în carea cea mai mare îngrijorare este consumul de energie, este posibil ca utilizatorul să dorească să opereze dispozitivul DMP al MPU-9250 pentru a procesa datele accelerometrului, ținând giroscoapele dezactivate. În acest caz, sistemul oscilator intern de relaxare este o alegere bună pentru ceas. Cu toate acestea, într-un alt scenariu, în care giroscoapele sunt active, selectarea acestora ca sursă de ceas asigură o sursă de ceas mai precisă.

Precizia ceasului este importantă, deoarece erorile de sincronizare afectează în mod direct calculele distanței și unghiurile realizate de catre DMP.

Există și alte condiții de pornire care trebuie luate în considerare. Când MPU-9250 pornește pentru prima dată, dispozitivul utilizeaza ceasul său intern până când este programat să funcționeze de la o altă sursă. Acest fapt îi conferă utilizatorului, de exemplu, posibilitatea de a aștepta pentru oscilatoarele MEMS să se stabilizeze înainte de a fi selectate ca sursă de ceas.

## ***Descrierea registrelor***

Aici se vor oferi informații privind registrele pentru Motion Processing Unit – MPU-9250. Se vor regăsi registrele care s-au folosit și au avut impact asupra proiectului. Numele acestora care se termină în \_H și \_L conțin octeții de high și low, respectiv valorile lor. Numele registrelor se va regăsi scris cu majuscule, în timp ce valorile vor fi și scrise italic. Spre exemplu, registrul ACCEL\_XOUT\_H conține 8 cei mai semnificativi biți, *ACCEL\_XOUT[15:8],* din cei 16 biți corespunzatori pentru axa X de măsurare a accelerometrului, *ACCEL\_XOUT.* Valoarea de reset este 0x00 pentru toate registrele, în afară de:

* Registrul 107 – Power Management 1, fiind 0x01;
* Registru 117 – WHO\_AM\_I, fiind 0x71.

### *1.9.1 Divizorul de rată de eșantionare*

Registrul 25 pe 8 biți SMPLRT\_DIV[7:0] împarte rata de eșantionare internă pentru a genera frecvența de eșantionare internă care controlează rata de ieșire a datelor senzorului, rata de eșantionare FIFO. Rata de actualizare a registrului senzorului este:

SAMPLE\_RATE= Internal\_Sample\_Rate / (1 + SMPLRT\_DIV). Datele trebuie să fie eșantionate la o frecvență de eșantionare mai mare sau egală; SMPLRT\_DIV este utilizat numai pentru eșantionarea internă de 1 kHz.

### *1.9.2 Configurație giroscop și accelerometru*

În cazul giroscopului, registrul 27 conține de la bitul 0 la 1, Fchoice\_b[1:0] utilizat pentru a ignora DLPF. Bitul 2 este rezervat. De la bitul 3 la 4 este selecția GYRO\_FS\_SEL[1:0] unde avem selectarea completă a gamei giroscopului cu următoarele:

* 00 = +250dps
* 01= +500 dps
* 10 = +1000 dps
* 11 = +2000 dps

Bitul 5 conține ZGYRO\_Cten cu funcția Z Gyro self-test, bitul 6 conține YGYRO\_Cten cu funcția Y Gyro self-test și bitul 7 conține XGYRO\_Cten cu funcția X Gyro self-test. Autotestarea face referire la valoarea dintr-un registru care precizează ieșirea de autotestare generată în timpul testelor de fabricație. Această valoare trebuie utilizată pentru a verifica cu ieșirile ulterioare de autotestare efectuate de către utilizatorul final.

În cazul accelerometrului, registrul 28 conține primii biți de la 0 până la 2 fiind rezervați. De la bitul 3 la 4 este selecția ACCEL\_FS\_SEL[1:0], unde indică selectarea completă a gamei accelerometrului cu următoarele: ±2g (00), ±4g (01), ±8g (10), ±16g (11). Biții 5,6 și 7 conțin az\_st\_en, ay\_st\_en, ax\_st\_en cu funcție de autotestare pentru fiecare axă. Mai sus este și explicația pentru aceasta care este valabilă și în cazul accelerometrului.

### *1.9.3 FIFO Enable*

Registrul 35 conține 8 biți care sunt: [0] - SLV\_0, [1] - SLV\_1, [2] - SLV\_2, [3] – ACCEL, [4] - GYRO\_ZOUT, [5] - GYRO\_YOUT, [6] - GYRO\_XOUT, [7] - TEMP\_OUT. Se vor prezenta cazurile doar când biții sunt pe 1, deoarece când biții au valoarea 0 funcționalitatea pentru fiecare dintre aceștia este dezactivată. SLV\_0, SLV\_1, SLV\_2 scriu registrele lui EXT\_SENS\_DATA(va fi detaliat mai târziu) asociate la SLV\_0, SLV\_1, SLV\_2 în FIFO la rata de eșantionare. ACCEL scrie ACCEL\_XOUT\_H, ACCEL\_XOUT\_L, ACCEL\_YOUT\_H, ACCEL\_YOUT\_L, ACCEL\_ZOUT\_H, ACCEL\_ZOUT\_L în FIFO la rata de eșantionare. GYRO\_ZOUT, GYRO\_YOUT, GYRO\_XOUT scriu GYRO\_ZOUT\_H și GYRO\_ZOUT\_L în FIFO la rata de eșantionare. Dacă este activat punerea în buffer a datelor are loc chiar dacă datele sunt în stand-by. TEMP\_OUT scrie TEMP\_OUT\_H și TEMP\_OUT\_L în FIFO la rata de eșantionare. Atunci când este activat, se întamplă exact același lucru ca și în cazul anterior cu biții [4], [5], [6].

### *1.9.4 MPU Configuration*

Registrul 26 are 8 biți care sunt reprezentați de: [2:0] - DLPF\_CFG[2:0], [5:3] EXT\_SYNC\_SET[2:0], [6] - FIFO\_MODE, [7] – rezervat. Biții [2:0] sugerează că DLPF\_CFG[[9]](#footnote-9) trebuie să aibă setat fchoice[1:0][[10]](#footnote-10) la 2'b11, fchoice\_b[1:0] fiind 2'b00. Câmpul de biți [5:3] este reprezentat de EXT\_SYNC\_SET[2:0][[11]](#footnote-11) care permite ca datele pinului FSYNC să fie eșantionate. Fsync va fi activat pentru a capta semnale scurte. Acest fapt se va produce astfel încât, daca fsync comută, valoare blocată se comută, dar nu se va comuta din nou până când noua valoare blocată este captată de semnalul ratei de eșantionare. Bitul [6] pentru FIFA\_MODE este utilizat atunci când coada este plină. Dacă bitul este setat pe 1, nu se va mai scrie nimic în aceasta. Dacă este setat pe 0, se vor înlocui datele vechi scriindu-se în coada altele noi. Bitul [7] este rezervat pentru alte funcționalități.

DLPF este configurat de DLPF\_CFG pentru FCHOICE\_B [1:0] = 2b'00. Giroscopul și senzorul de temperatură sunt filtrate în funcție de valoarea lui DLPF\_CFG și FCHOICE\_B corespunzator imaginii de mai jos. A se observa că FCHOICE menționat în tabel este valoarea inversată a lui FCHOICE\_B. Exemplu fiind FCHOICE = 2b’00 este echivalent cu FCHOICE\_B = 2b’11.

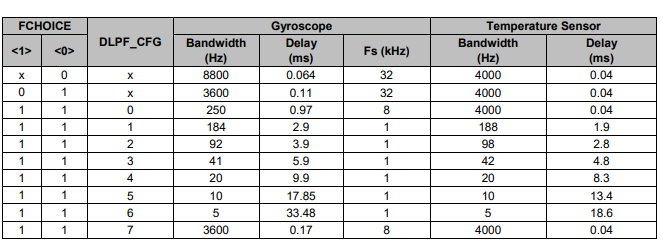


Figura 1.4 Filtrarea semnalelor giroscopului și senzorului de temperatură

### *1.9.5 External sensor data*

Octetul [7:0] numit D[7:0] reprezintă datele senzorului de la dispozitivele I2C externe prin intermediul interfeței master I2C. Datele stocate sunt controlate de registrele I2C\_SLV(0-4)\_ADDR, I2C\_SLV(0-4)\_REG și I2C\_SLV(0-4)\_CTRL. Aceste registre stochează datele citite de la senzorii externi de către Slave 0, 1, 2, 3 pe interfața auxiliară I2C. Datele citite de la Slave 4 sunt stocate în I2C\_SLV4\_DI(registrul 53).

Datele senzorilor externi sunt scrise în aceste registre la rata de eșantionare definită în registrul 25 de mai sus. Această rată de acces poate fi redusa prin utilizarea registrelor Slave Delay Enable (registrul 103).

Datele sunt plasate în aceste registre de date ale senzorilor externi în conformitate cu I2C\_SLV0\_CTRL, I2C\_SLV1\_CTRL, I2C\_SLV2\_CTRL și I2C\_SLV3\_CTRL (registrele 39, 42, 45 și 48). Când se citesc mai mult de zero octeți (I2C\_SLVx\_LEN > 0) de la un slave activat (I2C\_SLVx\_EN = 1), slave-ul este citit la rata de eșantionare (așa cum este definită în registrul 25) sau la rata întârziată. În timpul fiecărui ciclu de eșantionare, citirile slave-urilor sunt efectuate în ordinea numărului de slave-uri. În cazul în care toate slave-urile sunt activate cu mai mult de zero octeți de citit, ordinea va fi Slave 0, urmat de Slave 1, Slave 2 și Slave 3. Fiecărui slave activat i se vor asocia registrele EXT\_SENS\_DATA în funcție de numărul de octeți citiți. (I2C\_SLVx\_LEN) în ordinea numărului de slave, începând de la EXT\_SENS\_DATA\_00.

### *1.9.6 Power management 1*

Registrul 107 regăsit în documentație și cu numele de PWR\_MGMT\_1, are 8 biți.

Primii 3 denumiți și CLKSEL[2:0] reprezintă:

* Pentru 000 (0) și 110 (6) sursa de ceas este oscilatorul intern de 20Mhz.
* Pentru 001 (1), 010 (2), 011 (3), 100(4) și 101(5) sursa de ceas selectează automat cea mai bună sursă de ceas disponibilă – PPL dacă este pregatită, altfel se utilizează oscilatorul intern.
* Pentru 111 (7) sursa de ceas oprește ceasul și menține generatorul de sincronizare în resetare.

Bitul [3] numit și PD\_PTAT[[12]](#footnote-12) indică dezactivarea generatorului intern de tensiune PTAT și a convertorului analog-digital PTAT. Bitul [4] atunci când este setat sunt activate circuitele de acționare a giroscopului si PLL[[13]](#footnote-13), dar rutele de detecție sunt dezactivate. Aceasta este un mod cu consum redus de energie care permite activarea rapidă a circuitului giroscoapelor. În momentul setării bitului [5] CYCLE, atunci când și SLEEP și STANDBY nu sunt setate, cipul va efectua un ciclu între „sleep” și obținerea unei singure eșantionări la o rată determinate de registrul LP\_ACCEL\_ODR. În momentul în care axele accelerometrului sunt dezactivate prin PWR\_MGMT\_2 biții registrului și ciclul este activat, cipul se va „trezi” la rata determinată de registrele respective de mai sus, dar nu va lua nici o eșantionare. Nu va fi cazul dezactivării axelor accelerometrului pentru proiectul propus. SLEEP, Bitul [6] când este setat, cipul este pus pe modul „sleep”. Când bitul [7] este pus pe 1 acesta resetează registrele interne și restabilește setarile implicite.

# **Capitolul 2. Proiectarea aplicației**

Am menționat în introducere faptul că dispozitivul de urmărire a mișcării MPU-9250 și calculatorul de mici dimensiuni Raspberry Pi 2 comunică prin interfața seriala I2C. Master-ul care preia controlul fiind cel din urmă, iar MPU-9250 este de tip slave. Pentru o mai bună implementare a aplicației, am realizat o conectare securizată de la distanță prin protocolul SSH în vederea accesului la datele stocate de pe Raspberry Pi. Cu ajutorul acestui protocol am reușit să adaug aplicației și o arhitectură client-server, în care server-ul preia datele necesare de la senzor le trimite client-ului , adică pe sistemul de operare windows de pe laptop unde se vor procesa și se vor efectua calculele necesare detectării pașilor și precizarea calității somnului.

## ***Configurare Raspberry Pi***

În primă fază, trebuie instalat un sistem de operare numit Raspian pe plăcuța de mici dimensiuni. Acesta se va pune pe un card microSD de minimum 8GB. Eu am optat pentru unul de 128GB. Instalarea este facilitată de Raspberry Pi Imager care iți generează o imagine pe card. Dupa aceasta, sunt necesare un monitor, un mouse, un cablu HDMI și o tastatură pentru a efectua toate setările de pe sistemul de operare. Prima setare va fi activarea interfețelor I2C și SSH prin folosirea comenzii sudo raspi-config din terminal. Pentru acces la internet am ales să utilizez un adaptor USB wireless tenda care va ajuta să se conectez placa la internet. Comanda hostname -I va furniza adresa IP proprie care va fi utilizată în realizarea conectării cu nume de utilizator și parolă. Raspberry Pi și laptopul de unde se dorește acest lucru trebuie să fie conectate la aceeași rețea wifi.

Alimentarea plăcuței este un factor care de multe ori este neglijat. Pentru a evita sa se supună la o tensiune insuficientă am optat pentru un alimentator original Raspberry Pi recomandat de producător. Această soluție mi-a răspuns la multe intrebări pe parcursul proiectării și implementării proiectului. Comanda vcgencmd get\_throttled a indicat această problemă furnizând un cod de eroare de la kernel[[14]](#footnote-14).

## ***Proiectarea hardwarea a aplicației***

În urma configurărilor corespunzătoare pentru ca funcționarea să se realizeze în parametri optimi a Raspberry Pi, s-au efectuat legăturile între pinii celor două plăcuțe (Figura 2.1) . S-au legat firele pentru pinii de tensiunea de alimentare și masă, VCC respectiv GND, pinii pentru comunicarea serială I2C, linia care trimite și primește date și linia pentru semnalul de ceas, SDA, respectiv SCL (vezi imaginile Figura 2.2 , Figura 2.3 ).

O dată efectuate acestea, se verifică dacă conexiunea dintre MPU-9250 și Raspberry Pi prin I2C există. Acest lucru se face cu următoarea comandă scrisă în terminal: sudo i2cdetect –y 1. Comanda va afișa o matrice unde se regăsește o valoare 68 care reprezintă adresa dispozitivului MPU-9250 corespunzătoare accelerometrului și giroscopului ca fiind 0x68. Adresa pentru magnetometru este 0x0C care și se va afișa doar dacă activăm modul „Pass-Through Mode”. Se cunosc aceste valori și din fișa tehnică.

După ce s-au îndeplinit pașii de mai sus, putem folosi registrele senzorului care vor avea ca și scop final citirea valorilor axelor x, y și z pentru accelerometru și giroscop.

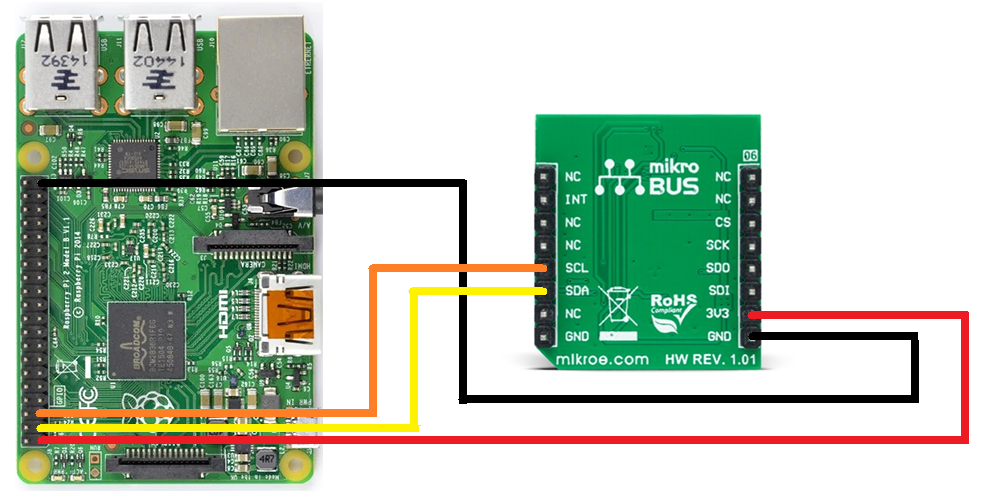


Figura 2.1 Diagrama cu cabluri dintre dispozitivul de urmărire MPU-9250 și Raspberry Pi

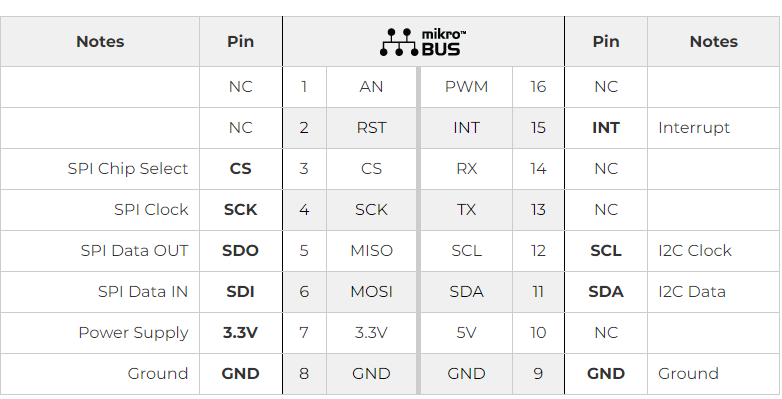


Figura 2.2 Tabel cu pinii dispozitivului MPU-9250

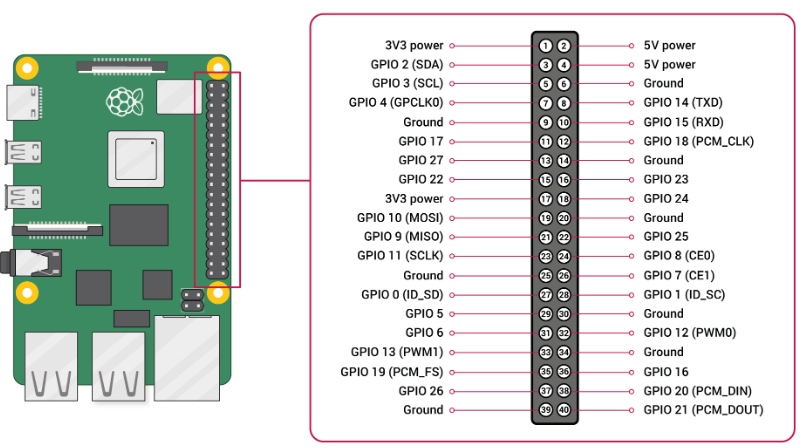


Figura 2.3 Diagrama cu pinii pentru Raspberry Pi

## ***Protocolul Secure Shell***

Protocolul SSH este folosit pentru conectarea securizată de la distanță și alte servicii de rețea securizate prin intermediul unei rețele nesigure. Acesta constă în trei componente majore:

* Protocolul stratului de transport (SSH-TRANS) este de obicei executat pe o conexiune TCP/IP, dar poate fi folosit și pe orice alt flux de date fiabil.
* Protocolul de autentificare a utilizatorului (SSH-USERAUTH) autentifică utilizatorul din partea clientului față de server. Acesta se execută pe protocolul stratului de transport.
* Protocolul de conectare (SSH-CONNECT) multiplexează canalul criptat în mai multe canale logice. Acesta rulează peste protocolul menționat anterior. Informații fac referire la (SSH Protocol Architecture RFC4251, 2006).

Pentru a se realiza conexiunea cu succes, clientul trimite o cerere după ce a fost stabilită o conexiune securizată a stratului de transport. O a doua cerere este trimisă dupa efectuarea unei autentificări complete. Acest lucru permite definirea unor noi protocoale care să coexiste cu protocoalele enumerate mai sus. Protocolul de conectare oferă canale care pot fi utilizate pentru o gamă largă de scopuri. Sunt furnizate metode standard pentru configurarea unor sesiuni shell interactive securizate și pentru redirecționarea porturilor TCP/IP. Face referire la (SSH Authentication Protocol RFC4252, 2006).

Protocolul utilizează criptarea pentru a securiza conexiunea dintre un client și un server. Toate comenzile de autentificare a utilizatorului, ieșirile și transferurile de fișiere sunt criptate pentru a proteja împotriva atacurilor în rețea. Informațiile enumerate mai sus sunt descrise de imaginea *Figura 2.4*. Face referire la (What is SSH (Secure Shell)?, 2023)

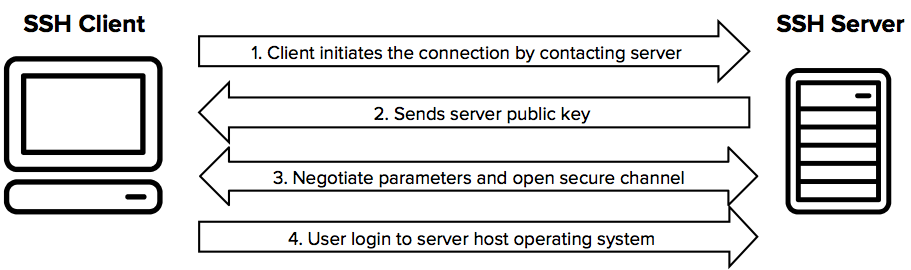


Figura 2.4 Realizarea conexiunii dintre client și server pentru protocolul Secure Shell

## ***Arhitecturi utilizate***

Așa cum am precizat și în capitolul 1 la secțiunea *1.4.1* , dispozitivul de urmărire a mișcării MPU-9250 comunică cu Raspberry Pi 2 prin intermediul interfeței seriale de comunicare I2C. Pentru comunicarea dintre laptopul cu windows și calculatorul de mici dimensiuni am folosit arhitectura client-server prin intermediul modulului socket din Python.

### *Arhitectura master-slave*

Este un model de comunicare pentru dispositive hardware în care un dispozitiv deține controlul asupra unuia/mai multor dispozitive. Componenta principală care coordonează și controlează operațiunile sistemului se numește master. Slave-ul reprezintă dispozitivul care primește instrucțiuni de la cel menționat mai sus, le execute și oferă rezultatele înapoi acestuia(vezi imaginea *Figura 2.5*. Dispozitivul master emite instrucțiuni dispozitivelor slave pentru a executa anumite acțiuni. Master-ul preia și interpretează răspunsurile primate de la acestea. Dispozitivul slave sunt conectate la master, primesc comenzi de la acesta și execută acțiuni. Slave-urile sunt pasive și nu au capacitatea de a iniția operații independent. Acestea pot fi reprezentate de senzori, module de intrare/ieșire, componente de stocare sau alte dispositive asemenea. Pentru ca cele două dispositive să comunice între ele, este necesar un mechanism de comunicare. Pentru acest lucru, am folosit comunicarea serial prin intermediul protocolului I2C (vezi explicația detaliată *aici*). În vederea unei comunicări eficiente între master și slave, folosirea cablurilor scurte și foarte scurte este obligatorie deoarece există riscuri să fie căderi de tensiune care pot influența negativ transmisiile de date.

Modelul master/slave este frecvent utilizat in industria tehnologică, atât în domeniul electronic cât și în cel mechanic. Acesta este folosit pentru a simplifica comunicarea. Spre exemplu, în loc să avem o interfață separate pentru a comunica cu fiecare unitate de disc, putem conecta majoritatea acestora printr-o singură interfată și un singur cablu. Calculatorul având posibilitatea să comunice doar cu o singură unitate care servește drept master, apoi orice date sunt transmise simplu și efficient de la master către slave.

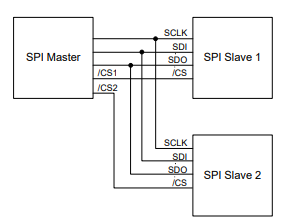


Figura 2.5 Exemplu de configurație master/slave folosind comunicarea serială SPI

### *Arhitectura client-server*

Arhitectura client-server este un model de proiectare utilizat în sistemele informatice și de rețea. Client-server este un sistem care îndeplinește atât funcția de client, cât și cea de server, astfel încât să îndeplinească schimbul de informații între ele. Acesta a devenit foarte cunoscut deoarece este utilizat în fiecare zi pentru o gamă largă de aplicații. Unele dintre protocoalele standardizate pe care clientul și serverele le folosesc pentru a comunica între ele includ protocolul de transfer fișiere FTP, protocolul de transfer de e-mail SMTP și HTTP. Sistemul client-server poate fi definit ca o arhitectură software alcătuită din client și server, în care serverul primește cereri de la client și răspunde la cererile trimise de acesta. Asigură o comunicare interprocesuală, deoarece implică schimb de date de la client și server, prin care fiecare dintre ele îndeplinește funcții diferite. (Sulyman, 2014)

Clientul este cel care solicită și utilizează resursele de la server. De obicei este o aplicație sau un program care rulează pe un dispozitiv precum calculator, laptop sau smartphone. Serverul este partea care gestionează și pune la dispoziție clientului resursele solicitate. Serverul primește solicitarea, o procesează și trimite un răspuns. Răspunsul va fi trimit înapoi la client și va folosi informația în funcție de cum a fost gândit.

Pentru a putea pune în aplicare ce am menționat mai sus, trebuie să folosim socket-uri. Un socket este o interfață de rețea, un punct final al unui flux de comunicare într-o rețea de calculatoare. Majoritatea comunicaților între calculatoare se bazează pe protocolul „Internet Protocol”. Un socket conține adresa socket-ului și date. Adresa socket-ului include adresa IP locală și adresa IP la distantă, care sunt adresele calculatorului nostru și adresa calculatorului țintă din rețea, precum și numărul portului care este utilizat pentru trimiterea și primirea datelor. (Zhang)

În vederea utilizării acestor concepte pentru socket-uri, trebuie să folosim funcțiile și metodele principale ale unui API pentru socket-uri care sunt: socket(), .bind(), .listen(), .accept(), .connect(), send(), .recv(), .close(). (realpython)

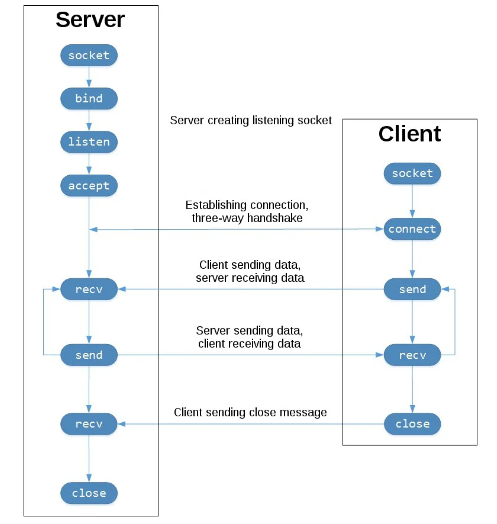


Figura 2.6 Conexiuni de socket-uri prin TCP

# **Capitolul 3. Implementarea aplicației**

Pentru a putea implementa, depana și testa aplicația propusă am folosit Visual Studio Code și PyCharm. Cele două reprezintă medii de dezvoltare integrată care facilitează editarea codului sursă, oferă o platformă puternică și flexibilă pentru programare și dezvoltare software. PyCharm este specializat în dezvoltarea de aplicații Python având o interfață ușor de utilizat și intuitivă.

Am ales să utilizez Visual Studio Code datorită popularității, diverselor extensii care ajută la creare de aplicații. Principalul motiv însă a fost existența extensiei SSH care permite accesul de la distanță și editarea de fișiere de pe alte dispozitive numite și servere. În mod evident, extensia utilizează protocolul SSH descris anterior (vezi *Protocolul Secure Shell*) care a ajutat la rularea script-urilor de la distanță. Prin intermediul acestui IDE[[15]](#footnote-15) am avut acces la fișierele de pe calculatorul de mici dimensiuni Raspberry Pi 2.

## ***Citirea valorilor de pe axele accelerometrului și giroscopului***

Înainte de a începe descrierea codului sursă care citește valorile senzorilor de pe dispozitivul MPU-9250, trebuie menționat faptul că accelerometrul și giroscopul conțin convertoare pe 16 biți care oferă ieșiri digitale cu valori corespunzătoare fiecărei axe în funcție de semnalul care a fost amplificat, demodulat și filtrat. Pentru giroscop rezultatul este proporțional cu viteza unghiulară, iar pentru accelerometru cu accelerația care s-a produs în urma mișcării pe axe. Tensiunea măsurată în volți este transformată cu ajutorul acestor convertoare analog-digitale în ieșiri digitale pentru a eșantiona fiecare axă.

În limbajul de programare Python, am folosit o clasă care conține metode pentru citirea și configurarea datelor de la senzor. Clasa este definită pentru a reprezenta o interfață de comunicare cu senzorul MPU-9250.

Constructorul are ca și parametru adresa 0x68 atunci când o vom instanția. Acesta inițializează o instanță a clasei și efectuează configurarea inițială a senzorului. Acesta este trezit din modul „sleep” prin scrierea valorii 0x00 în registrul *PWR\_MGMT\_1*. Metoda „read\_i2c\_word„ este utilă pentru citirea a două registre consecutive de la senzor folosind comunicarea I2C. Se combina cele două valori citite într-un singur cuvânt de 16 biți. ACCEL\_XOUT\_H reprezintă cei mai semnificativi 8 biți la adresa 0x3B și ACCEL\_XOUT\_L sunt cei mai nesemnificativi 8 biți la adresa 0x3C menționat si în documentație (InvenSense, 2015). Similar se întamplă și la giroscop, unde GYRO\_XOUT\_H se găsește la 0x43 și GYRO\_XOUT\_L la adresa 0x44. ACCEL\_XOUT\_L și ACCEL\_XOUT\_H registre corespunzătoare valorii citite de pe axa x a accelerometrului, ACCEL\_YOUT\_L și ACCEL\_YOUT\_H pentru axa y și ACCEL\_ZOUT\_L și ACCEL\_ZOUT\_H pentru axa Z. Metoda „read\_accel\_range” este pentru a citi intervalul de măsură al accelerometrului. Parametrul este de tip bool și daca acesta este setat pe „True”, metoda returnează valoarea brută citită din registrul ACCEL\_CONFIG. Dacă este setat pe „False„, metoda convertește valoarea într-un interval de măsură. Metoda „get\_accel\_data” este folosită pentru a citi datele de la accelerometru. În urma citirii registrelor cu metoda „read\_i2c\_word” valorile x, y și z sunt convertite în unități conform factorului de scalare corespunzător intervalului de măsură. Rezultatul este un dicționar cu valorile x, y și z. Similar, sunt metode pentru citirea valorilor de la giroscop care efectuează aceleași implementări. Combinăm două registre de un octet fiecare rezultând un cuvânt pe 16 biți. Metoda „read\_gyro\_range” are aceeași funcționalitate ca și „read\_accel\_range”, diferit fiind doar registrul de unde se citește valoarea brută acesta numindu-se GYRO\_CONFIG. Dupa ce valorile s-au citit le convertim în funcție de factorul de scalare. Metoda „get\_gyro\_data” returneaza un dicționar cu valorile vitezei unghiulare de pe fiecare axă.

## ***Obținerea și interpretarea datelor citite***

În urma comunicării dintre MPU-9250 și Raspberry Pi 2, avem posibilitatea să obținem datele ce vor fi importante în calculele ce vor determina numărul de pași și stabilirea calității somnului. Valorile astfel primite vor fi trimise clientului unde se vor efectua toate calculele. Am explicat anterior cum funcționează o arhitectura client-server (vezi *Arhitectura client-server).* În cazul implementării acesteia, vom avea un server care va fi plăcuța Raspberry Pi 2, iar clientul va fi laptopul personal. Serverul va prelua valorile accelerometrului și giroscopului de pe fiecare axă și le va trimite clientului, iar acesta le va procesa și va efectua calculele necesare.

Se vor utiliza socket-uri care vor stabili conexiunea dintre cele două. Serverul va crea un socket care va aștepta o conexiune de la client. Atunci când un client se conectează, serverul apelează .accept() pentru a accepta și finaliza conexiunea. Clientul apelează .connect() pentru a stabili conexiunea cu serverul și pentru a iniția legătura. Serverul va trimite date, clientul le va primi trimțând înapoi un răspuns. Datele sunt schimbatre întrei cei doi prin .send() și .recv(). În momentul în care clientul își inchide conexiunea, va trimite un mesaj de închidere serverului și socket-urile se vor inchide. (realpython)

În cazul proiectului, serverul va aștepta o singură conexiune de la un singur client. După ce clientul se conectează la server, acesta va trimite o data pe secundă valorile x, y și z pentru accelerometru și giroscop ca un șir de caractere (vezi *Figura 3.1*).

Mesajul trimis la client va fi de forma : „[0.1 , 0.2, 9.8, -0.1, -1.02, 3.45]”. Acesta indică o listă care este interpretată ca un șir de caractere. Fiecare element, inclusiv parantezele, reprezintă un caracter. Mesajul primit de client va trebui convertit într-o lista cu elemente de tip float. Pentru acest lucru am folosit json.loads care returnează un obiect Python.

După ce s-a realizat convertirea, este posibil să efectuăm calculele care ne vor ajuta mai departe. Dintr-o singură listă cu șase valori, am separat și am creat două liste cu trei valori fiecare. Prima listă indică valorile pentru accelerometru, iar a doua fiind pentru giroscop. Pentru a facilita și a evita crearea de erori, am creat alte trei liste aferente accelerometrului, de data aceasta fără niciun element unde se vor adăuga cele cinci seturi de valori din prima listă de mai sus. Cu alte cuvinte, întreg procesul va fi în felul următor: Primim datele de la server ca un șir de caractere. Acest șir va fi convertit și separat în două liste. O listă pentru accelerometru și una pentru giroscop. Pentru calculul numărului de pași vom utiliza doar accelerometrul. Se vor crea 3 liste goale pentru accelerometru unde vom adăuga valorile pentru cele cinci seturi pe baza cărora se vor calcula numărul de pași. Când cele 3 liste ajung la cinci elemente, se sterg și se adauga.

Odată ce conexiunea dintre client și server este stabilită, pe ecran se va deschide o fereastră unde se vor găsi două grafice cu valorile x, y și z pentru accelerometru și giroscop. Acesta se va actualiza dupa fiecare cinci seturi de valori trimise de la server la client. Se va găsi și numărul de pași și rezultatul stabilirii calității somnului. În următorul sub-capitol se va detalia cum funcționează acest lucru.



Figura 3.1 Implementarea serverului

## ***Algoritmi propuși***

### *Detecție pași*

Pentru detecția pașilor vom folosi cele trei liste menționate anterior. xAccel va fi lista în care se vor regăsi valorile pentru axa x a accelerometrului, yAccel va fi pentru axa y și zAccel va fi pentru axa z unde fiecare listă va conține cinci elemente. Vom utiliza norma unui vector tridimensional reprezentat de componente într-un set de coordonate. Formula arată în felul următor:

|  |  |
| --- | --- |
|  | () |

În funcție de câtă mișcare detectează senzorul pe axe, rezultatul ar trebui să fie în jurul valorii de 9.8, 9.9, 10 ,11 sau chiar mai mult. Vom folosi alte două liste în care vom stoca rezultatul normei pentru fiecare set de valori și diferența dintre rezultatul actual al normei și cel anterior. Am setat un prag de 1.2 care a fost descoperit experimental ca fiind o valoare optimă în acest caz.

Înainte de a explica cum se detectează efectuarea unui pas, trebuie să amintim cum detectăm existența mișcării. Altfel spus, dacă senzorul a fost mișcat sau dacă valorile accelerometrului s-au modificat în timp. Pentru a verifica acest lucru, identificăm axele care depașesc valoarea de 10. Aceasta nu este aleasă la întamplare, deoarece dacă dispozitivul este plasat pe o suprafață plană dispozitivul de urmărire a mișcării va măsura pe axele x și y o valoare aproximativ egală cu 0g și 1g pe axa x. Acest lucru este menționat la (InvenSense, 2016). 1g reprezintă accelerația gravitațională la suprafața pământului. În general, aceasta variază între 9.78 și 9.82 în funcție de pozița pe glob. În proiect, am luat în calcul ca fiind 1g = 9.80665 m/s2. Am considerat că orice valoare în modul care măsoară mai mult de 10 pe oricare axă înseamnă că s-a produs o mișcare. Pentru a nu exista confuzii, am luat în calcul și următorul scenariu: Ținem senzorul într-o anumită poziție nemișcată diferită de una din pozițiile ideale în care una din valorile axelor este și restul sunt 0. Există posibilitatea în care să ținem senzorul într-o poziție în care să fie îndeplinită și valoarea |xAccel| > 10 sau |yAccel| >10 sau |zAccel| > 10, dar și rezultatul normei celor trei componente să fie mai mare de pragul 1.2 menționat mai sus. Dar cu toate acestea, senzorul pentru o perioadă scurtă de timp să nu fi fost în mișcare. În altă oridine de idei, am considerat că accelerația actuală în modul de pe fiecare axă să fie diferită față de cea anterioară. Cu aceasta, se evită detectarea mișcării în mod eronat, dar poate fi utilă și în cazul când se produce o simplă mișcare bruscă ce nu înseamnă neaparat un pas efectuat. Dacă acestea sunt îndeplinite setăm variabila *move* pe „True” , în caz contrar pe „False”.

Așadar, dacă ambele cazuri sunt îndeplinite, incrementăm variabila *steps* cu 1 și considerăm ca s-a efectuat un pas.

### *Determinarea calitatății somnului*

În mod evident, un somn liniștit și de bună calitate înseamnă că pe parcursul nopții să ai bătăi regulate ale inimii, un nivel de oxigen în sânge de peste 90% și trezirile bruște din timpul somnului să fie momente episodice. Nu voi aborda cele menționate anterior, dar voi insista pe partea unde există mișcare și consider că este un factor important în stabilirea calității somnului. În timp ce dormim, ar trebui să ne mișcăm cât mai puțin, și în mod ideal să stăm în aceeași poziție.

Pentru acest lucru, voi calcula dispersia pentru valorile x, y și z pentru giroscop și accelerometru. Dupa o extragere de 60 de valori pentru ambii senzori, vom calcula cât de împraștiate sunt acestea în jurul valorii medii. Cu rezultatul dispersiilor se va face media aritmetică, valoarea obținută se va adăuga într-o listă cu maxim zece elemente care va fi considerată o variabilă ce stochează și mediile aritmetice anterioare. Dacă maximul valorii unui element din listă va fi mai mare decât pragul cu valoarea 5 în cazul giroscopului și pragul cu valoarea 6.5 în cazul accelerometrului se consideră un somn în care s-au produs mișcări ceea ce poate indica un somn agitat. Până când lista nu va atinge numărul maxim de valori pe care l-am setat ca fiind 10 nu se va putea stabili acest aspect.Cu cât mai multe valori de analizat, cu atât mai precis modul de apreciere. Am stabilit ca fiind satisfăcător 10 valori. Condiția este că după 60 de valori parcurse având calculate dispersia și efectuată media aritmetică pe baza lor putem trage o concluzie dacă senzorul a măsurat valori cu diferențe mari între ele care ar determina o serie de mișcări. Dacă lista are 10 elemente și ultimul element introdus a fost o medie ce depășește pragul impus, somnul va fi considerat agitat până când se suprascriu valorile existente și a 10-a valoare din setul anterior va fi suprascrisă cu o valoare din noul set care este sub acel prag. Există două liste asemenea cu lista descrisă anterior pentru ambii senzori care funcționează în acest fel care stochează media dispersiilor de la valorile pe axele accelerometrului și giroscopului.

### *Testarea aplicației*

În această secțiune, voi arăta și detalia scenariile care se pot regăsi pe parcursul execuției aplicației. Vom scoate în evidență rezultatele obținute în urma măsurării valorilor de pe senzori și vom discuta pe baza acestora. Testarea aplicației este o etapă esențială în dezvoltarea și livrarea unei soluții fiabile și eficiente. Scopul acesteia este de a identifica erori sau deficiențe din timpul rulării programului și de a le corecta.

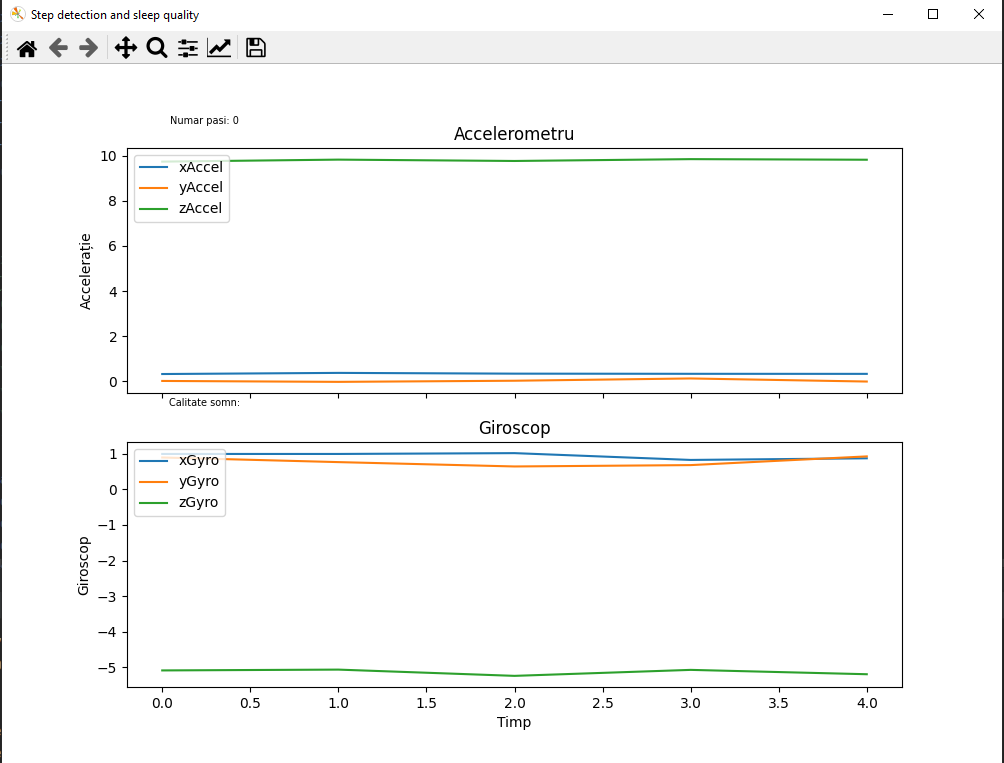


Figura 3.2 Graficele afișate la începutul execuției

Imaginea *Figura 3.2* reprezintă începutul execuției programului în care nu se produc mișcări senzorului el fiind pe o suprafață plană. Se observă faptul că accelerația axa z are valoare aproximativ egală cu accelerația gravitațională, adică 9.8 și accelerațiile pe axele x și y fiind aproximativ 0. Valorile măsurate de MPU-9250 pentru axele de la giroscop sunt valori normale având în vedere că senzorul este așezat pe o suprafață plană, altfel spus pe planul Oz în acest caz. Numărul de pași este 0 deoarece nu s-au îndeplinit niciuna din condițiile care să detecteze un pas.

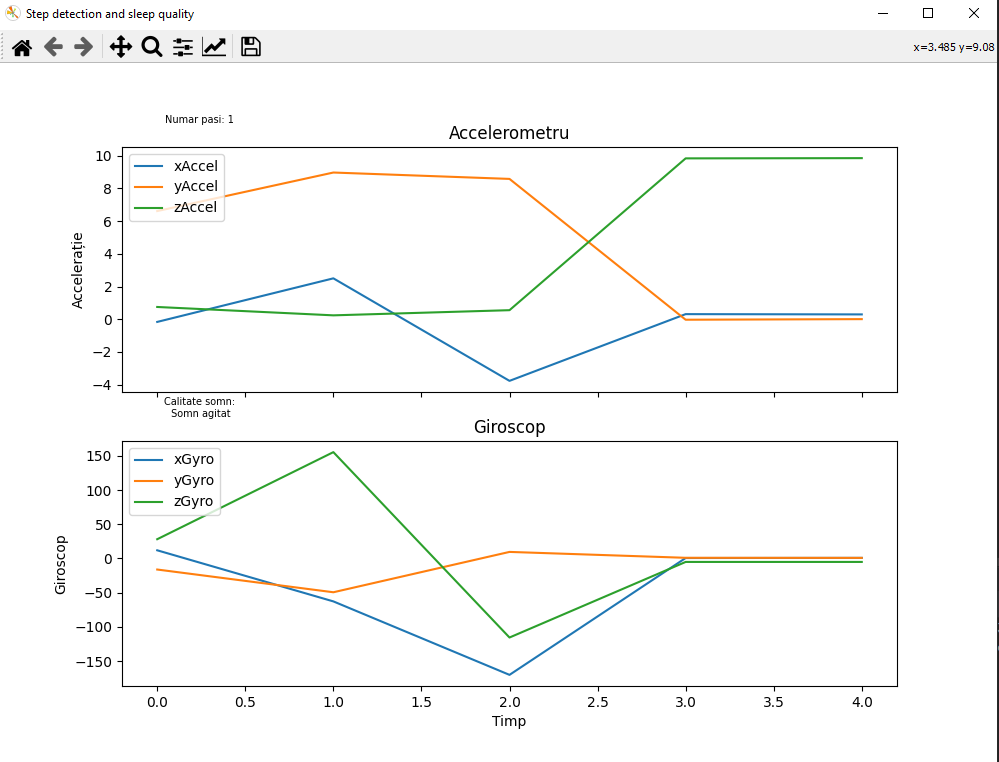


Figura 3.3 Modificarea valorilor în momentul mișării senzorului

Imaginea *Figura 3.3* indică faptul că s-a produs o serie de mișcări care a concluzionat cu efectuarea unui pas. Se observă faptul că acest lucru nu a fost de lungă durat și mișcarea s-a oprit. Somnul va fi catalogat drept somn agitat până când următoarele seturi de valori vor fi interpretate ca fiind neconcludente. Așa cum am explicat și anterior( vezi *Determinarea calitatății somnului* ), cele două liste cu mediile aritmetice ale dispersiei determinate din setul de 60 de valori pentru fiecare senzor să aibă maxim 10 elemente. După mai multe rotiri și mișcări aduse dispozitivului de mișcare MPU-9250, l-am lăsat în aceeași poziție pentru o perioadă de timp și conform așteptărilor somnul este declarat liniștit deoarece nu s-au măsurat valori ale axelor accelerometrului și giroscopului și numărul de pași a rămas același.

Durata în secunde din momentul achiziției de date, trimiterea la server, procesarea lor de către client și afișarea rezultatelor pe grafic este de 5 secunde. Acest lucru înseamnă că o dată la 5 secunde graficul generat se va actualiza. Altfel spus există o întârziere de 5 secunde și graficul va afișa acțiunea care s-a întamplat în anterior nu în momentul efectuării operațiunii. Imaginile de mai jos descriu acest lucru.

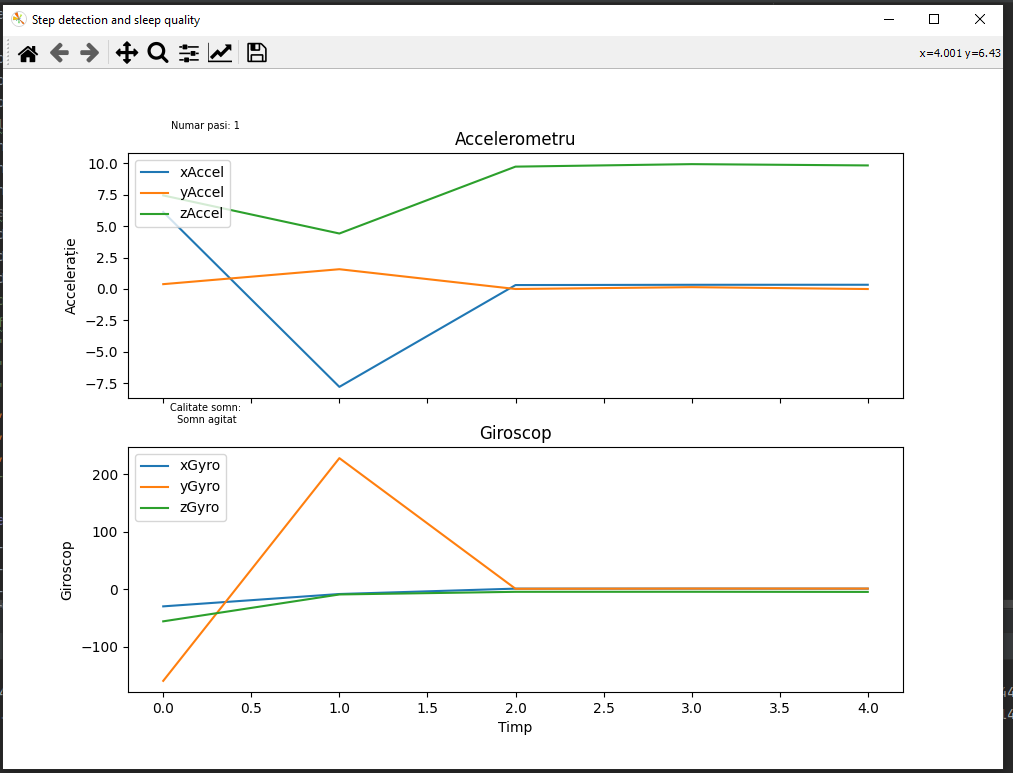


Figura 3.4 Imagine cu graficele când mișcarea este aproape de a se termina

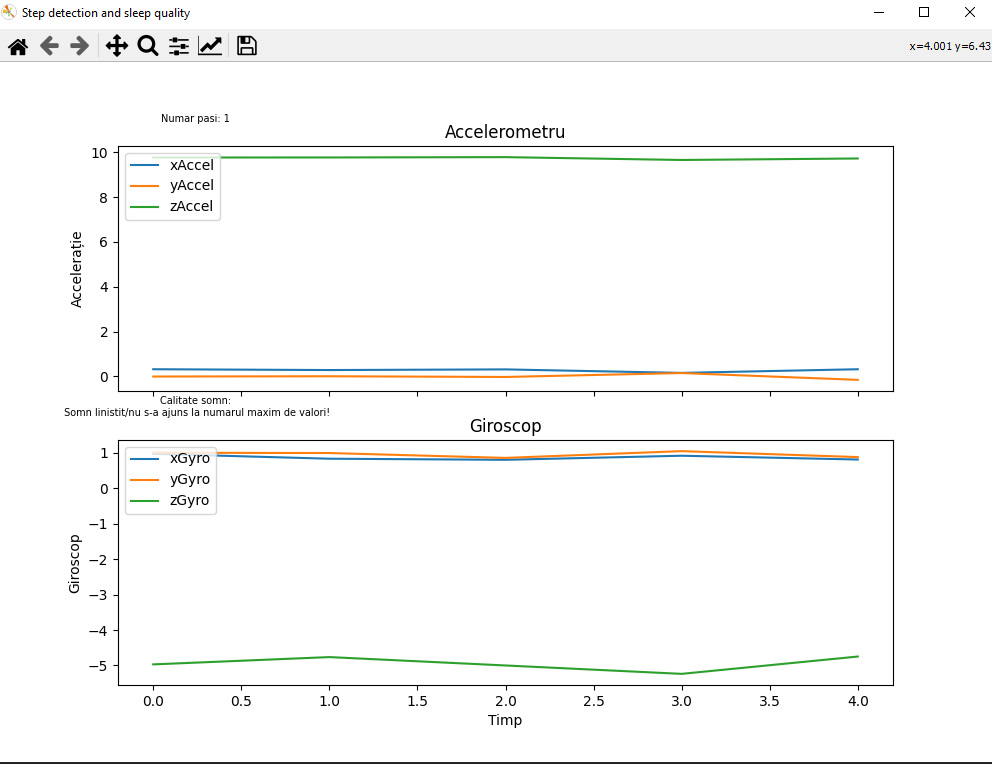


Figura 3.5 Imagine cu graficele când mișcarea s-a oprit de tot

# **Capitolul 4. Utilizarea proiectului**

## ***Funcționalitate***

## 

Capacitatea de a detecta lipsa sau activitatea fizică minimă a devenit un factor cheie în îngrijirea persoanelor în vârstă sau a celor după spitalizare, pentru care neregularitatea sau scăderea activității zilnice este primul simptom al unei sănătăți care se deteriorează. În plus, progresul tehnologic în continuă expansiune a condus la creșterea numărului de servicii oferite prin intermediul calculatoarelor, limitând obiceiurile noastre de activitate zilnică și determinând creșterea numărului de persoane care suferă de afecțiuni ale sistemului vascular de la an la an. Un dispozitiv care să urmărească și să măsoare numărul de pași ar avea un impact important înainte de a vizita un medic. Frecvente sunt cazurile în care spunem că avem dureri musculare sau simțim dureri în diferite zone ale corpului și nu știm de ce. Acest lucru poate fi o dovadă a sedentarismului. În medie, o persoană ar trebui să facă zilnic în jur de 6000 de pași pentru a nu fi considerat un om sedentar. Ideea din spatele acestui număr de pași se află sănătatea noastră. O persoană inactivă care nu reușește să atingă pragul de 5000-6000 pași zilnic va avea tendința să se îngrașe, se va confrunta cu stări de oboseală și o predispunere prematură la boli cardiace.

Funcționalitatea proiectului este intuitivă. Pe baza variațiilor vitezelor unghiulare măsurate de giroscop putem stabilim care este calitatea somnului și dacă s-au produse rotiri care să concludă faptul că somnul a fost liniștit sau agitat. Dacă cele trei valori de pe axe diferă cu mult una față de cealaltă, înseamnă că s-au măsurat viteze unghiulare semnificative. Pe baza valorilor citite de pe axele accelerometrului, se vor efectua calcule care să scoată în evidență faptul că dacă a existat mișcare pe una dintre axe mai mare decât accelerația gravitațională spunem că senzorul și-a schimbat poziția în spațiu. Dacă și modulul vectorului actual este mai mare și diferit de cel anterior, putem spune că s-a detectat un pas.

Proiectul actual ar putea deveni o bază solidă pentru un proiect de mari dimensiuni unde poate fi folosit într-un mediu controlat , cum ar fi un spital. Medicii ar fi mai eficienți în luarea deciziilor în privința pacienților. I-ar putea supraveghea mai ușor în spital pe pacienții care au ieșit dintr-o operație dacă pacienții respectivi ar avea la mână sau la picior un astfel de dispozitiv de mișcare care în urma unor măsurători să tragă concluzii. Ar putea exista centralizări cu tipologii de oameni care încep să se miște mai repede după o anumită operație sau intervenție chirurghicală. Dispozitive care vor fi purtate de oamenii din spitale doar pe parcursul șederii lor în spital. Aceste lucruri sunt de părere că ar ajuta la prevenirea și avertizarea viitorilor pacienți care doresc să facă intervenții chirurghicale și să știe ce se va întampla dupa operație. Și medicii vor fi mai eficienți cu astfel de dispositive care le vor ușura meseria.

## ***Îmbunătățiri ulterioare***

În ceea ce privește număratul de pași, există o metodă care se aplică smartwatch-urilor și folosește exclusiv doar variația câmpului magnetic. Aceasta utilizează mișcarea naturală a brațului în timpul mersului ca abordare de început. Brațul va oscila defazat în timpul mersului, adică brațul stâng se mișcă înainte și invers pentru piciorul și brațul opus. Această mișcare a brațului poate fi considerată de tip pendul cu poziții maxime înainte și înapoi. Mișcarea senzorului între max-f și max-b în câmpul magnetic al Pământului determină o modificare a componentelor câmpului magnetic. Senzorul măsoară modificarea câmpului magnetic între Bmax și Bmin. Bmax corespunde mișcării maxime înainte și înapoi a brațului, iar Bmin corespunde poziției perpendiculare a brațului față de sol. Datorită relației defazate dintre mișcarea brațului și a piciorului, variația periodică măsurată a câmpului magnetic poate fi legată de mișcarea piciorului, adică de pași. Informațiile din acest paragraf au referință la : (Wisniowski, 2021)

Pentru o monitorizare precisă și exactă a somnului, ar fi indicat de adăugat pentru o extindere a proiect, un senzor care să măsoare nivelul de oxigen din sânge și un senzor de presiune care să monitorizeze bătăile inimii. Cu senzorul de presiune, putem stabili exact cum respiră persoana purtătoare a dispozitivului și prin intermediul ritmului cardiac avem posibilitatea să precizăm în ce stare a somnului se află individul. Accelerometru și giroscopul determină activitatea din timpul somnului, existența mișcărilor de rotație și schimbările pozițiilor pe parcursul nopții. Ritmul cardiac și ritmul respirator sunt estimate prin detectarea datelor obținute de la un senzor de presiune. Stadiul de somn este clasificat în funcție de activitatea mișcării și de ritmul cardiac.În cele din urmă, calitatea somnului este calculată complet prin utilizarea poziției de somn, a prezenței/absenței apneei de somn și perioada în care individul doarme. Acest paragraf are referințe la (Yunyoung Nam, 2016).

# **Concluzie**

Tehnologia din zilele noastre are un rol semnificativ în monitorizarea și îmbunătățirea sănătății noastre. Dispozitivele moderne echipate cu astfel de senzori care au capacitatea să măsoare numărul de pași cum ar fi telefonul mobil smartphone și să emită rapoarte despre calitatea somnului cum ar fi ceasurile inteligente cunoscute ca și smartwatch-uri, au devenit foarte populare de-a lungul timpului. Este important să înțelegem faptul că acestea au un rol decisiv în luarea unor decizii care ar putea elimina anumite obiceiuri nesănătoase. Cel mai nesănătos obicei este statul mult într-un singur loc pentru o perioadă lungă de timp. Consider că majoritatea oamenilor ce dețin un smartphone în zilele noastre își iau telefonul asupra lor peste tot. Înclusiv când stăm fără să facem mișcare. Cu o simplă atingere pe aplicația care stochează informațiile despre analiza zilnică, observăm istoricul cu numărul de pași pe zi. Dacă am vedea că media pașilor efectuați în ultimele două luni ar fi sub 2000 de pași zilnic, ar trebui să ne punem un semn de întrebare. Acest fapt ar putea însemna că suntem sedentari și în viitorul apropiat ne putem aștepta la dureri musculare, osoase și de coloană vertebrală în zona cervicală sau lombară. O simplă informare poate trezi îngrijorări care duc la rezolvarea problemei din timp.

În ceea ce privește proiectul ales pentru lucrearea de diplomă, consider că mai există loc de îmbunătățiri pe care le-am detaliat și în *4.2* care pot completa implementarea propusă. Acesta este într-o stare functională și conform așteptărilor. Proiectul s-a axat pe monitorizarea activităților adică pe achiziția și prelucrarea datelor accelerometrului și giroscopului. Pentru a avea o imagine completă asupra calității somnului se poate adăuga la proiectul prezentat și un senzor de presiune care ajuta în determinarea ritmului cardiac și ritmului respirator. Acest lucru este util de utilizat pentru a verifica dacă individul este în stare de somn.

Pe parcursul proiectării și implementării proiectului, am aprofundat informații pe care le dețineam deja, dar am și descoperit căi necunoscute. M-a intrigat faptul nu m-am confruntat cu aceste caracteristici mai în profunzime pe care astăzi orice telefon mobil le deține. Mă informam zilnic despre câți pași am efectuat într-o anumită zi, dar nu m-am gândit mai departe și cum ar funcționa o astfel de aplicație. Cu această ocazie, am avut posibilitatea să mă documentez și să mă informez despre componentele necesare și cum ar trebui să implementez acest lucru. Până să încep proiectul nu am mai interacționat cu astfel de dispozitive de mișcare precum este MPU-9250. Acest lucru a fost o oportunitate să îmi descopăr și alte abilitățile care m-au ajutat să înțeleg cum ar funcționa și alte dispozitive asemănătoare. Cu plăcuțe Raspberry Pi am mai lucrat și pe parcursul celor patru ani de studii și nu a fost dificil să mă acomodez cu astfel de calculator.

În concluzie, am convingerea că dezvoltarea unei aplicații de acest tip îmi va fi de folos în formarea mea profesională și mă va ajuta în asimilarea de noi informații mai ușor. Monitorizarea activităților în urma măsurării datelor de la accelerometru și giroscop a fost o provocare care m-a făcut să înțeleg utilitatea dezvoltării de proiecte hardware și software în societate.

# **Bibliografie**

[1] „realpython” [Online] Disponibil la adresa: https://realpython.com/python-sockets/

[2] InvenSense. (2015, Iulie 1). *MPU-9250.* [Online] Disponibil la adresa Register Map and Descriptions Revision 1.6: https://invensense.wpenginepowered.com/wp-content/uploads/2017/11/RM-MPU-9250A-00-v1.6.pdf

[3] InvenSense. (2016, 06 20). *MPU-9250.* [Online] Disponibil la adresa Product Specification: https://download.mikroe.com/documents/datasheets/PS-MPU-9250A-01-v1.1.pdf

[4] *SSH Authentication Protocol RFC4252*. (2006, Ianuarie 1). [Online] Disponibil la adresa RFC4252: https://datatracker.ietf.org/doc/html/rfc4252

[5] *SSH Protocol Architecture RFC4251*. (2006, Ianuarie 1). [Online] Disponibil la adresa RFC4251: https://datatracker.ietf.org/doc/html/rfc4251#page-4

[6] Sulyman, S. H. (2014, Ianuarie). Journal of Computer Engineering. [Online] Disponibil la adresa: https://www.researchgate.net/profile/Shakirat-Sulyman/publication/271295146\_Client-Server\_Model/links/5864e11308ae8fce490c1b01/Client-Server-Model.pdf

[7] *What is SSH (Secure Shell)?* (2023, Ianuarie 01). [Online] Disponibil la adresa: https://www.ssh.com/academy/ssh

[8] Wisniowski, P. Ł. (2021, Noiembrie 23). *Method of Step Detection and Counting Based on Measurements of Magnetic Field Variations.* [Online] Disponibil la adresa mdpi.com: https://www.mdpi.com/1424-8220/21/23/7775

[9] Yunyoung Nam, Y. K. (2016, Mai 23). *Sleep Monitoring Based on a Tri-Axial Accelerometer and a Pressure Sensor.* Preluat de pe ncbi.nlm.nih.gov/pmc/articles: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4883440/

[10] Zhang, H. (fără an). *Architecture of Network and Client-Server model.* [Online] Disponibil la adresa https://arxiv.org/ftp/arxiv/papers/1307/1307.6665.pdf

1. QFN este un pachet de mici dimensiuni fără plumb, care este dreptunghiular cu rol în disiparea căldurii. [↑](#footnote-ref-1)
2. AKM este un producător de semiconductori cu sediul in Tokyo, Japonia. [↑](#footnote-ref-2)
3. Inter-Integrated Circuit, cunoscut ca și I2C este o magistrală de comunicație seriala sincronă. [↑](#footnote-ref-3)
4. DPS – radiani/secunda, g – accelerația gravitațională și µT – microTesla. [↑](#footnote-ref-4)
5. Serial Peripheral Interface [↑](#footnote-ref-5)
6. Inter-Integrated Circuit [↑](#footnote-ref-6)
7. Master-ul controlează întregul sistem, inițiază și gestionează datele primite de la slave. [↑](#footnote-ref-7)
8. Face referire la arhitectura master-slave, în care slave-ul reprezintă nodul care execută sarcinile primite de la master furnizându-i înapoi diferite rezultate ale procesării sale. [↑](#footnote-ref-8)
9. DLPF – Digital Low Pass Filter face referire la un filtru digital trece-jos fiind folosit în prelucrarea semnalelor de la senzori. Acesta este utilizat în vederea eliminării/reducerii zgomotelor sau semnalele cu frecvență înaltă care pot interfera cu cele de joasă frecvență. [↑](#footnote-ref-9)
10. Fchoice – câmp de biți din registrul de configurare DLPF care indică modul de funcționare al filrului trece-jos. [↑](#footnote-ref-10)
11. EXT\_SYNC\_SET[2:0] – câmp de biți utilizat pentru setarea sursei externe de sincronizare a circuitului integrat. [↑](#footnote-ref-11)
12. PD\_PTAT – Power-Down Proportional to Absolute Temperature se referă la un circuit senzor care generează o tensiune proporționala cu temperatura absolută. [↑](#footnote-ref-12)
13. Phase-Look Loop este un circuit electronic de control utilizat în pricipal pentru a genera semnal de ieșire cu o frecvență stabilită. [↑](#footnote-ref-13)
14. Kernel-ul reprezintă nucleul sistemului de operare care are acces la resursele hardware și software fiind o componentă fundamentală a sistemului. [↑](#footnote-ref-14)
15. Integrated development enviornment - o aplicație software care oferă instrumente în dezvoltarea de soft. [↑](#footnote-ref-15)