Vilcu Elena Roxana

# IMBUNATATIREA SI OPTIMIZAREA UNUI BRAT ROBOTIC CONTOLAT PRIN AI PENTRU MANIPULAREA PIESELOR DE SAH

Cuprins

[1. Introducere 1](#_Toc160527836)

[1.1 Scopul documentului 1](#_Toc160527837)

[2. Prezentare generală și abordări de proiectare 2](#_Toc160527838)

[2.1 Prezentare generală 2](#_Toc160527839)

[2.2 Presupuneri/ Constrângeri/ Riscuri 2](#_Toc160527840)

[2.2.1 Presupuneri 2](#_Toc160527841)

[2.2.2 Constrângeri 2](#_Toc160527842)

[2.2.3 Riscuri 3](#_Toc160527843)

[3. Considerațiii de proiectare 4](#_Toc160527844)

[3.1 Obiective și linii directoare (ghiduri) 4](#_Toc160527845)

[3.2 Metode de dezvoltare 4](#_Toc160527846)

[3.3 Strategii de arhitectură 4](#_Toc160527847)

[4. Arhitectura Sistemului și Proiectarea Arhitecturii 6](#_Toc160527848)

[4.1 Vedere logică 6](#_Toc160527849)

[4.2 Arhitectură hardware 6](#_Toc160527850)

[4.3 Arhitectură software 6](#_Toc160527851)

[4.4 Arhitectura informațiilor 7](#_Toc160527852)

[4.5 Arhitectura de comunicații interne 7](#_Toc160527853)

[4.6 Diagrama de arhitectură a sistemului 8](#_Toc160527854)

[5. Proiectarea sistemului 9](#_Toc160527855)

[5.1 Conversii de date 9](#_Toc160527859)

[5.2 Interfețe utilizator 10](#_Toc160527860)

[5.3 Proiectarea interfețelor cu utilizatorul 10](#_Toc160527863)

[6. Scenarii de utilizare 11](#_Toc160527864)

[7. Proiectare de detaliu 12](#_Toc160527865)

[7.1 Proiectare hardware de detaliu 12](#_Toc160527866)

[7.2 Proiectare software de deatliu 12](#_Toc160527867)

[7.3 Proiectare de detaliu pentru performanța sistemului 14](#_Toc160527869)

[7.4 Proiectare detaliată a comunicațiilor interne (între componente) 14](#_Toc160527870)

[8. Controale pentru verificarea integrității sistemului 15](#_Toc160527871)

[Anexa A: Gestiunea modificărilor documentului 16](#_Toc160527872)

[Anexa B: Acronime 17](#_Toc160527873)

## Introducere

Acest document descrie în detaliu specificațiile de proiectare pentru sistemul "Îmbunătățirea și optimizarea unui braț robotic controlat prin AI pentru manipularea pieselor de șah". Proiectul integrează o combinație de componente hardware și software pentru a automatiza desfășurarea unui joc de șah între un utilizator uman și un sistem controlat de inteligență artificială. Documentul definește arhitectura generală a sistemului, proiectarea componentelor hardware și software, precum și considerațiile de securitate, performanță și comunicații.

### Scopul documentului

Scopul acestui document este de a descrie modul în care cerințele funcționale și non-funcționale definite în faza anterioară de analiză sunt transpuse în specificații tehnice concrete pentru implementarea sistemului. Documentul servește drept ghid pentru echipa de dezvoltare, facilitând construcția și integrarea tuturor componentelor necesare pentru funcționarea completă a sistemului robotic dedicat jocului de șah.

## Prezentare generală și abordări de proiectare

### Prezentare generală

Sistemul constă într-o platformă hardware formată din Arduino Mega, multiplexoare CD74HC4067, senzori reed pentru detectarea poziției pieselor de șah, driver PCA9685 pentru controlul servomotoarelor și brațul robotic pentru mutarea pieselor. Partea software integrează Arduino IDE pentru programarea plăcii și Python pentru logica de joc și deciziile AI.

Datele colectate de la senzorii reed sunt transmise prin port serial către aplicația Python. AI-ul calculează următoarea mutare, iar comenzile sunt trimise înapoi către Arduino, care controlează brațul robotic.

### Presupuneri/ Constrângeri/ Riscuri

#### Presupuneri

Utilizatorul cunoaște regulile jocului de șah. Toate componentele hardware vor fi alimentate și conectate corect.

#### Constrângeri

**Mediu hardware sisoftware**  
Sistemul funcționează cu o placă Arduino Mega, care are un număr limitat de pini de intrare, motiv pentru care este necesară utilizarea multiplexoarelor pentru a conecta toți cei 64 de senzori reed. Pe partea software, Arduino IDE și Python sunt utilizate, necesitând configurare corectă pentru o comunicare stabilă.

**Disponibilitatea sau volatilitatea resurselor**  
Sistemul depinde de alimentarea stabilă a componentelor hardware. Întreruperile de curent sau variațiile de tensiune pot opri funcționarea corectă a sistemului.

**Conformitatea cu standardele**  
Sistemul folosește comunicație serială USB standard pentru transmiterea datelor, respectând protocoalele de comunicare obișnuite pentru dispozitive embedded.

**Cerințe de interoperabilitate**  
Arduino și aplicația Python comunică între ele printr-un protocol simplificat pe conexiune serială. Sistemul nu a fost proiectat pentru integrarea cu alte platforme externe.

**Cerințe de interfață/protocol**  
Protocolul de comunicare este simplu, bazat pe trimiterea secvențială de date prin serial USB. Structura datelor trebuie menținută clară pentru a evita erorile de interpretare între Arduino și Python.

**Cerințe pentru depozitarea și distribuția datelor**  
Sistemul nu necesită stocare permanentă de date. Toate informațiile sunt procesate în timp real și nu sunt salvate după finalizarea fiecărei mutări.

**Cerințe de securitate**  
Nu sunt necesare măsuri speciale de criptare a datelor, deoarece comunicația se face local și nu este expusă internetului sau rețelelor externe.

**Limitări ale capacității de memorie sau ale altor resurse**  
Arduino Mega are o memorie limitată, ceea ce impune o gestionare atentă a datelor colectate de la senzori și a comenzilor de control pentru servomotoare.

**Cerințe de performanță**  
Sistemul trebuie să răspundă prompt la fiecare acțiune a utilizatorului. De la apăsarea butonului până la finalizarea mutării AI, timpul total nu trebuie să depășească șapte secunde.

**Cerințe de verificare și validare (testare)**  
Sistemul necesită verificări manuale pentru confirmarea funcționării corecte. Testarea include verificarea detecției senzorilor și a preciziei mișcărilor brațului robotic.

#### Riscuri

Interferențele electromagnetice pot afecta citirea senzorilor reed. Blocajele mecanice ale brațului robotic pot întrerupe fluxul normal al jocului. Lipsa unui sistem de alimentare redundant poate duce la oprirea completă a sistemului.

## Considerații de proiectare

### Obiective și linii directoare (ghiduri)

Obiectivul principal este realizarea unui sistem stabil și precis, capabil să detecteze corect poziția pieselor și să efectueze mutările calculate de AI. Software-ul trebuie să fie bine organizat și comentat, pentru a permite ușor modificări sau îmbunătățiri în viitor. Este important ca toate componentele hardware să fie montate și testate cu atenție pentru funcționare corectă.

### Metode de dezvoltare

Pentru acest proiect s-a folosit o metodă de dezvoltare de tip **structurat**, având în vedere simplitatea sistemului și scopul educațional al aplicației. Hardware-ul și software-ul au fost dezvoltate separat, apoi integrate și testate împreună.

Codul pentru Arduino a fost scris procedural, cu funcții dedicate pentru fiecare sarcină: inițializare, citirea senzorilor și controlul servo. Aplicația Python a fost construită modular, separând partea de comunicare serială de partea de procesare AI.

Nu a fost necesară utilizarea unor metode complexe de tip UML sau design orientat pe obiecte, deoarece sistemul nu implică o arhitectură complicată.

### Strategii de arhitectură

Sistemul a fost gândit să fie simplu, clar și ușor de pus în practică, având în vedere scopul educațional și aplicativ.  
Deciziile de design luate pentru organizarea generală a sistemului se bazează pe separarea clară a responsabilităților între componentele hardware și software.

1. **Utilizarea unui anumit tip de produs**  
   Pentru partea hardware s-a ales Arduino Mega datorită numărului mare de pini și a compatibilității cu multiplexoarele și driverul PCA9685.  
   Pentru software, s-a folosit Arduino IDE pentru programarea microcontrolerului și Python pentru integrarea modelului AI și controlul comunicării cu Arduino. Am folosit biblioteci standard și ușor de folosit, evitând soluțiile comerciale sau complexe, pentru a menține proiectul simplu și accesibil.
2. **Planuri viitoare de extindere**  
   Sistemul este conceput astfel încât să poată fi extins cu ușurință. În viitor se pot adăuga o interfață grafică pentru utilizator sau chiar salvarea mutărilor într-o bază de date. Comunicarea USB permite și integrarea cu alte aplicații, dacă va fi nevoie.
3. **Paradigmele interfeței hardware și software**  
   Sistemul folosește o comunicare serială USB simplificată, cu format de date clar între Arduino și aplicația Python. Hardware-ul este organizat liniar: senzori > Arduino > Python > Arduino > Braț robotic.
4. **Politicile de gestionare a memoriei**  
   Arduino Mega are resurse limitate, de aceea codul a fost organizat astfel încât să consume minim de memorie. Matricea senzorilor este procesată rapid și nu sunt folosite structuri complexe de date.
5. **Managementul altor resurse**  
   Toate componentele hardware sunt alimentate printr-o sursă stabilă de curent, fără resurse distribuite sau externe.

## Arhitectura Sistemului și Proiectarea Arhitecturii

### Vedere logică

Sistemul urmează un flux simplu: citirea datelor de la senzori, procesarea deciziei prin AI și executarea mutării de către brațul robotic. Fiecare pas se desfășoară într-o ordine clară, asigurând o bună funcționare.

### Arhitectură hardware

Sistemul este centralizat, toate componentele fiind conectate la un calculator local.

Componente hardware principale:

* Arduino Mega 2560 — Microcontrolerul principal, gestionează citirea senzorilor și controlează brațul robotic.
* 4 multiplexoare CD74HC4067 — Extind capacitatea de citire pentru cei 64 de senzori reed.
* 64 senzori reed — Detectează prezența pieselor pe tabla de șah.
* Driver PCA9685 — Controlează cele șase servomotoare ale brațului robotic.
* Braț robotic cu 6 servomotoare — Mută piesele conform comenzilor AI.
* Buton de comandă — Inițiază citirea poziției pieselor.
* Buzzer — Anunță utilizatorul când este rândul său.
* Calculator local — Rulează aplicația Python și modelul AI. Configurație minimă: procesor Intel i3, 4 GB RAM, Windows 10.

Conectivitate:

* Arduino comunică cu calculatorul prin USB (serial).
* Senzorii reed sunt conectați prin multiplexoare la Arduino.
* Driverul PCA9685 este legat la Arduino prin magistrala I2C.
* Calculatorul primește datele de la Arduino și trimite înapoi comenzi pentru brațul robotic.

Estimări resurse:

* Arduino folosește sub 50% din capacitate.
* Calculatorul este folosit minim, aplicația Python și AI-ul nu solicită mult procesorul sau memoria.
* Nu este nevoie de stocare mare, datele sunt procesate în timp real.

Rețea:

Sistemul nu folosește rețea. Totul se face local, fără routere sau switch-uri.

### Arhitectură software

Sistemul software este alcătuit din două componente principale: codul rulat pe Arduino și aplicația Python rulată pe calculatorul local. Nu există baze de date sau sisteme software externe complicate, totul este gândit simplu pentru funcționare locală.

Componente software:

1. Arduino IDE   
   Folosit pentru scrierea și încărcarea codului în placa Arduino Mega. Codul este scris în C/C++ și gestionează citirea senzorilor, comunicarea serială și controlul brațului robotic prin driverul PCA9685.
2. Python

Rulează aplicația principală pe calculatorul local. Se ocupă de primirea datelor de la Arduino, procesarea pozițiilor pieselor și calculul mutării prin AI. Trimite comenzile de control înapoi către Arduino prin port serial USB.

### Arhitectura informațiilor

Sistemul procesează informații despre pozițiile pieselor pe tabla de șah, transformate într-o matrice logică, care este analizată de AI pentru a determina mutarea optimă. Informațiile se păstrează în memorie temporar, doar pe durata fiecărei mutări.

### Arhitectura de comunicații interne

Arduino și aplicația Python comunică printr-o conexiune serială USB. Informațiile de la senzori sunt trimise la Python, iar comenzile de mutare sunt trimise înapoi către Arduino, asigurând astfel fluxul complet de joc.

### Diagrama de arhitectură a sistemului

Structura generală a sistemului:

1. Utilizatorul
   * Mută piesele de pe tabla de șah.
   * Apasă butonul de comandă pentru a iniția citirea senzorilor.
2. Partea hardware
   * Senzori reed: detectează prezența pieselor pe tablă.
   * Multiplexoare CD74HC4067: ajută Arduino să citească toți cei 64 de senzori.
   * Arduino Mega: preia datele de la senzori și le transmite către calculator prin port USB. Primește comenzile pentru mișcarea brațului robotic.
   * Driver PCA9685: controlează cele 6 servomotoare ale brațului robotic.
   * Braț robotic: mută piesele conform comenzilor primite.
   * Buzzer: anunță utilizatorul că este rândul său să mute piesa.
3. Partea software
   * Arduino IDE: utilizat pentru programarea Arduino.
   * Cod Arduino: gestionează citirea senzorilor și mișcarea brațului robotic.
   * Python: primește datele de la Arduino, procesează poziția pieselor cu ajutorul AI-ului și trimite comanda de mutare înapoi la Arduino.
   * Biblioteci Python: pyserial pentru comunicare serială și python-chess pentru logica jocului de șah.
4. Comunicarea
   * Datele dintre Arduino și calculator se transmit prin conexiune serială USB.
   * Informațiile circulă într-un flux clar: Senzori -> Arduino -> Calculator (Python + AI) -> Arduino -> Braț robotic.
5. Fluxul informației
   * Utilizatorul mută piesa și apasă butonul.
   * Arduino citește poziția pieselor și trimite datele către aplicația Python.
   * Python folosește AI-ul pentru a decide mutarea.
   * Comanda este trimisă înapoi Arduino-ului.
   * Arduino controlează driverul PCA9685 pentru a executa mutarea cu brațul robotic.
   * După mutare, buzzer-ul anunță utilizatorul că este rândul său.

## Proiectarea sistemului

### Conversii de date

Semnalele binare de la senzori sunt transformate într-o matrice logică pentru a fi analizate de AI. Rezultatul este transmis Arduino-ului sub formă de comenzi pentru controlul mișcărilor brațului robotic.

### Interfețe utilizator

Utilizatorul interacționează cu sistemul prin apăsarea butonului de comandă pentru a declanșa citirea senzorilor. Semnalul sonor emis de buzzer anunță când mutarea AI-ului este completă și este rândul utilizatorului.

### Proiectarea interfețelor cu utilizatorul

Interfața constă într-un buton de comandă și un semnal sonor pentru informarea utilizatorului. Sistemul nu include o interfață grafică, fiind gândit pentru interacțiune directă și ușoară.

## Scenarii de utilizare

1. Utilizatorul mută o piesă pe tabla de șah.
2. Apasă butonul de comandă pentru a transmite mutarea.
3. Arduino citește datele de la senzori.
4. Datele sunt trimise aplicației Python prin cablu USB.
5. AI-ul analizează poziția pieselor și calculează următoarea mutare.
6. Comenzile pentru brațul robotic sunt trimise Arduino-ului.
7. Brațul robotic execută mutarea AI-ului.
8. Buzzer-ul semnalizează finalizarea mutării și anunță că este rândul utilizatorului.

## Proiectare de detaliu

### Proiectare hardware de detaliu

1. Arduino Mega 2560

* Alimentare: 5V prin USB sau sursă externă
* Conectori: USB tip B pentru comunicație cu calculatorul, pini digitali pentru multiplexoare și driver PCA9685
* Funcție: citește senzorii reed și controlează brațul robotic

1. Multiplexoare CD74HC4067

* Alimentare: 3.3V - 5V
* Semnal logic: TTL compatibil cu Arduino
* Conectori: pini digitali standard, conectați la Arduino
* Funcție: extind intrările Arduino pentru a citi toți cei 64 de senzori reed

1. Senzori reed

* Alimentare: pasiv, funcționează prin contact magnetic
* Conectori: fire conectate la multiplexoare
* Funcție: detectează prezența pieselor de șah pe tablă

1. Driver PCA9685

* Alimentare: 5V logic pentru control, până la 6V pentru alimentarea servo
* Conectori: magistrală I2C către Arduino (SCL, SDA), ieșiri PWM către servomotoare
* Funcție: controlează semnalele PWM pentru cele 6 servomotoare

1. Braț robotic cu 6 servomotoare

* Alimentare: până la 6V prin driverul PCA9685
* Conectori: PWM de la driver către fiecare servomotor
* Funcție: execută mutările calculate de AI

1. Buton de comandă

* Funcție: permite utilizatorului să inițieze citirea senzorilor și procesul de calcul AI

1. Buzzer

* Alimentare: 5V conectat la Arduino
* Funcție: oferă semnal sonor la finalizarea mutării

1. Calculator local (PC)

* Alimentare: standard 220V AC
* Procesor: minim Intel i3
* Memorie: minim 4 GB RAM
* Spațiu de stocare: minim 1 GB liber pentru aplicații
* Conectori: port USB pentru comunicare cu Arduino
* Funcție: rulează aplicația Python și AI-ul pentru deciziile de joc

1. Reprezentare grafică simplificată:  
   PC -> Arduino (USB) -> Multiplexoare -> Senzori reed  
   PC -> Arduino (USB) -> Driver PCA9685 -> Servomotoare braț robotic  
   Arduino -> Buzzer și Buton comandă

### Proiectare software de detaliu

1. **Colectare și transmitere date de la senzori (Arduino)**
   * **Clasificare**  
     Serviciu hardware-software integrat pentru colectare date
   * **Definiție**  
     Acest serviciu citește pozițiile pieselor de pe tabla de șah de la senzorii reed și trimite informațiile sub formă de matrice de date către aplicația Python, prin conexiune serială USB.
   * **Cerințe**  
     Citirea corectă a tuturor pozițiilor de pe tablă (64 senzori) și transmiterea fără erori a datelor către aplicația Python.
   * **Constrângeri**  
     Datele sunt transmise numai după apăsarea butonului de comandă. Comunicarea se face prin port serial, necesitând sincronizare manuală. Formatul datelor trebuie să fie simplu pentru procesare rapidă.
   * **Compoziție**  
     Serviciul include: inițializarea multiplexoarelor, citirea secvențială a senzorilor reed, formarea matricei de date și trimiterea prin USB către Python.
   * **Utilizatori/Interacțiuni**  
     Acest serviciu interacționează cu aplicația Python pentru transmiterea datelor și cu utilizatorul prin butonul de comandă.
   * **Procesare**  
     După apăsarea butonului, se pornește citirea senzorilor, se formează matricea și se trimite prin portul serial. Nu există procesare avansată, ci doar colectare și trimitere brută a datelor.
   * **Interfețe/Exporturi**  
     Exportă matricea senzorilor prin conexiune serială USB către aplicația Python. Fiecare element al matricei este transmis ca un caracter binar ('0' sau '1').

### 7.3 Proiectare de detaliu pentru performanța sistemului

* Cerințe/estimări de capacitate și volum.  
  Sistemul procesează date de la 64 de senzori reed pentru fiecare mutare. Volumul de date este redus, aproximativ 64 de valori binare per citire.
* Așteptări de performanță.  
  De la apăsarea butonului până la finalizarea mutării, întregul proces trebuie să dureze maxim 10 secunde.
* Cerințe de disponibilitate.  
  Sistemul trebuie să funcționeze continuu pe durata sesiunii de joc, fără întreruperi. Disponibilitatea cerută este de 100% pe durata utilizării.
* Proiectare de fiabilitate pentru a îndeplini cerințele de disponibilitate.  
  Toate conexiunile hardware sunt verificate înainte de utilizare. Alimentarea sistemului este stabilă pentru a preveni căderile de tensiune.

### 7.4 Proiectare detaliată a comunicațiilor interne (între componente)

Sistemul are un **singur server** (calculatorul local) și un **client hardware** (Arduino Mega), comunicând prin conexiune directă USB. Alte componente (multiplexoare, driver PCA9685, senzori reed, braț robotic) sunt conectate direct la Arduino și nu necesită rețea dedicată.

## Controale pentru verificarea integrității sistemului

Accesul la componente și date este controlat fizic de către utilizator. Nu există proceduri automate de audit sau jurnalizare, verificările fiind realizate manual, prin observarea directă a funcționării sistemului. În această etapă, sistemul nu implementează piste de auditare a aplicațiilor și nu folosește tabele electronice pentru validarea datelor, deoarece toate informațiile sunt procesate în timp real și nu sunt stocate. De asemenea, nu există procese automate pentru adăugarea, ștergerea sau actualizarea datelor, iar identificarea informațiilor de auditare, precum utilizatorul sau terminalul de rețea, nu este necesară întrucât sistemul este complet local, cu operare manuală și controlată direct de utilizator.

Anexa A: Gestiunea modificărilor documentului

Tabel 1 – Înregistrarea modificărilor asupreaa documentului curent

| versiune | Data | Autorul/Deținătorul | Descriere |
| --- | --- | --- | --- |
| 1.0 | |  | | --- | | 13/04/2025 |  |  | | --- | |  | | |  | | --- | |  |  |  | | --- | | Vilcu Elena Roxana | | |  | | --- | |  |  |  | | --- | | Creare inițială a documentului completat pentru proiectul brațului robotic controlat prin AI | |
| 1.0 | 14/04/2025 | |  | | --- | |  |  |  | | --- | | Vilcu Elena Roxana | | |  | | --- | |  |  |  | | --- | | Adăugare detalii suplimentare pentru cerințele de proiectare hardware și software, actualizare capitole finale | |
| 1.0 | |  | | --- | |  |  |  | | --- | | 14/04/2025 | | |  | | --- | |  |  |  | | --- | | Vilcu Elena Roxana | | |  | | --- | |  |  |  | | --- | | Finalizare completă document, integrare toate cerințele din șablon și pregătire pentru predare | |

Anexa B: Acronime

Tabel 2 - Acronime

| Acronim | Forma completă |
| --- | --- |
| AI | |  | | --- | |  |  |  | | --- | | Inteligență artificială | |
| IDE | Integrated Development Environment |
| PC | |  | | --- | |  |  |  |  | | --- | --- | | Personal Computer |  | |
| |  | | --- | | PWM |  |  | | --- | |  | | |  | | --- | |  |  |  | | --- | | Pulse Width Modulation | |
| USB | |  | | --- | |  |  |  | | --- | | Universal Serial Bus | |