Digital Chess Clock - Popescu Maria Bianca

Introducere

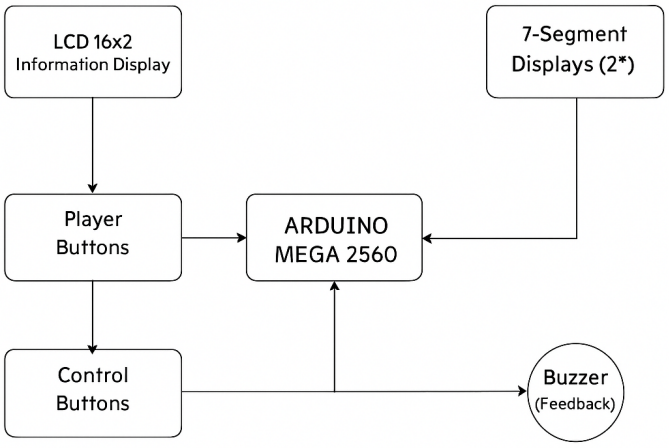
Proiectul implementează un ceas de șah digital folosind Arduino, afișaje cu 7 segmente și un LCD. Ceasul oferă funcționalități standard pentru turneele de șah:

- Cronometrează timpul fiecărui jucător
- Permite setarea diferitelor moduri de joc (Blitz, Rapid, Clasic)
- Oferă feedback sonor pentru acțiunile jucătorilor
- Permite pauză/reluare a timpului

Ideea a pornit de la dorința de a crea un instrument practic pentru jucătorii de șah, care să fie în același timp un exercițiu educațional de programare și electronică. Utilitatea acestui proiect constă în:

- Economisirea costurilor comparativ cu ceasurile de șah comerciale
- Personalizarea funcționalităților conform nevoilor specifice
- Înțelegerea principiilor de programare și electronică în contextul unui sistem embedded funcțional

Descriere generală



Hardware Design

Ceasul de șah digital este un sistem complex care îmbină componente hardware și software pentru a oferi o soluție funcțională și intuitivă de cronometrare pentru jocul de șah. Proiectul utilizează o singură placă Arduino MEGA 2560 pentru a controla toate componentele și a implementa logica

necesară.

Descrierea modulelor și interacțiunilor

1. Unitatea centrală de procesare (Arduino MEGA 2560)

Arduino MEGA 2560 reprezintă “creierul” sistemului, coordonând toate funcționalitățile și procesând interacțiunile utilizatorului. Acesta a fost ales datorită numărului mare de pini I/O disponibili (54 pini digitali și 16 pini analogici), care permit conectarea simultană a tuturor componentelor necesare. Arduino rulează firmware-ul care implementează logica jocului, gestionează timerii, procesează input-ul de la butoane și controlează afișajele și feedback-ul audio.

2. Modulul de afișare a informațiilor (LCD 16x2)

LCD-ul oferă o interfață vizuală pentru utilizator, afișând informații despre:

- Meniul de selectare a modurilor de joc (Blitz, Rapid, Clasic)
- Starea curentă a jocului (în desfășurare, pauză, terminat)
- Instrucțiuni pentru utilizator

3. Module de cronometrare (Afișaje cu 7 segmente)

Cele două afișaje cu 7 segmente și 4 digiti sunt utilizate pentru a afișa timpul rămas pentru fiecare jucător în format MM:SS (minute:secunde). Aceste afișaje utilizează tehnica de multiplexare pentru a controla toate cele 32 de segmente (8 segmente × 4 digiti) cu doar 12 pini pentru fiecare afișaj. Multiplexarea implică activarea rapidă și secvențială a câte unui digit la un moment dat, creând iluzia că toate sunt aprinse simultan datorită persistenței vizuale.

4. Modulul de interacțiune cu utilizatorul (Butoane)

Sistemul include patru butoane tactile:

- Două butoane pentru jucători: Fiecare jucător apasă butonul său când termină mutarea, transferând timpul la adversar
- Două butoane pentru control: Un buton pentru navigarea prin opțiuni și un buton pentru confirmarea selecției, utilizate în principal pentru configurarea modurilor de joc

Butoanele sunt conectate la Arduino folosind intrările analogice și utilizează rezistențe pull-up interne pentru stabilitate.

5. Modulul de feedback audio (Buzzer)

Buzzer-ul oferă feedback audio pentru diverse evenimente din timpul jocului:

- Confirmarea apăsării butoanelor
- Alerte pentru timpul scăzut rămas
- Semnale pentru începerea/terminarea jocului

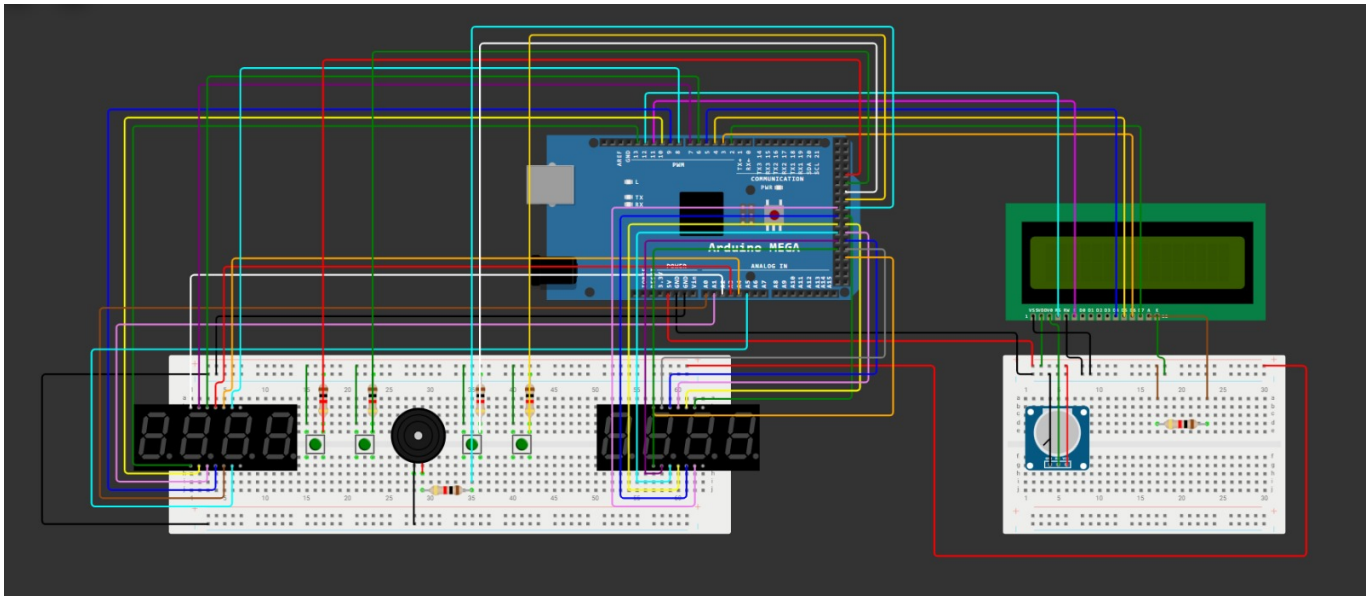
Interacțiunea dintre module

Modulele interacționează într-un flux logic care permite funcționarea coerentă a ceasului de șah:

La pornire, sistemul afișează meniul principal pe LCD, permițând utilizatorului să selecteze modul de joc folosind butoanele de control. După selectarea modului, timpii inițiali sunt setați pentru ambii jucători și afișați pe cele două afișaje cu 7 segmente. Jocul începe cu cronometrul jucătorului 1 activat. Când acesta termină mutarea și apasă butonul său, timpul său se oprește și începe cronometrul jucătorului 2. Buzzer-ul emite sunete pentru a indica transferul de control între jucători și pentru a avertiza când timpul este pe cale să expire. Arduino MEGA gestionează continuu afișajele, actualizându-le pentru a reflecta timpul rămas, și procesează input-ul de la butoane pentru a detecta schimbările de stare (pauză, terminare mutare, etc.). LCD-ul afișează informații suplimentare despre starea jocului, cum ar fi jucătorul activ, modul de joc curent, și eventuale notificări.

Această arhitectură modulară permite o implementare clară a funcționalităților și facilitează eventuale extinderi ulterioare ale sistemului, cum ar fi adăugarea de moduri de joc suplimentare sau funcții avansate de cronometrare.

Hardware Design



Lista de componente

1 x Arduino MEGA 2560

- Tensiune de operare: 5V
- Curent per pin I/O: 20mA (maxim)
- Frecvență: 16 MHz
- 54 pini digitali I/O (15 PWM)
- 16 intrări analogice

1 x LCD 16x2

- Tensiune de operare: 5V
- Curent consumat: ~1mA (fără backlight)

2 x Afișaj cu 7 segmente cu 4 cifre

- Tensiune de operare: 5V
- Curent per segment: ~20mA

4 x Butoane

- Configurație: Normally open (NO)
- 4 pini (2 perechi conectate intern când sunt apăsate)

1 x Buzzer

- Tensiune de operare: 5V
- Frecvență: 2-5 kHz (tonuri optime)

1 x Potențiometru

Pentru reglarea contrastului LCD-ului

Rezistențe

Componente de interconectare

- 2 x Breadboard
- Fire de conexiune (jumperwires) diverse lungimi

Schema electrică LCD 16×2:

- RS (Register Select) → Arduino pin 12
- E (Enable) → Arduino pin 11
- D4 → Arduino pin 5
- D5 → Arduino pin 4
- D6 → Arduino pin 3
- D7 → Arduino pin 2
- VSS → GND
- VDD → 5V
- V0 → Mijlocul potențiometrului de 10K
- Pinii extremi ai potențiometrului: unul la 5V, celălalt la GND
- RW → GND (mod scriere)
- Backlight (A) → 5V (prin rezistență dacă este necesar)
- Backlight (K) → GND

Afișaj cu 7 segmente pentru Jucătorul 1:

- Segment A → Arduino pin 6
- Segment B → Arduino pin 7
- Segment C → Arduino pin 8
- Segment D → Arduino pin 9
- Segment E → Arduino pin 10
- Segment F → Arduino pin 13
- Segment G → Arduino pin A0
- Segment DP → Arduino pin A1
- Digit 1 → Arduino pin A2

- Digit 2 → Arduino pin A3
- Digit 3 → Arduino pin A4
- Digit 4 → Arduino pin A5

Afișaj cu 7 segmente pentru Jucătorul 2:

- Segment A → Arduino pin 49
- Segment B → Arduino pin 51
- Segment C → Arduino pin 43
- Segment D → Arduino pin 26
- Segment E → Arduino pin 22
- Segment F → Arduino pin 31
- Segment G → Arduino pin 28
- Segment DP → Arduino pin 24
- Digit 1 → Arduino pin 53
- Digit 2 → Arduino pin 47
- Digit 3 → Arduino pin 45
- Digit 4 → Arduino pin 30

Butoane:

Buton Jucător 1:

- Un pin → Arduino pin 32
- Alt pin (diagonal) → GND
- Rezistență 10K între 32 și 5V (pull-up intern activat)

Buton Jucător 2:

- Un pin → Arduino pin 38
- Alt pin (diagonal) → GND
- Rezistență 10K între 38 și 5V (pull-up intern activat)

Buton Navigare Mod:

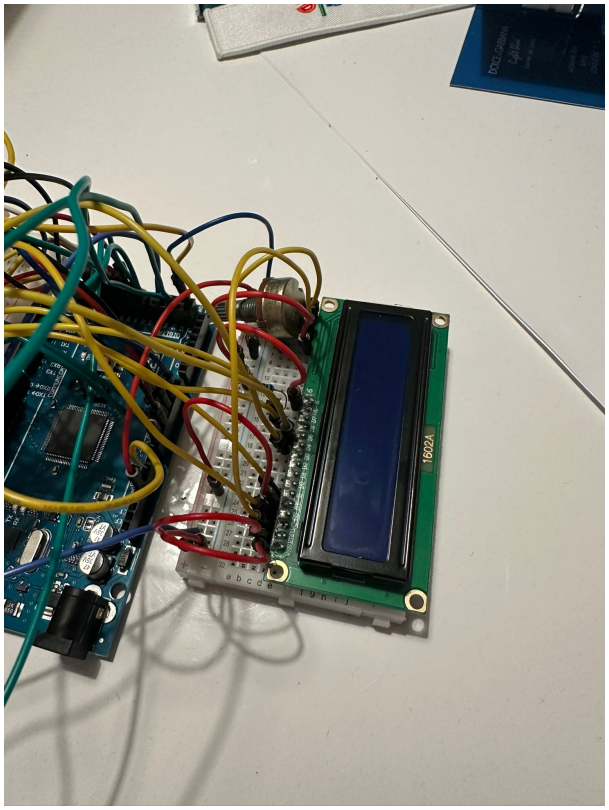
- Un pin → Arduino pin 36
- Alt pin (diagonal) → GND
- Rezistență 10K între 36 și 5V (pull-up intern activat)

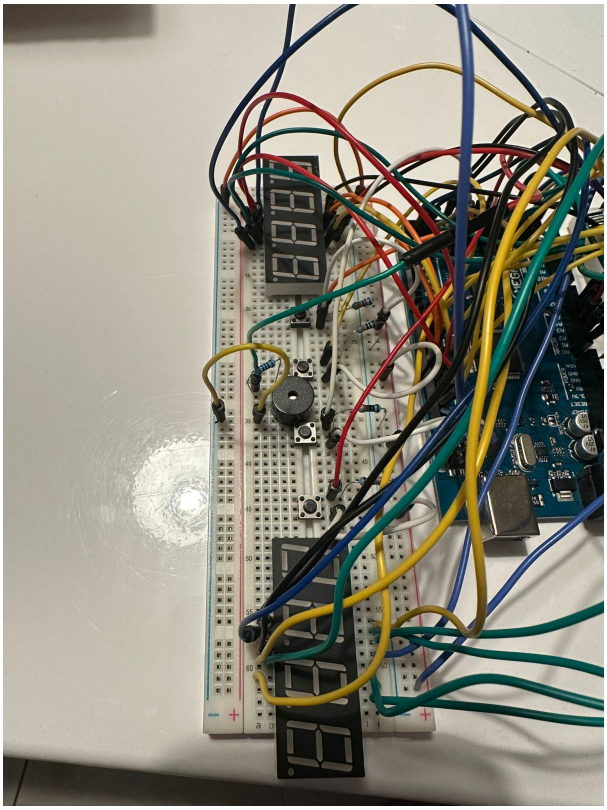
Buton Selectare Mod:

- Un pin → Arduino pin 34
- Alt pin (diagonal) → GND
- Rezistență 10K între 34 și 5V (pull-up intern activat)

Buzzer:

- Pin pozitiv (+) → Arduino pin 41 (PWM)
- Pin negativ (-) → GND





Software Design

Descrierea codului aplicaţiei

- Mediu de dezvoltare: proiectul a fost dezvoltat folosind mediul de programare **Arduino IDE**, utilizând limbajul C++ specific microcontrolerelor compatibile cu Arduino. Testele au fost realizate direct pe placă, folosind atât afișajele fizice cât și Serial Monitor pentru depanare și verificare a stărilor interne.
- Biblioteci utilizate:
  - **LiquidCrystal.h** – pentru afișarea mesajelor text pe un LCD 16×2, controlat în mod paralel.
  - **EEPROM.h** – pentru salvarea persistentă a modului de joc preferat de utilizator, astfel încât

selecția să fie reținută și după oprirea alimentării.
- **Arduino.h** – biblioteca implicită ce furnizează funcționalitățile de baza

Structura aplicației

Codul aplicației este structurat în jurul a două componente principale:

- **Interfața utilizatorului** – compusă din butoane fizice, LCD și buzzer. Utilizatorul poate naviga prin modurile de joc, poate porni partida, schimba tura, reseta jocul sau pune pauză.
- **Controlul afișajelor și timpului** – două afișaje cu 4 cifre (7 segmente) sunt folosite pentru a arăta timpul fiecărui jucător în format MM:SS. Timpul este decrementat în funcție de tură, iar logica internă se ocupă de comutarea între jucători și semnalizarea stărilor prin buzzer și LCD.

Logica principală se bazează pe funcțiile standard Arduino: `setup()` (inițializare) și `loop()` (execuție continuă). În cadrul `loop()`, se gestionează evenimentele produse de butoane și se actualizează timpul și afișajele în mod continuu, în funcție de starea curentă a jocului.

Algoritmi și structuri implementate

Aplicația implementează următoarele funcționalități software relevante:

- **Selectarea modului de joc** – utilizatorul navighează între modurile Blitz (5 minute), Rapid (10 minute) și Classic (30 minute) folosind două butoane. La selecție, modul este salvat în EEPROM pentru a fi reîncărcat automat la o repornire viitoare.
- **Cronometru decremental** – timpul fiecărui jucător este decrementat o dată pe secundă, folosind funcția `millis()` pentru a asigura temporizare fără a bloca execuția restului codului.
- **Multiplexare afișaj 7 segmente** – fiecare dintre cele două afișaje este controlat direct, fără un driver dedicat. Pentru a reduce numărul de pini utilizați și a menține afișajul stabil, s-a implementat o rutină de multiplexare rapidă a celor 4 cifre.
- **Debounce software pentru butoane** – toate butoanele sunt gestionate cu o întârziere software de 200 ms, pentru a preveni detectarea accidentală a unor apăsări multiple cauzate de rebond mecanic.
- **Gestionarea logicii de joc** – cronometrul rulează doar pentru jucătorul activ. La apăsarea butonului propriu, jucătorul oprește timpul său și pornește timpul adversarului. Dacă timpul unui jucător expiră, jocul se oprește automat și este declanșată o alarmă sonoră.

Funcții importante implementate în cod

Funcție	Descriere
`setup()`	Configurează pinii, inițializează LCD-ul, încarcă modul salvat din EEPROM și pregătește componentele pentru execuția logicii principale.

`loop()`	Gestionează selecția modului de joc, desfășurarea partidei, schimbarea turelor, decrementarea timpului și actualizarea afișajelor. Toate sunt controlate de un state machine simplificat.
`updateDisplay(...)`	Actualizează valorile de timp pe afișajul 7 segmente prin multiplexare. Activează fiecare cifră pe rând cu un delay scurt pentru a simula refresh continuu.
`showDigit(...)`	Afișează o cifră pe una dintre cele patru poziții ale unui afișaj. Activează doar segmentele necesare.
`playSwitchBeep()`	Redă un sunet scurt prin buzzer la schimbarea de tură.
`playAlarm()`	Declanșează o secvență sonoră repetitivă când timpul unui jucător a expirat.

Utilizarea memoriei EEPROM

Pentru a îmbunătăți experiența utilizatorului și a evita selecția repetată a modului de joc la fiecare pornire, proiectul folosește EEPROM pentru a salva indexul modului de joc selectat.

- La pornirea sistemului, valoarea este citită de la adresa 0 folosind `EEPROM.read(0)` și validată.
- Dacă este validă, aceasta este folosită ca mod implicit.
- După selectarea unui mod nou, valoarea este salvată cu `EEPROM.update(0, currentMode)` pentru a evita uzura memoriei EEPROM.

Cod sursă

Codul complet al aplicației poate fi consultat în repository-ul public de pe GitHub:
https://github.com/biancapopescu0411/arduino-digital-chess-clock

Rezultate Obţinute

Acesta este un video demonstrativ cu functionalitatea proiectului:

Concluzii

Proiectul realizat este un ceas digital de sah complet functional, care permite selectarea modului de joc, afisarea timpului pentru fiecare jucator si semnalizarea sonora in momentele importante. Am folosit un LCD pentru interfata cu utilizatorul, doua afisaje cu 7 segmente pentru a afisa timpii si mai multe butoane pentru interactiune directa.

Pe parcursul implementarii am aplicat notiuni invatate la mai multe laboratoare, cum ar fi lucrul cu afisaje digitale, butoane, semnalizare cu buzzer si salvarea datelor in EEPROM. Proiectul a fost o ocazie buna de a pune in practica teoria si de a intelege mai bine cum functioneaza un sistem embedded complet.

Am facut inclusiv lucru manual, construind si o carcasa pentru a proteja componentele si pentru a

oferi un aspect mai profesional. Pe viitor, as putea adauga si alte functii, cum ar fi increment de timp sau salvarea scorurilor.

In final, proiectul mi-a oferit o experienta completa si m-a ajutat sa inteleg mai bine ce inseamna sa construiesti un dispozitiv electronic real, de la idee pana la testare.

Bibliografie/Resurse

Listă cu documente, datasheet-uri, resurse Internet folosite, eventual grupate pe **Resurse Software** şi **Resurse Hardware**.

Export to PDF