Șah Interactiv pe Microcontroler (Console Chess)

Introducere

- **Ce face:** Proiectul „Console Chess” este o consolă portabilă bazată pe ATmega328P, care permite jucarea unei partide de șah (Player vs. Player). Folosește un ecran TFT de 2.8” pentru afișarea tablei, un D-Pad fizic (5 butoane) pentru mutări, un LED RGB și un buzzer pasiv pentru feedback vizual și sonor în timp real.
- **Care este scopul lui:** Implementarea logicii complexe a șahului pe un microcontroler pe 8 biți în C (bare-metal), optimizând utilizarea memoriei și gestionând eficient multiple periferice (comunicație SPI, generare semnale PWM, citire pini digitali).
- **Care a fost ideea de la care am pornit:** Construirea unui sistem interactiv complet de tip „retro-console” de la zero, îmbinând un joc clasic de strategie cu provocările ingineriei hardware la nivel de regiștri.
- **De ce este util pentru alții și pentru mine:** Pentru mine, este mediul ideal de a aprofunda interfațarea componentelor cu niveluri logice diferite (5V vs 3.3V) și lucrul avansat cu Timere și SPI. Pentru alți studenți, proiectul servește drept ghid practic și bază de cod (arhitectură software) pentru dezvoltarea de jocuri embedded pe arhitectura AVR.

Descriere generală

Arhitectura proiectului este structurată în patru mari blocuri funcționale (Alimentare, Input, Procesare și Output), care interacționează constant pentru a oferi o experiență de joc fluidă.

Descrierea modulelor și a modului de interacțiune:

- **Alimentare USB 5V:** Reprezintă sursa de energie a întregului sistem. Tensiunea este preluată direct din portul USB (prin placa ATmega328P) și este distribuită către microcontroler și restul perifericelor de pe breadboard, asigurând funcționarea continuă.
- **Blocul de Input (Butoane D-Pad):** Este format din 5 butoane tactile configurate direcțional (Sus, Jos, Stânga, Dreapta, Select). Microcontrolerul interoghează constant starea acestor pini digitali (citire pini) pentru a capta intențiile jucătorului, cum ar fi navigarea cursorului pe tablă și selectarea pieselor de șah.
- **Blocul de Procesare (Microcontroler ATmega328P):** Reprezintă „creierul” consolei. La nivel software, acesta rulează întreaga logică a jocului: stochează matricea tablei de șah, validează mutările conform regulilor, gestionează rândul fiecărui jucător și declanșează evenimente (șah, șah-mat). El preia datele din Input, calculează noua stare a jocului și comandă perifericele din Output.
- **Blocul de Output:** Este responsabil pentru feedback-ul vizual și auditiv oferit utilizatorului:
  - **Convertor Nivel Logic & Ecran TFT 2.8 inch:** Microcontrolerul trimite date grafice (coordonate, culori, imagini cu piese) folosind protocolul SPI la o tensiune de 5V. Deoarece

ecranul funcționează strict la 3.3V, convertorul de nivel logic acționează ca o punte de siguranță, scăzând tensiunea semnalelor SPI fără a pierde viteza de transmisie a datelor.
- **Buzzer Pasiv:** Comunică cu microcontrolerul prin intermediul timerelor hardware. Primește semnale PWM (Pulse Width Modulation) de diferite frecvențe pentru a genera sunete specifice la navigare, erori (mutări invalide) sau victorii.
- **LED RGB:** Este controlat direct prin pinii digitali de ieșire (I/O). Software-ul schimbă starea acestor pini pentru a modifica culoarea LED-ului, oferind o indicație vizuală rapidă (de exemplu, roșu pentru Player 1 și albastru pentru Player 2).

Hardware Design

Stadiul actual al implementării (Milestone 2): Hardware-ul a fost asamblat cu succes pe breadboard. Microcontrolerul ATmega328P comunică corect cu ecranul TFT, generând imaginea tablei de șah. Divizorul de tensiune implementat manual protejează ecranul, iar perifericele de feedback (LED RGB, Buzzer) sunt integrate și gata de utilizare. Din cauza constrângerilor de spațiu fizic de pe breadboard (datorate rezistențelor), numărul de butoane pentru D-Pad a fost redus de la 5 la 4.

Nume Componentă	Cantitate
Placă de dezvoltare ATmega328P Xplained Mini	1
Ecran afișaj TFT LCD 2.8'' (Interfață SPI)	1
Breadboard 830 puncte	1
Butoane tactile (Microîntrerupătoare 6x6x6 mm) pentru D-Pad	4
Modul Buzzer Pasiv (suport semnal PWM)	1
LED RGB (pentru indicarea turei jucătorilor)	1
Rezistențe (1k Ohm, 2k Ohm, 220 Ohm) pt. divizor tensiune și LED	Set
Set fire de conexiune Dupont (Tată-Tată și Tată-Mamă de 20 cm)	1
Cablu USB (pentru alimentare și programare)	1

Maparea Pinilor și Justificarea Alegerii Lor:

Componentă	Pin ATmega328P	Rol și Justificare
TFT - MOSI	PB3 (Digital 11)	Linia de date SPI. Folosit pinul hardware SPI pentru viteză maximă de desenare.
TFT - SCK	PB5 (Digital 13)	Linia de ceas SPI. Folosit pinul hardware SPI.
TFT - CS	PB2 (Digital 10)	Chip Select. Semnalizează ecranului când să asculte magistrala SPI.
TFT - DC	PB1 (Digital 9)	Data/Command. Diferențiază între datele pentru culori și comenzile de configurare.
TFT - RST	PB0 (Digital 8)	Hardware Reset pentru controlerul ecranului.
LED RGB - Roșu	PD5 (Digital 5)	Pin cu suport PWM. Permite reglarea intensității sau folosirea ca pin digital simplu.
LED RGB - Albastru	PD6 (Digital 6)	Pin cu suport PWM, folosit pentru a indica rândul celui de-al doilea jucător.
Buzzer Pasiv	PD3 (Digital 3)	Pin cu suport PWM (Timer hardware), necesar pentru a genera tonuri de diferite frecvențe.

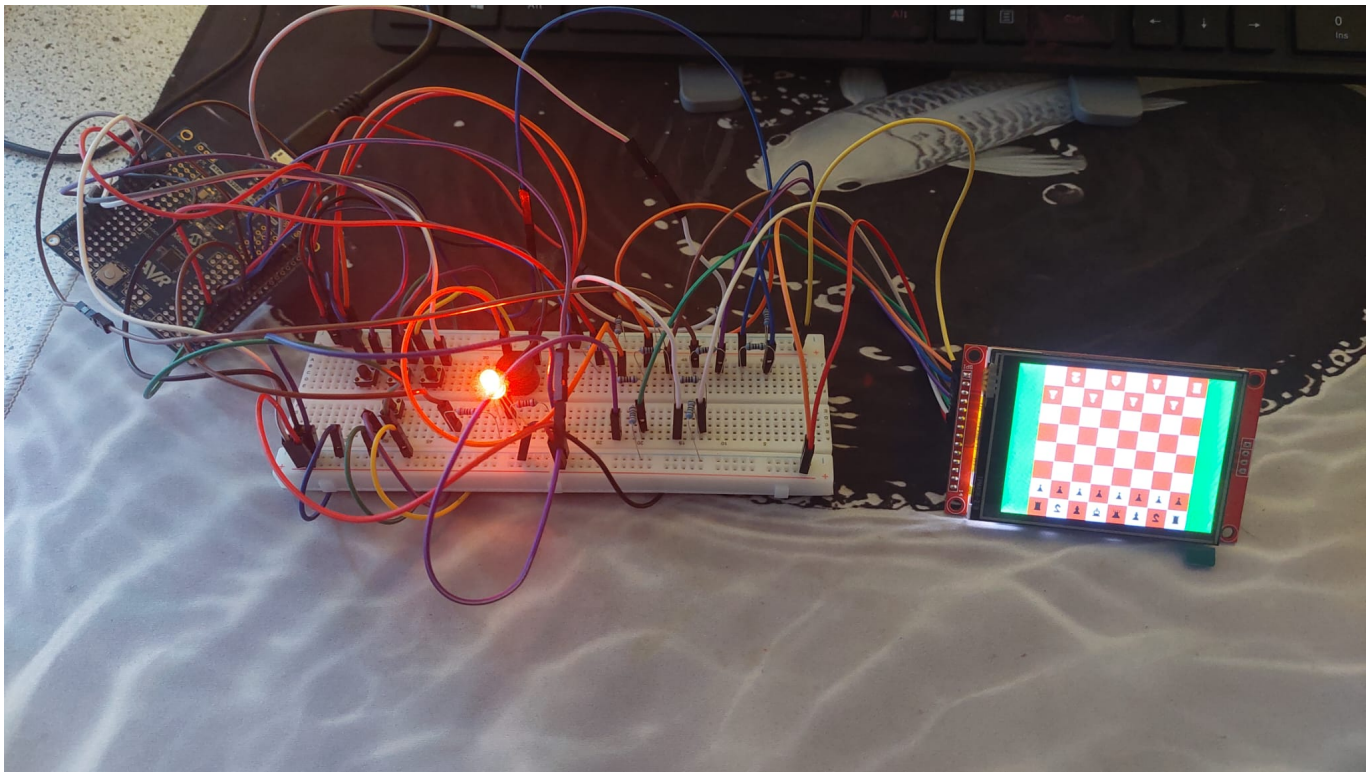
Butoane D-Pad	PC0-PC3 (Analog A0-A3)	Deoarece pinii digitali sunt ocupați de SPI și PWM, am folosit pinii analogici configurați ca intrări digitale (I/O).

Schema Electrică (Simulare vs. Realitate):

Schema de mai sus a fost primul meu pas, realizat în mediul Wokwi pentru a valida logica conexiunilor. Totuși, trecerea de la simulator la breadboard-ul real a venit cu provocări practice pe care a trebuit să le rezolv:

- **Microcontrolerul:** Deși în simulare apare un Arduino Uno clasic (din lipsa altor modele în simulator), fizic folosesc placa ATmega328P Xplained Mini. Din fericire, arhitectura cipului este aceeași, deci logica pinilor s-a potrivit perfect.
- **Problema Nivelului Logic (Protejarea Ecranului):** Aceasta a fost cea mai mare diferență față de simulare. În Wokwi, ecranul acceptă semnale de 5V direct din microcontroler, dar în realitate, ecranul meu funcționează strict cu logică de 3.3V. Dacă aș fi conectat pinii de date (MOSI, SCK, CS, DC, RST) direct la 5V, aș fi prăjit controlerul ecranului. Pentru că nu am folosit un modul dedicat de tip Level Shifter, a trebuit să implementez manual **divizori de tensiune** pentru fiecare fir de date.
- **Cum funcționează divizorul:** Am folosit o rețea de rezistențe. Din pinul plăcii pleacă o rezistență de **1k Ohm** (R1), urmată de o intersecție (nod) din care pleacă firul spre ecran, iar mai departe o rezistență de **2k Ohm** (R2) duce la masă (GND). Aplicând formula divizorului de tensiune ($V_{out} = V_{in} \times \frac{R_2}{R_1 + R_2}$), am obținut $5V \times \frac{2}{3} \approx 3.33V$. Așa am asigurat o tensiune sigură pentru ecran, menținând viteza SPI.
- **Compromisul Butoanelor:** În planul din simulare aveam 5 butoane. Când am transpus asta pe breadboard, construcția celor 5 divizori de tensiune (implicând 10 rezistențe și multe punți la GND) a ocupat foarte mult spațiu fizic. Pentru a păstra cablajul ordonat și a evita scurtcircuitele, am decis să scot butonul de “Select” și să păstrez un D-Pad curat cu doar 4 direcții.

Rezultatul Hardware:



Imaginea de mai sus arată montajul final pe breadboard și confirmă că totul funcționează cum trebuie pentru acest milestone:

- **Ecranul merge perfect:** Se vede clar tabla de șah afișată pe ecran. Asta înseamnă că divizorii de tensiune făcuți din rezistențe funcționează bine, iar datele trimise de placă ajung corect și rapid la ecran fără să se piardă pe drum.
- **LED-ul RGB reacționează:** În poză se vede LED-ul aprins pe roșu. Asta îmi confirmă că pot folosi pinii digitali ca să arăt vizual starea jocului (de exemplu, când este rândul jucătorului cu piesele negre).
- **Organizarea firelor:** Chiar dacă par foarte multe conexiuni, am încercat să le grupez logic: butoanele sunt în partea stângă, placa de control este sus, iar divizorii de tensiune și ecranul sunt așezate în partea dreaptă.

Software Design

Descrierea codului aplicaţiei (firmware):

- mediu de dezvoltare (if any) (e.g. AVR Studio, CodeVisionAVR)
- librării şi surse 3rd-party (e.g. Procyon AVRlib)
- algoritmi şi structuri pe care plănuiţi să le implementaţi
- (etapa 3) surse şi funcţii implementate

Rezultate Obţinute

Care au fost rezultatele obţinute în urma realizării proiectului vostru.

Concluzii

Download

O arhivă (sau mai multe dacă este cazul) cu fişierele obţinute în urma realizării proiectului: surse, scheme, etc. Un fişier README, un ChangeLog, un script de compilare şi copiere automată pe uC crează întotdeauna o impresie bună .

Fişierele se încarcă pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul în care se încarcă fişierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (dacă este cazul).

Exemplu: Dumitru Alin, 331CC → **:pm:prj2009:cc:dumitru_alin**.

Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

Bibliografie/Resurse

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

Export to PDF