TimeStrike

Autor: Catarama Dmitrii

Grupa: 332CD

Introducere

Proiectul este un joc interactiv cu două moduri: unul în care jucătorii trebuie să apese rapid un buton după un semnal și altul în care trebuie să apese cât mai aproape de un timp țintă (ex. 11 secunde), ambele afișând rezultatele pe un ecran LCD.

Scopul este să testeze și să antreneze viteza de reacție și simțul timpului al jucătorilor, oferind în același timp o experiență competitivă și distractivă în multiplayer.

Am pornit de la ideea unui joc simplu de reflexe, apoi am adăugat și o provocare diferită – estimarea timpului – pentru a diversifica jocul și a-l face mai captivant.

Este util pentru dezvoltarea reflexelor și a percepției temporale, fiind ușor de folosit și distractiv. În plus, ne ajută să punem în practică concepte învățate la laborator, precum lucrul cu întreruperi, afișaje LCD, butoane și module de timp.

Descriere generală

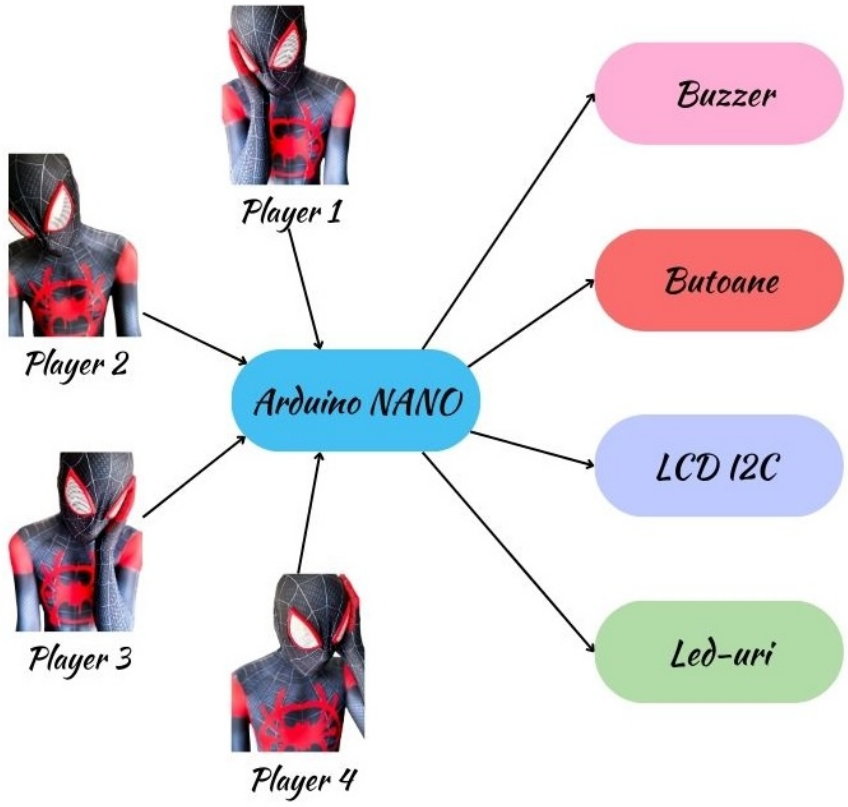
Proiectul “TimeStrike” are doua moduri de joc:

```
1. Modul 1 (Reflex): jucatorii trebuie sa apese butonul intr-un interval cat
mai scurt dupa un semnal (ex: sub 0.4s la nivel easy). Poate fi
single-player, multi-player.
2. Modul 2 (Tinta de timp): jucatorii trebuie sa apese butonul cat mai
aproape de o secunda tinta (ex: 11s).
```

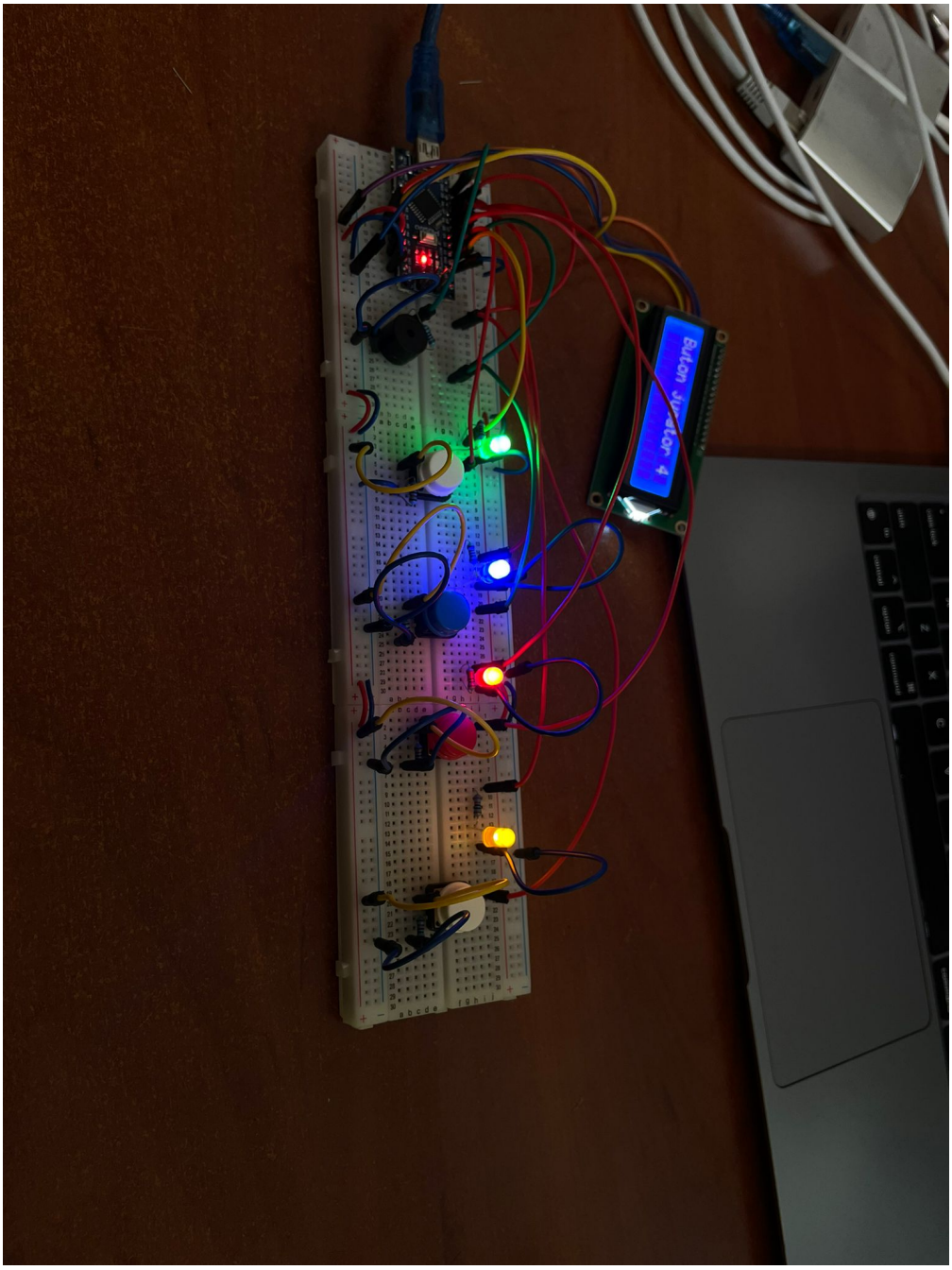
Module hardware:

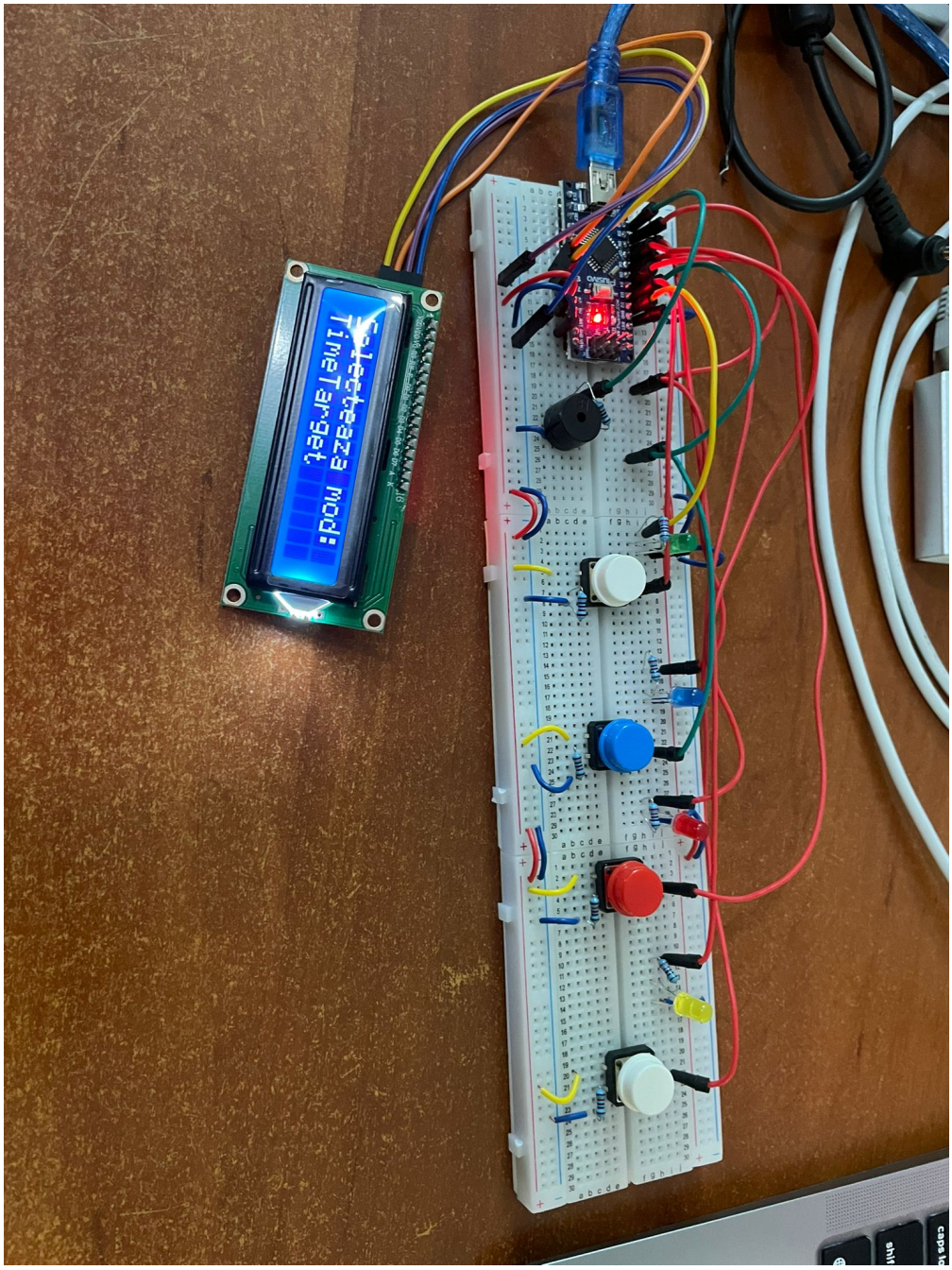
- Arduino Nano (cu ATmega328P);
- LCD 1602 cu interfata I2C: afiseaza mesaje pentru jucatori, cronometre si rezultatele jocului;
- 4 butoane tactile (cate unul pentru fiecare jucator);
- 4 LED-uri (cate unul pentru fiecare jucator);
- Buzzer pasiv: genereaza semnale sonore diferite pentru start, castig sau greseala;
- Breadboard si fire jumper: interconectarea componentelor.
- Rezistente 220 Ohm - 4, pentru led-uri
- Rezistente 10K Ohm - 4, pentru butoane de pull down

- Rezistente 150 Ohm - 1, pentru buzzer



Hardware Design


Buton Jucator 4



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

Aplicatia a fost dezvoltata si compilata folosind Arduino IDE.

Proiectul nu foloseste librarii externe 3rd-party, fiind implementat complet cu functionalitatea nativa a microcontrollerului AVR:

- avr/io.h - pentru accesul la registrele hardware;
- avr/interrupt.h - pentru gestionarea intreruperilor;
- util/delay.h - pentru functiile de delay;
- stdlib.h - pentru functii standard (itoa, etc.);
- string.h - pentru manipularea string-urilor;
- math.h - pentru operatii matematice (pow, fabs);

Aplicatia foloseste o masina de stari pentru a gestiona flow-ul jocului:

```
typedef enum {
   MENU,
   SELECT_REFLEX_MODE,
   SELECT_DIFFICULTY,
   WAIT_SIGNAL,
   SHOW_SIGNAL,
   RESULT,
   SELECT_TIMING_PLAYERS,
   WAIT_TARGET,
   TARGET_RESULT
} GameState;
```

2. Timer Management

- Timer0 cu interrupt pentru implementarea functiei millis();
- Timer1 pentru generarea PWM pentru buzzer;
- Sistem de timing cu precizie de milisecunde;

3. Comunicatie I2C

- Implementare completa a protocolului I2C pentru comunicarea cu LCD-ul:
- Functii de start/stop;
- Transmisia de date;
- Control al LCD-ului in modul 4-bit prin I2C expander;

4. Random Number Generator - pentru a crea intr-un mod realistic procesul de asteptare a start-ului. Generator de numere pseudo-aleatoare folosind Linear Feedback Shift Register (LFSR):

```
uint32_t lfsr = 1;
uint16_t random_range(uint16_t min, uint16_t max) {
    lfsr = (lfsr >> 1) ^ (-(lfsr & 1u) & 0xB400u);
    return min + (lfsr % (max - min));
}
```

5. Anti-Cheat System Sistem de detectare a apasarilor premature care marcheaza jucatorii ca “trisori”.

Surse si functii implementate

1. Functii de sistem:

```
timer_init() - initializare timer pentru millis counter;
millis() - functie echivalenta cu Arduino millis();
delay_ms() - delay in milisecunde;
```

2. Functii I2C si LCD

```
i2c_init(), i2c_start(), i2c_stop(), i2c_write() - protocol I2C;
lcd_init(), lcd_clear(), lcd_set_cursor(), lcd_print() - control LCD;
lcd_print_int(), lcd_print_float() - afisare numere pe LCD;
```

3. Functii GPIO

```
gpio_init() - configurare pini input/output;
digital_read_btn() - citire stare butoane;
digital_write_led() - control LED-uri;
```

4. Functii audio

```
play_tone_start(), play_tone_stop() - control buzzer cu PWM;
play_tone_blocking() - redare ton cu durata specificata;
```

5. Functii de joc

```
handle_menu() - gestionare meniu principal;
select_reflex_mode(), select_difficulty() - selectie optiuni;
wait_signal(), handle_signal() - logica joc reflex;
wait_target() - logica joc timing;
show_reflex_result(), show_timing_result() - afisare rezultate;
```

Laboratoare utilizate Proiectul integreaza concepte din urmatoarele laboratoare:

- Lab 0 - Introducere in toolchain-ul AVR si concepte de baza;
- Lab 1 - GPIO si control pini digitali pentru butoane si LED-uri;
- Lab 2 - Timere si PWM pentru buzzer si sistem de timing;
- Lab 4 - Comunicatie seriala si protocoale (I2C pentru LCD);
- Lab 5 - ADC si senzori (concepte folosite pentru timing precision);

Aceasta arhitectura permite mentenanta usoara si extensibilitatea pentru functionalitati viitoare.

Rezultate Obţinute

- In urma realizarii proiectului, am reusit sa construiesc un sistem functional care detecteaza reactia jucatorului, afiseaza informatii pe LCD si interactioneaza cu utilizatorii prin butoane si buzzer.

- Am implementat doua moduri de joc care functioneaza stabil, iar rezultatele sunt afisate clar.
- Componentele hardware au fost integrate corect si testate atat pe Wokwi, cat si fizic pe placa Arduino Nano.
- Per total, proiectul si-a atins obiectivele si demonstreaza o aplicatie interactiva bazata pe Atmega 328p.

De asemenea, puteti accesa si contribui la proiect aici: https://github.com/dmitrii-catarama/TimeStrike

Concluzii

- Proiectul m-a ajutat sa intelegem mai bine cum se folosesc microcontrolerele in aplicatii practice.
- Am invatat sa lucrez cu mai multe componente electronice si sa gestionez interactiunea dintre ele prin cod.
- A fost o experienta utila care a combinat teoria cu practica, acum stiu cum sa implementez un proiect de la 0 cu mai multe componente cum ar fi buzzer, butoane, led-uri etc.
- Consideram ca proiectul a fost un succes si ca mi-a oferit o baza solida pentru viitoare proiecte embedded.

Download

Proiectul: sketch_pm.zip

Bibliografie/Resurse

Suport de laborator: https://ocw.cs.pub.ro/courses/pm/lab/lab3-2023-2024

Ajutor PWM: https://wolles-elektronikkiste.de/en/timer-and-pwm-part-1-8-bit-timer0-2

Suport legare + programare buton cu pull-down:
https://docs.arduino.cc/built-in-examples/digital/Button/

Info pentru buzzer: https://balau82.wordpress.com/2014/10/15/using-a-buzzer-with-arduino-in-pure-c/