Whack-a-Mole Game

Introducere

Ce face: Proiectul reprezintă o adaptare electronică a clasicului joc arcade „Whack-a-Mole”. În loc de cârtițe mecanice, jocul folosește butoane Arcade iluminate. Microcontrolerul va aprinde aleatoriu LED-ul unuia dintre butoane, iar jucătorul trebuie să apese butonul respectiv cât mai repede posibil.

Scopul lui: Scopul principal este testarea și îmbunătățirea timpului de reacție al jucătorului. Din punct de vedere academic, scopul este implementarea unui sistem interactiv complet (bare-metal) care implică citirea de intrări digitale și analogice, generarea de semnale de ieșire multiplexate și utilizarea timerelor.

Ideea de la care am pornit: Ideea a pornit de la dorința de a recrea sentimentul și dinamica jocurilor mecanice retro folosind componente electronice moderne. Am vrut un proiect care să fie nu doar o demonstrație tehnică, ci și un produs final distractiv și interactiv.

Utilitate: Pentru mine, este o oportunitate excelentă de a aprofunda lucrul cu registrele ATmega328p, timerele, întreruperile și optimizarea pinilor folosind un Shift Register. Pentru utilizatori, este un joc antrenant, perfect pentru pauze, care stimulează reflexele.

Descriere generală

Sistemul este centrat în jurul microcontrolerului ATmega328p (pe platforma Arduino UNO). Acesta preia date din lumea fizică, le procesează și oferă feedback vizual și sonor.

Interacțiunea modulelor:

- **Unitatea de procesare:** Placa de dezvoltare generează secvența aleatoare de aprindere a LED-urilor, cronometrează timpul de reacție și calculează scorul curent.
- **Modulul de Input (Senzorii):** Este compus din cele 4 butoane Arcade (microswitch-uri) și un potențiometru liniar. Potențiometrul este citit prin convertorul analog-digital (ADC) înainte de începerea jocului pentru a stabili nivelul de dificultate (timpul pe care jucătorul îl are la dispoziție pentru a apăsa butonul).
- **Modulul de Output Vizual (Afișajul):** Include LED-urile integrate în butoanele Arcade (care indică ținta) și un afișaj cu 7 segmente și 4 digiți (care afișează scorul). Pentru a economisi pinii limitați ai microcontrolerului, afișajul cu 7 segmente este controlat prin intermediul unui Shift Register (74HC595).
- **Modulul de Output Sonor:** Un buzzer activ oferă feedback acustic imediat (ex: un ton scurt pentru o lovitură reușită, un ton diferit pentru o lovitură ratată sau expirarea timpului).

Planificare și Ipoteză

Ipoteza proiectului

Credem că implementarea unei arhitecturi software non-blocante (zero-delay) bazată pe un automat cu stări finite (FSM) și pe un timer hardware dedicat pentru `millis()` va îmbunătăți calitatea multiplexării vizuale și va garanta un timp de scanare a inputurilor (butoane) sub 5ms. Presupunem că, deoarece microcontrolerul nu va mai irosi cicluri de ceas în bucle de așteptare blocante, acesta va permite actualizarea display-ului și citirea jucătorului în mod simultan, fără efect de flicker.

Planificarea sarcinilor (Gantt)

Următorul tabel descrie activitățile principale desfășurate pe parcursul celor 4 săptămâni alocate proiectului.

Săptămâna	Activitate asumată
Săptămâna 1	**Documentare și Arhitectură:** Alegerea componentelor, studierea datasheet-urilor (ATmega328P, 74HC595) și definirea ipotezei proiectului, redactarea documentației pe OCW.
Săptămâna 2	**Hardware Design:** Proiectarea schemei electrice, asamblarea componentelor pe breadboard și testarea conexiunilor la pini (LED-uri, butoane, display).
Săptămâna 3	**Software Design:** Implementarea driverelor bare-metal (Timer0 pentru millis, Timer1 pentru generare ton, ADC, EEPROM, SPI/ShiftOut) și scrierea logicii FSM.
Săptămâna 4	**PM Fair & Profilare:** Profilarea codului, ajustarea dificultății, curățarea finală a repository-ului de Git și pregătirea prezentării.

Hardware Design

Listă de piese

- Microcontroler ATmega328p (Datasheet Microchip) - integrat pe o placă de dezvoltare compatibilă Arduino UNO R3.
- 4 x Buton Arcade 24 mm - Verde (Datasheet Adafruit / Mouser) - cu microswitch și LED intern alimentat la 5V.
- 1 x Buzzer Activ 3V/5V (Datasheet) - controlat prin semnal digital.
- 1 x Potențiometru liniar 10kΩ tip WH148 (Datasheet) - pentru modulul de divizor de tensiune (ADC).
- Kit de componente electronice standard, din care s-au utilizat:
  - 1 x Shift Register 74HC595 (Datasheet Texas Instruments) - convertor SIPO pe 8 biți.
  - 1 x Afișaj 4-Digit 7-Segmente Catod Comun (Datasheet referință) - multiplexat.

- 9 x Rezistențe de limitare a curentului (220Ω / 330Ω).
- 1 x Breadboard 830 puncte.
- Fire de conexiune tip Dupont (Tată-Tată și Mamă-Tată).

Software Design

Mediu de dezvoltare

Cod scris în **C bare-metal** pentru ATmega328P, fără framework Arduino. Toolchain:

- **avr-gcc** 7.3.0 — compilator
- **avr-libc** 2.0.0 — pentru macro-uri de registre din `<avr/io.h>`, `<avr/interrupt.h>`, `<util/delay.h>`
- **avrdude** — flash prin bootloader-ul Arduino

```
avr-gcc -mmcu=atmega328p -DF_CPU=16000000UL -Os -Wall whack_a_led.c -o
whack.elf
avr-objcopy -O ihex -R .eeprom whack.elf whack.hex
avrdude -p atmega328p -c arduino -P /dev/ttyACM0 -b 115200 -U
flash:w:whack.hex
```

Binar rezultat: **~4.8 KB flash + 150 bytes RAM**.

Librării 3rd-party

Niciuna. Toate perifericele sunt programate direct la nivel de registre. Singurele include-uri sunt header-ele standard avr-libc pentru definiţiile registrelor şi macro-urile de întreruperi.

Algoritmi şi structuri

- **Automat cu stări finite** — 3 stări: IDLE, PLAYING, GAME_OVER, cu tranziţii într-un `switch` din main loop.
- **Multiplexare display** — cele 4 cifre împart 8 linii de segmente prin shift register 74HC595. Hold de 2 ms per cifră → refresh ~125 Hz. Anti-ghosting prin stingerea tuturor digiților înainte de încărcarea segmentelor.
- **Generare ton** — Timer1 în mod CTC; pentru că PD6 nu e pin OC, comutarea se face manual dintr-un ISR. Frecvența calculată cu `OCR1A = F_CPU / (2 × prescaler × freq) - 1`.
- **millis() bare-metal** — Timer0 CTC cu întrerupere la fiecare 1 ms care incrementează un contor `uint32_t`; citirea protejată cu `cli/sei` pentru atomicitate.
- **ADC** — citire potențiometru (canal 4) și seed pentru PRNG (canal 5, pin neconectat).
- **PRNG xorshift32** — generator pseudo-aleator rapid, seed-uit din zgomotul ADC.
- **EEPROM** — persistență high score (2 bytes + magic byte pentru detectarea chip-ului virgin).

Secvența EEMPE→EEPE protejată cu `cli/sei`.
- **Delay non-blocant** — toate așteptările folosesc o buclă bazată pe `millis()` care continuă să refresheze display-ul, eliminând pâlpâirea în timpul melodiilor şi animaţiilor.

Metrici și Profilarea Codului

Pentru a valida ipoteza și a garanta un gameplay fluid, am urmărit următoarele metrici de performanță:

- **Latența de Input (Polling Rate):** Ținta asumată a fost un timp de răspuns sub 5ms. La profilarea codului (analizând funcția `display_digits`), am observat că fiecare din cele 4 cifre necesită un `_delay_us(2000)` pentru a garanta o luminozitate optimă a segmentelor. Astfel, un ciclu complet de reîmprospătare durează minimum 8ms, plus overhead-ul shiftării pe 8 biți. Așadar, butoanele sunt scanate la aproximativ 8.5 - 9ms. Deși depășește pragul ipotetic de 5ms, valoarea este suficient de mică pentru a preveni pierderea oricărui input generat de reflexele umane.
- **Frecvența de Refresh (Display):** Un timp de execuție a buclei de ~8.5ms se traduce într-o frecvență de refresh de aproximativ **115-120 Hz**. Aceasta metrică a atins perfect ținta, rezultând într-un afișaj luminos, clar și absolut lipsit de pâlpâire (flicker).
- **Amprenta pe Memorie:** Executabilul final bare-metal folosește doar ~4.8 KB din memoria Flash și 150 bytes de memorie RAM, demonstrând eficiența optimizării la nivel de registre.

Proiectul a fost o oportunitate excelentă de a aprofunda lucrul **bare-metal** cu ATmega328P. Renunţarea la framework-ul Arduino a oferit control complet asupra perifericelor. Cele mai utile decizii s-au dovedit a fi structurarea logicii ca automat cu stări finite și multiplexarea non-blocantă. Cele mai dificile obstacole au fost:

- Generarea de tonuri pe PD6 (deoarece nu este un pin OC al Timer1, a necesitat crearea unei întreruperi manuale `TIMER1_COMPA_vect`).
- Atomicitatea citirii contorului `millis_count` pe arhitectura AVR de 8 biți (rezolvată folosind blocul `cli()/sei()`).
- Scrierea în EEPROM (respectarea timpului critic de 4 cicluri de ceas pentru secvența EEMPE → EEPE).

Surse şi funcţii implementate

Cod organizat într-un singur fişier (`whack_a_led.c`, ~810 linii) cu următoarele grupuri funcţionale:

- **Drivere low-level**: `timer0_init`, `millis`, `tone_start/stop`, `adc_read`, `eeprom_read/write_byte_raw`, `prng_next`
- **Display**: `shift_out_msb`, `write_segments`, `display_digits`, `display_score`, `display_idle`, `display_game_over`, `compute_dp_mask`
- **Audio**: `tone_blocking`, `play_melody`
- **Secvenţe de joc**: `display_countdown`, `start_new_game`, `pick_next_led`, `trigger_game_over`, `score_roll_animation`, `level_up_sequence`
- **Input**: `get_pressed_button`
- **Persistenţă**: `load_high_score`, `save_high_score`
- **Main loop**: inițializare periferice + `while(1)` cu refresh display + FSM

Elementul de Noutate

Elementul principal de noutate și originalitate (5% din evaluare) constă în **sistemul hibrid și dinamic de ajustare a dificultății**, secondat de inovații vizuale pe afișajul 7-segmente:

- **Dificultatea Scalabilă Hibrid:** Fereastra de timp pe care o are jucătorul la dispoziție nu este un simplu parametru static stabilit de potențiometru. Potențiometrul setează doar pragul “de bază” (între 3s și 1s). Peste acesta se aplică un algoritm de “Accelerare Progresivă”: pe măsură ce scorul crește, din timpul total disponibil se scad automat câte 5 ms per punct acumulat, forțând o escaladare organică a dificultății. Timpul este totuși plafonat hardware (hard-limit la 600ms) pentru ca jocul să rămână fizic posibil.
- **Punctaj Proporțional:** Jucătorul nu primește mereu 1 punct per lovitură. Logica este împărțită pe trepte de reacție (se acordă 10, 5 sau 1 punct în funcție de plasarea timpului de reacție în treimea corespunzătoare a ferestrei alocate), recompensând reflexele extreme.
- **Bara de Progres pe Display (DP HUD):** Pentru a indica vizual cât timp mai are jucătorul înainte să expire runda curentă fără să întrerupă afișarea scorului, am folosit inteligent cele 4 puncte zecimale (DP) de pe afișaj. Acestea se sting rând pe rând pe măsură ce fereastra de timp trece (1111 → 0111 → 0011 → 0001), servind drept indicator vizual (Timer Bar).

Rezultate Obţinute

Proiectul a fost finalizat cu succes în cele 4 săptămâni alocate. Sistemul funcţionează stabil ca un joc arcade complet, cu input multiplu (4 butoane cu pull-up intern + potenţiometru), output vizual multiplexat şi feedback sonor (melodii redate printr-un ISR manual pe Timer1).

Concluzii și Validarea Ipotezei

Validarea Ipotezei: Ipoteza inițială a fost validată **parțial**, însă cu un rezultat practic excelent. Presupunerea că timpul de polling va scădea sub 5ms s-a dovedit incorectă din cauza necesității hardware de a menține cifra aprinsă timp de 2ms pentru persistența retinei (rezultând într-o buclă de minim 8ms). Cu toate acestea, nucleul ipotezei s-a confirmat pe deplin: arhitectura cu FSM și eliminarea totală a funcțiilor blocante (înlocuite cu funcția custom `custom_delay` bazată pe `millis()`) a permis ca sistemul să actualizeze display-ul și să scaneze inputurile simultan, fără nicio blocare sau ghosting.

Download

Codul sursă complet şi schemele proiectului sunt disponibile pe GitHub: Whack-a-Mole-PM

Repository-ul conţine următoarele fișiere:

- `whack_a_led.ino` — codul sursă al jocului
- `pm_schem.png` — schema electrică a proiectului
- `README.md` — documentația și instrucțiunile proiectului
- `LICENSE` — licența MIT a proiectului

Bibliografie/Resurse

Resurse Hardware

- **ATmega328P Datasheet** (Microchip/Atmel) — referinţa oficială pentru registrele perifericelor: Timer/Counter, ADC, EEPROM, USART, întreruperi. datasheet PDF
- **74HC595 Datasheet** (Texas Instruments / NXP) — shift register 8-bit cu latch, folosit pentru controlul segmentelor display-ului. datasheet PDF
- **KH5461AB Datasheet** — display 7-segmente 4 cifre, common cathode. Conţine pinout-ul şi specificaţiile electrice ale segmentelor.
- **Arduino UNO R3 Schematic** — pentru verificarea conexiunilor pinilor expuşi pe board şi a circuitului de reset.

Resurse Software

- **AVR Libc Reference Manual** — documentaţia oficială pentru avr-libc, în special pentru macro-urile din `<avr/io.h>`, `<avr/interrupt.h>` şi funcţiile din `<util/delay.h>`. nongnu.org/avr-libc
- **AVR-GCC Documentation** — opţiuni de compilare specifice arhitecturii AVR (`-mmcu`, `-Os`, `-DF_CPU`).
- **AVRDUDE User Manual** — opţiuni pentru flash-area HEX-ului prin bootloader. nongnu.org/avrdude

Inspiraţie & resurse de muzică

- **Mario Bros Game Over (Koji Kondo, 1985)** — transcrierea melodiei NES pentru buzzer, adaptată din proiecte open-source comunitare de Arduino.
- **Note frequency table** — tabela standard de frecvenţe pentru notele muzicale (C4 = 262 Hz, A4 = 440 Hz, etc.).