Arduino Pong Game

Introducere

Proiectul presupune ceva destul de rudimentar : **Pong** - joc video clasic într-un format **Arduino** și care necesită două persoane pentru a putea fi jucat . Proiectul utilizează un ecran OLED , în care se va afișa scorul fiecărui jucător, mingea și cele două palete.

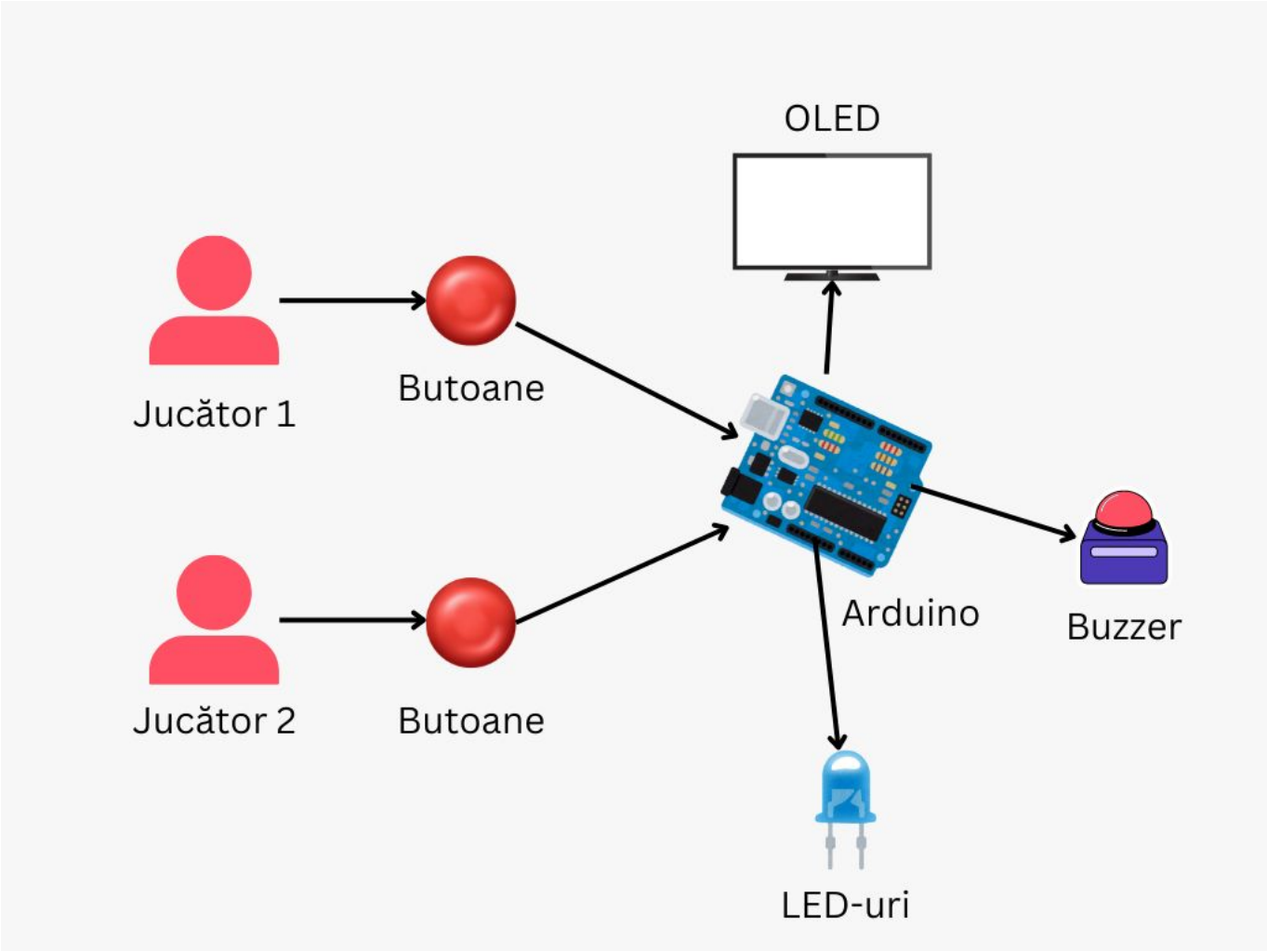
Scopul acestui proiect este de a oferi o interfață interactivă prin care utilizatorii să poată explora conceptele de bază ale programării și electronicii.

Ideea proiectului a venit din dorința de a recrea un joc video clasic într-un format nou și accesibil, combinând nostalgia cu tehnologia modernă.

Descriere generală

- Fiecare jucător are câte 2 butoane , pentru a controla paleta în sus sau în jos .
- Buzzer-ul va scoate sunete în momentul în care mingea atinge paleta sau atunci când este înscris un punct .
- LED-ul verde se va aprinde pentru jucătorul care a înscris punctul iar cel roșu pentru pierzător.
- Cel care are 7 puncte este câștigător iar LED-urile vor pâlpâi de câteva ori .

Schemă bloc

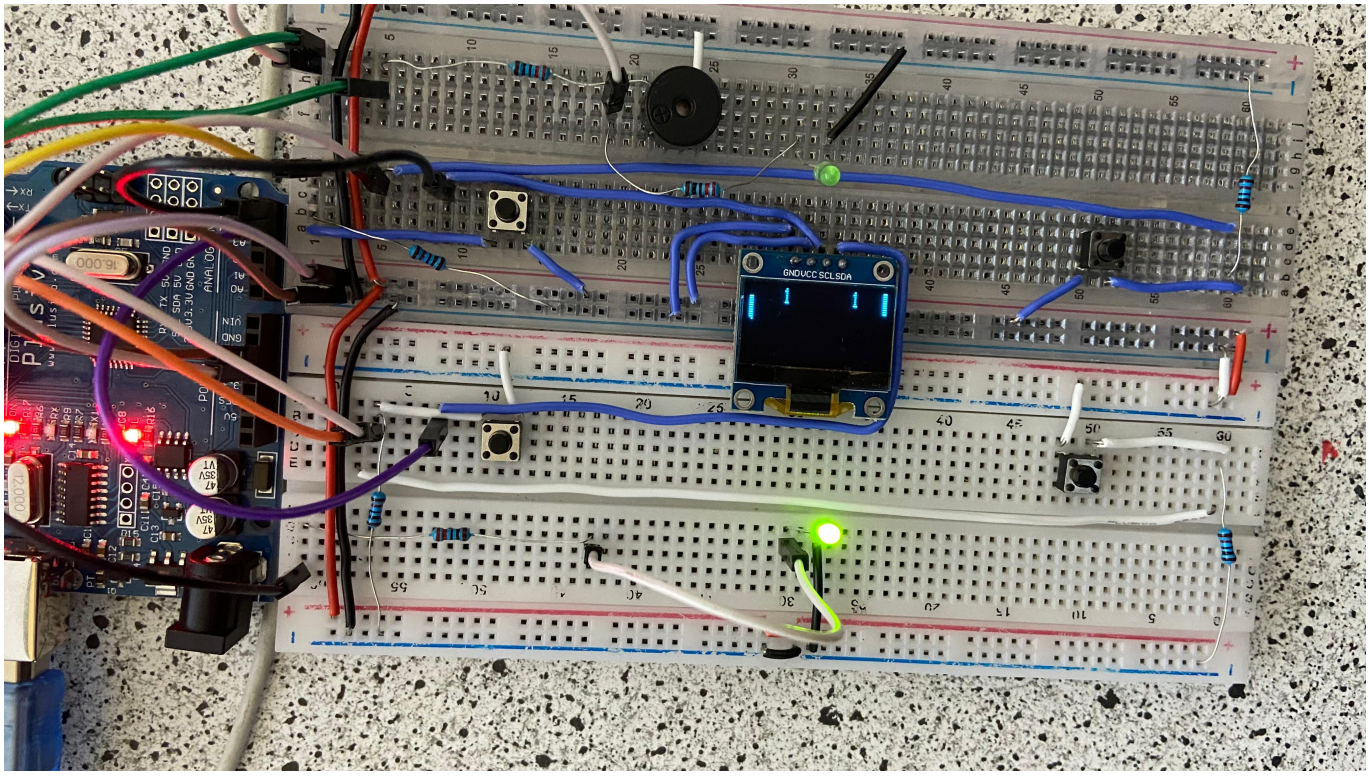


Hardware Design

Piese folosite:

- Arduino UNO
- Buzzer
- OLED I2C 128×64
- 4 butoane
- 2/4 LED-uri

În schematic, OLED-ul era prea mare pentru a fi pus pe breadboard. în realitate , eu l-am lipit de breadboard. Valabil și pentru buzzer.



Piesele sunt funcționale și deja am început să lucrez pentru Software design.

Software Design

Dezvoltarea codului am realizat-o in Arduino IDE

Pe partea de software am folosit bibliotecile:

1. Wire.h: librărie standard pentru comunicarea I2C pe plăcile Arduino.
2. Adafruit_GFX.h: librărie grafică utilizată pentru a desena forme și texte pe display-uri.
3. Adafruit_SSD1306.h: ibrărie specifică pentru controlul display-ului OLED SSD1306 prin intermediul comunicării I2C.

Algoritmi și Structuri Implementate:

Setări Inițiale:

- Configurarea pinilor pentru butoane și LED-uri.
- Inițializarea comunicării I2C cu display-ul OLED.
- Afișarea ecranului de start cu un splash screen “Arduino Pong”.

Controlul Jocului:

- Butoanele sunt utilizate pentru a controla mișcarea paletelor stânga și dreapta.
- Algoritm pentru actualizarea pozițiilor paletelor bazat pe intrările butoanelor.
- Algoritm pentru actualizarea poziției mingii și detectarea coliziunilor cu paletele și marginile

ecranului.
- Algoritm pentru gestionarea punctajului și resetarea jocului atunci când un jucător câștigă.

Desenarea Jocului:

- Desenarea mingii, paletelor și scorurilor pe display-ul OLED folosind funcțiile oferite de Adafruit_GFX și Adafruit_SSD1306.

Gestionarea Vitezei și Poziției Mingi:

- Funcții pentru ajustarea vitezei mingii după fiecare coliziune cu o paletă sau cu marginile ecranului.
- Funcții pentru resetarea poziției mingii și ajustarea dimensiunilor paletelor după fiecare punct marcat.

Anunțarea Câștigătorului:

- Funcție pentru anunțarea câștigătorului prin clipirea unui LED specific de mai multe ori.

Surse și Funcții Implementate:

Setup și Loop:

```
void setup() {
    pinMode(l_up_button, INPUT);
    pinMode(l_down_button, INPUT);
    pinMode(r_up_button, INPUT);
    pinMode(r_down_button, INPUT);
    pinMode(led1, OUTPUT);
    pinMode(led2, OUTPUT);
    Serial.begin(9600);
    randomSeed(analogRead(0));

    display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
    display.display();
    display.clearDisplay();

    display.setTextSize(1);
    display.setTextColor(WHITE);
    display.setCursor(display.width() / 2 - 20, 0);
    display.println("Arduino");
    display.setCursor(display.width() / 2 - 10, 8);
    display.println("Pong");
    display.display();
    delay(2000);
}

void loop() {
    unsigned long currentMillis = millis();
```

```
    if (currentMillis - previousMillis >= interval) {
        previousMillis = currentMillis;
        updateGame();
        drawGame();
    }
}
```

Funcții pentru Actualizarea Jocului și Desenare:

```
void updateGame() {
    if (l_score >= max_score) {
        announce_winner(led1);
        reset_game();
        return;
    } else if (r_score >= max_score) {
        announce_winner(led2);
        reset_game();
        return;
    }

    if (digitalRead(l_up_button) && l_pos > 0) { l_pos -= 2; }
    else if (digitalRead(l_down_button) && l_pos < (y_pixels -
l_paddle_height)) { l_pos += 2; }

    if (digitalRead(r_up_button) && r_pos > 0) { r_pos -= 2; }
    else if (digitalRead(r_down_button) && r_pos < (y_pixels -
r_paddle_height)) { r_pos += 2; }

    x_pos += x_vel;
    y_pos += y_vel;

    if (y_pos >= y_pixels - 1 || y_pos <= 0) { y_vel = -y_vel; adjust_speed();
}
    if (ball_on_right_paddle()) {
        x_vel = -x_vel;
        adjust_speed();
        tone(speakerPin, 300, 100);
    } else if (ball_on_left_paddle()) {
        x_vel = -x_vel;
        adjust_speed();
        tone(speakerPin, 250, 100);
    }

    if (x_pos >= x_pixels - 1) {
        ball_reset(false);
        l_score += 1;
        tone(speakerPin, 50, 100);
        digitalWrite(led1, HIGH);
```

```
        delay(500);
        digitalWrite(led1, LOW);
    } else if (x_pos <= 0) {
        ball_reset(true);
        r_score += 1;
        tone(speakerPin, 50, 100);
        digitalWrite(led2, HIGH);
        delay(500);
        digitalWrite(led2, LOW);
    }
}

void drawGame() {
    display.clearDisplay();
    display.drawPixel(x_pos, y_pos, WHITE);
    display.fillRect(0, l_pos, paddle_width, l_paddle_height, WHITE);
    display.fillRect(x_pixels - paddle_width, r_pos, paddle_width,
r_paddle_height, WHITE);

    display.setTextSize(1);
    display.setTextColor(WHITE);
    display.setCursor(display.width() / 4, 0);
    display.println(l_score);
    display.setCursor(display.width() * 3 / 4, 0);
    display.println(r_score);

    display.display();
}
```

Funcții de Utilitate:

```
bool ball_on_right_paddle() {
    return ((x_pos + x_vel) >= x_pixels - paddle_width - 1 && y_pos >= r_pos
&& y_pos <= (r_pos + r_paddle_height));
}

bool ball_on_left_paddle() {
    return ((x_pos + x_vel) <= paddle_width - 1 && y_pos >= l_pos && y_pos
<= (l_pos + l_paddle_height));
}

void ball_reset(bool left) {
    y_pos = random(display.height());
    y_vel = random(2) ? 3 : -3;

    if (left) {
        x_vel = 2;
        x_pos = paddle_width;
```

```
    } else {
        x_vel = -2;
        x_pos = display.width() - paddle_width - 1;
    }

    adjust_paddles();
}

void adjust_speed() {
    if (x_vel > 0 && x_vel < max_vel) {
        x_vel += 1;
    } else if (x_vel < 0 && x_vel > -max_vel) {
        x_vel -= 1;
    }

    if (y_vel > 0 && y_vel < max_vel) {
        y_vel += 1;
    } else if (y_vel < 0 && y_vel > -max_vel) {
        y_vel -= 1;
    }
}

void adjust_paddles() {
    int new_height = random(5, 15);
    int mode = random(3);

    if (mode == 0) {
        l_paddle_height = new_height;
        r_paddle_height = new_height;
    } else if (mode == 1) {
        l_paddle_height = new_height;
        r_paddle_height = paddle_height;
    } else {
        l_paddle_height = paddle_height;
        r_paddle_height = new_height;
    }
}

void announce_winner(int winner_led) {
    for (int i = 0; i < 6; i++) {
        digitalWrite(winner_led, HIGH);
        delay(250);
        digitalWrite(winner_led, LOW);
        delay(250);
    }
}

void reset_game() {
    l_score = 0;
    r_score = 0;
    l_paddle_height = 10;
```

```
    r_paddle_height = 10;
    ball_reset(random(2));
}
```

Download

pong.zip

Jurnal

- 28.04.2023 - Am finalizat ideea de bază a jocului, am creat flow-ul și am ales piesele potrivite pentru circuit
- 11.05.2023 - Am comandat piesele pentru proiect.
- 12.05.2023 - Mi-au venit piesele pentru hardware.
- 15.05.2023 - Am început să lucrez la hardware , iar o parte din breadboard nu merge. Laborantul mi-a spus ca unul din breadboard-uri era mai vechi și trebuiau legate părțile între ele , după cum se poate observa în poză .
- 16.05.2023 - Deja terminasem cu hardware-ul , așa că m-am apucat din timp de software, știind că acolo va fi mai greu.
- 23.05.2023 - Aproape termin cu soft-ul , momentan întâmpin niște bug-uri , bila merge prea lent , paletele prea rapid iar jocul e prea ușor.
- 25.05.2023 - Ajung într-un punct în care codul este suficient de ok , am mărit bila și velocitatea ei , am făcut o funcție de reset , am făcut ca paletele să aibă dimensiuni random atunci când un jucator dă gol , pentru a fi mai provocator jocul.

Concluzii

A fost un proiect destul de interesant, mi-a plăcut să lucrez la el , exceptând momentele în care a trebuit să fac debug pentru chestii simple.

Bibliografie

- https://www.arduino.cc/reference/en/libraries/adafruit-gfx-library/
- https://www.arduino.cc/reference/en/language/functions/communication/wire/
- https://github.com/adafruit/Adafruit_SSD1306