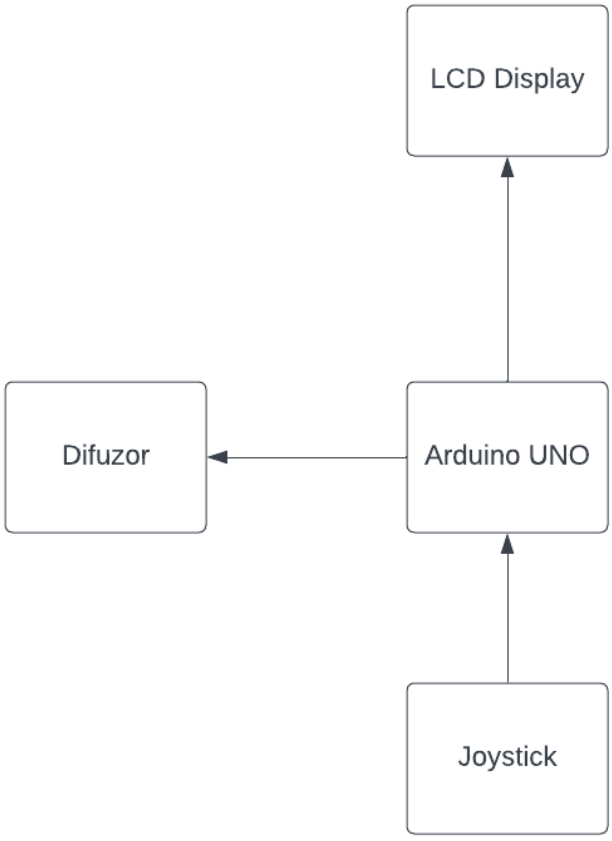
# Snake Game

## Introducere

Jocul clasic de pe telefoanele vechi, Snake, recreat pe un Arduino UNO, folosind un buzzer pentru a anunta deplasarea sarpelui. De asemenea, fata de varianta de pe telefonul, pentru input, este folosit un joystick. }

## Descriere generală

Schema bloc:



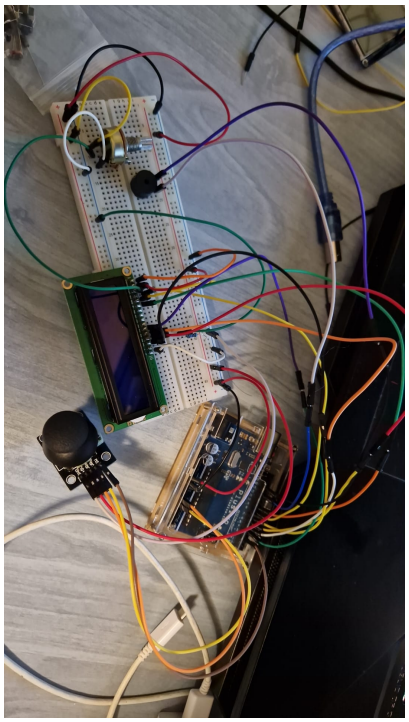

## Hardware Design

Componente folosite:

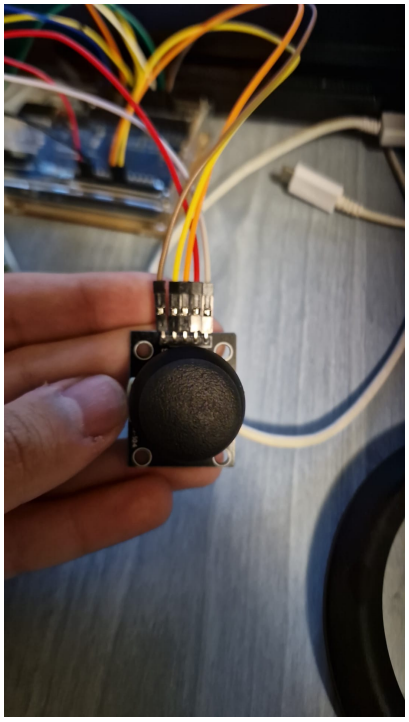
- Arduino UNO
- Joystick
- LCD Display 16×2
- Buzzer
- Potentiometru

In timpul dezvoltarii proiectului, am avut de-a face cu probleme tehnice, astfel ca LCD-ul de tip TFT comandat se aprindea, insa nu puteam transmite comenzi catre acesta. Datorita acestei probleme, am fost nevoit sa modific design-ul si componentele folosite pe ultima suta de metrii.

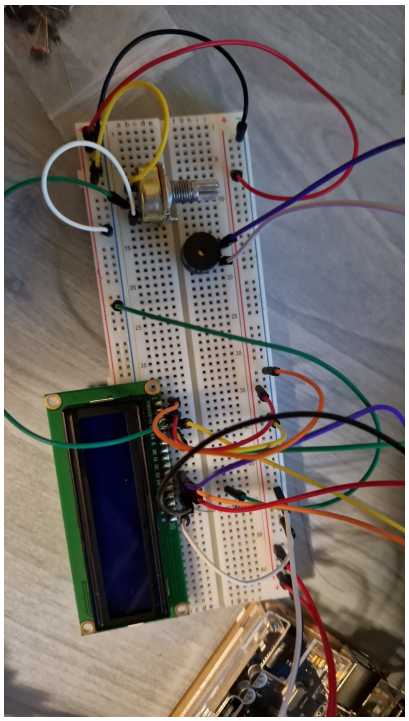
Ansamblul Arduino, LCD Display, Joystick si Difuzor.



Joystick



Breadboard



## Software Design

Mediu de dezvoltare: Arduino IDE Librarii: pitch.h, LiquidCrystal

Am incercat sa recreez in interiorul mediului de dezvoltare Arduino. Am avut ca si tel sa mentin functiile principale ale jocului clasic. Am dorit sa adaug si cateva functii noi precum un soundtrack si un buton cu abilitatea de a micsora sarpele prin injumatatirea punctelor, insa, datorita constrangerilor de timp, nu am reusit sa implementez aceste functii.

Mai jos am adaugat cateva functii folosite in cod:

Algoritmi:

Functie pentru deplasarea sarpelui, alcatuita din doua metode:

```
void moveHead()
{
  switch(head->dir) // 1 step in direction
  {
    case 0: head->row--; break;
    case 1: head->row++; break;
    case 2: head->column++; break;
    case 3: head->column--; break;
    default : break;
  }
  if (head->column >= 80) head->column = 0;
  if (head->column < 0) head->column = 79;
  if (head->row >= 16) head->row = 0;
  if (head->row < 0) head->row = 15;

  if (levelz[selectedLevel][head->row / 8][head->column / 5]) gameOver =
true; // wall collision check

  part *p;
  p = tail;
  while (p != head && !gameOver) // self collision
  {
    if (p->row == head->row && p->column == head->column) gameOver = true;
    p = p->next;
  }
  if (gameOver)
    gameOverFunction();
  else
  {
  x[head->row][head->column] = true;

  if (head->row == pr && head->column == pc) // point pickup check
  {
    collected++;
    if (gameSpeed < 25) gameSpeed+=3;
    newPoint();
  }
  }
}

void moveAll()
{
  part *p;
  p = tail;
  x[p->row][p->column] = false;
```

```
  while (p->next != NULL)
  {
    p->row = p->next->row;
    p->column = p->next->column;
    p->dir = p->next->dir;
    p = p->next;
  }
  moveHead();
}
```

Game End Screen:

```
void gameOverFunction()
{
  delay(1000);
  lcd.clear();
  freeList();
  lcd.setCursor(3,0);
  lcd.print("Game Over!");
  lcd.setCursor(4,1);
  lcd.print("Score: ");
  lcd.print(collected);
  delay(1000);
}
```

Start Screen:

```
void startF()
{
  gameOver = false;
  gameStarted = false;
  selectedLevel = 1;

  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Select level: 1");
  for(i=0;i<8;i++)
  {
    lcd.createChar(i,mySnake[i]);
    lcd.setCursor(i+4,1);
    lcd.write(byte(i));
  }
  collected = 0;
  gameSpeed = 15;
  createSnake(3);
  time = 0;
}
```

Transformarea la Joystick input in Game Input:

```
int joystick_translate(){
  if (analogRead(VRx) <= 200 && analogRead(VRy)>400 && analogRead(VRy) <
600){
    translated_input = 0; //up
  }
  if (analogRead(VRx) >= 900 && analogRead(VRy)>400 && analogRead(VRy) <
600){
    translated_input = 1; //down
  }
  if (analogRead(VRy) <= 200 && analogRead(VRx)>400 && analogRead(VRx) <
600){
    translated_input = 2; //right
  }
  if (analogRead(VRy) >= 900 && analogRead(VRx)>400 && analogRead(VRx) <
600){
    translated_input = 3; //left
  }
  if (SW_state==0){
    translated_input = 4; //button press
  }

delay(500);
return translated_input;
}
```

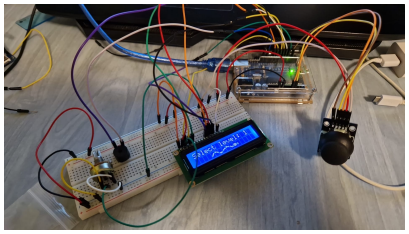
Redarea unui sunet la fiecare deplasare, aflat la inceput de loop:

```
for (int thisNote = 0; thisNote < 1; thisNote++) {
    tone(12, melody[thisNote], duration);

    delay(500);
  }
```

## Rezultate Obţinute

Rezultatele obtinute au fost un mic joc de snake interesant. Desi am fost nevoit sa schimb design-ul pe ultima suta de metrii datorita dificultatilor tehnice mentinoate in partea de hardware design, am reusit sa fac schimbarile necesare in timp util, insa codul mai trebuie optimizat, deoarece foloseste foarte multa memorie, iar codul intampina memory overload cand 2 sau mai multe puncte sunt acumulate.



## Concluzii

A fost un proiect interesant care mi-a dezvoltat gandirea si mi-a indicat diverse greseli in abordarea problemelor si anumite probleme de organizare sau tehnice. Descoperirea defectului de la display TFT LCD a creat o stiuatie stresanta, astfel ca, asa cum am mentionat si in sectiunea de hardware, am fost nevoit sa schimb design-ul proiectului pe ultima suta de metrii. Dezvoltarea codului a fost interesant si provocator, mai ales din cauza dificultatilor tehnice care m-au obligat sa rescriu codul, astfel ca am avut multe de invatat si de descoperit datorita contruirii si dezvoltarii acestui proiect.

## Download

Codul si librarii folosite:

snake.zip

## Bibliografie/Resurse

Librarie melodii buzzer
Utilizare LCD display
Utilizare Joystick
Laboratorul 4: ADC
Laboratorul 3: Timere, PMW
Laboratorul 1: UART

Export to PDF