Snake - Stanca Adelin-Nicolae 331CA

Stanca Adelin-Nicolae 331CA

Introducere

Prezentarea pe scurt a proiectului:

- Proiectul consta in implementarea jocului Snake, pe LCD TFT 2.4 inch. Scopul jucatorului este sa isi mentina playerul cat mai mult timp in viata si sa acumuleze un scor cat mai mare, fara a se lovi de pereti si fara a se lovi de sine insusi
- Am ales acest proiect deoarce consider ca ma va ajuta sa aplic cunostintele invatate la laborator.
- Ideea proiectului a venit din dorinta de a implementa un joc usor si cu un posibil final fericit pe un dispozitiv cu care se pot juca si copiii fara grija.

Descriere generală

Jocul debuteaza in momentul in care este apasat de catre pen ecranul display-ului LCD TFT. La inceput este generat un punct random pe display, dupa care se va genera playerul care va urmari miscarea penului (sau ultima sa miscare). Scopul este de a atinge punctul random, moment in care se incrementeaza dimensiunea playerului si se genereaza un alt punct. Jocul se incheie in momentul in care jucatorul se loveste de sine insusi sau in momentul in care penul il dirijeaza catre un perete, moment in care se va afisa pe LCD scorul final, precum si scorul maxim. Pentru a reseta jocul, la 10 secunde de afisare a scorului, jocul se reinitializeaza, iar jucatorul trebuie doar sa atinga o data ecranul pentru a reincepe jocul.

Hardware Design

Lista componentelor:

- Breadboard
- Arduino Uno
- LCD TFT 2.4''
- Pini mama-mama
- Cablu USB

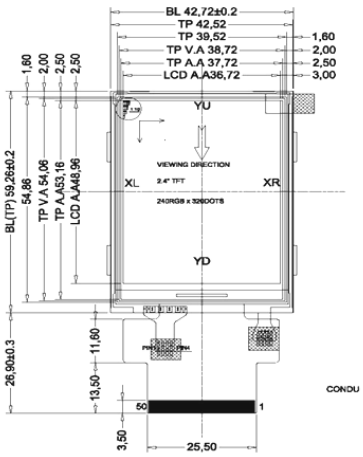
- Fire
- Pen

Ecranul LCD TFT 2.4 inch

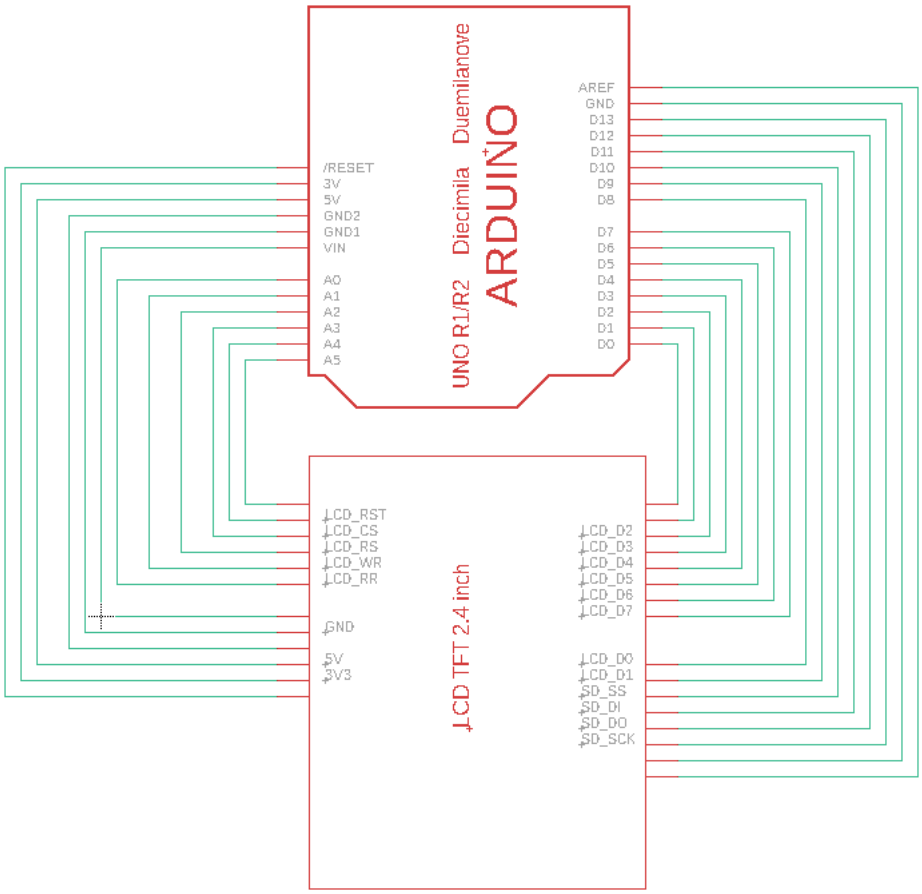
Pin Name	Category
LCD_RST	LCD Command Pins
LCD_CS	
LCD_RS	
LCS_WR	
LCD_RD	
LCD_D0	LCD Data Pins
LCD_D1	
LCD_D2	
LCD_D3	
LCD_D4	
LCD_D5	
LCD_D6	
LCD_D7	
SD_SS	SD Card Data Pins
SD_DI	
SD_DO	
SD_SCK	
GND	Power Pins
5V	
3.3V	

Am reusit in cele din urma sa lipesc cele 2 componente principale, fapt ce m-a ajutat ca la final sa nu mai am nevoie de majoritatea componentelor auxiliare, precum firele si placuta.

Diagrama 2D a LCD TFT 2.4 inch



Schema electrica



Software Design

Explicarea codului

Toata functionalitatea este implementata in cadrul fisierului **inceput.ino** in cadrul caruia am incercat sa conserv o abordare cat se poate de modularizata, iar codul sa fie cat se poate de lizibil. Jocul debuteaza cu functia de **setup** care practic afiseaza la LCD numele proiectului pentru pagina de inceput, dupa care se initializeaza bara de Loading implementata tot in aceeasi functie, insa care va aparea separat pe ecran.

```
void setup() {
  tft.reset();
  tft.begin(0x9341);
  tft.setRotation(1);
  tft.fillScreen(BLACK);
  tft.setCursor(55, 80);
  tft.setTextSize(3);
  tft.setTextColor(GREEN);
  tft.println("Snake Game");
  tft.setCursor(85, 120);
  tft.setTextSize(1);
```

```
  tft.setTextColor(WHITE);
  tft.println("Proiect realizat de");
  tft.setCursor(70, 150);
  tft.setTextSize(2);
  tft.setTextColor(BLUE);
  tft.println("Adelin Stanca");
  delay(1000);

  tft.fillScreen(BLACK);
  tft.setCursor(95, 110);
  tft.setTextColor(GREEN);
  tft.println("Loading...");
  tft.drawRect(60, 140, 200, 20, GREEN);
  for (int i = 0; i < 100; i++) {
    tft.fillRect(60, 140, i * 2, 20, GREEN);
    if (i == 50) {
      delay(150);
    }
    delay(50);
  }
  tft.fillScreen(color);
}
```

Exista 4 stari principale pe care jocul meu le traverseaza, acestea sunt setupGame, pausedGame, inGame si gameLost. Fiecarei stari ii corespunde o functie principala la care se adauga altele auxiliare. Jocul debuteaza din starea gameSetup. In cadrul acelei stari, practic se genereaza o pozitie random pentru jucator, apoi o pozitie aleatoare pentru 'hrana' sarpelui, tinandu-se cont de pozitiile ocupate deja de sarpe. Dupa aceste lucruri, se trece automat in starea pausedGame, unde practic se asteapta o prima atingere cu penul a ecranului pentru a debuta jocul. Pentru folosirea penului am avut nevoie de biblioteca Touchscreen pe care am folosit-o in felul urmator:

```
void gamePaused() {
  TSPoint p = ts.getPoint();
  pinMode(XM, OUTPUT);
  pinMode(YP, OUTPUT);
  if ((p.z > MINPRESSURE) && (p.z < MAXPRESSURE)) {
    p.x = map(p.x, TS_MINX, TS_MAXX, 0, tft.width());
    p.y = 240 - map(p.y, TS_MINY, TS_MAXY, 0, tft.height());
    state = inGame;
  }
  draw();
  delay(300);
}
```

Odata ce jocul a pornit, trec in starea inGame in care se petrec urmatoarele 3 lucruri: verificarea coliziunilor, preluarea inputului si updatarea valorilor si a ecranului. Mai intai, pentru verificarea coliziunilor am verificat daca jucatorul atinge 'hrana', caz in care se genereaza o noua 'hrana', se incrementeaza dimensiunea jucatorului si i se actualizeaza scorul. Daca 'hrana' ii este desenata cu verde, in cazul in care o prinde, punctajul sau se va dubla. Apoi, verific coliziunea cu peretii si cu sine insusi. Codul pentru tratarea coliziunilor este urmatorul:

```
void handleColisions() {

  //check if snake eats food
  if (snake[0].X == snakeFood.X && snake[0].Y == snakeFood.Y) {
    //increase snakeSize
    snakeSize++;
    if(score == 50) {
      score = score * 2;
    } else {
      score += 10;
    }
    //regenerate food
    spawnSnakeFood();
  }

  //check if snake collides with itself
  else {
    for (int i = 1; i < snakeSize; i++) {
      if (snake[0].X == snake[i].X && snake[0].Y == snake[i].Y) {
        state = gameOver;
      }
    }
  }
  //check for wall collisions
  if ((snake[0].X < 20) || (snake[0].Y < 20) || (snake[0].X > 320) || (snake
[0].Y > 240)) {
    delay(1000);
    state = gameOver;
  }
}
```

O problema cu adevarat importanta este tratarea inputului de la touchscreen. Algoritmul folosit pentru a detecta schimbarea miscarii jucatorului este urmatorul: am impartit ecranul in “stanga” si “dreapta” jucatorului (in care se va muta in functie de unde a fost atins ecranul), acestea fiind date de directia si sensul sau de deplasare, presupunerea fiind ca daca doreste sa isi pastreze actuala deplasare, jucatorul nu va atinge ecranul. Aceasta abordare rezolva inclusiv problema intoarcerii in sensul invers, lucru interzis in Snake dar care se poate realiza printr-o atingere dubla a ecranului. Codul este urmatorul:

```
void handleInput() {
  TSPoint p = ts.getPoint();
  pinMode(XM, OUTPUT);
  pinMode(YP, OUTPUT);
  if ((p.z > 0) && (p.z < MAXPRESSURE)) {
    p.x = map(p.x, TS_MINX, TS_MAXX, 0, tft.width());
    p.y = 240 - map(p.y, TS_MINY, TS_MAXY, 0, tft.height());
    if (snakeDir == UP) {
      if (p.x > snake[0].X) {
        snakeDir = RIGHT;
      } else if (p.x < snake[0].X) {
```

```
        snakeDir = LEFT;
      } else if (p.y < snake[0].Y) {
        snakeDir = UP;
      }
    } else if (snakeDir == DOWN) {
      if (p.x > snake[0].X) {
        snakeDir = RIGHT;
      } else if (p.x < snake[0].X) {
        snakeDir = LEFT;
      } else if (p.y > snake[0].X) {
        snakeDir = DOWN;
      }
    } else if (snakeDir == RIGHT) {
      if (p.y > snake[0].Y) {
        snakeDir = DOWN;
      } else if (p.y < snake[0].Y) {
        snakeDir = UP;
      } else if (p.x > snake[0].X) {
        snakeDir = RIGHT;
      }
    } else {
      if (p.y > snake[0].Y) {
        snakeDir = DOWN;
      } else if (p.y < snake[0].Y) {
        snakeDir = UP;
      }
      if (p.x < snake[0].X) {
        snakeDir = LEFT;
      }
    }
  }
}
```

Partea de updatare a valorilor se face practic prin mutarea pozitiei corpului i al snake-ului catre corpul i + 1, dupa care in functie de tipul de miscare efectuata, se modifica si pozitia primului element din array care este practic cea mai importanta, celelalte fiind doar pozitia din momente de timp trecuta. Tot aici golesc spatiul unde trebuie rescris scorul. Codul este urmatorul:

```
void updateValues() {
  //update all body parts of the snake excpet the head
  for (int i = snakeSize - 1; i >= 0; i--) {
    tft.fillRect(snake[i].X, snake[i].Y, gameItemSize, gameItemSize, color);
  }
  tft.fillRect(0, 0, 150, 20, color);

  for (int i = snakeSize - 1; i > 0; i--) {
    snake[i] = snake[i - 1];
  }
  //Now update the head
  //move left
  if (snakeDir == 0) {
```

```
    snake[0].X -= gameItemSize;
  }
  //move right
  else if (snakeDir == 1) {

    snake[0].X += gameItemSize;
  }

  //move down
  else if (snakeDir == 2) {

    snake[0].Y += gameItemSize;
  }

  //move up
  else if (snakeDir == 3) {

    snake[0].Y -= gameItemSize;
  }
}
```

In starea de gameLost, se actualizeaza scorul maxim, se reseteaza principalele variabile globale si se trece din noua in pausedGame dupa o pauza de cateva secunde. Functia **loop** este doar o apelare a uneia dintre celelalte 4 functii principale in functie de valoare state-ului, la care se adauga si un delay care scade in functie de scorul atins, fapt ce genereaza o crestere constanta a dificultatii jocului.

Dificultati intampinate

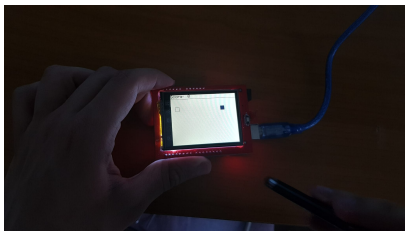
Majoritatea problemelor au fost generate de faptul ca nu am mai folosit pana acum acest tip de LCD. Exista cateva biblioteci speciale dedicate acestui LCD (precum **SPFD5408** si **TouchScreen**) care au insa cateva probleme pe care a trebuit sa le rezolv inainte sa incep implementarea, cum ar fi faptul ca afisarea textului se facea in oglinda. A fost nevoie sa intru si sa modific fisierul SPFD5408_Adafruit_TFT_LCD.cpp care a fost o provocare destul de mare, dat fiind ca implementarea este una destul de greoaie. De asemenea, biblioteca TouchScreen si ecranul foloseau sisteme diferite de coordonate pentru puncte (ecranul foloseste cadranul 4, iar TouchScreen foloseste cadranul 1). Aceasta problema mi-a dat multe batai de cap tocmai in etapa de preluarea a informatiei de la input, cand jucatorul meu decidea sa o ia in sensul opus celui indicat de mine. Problema s-a rezolvat prin maparea coordonatelor X si Y si schimbarea coordonatei Y in 240 - Y. De asemenea, faptul ca este nevoie sa curat eu de fiecare data ecranul pentru a putea sa afisez versiunea curenta a jocului este un amanunt obositor deoarece duce la creare de cod repetitiv care putea fi evitat, insa nu am gasit alta solutie automata.

Mediu de dezvoltare

Am folosit Arduino IDE pentru scrierea si incarcarea codului pe Arduino. Problema principala a fost lipsa functionalitatilor specifice unui IDE, precum autocomplete.

Rezultate Obţinute

- Jocul este perfect functional, desi mai exista momente cand touchscreenul functioneaza mai greu (probabil ar fi nevoie de o mica investitie in infrastructura)
- Consider ca o implementare cu butoane ar fi fost mult mai usor de realizat decat una cu touchscreen-ul, insa in cele din urma am reusit sa gestionez toate problemele aparute, desi am fost nevoit sa improvizez un pen deoarece pen-ul initial era foarte greu de receptat de catre LCD. Aceste probleme au facut ca jucatorul sa isi schimbe miscarea intr-o directie incorecta sau sa nu raspunda cerintelor utilizatorului, fapt ce duce la pierderea jocului. Dupa ce am reusit sa rezolv aceste probleme (de exemplu, cea cu maparea si cu modificarea sistemului de coordonate pentru punctul obtinut prin biblioteca TouchScreen a fost destul de greu de identificat), lucrurile au mers mult mai cursiv.
- Sunt multumit de rezultatele pe care le-am avut la coliziuni, acestea par sa functioneze in totalitate. Lipirea celor 2 componente principale a fost o provocare, insa sunt bucuros ca totul este acum functional.
- Linkul catre un mic demo este urmatorul: https://drive.google.com/file/d/1pseBfO7j4eTW-BCas4DWn_sqxfInbeqs/view?usp=sharing



Concluzii

A fost o experienta interesanta sa combin diverse tipuri de cunostinte din timpul semestrului, precum si cateva dintre informatiile de la EGC din semestrul anterior. Ma bucur ca am reusit sa folosesc si cateva componente hardware si sa interactionez cu ele cu succes. De asemenea, am inteles cat de

important este sa cunosti bine resursele hardware disponibile pentru a putea sa scrii un cod care sa se adapteze la cerintele pe care componentele fizice le cer si sa fie si compatibil.

Download

snake.pdf

snake.zip

Jurnal

- 10.05.2022 - Alegerea temei pentru proiect
- 15.05.2022 - Milestone 1
- 29.05.2022 - Milestone 2

Bibliografie/Resurse

- Biblioteca SPFD5408 - https://github.com/JoaoLopesF/SPFD5408
- Tutorial introductiv TFT LCD 2.4 inch - https://youtu.be/D3lv0eySz8A
- Documentatie ecran TFT LCD 2.4 inch - https://electropeak.com/2-4-tft-lcd-display-shield
- Introducere utilizare ecran TFT LCD 2.4 inch - https://create.arduino.cc/projecthub/electropeak/arduino-2-4-touch-screen-lcd-shield-tutorial-fe6f05
- Reparare oglindire text pe LCD - https://youtu.be/_mJIf1BnlU8

Export to PDF