Student: Andrei Petrea
Grupa: 331CC

PacMan

Introducere

PacMan este un joc arcade dezvoltat de compania Namco in anul 1980, fiind considerat unul dintre cele mai influente jocuri din toate timpurile, fiind responsabil pentru popularizarea jocurilor arcade si a jocurilor video in constientul maselor, generand venituri de 14 miliarde de dolari si vanzand peste 43 de milioane de unitati.

Scopul proiectului meu este sa portez acest joc, de pe arcade pe un chip Arduino, pentru a ma putea juca acest joc intr-un mod cat mai autentic cu masina originala cu care venea odinioara. Ma voi folosi de butoane, ecran lcd, difuzor si led-uri pentru a realiza acest lucru.

Descriere generală

Voi folosi un Arduino Uno pe post de unitate de procesare, la care voi atasa un ecran de 1.8” LCD pentru a afisa jocul, un speaker pentru a canta melodii si 5 butoane pentru a controla cele 4 directii (sus, jos, stanga, dreapta) + buton de start. Ecranul vine si cu un adaptor SD prin intermediul caruia care voi afisa imagine de start. La final ma voi folosi de un LED RGB pentru a afisa finalul (RED - pierdere, BLUE - castig).

Hardware Design

Lista de piese

- Arduino Uno R3 ATMega328P
- Ecran LCD 1.8” SPI
- DFROBOT - FIT0449 - Add-On Board, Speaker Module, Gravity Series, Arduino, Digital Interface
- Modul adaptor SD
- 5x Butoane

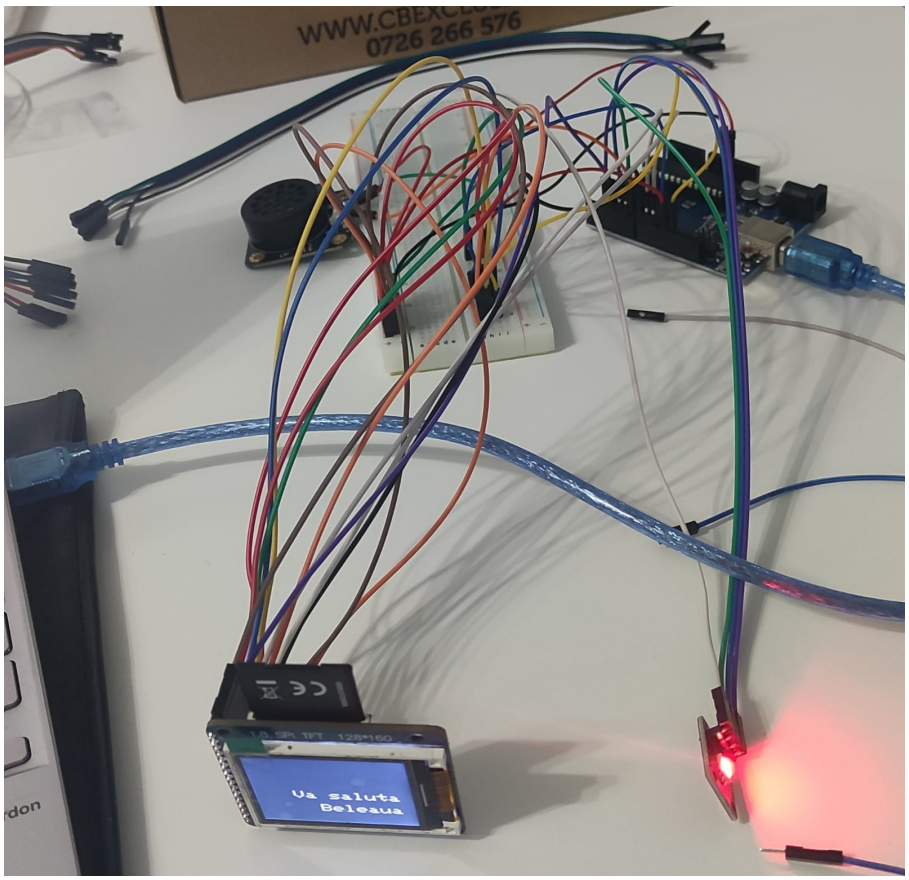
- Modul LED RGB
- Breadboard
- Fire Jumper
- Rezistente

Schema electrica

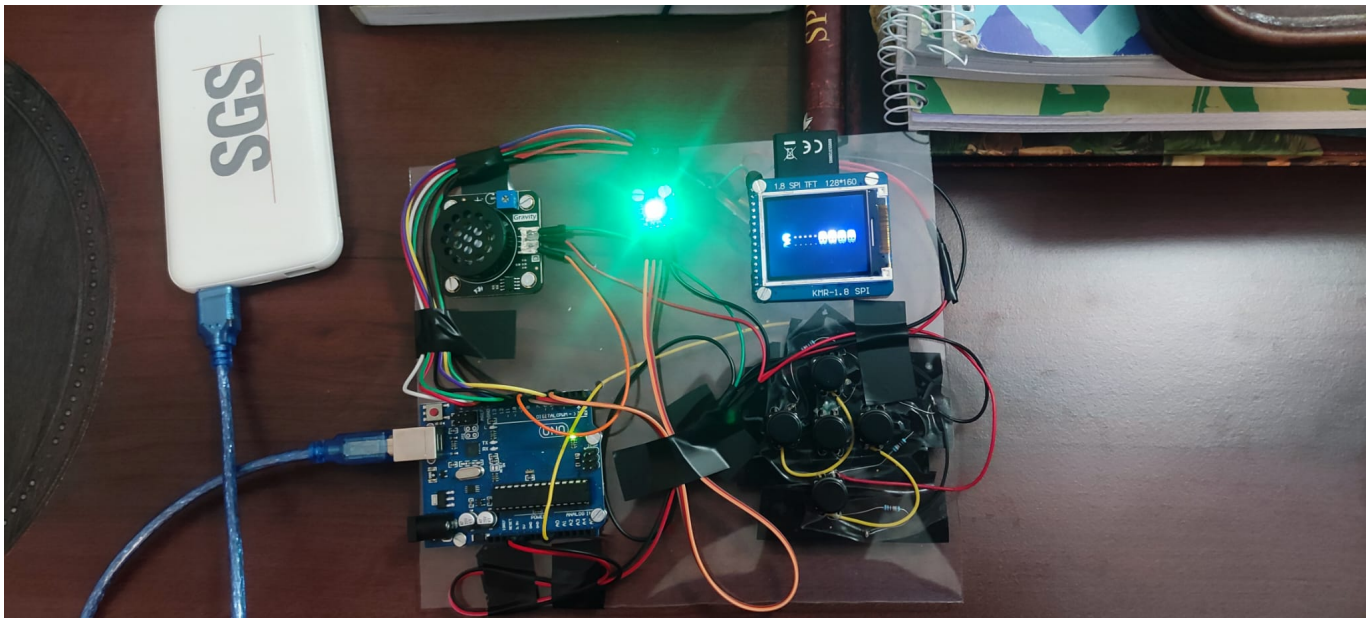
Pinii folositi

- **Butoanele** - cele 5 butoane vor fi conectate la accelasi pin analogic *A0*, folosindu-ma de ADC pentru am-mi da seama ce buton este apasat.
- **Ecran LCD 1.8” SPI** - ecranul foloseste SPI, asa ca voi trebuie sa conectez pinii *MOSI*, *MISO* si *CS*, puse pe pinii *11*, *12*, si *10* respectiv. Pe langa acestia, voi connect pinul de clock *SCK* la *13*, la fel si pinul *SCL*, iar pinul *SDA* la *11*, la fel ca cel *MOSI*. Pinul de reset va fi la pinul *8* iar *A0* la pinul *9*.
- **Modul SD** - acesta vine integrat cu ecranul, asa ca toti pinii lui SPI vor fi cei ai ecranului, cu exceptia *CS*-ului pe care l-am setat pe pinul *4*.
- **Speaker** - Acesta vine cu o interfata Gravity, care are 3 pinii: *GND*, *VCC* si *Signal*, acest din urma mergand conectat la orice pin digital, asa ca am ales pinul *7*.
- **Modulul LED RGB** - il voi connecta la 3 pini digital cu *PWM*, acestia fiind *5*, *3* si *6*.

Forma Initiala



Forma Finala



Software Design

Mediu de dezvoltare

Arduino IDE, limbaj C/C++

Biblioteci folosite

- **SD.h** → folosita pentru interfatarea cu cardul SD
- **TFT.h** → folosita pentru afisarea pe display
- **SPI.h** → folosita pentru SD
- **tone()**, **noTone()** → pentru speaker
- **Arduino Songs** → pentru melodia PacMan clasica

Link GitHub

PacMan

Descrierea implementarii

Structura unui obiect

Pentru a opera mai usor cu logica jocului, mi-am creat structura *PacmanObject*, care contine 4 campuri

- x → abscisa obiectului
- y → ordonata obiectului
- draw → flag pentru afisarea la ecran
- symbol → ce afisez la ecran

```
// Symbols depending on object type
#define PACMAN "C"
#define FOOD "*"
#define ENEMY "#"

struct PacmanObject {
  int16_t x;
  int16_t y;
  bool draw;
  String symbol;
};
```

Start-up screen

La inceput, afisez o imagine sugestiva cu PacMan, acompaniata de muzica clasica si aprinderea led-ului in culoarea Verde. In cazul in care apar erori la incarcarea imaginii, voi afisa un mesaj pe ecran.

```
void intro() {
  if (!imageError) {
    TFTscreen.image(logo, 20, 25);
  } else {
    TFTscreen.text("Va saluta", 40, 64);
    TFTscreen.text("Andrei", 64, 90);
  }
  analogWrite(PIN_GREEN, 64);
  playPacmanIntro();
  TFTscreen.text("Press START", 18, 85);
}
```

Game Loop

Ca sa inceapa jocul, jucatorul trebuie sa apese butonul de START (butonul din centru), care va crea o noua arena, plasand PacMan-ul in centru ei, si plasand 4 inamici si 7 puncte aleatoriu pe harta.

```
void loop()
{
  int value = analogRead(A0);
  if (!gameStarted) {
    if (checkRange(value, 260, 330)) {
      gameStarted = true;
      gameOver = false;
      ateAll = false;
      score = 0;
      initBoard();
      analogWrite(PIN_RED, 0);
      analogWrite(PIN_BLUE, 0);
      analogWrite(PIN_GREEN, 64);
    }
  }
  // Rest of code...
}
```

Pentru realizarea randomizarii pozitiilor, am folosit functiile *rand* si *srand* din biblioteca standard C, seed-ul pe care l-am ales fiind obtinut prin citirea unui pin analogic neconectat (in cazul meu, A1).

```
void setup()
{
  // Rest of code...
  srand(analogRead(A1));
}
```

Odata intrat in joc, ne putem deplasa *UP*, *DOWN*, *LEFT* si *RIGHT* prin apasarea celor 4 butoane, dispuse conform celor 4 puncte cardinale. In functie de care buton a fost apasat, voi apela functia *move*, care are ca argument, deplasarea pe care i-o voi da PacMan-ului.

```
void loop()
{
   // Rest of code...
   if (!gameOver) {
      draw();
      if (checkRange(value, 120, 170)) {
        move(0, -1);
      } else if (checkRange(value, 190, 230)) {
        move(-1, 0);
      } else if (checkRange(value, 420, 500)) {
        move(1, 0);
      } else if (checkRange(value, 900, 1024)) {
        move(0, 1);
```

```
      }
    }
      // Rest of code...
}
```

In functia *move*, voi muta PacMan-ul la pozitia noua, si voi calcula folosind functia *checkCollision*, daca ma intersectez cu obiectul de tip *FOOD* sau cu *ENEMY*. Ma voi folosi si de functia *checkRange*, pentru a oferi toleranta la detectia coliziunii.

```
bool checkCollision(PacmanObject obj1, PacmanObject obj2) {
  return (checkRange(obj1.x, obj2.x - 2, obj2.x + 2) && checkRange(obj1.y,
obj2.y - 2, obj2.y + 2) && checkRange(obj2.x, obj1.x - 2, obj1.x + 2) &&
checkRange(obj2.y, obj1.y - 2, obj1.y + 2));
}

bool checkRange(int val, int low, int high) {
  return (val >= low && val <= high);
}
```

Inedit, pentru inamici, am folosit un algoritm de tip **Hill-Climbing**, care este apelat atunci cand jucatorul face o mutare, facand ca acestia sa se deplaseze catre PacMan. Fiecare inamic se va uita la starile sale vecine (*UP*, *DOWN*, *LEFT*, *RIGHT* in ordinea aceasta) si va alege prima stare mai apropiata decat pozitia sa curenta, folosind distanta Manhattan ca euristica.

```
void move(int16_t dx, int16_t dy)
{
   // Rest of code...
   for (i = 0; i < NR_ENEMIES; i++) {
    int8_t j, k;
    int16_t currDistance = manhattanDistance(pacMan, enemies[i]);
    PacmanObject aux;
    for (j = -1; j <= 1; j++) {
      bool ok = true;
      for (k = -1; k <= 1; k++) {
        if (abs(j) != abs(k)) {
          aux.x = enemies[i].x + j;
          aux.y = enemies[i].y + k;
          int16_t distance = manhattanDistance(pacMan, aux);
          if (distance < currDistance) {
            enemies[i].x = aux.x;
            enemies[i].y = aux.y;
            ok = false;
            break;
          }
        }
      }
      if (!ok) {
        break;
      }
    }
  }
```

```
}
}

int16_t manhattanDistance(PacmanObject obj1, PacmanObject obj2) {
  return (abs(obj2.x - obj1.x) + abs(obj2.y - obj1.y));
}
```

Final screen

La final, jucatorul ori a colectat toate punctele ori a fost ucis de catre inamic. In functie de cele 2 urmari, se va afisa la ecran un mesaj corespunzator, se va schimba culoarea led-ului intr-una care reflecta rezultatul final si se va pune la speaker o melodie aferenta.

```
void loop()
{
  // Rest of code...
      TFTscreen.fillScreen(TFT_BLACK);
      char scoreSir[11];
      sprintf(scoreSir, "Scor: %d\n", score);
      if (ateAll) {
        TFTscreen.text("Ati castigat! :)", 15, 50);
        TFTscreen.text(scoreSir, 40, 70);
        analogWrite(PIN_GREEN, 0);
        analogWrite(PIN_BLUE, 64);
        playPacmanIntro();
      } else {
        TFTscreen.text("Ati pierdut! :(", 15, 50);
        TFTscreen.text(scoreSir, 40, 70);
        analogWrite(PIN_GREEN, 0);
        analogWrite(PIN_RED, 64);
        playFailedSong();
      }
      gameStarted = false;
   // Rest of code...
}
```

Concluzii

- Proiectul a fost amuzant de facut (exceptand partea hardware :( ), si acum simt ca inteleg mai bine arhitectura calculatoarelor
- Nu pot sa folosesc memoria la fel ca si pe calculator, deoarece am foarte putina memorie si se umple foarte repede
- Sa va uitati de 10 ori in datasheet sa verificati ca ati legat bine firele :)
- Sa va aveti planuri cum o sa arate proiectul, ca sa nu va cumparati piese degeaba si sa va cumparati piese de calitate

Rezultate Obţinute

Jurnal

- **27.04** - Initializat pagina de proiect
- **03.05** - Adaugat documentatia initiala
- **05.05** - Adaugat imaginea de coperta
- **16.05** - Adaugat schema electrica + detalii pini + jurnal
- **19.05** - Adaugat legarea initiala
- **21.05** - Adaugat design software initial + bibliografie + legarea finala
- **25.05** - Finalizat design software + pagina OCW

Bibliografie/Resurse

Resurse Hardware

- circuit.io
- Arduino UNO R3 Datasheet
- KMR 1.8" TFT LCD Datasheet
- TUTORIAL: How to work with a 1.8" SPI TFT with strange incorrect labelling!
- Accessing 5 Buttons Through 1 Arduino Pin - Revisited

Resurse Software

- Documentatia Arduino
- Arduino IDE
- Arduino Songs
- Hill climbing

Download

Download archive

Export to PDF