UNIVERSITATEA DIN BUCUREȘTI

FACULTATEA

DE

MATEMATICĂ ȘI INFORMATICĂ

SPECIALIZAREA INFORMATICĂ

Lucrare de licență

CRIME IN THE BLOCK

Absolvent

Zaharia Diana Cristiana

Coordonator științific

Conf. Dr. Mihai Prunescu

București, septembrie 2023

Rezumat

Jocul “Crime in the block” este un prototip cu tematică de mister conceput să testeze atenția utilizatorului la detalii si a capacității acestuia de a rezolva puzzle-urile de tip poveste, fiind rezultatul a ceea ce se poate realiza când arta și tehnologia cooperează.

Ceea ce pare inițial a fi doar un tablou pictat, prin simpla conectare la o sursa de curent, devine un joc interactiv care te pune în pielea unui detectiv și te invită să intervii în rezolvarea cazului, prin găsirea criminalului din bloc. Jocul este alcătuit din patru componente esențiale: hardware, software, fișiere audio și pictură. Hardware-ul, pictura și fișierele audio sunt interfața jocului. Componentele luminoase se integrează în tablou și alături de boxa ce transmite fișierele audio, aduc povestea la viață. Software-ul este integral realizat în Arduino IDE, folosind librăriile care permit controlul cel mai eficient al hardware-ului și un algoritm de mașină de stări .

Prin acest joc am urmărit să îmbin tehnicul cu creativul și să demonstrez că tehnologia nu trebuie să îngrădească, să ducă la pierderea abilităților tradiționale, să includă ecrane dăunătoare ochilor, să fie o distragere sau să fie strict pentru comoditate. Tehnologia poate să facă parte din frumusețe și să ofere noi modalități de a crea, de a interacționa cu arta și de a da viață imaginației noastre, fără de care nimic nu ar fi fost inventat sau creat. La fel ca și în cazul proiectului meu, totul a pornit în cele din urmă de la o simplă idee.

Abstract

The game "Crime in the Block" is a mystery-themed prototype meticulously designed to evaluate the user's attention to detail and problem-solving prowess within the context of story-driven puzzles. It stands as a testament to the harmonious collaboration between artistry and technology.

What initially appears as nothing more than a painted canvas is transformed into an interactive game by the mere connection to a power source, immersing the participant in the role of a detective and beckoning them to engage in resolving the case by finding the perpetrator within the block. The game comprises four essential components: hardware, software, audio files, and painting. The hardware, tableau, and audio files collectively constitute the game's interface. The luminous elements seamlessly integrate into the canvas and, coupled with the speaker, bring the story to life. The software is

entirely constructed within the Arduino IDE, employing libraries that enable the most efficient control and incorporating a finite state machine algorithm.

In conceiving this game, the goal was to seamlessly merge the technical with the creative, dispelling the notion that technology ought to constrain, lead to the erosion of traditional skills, introduce harmful screen-centric habits, serve as a distraction, or exist solely for convenience. Technology can be an integral facet of beauty, offering novel means of creation, interaction with art, and giving life to our imagination, without which nothing would have been invented or created. Much like my project, it all began with a simple idea.

Cuprins

Capitolul 1

Introducere

1.1 Motivație

În anul trei de facultate m-am înscris la opționalul de robotică și am văzut o nouă față a informaticii. Am reușit să înțeleg mai bine sinergia dintre hardware și software și am avut oportunitatea să realizez proiecte, la scară mică, unde codul părăsea ecranul laptopului și se materializa într-un obiect.

Unul din proiectele mai ample din cadrul cursului mi-a oferit șansa de a îmi folosi partea creativă, care de ceva timp se simțea neglijată și mi-a întărit cunoștințele în limbaje formale automate și rețele de calculatoare. Atunci am utilizat pentru prima oară o componentă ce folosea comunicarea SPI și un algoritm de mașină de stări pentru a crea un joc simplu pe o singură matrice de leduri 8x8. Realizarea unui produs finit, proiectat, construit și programat complet de mine mi-a adus o satisfacție sporită și am știut în acel moment că proiectul meu de licența va fi tot un joc, tot în robotică, dar la o scară mult mai mare, cu componente noi, un alt concept și un algoritm complex. Eram pregătită pentru o nouă provocare atât creativă cât și tehnică.

Până la găsirea tematicii și funcționalității finale, proiectul a avut multe forme. Nu este o surpriză că realizarea integrală a unui proiect de dimensiunea aleasă(80x80 cm), care implică și părți non-tehnice, într-un timp limitat și cu resurse limitate, s-a dovedit a necesita o adaptare constantă, atât în logica jocului, cât și în poziționarea, conectarea și alegerea componentelor utilizate.

Am ales într-un final să fie un joc de tip puzzle cu tematică de mister, deoarece citesc cu drag de mică romane polițiste, urmăresc seriale polițiste și îmi place să merg cu prietenii la localurile de tip Escape Room și știu că sunt mulți alții ce îmi împărtășesc aceste pasiuni.

Am vrut să am un tablou pictat, deoarece simțeam nevoia de a se vedea personajele din scenariul scris de mine așa cum mi le-am imaginat eu atunci când cineva se joacă, pentru că eu consider că și vizualul face parte din analizarea unui suspect. De asemenea, mi-a plăcut faptul că rupeam bariera între oameni și tablouri, pe care de cele mai multe

ori trebuie să le admiri de la distanță, jocul prezentat în schimb necesită ca utilizatorii să atingă pictura.

1.2 Scopul

Acest joc a fost creat atât pentru a fi expus in muzee care încurajează arta interactivă sau ieșită din comun, care sunt cele mai frecventate de tineri precum: Muzeul Simțurilor și Muzeul Național de Artă Contemporană, dar și ca un posibil mini-game în cadrul unui local de tip Escape Room. Poate fi pus fie ca parte dintr-o cameră sau în exterior ca un bonus pentru clienți, care daca reușesc să depășească recordul, pot să primească diferite oferte promoționale. În plus poate chiar sa fie un produs de promovare a localurilor de acest tip.

Interfața este bazată predominant pe audio redat de o boxa cu ajutorul unui cititor de carduri SD, componente luminoase și în rest de pictură și indicii desenate. Astfel îi oferă proiectului puterea unui cameleon. Se poate introduce scenariul pe un nou card SD în orice limbă, cu o nouă poveste, alături chiar de altă pictură și indicii desenate, cu orice tematică, pentru orice vârstă. Atâta timp cât se păstrează structura de opt mini-games și o întrebare finală, hardware-ul poate fi reutilizat integral, iar codul necesită modificarea unei singure variabile pentru a schimba care este răspunsul corect din variantele multiple și dacă se dorește restructurarea celor opt jocuri scurte intermediare, câteva schimbări de complexitate minimală, dat modul eficient și ușor de citit al realizării algoritmului.

1.3 Jocuri și proiecte asemănătoare

- Cluedo - variantă pe calculator sau joc de societate[i]

Cluedo este un joc de mister în care jucătorii trebuie să ghicească detaliile corecte ale unei crime: criminalul, arma și locația. Jocul începe cu trei cărți care sunt puse într-un plic. De acolo, fiecare jucător primește și el trei cărți, fiecare conținând o persoană, o locație și o armă.

Se aruncă pe rând zarul și se folosește procesul de eliminare pentru a ghici care sunt cele trei cărți din plicul inițial. Se ține evidența indiciilor pe o listă.

Jocul Cluedo – Fig.1

- Unheard – Voices of Crime – joc pe calculator[ii]

Unheard pune abilitățile auditive la încercare. Acest joc este bazat pe narativă și permite explorarea din cameră în cameră. Jucătorul nu este în rolul unui personaj obișnuit, ci ascultă în secret într-o stație de poliție.

Există o listă de întrebări și trebuie indetificată persoana corectă pentru a rezolva cazul.

Jocul Unheard – Voices of Crime – Fig.2

- Arduino/Art: voice your opinion[iii]

Acesta primește intrări vocale și le redă sub forma aprinderii unor LED-uri, unde numărul de LED-uri aprinse reflectă nivelul de intensitate vocală înregistrat în timpul conversației. Pentru a conecta LED-urile, au fost utilizate fire pentru conexiune arduino pentru programarea lor și a microfonului Electret, folosit pentru a măsura variațiile de presiune ale sunetului și pentru a le converti în semnale electrice. Proiectul funcționează în intervale de 100 de milisecunde, selectând în ordine aleatoare LED-urile aprinse în funcție de intensitatea vocii respective.

Arduino/Art: voice your opinion -Fig.3

1.4 Ce aduce nou?

Interfața formată din componente luminoase cu jocuri de lumini în combinație cu narativa audio și tabloul pictat sunt un format nemaiîntâlnit în cadrul unui joc de mister sau prototip în robotică.

În plus, algoritmul complex de mașină de stări permite jucătorului accesarea nivelelor intermediare în orice ordine deduce acesta a fi mai eficient, făcând parte din strategia jocului. Astfel experiența este mai personalizată decât în cazul unui joc liniar sau care funcționează prin aruncarea unui zar.

O altă caracteristică menționată și mai sus este faptul ca jocul poate fi adaptat unei noi tematici și limbi utilizând resurse financiare și umane minime.

1.5 Structura lucrării

În **Capitolul** 1 am descris detalii generale despre proiect.

În **Capitolul 2** este descrisă partea fizică a sistemului informatic accentul fiind pus pe conexiuni, procesul de testare al componentelor și tehnologiile utilizate pentru controlul acestora

În **Capitolul 3** sunt prezentate tehnologiile utilizate în cadrul realizării jocului.

În **Capitolul 4** este prezentat algoritmul de mașini de stări alături de o anexă a codului și imagini a cum arată jocul și funcționalitatea fiecărei stări din perspectiva unui utilizator precum și o descriere mai amplă a jocului alături de anexele cu instrucțiunile și scenariul.

Capitolul 5 reprezintă concluzia lucrării și direcții de cercetare și dezvoltare ce vor urma, pornind de la acest prototip realizat cu resurse financiare, umane și de timp restrânse.

Capitolul 2

Componente Hardware, Conexiuni, Testări

2.1. Arduino Mega 2560[iv]

Arduino Mega 2560 este folosit des în domeniul informaticii și al electronicelor pentru prototipare rapidă și dezvoltarea proiectelor bazate pe microcontroler. Are arhitectura bazată pe un microcontroler AVR ATmega2560. Acest dispozitiv furnizează un mediu eficient și versatil pentru dezvoltarea software-ului și controlul dispozitivelor hardware.

Arduino Mega 2560 – Fig.4

De asemenea, Arduino MEGA 2560 a fost conceput pentru proiectele care necesită o cantitate mai mare de linii de intrare/ieșire (I/O), mai multă memorie pentru cod și o capacitate mai mare de memorie RAM. Are mai mulți pini digitali (54) pentru In/Out, multiple intrări analogice (16) și o arie de stocare mai generoasă pentru codul proiectului, această placă este recomandată în special pentru utilizarea în imprimantele 3D și proiectele de robotică. Aceasta asigură proiectelor ample resurse și oportunități, păstrând totodată caracteristicile de simplitate și eficacitate ale platformei Arduino.

Conexiunea USB cu PC-ul este imperativă pentru programarea plăcii și nu doar pentru a furniza energie acesteia.

Principalele caracteristici software ale Arduino Mega 2560 includ:

- IDE include: limbaj similar C/C++, compilator, editor pentru cod, instrumentele necesare pentru gestionarea și încărcarea programelor pe placa Arduino.

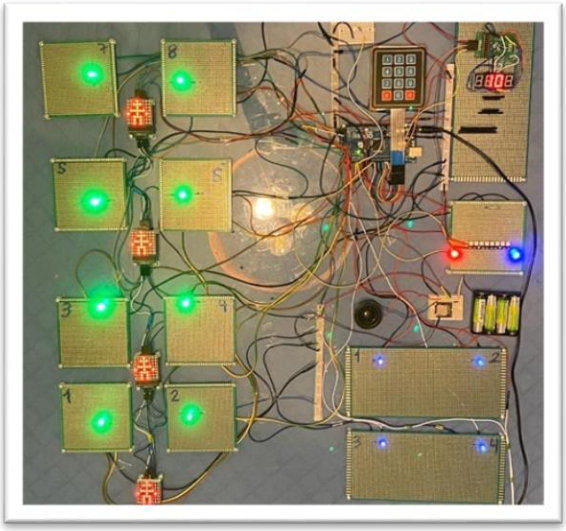
- Bibliotecă standard Arduino: Placa Arduino Mega 2560 este însoțită de o bogată bibliotecă standard care facilitează accesul la funcționalități hardware și simplifică dezvoltarea aplicațiilor. Această bibliotecă abstractizează multe aspecte ale hardware-ului. Astfel, dezvoltatorii se pot concentra mai mult pe logică și interacțiunea cu componentele hardware.

Comunitatea Arduino oferă o gamă diversă de librării și extensii, care pot fi folosite pentru a extinde funcționalitatea plăcii Arduino Mega 2560.

2.2. LED-uri[v]

În proiect am utilizat 8 LED-uri RGB de 5V de la care am conectat doar verdele și roșul, 2 LED-uri RGB de la care am conectat unul din ele doar roșu și celălalt doar albastru pentru mașina de poliție și 5 LED-uri comune de 5V(unul din ele se aprinde la apăsarea de taste, nu este aprins în poza de mai jos).

Componentele luminoase aprinse – Fig.5



LED-urile RGB și cele monocromatice sunt două tipuri de diode emițătoare de lumină cu caracteristici diferite.

LED-uri RGB (Red, Green, Blue):

Culori multiple: LED-urile RGB sunt capabile să producă o gamă largă de culori, deoarece sunt compuse din trei tipuri de LED-uri individuale: roșu, verde și albastru. Prin amestecul acestor culori primare, puteți obține o paletă variată de culori, inclusiv culori intermediare și culori personalizate.

Control individual: Fiecare dintre cele trei LED-uri (roșu, verde și albastru) poate fi controlat individual, permițând ajustarea intensității fiecărei culori pentru a obține culoarea dorită.

Aplicații colorate: LED-urile RGB sunt folosite în mod obișnuit în aplicații care necesită iluminare colorată sau efecte de iluminare dinamice, precum iluminarea ambientală, decorul de interior, spectacole de lumini, etc.

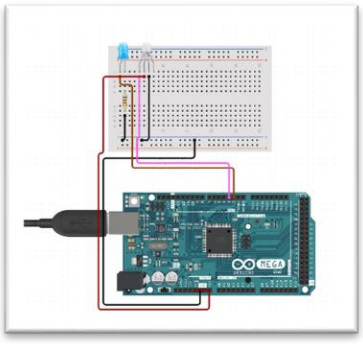
LED-uri monocromatice:

Culoare singulară: LED-urile comune sunt specializate în emisia unei singure culori (roșu, alb, verde, galben, albastru). Fiecare LED monocromatic emite doar o anumită lungime de undă, ceea ce înseamnă că culoarea lor nu poate fi modificată.

Simplu de utilizat: LED-urile monocromatice sunt simple de utilizat și de controlat, deoarece nu necesită ajustarea culorii sau a intensității culorii.

Aplicații specifice: LED-urile comune sunt adesea folosite în dispozitive care necesită un anumit tip de lumină sau semnal, precum indicatoarele, afișajele, afișajele cu 7 segmente și multe altele.

Diagrama de conectare – Fig.6



Funcție testare aprindere becuri pentru un bec RGB si un bec normal:

(ce este comentat aprinde partea roșie din becul RGB, șterg comentarea lui roșu și comentez verde pentru testare)

```
void testAllBright(){

  digitalWrite(LED_PIN1V, HIGH); //digitalWrite(LED_PIN1R, HIGH);

   digitalWrite(LED_PINI1, HIGH);}
```

2.3. Pad de tastatură pentru Arduino[vi]

Un pad de tastatură dedicat utilizării cu platforma Arduino reprezintă un element cheie în cadrul construcției interfeței de interacțiune pentru aceste dispozitive. Este o componentă care permite utilizatorilor să introducă comenzi sau să furnizeze date către un microcontroler Arduino sau un sistem similar, prin intermediul unui set de butoane sau taste.

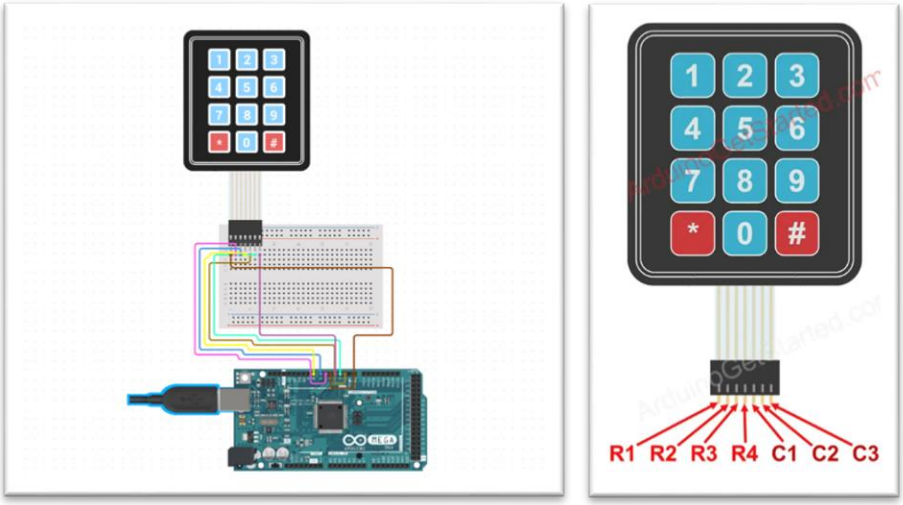
Cunoscute și sub denumirea de "keypads" în limba engleză, aceste dispozitive se prezintă într-o varietate de configurații, însă în mod obișnuit sunt constituite dintr-o matrice de butoane, fiecare dintre acestea având alocată o literă, cifră, simbol sau funcție specifică.

Un pad de tastatură destinat utilizării cu platforma Arduino se bazează pe principiul scanării matricei de butoane pentru a detecta și interpreta apăsările acestora. Acest proces funcționează în următoarele etape:

Conexiune la Arduino:

Pad-ul de tastatură este conectat la pini specifici ai microcontroler-ului Arduino. Fiecare buton este asociat unui pin, iar matricea de butoane este organizată astfel încât să permită citirea eficientă a stării acestora.

Diagrama de conexiune pad – Fig.7.1



Mod de funcționare:

Scanarea matricei: Pad-ul de tastatură utilizează o tehnică de scanare pentru a citi starea butoanelor. Într-un ciclu, un rând sau o coloană este activat, iar Arduino citește starea tuturor butoanelor din acea zonă. Acest proces se repetă pentru fiecare rând sau coloană.

Detectarea apăsărilor: Arduino compară starea curentă a butoanelor cu starea anterioară, identificând butoanele care au fost apăsate (au trecut de la starea "neapăsat" la "apăsat").

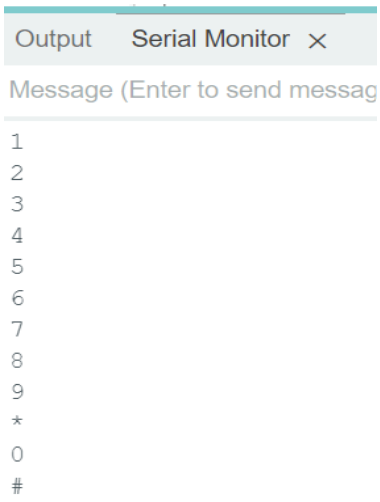
Interpretarea apăsărilor: După detectarea unei apăsări, Arduino poate interpreta această acțiune și poate executa o acțiune corespunzătoare, cum ar fi activarea unei funcționalități sau afișarea unui mesaj pe un afișaj.

Actualizare continuă: Procesul de scanare și interpretare a apăsărilor butoanelor este repetat într-un ciclu continuu pentru a permite interacțiunea continuă cu utilizatorul.

Secțiune cod testare funcționare:

```
char cheie = keypad.getKey();
if (cheie){
  Serial.println(cheie);
}
```

Output– Fig.7.2



Am utilizat librăria Keypad.h pentru o utilizare eficientă și voi prezenta mai în detaliu în capitolul 3 modul de funcționare.

Pad-urile de tastatură pot varia în funcție de numărul de butoane și dispunerea acestora. Ele pot prezenta configurații de 4x4, 3x4 sau altele similare și pot include butoane speciale, cum ar fi cele destinate direcțiilor sau funcțiilor specifice.

2.4. MAX7219 LED driver [vii]

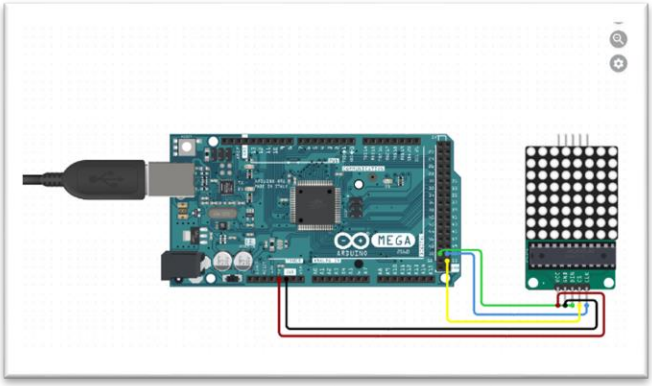
MAX7219 este un driver de afișaj compact, cu intrare/ieșire serială. Acest driver LED are capacitatea de a controla afișaje cu 7 segmente, cu până la 8 cifre, afișaje de tip bara-graf sau

64 de LED-uri individuale. Comunicarea între driverulMAX7219 și Arduino are loc prin intermediul protocolului SPI pe care îl voi detalia în capitolul 3.

DeoareceMAX7219 poate controla un număr maxim de 64 de LED-uri, dimensiunea maximă a unui afișaj cu matrice punctată pe care îl poate gestiona este de 8x8 pixeli. Cu toate acestea, este posibil să conectați în serie mai mulți driveri și matrice pentru a controla afișaje mult mai mari, cum ar fi cele de 8x32, 8x64 sau chiar mai extinse. Notabil, pentru a controla toate aceste cipuri, este necesară utilizarea doar a trei fire de conexiune, ceea ce înseamnă că veți economisi pini de intrare/ieșire ai Arduino. Procedura implică conectarea pinului DOUT al primului afișaj la pinul DIN al următorului afișaj în serie. Pinii VCC, GND, CLK și CS sunt partajați între toate display-urile pentru o gestionare eficientă a conexiunilor.

Eu l-am utilizat în cadrul unui modul de controlare a matricei cu leduri 8x8 și pentru controlarea unui afișaj cu patru cifre și șapte segmente de care voi vorbi mai jos.

Diagrama de conexiuni modul matrice – Fig.8



Am folosit librăria LedControl. Un singur driver LED MAX7219 poate controla 64 de LED-uri. Biblioteca suporta pana la 8 drivere MAX7219 conectate in serie. Controlul a 512 LED-uri este de obicei suficient pentru majoritatea scopurilor. Voi intra în detaliu în Capitolul 3.

MAX7219 este folosit în numeroase aplicații, inclusiv în dispozitive electronice de consum, panouri publicitare cu LED-uri, afișaje digitale în sisteme de control industrial, jocuri electronice și multe altele.

2.5. Afișaj cu patru cifre și șapte segmente [viii]

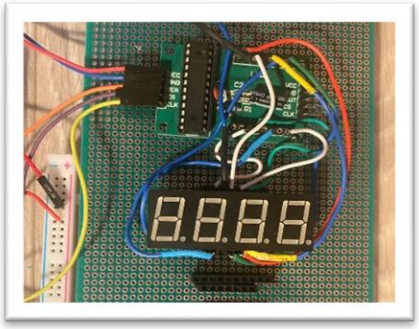
Afișajul cu patru cifre și șapte segmente reprezintă o componentă semnificativă în domeniul electronicelor. Cu un design compact, acesta dispune de patru cifre, fiecare alcătuită din șapte segmente individuale. Aceste segmente pot fi controlate independent, permițând afișarea

cifrelor și chiar a unor caractere alfanumerice. Utilizarea acestui tip de afișaj este comună în dispozitive precum ceasuri digitale, termometre sau alte aplicații. In acest proiect acesta este utilizat pentru afișarea scorului.

Conexiunile se realizează cu ajutorul unor pini, iar funcționarea sa este gestionată prin software, oferindu-se control asupra fiecărui segment pentru a afișa cifrele și caracterele dorite.

Pentru un control mai bun asupra multiplexării și utilizarea unui număr redus de pini(doar 3 din arduino) am conectat display-ul la un MAX7219.

Conexiune display- Fig.9



Librăria folosita este LedControl.

2.6. Matrice de LED-uri 8x8 [ix]

Matricea de LED-uri 8x8 reprezintă o interfață complexă pentru afișarea de imagini sau modele grafice. Software-ul controlează individual fiecare LED, permițând afișarea de imagini, animații sau alte informații vizuale. De obicei, conexiunile se realizează prin intermediul pinilor de control și comunicație digitală, cum ar fi protocolul SPI. Această matrice de LED-uri are aplicații diverse, de la afișarea mesajelor în dispozitive publicitare până la componente de design pentru proiecte electronice complexe.

Eu am folosit conexiunea în “daisy chain”. Aceasta reprezintă o tehnică de interconectare utilizată în domeniul electronicii pentru a controla mai eficient mai multe dispozitive similare. În contextul acestui studiu, patru module MAX7219 , fiecare fiind responsabil pentru controlul unei matrice de LED-uri, sunt conectate în serie într-un lanț continuu, asemenea margaretelor într-un șir.

Fiecare modul MAX7219 este dotat cu porturi de intrare și ieșire specifice care permit transmiterea semnalului de control și a datelor de la un modul la altul într-o manieră liniară. Prin conectarea acestor module într-un lanț de comunicație “daisy chain”, semnalul este transmis secvențial prin intermediul fiecărui modul, permițând unui singur dispozitiv controlor,

în cazul acestui proiect, Arduino Mega, să gestioneze și să sincronizeze operațiunile pentru toate modulele implicate.

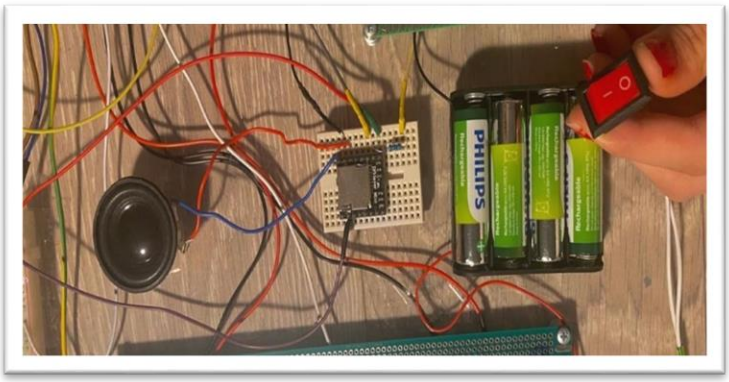
Această abordare prezintă avantajul consolidării și simplificării controlului în cazul proiectelor care implică utilizarea a mai multor afișaje cu matrice de LED-uri. Ea asigură economisirea resurselor de control și optimizează interacțiunea cu dispozitivele într-o configurație mai amplă. Mai multe despre asta și librăria utilizată, LedControl în capitolul 3.

2.7. DFPlayer mini + Boxa + Baterie [x]

DFPlayer mini, în combinație cu o boxă și o sursă de alimentare pe bază de baterie, reprezintă o soluție completă pentru redarea sunetelor sau melodiilor în proiectele electronice. DFPlayer mini este un modul specializat pentru redarea fișierelor audio, care poate fi controlat prin comenzi software. Împreună cu o boxă, acesta poate reproduce sunete sau chiar melodii. Pentru a asigura portabilitate și independență energetică, se poate adăuga o sursă de alimentare pe bază de baterie. Această configurație este folosită în proiecte precum jucării interactive, dispozitive de semnalizare sonoră sau orice aplicație care necesită redarea de sunete.

Am alimentat separat DFPlayer-ul pentru a asigura o alimentare stabilă(de 4,8V, el necesitând o valoare între 3,3V-5V), utilizând deja foarte multe componente nu merge prin alimentarea directă la Arduino Mega. Am adăugat un switch on/off pentru oprirea alimentării când jocul nu este utilizat.

DFPlayer mini + Boxa + Baterie - Fig.10.1



Este controlat prin librăria DFPlayerMini_Fast.h despre care o să vorbesc mai multe în capitolul 3.

Atașez codurile de testare a boxei și a întregului setup sonor:

Boxa:

```
for (int aceastaNota= 0; aceastaNota< 112; aceastaNota++) {
   int notaDurata = 750 / noteDurata[aceastaNota];
   tone(10, melodie[aceastaNota], notaDurata);
   int pauzaNote = notaDurata * 1.30;
   delay(pauzaNote);
   noTone(10);
 }
```

Unde melodie este un array de note declarate într-un fișier extern numit pitches.h, iar notaDurata e un array ce conține durata fiecărei note.

Tot:

```
#include <DFPlayerMini_Fast.h>
DFPlayerMini_Fast myMP3;
bool trackPlayed = false; // Indicator pentru a urmări dacă piesa a fost redată
int state = 0;
void setup()
{
  Serial.begin(9600);
  Serial1.begin(9600); // Hardware Serial 1 pe pini 18 (RX1) și 19 (TX1)
  myMP3.begin(Serial1, true);
  delay(1000); // Așteptați 1 secundă pentru a permite inițializarea DFPlayer Mini
  Serial.println("Setare volum la maxim");
  myMP3.volume(25);
}
void redaPiesaSpecificaOdata(uint16_t numarPiesa)
{
  if (!trackPlayed)
  {
    Serial.print("Redare piesa ");
    Serial.println(numarPiesa);
    myMP3.play(numarPiesa);
    trackPlayed = true; // Setează indicatorul pentru a arăta că piesa a fost redată
  }
```

```
}
void loop()
{
 // Apelează funcția pentru a reda o anumită piesă doar o dată
 // Înlocuiți 2 cu numărul piesei dorite
 if (state == 0)
 {
  redaPiesaSpecificaOdata(2);
  delay(4000);
  state = 1;
  trackPlayed = false;
 }
 if (state == 1)
 {
  redaPiesaSpecificaOdata(3);}}
```

Output cod –Fig.10.2

Parametrul "_stack" este un obiect de tip structură (struct) care conține un pachet de date de configurare sau comandă. Această structură este definită în interiorul clasei "DFPlayerMini_Fast" și este folosită pentru a organiza și stoca informațiile necesare pentru comunicarea cu playerul MP3 DFPlayer Mini. Mai multe despre librărie în capitolul 3.

Structura "_stack" are următoarele atribute publice, fiecare având un anumit rol în pachetul de date:

"start_byte": Acest atribut reprezintă un octet care indică începutul pachetului de date. Este folosit pentru a marca începutul informațiilor transmise.

"version": Acest atribut indică versiunea sau tipul de pachet de date. Poate fi folosit pentru a distinge diferite tipuri de comenzi sau configurări.

"length": Acest atribut indică lungimea totală a pachetului de date, adică câte octeți sunt incluși în pachet.

"commandValue": Acest atribut conține valoarea specifică a comenzii sau configurației. De obicei, aceasta indică acțiunea dorită de efectuat de către playerul MP3.

"feedbackValue": Acest atribut poate conține valori de feedback sau de răspuns de la player, dacă este cazul.

"paramMSB" și "paramLSB": Aceste două atribute reprezintă octeții de parametri ai comenzii sau configurației. Împreună, acești doi octeți pot forma valori mai mari sau cu precizie sporită.

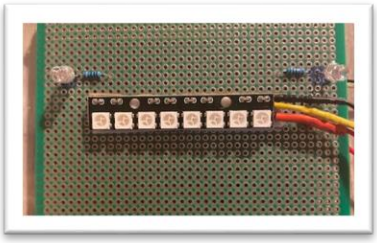
"checksumMSB" și "checksumLSB": Aceste două atribute conțin octeții de verificare a corectitudinii datelor din pachetul de comandă. Acești octeți sunt utilizați pentru a asigura integritatea informațiilor transmise.

"end_byte": Acest atribut indică sfârșitul pachetului de date și marchează încheierea informațiilor transmise.

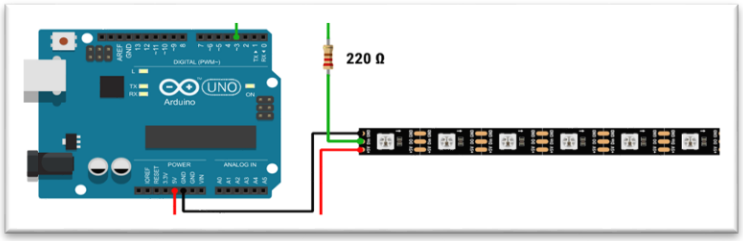
2.8. WS2812B [xi]

WS2812B este un tip de LED adresabil individual, cunoscut și sub numele de "Neopixel". Aceste diode emițătoare de lumină pot fi controlate individual prin intermediul unui protocol de comunicare(pe care îl voi detalia în capitolul 3), permițând afișarea unei game variate de culori și efecte luminoase. Conexiunile se realizează în cascadă, conectând un LED la altul. Software-ul poate controla intensitatea culorilor și modul în care acestea se schimbă, oferind posibilități creative nelimitate. Am folosit librăria FastLED.h pentru controlul lor despre care voi intra în amănunte în capitolul3.

WS2812B – Fig.11.1



Diagramă conectare WS2812B – Fig.11.2



Capitolul 3

Tehnologii utilizate

3.1. Arduino Ide

3.1.1. Caracteristici[xii]

Arduino IDE este un software open-source dezvoltat de Arduino.cc, folosit pentru a scrie, compila și încărca cod pe modulele Arduino. Acesta rulează pe diverse sisteme de operare, oferind un mediu de dezvoltare cu editor, compilator și monitor serial. Cu ajutorul IDE-ului, utilizatorul poate crea "sketch-uri" (cod), compila și încărca acest cod pe placa Arduino. IDE-ul acceptă limbajele C și C++, iar utilizatorul poate accesa biblioteci specializate pentru funcții Arduino.

Monitorul Serial inclus este o unealtă importantă pentru depanarea și comunicarea cu Arduino. Este necesar ca modulul Arduino să fie conectat la computer printr-un cablu USB pentru a-l utiliza.

Codul principal(sketch-ul) generează un fișier Hex care este apoi transferat și încărcat în controlorul de pe placa Arduino. Procesul tehnic propriu-zis de transmitere a fișierului Hex pe o placă Arduino implică transformarea codului sursă în limbaj de asamblare, generarea unui fișier binar și apoi a unui fișier Hex. Acest fișier Hex este transmis către placa Arduino printr-un protocol serial (UART),printr-un port USB, folosind un programator sau un bootloader special, astfel încât să poată fi programată cu noul cod. Atunci când se utilizează Arduino IDE pentru a încărca un program pe o placă Arduino Mega, IDE-ul transmite fișierul hexazecimal printr-o conexiune serială către bootloader.

3.1.2. Librării[xiii]

Bibliotecile reprezintă un instrument extrem de util pentru adăugarea de funcționalități suplimentare în cadrul Modulului Arduino.

Majoritatea bibliotecilor sunt preinstalate și sunt incluse în software-ul Arduino. Cu toate acestea se pot și descărca din surse externe.

3.1.2.1. "arduino.h"

"arduino.h" în limbajul de programare Arduino este o bibliotecă fundamentală care furnizează funcționalități esențiale pentru programarea plăcilor Arduino. Iată mai multe detalii tehnice despre această bibliotecă:

"arduino.h" (sau "Arduino.h" în cazul literelor mari) este inclus implicit în mediul Arduino IDE. Această bibliotecă furnizează definiții și funcții esențiale pentru manipularea hardware-ului și a porturilor de pe placa Arduino. Ea face parte din nucleul software al platformei Arduino și este inclusă automat în fiecare program Arduino creat în Arduino IDE.

"arduino.h" definește tipuri de date specifice platformei, cum ar fi "byte" (un octet), "word" (două octeți) și "boolean" (valori true/false), pentru a ajuta la gestionarea datelor în mod eficient pe placa Arduino.

Această bibliotecă oferă funcții pentru citirea și scrierea datelor pe porturile digitale ale plăcii Arduino utilizând funcții precum "digitalRead()" și "digitalWrite()". Aceste funcții sunt esențiale pentru controlul componentelor externe și a senzorilor conectați la placa Arduino. De asemenea, aceste comenzi sunt sensibile la literele utilizate, adică trebuie să fie scrise exact așa cum sunt specificate, de exemplu "digitalWrite" cu "d" mic și "W" mare. Folosirea altor variante, precum "digitalwrite" sau "Digitalwrite", nu va activa sau adresa nicio funcție specifică.

"Arduino.h" furnizează funcții pentru citirea valorilor analogice de pe porturile analogice ale plăcii Arduino prin intermediul funcției "analogRead()". Această funcționalitate este esențială pentru citirea datelor de la senzori analogici (lumină, temperatură, etc).

Biblioteca oferă funcții pentru gestionarea funcțiilor de temporizare, cum ar fi funcții pentru întârzieri specifice sau funcții pentru obținerea milisecundelor actuale de la pornirea plăcii Arduino.

Această bibliotecă include funcții pentru inițializarea plăcii Arduino, durata de viață a programului și configurarea plăcii ("setup()" apelată doar o dată la pornirea plăcii sau a rulării programului, "loop()" care rulează în buclă) .

3.1.2.2. LedControl.h[xiv]

LedControl este o bibliotecă pentru driverele de afișare Led MAX7221 și MAX7219. Pentru a controla 64 de LED-uri sau un afișaj cu 7 segmente sunt necesare doar câteva componente.

Hardware și Scheme:

Această bibliotecă suportă două drivere de afișare : MAX7219 și MAX7221. Ambele circuite pot controla până la 64 de LED-uri sau un afișaj cu 7 segmente cu 8 cifre. Driverele implementează o interfață compatibilă SPI care poate fi controlată de la Arduino folosind doar 3 pini digitali de ieșire.

Interfața Arduino:

Pinii GND ai MAX7219 trebuie conectați la unul dintre pinii GND ai plăcii Arduino pentru a funcționa la același nivel de tensiune. Pini de alimentare pozitivă (+5V/Vcc) pot fi conectați direct la placa Arduino numai pentru un număr limitat de LED-uri. Cele trei linii de semnal (DIN, CLK, Load(/CS)) pot fi conectate la oricare trei ieșiri digitale ale plăcii Arduino.

Din datasheet-ul pentru MAX7219 se poate vedea că driverele pot fi cascadate prin conectarea semnalului DOUT de la un cip la DIN pe cipul următor. Semnalele CLK și Load(/CS) trebuie conectate în paralel cu fiecare MAX7219. Nu există o limită strictă a câtor drivere pot fi cascade în acest fel. Cu toate acestea, interfața SPI nu este capabilă de verificare a erorilor de date transmise, astfel încât sunteți deja limitați de lungimea cablurilor care trec de la un MAX7219 la celălalt.

Intern, MAX7219 multiplexează rândurile matricei.

Multiplexarea înseamnă: Driverele trec rapid între rândurile matricei (aproximativ 800 de ori pe secundă). Aceasta dă impresia că toate LED-urile sunt în mod constant aprinse, în timp ce în realitate ele clipesc foarte, foarte repede. Marele avantaj este că în fiecare moment nu mai mult de 8 LED-uri (un rând) sunt aprinse.

Un display tip cifră cu 7 segmente nu este altceva decât 8 LED-uri (7 segmente și punct) montate într-un mod special. Pentru afișajele cu 7 segmente funcționează doar tipul cu catod comun. Nu există o modalitate ușoară de a face să funcționeze un afișaj cu 7 segmente cu anod comun cu MAX7219. Etichetele pinilor de pe cifrele cu 7 segmente corespund numelui pinilor de pe un MAX7219, cu pinul catod comun conectat la unul dintre pinii Dig0-7.

Un singur driver LED MAX7219 poate controla 64 de LED-uri. Biblioteca suportă până la 8 driveri MAX7219 în lanț.

Inițializarea Dispozitivelor:

Atunci când este creat un nou LedControl, biblioteca va inițializa hardware-ul cu următoarele setări:

- afișajul este gol

- luminozitatea este setată la minim

- dispozitivul este în modul de economisire a energiei

- este activat numărul maxim de cifre pe dispozitiv

Un afișaj oprit este probabil ceea ce doriți la pornire. Dar cu luminozitatea la minim și dispozitivul în modul de oprire, niciun LED nu se va aprinde în configurația de pornire. Majoritatea utilizatorilor vor face propria lor inițializare în funcția "setup()".Am utilizat librăria pentru a controla cele 4 matrici(liftul și alte afișaje) și afișajul de 4 cifre a câte 7 segmente(scorul).

Coduri:

Scor:

```
LedControl lcScor = LedControl(12, 11, 10, 1);


int Digits[10] = {B01111110, B00110000, B01101101, B01111001,
                  B00110011, B01011011, B01011111, B01110000,
                  B01111111, B01111011};
void setup() {
int dig2, dig3;
  lcScor.shutdown(0, false);
  lcScor.setIntensity(0, 5);
  lcScor.clearDisplay(0);
   turnOffDisplay();
}
void displayNumber(int number) {
 if (number < 0 || number > 99) {
   return;
 }
 dig2 = number / 10;
 dig3 = number % 10;
```

```
  // folosesc doar digiturile din centru scorul nedepășind vreodată 2 cifre.  lcScor.setRow(0, 1, Digits[dig2]);
  lcScor.setRow(0, 2, Digits[dig3]);
}
void turnOffDisplay() {
  for (int digit = 0; digit < 4; digit++) {
    lcScor.setChar(0, digit, ' ', false);  // stinge fiecare segment de la fiecare digit  }
}
```

Matrici:

```
LedControl lcMatrix[4] = {
  LedControl(DIN_PIN, CLK_PIN, CS_PIN, 4), // 1 driver
  LedControl(DIN_PIN, CLK_PIN, CS_PIN, 4), // 2 driver
  LedControl(DIN_PIN, CLK_PIN, CS_PIN, 4), // 3 driver
  LedControl(DIN_PIN, CLK_PIN, CS_PIN, 4)  // 4 driver
};
// Define the pattern example
byte human[] = {
  0b00010000,
  0b00001000,
  0b11101010,
  0b00111111,
  0b11101010,
  0b00001000,
  0b00010000,
  0b00000000
};
const uint8_t IMAGES_LEN = sizeof(IMAGES)/8; // patterns defined in original code
void setup() {
  for (int driverIndex = 0; driverIndex < 4; driverIndex++) {
    lcMatrix[driverIndex].shutdown(driverIndex, false);
    lcMatrix[driverIndex].setIntensity(driverIndex, 4);
    lcMatrix[driverIndex].clearDisplay(driverIndex);
```

```
  }
}
void displayHumanOnMatrix(int index){
  for (int i = 0; i < 4; i++) {
      if (i != index) {
        lcMatrix[i].clearDisplay(i);
      }}
  lcMatrix[index].setRow(index, 0, human[0]);
  lcMatrix[index].setRow(index, 1, human[1]);
  lcMatrix[index].setRow(index, 2, human[2]);
  lcMatrix[index].setRow(index, 3, human[3]);
  lcMatrix[index].setRow(index, 4, human[4]);
  lcMatrix[index].setRow(index, 5, human[5]);
  lcMatrix[index].setRow(index, 6, human[6]);
  lcMatrix[index].setRow(index, 7, human[7]);
}
void displayPatternOnMatrix(int matrixIndex, int patternIndex) {
  if (matrixIndex >= 0 && matrixIndex < NUM_MATRICES) {
    for (int i = 0; i < NUM_MATRICES; i++) {
      if (i != matrixIndex) {
        lcMatrix[i].clearDisplay(i); // Turn off all other matrices
      }
    }
    for (int row = 0; row < 8; row++) {
      lcMatrix[matrixIndex].setRow(matrixIndex, row, IMAGES[patternIndex][row]);
    } }}
```

3.1.2.3. Keypad.h [xv] & pgmspace.h [xvi]

Biblioteca Keypad este o resursă esențială pentru dezvoltatorii care lucrează cu platforma Arduino și doresc să integreze tastaturi de tip matrice în proiectele lor. Această bibliotecă

facilitează gestionarea tastelor și oferă funcționalități de abstractizare hardware pentru a îmbunătăți claritatea și structura codului.

Versiunea 3.0 a bibliotecii, lansată în iulie 2012, aduce suport pentru apăsări multiple de taste, ceea ce face Keypad și mai versatilă în numeroase aplicații.

Principalele funcții ale bibliotecii includ inițializarea tastelor, așteptarea unei apăsări de tastă (cu blocarea celorlalte operațiuni), obținerea tastelor apăsate, precum și obținerea stării curente a tastelor (cum ar fi apăsată, eliberată sau ținută apăsată). De asemenea, biblioteca oferă control asupra timpilor de reacție pentru apăsarea și eliberarea tastelor.

Biblioteca Keypad gestionează rezistențele pull-up interne și asigură că toate pinurile coloanelor neutilizate sunt într-o stare de înaltă impedanță, eliminând astfel nevoia de rezistoare sau diode externe.

Eu am utilizat librăria atât pentru controlul jocului cât și pentru recunoașterea unor cifruri.

Având multiple comenzi și cifruri a fost nevoie să folosesc alături de librăria Keypad.h și PROGMEM din librăria pgmspace.h, pentru o utilizare eficientă a memoriei, pe care o voi detalia mai jos, urmând să prezint codul propriu-zis.

PROGMEM:

Stocarea datelor în memoria flash (program) în loc de SRAM este o opțiune importantă în dezvoltarea pentru plăcile Arduino. Această tehnică permite economisirea spațiului prețios de memorie SRAM și este utilă în special atunci când avem nevoie să stocăm cantități semnificative de date.

Cuvântul cheie PROGMEM este un instrument esențial în acest context. Acesta este un modificator de variabilă folosit cu tipurile de date definite în pgmspace.h, o bibliotecă importantă pentru gestionarea memoriei program. Când utilizăm PROGMEM compilatorul stochează informațiile respective în memoria flash a plăcii Arduino, în loc să le aloce în memoria SRAM, care este mai limitată.

Pentru a utiliza PROGMEM, este necesară includerea bibliotecii pgmspace.h, lucru care se face automat în versiunile moderne ale Arduino IDE.

PROGMEM se utilizează pentru stocarea datelor în memorie flash, dar este mai util atunci când avem un bloc mai mare de date de stocat, cum ar fi un array. Totuși, utilizarea PROGMEM necesită o abordare în două etape. După ce datele sunt plasate în memoria Flash, este necesar să utilizăm funcții speciale, de asemenea definite în biblioteca pgmspace.h,

pentru a citi datele din memoria programului înapoi în memoria SRAM, astfel încât să le putem utiliza eficient în cadrul programului.

Acesta este codul care include doar un minigame din cele 8 de ales și un singur cifrul și doar o declarare a unei singure variabile PROGMEM și fară a avea toate verificările de stări pentru a avea o variantă succintă :

```
const char password_2[] PROGMEM = "100";
bool cod2Used = 0;
String input_password = "";
bool stateCode = 0;
bool keypadEnabledCodApartamente = false;
void proccesKeypadInputStart()
{
 if (keypadEnabledCodApartamente)
   return;
 char key = keypad.getKey();
 unsigned long currentMillisLed = millis();
 if (key){
   digitalWrite(LED_PINKEY, HIGH);
   ledOn = true;
   ledStartTime = currentMillisLed;
   //Serial.println(key);
    if(key == '5') { gameState = initiala; keypadEnabledCodApartamente = false; trackPlayed = false; stareBoxa = initialaSound;  }
    if(key == '6') { /* gameState = apartament;*/}
    if(key == '*') {gameState = joc; turnOffAllLEDs();
   turnOffDisplay();trackPlayed = false; stareBoxa = challange; keypadEnabledCodApartamente = true; digitalWrite(LED_PINKEY, LOW);}
    if(key == '0') {gameState = demo; scorDemo = 20; trackPlayed = false; stareBoxa = demoAp; keypadEnabledCodApartamente = true;   digitalWrite(LED_PINKEY, LOW);}
    if(key == '#') {gameState = instructiuni; trackPlayed = false; stareBoxa = instructiuniSound;}
 }
 if (!key && ledOn && currentMillisLed - ledStartTime >= LED_ON_DURATION) {
   digitalWrite(LED_PINKEY, LOW);
```

```
    ledOn = false;
  }
}
void processKeypadInputCodApartamente() {
    if (!keypadEnabledCodApartamente) {
    return; // nu procesează input când e disabled
  }
  char key = keypad.getKey();
  unsigned long currentMillisLed = millis();
    if (key){
    digitalWrite(LED_PINKEY, HIGH);
    ledOn = true;
    ledStartTime = currentMillisLed;
   if (stateCode == 0)
    {
     if(inGameState == intrebFinal){
       if(key != '6' && key!=0  && key!= '*' && key != "#" )
       { scor = scor/2; if (currentTime - startLastTime >= alesDuration) { inGameState =
gameOver;} }
       else if(key == '6') { scor = scor + 3*sumaApNeviz();if (currentTime - startLastTime >=
alesDuration) { inGameState = win;}}
      }
     if(key == '8') {
   displayHumanOnMatrix(max1);
   digitalWrite(LED_PIN8R, HIGH);
}
     if(key == '9') {
       if(inGameState != alegAp)
       pauseMP3();
       else
       {inGameState = cifru; trackPlayed= false;}
      }
         if(key == '0'){if(inGameState == alegAp) inGameState = intrebFinal;
disableKeypadCodApartamente(); key = 'none'; trackPlayed = false;}
```

```
    if(key == '7'){ resumeMP3();}
    if(key == '1') {
      if(inGameState == raspuns) {
     currentChoice = aleg1;
     if(v[currentApNr] == 1){ scor = scor + 2; stareBoxa = raspunsCorect; trackPlayed =
false;} else {if(scor >= 4) scor= scor - 4; else scor = 0; stareBoxa = raspunsGresit;trackPlayed
= false;}} }
    if(key == '2') {
       if(gameState == joc && inGameState != raspuns &&inGameState != intrebFinal){
     currentMatrix= max4;
     displayHumanOnMatrix(max4);
     digitalWrite(LED_PIN2R, HIGH);
     digitalWrite(LED_PINI1, LOW);
     currentAp = apartament2;
     currentApNr = 1;
     trackPlayed = false;
     startLastTime = currentTime;
     if(ledOn) digitalWrite(LED_PINKEY, LOW);
     vizAp[1] = 1;
    }
    if(inGameState == raspuns) {
     currentChoice = aleg2;
     if(v[currentApNr] == 2){ scor = scor + 2; stareBoxa = raspunsCorect; trackPlayed =
false;} else {if(scor >= 4) scor= scor - 4; else scor = 0; stareBoxa = raspunsGresit;trackPlayed
= false;}}}
    if(key == '3') {
    if(gameState == demo)
    {
   trackPlayed = false;
   stareBoxa = demoRasp3;
   startLastTime = currentTime;
   scorDemo=scorDemo + 2;
   if(ledOn) digitalWrite(LED_PINKEY, LOW);
   turnOffDisplay();
```

```
 }
   if( key!='*' && stateCode == 0 && inGameState == alegAp)
{ inGameState = apartament;}
}
if(key == '*') {
  stateCode = 1;
  input_password = ""; // Reset the input password
} else if(key == '#') {
   displayHumanOnMatrix(currentMatrix);
  if (compareProgmemString(redareInreg, input_password.c_str())){ if(gameState ==
demo){ stareBoxa = demoAp; trackPlayed = false; startLastTime = currentTime;
if(ledOn) digitalWrite(LED_PINKEY, LOW);}
if(gameState == joc){ inGameState = apartament; trackPlayed = false; startLastTime =
currentTime;currentChoice = 0; }
}
    if (compareProgmemString(raspunsIar, input_password.c_str())){ if(gameState ==
demo){ stareBoxa = demoIntrebare; trackPlayed = false; startLastTime = currentTime;
if(ledOn) digitalWrite(LED_PINKEY, LOW);}
else {inGameState = intrebare; trackPlayed = false; startLastTime = currentTime;
currentChoice = 0; }
}
    if (compareProgmemString(altAp, input_password.c_str())){
      inGameState = alegAp; trackPlayed = false; startLastTime = currentTime;
currentChoice = 0;
    }
if(inGameState == cifru)
  checkPassword();
  if(stareBoxa == demoRasp3 || stareBoxa == demoCod2)
   checkPasswordDemo();
   stateCode = 0;
}
else {
  if(inGameState == cifru){
  displayPatternOnMatrix(currentMatrix,key-'0');
  }
```

```
      input_password += key; // Append new character to input password string}}
  if (!key && ledOn && currentMillisLed - ledStartTime >= LED_ON_DURATION) {
    digitalWrite(LED_PINKEY, LOW);
    ledOn = false;}}
void disableKeypadCodApartamente() {
  keypadEnabledCodApartamente = false;
}
void enableKeypadCodApartamente() {
  keypadEnabledCodApartamente = true;
}
void checkPassword() {
  if(inGameState == cifru){
  if(compareProgmemString(password_2, input_password.c_str())||
compareProgmemString(password_7, input_password.c_str())
||compareProgmemString(password_5, input_password.c_str()) ||
compareProgmemString(password_6, input_password.c_str()) ) {
   if(compareProgmemString(password_2, input_password.c_str()) ){
     if(cod2Used == 0){
     scor = scor + 2;
     digitalWrite(LED_PINI1,HIGH);}
 if (currentTime - startLastTime >= titluDuration) { cod2Used = 1;} }
  } else {
    if(scor >= 2)
    scor = scor -2;
  }
  input_password = ""; // Reset the input password
   if (currentTime - startLastTime >= titluDuration) {
  inGameState = alegAp;
  startLastTime = currentTime;
  trackPlayed = false; }}}
void checkPasswordDemo() {
  if(compareProgmemString(codDemo, input_password.c_str())) {
    scorDemo = scorDemo + 2;
    stareBoxa= demoCodCorect;
```

```
    trackPlayed = false;
    startLastTime = currentTime;
    if(ledOn) digitalWrite(LED_PINKEY, LOW);
   } else {
    scorDemo = scorDemo - 2;
    stareBoxa= demoCodGresit;
    trackPlayed = false;
    startLastTime = currentTime;
    if(ledOn) digitalWrite(LED_PINKEY, LOW);
   }
  input_password = ""; // Reset the input password
}
bool compareProgmemString(const char* str1, const char* str2) {
  uint8_t i = 0;
  while (true) {
    char c1 = pgm_read_byte(str1 + i);
    char c2 = str2[i];
    if (c1 == '\0' && c2 == '\0') {
      return true; // Both strings are identical
    }
    if (c1 != c2) {
      return false; // Strings are different
    }
    i++; }}
```

3.1.2.3. DFPlayerMini_Fast.h [xvii]

Această bibliotecă, numită DFPlayerMini_Fast, oferă funcționalitate pentru interacționarea cu playerul MP3 DFPlayerMini folosind platforma Arduino. Acest driver a fost creat pentru a permite comunicarea și controlul ușor al playerului MP3 YX5200-24SS, folosind doar doi pini pentru transmiterea și recepția datelor prin UART.

Funcții Principale:

- “begin(Stream &stream, bool debug = false, unsigned long threshold = 100)”: Această funcție configurează clasa pentru comunicare cu playerul MP3. Parametrii includ referința

către instanța Serial (hardware sau software) utilizată pentru comunicarea cu playerul MP3, o opțiune pentru a afișa informații de depanare în consola serială și un prag de timp pentru timpul maxim de așteptare a răspunsului de la player.

- “play(uint16_t trackNum)”: Această funcție permite redarea unei piese muzicale specifice prin specificarea numărului acelei piese.

- “playFromMP3Folder(uint16_t trackNum)”: Similar cu funcția anterioară, această funcție permite redarea unei piese din folderul "MP3" al playerului MP3.

- “playAdvertisement(uint16_t trackNum)”:Această funcție permite întreruperea piesei curente și redarea unei piese publicitare specifice.

- “volume(uint8_t volume)”:Prin această funcție, volumul este setat la o valoare specifică între 0 și 30.

- “printStack(stack _stack)”: Această funcție este utilă în scopuri de depanare și permite afișarea întregului conținut al unui pachet de configurație/comandă specific pentru playerul MP3.

Această bibliotecă este destinată utilizării cu playerul MP3 DFPlayerMini pe platforma Arduino. Playerul utilizează comunicarea UART, necesitând doar doi pini (TX + RX) pentru interfațare cu plaforma Arduino. Totuși, pentru o funcționare optimă, este recomandată folosirea unei surse de alimentare externe și a unui amplificator audio.

Această bibliotecă depinde de prezența bibliotecii FireTimer.h. Biblioteca a fost scrisă de Power_Broker în calitate de hobbyist și reprezintă o soluție rapidă și ușor de înțeles pentru utilizarea modulului DFPlayer Mini MP3 de la DFRobot.com. Aceasta reprezintă o îmbunătățire semnificativă a vitezei de execuție comparativ cu biblioteca standard furnizată de DFRobot.com.

Această bibliotecă dispune de funcții pentru a controla playerul MP3 DFPlayerMini într-un mod rapid și eficient, facilitând astfel dezvoltarea de proiecte ce implică redarea de muzică sau sunete.

În vederea unei corecte funcționări a playerului MP3 DFPlayer Mini, este esențial ca fișierele audio (MP3) să fie copiate pe cardul micro SD într-o anumită ordine. Ordinea în care aceste fișiere sunt transferate pe card va dicta secvența lor de redare. De exemplu, funcția "play(1)" va reda primul fișier MP3 copiat pe card, funcția "play(2)" va reda al doilea fișier, și tot așa.

Acest lucru înseamnă că organizarea și copierea adecvată a fișierelor pe cardul micro SD sunt de o importanță crucială. Fișierele trebuie să fie structurate astfel încât secvența lor de redare să se potrivească scopului proiectului.

Utilizez algoritm de mașină de stări(detaliată în capitolul 4) alături de o funcție non-blocking și un parametru de verificare dacă se redă sau nu ceva. Puteam utiliza și funcția implicita din librărie isPlaying() însă mi s-a parut mai intuitivă opțiunea mea. De asemenea folosesc funcțiile resume() și stop() din librărie.

Cod :

```
//audio files - stareBoxa values

//exemplu declarare:

uint8_t stareBoxa = 1;
```

Funcții utilizate:

```
void pauseMP3() {
  if(isPaused == 0){
  myMP3.pause();
  isPaused = 1;
  }
}
void resumeMP3() {
  if(isPaused == 1){
  myMP3.resume();
  isPaused = 0;
  }
}
//  playSpecificTrackOnce(#);
void playSpecificTrackOnce(uint16_t trackNum)
```

```
{
 if (!trackPlayed)
 {
  myMP3.play(trackNum);
  trackPlayed = true;
 }
}
```

3.1.2.4 `EEPROM`.h [xviii]

Microcontroller-ul din placa Arduino și cele bazate pe AVR Genuino dispun de o zonă de memorie numită `EEPROM`, care păstrează informații chiar și atunci când dispozitivul este oprit, asemenea unui mic disc dur. Biblioteca `EEPROM` vă permite să citiți și să scrieți acești octeți din `EEPROM`.

Pe diferitele microcontrolere suportate de diversele plăci Arduino și Genuino, capacitatea `EEPROM` variază: 1024 de octeți pe ATmega328P, 512 octeți pe ATmega168 și ATmega8, 4 KB (4096 de octeți) pe ATmega1280 și ATmega2560. Plăcile Arduino și Genuino 101 dispun de un spațiu de memorie emulat `EEPROM` de 1024 de octeți.

Funcțiile disponibile în această bibliotecă sunt:

- “read(address)”: Citește un octet din `EEPROM`.

- “write(address, value)”: Scrie un octet în `EEPROM`.

- “update(address, value)”: Scrie un octet în `EEPROM` numai dacă acesta diferă de valoarea deja salvată la aceeași adresă.

- “get(address, data)”: Citește orice tip de date sau obiect din `EEPROM`.

- “put(address, data)”: Scrie orice tip de date sau obiect în `EEPROM`.

- “`EEPROM`[address]”: Permite folosirea identificatorului `EEPROM` ca un array pentru a citi și scrie direct în celulele `EEPROM`.

- “length()”: Returnează numărul de celule din `EEPROM`.

Am utilizat această librărie pentru a reține highscore-ul.

```
void saveHighScore(int currentScore) {
  int previousHighScore;
  `EEPROM`.get(highScoreAddress, previousHighScore); // Read the previous high score from `EEPROM`
    if (currentScore > previousHighScore) {
   isHighScore =1;
   // If the current score is higher than the previous high score, update it
   `EEPROM`.put(highScoreAddress, currentScore);
  }
}
```

3.1.2.4 FastLED.h [xix]

FastLED reprezintă o bibliotecă Arduino de mare performanță, care este cunoscută pentru rapiditatea, eficiența și ușurința sa de utilizare în programarea benzilor și pixelilor LED adresabili, precum WS2810, WS2811, LPD8806, Neopixel și multe altele. Această bibliotecă este o alegere de încredere pentru mii de dezvoltatori angajați într-o varietate de proiecte artistice și hobby-uri, dar și pentru producția de produse comerciale.

FastLED se remarcă prin compatibilitatea sa excelentă, oferind suport pentru o gamă largă de LED-uri populare, inclusiv Neopixel, WS2801, WS2811, WS2812B, LPD8806, TM1809 și altele. Este compatibilă cu o serie de plăci Arduino și variante compatibile, inclusiv cele bazate pe microcontrolere AVR și ARM.

Caracteristicile notabile ale bibliotecii FastLED includ:

- suport complet pentru culorile HSV, precum și RGB tradiționale.
- posibilitatea de a regla cu precizie luminozitatea master (fără a o distruge), ceea ce influențează luminozitatea, consumul de energie și durata de viață a bateriilor.
- funcții matematice și operații de memorie cu o performanță de până la 10 ori mai rapidă decât cele oferite de bibliotecile Arduino standard.
- utilizatori care împărtășesc sfaturi, idei și oferă asistență.
- istorie lungă de dezvoltare continuă, desfășurată pe parcursul mai multor ani.
- atenție deosebită acordată eficienței, cu o dedicație pentru performanță și pentru a asigura culori RGB uniforme și frumoase.

FastLED interacționează cu componenta WS2812B (și cu benzi LED similare) prin intermediul unui protocol specific, protocolul WS2812B (sau "protocolul Neopixel"), pe care urmează să-l detaliez în continuare.

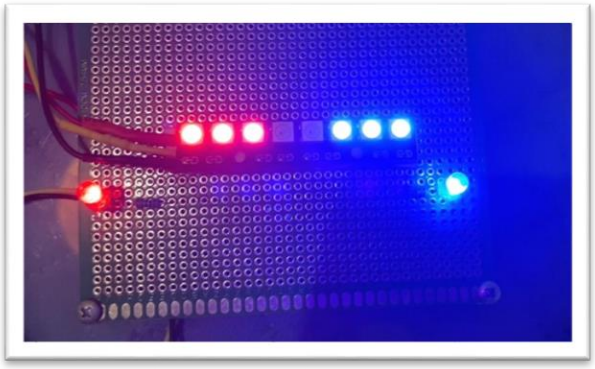
Cod:

```
// Funcție pentru simularea luminilor de poliție

void politie() {
  static int faza = 0; // 0 pentru Roșu, 1 pentru Albastru
  static unsigned long previousMillisPolitie = 0;
  const long intervalPolitie = 160; // Interval pentru schimbarea luminilor de poliție în milisecunde
  unsigned long currentMillis = millis();
  // Verificăm dacă a trecut intervalul pentru schimbarea luminilor de poliție
  if (currentMillis - previousMillisPolitie >= intervalPolitie) {
    previousMillisPoliție = currentMillis;
    // Iterăm prin prima jumătate a benzii LED
    for (int i = 0; i < NUM_LEDS / 2; i++) {
      // Dacă suntem în faza 0 (Roșu)
      if (faza == 0) {
        leds[i] = CRGB::Red; // Setăm culoarea LED-ului la Roșu
        leds[i + NUM_LEDS / 2] = CRGB::Black; // Al doilea LED rămâne stins
        digitalWrite(LED_PINPOL1, HIGH); // Aprindem LED-ul 1
        digitalWrite(LED_PINPOL2, HIGH); // Aprindem LED-ul 2
      } else { // Altfel, suntem în faza 1 (Albastru)
        leds[i] = CRGB::Black; // Primul LED rămâne stins
        leds[i + NUM_LEDS / 2] = CRGB::Blue; // Setăm culoarea LED-ului la Albastru
        digitalWrite(LED_PINPOL1, LOW); // Stingem LED-ul 1
        digitalWrite(LED_PINPOL2, LOW); // Stingem LED-ul 2
      }
    }
    // Setăm LED-urile din mijloc la negru
    leds[NUM_LEDS / 2 - 1] = CRGB::Black;
    leds[NUM_LEDS / 2] = CRGB::Black;
```

```
    FastLED.show(); // Actualizăm benzile LED
    // Schimbăm faza
    faza = 1 - faza;
  }
}
// Funcție pentru stingerea tuturor luminilor de poliție
void oprește ToateLuminileDePolitie() {
  for (int i = 0; i < NUM_LEDS; i++) {
    leds[i] = CRGB::Black; // Setăm toate LED-urile la negru
  }
  FastLED.show(); // Actualizăm benzile LED
  digitalWrite(LED_PINPOL1, LOW); // Stingem LED-ul 1
  digitalWrite(LED_PINPOL2, LOW); // Stingem LED-ul 2
}
```

Girofar poliție - Fig.12.1 & 12.2



3.2. Protocoale și interfețe de comunicare

3.2.1. UART[xx]

"UART" reprezintă un circuit fizic esențial în dispozitive electronice diverse, cum ar fi microcontrolere sau circuite integrate autonome. Acesta diferențiază față de protocoalele de

comunicare precum SPI și I2C, deoarece funcționează ca un intermediar fizic specializat pentru transmiterea și recepționarea datelor în format serial.

În comunicarea UART, două module UART pot stabili comunicare directă între ele. Modulul UART de transmitere convertește datele paralele furnizate de un procesor, în format serial, și le trimite către modulul UART de recepție. Transferul de date se realizează de la pinul de transmitere, Tx, al modulului UART de transmitere către pinul de recepție, Rx, al modulului UART de recepție.

Deși protocoale precum SPI și USB sunt utilizate pentru comunicații rapide, UART este preferat atunci când viteza de transfer nu este o prioritate. Acesta este o opțiune de comunicare economică și simplă, implicând un singur transmițător și un singur receptor.

UART-ul se compune din două componente principale: transmițătorul și receptorul. Transmițătorul include un registru de păstrare a datelor pentru informațiile care urmează să fie transmise, un registru de deplasare a transmisiei și logica de control. La fel, receptorul conține un registru de păstrare a datelor primite, un registru de deplasare a recepției și logica de control. Ambele părți sunt sincronizate cu un generator de rate de baud, care stabilește viteza la care datele trebuie să fie transmise și recepționate.

Procesul de transmitere și recepție implică utilizarea unui bit de start, un bit de stop și parametri de sincronizare pentru a coordona datele între transmițător și receptor. Datele inițiale sunt în formă paralelă, dar pentru a fi transmise în formă serială, sunt convertite cu ajutorul unui convertor paralel-serial, care adesea utilizează flip-flop-uri D sau latch-uri.

Formatul protocolului UART începe cu un bit de start '0', care inițiază transferul datelor seriale, și se încheie cu un bit de stop. De asemenea, poate include un bit de paritate, fie par (reprezentat de '0', cu un număr par de biți la 1) sau impar (reprezentat de '1', cu un număr impar de biți la 1).

În timpul transmiterii, datele sunt trimise folosind o singură linie de transmitere (TxD), unde '0' este considerat spațiu, iar '1' este denumită stare de marcă. Transmițătorul trimite un singur bit la un moment dat, cu o întârziere specifică între biți. În procesul de recepție, linia RxD (receptor) este folosită pentru primirea datelor.

UART prezintă multiple avantaje, cum ar fi utilizarea a doar două fire, absența necesității unui semnal de ceas separat, posibilitatea de a verifica erorile cu ajutorul bitului de paritate, și flexibilitatea de a modifica structura pachetului de date, cu condiția ca ambele părți să fie configurate în consecință. De asemenea, acesta este bine documentat și larg utilizat.

Cu toate acestea, UART are și unele dezavantaje, cum ar fi limitarea dimensiunii cadrului de date la maximum 9 biți, incompatibilitatea cu sistemele cu mai mulți periferici sau mai multe controlere, și necesitatea ca ratele de baud ale fiecărui UART să difere cu cel mult 10%.

3.2.2. SPI - MAX7219 LED driver [xxi]

Protocolul SPI se distinge prin faptul că necesită conexiuni fizice scurte (pentru a minimiza interferențele) și implică un controller care acționează ca punct central pentru gestionarea comunicării cu dispozitivele periferice.

Elementele principale ale unei conexiuni SPI includ:

CIPO (Controller In Peripheral Out): Acesta este canalul prin care datele sunt trimise de la dispozitivele periferice către controller.

COPI (Controller Out Peripheral In): Această linie este folosită pentru transmiterea datelor de la controller către dispozitivele periferice.

SCK (Serial Clock): Semnalul de ceas, furnizat de controller, coordonează sincronizarea transmiterii datelor între controller și fiecare dispozitiv periferic.

CS (Chip Select): Fiecare dispozitiv periferic are un pin CS pe care controllerul îl utilizează pentru alegerea dispozitivului cu care se va face comunicarea. Când CS este la nivel logic scăzut, dispozitivul periferic comunică cu controllerul, iar când este la nivel logic înalt, dispozitivul periferic ignoră comunicarea. Această caracteristică permite conectarea mai multor dispozitive SPI la aceleași linii CIPO, COPI și SCK. În cazul în care există doar un pin CS, dispozitive multiple pot fi conectate în serie

O caracteristică distinctivă a SPI este capacitatea sa de a permite transferul continuu de date, fără întreruperi, în fluxuri de biți variabile.

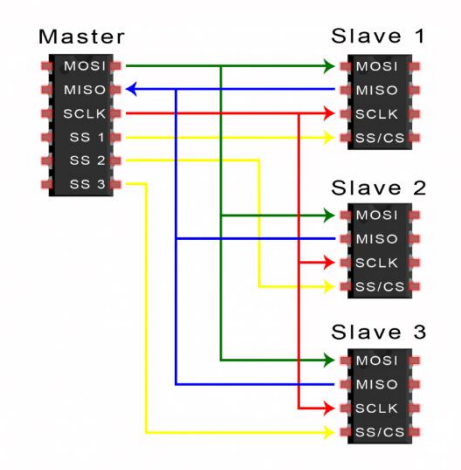
Periferice Multiple:

SPI poate funcționa cu un controller și un periferic. De asemenea poate să controleze mai mulți periferici. Există două modalități principale de a conecta la un controller dispozitive multiple într-o configurație SPI.

Este important de menționat că termenul anterior "master/slave" a fost înlocuit cu "controller/peripheral" în contextul Arduino, pentru a evita conotațiile negative și pentru a reflecta mai bine rolurile dispozitivelor în comunicarea SPI. Va apărea strict în imagini dat utilității lor în explicarea procesului.

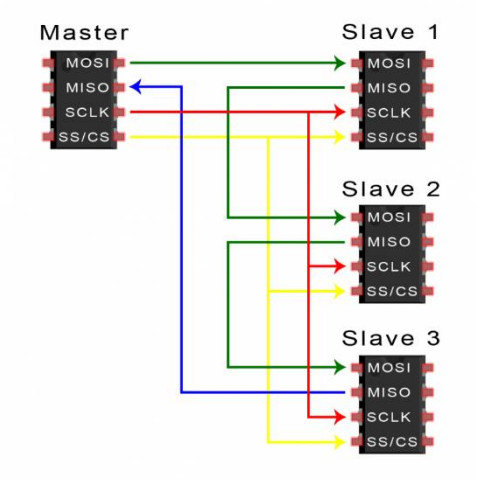
Perifericele pot fi conectate in paralel la existența mai multor pini CS (Fig.13).

Fig.13



Perifericele pot fi înlănțuite in serie la existența doar unui pin CS (Fig.14).

Fig.14



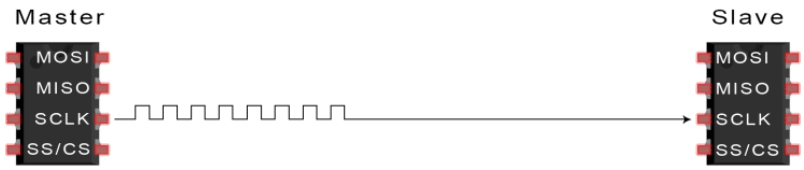
COPI și CIPO

Transmiterea datelor între controller și periferic se realizează într-un mod secvențial și serial prin intermediul liniilor COPI. Controllerul transmite date către periferic prin linia COPI, iar perifericul primește aceste date prin intermediul acestei linii. De obicei, datele trimise de la controller la periferic sunt structurate astfel încât cel mai semnificativ bit să fie transmis primul.

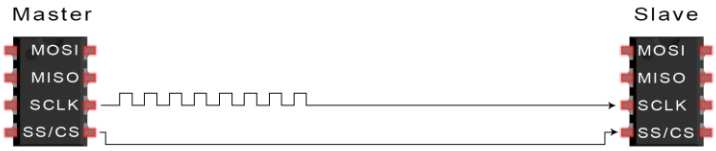
Pe lângă transmiterea datelor de la controller la periferic, perifericul are, de asemenea, capacitatea de a trimite date înapoi către controller prin linia CIPO. LSB este în general transmis primul.

PASII TRANSMISIEI DE DATE SPI:

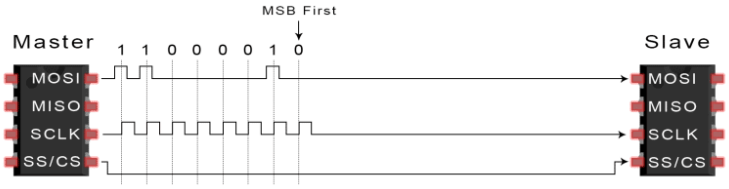
1. Controller-ul emite semnalul de ceas:



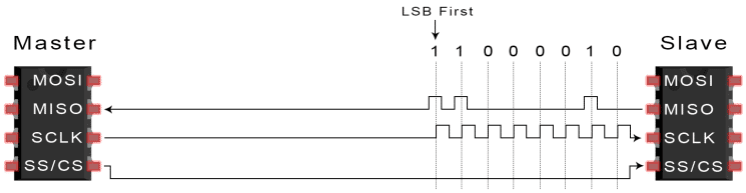
2. Controllerul comuta pinul CS la un nivel de tensiune scăzut, ceea ce activează perifericul:



3. Controller-ul trimite datele pe rând către periferic pe COPI. Biții sunt citiți de către periferic la momentul primirii:



4. Daca este necesar un răspuns, perifericul returnează datele pe rând către controller pe CIPO. Biții sunt citiți de către controller la momentul primirii:



Beneficii:

Transmiterea continuă a datelor- datorită lipsei pinilor stop și start

Adresarea perifericului se poate realiza printr-un sistem mai puțin complex, în contrast cu protocolul I2C (Inter-Integrated Circuit).

Viteză aproape dublă comparativ cu I2C.

Utilizează linii distincte pentru transmiterea (CIPO) și primirea (COPI) datelor, permițând comunicarea bidirecțională simultană.

Limitări:

Spre deosebire de I2C și UART (Universal Asynchronous Receiver/Transmitter), care folosesc mai puține conexiuni, SPI folosește mai multe fire.

Nu dispune de un mecanism de confirmare a succesului la recepția datelor, în contrast cu I2C care oferă o confirmare.

Nu include o formă de verificare a erorilor, spre deosebire de UART, care oferă această funcționalitate.

Suportă doar un singur controller într-o configurație dată, limitând extensibilitatea sistemului în comparație cu alte protocoale care pot gestiona mai multe controlere.

3.2.3. Protocolul WS2812B

Semnalul CH1 este un semnal binar provenit de la un microcontroler, având o lungime de 48 de biți, și este destinat controlului a două LED-uri WS2812B. Cu toate acestea, doar primele 24 de biți din acest semnal sunt transmiși ca semnalul CH2.
Fiecare LED WS2812B este capabil să afișeze culorile roșu, verde și albastru la 256 de niveluri de luminozitate. Acest control al luminozității este realizat prin intermediul unei secvențe binare de 8 biți pentru fiecare culoare, cu valori cuprinse între 0 și 255. Astfel, pentru a controla complet un singur LED, sunt necesari 24 de biți (8 pentru roșu, 8 pentru verde și 8 pentru albastru).
După ce primul LED primește datele, secvența de date pentru al doilea LED începe imediat cu culoarea verde, roșie și albastră. Acest proces continuă până când toate LED-urile sunt iluminate.
Primul LED, după ce primește informații pentru întregul lanț de LED-uri, transmite aceleași date mai departe, fără secvența la care a răspuns inițial, astfel transformându-se în primul element din lanț (din perspectiva sa).
Acest proces de transmitere continuă până când nu mai sunt secvențe binare de LED-uri de transmis.
În acest protocol, atât valoarea "unu" cât și "zero" au propriile lor intervale de timp pentru a fi semnalizate:
"Unu" este reprezentat prin menținerea liniei în stare înaltă pentru o anumită perioadă, urmată de trecerea liniei în stare scăzută pentru o altă perioadă specifică.
"Zero" este indicat prin menținerea liniei în stare scăzută pentru o anumită perioadă, urmată de revenirea liniei la stare înaltă pentru o altă perioadă corespunzătoare.
Acest proces se repetă pentru fiecare set de 24 de biți de culoare, apoi microcontrolerul trimite un semnal de resetare scăzut (cu o durată de 300 de microsecunde sau mai mult) pentru a indica

că trebuie să înceapă din nou. Notabil, cel mai semnificativ bit este transmis primul, iar fiecare semnal începe cu zero-uri pentru a asigura că există întotdeauna 8 biți în total pentru fiecare element de culoare GRB.

Capitolul 4

Prezentarea jocului

4.1. Ipostaze din timpul jocului - Mașină de stări

Despre mașini de stări în structura algoritmilor jocurilor:[xxii]

Mașina cu Stări Finite (FSM) este o tehnică veche în industria jocurilor, fiind folosită în jocuri clasice precum PACMAN și în jocuri moderne precum TOM RAIDER. În toate aceste jocuri, unul dintre obiectivele principale a fost să se facă personajele controlate de calculator mai inteligente. Deși există și tehnici mai avansate, FSM rămâne una dintre cele mai utilizate abordări pentru personajele non-jucător.

Pe măsură ce jocurile au început să devină din ce în ce mai mari, necesitatea de a avea un mediu de joc și personaje non-jucător (NPC) a crescut rapid. NPC-urile au fost adăugate în jocuri pentru a face interacțiunea cu mediul mai distractivă și mai realistă. În esență, NPC-urile sunt personaje care nu sunt controlate de jucător, ci sunt utilizate pentru a interacționa, ghida sau asista jucătorul în jocuri. Personajul jucabil interacționează cu NPC-urile, ceea ce face ca jocul nostru să pară real.

Una dintre metodele eficiente de control al NPC-urilor este utilizarea Mașinilor cu Stări Finite (FSM). FSM există în industria jocurilor de mult timp și este încă folosit în jocurile moderne de astăzi de către mulți dezvoltatori. FSM reprezintă mașini de stări simple în care diferite stări sunt conectate între ele printr-o serie de condiții. Dacă anumite condiții sunt îndeplinite, atunci trecem de la o stare la alta. În interiorul fiecărei stări putem realiza anumite acțiuni sau implementa algoritmi. În tot acest timp, NPC-ul nostru trebuie să fie într-o stare specifică.

Stările jocului:

Descriere în limbaj natural și cu bucăți din narațiune:

La inițierea jocului:

Apasă butonul "steluță" pentru a începe jocul.
Apasă butonul "diez" pentru a asculta instrucțiunile.
Apasă butonul "0" pentru a accesa demo-ul.

Starea Instrucțiuni:
Se redau instrucțiunile.

Starea Demo:
Începi demo-ul cu 20 de puncte.
Apasă butonul "8" pentru a ajunge la apartamentul 8. Se redă o poveste.
Apasă butonul "9" pentru a pune în pauză înregistrarea și "7" pentru a o relua.
Aici trebuie să faci o alegere. Presupunem că prima oară alegi "1", deci presupunem un răspuns greșit. Acum poți alege să redai înregistrarea pentru a te obișnui cu comanda, așa că apeși butonul "2" după ce asculți din nou. Apoi, apasă "8#*" pentru a reda înregistrarea.
Dacă alegi "2" (răspuns greșit), poți alege din nou să redai înregistrarea, așa că apeși "1*#*" pentru a alege din nou. Acum alegi "3".
Dacă alegi "3" (răspuns corect), poți alege alt apartament sau, dacă există, să introduci un cod. Presupunem că există un cod. Introdu codul "123". Nu uita că formatul este "steluță" + "cifră" + "cifră" + "cifră" + "diez". Codul este presupus greșit, așa că introduci acum codul "321" (corect). Nu primești nimic din plic, deoarece este doar un demo. Felicitări, ai terminat demo-ul. Mult succes în joc!

Starea Joc:
Începutul jocului:
Se aude un telefon fix sunând.
- Alo! Sunt Alina Ionescu, locuiesc pe strada Norocului 7, în Bloc 11, apartamentul 2. Mi-am găsit soțul mort înjunghiat în bucătărie. Vă rog să veniți cât mai repede!
- Imediat, doamnă, trimitem acum cel mai bun detectiv!
Se aude sirena de poliție.
Ajungi la blocul crimei, pe drum primești următoarea informație: Nimeni străin nu a intrat în bloc în ultimele zile, iar toți locatarii prezenți la momentul crimei sunt în clădire. Deduci astfel că vinovatul este sigur unul dintre locatari. Securizezi zona, anunți să se asigure că nu intră și nu iese nimeni și apoi mergi să investighezi.

PLanșa la început de joc - Fig.15

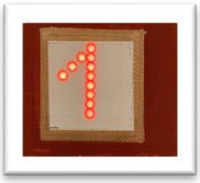
Starea Alegere Apartamente:

Selectează apartamentul la care vrei să mergi și apasă "0" pentru întrebarea finală sau "9" pentru a introduce un cifru.

Starea Cifru:

Apar cifrele pe matricea etajului curent la tastare ca în figura 16.

Fig.16



Dacă introduci cifrul greșit, pierzi 2 puncte și revii la starea de alegere a apartamentului.

Dacă introduci cifrul corect, primești 2 puncte, iar cifrul utilizată rămâne înregistrată pentru a evita punctarea dublă. În plus, becul corespunzător indiciului se aprinde ca în figurile 17.

Fig.17.1

Fig.17.2

Revii la starea de alegere a apartamentului.

Starea Apartament:

Se aprinde cu roșu becul din fața apartamentului, iar personajul se deplasează la acel etaj ca în figura 18.

Fig.18

Se redă o poveste.

Apoi, treci la starea întrebare (ascultă doar întrebarea) și apoi la starea răspuns.

Starea Răspuns:

Ai ales numărul #.

Starea Răspuns Greșit:

Răspuns greșit - pierzi 4 puncte.

Apeși "*1#" pentru a reveni la întrebare.

Apeși "*2#" pentru a reveni la alegerea apartamentului.

Ascultă din nou înregistrarea cu "*8#".

Starea Răspuns Corect:

Primești o nouă poveste și mesajul "Răspunsul este corect" + primești 2 puncte.

Revii la alegerea apartamentului.

Starea Întrebare Finală:

Cine este criminalul?

1. Ionescu Alina
2. Mogoșoaie Petre
3. Cantemir Maria

4. Popa Ion (răspuns corect)
5. Arici Andrei
6. Marinescu Letiția
7. Marinescu Laurențiu
8. Popescu Florența

Alegi un răspuns.
Starea Răspuns Final:
Ai ales numărul #.
Starea Răspuns Corect:
Se aude sirena.
S-a făcut arestul! Excelentă treabă, detectiv! (joc lumini - becuri verzi, altfel roșii)
Repornire (stare priză).

Dacă răspunsul este greșit, se aude:
- Game over, criminalul a scăpat. (joc lumini)
Repornire (stare priză).

Descriere a stărilor bazată pe variabilele utilizate în cod:

gameState - Starea principală a jocului:
priza: Jocul a început, dar nu a fost aleasă nicio opțiune încă. Se afișează un efect de lumini și sunet de crimă. Se trece la starea inițiala după încheierea sunetului.
initiala: Ecranul de pornire. Utilizatorul poate alege între opțiunile demo, joc sau instrucțiuni. După alegere, se trece la starea corespunzătoare.
instructiuni: Redă instrucțiunile jocului și trece înapoi la starea inițiala după ce sunetul instrucțiunilor se termină.
demo: Modul demonstrativ. Se efectuează o demonstrație a jocului.
joc: Starea de joc reală.
inGameState - Starea jocului efectiv (atunci când gameState este joc):
inceput: Așteptarea începerii jocului. Se trece la starea alegAp după sunetul de poliție.
alegAp: Alegerea unui apartament. Utilizatorul poate introduce un cod de apartament. Se trece la starea apartament când se introduce un cod valid.
apartament: Utilizatorul este într-un apartament și ascultă o întrebare.
intrebare: Așteptarea răspunsului la întrebare.

raspuns: Utilizatorul alege un răspuns și primește feedback.
cifrul: Introducerea cifrului de acces pentru apartamente.
intrebFinal: Așteptarea răspunsului final la întrebare.
povesteRaspCorect: Redă povestea după un răspuns corect.
gameOver: Redă mesajul de final "Game Over".
win: Redă mesajul de câștig "You Win".
stareBoxa - Starea player-ului audio:
Această variabilă indică ce melodie sau sunet este redat la un moment dat în joc, cum ar fi sunetul de crimă, sunetul de poliție, sunetul de început, instrucțiuni sau mesajele specifice întrebărilor și răspunsurilor.
currentAp și currentIntreb - Aceste variabile indică ce sunet sau mesaj trebuie să fie redat pentru apartamentul curent și întrebare.
currentChoice - Acesta stochează opțiunea aleasă de utilizator în timpul stării răspuns.
codXUsed - Variabilele pentru a urmări dacă codurile de acces pentru apartamentele 2, 5, 6 sau 7 au fost deja folosite.
input_password - Se ocupă de introducerea cifrelor pentru codul de acces al apartamentelor.

Atașez in anexă instrucțiunile, scenariul și codul din loop() care evidențiază stările.

Capitolul 5

Concluzii & Perspective

5.1. Concluzii

Jocul "Crime in the block" este o creație impresionantă ce demonstrează perfect cum tehnologia poate să se împletească armonios cu arta și creativitatea umană. Prin integrarea componentelor hardware într-un tablou interactiv, acest joc ne arată că tehnologia poate aduce viață în artă și poate stimula imaginația noastră într-un mod inovator.

Utilizarea tehnologiei în acest joc nu este doar o simplă adăugare, ci un element esențial al întregii experiențe. Componentele hardware, software-ul și narativa audio se îmbină perfect pentru a crea o aventură captivantă pentru jucători. De asemenea, abordarea modulară a hardware-ului permite adaptarea ușoară a jocului la noi tematici și limbi, făcându-l extrem de versatil.

Algoritmul complex de mașină de stări folosit în joc nu doar adaugă un element de personalizare, ci și o strategie unică pentru fiecare jucător. Acest lucru face ca experiența să fie mai profundă și mai interactivă, departe de jocurile liniare și previzibile.

Mai mult decât atât, jocul demonstrează cum tehnologia poate fi folosită pentru a crea interacțiuni realiste cu personaje non-jucător (NPC-uri), aducând o dimensiune nouă și mai autentică în lumea jocurilor. Prin utilizarea Mașinilor cu Stări Finite, jocul oferă jucătorilor control asupra modului în care explorează și rezolvă misterul, făcând experiența mai captivantă și personalizată.

În concluzie, "Crime in the block" nu este doar un joc, ci o demonstrație remarcabilă a modului în care tehnologia poate să sprijine și să îmbogățească creația artistică și experiențele interactive. Este un exemplu de cum inovația și imaginația pot colabora pentru a ne oferi noi modalități de a interacționa cu arta și cu lumea jocurilor, arătând că tehnologia poate să fie o unealtă puternică pentru exprimarea creativă și pentru captarea atenției publicului într-un mod autentic și captivant.

5.2. Perspective de dezvoltare

Optimizarea Algoritmului Mașină de Stări:

- Algoritmul mașinii de stări poate fi îmbunătățit prin reducerea stărilor redundante prin crearea de fișiere audio mai compacte.
- Se pot explora opțiunile de utilizare a unor funcții specifice predefinite din librării pentru a simplifica și eficientiza procesul.

Check-point-uri pentru Salvarea Progresului:

- Implementarea check-point-urilor pentru salvarea progresului jocului este o perspectivă utilă.
- Scorul, apartamentele vizitate și codurile utilizate pot fi salvate în memoria `EEPROM` după fiecare schimbare, iar o stare numită "joc neîncheiat" poate fi adăugată pentru a permite reluarea jocului chiar și după întreruperea curentului.

Șanse suplimentare la Întrebarea Finală:

- Introducerea unei variabile care să monitorizeze numărul de răspunsuri greșite și să ofere jucătorului o șansă suplimentară la întrebarea finală poate crește gradul de implicare și satisfacție al jucătorului.
- Se poate include o penalizare în puncte până în atingerea limitei maxime de răspunsuri greșite care va duce la pierderea jocului

Extinderea Bazei de Întrebări și Răspunsuri:

- Adăugarea unui număr mai mare de întrebări și variante de răspuns poate contribui la diversificarea experienței de joc.
- Mai multe opțiuni de interacțiune pot menține jucătorii implicați și interesați pe parcursul jocului.

Ajustarea Dimensiunii Jocului:

- În funcție de scopul utilizării, jocul poate fi adaptat prin rearanjarea componentelor pentru a utiliza spațiul mai eficient.
- De asemenea, se poate lua în considerare eliminarea sau adăugarea de componente pentru a ajusta nivelul de complexitate și durata jocului în funcție de preferințele utilizatorilor.

ANEXE:

Anexa1: Instrucțiuni

Urmează să intrați în pielea unui detectiv. Scopul acestui joc este de a rezolva misterul crimei petrecute din acest bloc prin cât mai puțini pași și cu cât mai puține erori, obținând astfel un scor cât mai mare. Trebuie să vă folosiți logica și de atenția la detalii, deci, mare grijă la răspunsuri și la ordinea interogării suspecților. Gândiți ca un detectiv adevărat, urmați firul logic, notați-vă pe ceva ce trebuie să rețineți, legați datele pe care le știți despre caz între ele.

Interogarea suspecților se face prin vizitarea apartamentelor, puteți vizita în orice ordine considerați a fi mai eficientă și oricând apăsați pe 0 puteți să accesați întrebarea finală și să alegeți persoana care considerați că a comis crima.

Aveți și 4 plicuri cu indicii. O să aveți de introdus un cifrul pe care îl obțineți din interogări pentru fiecare plic. Când cifrul e corect se aprinde becul de deasupra plicului și puteți lua indiciul. Acesta trebuie introdus după răspunsul la întrebare și înainte de a alege un nou apartament.

În 4 dintre interogări se va afla un cod, acela va fi cifrul. Ori va trebui să îl construiți ajutându-vă de un număr de 2 cifre și o cifră obținută dintr-un indiciu din alt plic, ori va fi direct un număr de 3 cifre. Nu va fi specificat faptul că e cod, o să facă parte din poveste în mod natural, deci mare atenție. Ca să știți că ați tastat bine urmăriți geamul de la liftul etajului la care sunteți, acolo vor apărea pe rând cifrele apăsate.

Fiecare cod corect introdus vă oferă 2 puncte, codurile greșite vă scad 2 puncte, deci grijă la tastare. Dacă observați că ați introdus greșit prima sau a doua cifră, apăsați pe diez înainte de a tasta 3 cifre. Codul va fi introdus în forma steluță cifră, cifră, cifră diez.

Fiecare răspuns corect la întrebările din timpul interogărilor vă oferă 2 puncte și bucăți noi de poveste în care se află indicii importante.

Fiecare răspuns greșit vă scade 4 puncte (dacă are de unde, altfel va trece la 0 puncte, nu veți avea punctaj negativ). Puteți răspunde la întrebări de oricâte ori doriți. Excepție face însă întrebarea finală unde aveți doar o șansă de răspuns. Dacă greșiți pierdeți jocul, criminalul scapă, scorul devine null și nu aflați răspunsul.

Controlul se face prin tastare. Pentru a vizita un apartament apăsați numărul său și veți observa că becul din dreptul lui va fi aprins roșu și povestea va începe. La un moment dat

vei avea o întrebare cu răspunsuri multiple. Vei introduce numărul corespunzător deciziei tale. Dacă răspunzi corect și extragi cifrul.

Prin apăsarea tastei 9 puteți pune pe pauză orice înregistrare și apăsând pe 7 o reporniți de unde a rămas. Puteți pune pe pauză de oricâte ori simțiți nevoia.

Pentru a reasculta o înregistrare apăsați tasta steluță urmată de cifra 8 apoi de diez.

Pe parcursul jocului vă puteți urmări scorul pe tabela de scor. Dacă ați câștigat și scorul dumneavoastră este record, la finalul jocului becurile apartamentelor se vor aprinde verzi, altfel vor fi roșii.

Anexa 2: Scenariu:

Apartament 2:

Ai ajuns la apartamentul cu scena crimei, apartamentul 2. Ușa este larg deschisă, se aude un plânset de femeie pe fundal. Îți faci simțită prezența.

- Bună ziua. Condoleanțe. Am ajuns și vă promit că vom găsi responsabilul și va plăti pentru moartea soțului dumneavoastră.

Soția dă din cap, dar nu spune nimic, continuă să plângă, vezi că încearcă să se adune, îi acorzi timp.

Până se calmează, inspectezi scena crimei alături de detaliile primite de la secția de poliție.

Decedatul este Ionuț Ionescu, soțul Alinei Ionescu care a declarat crima. A fost înjunghiat în propria bucătărie cu un cuțit ce nu pare a fi din colecția prezentă în bucătărie. Are o urmă de ruj roz, discretă, pe gât. Hainele miros a alcool și parfum de femeie. Rujul și parfumul nu par a fi cele ale soției. Se pot vedea două pahare murdare de vin și o sticlă de vin spartă în coșul de gunoi alături de chiștoace de țigară. O boxă mare pe podea. O foaie cu mai multe serii de numere, dintre care una încercuită de mai multe ori.

- Bună ziua, pot să vorbesc acum. Cu ce pot să fiu de ajutor?

- Întâi vreau să îmi spuneți ce făceați dumneavoastră în seara în care s-a produs crima și cum v-ați rănit la deget?

- Lucram, domnule polițist, și m-am tăiat în hârtie. Doar nu sugerați că mi-aș fi omorât eu soțul?

- Detectiv. Este protocolul, doamnă. Ați observat ceva ciudat când ați intrat în casă?

- Ușa era deschisă, soțul meu era pe jos, mort, totul este ciudat la asta.

- Mă refeream la ceva ce ar indica cine ar fi putut fi responsabil. De exemplu, ce știți de cuțit? Am remarcat că nu se potrivește colecției din bucătăria dumneavoastră. Sau faptul că era îmbrăcat așa elegant, deși dumneavoastră susțineți că era singur acasă. Cu cine ar fi putut să bea acel pahar de vin? Este clar că nu a fost singur.

- Cuțitul nu este al nostru, nu știu ce este cu el. Soțul meu ar fi știut, el se ocupa cu gătitul, îi plăcea enorm, a făcut vreo 100 de rețete. Legat de haine, azi era aniversarea noastră. Cred că imi pregătise ceva frumos și eu iar am ratat din cauză că am ales să stau peste program la serviciu. Probabil a băut el din ambele pahare când a vazut că nu mai vin și a spart sticla de supărare. Dacă veneam la timp, vai, poate mai era încă în viață!!

- Nu vă învinovățiți. Mulțumesc pentru informații, aveți cum să dovediți că erați la muncă?

- Nu, am ieșit și intrat prin spatele blocului unde nu sunt camere și pe unde nu se poate intra sau ieși decât cu cheie. Iar la serviciu nu era nimeni, doar eu mai lucrez așa târziu. Dar vă rog să mă credeți că nu mi-aș răni niciodată soțul!

- Am înțeles.

Ți se pare suspect că nu menționează rujul și parfumul, poate te înșeli că nu sunt ale dânsei, poate evită ca să nu își arate motivul din spatele crimei sau poate nu a observat din cauza situației stresante în care se află.

O întrebi:

1. L-ați suspectat pe soțul dumneavoastră vreodată că v-ar înșela?

2. Ce ruj purtați înainte să părăsiți casa și ce parfumuri dețineți?

3. Sunteți sigură că nu ne ascundeți nimic și nu observați nimic ciudat?

//Ai ales 2.

""Răspunsul este corect.""

- Purtam ruj roșu, iar, legat de parfum, e singurul pe care îl am, ce port acum. Nu văd de ce contează și de ce vă interesează.

Indiciu cod 100 : o cadă în care sunt 4 pantofi

Apartament 1:

Ai ajuns la ușa apartamentului 1, vecinii de vis-a-vis.

Răspunde o doamnă în vârstă.

- Bună ziua, ce doriți?

- Bună ziua, am venit să vă adresez câteva întrebări despre crima din apartamentul 2. Sunt detectivul acestui caz. Aș vrea să începeți prin a vă legitima și să îmi spuneți ce făceați în acea seară și dacă aveți vreo informație, dacă poate ați auzit ceva.

-Sunt Popescu Florența, eram singură acasă. Am auzit niște tocuri pe scări. Posibil să fi fost soția sau vecina de la apartamentul 4, Mărioara, că doar ele mai poartă tocuri, dragele, tinerelele, le aud mereu cum merg prin bloc.

Realizezi că putea fi doar vecina de la apartamentul 4, cum soția lucra la acel moment.

- Am auzit după câteva ore o ceartă mare între vecinul și un bărbat. Și am auzit mai târziu ceva spărgându-se. Nu mi s-a părut ciudat, știu că la ei el gătește și mai gătește cu câte cineva și se lasă mai intens. Vă imaginați că un bărbat în bucătărie mai sparge lucruri, dar doi care se mai și ceartă? După câteva ore, vecinul a dat drumul la muzică foarte tare, face asta de obicei când sărbătorește ceva și durează toată noaptea. Mi-am pus dopurile în urechi și m-am culcat să evit gălăgia. Rar e așa fericit, deci nu am făcut plângere pentru zgomot. Poate trebuia, totuși, să fac și ar mai fi fost în viață sărmanul. Îmi cer scuze, dar nu cunosc mai multe detalii, posibil să vă poată ajuta vecina de sus de la apartamentul 7, e o fire foarte curioasă, vorbăreață și băgăcioasă. Nu se înțelegea ea prea bine cu domnul Ionescu, Dumnezeu să îl ierte. I-a făcut soțul dânsei mare dezastru în bucătărie de multe ori când acesta l-a invitat pe Ionescu să gătească împreună. Mă îndoiesc că e prea tristă în acest moment, însă sigur o să se prefacă măcar, îi plac dramatismele și atenția.

Ce întrebare mai pui?

a. Ar putea fi sunetul de ceva spart de la o sticlă de vin?

b. Știți dacă vecinul își înșela soția?

c. Puteți sa descrieți vocea persoanei cu care se certa vecinul mai in detaliu?

//Ai ales 3.

""Răspunsul este corect.""

- Da. Era o voce groasă, ca de fumător. Dacă stau să mă gândesc, singurii vecini care ar putea fi ar fi domnul de mai sus de la apartamentul 3, Petrică sau domnul Arici, îi știu doar numele

de familie, de la etajul 2, nu știu numărul apartamentului. Fumează amândoi de se umple întreg blocul cu miros de tutun.

//aici nu se primește indiciu vizual nu există cod

Apartament 7:

Ai ajuns la apartamentul 7, vă răspunde o doamnă ce poartă ruj roz și un parfum similar cu cel ce se simțea în apartamentul unde a fost crima. Pare că vă aștepta privind pe vizor. Această curiozitate vă intrigă, e oare doar curioasă sau precaută?

- Bună ziua, sunt..

- Detectivul, da, desigur, am auzit sirena de poliție și am vorbit și cu vecina, săraca de ea. Dar unde îmi sunt manierele...doriți apă, ceai, cafea? Poate doriți ceva de mâncare? Soțul meu e acum în bucătărie. El și domnul Ionescu, Dumnezeu să îl odihnească, ierte și ajute, sărmanul, ce rău îmi pare, ei mereu vorbeau de rețete noi, ce ustensile de bucătărie mai folosesc, găteau împreună. Eu nu am prea vorbit cu el, doar cu soția. Laurențiule, ia vino aici că a venit la noi detectivul și tu stai acolo în bucătărie ca tembelul, nu se cuvine. Alături nu e nevoie să mergeți, să știți, nu e nimeni acasă, uitați!

//doamna iese repede pe ușă și ciocăne, nu răspunde nimeni, ușa e închisă, nu se aude nimic.

- Sunt doar amețiți și au lăsat draperiile deschise. Au plecat în vacanță. Familia Ploieșteanu, foarte drăguți, dar unul mai aiurit decât celălalt!

- Nu vreau nimic și vă rog să nu vă mai abateți de la subiect și să răspundeți la câteva întrebări.

- Desigur.

- Ce făceați în seara crimei? Aveți vreo informație utilă legată de caz? Remarc parfumul dumneavoastră, de unde este? Bineînțeles, este necesar să vă legitimați și dumneavoastră și soțul dumneavoastră.

- Cu siguranță! Eu sunt Marinescu Letiția, soțul meu e Marinescu Laurențiu, dar eu îi mai zic și Lulu. Am fost amândoi acasă. Eu cu căștile în urechi la maxim la mine în cameră urmărind filmul meu preferat '10 lucruri nu-mi ...' asta nu e relevant, îmi cer iertare, vai, ce groaznic. Soțul meu era în sufragerie, se uita la ceva meci din altă țară difuzat la oră târzie în boxele din casă de zici că erau fotbaliștii la mine în casă. Noroc cu căștile acestea noi, că altfel... În fine, niciunul nu am auzit și nu știm nimic, din mare, mare nefericire, dar vine și soțul meu acum, LAURENȚIULE, HAI!, și vă spune ce știe el. Cât despre parfum, nu vă

vedeam cunoscător, aveți vreo doamnă specială căreia doriți să îi dăruiți parfumul? Eu lucrez în domeniu, chiar i-am dat același parfum pe care îl port acum și Mariei, vecina de la apartamentul 4.

- Bună ziua, zice Laurențiu gâfâind și puțin bâlbâit, îmi cer iertare că vin abia acum. Doriți niște brioșe? întinde o farfurie plină cu brioșe.

- Nu doresc, vreau doar să îmi spuneți ce ați făcut seara trecută și dacă știți ceva relevant cazului.

- Nu știu nimic, îmi pare rău. Mă uitam la meci, nu am auzit altceva. Acum făceam niște brioșe la cuptorul cu microunde. Știti, l-am împrumutat de la un vecin acum ceva timp și va trebui să îl dau înapoi curând, profit cât îl mai am. Omul acesta împrumută câte ceva tuturor, nu m-ar mira să fie câte ceva de al lui în casa fiecăruia dintre noi.

- Înțeleg.

Crezi că mai ai ce informație să obții de la acești locatari?

1. Da, de la soț.

2. Da, de la soție.

3. Nu. -greșit spui La revedere

//Ai ales 1.

""Răspunsul este corect.""

- Și de acest cuțit știți ceva? întrebi ridicând cuțitul aflat acum într-o țiplă de dovezi.

- Da, e al domnului care mi-a dat cuptorul, dar asta nu înseamnă nimic. Putea fi în casa oricui, la cum împrumută el lucrurile, într-o zi o să rămână cu casa goală!

- Cine e acest vecin?

- Domnul de la etajul 1, apartamentul din stânga. Îl cheamă Petrică, nu îi știu numele de familie. Locuiește singur, nu prea aude săracul, deși e destul de tânăr, nu ca mine. Vă poate spune sigur la cine era cuțitul!

- Știu de la soția dumneavoastră că dumneavoastră și domnul Ionescu discutați des. Știți cumva cu cine altcineva mai vorbea dânsul?

- Știu că se mai vedea cu domnul de la apartamentul 5, Andrei, și cel de la 6, din familia Popa. Doar fiul domnilor a mai rămas, Dumnezeu să îi ierte. Poate știe unul din ei ceva în plus față de mine, noi, sincer să vă spun, vorbeam doar de gătit.

Indiciu 130: un arici și 3_ _ , o brioșă și _3_.

Apartament 8:

Ai ajuns la apartamentul 8, nu e nimeni acasă.

Ai părăsit apartamentul, alegi următorul apartament sau să fluturi mâna pentru a răspunde la întrebarea finală.

Apartament 5:

Ai ajuns la apartamentul 5, unde răspunde un domn cu un pansament pe mână, fumând.

- Bună ziua, cu poliția sunteți. Sunt Arici Andrei. Presupun că aveți deja pregătite peste 30 de întrebări. Vă spun de pe acum că eu și domnul Ionescu nu ne înțelegeam prea bine în ultima vreme, însă nu l-aș fi omorât niciodată.

- De ce? Doar nu credeți că doar cuvântul și moralitatea presupusă a dumneavoastră sunt suficiente și nici că erați prieteni nu vă acoperă cu nimic.

- Nimic din ce ați spus dumneavoastră. Omul îmi datora bani, bani pe care acum nu o să îi mai văd, că nu mă înjosesc să îi cer văduvei lui acum, sărmana, să îi plătească datoriile. La ce îmi e el bun mort? La nimic, gaură în buzunar. Eu dau din bună credință, că știu că am și știe și Domnul. Însă îmi e dator de prea mult timp și cu prea mult. Chiar ne-am certat de asta în seara aia și el îmi tot zicea că simte că o să aibă bani curând. De asta am și aruncat cu sticla goală de vin pe care o avea pe jos. Așa m-am și tăiat. Am plecat apoi de la el furios. Se poate, domnule, să își permită să dea atâția bani pe o sticlă de vin, pe haine elegante, dar să rămână dator la mine cu atâția bani? Mai am unul care îmi datorează, Ion Popa, vecinul de vis-a-vis. Îl înțeleg și pe el, că i-au murit părinții, l-am ajutat cu ce am putut, dar e bărbat în toată firea. Măcar el mi-a zis astăzi că îmi va da banii. Îmi pare rău că mă plâng, nu era relevant. Asta este tot ce știu, trebuie să intru într-un meeting la muncă acum, sper că v-am ajutat, o zi bună, Doamne ajută.

- Înțeleg. Mulțumesc, o zi bună.

*Aici nu există întrebare, trecem direct în stare alegere apartament.

La introducerea codului 330, indiciul: mulți bani și o agendă.

Apartament 6:

Ai ajuns la apartamentul 6, îți deschide un domn de nu mai mult de 35 de ani, cu un zâmbet mare pe față.

- Bună ziua!

- Bună ziua, detectivul. Vă rog să vă legitimați și să îmi spuneți ce făceați în seara trecută și orice detaliu ce ar putea ajuta cazului.

- Mă numesc Popa Ion, eram acasă, la mine in apartament. Mă uitam la televizor. Nu știu nimic, îmi pare rău.

- Dar cum de aveți această bună dispoziție?

- Eh, și dumneata, nu poate omul să fie fericit?

- Când cineva cunoscut moare nu se obișnuiește să aveți o stare prea fericită.

- Înțeleg, presupun că am avut eu alt noroc.

Te uiți prin apartament, ce obiect inspectezi?

a. Agenda de pe masă.

b. Geaca din cuier.

c. Portofelul.

//Ai ales 1.

""Răspunsul este corect.""

Găsești în agenda de 310 pagini un bilet de loto, îți notezi numerele de pe bilet pentru a verifica dacă e câștigător, afli că este și îl întrebi:

- De ce nu ați zis că motivul bucuriei este acest bilet câștigător de loto? Pot să îmi imaginez cum asta v-a schimbat dispoziția.

- Așa este, însă nu m-am gândit că este relevant. În plus, nu vreau să mă laud sau să cobesc până nu sunt fizic în posesia banilor. Sunt și eu superstițios, trebuie să mă înțelegeți.

La introducerea codului 310:

Indiciu: mai multe serii de numere tăiate agresiv într-o agendă, lângă o foaie cu câteva serii de numere tăiate subțire și cu un număr încercuit agresiv.

Apartament 3:

Ai ajuns la apartamentul 3, deschide ușa un tânăr de maxim 25 de ani cu un zambet mare pe față și cu un trabuc aprins.

- Bună ziua, detectivul, sunt aici să vă adresez câteva întrebări. Vă rog să vă legitimați, să îmi spuneți ce făceați seara trecută și orice detaliu care vi se mai pare important.

- Bună, doar eu locuiesc aici, mă numesc Petre Mogoșoaie. Nu aud prea bine și seara îmi scot aparatul auditiv, nici acum nu îl am, nu îl prea suport, deci, deseori îl scot, prefer liniștea. Știați că frigiderul scoate sunete? Oribil! Nu, mulțumesc. Eram singur în apartament, ușa larg deschisă că mor de cald și așa fac curent. Huh, noroc că nu m-au omorât pe mine, presupun. Haha, scuzați-mă, am un simț al umorului mai negru. În fine, sper că nu o acuzați pe soție, e o adevărată doamnă, îmi e ca o mamă, lucrează femeia aia zi și noapte. Săraca, ea întreține familia și el îi mănâncă banii cu alt pierde vară de la noi din bloc, Ion Popa de la apartamentul 6, pe păcănele, loto și pariuri. V-aș minți să vă spun că decesul dumnealui mă întristează.

Ce întrebare îi mai pui:

a. - Știți ceva de acest cuțit?

b. - Ce știți despre vecina de alături?

c. Niciuna.

//Ai ales 1.

""Răspunsul este corect.""

- Da, este al meu. Dar nu îl mai am de vreo câteva luni. Chiar lui Ionescu i l-am dat, măcar când gătea făcea și el ceva util. Nu pot să cred că cineva l-a folosit pentru așa ceva, probabil voiau să îmi însceneze mie crima, știind că nu îl prea suport. Nu aș face asta niciodată, totuși, ea îl iubea mult, nu aș vrea să o rănesc vreodată.

//aici nu se primește indiciu vizual nu există cod

Apartament 4:

Ai ajuns la apartamentul 4, unde vă deschide o tânără doamnă, cochetă, ce poartă ruj roz. În hol se pot vedea multiple perechi de pantofi cu toc. Vă primește în casă.

- Bună ziua!

- Bună ziua, știu cum funcționează procedura. Eram acasă, nu știu nimic. Mă cheamă Cantemir Maria. Acum, dacă nu vă supărați, sunt foarte ocupată că e zi de curățenie.

Faptul că e așa grăbită să vă facă să plecați vă pune un semn de întrebare. Vedeți că se uită panicată în spate, de unde se poate vedea ușa de la dormitor, de la baie și de la bucătărie.

Ce cameră inspectați?

1.Dormitorul.

2. Baia.

3. Bucătăria.

//Ai ales 2.

""Răspunsul este corect.""

- Mă scuzați, aș avea și eu nevoie la baie.

- Desigur, doar lăsați-mă să verific că este totul curat!

- Stați liniștită, nu judecam.

Vezi o rochie în cadă cu o pată de vin roșu, ca cel de la scena crimei. Îți amintești de rujul roz, de parfum și, cât stai să te gândești, doamna spune:

- Da, ok, nu mă mai judecați, așa e, sunt amanta domnului Ionescu, vă rog sa nu îi spuneți soției. Jur că nu l-am omorât. Ne-am văzut, am băut niște vin, am vărsat pe mine, așa că am plecat să mă schimb și am zis că îmi iau niște încălțăminte mai confortabilă. Când am revenit am auzit o ceartă, așa că am plecat înapoi la mine în apartament. Dacă l-a omorât domnul acela? Oare puteam eu sa îl opresc? Oare mă omora și pe mine?

- Liniștiți-vă, tot ce îmi spuneți este și rămâne confidențial cât timp nu sunteți găsită vinovată.

Anexa 3: Codul din loop() care evidențiază stările

```
void loop() {
  currentTime = millis();
 if (gameState == priza){
    lightShow();
    playSpecificTrackOnce(crimeSound);
     turnOffDisplay();
```

```
   if (currentTime - startLastTime >= crimeSoundDuration + titluDuration) {
    trackPlayed = false;
    gameState = initiala;
    stareBoxa = initialaSound;
    startLastTime = currentTime;
   }
}
  if( gameState == initiala){
     turnOffDisplay();
    turnOffAllLEDs();
    for (int i = 0; i < 4; i++) {
    lcMatrix[i].clearDisplay(i);
    }
     playSpecificTrackOnce(initialaSound);
   proccesKeypadInputStart();
    startLastTime = currentTime;
  }
  if(gameState == instructiuni){
   digitalWrite(LED_PINKEY, LOW);
   playSpecificTrackOnce(instructiuniSound);
    if (currentTime - startLastTime >= instructiuniSoundDuration) {
    trackPlayed = false;
    gameState = initiala;
    stareBoxa = initialaSound;
    startLastTime = currentTime;
  }
}
if(gameState == demo){
  processKeypadInputCodApartamente();
   if( stareBoxa == demoAp){
   playSpecificTrackOnce(demoAp);
   displayNumber(scorDemo);
}
```

```
  if(stareBoxa == demoIntrebare){
    playSpecificTrackOnce(demoIntrebare);
}
  if(stareBoxa == demoRasp1){
   playSpecificTrackOnce(demoRasp1);
   displayNumber(scorDemo);
   }
  if(stareBoxa == demoRasp2){
   playSpecificTrackOnce(demoRasp2);
   displayNumber(scorDemo);
   }
   if(stareBoxa == demoRasp3){
   playSpecificTrackOnce(demoRasp3);
  displayNumber(scorDemo);
   }
   if(stareBoxa == demoCodGresit){
   playSpecificTrackOnce(demoCodGresit);
   displayNumber(scorDemo);
    if (currentTime - startLastTime >= demoCodDuration) {
   stareBoxa = demoCod2;
   trackPlayed = false;
   startLastTime = currentTime;
    }}
   if(stareBoxa == demoCod2)
     playSpecificTrackOnce(demoCod2);
   if(stareBoxa == demoCodCorect ){ playSpecificTrackOnce(demoCodCorect);
   displayNumber(scorDemo);
   digitalWrite(LED_PINI1,HIGH);
    if (currentTime - startLastTime >= demoCodCorectDur) {
   stareBoxa = demoBravo;
   trackPlayed = false;
   startLastTime = currentTime;
    }}
```

```
      if(stareBoxa == demoBravo){
        turnOffAllLEDs();
        playSpecificTrackOnce(demoBravo);
        keypadEnabledCodApartamente = false;
         if (currentTime - startLastTime >= demoCodCorectDur) {
         stareBoxa = initialaSound;
         gameState = initiala;
         trackPlayed = false;
         startLastTime = currentTime;
      }
    }
  }
if(gameState == joc){
 displayNumber(scor);
 if(inGameState == inceput){
   if(stareBoxa == challange){
     playSpecificTrackOnce(challange);
     digitalWrite(LED_PIN2V,HIGH);
     for (int i = 0; i < 4; i++) {
     lcMatrix[i].clearDisplay(i);
     }
      if (currentTime - startLastTime >= apelDuration) {
     trackPlayed = false;
    stareBoxa = politie;
    startLastTime = currentTime;
   }
}
   if(stareBoxa == politie){
    turnOffAllLEDs();
    playSpecificTrackOnce(politie);
    police();
     if (currentTime - startLastTime >= politieDuration) {
     trackPlayed = false;
```

```
    stareBoxa = narator1;
    startLastTime = currentTime;
  }
}
  if(stareBoxa == narator1){
    turnOffAllPolice();
    playSpecificTrackOnce(narator1);
    if (currentTime - startLastTime >= naratorDuration) {
    trackPlayed = false;
    inGameState = alegAp;
    startLastTime = currentTime;
   enableKeypadCodApartamente();
  }
 }
 }
  if(inGameState == alegAp){
    turnOffAllLEDs();
     enableKeypadCodApartamente();
    playSpecificTrackOnce(alegereAp);
    processKeypadInputCodApartamente();
  }
 if(inGameState == cifru){  processKeypadInputCodApartamente();}
  if(inGameState == intrebFinal){ processKeypadInputCodApartamente(); if (currentTime
- startLastTime >= finalAlesDuration){ enableKeypadCodApartamente();} }
  if(inGameState == apartament){
     playSpecificTrackOnce(currentAp);
  if (currentTime - startLastTime >= apartamentDuration) {
    trackPlayed = false;
    inGameState = intrebare;
    startLastTime = currentTime;
  }
  }
  if(inGameState == intrebare){
```

```
    if(currentAp == apartament2)
    { stareBoxa = intrebare2; currentAp = apartament2; currentIntreb = intrebare2; }
      playSpecificTrackOnce(currentIntreb);
    if (currentTime - startLastTime >= intrebDuration) {
    trackPlayed = false;
    inGameState = raspuns;
    startLastTime = currentTime;
    currentChoice = 0;
  }
 }
  if(inGameState == raspuns){
        processKeypadInputCodApartamente();
        if(currentChoice != 0)
        playSpecificTrackOnce(currentChoice);
  if (currentTime - startLastTime >= alesDuration) {
    trackPlayed = false;
    startLastTime = currentTime;
    if(stareBoxa == raspunsCorect){ inGameState = povesteRaspCorect; stareBoxa = 1;
trackPlayed = false; startLastTime = currentTime;}
    if(stareBoxa == raspunsGresit){inGameState = povesteRaspCorect;stareBoxa =
raspunsGresit; trackPlayed = false; startLastTime = currentTime;}
    startLastTime = currentTime;
  }
}
  if(inGameState == povesteRaspCorect)
  {
     if(currentAp == apartament2 && stareBoxa == 1 ){playSpecificTrackOnce(ap2Aleg2);
    if (currentTime - startLastTime >= demoRaspDuration){stareBoxa = raspunsCorect;
trackPlayed = false; startLastTime = currentTime;
    }}
      if(stareBoxa == raspunsCorect) {playSpecificTrackOnce(raspunsCorect);
   if (currentTime - startLastTime >= demoCodCorectDur)
  { inGameState= alegAp; trackPlayed = false; startLastTime = currentTime;
  }
```

```
  }
    if(stareBoxa ==
raspunsGresit){playSpecificTrackOnce(raspunsGresit);processKeypadInputCodApartamen
te();}
 }
    if(inGameState == gameOver){ playSpecificTrackOnce(gameDone); lightShow2(); if
(currentTime - startLastTime >= gameOverDuration){resetFunc(); } }
    if(inGameState == win){ saveHighScore(scor); if( isHighScore == 1)
{
digitalWrite(LED_PIN1V,HIGH);digitalWrite(LED_PIN2V,HIGH);digitalWrite(LED_PI
N3V,HIGH);digitalWrite(LED_PIN4V,HIGH);digitalWrite(LED_PIN5V,HIGH);digitalWr
ite(LED_PIN6V,HIGH);digitalWrite(LED_PIN7V,HIGH);digitalWrite(LED_PIN8V,HIG
H);
}
else
{digitalWrite(LED_PIN1R,HIGH);digitalWrite(LED_PIN2R,HIGH);digitalWrite(LED_PI
N3R,HIGH);digitalWrite(LED_PIN4R,HIGH);digitalWrite(LED_PIN5R,HIGH);digitalWr
ite(LED_PIN6R,HIGH);digitalWrite(LED_PIN7R,HIGH);digitalWrite(LED_PIN8R,HIG
H);}
displayNumber(scor);if (currentTime - startLastTime >= titluDuration)
{playSpecificTrackOnce(gameWon); lightShow3();}if (currentTime - startLastTime >=
gameWonDuration + titluDuration){resetFunc();}} } }
```

Bibliografie

[i] Contributors to Wikimedia projects Wikipedia-Cluedo Ultima accesare 28 iulie 2023 URL: https://en.wikipedia.org/wiki/Cluedo

[ii] NEXT Studios Steam –Unheard-Voices of Crime joc publicat pe 29 Martie 2019 URL: https://store.steampowered.com/app/942970/Unheard__Voices_of_Crime/

[iii] FINNBAR TERALABS Ultima oară accesat pe 12 Ianuarie 2013 URL: https://teralabs.wordpress.com/2013/01/12/arduinoart-voice-your-opinion/

[iv] Getting Started with Arduino MEGA2560 | Arduino Ultima accesare 5 Februarie 2018, URL: https://www.arduino.cc/en/Guide/ArduinoMega2560

[v] “Arduino LED – Complete Tutorial | The Robotics Back-End" , Ultima accesare 30 Decembrie 2021, URL:https://roboticsbackend.com/arduino-led-complete-tutorial/

[vi] Arduino- Keypad Ultima accesare 2023 URL:
https://arduinogetstarted.com/tutorials/arduino-keypad#google_vignette
[vii] Benne de Bakker, ” MAX7219 LED dot matrix display Arduino tutorial”
Ultima accesare 1 August 2023 URL:https://www.makerguides.com/max7219-led-dot-matrix-display-arduino-tutorial/

[viii] mdrabe, ”How to control 4 digit 7 segment display with MAX7219”, Ultima accesare 9 Martie 2021,
URL: https://www.hackster.io/mdraber/how-to-control-4-digit-7-segment-display-with-max7219-a78779
[ix] Last Minute Engineers, ”Interfacing MAX7219 LED Dot Matrix Display with Arduino”, Ultima accesare 10 Octombrie 2022, URL: https://lastminuteengineers.com/max7219-dot-matrix-arduino-tutorial/
[x] “Datasheet DFPlayer mini” URL:
https://usermanual.wiki/Datasheet/DFR0299DFPlayerMiniManual.1373379751
[xi] circuitgeeks, ”How to Control WS2812B Addressable RGB LEDs using Arduino”,
Ultima accesare: 2 Martie 2022, URL: https://www.circuitgeeks.com/ws2812b-addressable-rgb-led-strip-with-arduino/#:~:text=We%20connect%20the%20Arduino's%205v,connect%20an%20external%20power%20supply
[xii] adnanaqeel, ”Introduction to Arduino IDE”,
Ultima accesare 3 Octombrie 2019, URL:
https://www.theengineeringprojects.com/2018/10/introduction-to-arduino-ide.html
[xiii] adnanaqeel, ”Introduction to Arduino IDE” Ultima accesare 3 Octombrie 2019,
URL: https://www.theengineeringprojects.com/2018/10/introduction-to-arduino-ide.html

[xiv] Eberhard Fahle- "LedControl” ,Ultima accesare: 20 Ianuarie 2015,
URL: http://wayoda.github.io/LedControl/
[xv] Mark Stanley, Alexander Brevig “Keypad Library for Arduino”,
URL: https://playground.arduino.cc/Code/Keypad/#Description
[xvi] ”PROGMEM” ,Ultima accesare 6 Iulie 2022,
URL:https://www.arduino.cc/reference/en/language/variables/utilities/progmem/
[xvii] Power_Broker Ultima accesare: 26 August 2021 URL:
https://powerbroker2.github.io/DFPlayerMini_Fast/html/index.html
[xviii] Arduino Ultima accesare 29 August 2023 URL:https://docs.arduino.cc/learn/built-in-libraries/`EEPROM`
[xix] Daniel Garcia & Mark Kriegsman Ultima accesare 9 Iunie 2023 URL:
https://github.com/FastLED/FastLED
[xx] Scott Campbell Ultima accesare 14 Noiembrie 2021 URL:
https://www.circuitbasics.com/basics-uart-communication/
[xxi] Scott Campbell Ultima accesare 14 Noiembrie 2021 URL:
https://www.circuitbasics.com/basics-of-the-spi-communication-protocol/
[xxii] Devang Jagdale, "Finite State Machine in Game Development International Journal of Advanced Research in Science Communication and Technology“, Octombrie 2021,

URL:https://www.researchgate.net/publication/355518086_Finite_State_Machine_in_Game_Development