



Proiect la disciplina
“Proiectare cu microprocesoare”

PIXEL-LOGIC



Blaj Sergiu-Emanuel, 30235

2021 – 2022



FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

Cuprins

1	Contextul problemei	3
2	Logica implementării.....	4
3	Diagrama de circuit	9
4	Montajul pe Arduino UNO.....	10
5	Bibliografie.....	12


FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

1 Contextul problemei

Nonogramele, cunoscute și sub numele de Hanjie, Paint by Numbers, Picross, Griddlers, Pic-a-Pix și diverse alte nume, sunt puzzle-uri logice cu imagini în care celulele dintr-o grilă trebuie să fie colorate sau lăsate goale în funcție de numerele de pe partea laterală a grilă pentru a dezvălui o imagine ascunsă. În acest tip de puzzle, numerele sunt o formă de tomografie discretă care măsoară câte linii neîntrerupte de pătrate completate există într-un anumit rând sau coloană. De exemplu, un indiciu de „4 8 3” ar însemna că există seturi de patru, opt și trei pătrate umplute, în această ordine, cu cel puțin un pătrat gol între seturi succesive.

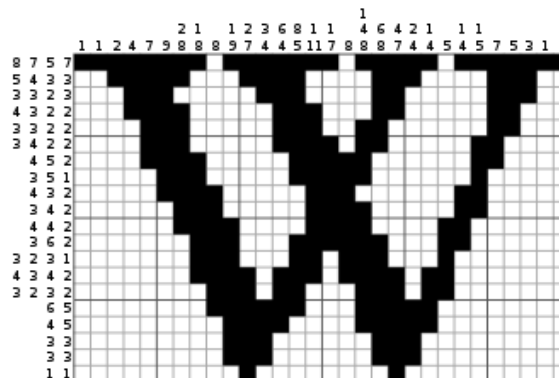


Figura 1.1

Aceste puzzle-uri sunt adesea alb-negru - descriind o imagine binară - dar pot fi și colorate. Dacă sunt colorate, indiciile numerice sunt de asemenea colorate pentru a indica culoarea pătratelor. Două numere colorate diferit pot avea sau nu un spațiu între ele. De exemplu, un patru negru urmat de un doi roșu ar putea însemna patru casete negre, câteva spații goale și două casete roșii, sau ar putea însemna pur și simplu patru casete negre urmate imediat de două casete roșii. Nonogramele nu au limite teoretice în ceea ce privește dimensiunea și nu sunt limitate la aspectul pătrat.

Nonogramele au fost numite după Non Ishida, unul dintre cei doi inventatori ai puzzle-ului. [1]

Scopul acestui proiect este de a pune logica și atenția utilizatorului la treabă. Probabil că, într-o variantă ulterioară de implementare, mini-jocul ar conține și un cronometru, pentru a motiva utilizatorul să rezolve puzzle-ul cât mai repede.



2 Logica implementării

Pentru implementarea acestui puzzle, am ales placa de dezvoltare Arduino UNO. Arduino Uno este o placă de dezvoltare bazată pe microcontrollerul ATmega328. Are 14 pini de intrare/ieșire (dintre care 6 pot fi folosiți ca ieșiri PWM), 6 intrări analog, un oscilator de 16MHz, o conexiune USB, mufă de alimentare și un buton de reset. [2]

Afișarea statusului curent al jocului se face pe o matrice LED 8x8. Ea are ledurile colorate în roșu, cărora li se poate seta o intensitate și se atașează mai ușor la Arduino decât un display led RGB.

La deplasarea pe matricea led am folosit un joystick. Joystick-ul analogic este similar cu două potențiometre conectate între ele, unul pentru mișcarea verticală (axa Y) și altul pentru mișcarea orizontală (axa X). Poate fi foarte util pentru jocuri retro, controlul roboților sau a mașinilor RC.

În timpul jocului, cu ajutorul a două buzzere, se aud sunete ambientale. Buzzerul este o “sonerie”, pe care o putem seta să producă anumite sunete, la anumite frecvențe.

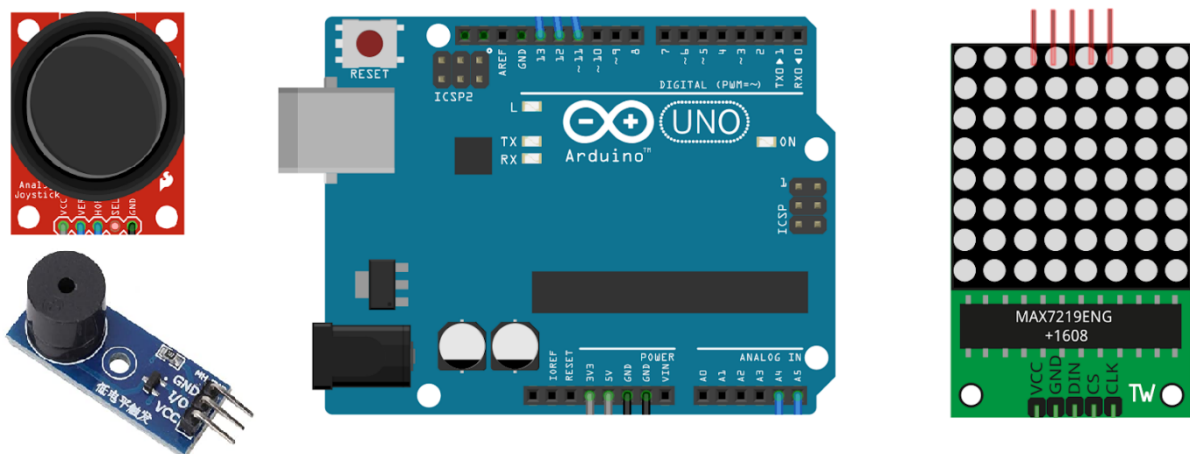


Figura 2.1

O parte din componentele utilizate se pot vedea în poza de mai sus. Au mai fost folosite butoane și fire, însă am considerat că nu este necesar reprezentarea lor. Butoanele au fost configurate să folosească rezistența de tip PULL_UP a plăcuței Arduino, pentru a evita scurt-circuitarea lor.


FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

Matricea led are dimensiunea de 8x8, așadar aceasta va fi dimensiunea maximă a unui desen. Într-o matrice de aceeași dimensiune, care are doar valori de 1 sau 0 este reprezentat desenul curent: 1 înseamnă căsuță colorată iar 0, căsuță necolorată.

Jucătorul se deplasează folosind joystick-ul. Funcția de mai jos citește valoarea redată de joystick: pentru fiecare axă, un număr de la 0 – o extremitate până la 1023 – cealaltă extremitate. Acest număr trebuie “tradus” în coordonate 0 – 8, cu ajutorul funcției *map()*. Notă: joystick-ul nu e de cea mai înaltă calitate, așadar pentru axa y observăm valori citite până la 950. Poziția curentă va fi mereu marcată pe hartă cu o bulinuță roșie. Maparea este transformată în 7 – 0 din cauza poziției joystick-ului față de matrice.

```
void readJoystick() {
    int xValue = analogRead(joyX);
    int yValue = analogRead(joyY);
    xMap = map(xValue, 0, 1023, 7, -1);
    yMap = map(yValue, 0, 950, 7, 0);
    lcd.clearDisplay();
    lcd.setLed(0, xMap, yMap, true);
}
```

Arduino UNO suportă întreruperi pe pinii digitali 2 și 3. Un buton dintre acestea două are rolul de a marca o căsuță, iar celălalt o va decolora. Joystick-ul are buton, însă am preferat să nu fac operația de colorare/ decolorare a celulei cu acesta din cauza calității acestuia (uneori, la apăsarea butonului, poziția se poate schimba).

```
attachInterrupt(digitalPinToInterrupt(markBtn), markCell,
RISING);

void markCell() {
    crtDrawing[xMap][yMap] = 1;
}
```

Au fost predefinite 5 desene. Acestea sunt stocate într-o matrice de dimensiunea 8 x 8, care conține valori de 1 sau 0, 1 însemnând o celulă colorată. Aceasta poate fi extinsă ulterior cu oricâte desene se dorește.


FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

Jucătorul trebuie să știe pozițiile în care să coloreze căsuțele, așadar, pe SerialMonitor este afișată starea jocului. Pentru fiecare linie și coloană, se calculează numărul de grupuri de celule. Dacă acesta a fost colorat în totalitate, se va afișa un 'V' în dreptul liniei/ coloanei respective; în caz contrar, va apărea un 'X'.

Mai jos, avem partea de cod care preia starea unei linii. Se verifică dacă rândul este completat corect cu desenul prestabilit, pentru a se ști dacă se va afișa 'X' sau 'V'. Apoi se numără lungimea secvențelor de 1, în variabila *crtSequenceLength*. Deoarece pot exista maxim 4 grupuri de 1 (1 1 1 1, ce va corespunde în desen cu 1 0 1 0 1 0 1 0), matricea *rowStatus* are 5 coloane, pe ultima fiind memorată starea liniei.

```
void getRowStatus() {
    for (int row = 0; row < DIMENSION; row++) {
        int rowStatusIndex = 3;
        bool rowFinished = true;
        int crtSequenceLength = 0;
        for (int col = DIMENSION - 1; col >= 0; col--) {
            if (crtDrawing[row][col] != allDrawings[row][col +
drawingNo * DIMENSION]) {
                rowFinished = false;
            }
            if (allDrawings[row][col + drawingNo * DIMENSION]) {
                crtSequenceLength++;
            } else {
                if (crtSequenceLength) {
                    rowStatus[row][rowStatusIndex--] =
crtSequenceLength;
                }
                crtSequenceLength = 0;
            }
        }
        if (crtSequenceLength) {
            rowStatus[row][rowStatusIndex--] = crtSequenceLength;
        }
        rowStatus[row][4] = rowFinished ? 1 : 0;
    }
}
```

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

Afișarea stării unei linii începe mai întâi cu afișarea grupulețelor de 1, după care se va afișa desenul curent.

```

void printRowStatus() {
    for (int i = 0; i < DIMENSION; i++) {
        String line = "";
        for (int j = 0; j < 4; j++) {
            if (rowStatus[i][j]) {
                line += rowStatus[i][j];
            } else {
                line += " ";
            }
            line += " ";
        }
        line += rowStatus[i][4] == 1 ? "V" : "X";
        line += " ";
        for (int j = 0; j < DIMENSION; j++) {
            if (crtDrawing[i][j]) {
                line += "1";
            } else {
                line += " ";
            }
            line += " ";
        }
        Serial.println(line);
    }
}

```

Un buton a fost folosit pentru a curăța desenul curent. Această funcție este utilă în momentul în care jucătorul s-a încurcat și vrea să o ia de la capăt. În

[illegible]

Figura 2.2

**FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE**
DEPARTAMENTUL CALCULATOARE

momentul în care jucătorul completează desenul, o bifă va apărea pe matricea led, marcând sfârșitul jocului.

Mai trebuie menționat faptul că, la finalul jocului, utilizatorul poate asculta o secvență din “Preludiu în C Major” de J. S. Bach. Din cauza faptului că buzzerele sunt active (funcționează pe logică negativă), acestora le trebuie scrisă valoarea *LOW* în timpul pauzei, pentru a nu fi deranjante.

```
for (int thisNote = 0; thisNote < songLength; thisNote++) {  
    int noteDuration = 1000 / NOTE_DURATION;  
    tone(musicPin, melody[thisNote], noteDuration);  
    digitalWrite(musicPin, HIGH);  
    int pauseBetweenNotes = noteDuration * 1.30;  
    delay(pauseBetweenNotes);  
    noTone(musicPin);  
}
```



3 Diagrama de circuit

Pieșele asamblate sunt prezentate în figura de mai jos. Una din uneltele folosite în realizarea acestui desen a fost Tinkercad.

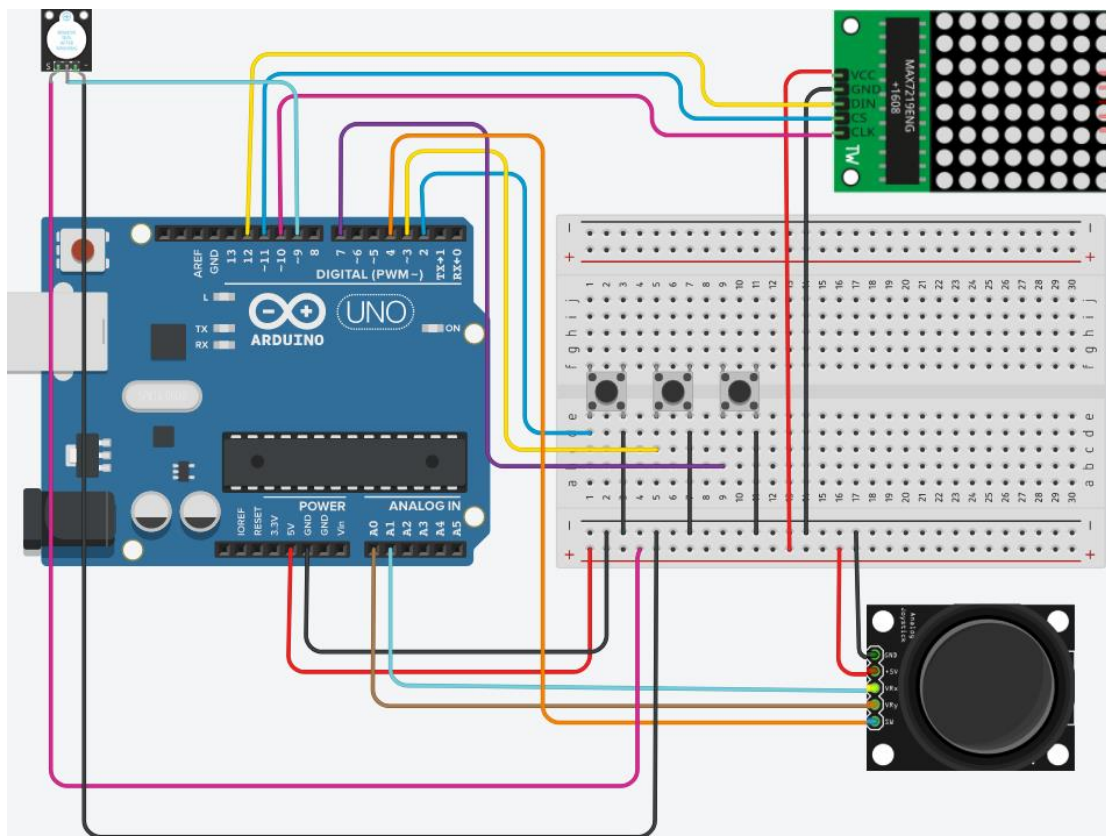


Figura 3.1

**FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE**
DEPARTAMENTUL CALCULATOARE

4 Montajul pe Arduino UNO

Figura de mai jos conține montajul pe Arduino UNO. Pe matricea led 8x8 se observă o fază a jocului, în care se încearcă ghicirea unui desen.

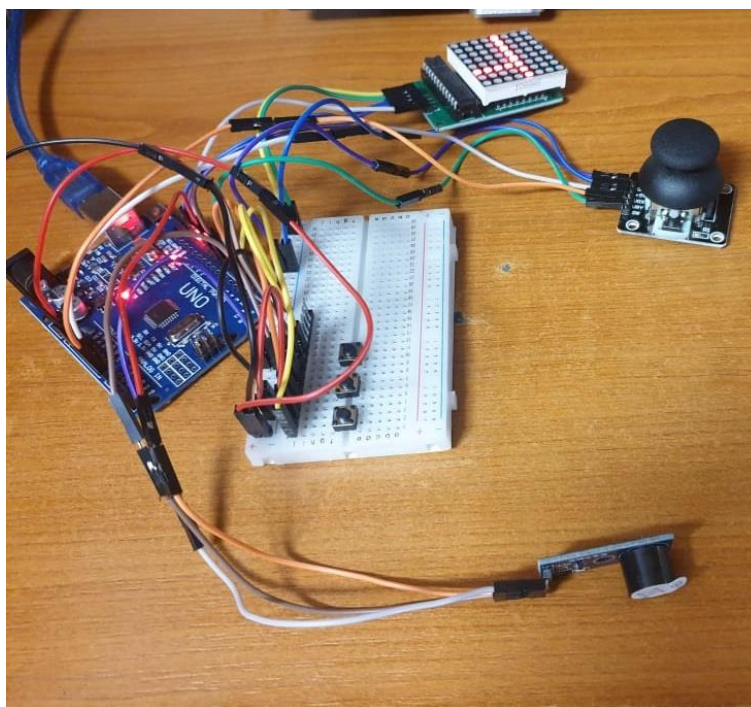


Figura 4.1

Următoarea figură mai are nevoie doar de un pixel colorat, iar desenul va fi complet.

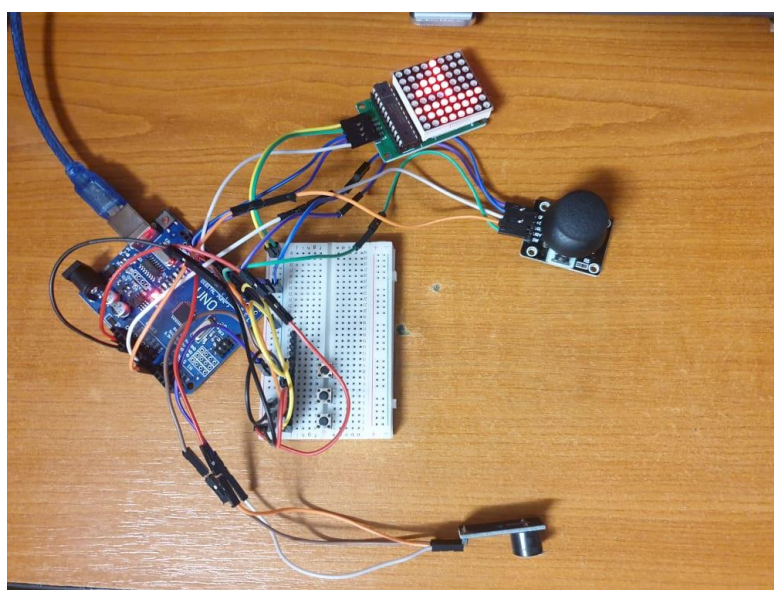


Figura 4.2

**FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE**
DEPARTAMENTUL CALCULATOARE

Jocul a fost terminat, iar o secvență din “Preludiu în C Major de J. S. Bach” începe să cânte. Pe matricea led apare o bifă, semn că jucătorul a completat corect desenul. [3]

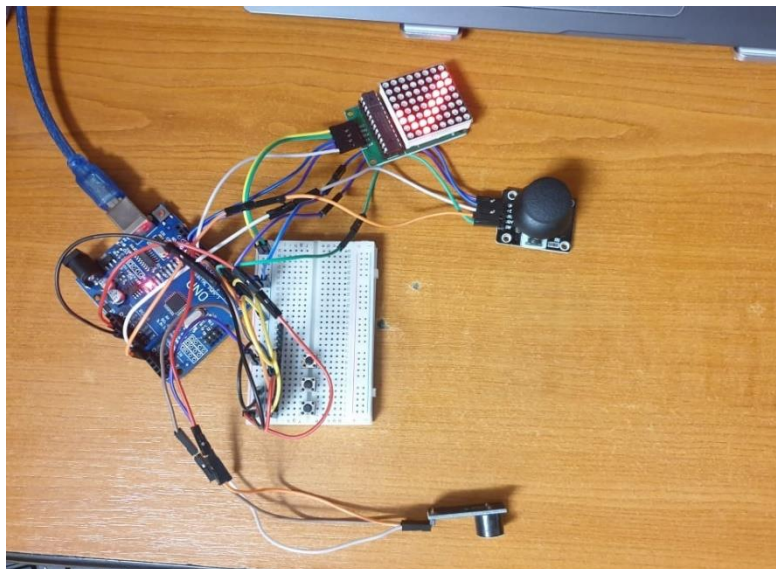


Figura 4.3



FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

5 Bibliografie

- [1] <https://en.wikipedia.org/wiki/Nonogram>
- [2] https://en.wikipedia.org/wiki/Arduino_Uno
- [3] https://en.wikipedia.org/wiki/Prelude_and_Fugue_in_C_major,_BWV_846