Proiect 2

Game of Life

Angan Lavinia

Gradea David

Grupa:433A

Profesor coordonator: Ulmamei Alexandru

Cuprins

Introducere

"**Game of Life**" se refera la implementarea automatonului celular al lui John Conway "Conway's Game of Life", pe un dispoitiv de tip microcontroller.Este un joc de simulare pe o grila bidimensionala de celule, fiecare celula poate sa fie moarta sau vie, acest joc evolueaza in functie de starea fiecarei celule determinate de starea celulelor vecine comform urmatoarelor reguli:

- ❖ Supravietuire: O celula vie cu 2 sau 3 vecini ramane vie
- ❖ Moarte:O celula vie cu mai putin de 2 sau mai mult de 3 vecini moare
- ❖ Nastere:O celula moarta cu exact 3 vecini devine vie

Aceste reguli simple conduc la comportamente complexe și interesante, cum ar fi formarea structurilor stabile, inclusiv nave spațiale, planere și oscilatori, precum și evoluția și extincția modelelor în timp. Jucătorii nu interacționează direct cu jocul, ci creează inițial configurații de celule și apoi observă evoluția acestora conform regulilor.

Scopul proiectului: Implementarea unui sistem de afișaj cu LED-uri care să poată afișa starea jocului "Game of Life".

Descrierea Proiectului

Proiectul constă în crearea unui sistem de afișare a jocului "Game of Life" pe un afișaj cu LED-uri. Microcontrollerul ATmega164A este folosit pentru controlul afișajului, iar modul MAX7219 (cu 4 matrice LED) este utilizat pentru gestionarea LED-urilor. Jocul se desfășoară pe o grilă bidimensională de LED-uri, iar evoluția acestuia este determinată de reguli simple implementate in cod in main.c:

```
// Aplicam regulile jocului Game of Life
if (grid[module][i][j] == 1) {
    if (neighbors < 2 || neighbors > 3)
        newGrid[module][i][j] = 0; // Celula moare
    else
        newGrid[module][i][j] = 1; // Celula supravietuieste
} else {
    if (neighbors == 3)
        newGrid[module][i][j] = 1; // Celula invie
    else
        newGrid[module][i][j] = 0; // Celula ramane moarta
}
```

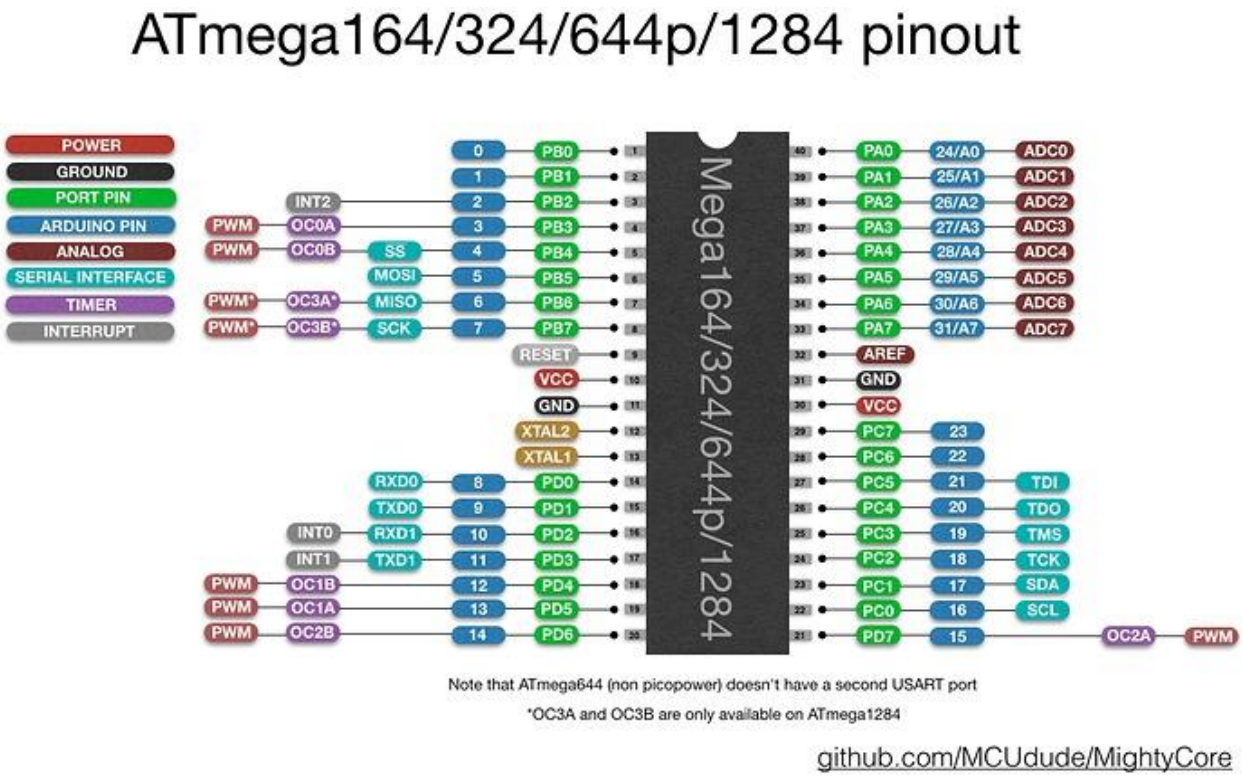
Detalii Tehnice

Partea Hardware

Microcontroller ATmega164A

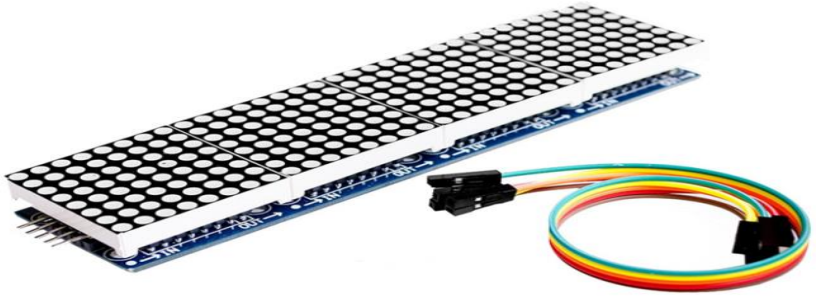
Microcontrollerul ATmega164A este ales pentru capacitatea sa de a controla un afișaj cu LED-uri și pentru puterea sa de calcul suficientă pentru implementarea regulilor jocului "Game of Life".

Acesta este un microcontroller AVR care functioneaza la o frecventa de 20 Mhz are 32 de pini(intrari-iesiri) de tip SMD, avand o memorie de 512B EEPROM,1kB SRAM, 16kB FLASH



Modul MAX7219 cu 4 matrice LED

Modulul MAX7219 are 4 matrice de LED-uri dispuse 8x8 este utilizat pentru gestionarea afișajului cu LED-uri. Acesta permite controlul independent al fiecărui LED și oferă o interfață simplificată pentru comunicarea cu microcontrollerul. Acesta lucreaza la o tensiune de 5V .



Pin Header

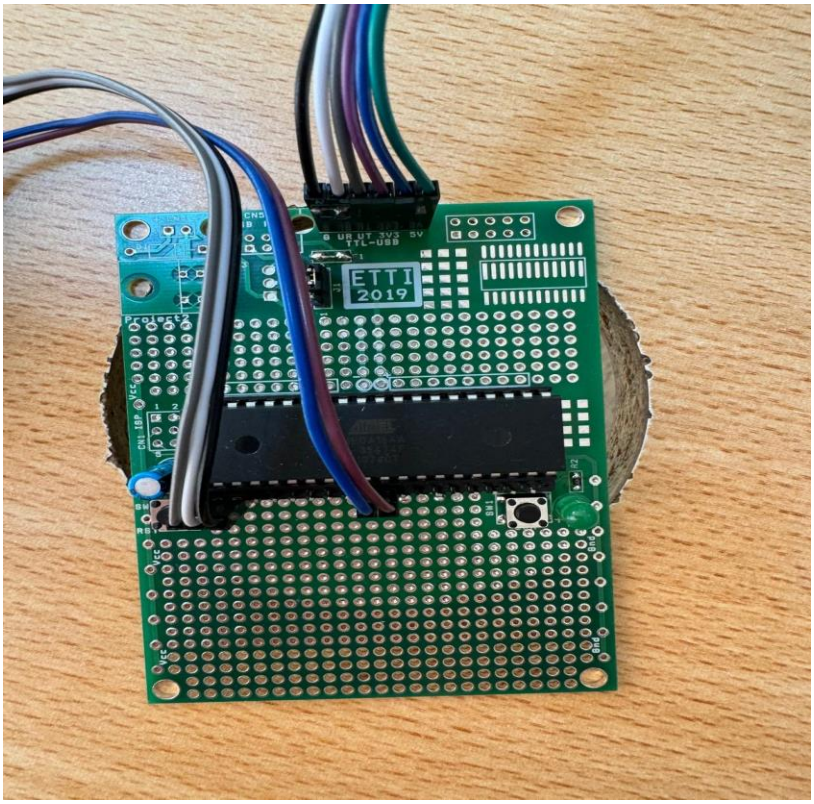
Convertor USB to TTL

Fire mama tata

Butoane

LED

Condensator Electrolit



Partea Software

Codul sursa Include bibliotecile necesare pentru funcționarea microcontrolerului și funcțiile standard.

In continuare se definesc dimensiunile matricei de LED-uri. MODULES reprezintă numărul de module MAX7219 folosite (4 în acest caz), fiecare având 8 rânduri și 8 coloane.

Funcția SPI_Write_Byte

Această funcție trimite un byte (8 biți) prin protocolul SPI către MAX7219. Funcția parcurge fiecare bit al byte-ului și setează pinii de date (DIN) și ceas (CLK) corespunzător.

```
void SPI_Write_Byte(unsigned short num)
{
    unsigned short t, Mask, Flag;
    CLK_Pin = 0;
    Mask = 128;
    for (t=0; t<8; t++) {
        Flag = num & Mask;
        if(Flag == 0) {
            DIN_Pin = 0;
        } else {
            DIN_Pin = 1;
            }
        CLK_Pin = 1;
        CLK_Pin = 0;
        Mask = Mask >> 1;
    }
}
```

Funcția MAX7219_INIT configurează fiecare modul MAX7219 pentru afișare. Sunt setate diverse registre pentru modul de decodare, intensitatea luminii, limita de scanare și testul de afișare.

```
void MAX7219_INIT() {
unsigned char module;
    for ( module = 0; module < MODULES; module++) {
        CS_Pin = 0;
        SPI_Write_Byte(0x09);
        SPI_Write_Byte(0x00);
        CS_Pin = 1;

        CS_Pin = 0;
        SPI_Write_Byte(0x0A);
        SPI_Write_Byte(0x05);
        CS_Pin = 1;

        CS_Pin = 0;
        SPI_Write_Byte(0x0B);
        SPI_Write_Byte(0x07);
        CS_Pin = 1;

        CS_Pin = 0;
        SPI_Write_Byte(0x0C);
        SPI_Write_Byte(0x01);
        CS_Pin = 1;

        CS_Pin = 0;
        SPI_Write_Byte(0x0F);
        SPI_Write_Byte(0x00);
        CS_Pin = 1;
    }
}
```

Această funcție trimite date specifice unui modul MAX7219, utilizând funcția SPI_Write_Byte pentru a specifica coloana și valoarea de afișat.

```
void Write_Byte(unsigned char module, unsigned short myColumn, unsigned short myValue)
{   unsigned char m;
    CS_Pin = 0; // selecteaza max7219.
    for ( m = 0; m < MODULES; m++) {
        if (m == module) {
            SPI_Write_Byte(myColumn); // trimite valoarea myColumn catre max7219
            SPI_Write_Byte(myValue); // trimite valoarea myValue catre max7219
        } else {
            SPI_Write_Byte(0x00);
            SPI_Write_Byte(0x00);
        }
    }
    CS_Pin = 1; // deselecteaza max7219.
}
```

Funcția Clear_Matrix șterge matricea de LED-uri setând toate valorile la 0.

```
void Clear_Matrix(void)
{    unsigned char module;
     unsigned short x;
    for ( module = 0; module < MODULES; module++) {
        for ( x = 1; x < 9; x++) {
            Write_Byte(module, x, 0x00);
        }
    }
}
```

Funcție Set_LED setează starea unui LED individual în matricea de LED-uri.

```
void Set_LED(unsigned char module, unsigned char col, unsigned char row, unsigned char state) {
    unsigned char ledValue = 1 << (31 - col); // Generam valoarea corecta pentru coloana dorita
    if (state) {
        Write_Byte(module, row + 1, ledValue); // 'row + 1' pentru ca rândurile în functie de MAX7219 încep de la 1
    } else {
        Write_Byte(module, row + 1, 0x00); // Stinge LED-ul în pozitia respectiva
    }
}
```

Funcția updateGrid actualizează grila de joc conform regulilor jocului "Game of Life", specificate mai sus in introducere.

Funcția updateDisplayFromGrid actualizează afișajul de pe matricea de LED-uri în funcție de starea actuală a grilei de joc.

```
void updateDisplayFromGrid() {
    unsigned char row, col, module;
    for (module = 0; module < MODULES; module++) {
        for (row = 0; row < ROWS; row++) {
            unsigned char rowData = 0;
            for (col = 0; col < COLS; col++) {
                if (grid[module][row][col])
                    rowData |= (1 << col);
            }
            Write_Byte(module, row + 1, rowData);
        }
    }
}
```

Funcția initializeGridRandomly inițializează grila de joc cu valori aleatorii (0 sau 1), stabilind celulele ca fiind fie moarte, fie vii la începutul jocului.

```
void initializeGridRandomly() {
    unsigned char i, j, module;
    for (module = 0; module < MODULES; module++) {
        for (i = 0; i < ROWS; i++) {
            for (j = 0; j < COLS; j++) {
                grid[module][i][j] = rand() % 2; // Initializare random cu 0 sau 1
            }
        }
    }
}
```

Funcția Principală

```
void main() {
    DDRB = 0xFF;

    MAX7219_INIT();
    Clear_Matrix();

    // Initializeaza grila cu valori random
    initializeGridRandomly();

    while (1) {
        // Apeleaza functia pentru a modifica grila
        updateGrid();
        // Actualizeaza afisajul MAX7219 cu noua grila
        updateDisplayFromGrid();
        delay_ms(2000); // Timp de intarziere intre generari
    }
}
```

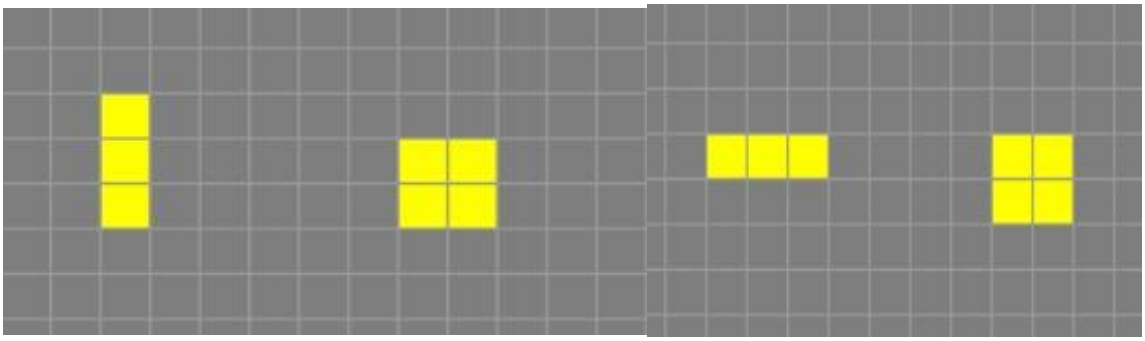
Exemplu de utilizare

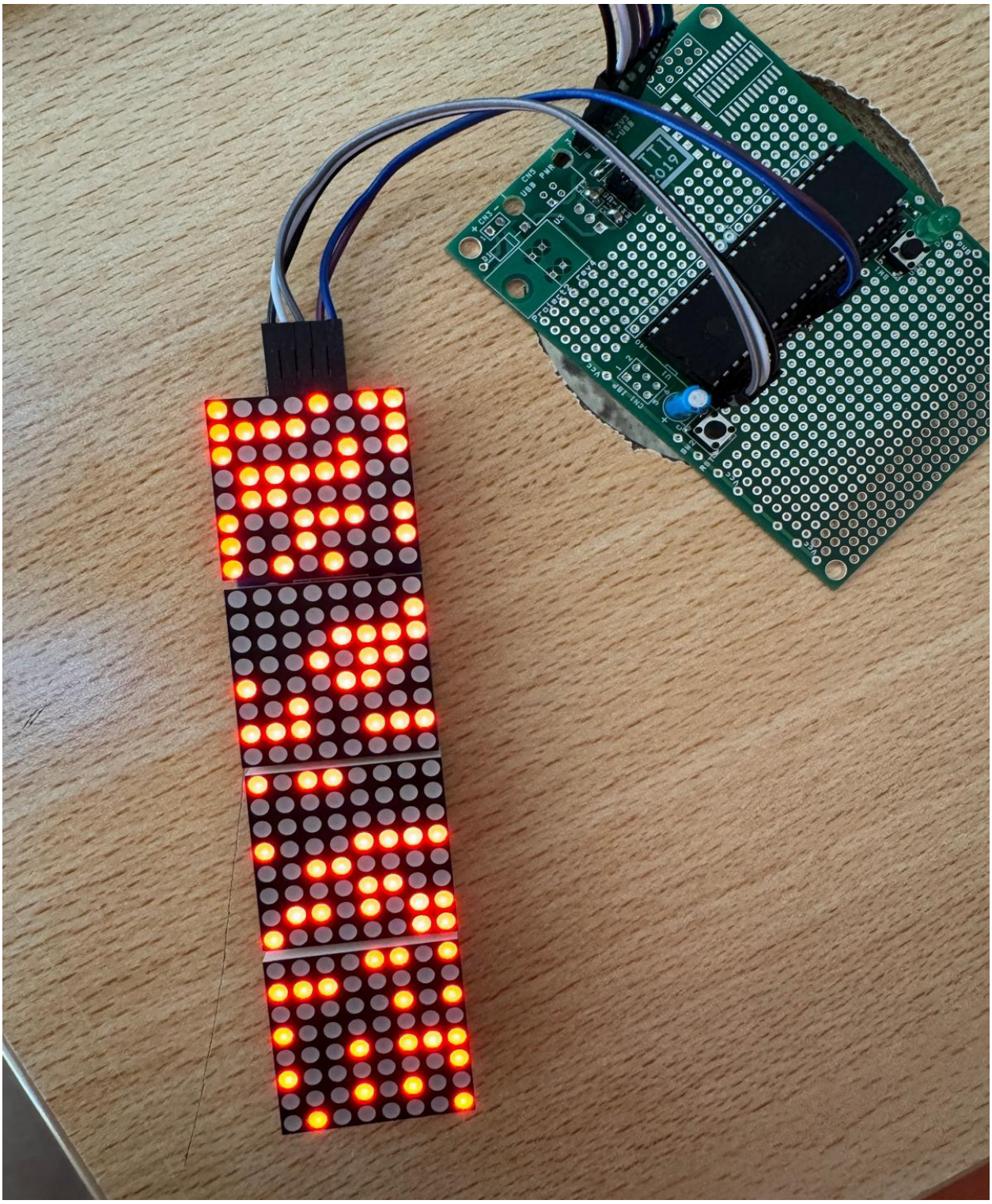
Se poate observa evoluția jocului "Game of Life" pe afișajul cu LED-uri și poate observa cum modelele se schimbă și interacționează în timp real.

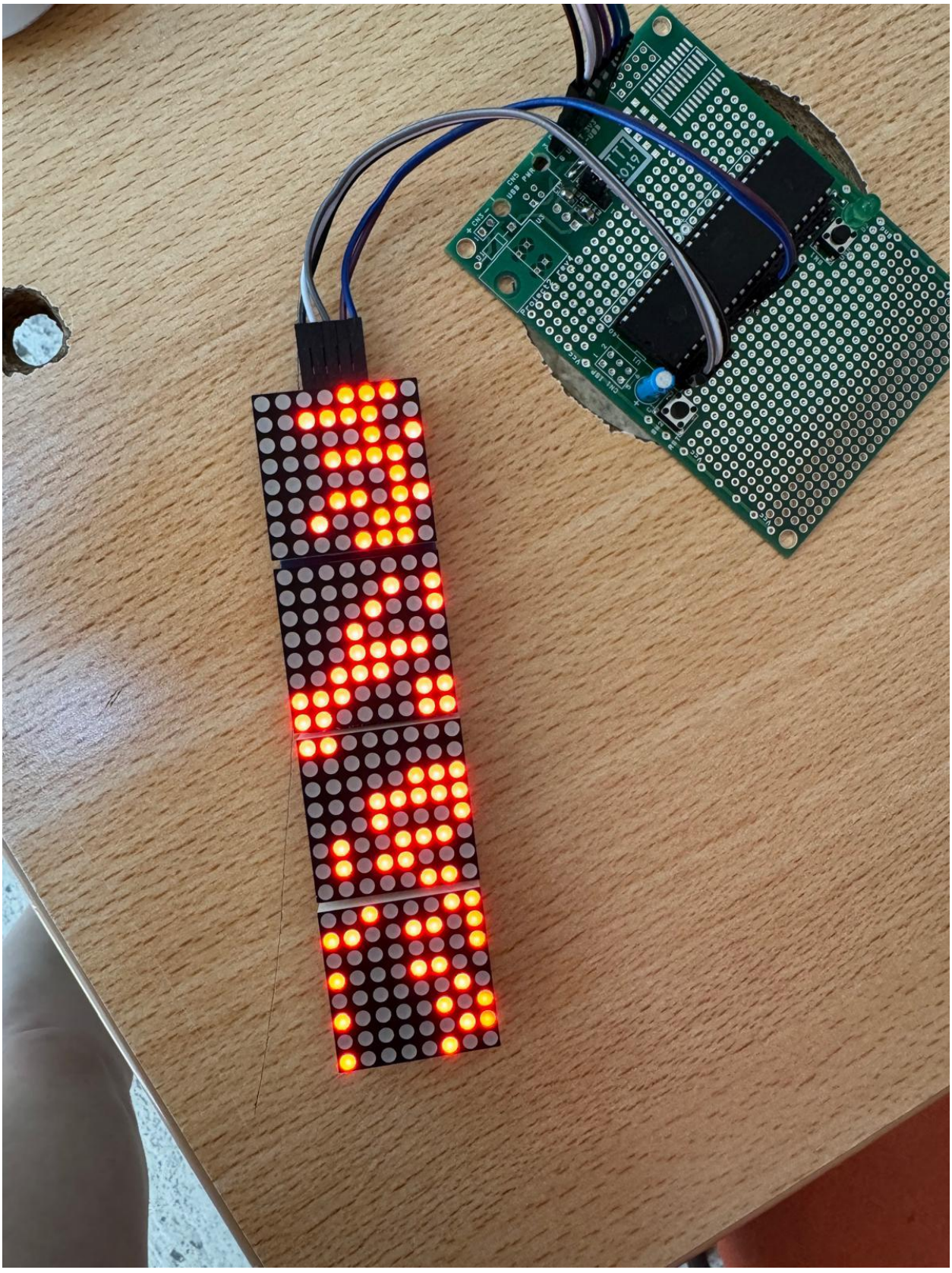
Se incepe cu initializarea aleatoare a modulelor cu LED-uri, iar apoi se aplica regulile, observand diferenta intre afisaje.

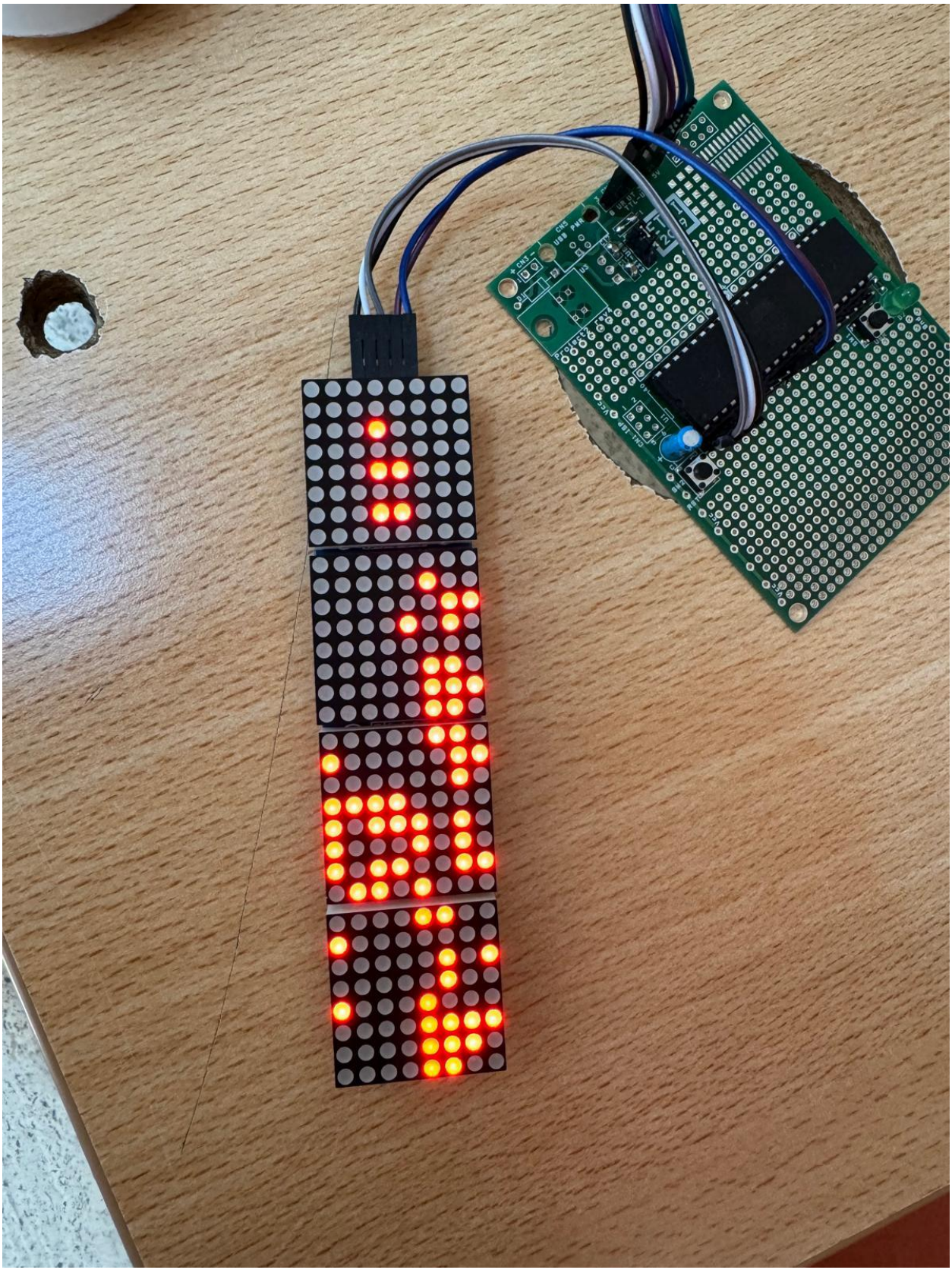
Led-urile dupa module respecta regulile, deoarece se observa cum acestea se sting complet de pe fiecare modul pana se ajunge la un model al LED-urilor care nu se mai poate modifica.

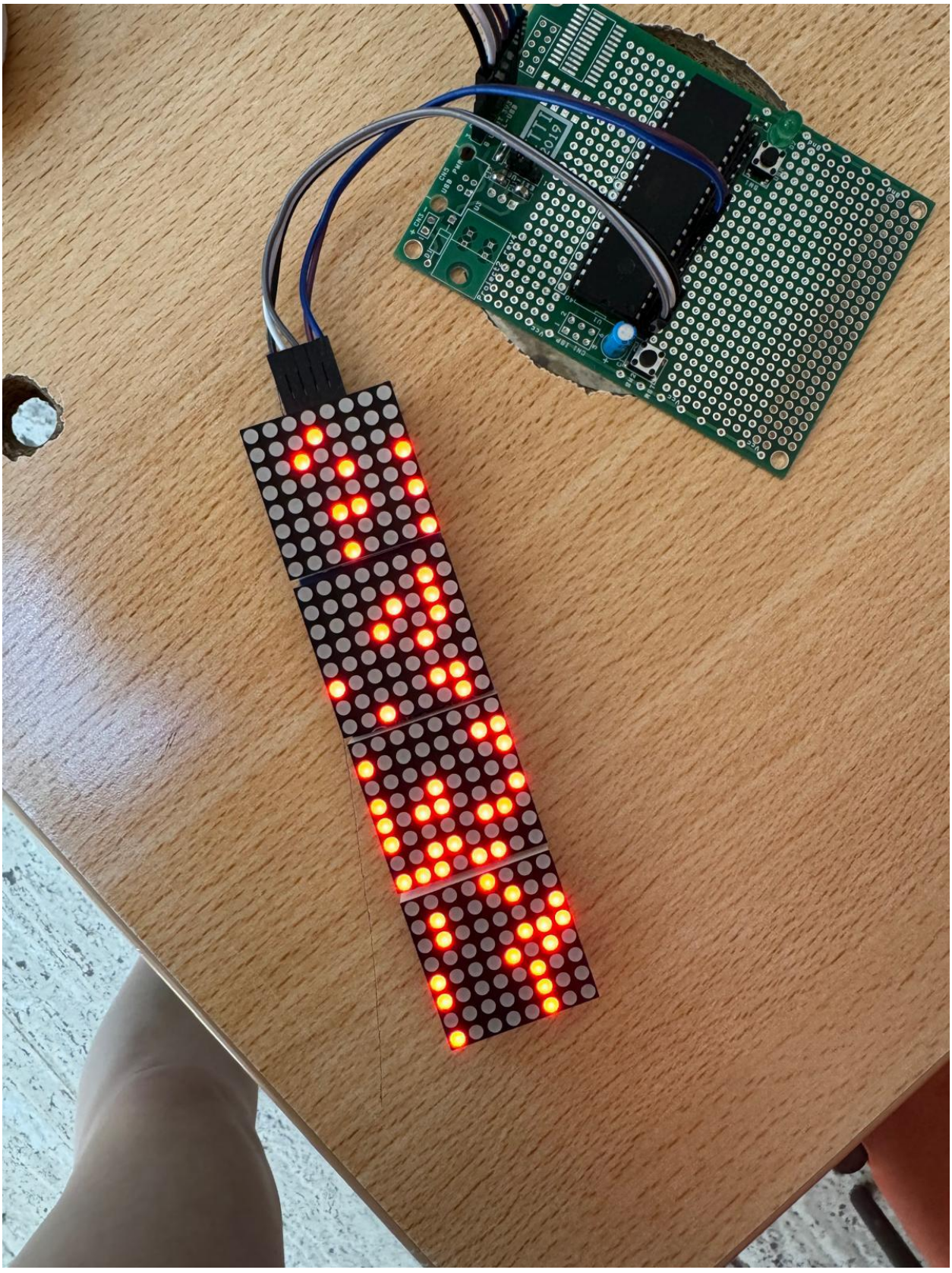
Am realizat verificarea cu simulatorul pentru acest joc:

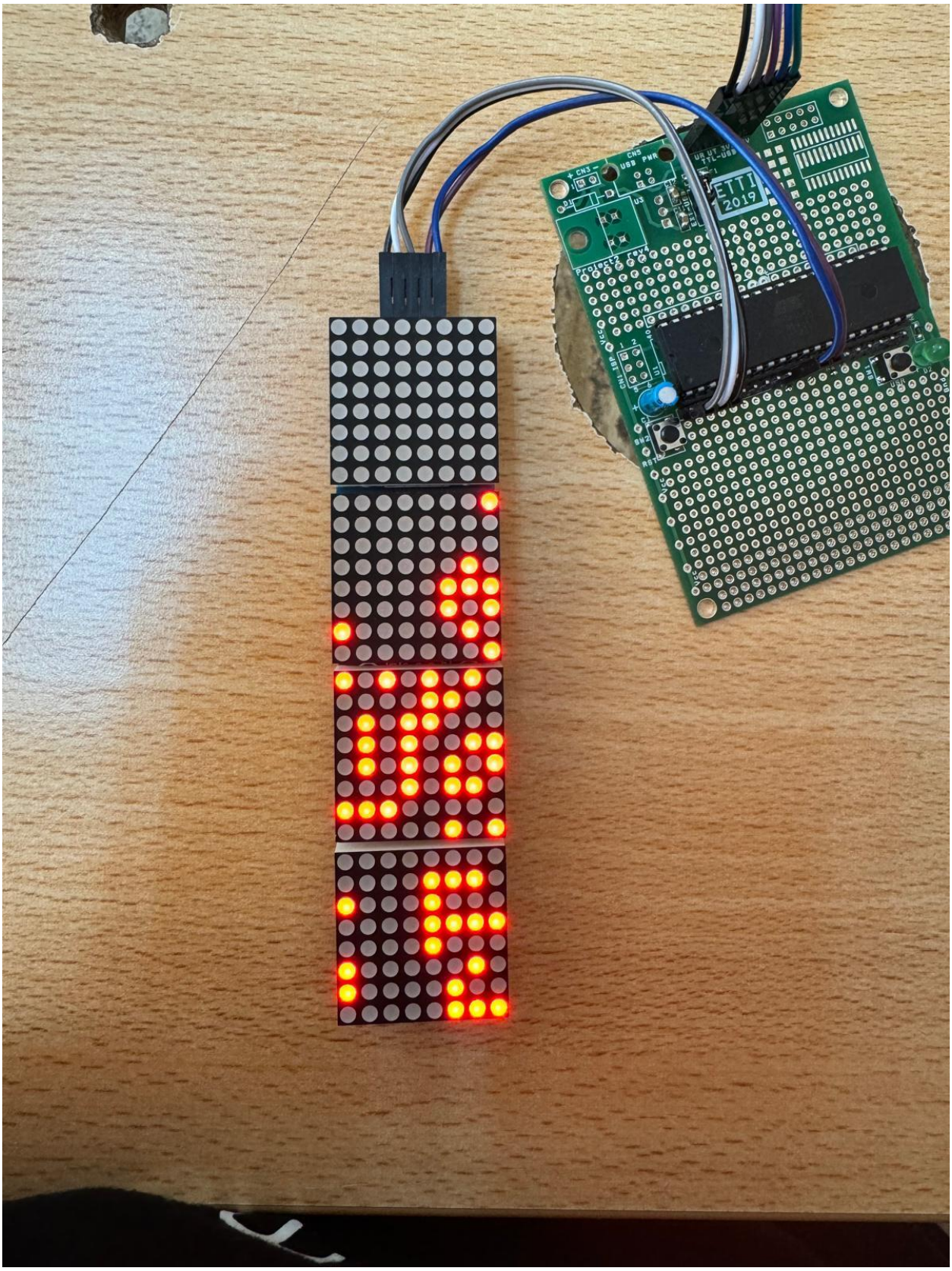


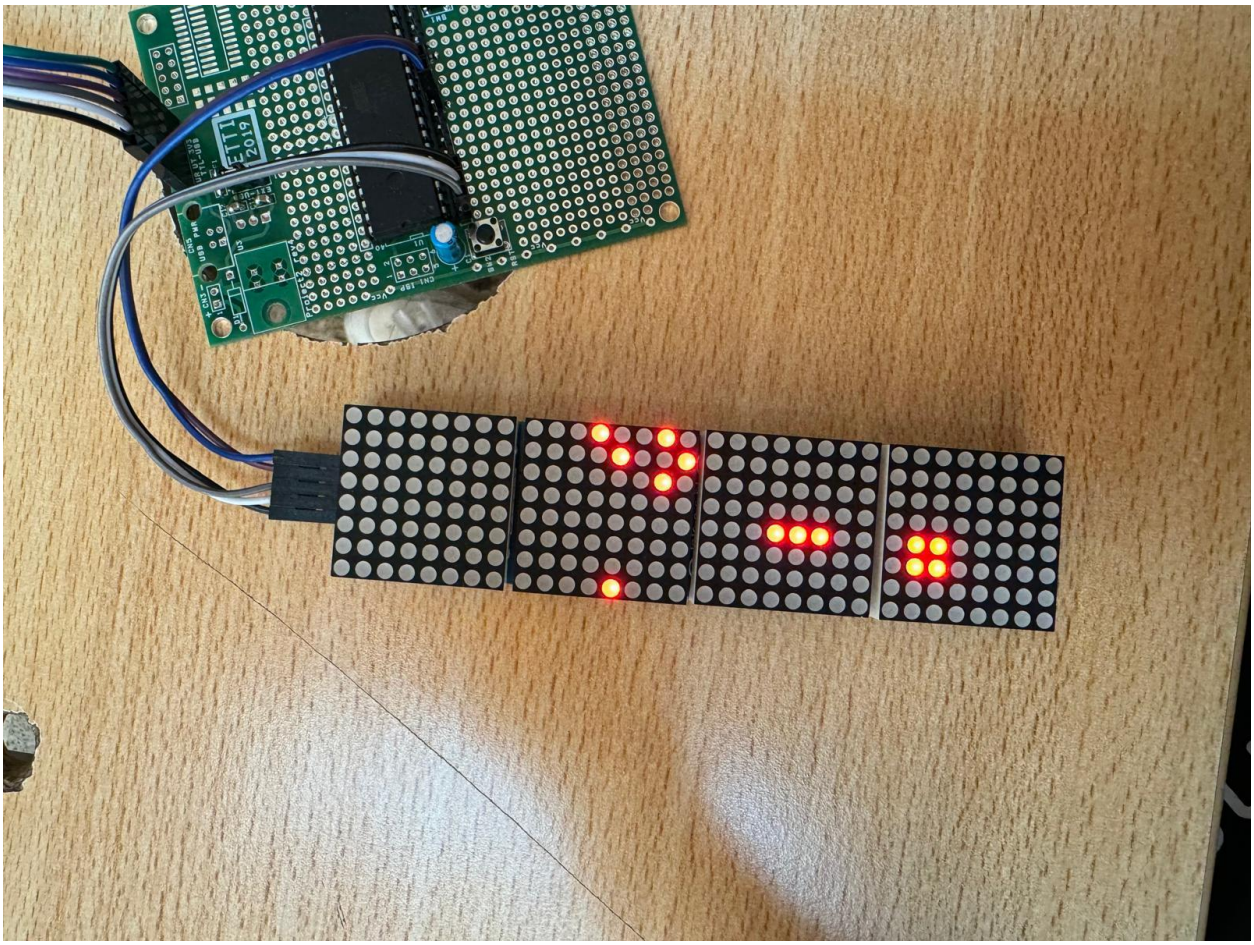

CN3
USB PWR

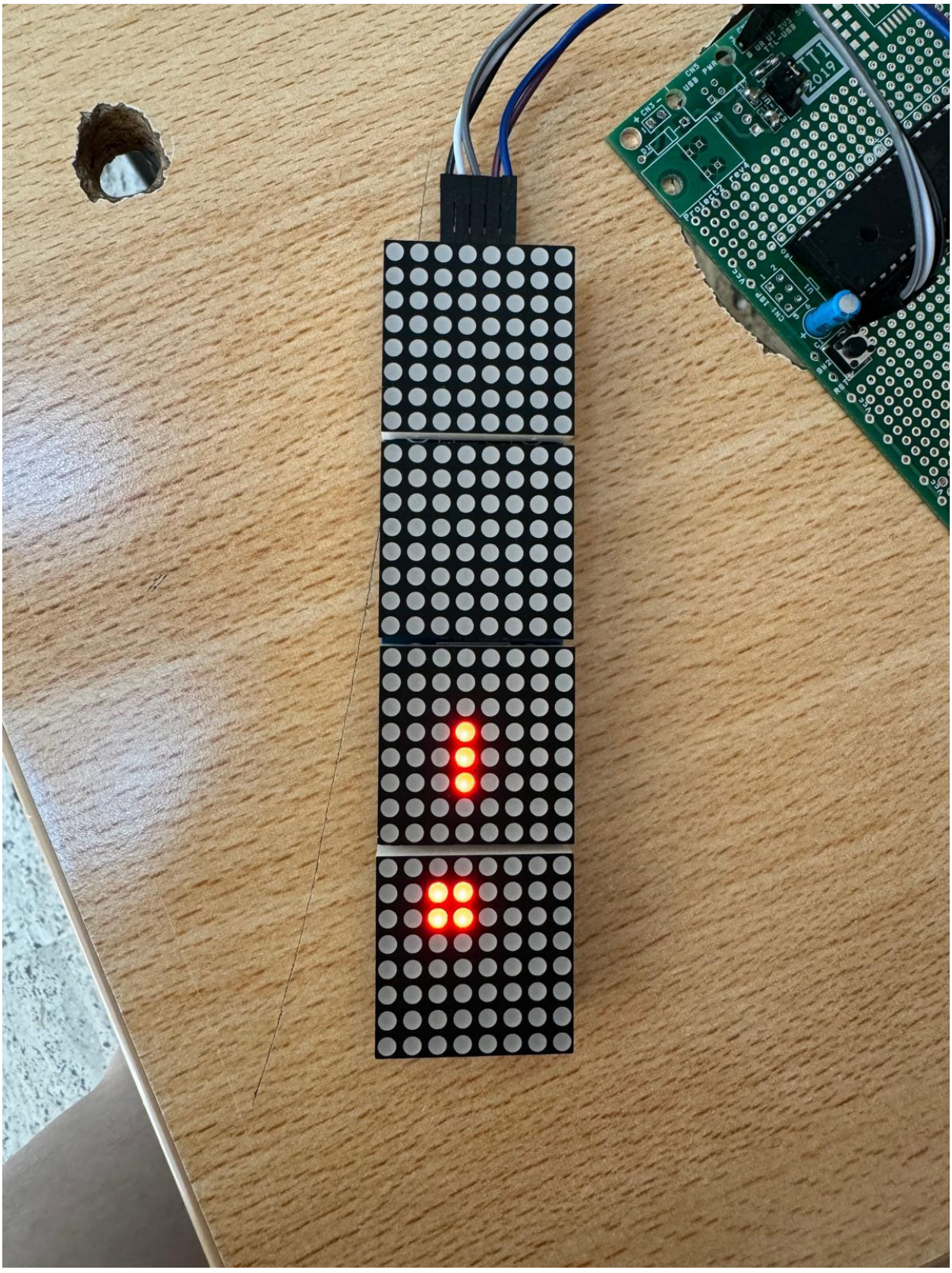







ETTI
2019


ETT
2019


CN5
USB PWR
2019

Bibliografie

https://www.tme.eu/ro/details/atmega164a-cu/gama-avr-8-bit/microchip-technology/

https://forum.arduino.cc/t/atmega164pa-pin-out/670518

https://www.optimusdigital.ro/en/led-matrices/4154-max7219-4-led-matrix-module.html?gad_source=1&gclid=CjwKCAjwx-CyBhAqEiwAeOcTdRYLuH1Fp8-BiLHt3IAPP15HZr0Y4i9gmKudpCGgOl1qYjReVFxdahoCJxkQAvD_BwE

https://playgameoflife.com/