Conway's Game of Life

Introduction

This project is a recreation of John Conway's game of life using Arduino, with LEDs representing the cells and their evolution displayed in real-time.

- The purpose of this project is entertainment and is inspired by an old game creating a physical representation of the cellular automaton for a more immersive and interactive experience.
- The Game of Life serves as a fascinating example of a cellular automaton, showcasing emergent behavior from simple rules. It has been extensively studied by mathematicians and computer scientists to understand patterns, complexity, and the principles of self-organization in dynamic systems.

Project Description

The rules are simple:

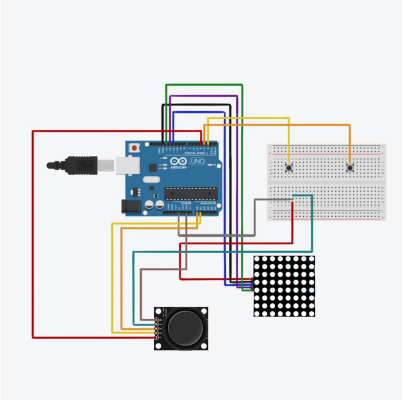
- Each cell on a grid can be in one of two states: alive or dead.
- The state of each cell in the next generation is determined by its current state and the states of its eight neighboring cells.
- If a live cell has fewer than 2 alive neighbors, it dies (underpopulation)
- If a live cell has more than 3 alive neighbors, it dies (overpopulation)
- If a dead cell has exactly 3 alive neighbors, it becomes alive (reproduction)
- Live cells with 2 or 3 alive neighbors continue to live on the next generation

Hardware Design

Components:

- Arduino Uno R3
- MAX7219 LED Matrix
- Wires

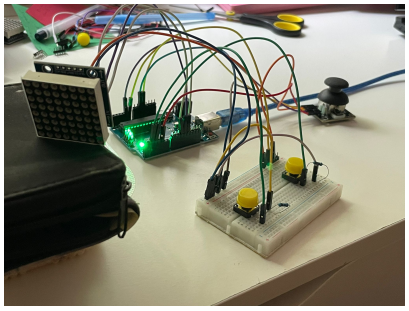
- Buttons
- Joystick



Software Design

- I am using LedControl.h library
- Source code proiect_game_of_life_care_merge_final.zip
- void randomize_cells(): gives a random configuration of cells, so the game has a starting point, or if you wanna change the cell configuration mid game;
- void countAliveNeighbors(int row, int col): counts the number of alive neighbors in the position world[row][col], diagonals also count;
- void next_generation(): figures out the next generation based on the rules i mentioned above, and is using countAliveNeighbors(row,col) and copies the new “world”(of cells) into the current one;
- void showOnLed(): shows on the physical LED, the configuration set on the integer matrix world[8][8] that is filled with ones and zeros
- For the joystick control part : if (joystickX != selectedCellX || joystickY != selectedCellY) - ensures that the selected cell state is updated only when the joystick position changes
- Inside the if condition, I updated the selectedCellX and selectedCellY variables using the map() function. This maps the joystick's analog values (ranging from 0 to 1023) to the corresponding cell coordinates in the world array (ranging from 0 to SIZE-1).
- For changing the state of the selected cell upon joystick click, I added code to check if the joystick click button (JOYSTICK_CLICK_PIN) is pressed and the joystickClicked flag is false. If the conditions are met, the state of the selected cell is toggled by calculating the opposite state using (cellState + 1) % 2. The state of the selected cell in the world array is updated, and the LED at the selected cell position is set to the new state using lc.setLed(). The joystickClicked flag is then set to true.

Obtained Results



Concluzii

- Apply what I learned in a practical way.
- Improve problem-solving skills.
- Develop presentation and documentation skills.

Download

proiect_game_of_life_care_merge_final.zip

Jurnal

Bibliografie/Resurse

https://www.instructables.com/Arduino-based-Bi-color-LED-Matrix-Game-of-Life/

https://steemit.com/utopian-io/@pakganern/how-to-control-8x8-led-matrix-using-joystick

https://forum.arduino.cc/t/help-with-8x8-led-matrix-and-max7219-code-conways-game-of-life-almost-done/131738

Export to PDF