EW-1: Final Evaluation Snake Game On Arduino

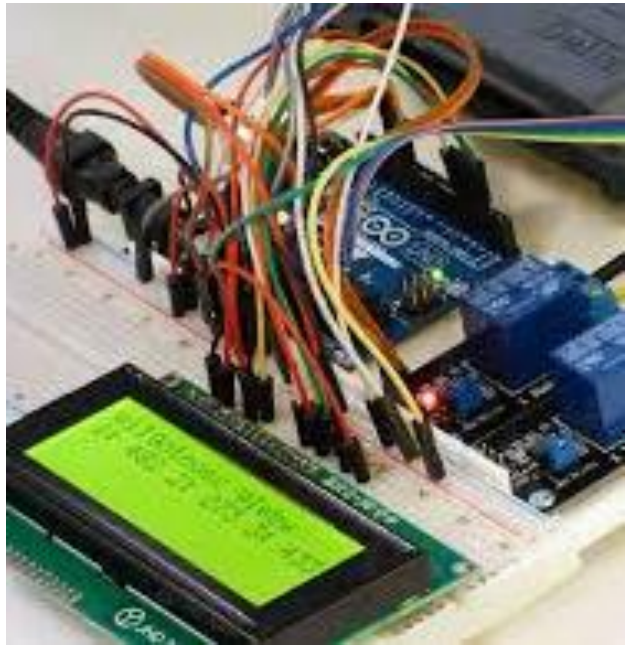
by Nikhil Venkat Atkuru and Sai Abhinav Nagamalla

Team ID : 4

TA : Soumil sir

Background and Motivation

- Building the Snake Game using Arduino and an LCD teaches foundational concepts in electronics, programming, and game design.
- It demonstrates how to interface components like joysticks and LCDs with microcontrollers.
- To test coding and algorithm skills.
- To face new challenges and acquire new skills.
- It is a great opportunity to understand working with microcontrollers like Arduino which will help us in future complex projects.
- To deepen understanding of Arduino functions (all the functions under LiquidI2C header file and random function working).
- Enjoyment of playing a game created by ourselves!
- To work as a team and coordinate with each other.

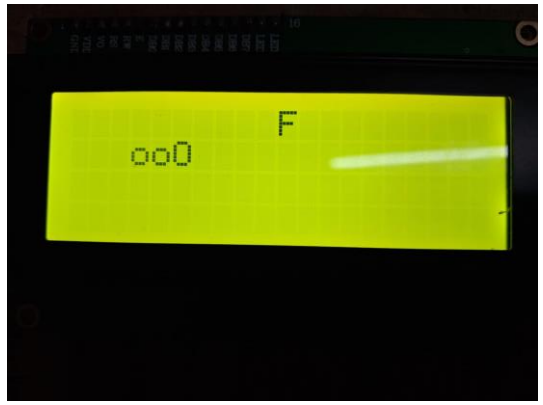


Problem Statement

- Make a classic arcade game “Snake Game” using Arduino
- Rules of the game are as follows:
  1. Generates apples and bombs at random position.
  2. Length of the snake increases as it eats an apple.
  3. Length increases till 10 after which the pace of the game increases with eating apples.
  4. Eating a bomb or colliding with snake itself ends the game.
  5. Rolling over the snake across the screen is allowed.
  6. Food is represented with F and Bomb is represented with B while head of snake is represented with O and the rest of the body with o
  7. The initial length of the snake is 3.
  8. Bomb disappears after 5 seconds while the food stays till either it is eaten, or after 7 seconds.

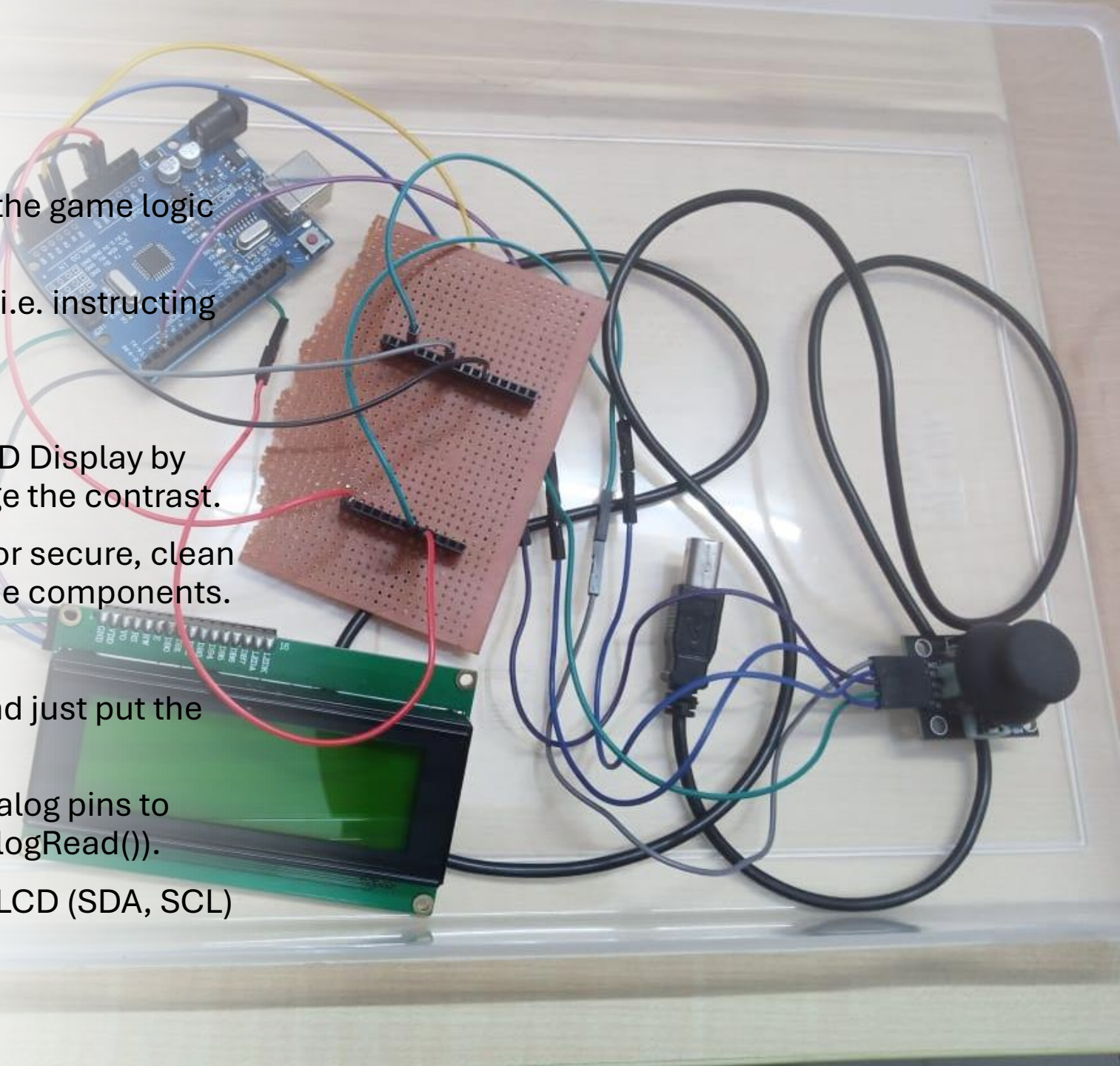
Salient features in our game:

1. Controlling the snake using a joystick.
2. Pausing and Resuming the game by clicking joystick.
3. Display the score and high score at the end of each round.
4. The Bomb and Food are not spawned within the three squares directly in front of the snake, ensuring the player has sufficient time to react.
5. The food disappears if not eaten in 7 seconds and the bomb disappears after 5 seconds
6. The snake head is generated at random position at the start of the game.
7. When the game is resumed or restarted, a 3 2 1 countdown is initialized, giving heads up to the player.



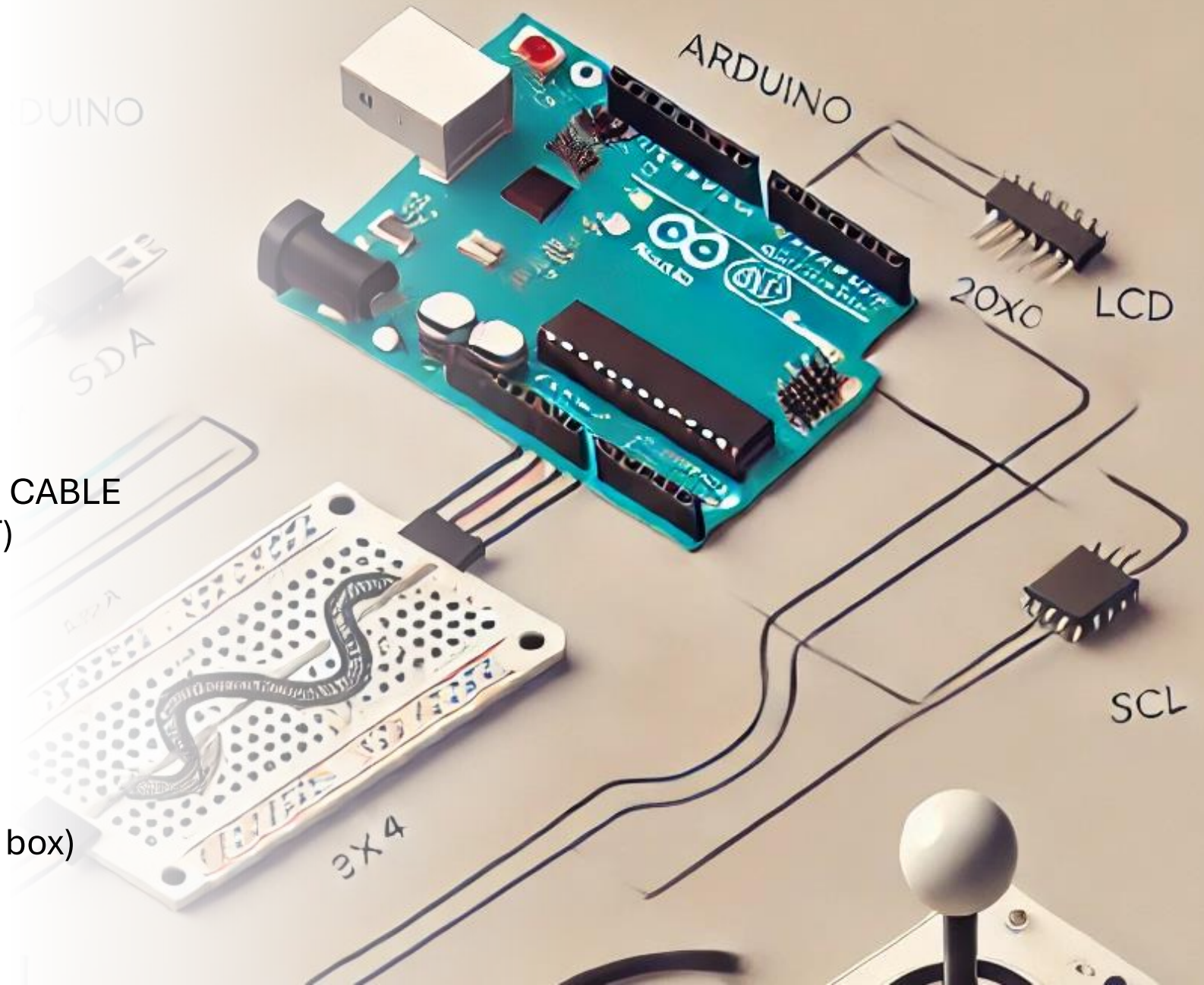
Proposed Design

- **Arduino UNO :** The main controller used to run the game logic and handle input/output.
- **Joystick:** To control the movement of the snake i.e. instructing the snake to move up, down, left or right.
- **LCD Display:** To show the gameplay.
- **I2C Module:** For the easier controlling of the LCD Display by minimizing the connections and easier to change the contrast.
- **PCB Soldering**: Solder components onto PCB for secure, clean connections of VCC and GND i.e. to power all the components.
- **Jumper Wires:** For the connections.
- **Game box:** Place all the wiring inside the box and just put the LCD display and the joystick out.
- **Joystick Control**: Connect joystick's X and Y analog pins to Arduino for snake movement (mapped with analogRead()).
- **LCD Display (I2C)**: Use I2C module to connect LCD (SDA, SCL) to Arduino for displaying game info.

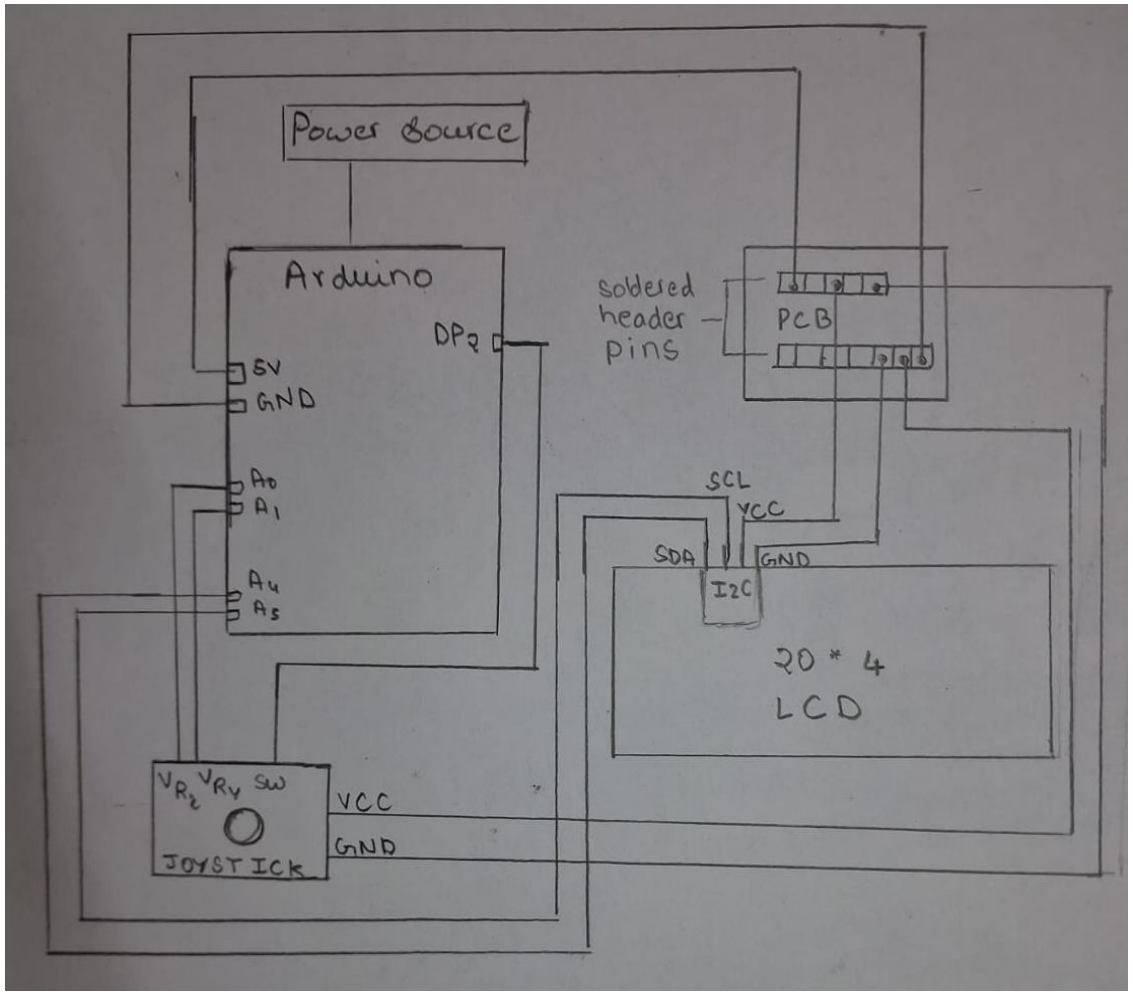


Components Used

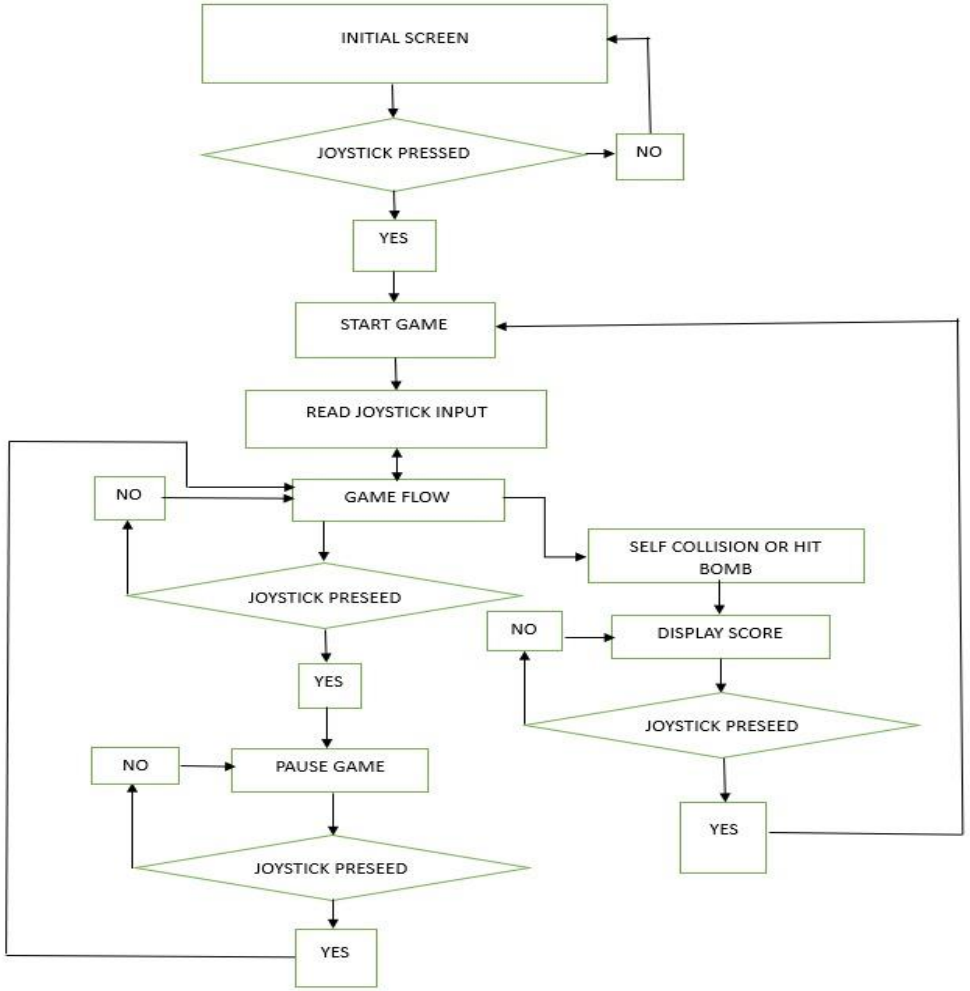
- 20 * 4 LCD
- JUMPER WIRES
- I2C MODULE
- ARDUINO UNO AND USB CABLE (TO POWER THE CIRCUIT)
- JOYSTICK
- PCB-0
- Soldering kit
- Power Source
- Carboard Box (as a game box)



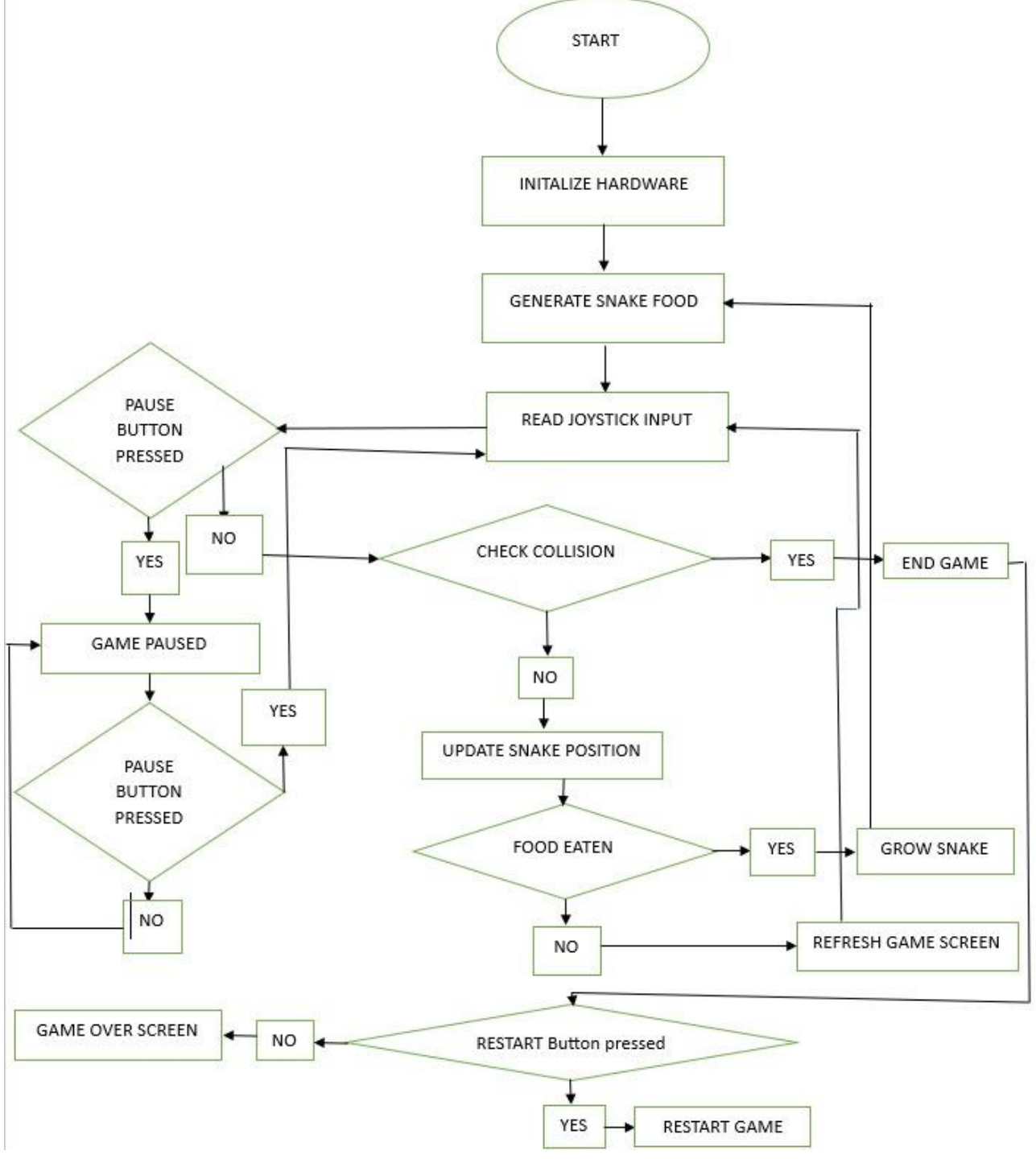
Circuit Diagram



Block Diagram



Flowchart (Overview of Code Flow)



Demo

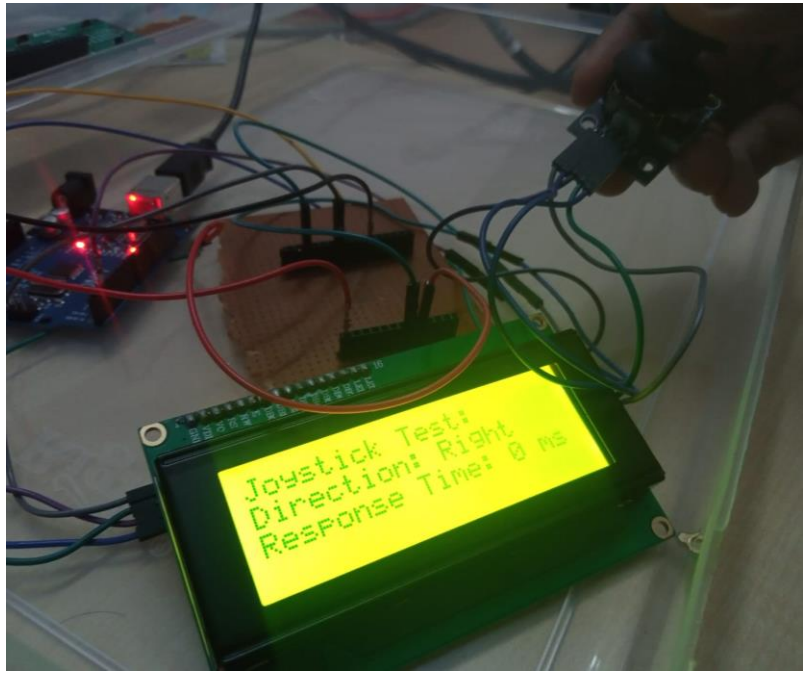
The google drive link for the same is provided below

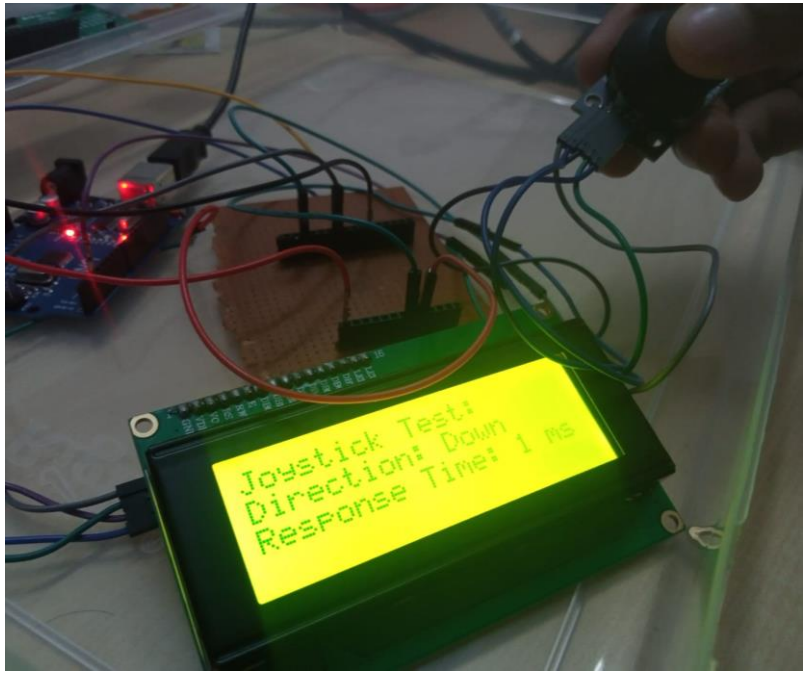
Demo link

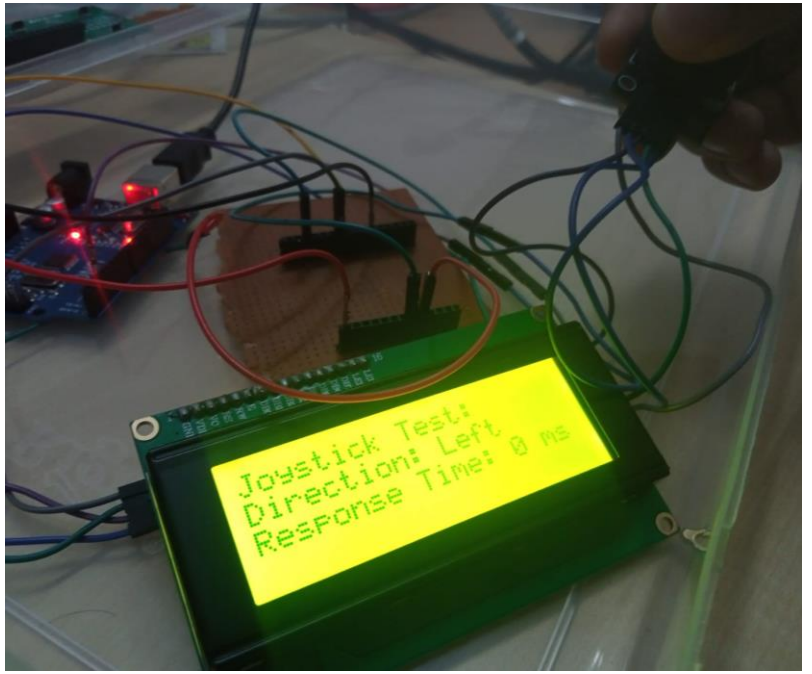
Key Performance Indicators

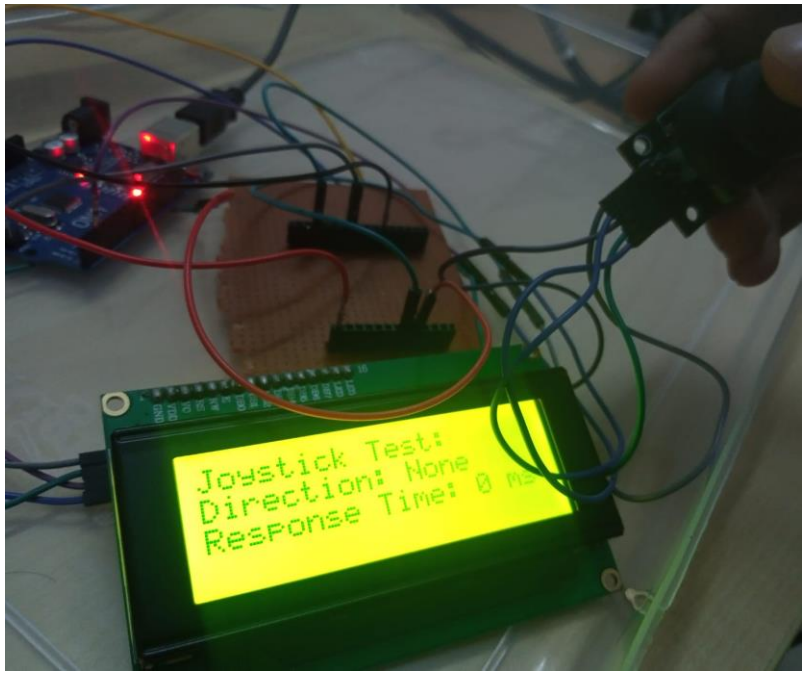
- **Program Storage Usage:** 6898 bytes (21%) used — within optimal limits, leaving 79% free.
- **Dynamic Memory Usage:** 720 bytes (35%) used — with 65% RAM free, which is sufficient for additional variables.
- **CPU Load:** Likely low, as memory usage is not near maximum capacity and game responsiveness is good.
- **Scalability:** Can add more features without performance loss.
- **Reliability:** Consistent performance over time.
- **User Experience:** Simple rules and user-friendly controls.
- **User Learning Curve:** Easy to get used to the game and get good at it.
- **Game Processing Time:** Likely under 6-7 ms per game loop, based on screen update time. Joystick reading is updated very quickly i.e. 0-1 ms.
- **System Resources:** Plenty of room for expansion, with 79% of program storage and 65% of dynamic memory available.

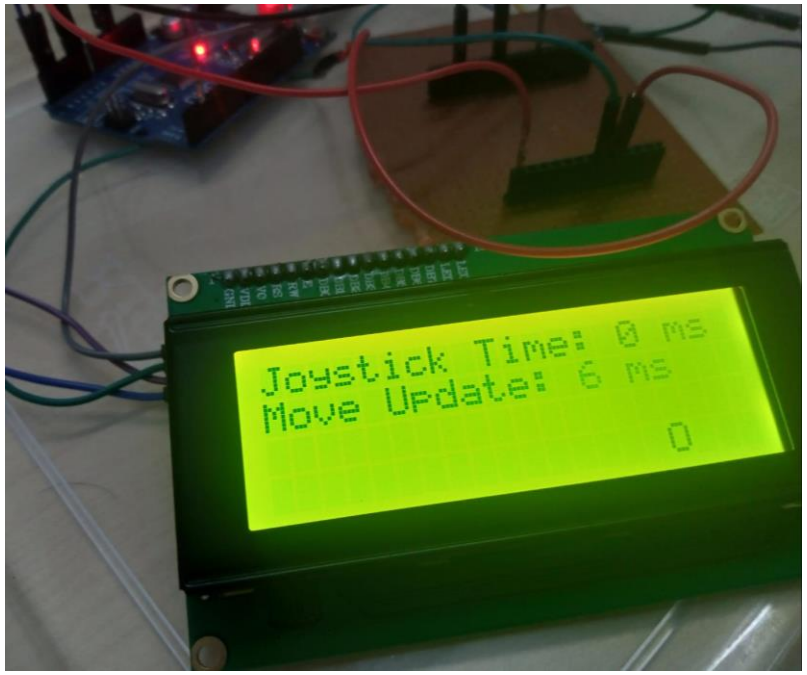
```
Sketch uses 6898 bytes (21%) of program storage space. Maximum is 32256 bytes.
Global variables use 720 bytes (35%) of dynamic memory, leaving 1328 bytes for local variables. Maximum is 2048 bytes.
```

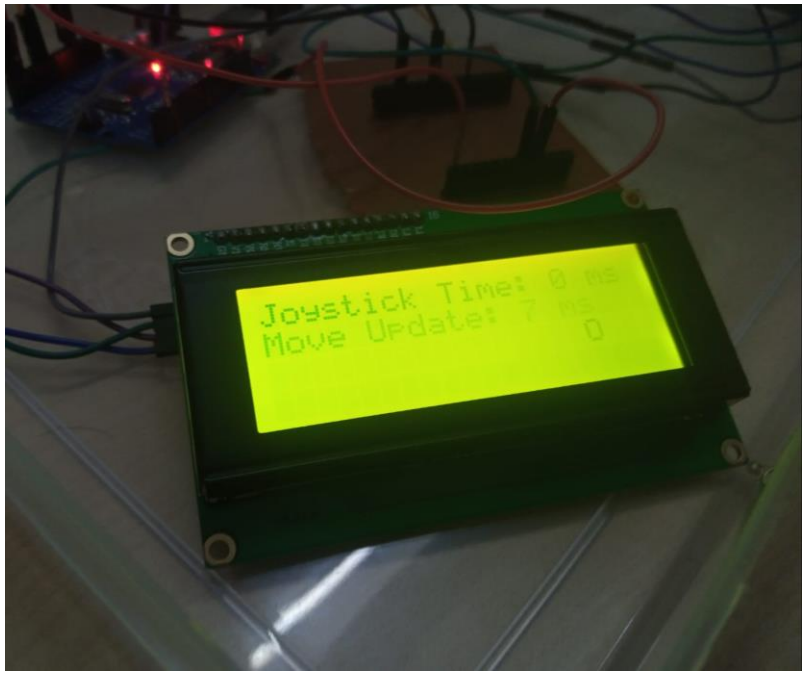

Joystick Test:
Direction: Right
Response Time: 0 ms


Joystick Test:
Direction: Down
Response Time: 1 ms


Joystick Test:
Direction: Left
Response Time: 0 ms


Joystick Test:
Direction: None
Response Time: 0 ms


Joystick Time: 0 ms
Move Update: 6 ms
0


Joystick Time: 0 ms
Move Update: 7 ms
0

Cost Efficiency and Practical Deployment

- **Ease of setup:** Low complexity since the hardware setup involves very few components.
- **Maintenance and Updates:** Can be done easily with changes in the code but requires understanding of game logic (Easy - Medium Complexity).
- **Deployment Scalability:** Scaling could involve replicating the setup, which is quite straightforward but not so easily for multiplayer setup in arcade setting.
- **Cost Efficiency:** Low-cost setup (the required components (e.g., Arduino, LCD, joystick) are inexpensive and widely available.)
- **Power Consumption:** Voltage = 5 V, Current drawn = 139.4 mA

$$P(\text{Power}) = V \times I = 5V \times 0.1394A = 0.697W$$

$$E(\text{Energy}) = 0.698W \times 1h = 0.698Wh$$

Cost of the solution

- Arduino UNO Market price = ₹ 420
- Joystick = ₹ 30
- Printed Circuit Board = ₹ 60
- The 20 * 4 LCD Display with I2C Module costs ₹660, offering more functionality. An alternative is the 8x8 LED dot matrix (₹360), but it has limited functionality as it only shows on/off states, making it impossible to differentiate between food and bomb.
- Miscellaneous costs for wires, glue gun, etc.
- Operational cost = Minimum due to very less power being drawn.

Our Individual Contributions...

Contributions by Abhinav:

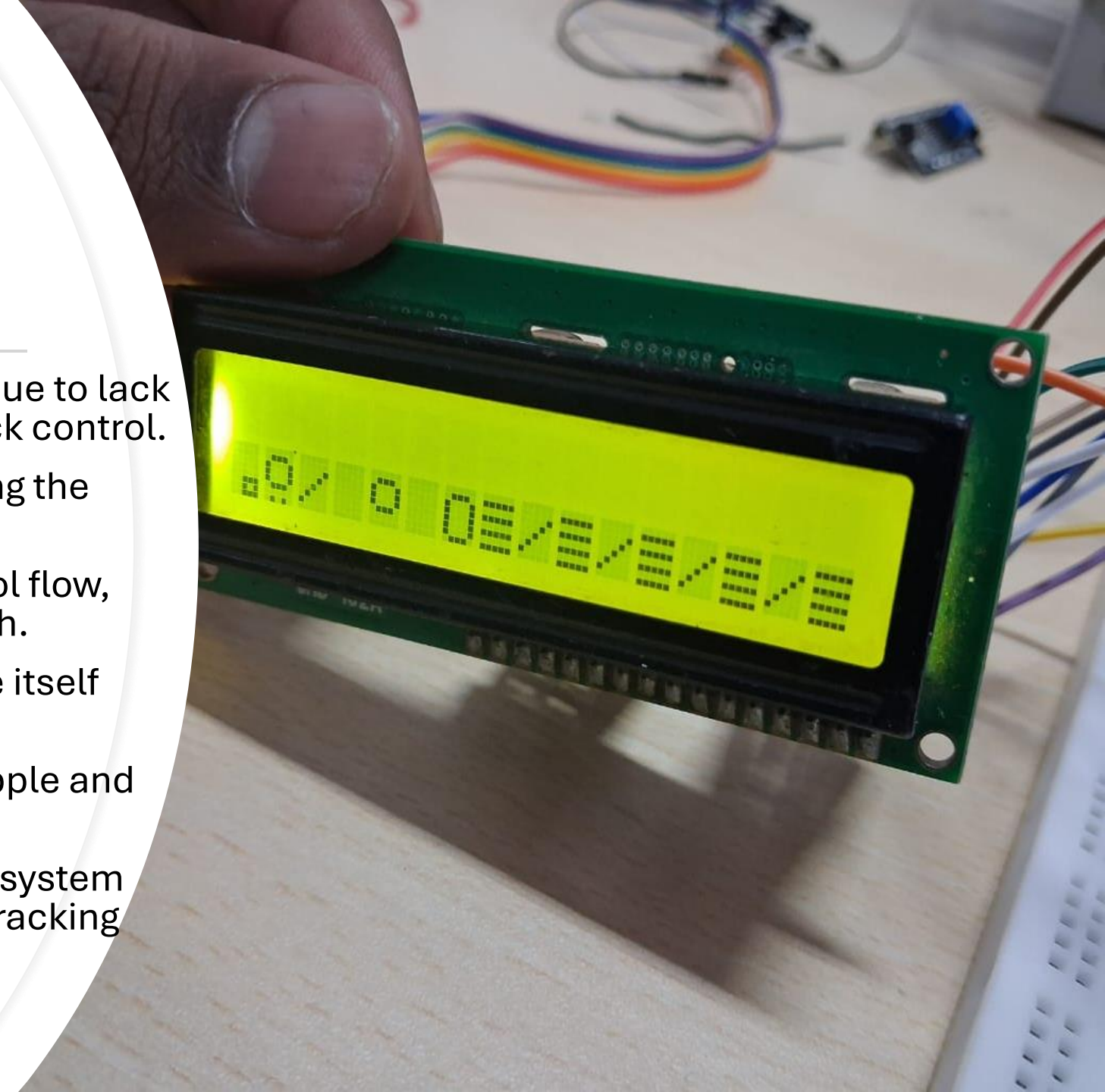
- Designed the basic structure of the code.
- Assisted in soldering and wiring
- Proposed various ideas for the game.
- Refined the game logic to improve gameplay.
- Prepared the presentation
- Compiled and authored the report

Contributions by Nikhil:

- Enhanced and optimized the code.
- Handled the soldering and wiring.
- Developed the core game logic.
- Implemented the game logic into the system.
- Co-authored the report.
- Made the game box

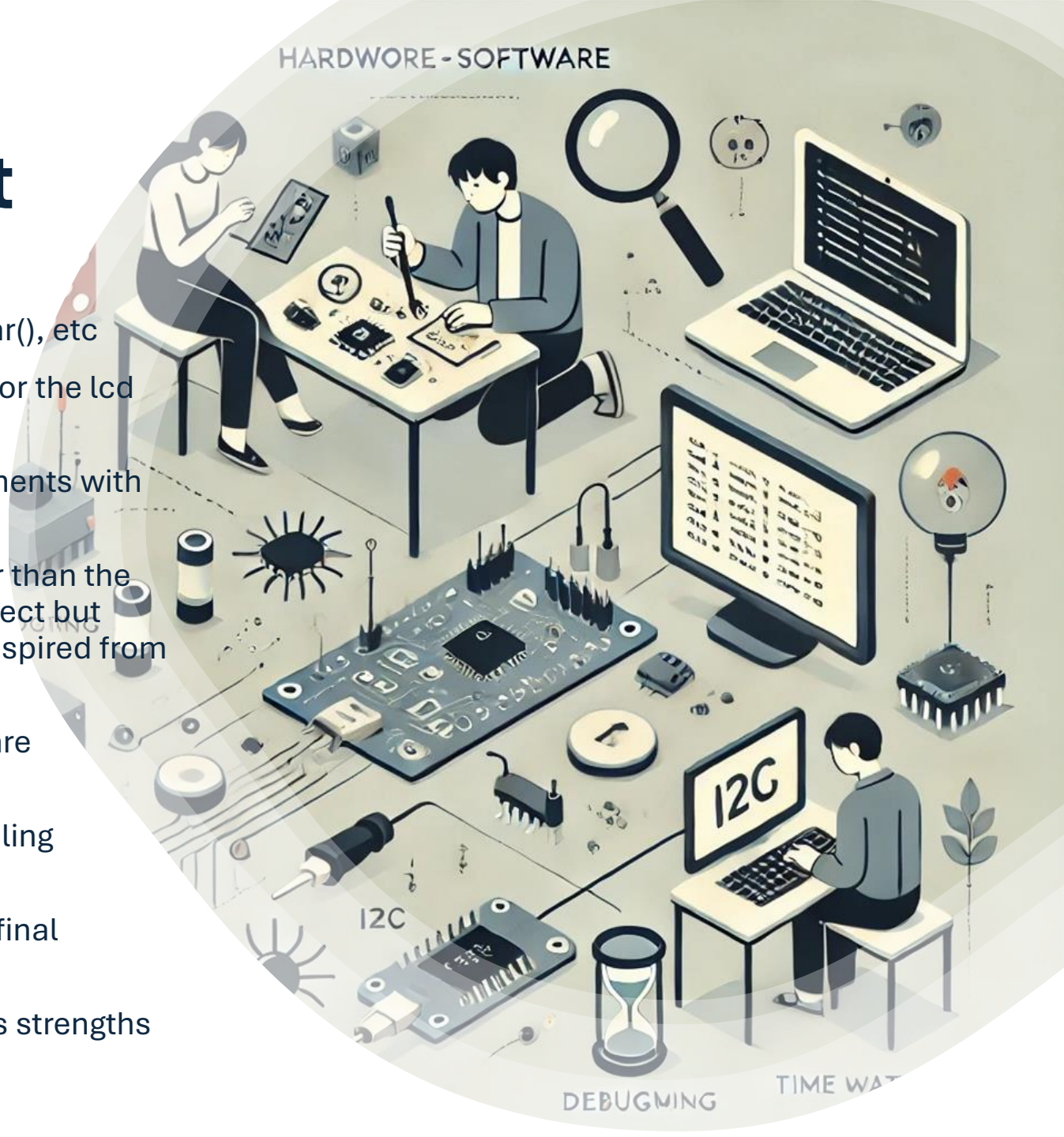
Technical difficulties and Challenges we Faced

- Unable to simulate the project on Tinkercad due to lack of support for the LiquidI2C library and joystick control.
- Faced challenges in developing and integrating the game logic effectively.
- Struggled with function placement and control flow, requiring us to revisit and refine it from scratch.
- Food and Bomb being generated on the snake itself (fixed it after a lot of effort!)
- Attempted to create pixelated icons for the apple and bomb but encountered display issues.
- Tried implementing a multi-player high-score system with menu selection but faced difficulties in tracking individual scores. (failed in this)



Learnings from the project

- How random generation works
- LiquidI2C header file and its functions like lcd.print(), lcd.clear(), etc
- Purpose of I2C Module and how does it simplify the controls for the lcd display
- Understanding how to seamlessly integrate hardware components with software logic for real-time performance.
- Other ways to approach the project i.e. other algorithms other than the one we finally used (we explored other algorithms for this project but found them too complex. So, we came up with our own one inspired from others)
- Developing strategies to identify and fix issues in both hardware connections and code logic effectively.
- Gaining practical experience in soldering, wiring, and assembling components into a functional system.
- Balancing time between experimentation, development, and final implementation stages of the project.
- How to work as a team and distribute work, know each other’s strengths and weaknesses and how to cover them.



References we used

- ChatGPT help to rectify our sequences of calling functions and rectify our game logic.
- To check the functioning of the lcd
- YouTube tutorial on how to use I2C Module
- A tutorial on our project done on a different display and different logic.
- Another approach to our project (different algorithm from what we used)


Thank You