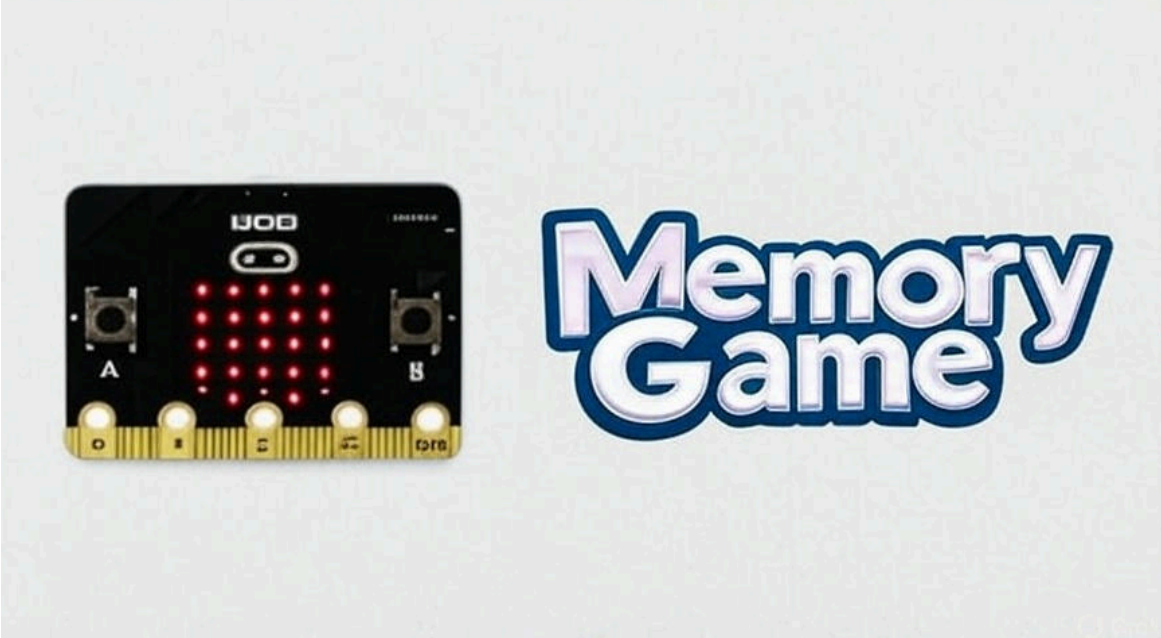
EGC 121 / Computer Architecture

Memory Game

Micro:Bit Project Report



Team Members:

- Ayush Patel (BT2024054)
- Kabir Ahuja (BT2024004)
- Kanav Kumar (BT2024021)
- Dayal Gupta (BT2024167)
- Parth Malhotra (BT2024197)
- Sachin Nain (BT2024203)
- Tanmay Dixit (BT2024016)

Introduction:

This project involved developing a **Memory Path Game** using the **BBC Micro:bit V2**. The team wrote the game logic in Python using the MicroPython environment and flashed the code to the Microbit using the **Micro:Bit Python Editor**. Once uploaded, the game was tested and run directly on the Microbit device.

The code was deployed via USB connection, and the Microbit's built-in features — including the **LED matrix**, **buttons**, and **accelerometer** — were utilized to create an interactive memory challenge. The game logic was validated through repeated playthroughs to ensure smooth input handling and correct path verification.

This project enabled the team to gain practical experience with embedded systems, MicroPython, and hardware-software interaction using Microbit.

Micro:Bit Functionality:

This project is a fun **memory-based game** built using the **BBC Micro:bit V2**. When the Microbit is **shaken**, it displays a **random LED path** on its 5x5 display. The player must replicate the path using:

- **Button A** – Move left (double press: up)
- **Button B** – Move right (double press: down)
- **Buttons A+B** – Quit the game

It uses the Micro:bit V2's key features:

- **5x5 LED matrix** for path display
- **Buttons A and B** for directional input
- **Accelerometer** to detect shakes and start the game
- **Timers** to detect single vs. double presses

The game tests memory and coordination while demonstrating interactive Micro:bit features.

Video Recording:

A **demonstration** of the working project has been recorded and uploaded to Google Drive. The video showcases the micro:bit detecting **various button presses and gestures**

Watch here:
https://drive.google.com/file/d/1PsbJtFaRprxVQib4Qu50ni7nIJhfd5Ud/view

Challenges Faced:

1. **Button Input Timing Conflicts**
   Differentiating between single and double button presses required precise timing logic and caused unintended moves during early testing.

2. **Path Display Visibility**
   Ensuring the random path was clearly visible on the LED matrix for the player to memorize required tuning the brightness and display timing.

3. **Limited Input Options**
   Mapping four directions to only two physical buttons (A and B) added complexity to the control design and user interaction.

4. **Debugging Without Serial Output**
   Debugging game logic on the Microbit was challenging due to limited real-time feedback, especially without using serial print statements.

5. **Edge Case Handling**
   Preventing out-of-bound moves and handling corner positions required extra care to avoid gameplay bugs.

Future Improvements:

1. **Scoring System**
   Add a points-based system to track user performance and encourage replayability.

2. **Difficulty Levels**
   Introduce multiple levels with increasing path lengths or faster display times to make the game more challenging.

3. **Sound Feedback**
   Use the Microbit V2's built-in speaker to provide audio cues for actions like correct moves, errors, or game outcomes.

4. **Wireless Multiplayer Mode**
   Implement radio communication between two Micro:bits for cooperative or competitive gameplay.

Conclusion:

The Memory Path Game project successfully demonstrated the capabilities of the **BBC Microbit V2** for creating interactive applications using MicroPython. By leveraging onboard features such as the LED matrix, buttons, and accelerometer, the team was able to build a functional and engaging memory-based game. The project also involved hands-on experience with coding, debugging, and flashing programs onto the Micro:bit device. Overall, this project not only showcased the practical application of embedded programming but also laid the foundation for future enhancements involving sound, difficulty levels, and multiplayer functionality.