Microbit Assignment Report

Team Members:

NAME	ROLLNO	NAME	ROLLNO
B Gagan Syam Reddy	Bt2024032	Christine	BT2024045
P Yashwanth Reddy	Bt2024103	M Sai Ranga Reddy	BT2024185
Sankalp Gadamsetty	Bt2024182	Tom Stanley	BT2024080
Aadyant Neog	Bt2024186	Aadyant Neog	BT2024186

Python Project: Interactive Two-Mode Game using Buttons and Accelerometer

1. Objective of the Project

The objective of this project was to explore the functionalities of the Microbit board by creating an interactive game that utilizes multiple onboard features including:

- LED matrix display
- Button inputs (A and B)
- Built-in accelerometer The game includes two modes:

1. A two-player tic-tac-toe game.
2. A single-player reflex game where players catch randomly positioned targets using tilt control.

2. Implementation Details

Hardware Used:

- Microbit v2 board

Features Used:

- Button_a, button_b: for input selection
- accelerometer.get_x(), accelerometer.get_y(): to control cursor movement
- display.set_pixel(), display.clear(), display.scroll(): for visual output

Mode 1: Two-Player Grid Game

- Triggered by pressing button A
- A pre-defined 5x5 grid with blocked cells is initialized
- Players move a cursor using tilt (accelerometer) and take turns placing marks:
    - Player 1 uses button A ○
      Player 2 uses button B
- A player wins by aligning 3 in a row (horizontal, vertical, or diagonal)
- A draw is declared if the board fills up

Mode 2: Target Catching Reflex Game

- Triggered by pressing button B
- A 30-second timer starts
- Player moves the cursor by tilting the Micro:bit
- A target randomly appears on the screen
- Player must "catch" the target by pressing button A when cursor overlaps it
- Score is incremented and new targets are generated

3. Challenges Faced

1. Motion Sensitivity

- **Issue:** Accelerometer values were too sensitive, causing jerky movement.
- **Solution:** Divided raw accelerometer values by a factor (400 in game 1, 195 in game 2) to smooth the control.

2. Button Debouncing

- **Issue:** Holding a button caused repeated inputs, leading to unwanted multiple actions.
- **Solution:** Added conditional checks and used sleep delays to stabilize input reading.

3. Grid Overflow

- **Issue:** Tilt-based movement risked cursor going off-grid.

- **Solution:** Manually bounded x and y values between 0 and 4 after each move.

4. Results and Future Improvements

Results

- Successfully implemented two distinct games using a single Micro:bit
- Game logic handles player turns, win/draw detection, and real-time scoring
- Demonstrated multiple features of the board in an engaging way

Future Improvements

- Save high scores using persistent storage
- Expand the game modes or add difficulty levels
- Add sound using the Microbit buzzer for feedback

5. Files Submitted

- microbit_game.py: Source code for the project
- microbit_game.hex: Compiled HEX for Microbit
- microbit_game_demo.mp4: Demo video of the game
- Microbit Assignment Report.pdf (Common for both the projects)

Arm Assembly Language Project: LED Cursor Navigation using Button A and Button B on Microbit

1. Objective of the Project

To implement an ARM assembly language program for the Microbit that allows the user to control a light pixel on the Microbit. The light pixel starts at a fixed position and moves left when Button A is pressed and right when Button B is pressed.

2. Implementation Details

Hardware Used:

- Microbit v2 board
- LED Matrix

- Button A (connected to P0.14) and Button B (connected to P0.23)

Microbit GPIO Reference:

- P0.14 --- Button A
- P0.23 --- Button B
- P0.31 --- COL3 (LED column control)
- P0.15 --- ROW3 (LED row control)

Program Logic:

Initialization:

- Set GPIO pins connected to the desired LED column and row as output.
- Ensure LED is initially turned ON at a defined location (e.g., column 0, row 2).

Polling Loop:

- Continuously check for button presses by reading input GPIO pins.
- If Button A (P0.14) is pressed:
  - Decrement cursor column index, clamp to 0.
  - Update the GPIOs to turn ON the new LED.
- If Button B (P0.23) is pressed:
  - Increment cursor column index, clamp to 4.
  - Update the GPIOs to turn ON the new LED.
- Clear previous LED before lighting the new one.

3. Challenges Faced:

- Mapping pin numbers to correct GPIO bit positions
- Managing the clearing and setting of LEDs through direct register manipulation

4. Conclusion:

This project successfully demonstrates low-level hardware control using ARM Assembly on the BBC Microbit platform. By directly manipulating GPIO registers, we gained an understanding of:

- Microcontroller GPIO configuration

- Bitwise operations for hardware control

5. Files Submitted:

- microbit_light_movement.s
- microbit_game.hex: Compiled HEX for Microbit
- microbit_light_movement_demo.mp4
- Microbit Assignment Report.pdf (Common for both the projects)