GE107 PROJECT REPORT

AIR MOUSE

GROUP-11

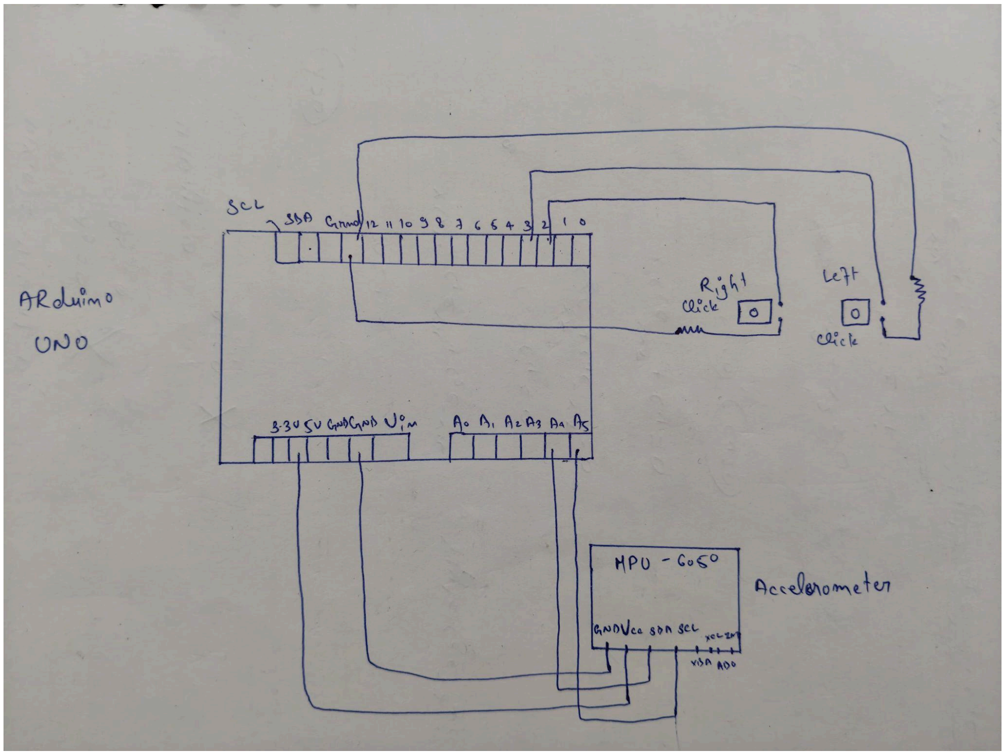
Objective

The goal of this project is to build an **Air Mouse** – a wireless motion-controlled cursor system – using an Arduino Uno and an MPU6050 (accelerometer + gyroscope) sensor. By interpreting movement data from the accelerometer and converting it into cursor movement commands, the device eliminates the need for a flat surface, making it an ideal input device for smart devices and computers.

Components Used:

S.No	Component	Specific Use
1	Arduino Uno	Acts as the main microcontroller to read sensor data and send it to the PC.
2	MPU6050 Accelerometer + Gyroscope	Detects hand movement and orientation in real time.
3	Breadboard	Used to build the circuit without soldering and easily connect components.
4	Jumper Wires (M-M and M-F)	Connect the Arduino, MPU6050, buttons, and other components on the breadboard.
5	Push Buttons	Used to simulate left-click and right-click mouse actions.
6	10kΩ Resistors	Used as pull-down resistors for button inputs to avoid false triggering.
7	Micro USB Cable	Provides power to the Arduino and enables serial communication with the PC.
8	Personal Computer/Laptop	Receives serial data and runs the Python script to control the mouse cursor.

Circuit Diagram:



Working Principle:

The MPU6050 detects hand motion through acceleration and angular velocity. The Arduino reads this data via I2C and converts it into mouse movement signals. A Python script (or Processing sketch) on the computer receives the serial data and uses a library like pyautogui to move the cursor.

- **MPU6050**: Measures orientation in real-time.
- **Arduino Uno**: Collects MPU6050 data and transmits over USB.
- **PC Python Script**: Receives data and simulates mouse movement.
- **Buttons**: Used for left and right click, volume control simulation.

Challenges Faced and Solutions:

Challenge	Description	Solution
1. MPU6050 not responding	Sensor data wasn't being read	Checked wiring and power – switched to 3.3V for VCC
2. Jittery cursor movement	Cursor was unstable	Added filtering logic and reduced sensitivity in code
3. Serial lag	Delay in real-time cursor tracking	Optimized code for faster serial communication and buffer clearing
4. Button debounce issues	Multiple clicks on a single press	Added debounce logic in Arduino code using `millis()`

Advancements and Conclusion:

- This project successfully demonstrates the use of motion sensors to control a computer cursor, highlighting its potential in gesture-based human-computer interaction.
- Future versions can be enhanced by integrating **wireless communication** (e.g., Bluetooth, ESP32) to eliminate the need for physical connections.
- **Gesture-based controls** can be developed further to replace mechanical buttons, offering a more seamless and intuitive user experience.
- Miniaturizing the setup with **compact hardware and battery support** could lead to wearable applications such as smart gloves or rings.
- Overall, this project provided valuable hands-on experience in sensor interfacing, embedded systems, and interactive design.

Output Demonstration

A short demo video is recorded and attached as separate file to showcase:

- Objective and Functionality
- Cursor movement with hand motion
- Button press actions
- Real-time responsiveness

Week-wise work

Week	Tasks
Week 1	Finalized project idea, researched components, listed and arranged materials.
Week 2	Built the circuit: connected MPU6050, push buttons, and set up breadboard.
Week 3	Wrote Arduino code and Python script for cursor control via motion.
Week 4	Tested system, resolved issues like sensor noise and button debounce.
Week 5	Continued optimization, recorded demo video, started report preparation.

Submitted By: Group-11

1. Aarya Choudha(2023MMB1403)
2. Akshit Choudhary(2023MMB1406)
3. Daksh Garg(2023MMB1410)
4. Urvashi Walade(2023MEB1396)
5. Yogendra Magdum(2023MEB1401)