Student: Tanish Sahu Roll Number: 23B1816

PROJECT TITLE : ACCELERATING PONG 🎮

Project abstract:

Introduce your project:

1) **What's the grand goal?**
   The main goal of this project was to create a challenging and exciting two-player Pong game on a 8✗8 LED matrix and *Arduino UNO* using simple controls, without relying on pre-made tools or libraries. This allowed us to explore and solve the tough problems that come with working directly with the hardware and programming libraries. As players compete, the game gets more intense because the ball moves faster and faster with time, making it a fun test of skill. By achieving this, we showed that even with basic components, we can create something innovative and engaging.

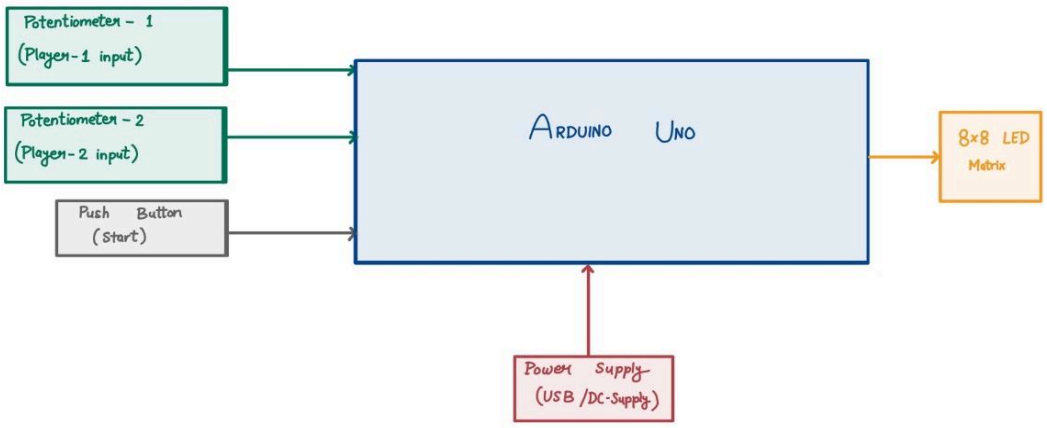
2) **What are the main inputs and outputs? i.e. during the demo what do you expect to demonstrate?**
   In this project, the main inputs are a push button and two 10K potentiometers. Pressing the button starts the game, and rotating the potentiometer knobs lets each player control their paddle. The main output is the 8✗8 LED matrix, which displays the moving ball and paddles. During the demo, I'll showcase player controls, ball movement, increasing speed over time, and how the game responds dynamically to player actions.

3) **What is the role of the Arduino in your project?**

- All 16 pins of the 8✗8 LED matrix—8 for rows(Cathodes) and 8 for columns(Anodes) —are directly connected to Arduino. It continuously updates the matrix to visually represent the current positions of the ball and both paddles.
- The Arduino reads the push button input after each game ends. Once the button is pressed, a new game starts and it stops taking the feedback until the game is running.
- After the game starts, in every loop, the Arduino reads analog values from both potentiometers. These values determine the real-time vertical position of each player's paddle, which are then updated on the LED matrix.
- Updates the position of the ball at regular intervals, rather than in every loop, to manage the game's pace and provide players with sufficient time to react.
- To make the game more exciting and competitive, the Arduino gradually increases the ball's speed after a fixed number collision of the ball with pedals. This introduces difficulty over time and challenges the reflexes of both players.
- Along with gameplay, the Arduino also runs simple starting and ending animations on the LED matrix. These animations enhance the user experience and give the project a polished, finished feel.

Project Detail:

1) Hardware block diagram



2) Give details of sensors and output mechanisms used.

- **Input Mechanism:**
  1. **Push Button:**

     Purpose: Starts the game after a previous game ends.

     Type: Digital input device (momentary switch).

     Connection: Connected to a digital pin on the Arduino with a pull-down or pull-up resistor configuration.

     Functionality: The Arduino continuously checks its state after game-over. Once pressed, a new game begins.

  2. **Two 10K Potentiometer:**

     Purpose: Control the vertical position of each player's paddle.

     Type: Analog input device.

     Connection: Each potentiometer's pin is connected to an analog input pin of the Arduino.

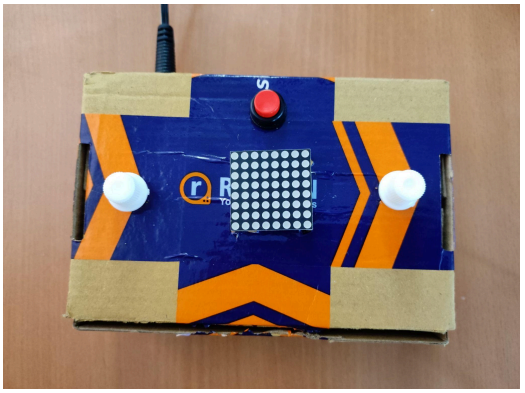
     Functionality: As the knob is turned, the voltage changes, which the Arduino reads and maps to the LED matrix paddle positions.

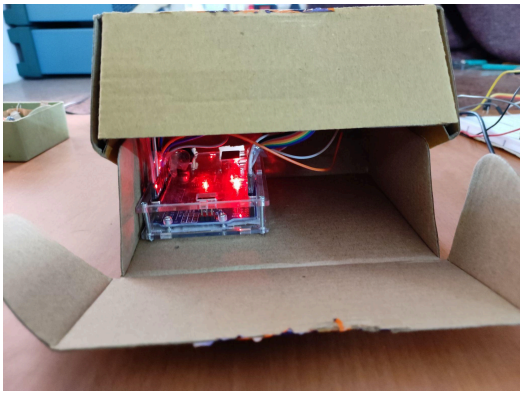
- **Output Mechanism:**

  The output mechanism of this project is the 8×8 LED matrix, which visually represents the entire game. It displays the ball, the two paddles, and various animations like the start and end screens. The Arduino rapidly scans and updates the LEDs by switching rows and columns to create the illusion of smooth motion. As the game progresses, it updates the ball's position at set intervals and continuously reflects the paddle movements based on player input, providing a dynamic and interactive visual output.

3) Photos and Videos:

- **Photos:**





- **Video Link:** PH222_Project_Pong.mp4

List the main milestones achieved in each week:

- Week 1: In this week, I collected all the parts needed for the project and learned how the pins of the 8×8 LED matrix are connected. I also tried out some simple codes to light up the LEDs in different patterns to understand how the matrix works.

- Week 2: In this week, I set up and tested both paddles using the two potentiometers. I wrote the code so that turning the knobs moved the paddles on the LED matrix. The paddles worked well and responded correctly.

- Week 3: In this week, I added the ball to the game and programmed its movement. I made sure the ball bounced off all the walls and paddles in the correct direction. I also added winning conditions so the game knows when a player misses the ball.

- Week 4: In this week, I added the ball's acceleration feature, so it moves faster as the game goes on. I also connected a push button to start the game and included starting and ending animations on the LED matrix.

- Week 5: In the last week, I removed all the circuits from the breadboard and neatly assembled everything into a compact box. The final setup had two knobs, one push button, and the LED matrix placed outside the box, giving it a clean and portable look like a mini arcade game.

Program Code:

```
const int buzzer = 1;
int player1 = A0, player2 = A1;
const int init_speed = 600;
int speed = init_speed;
int rowspin[8] = {13, A4, 9, 10, 2, 8, 3, 6};
int colspin[8] = {A5, 4, 5, 12, 7, 11, A3, A2};
int rows[14] = {13, A4, 9, 10, 2, 8, 3, 6, 3, 8, 2, 10, 9, A4};
int cols[12] = {A5, 4, 5, 12, 7, 11, A3, 11, 7, 12, 5, A2};
int s = 0, r = 3, c = 2;

void setup() {
  // put your setup code here, to run once:
  digitalWrite(1,0);
}
void movingBase(int feedback, int colspin[], int col_moving){
  digitalWrite(colspin[col_moving], 1);
    for(int i = 0; i<=2; i++){
      pinMode(rowspin[feedback], 1);
      pinMode(rowspin[feedback+1], 1);
      pinMode(rowspin[feedback+2], 1);
      //delay(1);
      pinMode(rowspin[feedback], 0);
      pinMode(rowspin[feedback+1], 0);
      pinMode(rowspin[feedback+2], 0);
    }
    digitalWrite(colspin[col_moving], 0);
}
void ending_anime(int wait  = 10){
  for(int k = 0; k <=1; k++){
    for(int j = 0; j <= 2; j++){
      for(int n = 0; n <= wait; n++){
        pinMode(rowspin[3-j], 1);
        for(int i = 0; i < (j+1)*2; i++){
          pinMode(colspin[3-j+i], 1);
          digitalWrite(colspin[3-j+i], 1);
          delay(1);
          digitalWrite(colspin[3-j+i], 0);
          pinMode(colspin[3-j+i], 0);
        }
        pinMode(rowspin[3-j], 0);
```

```
        pinMode(rowspin[4+j], 1);
        for(int i = 0; i < (j+1)*2; i++){
          pinMode(colspin[3-j+i], 1);
          digitalWrite(colspin[3-j+i], 1);
          delay(1);
          digitalWrite(colspin[3-j+i], 0);
          pinMode(colspin[3-j+i], 0);
        }
        pinMode(rowspin[4+j], 0);

        pinMode(colspin[3-j], 1);
        digitalWrite(colspin[3-j], 1);
        for(int i = 1; i < (j+1)*2-1; i++){
          pinMode(rowspin[3-j+i], 1);
          delay(1);
          pinMode(rowspin[3-j+i], 0);
        }
        digitalWrite(colspin[3-j], 0);
        pinMode(colspin[3-j], 0);

        pinMode(colspin[4+j], 1);
        digitalWrite(colspin[4+j], 1);
        for(int i = 1; i < (j+1)*2-1; i++){
          pinMode(rowspin[3-j+i], 1);
          delay(1);
          pinMode(rowspin[3-j+i], 0);
        }
        digitalWrite(colspin[4+j], 0);
        pinMode(colspin[4+j], 0);
      }
    }
  }
}
void show_ball(int rows[], int cols[], int r, int c){
  pinMode(rows[r], 1);
  pinMode(cols[c], 1);
  digitalWrite(cols[c], 1);
  delay(1);
  digitalWrite(cols[c], 0);
  pinMode(cols[c], 0);
  pinMode(rows[r], 0);
}
```

```
void showP(int colspin[], int rowspin[], int wait){
  for(int i = 0; i <= wait; i++){
    pinMode(rowspin[1], 1);
    for(int i = 1; i <= 5; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
      delay(1);
      digitalWrite(colspin[i], 0);
      pinMode(colspin[i], 0);
    }
    pinMode(rowspin[1], 0);

    pinMode(rowspin[6], 1);
    for(int i = 1; i <= 3; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
      delay(1);
      digitalWrite(colspin[i], 0);
      pinMode(colspin[i], 0);
    }
    pinMode(rowspin[6], 0);

    pinMode(rowspin[3], 1);
    for(int i = 3; i <= 4; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
      delay(1);
      digitalWrite(colspin[i], 0);
      pinMode(colspin[i], 0);
    }
    pinMode(rowspin[3], 0);

    pinMode(colspin[2], 1);
    digitalWrite(colspin[2], 1);
    for(int i = 2; i <= 5; i++){
      pinMode(rowspin[i], 1);
      delay(1);
      pinMode(rowspin[i], 0);
    }
    digitalWrite(colspin[2], 0);
    pinMode(colspin[2], 0);
```

```
    pinMode(colspin[5], 1);
    digitalWrite(colspin[5], 1);
    for(int i = 2; i <= 3; i++){
      pinMode(rowspin[i], 1);
      delay(1);
      pinMode(rowspin[i], 0);
    }
    digitalWrite(colspin[5], 0);
    pinMode(colspin[5], 0);
  }
}
void show1(int colspin[], int rowspin[], int wait){
  for(int i = 0; i <= wait; i++){
    pinMode(rowspin[6], 1);
    for(int i = 2; i <= 5; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
      delay(1);
      digitalWrite(colspin[i], 0);
      pinMode(colspin[i], 0);
    }
    pinMode(rowspin[6], 0);

    pinMode(rowspin[2], 1);
    for(int i = 2; i <= 2; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
      delay(1);
      digitalWrite(colspin[i], 0);
      pinMode(colspin[i], 0);
    }
    pinMode(rowspin[2], 0);

    pinMode(colspin[3], 1);
    pinMode(colspin[4], 1);
    digitalWrite(colspin[3], 1);
    digitalWrite(colspin[4], 1);
    for(int i = 1; i <= 6; i++){
      pinMode(rowspin[i], 1);
      delay(1);
      pinMode(rowspin[i], 0);
    }
```

```
    digitalWrite(colspin[3], 0);
    digitalWrite(colspin[4], 0);
    pinMode(colspin[3], 0);
    pinMode(colspin[4], 0);
  }
}
void showW(int colspin[], int rowspin[], int wait){
  for(int i = 0; i <= wait; i++){
    pinMode(rowspin[1], 1);
    for(int i = 0; i <= 2; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
      delay(1);
      digitalWrite(colspin[i], 0);
      pinMode(colspin[i], 0);
    }
    pinMode(rowspin[1], 0);

    pinMode(rowspin[1], 1);
    for(int i = 5; i <= 7; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
      delay(1);
      digitalWrite(colspin[i], 0);
      pinMode(colspin[i], 0);
    }
    pinMode(rowspin[1], 0);

    pinMode(rowspin[4], 1);
    for(int i = 3; i <= 4; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
      delay(1);
      digitalWrite(colspin[i], 0);
      pinMode(colspin[i], 0);
    }
    pinMode(rowspin[4], 0);

    pinMode(colspin[1], 1);
    digitalWrite(colspin[1], 1);
    for(int i = 2; i <= 6; i++){
      pinMode(rowspin[i], 1);
```

```
      delay(1);
      pinMode(rowspin[i], 0);
    }
    digitalWrite(colspin[1], 0);
    pinMode(colspin[1], 0);

    pinMode(colspin[6], 1);
    digitalWrite(colspin[6], 1);
    for(int i = 2; i <= 6; i++){
      pinMode(rowspin[i], 1);
      delay(1);
      pinMode(rowspin[i], 0);
    }
    digitalWrite(colspin[6], 0);
    pinMode(colspin[6], 0);

    pinMode(colspin[2], 1);
    digitalWrite(colspin[2], 1);
    for(int i = 5; i <= 5; i++){
      pinMode(rowspin[i], 1);
      delay(1);
      pinMode(rowspin[i], 0);
    }
    digitalWrite(colspin[2], 0);
    pinMode(colspin[2], 0);

    pinMode(colspin[5], 1);
    digitalWrite(colspin[5], 1);
    for(int i = 5; i <= 5; i++){
      pinMode(rowspin[i], 1);
      delay(1);
      pinMode(rowspin[i], 0);
    }
    digitalWrite(colspin[5], 0);
    pinMode(colspin[5], 0);
  }
}
void show2(int colspin[], int rowspin[], int wait){
  for(int i = 0; i <= wait; i++){
    pinMode(rowspin[1], 1);
    for(int i = 3; i <= 4; i++){
      pinMode(colspin[i], 1);
```

```
  digitalWrite(colspin[i], 1);
  delay(1);
  digitalWrite(colspin[i], 0);
  pinMode(colspin[i], 0);
}
pinMode(rowspin[1], 0);

pinMode(rowspin[6], 1);
for(int i = 2; i <= 5; i++){
  pinMode(colspin[i], 1);
  digitalWrite(colspin[i], 1);
  delay(1);
  digitalWrite(colspin[i], 0);
  pinMode(colspin[i], 0);
}
pinMode(rowspin[6], 0);

pinMode(rowspin[2], 1);
for(int i = 2; i <= 2; i++){
  pinMode(colspin[i], 1);
  digitalWrite(colspin[i], 1);
  delay(1);
  digitalWrite(colspin[i], 0);
  pinMode(colspin[i], 0);
}
pinMode(rowspin[3], 0);

pinMode(colspin[5], 1);
digitalWrite(colspin[5], 1);
for(int i = 2; i <= 3; i++){
  pinMode(rowspin[i], 1);
  delay(1);
  pinMode(rowspin[i], 0);
}
digitalWrite(colspin[5], 0);
pinMode(colspin[5], 0);

pinMode(colspin[4], 1);
digitalWrite(colspin[4], 1);
for(int i = 4; i <= 4; i++){
  pinMode(rowspin[i], 1);
  delay(1);
```

```
      pinMode(rowspin[i], 0);
    }
    digitalWrite(colspin[4], 0);
    pinMode(colspin[4], 0);

    pinMode(colspin[3], 1);
    digitalWrite(colspin[3], 1);
    for(int i = 5; i <= 5; i++){
      pinMode(rowspin[i], 1);
      delay(1);
      pinMode(rowspin[i], 0);
    }
    digitalWrite(colspin[3], 0);
    pinMode(colspin[3], 0);
  }
}
void show3(int colspin[], int rowspin[], int wait){
  for(int i = 0; i <= wait; i++){
    pinMode(rowspin[1], 1);
    for(int i = 3; i <= 4; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
      delay(1);
      digitalWrite(colspin[i], 0);
      pinMode(colspin[i], 0);
    }
    pinMode(rowspin[1], 0);

    pinMode(rowspin[2], 1);
    for(int i = 2; i <= 2; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
      delay(1);
      digitalWrite(colspin[i], 0);
      pinMode(colspin[i], 0);
    }
    pinMode(rowspin[2], 0);

    pinMode(rowspin[4], 1);
    for(int i = 3; i <= 4; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
```

```
  delay(1);
  digitalWrite(colspin[i], 0);
  pinMode(colspin[i], 0);
}
pinMode(rowspin[4], 0);

pinMode(rowspin[6], 1);
for(int i = 2; i <= 2; i++){
  pinMode(colspin[i], 1);
  digitalWrite(colspin[i], 1);
  delay(1);
  digitalWrite(colspin[i], 0);
  pinMode(colspin[i], 0);
}
pinMode(rowspin[6], 0);

pinMode(rowspin[7], 1);
for(int i = 3; i <= 4; i++){
  pinMode(colspin[i], 1);
  digitalWrite(colspin[i], 1);
  delay(1);
  digitalWrite(colspin[i], 0);
  pinMode(colspin[i], 0);
}
pinMode(rowspin[7], 0);

pinMode(colspin[5], 1);
digitalWrite(colspin[5], 1);
for(int i = 2; i <= 3; i++){
  pinMode(rowspin[i], 1);
  delay(1);
  pinMode(rowspin[i], 0);
}
digitalWrite(colspin[5], 0);
pinMode(colspin[5], 0);

pinMode(colspin[5], 1);
digitalWrite(colspin[5], 1);
for(int i = 5; i <= 6; i++){
  pinMode(rowspin[i], 1);
  delay(1);
  pinMode(rowspin[i], 0);
```

```
    }
    digitalWrite(colspin[5], 0);
    pinMode(colspin[5], 0);
  }
}
void showGO(int colspin[], int rowspin[], int wait){
  for(int i = 0; i <= wait; i++){
    pinMode(rowspin[2], 1);
    for(int i = 1; i <= 3; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
      delay(1);
      digitalWrite(colspin[i], 0);
      pinMode(colspin[i], 0);
    }
    pinMode(rowspin[2], 0);

    pinMode(rowspin[2], 1);
    for(int i = 6; i <= 6; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
      delay(1);
      digitalWrite(colspin[i], 0);
      pinMode(colspin[i], 0);
    }
    pinMode(rowspin[2], 0);

    pinMode(rowspin[4], 1);
    for(int i = 2; i <= 3; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
      delay(1);
      digitalWrite(colspin[i], 0);
      pinMode(colspin[i], 0);
    }
    pinMode(rowspin[4], 0);

    pinMode(rowspin[5], 1);
    for(int i = 1; i <= 3; i++){
      pinMode(colspin[i], 1);
      digitalWrite(colspin[i], 1);
      delay(1);
```

```
  digitalWrite(colspin[i], 0);
  pinMode(colspin[i], 0);
}
pinMode(rowspin[5], 0);

pinMode(rowspin[5], 1);
for(int i = 6; i <= 6; i++){
  pinMode(colspin[i], 1);
  digitalWrite(colspin[i], 1);
  delay(1);
  digitalWrite(colspin[i], 0);
  pinMode(colspin[i], 0);
}
pinMode(rowspin[5], 0);

pinMode(colspin[5], 1);
digitalWrite(colspin[5], 1);
for(int i = 3; i <= 4; i++){
  pinMode(rowspin[i], 1);
  delay(1);
  pinMode(rowspin[i], 0);
}
digitalWrite(colspin[5], 0);
pinMode(colspin[5], 0);

pinMode(colspin[7], 1);
digitalWrite(colspin[7], 1);
for(int i = 3; i <= 4; i++){
  pinMode(rowspin[i], 1);
  delay(1);
  pinMode(rowspin[i], 0);
}
digitalWrite(colspin[7], 0);
pinMode(colspin[7], 0);

pinMode(colspin[0], 1);
digitalWrite(colspin[0], 1);
for(int i = 3; i <= 4; i++){
  pinMode(rowspin[i], 1);
  delay(1);
  pinMode(rowspin[i], 0);
}
```

```
    digitalWrite(colspin[0], 0);
    pinMode(colspin[0], 0);
  }
}

int enable = 0;
void loop() {
  float start = digitalRead(0);
  if(start == 0){
    enable = 1;
    delay(300);
  }
  if(enable==1){
    if(s==0){
      show1(colspin, rowspin, 50);
      show2(colspin, rowspin, 50);
      show3(colspin, rowspin, 50);
      showGO(colspin, rowspin, 40);
    }
    float output1 = analogRead(player1);
    int feedback1 = output1/170.6;
    float output2 = analogRead(player2);
    int feedback2 = output2/170.6;

    pinMode(colspin[0], 1);
    pinMode(colspin[7], 1);

    // Player 1 base:
    movingBase(feedback1, colspin, 0);
    // Player 2 base:
    movingBase(feedback2, colspin, 7);
    // Ball Code:
    s++;
    if(s%(speed/20)==0){
      show_ball(rows, cols, r, c);
    }
    if(s%speed==0){
      s = 1;
      show_ball(rows, cols, r, c);
      if(c==1){
        if(speed > 150){
          speed = speed - 20;
```

```
        }
        if((rows[r] != rows[feedback1])&&(rows[r] != rows[feedback1+1])&&(rows[r] !=
rows[feedback1+2])){
          enable = 0;
          s = 0;
          speed = init_speed;
          ending_anime();
          showP(colspin, rowspin, 30);
          show2(colspin, rowspin, 45);
          showW(colspin, rowspin, 20);
          delay(500);
        }
      }
      if(c==6){
        if(speed > 160){
          speed = speed - 20;
        }
        if((rows[r] != rows[feedback2])&&(rows[r] != rows[feedback2+1])&&(rows[r] !=
rows[feedback2+2])){
          enable = 0;
          s = 0;
          speed = init_speed;
          ending_anime();
          showP(colspin, rowspin, 30);
          show1(colspin, rowspin, 45);
          showW(colspin, rowspin, 20);
          delay(500);
        }
      }
      c++;
      if(c%11==0){
        c = 1;
      }
      r++;
      if(r%14==0){
        r = 0;
      }
    }
  }
}
```