```
#include <LedControl.h>

const int BUTTON_PLAYER1_R = 5;
const int BUTTON_PLAYER1_L = 4;
const int BUTTON_PLAYER2_R = 3;
const int BUTTON_PLAYER2_L = 2;

const int Speaker = 12;

int STRUCT_NUMBER[16] = {1, 2, 8, 11, 16, 19, 24, 25, 26,
27, 32, 35, 40, 43, 49, 50};
int PATTERN_NUMBER[10][16] = {
  {1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1},
  {0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0},
  {1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1},
  {1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1},
  {0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0},
  {1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1},
  {1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1},
  {1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0},
  {1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1},
  {1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1},
};

const int WIDTH_GAME = 8;
const int HEIGHT_GAME = 32;

LedControl led_control = LedControl(11, 13, 10, 4); //
DIN, CLK, CS, Modules (pin)

void ledMatrixSetup()
{
  int device_count = led_control.getDeviceCount();
  for (int addr = 0; addr < device_count; addr++)
  {
    led_control.shutdown(addr, 0);
    led_control.setIntensity(addr, 1);
    led_control.clearDisplay(addr);
  }
}

bool button_pressed[4] = {0, 0, 0, 0};

void buttonSetup()
{
  pinMode(BUTTON_PLAYER1_R, INPUT_PULLUP);
  pinMode(BUTTON_PLAYER1_L, INPUT_PULLUP);
  pinMode(BUTTON_PLAYER2_R, INPUT_PULLUP);
  pinMode(BUTTON_PLAYER2_L, INPUT_PULLUP);
}

bool ButtonPressed(int pin)
{
  delay(10);
  if (digitalRead(pin) == LOW)
    return 1;
  return 0;
}

void speakerSetup()
{
  pinMode(Speaker, OUTPUT);
}

void playTone(int frequency, int delay_amount)
{
  tone(Speaker, frequency);
  delay(delay_amount);
  noTone(Speaker);
}
```

```
void resetDisplay() // Turn off all LED on dot matrix
{
  for (int n = 0; n < 4; n++)
  {
    for (int i = 0; i < 8; i++)
    {
      for (int j = 0; j < 8; j++)
      {
        led_control.setLed(n, i, j, 0);
      }
    }
  }
}

void drawPoint(int x, int y) // Turn On Led on position x,
y
{
  for (int n = 0; n < 4; n++)
  {
    for (int i = 0; i < 8; i++)
    {
      for (int j = 0; j < 8; j++)
      {
        if (i == x && j + (8 * n) == y)
        {
          led_control.setLed((3 - n), 7 - i, j, 1);
        }
      }
    }
  }
}

void deletePoint(int x, int y) // Turn Off LED on
position x,y
```

```
{
  for (int n = 0; n < 4; n++)
  {
    for (int i = 0; i < 8; i++)
    {
      for (int j = 0; j < 8; j++)
      {
        if (i == x && j + (8 * n) == y)
        {
          led_control.setLed((3 - n), 7 - i, j, 0);
        }
      }
    }
  }
}

void clearNumber(int x, int y) //Clear number on dot
matrix
{
  for (int i = 0; i < 16; i++)
  {
    int offset_y = STRUCT_NUMBER[i] / 8;
    int offset_x = STRUCT_NUMBER[i] - (offset_y * 8);
    deletePoint(x + offset_x, y + offset_y);
  }
}

void showNumber(int x, int y, int number) // Show number
on dot matrix
{
  for (int i = 0; i < 16; i++)
  {
    int offset_y = STRUCT_NUMBER[i] / 8;
    int offset_x = STRUCT_NUMBER[i] - (offset_y * 8);
    if (PATTERN_NUMBER[number][i])
    {
      drawPoint(x + offset_x, y + offset_y);
    }
    else
    {
      deletePoint(x + offset_x, y + offset_y);
    }
  }
}

void blinkScore(int score_1, int score_2, int winner)
//Blink winner score, player_score1, player_score2
{
  int player_1_x = 2, player_1_y = 7;
  showNumber(player_1_x, player_1_y, score_1);

  int player_2_x = 2, player_2_y = 18;
  showNumber(player_2_x, player_2_y, score_2);

  delay(800);

  if (winner == 1)
  {
    score_1++;
    clearNumber(player_1_x, player_1_y);
    delay(400);
    showNumber(player_1_x, player_1_y, score_1);
    playTone(659, 100);
    playTone(784, 100);
    playTone(1319, 100);
    playTone(1047, 100);
    clearNumber(player_1_x, player_1_y);
    playTone(1175, 100);
    playTone(1568, 100);
    delay(200);
    showNumber(player_1_x, player_1_y, score_1);
    delay(800);
  }
  else
  {
    score_2++;
    clearNumber(player_2_x, player_2_y);
    delay(400);
    showNumber(player_2_x, player_2_y, score_2);
    playTone(659, 100);
    playTone(784, 100);
    playTone(1319, 100);
    playTone(1047, 100);
    clearNumber(player_2_x, player_2_y);
    playTone(1175, 100);
    playTone(1568, 100);
    delay(200);
    showNumber(player_2_x, player_2_y, score_2);
    delay(800);
  }

  clearNumber(player_1_x, player_1_y);
  clearNumber(player_2_x, player_2_y);
  delay(600);
}

class Paddle
{
  public:
    int x;
    int y;
    int direction;
    int size;

    Paddle(int x, int y) // Control of the Paddle Object
```

```
on position x,y
    {
      this->x = x;
      this->y = y;
      this->direction = 0;
      this->size = 4;
    }

    void setDirection(int direction) // Set Paddle
control on the X axis
    {
      this->direction = direction;
    }

    void draw() // Draw Paddle Object
    {
      if (this->direction != 0)
      {
        for (int i = 0; i < this->size; i++)
        {
          deletePoint(this->x + i, this->y);
        }
        this->x += this->direction;
        if (this->x > 8 - this->size)
          this->x = 8 - this->size;
        if (this->x < 0) this->x = 0;
      }

      for (int i = 0; i < this->size; i++)
      {
        drawPoint(this->x + i, this->y);
      }
    }
};
```

```
class Puck
{
  public:
    float x;
    float y;
    float x_previous;
    float y_previous;
    float x_speed;
    float y_speed;

    Puck(float x, float y) // Control of the Puck Object
on position x,y
    {
      this->x = x;
      this->y = y;
      this->x_previous = x;
      this->y_previous = y;
      this->x_speed = 1;
      this->y_speed = -1;
    }

    void update() // Puck Update
    {
      if (this->x + this->x_speed >= WIDTH_GAME ||
this->x + this->x_speed < 0)
      {
        this->x_speed *= -1;
        playTone(123, 15);
        playTone(62, 15);
      }

      this->x += this->x_speed;
      this->y += this->y_speed;
    }
    int getWinner() // If Puck got out of Game = Check
    {
      if (this->y + this->y_speed >= HEIGHT_GAME)
        return 1;
      if (this->y + this->y_speed < 0)
        return 2;
      return 0;
    }

    bool collisionCheck(Paddle paddle, int direction) //
If Puck hits Paddle = Check
    {
      if (direction < 0)
      {
        if (this->y < paddle.y + 2 && this->y > paddle.y
&& this->x >= paddle.x - 1 && this->x < paddle.x + paddle.
size + 0.5)
        {
          this->y_speed *= -1;
          return 1;
        }
      }
      else if (direction > 0)
      {
        if (this->y > paddle.y - 2 && this->y < paddle.y
&& this->x >= paddle.x - 1 && this->x < paddle.x + paddle.
size + 0.5)
        {
          this->y_speed *= -1;
          return 1;
        }
      }
      return 0;
    }
```

```
void blink() // Blink when Puck out of Game
{
  drawPoint(this->x, this->y);
  playTone(196, 100);
  playTone(147, 100);
  deletePoint(this->x, this->y);
  playTone(98, 100);
  delay(100);
  drawPoint(this->x, this->y);
  delay(200);
  deletePoint(this->x, this->y);
  delay(200);
  drawPoint(this->x, this->y);
  delay(200);
  deletePoint(this->x, this->y);
  delay(800);
}

void reset() // Puck Reset
{
  this->x = WIDTH_GAME / 2;
  this->y = HEIGHT_GAME / 2;
  draw();
  playTone(494, 100);
  playTone(659, 100);

  deletePoint(this->x, this->y);
  delay(200);
  drawPoint(this->x, this->y);
  delay(200);
  deletePoint(this->x, this->y);
  delay(200);
  drawPoint(this->x, this->y);
  delay(400);
}
```

```
    void draw() // Draw Puck
    {
      if (round(this->x) != round(this->x_previous)
          || round(this->y) != round(this->y_previous))
       {
        deletePoint(this->x_previous, this->y_previous);
         this->x_previous = this->x;
         this->y_previous = this->y;
         drawPoint(this->x, this->y);
       }
    }
};

enum GameState
{
  GAME_WAITING,
  GAME_RUNNING,
  GAME_ENDING,
};

int game_state = GAME_RUNNING;

Paddle player_1(2, 2); //Load Player's Paddles
Paddle player_2(2, 29);
int player_1_score = 0;
int player_2_score = 0;

Puck puck(WIDTH_GAME / 2, HEIGHT_GAME / 2); //Load Puck

void gameSetup()
{


}
```

```
void gameWaiting()
{
  if (game_state != GAME_WAITING)
    return;
}
void gameRunning()
{
  if (game_state != GAME_RUNNING)
    return;

  if (puck.getWinner() == 1) // Player1 wins
  {
    puck.blink();
   blinkScore(player_1_score, player_2_score, 1);
    player_1_score++;
    puck.reset();
  }
  else if (puck.getWinner() == 2) // Player2 wins
  {
    puck.blink();
   blinkScore(player_1_score, player_2_score, 2);
    player_2_score++;
    puck.reset();
  }

  player_1.setDirection(0); // Player1's Paddle Control
  if (ButtonPressed(BUTTON_PLAYER1_R))
  {
   player_1.setDirection(-1);
  }
  if (ButtonPressed(BUTTON_PLAYER1_L))
  {
   player_1.setDirection(1);
  }
  player_2.setDirection(0); // Player2's Paddle Control
  if (ButtonPressed(BUTTON_PLAYER2_R))
  {
    player_2.setDirection(-1);
  }
  if (ButtonPressed(BUTTON_PLAYER2_L))
  {
    player_2.setDirection(1);
  }

  if (puck.collisionCheck(player_1, -1) // Update Puck
      || puck.collisionCheck(player_2, 1))
  {
    playTone(1568, 20);
    playTone(784, 20);
  }
  puck.update(); // Draw 2 Player's Paddle

  player_1.draw();
  player_2.draw();

  puck.draw(); // Puck Draw
}

void gameEnding()
{
  if (game_state != GAME_ENDING)
    return;
}

void setup()
{
  Serial.begin(9600);
  Serial.println();
  ledMatrixSetup();
  speakerSetup();
  buttonSetup();
  gameSetup();
}

void loop()
{
  gameWaiting();
  gameRunning();
  gameEnding();
}
```