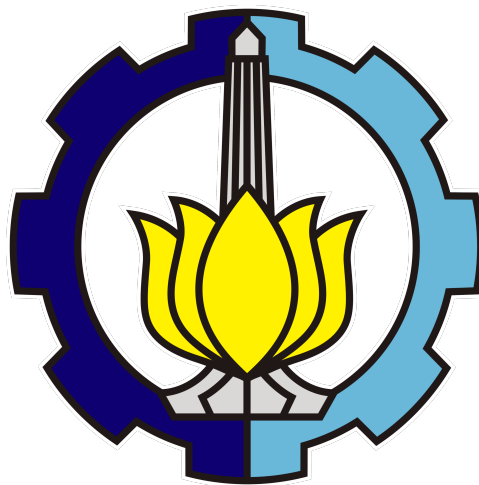


# Laporan Final *Game PingPong* pada *Dotmatrix Display*



**Dosen Pengampu**

Eko Pramunanto, S.T. M.T.

**Disusun Oleh:**

Muhammad Haekal Muhyidin Al-Araby

5024221004

Sistem Tertanam - A

**DEPARTEMEN TEKNIK KOMPUTER  
FAKULTAS TEKNOLOGI ELEKTRO DAN INFORMATIKA CERDAS  
INSTITUT TEKNOLOGI SEPULUH NOPEMBER  
2024**

# 1 Komponen

1. ESP32  
ESP32 sebagai mikrokontroller yang digunakan untuk mengendalikan dotmatrix melalui program yang telah dipasang atau flash.
2. MAX7219 8x32 LED Dot Matrix Display Module  
Modul terdiri dari 4 buah dot matrix dan IC MAX7219 yang digunakan untuk sebagai penghubung antara ESP32 dan display dotmatrix. Bertindak sebagai decoder dan selector.
3. PCB  
Sebagai tempat untuk merangkai barang yang ada dan menyambungkannya.
4. Pin Header Female  
Untuk menghubungkan ESP32 dengan PCB.
5. Pin Header Male Siku  
Menghubungkan display ke PCB
6. Push Button  
untuk input smash
7. Slide Potentiometer  
input gerakan player

## 2 Desain Sistem

### 2.1 Rangkaian Skematik

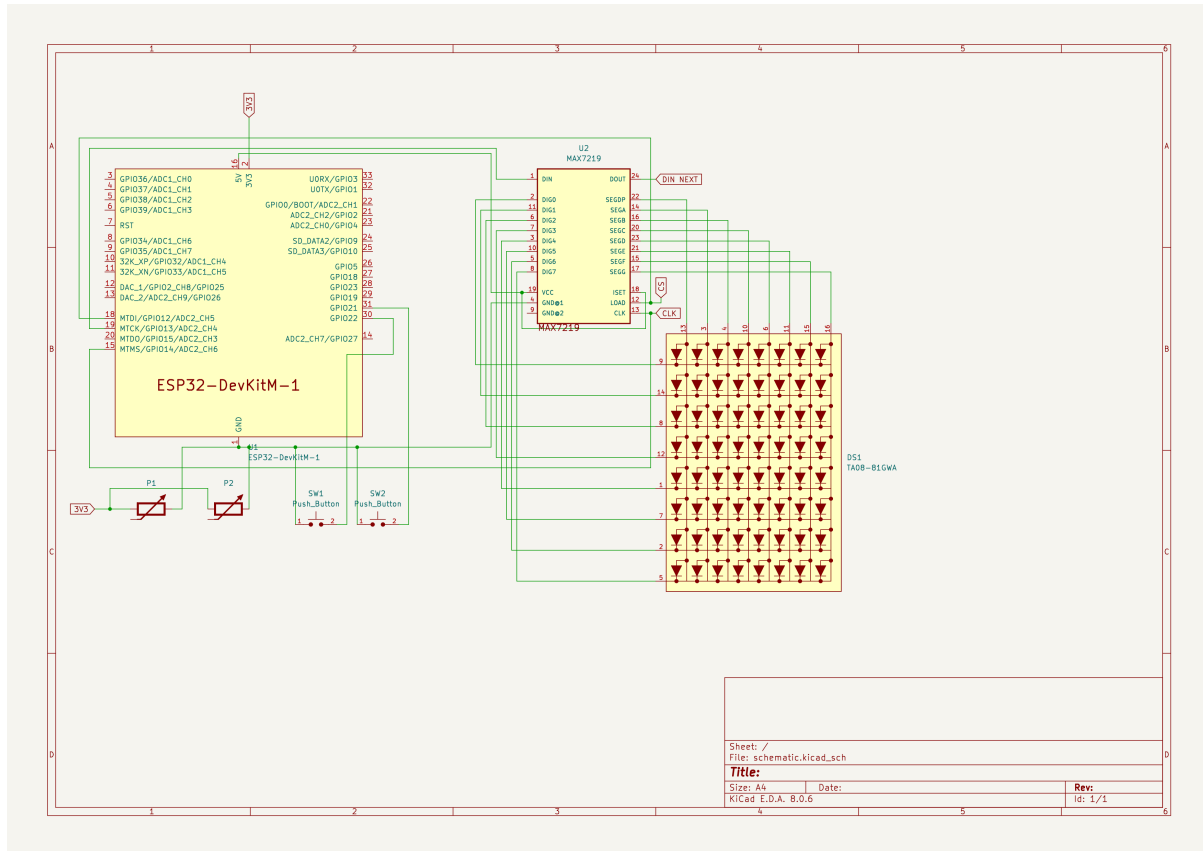


Figure 1: Schematic

Rangkaian menggunakan ESP32 yang terhubung pada IC MAX7219 dimana CS terhubung pada pin D12, DIN pada D13 dan clock pada D14 terdapat 4 IC MAX7219 yang saling terhubung secara seri. Lalu push button dihubungkan dengan ground dan GPIO dan potentiometer dihubungkan ke ground, vcc, dan GPIO.

### 2.2 Desain Game

1. Game memiliki 2 player yang dapat digerakkan dengan slide potentiometer
2. Bola akan memantul saat menabrak pembatas ataupun player
3. Bola memantul secara diagonal dengan kecepatan konstan
4. Bila player smash dan mengenai bola, bola akan bergerak lurus dan lebih cepat
5. Bila bola melewati player maka player lainnya akan mendapatkan skor
6. Player yang kebobolan akan mendapatkan bola dan dapat menggerakkannya dan menekan tombol smash untuk service.

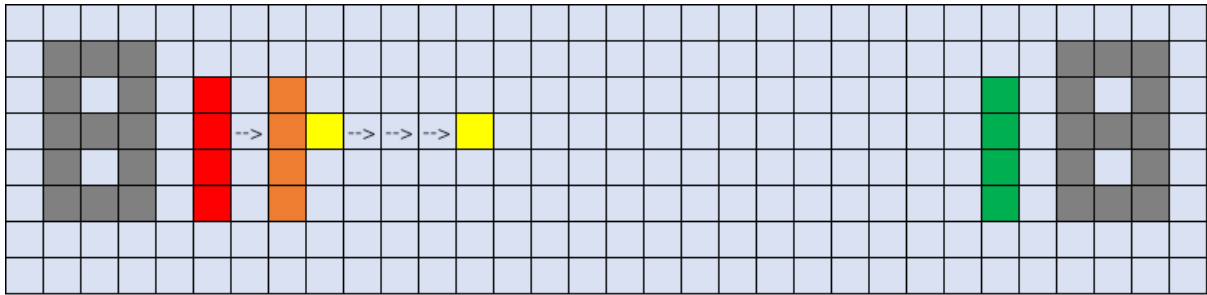


Figure 2: Desain Game

7. Game akan berakhir bila salah satu pemain mencapai skor 11 dan selisih 2 poin.
8. Game akan berakhirimbang ketika kedua pemain mencapai 15 poin

## 2.3 Program

### 2.3.1 Menampilkan

Untuk menampilkan ke dotmatrix dapat digunakan 2 library MD\_Parola dan MD\_MAX72xx. Dimana MD\_Parola dapat menampilkan huruf namun kurang dalam kontrol sehingga akan kesulitan untuk menampilkan Game. Sehingga MD\_Parola hanya akan digunakan saat menampilkan string. Sedangkan saat ingin menampilkan karakter akan menggunakan MD\_MAX72xx yang memberikan raw control pada dotmatrix display. Untuk itu akan dibuat sebuah fungsi yang menerima input berbentuk array 8\*32 untuk ditampilkan pada dotmatrix. Berikutnya akan disebut frame. Lalu tiap objek akan diupdate dengan diletakan pada frame tersebut. Sesuai dengan posisi.

### 2.3.2 Input

Input didapatkan dengan mengambil nilai analogRead pada potentiometer dan digitalRead pada button smash.

## 2.4 Source Code

Berisi program utama yang mengatur main loop.

```
#include <Arduino.h>
#include <cstdlib>
#include <ctime>

#include "MD_Parola.h"
#include "animation.hpp"
#include "device.hpp"
#include "esp32-hal-adc.h"
#include "esp32-hal-gpio.h"
#include "global.hpp"
#include "score.hpp"

void setup() {
    Setup();
}
```

```
Intro();
analogReadResolution(3);
display.displayScroll("PRESS ANY BUTTON TO START...", PA_CENTER,
PA_SCROLL_LEFT, 100);
while (!digitalRead(p1.smashInput) && !digitalRead(p2.smashInput))
{
    if (display.displayAnimate()) {
        display.displayReset();
    }
}
}

int last = 0;

void loop() {
    int current = millis();

    b.Update();
    p1.CheckInput();
    p2.CheckInput();

    if (current - last < 50) {
        return;
    }

    last = current;

    native_display.clear();
    p1.Draw();
    p2.Draw();
    b.Draw();

    RenderScore(p2.score, 9);
    RenderScore(p1.score, -16);

    if (p1.score >= 11 && p1.score - p2.score >= 2) {
        native_display.clear();
        display.displayText("P1 WIN", PA_CENTER, 50, 1000,
PA_SCROLL_RIGHT, PA_SCROLL_RIGHT);
        while (!digitalRead(p1.smashInput)&& !digitalRead(p2.smashInput
)) {
            if (display.displayAnimate()) {
                display.displayReset();
            }
        }
        p1.score = 0;
        p2.score = 0;
    }
    else if (p2.score >= 11 && p2.score - p1.score >= 2) {
        native_display.clear();
        display.displayText("P2 WIN", PA_CENTER, 50, 1000,
PA_SCROLL_LEFT, PA_SCROLL_LEFT);
        while (!digitalRead(p1.smashInput)&& !digitalRead(p2.smashInput
)) {
            if (display.displayAnimate()) {
                display.displayReset();
            }
        }
    }
}
```

```

        p1.score = 0;
        p2.score = 0;
    } else if (p2.score >= 15 && p1.score >= 15) {
        native_display.clear();
        display.displayText("TIE", PA_CENTER, 50, 1000, PA_SCROLL_UP,
PA_SCROLL_UP);
        while (!digitalRead(p1.smashInput)&& !digitalRead(p2.smashInput
)) {
            if (display.displayAnimate()) {
                display.displayReset();
            }
        }
        p1.score = 0;
        p2.score = 0;
    }
}

```

Listing 1: main.cpp

Program di bawah mengendalikan device dotmatrix menggunakan library Parola dan MDMAX

```

#pragma once

#include <MD_Parola.h>
#include <MD_MAX72xx.h>
#include <SPI.h>
#include <stdint>

constexpr MD_MAX72XX::moduleType_t HARDWARE_TYPE = MD_MAX72XX::FC16_HW;
constexpr uint8_t MAX_DEVICES = 4;
constexpr uint8_t CS_PIN = 12;
constexpr uint8_t DIN_PIN = 13;
constexpr uint8_t CLK_PIN = 14;
constexpr uint16_t SPEED = 100;

extern MD_Parola display;
extern MD_MAX72XX native_display;

void Setup();

```

Listing 2: device.hpp

```

#include "device.hpp"
#include "HardwareSerial.h"

MD_Parola display = MD_Parola(HARDWARE_TYPE, DIN_PIN, CLK_PIN, CS_PIN,
MAX_DEVICES);
MD_MAX72XX native_display = MD_MAX72XX(HARDWARE_TYPE, DIN_PIN, CLK_PIN,
CS_PIN, MAX_DEVICES);

void Setup() {
    native_display.begin();
    native_display.clear();
    native_display.control(MD_MAX72XX::INTENSITY, 0);

    display.begin();
    display.setIntensity(0);
    display.displayClear();
}

```

```
Serial.begin(9600);  
}
```

Listing 3: device.cpp

Program di bawah menampilkan untuk menampilkan Intro

```
#pragma once  
  
void Intro();
```

Listing 4: animation.hpp

```
#include "device.hpp"  
#include <cstdio>  
#include "animation.hpp"  
#include "global.hpp"  
  
void Intro() {  
    for (int i = 0; i < 8; i++) {  
        for (int j = 0; j < 8 * MAX_DEVICES; j++) {  
            native_display.setPoint(i, j, HIGH);  
            delay(100);  
            native_display.setPoint(i, j, LOW);  
        }  
        delay(100);  
    }  
    native_display.clear();  
  
    for (int i = 0; i < 8; i++) {  
        for (int j = 0; j < 8 * MAX_DEVICES; j++) {  
            native_display.setPoint(i, j, HIGH);  
            native_display.setPoint(i - 1, j, LOW);  
        }  
        delay(100);  
    }  
    native_display.clear();  
    for (int i = 8; i >= 0; i--) {  
        for (int j = 8 * MAX_DEVICES; j >= 0; j--) {  
            native_display.setPoint(i, j, HIGH);  
        }  
        delay(100);  
    }  
    native_display.clear();  
    for (int j = 0; j < 8 * MAX_DEVICES; j++) {  
        for (int i = 0; i < 8; i++) {  
            native_display.setPoint(i, j, HIGH);  
            native_display.setPoint(i, j - 1, LOW);  
        }  
        delay(100);  
    }  
    native_display.clear();  
    for (int j = 8 * MAX_DEVICES; j >= 0; j--) {  
        for (int i = 8; i >= 0; i--) {  
            native_display.setPoint(i, j, HIGH);  
        }  
        delay(100);  
    }  
}
```

```

    }
    native_display.clear();
    display.displayScroll("Muhammad Haekal Muhyidin Al-Araby 5024221030", PA_CENTER, PA_SCROLL_LEFT, 100);
    while(!display.displayAnimate());
    display.displayScroll("PING PONG", PA_CENTER, PA_SCROLL_RIGHT, 100);
    ;
    while(!display.displayAnimate());
}

```

Listing 5: animation.cpp

Program di bawah Berisi class vector2 untuk memudahkan posisi game object

```

#pragma once

class Vector2 {
public:
    int x = 0, y = 0;
    Vector2(int x = 0, int y = 0): x(x), y(y) {}

    Vector2 operator+(const Vector2& other) const {
        return Vector2(x + other.x, y + other.y);
    }
    Vector2 operator-(const Vector2& other) const {
        return Vector2(x - other.x, y - other.y);
    }
    Vector2 operator*(const Vector2& other) const {
        return Vector2(x * other.x, y * other.y);
    }
    Vector2 operator/(const Vector2& other) const {
        return Vector2(x / other.x, y / other.y);
    }
    Vector2 operator+(const float n) const {
        return Vector2(x + n, y + n);
    }
    Vector2 operator-(const float n) const {
        return Vector2(x - n, y - n);
    }
    Vector2 operator*(const float n) const {
        return Vector2(x * n, y * n);
    }
    Vector2 operator/(const float n) const {
        return Vector2(x / n, y / n);
    }
    Vector2 GlobalToDisplay() const;
    void Draw() const;
    void Clear() const;
};

```

Listing 6: vector2.hpp

```

#include "device.hpp"
#include "esp32-hal-gpio.h"
#include <vector2.hpp>

Vector2 Vector2::GlobalToDisplay() const {
    // (0, 0) -> row 3 column 15
    return *this * -1 + Vector2(15, 3) ;
}

```



```
}

void Vector2::Draw() const {
    Vector2 displayPos = this->GlobalToDisplay();
    native_display.setPoint(displayPos.y, displayPos.x, HIGH);
}

void Vector2::Clear() const {
    Vector2 displayPos = this->GlobalToDisplay();
    native_display.setPoint(displayPos.y, displayPos.x, LOW);
}
```

Listing 7: vector2.cpp

Program di bawah berfungsi mendapatkan input player dan mengupdate state player

```
#pragma once

#include "vector2.hpp"
#include <stdint>

class Player {
public:
    bool service = false;
    Vector2 position;
    Player(uint8_t dirInput, uint8_t smashInput, Vector2 position =
    Vector2());
    void Draw();
    void CheckInput();
    uint16_t score = 0;
    uint8_t power = 1;
    int last = 0;
    int ready = 0;
    uint8_t dirInput;
    uint8_t smashInput;
};
```

Listing 8: player.hpp

```
#include "global.hpp"
#include "esp32-hal-adc.h"
#include "esp32-hal-gpio.h"
#include "esp32-hal.h"
#include "player.hpp"

Player::Player(uint8_t dirInput, uint8_t smashInput, Vector2 position)
: dirInput(dirInput), smashInput(smashInput), position(position) {
    pinMode(smashInput, INPUT_PULLUP);
}

void Player::Draw() {
    this->position.Draw();
    (this->position + Vector2(0, 1)).Draw();
    (this->position + Vector2(0, -1)).Draw();
}

void Player::CheckInput() {
    this->position.y = analogRead(this->dirInput) - 4;
    if (this->position.y > 2) this->position.y = 2;
```

```

else if (this->position.y < -3) this-> position.y = -3;

if (this->service) {
    b.position.y = this->position.y;
}

int current = millis();
if (current - last >= 200 && this-> power != 1) {
    if (this->position.x < 0) this->position.x -= 2;
    if (this->position.x > 0) this->position.x += 2;
    this->power = 1;
}
if (digitalRead(this->smashInput) && current >= ready) {
    this->last = current;
    if (this->position.x < 0) this->position.x += 2;
    if (this->position.x > 0) this->position.x -= 2;
    this->power = 15;
    this->ready = current + 1000;
    if (service) {
        service = false;
        b.move = Vector2(b.position.x - this->position.x, 0);
    }
}
}
// Serial.printf("PlayerPos %d: %d %d\n", (this->dirInput == 35) + 1,
this->position.x, this->position.y);
}

```

Listing 9: player.cpp

Program di bawah untuk mengupdate state bola

```

#pragma once

#include "vector2.hpp"
class Ball {
public:
    Vector2 position;
    Vector2 move;
    Ball(Vector2 position = Vector2());
    void Draw();
    void Update();
    void HitCheck();
    void ScoreCheck();
    int lastUpdate = 0;
    int updateRate = 1;
private:
};

```

Listing 10: ball.hpp

```

#include "ball.hpp"
#include "MD_Parola.h"
#include "global.hpp"
#include "vector2.hpp"
#include "math.h"
#include <cstdlib>
#include "device.hpp"

Ball::Ball(Vector2 position) : position(position) {

```

```
        move = Vector2(-1, 0);
    }

    void Ball::HitCheck() {
        // Serial.printf("P1 X:%d Ball X:%d\n", p1.position.x, this->
        position.x);
        bool ballHitTheWall = this->position.y >= 3 || this->position.y <=
        -4;

        int deltaP1 = this->position.y - p1.position.y;
        int deltaP2 = this->position.y - p2.position.y;

        if (ballHitTheWall) {
            this->move.y *= -1;
        } else if (this->position.x == p1.position.x && abs(deltaP1) < 2 )
        {
            this->move.x *= -1;
            this->move.y = p1.power != 1 ? 0 : deltaP1;
            this->updateRate = p1.power;
        } else if (this->position.x == p2.position.x && abs(deltaP2) < 2) {
            this->move.x *= -1;
            this->move.y = p2.power != 1 ? 0 : deltaP2;
            this->updateRate = p2.power;
        }
    }

    void Ball::ScoreCheck() {
        if (this->position.x < p1.position.x) {
            buzzer.Buzz();
            display.displayText("SCORED", PA_CENTER, 10, 500,
            PA_SCROLL_LEFT, PA_SCROLL_LEFT);
            while(!display.displayAnimate());
            p2.score++;
            this->position = Vector2(-5, p1.position.y);
            this->move = Vector2(0,0);
            p1.service = true;
        } else if (this->position.x > p2.position.x) {
            buzzer.Buzz();
            display.displayText("SCORED", PA_CENTER, 10, 500,
            PA_SCROLL_RIGHT, PA_SCROLL_RIGHT);
            while(!display.displayAnimate());
            p1.score++;
            this->position = Vector2(4, p2.position.y);
            this->move = Vector2(0,0);
            p2.service = true;
        }
    }

    void Ball::Update() {
        int current = millis();
        if (current - this->lastUpdate < 75 / updateRate) {
            return;
        }
        this->lastUpdate = current;
        HitCheck();
        ScoreCheck();
        this->position = this->position + this->move;
    }
}
```

```
}

void Ball::Draw() {
    this->position.Draw();
}
```

Listing 11: ball.cpp

Program di bawah untuk menyalakan buzzer

```
#pragma once

#include <stdint>
class Buzzer {
public:
    Buzzer(uint8_t pin);
    void Buzz();
private:
    uint8_t pin;
};
```

Listing 12: buzzer.hpp

```
#include "buzzer.hpp"
#include "esp32-hal-gpio.h"
#include "esp32-hal.h"
#include <stdint>

Buzzer::Buzzer(uint8_t pin):pin(pin) {
    pinMode(pin, OUTPUT);
}

void Buzzer::Buzz() {
    digitalWrite(this->pin, HIGH);
    delay(100);
    digitalWrite(this->pin, LOW);
}
```

Listing 13: buzzer.cpp

Program di bawah untuk memunculkan skor

```
#pragma once
#include <stdint>

void RenderScore(uint8_t score, int x);
```

Listing 14: score.hpp

```
#include "score.hpp"
#include "HardwareSerial.h"
#include "vector2.hpp"
#include <stdint>
#include <random>

Vector2 numPosition[] = {
    Vector2(0, 0),
    Vector2(1, 0),
    Vector2(2, 0),
    Vector2(0, -1),
}
```

```
    Vector2(2, -1),
    Vector2(0, -2),
    Vector2(1, -2),
    Vector2(2, -2),
    Vector2(0, -3),
    Vector2(2, -3),
    Vector2(0, -4),
    Vector2(1, -4),
    Vector2(2, -4),
};

uint16_t binaryRep[] = {
    0b11111101111111, // 0
    0b1001010010100,  // 1
    0b1110111110111,  // 2
    0b1111011110111,  // 3
    0b1001011111101,  // 4
    0b1111011110111,  // 5
    0b1111111101111,  // 6
    0b1001010010111,  // 7
    0b1111111111111,  // 8
    0b1111011111111   // 9
};

void RenderScore(uint8_t score, int x) {
    Vector2 position = Vector2(x, 2);

    int firstDigit = 0;
    int secondDigit = 0;

    if (score < 10) {
        secondDigit = score;
    } else {
        firstDigit = score / 10;
        secondDigit = score % 10;
    }

    int index = 0;
    for (const auto& p : numPosition) {
        if ((binaryRep[firstDigit] >> index) & 0x1) {
            (position + p).Draw();
        }
        index++;
    }
    position.x += 4;
    index = 0;
    for (const auto& p : numPosition) {
        if ((binaryRep[secondDigit] >> index) & 0x1) {
            (position + p).Draw();
        }
        index++;
    }
}
```

Listing 15: score.cpp

Program di bawah berisi Global Variabel yang digunakan seluruh program

```
#pragma once
```

```
#include "ball.hpp"
#include "buzzer.hpp"
#include "player.hpp"
#include <stdint>

extern Ball b;
extern Player p1;
extern Player p2;
static uint8_t score = 0;
extern Buzzer buzzer;
```

Listing 16: global.hpp

```
#include "global.hpp"
#include "buzzer.hpp"

Ball b = Ball(Vector2(0, 0));
Player p1 = Player(33, 19, Vector2(-8, 0));
Player p2 = Player(35, 22, Vector2(7, 0));
Buzzer buzzer = Buzzer(32);
```

Listing 17: global.cpp

## 2.5 Hasil



Figure 3: Hasil tanpa package