Laporan Progress *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

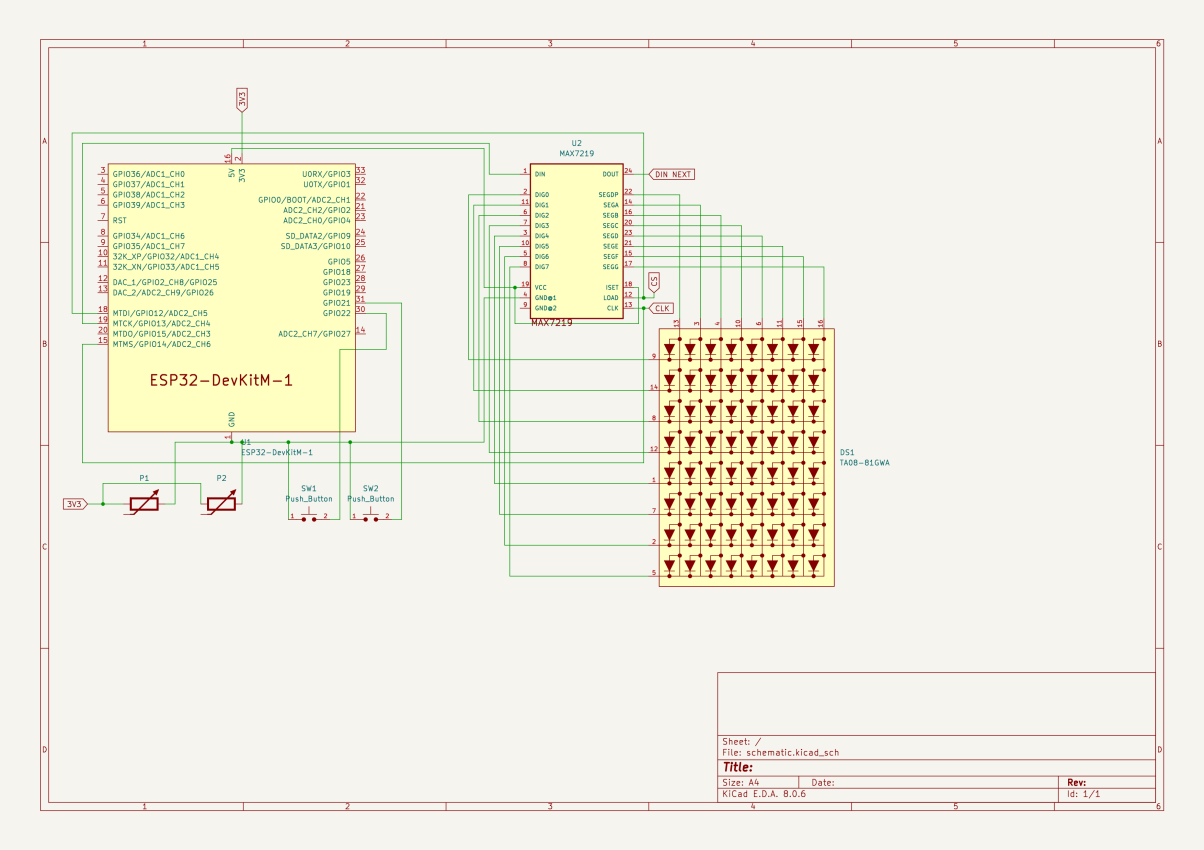

Figure 1: Schematic

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.

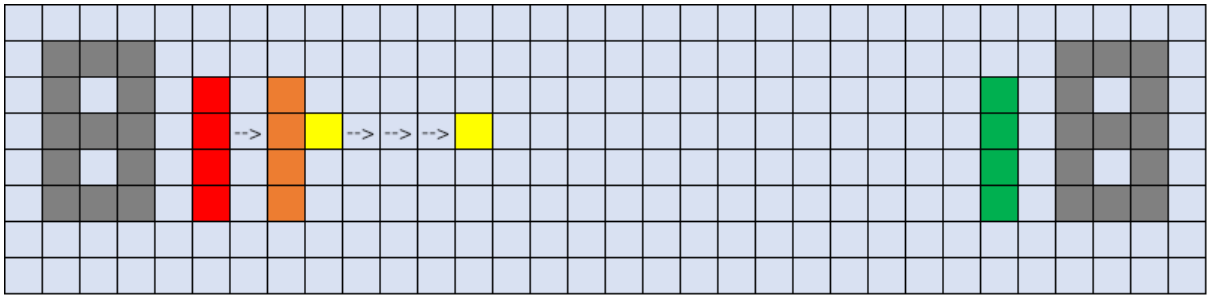

Figure 2: Desain Game

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

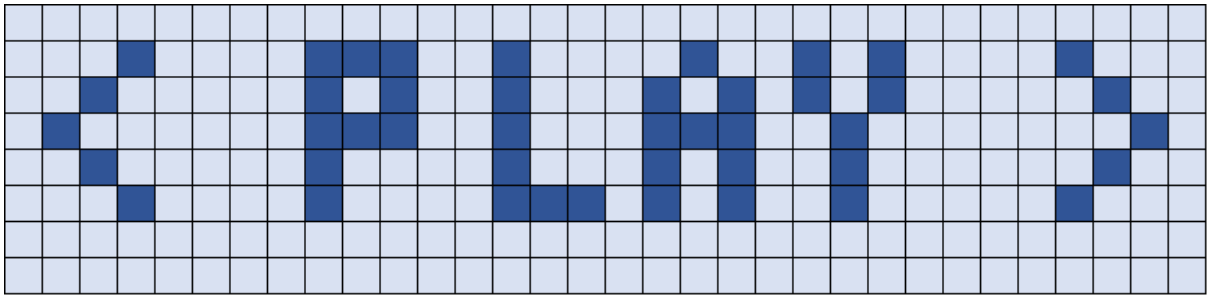

Figure 3: Desain Menu

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

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

#include "ball.hpp"
#include "device.hpp"
#include "esp32-hal.h"
#include "player.hpp"
#include "vector2.hpp"

void setup() {
    Setup();
    // Intro();
    analogReadResolution(12);
    analogSetAttenuation(ADC_0db);
    std::srand(std::time(0));
}

Ball n = Ball(Vector2(0, 0));
Player p1 = Player(0, 0, Vector2(-8, 0));
Player p2 = Player(0, 0, Vector2(7, 0));

void loop() {
    n.Draw();
    p1.Draw();
    p2.Draw();
    delay(100);
    native_display.clear();
    n.position = n.position + Vector2(std::rand()%3-1, std::rand()%3-1)
   ;
    if (abs(n.position.x) > 15 || abs(n.position.y) > 3) {
        n.position = Vector2(0, 0);
    }
    p1.position = p1.position + Vector2(0, std::rand()%3-1);
    if (abs(p1.position.y) > 3) {
        p1.position.y = 0;
    }
    p2.position = p2.position + Vector2(0, std::rand()%3-1);
    if (abs(p2.position.y) > 3) {
        p2.position.y = 0;
    }
}
```

Listing 1: main.cpp

```
#pragma once

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

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"

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();
}
```

Listing 3: device.cpp

```
#pragma once

void Intro();
```

Listing 4: intro.hpp

```
#include "device.hpp"
#include "intro.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());
}
```

Listing 5: intro.cpp

```
#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

```
#pragma  once

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

class Player {
public:
    Vector2 position;
    Player(uint8_t dirInput, uint8_t smashInput, Vector2 position =
   Vector2());
    void Draw();
    void CheckInput();
private:
    uint8_t dirInput;
    uint8_t smashInput;
};
```

Listing 8: player.hpp

```
#include "esp32-hal-gpio.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() {

}
```

Listing 9: player.cpp

```
#pragma once

#include "vector2.hpp"
class Ball {
public:
    Vector2 position;
    Ball(Vector2 position = Vector2());
    void Draw();
private:
};
```

Listing 10: ball.hpp

```
#include "ball.hpp"
#include "vector2.hpp"

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

}

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

Listing 11: ball.cpp

2.5 Hasil

2.6 Progress

1. Dapat menampilkan Intro
2. Dapat menampilkan paddle dan bola

2.7 Target

1. Membaca nilai input
2. Menampilkan Skor

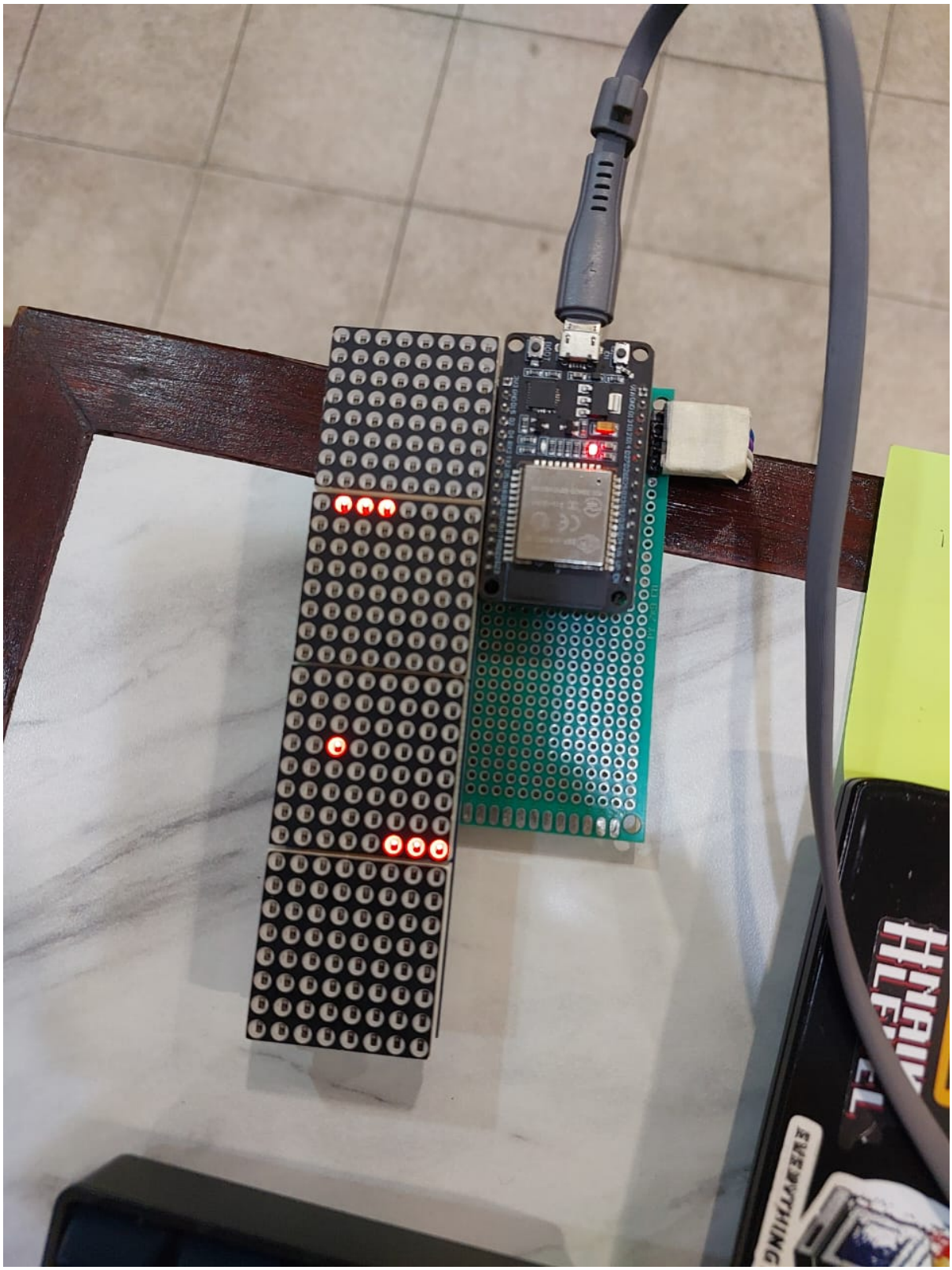

Figure 4: Hasil sementara