

# Vakitmatik P4 LED Panel Sistemi

## Nihai Gereksinim ve Altyapı Analizi

### 1. Proje Özeti

Bu proje; cami/mescit kullanımı için tasarlanan, 18 adet P4 (80x40) panelin birleşiminden oluşan toplam **240x240 piksel (57.600 LED)** devasa ekranın, **ESP32-S3** mikrodenetleyicisi ile titremesiz (flicker-free) ve yüksek performanslı sürülmesini sağlayacak özel yazılım (C++ Kütüphanesi) altyapısının geliştirilmesidir.

### 2. Kesinleşen Donanım Parametreleri

İncelenen donanım fotoğrafları doğrultusunda sistemin fiziksel yapısı tamamen netleşmiştir:

- **Ana Kontrolcü (MCU):** ESP32-S3 N16R8 (16MB Flash, 8MB PSRAM).
  - *Not:* 240x240 çözünürlüğün gerektirdiği yüksek bellek (RAM) ihtiyacı, bu kartın üzerindeki 8MB'lık PSRAM modülü tahsis edilerek çözülecektir.
- **LED Panel Tipi:** P4 RGB SMD LED Panel.
- **Tek Panel Çözünürlüğü:** 80 (Genişlik) x 40 (Yükseklik) Piksel.
- **Toplam Matris Dizilimi:** 3 Sütun x 6 Satır = 18 Panel (240x240 Piksel).
- **Sürücü Entegresi (Driver IC):** CHIPONE ICN2037BP (16-Kanal Sabit Akım Sürücü).
- **Sinyal Güçlendirici (Buffer IC):** DP245D.
- **Tarama Oranı (Scan Rate):** 1/10 Scan.

### 3. Yazılım Mimarisi ve "Custom Mapping" (Özel Haritalama)

Sistemin en kritik yazılımsal zorluğu, standart dışı olan **1/10 Scan ve 80x40 çözünürlük** kombinasyonudur.

- **Sorun:** HUB75 portunda 2 adet RGB kanalı bulunur. 1/10 tarama ile fiziksel olarak sadece 20 satır (2 x 10) kontrol edilebilir. Ancak panel 40 satırdır.
- **Çözüm (Folded Matrix Mimarisi):** Kütüphane çekirdeği (I2S DMA Engine), tek bir paneli 80x40 yerine **160x20** boyutundaymış gibi yapılandıracaktır. Üzerine yazılacak özel bir "**Koordinat Haritalama (Coordinate Mapper)**" algoritması sayesinde; kullanıcı ekrana yazı gönderdiğinde, sistem bu koordinatları otomatik olarak "katlanmış" piksellere çevirip donanıma iletecektir.

## 4. Geliştirilecek API (Kullanıcı Fonksiyonları)

Müşterinin karmaşık DMA veya Mapping matematiği ile uğraşmaması için aşağıdaki basit kullanım fonksiyonları (Wrapper) geliştirilecektir:

1. `vakitmatikBaslat()`: ESP32-S3 I2S DMA ayarlarını, PSRAM tahsisini ve ICN2037BP clock frekanslarını başlatır.
2. `pikselCiz(x, y, renk)`: Geliştirilen 240x240 matris üzerinde istenen noktayı yakar (Custom Mapper dahil).
3. `metinYazdir(x, y, "Metin", boyut, renk)`: İstenen koordinata metin basar (İmsak, Sabah, vb. yazıları için).
4. `ekraniTemizle()`: Tüm ekran buffer'ını sıfırlar.
5. `ekraniGuncelle()`: Arka planda hazırlanan görüntüyü titremesiz bir şekilde (Double Buffering) ekrana yansıtır.

## 5. İleriye Dönük Hazırlık (Aşama 2)

Şu an yazılacak olan kütüphane, ana döngüyü (loop) bloklamayan **Asenkron (Non-blocking)** bir yapıda tasarlanacaktır. Bu sayede bir sonraki aşamada projeye **Wi-Fi modülü ve Web Sunucu / Mobil Uygulama** eklendiğinde, ekrana veri akışı kesintiye uğramayacak ve görüntüde bozulma yaşanmayacaktır.

## 6. Proje Aşamaları ve Teslimat

- **Adım 1:** VS Code (PlatformIO) üzerinde kütüphane iskeletinin kurulması.
- **Adım 2:** I2S DMA ve PSRAM konfigürasyonlarının kodlanması.
- **Adım 3:** 80x40 1/10 Scan için özel Coordinate Mapper algoritmasının yazılması.
- **Adım 4:** Teslimat ve donanım testleri. (Müşteri ESP32-S3 kartını teslim aldığı anda yükleyip sonucu bildirecektir. Gerekirse renk kanalları (RGB → RBG vb.) için uzaktan ince ayar yapılacaktır).