



**ESKİŞEHİR OSMANGAZI UNIVERSITY**  
**DEPARTMENT OF ELECTRICAL-ELECTRONICS ENGINEERING**  
**AND**  
**DEPARTMENT OF COMPUTER ENGINEERING**

**INTRODUCTION TO MICROCOMPUTER**  
**TERM PROJECT**  
**2025-2026**

Akıllı Ev Sistemi

152120231126 Görkem GÖKSEL Computer Engineering

<https://github.com/grkmgoksel-jpg/MicroComputerProject>

151220222137 Kadir Önay Electric Electronic Engineering

[kadir0123/Microcomputers-Project-main: Microcomputers-Project-main](https://github.com/kadir0123/Microcomputers-Project-main)

151220202047 Yusuf TARGAY Electric Electronic Engineering

<https://github.com/YusufTargay/pic16f877A-Home-Automation>

152120231179 Vahit Furkan Bozdemir Computer Engineering

<https://github.com/Wayt-to/Microcomputers-Project>

151220202036 Serkan Sarıkaya Electric Electronic Engineering

[https://github.com/Sserkan577/pic16f877a-home\\_automation](https://github.com/Sserkan577/pic16f877a-home_automation)

**January 2026**

## **İÇİNDEKİLER**

<b>1. Giriş.....</b>	<b>2</b>
<b>2. Tasarım.....</b>	<b>3</b>
2.1 Fiziksel Donanım Tasarımı.....	3
2.1.1 Board #1.....	3
2.1.2 Board #2.....	6
2.2 Yazılım Mimari Tasarımı.....	10
2.2.1 Yazılım Mimari Yapısı (UML Sınıf Diyagramı Analizi).....	10
2.2.2 Haberleşme ve Donanım Entegrasyonu (Blok Diyagram Analizi).	11
2.2.3 Veri Güncelleme Stratejisi.....	12
2.2.4 Örnek Girdiler İçin Örnek Çıkışlar ve Kullanım Örnekleri.....	13
<b>3. Görev Dağılımı.....</b>	<b>19</b>
<b>4. Sonuç.....</b>	<b>19</b>
<b>5. Kaynakça.....</b>	<b>21</b>

# 1. Giriş

**Projenin Amacı:** Bu proje, PIC16F877A mikrodnetleyicileri kullanılarak entegre bir akıllı ev sistemi geliřtirmeyi amalamaktadır.Sistem;sıcaklık/basın kontrolü ve perde yönetimi olmak üzere iki işlevi yerine getiriyor.Projenin hedefi, bu donanımların assembly kodları ile yönetilmesini sağlamak ve aynı zamanda UART seri haberleşme protokolü üzerinden bir API ile uzaktan kontrol edilebilir hale getirmektir.

**Sistem Mimarisi:** Sistem, "Board #1" ve "Board #2" olarak adlandırılan iki ayrı PIC16F877A mikrodnetleyici modülünden ve kullanıcı arayüzü için API olmak üzere 3 kısımdan oluşmaktadır.

- **Board #1 (Klima Sistemi):** Ev otomasyon sisteminde ortam sıcaklığını kontrol eden birimdir ve PIC16F877A mikrodnetleyicisi kullanılarak tasarlanmıştır. Sistem, Temperature System modülünden alınan ortam sıcaklığına göre ısıtıcı ve soğutucu fanı otomatik olarak kontrol eder.

Kullanıcı tarafından girilen hedef sıcaklık değeri ile ölçülen ortam sıcaklığı karşılaştırılır. Ortam sıcaklığı hedef değerin üzerine çıktığında fan çalıştırılır, altına düřtüğünde ise ısıtıcı aktif hale getirilir. Fan hızı sistem tarafından izlenir.

- **Board #2 (Perde Kontrol Sistemi):** Ortamdaki ışık şiddetini (LDR) ve dış ortam verilerini (BMP180) izleyerek veya kullanıcının isteğine göre step motoru sürerek perdeyi açıp kapatmaktadır.
- **API:** Projenin yazılım ayağı olan Python tabanlı API; mikrodnetleyicilerle seri haberleşme (UART) kurarak donanım kontrolünü soyutlaştırır. Nesne yönelimli mimarisi sayesinde COM portları üzerinden gelen sensör verilerini işler, kullanıcı arayüzüne (GUI) aktarır ve kullanıcı komutlarını mikrodnetleyicilerin işleyebileceği binary formatlara dönüřtürerek sistemin uzaktan yönetimini sağlar.

## 2. Tasarım

### 2.1 Fiziksel Donanım Tasarımı

#### 2.1.1 Board #1

Board #1, ev otomasyon sistemi içerisinde ortam sıcaklığının kontrolünden sorumlu birimdir ve PIC16F877A mikrodenetleyicisi kullanılarak tasarlanmıştır. Sistem, PICSimLab Temperature System modülü üzerinden elde edilen ortam sıcaklığı bilgisine göre çalışmakta ve kullanıcı tarafından belirlenen hedef sıcaklık değerini referans alarak ısıtıcı ve soğutucu fanın kontrolünü otomatik olarak gerçekleştirmektedir.

- 1. Sıcaklık Kontrol Mekanizması (Temperature System):**Board #1’de ortam sıcaklığı, PICSimLab Temperature System modülü üzerinden okunmaktadır. Ölçülen sıcaklık değeri sistem belleğinde saklanmakta ve sürekli güncellenmektedir. Isıtıcı ve soğutucu fanın çalışma durumları, bu güncel ortam sıcaklığı değeri esas alınarak verilen kontrol kararları doğrultusunda belirlenmektedir.



Temperature System Bağlantıları

2. **Isıtıcı ve Soğutucu Fan Kontrolü:** Sistemde, kullanıcı tarafından belirlenen hedef sıcaklık değeri ile ölçülen ortam sıcaklığı karşılaştırılmaktadır. Ortam sıcaklığının hedef değerin üzerinde olması durumunda soğutucu fan aktif hale getirilirken, ortam sıcaklığının hedef değerin altına düşmesi halinde ısıtıcı çalıştırılmaktadır. Güvenli ve kararlı bir çalışma sağlamak amacıyla, ısıtıcı ve soğutucu fanın aynı anda çalışmasına izin verilmemektedir.

```
116 ;----- HVAC CONTROL -----
117 HVAC_CONTROL:
118     BANKSEL PORTC
119     BCF     PORTC, 4      ; Heater OFF
120     BCF     PORTC, 5      ; Cooler OFF
121
122     MOVF    tempC, W
123     SUBLW   DESIRED_INT
124
125     BTFSS   STATUS, C
126     GOTO    COOLER_ON
127
128     BTFSC   STATUS, Z
129     RETURN
130
131 HEATER_ON:
132     BANKSEL PORTC
133     BSF     PORTC, 4      ; Heater ON
134     RETURN
135
136 COOLER_ON:
137     BANKSEL PORTC
138     BSF     PORTC, 5      ; Cooler ON
139     RETURN
140
```

#### Isıtıcı ve Soğutucu Çalışma Mantığı

3. **Fan Hızı Ölçümü (Tachometer):** Soğutucu fanın dönüş hızı, Temperature System modülü içerisinde yer alan takometre yardımıyla ölçülmektedir. Takometreden elde edilen sinyaller kullanılarak fan hızı **dakikadaki devir sayısı (rpm)** cinsinden hesaplanır. Hesaplanan fan hızı değeri sistem belleğinde saklanır ve kontrol ile izleme amaçlı olarak sürekli güncel tutulur.



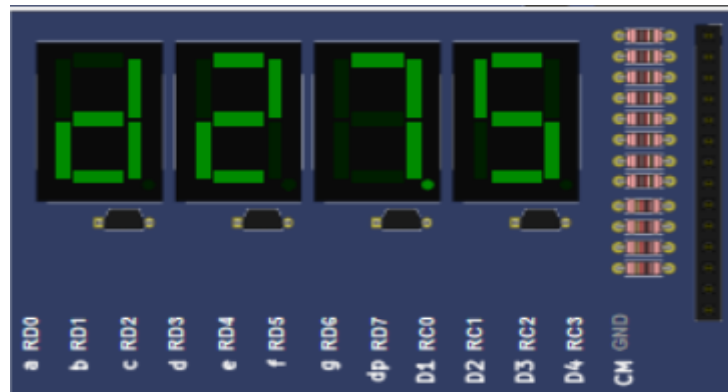
7-Segmentte Gösterilen Anlık Fan Hızı

4. **Kullanıcı Girişi (Keypad):**Board #1’de kullanıcı, hedef sıcaklık değerini keypad aracılığıyla sisteme girmektedir. Girilen sıcaklık değeri belirlenen sınırlar açısından kontrol edilmekte ve geçerli olması durumunda sistem belleğine kaydedilmektedir. Tanımlı aralıkların dışında kalan geçersiz girişler ise kabul edilmeyerek belleğe yazılmamaktadır.



Keypad Bağlantıları

5. **Görüntüleme (7-Segment / Display):**Board #1’de görüntüleme işlemi 7-segment display kullanılarak gerçekleştirilmektedir. Kullanıcı tarafından belirlenen hedef sıcaklık (27.5), ölçülen ortam sıcaklığı ve soğutucu fan hızı bilgileri belirli zaman aralıklarıyla ekranda gösterilmektedir. Bu sayede kullanıcı, sistemin anlık çalışma durumunu kolayca takip edebilmektedir.

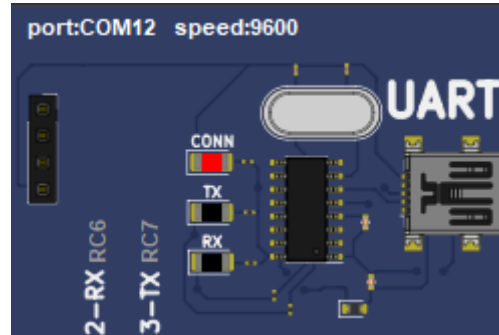


7-Segmen Display Bağlantıları (Desired)



(Real-time temperature)

6. **Uzaktan Haberleşme (UART)**Board #1, UART seri haberleşme protokolü aracılığıyla bilgisayar ile iletişim kurmaktadır. Ortam sıcaklığı, hedef sıcaklık ve fan hızı gibi sistem bilgileri PC'ye gönderilmekte, aynı zamanda PC üzerinden hedef sıcaklık değeri sisteme iletilerek uzaktan kontrol imkânı sağlanmaktadır.



*UART Modülü Bağlantıları*

### 2.1.2 Board #2

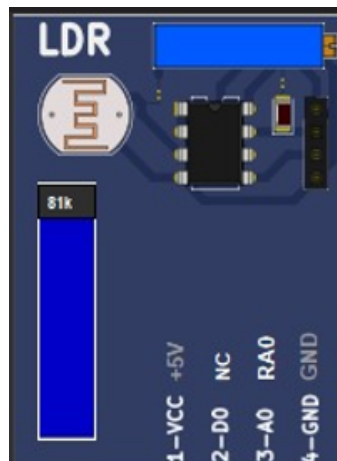
Board #2: Perde Kontrol ve Işık Algılama Sisteminin modülüdür. Akıllı ev sistemini perde kontrol aşamasıdır. Bu aşama içerisinde otomatik şekilde dış ortam aydınlığına bağlı olarak, mekanik olarak rotary potansiyometre ile birlikte manuel kontrol ve API üzerinden kullanıcı isteği ile beraber perdeyi açıp kapatır. Board #2, PIC16F877A mikrodenetleyicisi üzerinde çalışıyor olup belirli görevleri yerine getirir:

1. Perde Mekanizması Kontrolü (Step Motor): Perdenin açılıp kapanma hareketi, bir Step Motor kullanılarak kontrol edilmektedir. Motorun saat yönünde veya saat yönünün tersine dönmesi ile perdenin açıklık oranı ayarlanmaktadır.



Step Motor Bağlantıları

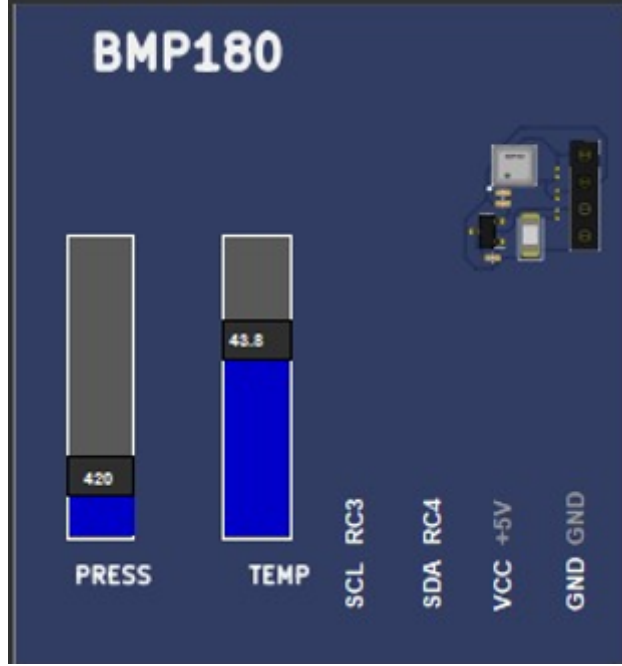
2. Otomatik Işık Algılama (LDR): Sistemde bulunan LDR ışık sensörü, ortamdaki ışık şiddetini sürekli izler. Işık seviyesi belirli bir eşik değerin altına düştüğünde, sistem otomatik olarak perdeyi tamamen kapatacak şekilde programlanmıştır ve eşik değerinin üstünde olduğu durumda perdeyi tamamen açacak şekilde programlanmıştır.





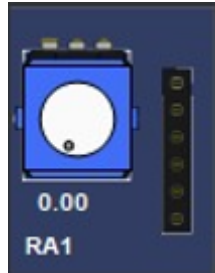
### *LDR Sensörü Bağlantıları*

3. Dış Ortam Veri Toplama (BMP180): Evin dışındaki hava koşullarını izlemek amacıyla BMP180 sensörü kullanılmıştır. Bu sensör aracılığıyla dış ortam sıcaklığı ve hava basıncı verileri okunmuştur ve bu veri LCD ekrana gönderilmiştir.



BPM 180 Sıcaklık ve Basınc Sensörü Bağlantıları

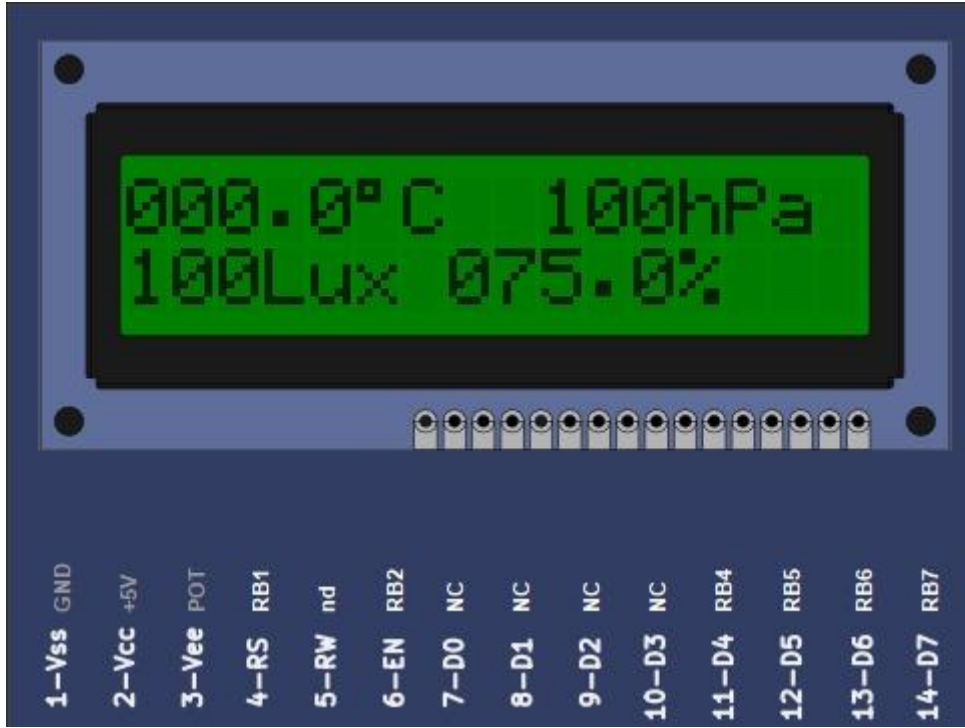
4. Manuel Perde Ayarlama: Kullanıcılar, sisteme bağlı bir rotary potansiyometre aracılığıyla perde açıklığını manuel olarak %0 ile %100 arasında ayarlayabilirler.



Rotary Potansiyometre Bağlantıları

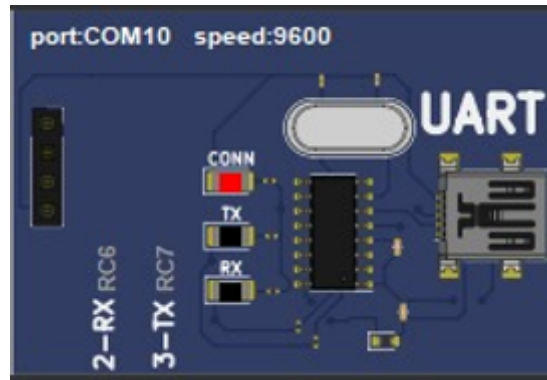
5. LCD ekran, sistemden anlık olarak alınan verileri kullanıcıya görsel olarak sunmak amacıyla kullanılmıştır. Ekranın **ilk satırında** dış ortam sıcaklığı (°C) ve hava basıncı (hPa) bilgileri gösterilmektedir. **İkinci satırda** ise ortamın ışık şiddeti (Lux) ve perdenin mevcut durumu (%) anlık olarak görüntülenmektedir. Bu sayede kullanıcı,

perde kontrol sistemiyle ilişkili tüm temel verileri tek bir ekranda kolayca takip edebilmektedir.



LCD Ekran Bağlantıları

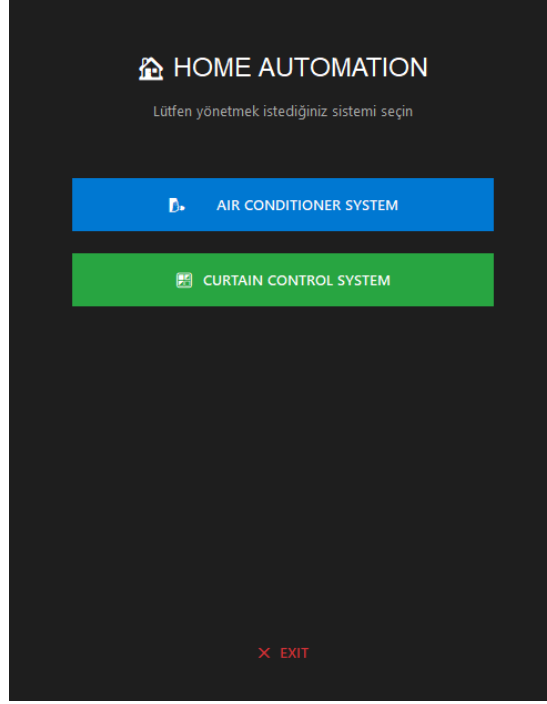
6. Uzaktan Haberleşme (UART): Diğer modülde olduğu gibi, Board #2 de topladığı tüm sensör verilerini ve aktüatör durumlarını UART protokolü üzerinden merkezi PC yazılımına iletir ve PC'den gelen "Perdeyi %50 aç" gibi komutları uygular.



UART Modülü Bağlantıları

7. Program ana giriş ekranı, kullanıcıya ev otomasyon sisteminin kontrol edilebileceği temel menüyü sunar. Bu ekranda kullanıcı, **Air Conditioner System** ve **Curtain**

**Control System** seçeneklerinden birini seçerek ilgili sisteme kolayca geçiş yapabilir. Alt kısımda yer alan **Exit** seçeneği ile program güvenli bir şekilde sonlandırılabilir. Sade ve anlaşılır tasarım sayesinde kullanıcı, sistem içinde hızlı ve rahat bir şekilde yönlendirilir.



*Program Ana Giriş Ekranı*

## 2.2 Yazılım Mimari Tasarımı

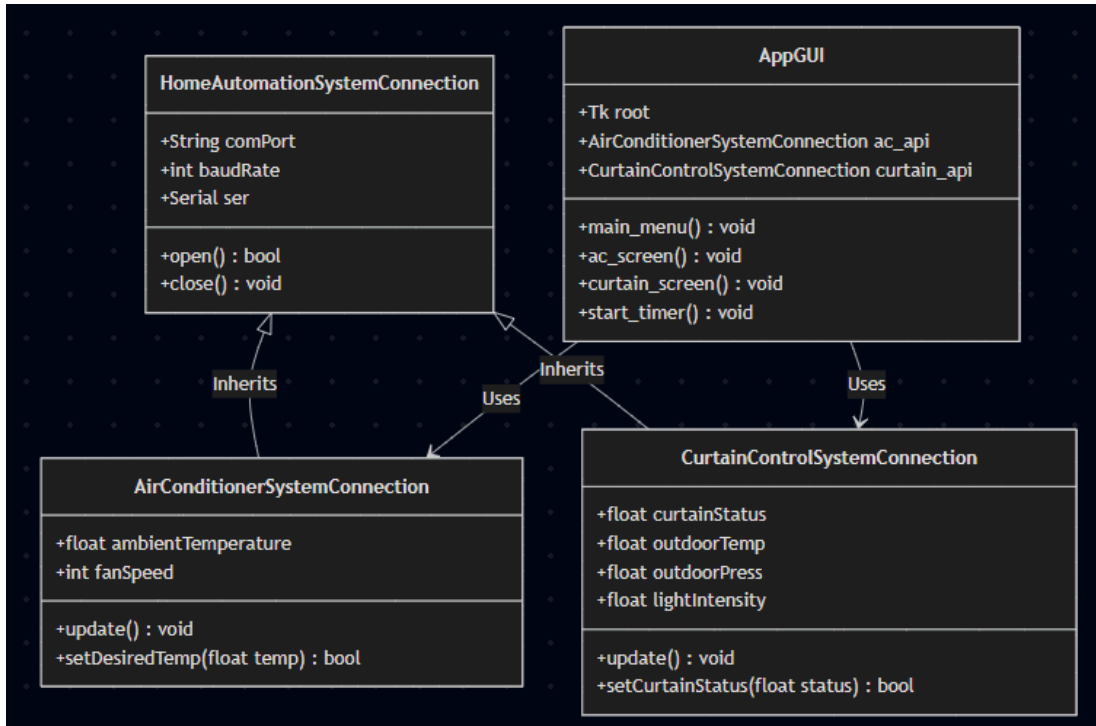
### 2.2.1 Yazılım Mimari Yapısı (UML Sınıf Diyagramı Analizi)

Yazılımın temelini oluşturan UML Sınıf Diyagramı, sistemin **Nesne Yönelimli Programlama (OOP)** yeteneklerini sergilemektedir:

- **Kalıtım ve Soyutlama:** HomeAutomationSystemConnection sınıfı, tüm haberleşme birimleri için bir üst katman (Base Class) olarak tanımlanmıştır. Seri port açma/kapama gibi temel fonksiyonlar bu sınıfta toplanarak kod tekrarı önlenmiş ve **Soyutlama (Abstraction)** sağlanmıştır.
- **Modüler Genişleme:** Klima (AirConditionerSystemConnection) ve Perde (CurtainControlSystemConnection) sınıfları, ana bağlantı sınıfından türetilerek

(Inheritance) kendilerine özgü veri işleme ve komut setlerini barındırır. Bu yapı, gelecekte sisteme yeni akıllı cihazların (aydınlatma, güvenlik vb.) kolayca eklenmesine olanak tanır.

- **Kontrol Katmanı:** AppGUI sınıfı, kullanıcı arayüzü ile donanım arasındaki köprüyü kurar. API katmanındaki sınıfları **Kompozisyon (Composition)** yoluyla bünyesinde barındırarak, kullanıcıdan alınan girdileri donanım komutlarına dönüştürür.



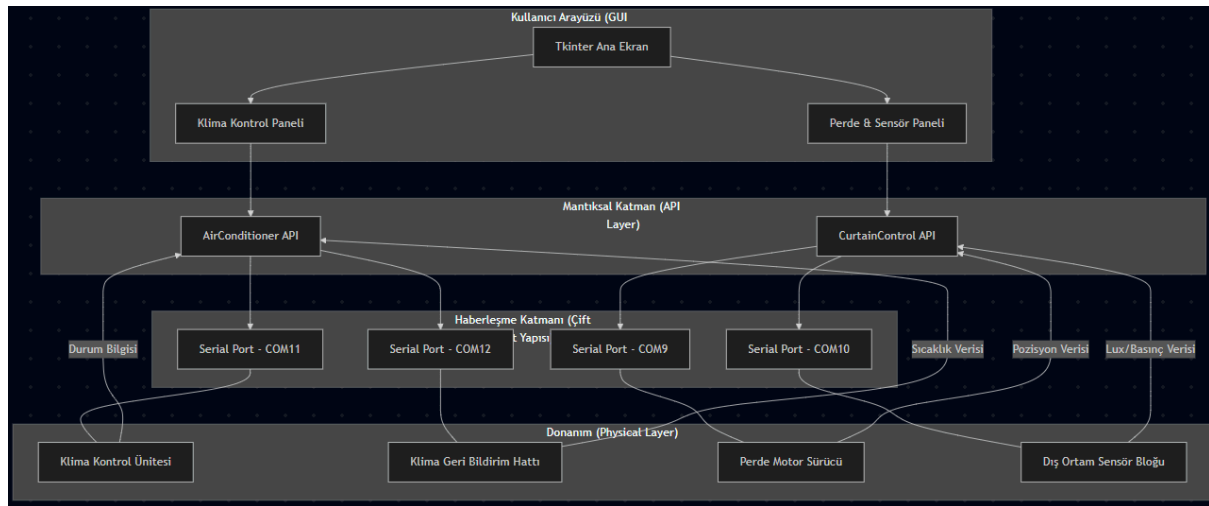
*UML Sınıf Diyagramı*

### 2.2.2 Haberleşme ve Donanım Entegrasyonu (Blok Diyagram Analizi)

Sistem blok diyagramı, kullanıcı arayüzünden fiziksel donanıma kadar olan veri akışını ve port yönetimini temsil eder. Tasarımda **Dört Katmanlı Model** benimsenmiştir:

1. **Arayüz Katmanı (GUI):** Tkinter kütüphanesi ile geliştirilen panel, kullanıcının sistem durumunu izlemesini ve parametreleri (sıcaklık, perde konumu) değiştirmesini sağlar.
2. **Mantıksal Katman (API):** Ham seri verileri anlamlı fiziksel değerlere (Celsius, Lux, hPa) dönüştüren ve komutları binary formatta paketleyen katmandır.

3. **Haberleşme Katmanı (Dual-Port Yapısı):** Sistemin güvenilirliğini artırmak adına port ayrımı yapılmıştır.
- **Klima Sistemi:** Kontrol ve geri bildirim verileri için **COM11** ve **COM12** portları ayrılmıştır.
  - **Perde ve Sensör Sistemi:** Hareket kontrolü ve sensör veri akışı için **COM9** ve **COM10** portları üzerinden asenkron bir haberleşme sağlanmaktadır.
4. **Fiziksel Katman (Hardware):** Mikrodenetleyici tabanlı kontrol üniteleri ve sensör bloklarından oluşur. Çift portlu yapı sayesinde, bir birimde oluşabilecek yoğun veri trafiğinin diğer sistemin tepki süresini (latency) etkilemesi engellenmiştir.

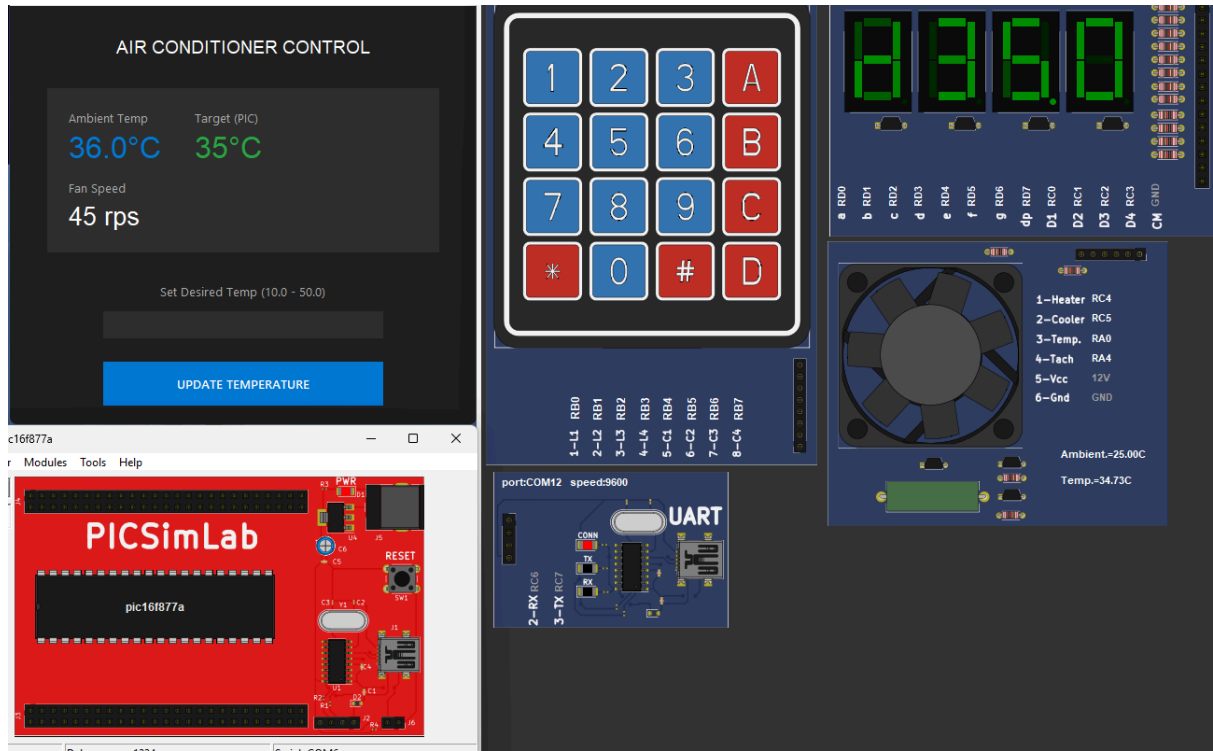


*Sistem Blok Diyagramı*

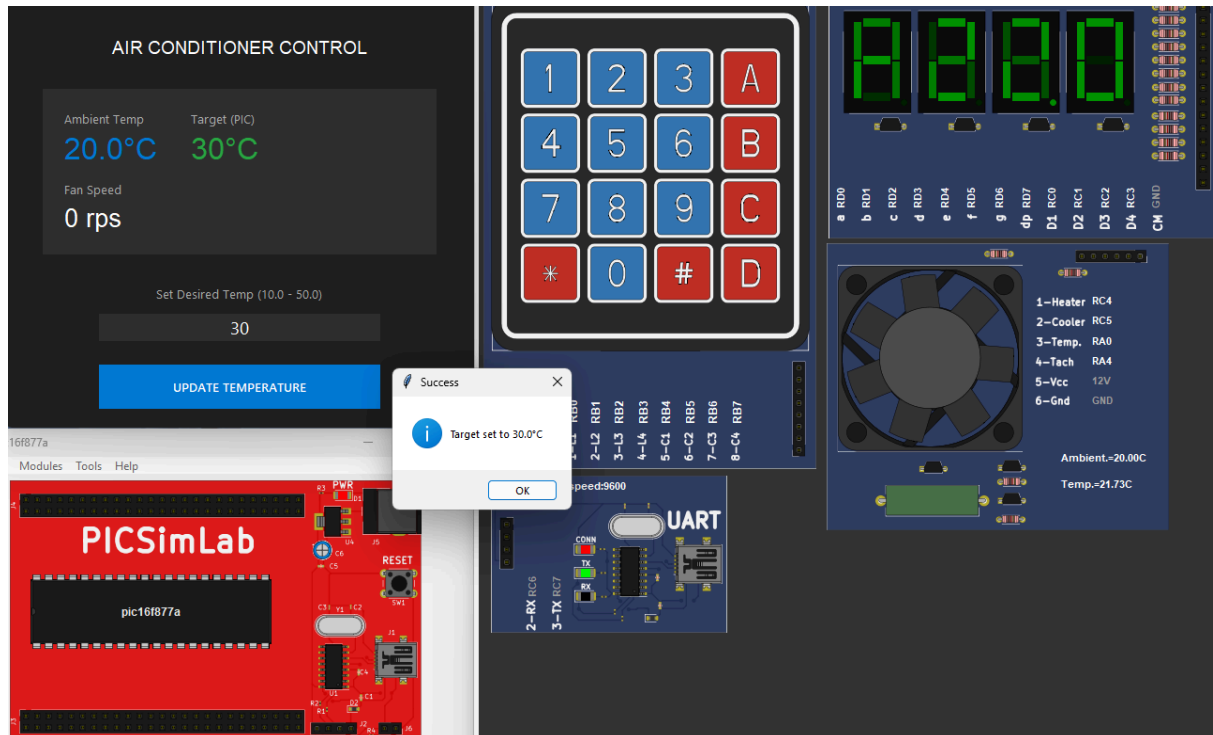
### 2.2.3 Veri Güncelleme Stratejisi

Sistemde veri akış sürekliliğini sağlamak için **Zamanlayıcı Tabanlı Sorgulama (Polling)** yöntemi kullanılmıştır. Yazılım katmanındaki `start_timer()` fonksiyonu, her 1000 milisaniyede bir donanıma sorgu göndererek arayüzdeki verilerin anlık olarak güncellenmesini sağlar. Bu, kullanıcıya kesintisiz bir izleme deneyimi sunarken sistem kaynaklarını da optimize eder.

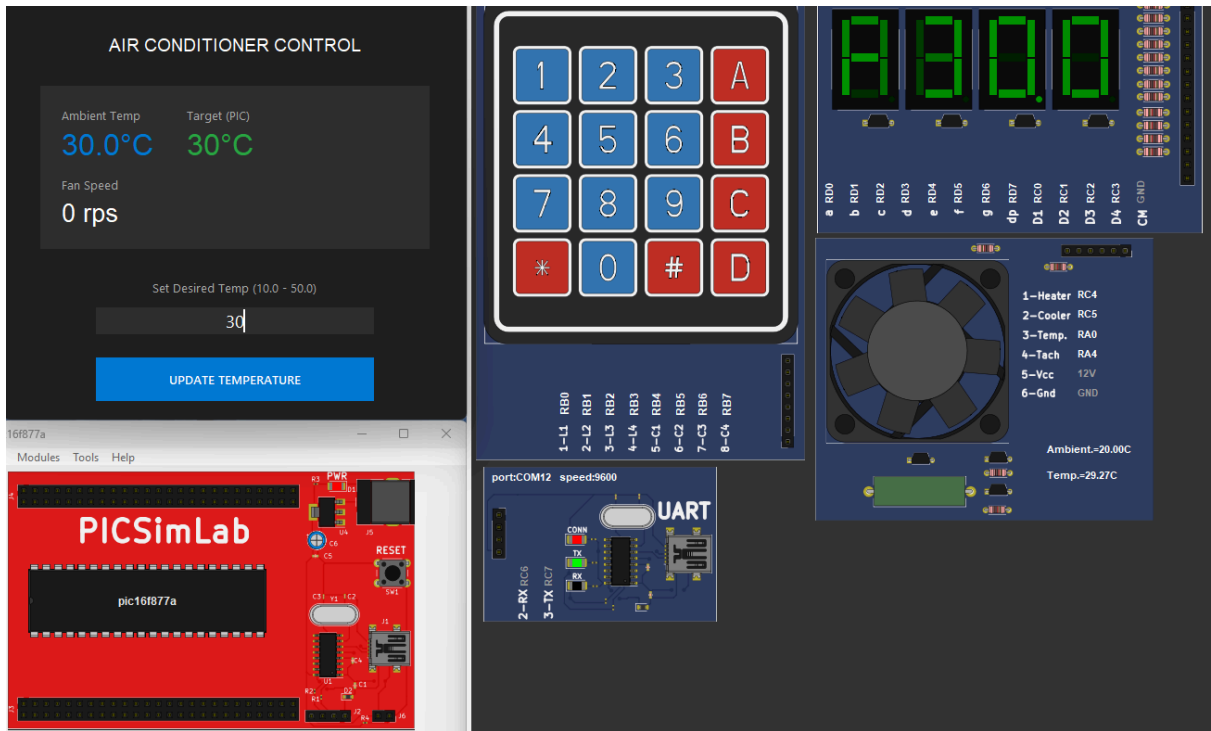
## 2.2.4 Örnek Girdiler İçin Örnek Çıktılar ve Kullanım Örnekleri



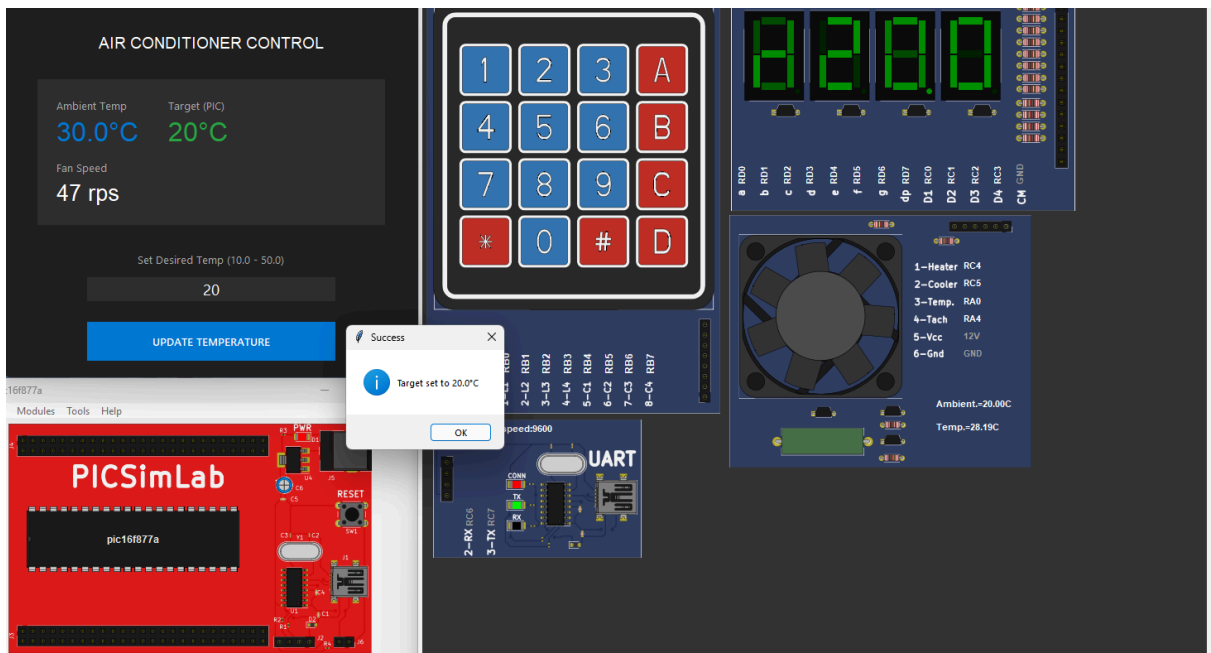
### Board1 Geneli



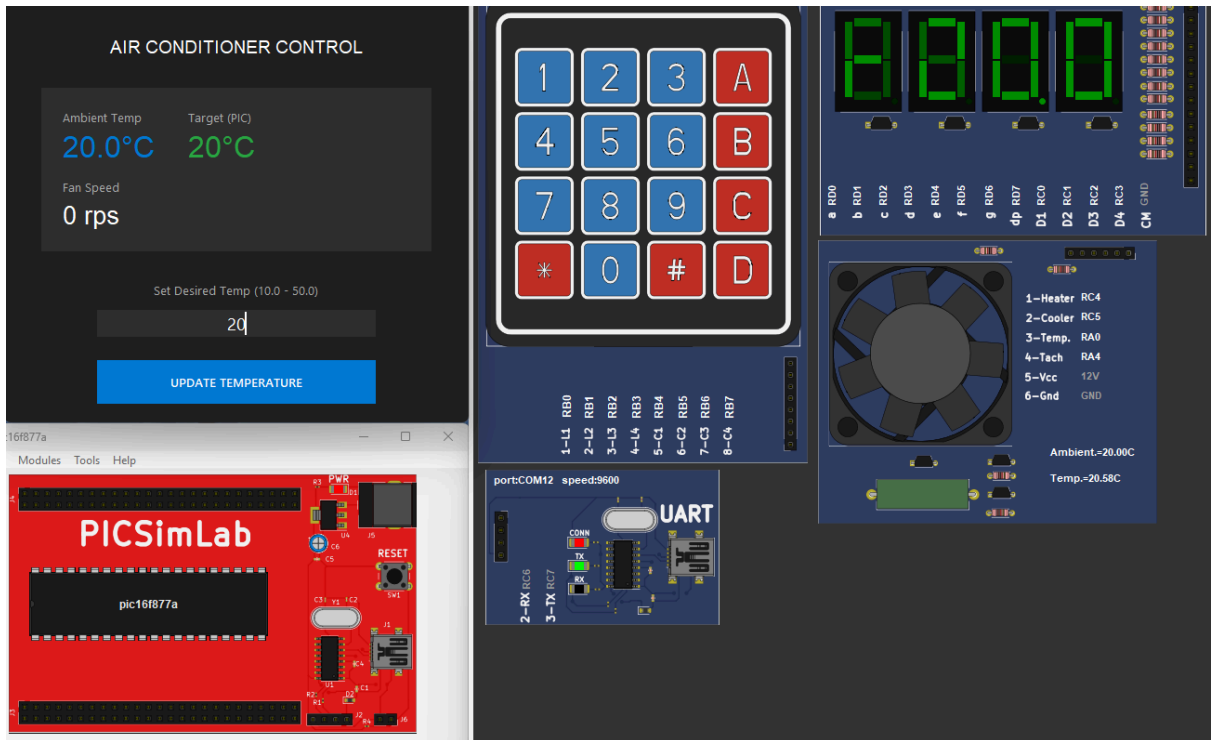
### Sıcaklık Arttırma



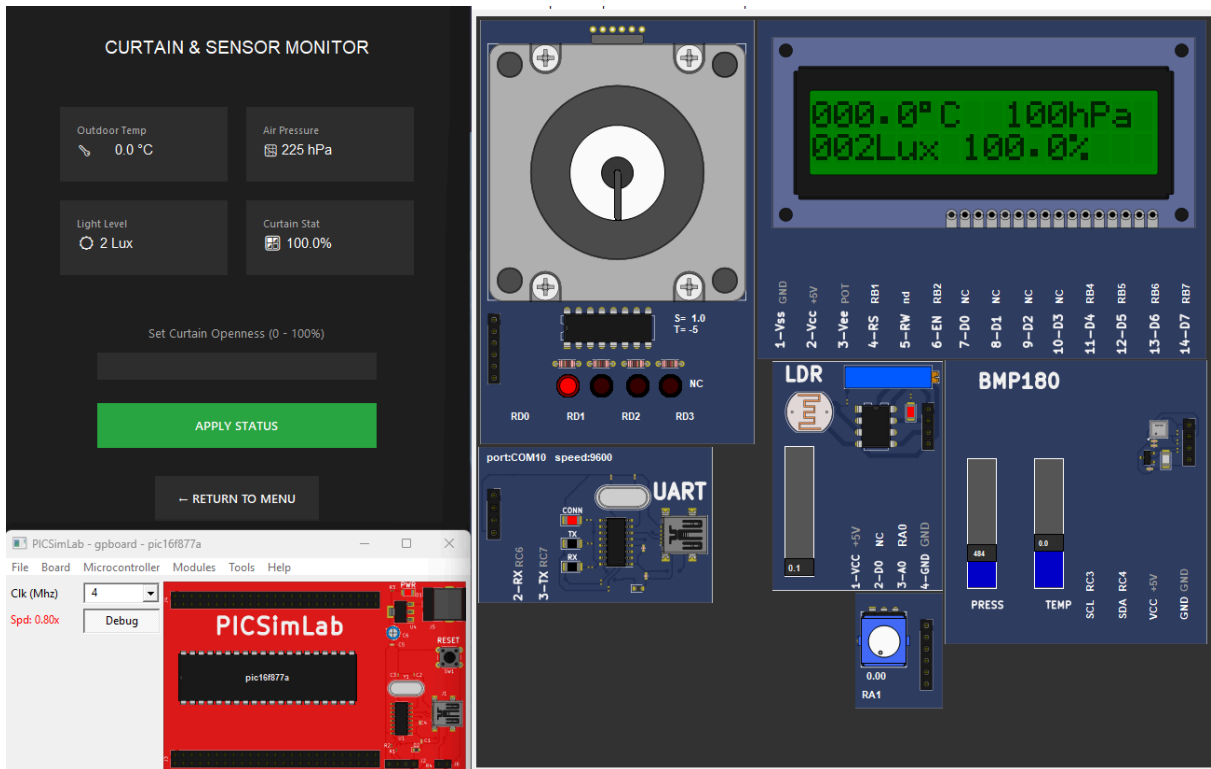
*Artmış Sıcaklık*



*Düşürülmek İstenen Sıcaklık*

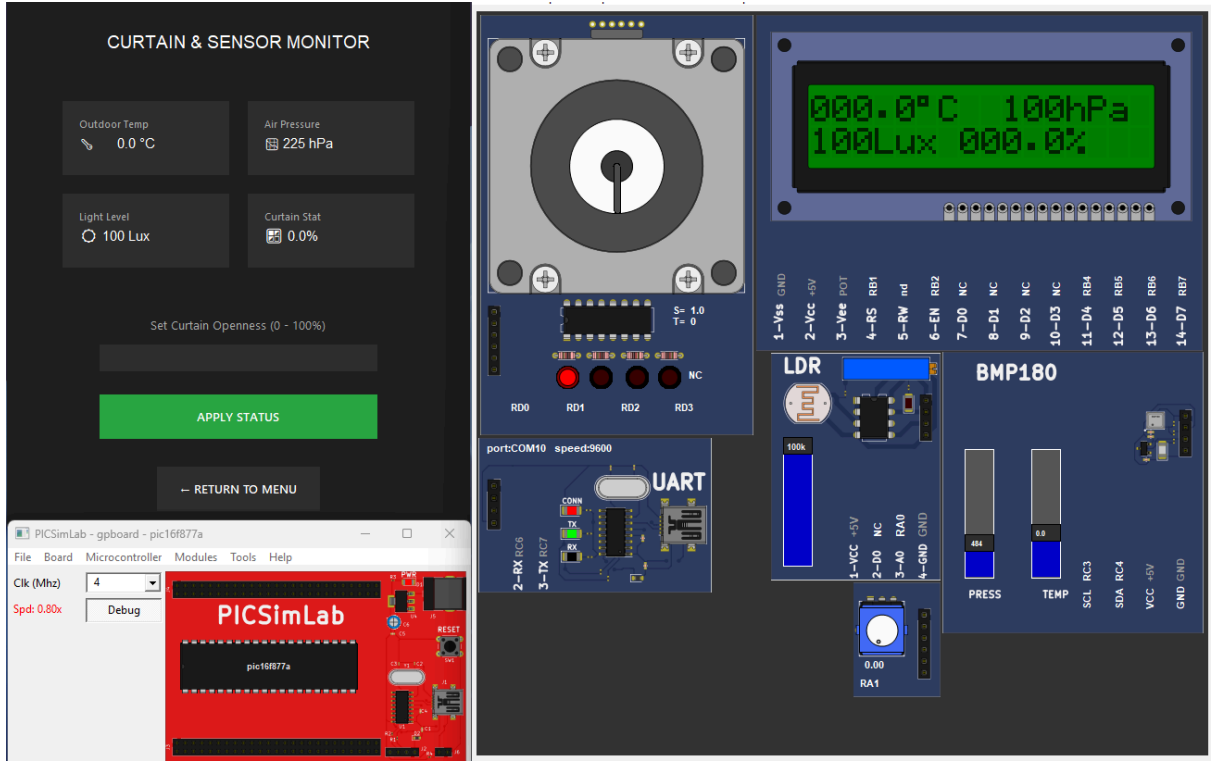


### *Düşmüş Sıcaklık*

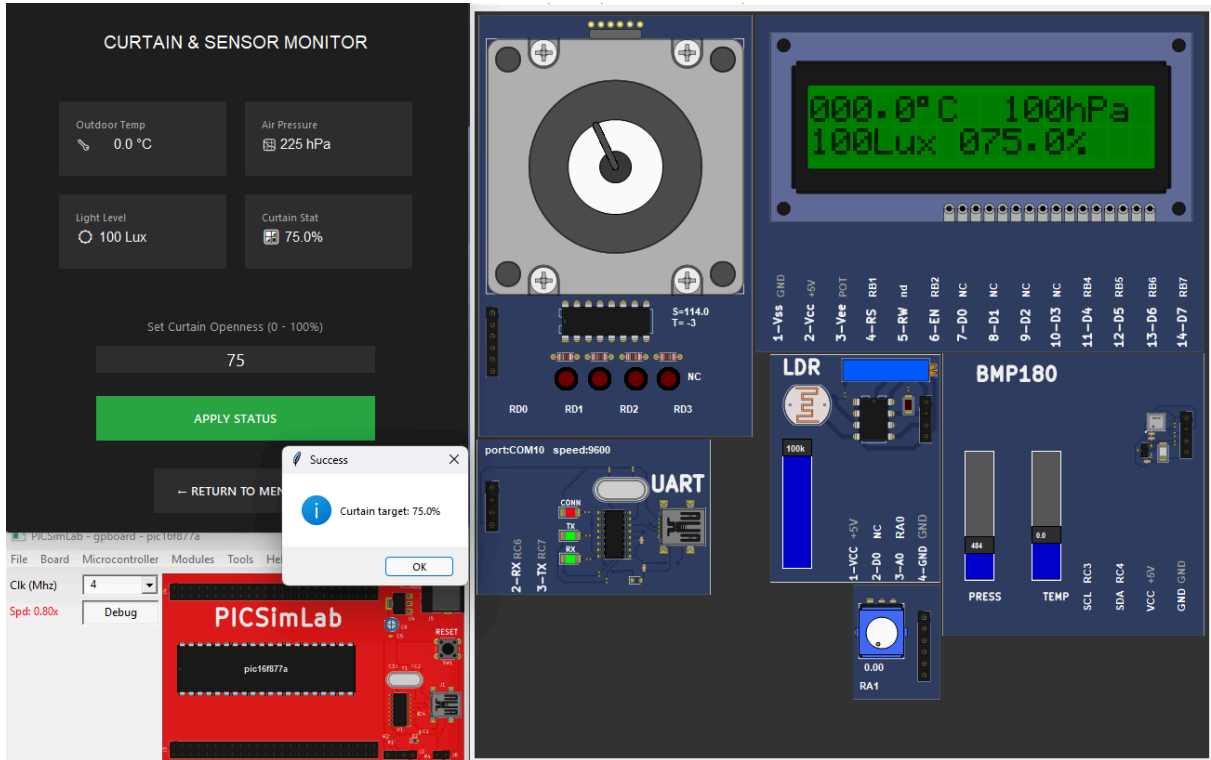


### *LDR Durumuna Göre Perde Durumu (%100)*





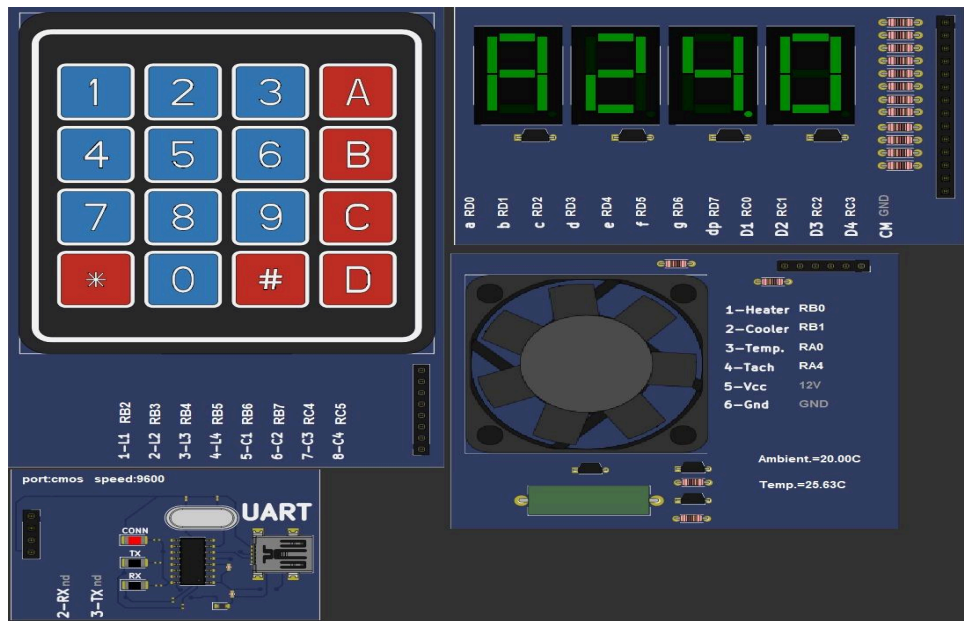
*LDR Durumuna Göre Perde Durumu (%)*



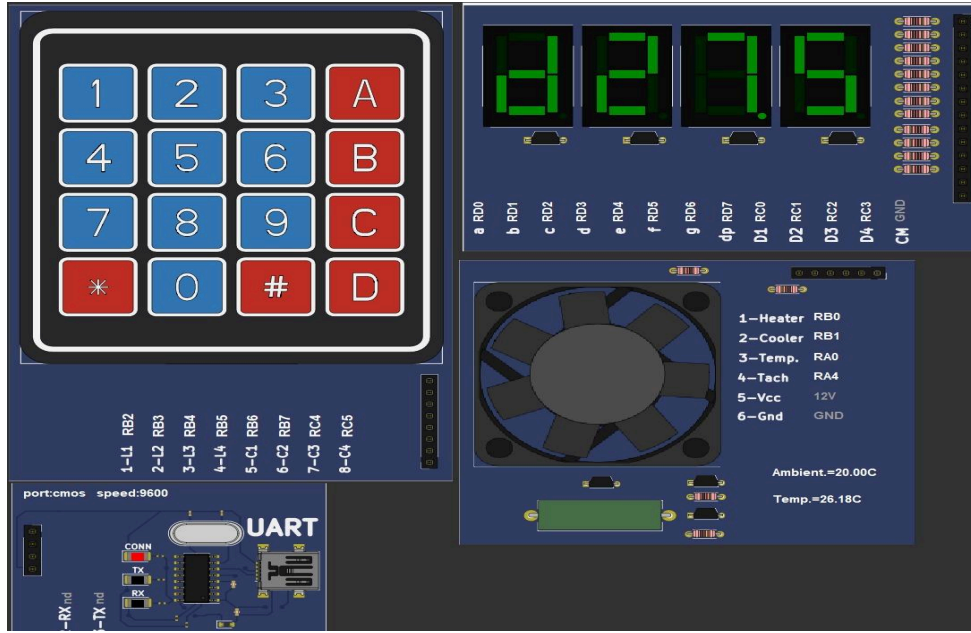
*API İsteği İle Ayarlama*



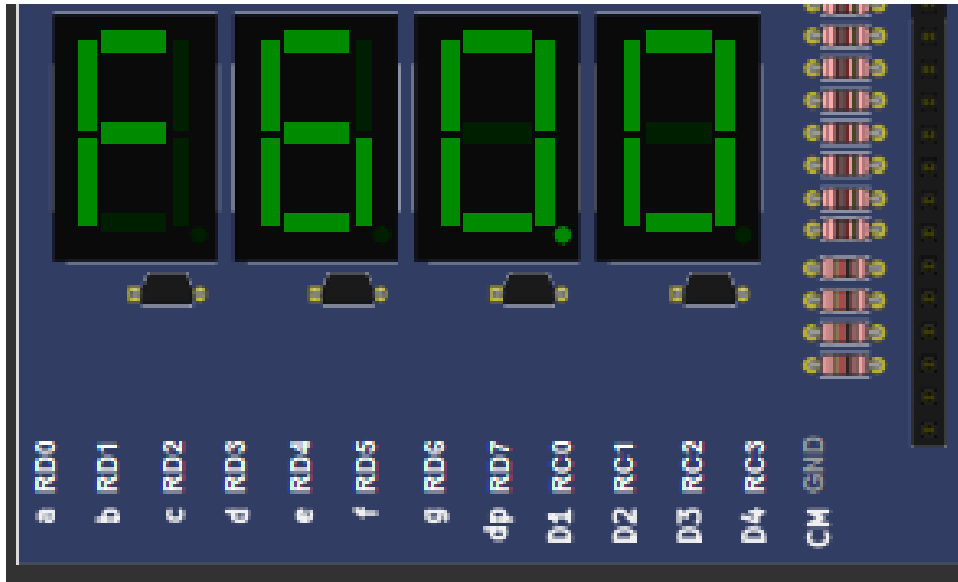
*Potansiyometre İle Manuel Perde Ayarlama*



*Anlık Sıcaklık Değerinin 7-Segment Displayde Gösterimi*



Keypadden Girilen İstenen Sıcaklık Değeri



Fan Dönüş Hızının 7-Segmentte Gösterimi

### 3. Görev Dağılımı

Grup Üyeleri	Görev Dağılımı
Kadir Önay	Step Motor, Ldr, Uart Modülü ve Haberleşme, Lcd ekran, Rotary Potansiyometre
Görkem Göksel	Rotary Potansiyometre, Step Motor, LCD
Vahit Furkan Bozdemir	API ve Uygulama Arayüzü, LCD, Rotary Potansiyometre, Uart Modülleri ve Haberleşme
Yusuf Targay	Keypad, 7 Segment Display, Temperature System, Uart Module
Serkan Sarıkaya	Keypad, 7 Segment Display, Temperature System, Uart Module

### 4. Sonuç

#### Projenin Genel Değerlendirmesi

Bu proje kapsamında, mikrodenetleyici tabanlı bir ev otomasyon sistemi tasarlanmış ve simülasyon ortamında çalışır hale getirilmiştir. Sıcaklık kontrolü, perde kontrolü, sensör okuma ve PC ile haberleşme gibi farklı konuların tek bir sistem altında toplanması, projenin kapsamını oldukça genişletmiştir. Proje süreci, teorik bilgilerin pratikte nasıl uygulandığını görme açısından öğretici ve geliştirici olmuştur.

## **Proje Sonuçlarının Değerlendirilmesi**

Proje sonunda, her iki mikrodenetleyici kartının kendisine verilen görevleri yerine getirdiği görülmüştür. Sensörlerden alınan veriler doğru şekilde işlenmiş, motor ve diğer çıkış birimleri beklenen şekilde kontrol edilmiştir. UART haberleşmesi sayesinde sistem ile PC arasında veri alışverişi sağlanmış ve kullanıcı, sistem durumunu takip edebilmiştir. Elde edilen sonuçlar, projenin hedeflerine büyük ölçüde ulaştığını göstermektedir.

## **Takım Çalışması Hakkında Yorumlar**

Proje süreci boyunca ekip çalışmasının önemi net bir şekilde anlaşılmıştır. Her ekip üyesinin farklı bir modül üzerinde çalışması, iş yükünün dengeli dağılmasını sağlamıştır. Zaman zaman yaşanan teknik sorunlar, ekip içinde yapılan fikir alışverişleri sayesinde aşılmıştır. Bu süreç, yalnızca teknik bilgi değil; iletişim, sorumluluk alma ve birlikte problem çözme becerilerini de geliştirmiştir.

## **Projenin Artıları ve Eksileri**

### **Artıları:**

- Farklı sensör ve donanımların aynı sistem içinde entegre edilmesi
- Modüler yapı sayesinde sistemin anlaşılır ve düzenli olması
- Assembly dili ile mikrodenetleyici mantığının daha iyi kavranması
- PC tarafında yazılım geliştirme ile donanımın birlikte kullanılması

## Eksileri:

- Projenin simülasyon ortamıyla sınırlı kalması
- Assembly dilinin hata ayıklama sürecini zorlaştırması
- Kullanıcı arayüzünün temel seviyede olması

## İleride Yapılabilecek Geliştirmeler

İlerleyen çalışmalarda sistem gerçek donanım üzerinde uygulanarak performans ve güvenilirlik testleri yapılabilir. Kablosuz haberleşme eklenerek sistem uzaktan kontrol edilebilir hale getirilebilir. PC uygulaması daha kullanıcı dostu bir arayüzle geliştirilebilir. Ayrıca daha gelişmiş kontrol algoritmaları ve veri kayıt özellikleri eklenerek sistemin fonksiyonelliği artırılabilir.

## 5. Kaynakça

[1] Microchip Technology Inc., *PIC16F87X Data Sheet – 28/40/44-Pin Enhanced Flash Microcontrollers*, Mid-Range PIC Microcontroller Family,  
Available: <https://ww1.microchip.com/downloads/en/DeviceDoc/30292c.pdf>

[2] L. C. G. Gamboa, *PICSimLab – Programmable IC Simulator Laboratory*, GitHub Repository,  
Available: <https://github.com/lcgamboa/picsimlab>

[3] L. C. G. Gamboa, *PICSimLab Documentation – Temperature System Module*,  
Available: [https://lcgamboa.github.io/picsimlab\\_docs/0.9.2/TemperatureSystem.html](https://lcgamboa.github.io/picsimlab_docs/0.9.2/TemperatureSystem.html)

[4] L. C. G. Gamboa, *PICSimLab Documentation – UART IO Module*,  
Available: [https://lcgamboa.github.io/picsimlab\\_docs/0.9.2/IOUART.html](https://lcgamboa.github.io/picsimlab_docs/0.9.2/IOUART.html)

[5] L. C. G. Gamboa, *PICSimLab Documentation – Serial Communication & Virtual Ports*,  
Available: [https://lcgamboa.github.io/picsimlab\\_docs/0.9.2/SerialCommunication.html](https://lcgamboa.github.io/picsimlab_docs/0.9.2/SerialCommunication.html)

[6] L. C. G. Gamboa, *PICSimLab Documentation – 7-Segment Display Module*,  
Available: [https://lcgamboa.github.io/picsimlab\\_docs/0.9.2/7SegmentsDisplay.html](https://lcgamboa.github.io/picsimlab_docs/0.9.2/7SegmentsDisplay.html)