# Prolab2 2. Proje
# Araç Kontrol Sistemi

Yunus Emre Arı
230201096
Bilgisayar
Kocaeli
Kocaeli, Türkiye

Elifnur Caştur
210201002
Bilgisayar Mühendisliği
Kocaeli Üniversitesi
Kocaeli, Türkiye

## Giriş

Bu proje, **araç içi güvenlik ve otomatik kontrol sistemlerini** simüle eden bir Arduino Mega tabanlı çalışmadır. Gerçek bir aracın içindeki bazı önemli güvenlik ve konfor özelliklerini modellemek amacıyla hazırlanmıştır. Simülasyon ortamı olarak Proteus kullanılmıştır ve tüm sistem sanal bir ortamda test edilebilir durumdadır. Sistemin amacı, kullanıcının emniyet kemeri takıp takmadığını, aracın kapısının açık mı kapalı mı olduğunu, sıcaklık, yakıt seviyesi ve çevresel ışık gibi verileri izlemek, bu verilere göre motor çalıştırma izni vermek veya uyarılar yapmaktır. Aynı zamanda, ışık ve sıcaklık gibi çevresel koşullara bağlı olarak farların veya klimanın otomatik kontrolünü de sağlamaktadır.

## Proje Yetenekleri

Projede temel olarak bir **LCD ekran** kullanılarak kullanıcıya sürekli sistem bilgisi verilir. LCD, sıcaklık ve yakıt seviyesi gibi bilgilerle güncellenir ve gerektiğinde özel durum mesajlarını da gösterebilir. Bunun dışında sistemde çeşitli **LED'ler**, bir **buzzer**, bir **DC motor** ve analog girişler (sensor verileri) kullanılmıştır. Motorun çalışıp çalışmadığı, aracın içinde alınan güvenlik önlemlerine bağlıdır: kullanıcı emniyet kemerini takmadan ya da kapı tam olarak kapalı olmadan motoru çalıştıramaz. Ancak, motor çalıştıktan sonra kapı açılır veya emniyet kemeri çıkarılırsa motor durdurulmaz, fakat sistem sesli bir şekilde (buzzer kullanılarak) kullanıcıyı uyarır. Bu da gerçek bir araçta güvenlik açısından önemli bir özelliktir: motoru kapatmak yerine sürücüyü dikkatli olması için uyarır.

Motorun çalışması için iki temel güvenlik şartı kontrol edilir: **emniyet kemeri takılı olmalı** ve **kapı kapalı olmalıdır**. Eğer bu şartlar sağlanmıyorsa motoru başlatmak mümkün değildir. Motor butonuna basıldığında sistem, bu şartları kontrol eder; şartlar sağlanıyorsa motoru çalıştırır ve bir LCD mesajı ile kullanıcıya bilgi verir. Eğer şartlar sağlanmıyorsa, motor çalıştırılmaz ve yine bir hata mesajı LCD'de gösterilir. Sistem, kullanıcının hem emniyet kemeri hem kapı durumu değişimlerini sürekli izler ve değişiklikler olduğunda da kullanıcıya bildirim yapar.

Bunun dışında, projede bir **ışık sensörü (LDR)** kullanılarak ortam ışık seviyesi ölçülür. Eğer çevre karanlık ise (LDR'den gelen düşük değerler), aracın farları otomatik olarak açılır. Ortam yeterince aydınlık olduğunda ise farlar kapatılır. Bu farların açılıp kapanması da LCD üzerinde bilgilendirme mesajları ile kullanıcıya duyurulur. Bu sayede, sürücünün farları manuel olarak açıp kapamasına gerek kalmaz; sistem çevresel koşullara göre bunu otomatik yapar.

Ayrıca, sistemde bir **sıcaklık sensörü (LM35)** yer almaktadır. Ortam sıcaklığı belirli bir eşiğin üzerine çıkarsa (örneğin 25°C), aracın kliması otomatik olarak açılır. Sıcaklık normale dönerse klima kapanır. Bu, sıcak havalarda araç içi konforu sağlamak için tasarlanmıştır. Yine, klima açılıp kapandığında LCD üzerinde bilgilendirme mesajları gösterilir.

Yakıt seviyesi ise başka bir önemli izlenen parametredir. Arduino, yakıt sensöründen (simülasyonda potansiyometre kullanılıyor) analog bir değer okuyarak aracın yakıt yüzdesini hesaplar. Eğer yakıt seviyesi %10'un altına düşerse yakıt seviyesi LED'i yanıp sönmeye başlar, sürücüyü düşük yakıt konusunda uyarır. Yakıt %5'in altına düşerse bu kritik seviyekabul edilir ve LED yine yanıp sönerek ciddi bir uyarı verir. Yakıt seviyesi bilgisi de sürekli LCD üzerinde güncellenmektedir.

Projede kullanıcı ile etkileşim hem donanımsal (butonlar ve ledler) hem de görsel/işitsel (LCD ve buzzer) yollarla sağlanır. Kullanıcının motoru çalıştırmak için butona basması gerekir, fakat sistem sürekli bir şekilde kendi iç kontrollerini yaparak kritik durumları izler. Özellikle motor çalışırken kapı

açılırsa veya kemer çözülürse, sistem motoru durdurmaz, bunun yerine sadece buzzer ile sesli uyarı verir. Ayrıca motor çalışmaya başlarken eğer kapı açık ya da kemer takılı değilse, yine buzzer devreye girerek anında sürücüyü uyarır.

## Projenin Kodlanması

Bu projenin kodlamasında, Arduino Mega üzerinde çalışan bir gömülü sistem mantığı kullanılmıştır. Kod yapısında önce sistemde kullanılacak tüm donanım bileşenlerinin pin tanımlamaları yapılmıştır. Giriş pinleri motor butonu, emniyet kemeri butonu, sıcaklık sensörü (LM35), ışık sensörü (LDR), yakıt seviyesi sensörü ve kapı butonu gibi girişlerdir. Çıkış pinleri ise LED'ler, klima kontrolü, motor kontrolü ve buzzer gibi birimlerdir. Bu tanımlamalar sayesinde kodun ilerleyen kısımlarında her bileşen için sadece anlamlı bir isim kullanılarak okunabilirlik artırılmıştır. Ayrıca, LCD ekran için LiquidCrystal kütüphanesi kullanılarak LCD bağlantıları atanmış ve ekranın başlangıçta düzgün çalışması sağlanmıştır.

Kodun başında sistemde kullanılacak çeşitli durum değişkenleri tanımlanmıştır. Örneğin, motor çalışıyor mu, kemer takılı mı, kapı açık mı, farlar açık mı ve klima açık mı gibi bool türünde değişkenler kullanılarak sistemin her anki durumu izlenmektedir. Bunun yanında sıcaklık, yakıt seviyesi ve ışık değeri gibi analog veriler için de değişkenler tanımlanmıştır. Kodun zamanlama kısmı için millis() fonksiyonu temel alınmış, buton okuma, genel kontrol işlemleri, LCD güncelleme ve LED yanıp sönme gibi işlemler belirli aralıklarla çalışacak şekilde zamanlayıcı mantığıyla yönetilmiştir. Böylece sistem hem hızlı tepkiler verebilmekte hem de gereksiz yere işlemciyi yormadan doğru zamanda doğru kontrolleri yapabilmektedir.

Kurulum (setup) fonksiyonunda pin modları doğru şekilde ayarlanmış ve LCD ekran bir açılış mesajı ile başlatılmıştır. Ayrıca sistem açılırken LED'lerin ve buzzer'ın yanıp sönmesi sağlanarak kullanıcının sistemin çalışır durumda olduğunu görmesi için bir test sekansı yapılmıştır. Bu görsel ve işitsel geri bildirimler, gerçek bir araçta sistemin hazır olduğuna dair gösterge lambalarının yanıp sönmesine benzer bir işlev görmektedir. Kurulumdan sonra LCD temizlenerek sistem normal çalışmaya başlamaya hazır hale getirilmiştir.

Ana döngü (loop) içinde zamanlayıcılara göre farklı fonksiyonlar çağrılmaktadır. Butonlar belirli bir aralıkta kontrol edilir; özellikle motor butonunun durumu değiştiğinde yani butona basıldığında sistem önce emniyet kemeri ve kapı durumunu kontrol eder. Eğer ikisi de doğruysa motor çalıştırılır ya da durdurulur ve buna uygun bir mesaj LCD'ye yazdırılır. Eğer şartlar sağlanmıyorsa motor çalıştırılmaz ve kullanıcıya emniyet kemeri veya kapı hatası olduğunu belirten bir hata mesajı gösterilir. Bu mekanizma, motorun güvenli bir şekilde başlatılması için kodun merkezinde yer alan kritik bir kuraldır.

Bunun dışında, sürekli olarak sensörlerden veri okunur. Emniyet kemeri ve kapı durumu dijital girişlerden doğrudan okunarak değişkenlere aktarılır. Sıcaklık sensöründen gelen analog değer hesaplamaya sokularak gerçek sıcaklık değerine dönüştürülür. LDR'den gelen ışık değeri ve yakıt sensöründen gelen analog değer de benzer şekilde okunur ve ihtiyaç duyulan mantıksal değerlere çevrilir. Bu veriler sistemin davranışlarını belirlerken temel kriterlerdir. Mesela sıcaklık belli bir eşikten yüksekse klima açılır ve sıcaklık normale dönerse kapanır. Işık seviyesi düşükse farlar otomatik açılır ve ortam yeterince aydınlık olduğunda kapanır. Farların ve klimanın açılıp kapanması durumunda da sistem bunu LCD'de kısa süreli mesajlar göstererek kullanıcıya bildirir.

Motor çalıştırılması ve güvenlik kontrolleri dışında sistemde yakıt seviyesi sürekli izlenmektedir. Yakıt seviyesi %10'un altına düştüğünde yakıt LED'i yanıp sönmeye başlar. Eğer yakıt %5'in de altına düşerse LED yine yanıp sönmeye devam eder ve sürücü daha kritik bir düşük yakıt seviyesine karşı uyarılmış olur. Bu yanıp sönme işlemi millis() zamanlayıcısıyla yapılır, böylece delay gibi işlemciyi bloklayan yöntemler kullanılmadan sürekli bir şekilde sistemin diğer işlevleri de çalışmaya devam edebilir. Sistemde zamanlayıcı kontrollü bu tür non-bloklama yapılar, hem sistem güvenliğini hem de performansı artırmaktadır.

Kodun önemli özelliklerinden biri de motor çalışırken veya çalışmaya başladığı anda, eğer kapı açık ya da emniyet kemeri çıkarılmışsa motorun durdurulmaması ama sürücünün buzzer yardımıyla uyarılmasıdır. Bu gerçek araçlardaki davranışa oldukça uygundur. Motor çalışmaya devam ederken sürücüyü anlık uyararak dikkat çekmek hem güvenliği sağlar hem de motorun ani şekilde durdurulup başka sorunlar yaratmasının önüne geçer. Buzzer ile sesli uyarılar yapılırken motor kontrolü

doğrudan kesilmemektedir. Eğer motoru kapatmak isterse sürücü manuel olarak motor butonuna tekrar basmalıdır.

LCD ekran sürekli olarak güncellenmektedir. Ana bilgiler olan yakıt seviyesi ve sıcaklık bilgileri LCD üzerinde her saniyede bir güncellenir. Eğer bir olay gerçekleşirse —örneğin farların açılması, klimanın çalışmaya başlaması veya motorun çalışması gibi— LCD'de o olayla ilgili bir mesaj geçici olarak gösterilir. Bu mesajlar belli bir süre gösterildikten sonra ekran temizlenir ve standart yakıt-sıcaklık bilgisi geri gelir. Bu mesaj gösterme sistemi, kullanıcının her önemli olayı kaçırmadan fark etmesini sağlar ve sistemin çok daha profesyonel çalışmasına yardımcı olur.

Kodda tüm bu işlemler fonksiyonlara bölünerek yazılmıştır. Motor kontrolü, kemer kontrolü, kapı kontrolü, klima kontrolü, far kontrolü, yakıt seviyesi kontrolü ve LCD güncellemesi gibi görevler ayrı fonksiyonlar içinde modüler şekilde organize edilmiştir. Bu modüler yapı, kodun okunabilirliğini ve sürdürülebilirliğini önemli ölçüde artırmıştır. Gelecekte sisteme yeni sensörler, yeni güvenlik önlemleri veya yeni gösterge sistemleri eklemek istendiğinde bu yapıyı bozmak gerekmeden yeni fonksiyonlar eklenebilir. Bu da kodun ileriye dönük gelişime açık olduğunu gösterir.

## FONKSİYONLAR

**setup()**

Bu fonksiyon, Arduino'nun açılışta bir kere çalışan hazırlık bölümüdür. Burada tüm giriş ve çıkış pinlerinin modları belirlenmiştir. Girişler `INPUT_PULLUP` kullanılarak ayarlanmıştır, böylece butonlar boşta iken HIGH (1) durumunda olurlar, basıldıklarında LOW (0) olurlar. Çıkışlar için LED'ler, motor, klima ve buzzer ayarlanmıştır. Ayrıca LCD ekran başlatılmış ve açılış mesajı gösterilmiştir. LED'ler ve buzzer bir test sekansı ile birkaç kez yanıp sönerek tüm sistem bileşenlerinin doğru çalıştığını doğrulamıştır.

**loop()**
Sürekli çalışır. Butonları okur, sensörleri günceller, motor ve diğer sistemleri kontrol eder, LCD ekranı ve LED'leri yönetir. utonlar belirli bir aralıkla okunur, sensörler sürekli izlenir, motor durumu ve diğer çıkışlar yönetilir.

**butonKontrol()**
Motor butonuna basıldığında motoru çalıştırır veya durdurur. Gerekirse hata mesajı gösterir. Bu fonksiyon hem sürücünün hatalı çalışmasını önler hem de motorun güvenli koşullarda çalıştırıldığından emin olur.

**sensorOku()**
Emniyet kemeri, kapı, sıcaklık, ışık ve yakıt seviyelerini okur ve günceller. LDR'den okunan değer doğrudan ışık seviyesini temsil eder. Yakıt seviyesi ise analog okumadan sonra 0 ile 100 arasında bir yüzde değere dönüştürülür.

**motorKontrol()**
Motorun çalışması için gerekli şartların sağlanıp sağlanmadığını denetler. Bu fonksiyon sürücünün sorumluluğunu hatırlatır ama motorun işleyişini bozmaz.

**kemerKontrol()**
Emniyet kemeri takılı değilse kemer LED'ini yakar.

**kapiKontrol()**
Kapı açıksa LED'lerle uyarı verir, kapı kapalıysa LED'leri kapatır.

**klimaKontrol()**
Sıcaklık belirli bir seviyenin üstündeyse klimayı çalıştırır, değilse kapatır.

**farKontrol()**
Havanın karanlık veya aydınlık olmasına göre far LED'lerini açar veya kapatır, LCD'de bilgi verir.

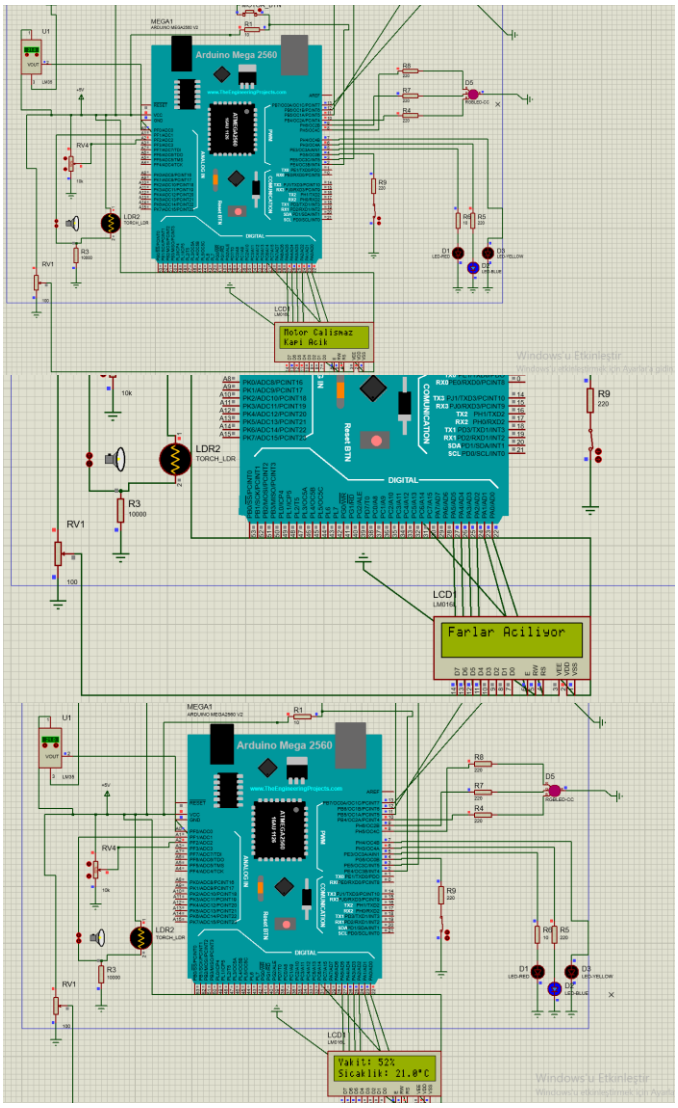
**yakitKontrol()**
Yakıt seviyesi düşükse yakıt LED'inin yanıp sönmesini sağlar.

**yakitLedKontrol()**
Yakıt seviyesi düşük olduğunda LED'in belirli aralıklarla yanıp sönmesini yönetir.

**lcdGuncelle()**
Ekranda yakıt seviyesi ve sıcaklık gibi bilgileri düzenli olarak gösterir.

**mesajGoster()**
LCD ekranda geçici bir bilgi mesajı gösterir ve belli bir süre sonra ekranı temizler.

# Örnek Görüntüler




KAYNAKLAR

[1] https://www.theengineeringprojects.com/2015/12/arduino-mega-2560-library-proteus.html

[2] https://medium.com/@erenpaslioglu/how-to-use-proteus-guide-for-beginners-31165afd78b9