# Arduıno Uzay Gemisi Oyunu Proje Raporu

Taha Furkan Genç
*Öğrenci No :210201077*

Emir Muharrem Acar
*Öğrenci No: 210201059*

Özet

Bu rapor 2022-2023 öğretim yılı Kocaeli Üniversitesi 2. sınıf öğrencileri Taha Furkan Genç ve Emir Muharrem Acar tarafından Programlama Labaratuvarı 2 dersinin 2. projesi için hazırlanmıştır. İçerikte projeden genel anlamıyla bahsedilmiş olup gerçekleştirilen eylemler, verimlilik, sunum, analiz ve projenin hedefteki istenilen amaca hizmet etme şekli ve miktarından bahsedilmiştir.

## I. Giriş

### A. Projenin Genel Tanımı

Uzay Şavaşı oyunu projesi bir uzay gemisinin engellerden kaçması veya o engelleri sınırlı mühimmatını kullanıp yok ederek gidebildiği yere kadar gitmeye çalıştığımız bir Arduino oyun projesidir. Bu proje Proteus üzerinden bir simülasyon olarak çalışır .Bu oyunun kolay mod ve zor mod olmak üzere iki farklı modu vardır.

### B. Giriş

Bu kısımda projede istenilenler ve bize verilen bilgiler doğrultusunda adımların belirlenmesi sağlanacaktır. Adımlar şöyle belirlenmiştir:

- Proteus uzerinden devreyi kurmak.
- Devre göstergelerini ayarlamak.
- Oyun alanında kullanılacak şekilleri oluşturmak
- Matris bilgilerini ekranda yansıtmak.
- Uzay gemisinin hareketini sağlamak.
- Oyun alanının hareketi.
- Yeni engel ve itemlerin oluşturulması.
- Menüyü oluşturmak.

## II. Yöntem

### A. Proteus uzerinden devreyi kurmak

Proteus üzerinden devreyi kurabilmemiz için öncelikle kullanacağımız Arduino Kartını belirlememiz gerekliydi. Biz burada 3 adet kullanılacak olan 7 Segment Display'in çok fazla pin kullanacağını düşündüğümüzden ve 8 kilobayt belleğin işimize yarayacağını düşündüğümüzden Arduino Mega 2560 kartını kullanmaya karar verdik. Devreye bir terminal ekranı eklemek geliştirme sürecinde işimize fazlaca yarayacağından dolayı Virtual Terminal eklememiz kaçınılmaz bir karar olmuştu (Fig. 1). Bu opsiyonel olarak seçebileceğimiz son unsurdu. Bundan sonrası bize verilen görevde belirtilen rapordan oluşturulmuştur. Devrede uzay gemisinin sahip olduğu can hakları ve mermi haklarının gösterilmesi için 3 er adet olmak üzere tam 6 tane LED vardır. Bu LED lambalar ,karta bir direnç ile bağlıdırlar (Fig. 2). Devre aynı zamanda 4 adet buton içermektedir. Bu 4 butonun üçü sadece menü ekranı için konumlandırılmıştır. Bu tuşlardan biri menüde yukarıdaki seçeneği seçmek için oluşturulmuş " YUKARI " butonudur. Bu tuşlardan bir diğeri de aynı " YUKARI " tuşu gibi aşağıdaki seçeneği seçmek için oluşturulmuş " ASAGI " tuşudur. Bunlardan bir diğeri de menü ekranı için oluşturulmuş son buton olan " SECIM " butonudur. Bu butonun görevi de o an için seçilmiş olan modu oyuna gönderip oyunumuzun başlamasını sağlamasıdır. Bu butona basarak oyundan ayrılırız. Bu 3 tuş tamamen menüde kullanılan tuşlardı. Bunlardan ayrı 1 tuşumuz daha vardır. Bu tuş ise oyun anında kullanılabilir. Bu tuşumuz " ATES " tuşudur. Bu tuşun görevi oyun anında uzay gemisinden ateşleme emrini ilgili kod parçasına gönderip eğer uygunsa ve muhimmat varsa ateşlenmeyi gerçekleştirmektir(Fig. 3).

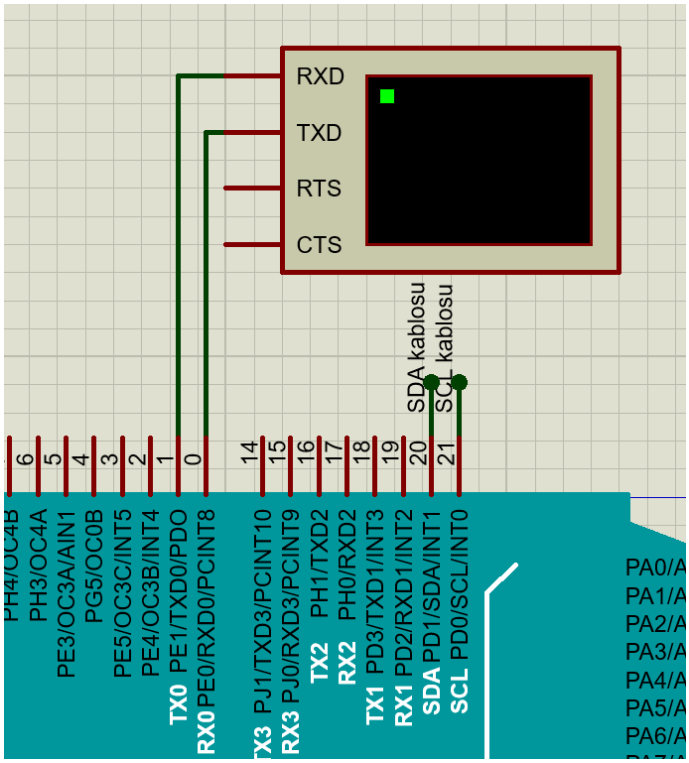


Fig. 1. Virtual terminal

### B. Devre elemanlarını ve göstergelerini ayarlamak

Devreye giriş olarak 3 farklı türde eleman kullanıyoruz. Bunlar tuş , potansiyometre ve LDR sensörüdür. Bütün bunlar devreye bağlanırken direnç ile akımı düzenlenmiştir ve güç verilmiştir. Bütün girişler ve çıkışlar setup() fonksiyonunda

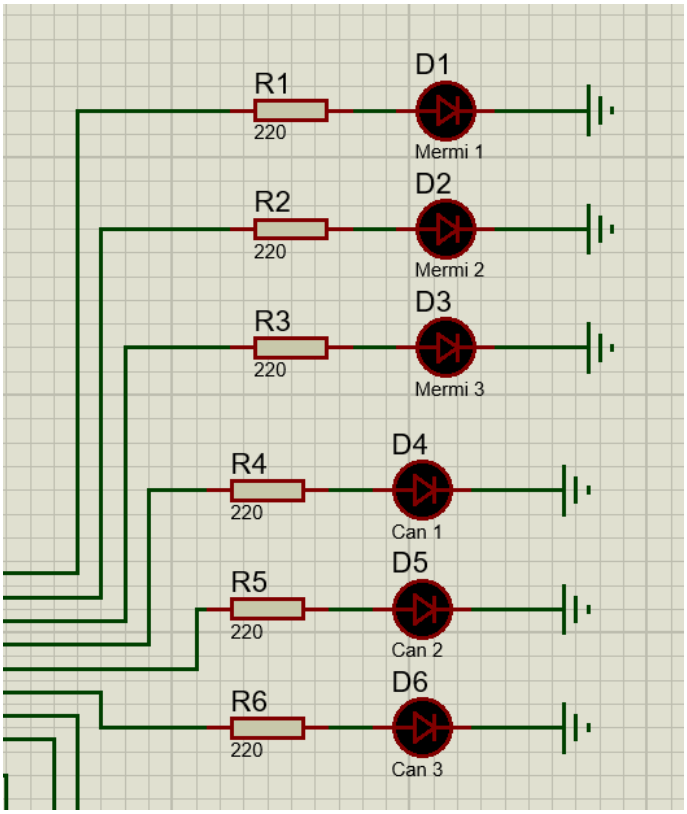


Fig. 2. Ledler

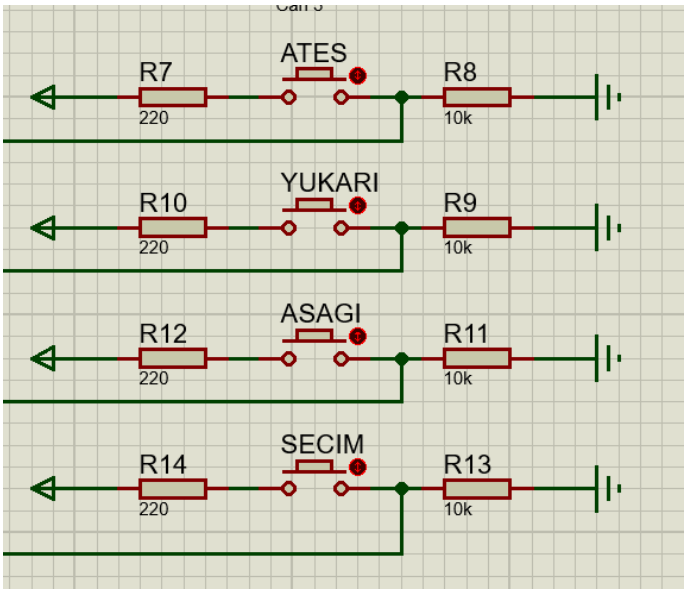


Fig. 3. Butonlar

tanımlanmıştır. Devrede kullandığımız her elemanın değeri de define kullanılarak kodda tutulmuştur. Define kullanmamızın sebebi arduinoda gereksiz bellek kullanımının önüne geçmekti. Devrede işitsel ve görüşsel çıkışlar vardır. İşitsel çıkışımız oyunda engele çarpıp can kaybettiğimiz zaman bize uyarı verir . Görüşsel çıkışlarımızdan biri Ledlerdir. Bu ledleri bize can hakkımızı ve silah hakkımızı gösterir. Bu gösterimi oluşturmak için canhakkigoster() ve silahhakkigoster() olmak üzere 2 ayrı fonksiyon oluşturduk. Bu fonksiyonlar açılacak led sayısını alır ve 4 farklı sorgulama komutu ile ledleri yakar.Görüşsel çıkışlarımızdan biri ise 7 Segment Display'dir(Fig. 4). 7 Segment Display için de ayrı bir fonksiyon yazdık. 7 Segment Display için yazdığımız fonksiyonun mantığı LED göstergeler için yazdığımız fonksiyonlar ile aynı mantıktadır. 10 adet rakam için ayrı ayrı sorgulama blokları oluşturup bu bloklara göre 7 Segment Display'in segmentlerine elektrik verdik. Dışardan gireceğimiz puan durumunun da üç basamaklı sayı olma ihtimalini bildiğimizden dolayı ayrı bir fonksiyon daha oluşturduk ve 1 adet 7 Segment Display kontrol ettiğimiz fonksiyonu bu fonksiyonda 3 kere çağırdık. Alınan parametredeki sayıyı rakamlarına ayırıp buna göre oluşan 3 farklı int değişkenindeki rakamları sırasına göre 7 Segment Display'de gösterebiliyoruz.Görüşsel çıkışlarımızdan biri ise OLED ekrandır. OLED ekran tüm oyunu görebileceğimiz , devrenin en önemli unsurudur.OLED ekranın tanımlanırken bir kontrol şeması ile bellek kontrolü sağlarız. Eğer bu bellek kontrolü olumsuz sonuç döndürürse kodumuz çalışmaz. Çalışsa bile ekranı göremeyiz.

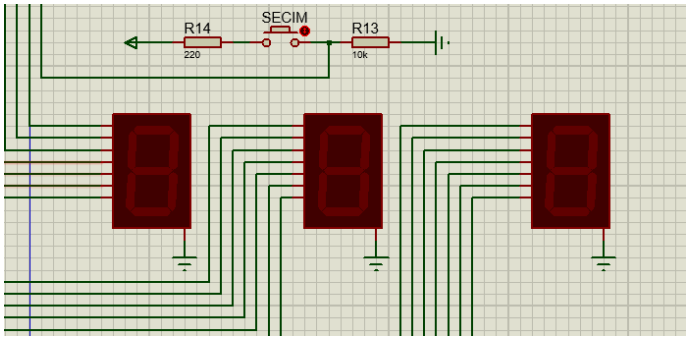


Fig. 4. 7 segment Display

### *C. Oyun alanında kullanılacak şekilleri oluşturmak ve Matris bilgilerini ekrana yansıtmak*

Oyun alanında kullanılacak elemanları belirlerken bitmap yerine bitmap mantığında kendi fonksiyonlarımızı yazdık(Fig. 5). Oyun alanının tutulduğu ama etmen olan matrisimiz 8X16 boydadır. Oyun ekranı her loop() fonksiyonunun başında temizlenir ve sonunda güncellenir. Bu arada ise her adımda matrisin ekrana yansıtılması gereklidir. Bunu yaparken her loopta iç içe for döneriz ve bu tüm matrisi döner. Bu matris dönerken her matris elemanında kontrol (sorgulama) blokları çalışır.Eğer mevcutta olan durum 0 ise o bölgenin koordinatları alınan parametreye göre hesaplanır ve sol üst köşeden başlayarak uzay boşluğu çizilir. Bu uzaybosluguciz() fonksiyonunun çalışma mantığıdır.Eğer mevcutta olan durum 1 ise o bölgenin koordinatları alınan parametreye göre hesaplanır ve sol üst köşeden başlayarak mermi çizilir. Bu mermiciz() fonksiyonunun çalışma mantığıdır. Eğer mevcutta olan durum 2 ise

o bölgenin koordinatları alınan parametreye göre hesaplanır ve sol üst köşeden başlayarak uzay gemisi çizilir. Bu uzaygemisiciz() fonksiyonunun çalışma mantığıdır. Eğer mevcutta olan durum 3 ise o bölgenin koordinatları alınan parametreye göre hesaplanır ve sol üst köşeden başlayarak uzay çöplüğü çizilir. Uzay çöplüğü pixellerin en fazla yarısını rastgele yakar ve diğerlerini bırakır(Fig. 6). Bu uzaycopluguciz() fonksiyonunun çalışma mantığıdır. Eğer mevcutta olan durum 4 ise o bölgenin koordinatları alınan parametreye göre hesaplanır ve sol üst köşeden başlayarak mermi ile hasar almış meteorun görseli çizilir. Bu hasarlimeteorciz() fonksiyonunun çalışma mantığıdır. Eğer mevcutta olan durum 5 ise o bölgenin koordinatları alınan parametreye göre hesaplanır ve sol üst köşeden başlayarak hasar almamış meteorun görseli çizilir. Bu hasarsizmeteorciz() fonksiyonunun çalışma mantığıdır. Eğer mevcutta olan durum 6 ise o bölgenin koordinatları alınan parametreye göre hesaplanır ve sol üst köşeden başlayarak Can hakkı çizilir. canhakkiciz() fonksiyonunun çalışma mantığıdır. Eğer mevcutta olan durum 7 ise o bölgenin koordinatları alınan parametreye göre hesaplanır ve sol üst köşeden başlayarak Dokunulmazlık hakkı çizilir. Bu dokunulmazlikhakkiciz() fonksiyonunun çalışma mantığıdır.Analog girişe bağlamış olduğumuz LDR sensöründen gelen değişkene göre oyun alanının aydınlık ve karanlık durumu değiştirilir. Eğer LDR den gelen değere göre ışık çok ise matriste 0 yazan yerler aydınlatılıp diğer pixeller karartılır. Eğer LDR den gelen değere göre ışık az ise matriste 1 yazan yerler aydınlatılıp diğer pixeller karartılır(Fig. 7).

### *D. Uzay gemisinin hareketini sağlamak*

Uzay gemisinin hareketi potansiyometre ile alınır(Fig. 8). Potansiyometre analog giriş veren bir eleman olduğundan dolayı Arduino kartında Analog girişlere bağlıdır. Analog girişlerde 0 dan 1027 ye olmak üzere 1028 aralıklı girişi olduğundan dolayı 128 aralıklı giriş destekleyen dijital pinlere uyum sağlamaz ve bu yüzden map fonksiyonunu kullanırız. Fakat bu aralıktan daha da düşük bir aralık kullanmamız gerektiğinden bu durum bizim için sorun olmadı. Map fonksiyonunu kullanarak Potansiyometreden aldığımız 1028 aralıklı girişi 0 ile 7 arası olmak üzere 8 aralıklı girişe düşürdük ve bunu tam tersi hale getirip uzay gemimizin olacağı konumun satır koordinatını ayarlamış olduk. Uzay gemimiz zaten son sütundan ileriye gitmeyeceğinden dolayı konum için daha fazla girdiye ihtiyacımız kalmadı ve son sütunun alınan satırını uzay gemisi anlamına gelen 2 rakamı ile güncelleyerek oyunda aktif olarak uzay gemisini gösterebildik. Bu işlemlerin her seferinde tekrar edebilmesi için delay kullanmadık. Delay kullanımı tuşların aktifliğini engellediğinden dolayı millis() kullanımı daha mantıklıydı. millis() fonksiyonu ile en son kaydırma işlemi arasında olan fark bize bu süreyi verir. Eğer bu fark bekleme süresi ile aynıysa veya daha fazlaysa bu if çalışır ve en son kaydırma işleminin tutulduğu değişken millis ile güncellenir.

```
void uzaybosluguciz(int matrisI,int matrisJ)
{
  int matris[8][8] = {
   {0, 0, 0, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 1, 0, 0},
   {0, 0, 0, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0, 0, 0},
   {0, 0, 1, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0, 0, 0}
  };
  for (int row = 0; row < 8; row++) {
    for (int col = 0; col < 8; col++) {
     if (matris[col][row] == 1) {
        if(ldrvalue>500){
            myOLED.setPixel((matrisI*8)+col,(matrisJ*8)+row); // karanlık
        }
        else {
            myOLED.clrPixel((matrisI*8)+col,(matrisJ*8)+row); //aydınlık
        }

     }
     if (matris[col][row] == 0) {
       if(ldrvalue>500){
         myOLED.clrPixel((matrisI*8)+col,(matrisJ*8)+row);
       }
       else{
         myOLED.setPixel((matrisI*8)+col,(matrisJ*8)+row);
       }

     }
    }
  }
}
```

Fig. 5. Uzay boslugu çiz fonksiyonu

```
};
  for(int i=0; i<32;i++){
    int randomi=random(8);
    int randomj=random(8);
    if(matris[randomi][randomj]==0){
      matris[randomi][randomj]=1;
    }
  }
```

Fig. 6. Uzay çöplügünün random oluşması

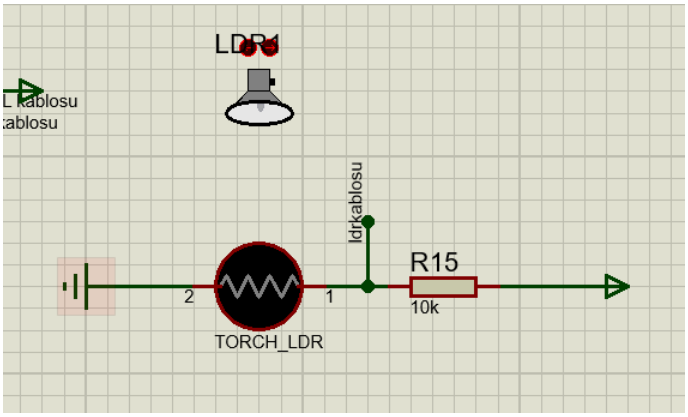


Fig. 7. LDR

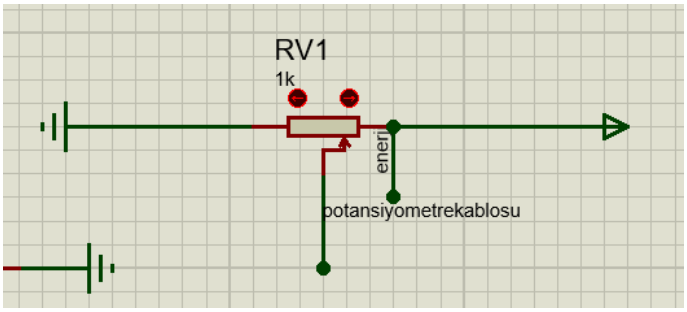


Fig. 8. Potansiyometre

### *E. Oyun alanının hareketi*

Oyunun durumlarının tutulduğu ana matrisimizin olduğunu yazmıştık. Bu matrisimiz oncekimatris[][] matrisidir. Bundan ayrı bir matrisimiz daha bulunmaktadır. Bu matrisimizin adı da sonrakimatris[][] idir. Bu sonrakimatris[][] matrisimiz , adı üstünde , oyunun bir sonraki seferde oluşacak matrisini tutacaktır. Bu matris oluşmadan önde bir adet dizi oluşturulur. Bu dizi oyuna yeni eklenecek engellerdir. Bu engellerin oluşacağı noktalar random() fonksiyonu ile oluşturulur. eğer random fonksiyonundan gelen değişken uzay boşluğundan farklı ise engel konulduktan sonra yerleştirilecek olan nesnenin uzay boşluğu olması zorunludur. Her nesne oluşumunda for döngüsünün sayaç değişkeni bir kere daha artar. sonrakimatris[][] oluşturulurken iç içe for döngüsünün içinde bir kontrol yapılır. sonrakimatris[][] fonksiyonun birinci sütununda ise o alana oluşturduğumuz bu dizinin sonrakimatris[][] in olduğu satırın indisindeki değer verilir. Eğer birinci sutunda değilse sonrakimatris[][] e verilen değer oncekimatris[][] değerinin bir sağa kaymış halidir. Bu sayede oyunda aktif hareket ve engel oluşturma durumları elde edilir. oyun alanı 1 birim sağa kayarken eğer bir mermi varsa o bir birim sola kayar. Bağıl hız nedeniyle her adımda 2 kere kaydırmamız gerek fakat birleşme anımızı oluşturmak için bu hareketi 2 parçaya bölmemiz gerek. İlk adımda merminin bir soluna bakarız. Eğer tek canı kalmış meteor veya uzay çöplüğü varsa mermi kendini ve bu engeli yok eder. Aynı durum Dokunulmazlık hakkı ve Can hakkı için de geçerlidir. Eğer iki canlı meteor varsa mermi kendini yok eder ve iki canlı meteoru tek canlı meteor ile değiştirir. Eğer uzay boşluğu varsa mermi bir ileri daha gider ve yine bir kontrol yapar. Bu da bu aşamaların ikinci adımıdır. Burada da eğer bir tek canlı engel veya hak karakterleri varsa onu ve kendisini yok eder. Eğer iki canlı engel varsa yine kendisini yok eder ve engelin mertebesini düşürür. Eğer uzay boşluğu varsa bir ileri gider. Uzay boşluğu bir ileri giderken şu mantığı kullanır : matrisin içinde olduğun konumun bir sağını da mermi ( yani ” 1 ”) yapar ve önceki olduğu yeri uzay boşluğu ( yani ” 0 ” ) ile değiştirir(Fig. 9)(Fig. 10).

### *F. Dokunulmazlık ve can karakterleri*

Dokunulmazlık hakkı alındığında oyunumuzdaki uzay gemisinin 3 saniye boyunca bir yere çarpsa bile can hakkından eksilme yaşamaması gerekiyor. Fakat bir yandan oyunun da akması gerektiğinden dolayı oyun ekranının kaydırılma

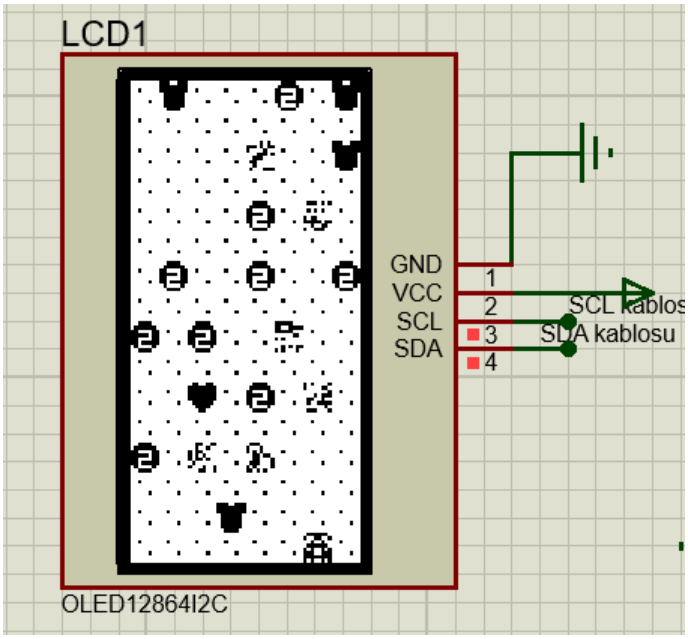


Fig. 9. Aydınlık mod

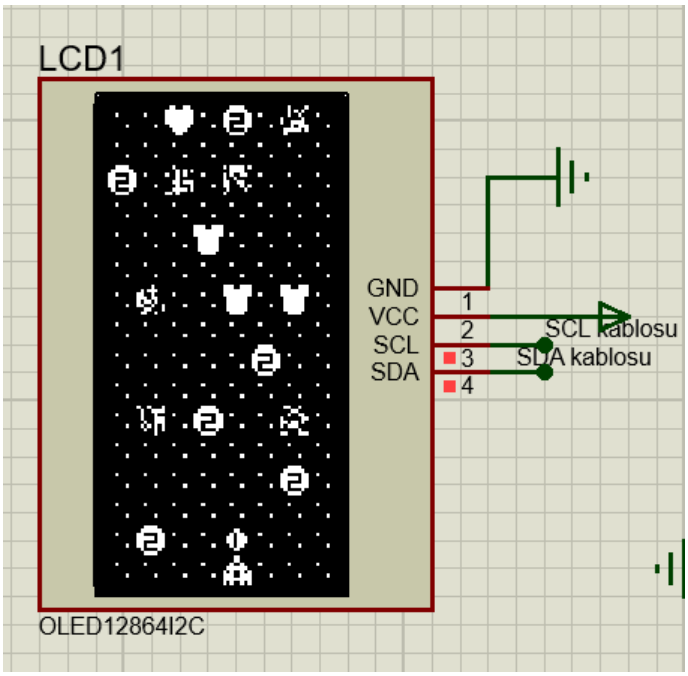


Fig. 10. Karanlık mod

mantığı gibi burada da delay kullanmadık. millis() fonksiyonu ile en son kaydırma işlemi arasında olan fark bize bu süreyi verir. Eğer bu fark dokunulmazlık süresinden daha az ise bu if çalışır ve dokunulmazlıkvarmi boolean değişkeni true olarak kalır. Eğer bu if çalışmazsa else sekmesi çalışır ve tam tersi olarak dokunulmazlikvarmi boolean değişkeni false olarak çevrilir. Eğer dokunulmazlikvarmi değişkeni true ise uzay gemisine bir kalkan eklenir. False ise kaldırılır

```
if(dokunulmazlikvarmi){ // dokunulmazlık varsa kalkan ekle
  for(int i=1;i<7;i++){
    matris[0][i]=1;
  }
  matris[1][0]=1;
  matris[1][7]=1;
}
```

Fig. 11. Kalkan oluşması

### G. Menüyü oluşturmak

Menü ekranının gelip gelmeyeceği bir boolean değişken ile belirlenir. Eğer boolean değişken true ise menü ekranı gelir. Bu menü ekranının oluşturulduğu fonksiyon olan menuekrani() fonksiyonu çalışır. Bu fonksiyonda da bir loop fonksiyonu eklememiz gerekiyordu. Bu durumu while(1) ile hallettik. Döngünün başında ekranı temizleyip sonunda güncelledik ve arasına kodlarımızı yazdık. Daha öncesinde oluşturduğumuz oyunzormoddami boolean değişkenine göre menü ekranının görüntüsünü değiştiriyoruz. Eğer oyunzormoddami boolean değişkeni true ise menüdeki ” Zor Mod ” yazısı yerine ” -¿ Zor Mod ¡- ” yazıyoruz. Kolay mod ifadesinde ise ok kullanmıyoruz. Eğer oyunzormoddami boolean değişkeni false ise menüdeki ” Kolay Mod ” yazısı yerine ” -¿ Kolay Mod ¡- ” yazıyoruz. Zor mod ifadesinde ise ok kullanmıyoruz. Bu kontrol de if else bloğu ile tutuluyor. Eğer Seçim tuşuna tıklanırsa while döngüsü break komutu ile kırılır ve o sırada hangi mod seçilmişse oyun o mod ile başlar. Tüm değişkenler varsayılan değerleri ile güncellenir. Matris tamamen temizlenir. Menü ekranında ayrıca oyunda oynanılan en son oyunun puanı da gösterilir. Eğer bu değer 0 ise arduino daha yeni çalıştırılmış demektir(Fig. 12).

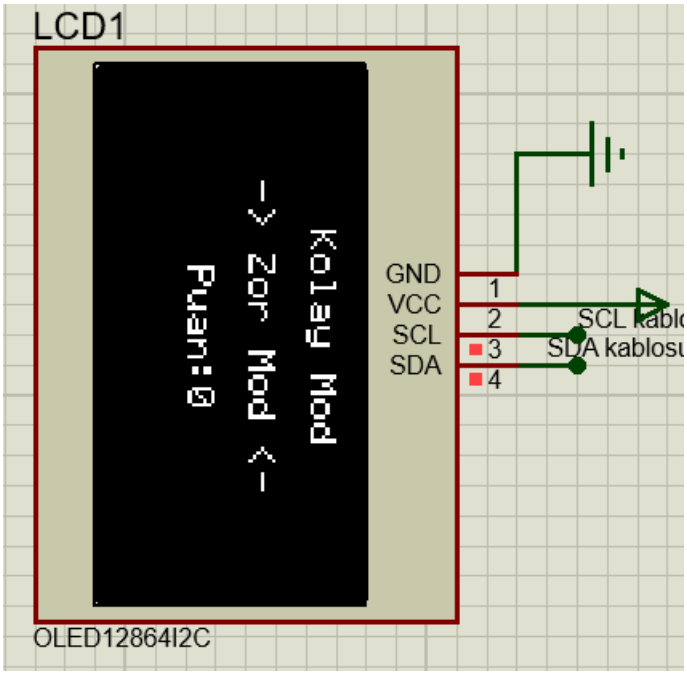


Fig. 12. Menü ekranı

## III. Sonuç

### A. Programın Yapabildikleri ve İşlevsellikler Hakkında

Programın yöntemler ve kod ifadeleriyle de anltıldığı üzere yerine getirilmesi istenen işlevlerin büyük kısmını yapabilir olup bunlar : iki farklı oyun modunu ayarlamak, menü seçim tuşları ve ateş tuşlarını ayarlayabilmek, potansiyometre ile uzay gemisi hareketi sağlamak, can ve mermi ledlerini doğru şekilde ayarlayabilmek, 7 Segment Display üzerinden aktif olarak puan gösterebilmek, Engele çarpıldığı zaman sesli uyarı verebilmek, oyunda kaybedildiği zaman ana menüye geri dönebilmek, LDR sensörlerine göre renk dağılımını değiştirebilmek ve bütün bunları SSD1306 OLED ekranında gösterebilmektir(Fig. 13).

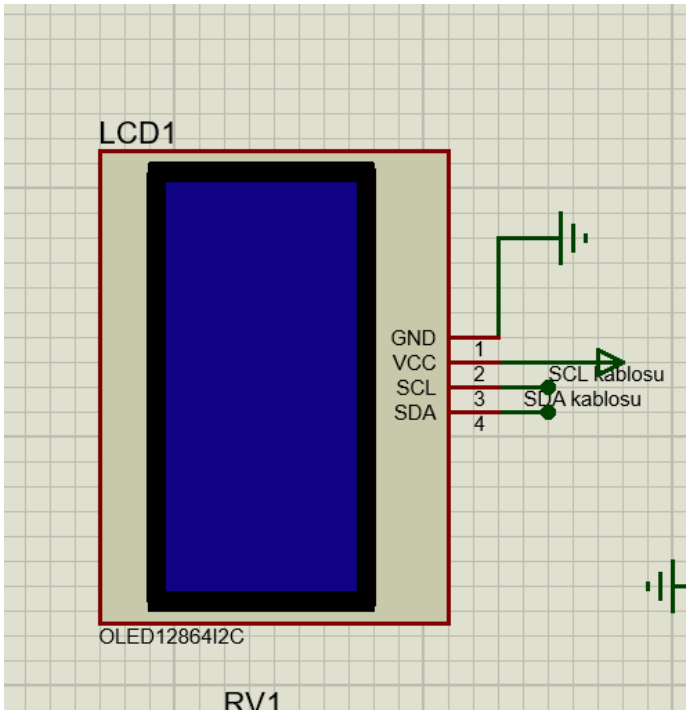


Fig. 13. Oled ekran

## IV. Deneysel Çıktılar

### A. Deneysel Çıktıların Açıklamalarıyla Verilmiş Halleri

## V. Kapanış ve Kaynakça

Çıktılar yüklenen dosya üzerindeki işlemlerden alınmıştır.

### A. Kaynakça


https://reference.arduino.cc/reference/en/language/functions/time/millis/

https://forum.arduino.cc/t/will-the-final-release-of-arduino-ide-2-0-support-multiple-cursor-editing/873493/4

https://www.ctrlbizde.com/index.php/egitimler/arduino-programlama/item/632-arduino-da-rastgele-sayi-uretme-random-randomseed-arduino-programlama-11

https://www.devrelerim.com/2022/11/arduino-oled-ekran-kullanimi-ssd1306.html

https://randomnerdtutorials.com/guide-for-oled-display-with-arduino/

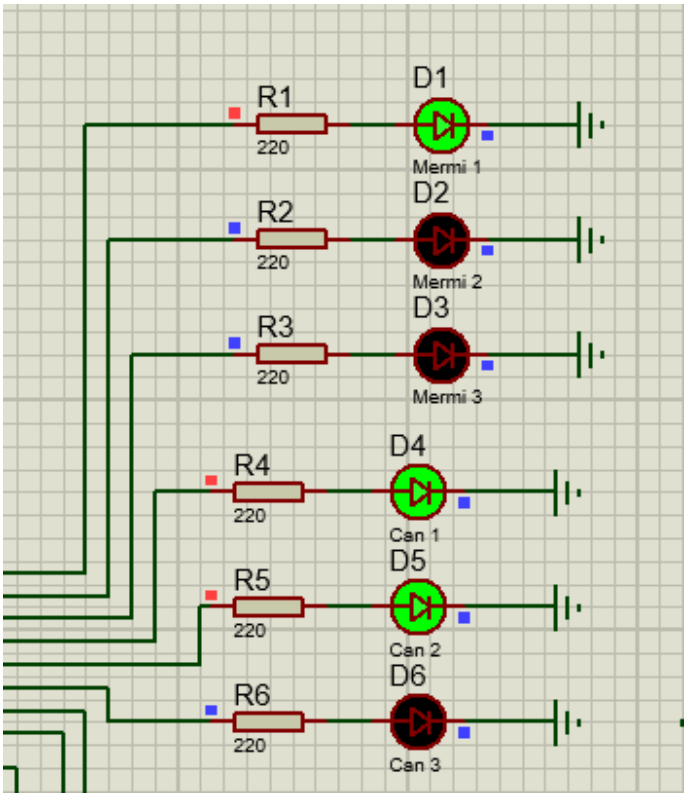


Fig. 14. Ledler

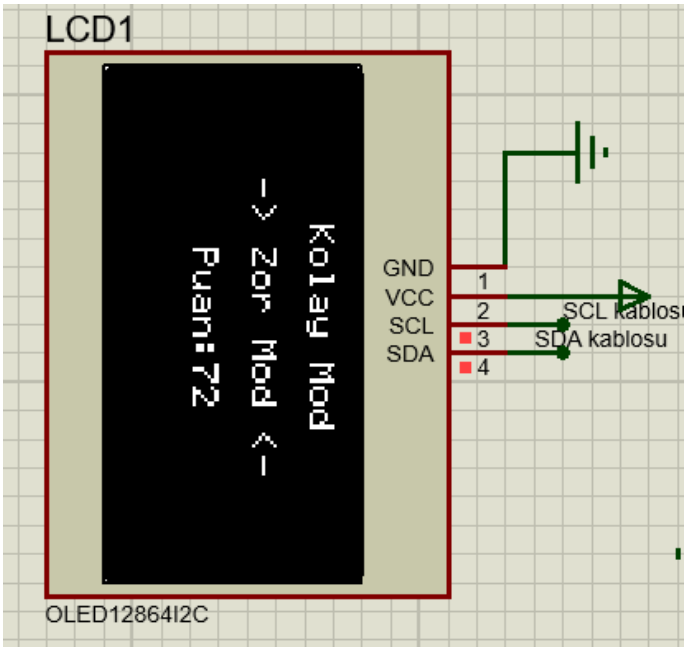


Fig. 15. Menü ve puan

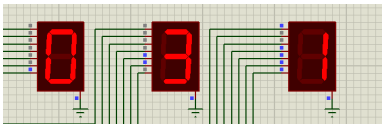

Fig. 16. 7 segment puan gösterimi

https://www.youtube.com/watch?v=UmYiHTOz-5k

https://www.youtube.com/playlist?list=PLFOSdDqm35feuJXiWzI64TAmnLJ0y29-OcMH-xEGYuDCz

(Bazı kaynakların URL'sinde LaTeX komutlarını çalıştıran karakterler olduğundan dolayı rapora eklenememiştir.)

https://www.youtube.com/playlist?list=PL1J0y2v7mkQzbJFmnxI-