UZAY OYUNU

1st Yalçın Dağbaşı
Kocaeli üniversitesi

Kocaeli/Türkiye
210202040

2nd Emir Çınar
Kocaeli üniversitesi

Kocaeli/Turkiye
210202014

I. ÖZET

Bu projenin amacı, çeşitli sensörler kullanarak mikrodenetleyici tabanlı bir oyun makinesi geliştirmektir.

II. PROJE TANIMI

Aşağıdaki isterlere göre mikrodenetleyici tabanlı oyun makinesi yapılması istenmiştir

III. İSTERLER

- Oyun ekranı açıldığında menüde başlat ve zorluk seviyesi olmak üzere 2 seçenek olacaktır. Kullanıcı oyuna 3 can hakkı ile başlayacaktır. 1. zorluk seviyesinde platform her zaman saniyede 1 kare aşağıya doğru hareket edecektir. 2. zorluk seviyesinde platform her 10 saniyede bir sağlayacak ve seçim işlemi gerçekleştirecek 3 buton kullanılması beklenmektedir.
- Başlat düğmesine bastıktan sonra oyun başlayacaktır. Uzay aracı matrisin 1.satırında olacak şekilde sağ ve sola hareket edebilecektir. Sağa ve sola hareket için potansiyometre kullanılarak yeni bir oyun kolu tasarlanması beklenmektedir.
- Oyun başladığında kullanıcıda 3 silah hakkı olacaktır. Bunlar led ışıkla gösterilecektir. Kullanıldığında sayı 1 azalacak ve buna göre ledlerden biri sönecektir. Hak tekrar kazanıldığında buna göre led yanacaktır. 3 adet led kullanılması beklenmektedir. Atış yapabilmek için ekstra buton kullanılacaktır.
- Kullanıcın hakları led ışıklarla aynı şekilde gösterilecektir. Uzay aracı engele çarpıp can kaybettiğinde otomatik olarak 3 saniye engele çarpsa bile 2.kere can kaybetmeyecektir. Can hakkı kazandığında led ışıkta artma olacaktır. Bu ledler silah hakkındaki ledlerden hariç tasarlanmalıdır. Ayrıca uzay aracı engele çarptığında buzzer kullanılarak uyarı verilecektir.
- Oyuncunun tüm hakları bittiğinde sistem tekrar ana menüye dönecektir.
- Oyuncu her satır atladığında skor puanı 1 artacaktır ve skor puanı 7 segment display ile gösterilmelidir. Skor tablosu için 3 adet 7 segment display bulunmalıdır.
- Arduino LDR ışık sensörü kullanılarak oyun ortamının renk değiştirmesi beklenmektedir. LDR ışık sensöründen alınan verilere göre oyundaki siyah beyaz renk dağılımı değiştirilecektir.
- Rastgele oluşturulan nesneler için farklı şekiller belirlenecektir. Buna imkan sağlayacak bir OLED ekran kullanılması beklenmektedir. SSD1306 gibi ekranlar tercih edilebilir.

Fig. 1. 7 SEGMENT DİSPLAY

IV. VOID SETUP()

Bu fonksiyon, Arduino'nun kurulum sırasında sadece bir kez çalıştırılır ve cihazın giriş/çıkış pinlerinin başlangıç durumunu ayarlar. Bu fonksiyon aynı zamanda, OLED ekranın başlatılması ve varsayılan yazı rengi ayarlanması gibi işlemleri de yapar. Fonksiyonda, öncelikle OLED ekranın başlatılması ve ekranın temizlenmesi için display.begin(SSD1306 SWITCHCAPVCC, OLED ADDR) ve display.clearDisplay() fonksiyonları kullanılır. Daha sonra, oyunda kullanılacak pinlerin modları belirlenir. pinMode() fonksiyonu kullanılarak BT LEFT, BT RIGHT ve BT FIRE butonlarının giriş, LED BLT1, LED BLT2, LED BLT3, LED HP1, LED HP2 ve LED HP3 ledlerinin çıkış olarak belirlenmesi yapılır. Oyunun ana karakteri olan geminin başlangıç pozisyonu belirlenir. ship pos x ve ship pos y değişkenleri, geminin 0. satır ve 4. sütundaki pozisyonunu belirler. Son olarak, OLED ekran üzerinde, gemi karakterinin varsayılan renk (siyah) ve boyut (1x) ile konumlandırılması için ilgili işlemler yapılır. displaysetTextSize(1) fonksiyonu, yazı boyutunu belirler. display.setTextColor(BLACK) fonksiyonu, yazı rengini siyah olarak belirler. display.setCursor() fonksiyonu ile geminin konumlandırılması yapılır. Ardından display.println(ship) fonksiyonu kullanılarak, gemi karakteri

OLED ekrana yazdırılır. Bu işlemler tamamlandıktan sonra, delay(1000) fonksiyonu ile cihaz bir saniye boyunca bekletilir.

V. VOID RESTART()

- Bu fonksiyon, oyunun bitmesi durumunda tüm değişkenlerin ve matrislerin başlangıçtaki değerlerine geri döndürülmesi için kullanılır. Bu işlemler aşağıdaki sıra ile yapılır.
- ship pos x ve ship pos y değişkenleri sıfırlanarak gemi başlangıç pozisyonuna getirilir.
- OLED ekranında geminin başlangıç pozisyonuna imleç konulur.
- life, ammo ve skor değişkenleri sıfırlanır.
- game matrisi temizlenir, yani tüm elemanları sıfırlanır.
- gameState değişkeni ana menüye dönmek için sıfırlanır.
- mainMenu() fonksiyonu çağrılarak oyun yeniden başlatılır.

VI. VOID MOVESHIP()

moveShip fonksiyonu, uzay gemisinin hareket etmesini kontrol eder. Uzay gemisi projede verilen direktifler doğrultusunda potansiyometre ile sağa veya sola hareket edebilecek şekilde ayarlanmıştır. Fonksiyon aynı zamanda geminin mevcut pozisyonunu ve yöneği tutar ve kullanıcının hareket etmek için sol veya sağ düğmeleri basılı tutup tutmadığını kontrol eder. Ayrıca, ilgili butona tıklandığında gemiden bir mermi fırlatılması işi de bu fonksiyonda yapılmıştır.

VII. VOID SPAWNMETEOR()

Bu fonksiyon en basit anlamıyla oyuna matriste rastgele bölgelere çöp üretmesi için oluşturulmuştur. Çöp ve meteorların oluşturulma sıklığını da rastgele bir şekilde belirlemek için meteorRate ve junkRate değişkenleri tanımlanmıştır. Bu değişkenler, meteorların ve çöplerin çıkma olasılığını kontrol eder. Bu olasılıklar, rastgele bir sayı oluşturarak kontrol edilir. Bu işlemler aşağıdaki algoritma ile yapılır. İlk olarak, 0-99 arasında rastgele bir sayı üretilir ve bu sayı meteorRate değişkeninden küçükse meteor belirir, değilse çöp belirir. Eğer belirlenen sayı meteorRate'ten küçükse, 0-7 arasında rastgele bir sayı daha seçilir ve bu sayı kullanılarak oyun matrisinde 15. sıradaki ilgili sütuna meteor eklenir Eğer belirlenen sayı meteorRate'ten büyük veya eşit ama junkRate'ten küçükse, aynı şekilde 0-7 arasında rastgele bir sayı seçilir ve bu sayı kullanılarak oyun matrisinde 15. sıradaki ilgili sütuna çöp eklenir Eğer belirlenen sayı hem meteorRate'ten büyük veya eşit hem de junkRate'ten küçük değilse, hiçbir engel eklenmez ve fonksiyon bitirilir.

VIII. VOID MOVEBLOCK()

Bu fonksiyon, oyun alanındaki engellerin hareketini kontrol eder. Yani, her adımda tüm engelleri bir adım ileri taşır. Fonksiyon, yukarıdan aşağıya hareket eden bir blok şeklindeki engelleri temsil eder.

Fonksiyon, iki parametre alır: y ve x. y, engelin dikey konumunu, x ise yatay konumunu temsil eder.

İlk koşulda, engel oyun alanının sonuna geldiğinde game matrisindeki ilgili konum sıfırlanır ve puan (skor) bir arttırılır. Ayrıca belirli skorlarda , oyuncunun canı ve mermi sayısı yenilenir (addAmmoHP() fonksiyonu tarafından kontrol edilir).

İkinci koşulda, engel gemiye çarptığında, buzzer ile birlikte bir yanıp sönen işlemi gerçekleştirilir ve oyuncunun canı (life) bir azaltılır.

Üçüncü koşulda, engel mermiye çarptığında, hem engelin hem de mermi matrisindeki konumu sıfırlanır.

Son olarak, eğer engel bir meteor veya uzay çöpü ise, bir sonraki adıma ilerletilir

IX. VOID MOVEBULLET()

Bu fonksiyon, bir mermiyi hareket ettirmek ve oyun alanında sürmesini sağlamak için oluşturulmuştur. Fonksiyon, oyun alanının dışına çıkıp çıkmadığını kontrol eder. Eğer çıktıysa, mermiyi oyun alanından siler. Eğer mermi bir engelle (meteor veya uzay çöpü) çarparsa, engeli ve mermiyi oyun alanından siler. Aksi takdirde, mermiyi bir sonraki pozisyona taşır ve oyun alanındaki bu pozisyonda bir mermi olduğunu işaretler. Bu fonksiyon x ve y parametrelerini girdi olarak alır. Ayrıca oyun alanının matris biçimi olan game degiskeni burada kullanılmıştır. fonksiyonun ilerleyişi şu şekilde olmuştur :

- İlk olarak, mermi oyun alanının sonuna (x=15) geldiyse, game matrisindeki bu pozisyonu sıfırlayarak mermiyi oyun alanından siler.
- Mermi bir engelle (meteor veya uzay çöpü) çarparsa, game matrisindeki hem mermi hem de engel pozisyonunu sıfırlayarak her ikisini de oyun alanından siler.
- Eğer mermi, engelle çarpmadıysa, game matrisindeki mermi pozisyonunu sıfırlar ve bir sonraki pozisyona taşır.

X. VOID PRINTALL()

Bu fonksiyon basitçe tüm her şeyin yazdırılması durumunu ifade eder. Fonksiyon, bir döngü kullanarak 8 satır ve 16 sütundan oluşan bir matrisi dolaşır. Matrisin her bir elemanı, oyundaki farklı nesneleri temsil eder.

Döngü her bir elemana ulaştığında(meteor,çöp,mermi vs.) display.setCursor() fonksiyonu ile ekrandaki imlecin konumunu belirler ve switch-case bloğuna girer. Bu blok, matristeki elemanın değerine göre farklı işlemler yapar.

- Eğer matristeki elemanın değeri 0 ise (yani herhangi bir nesne yoksa), hiçbir işlem yapılmaz ve döngü devam eder.
- Eğer matristeki elemanın değeri 1 ise (yani oyuncunun gemisi varsa), display.print() fonksiyonu ile gemi karakteri ekrana yazdırılır.
- Eğer matristeki elemanın değeri 2 ise (yani meteor varsa), moveBlock() fonksiyonu çağrılır ve meteor karakteri ekrana yazdırılır.
- Eğer matristeki elemanın değeri 3 ise (yani atık varsa), moveBlock() fonksiyonu çağrılır ve atık karakteri ekrana yazdırılır.
- Eğer matristeki elemanın değeri 4 ise (yani oyuncunun mermisi varsa), moveBullet() fonksiyonu çağrılır ve mermi karakteri ekrana yazdırılır. Bu durumda, sütun sayısı 1 artırılır, çünkü mermi bir sütun ilerlemiş olur.

Döngü tamamlandıktan sonra, ekrandaki skor ve oyuncunun kalan can sayısı (life) yazdırılır.

XI. VOID CHECKLEDS()

Bu fonksiyon en basit haliyle can,mermi vs. değerlerin ledlere göre yakılması için oluşturulmuştur. İlk olarak, switch case yapısı kullanılır. Ammo değişkeninin değeri sıfırsa, tüm mermi LED'leri kapatılır. Eğer ammo değişkeni bir ise, yalnızca LED BLT1 açılır. Ammo değişkeni iki ise LED BLT1 ve LED BLT2 açılır. Ammo değişkeni üç ise tüm mermi LED'leri açılır. Can sayısı için de aynı şey geçerlidir. Life değişkeni sıfırsa, tüm can LED'leri kapatılır. Life değişkeni bir ise, yalnızca LED HP1 açılır. Life değişkeni iki ise LED HP1 ve LED HP2 açılır. Life değişkeni üç ise tüm can LED'leri açılır. Yani en nihayetinde oyuncunun özelliklerinin durumlarını belirtmek için oluşturulmuştur.

XII. VOID GAMEOVER()

Bu fonksiyon, oyunun bitiş ekranını gösterir. İlk olarak ekranı temizler ve eğer "lightMode" özelliği etkinse arkaplanı beyaz yapar. Ardından, 7 segment göstergesini temizler ve skoru gösterir. Ekranın alt kısmında "OYUN BITTI" ve skor yazısı yazdırılır. Buzzer ile de birkaç nota çalınır ve 5 saniye beklenir. Son olarak, restart() fonksiyonu çağrılarak oyun yeniden başlatılır.

Fig. 2. OYUN SONU

XIII. VOID ADDAMMOHP()

Can ve mermi belli durumlarda artış gösterdiğini projede öğrenmiştik. Bu fonksiyonda can ve mermi artışı gerçekleştirilir.

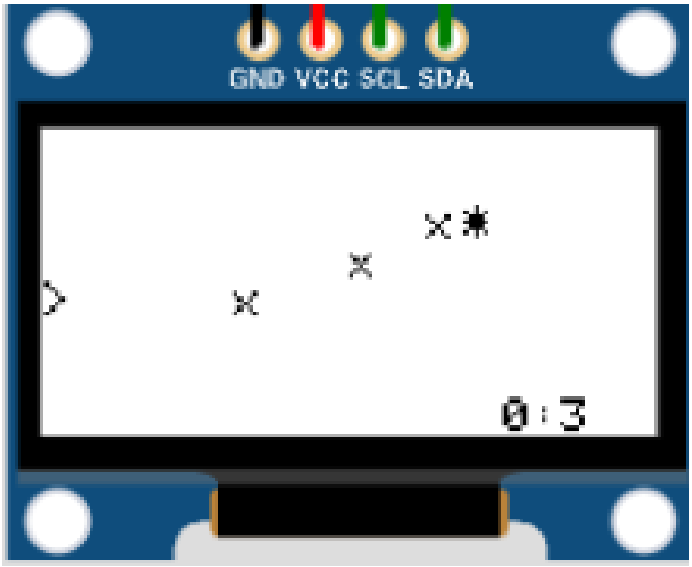

Fig. 3. OYUN EKRANI

XIV. VOID GAMELOOP()

Fonksiyonun işlevi temel olarak en basit anlamıyla her turda sırayla, geminin hareketini, meteorların doğmasını, tüm objelerin ekrana yazdırılmasını, LED'lerin durumlarının kontrolünü, oyuncunun skorunu 7 segment LED ekranda gösterilmesini ve oyunun sona erip etmeyip etmediğini kontrol etmeyi içermektedir. İlk olarak, ışık sensörünün okuma değeri kontrol edilir. Eğer okunan değer 700'den küçükse, yani ortam karanlıksa, lightMode değişkeni true yapılır ve beyaz bir arka plan rengi tercih edilir. Aksi takdirde, siyah bir arka plan tercih edilir ve lightMode değişkeni false yapılır. moveShip() fonksiyonu, kullanıcının gemisini hareket ettirmesini sağlar. spawnMeteor() fonksiyonu, bir meteorun ekrana rasgele yerleştirilmesi ve hareket etmesi için çağrılır. printAll() fonksiyonu, oyun alanındaki tüm nesnelerin yazdırılmasını sağlar. Bu nesneler arasında gemi, meteorlar, çöpler ve mermiler yer alır. Oyuncunun skoru, skor değişkeninden alınır ve 7 segment LED ekranında gösterilir. checkLEDs() fonksiyonu, oyuncunun mermi sayısı ve can puanlarına göre LED'lerin durumunu kontrol eder ve uygun şekilde yakar veya söndürür. life değişkeni 0'a eşit veya daha az olduğunda, gameOver() fonksiyonu çağrılır. Bu fonksiyon, oyunu durdurur, skoru gösterir, bir ses çalar ve 5 saniye bekler, ardından oyunu yeniden başlatır. Son olarak, delay() fonksiyonu, her turun süresini kontrol eder. Bu süre, tüm objelerin hareketi, ekrana yazdırılması ve LED'lerin durumunun kontrol edilmesi için yeterli olacak kadar kısa bir süre olmalıdır.

XV. VOID MAINMENU()

Bu fonksiyon, oyuna başlamadan önce kullanıcının ana menüde seçenekleri seçebileceği bir döngü içerir. Fonksiyon, kullanıcının oyunu başlatmasını veya oyunun zorluğunu değiştirmesini sağlar. Ana menü döngüsü, oyunun "gameState" değişkenine göre kontrol edilir. "gameState" değişkeni 0 ise, ana menü gösterilir ve kullanıcının seçim yapması beklenir.

”gameState” değişkeni 1 ise, oyun başlatılır ve ”gameloop” fonksiyonu çalıştırılır. Ana menü döngüsü, sonsuz bir döngüde çalışacak şekilde oluşturulmuştur, bu nedenle kullanıcı çıkış yapana kadar ana menüde kalır ve seçimlerini yapabilir.

Fig. 4. Menü Ekranı

XVI. VOID LOOP()

MainMenu() ve gameloop() fonnksiyonları bu fonksiyonda çalıştırılmıştır.

XVII. SONUÇLAR

- Arduino hakkında bilgi edinildi.
- Proteus hakkında bilgi edinildi.
- Gömülü sistemde kullanılan mikroişlemci kartlar hakkında bilgi edinildi.

XVIII. KAYNAKÇA

- https://www.youtube.com/watch?v=HVHVkKt-ldc
- https://www.youtube.com/watch?v=kBe75w0EUys
- https://www.youtube.com/watch?v=xAFnED4UZMM