

**Mühendislik ve Mimarlık Fakültesi**

**Bilgisayar Mühendisliği Bölümü**

**Gömülü Sistemler Dersi Final Projesi**

**Proje**

Kuluçka Makinesi

**Danışman**

DR.ÖĞR.ÜYESİ MOHAMMED HUSSEIN IBRAHIM

**Öğrenci**

Ender ÇELİK

19010011069

2022 – KONYA

**İçindekiler**

1. [Proje Bilgileri2](#ProjeBilgileri_1)
   1. [Proje Konusu2](#ProjeKonusu_1_1)
   2. [Proje Amacı ve Hedefi2](#ProjeAmaciVeHedefi_1_2)
   3. [Proje Özeti2](#ProjeOzeti_1_3)
2. [Projede Kullanılan Donanım ve Yazılımlar3](#ProjedeKulDonVeYaz_2)
   1. [Donanım Bilgileri3](#DonanimBilgileri_2_1)
   2. [Yazılım Bilgileri5](#YazilimBilgileri_2_2)
3. [Proje Çizimi5](#ProjeCizim_3)
4. [Proje Akış Diyagramı ve Algoritması7](#ProjeDiyagram_4)
5. [Proje Yazılım Kodları](#ProjeKodları_5)6
   1. [Arduino Proje Kodu](#ProjeArduinoKodu_5_1)6
6. [Kaynaklar](#ProjeKaynaklar_6)12
7. **Proje Bilgileri**
   1. **Proje Konusu**

Kuluçka sistemi için gerekli araçların kullanıcıya hitap etmeyen fiyatları ve özelleştirilebilir yanlarının çok az olması.

* 1. **Proje Amacı ve Hedefi**

Mevcut piyasadaki sistemlerden daha az maliyetli, direkt olarak amaca hizmet eden ve kullanıcı tarafından da rahatlıkla özelleştirilebilir bir sistem ortaya koymak.

* 1. **Proje Özeti**

Günümüzde direkt gelir amaçlı bu işe uygun ortamlarda veya eğitim amaçlı laboratuvarlarda çeşitli çevresel etmenler olması gereken doğal koşullara olabildiğince yaklaştırılarak yapay bir kuluçka ortamı ortaya çıkarılmakta ve sistemin elverdiği adet miktarınca embriyo sistemli bir şekilde yumurtasında büyütülmekte daha sonra bundan gelir elde edilmekte, eğitim amaçlı ise de gözlem yapabilme fırsatı elde edilmektedir. Yapay kuluçka ortamının oluşturulmasına yardımcı gereçler genellikle kullanıcıya hitap etmekten uzak, karmaşık ve bu karmaşıklığının bir sonucu olarak farklı parçaların kar marjları nedeniyle pahalı gereçlerdir. Bunun yanında kullanıcı tarafından bahsedilen kar marjı durumu nedeniyle görece gereksiz maliyetle edinilen bu ürünler özelleştirilebilmekten uzak yapıları sonucunda kullanıcının ufak sistemsel müdahaleler yapmasına elverişli değillerdir. Bu proje ile asıl amaç yani yumurtaları uygun sıcaklıklarda tutacak otomatik bir sistem geliştirilmesi amaçlanmış. Bu doğrultuda maliyetler oldukça azaltılarak bir adet arduino, bir adet LM35 sıcaklık sensörü, bir adet potansiyometre, farklı renklerde dip ledler, dirençler ve Jumper kablolar kullanılarak aynı işlevi daha az maliyet ve emsallerine oranla daha özelleştirmeye uygun yapıya sahip bir kuluçka makinesi ortaya konmuştur. Ortaya konan bu sistem ile hem amacın dışına çıkan gereksiz parçalar sistem dışına itilerek maliyet düşürülmüş hem de sade ve öz yapısıyla kullanıcının özelleştirmesine uygun bir yapı ortaya konmuştur.

1. **Projede Kullanılan Donanım ve Yazılımlar**
   1. **Donanım Bilgileri**

* **Arduino Leonardo**

Atmega32u4 temelli bir mikro denetleyici  kartıdır. Üzerinde 20 adet dijital giriş/çıkış pini (7 tanesi PWM çıkışı, 12 tanesi analog giriş olarak kullanılabilir), 16Mhz kristal, mikro usb soketi, güç soketi, ICSP konektörü ve reset tuşu bulundurmaktadır. Kart üzerinde mikro denetleyicinin çalışması için gerekli olan her şey bulunmaktadır. Kolayca usb kablosu üzerinden bilgisayara bağlanabilir, adaptör veya pil ile çalıştırılabilir.

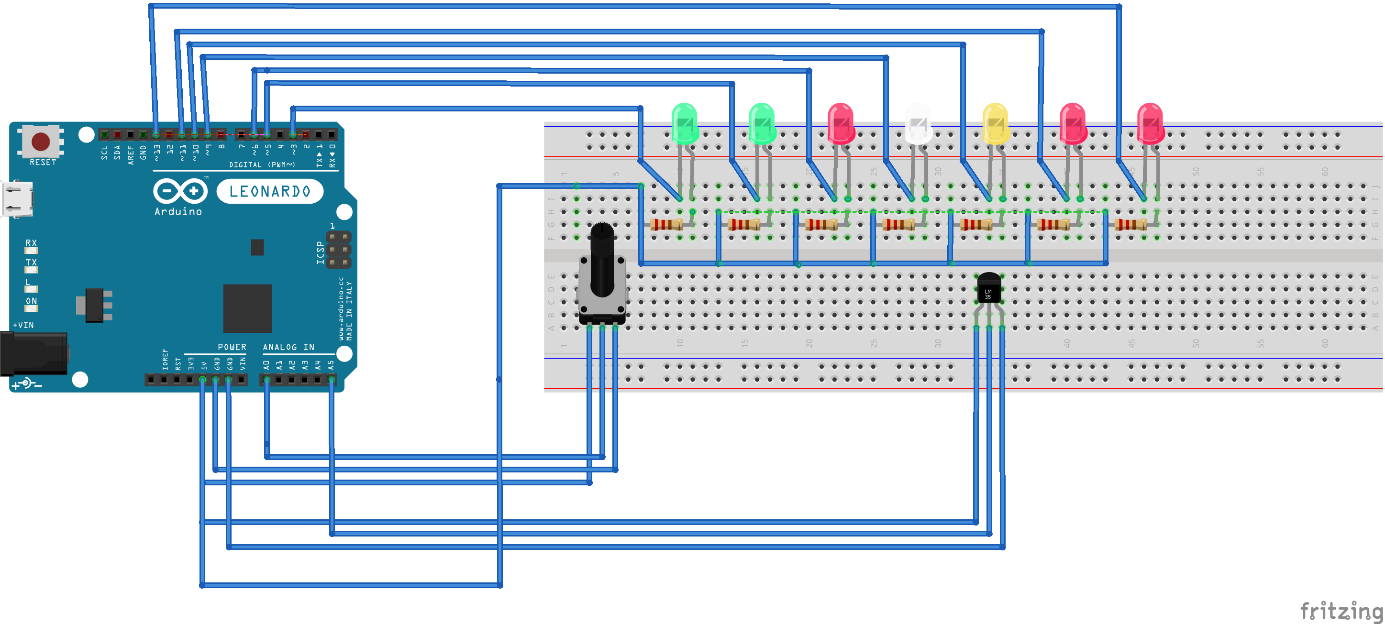
|  |  |
| --- | --- |
| Mikrodenetleyici | ATmega32u4 |
| Çalışma Voltajı | 5V |
| Giriş Voltajı (önerilen) | 7-12V |
| Giriş Voltajı (limit) | 6-20V |
| Dijital G/Ç Pinleri | 20 |
| PWM Kanalı | 7 |
| Analog Giriş Kanalı | 12 |
| Pin başına DC akım | 40 mA |
| 3.3V Pin için DC akım | 50 mA |
| [Flash](http://arduinoturkiye.com/etiket/flash/) Bellek | 32 KB (ATmega32u4) 4 KB bootloader tarafından kullanılıyor |
| SRAM | 2.5 KB (ATmega32u4) |
| EEPROM | 1 KB (ATmega32u4) |
| Clock Frekansı | 1. MHz |

* **LM35 Sıcaklık Sensörü**
  + Derece Santigrat olarak doğrudan kalibre edilir
  + Doğrusal + 10.0 mV / ˚C skala faktörü
  + 0.5˚C hassasiyet garantisi (+ 25˚C'de)
  + Tam −55˚ ila + 150˚C aralığında
  + 5V çalışır
  + 60 µA'dan az akım
* **Potansiyometre**
  + Kontrol edilebilir direnç olmasıdır. Elektroniğin temel elemanlarından biridir ve kontrol gerektiren devrelerin birçoğunda bulunmaktadır. Sembolü de normal bir direncin üzerine ok eklenmesiyle meydana gelir. Bunun sebebi de direnç değerinin anlık kontrol edilebildiğini göstermesidir.
* **Dip Led (Kırmızı, Beyaz, Yeşil, Sarı)**
  + LED, **L**ight **E**mitting **D**iode (Türkçesi ışık yayan diyot) sözcüklerinin baş harflerinden oluşan bir kısaltmadır. İsminden de anlaşılacağı üzere LED, bir diyottur. Bildiğimiz üzere diyot, akımın yalnızca bir yönden geçmesini sağlayan iki bacaklı yarı-iletken bir devre elemanıdır.
* **Direnç**
  + Elektrik devrelerinde direnç, bir iletken üzerinden geçen elektrik akımının karşılaştığı zorlanmadır. Mekanik sistemlerdeki sürtünmeye benzer özellikler gösterir. Direncin birimi **Ohm (Ω)**’dur. Denklemlerde **R** harfi ile gösterilir.
  + Dirençler, elektrikli devrelerde akımı sınırlayarak belli bir değerde tutmaya yararlar. Bunun haricinde hassas devre elemanlarının üzerlerinden yüksek akım geçmesini önlerler, besleme gerilimini ve akımı bölmek için de kullanılırlar. Ayrıca dirençlerin üzerlerine düşen akım değeri yükseldikçe ısınmalarından da faydalanılmaktadır.
* **Jumper Kablo**
  + Arduino, RaspberryPI, Breadboard gibi devre kartları üzerindeki elemanlar arası bağlantı sağlamak için kullanılan bağlantı kablosudur.
  1. **Yazılım Bilgileri**

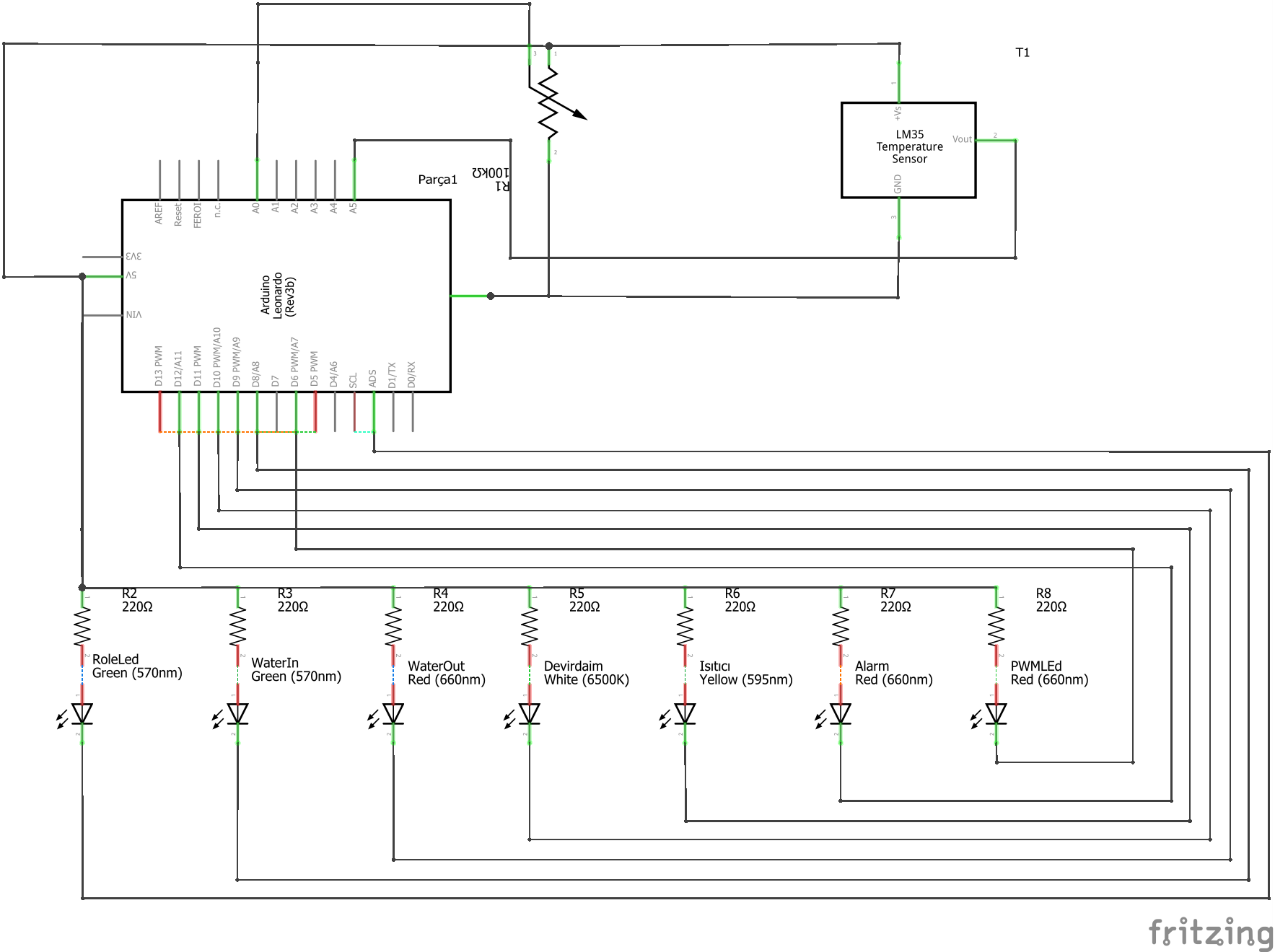
Kart üzerindeki yazılım işlemleri Arduino yazılım dili ve kendi IDE si üzerinden gerçekleştirilmiştir. Seriport, Dijital I/O ve Analog I/O kütüphanelerinden yararlanılmıştır. Proje içerisinde Seriport, ADC, PWM, Dijital I/O, Zamanlayıcı ve kütüphane özellikleri kullanılmıştır.

Bağlantı şeması çizimi için fritzing paket programından yararlanılmıştır. Algoritma şeması için ise Flow Chart programından faydalanılmıştır.

1. **Proje Çizim**

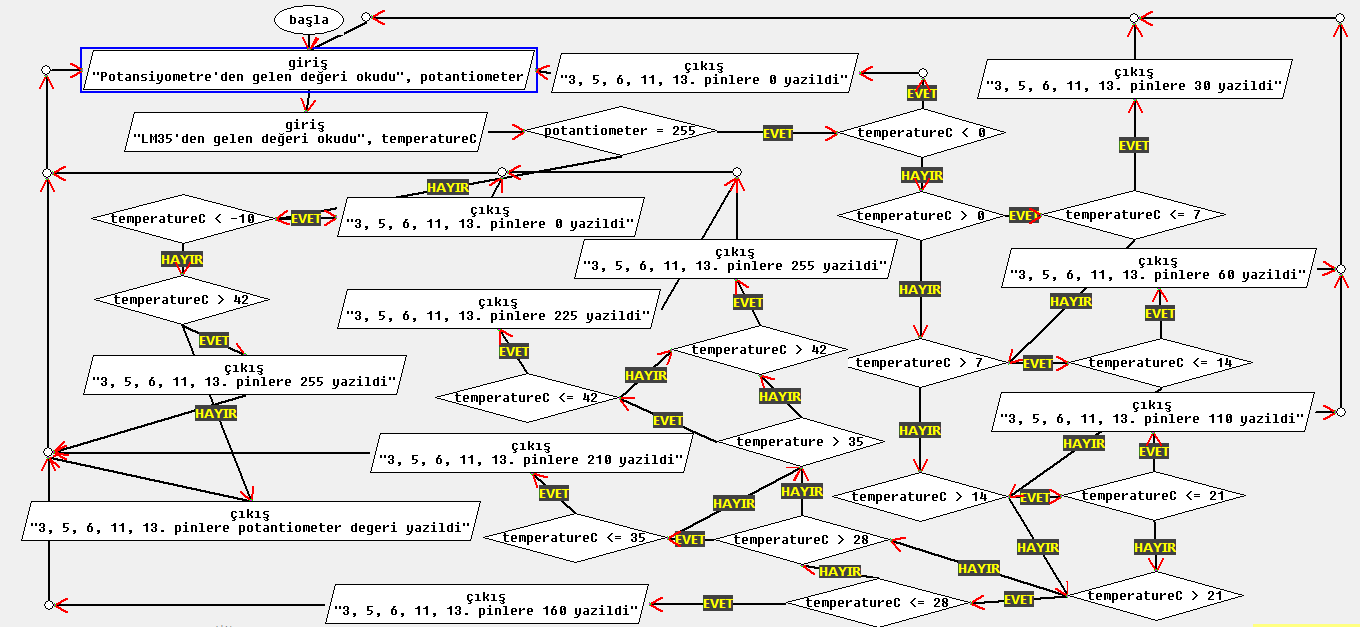


Şekil - Proje çizimi (Devre tasarım görünümü)



Şekil - Proje çizimi (Şematik Görünüm)

1. **Proje Akış Diyagramı ve Algoritması**



Şekil - Proje akış diyagramı

1. **Proje Yazılım Kodları**
   1. **Arduino Proje Kodu**

#include <EEPROM.h> //EEPROM.h kütüphanesini dahil ettik.

boolean toogle = LOW; //Sistemin çalıştığını yanıp sönerek belli edecek led için //kullanılacak boolean değişken.

int readValue;

int potentiometerValue;

float voltage;

float temperatureC;

#define analogPin0 A0 //Pinleri adlandırdık.

#define analogPin5 A5

#define ledGreen1 3

#define ledGreen2 5

#define ledRed1 6

#define ledWhite1 9

#define ledYellow1 10

#define ledRed2 11

#define ledRed3 13

void setup() {

Serial.begin(9600); //Haberleşmeyi başlattık.

EEPROM.write(0,0); //EEPROM'a daha sonra kullanacağım değerleri sırayla yazıyorum.

EEPROM.write(1,7);

EEPROM.write(2,14);

EEPROM.write(3,21);

EEPROM.write(4,28);

EEPROM.write(5,35);

EEPROM.write(6,42);

pinMode(analogPin0, INPUT); //Pinlerin kullanım durumları ayarlandı.

pinMode(analogPin5, INPUT);

pinMode(ledGreen1, OUTPUT);

pinMode(ledGreen2, OUTPUT);

pinMode(ledRed1, OUTPUT);

pinMode(ledWhite1, OUTPUT);

pinMode(ledYellow1, OUTPUT);

pinMode(ledRed2, OUTPUT);

cli(); //Interrupt'lar durduruldu.

TCCR1A = 0;

TCCR1B = 0;

TCNT1 = 0;

OCR1A = 15624; //1 saniyelik kesme ayarlandı. 3 saniyelik için: 46875

TCCR1B |=(1<<WGM12);

TCCR1B |=(1<<CS12)|(1<<CS10);

TIMSK1 |=(1<<OCIE1A);

sei();

}

ISR(TIMER1\_COMPA\_vect)

{

toogle = !toogle;

digitalWrite(ledWhite1, toogle); //Sistemin çalıştığını işaret eden ledin södürülüp //yakılması.

potentiometerValue = analogRead(analogPin0); //Potansiyometreden gelen değeri //okuduk.

potentiometerValue = map(potentiometerValue, 0, 1023, 0, 255); //Yukarıdaki değeri //kullanabileceğimiz aralığa oranladık.

//Serial.println(potentiometerValue); Potansiyometredeki değerin görülmek istenmesi //durumunda...

if(potentiometerValue == 255) //Potansiyometre 255 durumundayken işlenecek blok: //Kontrol Arduino'da.

{

analogWrite(ledYellow1, 255); //Kontrolün Potansiyometre de olduğunu belirten sarı //led'i söndürdük.

readValue = analogRead(analogPin5); //LM35 Sıcaklık sensöründen gelen değeri //okuduk.

voltage = readValue \* 4.68;

voltage /= 1024.0; //Santigrat'a dönüştürdük.

temperatureC = (voltage - 0.5) \* 100;

temperatureC = temperatureC + 6;

//Serial.println(temperatureC); //LM35'den gelen değerin görülmek istenmesi //durumunda.

if(temperatureC < EEPROM.read(0)) //EEPROM'a daha önce yazdığımız sıcaklık //değerlerine göre led'lerin parlaklığını ayarlayan if blokları:

{

Serial.println("ARALIK 1"); //Debug yapmak istenirse.

analogWrite(ledGreen1, 0);

analogWrite(ledGreen2, 0);

analogWrite(ledRed1, 0);

analogWrite(ledRed2, 0);

analogWrite(ledRed3, 0);

}

if(EEPROM.read(0)< temperatureC && temperatureC<= EEPROM.read(1)) // 0 ile 7

{

Serial.println("ARALIK 2"); //Debug yapmak istenirse.

analogWrite(ledGreen1, 30);

analogWrite(ledGreen2, 30);

analogWrite(ledRed1, 30);

analogWrite(ledRed2, 30);

analogWrite(ledRed3, 30);

}

if(EEPROM.read(1)< temperatureC && temperatureC<= EEPROM.read(2)) //7 ile 14

{

Serial.println("ARALIK 3"); //Debug yapmak istenirse.

analogWrite(ledGreen1, 60);

analogWrite(ledGreen2, 60);

analogWrite(ledRed1, 60);

analogWrite(ledRed2, 60);

analogWrite(ledRed3, 60);

}

if(EEPROM.read(2)< temperatureC && temperatureC<= EEPROM.read(3))

{

Serial.println("ARALIK 4"); //Debug yapmak istenirse.

analogWrite(ledGreen1, 110);

analogWrite(ledGreen2, 110);

analogWrite(ledRed1, 110);;

analogWrite(ledRed2, 110);

analogWrite(ledRed3, 110);

}

if(EEPROM.read(3)< temperatureC && temperatureC<= EEPROM.read(4))

{

Serial.println("ARALIK 5"); //Debug yapmak istenirse.

analogWrite(ledGreen1, 160);

analogWrite(ledGreen2, 160);

analogWrite(ledRed1, 160);

analogWrite(ledRed2, 160);

analogWrite(ledRed3, 160);

}

if(EEPROM.read(4)< temperatureC && temperatureC<= EEPROM.read(5))

{

Serial.println("ARALIK 6"); //Debug yapmak istenirse.

analogWrite(ledGreen1, 210);

analogWrite(ledGreen2, 210);

analogWrite(ledRed1, 210);

analogWrite(ledRed2, 210);

analogWrite(ledRed3, 210);

}

if(EEPROM.read(5)< temperatureC && temperatureC<= EEPROM.read(6))

{

Serial.println("ARALIK 7"); //Debug yapmak istenirse.

analogWrite(ledGreen1, 225);

analogWrite(ledGreen2, 225);

analogWrite(ledRed1, 225);

analogWrite(ledRed2, 225);

analogWrite(ledRed3, 225);

}

if(EEPROM.read(6)< temperatureC)

{

Serial.println("ARALIK 8"); //Debug yapmak istenirse.

analogWrite(ledGreen1, 255);

analogWrite(ledGreen2, 255);

analogWrite(ledRed1, 255);

analogWrite(ledRed2, 255);

analogWrite(ledRed3, 255);

}

}

if(potentiometerValue != 255) //Potansiyometre 255 [HIGH] durumunda değilken //işlenecek blok: Kontrol Potansiyometre'de yani kullanıcı da.

{

readValue = analogRead(analogPin5); //LM35 Sıcaklık sensöründen gelen değeri //okuduk.

voltage = readValue \* 4.68;

voltage /= 1024.0; //Santigrat'a dönüştürdük.

temperatureC = (voltage - 0.5) \* 100;

temperatureC = temperatureC + 6.0;

if(temperatureC < - 10)

{

Serial.println("Düşük sıcaklık kontrol"); //Debug yapmak istenirse.

analogWrite(ledYellow1, 255); //Kontrolün Potansiyometre de olduğunu belirten sarı //led'i söndürdük.

analogWrite(ledGreen1, 0);

analogWrite(ledGreen2, 0);

analogWrite(ledRed1, 0);

analogWrite(ledRed2, 0);

analogWrite(ledRed3, 0);

}

if(temperatureC >= 45)

{

Serial.println("Yüksek sıcaklık kontrol"); //Debug yapmak istenirse.

analogWrite(ledYellow1, 255); //Kontrolün Potansiyometre de olduğunu belirten sarı //led'i söndürdük.

analogWrite(ledGreen1, 255);

analogWrite(ledGreen2, 255);

analogWrite(ledRed1, 255);

analogWrite(ledRed2, 255);

analogWrite(ledRed3, 255);

}

if(temperatureC >= -10 && temperatureC <45)

{

Serial.println("Normal sıcaklık aralığında kullanıcı kontrolünde"); //Debug yapmak //istenirse.

analogWrite(ledYellow1, 0); //Kontrolün Potansiyometre de olduğunu belirten sarı //led'i yaktık.

analogWrite(ledGreen1, potentiometerValue);

analogWrite(ledGreen2, potentiometerValue);

analogWrite(ledRed1, potentiometerValue);

analogWrite(ledRed2, potentiometerValue);

analogWrite(ledRed3, potentiometerValue);

}

}

}

void loop() {}

1. **Kaynaklar**

<https://fritzing.org>

https://www.lucidchart.com