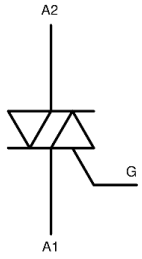
Proje, bilgisayar üzerinde çalışan sera simülasyonında seranın sıcaklık değerinin otomatik olarak kontrol edilmesini amaçlamaktadır.

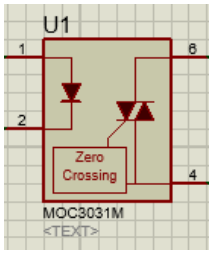
Model benzetimi için Proteus kullanılmıştır. Sıcaklık kontrolünü sağlamak için kullanılacak triyak devre elemanını çalıştırmak için ve Arduino mikrokontrolöründen tetikleme sinyali üretilmiştir. Üretilen bu tetikleme sinyali sıfır geçiş dedektörü yardımıyla ısıtıcının çalışması sağlanmıştır.

Simülasyon içerisinde dış ortamın sıcaklığı düşürücü etkisi ve ısıtma sisteminin ürettiği sıcaklık elektrik sinyalleri ile modellenerek bu sinyaller bir PID kontrol devresi ile kontrol edilmiş ve istenilen sıcaklıkta kalması sağlanmıştır.

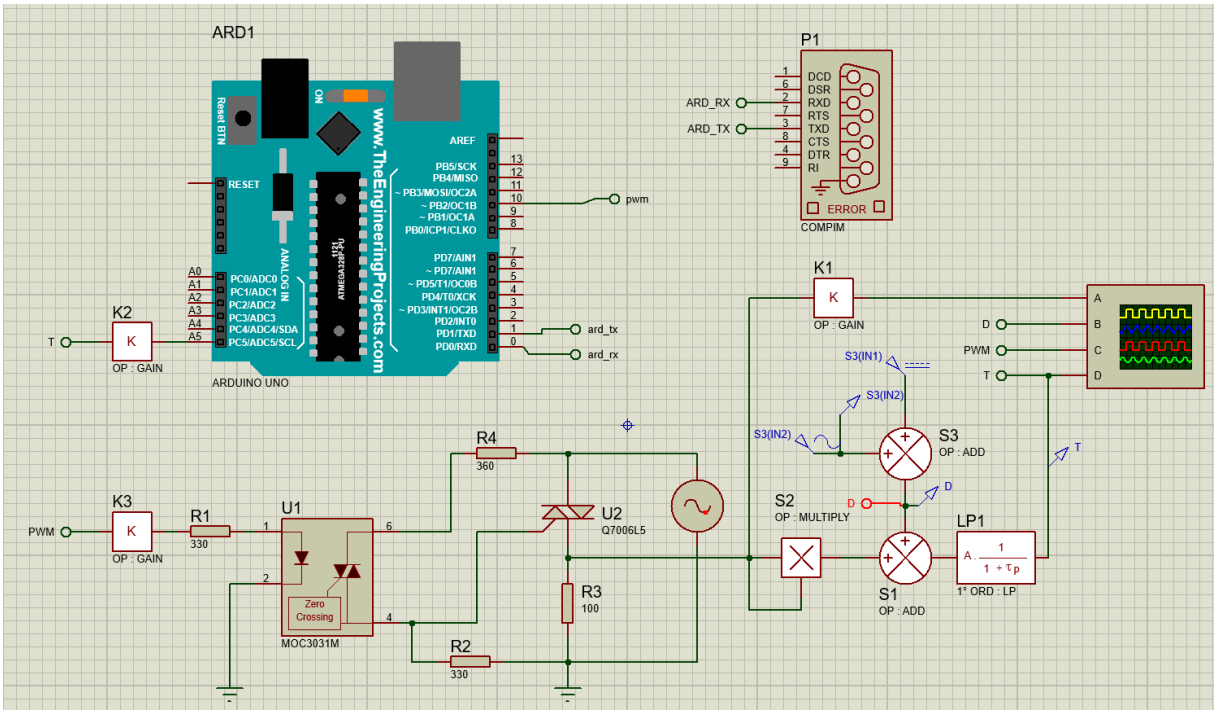
Seranın modellenmesi için gerekli kriterler belirlenmiş ve ardından Proteus programının uygun olduğu karar verilmiştir. Kullanılacak elemanlar belirlendikten sonra seranın benzetimi elektriksel bir devre olarak kurulmuştur. Isıtıcının çalışmasını kontrol etmek için triyak devreye eklenmiştir. Daha verimli bir sistem oluşturmak adına sıfır geçiş dedektörü kullanılmıştır. Bu devre elemanı yardımıyla pozitiften negatife veya negatiften sinüzoidal bir dalga formunun pozitif seviyesine değişimi sırasında triyak tetiklemesi gerçekleşmektedir. Sisteme bozucu etki iki sinyalin birleşimi şeklinde verilmiştir. +10 DC volt, büyüklüğü 10 ve frekansı 17mHz olan bir sinüs dalgası kullanılmıştır. Bu yapının amacı günün 0-20 derece aralığında değişen sıcaklığını dış etken olarak devreye vermektir. Seri haberleşmeyi sağlamak adına COMPIM elemanı kullanılmıştır.



Şekil 1: Triyak



Şekil 2: MOC3031M



Şekil 3: Sera Modeli

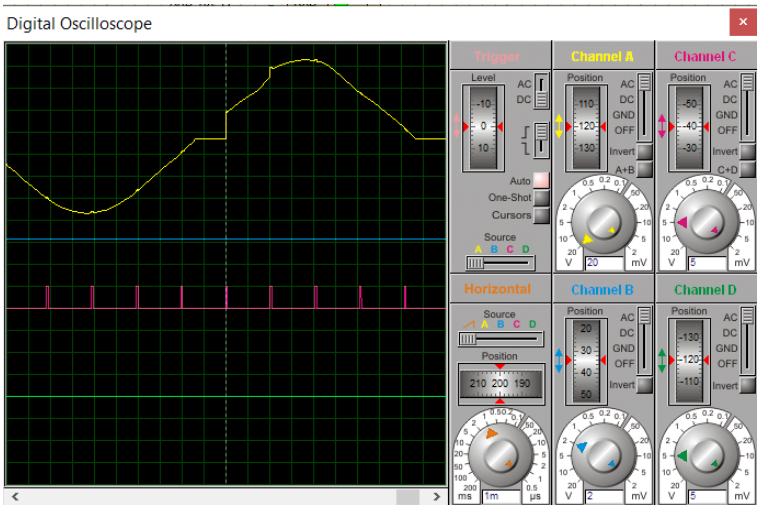
Isıtıcının kontrol sinyalinin üretilmesinde Arduino mikrokontrolörünün kullanımına karar verilmiştir. Önce devredeki sıcaklık değeri okunmuş, sonrasında PID kontrol yöntemiyle tetikleme sinyali oluşturulmuştur.

Tetikleme sinyali Arduino'nun analog pinlerinden biri olan A5 pininden sıcaklık değeri okunmuş, yine analog pinlerinden biri olan 10. pininden tetikleme sinyali sisteme verilmiştir.

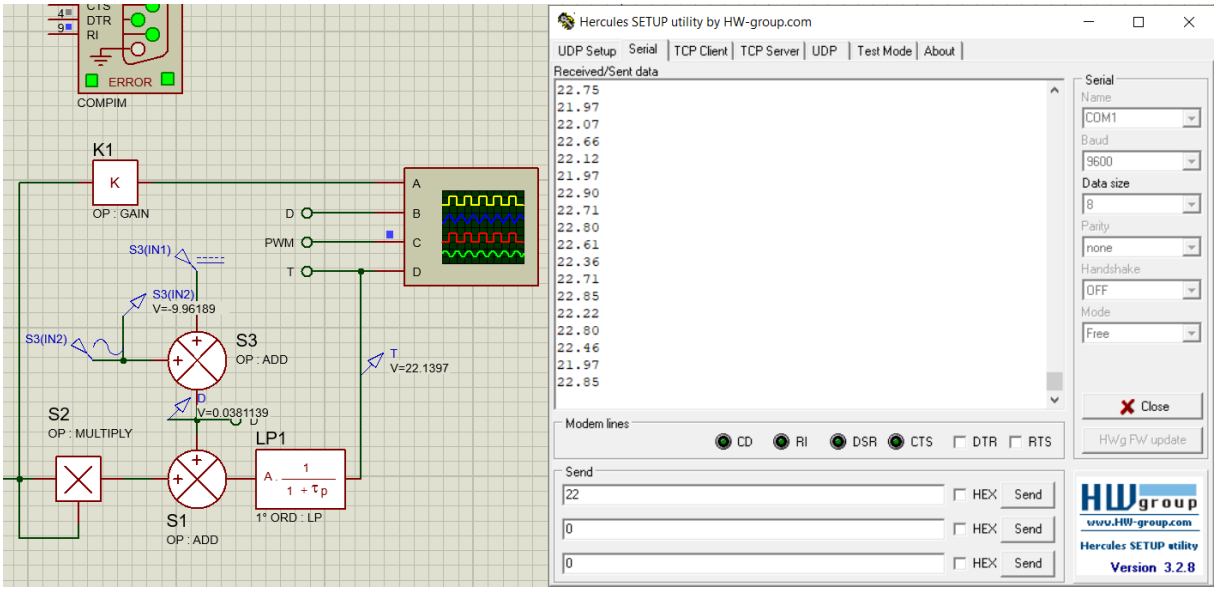
```
#define sicaklikOkumaPin A5
#define pwmPin 10

float okunanVoltaj;
float sicaklikDeger, sicaklikGerilim;
float kp = 10,  kd = 1.5, ki=3, zamanDegisimi = 0.01;
int ref=25, pwm, pidValue;
double sampleTime, sample;
void setup() {
  Serial.begin(9600);
  pinMode(10, OUTPUT);
  pinMode(A5, INPUT);
}

void loop() {
  if(Serial.available()){
    ref=Serial.parseInt();
    Serial.println(ref);
  }
  okunanVoltaj = analogRead(sicaklikOkumaPin);
  sicaklikDeger = okunanVoltaj / 20.48;
  analogWrite(pwmPin, pidValue);
  sampleTime = millis() - sample;
  if (sampleTime > 250) {
    sample = millis();
    sampleTime = 0;
    //Serial.print("\r Sicaklik: ");
    Serial.println(sicaklikDeger);
  }
  pidValue = pid(ref, sicaklikDeger);
}
int pid(int ref, int out) {
  int hata;
  static int sonHata;
  static int toplamHata;
  hata = ref - out;
  if (hata > 0) {
    pwm = (hata) * kp + ((hata - sonHata) / zamanDegisimi) * kd + toplamHata*zamanDegisimi*ki;
    sonHata = hata;
    return pwm;
    if (pwm > 255) {
      pwm = 255;
      return pwm;
    }
  }
  if (hata <= 0) {
    pwm = 0;
    return pwm;
  }
}
```

Şekil 4: Devre Kontrolünü Sağlayan Arduino Kodu

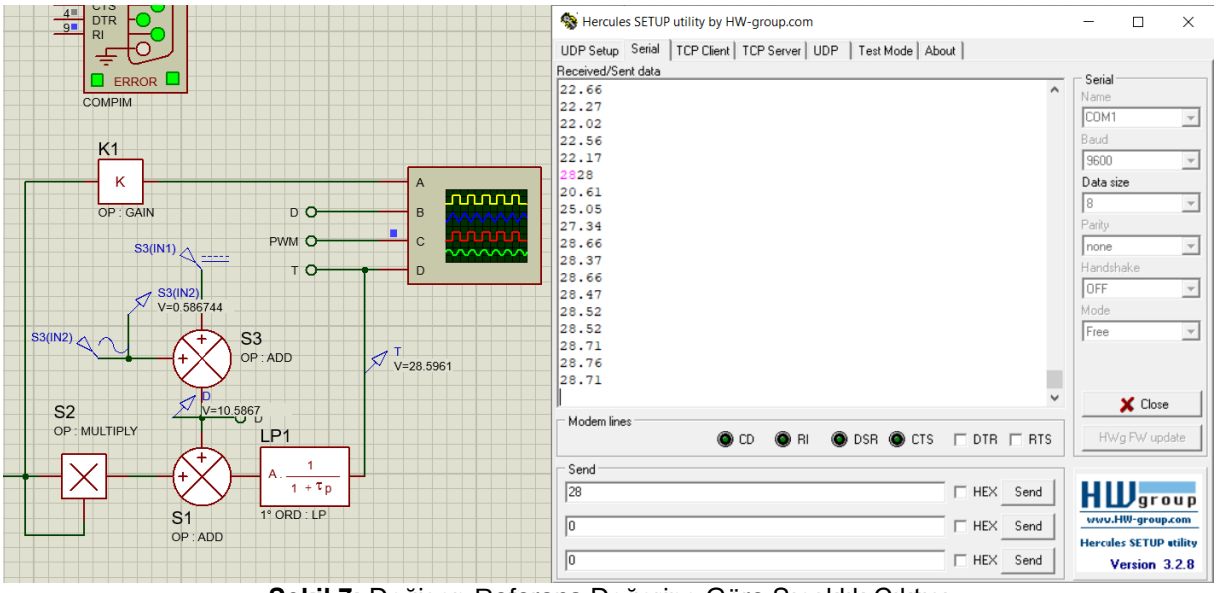
Seri haberleşme kullanarak referans değerini elde edilmesi ve bunun sistemde uygulanması amacıyla Hercules Serial Port Terminal kullanılmıştır.



Şekil 5: Elde Edilen Osiloskop Çıktısı



Şekil 6: Sabitlenen Sıcaklık Değeri



Şekil 7: Değişen Referans Değerine Göre Sıcaklık Çıktısı