

**T.C.**

**MİLLİ EĞİTİM BAKANLIĞI**

**SİVAS MERKEZ**

**Sivas Fen Lisesi**

**Bilgisayar Bilimi Dersi**

**Rapor**

|  |  |
| --- | --- |
| **Rapor No** | Proje-2 |
| **Rapor Tarih** | 30.05.2018 |
| **Proje Adı** | SAAT YAZAN ROBOT |

Bilgisayar Bilimi Öğretmeni

Ersin TÜTÜNCÜ

2017-2018



**T.C.**

**MİLLİ EĞİTİM BAKANLIĞI**

**SİVAS MERKEZ**

**Sivas Fen Lisesi**

**Proje Grup**

|  |  |  |
| --- | --- | --- |
| **Proje Görev** | **Numara** | **Ad-Soyad** |
| Proje Yönetimi | 318 | SUDE ÖRNEK |
| Doküman Yönetimi | 165 | GÜLCİHAN TUFAN |
| Lojistik Yönetim | 191 | MELİK SENCER |
| Yazılım Geliştirme | 3 | BEYZA TOKER |
| Web ve GitHub Yönetimi | 471 | HASAN HÜSEYİN ŞİRANLI |
| Sunum Yönetimi | 450 | ŞEVVAL YILDIRIM |

İÇİNDEKİLER

[ÖZET 3](#_Toc477023339)

[Anahtar Kelimeler 3](#_Toc477023340)

[ABSTRACT 3](#_Toc477023341)

[Key Words 3](#_Toc477023342)

[Proje Görev Dağılımı Listesi ve Görev Dağılımı Açıklaması 4](#_Toc477023343)

[ Görev Dağılımı ve Sorumlusu 4](#_Toc477023344)

[ Görev süresince sürdürülen eylemler 5](#_Toc477023345)

[ Görevlerin iş yükü şeması 6](file:///C:\Users\HİLAL\Desktop\düzenlendi.docx#_Toc477023346)

[ Yoklama Çizelgeleri 7](#_Toc477023347)

[ Haftalık İş Katkı Cetvelleri 8](#_Toc477023348)

[GİRİŞ 8](#_Toc477023349)

[1.Projenin Açıklaması 8](#_Toc477023350)

[2.UML Diyagramlar 8](#_Toc477023351)

[3.Donanım Yapısı: 9](#_Toc477023352)

[a.Gömülü Sistemler Mimarisi ve Devre Tasarımı 9](#_Toc477023353)

[b.Mekanik Sistem Mimarisinin Tanıtılması: 12](#_Toc477023354)

[PID 17](#_Toc477023355)

[4.Yazılım Yapısı 18](#_Toc477023356)

[SONUÇ 19](#_Toc477023357)

[1.Bilgi Düzeyine Katkıları: 19](#_Toc477023358)

[2.Teknolojik Katkıları: 19](#_Toc477023359)

[3.Ekip Çalışması Katkıları 19](#_Toc477023360)

[4.Aksayan Yönler: 20](#_Toc477023361)

[5.Görüş ve Öneriler: 20](#_Toc477023362)

## ÖZET

2.Proje kapsamında Plotclock yani saat yazan robot yapılacaktır.Proje tamamlandığı takdirde robot yapısındaki kod ve arduino uno sayesinde saati algılayıp yazabilecektir.Proje gerçekleştirilirken devre tasarımı, devre elemenlarının lehimlenmesi,test ve deneme aşamaları,kodlama,sunum gibi aşamalar gerçekleştirilecektir.

## Anahtar Kelimeler

PLOTCLOCK ARDUİNO UNO

## ABSTRACT

The robot that writes time in the scope of the 2nd project will be done. if the project is completed, the robot code and arduino uno and code will be able to detect and write the clock

. During the project, stages such as circuit design, soldering of circuit elements, testing and testing stages, coding, presentation will be realized.

## Key Words

## SERVO MOTOR AND ARDUİNO UNO

## Proje Görev Dağılımı Listesi ve Görev Dağılımı Açıklaması

## Görev Dağılımı ve Sorumlusu

|  |  |  |
| --- | --- | --- |
| Proje Yönetimi | Görev dağılımı ve takibinden sorumlu kişi, aynı zamanda proje grubunun çalışma takvimini ve düzenini ayarlamaktadır.Grupta bulunan kişilerle iletişim halinde olup projenin yönetimini sağlar. | SUDE ÖRNEK |
| Döküman Yönetimi | Projenin tüm tasarım ve çizimlerinden,proje raporunun sunulmasından,dökümanların uygun forma getirilmesinden kodlamaya ait diagram ve modellerin hazırlanması ve web sitesi tasarımından sorumlu olan kişidir. | GÜLCİHAN TUFAN |
| Lojistik Yönetimi | Projede kullanılacak tüm elemanların, malzemelerin belirlenmesi ve temin edilmesi,en uygun tasarımın yapılması için geliştirmelerin yapılmasıyla ve projenin donanımsal kısmının tanıtılması ile ilgilenen kişidir. | MELİK SENCER |
| Yazılım Geliştirme Yönetimi | Yazılım için araştırmaların yapılması, yazılım aşamalarının proje grubuna dağıtılması,Yazılım ile ilgili raporların hazırlanarak ilgili bölüme(döküman yönetimine) aktarılması yazılım ve süreç testlerinin gerçekleştirilmesi ile ilgilenen kişidir. | BEYZA TOKER |
| WEB ve GitHub Yönetimi | Proje tanıtımı için WEB sayfasının hazırlanması, projenin GitHub yönetiminin yapılması,döküman yöneticisinden almış olduğu raporlar ile WEB sitesine ve GitHub'a işlemekle sorumlu olan kişidir. | HASAN HÜSEYİN ŞİRANLI |
| Sunum Yönetimi | Proje teslim zamanında sunumun, yapılan tüm işlemlerin uygun bir biçimde anlatılmasından,rapor ve evrakların eksiksiz bir şekilde sunulmasından ve önerilere,  sorulara uygun çözümler üretmekten sorumludur. | ŞEVVAL YILDIRIM |

## Görev süresince sürdürülen eylemler

|  |  |
| --- | --- |
| 1.Hafta | Proje Hakkında Bilgi Edinme |
| 2.Hafta | Malzeme Seçimi |
| 3.Hafta | Mekanik ve Elektronik Tasarım |
| 4.Hafta | Yazılım |
| 5.Hafta | Grup elemanlarına ait iş yükünün tamamlanması (rapor,web,github) |

## Görevlerin iş yükü şeması

SUNUM

## Yoklama Çizelgeleri

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Grup**  **Tarihler** | **02.05.2018** | **09.05.2018** | **16.05.2018** | **23.05.2018** | **30.05.2018.** |
| **MELİK SENCER MATOĞLU** | **+** | **+** | **+** | **+** | **+** |
| **SUDE ÖRNEK** | **+** | **+** | **+** | **+** | **+** |
| **BEYZA TOKER** | **+** | **+** | **+** | **+** | **+** |
| **ŞEVVAL YILDIRIM** | **+** | **+** | **+** | **+** | **+** |
| **GÜLCİHAN TUFAN** | **+** | **+** | **+** | **+** | **+** |
| **HASAN HÜSEYİN ŞİRANLI** | **+** | **+** | **+** | **+** | **+** |

## Haftalık İş Katkı Cetvelleri

|  |  |
| --- | --- |
|  | **Yapılan iş** |
| **02.05.2018** | Proje araştırması |
| **09.05.2018** | Malzeme Teminatı |
| **16.05.2018** | Elektronik ve Mekanik Tasarım |
| **23.05.2018** | Yazılım |
| **30.05.2018** | Deneme ve Test Aşamaları |
|  |  |
|  |  |

# 

# GİRİŞ

## 2.Projenin Açıklaması

Yapılan 2. Projenin amacı plotcloğun bildirilen saati yapısındaki arduino uno sayesinde algılayıp sayesinde hareketlerini kolaylaştırarak bildirilen saati altındaki mekanik parçanın üstündeki düz zemine yazmaktır.aynı zamanda servo motorlar sayesinde hareketlerini kolaylaştırarak bildirilen saati altındaki mekanik parçanın üstündeki düz zemine yazmaktır.

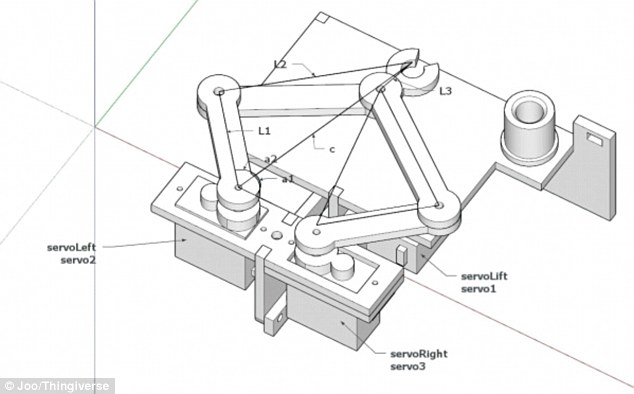
## 3.Donanım Yapısı:

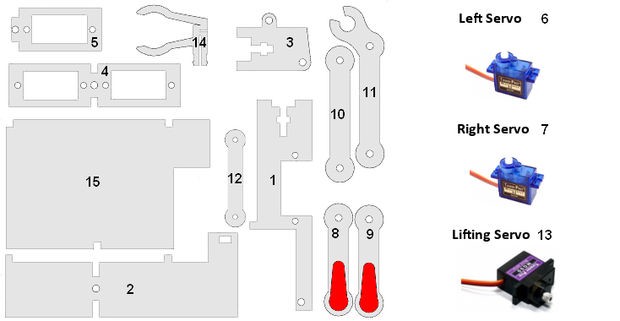
### a.Gömülü Sistemler Mimarisi ve Devre Tasarımı

|  |  |
| --- | --- |
| **C:\Users\HP\Downloads\arduino-uno-r3-klon-usb-kablo-hediyeli-usb-chip-ch340-16006-27-B.jpg** | **ARDUİNO UNO**  14 tane dijital giriş / çıkış pini vardır. Bunlardan 6 tanesi PWM çıkışı olarak kullanılabilir. Ayrıca 6 adet analog girişi, bir adet 16 MHz kristal osilatörü, USB bağlantısı, power jakı (2.1mm), ICSP başlığı ve reset butonu bulunmaktadır. |
| **C:\Users\hp\Downloads\ser1.jpg** | SERVO MOTORLAR  Radyo teknolojilerinde ve robotlarda en çok kullanılan motor çeşididir.İvme ve hız kontrolünü ayarlar yani hareketin düzenli ve eksiksiz yapılabilmesini sağlar.Amaç ise komut gelmediği müddetçe servo motorların istenilen pozisyonda kalabilmesidir. |
| **C:\Users\hp\Downloads\plotclock-4.jpg** | **Mekanik Parçalar**  **Düz bir zemin oluşturup saatin yazılabilmesini sağlar .** |
| **C:\Users\hp\Downloads\indir (2).jpg** | VİDALAR  Mekanik parçaların birleşmesini sağlayan parçalardır.Yaklaşık 0.7 mmdir |
|  | **Jumper Kablo (E-E/D-E)**  Devre elemanlarının bağlantılarını gerçekleşirmek için kullanabiliriz. |
|  |  |
| **C:\Users\hp\Downloads\snowmantahtakalemi13126670834e3db5cb2eb75.jpg** | **KALEM**  **Saatin yazılmasını sağlayan parçadır.** |
|  |  |

#### Devre Tasarımı: C:\Users\hp\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\IMG-20180526-WA0002.jpg

**PLOTCLOCK TASARIM**

****

**DEVRE ELEMANLARI**

**1.SERVO MOTOR**

****

**Servo motorlar**   Endüstriyel kontrol alanındaki teknolojik gelişmeler birçok [özel motorun](http://www.inverter-plc.net/motor/motor.html)   
ortadan kalkmasına uygulamaların çoğunun nispeten az sayıda motor   
tipiyle gerçekleşmesini dikkatin motordan, kaynak ve kontrol   
düzenlemelerine kaymasına neden olmuştur. Böylece üstün bir performans  
ve esneklik sağlanmıştır.   
    Ancak sabit hızlı bir motordan daha fazlası gerekli ise örneğin,   
[pozisyonlama](http://www.inverter-plc.net/servo_sistem/pozisyon_kontrol.html), yüksek kararlılık, periyodik çalışma, dinamik yük ve hız   
değişikliği isteniyorsa kullanıcının bir motor ve [sürücü](http://www.inverter-plc.net/servo_sistem/servo_motor_s%C3%BCr%C3%BCc%C3%BC_sistemleri.html) devresini satın   
alması gerekir.   
Burada servo motor ve sürücüleri hakkında bilgi sahibi olmanızı  
ve bir sistem için en iyi servo motor ve sürücüsünü seçmenize,   
sistemin gerektirdiği parametre değişikliğini yapmanıza, kontrol   
programlama yazılımını hatasız olarak yapmanıza yardımcıdır.

**2.ARDUİNO UNO**

##### C:\Users\HP\Downloads\arduino-uno-r3-klon-usb-kablo-hediyeli-usb-chip-ch340-16006-27-B.jpg

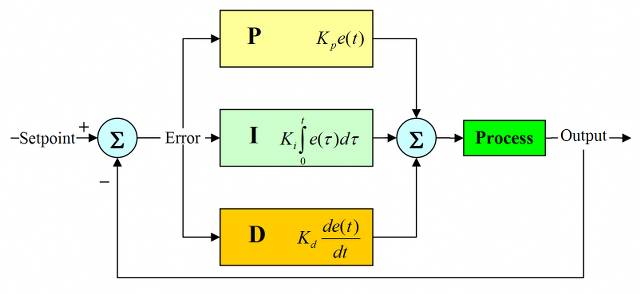
**Özellikler**

|  |  |
| --- | --- |
| Mikrodenetleyici | ATMega 328 |
| Çalışma Gerilimi | 5V |
| Giriş Gerilimi(önerilen) | 7-12V |
| Dijital I/O Pinleri | * 14 (6 tanesi PWM çıkışı) |
| Analog Giriş Pinleri | 6 |
| Her I/O için Akım | 40mA |
| * 3.3V Çıkış için Akım | * 50 mA |
| EEPROM | * 1 KB |
| SRAM | * 2 KB |
| Flash Hafıza | * 32 KB (Atmega2560) 0.5 KB kadarı bootloader tarafından kullanılmaktadır |
| Uzunluk | * 68.6 mm |
| Genişlik | 53.4mm |
| Ağırlık | * 25gr |

Arduino Uno bir USB kablosu ile bilgisayar bağlanarak çalıştırılabilir ya da harici bir güç kaynağından beslenebilir. Harici güç kaynağı bir [AC-DC adaptör](http://www.robotiksistem.com/pil_aku_batarya.html) ya da bir pil / batarya olabilir. Adaptörün 2.1 mm jaklı ucunun merkezi pozitif olmalıdır ve Arduino Uno 'nun power girişine takılmalıdır. Pil veya bataryanın uçları ise power konnektörünün GND ve Vin pinlerine bağlanmalıdır.

Güç pinleri:

* **VIN:**Harici güç kaynağı kullanırken 7-12V arası gerilim giriş pini.
* **5V:**Regülatörden çıkan 5V çıkışı verir. Eğer kart sadece USB (5V) üzerinden çalışıyor ise USB üzerinden gelen 5V doğrudan bu pin üzerinden çıkış olarak verilir. Eğer karta güç Vin (7-12V) veya güç soketi (7-12V) üzerinden veriliyorsa regülatörden çıkan 5V doğrudan bu pin üzerinden çıkış olarak verilir.
* **GND:**Toprak pinleridir.
* **3.3V :** Arduino kart üzerindeki regülatörden sağlanan 3,3V çıkışıdır. Maksimum 50 mA dir.
* **IOREF :** Arduino kartlar üzerindeki bu pin, mikrodenetleyicinin çalıştığı voltaj referansını sağlar. Uygun yapılandırılmış bir shield IOREF pin voltajını okuyabilir ve uygun güç kaynaklarını seçebilir ya da 3.3 V ve 5 V ile çalışmak için çıkışlarında gerilim dönüştürücülerini etkinleştirebilir.

****

**Şekil 9: PID kontrol Diyagramı**

PID,Oransal İntegral Türev için kullanılan bir kısaltmadır.En genel tanımıyla bir kontrol geri bildirim mekanizmasıdır.PID yönteminin en temel amacı hatayı minimize etmek,en aza indirmektir.PID kontrolünü uygulamak ve kavramak oldukça zordur.PID kontrolünde öncelikle hata tanımlaması yapılmalıdır.Hata ise referans değere olan uzaklık olarak tanımlanabilir.Açıklamak gerekirse;

Referans=İstenilen değer

Gelen=Şuan ki Konum

HATA=Referans-Gelen

**Oransal Terim(P):**

Oransal terim, sistemden gelen hatayı bir katsayı ile çarparak hatayı küçültmeyi hedefler.Bozucu etkileri de mevcuttur.Projemizin yazılım kısmında oldukça ağırlık verdiğimiz PID kontrolünde deneme ve test aşamasında bu katsayıya büyük bir değer vermemiz gerektiğini anladık.

P=Kp\*HATA

**Integral Terimi(I):**

Integral hatanın alanını bulmak anlamına gelir. Integralin çok yükselmesini önlemek için sınırlandırmak gereklidir. Sürekli toplandığı için integral çok artarsa tekrar azalmasını beklemek zaman alır. Bu yüzden integrali sınırlamak sistemin çabuk toparlamasını sağlayacaktır.

I = I + (Ki \* HATA \* dt)

dt:PID fonksiyonuna her girdiğinde geçen zaman.

**Türev Terimi(D):**

Türev sistemdeki iki örnek arasındaki zamanı hesaplar.Eğer hatada bir değişim olmadıysa türev sıfır olur.

EHata=Bir önceki hatanın değeri

HD=HATA-EHata

D=(Kd\*HD)/dt

**PID Algoritması**

Kp , Ki ve Kd katsayılarından oluşur.Bu katsayılar deneme yanılma yöntemiyle bulunur.Yapacağınız sistemde optimum katsayıları bulmak için değerde değişiliklik yapıp sistemi gözlemlemeniz gerekmektedir.

HATA = Referans - Gelen

HD = HATA - EHata

P = Kp \* Hata

I = I + (Ki \* HATA\* dt)

D = (Kd \* HD)/dt

PID = P + I + D

EHata = Hata

### 

### 4.Yazılım Yapısı

//#define CALIBRATION   
#define SERVOFAKTORLEFT 500  
#define SERVOFAKTORRIGHT 400  
   
#define SERVOLEFTNULL 2400  
#define SERVORIGHTNULL 400  
   
#define SERVOPINLIFT 2  
#define SERVOPINLEFT 3  
#define SERVOPINRIGHT 4  
   
   
#define LIFT0 1080   
#define LIFT1 925   
#define LIFT2 725   
   
#define LIFTSPEED 1500  
   
#define L1 35  
#define L2 55.1  
#define L3 13.2  
   
#define O1X 22  
#define O1Y -25  
#define O2X 47  
#define O2Y -25  
   
#include <TimeLib.h> // Time kütüphanesini çağırdık  
#include <Servo.h> // Servo motor için gerekli olan kütüphaneyi çağırdık  
   
#ifdef REALTIMECLOCK  
   
 #include <Wire.h>  
 #include <DS1307RTC.h>   
#endif  
   
int servoLift = 1500;  
   
Servo servo1; //   
Servo servo2; //   
Servo servo3; //   
   
volatile double lastX = 75;  
volatile double lastY = 47.5;  
   
int last\_min = 0;

void setup()   
{   
#ifdef REALTIMECLOCK  
 Serial.begin(9600);  
   
tmElements\_t tm;

if (RTC.read(tm))   
 {  
 setTime(tm.Hour,tm.Minute,tm.Second,tm.Day,tm.Month,tm.Year);  
 Serial.println(“DS1307 time is set OK.”);  
 }   
 else  
 {  
 if (RTC.chipPresent())  
 {  
 Serial.println(“DS1307 is stopped. Please run the SetTime example to initialize the time and begin running.”);  
 }   
 else  
 {  
 Serial.println(“DS1307 read error! Please check the circuitry.”);  
 }   
 // Set current time only the first to values, hh,mm are needed  
 setTime(16,05,0,0,0,0);  
 }  
#else   
 // Set current time only the first to values, hh,mm are needed  
 setTime(16,05,0,0,0,0);  
#endif  
   
 drawTo(75.2, 47);  
 lift(0);  
 servo1.attach(SERVOPINLIFT); // kaldırma servosu  
 servo2.attach(SERVOPINLEFT); // sol servo  
 servo3.attach(SERVOPINRIGHT); // sağ servo  
 delay(1000);  
   
}   
   
void loop()   
{   
   
#ifdef CALIBRATION  
   
 // Servohorns will have 90° between movements, parallel to x and y axis  
 drawTo(-3, 29.2);  
 delay(500);  
 drawTo(74.1, 28);  
 delay(500);  
   
#else   
   
   
 int i = 0;  
 if (last\_min != minute()) {  
   
 if (!servo1.attached()) servo1.attach(SERVOPINLIFT);  
 if (!servo2.attached()) servo2.attach(SERVOPINLEFT);  
 if (!servo3.attached()) servo3.attach(SERVOPINRIGHT);  
   
 lift(0);  
   
 hour();  
 while ((i+1)\*10 <= hour())  
 {  
 i++;  
 }  
   
 number(3, 3, 111, 1);  
 number(5, 25, i, 0.9);  
 number(19, 25, (hour()-i\*10), 0.9);  
 number(28, 25, 11, 0.9);  
   
 i=0;  
 while ((i+1)\*10 <= minute())  
 {  
 i++;  
 }  
 number(34, 25, i, 0.9);  
 number(48, 25, (minute()-i\*10), 0.9);  
 lift(2);  
 drawTo(74.2, 47.5);  
 lift(1);  
 last\_min = minute();  
   
 servo1.detach();  
 servo2.detach();  
 servo3.detach();  
 }  
   
#endif  
   
}   
   
// Writing numeral with bx by being the bottom left originpoint. Scale 1 equals a 20 mm high font.  
// The structure follows this principle: move to first startpoint of the numeral, lift down, draw numeral, lift up  
void number(float bx, float by, int num, float scale) {  
   
 switch (num) {  
   
 case 0:  
 drawTo(bx + 12 \* scale, by + 6 \* scale);  
 lift(0);  
 bogenGZS(bx + 7 \* scale, by + 10 \* scale, 10 \* scale, -0.8, 6.7, 0.5);  
 lift(1);  
 break;  
 case 1:  
   
 drawTo(bx + 3 \* scale, by + 15 \* scale);  
 lift(0);  
 drawTo(bx + 10 \* scale, by + 20 \* scale);  
 drawTo(bx + 10 \* scale, by + 0 \* scale);  
 lift(1);  
 break;  
 case 2:  
 drawTo(bx + 2 \* scale, by + 12 \* scale);  
 lift(0);  
 bogenUZS(bx + 8 \* scale, by + 14 \* scale, 6 \* scale, 3, -0.8, 1);  
 drawTo(bx + 1 \* scale, by + 0 \* scale);  
 drawTo(bx + 12 \* scale, by + 0 \* scale);  
 lift(1);  
 break;  
 case 3:  
 drawTo(bx + 2 \* scale, by + 17 \* scale);  
 lift(0);  
 bogenUZS(bx + 5 \* scale, by + 15 \* scale, 5 \* scale, 3, -2, 1);  
 bogenUZS(bx + 5 \* scale, by + 5 \* scale, 5 \* scale, 1.57, -3, 1);  
 lift(1);  
 break;  
 case 4:  
 drawTo(bx + 10 \* scale, by + 0 \* scale);  
 lift(0);  
 drawTo(bx + 10 \* scale, by + 20 \* scale);  
 drawTo(bx + 2 \* scale, by + 6 \* scale);  
 drawTo(bx + 12 \* scale, by + 6 \* scale);  
 lift(1);  
 break;  
 case 5:  
 drawTo(bx + 2 \* scale, by + 5 \* scale);  
 lift(0);  
 bogenGZS(bx + 5 \* scale, by + 6 \* scale, 6 \* scale, -2.5, 2, 1);  
 drawTo(bx + 5 \* scale, by + 20 \* scale);  
 drawTo(bx + 12 \* scale, by + 20 \* scale);  
 lift(1);  
 break;  
 case 6:  
 drawTo(bx + 2 \* scale, by + 10 \* scale);  
 lift(0);  
 bogenUZS(bx + 7 \* scale, by + 6 \* scale, 6 \* scale, 2, -4.4, 1);  
 drawTo(bx + 11 \* scale, by + 20 \* scale);  
 lift(1);  
 break;  
 case 7:  
 drawTo(bx + 2 \* scale, by + 20 \* scale);  
 lift(0);  
 drawTo(bx + 12 \* scale, by + 20 \* scale);  
 drawTo(bx + 2 \* scale, by + 0);  
 lift(1);  
 break;  
 case 8:  
 drawTo(bx + 5 \* scale, by + 10 \* scale);  
 lift(0);  
 bogenUZS(bx + 5 \* scale, by + 15 \* scale, 5 \* scale, 4.7, -1.6, 1);  
 bogenGZS(bx + 5 \* scale, by + 5 \* scale, 5 \* scale, -4.7, 2, 1);  
 lift(1);  
 break;  
   
 case 9:  
 drawTo(bx + 9 \* scale, by + 11 \* scale);  
 lift(0);  
 bogenUZS(bx + 7 \* scale, by + 15 \* scale, 5 \* scale, 4, -0.5, 1);  
 drawTo(bx + 5 \* scale, by + 0);  
 lift(1);  
 break;  
   
 case 111:  
   
 lift(0);  
 drawTo(70, 46);  
 drawTo(65, 43);  
   
 drawTo(65, 49);  
 drawTo(5, 49);  
 drawTo(5, 45);  
 drawTo(65, 45);  
 drawTo(65, 40);  
   
 drawTo(5, 40);  
 drawTo(5, 35);  
 drawTo(65, 35);  
 drawTo(65, 30);  
   
 drawTo(5, 30);  
 drawTo(5, 25);  
 drawTo(65, 25);  
 drawTo(65, 20);  
   
 drawTo(5, 20);  
 drawTo(60, 44);  
   
 drawTo(75.2, 47);  
 lift(2);  
   
 break;  
   
 case 11:  
 drawTo(bx + 5 \* scale, by + 15 \* scale);  
 lift(0);  
 bogenGZS(bx + 5 \* scale, by + 15 \* scale, 0.1 \* scale, 1, -1, 1);  
 lift(1);  
 drawTo(bx + 5 \* scale, by + 5 \* scale);  
 lift(0);  
 bogenGZS(bx + 5 \* scale, by + 5 \* scale, 0.1 \* scale, 1, -1, 1);  
 lift(1);  
 break;  
   
 }  
}  
   
   
   
void lift(char lift) {  
 switch (lift) {  
 // room to optimize !  
   
 case 0: //850  
   
 if (servoLift >= LIFT0) {  
 while (servoLift >= LIFT0)   
 {  
 servoLift — ;  
 servo1.writeMicroseconds(servoLift);   
 delayMicroseconds(LIFTSPEED);  
 }  
 }   
 else {  
 while (servoLift <= LIFT0) {  
 servoLift++;  
 servo1.writeMicroseconds(servoLift);  
 delayMicroseconds(LIFTSPEED);  
   
 }  
   
 }  
   
 break;  
   
 case 1: //150  
   
 if (servoLift >= LIFT1) {  
 while (servoLift >= LIFT1) {  
 servoLift — ;  
 servo1.writeMicroseconds(servoLift);  
 delayMicroseconds(LIFTSPEED);  
   
 }  
 }   
 else {  
 while (servoLift <= LIFT1) {  
 servoLift++;  
 servo1.writeMicroseconds(servoLift);  
 delayMicroseconds(LIFTSPEED);  
 }  
   
 }  
   
 break;  
   
 case 2:  
   
 if (servoLift >= LIFT2) {  
 while (servoLift >= LIFT2) {  
 servoLift — ;  
 servo1.writeMicroseconds(servoLift);  
 delayMicroseconds(LIFTSPEED);  
 }  
 }   
 else {  
 while (servoLift <= LIFT2) {  
 servoLift++;  
 servo1.writeMicroseconds(servoLift);   
 delayMicroseconds(LIFTSPEED);  
 }  
 }  
 break;  
 }  
}  
   
   
void bogenUZS(float bx, float by, float radius, int start, int ende, float sqee) {  
 float inkr = -0.05;  
 float count = 0;  
   
 do {  
 drawTo(sqee \* radius \* cos(start + count) + bx,  
 radius \* sin(start + count) + by);  
 count += inkr;  
 }   
 while ((start + count) > ende);  
   
}  
   
void bogenGZS(float bx, float by, float radius, int start, int ende, float sqee) {  
 float inkr = 0.05;  
 float count = 0;  
   
 do {  
 drawTo(sqee \* radius \* cos(start + count) + bx,  
 radius \* sin(start + count) + by);  
 count += inkr;  
 }   
 while ((start + count) <= ende);  
}  
   
   
void drawTo(double pX, double pY) {  
 double dx, dy, c;  
 int i;  
   
 // dx dy of new point  
 dx = pX — lastX;  
 dy = pY — lastY;  
 //path lenght in mm, times 4 equals 4 steps per mm  
 c = floor(4 \* sqrt(dx \* dx + dy \* dy));  
   
 if (c < 1) c = 1;  
   
 for (i = 0; i <= c; i++) {  
 // draw line point by point  
 set\_XY(lastX + (i \* dx / c), lastY + (i \* dy / c));  
   
 }  
   
 lastX = pX;  
 lastY = pY;  
}  
   
double return\_angle(double a, double b, double c) {  
 // cosine rule for angle between c and a  
 return acos((a \* a + c \* c — b \* b) / (2 \* a \* c));  
}  
   
void set\_XY(double Tx, double Ty)   
{  
 delay(1);  
 double dx, dy, c, a1, a2, Hx, Hy;  
   
 // calculate triangle between pen, servoLeft and arm joint  
 // cartesian dx/dy  
 dx = Tx — O1X;  
 dy = Ty — O1Y;  
   
 // polar lemgth © and angle (a1)  
 c = sqrt(dx \* dx + dy \* dy); //   
 a1 = atan2(dy, dx); //  
 a2 = return\_angle(L1, L2, c);  
   
 servo2.writeMicroseconds(floor(((a2 + a1 — M\_PI) \* SERVOFAKTORLEFT) + SERVOLEFTNULL));  
   
 // calculate joinr arm point for triangle of the right servo arm  
 a2 = return\_angle(L2, L1, c);  
 Hx = Tx + L3 \* cos((a1 — a2 + 0.621) + M\_PI); //36,5°  
 Hy = Ty + L3 \* sin((a1 — a2 + 0.621) + M\_PI);  
   
 // calculate triangle between pen joint, servoRight and arm joint  
 dx = Hx — O2X;  
 dy = Hy — O2Y;  
   
 c = sqrt(dx \* dx + dy \* dy);  
 a1 = atan2(dy, dx);  
 a2 = return\_angle(L1, (L2 — L3), c);  
   
 servo3.writeMicroseconds(floor(((a1 — a2) \* SERVOFAKTORRIGHT) + SERVORIGHTNULL));  
   
}

## 1.Bilgi Düzeyine Katkıları:

Herşeyden önce proje yönetiminin nasıl olması gerektiğini ve projenin sunumunun nasıl yapılması gerektiğini öğrendik. 4 Haftalık süreçte birçok devre elemanının kullanımıyla ilgili tecrübeler edindik.Devrenin kurulumu,gerekli kodlamanın yapılması,devrelerin şematik olarak gösterilmesi,fritzing,proteus kullanımı ve daha birçok konuda bilgi edindik.Bir robotun elektonik ve mekanik tasarımının nasıl olması gerektiğini öğrendik ve yaptığımız yanlışlar ile tasarım aşamasında yapılmaması gerekenleri farkettik.

PID kontrolü konusuna yoğunlaştık.Hata oranını minimize ederek, robotun sorunsuz bir şekilde çizgiyi tam anlamıyla takip etmesi için araştırmalar yapıp Kp,Kd ve Ki değerlerine deneme ve yanılma yöntemiyle uygun katsayıları bulmaya çalıştık.Bizi en zorlayan kısım PID kontrolüve robotun çizgi takip işlemini gerçekleştirirken hız ve açısal konumunun ayarlanması oldu. Test aşamasında gördüğümüz hatalarımızı düzelttik. 1.Proje sayesinde devre tasarımında devre elemanlarının zarar görmemesi için yapılması gerekenleri devre tasarımının nasıl olması gerektiğini ve bağlantıları öğrendik.

## 2.Teknolojik Katkıları:

Çizgi izleyen robotlar günümüzde gelişen teknolojiyle birlikte endüstriyel alanda bir süredir kullanılmaktadır.Genellikle lojistik ve otomasyon bölümleri içerisinde oldukça gereksinim duyulan niteliksiz insan gücü ile yapılan taşıma işlemlerini bir süredir çizgi izleyen robotlar yapmaya başlamışlardır.Günümüzde bu kadar yaygın olarak kullanılan çizgi takip eden robot tasarımını gerçekleştirmiş olmamız bize iş hayatımızda kazanım olarak döneceğini ve bizim için önemli bir tecrübe,deneyim olduğunu düşünüyoruz.

3.Ekip Çalışması Katkıları**:**

Ekip çalışmasının bize kattıkları:

* Her grup elemanının teknik bilgisini geliştirdi. konuya bütünlük açısından bakmamıza yardımcı oldu.
* Proje çalışanların sorun çözme alışkanlığını geliştirdi.
* Çalışanların birbiriyle iletişim alışkanlıklarını geliştirdi.
* Ekip, bir kişinin tek başına üretebileceğinden daha fazla fikir üretebilir ve çözüm önerisi geliştirebilir.Yani; bir kişinin çözüm üretmesinin zor olduğu durumlarda ekip olarak çalışmanın faydalı olduğunu gördük.
* Yanlış karar verme ve yanlış uygulama riski ekip çalışması ile en aza indirgenmiş oldu.

## 4.Aksayan Yönler:

Projeyi gerçekleştirirken karşılaştığımız sorunlar

Servo motorların mekanik parçalara takılırken sıkıntı yaşandı

Mekanik parçaların ve vidaların uyuşmaması

Mekanik parçaları hareket ederken zorlanması

Gibi sıkıntılar yaşadık

## 5.Görüş ve Öneriler:

BUGÜN ÜLKEMİZDE YAPILAN FAYDALI VE ÖĞRENCİLERİN YETENİKLERİNİ ORTAYA ÇIKARAN ÇALIŞMALARDAN BİRİ DE ROBOT YAPIM ÇALIŞMALARIDIR.BİZ DE BU ÇALIŞMADA GRUP OLARAK ÇALIŞMAKLA BİRLİKTE BAZI ŞEYLERİN NEDEN AMAÇLARINI ÖĞRENMİŞ OLDUK . BİZE GÖRE SAAT YAZAN ROBOT SADECE SAAT YAZMADA DEĞİL AYNI ZAMANDA BİRÇOK ÇALIŞMANIN ALT YAPISINDA KULLANILABİLİR.