# TEMEL ELEKTRONİK & ARDUİNO EĞİTİMİ

HASBİ SEVİNÇ

## Yazar Hakkında: Hasbi SEVİNÇ

Ocak 1992'de Balıkesir'de gözlerimi dünyaya açtım. 10 gibi küçük bir yaşta bilgisayar ile tanışmam beni bu alana yönlendirdi. Bilgisayarımın alınması ile birlikte Paint'te çizimler yaptım ve bu çizimleri PowerPoint'te bir araya getirerek kısa çizgi filmler yapmaya başladım. Bilgisayarımı formatlamak için gelen Mert Abimin önerisiyle FLASH animasyon programını kullanmaya ve yaşıma göre güzel animasyonlar hazırlamaya başladım. 15 Yaşımda eve internetin bağlanması çalışma alanımı genişletti ve programlamaya yöneldim. Önce C programlamaya başladım ve kısa sürede arkadaşlarımın ihtiyaç duyduğu çeşitli programlar yazdım. Web sitelerine merakımın olması nedeniyle kendime de bir tane yapmak istedim. Bu yüzden 16 yaşımda ASP programlamaya başladım. ASP ile web sitesi yazmak yerine daha çok bot programları yazdım. Bu programlar ile web sitelerimi güncel tutarak kendime maddi destek sağladım. Fakat bu becerilerimi iyi bir üniversite ile taçlandırmadıktan sonra bir anlamı olmadığını fark ettim. Bunun için 17-18 yaşlarımda kendimi tamamen derslere adadım. Lise 1 ve 2 de hiç ders çalışmamama rağmen son sınıfa doğru bütün eksiklerimi kapattım ve YGS/LYS girdim ( Bu arada PHP programlamayı da ileri düzeyde öğrenim). Tek hedefim olan İTÜ Elektronik-Haberleşme Mühendisliğini kazandım. Üniversitemin ilk yılında kendimi İngilizce ve temel elektronik üzerine geliştirdim. Bu senede ayrıca CCS C ile PIC programlamayı öğrendim. Hazırlığı atlayıp bölüme başladığımda İTÜ'deki projeleri inceledim. İlk senemde iki projeye girdim ve girdiğim projelerden birinde elektronik takım kaptanı oldum. Fakat bu projeler başarısızlıkla sonuçlandı. İkinci sınıfta kendimi daha da geliştirmek için kendi projelerimi yaptım. Yaptığım projeler ile yarışmalara katıldım ve birkaç televizyon kanalına röportaj verdim. Arduino ile ilk tanışmam da bu zamanlardadır. Üçüncü sınıfın başında birinci sınıfta başarısız olduğumuz projeyi tekrar yapmaya ve bu projede takım kaptanı olmaya karar verdim. Bu proje ile NASA'nın sponsorluğunda düzenlenen 'Cansat Model Uydu' yarışmasına katıldık. 13 farklı ülkeden 59 proje takımını geride bırakarak rekor puanla Dünya Birincisi olduk. Bu projeyi yıllar sonra bile ilk günkü gibi hatırlayacağımdan çok eminim. Bu sene de 'İTÜ Roket Takımını' kurduk. Amacımız yüzde yüz yerli bir roket üretmek ve Amerika'da düzenlenen roket yarışmasında ülkemize derece getirmektir.

Kusura bakmayın… Laf lafı açtı, yazım biraz uzun oldu.

Beni takip etmek isterseniz iletişim bilgilerim aşağıdadır.

**facebook.com/flashasbi**

**hasbi.sevinc@gmail.com**

**hasbisevinc.com**

## İTÜ ARI TEKNOLOJİ GELİŞTİRME KULÜBÜ [ARIGE]

2013 yılında, İstanbul Teknik Üniversitesi Elektrik Elektronik Fakültesinde kurulan Arı Teknoloji Geliştirme Kulübü [ARIGE], kişisel gelişim ve kariyer odaklı çalışmalara odaklanan yeni bir kulüptür. ARIGE, kısa sürede başarılı projeler gerçekleştirmiştir

ARIGE'nin vizyonu; üniversite öğrencilerinin yeni teknolojik gelişmeleri yakından takip etmesini, derslerde öğrendikleri teorik bilgiler ile bu gelişmeleri harmanlayarak, daha inovatif teknolojileri yerli olarak projelendirip üretmelerini sağlamaktır. Bu projelerden çıkabilecek yayınlar aracılığı ile akademik birikime de katkı sağlanması hedeflenmektedir.

ARIGE, bu süreçte gerekli sanayi- üniversite işbirliğini sağlayıp, eğitim seminerleri düzenleyerek ilgili projelerin endüstriyel bir boyut ve ticari değer kazanmalarına yardımcı olur.

Kulüp aynı zamanda kendisinin oluşturacağı proje ve komiteler içerisinde öğrencilerin kalifiye bir şekilde yetişmesini sağlayarak İTÜ'deki diğer öğrenci projelerine iş gücü yaratmayı hedeflemektedir. Bu sayede projeler arası iletişim de kuvvetlendirilmiş olacaktır.

Bu amaçlar doğrultusunda, ARIGE, üyelerinin akademik hayatlarında öğrendikleri teorik bilgileri, çalışma hayatına aktarabilmesini, ders dışı zamanlarda geliştirilecek olan projeler ile üretim ve yönetim konusunda bilgilenmesini, deneyimli ve araştırmacı bir ruha sahip olarak mezun olmasını hedefler.

## ÖNSÖZ

Hiçbir kitabın önsözünü okumam. Fakat okuyanlar için kitabı tanıtan kısa bir yazı yazmak istedim. Türkiye ne yazık ki teknolojiyi geriden izleyen bir toplum konumundadır. Bu durumdan kurtulmamız için herkesin bildiğini paylaşması gerektiğini düşünmekteyim. Bu yüzden bu kitabı yazmaya karar verdim. Kitapta anlattıklarımı anlatan web siteleri internette mevcut. Ben bu kitapta temel düzeyde lazım olacak tüm bilgileri bir yerde toplamak istedim.

Umarım kitabım sizler için yararlı olur…

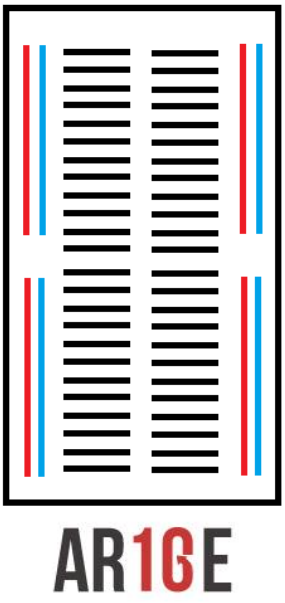
# İÇİNDEKİLER

## Neden Arduino?

Arduino Atmel marka işlemcilerin kullanıldığı hazır bir devre kartıdır. Açık kaynak kodludur ve isteyen kişi kendi Arduino'sunu yapabilir. Arduino projelerde kullanım kolaylığı sağlar ve projelerin daha hızlı ve stabil olarak yapılmasını sağlar. Arduino yapı ve özelliklerine göre çeşitli türlerden oluşmaktadır. Biz projemizde Arduino Uno modelini kullanacağız.

Arduino Üzerinde USB bağlantısı, güç bağlantısı ve giriş çıkış pinleri bulunmaktadır. USB bağlantısı karta program yollama, bilgisayar ile kart arasındaki bilgi aktarımına ve kartı beslemek için kullanılır. Güç bağlantısı da aynı şekilde harici beslemeler için kullanılır. 9 Voltluk besleme önerilir. USB'den beslemek bu besleme türüne göre daha risklidir. Çünkü bilgisayarın USB çıkışı yeterli akım vermeyebilir Arduino için (motor ile ilgili uygulamalarda özellikle). Arduino ile programlamaya başlamadan önce bazı temel elektronik bilgilerimizi gözden geçirelim.

## Temel Elektronik




### Breadboard

Resimde görüldüğü gibi breadboardımızın iki kenarında aşağıya doğru uzanan delikler vardır. Bu delikler besleme kanalımızdır. Aşağıya doğru inen çizgilere karşılık gelen delikler kısa devre durumundadır. Yani sol üstteki kırmızıdan bağlanan bir kablo aynı çizgi üzerinden bağlanacak kablolar ile birleşiktir. Ayı şekilde orta kısımdaki yatay çizgilere karşılık gelen delikler de birbirleri ile bağlantılıdır. Fark ettiyseniz yatay çizgilerin ortasından bir yarık geçmektedir. Bu yarığın amacı entegrelerimizi kolaylıkla takabilmemizi sağlamasıdır.

## Direnç

Hemen hemen herkesin bildiği bilgilerimizi de gözden geçirelim. V=I*R formülünü inceleyelim; bu formüle göre sabit bir voltaj altında direncimizi arttırırsak Akımımız azalmaktadır. Bu özellik ile fazla akım çekmemesini istediğimiz elemanların önüne direnç koymalıyız.

## Voltaj Bölücü

Giriş voltajının istenilen voltaja düşürülmesini sağlayan devredir. Şekildeki gibi kurulur. Çıkış voltajı R1 ve R2 dirençlerine bağlıdır. Vout = Vgiriş*R2/(R1+R2) şeklinde yazılır. Örneğin R1=10k R2= 10k ohm olarak seçilir ve giriş voltajımız da 5 volt olursa, çıkış voltajımız = 5*10K/(10K+10K) = 2.5 Volt olarak bulunur.

## Diyotlar

Kullanım alanlarına göre çeşitli diyotlar bulunmaktadır. Led, fotosel, zener diyotlar bunlara örnek olarak verilebilir. Lise bilgilerimizden de bildiğimiz gibi diyotlar tek yönde geçirgendir. Şekildeki resimde akım geçirme yönü belirtilmiştir. Fakat ne yazık ki gerçek dünyada diyotlar kağıt üzerindeki gibi ideal davranmazlar. Diyot akım geçirme yönünde kullanılsa bile bir voltaj farkına neden olmaktadır. Bu voltaj farkı diyotun yapısına göre değişmekle beraber genellikle 0,6 – 0,7 Volt arasında olmaktadır.

## Zener Diyotlar

Genellikle devreye diğer diyotlardan farklı olarak ters bağlanırlar. Ters (tıkama) yönünde oluşturdukları gerilim farkı sayesinde kol üzerindeki gerilim sabitlenmektedir. Devre üzerinde kullanımı şekildeki gibi gösterilmiştir. Voltaj farkı diyotun yapısına göre değişmektedir. Genellikle 5.1 Voltluk diyotlar kullanacağız. Bu diyotlar ile devre üzerindeki fazla gerilimi ve olası dalgalanmaları engellemeyi umuyoruz.

## Tranzistör

Girişine uygulanan sinyali kuvvetlendiren devre elemanıdır. Aynı zamanda anahtarlama elemanı olarak da kullanılmaktadır.

## Temel Yazılım Bilgileri

Arduino programlamaya geçmeden önce temel yazılım kodlarına bakalım. Projelerimizde kullanacağımız bazı fonksiyon ve kod yapılarını tanıtacağım.

**Koşullar (if-else-elseif):** Yazılım dillerinin en temel komutlarından birisidir. Elinizde bir durum var ve bu durum doğru ise a işini yapmasını eğer yanlış ise b işini yapmasını istiyorsanız, bu kod sistemi tam bu işe uygun. Kısaca kod şemasına bakalım.

İf( a == 5 ){

//Buraya doğru durumda çalışması istenilen kodlar

}else{

A'nın 5 olmadığı durumda çalışacak kodlar.

}

a == 5 burada bizim koşulumuzu oluşturmaktadır. Dikkat edilmelidir ki iki tane eşittir kullanılmaktadır. Yani denk midir anlamına gelmektedir. Koşul alanında kullanabileceğim ifadeler,

| | | | | |
|---|---|---|---|---|
| == | Denk ise | | != | Eşit değilse |
| < | Büyüktür | | > | Küçüktür |
| <= | Büyük veya eşitse | | >= | Küçük veya eşitse |
| Koşul1 &&Koşul 2 | Ve | | Koşul1 \|\|Koşul 2 | Veya |

Koşullarımıza else kullanımı zorunlu değildir. Sadece if kullanımı yapılabilir, böylece koşul doğru değilse extradan bir kod çalışmaz. Birden fazla koşulumuz var ise elseif ile yeni koşullar ekleyebiliriz.

İf( a == 1 ){

// a = 1 durumunda burası çalışır

}

```
elseif (a == 2 ) {

// a = 2 durumunda burası çalışır

}

elseif (a == 3 ) {

// a = 3 durumunda burası çalışır

}
```

**For döngüsü:** Yazdığımız kodların belli bir süre tekrar etmesini isteyebiliriz. Bunun için döngüler kullanmalıyız.

```
For(int i =0; i < 10; i ++){

// burası 10 kere okunacak

// program buraya her uğradığında i değeri bir arttırılacak

// i değeri 10'a ulaşana kadar döngü devam edecek

}
```

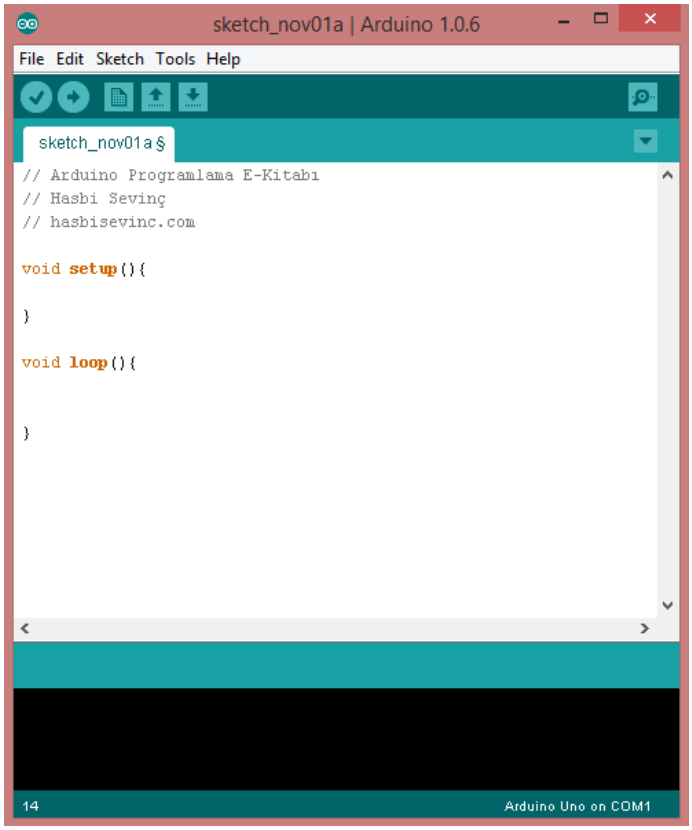
**While döngüsü:** For gibi while kodları da döngü amacıyla kullanılır.

```
b = 20;

while( b > 10){

// b değişkeninin 10'dan büyük olduğu durumlarda döngü devam eder

b = b – 1 ; // her döngüde b'nin değerini bir azalttık

// dikkat edilmelidir ki eğer b'nin değerini değiştirmeseydik

// döngü koşulu hep doğru olacağından program burada takılı kalacaktı

}
```

## Arduino’nun Kurulumu ve Arduino Programı

Arduino’yu kullanmaya başlıyoruz. Öncelikle arduinomuzun USB kablosunu arduino’ya bağlayalım. Daha sonra bilgisayarımıza takalım. Win 7 ve üst sürümlerinden arduino otomatik olarak yüklenecektir. Biraz bekledikten sonra artık arduino bilgisayar tarafından tanınacaktır. Arduino’nun bilgisayar tarafından tanınıp tanınmadığını görmek için aygıt yöneticisinden kontrol edelim ve arduino'nun bağlı olduğu COM Portunu aklımızda tutalım ( aygıt yöneticisi:  bilgisayarıma sağ tıklayın -> yönet -> aygıt yöneticisi )

Arduino’yu bilgisayarımıza tanıtırken bizde bir yandan kullanacağımız programı indirelim. Bunun için http://arduino.cc/en/Main/Software adresine girelim ve Arduino programını indirip programı kuralım. Programı hatasız bir şekilde yükledikten sonra programı açalım.




Program çalıştığında yandaki gibi bir ekran sizi karşılayacak. Bu ekranın ortasındaki kısma kodlarımızı yazıyoruz. Programın sağ en altındaki yer bize Arduino Uno üzerinden çalıştığımızı ve arduino’nun COM1 portuna bağlı olduğunu gösteriyor. Eğer bu doğru değilse bu ayarları değiştirmemiz lazım öncelikle.

**Tools menüsü:** Buradan kullandığımız arduino türünü ve arduinomuzun bağlı olduğu COM portunu seçelim. Eğer bunu bilmiyorsak aygıt yöneticisinden bakabiliriz.

 **Program compile:** Bu buton ile yazdığımız programı kontrol edebiliriz. Eğer kodda hata varsa alttaki siyah bölümde turuncu yazı şeklinde yaptığımız hata ve satırı yazacaktır. Hatırlatma: Bulunduğumuz satırın sayısı sol atta yazmaktadır.

 **Program Compile & Upload:** Bu buton ile önce yazdığımız kod derlenir. Eğer kodda hata yoksa yazılan kod Arduino'nun anlayacağı dile çevrilir ve otomatik olarak Arduino'ya atılır. İşlem sırasında ilerleyen çubuktan işlem durumunu görebilirsiniz. Ayrıca Tam program atılma sürecinde Arduino üzerinde bulunan Tx ve Rx LED'leri hızlı bir şekilde yanıp sönecektir.

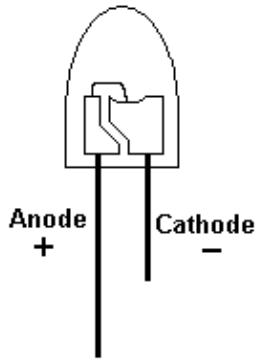
Programı bir kere Arduino'ya atmanız yeterli olacaktır.

 **Serial Monitor:** Yeni pencere açan bu buton ile Arduino'dan yolladığımız verileri görebiliriz. İlerleyen konularımızda bu butonu sıkça kullanacağız.

Şimdiden Arduino çalışmalarınızda başarı dilerim…

Haydi Başlayalım.

## Arduino ile Yanıp Sönen Led Yapımı




Öncelikle LED bağlamayı öğrenelim. LED'in ayaklarına bakıldığında birinin uzun birinin kısa olduğu görülmektedir. Buradan anlamamız gereken şey, uzun ayağın + uca kısa ayağında da – uca bağlanması gerektiğidir. Yani uzun ayağımızı arduinoya bağlayacağız. Fakat ayaklar arasında bir bağlantı yoksa, LED'in içindeki kısa olan yere bağlı ayak + ucu, diğer ayak – ucu göstermektedir. LED'imizin çalışma akımı vardır. Bunu sağlaması ve fazla akım çekmemesi için LED'in artı ucu ile arduino(yada +5 volt besleme) arasına 220 ohm'luk bir direnç bağlamalıyız. Aksi taktirde LED'imiz patlayabilir. İçerisinden çıkan gaz sağlık için zararlı olabilir.

Şimdi programlamaya giriş yapabiliriz. Arduino programları iki ana fonksiyondan oluşur.

**void setup() fonksiyonu:** Bu fonksiyon program ilk açıldığında bir kere çalışır ve gerekli kalibrasyon, setup komutlarını buraya yazarız.

**void loop() fonksiyonu:** diğer programlama dillerinden alışık olduğumuz main() fonksiyonu gibidir. Farklı olarak loop fonksiyonu işlevi bitince tekrar baştan başlar yani sonsuz bir döngüdür aslında.

Arduino Programlama Şeması:

```
// İlk başta kütüphaneleri ekleyebiliriz

// Global cinsteki değişkenlerimizi tanımlayabiliriz

// Fonksiyonlarımızı burada yazabiliriz

void setup(){

// ilk başta çalışmasını istediğimiz kodlar buraya yazılır

}

void loop(){

// sonsuz döngü şeklindeki main fonksiyonumuz
```

// programı buraya yazıyoruz

}

Arduino'yu test etmek ve ilk programımızı yazmak için bu projeyi yapabiliriz. 1 Saniye aralıklar ile LED'imiz yanıp sönecektir.

```
int LED = 10; // burada arduinonun 10. Ayağına ledimizi bağladığımızı
söylüyoruz

void setup(){
  pinMode(LED, OUTPUT); // LED pini yani Arduino'nun 10. Ayağı çıkış
yapıldı
}

void loop(){
  digitalWrite(LED,HIGH); // LED'in bağlı olduğu pinden 5 volt gerilim
sağlandı
  delay(1000); // 1000 ms yani 1 saniye program duruyor
  digitalWrite(LED,LOW); // LED söndürülüyor
  delay(1000); // bir saniye bekliyoruz
}
```

## Arduino ile Kara Şimşek Devre

Arduino'ya pratik olarak kara şimşek diye tabir edilen, paralel bağlı ledlerden oluşan örnek devremizle başlayalım. Devremizi istediğimiz kadar ledle oluşturabiliriz. Biz bu örnekte 5 adet led ile kurulum yapmış gibi davranacağız.

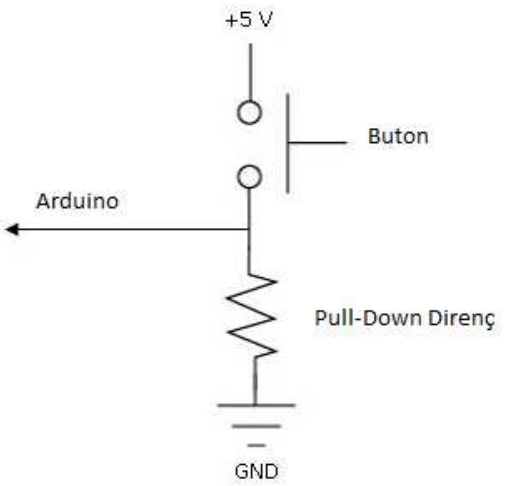
```
void setup(){
pinMode (13, OUTPUT);
pinMode (12, OUTPUT);
pinMode (11,OUTPUT);
pinMode (10, OUTPUT);
pinMode (9, OUTPUT);
}
void loop(){
digitalWrite (9, HIGH);
delay (10);
digitalWrite (9, LOW);
digitalWrite (10, HIGH);
delay(10);
digitalWrite (10, LOW);
digitalWrite (11 , HIGH);
delay(10);
digitalWrite (11, LOW);
digitalWrite (12, HIGH);
delay(10);
digitalWrite (12, LOW);
digitalWrite (13, HIGH);
delay(10);
digitalWrite (13, LOW);
}
```

Kara şimşek uygulamasının çalışma mantığının kavranması için yukarıda verilen kodların açıklayıcı olduğunu düşünmekteyim. Fakat bu kodlar başlangıç düzeyindedir ve gereksiz olarak her led tek tek kontrol edilmiştir. Bir sonraki aşama olarak yukarıdaki uygulamayı yapan fakat daha profesyonel olan bir program yazalım.

```
const int ledPini [] = {9,10,11,12,13};
void setup () {

for(int i=0; i<5;i++)
{
  pinMode(ledPini [i], OUTPUT); // LED pinlerini çıkış olarak tanımladık
}
}
void loop() {
  for(int i=0; i<5; i){
    digitalWrite(ledPini[i],HIGH);
    delay(50);
    digitalWrite(ledPini[i],LOW);
  }
  for(int j=4;j>-1; j--)
  {
    digitalWrite(ledPini[j],HIGH);
    delay(50);
    digitalWrite(ledPini[j], LOW);
  }
}
```

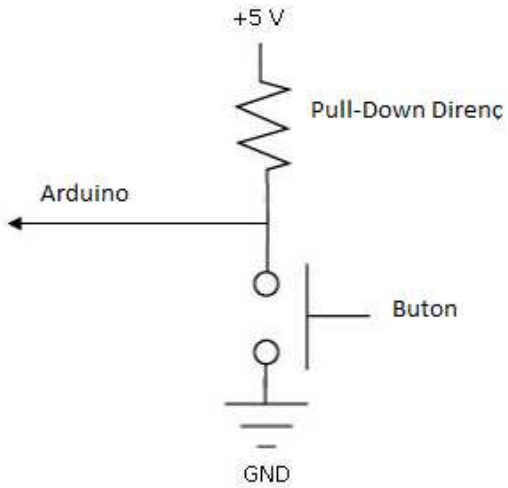
## Arduino ile Buton Kullanımı

Uygulamalarımızda çeşitli görevler için butonlar kullanacağız. İlk olarak butonun çalışma mantığını inceleyelim. Kullanıcı butona tıkladığında butonun iki ucu kısa devre olur ve böylece akım akmaya başlar. Kullanıcı butondan elini çektiğinde devrenin önceki durumuna dönmesi için pull-up/pull-down direnç sistemi kullanılır. Kullanılan bu dirençler ile buton elektriksel olarak basılı kalmaktan kurtarılır.




### Pull-Down Direnç

Butona basıldığında 5 Volt arduinonun input ayağına ulaşır. Fakat butondan elimizi kaldırdığımızda arduinoun pininde 5 volt gerilimi kalmaktadır. Bu durumdan kurtulmak için genellikle 10K ohm değerinde bir direnç arduinoun input ayağından toprağa bağlanır.




### Pull-up Direnç

Butona basılmadığı durumlarda arduinonun input ayağı 5 volttadır. Butona basıldığında akım arduinonun input ayağı yerine toprağa ulaşmaktadır. Pull-Down direnç sisteminin tam tersi şeklinde çalışmaktadır. Direncin konulma nedeni butona basıldığında 5 Voltun doğrudan toprağa (-uça) ulaşmasını engellemektir. Genellikle 10K değerinde direnç kullanılmaktadır.

Bu kadar teorik bilginin yeterli olduğunu düşünüyorum. Butona bastığımızda lambanın yanmasını ve elimizi çektiğimizde sönmesini sağlayan bir program yazalım.

```
const int buttonPin = 2;     // Butonun pin numarası
const int ledPin =  13;  // Ledin bağlandığı pin numarası
int buttonState = 0; // Butonun durumu
void setup() {
  pinMode(ledPin, OUTPUT);     // Ledin bağlı olduğu pini çıkış olarak
ayarlıyoruz.
  pinMode(buttonPin, INPUT);     // Butonun bağlı olduğu pini çıkış olarak
ayarlıyoruz.
}
void loop(){
  buttonState = digitalRead(buttonPin); // butonun basılı olup olmadığını
kontrol ediyoruz.
// Buton basılı ise bize HIGH değil ise LOW olarak geri dönmektedir (1 veya
 0)
  if (buttonState == HIGH) {     //buton basılı mı?
    digitalWrite(ledPin, HIGH);  // ledi yak
  }
  else {
    digitalWrite(ledPin, LOW);  // buton basılı değil ledi söndür
  }
}
```

Ark olaylarını engellemek için digitalWrite kodlarından sonra az bir bekleme verilebilir, delay(100) şeklinde.

Buton kontrol etmeyi öğrendiğimize göre bir adım daha ileriye gidelim ve sayaç yapalım. Fakat bunun için öncelikle bilgisayar ile arduinomuzu nasıl haberleştireceğimize bakalım.

## Arduino ile SeriPort Üzerinden Bilgisayara Veri Yollama

Gerek arduinonun kontrolü gerekse işlenen değerlerin kullanıcıya aktarılması için arduinomuz ile bilgisayarımız arasındaki haberleşmeyi yapmamız gerekir. Bu haberleşme arduinonun RX ve TX ayaklarından gerçekleşecektir. Haberleşme için her zamanki gibi arduinoyu USB üzerinden bilgisayarımıza takmamız yeterli olmaktadır. Haberleşme için setup fonksiyonumuza başlatma kodunu yazmayı unutmayalım. Aşağıdaki kod ile bilgisayara her saniye artan sayımızı yolluyoruz.

```
void setup() {
  Serial.begin(9600); // bilgisayar ile arduinomuzun haberleşmesini
başlatıyoruz.
// Bilgisayarın ve arduinonun aynı hızda çalışması için 9600 yazdık.
// Yani saniyede 9600 tane bit transferi olacaktır.
}
int sayici = 0;
void loop() {
  Serial.print("Sayicimizin degeri= "); // Ekrana mesajımızı yazdırıyoruz.
  Serial.println(sayici); // sayaç değerimizi ekrana yazdırıp yeni satıra
geçiyoruz.
  delay(1000);        // Bir saniye bekle
}
```

Bütün işlem bu kadar kolay. Bundan sonrası tamamen size kalmış. Bu bağlantı ile yazdığınız programı kolaylıkla test etme debug etme imkanı sağlamaktasınız.

Şimdi daha önce öğrendiğimiz buton kontrolüyle haberleşmemizi birleştirelim ve butona her basıldığında artan bir sayaç yapalım.

```
const int buttonPin = 2;     // Butonun pin numarası
int sayici = 0; //sayacimiz
int buttonState = 0; // Butonun durumu
void setup() {
  pinMode(buttonPin, INPUT);
  Serial.begin(9600);
}
void loop() {
  buttonState = digitalRead(buttonPin);
  if (buttonState == HIGH) {
    sayici ++; // sayici = sayici + 1;
    Serial.print("Sayicimizin degeri= "); // Ekrana mesajımızı
yazdırıyoruz.
    Serial.println(sayici); // sayaç değerimizi ekrana yazdırıp yeni satıra
geçiyoruz.
    while(buttonState == HIGH){ // Butondan elini çekene kadar program
burada kalır
  // Böylece butona her basıldığında sadece bir kere değer arttırılır
      buttonState = digitalRead(buttonPin);
    }
```

```
  }
}
```

Hep arduino mu veri yollayacak, biraz da karşıdan veri bekleyelim. Bu kod ile bilgisayardan gelen veriler arduinoda işlenmektedir. Unutmayın ki tüm veriler karakter olarak tek tek gelmektedir.

```
int incomingByte = 0;   // gelen veriler
void setup() {
        Serial.begin(9600);
}
void loop() {
  if (Serial.available() > 0) { // bilgisayardan veri gelmesini bekliyoruz
      incomingByte = Serial.read(); // gelen veriyi oku
      Serial.print("gelen veri:  ");
      Serial.println(incomingByte, DEC); // gelen veriyi bilgisayara geri
yolla
  }
}
```

## Arduino ile Bluetooth Haberleşmesi

Temel olarak yaptığımız iş bilgisayar ile arduino üzerinden haberleşmeye benzemektedir. Tek fark donanımsal olarak bluetooth modülümüzü arduinomuza bağlayacağız. Piyasada çeşitli modelleri bluetooth modülleri genellikle 4 ayağa sahiptir. Üzerinde yazanlardan da anlaşılacağı gibi 2 besleme 2 haberleşme ayağı vardır. Beslemeleri modülümüzün datasheetine göre 5 veya 3.3 volta bağlayalım. Hatalı bağlantı yapılması durumunda modülümüz çöp olabilir. Geriye kalan 2 ayak arduinomuza bağlanacaktır. İlk uygulama olarak bu ayakalrımızı arduinomuzun Rx ve Tx ayaklarına bağlayalım. Bluetoothun Tx ayağı arduinonun Rx ayağına, Bluetoothun Rx ayağı ise arduinonun Tx ayağına bağlanacaktır. Yani ters olarak bağlanması gerekmektedir. Hep yapılan ve yapılamaya devam edecek bir hata var burada. Arduinomuzun Rx ve Tx ayaklarına bir şeyler bağlı iken arduinoyu bilgisayara bağlayarak kod atmaya çalışırsanız hata alırsınız. Çünkü bu ayaklar aynı zamanda bilgisayarın kullandığı ayaklardır ve oraya başka bir modülün bağlı olması iletişiminizi bozar. Eğer bu ayakları kullanacaksak kod atacağımız zaman bu kabloları sökmeyi unutmayın.

İşin arduino üzerindeki kısmı çok zor değil. Fakat bluetooth modülümüzü bilgisayarımızla eşleştirmemiz gerekmektedir. Bluetoothumuzun besleme bağlantılarını yaptıktan sonra kırmızı ışığı yanıp sönecektir. Bu ışık bağlantı bekliyorum anlamına gelmektedir. Bilgisayarımızın bluetoothundan arama yaparak modülümüzü bulalım. Cihazları eşleştirelim. Şifre sorarsa genellikle ilk şifremiz 1234 olmaktadır. Eğer düzgün bir şekilde bağlantımızı yaptıysak kırmızı ışık sürekli yanacaktır. Bu kısım ile ilgili daha detaylı bilgiyi aşağıdaki linkten ulaşabilirsiniz. Modül için yeni bir COM Port açılacaktır ve bu modülü ilerleyen programlarımızda kullanacağız.

http://www.mcu-turkey.com/stm8s-hc-06-bluetooth-modul-ile-haberlesme-uygulamasi/

Artık kod yazalım. Aslında çok farklı bir şey yapmayacağız. Seri haberleşme konusundaki örnek kodu yükleyebiliriz arduinomuza.

```
void setup() {
  Serial.begin(9600);
}
int sayici = 0;
void loop() {
  Serial.print("Sayicimizin degeri= ");
  Serial.println(sayici);
  delay(1000);
```

```
}
```

Programımızı arduinomuza attık ve bluetooth modülünün Tx ve Rx ayaklarını dikkatlice arduinoya bağladık. Hala modüldeki kırmızı ışık yanıp sönmektedir. Çünkü bilgisayarın modüle bağlanması gerekmektedir (Master -> Slave ilişkisi)

Bu adımda ek bir programa ihtiyaç duyuyoruz. Bu program bilgisayarımızın COM Portlarını dinleyen programdır. Çok özel bir program olmasına gerek yok piyasada bu işi yapan çeşitli programlar vardır. Benim önerim Tera Term'dür. Aşağıdaki linkten indirebilirsiniz.

http://download.cnet.com/Tera-Term/3000-20432_4-75766675.html

Programı çalıştıralım ve modülümüze bağlanalım. Bağlantı ayarlarında daha önce modülü bilgisayara tanıttığımız COM portunu seçelim. Her şey doğru ise artık kırmızı ışık sürekli yanmakta ve veri akışımız gerçekleşmektedir. Eğer veri akışı yok ise ayarlarımızı kontrol edelim ve diğer bluetooth COM portunu seçelim, tekrar deneyelim. Umarım veri akışını sağlayabilmişsinizdir. Tüm adımlar dikkatlice yapılmalıdır.

Şimdi şunu sorabilirsiniz ' Her seferinde Tx Rx kablolarıyla mı uğraşacağız?' Cevabımız HAYIR.

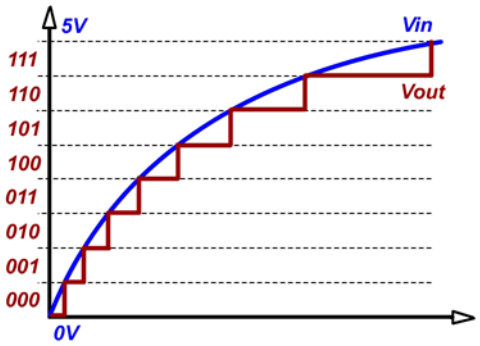
Arduinomuza yeni haberleşme ayakları tanımlayabiliriz. Böylece bilgisayar haberleşmesi ile bir çakışma gerçekleşmez.

Aşağıdaki kod ile artık modülümüzü bağlayacağımız ayaklar 10 ve 11dir.

```
#include <SoftwareSerial.h>
SoftwareSerial yeniHaberlesmeKanali(10, 11); // 10 = RX, 11 =  TX yani 10 -
> bluetooht'u Tx'ine 11 -> Rx'ine
void setup()
{
  Serial.begin(9600); // bilgisayar ile haberleşmemiz
  yeniHaberlesmeKanali.begin(4800); // yeni haberleşme kanalımız
}
void loop()
{
  Serial.println("Bu yazi bilgisayara");
  yeniHaberlesmeKanali.println("Bu yazi bluetootha");
  delay(100);
}
```

Yukarıdaki kod yardımı ile arduino hem bluetootha hemde bilgisayara farklı mesajlar yollamaktadır. Mesajları görmek için Tera Term'ü ve arduinonun kendi Serial Monitor'ünü açabilirsiniz.

## Arduino ile Analog Veri Okuma

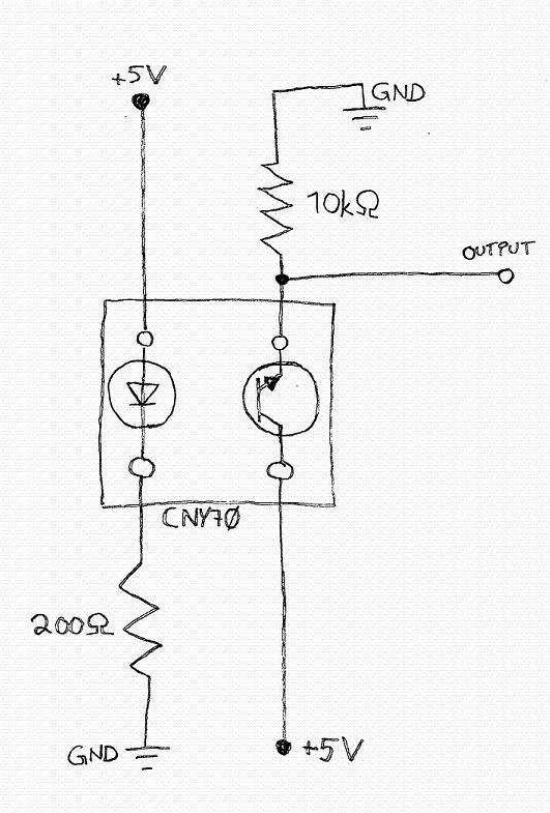



Gerçek dünyada her şey analog haldedir. Fakat dijital dünyada bu mümkün değildir. Analog verilerin saklanabilmesi ve işlenebilmesi için dijital hale getirilmesi gerekmektedir. Fazla teoriye girmeyi planlamıyorum fakat olayın nasıl olduğunu anlamanız için yandaki resmi paylaştım. Şöyle düşünelim 0 ile 5 volt arasında değişen bir girişimiz var. Bu girişin değerlerini Arduino'ya tanıtmak için ADC (Analog Digital Convert) işlemini kullanacağız. 0 Volta 0 değerini vereceğiz ve 5 Volta da 1023 değerini vereceğiz. Yani 10 bit çözünürlüklü bir ADC kullanacağız. Oran liner olarak devam etmektedir. Yani girişimizdeki 0,004 Voltluk bir değişim dijitalde 1 değere karşılık gelecektir. Yani 0,44 Volt => 100 değeri.

Arduino bu dönüşümü bizim için kendisi yapmaktadır. Fakat ölçüm yapacağımız kabloları Arduino'nun analog ayaklarına bağlamamız gerekmektedir. Girişimizdeki analog değeri dijital olarak bilgisayardan okuyalım:

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  int sensorDegeri = analogRead(A0); // Arduino'nun A0 ayağına bağlanan
kablodaki gerilim ölçülüyor
  Serial.println(sensorDegeri); // Okuduğumuz değer ekrana yazdırılıyor
  delay(1); // Düzgün çalışabilmesi için kısa bir bekleme veriyoruz.
}
```

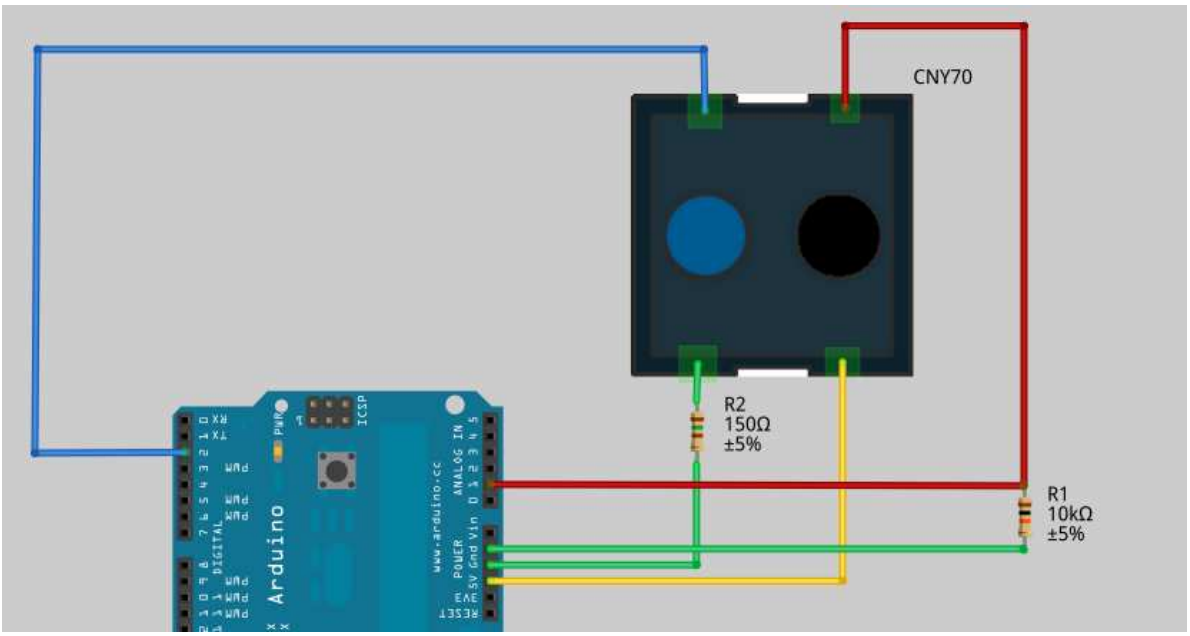
## Arduino ile Siyah Beyaz Kontrolü




Hemen hemen herkesin aklına ilk gelen uygulama olan çizgi izleyen robotun temel taşını oluşturan siyah beyaz sensörleri inceleyelim. Kitabımızda piyasada bu iş için kullanılan ve rahatlıkla bulabileceğiniz CNY70 sensörünü inceleyeceğiz. Piyasadaki diğer sensörler de aynı man tıkla çalışmaktadır.

**CNY70 Nasıl Çalışır:** Üzerinde kızılötesi ışık yayan ve bu ışığı geri toplayabilen iki LED bulunmaktadır. Kızılötesi LED'den çıkan ışın yüzeye çarpar ve toplayıcı LED'e gelir. LED bu ışığın şiddetini ölçer. Yüzeyin siyah veya beyaz olması yansıyan ışığın şiddetini değiştirir. Bu da bizim ihtiyacımız olan siyah beyaz ayrımını sağlar. Sensörümüz analog olarak çalışmaktadır. Yani bi önceki konuda öğrendiğimiz ADC-analog okuma- fonksiyonunu kullanacağız.

Öncelikle devremizi kurmamız gerekiyor. Bunun için delikli pertinaks'a yandaki gibi devremizi kuralım. Sensörü yerleştirirken ayaklarının şekildeki gibi olmasına dikkat ediniz.



Örnek olarak ilk başta bir tane sensörün bağlantılarını yapalım ve tek sensör ile bir proje gerçekleştirelim. İlerleyen projelerimizde 3 tane sensörü birleştirip çizgi izleyen basit bir robot yapacağız.

Resimdeki bağlantıları breadboardımıza kuralım. Dikkat edin ki bağlantıları yanlış bağlamayın. Sensörün karo-baklava dilimi şeklinde breadboarda konması kablolamayı kolaylaştırır.

Resimdeki gibi kurulumu gerçekleştirdiğimize göre kod yazmaya başlayalım.

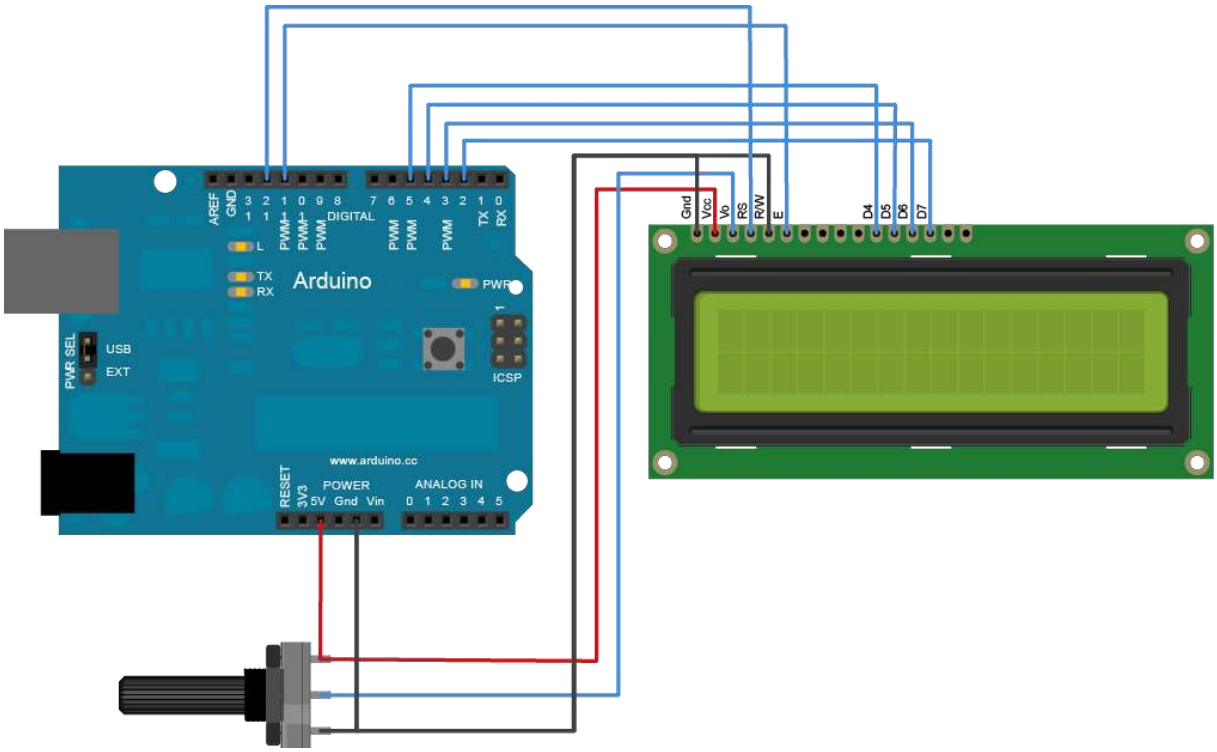
```
int referansDegeri = 800; // siyah beyaz için eşik değeri
void setup() {
  Serial.begin(9600);
}
void loop() {
  int sensorDegeri = analogRead(A1); // Arduino'nun A1 ayağına bağlanan
kablodaki gerilim ölçülüyor
  Serial.print(sensorDegeri); //Okuduğumuz değer ekrana yazdırılıyor
  if (sensorDegeri > referansDegeri){
    Serial.println(" Siyah");
  }
  else{
    Serial.println(" Beyaz");
  }
  delay(1); // Düzgün çalışabilmesi için kısa bir bekleme veriyoruz.
}
```

Programdaki referansDegeri bizim eşiğimizi oluşturmaktadır. Bu değeri kendi sensörünüze göre ayarlamanız gerekmektedir. Programı çalıştırıp aldığınız değerleri inceleyiniz. Siyahtaki değer ile beyazdaki değeri toplayıp ikiye bölünüz. Bu yeni değer artık sizin referansDeger'inizi oluşturacaktır.

Sensörün çalışıp çalışmadığını düşünüyorsanız kontrol amaçlı Android tabanlı telefonunuzun kamerasını sensöre doğru tutunuz (Bazı android ve iphone'larda çalışmaz). Eğer sensörde ışık yandığını görüyorsanız besleme bağlantınız doğrudur. Sensörünüz hala çalışmıyorsa analog ayağıyla sensör bağlantılarını bir kez daha kontrol ediniz.

## Arduino ile LCD Kullanımı

Arduino ile sıklıkla kullanmasam bile LCD kullanımını öğrenmek yararlı olacaktır. Her zaman verilerimizi göstermek için bilgisayar bulamayabiliriz. Bu gibi durumlarda LCD'mizi kullanarak elde ettiğimiz durumları ekrana yazdırabiliriz. Kablo bağlantıları biraz karışık olabilir ve ilk denemenizde hata yapabilir ve LCD'yi düzgün bir şekilde kullanamayabilirsiniz. Bu yüzden devre bağlantılarınızı dikkatli yapın.




Devremizi dikkatli bir şekilde kurduysak artık programlamaya geçebiliriz. Bütün karakterler daha önceden bizim için tanımlanmış. Bu karakterlere ulaşmak için öncelikle kütüphanemizi programımıza dahil edeceğiz. Daha sonra başlangıç ayarlarımızı yapacağız ve LCD'mizi kullanmaya başlayacağız.

Haydi ekranımıza bir şeyler yazalım.

```
#include <LiquidCrystal.h> // kütüphanemizi ekledik
LiquidCrystal lcd(12, 11, 5, 4, 3, 2); // kablo bağlantılarımızı programa
tanıttık
void setup() {
  lcd.begin(16, 2); // LCDmizin satır ve sütun sayısını girdik
  lcd.print("Hasbi Sevinc"); // Ekrana bir şeyler yazdık
}
void loop() {
  lcd.setCursor(0, 1); // imlecimizi 2. Satıra indirdik
  lcd.print(millis()/1000); // programın başlangıcından beri geçen süre
}
```

## Dahili EEPROM'a Yazma ve Okuma

Bulduğumuz sonuçları ve değişkenlerimizdeki değerleri enerji kesilse dahi Arduino içerisinde kaydetmek isteyebiliriz. Bunu sağlayabilmemiz için Arduino'nun içerisinde dahili EEPROM Bulunmaktadır. EEPROM programımız ile ilgili değişken değerlerini tutabilen, elektriksel olarak yazılıp silinebilen küçük depolama birimidir. Arduino üzerindeki mikrokontrolcü türüne göre EEPROM kapasitesi değişmektedir. ATmega328'in 1024 byte, ATmega168 ve ATmega8'in 512 byte, ATmega1280 ve ATmega2560'ın ise 4 KB depolama alanı vardır.

Bu depolama alanları programımız için gerekli değerlerin depolanması için yeterli olmaktadır.

Hadi örnek bir program yazalım

```
#include <EEPROM.h> // EEPROM kullanımı için kütüphane eklenmesi
int yazdirilacakAdres , yazdirilacakDeger;
int okunacakAdres, okunanDeger;
void setup()
{
  Serial.begin(9600); // Bilgisayara veri göndermek için
}
void loop()
{
  int yazdirilacakAdres = 10; // 0-255 arasında adres giriyoruz
  yazdirilacakDeger = 50; // kaydedilecek değerimiz
  EEPROM.write(yazdirilacakAdres, yazdirilacakDeger); // 10 adresinde artık
50 değerimiz saklanıyor
  okunacakAdres  = yazdirilacakDeger; // 10 adresini okuyacağız
  okunanDeger = EEPROM.read(okunacakAdres ); // 10 adresini okuyoruz ve
oradaki dğeri okunan değer değişkenine yüklüyoruz
// Sonuçları ekrana yazdırıyoruz
  Serial.print(okunacakAdres);
  Serial.print("Adresindeki Deger= ");
  Serial.print(okunanDeger);
  Serial.println();
  delay(500); // biraz bekleyelim
}
```

## Arduino ile Servo Kullanımı

Yavaş yavaş arduinomuzla dış dünyayı kontrol etmeye başlıyoruz. Arduino ile servo motor sürmek diğer mikrokontrolcülere göre (özellikle PIC'e göre) daha kolaydır. PIC ile haftalarca süren servo motor kontrolü arduino ile dakikalar sürmektedir.

**Servo Motor:** 1 derece hassasiyetle çalışan 0-180 dereceler arasında hareket edebilen bir motor çeşididir. Dediğim gibi sadece 0 ile 180 derece arasında dönmektedir. Yani tam bir tur atamamaktadır. Genellikle robot kollarda kullanılmaktadır. Servonun çeşitleri genellikle taşıyabileceği yüke göre belirlenmektedir. Piyasada genellikle 1.4 kg*cm torkundaki servolar bulunmaktadır. Bu demek oluyor ki motor milinize bağlı 1 CM uzunluğunda bir çubuk ucuna bağlı yük 1.4 KG'dan fazla ise motorunuzun gücü mili döndürmeye yetmez. Eğer çubuğunuz 10 CM ise 140 gram kaldırabilirsiniz. Birçok kişi burada hata yapmaktadır. Kullanacağınız yüke göre servo seçmelisiniz. Piyasada aynı mantıkla çalışan daha kuvvetli servo motorlar vardır. Peki Servo motoru nasıl kontrol ediyoruz derseniz, teorik olarak şöyle özetleyebilirim. PWM adını verdiğimiz kare dalga sinyali vardır. Bu sinyalin HIGH ve LOW olduğu yerlere göre servomuzun konumu değişmektedir.
Servonun 3 bağlantı kablosu bulunmaktadır. Kırmızı kablomuzu +5 volta bağlayalım, kahverengi kablomuzu toprağa (- uca) ve turuncu kablomuzu da arduinomuzun PWN (~) yazan ayaklarına bağlayalım. Turuncu kablomuz data kablomuz olmaktadır ve konumumuz bu kablo üzerinden aktarılır. Unutulmamalıdır ki servolar vb mekanik elemanlar fazla akımla çalışmaktadır. Arduinomuzun 5 volt çıkışı servomuzu beslemeye yeterli olmadığı durumlar olabilir. Ayrıca servo yüzünden devrede gürültü oluşabilir. Bu yüzden devremizin + ve – uçlarına kapasite bağlamamız bu gürültüyü engelleyebilir.

```
#include <Servo.h>  // servo kütüphanesini programımıza dahil ettik
Servo servoNesnesi;  // servo kontrolü için bir nesne yarattık
void setup()
{
  servoNesnesi.attach(9);  // Servomuzun turuncu kablosunu 9 nolu pine
taktığımızı söylüyoruz
}

void loop()
{
  servoNesnesi.write(100);                  // Servomuzu 100 dereceye
döndürdük
  delay(1000);                            // biraz bekleyelim
servoNesnesi.write(20);                   // Servomuzu 20 dereceye döndürdük
  delay(1000);                            // biraz bekleyelim
}
```

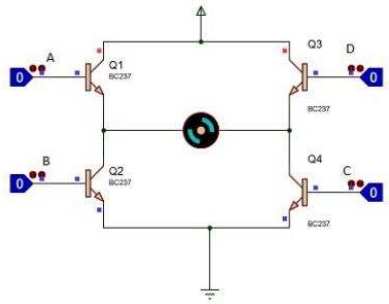
## Arduino ile DC Motor Sürme

Arduinonun pin çıkışlarını doğrudan motora bağlamamız motor için gerekli akımı sağlamamaktadır. Arduinonun pin çıkışları ancak bir ledi yakacak kadar akım vermektedir. Fakat biz bu akımı tetikleme işlevinde kullanıp DC motorları ileri veya geri yönde çalıştırabiliriz. Bunun için hazır devre kartları vardır. Bunlardan birini alıp kolaylıkla kullanabiliriz. Örnek hazır karta linkten ulaşabilirsiniz.

http://www.aliexpress.com/item/Special-promotions-5pcs-lot-L298N-motor-driver-board-module-L298-for-arduino-stepper-motor-smart-car/1827893830.html

Bu kartı aldığınızda birazdan anlatacağım şekilde kablo bağlantılarını yapabilir ve motorunuzu kolaylıkla kontrol edebilirsiniz.

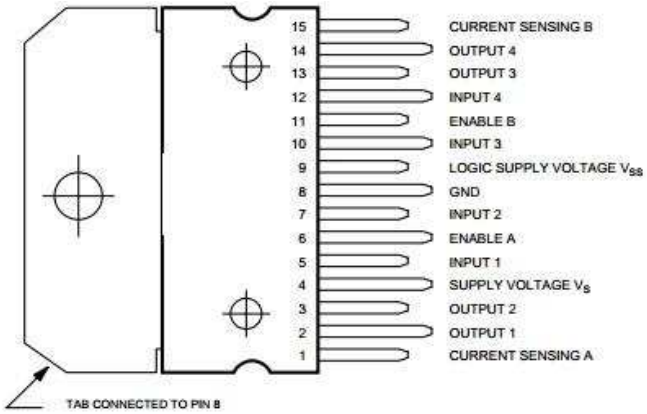
Fakat tüm devreyi kendiniz yapmak istiyorsanız bu iş biraz uğraştırıcı ve karmaşık olabilir. Fakat sistemin nasıl çalıştığını anlamak için en azından bir kere devrenin kurulması yararlı olabilir. Motor sürücü için akımı kuvvetlendirmek ve motoru kontrol etmek için L298 entegresini kullanacağız. Benzer entegreler de aynı görevi yapmaktadır. L298 entegresinin en önemli özellikleri, 2 ampere kadar dayanabilmesi ve iki adet H köprüsünün bulunmasıdır.

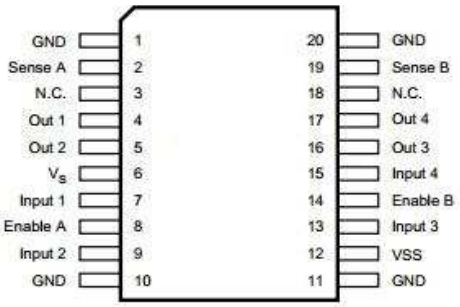



**H Köprüsü:**

H köprüsü motorumuzun ileri ve geri yönde sürülmesini sağlayan yapıdır. 4 adet tranzistörden oluşmaktadır.

Entegremizde toplam 15 adet ayak bulunmaktadır. Bu ayakların bazıları motorlarımıza bazıları Arduino'ya ve bazıları da beslemeye bağlanacaktır. Entegrenin pin yapılanması için resimden yararlanabilirsiniz.




Kısaca bağlayacağımız pinleri tanıtmak istiyorum;

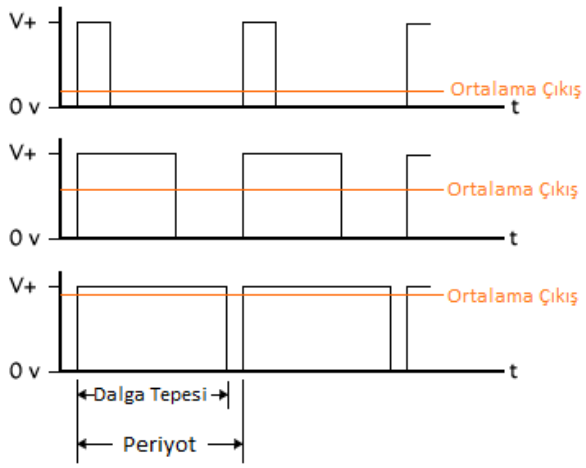
**INPUT 1 ve INPUT 2 (5. ve 7. Ayaklar):** Bu pinlerimizi Arduino'ya bağlayacağız. Input 1'e 5 volt verip input 2'ye 0 volt verdiğimizde motorumuz ileri doğru, tam tersini yaptığımızda da geri doğru gidecektir.

**INPUT 3 ve INPUT 4 (10. ve 12. Ayaklar):** Bu pinler de INPUT 1 ve INPUT 2 gibi çalışmaktadır.

**OUTPUT 1 ve OUTPUT 2 (2. Ve 3. Ayaklar):** Bu iki pin motorun iki ucuna bağlanması gerekmektedir.

**OUTPUT 3 ve OUTPUT 4 (13. Ve 14. Ayaklar):** Bu iki pin diğer motorun iki ucuna bağlanması gerekmektedir.

ENABLE A ve ENABLE B (6. ve 11. Ayaklar): Bu iki pin bizim hızımızı ayarlayacak pinlerdir. Bu yüzden bu pinleri Arduino'nun PWM ayaklarına bağlamamız gerekir. Verilen PWM sinyaline göre hızımız belli olacak ve INPUT ayaklarına verdiğimiz veya vermediğimiz 5 volta göre de motorun yönü belli olacaktır.




**PWM ile Hız Kontrolü:** PWM sinyali LOGIC-HIGH (5 Volt) ve LOGIC-LOW (0 Volt) sinyallerden oluşmaktadır. HIGH ve LOW sinyalin oranına göre belli olan ortalama çıkış sinyali sayesinde motorumuzun hızı 0-255 arasında değerler alabilmektedir.

**VSS (LOGIC SUPPLY VOLTAGE – 9. Ayak):** Adından da anlaşıldığı gibi bu pinin 5 volta bağlanması gerekmektedir. Devrenin kararsızlığını azaltmak için bu pin ile toprak arasına 100nF'lık kondansatör bağlanabilir.

**GND (8. Ayak) :** Bu pini toprağa bağlamamız gerekmektedir. Ayrıca entegrenin tepesindeki metal de toprak olmaktadır. Bunu belirtmemin nedeni yanlışlıkla kısa devre yapmamanız içindir.

**VS (4. Ayak):** Entegremizin motorlara gidecek enerjiyi sağladığı asıl besleme ayağıdır. Buraya verdiğimiz motor kontrolümüz doğrultusunda motorlara verilecektir. Motorumuzun özellikleri göze alınarak uygulamalarımızda buraya 12 Volt bağlayacağız.

Entegremizin pin yapılanmasını öğrendiğimize göre devremizi oluşturmaya başlayalım. Test için entegreyi breadboard üzerinde kullanabilirsiniz. Fakat diğer uygulamalarda kullanmak istediğinizde, kablo kalabalığı ve sürekli çıkan/temassızlık yapan kablolar yüzünden tam verim alamazsınız. Bu yüzden devreyi test ettikten sonra delikli pertinaks'a veya baskı devreye kurmak isteyebilirsiniz. Veya benim önerim hazır devre kartını almanız. Yurtdışından 2-3$'a alabilirsiniz.

Hazır kart aldığınızda da üzerinden demin bahsettiğim pinleri göreceksiniz.

Sıra geldi Arduino programını yazmaya. Aşağıdaki kodda oluşturuğum fonksiyonları diğer uygulamalarımızda da kullanacağız. Öncelikle INPUT ayakları için değişkenlerimizi tanımlıyoruz.

```
const int sagileri = 9;
const int saggeri = 8;
const int solileri = 12;
const int solgeri = 13;
const int solenable = 11;
const int sagenable = 10;
void ileri(int hizsag, int hizsol){
 analogWrite(sagenable, hizsag);
 digitalWrite(sagileri,HIGH);
 digitalWrite(saggeri,LOW);

analogWrite(solenable, hizsol);
 digitalWrite(solileri, HIGH);
 digitalWrite(solgeri,LOW);
}
void geri(int hizsag, int hizsol){ // ilk değişkenimiz sag motorun ikincisi
sol motorun hızını göstermektedir.
// motorlarımızın hızı 0-255 arasında olmalıdır.
// Fakat bazı motorların torkunun yetersizliğiniden 60-
255 arasında çalışmaktadır.
// Eğer motorunuzdan tiz bir ses çıkıyorsa hızını arttırmanız gerekmektedir
.
 analogWrite(sagenable, hizsag);
 digitalWrite(sagileri,LOW);
 digitalWrite(saggeri, HIGH);

analogWrite(solenable, hizsol);
 digitalWrite(solileri, LOW);
 digitalWrite(solgeri, HIGH);
}
void dur()
{
  digitalWrite(sagileri, HIGH);
  digitalWrite(saggeri, HIGH);
  digitalWrite(solileri, HIGH);
  digitalWrite(solgeri, HIGH);
  }
void setup(){
pinMode(sagileri,OUTPUT);
pinMode(saggeri,OUTPUT);
pinMode(solileri,OUTPUT);
pinMode(solgeri,OUTPUT);
pinMode(sagenable,OUTPUT);
pinMode(solenable,OUTPUT);
}
void loop(){
ileri(100,100); // ileri gidiyoruz
delay(1000);
dur(); // durduk
delay(1000);
ileri(150,100) // hafif sola doğru gidiyoruz
delay(1000);
```

```
dur(); // durduk
delay(1000);
ileri(100,150) // hafif sağa doğru gidiyoruz
delay(1000);
dur(); // durduk
delay(1000);
geri(100,100); // geri gidiyoruz
delay(1000);
dur(); // durduk
delay(1000);
}
```

Umarım kodlar yararlı olmuştur. Şimdi bir de yazılımda yazdığımız kodlarla yani Arduino ayaklarıyla motor sürücümüzün ayakları arasındaki bağlantıları bir tabloda gösterelim.

| **Arduino** | **Motor Sürücü** |
|---|---|
| 8 | INPUT 1 |
| 9 | INPUT 2 |
| 13 | INPUT 3 |
| 12 | INPUT 4 |
| 11 | ENABLE A |
| 10 | ENABLE B |

| **Motor** | **Motor Sürücü** |
|---|---|
| Motor1 + | OUTPUT 1 |
| Motor1 - | OUTPUT 2 |
| Motor2 + | OUTPUT 3 |
| Motor2 - | OUTPUT 4 |

(Motorun + veya – ucunun hangisi olduğu farketmez)

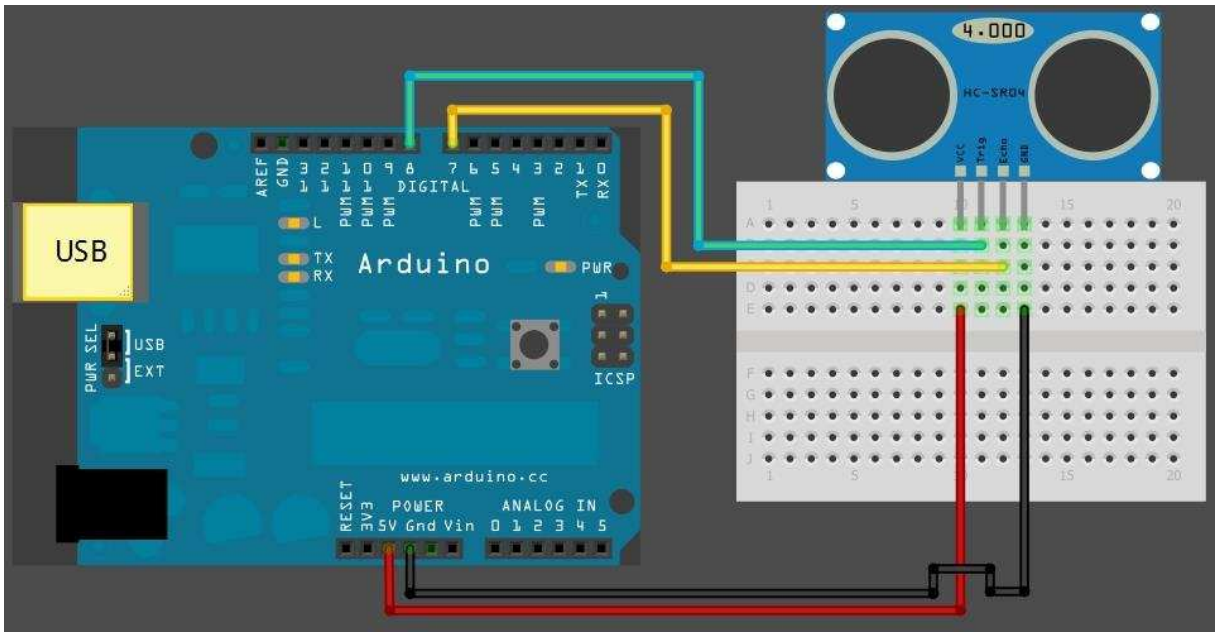
| **Besleme** | **Motor Sürücü** |
|---|---|
| +12 Volt | VCC |
| Toprak (- uç) | GND |
| +5 Volt | VS |

## Arduino ile Uzaklık Sensörü

Kitabımızda HC-SR04 Mesafe Sensörü ile uygulama yapacağız. Bu sensör piyasada bolca bulunan fakat ülkemizde pahalı olan bir sensördür. Çin'den tanesini 1-2$'a alabilirsiniz. Sensör 2-200 CM arasındaki uzaklıkları ölçmektedir. Fakat 200 CM'e doğru sensör kalitesi bozulmaktadır.

Sensör insan kulağının duyamayacağı bir frekansta ses yollar. Ses eğer bir yere çarpar ise geri yansız ve sensörümüze gelir. Sensör bu sesin gidip gelme süresini hesaplar ve böylece cismin uzaklığını bulur. Bu yüzden bu sensöre ultrasonik ses sensörü de denir.

Devre şeması için aşağıdaki resme bakabilirsiniz.




Resimde de görüldüğü gibi bağlantı şeması

**Sensör -> Arduino**

VCC -> 5 Volt

GND -> GND

Trig -> 8 (değişebilir)

Echo -> 7 (değişebilir)

Şimdi sensörden aldığımız uzaklığı bilgisayar ekranına yazdıran bir uygulama programlayalım.

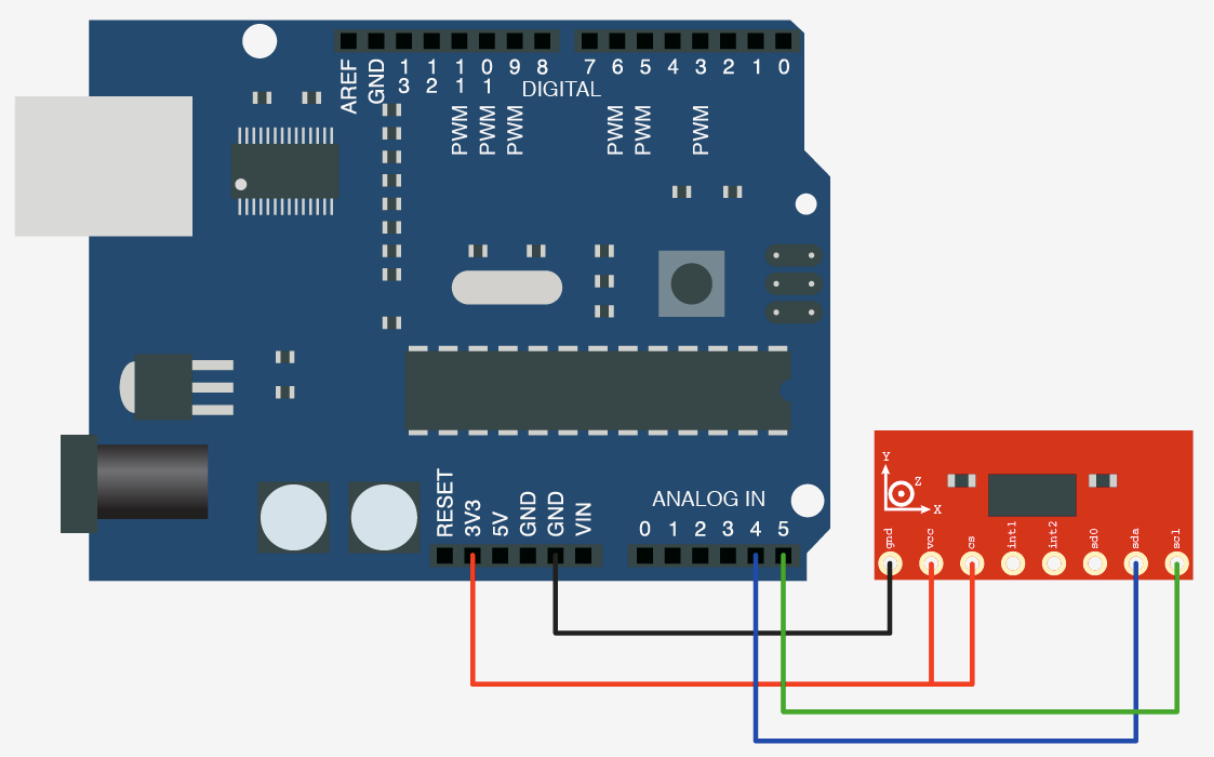
```
int trigPin = 6; // sensörün trig pinine bağlanacak arduino pini
int echoPin = 7;  // sensörün echo pinine bağlanacak arduino pini
long olcum;
long cm; // sensörümüzden okuduğumuz uzaklık
void setup(){
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin,INPUT);
 Serial.begin(9600);
}
void loop()
{
  digitalWrite(trigPin, LOW); // sensör ilk başta ses yollamasın
  delayMicroseconds(5);
  digitalWrite(trigPin, HIGH); // Burada ses dalgasını yolluyoruz
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);  // Tek bir ses dalgası yolladık
  olcum = pulseIn(echoPin, HIGH); // Eğer ses geri dönerse echo pinine geri
dönecektir.
// Burada geçen süreyi hesaplıyoruz.
 cm= olcum /29.1/2; // ölçüm değerini zamandan -> CM'ye çeviriyoruz
Serial.println(cm); // sonucu Serial Monitor'den görmek için bilgisayara
yolluyoruz
  delay(100);
}
```

Eğer sensörün önünde engel yoksa sensörde bir miktar yavaşlama olacaktır. Bundan kurtulmak için MAX bekleme değerini girmemiz lazımdır. Bu değeri arttırıp azaltarak kendi sensörünüze uygulayabilirsiniz. Değiştirmeniz gereken kod:

digitalWrite(trigPin, HIGH, **2895**); // 2895 sayısını değiştirebilirsiniz.

## Arduino ile İvme Ölçümü

İvme ölçümü için piyasada en çok kullanılan ADXL345'i kullanacağız. Bu sensör ile 3 eksende açısal ivme ölçülebilir. Sensör $I^2C$ ve SPI hattı üzerinden haberleşmektedir. Haydi bağlantılarımızı kuralım:



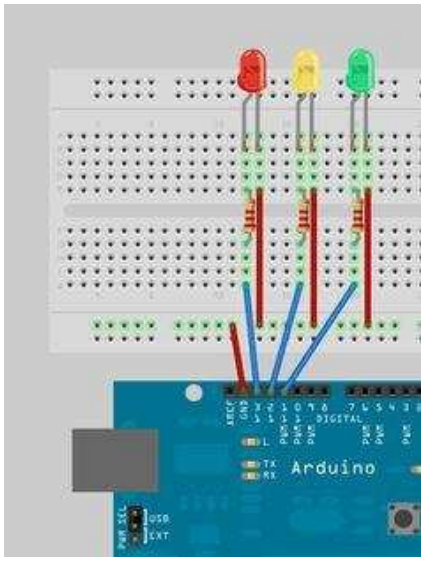

Devremize bağladığımıza göre programlamaya geçebiliriz. Sensörümüzü kullanmadan önce hazır fonksiyonlarının bulunduğu ADXL345 kütüphanesini indirelim. Linkten indirebilirsiniz. http://code.bildr.org/download/959.zip

Kütüphaneyi Arduino'nun program dosyaları arasındaki libraries klasörüne dosya halinde atalım. Kütüphane kurulumu tamamladığına göre örnek kodu inceleyelim.

```
#include <Wire.h>
#include <ADXL345.h>
ADXL345 adxl; //adxl adında nesne yaratıyoruz
void setup(){
 Serial.begin(9600);
 adxl.powerOn(); // sensörü çalıştırıyoruz
}
void loop(){
int x,y,z;
 adxl.readAccel(&x, &y, &z); //sensörden gelen değerleri okuyoruz
 Serial.print(x);
 Serial.print(y);
 Serial.println(z); // sensör değerlerini ekrana yazdırdık.
Delay(10);
// sensörler hem yer çekim ivmesini hemde aktif ivmeyi vermektedir
}
```

# ARDUİNO UYGULAMALARI

## Trafik Lambaları




İlk uygulamamız olarak basit bir proje ile başlayalım. Bu projede 3 adet(kırmızı,sarı,yeşil) LED kullanacağız. Daha önce de bahsettiğimiz gibi LED'in uzun ayağı arduinoya, kısa ayağı ise toprağa takılmalıdır. 3 LED'i sırası ile Arduino'nun 11, 12 ve 13. pinlerine takalım. Her LED ile Arduino bağlantısının arasına yaklaşık 220 ohm değerinde dirençler takalım.

Amaç: Kırmızı ışık 5 saniye boyunca yanacak. Daha sonra sarı ışık yarım saniye ve sonra da yeşil ışık 3 saniye boyunca yanar. Fakat daha sonra sarı ışık bir saniye yanar ve kırmızı ışığa geçilir.

```
int kirmizi = 13;
int sari = 12;
int yesil = 11;
void setup()
{
  pinMode(kirmizi, OUTPUT);
  pinMode(sari, OUTPUT);
  pinMode(yesil, OUTPUT);
}
void lambaDegistir(int lamba) // kirmizi = 1, sari = 2, yeşil = 3
{
  digitalWrite(kirmizi, LOW);
  digitalWrite(sari, LOW);
  digitalWrite(yesil, LOW);
  switch(lamba){
     case 1:
          digitalWrite(kirmizi, HIGH);
     break;
     case 2:
          digitalWrite(sari, HIGH);
     break;
     case 3:
          digitalWrite(yesil, HIGH);
     break;
  }
}
void loop()
{
  lambaDegistir(1);
  delay(5000);
  lambaDegistir(2);
  delay(500);
  lambaDegistir(3);
  delay(3000);
  lambaDegistir(2);
  delay(1000);
}
```

## Arduino ile Voltmetre Robot Yapımı

Bu projemizde 0 ile 5 volt arasındaki değerleri ölçebilen bir voltmetre tasarlayacağız. Amacımız ADC'yi daha iyi kavramaktır. Bağlantılarımızda sadece iki kablo kullanacağız.

İlk kablomuzu Arduino'nun ground ayağına bağlayalım. İkinci kablomuzu A0 ayağına bağlayalım. Yaptığımız voltmetreyi test etmek için test devresi kurabilir ve bunun çeşitli yerlerindeki gerilimleri ölçebilirsiniz.

Hemen yazılıma geçelim. ADC örneğimizden farklı olarak bu sefer sayısal değere karşı gelen gerilimi hesaplayıp bilgisayar ekranına yazdıracağız.

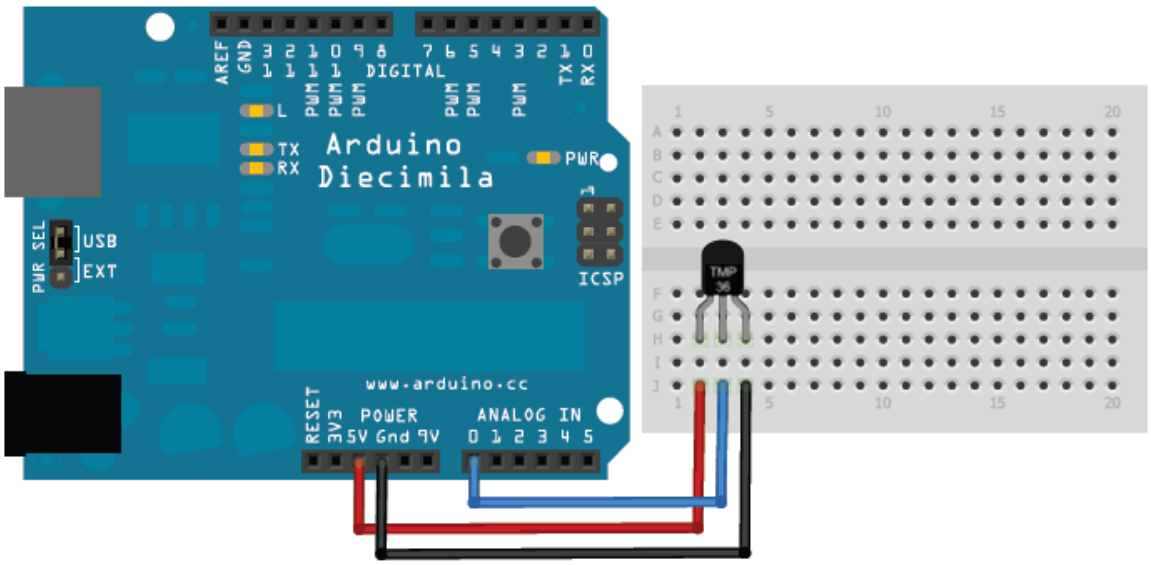
```
void setup() {
  Serial.begin(9600);
}
void loop() {
  int sensorDegeri = analogRead(A0); // A0'daki gerilimin sayısal değeri
  float gerilim = (5/1023)*sensorDegeri; // 5 volt max. 1023 ile
ölçülüyordu. Bu yüzden adım aralığını bulmak için 5/1023 işlemini yaptık.
Bu sonucuda ADC'nin değeri ile çarptık Böylece gerilimi bulmuş olduk.
  Serial.print(gerilim);
  Serial.println(" Volt");
  delay(100);
}
```

Ölçüm için gerilimi ölçmek istediğiniz yere A0'dan gelen kabloyu, devrenizin toprağına da GND'den gelen ucu bağlayınız.

**DİKKAT: Asla ve Asla 5 volt üzerindeki değerleri ölçmeye çalışmayın. Böyle bir hatayı engellemek için devrenize 5 Volt değerinde bir Zenner Diyotu ters bağlamanız yararlı olabilir.**

## LM35 ile Sıcaklık ölçümü

En temel uygulamalardan birisi olan LM35 sıcaklık sensörüyle ölçüm yapalım. LM35 sensörünün 3 pini vardır. Bunlar +5V, GND ve DATA ayaklarıdır. Data ayağından gelen değeri ADC ile dijitale çevirmemiz gerekir.




Bağlantıları şekildeki gibi yapalım. Sensörün göbekli yeri arkaya baktığında, yani yazısı bize baktığında ilk Pin 5 Volt, ortadaki pin Arduino'nun analog girişine ve 3. Pin ise toprağa (GND) bağlanmalıdır.

Haydi, yazılıma geçelim… Yapmamız gereken analog sinyali okumak ve matematiksel işlem ile sıcaklığa çevirmektir. Çevirdiğimiz değeri bilgisayara yollayalım.

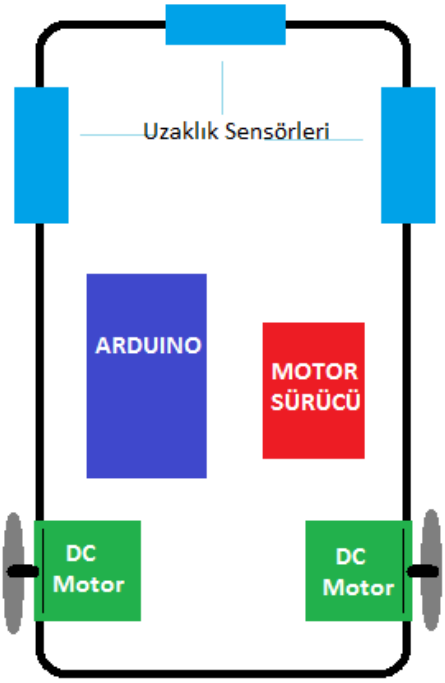
```
float sicaklik;
void setup()
{
  Serial.begin(9600);
}
void loop()
{
  sicaklik = analogRead(A0);
  sicaklik = sicaklik * 0.48828125;
  Serial.print("SICAKLIK = ");
  Serial.print(sicaklik);
  Serial.println(" C");
  delay(500);
}
```

## Arduino ile Çarpmayan Robot Yapımı

İlk robotumuzu yapmaya başlayalım. Robotumuzu yapmak için derste gördüğümüz uzaklık sensörümüzü ve DC motor sürücümüzü kullanacağız. Eğer bu konular hakkında bilgi sahibi değilseniz bu konuları tekrardan gözden geçirmeniz yararlı olabilir.

Haydi Başlayalım… Öncelikle malzeme listemizi çıkartalım:

- 1 adet şase: Tüm malzemeleri ve motorları sabitlemek için bir plaka
- 2 adet DC motor
- 2 adet tekerlek: DC motorun ucuna tam uymalı
- 1 adet motor sürücü: Kendi yaptığımız da olur
- 1 adet sarhoş tekerlek: Veya yüzeyle sürtünmeyi azaltacak bir çubuk ( ben genellikle LED'in kafasını kullanmaktayım)
- Pil : 11.1 Voltluk lipo piller uygun olacaktır. 9 Voltluk pil motorlar için yetersiz kaldığı oluyor Denemek lazım
- 3 adet HC-SR04 uzaklık sensörü
- Ve tabii ki bir Arduino




İlk başta robotun mekaniğini yapalım. Şasemizin arkasına iki motoru yerleştirelim. Robotun ön tarafına sarhoş tekerimizi yerleştirelim. Sarhoşumuz yoksa dengeyi sağlayacak şekilde robotun altına LED'ler koyalım. DC motorlarımıza tekerlerimizi takalım ve robotumuzun dengede durmasını sağlayalım. Şimdi Arduino ve motor sürücümüz için güzel bir yer ayarlayalım. Kablolamayı iyi yapmanız robotu rahat test etmenizi sağlayacaktır. Son olarak da sensörleri 0-90-180 derece şeklinde yerleştirelim.

Robotun yerleşimi resimdeki gibi olacaktır.

Elektronik bağlantıları daha önceki konularda bahsettiğimiz gibi dikkatlice yapalım.

Kablolamayı bitirdiğimize göre yazılıma geçelim ve robotumuzu test edelim.

```
const int sagileri = 9;
const int saggeri = 8;
const int solileri = 12;
const int solgeri = 13;
const int solenable = 11;
const int sagenable = 10;
/*
trigPin1 ve echoPin1 = soldaki uzaklık sensörümüz
trigPin2 ve echoPin2 = öndeki uzaklık sensörümüz
trigPin3 ve echoPin3 = sağdaki uzaklık sensörümüz
*/
int trigPin1 = 6;
int echoPin1 = 7;
int trigPin2 = 4;
int echoPin2 = 5;
int trigPin3 = 2;
int echoPin3 = 3;
float uzaklik(int trigPin, int echoPin){
  float olcum;
  float cm;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(5);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  olcum = pulseIn(echoPin, HIGH);
  cm= olcum /29.1/2;
  return cm;
}
void ileri(int hizsag, int hizsol){
 analogWrite(sagenable, hizsag);
 digitalWrite(sagileri,HIGH);
 digitalWrite(saggeri,LOW);

 analogWrite(solenable, hizsol);
 digitalWrite(solileri, HIGH);
 digitalWrite(solgeri,LOW);
}
void geri(int hizsag, int hizsol){
 analogWrite(sagenable, hizsag);
 digitalWrite(sagileri,LOW);
 digitalWrite(saggeri, HIGH);

 analogWrite(solenable, hizsol);
 digitalWrite(solileri, LOW);
 digitalWrite(solgeri, HIGH);
}
void dur()
{
  digitalWrite(sagileri, HIGH);
  digitalWrite(saggeri, HIGH);
  digitalWrite(solileri, HIGH);
  digitalWrite(solgeri, HIGH);
}
void setup(){
  pinMode(sagileri,OUTPUT);
  pinMode(saggeri,OUTPUT);
  pinMode(solileri,OUTPUT);
```

```
  pinMode(solgeri,OUTPUT);
  pinMode(sagenable,OUTPUT);
  pinMode(solenable,OUTPUT);

  pinMode(trigPin1, OUTPUT);
  pinMode(echoPin1,INPUT);
  pinMode(trigPin2, OUTPUT);
  pinMode(echoPin2,INPUT);
  pinMode(trigPin3, OUTPUT);
  pinMode(echoPin3,INPUT);
}
void loop(){
  while( uzaklik(trigPin2, echoPin2 ) < 10 ){ // önüne engel gelene kadar
düz git
    ileri(100,100);
  }
  dur(); // engel geldikten sonra dur
  delay(1000); // 1 saniye bekle
  if(  uzaklik(trigPin1, echoPin1) > 10 ){ // soluna bak
    ileri(150,0); // engel yoksa sola dön
    delay(500); // 90 derece dönene kadar geçecek süre. Bu süreyi kendinize
göre ayarlayın
    dur(); // dur
    delay(1000);
  }else if(uzaklik(trigPin3, echoPin3) > 10 ){ // sol dolu ise sağa bak
     ileri(0,150); // engel yoksa sağa dön
     delay(500); // 90 derece dönene kadar geçecek süre. Bu süreyi
kendinize göre ayarlayın
     dur(); // dur
     delay(1000);
  }else { // sağ ve solda engel varsa geri gidip dönelim
     geri(100,100);
     delay(1000);
     ileri(150,0);
     delay(500);
     dur();
     delay(1000);
  }
}
```

Yaptğınız robot ve kodların uygun olarak çalışması için kodlar üzerinde bazı değişiklikler yapmak gerekebilir.

Bu robot ile labirent çözen robot yapılabilir. Burada tek önemli olan robotun tam 90 derece dönmesini sağlamaktır.

## Arduino ile Basit Çizgi İzleyen Robot Yapımı

Bir önceki projemizdekine benzer bir robot tasarlayalım. Fakat bu sefer uzaklık sensörlerini sökelim ve robotun en önüne yere bakacak ve yerden 1-2 CM yukarıda olacak şekilde, siyah/beyaz sensörlerimizi koyalım. Bu robotumuzda 3 tane CNY70 kullanacağız. Bu yüzden daha önce anlattığımız şekilde bu sensörlerden 3 tane bağlayalım.

Öncelikle söylemeliyim ki robot yarışmalarında gördüğümüz kadar kaliteli bir çizgi izleyen yapmayacağız. O tarz bir robot yapmak için PID kullanmamız gerekir. Fakat biz buna gerek kalmadan daha basit bir robot yapacağız.

Haydi Başlayalım… Öncelikle malzeme listemizi çıkartalım:

- 1 adet şase: Tüm malzemeleri ve motorları sabitlemek için bir plaka
- 2 adet DC motor
- 2 adet tekerlek: DC motorun ucuna tam uymalı
- 1 adet motor sürücü: Kendi yaptığımız da olur
- 1 adet sarhoş tekerlek: Veya yüzeyle sürtünmeyi azaltacak bir çubuk ( ben genellikle LED'in kafasını kullanmaktayım)
- Pil : 11.1 Voltluk lipo piller uygun olacaktır. 9 Voltluk pil motorlar için yetersiz kaldığı oluyor Denemek lazım
- 3 adet CNY70 : Bağlantıları bir kartın üzerine yaparak kullanabilirsiniz
- Ve tabii ki bir Arduino

Programlamaya başlayalım. Önceki konularda öğrendiğimiz kodlar üzerinden projemizi yazacağız.

```
const int sagileri = 9;
const int saggeri = 8;
const int solileri = 12;
const int solgeri = 13;
const int solenable = 11;
const int sagenable = 10;
int sensorpin[3] = {A0,A1,A2}; // Sensörlerimizin bağlantıları
int dijitalize(int deger,int sensor){
  int esikdegeri;
  switch(sensor){
   // Eşik değerlerini kendi sensörünüze göre ayarlamayı unutmayın
   // Eşik değerleri için "Arduino ile Siyah Beyaz Kontrolü" konusuna
bakabilirsiniz
     case 0:
        esikdegeri = 900;
```

```
      break;
      case 1:
         esikdegeri = 820;
      break;
      case 2:
         esikdegeri = 820;
      break;
  }
  if (esikdegeri <deger)
    return 1;
  else
    return 0;
}
void ileri(int hizsag, int hizsol){
 analogWrite(sagenable, hizsag);
 digitalWrite(sagileri,HIGH);
 digitalWrite(saggeri,LOW);

 analogWrite(solenable, hizsol);
 digitalWrite(solileri, HIGH);
 digitalWrite(solgeri,LOW);
}
void setup(){
  pinMode(sagileri,OUTPUT);
  pinMode(saggeri,OUTPUT);
  pinMode(solileri,OUTPUT);
  pinMode(solgeri,OUTPUT);
  pinMode(sagenable,OUTPUT);
  pinMode(solenable,OUTPUT);
}
void loop(){
  if(dijitalize(analogRead(sensorpin[1]),1)){
    ileri(100,100);
  }

  if(dijitalize(analogRead(sensorpin[0]),0)){
    ileri(0,100);
    while(!dijitalize(analogRead(sensorpin[0]),0));
  }

  if(dijitalize(analogRead(sensorpin[2]),2)){
    ileri(100,0);
    while(!dijitalize(analogRead(sensorpin[2]),2));
  }
}
```

## Arduino ile Bluetooth Kontrollü Robot Yapımı

Öncelikle Arduino ve Bluetooth kontrolü yaptığımız konuyu anladığınızdan emin olunuz. Bu projede robotumuzu TERA TERM programını kullanarak bluetooth üzerinden yöneteceğiz. Bir önceki projelerde kurduğumuz robotumuzu kuralım. Sensörlerimizi çıkartalım ve bluetooth modülümüzü bağlayalım.

- 1 adet şase: Tüm malzemeleri ve motorları sabitlemek için bir plaka
- 2 adet DC motor
- 2 adet tekerlek: DC motorun ucuna tam uymalı
- 1 adet motor sürücü: Kendi yaptığımız da olur
- 1 adet sarhoş tekerlek: Veya yüzeyle sürtünmeyi azaltacak bir çubuk ( ben genellikle LED'in kafasını kullanmaktayım)
- Pil : 11.1 Voltluk lipo piller uygun olacaktır. 9 Voltluk pil motorlar için yetersiz kaldığı oluyor Denemek lazım
- Bluetooth modülü
- Ve tabii ki bir Arduino

Eğer her şeyi doğru yaptıysak Bluetooth modülündeki ışık yanıp sönecektir. Lafı fazla uzatmadan programlamaya geçmek istiyorum. Robotumuz harekete geçmek için bilgisayardan komut bekleyecektir. Bu komutu TERA TERM programı üzerinden klavyemizle vereceğiz. Robotumuz 'w' ile ileri, 'x' ile geri gidecek; 'a' ile sola, 'd' ile sağa dönecek ve 's' tuşu ile duracaktır.

Haydi başlayalım…

```
const int sagileri = 9;
const int saggeri = 8;
const int solileri = 12;
const int solgeri = 13;
const int solenable = 11;
const int sagenable = 10;
void ileri(int hizsag, int hizsol){
 analogWrite(sagenable, hizsag);
 digitalWrite(sagileri,HIGH);
 digitalWrite(saggeri,LOW);

 analogWrite(solenable, hizsol);
 digitalWrite(solileri, HIGH);
 digitalWrite(solgeri,LOW);
}
void geri(int hizsag, int hizsol){
```

```
 analogWrite(sagenable, hizsag);
 digitalWrite(sagileri,LOW);
 digitalWrite(saggeri, HIGH);

 analogWrite(solenable, hizsol);
 digitalWrite(solileri, LOW);
 digitalWrite(solgeri, HIGH);
}
void dur()
{
  digitalWrite(sagileri, HIGH);
  digitalWrite(saggeri, HIGH);
  digitalWrite(solileri, HIGH);
  digitalWrite(solgeri, HIGH);
}
void setup(){
  Serial.begin(9600);
  pinMode(sagileri,OUTPUT);
  pinMode(saggeri,OUTPUT);
  pinMode(solileri,OUTPUT);
  pinMode(solgeri,OUTPUT);
  pinMode(sagenable,OUTPUT);
  pinMode(solenable,OUTPUT);
}
void loop(){
  if (Serial.available() > 0) {   //Bluetooth'tan veri bekliyoruz
    char tus = (char)Serial.read();
    if( tus == 'w' )
      ileri(100,100);
    if( tus == 's' )
      dur();
    if( tus == 'a' )
      ileri(0,100);
    if( tus == 'd' )
      ileri(100,0);
    if( tus == 'x' )
      geri(100,100);
  }
}
```