ARDUİNO İLE ROBOTİK KODLAMA

İÇİNDEKİLER

BÖLÜM 1-GİRİŞ

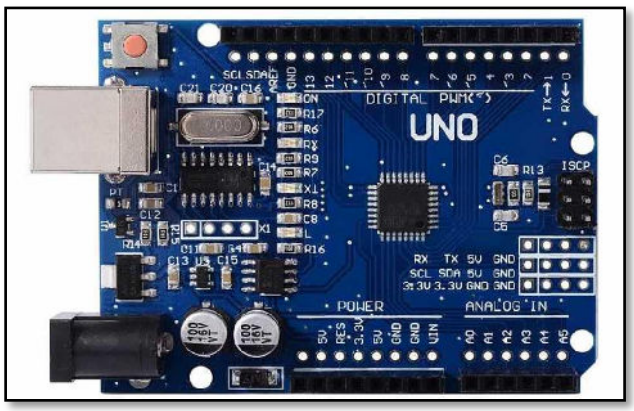
Bu kitap 2019-2020 eTwinning Arduino İle Robotik Kodlama projesinin ortak ürünü olarak hazırlanmıştır.

Projemizde, gelişen teknoloji ile birlikte günümüzde Arduino kullanımı büyük önem taşıdığı düşünülerek, öğrencilerimizin hayal güçlerini kullanarak teknolojiye uyum sağlayıp Arduino ile robotik kodlama alanında kendilerini geliştirmelerini amaçlanmış ve bu doğrultuda çalışmalar yapılmıştır. Ortaya çıkan bu ürün, tüm yapılan çalışmaların sonucu olarak ortaya çıkmıştır. Dokuz bölümden oluşan kitabın her bölümünü, bir okulumuz titizlikle hazırlamıştır. Arduino öğrenmede başkalarına da yardımcı olacağını düşündüğümüz bu çalışmanın oluşturulmasında emeği geçen herkese teşekkürler.

Arduino Nedir?

Arduino, projelerde kullanım kolaylığı sağlayan, açık kaynak kodlu oluş geliştirilebilen bir hazır devre kartıdır. Arduino kartlarında mikrodenetleyici ve devre bağlantıları yapmak için çeşitli elektrik bileşenleri bulunmaktadır. Arduino kolay kullanımı ve açık kaynak koduna sahip olduğu için tercih edilmektedir. Yazılan kodlar bir USB kablo aracılığıyla kolayca karta yüklenebilmektedir. Birçok projeyi kısa sürede gerçekleştirmek için Arduino kullanılmaktadır.

Arduino Özellikleri



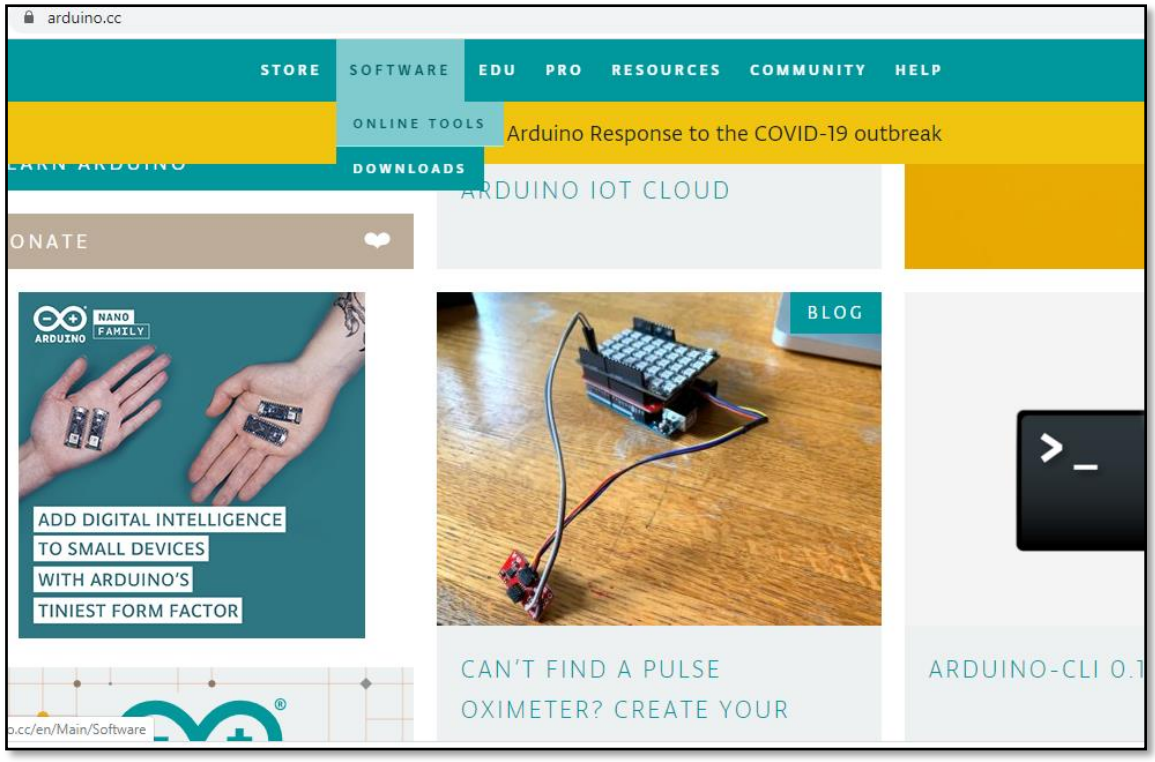
Arduino kartların çok çeşitli modelleri vardır ve bunlar kullanım amacına göre farklılık göstermektedir. Bu bölümde hem yaygın kullanılması hem temel anlamda hem de ileri seviyede kullanıldığı içi Arduino Uno kart özelliklerinden bahsedilmiştir.

Arduino Uno ile birçok şekilde haberleşme işlemi gerçekleştirilir. 14 adet dijital giriş / çıkışı mevcuttur. Bu çıkışlardan 6 tanesi PWM çıkışı olarak kullanılırken, 6 tanesi de analog giriş olarak kullanılmaktadır. RX ve TX pinleri ile seri haberleşme yapılmaktadır. Arduino IDE bulunan seri monitör ile Arduino ile bilgisayar arasında metin tabanlı bilgilerin gönderilip alınmasını sağlanmaktadır. Arduino ile bilgisayar arasında USB üzerinden bir haberleşme algılandığında, Arduino üzerindeki RX ve TX yazan LED'lerin yandığı görülmektedir. Arduino Uno'da bir tane seri port bulunurken çeşitli kütüphaneler kullanılarak bu sayı yazılımsal olarak arttırılabilmektedir.

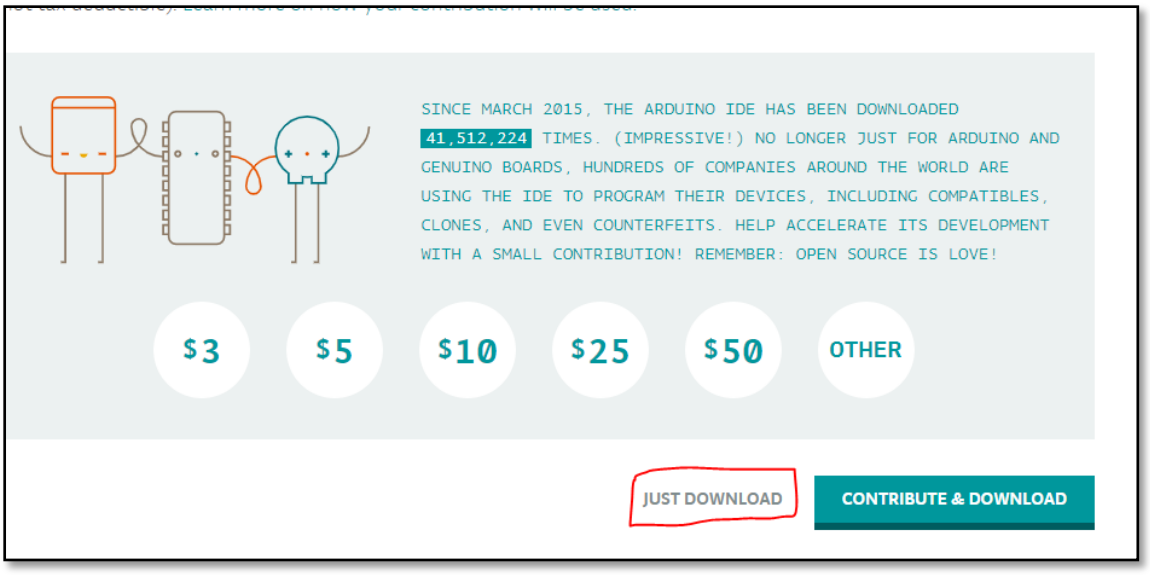
Arduino Yazılımı

Arduino yazılımı ile Arduino boardını programlamak için, Windows, Mac ve Linux işletim sistemlerinde kullanılabilmektedir. Yazılımın herhangi bir yükleyici (setup) programı yoktur. Arduino programı klasör içinde sıkıştırılmış durumdadır.

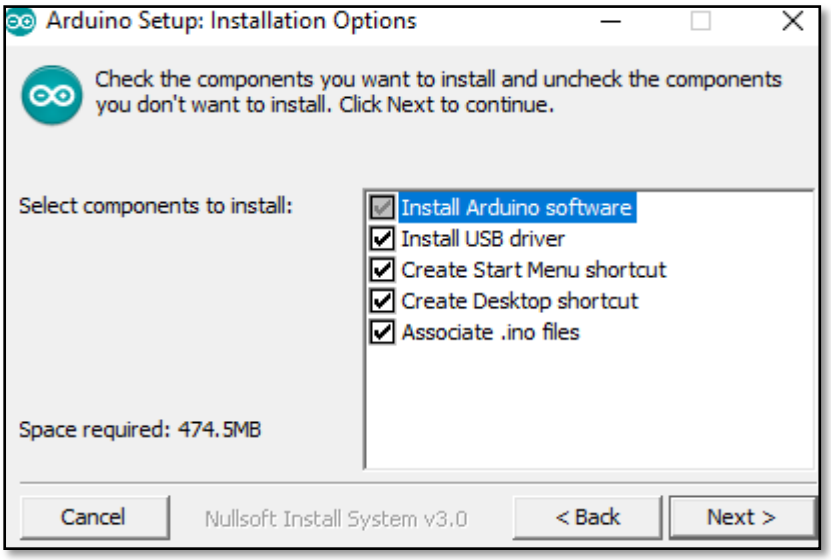
Arduino yazılımının yüklenmesi için öncelikle internet tarayıcısına **www.arduino.cc** adresinin yazılması gerekmektedir. Açılan siteden **Software**-**Download** sekmesi tıklanır.



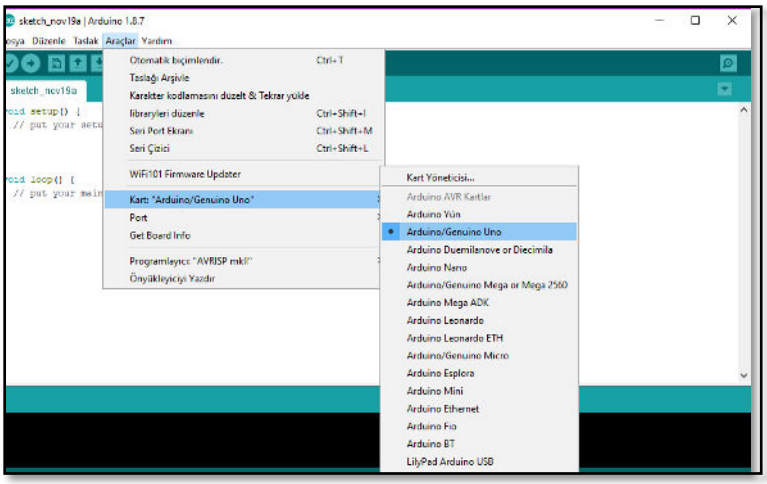
Açılan sayfadan uygun işletim sistemi seçilir ve **Just Download** seçeneği tıklanır.



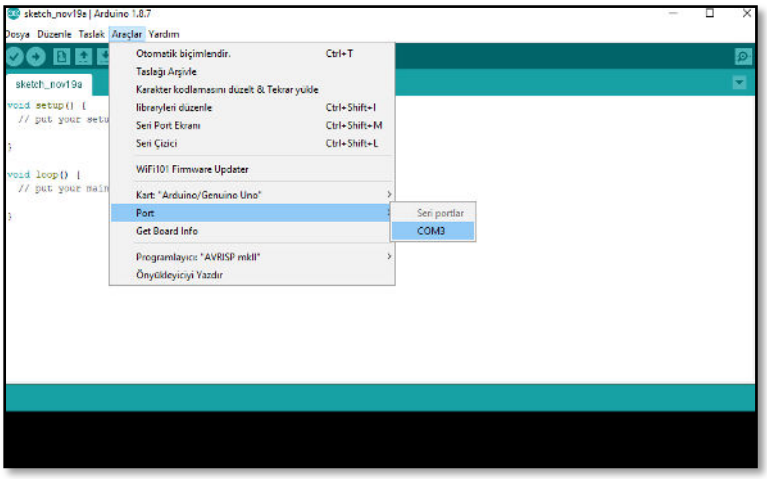
Bundan sonra yazılım kurulum dosyası inmeye başlamaktadır. İndirme işlemi bittikten sonra dosyayı açarak kurulum işlemi başlatılmalıdır. Kurulum sırasında çıkan "**Install USB driver**" seçeneği işaretlenmelidir.



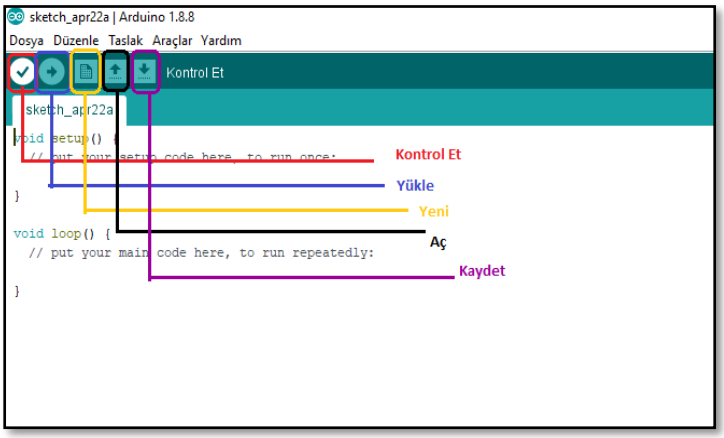
Kurulum işlemi bittikten sonra, Arduino kartın USB kablosu bilgisayara bağlanır. Bilgisayarınızda "**Yeni donanım bulundu**" penceresi açılacak ve sürücülerini otomatik olarak yükleyecektir. Bu işlemden sonra artık Arduino programı açılır. Program açıldıktan sonra ilk yapılması gereken şey, programın Arduino UNO kartı ile çalışacak şekilde ayarlanmasıdır. **Araçlar > Kart** menüsünden Arduino UNO seçeneğini tıklanmalıdır.

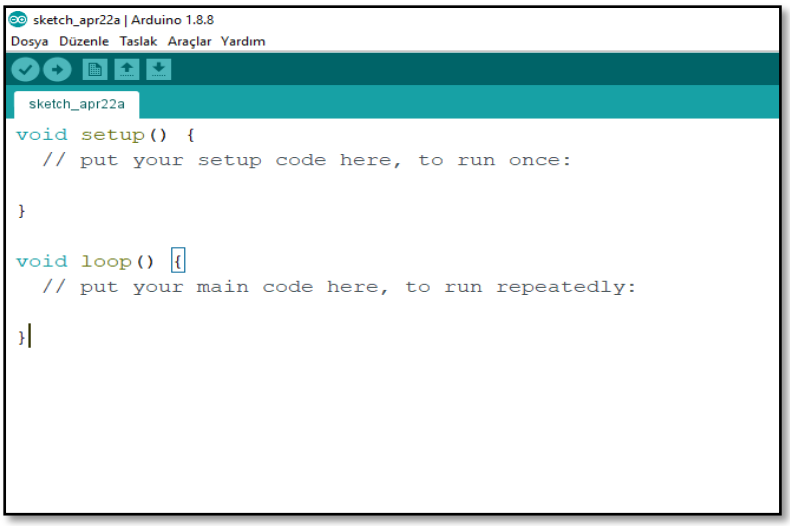


Daha sonra, yine Araçlar menüsünden Port alt menüsü altında Arduino kartın bağlı göründüğü port seçilmelidir. Bu port numarası, her bilgisayarda farklı olabilmektedir.



Arduino yazılımı artık programlama için hazırdır.



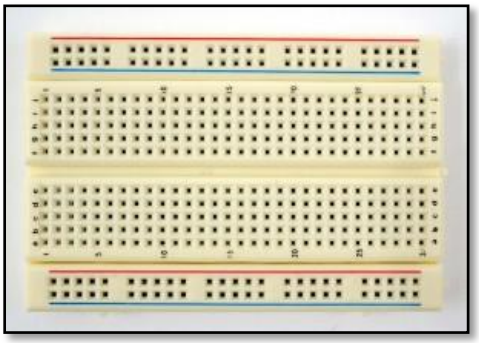


Programda **void setup()** kısmına yazılan fonksiyonlar, kart ilk enerji alıp çalıştığında sadece bir kere çalışmaktadır. Kullanılacak giriş/çıkış pinleri, seri port konfigürasyonu vb. ayarlar bu kısımda yapılmaktadır. **void loop()** kısmında ise, setup fonksiyonundaki komutlar çalıştıktan sonra kartın enerjisi kesilene kadar sürekli çalışacak olan foknsiyonlar yer almaktadır.

Program yazıldıktan sonra karta yüklenmek istenildiğinde, öncelikle "**Kontrol Et**" seçeneğine tıklanır. Program, yazılan kodun öncelikle bilgisayara bir klasöre kaydedilmesini ister ve daha sonra da yazılan kodu derleyerek herhangi bir hata varsa bu hatayı bildirir.

Arduino Bileşenleri

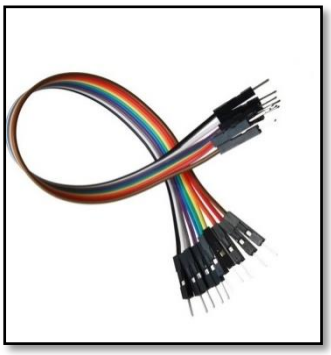
1.Breadboard: Devre elemanı da denilen breadboard içerisinde birbirine paralel hatlar bulunur ve üzerine birçok devre elamanı takılır.



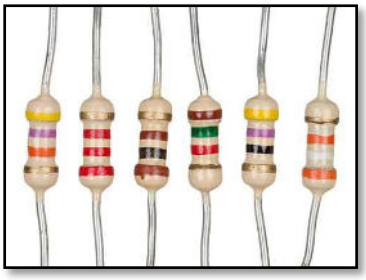
2. Jumper Kablo: Bu bağlantı kabloları, özellikle devre tahtası ile Arduino, EasyPIC gibi geliştirme kartlarının bir arada kullanıldığı devreler için oldukça uygundur. Uçlarında dişi ve erkek girişlerin olduğu üç çeşidi bulunmaktadır.Erkek-erkek,

- Erkek-dişi ve
- Dişi-dişi.

Bağlantı yapılacak girişlere göre, bu çeşitlerden uygun olanlar seçilir.



3. Direnç: Elektronik devrelerde akımı sınırlayarak belli bir değerde tutmaya yararlar. Bunun haricinde hassas devre elemanlarının üzerinden yüksek akım geçmesini önlerler, besleme gerilimini ve akımı bölmek için de kullanılırlar.



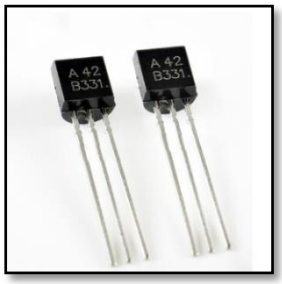
4. LED (Light Emitting Diode): Işık yayan diyot anlamına gelmektedir. Arduino için kullanılan LED'ler 5V ile çalışır. LED'lerimizin anot ve katot olmak üzere iki adet bacağı vardır. Anot bacağı + kutbuna, katot bacağı ise – kutbuna denk gelir. Katot yani – bacağı devremizde GND(toprak hattına) bağlanmalıdır.

5. Potansiyometre: Direnç değeri kullanıcı tarafından ayarlanabilen ya da değiştirilebilen bir devre elemanıdır. Potansiyometre arduino projelerinde led parlaklığı ayarlama, RGB led kontrolü, servo motor kontrolü, motor hız kontrolü, lcd ekran kontrolü ve diğer hassas ayarlama gerektiren tüm projelerde kullanılabilir.

6. RGB Ledler: Ortak anot ve ortak katot olmak üzere iki farklı yapıdadır. Ortak anot ledlerde R, G ve B nin (+) uçları, ortak katotlu ledler de ise (-) uçları birleştirilmiştir. Ortak anotlu RGB ledlerde ortak uca 5V, ortak katotlu ledlerde ortak uca toprak hattı (GND) bağlanmalıdır. Diğer 3 bacakla kırmızı, yeşil ve mavi renkleri kontrol edeceğimizden arduino üzerindeki PWM pinlerinden herhangi birine bağlanmalıdır.

7. Kondansatörler: Elektrik enerjisini depolayan iki uçlu devre elemanı olup elektriği depolamak için kullanılır.

8. Transistör: Transistör yan yana birleştirilmiş iki PN diyotundan oluşan, girişine uygulanan sinyali yükselterek akım ve gerilim kazancı sağlayan, gerektiğinde anahtarlama elemanı olarak kullanılan yarı iletken bir devre elemanıdır.



9. Röle: Düşük akımlar kullanarak yüksek akım çeken cihazları anahtarlama görevinde kullanılan devre elemanıdır. Rölenin bobinine enerji verildiğinde mıknatıslanan bobin bir armatürü hareket ettirerek kontakların birbirine temasını sağlar ve devrede iletim sağlanmış olur.

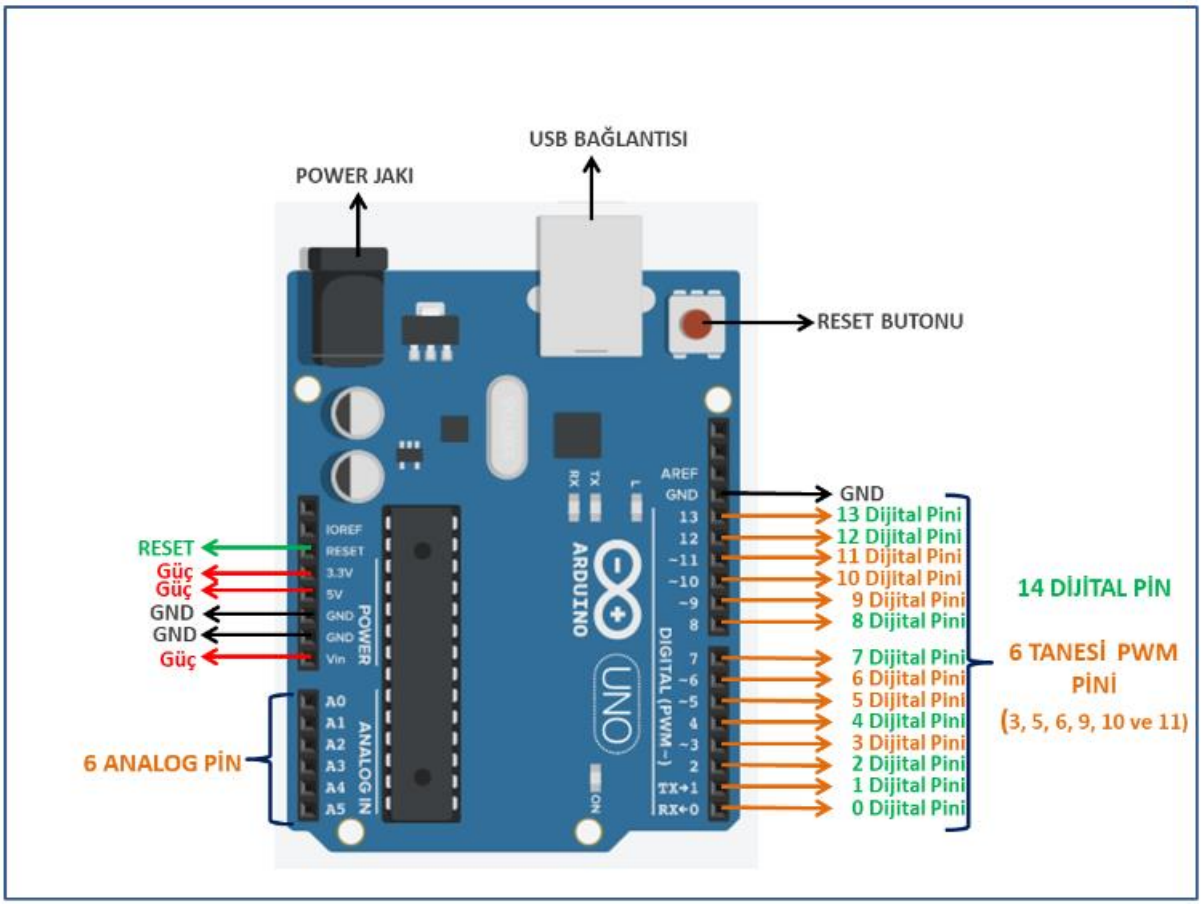
10. Buzzer: Bu devre elemanına ufak bir hoparlör de denebilir. Hoparlörler kadar yüksek ve detaylı ses üremeseler de, "bip" leme seslerini çıkartmada oldukça başarılıdırlar.

11. Servo Motorlar: Arduino projelerinin en çok kullanılan motor çeşitlerinden bir tanesidir. **Servo motorlar** genellikle 0 ile 180 derece arasında hareket edebilen motorlardır. Aynı zamanda 360 derece dönüş yapan çeşitleri de bulunmaktadır.

12. LCD Ekran: Sözcükleri veya sensörleri kullanarak alınan değerleri ekrana yazdırmak için arduino ile kullanılan bir ekran çeşitidir.

BÖLÜM 2- TEMEL GİRİŞ-ÇIKIŞ BİLGİLERİ

Arduino'nun en çok kullanılan kartı olan ARDUINO UNO ATMega328 Mikrodenetleyici kartı içerir. ARDUINO UNO kartı üzerinde 14 adet Dijital Giriş-Çıkış pini bulunmaktadır. (D0-D13) Dijital pinlerin 6 tanesi PWM pinidir.(3, 5, 6, 9, 10, ve 11) Ayrıca 6 adet Analog giriş-çıkış pini bulunmaktadır. (A0-A5) Bir adet USB bağlantısı, 1 adet power jakı ve reset butonu bulunmaktadır. Güç (5V, Vin, 3,3V) ve Topraklama (GND) pinleri de yer almaktadır.



GÜÇ VE TOPRAKLAMA PİNLERİ

Arduino UNO USB bağlantısı ile bilgisayardan veya pil-batarya-AC/DC adaptör gibi harici bir güç kaynağından beslenir. Arduino UNO kartı DC power jakından 7-12 V, USB bağlantısından ve 5V güç pininden 5V, Vin Pininden 7-12 V ile beslenir. 3,3 V güç pininden ise 3,3 volt voltaj beslemesi sağlanır. 3,3 V güç pini bazı sensör veya bileşenlerin beslenmesinde kullanımına ihtiyaç duyulmaktadır. Bileşenlerin bağlantıları yapılırken mutlaka internetten bileşenin adı ile kullanım kılavuzu (manueli) incelenerek kaç volt beslemeye, veya pin bağlantı noktalarına bakılmalıdır. Bu bağlantılarda hataların minimum düzeyde kalmasında yardımcı olmaktadır.

Arduino UNO üzerinde yer alan 3 adet GND pini ise topraklama bağlantısı için kullanılmaktadır..

GİRİŞ ÇIKIŞ PİNLERİ

1. **Dijital pinler :** Arduino Uno üzerinde 14 adet Dijital giriş-çıkış pini bulunmaktadır. Dijital pinlerden 0 ya da 1 sinyalleri alınır, gönderilir. 0 sinyali 0 Volt –LOW yani pin üzerinde herhangi bir sinyal yok anlamına gelir. 1 sinyali ise pin üzerinde 5V, HIGH yani pin üzerinde sinyal var anlamına gelir.
2. **PWM Pinleri:** Dijital giriş-çıkış pinlerinin 6 tanesi PWM pinidir. 3,5,6,9,10 ve 11 dijital pinleri ayrıca PWM pini olarak da kullanılmaktadır. Dijital pinler 0 veya 1 sinyal değerleri okurken, PWM pinleri 0-255 arasında değer okuyabilir.

3. **Analog Pinler:** Arduino UNO kartında 6 adet Analog giriş pini bulunmaktadır. Analog pinler 0-1023 arasında değer okuyabilir. Arduino kartında analog çıkış pini bulunmamaktadır. Analog çıkışa ihtiyaç duyulduğunda PWM pinleri analog çıkış pini olarak kullanılmaktadır. Analog giriş pininden okunan 0-1023 arasındaki değer, 0-255 arasındaki PWM pininin alabileceği değere dönüştürülerek analog çıkış pini olarak kullanılabilmektedir.

TEMEL FONKSİYONLAR

Arduino, C programlama dili tabanlı bir yazılımdır. Bütün programlama dillerinde olduğu gibi Arduino'da da fonksiyonlar işlemlerimizi kolaylaştırmaktadır. Arduino yazılımı açıldığında karşımıza çıkan **void setup()** ve **void loop()** adında iki temel fonksiyon vardır.

1. Değişken Tanımlama

Arduino kartına bağlanacak devre bileşenleri (Led, Buton vb) ve kart üzerinde bağlandıkları pin numaraları bir değişken olarak tanımlanabilir.

```
#Define Led 2;          #Define Buton 10;

int Led=2;              int Buton=10;
```

2. Void setup() Fonsiyonu

Setup fonksiyonu program çalıştırıldığında ilk okunan bloktur ve programın çalışması esnasında sadece bir defa okunur.Bu fonksiyon projede kullanılacak devre bileşenleri ve bu bileşenlerin Arduino kartı üzerinde bağlandıkları pinlerin giriş ya da çıkış pini olarak tanımlanmaları için kullanılır. Örneğin; Arduino kartına bir led bağlamak ve bu ledden dışarıya bir ışık üretmek istiyorsak Ledimizi çıkış olarak tanımlamamız gerekmektedir.

Setup fonksiyonu içerisinde devre bileşenlerini giriş ya da çıkış olarak tanımlamak için kullandığımız fonksiyon ise PinMode() fonksiyonudur.

PinMode(Pin,Mode) fonksiyonu

PinMode fonksiyonunda **Pin** yerine devre bileşeninin Arduino kartı üzerinde bağlandığı pin numarası veya bu pin için tanımladığımız değişken adı, **Mode** yerine ise Arduino kart bir değer okuyacaksa INPUT(giriş), Arduino kart dışarıya bir değer gönderecekse OUTPUT(çıkış) yazılmalıdır.

Buton bir giriş elemanıdır. Buton'a tıklanıldığında Arduino Uno kartına bir sinyal gönderilir ve buton için;

pinMode(Buton,INPUT); (Buton GİRİŞ bileşeni olarak tanımlanmıştır.)

Led ise bir çıkış elemanıdır. Led'e tıkladığımızda LED yanar, dışarıya ışık verir yani bir çıkış özelliği gösterir.

pinMode(Led,OUTPUT); (Led ÇIKIŞ bileşeni olarak tanımlanmıştır.)

3. Void Loop() Fonksiyonu

Loop fonksiyonu setup fonksiyonu okunduktan sonra, programın çalıştığı ana fonksiyondur. Arduino devre şema bağlantısı oluşturulduktan sonra, devrede yapılması istenilen kodlar burada yazılır. Loop fonksiyonu program çalıştığı sürece sürekli devam eden sonsuz bir döngü bloğudur. Yani program çalıştığı sürece devam eder.

Loop () fonksiyonu içerisinde kullanılan 4 adet temel giriş-çıkış komutu bulunmaktadır. Bunlar; DigitalRead(), DigitalWrite(), AnalogRead() ve AnalogWrite() komutlarıdır.

digitalWrite() Komutu

Digitalwrite() komutu; Arduino programı çalıştırıldığında setup() fonksiyon bloğunda ÇIKIŞ (OUTPUT) olarak tanımlanan bileşene sinyal göndermek için kullanılır. HIGH ve LOW olmak üzere iki adet çıkış durumu vardır.

HIGH → 5V'a yakın enerji var anlamına gelir.

LOW → 0V enerji yok anlamına gelir.

Kullanımı;

DigitalWrite(Led,HIGH);

DigitalWrite(Led,LOW);

Örneğin; Arduino kartı üzerine bağlanılan bir Led yakılmak istenildiğinde;

digitalWrite(Led,HIGH); (LED YAK)

Led'i söndürülmek istenildiğinde ise;

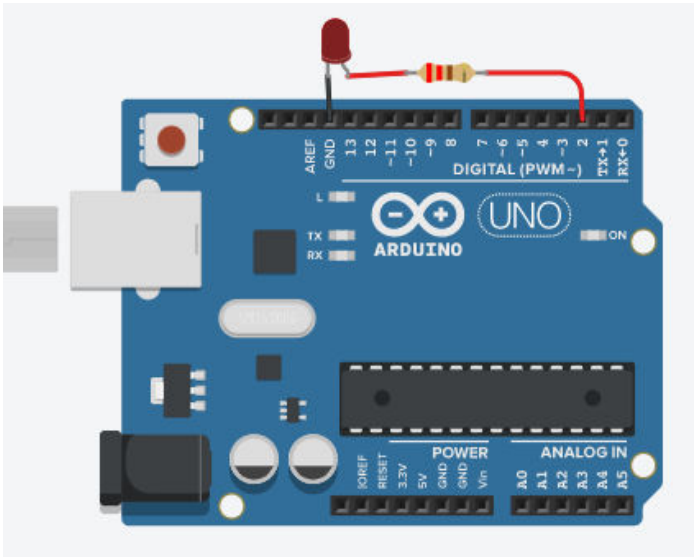
digitalWrite(Led,LOW); (LED SÖNDÜR) şeklinde komut yazılır

DigitalWrite() komutunun dahi iyi anlaşılması için Arduino Led Yak/Söndür uygulaması incelenebilir.

Malzemeler

1 adet Arduino UNO,1 adet Led, 1 adet 220 ohm direnç, Jumper Kablolar.

Devre Şeması



Kodlar

```
void setup()

{
 pinMode(2, OUTPUT);
}

void loop()

{
 digitalWrite(2, HIGH); // Led Yanar.
 delay(1000);  // 1saniye bekler.
 digitalWrite(2, LOW); // Led Söner
 delay(1000); // 1 saniye bekler.}
```

digitalRead() Komutu

Setup fonksiyon bloğunda INPUT(GİRİŞ) olarak tanımlanan bileşenden değer okumak ve sonucunda bir değer döndürmek için kullanılan komuttur. DigitalRead() komutu HIGH ya da LOW olarak değer döndürür.

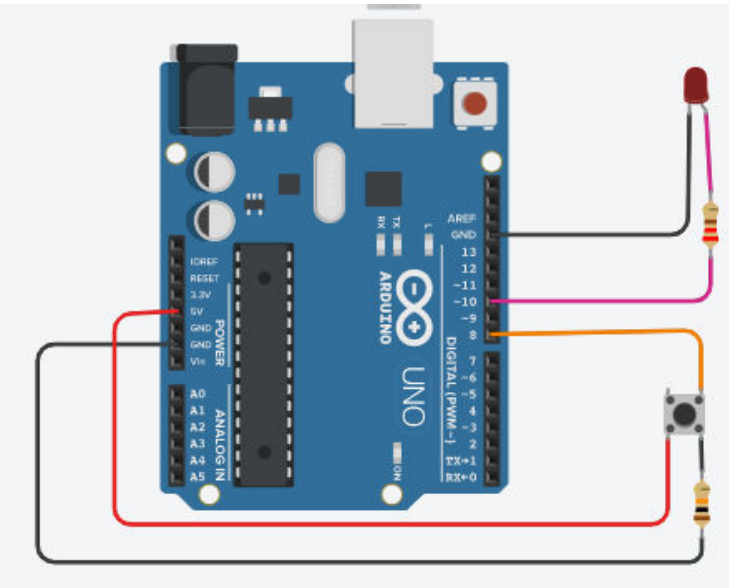
Örneğin; Buton bir giriş bileşeni olarak kullanılır. Buton Arduino kartına jumper kablolar aracılığıyla sinyal gönderir. Butona basıldığı sürece HIGH, basılmadığı sürece ise LOW sinyali gönderir.

DigitalRead() komutunun dahi iyi anlaşılması için Arduino Led Yak/Söndür uygulaması incelenebilir.

Malzemeler

1 adet Arduino UNO,1 adet Led, 1 adet 220 ohm direnç,1 adet buton, 1 adet 10 kiloohm direç, jumper kablolar.

Devre Şeması



Kodlar

```
#define Buton 8                    // Butonun 8.pine bağlandığı burada tanımlanır.
#define Led 10                     // Ledin 10. Pine bağlandığı burada tanımlanır.
void setup()
{
 pinMode(Buton, INPUT);     // Buton giriş olarak tanımlanmıştır.
 pinMode(Led, OUTPUT);     // Led çıkış olarak tanımlanmıştır.
}

void loop()
{
 if (digitalRead(Buton) == 1)   // Eğer butondan dönen değer 1 ise (Butona basıldıysa)
 digitalWrite(Led,HIGH);       // Led Yanar
 else                          // Butona basılmadıysa
 digitalWrite(Led,LOW);       // Led Söner.
}
```

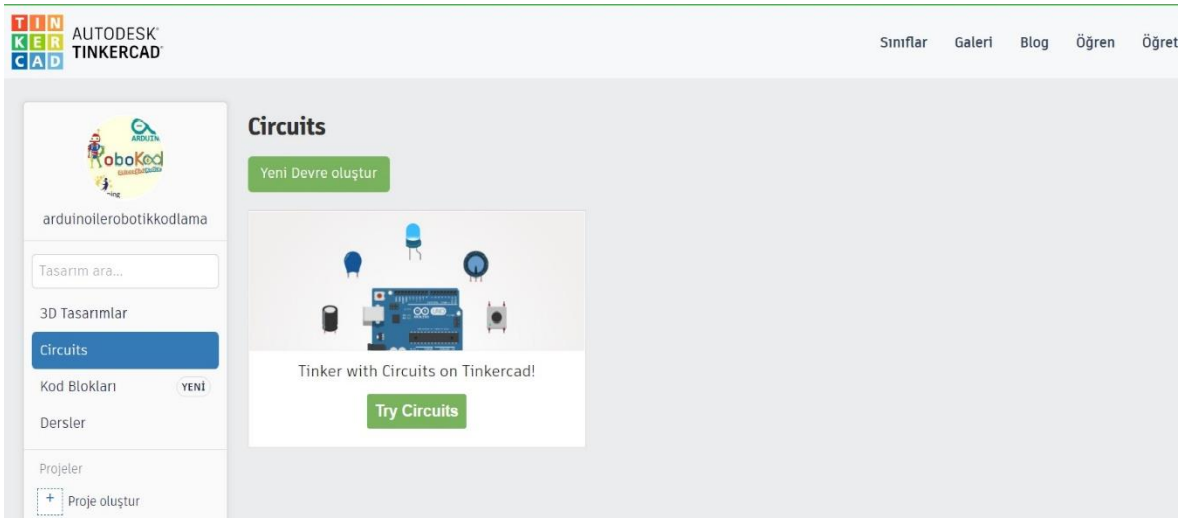
BÖLÜM 3- ARDUİNONUN GÖRSEL OLARAK KODLANMASI

TİNKERCAD İLE ARDUİNO

İlkokullardan başlayarak üniversitelere kadar öğrenciler ve robotik kodlamayla ilgilenen kişiler Arduino kullanarak kendi projelerini canlandırmaya çalışıyorlar. Önce bir ARDUINO kit almak gerekiyor. Sonra IDE denilen ortamı kurup, sürücü ve kütüphaneleri ayarlayıp, yazılımı öğrenmek lazım. Burada da hangi yazılım dilini öğreneyim diye düşünmeye başlıyorlar. Elektronik modüllerin ve bilgisayar bileşimlerinin sentezi olarak hayal ettiğiniz projenizi gerçekleştirmek bazen sizi epey uğraştırıyor ve kaçınılmaz yanlışlar yapmanıza sebep oluyor. AUTOCAD firmasının sunduğu TINKERCAD sitesi bu işe girişmek isteyenler için bir seçenek sunuyor. Web tarayıcısında çalışan bir simülatörle ARDUINO sanal olarak hayata geliyor. Yeni başlayanlar için yeteri kadar elektronik komponentler, modüller, ve hazır Arduino devreleri sunarak bu işe çabucak başlamanızı sağlıyor. Hazır olan veya kendi geliştirdiğiniz proje devreleri kullanarak, veya kitaplardan öğrendiğiniz uygulamaları burada kurarak sanal bir ortamda hiç bir para sarf etmeden ve elektronik problemlerle uğraşmadan öğreniminize başlayabiliyorsunuz.

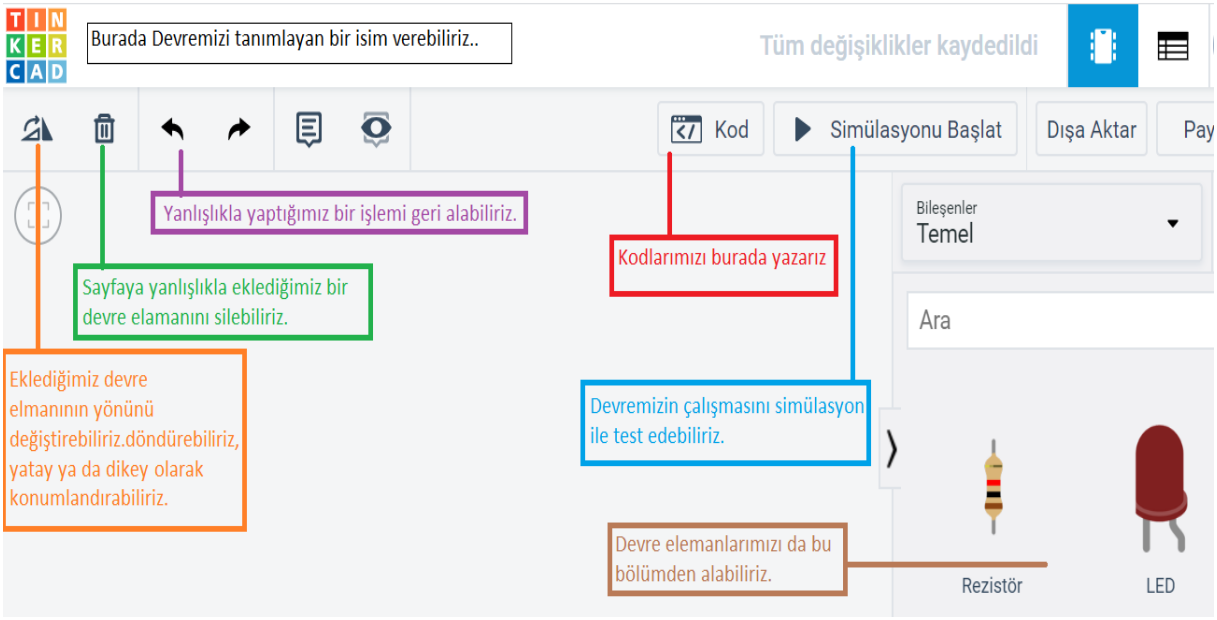
Gerçekleştirdiğiniz devrelerin görsel kodlamaları otomatik olarak ARDUINO C++ koduna çevriliyor ve aynı zamanda gereken kodlamayı da öğrenmeye başlıyorsunuz. Eğer kendi kodunuzu yazacak seviyeye gelmişseniz, bunları da ortama katmak gayet kolay. İsterseniz değişik kaynaklardan bulduğunuz kodları da bir noktaya kadar burada deneyebiliyorsunuz.

İlk olarak TINKERCAD sitesinde bir hesap açmanız gerekiyor: www.tinkercad.com



EDİTÖR

TINKERCAD editörü gayet basit fakat işlevsel bir çalışma ortamıdır. Sadece bir kaç ikon ve seçenekle projenizin detaylarını gerçekleştirmenizi sağlar.

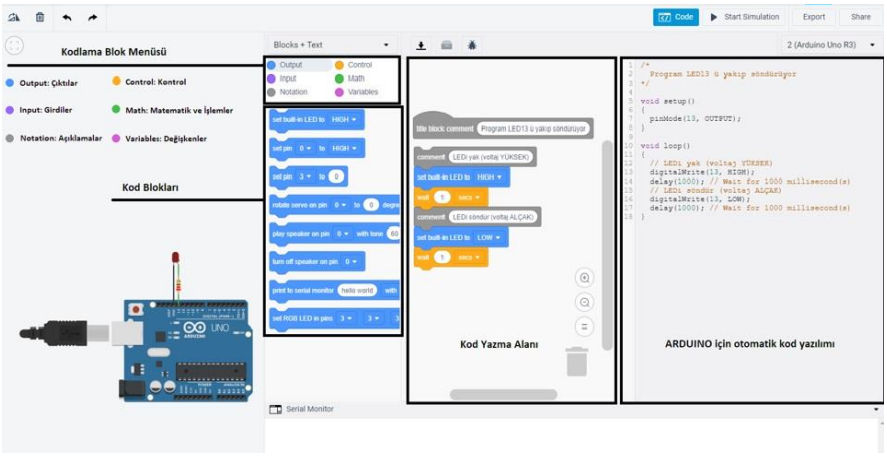


KOD– Arduino devresinin kodlandığı çalışma paneli. Burada hem **GÖRSEL** hem de **METİN** olarak hazırlanmış Arduino kodlarını göreceksiniz. Görsel kodlama çok popüler olan ve yeni başlayanlar için gayet kolay bir yazılım tekniğidir. SCRATCH veya SNAP gibi ortamlarla evvelden çalışmışsanız buna hemen alışırsınız. Değilse bunu kısa zamanda öğrenip uygulamaya başlayabilirsiniz. Metin kodlama ARDUINO IDE ortamında kullanılan aynı dil ve tekniği uygular. Eğer ARDUINO IDE'yi kurmuşsanız, burada gördüğünüz C++ kodunu kopyalayıp IDE editörüne yapıştırabilirsiniz. Bilginiz ilerledikçe bu ayarı yalnız 'metin' moduna koyarak tamamen sizin yazdığınız kodları çalıştırabilirsiniz.

SİMÜLASYONU BAŞLAT / DURDUR – ARDUINO devrenizi tamamladıktan sonra bu butona basarsanız çalışmasını izleyebilirsiniz. LED'ler yanıp söner, motorlar döner, elektronik komponentler gözle görülmese bile vazifelerini yaparlar. Ölçme modülleri kullanarak değişik portların ve komponentlerin voltaj, akım, ve resistans değerlerini ölçebilirsiniz. Aynı hakiki ARDUINO IDE ortamında olduğu gibi SERI monitörü kullanarak değer girdi çıktısı yapabilirsiniz.

KODLAMA PANELİ

Tinkercad'in araç çubuğundaki CODE(Kod) butonuna bastığınızda aşağıdaki kodlama panelini göreceksiniz.



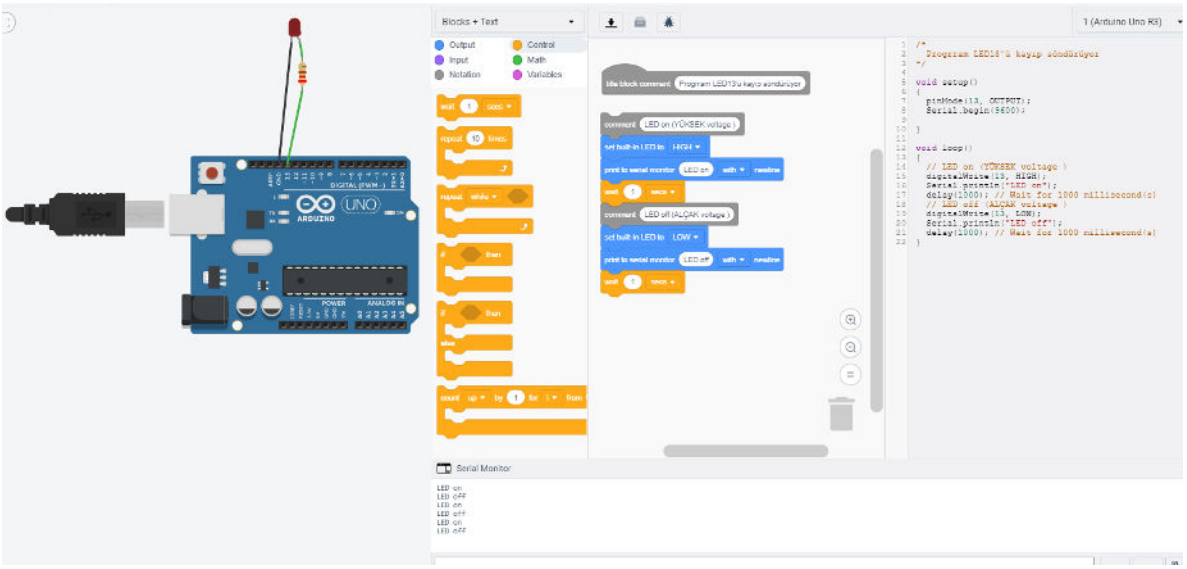
Panelin sol üstünde Kodlama Blok Menüsü var. Burada değişik kategorilerdeki kodlama bloklarını gösteren seçenekler var.

Herhangi birine tıkladığınızda hemen altındaki kod blokları seçtiğiniz gruba göre değişecektir. Bütün gruplardaki toplam blok sayısı 32 tanedir. Yani sadece 32 komut öğrenerek değişik ARDUINO projelerinizi programlamanız mümkündür.

Kod blokları değişik şekillerde olup bazı gruplar birbirine uyacak şekilde dizayn edilmişlerdir. Örneğin: ÇIKTILAR, AÇIKLAMALAR, ve KONTROL blokları birbirine uyar. GİRDİLER, MATEMATİK, ve DEĞİŞKEN blokları ise diğer bloklarla beraber kullanıldığından şekilleri ovaldır. Bunlar genelde öteki kod bloklarının içine yerleştirilerek kullanılırlar.

Kodlama bloklarının sağ tarafında Kod Yazma Alanında projenizi çalıştıran kodları gerçekleştireceksiniz. Bu GÖRSEL ortamda kod yazmak 'sürükle / bırak' yöntemiyle yapılıyor. Kullanmak istediğiniz kod bloklarını tıklayarak seçip bu alana sürüklüyorsunuz. Bloklar şekillerine göre ya birbirlerine uyarak peş peşe ekleniyorlar, ya da birbirlerine girdi değer olarak kullanılıyorlar.

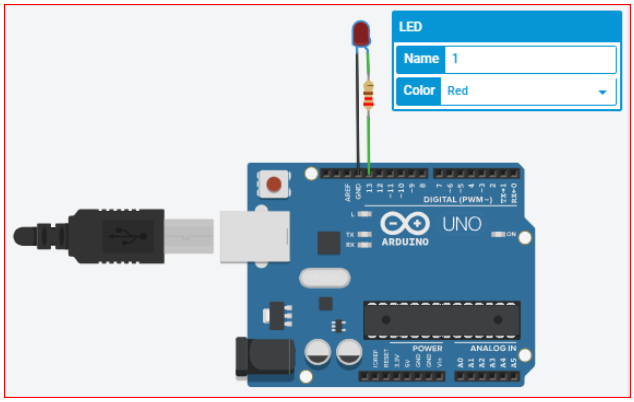
Aşağıdaki resimde ARDUINO'nun klasik başlangıç projesi olan BLINK (LED yak söndür) devresinin kod panel görselini yer almaktadır.



Ekranın en solunda ARDUINO devresinin modeli görünüyor. Kod panelinin sol kenarında GÖRSEL kodlama menüsü ve ikonları var. Kod panelinin ortasında görsel kodlamanın yazılımı görünüyor. Birbirlerine takılan işlev ikonları altalta dizilerek LED'i bir saniye arayla yakıp söndüren kodu oluşturuyor. Kod panelinin en sağında ise aynı kodun METİN versiyonu, yani Arduino C++ yazılımı görünüyor.

ARDUINO devresine yaptığınız her değişiklik için, örneğin bir LED veya düğme eklediniz, görsel ikonları kullanarak bunu çalıştıracak kodu yazıp deneyebilirsiniz. Arduino C++ kodu otomatikman yazılacaktır.

ÖRNEK UYGULAMA1:



a. Ortada ARDUINO UNO var.

b. Sol tarafta USB kablo bağlantısı var.

c. Üst tarafta bir kırmızı LED var.

d. LED in sağ altında 220 ohm'luk bir resistans var.

e. LED eksi ayağı (katod) Arduino toprak (GND) pinine siyah kablo ile bağlı.

f. LED artı ayağı (anod) resistansa yeşil kablo ile bağlı.

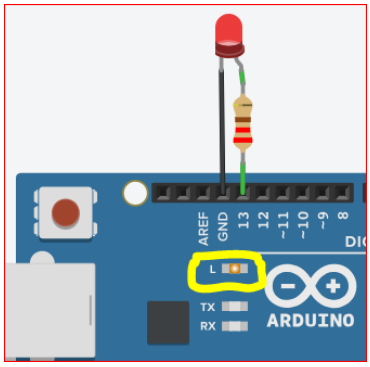
g. Resistansın alt ucu Arduino D13 pinine yeşil kablo le bağlı.

Herhangi bir komponenti tıkladığınızda ona ait bilgileri gösteren ufak bir pencere açılır. bu penceredeki alanları ayarlayarak komponentin adını ve uygun olan değerleri değiştirebilirsiniz. Örneğin LED'in penceresinden rengini değiştirebilirsiniz. Resistansın penceresinden ohm değerini ayarlayabilirsiniz.Örnek uygulamamızı çalıştırdığımız zaman;

a. USB kablosu ARDUINO ya girer ve ufak bir 'GÜÇ' işareti kablo üzeride gözükür. Arduino çalışmaya başlamış demektir.

b. Kırmızı LED komponent yanıp sönmeye başlar.

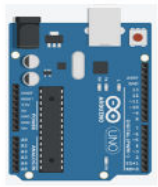
c Aynı zamanda Arduino ana kardının üzerinde bulunan solid-state (katıhal) LED de yanıp söner.



d. Son olarak, eğer ARDUINO'nın RESET düğmesine basarsanız, simülasyon / program baştan başlar.

ÖRNEK UYGULAMA2:

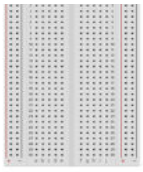
Üç ledi sırayla yakıp söndürme uygulaması için gereken malzemeler aşağıdadır;



1 ADET ARDUINO R3

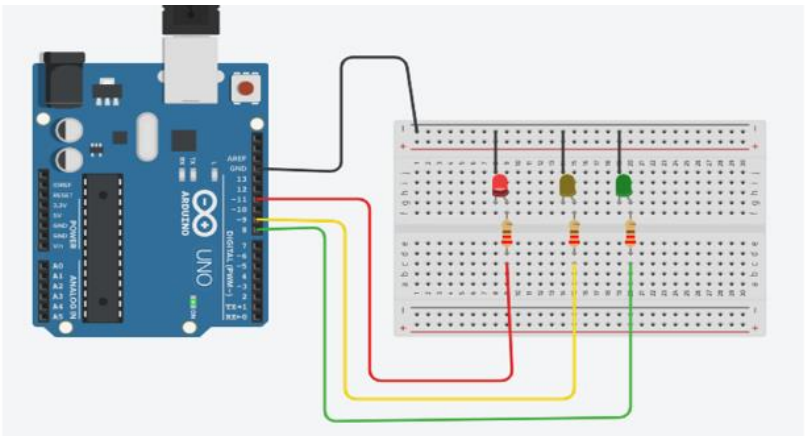
1 ADET KIRMIZI LED
1 ADET SARI LED
1 ADET YEŞİL LED

1 ADET
220 OHM DİRENÇ



1 ADET BOARD

Tinkercad uygulamasında oluşturduğumuz devre;



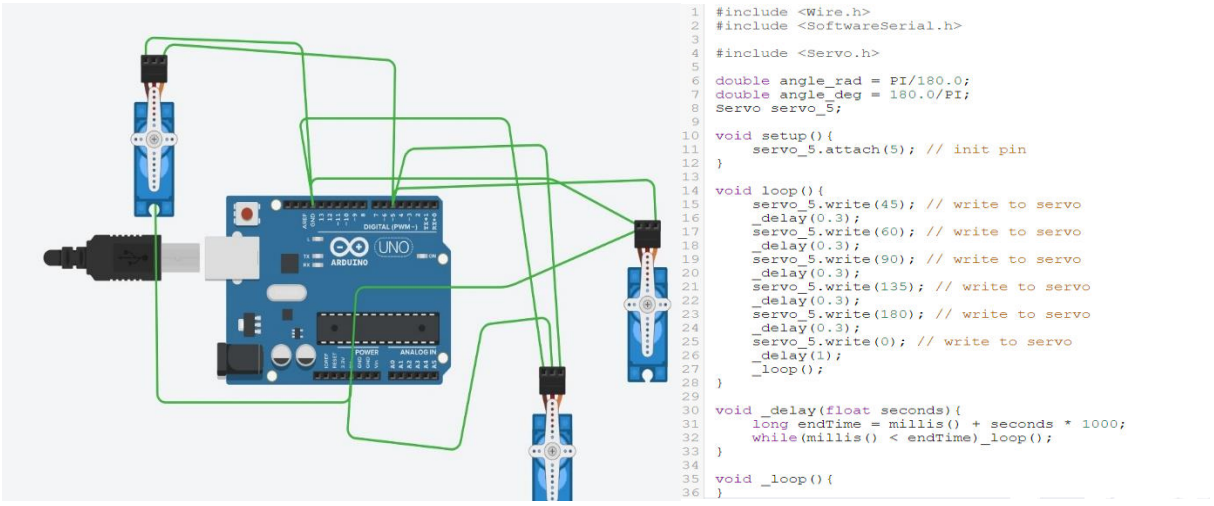
Devremizin kodları;

```
int ledkirmizi=11;
int ledsari=9;
int ledyesil=8;

void setup()
{
 pinMode(ledkirmizi,OUTPUT);
pinMode(ledsari,OUTPUT);
pinMode(ledyesil,OUTPUT);
}
```

```
void loop()
{
 digitalWrite(ledkirmizi,HIGH);
 digitalWrite(ledsari,LOW);
 digitalWrite(ledyesil,LOW);
 delay(2000);
 digitalWrite(ledkirmizi,LOW);
 digitalWrite(ledsari,HIGH);
 digitalWrite(ledyesil,LOW);
 delay(2000);
 digitalWrite(ledkirmizi,LOW);
 digitalWrite(ledsari,LOW);
 digitalWrite(ledyesil,HIGH);
 delay(2000);
}
```

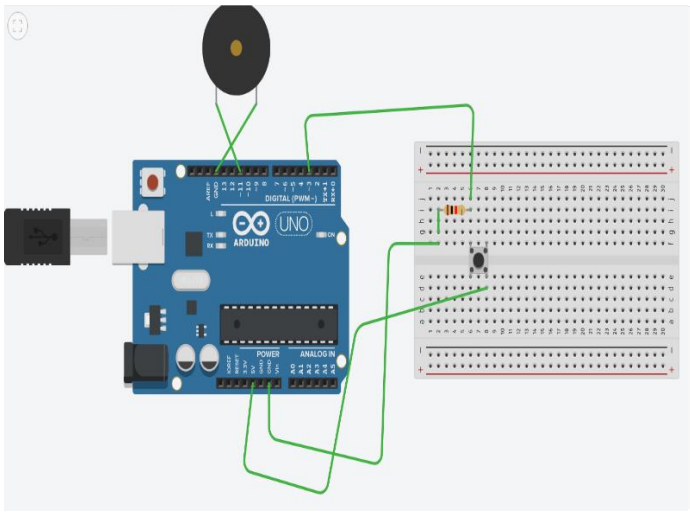
ÖRNEK TİNKERCAD UYGULAMALARI:

1.Servemotor uygulaması



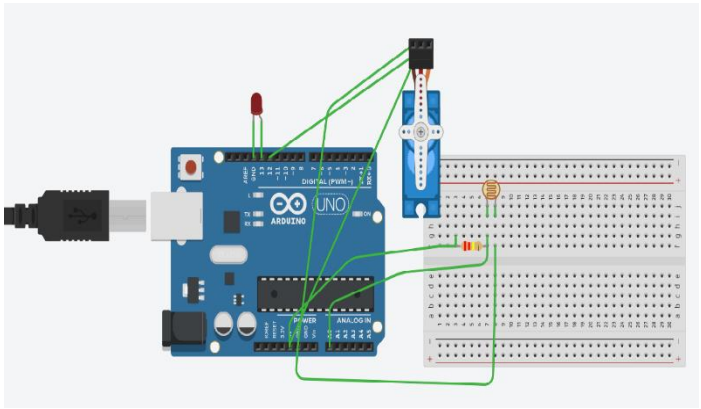
```
#include <Wire.h>
#include <SoftwareSerial.h>

#include <Servo.h>

double angle_rad = PI/180.0;
double angle_deg = 180.0/PI;
Servo servo_5;

void setup(){
    servo_5.attach(5); // init pin
}

void loop(){
    servo_5.write(45); // write to servo
    _delay(0.3);
    servo_5.write(60); // write to servo
    _delay(0.3);
    servo_5.write(90); // write to servo
    _delay(0.3);
    servo_5.write(135); // write to servo
    _delay(0.3);
    servo_5.write(180); // write to servo
    _delay(0.3);
    servo_5.write(0); // write to servo
    _delay(1);
    _loop();
}

void _delay(float seconds){
    long endTime = millis() + seconds * 1000;
    while(millis() < endTime)_loop();
}

void _loop(){
}
```

2. Buton ses uygulaması



```
#include <Wire.h>
#include <SoftwareSerial.h>

double angle_rad = PI/180.0;
double angle_deg = 180.0/PI;

void setup(){
    pinMode(3,INPUT);
    pinMode(11,OUTPUT);
}

void loop(){
    if(((digitalRead(3))==(HIGH))){
        digitalWrite(11,1);
    }else{
        digitalWrite(11,0);
    }
    _loop();
}

void _delay(float seconds){
    long endTime = millis() + seconds * 1000;
    while(millis() < endTime)_loop();
}

void _loop(){
}
```

3.Ldr sensör uygulaması



```
#include <Wire.h>
#include <SoftwareSerial.h>

#include <Servo.h>

double angle_rad = PI/180.0;
double angle_deg = 180.0/PI;
Servo servo_5;
Servo servo_13;
Servo servo_12;

void setup(){
    servo_5.attach(5); // init pin
    pinMode(A0+0,INPUT);
    pinMode(13,OUTPUT);
    servo_13.attach(13); // init pin
    servo_12.attach(12); // init pin
}

void loop(){
    servo_5.write(45); // write to servo
    _delay(0.3);
    servo_5.write(60); // write to servo
    _delay(0.3);
    servo_5.write(90); // write to servo
    _delay(0.3);
    servo_5.write(135); // write to servo
    _delay(0.3);
    servo_5.write(180); // write to servo
    _delay(0.3);
    servo_5.write(0); // write to servo
    _delay(1);
    if((analogRead(A0+0)) > (450)){
        digitalWrite(13,1);
        servo_13.write(90); // write to servo
        servo_12.write(90); // write to servo
    }
    digitalWrite(13,0);
    servo_13.write(0); // write to servo
    servo_12.write(0); // write to servo
    _loop();
}

void _delay(float seconds){
    long endTime = millis() + seconds * 1000;
    while(millis() < endTime)_loop();
}
```

BÖLÜM 4-ARDUİNO'NUN TEXT TABANLI KODLANMASI

Arduino Temel Komutlar

void setup()

{

// komutlar. Bu cihaz her açıldığında bir kere çalışacak komutları içerir. Genellikle ayar komutlarıdır.

}

Setup() fonksiyonu, Arduino'ya yüklenmiş olan **.ino** uzantılı kod parçasının, Arduino başlatıldığında veya yeniden başlatıldığında ilk çalıştırılan kısmıdır. Setup() fonksiyonu, çalışma ortamını başlangıç için bizlere hazırlar ve görevini tamamladıktan sonra bir yeniden başlatmaya kadar tekrar çalıştırılmaz.

void loop()

{

//komutlar. program boyunca sürekli çalıştırılacak komutlar buraya yazılır.

}

Loop() fonksiyonu, setup fonksiyonu çalıştırıldıktan sonra çalıştırılır ve bir sonsuz döngü işlevi görür. Loop fonksiyonun bu sonsuz döngü özelliği kullanılarak sürekli tekrar edecek olan işlemlerimizin gerçekleştirilmesini sağlanır. Örneğin; Arduino'nun belkide en temel örneği olan **Blink** örneğinde olduğu gibi 1'er saniye aralıkla bir ledi yakmak ve söndürmek gibi tekrar eden bir işlemlerde kullanılabilir.

*****Değişken kavramı ve tipleri*****

Değişkenler arduino hafızasında saklamak istediğimiz verileri tutmak için kullanılırlar. Örnek olarak arduinoyu bir dolap olarak düşünelim.

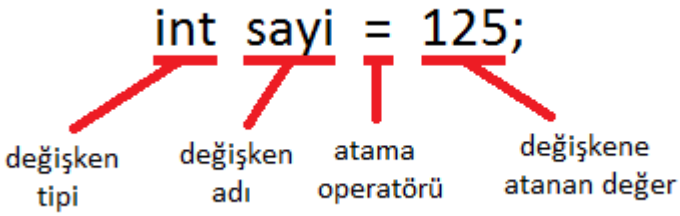
Değişken tanımlamak 3 adımdan oluşur. Bunlar;

1-Değişken türünü belirtmek

2-Değişken adını belirtmek

3-Değişkene değer atamak.

Değişken tanımlama işlemi aşağıdaki gibi yapılır.



Değişken tanımladıktan sonra hemen değer atama zorunluluğumuz yok. Değişkene çalışma zamanında da değer atayabiliriz.

Her şeyde olduğu gibi değişken tanımlamanın da bazı kuralları vardır. Bunlar;

- Değişken adı sayı ile başlamaz ancak sayı içerebilir
- Değişken adları Türkçe karakter içermez,
- Değişken adları özel karakter içermez,
- Değişken adları alt tire ile başlayabilir ve alt tire içerebilir,
- Değişken adları boşluk içermez.

İşte arduino programlarken kullanacağımız değişkenler ve tutabildiği veri tipleri:

1- bool :

Boolean değişken tipimiz 1 bitlik veri saklama kapasitesine sahip. Bu veri 1 veya 0 olabileceği gibi True (Doğru) veya False (Yanlış) olabilir.

2- char :

Karakter türü değişken tipidir. Bir byte yani 8 bitlik unicode karakter tipinde veri saklar. İşaretli -127 ve 127 arasında, işaretsiz 0 ve 255 arasında sayı tutabilir.

3- byte :

Sayı türü değişken tipidir. Bir byte yani 8 bit işaretsiz tam sayı türünden verileri tutabilir. 0 ile 255 arasındaki sayıları tutmak için kullanılır.

4- int :

Tamsayı türü değişken tipidir. Bir iki byte yani 16 bit işaretli tam sayı türünden verileri tutabilir. -32,768 ile 32,767 arasındaki sayıları tutmak için kullanılır.

5- usigned int :

Sayı türü değişken tipidir. Bir iki byte yani 16 bit işaretsiz tam sayı türünden verileri tutabilir. 0 ile 65535 arasındaki sayıları tutmak için kullanılır.

6- word :

Sayı türü değişken tipidir. Bir iki byte yani 16 bit işaretsiz tam sayı türünden verileri tutabilir. 0 ile 65535 arasındaki sayıları tutmak için kullanılır.

7- long :

Sayı türü değişken tipidir. Bir 4 byte yani 32 bit işaretli tam sayı türünden verileri tutabilir. -2,147,483,648 ile 2,147,483,647 arasındaki sayıları tutmak için kullanılır. Değer atanırken atanan değer sonuna "L" harfi konulur.

8- usigned long :

Sayı türü değişken tipidir. Bir 4 byte yani 32 bit işaretsiz tam sayı türünden verileri tutabilir. 0 ile 4,294,967,295 arasındaki sayıları tutmak için kullanılır. Değer atanırken atanan değer sonuna "L" harfi konulur.

9- short :

Sayı türü değişken tipidir. Bir iki byte yani 16 bit işaretli tam sayı türünden verileri tutabilir. -32,768 ile 32,767 arasındaki sayıları tutmak için kullanılır.

10- float:

Sayı türü değişken tipidir. 32 bit ondalıklı sayı türleri için kullanılır. -3.4028235E+38 ile 3.4028235E+38 arasındaki sayıları tutabilir.

11- double :

Sayı türü değişken tipidir. 32 bit ondalıklı sayı türleri için kullanılır. Kullanımı float ile aynıdır.

12- string :

Karakter türünde bir değişkendir. Kapsamı bellek boyutu ile sınırlıdır.

*****Şart (Koşul) Cümleleri*****

IF, eğer anlamına gelmektedir ve herhangi bir sorgulama işlemini kontrol ederek sonucunun doğru ya da yanlış olmasına göre farklı komutu/komutları çalıştırmaktadır. Örneğin, bir butona basıldı ise ledi yak, basılmadı ise ledi söndür…

IF-ELSE komutunun kullanımı önceki konuda anlatılan if yapısından farklı değildir. Sadece bize daha fazla komut satırı yazmamıza olanak sağlamaktadır. If koşulu yazıldıktan sonra koşul doğru ise if satırından, else satırına kadar olan komutlar işletilecektir. Yanlış ise else komut satırından sonra yazılan satır işletilir. Eğer else komut satırından sonra birden fazla komut satırı kullanılacak ise süslü parantez kullanılmalıdır"**{……..}**".

IF-ELSE IF Komutu:

Bu komutun diğer kullanım formatlarından farkı, birinci koşul sağlamadığı durumda ikinci bir koşulu sorgulatmamıza imkan verir.

if (şart)

{ komutlar ; }

if (şart)

{komutlar;}

else

{komutlar;}

```
if (x < 500) { digitalWrite(led, HIGH);}

else if (x >= 1000) { digitalWrite(led, LOW);}

else { // digitalWrite(led, HIGH);

delay(500);

digitalWrite(led, LOW);}
```

Bu tanımlamamızda x değeri 500 den küçük ise ledi aç,1000 den büyük ise ledi kapat, 500 den büyük 1000 den küçük ise ledi 500 ms açıp kapatacaktır.Burada if ile else if aynı yapıda, else operatörü ise tanımlanan ifadelerin dışındaki ifadeleri kapsamaktadır.Yani else operatörüne tanım aralığı girmiyoruz.

*****Döngüler*****

Döngüler; bir şart sağlandığı sürece belirli kod bloklarını tekrar tekrar çalıştıran yapılardır.

Arduino programlama dilinde 3 adet döngü vardır. Bunlar;

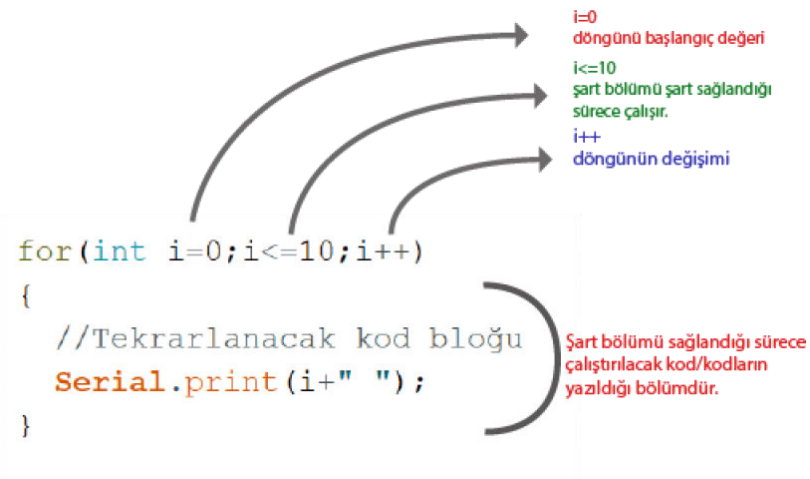
- For Döngüsü
- While Döngüsü
- Do-While Döngüsüdür.

Bu döngülerin söz dizimi farklı olsa da hepsi aynı işi görebilmektedir. Diğerlerinden farklı olarak yalnızca do-while döngüsü şartları kontrol etmeden bir defaya mahsus çalışmaktadır ve döngü sonunda şartları kontrol etmektedir. Şimdi dilerseniz bu döngülere teker teker bakalım.

For Döngüsü

for (değişkentanımlama;şart;işlem)

{komutlar}



Yukarıdaki örnekte **i=0** ile **i** değişkenine ilk değer olarak **0(sıfır)** girilmiştir. **i<=10** yani i değeri 10 ve altında olduğu sürece Serial ekranına (süslü parantez içinde sadece bu kod mevcut şuan) i değerini yazdıracaktır. i++ kısmında ise i değeri döngü her çalıştırıldığında bir arttırılmaktadır.
i=0 : döngünün başlangıç değerini ifade eder. Burada **i** bir değişkendir. Başka değişkenlerde başlangıç için kullanılabilir.
i<=10 : bu kısım döngünün şart kısmıdır. İlk değer verildikten sonra şart kontrol edilir. Şart doğru ise süslü parantez içindeki kodlar çalıştırılır. Sonra her artım yada azalımdan sonra **şart** tekrar kontrol edilir. Şart sağlanıyorsa süslü parantezler arası tekrar çalıştırılır.
i++: Döngünün kırılması için şart içinde kullanılan değişkenin değiştirildiği bölümdür. Bu örnekte şart 1 arttırılmıştır.

Örnek:

```
int Led = 13; // Arduino 13 numaralı dijital pinine LED bağlanıyor

void setup()
{
pinMode(Led, OUTPUT); // LED çıkış olarak tanımlanıyor
}
int gecikmeSuresi = 1000; // Gecikme süresi 1sn olarak bir değişkenle tanımlanıyor
void loop()
{
for(int i = 0; i <10;i++) // 10 defa döndür
{
digitalWrite(Led, HIGH); // LED yanıyor
delay (gecikmeSuresi); // Gecikme süresi kadar bekleniyor
digitalWrite(Led, LOW); // LED söndürülüyor
delay(gecikmeSuresi); // Gecikme süresi kadar bekleniyor
```

```
}
delay(2000); // 2 sn bekleniyor
}
```

While Döngüsü

while(şart)

{komutlar}

While döngüsü genellikle bir koşula bağlanır. Eğer koşul doğru ise döngü devam eder. Koşul bozulduğunda döngü biter ve program kaldığı yerden devam eder.

Örneğin while dışında 'h' değişkeni tanımlayalım. Bu değişkenin ilk değeri 5 olsun. Eğer 'h' değişkeni 100'den küçükse döngü devam etsin. Döngünün her turunda da 'h' değişkeninin değeri 2 katına çıksın.

Dikkat: Her döngüde iki katına çıkan değişkenlerin bulunduğu kodlarda genellikle ilk değişken hatası yapılmaktadır. Değişkenin ilk değeri kesinlikle belirlenmeli ve çarpma işlemi olduğu için bu değer '0' olmamalıdır. Aksi durumda koşul sonsuza kadar doğru olur ve döngü hiçbir zaman sona ermez.

Örnek:

```
int h = 5;
while( h < 100){
/*
Burası h değişkeni 100den küçük olduğu sürece çalışacak
*/
h = h * 2;
/*
Üst satırdaki h'ı iki katına çıkartma işlemi alttaki satırdaki gibi de tanımlanabilirdi
h *= 2;
Eğer h'ın değeri arttırılmazsa döngü koşulu doğru olduğu için döngü sonsuza kadar devam edecektir
*/

}
```

Do…While Döngüsü

do

{komutlar;}

while(şart); //Noktalı virgüle dikkat

do – while döngüsü while döngüsü ile aynıdır. Tek farkı döngüye bir kez girilir, şart döngü sonunda kontrol edilir. Dolayısıyla şart yanlış olsa bile döngüye 1 kez girilir.

Örnek:

```
int i = 0;
do
{
   i++;
   Serial.println(i);
   delay_ms(100);
}while(i < 100)
Serial.println("Donguden cikildi");
```

Örnekte i adında içeriği 0 olan tam sayı değişkeni tanımlanmıştır. do-while döngüsü şartında i 100'den küçük olduğu sürece döngü çalışacaktır. Döngüye girildiği an i değer 1 olur ve seri portdan gönderilir. 100 msn beklendikten sonra şart kontrol edilir. En son i değeri 100 olacağı için ekranda 1'den 100'e kadar olan sayılar 100 msn aralıklarla gönderilmektedir.

break; // Döngüden çıkma.

continue; //Döngüyü terk etmeden bir adımın atlanması isteniyorsa kullanılır.

return deger; // Fonksiyondan değer döndürür

etiket: goto etiket; // Hiç tavsiye edilmeyen etiket kullanım biçimidir.

Özel işaretler

; komut satırı sonu

{} çoklu komut satırı

// tek satır açıklama

/* çok satır açıklama */

Noktalı Virgül " ; " : Noktalı virgül orta seviye programlama dillerinde satırın bittiğini belirtmek için kullanılır. Programda en çok kullanılan ve unutulması nedeniyle en çok hataya sebep olan karakterdir. Döngü, kontrol yapıları ve fonksiyon tanımlamaları hariç hemen hemen her satırın sonunda kullanılırlar.

Yorum Satırı " // " : Yorum satırı programlamada satırlara, o satırda ne yapıldığına dair açıklama yapmak için kullanılır. Tek satırdan oluşur. Alt satıra geçildiğinde özelliğini yitirir. Türkçe karakter kullanılabilir. Bu işaretin konulduğu yerden sonra yazılan yazılan programda derlenmez ve çalışmaz.

Yorum Alanı " /* */ " : Birden fazla satırda açıklama yapmak için kullanılır. Açılış işareti /* ve kapanış işareti */ 'dır.

*****Program başı direktifler*****

#define komutu

#define degisken deger // const yerine kullanılabilecek bir deyim olup ; kullanılmadan yazılır

#define : Sabit tanımlamak için kullanılır. Sabitlere tanımlandığında atanan değer program boyunca değişmez. Örnek olarak pi sayısı değeri 3,14'tür. Biz bu sayının program içerisinde hiç değişmesini istemeyiz. Böyle durumlarda sabit tanımlamaya ihtiyaç duyabiliriz.

Örnek olarak pi sayısını sabit olarak tanımlayalım;

#define pi = 3.14;

Yukarıdaki kodu yazdığımızda programımız artık her "pi" gördüğü yerde 3.14 sayısı varmış gibi davranır.

Kütüphane ekleme- include komutu

Yeni bir kütüphane eklemek için kütüphane dosyalarını Arduino programını kurduğunuz dizinin altında bulunan 'libraries' klasörüne taşıyın. Eğer bu sırada Arduino programı açıksa, taşıma işlemi bittikten sonra, kapatıp yeniden açın. Dosyanın en başında kütüphaneyi projenize ekleyin. Bunun için aşağıdaki kodu kullanabilirsiniz.

#include <kutuphaneadi.h>

Artık kütüphanenin içerisinde bulunan fonksiyonları kullanabilirsiniz.

#include <servo.h >

*****Aritmetik operatörler*****	*****Karşılaştırma Operatörleri*****
= //atama	== //eşittir
+ //ekleme	!= //eşit değildir
- //çıkarma	< //küçüktür
* //katlama	> //büyüktür
/ //bölme	<= //küçük eşit
% //mod alma	>= // büyük eşit

*****Boolean Operatörler*****

&& // Ve

|| // veya

! //değil

*****Bit işlem operatörleri*****

& //ve

| //veya

^ //XOR

~ //değil

<< // sola kaydır

>> // sağa kaydır

*****Birleşik Operatörler*****

++ // bir artır

-- // bir azalt

+= // artır

-= // azalt

*= // çarp

/= // böl

&= /// bit and

|= // bit or

BÖLÜM 5- ANALOG İŞLEMLER

Analog Giriş Pimleri:

Arduino üzerindeki ATmega mikrodenetleyicisi, analogdan dijitale dönüştürücü içermektedir. Dönüştürücü 10 bit çözünürlüğe sahiptir ve 0 ile 1023 arasında tamsayı döndürür. Arduinoda UNO'da kullanılan analog 6 pim (A0, A1, ... A5) vardır. Arduino kullanıcısı için analog pimlerin ana işlevi analog sensörleri okumak olsa da, analog pimler genel amaçlı giriş / çıkış pinlerinin tüm işlevlerine sahiptir.

analogRead( ) Fonksiyonu:

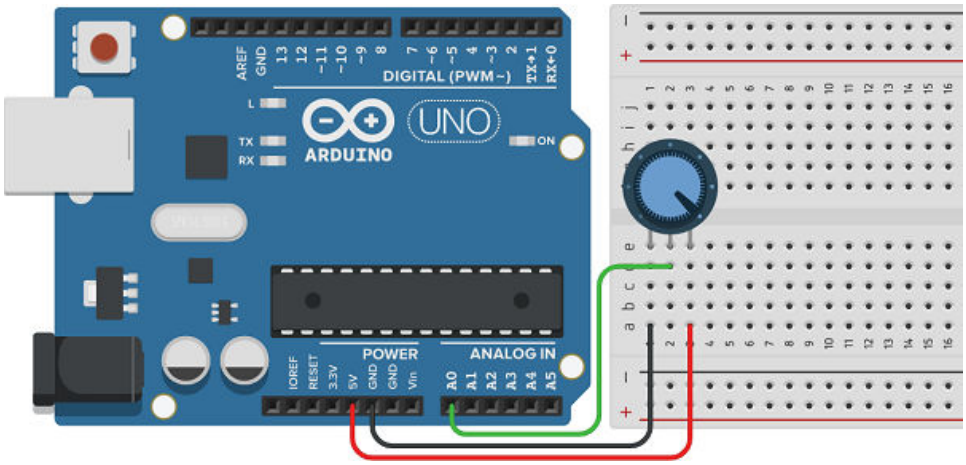
Belirtilen analog pimden değeri okur. Arduino çok kanallı dönüştürücülerin 0 ile 5V arasındaki giriş voltajlarını 0 ile 1023 arasındaki tamsayı değerlere eşleyeceği anlamına gelir.

Potansiyometre ile Analog Değer Okuma:

Gerekli Malzemeler:

- Arduino UNO
- Breadboard
- Potansiyometre
- Jumper kablolar

Devre Şeması:



Blok Kodu:

IDE Kodu:

```
int x = 0;
void setup( ) {
  Serial.begin(9600);
}
void loop( ) {
  x = analogRead(A0);
  Serial.println(x);
}
```

Kodu yükledikten sonra potansiyometrede yaptığınız değişiklikleri seri port ekranından gözlemleyebilirsiniz.

Not 1 :

Potansiyometrenin 5V ile GND'ye bağlı pimlerini yer değiştirerek seri port ekranında yazan sayıların artış-azalış yönünü tersine çevirebilirsiniz.

Not 2 :

Arduino IDE programında Dosya>Örnekler>01.Basics>AnalogReadSerial adresinden örnek kodu da yükleyebilirsiniz.

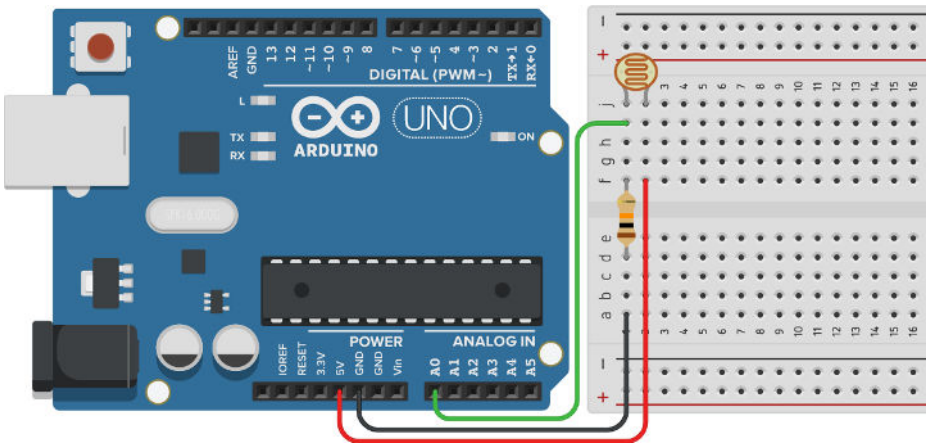
Kodlar hazır yüklüyken fotodirenç (LDR) devresi kurarak yeni bir uygulama yapalım.

Fotodirenç ile Analog Değer Okuma:

Gerekli Malzemeler:

- Arduino UNO
- Breadboard
- Fotodirenç
- 10 kΩ direnç
- Jumper kablolar

Devre Şeması:



Kodu daha önce yüklediğimiz için fotodireç üzerine farklı parklaklıkta ışık tutarak yaptığınız değişiklikleri seri port ekranından gözlemleyebilirsiniz.

Not :

Fotodirecin 5V ile GND'ye bağlı pimlerini yer değiştirerek seri port ekranında yazan sayıların artış-azalış yönünü tersine çevirebilirsiniz. 10 kΩ direnç yerine farklı dirençler kullanarak seri port ekranında yazan sayıların hassasiyetini değiştirebilirsiniz.

analogWrite( ) Fonksiyonu:

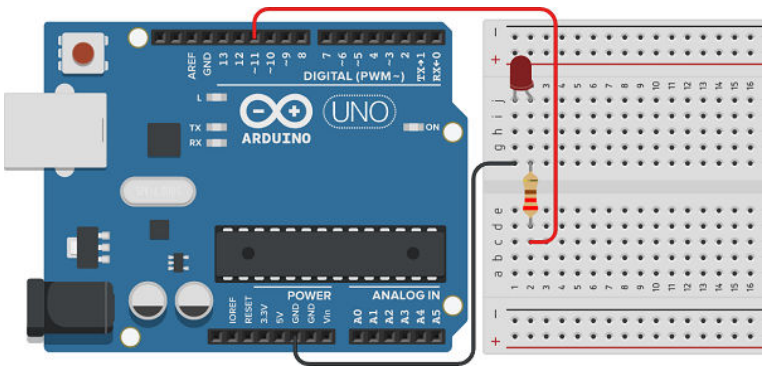
Belirtilen PWM (Arduino UNO'da 3, 5, 6, 9, 10 ve 11 pimleri) dijial pimlerine analog değer yazar. Bir LED'i farklı parlaklıklarda yakmak veya çeşitli hızlarda bir motor sürmek için kullanılabilir.

Farklı Parlaklıkta Led Yakma (for döngüsü):

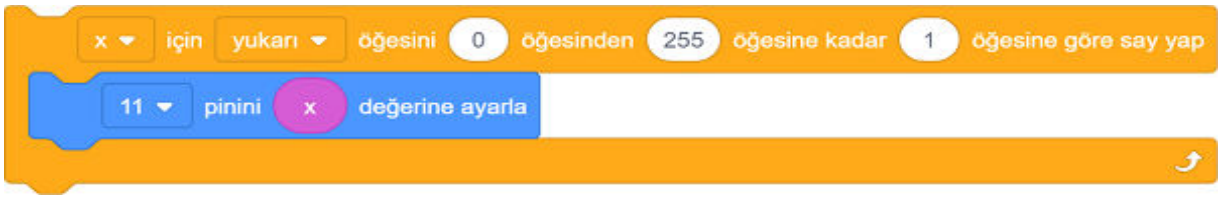
Gerekli Malzemeler:

- Arduino UNO
- Breadboard
- Led
- 220 Ω direnç
- Jumper kablolar

Devre Şeması:



Blok Kodu:



IDE Kodu:

```
int x = 0;
void setup( ) {
 pinMode(11, OUTPUT);
}
void loop( ) {
 for (x = 1; x <= 255; x += 1) {
  analogWrite(11, x);
 }
}
```

Kodu yükledikten sonra leddeki değişiklikleri gözlemleyebilirsiniz.

Not 1 : Artış miktarını değiştirerek ledin parlama periyodunu kısaltabilirsiniz.

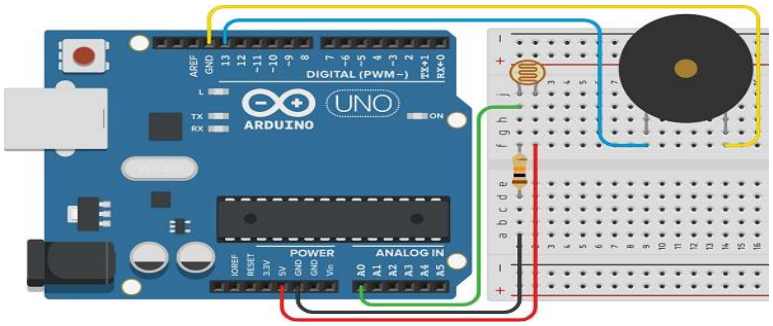
Not 2 : Arduino IDE programında Dosya>Örnekler>03.Analog>Fading adresinden örnek kodu da yükleyebilirsiniz. (ledPin = 9 değerini ledPin = 11 yapınız)

Foto**direnç** ile Buzzer Kontrolü (if döngüsü):

Gerekli Malzemeler:

- Arduino UNO
- Breadboard
- Aktif Buzzer
- Fotodirenç
- 10 kΩ direnç
- Jumper kablolar

Devre Şeması:



Blok Kodu:

IDE Kodu:

```
int x = 0;
void setup( ) {
 pinMode(13, OUTPUT);
}
void loop( ) {
 x = analogRead(A0);
 if (x < 500) {
  digitalWrite(13, HIGH);
 } else {
  digitalWrite(13, LOW);
 }}
```

Kodu yükledikten sonra fotodireç üzerine farklı parklaklıkta ışık tutarak buzzerdan ses çıkmasını sağalayabilirsiniz.

Not :

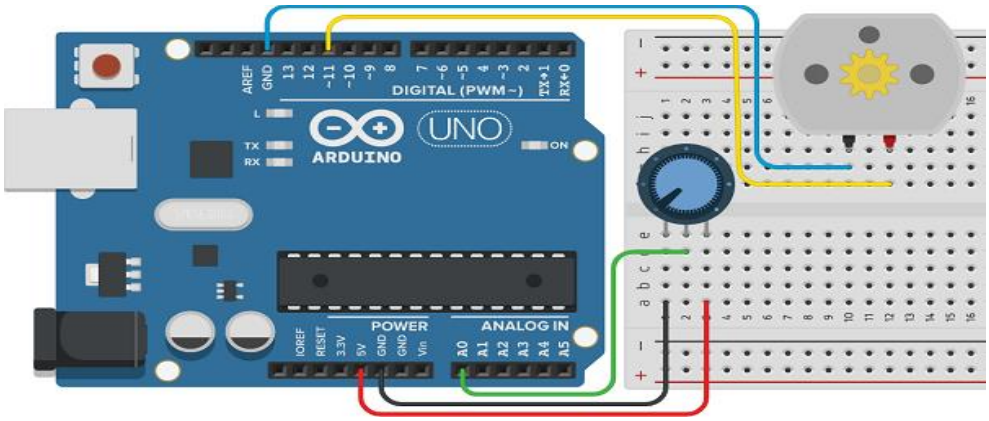
Koşuldaki sayıyı değiştirerek kontrolü sağlayabilirsiniz. Kodlarda değişiklik yaparak (tone, noTone kodu) pasif buzzer kullanılarak da yapabilirsiniz. Fotodirenç üzerine lazer ışık tutarak güvenlik sistemi yapabilirsiniz.

Potansiyometre ile Motorda Hız Kontrolü (map fonksiyonu):

Gerekli Malzemeler:

- Arduino UNO
- Breadboard
- **Potansiyometre**
- **Motor**
- Jumper kablolar

Devre Şeması:



Blok Kodu:

IDE Kodu:

```
int x = 0;
void setup( ) {
 pinMode(11, OUTPUT);
}
void loop( ) {
 x = analogRead(A0);
 x = map(x, 0, 1023, 0, 255);
 analogWrite(11, x);}
```

Kodu yükledikten sonra potansiyometrede yaptığınız değişikliklerle motordaki hızlanma-yavaşlamayı gözlemleyebilirsiniz.
Not : Arduino IDE programında Dosya>Örnekler>03.Analog>AnalogInOutSerial adresinden örnek kodu da yükleyebilirsiniz. (analogOutPin = 9 değerini analogOutPin = 11 yapınız)

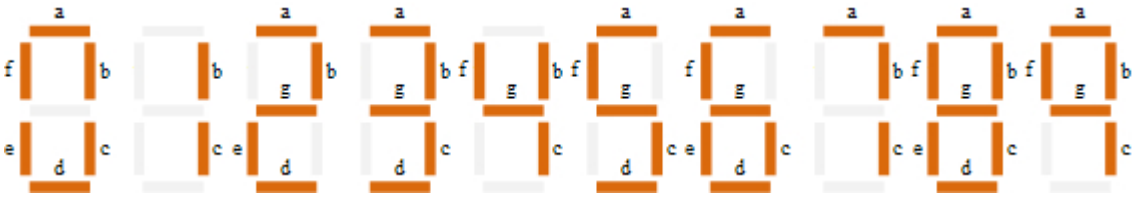
BÖLÜM 6:GÖSTERGE OLARAK 7SEGMENT DİSPLAY VE LCD KULLANIMI

DİSPLAY NEDİR?ÇALIŞMA PRENSİBİ NASILDIR?

Günlük hayatta bir çok alanda karşımıza çıkan ve 7 Segment Display diye adlandırılan sayısal göstergeler oldukça yaygın şekilde kullanılan elektronik devre elemanlarıdır. 7 Segment Display'ler LED tabanlı göstergelerdir. İçerisinde bulunan 7 adet LED ile aydınlatılan 7 ayrı bölmeden oluşan bir sistem bütünüdür. 7 Segment Display'lerde LED'lerin kullanılmasının en önemli nedeni LED'lerin çok küçük hacimlerde olan türlerinin olmasıdır.

Elemanın yapısında bulunan 7 LED'in her biri bir segment olarak adlandırılır.
Çünkü sayısal rakamların (onluk ve on altılık tabanlarda) kesim formları görüntülenmek üzere parçalar halinde aydınlatılmıştır. Ayrıca 7 Segment Display'in yapısında ek olarak 1 LED daha bulunmaktadır. Bu Decimal point(DP) olarak adlandırılan bu LED sayısal göstergelerde küsuratlı sayıların ondalık noktalarının gösterilmesi için kullanılır.

Elemanın yapısındaki 7 LED'i, 7 bölge olarak düşünecek olursak;konumsal olarak kısımlandırılmış bu 7 bölge "a" ile "g" harfleri arasındaki 7 harfle (a,b,c,d,e,f,g,) etiketlendirilir.



Şekil1.7Segment Display Doğruluk Tablosu

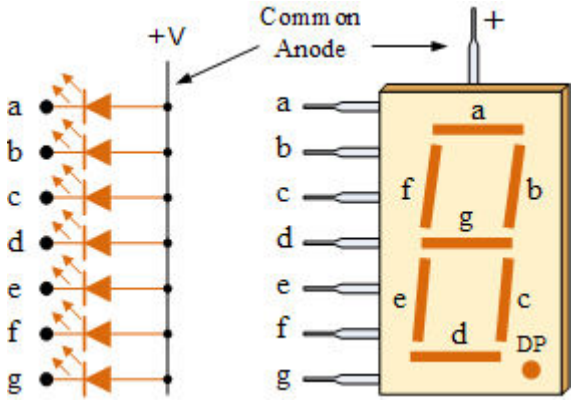
7 Segment Display'de LED'lerin bir uçları ortak uç olduğundan bu ucun toprak ya da besleme ucu olmasına göre 7 Segment Display'ler ortak anot ve ortak katot olmak üzere iki çeşide ayrılırlar.

Displaylerin iki tipi vardır:

Ortak Anotlu Display **Ortak Katotlu Display**

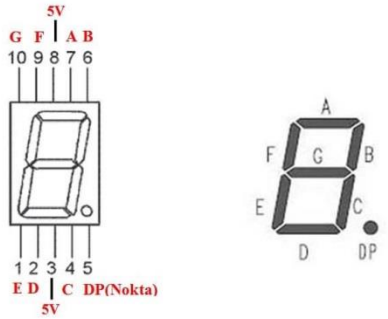
Ortak Anotlu 7Segment Display:

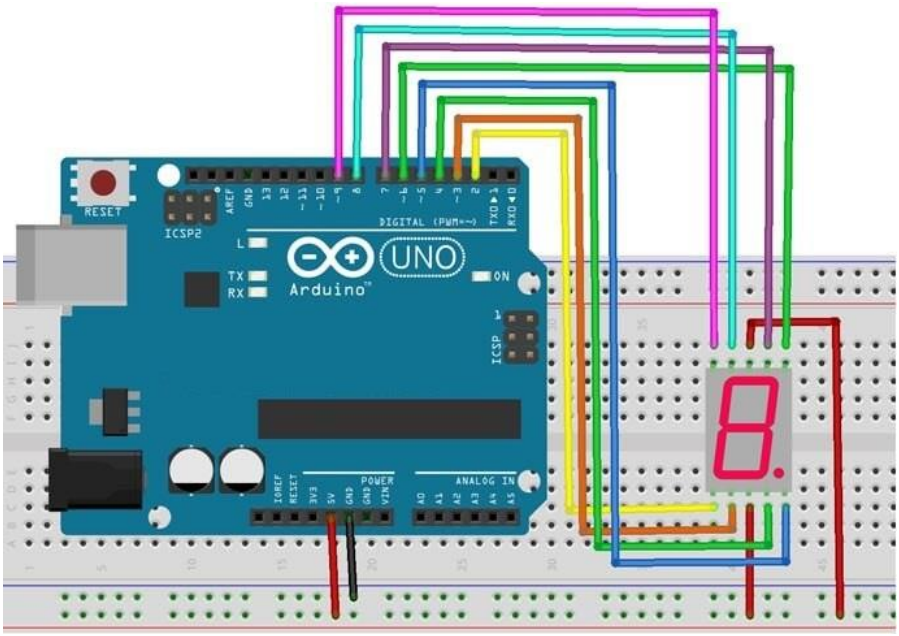
LED'lerin anot uçları birleştirilerek, 7 Segment Display'in arkasından çıkan ortak anot pinine bağlanmıştır. Yukarıdaki görüntüleme elemanı devreye bağlanırken ortak anot ucu güç kaynağının pozitif ucuna, a-g arasında harflendirilen 7 katot ucu devrenin yapısına göre ya direk olarak güç kaynağının negatif ucuna ya da 7 Segment Display sürmeye yarayan entegrelerin bacaklarına bağlanır. Burada da dikkat edilmesi gereken nokta LED'lerin yüksek güce maruz kalmamaları için ortak anot ucunu güç kaynağına ya da entegrelere bağlamadan önce, LED'lerin anot uçlarından hemen öncesine, devrenin ortak anottan uygulanan gerilim değerine göre, uygun değerde bir direnç bağlanmasıdır. Aksi taktirde görüntületicinin yapısındaki LED'ler kullanılmaz hale gelebilir.



PEKİ ORTAK ANOTLU 7SEGMENT DİSPLAY 'İ ARDUİONO İLE NASIL KULLANABİLİRİZ?

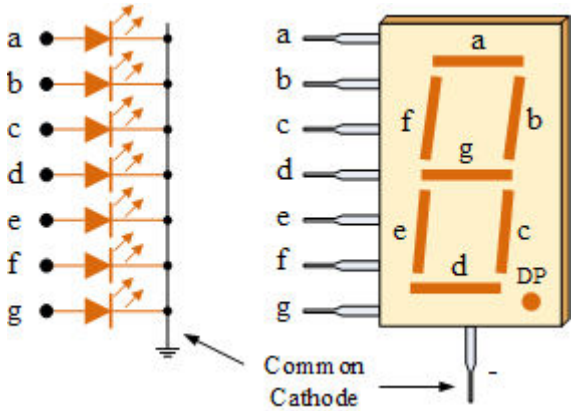
Ortak anot 7 segment display, tüm ledlerin + bacaklarının ortak kullanıldığı led display çeşitidir. Bu durumda 3. ve 8. bacaklar arduino 5V pinine bağlanmalıdır.





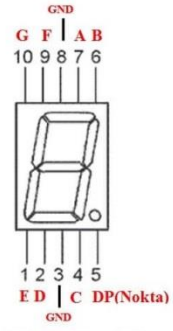
Ortak Katotlu 7Segment Display:

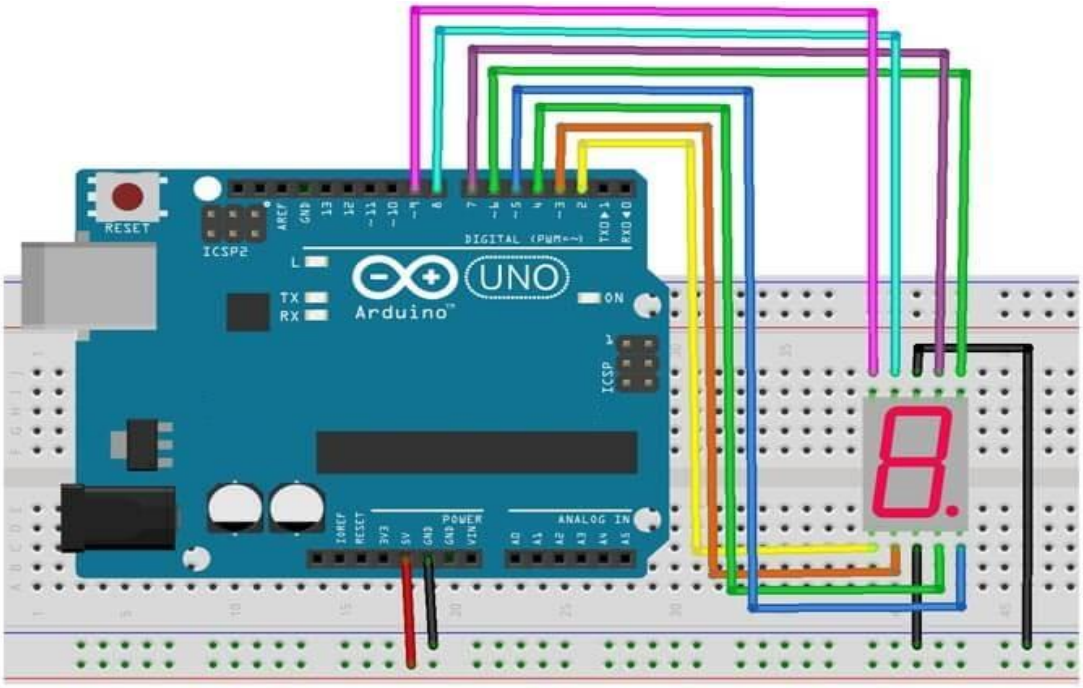
7 Segment Display'in arkasından çıkan ortak katot pinine bağlanmıştır. Görüntüleme elemanı devreye bağlanırken ortak katot ucu güç kaynağının negatif ucuna, a-g arasında harflendirilen 7 anot ucu devrenin yapısına göre ya direk olarak güç kaynağının artı ucuna ya da 7 Segment Display sürmeye yarayan entegrelerin bacaklarına bağlanır. Burada dikkat edilmesi gereken nokta, görüntüleyicinin arkasından çıkan 7 adet anot ucunu, güç kaynağına ya da entegrelere bağlamadan önce, LED'lerin anot uçlarının hemen öncesine, devrenin anot uçlarından uygulanan gerilim değerine göre, uygun değerde bir direnç bağlanmasıdır. Çünkü görüntüleyici yapısındaki LED'ler güce maruz kaldığında kullanılmaz hale gelebilirler.



PEKİ ORTAK KATOTLU 7SEGMENT DİSPLAY 'İ ARDUİONO İLE NASIL KULLANABİLİRİZ?

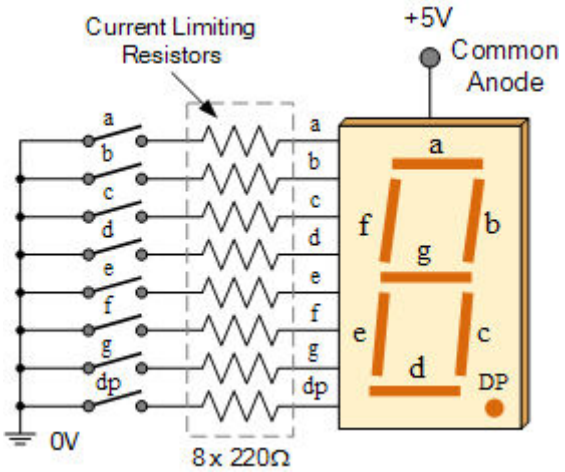
Ortak katot 7 segment display, tüm ledlerin – bacaklarının ortak kullanıldığı led display çeşitidir. Bu durumda 3. ve 8. bacaklar arduino GND pinine bağlanmalıdır.





7 Segment Display'in Sürülmesi:

7 Segment Displaylerin yapısında bulunan LED'lerin her LED kullanımında olduğu gibi aşırı akıma karşı korunmaları gerekmektedir. LED'lerin ışık güçleri artan akımla birlikte,bir noktadan sonra LED aşırı akımla üzerine yüklenen yükü kaldıramaz duruma gelir ve yüksek akımdan dolayı yanar. Bu tür aksaklıkların engellenmesi için LED'lerin bulunduğu kol üzerine harici dirençlerin bağlanması gerekmektedir.



PEKİ DİSPLAY İLE 0'DAN 9'A KADAR SAYMA ÖRNEĞİNİ İNCELEYELİM

Malzemeler:

Arduino Uno

Ortak Katotlu 7Segment Display

Jumper

Ortak Katotlu Display'imizi yukarıdaki şemaya göre bağlıyoruz.

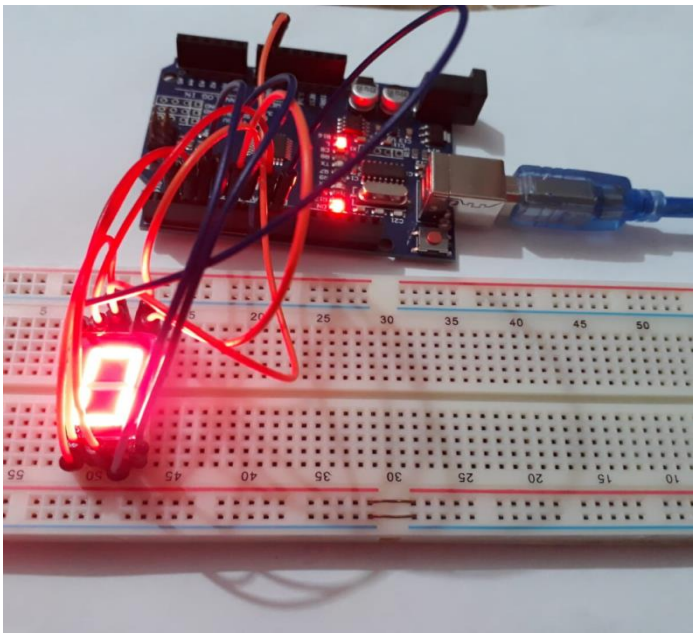
Şimdi de yazılım aşamasına geçelim:

```
int E=2;
int D=3;
int C=4;
int P=5;
int B=6;
int A=7;
int F=8;
int G=9;
void setup() {
pinMode(E, OUTPUT);
pinMode(D, OUTPUT);
pinMode(C, OUTPUT);
pinMode(A, OUTPUT);
pinMode(B, OUTPUT);
pinMode(F, OUTPUT);
pinMode(G, OUTPUT);
pinMode(P, OUTPUT);
}
void loop() {
for(int i=0; i<=9;i++){
RakamYaz(i);
delay(500);
}
}
void RakamYaz(int rakam)
{
switch(rakam)
{
case 0 :
digitalWrite(E,HIGH);
digitalWrite(D,HIGH);
digitalWrite(C,HIGH);
digitalWrite(B,HIGH);
digitalWrite(A,HIGH);
digitalWrite(F,HIGH);
digitalWrite(G,LOW);
break;
case 1:
digitalWrite(E,LOW);
digitalWrite(D,LOW);
digitalWrite(C,HIGH);
digitalWrite(B,HIGH);
digitalWrite(A,LOW);
digitalWrite(F,LOW);
digitalWrite(G,LOW);
break;
case 2:
```

```
digitalWrite(E,HIGH);
digitalWrite(D,HIGH);
digitalWrite(C,LOW);
digitalWrite(B,HIGH);
digitalWrite(A,HIGH);
digitalWrite(F,LOW);
digitalWrite(G,HIGH);
break;
case 3:
digitalWrite(E,LOW);
digitalWrite(D,HIGH);
digitalWrite(C,HIGH);
digitalWrite(B,HIGH);
digitalWrite(A,HIGH);
digitalWrite(F,LOW);
digitalWrite(G,HIGH);
break;
case 4:
digitalWrite(E,LOW);
digitalWrite(D,LOW);
digitalWrite(C,HIGH);
digitalWrite(B,HIGH);
digitalWrite(A,LOW);
digitalWrite(F,HIGH);
digitalWrite(G,HIGH);
break;
case 5:
digitalWrite(E,LOW);
digitalWrite(D,HIGH);
digitalWrite(C,HIGH);
digitalWrite(B,LOW);
digitalWrite(A,HIGH);
digitalWrite(F,HIGH);
digitalWrite(G,HIGH);
break;
case 6:
digitalWrite(E,HIGH);
digitalWrite(D,HIGH);
digitalWrite(C,HIGH);
digitalWrite(B,LOW);
digitalWrite(A,HIGH);
digitalWrite(F,HIGH);
digitalWrite(G,HIGH);
break;
case 7:
digitalWrite(E,LOW);
digitalWrite(D,LOW);
digitalWrite(C,HIGH);
digitalWrite(B,HIGH);
digitalWrite(A,HIGH);
digitalWrite(F,LOW);
digitalWrite(G,LOW);
break;
case 8:
digitalWrite(E,HIGH);
digitalWrite(D,HIGH);
digitalWrite(C,HIGH);
digitalWrite(B,HIGH);
digitalWrite(A,HIGH);
digitalWrite(F,HIGH);
digitalWrite(G,HIGH);
break;
case 9:
digitalWrite(E,LOW);
digitalWrite(D,HIGH);
digitalWrite(C,HIGH);
digitalWrite(B,HIGH);
digitalWrite(A,HIGH);
digitalWrite(F,HIGH);
```

digitalWrite(G,HIGH); }

break; }

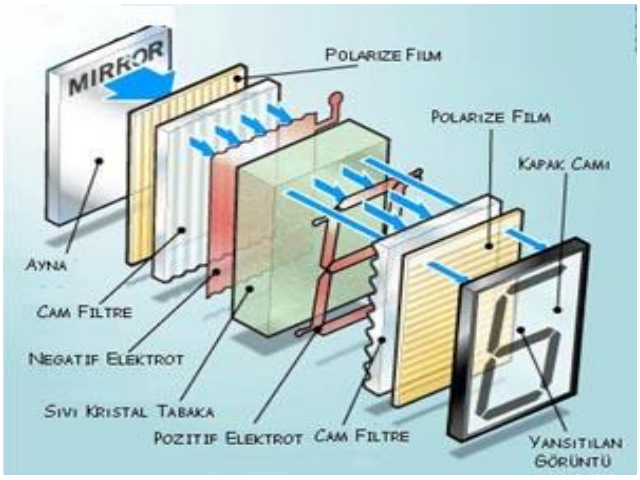


Örneğimizin nasıl çalıştığının videosunu proje youtube kanalımızdan ya da aşağıdaki link ile ulaşabilirsiniz.

https://www.youtube.com/watch?v=gUH-S0A4saM

LCD NEDİR?ÇALIŞMA PRENSİBİ NASILDIR?

LCD, Liquid Crystal Display yani Sıvı Kristal Ekran elektrikle kutuplanan sıvının ışığı tek fazlı geçirmesi ve önüne eklenen bir kutuplanma filtresi ile gözle görülebilmesi ilkesine dayanan bir görüntü teknolojisidir.



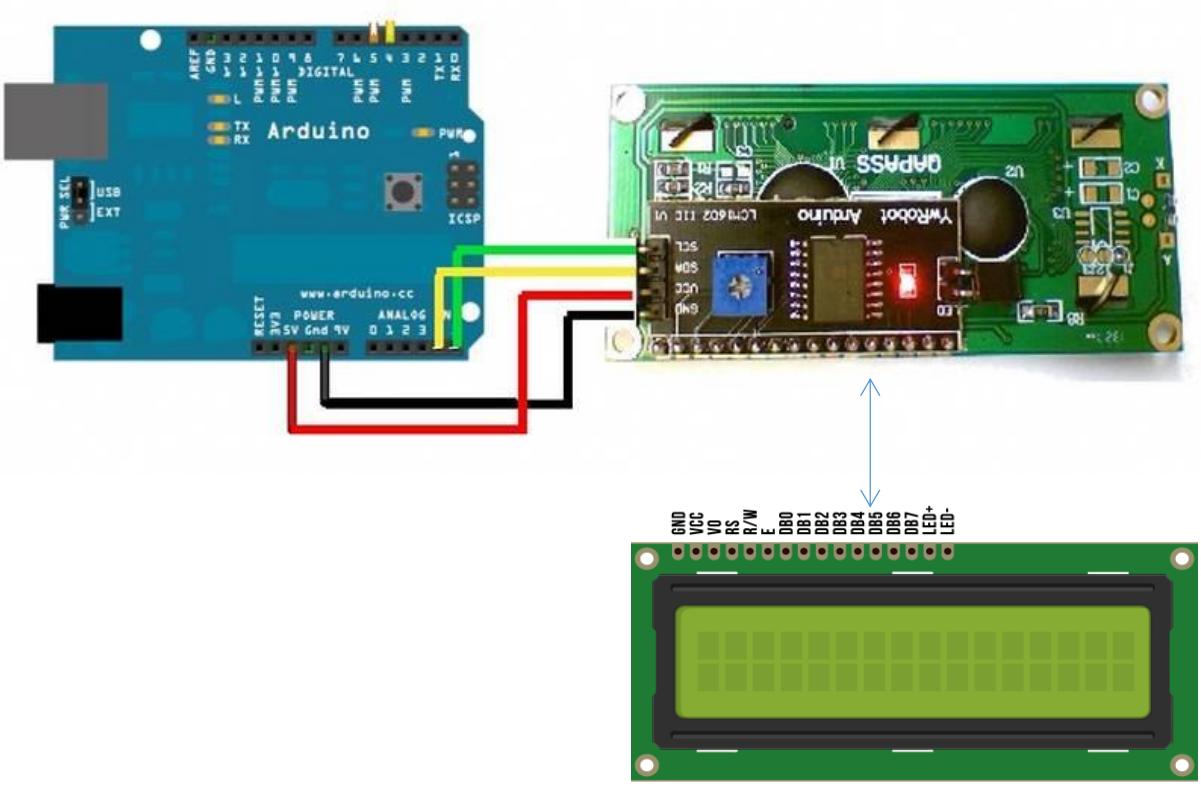
LCD'lerin yapısı şekilde de görüldüğü gibi farklı katmanlardan oluşmaktadır. LCD panelleri robot projelerinde ya da otomasyon projelerinde kullanmak için bilgisayarınızın seri ya da parelel portundan veya bir PIC mikrodenetleyici kullanarak kontrol edebilir veya Arduiono vb. projelerinizde kullanabilirsiniz. LCD paneller piyasada sütun ve satır sayılarına göre 8x1, 8x2, 16x1, 16x2, 20x1, 20x2, 40x1 ve 40x2 gibi farklı boyutlarda bulunmaktadır. Bunlar arasında robot projelerinde yaygınlıkla 16x2 boyutlarındaki LCD paneller kullanılmaktadır.

Günümüzde üretilen LCD panellerin çoğunda tek sıra halinde 16 pin bulunur. Bu pinlerden ilk 14 tanesi kontrol için son iki tanesi ise eğer varsa arka ışık için kullanılır. Bazı LCD 'lerde kontrol için kullanılan 14 pin 2 adet 7 li sıra halinde de bulunabilir.

LCD Pin No:	**LCD PİN:**	**Fonksiyonları:**
1	Vss	Toprak(Ground)
2	Vcc	+5V
3	VEE	Kontrast
4	RS	Register Select
5	RW	Read/Write
6	E	Enable
7-14	D0-D7	Data Girişleri
15	BL+	Arka Panel Işığı Pozitif Ucu
16	BL-	Arka Panel Işığı Negatif Ucu

PEKİ LCD'Yİ ARDUİNO İLE NASIL KULLANABİLİRİZ?

Yukarıda LCD'nin çalışma prensibini görmüştük,şimdi ise Arduino Projemizde nasıl kullanabileceğimize bakalım.



LCD'yi şekilde görüldüğü şekilde Arduino'ya bağlıyoruz. Bağlantıyı tamamladıktan sonra Arduino'yla kodlama kısmına geçiyoruz. Ancak burada dikkat etmemiz gereken bir nokta var;eğer arduino kütüphanesinde LCD'yi çalıştırmak için gerekli olan içerik yüklü değilse yazdıklarımızı tanımlayamayacaktır. O yüzden örneğimizde kullanacağımız "I2C" kütüphanesini eklememiz gerekiyor.

Malzemeler:

Arduino

LCD 16x2

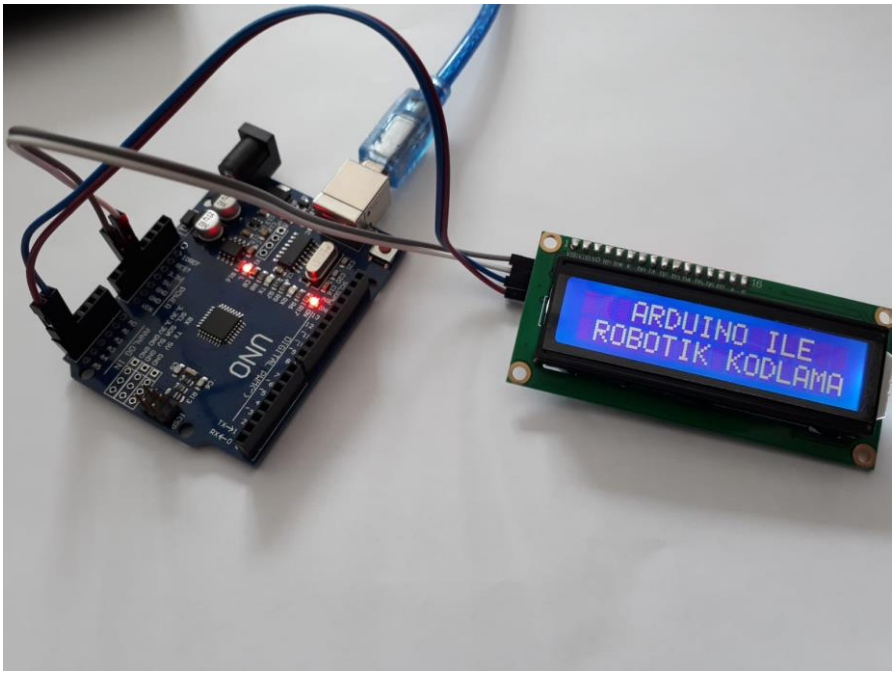
Jumper

LCD Kodları:

```
#include <LiquidCrystal_I2C_AvrI2C.h>

LiquidCrystal_I2C_AvrI2C lcd(0x27,16,2);

void setup()

{

  lcd.begin();

  lcd.backlight();


  lcd.setCursor(3,0);
```

Bilindiği gibi LCD'imizin her satırına 16karakter yazdırabiliyoruz. "Arduino ile" yazısı toplamda boşluk ile beraber 11karakter geriye 5karakterlik yer kalıyor. Lcd.setCursor(3,0) yazma sebebimiz yazıyı ortalamak adına baştan "3 " karakter atlatıyor olmamız, yanındaki "0" ise 1.satırı temsil ediyor.

```
  lcd.print("ARDUINO ILE");


  lcd.setCursor(1,1);
  lcd.print("ROBOIİK  KODLAMA");
}

void loop()
{
}
```



BÖLÜM 7-DC MOTOR KULLANIMI

DC motorlar kullanımı kolay olduğundan endüstride sıkça kullanılmaktadır. Kullanımları oldukça basit olduğundan sıkça kalkış, durma, frenleme ve devir yönü değişimi gerektiren çeşitli endüstriyel uygulamalarda yaygın bir şekilde kullanılmaktadır. Arduino pinlerinden verilebilen akım motorları çalıştırmak için yeterli olmamaktadır. Tüm bu uygulamaların gerçekleştirilebilmesi ve DC motor için gerekli akımın sağlanabilmesi için motor sürücüler kullanılmalıdır. Motor sürücü olarak akımı kuvvetlendirmek ve motoru kontrol etmek için L298 entegresini kullanacağız. Benzer entegreler de aynı görevi yapmaktadır.

L298N Voltaj Regulatörlü Çift Motor Sürücü Kartı piyasada rahatlıkla bulunabilen ucuz bir motor sürücü kartıdır.

Dönüş hızı uygulanan DC gerilimle orantılıdır. **ENABLE A ve ENABLE B, b**u iki pin motorların dönüş hızını ayarlamak için kullanılır. Bu yüzden bu pinleri Arduino'nun PWM ayaklarına bağlamamız gerekir. Eğer hız kontrolü yapılmayacaksa bu pinler 5 volt hattına bağlanabilir.

PWM, 0-5V aralığında değişen farklı gerilim değerlerinin çıkışa aktarılmasıdır. Ancak bu sinyal gerçekte birebir analog sinyal değildir. Yalnızca dijital formda üretilen palslerin süresi ile oynama yapılarak analog sinyalin taklit edilmesinden ibarettir. PWM sinyal çözünürlüğü 8-bittir. Yani 0 ile 255 aralığında girdiğimiz değerler 0 ile 5V aralığında karşılanmaktadır. Örneğin 0V, 0 değerine karşılık gelirken 5V – 255 değerine karşılık gelmektedir.

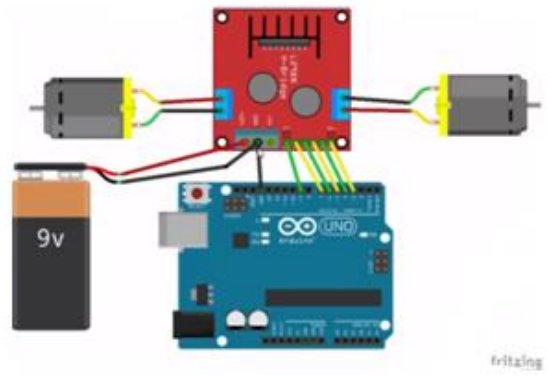
Enable girişine 255 değeri girildiğinde L298 motor sürücüsü motorlara Vcc geriliminin tamamını uygular ve motorun hızlı dönmesini sağlar. 150 değeri girildiğinde ise daha düşük gerilim uygulayarak motorun daha yavaş dönmesi sağlanır. Örneğin Vcc=9V olduğunu düşündüğümüzde 9/255*150=5,29V motora uygulanmış olur.

INPUT 1, 2, 3 ve 4 pinleri motorların dönme yönünün kontrolü için Arduino'ya bağlanır. INPUT 1 ve 2 pinleri 1. motorun, INPUT 3 ve 4 pinleri ise 2. motorun kontrolünde kullanılır. Örneğin 1. Motorun kontrolü için, INPUT 1 pini 5 volt, INPUT 2 pini 0 volt yapılır ise motor ileri yönde dönmeye başlar. Eğer INPUT 1 pini 0 volt ve INPUT 2 pini 5 volt yapılır ise motor geri yönde dönmeye başlar. İki pinin aynı anda 5 volt olması motoru kilitleyerek fren yapmasını sağlar.

Gerekli Malzemeler
L298N çift motor sürücü kartı
DC motor
Arduino
Jumper kablo
Pil

Amaç
DC motorların hız kontrolünü öğrenmek.
DC motorların yön kontrolünü öğrenmek.

Devre şeması



Kodlar:

```
const int in1=7;    //sol motor 1. girişi
const int in2=6;    //sol motor 2. girişi
const int in3=5;    //sağ motor 1. girişi
const int in4=4;    //sağ motor 2. girişi
const int enA=9;    //sol motor pwm pini
const int enB=3;    //sağ motor pwm pini

int hiz=;180  //motorların hızını belirler 0-255 arası değer girilebilir.
void setup() {
pinMode(in1,OUTPUT);
pinMode(in2,OUTPUT);
pinMode(in3,OUTPUT);
pinMode(in4,OUTPUT);
pinMode(enA,OUTPUT);
pinMode(enB,OUTPUT);
}
```

```
void loop() {
 ileri();
delay (1000);
geri();
delay (500);
sol();
delay(500);
sag();
delay(500);
void dur(){
digitalWrite(in1,HIGH);
digitalWrite(in2,HIGH);
digitalWrite(in3,HIGH);
digitalWrite(in4,HIGH);
  analogWrite(enA,0);
  analogWrite(enB,0);
}
void sag(){
dur();
delay(1000);
}
void ileri(){
digitalWrite(in1,HIGH);
digitalWrite(in2,LOW);
digitalWrite(in3,HIGH);
digitalWrite(in4,LOW);
  analogWrite(enA,hiz);
digitalWrite(in1,HIGH);
digitalWrite(in2,LOW);
digitalWrite(in3,LOW);
digitalWrite(in4,HIGH);
  analogWrite(enA,hiz);
  analogWrite(enB,hiz);
}
void sol(){
digitalWrite(in1,LOW);
  analogWrite(enB,hiz);
}
void geri(){
digitalWrite(in1,LOW);
digitalWrite(in2,HIGH);
digitalWrite(in3,LOW);
digitalWrite(in4,HIGH);
  analogWrite(enA,hiz);
  analogWrite(enB,hiz);  }
digitalWrite(in2,HIGH);
digitalWrite(in3,HIGH);
digitalWrite(in4,LOW);
  analogWrite(enA,hiz);
  analogWrite(enB,hiz);
}
```

Kod Blokları

BÖLÜM 8- SENSÖR KAVRAMI VE ÖRNEKLER

SENSÖR NEDİR?NEDEN KULLANIRIZ?

Projelerimizi gerçekleştirirken ışık, sıcaklık, mesafe gibi fiziksel büyüklükleri elektrik sinyallerine dönüştürmek ve bu bilgileri işleyecek karar mekanizmaları kurabilmek için sensörleri kullanırız.

PEKİ SENSÖRLERİN ÇALIŞMA PRENSİBİ NEDİR?

Bu aşamada sensörleri iki tipte inceleriz:

ANALOG SENSÖRLER

DİJİTAL SENSÖRLER

ANALOG SENSÖRLER:

Analog sensörler, algıladıkları fiziksel büyüklüğe orantılı olarak değişen bir akım veya gerilim çıktısı verirler.

Peki bu sensörleri dijital bir devre kartına bağlamamız gerekirse? O zamanda analog-dijital (ADC) çeviriciler kullanırız.

DİJİTAL SENSÖRLER:

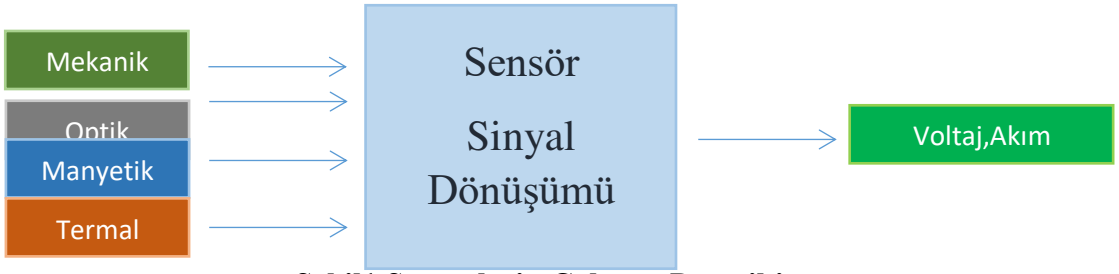
Dijital sensörler ise genellikle I2C, SPI, OneWire vb bir haberleşme protokolü aracılığıyla bilgisayar (mikroişlemci) ile konuşurlar.

Bu iki tip sensöre kısaca değindiğimize göre şimdi de aktif sensörleri ve pasif sensörleri inceleyelim.

Aktif Sensörler,kendi sinyallerini ürettikten sonra bu sinyalin ortamdaki değişimini kontrol ederek algılama işlemini gerçekleştirirler. Ultrasonik ve kızılötesi sensörler bu gruba dahildir.

Pasif Sensörler ise, ortamdan aldıkları sinyalleri kontrol ederek algılama işlemini gerçekleştirirler. LDR (ışığa duyarlı direnç), NTC/PTC (ısıya duyarlı dirençler), fototransistör (ışığa duyarlı transistör) bu gruba örnek olarak gösterilebilirler.

SENSÖR ÇEŞİTLERİ



Şekil1.Sensörlerin Çalışma Prensibi

Sensörler algılama şekillerine göre altıya ayrılırlar. Şimdi bu sensörleri ve algılama özelliklerini inceleyelim:

- Mekanik sensörler (Uzunluk, alan, miktar, kütlesel akış, kuvvet, tork, basınç, hız, ivme, pozisyon, ses dalga boyu ve yoğunluğu)
- Termal sensörler (Isı akışı ve sıcaklık)
- Elektriksel sensörler (Voltaj, akım, direnç, endüktans, kapasitans, dielektrik katsayısı, polarizasyon, elektrik alanı, frekans)
- Manyetik sensörler (Alan yoğunluğu, akı yoğunluğu, manyetik moment, geçirgenlik)
- Işıma sensörleri (Yoğunluk, dalga boyu, polarizasyon, faz, yansıtma, gönderme)
- Kimyasal sensörler (Yoğunlaşma, içerik, oksidasyon/redaksiyon, reaksiyon hızı, pH miktarı)

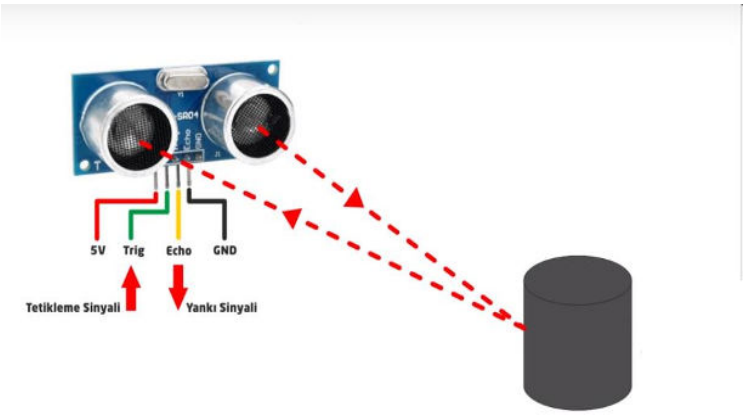
Robotlarda ve sistemlerde en yaygın kullanılan sensörleri biraz daha ayrıntılı inceleyecek olursak;

- Mesafe sensörleri (Ultrasonik, PIR, Kapasitif, Endüktif, Kızılötesi Optik…)
- Kuvvet/Ağırlık/Basınç sensörleri (
- Eğim sensörleri (Flex, Lineer/Esnek Potansiyometre…)
- Manyetik sensörler (Hall effect, reed röle…)
- Sıcaklık/Nem/Su Seviyesi sensörleri (NTC,PTC, Yağmur Sensörü…)
- Ses sensörleri (Dinamik/Kapasitif/Şeritli/Kristal/Karbon Tozlu Mikrofon)
- Işık/renk sensörleri (LDR, RGB, UV, Fototransistör, Fotodiyot…)

Yaygın olarak kullandığımız sensörlerin kullanım alanlarını gözden geçirelim;

1) **ULTRASONİK SENSÖRLER:**

Ultrasonik sensör ismini "**ultra**" ve "**sonic**" kelimelerinin birleşmesinden alır. "Daha yüksek ses" anlamına gelmektedir. Bu sensörler mesafe ölçme amaçlı kullanılmaktadırlar. Çalışma prensipleri ise şu şekildedir: Ultrasonik sensörler dışarıya bir ses dalgası sinyali gönderirler. Gönderdikleri ses dalgasının bir cisme ulaşıp kendisine geri dönmesini beklerler.Bu sensörü "Engelden Kaçan Robot " yapında kullanabiliriz.

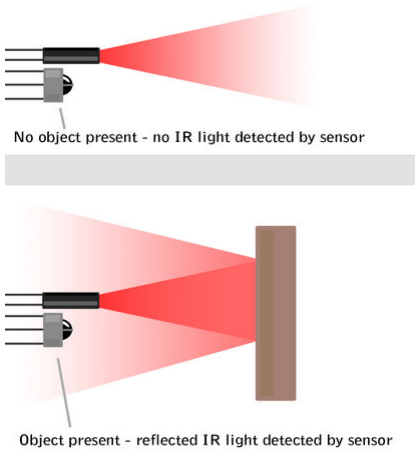


Şekil2. HC-SR04 Ultrasonik Sensörün Çalışma Prensibi

2) **INFRA-RED (Kızılötesi) SENSÖRLER:**

Kızılötesi sensörler mesafe ve karanlık/aydınlık algılama amaçlarıyla kullanılan sensörlerdir. Kızılötesi sensörlerin yapısında genellikle kızılötesi ışın yayan bir LED ve bu ışının yansımasını kontrol eden bir foto komponent bulunur (fotodiyot, fototransistör gibi). Sensörün içinde bulunan LED, kontrol etmek istediğimiz bilgi ile aynı dalga boyuna sahip bir ışın üretir.

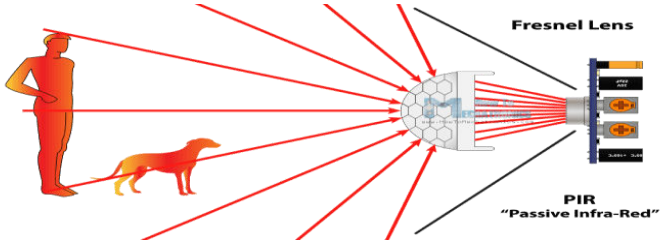
Cismin üzerine düşen ve geri yansıyan kızılötesi ışınları bir foto komponent denetler ve sensör geri dönen ışın sinyaline göre algılama işini gerçekleştirir. Bu sensörü "Çizgi İzleyen Robot" yapında kullanabiliriz.



Şekil3.Infra-Red Sensör Çalışma Prensibi

3) PASSIVE INFRA-RED (PIR) SENSÖRÜ:

PIR sensörleri ortamdaki sıcaklık ve kızılötesi dalga değişimlerine göre hareket algılayan sensörlerdir. Bu yüzden hem kızılötesi hem termal sensör mantığına sahiptir.Çalışma prensiplerine bakacak olursak; Tüm nesneler bulundukları ortama sıcaklık ve kızılötesi dalga yayarlar. PIR sensörlerin yapısında da bir fresnel lens vardır. Bu mercek sayesinde ortamdaki ışınlar sensörün tam üzerine düşüp odaklanmasını sağlar. Sensör, sabit olan sıcaklık ve kızılötesi dalgalar değiştiğinde bunu algılayarak sisteme bildirir. Bu sensörler en yaygın olarak alarm sistemlerinde ve otomatik aydınlatmalarda kullanılırlar.



Şekil4.PIR Sensörü Çalışma Prensibi

4) HALL EFFECT SENSÖR:

Hall effect sensörleri, manyetik alan algılayarak sinyal çıkışı sağlayan sensörlerdir. Bu sensörler mesafe, hız, akım algılamada ve konumlandırmada kullanılırlar.

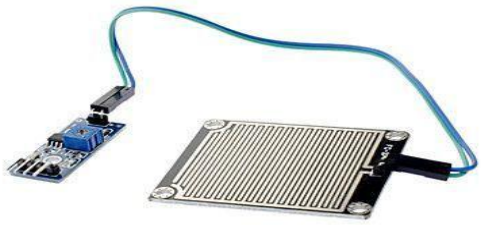
5) NTC/PTC SENSÖRLER:

NTC (Negative Temperature Coefficient) ve PTC (Positive Temperature Coefficient) ısıya duyarlı dirençlerdir. bu sensörler kendileri bir sinyal üretip geri dönüşünü beklemez, doğrudan ortamdan etkilenirler.

NTC, üzerine düşen sıcaklık arttıkça sahip olduğu direnç değeri düşer. Yani algıladığı ısı değeri ile ters orantılıdır.

PTC'nin ise üzerine düşen sıcaklık arttıkça sahip olduğu direnç değeri de artar. Yani ısı ile doğru orantılıdır.

6) **YAĞMUR SENSÖRLERİ:**Bu tip sensörler su seviyesi ölçümü, su damlası tespiti ve yağmur sensörleri olarak kullanılabilirler.



Şekil5.Yağmur Sensörü

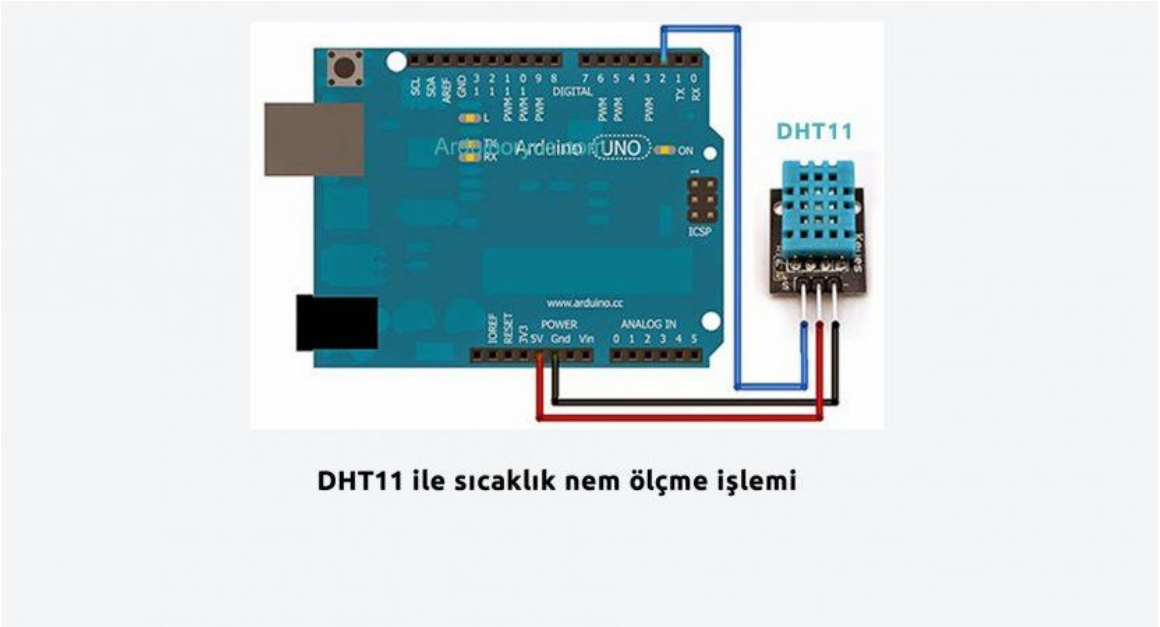
ARDUİNO İLE SENSÖR UYGULAMA ÖRNEKLERİ

Uygulama1- DHT11 ile Sıcaklık /Nem Ölçme Örneği

Gerekli Malzemeler:

- Breadboard (Devre Tahtası)
- Jumper Kablo
- LCD
- Ve Arduino

DHT11 Bağlantı Seması:



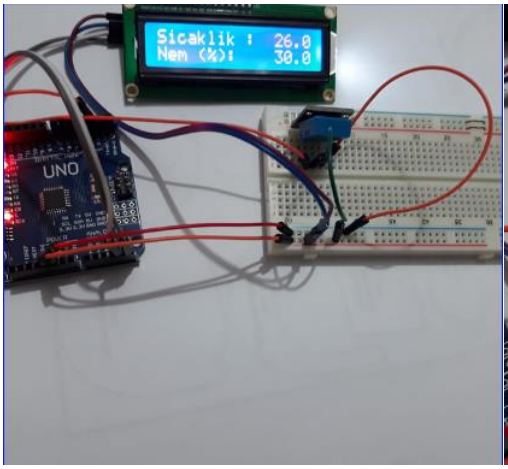
DHT11 ile sıcaklık nem ölçme işlemi

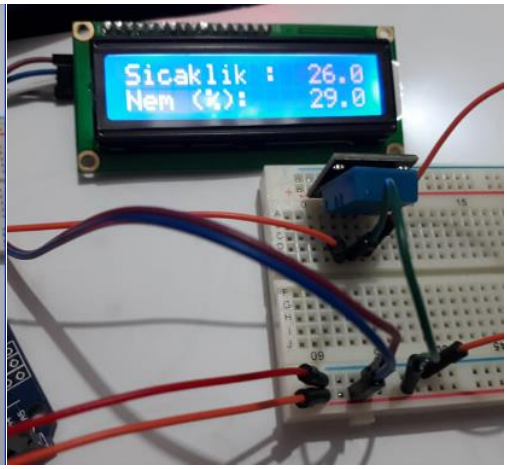
Projemizin Arduino Yazılım Aşamaları:

dht11Sicakl_kNem

```
#include <LiquidCrystal_I2C_AvrI2C.h>
LiquidCrystal_I2C_AvrI2C lcd(0x27,16,2);

#include <dht11.h> // dht11 kütüphanesini ekliyoruz.
#define DHT11PIN 2 // DHT11PIN olarak Dijital 2"yi belirliyoruz.

dht11 DHT11;

void setup()
{
  lcd.begin();
  lcd.backlight();
}

void loop()
{
  int chk = DHT11.read(DHT11PIN);//Bu değer 0 ise DHT11 çalışıyor.

  // Sensörden gelen verileri serial monitörde yazdırıyoruz.
  lcd.setCursor(0,0);
  lcd.print("Sicaklik : ");
  lcd.setCursor(12,0);
  lcd.print((float)DHT11.temperature, 2);

  lcd.setCursor(0,1);
  lcd.print("Nem (%): ");
  lcd.setCursor(12,1);
  lcd.print((float)DHT11.humidity, 2);

  delay(2000);
}
```

Ölçüm Sonuçları:





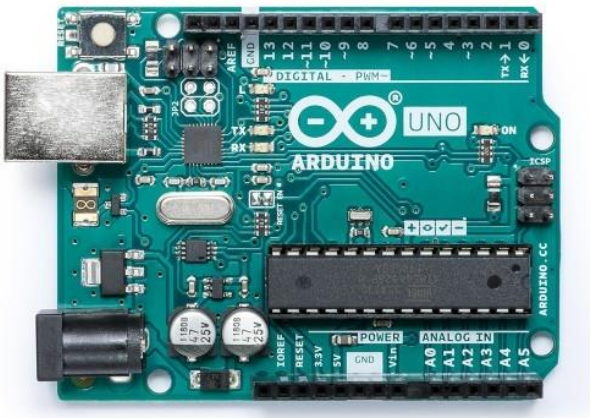
BÖLÜM 9- BASİT ROBOT UYGULAMASI

Engelden Kaçan Robot

Engelden kaçan robot, diğer adıyla engel algılayan robot, otonom olarak çevre kontrolü yapabilen ve hareketini önleyebilecek cisimleri atlatabilen robot tipidir. Çevre kontrolünü sağlayabilmesi için ultrasonik, kızılötesi vb. gibi çeşitli sensörlere ihtiyaç duyar. Yazılımınıza göre kendi yolunu belirleyerek engellere çarpmadan yoluna devam eder. Hc-sr04 ya da Mz80 gibi sensörler bu proje için idealdir. Projede hc-sr04 kullanılacaktır.

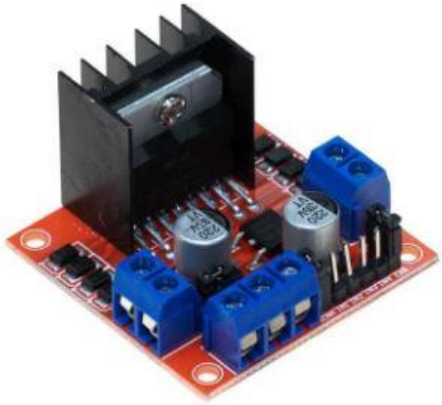
Gerekli Malzemeler:

- Arduino Uno



- Bir adet 2WD araç kiti

- L298N Voltaj Regulatörlü Çift Motor Sürücü Kartı



- HC-SR04 Ultrasonik Mesafe Sensörü

- 6'lı AA Pil Yuvası ve pilleri

- Jumper kablo

Ultrasonik Mesafe Sensörü Nedir Ne İşe Yarar?

Hc-sr04 Ultrasonik sensör sonar (Sound Navigation and Ranging) iletişim kullanarak karşısındaki nesneye olan mesafeyi hesaplayan bir kaynaktır.Sonar dediğimiz sistem ses dalgalarını kullanarak cismin uzaklığını hesaplamamıza yardımcı olur. Bu tür sensörlerin esin kaynağı yunuslar ve yarasalardır. Yunuslar ve yarasalarda ses dalgası göndererek karşısına çıkabilecek engellerin mesafelerini hesaplayabilmektedirler.

Bu sensör elektronik/robotik malzeme satan mağazalarda kolaylıkla bulunabilir. Sensör üzerinde giriş ve çıkış olmak üzere iki yüzey bulunmaktadır. Çıkış yüzeyinden ortama belirli bir frekansta ultrasonik ses dalgası salınır. Giriş yüzeyi de çıkış yüzeyinin ortama saldığı belirli frekanslardaki ses dalgalarını toplar. Uzaklık ölçümü için öncelikle çıkış yüzeyinden ortama ses dalgası salınır. Salınan ses dalgası 15 derece açıyla ortamda yayılır. Yayılan ses dalgası bu alanda bulunan bir cisme çarptığında, cisim yüzeyinden sensöre geri yansır. Yansıyan dalganın giriş yüzeyine gelmesiyle işlem tamamlanır. Dalganın çıkış yüzeyinden çıkmasıyla giriş yüzeyine ulaşması arasında geçen süre ölçülerek, cismin uzaklığı hesaplanır. Bu basit mantıkla çalışan sensör, 2 cm ile 200 cm arasındaki uzaklıkları 1 cm hassasiyetle ölçebilmektedir.

Sensör bu aralık dışındaki uzaklıkları istikrarlı olarak ölçememektedir.

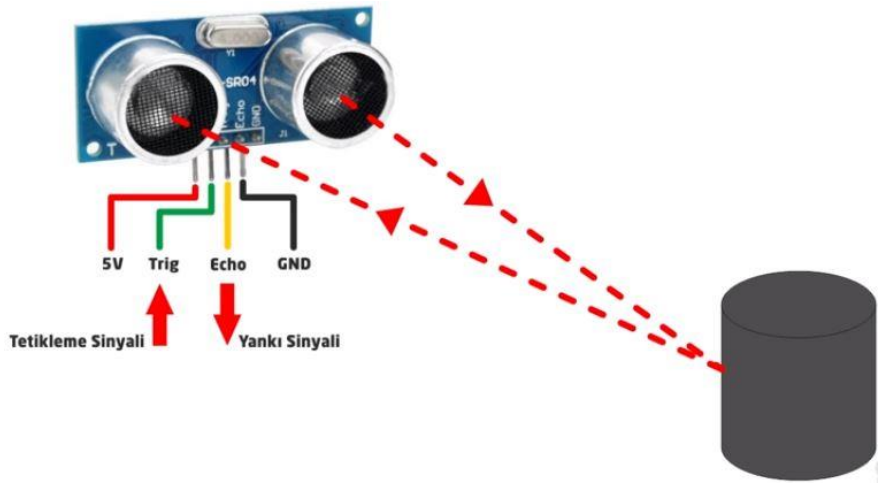
Hc-sr04 sensörümüzün 4 adet bacağı bulunmaktadır, bunlar:

Vcc = 5v kaynağı.Gnd = Topraklama bacağı.

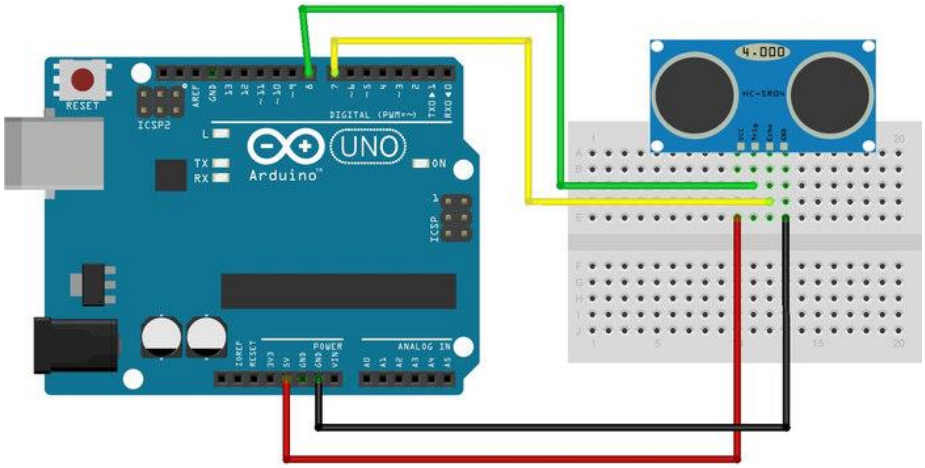
Trig = Sensörün ses dalgası gönderen kısmı.

Echo = Gönderilen ses dalgasını alan kısmı.

Hc-sr04 sensörümüz 5 volt ile çalışmaktadır. En verimli ölçüm yaptığı mesafe 2-200 cm arasındadır. 200 cm'den fazla mesafelerde verimli bir şekilde ölçüm yapmamaktadır.



HC-SR 04 Test Kodu ve Bağlantı Şeması

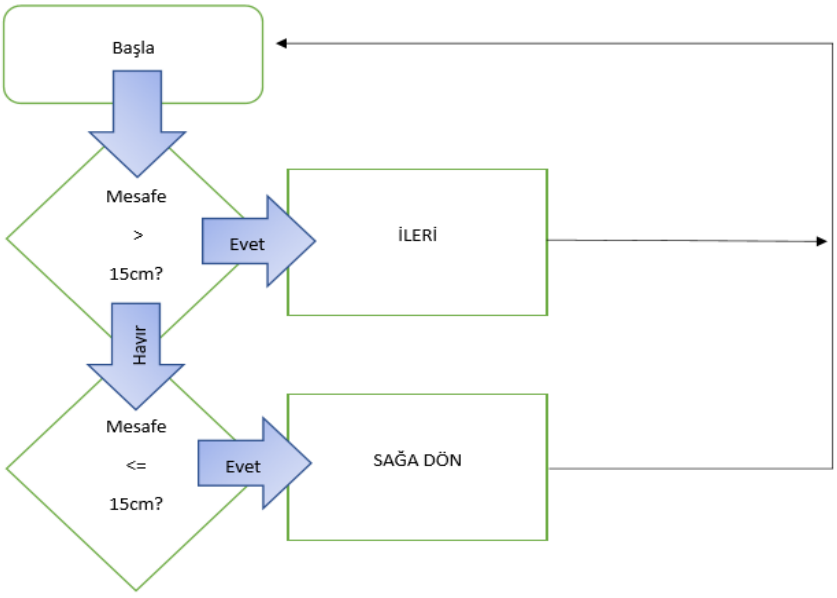



```
int trigPin = 6;    //Sensorun trig pini Arduinonun 6 numaralı ayağına bağlandı.
int echoPin = 7;  // Sensorun echo pini Arduinonun 7 numaralı ayağına bağlandı.

long sure;
long uzaklik;

void setup(){
  pinMode(trigPin, OUTPUT); // trig pini çıkış olarak ayarlandı.
  pinMode(echoPin,INPUT);    // echo pini giriş olarak ayarlandı.
  Serial.begin(9600);                 // Seri haberlesme başlatıldı.
}
void loop()
{
```

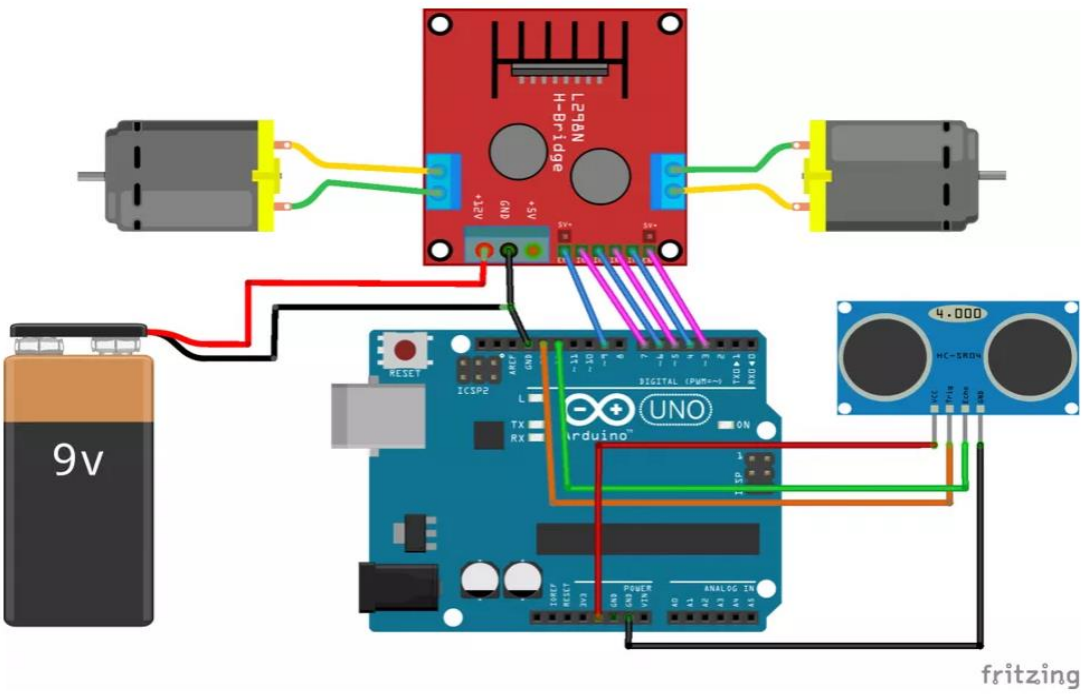
```
  digitalWrite(trigPin, LOW); // sensör pasif hale getirildi.
  delayMicroseconds(5);
  digitalWrite(trigPin, HIGH); // Sensore ses dalgasının üretmesi için emir verildi.
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);  /* Yeni dalgaların üretilmemesi için trig pini LOW konumuna getirildi */
  sure = pulseIn(echoPin, HIGH); // ses dalgasının geri dönmesi için geçen sure ölçülüyor.
  uzaklik= sure /29.1/2; //ölçülen sure uzaklığa çevriliyor.

  if(uzaklik > 200)
    uzaklik = 200;
  Serial.print("Uzaklik ");
  Serial.print(uzaklik); // hesaplanan uzaklık bilgisayara aktarılıyor.
  Serial.println(" CM olarak olculmustur.");
  delay(500);
}
```

Engelden Kaçan Robotun Algoritması:

Robotumuzu kodlamaya başlamadan önce hangi işlemleri takip edeceğimizi bilmemiz gerekmektedir. Temel amacımız robotun bir engele takılmaması olacaktır. Örnek olarak geniş bir zeminde engelle karşılaştığında sağ yöne hamle yapan bir robotun temel algoritması şekilde gösterilmiştir.



Engelden Kaçan Robotun Bağlantı Şeması:



Gerekli Kod Bloğu:

```
#define echo 12
#define trig 13
#define sol 7
#define sol1 6
#define e1 9
#define sag 5
#define sag1 4
#define e2 3

int  sure = 0;
int mesafe = 0;

void setup()
{
  pinMode(trig , OUTPUT);
  pinMode(echo , INPUT );
  pinMode(sol, OUTPUT);
  pinMode(sol1, OUTPUT);
  pinMode(sag, OUTPUT);
  pinMode(sag1, OUTPUT);
}

void loop()
{
  digitalWrite(trig, LOW);
  delayMicroseconds(5);
  digitalWrite(trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig, LOW);

  sure = pulseIn(echo, HIGH);
  mesafe = sure / 29.1 / 2;
  Serial.println(mesafe);

  if (mesafe < 20 )
  {
```

```
    digitalWrite(sol , LOW);
    digitalWrite(sol1 , HIGH);
    digitalWrite(sag , LOW);
    digitalWrite(sag1 , HIGH);
    delay(150);
    digitalWrite(sol , LOW);
    digitalWrite(sol1 , HIGH);
    digitalWrite(sag , HIGH);
    digitalWrite(sag1 , LOW);
    delay(250);
  }
  else
  {
    digitalWrite(sol , HIGH);
    digitalWrite(sol1 , LOW);
    digitalWrite(sag , HIGH);
    digitalWrite(sag1 , LOW);
  }
}
```

Kod Bloğunun açıklanması:

long sure;

long uzaklık;

Burada öncelikle bizim ses dalgası gönderen ve ses dalgasını alan pinlerimizin Arduino kartımız üzerindeki pinlerimizi belirliyoruz. Daha sonra süre ve uzaklık adında iki tane değişken atıyoruz.

pinMode(trigPin, OUTPUT);

pinMode(echoPin,INPUT);

Serial.begin(9600);

Burada ise trig pinimizi OUTPUT olarak belirliyoruz. Bunun sebebi ise ses dalgasını gönderen kısmımız trig pinimiz olmasından dolayı. Echo pinimiz ise gönderilen ses dalgasını aldığı için INPUT olarak belirtiyoruz ve son olarak Serial.begin ile de seri haberleşmemizi başlatıyoruz.

digitalWrite(trigPin, LOW);İlk olarak trip pinimizi low durumunda başlatıyoruz.

delayMicroseconds(5); 5 Mikrosaniye(saniyenin milyonda biri) beklemesini belirtiyoruz.

digitalWrite(trigPin, HIGH); Daha sonra pinimizin ses dalgası göndermesi için emir veriyoruz.

delayMicroseconds(10);

digitalWrite(trigPin, LOW); Yeni ses dalgası üretebilmesi için trig pinimizi pasif durumuna getiriyoruz

sure = pulseIn(echoPin, HIGH); Gönderilen ses dalgasının geri dönmesindeki süre ölçülüyor.

uzaklik= sure /29.1/2;Ölçütüğümüz süre uzaklığa çevriliyor.

if(uzaklik > 200) if komutu ile 200 cm ve üzeri bütün uzaklıklar 200 cm

Serial.print(“Uzaklik “);

Serial.print(uzaklik); Ölçtüğümüz uzaklığımız bilgisayarımıza yani Arduino programımızda Araçlar>Seri Port Ekranı kısmına yazılıyor.

Serial.println(” CM “);

delay(100); }

KAYNAKÇA

1) https://maker.robotistan.com/arduino-yazilim-kurulum/

2)Hasbi SEVİNÇ, Temel Elektronik Arduino Eğitimi

3)Aslı Ergün, Arduino İle Programlama

4)https://maker.robotistan.com/arduino-dersleri-10-16x2-lcd-ekran/

5) http://www.teknoyolcu.com/2018/09/ders-2-arduino-bilesenleri/

6) https://www.projehocam.com/arduino-tinkercad-kullanarak-cabucak-baslayin/

7) https://tinkercadilearduino.blogspot.com/

8) http://www.tinkercad.com/

9)https://www.arduino.cc/en/Tutorial/AnalogInputPins

10)https://www.arduino.cc/reference/en/language/functions/analog-io/analogread

11)https://www.arduino.cc/reference/en/language/functions/analog-io/analogwrite

12)https://www.arduinomedia.com/arduino-ile-7-segment-display-led-kullanimi/

13)https://www.elektrikport.com/makale-detay/7-segment-display-nedir/18475#ad-image-0

14)http://www.robotiksistem.com/lcd_yapisi_calismasi.html

15) https://maker.robotistan.com/robot-kontrolculeri-sensorler/#Sensor-Nedir-Ne-Demek-Ne-Ise-Yarar

16)https://www.arduinoryus.com/

17)https://gelecegiyazanlar.turkcell.com.tr/konu/arduino/egitim/arduino-301/arduino-ile-uzaklik-olcumu

18)https://maker.robotistan.com/engelden-kacan-robot-yapimi/

19)https://www.motorobit.com/blog/icerik/engelden-kacan-robot-yapimi

20)https://hayaletveyap.com/arduino-ile-engelden-kacan-robot/

21)https://lezzetlirobottarifleri.com/arduino-ile-engelden-kacan-robot-v1-00-1-bolum/

22)http://roboturka.com/arduino/engelden-kacan-robot-yapimi/

23)https://www.mobilhanem.com/arduino-setup-loop-fonksiyonlari-ve-merhaba-dunya/

24)https://forum.donanimhaber.com/arduino-temel-komutlar--108616408

25)https://koddefteri.net/arduino/temel-arduino-dersleri/arduino-donguler.html

26)https://www.algoritmaornekleri.com/arduino/arduino-for-dongusu/

27)https://gelecegiyazanlar.turkcell.com.tr/konu/arduino/egitim/arduino-101/donguler

28) https://koddefteri.net/arduino/temel-arduino-dersleri/arduino-sozdizimi-syntax.html

Bu kitap 2019-2020 Arduino ile Robotik Kodlama Etwinning Projesi için proje ortaklarımızla birlikte hazırlanmıştır.