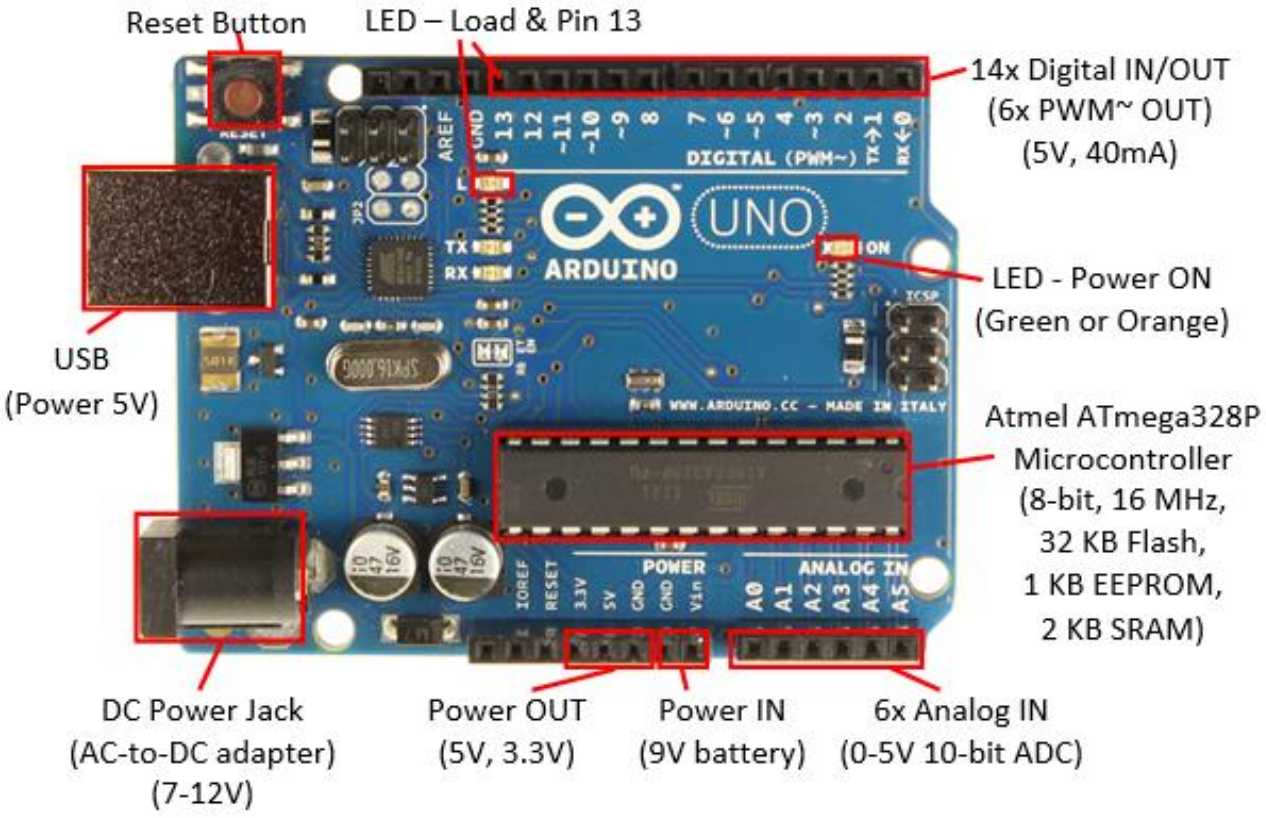
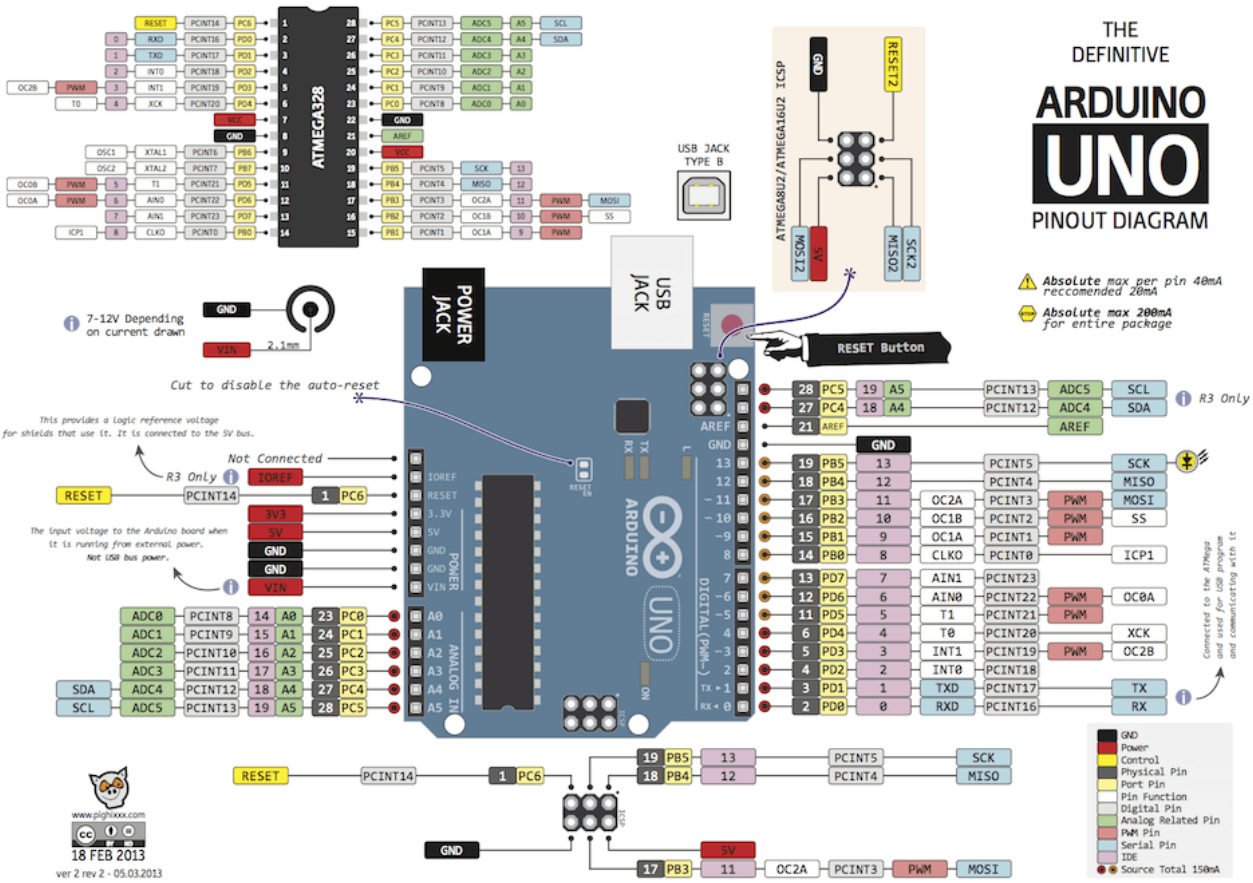
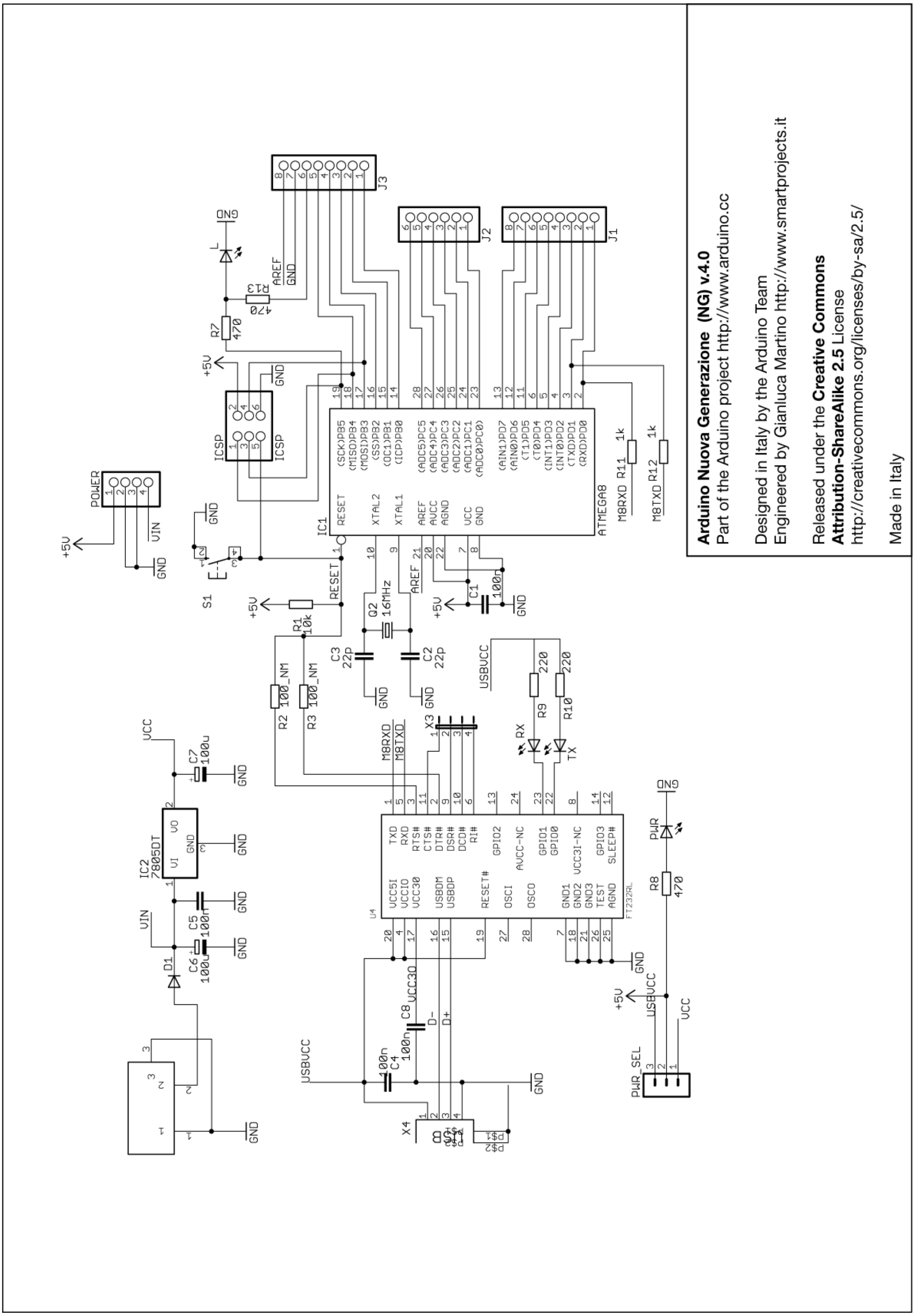
ARDUİNO UYGULAMALARI (İLERİ SEVİYE) KURSU

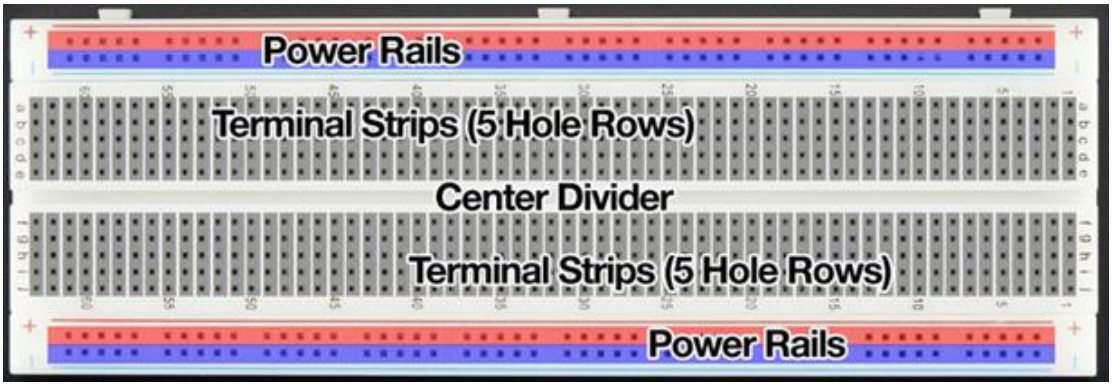
ARDUINO UNO

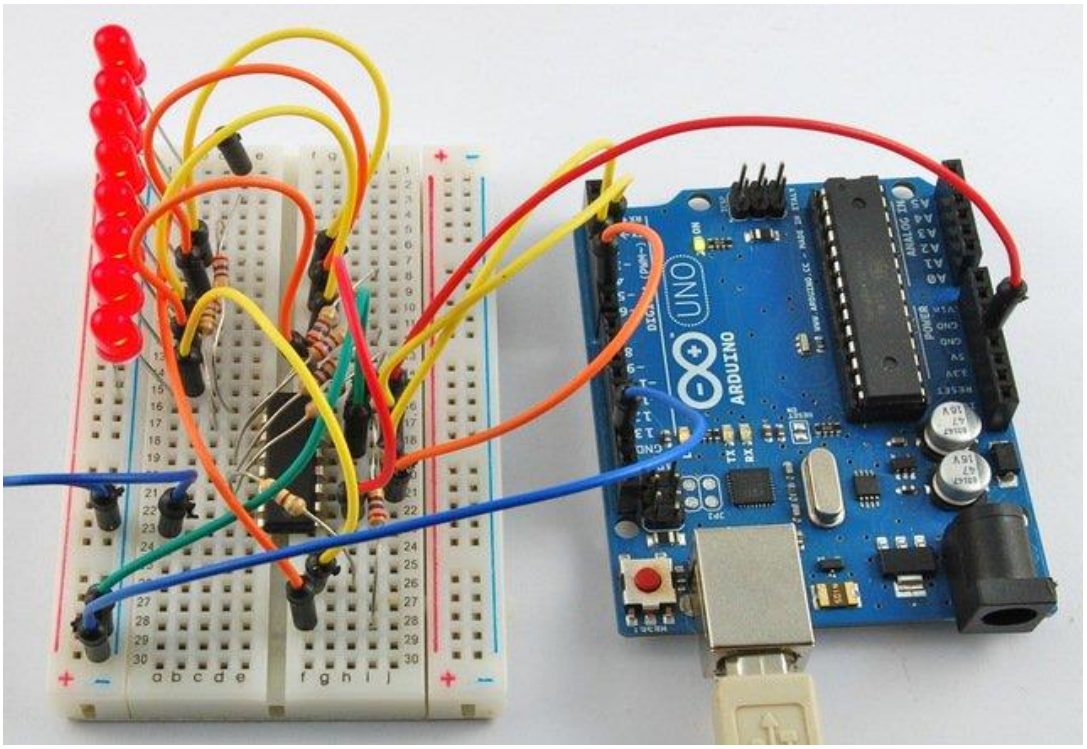


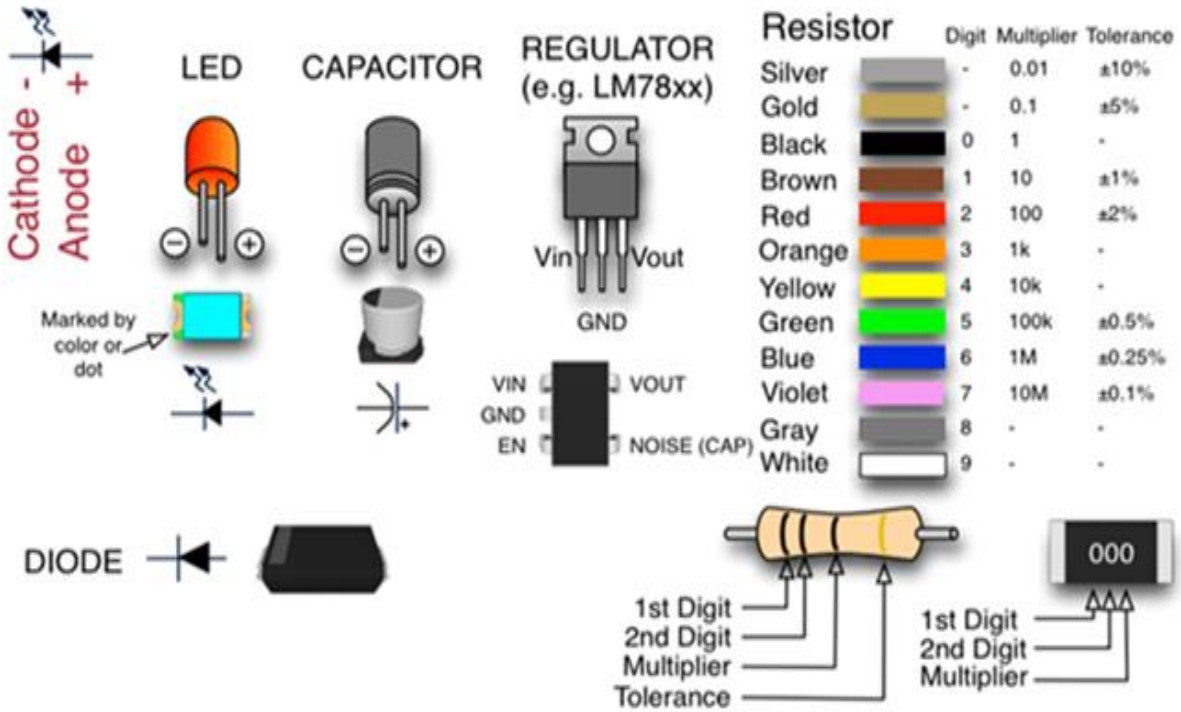



Arduino Nuova Generazione (NG) v.4.0
Part of the Arduino project http://www.arduino.cc
Designed in Italy by the Arduino Team
Engineered by Gianluca Martino http://www.smartprojects.it
Released under the Creative Commons
Attribution-ShareAlike 2.5 License
http://creativecommons.org/licenses/by-sa/2.5/
Made in Italy

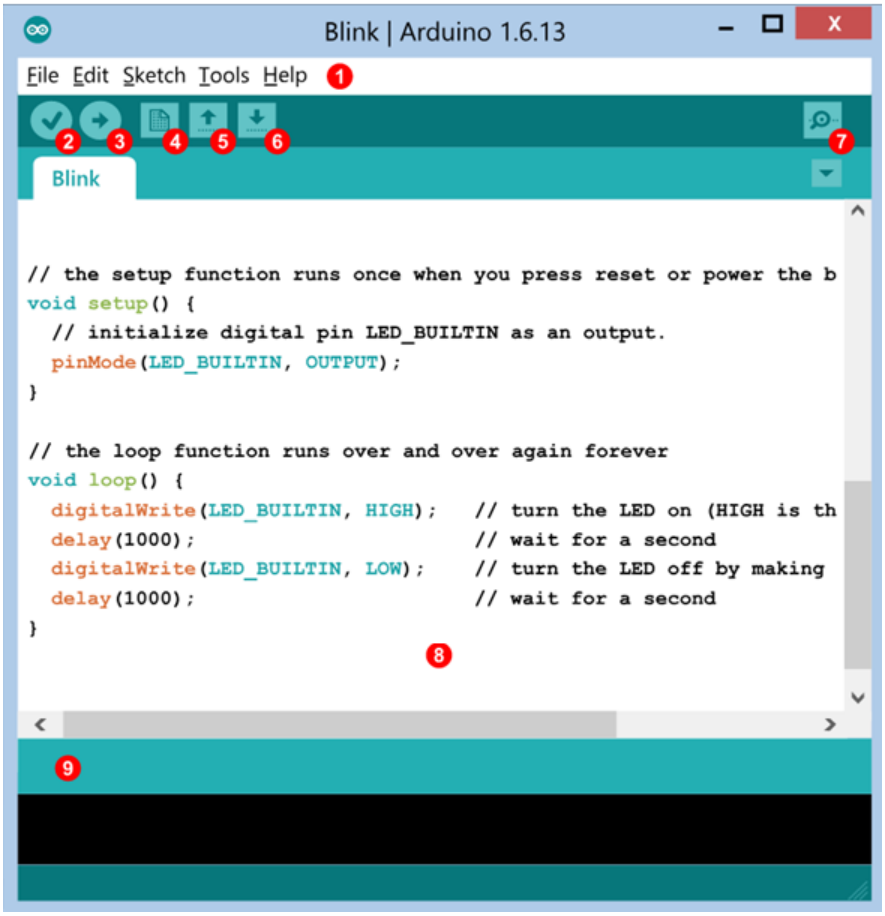
BREADBOARD YAPISI VE DEVRE ELAMANLARI







ARDUINO IDE



1) Menü satırı
2) Yazılan programı derler.
3) Yükle : Programı karta yükler.
4) Yeni dosya açar.
5) Kaydedilmiş dosyaları açar.
6) Programı kaydeder.
7) Seri haberleşme penceresini açar.
8) Program yazım alanı.
9) Mesaj alanı.

Seri İletişim ve Komutları;

Arduino kartı ile bilgisayar veya diğer cihazlar arasındaki iletişim için kullanılır. Tüm Arduino kartlarının en az bir seri portu vardır (UART veya USART olarak da bilinir): Seri. Dijital pin 0 (RX) ve 1 (TX) ile bilgisayar üzerinden USB üzerinden haberleşir. Bu nedenle, bu işlevleri kullanırsanız, dijital giriş veya çıkış için 0 ve 1 pimlerini de kullanamazsınız.

Arduino ortamındaki yerleşik seri monitörü Arduino kartıyla iletişim kurmak için kullanabilirsiniz. Araç çubuğundaki seri monitör düğmesine tıklayın ve başlamak için kullanılan aynı baud hızını seçin.

TX / RX pinleri üzerindeki seri iletişim TTL mantık seviyelerini kullanır (panele bağlı olarak 5V veya 3,3V). Bu pinleri doğrudan bir RS232 seri portuna bağlamayın; (+/- 12V olarak çalışırlar ve Arduino kartınıza zarar verebilirler.)

Serial.begin ()

Seri veri iletimi için veri hızını saniyede bit (baud) olarak ayarlar. Bilgisayarla iletişim kurmak için şu hızlardan birini kullanır; 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 veya 115200. 0 ve 1 pimleri üzerinde iletişim kurabilir.

Serial.print ()

Verileri ASCII olarak seri bağlantı noktasına yazdırır. Rakamları her rakam için bir ASCII karakteri kullanılarak yazdırılır. Baytlar tek bir karakter olarak gönderilir.

Serial.available ()

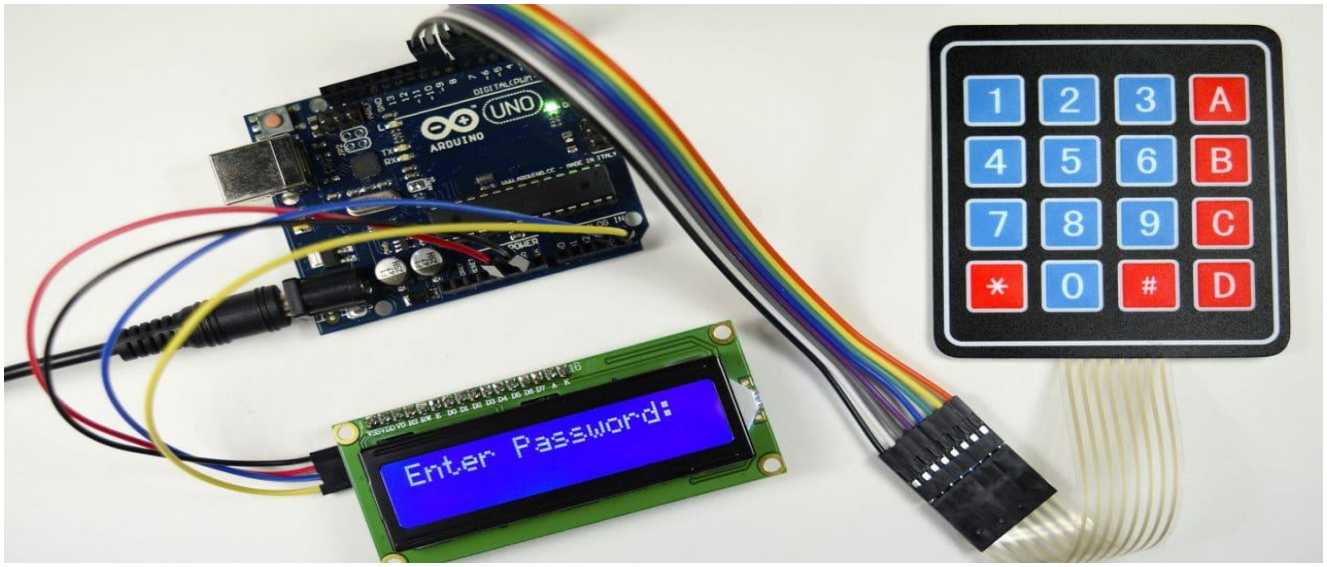
Seri bağlantı noktasından mevcut bayt sayısını (karakterleri) alır. Seri alma arabelleğinde (64 bayt max.) saklanan verileri sayar.

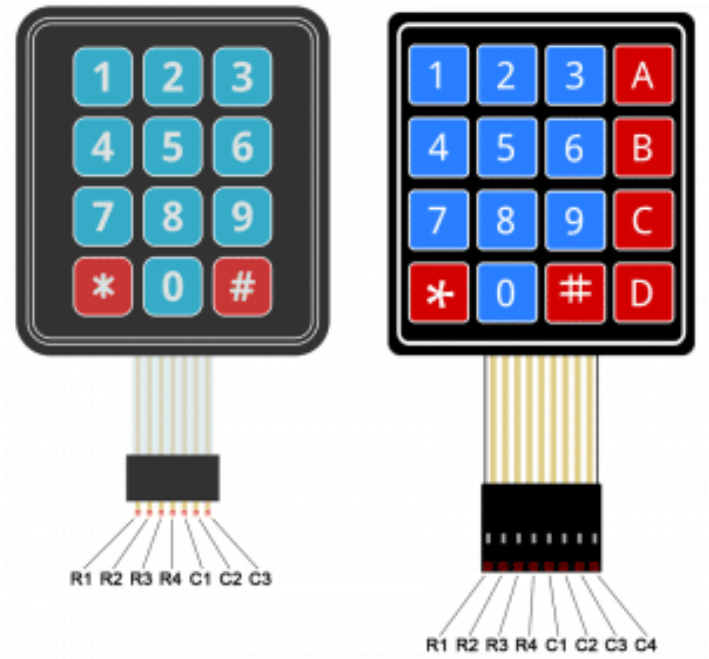
Gelen seri stream.parseInt () Akış yardımcı sınıfı sınıfından gelen bir sonraki geçerli tamsayıya bakar.

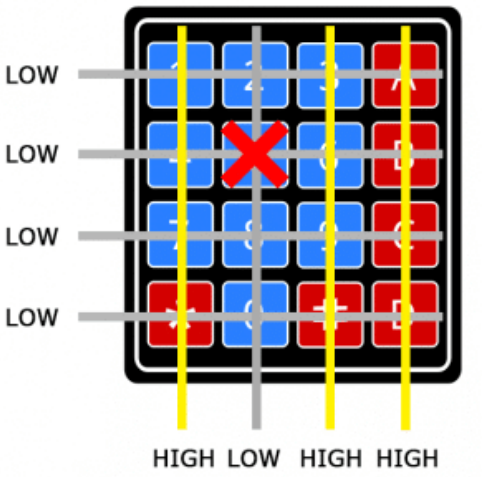
Serial.parseInt ()

Alınan bilgilerin sayısal kısmına kadar olan bölümünü alır.

Flex Keypad ve LCD Ekran kullanımı ve uygulama örnekleri



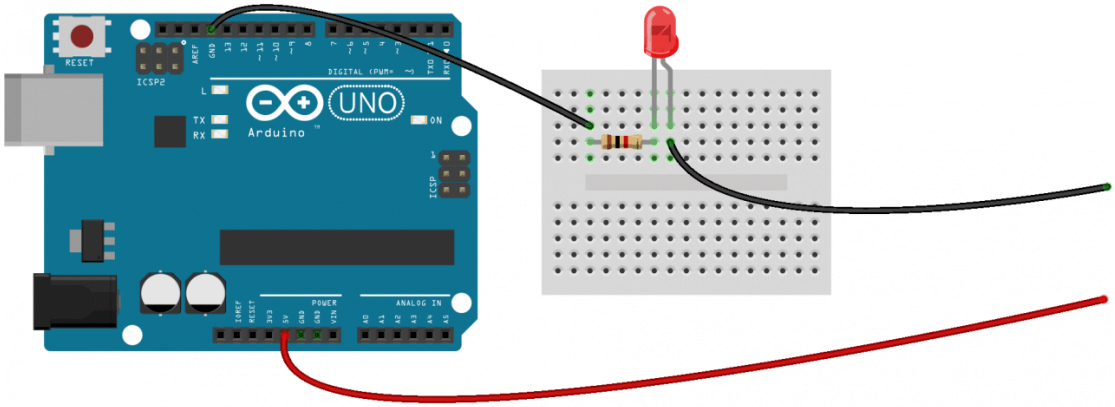


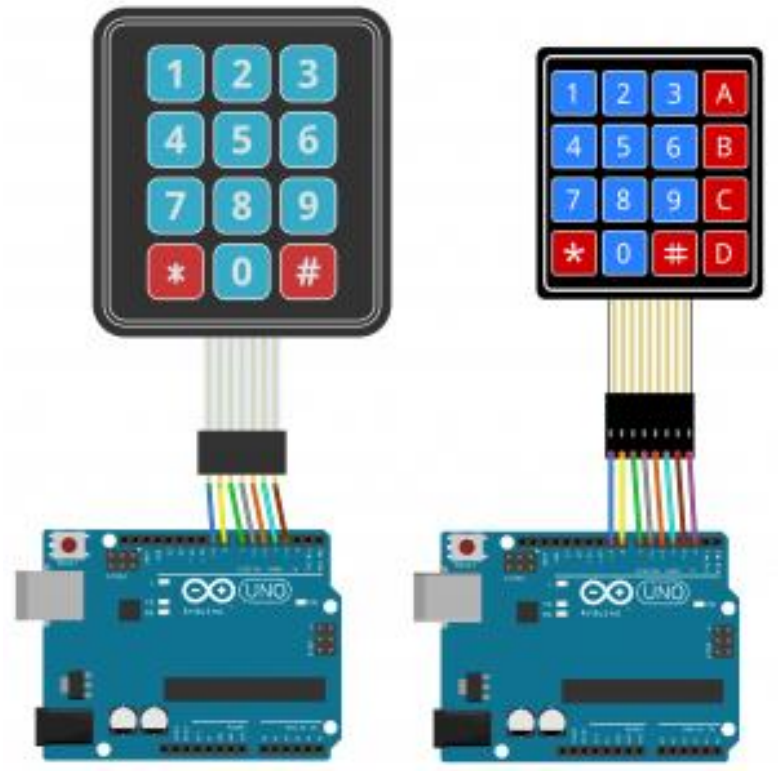


KEYPAD'IN PINLERİNİ BULMA

Tuş takımınızın pin düzeni yukarıdakilerle eşleşmiyorsa, Arduino'ya (veya herhangi bir 5V güç kaynağına) bir LED ve bir akım sınırlayıcı rezistansı bağlayarak bir test devresi oluşturmanız gerekir. Öncelikle kabloyu soldaki ilk pime takın, 1. satırdaki herhangi bir düğmeye basın ve basılı tutun. Şimdi kabloyu diğer pinlerin her birine takın. LED pimlerden birinde yanarsa, 1. satırda başka bir tuşa basın ve basılı tutun, ardından pozitif kabloyu diğer pinlerin her birine tekrar takın. LED'in farklı bir pimle yanması durumunda, GND kablosu 1. sıraya yerleştirilir. 1. satırdaki düğmelerden hiçbiri LED'i yanmazsa, GND kablosu 1. sıraya bağlı değildir.

Şimdi GND telini bir sonraki pime hareket ettirin, farklı bir sıradaki düğmeye basın ve yukarıdaki işlemi tekrarlayın. her satır için pimi bulun. Sütunların hangi pimlerine bağlı olduğunu anlamak için, GND telini 1. satır olduğunu bildiğiniz pime yerleştirin. Şimdi, bu satırdaki düğmelerden herhangi birine basın ve basılı tutun. Şimdi artı telleri kalan pimlerin birine takın. LED'i aydınlatan pim, bu düğmenin sütuna bağlı olan pimdir. Şimdi aynı sıradaki başka bir düğmeye basın ve pozitif kabloyu diğer pinlerin her birine takın. Her biri birbirinden ayrılana kadar diğer sütunların her biri için bu işlemi tekrarlayın.





Tuş takımının satır (R) ve sütün (C) numaralarını bulduktan sonra bağlantı yaptığımız dijital pinlerini satır ve sütün olarak programa aşağıdaki gibi yazıyoruz.

```
byte rowPins[ROWS] = { 8, 7, 6, 5};
byte colPins[COLS] = { 4, 3, 2};
```

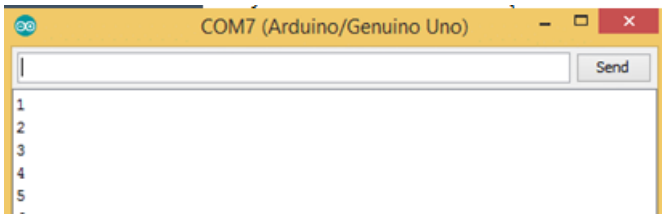
Keypad.h dosyasını kütüphaneye kurduktan sonra aşağıdaki tanımlamaları programımızın programımıza ekliyoruz. Kullandığımız ölçülere göre başına yazmamız gerekiyor.

```
#include <Keypad.h>

const byte ROWS = 4;
const byte COLS = 3;

char hexaKeys[ROWS][COLS] = {
  {'1', '2', '3'},
  {'4', '5', '6'},
  {'7', '8', '9'},
  {'*', '0', '#'}
```

```
#include <Keypad.h>

const byte ROWS = 4;
const byte COLS = 4;

char hexaKeys[ROWS][COLS] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
```

keypad_serial §

```
#include <Keypad.h>
const byte ROWS = 4; // dört satır
const byte COLS = 3; // üç kolon
//tuş takımındaki tuşların yerleşimi iki boyutlu
//'Keys' değişkenine tanımlanıyor
char Keys[ROWS][COLS] = {
  {'1','2','3'},
  {'4','5','6',},
  {'7','8','9',},
  {'*','0','#',}        };
byte rowPins[ROWS] = { 8, 7, 6, 5};
byte colPins[COLS] = { 4, 3, 2};
Keypad customKeypad = Keypad( makeKeymap(Keys), rowPins, colPins, ROWS, COLS);

void setup(){
  Serial.begin(9600);
}
void loop(){
  char customKey = customKeypad.getKey();
  if (customKey){
    Serial.println(customKey);
  }
}
```

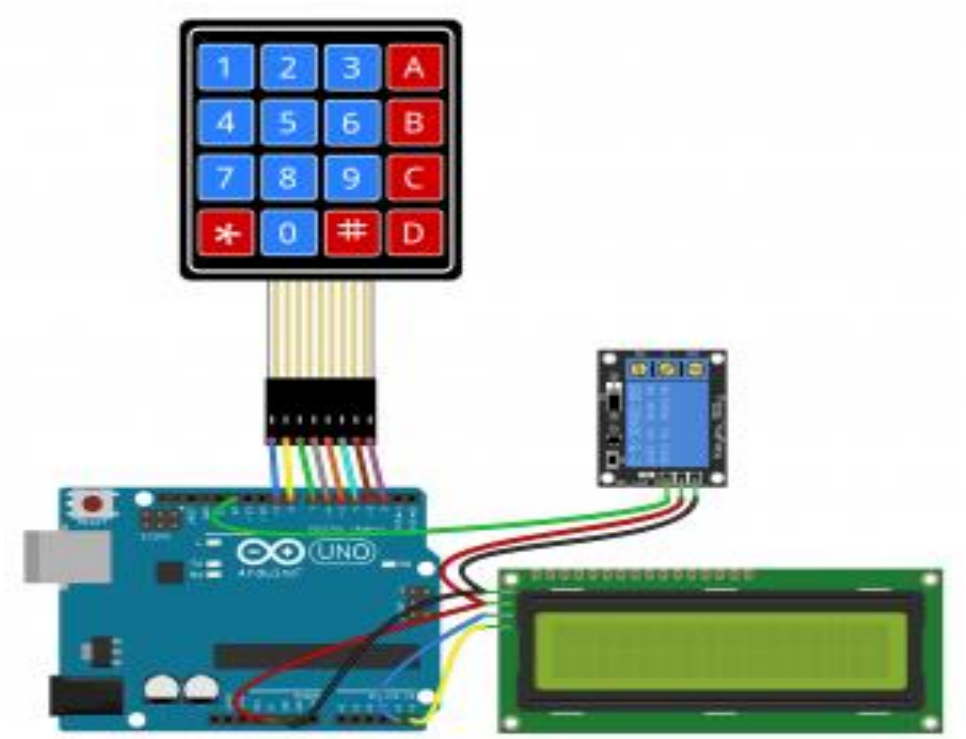


Programı yazıp yüklediğimizde seri haberleşme penceresini açıp tuşlara sıra ile basalım , tuş değerini ekranda görelim.

Tuş takımı ile seri LCD bağlantısını aşağıdaki yaptıktan sonra örnek prgramı yazıp bağlantımızı test edelim. Daha sonra şifre girişi ile bir röleyi açmak için gerekli programı *yazalım.*

keypad_lcd §

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <Keypad.h>
const byte ROWS = 4;
const byte COLS = 3;
char hexaKeys[ROWS][COLS] = {
  {'1','2','3' },
  {'4','5','6',},
  {'7','8','9',},
  {'*','0','#',} };
byte rowPins[ROWS] = {8, 7, 6, 5};
byte colPins[COLS] = {4, 3, 2};
Keypad customKeypad = Keypad(makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);
LiquidCrystal_I2C lcd(0x27, 16, 2);
void setup(){
  lcd.begin();
  lcd.clear();
}
void loop(){
  char customKey = customKeypad.getKey();
  if (customKey){
    //lcd.clear();
    //lcd.setCursor(0, 0);
    lcd.print(customKey);
  }
}
```



keypad_pas §

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <Keypad.h>
#define Password_Length 8
int signalPin = 12;
char Data[Password_Length];
char Master[Password_Length] = "012*34#";
byte data_count = 0, master_count = 0;
bool Pass_is_good;
char customKey;
const byte ROWS = 4;
const byte COLS = 3;
char hexaKeys[ROWS][COLS] = {
  {'1','2','3' },
  {'4','5','6',},
  {'7','8','9',},
  {'*','0','#',} };
byte rowPins[ROWS] = {8, 7, 6, 5};
byte colPins[COLS] = {4, 3, 2};
Keypad customKeypad = Keypad(makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);
LiquidCrystal_I2C lcd(0x27, 16, 2);
void setup(){
  lcd.begin();
  lcd.clear();
  pinMode(signalPin, OUTPUT);
  digitalWrite(12,HIGH);
}
```

```
28 void loop(){
29   lcd.setCursor(0,0);
30   lcd.print("Enter Password:");
31   customKey = customKeypad.getKey();
32   if (customKey){
33     Data[data_count] = customKey;
34     lcd.setCursor(data_count+6,1);
35     lcd.print("*");
36     data_count++;
37     }
38   if(data_count == Password_Length-1){
39     delay(500);
40     lcd.clear();
41     if(!strcmp(Data, Master)){
42       lcd.print("Correct");
43       digitalWrite(signalPin, LOW);
44       delay(5000);
45       digitalWrite(signalPin, HIGH);
46       }
47     else{
48       lcd.print("Incorrect");
49       delay(1000);
50       }
51     lcd.clear();
52     clearData();
53   }
54 }
55 void clearData(){
56   while(data_count !=0){
57     Data[data_count--] = 0;
58   }
59   return;
60 }
```

http://www.circuitbasics.com/how-to-set-up-a-keypad-on-an-arduino/

Lehim yapma temel becerileri eğitimi ve uygulamaları

Lehimleme erime noktaları düşük metalleri tutturma işlemlerinde kullanılan, kalay ve kurşun alaşımlarının genel adıdır. Buradan anlaşılacağı gibi metalleri birbirine tutturacağız. Peki, bunu yaparken neleri kullanacağız? **Havya** ve **lehim teli**.

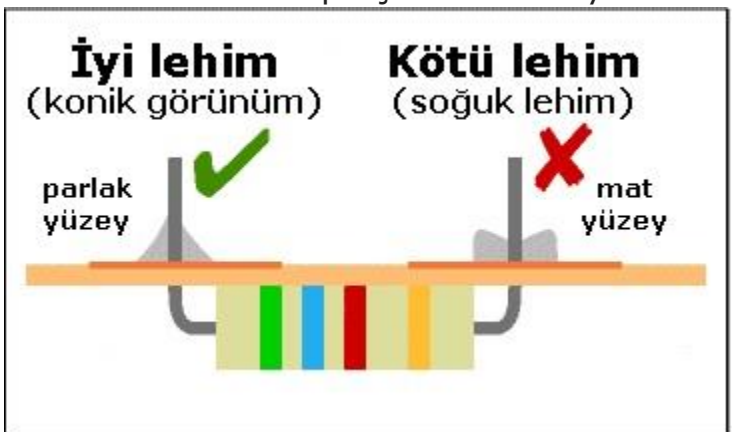
Havyanın metal kısmı yüksek derecede sıcaklığa sahip olduğundan elinizi oraya değdirmemeye dikkat etmelisiniz. Havyadan meydana gelen yanıklarda elinizi basınçsız soğuk su altında tutmanız ve daha büyük bir yanıkta ise doktora gitmeniz sizin için yararlı olacaktır.

Lehim yapılacak ortamda hava dönüşümü olması gerekmektedir. Lehim sırasında açığa çıkan gaz asit içerir ve sağlığınız için tehdit oluşturabilir. Bu yüzden kesinlikle solunmamalıdır ve maske kullanılmalıdır.

Lehimleme bittiğinde ellerinizi yıkamaya özen göstermelisiniz. Çünkü deri üzerine lehimleme sırasında çıkan gazlar ya da zehirli bir madde olan kurşun yapışmış olabilir. Deri yolu ile ya da yutarak size zarar vermesine ön verebilirsiniz.

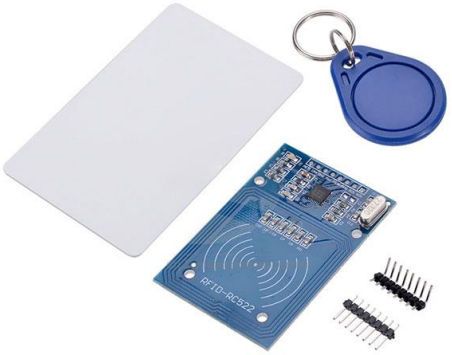
Öncelikle havyayı hazırlamamız gerekir. Havya fişe takıldıktan yaklaşık olarak 5-10 dk. sonra hazır olacaktır. Hazır olduğunu gözlemlemek için lehim telini değdirirsek birden bire sıvı olduğu görülecektir. Havyanın ucu nemli bir sünger ile temizlenmelidir. (Fakat fazla ıslak olmamasına **dikkat edilmelidir**. Çünkü, böyle bir durum süngerin her temasında havyanın sıcaklığını değiştirecektir.) Havyanın ucunda kullanıma bağlı olarak oksitlenmiş kurşun ve reçine tortuları birikmiş olur. Bu da ısı iletimini engellemiş olur ya da temassızlık meydana getirir. Havyanın ucuna biraz lehim eritilir bir başka değişle kalaylama işlemi yapılır ve havyanın ucunun her temizlenişinde bu işlemin yapılması gerekir. Artık havya ısındı ve hazır bir şekilde onu yönetmenizi bekliyor. Havyayı kullanırken metal kısma değmemeye dikkat etmeyi **unutmamalısınız**. Sanki elinizde kalem varmış ve yazı yazacakmışsınız gibi düşünün.

Lehimlenecek parça ve zemin yeteri miktarda ısıtılmalı ve unutulmamalı ki lehim teli havya ucuna değil ısınmış parçalara değdiği zaman erimesi gerekmektedir. Parçalar yeterli sıcaklığa ulaştıysa lehim eriyerek baskı devrede şeklini alır. Havya ucu lehim üzerinde kalırsa lehim oksitlenmeye başlayıp matlaşır ve meydana soğuk lehim dediğimiz olay çıkar. Önce lehim teli daha sonrada havya ucunu parçaları sarsmadan geri çekin. Parçaları oynatmadan önce iyice soğuduğundan **emin olun**.

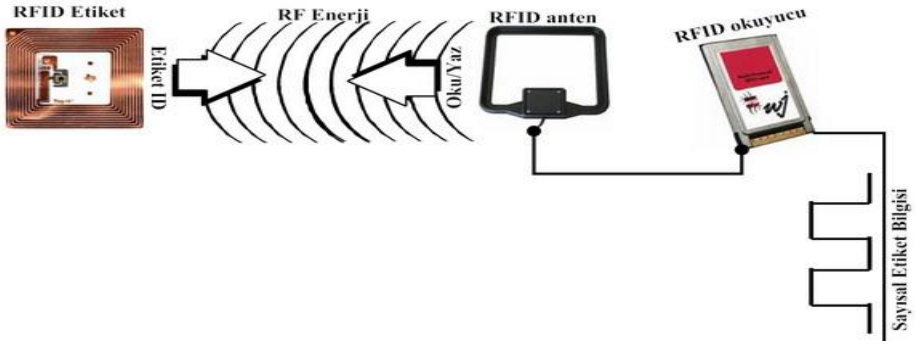


Arduino ile RFID Modülü kullanımı ve uygulama örnekleri

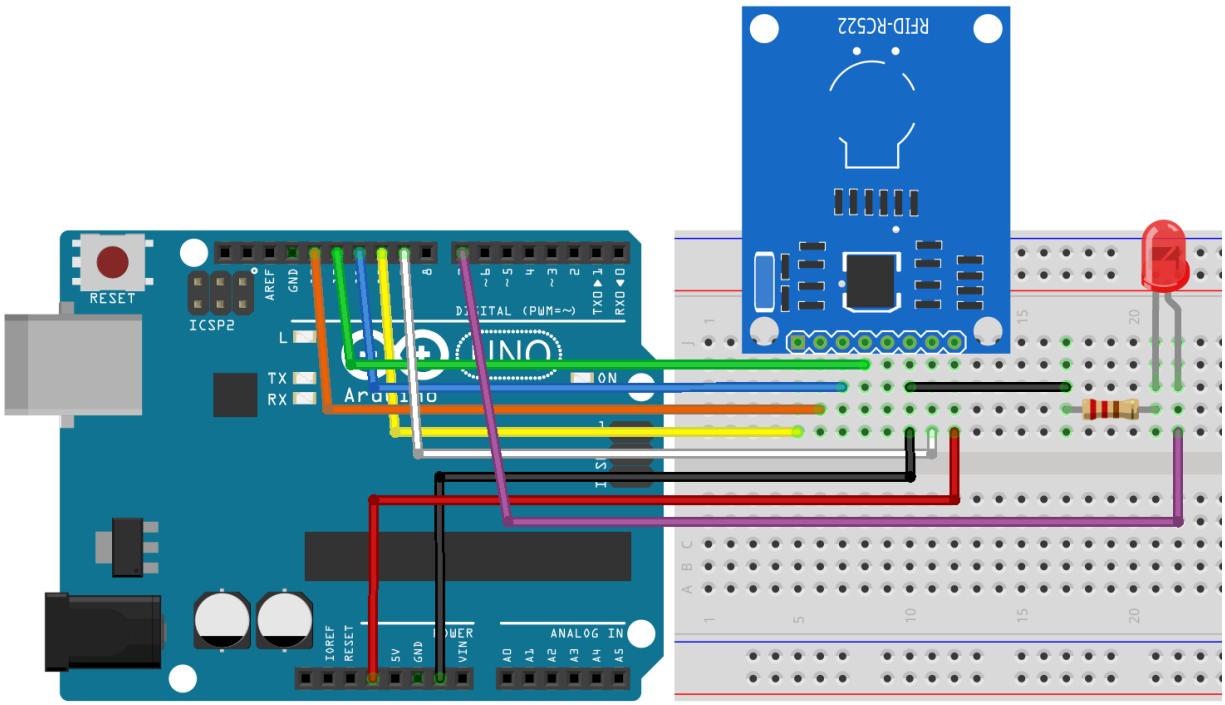
RFID bir okuyucu ve bir etiketten meydana gelen otomatik bir tanıma sistemidir. Etiketin içinde bir mikroçip ve mikroçipi saran bir anten bulunmaktadır. Okuyucu ile etiket arasında, elektromanyetik dalgalar vasıtasıyla iletişim kurulmaktadır. Okuyucunun yaydığı elektromanyetik dalgalar, bir enerji olarak çiple buluşup onu harekete geçirmekte ve etiketten okuyucuya veri transferi yapılmaktadır. Tüm bunlar belli bir mesafede, herhangi bir temas olmadan ve kablosuz olarak gerçekleşmektedir.



Okuyucu aldığı veri dalgasını sayısal dalga biçimine dönüştürerek seri veri olarak aktarmaktadır.



Kullandığımız kartların kendilerine ait UID isimli bir numarası vardır. Bu numara, her kart için farklıdır. Okuyucumuza kartımızı veya anahtarlığımızı yaklaştırdığımızda bu numara okunarak işlem yapılır. Bu uygulamada, öncelikle elimizdeki kartların UID'lerini Arduino'muzun dahili EEPROM'una kaydederek ve daha sonra okuttuğumuz kartın UID'sini, bellekteki UID değerleriyle karşılaştırarak işlem yapacağız. Devre şemamız şu şekilde:



Kart_yukleme §

```
#include <SPI.h>
#include <MFRC522.h>
#include <EEPROM.h>
#define RST_PIN 9
#define SS_PIN 10
byte readCard[4];
int successRead;
MFRC522 mfrc522(SS_PIN, RST_PIN);
MFRC522::MIFARE_Key key;
void setup()
{
  Serial.begin(9600);
  SPI.begin();
  mfrc522.PCD_Init();
  Serial.println("RFID KART KAYIT UYGULAMASI");
  Serial.println("--------------------------");
  Serial.println("Lutfen 1 numarali karti okutun");
  Serial.println();
  do {
    //okuma başarılı olana kadar getID fonksiyonunu çağır
    successRead = getID();
  }
  while (!successRead);
  for ( int i = 0; i < mfrc522.uid.size; i++ )
  {
    //kartın UID'sini EEPROM'a kaydet
    EEPROM.write(i, readCard[i] );
  }
```

```
29    Serial.println("Kart EEPROM'a kaydedildi.");
30    Serial.println();
31    Serial.println("Lutfen 2 numarali karti okutun.");
32    Serial.println();
33    do {
34      successRead = getID();
35    }
36    while (!successRead);
37    for ( int i = 0; i < mfrc522.uid.size; i++ )
38    {
39      EEPROM.write(i + 4, readCard[i] );
40    }
41    Serial.println("Kart EEPROM'a kaydedildi.");
42    Serial.println();
43    Serial.println("Kart kayit islemi basarili!");
44  }
45  void loop(){
46  }
47  int getID() {
48    //yeni bir kart okunmadıysa 0 döndür
49    if ( ! mfrc522.PICC_IsNewCardPresent()) {
50      return 0;
51    }
52    if ( ! mfrc522.PICC_ReadCardSerial()) {
53      return 0;
54    }
55    Serial.print("Kart UID'si: ");
56    //kartın UID'sini byte byte oku ve seri monitöre yaz
57    for (int i = 0; i < mfrc522.uid.size; i++) {  //
58      readCard[i] = mfrc522.uid.uidByte[i];
59      Serial.print(readCard[i], HEX);
60    }
61    Serial.println("");   //kart okumayı durdur ve 1 döndür (okuma başarılı)
62    mfrc522.PICC_HaltA();
63    return 1; }
```

Kart_okuma §

```
#include <SPI.h>
#include <MFRC522.h>
#include <EEPROM.h>
#define RST_PIN 9
#define SS_PIN 10
#define ledPin 7
MFRC522 mfrc522(SS_PIN, RST_PIN);
String lastRfid = "";
String kart1 = "";
String kart2 = "";
MFRC522::MIFARE_Key key;
void setup()
{
  Serial.begin(9600);
  SPI.begin();
  mfrc522.PCD_Init();
  pinMode(ledPin, OUTPUT);
  Serial.println("RFID KART OKUMA UYGULAMASI");
  Serial.println("--------------------------");
  Serial.println();
  //EEPROM'dan kart bilgisini oku
  readEEPROM();
}
void loop()
{
  //yeni kart okunmadıkça devam etme
  if ( ! mfrc522.PICC_IsNewCardPresent())
  {
    return;
  }
```

```
31  if ( ! mfrc522.PICC_ReadCardSerial())
32  {
33    return;
34  }
35  //kartın UID'sini oku, rfid isimli string'e kaydet
36  String rfid = "";
37  for (byte i = 0; i < mfrc522.uid.size; i++)
38  {
39    rfid += mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ";
40    rfid += String(mfrc522.uid.uidByte[i], HEX);
41  }
42  //string'in boyutunu ayarla ve tamamını büyük harfe çevir
43  rfid.trim();
44  rfid.toUpperCase();
45    if (rfid == lastRfid)
46    return;
47  lastRfid = rfid;
48  Serial.print("Kart 1: ");
49  Serial.println(kart1);
50  Serial.print("Kart 2: ");
51  Serial.println(kart2);
52  Serial.print("Okunan: ");
53  Serial.println(rfid);
54  Serial.println();
55  //1 nolu kart okunduysa LED'i yak,
56  //2 nolu kart okunduysa LED'i söndür
57  if (rfid == kart1)
58  {
59    digitalWrite(ledPin, HIGH);
60    Serial.println("LED yandi.");
```

```
61    }
62    if (rfid == kart2)
63    {
64      digitalWrite(ledPin, LOW);
65      Serial.println("LED sondu.");
66    }
67    Serial.println();
68    delay(200);
69
70  }
71
72  void readEEPROM()
73  {
74    //EEPROM'dan ilk kartın UID'sini oku (ilk 4 byte)
75    for (int i = 0 ; i < 4 ; i++)
76    {
77      kart1 += EEPROM.read(i) < 0x10 ? " 0" : " ";
78      kart1 += String(EEPROM.read(i), HEX);
79    }
80    //EEPROM'dan ikinci kartın UID'sini oku
81    for (int i = 4 ; i < 8 ; i++)
82    {
83      kart2 += EEPROM.read(i) < 0x10 ? " 0" : " ";
84      kart2 += String(EEPROM.read(i), HEX);
85    }
86    //Okunan stringleri düzenle
87    kart1.trim();
88    kart1.toUpperCase();
89    kart2.trim();
90    kart2.toUpperCase();
91  }
```

Joy-stick ile Servo Motor kullanımı ve uygulama örnekler

Joystick pinlerinden VRx yatay eksenindeki sinyalleri, VRy dikey eksendeki sinyalleri, SW pini ise joystick tıklama buton pin değerini okumayı sağlar. Joystick modülü yatay ve dikey eksende 0 ile 1023 arasında analog değerler üretir.

Joystick modülünü arduinoya bağlamak için;

Gnd pinin arduino Gnd pinine,

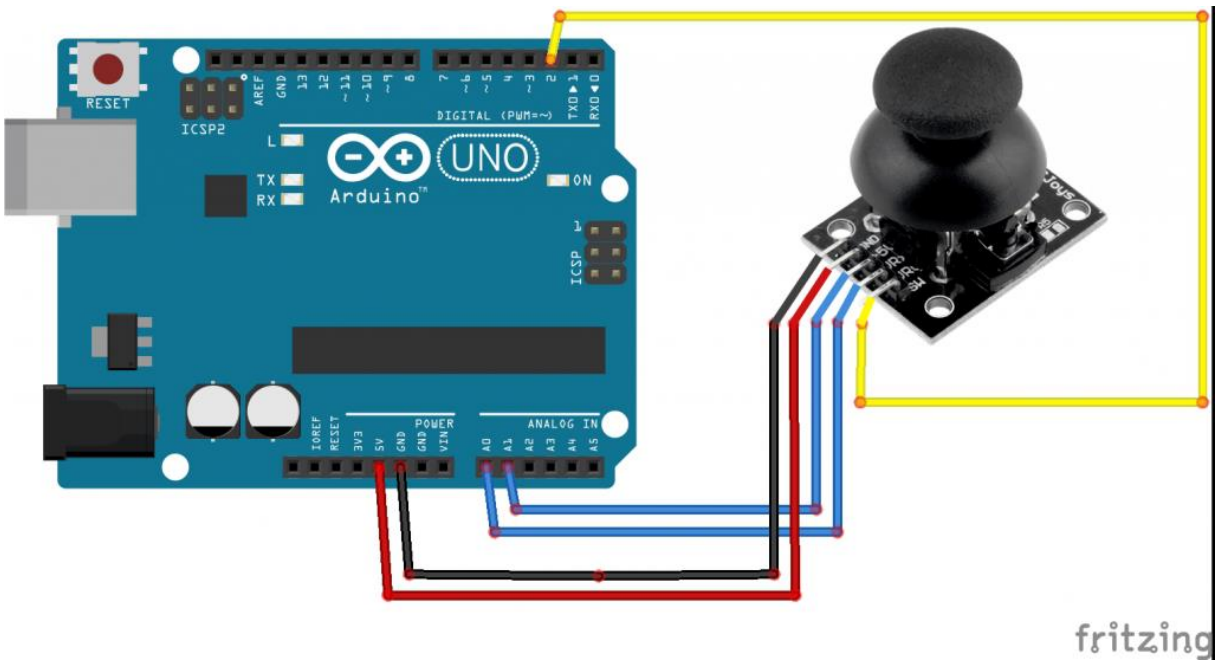
Vcc pinin arduino 5V pinine,

VRx pinin arduino A0-A5 analog pinlerinden birine,

VRy pinin arduino A0-A5 analog pinlerinden birine,

SW pinin arduino dijital pinlerinden birine bağlanmalıdır.

Yapacağımız örnek için VRx pinini A0, VRy pinini A1, ve SW pinin arduino 2 numaralı pine bağladık.



x_y_serial §

```
int xPin = A0;
// A0-A5 analog pinlerinden herhangi birine bağlanabilir.
int yPin = A1;
// A0-A5 analog pinlerinden herhangi birine bağlanabilir.
int butonPin = 2;
// Joystick buton pini arduino bağlantısı (Joystick SW çıkışı)
int xPozisyonu = 0;
int yPozisyonu = 0;
int butonDurum = 0;
void setup() {
Serial.begin(9600);
pinMode(xPin, INPUT);
pinMode(yPin, INPUT);
pinMode(butonPin, INPUT_PULLUP);
}
void loop() {
xPozisyonu = analogRead(xPin);
yPozisyonu = analogRead(yPin);
butonDurum = digitalRead(butonPin);
Serial.print("X Pozisyonu: ");
Serial.print(xPozisyonu);
Serial.print(" | Y Pozisyonu: ");
Serial.print(yPozisyonu);
Serial.print(" | Buton Durum: ");
Serial.println(butonDurum);
delay(400);
}
```

Yukarıdaki bağlantıları yapıp programımızı yükledikten sonra X Y değerlerini 512 ye yakın olarak seri ekranda görebiliriz. Joystick ile yatay hareket ile X , dikey hareket ile Y değerinin 0 ile 1023 arasında değiştiğini görebiliriz. Buton durumunun değişimini bastıkça görebiliriz.

Şimdi servo motorlar ile X Y hareketini sağlayacak programı inceleyelim. Servo motorların sinyal pinlerini programdaki port numaralarına bağlayarak joystick ile hareket ettirebiliriz.

x_y_servo §

```
#include <Servo.h>
Servo servox;  // create servo object to control a servo X
Servo servoy;  // create servo object to control a servo Y
int potpinx = 0;  // analog pin used to connect the potentiometer X
int potpiny = 1;  // analog pin used to connect the potentiometer Y
int valx;    // variable to read the value from the analog pin0
int valy;    // variable to read the value from the analog pin1
void setup() {
  servox.attach(9);  // servo X on pin 9
  servoy.attach(10);  //servo Y on pin 10
  pinMode(7,INPUT_PULLUP);
  pinMode(8,OUTPUT);
}
void loop() {
  if (!digitalRead(7))
    digitalWrite(8,HIGH);
    else
    digitalWrite(8,LOW);
  valx = analogRead(potpinx);            // reads the value of the p
  valy = analogRead(potpiny);            // reads the value of the p
  valx = map(valx, 0, 1023, 0, 180);     // scale it to use it with
  valy = map(valy, 0, 1023, 0, 180);     // scale it to use it with
  servox.write(valx);                  // sets the servoX position a
  servoy.write(valy);                  // sets the servoY position a
  delay(50);                           // waits for the servo to get
}

```

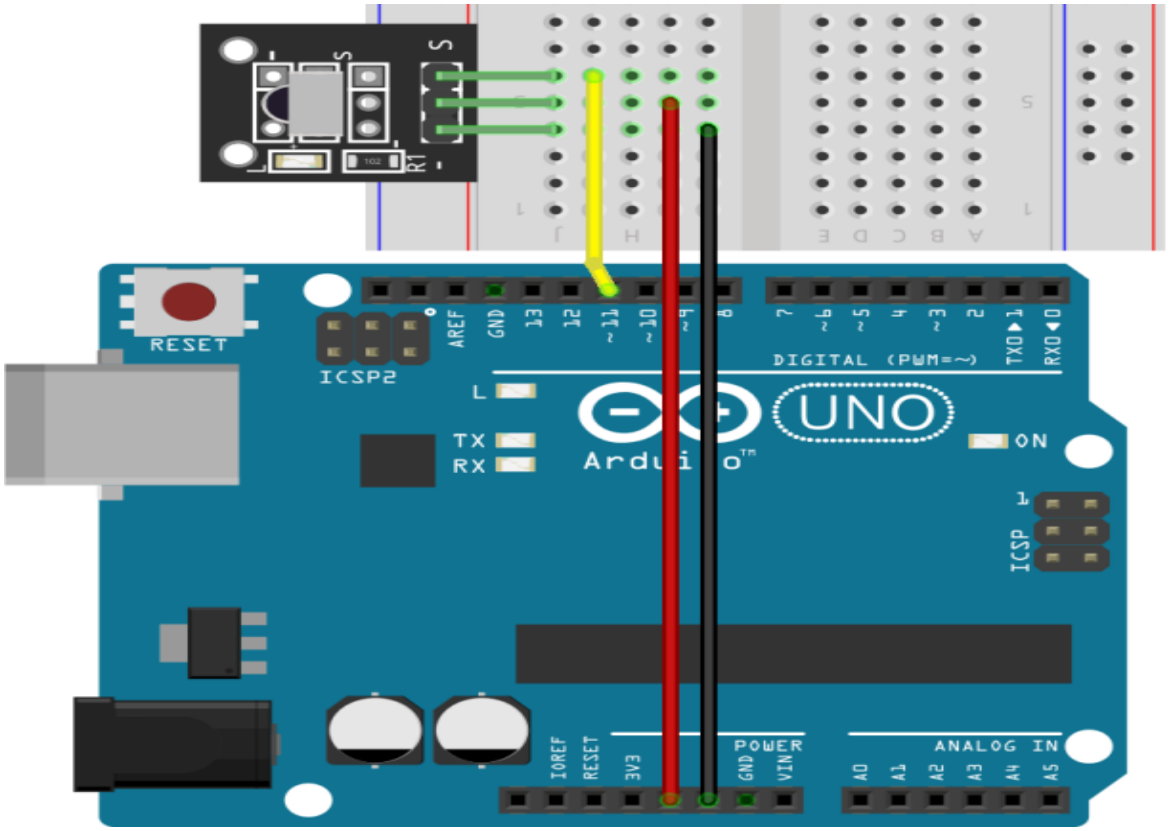
Arduino ile Kızıl ötesi modülü kullanımı ve uygulama örnekleri

Kumanda üzerinde bir kızılötesi (infrared, ir) LED bulunur. Bu LED, kumanda üzerindeki herhangi bir tuşa bastığımızda önceden belirlenmiş bir kod verecek şekilde belirli bir frekansta yanıp söner. Çoğu kumanda için bu frekans 38 kHz'tir. 38 kHz'lik taşıyıcı sinyale her bir tuş için farklı bir kod oluşturacak şekilde modülasyon uygulanır. Her marka için farklı modülasyon ve kodlama teknikleri mevcuttur. Projede kullandığımız 38 kHz kızılötesi alıcı, aldığı sinyali demodüle ederek Arduino'ya doğrudan basılan buton ile ilgili kodu göndermektedir. Bu sayede farklı marka ve model kumandaları 38 kHz taşıyıcı sinyale sahip olduğu sürece bu alıcı ile kullanabilmekteyiz."

Öncelikle Arduino ile kızılötesi kumandayı kullanabilmemiz için bir kütüphaneye ihtiyaç duyuyoruz. Bu kütüphaneyi aşağıdaki adresten indirebiliriz:

https://github.com/z3t0/Arduino-IRremote

Kumandamızdaki tuşları kullanabilmemiz için öncelikle tuşlara ait kodları öğrenmemiz gerekli. Kütüphaneyi yükledikten sonra örnek kodlar içerisinde yer alan IRrecvDemo isimli programı Arduino'muza yüklememiz gerekiyor. **Dosya -> Örnekler -> IRremote ->RrecvDemo** adımlarını takip ederek gerekli koda ulaşabiliriz. Alıcımızı yandaki devre şemasına göre Arduino kartımıza bağlıyoruz.



IR_serial

```
#include <IRremote.h>
int RECV_PIN = 2;
IRrecv irrecv(RECV_PIN);
decode_results okunan;
void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver
  Serial.println("InfraRed kod okuma ");
}
void loop() {
  if (irrecv.decode(&okunan)) {
    long kod=okunan.value;
    Serial.print("Okunan kod = ");
    Serial.println(kod, DEC);
    irrecv.resume(); // Receive the next value
  }
 delay(100);
}
```

kumandanın tuşlarına sıra ile basarak seri port ekranında listelenen kodları bir sonraki uygulamamızda kullanacağız. Yeni uygulamada kumanda üzerindeki tuşlar ile dört adet ledi yakıp söndürelim.

IR_led

```
#include <IRremote.h>
int RECV_PIN = 2;
IRrecv irrecv(RECV_PIN);
decode_results results;
#define BUTON1 16753245
#define BUTON2 16736925
#define BUTON3 16769565
#define BUTON4 16720605
#define BUTON5 16712445
#define BUTON6 16711935
#define BUTON7 16769055
#define BUTON8 16754775
#define BUTON9 16748655
#define BUTON0 16750695
#define BUTON_UP      16718055
#define BUTON_DOWN    16730805
#define BUTON_RIGHT   16734885
#define BUTON_LEFT    16716015
#define BUTON_OK      16726215
int led1 = 7;
int led2 = 6;
int led3 = 5;
int led4 = 4;
```

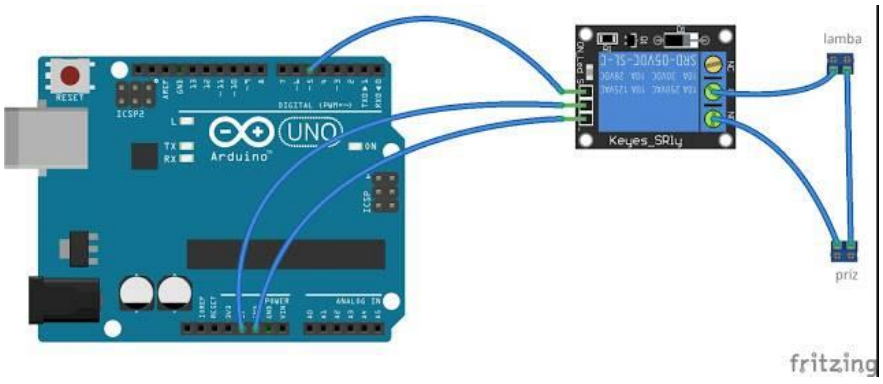
IR_led

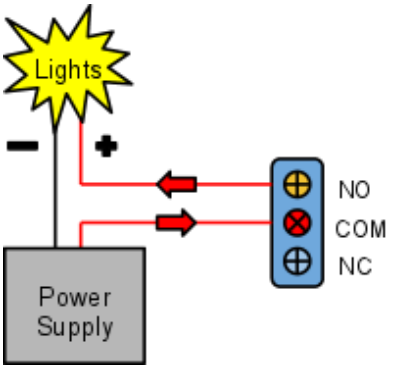
```
24 void setup()
25 {
26   pinMode(led1, OUTPUT);
27   pinMode(led2, OUTPUT);
28   pinMode(led3, OUTPUT);
29   pinMode(led4, OUTPUT);
30   Serial.begin(9600);
31   irrecv.enableIRIn();
32 }
33 void loop() {
34   if (irrecv.decode(&results))
35   {
36     if (results.value == BUTON1)
37     {
38       digitalWrite(led1, !digitalRead(led1));
39       if (digitalRead(led1) == HIGH)
40       {        Serial.println("LED 1 yandi");       }
41       else
42       {        Serial.println("LED 1 sondu");       }
43     }
44     if (results.value == BUTON2)
45     {      digitalWrite(led2, !digitalRead(led2));
46       if (digitalRead(led2) == HIGH)
47       {        Serial.println("LED 2 yandi");       }
48       else
49       {        Serial.println("LED 2 sondu");       }
50     }
51     if (results.value == BUTON3)
52     {       digitalWrite(led3, !digitalRead(led3));
53       if (digitalRead(led3) == HIGH)
54       {        Serial.println("LED 3 yandi");       }
55       else
56       {        Serial.println("LED 3 sondu");       }
57     }
58     if (results.value == BUTON4)
59     {       digitalWrite(led4, !digitalRead(led4));
60       if (digitalRead(led4) == HIGH)
61       {        Serial.println("LED 4 yandi");       }
```

```
62        else
63        {         Serial.println("LED 4 sondu");       }
64      }
65      if (results.value == BUTON0)
66      { digitalWrite(led1, LOW);
67        digitalWrite(led2, LOW);
68        digitalWrite(led3, LOW);
69        digitalWrite(led4, LOW);
70        Serial.println("Tum LED'ler sondu");      }
71      if (results.value == BUTON5)
72      { digitalWrite(led1, HIGH);
73        digitalWrite(led2, HIGH);
74        digitalWrite(led3, HIGH);
75        digitalWrite(led4, HIGH);
76        Serial.println("Tum LED'ler yandi");      }
77      irrecv.resume();
78    }
```

Arduino ile Role modülü kullanımı ve uygulama örnekleri

Arduino röle uygulaması ile yüksek akım elektrik çeken cihazların kontrolünü sağlayabilirsiniz.Örnek verirsek evinize cep telefonu kontrollü akıllı bir lamba sistemi yapmak istiyorsunuz.Bu sistemi yapabilmek için röle kullanmanız gerekecek.Çünkü evde kullandığımız ampuller 220 volt ile çalışır arduino üzerinden direk akım çektiremezsiniz arduino nun vereceği maksimum 5 volt tur. Arduino uyumlu röle modülleri piyasada hazır ve çokça satılmaktadır.Eğer yukarıda bahsettiğim gibi akıllı bir ev sistemi tasarlamak istiyorsanız röle kullanmanız kaçınılmaz olacaktır. Bu modülde giriş için üç pin mevcut: VCC, GND, IN pinleri. VCC ve GND pinleri 5V enerjinin verildiği pinlerdir. IN pini ise röleyi kontrol ettiğimiz pindir. Modülün çıkışında ise rölenin C, NO, NC pinleri vardır. Röle bobini enerjisiz iken, yani röle kontağını çekmemiş ise C ve NC kısa devredir, röle kontağını çektiğinde ise C ve NO kısa devre olur.





Arduino ile Bluetooth modülü kullanımı ve uygulama örnekleri

HC06 Bluetooth-Serial Modül Kartı, Bluetooth SSP(Serial Port Standart) kullanımı ve kablosuz seri haberleşme uygulamaları için tasarlanmıştır. Hızlı prototiplemeye imkan sağlaması, breadboard, arduino ve çeşitli devrelerde rahatça kullanılabilmesi için gerekli pinler devre kartı sayesinde dışarıya alınmıştır.
Standart pin yapısı sayesinde istenilen ortamlarda rahatça kontrol edilebilir. Bununla beraber ürün beraberinde gönderilen jumper kablolar ile bağlantılar rahatlıkla yapılabilir. Bluetooth 2.0'ı destekleyen bu kart, 2.4GHz frekansında haberleşme yapılmasına imkan sağlayıp açık alanda yaklaşık 30 metrelik bir haberleşme mesafesine sahiptir. Bir çok hobi, robotik ve akademik projede kullanılabilir.
HC05'in aksine yanlızca slave modda kullanılabilir.
Özellikleri:

- Bluetooth Protokolü: Bluetooth 2.0+EDR(Gelişmiş Veri Hızı)
- 2.4GHz haberleşme frekansı
- Hassasiyet: ≤-80 dBm
- Çıkış Gücü: ≤+4 dBm
- Asenkron Hız: 2.1 MBps/160 KBps
- Senkron Hız: 1 MBps/1 MBps
- Güvenlik: Kimlik Doğrulama ve Şifreleme
- Çalışma Gerilimi: 1.8-5V(Önerilen 3.3V)

HC-06 Modülü Set Etme(AT Komutları)

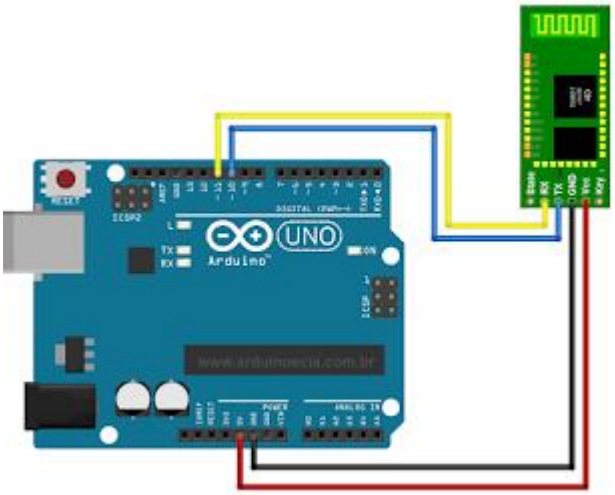
Öncelikle arduinomuza kodları yüklüyoruz. Burada önemli olan Tx ve Rx pinlerine dikkat etmenizdir. Mevcut kütüphane ile Rx Tx pinlerini 10,11 yapıyoruz, bunun amacı HC-06 Modül ile haberleşmek için 10,11 pinlerini kullanırken pc ile haberleşirken 0,1 pinlerini kullansın ki modülümüz ile bilgisayar haberleşmesi çakışmasın. arduino ile Hc 06 nın Rx Tx ini çapraz bağlıyoruz(rx>>tx tx>>rx(alıcı-verici verici-alıcı şeklinde eşleşiyor)).Kodları arduinoya yükledikten sonra Hc 06 modülün takıyoruz ve seri porta AT komutlarını yazıyoruz.

hc06_name §

```
#include <SoftwareSerial.h>  //  Seri Haberleşme kütüphanesini ekledik
SoftwareSerial BTSerial(10, 11); // RX, TX  // BT için ayrıca bir Seri Haberleşme pini tanımladık
void setup() {
Serial.begin(9600);
Serial.println("Enter AT commands:");
BTSerial.begin(9600);  //BT Seri haberleşmesini 9600 ile başlattık*
}
void loop()
{
if (BTSerial.available())
Serial.write(BTSerial.read());
if (Serial.available())
BTSerial.write(Serial.read());
}
```

Seri porta "AT" komutu yazınca size "ok" cevabı vermesi lazım. "ok" cevabını aldıktan sonra seri porta "AT+NAMEmodüle_verilecek_isim" yazıyoruz ve karşılığında "OKName" cevabını almayı bekliyoruz. "AT+PINxxxx" yazıyoruz ve karşılığında "OKsetpin" cevabını almayı bekliyoruz. "AT+BAUDx" yazıyoruz ve karşılığında "OKx" cevabını almayı bekliyoruz. buradaki x altta verilen baund değerleri içindir. örneğin "OK4" demek bound ayarını 9600 yaptığını belirtir. Burada önemli olan şey baund ayarını 9600 yaptıktan sonra arduinomuzunda baund ayarını 9600 yapmalıyız aksi halde haberleşme baund ları denk gelmediğinde haberleşme gerçekleşmez.

x değieri için
1---------1200
2---------2400
3---------4800
4---------9600
5---------19200
6---------38400
7---------57600
8---------115200



Android cihazlar için PlayStore uygulamasında **Color LED Controller** uygulamasını indirdikten sonra aşağıdaki programı UNO ya yüklüyoruz. RGB led ayaklarını 330 ohm direnç ile 9,10,11 nolu pinlere bağlıyoruz. Uygulamadaki BT List butonu ile HC-06 kartımızı seçiyoruz. Daha sonra renk çarkından renk seçerek RGB ledin rengini ayarlayalım.

RGB §

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(7, 8); // 7:RX, 8:TX
int kirmizi,yesil,mavi;
int R=9;
int G=10;
int B=11;
void setup() {
  Serial.begin(9600);
  mySerial.begin(9600);
 mySerial.setTimeout(50);
}
void loop() {
   while (mySerial.available() > 0) {
     kirmizi = mySerial.parseInt();
    yesil = mySerial.parseInt();
    mavi = mySerial.parseInt();
     if (mySerial.read() == ')')  // )son karakter
```

Arduino ile H Bridge modülü ile DC motor sürme uygulamaları

DC MOTORLAR

DC motorlar ucuz, küçük ve etkilidir. DC motorlar robotlarda veya herhangi bir sistemde direkt ya da dişli kutularıyla (redüktörlü ya da redüktörsüz olarak) birlikte kullanılabilirler. DC motorların robotlarda kullanımına dair temel özellikler aşağıda açıklanmıştır.

Yön: DC motorlara bir güç kaynağı bağlandığında DC motorun dönüş yönü akımın yönüne bağlıdır. Akımın yönü terslendiğinde DC motorun dönüş yönü de terslenmiş olur.

Hız: Bir motorun hızı rpm (rotations per minute - bir dakikada tamamlanan devir sayısı) ile ölçülür. Motorun hızı voltaja ve yüke bağlıdır.

Bir DC motorun hızının voltaja ve yüke göre değişimini değerlendirmek için iki durum düşünülebilir. Bunlardan ilki; DC motora yük binmeyen ya da sabit bir yükün olduğu bir sistemdir. Böyle bir sistemde DC motorun hızı uygulanan voltaja bağlıdır ve voltaj arttıkça hız da artar. İkinci durum ise; DC motora binen yükün zamana ya da gerçekleştirilen göreve göre değiştiği bir sistemdir. Bu durumda DC motorun hızı yüke bağlı olacaktır. Yük arttıkça uygulanan güç de artar ve güç arttıkça hız azalır.

Voltaj: Küçük DC motorlar 1,5 V ile 48 V arasında değişen voltaj değerlerine sahip olarak bulunabilirler. Her bir DC motor için belirtilen voltaj değeri, o DC motorun kendi verilen hız, güç ve akım değerlerinde stabil çalıştığı voltaj değeridir. Robotlarda ve diğer sistemlerde DC motorları kullanırken de bu voltaj değeri, DC motora verilecek maksimum çalışma voltajını belirlediği için önemlidir.

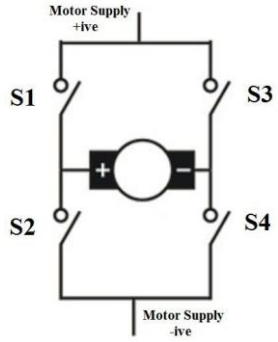
Akım: Bir DC motor belirtilen voltaj değerinde çalıştırıldığında DC motorun çekeceği akım yüke bağlıdır. Yük arttıkça DC motorun çektiği akım da artar. DC motor, maksimum akım sınırının aşılacağı fazla bir yükle çalıştırılmamalıdır. Böyle bir durumda DC motor kısa devreye neden olur ve uygulanan güç ısıya dönüşür. Bu durum uzun sürerse DC motor yanabilir. Genellikle DC motorların uygulama akımı aralığı 50 mA den başlayıp 2A üzerine kadar çıkabilir.

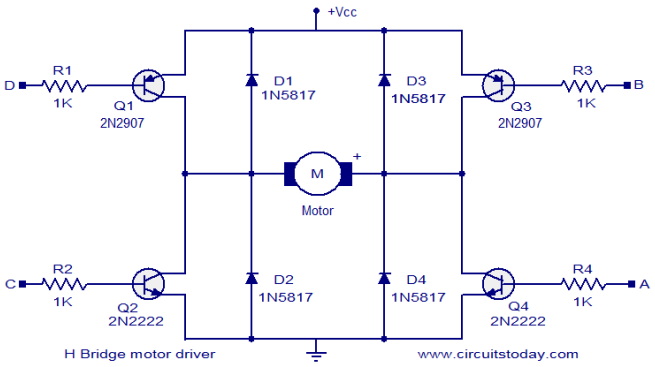
Güç: Güç bir motorun akımı ve voltajının çarpım değeridir. Ancak robot projelerinde ve mekanik sistemlerde bir motorun ürettiği kuvvetin tork (motorun dönme momenti) cinsinden değerlendirilmesi normaldir.

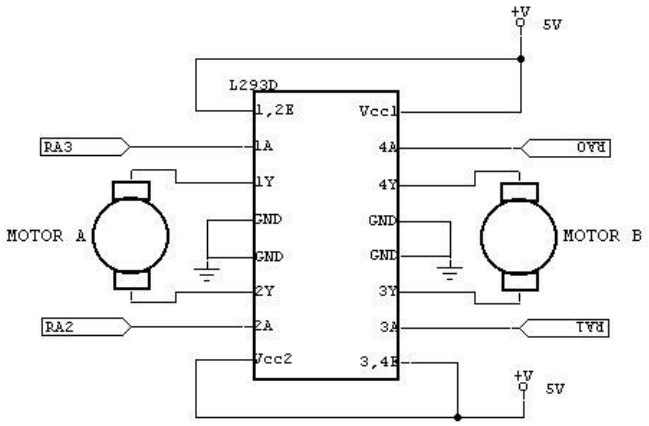
Tork motorun dönme momentidir. Torku yüksek olan motor düşük olana göre daha güçlüdür. Tork motorun elektriksel ve mekanik karakteristiklerine ve motor şaftının yarı çapına bağlıdır. Bir motorun torku motora bağlanan dişli kutularıyla (redüktör) değiştirilebilir. Dişli kutuları hızın azaltılmasını ve gücün arttırılmasını sağlar. Örneğin; motor şaftının yarıçapının 10 katı yarıçapa sahip bir dişli motora eklendiğinde, motorun hızı 10 kat düşer ve gücü de 10 kat artar.

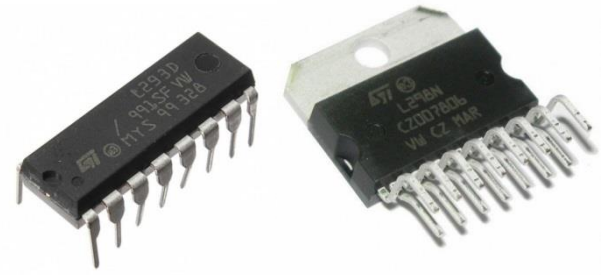
Robotikte, çeşitli boyutlarda ve redüksiyon oranlarında dişli kutuları motorun karakteristik özelliklerini isenilen işi yapabilecek düzeye getirmek için sıklıkla kullanılır. Bir motoru kullanırken torkunu bilmek önemlidir. Tork ve redüksiyon oranı bilindiğinde sistemin son çıkış gücü kolaylıkla belirlenebilir.

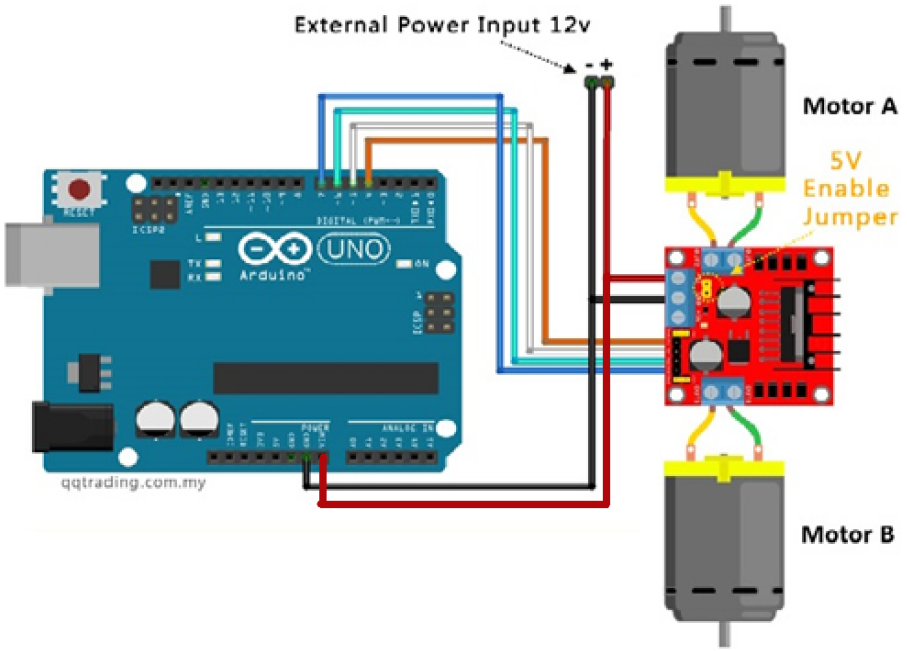
DC motorların yön kotrolü için H köprürsü yaygın olarak kullanılır. Transistörler ile yapılabildiği gibi özel entegrelerde kullanılır. L293 entegresi bağlantılarını aşağıdaki resimden inceleyelim

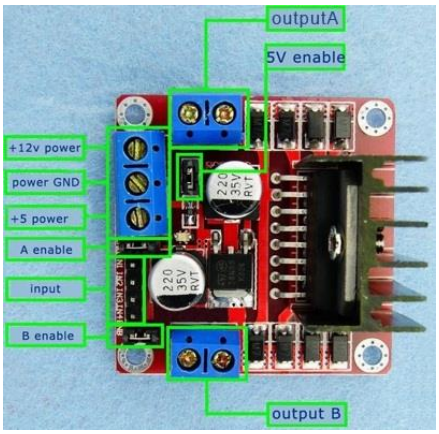












L298 Entegresi ile daha yüksek güçlü motorlar sürülebilir.

dc_motor_L298

```
// Motor sürücümüze bağladığımız pinleri tanımlıyoruz
const int in1 = 4;
const int in2 = 5;
const int in3 =  6;
const int in4 =  7;

void setup()
//Tüm pinlerden güç çıkışı olacağı için OUTPUT olarak ayarladık
{
pinMode(in1, OUTPUT);
pinMode(in2, OUTPUT);
pinMode(in3, OUTPUT);
pinMode(in4, OUTPUT);

}

void loop()

{
// motor A ileri
digitalWrite(in1, HIGH);
digitalWrite(in2,  LOW);
// motor B ileri
digitalWrite(in3, HIGH);
digitalWrite(in4,  LOW);
}
```

DC motorlar ters dönüyorsa motor kablolarını yer değiştirelim. Motorları dönüş yönünü değiştirmek için programın loop döngüsünü aşağıdaki gibi değiştirelim.

```
{
// motor A geri
digitalWrite(in1, LOW);
digitalWrite(in2,  HIGH);
// motor B geri
digitalWrite(in3, LOW);
digitalWrite(in4,  HIGH);
}
```

dc_motor_serial §

```
#define In1 8 //A Motor ileri pini
#define In2 7 //A Motor geri pini
#define In3 6 //B Motor ileri pini
#define In4 5//B Motor geri pini
int data;
void setup() {
pinMode(In1,OUTPUT); // kontrol pin numaraları çıkışa yönlendiriliyor
pinMode(In2,OUTPUT);
pinMode(In3,OUTPUT);
pinMode(In4,OUTPUT);
Serial.begin(9600);
Serial.print("1: ileri  2: geri  3: sag   4: sol   5:");
dur();
}
void dur() {
  digitalWrite(In1,LOW);
  digitalWrite(In2,LOW);
  digitalWrite(In3,LOW);
  digitalWrite(In4,LOW);
  Serial.println(" DUR ");
  delay(500);
}
void ileri() {
  digitalWrite(In1,HIGH);
  digitalWrite(In2,LOW);
  digitalWrite(In3,HIGH);
  digitalWrite(In4,LOW);
  Serial.println(" ileri ");
}
void geri() {
  digitalWrite(In1,LOW);
  digitalWrite(In2,HIGH);
  digitalWrite(In3,LOW);
  digitalWrite(In4,HIGH);
  Serial.println(" Geri ");
}
void sag() {
  digitalWrite(In1,LOW);
  digitalWrite(In2,HIGH);
  digitalWrite(In3,HIGH);
  digitalWrite(In4,LOW);
  Serial.println(" SAG ");
}
void sol() {
  digitalWrite(In1,HIGH);
  digitalWrite(In2,LOW);
  digitalWrite(In3,LOW);
  digitalWrite(In4,HIGH);
  Serial.println(" SOL ");
}
void loop() {
  if (Serial.available()>0){
    data=Serial.read();
    Serial.print(data);
    if (data==49)       ileri();
    if (data==50)       geri();
    if (data==51)       sag();
    if (data==52)       sol();
    if (data==53)       dur();
    data=0;
  } }
```

Motor hız kontrolü için yukarıdaki programa bazı ilaveler yapabiliriz. analogWrite komutu ile motor sürücünün EnA ve EnB girişleri kontrol edilebilir. Bunun için aşağıdaki kodlar programa yazılabilir.

dc_motor_serial_hiz§

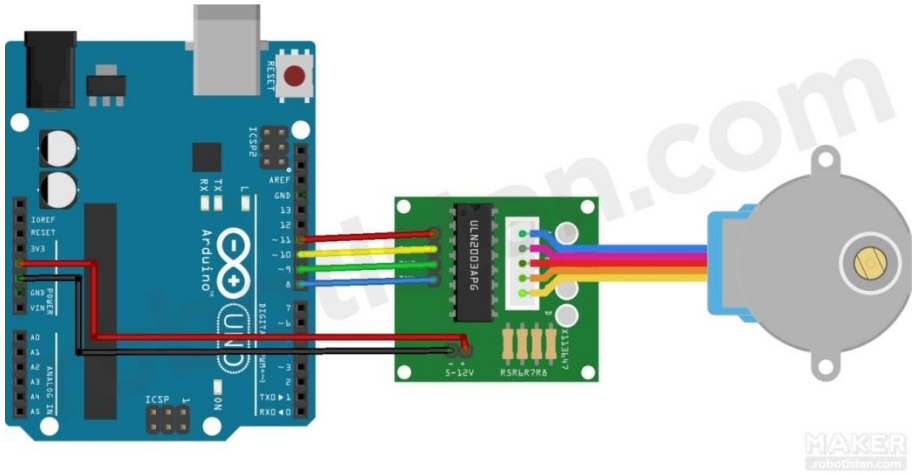
```
void setup() {
pinMode(In1,OUTPUT); // kontrol pin numaraları çıkışa yönlendiriliyor
pinMode(In2,OUTPUT);
pinMode(In3,OUTPUT);
pinMode(In4,OUTPUT);
analogWrite(9,hiz;
analogWrite(10,hiz);
Serial.begin(9600);
Serial.print("1:ileri  2:geri  3:sag   4:sol   5:dur 6:HIZ+  7:HIZ- ");

}
```

dc_motor_serial_hiz§

```
void loop() {
  if (Serial.available()>0){
    data=Serial.read();
    Serial.print(data);
    if (data==49)       ileri();
    if (data==50)       geri();
    if (data==51)       sag();
    if (data==52)       sol();
    if (data==53)       dur();
    if (data==54)       {
      hiz=hiz+50;
      if(hiz>250) hiz=250 }
    if (data==55)       {
      hiz=hiz-50;
      if(hiz<50) hiz=50 }
    }
    data=0;
```

Arduino ile ULN2003A modülü ile Step motor sürme uygulamaları

Step motorlar elektrik enerjisini dönme hareketi ile fiziksel enerjiye çeviren elektromekanik aygıtlardır. İsimlerinden de anlaşıldığı üzere adım adım hareket eden motorlardır. Biraz daha detaylı olarak açıklayacak olursak, girişlerine uygulanan pals sinyallerine karşı analog dönme hareketi çıkışı üreten, bu dönme hareketini adım adım ve çok hassas kontrolle sağlayan sabit mıknatıs kutuplu motorlardır. Step motorların yapıları rotor, stator ve rulmanlardan oluşmaktadır. Rulmanlar, rotora bağlı şaftın rahat hareket etmesini sağlarlar. Statorun birden fazla kutbu vardır. Kutup sayısı motordan motora değişmektedir ancak genellikle bu sayı sekizdir. Kutupların polaritesi elektronik anahtarlar vasıtasıyla sürekli değişir. Rotorun mıknatıslığı ya sabit mıknatıs ile ya da dış uyarım teknikleri ile meydana getirilir.



step_direc_drive_1way §

```
static const int Pin0 = 8;
static const int Pin1 = 9;
static const int Pin2 = 10;
static const int Pin3 = 11;
boolean forward = true;
int step = 0;
void setup() {
  pinMode(Pin0, OUTPUT);
  pinMode(Pin1, OUTPUT);
  pinMode(Pin2, OUTPUT);
  pinMode(Pin3, OUTPUT);
}
void loop() {
  switch(step) {
    case 0:
      digitalWrite(Pin0, LOW);
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, HIGH);
      break;
    case 1:
      digitalWrite(Pin0, LOW);
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, HIGH);
      digitalWrite(Pin3, HIGH);
    case 2:
      digitalWrite(Pin0, LOW);
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, HIGH);
      digitalWrite(Pin3, LOW);
      break;
    case 3:
      digitalWrite(Pin0, LOW);
      digitalWrite(Pin1, HIGH);
      digitalWrite(Pin2, HIGH);
      digitalWrite(Pin3, LOW);
      break;
    case 4:
      digitalWrite(Pin0, LOW);
      digitalWrite(Pin1, HIGH);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, LOW);
      break;
    case 5:
      digitalWrite(Pin0, HIGH);
      digitalWrite(Pin1, HIGH);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, LOW);
      break;
    case 6:
      digitalWrite(Pin0, HIGH);
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, LOW);
      break;
    case 7:
      digitalWrite(Pin0, HIGH);
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, HIGH);
      break;
  }
  if(forward) step++;
  else step--;
  if(step>7) step=0;
  if(step<0) step=7;
  delay(1);
}
```

serial_step_konum §

```
#include <Xl13647Stepper.h>
static const int STEPS_PER_REVOLUTION =64*32 ;  // change this to fit the number of steps
// initialize the stepper library on pins 8 through 11:
Xl13647Stepper myStepper(STEPS_PER_REVOLUTION, 8, 9, 10, 11);
int konum=0;
void setup() {
  myStepper.setSpeed(10);
  Serial.begin(9600);
  Serial.println("STEP MOTOR KONTROLÜ ( Saatin ters yönü için - açısal değer girin )");
}
void loop() {
  if(Serial.available()>0)
  {
   long aci=Serial.parseInt();
   myStepper.step(aci*64*32/360);
   konum=konum+aci;
   Serial.print("Motor açısal konumu = ");
   Serial.print(konum%360);
   Serial.print("     ");
   Serial.print(" Dönüş sayısı = ");
   Serial.println(konum/360);
  delay(100);
  }
}
```

Arduino ile Bluetooth kontrollü robot yapımı

HC06 Bluetooth modülümüzü UNO 0,1 haberleşme portlarına yükleme işleminden sonra bağlıyoruz. Android cihazımıza ***'Arduino Bluetooth RC Car'*** uygulamasını yükleyip, bağlantı ayarlarından sonra arabamızı kontrol ediyoruz.

Bluetooth_araba

```
#define In1 8 //A Motor ileri pini
#define In2 7 //A Motor geri pini
#define In3 6 //B Motor ileri pini
#define In4 5//B Motor geri pini
int data;
void setup() {
pinMode(In1,OUTPUT);
pinMode(In2,OUTPUT);
pinMode(In3,OUTPUT);
pinMode(In4,OUTPUT);
Serial.begin(9600);
}
void dur() {
  digitalWrite(In1,LOW);
  digitalWrite(In2,LOW);
  digitalWrite(In3,LOW);
  digitalWrite(In4,LOW);
  delay(10);
}
void ileri() {
  digitalWrite(In1,HIGH);
  digitalWrite(In2,LOW);
  digitalWrite(In3,HIGH);
  digitalWrite(In4,LOW);
}
```

Bluetooth_araba §

```
void geri() {
  digitalWrite(In1,LOW);
  digitalWrite(In2,HIGH);
  digitalWrite(In3,LOW);
  digitalWrite(In4,HIGH);
}
void sag() {
  digitalWrite(In1,LOW);
  digitalWrite(In2,HIGH);
  digitalWrite(In3,HIGH);
  digitalWrite(In4,LOW);
}
void sol() {
  digitalWrite(In1,HIGH);
  digitalWrite(In2,LOW);
  digitalWrite(In3,LOW);
  digitalWrite(In4,HIGH);
 }
void loop() {
  if (Serial.available()>0){
    data=Serial.read();
    Serial.print(data);
    if (data=='F')    ileri();
    if (data=='B')    geri();
    if (data=='R')    sag();
    if (data=='L')    sol();
    if (data=='S')    dur();
    data=' ';
  } }
```

Arduino ile engelden kaçan robot yapımı

Arabamıza 3d printer ile oluşturduğumuz sensör tutucu ile ultrasonik mesafe sensörümüzü monte ediyoruz. Sensör pin bağlantılarını programda yazılan portlara bağlıyoruz. Arabamız engel algılayana kadar ileri modunda çalışıyor. Engel algıladığında dur ve 1sn. geri sonra sağ yada sola fonksiyonunu 2sn. çalıştırıp engel algılamıyorsa ileri fonksiyonu ile devam ediyor. Gerekli bağlantıları yapıp işlemi gerçekleştiren programı arduino ya yüklüyoruz.