İSTANBUL GELİŞİM ÜNİVERSİTESİ

TEKNOLOJİ TRANSFER OFİSİ

UYGULAMA ve ARAŞTIRMA MERKEZİ

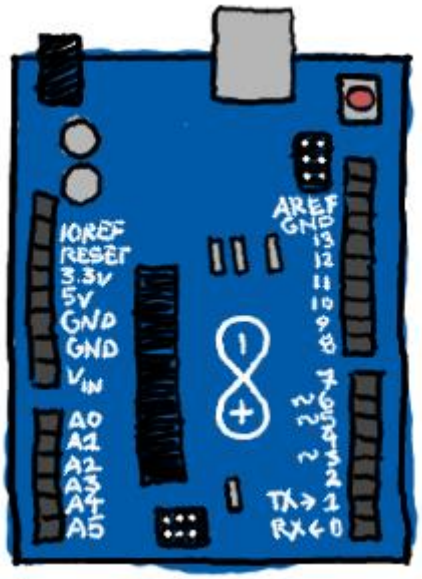
ARDUINO İLE KODLAMA EĞİTİMİ

İstanbul Gelişim Üniversitesi

Öğr. Gör. Ali ÇETİNKAYA

alcetinkaya@gelisim.edu.tr

Arduino ile Uygulamalı Programlama Dersleri



1. Arduino ile Kodlama
- Kodlama – programlama nedir?
- Mikroişlemci – mikrodenetleyici nedir?
  - Mikroişlemci
  - Mikrodenetleyici
  - Arduino IDE ve Atmel Yapısı
2. Gerekli ortamların kurulması
- Arduino
- Fritzing
- VS Atmel Studio
- Kurulum Notları
3. Dijital Dünyanın Gerçekliği
- “1” ve “0” Kavramı
- Gerilim – Volt Veri Kavramı
  - Gerilim
  - Volt
- Analog - Dijital Veri Kavramı
  - Analog
  - Dijital
- Arduino Analog ve Dijital Dünyası
4. Mantıksal kavramlar
- Değil – Not Kapısı
- Ve – And Kapısı
- Veya – Or Kapısı
5. Koşul Durumları
- Mantıksal Değerler
  - “=” Kavramı
  - “>” Kavramı
  - “<” Kavramı
  - “==” Kavramı
  - “=>” Kavramı
  - “=<” Kavramı
- if – else Yapısı
6. Döngü Yapıları
- Döngü Yapılarına Giriş
- For Yapısı
- While Yapısı
- Break Yapısı
- Continue Yapısı

7. Fonksiyonlar
- Metot Kavramı
- Fonksiyonlarda “Return” Kavramı
- Fonksiyonlarda Parametre Türleri

8. Arduino ile Gömülü Fonksiyonlar
- Setup Fonksiyonu
- Loop Fonksiyonu
- Map Fonksiyonu

9. Kütüphane Kullanımı
- Kütüphane Kullanım Mantığı
- Arduino ile Kütüphane Kullanmak

10. IGU Arduino Deney Kiti
11. Uygulama - 1 “ LED Yakma – Söndürme ”
12. Uygulama - 2 “ Buton ile LED Yakma – Söndürme ”
13. Uygulama - 3 “ Buzzer Kontrol ”
14. Uygulama - 4 “ Buton İle Röle Kontrol ”
15. Uygulama - 5 “ Seven Segment Uygulaması ”
16. Uygulama - 6 “ Seri Port İle Seven Segment ”
17. Uygulama - 7 “ Seri Port İle Role Uygulaması ”
18. Robotik Uygulamalar ve Çalışmalar

1. Arduino ile Kodlama

Genelde Arduino nedir sorusundan sonra "Nasıl kullanacağım, neyi öğrenmeliyim?" konusunda bir kafa karışıklığı yaşayan; kodlamaya meraklı, robot çalışmalarına meraklı ve bir şeyler yapmam gerek diyen herkes için kolaylık olması için IGU TTO olarak bir Arduino öğrenim yol haritasını oluşturduk.

Baştan yanılmamanız için şunu belirtmek istiyorum. "Arduino Basittir ama kolay değildir"

Yıllarını elektronikte harcamış bir çok abimiz Arduino' da 1 satırlık komutu görüp, kendi yazdıkları 100 satır kod karşısında, hem dudak bükerler, hem küçümserler, hem de kıskanırlar[1].

Önemli olan işi yapmaksa Arduino bunu size basit yoldan sağlar ve prototipleme aşamasını hızlandırır..

- Kodlama – programlama nedir?

Kodlama = Bir metnin önceden belirlenmiş simgeler dizisine göre işaretlenmesidir. Programlama bilgisayara ya da elektronik devre ve mekanik sistemlerden oluşan düzeneklere bir işlemi yaptırmak için yazılan komutlar dizisinin bütünü veya bir kısmı olarak tanımlanır[2].

Kodlamanın Temeli: Algoritmadır.

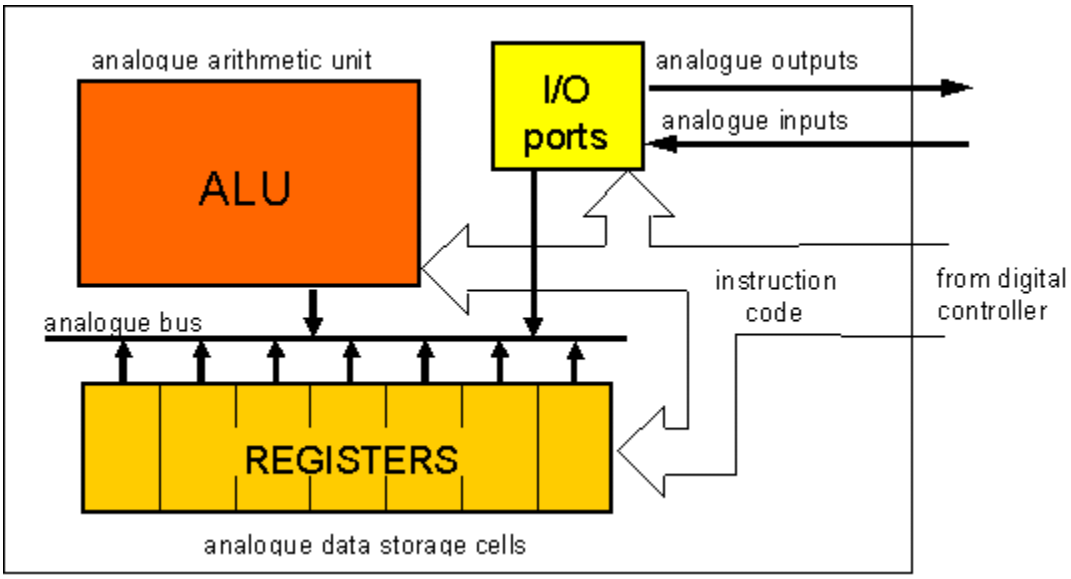
Algoritma; bir problemin ya da bir sorunun çözümü için ihtiyaç duyulan tüm işlemlerin mantıksal olarak, sıralı bir şekilde ifade edilmesidir. Algoritma kavramı programlama ya da yazılım geliştirmenin yanı sıra, günlük hayattaki aktiviteler, matematik vb. derslerde karşılaşılan sorunları daha kolaylıkla çözme olanağı sağlar.

Kodlama eğitimine direkt olarak C# veya JavaScript gibi geleneksel kodlama dilleriyle karşılaştığımızda sıkılıp hevesimizi kaybedebiliriz.

Programlama Dillerinden en önemlileri; JAVA, C, PYTHON, PHP, VISUAL BASIC, JAVASCRİPT, R, GO, ... , PASCAL, DELPHI, MATLAB, C# v.b. şekilde çoğaltabiliriz.

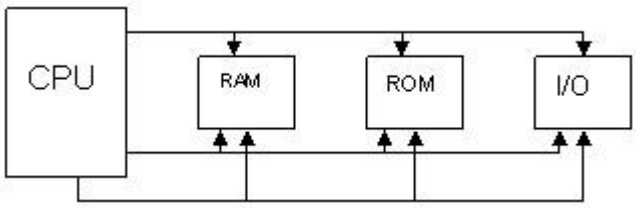
- Mikroişlemci – mikrodenetleyici nedir?

  - Mikroişlemci

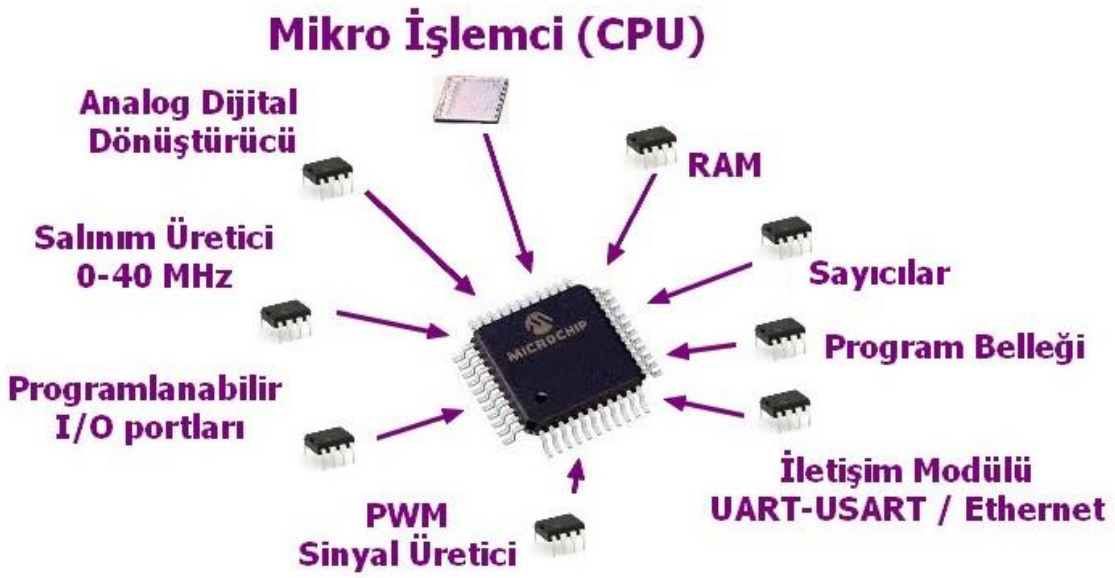


Sadece CPU içerir, RAM, ROM, I/O, timer vb. ayrıca bağlanır. Tasarımcı ROM, RAM ve I/O portlarının büyüklerini kendisi belirler ve ona göre tasarımı gerçekleştirir. Mikro işlemcinin temel işlevi, işlenen ve kullanılan verileri birimler arasında iletmek, iletilen verileri işlemek, verileri bir durumdan diğerine çevirmek ve verileri saklamak.

  - Mikrodenetleyici



Mikro denetleyicilerin içerisinde, mikro işlemci bulunmakla birlikte RAM, ROM, Program Belleği, Sayıcılar, İletişim Modülü (UART-USART / Ethernet) , PWM Sinyal Üretici, Programlanabilir I/O portları, Salınım Üretici 0-40 MHz, Analog Dijital Dönüştürücü bulunur.

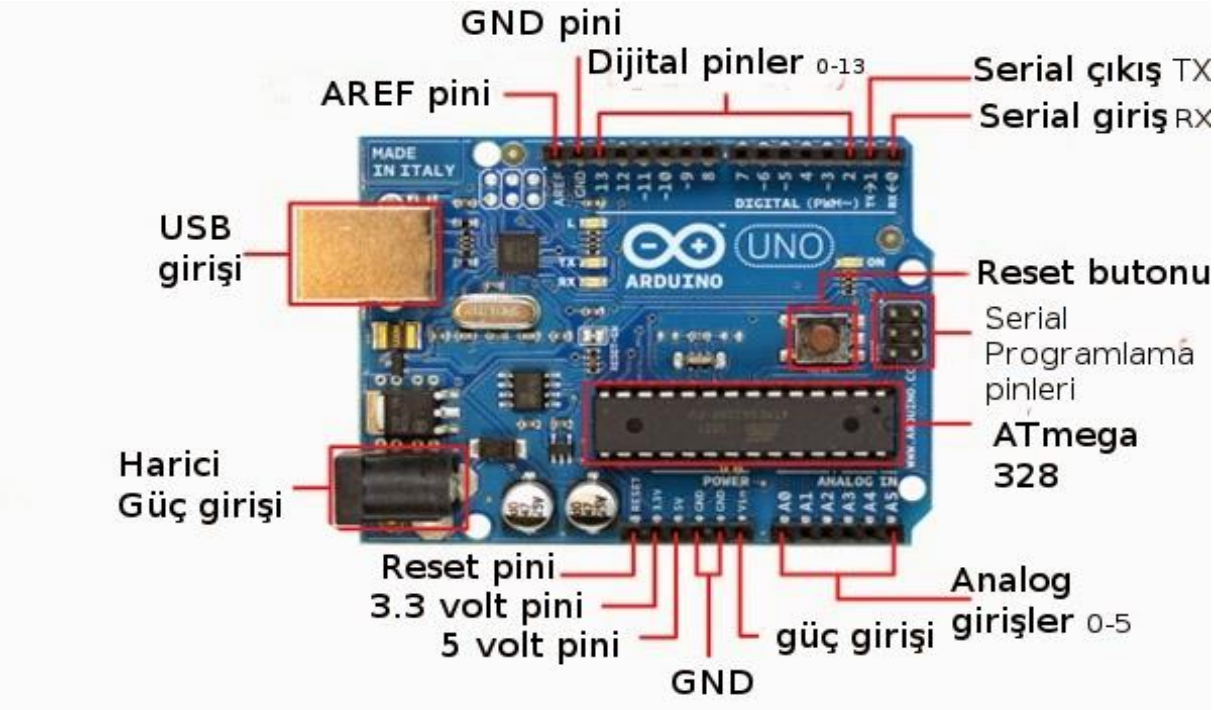


Mikro Denetleyici

CPU, RAM, ROM, I/O, timer vb. birimler tek bir çip içerine konulmuştur.
Dahili ROM, RAM ve I/O portları mevcuttur, ayrıca bir tasarım gerektirmez. Mikroişlemci aritmetik ve mantıksal işlemler yapabilen ve bu işlemlerin sonucuna göre çalışmasını yönlendirebilen tümleşik bir devre elemanıdır. Mikrodenetleyici, içerisinde RAM, ROM ve giriş-çıkış arabirimini içeren bir mikroişlemcidir.

- Arduino IDE ve Atmel Yapısı

1- Dosya menüsü
2- Düzenle menüsü
3- Derleme
4- Derle-Yükle
5- Seri Port
6- Mikrodenetleyici Tipi
7- Araçlar Menüsü



2. Gerekli ortamların kurulması

- Arduino

  https://www.arduino.cc/en/main/software

- Fritzing

  http://fritzing.org/download/

- VS Atmel Studio

  http://www.atmel.com/microsite/atmel-studio

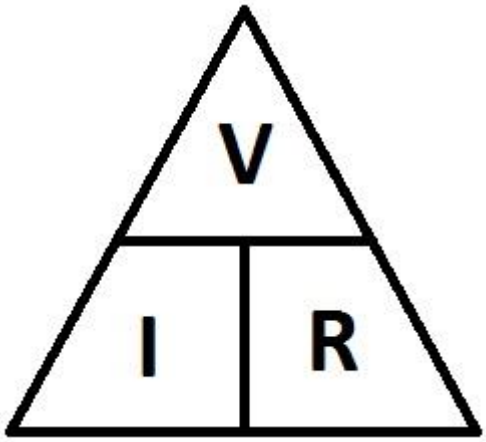
- Kurulum Notları

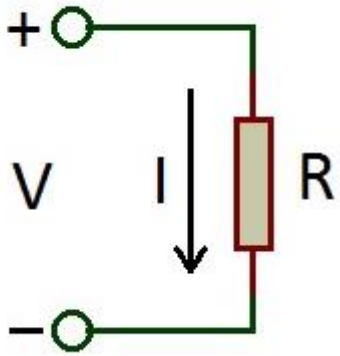
3. Dijital Dünyanın Gerçekliği

- “1” ve “0” Kavramı

  Dijital Dünyada sadece ve sadece “var” ve “yok” kavramı vardır. Bazı bilimkurgu filmlerin afişlerinde ya da dijital çağ kavramının olduğu resimlerde, onlarca belki de yüzlerce 1 ve 0 sayıları görülür. Bunun sebebi, dijital dünyada “var” ve “yok” kavramlarının 1 ve 0 sayıları ile ifade edilmesidir. Ancak 1 ve 0 sayısı sadece “kabul” olarak vardır. Her şey ama her şey bu iki sayıya dönüştürülerek dijital dünyada yer bulabilir. Aksi takdirde bu âleme dâhil edilemez.

- Gerilim – Volt Veri Kavramı

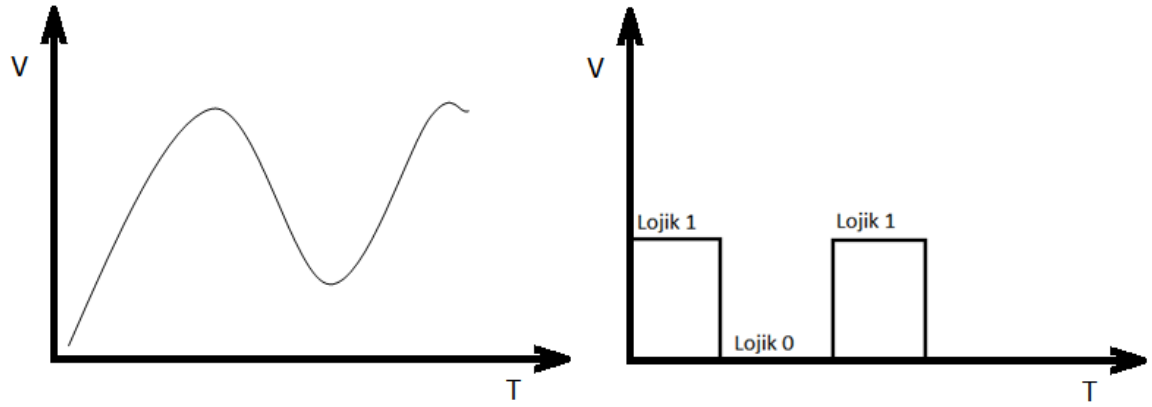




Bir direnç için potansiyel farkını ve üzerinden geçen akımı göstermektedir. Eğer dirence pozitif bir voltaj uygulanırsa, ok ile gösterilen yönde pozitif bir akım geçer. Bir dirence pozitif voltaj uygulanması, ++ ile gösterilen tarafa uygulanan voltajın, –– ile gösterilen taraftakinden daha yüksek olduğu anlamına gelir. Burada **önemli olan, ++ ve –– uçlar arasındaki voltaj farkıdır**

$$V = I * R$$

- Analog - Dijital Veri Kavramı



Analog, bize her değişimin olduğu gibi yansıtılmasını sağlar. Dijitalde olduğu gibi iki farklı değerden bahsedilmez. Analog sinyaller herhangi bir değeri alabilirler.

- Arduino Analog ve Dijital Dünyası

Eğitimlerimizde kullandığımız Arduino UNO kartında:
0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13 olmak üzere 14 adet dijital pin bulunuyor.

Bu pinler HIGH(yüksek) ve LOW(alçak) değerlerini alıyor ve veriyor.
Pinleri giriş olarak kullanmak için void setup içerisinde
pinMode(pinno, INPUT);
pinMode(pinno, OUTPUT); // çıkış olarak kullanmak için kodlarını void loop içerisinde digitalWrite(); ve digitalRead(); kodlarını kullanıyoruz.

Eğitimlerimizde kullandığımız Arduino UNO kartında A0, A1, A2, A3, A4, A5 olmak üzere 6 adet analog pin bulunuyor. Bu pinler 0 ile 1023 arasında toplam 1024 adet sayısal değer alıyor ve veriyor.

Pinleri giriş olarak kullanmak için void setup altında pinMode(pinno, INPUT); çıkış olarak kullanmak için pinMode(pinno, OUTPUT); ve void loop altında genel olarak analogWrite(); ve analogRead(); kodlarını kullanıyoruz.

4. Mantıksal kavramlar

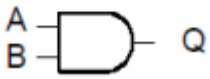
- Değil – Not Kapısı

(a) Sembolü

Giriş	Çıkış
A	Q
0	1
1	0

(b) Doğruluk Tablosu

- Ve – And Kapısı



(a) Sembolü

Girişler		Çıkış
A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

(b) Doğruluk Tablosu

- Veya – Or Kapısı

(a) Sembolü

Girişler		Çıkış
A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1

(b) Doğruluk Tablosu

5. Koşul Durumları

- Mantıksal Değerler

  - “=” Kavramı
    Atama operatörüdür. “ int a = 10, b = 20; ”

  - “>” Kavramı
    Büyüktür operatörüdür. “ b > a ”

  - “<” Kavramı

    Küçüktür operatörüdür. “ a < b ”

  - “==” Kavramı
    Eşit mi ? operatörüdür. “ a == b ”

  - “=!” Kavramı
    Eşit değil mi ? operatörüdür. “ a =! b ”

  - “=>” Kavramı

    Büyük ve eşit mi? operatürüdür. “ b => a ”

  - “=<” Kavramı
    Küçük ve eşit mi? Operatörüdür. “ a =< b ”

- if – else Yapısı

  İf ‘eğer’ anlamına gelmektedir. Karşılaştırma yapmak için sıklıkla kullanılan operatörlerin başında gelir.

```
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.

6. Döngü Yapıları

- Döngü Yapılarına Giriş

- For Yapısı

Genellikle artış veya azalış sayacı olarak kullanılır. Bir değerden başka bir değere kadar artıp ya da azalarak son değerden sonra döngüyü bitiren operatördür.

```
for (int x = 1; x < 100; x = x+2)
{
println(x);
}
```

Bu örneğimizde x değerini birden başlatarak 2 şer arttırımla 100 den küçük olana kadar tüm değerlerin çıktısını aldık. 1,3,5,7,9,….97,99

- While Yapısı

While döngüsü bir şart sağlanıyor *iken* sürekli içindeki komutları çalıştırır. Kelime anlamı olarak "olduğu müddetçe" anlamı çıkar. Yapısı şöyledir:

```
while (koşul)  // buradaki koşul anlamı genel bir tanım…..
{
   //komut 1
   //komut 2
   //…
   //komutlar
}
```

- Break Yapısı

Bir döngü bloku içinde verilen bir *break* komutu, döngünün hemen o anda bitirilmesine yol açar. Program akışı döngüden sonra her ne varsa onunla devam eder. Sözgelişi, aşağıdaki program kullanıcı -99 girene kadar, verilen sayıları toplama ekler.

- Continue Yapısı

continue komutu da, aynı *break* gibi, sadece bir döngü içinde ve bir *if* şartı altında mânâ ifade eder. *continue* döngü blokunun işlemesini yarıda keser ve başa döner. *break*'den farkı, programın döngünün dışına çıkmaması, ama döngünün başına dönmesi ve tekrar başlatmasıdır. Bu arada döngü şartının doğru olup olmadığı da kontrol edilir.

--------Örnek Uygulamalar-------

Uygulama – 1. "Dış ortamdan arduinoya gelen verilerin kullanıcıya tekrar bildirilmesi"

```
String gelen_veri = "";
boolean durum = false;


void setup()
{
 Serial.begin(9600);
 gelen_veri.reserve(200);
}


void loop()
{
 while (Serial.available())
 {
  char gelen_degerler = (char)Serial.read();
  gelen_veri += gelen_degerler;
  if (gelen_degerler == '\n')
  {
   durum = true;
  }
```

```
  if (durum)
  {
   Serial.println(gelen_veri);
   gelen_veri = "";
   durum = false;
  }

 }
}
```

Uygulama – 2. “Dış ortamdan arduino’ya gelen verilerin taranıp işlemlerin gerçekleştirilmesi”

```
String gelen_veri = "";
boolean durum = false;
void setup()
{
 Serial.begin(9600);
 gelen_veri.reserve(200);
}
void loop()
{
 while (Serial.available())
 {
  char gelen_degerler = (char)Serial.read();
  gelen_veri += gelen_degerler;
  if (gelen_degerler == '\n')
  {
   durum = true;
  }
```

```
  if (durum)
  {
   if(gelen_veri.toInt()>10)
   {
    Serial.println("Gelen Veri 10'dan büyük");
   }
   else if(gelen_veri.toInt()<10)
   {
    Serial.println("Gelen Veri 10'dan küçük");
   }
   else if(gelen_veri.toInt()==10)
   {
    Serial.println("Gelen Veri 10'a eşit");
   }
   else
   {
    Serial.println("Gelen Veri Tanimsiz");
    gelen_veri = "";
    durum = false;
   }

   gelen_veri = "";
   durum = false;
  }
 }
}
```

7. Fonksiyonlar

- Metot Kavramı

  Nesneye yönelik programlama dillerinde genellikle fonksiyonlar ***"metot"*** olarak isimlendirilirler. Metot ve fonksiyon olarak ifade edilecek kavramlar aynı anlamda kullanılacaktır.
  Programın herhangi bir yerinde kullanmak için belirli bir işi yerine getirmek amacıyla tasarlanmış alt programlara ***metot*** denir.

- Fonksiyonlarda "Return" Kavramı
  Fonksiyonların girdilerine parametreler ya da argümanlar denir. Bir fonksiyon bu parametreleri alıp bir işleme tabi tutar ve bir değer hesaplar. Bu değer, çıktı veya geri dönüş değeri (return value) olarak adlandırılır. Unutmayın ki, bir fonksiyonun kaç girişi olursa olsun sadece bir çıkışı vardır.

- Fonksiyonlarda Parametre Türleri

C fonksiyonları, işlevleri farklı olan iki tür değişken kullanır. Fonksiyon adını izleyen () parantezi içine yazılanlara parametre ya da argüman (parameter, argument) denilir. Parametreler birer değişkendir. Hiç parametre olmayabileceği gibi bir ya da daha çok parametre olabilir. Parametre yoksa () parantezi içine hiçbir şey yazılmaz ya da void yazılır.

8. Arduino ile Gömülü Fonksiyonlar

Arduino projenizi ilk açtığınızda karşınıza iki fonksiyon çıkar. Bunlar setup ve loop fonksiyonlarıdır.

- Setup Fonksiyonu

Setup fonksiyonu, kod çalışmaya başladığında Arduino'nun ilk olarak okuduğu yerdir. Arduino bu kısmı okuduktan sonra diğer kısımları okumaya başlar. Bu kısım sadece bir kere okunur ve program esnasında yeniden okunmaz. Bu alanda, pinlerin çalışma modları gibi önemli ve bir kere yapılması yeterli olacak ayarlamalar yapılır.

- Loop Fonksiyonu

Loop fonksiyonu, setup fonksiyonu okunduktan sonra okunur. Bu bir ana fonksiyondur ve yapılmasını istediğiniz görevler buraya yazılır. Loop fonksiyonu, sonsuz döngü şeklindedir, yani buradaki görevler tamamlandığında, program tekrar başa dönerek işlemleri yeniden yapar. Bu döngü, Arduino çalıştığı sürece devam eder.

- Map Fonksiyonu

Map fonksiyonu temel olarak belirli bir değer aralığında olan tam sayı değerlerini başka bir değer aralığına çevirir. Örneğin Analog pinlerini kullanarak dışarıdan bir değer alıyorsanız buradan gelecek değer 0-1024 arasında olacaktır. Siz bunu map fonksiyonunu kullanarak istediğiniz aralığa oranlayabilirsiniz. Örnek kullanımı şu şekilde;

int sonuc=map(x, y, z, t, u);

Burada;

x, mevcut değerinizi(değişken olabilir)

y, mevcut değerinizin alabileceği en küçük değeri

z, mevcut değerinizin alabileceği en yüksek değeri

t, yeni değerinizin almasını istediğiniz en küçük değer

u, yeni değerinizin almasını istediğiniz en büyük değer

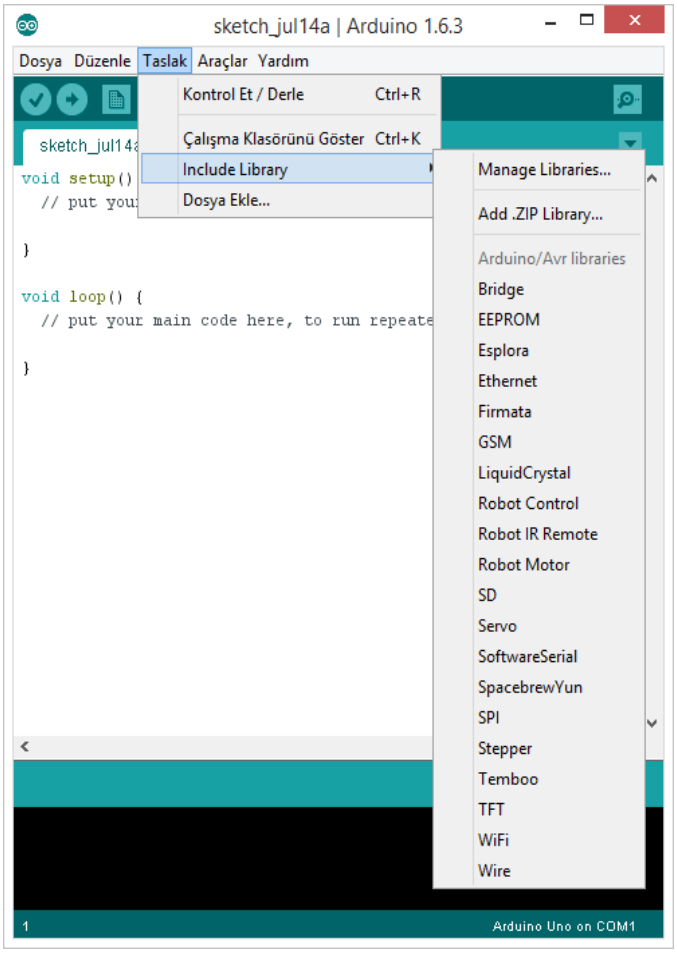
9. Kütüphane Kullanımı

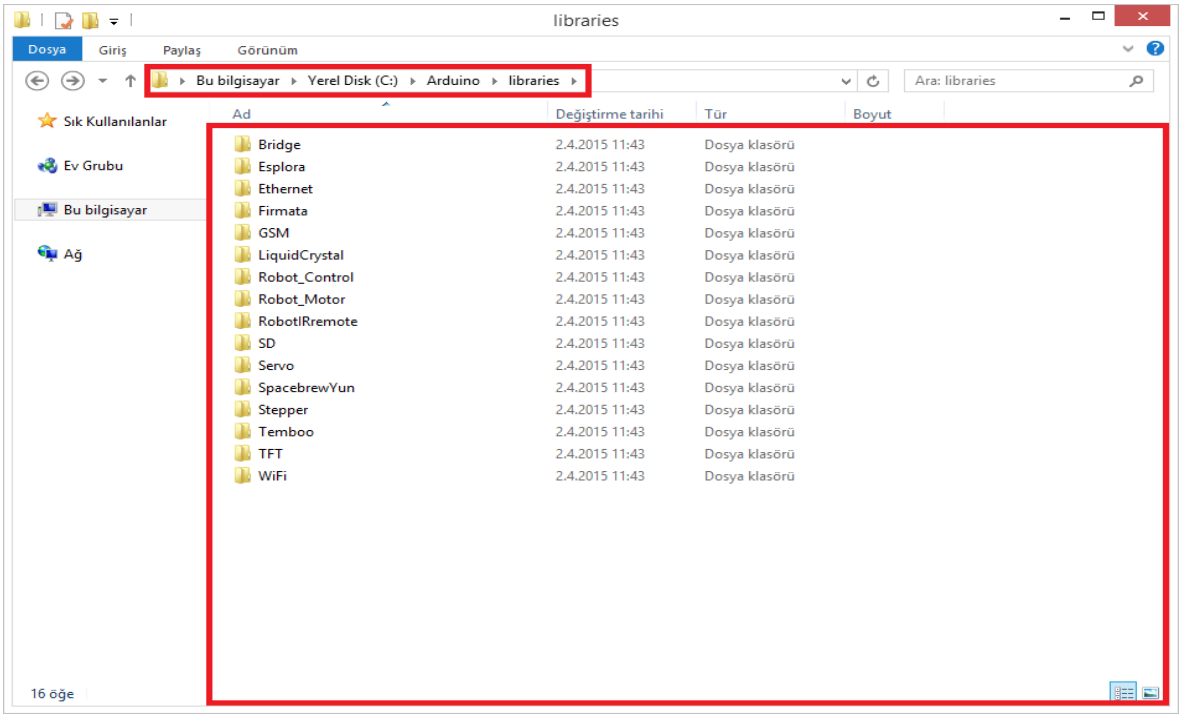
Yeni bir kütüphane modülü nasıl oluşturulabiliriz ve açık kaynak kütüphane ekosistemine nasıl katkıda bulunabiliriz? İyi bir geliştirici asla Amerika'yı yeniden keşfetme işine girmez. Var olanı geliştirmeyi amaçlar. Bu bağlamda Arduino geliştiricileri, kendi uygulamalarında özgürce kullanabileceği, canlı ve güçlü bir açık kaynak kütüphane ekosistemine sahiptir. Geliştiriciler, bu kütüphaneleri diledikleri gibi kullanabilir. Bunun yanında gelişimlerine destek olabilir. Biz de bu yazımızda yeni bir kütüphane modülü nasıl oluşturulabiliriz, bunu nasıl yayınlayabiliriz ve bu açık kaynak kütüphane ekosistemine nasıl katkıda bulunabiliriz konularına değineceğiz.

- Kütüphane Kullanım Mantığı

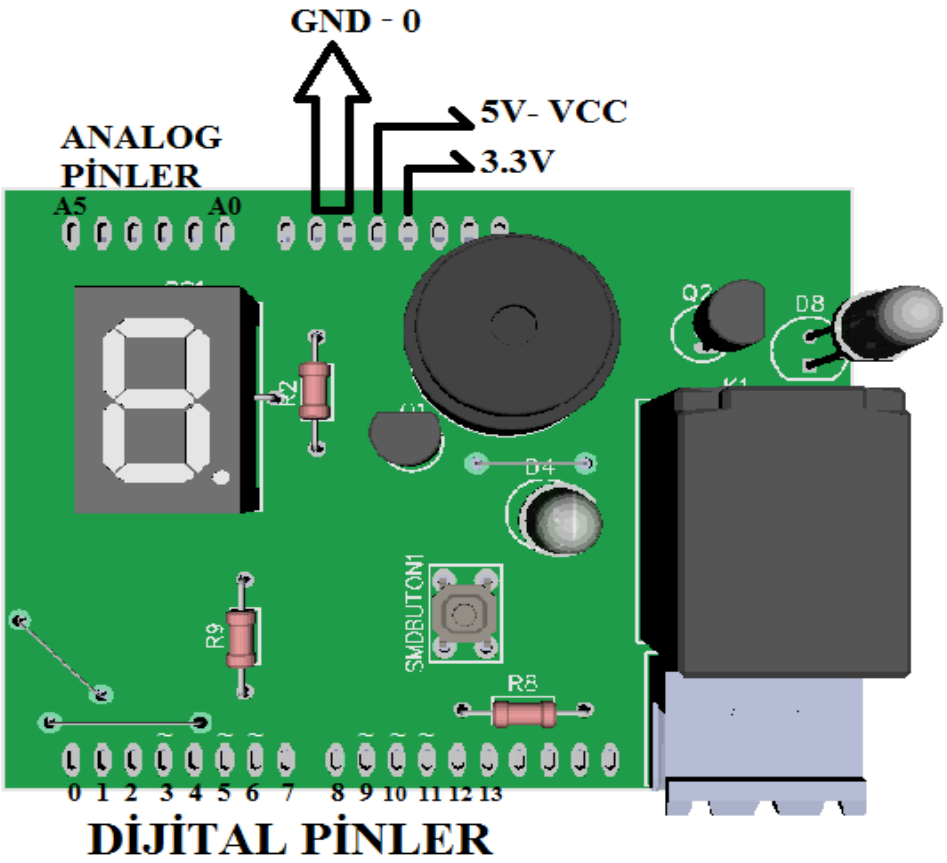
Arduino'da kütüphaneler kullanarak belli kalıptaki yazılım cümleleri ile kısa zamanda daha çok iş yapabiliriz. Bunun için arduino kütüphanelerini kullanıyoruz. Eğer yazılım bilginiz iyiyse programlanabilir bir devre elemanı için kendi Arduino kütüphanenizi oluşturup kullanabilirsiniz.

- Arduino ile Kütüphane Kullanmak





10. IGU Arduino Deney Kiti



11. Uygulama - 1 " LED Yakma – Söndürme "

```
const int ledPin =  13;      // tanımlı olan led pini

void setup()                 // sistem ayarları
{
  pinMode(ledPin, OUTPUT);  // led pininin çıkış olarak ayarlanması
}

void loop()      // loop döngüsü
{
  digitalWrite(ledPin, HIGH);   // ledin ışık vermesi durumu
  delay(2000);                  // ledin ışık vermesi durumu 2 saniye sürmesi

  digitalWrite(ledPin, LOW);    // ledin ışık sönmesi durumu
  delay(1000);                  // ledin ışık sönmesi durumunun 1 saniye sürmesi
}
```

12. Uygulama - 2 " Buton ile LED Yakma – Söndürme "

```
const int butonPin = 12;    // tanımlı olan buton
const int ledPin =  13;      // tanımlı olan led
int butonDurum = 0;         //  butonun durumu için oluşturulmuş değişken.

void setup()
{
  pinMode(ledPin, OUTPUT);  // tanımlı led'in çıkış olarak ayarlanması
  pinMode(butonPin, INPUT); // tanımlı butonun diriş olarak ayarlanması
}

void loop() // loop döngüsü
{
  butonDurum = digitalRead(butonPin);  // butonun durumunun sorgusu
  if (butonDurum == HIGH) // if döngüsü ile buton durumununun kontrolü
  {
        digitalWrite(ledPin, HIGH); // eğer butona basıldıysa led ışık verir
  }
  else
  {
    digitalWrite(ledPin, LOW); // butona basılı değilse led sönme durumunda
kalır.
  }
}
```

13. Uygulama - 3 “ Buzzer Kontrol ”

```
int Buzzer = 11; // tanımlı olan buzzer pini

int ton_sayisi = 9; // buzzer için

int buzzer_tonlar[] = {100, 150, 200, 250, 300, 350, 400, 450, 500};
// buzzer ses tonları

void setup() // kurulum ayarları
{

}

void loop() // loop döngüsü
{
 for (int i = 0; i < ton_sayisi; i++)    // döngü
 {
 tone(Buzzer, buzzer_tonlar[i]);      // buzzer ulaşan data
 delay(600);                          // buzzere ulaşan datayı 600ms'ye çal.
 }
 noTone(Buzzer);                      // buzzer tonları bitince buzzeri sustur.
}
```

14. Uygulama - 4 “ Buton İle Röle Kontrol ”

```
const int butonPin = 12;    // tanımlı olan buton
const int rolePin =  7;     // tanımlı olan röle

int roleDurum = 0;        // butonun durumu için oluşturulmuş değişken.

void setup()
{
  pinMode(rolePin, OUTPUT); // tanımlı led'in çıkış olarak ayarlanması
  pinMode(butonPin, INPUT); // tanımlı butonun diriş olarak ayarlanması
}

void loop() // loop döngüsü
{
  roleDurum = digitalRead(butonPin); // butonun durumunun sorgusu
  if (roleDurum == HIGH)  // if döngüsü ile buton durumununun kontrolü
  {
    digitalWrite(rolePin, HIGH); // eğer butona basıldıysa röle aktif olur.
  }
  else
  {
    digitalWrite(rolePin, LOW); // butona basılı değilse röle pasif durumunda kalır.
  }
}
```

15. Uygulama - 5 " Seven Segment Uygulaması "

// Gerekli kütüphane dosyasını C:\ Users \BilgisayarınızınİsmiDocuments
// \Arduino \libraries klasörü içerisine kopyalayın.

```
#include <AH_74HC595.h>
#define SER_Pin   4      //Serial data input
#define RCLK_Pin  3    //Register clock
#define SRCLK_Pin 2   //Shift register clock
// Seven segment ayarları
AH_74HC595 seven_segment(SER_Pin, RCLK_Pin, SRCLK_Pin);
void setup()
{
 seven_segment.clear();  // seven segment üzerinde oluşturulan karakteri                    //
 temizledik.
}
void loop() // loop döngüsü
{
 for (int i=1;i<10;i++) // seven segment üzerinde tek sayıların görüneceği döngü
 {
   seven_segment.showNumber(i);
   delay(500);
 }
 seven_segment.clear();      // seven segment üzerinde oluşturulan karakteri temizledik.
 delay(1000);
 seven_segment.setPoint(true);     // seven segment üzerinde bulunan nokta aktif edildi.
 delay(1000);
 seven_segment.setPoint(false);    // seven segment üzerinde bulunan nokta pasif edildi.
}
```

16. Uygulama - 6 " Seri Port İle Seven Segment "

// gerekli kütüphane dosyasını

// C:\Users\BilgisayarınızınİsmiDocuments\Arduino\libraries

// klasörü içerisine kopyalayın.

```
#include <AH_74HC595.h>
#define SER_Pin   4   //Serial data input
#define RCLK_Pin  3   //Register clock
#define SRCLK_Pin 2   //Shift register clock
// Seven segment ayarları
AH_74HC595 seven_segment(SER_Pin, RCLK_Pin, SRCLK_Pin);
void setup()
 {
 Serial.begin(9600); // seri port bağlantı noktası hızı ayarlaması
 seven_segment.clear();  // seven segment üzerinde oluşturulan karakteri
temizledik.
}
void loop()
{
 while(Serial.available());
{
 int data = Serial.read();
 switch(data)
   {
  case '0':
   {
   seven_segment.setValue(B00111111);
   delay(500);
   break;
   }
  case '1':
   {
   seven_segment.setValue(B00000110);
   delay(500);
   break;
   }
  case '2':
   seven_segment.setValue(B01011011);
   delay(500);
   break;
  case '3':
   seven_segment.setValue(B01001111);
```

```
    delay(500);
    break;
   case '4':
    seven_segment.setValue(B01100110);
    delay(500);
    break;
   case '5':
    seven_segment.setValue(B01101101);
    delay(500);
    break;
   case '6':
    seven_segment.setValue(B01111101);
    delay(500);
    break;
   case '7':
    seven_segment.setValue(B00000111);
    delay(500);
    break;
   case '8':
    seven_segment.setValue(B01111111);
    delay(500);
    break;
   case '9':
    seven_segment.setValue(B01101111);
    delay(500);
    break;
   default:
   {
    //seven_segment.clear();
    // seven segment üzerinde oluşturulan karakteri temizledik.
    //delay(1000);
    seven_segment.setPoint(true);
    // seven segment üzerinde bulunan nokta aktif edildi.
    delay(1000);
    seven_segment.setPoint(false);
    // seven segment üzerinde bulunan nokta pasif edildi.
    delay(1000);
   break;
   }
  }
 }
}
```

17. Uygulama - 7 " Seri Port İle Role Uygulaması "

```
const int butonPin = 12;   // tanımlı olan buton
const int rolePin =  7;    // tanımlı olan röle
int roleDurum = 0;   // butonun durumu için oluşturulmuş
//değişken.
String data;

void setup()
{
 Serial.begin(9600); // seri port bağlantı noktası hızı ayarlaması
 pinMode(rolePin, OUTPUT); // tanımlı led'in çıkış olarak
//ayarlanması
 pinMode(butonPin, INPUT); // tanımlı butonun diriş olarak
//ayarlanması
}

void loop()
{
 while(Serial.available())
 {
 data = char(Serial.read());
 if (data=="1")  // if döngüsü ile buton durumununun kontrolü
  {
   digitalWrite(rolePin, HIGH); // eğer butona basıldıysa röle aktif
//olur.
   Serial.println("Role Aktif"); // Seri port üzerinden görünecek
//mesaj
  }
 else if(data=="0")
  {
   digitalWrite(rolePin, LOW); // butona basılı değilse röle pasif
//durumunda kalır.
   Serial.println("Role Pasif"); // Seri port üzerinden görünecek
//mesaj
  }
 }
}
```

18. Robotik Uygulamalar ve Çalışmalar

Youtube ➔ Altubilisimrobotik

https://www.youtube.com/channel/UC4Jb8uWXkV-3cwejcIdvFrg