Gömülü Sistemler

- Mini bilgisayarlarda yada mikrodenetleyiciler üzerinde çalışan sistem
- Az sayıda atanmış görev

Normal Programlamadan farkı

- Kısıtlamalar çok daha önemli
- Güvenlik, performans sorunları
- Gerçek zamanlı
- Donanımı da göz önünde bulundurmak gerekli.

Gömülü Denetleyiciler

genel amaçlı işlemci
esneklik fazla

ASIP	düşük güç yüksek performans
FPGA	yüksek hız performans
ASIC	fazla performans düşük güç
DSP	sinyal işleme için ayrılmış

Mikrocontroller (— — denetleyici)

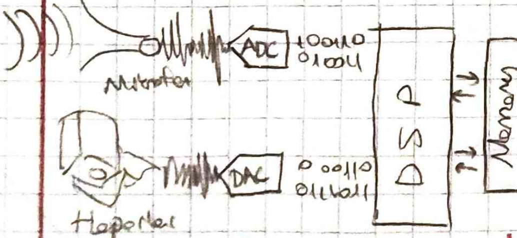
- Tek bir görevi yönetmek için
- Tasarım donanım maliyeti düşük
- Değiştirilmesi kolay
- Çalışırken az güç tüketir (CMOS)
- CPU, RAM, ROM, I/O dahil,

Mikroprocessor (— — işlemci)

- Büyük uygulamalar için kullanılır
- Tasarım ve donanım maliyeti yüksek
- Değiştirilmesi daha kolay değil
- Güç tüketimi yüksek
- RAM, ROM, I/O içermez
- Çevresel cihazlara bağlanmak için pinleri kullanılır

DSP (Digital Signal Process)

Dijitalleştirilmiş ses, video, sıcaklık basınç veya konum gibi analog sinyalleri alır ve matematiksel olarak değiştirir.



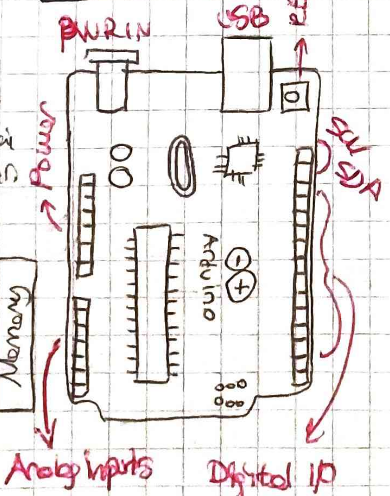
Hoparlör

- Program Belleği
- Veri belleği
- İşlem ahtapotu
- I/O

FPGA (Field Programmable Gate Arrays)

Yapılandırılabilir mantık kapıları içeren cihazlardır. Üretimden sonra istenildiği gibi yeniden tasarlanıp programlanabilir.

DSP	Orta fiyat	Sinyal uygulamaları	Orta (Matematik gerektirir) zorluk	Sinyal uygulamalarında mükemmel	Orta Güç
Mikrocontroller	Ucuz	Düşük güç, Çevre birimi ihtiyacı varsa	Kolay zorluk	Fazla çalıştırmak fazla hız	En düşük güç
FPGA	Pahalı	Yüksek işlem hızı	Zor	En hızlısı (Paralel)	Yüksek güç



OHM LAW
$V = I \cdot R$

NOT: Gerilim ölçülürken paralel bağlanır.
Akım ölçülürken seri bağlanır.

Breadboard. devre yaparken kolaylık sağlayan deliklerden oluşan iletim ortamı (masa)

Kütüphane Ekleme

#include <kütüphaneadı.h>

Genel görünüm

```
void setup()
{
// Burası ilk açılışta bir kere çalışır
ilk ve genel ayarlar yapılır.
}
void loop()
{
// Bu fonksiyon sonsuza kadar çalışır
Yapılacak işler belirtilir
}
```

Basit komutlar

pinMode (X, Y);

X: Y durumundan etkilenecek olan pin numarası.
Y: INPUT / OUTPUT — Seçilen pinin program boyunca giriş mi çıkış mı olacağı seçilir.

analogWrite(PinNum, Deger);

PinNum: Değer'de yazan veriye göre işlem yapılacak pin Numarası girilir.
Deger: 0 ile 255 arasında değer verilebilir.
0 → 0V
255 → 5V

PWM sinyalini üretir.

digitalWrite(PinNum, Y);

Y: LOW, HIGH değerlerinden birini alır

LOW	0V
HIGH	5V

state = digitalRead(pinNum);

pin numarasındaki pin'de bir dijital okuma yapar. 5V'ta yakın değerler için 1, TRUE, HIGH döndürür. Diğerleri için 0, FALSE, LOW döndürür.

state = analogRead(pinMode);

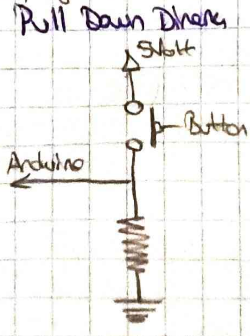
pin numarasındaki pin'den gelen Volt verisini default olarak 0-1023 arasında bir değere dönüştürür.

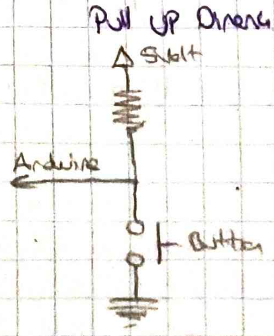
0V	0
5V	1023
2,5V	512
⋮	⋮

delay(milisecond);

milisecond kısmına yazılan sayı kadar programın çalışmasını durdurur.

Pull Down Direnç — Pull Up Direnç



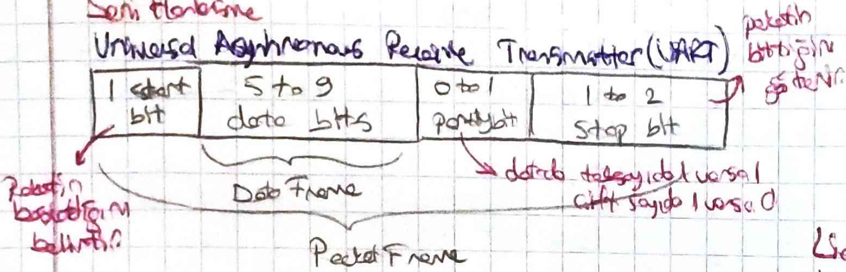


Pull Down: Basılı (+) "1", Değil (−) "0"

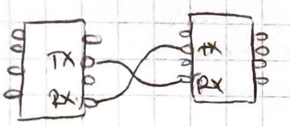
Pull Up: Basılı (−) "0", Değil (+) "1"

Seri Haberleşme

Universal Asynchronous Receive Transmitter (UART)



UART1 UART2



Baudrate
9600, 1200, 2400, 4800
19200, 38400, 57600
115200
bps (bit per second)

ARDUINO ÜZERİNDE

pin0 RX
pin1 TX

NOT: Dosya gönderimi sırasında 0 ve 1 nolu pinlere bağlantı yapılmaz.

USB bağlantısı aynı pinlere bağlıdır.

Kodlar

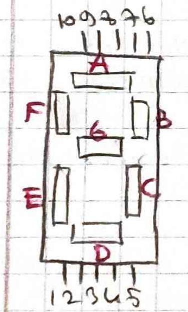
Serial.begin(9600);
→ haberleşme hızı ayarlanır
→ Seri haberleşme başlatılır.

Serial.println("Merhaba");
→ Arkasından veri bu kısma yazılır
→ ekrana yazdırma fonksiyonu.

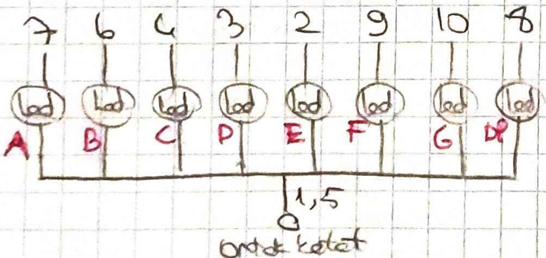
Serial.read(); bilgisayardan gelen karakteri dinle, oku

Serial.available(); bilgisayardan veri geldiği anda 1 gelmediği zamanlarda 0 olur.

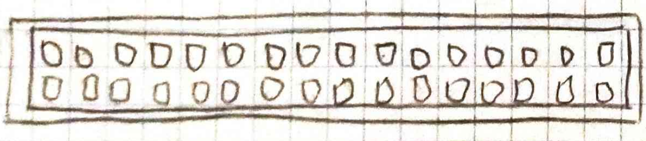
SEVEN SEGMENT



- 7 adet led kullanarak bazı karakterleri ekrana yazdırmada kullanılır.
- Ortak Anot ve ortak katot olmak üzere iki türü vardır.
- Ortak katot (tüm ledlerin katotları birleştirilmiş) anotlar için güç verilir.
- Ortak anod (tüm ledlerin anotları birleştirilmiş) katotlar için güç kesmek gerek.



2X16 LCD EKRAN



+5Volt ile çalışır.

#include <LiquidCrystal.h>

D4, D5, D6, D7 data pinleri

GND, R/W toprağa bağlanır.

RS 12

E(Enable) 11

V0 (kontrast) gerilim

LED+ LED- ekran ışığı

LiquidCrystal (12, 11, 5, 4, 3, 2);
pinler tanıtılıyor

LiquidCrystal.begin(16, 2);
LCD ekranı başlatır. 16 sütun 2 satır

LiquidCrystal.print("Hello");
Kursor 0,0 noktasından başlar yazı yazmaya

LiquidCrystal.clear();
Ekranı temizler.

LiquidCrystal.setCursor(10,0)
Kursorun bulunduğu konumu 10,0 noktasına getirir.

Analog Sinyal Üretme

PWM (Pulse Width Modülation)

Duty (Görev) sinyalinin aktif olduğu %

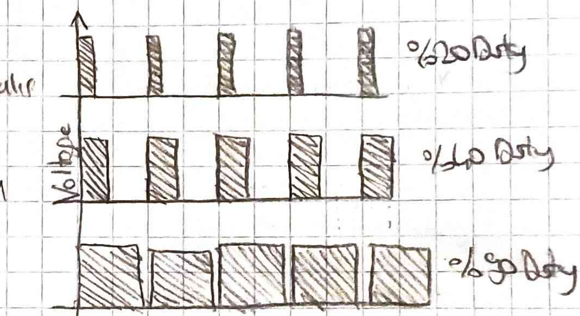
Çoğu PWM pini 490KHz

Sus 6 pinleri 980KHz periyoda sahiptir.

Periyodu 256 birime bölünür.

5V (%100 PWM) 255 değeri

0V (%0 PWM) 0 değeri



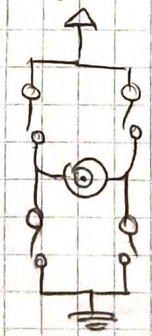
Arduinodaki ~ işaretli pinler analogWrite fonksiyonu ile PWM yapabilir.

DC MOTORLAR

Yüksek akım çeken, güç harcayan motorlardır.

Arduino Uno'dan en fazla 40mA 5V alt gönderilebilir.

DC motorların çalıştırılması için sürücü devrelere ihtiyaç vardır.



PWM ile dc motorun hız kontrolü yapılabilir.

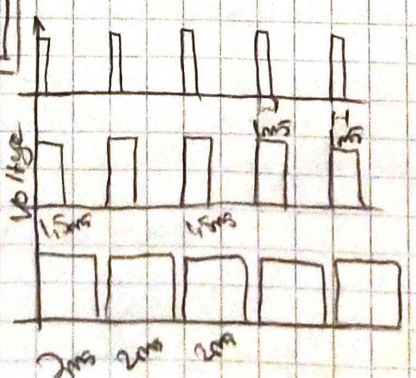
SERVO MOTORLAR

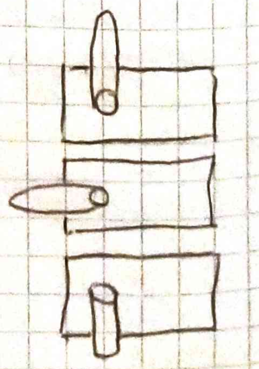
Periyodu 20ms dan ve 1ms ve 2ms ile değişen pulslarla servo motorun açısı ayarlanır.

1ms puls 0 derece

1,5ms puls 90 derece

2ms puls 180 derece





Servo kodları

#include <Servo.h> → kütüphaneyi ekliyoruz

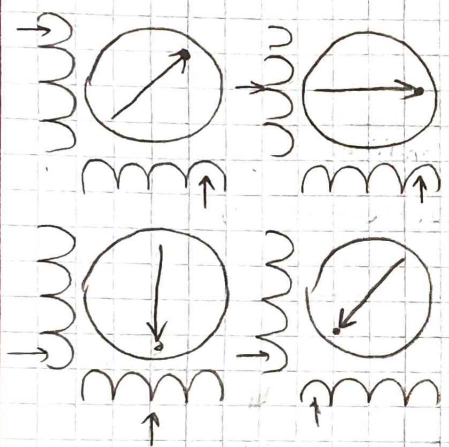
Servo srv; → isim veriyoruz

Servo.attach(9); bağlı olduğu pin numarasını belirtiyoruz.

Servo.write(90); Servonun 90 derece dönmesini sağlıyoruz

STEP MOTORLAR

Tur başına adım sayısı bulunmaktadır.



Index	1a	1b	2a	2b
1	1	0	0	1
2	1	1	0	0
3	0	1	1	0
4	0	0	1	1
5	1	0	0	1
6	1	1	0	0
7	0	1	1	0
8	0	0	1	1

clockwise ↓

Yarım adım ve tam adım modlarında çalıştırılabilir, Unipolar, Bipolar çeşitleri var. Bipolar olanlarda sargıların arasında ortak uç yok. Unipolarda ortak uç var. Ortak uç sargının tamamının çalışmasını engeller.

EEPROM

Elektrik kesildiğinde bilgiler silinmez. Unoda 1024 byte EEPROM bellek bulunmaktadır. 0-1023 adresleri arasında. EEPROM flash bellek yapısıdır.

#include <EEPROM.h>

EEPROM kütüphanesini kullanmak üzere sisteme ekledik

EEPROM.read(address);

address değişkenindeki adreste saklanan tutulan veriyi getirir.

EEPROM.write(address, value);

address kısmında belirtilen adrese value kısmında belirtilen değer saklanabilir.

KESME (INTERRUPT)

Mikroişlemci rutin işlemlerini gerçekleştirirken acilen işlenmesi gereken işlem gerçekleştiğinde, işleme ara verip acil işi yapmasına denilen isim.

ARDUİNODA kesmeler → Zaman kesmesi, Dış kesmeler

Zaman kesmesi

Belirli zaman aralıklarında işlemleri gerçekleştirmek için kullanılırlar

⊕ Sinyal örnekleme
⊕ İki olay arası zaman ölçme
⊕ Kare dalga gönderme
⊕ Seri iletişimde

Üç adet timer var Timer0,1,2

Timer0, Timer2 8bit (0-255)

Timer1 16 bit (0-65535)

Sayıcı bölme oranı 8,64,256,1024

Timer istenilen değere ulaştığında kesme yapar.

(timer speed(Hz)) = (Arduino clock speed 16MHz) / prescaler

Timer değeri hesaplanırken

[16.000.000Hz / (prescaler * istenilen frekans)] -1

→ 8,64,256,1024

Dış kesmeler

Bazı özel pinlerde voltaj değişimlerine göre çalışan kesmelerdir.

UNO 2,3

Low 0Volt High 5Volt

⊕ Döner kodlayıcı, rotary encoder
⊕ Tıklama sayıcı
⊕ Ses sensörü ...

Timer kesmeleri setup içine yazılmalıdır.

INTERRUPT SERVICE ROUTINS (ISR)

ISR'nin parametresi olmaz
Geriye veri döndürmezler
Mümkün olduğu kadar kısa ve hızlı olmalıdır.
millis() sayıcısı, interrupt içinde (ISR) saymayı durdurur
delay() ISR içinde çalışmaz
mikros() başlangıçta çalışır sonra bozulur.

volatile int state = LOW;

ISR ile global değişken tanımı

attachInterrupt(digitalPinToInterrupt(pin), ISR, mode)

pin: pinler UNO'da kullanılan kesme pinleridir 2,3

ISR: kesme alt programıdır fonksiyonun ismidir

mode: kesmenin ne zaman etkileneceği

Mode öntanımlıdır

4 farklı mode var

LOW kesme pini sıfır olduğunda
CHANGE kesme pini her değiştiğinde
RISING LOW → HIGH olduysa
FALLING HIGH → LOW olduysa

detachInterrupt(digitalPinToInterrupt(pin))

dış kesmeleri kapatır

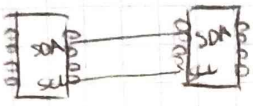
TIMER KESMELERİ

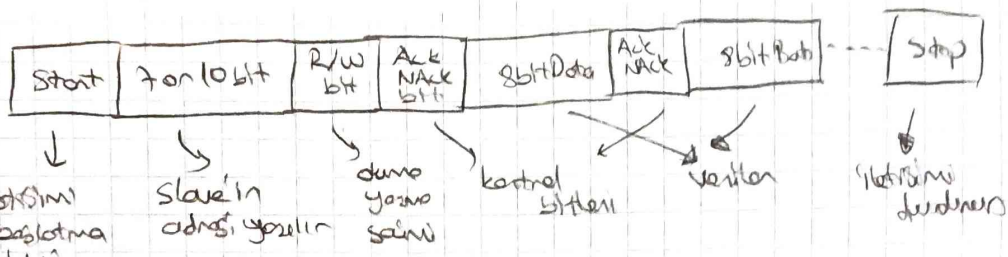
```
ISR(TIMER0_COMPA_vect)
{
  //kesme işlemi
}
```

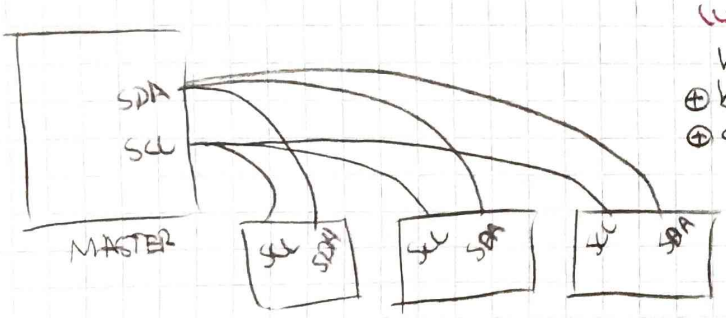
→ timer kesme fonksiyonunu çağırmada kullanılır.

cli() ve sei() Timer kesmelerini açıp kapatan fonksiyonlar

I²C Protocolü







`wire.begin();`
haberleşmeyi başlatır.
⊕ başa master
⊕ adresse slave

`wire.endTransmission();`
Verilerin yazılıp yazılmadığını kontrol eder

`wire.beginTransmission(slave_adresi);`
hangi slave ile haberleşeceğini seçebilir.
`wire.endTransmission();`

`wire.Read();` veri hattından gelen veriyi okur

`Wire.write();` yazılan parametreyi veri hattına aktarır

`wire.RequestFrom();`
Slave cihazından veri istenir
1. parametre - Slave adresi
2. parametre - kaç baytlık veri beklediği

`wire.onReceive(işlev Fonksiyon);`
Slave olarak tanımlı cihaza veri gelince istenen fonksiyon çalışmağa başlar.

SPI (Serial Peripheral Interface)

Bir master, çok slave
Data hattı iki tane (Duplex)
Data alıp gönderme aynı anda yapılabilir

MOSI (Master Out Slave In) (11)
MISO (Master In Slave Out) (12)
SCLK (Synchron Clock) (13)

SS → istediğin yere tek kanal
Slave adresini belirtir