# ROBÒTICA · 3r ESO

UNITAT 5: ENTRADES ANALÒGIQUES · PWM · SERVOMOTORS

Nom: _____ Curs/Grup: _____ Data: _____

Setmanes 8-9. Al final sabràs:

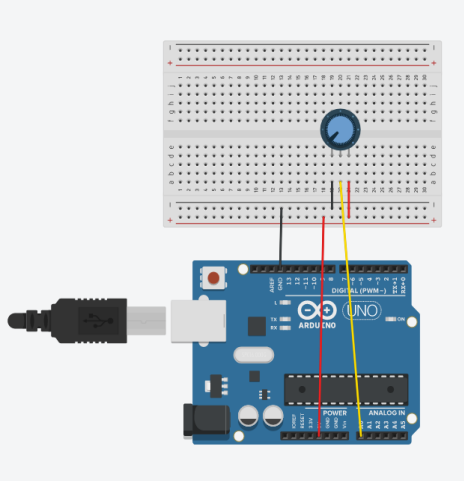
- Llegir sensors analògics amb analogRead (0-1023).
- Controlar sortides PWM amb analogWrite (0-255).
- Reescalar valors amb map().
- Posicionar un servomotor de rotació limitada amb Servo.h.

## 1. Entrades analògiques: analogRead

Les entrades analògiques (A0-A5 a la UNO) llegeixen tensions de 0 a 5 V i les converteixen en un valor entre 0 i 1023 (precisió de 10 bits).

> *analogRead(A0) — retorna un enter entre 0 (0 V) i 1023 (5 V)*

El potenciòmetre és el sensor analògic més senzill: és una resistència variable. En girar el seu eix obtenim un voltatge variable entre 0 i 5 V segons quanta gira.



*Potenciómetre en breadboard: GND, 5V i senyal a A1.*

## 2. Sortides 'analògiques': PWM

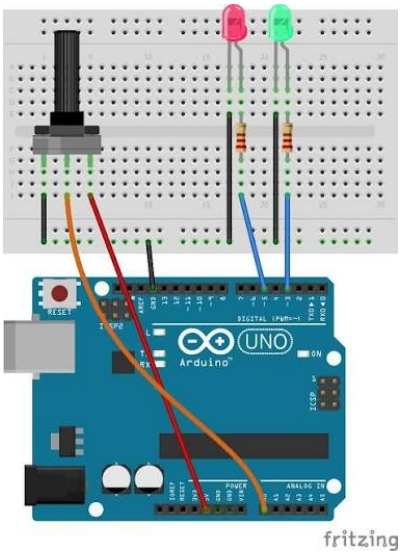
> *PWM = Pulse Width Modulation (modulació per amplada de pols). La sortida PWM no és analògica real: és una senyal digital que s'encén i s'apaga molt ràpid, de manera que la MITJANA sembla un voltatge intermedi.*

La UNO té 6 sortides PWM (els pins marcats amb ~). analogWrite accepta valors de 0 a 255 (precisió de 8 bits).

> *analogWrite(pin, valor) — valor entre 0 (0 V) i 255 (5 V)*

⚠ Compte: les entrades analògiques donen 0-1023 (10 bits) però la sortida PWM només admet 0-255 (8 bits). Per això cal dividir per 4 o usar map().

## 3. PROPOSTA 8 · Control de la brillantor d'un LED amb potenciòmetre




*Cablejat muntatge de 2 LED i un potenciòmetre de control.*

- 1. Connecta un LED al pin PWM 11 (amb resistència).
- 2. Connecta el potenciòmetre al pin analògic A0.
- 3. Llegeix A0 amb analogRead i divideix per 4 (1024/256).
- 4. Escriu el resultat al LED amb analogWrite.

El codi:

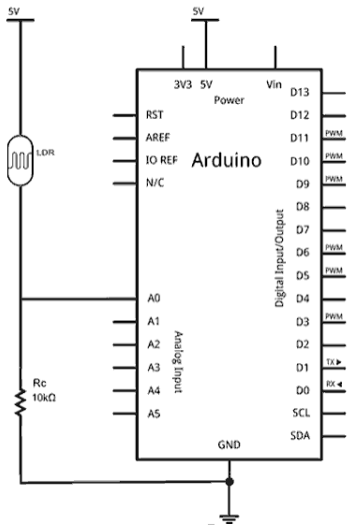
```
/* CONTROL LLUENTOR AMB POTENCIÒMETRE */
int pinLed = 11;
int valorSensor = 0;
```

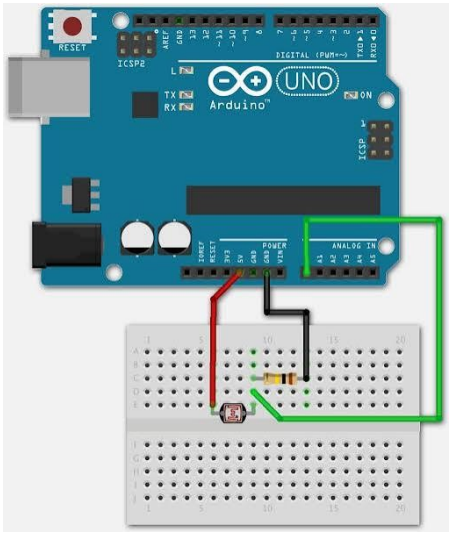
```
void setup()
{
pinMode(pinLed, OUTPUT);
}
void loop()
{
valorSensor = analogRead(A0);
analogWrite(pinLed, valorSensor / 4);
}
```

Puja'l i gira el potenciòmetre: la brillantor del LED canvia segons l'angle que giris.

## Ara amb un LDR (Light Depending Resistor)

Una LDR és una resistència depenent de la llum, que fa que en funció de la lluminositat que capta pot posar posar la seva resistència a un valor tan baix que que simula un polsador tancat.




*Cablejat de la LDR: divisor de tensió amb 10 kΩ, punt mig a A0.*

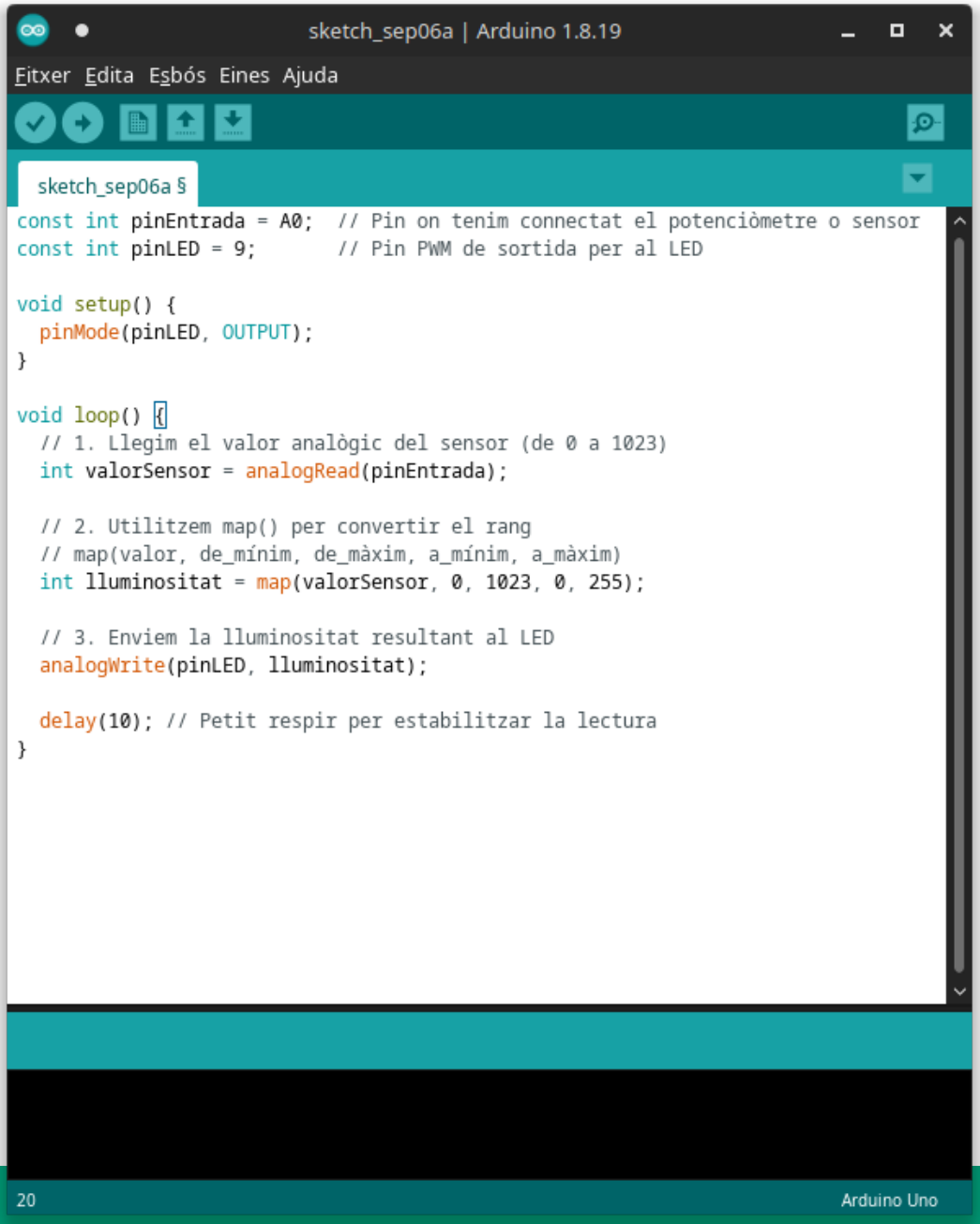
## La funció map()

Una alternativa més elegant que dividir és map(), que ajusta uns valors d'entrada a uns de sortida de forma proporcional:

```
map(variable, val_in_min, val_in_max, val_out_min, val_out_max)
```

EXEMPLE: map(valorSensor, 0, 1023, 0, 255) — igual que dividir per 4 però més clar. Al bloc 6 la farem servir també per percentatges.

El codi que podeu provar:



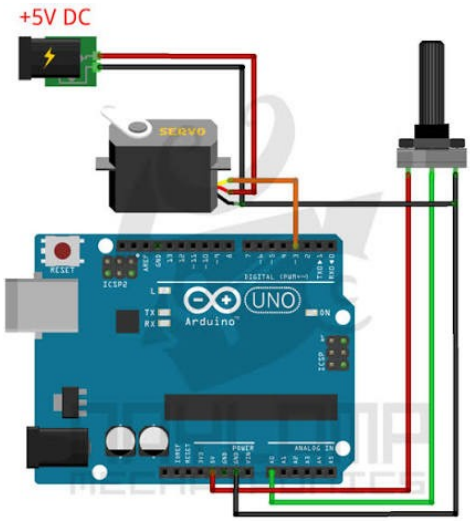

**Activitat 8**

Munta i puja el programa. Després:

- a) Inverteix la lògica: més gir = MENYS brillo.
- b) Canvia el LED per un RGB (prova un sol canal).
- c) EXTRA: fes que sobre un cert llindar (>800) el LED parpellegi.
- Pots fer que s’encengui un LED i a partir de cert valor se n’encengui un altre?

## 4. Servomotors

Un dels majors problemes de les plaques Arduino és que no poden proporcionar pels seus pins de sortida la suficient intensitat de corrent per alimentar directament certs elements com els motors. És per això que és recomanable no alimentar el SERVOS directament des d ela placa i impossible si en tenim diversos de connectat. Cap fer-ho amb una font externa.
Els SERVOMOTORS es controlen amb senyals PWM especials i la llibreria Servo.h.




*Cablejat del servo: vermell a 5V, negre a GND, senyal (taronga/blanc) al pin 11.*

### Servo de rotació limitada

Aquest tipus de servo es posiciona en un ANGLE entre 0° i 180°. Té 3 cables: vermell (5 V), negre o marró (GND) i blanc/o taronja (senyal de control al pin PWM).

⚠ CONSIDERACIONS DE SEGURETAT:

- Els servos consumeixen MOLTA intensitat, sobretot si els forces. Si la placa es resseteja sovint, alimenta-la externament (piles o carregador).
- Connecta'ls sempre correctament: si confons alimentació i senyal, pots cremar el servo o la placa.
- Cada servo té un voltatge màxim (5 o 6 V generalment). Més tensió = servo cremat.

### PROPOSTA 9 · Moviment del servo en intervals (salts)

Objectiu d'aquest codi: el servo es mou en salts de 10° des de 0° fins 170°, parant 1 segon a cada posició. Al final torna a l'inici i torna a començar.

```
#include <Servo.h>
```

```
Servo meuServo;
int angle = 0;

void setup()
{
meuServo.attach(11);
}

void loop()
{
for (int i = 0; i < 18; i++)
{
meuServo.write(angle);
angle = angle + 10;
delay(1000);
}
}
```

Elements nous:

- #include <Servo.h>: afegeix la llibreria del servo.
- Servo meuServo: crea l'objecte servo.
- meuServo.attach(11): el connecta al pin 11 (dins setup()).
- meuServo.write(angle): posiciona el servo a l'angle indicat (0-180).
- for (int i = 0; i < 18; i++): bucle que es repeteix 18 vegades (0 a 170 en salts de 10).

**Activitat 9**

Munta el servo (comprova vermell=5V, negre/marró=GND, blanc/taronja=pin 11) i puja el programa.

- a) Canvia els salts a 30° (quants bucles calen ara?).
- b) Fes que vagi més ràpid (delay 200 ms).
- c) EXTRA: fes un 'escombrat' suau: 0→180 ràpid i 180→0 lent (2 bucles for).

## 6. Sensor d'ultrasons: mesura de distància

El sensor HC-SR04 emet un pols d'ultrasons (pin TRIGGER) i escolta l'eco (pin ECHO). El temps entre pols i eco dona la distància.

> *distancia_cm = temps_echo / 58 (aproximació estàndard)*

## Pràctica — Lectura en cm de la distància d'un objecte

Muntatge: HC-SR04 (VCC, GND, Trig, Echo) + lectura per Serial Monitor o a un LED, brunzidor, pantalla LCD...

Aquí tens un exemple de codi per a Arduino utilitzant el sensor d'ultrasons **HC-SR04**.

Aquest programa mesura la distància constantment i, si detecta un objecte a una distància igual o inferior a **10 cm**, encén un LED (o pot activar un brunzidor/zumbador) per avisar-te. Si està a més de 10 cm, s'apaga.

**Connexions bàsiques:**

- **VCC**: 5V de l'Arduino
- **GND**: GND de l'Arduino
- **Trig**: Pin digital 9
- **Echo**: Pin digital 8
- **LED**: Pin digital 13 (o pots utilitzar el LED integrat a la placa)


sketch_sep06a | Arduino 1.8.19
Fitxer Edita Esbós Eines Ajuda
sketch_sep06a §
// Definim els pins del sensor i del LED
const int trigPin = 9;
const int echoPin = 8;
const int ledPin = 13; // O pots posar un brunzidor (brunzidor)
void setup() {
// Inicialitzem el monitor sèrie per veure les mesures
Serial.begin(9600);
// Configurem els pins
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
pinMode(ledPin, OUTPUT);
}
void loop() {
// 1. Enviem un pols de 10 microsegons pel pin Trig
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
// 2. Llegim el temps que triga el pols a tornar pel pin Echo (en microsegons)
long durada = pulseIn(echoPin, HIGH);
// 3. Calculem la distància en centímetres
// La velocitat del so és de 343 m/s (o 0.0343 cm/microsegon).
// Com que el so va i torna, dividim el resultat entre 2.
float distancia = durada * 0.0343 / 2;
// Mostrem la distància al monitor sèrie
Serial.print("Distancia: ");
Serial.print(distancia);
Serial.println(" cm");
// 4. Comprovem si està a 10 cm o menys
if (distancia > 0 && distancia <= 10.0) {
digitalWrite(ledPin, HIGH); // Encén l'avís (LED o brunzidor)
} else {
digitalWrite(ledPin, LOW); // Apaga l'avís
}
// Esperem un xic abans de la següent mesura
delay(100);
}
46
Arduino Uno

Modifica el programa per fer aquestes distàncies i compara amb un regle:

| Distància real (regla) |
|---|
| 10 cm |
| 20 cm |
| 50 cm |

Quina precisió té? A quina distància falla?

## PRODUCCIÓ U5 · Sistema de mesura amb sensor


Un sistema que llegeixi un sensor (LDR o ultrasó) i doni informació útil: avis lluminós de proximitat, engegada automàtica de llum, mesurador digital per Serial, alerta sonora si supera un llindar... o fins i tot un brunzidor que canviï de to en funció de la distància mesurada.
Et dono una pista pel que fa la brunzidor. La resta, és cosa teva:

## Repàs i

## autoavaluació

### Rúbrica: 1 = No assolit · 2 = Satisfactori · 3 = Notable · 4 = Excel·lent

| Objectiu |
| --- |
| Llegeixo sensors analògics amb analogRead (0-1023) |
| Expliqu què és el PWM i uso analogWrite (0-255) |
| Reescalo valors amb map() o dividint per 4 |
| Connecto un servo amb les precaucions de seguretat |
| Posiciono el servo a diferents angles amb Servo.h i bucles for |

### Glossari

- analogRead(A0): llegeix 0-1023 segons la tensió (0-5 V).
- PWM: modulació per amplada de pols; simula analògic.
- analogWrite(pin, 0-255): escriu valor PWM (pines amb ~).
- map(): reescala proporcionalment un rang a un altre.
- Potenciómetre: resistència variable (divisor de tensió).
- Servo rotació limitada: motor que posiciona un angle 0-180°.
- Servo.h: llibreria dels servomotors.
- attach / write: connectar el servo al pin / posicionar a l'angle.