PROF. DELBARBA LUCA

27/07/2023

PARTE 2

Automazione Industriale con EXCEL, ARDUINO, TINKERCAD e WOKWI

WOKWi

INTRODUZIONE

Questo testo contiene una serie di lezioni ed esercitazioni che possono essere realizzate con

- il simulatore **ThinkerCad** disponibile sul sito https://www.tinkercad.com/
- il simulatore **Wokwi** disponibile sul sito https://wokwi.com/
- il simulatore **CircuitJS** disponibile sul sito https://www.falstad.com/circuit/circuitjs.html
- il simulatore **SimulIDE** disponibile sul sito https://www.simulide.com/p/home.html
- il simulatore **NL5 lite** disponibile sul sito https://sidelinesoft.com/nl5/index.php?page=download
- un foglio di calcolo (Excel, Calc …)
- un kit Arduino R1

I simulatori in oggetto permettono di programmare una scheda Arduino UNO e di risolvere semplici problemi di automazione industriale utilizzando i più comuni componenti elettronici ed una serie di sensori ed attuatori.

La maggior parte delle esercitazioni proposte contiene una breve descrizione dei componenti utilizzati. Per ulteriori dettagli è necessario fare riferimento a testi specifici di elettronica ed automazione.

Una conoscenza di base dell'elettronica e dell'elettrotecnica è necessaria per capire gli schemi proposti.

Quasi tutti gli esercizi presentano una possibile soluzione software da caricare su una scheda Arduino.

Il vantaggio offerto dall'utilizzo di ThinkerCAD, rispetto ad altri software di simulazione, è la possibilità di replicare in modo identico il circuito su una breadboard e di utilizzare lo stesso programma simulato sulla scheda Arduino.

Il testo aggiornato periodicamente è reperibile a questo indirizzo web:
http://www.energiazero.org/cartelle.asp?dir=thinkercad

Si ringrazia in anticipo per eventuali segnalazioni di errori e/o miglioramenti apportabili al testo alla seguente mail: energiazero.org@gmail.com

NOTA BENE:
Alcuni esempi e immagini sono stati reperiti sul web cercando materiale che non fosse coperto da copyright. Si ringraziano tutti coloro che hanno reso disponibile il materiale.
Se per errore fosse stato inserito materiale protetto, cortesemente segnalatelo alla mail sopra citata.

SOMMARIO

SENSORI E TRASDUTTORI

Sensore e trasduttore sono due termini concettualmente diversi:

- il primo identifica un dispositivo che converte una determinata grandezza in ingresso in grandezza elettrica
- il secondo trasforma una forma di energia in un'altra

Spesso non viene fatta una netta distinzione tra i due dispositivi e spesso i termini vengono usati come sinonimi.
Il trasduttore, generalmente costituito da uno o più sensori e da due blocchi di interfacciamento, è il primo elemento che compone il sistema di acquisizione dati e provvede alla conversione di un tipo di energia in ingresso in un altro tipo di energia in uscita.

L’interfaccia ingresso-sensore realizza una prima conversione del misurando in grandezza adatta al sensore, il quale rilevando le variazioni della grandezza al suo ingresso produce una variazione del segnale elettrico in uscita.

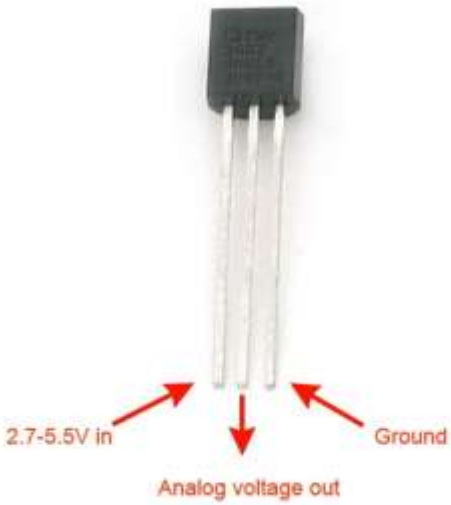
L’interfaccia sensore-uscita realizza una connessione fisica tra il sensore e l’apparecchio successivo del sistema DAQ (data acquisition).

Un elenco dei più comuni trasduttori è riportato nella seguente tabella.

Fenomeno da misurare	Trasduttori impiegati
Temperatura	Termocoppie Termistori (PTC, NTC) Rivelatori resistivi di temperatura Trasduttori basati su circuiti integrati
Luce	Fotocellule, fotodiodi e fototransistors CCD
Suono	Microfoni
Forza e pressione	Estensimetri Trasduttori piezoelettrici Trasduttori piezoresistivi
Posizione, spostamento e rotazione	Potenziometri Trasduttori induttivi di spostamento Encoder ottici Giroscopi ottici
Presenza o prossimità di un oggetto	Trasduttori induttivi e capacitivi Trasduttori magnetici Trasduttori ad ultrasuoni (Sonar)
Flusso di fluidi	Misuratori diretti di portata Misuratori di velocità di un fluido
Livello pH	Sonde pH

SENSORE DI TEMPERATURA TMP36

Il componente elettronico TMP36 è un dispositivo integrato ad alta precisione utilizzato per misurare la temperatura ambientale.



Dato il basso costo e l'ampia scala di valori ammissibili (ovvero da -40°C fino a 125°C) questi dispositivi sono particolarmente diffusi. Non è necessaria nessuna operazione calibrazione per ottenere valori di accuratezza pari a ±1°C ad una temperatura di circa +25°C e ±2°C nel range di temperature −40°C to +125°C.

Nel caso specifico, osservando il grafico che riporta la caratteristica tensione/temperatura (per il TMP36 la linea è evidenziata in rosso) per una tensione di uscita di 0.5V il sensore rileva la temperatura di 0°C.

Valori di tensione inferiori a 0.5V indicano una temperatura sotto lo zero, mentre valori di tensione superiori a 0.5V indicano una temperatura positiva.
Inoltre è importante considerare che "una variazione 1 grado corrisponde ad una variazione di tensione di 10mV".

Quindi, se sul pin di input analogico sono presenti 550mV significa che il sensore sta rilevando una temperatura di 5°C (550mV – 500mV = 50 mV → variazione di 5°C).

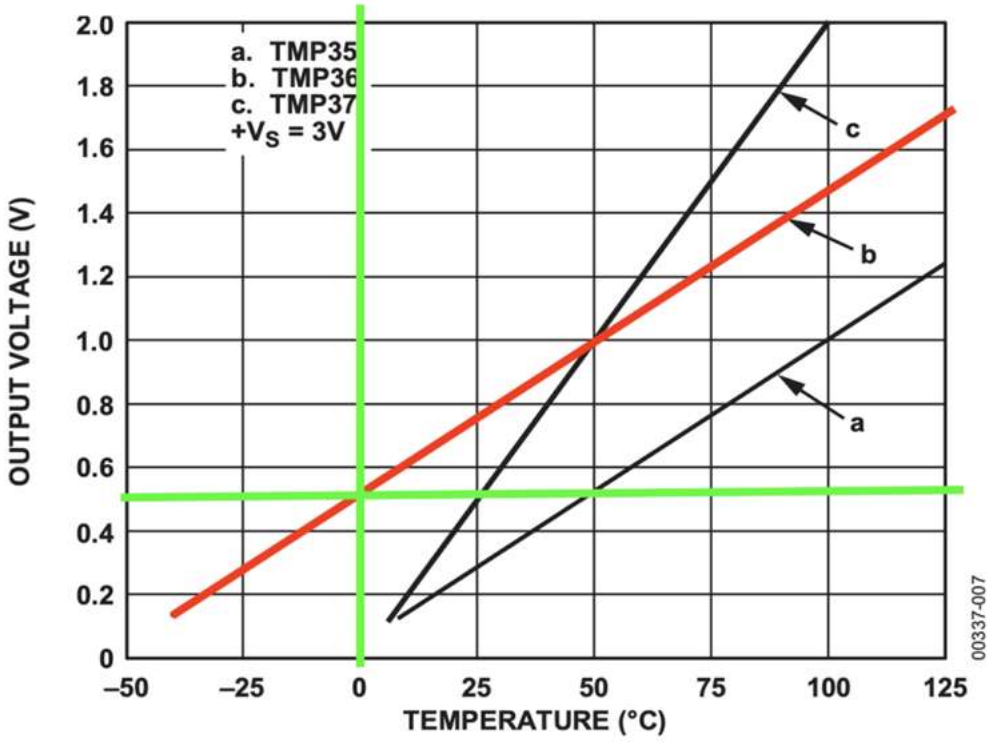


Figure 6. Output Voltage vs. Temperature

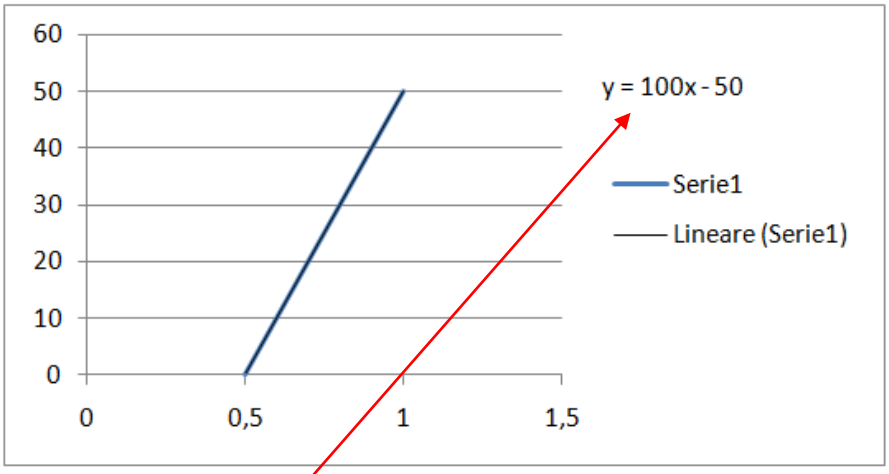
CURVA CARATTERISTICA DEL SENSORE TMP36

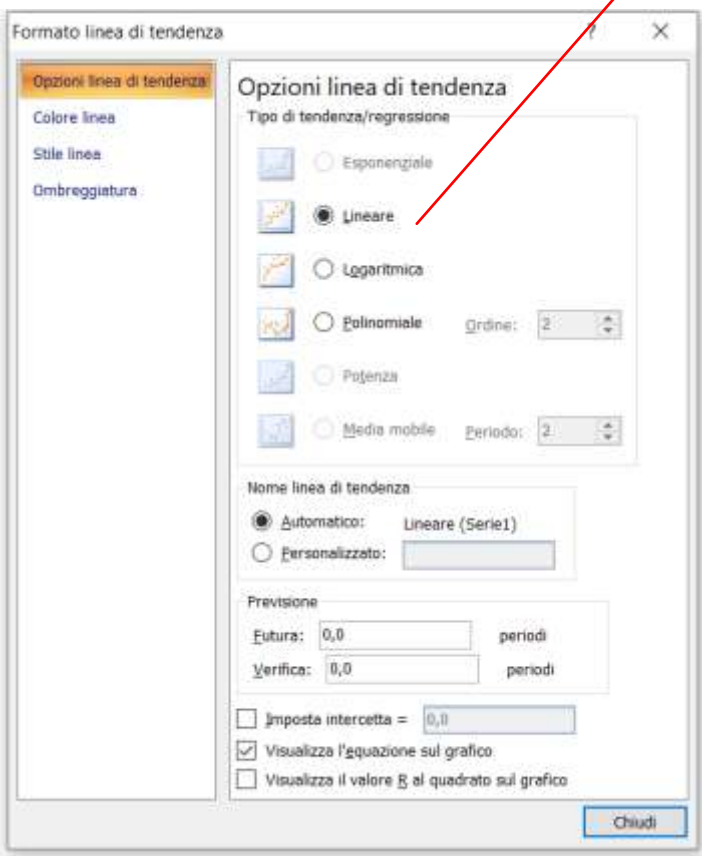
La curva caratteristica del sensore è la funzione matematica che permette di calcolare la grandezza fisica (T°C) in funzione della grandezza elettrica misurata (Volt).
Se la funzione NON è lineare torna utile un *"foglio di calcolo"* per determinare la curva di tendenza che approssima al meglio la curva caratteristica del sensore.

Nel caso del TMP36 la curva è una semplice retta: T(°C) = 100 Volt – 50.
Se la tensione fornita dal sensore viene rilevata tramite Arduino su un PIN analogico (10 bit) si dovrà convertire il risultato della lettura nel seguente modo:

```
float volt = analogRead(PIN) * 5.0/1024.0;    // usare i decimali per le divisioni!
float temperatura = 100 * volt – 50;
```

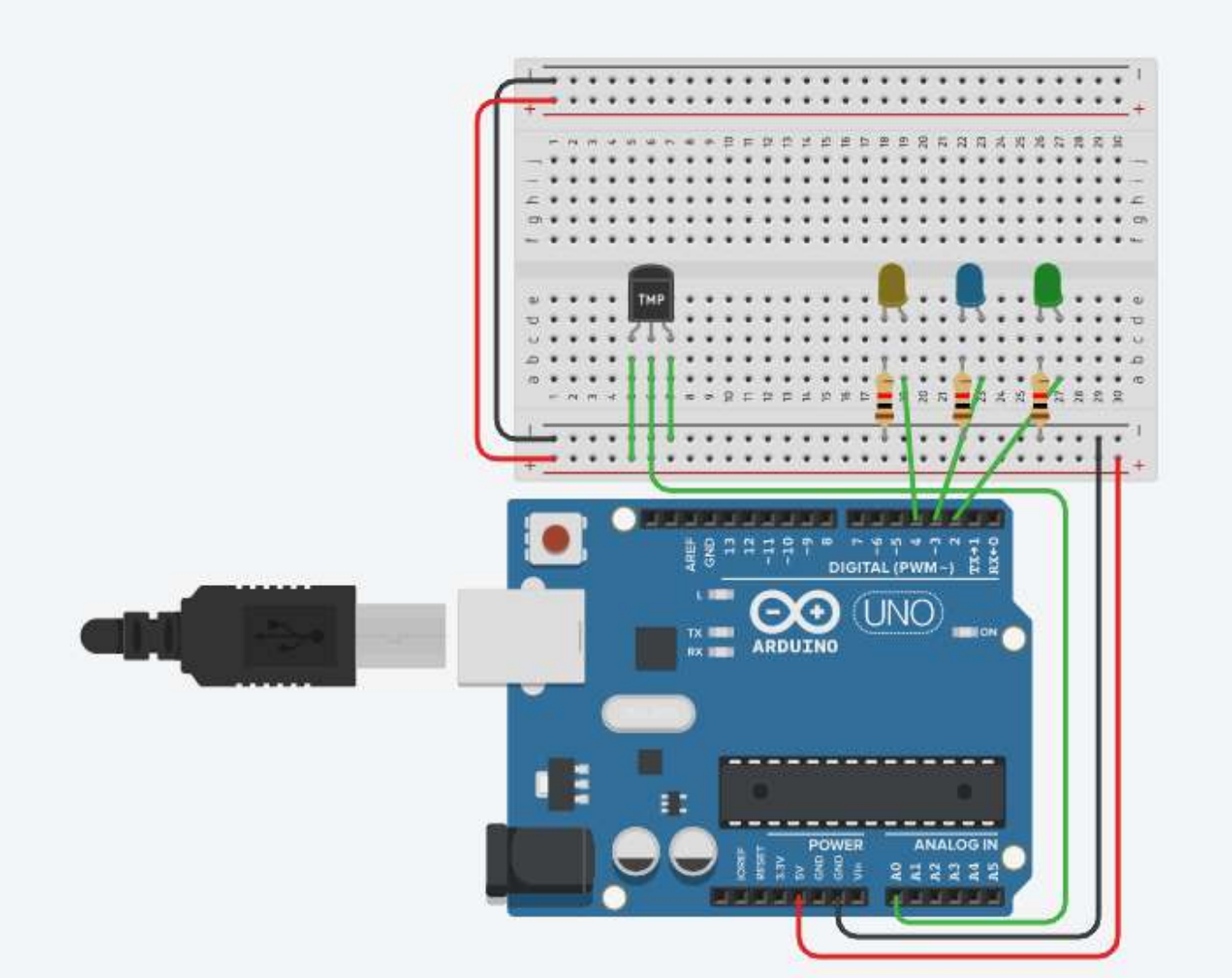
Volt	T °C
0,5	0
1	50





ESERCIZIO CON SENSORE TMP36

Accendere una striscia di 3 led in modo proporzionale alla temperatura rilevata dal sensore.

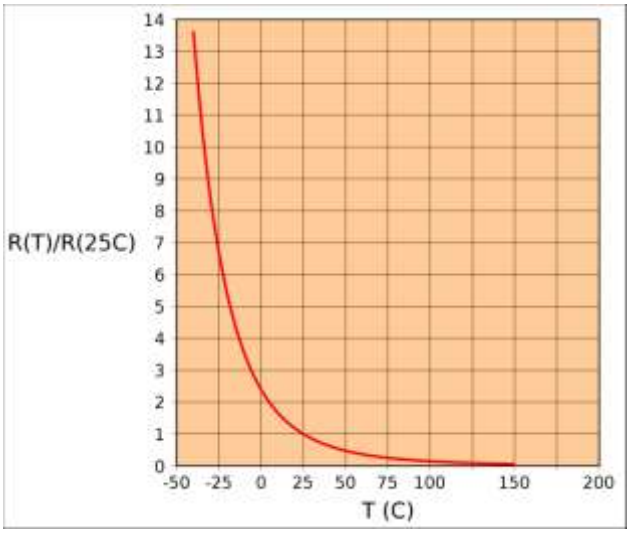


CODICE

```
float volt;
float temperatura= 0;

void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(9600);

  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
}

void loop()
{
  volt = analogRead(A0) * 5.0/1024.0;  // usare i decimali nella divisione!
  temperatura = 100 * volt – 50;
  Serial.print(temperatura);
  Serial.println(" C");

  if (temperatura < 0) {
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    Serial.println("SOTTO ZERO");
  }
  if (temperatura >= 0 && temperatura < 10) {
    digitalWrite(2, HIGH);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    Serial.println("BASSA");
  }
  if (temperatura >= 10 && temperatura < 20) {
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, LOW);
    Serial.println("MEDIA");
  }
  if (temperatura >= 20) {
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, HIGH);
    Serial.println("ALTA");
  }
  delay(1000);
}
```

TERMISTORE NTC (NEGATIVE TEMPERATURE COEFFICENT)

Un termistore è un resistore il cui valore di resistenza varia con la temperatura.

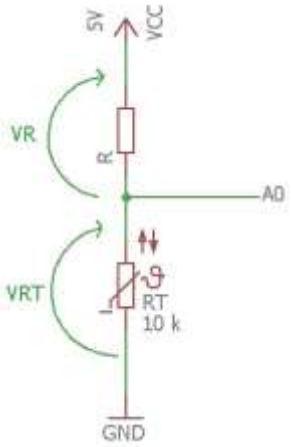


I termistori NTC possono essere caratterizzati con un'equazione detta equazione con parametro B o beta value:

$$\frac{1}{T} = \frac{1}{T_0} + \frac{1}{B}\ln\left(\frac{R}{R_0}\right)$$

dove le temperature sono in kelvin (K) e R_0 è la resistenza alla temperatura T0 (di solito 25 °C=298,15 K).

B è costante solo in prima approssimazione e di solito ne viene indicato l'intervallo di temperature in cui è valida e la sua tolleranza in % (ad esempio B25/85 ± 2% indica che B tra 25 °C e 85 °C ha un errore massimo di ± 2%.

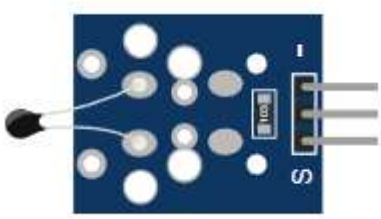


$$RT = R0\, e^{B\left(\frac{1}{T} - \frac{1}{T0}\right)}$$

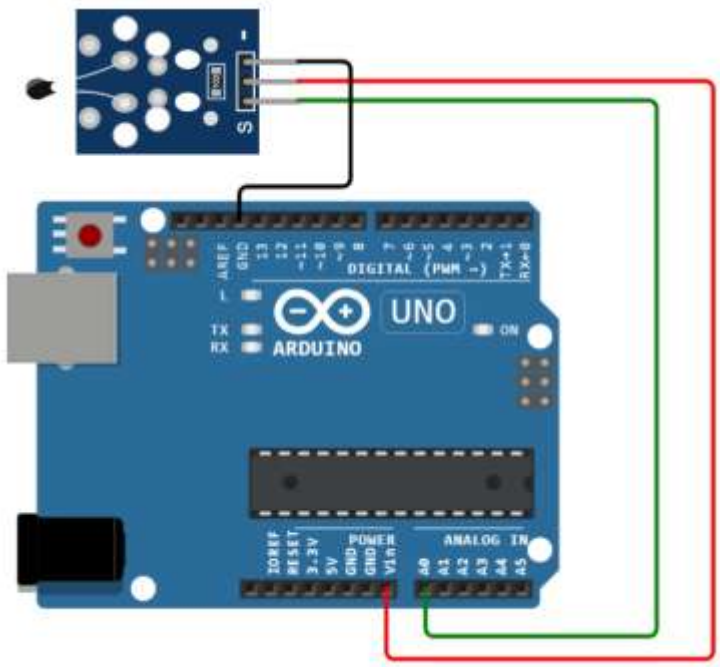
RT = VRT / (VR/R)

$$T = \frac{1}{\frac{\ln\left(\frac{RT}{R0}\right)}{B} + \frac{1}{T0}}$$

Il modulo sensore di temperatura per Arduino include un termistore NTC da 10K in serie con un resistore da 10K.



MONITORARE TEMPERATURA TRAMITE TERMISTORE NTC

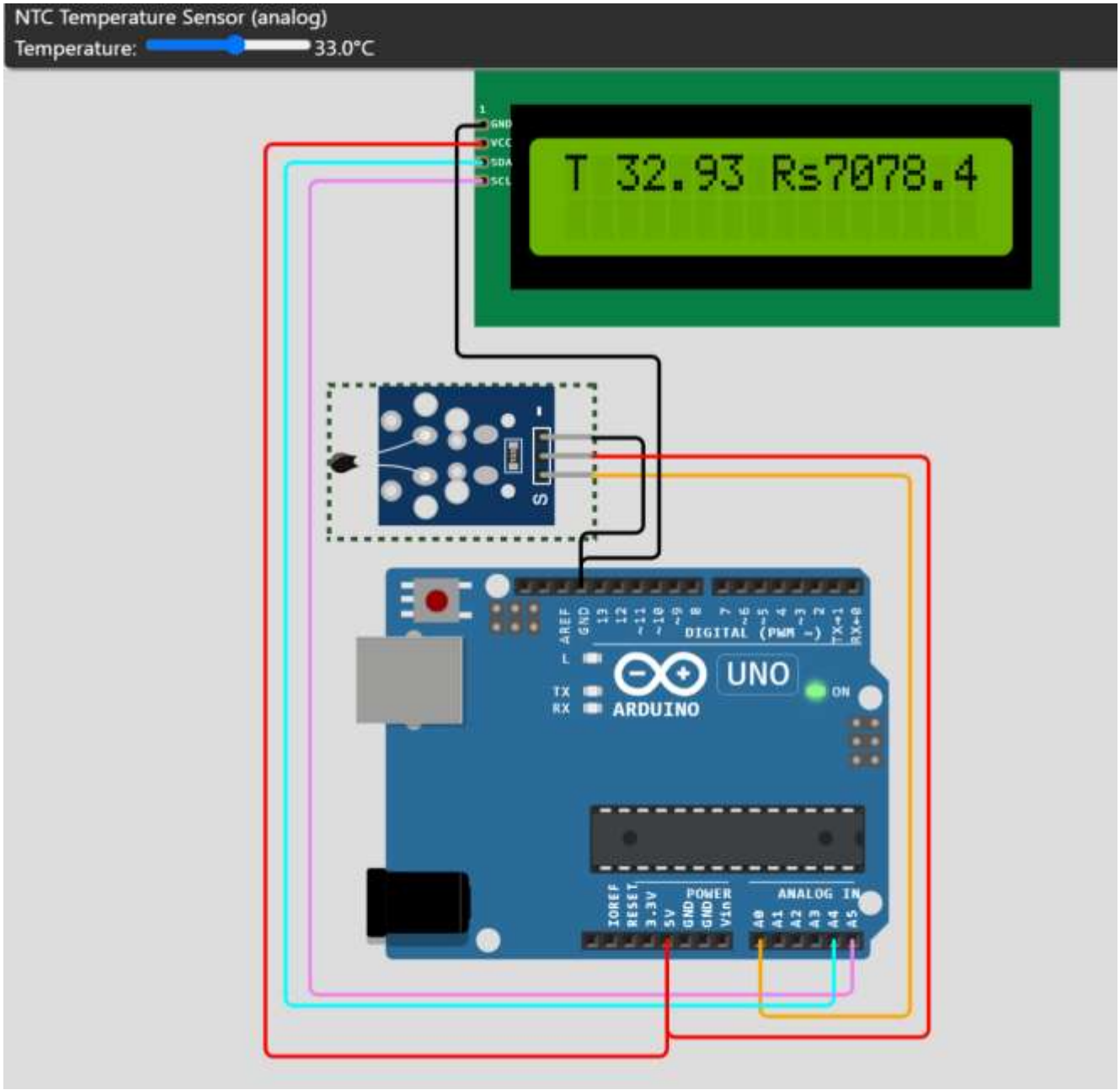


simulabile su "wokwi.com"

CODICE

```
//Thermistor parameters:  RT0: 10KΩ   B: 3977 K +- 0.75%   T0:  25 C +- 5%
//From datasheet
#define RT0 10000   // Ω
#define B 3977      // K
//--------------------------------------

#define VCC 5    //Supply voltage
#define R 10000  //R=10KΩ

//Variables
float RT, VR, ln, TX, T0, VRT;

void setup() {
  Serial.begin(9600);
  T0 = 25 + 273.15;
}

void loop() {
  VRT = analogRead(A0);  // 0-1023 → tensione sul termistore
  VRT = (5.00 / 1023.00) * VRT;     // converto in V
  VR = VCC - VRT;  // tensione sulla resistenza R da 10K
  RT = VRT / (VR / R);    // Resistenza di RT  (V/I)
  ln = log(RT / RT0);
  TX =1/ (ln / B + 1 / T0);  //Temperature from thermistor in K
  TX = TX - 273.15;               //Conversion to °C

  Serial.print("Temperatura: ");
  Serial.print(TX);
  Serial.println(" °C");
  delay(1000);
}
```

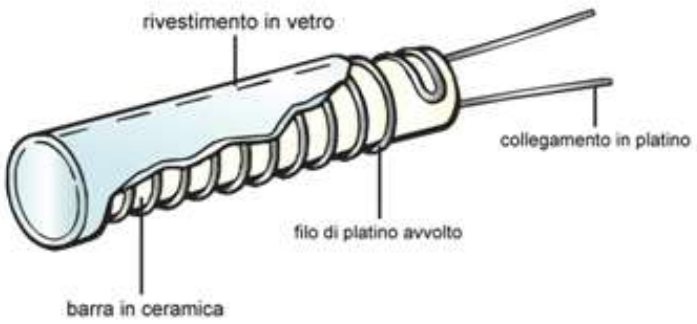
MONITORARE TEMPERATURA CON TERMISTORE NTC E LCD 16X2 I2C



simulabile su "wokwi.com"

```
//Thermistor parameters:  RT0: 10KΩ   B: 3977 K +- 0.75%   T0:  25 C +- 5%
//From datasheet
#define RT0 10000   // Ω
#define B 3977      // K
#define VCC 5    //Supply voltage
#define R 10000.0  //R=10KΩ
//--------------------------------------

// Voltmetro
float Rref= 660.0;

#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 20, 4);
// Arduino: LiquidCrystal_I2C lcd(0x3f, 16, 2);

int pinSensor = A0;

//Variables
float RT, VR, ln, TX, T0, VRT;


String riga1 = "Display LCD con";
String riga2 = "interfaccia I2C";

void setup(){
  lcd.init();
  lcd.backlight();
  pinMode(pinSensor, INPUT);

  lcd.clear();
  lcd.setCursor(0, 0);
  typewriting(riga1);
  lcd.setCursor(0, 1);
  typewriting(riga2);

  delay(1500);

  lcd.clear();
}

void loop(){
  T0 = 25 + 273.15;
  VRT = analogRead(A0);  // 0-1023 ◊ tensione sul termistore
  VRT = (5.00 / 1023.00) * VRT;      // converto in V
  VR = VCC - VRT;  // tensione sulla resistenza R da 10K
  RT = VRT / (VR / R);    // Resistenza di RT  (V/I)
  ln = log(RT / RT0);
  TX =1/ (ln / B + 1 / T0);  //Temperature from thermistor in K
  TX = TX - 273.15;                 //Conversion to °C

  lcd.setCursor(0, 0);
  lcd.print("T "); lcd.print(TX); lcd.print(" Rs"); lcd.print(RT);

  delay(1000);
}

void clearRow(byte rowToClear)
{
  lcd.setCursor(0, rowToClear);
  lcd.print("                ");
}

void typewriting(String messaggio){
  int lunghezza = messaggio.length();
  for(int i = 0; i < lunghezza; i++){
    lcd.print(messaggio[i]);
    delay(25);
  }
}
```

TERMORESISTENZE



Molte industrie utilizzano le termoresistenze per misurare la temperatura e, la maggior parte di questi dispositivi, utilizza un sensore Pt100 o Pt1000. Questi due sensori di temperatura hanno caratteristiche simili, ma la loro differenza nella resistenza nominale determina quale sia la scelta ideale in bassa alla propria applicazione.

I rilevatori a resistenza di temperatura (RTD – Resistance temperature detectors), detti anche termoresistenze, sono noti dispositivi di misura della temperatura grazie alla loro affidabilità, accuratezza, versatilità, ripetibilità e facilità di installazione.

Il principio di base di una termoresistenza è che il suo sensore a filo, realizzato in un metallo con una resistenza elettrica nota, cambia il suo valore di resistenza quando la temperatura sale o scende. Sebbene le termoresistenze abbiano alcune limitazioni, tra cui una temperatura massima di misura di circa 600 ° C, nel complesso rappresentano la soluzione di misura della temperatura ideale per una moltitudine di processi.

PERCHÉ UTILIZZARE UN SENSORE AL PLATINO

I fili dell'elemento di misura di una termoresistenza possono essere realizzato in nichel, rame o tungsteno, ma il platino (Pt) è oggi di gran lunga il metallo più popolare utilizzato. È più costoso di altri materiali, ma il platino ha diverse caratteristiche che lo rendono particolarmente adatto per le misure di temperatura, tra cui:

- Relazione quasi lineare tra resistenza e temperatura
- Alta resistività (59 Ω / cmf rispetto a 36 Ω / cmf per il nichel)
- Resistenza elettrica non degradabile nel tempo
- Eccellente stabilità
- Ottima passività chimica
- Elevata resistenza alla contaminazione

DIFFERENZA TRA PT100 E PT1000

Tra le termoresitenze in platino, le Pt100 e Pt1000 sono le più comuni. Le Pt100 hanno una resistenza nominale di 100 Ω al punto di fusione del ghiaccio (0 ° C). La resistenza nominale delle Pt1000 a 0 ° C è invece di 1.000 Ω. La linearità della curva caratteristica, il campo di temperatura operativo e il tempo di risposta sono gli stessi per entrambi. Anche il coefficiente di temperatura della resistenza è lo stesso.

Tuttavia, a causa della diversa resistenza nominale, le letture delle sonde Pt1000 sono maggiori di un fattore 10 rispetto alle Pt100. Questa differenza diventa evidente quando si confrontano configurazioni a 2 fili, in cui si verifica l'errore di misura. Ad esempio, l'errore di misura in una Pt100 potrebbe essere di + 1,0 ° C, e quello di una Pt1000 con la stessa esecuzione potrebbe essere di + 0,1 ° C.

COME SCEGLIERE IL GIUSTO SENSORE AL PLATINO

Entrambi i tipi di sensori funzionano bene nelle configurazioni a 3 e 4 fili, dove i cavi e i connettori aggiuntivi compensano gli effetti della resistenza dei fili conduttori sulla misura della temperatura. Le due tipologie di configurazione hanno un prezzo simile. Le sonde Pt100, tuttavia, sono più popolari delle Pt1000 per un paio di motivi:

Una sonda Pt100 è disponibile sia in esecuzione a filo avvolto che a film sottile, offrendo agli utenti la possibilità di scelta e flessibilità. Le sonde Pt1000 sono quasi sempre solo a film sottile

Poiché il loro uso è così diffuso in tutti i settori, le sonde Pt100 sono compatibili con una vasta gamma di strumenti e processi.

Quindi, perché si dovrebbe optare per la sonda Pt1000? Le situazioni in cui la maggiore resistenza nominale ha un vantaggio evidente sono le seguenti:

Una sonda Pt1000 è migliore nella configurazione a 2 fili e quando viene utilizzata con lunghezze di cavo più lunghe. Minore è il numero di fili e più lunghi essi sono, maggiore è la resistenza che si aggiunge alle letture, causando in tal modo imprecisioni. La maggiore resistenza nominale della sonda Pt1000 compensa questi errori aggiunti

Una sonda Pt1000 è migliore per le applicazioni alimentate a batteria. Un sensore con una resistenza nominale più elevata utilizza meno corrente elettrica e, pertanto, richiede meno energia per funzionare. Il consumo energetico ridotto prolunga la durata della batteria e l'intervallo tra la manutenzione, riducendo i tempi di fermo impianto e i costi

Poiché una Pt1000 consuma meno energia, l'autoriscaldamento è inferiore. Ciò significa meno errori di lettura a causa di temperature superiori a quelle ambientali

In generale, i le sonde temperatura Pt100 sono più comunemente utilizzate nelle applicazioni di processo, mentre le Pt1000 sono utilizzate nei settori della refrigerazione, riscaldamento, ventilazione, automotive e dei costruttori di macchine.

SOSTITUZIONE DELLE TERMORESISTENZE: NOTA SULLE NORME INDUSTRIALI

Le termoresistenze sono facili da sostituire, ma non si tratta semplicemente di sostituirle l'una con l'altra. Il problema a cui gli utenti devono prestare attenzione quando sostituiscono le sonde Pt100 e Pt1000 esistenti è la norma nazionale o internazionale.

La norma U.S.A. più vecchia, ad esempio, indica il coefficiente di temperatura del platino come 0,00392 Ω / Ω / ° C (ohm per ohm per grado centigrado). Nella nuova norma europea DIN / IEC 60751, che viene utilizzata anche in Nord America, è 0,00385 Ω / Ω / ° C. La differenza è trascurabile a temperature più basse, ma diventa evidente al punto di ebollizione dell'acqua (100 ° C), quando la norma più vecchia leggerà 139,2 Ω mentre quella più recente leggerà 138,5 Ω.

CONVERTIRE LA RESISTENZA PT100/PT1000 IN TEMPERATURA

La variazione di una resistenza elettrica dipende dalla differenza di temperatura e dai coefficienti termici del materiale utilizzato.
La resistenza nominale a 0 °C è pari a 100 Ω per Pt100 e 1 kΩ per Pt1000.

I coefficienti termici del platino sono pari a
A = 3,91 · 10−3 [K-1]

B = -0,588 · 10−6 [K-2]

La formula generale per calcolare una resistenza in funzione della temperatura è la seguente:

$$R(\vartheta) = R_{\vartheta\,0} \cdot (1 + \mathrm{A} \cdot (\vartheta - \vartheta_0) + \mathrm{B} \cdot (\vartheta - \vartheta_0)^2)$$

R(ϑ):	Resistenza in funzione della temperatura [Ω]
R0:	Resistenza nominale elettrica a 0 °C [Ω]
ϑ:	Temperatura [°C]
ϑ0:	Temperatura di riferimento [°C]
A:	Coefficiente termico lineare [K-1]
B:	Coefficiente termico quadrato [K-2]

Il range di temperatura da 0 °C a 100 °C può essere descritto con un'equazione approssimata lineare.
A tale scopo si sceglie la temperatura di riferimento ϑ0 = 0 °C.

I coefficienti A e B vengono sostituiti dal coefficiente medio α = 3,91 · 10−3 K-1.

$$R(\vartheta) = R_0 \cdot (1 + \alpha \cdot \vartheta)$$

R(ϑ):	Resistenza in funzione della temperatura [Ω]
R0:	Resistenza nominale elettrica a 0 °C [Ω]
Θ:	Temperatura [°C]
α:	Coefficiente termico medio [K-1]

Modificando la formula è possibile convertire in temperatura la resistenza misurata:

$$\vartheta(\mathrm{R}) = \frac{R - R_0}{R_0 \cdot \alpha}$$

$$\vartheta(\mathrm{R}) = \frac{\Delta R}{R_0 \cdot \alpha} \qquad \text{con} \quad \Delta R = R - R_0$$

ϑ(R):	Temperatura in funzione della resistenza [°C]
α:	Coefficiente termico medio [K-1]
R:	Resistenza misurata della sonda Pt [Ω]
R0:	Resistenza nominale elettrica a 0 °C [Ω]
ΔR:	Variazione misurata della resistenza [Ω]

Quindi nota la variazione di resistenza si può rilalire risalire alla temperature:

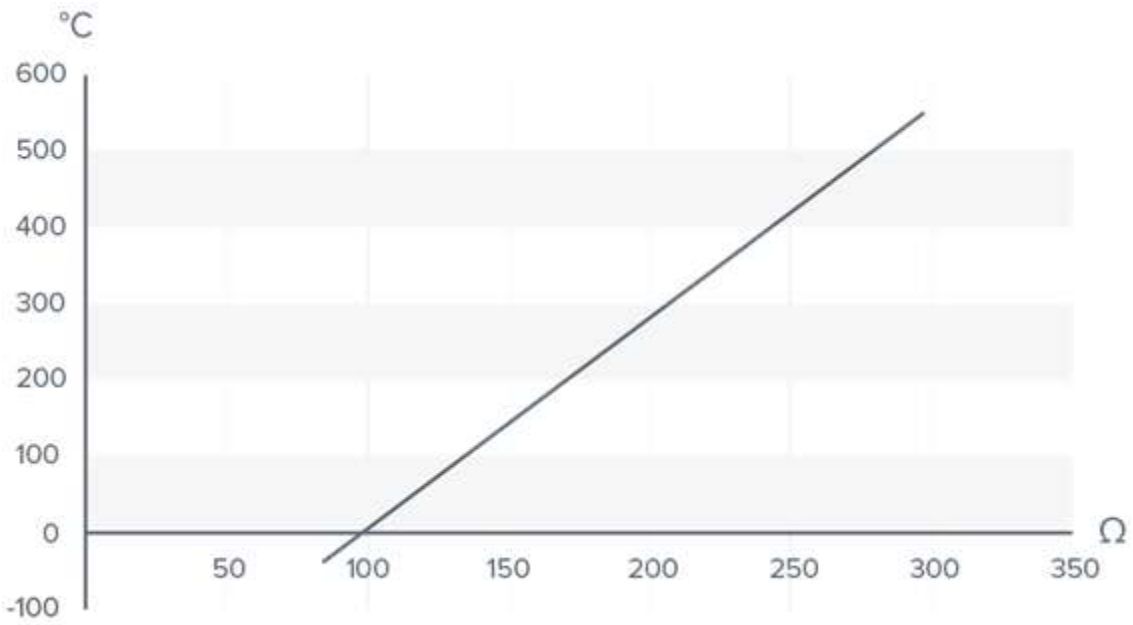
ΔR = 2*R*(Vab/E)/(0.5-Vab/E); → Rpt1000= R+ ΔR → T = (Rpt1000/1000-1)/0.00385

CURVA CARATTERISTICA DELLE TERMORESISTENZE

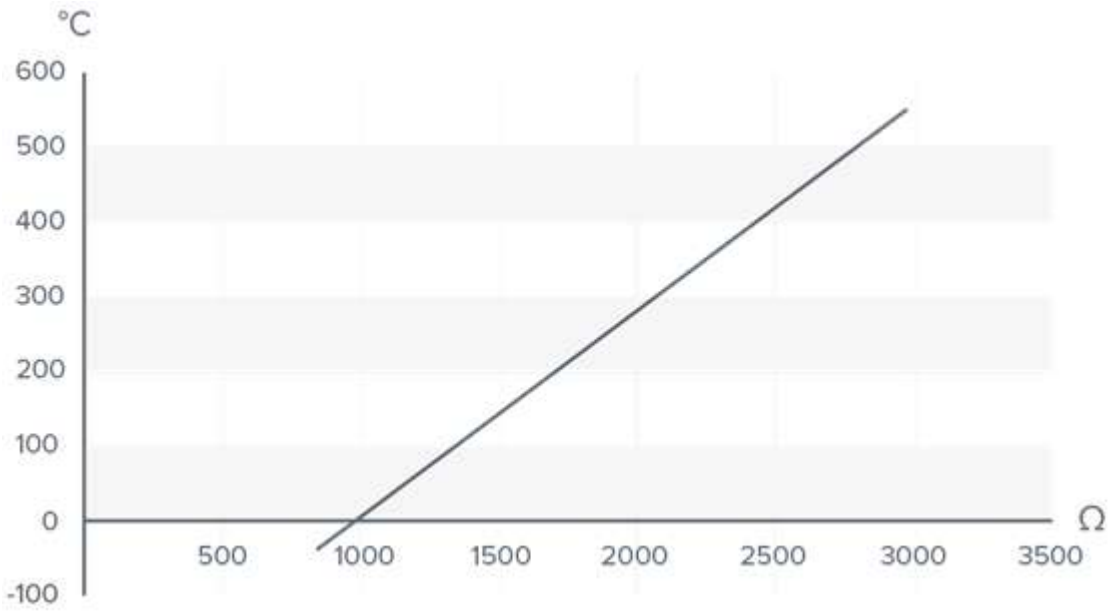
La curva caratteristica rappresenta il rapporto lineare tra la resistenza elettrica e la temperatura.
I valori concreti di Pt100 e Pt1000 possono essere dedotti graficamente dalle curve caratteristiche Pt100 / Pt1000 o letti direttamente dalle tabelle Pt100 / Pt1000.

Il platino si presta particolarmente bene come materiale, grazie alla sua elevata stabilità a lungo termine e alle caratteristiche elettriche piuttosto costanti alle alte temperature.
Per questo la curva caratteristica delle resistenze al platino è estremamente lineare anche a fronte di temperature elevate.
Aggiungendo al platino altre sostanze si ottengono risultati ancora migliori.

CURVA CARATTERISTICA PT100

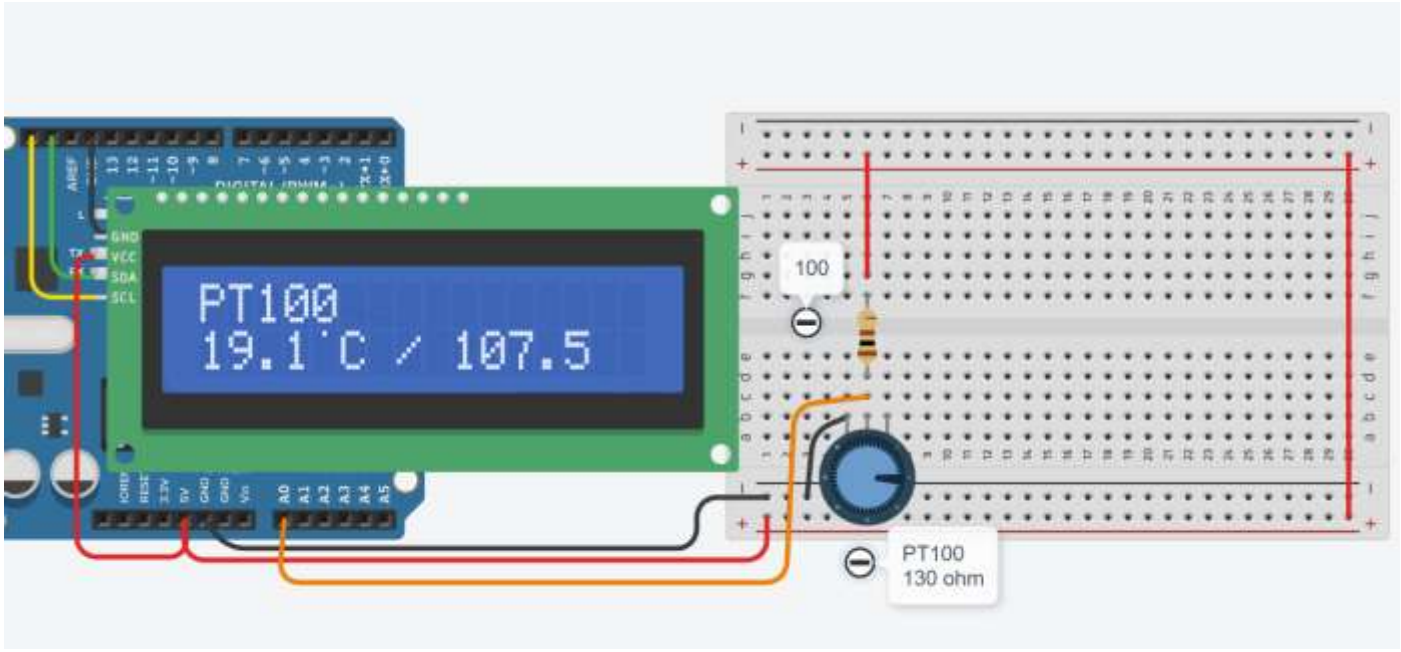


CURVA CARATTERISTICA PT1000



TERMORESISTENZA PT100 CON PARTITORE DI TENSIONE

La termoresistenza viene simulata tramite un potenziometro (130 ohm fondo scala) .
La temperatura viene rilevata tramite un partitore di tensione con una resistenza di 100 ohm.
Attenzione a fare I calcoli con numeri reali (double) per non perdere decimali nelle operazioni.



Codice

```
#include <Adafruit_LiquidCrystal.h>

int sensorValue = 0;
double sensoreVolt, corrente, resistenzaPT100, temperaturaPT100;

Adafruit_LiquidCrystal lcd_1(0);

void setup()
{
  pinMode(A0, INPUT);
  lcd_1.begin(16, 2);
  lcd_1.setCursor(0, 0);
  lcd_1.print("PT100");
  Serial.begin(9600);
}

void loop()
{
  // read the analog in value:
  sensorValue = analogRead(A0);
  sensoreVolt = roundTo(sensorValue * 5.0/1023,100.0);
  corrente= (5.00-sensoreVolt)/100.0;
  resistenzaPT100= sensoreVolt / corrente;
  temperaturaPT100 = ((resistenzaPT100/ 100.00)-1.0) / 0.00391;

  lcd_1.setCursor(0, 1);
  lcd_1.print(String(temperaturaPT100,1));
  char gr = char(176);
  lcd_1.print(gr);
  lcd_1.print("C / ");
  lcd_1.print(String(resistenzaPT100,1));
```

```
  Serial.print(sensorValue);      Serial.print('/');
  Serial.print(sensoreVolt,5);    Serial.print('/');
  Serial.print(corrente,5);       Serial.print('/');
  Serial.print(resistenzaPT100);  Serial.print('/');
  Serial.print(temperaturaPT100); Serial.println(char(176));

  delay(100);
}

// arrotonda decimali --> 10.0=1 dec, 100.0=2 dec, 1000.0=3 dec ...
double roundTo(double num, float dec)
{
  return (int)(num*dec + 0.5) / dec;
  //long numero= int(num *dec);
  //long decimali= dec* (num - numero)+ 0.5/dec;
  //return numero + decimali/ dec;
}
```

ESERCIZI:

1- SIMULARE IL CIRCUITO DI RILEVAZIONE DELLA TEMPERATURA CON UNA PT1000

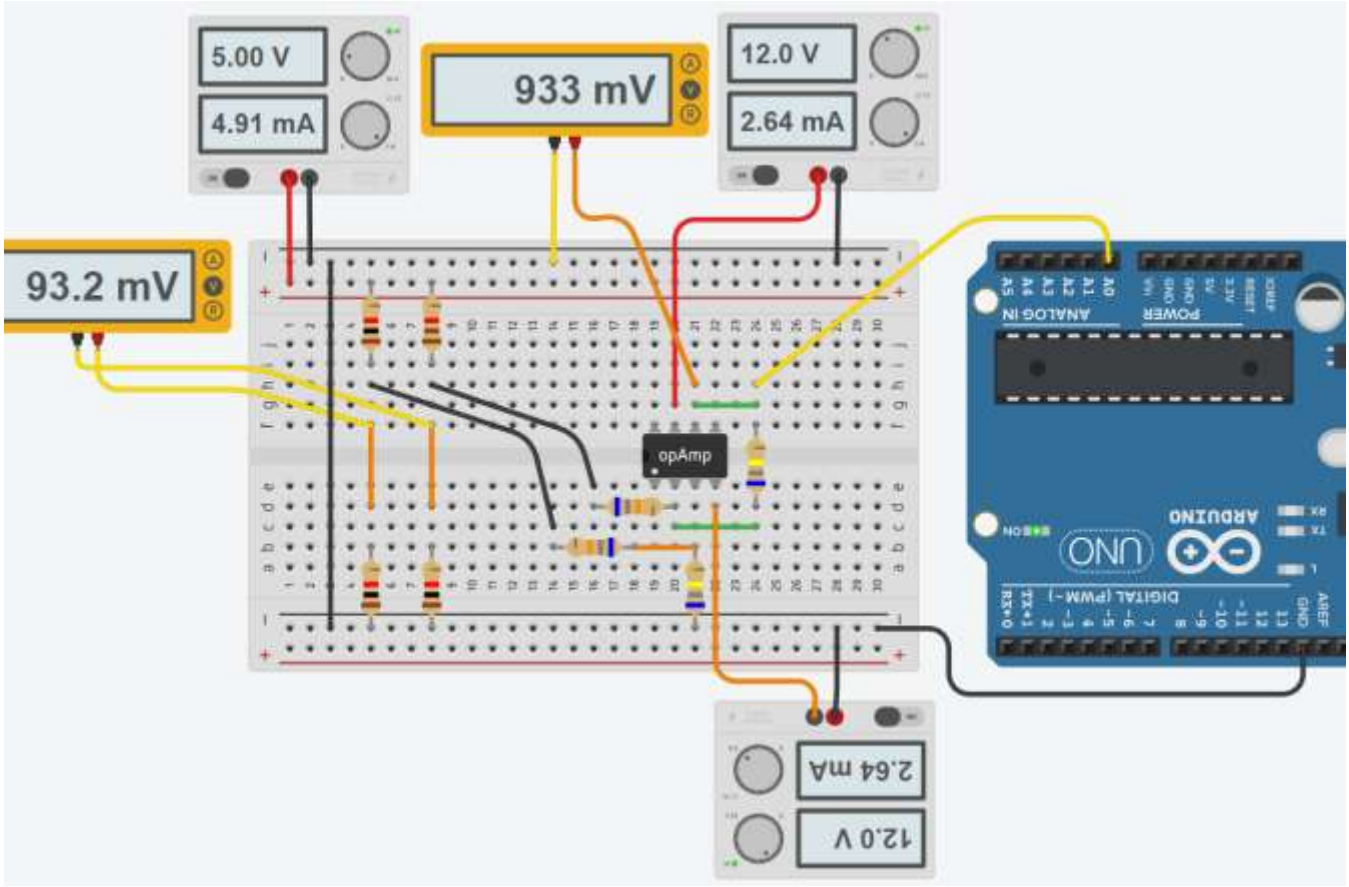
2- SIMULARE UN SISTEMA DI CONTROLLO CHE ATTIVA UN MOTORE (TRAMITE RELE') CHE ATTIVA IL MOTORE DELLA POMPA DELL'IMPIANTO DI RISCALDAMENTO QUANDO T<20°C

3- SIMULARE UN SISTEMA DI CONTROLLO CHE ATTIVA UN MOTORE (TRAMITE TRANSITOR) CHE REGOLA LA VELOCITA' DEL MOTORE DELLA POMPA DELL'IMPIANTO DI RISCALDAMENTO QUANDO T<20°C

TERMORESISTENZA PT1000 CON AMPLIFICATORI DIFFERENZIALE

Per gestire piccolo variazioni di temperature (e quindi tensioni dell'ordine di pochi mV) tramite un termistore è necessario utilizzare un circuito AMPLIFICATORE che amplifichi la differenza di tensione in uscita a un ponte di Wheatstone in cui è inserita la termoresistenza.
Si utilizza l'amplificatore operazionale in configurazione DIFFERENZIALE.

Le R devono essere molto alte, rispetto a quelle del ponte, per non disturbare la variazione di tensione letta dal ponte (es. 100K e 1000K → 93.2mV al posto di 93.8mV).



CODICE

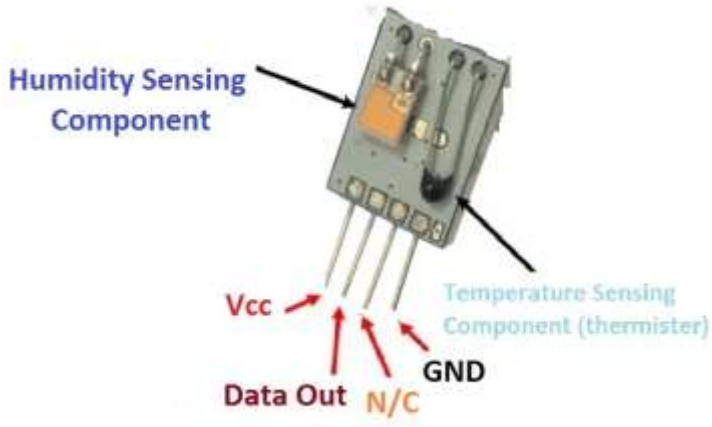
```
// PT1000 range 0 - 100 °C
// ponte Wheatstone
// AO amplificatore operazionale LM741

int amplificatoreV;
float deltaR;
int R=1000;
int E=5;
int guadagnoAmplificatore=10;
float Vab,Vo,Rpt1000, T;

void setup()
{
  Serial.begin(9600);
}

void loop()
{
 amplificatoreV = analogRead(A0); // 10 bits
 Vo = amplificatoreV/1024.00*5.0; // tensione in mV
 Vab = Vo/guadagnoAmplificatore;
 Serial.println(Vab);

deltaR= 2*R*(Vab/E)/(0.5-Vab/E);
 Serial.println(deltaR);

 Rpt1000 = R+ deltaR;
 Serial.println(Rpt1000);

 // pt1000:  Rpt1000=1000*(1+0.00385*T)
T = (Rpt1000/1000-1)/0.00385;
 Serial.print("T: ");
 Serial.print(T);
 Serial.print(char(176));
 Serial.println("C");

 delay(1000);
}
```

SENSORE DI UMIDITA' DHT22

Il DHT22 è un trasduttore di temperatura (da -40°C a +80°C) e umidità relativa (da 0% a 100%).
Dispone di interfaccia seriale a filo singolo che ne facilita l'utilizzo. Il sensore DHT22 viene calibrato in modo estremamente preciso, i coefficienti di calibrazione sono memorizzati nella memoria OTP e vengono richiamati durante il processo di rilevamento, in questo modo non vi è alcuna necessità di ricalibrare il sensore. È particolarmente adatto per prodotti di consumo, stazioni meteo, applicazioni HVAC (Heating, Ventilation and Air Conditioning), ecc.

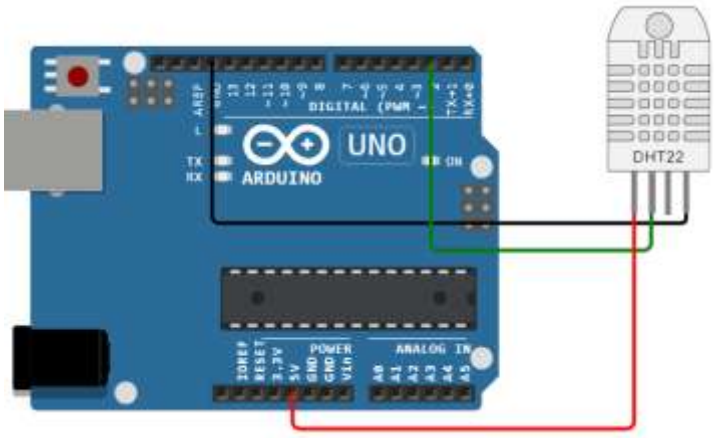
DATI TECNICI

-Alimentazione: da 3 a 5 VDC
-Consumo: max. 2,5 mA
-Range Umidità: da 0 a 100% con precisione del 2-5%
-Range di Temperatura: da -40°C a +80°C +-0,5°C di precis.
-Velocità di campionamento: <= 0,5 Hz (1 volta ogni 2 sec.)
-Uscita dati: seriale a filo singolo (non è Dallas One Wire)



CODICE

```
#include "DHT.h"

#define DHTPIN 2
#define DHTTYPE DHT22

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  dht.begin();
}

void loop() {
  float temperature = dht.readTemperature();
  float humidity = dht.readHumidity();

  // Check if any reads failed and exit early (to try again).
  if (isnan(temperature) || isnan(humidity)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }

  Serial.print(F("Humidity: "));
  Serial.print(humidity);
  Serial.print(F("%  Temperature: "));
  Serial.print(temperature);
  Serial.println(F("°C "));

  // Wait a few seconds between measurements.
  delay(2000);
}
```



simulabile su "wokwi.com"

SENSORI DI PROSSIMITA'

Un sensore di prossimità è un sensore elettrico che serve a rilevare la presenza di oggetti (specie metallici) posti nelle sue vicinanze. La rilevazione avviene sia se l'oggetto viene posto a contatto con il sensore di prossimità, sia a distanza.
La distanza di rilevazione viene detta "portata nominale" o "campo sensibile".
I sensori di prossimità possono essere utilizzati per migliorare le manovre di parcheggio o, nell'industria, per ottimizzare gli impianti di illuminazione, per monitorare le linee di produzione, per misurare il livello dei fluidi dall'esterno, per rilevare anomalie in condizioni difficili, per rendere più efficiente il controllo qualità e più sicuro l'accesso agli impianti.

COME FUNZIONA UN SENSORE DI PROSSIMITÀ

Come tutti i sensori, anche il sensore di prossimità interagisce con il fenomeno da misurare: è proprio questa interazione che causa una variazione della proprietà della grandezza e ne permette la misura. Il sensore di prossimità misura in ingresso la grandezza (es. oscillazione del campo magnetico) e acquisisce l'informazione rilevante per lo strumento terminale (es. un display) a cui è collegato o per i sistemi di regolazione e controllo in cui è inserito.

Nei sistemi IoT, i sensori di prossimità possono essere connessi tra loro tramite fili o reti wireless, ad esempio con la tecnologia LoRaWAN, in cui diventano dei "dispositivi periferici" che dialogano con dei concentratori (gateway) posizionati nell'ambiente. La rete raccoglie i dati dei sensori e li invia in cloud al network server che a sua volta li passa all'application server per renderli disponibili via app o web.

A seconda della tipologia di funzionamento, i sensori di prossimità si differenziano in induttivi, capacitivi, magnetici, a ultrasuoni, fotoelettrici o ottici.

TIPI DI SENSORI DI PROSSIMITÀ

Sensori di prossimità induttivi, cosa sono e funzionamento

I sensori di prossimità induttivi sfruttano l'induzione elettromagnetica, ovvero la corrente elettrica indotta che si genera quando, in un circuito chiuso, varia il campo magnetico. Infatti, ogni sensore di prossimità induttivo ha al suo interno: un circuito oscillante, un selettore di segnali e un amplificatore di commutazione.

Il circuito oscillante genera un campo elettromagnetico alternato ad alta frequenza, emesso sulla superficie attiva del sensore: in presenza di un oggetto o un materiale ferromagnetico, il campo varia e l'oggetto si "elettrizza". Le correnti indotte nell'oggetto abbassano la sua riluttanza, l'opposizione al flusso elettromagnetico: questa variazione è misurabile dal sensore di prossimità che a sua volta, a causa delle correnti indotte, avrà "perso" energia e oscillerà meno (altra variazione misurabile).

Il selettore di segnali converte queste variazioni in un segnale di commutazione univoco: il sensore è così in grado di calcolare sia la presenza che la distanza dall'oggetto-target. Oggetto che, nel caso dei sensori di prossimità induttivi, può essere solo ferromagnetico.

Sensori di prossimità capacitivi, cosa sono e funzionamento

I sensori di prossimità capacitivi prendono il nome dal condensatore (in inglese, capacitor), un componente elettrico composto da due armature, caricate con segno opposto, che immagazzina l'energia potenziale in un campo elettrico.

Semplificando, le due armature generano un campo elettrico: se un'armatura è il sensore, l'eventuale oggetto nelle vicinanze diventa l'altra. La corrente che viene generata cambia la distanza tra le due, una distanza che può essere rilevata e misurata. La misura è più accurata se l'oggetto-target è piatto e parallelo al sensore.

Rispetto ai sensori di prossimità induttivi, i sensori di prossimità capacitivi possono segnalare la presenza di qualsiasi oggetto, non solo ferromagnetico.

Sensori di prossimità magnetici, cosa sono e come funzionano

I sensori di prossimità magnetici rilevano la presenza e la distanza dell'oggetto da misurare in base al campo generato dal magnete posto sull'oggetto stesso: si basano sul principio dei contatti Reed o sull'effetto Hall.

Un contatto Reed è un interruttore formato da due lamine ferromagnetiche separate e in parte sovrapposte, custodite in un bulbo di vetro riempito di gas inerte: in presenza di un campo magnetico, alle estremità delle lamine si formano poli di segno opposto che tendono ad attrarsi, quindi a chiudere il circuito. I sensori con all'interno un contatto Reed sono usati come base per i sistemi antifurto: si posiziona il sensore sullo stipite di un infisso e un magnete sul suo bordo. Se la porta (o finestra) è chiusa, il circuito resta chiuso; nel caso in cui la porta si aprisse in modo non autorizzato, l'interruzione del circuito magnetico farebbe scattare l'allarme.

Altri sensori di prossimità magnetici, più propriamente trasduttori, funzionano invece con l'effetto Hall, ovvero con la differenza di tensione elettrica che genera un conduttore attraversato da corrente elettrica quando è sottoposto a un campo magnetico.

Sensori di prossimità ad ultrasuoni cosa sono e come funzionano

I sensori di prossimità ad ultrasuoni utilizzano le onde sonore con frequenza superiore ai 20.000 Hz, oltre l'intervallo udibile dall'orecchio umano, per misurare la distanza da un bersaglio specifico e rilevarne quindi la presenza o assenza. In pratica, funzionano sul principio del sonar: il sensore emette un'onda di ultrasuoni ad una specifica frequenza verso l'eventuale target e aspetta il tempo necessario per l'eco di ritorno. Durante questo intervallo, misura la presenza e la distanza. I sensori di prossimità ad ultrasuoni sono spesso dotati di un software che ne programma il settaggio, distanza e campo sensibile compresi.

Sensori di prossimità ottici o fotoelettrici, cosa sono e funzionamento

I sensori di prossimità ottici o fotoelettrici, più propriamente trasduttori, rilevano i raggi luminosi e li trasformano in segnali elettronici. Un sensore di prossimità ottico è composto generalmente da una sorgente luminosa (es. Led) e da un ricevitore (es. fotodiodo): la misurazione avviene quando l'oggetto interrompe o riflette la quantità di luce emessa. Generalmente, il fascio di luce emesso è a infrarossi per non confonderlo con altre fonti di luce ambientale.

Tra i più comuni sensori ottici di prossimità troviamo le fotocellule, o fotorivelatori, che si differenziano a seconda della posizione dell'oggetto da rilevare rispetto a emettitore e ricevitore: in mezzo (sistema a barriera), di fronte (sistema a catarifrangente), orientata verso il ricevitore (sistema a riflessione diretta). In casi particolari, per trasportare la luce dall'emettitore al ricevitore, al posto dei componenti fotoelettrici classici si utilizza la fibra ottica.

COME SCEGLIERE UN SENSORE DI PROSSIMITÀ E PERCHÉ

Prima di scegliere un sensore di prossimità, è consigliabile riflettere sia sulle caratteristiche dell'oggetto da rilevare/misurare, come ad esempio la forma, lo stato, il materiale, sia sulla distanza tra il sensore e l'oggetto.

- I sensori di prossimità induttivi rilevano a una distanza fino a 80 millimetri, non si usurano facilmente, sono particolarmente resistenti agli urti, alle vibrazioni e alla polvere, oltre ad avere una frequenza di commutazione sufficiente a monitorare il passaggio rapido di oggetti anche in rotazione. Sono quindi indicati per distanze di misurazione brevi e per il controllo di linee di produzione in ambienti industriali su oggetti esclusivamente ferromagnetici e metallici.

- I sensori di prossimità capacitivi hanno una portata ancora minore, fino a 60 millimetri, ma un costo leggermente superiore, perché sono in grado di rilevare qualsiasi tipo di oggetti, non solo ferromagnetici. Non si usurano facilmente e sono immuni a disturbi elettromagnetici ma non sono adatti ad ambienti umidi o dai vapori densi, condizioni alle quali sono particolarmente sensibili. Sono quindi indicati per rilevazioni a distanze molto brevi e, nell'industria, per il

rilevamento a poca distanza del livello di liquidi nei contenitori.

- I sensori di prossimità magnetici hanno una portata leggermente superiore rispetto agli induttivi e ai capacitivi, fino a 100 millimetri: la portata dipende dalla potenza del campo generato dal magnete, quindi dalla sua grandezza. I più economici sensori di prossimità non si usurano e non sono sensibili alle vibrazioni. Sono indicati per la rilevazione di oggetti esclusivamente magnetici o magnetizzati in ambienti anche industriali ma lontano da fonti elettromagnetiche come motori o linee di alimentazione.

- I sensori di prossimità a ultrasuoni arrivano fino a 15 metri di portata. Dal costo elevato, possono rilevare oggetti di qualsiasi materiale e forma tranne i fonoassorbenti. Il loro tempo di risposta dipende dalla velocità di propagazione del suono nell'aria. Sono indicati per la rilevazione di oggetti in ambienti non sottoposti a sbalzi di temperatura e correnti d'aria.

- I sensori di prossimità ottici o fotoelettrici rilevano oggetti fino a 200 metri, compresi i materiali trasparenti. Dal costo medio-alto, non sono indicati per ambienti polverosi o in cui non sia possibile mantenere uno standard di pulizia elevato.

ESEMPI E AMBITI APPLICATIVI DEI SENSORI DI PROSSIMITÀ

I sensori di prossimità vengono utilizzati per misurazioni accurate, tempestive e in tempo reale, che aiutano a ottimizzare processi e risorse. Sensori di prossimità inseriti in sistemi Industrial IoT consentono di aumentare la sicurezza, evitare sprechi, costruire database di misurazioni utili a prendere decisioni.

I sensori di prossimità induttivi per la loro resistenza sono usati nelle macchine utensili, nelle linee di assemblaggio, per il rilevamento di oggetti ferromagnetici in condizioni difficili, per il monitoraggio dello scorrimento di oggetti ferromagnetici sulle linee di produzione, per il rilevamento della presenza dei sigilli nel packaging o per la misurazione dello spessore delle bobine in movimento.

I sensori di prossimità capacitivi sono particolarmente utilizzati nelle tarature, nella misurazione e nel monitoraggio del riempimento di fluidi nei contenitori, quindi negli impianti di confezionamento nonché nelle fasi finali di controllo dei processi di imballaggio.

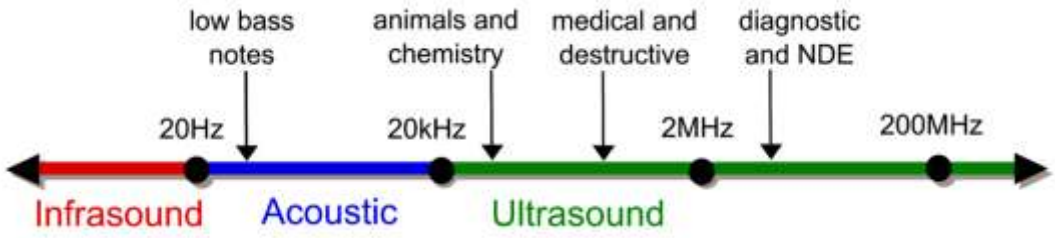
I sensori di prossimità magnetici vengono utilizzati per la sicurezza degli impianti, inseriti in dispositivi anti-infrazione o per le rilevazioni in ambienti difficili, grazie all'immunità a polvere, calore e vibrazioni.

I sensori di prossimità a ultrasuoni sono usati per le rilevazioni di oggetti trasparenti a lunga distanza, per la misurazione del livello di liquidi e granulati nei contenitori, per il monitoraggio del flusso dei nastri trasportatori degli impianti, per la rilevazione delle anomalie nelle vasche di alimentazione o nell'assemblaggio delle componenti automobilistiche, oltre che inseriti come sensori di parcheggio.

I sensori di prossimità ottici o fotoelettrici per la loro significativa portata vengono usati in tutte le situazioni in cui sia necessario rilevare la presenza di persone, veicoli o animali: dalla movimentazione al trasporto, dalla robotica al settore edile, a qualsiasi impianto che necessiti di sensori per la rilevazione dei pezzi prodotti.

SENSORE A ULTRASUONI

Gli ultrasuoni sono onde sonore con frequenze superiori a quelle udibili dall'orecchio umano: stiamo quindi parlando di frequenze che superano i 20 kHz e che trovano impiego per lo più in campo medico ed industriale.



utilizzare un sensore ad ultrasuoni come misuratore di distanze, in attività didattiche dove non sia richiesta un'elevata "qualità della misura".
Lasciamo quindi a successivi sviluppi la ricerca di misure precise ed accurate, per le quali dovremmo considerare la velocità istantanea del suono che è influenzata, principalmente, dalla temperatura e dall'umidità relativa del mezzo.

Per quanto premesso possiamo assumere come costante il valore della velocità del suono in un determinato mezzo, in particolare l'aria, dove le onde sonore viaggiano a 343,8 m/s a 20°C.

Anche gli ultrasuoni, come tutte le onde, sono soggetti a fenomeni di riflessione. Questa caratteristica ci permette di utilizzare il sensore per rilevare misure di distanza tra la sorgente emettitrice del segnale sonoro e l'oggetto colpito.

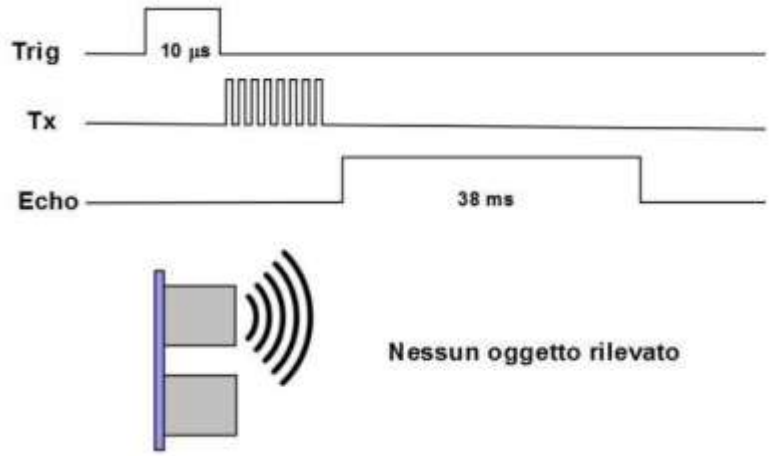
FUNZIONAMENTO DEL SENSORE PER ARDUINO

Un impulso a 5 volt di almeno 10 µS (microsecondi) di durata viene applicato al pin Trigger.

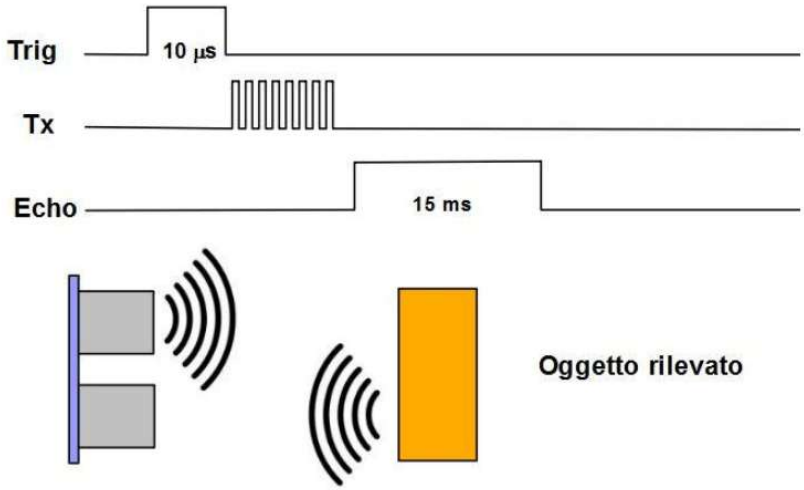
Si genera un treno di 8 impulsi ultrasonici a 40 KHz che si allontanano dal sensore viaggiando nell'aria circostante.
Si ottengono misure più accurate se l'ostacolo si trova di fronte al sensore o in un ipotetico settore circolare di 30° d'ampiezza (15° da ambo i lati rispetto alla direzione frontale).

Il segnale sull'Echo intanto diventa alto ed inizia la registrazione del tempo di ritorno in attesa dell'onda riflessa.

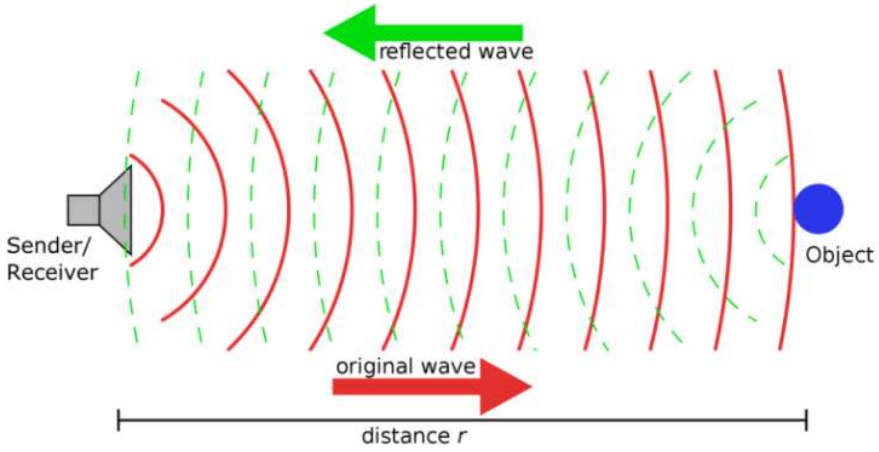
Se l'impulso non viene riflesso il segnale su Echo torna basso dopo 38 ms (millisecondi) e va interpretato come assenza di ostacolo. Ricordiamo l'HC-SR04 è in grado di misurare distanze comprese tra i 2 e i 400 cm corrispondenti, per il limite massimo, a circa 23 ms di durata del segnale su Echo.



5. Se invece il treno di onde ultrasoniche viene riflesso all'indietro da un oggetto, il segnale sul pin Echo diventa basso e contestualmente termina il rilievo della sua durata.



6. Il tempo ottenuto servirà per calcolare la distanza dell'oggetto: bisogna però tenere presente che l'onda ha percorso per due volte quella distanza, quando emessa verso l'oggetto e dopo la riflessione verso il sensore. Bisognerà quindi dividere per due la distanza calcolata con questo tempo.

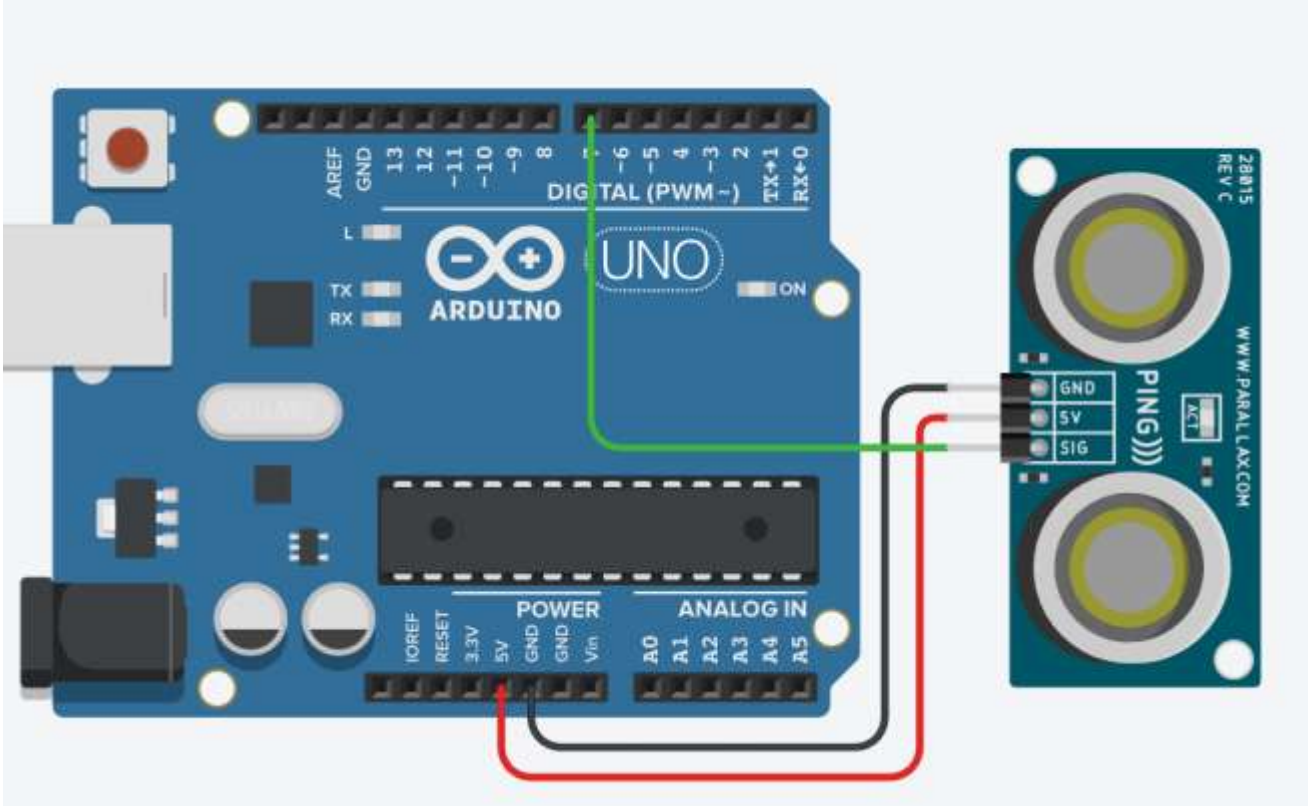


La velocità del suono nell'aria è di circa 343 m/s.
La funzione **pulseIn()** introdotta nello sketch ci permette di ottenere la durata dell'impulso ALTO sul pin Echo in microsecondi.

$$343\ \frac{m}{s} = \frac{34300}{1000000}\ \frac{cm}{\mu s} = 0{,}0343\ \frac{cm}{\mu s}$$

$$distanza\ [cm] = \left(0{,}0343\ \left[\frac{cm}{\mu s}\right] \cdot\ durata\ ALTO\ su\ Echo\ [\mu s]\right) \div 2$$

ESERCIZIO MISURA DISTANZA CON SENSORE ULTRASUONI



CODICE

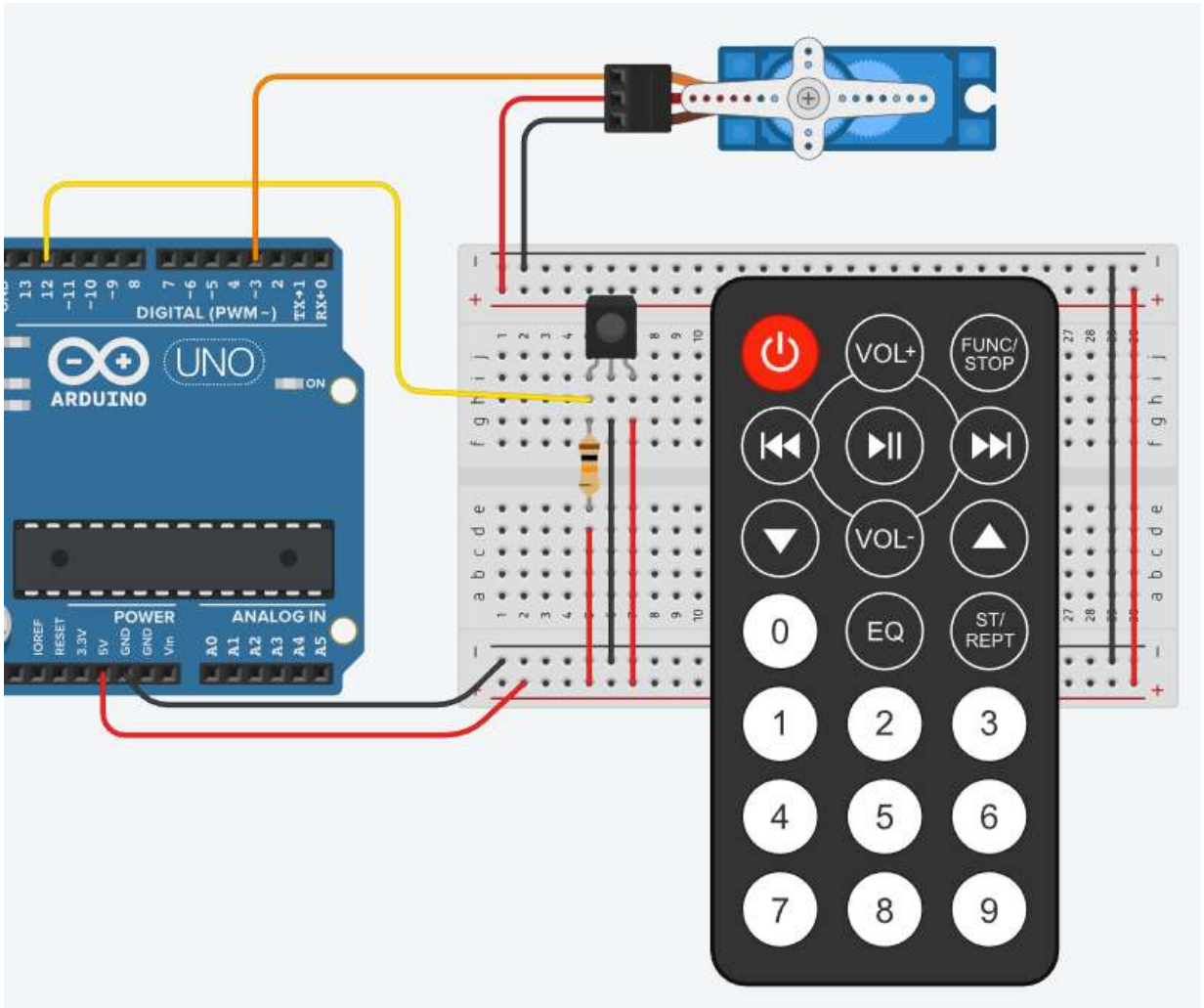
```
int cm = 0;

void setup()
{
  Serial.begin(9600);
}

void loop()
{
  // measure the ping time in cm
  cm = readUltrasonicDistance(7, 7);
  Serial.print(cm);
  Serial.println("cm");
  delay(500); // Wait for 100 millisecond(s)
}

long readUltrasonicDistance(int triggerPin, int echoPin)
{
  pinMode(triggerPin, OUTPUT);  // Clear the trigger
  digitalWrite(triggerPin, LOW);
  delayMicroseconds(2);
  // Sets the trigger pin to HIGH state for 10 microseconds
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);
  pinMode(echoPin, INPUT);
  // Reads the echo pin, and returns the sound wave travel time in microsecondo
  // measure the ping time in cm → d= velocità suono * tempo
  return (0.01723 * pulseIn(echoPin, HIGH));
}
```

SENSORE IR (INFRAROSSI)

I ricevitori IR (a infrarossi) sono dispositivi progettati per inviare e ricevere un segnale a infrarossi (radiazione luminosa non visibile all'occhio umano) codificato da un dispositivo all'altro.

ESERCIZIO COMANDO SERVOMOTORE CON TELECOMANDO IR

Si vuole comandare la posizione di un servo motore (0°, 30°, 60° e 90°) tramite un telecomando e un ricevitore ad infrarossi.



CODICE

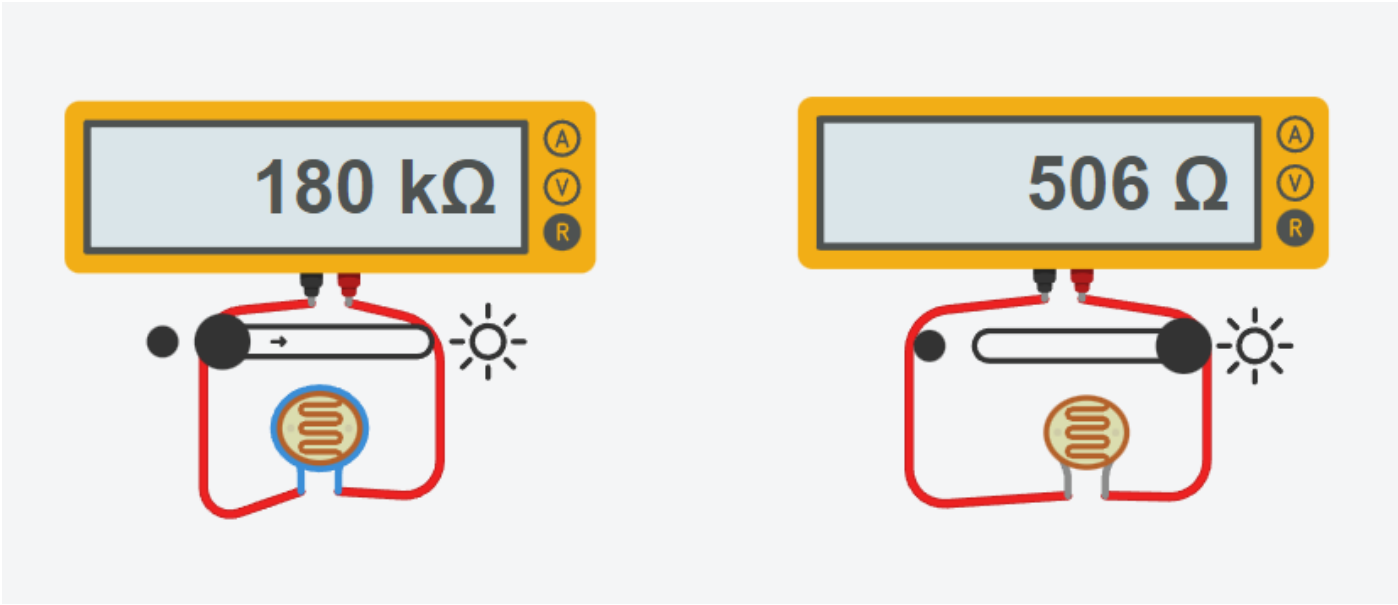
```
#include <IRremote.h>
#include <Servo.h>

Servo servo1;
int servo1pos = 0; // variable to store the servo position
int servo1Pin = 3; // define the pin of servo signal line

// Infrared receiving pin
int RECV_PIN = 12;
IRrecv irrecv(RECV_PIN); // Create a class object used to receive class
decode_results results; // Create a decoding results class object

// COMANDI
long ir_uno = 16582903;   // numero intero ricevuto quando si preme 1 sul telecomando
long ir_due = 16615543;
long ir_tre = 16599223;
long ir_stop = 16597183;
long ir_piu  = 16613503;
long ir_meno = 16617583;
long ir_sx   = 16589023;
long ir_dx   = 16605343;
long ir_su   = 16601263;
long ir_giu  = 16584943;
long ir_eq   = 16625743;
long ir_rept = 16609423;


void setup()
{
  Serial.begin(9600);
  Serial.println("Servo pronto!");

  // Start the receiver
  irrecv.enableIRIn();

  servo1.attach(servo1Pin);
  servo1.write(0);
}

void loop() {
  // Attendo comando da telecomando IR
  if (irrecv.decode(&results)) { // Waiting for decoding
    Serial.println(results.value); // Print out the decoded results

    if (results.value == ir_uno) {
      Serial.println("UNO");
      servo1.write(30);
    }
    else if (results.value == ir_due) {
      Serial.println("DUE");
      servo1.write(60);
    }
    else if (results.value == ir_tre) {
      Serial.println("TRE");
      servo1.write(90);
    }

    else if (results.value == ir_stop) {
      Serial.println("STOP");
      servo1pos= 0;
      servo1.write(0);
    }

    irrecv.resume(); // Receive the next value
  }

  delay(100);
}
```

FOTORESISTENZA

La fotoresistenza è un componente elettronico la cui resistenza è inversamente proporzionale alla quantità di luce che lo colpisce. Si comporta come un normale resistore, ma il suo valore in ohm diminuisce a mano a mano che aumenta l'intensità della luce che la colpisce.

ESERCITAZIONE ARDUINO

Rilevare il campo di variazione della resistenza del sensore.
Le misure della resistenza del sensore vanno fatte scollegate dall'alimentazione!

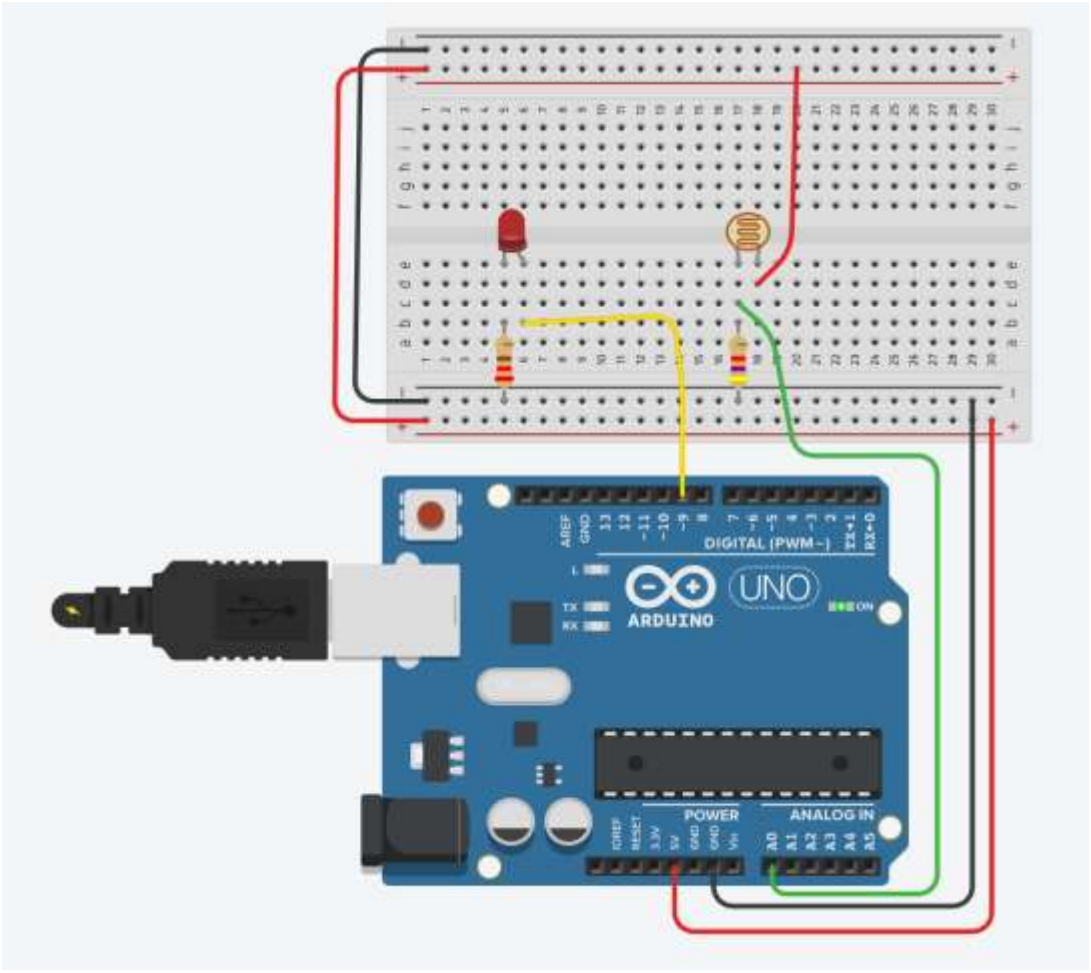


Al buio

Alla massima luminosità

ESERCIZIO MISURA LUMINOSITA' FOTORESISTENZA

Regolare la luminosità di un diodo led in modo proporzionale alla intensità luminosa rilevata col sensore.



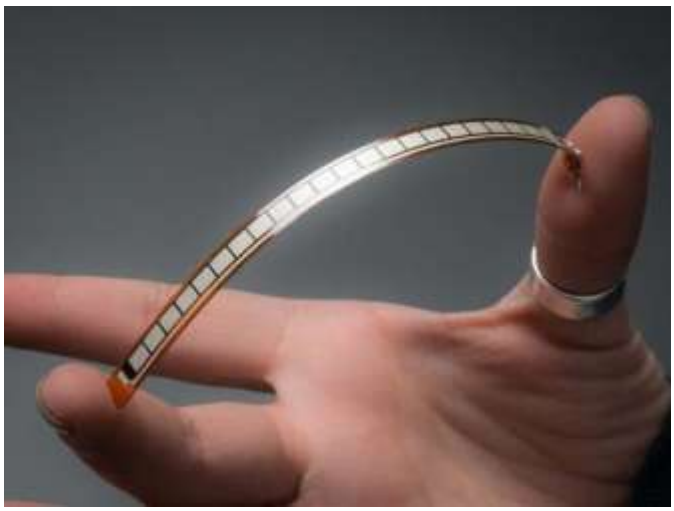
Resistenza da 4.7K in serie alla foto resistenza.

CODICE

```
int sensorValue = 0;

void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(9600);
  pinMode(9, OUTPUT);
}

void loop()
{
  // read the value from the sensor
  sensorValue = analogRead(A0);
  // print the sensor reading so you know its range
  Serial.println(sensorValue);
  // map the sensor reading to a range for the LED
  analogWrite(9, map(sensorValue, 0, 1023, 0, 255));
  delay(100); // Wait for 100 millisecond(s)
}
```

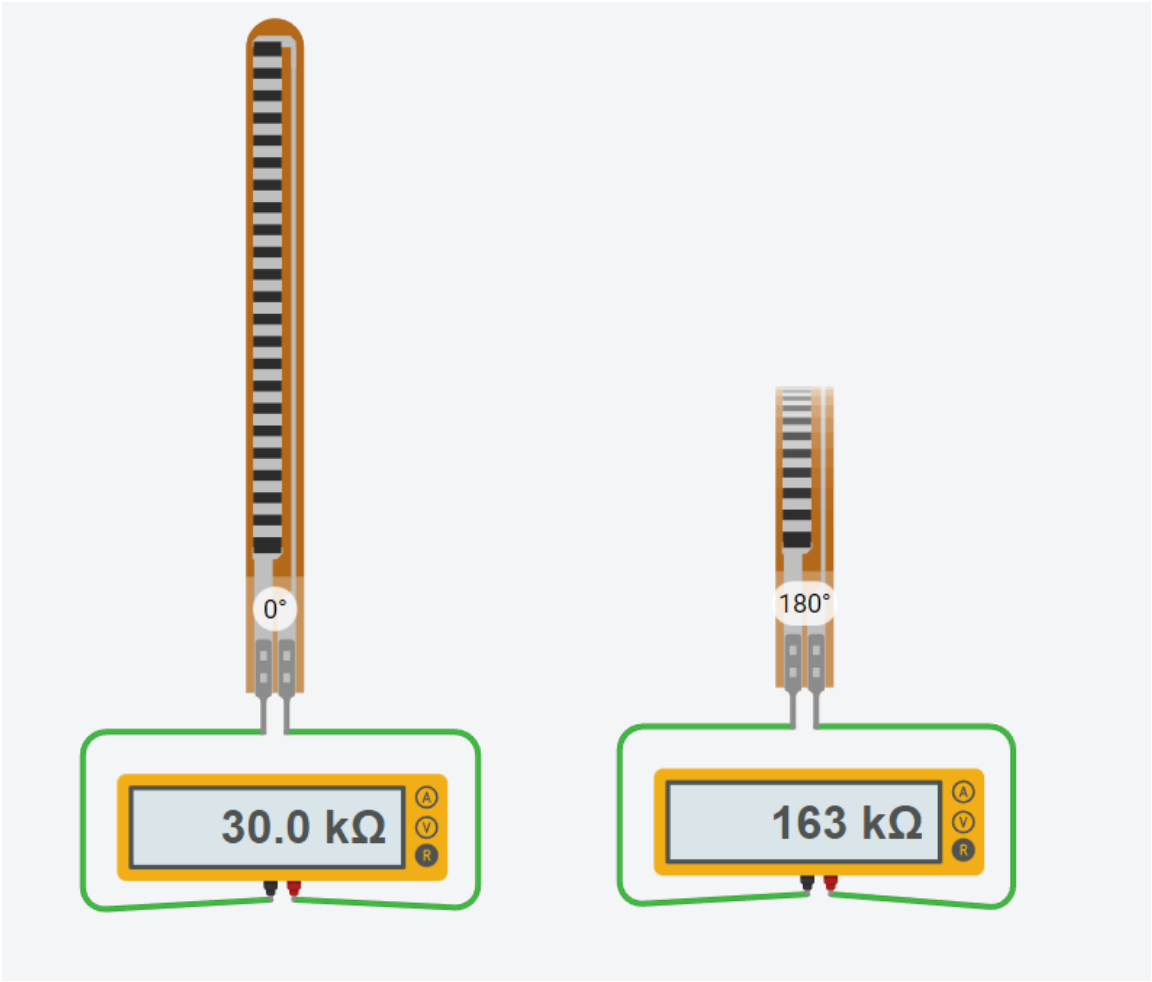
SENSORE DI FLESSIONE ANGOLARE (FLEX SENSOR)

Il sensore di flessione (resistivo) contiene un inchiostro polimerico a base di particelle conduttive.
Quando il sensore è piegato, le aree si allontanano, riducendo la conduttività e quindi aumentando la resistenza elettrica.

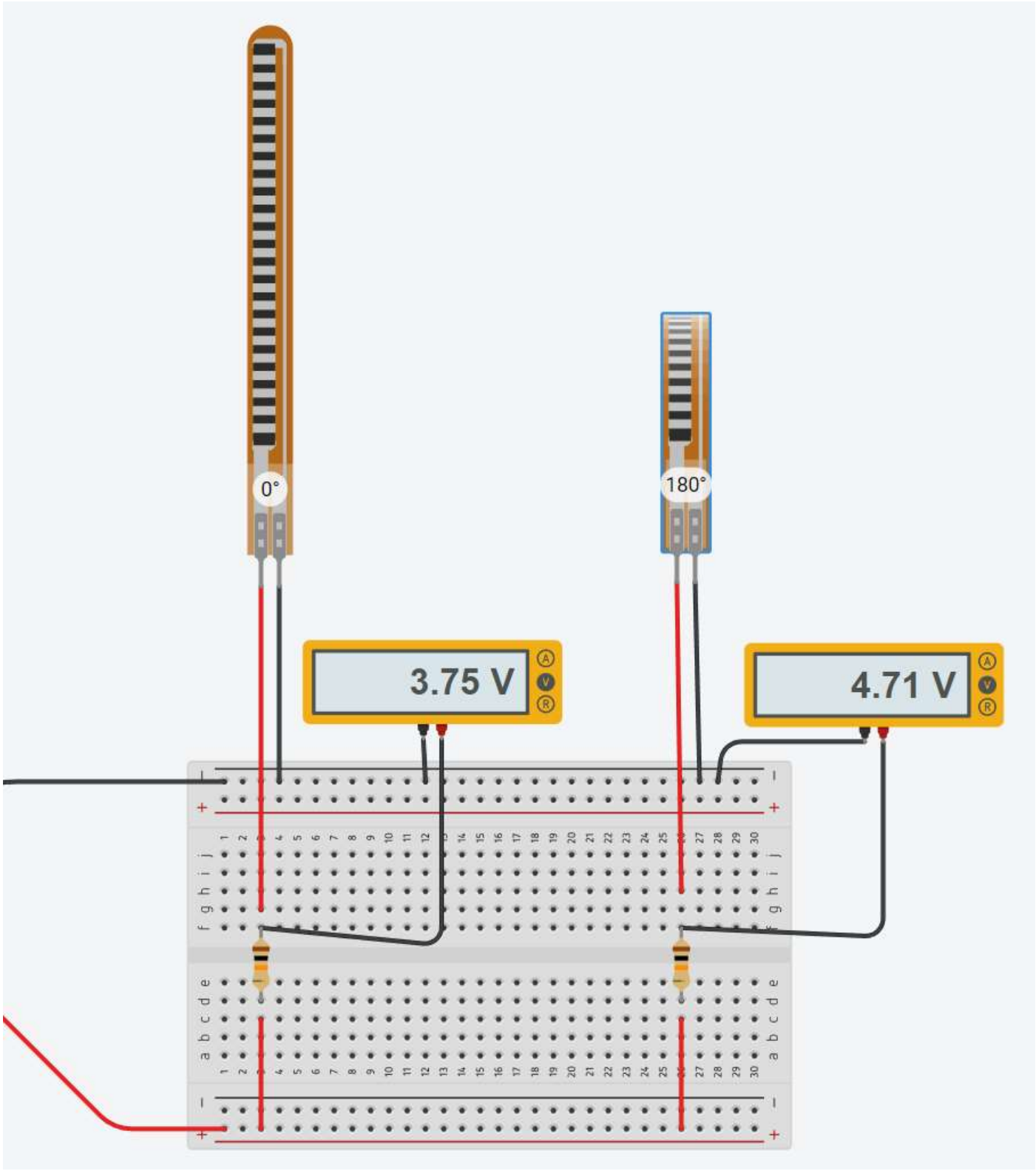


ESERCITAZIONE ARDUINO

Rilevare il campo di variazione della resistenza del sensore.
Le misure della resistenza del sensore vanno fatte scollegate dall'alimentazione!



Rilevare la variazione di tensione dovuta alla variazione di resistenza del sensore (potenziometro).

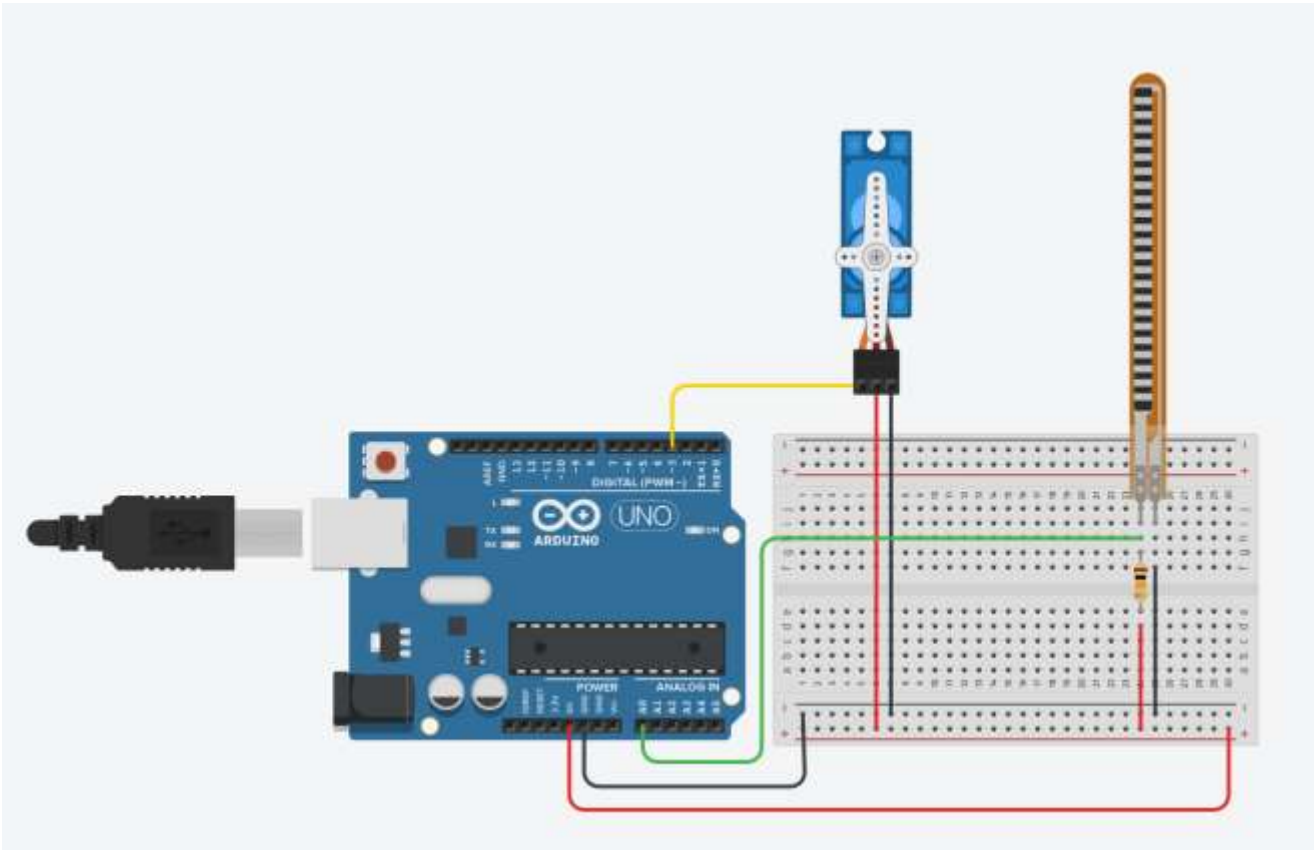


Il campo di variazione della tensione è di 0.96V.

ESERCIZIO CON FLEX SENSOR

Si vuole comandare un servo motore tramite un sensore di forza flessibile.
Una applicazione tipica è quella che consente di comandare un robot tramite il movimento delle dita di una mano che indossa un guanto dotato di sensori flessibili su ogni dito.

Componenti: 1 servo SG90, 1 resistenza da 10K e 1 sensore di flessione da 30K a riposo



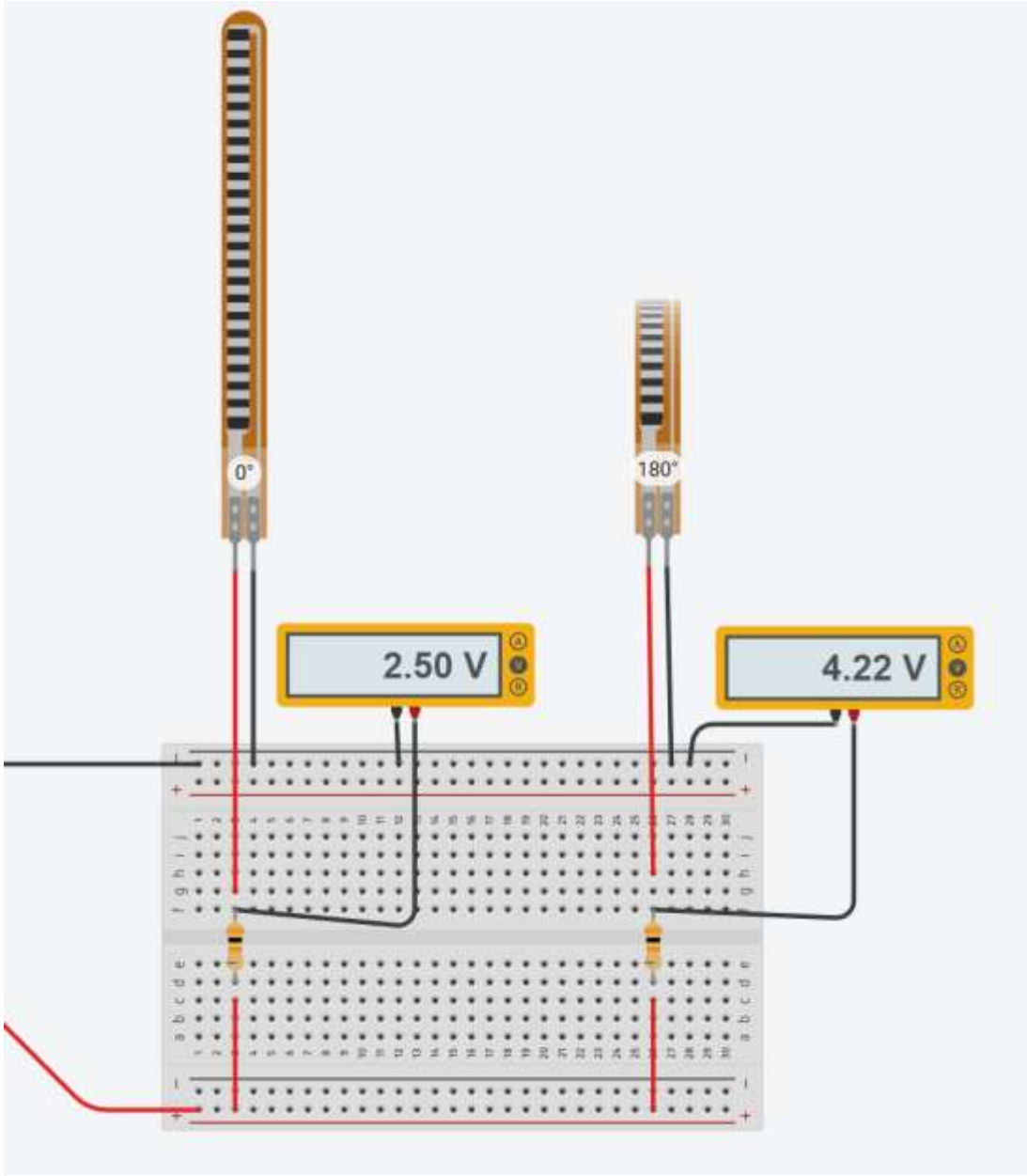
CODICE

```
#include <Servo.h>
Servo myServo;
# define flexPin A0
void setup()
{
  myServo.attach(3);
  Serial.begin(9600);
}

void loop()
{
  int flexValue;
  int servoPosition;
  flexValue = analogRead(flexPin);
  servoPosition = map(flexValue, 770, 950, 0, 180);
  servoPosition = constrain(servoPosition, 0, 180);

  myServo.write(servoPosition);

  Serial.print("sensor =");
  Serial.println(flexValue);
  Serial.print("servo =");
```

```
  Serial.println(servoPosition);

  delay(20);
}
```

COMPITO

1- Estendere il progetto in modo da gestire due servo motori con due sensori di flessione
2- Utilizzare il sensore di flessione per comandare il numero di giri di un motore DC 5V.
3- Valutare un valore di resistenza che permetta una maggiore variazione di tensione in ingresso ad Arduino (maggiore precisione)



Il campo di variazione della tensione è di 1.72 Volt rispetto ai 0.96V di prima.
Modificare il programma di conseguenza.

ESTENSIMETRI INDUSTRIALI

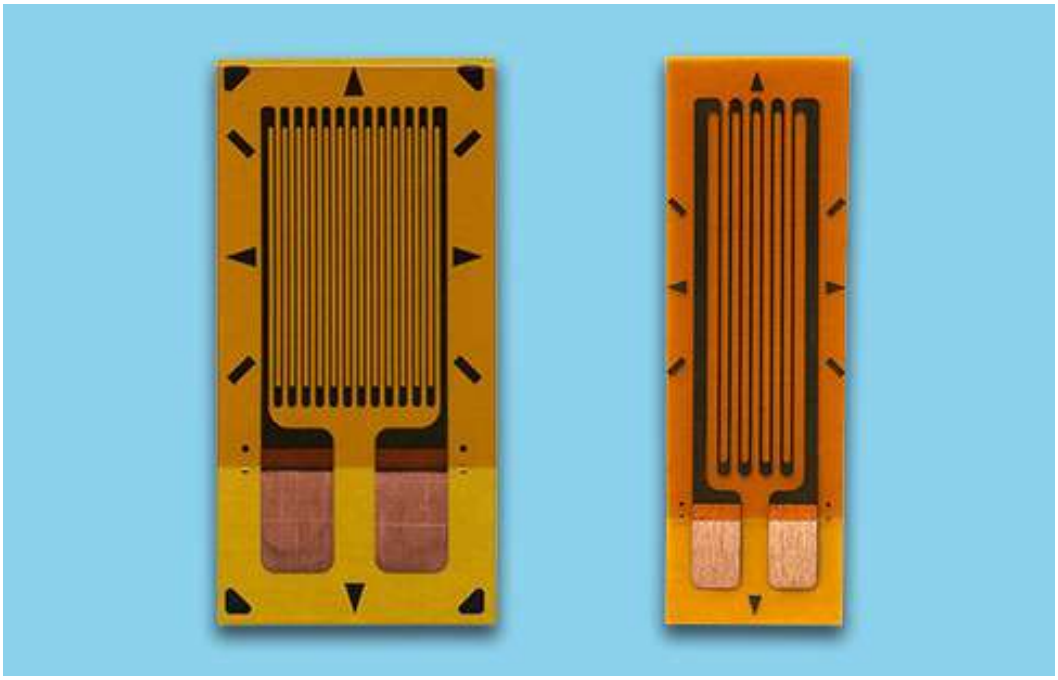
L'estensimetro è uno strumento di misura utilizzato per rilevare piccole deformazioni dimensionali di un corpo sottoposto a sollecitazioni meccaniche o termiche (es. applicazione di carichi o variazioni di temperatura).

Conoscendo a priori le caratteristiche meccanico/fisiche del materiale, misurando le deformazioni si possono facilmente ricavare i carichi a cui il materiale è sottoposto.
Inoltre, utilizzando estensimetri di giusta tipologia e applicandoli in modo oculato, si possono rilevare la direzione e il verso di queste deformazioni, e di conseguenza il vettore delle forze applicato al materiale sotto esame.

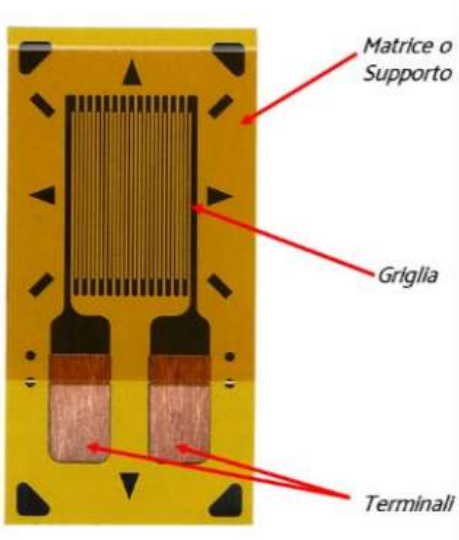
I campi d'applicazione sono molteplici:

- prove in laboratorio su componenti meccanici o materiali;
- analisi statiche e dinamiche di componenti o sistemi meccanici già montati in situ;
- monitoraggio di componenti o sistemi strutturali;
- elemento sensore per trasduttori di grandezze meccaniche;

misure del carico assiale su braccetti di sterzo con estensimetri elettrici a resistenza



Gli elementi principali di un estensimetro sono la matrice e la griglia.



I materiali più comuni con cui si realizzano le griglie sono:

Tipo di lega	Composizione %
Costantana	45Ni, 55Cu
Karma	74 Ni, 20 Cr, 3 Fe, 3 Al
Isoelastica	36 Ni, 55,5 Fe, 8 Cr, 0,5 Mo
Platino Tungsteno	92 Pt, 8 W
Nicromo	80 Ni, 20 Cr
Kanthal	30 Cr, 30 Fe, 7,5 Al, 32,5 Co

Le griglie sono disponibili in lunghezze che variano da 0,2 mm a 120 mm.
Le matrici (dette anche supporto) vengono realizzate con delle resine anche rinforzate con fibra di vetro per migliorarne le prestazioni alle alte temperature.

Tipi di Resine	Temperatura di Utilizzo
Resina Epossidica	da -50 a 100 °C
Resina Fenolica	da -200 a 250 °C
Resina Poliammidica	da -200 a 200 °C
Resina Epossidica + Fibra di vetro	da -269 a 230 °C
Resina Fenolica + Fibra di vetro	da -200 a 300 °C
Resina Epossidica Fenolica + Fibra di vetro	da -269 a 300 °C
Piastrina metallica	da -269 a 300 °C

La piastrina metallica è il supporto tipico degli estensimetri saldabili applicati alle superfici con una saldatura per punti.

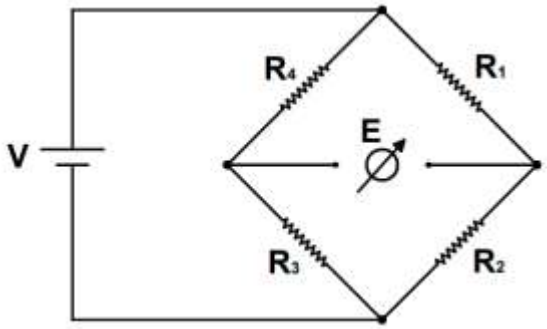
RESISTENZA DEGLI ESTENSIMETRI

Gli estensimetri sono disponibili con resistenze da 120, 350 e 1000 Ohm.
Per le classiche misure di stress analysis si possono usare sia i 120 che i 350 Ohm (questi ultimi soprattutto su materiali cattivi conduttori di calore).
Per realizzare i trasduttori estensimetrici si usano sia i 350 che i 1000 Ohm.

La variazione di resistenza elettrica di un estensimetro deformato è dell'ordine delle frazioni di Ω .
Queste variazioni di resistenza vanno misurate su circuiti che hanno resistenze assolute di centinaia di Ω .
Ciò richiede l'utilizzo di un particolare circuito di misura, detto circuito a ponte o ponte di Wheatstone.

IL PONTE DI WHEATSTONE

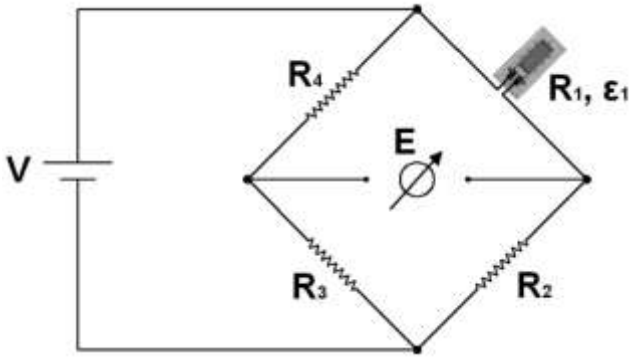
E' costituito da 4 resistenze elettriche che occupano 4 lati di un rombo.
Il ponte viene alimentato da una tensione V ai capi della «diagonale di alimentazione».
Ai capi dell'altra diagonale, detta «diagonale di misura», si misura la tensione di sbilanciamento Eponte.



$$E = V\left(\frac{R_1}{R_1+R_2} - \frac{R_4}{R_3+R_4}\right) = V\frac{R_1R_3 - R_2R_4}{(R_1+R_2)(R_3+R_4)}$$

COLLEGAMENTO A QUARTO DI PONTE

Si utilizza UN solo estensimetro che occupa uno dei lati del ponte.



$$Eponte = \frac{Vcc}{4} * \frac{\Delta R}{R}$$

Scegliendo le alter 3 resistenze R2,R3 e R4 (di precisione) di valore uguale a quella dell'estensimetro a riposo l'equazione del ponte si semplifica nella seguente formula (valida per variazioni piccole rispetto alle R):

$$Eponte = \frac{Vcc}{4} * \frac{\Delta R}{R}$$

con R=R1=R2=R3=R4 e ΔR= R1-$Rg_{nominale}$ e Rg = resistenza iniziale dell'estensimetro [Ω]

LEGAME DEFORMAZIONE ELASTICA E VARIAZIONE DI RESISTENZA ELETTRICA

Il legame tra deformazione e variazione di resistenza elettrica si esprime con la seguente relazione:

$$\varepsilon = \frac{1}{k} \cdot \frac{\Delta R}{R_g}$$

ε = deformazione [µm]
k = gage factor
Rg = Resistenza iniziale dell'estensimetro [Ω]
ΔR = Variazione di Resistenza dell'estensimetro [Ω] = Rf – Rg
Rf = Resistenza finale dell'estensimetro [Ω]

Il gage factor K (anche detto fattore di taratura o sensibilità alla deformazione) è una quantità adimensionale che viene ottenuta sperimentalmente. I valori tipici del gage factor in funzione del tipo di griglia sono rappresentati nella Tabella:

Tipo di Lega	Gage Factor Nominale a +24°C
Costantana	2.0
Karma	2.1
Isoelastica	3.2
Nicromo	2.0
Platino Tungsteno	4.0
Kanthal	2.4

MISURA DELLA DEFORMAZIONE E DELLA FORZA ASSIALE

Su un provino di lunghezza L_0 applichiamo una forza di trazione F.
Per effetto della forza il provino si allunga. ΔL rappresenta la variazione di lunghezza.
Se sulla superficie del provno è incollato l'estensimentro come disposto in figura allora anch'esso subirà la stessa deformazione.
Il rapporto tra la variazione di lunghezza ΔL e la lunghezza iniziale L è nota come deformazione meccanica e viene indicata con la lettera greca ε.

$$\frac{\Delta L}{L} = \varepsilon \qquad \left[\frac{m}{m}\right]$$

Essendo il rapporto tra due lunghezze la deformazione **ε** è una quantità adimensionale.
Essendo la ε una quantità piccola si preferisce usare un sottomultiplo del metro e cioè il µm = 10^{-6} m.
Per cui la deformazione viene espressa in µm/m. È ormai diventato di uso comune esprimere questa quantità in µε.
Spesso la deformazione si trova anche espressa in %.
Così è del tutto equivalente scrivere:

$$200 \cdot 10^{-6}\ \frac{m}{m} = 200\ \frac{\mu m}{m} = 200\ \mu\varepsilon = 0{,}02\%$$

Se l'estensimetro è collegato a quarto di ponte si ha:

$E\ ponte = \frac{Vcc}{4} * \frac{\Delta R}{R}$ da cui si ricava la $\Delta R = \frac{4*E*R}{V}$

Nota la ΔR si ricava la ε dalla relazione $\varepsilon = \frac{1}{k} \cdot \frac{\Delta R}{Rg}$ ed quindi la deformazione ΔL= ε * L

Secondo la legge di Hooke, carico specifico e allungamento unitario per piccoli valori del carico sono proporzionali ed il loro rapporto è definito come il modulo di Young o modulo di elasticità E lineare:

$$E = \frac{\sigma}{\varepsilon} \rightarrow \varepsilon = \frac{1}{E}\sigma \rightarrow \frac{\Delta L}{L} = \frac{F}{ES} \rightarrow$$

$$F = E * S * \frac{\Delta L}{L}$$

ESERCIZIO

PROVINO in ACCIAO sottoposto a trazione

E	206000	N/mm2
Sez. 20x5mm	100	mm2
L	100	mm

ESTENSIMETRO in Platino collegato a quarto di ponte

Vcc	5	V
Rg	120	ohm
k	4,000	
E ponte	0,5	mV

Variazione di resistenza elettrica

ΔR	0,048	ohm

Deformazione relativa

				μm/m
ε	0,0001	m	100	=με

Allungamento

ΔL	0,00001	m	0,01	mm

Forza di trazione

F	2060	N

AMPLIFICATORE CON CAMPO VARIAZIONE FORZA

F	2000	N	1000	N
ΔL	0,00971	mm	0,004854	mm
ε	0,000097		0,0000485	
ΔR	0,0466019	ohm	0,02330097	ohm
E ponte	0,0004854	V	0,000243	V
	0,485	mV	0,243	mV

Guadagno amplificatore per arrivare a 5V

A	10300	
Vamplif. Max.	5000	mV
Vamplif. Min	2500	mV

ESERCIZIO

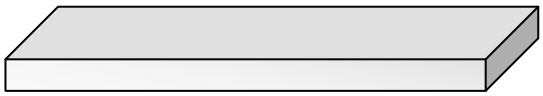
Valutare l'entità della forza nel caso in cui sia applicata perpendicolarmente all'estremita libera del pezzo (mensola).

ESTENSIMETRO CON AMPLIFICATORE DIFFERENZIALE

Si deve monitorare con Arduino il carico assiale applicato ad una trave sapendo che la sollecitazione può arrivare fino ad un massimo di 7725N. Si utilizza un semplice amplificatore differenziale (guadagno 200) per leggere la tensione del ponte.

PROVINO in ACCIAO sottoposto a trazione

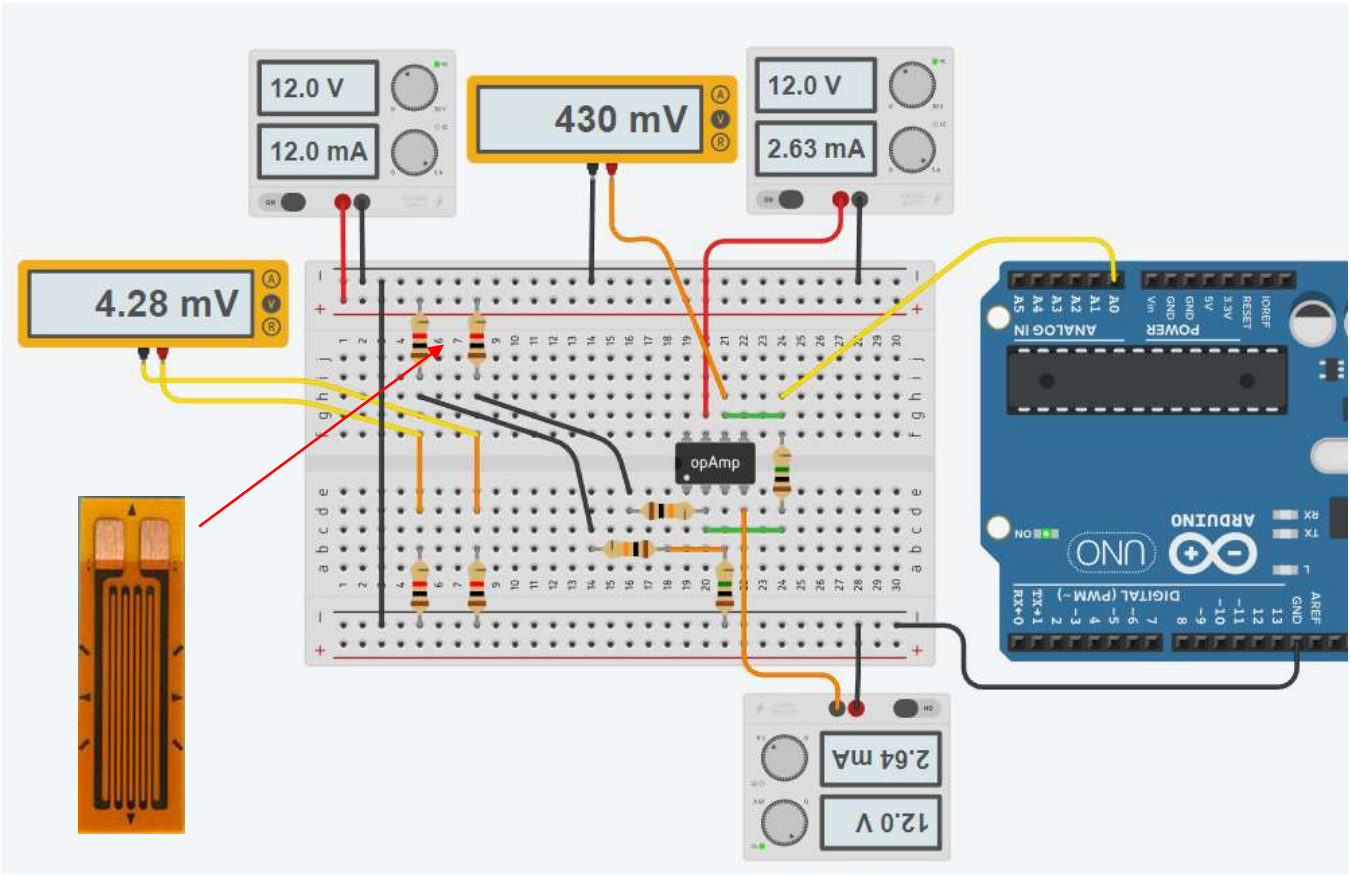
E	206000	N/mm2
Sez. 20x5mm	100	mm2
L	100	mm



ESTENSIMETRO in platino collegato a quarto di ponte

Vcc	12	V
Rg	1000	ohm
k	4	

Simulare il circuito su Thinkercad con l'estensimetro sollecitato a trazione F=7725N.



Notare che la tensione al ponte senza la presenza dell'amplificatore sarebbe di 4.5 mV come previsto dai calcoli.
La presenza dell'operazioneale e relative resistenza va a "disturbare" la tensione.
Questo effetto può essere attenuato aumentando i valori delle R dell'amplificatore fino ad un certo limite …

COMPITO

-Disegnare lo schema elettrico per effettuare la misura tramite Arduino e un *amplificatore differenziale*
-Simulare su Thinkercad un sistema di monitoraggio del provino che visualizzi su seriale e schermo LCD 16x2 la forza applicata e la deformazione della trave.
-Si deve attivare una lampada di emergenza (12V-500mA) tramite un relè, quando la sollecitazione supera i 7kN.

PROVINO ACCIAO sottoposto a trazione

E	206000	N/mm2
Sez. 20x5mm	100	mm2
L	100	mm

ESTENSIMETRO in Platino collegato a quarto di ponte

Vcc	12	V
Rg	1000	ohm
k	4	
E ponte	4,5	mV

Variazione di resistenza elettrica

ΔR	1,500	ohm

Deformazione relativa

				$\mu m/m$
ε	0,000375	m	375	$=\mu\varepsilon$

Allungamento

ΔL	0,000038	m	0,0375	mm

Forza di trazione rilevata

F	7725	N

TENSIONE AMPLIFICATA CON FORZA NOTA

F	7000	N
ΔL	0,03398	mm
ε	0,000340	
ΔR	1,3592233	ohm
E ponte	0,0040777	V
	4,078	mV

Tensione amplificata di 200

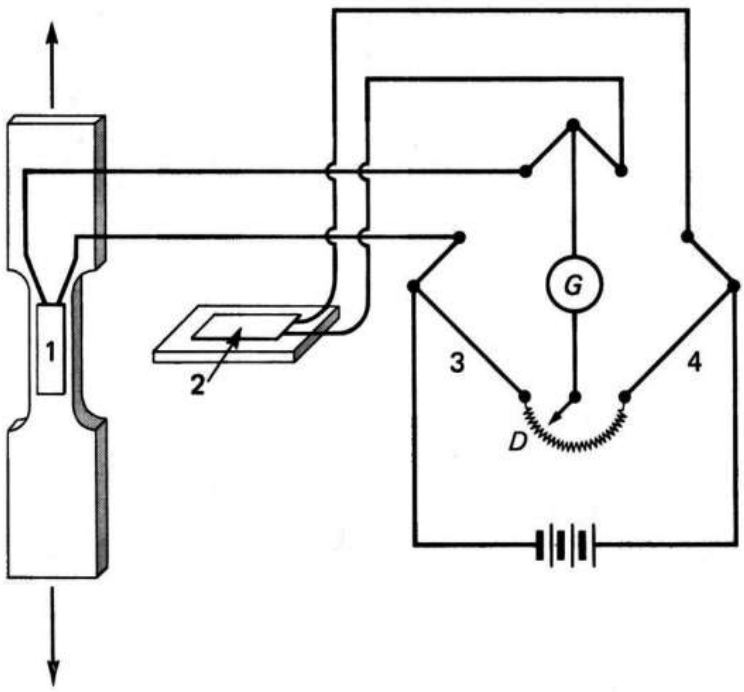
A	200	
Vamplif. Max.	816	mV

MISURA DELLA FORZA E DELLA DEFORMAZIONE IN UNA PROVA DI TRAZIONE

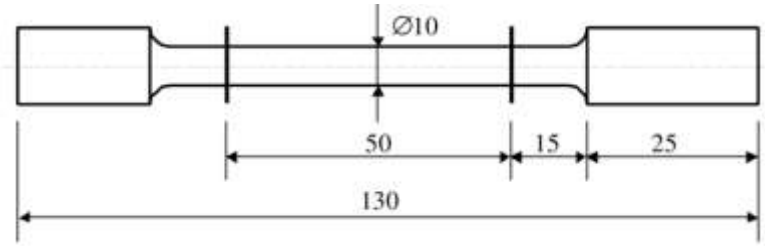
Per la misura della deformazione a trazione di un provino si può fare riferimento al collegamento riportato nella figura, dove le resistenze di completamento del ponte (3) e (4) sono state inglobate nel potenziometro di azzeramento D.

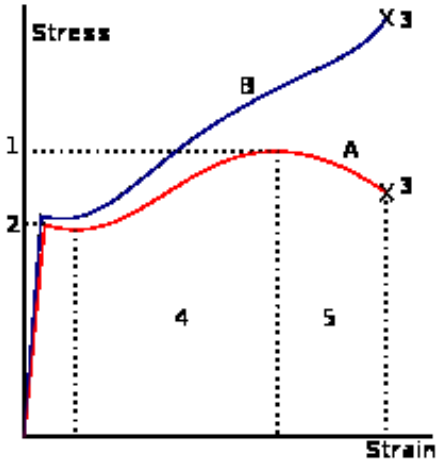
Si ipotizzi di effettuare la misura della deformazione di trazione del provino a temperatura standard (25°C), mediante un solo estensimetro (1), incollato con la griglia disposta longitudinalmente con l'asse del provino.

L'estensimetro (2) non soggetto a deformazione viene collegato al ramo opposto del ponte nelle vicinanze del sesnore (1) in modo da compensare eventuali variaizoni di resistenza dovute alla variazione delle temperatura ambiene (la T fa varlare la R del sensore!).



Valutare la tensione in presenza di un allungamento di 1/100 di mm del tratto utile del provino in Al di figura con un estensimetro in Platino da 350 ohm.



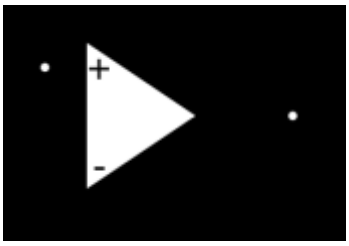


COMPITO

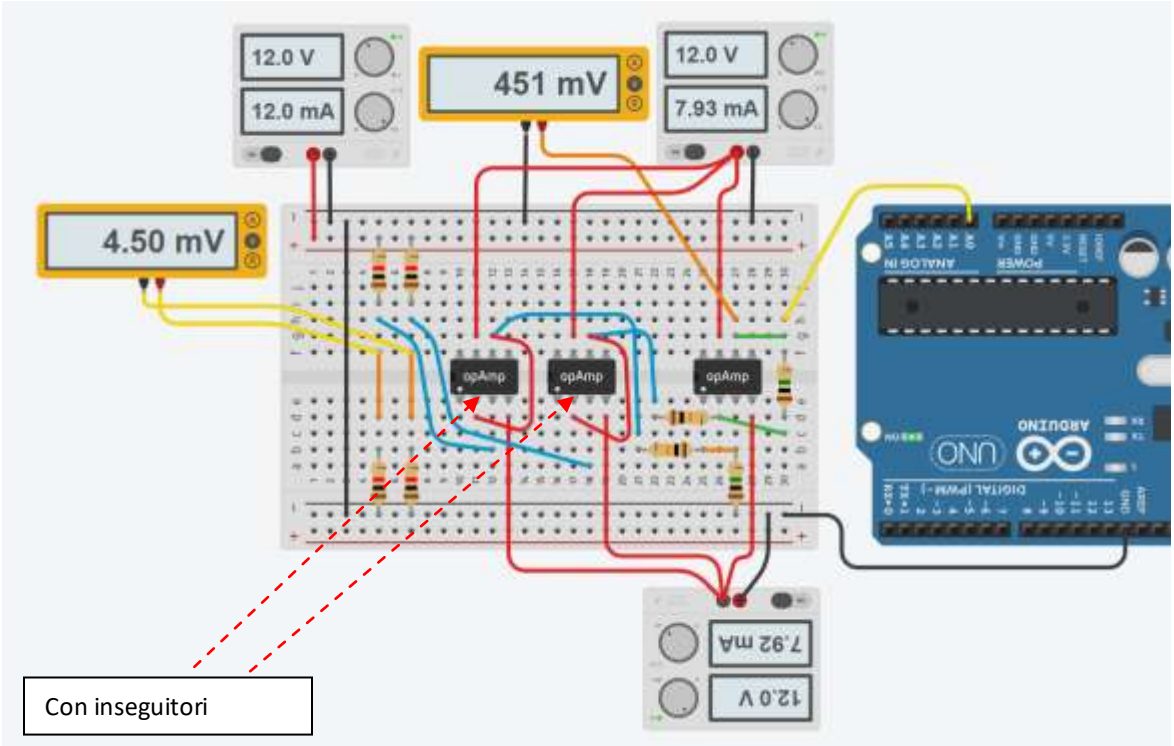
-Disegnare lo schema elettrico per effettuare la misura tramite Arduino e un *amplificatore differenziale da strumentazione*.
-Calcolare le resistenze dell'amplificatore differenziale in modo da amplifcare la tensione di 200 volte.
-Simulare su Thinkercad un sistema di monitoraggio del provino che visualizzi su seriale e schermo LCD 16x2 la forza applicata e la deformazione del provino in Al assegnato.
-Si deve attivare una lampada di emergenza (12V-500mA) tramite un transistor TIP120, quando la sollecitazione raggiunge quella di snervamento del materiale.

CIRCUITO CON AMPLIFICATORE DIFFERENZIALE DA STRUMENTAZIONE

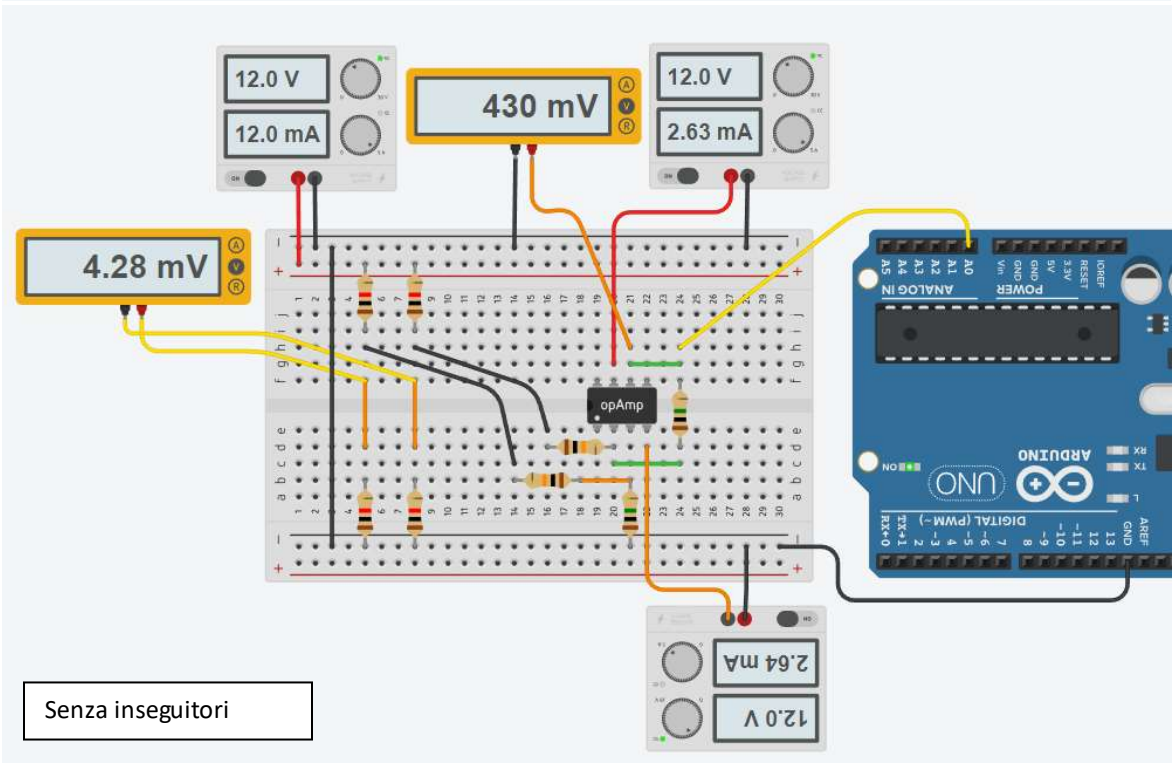
E' previsto l'utilizzo di due amplificatori in configurazione di "inseguitore" con l'obiettivo di ottenere una resistenza in ingresso all'amplificatore differenziale molto alta in modo da non influenzare la tensione letta dal ponte.



Tipicamente, un amplificatore separatore viene impiegato nel trasferimento di una tensione da un primo circuito, ad elevato livello d'impedenza, ad un secondo circuito, a livello d'impedenza inferiore.
L'amplificatore separatore interposto impedisce che il secondo circuito sovraccarichi il primo circuito e ne alteri il suo funzionamento.
Se la tensione viene trasferita inalterata, l'amplificatore separatore è un amplificatore a guadagno unitario detto "inseguitore di tensione".



Con inseguitori



Senza inseguitori

CELLE DI CARICO

Una cella di carico è un componente elettronico (trasduttore) impiegato per misurare una forza applicata su un oggetto (in genere un componente meccanico) tramite la misura di un segnale elettrico che varia a causa della deformazione che tale forza produce sul componente.

L'applicazione più comune è nei sistemi di pesatura elettronici e nella misura di sforzi meccanici di compressione e trazione.

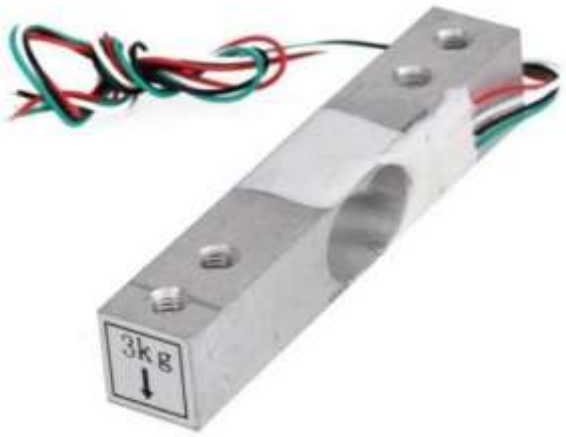
Questo componente è generalmente costituito da un corpo metallico (Acciaio inox martensitico o Alluminio).

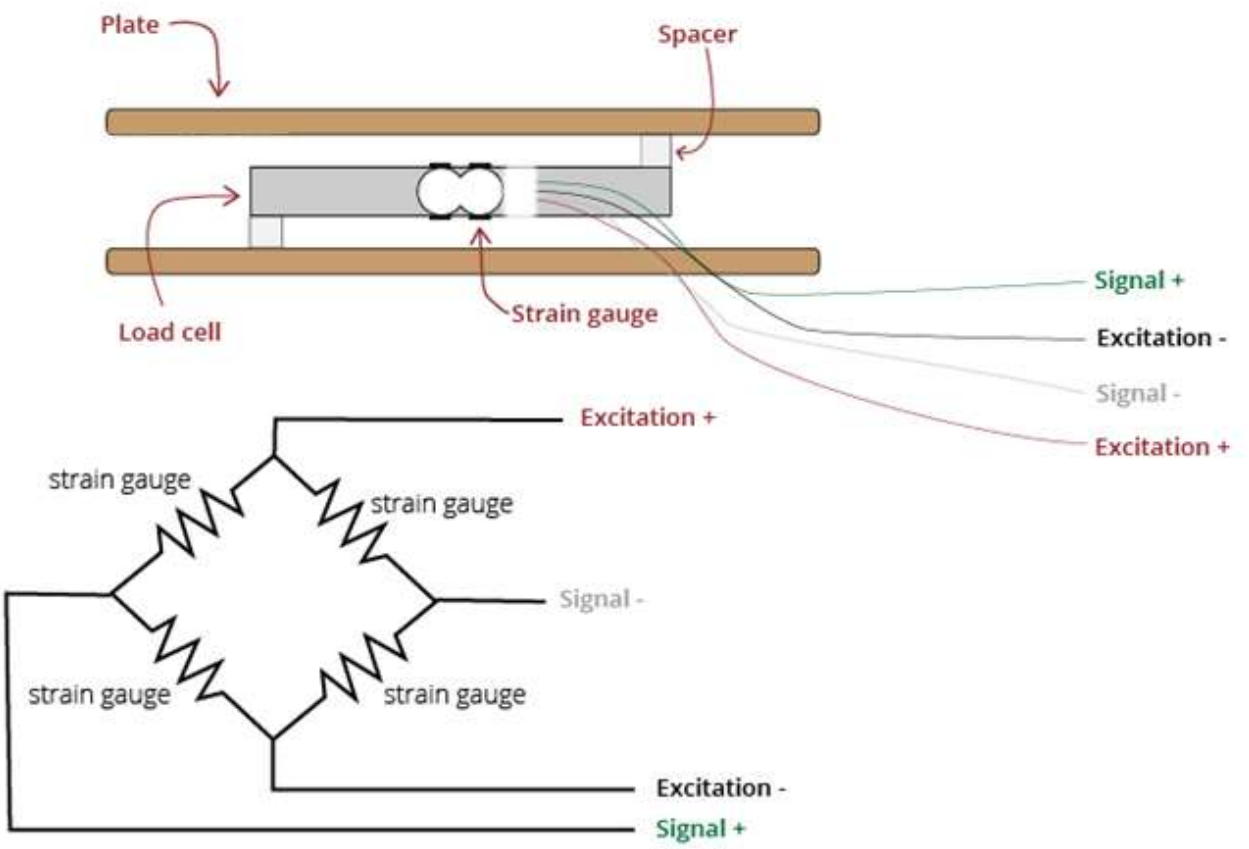
Al corpo della cella di carico vengono applicati uno o più estensimetri che rilevano la deformazione meccanica di compressione o trazione subita dal materiale tramite la variazione di resistenza elettrica causata dalla deformazione stessa.

Per aumentare la sensibilità dello strumento e migliorare così la qualità della misura la scelta più comune è quella di usare quattro estensimetri collegati tra di loro in una configurazione a ponte di Wheatstone, con i due estensimetri adiacenti posti a 90° l'uno rispetto all'altro; questa configurazione permette di aumentare la tensione in uscita dal ponte di circa 2,6 volte (per celle di carico in acciaio) rispetto alla tensione che restituirebbe una configurazione a quarto di ponte, inoltre permette di compensare l'effetto della temperatura che eventualmente comporterebbe errori.

Esistono comunque configurazioni più semplici che prevedono l'impiego di uno o due estensimetri. Il segnale elettrico ottenuto (differenziale) è normalmente dell'ordine di pochi millivolt e richiede un'ulteriore amplificazione ottenibile con un amplificatore da strumentazione prima di essere utilizzato.

Il segnale viene poi eventualmente elaborato mediante un algoritmo per calcolare la forza applicata al trasduttore.



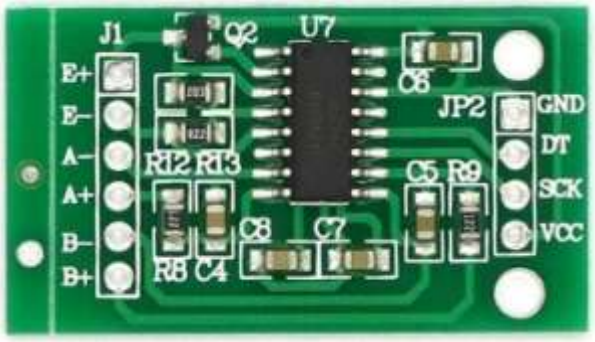


I fili provenienti dalla cella di carico hanno solitamente i seguenti colori:

- Rosso: VCC (E+)
- Nero: GND (E-)
- Bianco: Uscita – (LA-)
- Verde: Uscita + (LA+)

SCHEDA ELETTRONICA PER CELLA DI CARICO - HX711

Le celle di carico devone essere accoppiate a degli opportuni amplificatore
Il chip HX711 è composto da un amplificatore a guadagno variabile e da un convertitore analogico-digitale di precisione a 24 bit.
Presenta un'alta velocità di risposta e un'alta immunità ai disturbi.

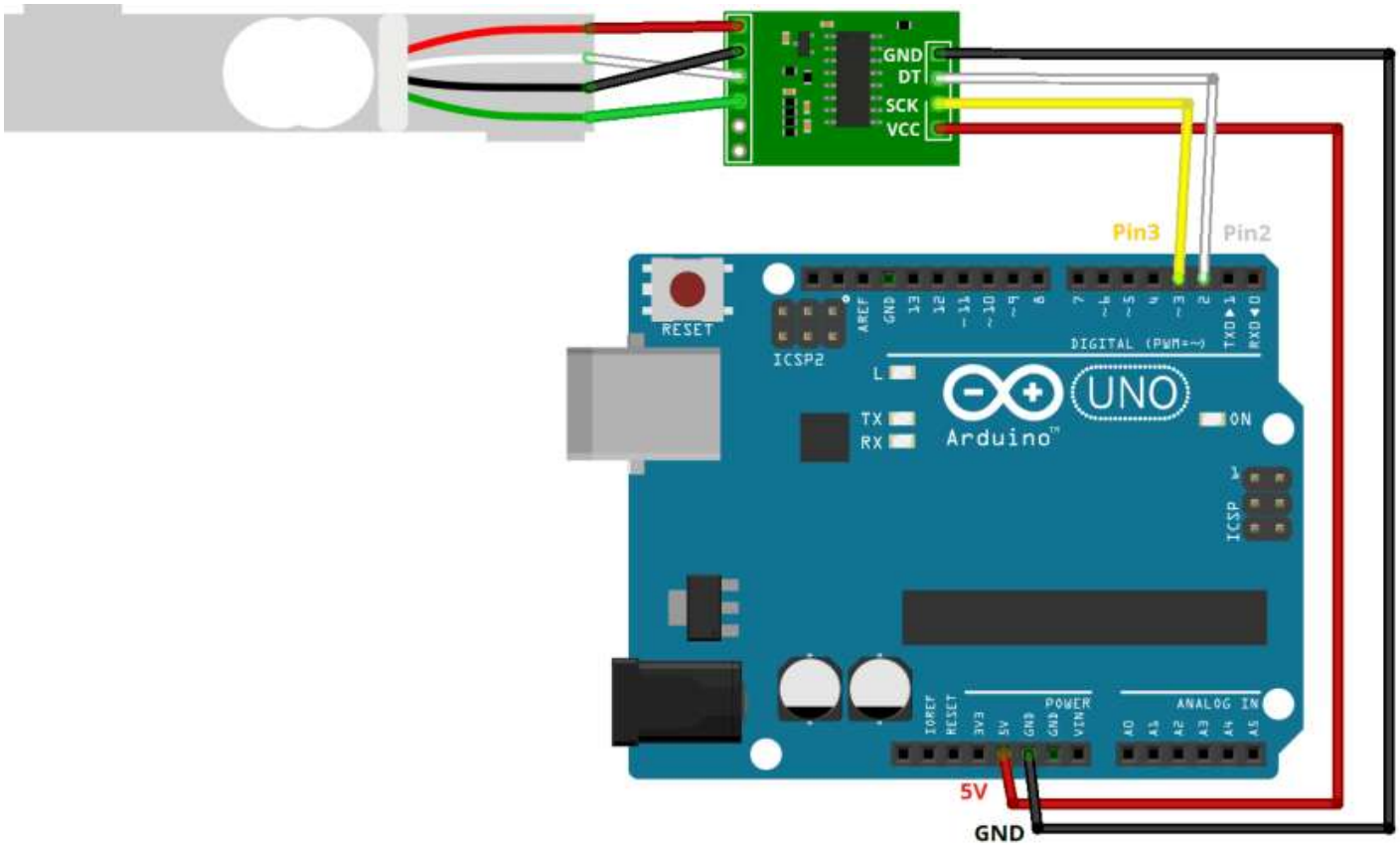


SCHEMA DI COLLEGAMENTO AD ARDUINO

L'amplificatore HX711 comunica tramite un'interfaccia a due fili.

Puo essere collegato a un qualsiasi pin digitale della scheda Arduino.

Cella di carico	HX711	HX711	Arduino
Rosso(Mi+)	E+	GND	GND
Nero(E-)	E-	DT	pin 2
Bianco(UN-)	UN-	SCK	pin 3
Verde(LA+)	A+	VCC	5V



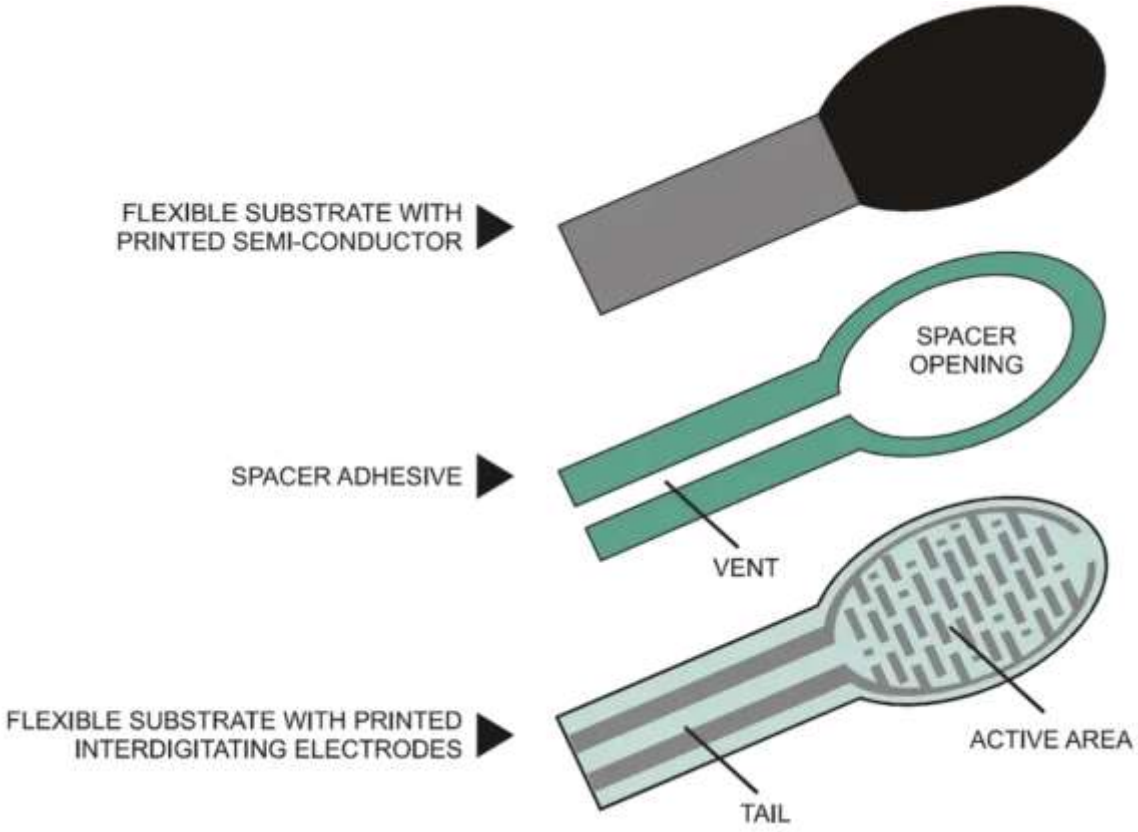
Per poter essere utilizzate correttamente le celle di carico necessitano di una accurata taratura inziale effettuata con una sollecitazione di intensità nota.

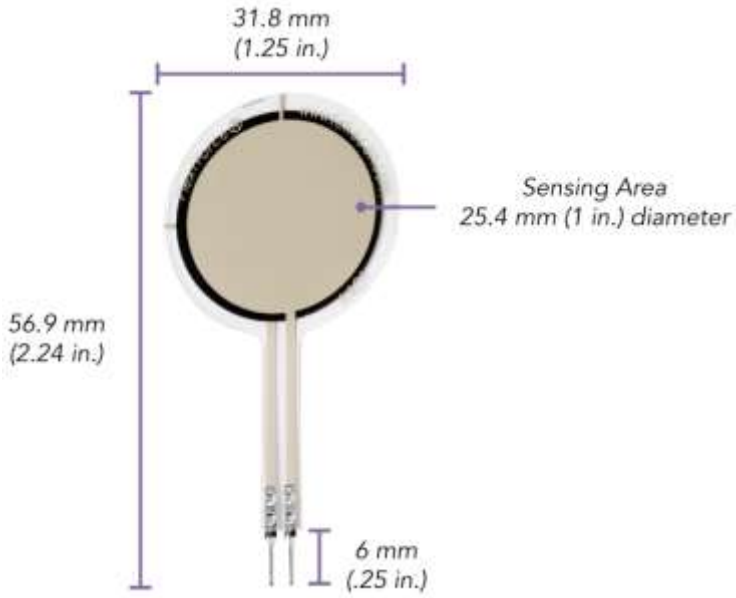
SENSORE DI FORZA (FSR FORCE SENSITIVE RESISTOR)

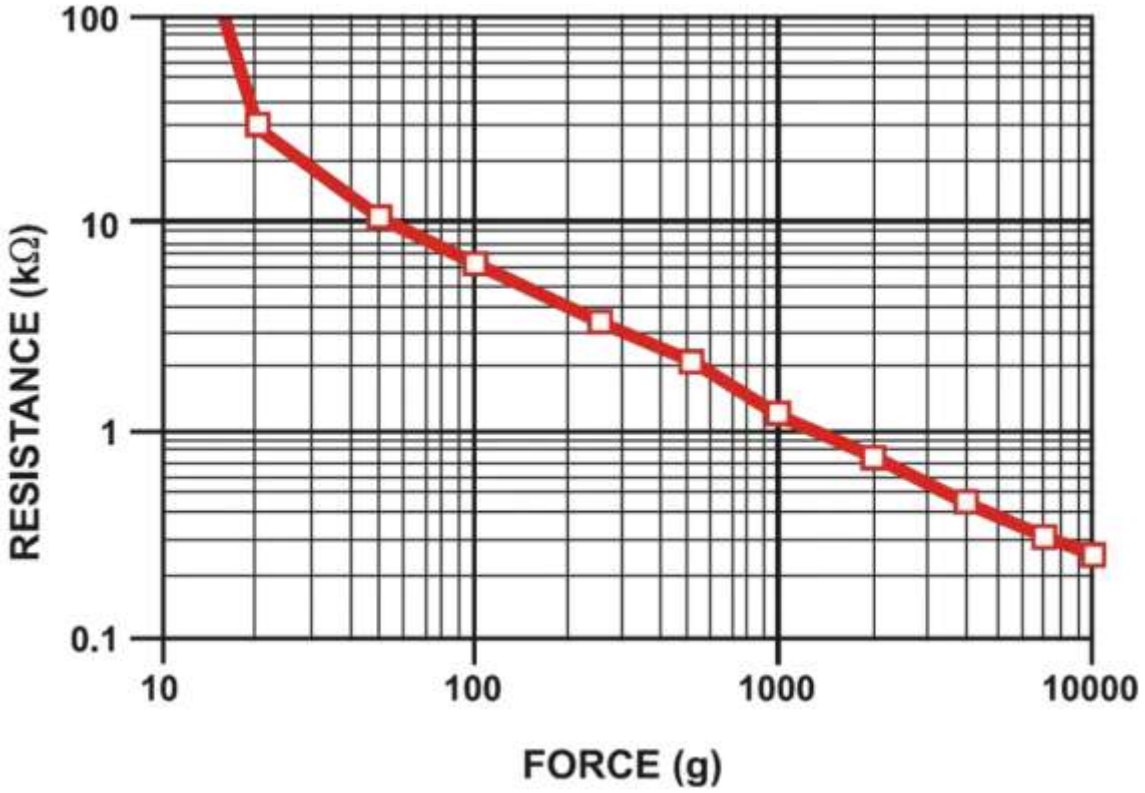
I resistori di rilevamento della forza (FSR) sono un dispositivo a film spesso polimerico (PTF) che mostrano una diminuzione della resistenza con un aumento della forza applicata alla superficie attiva.

La sensibilità alla forza è ottimizzata per l'uso nel controllo tattile umano di dispositivi elettronici.

Gli FSR non sono una cella di carico o un estensimetro, sebbene abbiano proprietà simili. Gli FSR non sono adatti per misurazioni di precisione.



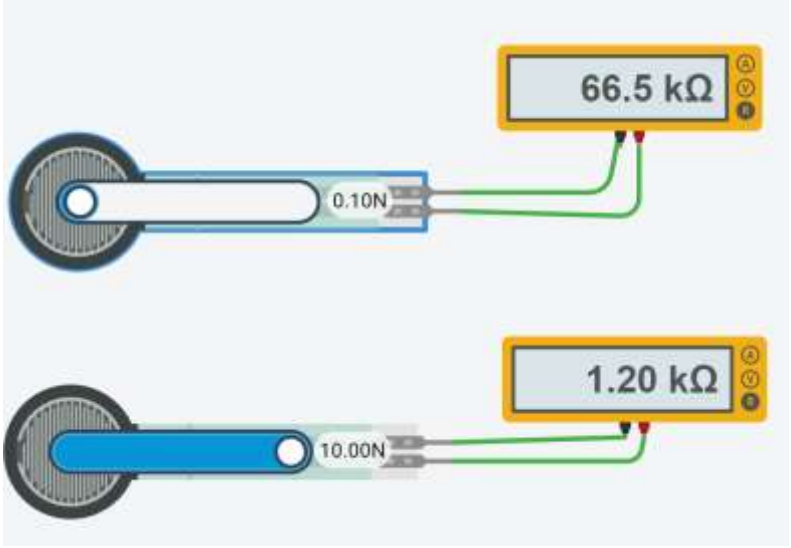




Facendo riferimento alla figura, all'estremità della forza inferiore della caratteristica forza-resistenza, è evidente una risposta simile a un interruttore. Questa soglia di attivazione, o "forza di rottura", fa oscillare la resistenza da più di 100 kΩ a circa 10 kΩ (l'inizio dell'intervallo dinamico che segue una legge di potenza) è determinato dallo spessore e dalla flessibilità del substrato e del rivestimento, dalle dimensioni e dalla forma dell'attuatore e dallo spessore dell'adesivo distanziatore (lo spazio tra gli elementi conduttivi affacciati). La forza di rottura aumenta con l'aumentare della rigidità del substrato e del rivestimento, delle dimensioni dell'attuatore e dello spessore dell'adesivo distanziatore. Eliminare l'adesivo o tenerlo ben lontano dall'area in cui viene applicata la forza, come il centro di un grande dispositivo FSR, gli darà un riposo inferiore resistenza (ad esempio resistenza stand-off).

ESERCITAZIONE ARDUINO

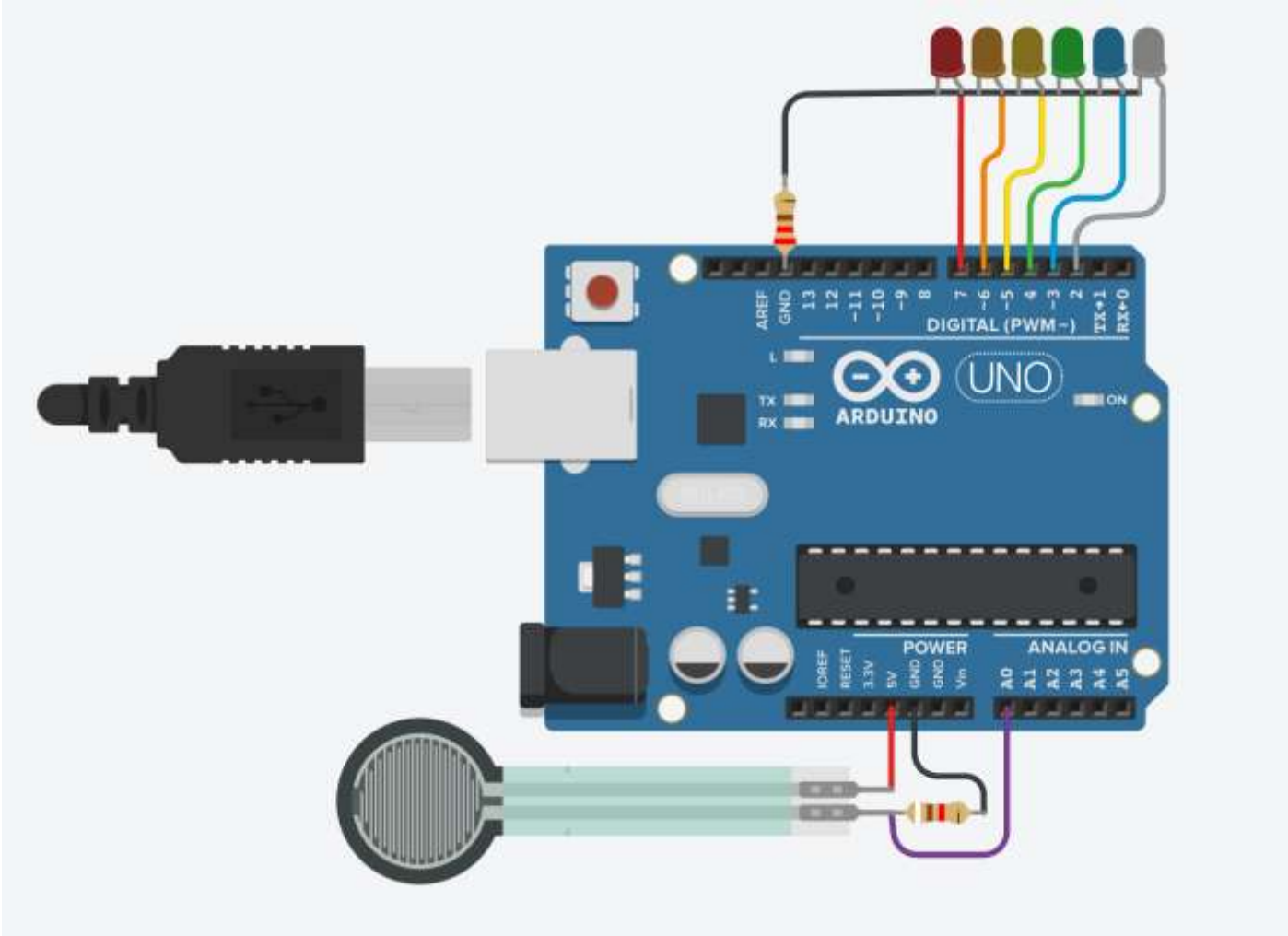
Rilevare il campo di variazione della resistenza del sensore in Thinkercad.



NB:
Le misure della resistenza del sensore vanno fatte scollegate dall'alimentazione!

ESERCIZIO CON SENSORE DI FORZA

Accendere una striscia di led in modo proporzionale alle forza rilevata dal sensore.



CODICE

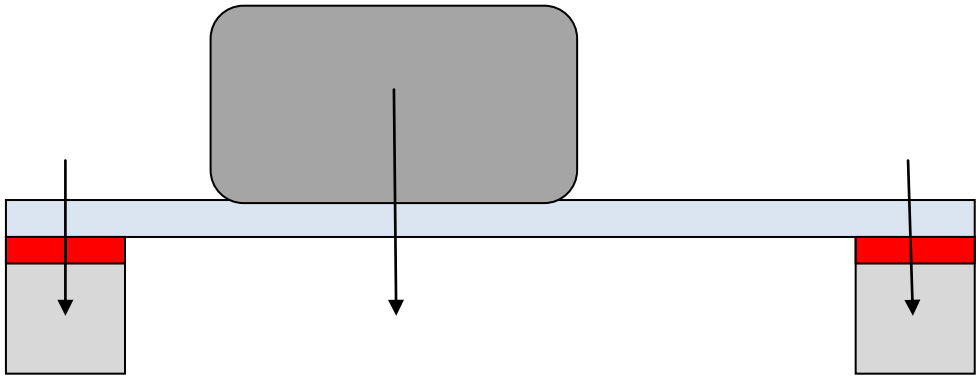
```
#define fsrpin A0
#define led1 2
#define led2 3
#define led3 4
#define led4 5
#define led5 6
#define led6 7

int fsrreading;

void setup() {
  Serial.begin(9600);
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(led3, OUTPUT);
  pinMode(led4, OUTPUT);
  pinMode(led5, OUTPUT);
  pinMode(led6, OUTPUT);
}
```

```
void loop() {

  fsrreading = analogRead(fsrpin);

  Serial.println(fsrreading);

  if (fsrreading > 200) {
    digitalWrite(led1, HIGH);
  }
  else digitalWrite(led1, LOW);
  if (fsrreading > 450) {
    digitalWrite(led2, HIGH);
  }
  else digitalWrite(led2, LOW);
  if (fsrreading > 550) {
    digitalWrite(led3, HIGH);
  }
  else digitalWrite(led3, LOW);
  if (fsrreading > 650) {
    digitalWrite(led4, HIGH);
  }
  else digitalWrite(led4, LOW);
  if (fsrreading > 800) {
    digitalWrite(led5, HIGH);
  }
  else digitalWrite(led5, LOW);
  if (fsrreading > 900) {
    digitalWrite(led6, HIGH);
  }
  else digitalWrite(led6, LOW);
}
```

COMPITO

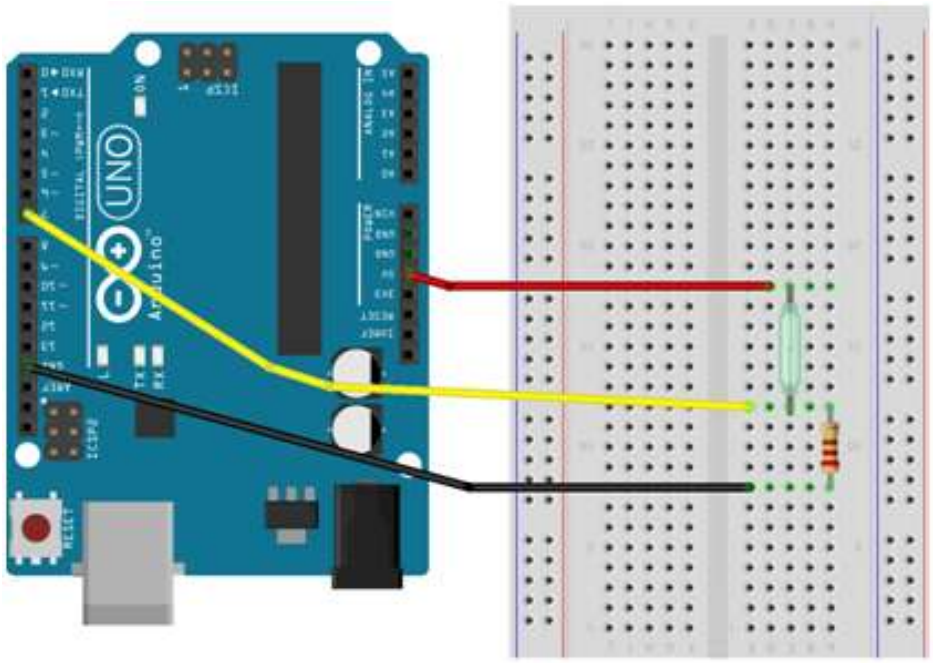
1- Utilizzare il sensore di forza per avviare un motore DC 5V quando la forza applicata supera i 5N.

2- Utilizzare due sensori di forza per verificare se un oggetto è posizionato al centro di un appoggio (la forza rilevata dai sensori sarà uguale) e usare due led di colore diverso per indicare se è sbilanciato a destra o sinistra.



SENSORI REED

Nell'esempio sottostante, con alimentazione a 5V e resistenza 10K, la corrente che circola nel sensore vale
I = 5 / (10K + R sensore) = 0,5 mA (la resistenza del sensore è di pochi ohm …)



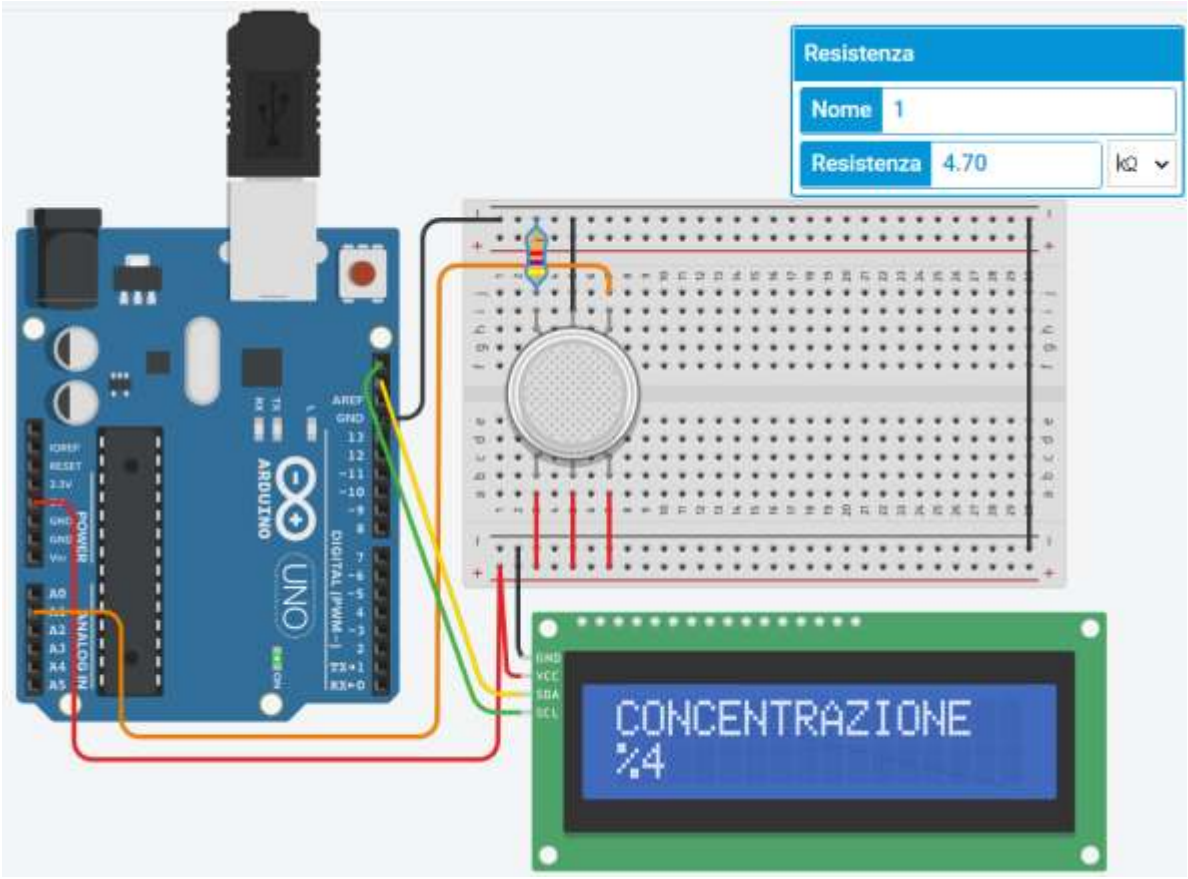
CODICE

Se un magnete si avvicina all'interruttore reed questo si chiude e Arduino attiverà il led interno pin 13.

```
#define reedPin 8
#define led 13
int reedState = 0;

void setup()
{
        pinMode(led, OUTPUT);
        pinMode(reedPin, INPUT);
        Serial.begin(9600);
}

void loop() {
        reedState = digitalRead(reedPin);
        if (reedState == HIGH){
                Serial.println("reed ON");
                digitalWrite(led, HIGH);
        }
        else{
                Serial.println("reed OFF");
                digitalWrite(led, LOW);
        }
        delay(500);
}
```

SENSORE GAS

Il sensore di gas fornito in Tinkercad ha 6 terminali, ovvero A1, H1, A2, B1, H2 e B2. Collegare i terminali A1, H1 e A2 all'alimentazione. Successivamente collegare il terminale B1 all'alimentazione ed i terminali H2 e B2 a massa. Per aumentare la precisione e la sensibilità del sensore di gas, colleghiamo una resistenza tra il terminale B2 e la massa.



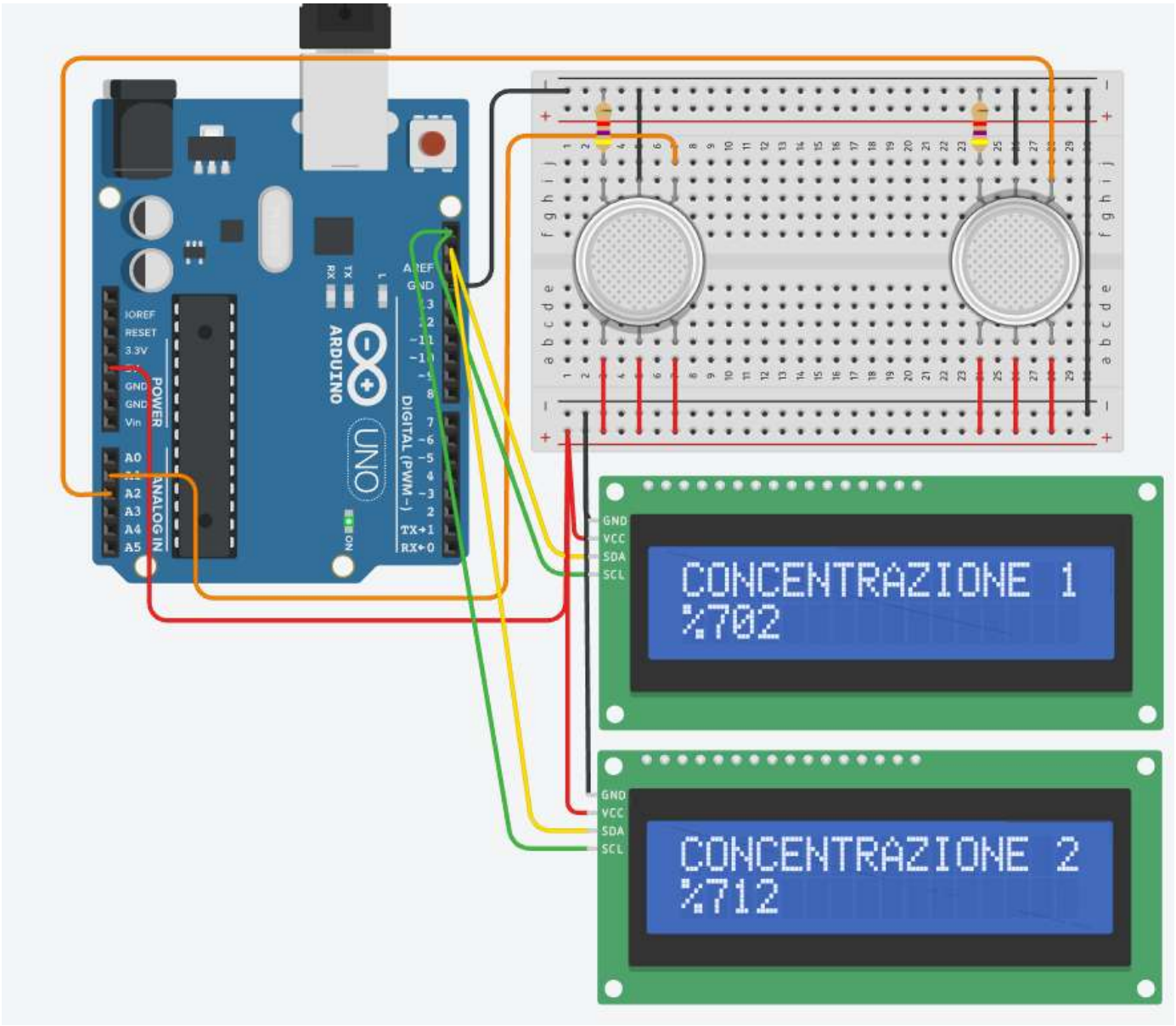
CODICE

```
#include <Adafruit_LiquidCrystal.h>
Adafruit_LiquidCrystal lcd_1(0);  // 0=1° indirizzo I2C 32

void setup()
{
  //lcd_1.clear();
  lcd_1.begin(16, 2);
  lcd_1.print("CONCENTRAZIONE");
}

void loop(){
  int valGas = analogRead(A1);
  valGas = map(valGas, 300, 750, 0, 100);
  lcd_1.setCursor(0, 1);
  lcd_1.print("%");
  lcd_1.print(valGas);
  lcd_1.print(" ");
  lcd_1.setBacklight(1);
  delay(250);
}
```

RILEVAZIONE VALORE MEDIO CONCENTRAZIONE GAS

Misurare la concentrazione di gas in un locale tramite due sensori e visualizzare il valore su due display I2C.

Valutare la media fornita dai due sensori e visualizzarla sulla seriale.

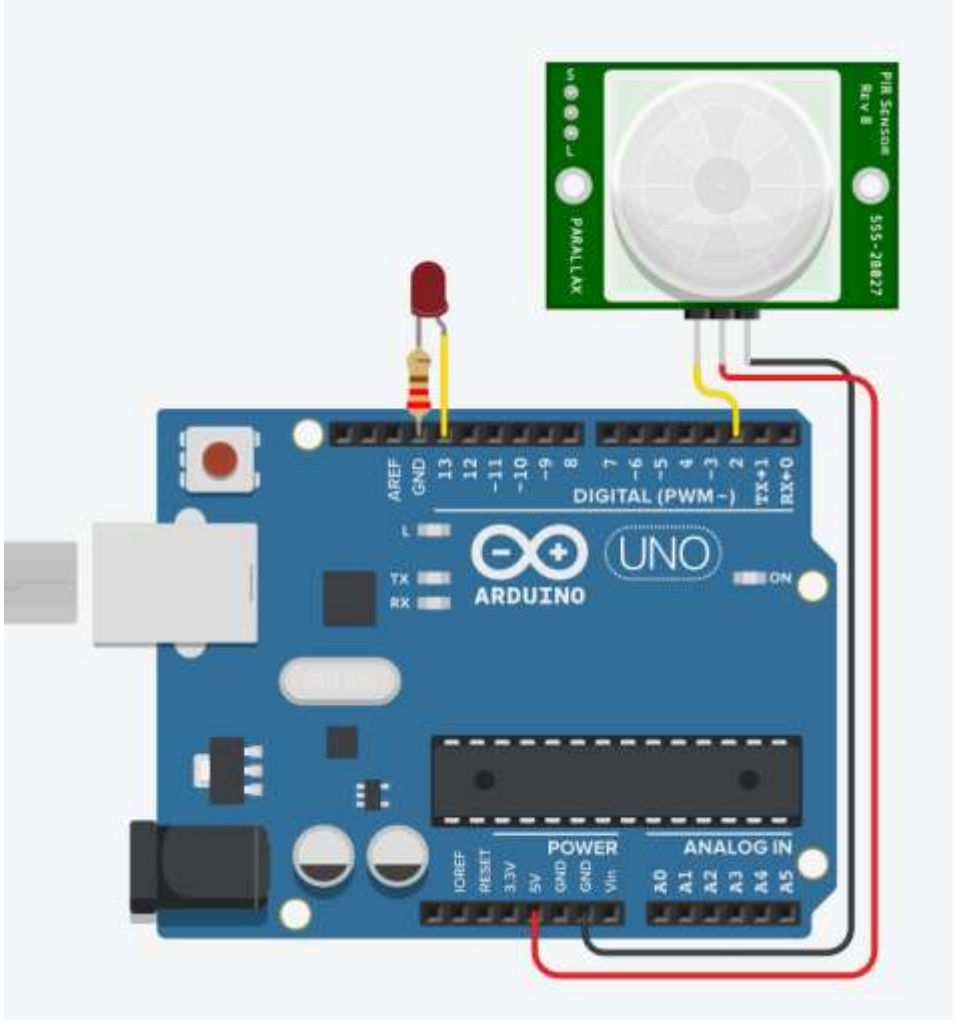


CODICE

```
#include <Adafruit_LiquidCrystal.h>
Adafruit_LiquidCrystal lcd_1(0);  // 0=1° indirizzo I2C 32
Adafruit_LiquidCrystal lcd_2(1);  // 1=2° indirizzo I2C 33


void setup()
{
  pinMode(A1, INPUT);
  pinMode(A2, INPUT);

 Serial.begin(9600);

  lcd_1.begin(16, 2);
  lcd_1.print("CONCENTRAZIONE 1");
  lcd_2.begin(16, 2);
  lcd_2.print("CONCENTRAZIONE 2");
}

void loop(){
  int valGas1 = analogRead(A1);
  valGas1 = map(valGas1, 300, 750, 0, 100);
  //lcd_1.clear();
  lcd_1.setCursor(0, 1);
  lcd_1.print("%");
  lcd_1.print(valGas);
  lcd_1.print(" ");
  lcd_1.setBacklight(1);

  int valGas2 = analogRead(A2);
  valGas2 = map(valGas1, 300, 750, 0, 100);
  //lcd_1.clear();
  lcd_2.setCursor(0, 1);
  lcd_2.print("%");
  lcd_2.print(valGas2);
  lcd_2.print(" ");
  lcd_2.setBacklight(1);

  float media= (valGas1 + valGas2) / 2.0;
  Serial.println(media);

  delay(250);
}
```

SENSORE DI MOVIMENTO (PIR)

Un sensore ad infrarossi passivo (PIR sensor, acronimo di Passive InfraRed) è un sensore elettronico che rileva la radiazione infrarossa (IR) irradiata dagli oggetti nel suo campo visivo.
Rileva repentine variazioni di temperatura che modificano lo stato che il PIR aveva "memorizzato come normale".



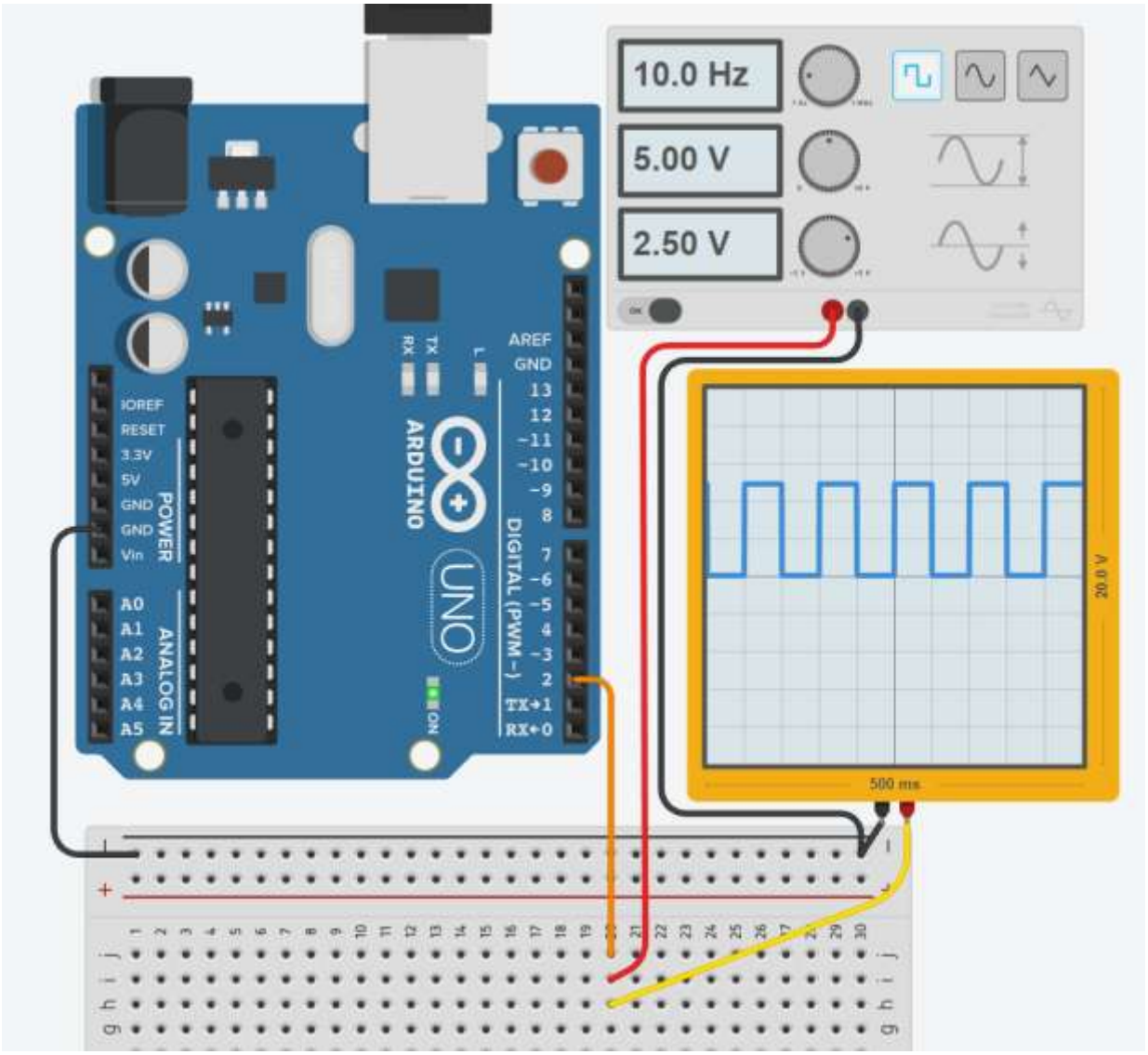
CODICE

```
int pirState = 0;

void setup()
{
  pinMode(2, INPUT);
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop()
{
  // controllo stato PIR
  pirState = digitalRead(2);
  if (pirState == HIGH) {
    digitalWrite(LED_BUILTIN, HIGH);
  } else {
    digitalWrite(LED_BUILTIN, LOW);
  }
  delay(1); // Delay a little bit to improve simulation performance
}
```

INTERRUPT E CONTEGGIO IMPULSI DA UN TRASDUTTORE

Un interrupt (*interruzione*) è un evento che viene generato in presenza di una variazione di livello (da 0→5V o 5→0V) su un particolare pin di Arduino (pin 2 e 3 per la scheda Arduino UNO).
Questo evento viene gestito direttamente dal microcontrollore ed è controllabile via software tramite delle apposite istruzioni.
Quando viene generata una interruzione è possibile eseguire del codice in modo automatico che interrompe momentaneamente il normale flusso di codice all'interno del blocco loop().

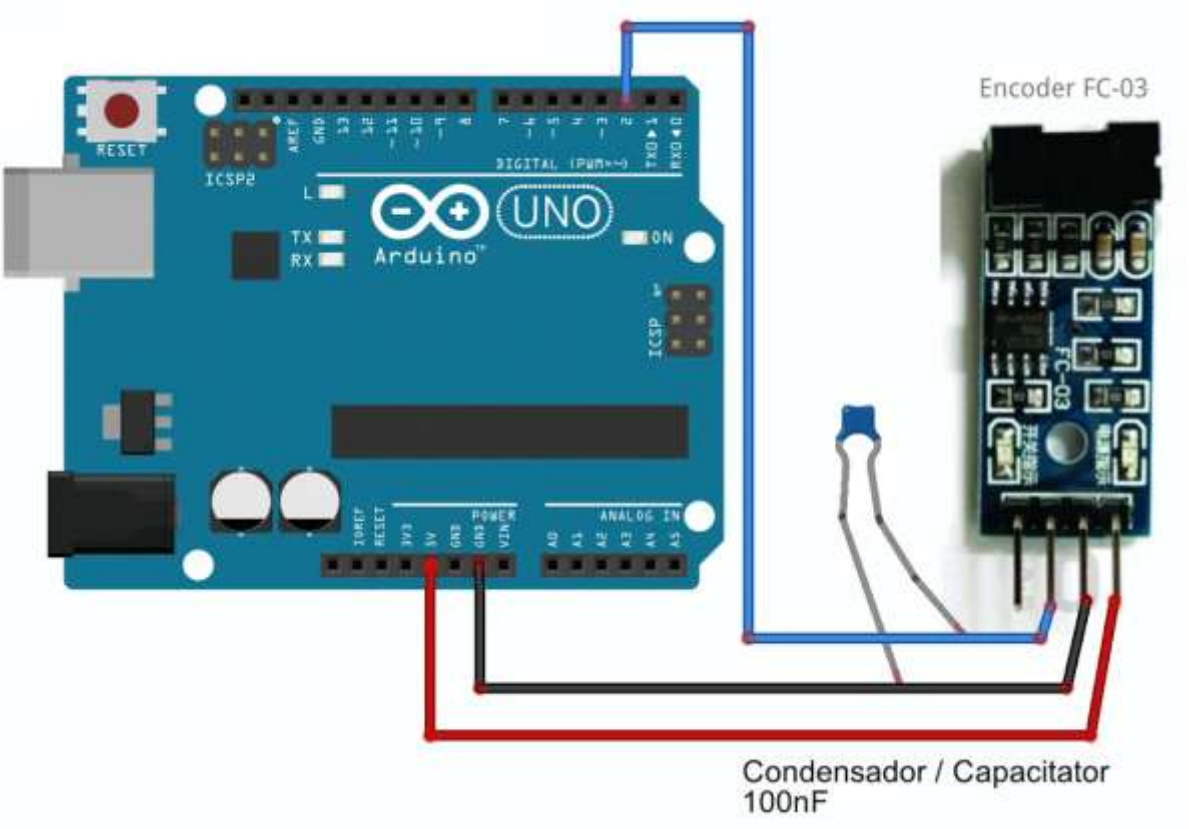
Nello schema di figura il generatore di funzioni d'onda viene utilizzato per simulare un trasduttore (es. encoder ottico) che genera un treno di impulsi con frequenza proporzionale al valore della grandezza fisica misurata.



Variabili di tipo "volatile" ***(salvata nella memoria RAM di Arduino)***

Una variabile deve essere dichiarata volatile ogni volta che il suo valore può essere modificato da qualcosa al di fuori del controllo della sezione di codice in cui appare come ad esempio in un thread (processo parallelo) in esecuzione contemporaneamente al codice principale.
In Arduino, l'unico posto in cui è probabile che ciò avvenga è nelle sezioni di codice associate agli interrupt, chiamate routine di servizio di interrupt.

volatile int contatore = 0;



Utilizzo shield Arduino FC-03 per contare gli impulsi rilevati da un encoder

CODICE

```
volatile int contatore = 0;

void interrupt0()
{
  contatore++;
}

void setup() {
  Serial.begin(9600);
  // uso il pin2 per l'interrupt (solo il 2 o il 3 di Arduino sono abilitati agli interrupt)
  attachInterrupt(digitalPinToInterrupt(2),interrupt0,RISING);
}

void loop() {
  delay(1000);
  Serial.print(contatore);
  Serial.println(" impulsi");
  contatore = 0;
}
```

ENCODER

L'encoder è un dispositivo elettromeccanico che converte la posizione angolare meccanica del suo asse rotante in posizione angolare elettrica sotto forma di segnale elettrico numerico digitale e/o analogico.

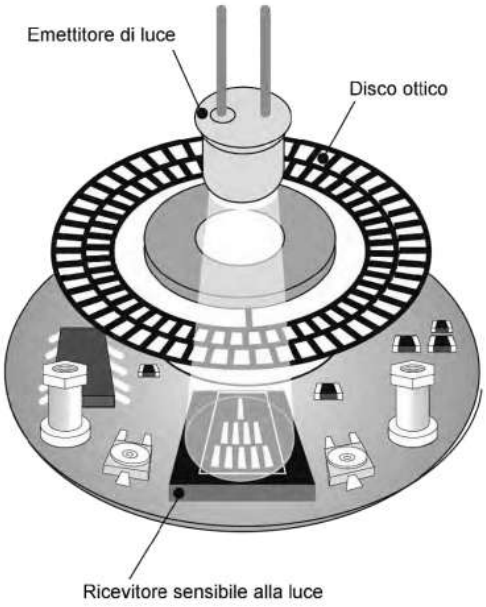
Collegato ad opportuni circuiti elettronici e con appropriate connessioni meccaniche, l'encoder è in grado di misurare spostamenti angolari, movimenti rettilinei e circolari nonché velocità di rotazione e accelerazioni.

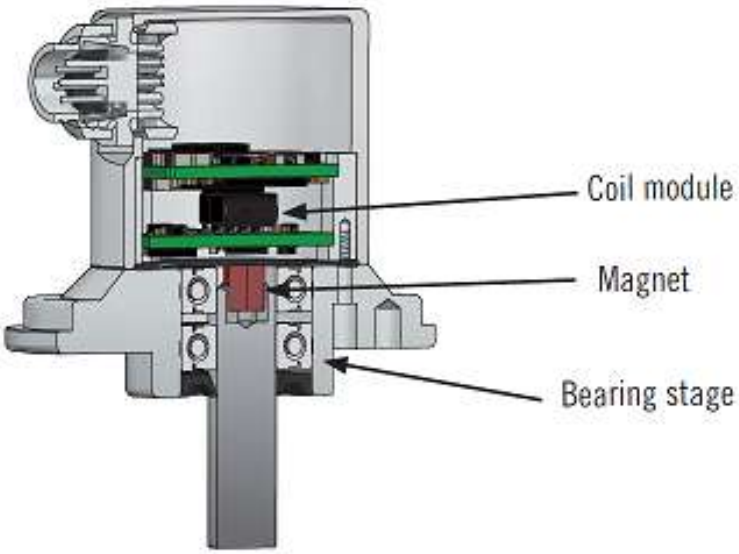
Esistono varie tecniche per il rilevamento del movimento angolare: capacitiva, magnetica, induttiva, potenziometrica e fotoelettrica.

Gli encoder si possono classificare nelle seguenti categorie:

- ottici
- a variazione di campo magnetico/elettrico.

Gli encoder vengono principalmente impiegati nei seguenti settori applicativi: controllo dei processi industriali, robot industriali, macchine utensili, strumenti di misura, confezionamento, plotter, laminatoi e macchine per il taglio delle lamiere, bilance e bilici, antenne, telescopi, impianti ecologici, macchine da stampa e da imballaggio, macchine tessili e conciarie, gru, carri ponte, presse, macchine per la lavorazione del legno, della carta, del marmo , del cemento, del vetro ecc.



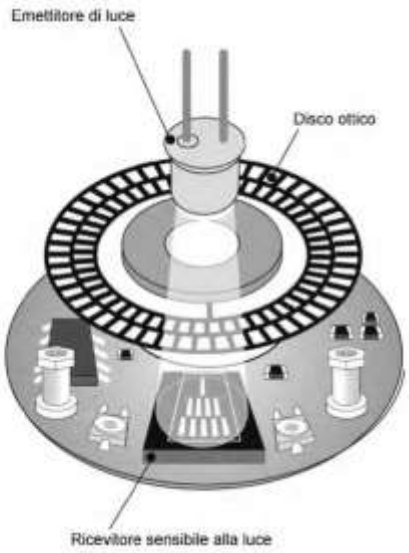


ENCODER OTTICI

L'encoder è uno strumento per la misura di una posizione angolare basato sul **principio fotoelettrico**

Il principio base di funzionamento è il seguente: davanti a una sorgente luminosa è posto un disco che presenta alternativamente aree opache e trasparenti. Un fotorilevatore rileva l'intensità luminosa che attraversa il disco e fornisce quindi un'informazione indiretta sulla posizione angolare

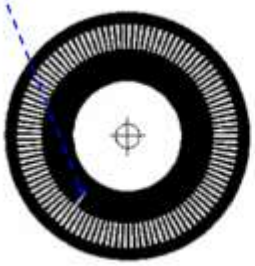
L'encoder può essere **incrementale** o **assoluto**



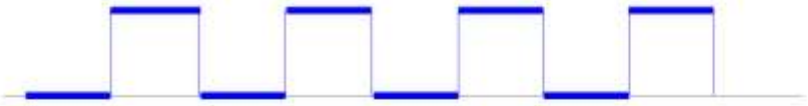
ENCODER INCREMENTALE

Nella sua versione più semplice l'encoder incrementale è costituito da una sola traccia con zone alternativamente trasparenti o opache

Tacca per la definizione di uno zero meccanico assoluto.

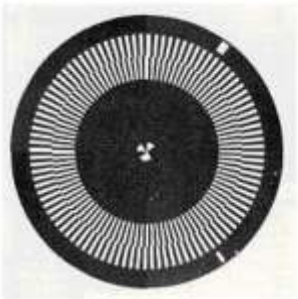


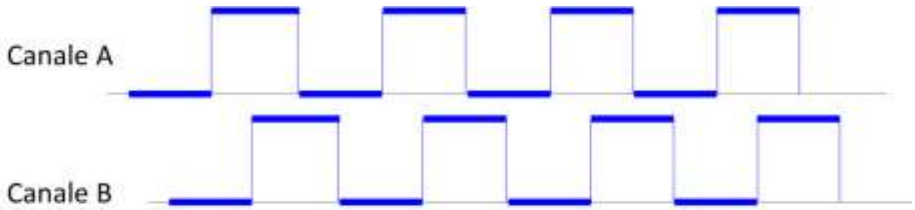
L'uscita del fotorivelatore viene squadrata da un circuito elettronico e dà luogo a un treno di impulsi.



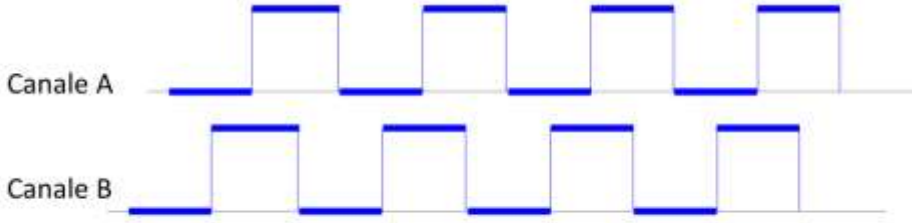
- Non si ha la posizione angolare assoluta ma si possono solo contare gli impulsi dal momento dell'accensione
- Non si può rilevare il verso di rotazione

Per stimare anche il **verso di rotazione**, si aggiunge una **seconda traccia**, sfasata di 1/4 di passo.



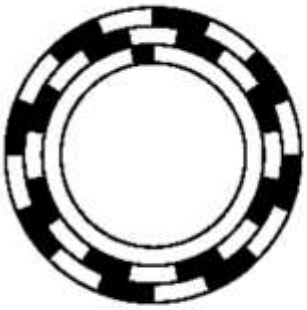


(A in anticipo su B: rotazione **oraria**)



(B in anticipo su A: rotazione **antioraria**)

ENCODER INCREMENTALE: RISOLUZIONE

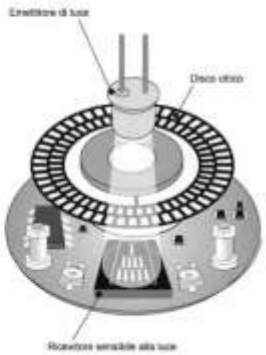


N: numero passi (numero di zone chiare/scure per giro).

Poiché i due segnali sono sfasati di 1/4 di passo:

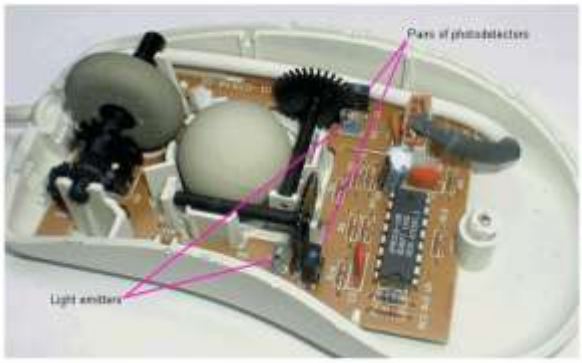
Risoluzione: $360^{\circ}/(4N)$

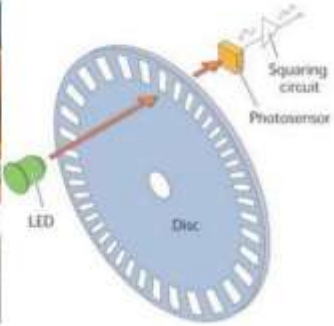
Con $N = 1000$ la risoluzione è di 0.090°



ENCODER INCREMENTALE: ESEMPIO D'USO

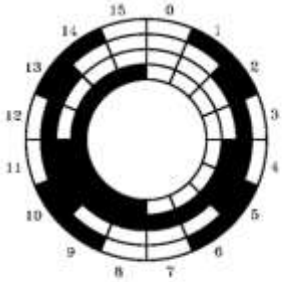
Encoder incrementali su un mouse (a pallina)





ENCODER ASSOLUTO

- L'**encoder assoluto** è un disco con aree trasparenti e opache, disposte su corone circolari concentriche
- La traccia più interna suddivide l'angolo giro in due, la seconda traccia suddivide ciascun angolo piatto in due angoli di 90° e così via



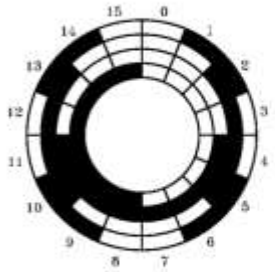
Con N corone circolari si ha quindi:

Risoluzione: $360^{\circ}/2^{N}$

Per le applicazioni robotiche sono richieste almeno 12 tracce (risoluzione di 360°/4096 = 0.088°).

Viene codificata la **posizione angolare assoluta**: all'accensione, lo strumento fornisce l'angolo assoluto

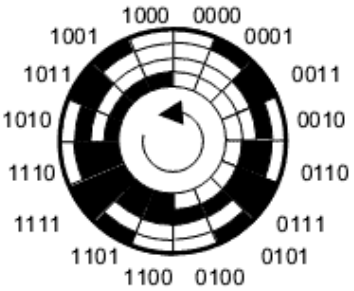
Per evitare ambiguità di lettura si utilizzano codici binari a variazione singola (**codice Gray**): da un settore a quello successivo cambia una sola cifra.



#	Codice	#	Codice
0	0000	8	1100
1	0001	9	1101
2	0011	10	1111
3	0010	11	1110
4	0110	12	1010
5	0111	13	1011
6	0101	14	1001
7	0100	15	1000

Encoder a 12 bit con codifica Gray

Numero decimale	Numero in binario puro	Numero in Gray
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

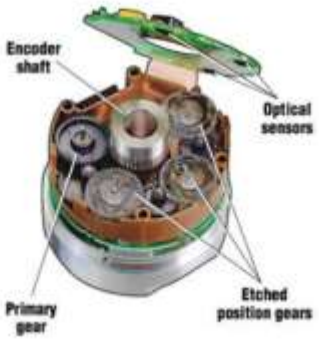


Gray code

ENCODER ASSOLUTO: SINGLE-TURN O MULTI-TURN

Gli encoder **single-turn** misurano solo la posizione sull'angolo giro: un encoder single-turn a 13 bit ha una risoluzione sul giro di $360°/2^{13} = 0.044°$

Gli encoder **multi-turn** misurano anche il numero delle rotazioni, mediante opportuni ingranaggi: un encoder multi-turn a 29 bit può avere 13 bit dedicati alla misura sul giro e gli altri 16 a misurare fino a $2^{16} = 65.536$ giri



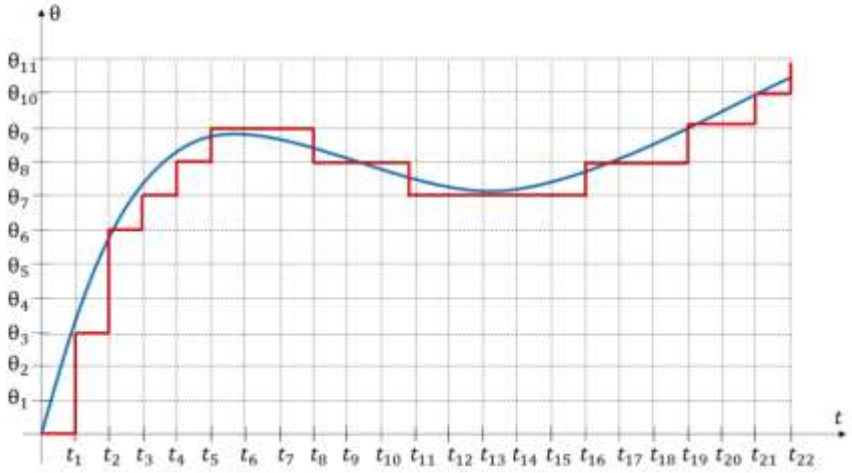
Encoder assoluto Sick

MISURA DI VELOCITÀ DAL SEGNALE ENCODER

Dagli impulsi prodotti da un encoder (incrementale o assoluto) si può ricavare una **misura di velocità**. Esistono due metodi:

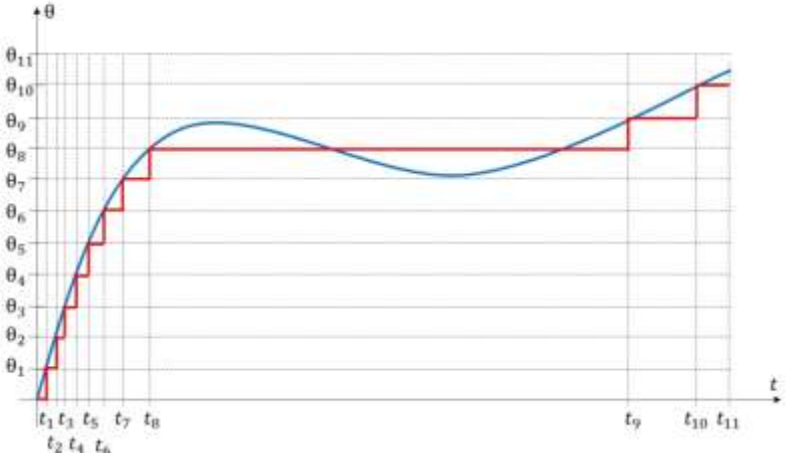
A tempo fisso: si campiona l'uscita dell'encoder a passo temporale fissato T_s

$$\omega(k) = \frac{\theta(k) - \theta(k-1)}{T_s}$$



A spazio fisso: si misura il tempo necessario per cambiare l'uscita dell'encoder di un bit (variazione $\Delta\theta$)

$$\omega(k) = \frac{\Delta\theta}{t_k - t_{k-1}}$$

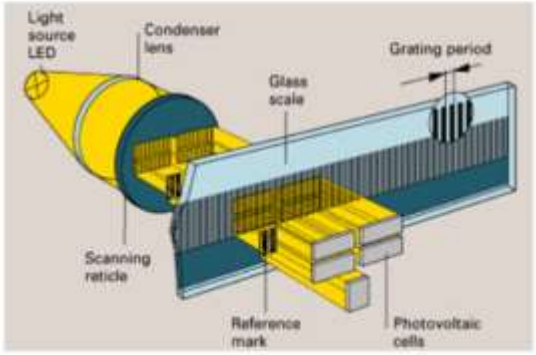


In genere si ottengono risultati migliori con i metodi a spazio fisso, in particolare associando metodi di filtraggio

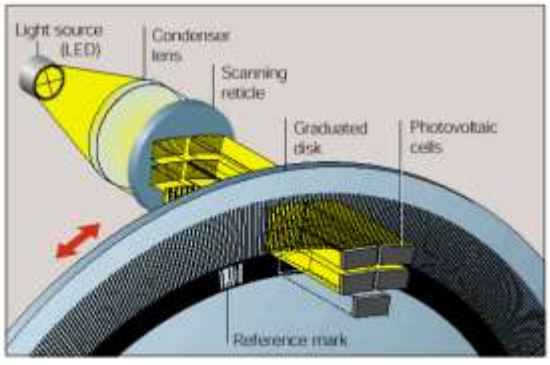
ENCODER AVANZATI

Per grandi precisioni, si usano sistemi basati sull'**effetto di Moiré**: se si fanno scorrere parallelamente e uno sull'altro due reticoli di divisione, si rilevano oscillazioni periodiche di luminosità , che si possono convertire in segnali elettrici.

Riga ottica



Encoder rotativo



ESERCIZIO INCREMENTALE

Quale risoluzione (numeri di impulsi per giro) deve avere un encoder per una misura angolare di 0.25°?

Supponendo che l'encoder sia solidale con un albero di un motore in rotazione alla velocità di 720 r.p.m. (Fig. 1), calcolare il periodo del segnale rilevato dal fototransistor.

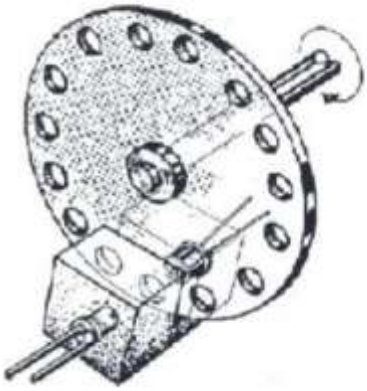


Fig. 1

SOLUZIONE

Numero impulsi = 360° / 0.25 = 1440 (impulsi per ogni giro)

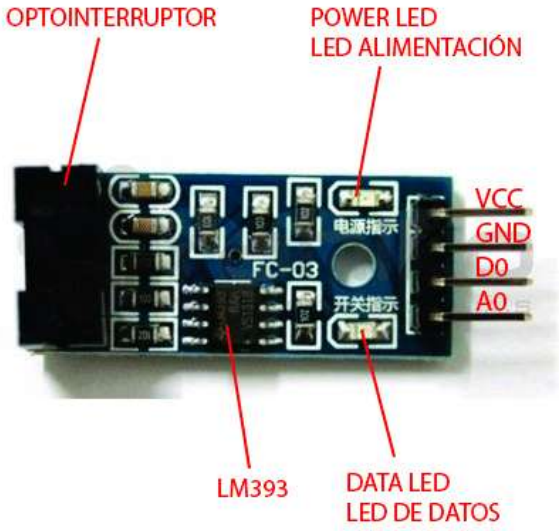
Rotazioni al secondo = 720 / 60 = 12 giri/s

Frequenza (impulsi al secondo) = 1440 • 12 = 17280 Hz

Periodo = 1 / f = 1÷ 17280 = 57,8 µs

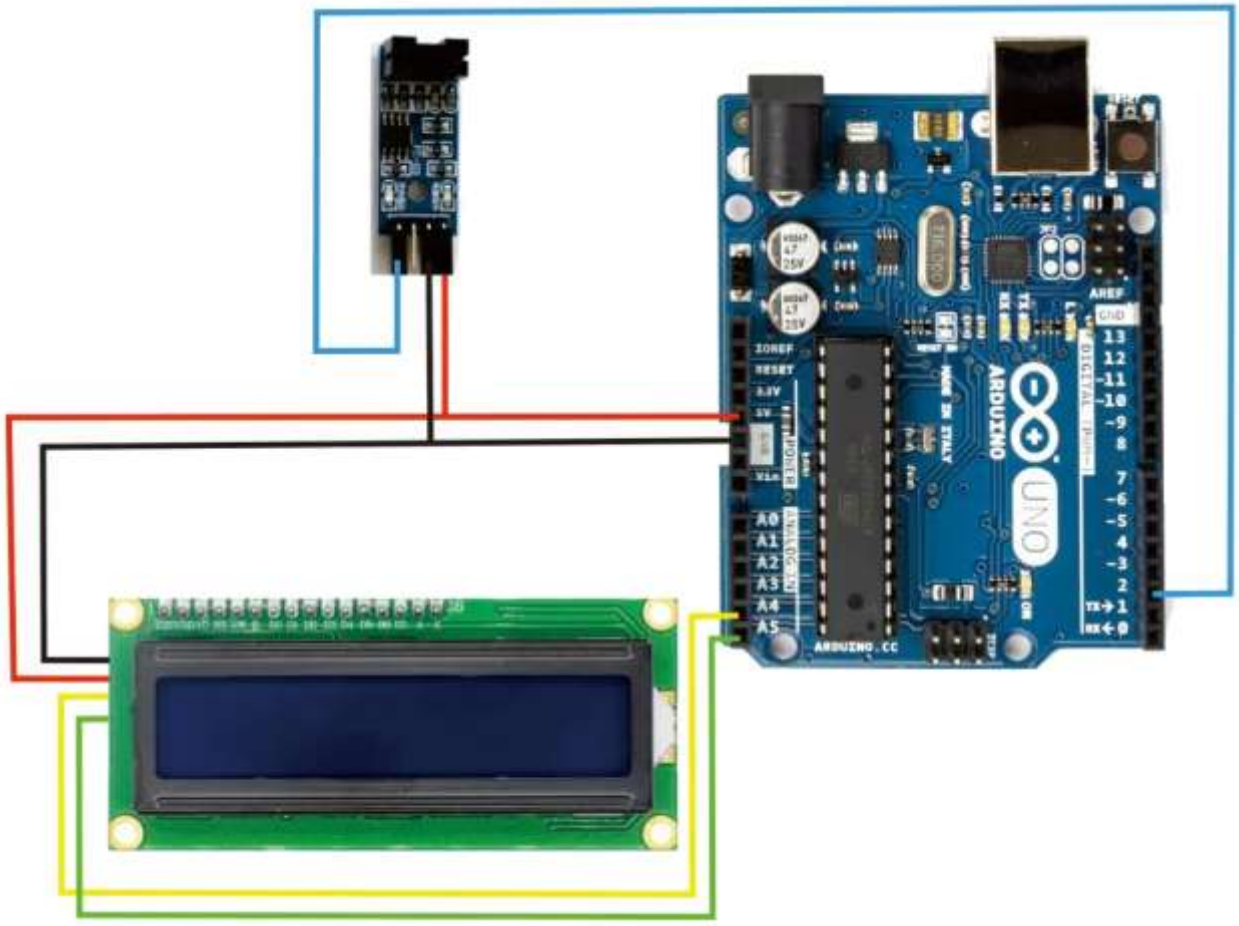
ENCODER OTTICO AD INFRAROSSI

L'encoder ottico a infrarossi (emettitore → fotodiodo IR, ricevitore → fototransistore) è un sensore di velocità.
Viene utilizzato per misurare la velocità di un oggetto rotante come un l'albero di un motore.

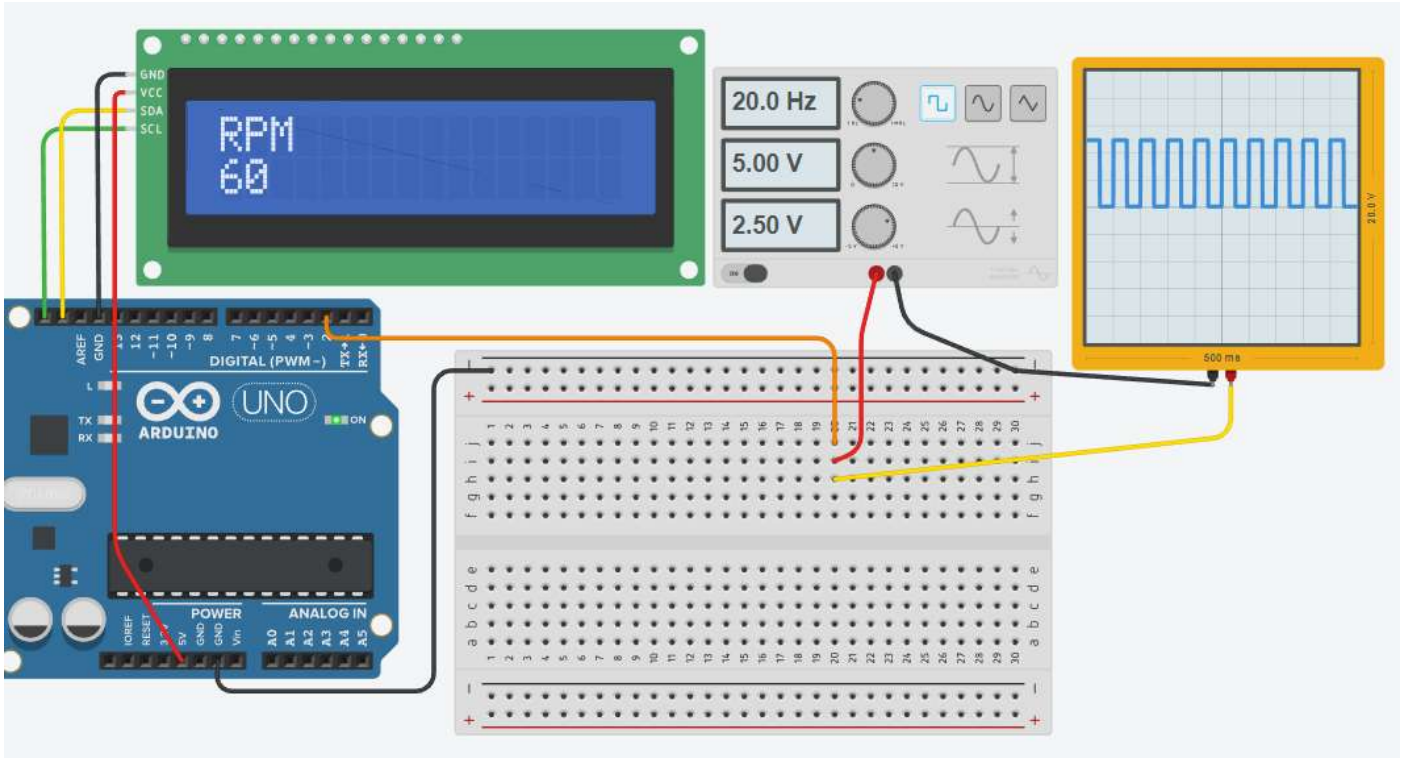


Disco encoder 20 fori da fissare sull'albero motore

Schema reale



SIMULARE L'ENCODER CON UN GENERATORE DI IMPULSI



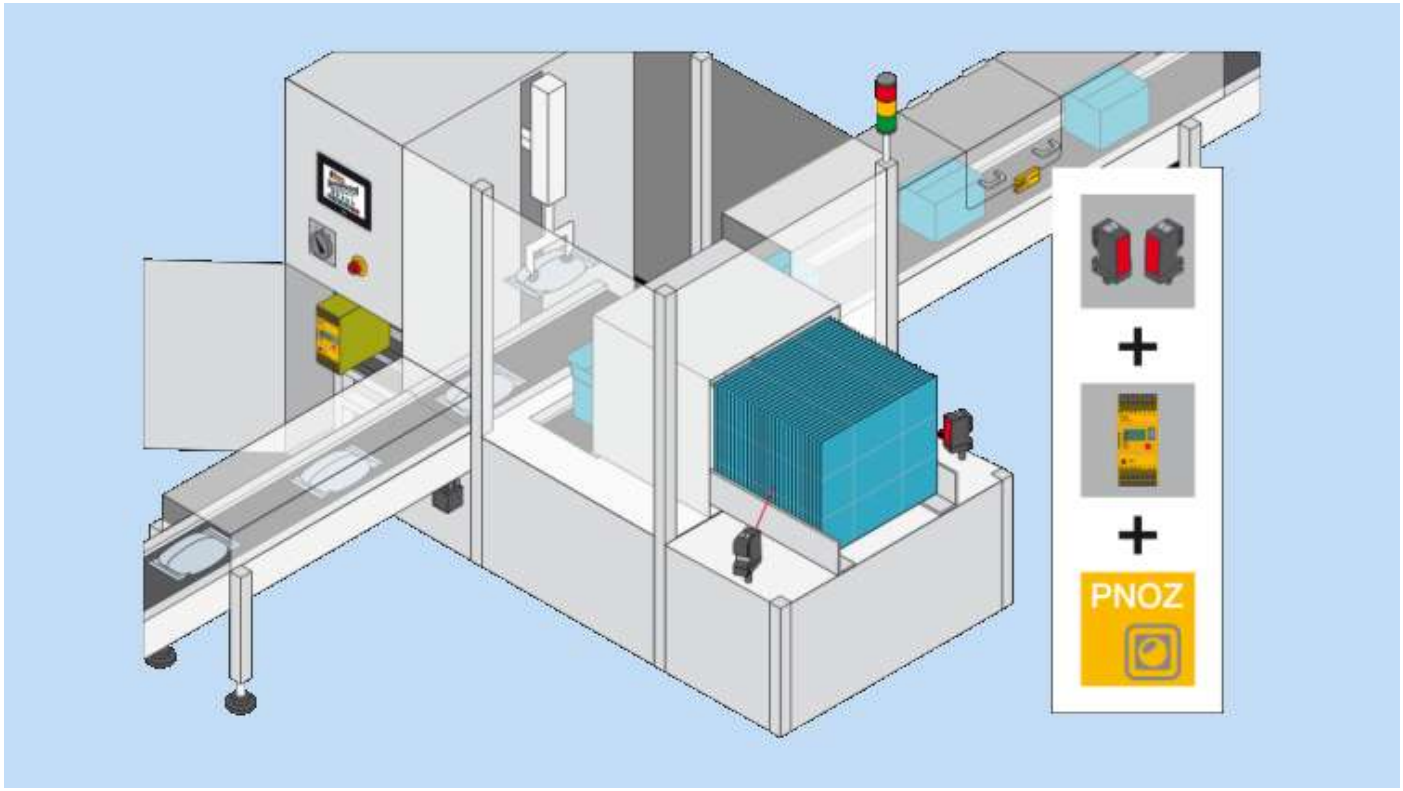
CODICE

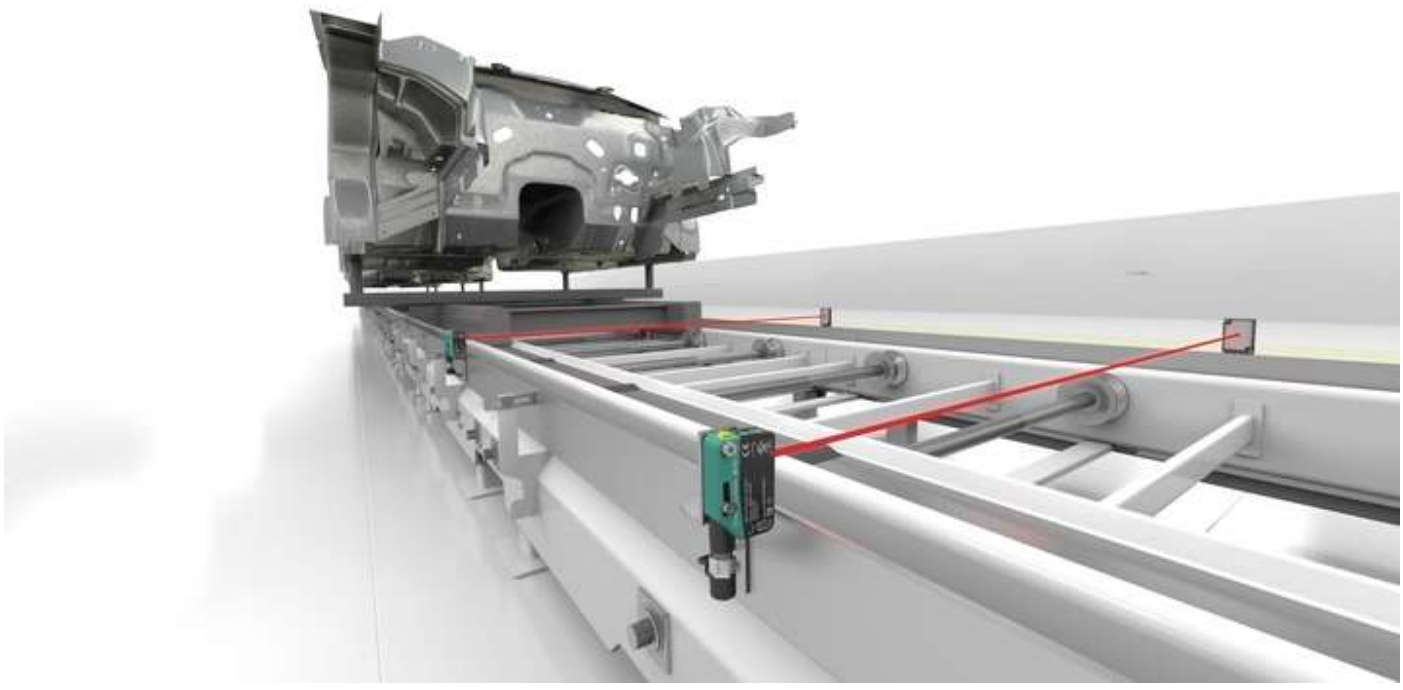
```
#include <Adafruit_LiquidCrystal.h>
Adafruit_LiquidCrystal lcd_1(0);
volatile int contatore = 0;
const int nfori= 20;
int rpm= 0;

void interrupt0()  {  contatore++;  }

void setup() {
  lcd_1.begin(16, 2);
  lcd_1.print("RPM");
  Serial.begin(9600);
  // uso il pin2 per l'interrupt (solo il 2 o il 3 di Arduino)
  attachInterrupt(digitalPinToInterrupt(2),interrupt0,RISING);
}

void loop() {
  delay(1000);
  rpm= 60* contatore / nfori;
  Serial.print(rpm);
  Serial.println(" impulsi");
  lcd_1.setCursor(0, 1);
  lcd_1.print(rpm);

  contatore = 0;
}
```

CODICE con ENCODER

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2); // set the LCD address to 0x27 for a 16 chars and 2 line display

const nholes= 20.0; // numero di fori disco encoder
float rpm = 0;
int pid;
unsigned long millisBefore;
volatile int holes;

void setup()
{
	Serial.begin(9600);
	lcd.init();
	lcd.backlight();
	lcd.setCursor(0, 0);
	lcd.print("Speed Sensor");
	lcd.setCursor(0, 1);
	lcd.print("Test");
	pinMode(2, INPUT);
	attachInterrupt(digitalPinToInterrupt(2), count, FALLING);
	delay(1000);
	lcd.clear();
}

void loop()
{
	print_to_LCD();
	if (millis() - millisBefore > 1000) {
	rpm = (holes / nholes)*60;
	holes = 0;
	millisBefore = millis();
	}
	delay(100);
}

void print_to_LCD() {
	lcd.setCursor(0, 0);
	lcd.print("Holes : ");
	lcd.print(holes);
	lcd.print(" ");
	lcd.setCursor(0, 1);
	lcd.print("RPM : ");
	lcd.print(rpm);
	lcd.print(" ");
}

void count() {
	holes++;
}
```

ESEMPI
APPLICAZIONI SENSORI





SISTEMA DI CONTROLLO QUALITA' SACCHI DI CEMENTO

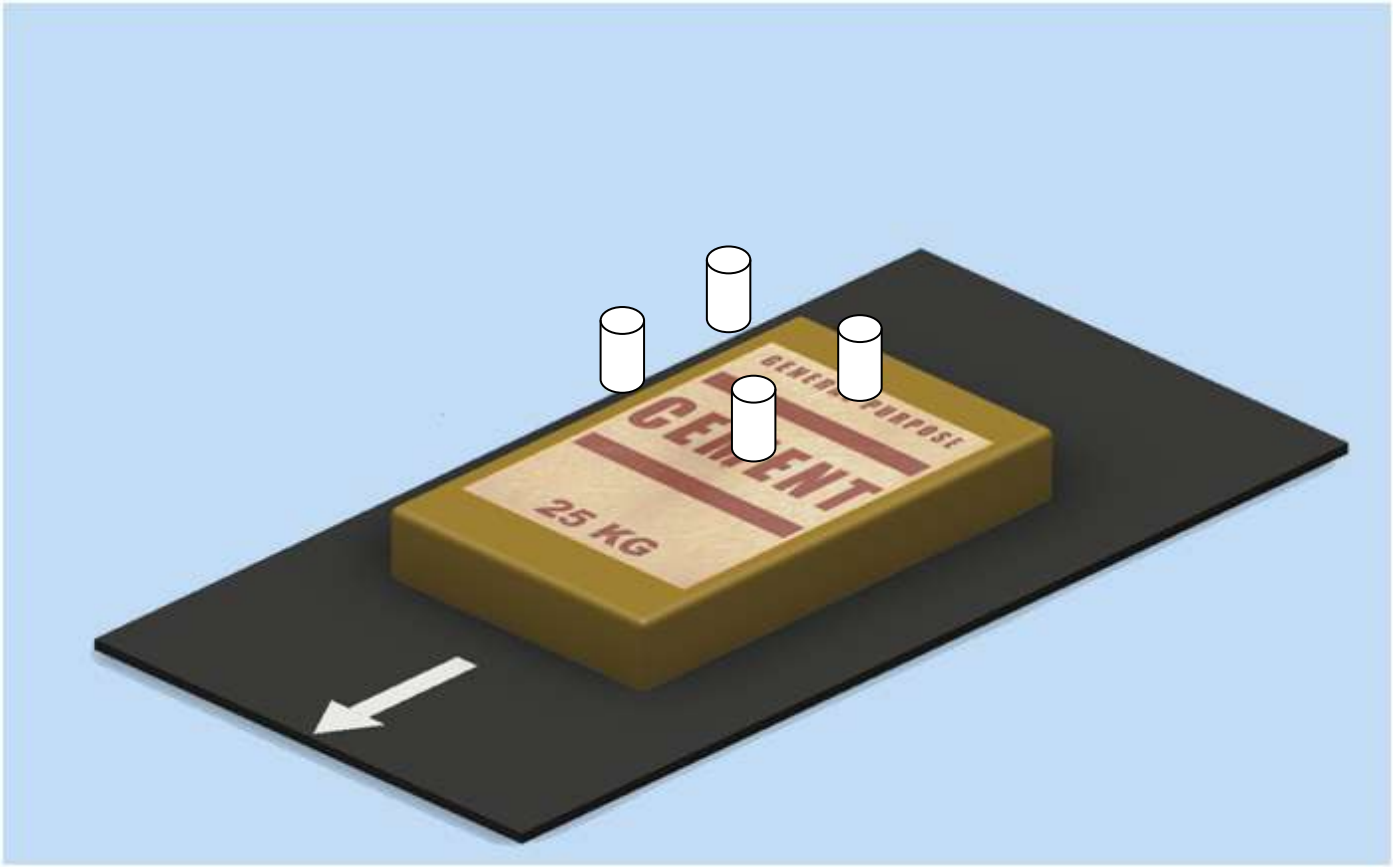
Realizzare un sistema di controllo di qualità dell'integrità di sacchi di cemento su nastro trasportatore.

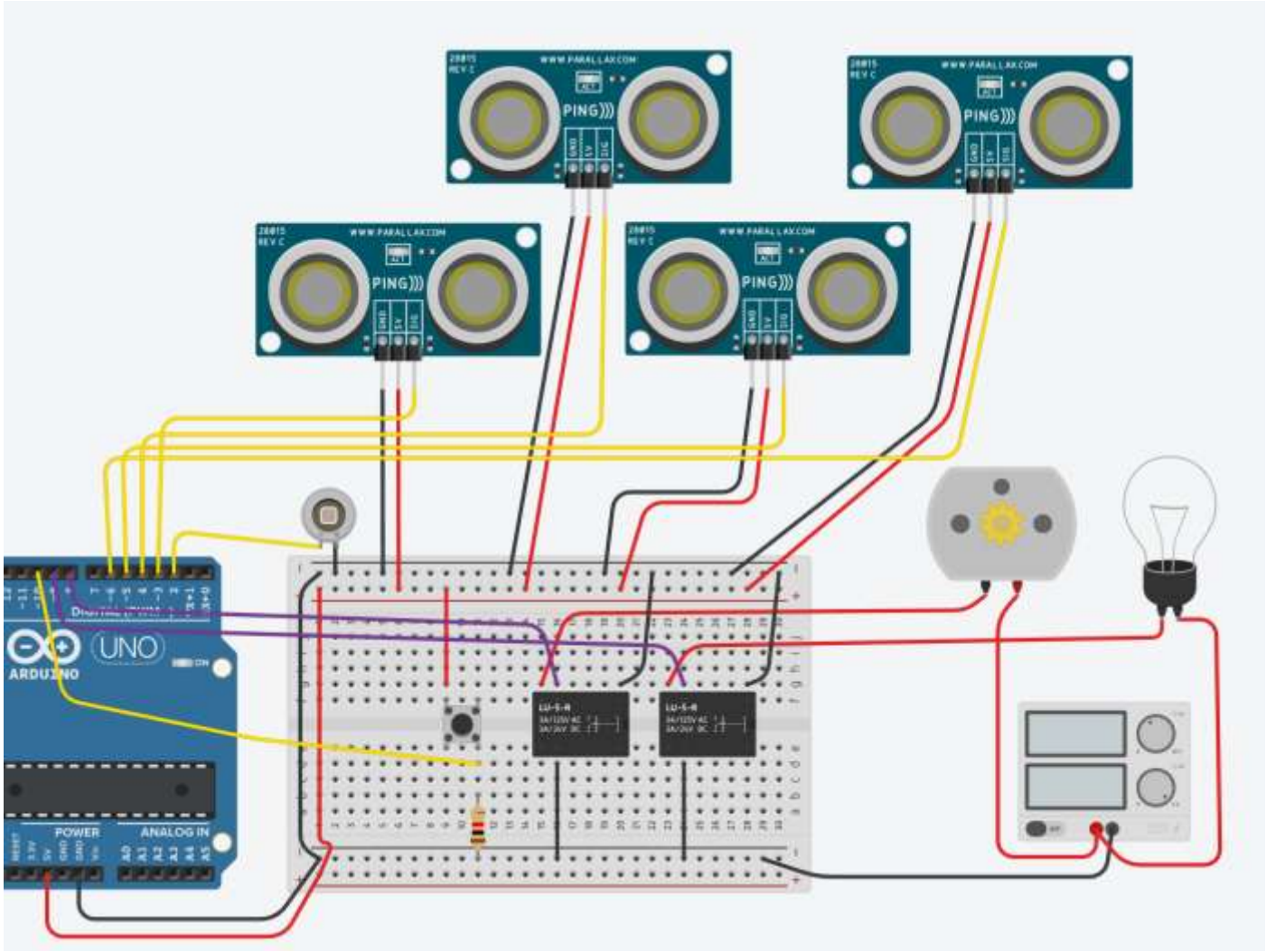
Si utilizzino 4 sensori ad ultrasuoni per misurare la distanza dalla superficie del sacco (in 4 punti diversi).
Se la misura di un sensore è inferiore a una certa distanza di setup (es. 100cm) il sacco è rotto.

Un sensore di prossimità (fotocellula FC) segnala la presenza del sacco sotto i sensori ad ultrasuoni e consente di fermare il nastro trasportatore per effettuare il controllo di qualità (0.5s).

Se il sacco è rotto viene attivata una lampada di emergenza e il nastro trasportatore viene mantenuto fermo.

Solo quando un operatore attiva il pulsante di START il sistema riprende il funzionamento.





CODICE

```
// ingressi sensori
int pinC1 = 3;
int pinC2 = 4;
int pinC3 = 5;
int pinC4 = 6;
int pinFC = 2;
int pinStart=10;

// uscite
int pinM = 8;
int pinEM = 9;

// variabili
int statoFC = 0;
int misC1 = 0;
int misC2 = 0;
int misC3 = 0;
int misC4 = 0;
int statoStart=0;
int statoM=1;
int statoSacco=0; // 0=ok; 1=rotto

void setup()
{
  Serial.begin(9600);
```

```
  pinMode(pinC1, INPUT);
  pinMode(pinC2, INPUT);
  pinMode(pinC3, INPUT);
  pinMode(pinC4, INPUT);
  pinMode(pinStart, INPUT);
  pinMode(pinFC, INPUT_PULLUP);

  pinMode(pinM, OUTPUT);
  pinMode(pinEM, OUTPUT);

  // attivo motore M nastro
  digitalWrite(pinM, HIGH);
  // spengo emergenza EM
  digitalWrite(pinEM, LOW);
}

void loop()
{
 // leggo stato fotocellula FC
 statoFC = digitalRead(pinFC);
 Serial.print("FC="); Serial.println(statoFC);

 // controllo presenza SACCO
 if (statoFC==1) {
  Serial.println("Presenza SACCO");
  // fermo motore M
  digitalWrite(pinM, LOW); statoM= LOW;  // fermo

  // misura distanze in 4 punti; se una inferiore a 100 sacco rotto
  misC1 = readUltrasonicDistance(pinC1, pinC1);
  delay(10); // Wait for 100 millisecond(s)
  Serial.print("C1="); Serial.println(misC1);
  misC2 = readUltrasonicDistance(pinC2, pinC2);
  delay(10); // Wait for 100 millisecond(s)
  Serial.print("C2="); Serial.println(misC2);
  misC3 = readUltrasonicDistance(pinC3, pinC3);
  delay(10); // Wait for 100 millisecond(s)
  Serial.print("C3="); Serial.println(misC3);
  misC4 = readUltrasonicDistance(pinC4, pinC4);
  delay(10); // Wait for 100 millisecond(s)
  Serial.print("C4="); Serial.println(misC4);
  delay(500);

  // se una delle distanze è inferiore a 100 il sacco è rotto
  if ( (misC1<100) || (misC2<100) || (misC3<100) || (misC4<100) ) {
   Serial.println("SACCO rotto");
   statoSacco= 1;
   // accendo energenza EM
   digitalWrite(pinEM, HIGH);
  }
  // se sacco rotto rimosso e quello attuale non è rotto
  else if (statoSacco==0) {
   Serial.println("SACCO OK");
   statoSacco= 0;
   // accendo motore M e spengo EM
   digitalWrite(pinM, HIGH); statoM= HIGH;
   digitalWrite(pinEM, LOW);
   statoM= HIGH;
  }
 }

 // controllo stato pulsante START
 statoStart= digitalRead(pinStart);
 Serial.print("Start="); Serial.println(statoStart);
 if (statoStart== HIGH) {
  Serial.println("Avvio NASTRO");
  statoSacco=0;  // sacco rotto rimosso --> riparte nastro
```

```
    digitalWrite(pinM, HIGH); statoM= HIGH;
    digitalWrite(pinEM, LOW);
   }

  delay(1000);
}

// torna distanza in cm sulla base del tempo rilevato dal sensore
long readUltrasonicDistance(int triggerPin, int echoPin)
{
  pinMode(triggerPin, OUTPUT);  // Clear the trigger
  digitalWrite(triggerPin, LOW);
  delayMicroseconds(2);
  // Sets the trigger pin to HIGH state for 10 microseconds
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);
  pinMode(echoPin, INPUT);
  // Reads the echo pin, and returns the sound wave travel time in microseconds
  return 0.01723 * pulseIn(echoPin, HIGH);
}
```

SISTEMA DI CONTROLLO QUALITA' SACCHI DI CEMENTO CON SCARTO

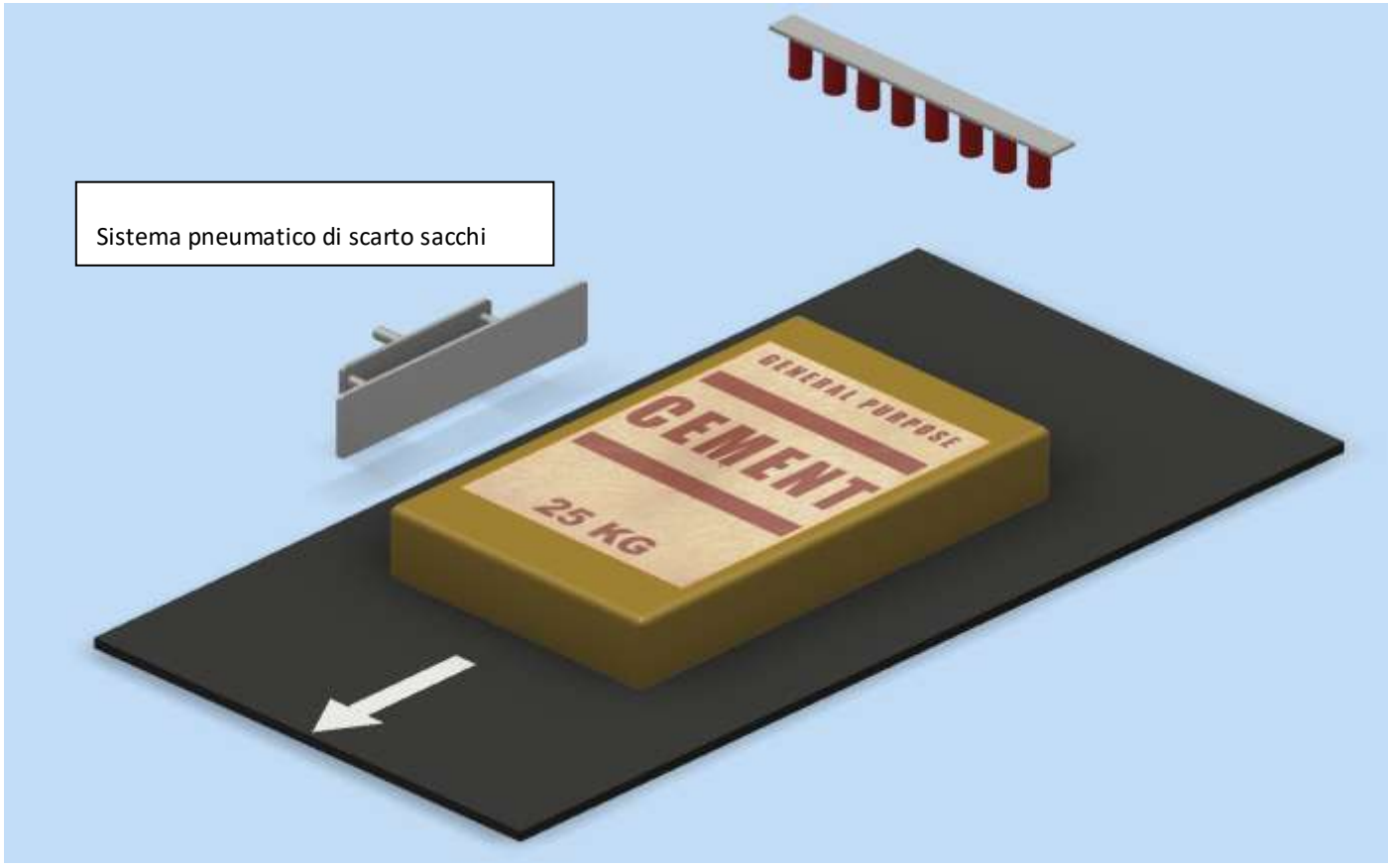
Progettare un sistema di controllo qualità sacchi di cemento da 25 Kg.

Un serie di 8 sensori analogici fornisce lo stato del cartone del sacco in tempo reale mentre questo scorre sul nastro trasportatore. Prevedere una lettura ogni 0.5sec per un totale di 10 letture.

Per semplificare la simulazione in Thinkercad usare un solo sensore ad ultrasuoni U1.
Un sensore di prossimità (fotocellula FC) indica la presenza del sacco davanti al sistema di scarto (controllo qualità terminato).

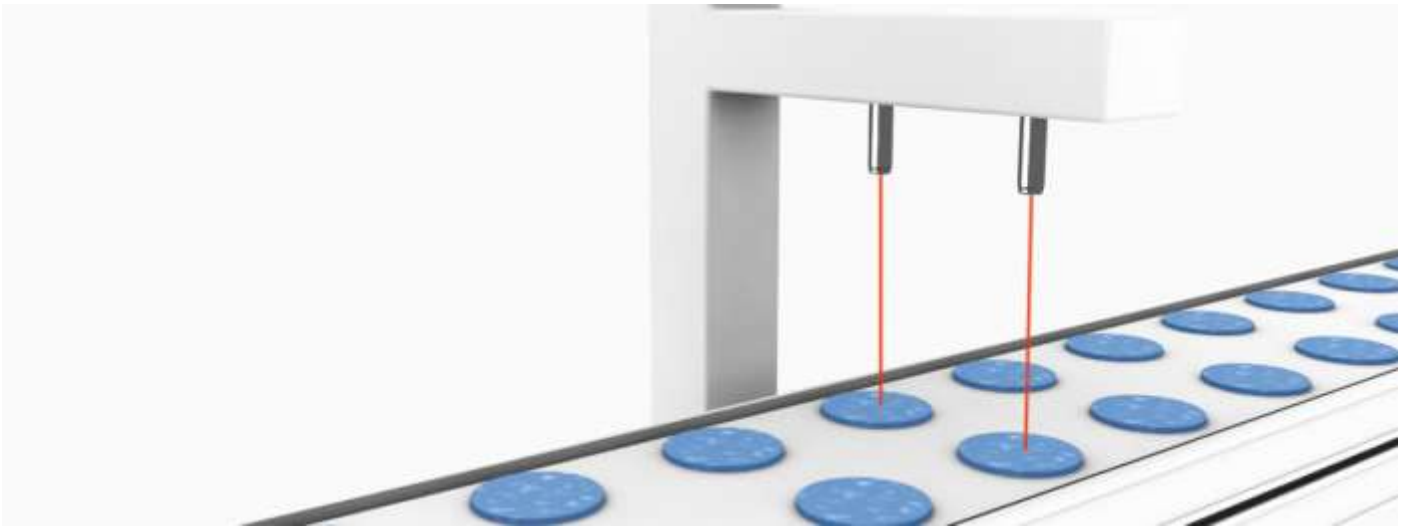
Se il sacco non ha passato il controllo di qualità allora il nastro viene fermato e un sistema pneumatico spinge il sacco fuori dal nastro traspostatore.

Dimensionare il cilindro pneumatico necessario per spostare il sacco di cemento ipotizzando un coefficiente di attrito pari a 0.8. Scegliere poi da catalogo MW il cilindro idoneo.

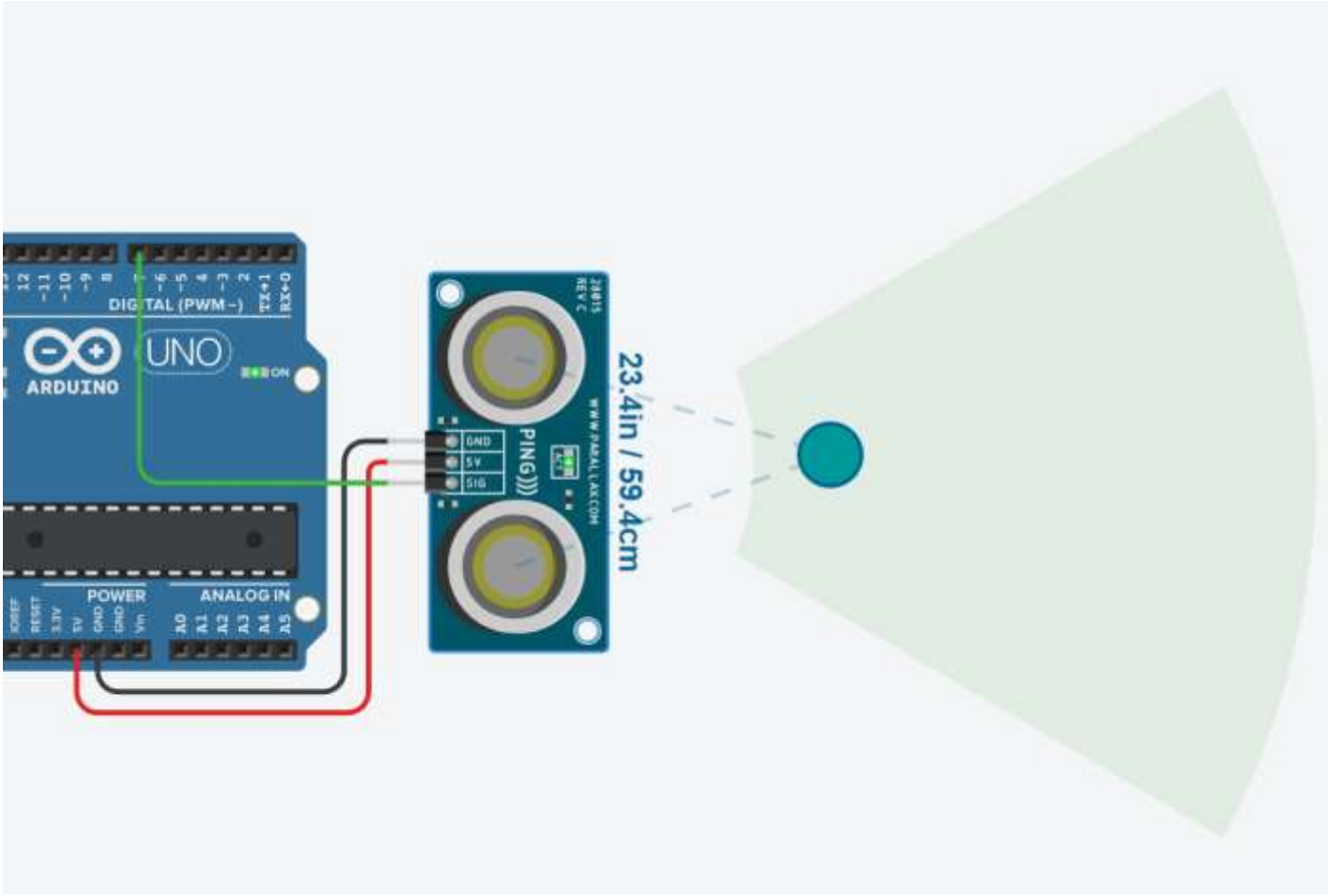


SISTEMA CONTA PEZZI CON SENSORE ULTRASUONI

Realizzare un sistema “conta pezzi” che passano di fronte a un sensore ad ultrasuoni (es. nastro trasportatore).



Schema Arduino con sensore Parallax



CODICE

```
int state= 0;        // pezzo non presente
int laststate=0;   // pezzo non presente
int counter= 0;   // numero di pezzi passati
int cm = 0;         // distanza sotto la quale si considera pezzo presente

void setup()
{
  pinMode(7, INPUT);  // sensore ultrasuoni
  Serial.begin(9600);
}

void loop()
{
  // rilevo fronte di discesa segnale sensore
  if (state==1 &&  laststate==0) {
     counter++;
     Serial.print(counter);
     Serial.println(" pezzi");
  }
  laststate = state;  // aggiorno ultimo stato

  // measure the ping time in cm
  cm = 0.01723 * readUltrasonicDistance(7, 7);
  delay(10); // Wait for 100 millisecond(s)
  if (cm <=100) { state= 1; } else { state= 0; }
}

long readUltrasonicDistance(int triggerPin, int echoPin)
{
  pinMode(triggerPin, OUTPUT);  // Clear the trigger
  digitalWrite(triggerPin, LOW);
  delayMicroseconds(2);
  // Sets the trigger pin to HIGH state for 10 microseconds
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);
  pinMode(echoPin, INPUT);
  // Reads the echo pin, and returns the sound wave travel time in microseconds
  return pulseIn(echoPin, HIGH);
}
```

ATTUATORI

In ingegneria, gli attuatori sono capaci di trasformare un segnale di input (normalmente elettrico) in movimento, come esempi di attuatori sono i motori elettrici, pistoni idraulici, relè, polimeri elettroattivi, attuatori piezoelettrici, ecc.

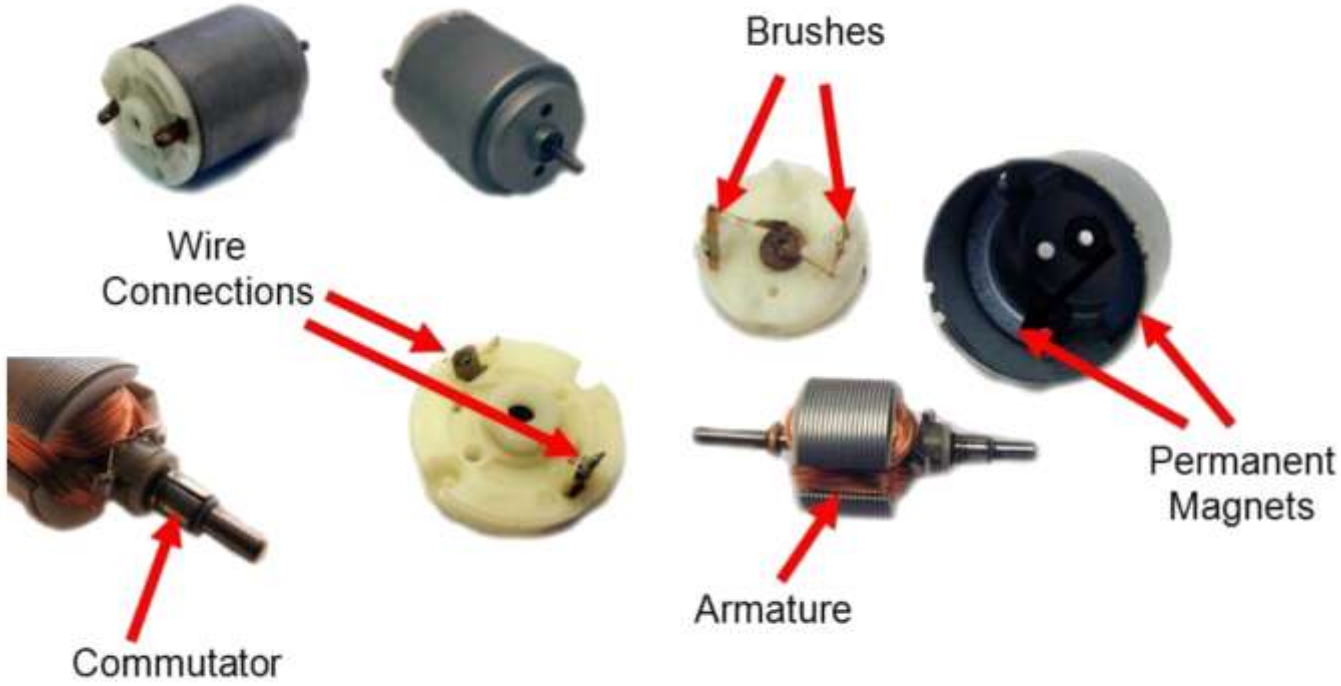
I motori sono usati soprattutto quando si richiedono movimenti circolari, ma possono essere impiegati per applicazioni lineari trasformando un movimento da circolare a lineare utilizzando un trasduttore a vite senza fine. D'altra parte, alcuni attuatori, come quelli piezoelettrici, sono intrinsecamente lineari.

Attuatori lineari ibridi

MOTORE IN CORRENTE CONTINUA (C.C.)

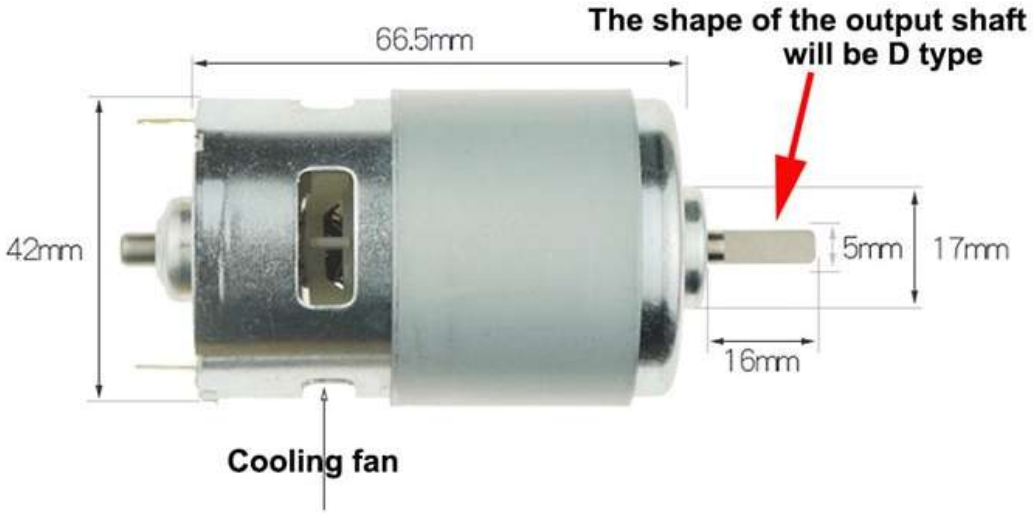
Un motore in corrente continua (CC) è una macchina elettrica che converte l'energia elettrica (V, I) in energia meccanica (Coppia motrice , n° giri).

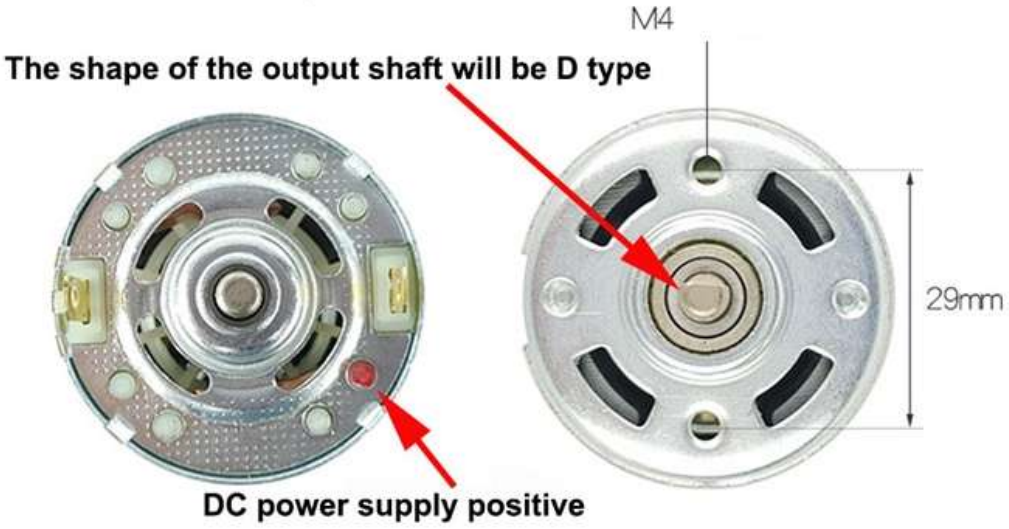
Applicando la massima tensione per cui il motore è stato progettato si otterrà la massima velocità.
Diminuendo la tensione applicata il numero di giri calerà di conseguenza (in generale insieme alla coppia motrice).



https://www.youtube.com/watch?v=peGZkxusheI

775 D SHAFT



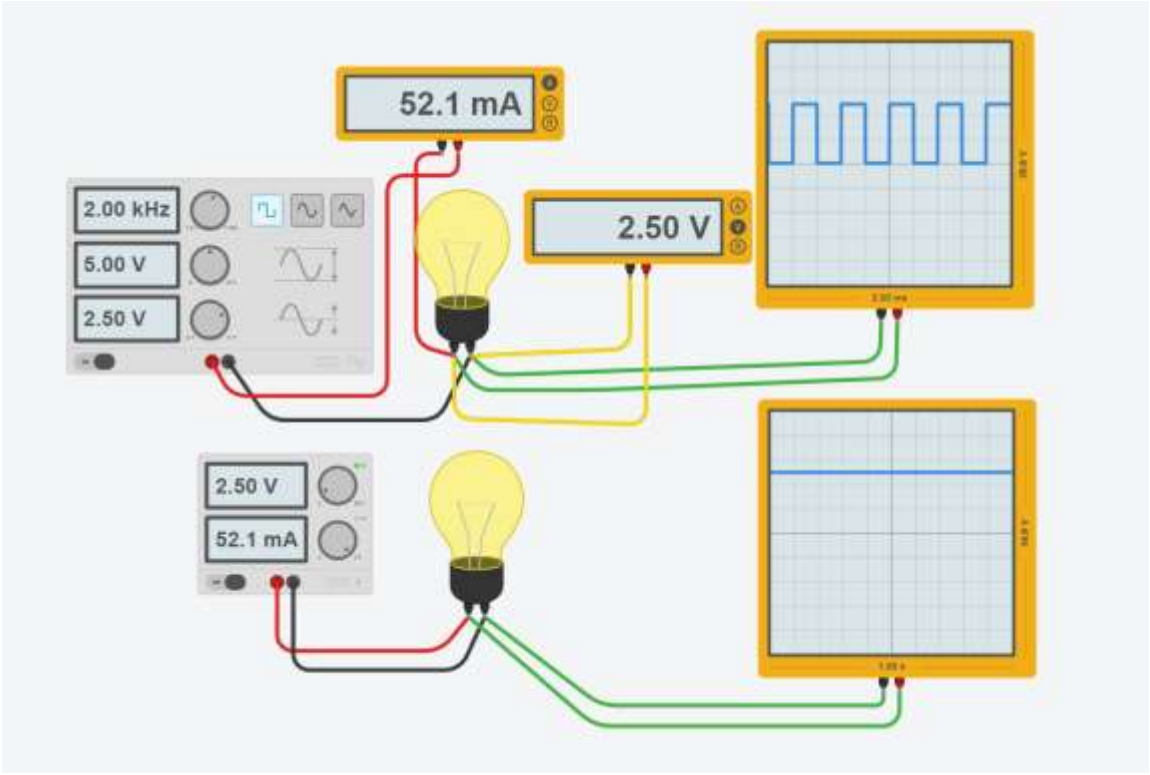


PWM (PULSE WIDE MODULATION): MODULAZIONE DI LARGHEZZA D'IMPULSO

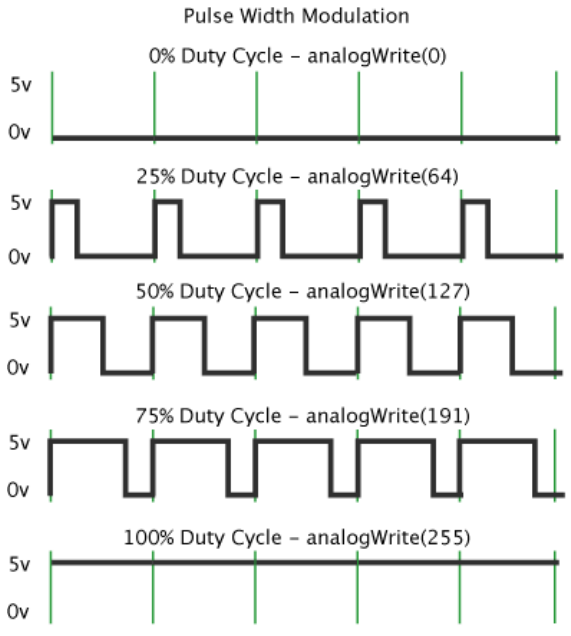
Un microcontrollore come Arduino non è in grado di generare un segnale analogico di tensione.
Tuttavia utilizzando la PWM è in grado di generare un'onda quadra ad ***alta frequenza*** modulata in ampiezza che viene percepita dalla maggior parte degli utilizzatori (resistenze, lampadine, motori CC) come una tensione continua.

Il circuito sottostante mostra l'effetto di una tensione periodica a 2kHz a onda quadra di ampiezza 5V e duty cycle del 50% (frazione di tempo in cui l'onda è allo stato attivo in proporzione al periodo totale).

Si nota, dalla tensione media e dalla corrente assorbita, che l'effetto sulla lampada è lo stesso generato dall'alimentazione a CC a 2.5V.



Con Arduino si può generare un segnale PWM a 8 bit (255 combinazioni) da 0 a 5V (risoluzione = 5/255=0.196V).



→ 5*0 = 0V

→ 5*0,25 = 1,25V

→ 5*0,5 = 2,5V

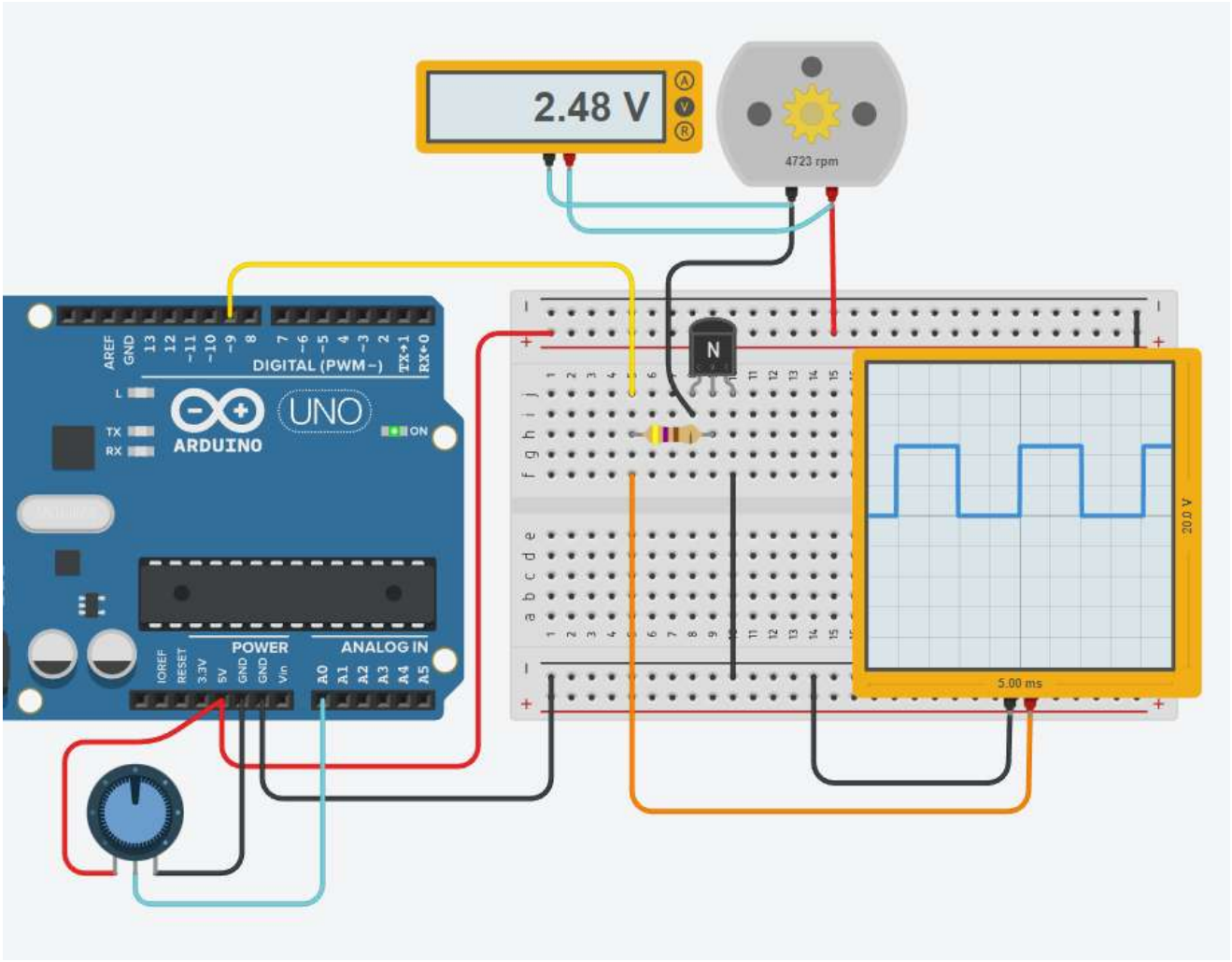
→ 5*0,75 = 3,75V

→ 5*1 = 5V

ESERCIZIO PWM MOTORE CC

Regolare la velocità di rotazione del motore mediante un potenziometro e la tecnica PWM.

NB: ai capi di un motore va sempre messo un diodo di protezione del transistor di comando (anodo sul +) che è stato omesso per semplificare lo schema.



Il motore DC non può essere alimentato direttamente da un PIN di Arduino poiché la corrente richiesta dal motore è superiore a quella fornita da un PIN. Se la corrente richiesta dal motore è di poche centinaia di mA si può usare l'uscita 5V di Arduino. Se si usa un alimentatore dedicato è necessario mettere la massa in comune con quella di Arduino per garantire il corretto funzionamento (serve lo stesso riferimento per la massa).

La regolazione della velocità del motore DC si effettua in 2 modi:
- regolando la tensione a capi del motore (ad esempio con un potenziometro)
- regolando il tempo (PWM) in cui la tensione massima di alimentazione del motore viene applicata ai suoi capi

Il 2° metodo permette di regolare la velocità mantenendo anche la coppia motrice sempre elevata mentre nel 1° modo la coppia cala proporzionalmente alla tensione applicata.

Per fornire una corrente sufficiente al motore è necessario utilizzare un amplificatore (transistor) comandato in PWM da un PIIN di Arduino.

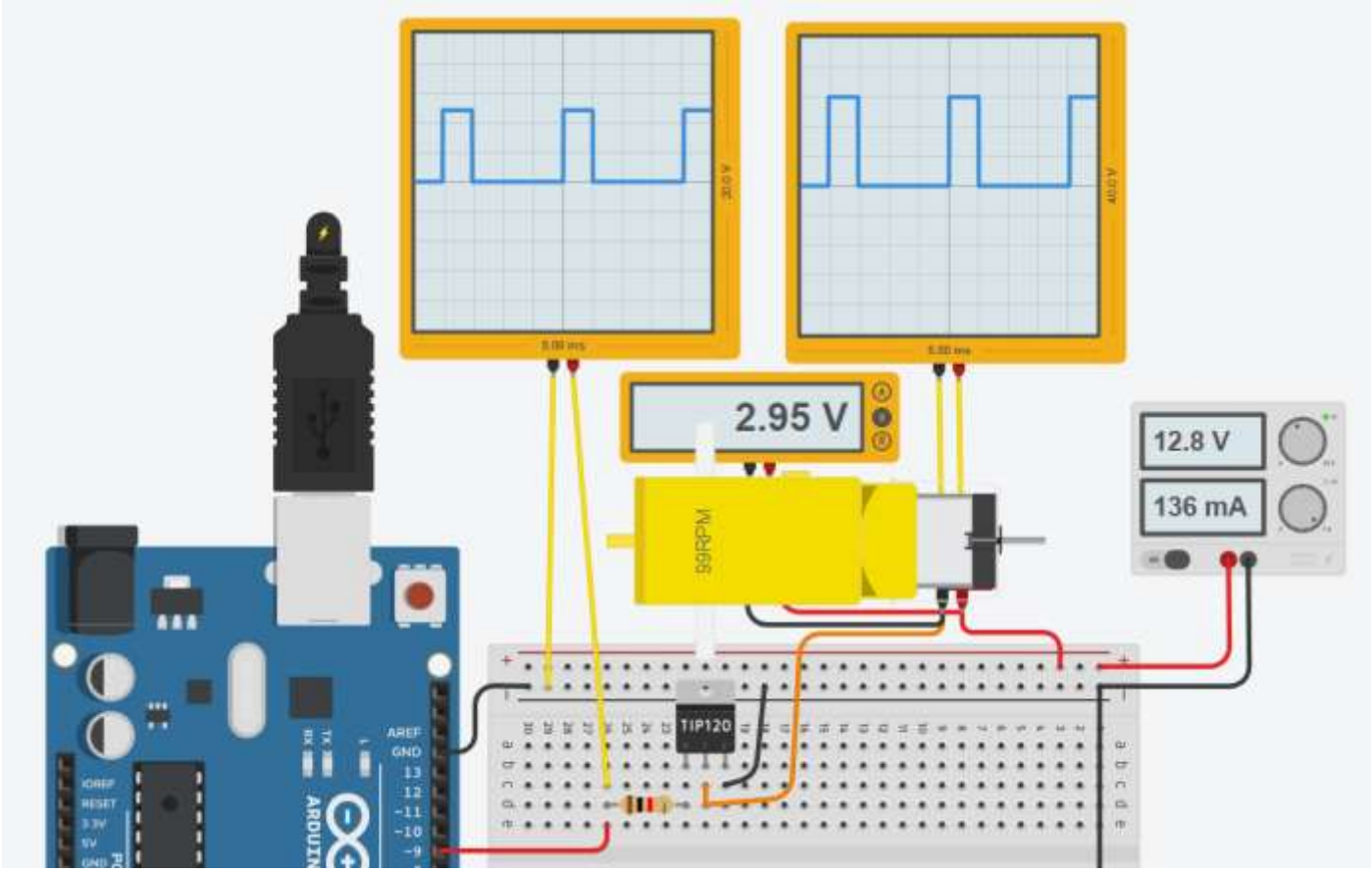
L'oscilloscopio permette di visualizzare il segnale di regolazione PWM (0-5V) generato da un PIN di Arduino.

CODICE

```
int motorPin = 9;  // PWM
int potPin = A0;   // POTENTIOMETER
int potValue = 0;

void setup()
{
  pinMode(potPin, INPUT);
  pinMode(motorPin, OUTPUT);
  Serial.begin(9600);
}

void loop()
{
  // read the value from the sensor
  if (analogRead(potPin) != potValue)
  {
    potValue = analogRead(potPin);
    analogWrite(motorPin, potValue/4);
    Serial.println(potValue);
  }
  delay(20); // Wait for 20ms
}
```

ESERCIZIO RICAVARE LA CURVA “V- N°” E “V-POT.” DEL MOTORE C.C. A 12V ASSEGNATO

Per ricavare la curva “V-n°” e “V-Pot.” del motore si utilizza la tecnica PWM facendo variare la tensione sul motored a 0 a 12V con passo 1 volt e leggendo il numero di giri sul corpo del motore.
Tramite la tabella in EXCEL disegnare i grafici x-y.



Nota: la tensione dell’alimentatore deve essere 12.8V per compensare la Vce del transistor.

Codice

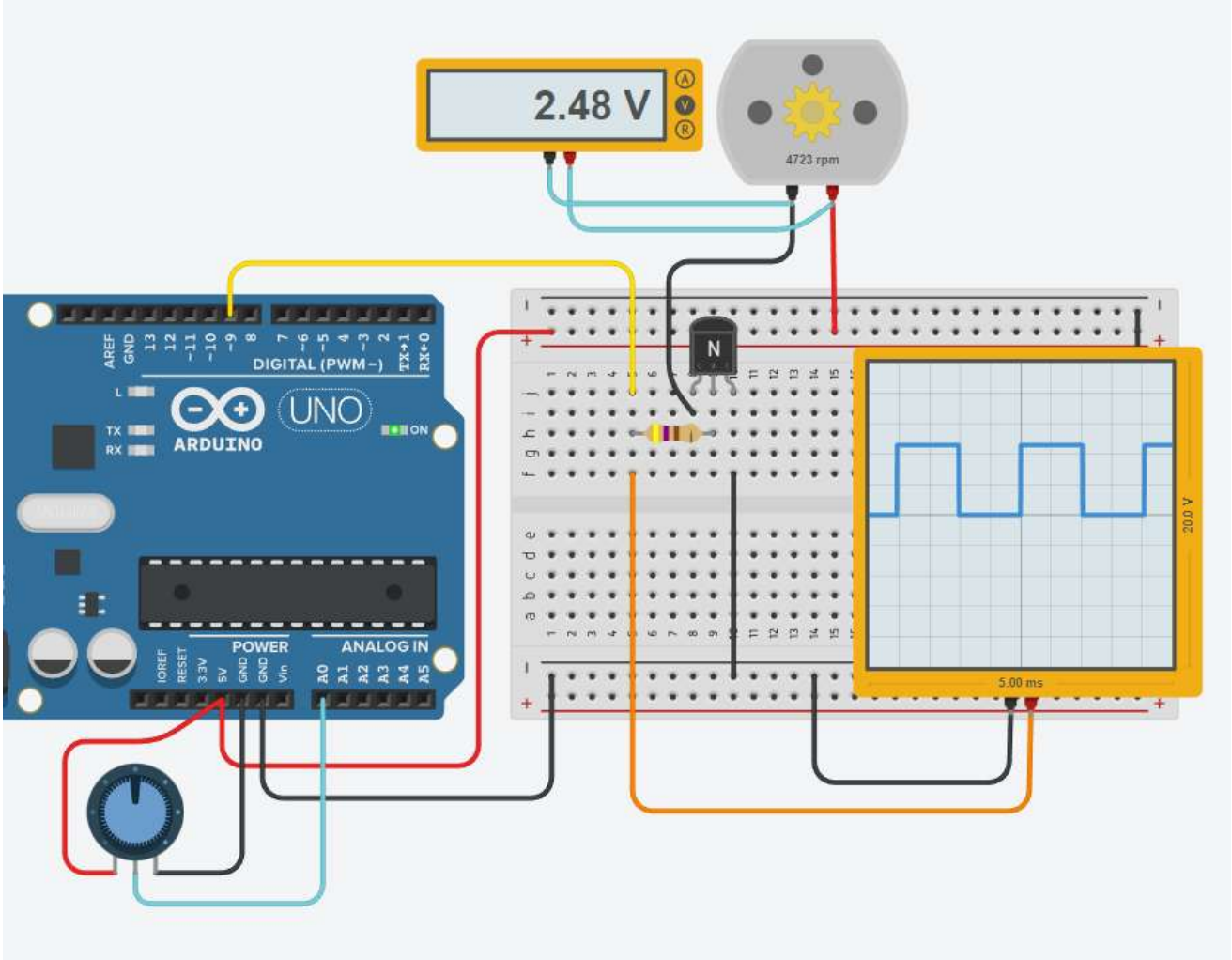
```
#define DC_MOTOR_PIN 9

void setup() {
  pinMode( DC_MOTOR_PIN, OUTPUT );
}

void loop() {
  // Faccio varia tensione 0->12V passo 1v
  for( int i = 1; i <= 12; i=i+1 ){
    analogWrite(DC_MOTOR_PIN, int(255*i/12));
    delay(1000);
  }

}
```

Tensione	N° giri/min	I (mA)	Pot. (W)
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			

ESERCIZIO PWM MOTORE CC + COMANDI SU SERIALE

Regolare la velocità di rotazione del motore con un comando inviato tramite il monitor seriale.
I comandi da inviare sulla seriale sono numeri compresi tra 0 e 100 (0 → 100% della velocità massima).

NB: ai capi di un motore va sempre messo un diodo di protezione del transistor di comando (anodo sul +) che è stato omesso per semplificare lo schema.



CODICE PER RILEVARE NUMERI SULLA SERIALE

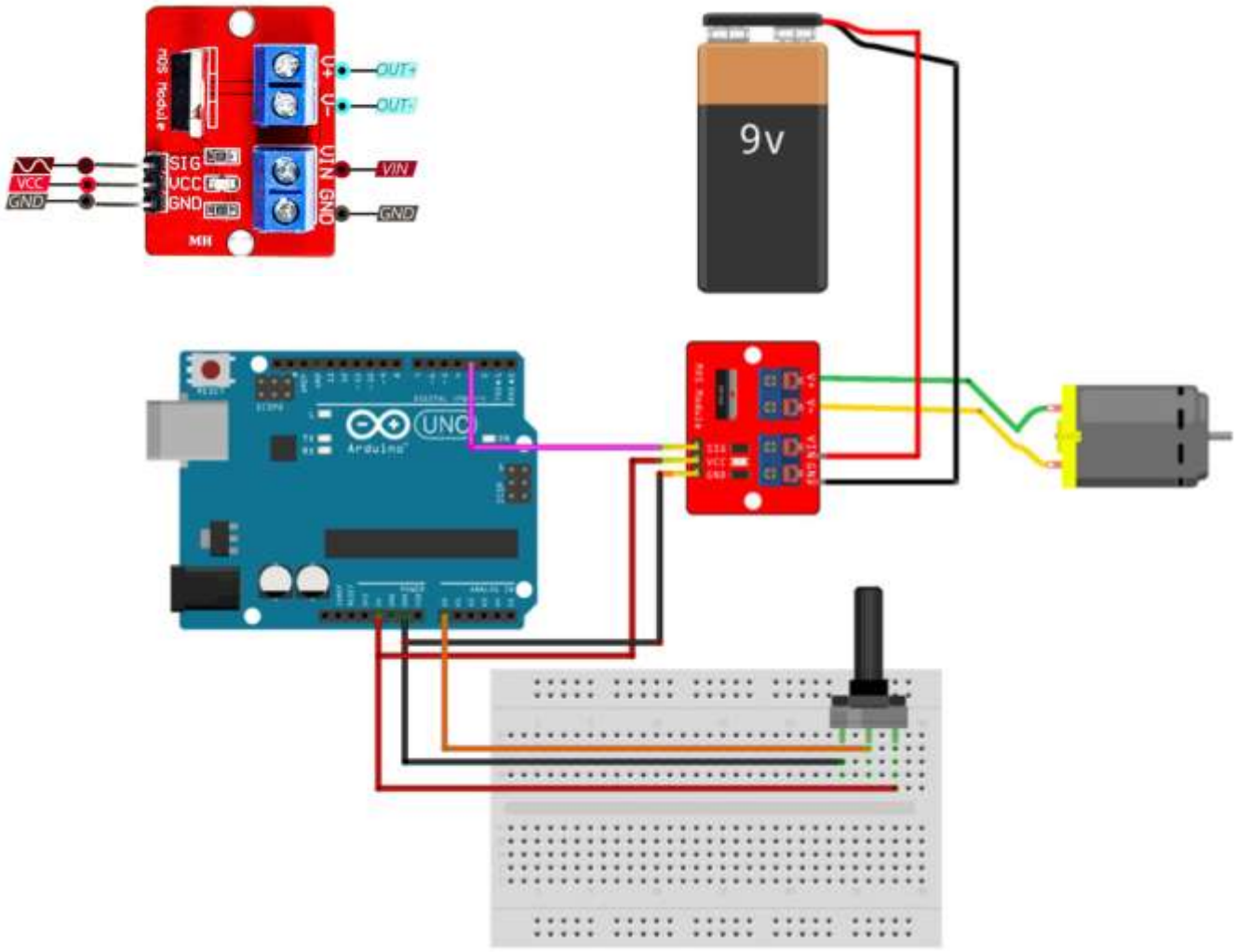
```
if (Serial.available() > 0)
{
   Int numeroSeriale = Serial.parseInt();  // converte in un numero i caratteri ricevuti sulla porta seriale
   Serial.println(numeroSeriale);

   if (numeroSeriale == 0) {
     Serial.println(numeroSeriale);
   }
   else if (numero Seriale==100) {
     Serial.println(numeroSeriale);
  }
  else {
    Serial.println("non valido");
  }
}
```

CODICE

```
int motorPin = 9;  // PWM
int potPin = A0;  // POTENTIOMETER
int potValue = 0;
int numeroSeriale;   // variabile che contiene i dati ricevuti sulla seriale

void setup()
{
  pinMode(potPin, INPUT);
  pinMode(motorPin, OUTPUT);
  Serial.begin(9600);
}

void loop()
{
  //SERIALE: controllo se ci sono dati …
  if (Serial.available() > 0)
  {
    numeroSeriale = Serial.parseInt();  // converte in un numero i caratteri ricevuti sulla porta seriale
    Serial.println(numeroSeriale);

    if (numeroSeriale >=0 && numeroSeriale <=100)
    {
      int speed = numeroSeriale *255 / 100;  // converte 0-100 in 0-255
      Serial.println(speed);
      analogWrite(motorPin, speed);
    }
  }

  // read the value from the sensor
  if (analogRead(potPin) != potValue)
  {
    potValue = analogRead(potPin);
    analogWrite(motorPin, potValue/4);
    Serial.println(potValue);
  }

  delay(20); // Wait for 20ms
}
```

REGOLAZIONE VELOCITA' MOTORE C.C. CON MODULO MOSFET IRF520

Regolare il numero di giri del motore CC tramite un potenziometro e il modulo MOSFET IRF520.
Attenzione a non superare la tensione massima richiesta dal motore CC.



"non simulabile"

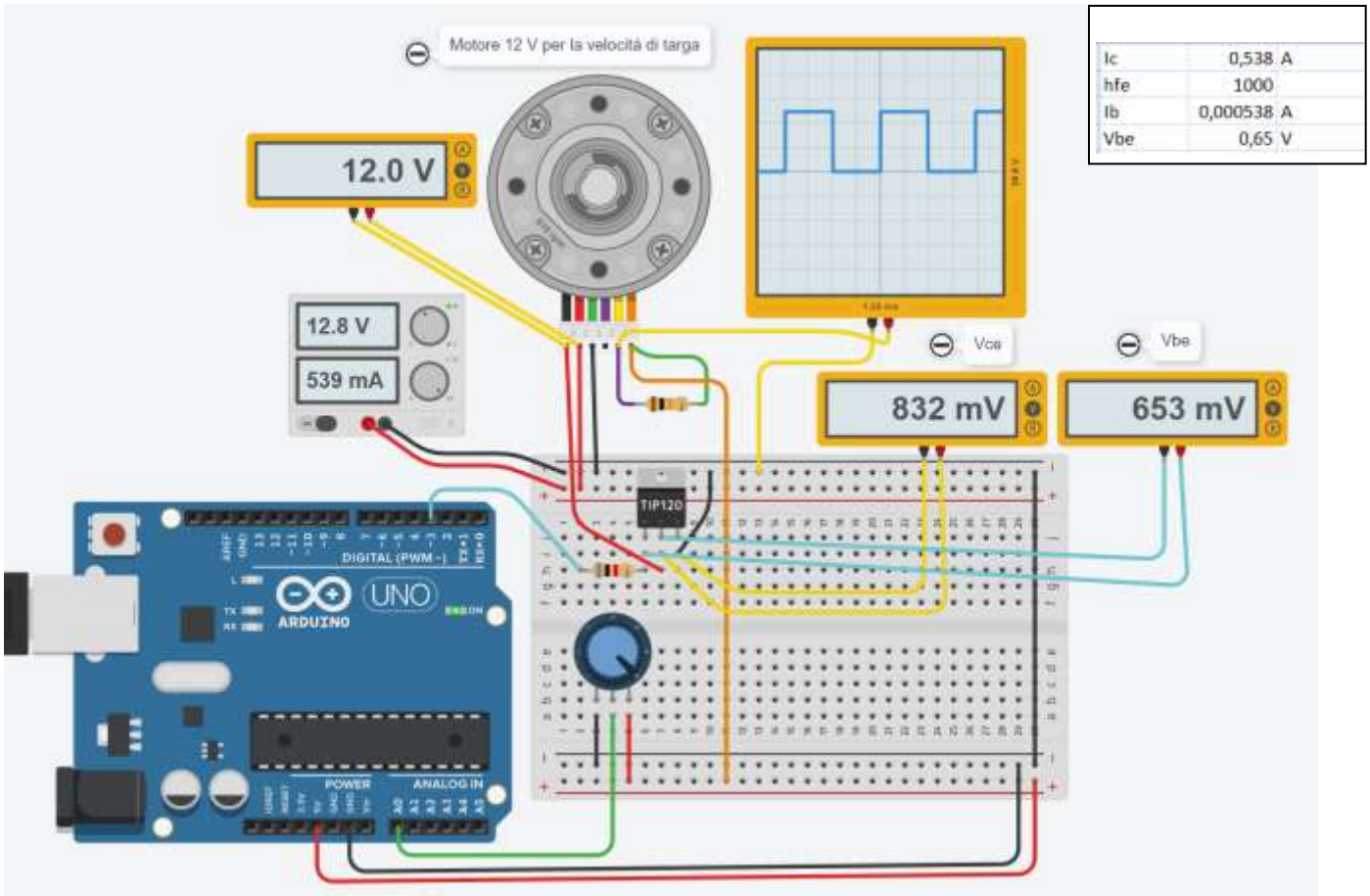
CODICE

```
#define PWM 3  // solo alcuni PIN sono abilitati a uscita PWM
int pot;
int out;

void setup() {
  Serial.begin(9600);
  pinMode(PWM,OUTPUT);
}

void loop() {
  pot=analogRead(A0);
  out=map(pot,0,1023,0,255);  // 255 → massima tensione → massima velocità
  analogWrite(PWM,out);
}
```

ENCODER

L'encoder è un apparato elettromeccanico che converte la posizione angolare del suo asse rotante in un segnale elettrico digitale. Viene generalmente collegato all'albero di un motore per misurare il numero di giri o lo spostamento angolare.

Si vuole regolare la velocità di rotazione di un motore DC di potenza (12V – 550mA) dotato di encoder e si vuole visualizzare il segnale fornito dall'encoder sull'oscilloscopio.

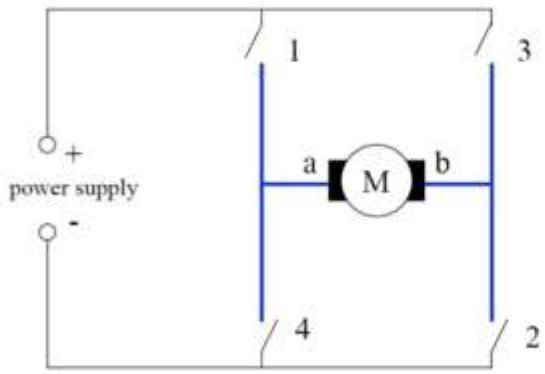


CODICE

```
#define PWM_OUT 3
#define POT_IN A0
int setpoint = 0;

void setup() {
Serial.begin(9600); // begins the serial communication
pinMode(POT_IN, INPUT); // sets the potentiometer as an input and controller
pinMode(PWM_OUT, OUTPUT); // sets pin 3 as a PWM output for the speed control
}

void loop() {
  setpoint = analogRead(POT_IN);
  analogWrite(PWM_OUT, setpoint/4);
  Serial.println(setpoint);
}
```

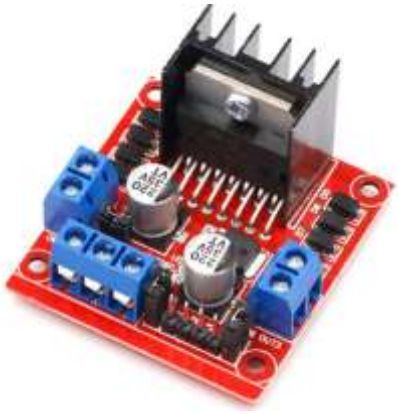
INVERSIONE VERSO DI ROTAZIONE MOTORE C.C. CON 2 RELE'

Per regolare il verso di rotazione di un motore CC sono necessari due relè opportunamente collegati al motore.
Questo sistema NON consente anche la regolazione del numero di giri del motore.
Per ottenere il duplice effetto (verso e regolazione velocità) è necessario un **ponte ad H** costituito da 4 transistor.



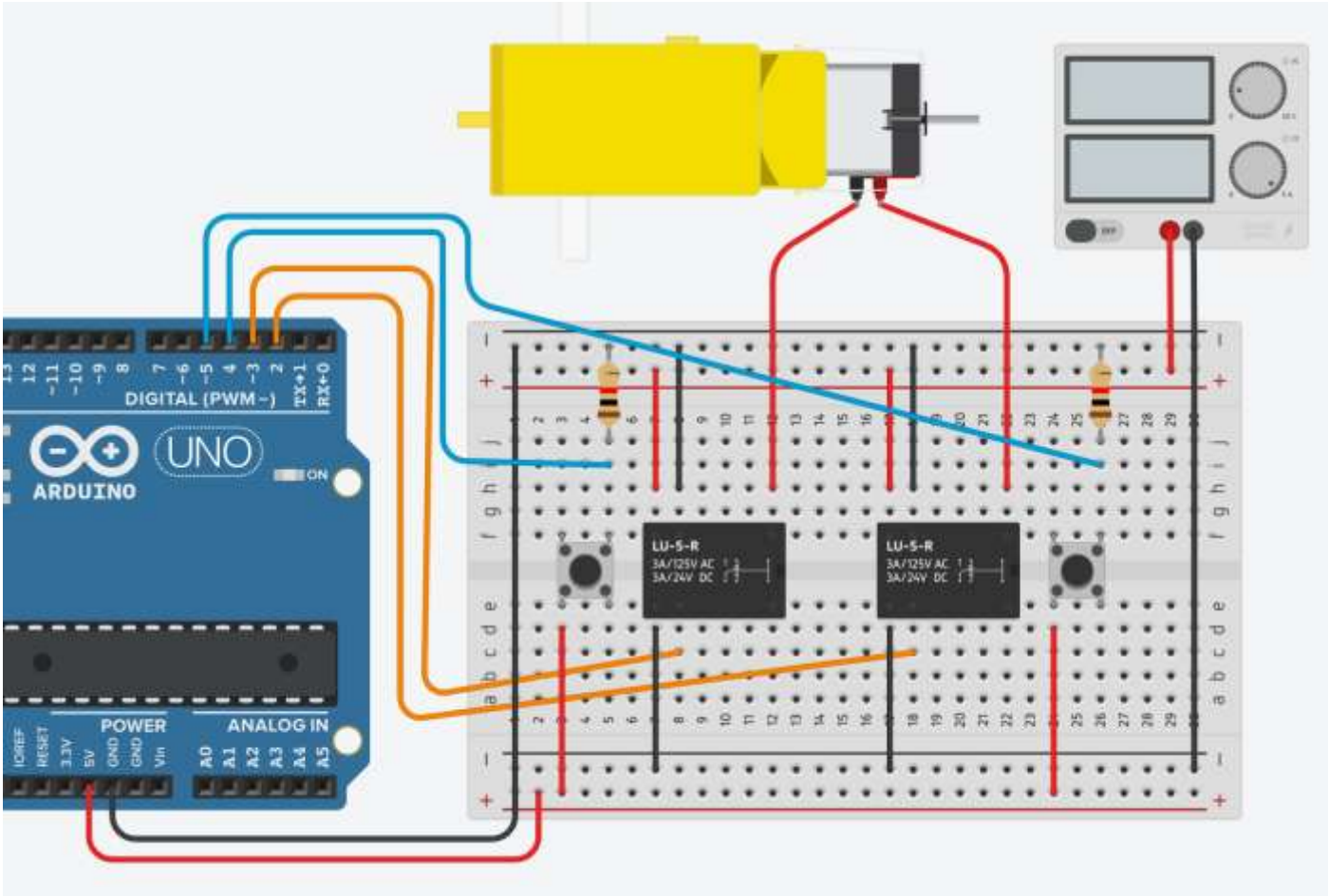
Drive forward:
- Close 1 and 2

Drive backward:
- Close 3 and 4



ESERCIZIO VERSO ROTAZIONE MOTORE CON RELE'

Controllare il verso di rotazione del motore C.C. con due pulsanti utilizzando 2 relè.

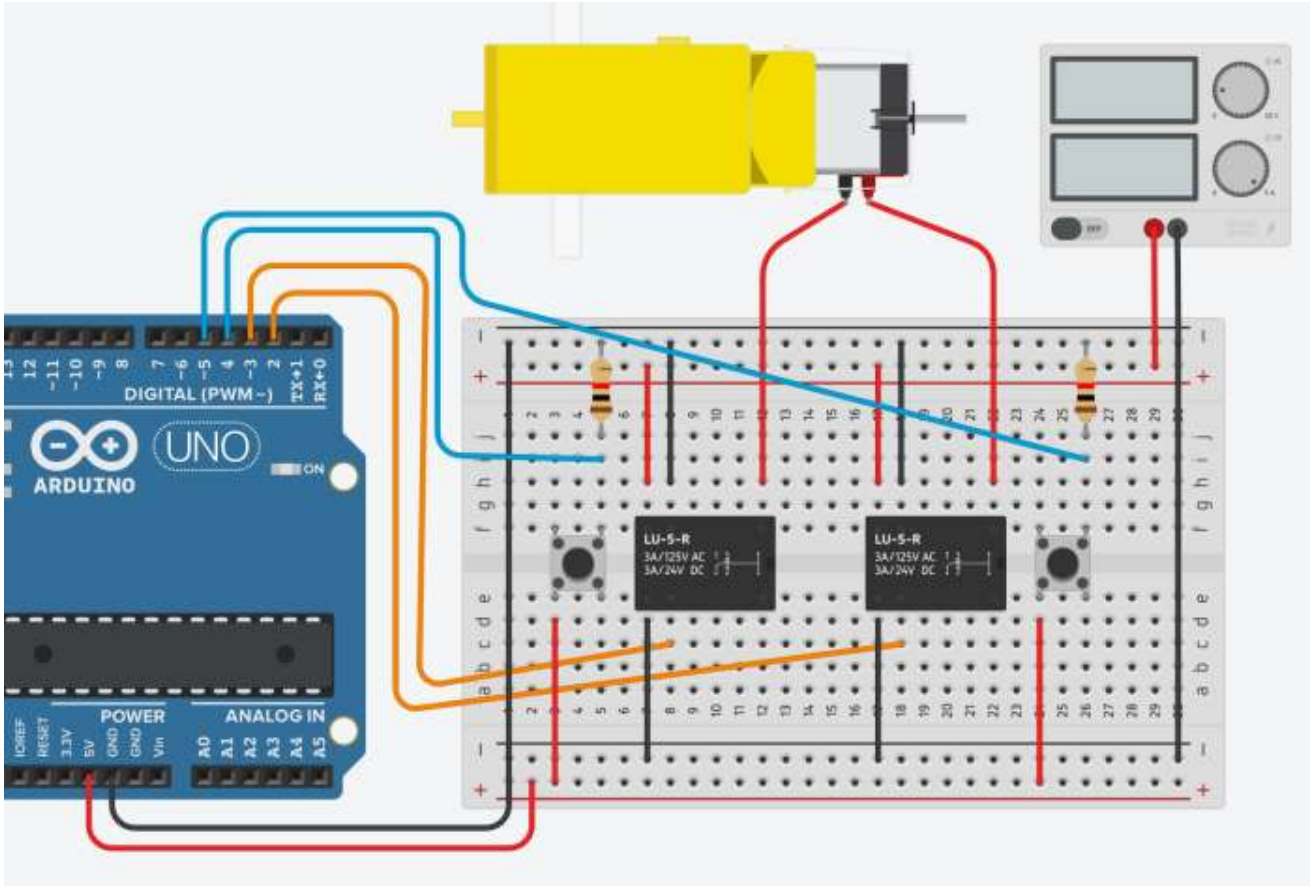


CODICE

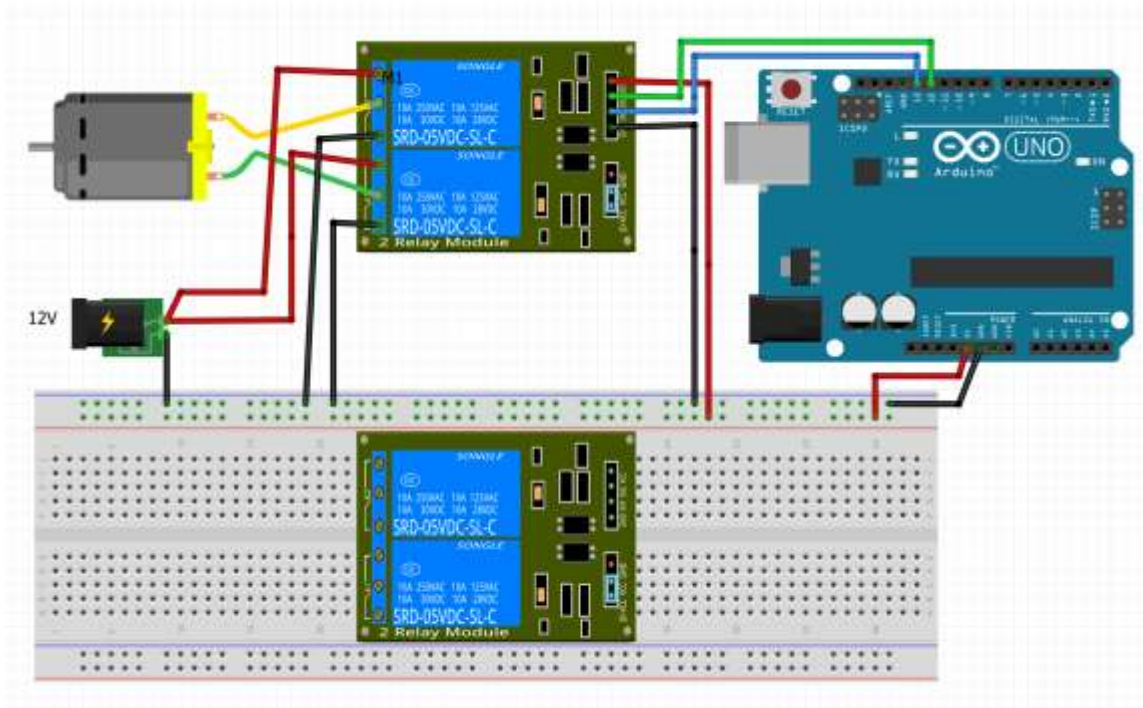
```
int incomingByte = 0; // for incoming serial data

void setup() {
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);

  pinMode(4, INPUT);
  pinMode(5, INPUT);

  Serial.begin(9600);
}

void loop() {

  int statoP1= digitalRead(4);
  if (statoP1== HIGH) {
     Serial.println("M1 ORARIO");
     digitalWrite(2, HIGH);
     digitalWrite(3, LOW);
  }
  else
  {
     Serial.println("M1 STOP");
     digitalWrite(2, LOW);
  }

  int statoP2= digitalRead(5);
  if (statoP2== HIGH) {
     Serial.println("M1 ANTIORARIO");
     digitalWrite(2, LOW);
     digitalWrite(3, HIGH);
  }
  else
  {
     Serial.println("M1 STOP");
     digitalWrite(3, LOW);
  }

  delay(100);
}
```

ESERCIZIO VERSO ROTAZIONE MOTORE C.C CON RELE' + COMANDI SERIALE

Impostare il verso di rotazione del motore DC attraverso comandi inviati dal monitor seriale.
1 → rotazione oraria
2 → rotazione antioraria
3-4 → stop



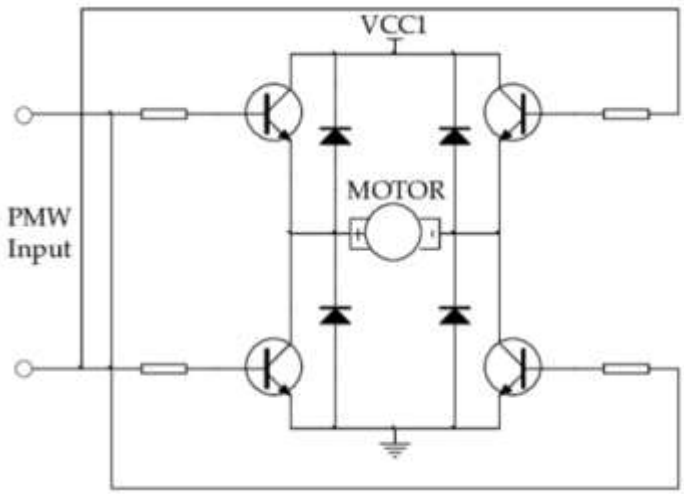
Per Arduino sono disponibili delle schede (shield) con 2-4-8-16 relè che permettono di semplificare il circuito.



CODICE

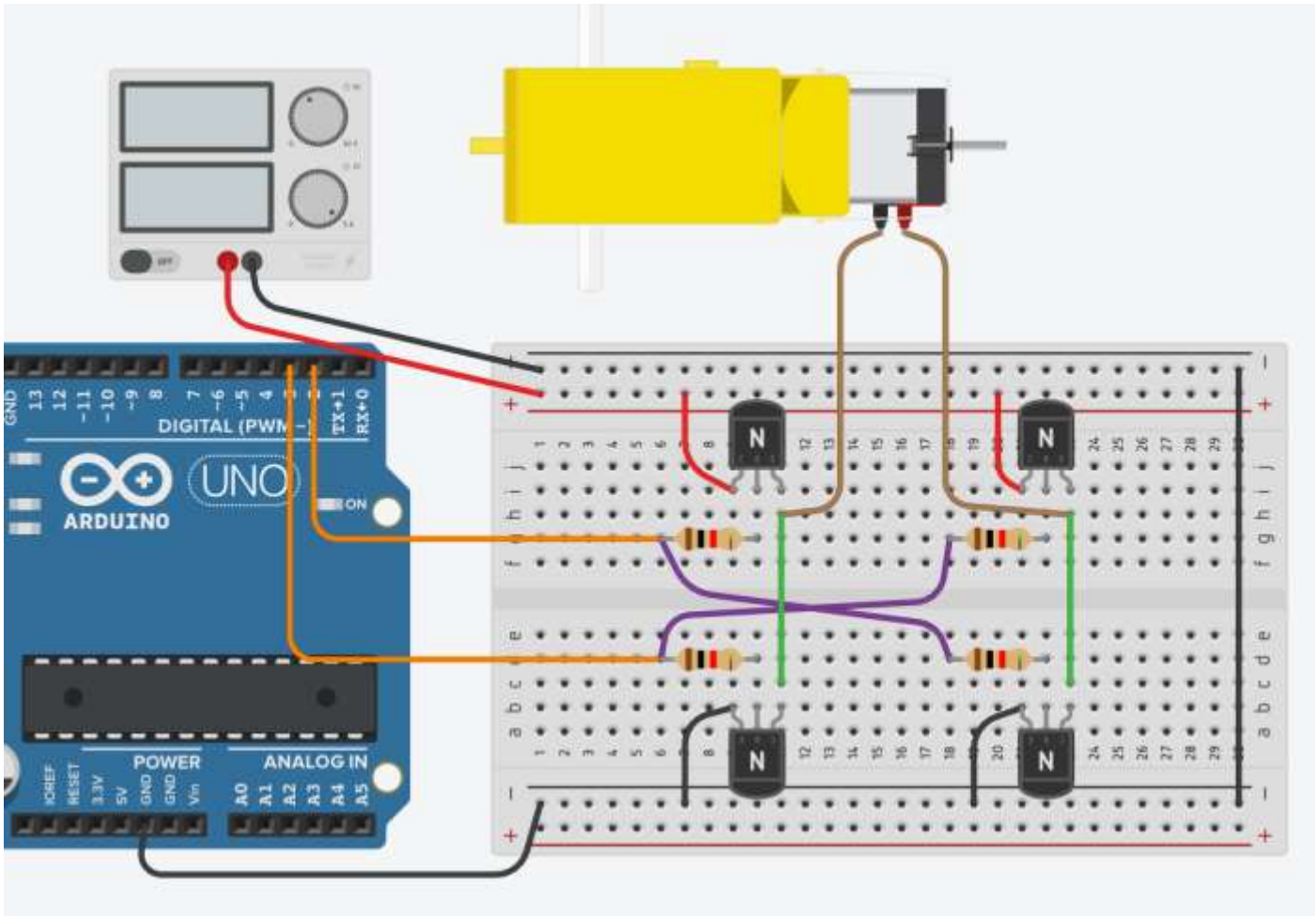
```
int incomingByte = 0; // for incoming serial data

void setup() {
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);

  pinMode(4, INPUT);
  pinMode(5, INPUT);

  Serial.begin(9600);
}

void loop() {

  //SERIALE leggo numeri da 0-9 (1 cifra)
  if (Serial.available() > 0) {
    incomingByte = Serial.parseInt();
    Serial.println(incomingByte);

    if (incomingByte==1) {
      Serial.println("M1 ORARIO");
      digitalWrite(2, HIGH);
      digitalWrite(3, LOW);
    }
    else if (incomingByte==2) {
      Serial.println("M1 ANTIORARIO");
      digitalWrite(2, LOW);
      digitalWrite(3, HIGH);
    }
    else if (incomingByte==3) {
      Serial.println("STOP");
      digitalWrite(2, HIGH);
      digitalWrite(3, HIGH);
    }
    else if (incomingByte==4) {
      Serial.println("STOP");
      digitalWrite(2, LOW);
      digitalWrite(3, LOW);
    }
    else
    {
      Serial.println("non valido");
    }
  }

  delay(100);
}
```

GESTIONE VERSO DI ROTAZIONE MOTORE C.C. CON 4 BJT



Per ottenere il duplice effetto di regolare il verso di rotazione e la velocità di rotazione del motore è necessario un ponte ad H costituito da 4 transistor.
La soluzione proposta impiega 4 transistor di tipo N.

I diodi in parallelo ai transistor servono da protezione contro le corrente inverse generate dal motore all'avvio e allo spegnimento.



ESERCIZIO VERSO ROTAZIONE MOTORE C.C CON BJT + COMANDI SERIALE

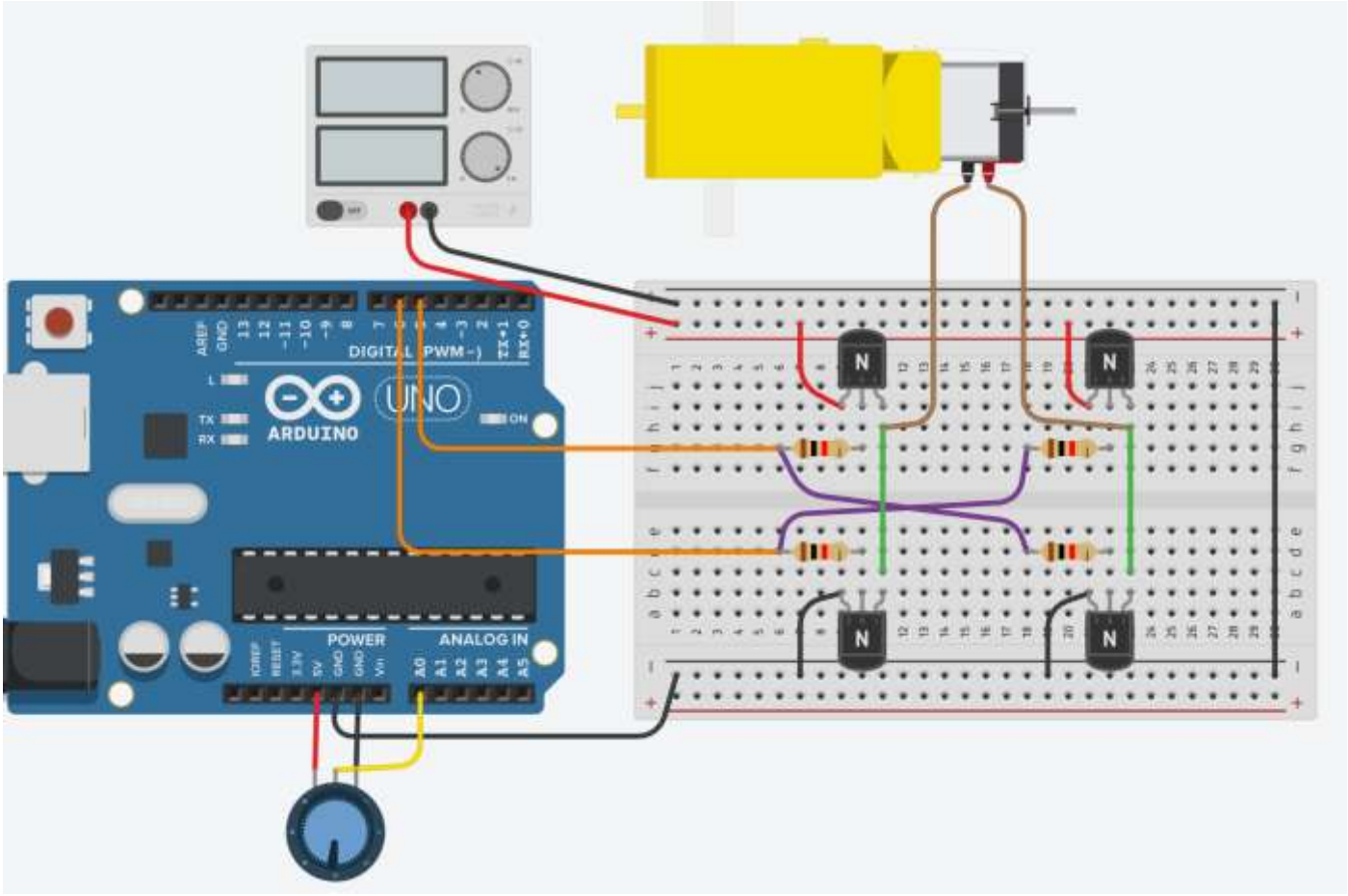
Impostare il verso di rotazione del motore DC attraverso comandi inviati dal monitor seriale.
1 → rotazione oraria
2 → rotazione antioraria
3-4 → stop

CODICE

```
int pinBJT1= 2;
int pinBJT2= 3;
int speed=255;
int incomingByte = 0; // for incoming serial data

void setup() {
  pinMode(pinBJT1, OUTPUT);
  pinMode(pinBJT2, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  //SERIALE leggo numeri da 0-9 (1 cifra) associati
  if (Serial.available() > 0) {
    incomingByte = Serial.parseInt();
    Serial.println(incomingByte);

    if (incomingByte==1) {
      Serial.println("M1 ORARIO");
      digitalWrite(pinBJT1,HIGH);
      digitalWrite(pinBJT2,0);
    }
    else if (incomingByte==2) {
      Serial.println("M1 ANTIORARIO");
      digitalWrite(pinBJT1,0);
      digitalWrite(pinBJT2,HIGH);
    }
    else if (incomingByte==3) {
      Serial.println("STOP");
      digitalWrite(pinBJT1, HIGH);
      digitalWrite(pinBJT2, HIGH);
    }
    else if (incomingByte==4) {
      Serial.println("STOP");
      digitalWrite(pinBJT1, 0);
      digitalWrite(pinBJT2, 0);
    }
    else
    {
      Serial.println("Non valido!");
    }

  }

  delay(100);
}
```

ESERCIZIO VERSO ROTAZIONE MOTORE CON BJT + COMANDI SERIALE + VELOCITA'

Impostare il verso di rotazione del motore DC attraverso comandi inviati dal monitor seriale e la velocità tramite un potenziometro.

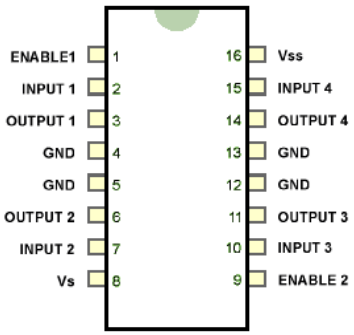


CODICE

```
int pinBJT1= 5;
int pinBJT2= 6;
int pinPotenziometro=A0;
int speed=255;
int incomingByte = 0; // for incoming serial data

void setup() {
  pinMode(pinBJT1, OUTPUT);
  pinMode(pinBJT2, OUTPUT);
  pinMode(pinPotenziometro, INPUT);
  Serial.begin(9600);
}

void loop() {
  speed = analogRead(pinPotenziometro)/4; //1024--> 256)

  //SERIALE leggo numeri da 0-9 (1 cifra) associati
  if (Serial.available() > 0) {
    incomingByte = Serial.parseInt();
    Serial.println(incomingByte);

    if (incomingByte==1) {
      Serial.println("M1 ORARIO");
      analogWrite(pinBJT1,speed);
      analogWrite(pinBJT2,0);
    }
    else if (incomingByte==2) {
      Serial.println("M1 ANTIORARIO");
      analogWrite(pinBJT1,0);
      analogWrite(pinBJT2,speed);
    }
    else if (incomingByte==3) {
      Serial.println("STOP");
      analogWrite(pinBJT1, 255);
      analogWrite(pinBJT2, 255);
    }
    else if (incomingByte==4) {
      Serial.println("STOP");
      analogWrite(pinBJT1, 0);
      analogWrite(pinBJT2, 0);
    }
    else
    {
      Serial.println("Non valido!");
    }
    Serial.print("v= "); Serial.println(speed);
  }

  delay(100);
}
```

GESTIONE VELOCITA' E VERSO DI ROTAZIONE MOTORE CC CON DRIVER PONTE AD H LD293D

Il ponte ad H L293D permette ad una scheda Arduino di pilotare 2 motori DC, controllando sia la velocità che la direzione di ciascun canale in modo indipendente.
E' possibile utilizzare motori CC con alimentazioni da 4,5 a 36 Volt e una corrente massima di 600mA per canale (1,2A di picco).



L293D
L293DD

PUSH-PULL FOUR CHANNEL DRIVER WITH DIODES

- 600mA OUTPUT CURRENT CAPABILITY PER CHANNEL
- 1.2A PEAK OUTPUT CURRENT (non repetitive) PER CHANNEL
- ENABLE FACILITY
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)
- INTERNAL CLAMP DIODES

SO(12+4+4) Powerdip (12+2+2)

ORDERING NUMBERS:

L293DD L293D

DESCRIPTION

The Device is a monolithic integrated high voltage, high current four channel driver designed to accept standard DTL or TTL logic levels and drive inductive loads (such as relays solenoides, DC and stepping motors) and switching power transistors.

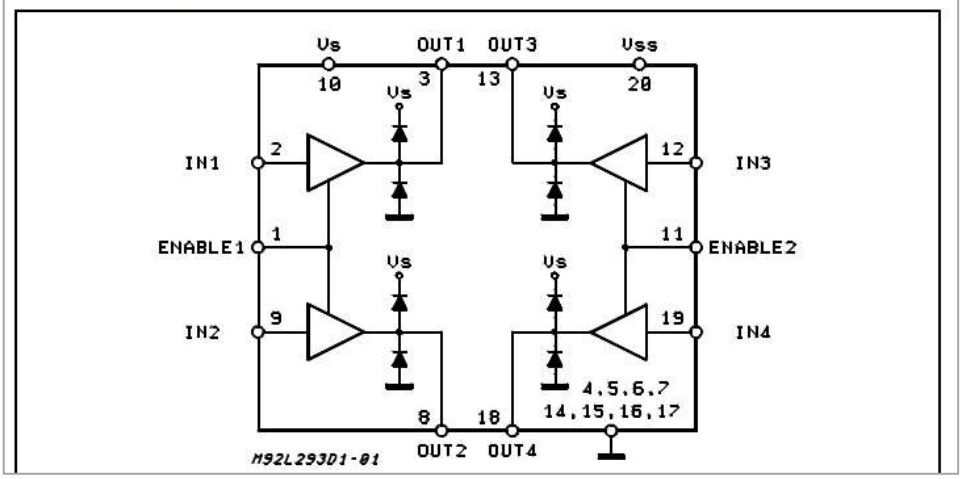
To simplify use as two bridges each pair of channels is equipped with an enable input. A separate supply input is provided for the logic, allowing operation at a lower voltage and internal clamp diodes are included.

This device is suitable for use in switching applications at frequencies up to 5 kHz.

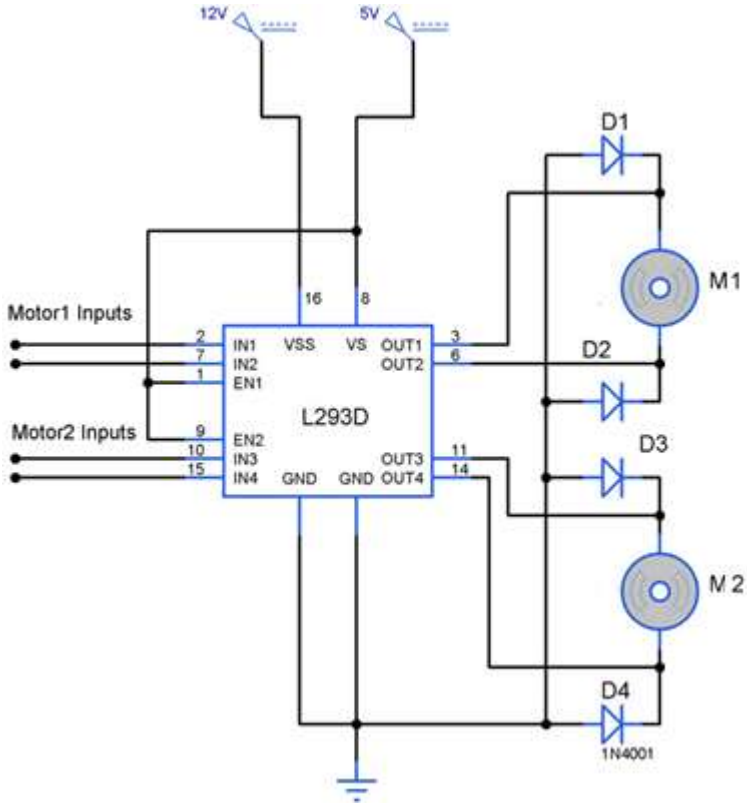
The L293D is assembled in a 16 lead plastic packaage which has 4 center pins connected together and used for heatsinking

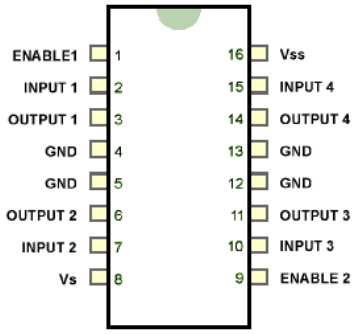
The L293DD is assembled in a 20 lead surface mount which has 8 center pins connected together and used for heatsinking.

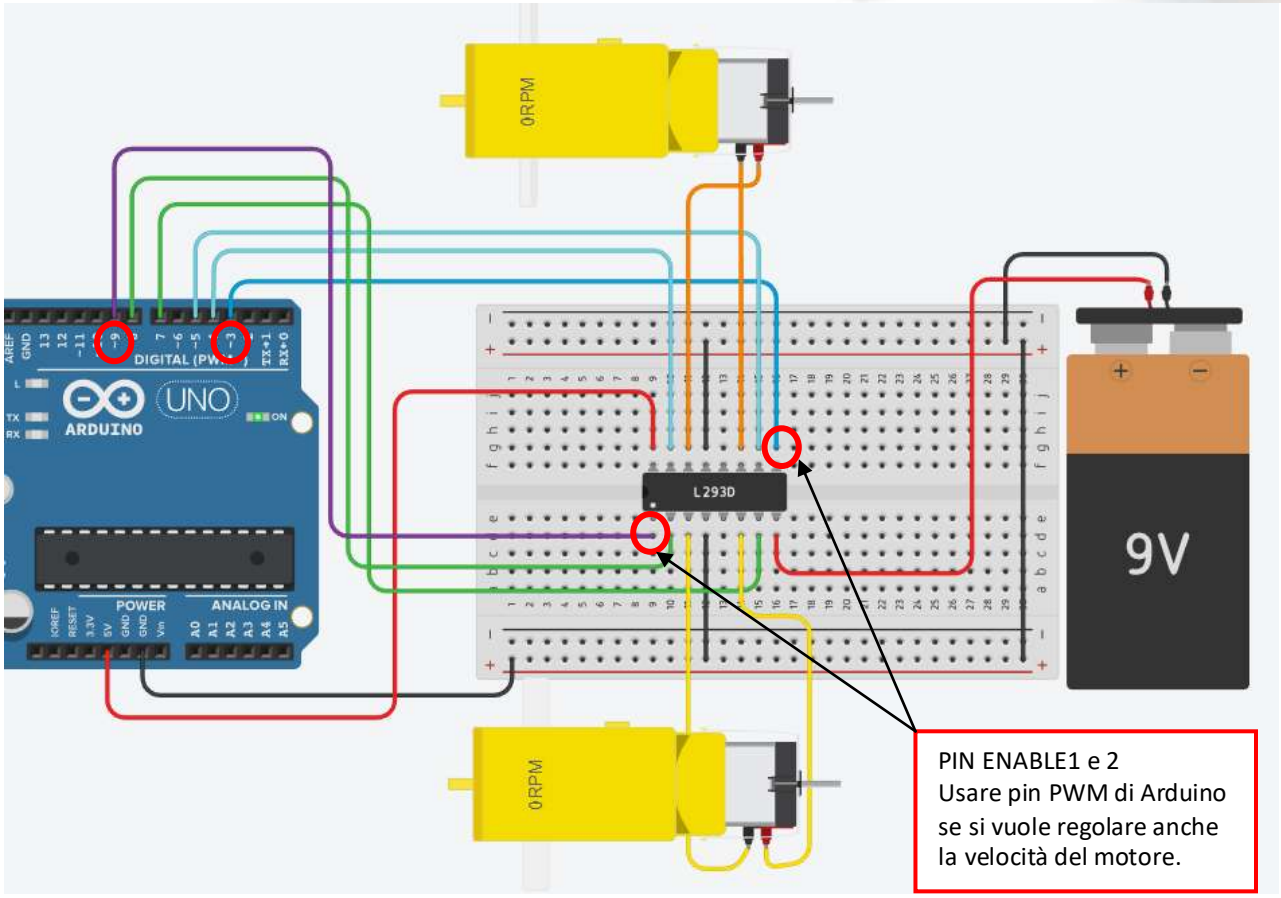
BLOCK DIAGRAM



SCHEMA CONTROLLO DUE MOTORI C.C. DRONE CON L293D



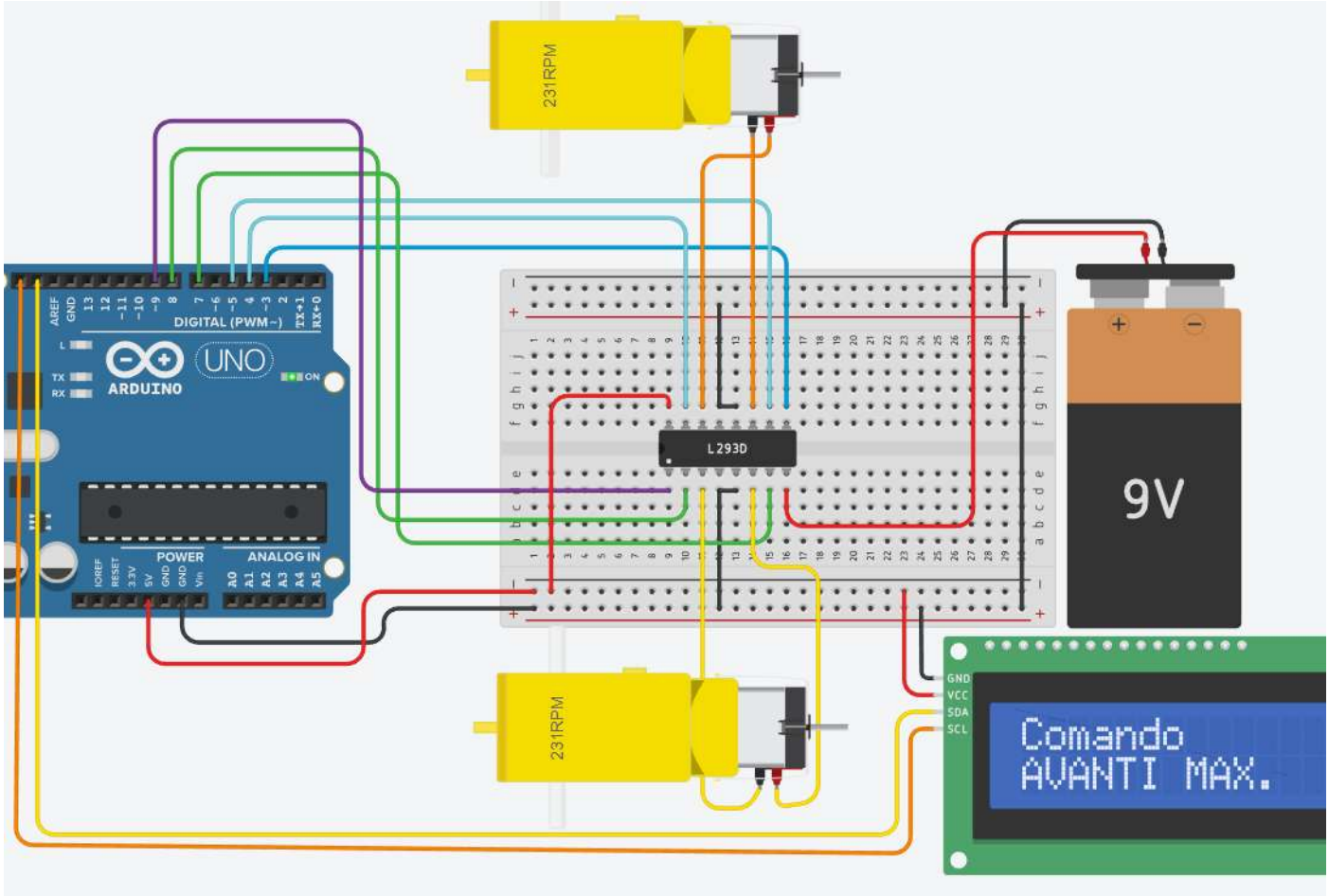




Codice Arduino

```
//DRIVER L293D
// Motore A
int enB = 3;  // PWM per regolare velocità
int in3 = 5;
int in4 = 4;

// Motore B
int enA = 9;  // PWM per regolare velocità
int in1 = 8;
int in2 = 7;

void setup() {
		// Pin per driver
		pinMode(enA, OUTPUT);
		pinMode(enB, OUTPUT);
		pinMode(in1, OUTPUT);
		pinMode(in2, OUTPUT);
		pinMode(in3, OUTPUT);
		pinMode(in4, OUTPUT);

		// Spengo motori
		 ImpostaVelocita(0);
}

void loop() {

   // Avanti a massima velocità
   Avanti(100); delay(4000);

   // Indietro a massima velocità
   Indietro(100); delay(4000);

   // Fermo motori
   ImpostaVelocita(0); delay(4000);

   // Avanti a 50% velocità
   Avanti(50); delay(4000);

   // Indietro a 50% velocità
   Indietro(50); delay(4000);

		// Fermo motori
   ImpostaVelocita(0); delay(4000);
}

void Avanti(int v) {
   ImpostaVelocita(v);
   digitalWrite(in1, LOW); digitalWrite(in2, HIGH);
   digitalWrite(in3, LOW); digitalWrite(in4, HIGH);
}

void Indietro(int v) {
   ImpostaVelocita(v);
   digitalWrite(in1, HIGH); digitalWrite(in2, LOW);
   digitalWrite(in3, HIGH); digitalWrite(in4, LOW);
}

void Destra(int v) {
   ImpostaVelocita(v);
}

void Sinistra(int v) {
   ImpostaVelocita(v);
}

// v--> 0-100%
void ImpostaVelocita(int v) {
 if (v==0) {
		// Turn off motors
		digitalWrite(in1, LOW);   digitalWrite(in2, LOW);   digitalWrite(in3, LOW);   digitalWrite(in4, LOW);
 }
 else {
		analogWrite(enA, map(v,0,100,0,255));  // da  0-100% a 0-255
		analogWrite(enB, map(v,0,100,0,255));
 }
}
```

SCHEMA CONTROLLO DUE MOTORI C.C. DRONE CON L293D + LCD 16X2 I2C



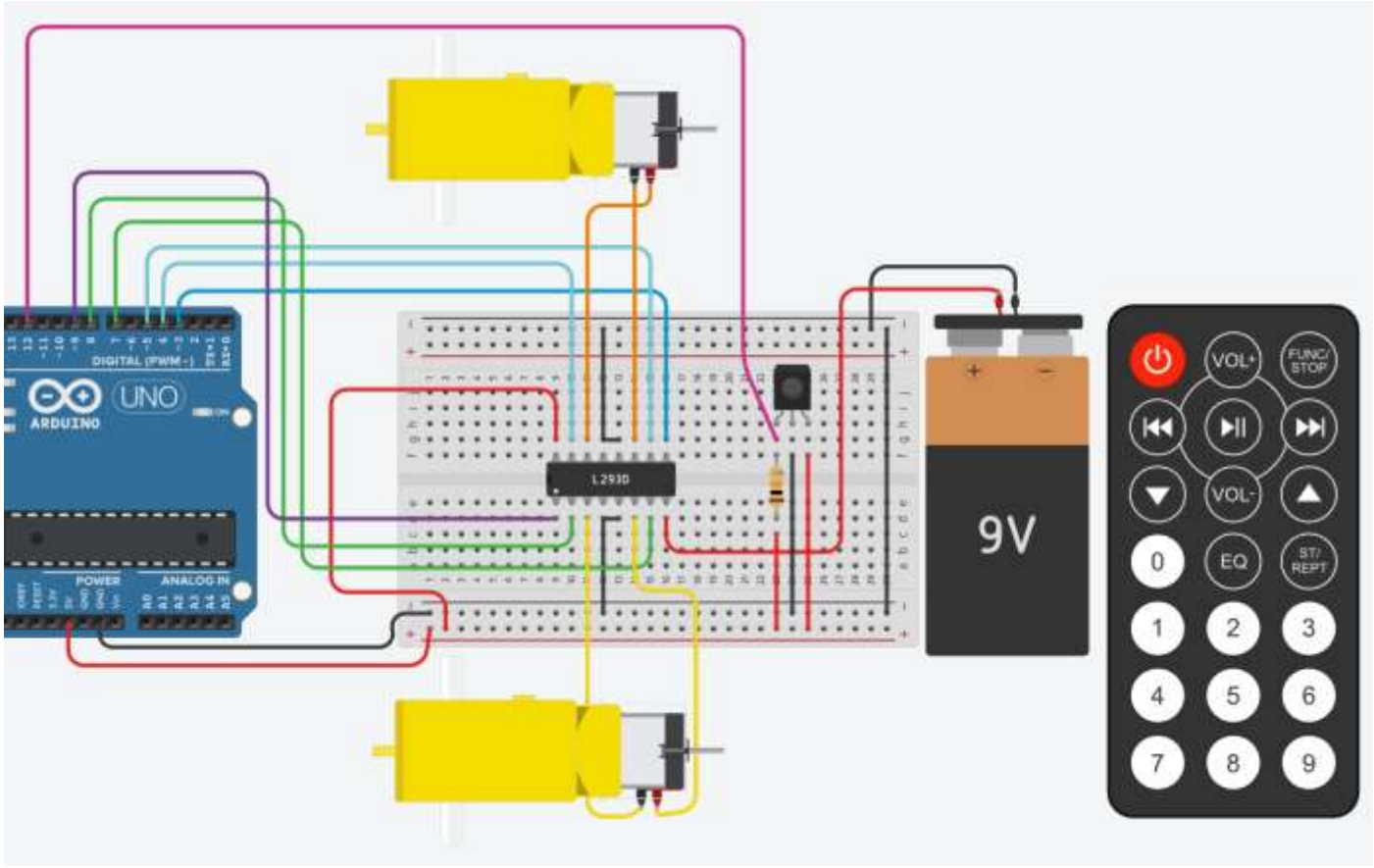
Codice

```
//DRIVER L293D
#include <Adafruit_LiquidCrystal.h>
Adafruit_LiquidCrystal lcd_1(0);

// Motore A
int enB = 3;  // PWM per regolare velocità
int in3 = 5;
int in4 = 4;

// Motore B
int enA = 9;  // PWM per regolare velocità
int in1 = 8;
int in2 = 7;


void setup() {
  lcd_1.begin(16, 2);
  lcd_1.setCursor(0, 0);
  lcd_1.print("Comando");
  lcd_1.setCursor(0, 1);
  lcd_1.print("...");

 // Pin per driver
 pinMode(enA, OUTPUT);
 pinMode(enB, OUTPUT);
 pinMode(in1, OUTPUT);
 pinMode(in2, OUTPUT);
 pinMode(in3, OUTPUT);
 pinMode(in4, OUTPUT);

 // Spengo motori
  ImpostaVelocita(0);
}
```

```
void loop() {
   // Avanti a massima velocità
   lcd_1.setCursor(0, 1); lcd_1.print("AVANTI MAX.    ");
   Avanti(100); delay(2000);

   // Indietro a massima velocità
   lcd_1.setCursor(0, 1); lcd_1.print("INDIETRO MAX.  ");
   Indietro(100); delay(2000);

   // Fermo motori
   lcd_1.setCursor(0, 1); lcd_1.print("STOP          ");
   ImpostaVelocita(0); delay(2000);

   // Avanti a 50% velocità
   lcd_1.setCursor(0, 1); lcd_1.print("AVANTI 50%     ");
   Avanti(50); delay(2000);

   // Indietro a 50% velocità
   lcd_1.setCursor(0, 1); lcd_1.print("INDIETRO 50%    ");
   Indietro(50); delay(2000);

   // Fermo motori
   lcd_1.setCursor(0, 1); lcd_1.print("STOP          ");
   ImpostaVelocita(0); delay(2000);

   // Avanti a 50% velocità
   lcd_1.setCursor(0, 1); lcd_1.print("AVANTI ACC.    ");
   Avanti2(100,2000); delay(2000);

   // Fermo motori
   lcd_1.setCursor(0, 1); lcd_1.print("STOP          ");
   ImpostaVelocita(0); delay(2000);

   // Avanti a 50% velocità
   lcd_1.setCursor(0, 1); lcd_1.print("INDIETRO ACC.   ");
   Indietro2(100,2000); delay(2000);
}

void Avanti(int v) {
   ImpostaVelocita(v);
   digitalWrite(in1, LOW); digitalWrite(in2, HIGH);
   digitalWrite(in3, LOW); digitalWrite(in4, HIGH);
}

void Indietro(int v) {
   ImpostaVelocita(v);
   digitalWrite(in1, HIGH); digitalWrite(in2, LOW);
  digitalWrite(in3, HIGH); digitalWrite(in4, LOW);
}

void Destra(int v) {
   ImpostaVelocita(v);
}

void Sinistra(int v) {
   ImpostaVelocita(v);
}

// v--> 0-100%
void ImpostaVelocita(int v) {
 if (v==0) {
            // Turn off motors
            digitalWrite(in1, LOW);digitalWrite(in2, LOW);digitalWrite(in3, LOW);digitalWrite(in4, LOW);
 }
 else {  analogWrite(enA, map(v,0,100,0,255));  analogWrite(enB, map(v,0,100,0,255));   }
}
```

```
// Accellera da 0 a v in t sec circa
void Avanti2(int v, int t) {
  int v_pwm= map(v,0,100,0,255);
  int dt= t / v_pwm;  // calcolo la durata della pausa per stare nei secondi indicati
  for (int i=1; i<=v_pwm; i++) {
    analogWrite(enA, i);
    analogWrite(enB, i);
    digitalWrite(in1, LOW); digitalWrite(in2, HIGH);
    digitalWrite(in3, LOW); digitalWrite(in4, HIGH);
    delay(dt);
  }
}

// Accellera da 0 a v in 2 sec circa
void Indietro2(int v, int t) {
  int v_pwm= map(v,0,100,0,255);
  int dt= t / v_pwm;
  for (int i=1; i<=v_pwm; i++) {
    analogWrite(enA, i);
                analogWrite(enB, i);
                digitalWrite(in1, HIGH); digitalWrite(in2, LOW);
                digitalWrite(in3, HIGH); digitalWrite(in4, LOW);
    delay(dt);
  }
}
```

SCHEMA CONTROLLO DUE MOTORI C.C. DRONE CON L293D CON TELECOMANDO INFRAROSSI



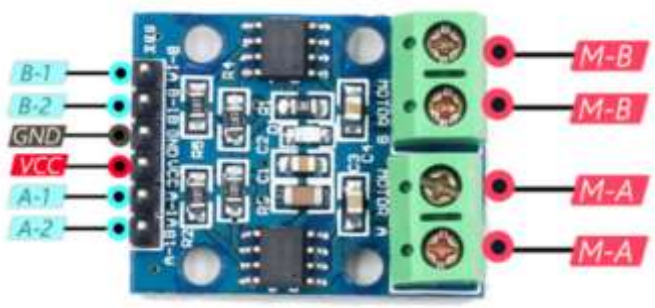

```
//DRIVER L293D

#include <IRremote.h>

// Motore A
int enB = 3;  // PWM per regolare velocità
int in3 = 5;
int in4 = 4;

// Motore B
int enA = 9;  // PWM per regolare velocità
int in1 = 8;
int in2 = 7;

// Infrared receiving pin
int rcvPin = 12;

// COMANDI
long ir_uno = 4010852096;          //
long ir_due = 3994140416;    //
long ir_tre = 3977428736;    //
long ir_quattro = 3944005376;   //
long ir_cinque  = 3927293696;   //
long ir_sei = 3910582016;   //
long ir_sette   = 3877158656;   //
long ir_otto   = 3860446976;   //
long ir_nove   = 3843735296;   //
long ir_play   = 4194680576;

void setup() {
        // Pin per driver
        pinMode(enA, OUTPUT);
        pinMode(enB, OUTPUT);
        pinMode(in1, OUTPUT);
        pinMode(in2, OUTPUT);
        pinMode(in3, OUTPUT);
        pinMode(in4, OUTPUT);

        Serial.begin(9600); Serial.println("Pronto!");
```

```
   // Attibo ricevitore
   IrReceiver.begin(rcvPin, true);

   // Spengo motori
   ImpostaVelocita(0);
}

void loop() {

  // Attendo comando da telecomando IR
  if(IrReceiver.decode()){
   auto value= IrReceiver.decodedIRData.decodedRawData;
   Serial.println(value); // Print out the decoded results

   if (value == ir_due) {
     Serial.println("AVANTI");
     //Avanti(100);
     Avanti2(100,2000);
   }
   else if (value == ir_otto) {
     Serial.println("INDIETRO");
     //Indietro(100);
     Indietro2(100,2000);
   }
   else if (value == ir_quattro) {
     Serial.println("DESTRA");

   }

   else if (value == ir_sei) {
     Serial.println("SINISTRA");

   }

   else if (value == ir_cinque) {
     Serial.println("STOP");
     ImpostaVelocita(0);
   }
   IrReceiver.resume(); // Receive the next value
  }
  delay(100);
}

// v--> 0-100%
void ImpostaVelocita(int v) {
  if (v==0) {
              // Turn off motors
              digitalWrite(in1, LOW);
              digitalWrite(in2, LOW);
              digitalWrite(in3, LOW);
              digitalWrite(in4, LOW);
  }
  else {
              analogWrite(enA, map(v,0,100,0,255));  // da  0-100% a 0-255
              analogWrite(enB, map(v,0,100,0,255));
  }
}

void Avanti(int v) {
   ImpostaVelocita(v);
              digitalWrite(in1, LOW); digitalWrite(in2, HIGH);
              digitalWrite(in3, LOW); digitalWrite(in4, HIGH);
}

void Indietro(int v) {
   ImpostaVelocita(v);
              digitalWrite(in1, HIGH); digitalWrite(in2, LOW);
              digitalWrite(in3, HIGH); digitalWrite(in4, LOW);
}

void Destra(int v) {
   ImpostaVelocita(v);
}
```

```
void Sinistra(int v) {
   ImpostaVelocita(v);
}

// Accellera da 0 a v in 2 sec circa
void Avanti2(int v, int t) {
  int v_pwm= map(v,0,100,0,255);
  int dt= t / v_pwm;
  for (int i=1; i<=v_pwm; i++) {
    analogWrite(enA, i);
                analogWrite(enB, i);
                digitalWrite(in1, LOW); digitalWrite(in2, HIGH);
                digitalWrite(in3, LOW); digitalWrite(in4, HIGH);
    delay(dt);
  }
}

// Accellera da 0 a v in 2 sec circa
void Indietro2(int v, int t) {
  int v_pwm= map(v,0,100,0,255);
  int dt= t / v_pwm;
  for (int i=1; i<=v_pwm; i++) {
    analogWrite(enA, i);
                analogWrite(enB, i);
                digitalWrite(in1, HIGH); digitalWrite(in2, LOW);
                digitalWrite(in3, HIGH); digitalWrite(in4, LOW);
    delay(dt);
  }
}
```

DRIVER L9110S DUAL-CHANNEL H-BRIDGE

IL modulo L9110S è un driver a ponte H doppio canale molto compatto.
Il modulo incorpora due circuiti integrati per consentire il controllo indipendente di un motore passo-passo o di due motori CC.
Accetta una tensione di alimentazione compresa tra 2,5 e 12 V.
La corrente massima per ogni canale dell'azionamento è 800 mA (picco 1.5 ~2.0A).
I pin di intestazione con passo da 0,1" consentono il collegamento diretto a un microcontrollore come un Arduino e richiede solo due pin digitali per il controllo di un motore CC in entrambe le direzioni.
Le morsettiere a vite forniscono un comodo collegamento ai motori.



- Doppio driver del motore con chipset L9110S
- Tensione di ingresso: 2,5 V ~ 12 V
- Corrente: 800 mA
- Può pilotare due motori CC e un motore passo-passo bifase a 4 fili

	AIN1	AIN2
Forward Direction	HIGH	LOW
Reverse Direction	LOW	HIGH
Brake / Stopped	LOW	LOW
Brake / Stopped	HIGH	HIGH

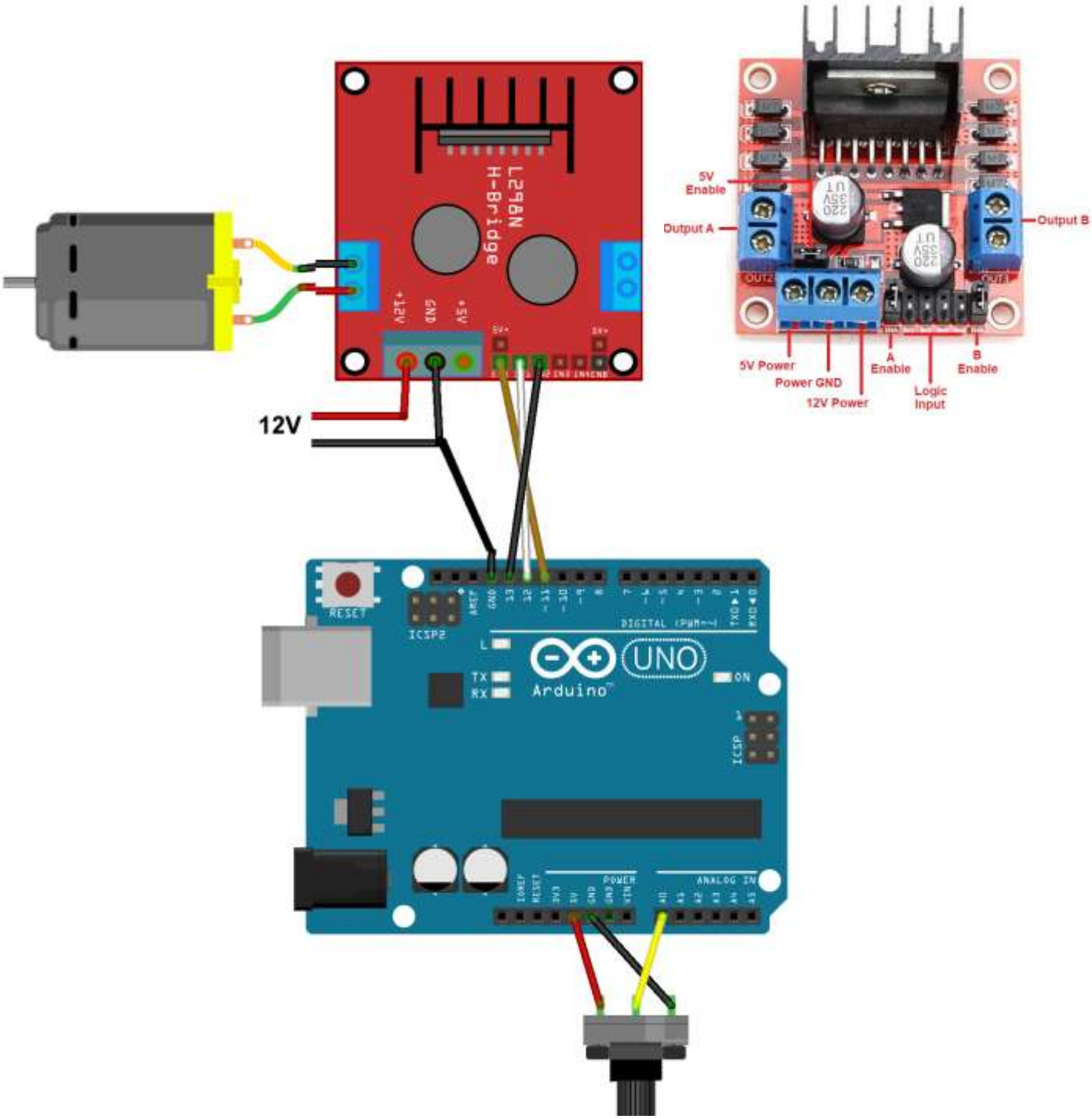
Codice

```
#define A1 5  // Motor A pins
#define A2 6
#define B1 10 // Motor B pins
#define B2 11

int incomingByte = 0; // for incoming serial data
int  input = 0;

void setup() {

  pinMode(A1, OUTPUT);
  pinMode(A2, OUTPUT);
  pinMode(B1, OUTPUT);
  pinMode(B2, OUTPUT);

  digitalWrite(A1, LOW);
  digitalWrite(A2, LOW);
  digitalWrite(B1, LOW);
  digitalWrite(B2, LOW);

  Serial.begin(9600); // opens serial port, sets data rate to 9600 bps

  Serial.println("select direction of movement");
  Serial.println("1.forward");
  Serial.println("2.backward");
  Serial.println("3.stop");

}

void loop() {

  // send data only when you receive data:
  if (Serial.available() > 0) {
    // read the incoming byte:
    incomingByte = Serial.read();
    input = incomingByte - 48;  //convert ASCII code of numbers to 1,2,3
```

```
  switch (input) {
    case 1:
      forward();
      break;
    case 2:
      backward();
      break;
    case 3:
      Stop();
      break;
  }
  delay(200);
  input=0;
}
}

void forward() {        //function of forward
  analogWrite(A1, 255);
  analogWrite(A2, 0);
  analogWrite(B1, 255);
  analogWrite(B2, 0);
}

void backward() {       //function of backward
  analogWrite(A1, 0);
  analogWrite(A2, 210);
  analogWrite(B1, 0);
  analogWrite(B2, 210);
}

void Stop() {           //function of stop
  digitalWrite(A1, LOW);
  digitalWrite(A2, LOW);
  digitalWrite(B1, LOW);
  digitalWrite(B2, LOW);
}
```

DRIVER L298N H-BRIDGE

Questa scheda di controllo per motori è basata sul driver Dual H-Bridge L298N e permette di pilotare due motori C.C. oppure un motore passo-passo bipolare con tensione operativa compresa nel range tra 5V e 35V e una corrente massima di 2A, controllandone la velocità e la direzione.



NOTA BENE:

Per motori a bassa resistenza interna (tipica degli stepper) è necessario un driver di corrente e non un driver di tensione come l'L298N. I motori a bassa impedenza in generale vanno sono controllati in corrente e non in tensione.

Per valori di resistenza degli avvolgimenti del motore oltre 30-60 ohm un L298N funziona senza bruciarsi, ma la velocità massima è inferiore rispetto a quella ottenibile con un driver di corrente.

CODICE

```
//L298N pilotare un motore DC con Arduino

//definizione dei pin
static int pinPotenziometro = A0; //pin analogico per valori del potenziometro
static int mA = 12; //pin digitale per gli stati logici da inviare al modulo
static int mB = 13; //pin digitale per gli stati logici da inviare al modulo
static int pinMotore = 11; //pin PWM per variare velocità motore

//variabili
int potenziometro;  //valore letto dal potenziometro sul pin A0
int velocita;       //valore PWM in uscita dal pin 11

void setup() {
  Serial.begin(9600);

  //inizializzo variabili
  potenziometro = 0;
  velocita = 0;

  //definisco tipologia pin
  pinMode(pinPotenziometro, INPUT);  //input da potenziometro per la velocità
  pinMode(mA, OUTPUT); //output per lo stato logico del pin IN1 del modulo L298N
  pinMode(mB, OUTPUT); //output per lo stato logico del pin IN2 del modulo L298N
  pinMode(pinMotore, OUTPUT);  //output PWM per il pin EN1 del modulo L298N


  //Imposto verso di rotazione del motore
  /*
    mA |   mB  | Evento
  -----|-------|----------------------
  LOW  | LOW   | fermo
  LOW  | HIGH  | rotazione oraria
  HIGH | LOW   | rotazione antioraria
  HIGH | HIGH  | Fermo
  */

  digitalWrite(mA, LOW);
  digitalWrite(mB, HIGH);
}


void loop() {

  //leggo il valore analogico del potenziometro sul pin A0 (0-1023.
  potenziometro = analogRead(pinPotenziometro);

  // Il range dei valori PWM  e' da 0 a 255
  velocita = map(potenziometro, 0, 1023, 0, 255);

  Serial.print("velocita = ");
  Serial.print(velocita);

  analogWrite(pinMotore, velocita);
}
```

LA POTENZA NEI MOTORI CC

Ci sono due tipi di potenza quando si parla di motori CC:

Potenza elettrica (W o W in): è la quantità di potenza (tensione fornita e corrente assorbita) fornita al motore.

Potenza meccanica (W o W out): è la quantità di lavoro (velocità x coppia) che il motore produce.

La potenza meccanica può essere calcolata utilizzando la velocità e la coppia:

$$Mechanical\ Power\ (W) = Speed\ (RPM) * Torque\ (Nm) * 2 * Pi / 60$$

Un motore produce la massima potenza a metà della coppia di stallo o metà della velocità libera.
Questo punto è chiamato potenza di picco.
Sulle curve del motore è preferibile rimanere sul lato destro della potenza di picco.
Una volta superata la potenza di picco, il motore assorbe più energia elettrica per produrre meno potenza meccanica !

Quando si progetta un azionamento meccanico per eseguire un'azione è fondamentale anche sapere in che lasso di tempo viene eseguito il lavoro richiesto.
Conoscere solo la coppia del motore selezionato garantisce che il motore è in grado di svolgere quell'attività, ma non dice nulla su quanto velocemente può essere eseguita l'attività.

Esempio 1:

Hai un ascensore azionato da un tamburo da 1,25" e il carico di coppia è di 20 Nm.
Decidi di utilizzare un motore che produce 0,2 Nm di coppia a 3.000 giri/min (62,8 W).

Per produrre una coppia sufficiente, dovrai ridurre di 100:1 il motore che riduce la velocità fino a 30 giri / min.
Su un tamburo da 1,25", il sollevatore si sposterà a circa 0,16 pollici/sec.

Esempio 2:

Hai la stessa alzata nell'esempio 1.
Questa volta decidi di utilizzare un motore che produce 0,2 Nm di coppia a 10.000 giri/min (209,4 W).

Per produrre una coppia sufficiente, dovrai comunque utilizzare una riduzione di 100:1 sul motore.
Questo riduce la velocità fino a 100 giri/min. Su un tamburo da 1,25", il sollevatore si sposterà a 0,54 pollici/sec.

Come puoi vedere, anche se entrambi i motori producono la stessa quantità di coppia, il motore utilizzato nell'Esempio 2 ha una potenza oltre 3 volte superiore perché è più veloce.
Pertanto, l'ascensore si sposterà 3 volte più veloce dell'ascensore nell'Esempio 1.

Poiché ogni azione di un meccanismo può essere descritta come un carico (coppia) spostato in una quantità di tempo (velocità), puoi facilmente capire quanta potenza è necessaria per ottenere quell'azione.
La quantità di potenza non è influenzata da alcuna riduzione. Questo perché una riduzione riduce la velocità e aumenta proporzionalmente la coppia.

UTILIZZO DI UNA CURVA MOTORE CC

Le curve del motore vengono utilizzate principalmente in due scenari:

- determinare quale motore (e riduttore) utilizzare in una particolare applicazione
- apprendere di più sullo stato di un motore attualmente in funzione in un sistema.

DETERMINAZIONE DI QUALE MOTORE (E RIDUTTORE) UTILIZZARE

Consideriamo un pezzo che pesa 40N, sollevato da un braccio lungo 0.5 m, che scorre attraverso un cambio 100:1 :

$$Torque\ @\ Arm = Force \times Distance$$
$$Torque\ @\ Arm = 40.0\ N \times 0.5\ m$$
$$Torque\ @\ Arm = 20\ N \cdot m$$

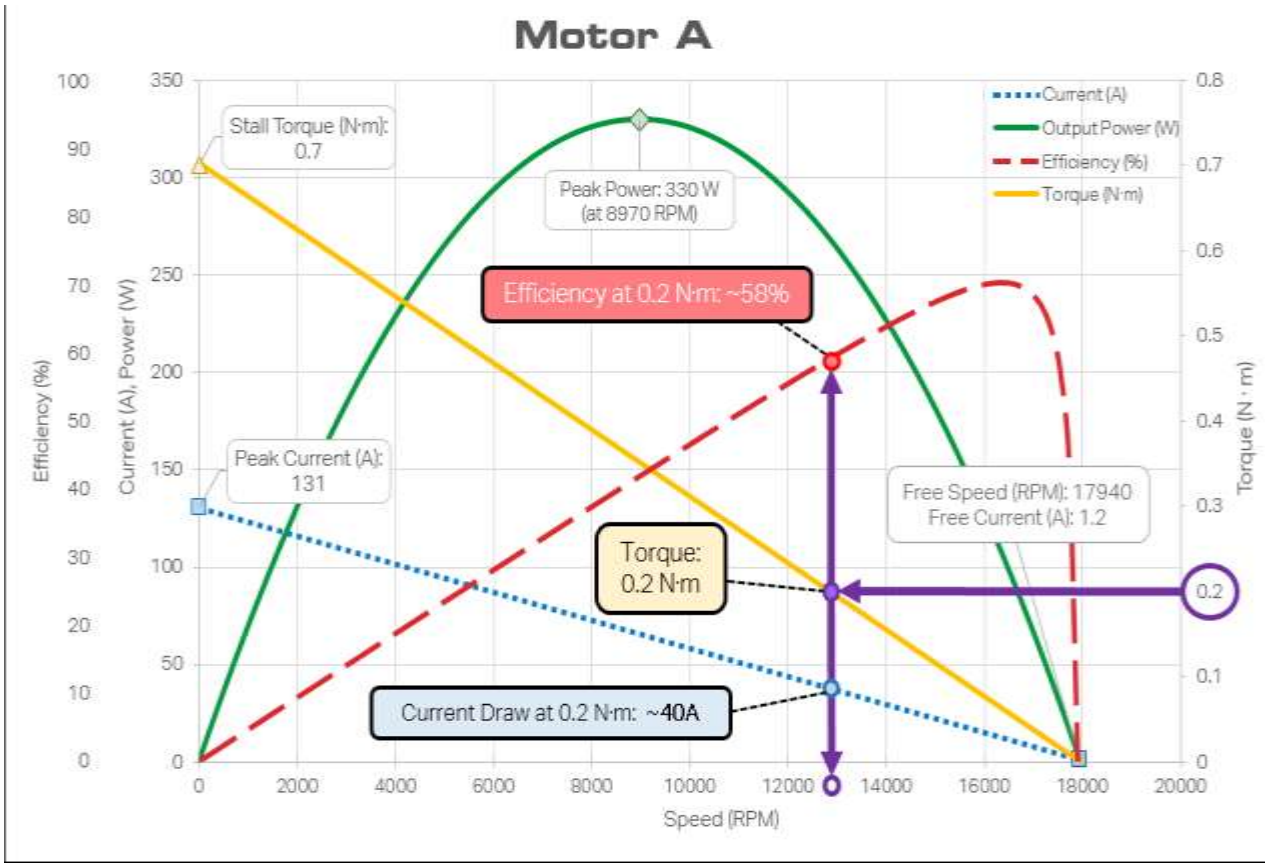
$$Torque\ @\ Motor = (Torque\ @\ Arm) \div (Gear\ Reduction)$$

$$Torque\ @\ Motor = \frac{20\ N \cdot m}{100}$$

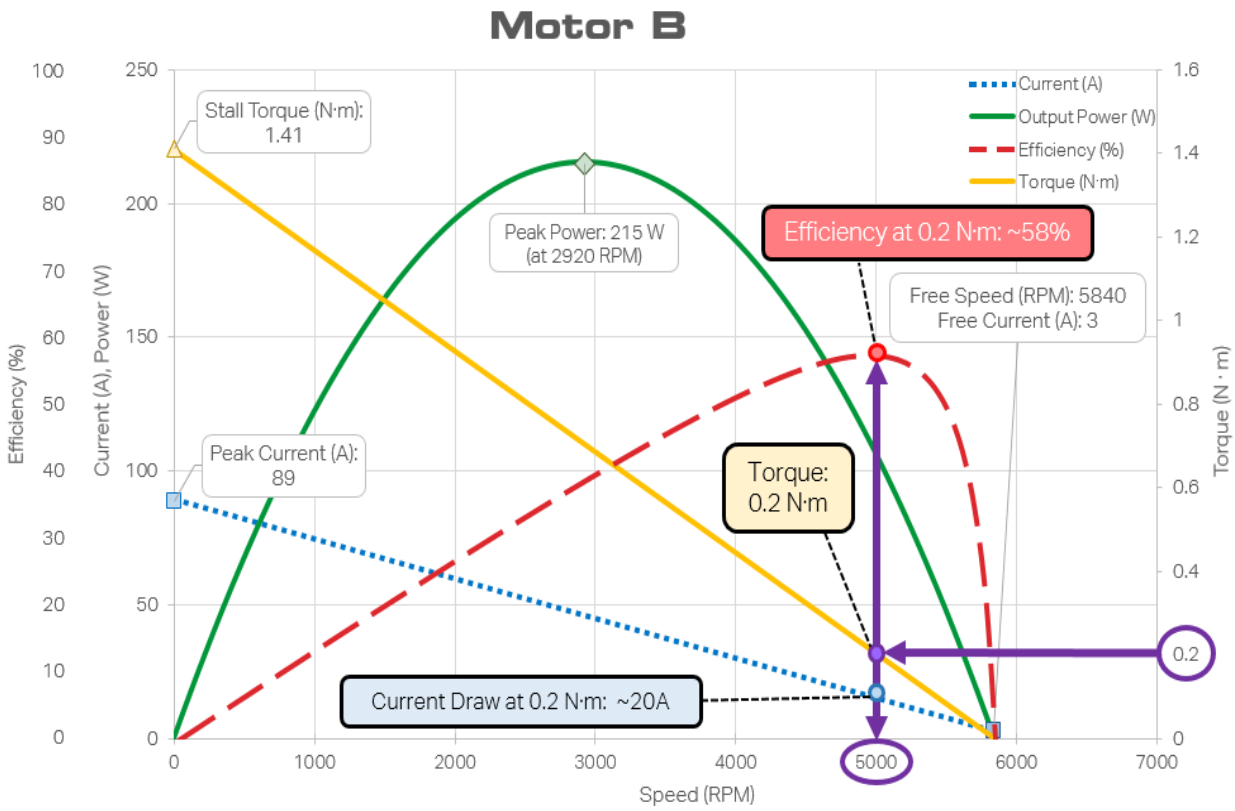
$$Torque\ @\ Motor = 0.2\ N \cdot m$$

Il motore che aziona questo braccio dovrà produrre una coppia motrice di 0,2 N · m .

Questo requisito di coppia in uscita può quindi essere confrontato con le curve motore pubblicate per saperne di più sullo stato del motore durante questa azione.



Il motore A raggiunge 0,2 N · m di coppia a circa 13000 giri/min, assorbendo circa 40 A.
Con un cambio 100:1, ciò equivale a una velocità del braccio di 130 giri/min.
A questa coppia, il motore funziona con un'efficienza di poco inferiore al 60%, al di sotto del suo picco.



Il motore B può raggiungere 0,2 N · m di coppia a circa 5000 giri/min, assorbendo circa 20 A.
Con un cambio 100:1, ciò equivale a una velocità del braccio di 50 giri/min.
A questa coppia, il motore funziona a circa il 60% di efficienza, molto vicino al suo picco.

Utilizzando questi tipi di calcoli, un tecnico può combinare queste informazioni con altri dettagli del sistema per determina re il miglior motore per la propria applicazione:
-50 giri sono troppo lenti?
-40 A sono troppo alti per un assorbimento di corrente?
-è necessaria una maggiore efficienza per motivi termici o di batteria?

E' evidente che ci sono una serie di variabili coinvolte nel fare questa determinazione:
rapporto di trasmissione, lunghezza del braccio e persino peso del pezzo di gioco.

Nota: assorbimento di corrente al <u>picco di potenza</u>

Per una scelta più rapida, noti il tempo necessario a svolgere un'azione, si può calcolare semplicemente la quantità di lavoro eseguita (Lavoro = Massa × Gravità × Altezza) in un determinato periodo di tempo (Potenza = Lavoro / Tempo) e selezionano un motore corrispondente quel fabbisogno di potenza.

Ad esempio, se un meccanismo deve sollevare un oggetto di 20kg a 1m di altezza in 1 secondo:

$$Power = \frac{Work}{Time}$$

$$Power = \frac{Mass \times Gravity \times Height}{Time}$$

$$Power = \frac{20\,kg \times 9.8\,{}^{m}/_{s^2} \times 1.0\,m}{1.0\,s}$$

$$Power = 196\,W$$

Nell'esempio sopra, il motore B è la scelta migliore per un fabbisogno di potenza di 196 W (215W di picco contro i 330W del motore A).

Tuttavia, cosa succede quando ci sono due motori che corrispondono a quella richiesta di potenza di picco?

In genere è meglio scegliere il motore con l'assorbimento di corrente più basso, poiché ciò prolungherà la durata della batteria e ridurrà lo sforzo sull'impianto elettrico. Ciò diventa particolarmente importante sotto carico sostenuto, quando è necessario tenere conto dei limitatori di corrente o degli interruttori automatici.

APPROFONDIRE LO STATO DI UN MOTORE ATTUALMENTE IN FUNZIONE IN UN SISTEMA

"Perché un motore si brucia dopo aver sollevato l'oggetto?"

Se un motore è già stato installato e un ingegnere vuole saperne di più sullo stato del sistema, la stessa teoria di cui sopra può essere invertita. A una tensione nota, è necessario un valore misurato (come l'assorbimento di corrente) per determinare il resto degli attributi del motore in quel momento.

Ad esempio, si consideri ancora il motore A.
Se si utilizza un amperometro per misurare un assorbimento di corrente di 25 A, ora è noto che il motore sta esercitando circa 0,11 N · m di coppia e sta funzionando intorno al suo picco di efficienza di poco inferiore a 70 %. Tuttavia, se l'amperometro sta leggendo 140 A, il motore sta attualmente funzionando in una condizione di stallo estremo.

Le curve dei motore CC spesso sono realizzate a 12 V.
Le quattro caratteristiche chiave (velocità libera/corrente, coppia di stallo/corrente) si adattano approssimativamente in modo proporzionale alla tensione del sistema.
Se il motore A funzionasse a 6 V, la sua corrente di stallo scenderebbe da 130 A a 65 A e la sua coppia di stallo scenderebbe da 0,7 N · m a 0,35 N · m. Se una lettura dell'amperometro mostra un assorbimento di corrente di 25 A:

$$\frac{Stall\ Current}{Measured\ Current} = \frac{65\ A}{25\ A} = 2.6$$

$$Output\ Torque = \frac{Stall\ Torque}{2.6} = \frac{0.35\ N\ \cdot m}{2.6}$$

$$Output\ Torque = \ 0.13\ N\ \cdot m$$

MASSA TERMICA

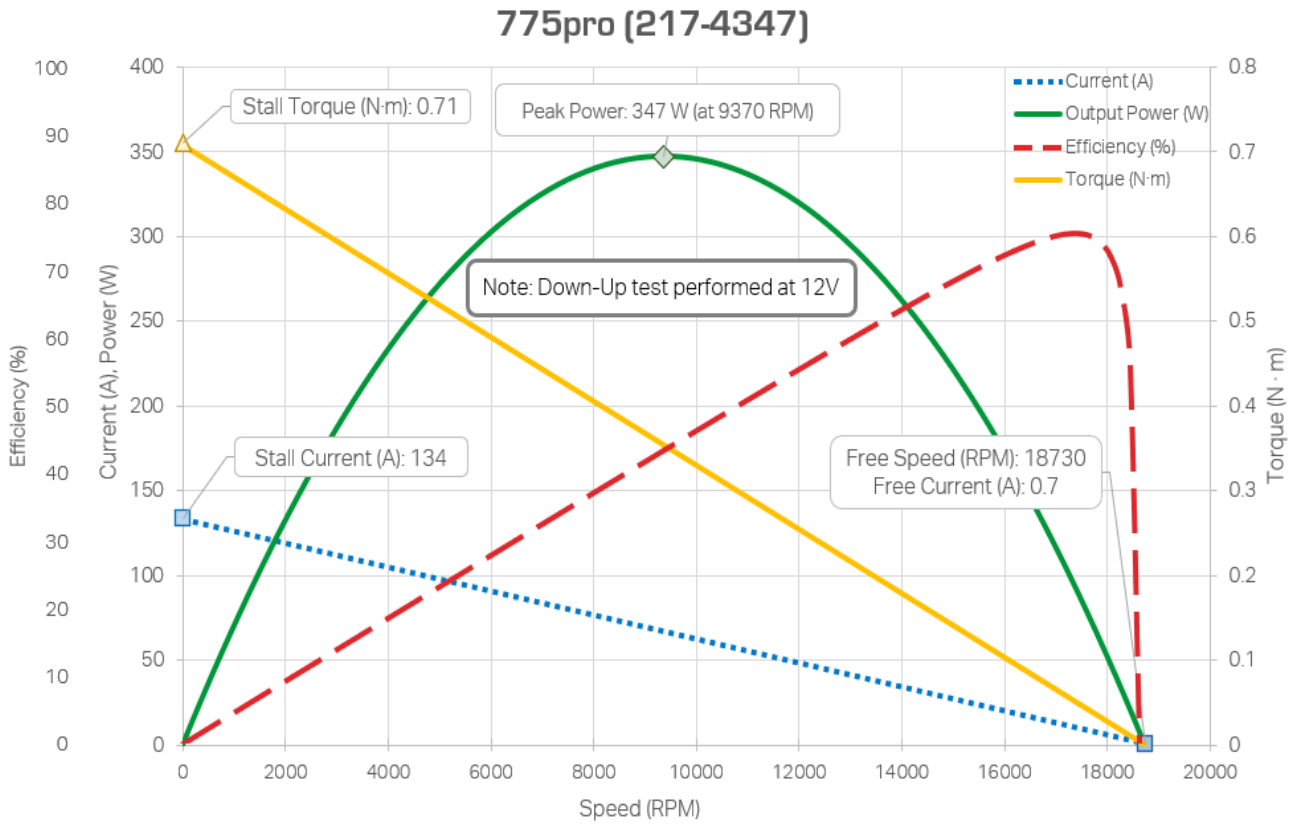
La maggior parte dei motori, se spinti alla coppia di stallo o alla potenza di picco, si bruceranno se lasciati lì per troppo tempo. Tuttavia, quel tempo accettabile varia da motore a motore. Alcuni motori possono avere una potenza di picco molto elevata, ma possono funzionare a piena potenza solo in brevi raffiche. Altri motori non hanno problemi a rimanere alla loro potenza di picco, ma possono presentare altri inconvenienti (assorbimento di corrente maggiore, potenza di picco inferiore, ecc.).

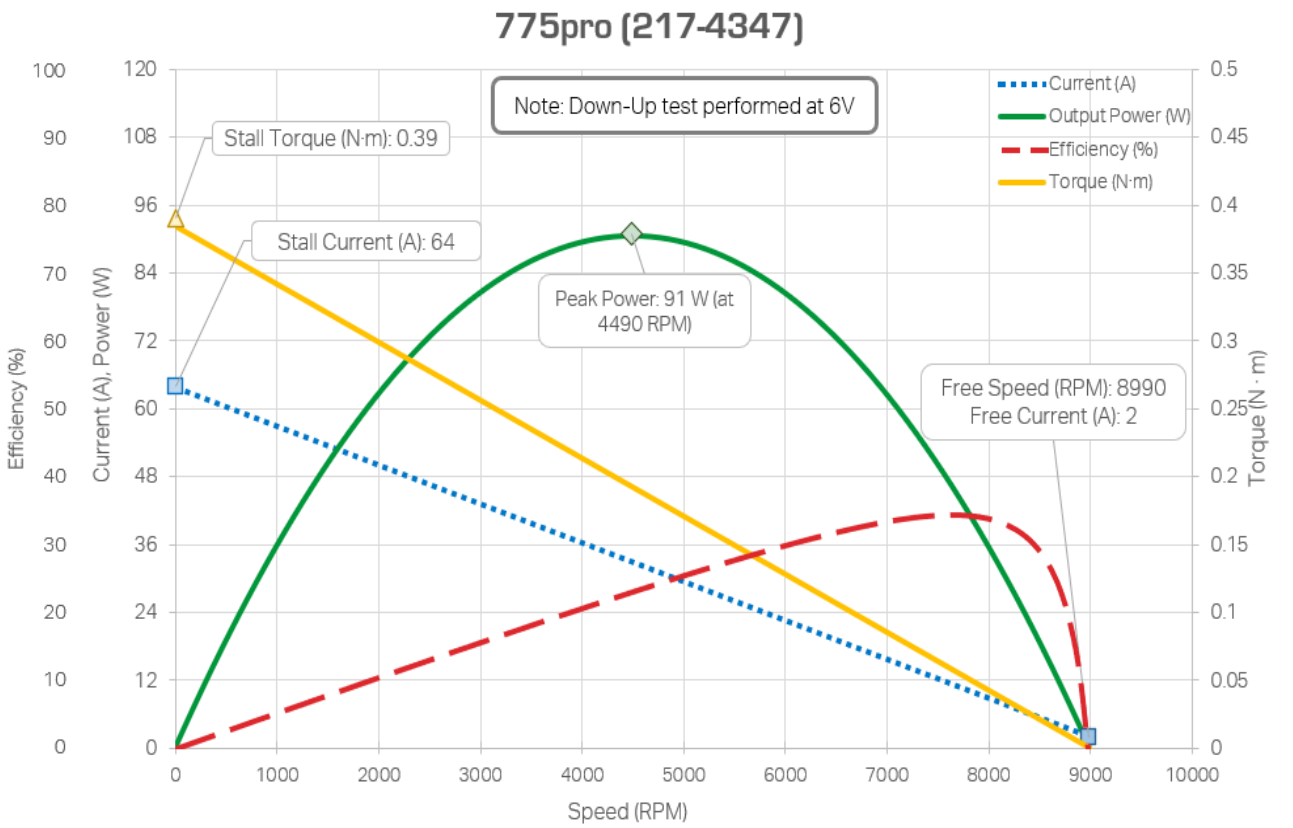
Poiché questa proprietà è così intrinsecamente dipendente dal sistema e dall'applicazione, in genere non viene pubblicata dai produttori.
La massa termica può essere approssimata dividendo la potenza di picco di un motore per il suo peso. Per esempio:

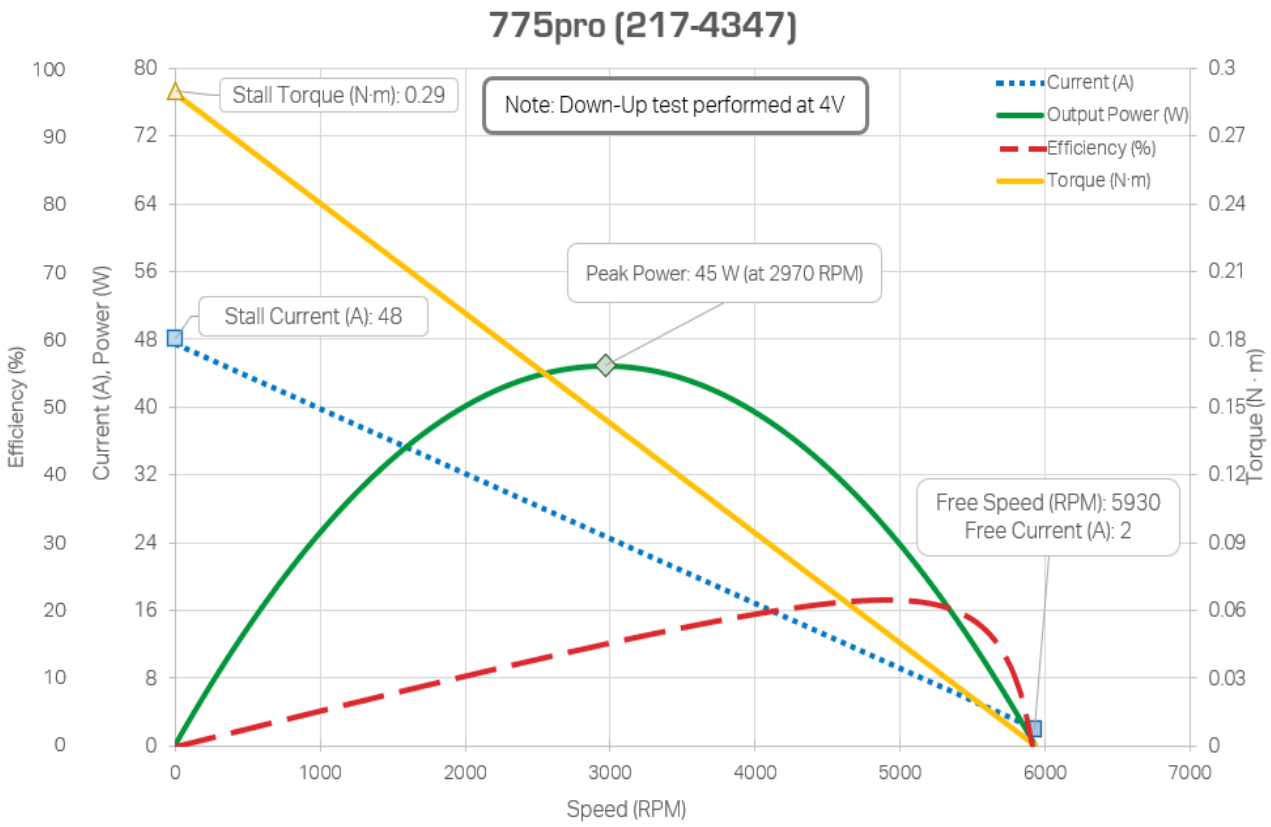
	Picco di potenza	Peso del motore	Massa termica
Motore A	330 W	0,75 libbre	440 W/libbra
Motore B	215 W	2,16 libbre	99,5 W/libbra

In questo scenario, il motore A sarebbe più utile per brevi raffiche di potenza elevata, mentre il motore B potrebbe sostenere la sua potenza molto più a lungo.

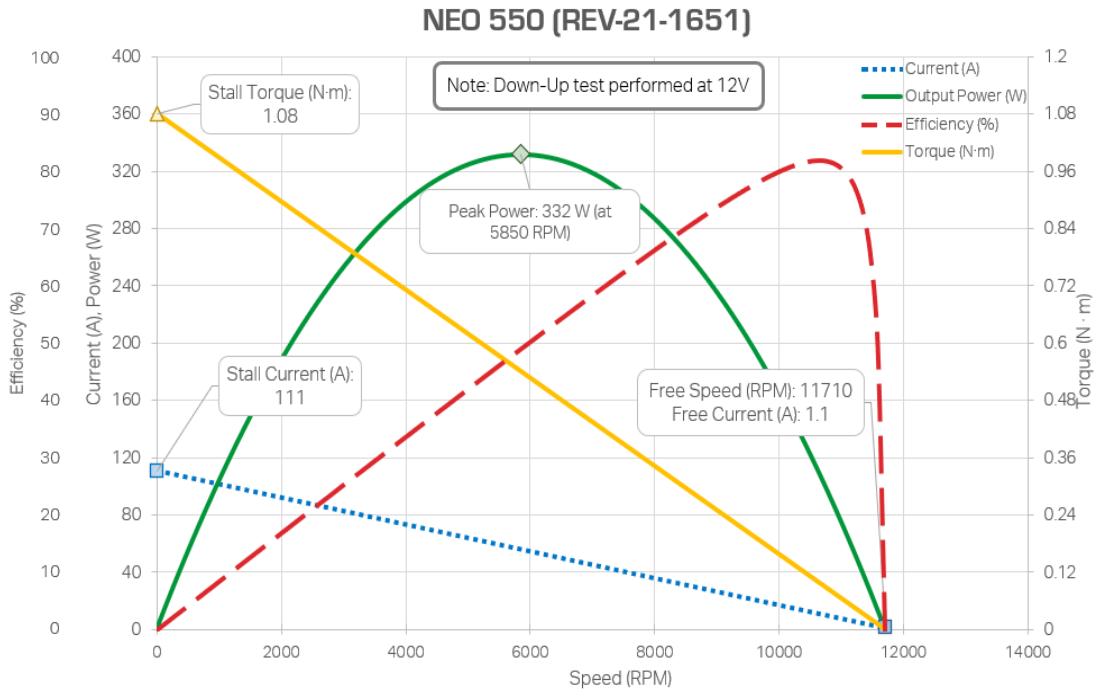
ESEMPI CURVE DI POTENZA MOTORE 775 A 12- 6- 4 VOLT

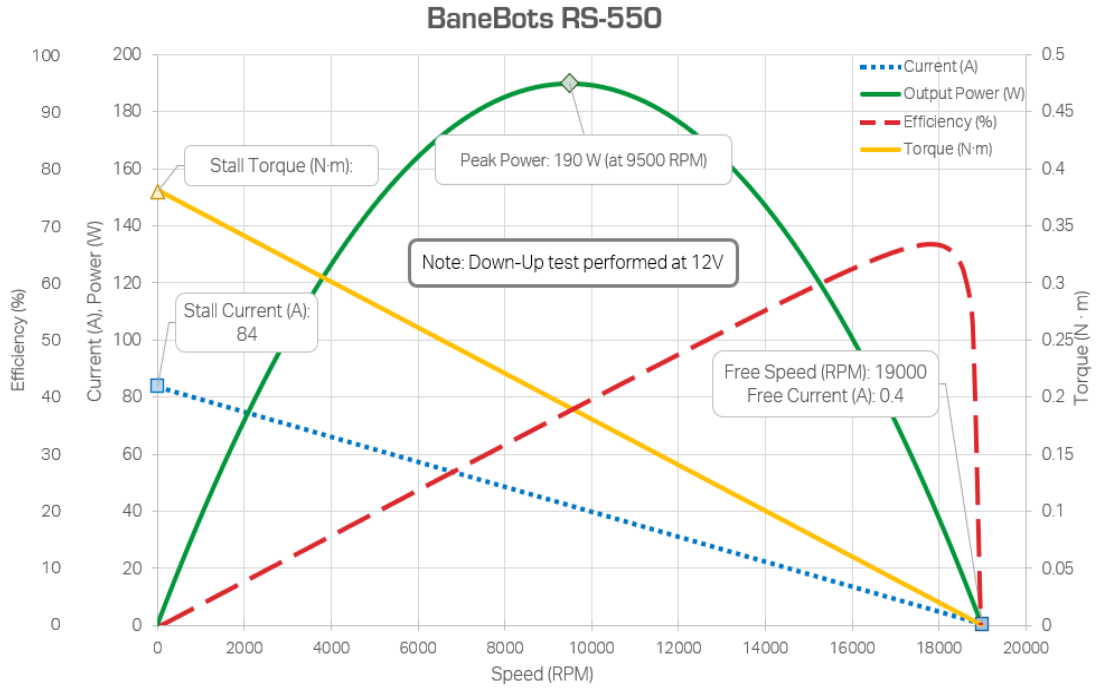


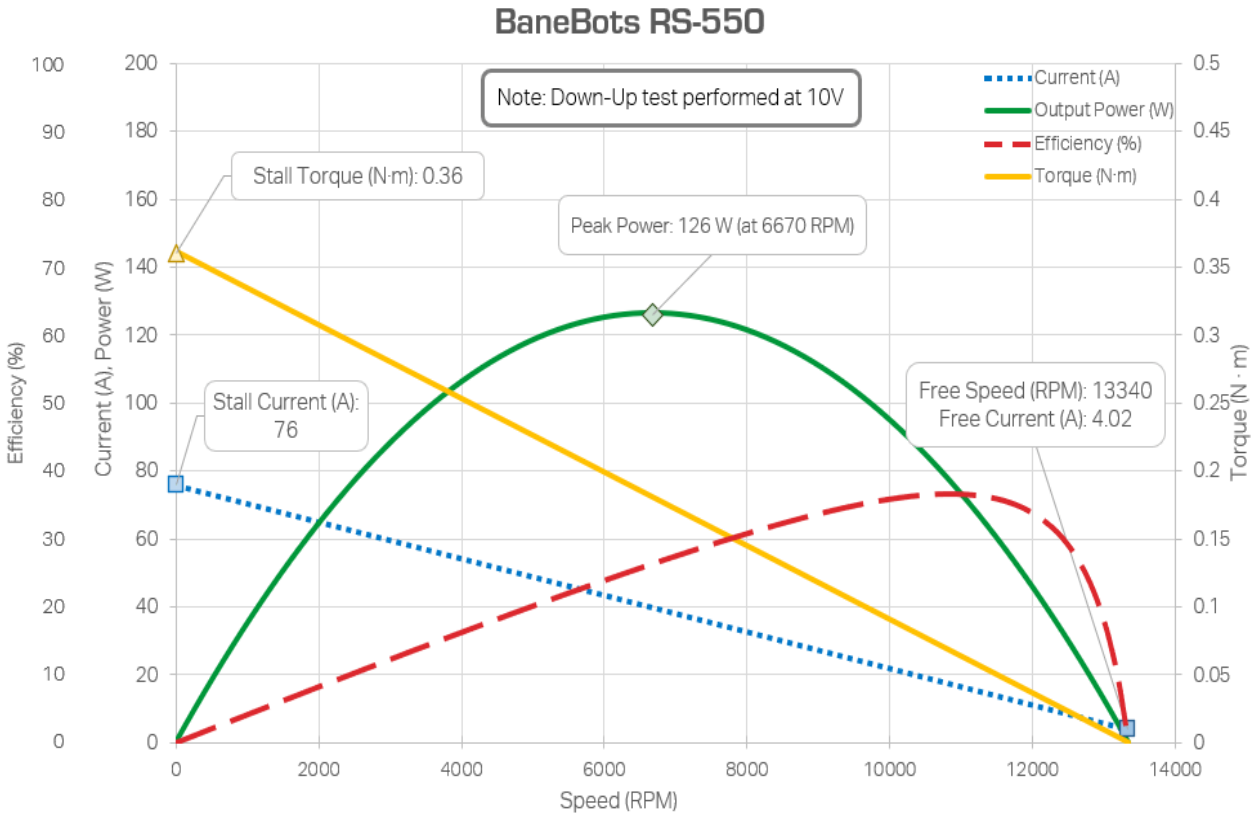


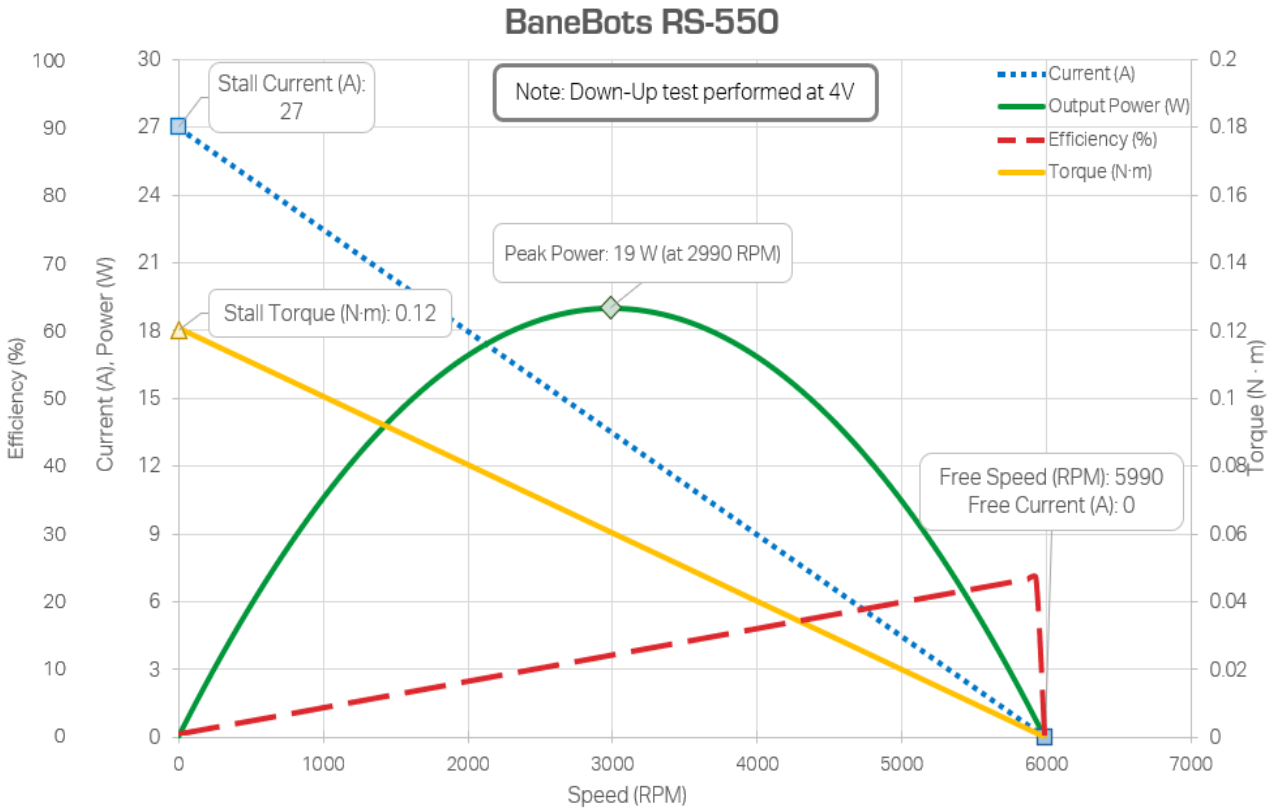

775pro (217-4347)
Stall Torque (N·m): 0.29
Note: Down-Up test performed at 4V
Current (A)
Output Power (W)
Efficiency (%)
Torque (N·m)
Peak Power: 45 W (at 2970 RPM)
Stall Current (A): 48
Free Speed (RPM): 5930
Free Current (A): 2
Efficiency (%)
Current (A), Power (W)
Torque (N · m)
Speed (RPM)
0
1000
2000
3000
4000
5000
6000
7000

CURVE POTENZA MOTORI DC RS-550






BaneBots RS-550
Note: Down-Up test performed at 10V
Stall Torque (N·m): 0.36
Peak Power: 126 W (at 6670 RPM)
Stall Current (A): 76
Free Speed (RPM): 13340
Free Current (A): 4.02
Current (A)
Output Power (W)
Efficiency (%)
Torque (N·m)
Efficiency (%)
Current (A), Power (W)
Torque (N · m)
Speed (RPM)

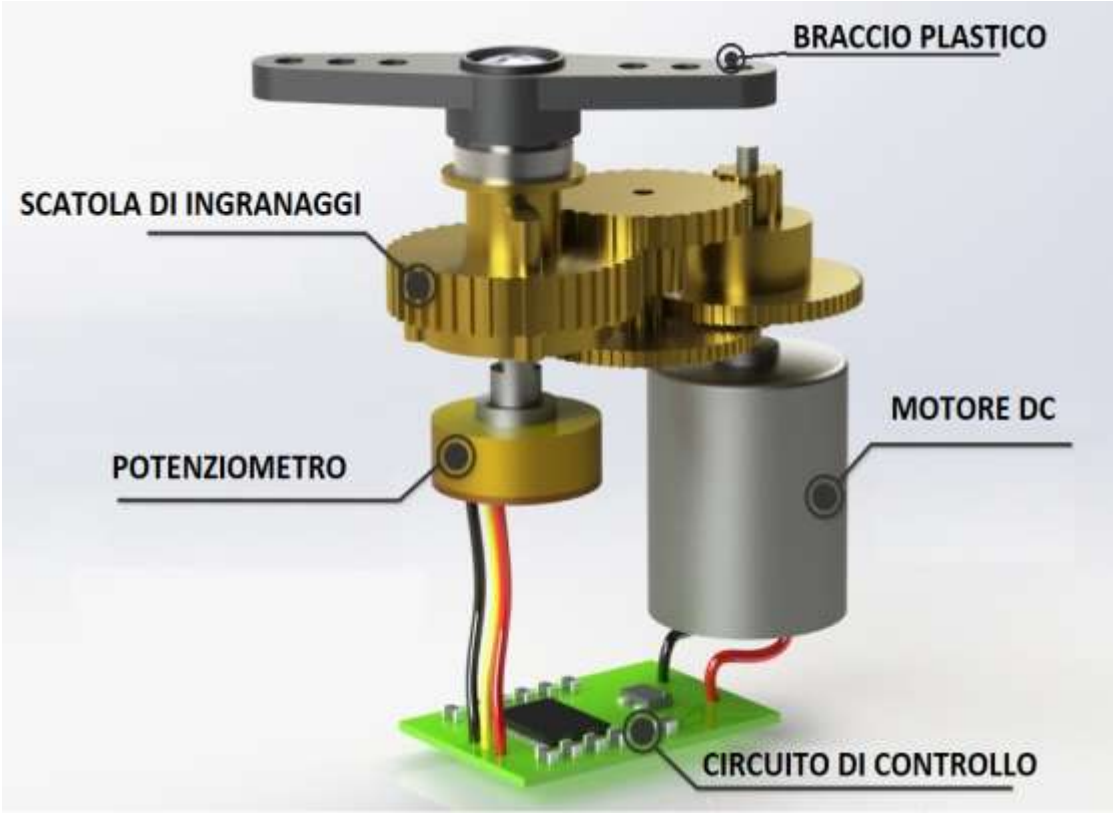

BaneBots RS-550
Note: Down-Up test performed at 4V
Stall Current (A): 27
Peak Power: 19 W (at 2990 RPM)
Stall Torque (N·m): 0.12
Free Speed (RPM): 5990
Free Current (A): 0
Current (A)
Output Power (W)
Efficiency (%)
Torque (N·m)
Efficiency (%)
Current (A), Power (W)
Torque (N · m)
Speed (RPM)

SERVOMOTORI

Un servomotore elettrico rotativo è un motore che permette il controllo di precisione della posizione angolare.
Il servomotore classico è composto da due elementi principali: il sensore di posizione o feedback e il motore a cui si può aggiungere un riduttore e un freno in caso di necessità.
Richiede inoltre un azionamento e/o un controllore più o meno sofisticato a seconda del livello di controllo che si vuole raggiungere.

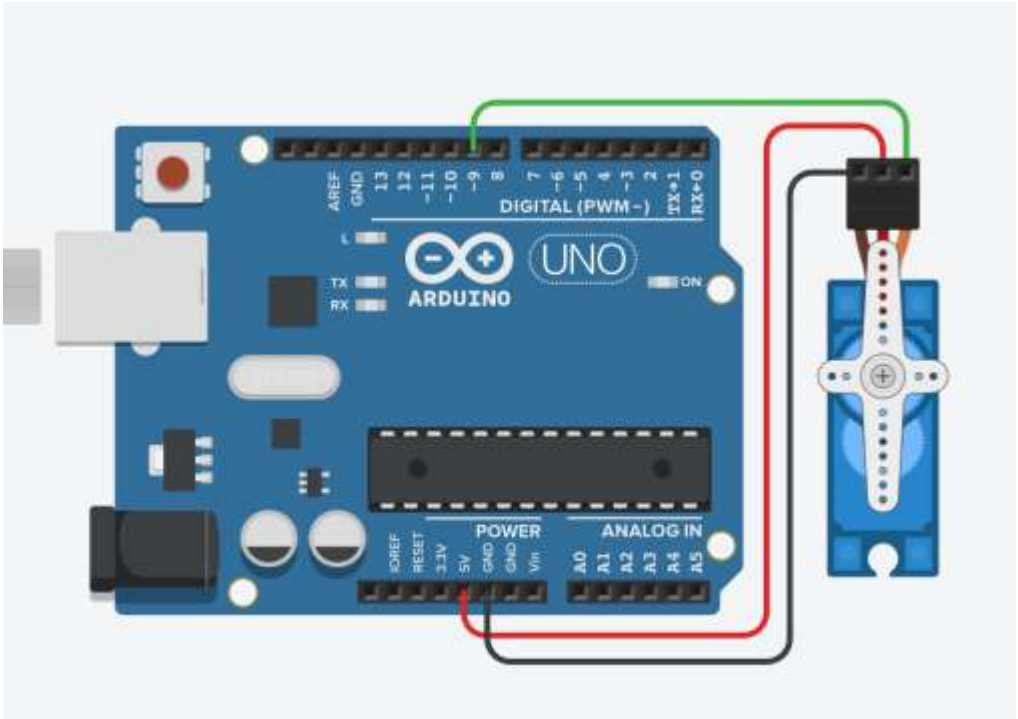
Hobby

Industriale



GESTIONE SERVOMOTORE DIRETTA CON ARDUINO

Muovere l'albero del servo a destra e a sinistra (180°→0° e viceversa) con passo di 1°.



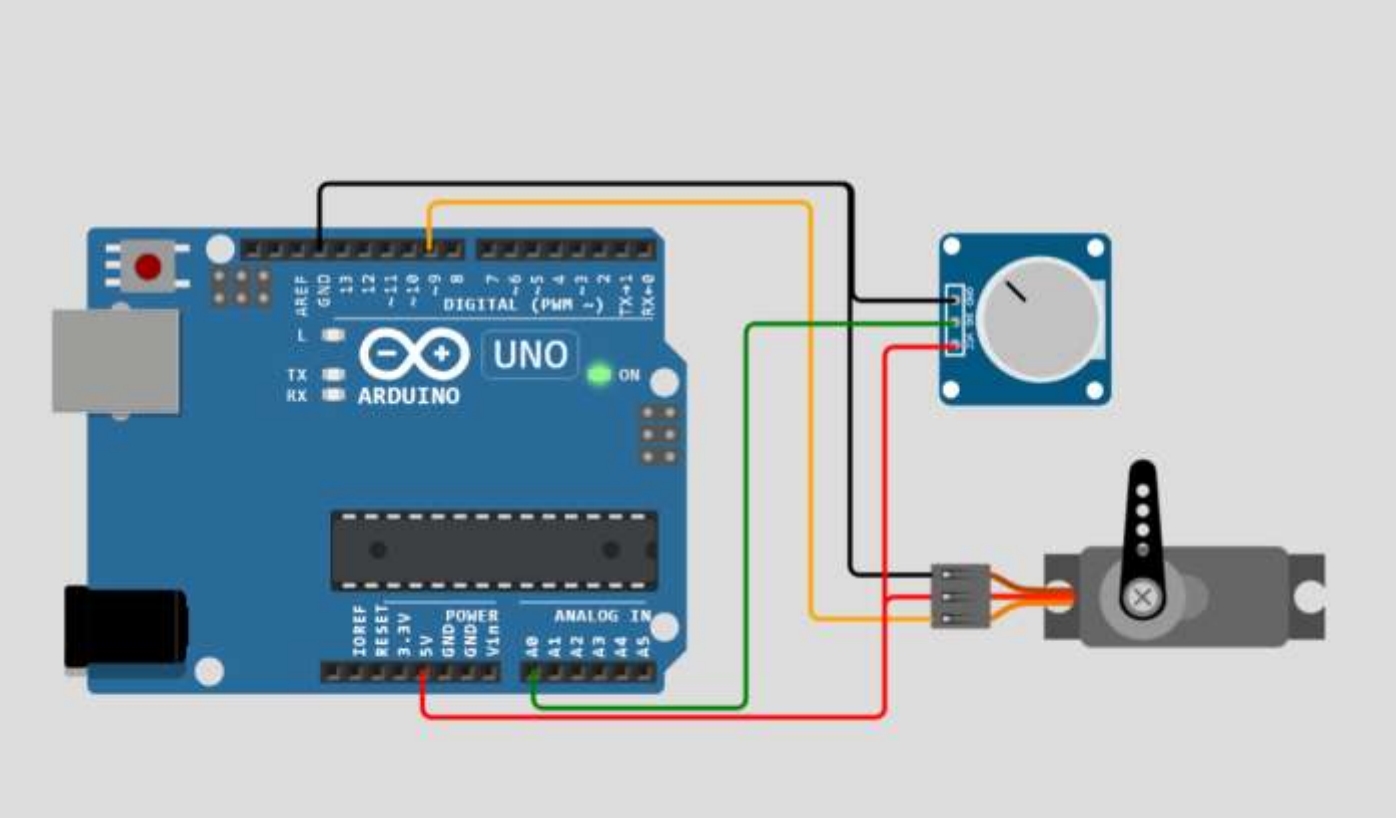
CODICE

```
#include <Servo.h>
Servo servo_9;
int pos = 0;

void setup()
{
  servo_9.attach(9, 500, 2500);
}

void loop()
{
  // sweep the servo from 0 to 180 degrees in steps of 1 degrees
  for (pos = 0; pos <= 180; pos += 1) {
    servo_9.write(pos);
    delay(15); // Wait for 15 millisecond(s)
  }
  for (pos = 180; pos >= 0; pos -= 1) {
    servo_9.write(pos);
    // wait 15 ms for servo to reach the position
    delay(15); // Wait for 15 millisecond(s)
  }
}
```

COMPITO

1. Modificare il circuito affinché il servomotore abbia una alimentazione dedicata con transistor NPN.
2. Modificare il codice per gestire due servomotori tramite comandi inviati dal monitor seriale (1 → motore DX, 2 → motore SX, 3 → entrambi i motori e 4 → stop).

ESERCIZIO GESTIONE SERVOMOTORE CON ARDUINO E POTENZIOMETRO



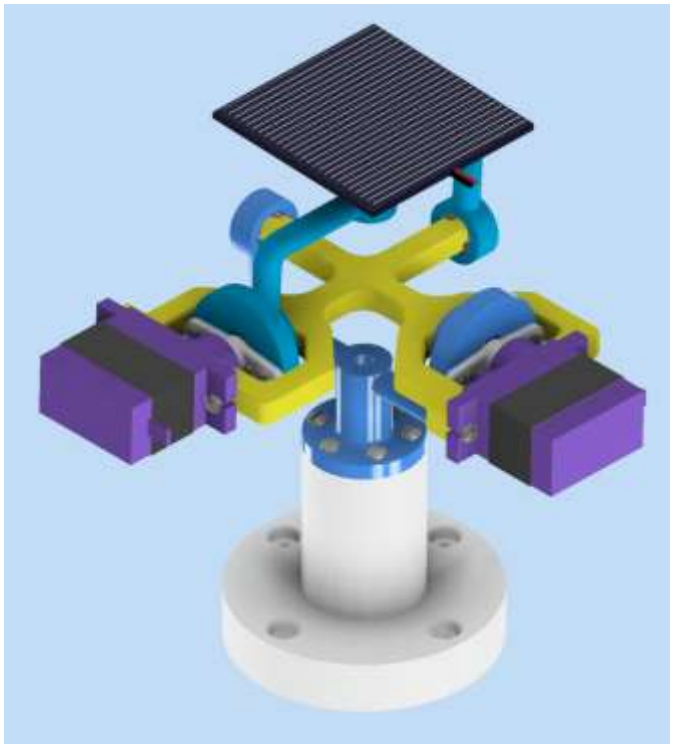
simulabile su "wokwi.com"

CODICE

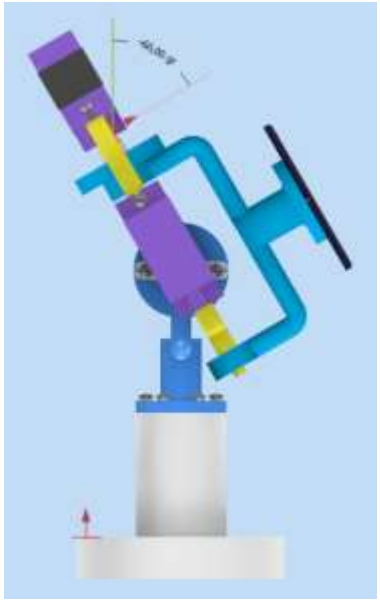
```
#include <Servo.h>


Servo myservo;  // create servo object to control a servo

int potpin = 0;  // analog pin used to connect the potentiometer
int val;    // variable to read the value from the analog pin

void setup() {
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}

void loop() {
  val = analogRead(potpin);            // reads the value of the potentiometer (value between 0 and 1023)
  val = map(val, 0, 1023, 0, 180);     // scale it to use it with the servo (value between 0 and 180)
  myservo.write(val);                  // sets the servo position according to the scaled value
  delay(15);                           // waits for the servo to get there
}
```

INSEGUITORE SOLARE CON 2 SERVO

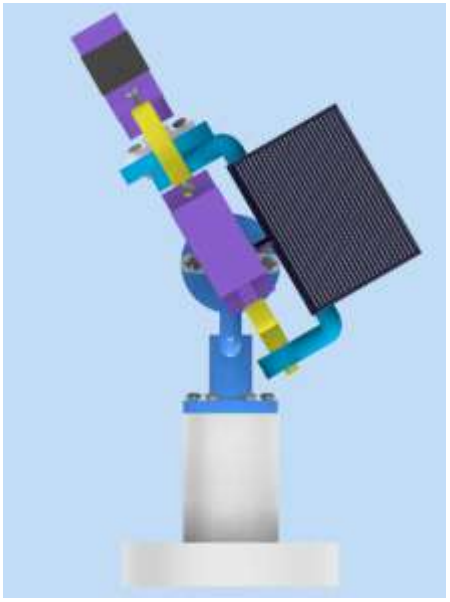
Un inseguitore solare è un sistema che consente di rilevare la posizione del sole per sfruttare il massimo irraggiamento possibile con griglie di pannelli fotovoltaici orientabili.
Il sistema in figura è costituito da due servomotori che permettono di muovere la piccola cella solare nello spazio su due piani.



0°;0°



60°;0°



60°;-45°

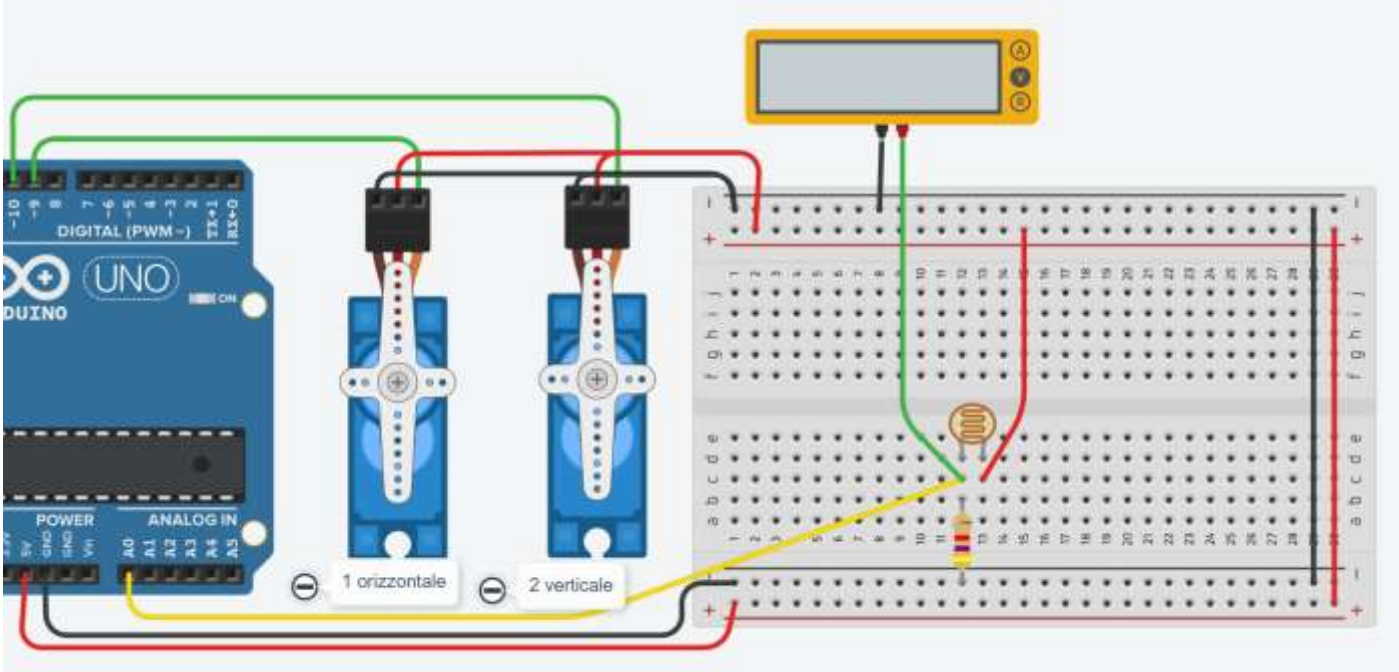
60°;45°

ESERCIZIO INSEGUITORE SOLARE

La cella solare fornisce una tensione da 0 → 5 volt proporzionale all'intensità dell'irraggiamento solare.

Si devono ruotare i due servo nello spazio in modo da individuare la direzione in cui si ha il massimo irraggiamento.

Per semplicità testare solo un angolo rispetto al piano orizzontale (30°) mentre per il movimento rispetto al piano verticale spostarsi da -90° a +90° con passo 1°.



Il servo 1 si inclina a 30° mentre il servo 2 spazia da 180 a 0°.

Ad ogni movimento rileva l'intensità luminosa tramite una fotoresistenza (usata per simulare lo spostamento del sole) e aggiorna il valore massimo e la posizione a cui è stato rilevato.

CODICE

```
#include <Servo.h>

Servo servo_o;
Servo servo_v;

int servo_o_pos0= 30;
int servo_v_pos0= 180;
int servo_v_pos_max=180;
int sensorValue;
int sensorValueMax=0;

void setup()
{
  Serial.begin(9600);
  servo_o.attach(9);
  servo_v.attach(10,500,2500);
  servo_o.write(servo_o_pos0);
  servo_v.write(servo_v_pos0);
  pinMode(A0, INPUT);
  delay(2000);
}

void loop()
{

for (int i = 0; i <= 180; i++) {

  sensorValue = analogRead(A0);
  if (sensorValue>sensorValueMax) {
    sensorValueMax = sensorValue;
    servo_v_pos_max = i;
    Serial.println(sensorValue);
    Serial.println(servo_v_pos_max);
  }

  servo_v.write(servo_v_pos0-i);
  delay(100);
}

delay(1000);
}
```

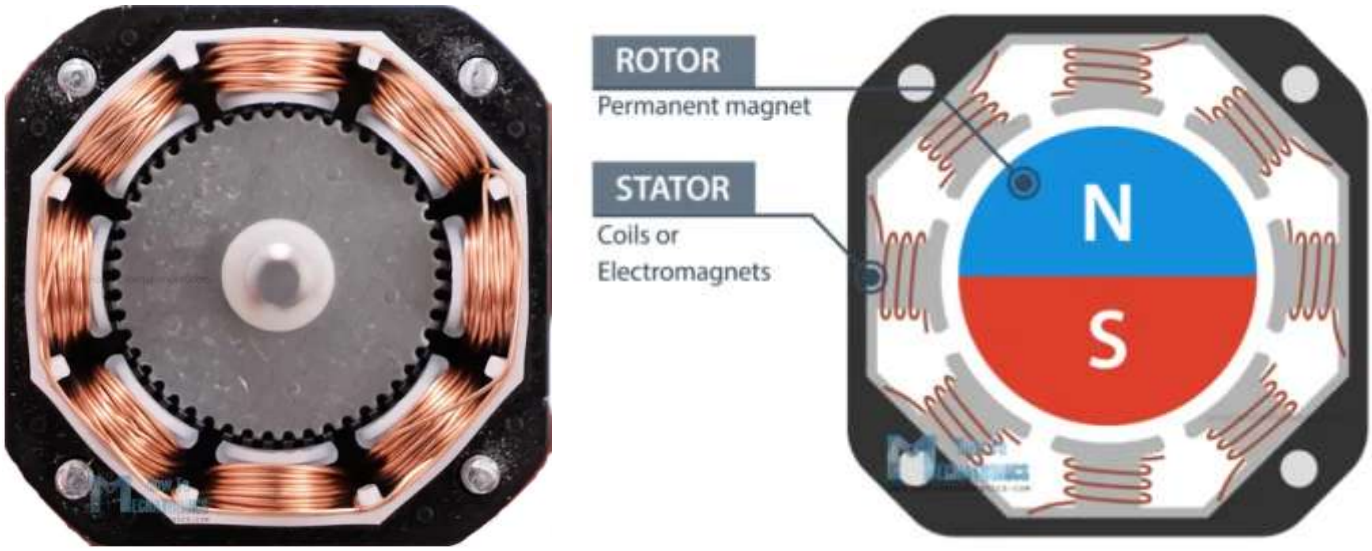
COMPITO

1. Modificare il circuito affinché il servomotore 1 rilevi la posizione a massimo irraggiamento sul piano orizzontale 30° e 40°.

MOTORE STEPPER (PASSO-PASSO)

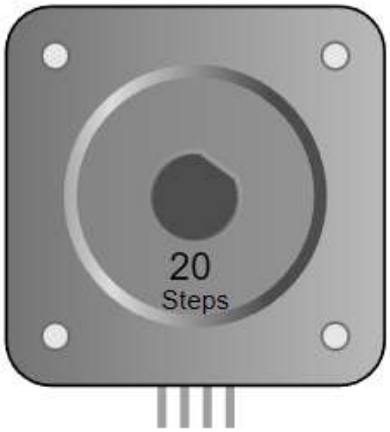
Il motore passo paso è un tipo di motore a corrente continua sincrono senza spazzole che, diversamente dagli altri tipi standard di motori elettrici, non ruota in continuazione per un numero arbitrario di giri fino a che non viene interrotto il flusso di corrente continua. Al contrario, i motori passo passo sono dispositivi con controllo digitale delle sorgenti in input e in output, per l'avvio e l'arresto di precisione. Sono costruiti in modo tale che la corrente che li attraversa passi in una serie di bobine disposte in fase, che possono essere attivate o disattivate in rapida sequenza. Questo permette al motore di girare una frazione di rotazione alla volta, ed è a ciascuno di questi predeterminati passi (step in inglese) che il motore deve il suo nome (motore stepper).

Un motore passo passo è costruito in modo da suddividere una singola rotazione completa in un numero di gran lunga minore di rotazioni parziali uguali.



Il risultato finale è che un motore passo passo può trasferire movimenti minuziosamente accurati a parti meccaniche che richiedono elevati gradi di precisione. Di default il motore passo-passo sposta l'albero motore di 1,8 gradi per passo (200 passi per giro). In generale la velocità massima di un motore stepper è di circa 1000 rpm. All'aumentare della velocità di rotazione diminuisce la coppia motrice disponibile (che può essere aumentata montando un riduttore).

Usando opportuni driver il motore supporta anche mezzo passo (0,9 gradi per passo / 400 passi per giro) ed anche micropassi più piccoli (ad es. ½, 1/4 o 1/8 di passo).



Name	Description
A-	Coil A negative signal
A+	Coil A positive signal
B+	Coil B positive signal
B-	Coil B negative signal

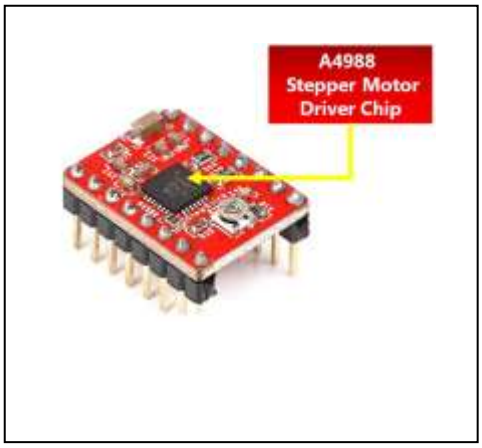
DRIVER A4988

Quando si utilizza un motore passo-passo è necessario un chip driver (es. driver A4988) in grado di fornire l'elevata quantità di corrente richiesta dalle bobine del motore.
I motori passo-passo standard hanno 200 passi per giro (i passi sono distanziati di 1,8 gradi).

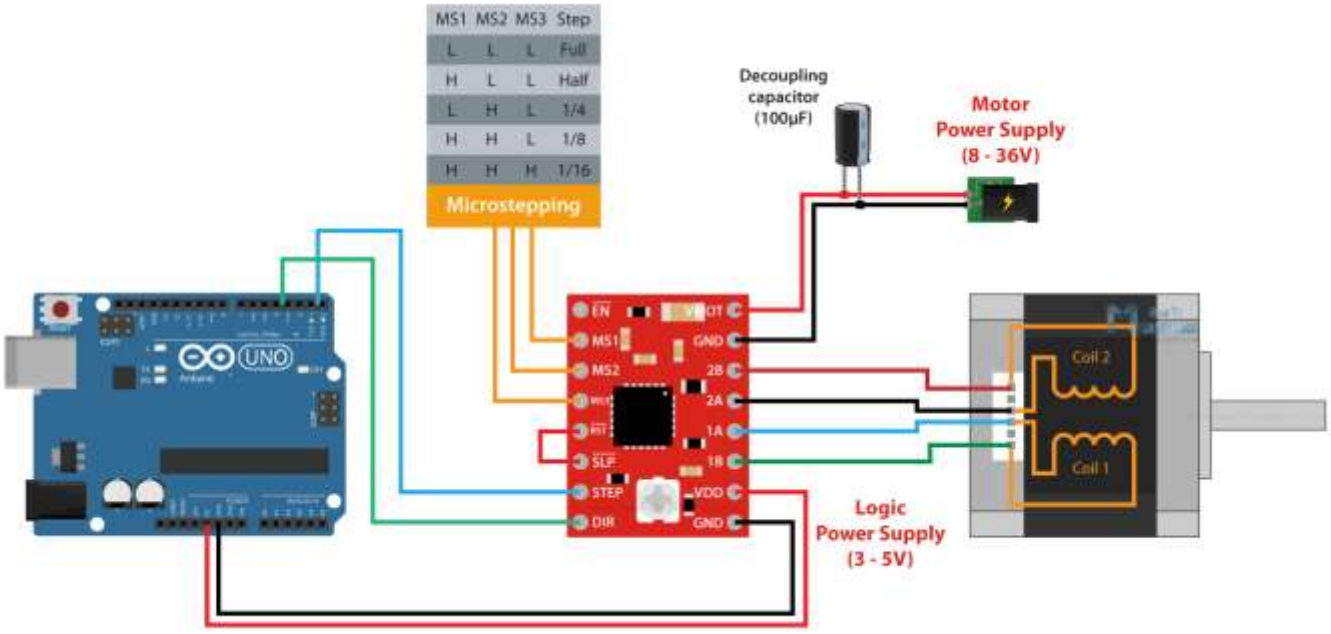
Il driver stepper supporta il microstepping: consnete di muovere il motore a meno di un passo per ogni impulso.
Il microstepping consente un controllo più preciso del movimento del motore (con una riduzione della coppia motrice).

Utilizzare i pin MS1/MS2/MS3 per selezionare la configurazione microstepping per il driver stepper:

MS1	MS2	MS3	Operation mode	Degrees	Microsteps/revolution
0	0	0	Full step (default)	1.8	200
1	0	0	Half step	0.9	400
0	1	0	1/4 step*	0.45	800
1	1	0	1/8 step*	0.225	1600
1	1	1	1/16 step*	0.1125	3200



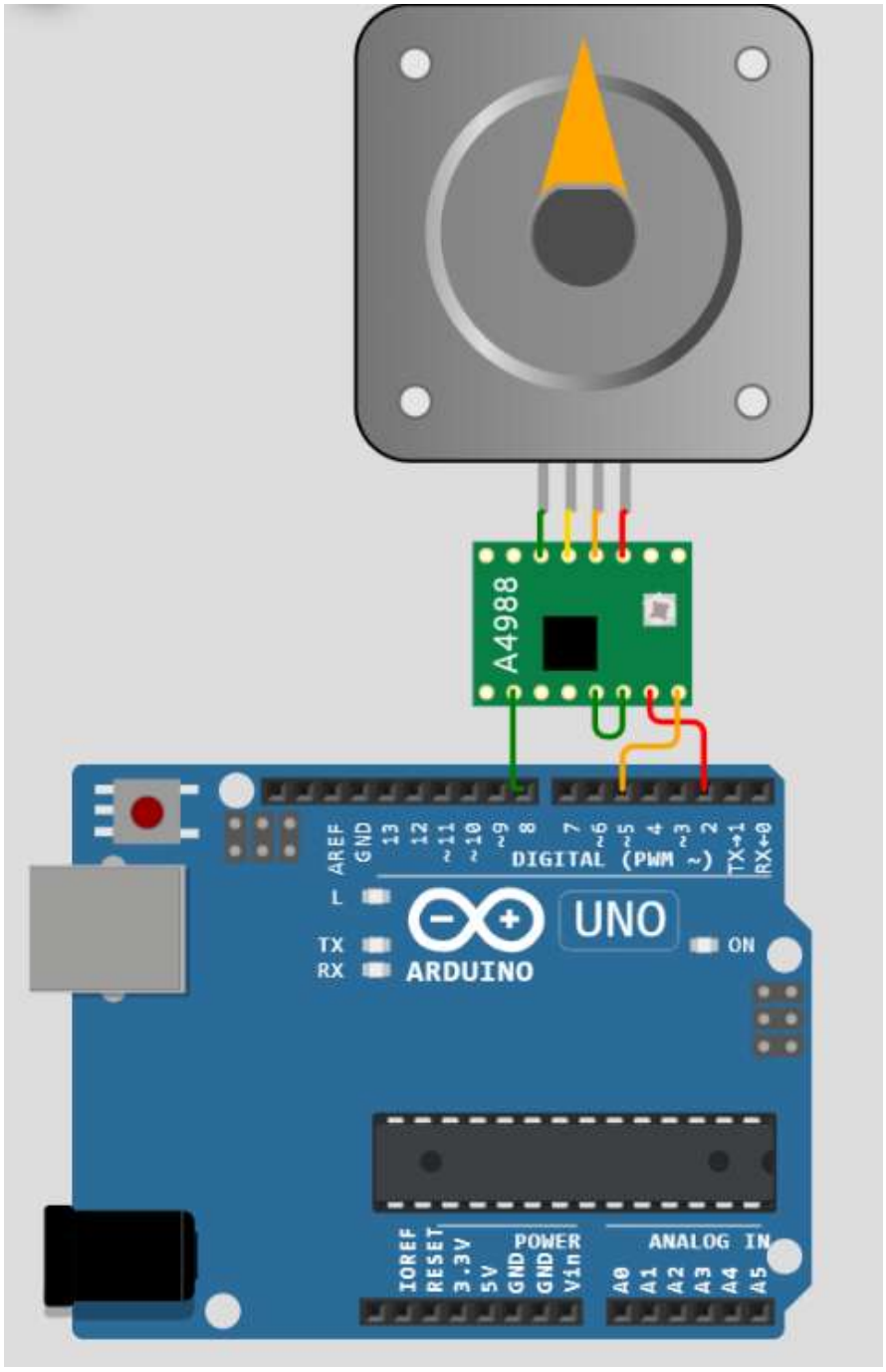
Come collegare il driver A4988 con il motore passo-passo e il controller Arduino.



NOTA BENE:
Per motori a bassa resistenza interna (tipica degli stepper) è necessario un driver di corrente e non un driver di tensione come l'L298N. I motori a bassa impedenza sono controllati in corrente, non in tensione.
Per valori di resistenza dell'avvolgimento oltre 30-60 ohm un L298N funziona senza bruciarsi, ma la velocità massima è molto inferiore rispetto a quella ottenibile con un driver di corrente.

UTILIZZO DEL DRIVER PASSO-PASSO A4988

Collegare i pin del motore passo-passo ai pin 1B/1A/2A/2B del driver.
Il pin RESET deve essere HIGH e quindi si può collegare al pin SLEEP adiacente che è impostato HIGH di default.
Utilizzare il pin STEP per spostare il motore passo-passo.
Ogni impulso ALTO su questo pin sposterà il motore di un passo (o micropasso, a seconda dei pin MS1/MS2/MS3).
Quando il pin DIR è ALTO, il motore passo-passo si sposterà in senso orario.
Quando il pin DIR è BASSO, il motore si muoverà in senso antiorario.
Ad esempio, se DIR, MS1 e MS3 sono LOW e MS2 è HIGH (modalità 1/4 step), l'impulso del pin STEP sposterà il motore di 1/4 step (0,45 gradi) in senso antiorario.



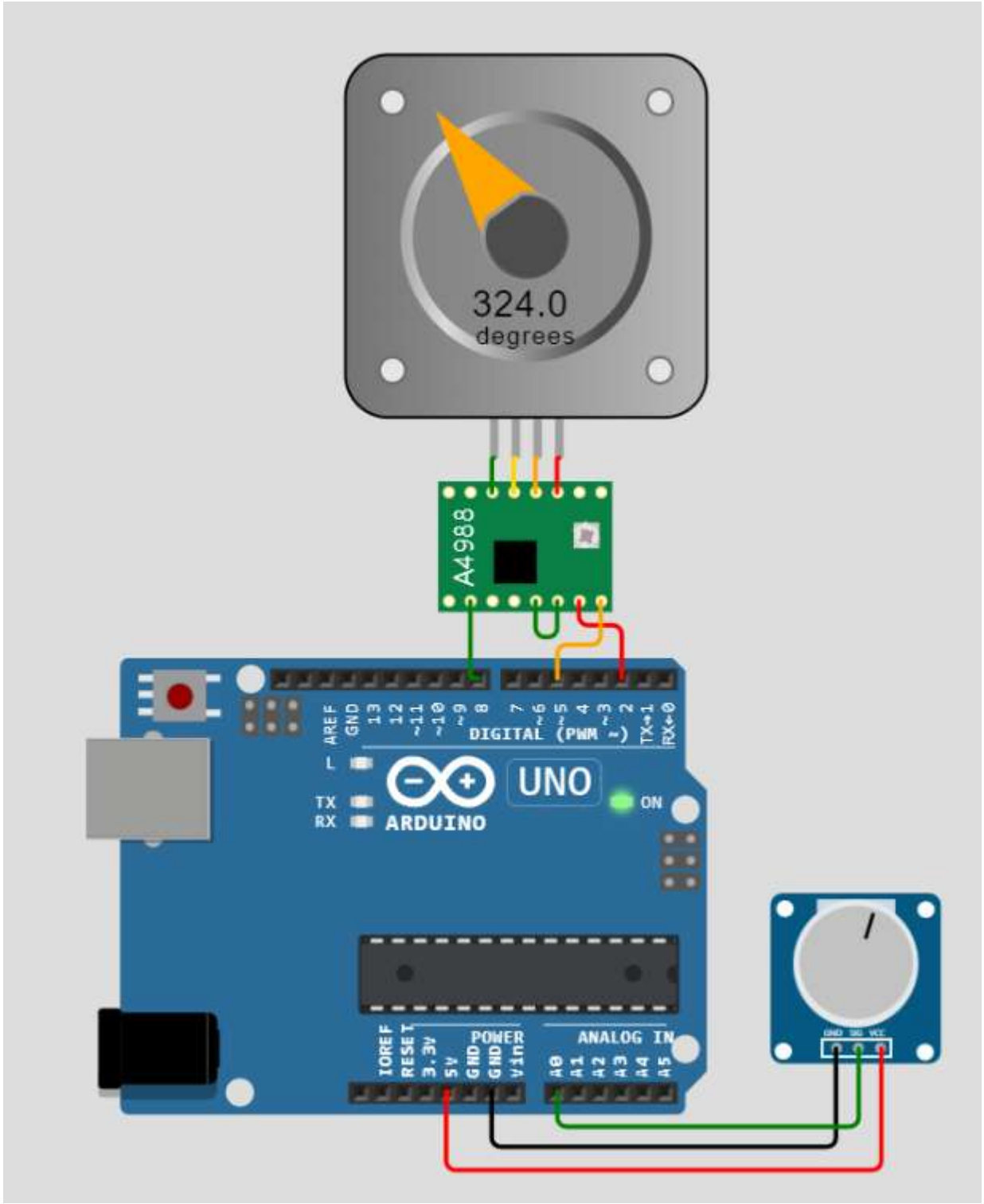
simulabile su "wokwi.com"

CODICE

```
#define DIR_PIN  5  // X
#define STEP_PIN 2  // X
#define EN_PIN 8

#define DELAY_ST 5000

int stps=400;  // 2 giri completi

void setup() {
  pinMode(DIR_PIN, OUTPUT);
  pinMode(STEP_PIN, OUTPUT);
  pinMode(EN_PIN, OUTPUT);
  digitalWrite(EN_PIN, LOW);
  delay(1000);
}

void loop() {
  // rotazione ORARIA
  step(true, DIR_PIN, STEP_PIN, stps);
  delay(1000);
  // rotazione ANTIORARIA
  step(false, DIR_PIN, STEP_PIN, stps);
  delay(1000);
}

// dir = true= oraria
void step(boolean dir, byte dirPin, byte stepperPin, int steps)
{
  digitalWrite(dirPin, dir);
  delay(100);
  for (int i = 0; i< steps; i++)  {
    digitalWrite(stepperPin, HIGH);
    delayMicroseconds(DELAY_ST);
    digitalWrite(stepperPin, LOW);
    delayMicroseconds(DELAY_ST);
  }
}
```

UTILIZZO DEL DRIVER PASSO-PASSO A4988 + POTENZIOMETRO

Regolare al velocità di rotazione del motore stepper tramite un potenziometro.

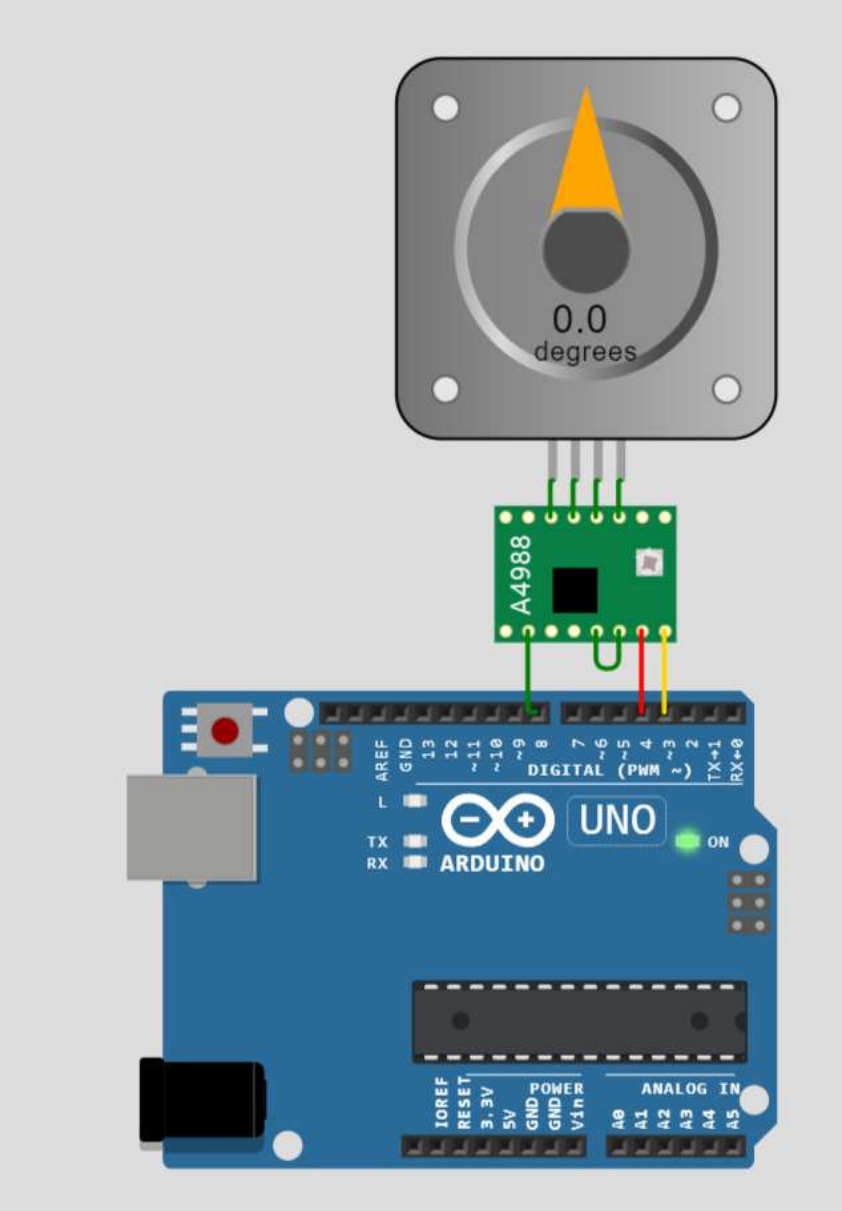


simulabile su "wokwi.com"

CODICE

```
#define DIR_PIN  5  // X
#define STEP_PIN 2  // X
#define EN_PIN 8

int stps=400;  // 2 giri completi
int pot, delay_step;

void setup() {
  pinMode(DIR_PIN, OUTPUT);
  pinMode(STEP_PIN, OUTPUT);
  pinMode(EN_PIN, OUTPUT);
  digitalWrite(EN_PIN, LOW);
  delay(1000);
}

void loop() {
  speedControl();
  // rotazione ORARIA
  step(true, DIR_PIN, STEP_PIN, stps);
  delay(1000);
  // rotazione ANTIORARIA
  step(false, DIR_PIN, STEP_PIN, stps);
  delay(1000);
}

// dir = true= oraria
void step(boolean dir, byte dirPin, byte stepperPin, int steps)
{
  digitalWrite(dirPin, dir);
  delay(100);
  for (int i = 0; i< steps; i++)  {
    digitalWrite(stepperPin, HIGH);
    delayMicroseconds(delay_step);
    digitalWrite(stepperPin, LOW);
    delayMicroseconds(delay_step);
  }
}

void speedControl() {
  pot = analogRead(A0); // Read the potentiometer value
  delay_step = map(pot, 0, 1023, 1000, 5000); // Convert the analog input from 0 to 1024, to 300 to 3000
}
```

UTILIZZO DEL DRIVER PASSO-PASSO A4988 CON MEZZO PASSO



simulabile su "wokwi.com"

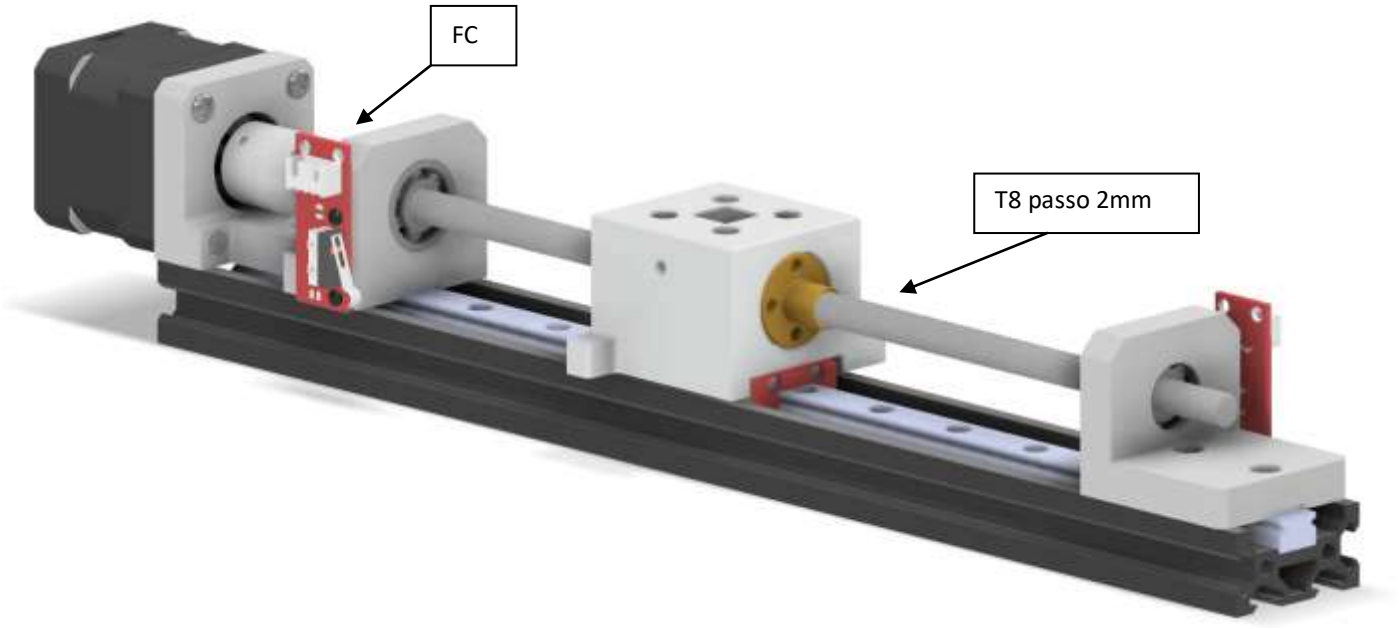
CODICE

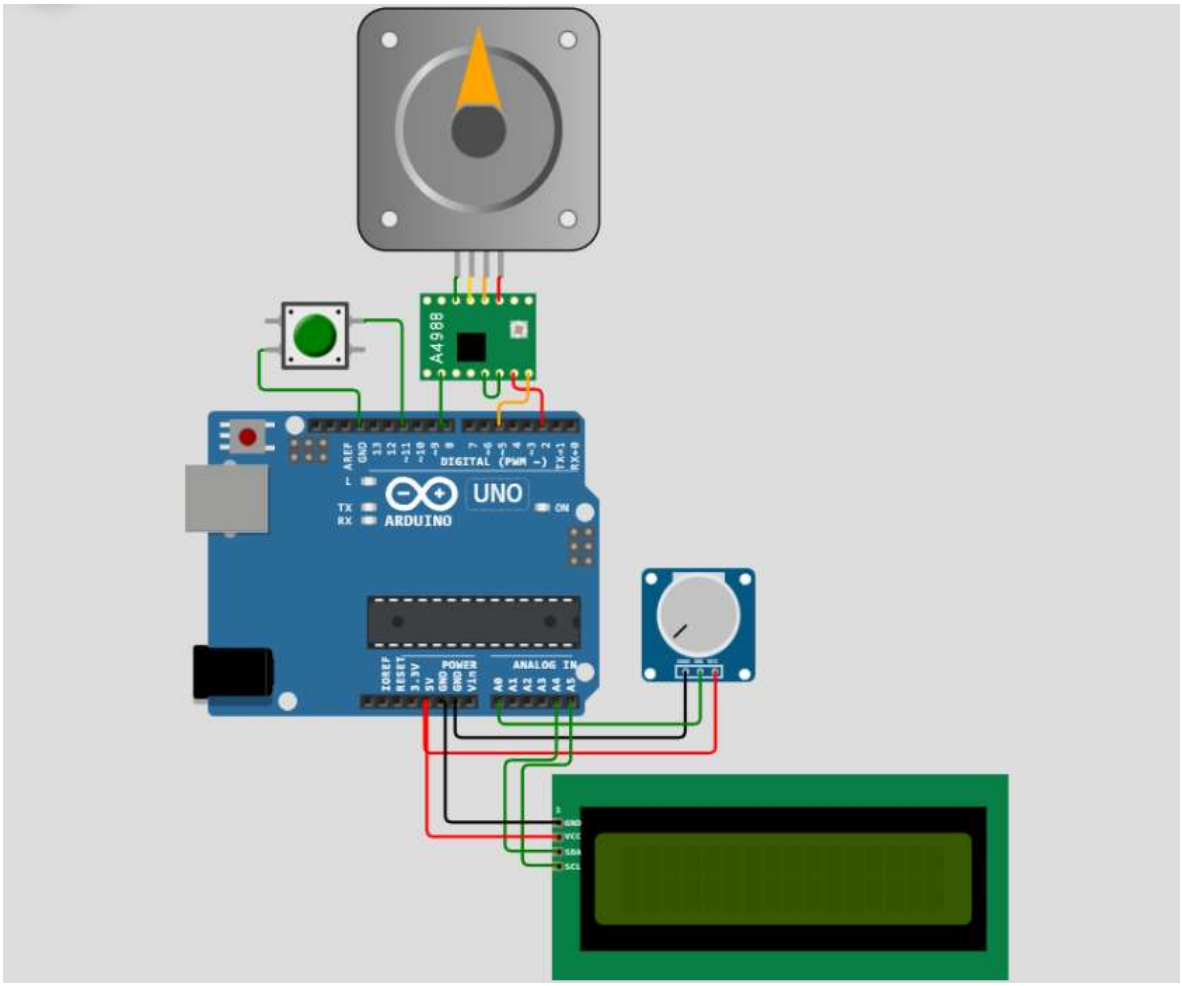
```
#define DIR_PIN  3
#define STEP_PIN 4
#define MS1_PIN 8

#define DELAY_ST 2000

void setup() {
  pinMode(DIR_PIN, OUTPUT);
  pinMode(STEP_PIN, OUTPUT);
  pinMode(MS1_PIN, OUTPUT);
  digitalWrite(MS1_PIN, LOW);
  delay(1000);
}

void loop() {

  // rotazione ORARIA
  digitalWrite(DIR_PIN, HIGH);

  // 1 giro completo
  for (int i = 0; i < 200; i++) {
```

```
    // un passo
    digitalWrite(STEP_PIN, HIGH);
    delayMicroseconds(DELAY_ST);
    digitalWrite(STEP_PIN, LOW);
    delayMicroseconds(DELAY_ST);
  }

  delay(1000);

  // // rotazione ANTIORARIA
  digitalWrite(DIR_PIN, LOW);

  // 1 giro completo
  for (int i = 0; i < 200; i++) {
    //un passo
    digitalWrite(STEP_PIN, HIGH);
    delayMicroseconds(DELAY_ST);
    digitalWrite(STEP_PIN, LOW);
    delayMicroseconds(DELAY_ST);
  }

  delay(1000);

  // microstepping 1/2 --> MS1 alto
  digitalWrite(MS1_PIN, HIGH);

  // rotazione ORARIA
  digitalWrite(DIR_PIN, HIGH);

  // 1/2 giro
  for (int i = 0; i < 200; i++) {
    // un passo
    digitalWrite(STEP_PIN, HIGH);
    delayMicroseconds(DELAY_ST);
    digitalWrite(STEP_PIN, LOW);
    delayMicroseconds(DELAY_ST);
  }

  delay(1000);

  //  rotazione ANTIORARIA
  digitalWrite(DIR_PIN, LOW);

  // 1/2 giro
  for (int i = 0; i < 200; i++) {
    //un passo
    digitalWrite(STEP_PIN, HIGH);
    delayMicroseconds(DELAY_ST);
    digitalWrite(STEP_PIN, LOW);
    delayMicroseconds(DELAY_ST);
  }

  delay(1000);
  digitalWrite(MS1_PIN, LOW);
}
```

GUIDA LINEARE CON MOTORE STEPPER E BARRA FILETTATA T8 PASSO 2MM

La guida è dotata di un finecorsa meccanico “FC” che fornisce segnale “1” quando NON è premuto e “0” quando è premuto.
All’accensione il blocco mobile deve portarsi alla “HOME” definita da stato “FC=0”.
A partire da “HOME” si potranno poi effettuare gli spostamenti assegnati (in mm) dal ciclo proposto.



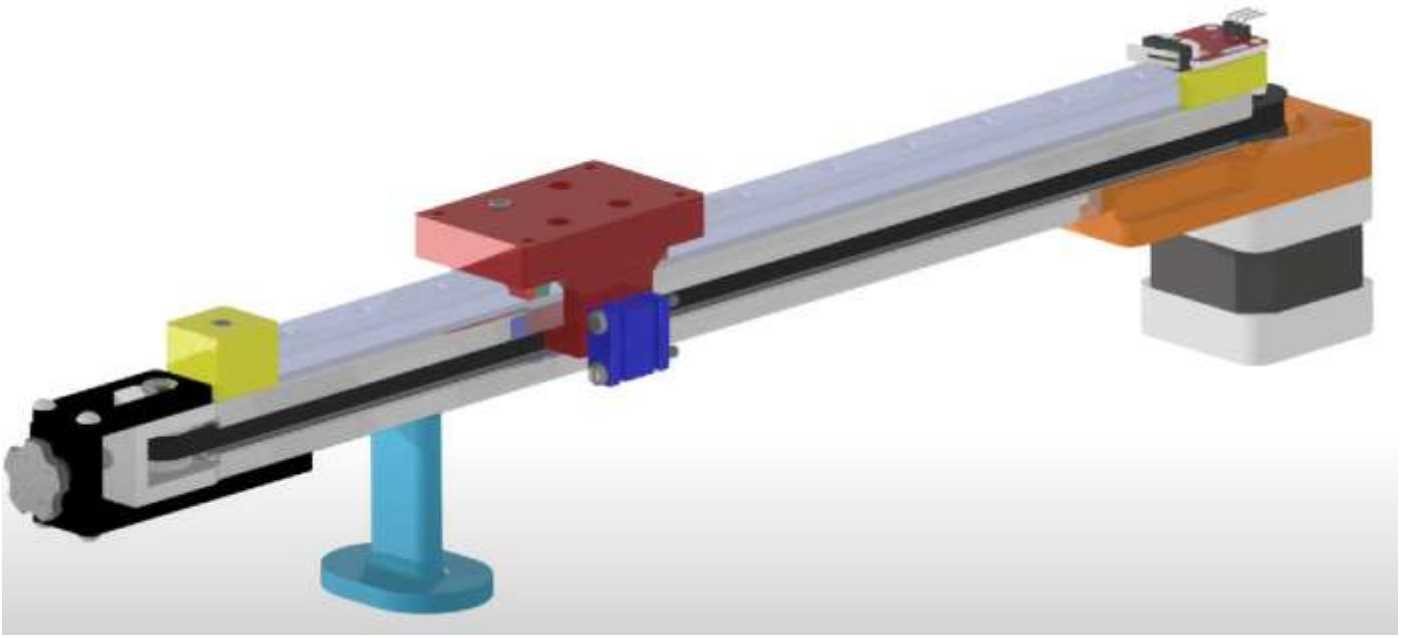


```
#include <LiquidCrystal_I2C.h>
#define I2C_ADDR    0x27
#define LCD_COLUMNS 20
#define LCD_LINES   4
LiquidCrystal_I2C lcd(I2C_ADDR, LCD_COLUMNS, LCD_LINES);

// FC finecorsa posizione HOME
// rotazione oraria --> allontana slitta dal FC
// rotazione oraria --> avvicina slitta al FC
#define DIR_PIN  5  // X
#define STEP_PIN 2  // X
#define EN_PIN 8    // pin abilitazioe driver
#define FC_PIN 11   // driver

int steps_x_round= 200;  // 200 steps al giro
int pot;  // potenziometro per velocità
int delay_step=1000;
int FC_STATE = LOW;  // stato finecorsa
int HOME_STATE=LOW;  // per sapere se sono a HOME
int FLAG_STOP=LOW;
float position;

void setup() {
  Serial.begin(115200);
  lcd.begin(16, 2);

  pinMode(DIR_PIN, OUTPUT);
  pinMode(STEP_PIN, OUTPUT);
  pinMode(EN_PIN, OUTPUT);
  pinMode(FC_PIN, INPUT_PULLUP);
  digitalWrite(EN_PIN, LOW);

  // Attivo LCD
  lcd.init(); lcd.backlight();
  lcd.setCursor(0, 0);  lcd.print("...");

  delay(1000);
}

void loop() {
  //HOME antioraria --> vado alla posizione di riposo
  stepHOME(false, DIR_PIN, STEP_PIN, 20000);  // 200 mm di corsa max
  delay(1000);

  if (FLAG_STOP== LOW) {
    //oraria--> mi spoto di 5mm --> 5/2mm=2.5 * 200 passi=500 step
    step(true, DIR_PIN, STEP_PIN, 500);
    FLAG_STOP= true;
  }

  delay(1000);

}

// dir = true= oraria
void step(boolean dir, byte dirPin, byte stepperPin, int steps)
{
  digitalWrite(dirPin, dir);
  for (int i = 0; i< steps; i++)  {
    FC_STATE = digitalRead(FC_PIN);
    if (FC_STATE == LOW) {
      Serial.println("premuto");
      HOME_STATE= HIGH;
      position= 0.0;
      break;
    }
    if (FC_STATE == HIGH) {
      Serial.println("non premuto");
      digitalWrite(stepperPin, HIGH);
      delayMicroseconds(delay_step);
      digitalWrite(stepperPin, LOW);
```

```
      delayMicroseconds(delay_step);

      position= 2.0 * i / steps_x_round;
      Serial.println("mm "); Serial.println(position);
      lcd.setCursor(0, 0); lcd.print("mm    "); lcd.print(position);
    }
  }
  position= position + 2.0 / steps_x_round;
  Serial.println("mm "); Serial.println(position);
  lcd.setCursor(0, 0); lcd.print("mm    "); lcd.print(position);
}

void stepHOME(boolean dir, byte dirPin, byte stepperPin, int steps)
{
  // SE NON SONO A HOME
  if (HOME_STATE == LOW) {
    digitalWrite(dirPin, dir);
    delay(100);
    for (int i = 0; i< steps; i++)  {
      FC_STATE = digitalRead(FC_PIN);
      if (FC_STATE == LOW) {
        Serial.println("premuto");
        HOME_STATE= HIGH;
        position= 0.0;
        break;
      }
      else if (FC_STATE == HIGH) {
        Serial.println("non premuto");
        digitalWrite(stepperPin, HIGH);
        delayMicroseconds(delay_step);
        digitalWrite(stepperPin, LOW);
        delayMicroseconds(delay_step);
      }
    }
  }
}
```

GUIDA LINEARE CON MOTORE STEPPER E CINGHIA 2GT

La guida è dotata di due finecorsa meccanici “FC” che forniscono segnale “1” quando NON premuti e “0” quando premuti.
All’accensione il blocco mobile deve portarsi alla “HOME” definita da stato “FC_X=0”.
A partire da “HOME” si potranno poi effettuare gli spostamenti assegnati (in mm) dal ciclo proposto.
Il finecorsa “FC_Y” consente di evitare spsotamenti fuori scala.



DRIVER DRV8825 CONTRO A4988

Possiamo controllare i motori passo-passo con altri driver come il DRV8825.

Il principio di funzionamento, le connessioni e la codifica sono quasi le stesse per entrambi questi driver.

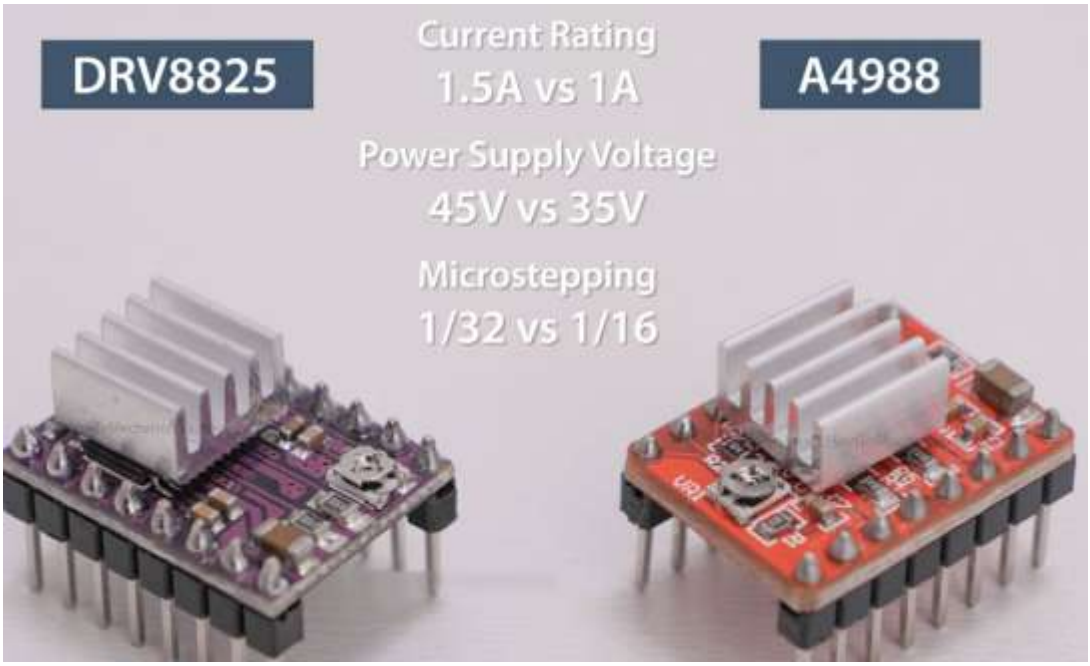
La differenza tra loro sta nelle loro caratteristiche tecniche.

Il DRV8825 è un driver passo-passo di Texas Instruments che può essere utilizzato come sostituto diretto del driver Allegro A4988 poiché le loro connessioni sono le stesse.

Le tre differenze principali tra loro sono che il DR8825

- può fornire più corrente rispetto all'A4988 senza raffreddamento aggiuntivo (1,5 A vs 1 A)
- ha una tensione di alimentazione massima più alta (45 V vs 35 V)
- offre una risoluzione microstepping più elevata (32 vs 16 microstep)

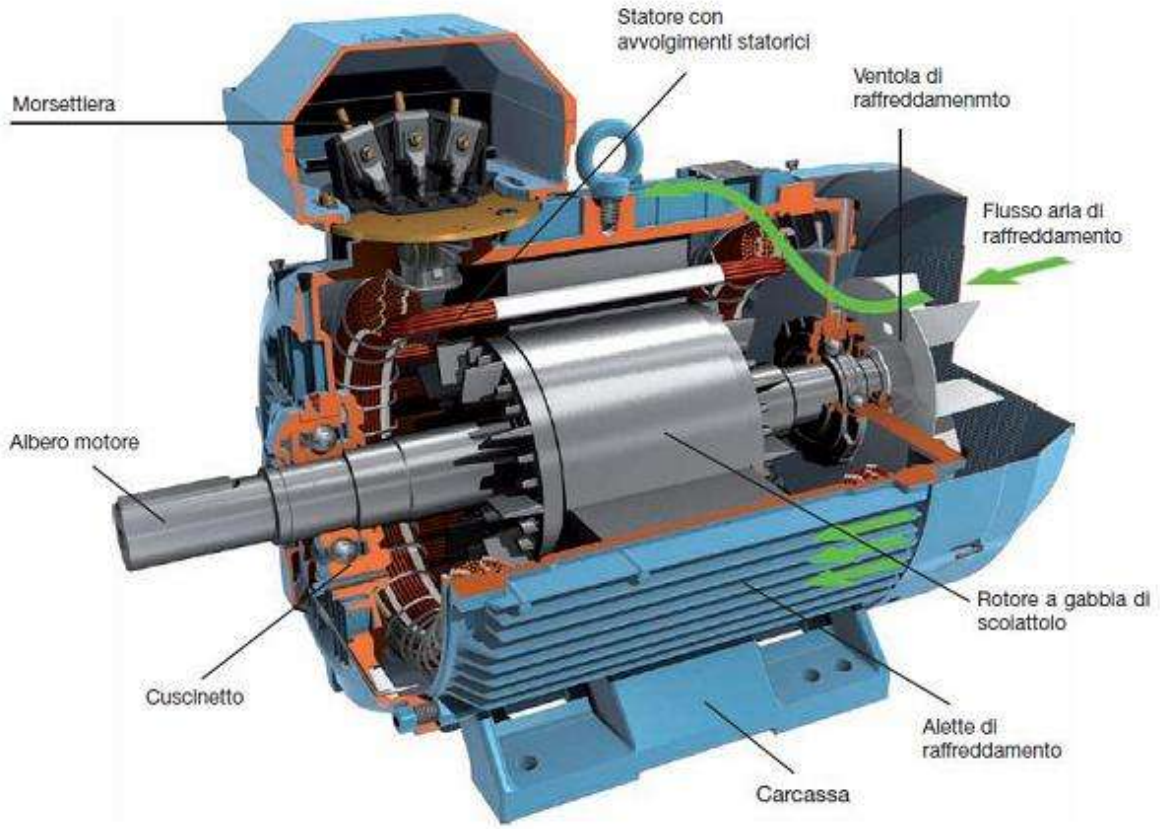
Altri driver più recenti come il TMC2208 presentano caratteristiche ancora migliori e soprattutto una silenziosità in funzionamento decisamente migliore dei precedenti.



MOTORI ASINCRONI 230V / 400V

Il motore asincrono è chiamato anche motore a induzione poiché funziona secondo il principio dell'induzione elettromagnetica. Il motore asincrono è generalmente abbreviato in ASM o IM. Il rotore di un motore asincrono gira più lentamente del campo magnetico rotante presente nello statore, cioè in modo asincrono rispetto allo statore.
La differenza tra la velocità dello statore e la velocità del rotore è chiamata anche scorrimento "s". Se la velocità del rotore è uguale alla velocità dello statore, lo scorrimento è nullo e il motore asincrono non eroga alcuna coppia positiva.
Nel funzionamento come generatore il rotore gira più velocemente del campo rotante dello statore. A causa della differenza di velocità, si genera una coppia negativa che cerca di frenare il rotore.
I motori asincroni che funzionano direttamente con corrente alternata bifase o trifase senza inverter hanno un'efficienza inferiore rispetto ai motori sincroni a magneti permanenti. Tuttavia con un inverter possono raggiungere rendimenti simili.

Un motore asincrono è costituito dai componenti indicati nella figura sottostante:



Spaccato di un motore asincrono

Si distinguono due tipologie principlai di motore asincrono:

- con ***rotore avvolto*** chiamato anche "***ad anello scorrevole***"
- con ***rotore in cortocircuito*** o più comunemente definito come rotore "***a gabbia di scoiattolo***".

La principale differenza tra i due tipi risiede proprio nella struttura del rotore. Lo statore è molto simile per entrambi.

Per la tipologia ***"ad anello scorrevole"***, il rotore è costituito da avvolgimenti veri e propri come quelli dello statore, presenta una struttura più complessa (spazzole che strisciano sul rotore con possibile interposizione di resistenze per il controllo della fase di avviamento), necessità di manutenzione periodica e dimensioni d'ingombro elevate.

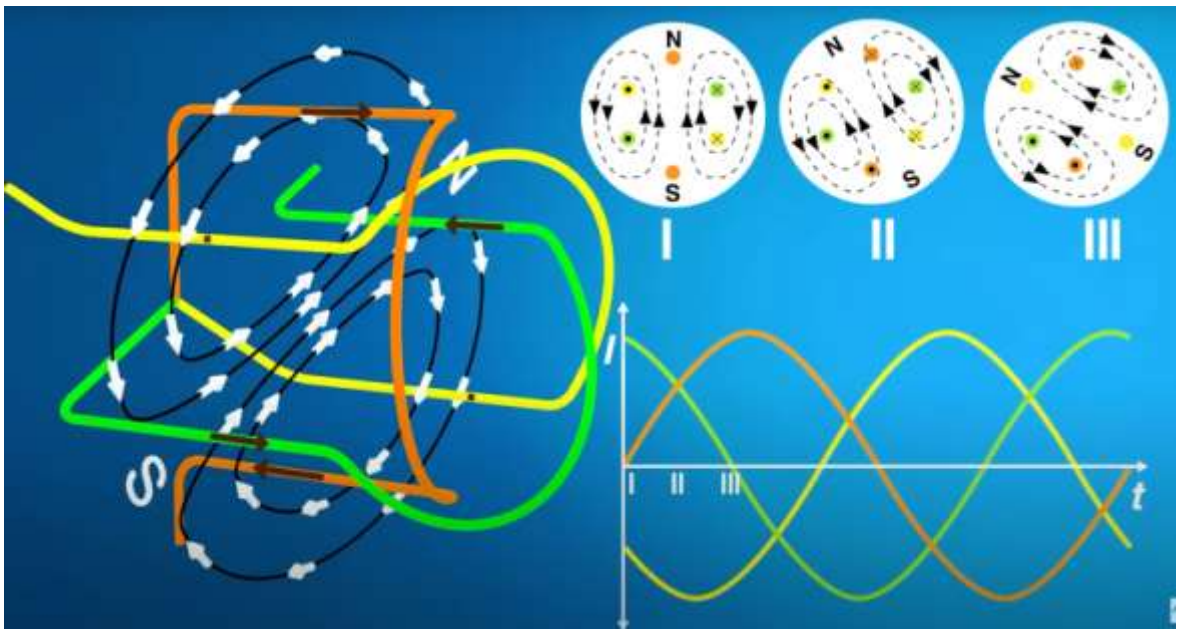
La tipologia ***"a gabbia di scoiattolo"*** ha invece un rotore costituito da sbarre chiuse in cortocircuito che garantiscono una maggiore semplicità costruttiva, robustezza ed economicità.
Grazie allo sviluppo dell'elettronica di controllo che permette la regolazione della velocità in modo molto semplice ed efficace, tutte quelle applicazioni che vedevano l'impiego di motori in corrente continua (più facilmente regolabili in velocità con le vecchie tecnologie) hanno lasciato il posto ai motori asincroni, in particolare a quelli a gabbia di scoiattolo, che vengono comunemente utilizzati per comandare gli azionamenti industriali più svariati.

PRINCIPIO DI FUNZIONAMENTO DEL MOTORE A INDUZIONE

Immagini tratte dal video https://www.youtube.com/watch?v=NoWFPuPnQBs by JAES Company.

Lo statore costituito da lamelle impaccate per aumentare intesità del campo magnetico statorico.

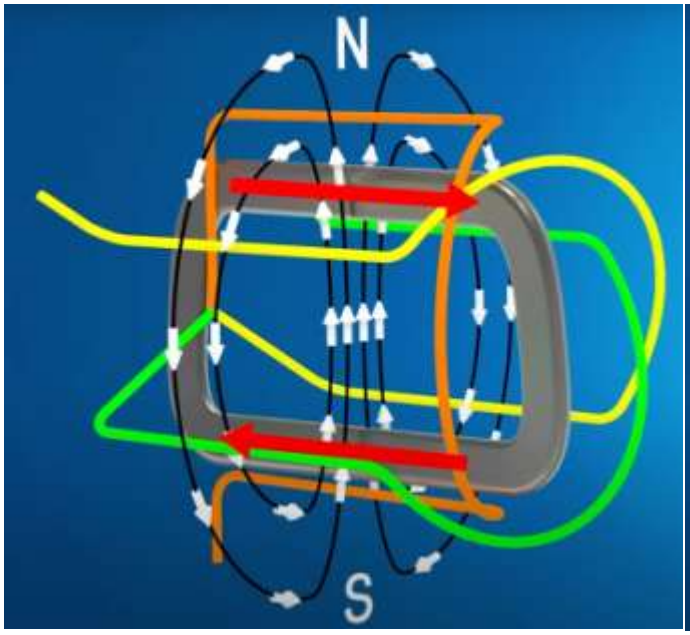
Nello statore del motore vengono fissati dei cavi di alimentazione.
Abbiamo tre terminali distinti: giallo, verde e arancio alimentati dalla tensione trifase.

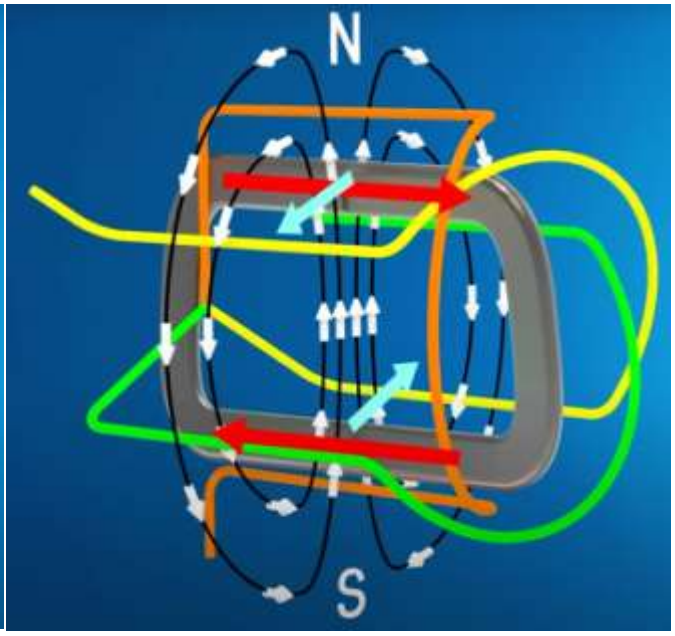


I cavi (nella realtà bobine) sono sfalsate di 120° come la tensione alternate che li alimenta.

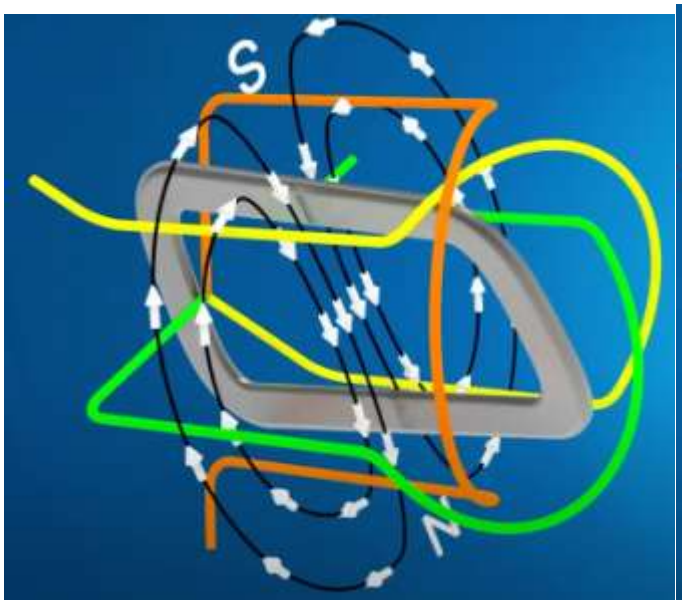
Le tensioni alternate applicate generano un campo magnetico statorico ROTANTE (velocità di sincronismo).

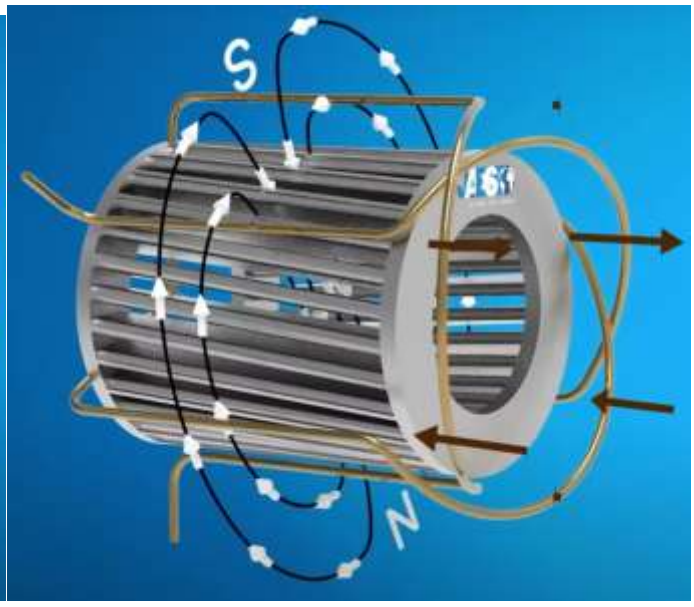
All’interno dello statore trova spazio una spira chiusa metallica. A causa del campo magnetic statorico rotante viene indotta una corrente elettrica che a sua volta genera un campo magnetico nella spira che inizierà a ruotare per raggiungere la velocitò del campo magnatico statorico.



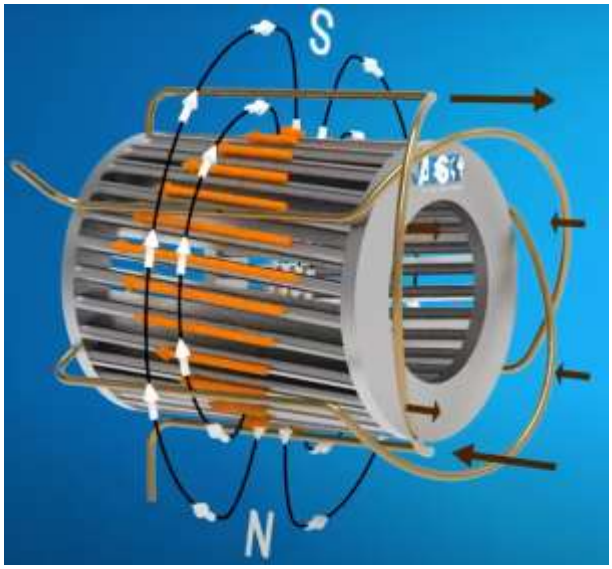


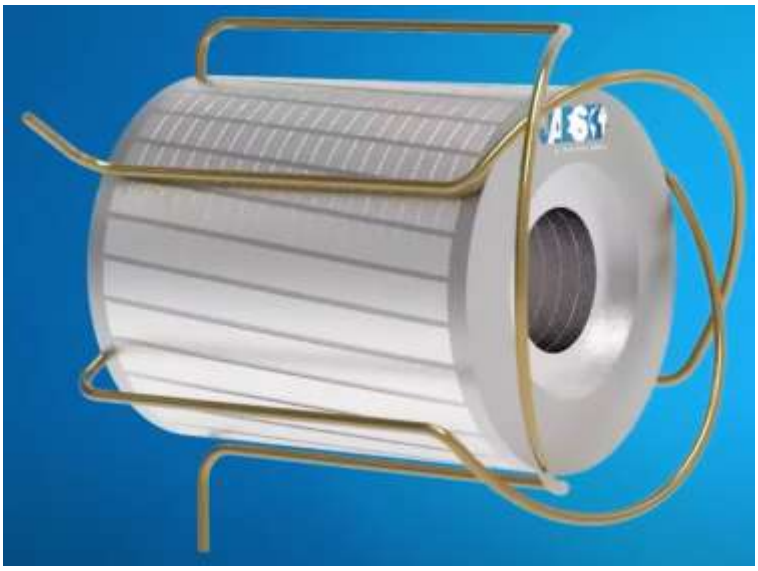
Al posto della singola spira si impiega comunemente un rotore a “gabbia di scoiattolo” costituito da barre collegate alle estermità a dei dischi forati



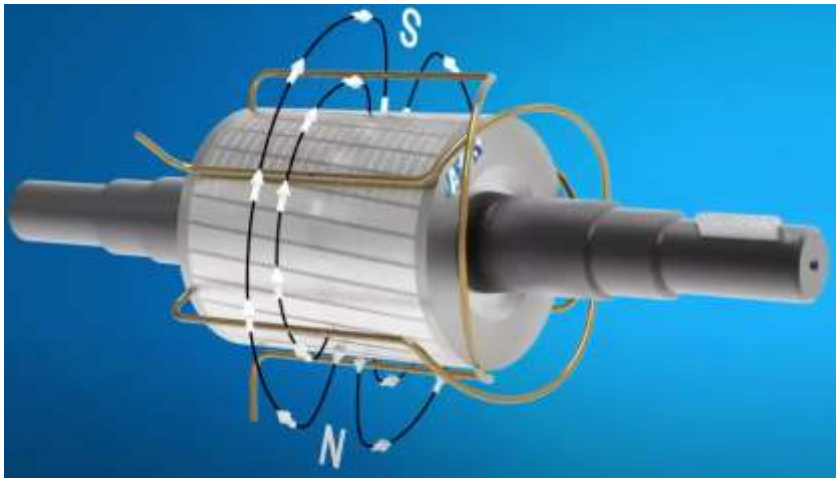


La corrente elettrica pulsante scorre nelle barre del rotore come indicato in figura.
Per aumentare l'efficienza le barre sono inserite in un pacco di lamelle.

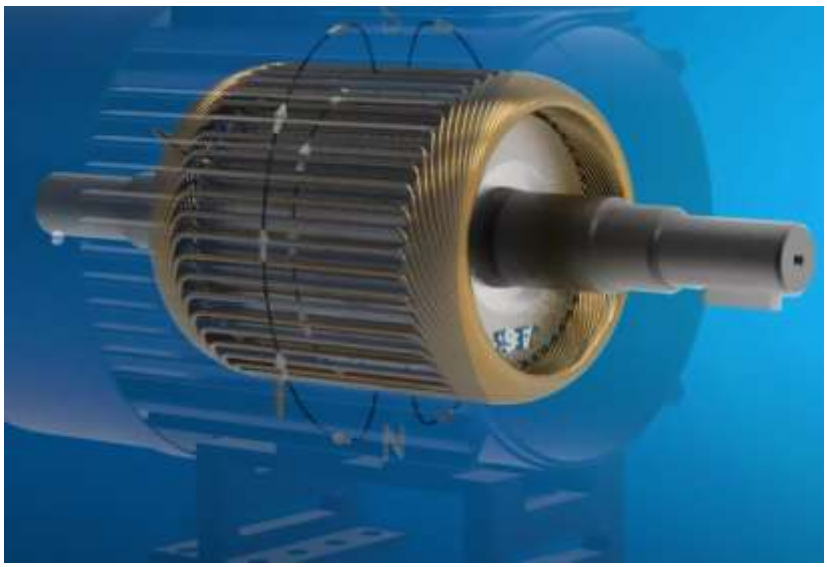




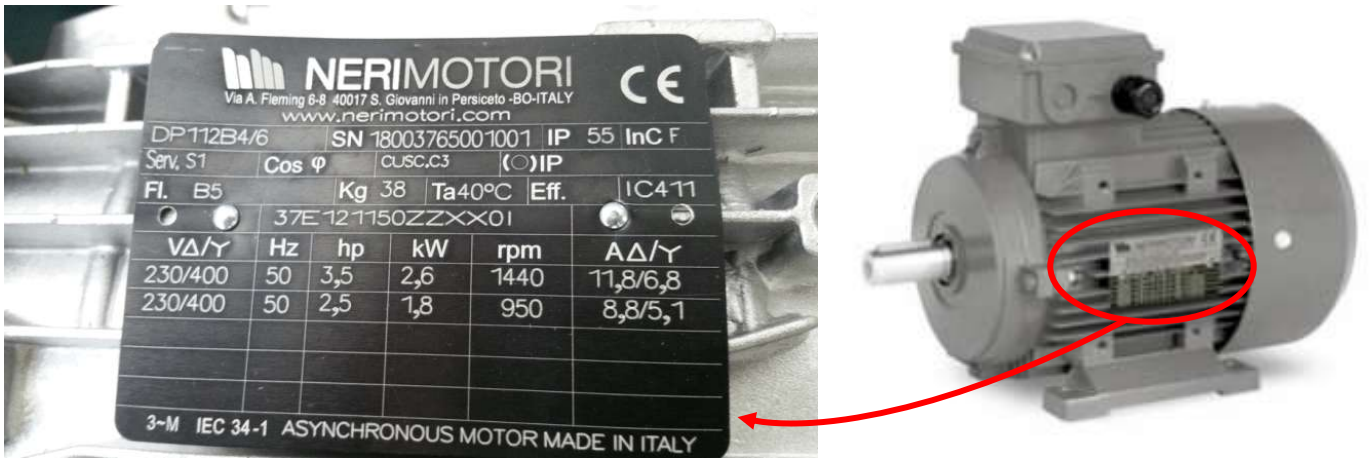
Il rotore viene opportunamente fissato su un albero motore.



I cavi dello statore assumono la forma di bobine con molteplici spire.

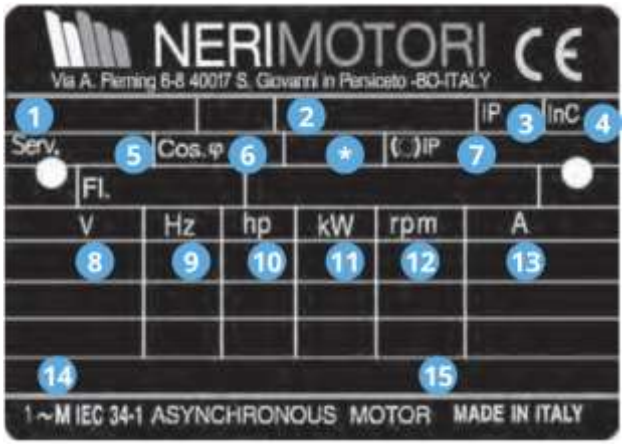


DATI DI TARGA DI UN MOTORE AC



SIGNIFICATO DEI DATI

Il numero in basso a sinistra “3-M” indica TRIFASE (380V) mentre “1-M” indica MONOFASE (220V).



1- Tipo di motore (T trifase AT autofrenante DP doppia polarità ME monofase con condensatore elettronico…).
Grandezza della cassa del motore (da 56 a 355). Numero di poli motore (2-4-6-8-4/6-4/8…)
es: DP112B4/6 motore doppia polarità grandezza 112B a 4 e 6 poli.
2- Matricola o Serial number assegnato dal costruttore
3- Grado di protezione da agenti esterni, IP 55 è standard
4- Classe di isolamento degli avvolgimenti:
in Cl.F temperatura massima ammissibile 165°, in Cl.H temperatura massima ammissibile 180°
5- Tipo di servizio in funzionamento:
S1 servizio continuo – S2 servizio di durata limitata – S3 servizio intermittente periodico
6- Fattore di potenza
7- Dati specifici del freno se presente:
DC freno in corrente continua – AC freno in corrente alternata…
8- Tensione di alimentazione del motore, voltaggi variabili a seconda del Paese di utilizzo
9- Frequenza: 50 o 60Hz
10- Potenza del motore espressa in hp
11- Potenza del motore espressa in kW
12- Giri del motore al minuto
13- Corrente nominale – assorbimenti
14 e 15- Dati del condensatore

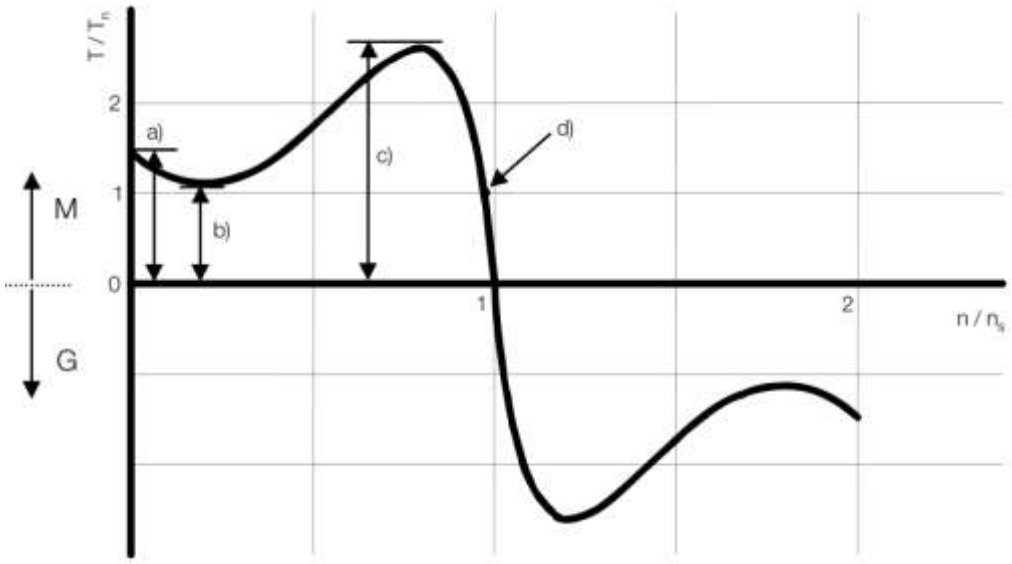
* Altri dati particolari del motore: es. C3 cuscinetti C3 – T motore tropicalizzato – 1S motore con 1 scaldiglia anticondensa – VL motore con volano – A motore con fori anticondensa…

CARATTERISTICHE DEL MOTORE A INDUZIONE AC

I motori a induzione convertono l'energia elettrica in energia meccanica. La conversione dell'energia si basa sull'induzione elettromagnetica. Il fenomeno dell'induzione determina lo scorrimento "s" del motore.
Tale scorrimento viene spesso definito come il punto nominale del motore (frequenza (fn), velocità (nn), coppia (Tn), tensione (Un), corrente (In) e potenza (Pn)).

Al punto nominale: $s_n = \frac{n_s - n_n}{n_s} * 100\ \%$ con ns la velocità sincrona: $n_s = \frac{2 * f_n * 60}{\text{numero poli}}$

Quando il motore è collegato a un alimentazione con tensione e frequenza costanti, ne risulta una curva della coppia:



Tipica curva coppia/velocità di un motore a induzione collegato alla rete di alimentazione (D.O.L., Direct-On-Line). Nell'immagine a) è la coppia bloccata del rotore, b) è la coppia d'arresto, c) è la coppia massima del motore, T_{max} e d) è il punto nominale del motore.

La coppia massima di un motore a induzione standard (Tmax, detta anche coppia massima in esercizio continuo o coppia alla tensione di scarica) è normalmente pari a 2-3 volte la coppia nominale.

La coppia massima è disponibile con scorrimento smax, che è maggiore dello scorrimento nominale.
Per utilizzare in modo efficiente un motore a induzione, lo scorrimento del motore dovrebbe rientrare nel campo (- smax ... smax), che si ottiene controllando la tensione e la frequenza.

Il controllo può essere effettuato utilizzando un ***convertitore di frequenza***.
I convertitori di frequenza limitano normalmente la coppia massima disponibile al 70% di Tmax.

Il campo di frequenza al di sotto della frequenza nominale è denominato ***campo a flusso costante***.
La coppia massima di un motore a induzione è proporzionale al quadrato del flusso magnetico (Tmax ~ ψ 2).
Ciò significa che la coppia massima è tendenzialmente costante in corrispondenza del campo di flusso costante.

Al di sopra della frequenza/velocità nominali, il motore funziona nel range di indebolimento di campo.
Nel range di indebolimento di campo, il motore può funzionare a potenza costante, e pertanto il range di indebolimento di campo viene talvolta definito campo di potenza costante.

La coppia massima di un motore a induzione è proporzionale al quadrato del flusso magnetico (Tmax ~ ψ 2).
Ciò significa che la coppia massima è tendenzialmente costante in corrispondenza del campo di flusso costante.
Al di sopra del punto di indebolimento di campo, la riduzione della coppia massima è inversamente proporzionale al quadrato della frequenza.

CORRENTE ASSORBITA DAL MOTORE AC

La corrente del motore a induzione comprende due componenti: corrente reattiva (Isd) e corrente attiva (Isq).
La componente reattiva è legata alla magnetizzazione che si genera nel motore, mentre la corrente attiva è quell ache genera la coppia motrice del motore.

La corrente totale assorbita dal motore è pari a: $i_m = \sqrt{i_{sd}^2 + i_{sq}^2}$

Si può riscontrare che a coppia motore uguale a zero, la componente di corrente attiva è uguale a zero.
Con valori di coppia più elevati, la corrente del motore diventa quasi proporzionale alla coppia.

La corrente di magnetizzazione può essere calcolata con la: $I_{sd} = I_n \sin(\varphi_n)$ dove φ_n è il ***fattore di potenza*** del motore

Una buona approssimazione della corrente totale del motore è: $i_m = \frac{T_{load}}{T_n} * I_n$ quando 0,8 * Tn ≤ Tload ≤ 0.7 * Tmax

ES: Il motore da 15 kW è caratterizzato da una corrente nominale di 32 A e da un fattore di potenza di 0,83.
Qual è approssimativamente la corrente di magnetizzazione del motore al punto nominale?
Qual è la corrente approssimativa totale al 120 % della coppia al di sotto del punto di indebolimento di campo?

$$I_{sd} = I_n \sin(\varphi_n) = 32 * \sqrt{1 - 0.83^2} = 17.8\ A$$

$$i_m = \frac{T_{load}}{T_n} * I_n = 1.2 * 32\ A = 38.4\ A$$

Al di sopra del punto di indebolimento del campo elettromagnetico le componenti di corrente dipendono anche dalla velocità.

POTENZA DEL MOTORE A INDUZIONE AC

La potenza (di uscita) meccanica del motore può essere calcolata partendo dalla velocità e dalla coppia utilizzando le seguenti formule:

$$P_{out}\,[W] = T\,[Nm] * \omega\,[rad/s]$$

$$P_{out}\,[kW] = \frac{T\,[Nm] * n\,[rpm]}{9550}$$

La potenza di ingresso del motore può essere calcolata a partire dalla tensione U, dalla corrente I e dal fattore di potenza:
$P_{in} = \sqrt{3} * U * I * \cos(\varphi)$

L'efficienza del motore è il valore della potenza di uscita diviso per la potenza di ingresso: $\eta = \frac{P_{out}}{P_{in}}$

ES: La potenza nominale del motore è pari a 15 kW e la velocità nominale a 1.480 giri/min. Qual è la coppia nominale Tn?

$$T_n = \frac{9550 * 15}{1480}\ Nm = 96.8\ Nm$$

ES: Qual è l'efficienza nominale di un motore da 37 kW (Pn = 37 kW, Un =380 V, In =71 A e cos(φn) = 0,85)?

$$\eta_n = \frac{P_{out}}{P_{in}} = \frac{P_n}{\sqrt{3} * U_n * I_n * \cos(\varphi_n)} = \frac{37000}{\sqrt{3} * 380 * 71 * 0.85} \approx 0.931$$

CAPACITÀ DI CARICO TERMICO DEL MOTORE AC

La capacità di carico termico del motore indica la capacità del motore di mantenere una coppia motrice a lungo termine senza surriscaldarsi e danneggiarsi.

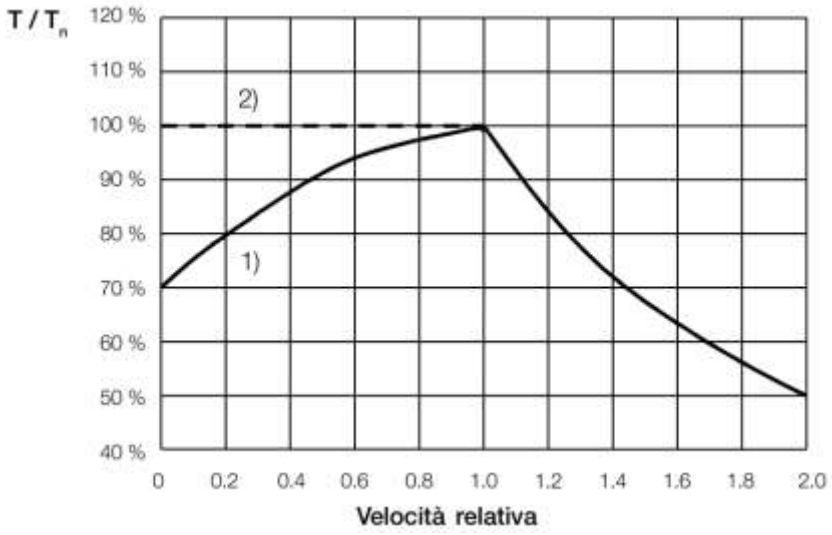
I motori a induzione standard sono generalmente dotati di ventilazione propria (vedi figura).

A causa di questa caratteristica, la capacità di carico termico del motore decresce proporzionalmente alla diminuzione della sua velocità.
Questo tipo di comportamento limita la coppia continua disponibile alle basse velocità.

I motori con sistema di raffreddamento separato possono essere caricati anche alle basse velocità.

Spesso il sistema di raffreddamento è dimensionato affinchè l'effetto raffreddante sia lo stesso di quello al punto nominale.

Sia con sistemi di raffreddamento propri che separati, la coppia è termicamente limitata nel range di indebolimento di campo.



La capacità di carico tipica di un motore a induzione a gabbia di tipo standard in un azionamento controllato in frequenza

1) senza sistema di raffreddamento separato

2) con sistema di raffreddamento separato.

E'possibile sovraccaricare un motore in c.a. per brevi periodi di tempo senza surriscaldarlo.
Il sovraccarico di breve termine è prevalentemente limitato da Tmax (verificare i margini di sicurezza).

In termini generici, la capacità di sovraccarico termico di breve termine del convertitore di frequenza è spesso più critica di quella del motore.

Il tempo di rialzo termico del motore normalmente è superiore ai 15 minuti (motori di piccole dimensioni) fino a qualche ora (motori più grandi) in base alle dimensioni del motore.

Il tempo di rialzo termico del convertitore di frequenza (normalmente di pochi minuti) è specificato nei manuali dei singoli prodotti.

STATORE DI UN MOTORE ASINCRONO

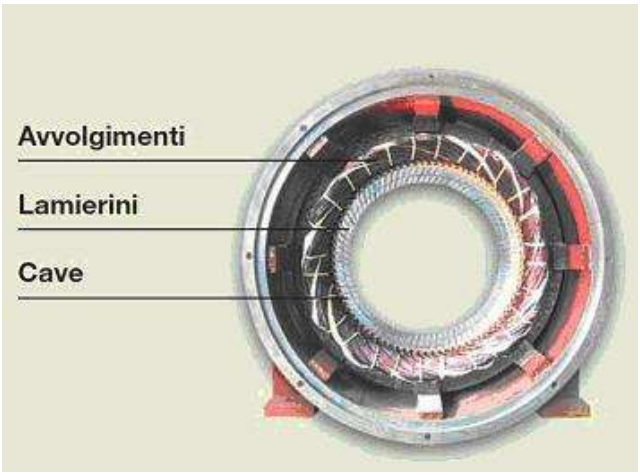
Questo componente può essere definito come l'insieme delle parti fisse e costituisce la parte del circuito magnetico che contiene gli avvolgimenti induttori alloggiati in apposite cave in esso ricavate in corrispondenza della sua superficie interna.

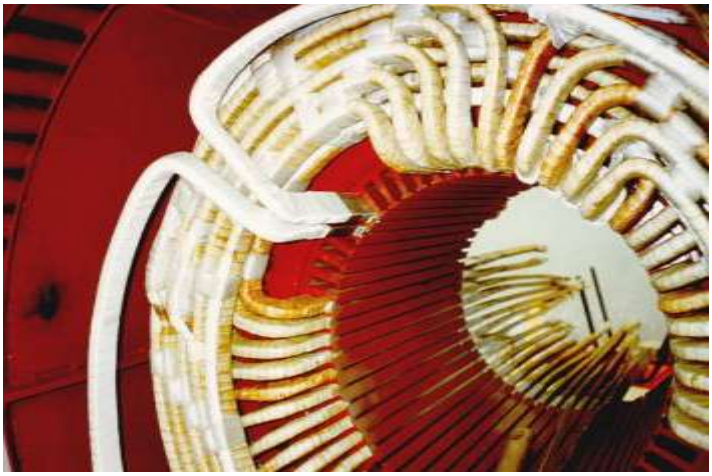
La struttura dello statore è la stessa per entrambe le tipologie di motore AC.

Per condurre il flusso magnetico nel motore elettrico, lo statore e il rotore sono costituiti da diversi strati di lamierino elettrico, solitamente di 0,5 mm di spessore. Quanto più sottile è il foglio elettrico, tanto minori sono le perdite per correnti parassite nel motore elettrico e maggiore è la sua efficienza.

Lo statore porta gli avvolgimenti in cui scorre la corrente trifase.

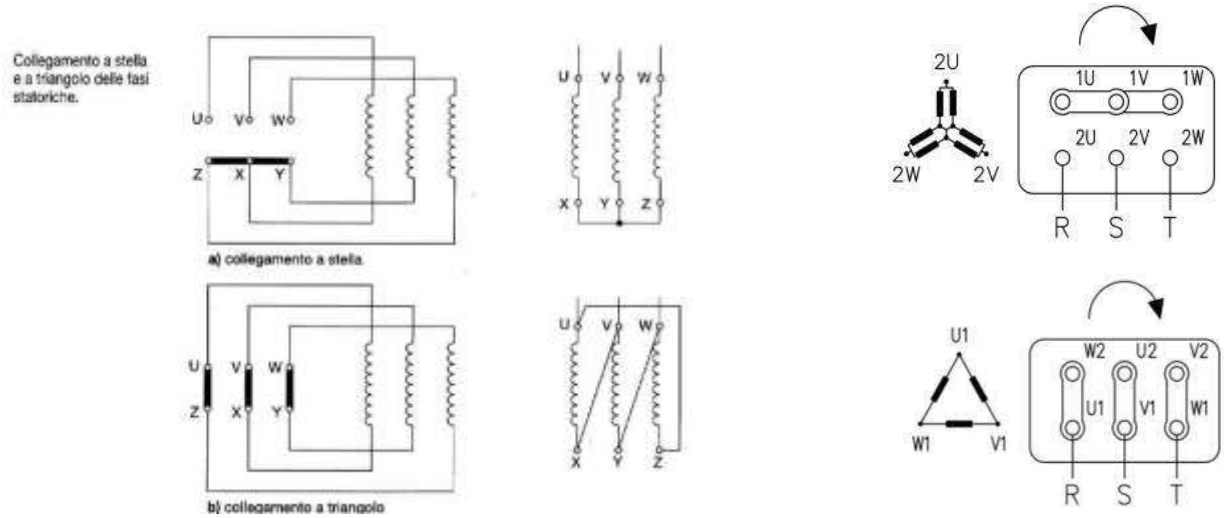
Di norma, lo statore ha tre fasi del motore, che possono essere collegate in configurazione a stella o a triangolo.
Il rotore contiene barre conduttrici o avvolgimenti in cortocircuito, a seconda del tipo di motore asincrono.



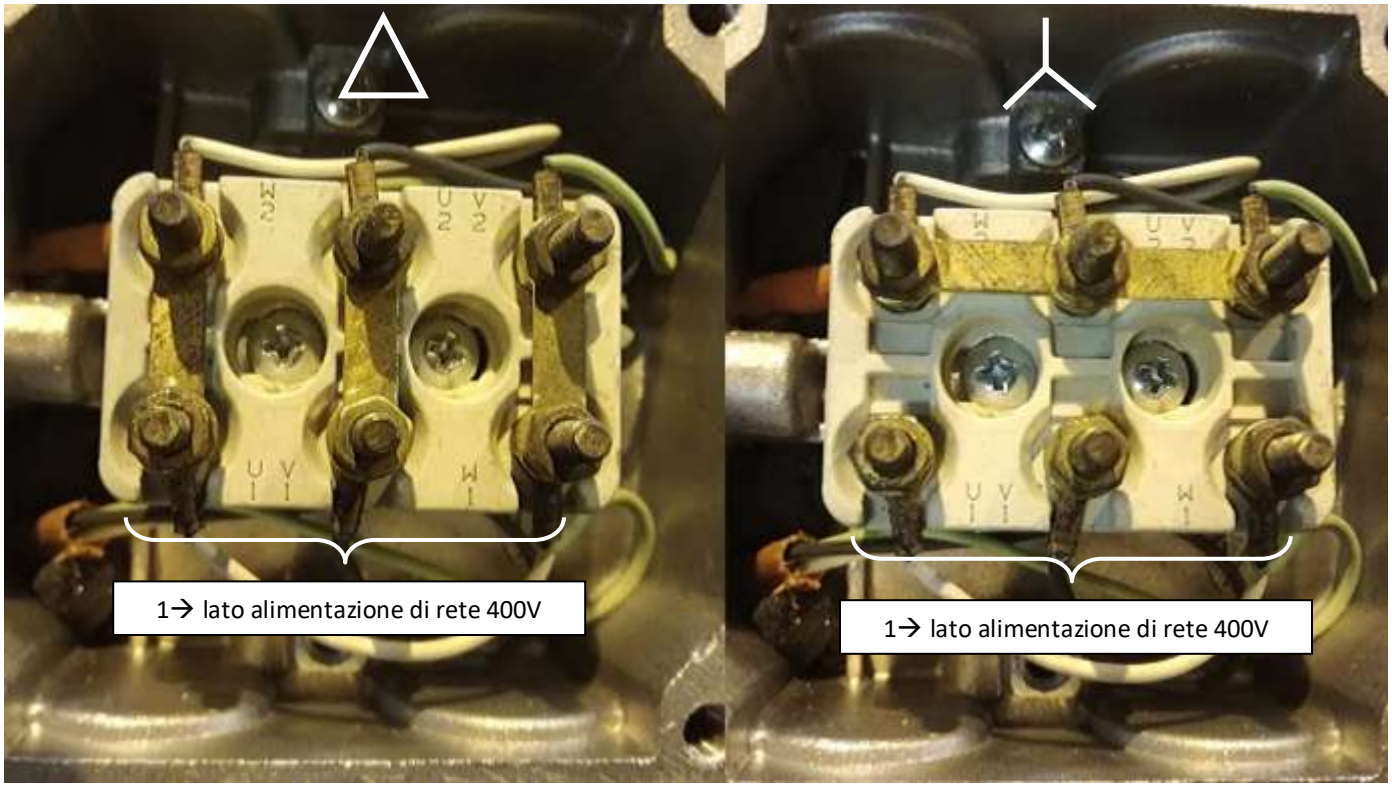


Gli avvolgimenti statorici trifase possono essere collegati a stella oppure a triangolo se il motore è dotato di morsettiera con 6 morsetti. In questo modo è possibile alimentare lo stesso motore con tensioni trifase di rete differenti. Infatti la condizione necessaria al funzionamento del motore è che ciascun avvolgimento statorico sia sottoposto alla sua tensione nominale.

Perciò collegare a stella con tensione concatenata (fase-fase) 400V o a triangolo con tensione concatenata (fase-fase) 230V sarà del tutto equivalente in quanto gli avvolgimenti saranno sempre sottoposti ad una tensione di 230V (400V/1,73).



Collegamenti stella/triangolo del motore asincrono

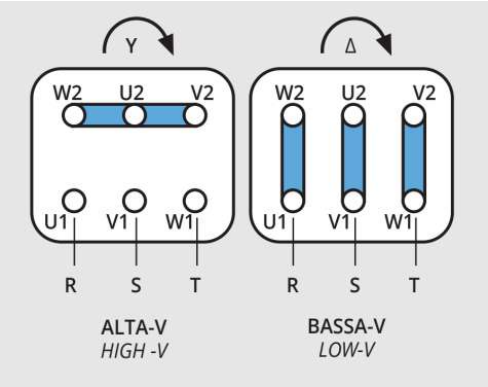


Motore trifase ad una velocità	
Collegamento a stella	Collegamento a triangolo
Motore trifase a due velocità, unico avvolgimento (Dahlander)	
Bassa velocità	Alta velocità
Motore trifase a due velocità, due avvolgimenti separati	
Bassa velocità	Alta velocità
Motore monofase a condensatore permanente	
Rotazione in un senso	Rotazione nel senso contrario

COLLEGAMENTO A STELLA E A TRIANGOLO

Nel collegamento ***a triangolo*** i tre avvolgimenti sono collegati tra loro (un capo dell'avvolgimento R sarà collegato al capo di S; l'altro capo S sarà collegato al capo T e l'altro capo T all'altro capo S).
Se si prova a disegnare questo collegamento, si ottiene, appunto, un triangolo.
A livello elettrico, ad ognuno dei tre capi di giunzione si collega, sempre singolarmente, ognuna delle 3 fasi.
In questa maniera la differenza di potenziale ai capi di ognuno degli avvolgimenti sarà di 380 V.

Nel collegamento ***a stella*** si collega ogni fase (singolarmente) ai tre capi di tre avvolgimenti (resistivi o induttivi); gli altri tre capi si collegano tra loro per formare il centro stella.
Se si prova a disegnare lo schema di collegamento, si ottiene una stella a tre punte.
A livello elettrico, il centro stella ha potenziale zero mentre la differenza di potenziale tra il centro stella ed ognuna delle fasi (e quindi la differenza di potenziale ai capi di ogni avvolgimento) sarà, nel caso di trifase a 380V, di 220V.
Generalmente il collegamento triangolo è quello che corrisponde alla potenza nominale del motore.



- Le fasi RST sono quelle della linea di alimentazione a monte del motore.
- Le fasi U V W sono quelle all'entrata delle fasi del motore;
- Le fasi X Y Z sono quelle finali (lato centro stella se c'e') delle fasi del motore.

OSSERVAZIONI

Cosa cambia collegare un motore trifase a stella o a triangolo?
La corrente assorbita nel collegamento a triangolo è 3 volte quella assorbita nel collegamento a stella.
Di conseguenza anche la coppia motrice è 3 volte maggiore. Gli avvolgimenti di un motore progettati per una tensione nominale di 220 V, non possono essere collegati a triangolo in un sistema trifase a 380 V, ma solo a stella; possono ovviamente essere collegati a triangolo in un sistema trifase di 220 V.
Un collegamento a stella ha minore assorbimento?
Non bisogna illudersi che un motore, le cui caratteristiche sono riferite al collegamento a triangolo, assorba meno corrente a carico con il collegamento a stella. Se il carico è immutato ed il motore è in grado di avviarsi anche collegato a stella, a regime funzionerà con uno scorrimento più elevato ma (ATTENZIONE!) con un surriscaldamento che può essere eccessivo.
Il numero di giri varia se collego a triangolo o a stella?
A vuoto la velocità del motore è la stessa in entrambi i collegamenti.
A carico invece occorre fare le seguenti considerazioni.
Con il collegamento a stella la coppia si riduce ad un terzo di quella a triangolo, a parità di tensione di linea.
Se la coppia del carico è costante (esempio: motore di un argano che solleva un dato peso), il motore deve rallentare per aumentare la coppia. Aumenta quindi lo scorrimento, aumentano le perdite ed il motore si scalda di più.
Nel caso in cui la coppia diventasse insufficiente, potrebbe capitare che il motore addirittura si fermi.
Se cambio il collegamento nella morsettiera Da stella a triangolo cosa succede?
Se il collegamento nella morsettiera viene modificato per essere da stella a triangolo, aumenta la tensione ai capi di ogni avvolgimento del 73% per cui è disponibile una coppia massima 3 volte maggiore.
Quando posso utilizzare il collegamento a triangolo? Perchè?
Se la tensione concatenata non è superiore alla tensione nominale dell'avvolgimento è possibile utilizzare il collegamento a triangolo, altrimenti il motore rischierebbe di bruciarsi.
Perchè se inverto le fasi in un motore lo stesso inverte il senso di rotazione?
Perché cambia il senso di rotazione del campo rotante.
Perchè il collegamento stella triangolo viene preferito rispetto al collegamento diretto?
Perché si ritiene troppo elevata, quindi dannosa, la corrente di avviamento diretta.
Ma, nella maggior parte dei casi, è un timore infondato se il motore è alimentato direttamente dalla rete.
Con avviamento stella-triangolo il motore si scalda di più che in modo diretto?
Dipende dalla durata dell'avviamento.La coppia accelerante si riduce a stella, quindi aumenta il tempo di avviamento.

ROTORE DEL MOTORE ASINCRONO

Il rotore viene posizionato all'interno dello statore e costituisce il circuito indotto della macchina.

Per un motore a "***gabbia di scoiattolo***", il rotore è costituito da un sistema di sbarre conduttrici (rame o alluminio) coassiali all'asse di rotazione e pressofuse direttamente nelle cave ricavate lungo tutta la periferia esterna del nucleo ferromagnetico.

Le sbarre vengono chiuse in cortocircuito da due anelli conduttori posti agli estremi che costituiscono anche un fissaggio meccanico per le sbarre stesse.
Si ottiene così un rotore estremamente compatto e robusto, al quale si fissa anche l'albero del motore.

Il campo magnetico indotto che costituisce il principio di funzionamento del motore porta quindi in rotazione l'albero del motore convertendo così energia elettrica in meccanica.

Il rotore a gabbia di scoiattolo è il più utilizzato perché non ha anelli di scorrimento e quindi ha una durata maggiore.
Inoltre, la produzione del rotore è molto più economica.

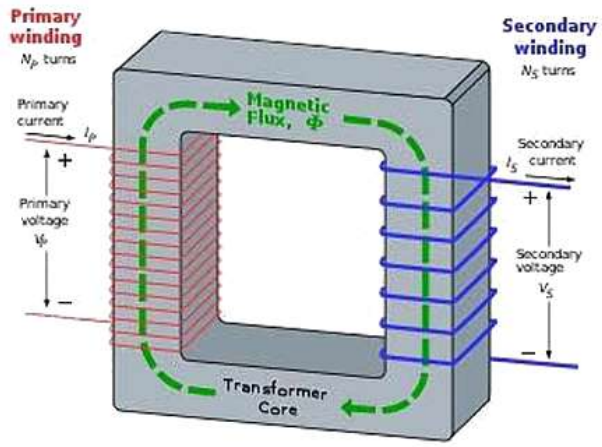
In un rotore "***ad anello scorrevole***", il rotore è costituito da avvolgimenti anziché da barre.

Gli avvolgimenti non sono cortocircuitati nel rotore, ma sono condotti all'esterno tramite anelli di scorrimento e cortocircuitati tramite resistenze aggiuntive.

Il flusso di corrente nel rotore può essere influenzato da resistenze esterne al motore elettrico.

VELOCITA' DI ROTAZIONE DEL MOTORE A INDUZIONE AC

I motori CA funzionano tramite il principio dell'induzione (come per trasformatori).
Quando la la corrente scorre in una bobina si crea un campo magnetico.
Il campo magnetico prodotto può indurre una tensione e una corrente in una bobina vicina.



La tensione d'ingresso, posta ai capi dell'avvolgimento primario, produce un flusso magnetico che va a concatenarsi con l'avvolgimento secondario, producendo su quest'ultimo una tensione dipendente dal numero di spire dei due avvolgimenti.
Ad esempio, se l'avvolgimento primario conta 10000 spire, mentre quello secondario solo 1000, il rapporto tra le tensioni sarà 1/10, quindi applicando sul primario la nostra tensione da 220V avremo sul secondario 22V.
Altra caratteristica interessante è che la potenza assorbita dal secondario è la stessa che viene erogata dal primario, ne consegue che se l'uscita è aperta, in ingresso non viene assorbita potenza (in realtà c'è una piccola potenza dissipata) anche se apparentemente i due cavi sono cortocircuitati da un conduttore!
Inoltre, se il secondario eroga 220W, quindi 10A, avremo che nel primario circola solo 1A, motivo per cui nei trasformatori la bobina con meno spire utilizza un conduttore dalla sezione maggiore.

Questo fenomeno di induzione non è limitato solo a una bobina vicina. Può verificarsi in qualsiasi oggetto metallico.
Nel caso di un motore a corrente alternata, il campo magnetico creato nelle bobine dello statore può indurre una tensione e una corrente nelle barre conduttive del rotore. Quella tensione e quella corrente produrranno il proprio campo magnetico, che quindi interagirà con il campo che lo ha prodotto.

La velocità alla quale il campo magnetico si muove (ruota) attorno allo statore è nota come velocità sincrona Ns e dipende dalla frequenza CA e dal numero di poli nello statore. È data da

Ns = 120 f / P dove: Ns = velocità sincrona, f = frequenza di rete Hz, P = numero di poli (per fase) nello statore

Per un motore a due poli funzionante a 60 Hertz, la velocità sincrona è di 3.600 giri/min. A 50 Hz è di 3.000 giri/min. .

Se si aumenta il numero di poli a quattro, la velocità si riduce a 1.800 giri/min a 60Hz e 1.500 a 50Hz (la velocità di sincronismo si dimezza poiché il campo magnetico percorre solamente 180° nello spazio dei 360° dell'onda sinusoidale).

La velocità alla quale ruota il rotore è nota come velocità di scorrimento Nr e sarà sempre inferiore alla velocità sincrona nello statore. La ragione di ciò è perché nessuna tensione e corrente viene indotta nel rotore quando viaggiano in modo sincrono.

La velocità di slittamento effettiva dipende dal design del motore e varia a seconda del modello e della potenza.

La velocità del rotore Nr in condizioni nominali è sempre minore di un 3-6% di quella di sincronismo:

è il fenomeno dello scorrimento (slip) che consente la produzione della coppia.

Dalla formula che definisce lo scorrimento è possibile esprimere la velocità di rotazione effettiva del rotore:

s = (Ns -Nr) / Ns

dove s è lo scorrimento, Ns è la velocità di sincronismo e Nr è la velocità reale alla quale ruota il rotore.

Per i motori a potenza frazionaria a pieno carico, la velocità di slittamento può arrivare fino al 95 percento di Ns , mentre i modelli con potenza superiore possono funzionare al 99 percento di Ns .

Come discusso nella mia serie sull'alimentazione CA, l'onda sinusoidale CA monofase raggiunge la sua tensione di picco due volte durante un ciclo di 360 gradi e questi picchi si verificano a intervalli di 180 gradi. In un circuito trifase, la fase 2 ritarda la fase uno di 120 gradi e la fase 3 ritarda la fase due di 120 gradi.

Quando tutte e tre le fasi scorrono insieme, la tensione raggiunge picchi ogni 60 gradi.

Questa relazione è illustrata nella Figura 2. Le frecce mostrano la separazione di 120 gradi delle tre fasi e le linee verticali colorate mostrano i picchi di tensione di fase ogni 60 gradi. Questa relazione di picco non solo fornisce un'alimentazione più uniforme, ma può anche produrre un campo magnetico rotante nello statore di un motore trifase.

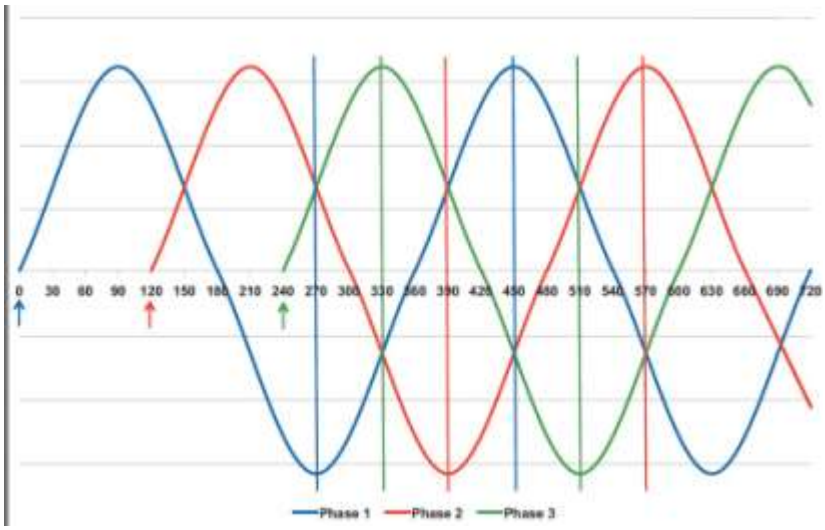


Figura 2. Onda sinusoidale del motore AC trifase e picchi di tensione

La Figura 3 mostra il posizionamento dei poli per un motore trifase a due poli.

Ci sono un totale di sei poli o due poli per fase.
I poli della Fase 1 si trovano a 360 e 180 gradi mentre i poli della Fase 2 sono a 300 e 120 gradi.
I poli della Fase 3 si trovano a 60 e 240 gradi. Il risultato è un totale di sei poli distanziati di 60 gradi l'uno dall'altro.
Questa separazione di 60 gradi non è una coincidenza.
Viene fatto appositamente per sfruttare la separazione di 60 gradi dei picchi di tensione trifase.

Perché i poli di fase si trovano in questa particolare sequenza?
Il polo primario della Fase 2 è a sinistra del primario della Fase 1 e il polo primario della Fase 3 è a destra.
Con riferimento alla Figura 2, il picco che segue il picco della Fase 1 è la Fase 3 e il picco successivo è la Fase 2.
I motori sono avvolti in questo modo per fornire una direzione di rotazione prevedibile.

In questo caso particolare la rotazione sarebbe oraria. L'inversione di due qualsiasi dei collegamenti di fase cambierà le relazioni di picco di fase e farà ruotare il motore nella direzione opposta. Il "rotolamento" di tali connessioni (ad esempio, lo spostamento da 1 a 2, da 2 a 3 e da 3 a 1) non cambierà le relazioni di fase e, pertanto, la direzione di rotazione rimarrà la stessa.

IL CAMPO MAGNETICO ROTANTE

Abbiamo visto come la tensione può raggiungere il picco in un circuito trifase e come i poli dello statore sono allineati per corrispondere ai picchi di tensione, ma perché il campo magnetico rotazionale si verifica automaticamente?

La Figura 4 pone il flusso lineare dei picchi di tensione mostrati nella Figura 2 e le posizioni dei poli mostrate nella Figura 3 in una prospettiva rotazionale.

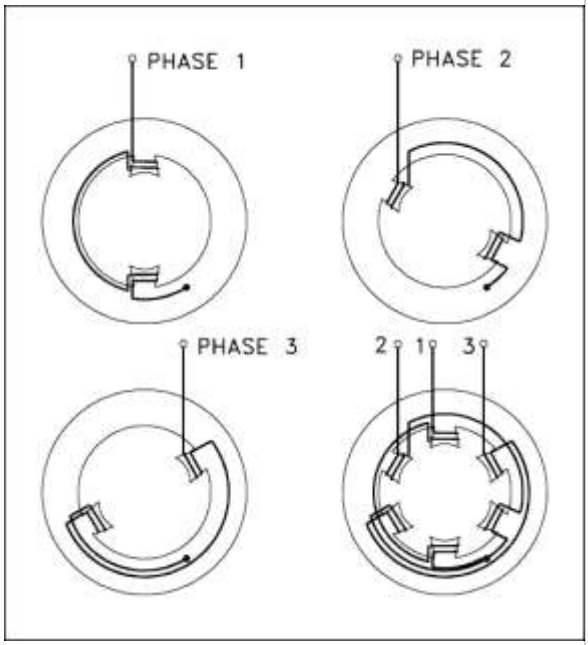


Figura 3. Posizionamento dei poli del motore CA

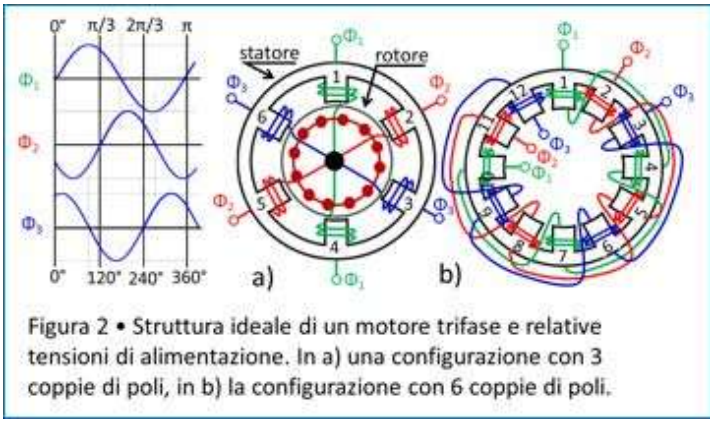


Figura 2 • Struttura ideale di un motore trifase e relative tensioni di alimentazione. In a) una configurazione con 3 coppie di poli, in b) la configurazione con 6 coppie di poli.

a) Motore a 2 poli per fase (3 fasi x 1 coppia di poli = 3 coppie)
b) Motore a 4 poli per fase (3 fasi x 2 coppie di poli = 6 coppie)

Le immagini dello statore mostrano i tre gruppi di poli e la loro polarità dai punti da 1 a 7.
L'immagine del grafico mostra i picchi di tensione di fase per gli stessi punti.

Al punto 1, la fase 1 è al suo picco positivo e viene generato un campo magnetico massimo nei poli 1 e 1A.

Al punto 2, la fase 3 è al suo picco negativo e il campo magnetico massimo è generato nei poli 3 e 3A.

Al Punto 3, il campo massimo si è spostato ai Poli 2 e 2A.

Se studi gli altri punti vedrai che questa tendenza continua in senso orario.
Di conseguenza, le tre fasi creano un campo rotante automatico nello statore.
Se due dei conduttori di fase in ingresso vengono scambiati, il campo magnetico ruoterà in senso antiorario.

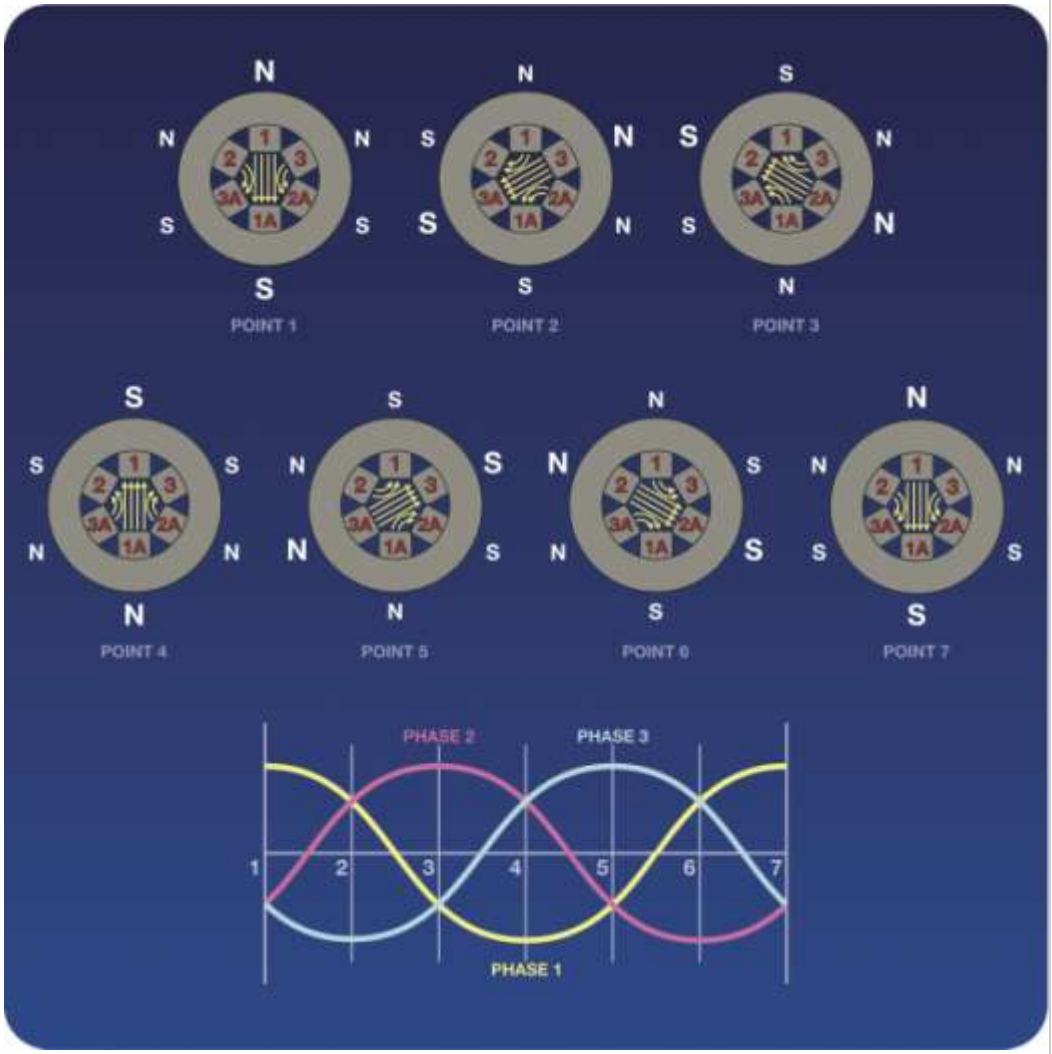


Figura 4. Il campo magnetico rotante

Come accennato in precedenza, la velocità del motore dipende sia dalla frequenza che dal numero di poli.

La velocità del motore cambierà in modo direttamente proporzionale alla variazione della frequenza. Ad esempio, a 30 Hertz un motore a 1.800 giri/min ruoterà a 900 giri/min.

Se si aggiunge un ulteriore set di poli a ciascuna fase dello statore mostrato nella Figura 3, anche la sua velocità diminuirà del 50 percento. Il tempo necessario per una rotazione di 360 gradi del campo dello statore è proporzionale sia alla frequenza che al numero di poli.

I motori trifase possono essere progettati per funzionare a due diverse velocità e la relazione di velocità dipende dal metodo di avvolgimento utilizzato.

I motori a due velocità e ad avvolgimento singolo utilizzano uno statore avvolto per una singola velocità, ma quando l'avvolgimento è collegato in modo diverso, cambia anche il numero di poli collegati.

Ad esempio, in una connessione sono collegati quattro poli, ma con la connessione alternata ne sono collegati otto.

Con questo metodo di avvolgimento, esisterà sempre un rapporto di velocità due a uno (1.800 giri/min/900 giri/min).

Di solito, la potenza del freno (BHP) a bassa velocità sarà un quarto di quella a piena velocità.

Tuttavia, i progetti a coppia costante manterranno mezzo BHP alla velocità inferiore.

I motori a due velocità e due avvolgimenti sono in realtà due motori avvolti su un unico statore.

Sebbene questi motori siano tipicamente più grandi e più costosi, non sono limitati al rapporto di velocità due a uno dei motori a singolo avvolgimento.

MOTORE DC O MOTORE AC?

La differenza fondamentale tra le due tipologie di motore elettrico è innanzitutto il tipo di alimentazione:

- il motore DC è un motore in corrente continua, monofase
- il motore AC è un motore in corrente alternata, monofase o trifase

Il motore DC è ampiamente utilizzato sia per applicazioni che richiedono piccole potenze, come apparecchiature ad uso domestico, che per applicazioni con potenze anche di diversi kW, come ad esempio trazioni ferroviarie e marine.
Il motore in corrente continua è inoltre adatto per applicazioni che richiedono alta precisione come robot industriali e macchine utensili.
Il motore AC è invece il più diffuso nell'industria ed è adatto per applicazioni in cui è necessario effettuare movimenti continui e con pochi cambi di velocità e in cui non è necessario fare posizionamento, come ad esempio nastri trasportatori, pompe, ventole, ecc.
Questi due tipi di motore elettrico si differenziano tra loro anche per la velocità che riescono a raggiungere.
Il motore AC riesce a raggiungere una velocità di rotazione superiore rispetto al motore DC, questo perchè nel motore a corrente alternata la velocità viene controllata variando la corrente nel motore, mentre nel motore a corrente continua la velocità viene controllata variando la frequenza, (di solito per mezzo di un convertitore di frequenza).

VANTAGGI DI UN MOTORE AC:

Il motore a corrente alternata presenta diversi vantaggi rispetto al motore DC:

- più economico in quanto consuma meno in fase di avviamento;
- richiede poca manutenzione;
- struttura più semplice;
- più robusto e resistente;
- meno soggetto ad usura;
- più adatto ad applicazioni che richiedono alte potenze.

VANTAGGI DI UN MOTORE DC:

- facilità da installazione, anche in sistemi mobili (alimentati a batteria);
- maggiore precision di posizionamento
- controllo della velocità variando la tensione di alimentazione;
- coppia elevata;
- maggiore rapidità nell'avviamento, l'arresto, l'accelerazione e l'inversione di marcia.

REGOLAZIONE DELLA VELOCITA' DEL MOTORE AC → INVERTER

Uno degli apparecchi elettronici che ha una posizione predominante nel mondo delle applicazioni di potenza, da quelle più contenute a quelle estremamente elevate è l'inverter che vede le applicazioni più estese nel pilotaggio di motori AC trifase.

Questo dispositivo ha il compito di convertire l'energia fornita da una sorgente in corrente continua in una uscita, ai suoi morsetti, di grandezze alternate sinusoidali, con ampiezze e frequenze che possono essere opportunamente controllate. Generalmente nel gergo industriale si intende un dispositivo atto alla regolazione della velocità dei motori trifase.

L'inverter è essenzialmente costituito da sei dispositivi di commutazione S1 – S6 (nella figura rappresentati come MOSFET). Questi sei interruttori sono collegati a due a due in configurazione a mezzo ponte e il punto comune di ognuno dei tre rami pilota una fase del motore. Comandando l'attivazione dello switch superiore di un ramo si mette in collegamento la fase relativa del motore al positivo dell'alimentazione. Ovviamente è indispensabile che non avvenga mai che i due switch di ogni ramo siano accesi contemporaneamente (pena un corto circuito).

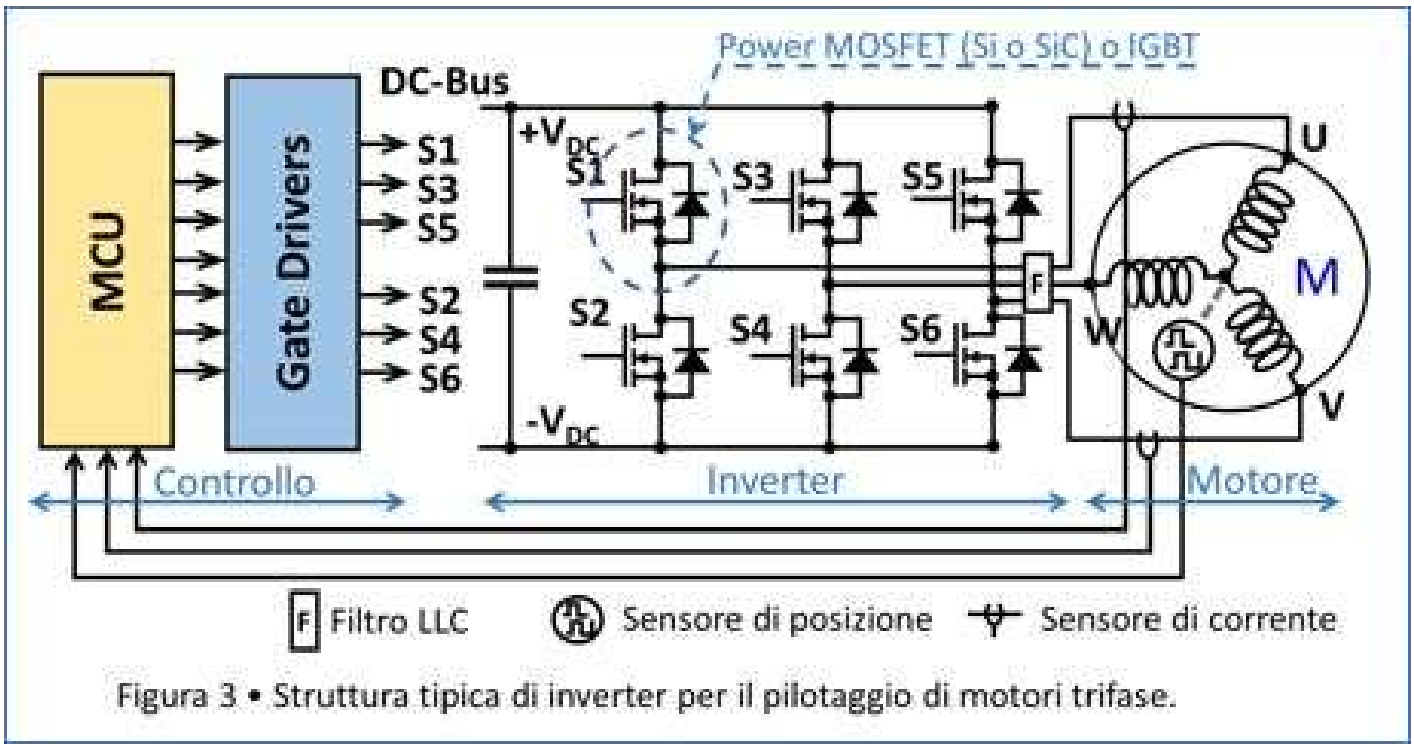
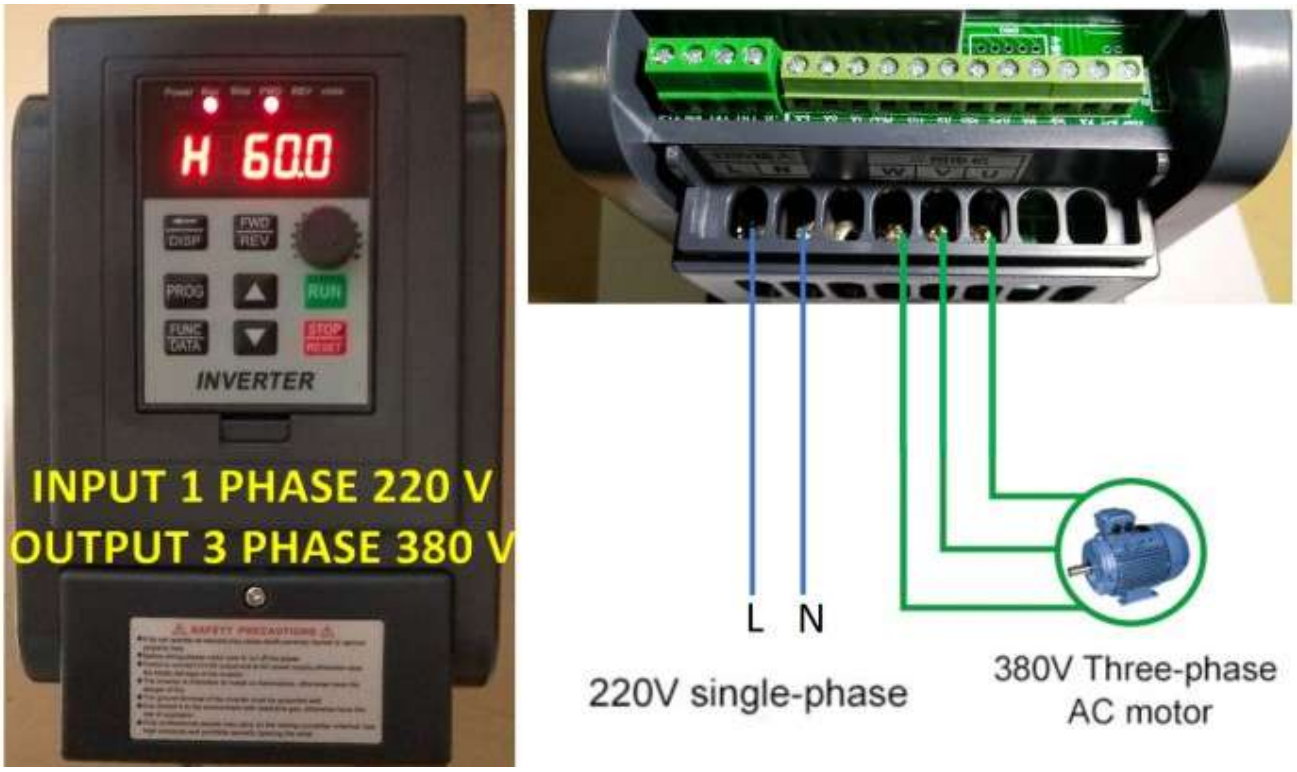


Figura 3 • Struttura tipica di inverter per il pilotaggio di motori trifase.

Vale la pena di sottolineare che ad ognuno dei MOSFET è collegato in antiparallelo un diodo che da una parte serve per consentire una via di richiusura delle correnti e consentire un ritorno dell'energia reattiva dal motore verso il bus di alimentazione.

Ma questi stessi diodi fungono invece da raddrizzatori quando il motore dovesse agire da generatore e traferire così energia dal motore verso il DC Link (per esempio in un veicolo elettrico durante la frenata il motore cambia la sua funzione e diventa generatore, consentendo così di recuperare energia).

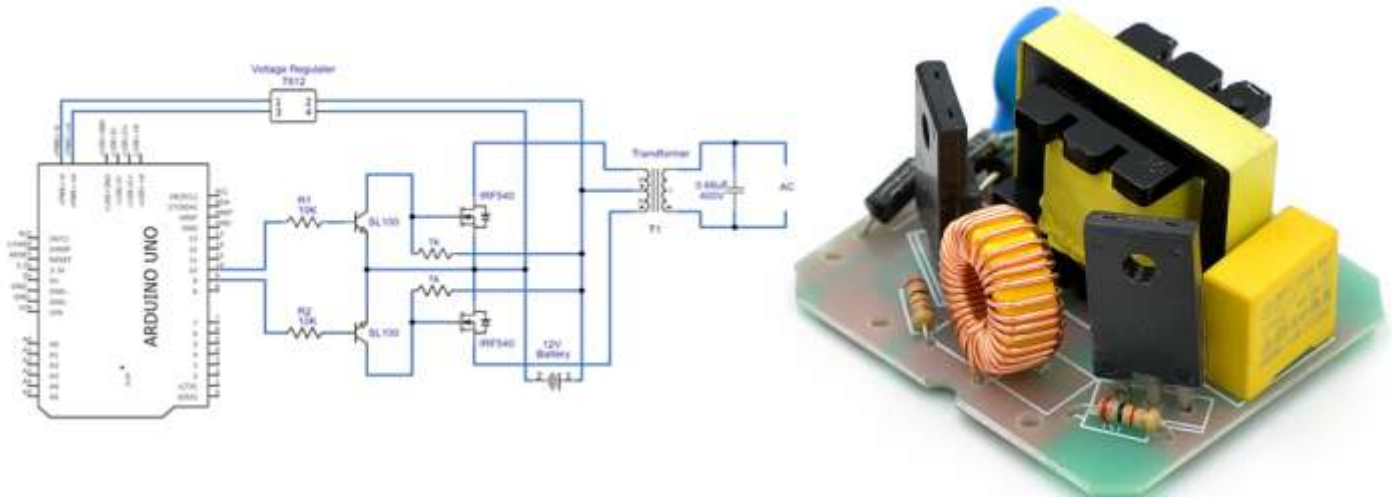


Inverter 220V → 380V (per motori bassa potenza)

Inverter 380V → 380V (per motori alta potenza)

CIRCUITO INVERTER BASATO SU ARDUINO

Un inverter è un dispositivo elettrico che converte la tensione CC.

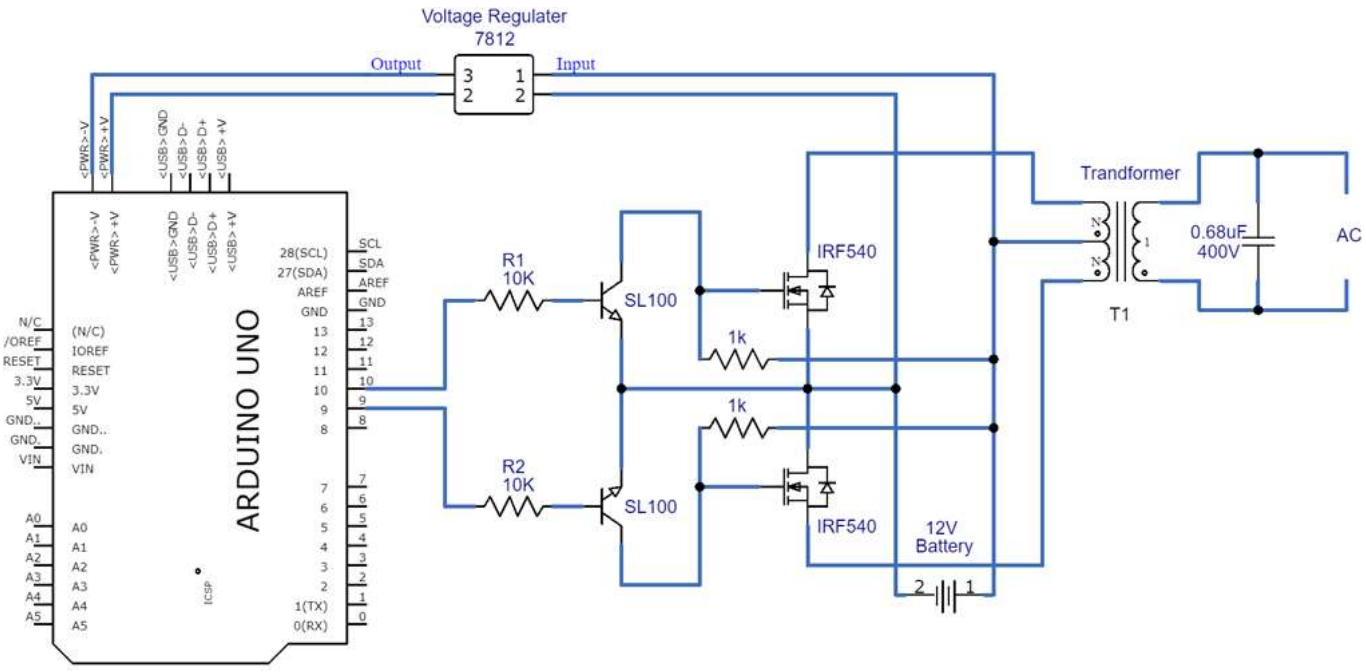


Questo circuito implementa un semplice inverter programmabile con Arduino per ottenere un'uscita CA a gradini, un'uscita CA sinusoidale modificata o un'uscita sinusoidale pura.

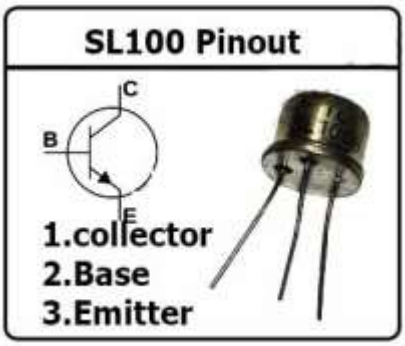
Hardware richiesto

S.No	Componente	Qtà
1	Regolatore di tensione 7812 IC	1
2	Transistore SL100	2
3	MOSFET IRF540	2
4	Trasformatore (12-0-12 V CA)	1
5	ArduinoUno	1
6	Resistenza 1KΩ,10KΩ	2,2
7	Fili di collegamento	–
8	Batteria 12V	1

Schema elettrico

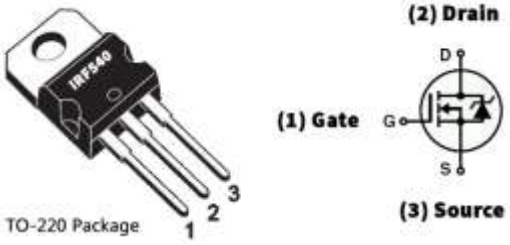


Configurazione dei pin del transistor SL100



Il transistor SL100 è un transistor NPN di media potenza per uso generico.

Configurazione pin Mosfet IRF540



Tipo di transistor: MOSFET

Polarità del transistor: canale N

Corrente di drenaggio (Id Max): 33A

Voltaggio Vds Max: 100 V.

Potenza (max): 120 W.

FUNZIONAMENTO DEL CIRCUITO

Come possiamo vedere nel circuito, sono coinvolti tre stadi e una batteria SLA da 12 V 5,0 Ah come sorgente di CC.

Il primo stadio è costituito dalla scheda del microcontrollore Arduino, che è programmata per fornire un segnale SPWM (Sinusoidal Pulse Width Modulation). È possibile modificare il codice per produrre output diversi dai pin Arduino.

Il secondo stadio è lo stadio di commutazione e pilota. L'impulso di uscita dai pin digitali Arduino pilota i transistor di commutazione SL100 NPN che a loro volta pilotano i MOSFET di potenza IRF540.

Il terzo stadio è lo stadio di uscita, che è costituito da un trasformatore dotato di presa centrale (primario 230 VAC / secondario 12-0-12 VAC). È collegato in modo inverso con il circuito di pilotaggio:

- il lato secondario (12-0-12 VAC) è collegato al MOSFET di potenza
- il lato primario del trasformatore è libero per fornire la tensione in uscita di 230V.

Quando la batteria è collegata a questo circuito, il regolatore di tensione 7812 alimenta la scheda Arduino a tensione costante (anche se la tensione della batteria varia) e inizia a produrre impulsi di uscita a seconda dello sketch.

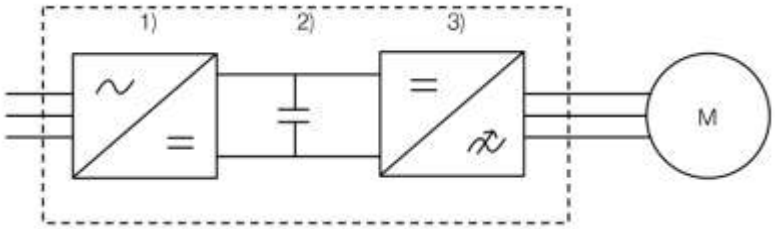
Questi impulsi pilotano il transistor SL100 e alimentano il MOSFET IRF540.
L'avvolgimento secondario del trasformatore collegato al MOSFET riceve energia e induce sul secondario un'uscita CA ad alta tensione 230V.

Codice dell' inverter Arduino

```
//Questo codice produce SPWM sui pin D9 e D10 della scheda Arduino Uno.
const int SpwmArry[] = {500,500,750,500,1250,500,2000,500,1250,500,750,500,500}; // Array of SPWM values.
const int SpwmArryValues = 13; //Put length of an Array depends on SpwmArray numbers.

// Declare the output pins and choose PWM pins only
const int sPWMpin1 = 10;
const int sPWMpin2 = 9;

// enabling bool status of Spwm pins
bool sPWMpin1Status = true;
bool sPWMpin2Status = true;

void setup() {
 pinMode(sPWMpin1, OUTPUT);
 pinMode(sPWMpin2, OUTPUT);
}

void loop() {
 // Loop for Spwm pin 1
 for(int i(0); i != SpwmArryValues; i++)
 {
  if(sPWMpin1Status)   {
   digitalWrite(sPWMpin1, HIGH);
   delayMicroseconds(SpwmArry[i]);
   sPWMpin1Status = false;
  }
  Else   {
   digitalWrite(sPWMpin1, LOW);
   delayMicroseconds(SpwmArry[i]);
   sPWMpin1Status = true;
  }
 }

 // Loop for Spwm pin 2
 for(int i(0); i != SpwmArryValues; i++)
 {
  if(sPWMpin2Status)   {
   digitalWrite(sPWMpin2, HIGH);
   delayMicroseconds(SpwmArry[i]);
   sPWMpin2Status = false;
  }
  Else   {
   digitalWrite(sPWMpin2, LOW);
   delayMicroseconds(SpwmArry[i]);
   sPWMpin2Status = true;
  }
 }
}
```

AZIONAMENTI AC

Un ***azionamento*** in AC comprende normalmente un trasformatore di ingresso (o un alimentatore elettrico), un convertitore di frequenza, un motore in AC e un carico (ventilatore, nastro trasportatore ecc.).
All'interno del singolo ***convertitore di frequenza*** si trovano un raddrizzatore, un collegamento in c.c. e un'unità inverter.



Un singolo convertitore di frequenza comprende 1) raddrizzatore, 2) collegamento in c.c., 3) unità inverter e 4) alimentatore elettrico.

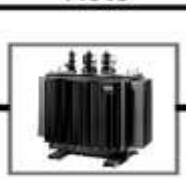
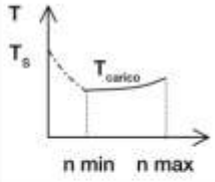
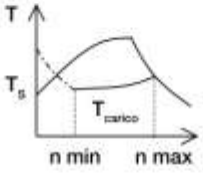
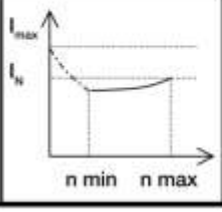
SELEZIONE DEL MOTORE

Il motore elettrico va considerato come una sorgente di coppia. Il motore deve resistere a sovraccarichi di processo ed essere in grado di produrre una determinata quantità di coppia. La capacità di sovraccarico termico del motore non deve essere superata. Per determinare la coppia massima disponibile nella fase del dimensionamento è necessario prevedere un margine del 30% per la coppia massima del motore

SELEZIONE DEL CONVERTITORE DI FREQUENZA

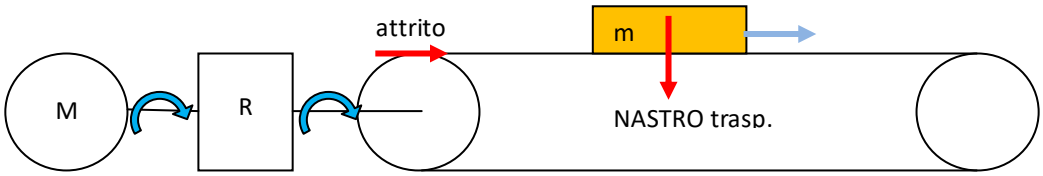
Il convertitore di frequenza viene selezionato in base alle condizioni iniziali e al motore selezionato.
E' necessario verificare che convertitore di frequenza sia idoneo a produrre la corrente e la potenza richieste.
Verificare anche la capacità di sovraccarico potenziale del convertitore di frequenza in caso di un carico ciclico di breve termine.

DIMENSIONAMENTO DELL'AZIONAMENTO

Fase di dimensionamento	Rete	Convertitore	Motore	Carico
				
1) Verificare le condizioni iniziali della rete e del carico	f_N=50Hz, 60Hz U_N=380...690V			
2) Selezionare il motore in base a: • Capacità di sovraccarico termico • Gamma di velocità • Coppia massima richiesta				
3) Selezionare il convertitore di frequenza in base a: • Tipologia del carico • Corrente massima e continua • Condizioni della rete				

AZIONAMENTI MECCANICI CON MOTORI ELETTRICI A INDUZIONE

Consideriamo un nastro trasportatore che deve movimentare dei corpi di massa ***m***.



Il motore AC a induzione M è collegato ad un riduttore a ingrannaggi R chè è collegato a sua volta al rullo di traino del nastro.

Il motore M è caratterizzato da una coppia motrice Mm che dipende dalla sua potenza nominale Pn e dal numero di giri n°:

Pn = Mm * ωm [w] con $\omega = 2\,\pi\,n°/\,60$ [rad/s] n° = rpm= numero di giri /minuto

Il riduttore R ha lo scopo di ridurre il numero di giri del motore (tipicamente alto 1500, 3000 … giri/min.) ad un valore compatibile con il sistema di movimentazione da implementare. Trascurando il rendimento meccanico del riduttore abbiamo:

Pot = Mm * ωm = Mr * ωr con

- i = rapporto riduzione= ωm / ωr
- Mr = coppia in uscita al riduttore [Nm]
- ωr = velocità in uscita al riduttore [rad/s]

L'effetto del riduttore, oltre che a diminuire la velocità di rotazione, è quello di aumentare la coppia Mr per il carico.

Nel moto rettilineo di un corpo che da fermo viene messo in movimento tramite una forza abbiamo: F= m *a [N]

Nel caso di un corpo che viene messo in rotazione rispetto ad un asse abbiamo: M = J * α [Nm] con
J = momento di inerzia del corpo rispetto all'asse di rotazione
α = accelerazione angolare del corpo = dω/dt → dt è il tempo necessario per portare a regime il sistema da fermo

Per una massa ***m*** che ruota ad una distanza ***r*** rispetto al'asse di rotazione abbiamo: J= m*r^2 [Kg m2]

Il calcolo del momento di inerzia J è fattibile per semplici sistemi con geometrie ben definite (cilindri, dischi ecc.).
Per machine e forme complesse il momento di inerzia si ricava con opportune prove sperimentali.

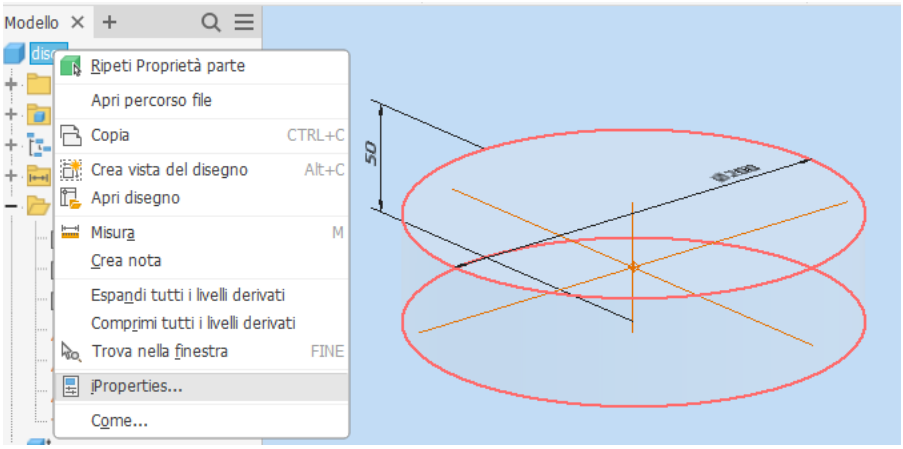
Nel caso di nastri trasportatori, ad esempio, è il costruttore che fornisce la ***coppia di primo distacco a pieno carico***, cioè il valore di coppia necessario per far partire il nastro trasportatore (attrito statico) nelle condizioni di massimo carico e massima accelerazione possible: M distacco = J*α [Nm]

Subito dopo la partenza (primo distacco) è necessario un certo intervallo di tempo Δt per raggiungere la velocità di regime prevista. In questa fase la coppia motrice dovrà vincere l'inerzia della masse movimentate e l'attrito volvente (non deve più vincere l'inerzia dei rulli del nastro). Si parla quindi di forza e coppia accelerante:
Fa = (m*g) * a + (m*g) μ con a = v/Δt e v= velocità lineare a regime del nastro e μ attrito volvente del nastro sui rulli
Ma = Fa * d/2 con d=diametro rullo di traino

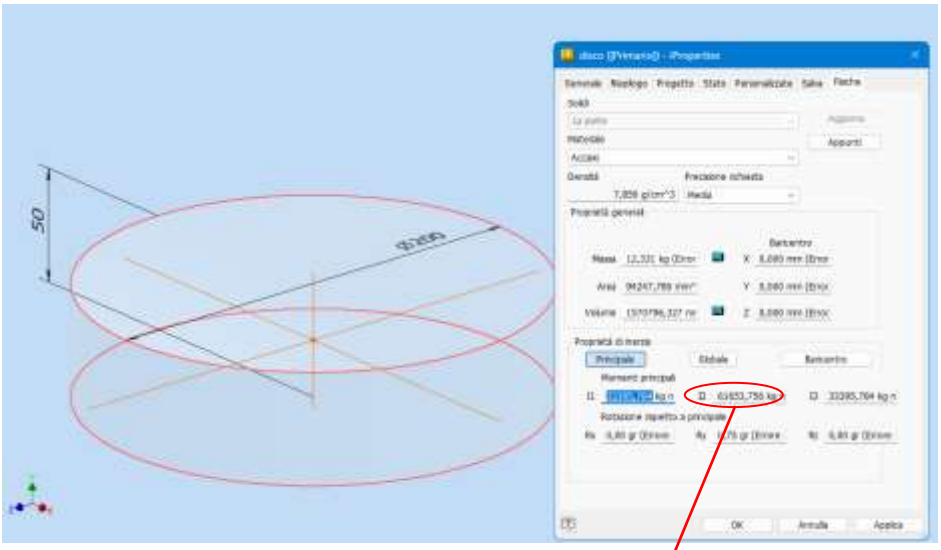
Quando il sistema raggiunge la velocita di rotazione prevista (a regime) la coppia motrice non deve più vincere l'inerzia del sistema e di conseguenza la potenza richiesta al motore è generalmente più bassa di quella di spunto.
-F regime = (m*g) *μ con μ attrito volvente del nastro sui rulli
-M regime = Freg. * d/2 con d=diametro rullo di traino

MOMENTO DI INERZIA DI PEZZI COMPLESSI

Tramite il menu “iProperties” è possibile ottenere una stima accurata delle proprietà meccaniche di un pezzo.

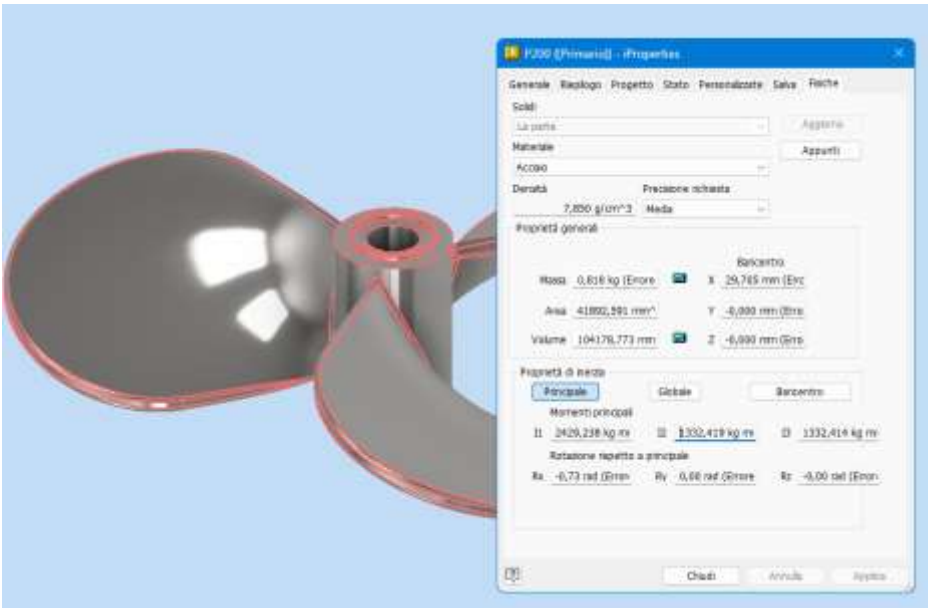


Alla sezione “fisiche” si può impostare il materiale e calcolare le proprietà principali (baricentro, momenti inerzia …).



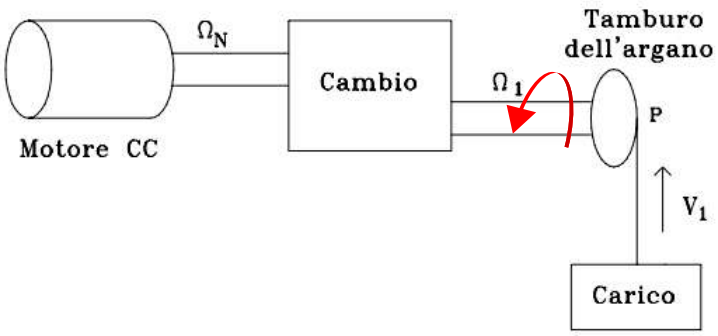
Con FORMULA: $Iy = ½ * m * r^2 = 0,5 * 12,331 * 0,1^2 = 61655$ kg mm^2 da Inventor → 61654 kg mm^2

L’esempio sottostante mostra una situazione pià complessa difficilmente risolvibile a *mano*.



ARGANO PER SOLLEVAMENTO

Si consideri il meccanismo di sollevamento riportato in figura.
Il momento d'inerzia del motore è J_N = 1 Kgm2.
Il cambio ha un rapporto di riduzione i=10:1 .
Il momento d'inerzia J_r del riduttore, riferito al motore, è pari a 0,2 Kgm2.
Il tamburo dell'argano ha un raggio ra = 0.3 m ed un momento d'inerzia J_{tamb}=3 Kgm2.
Il carico da sollevare ha massa mc= 1000 Kg.



Nel caso in esame, si può pensare che al momento d'inerzia proprio del tamburo si sovrapponga quello del peso che, per l'ipotesi di anelasticità del cavo, può essere riportato sulla circonferenza del tamburo stesso (punto P di figura).
Il momento d'inerzia del peso è dunque: $J_p = m_c * r^2$ = 1000 * 0,3^2= 90 Kg m2

Il momento complessivo a valle del riduttore (tamburo + peso) vale quindi:
$Jc = J_{tamb} + J_p$= 90 + 3 = 93 Kg m2

Se si suppone che il cambio sia privo di perdite, la potenza meccanica viene tutta trasmessa, quindi vale la relazione:

Pot= ω_n*Cn = ω_t* Ct dove
Ct = coppia trasmessa al tamburo dell'argano
Cn = coppia nominale del motore

Per un carico puramente inerziale, la coppia è legata alla velocità di rotazione dalla relazione:
Ct = Jc * α_c = Jc * $d\omega_t/dt$ con α_c = accelerazione angolare del tamburo

Essendo il rapporto di riduzione i= ω_n / ω_t e Ct= Cn*ω_n / ω_t = Cn*i sostituendo nella precedente:

Cn * i = Jc * $d(\omega_n/i)/dt$ = Jc * 1/i* $d(\omega_n)/dt$ abbiamo

Cn = Jc / i^2 * $d(\omega_n)/dt$ = Jc / i^2 * α_n con α_n = accelerazione angolare del motore

Si definisce J_{NC} = Jc / i^2 momento d'inerzia del carico riportato al motore (momento riflesso)

Nel nostro caso: J_{NC} = 93 / 10^2 = 0,93 Kgm2

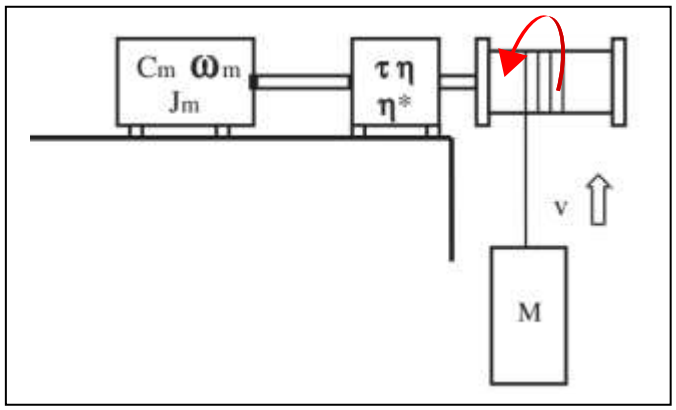
Il momento di inerzia totale visto dal motore comprenderà inoltre quello proprio del motore J_N e quello del riduttore J_r :

J_{Ntot} = J_N + J_r + J_{NC} =1+ 0,2+ 0,93 = 2,13 Kgm2

ARGANO PER SOLLEVAMENTO 2

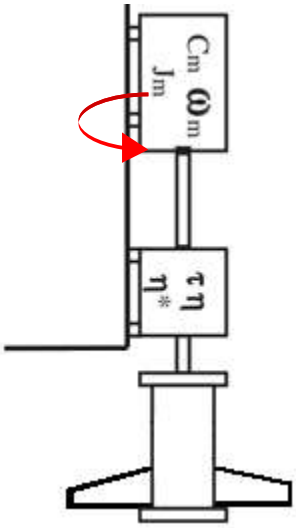
Un motore asincrono deve sollevare / abbassare un carico tramite una fune che si avvolge su un tamburo di diametro D = 0.5 m.
Il carico si deve muovere alla velocità costante approssimativa v = 0.6 m/s.
La massa del carico è pari a 200 kg e la puleggia ha inerzia trascurabile.
Sono disponibili tre riduttori di velocità con riduzioni τ=1/50, 1/60, 1/80, rendimento diretto e retrogrado η=0.7 e inerzia riduttore trascurabile. E' richiesto di:

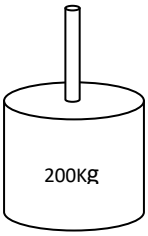
- scegliere il motore ed il riduttore più adatti;
- verificare quale sarà la velocità di regime;
- calcolare approssimativamente il tempo di avviamento per il caso di salita;



AZIONAMENTI MECCANICI: AGITATORE PER LIQUIDI

Un motore asincrono comanda, tramite un riduttore, un agitatore per liquidi di elevata densità.
Il riduttore ha un rendimento pari a 0,9.
L'agitatore ha una massa complessiva di 20Kg e può essere approssimato da un cilindro pieno di diametro 500mm.
E' richiesta una coppia motrice di 1000 Nm alla velocità di 20 rpm per mescolare il fluido.
Il momento d'inerzia del motore è J_N = 1 Kgm2.
Il momento d'inerzia J_r del riduttore, riferito al motore, è pari a 0,2 Kgm2.
Tempo di accelerazione da fermo pari 10 sec.





Il momento di inerzia dell'agitatore è dato da:
I agit. = 1/2*m*r^2 = 0.5 * 20 * $0,5^2$ = 0,625 Kg m2

La velocità angolare a regime dell'agitatore è data:
ω_t = 6.28 *n° / 60 =6.28* 20/60= 2,1 rad/sec.

La potenza che deve essere trasmessa al carico a REGIME vale quindi:
Pt = ω_t* Ct = 2,1* 1000 Nm = 2100 w

Not oil rendimento del riduttore si ricava la potenza nominale del motore:
Pn= Pt/η = 2100 / 0.9= 2333 w.

Suppondendo di utilizzare un motore a induzione a due poli da 1420 rpm dobbiamo adottare il seguente rapporto di riduzione:
i = 1420 / 20 = 71

11.2.1 Motore asincrono: parametri principali, regolazione

Testo esercizio

Si consideri un motore asincrono trifase con i dati di targa di seguito riportati. Si chiede di determinare:

- la coppia nominale del motore;
- la frequenza di alimentazione necessaria ad ottenere la rotazione del campo a 2500 RPM.

Dati Da catalogo si desumono i seguenti dati di targa:

- potenza nominale: $W_n = 7.5\ kW$
- tensione nominale: $V_n = 380\ V$
- corrente nominale: $I_n = 15.5\ A$
- numero di poli: 4
- scorrimento nominale percentuale: $s_n = 3.33\%$

Coppia Nominale

Conoscendo la potenza e velocità nominali è possibile determinare la coppia nominale.
La velocità nominale è proporzionale alla velocità a vuoto (velocità di sincronismo):

$$\omega_n = (1 - s_n)\,\omega_0 \qquad\qquad N_n = (1 - s_n)\,N_0$$

dove

$$\omega_0 = \frac{2\pi f}{p} = \frac{2\pi 50}{2}\ [rad/s] \qquad\qquad N_0 = \frac{f60}{p} = \frac{50\ 60}{2}\ [giri/min]$$

Il parametro f è la frequenza della corrente alternata della rete, che in Italia è 50 $[Hz]$. Sostituendo i valori numerici si ottiene:

$$\omega_0 = 157\ [rad/s] \qquad\qquad \omega_n = 151.85\ [rad/s]$$

$$N_0 = 1500\ [RPM] \qquad\qquad N_n = 1451\ [RPM]$$

La coppia nominale è, quindi:

$$C_n = \frac{W_n}{\omega_n} = \frac{7500}{151.85} = 49.39\ [Nm]$$

Frequenza di alimentazione

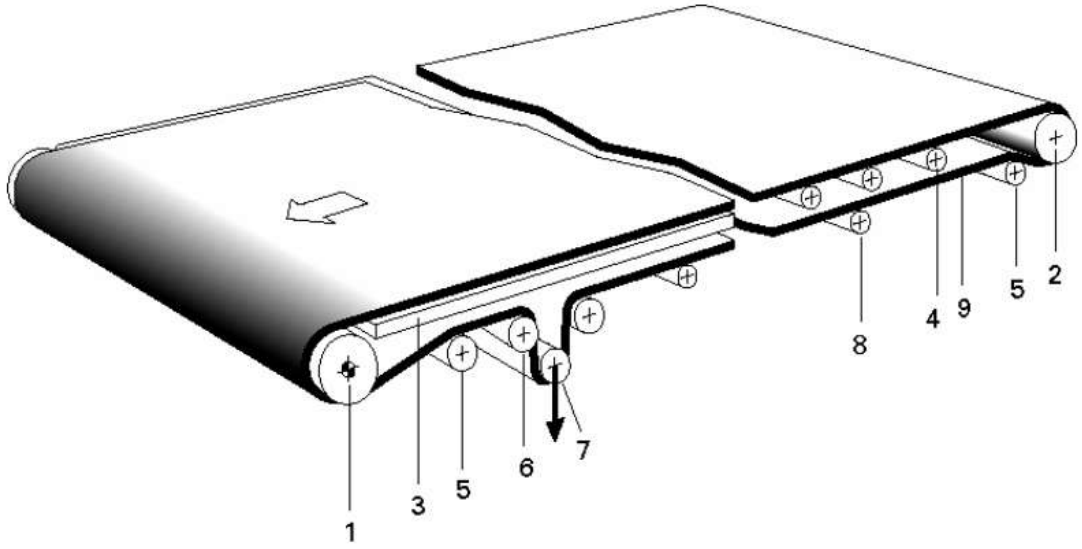
La velocità di rotazione del campo magnetico è proporzionale alla frequenza di alimentazione. Per ottenere una velocità pari a $\bar{N} = 2500\ [RPM]$ occorre una frequenza $\bar{f}$ che si determina con la seguente proporzione:

$$\bar{f} : f = \bar{N} : N_0$$

$$\bar{f} = \frac{2500}{1500}50 = 83.33\ [Hz]$$

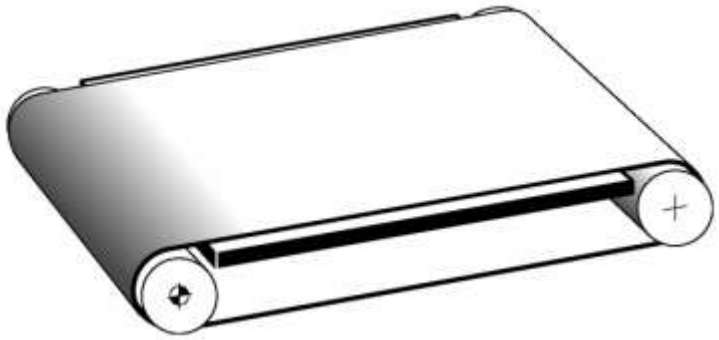
NASTRO TRASPORTATORE

Nella sua forma più semplice, un trasportatore è composto da una carpenteria che comprende il sostegno del nastro (piano di scorrimento o rulli di supporto), un tamburo motore, che normalmente è il tamburo "di testa", un rullo di rinvio, che normalmente è il rullo "di coda" e un nastro trasportatore. Sistemi più complessi avranno componenti aggiuntivi come gruppi di traino e di tensionamento, elementi di centraggio del nastro, deviatori di prodotto, accumulatori, sensori, ecc.



1 rullo di traino 6 rullo di controflessione 2 rullo di rinvio 7 rullo di tensionamento 3 piano di scorrimento 8 rullo di supporto (sul lato di ritorno) 4 rullo di supporto 9 nastro trasportatore 5 controrullo 10 carpenteria (non indicata)

Nastro con piano di scorrimento

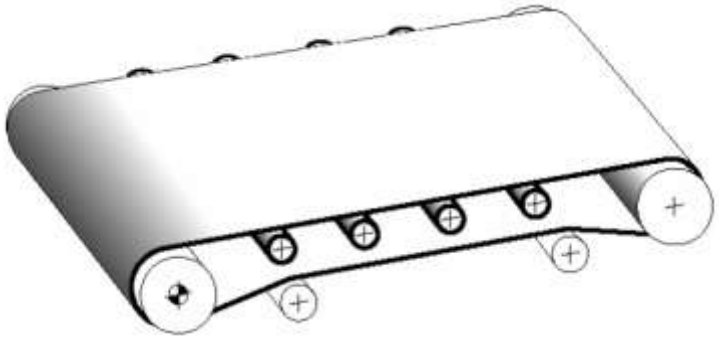


I vantaggi di un nastro supportato da un piano di scorrimento sono principalmente la maggiore stabilità dei prodotti trasportati e la limitata influenza sul centraggio del nastro – un vantaggio che distingue questa soluzione da quella che prevede l'utilizzo di rulli di supporto. Selezionando in maniera corretta il materiale del lato di scorrimento del nastro e il piano di scorrimento stesso, è possibile variare in nostro favore il coefficiente di attrito, la rumorosità e la vita utile del nastro.

I materiali consigliati per il piano di scorrimento sono:
- Lamiera di acciaio decapato (lamiera di acciaio disincrostata chimicamente)
-Lamiera di acciaio inossidabile (utilizzata in particolare nel settore alimentare)
- Plastiche dure (come la resina fenolica, ecc.) utilizzate principalmente come copertura di pannelli in truciolato o compensato
- Fogli laminati di legno duro (faggio, quercia)

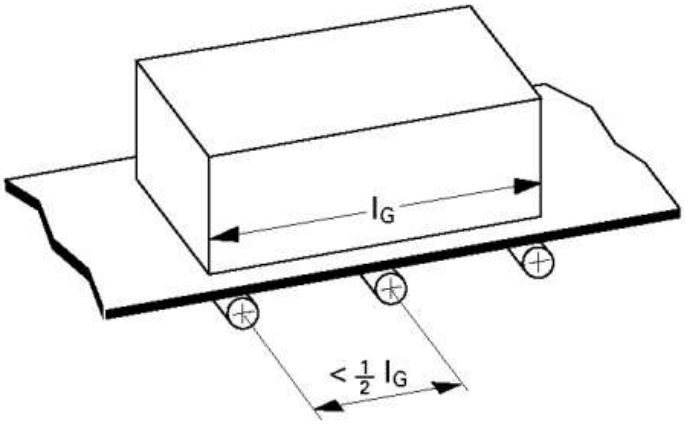
L'attrito tra il piano di scorrimento e il nastro viene notevolmente influenzato dal tipo di materiale e dalla finitura superficiale del piano di scorrimento, dall'umidità, dalla polvere, dalla sporcizia, ecc.

Nastro con rulli di scorrimento



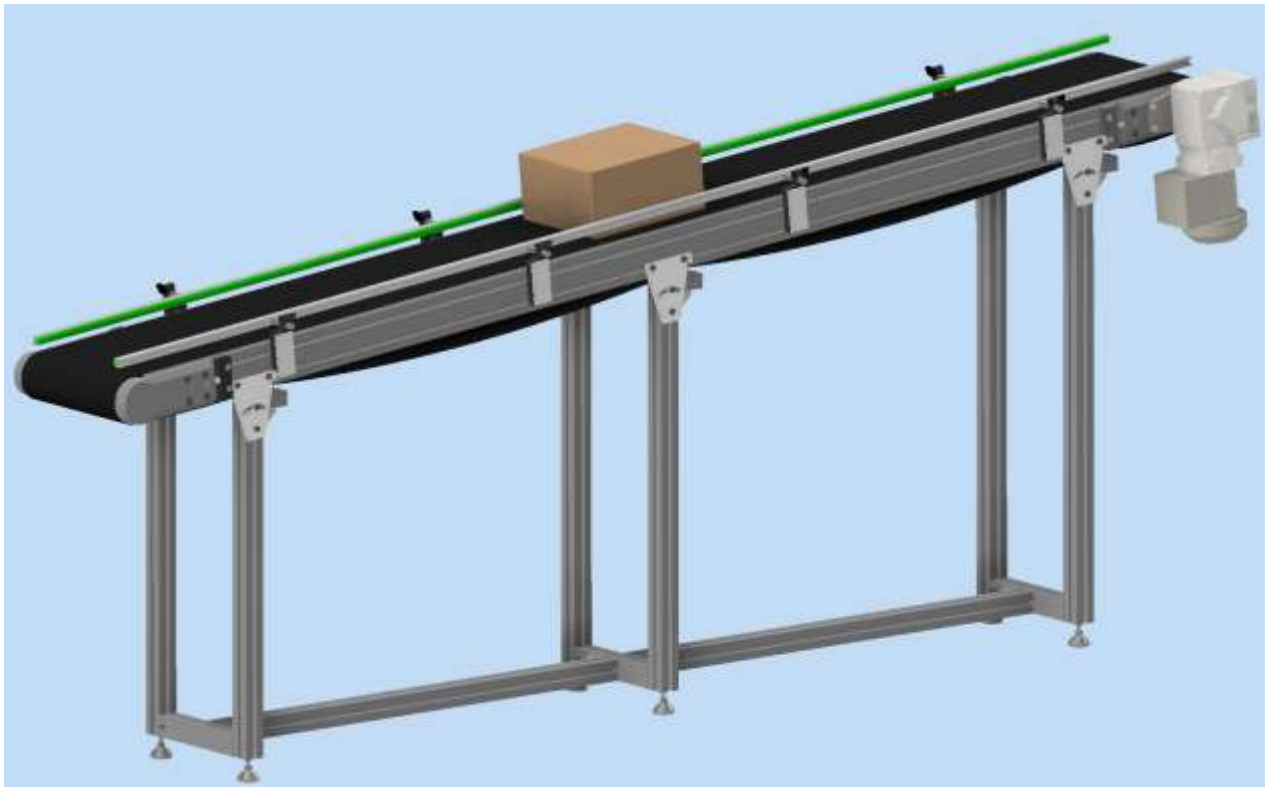
In presenza di trasportatori lunghi e carichi grandi/pesanti si utilizzano rulli di supporto in sostituzione del piano di scorrimento. I rulli di supporto riducono la perdita per attrito, la forza periferica e la potenza richiesta per il traino.

I rulli maggiormente utilizzati sono realizzati da tubi di precisione in acciaio supportati da cuscinetti a rulli.



La distanza tra i rulli di supporto deve essere inferiore a metà della lunghezza delle unità di carico trasportate lG, in modo che i prodotti trasportati appoggino sempre su almeno due rulli.

AZIONAMENTO PER NASTRO TRASPORTATORE



Si deve dimensionare un motore asincrono trifase 400V che gestisce un nastro trasportatore a rulli.

Lunghezza nastro	30	m	
Massa nastro /m	5	Kg	
Massa nastro	300	Kg	2x lunghezza
Velocità lineare nastro	1,5	m/s	
Tempo di accelerazione	5	s	da fermo
Massa trasportata /m	30	Kg	
Massa tot. trasportata	900	Kg	
Massa totale movimentata	1200		comprensiva di nastro
Diametro puleggia	0,2	m	
Attrito volvente	0,07		
Attrito statico	0,09		
Md coppia primo distacco	140	Nm	→ dal costruttore del nastro
Rendimento riduttore	0,98		
Motore 4 poli trifase	1475	rpm	
Capacità sovraccarico	150	% per 30s	
Tensione alimentazione	400	V trifase	
Fattore di potenza	0,85		
ω motore	154,4	rad/s	
Rendimento inverter	0,98		
Capacità sovraccarico	150	% per 60s	

RAPPORTO DI TRASMISSIONE DEL RIDUTTORE

ωp = velocità puleggia	15	rad/s	v nastro = ω * D/2
i = rapporto trasmissione	10,3		ωmotore / ωp

COPPIA DI PRIMO DISTACCO

Md	140,0	Nm	→ dal costruttore del nastro
Pd	2100	W	potenza primo distacco

COPPIA ACCELERANTE (x vincere inerzia del carico e attrito volvente dopo il distacco)

a = v / t = 1,5/5	0,3	m/s2	accelerazione lineare nastro
Fa	1184	N	forza accelerante
Ma	118	Nm	coppia accelerante
Pa	1812	W	potenza accelerante

COPPIA A REGIME (volvente)

a	0	m/s2	omega costante
Fr	824	N	forza a regime
Mr	82	Nm	coppia a regime
Pr	1261	W	potenza a regime

La coppia accelerante è presente finché il motore non raggiunge la velocità massima.
La potenza accelerante si trova con la coppia accelerante alla massima velocità

Un motore elettrico a induzione in genere ha una capacità di sovraccarico del 150% per 30s

Pa con sovraccarico 150%	1208	W	sovraccarico per 5s
Si puo' quindi prendere un motore elettrico da			1,5 kW
Ia	3,1	A	corrente all'avvio assorbita dal motore

La corrente di primo distacco invece vale

Id	3,6	A	

Il convertitore di frequenza può sopportare un sovraccarico del 150% per 60s

I nominale convertitore	2	A	

SISTEMI DI REGOLAZIONE

Un sistema di ***REGOLAZIONE*** non prevede l'utilizzo di sensori ma si base sulle leggi fisiche che governano il sistema.
Ad esempio per mantenere una certa temperatura dell'acqua in un recipiente è possible far ricorso alle leggi della termotecnica per calcolare la dispersioni termiche dell'involucro e quindi la potenza termica che deve essere fornita tramite un elemento riscaldante.
Applicando correttamente le leggi della fisica si può ottenere il risultato richiesto senza un controllo continuo del sistema.

Nel caso di presenza di disturbi esterni (non adiabaticità del recipiente, prelievo di acqua, ecc.) il risultato non può essere raggiunto senza la presenza di opportune ***sensori*** che misurano in tempo reale la grandezza da controllare.

SISTEMA DI RISCALDAMENTO RESISTIVO

Progettare un sistema di **REGOLAZIONE** della temperatura dell'acqua in un recipiente adiabatico da 10 litri.

La temperatura dell'acqua deve essere portata da 20°C a 50°C con una resistenza termica da 115 watt quando si preme un pulsante di accensione. Il sistema NON prevede l'utilizzo di sensori di temperatura e per questo motivo si parla di REGOLAZIONE e non di CONTROLLO.

Se si applicano correttamente le leggi della fisica alla base del processo di riscaldamento dell'acqua si può ottenere il risultato richiesto. Nel caso di presenza di disturbi esterni (non adiabaticità, prelievo acqua prima del termine fase di riscaldamento, ecc.) il risultato non può essere raggiunto senza la presenza di sensori.

Leggi fisiche coinvolte:

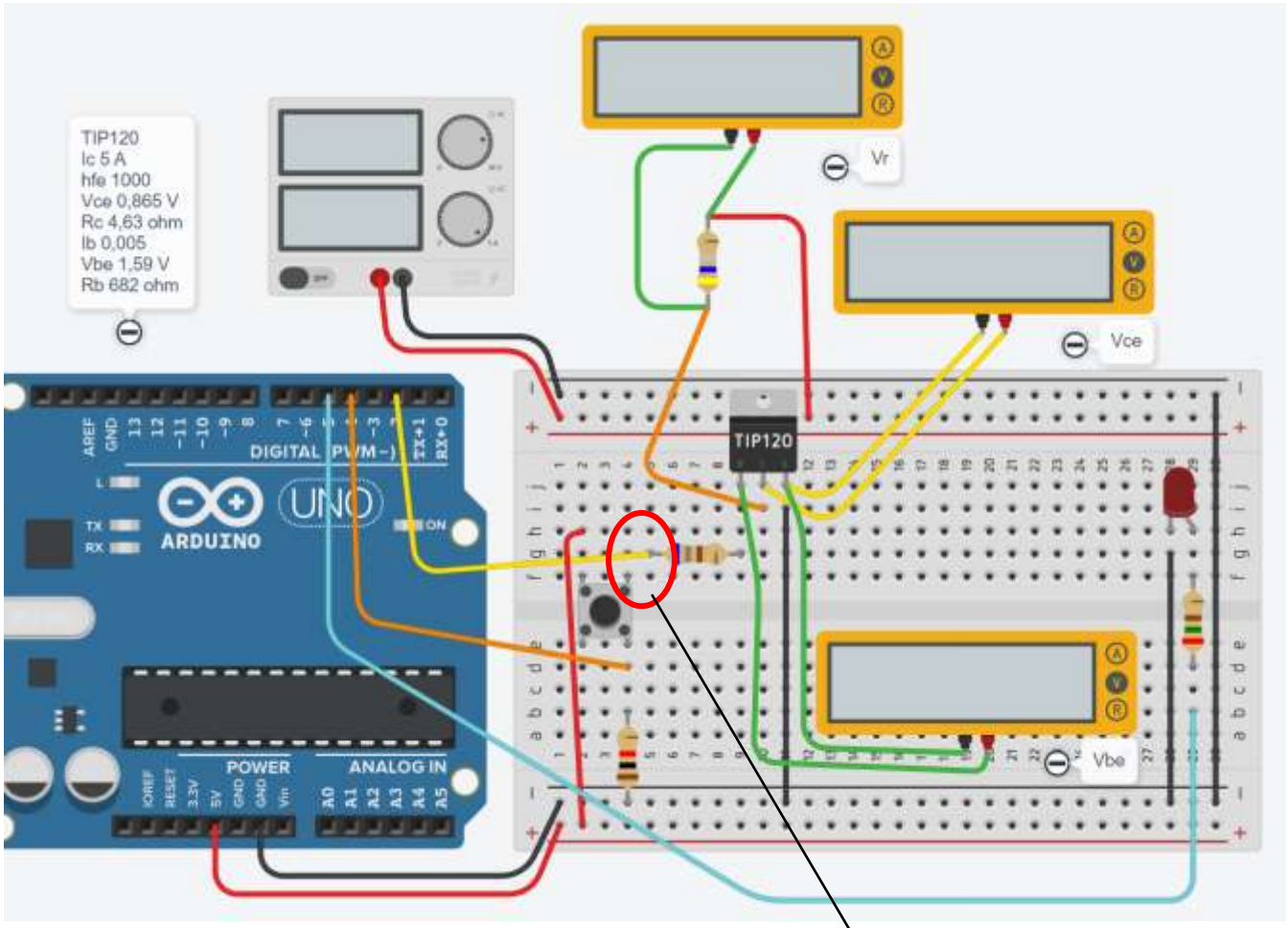
1- Energia termica fornita all'acqua: **Q= m*Ct* (Tf-Ti)** [Joule]
2- Potenza termica fornita dall'elemento riscaldante: **Pot.= Q / t** [w=J/s]

Ricavando il calore fornito dalla 2° equazione e sostituendolo nella prima si ottiene il tempo "t" necessario ad ottenere il risultato richiesto.

Il microcontrollore dovrà quindi attivare l'elemento riscaldante per il tempo calcolato.

t = m*Ct* (Tf-Ti) / Pot. = 10 * 4186* (50-20) / 115 = 10920 s = 95 minuti

Per valutare il tempo trascorso in Arduino si deve impiegare la funzione "millis()" che ritorna il numero di millisecondi trascorsi dall'accensione.



Attenzione a non mettere i pin del pulsante allineati in verticale alla resistenza!

```
long t0;
int tempoAttivazione= 10*1000;  // secondi
bool flagAttivo = false; // flag per sapere se è attivo riscaldamento

void setup()
{
  pinMode(2, OUTPUT);  // TIP120
  pinMode(5, OUTPUT);  // LED
  pinMode(4, INPUT);   // START
  Serial.begin(9600);
}

void loop()
{
  int statoStart= digitalRead(4);  // STATO BOTTONE START

  if (statoStart==HIGH) {
    Serial.println("Attivazione");
    flagAttivo= true;
    t0= millis();  // ISTANTE ATTIVAZIONE
    digitalWrite(2, HIGH);
    digitalWrite(5, HIGH);
  }
  // CONTROLLO SE è PASSATO IL TEMPO DI ACCENSIONE PREVISTO
  if ( ((millis() - t0) > tempoAttivazione)  && (flagAttivo== true)) {
    Serial.println("Fine riscaldamento");
    digitalWrite(2, LOW);
    digitalWrite(5, LOW);
    flagAttivo= false;

  }

  delay(200);
}
```

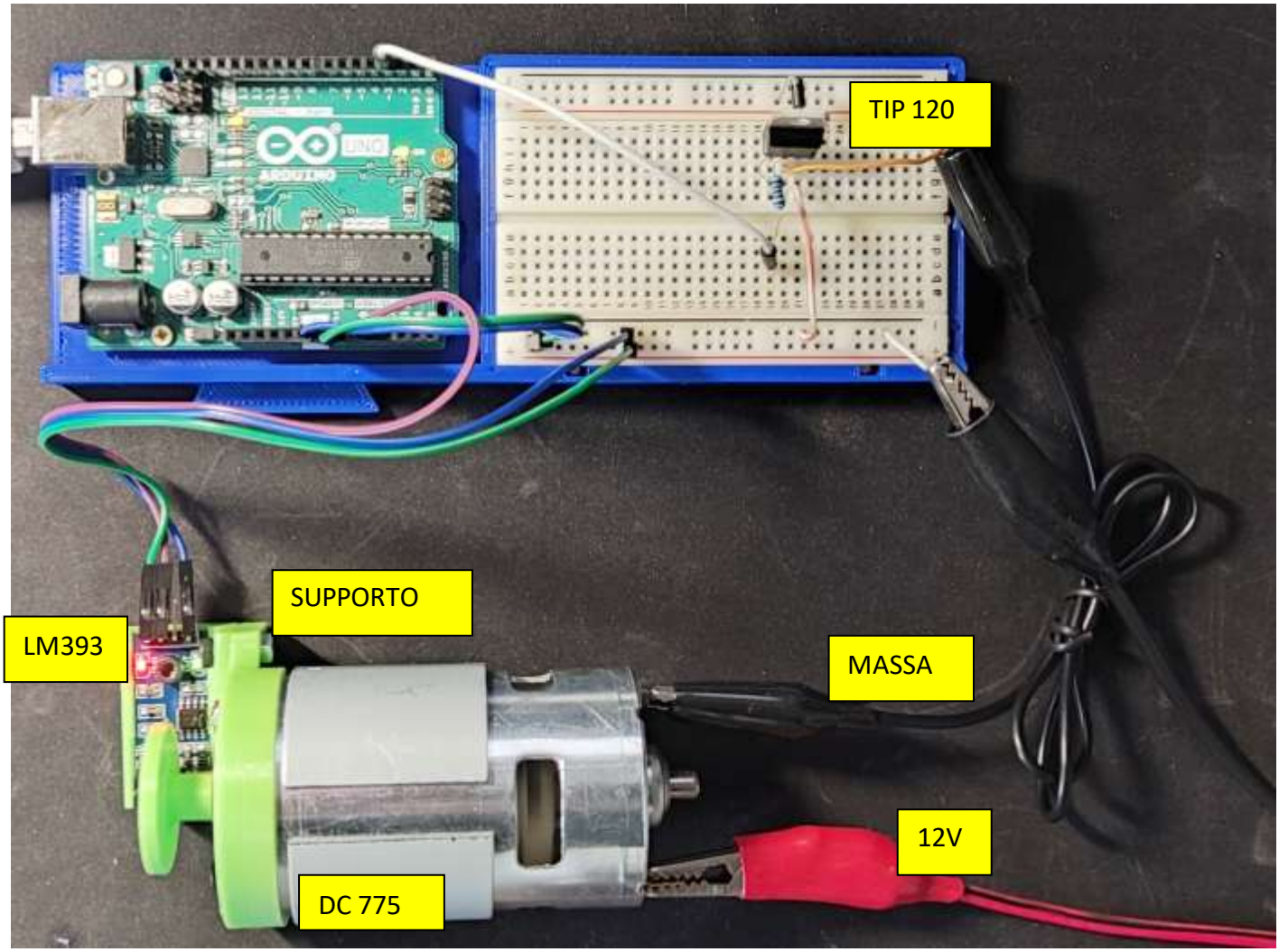
REGOLAZIONE DEL NUMERO DI GIRI DI MOTORE C.C. AD ALTA VELOCITA'

La fotografia mostra un motore CC da 12 V a 6000 giri/minute la cui velocità viene regolata tramite un transistor di potenza TIP120 a sua volta controllato con un segnale PWM da Arduino.

Lo scopo del circuito è quello di regolare la tensione dei alimentazione del motore per ottenre il il numero di giri a vuoto desiderato.

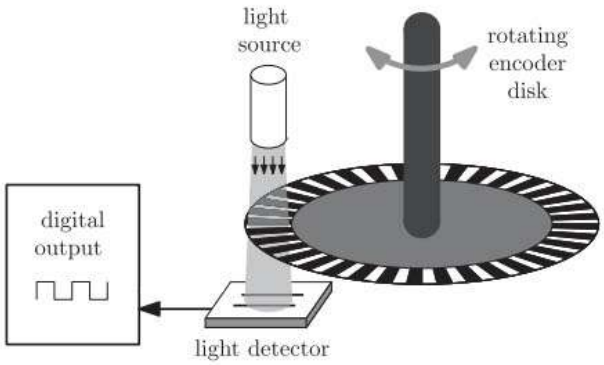
Per rilevare la velciotà del motore si utilizza il modulo Arduino LM393 dotato di emettiore e ricevitore IR a forcella.

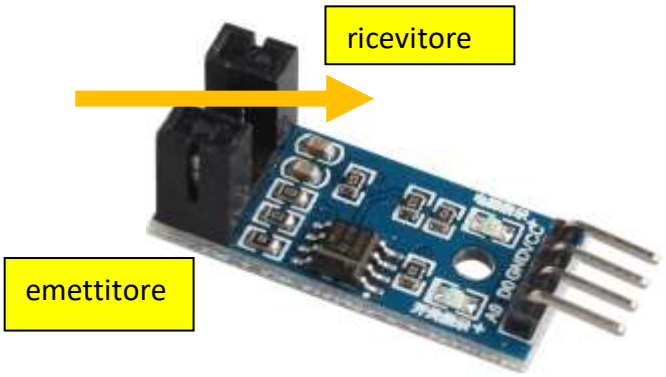
Il pezzo verde (supporto del modulo) è stato disegnato in 3D e poi stampato in 3D.



MODULO IR LM393

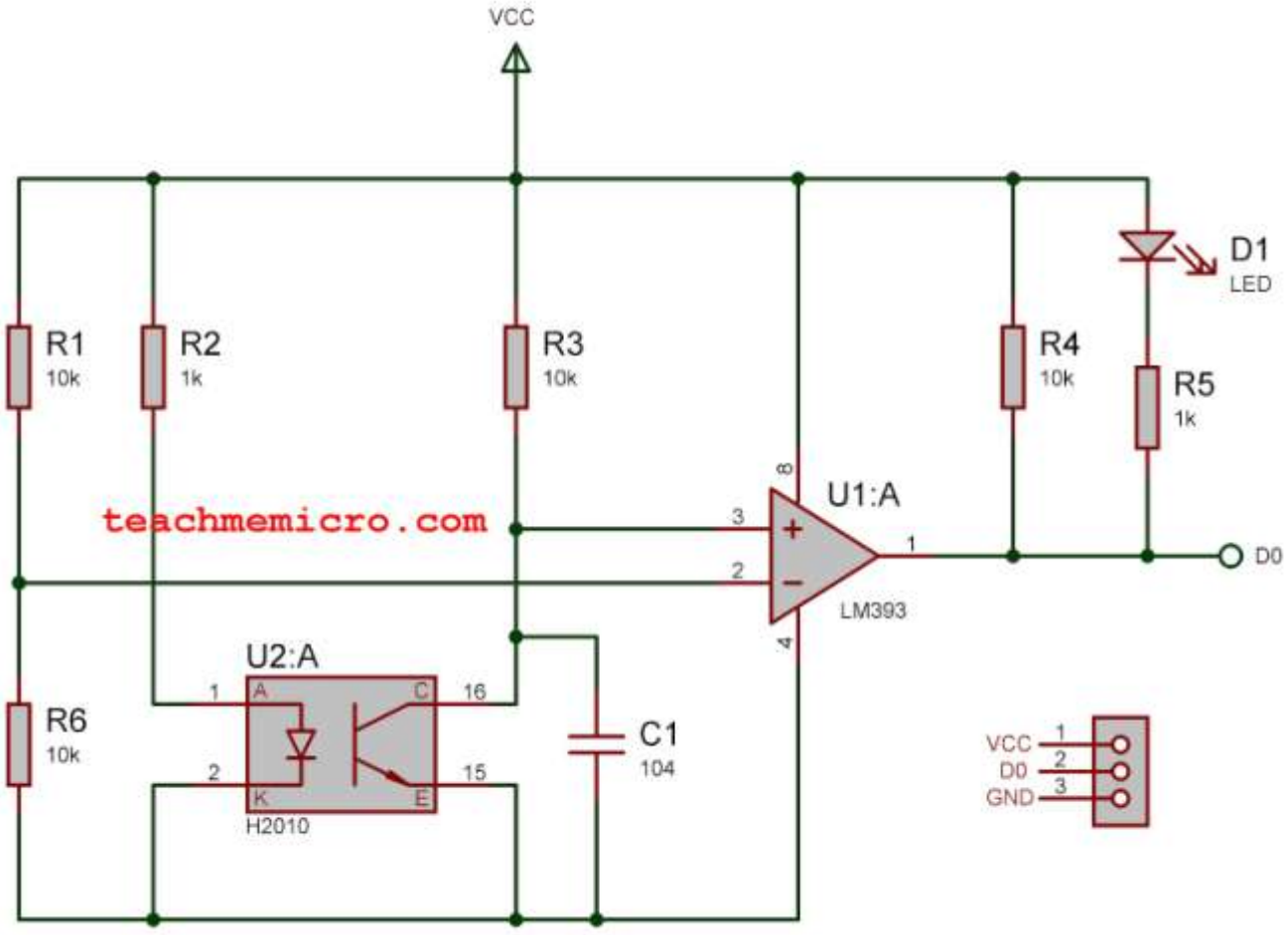
Il modulo ha due colonne verticali con un LED IR su una colonna e un fototransistor sull'altra.
Ogni volta che il percorso tra il LED IR e il fototransistor viene interrotto, il pin D0 va alto (0→5V).





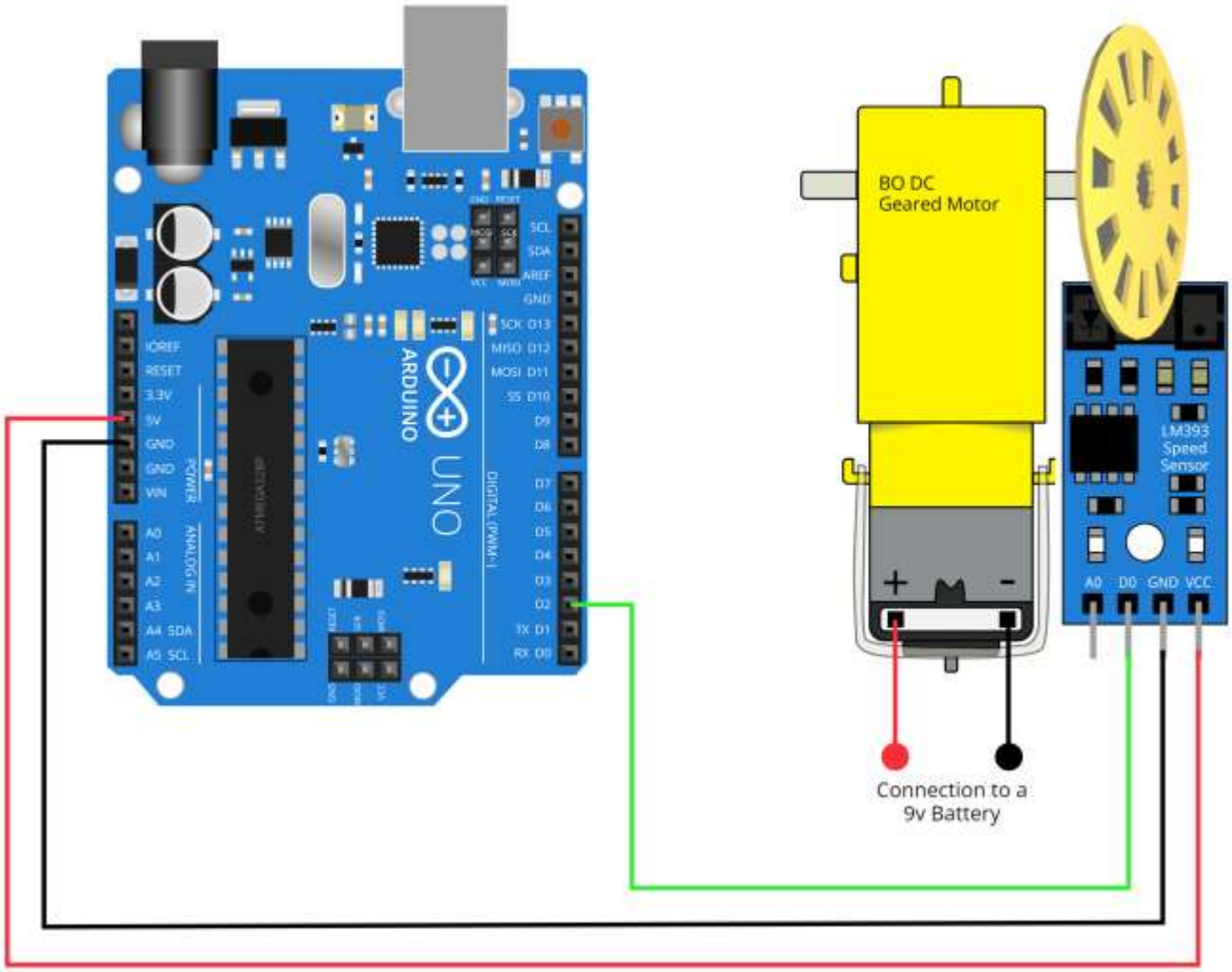
Questo modulo viene chiamato impropriamente "sensore di velocità LM393".

LM393 è in realtà un comparatore e non un sensore. Lo schema del modulo è il seguente:



Senza ostruzioni tra il LED IR e il fototransistor, la tensione tra i terminali positivo e negativo del comparatore è uguale.
Quando il fototransistor è bloccato, assorbirà una tensione maggiore, portando il terminale positivo del comparatore ad una tensione maggiore del terminale negativo.
Pertanto, una tensione positiva, pari a VCC, sarà disponibile al terminale D0.

La figura sottostante mostra una tipica apllicazione del modulo per rilevare il numero di giri del motore di un drone.

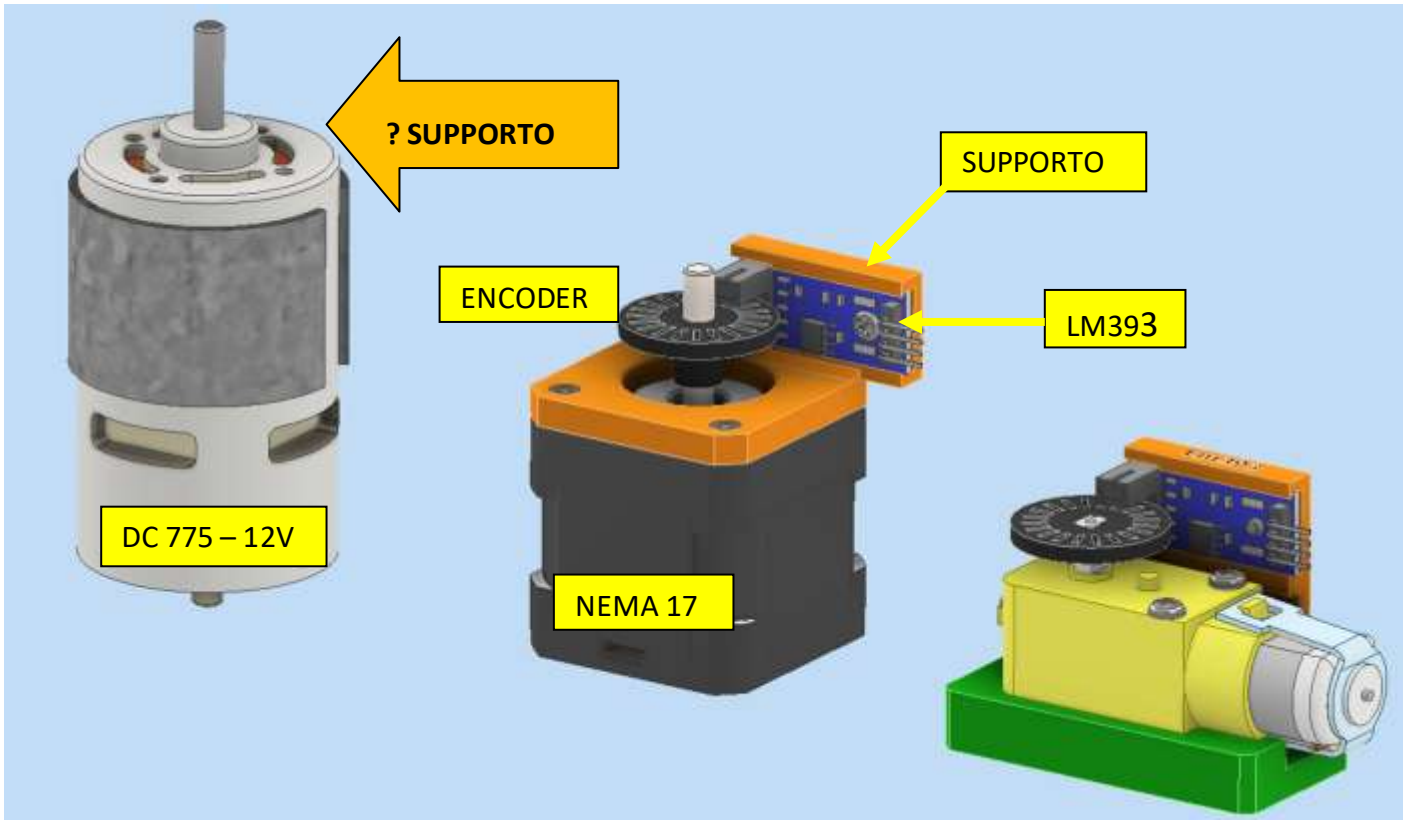


Un disco forato (20 fori) è fissato sull'albero del motore CC.

Il modulo LM393 è collegato al motore in modo che il disco possa ruotare liberamente tra le forcelle del snesore.

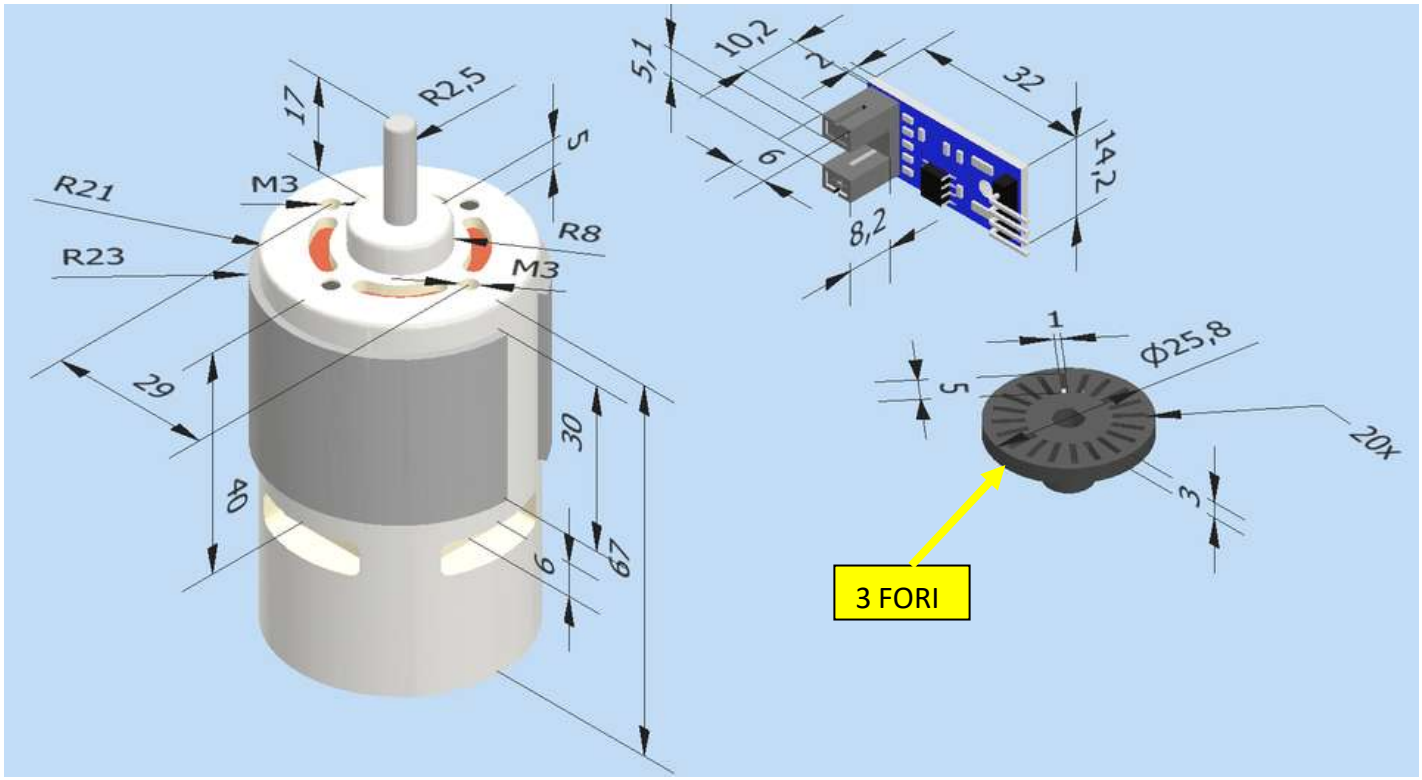
Ogni volta che il disco interrompe il raggio IR del LED il fototransistor rileva l'inerruzione e il pin D0 va alto (0→5V).

DISEGNARE IL SUPPORTO PER IL MODULO IR LM393 E IL DISCO FORATO (ENCODER)



NOTA:
Per rilveare velocità elevate del motore è necessario ridurre il numero di fori presenti nell'encoder.
Con soli 3 fori si rilevano velocità oltre I 6000 diri/minuto.

***Domanda**: quale svantaggio si ha riducendo il numero di fori sul disco?*



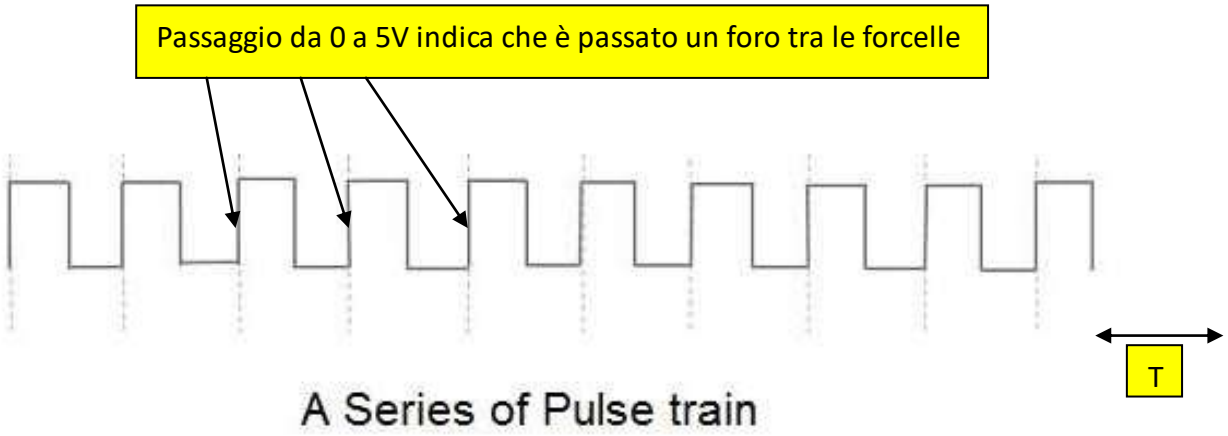
ESERCITAZIONE

Ricavare dalla fotografia del circuito con il motore collegato lo schema elettrico del sistema di regolazione della velocità del motore e replicarlo su Thinkercad.

Simulare il modulo LM393 con un generatore di funzioni d'onda quadra 0-5V (regolare la frequenza a bassi valori).

Scrivere Il programma Arduino che conta gli impulsi che arirvano dal modulo LM393 (simulato con generatore funzioni d'onda).

Sapendo che il disco collegato all'albero presenta 3 fori calcolare e mostrare su seriale e/o display LCD 16x2 I2C il numero di giri del motore.



NOTA:

Contare gli impulsi solo quando viene rilevato il passaggio da 0V a 5V (fronte di salita) e NON ogni volta che si rilevano 5V!

Il tempo di campionamento deve essere compatibile con il numero di giri del motore.

Ad esempio:

con 6000 giri/minuto

→ 100 giri /secondo

→ x 3 fori

→ 300 impulsi /secondo

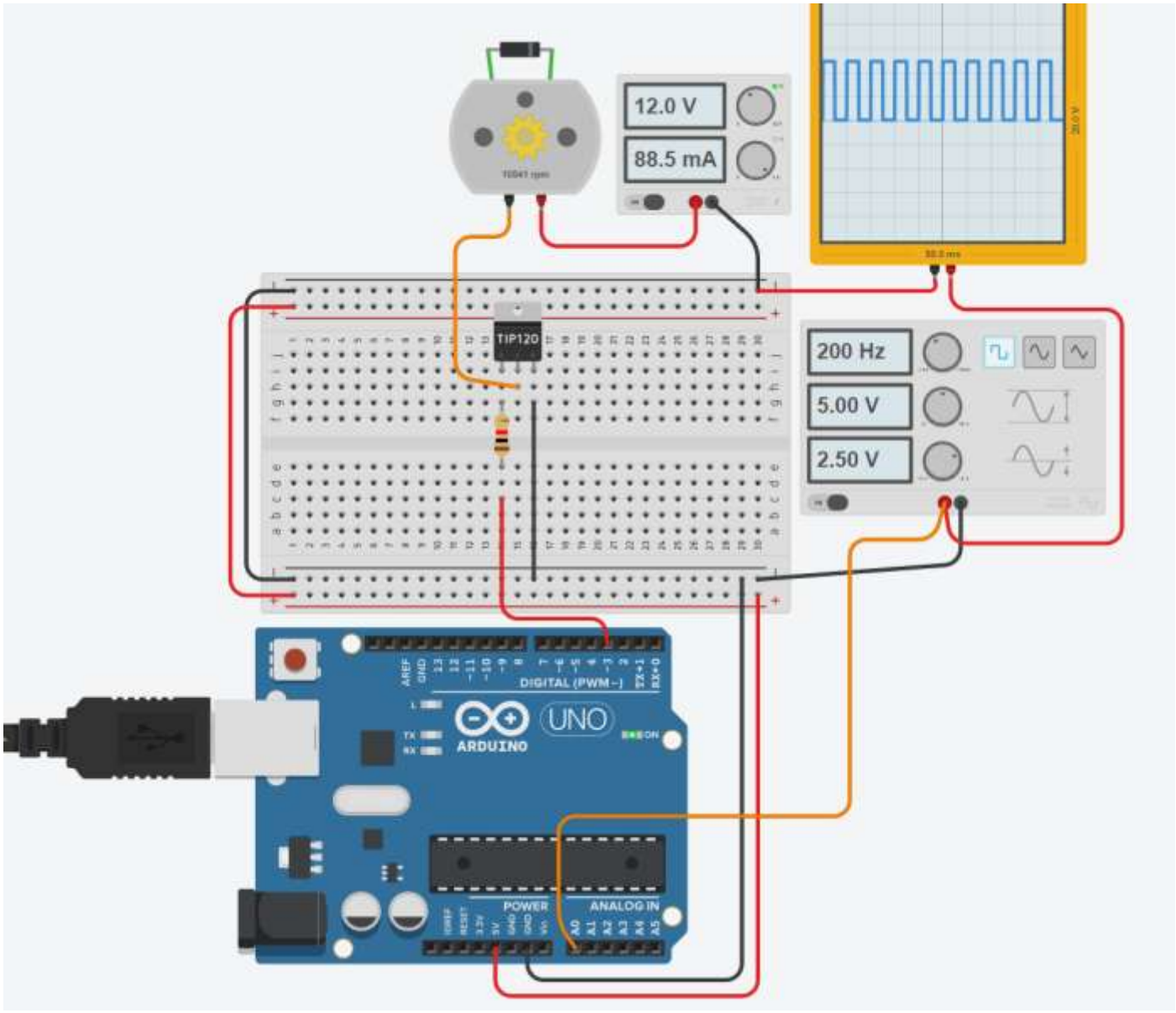
→ periodo T= 1/300= 3ms

→ campionamento ogni 1ms per non perdere impulsi.

SCHEMA THINKERCAD

Simulare il modulo LM393 con un generatore di funzioni d'onda quadra 0-5V.
Mantenere la frequanza bassa per non appesantire la simulazione nel cloud.
Visulaizzare sull'oscilloscopio il treno di impulse del generatore d'onda.

Prevedere la presenza del diodo di protezione del transitor.

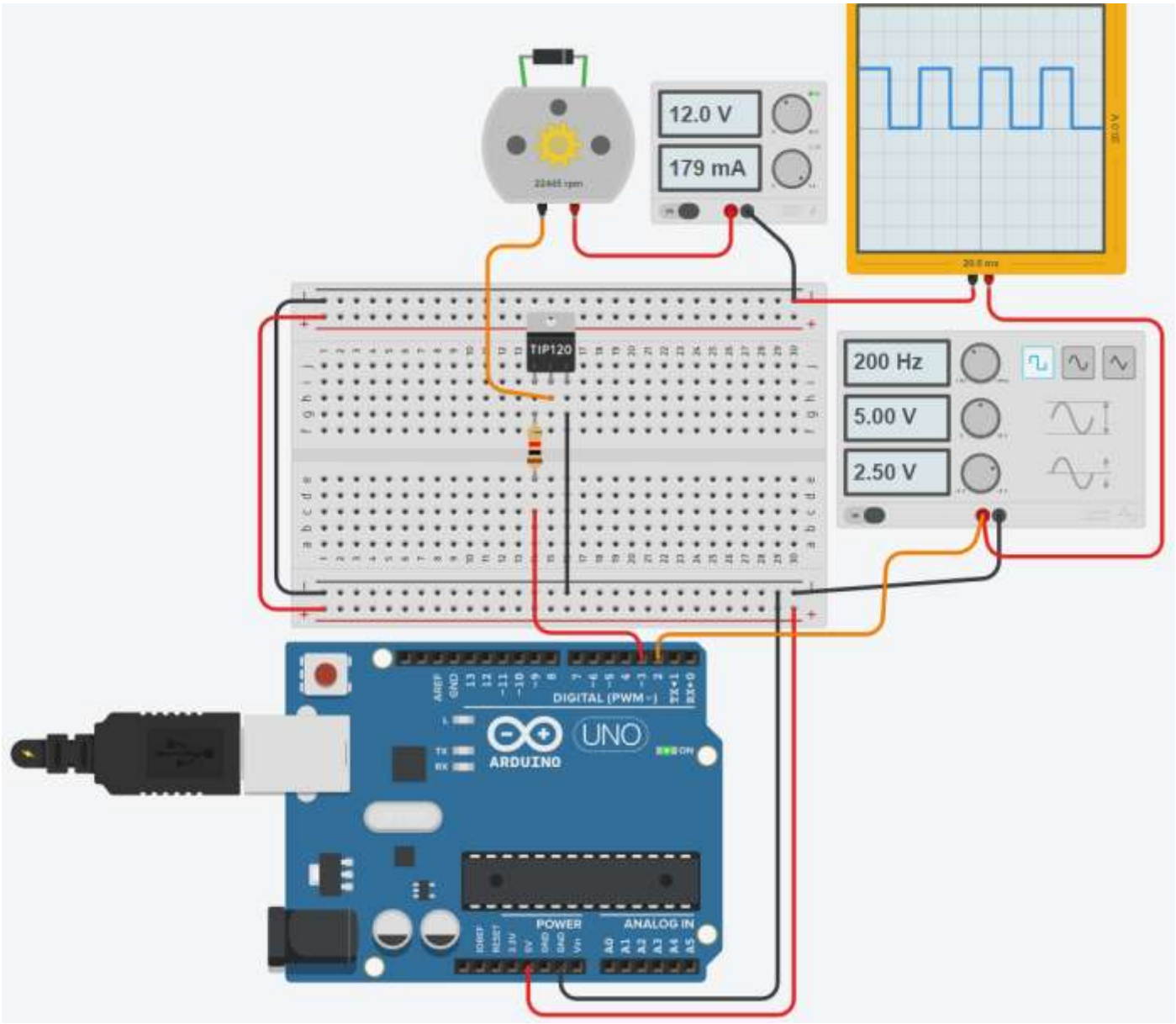


CODICE

```
// PIN
int transistorPin = 3;
int sensorePin = A0;

//volatile -->
//salvata in RAM perchè modificabile in + thread
volatile long counter=0;
volatile long counter_tot=0;

// stato del sensore
int nfori= 3;
int stato=0;
int stato_prec=1;
```

```
int stop_read= 0;
int delta_t= 1000;
long t;
// seriale
int incomingByte = 0; // for incoming serial data
int  input = 0;

void setup() {
  pinMode(A0,INPUT);
  pinMode(transistorPin, OUTPUT);
  Serial.begin(9600);
  t= millis();
  analogWrite(transistorPin, 255/2);
}

void loop() {
  if (Serial.available() > 0) {
    incomingByte = Serial.read();  // leggo carattere
    input = incomingByte - 48; //converto codice ASCII carattere in numero 1,2,3

  switch (input) {
    case 0:
      analogWrite(transistorPin, 0);
      break;
    case 1:
      analogWrite(transistorPin, 255/3);
      break;
    case 2:
      analogWrite(transistorPin, 255/2);
      break;
    case 3:
      analogWrite(transistorPin, 255);
      break;
  }
    input=0;
  }

  // stato sensore
  stato= digitalRead(sensorePin);
  // se è cambiato rispetto a prima
  if (stato!=stato_prec && stop_read==0) {
     // se passo da 0-->5
     if (stato==1) counter = counter+ 1;
     stato_prec= stato; // aggiorno stato_prec
  }

  // durante la stampa non leggo impulsi per non sfalsare calcolo)
  if ((millis() - t)>=delta_t) {
            stop_read= 1;
            counter_tot= counter_tot + counter;
    // rpm con delta_t= 1 sec
            float rpm= counter *20.0; //counter * 60/nfori
            Serial.print("rpm "); Serial.println(rpm);
            counter= 0;
            t= millis();
            stop_read= 0;
   }

//delayMicroseconds(10);
}
```

SCHEMA THINKERCAD CON UTILIZZO DEGLI INTERRUPT




```
// PIN
int transistorPin = 3;
// L'interrupt 0 di Arduino è associato al pin digitale 2
int sensorePin = 2;  // --> interrupt

//volatile --> salvata in RAM perchè modificabile in un interrupt
volatile long counter=0;
volatile long counter_tot=0;

// stato del sensore
int nfori= 3;
int delta_t= 1000;
long t;

// seriale
int incomingByte = 0; // for incoming serial data
int  input = 0;

void setup() {
  attachInterrupt(0,countpulse,RISING); //interrupt PIN 2
```

```
  pinMode(A0,INPUT);
  pinMode(transistorPin, OUTPUT);
  Serial.begin(9600);
  analogWrite(transistorPin, 255);
  t= millis();
}

void loop() {
  if (Serial.available() > 0) {
    incomingByte = Serial.read();  // leggo carattere
    input = incomingByte - 48; //converto codice ASCII carattere in numero 1,2,3

  switch (input) {
    case 0:
      analogWrite(transistorPin, 0);
      break;
    case 1:
      analogWrite(transistorPin, 255/3);
      break;
    case 2:
      analogWrite(transistorPin, 255/2);
      break;
    case 3:
      analogWrite(transistorPin, 255);
      break;
    }
    input=0;
  }

  // durante la stampa non leggo impulsi per non sfalsare calcolo)
  if ((millis() - t)>=delta_t) {
    // rpm con delta_t= 1 sec
            float rpm= counter *20.0; //counter * 60/nfori
            Serial.print("rpm "); Serial.println(rpm);
            counter= 0;
            t= millis();
   }
   //delayMicroseconds(10);
}

void countpulse(){
  counter++;
}
```

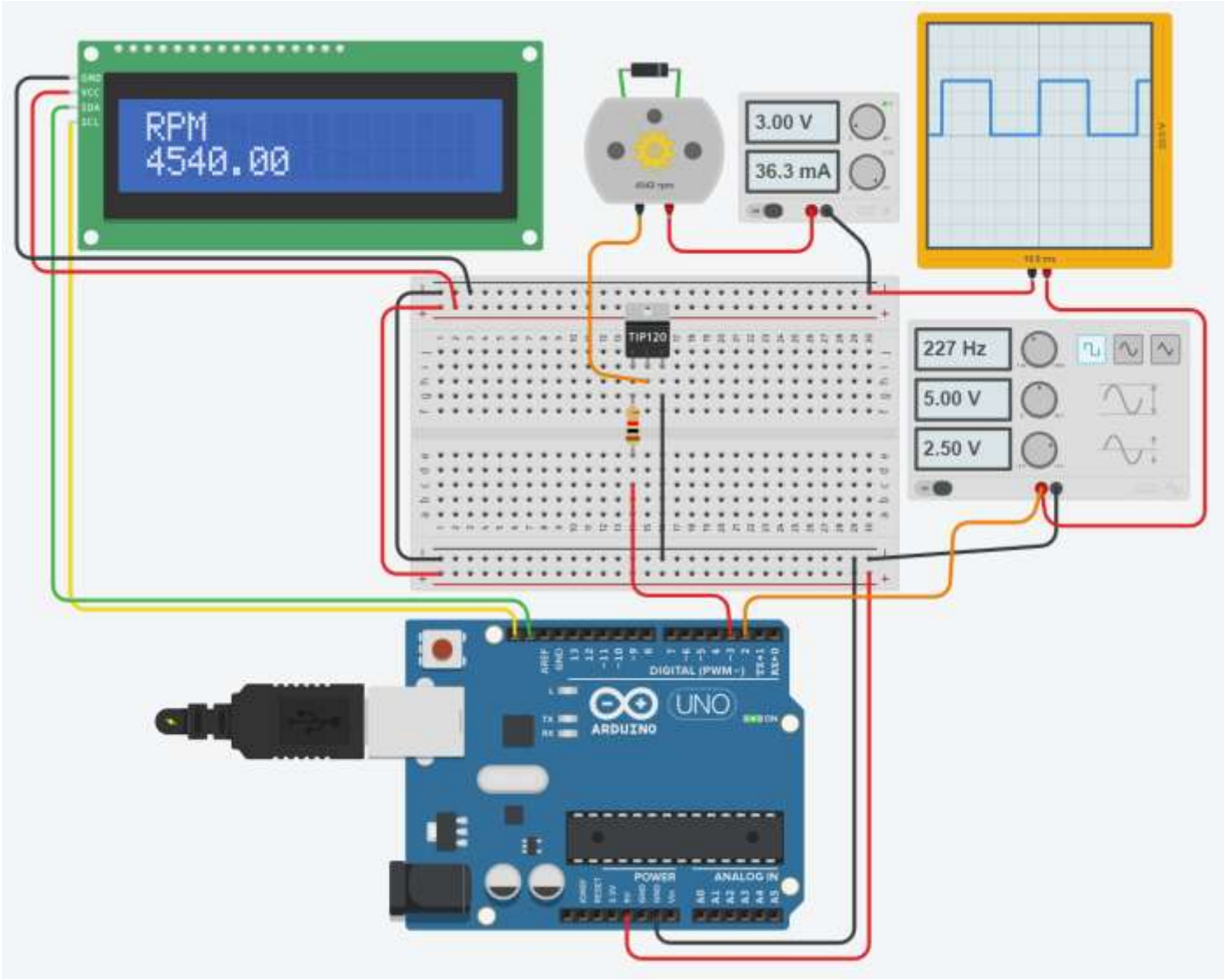
Molti dei sensori per hobby a basso costo hanno comparatori basati sull'LM393 senza alcun feedback di isteresi e non è raro riscontrare interruzioni multiple/problemi di rumore con questi moduli comparatori LM393 di base.

Quattro suggerimenti da provare per migliorare il funzionamento del modulo:

- routine di antirimbalzo nell'interrupt
- stabilizzare il punto di trigger con un condensatore di C=0. 1µF tra GND e l'ingresso negativo del comparatore.
- aggiungere un filtro passa-basso all'uscita D0 del modulo R=1K, C=0.01µF (100nF).
- Alimentare il modulo a 3.3V al posto di 5V

Anche solo un condensatore C= 0,01 µF (100 nF) tra uscita D0 e GND del modulo migliora le cose.

SCHEMA THINKERCAD CON LCD 16X2 E CON UTILIZZO DEGLI INTERRUPT



CODICE

```
#include <Adafruit_LiquidCrystal.h>
Adafruit_LiquidCrystal lcd_1(0);

// PIN
int transistorPin = 3;
// L'interrupt 0 di Arduino è associato al pin digitale 2
int sensorePin = 2;  // --> interrupt

//volatile --> salvata in RAM perchè modificabile in un interrupt
volatile long counter=0;
volatile long counter_tot=0;

// stato del sensore
int nfori= 3;
int delta_t= 1000;
long t;

// seriale
int incomingByte = 0; // for incoming serial data
int  input = 0;
```

```
void setup() {
  attachInterrupt(0,countpulse,RISING); //interrupt PIN 2
  pinMode(A0,INPUT);
  pinMode(transistorPin, OUTPUT);
  Serial.begin(9600);

  lcd_1.begin(16, 2);
  lcd_1.setCursor(0, 0);
  lcd_1.print("RPM");
  lcd_1.setCursor(0, 1);
  lcd_1.print("giri");

  analogWrite(transistorPin, 255);
  t= millis();
}

void loop() {
  if (Serial.available() > 0) {
    incomingByte = Serial.read();  // leggo carattere
    input = incomingByte - 48; //converto codice ASCII carattere in numero 1,2,3

  switch (input) {
    case 0:
      analogWrite(transistorPin, 0);
      break;
    case 1:
      analogWrite(transistorPin, 255/3);
      break;
    case 2:
      analogWrite(transistorPin, 255/2);
      break;
    case 3:
      analogWrite(transistorPin, 255);
      break;
    }
    input=0;
  }


  // durante la stampa non leggo impulsi per non sfalsare calcolo)
  if ((millis() - t)>=delta_t) {
    // rpm con delta_t= 1 sec
            float rpm= counter *20.0; //counter * 60/nfori
            Serial.print("rpm "); Serial.println(rpm);

            lcd_1.setCursor(0, 1); lcd_1.print(rpm);

            counter= 0;
            t= millis();
   }
   //delayMicroseconds(10);
}


void countpulse(){
  counter++;
}
```

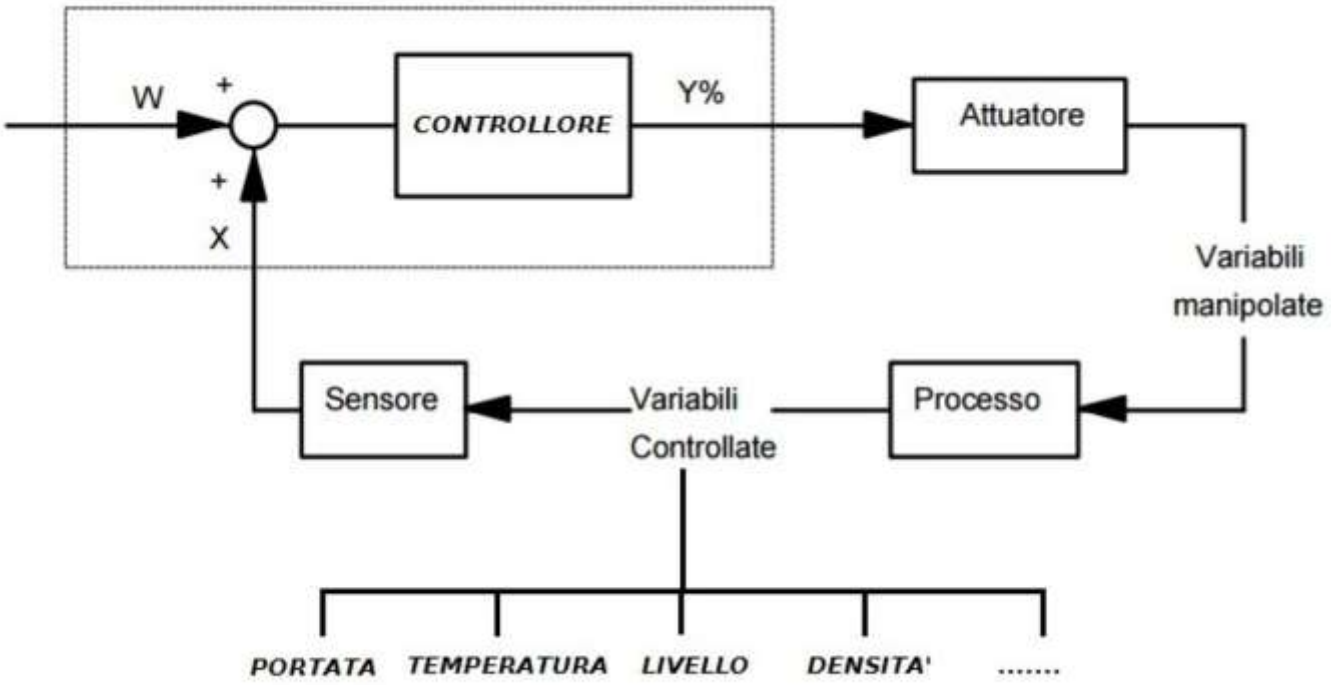
SISTEMI DI CONTROLLO

Un sistema di ***REGOLAZIONE*** non prevede l'utilizzo di sensori ma si base sulle leggi fisiche che governano il sistema.
Ad esempio per mantenere una certa temperatura dell'acqua in un recipiente è possible far ricorso alle leggi della termotecnica per calcolare la dispersioni termiche dell'involucro e quindi la potenza termica che deve essere fornita tramite un elemento riscaldante.
Applicando correttamente le leggi della fisica si può ottenere il risultato richiesto senza un controllo continuo del sistema.

Nel caso di presenza di disturbi esterni (non adiabaticità del recipiente, prelievo di acqua, ecc.) il risultato non può essere raggiunto senza la presenza di opportune ***sensori*** che misurano in tempo reale la grandezza da controllare.

Sulla base della misura si interviene poi degli ***attuatori*** per portare la grandezza controllata al livello desiderato (retroazione).
In questa situazione si parla quindi di sistema di **CONTROLLO AD ANELLO CHIUSO**.

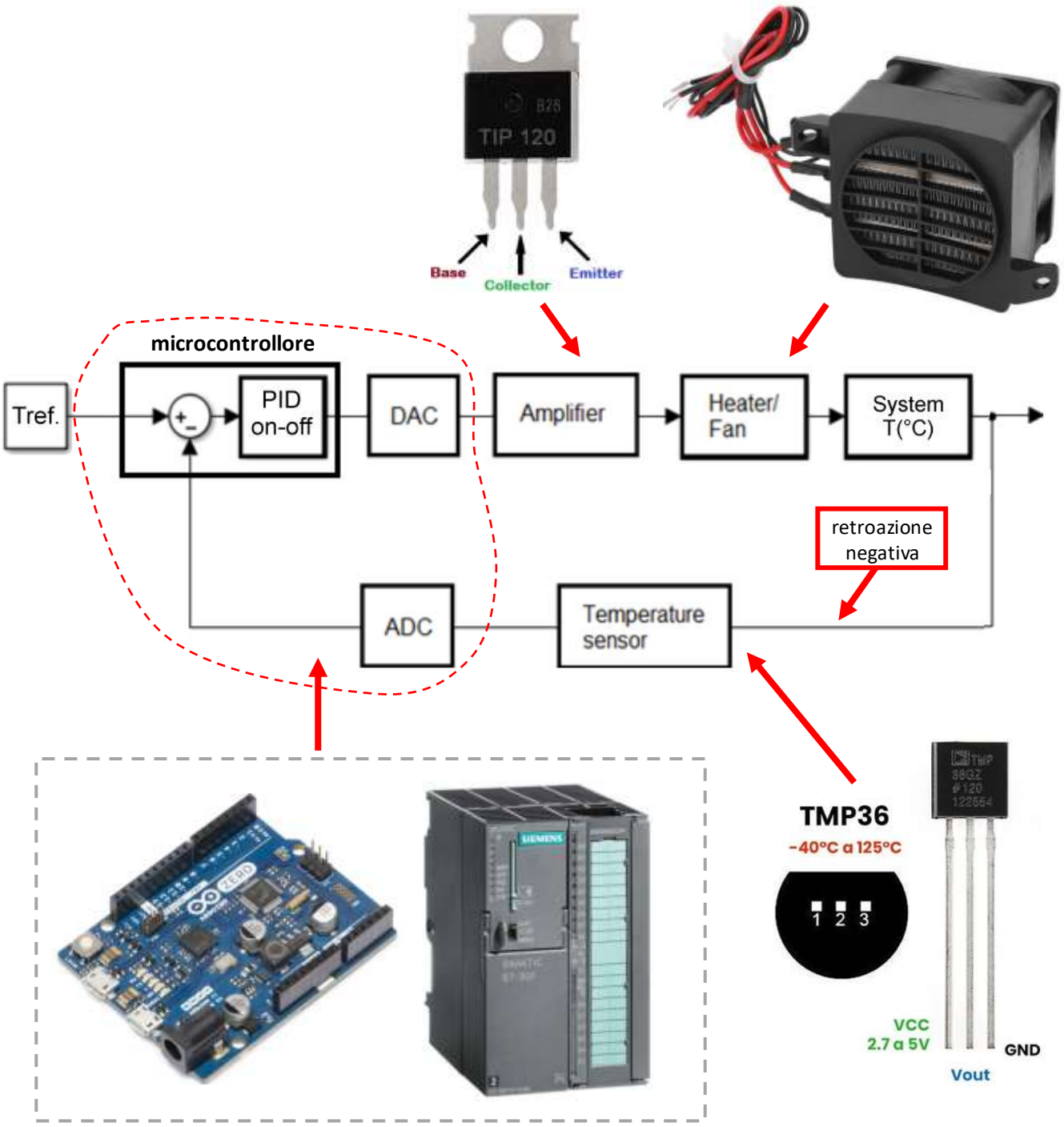
L'immagine seguente mostra lo schema di massima di un generico sistema di controllo.



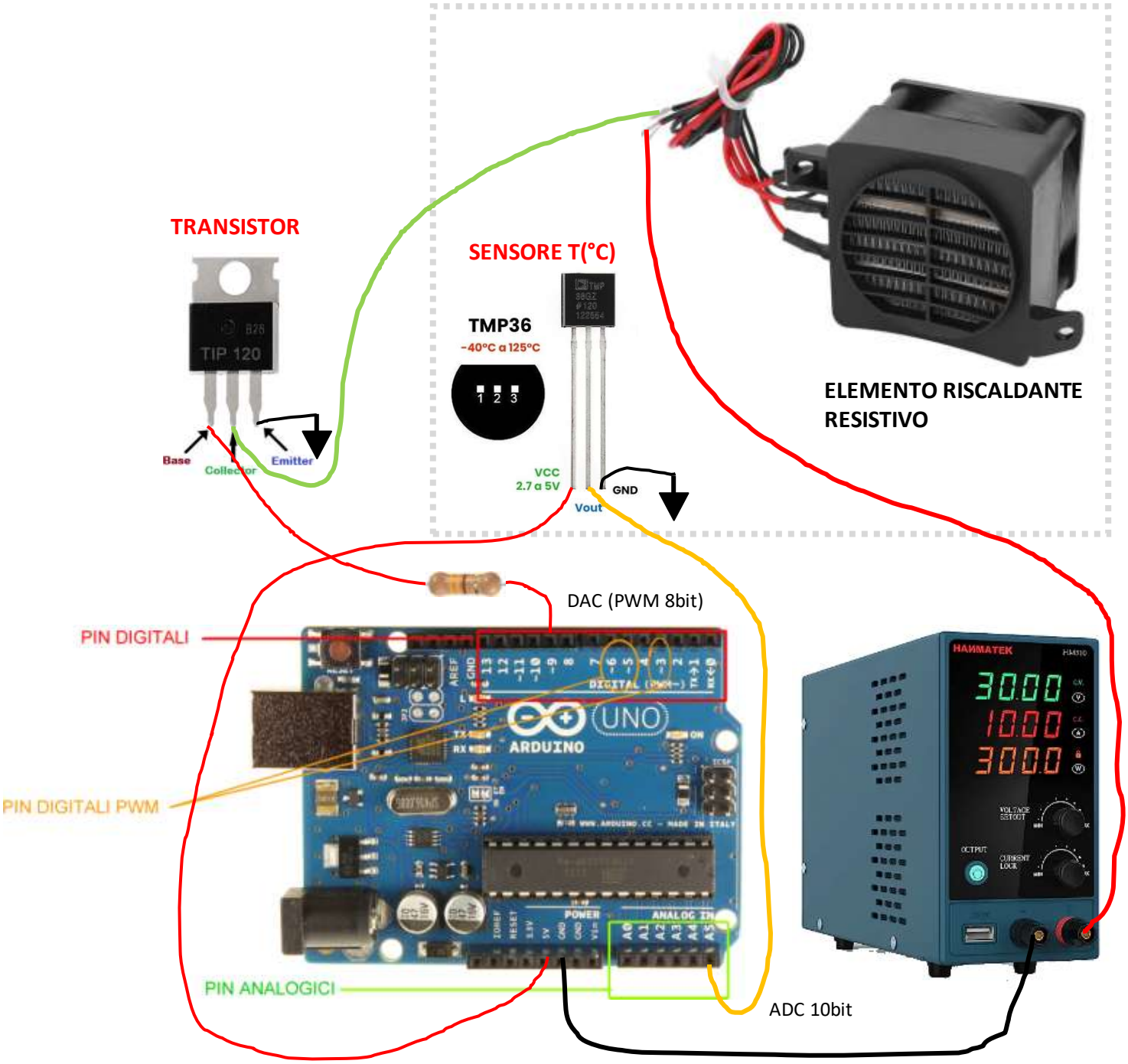
SCHEMA A BLOCCHI DI SISTEMA DI CONTROLLO DI TEMPERATURA

La figura sottostante mostra lo schema di un sistema di controllo della temperatura dove abbiamo:

- Tref = temperatura di riferimento ("set-point") da mantenere nel sistema controllato (es. forno)
- PID / ON-OFF = tipologia di controllo attuata (proporzionale-integrale-derivativo, acceso-spento)
- DAC = convertitore da digitale ad analogico (converte un numero in una tensione → 8bit → 0-255)
- Amplificatore = amplifica il segnale in uscita dal DAC (es. Transistor BJT)
- Attuatore = elemento riscaldante che serve ad aumentare la temperature del sistema controllato
- Sensore = termistore, termoresistenza, termocoppia ecc.
- ADC = convenrtitore da analogico (temperatura/tensione) a digitale (numero digitale → 10bit → 0-1024)



ESEMPIO CONTROLLO PID CON TRANSISTOR

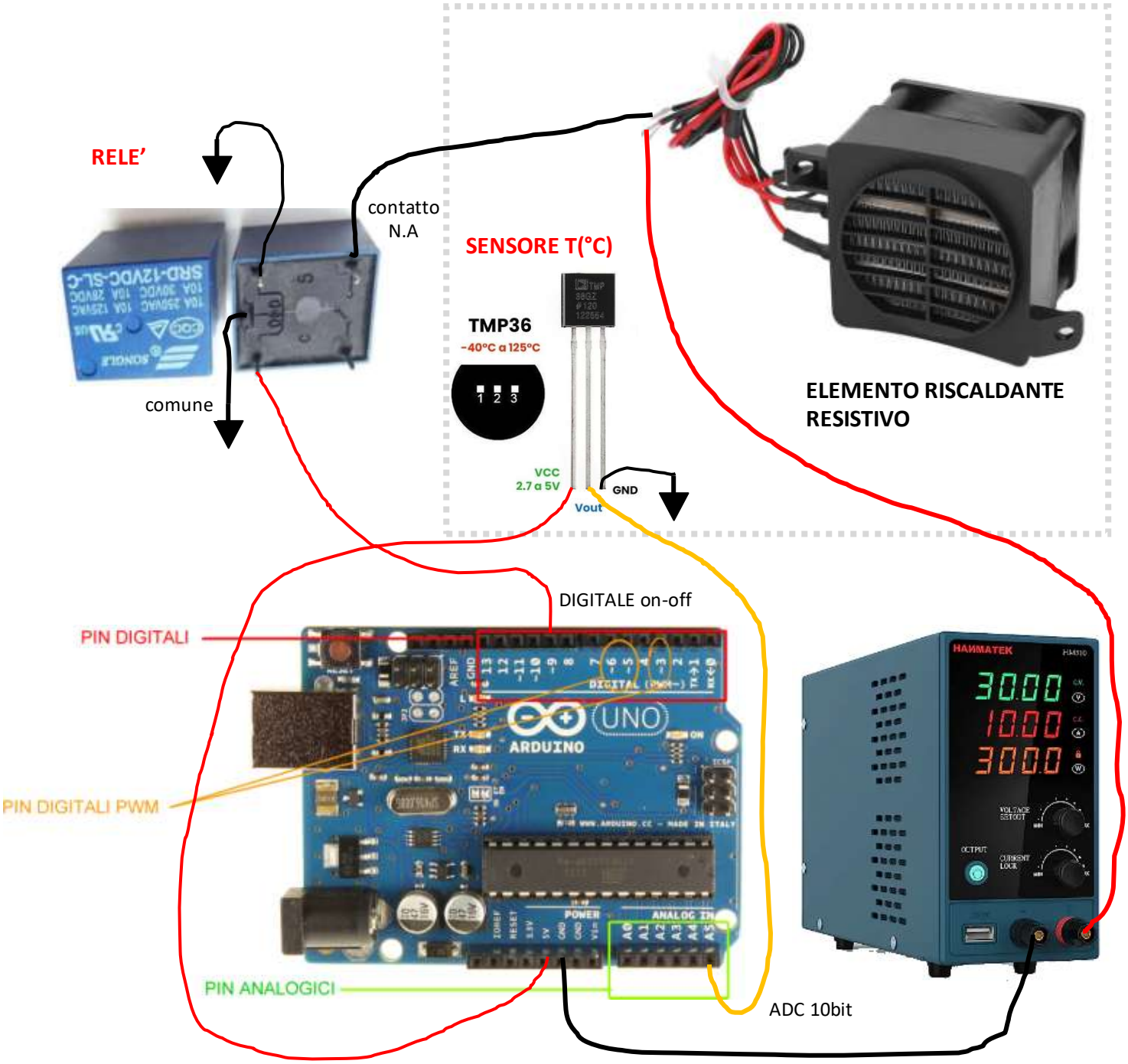


Il microcontrollore e l'alimentatore di potenza devono sempre avere la massa in comune.

Il sensore e la base del transistor vengono alimentati dal microcontrollore.

L'attuatore è alimentato dal generatore di potenza e collegato al collettore del transistor.

ESEMPIO CONTROLLO ON-OFF CON RELE'



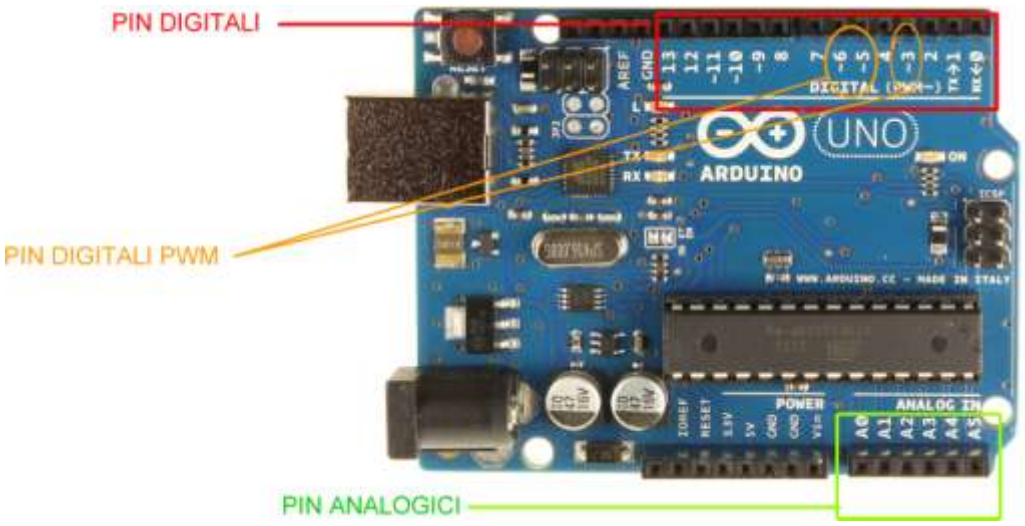
Il microcontrollore e l'alimentatore di potenza devono sempre avere la massa in comune.

Il sensore e la bobina del rele' vengono alimentati dal microcontrollore.

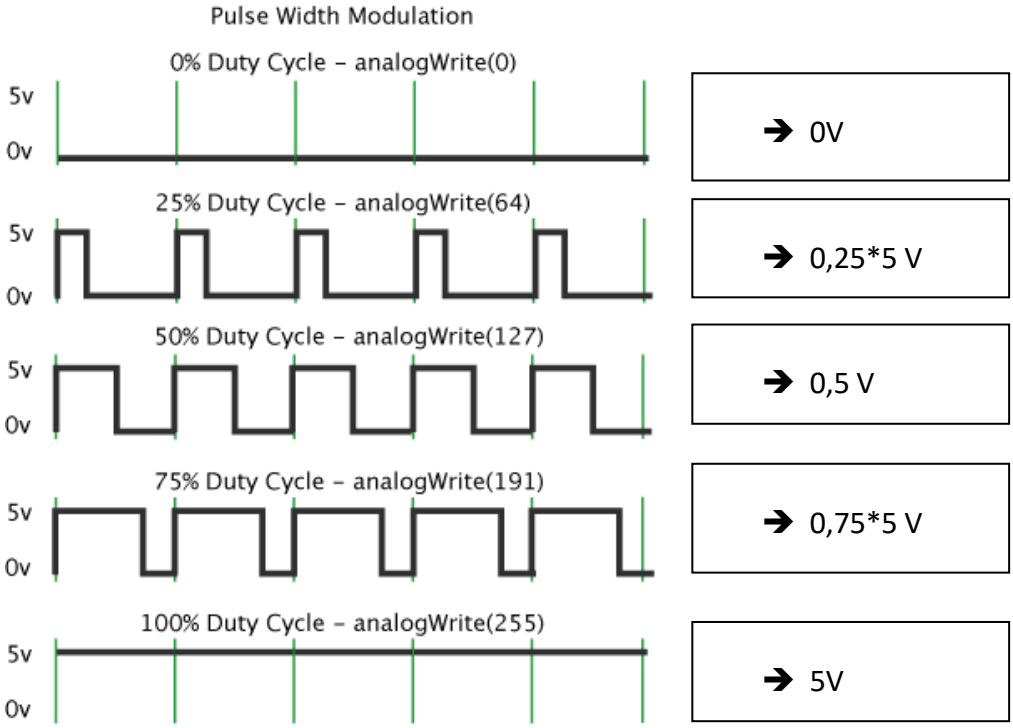
L'attuatore è alimentato dal generatore di potenza e collegato al collettore del transistor.

GENERARE SEGNALI ANALOGICI (DAC) CON ARDUINO

I pin digitali si dividono in base al supporto o meno della funzione PWM (pulse wide modulation).
I pin che hanno la PWM sono: 3,5,6,9,10,11.



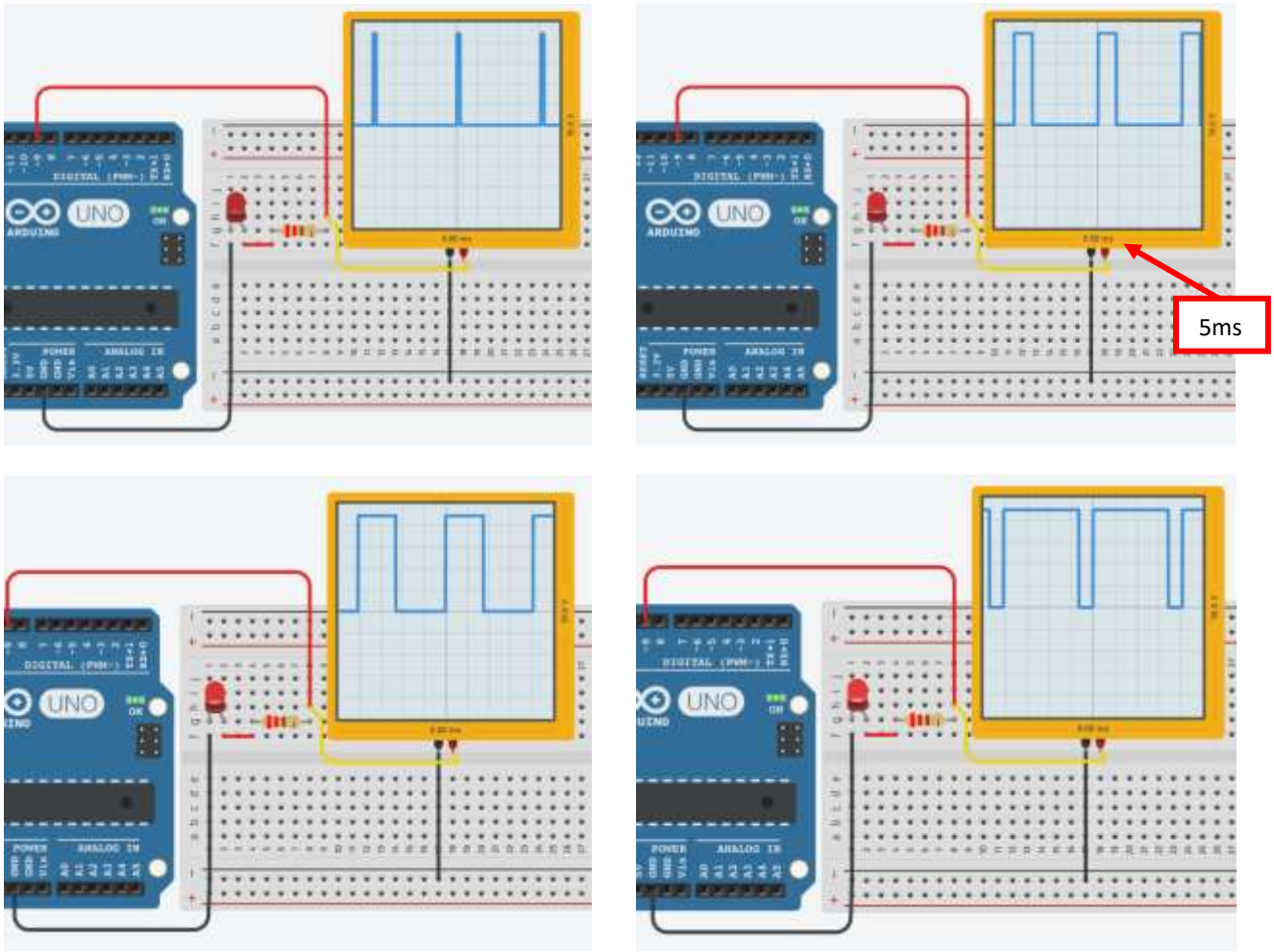
Con un pin PWM è possibile generare in uscita un segnale analogico da 0-5V con una risoluzione di 8 bit (5/255 volt ≅0,02V) .
Un segnale PWM è in termini molto semplicistici, un onda quadra 0-5V (ad alta frequenza) con delle durate prestabilite per la parta alta (5V).
Ciò permette di simulare un valore analogico di tensione compreso tra 0-5V con uno digitale con la maggior parte degli attuatori (transistor, relè, motori CC ...).



I segnali analogici in uscita sono di fondamentale importanza per poter effettaure sistemi di controllo evoluti come ad esempio il PID.
Tramite un segnale analogico che varia da 0 a 5V si può regolare la potenza assorbita da un attuatore (motori, elementi riscadanti, generatori di forza ecc.) e di conseguenza l'effetto sul sistema controllato.

ESERCIZIO: VARIARE LA LUMINOSITA' DI UN DIODO LED

Tramite la PWM si può variare la corrente che scorre in un diodo LED (variando la tensione sulla resistenza) e di conseguenza la sua luminosità. Questa tecnica viene usata nelle lampadine in CC trimmerabili.

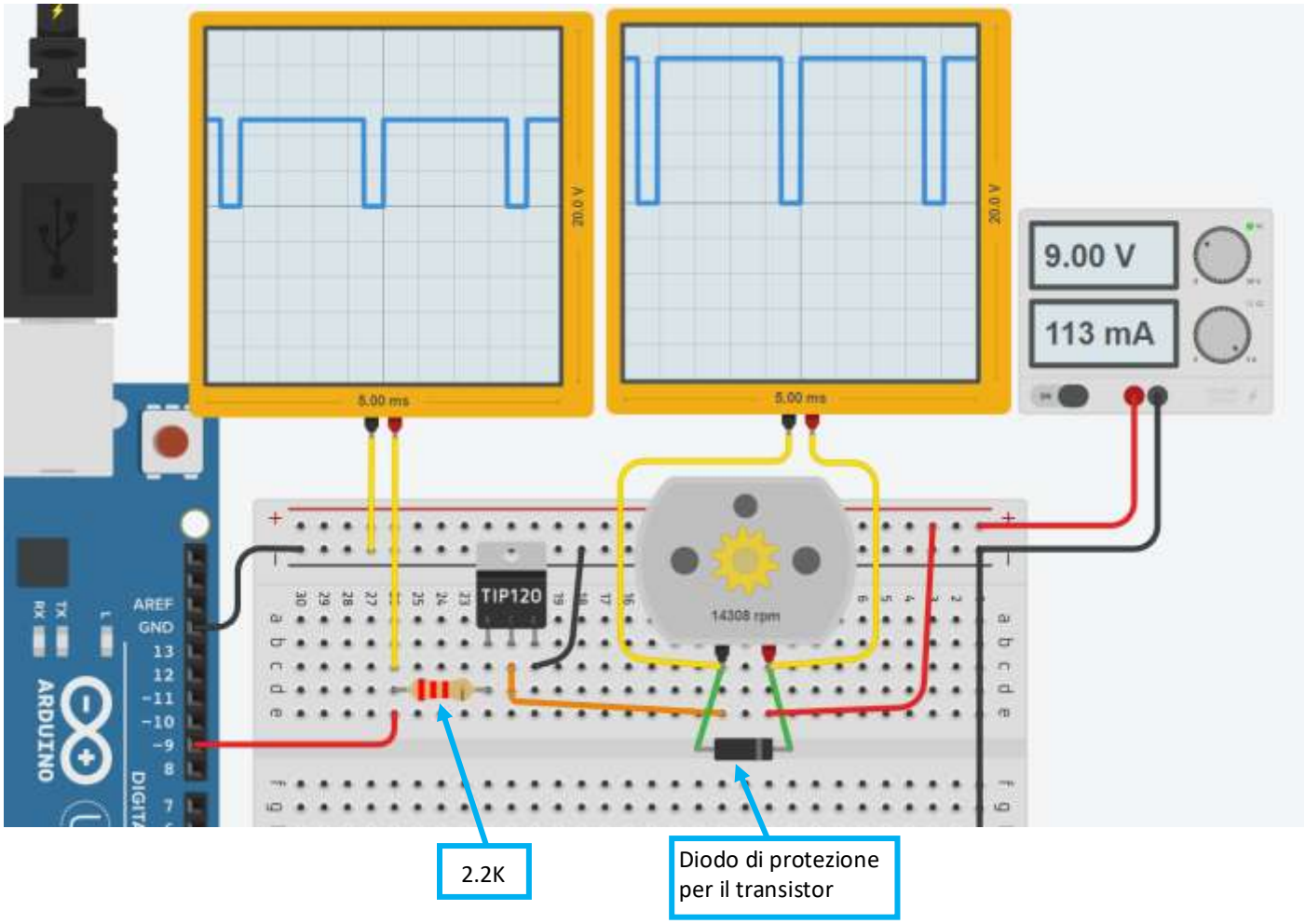


CODICE:

```
int ledPin = 9;  // LED su Pin 9

void setup(){
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
}

void loop(){
 // aumento luminosità da 0 al massimo
  for(int dimValue = 0; dimValue <= 255; dimValue = dimValue + 5){
    analogWrite(ledPin, dimValue);
    Serial.println(dimValue);
    delay(30);
  }
// diminuisco luminosità dal massimo a 0
  for(int dimValue = 255; dimValue >= 0; dimValue = dimValue - 5){
    analogWrite(ledPin, dimValue);
    Serial.println(dimValue);
    delay(30);
  }
}
```

COME VARIARE LA VELOCITA' DI UN MOTORE C.C. MANTENENDO ALTA LA COPPIA MOTRICE

Tramite la PWM si può variare la corrente che scorre nel motore e di conseguenza la sua velocità.
Poichè la corrente assorbita dal motore è superiore ai 30-40 mA fornibili da Arduino è necessario utilizzare un transistor che viene comandato da Arduino tramite un segnale PWM.



CODICE

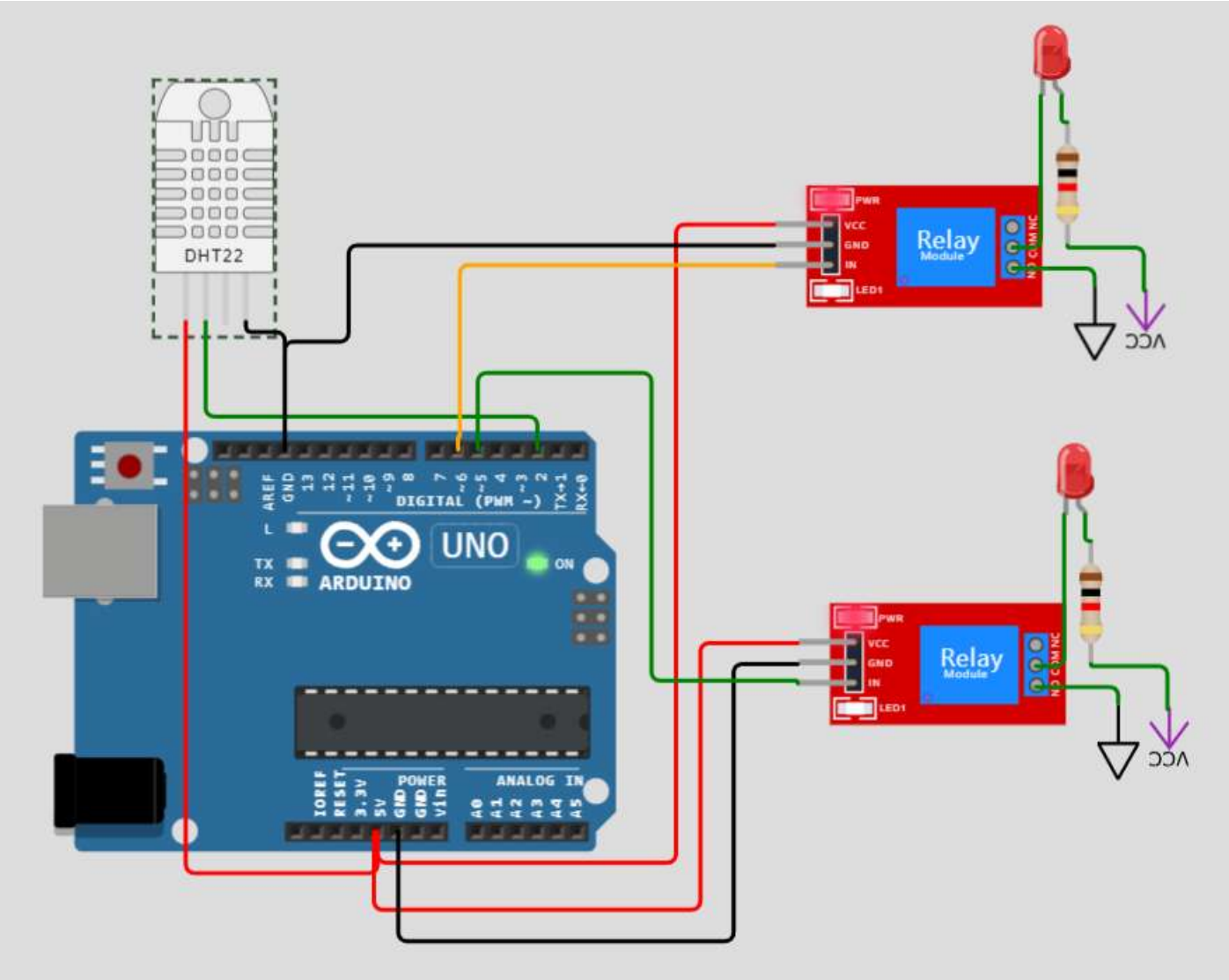
```
#define DC_MOTOR_PIN 9

void setup() {
  pinMode( DC_MOTOR_PIN, OUTPUT );
}

void loop() {
 for( int i = 0; i < 255; i=i+5 ){
  analogWrite(DC_MOTOR_PIN, i);
  delay(50);
 }

 for( int i = 255; i > 0; i=i-5 ){
  analogWrite(DC_MOTOR_PIN, i);
  delay(50);
 }

}
```

SISTEMA DI CONTROLLO TEMPERATURA E UMIDITA'

Si vuole controllare la temperatura e l'umidità in un locale in modo da mantenerla nelle condizioni di benessere (20°C +-2°C e umidità relativa del 50% +-10%.

Utilizzare come sensore un DHT22 e simulare il sistema di riscaldamento e deumidificazione (si ipotizzi presenza di persone che aumentano sempre l'umidità del locale) tramite dei led comandato da rele'.

Prendendo spunto dal programma allegato integrarlo per adempiere alle richieste.



simulabile su "wokwi.com"

CODICE

```
#include "DHT.h"

const int DHTPIN=2;
const int pinLed1=6;
const int pinLed2=5;

DHT dht(DHTPIN, DHT22); // DHT 22  (AM2302), AM2321

int Tsp= 20;  // temperatura set point
int Hsp= 50;  // umidità set point

void setup() {
  Serial.begin(115200);
  Serial.println(F("DHT22 example!"));

  pinMode(DHTPIN, INPUT);
  pinMode(pinLed1, OUTPUT);
  pinMode(pinLed2, OUTPUT);

  dht.begin();
}

void loop() {
  float temperature = dht.readTemperature();
  float humidity = dht.readHumidity();

  // Check if any reads failed and exit early (to try again).
  if (isnan(temperature) || isnan(humidity)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }

  Serial.print(F("Humidity: "));
  Serial.print(humidity);
  Serial.print(F("%  Temperature: "));
  Serial.print(temperature);
  Serial.println(F("°C "));


  digitalWrite(pinLed2, HIGH);

  if (temperature>=20) {digitalWrite(pinLed1, LOW);}
  else {digitalWrite(pinLed1, HIGH);}

  if (humidity>=50) {digitalWrite(pinLed2, LOW);}
  else {digitalWrite(pinLed2, HIGH);}

  // Wait a few seconds between measurements.
  delay(2000);
}
```

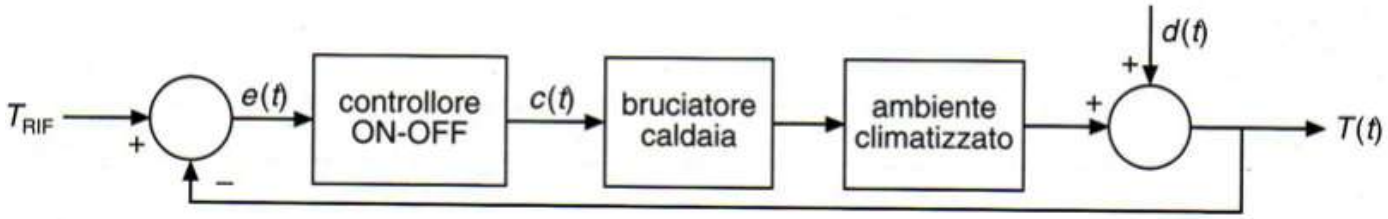
SISTEMA DI CONTROLLO ON-OFF

Il controllo ON-OFF è un controllo ad anello chiuso nel quale l'azione del controllore è discontinua.

Il controllore decide quando intervenire in base alla misura dello scostamento tra valore atteso e valore reale dell'uscita come nel controllo continuo, con la differenza che l'aggiustamento non viene applicato con continuità bensì quando lo scostamento oltrepassa una soglia predeterminata.

Prendiamo come esempio il controllo ON-OFF di temperatura per la climatizzazione di un ambiente, dove lo scopo del controllo è quello di mantenere il livello di temperatura di una stanza entro margini prestabiliti rappresentati da una soglia inferiore Tinf e una superiore Tsup.

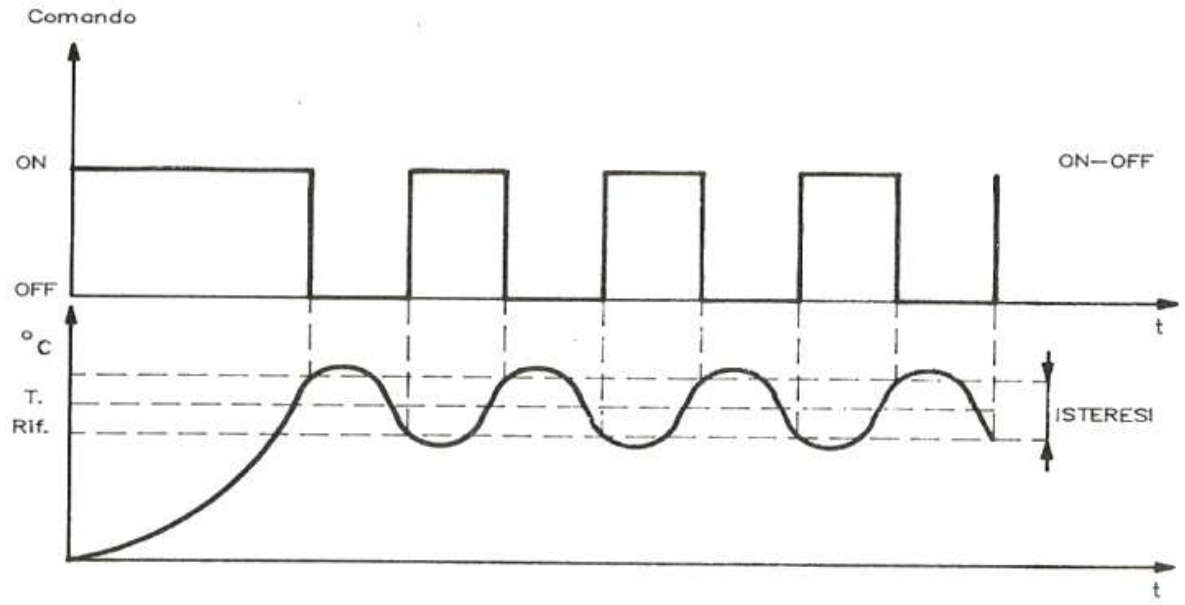
Il valore della variabile in uscita T(t) viene confrontato con il valore desiderato Trif.
La differenza tra i due valori (errore "e") viene utilizzata per pilotare il controllore che interviene in modalità "tutto o niente" comandando con la sua uscita l'attivazione (ON) o la disattivazione (OFF) del bruciatore della caldaia.



Altro esempio è la cisterna d'acqua, dove lo scopo è mantenere il livello del liquido entro margini prefissati.

Quando il livello risulta inferiore alla soglia minima, il sistema interviene comandando l'apertura di un'elettrovalvola (rubinetto controllabile elettricamente) che rimane aperta (stato "ON") fin quando il livello sale e supera la soglia massima.

L'attuatore viene accesso e spento (mediante relè o transistor) sulla base del valore misurato della grandezza controllata con una oscillazione definita "isteresi" (es. +-1°C).



SISTEMA DI CONTROLLO PID (PROPORZIONALE – INTEGRALE – DERIVATIVO)

E' un sistema in retroazione negativa ampiamente impiegato nei sistemi di controllo automatico.
È molto comune nell'industria, in particolare nella versione PI (senza azione derivativa).
Grazie a un input che determina il valore attuale, è in grado di reagire a un eventuale errore positivo o negativo tendendo verso il valore 0.

Il controllore acquisisce in ingresso un valore da un processo e lo confronta con un valore di riferimento.
La differenza, il cosiddetto segnale di errore, viene quindi usata per determinare il valore della variabile di uscita del controllore, che è la variabile manipolabile del processo.

Il PID regola l'uscita in base a:

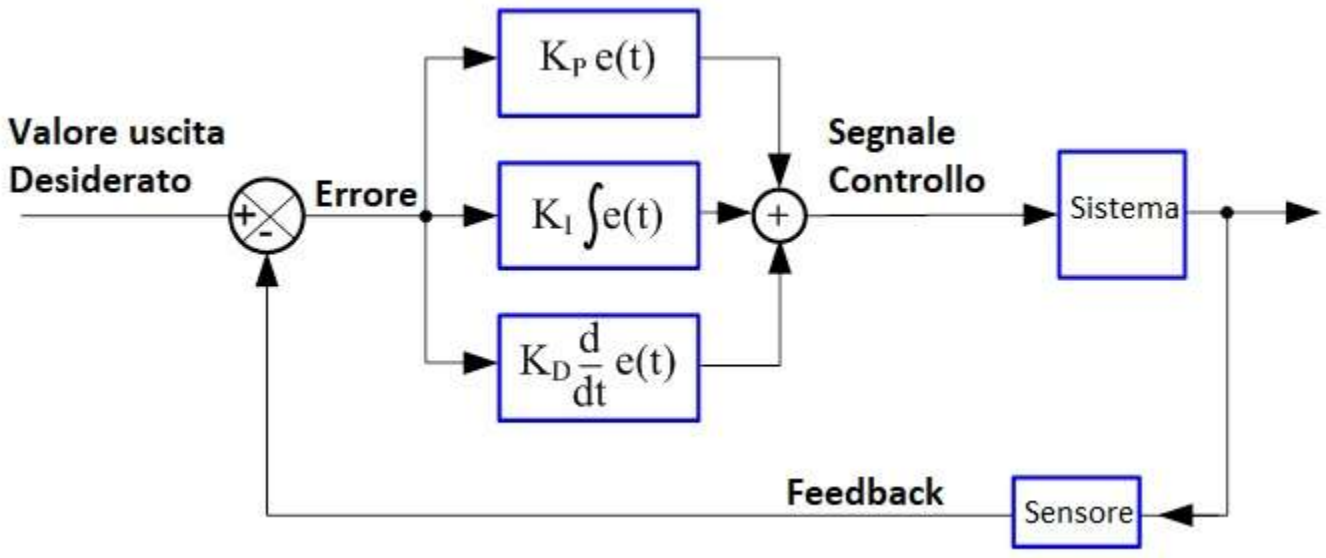
- il valore del segnale di errore (azione proporzionale → coefficiente Kp);
- i valori passati del segnale di errore (azione integrale → coefficiente Ki);
- quanto velocemente il segnale di errore varia nel tempo (azione derivativa → coefficiente Kd).

I controllori PID sono relativamente semplici da comprendere, installare e tarare, al confronto con più complessi algoritmi di controllo basati sulla teoria del controllo ottimo e del controllo robusto.

La taratura dei parametri avviene di solito attraverso semplici regole empiriche, come i metodi di Ziegler-Nichols, che risultano in controllori stabilizzanti di buone prestazioni per la maggior parte dei processi.
Molto spesso l'azione derivativa viene rimossa, risultando nel comunissimo controllore PI.

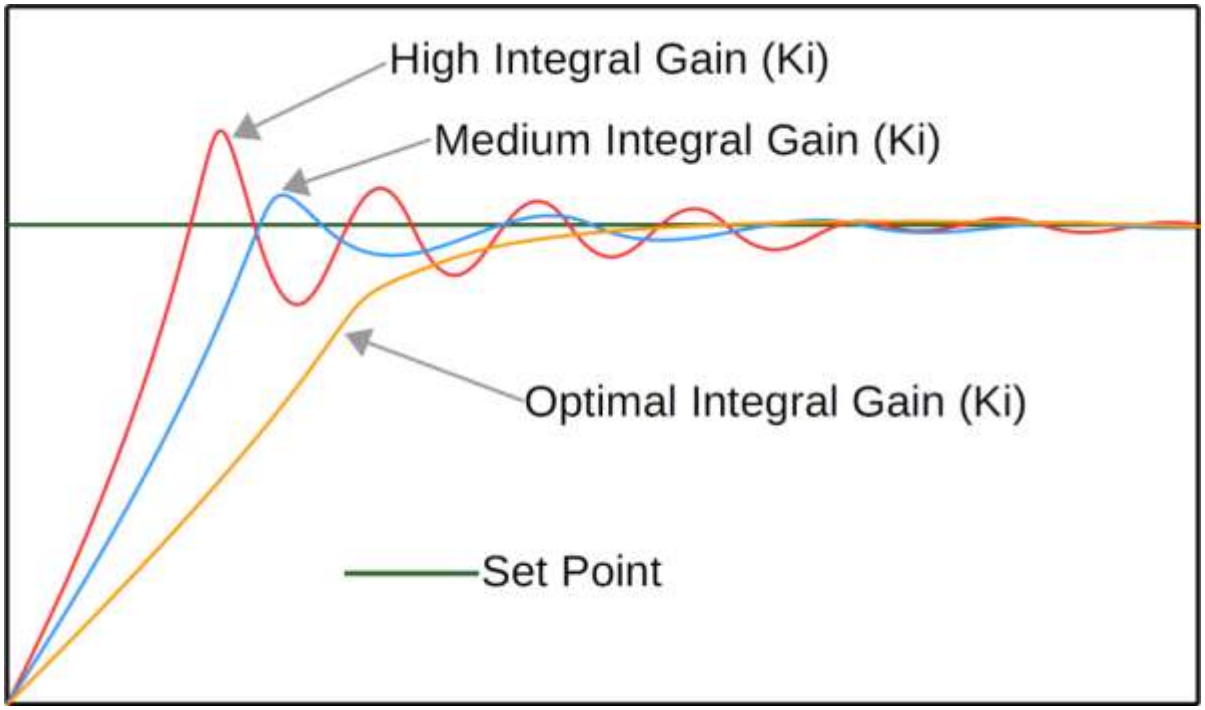
I controllori PID sono spesso sufficienti a controllare processi industriali anche complessi, ma la loro semplicità risulta in una serie di limiti che è bene tener presente:

- Non sono in grado di adattarsi a cambiamenti nei parametri del processo;
- Non sono stabili, a causa della presenza dell'azione integrale;
- Alcune regole di taratura, come quelle di Ziegler-Nichols, reagiscono male in alcune condizioni;
- Sono intrinsecamente monovariabili, non possono quindi essere usati in sistemi inerentemente multi variabili

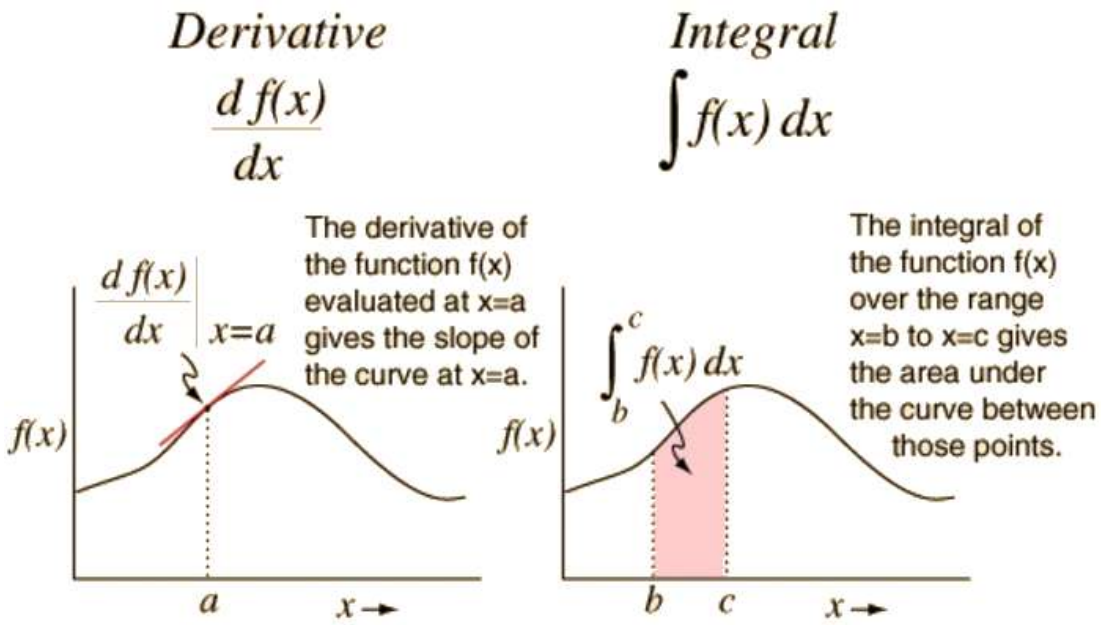


Segnale controllo = $Kp * e(t) + Ki * \int e(t)\, dt + Kd * de(t)/dt$ (in genere una tensione …)

La scelta dei parametri Kp, Ki e Kd è fondamentale per ottenere il risultato desiderato.
Valore non corretti possono rendere il sistema instabile.



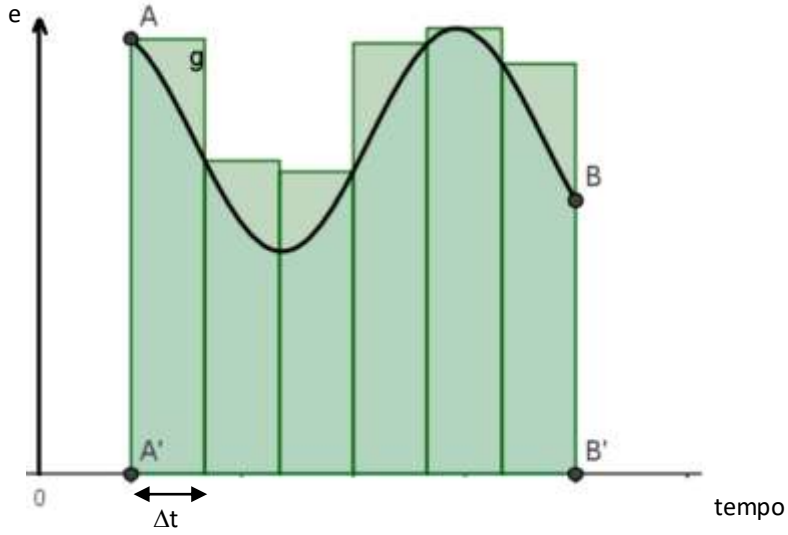
IMPLEMENTAZIONE NUMERICA PID



Integrale dell'errore → somma aree → $\sum \Delta e * \Delta t$

Derivata dell'errore → variazione dell'errore nell'intervallo di tempo → $\Delta e / \Delta t$

INTEGRAZIONE NUMERICA DELL'ERRORE



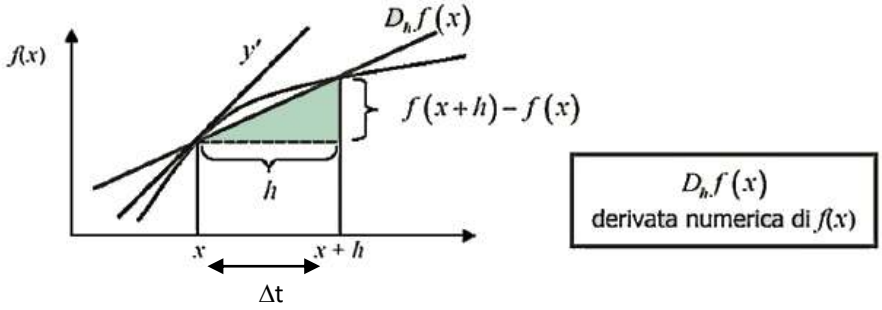
Sommando le aree ottenuto col prodotto e*Δt andiamo ad approssimare l'integrale della curva errore.

∫ e(t) dt = ∑ e*Δt più il Δt è piccolo e più è preciso il calcolo

DERIVAZIONE NUMERICA DELL'ERRORE

La derivata puntale di una funzione si può approssimare con il suo rapporto incrementale fissando un opportuno Δt

$$y' = \lim_{h\to 0} \frac{f(x+h)-f(x)}{h} \Rightarrow y' \approx \frac{f(x+h)-f(x)}{h} = D_h f(x)$$

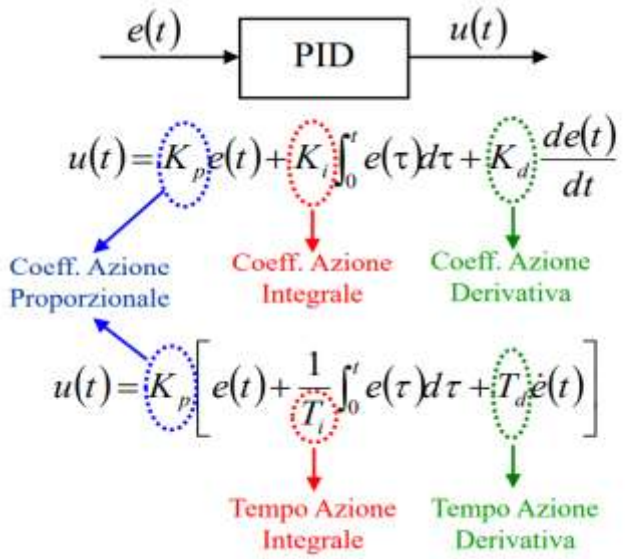


e'(t) = Δe/Δt più il Δt è piccolo e più è preciso il calcolo

REGOLE DI ZIEGLER-NICHOLS

Il metodo di Ziegler-Nichols, risalente al 1942, è tra i più usati ed è apprezzato per la sua semplicità, per il fatto di non richiedere un modello matematico del processo e per le prestazioni che riesce a produrre. Serve a trovare il cosiddetto "guadagno critico Ku", dal quale si deriveranno gli altri parametri del PID:

- Il processo viene fatto controllare da un controllore esclusivamente proporzionale (Ki e Kd vengono impostati a zero);
- Il guadagno Kp del controllore proporzionale viene gradualmente aumentato;
- Il guadagno critico Ku è il valore del guadagno per cui la variabile controllata presenta oscillazioni sostenute, cioè che non spariscono dopo un transitorio (questa è una misura dell'effetto dei ritardi e della dinamica del processo);
- Si registra il periodo critico Pu delle oscillazioni sostenute e con la tabella allegata si determinano le costanti per il controllore P, PI o PID;

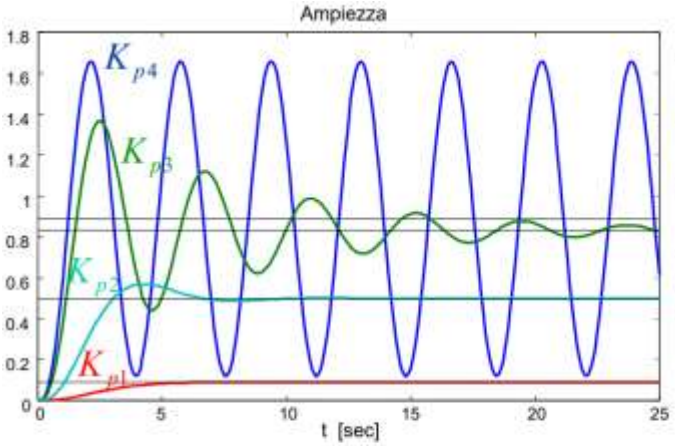


$$T_i=\frac{K_p}{K_i}$$

$$T_d=\frac{K_d}{K_p}$$

Tipo	K_p	Ti	Td
P	$0{,}50K_u$	-	-
PI	$0{,}45K_u$	$P_u/1{,}2$	-
PID	$0{,}60K_u$	$P_u/2$	$P_u/8$

ESEMPIO CALCOLO COSTANTI PID



$$K_{p1}=0.1$$

$$K_{p2}=1$$

$$K_{p3}=5$$

$$K_{p4}=8\leftarrow\overline{K}_p$$

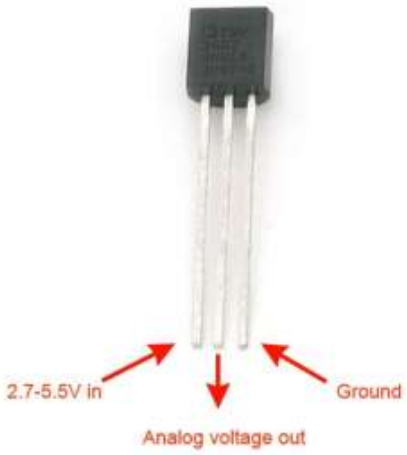
$$\overline{T}\cong 3.6$$

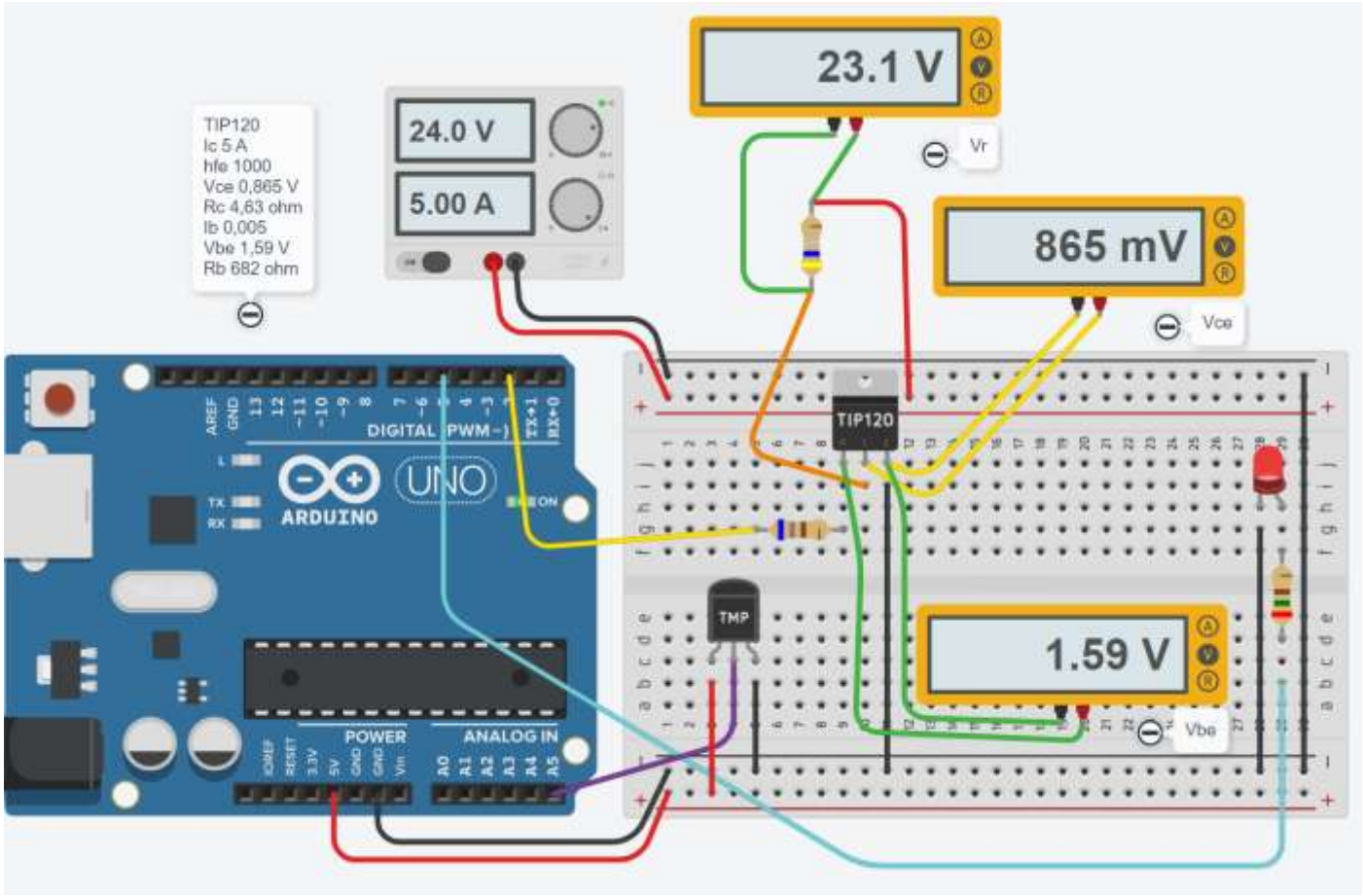
PID "ideale"

$$\begin{cases} K_p=0.6\overline{K}_p=4.8 \\ T_i=0.5\overline{T}=1.8138 \\ T_d=0.125\overline{T}=0.4534 \end{cases}$$

CONTROLLO DI TEMPERATURA ON-OFF CON SENSORE TMP36

Progettare un sistema di CONTROLLO della temperatura dell'acqua in un recipiente.
L'acqua deve essere mantenuta alla temperatura di 50°C con una tolleranza di +-1°C.
Si utilizzi come sensore di temperatura il TMP36.
Si adotti un semplice sistema di regolazione ON-OFF dell'elemento riscaldante con tempo di campionamento (rilevazione) della temperatura di 30 sec.





CODICE

```
long t0;
long tempoPrint=1000; // tempo frequenza stampa a video
int statoRiscaldamento = 0;  // 0=spento; 1=attivo; 2=mantengo
float tempSetPoint= 50.0;  // SET POINT +-1°C
float volt;
float temperatura= 0;

void setup()
{
  pinMode(2, OUTPUT);  // TIP120
  pinMode(5, OUTPUT);  // LED
  pinMode(A5, INPUT);   // TMP36
  Serial.begin(9600);
  t0= millis();
}

void loop()
{
  volt = analogRead(A5) * 5.0/1024.0;  // usare i decimali nella divisione!
  temperatura = 100 * volt – 50;

  if (temperatura <= 49.0) {
    statoRiscaldamento= 1;  // attivo
    digitalWrite(2, HIGH);
    digitalWrite(5, HIGH);
  }
  else if (celsius> 49.0 && celsius< 51.0)  {
    statoRiscaldamento= 2;  / /mantengo
  }
  else {
     statoRiscaldamento= 0;  // spengo
     digitalWrite(2, LOW);
     digitalWrite(5, LOW);
  }

  //stampa stato processo
  if ( (millis() - t0) >= tempoPrint) {
    t0= millis();
    Serial.print("T="); Serial.println(celsius);
    switch (statoRiscaldamento) {
      case 1: Serial.println("Attivo");
        break;
      case 2: Serial.println("Mantengo");
        break;
      case 0: Serial.println("Spento");
        break;
    }
  }

  delay(30000);  // 30 sec
}
```

Al posto dello “switch” si può usare

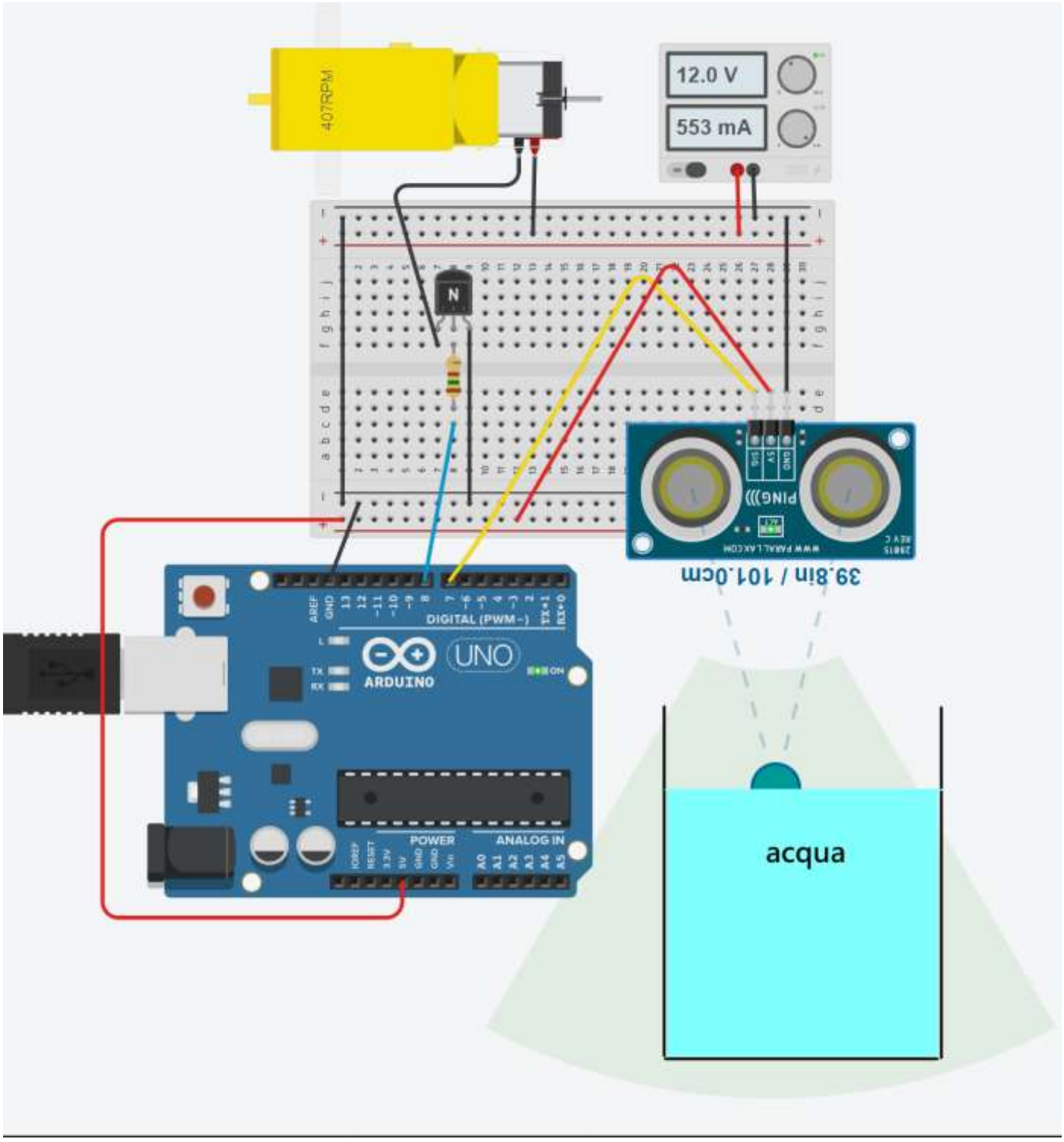
If

Else if

Else if

CONTROLLO LIVELLO ON-OFF CON SENSORE ULTRASUONI

Si vuole avviare una pompa di riempimento quando il livello dell'acqua scende sotto un valore minimo (es. 100cm).

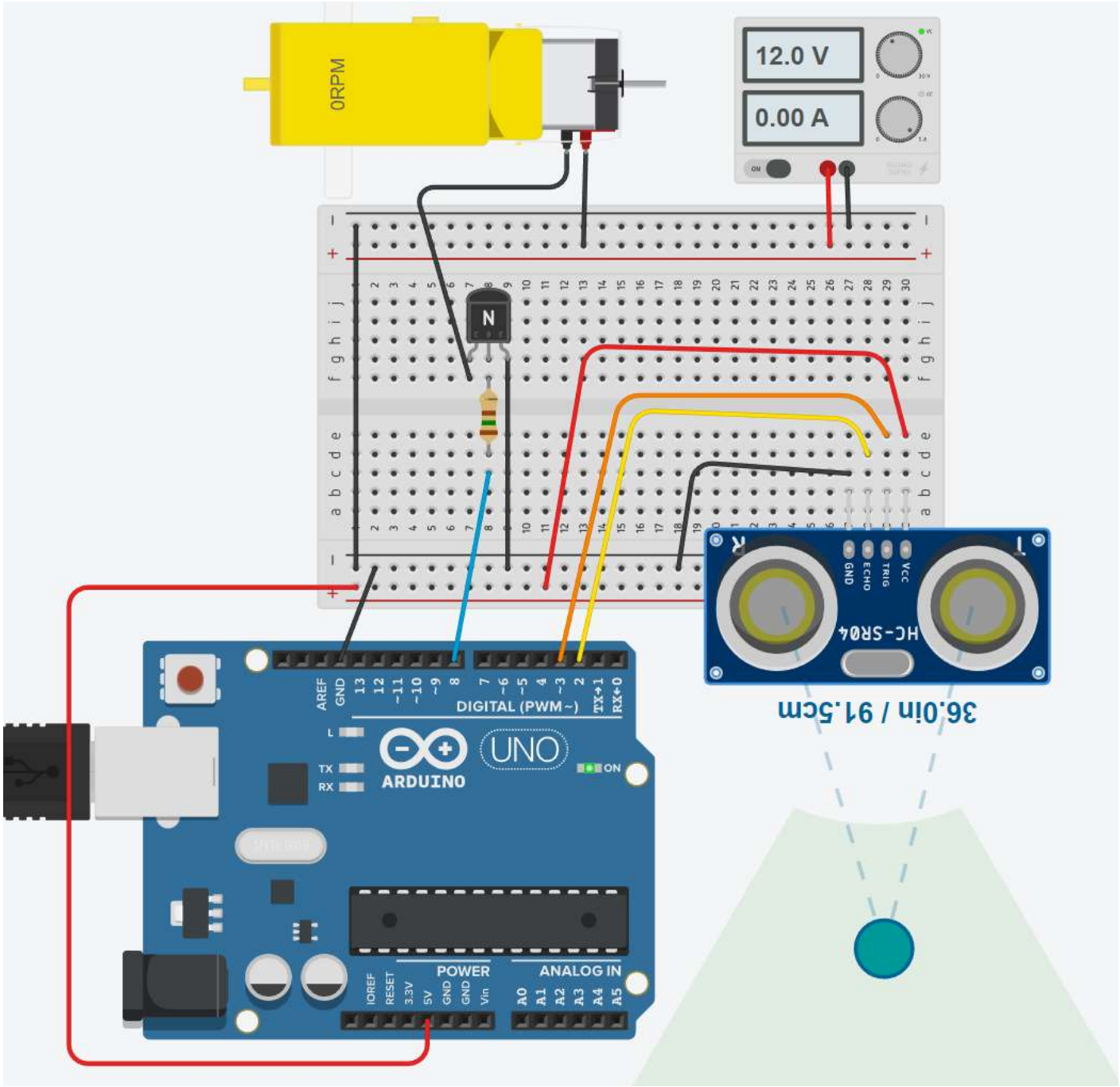


CODICE

```
int motorPin=8;
int cm = 0;
int statoM=0;   //0 spento; 1 acceso
int livello=100;
int errore=2;

long readUltrasonicDistance(int triggerPin, int echoPin)
{
  pinMode(triggerPin, OUTPUT);  // Clear the trigger
  digitalWrite(triggerPin, LOW);
  delayMicroseconds(2);
  // Sets the trigger pin to HIGH state for 10 microseconds
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);
  pinMode(echoPin, INPUT);
  // Reads the echo pin, and returns the sound wave travel time in microseconds
  return pulseIn(echoPin, HIGH);
}

void setup()
{
  pinMode(motorPin, OUTPUT);  // Clear the trigger
  Serial.begin(9600);
}

void loop()
{
  // measure the ping time in cm
  cm = 0.01723 * readUltrasonicDistance(7, 7);
  Serial.print(cm);
  Serial.println("cm");

  if (cm<(100-errore/2)) {
    digitalWrite(8,HIGH);
    Serial.println("Attivo motore");
    statoM=1;
  }
  else if (cm>=(100+errore/2)) {
    digitalWrite(8,LOW);
    Serial.println("Spengo motore");
    statoM=0;
  }
  else {
              if (statoM=0) {
                Serial.println("spento");
              }
              else {
                Serial.println("acceso");
              }
  }

  delay(100);  // Wait for 100 millisecond(s)
}
```

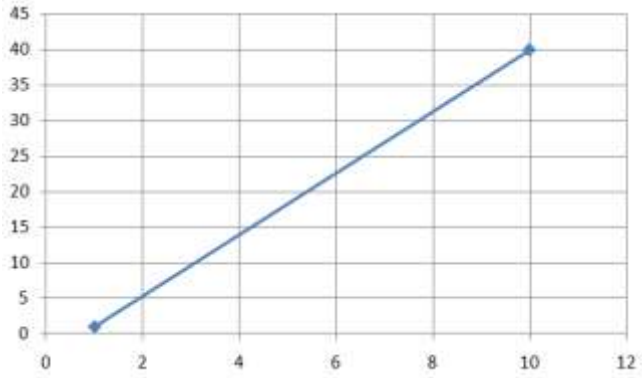
CONTROLLO LIVELLO ON-OFF CON SENSORE ULTRASUONI 2

Si vuole avviare una pompa di riempimento quando il livello dell'acqua scende sotto un valore minimo (es. 100cm).
Si utilizzi il sensore ad ultrasuoni per Arduino HC-SR04 dotato di due distinti pin per "trigger" e "echo".



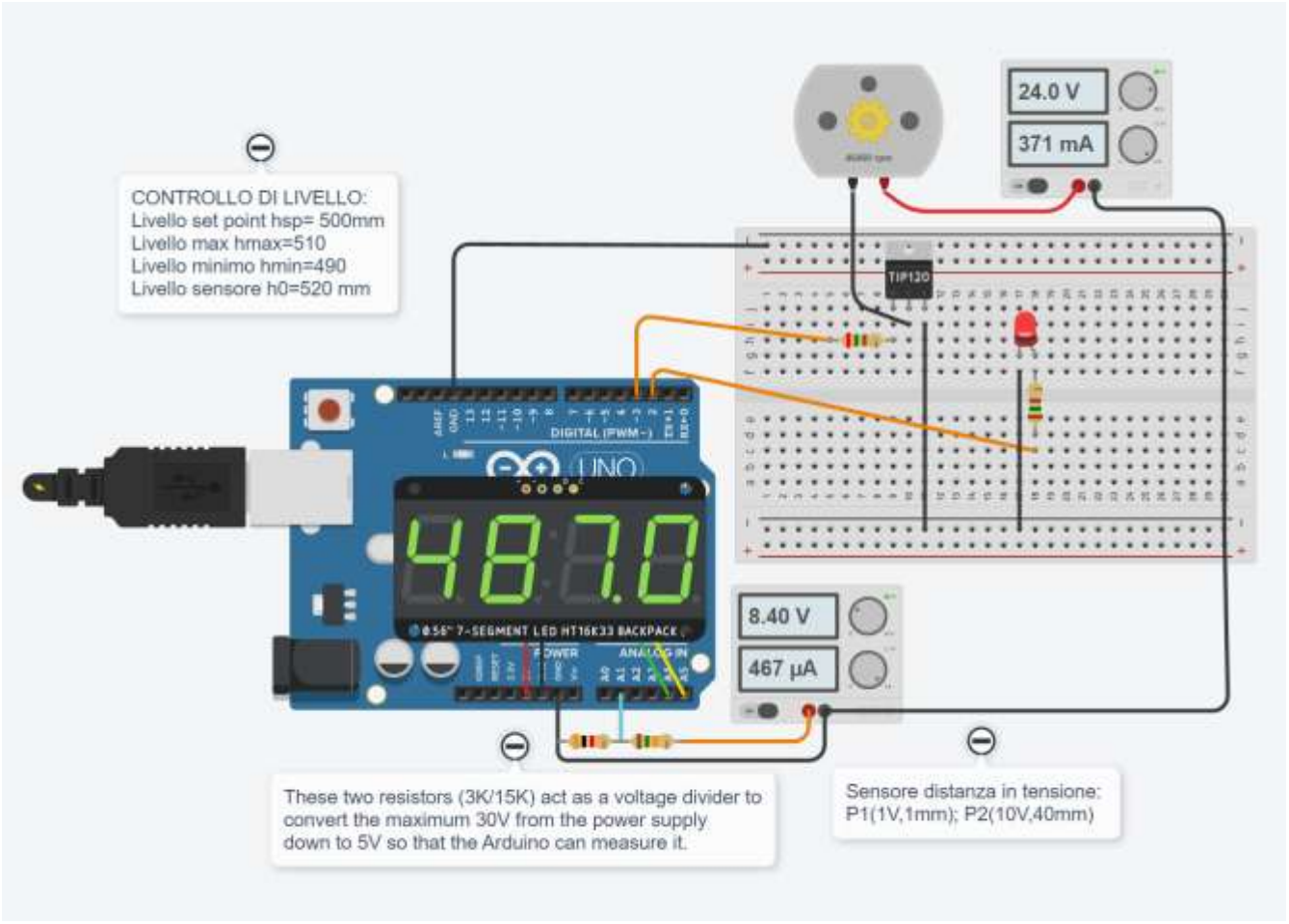
CONTROLLO DI LIVELLO CON SENSORE ANALOGICO

Si vuole mantenere il livello di acqua in un serbatoio a 500mm con una tolleranza di +/-10mm.
Si ha disposizione
-una pompa con motore CC a 24V (I=350mA)
-un transistor di potenza TIP120 (hfe=1000, Vbe=0,7v)
-un sensore di livello analogico in tensione con la seguente caratteristica lineare V (volt) - distanza [mm]:



Curva del sensore:

distanza = 39*(volt-1) /9 + 1 [mm]



Utilizzare un LCD a 7 segmenti per mostrare la temperatura attuale e usare la seriale per indicare lo stato attuale della pompa (accesa, spenta, mantengo accesa, mantengo spenta).
Il sensore può essere simulato con un generatore di tensione continua 0-30V e un partitore di tensione 1k-5k che riduce i 30V max. a 5V max. (30/5=6 volte) in ingresso ad Arduino.

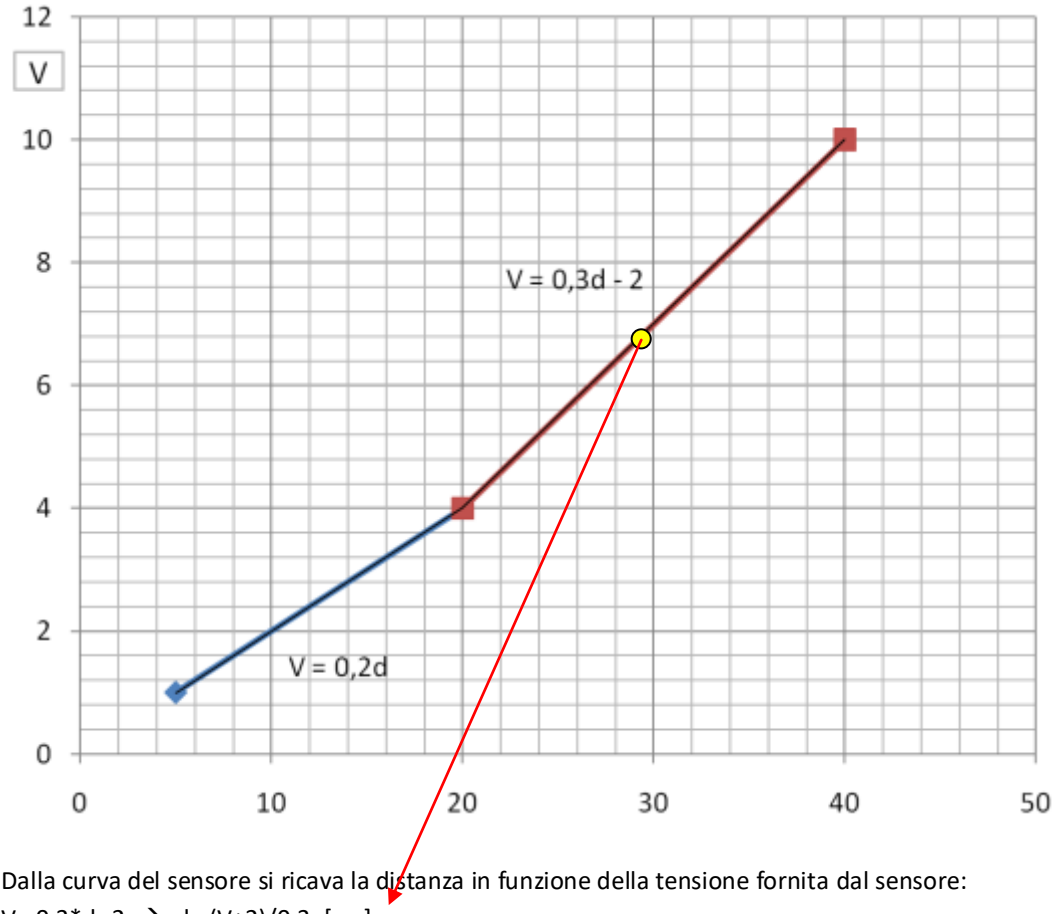
EX: risolvere lo stesso problema utilizzando un sensore ad ultrasuoni

CODICE

```
#include "Adafruit_LEDBackpack.h"
float h0=520.0;
float hsp=500.0;
float delta=20.0;
float volt;
float distanza;
float altezza;
float hmax = hsp+delta/2.0;
float hmin = hsp-delta/2.0;
int stato_pompa=0;  // 0 off; 1 on

Adafruit_7segment led_display1 = Adafruit_7segment();

void setup()
{
  led_display1.begin(112);
  pinMode(A1, INPUT);
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  Serial.begin(9600);
}

void loop()
{
  // Convert from 0-1023 range to 0-30V range
  volt= analogRead(A1) * 5.0 / 1023.0 * 6;  // moltiplico per 6 per ottenere 30V
  // Convert voltage to distance
  distanza= 39.0/9.0*(volt-1.0)+1.0;
  // Get heigh level
  altezza= h0-distanza;

  led_display1.println(altezza);
  led_display1.writeDisplay();

  Serial.println(altezza);

  if (altezza >= hmax) {
    digitalWrite(3, LOW);
    digitalWrite(2, LOW);
    stato_pompa= 0;
    Serial.println("Spento");
  }
  else if (altezza <= hmin) {
    digitalWrite(3, HIGH);
    digitalWrite(2, HIGH);
    stato_pompa= 1;
    Serial.println("Acceso");
  }
  else {
    if (stato_pompa== 1) {
      Serial.println("Mantengo Acceso");
  }
    else {
      Serial.println("Mantengo Spento");
    }
  }

  delay(2000);
}
```

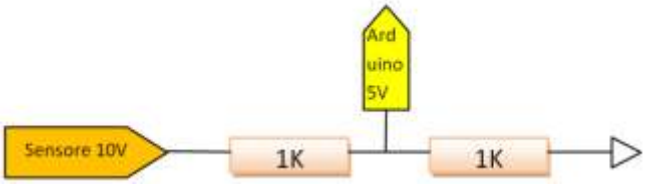
CONTROLLO LIVELLO CON SENSORE ANALOGICO NON LINEARE

Realizzare un sistema di controllo ON-OFF di livello che mantenga il livello dell'acqua in un serbatoio a 32cm.
Il sensore è montato ad una quota di 60cm dal fondo. La tolleranza del sistema è di +-2cm.
Il sensore di livello analogico assegnato presenta la seguente curva Distanza [cm] –Tensione [volt]



Dalla curva del sensore si ricava la distanza in funzione della tensione fornita dal sensore:
V= 0,3*d -2 → d= (V+2)/0,3 [cm]

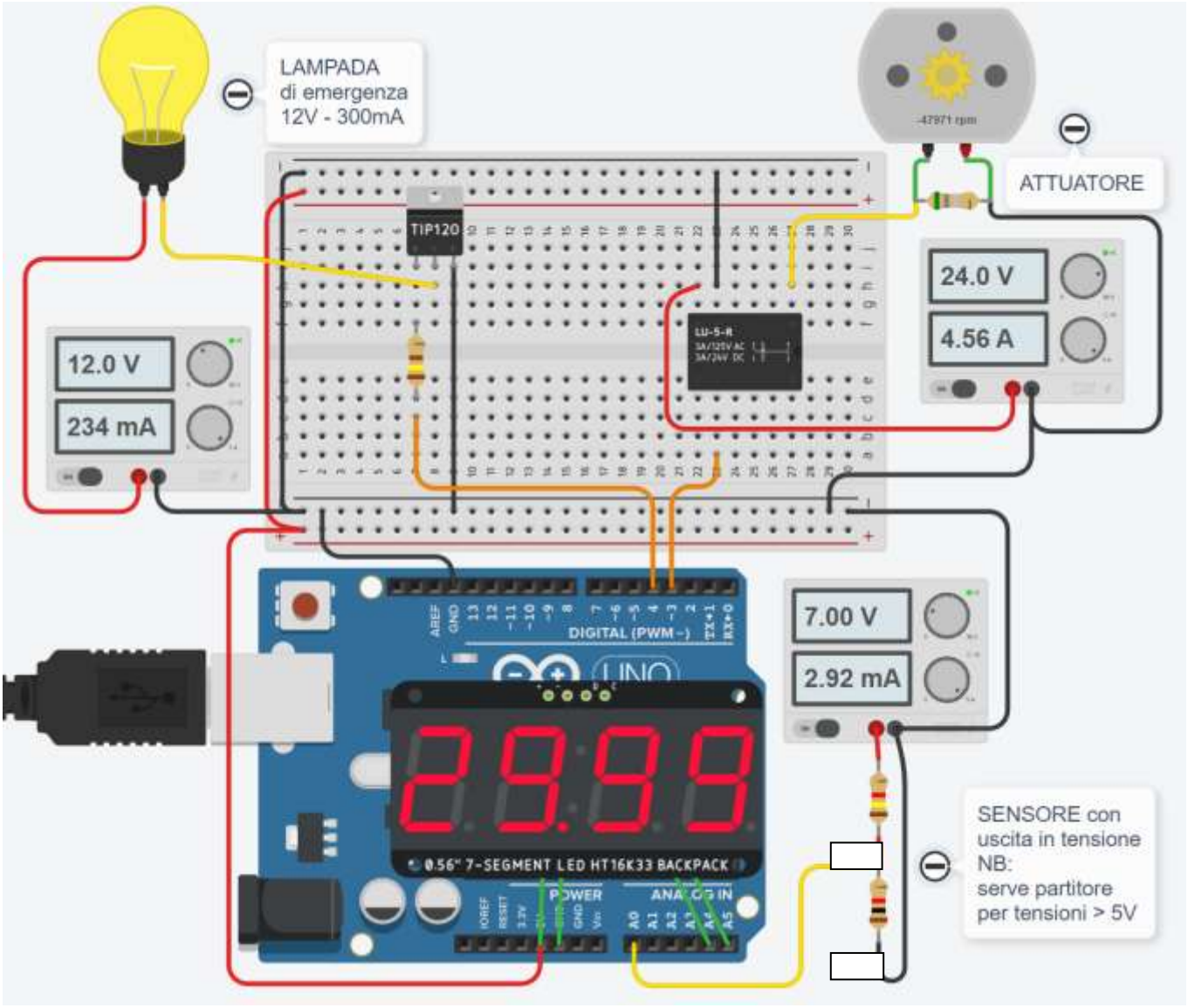
La tensione max. in uscita dal sensore è 10V.
Va ridotta a 5V in ingresso ad Arduino tramite un partitore
(ad es. 1K+1K → riduco 10V a 5V → 2 volte).



CODICE ARDUINO
// moltiplico x 2 per ottenere i V effettivi del sensore da usare nella formula della d(cm)
Volt = analogRead(pinS)*5/1023 * 2; → Volt = analogRead(pinS)*10/1023;

DIMENSIONAMENTO DEL PARTITORE DI TENSIONE

Se il sensore ha una tensione di uscita max. di 12V allora devo applicare la legge di Ohm per trovare le R:
→ fisso ad es. la R1=1000 ohm (alta per far circolare correnti basse mA)
→ I=5/1000 A
→ R2= (12-5) / I = 1400 ohm



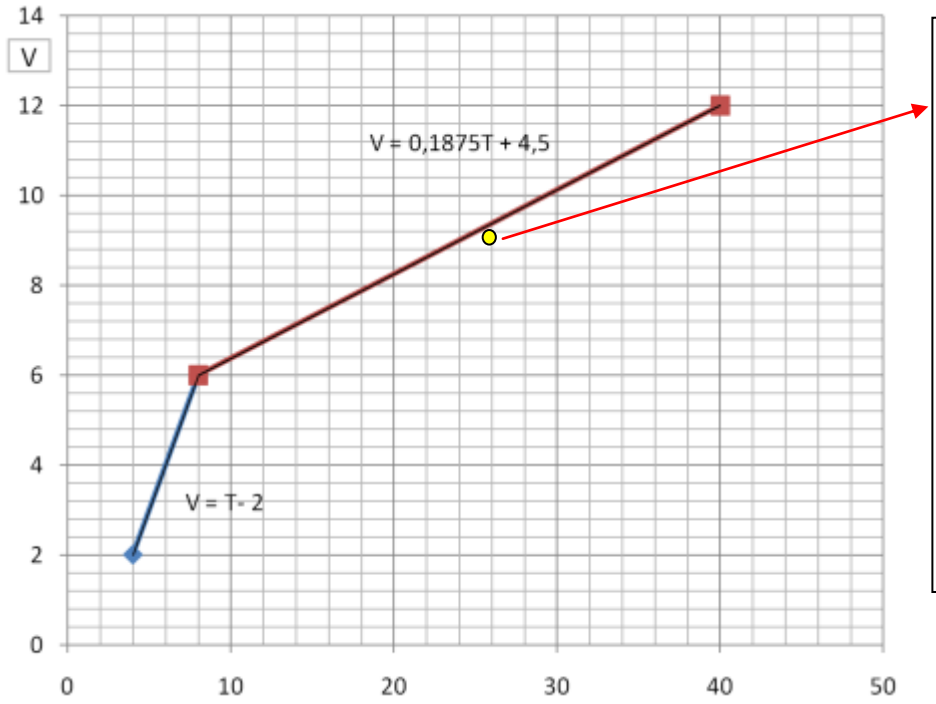
CODICE

```
#include "Adafruit_LEDBackpack.h"

Adafruit_7segment led_display1 = Adafruit_7segment();

int pinPompa = 3;
int pinLampada = 4;
int pinSensore = A0;

float h0=60.0;   // quota sensore dal fondo
float hsp=32.0;  // livello SET-POINT
float delta=2.0; // errore tollerato

float volt;
float distanza;
float altezza;
float hmax = hsp+delta/2.0;
float hmin = hsp-delta/2.0;
int stato_pompa=0;  // 0 off; 1 on

void setup()
{
  pinMode(pinSensore, INPUT);
  pinMode(pinPompa, OUTPUT);  // POMPA
```

```
  pinMode(pinLampada, OUTPUT);  // LAMPADA
  led_display1.begin(112);
  Serial.begin(9600);
}


void loop()
{
  // Convert from 0-1023 range to 0-12V range (partitore riduce a 5!)
  volt= analogRead(pinSensore) * 12.0 / 1023.0;
  // Convent voltage to distance  --> d= (V+2)/0,3  [cm]
  distanza= (volt+2)/0.3;
  Serial.print("dist cm "); Serial.println(distanza);
  // Get heigh
  altezza= h0-distanza;
  Serial.print("h cm "); Serial.println(altezza);
  led_display1.println(altezza);
  led_display1.writeDisplay();

  if (altezza >= hmax) {
    digitalWrite(pinPompa, LOW);
    digitalWrite(pinLampada, LOW);
    stato_pompa= 0;
    Serial.println("Spento");
  }
  else if (altezza <= hmin) {
    digitalWrite(pinPompa, HIGH);
    digitalWrite(pinLampada, HIGH);
    stato_pompa= 1;
    Serial.println("Acceso");
  }
  else {
    if (stato_pompa== 1) {Serial.println("Mantengo Acceso");}
    else { Serial.println("Mantengo Spento");  }
  }

  delay(2000);
}
```

CONTROLLO TEMPERATURA CON SENSORE ANALOGICO NON LINEARE

Realizzare un sistema di controllo ON-OFF di temperatura che mantenga la temperatura in un sistema chiuso a 25°C.
La tolleranza del sistema è di +-2°C. Il tempo di campionamento è di 1 secondo.
Il sensore di temperatura analogico assegnato presenta la seguente curva Temperatura [°C] – Tensione [Volt]

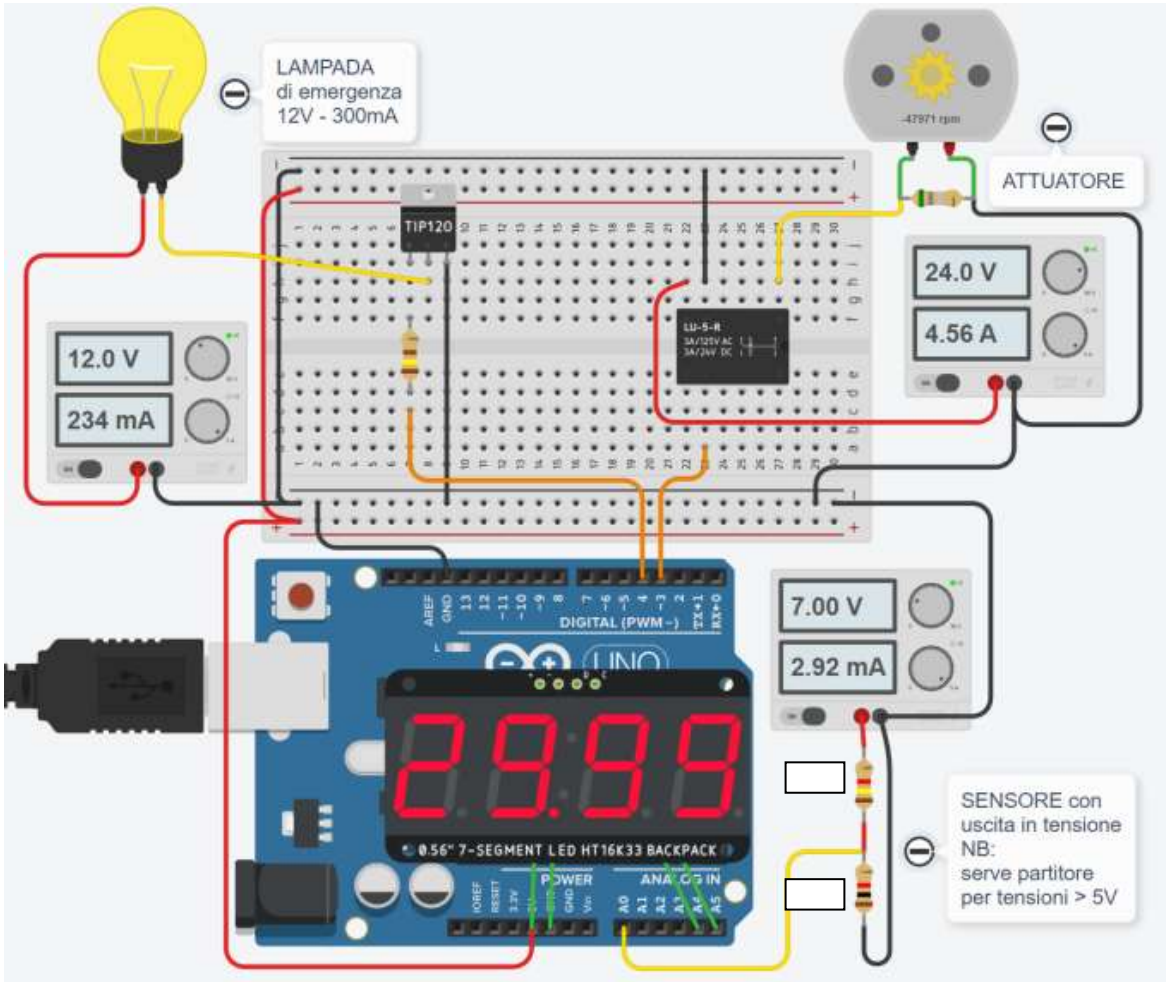


Dalla curva
V= 0,1875*T +4,5 → T= (V-4,5)/0,1875 [°C]

La tensione max. in uscita dal sensore è 12V. Va portata sotto i 5V in ingresso ad Arduino tramite un partitore di tensione.

Fisso ad es. la R1=1000 ohm
→ I=5/1000 A
→ R2= (12-5) / I = 1400 ohm

CODICE ARDUINO
// moltiplico x 12/5 per ottenere i V effettivi del sensore da usare nella formula della T(°C)
Volt = analogRead(pinS)*5/1023 * *12/5;*

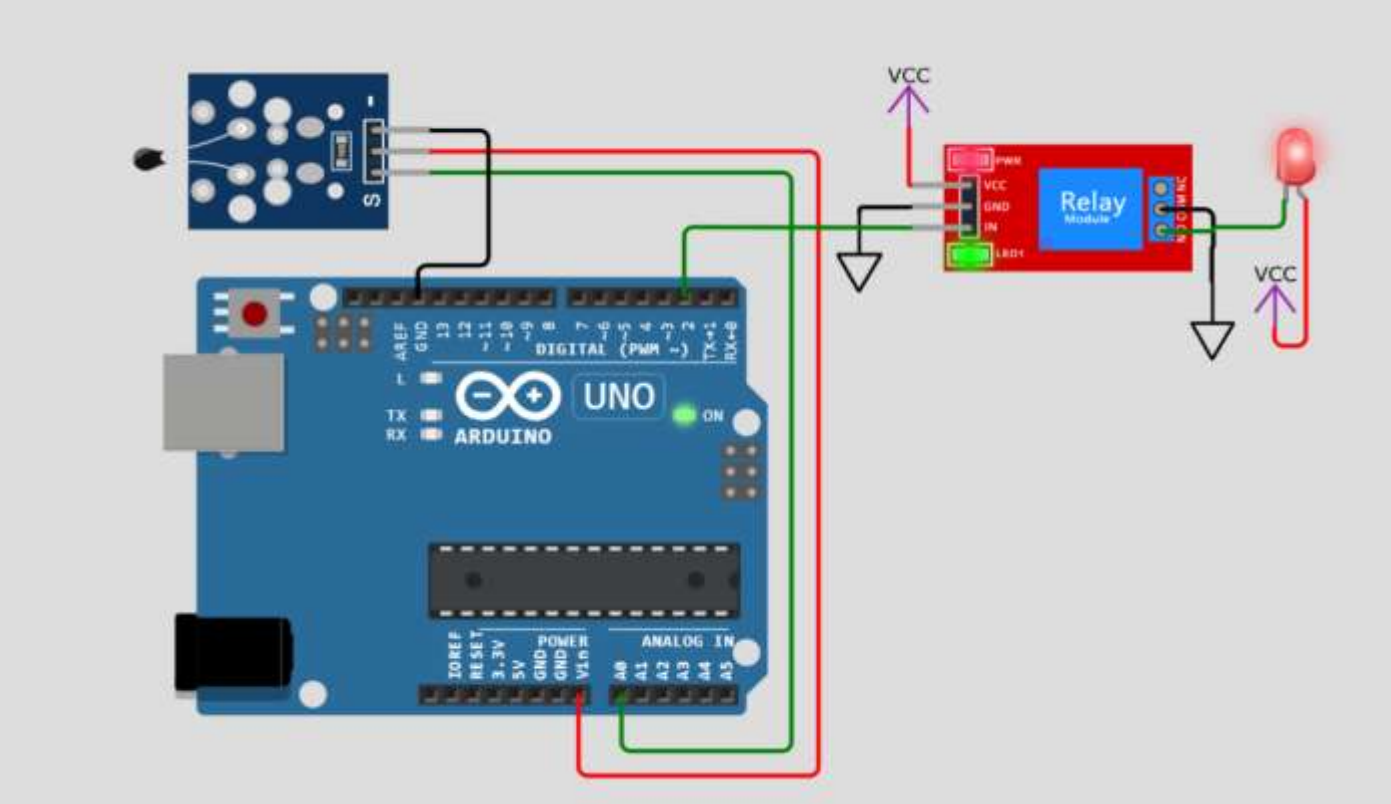


CONTROLLO DI TEMPERATURA CON TERMISTORE NTC E RELE'

Si vuole mantenere a 30°C la temperatura in un recipiente isolato con tolleranza +-1°C.

Si utilizza come elemento riscaldante una lampadina alogena da 30w attivabile tramite un relè comandato direttamente da Arduino.

Thermistor parameters: RT0: 10KΩ B: 3977 K +- 0.75% T0: 25 C +- 5%



simulabile su "wokwi.com"

CODICE

```
//thermistor parameters:  RT0: 10 000 Ω   B: 3977 K +- 0.75%   T0:  25 C  +- 5%

//These values are in the datasheet
#define RT0 10000   // Ω
#define B 3977      // K
//--------------------------------------
#define VCC 5    //Supply voltage
#define R 10000  //R=10KΩ
//--------------------------------------
int Tsp=30;   // set point
int deltaT=2; // errore tollerato

int pinRele= 2;
int statoLamp= 0;
//Variables
float RT, VR, ln, TX, T0, VRT;

void setup() {
  Serial.begin(9600);
  pinMode(pinRele, OUTPUT);

  T0 = 25 + 273.15;  // convento un Kelvin
}

void loop() {
  VRT = analogRead(A0);
  VRT = (5.00 / 1023.00) * VRT;      //Conversion to voltage
  VR = VCC - VRT;
  RT = VRT / (VR / R);               //Resistance of RT
  ln = log(RT / RT0);
  TX =1/(ln/ B+1/T0);                //Temperature from thermistor in K
  TX = TX - 273.15;                  //Conversion to °C

  Serial.print("Temperatura: ");
  Serial.print(TX);
  Serial.println(" °C");

  if (TX<(Tsp-deltaT/2)) {
    digitalWrite(pinRele, HIGH);
    Serial.println("ACCESO");
    statoLamp = 1;
  }
  else if ((TX>(Tsp+deltaT/2)))
  {
    digitalWrite(pinRele, LOW);
    Serial.println("SPENTO");
    statoLamp = 0;
  }
  else {
    if (statoLamp==0) {Serial.println("MANTENGO SPENTO");}
    else {Serial.println("MANTENGO ACCESO");}
  }

  delay(1000);
}
```

CONTROLLO DI TEMPERATURA ON-OFF CON TERMISTORE NTC E NMOS

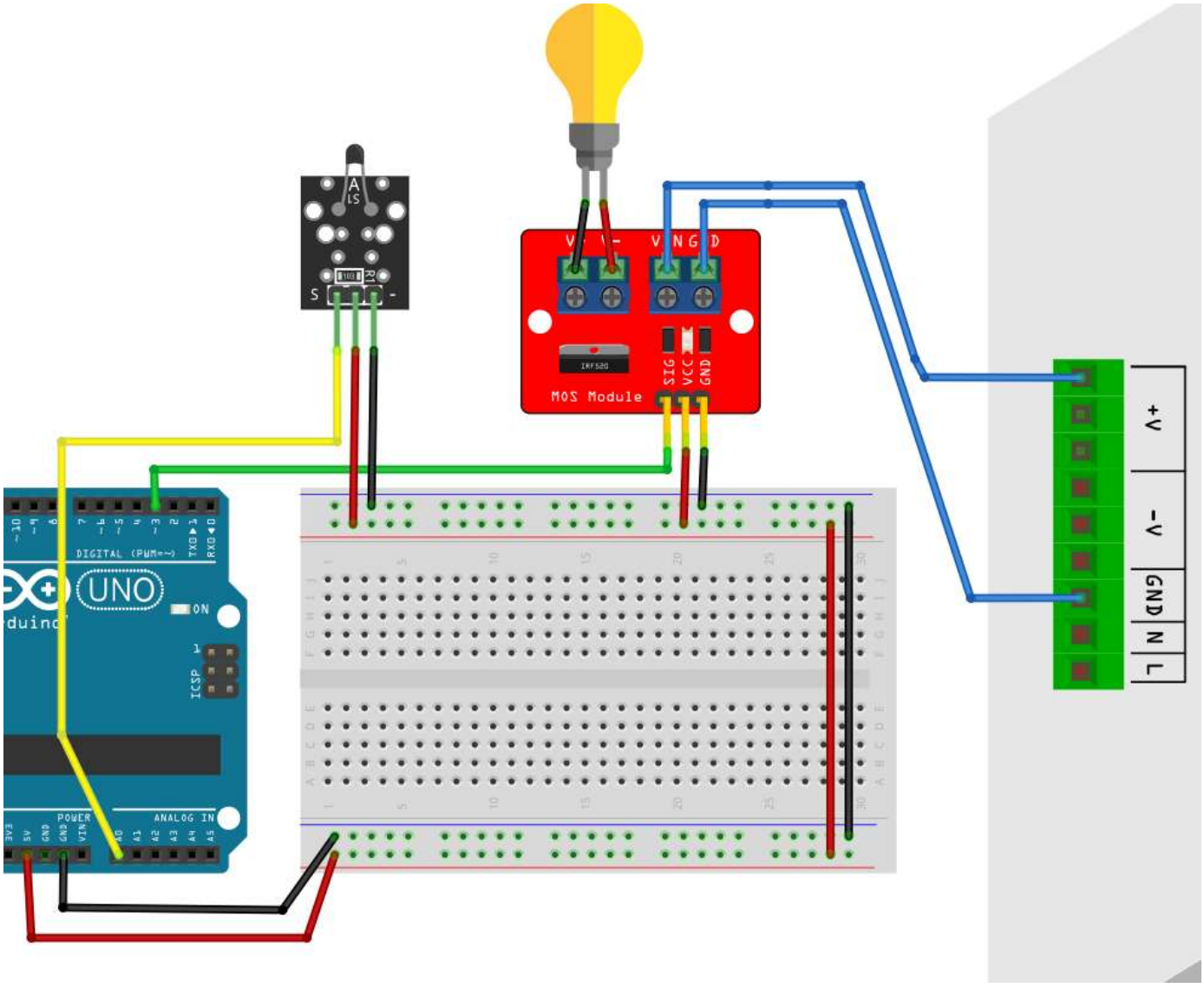
Si vuole mantenere costante la temperatura all'interno di un recipiente coibentato.

Come sensore si utilizza un termistore NTC (RT0: 10KΩ B: 3977 K +- 0.75% T0: 25 C +- 5%).

Come attuatore una lampadina alogena che alimentata a 10V fornisce circa 15w di potenza elettrica.

La lampadina viene attivata da un modulo MOSFET IRF520 per Arduino.

Il tipo di controllo è ON-OFF con tolleranza +-1°C.



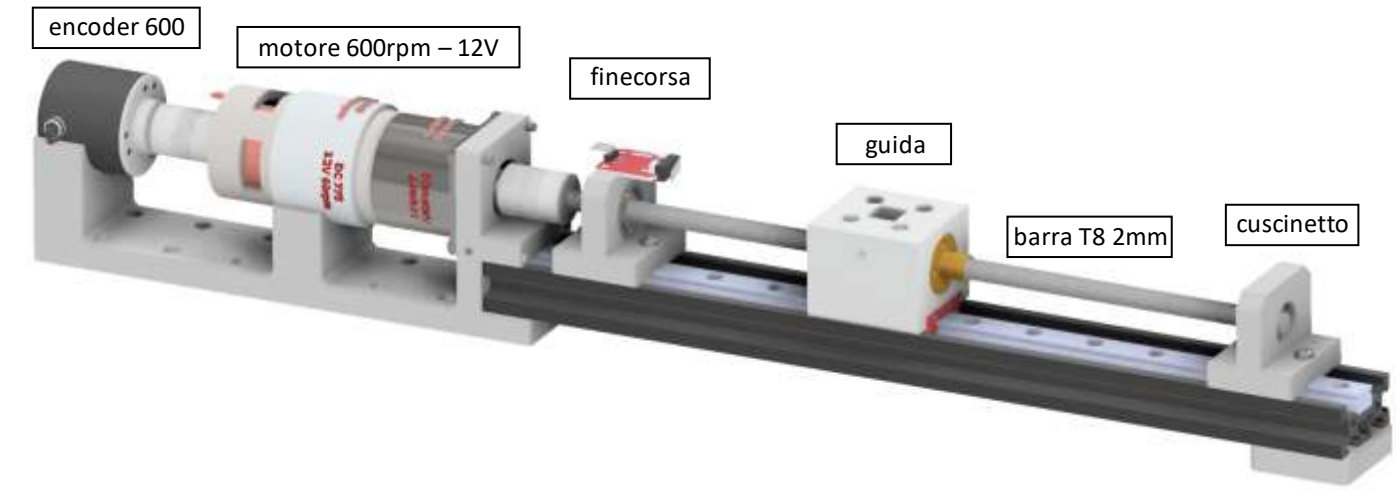
"non simulabile"

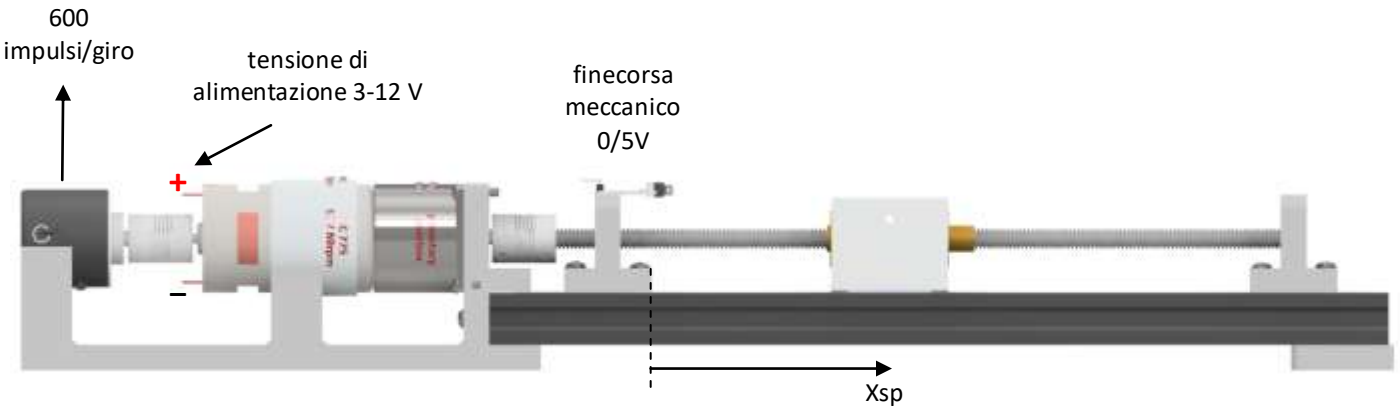
CODICE

```
//Thermistor parameters:  RT0: 10KΩ   B: 3977 K +- 0.75%   T0:  25 C +- 5%
//From datasheet
#define RT0 10000   // Ω
#define B 3977      // K
#define VCC 5    //Supply voltage
#define R 10000  //R=10KΩ
//--------------------------------------

int pin_rele=2;
int pin_T=A0;
long t;
float delta=0.5;

//Variables
float RT, VR, ln, TX, T0, VRT;

void setup() {
  pinMode(pin_rele, OUTPUT);
  pinMode(pin_T, INPUT);

  Serial.begin(9600);
  T0 = 25 + 273.15;
  t= millis();
}

void loop() {
  VRT = analogRead(pin_T);  // 0-1023  tensione sul termistore
  VRT = (5.00 / 1023.00) * VRT;      // converto in V
  VR = VCC - VRT;  // tensione sulla resistenza R da 10K
  RT = VRT / (VR / R);    // Resistenza di RT  (V/I)
  ln = log(RT / RT0);
  TX =1/ (ln / B + 1 / T0);  //Temperature from thermistor in K
  TX = TX - 273.15;                 //Conversion to °C

  if (TX>=(40+ delta/2)) {
    digitalWrite(pin_rele, LOW);
  }
  else if (TX<=(40-delta/2)) {
    digitalWrite(pin_rele, HIGH);
  }


  if (  (millis()-t)>1000 ) {
  t= millis();
  Serial.print("T:");
  Serial.print(TX);
  Serial.print(",");
  Serial.print("err:");
  Serial.println(err);
  }

  delay(100);
}
```

CONTROLLO IN POSIZIONE DI UNA GUIDA LINEARE CON MOTORE C.C. E ENCODER OTTICO INCREM.

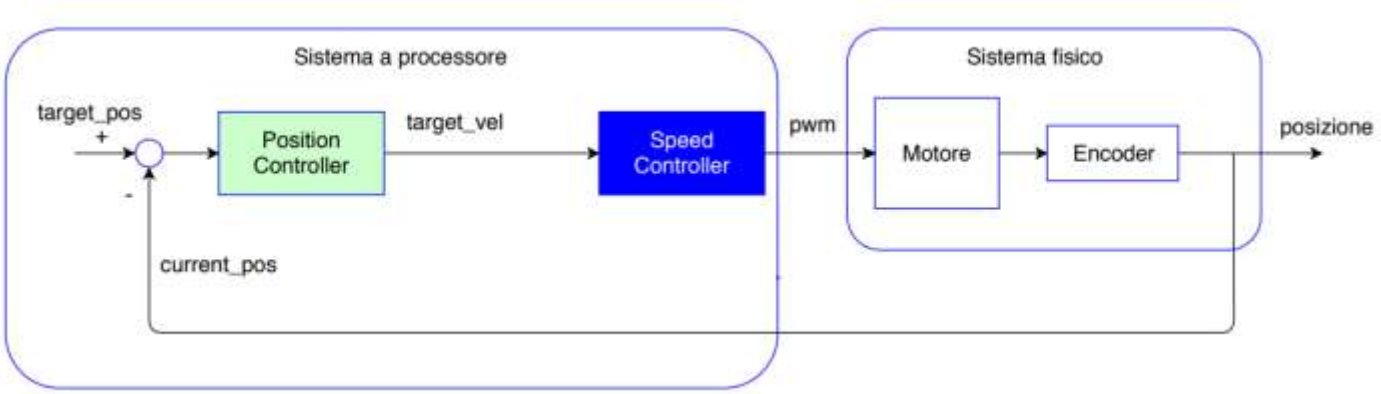
L' obiettivo è portare la guida nella posizione di SET-POINT indicata con un basso margine di errore.
Se "siamo lontani" dalla posizione Xsp di SET-PONT imponiamo una velocità di rotazione del motore alta che diminuisce man mano che ci si avvicina al SET-POINT.





Il controllore di posizione tramite un encoder incrementale fornisce il riferimento di posizione a un controllore di velocità (tipicamente un transistor di potenza con motori C.C.) che fornisce a sua volta la tensione di alimentazione del motore C.C. e fissa quindi la velocità di spostamento della guida.

SCHEMA A BLOCCHI



LOGICA DEL SISTEMA DI CONTROLLO

Se “siamo lontani” dalla posizione Xsp di SET-PONT imponiamo una velocità di rotazione del motore alta che diminuisce man mano che ci si avvicina al SET-POINT.

Usiamo quindi un controllore di tipo PROPORZIONALE per determinare la velocità del motore:

➔ vm = Kp* errore = Kp *(Xsp – Xc) Kp= coeff. Proporzionale, Xc=posizione corrente

In un motore C.C. la velocità proporzionale alla tensione di alimentazione (controllata in PWM) quindi:

➔ Vm= Kp* errore = Kp *(Xsp – Xc) Kp= coeff. Proporzionale , Xc=posizione corrente

Oltre e sotto una certa velocità il motore in C.C. non può andare e pertanto è necessario prevedere una condizione di saturazione (limite sia sulla velocità massima che minima).

Per il motore in C.C. a disposizione abbiamo i seguenti limiti operativi:

- n° max = 600 rpm con Vm=12 V → 10 giri/s → a_max = 20 mm/s (100%)
- n° min = 50 rpm con Vm=1 V → 0,834 giri/s → a_min = 1,667 mm/s (10%)

Curva del motore lineare: n° = (10/12)*Vm [giri/s]

L’encoder ottico di tipo incrementale fornisce 600 impulsi al giro.

Noto il numero di impulsi nell’intervallo di tempo di campionamento Δt lo spostamento della guida vale quindi:

s= (n_impulsi/ 600) * 2 [mm]

La velocità di spostamento della guida vale:

v_a = s / Δt [mm/s]

SIMULAZIONE CON EXCEL DEL SISTEMA DI CONTROLLO PROPORZIONALE

E' necessario fissare un valore di Kp che permetta il posizionamento della guida con un margine di errore adeguato alle richieste (es. 0.1mm) e una decelerazione accettabile.

CONTROLLO POSIZIONE GUIDA PROPORZIONALE

Xsp= 30 Kp= 1

Vmax 12 dt= 0,1

t [s]	Vm [V]	n° [giri/s]	s [mm]	vel.	accel.	err. [mm]	Vm_teor.	Vm_eff [V]
0,00	12,00	10,00	0,000	0,0	0,0	30,000	30,000	12,00
0,10	12,00	10,00	2,000	20,0	200,0	28,000	28,000	12,00
0,20	12,00	10,00	4,000	20,0	0,0	26,000	26,000	12,00
0,30	12,00	10,00	6,000	20,0	0,0	24,000	24,000	12,00
0,40	12,00	10,00	8,000	20,0	0,0	22,000	22,000	12,00
0,50	12,00	10,00	10,000	20,0	0,0	20,000	20,000	12,00
0,60	12,00	10,00	12,000	20,0	0,0	18,000	18,000	12,00

Tensione teorica di controllo del motore

Xsp= 30 Kp= 1

Vmax 12 dt= 0,1

t [s]	Vm [V]	n° [giri/s]	s [mm]	vel.	accel.	err. [mm]	Vm_teor.	Vm_eff [V]
0,00	12,00	10,00	0,000	0,0	0,0	30,000	=$D$2*G6	

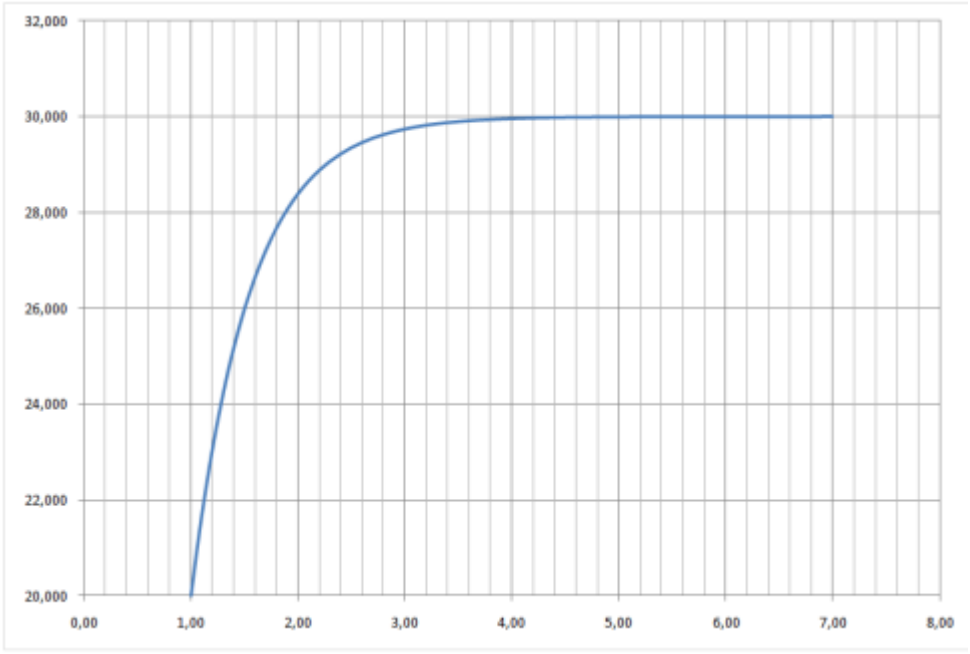
Xsp= 30 Kp= 1

Vmax 12 dt= 0,1

t [s]	Vm [V]	n° [giri/s]	s [mm]	vel.	accel.	err. [mm]	Vm_teor.	Vm_eff [V]
0,00	12,00	10,00	0,000	0,0	0,0	30,000	30,000	=SE(G6>=0;SE(H6>12;12;H6);0)
0,10	12,00	10,00	2,000	20,0	200,0	28,000	28,000	

SE(**test**; [se_vero]; [se_falso])

Il posizionamento si raggiunge dopo circa 4 secondi con una a=-33 mm/s2.



SIMULAZIONE CON EXCEL DEL SISTEMA DI CONTROLLO PID

E' necessario fissare dei valori di Kp, Ki e Kd che permettano un posizionamento della guida con un margine di errore adeguato alle richieste (es. 0.1mm) e una decelerazione accettabile.

CONTROLLO POSIZIONE GUIDA Pid

Xsp= 30 Kp= 1 Ki= 0,01 Kd= 0,01

Vmax 12 dt= 0,1

						errore				
t [s]	Vm [V]	n° [giri/s]	s [mm]	vel.	accel.	err. [mm]	integrale	derivata	Vm_teor.	Vm_eff [V]
0,00	12,00	10,00	0,000	0,0	0,0	30,000	0,000	0,000	30,000	12,00
0,10	12,00	10,00	2,000	20,0	200,0	28,000	3,000	20,000	28,230	12,00
0,20	12,00	10,00	4,000	20,0	0,0	26,000	5,800	20,000	26,258	12,00
0,30	12,00	10,00	6,000	20,0	0,0	24,000	8,400	20,000	24,284	12,00
0,40	12,00	10,00	8,000	20,0	0,0	22,000	10,800	20,000	22,308	12,00
0,50	12,00	10,00	10,000	20,0	0,0	20,000	13,000	20,000	20,330	12,00
0,60	12,00	10,00	12,000	20,0	0,0	18,000	15,000	20,000	18,350	12,00
0,70	12,00	10,00	14,000	20,0	0,0	16,000	16,800	20,000	16,368	12,00
0,80	12,00	10,00	16,000	20,0	0,0	14,000	18,400	20,000	14,384	12,00
0,90	12,00	10,00	18,000	20,0	0,0	12,000	19,800	20,000	12,398	12,00
1,00	12,00	10,00	20,000	20,0	0,0	10,000	21,000	20,000	10,410	10,41
1,10	10,41	8,68	21,735	17,4	-26,5	8,265	22,000	17,350	8,659	8,66

Integrale errore → somma aree errore nell'intervallo di tempo

Xsp= 30 Kp= 1 Ki= 0 Kd= 0

Vmax 12 dt= 0,1

						errore		
t [s]	Vm [V]	n° [giri/s]	s [mm]	vel.	accel.	err. [mm]	integrale	derivata
0,00	12,00	10,00	0,000	0,0	0,0	30,000	0,000	0,000
0,10	12,00	10,00	2,000	20,0	200,0	28,000	=G6*$D$3+H6	

Derivata errore → variazione errore nell'intervallo di tempo

Xsp= 30 Kp= 1 Ki= 0 Kd= 0

Vmax 12 dt= 0,1

						errore			
t [s]	Vm [V]	n° [giri/s]	s [mm]	vel.	accel.	err. [mm]	integrale	derivata	Vm_te
0,00	12,00	10,00	0,000	0,0	0,0	30,000	0,000	0,000	30
0,10	12,00	10,00	2,000	20,0	200,0	28,000	3,000	=(G6-G7)/$D$3	

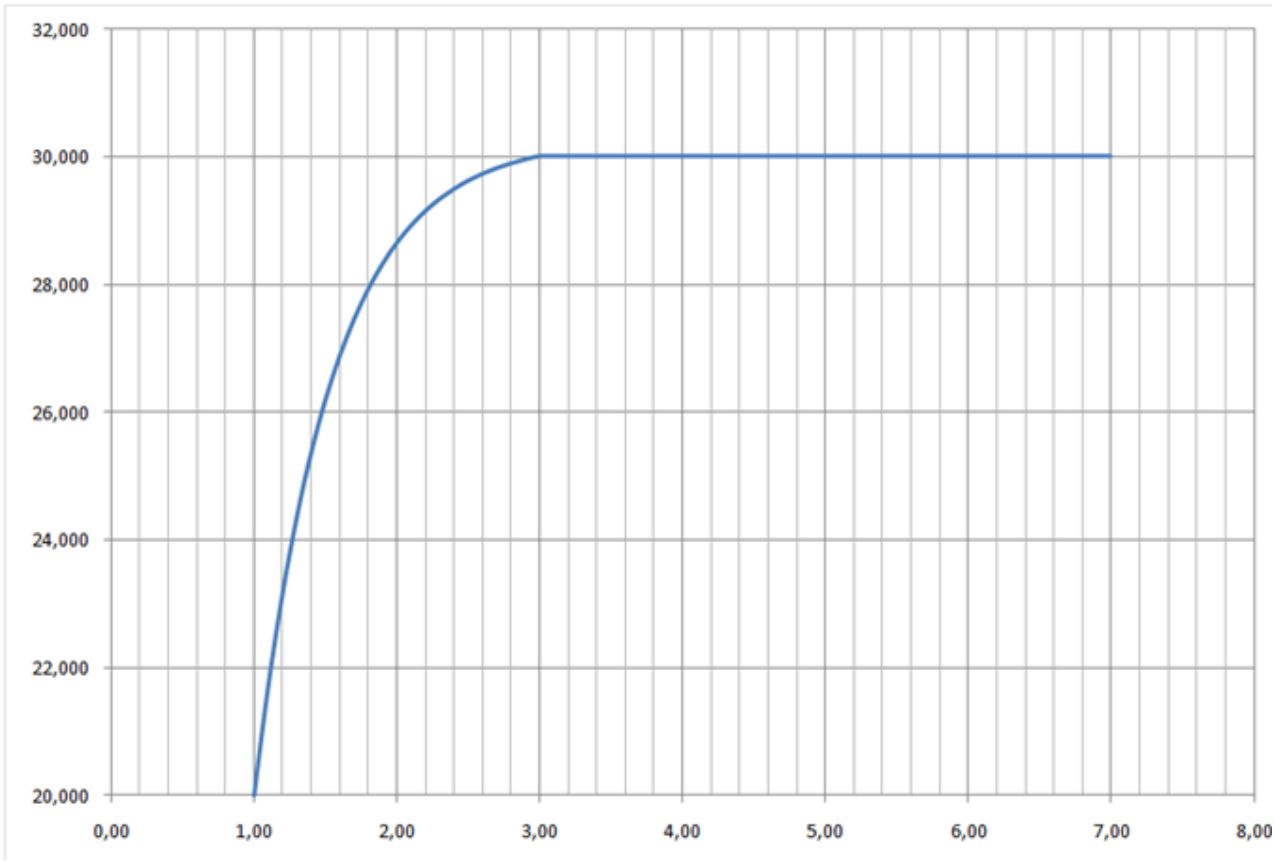
Tensione di controllo del motore

Xsp= 30 Kp= 1 Ki= 0 Kd= 0

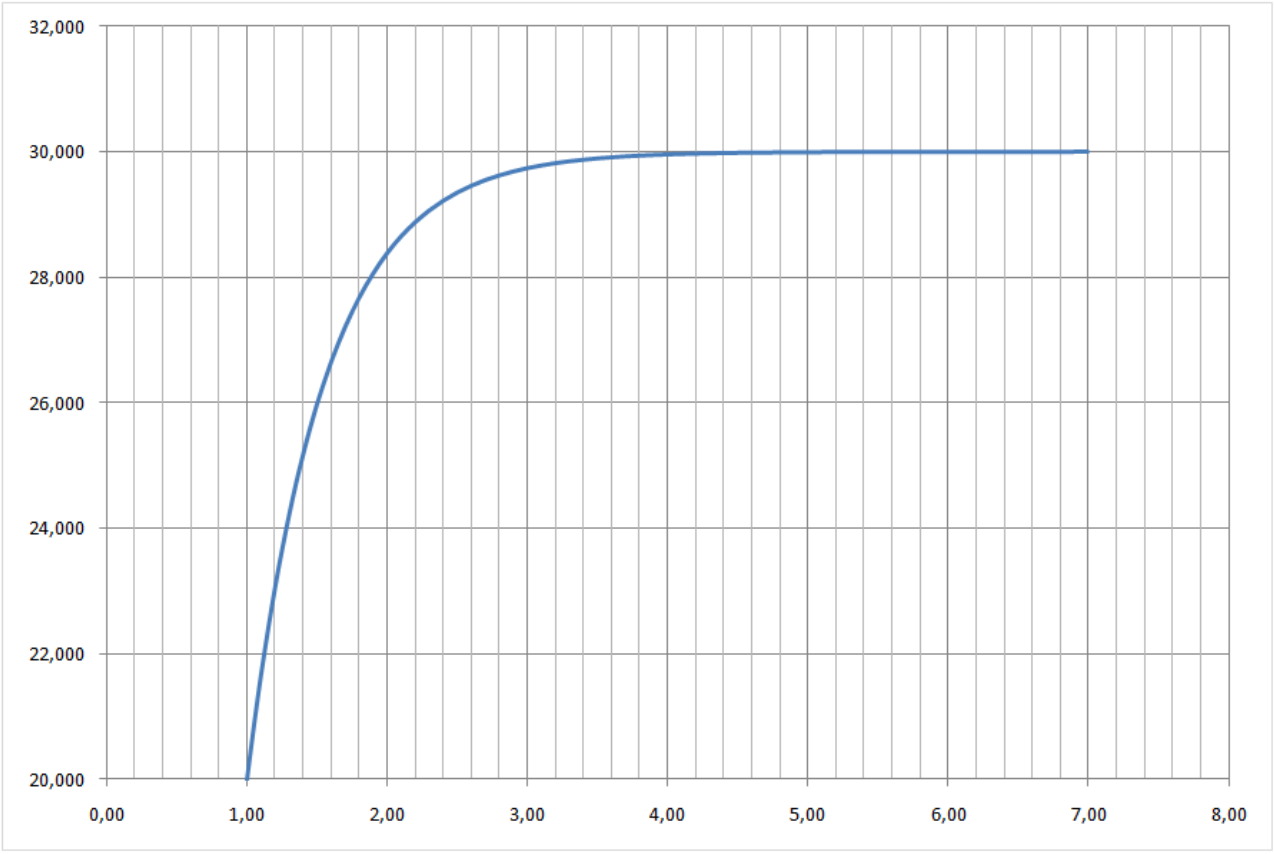
Vmax 12 dt= 0,1

						errore				
t [s]	Vm [V]	n° [giri/s]	s [mm]	vel.	accel.	err. [mm]	integrale	derivata	Vm_teor.	Vm_eff [V]
0,00	12,00	10,00	0,000	0,0	0,0	30,000	0,000	0,000	=$D$2*G6+$F$2*H6+$H$2*I6	

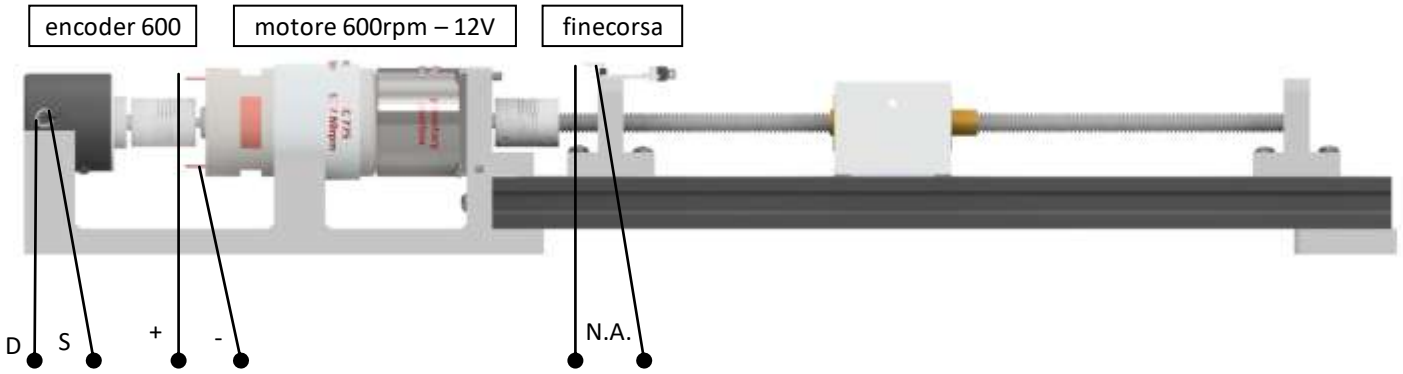
Con il PID il posizionamento si raggiunge in oltre 3 secondi con una a=-26 mm/s2

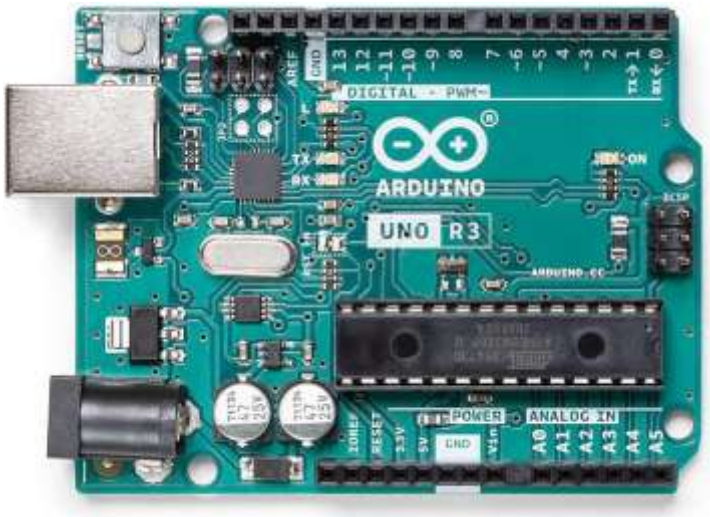


Senza integrale e derivata dell' errore il posizionamento si raggiunge in oltre 4 secondi con una a=-33 mm/s2



SCHEMA SISTEMA DI CONTROLLO POSIZIONE CON ARDUINO E TRANSISTOR DI POTENZA TIP120



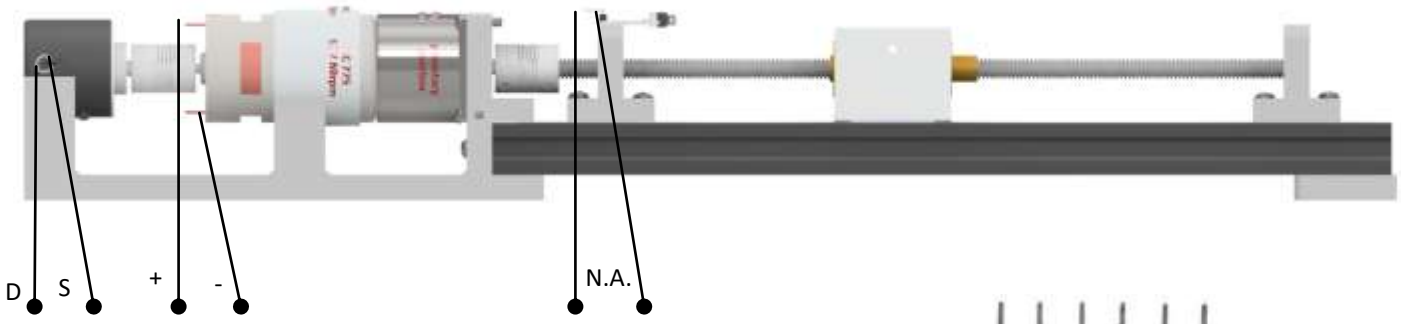


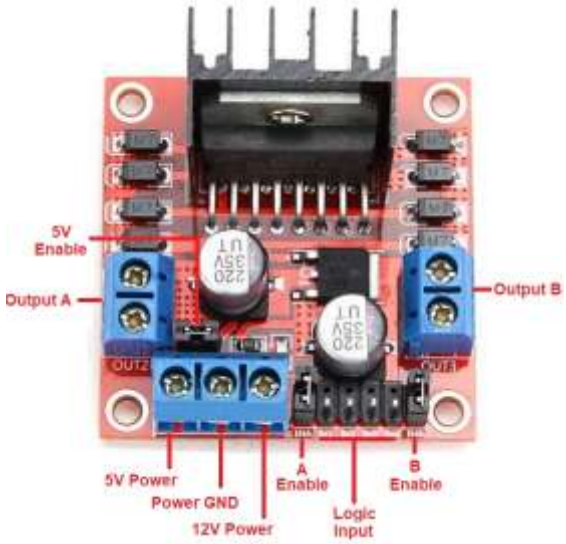
Completare lo schema del sistema di controllo.
Dimensionare eventuali componenti mancanti.
Utilizzare il finecorsa meccanico N.A. come un semplice pulsante in modalità PULL-UP (0 se premuto).

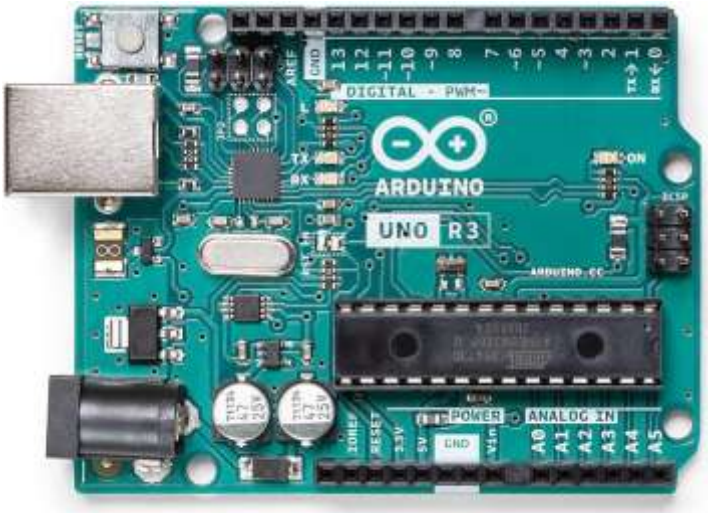
Scrivere il programma Arduino che implementa il sistema di controllo proporzionale della posizione.
Ipotizzare che all'accensione la guida si trovi nella posizione X=0 (finecorsa premuto).

SCHEMA SISTEMA DI CONTROLLO CON TRANSISTOR DI POTENZA TIP120 E PONTE H L298N

encoder 600 | motore 600rpm – 12V | finecorsa







Completare lo schema del sistema di controllo.
Dimensionare eventuali componenti mancanti.
Utilizzare il finecorsa meccanico N.A. come un semplice pulsante in modalità PULL-UP (0 se premuto).

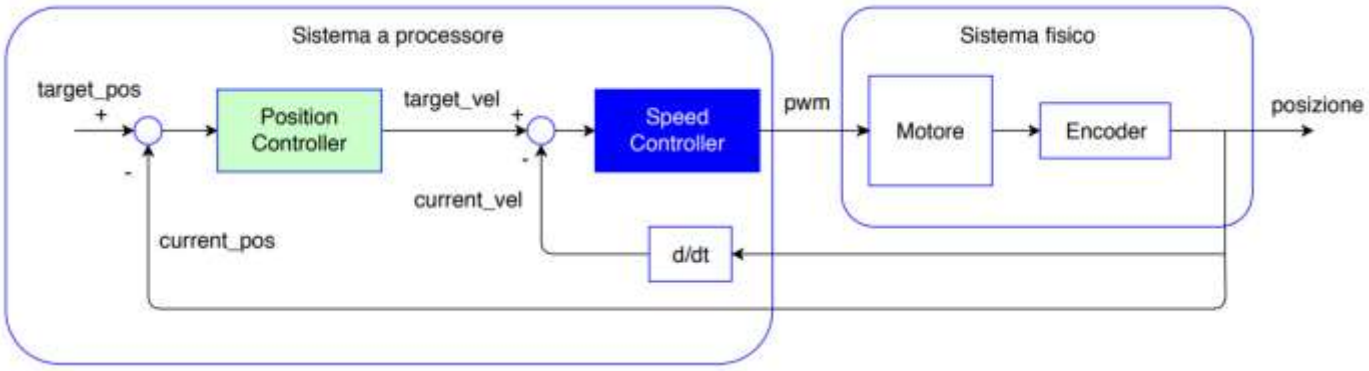
Scrivere il programma Arduino che implementa il sistema di controllo proporzionale della posizione.
All'accensione la guida deve essere portata nella posizione X=0 (finecorsa premuto).

CONTROLLO IN POSIZIONE E IN VELOCITA'

Lo schema usato e quello dei controllori in cascata.
Un controllore di posizione piu esterno (outer loop) fornisce il riferimento (di velocità) a un controllore di velocità interno (inner loop).

Concettualmente: se "siamo lontani" imponiamo una velocità alta che diminuisce man mano che ci si avvicina al target.
Praticamente usiamo un controllore proporzionale:
target_speed = Kp* error_pos = Kp *(target_pos – current_pos) Kp= coeff. proporzionale

Tuttavia, oltre una certa velocità non potremo andare. Pertanto e necessario inserire una saturazione (limite sulla vel. massima).

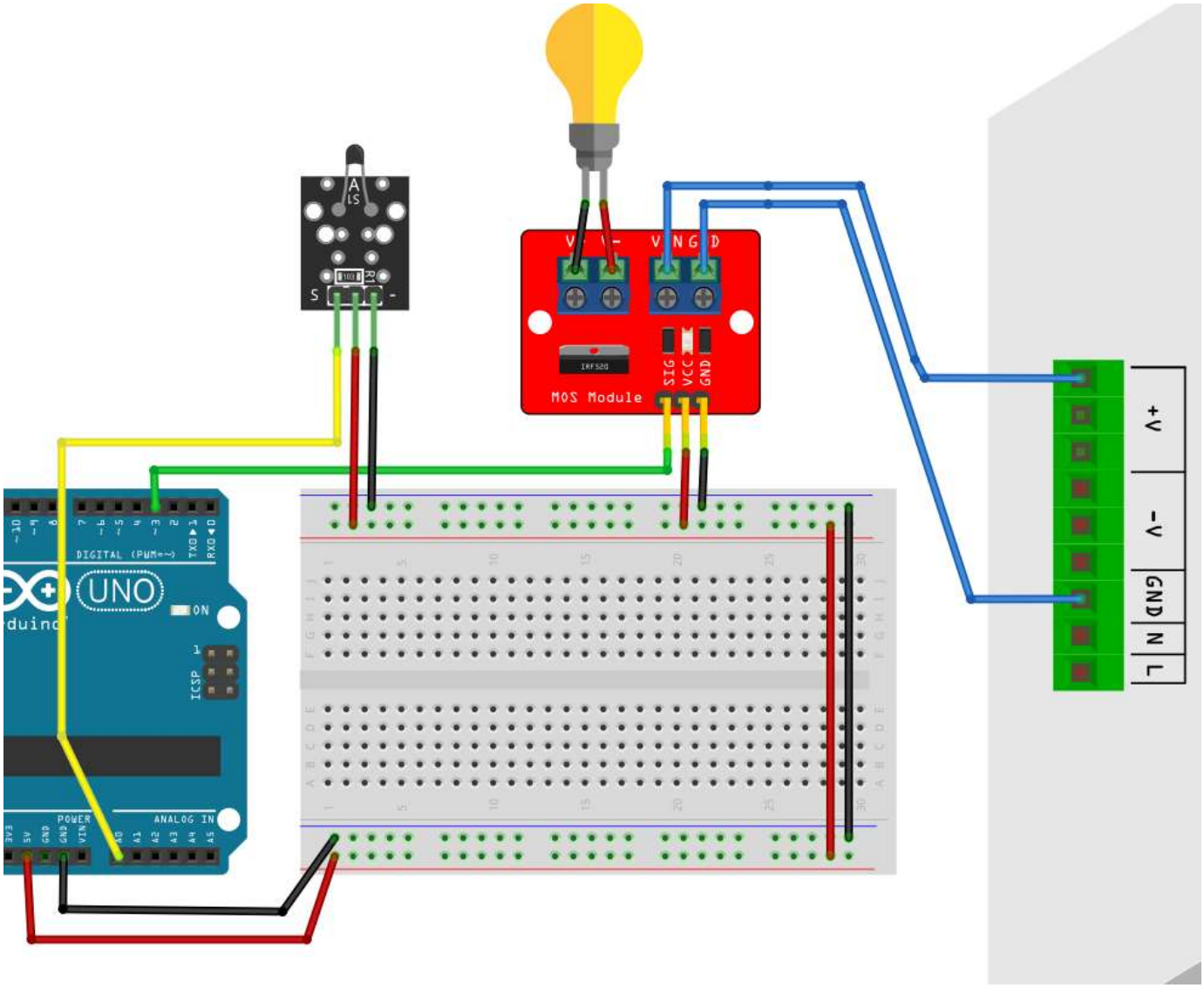


Algoritmo del sistema di controllo

```
void loop(){
  current_pos = read encoder();
  current speed = Δcurrent pos Δt ;
  pos error = target pos − current pos;
  target speed = position controller(pos error);
  speed error = target speed − current speed;
  pwm = speed controller(speed error);
  drive motor(pwm);
  delay( Δt);
}
```

CONTROLLO DI TEMPERATURA “P.I.D.” CON NTC E RF520

Si vuole mantenere costante la temperatura all’interno di un recipiente coibentato.
Come sensore si utilizzi un termistore NTC (RT0: 10KΩ B: 3977 K +- 0.75% T0: 25 C +- 5%).
Come attuatore una lampadina alogena che alimentata a 10V fornisce circa 15w di potenza elettrica.
La lampadina viene attivata da un modulo IRF520 per arduino.
Il tipo di controllo è un P.I.D.

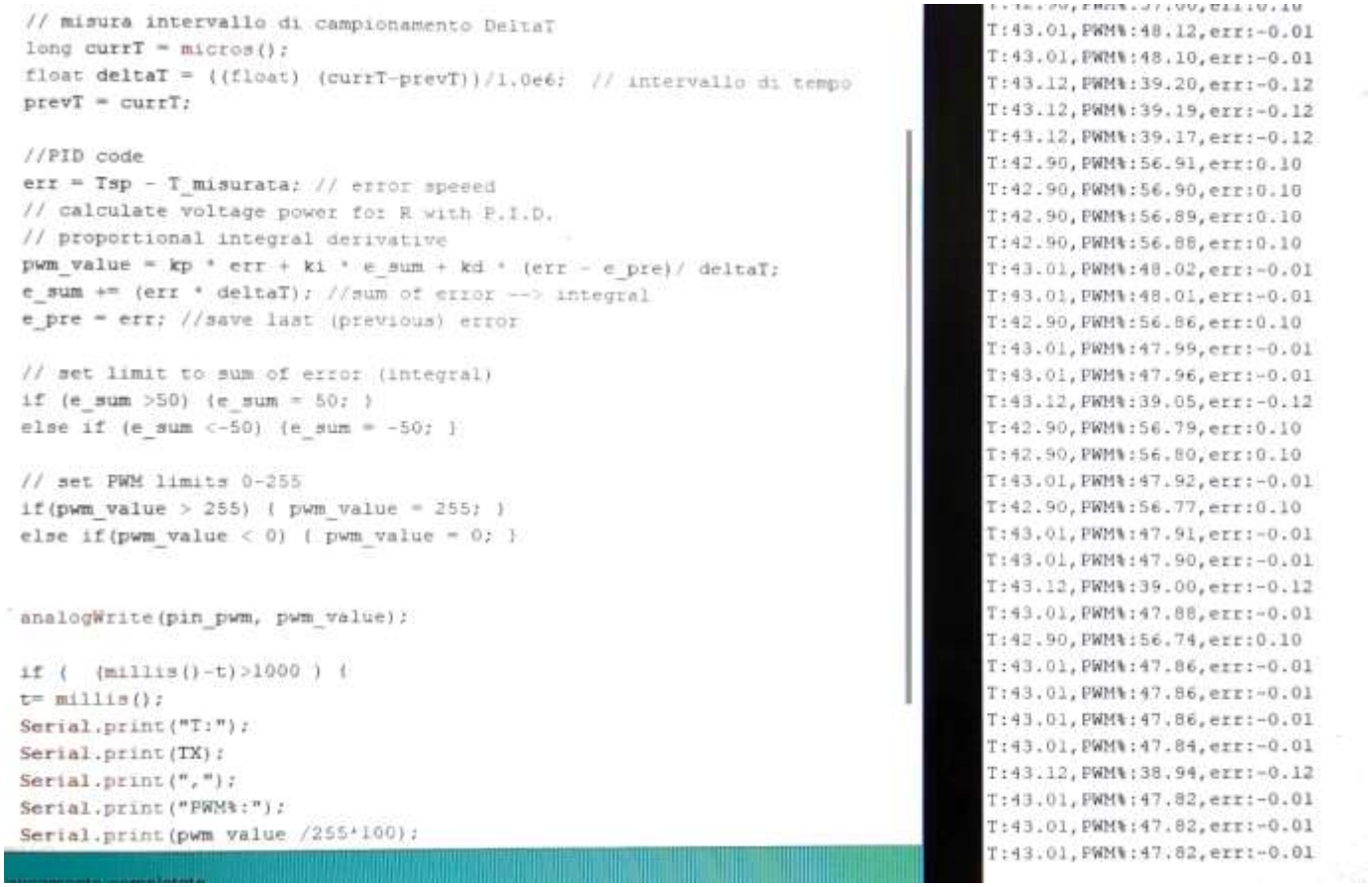


CODICE

```
//Thermistor parameters:  RT0: 10KΩ   B: 3977 K +- 0.75%   T0:  25 C +- 5%
//From datasheet
#define RT0 10000   // Ω
#define B 3977      // K
#define VCC 5    //Supply voltage
#define R 10000  //R=10KΩ
//--------------------------------------

long t;
int pin_pwm=3;
int pin_T=A0;

float Tsp=43.0;
float T_misurata=20.0;
double err;
double e_pre = 0; //last error of speed
double e_sum = 0; //sum error of speed
double pwm_value = 0; //this value is 0~255

long prevT;
long currT;
float deltaT;

double kp = 200;
double ki = 6;
double kd = 0.0;

//Variables per termistore
float RT, VR, ln, TX, T0, VRT;

void setup() {
  pinMode(pin_pwm, OUTPUT);
  pinMode(pin_T, INPUT);
  Serial.begin(9600);
  T0 = 25 + 273.15;
  t= millis();
}

void loop() {
  // misura T con termistore NTC
  VRT = analogRead(pin_T);  // 0-1023 → tensione sul termistore
  VRT = (5.00 / 1023.00) * VRT;      // converto in V
  VR = VCC - VRT;  // tensione sulla resistenza R da 10K
  RT = VRT / (VR / R);    // Resistenza di RT  (V/I)
  ln = log(RT / RT0);
  TX =1/ (ln / B + 1 / T0);  //Temperature from thermistor in K
  TX = TX - 273.15;               //Conversione in  °C
  T_misurata = TX;
  //-------------------------------------------------------

    // misura intervallo di campionamento DeltaT
  currT = micros();
  deltaT = ((float) (currT-prevT))/1.0e6;  // intervallo di tempo
  prevT = currT;

  //PID code
  err = Tsp - T_misurata; // error speeed
  // calculate voltage power for R with P.I.D.
  // proportional integral derivative
  pwm_value = kp * err + ki * e_sum + kd * (err - e_pre)/ deltaT;
  e_sum += (err * deltaT); //sum of error --> integral
  e_pre = err; //save last (previous) error

  // set limit to sum of error (integral)
  if (e_sum >50) {e_sum = 50; }
  else if (e_sum <-50) {e_sum = -50; }
```

```
// set PWM limits 0-255
if(pwm_value > 255) { pwm_value = 255; }
else if(pwm_value < 0) { pwm_value = 0; }


analogWrite(pin_pwm, pwm_value);

if (  (millis()-t)>1000 ) {
t= millis();
Serial.print("T:");
Serial.print(TX);
Serial.print(",");
Serial.print("PWM%:");
Serial.print(pwm_value /255*100);
Serial.print(",");
Serial.print("err:");
Serial.println(err);
}

}
```

Con le costati PID calibrate opportunamente l'errore a regime è di +-0,1°C !!!

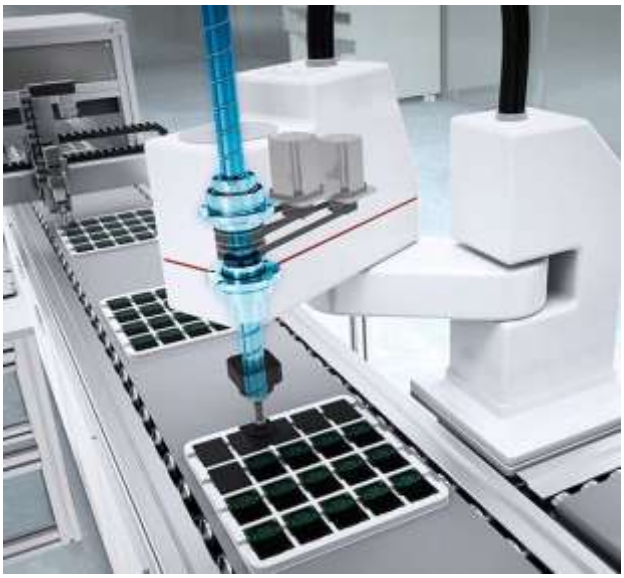


ROBOTICA INDUSTRIALE

La robotica industriale è un settore sviluppato negli ultimi anni, che vede l'utilizzo di sistemi automatici come parte integrante del lavoro industriale. Nella robotica industriale, un sistema di automazione dunque, sostituisce un uomo nella catena di montaggio, compiendo sempre lo stesso lavoro ad un ritmo costante e frenetico.
Tutti gli strumenti meccanici progettati per compiere un determinato lavoro in autonomia rientrano a far parte della robotica industriale. Il sistema automatico nella fattispecie viene chiamato robot.

ISO 8373

Esiste poi un'altra definizione che risale all'ISO 8373, che generalmente rappresenta il vocabolario per la materia di robotica industriale e dei robot implementati in questo settore. La definizione in questione afferma che un robot è un sistema con controllo automatico, riprogrammabile e multi-funzione, che può essere sia fisso a terra sia mobile, di tre o più assi ed il suo scopo è quello di essere utilizzato per operazioni di automazione industriale.

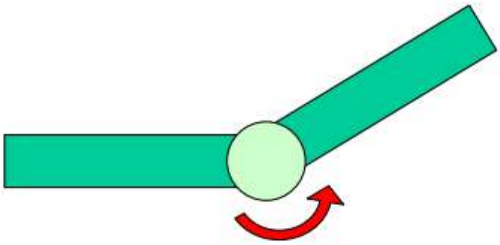


SISTEMI ROBOTICI

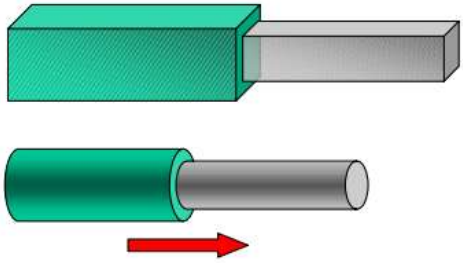
I sistemi robotici sono sistemi costituiti da un insieme di corpi rigidi (link) connessi mediante giunti rotazionali (R) o prismatici (P). Il numero di giunti indipendenti definisce I gradi di libertà del robot DOF (degrees of freedom).

TIPI DI GIUNTO

Le tipologie principali di giunti sono due: rotazionali “R” e prismatici “P”.



R = rotazionale

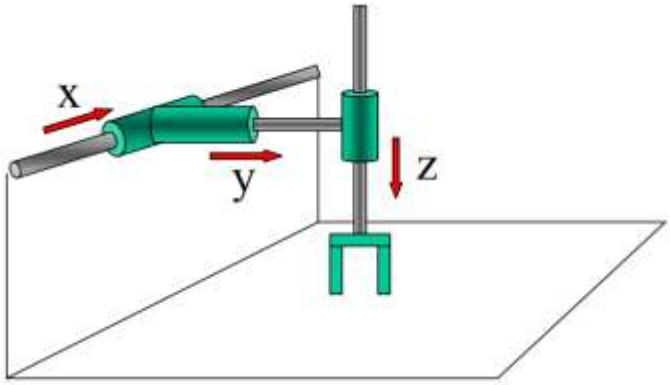


P= prismatico

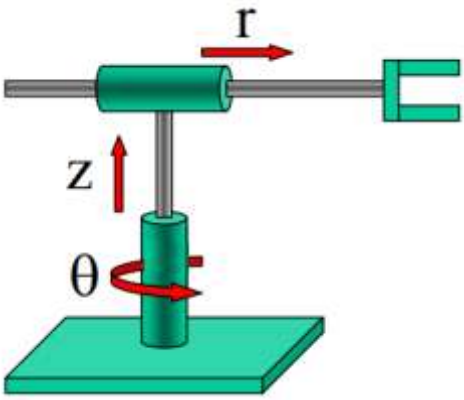
TIPI DI ROBOT

Le tipologie principali di robot si differenziano in base al tipo di giunti adottato.

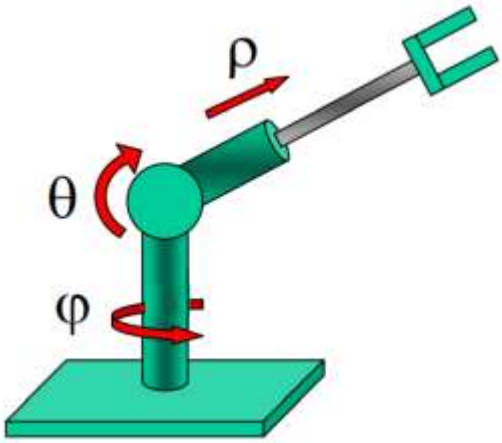
CARTESIANO PPP



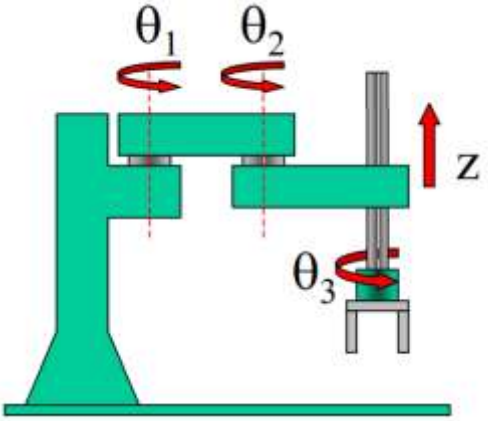
CILINDRICO RPP



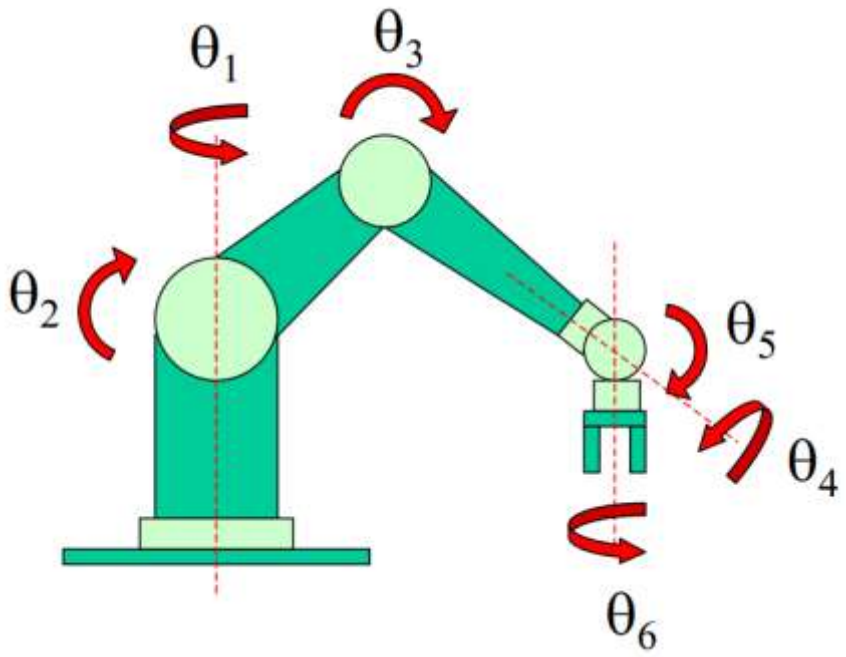
POLARE RRP



SCARA RRP



ANTROPOMORFO RRR



ROBOT COLLABORATIVI (COBOT)

Robot collaborative (collaborative robot) o più semplicemente cobot.
I cobot aiutano ogni giorno gli operatori in tutte le attività pesanti o di precisione, portando nelle filiere qualità, velocità, sicurezza ed efficienza. Il tutto in un'unica soluzione.

INDUSTRIAL ROBOTS

- Installazione difficile
- Alte competenze a livello di programmazione
- Installazioni fisse
- Disponibilità di ampi spazi
- Barriera di sicurezza necessaria
- Elevati costi addizionali

VS.

COLLABORATIVE ROBOTS

- Installazione facile
- Possono essere programmati da chiunque
- Distribuzione flessibile
- Requisiti di spazio ridotti
- Collaborano fianco a fianco con gli operatori umani
- Economicamente convenienti e con un ROI molto rapido

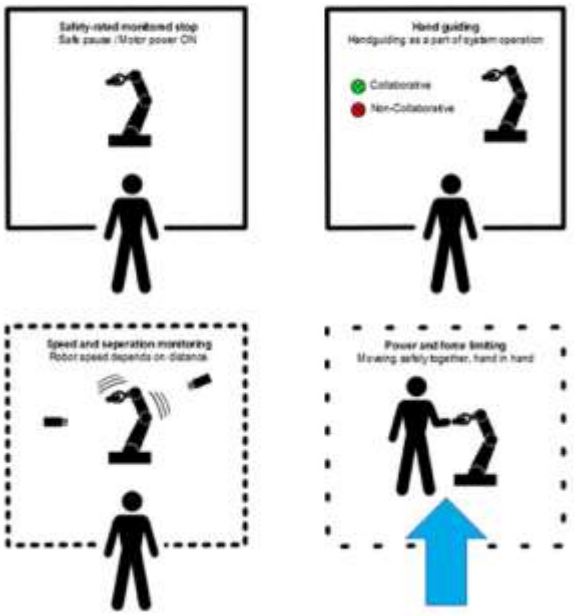
Collaborative Workspace: spazio in cui il sistema robotico e l'uomo possono svolgere compiti contemporaneamente

Collaborative Operation: sistema robotico appositamente progettato per lavorare insieme all'operatore umano

APPLICAZIONI COLLABORATIVE: ATTENZIONE AI RISCHI!!!

ISO 10218
TS 15066

4 TIPOLOGIE DI OPERAZIONI COLLABORATIVE



LE DIFFERENZE TRA ROBOT E COBOT: 4 COSE DA SAPERE

Le normative ISO 10218-1 e -2 definiscono 4 diversi metodi per ottenere operazioni collaborative, mentre l'ISO / TS 15066 aggiunge in termini di Safety& Security ulteriori linee guida e alcuni requisiti aggiuntivi.

ARRESTO MONITORATO

Un robot tradizionale viene utilizzato all'interno di una recinzione. Una persona può entrare nell'area di lavoro attraverso un'apertura (interruttore della porta, barriera fotoelettrica o soluzione di sicurezza della telecamera che identifica che una persona sta entrando). All'ingresso dell'operatore, il robot viene messo in pausa. Una funzione di arresto di sicurezza viene utilizzata mentre la persona entra e fa il suo lavoro (ad es. Carico / scarico di pezzi). Quando la persona lascia l'area di lavoro, il robot può riprendere il funzionamento automatico. Chiamare questo metodo collaborativo sembra un po' strano, ma gli standard lo definiscono collaborativo perché l'elettricità ai motori dei robot viene mantenuta mentre è presente una persona. La sicurezza dei cobot è tale per cui non sono previste recinzioni e la messa in pausa non deve essere programmata perché avviene in automatico.

GUIDA MANUALE

La guida manuale fa parte del meccanismo di funzionamento di un robot che, tipicamente, esegue il compito che gli è stato assegnato. Un task che non va confuso con la programmazione manuale dei cobot. Questo metodo utilizza un robot tradizionale all'interno di una recinzione. Una persona entra periodicamente nella cella per eseguire un compito come, ad esempio, avvitare alcune viti. Quando entra, il robot industriale tradizionale passano da modalità non collaborativa a modalità collaborativa (ad es. Velocità massima 100 mm / s e movimento di regolazione massima +/- 50 mm). L'industria automobilistica utilizza questo metodo da anni per posizionare le sedie all'interno delle automobili e per tenere i paraurti mentre vengono avvitati. La programmazione di un cobot non prevede alcuna compilazione del codice da parte degli utenti: basta scaricare il software funzionale ai task e impostare in maniera intuitiva l'iter dei vari movimenti che eseguirà il cobot (potendoli modificare quando necessario).

MONITORAGGIO DELLA VELOCITÀ E DELLA SEPARAZIONE

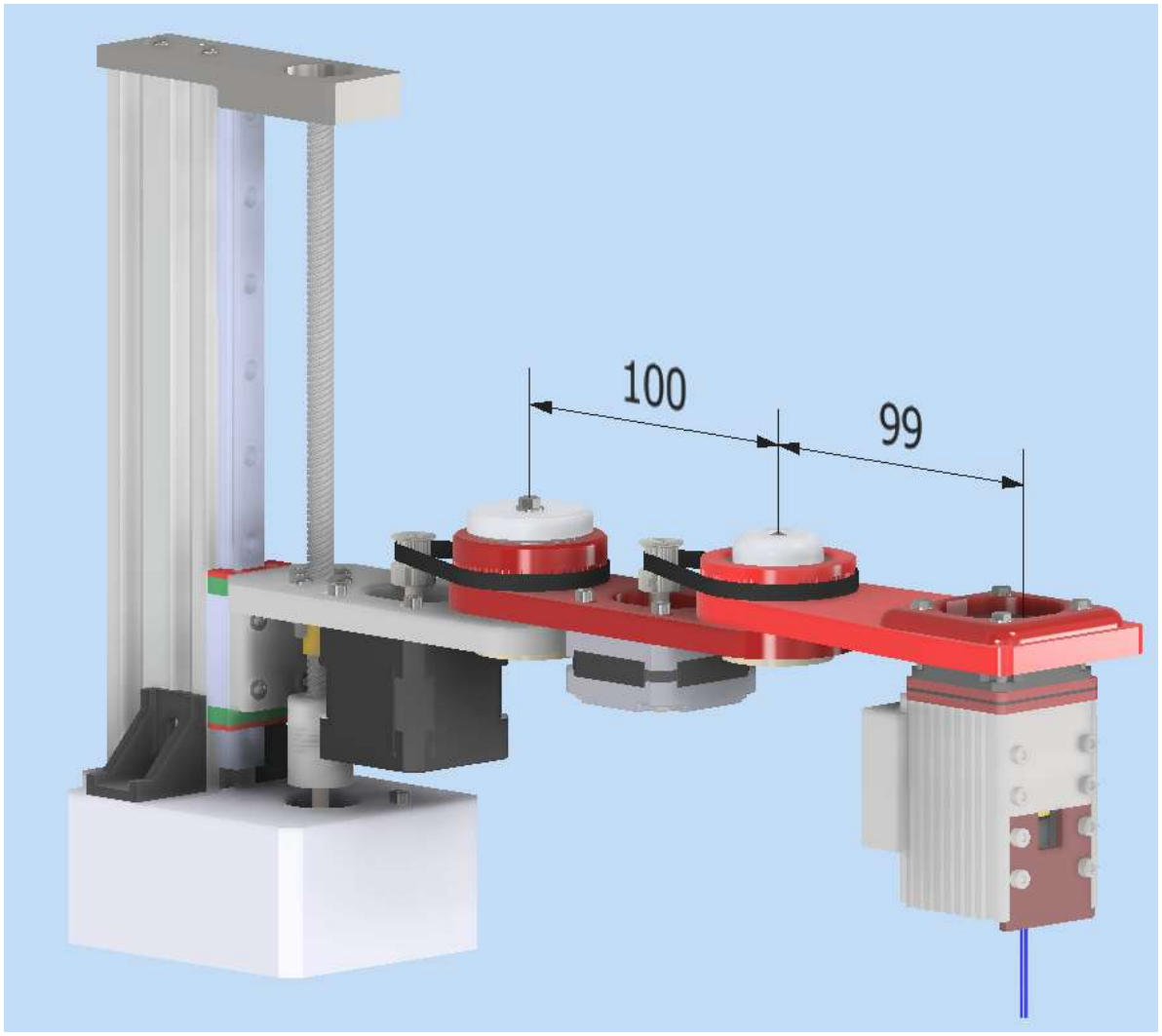
Questo metodo è simile allo "stop di sicurezza monitorato" ma, invece di mettere in pausa il robot, la procedura riduce la velocità del robot in base alla distanza tra il robot e l'operatore. Un modo per farlo è utilizzare una telecamera che determina la distanza in modo sicuro. Un altro modo è utilizzare una pellicola sensorizzata integrata nel robot, che percepisce quando una persona è vicina al robot e quindi lo blocca prima che questo arrivi a toccare l'operatore. Il cobot, essendo dotato di appositi sensori, si muove mantenendo sempre una distanza di sicurezza da cose e persone, arrivando a bloccarsi e a ripartire da solo.

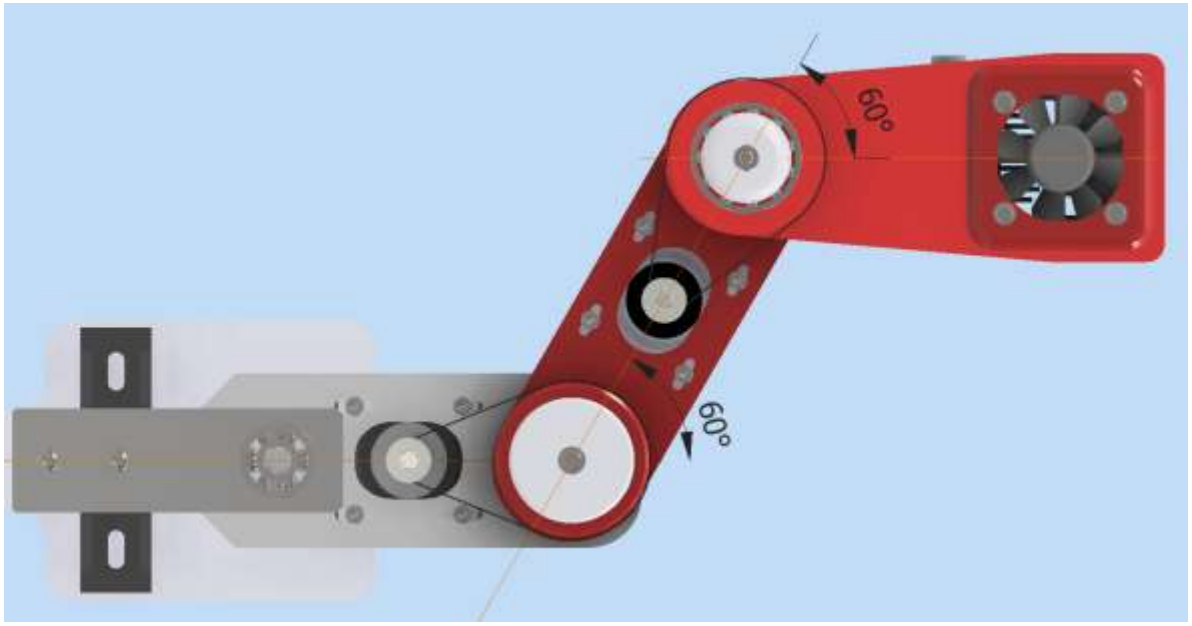
LIMITAZIONE DI POTENZA E FORZA

Una caratteristica unica dei cobot UR è la limitazione della potenza e della forza. I cobot, infatti, sono sensibili al movimento e quindi si fermano prima di mettere a rischio l'operatore. La regolazione della velocità, della forza e della pressione, infatti, sono determinanti e sono le caratteristiche peculiari dei robot collaborativi UR.

ROBOT PLANARE

Un robot planare in genere presenta solo 2 gradi di libertà.
In un piano orizzontale si muovono 2 bracci articolati, incernierati ad una estremità con un asse verticale fisso, mentre all'altra estremità libera si trova l'end effector (ad es. laser).
In alcune applicazioni è previsto un terzo asse che permette un movimento lineare verticale dei 2 bracci (3 gradi di libertà).





Il **PETG** è un capoliestere di polietilene tereftalato trasparente: è una versione modificata di PET.
La "G" sta per "glicole modificato", che viene aggiunto alla composizione del materiale durante la polimerizzazione.

Proprietà fisiche	Valore tipico	Metodo
Peso specifico [g/cm3]	1.27	ISO 1183
Assorbimento umidità in 24 ore [%](1)	0.2	Prusa Polymers
Assorbimento umidità in 7 giorni [%](1)	0.3	Prusa Polymers
Assorbimento umidità in 4 settimane [%](1)	0.3	Prusa Polymers
Temperatura di deflessione del calore (0,45 MPa) [°C]	68	ISO 75
Filamento di resistenza alla trazione [MPa]	46 ± 1	ISO 527

PROPRIETÀ MECCANICHE DEI CAMPIONI DI PROVA STAMPATI

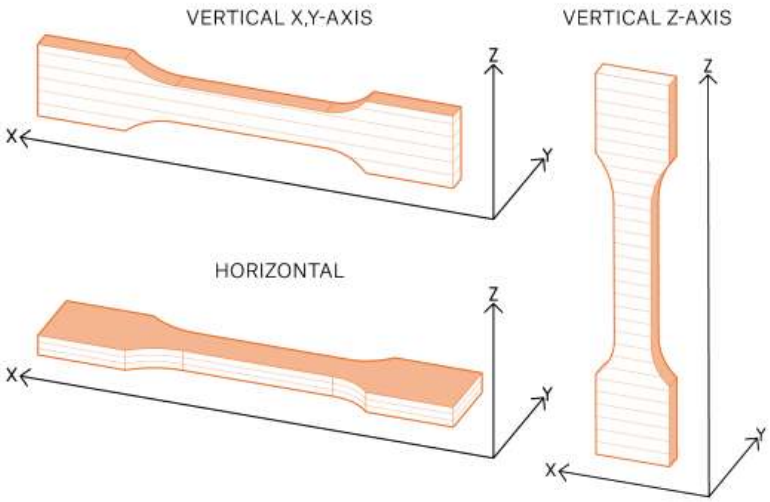
Proprietà / direzione stampa	Orizzontale	Verticale Asse-X,Y	Asse Z verticale	Metodo
Resistenza alla trazione [MPa]	47 ± 2	50 ± 1	30 ± 5	ISO 527-1
Modulo di trazione [GPa]	1.5 ± 0.1	1.5 ± 0.1	1.4 ± 0.1	ISO 527-1
Allungamento al punto di trazione [%]	5.1 ± 0.1	5.1 ± 0.1	2.5 ± 0.5	ISO 527-1
Resistenza all'urto Charpy (3)[kJ/m2]	NB(C)(4)	NB(4)	5 ± 1	ISO 179-1

(1) 30 °C; umidità 30%

(2) La stampante 3D Original Prusa i3 MK3 è stata usata per i campioni di prova. È stato utilizzato Slic3r Prusa Edition 1.40.0 per generare i G-code con le seguenti impostazioni: Filamento Prusa PETG; Impostazioni di stampa 0,20mm FAST (layer 0,2mm); layer solidi superiori:0 Inferiori:0; Riempimento 100% Rettilineo, velocità di stampa riempimento 100mm/s; moltiplicatore estrusione 1.07; temperatura di estrusione 260°C per tutti i layer; temperatura piano 90°C per tutti i layer; altri parametri impostati come predefinito

(3) Charpy non intagliato - Direzione laterale del colpo secondo ISO 179-1

(4) NB (nessuna rottura); C (rottura completa) tra parentesi secondo tipo di guasto più frequente > 1/3



** Modulo trazione: gradiente della curva nel diagramma sforzo-deformazione*

PETG Veralite® 200 - Scheda tecnica			
PROPRIETA' FISICHE			
Proprietà	Metodo	Unità di misura	Veralite® 200
Peso specifico	ISO 1183	g/cm^3	1,27
Assorbimento d'acqua	ISO 62	%	0,15
PROPRIETA' MECCANICHE			
Proprietà	Metodo	Unità di misura	Veralite® 200
Resistenza alla trazione	ISO 527	Mpa	51,5
Allungamento a rottura	ISO 527	%	> 100
Modulo di trazione	ISO 527	Mpa	± 2200
Resistenza all'urto senza intaglio	ISO 180	KJ/m^2	no scoppio
Resistenza all'urto con intaglio	ISO 180	KJ/m^2	9,0
Durezza Rockwell	DIN 2039	M/R	M85/R115
PROPRIETA' TERMICHE			
Proprietà	Metodo	Unità di misura	Veralite® 200
Coefficiente di dilatazione	ASTM D696	mm/mC°	± 0,060
Calore specifico	DSC	J/gC°	1,13
Temperatura di inflessione di calore (0,45 Mpa)	ISO 75	°C	72
Temperatura di inflessione di calore (1,82 Mpa)	ISO 75	°C	68
Punto di rammollimento Vicat (1kg)	ISO 306	°C	82
Punto di rammollimento Vicat (5kg)	ISO 306	°C	72
PROPRIETA' OTTICHE			
Proprietà	Metodo	Unità di misura	Veralite® 200
Trasmissione della luce	ASTMD 1003	%	86 - 90*
Satin	ASTMD 1003	%	< 1
Lucidatura	ASTMD 1003	units	159

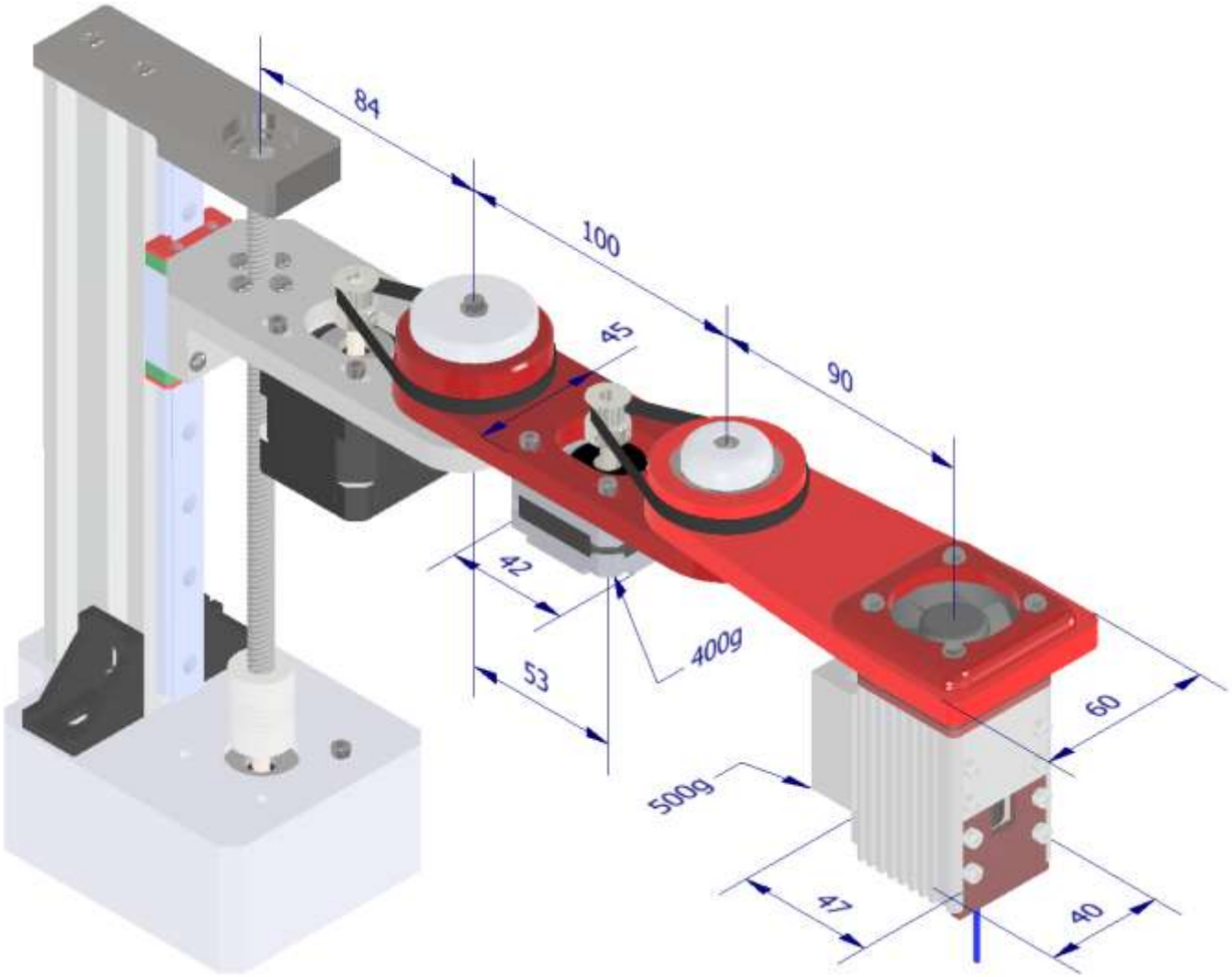
DIMENSIONARE I LINK DEL ROBOT PLANARE ASSEGNATO

Dimensionare i link (sezione rettangolare) del robot planare assegnato sapendo che sono realizzati in material **PETG**.
La velocità massima del motore stepper è di 1200 rpm.
Contenere la deformazione dei link (freccia) nella posizione assegnata a 0.5mm..

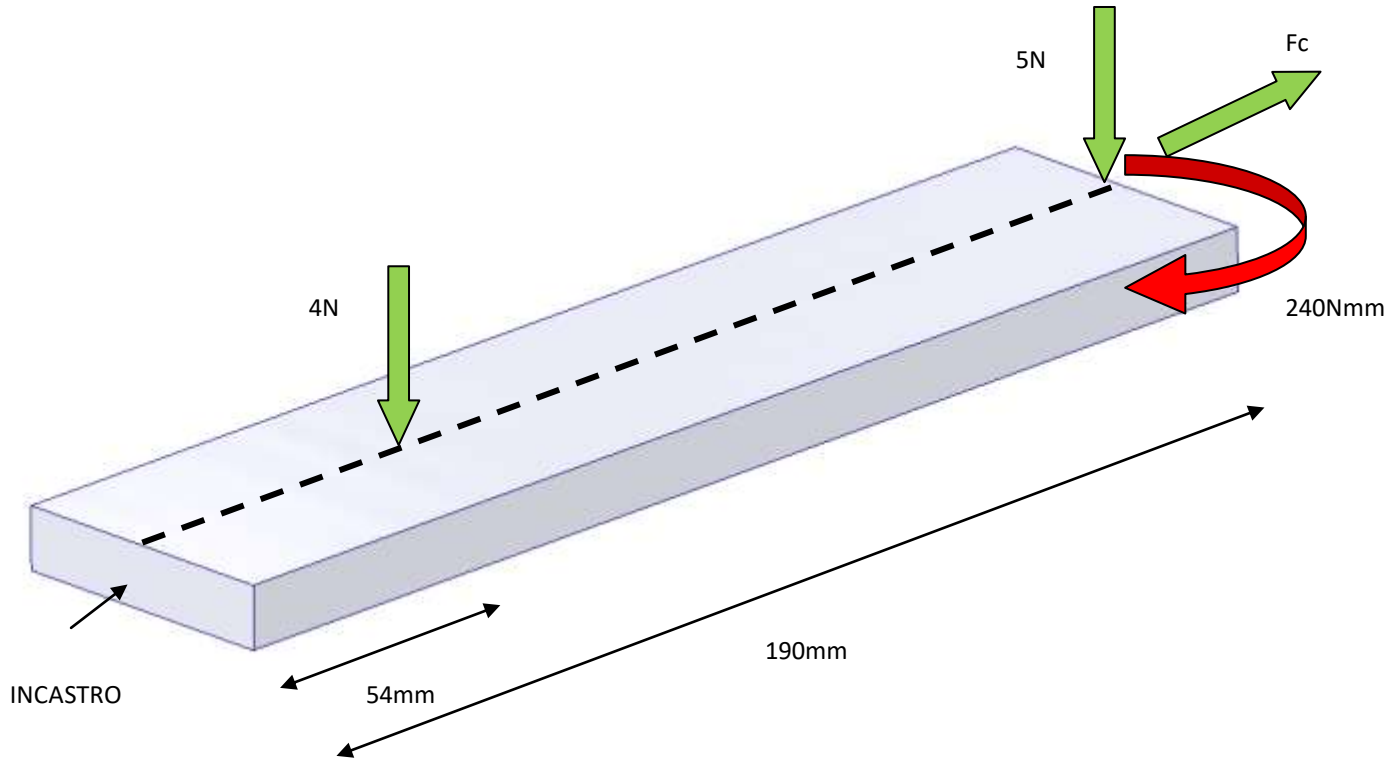
Motore NEMA 17

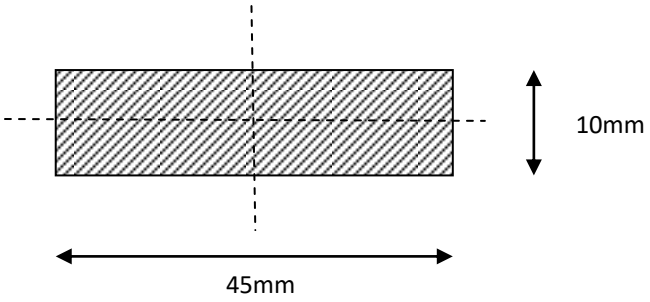
Passo Angle 1.8 °	Coppia motrice massima 59 Nmm (83.6oz.in)
Corrente nominale/phase2.0A	Fase Resistance 1.4ohms
Voltage2.8V	Inductance 3.0mH ± 20%(1KHz)
Weight 400g	

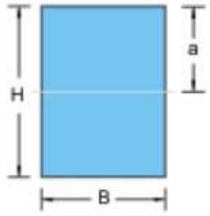
Laser 10watt: massa=500g; parallepipedo 47x40mm



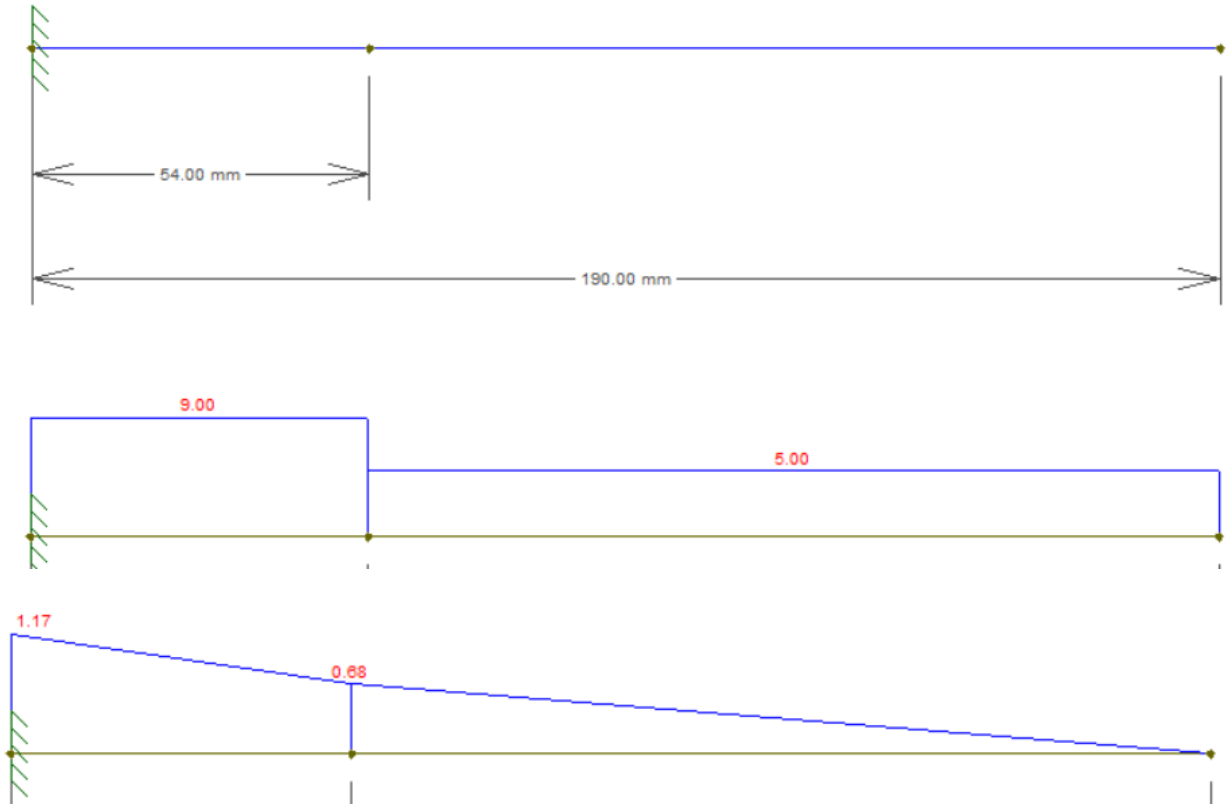
SOLLECITAZIONI SUI LINK DEL ROBOT PLANARE NELLA POSIZIONE DISTESA





Sezione	Area della sezione A	Distanza dal baricentro a	Momento di inerzia J	Modulo di resistenza W
	cm^2	cm	cm^4	cm^3
	$B \cdot H$	$\frac{H}{2}$	$\frac{B \cdot H^3}{12}$	$\frac{B \cdot H^2}{6}$

PIANO VERTICALE: TAGLIO + FLESSIONE



Tmax= 9N

Mf max=1.17Nm

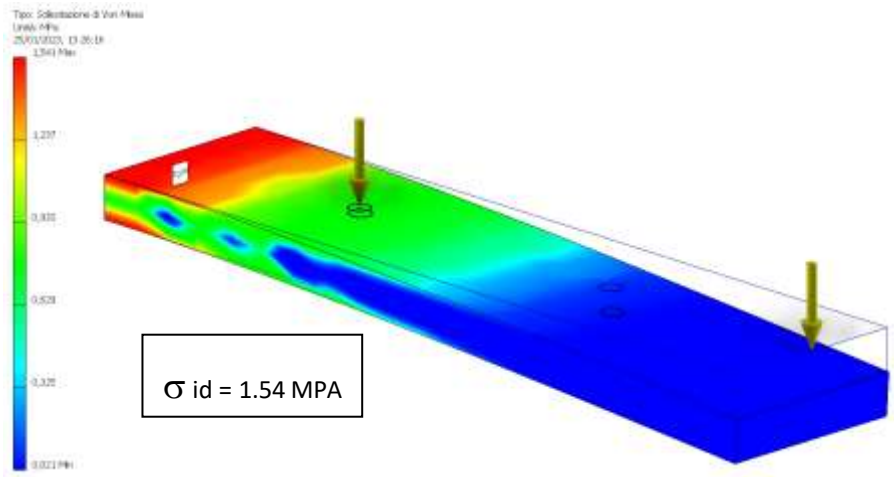
PIANO ORIZZONTALE

Mf max=0.24Nm

CALCOLO SFORZI E DEFORMAZIONE PETG NELLA POSIZIONE DISTESA

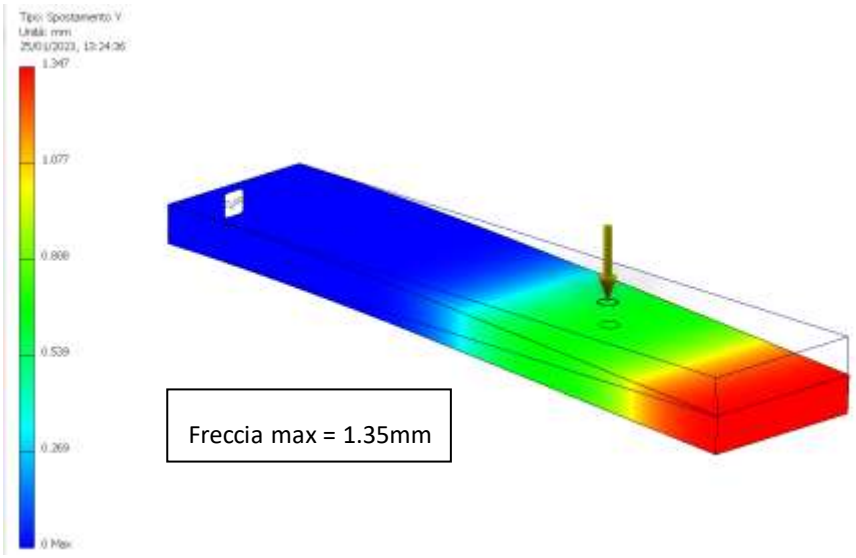
PETG

R	50	N/mm2
E	2200	N/mm2
ksic. min	4,5	1,5*3
σ amm	11,11	N/mm2
τ amm	6,40	N/mm2
b	45	mm
h	10	mm
l	190	mm
l'	54	mm Nema
Wfx=bh^2/6	750	mm3
Ix=bh^3/12	3750	mm4
Wfy=hb^2/6	3375	mm3



SOLLECITAZIONI MASSIME

Mf max verticale	1170	Nmm
T max verticale	9	N
Mf max orizz.	240	Nmm



FORZA CENTRIFUGA (m*w^2*r)

ngiri	1200	rpm
ω	125,66	rad/s
massa Laser	0,5	Kg
Fc laser	1500,18	N
massa Nema 17	0,40	Kg
Fc motore	341,09	N

FLESSIONE

σ max vert.	1,56	N/mm2
σ max orizz.	0,07	N/mm2

TRAZIONE

σ max	4,09	N/mm2

SFORZO ASSIALE MASSIMO

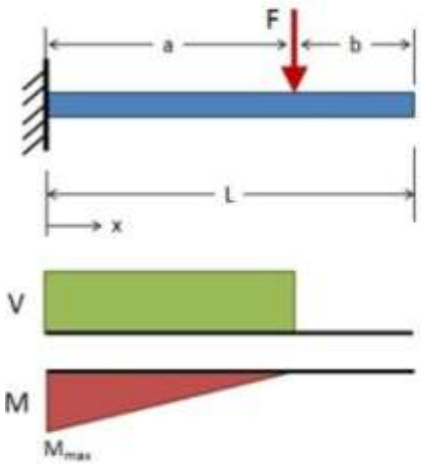
σ max tot.	5,72	N/mm2

TAGLIO

τ max	0,03	N/mm2

DEFORMAZIONE ELASTICA

F1	4,0	
F2	5,0	
R	9,0	N
distanza R	129,6	mm
deformazione	1,34	mm



Freccia:

$$\delta = -\frac{Fx^2}{6EI}(3a - x) \qquad (0 \le x \le a)$$

$$\delta = -\frac{Fa^2}{6EI}(3x - a) \qquad (a \le x \le L)$$

$$\delta_{max} = \frac{Fa^2}{6EI}(3L - a) \qquad @\, x = L$$

Pendenza:

$$\theta = -\frac{Fx}{2EI}(2a - x) \qquad (0 \le x \le a)$$

$$\theta = -\frac{Fa^2}{2EI} \qquad (a \le x \le L)$$

Taglio:

$$V = +F \qquad (0 \le x \le a)$$

$$V = 0 \qquad (a \le x \le L)$$

Momento flettente:

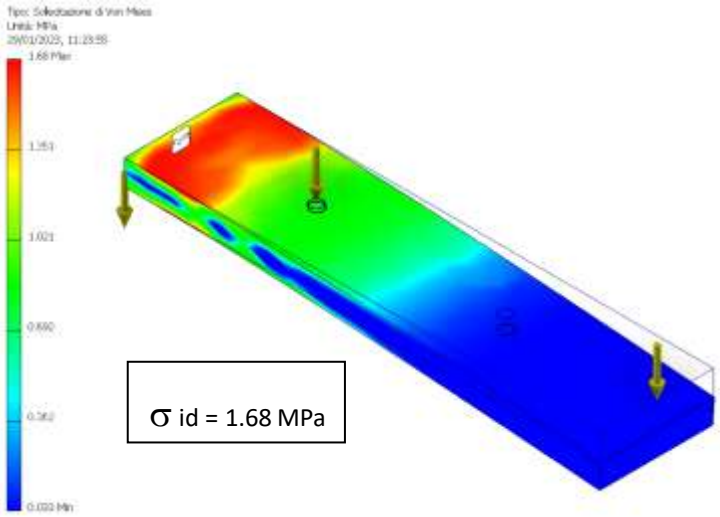
$$M = -F(a - x) \qquad (0 \le x \le a)$$

$$M = 0 \qquad (a \le x \le L)$$

CALCOLO SFORZI E DEFORMAZIONE ALLUMINIO 6061 NELLA POSIZIONE DISTESA

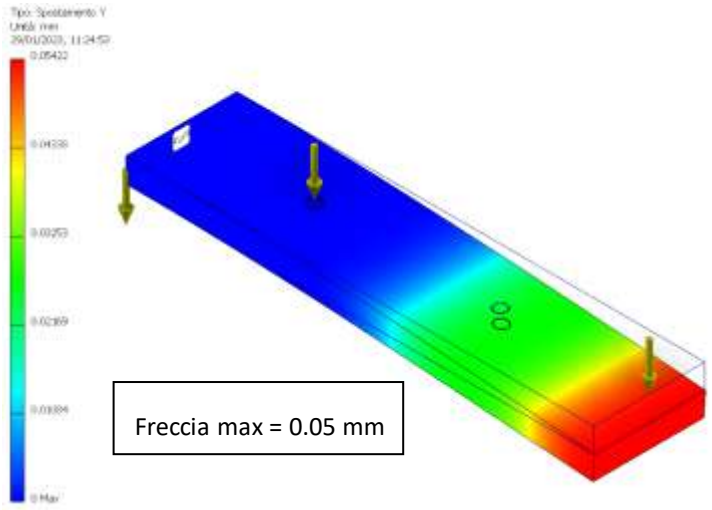
ALLUMINIO

R	309,99	N/mm2
E	68899	N/mm2
ksic. min	4,5	1,5*3
σ amm	68,89	N/mm2
τ amm	39,68	N/mm2
b	45	mm
h	10	mm
l	190	mm
l'	54	mm Nema
Wfx=bh^2/6	750	mm3
Ix=bh^3/12	3750	mm4
Wfy=hb^2/6	3375	mm3



SOLLECITAZIONI MASSIME

Mf max verticale	1170	Nmm
T max verticale	9	N
Mf max orizz.	240	Nmm



FORZA CENTRIFUGA (m*w^2*r)

ngiri	1200	rpm
ω	125,66	rad/s
massa Laser	0,5	Kg
Fc laser	1500,18	N
massa Nema 17	0,40	Kg
Fc motore	341,09	N

FLESSIONE

σ max vert.	1,56	N/mm2
σ max orizz.	0,07	N/mm2

TRAZIONE

σ max	4,09	N/mm2

SFORZO ASSIALE MASSIMO

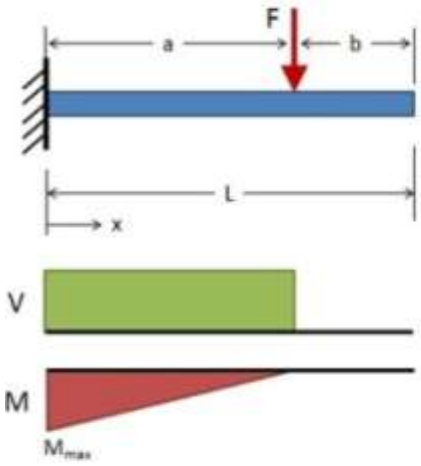
σ max tot.	5,72	N/mm2

TAGLIO

τ max	0,03	N/mm2

DEFORMAZIONE ELASTICA

F1	4,0	
F2	5,0	
R	9,0	N
distanza R	129,6	mm
deformazione	0,04	mm



Freccia:

$$\delta = -\frac{Fx^2}{6EI}(3a - x) \qquad (0 \le x \le a)$$

$$\delta = -\frac{Fa^2}{6EI}(3x - a) \qquad (a \le x \le L)$$

$$\delta_{max} = \frac{Fa^2}{6EI}(3L - a) \qquad @\, x = L$$

Pendenza:

$$\theta = -\frac{Fx}{2EI}(2a - x) \qquad (0 \le x \le a)$$

$$\theta = -\frac{Fa^2}{2EI} \qquad (a \le x \le L)$$

Taglio:

$$V = +F \qquad (0 \le x \le a)$$

$$V = 0 \qquad (a \le x \le L)$$

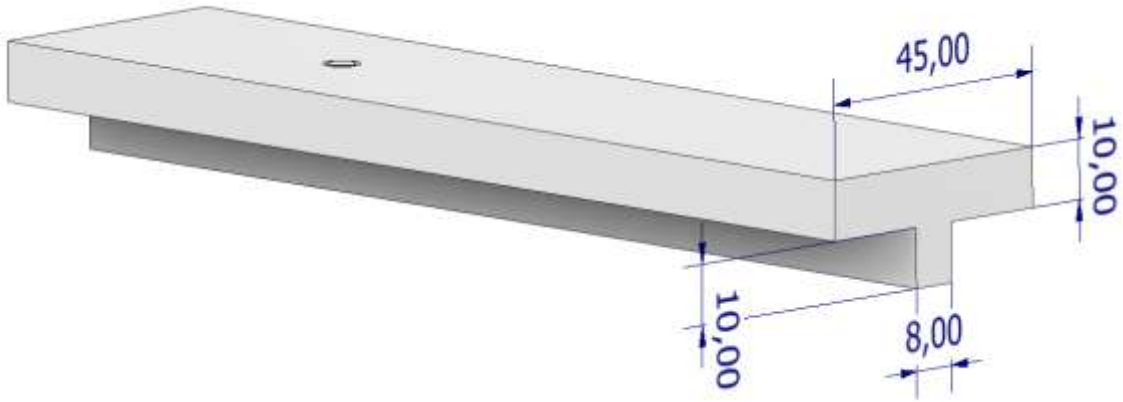
Momento flettente:

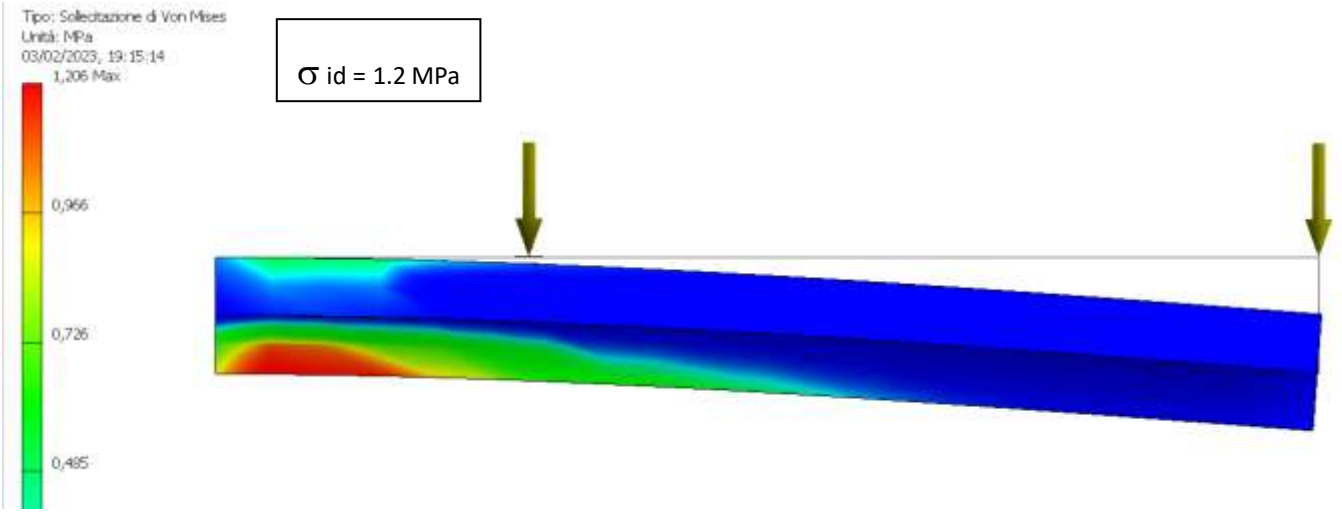
$$M = -F(a - x) \qquad (0 \le x \le a)$$

$$M = 0 \qquad (a \le x \le L)$$

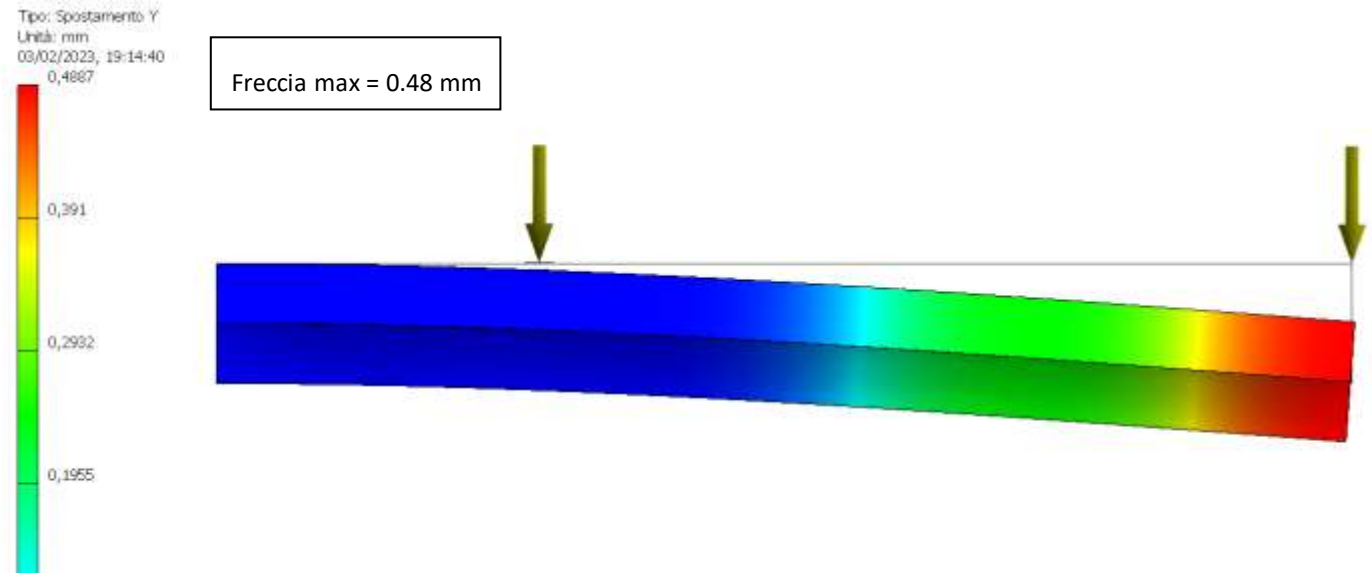
MIGLIORARE LA RESISTENZA A DEFORMAZIONE ELASTICA TRAMITE NERVATURE LATERALI

Adottoando un profile a T si ottiene un netto miglioramento.



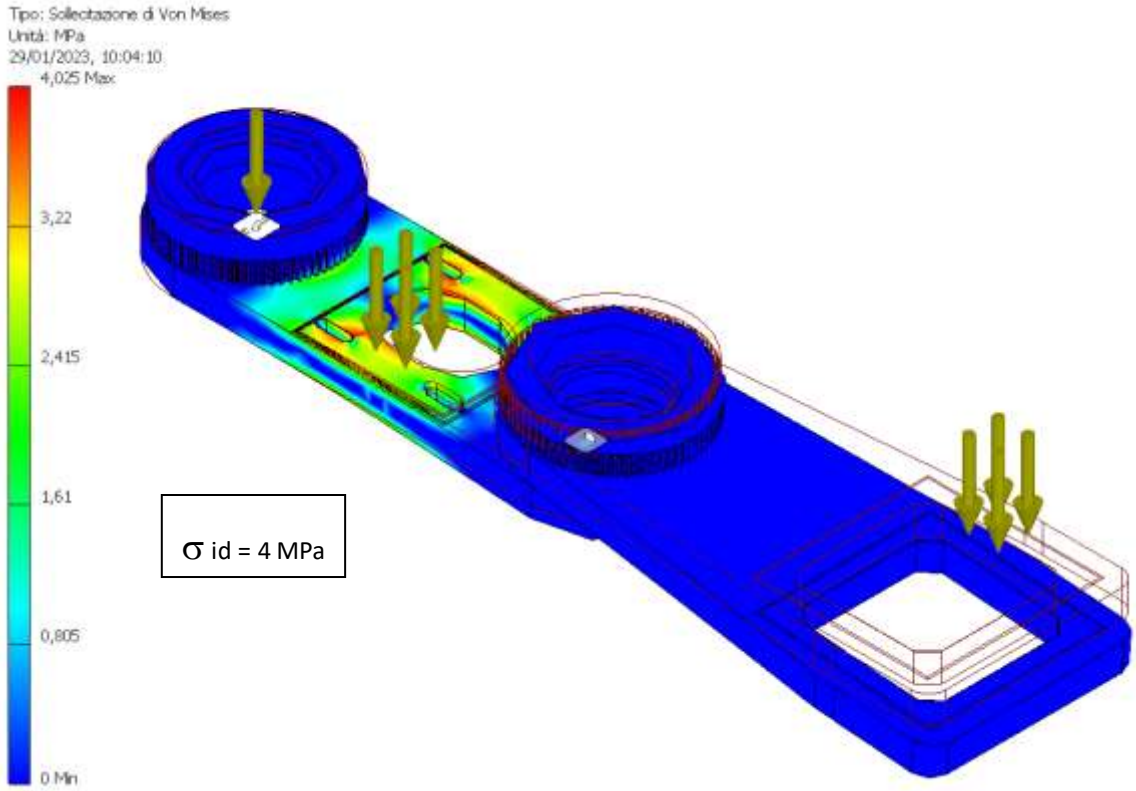


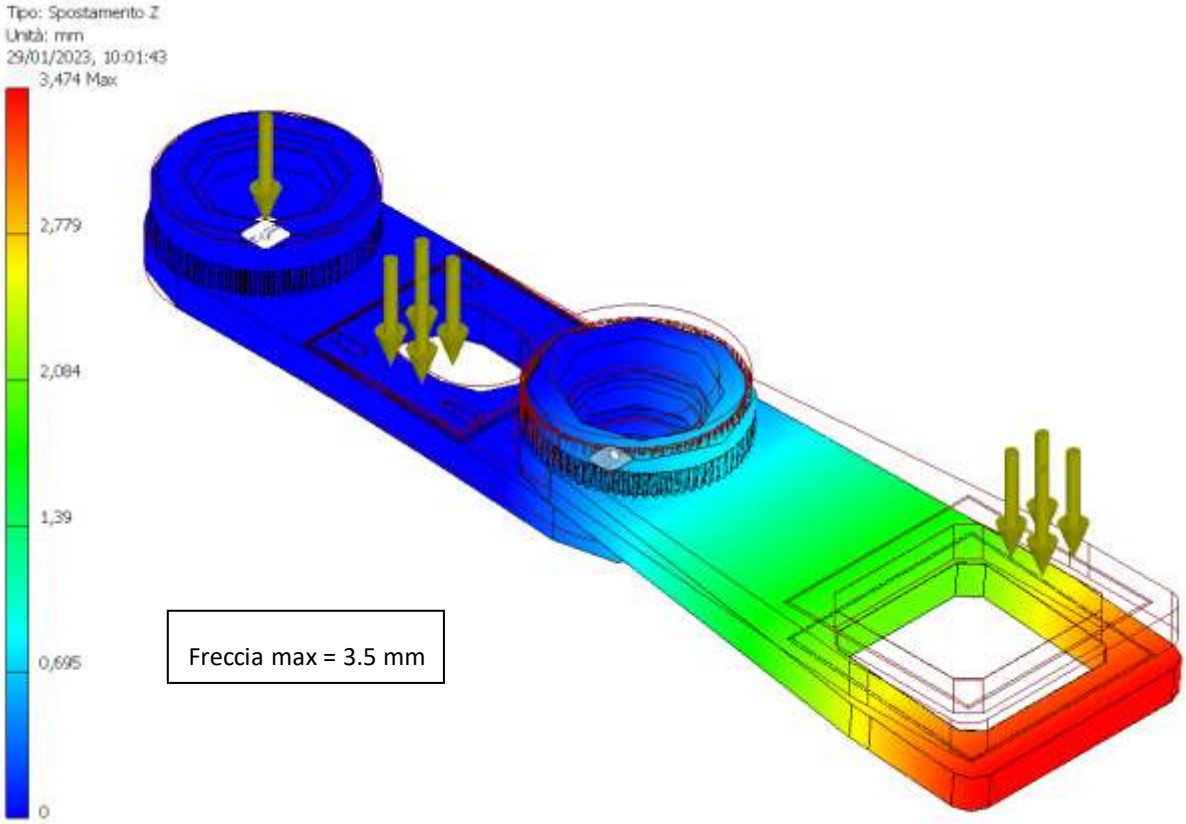
σ id = 1.2 MPa



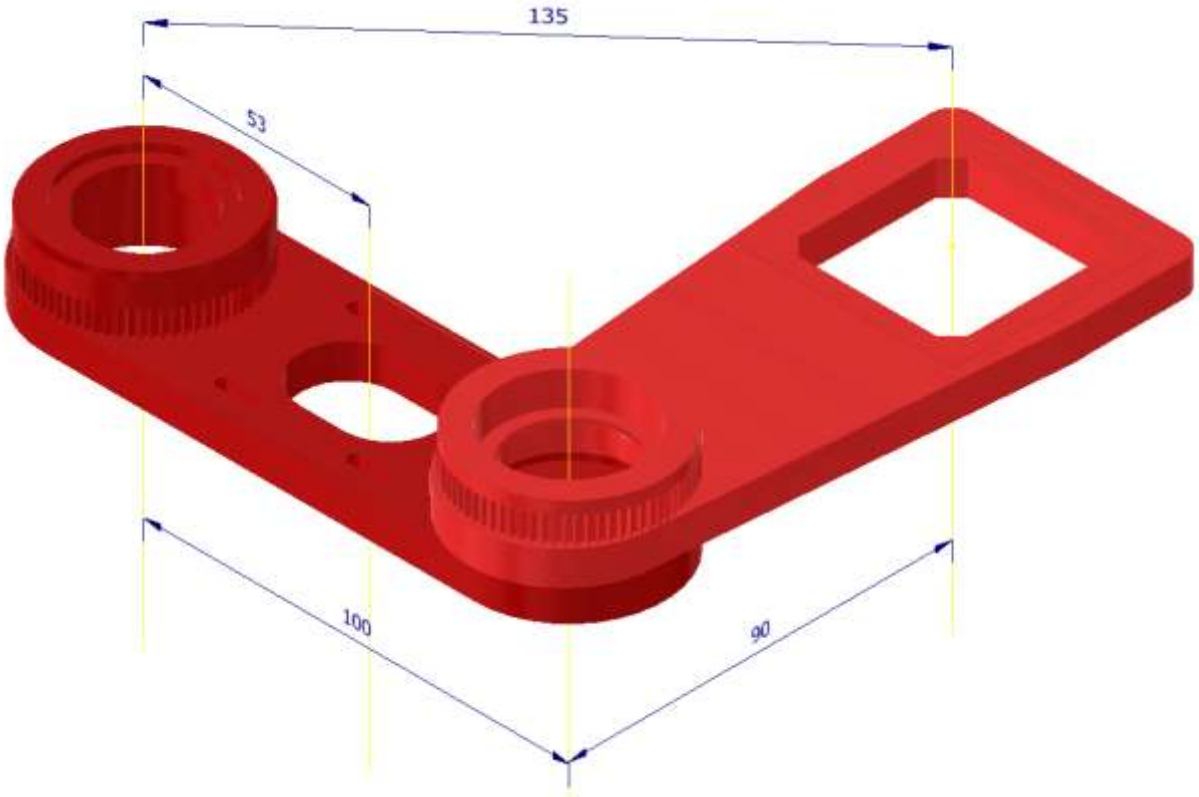
Freccia max = 0.48 mm

CALCOLO SFORZI E DEFORMAZIONE SUL MODELLO EFFETTIVO IN ABS NELLA POSIZIONE DISTESA

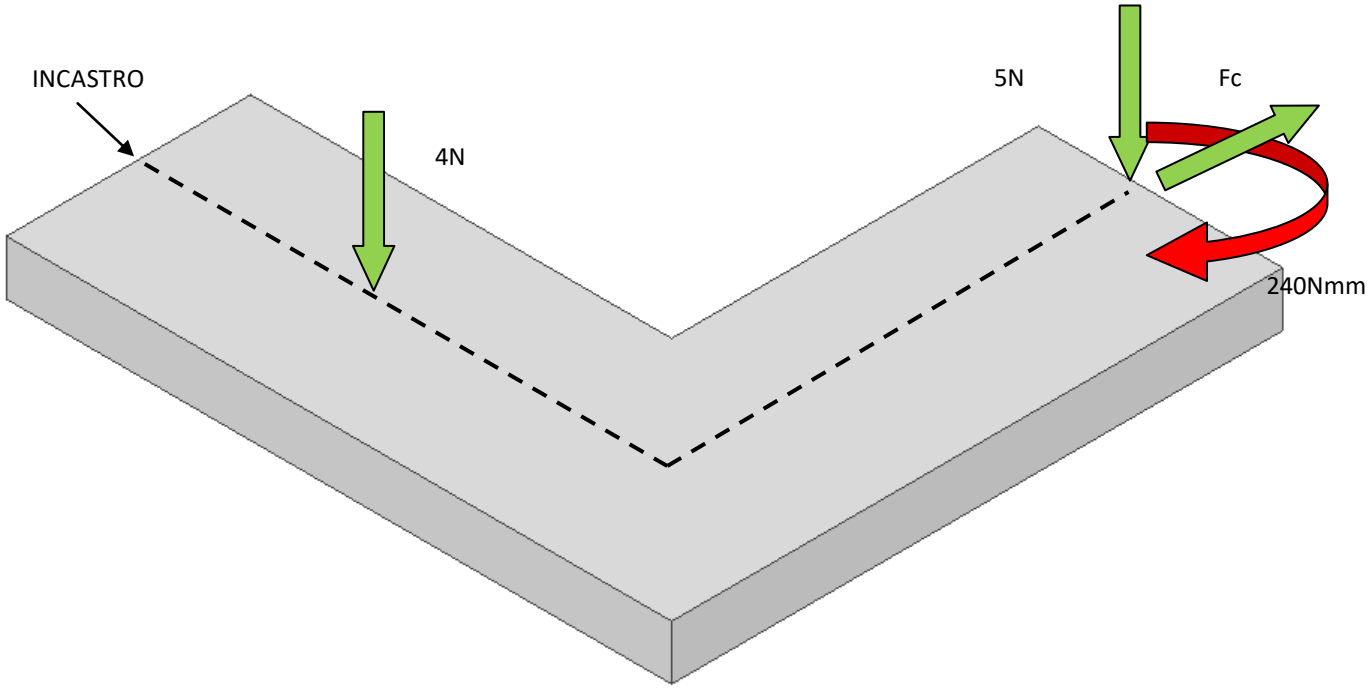




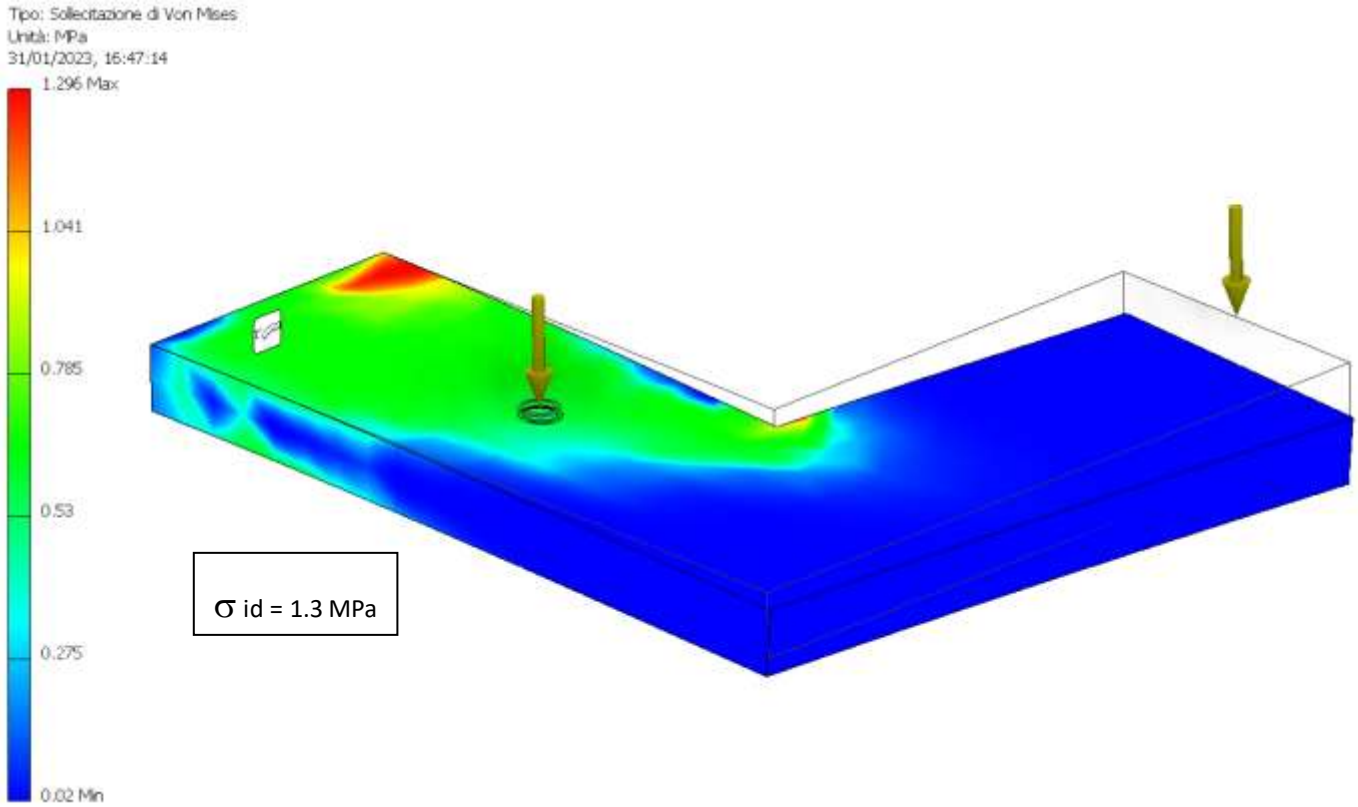
SOLLECITAZIONI SUI LINK DEL ROBOT PLANARE NELLA POSIZIONE AD ANGOLO RETTO

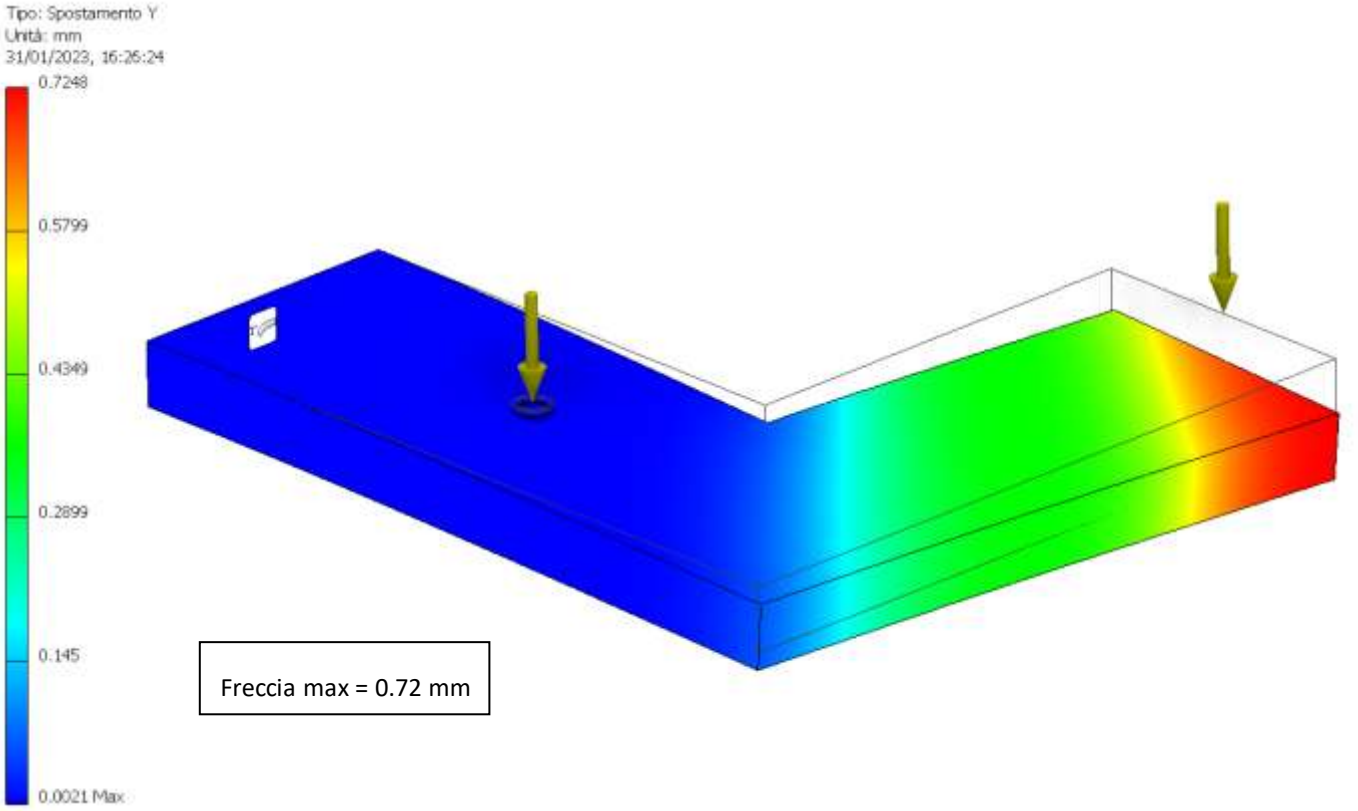


In questa posizione la forza di taglio da 5N genera anche un momento torcente nella sezione incastrata.



Determinare la σ id .





In questa posizione la presenza di un momento torcente risulta comunque meno gravosa del momento flettente maggiore che si ha nella posizione distesa.

CINEMATICA DEL ROBOT

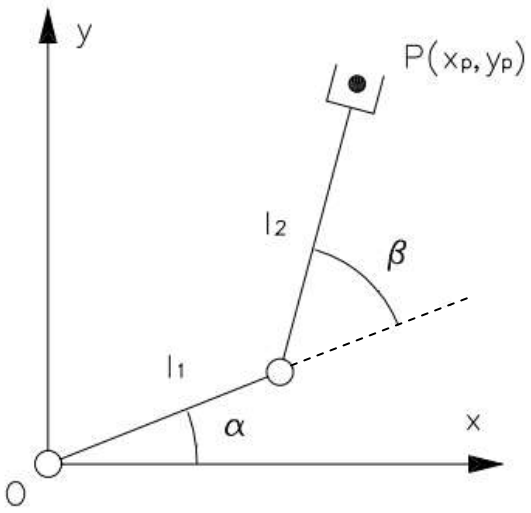
La cinematica del robot è lo studio del suo movimento prescindendo dalle cause che lo generano.

Il robot viene visto come una catena di corpi rigidi, dalla base all'end effector, connessi da giunti che consentono un singolo grado di libertà. La conoscenza del modello cinematico del robot è essenziale nei problemi di pianificazione del moto e controllo

CINEMATICA DIRETTA DEL ROBOT PLANARE A 2 LINK

La cinematica diretta, noti gli angoli dei link 1 e link 2, permette di ricavare la posizione finale P(xp,yp) della pinza.

Gli angoli si misurano come indicato in figura e sono positive se in senso antiorario e neagativi in senso orario.



Da semplici considerazioni geometriche:

$$\begin{cases} x_p = l_1 \cos\alpha + l_2 \cos(\alpha+\beta) \\ y_p = l_1 \sin\alpha + l_2 \sin(\alpha+\beta) \end{cases}$$

FOGLIO DI CALCOLO

Il problema presenta due possibili soluzioni dette a "gomito alto" e a "gomito basso".

x =$C$4*COS(RADIANTI($C$7))+ $C$5*COS(RADIANTI($C$7+$C$8))
y =$C$4*SEN(RADIANTI($C$7))+ $C$5*SEN(RADIANTI($C$7+$C$8))

CINEMATICA DIRETTA

Noti angoli dei giunti trovare la posizione pinza (x,y)

l1= 100 mm

l2= 100 mm

Angoli dei giunti

θ1	**60 °**	**30 °**
θ2	**-30 °**	**30 °**

gomito alto		
x	136,60	mm
y	136,60	mm

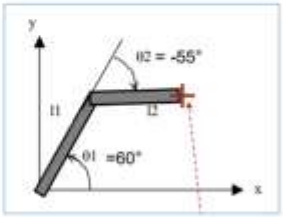
gomito basso		
x	136,60	mm
y	136,60	mm

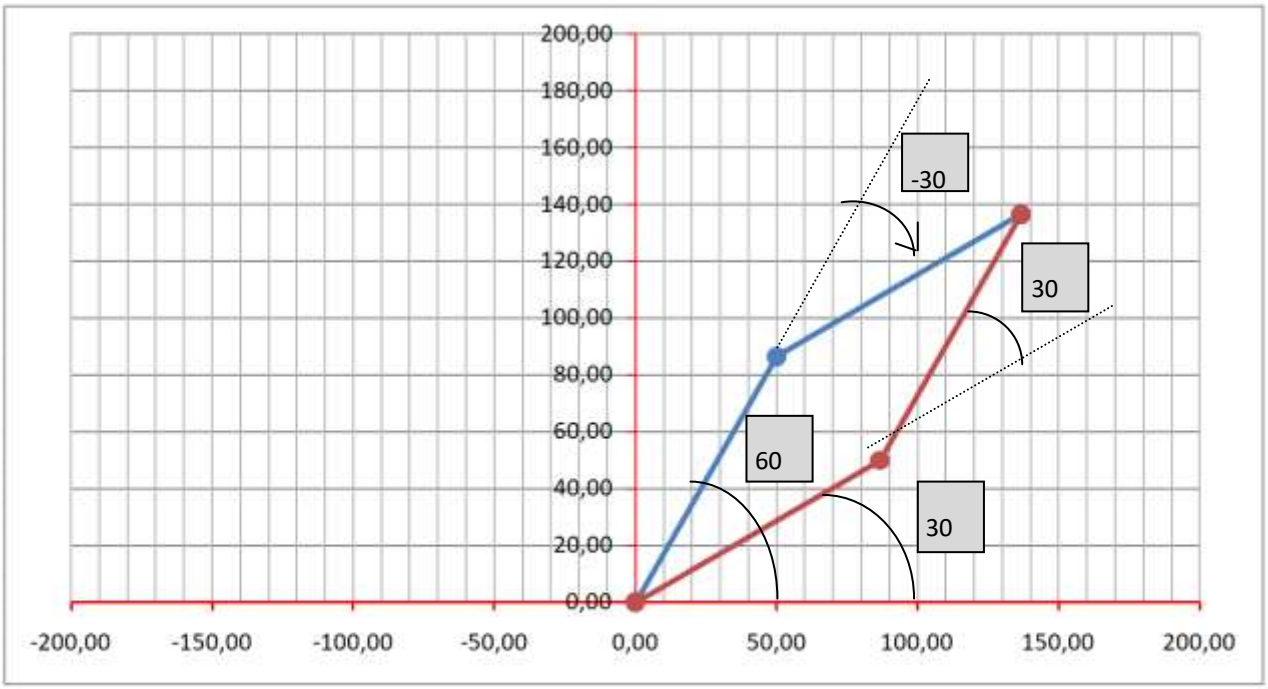
gomito alto

Link	x	y
0	0,00	0,00
1	50,00	86,60
2	136,60	136,60

gomito basso

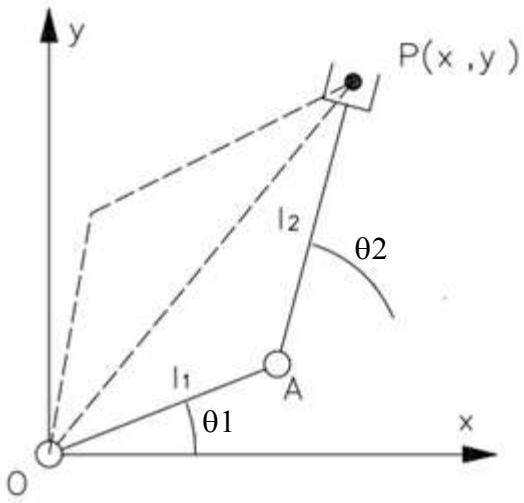
Link	x	y
0	0,00	0,00
1	86,60	50,00
2	136,60	136,60





CINEMATICA INVERSA DEL ROBOT DEL ROBOT PLANARE

Nota la posizione P(xp,yp) che si vuole raggiungere si devono ricavare gli angoli necessari.



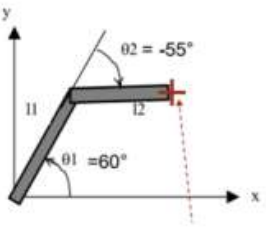
$$\vartheta 2 = \arccos\left[\frac{\left(x^2 + y^2 - l_1^{\,2} - l_2^{\,2}\right)}{2 \cdot l_1 \cdot l_2}\right]$$

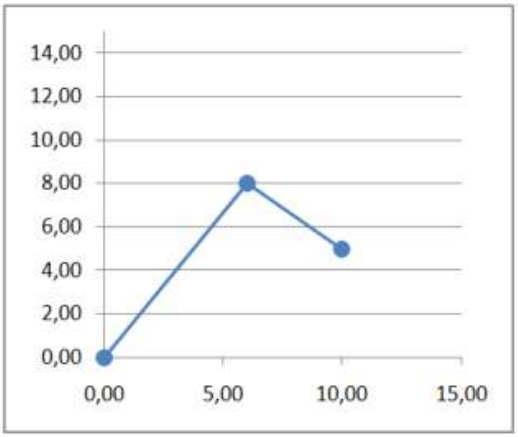
$$\vartheta 1 = \tan^{-1}\left(\frac{y}{x}\right) - \tan^{-1}\left(\frac{l_2 \cdot S2}{l_1 + l_2 \cdot C2}\right)$$

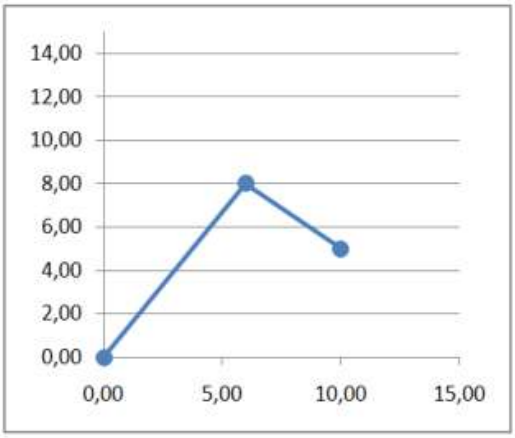
S2 = sen θ2 C2 = cos θ2

FOGLIO DI CALCOLO

	A	B	C	D	E	F
1	**CINEMATICA DIRETTA**					
2	Noti gli angoli dei giunti trovare la posizione del polso (x,y)					
3	l1=	10 CM				
4	l2=	5 CM		ALTO	x	y
5				0	0,00	0,00
6	θ1	53,1 °		1	6,00	8,00
7	θ2	-90 °		2	10,00	4,99
8						
9	x	10,00				
10	y	5,0				
11						
12						
13						
14						
15						
16	**CINEMATICA INVERSA**					
17	Nota la posizione (x,y) del polso trovare gli angoli dei giunti					
18	l1	10 cm				
19	l2	5 cm		ALTO	x	y
20				0	0,00	0,00
21	x	10,00 cm		1	6,00	8,00
22	y	5,00 cm		2	10,00	5,00
23						
24	ANGOLI GOMITO ALTO					
25	θ2	-90 °				
26	θ1	53,1 °				
27						
28						
29	Formule Excel					
30	θ2	=GRADI(-ARCCOS(($B$21^2+$B$22^2-$B$18^2-$B$19^2)/(2*$B$19*$B$18)))				
31	θ1	=GRADI(ARCTAN($B$22/$B$21))-GRADI(ARCTAN($B$19*SEN(RADIANTI($B$25))/($B$18+$B$19*COS(RADIANTI($B$25)))))				







Cinematica Inversa Robot Planare

l1 =	100,0	mm
l2 =	90,0	mm
xA =	190,0	
yA =	0,0	

l1 =	100,0	mm
l2 =	90,0	mm
xB =	120,0	
yB =	30,0	

l1 =	100,0	mm
l2 =	90,0	mm
xC =	100,0	
yC =	100,0	

cosθ2 =	1,0
θ2 =	0,0
senθ2 =	0,0
θ1 =	0,0
x1 =	100,0
y1 =	0,0

cosθ2 =	-0,156
θ2 =	-98,9
senθ2 =	-0,988
θ1 =	60,0
x1 =	50,0
y1 =	86,6

cosθ2 =	0,106
θ2 =	-83,9
senθ2 =	-0,994
θ1 =	84,3
x1 =	10,0
y1 =	99,5

A	x	y
0	0,0	0,0
1	100,0	0,0
2	190,0	0,0

B	x	y
0	0,0	0,0
1	50,0	86,6
2	120,0	30,0

C	x	y
0	0,0	0,0
1	10,0	99,5
2	100,0	100,0

A Gomito alto

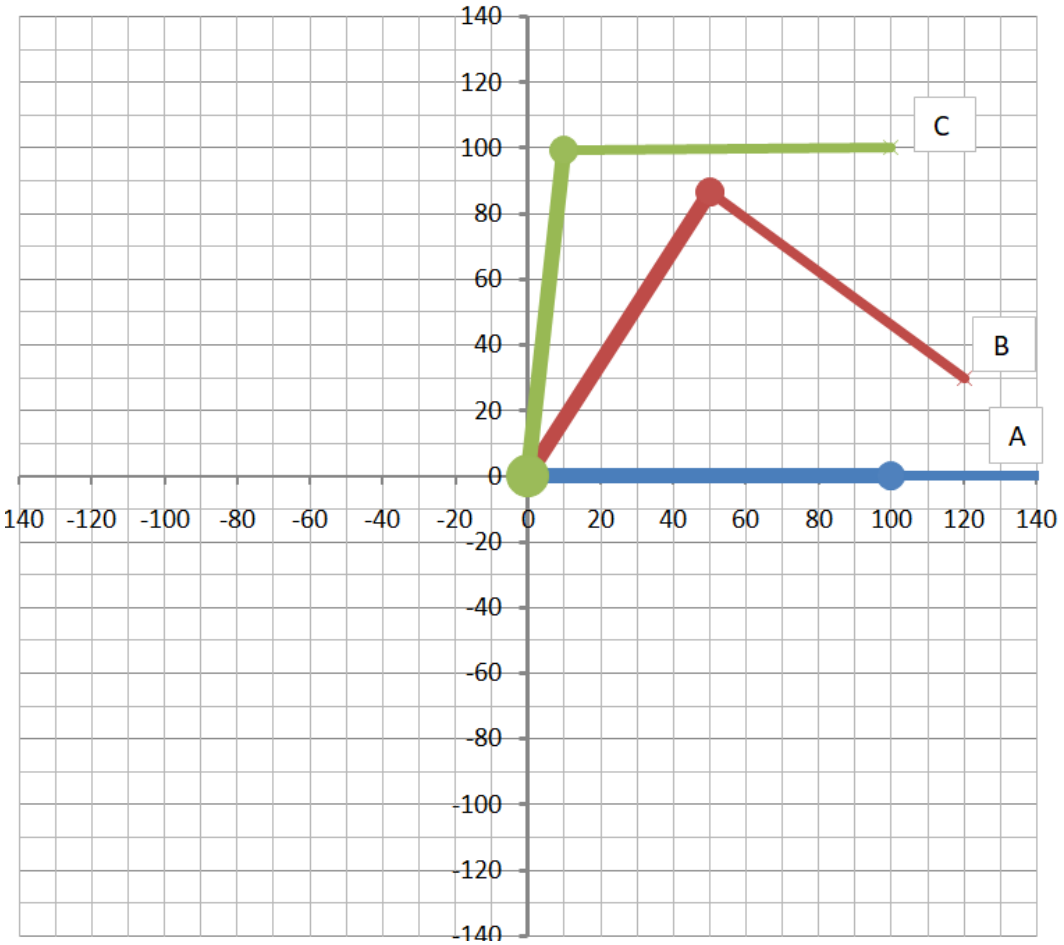
θ1	0,0
θ2	0,0

B Gomito alto

θ1	60,0
θ2	-98,9

C Gomito alto

θ1	84,3
θ2	-83,9



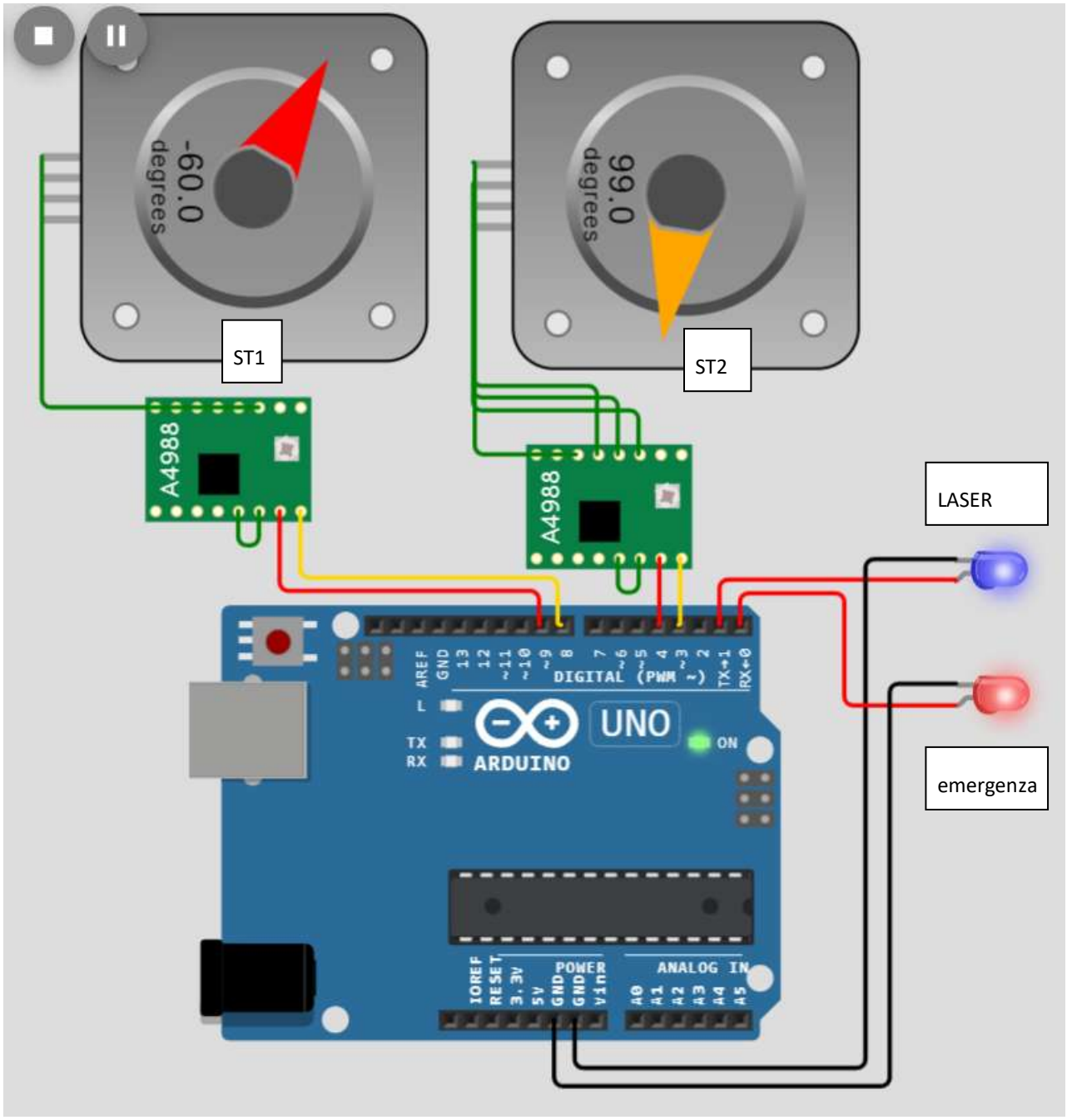
ESERCIZIO TAGLIO LASER SCARA 2 ASSI

Simulare un TAGLIO LASER SCARA dotato di 2 motori stepper.

- ST1 → movimento angolare link1 (motore con rapporto riduzione 1.8:1 → 360 step per giro)
- ST2 → movimento angolare link2 (motore con rapporto riduzione 1.8:1 → 360 step per giro)

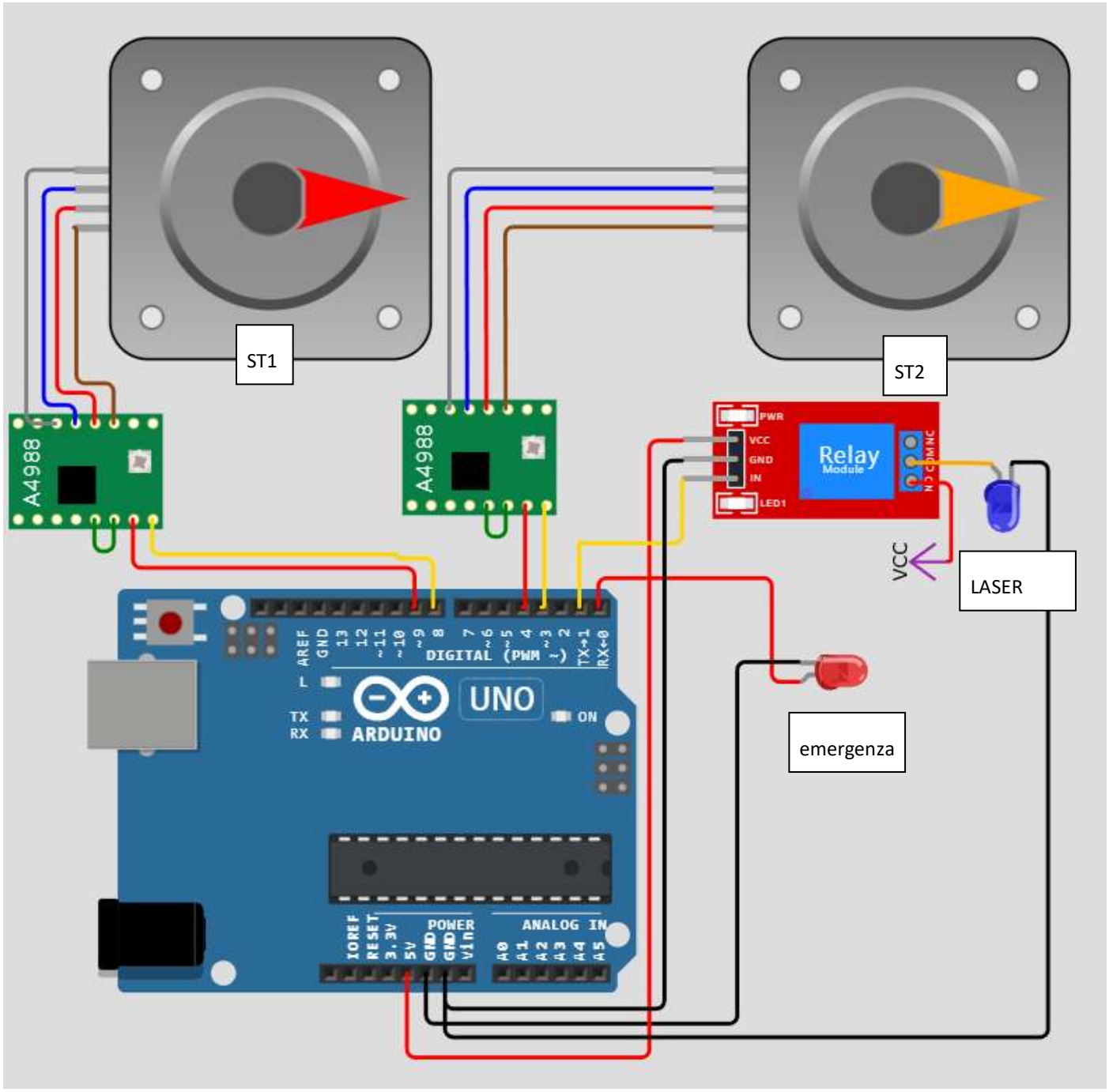
Il robot, partendo dalla posizione di riposo A deve raggiungere prima la posizione B e poi C e attivare il LASER per 1s.
Durante gli spostamenti deve essere accesa una lampadina di energenza a 12V – 150mA.

Al termine del ciclo si deve rientrare alla posizione di riposo A.



simulabile su "wokwi.com"

Schema con relè per attivare il laser



simulabile su "wokwi.com"

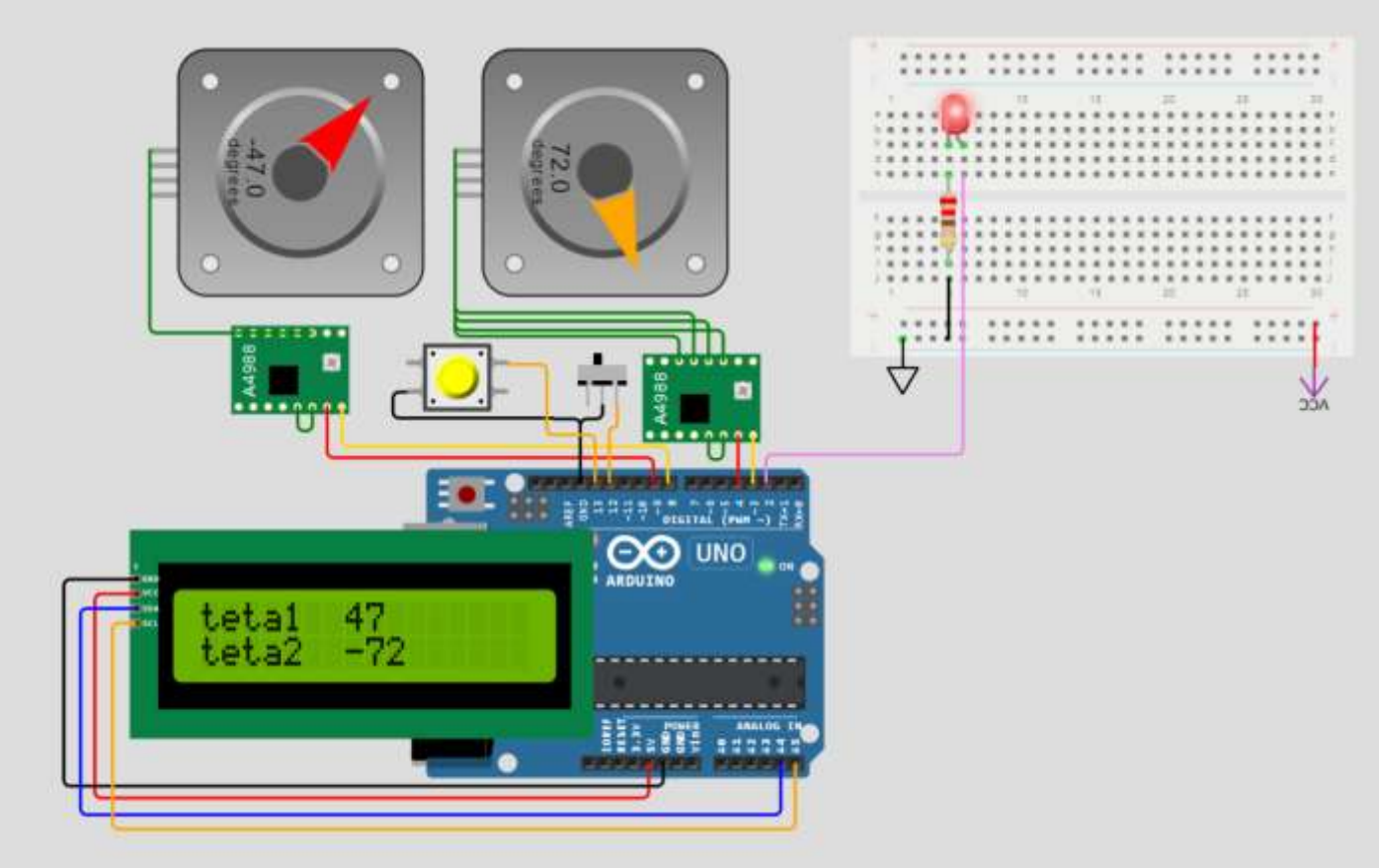
CODICE

```
// Nema 17 200 passi per giro accoppiato a riduttore 1.8:!
// "arrow": "orange", "display": "angle", "gearRatio": "1.8:1"
#define DIR_PIN1  8
#define STEP_PIN1 9   // set gearRatio 1.8:1 to get 1° for 1 step  (1.8=360/200)
#define DIR_PIN2  3
#define STEP_PIN2 4  // set gearRatio 1.8:1 to get 1° for 1 step  (1.8=360/200)
#define DELAY_ST 10000  // 2000 micros

#define LASER_PIN  1
#define LED_PIN  0

int idMotor;  // 1,2 ...
```

```
void setup() {
  pinMode(DIR_PIN1, OUTPUT);  pinMode(STEP_PIN1, OUTPUT);
  pinMode(DIR_PIN2, OUTPUT);  pinMode(STEP_PIN2, OUTPUT);
  pinMode(LASER_PIN, OUTPUT);
  pinMode(LED_PIN, OUTPUT);
  delay(1000);
}

void loop() {
  attivaEmergenza(HIGH);
  // B  60 antior ; 99 orario
  attivaStepper(1,60,LOW); attivaStepper(2,99,HIGH); delay(500);
  attivaLaser(1000);
  delay(3000);

  // C  84 antior; 84 antior
  attivaStepper(1,abs(84-60),LOW); attivaStepper(2,abs(84-99),LOW);   delay(500);
  attivaLaser(1000);
  delay(3000);

  // A  0 orario; 0 antior
  attivaStepper(1,84,HIGH); attivaStepper(2,84,LOW);
  attivaEmergenza(LOW);
  delay(3000);
}

// orientation --> LOW=antiorario; HIGH=orario
void attivaStepper(int id, int postion, int orientation) {
  // stepper 1
  if (id==1) {
    digitalWrite(DIR_PIN1, orientation);
    for (int i = 0; i < postion; i++) {
      digitalWrite(STEP_PIN1, HIGH); delayMicroseconds(DELAY_ST);
      digitalWrite(STEP_PIN1, LOW);  delayMicroseconds(DELAY_ST);
    }
  }
  // stepper 2
  else if (id==2) {
    digitalWrite(DIR_PIN2, orientation);
    for (int i = 0; i < postion; i++) {
      digitalWrite(STEP_PIN2, HIGH); delayMicroseconds(DELAY_ST);
      digitalWrite(STEP_PIN2, LOW);  delayMicroseconds(DELAY_ST);
    }
  }

}

void attivaLaser(int sec) {
  digitalWrite(LASER_PIN, HIGH);  delay(2000); digitalWrite(LASER_PIN, LOW);
}

void attivaEmergenza(int stato) {
   if (stato==HIGH) { digitalWrite(LED_PIN, HIGH);}
   if (stato==LOW) { digitalWrite(LED_PIN, LOW);}
}
```

ESERCIZIO TAGLIO LASER SCARA 2 ASSI CON EMERGENZA E RESET

Il laser deve raggiungere le tre poszioni B,C,D assegnate.
Se viene attivata l'emergenza (slider) il laser si deve FERMARE.
Per ripartire deve essere sbloccata l'emergenza e poi premuto il reset (push button giallo).



FOGLIO DI CALCOLO

Cinematica Inversa Robot Planare

l1 =	100,0	mm	l1 =	100,0	mm	l1 =	100,0	mm	l1 =	100,0	mm
l2 =	80,0	mm	l2 =	80,0	mm	l2 =	80,0	mm	l2 =	80,0	mm
xA =	180,0		xB =	140,0		xC =	100,0		xC =	60,0	
yA =	0,0		yB =	40,0		yC =	140,0		yC =	160,0	
cosθ2 =	1,0		cosθ2 =	0,300		cosθ2 =	0,825		cosθ2 =	0,800	
θ2 =	0,0		θ2 =	-72,5		θ2 =	-34,4		θ2 =	-36,9	
senθ2 =	0,0		senθ2 =	-0,954		senθ2 =	-0,565		senθ2 =	-0,600	
θ1 =	0,0		θ1 =	47,6		θ1 =	69,7		θ1 =	85,8	
x1 =	100,0		x1 =	67,5		x1 =	34,7		x1 =	7,4	
y1 =	0,0		y1 =	73,8		y1 =	93,8		y1 =	99,7	

A	x	y
0	0,0	0,0
1	100,0	0,0
2	180,0	0,0

B	x	y
0	0,0	0,0
1	67,5	73,8
2	140,0	40,0

C	x	y
0	0,0	0,0
1	34,7	93,8
2	100,0	140,0

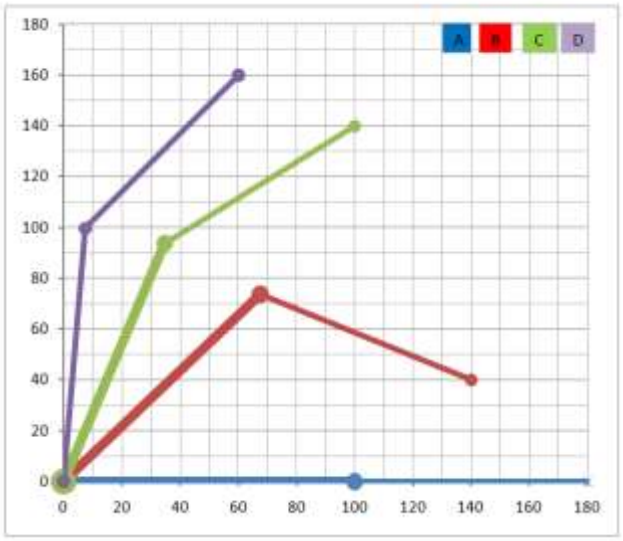
D	x	y
0	0,0	0,0
1	7,4	99,7
2	60,0	160,0

A Gomito alto	
θ1	0,0
θ2	0,0

B Gomito alto	
θ1	47,6
θ2	-72,5

C Gomito alto	
θ1	69,7
θ2	-34,4

C Gomito alto	
θ1	85,8
θ2	-36,9



CODICE

```
#include <LiquidCrystal_I2C.h>
#include <Servo.h>
LiquidCrystal_I2C lcd(0x27, 20, 4);

// Nema 17 200 passi per giro accoppiato a
// vite T8 Trapezoidale Senza Fine Ø8 Mm Pitch 2mm 1 Principio
#define DIR_PIN1  8
#define STEP_PIN1 9   // set gearRatio 1.8:1 to get 1° for 1 step  (1.8=360/200)
#define DIR_PIN2  3
#define STEP_PIN2 4  // set gearRatio 1.8:1 to get 1° for 1 step  (1.8=360/200)
#define DELAY_ST 15000  // 2000 micros

#define LED_PIN  2
#define POLSO_PIN  5
#define PINZA_PIN  6
#define RESET_PIN 13
#define EM_PIN 12

int idMotor;  // 1,2 ...
int statoEmergenza;
int segno,teta1, teta2;

void setup() {
  pinMode(DIR_PIN1, OUTPUT);
  pinMode(STEP_PIN1, OUTPUT);
  pinMode(DIR_PIN2, OUTPUT);
  pinMode(STEP_PIN2, OUTPUT);
  pinMode(EM_PIN, INPUT_PULLUP);
  pinMode(RESET_PIN, INPUT_PULLUP);
  pinMode(LED_PIN, OUTPUT);

  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0); lcd.print("teta1 ");
  lcd.setCursor(0, 1); lcd.print("teta2 ");

  // all'inizio devo garantire di essere in HOME
  teta1=0;
  teta2=0;
  statoEmergenza= LOW;

  Serial.begin(115200);
  delay(1000);
}

void loop() {
  lcd.setCursor(7, 1);

  // RESET PREMUTO?
  if (digitalRead(RESET_PIN) == LOW) {
    statoEmergenza= LOW;
    // HOME
    muoviLink1(-teta1);
    muoviLink2(-teta2);
    attivaLampada(LOW);
    delay(1000);
  }

  // EMERGENZA PREMUTA?
  if (digitalRead(EM_PIN) == LOW ) { statoEmergenza=HIGH; attivaLampada(HIGH); }

  if (statoEmergenza==LOW)  {
    muoviLink(1,47);
```

```
    delay(1000);
    muoviLink(2,-72);
    delay(1000);

    muoviLink(1,70-teta1);
    delay(1000);
    muoviLink(2,-34-teta2);
    delay(1000);

    muoviLink(1,86-teta1);
    delay(1000);
    muoviLink(2,-40-teta2);
    delay(1000);

    muoviLink(1,-teta1);
    delay(1000);
    muoviLink(2,-teta2);
    delay(1000);
  }
}

// +antiorario; - orario
void muoviLink(int link, int angolo) {
    int position;
    int orientation;
    position= abs(angolo);
    if (angolo>=0) {segno=1; orientation=LOW;}
    else {segno=-1;orientation=HIGH;}

    if (link==1) {
      digitalWrite(DIR_PIN1, orientation);
      for (int i = 0; i < position; i++) {
        if (digitalRead(EM_PIN) == HIGH && statoEmergenza==LOW) {
          statoEmergenza=0;
          teta1= teta1 + segno;
          digitalWrite(STEP_PIN1, HIGH); delayMicroseconds(DELAY_ST);
          digitalWrite(STEP_PIN1, LOW);  delayMicroseconds(DELAY_ST);
          lcd.setCursor(7, 0); lcd.print(teta1); Serial.println(teta1);
        }
        else { statoEmergenza=HIGH; break;}
      }
    }
    if (link==2) {
      digitalWrite(DIR_PIN2, orientation);
      for (int i = 0; i < position; i++) {
        if (digitalRead(EM_PIN) == HIGH && statoEmergenza==LOW) {
          statoEmergenza=0;
          teta2= teta2 + segno;
          digitalWrite(STEP_PIN2, HIGH); delayMicroseconds(DELAY_ST);
          digitalWrite(STEP_PIN2, LOW);  delayMicroseconds(DELAY_ST);
          lcd.setCursor(7, 1); lcd.print(teta2); Serial.println(teta2);
        }
        else { statoEmergenza=HIGH; break;}
      }
    }
}

void attivaLampada(int stato) {
   if (stato==HIGH) { digitalWrite(LED_PIN, HIGH);}
   if (stato==LOW) { digitalWrite(LED_PIN, LOW);}
}
```

ESERCIZIO TAGLIO LASER SCARA 3 ASSI

Simulare un TAGLIO LASER SCARA dotato di 3 motori stepper.

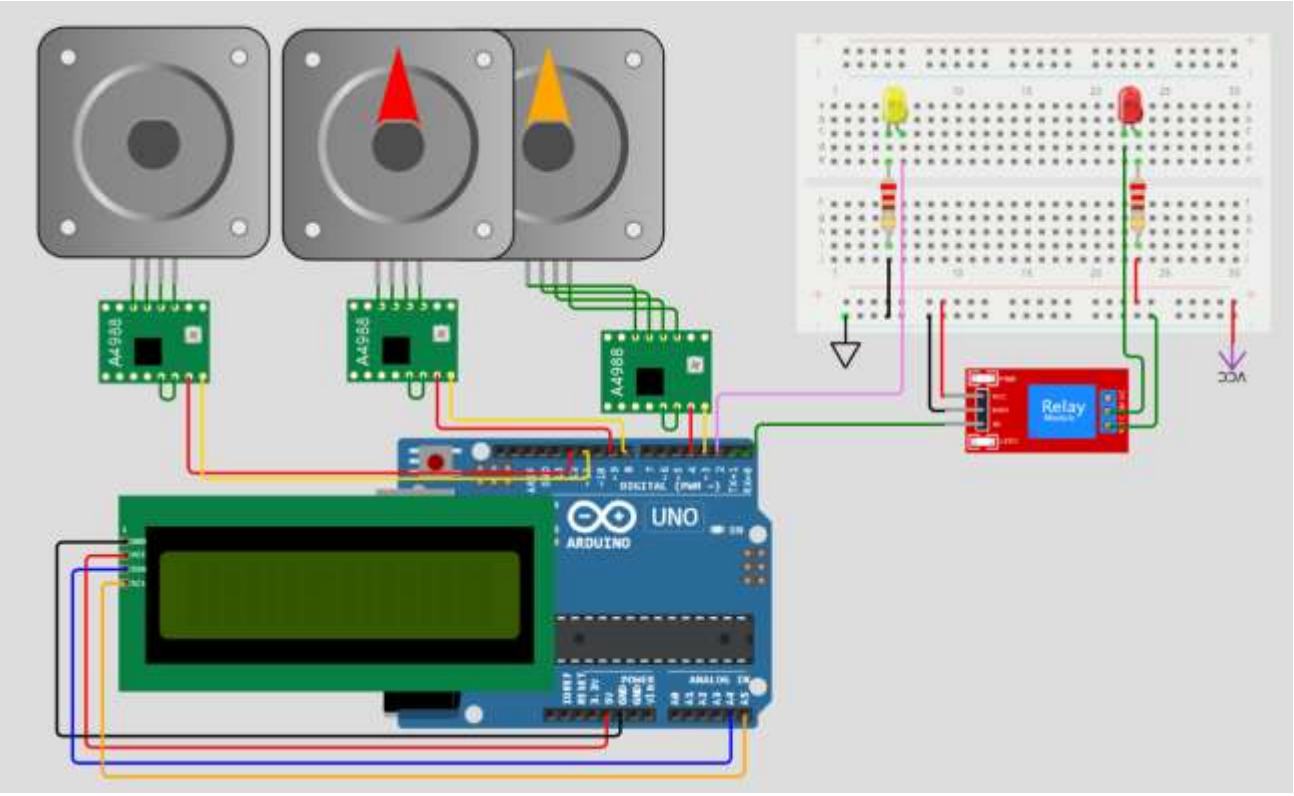
- ST0 → movimento verticale (asse Z) con vite T8 Trapezoidale Senza Fine Ø8 Mm Pitch 2mm 1 Principio
- ST1 → movimento angolare link1 (motore con rapporto riduzione 1.8:1 → 360 step per giro)
- ST2 → movimento angolare link2 (motore con rapporto riduzione 1.8:1 → 360 step per giro)

Il robot deve raggiungere la posizione P(X,Y,Z) e attivare il LASER per 1s.

Il laser viene attivato emndiante un rele'.

Quando il laser è attivo viene acceso un led giallo.

Al termine il robot torna alla posizone di riposo iniziale.



simulabile su "wokwi.com"

CODICE

```
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 20, 4);

// Nema 17 200 passi per giro accoppiato a
// vite T8 Trapezoidale Senza Fine Ø8 Mm Pitch 2mm 1 Principio
#define DIR_PIN0  11
#define STEP_PIN0 12   // gearRatio 1:1 --> 1 giro= 200 step --> 2mm di spostamento
#define DIR_PIN1  8
#define STEP_PIN1 9   // set gearRatio 1.8:1 to get 1° for 1 step  (1.8=360/200)
#define DIR_PIN2  3
#define STEP_PIN2 4  // set gearRatio 1.8:1 to get 1° for 1 step  (1.8=360/200)
#define DELAY_ST 2000  // 2000 micros
#define DELAY_STZ 1000  // 2000 micros

#define LASER_PIN  2
#define LASER_PIN_R  1

int idMotor;  // 1,2 ...

void setup() {
  pinMode(DIR_PIN0, OUTPUT);
  pinMode(STEP_PIN0, OUTPUT);
  pinMode(DIR_PIN1, OUTPUT);
  pinMode(STEP_PIN1, OUTPUT);
  pinMode(DIR_PIN2, OUTPUT);
  pinMode(STEP_PIN2, OUTPUT);
  pinMode(LASER_PIN, OUTPUT);
  pinMode(LASER_PIN_R, OUTPUT);

  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0); lcd.print("Z(mm)");
  lcd.setCursor(0, 1); lcd.print("v(mm/s)");
  delay(1000);
}

void loop() {

 lcd.setCursor(7, 1);
  // rotazione ORARIA 30°
  movelinkTo(1,30,HIGH);
  delay(1000);

  movelinkTo(2,90,HIGH);
  delay(1000);

  movezTo(30,LOW);
  delay(1000);

  digitalWrite(LASER_PIN, HIGH);
  digitalWrite(LASER_PIN_R, HIGH);
  delay(2000);
  digitalWrite(LASER_PIN, LOW);
  digitalWrite(LASER_PIN_R, LOW);

  movelinkTo(2,90,LOW);
  delay(1000);

  movelinkTo(1,30,LOW);
  delay(1000);

  movezTo(30,HIGH);
  delay(1000);

}

void movelinkTo(int id, int postion, int orientation) {
  if (id==1) {
    digitalWrite(DIR_PIN1, orientation);
    for (int i = 0; i < postion; i++) {
      digitalWrite(STEP_PIN1, HIGH); delayMicroseconds(DELAY_ST);
      digitalWrite(STEP_PIN1, LOW);  delayMicroseconds(DELAY_ST);
```

```
    }
  }
  else if (id==2) {
    digitalWrite(DIR_PIN2, orientation);
    for (int i = 0; i < postion; i++) {
      digitalWrite(STEP_PIN2, HIGH); delayMicroseconds(DELAY_ST);
      digitalWrite(STEP_PIN2, LOW);  delayMicroseconds(DELAY_ST);
    }
  }

}

// spostamento relativo in mm
void movezTo(int z, int orientation) {
    int t0= millis();
    int t= millis();

    // calcolo numero di step necessari (1mm=100 step)
    int step = z * 100;
    digitalWrite(DIR_PIN0, orientation);
    for (int i = 0; i < step; i++) {
       /* questo codice rallenta velocità motore dopo 2-3 volte !!!!
       float cz= (i+1)/100;
       if ( (millis() - t) >= 500) {
         t = millis();
         if (orientation==LOW) {lcd.setCursor(6, 0); lcd.print("-");}
         else {lcd.setCursor(6, 0); lcd.print("+");}
         lcd.setCursor(7, 0); lcd.print(cz);

         // calcolo rpm  -> 1 giro = 200 step
        float vel = 1000 * cz / (millis() - t0);
        lcd.setCursor(8, 1); lcd.print(vel);

       }
       */
       digitalWrite(STEP_PIN0, HIGH); delayMicroseconds(DELAY_STZ);
       digitalWrite(STEP_PIN0, LOW);  delayMicroseconds(DELAY_STZ);
    }

    float cz= step/100;
    if (orientation==LOW) {lcd.setCursor(6, 0); lcd.print("-");}
    else {lcd.setCursor(6, 0); lcd.print("+");}
    lcd.setCursor(7, 0); lcd.print(cz);

    float vel = 1000* cz / (millis() - t0);
    lcd.setCursor(8, 1); lcd.print(vel);
}
```

ROBOT SCARA

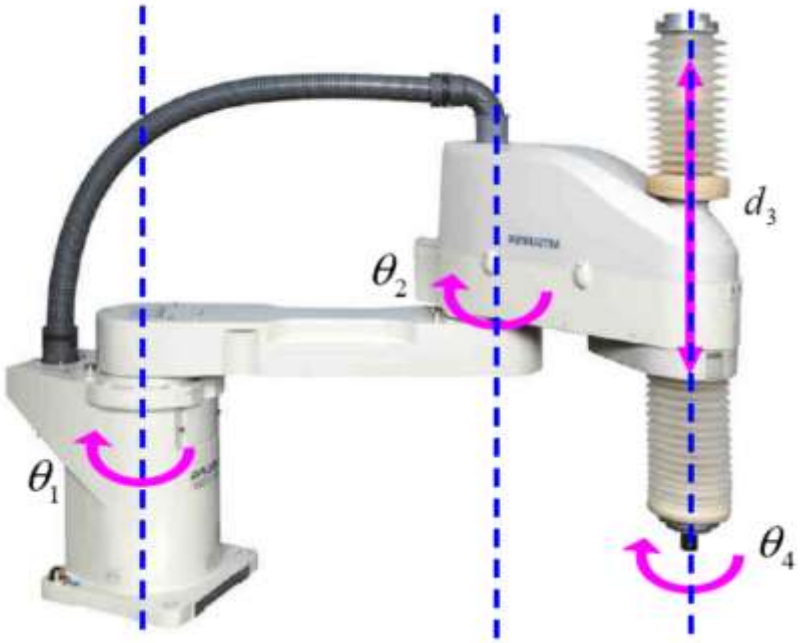
Il robot SCARA (Selective Compliance Assembly Robot Arm) è stato concepito per operazioni veloci e precise.
La cinematica del robot SCARA è stata sviluppata all'inizio degli anni '70 in seguito all'osservazione secondo la quale i cicli di movimento più frequenti sono realizzabili con 4 assi.

Il vantaggio che presenta questo tipo di robot rispetto ad altri è dovuto al fatto che per sollevare un pezzo il movimento avviene su un solo asse. Il che ne semplifica la struttura rendendolo più affidabile.

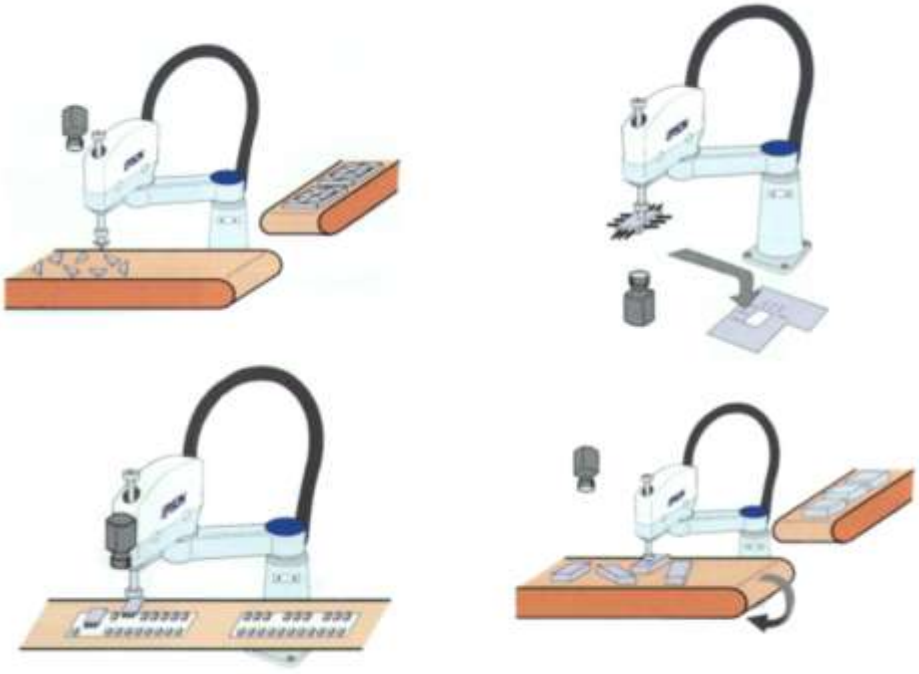
Perciò, laddove è possibile la movimentazione di parti su un livello, i vantaggi dello SCARA prevalgono sensibilmente rispetto a quelli delle altre cinematiche.

Il robot Scara presenta quindi 4 gradi di libertà. In un piano orizzontale si muovono 2 bracci articolati, incernierati ad una estremità con un asse verticale fisso, mentre all'altra estremità libera si trova 1 asse Z, il quale può muoversi sia verticalmente che ruotare intorno al proprio asse.

MOVIMENTI E ANGOLI DEL ROBOT SCARA

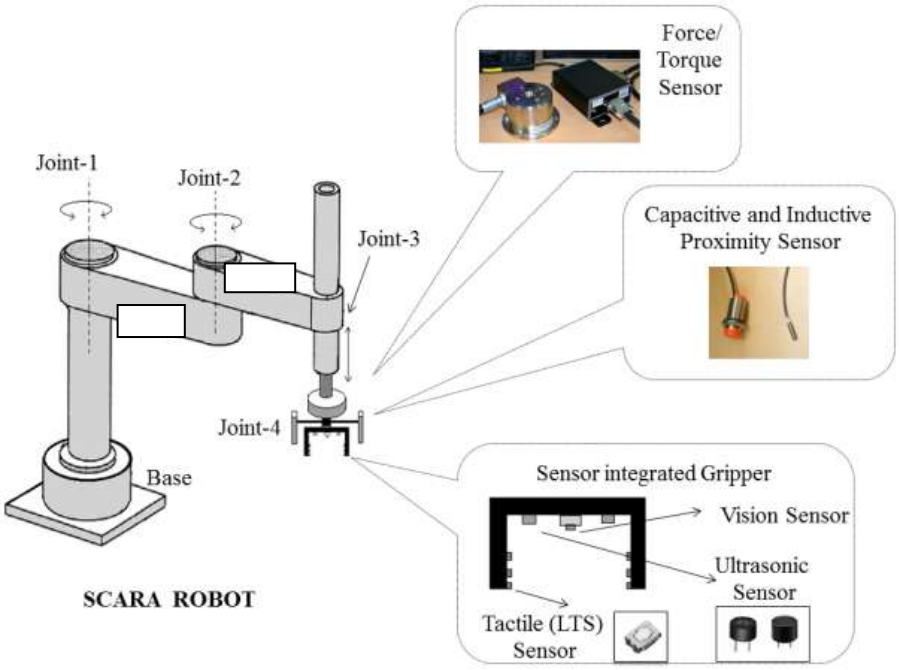


APPLICAZIONI TIPICHE DEL ROBOT SCARA



I robot SCARA offrono il massimo delle prestazioni di ripetibilità rispetto a tutti i tipi di robot.
Gli errori che si verificano nella posizione X-Y sono dovuti all'utilizzo di due motori in J1 e J2.
Gli altri tipi di robot utilizzano tre o più motori per contribuire alla posizione X-Y.
Il numero dei motori è direttamente proporzionale agli errori che potrebbero verificarsi.
L'eccellente ripetibilità è un elemento fondamentale per le piccole applicazioni di assemblaggio, in cui occorre rispettare tolleranze inferiori a diversi micron. Ad esempio, può trattarsi dell'inserimento dei connettori nelle schede elettroniche o dello spostamento di un ago in una piccola fessura per la distribuzione.
Uno SCARA può permettere raggi di azione da 100 mm a 1.200 mm, con capacità di carico pagante da 1 kg a 200 kg.

END EFFECTOR



ESERCIZIO ROBOT SCARA

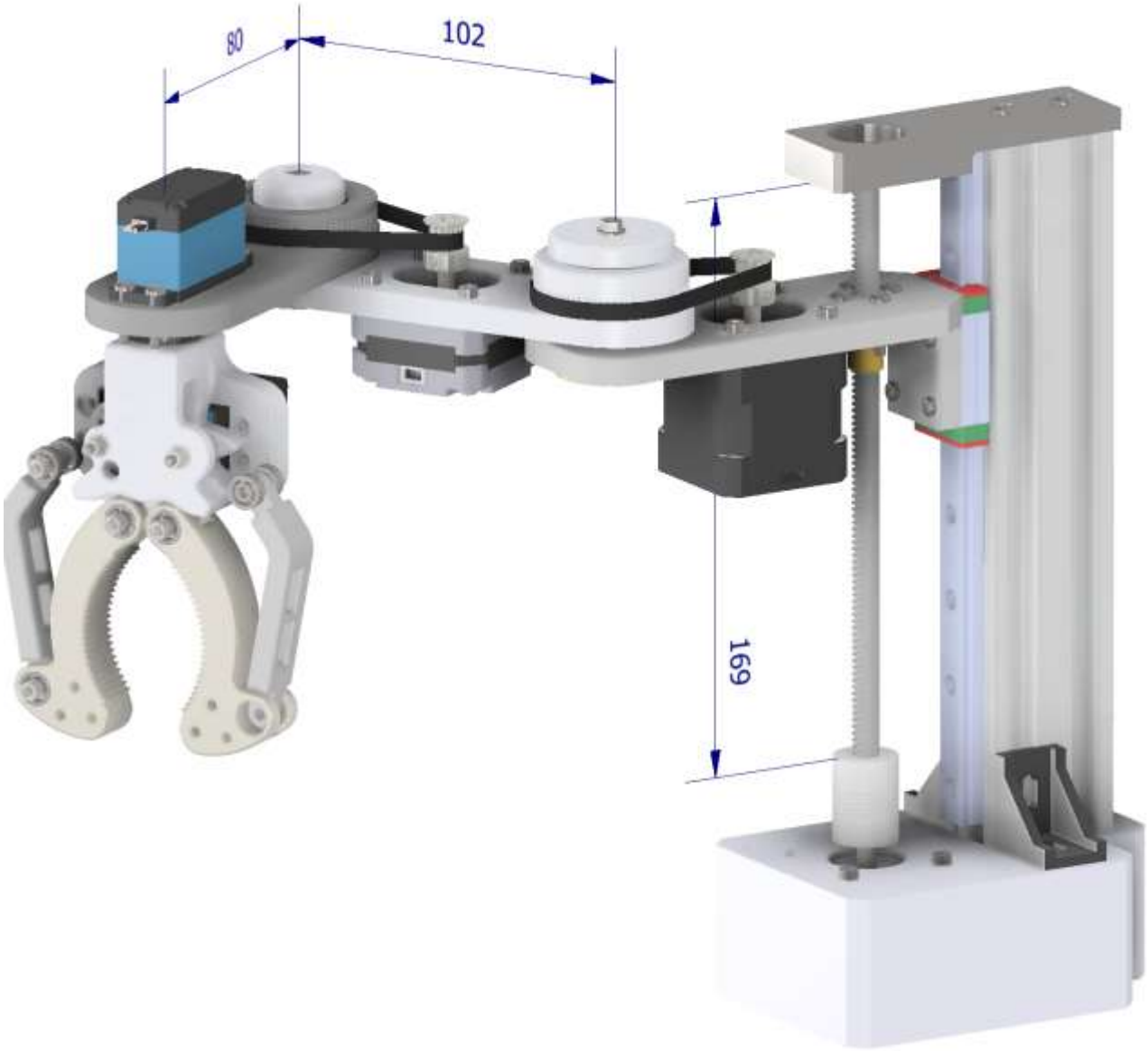
Simulare il movimento di un robot SCARA dotato di 3 motori stepper e 2 servomotori:

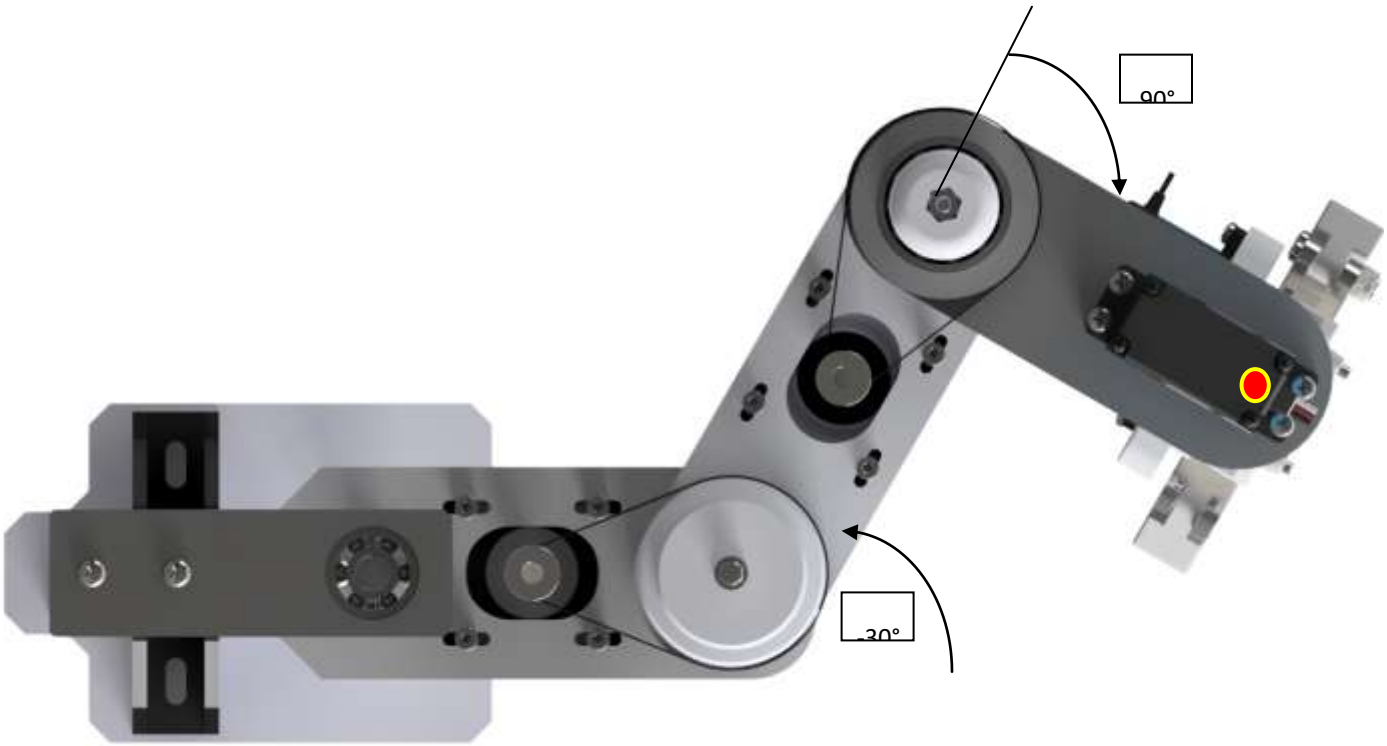
- ST0 → movimento verticale (asse Z) con vite T8 Trapezoidale Senza Fine Ø8 Mm Pitch 2mm 1 Principio
- ST1 → movimento angolare link1 (motore con rapporto riduzione 1.8:1 → 360 step per giro)
- ST2 → movimento angolare link2 (motore con rapporto riduzione 1.8:1 → 360 step per giro)
- SV1 →movimento polso pinza 0-180°
- SV2 →movimento griffe pinza 0°=aperta, 180°=chiusa

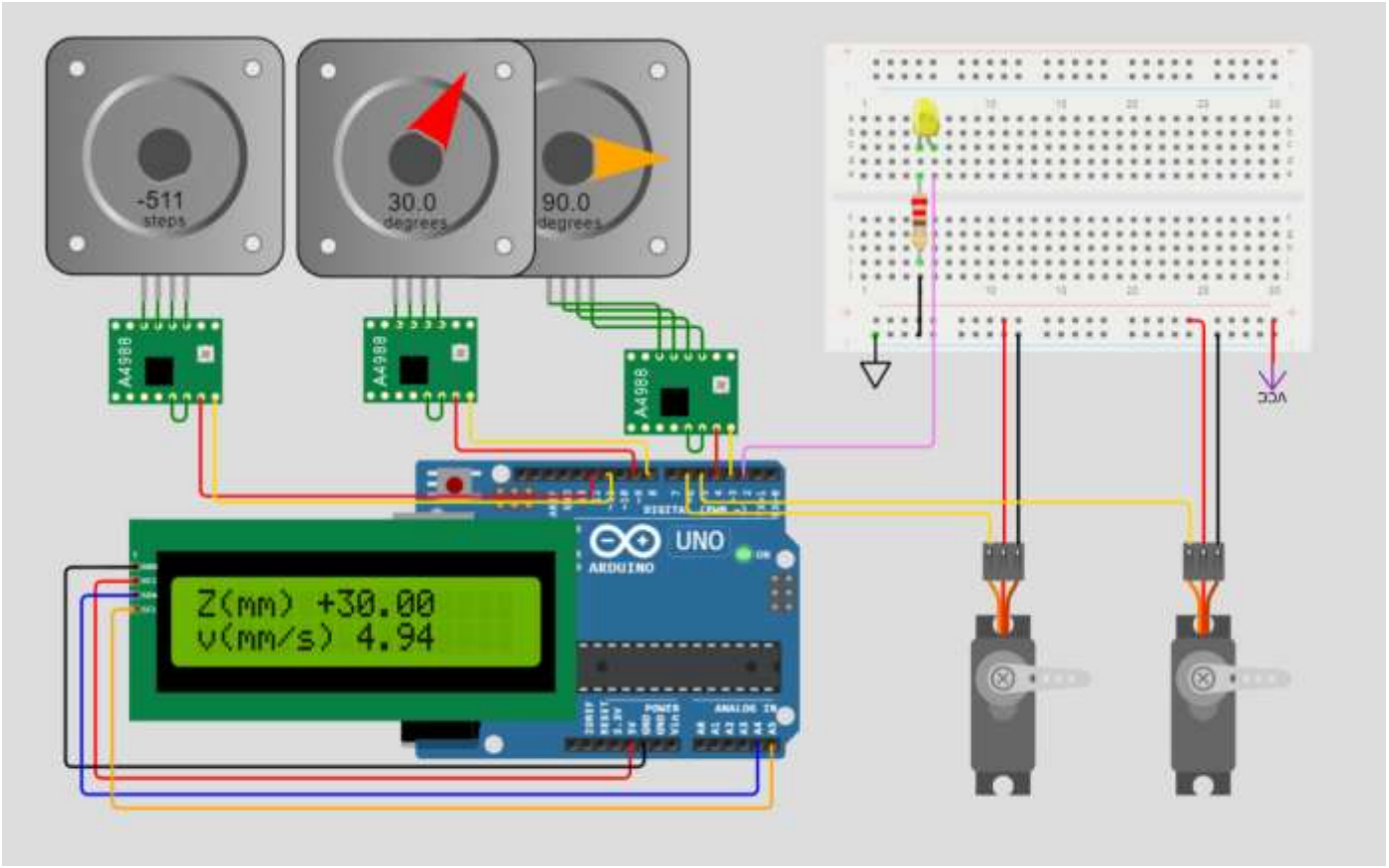
Link1 → 102mm
Link2 → 80

Posizione a riposo della pinza chiusa Z=120mm dal piano di appoggio.

Visualizzare la posizione verticale e la velocità di spostamento su un LCD 16x2 I2C.







simulabile su "wokwi.com"

CODICE

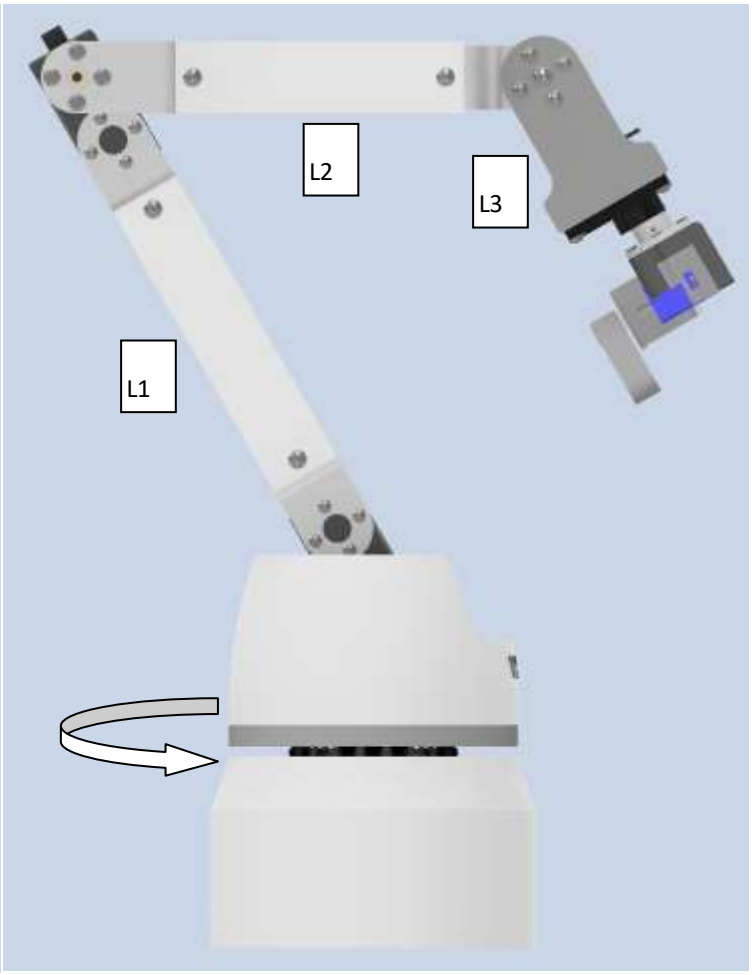
```
#include <LiquidCrystal_I2C.h>
#include <Servo.h>

LiquidCrystal_I2C lcd(0x27, 20, 4);
Servo servoPolso;  // create servo object to control a servo
Servo servoPinza;  // create servo object to control a servo

// Nema 17 200 passi per giro accoppiato a
// vite T8 Trapezoidale Senza Fine Ø8 Mm Pitch 2mm 1 Principio
#define DIR_PIN0  11
#define STEP_PIN0 12   // gearRatio 1:1 --> 1 giro= 200 step --> 2mm di spostamento
#define DIR_PIN1  8
#define STEP_PIN1 9   // set gearRatio 1.8:1 to get 1° for 1 step  (1.8=360/200)
#define DIR_PIN2  3
#define STEP_PIN2 4  // set gearRatio 1.8:1 to get 1° for 1 step  (1.8=360/200)
#define DELAY_ST 2000  // 2000 micros
#define DELAY_STZ 1000  // 2000 micros
#define LED_PIN  2
#define POLSO_PIN  5
#define PINZA_PIN  6
int idMotor;  // 1,2 ...

void setup() {
  pinMode(DIR_PIN0, OUTPUT);
  pinMode(STEP_PIN0, OUTPUT);
  pinMode(DIR_PIN1, OUTPUT);
  pinMode(STEP_PIN1, OUTPUT);
  pinMode(DIR_PIN2, OUTPUT);
  pinMode(STEP_PIN2, OUTPUT);
  pinMode(LED_PIN, OUTPUT);
  pinMode(POLSO_PIN, OUTPUT);
  pinMode(PINZA_PIN, OUTPUT);
  servoPolso.attach(POLSO_PIN); servoPolso.write(0);
  servoPinza.attach(PINZA_PIN); servoPinza.write(0);  // pinza aperta
  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0); lcd.print("Z(mm)");
  lcd.setCursor(0, 1); lcd.print("v(mm/s)");
  delay(1000);
}

void loop() {
 lcd.setCursor(7, 1);
  // rotazione ORARIA 30°
  movelinkTo(1,30,HIGH);
  delay(1000);
  movelinkTo(2,90,HIGH);
  delay(1000);
  movezTo(30,LOW);
  delay(1000);
  digitalWrite(LED_PIN, HIGH);
  servoPolso.write(90);    // polso ruotato di 90°
  delay(500);
  servoPinza.write(180);  // pinza chiusa
  delay(2000);
  digitalWrite(LED_PIN, LOW);
  movelinkTo(2,90,LOW);
  delay(1000);
  movelinkTo(1,30,LOW);
  delay(1000);
  servoPolso.write(0);
  delay(500);
  servoPinza.write(0);  // pinza aperta
  delay(1000);

  movezTo(30,HIGH);
  delay(1000);
}

void movelinkTo(int id, int postion, int orientation) {
  if (id==1) {
    digitalWrite(DIR_PIN1, orientation);
```

```
    for (int i = 0; i < postion; i++) {
      digitalWrite(STEP_PIN1, HIGH); delayMicroseconds(DELAY_ST);
      digitalWrite(STEP_PIN1, LOW);  delayMicroseconds(DELAY_ST);
    }
  }
  else if (id==2) {
    digitalWrite(DIR_PIN2, orientation);
    for (int i = 0; i < postion; i++) {
      digitalWrite(STEP_PIN2, HIGH); delayMicroseconds(DELAY_ST);
      digitalWrite(STEP_PIN2, LOW);  delayMicroseconds(DELAY_ST);
    }
  }
}

// spostamento relativo in mm
void movezTo(int z, int orientation) {
    int t0= millis();
    int t= millis();

    // calcolo numero di step necessari (1mm=100 step)
    int step = z * 100;
    digitalWrite(DIR_PIN0, orientation);
    for (int i = 0; i < step; i++) {
       /* questo codice rallenta velocità motore dopo 2-3 volte !!!!
       float cz= (i+1)/100;
       if ( (millis() - t) >= 500) {
         t = millis();
         if (orientation==LOW) {lcd.setCursor(6, 0); lcd.print("-");}
         else {lcd.setCursor(6, 0); lcd.print("+");}
         lcd.setCursor(7, 0); lcd.print(cz);
         // calcolo rpm  -> 1 giro = 200 step
        float vel = 1000 * cz / (millis() - t0);
        lcd.setCursor(8, 1); lcd.print(vel);
       }
       */
       digitalWrite(STEP_PIN0, HIGH); delayMicroseconds(DELAY_STZ);
       digitalWrite(STEP_PIN0, LOW);  delayMicroseconds(DELAY_STZ);
    }

    float cz= step/100;
    if (orientation==LOW) {lcd.setCursor(6, 0); lcd.print("-");}
    else {lcd.setCursor(6, 0); lcd.print("+");}
    lcd.setCursor(7, 0); lcd.print(cz);

    float vel = 1000* cz / (millis() - t0);
    lcd.setCursor(8, 1); lcd.print(vel);
}
```

ROBOT ANTROPOMORFO

I robot antropomorfi sono robot con movimenti su 5 o più assi che ricordano nella forma e nelle possibilità di articolazione il braccio umano. Per questo motive anche denominati bracci robotici antropomorfi.

Esempio piccolo robot hobbistico.



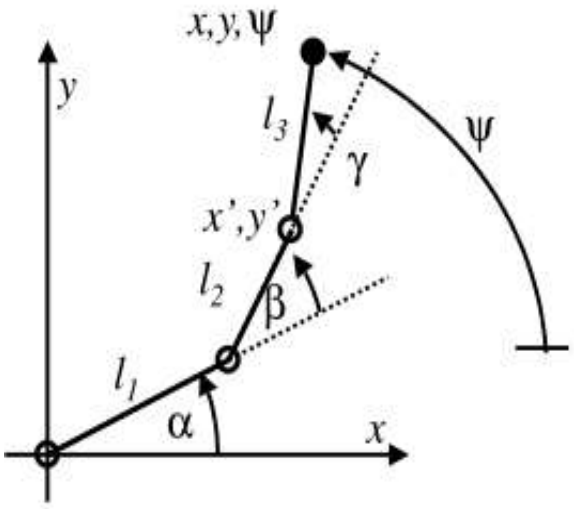
La cinematica diretta è risolta dalle seguenti equazioni:

$$\begin{cases} x &= l_1\cos(\alpha)+l_2\cos(\alpha+\beta)+l_3\cos(\alpha+\beta+\gamma) \\ y &= l_1\sin(\alpha)+l_2\sin(\alpha+\beta)+l_3\sin(\alpha+\beta+\gamma) \\ \psi &= \alpha+\beta+\gamma \end{cases}$$

La cinematica inversa si può risolvere calcolando inizialmente le coordinate x' e y' del centro del terzo accoppiamento rotoidale e applicando poi ad esse la soluzione del robot SCARA classico:

$$\begin{cases} x' &= x-l_3\cos(\psi) \\ y' &= y-l_3\sin(\psi) \\ \beta &= \pm\operatorname{acos}\left(\dfrac{x'^2+y'^2-l_1^2-l_2^2}{2l_1l_2}\right) \\ \alpha &= \operatorname{atan2}(y',x')-\operatorname{atan2}(l_2\sin(\beta),l_1+l_2\cos(\beta)) \\ \gamma &= \psi-\alpha-\beta \end{cases}$$

Il robot ha evidentemente due soluzioni e le configurazioni singolari si hanno per $\beta = 0, \pi$.

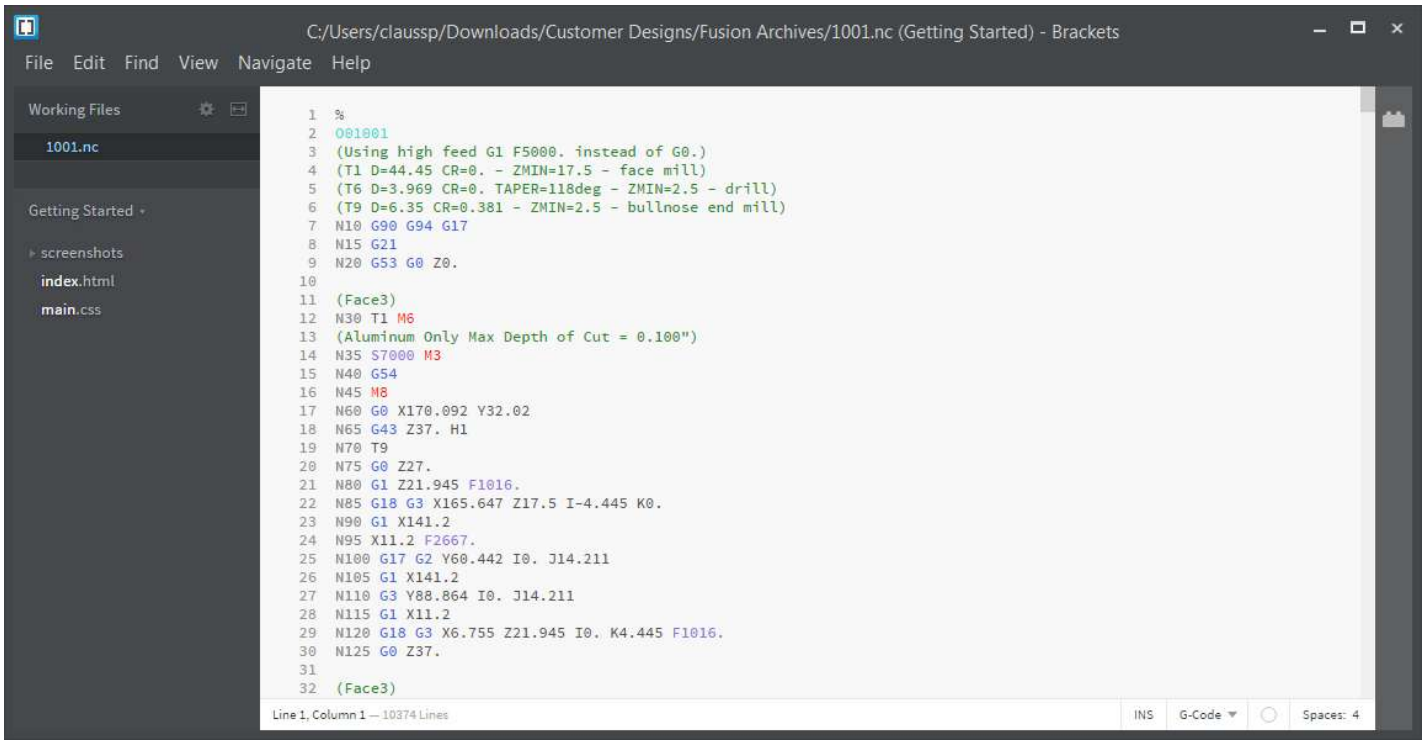


NB: l'angolo Ψ deve essere assegnato (dato di input noto).

CODICE G PER LA PROGRAMMAZIONE CNC

IL CODICE G IN SINTESI

I produttori di tutto il mondo utilizzano la programmazione CNC per controllare gli utensili di una macchina e produrre pezzi. Il cuore di questo processo di produzione automatizzato è costituito da una serie di istruzioni che indicano a un macchinario CNC dove e come muoversi.
Queste istruzioni sono chiamate Codice G (G-Code).



Il codice G è stato creato negli anni ’60 dall’Electronics Industry Association (EIA).

Sebbene il linguaggio ufficiale sia documentato come RS-274D, tutti si riferiscono ad esso come codice G.

Perché?

Molti dei termini o dei singoli frammenti di codice che compongono questo linguaggio iniziano con la lettera G.

Anche se il codice G dovrebbe essere uno standard universale, scoprirai che molte aziende produttrici di macchine CNC hanno sviluppato il loro gusto unico. Tutti apprezziamo un buon gelato, ma una Haas potrebbe essere alla fragola e una Tormach al cioccolato. A causa di questa differenza nei gusti del codice G, è fondamentale capire come il proprio macchinario utilizza il codice G.

Perché esistono differenze nei gusti del codice G? La questione è legata alle capacità di ogni macchina. Prendiamo una macchina in grado di elaborare una rotazione del sistema di coordinate in base agli input della sonda. Avrai bisogno di una serie di comandi in codice G in grado di attivare o disattivare questa rotazione. Un'altra macchina che non ha questa capacità di regolazione non avrà bisogno di questo codice G.

In caso di dubbi, fai sempre riferimento alla documentazione della tua macchina CNC mentre leggi il resto di questo articolo. Ti illustreremo le nozioni di base, ma non è detto che la tua macchina non debba seguire un percorso leggermente diverso per raggiungere la stessa destinazione finale.

BLOCCHI DI CODICE G

Gli standard del codice G sono stati pubblicati all'epoca in cui le macchine avevano una piccola quantità di memoria. A causa di questa limitazione di memoria, il codice G è un linguaggio estremamente compatto e conciso che a prima vista potrebbe sembrare arcaico. Prendiamo ad esempio questa riga di codice:

G01 X1 Y1 F20 T01 M03 S500

In questa singola riga, stiamo dando alla macchina una serie di istruzioni:

- G01 – Esegue un avanzamento lineare
- X1/Y1 – Si sposta su queste coordinate X e Y
- F20 – Spostamento con avanzamento 20
- T01 – Utilizzo dell'utensile 1 per portare a termine il lavoro
- M03 – Aziona il mandrino
- S500 – Imposta una velocità del mandrino pari a 500

Linee multiple di codice G come queste si combinano per formare un programma CNC completo.

Le macchine CNC leggono il codice una riga alla volta, da sinistra verso destra e dall'alto verso il basso, come se leggessero un libro.

Ogni serie di istruzioni si trova su una linea separata o su un blocco.

PROGRAMMI IN CODICE G

L'obiettivo di ogni programma di codice G è quello di produrre pezzi nel modo più sicuro ed efficiente possibile. Per raggiungere questo obiettivo, in genere, i blocchi di codice G sono disposti in un ordine particolare come il seguente:

1. Avvia il programma CNC.
2. Carica l'utensile richiesto.
3. Attiva il mandrino.
4. Attiva il refrigerante.
5. Spostati in una posizione al di sopra del pezzo.
6. Avvia il processo di lavorazione.
7. Disattiva il refrigerante.
8. Disattiva il mandrino.
9. Allontanati dal pezzo verso una posizione sicura.
10. Termina il programma CNC.

Questo flusso è un programma semplice che utilizza un solo strumento per un'unica operazione. In pratica, in genere si ripetono i passaggi da 2 a 9. Ad esempio, il programma in codice G riportato di seguito comprende tutti i blocchi di codice precedenti con sezioni ripetute dove necessario:

Block	Description	Purpose
%	Start of program.	Start Program
O0001 (PROJECT1)	Program number (Program Name).	
(T1 0.25 END MILL)	Tool description for operator.	
N1 G17 G20 G40 G49 G80 G90	Safety block to ensure machine is in safe mode.	
N2 T1 M6	Load Tool #1.	Change Tool
N3 S9200 M3	Spindle Speed 9200 RPM, On CW.	
N4 G54	Use Fixture Offset #1.	Move To Position
N5 M8	Coolant On.	
N6 G00 X-0.025 Y-0.275	Rapid above part.	
N7 G43 Z1. H1	Rapid to safe plane, use Tool Length Offset #1.	
N8 Z0.1	Rapid to feed plane.	
N9 G01 Z-0.1 F18.	Line move to cutting depth at 18 IPM.	
N10 G41 Y0.1 D1 F36.	CDC Left, Lead in line, Dia. Offset #1, 36 IPM.	Machine Contour
N11 Y2.025	Line move.	
N12 X2.025	Line move.	
N13 Y-0.025	Line move.	
N14 X-0.025	Line move.	
N15 G40 X-0.4	Turn CDC off with lead-out move.	
N16 G00 Z1.	Rapid to safe plane.	
N17 M5	Spindle Off.	Change Tool
N18 M9	Coolant Off.	
(T2 0.25 DRILL)	Tool description for operator.	
N19 T2 M6	Load Tool #2.	
N20 S3820 M3	Spindle Speed 3820 RPM, On CW.	
N21 M8	Coolant On.	Move To Position
N22 X1. Y1.	Rapid above hole.	
N23 G43 Z1. H2	Rapid to safe plane, use Tool Length Offset 2.	
N24 Z0.25	Rapid to feed plane.	
N25 G98 G81 Z-0.325 R0.1 F12.	Drill hole (canned) cycle, Depth Z-.325, F12.	Drill Hole
N26 G80	Cancel drill cycle.	
N27 Z1.	Rapid to safe plane.	
N28 M5	Spindle Off.	End Program
N29 M9	Coolant Off.	
N30 G91 G28 Z0	Return to machine Home position in Z.	
N31 G91 G28 X0 Y0	Return to machine Home position in XY.	
N32 G90	Reset to absolute positioning mode (for safety).	
N33 M30	Reset program to beginning.	
%	End Program.	

MODALI E CODICI DI INDIRIZZO

Come altri linguaggi di programmazione, il codice G può ripetere un'azione all'infinito finché non viene interrotta. Questo processo di looping utilizza un codice modale, che agisce fino a quando non viene disattivato o modificato con un altro codice modale. Ad esempio, M03 è un codice modale che fa girare un mandrino all'infinito finché non gli ordini di fermarsi con M05. Ora, aspetta un attimo. Questa parola (ricorda: una parola è un piccolo pezzo di codice) non inizia con la G, ma è comunque un codice G. Le parole che iniziano con una M sono codici macchina e attivano o disattivano funzioni della macchina come il refrigerante, il mandrino e i morsetti. Ne elencheremo alcuni comuni nella prossima sezione, ma puoi trovare un elenco dei codici M della tua macchina nella sua documentazione.

Il codice G include anche un elenco completo di codici di indirizzo. Puoi considerarli come il dizionario del codice G che definisce particolari comportamenti. I codici di indirizzo iniziano con la lettera di designazione, come G, e seguono con una serie di numeri. Ad esempio, X2 definisce un codice di indirizzo per la coordinata X, dove 2 è il valore sull'asse X su cui spostare la macchina.

L'elenco completo dei codici di indirizzo comprende:

Code	Meaning
A	Rotation about X-axis.
B	Rotation about Y-axis.
C	Rotation about Z-axis.
D	Cutter diameter compensation (CDC) offset address.
F	Feed rate.
G	G-Code (preparatory code).
H	Tool length offset (TLO).
I	Arc center X-vector, also used in drill cycles.
J	Arc center Y-vector, also used in drill cycles.
K	Arc center Z-vector, also used in drill cycles.
M	M-Code (miscellaneous code).
N	Block Number.
O	Program Number.
P	Dwell time.
Q	Used in drill cycles.
R	Arc radius, also used in drill cycles.
S	Spindle speed in RPM.
T	Tool number.
X	X-coordinate.
Y	Y-coordinate.
Z	Z-coordinate.

A un programma in codice G possono essere aggiunti diversi codici di caratteri speciali. In genere vengono utilizzati per avviare un programma, eliminare il testo o ignorare i caratteri. Includono:

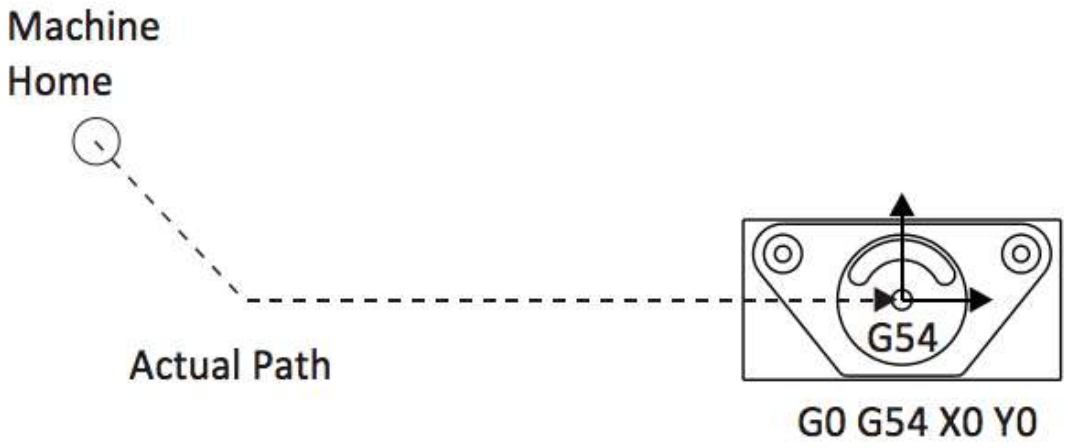
- % Inizia o termina un programma CNC
- () Definisce un commento scritto da un operatore CNC; occasionalmente deve essere scritto tutto in maiuscolo.
- / Ignora tutti i caratteri che vengono dopo lo slash
- ; Determina la fine di un blocco di codice, non visualizzabile in un editor di testo.

PANORAMICA DEI CODICI G E DEI CODICI M

I codici G e M costituiranno la maggior parte del tuo programma CNC. I codici che iniziano con G preparano la tua macchina a eseguire un tipo specifico di movimento. I codici G più comuni che si incontrano più volte in ogni programma CNC sono:

G0 – MOVIMENTO RAPIDO

Questo codice indica a una macchina di spostarsi il più velocemente possibile verso una posizione di coordinate specificata. G0 sposterà la macchina asse per asse, il che significa che si muoverà prima lungo entrambi gli assi e terminerà lo spostamento su quello che non è in posizione. Nella figura seguente è mostrato un esempio di questo movimento:



G1 – MOVIMENTO LINEARE

Questo codice indica a una macchina di muoversi in linea retta verso una posizione coordinata con un avanzamento definito. Ad esempio, G1 X1 Y1 F32 sposterà la macchina verso le coordinate X1, Y1, con un avanzamento di 32.

G2, G3 – Arco in senso orario, arco in senso antiorario

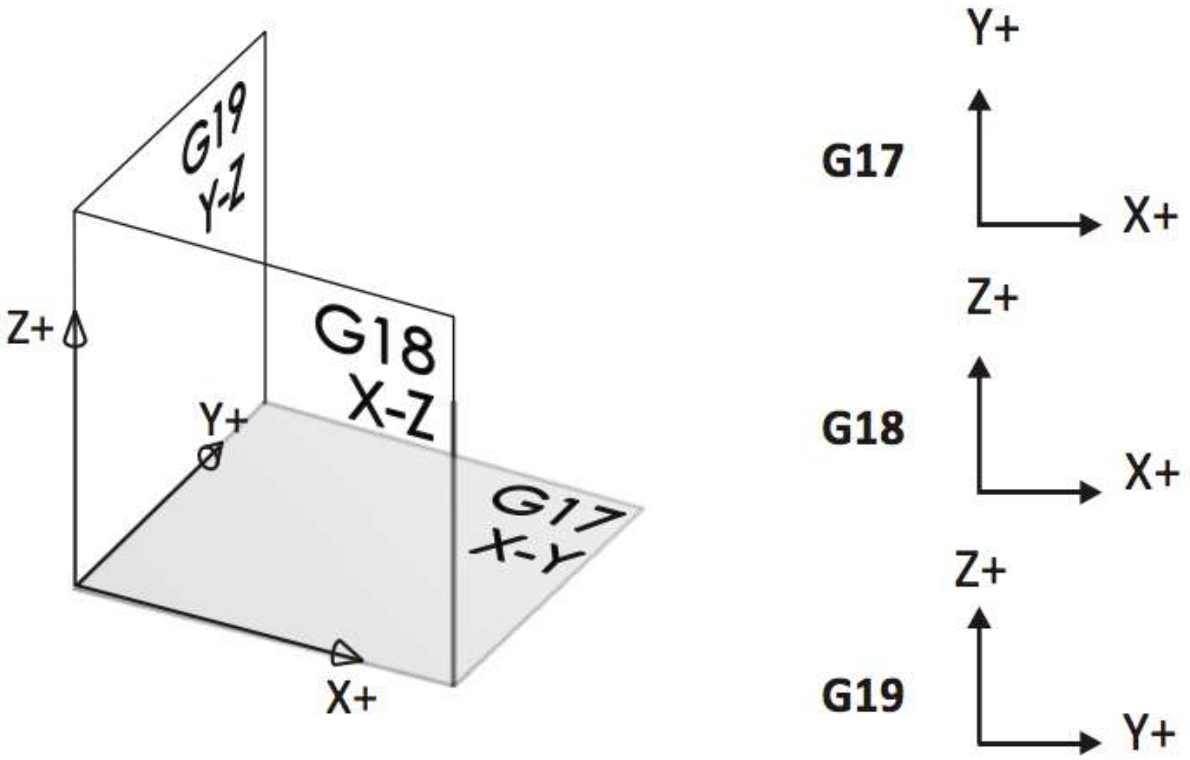
Questi codici indicano alla macchina di muoversi in un arco verso una coordinata di destinazione. Due coordinate aggiuntive, I e J, definiscono la posizione centrale dell'arco, come mostrato di seguito:

G2 X0. Y-.25 I0. J-.25

G3 X-.25 Y0. I-.25 J0.

G17, G18, G19 – DESIGNAZIONI DEI PIANI

Questi codici definiscono su quale piano verrà lavorato un arco. Per impostazione predefinita, la tua macchina CNC utilizzerà G17, che è il piano XY. Gli altri due piani sono mostrati nell'immagine sottostante:



G40, G41, G42 – COMPENSAZIONE DEL DIAMETRO DELL'UTENSILE

Questi codici definiscono la compensazione del diametro della lama, o CDC, che permette a una macchina CNC di posizionare l'utensile a sinistra o a destra di un percorso definito.
Un registro D memorizza la compensazione per ogni utensile.

Tool Diameter Offset	Value
D1	0.0020
D2	0.0000
D3	0.0000
D4	0.0000
D5	0.0000
D6	0.0000

G43 – Compensazione della lunghezza dell'utensile

Questo codice definisce la lunghezza dei singoli utensili utilizzando l'altezza dell'asse Z.
Ciò consente alla macchina CNC di capire dove si trova la punta di un utensile rispetto al pezzo su cui sta lavorando.
Un registro definisce le compensazioni della lunghezza dell'utensile, dove H è l'offset della lunghezza dell'utensile e Z è la sua lunghezza.

Tool Length Resister	Z
H1	12.6280
H2	6.3582
H3	9.7852
H4	6.8943
H5	10.5673
H6	7.1258

G54 – COMPENSAZIONE DI LAVORAZIONE

Questo codice viene utilizzato per definire una compensazione, che determina la distanza tra le coordinate interne di una macchina e l'origine di un pezzo.
Nella tabella sottostante, solo G54 ha una definizione di compensazione.
Tuttavia, si possono programmare più compensazioni se un lavoro richiede la lavorazione contemporanea di più pezzi.

Work Offset	X	Y	Z
G54	14.2567	6.6597	2.0183
G55	0.0000	0.0000	0.0000
G56	0.0000	0.0000	0.0000
G57	0.0000	0.0000	0.0000
G58	0.0000	0.0000	0.0000
G59	0.0000	0.0000	0.0000

CODICI M

I codici M sono codici macchina che possono differire tra le macchine CNC.
Questi codici controllano le funzioni della macchina CNC, come le direzioni del refrigerante e del mandrino.

Alcuni dei codici M più comuni sono i seguenti:

Code	Meaning
M0	Program stop. Press Cycle Start button to continue.
M1	Optional stop. Only executed if Op Stop switch on the CNC control is turned ON.
M2	End of program.
M3	Spindle on Clockwise.
M4	Spindle on Counterclockwise.
M5	Spindle stop.
M6	Change tool.
M8	Coolant on.
M9	Coolant off.
M30	End program and press Cycle Start to run it again.

CICLI FISSI IN CODICE G

L'ultimo aspetto del codice G da toccare è quello dei cicli fissi. Sono simili ai metodi o alle funzioni della programmazione informatica.

Consentono di eseguire un'azione complicata con poche righe di codice, senza dover digitare tutti i dettagli.

Prendiamo, ad esempio, il seguente ciclo fisso. In questo caso stiamo dicendo allo strumento CNC di creare un foro con una perforatrice in sole due righe di codice sulla sinistra.

La stessa azione richiede oltre 20 righe di regolare codice G.

Canned Cycle	Equivalent Motion: Expanded Code
N70 G98 G83 X1. Y1. Z-1.04 R0.06 Q0.15 P0 F9. N75 G80	N70 Z0.06 N75 Z0.04 N80 G01 Z-0.19 F9. N85 G00 Z0.06 N90 Z-0.11 N95 G01 Z-0.34 N100 G00 Z0.06 N105 Z-0.26 N110 G01 Z-0.49. N115 G00 Z0.06 N120 Z-0.41 N125 G01 Z-0.64. N130 G00 Z0.06 N135 Z-0.56 N140 G01 Z-0.79 N145 G00 Z0.06 N150 Z-0.71 N155 G01 Z-0.94. N160 G00 Z0.06 N165 Z-0.86 N170 G01 Z-1.04. N175 G00 Z0.25

ELETTROPNEUMATICA

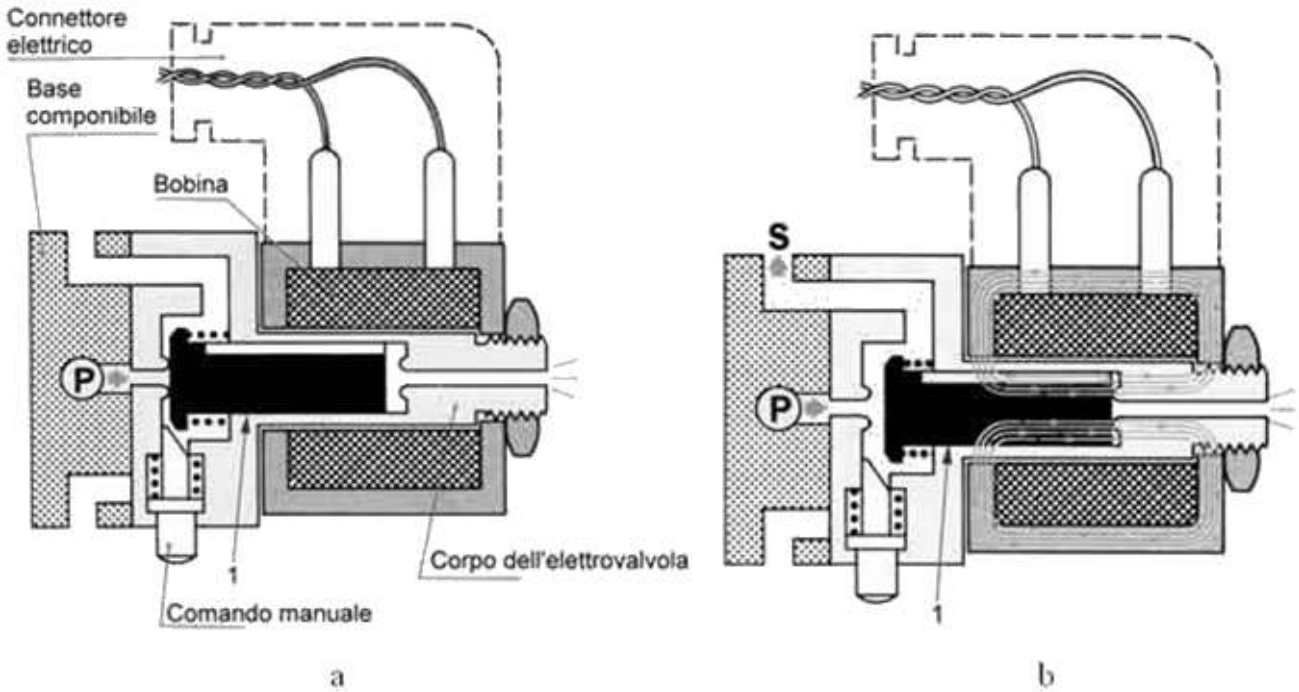
La tecnologia pneumatica è basata sull'utilizzo di un gas compresso (aria) per produrre un movimento meccanico. Questa tecnologia appartiene al campo della tecnologia dei fluidi, assieme all'oleodinamica. Tuttavia, a differenza di quest'ultima che utilizza i fluidi come mezzo propulsivo, la pneumatica utilizza l'aria compressa che è un'alternativa economica ed ecologica per la la movimentazione di macchine ed utensili.

Nella pneumatica il passaggio di aria compressa che genera un movimento meccanico avviene tramite comandi di tipo meccanico. Se si utilizzano elettrovalvole e segnali di comando elettrici si parla di ELETTROPNEUMATICA.

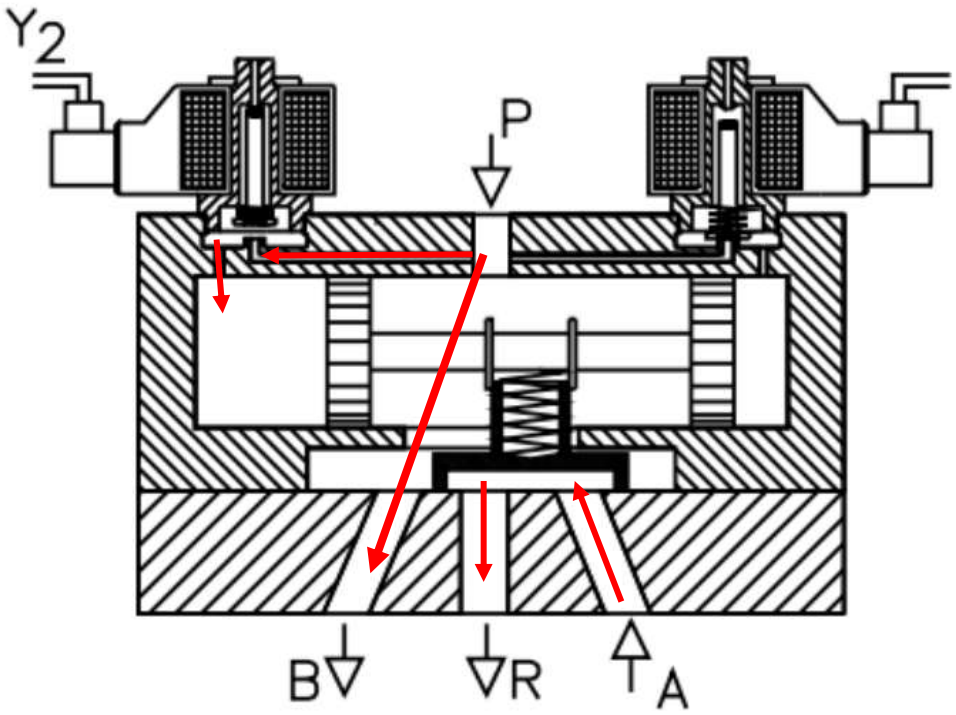
ELETTROVALVOLE PNEUMATICHE

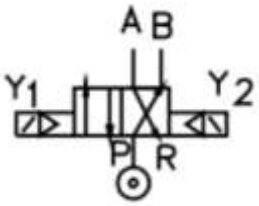
Una elettrovalvola (o valvola a solenoide) è una valvola che utilizza la forza elettromagnetica per funzionare.
Quando una corrente elettrica viene fatta passare attraverso la bobina del solenoide (generalmente alimentata a 24V), viene generato un campo magnetico che provoca il movimento di un perno metallico che permette il passaggio dell'aria da una via ad un'altra.

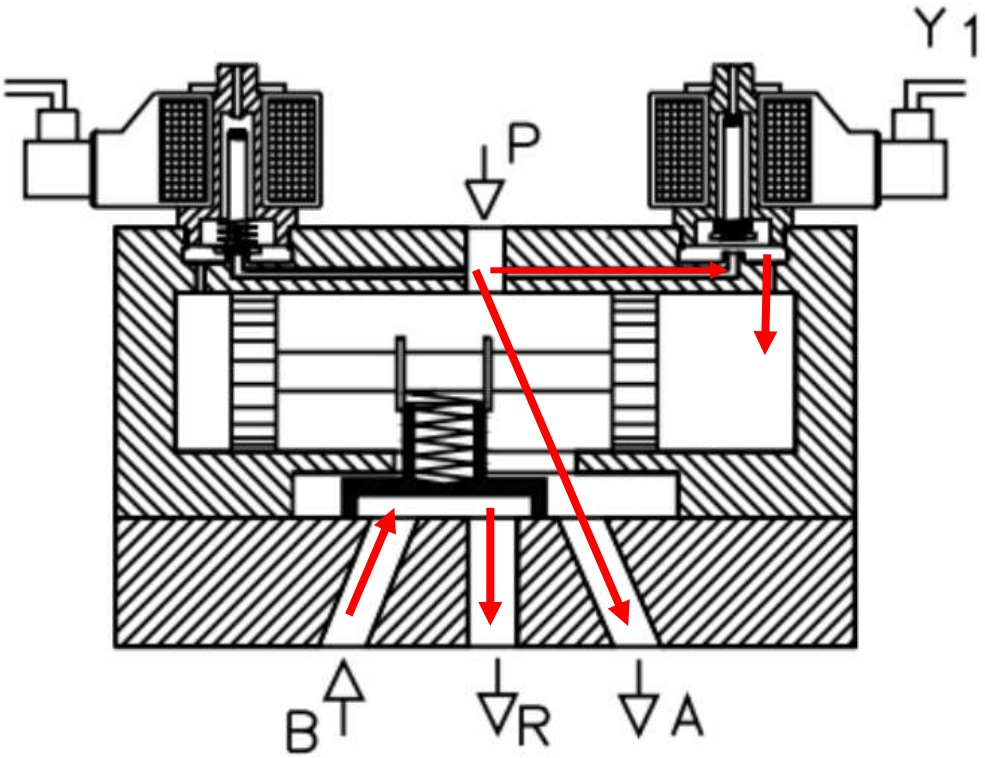
Schema funzionamento della bobina per una valvola unidirezionale



Schema funzionamento delle 2 bobine di una elettrovalvola 4/2 bistabile con piattello scorrevole


Y2
P
B
R
A


A
B
Y1
Y2
P
R


Y1
P
B
R
A

COMANDO ATTUATORI ELETTROPNEUMATICI CON ARDUINO

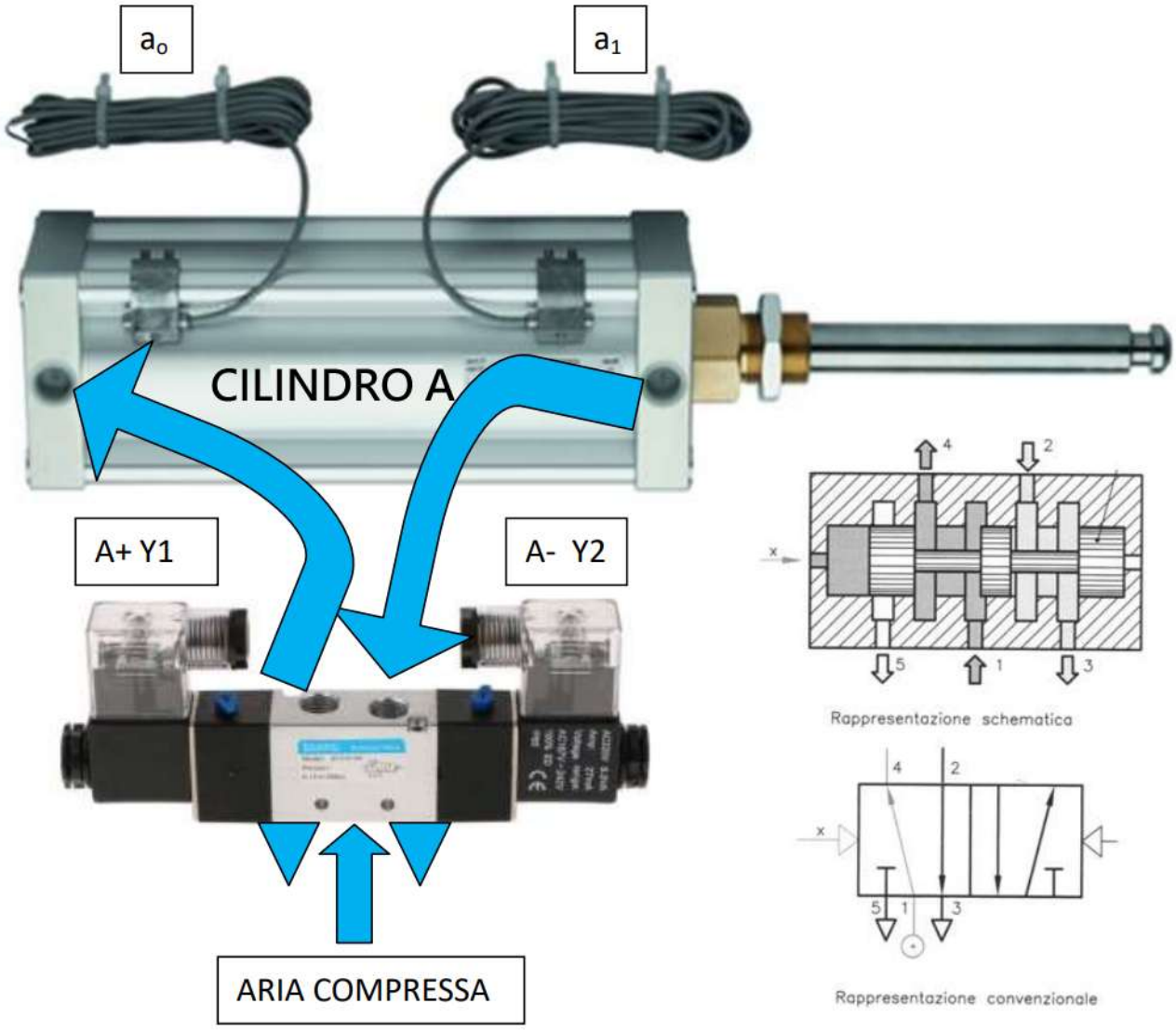
Utilizzando le valvole elettropneumatiche è possibile realizzare automatismi controllabili da un microcontrollore come Arduino.

Le bobine dell'elettrovalvola generalmente vanno alimentate a 24V per consentirne l'attivazione e di conseguenza il passaggio dell'aria 1→4 (uscita pistone) oppure 1→2 (rientro cilindro).

Per avviare le bobine si possono utilizzare dei relè comandati da Arduino tramite una uscita digitale a 5V.

Tramite il contatto NA del relè si connette la bobina dell'elettrovalvola al generatore 24V.

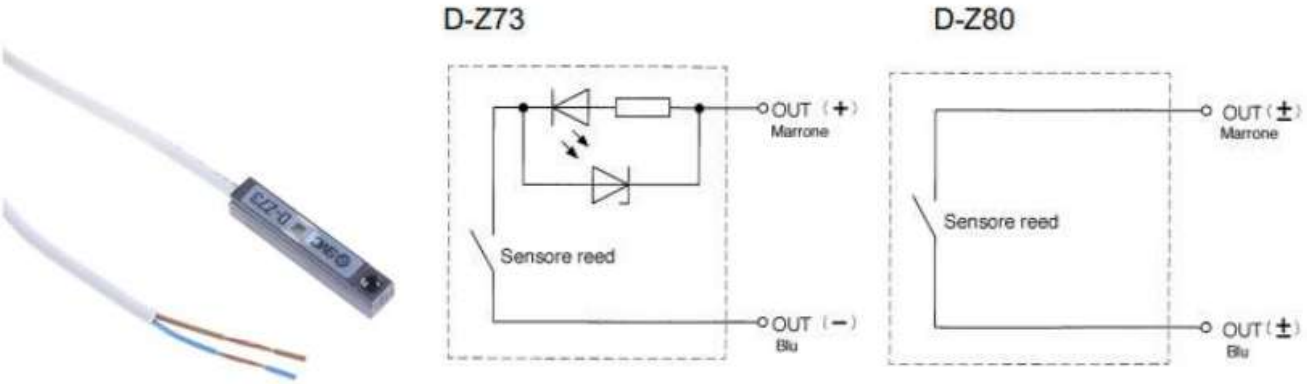
L'attivazione della bobina Y1 (con la Y2 disattivata) avvia la fase A+ (uscita del pistone) mentre l'attivazione della bobina Y2 (con la Y1 disattivata) avvia la fase A- (rientro del pistone).

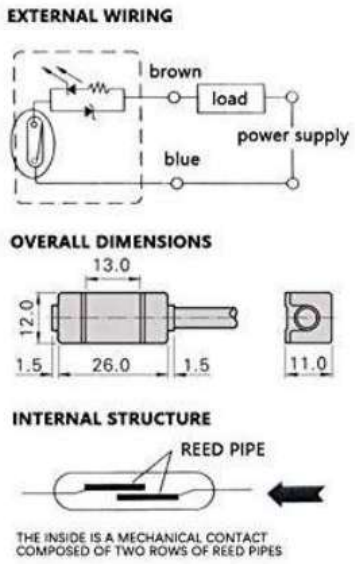


La valvola 5/2, rispetto alla 4/2 consente di avere due scarichi distinti sull'andata e il ritorno del pistone per permettere la regolazione della la velocità in modo differenziato.

SENSORI MAGNETICI (REED SWITCHES)

Sono degli interruttori che si attivano in presenza di un campo magnetico.
Si trovano sotto forma di una capsula di vetro con due steli metallici alle estremità o completamente avvolti in un case plastico/metallico che garantisce una maggiore resistenza.





In campo elettropneumatico vengono impiegati abbinati a cilindri magnetici per rilevare la posizione del pistone (dotato di fascia magnetica) all'interno del cilindro funzionando così da FINECORSA.

Nel caso di sensori a due fili è necessario alimentare il sensore tramite un carico resistivo per limitare la corrente a pochi *mA* quando il circuito è chiuso (presenza di campo magnetico).

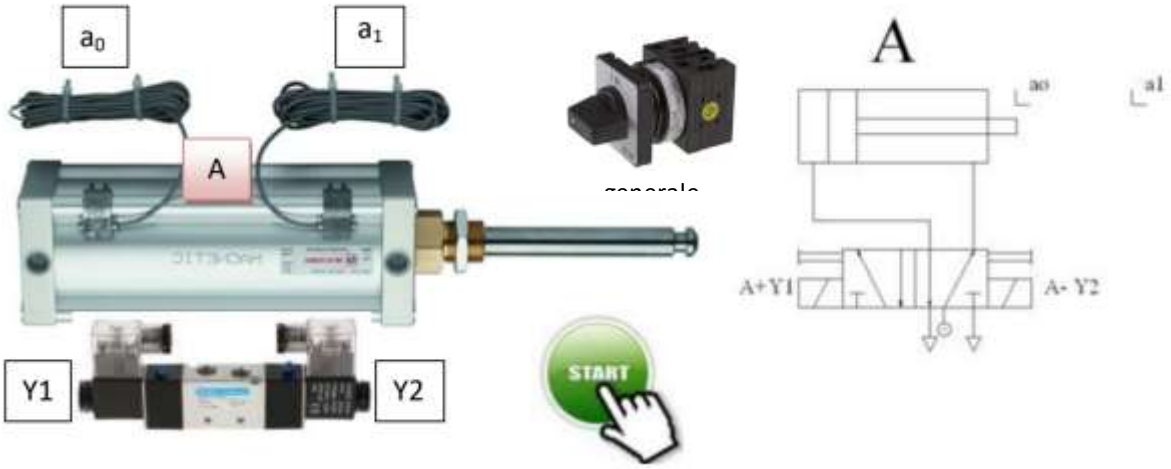
In campo industriale in generale la tensione di alimentazione varia da 4 a 24V con una corrente massima di 50mA.

Se il sensore è dotato di led in generale ha già integrata una resistenza limitatrice.

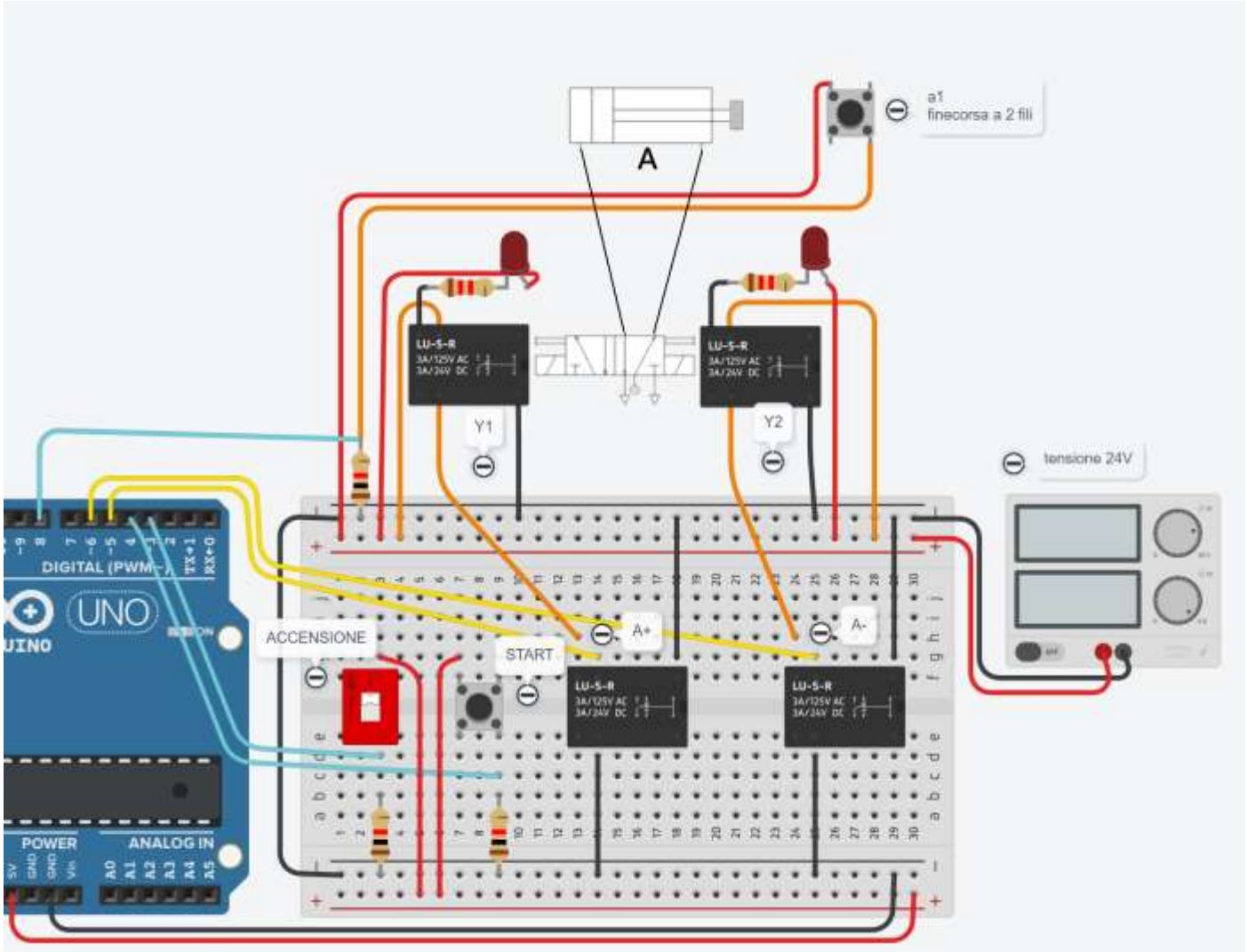
Il led si accende in presenza di un campo magnetico (presenza pistone con anello magnetico).

ESERCITAZIONE SEQUENZA PNEUMATICA

Implementare la sequenza logica “A+/pausa 5s/A-“ con un attuare pneumatico comandato da una elettrovalvola 5/2 (24V) tramite una scheda Arduino che utilizza 2 relè a 5V.
L’attuatore pneumatico è dotato di un sensore di prossimità (a_1) a 2 fili senza resistenza interna (24V).
La sequenza viene avviata premendo il pulsante START solo se è attivo un interruttore generale.
Le bobine dell’elettrovalvola vanno simulate tramite 2 relè mentre il finecorsa “a_1” con un pulsante.



SCHEMA THINKERCAD



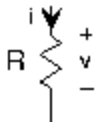
CODICE

```
int statoStart;
int statoAccensione;
int statoA1;
bool sequenzaAttiva=false;

void setup()
{
 Serial.begin(9600);
 pinMode(3, INPUT);          // interruttore generale
 pinMode(4, INPUT);          // pulsante START
 pinMode(5, OUTPUT);         // bobina Y1
 pinMode(6, OUTPUT);         // bobina Y2
 pinMode(8, INPUT);          // finecorsa a1
}

void loop()
{
 statoAccensione= digitalRead(3);

 statoStart= digitalRead(4);
 if (statoStart== HIGH && statoAccensione== HIGH) {
   sequenzaAttiva= true;
   Serial.println("Avvio sequenza");
 }

 if (sequenzaAttiva== true && statoAccensione== HIGH) {
  digitalWrite(5, HIGH);
  digitalWrite(6, LOW);
  Serial.println("A+");

  statoA1= digitalRead(8);
  // finchè il finecorsa a1 non passa ad alto attendo
  while (statoA1== LOW) {
    statoA1= digitalRead(8);
    delay(100);
  }
  Serial.println("a1 ALTO");
  Serial.println("Pausa 5s");
  delay(5000);

  digitalWrite(5, LOW);
  digitalWrite(6, HIGH);
  Serial.println("A-");
  Serial.println("Pausa 1s");  // pausa per garantire rientro pistone
  delay(1000);

  Serial.println("Disattivo relè");
  digitalWrite(5, LOW);
  digitalWrite(6, LOW);
  Serial.println("Fine sequanza");

  sequenzaAttiva= false;
 }

}
```

FORMULE ELEMENTI CIRCUITALI IDEALI

Gli elementi elementari del circuito - il resistore, il condensatore e l'induttore - impongono relazioni lineari tra tensione e corrente.

RESISTORE



$$v = Ri$$

Figura 1. Resistenza.

Il resistore è di gran lunga l'elemento circuitale più semplice. In un resistore la tensione è proporzionale alla corrente, con la costante di proporzionalità , noto come la resistenza .

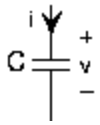
$$v\left(t\right) = Ri\left(t\right)$$

La resistenza ha unità di ohm, denotate dal nome dello scienziato elettrico tedesco Georg Ohm . Quando la resistenza è positiva, come nella maggior parte dei casi, un resistore consuma energia. Il consumo energetico istantaneo di un resistore può essere scritto in due modi.

$$p\left(t\right) = Ri^2\left(t\right) = \frac{1}{R}v^2\left(t\right)$$

Quando la resistenza si avvicina all'infinito, abbiamo quello che è noto come un circuito aperto : nessuna corrente scorre ma una tensione diversa da zero può apparire attraverso il circuito aperto. Quando la resistenza diventa zero, la tensione va a zero per un flusso di corrente diverso da zero. Questa situazione corrisponde ad un cortocircuito . Un superconduttore realizza fisicamente un cortocircuito.

CONDENSATORE



$$i = C\frac{dv\left(t\right)}{d_t}$$

Figura 2. Condensatore.

Il condensatore immagazzina la carica e la relazione tra la carica immagazzinata e la tensione risultante è

$$q = Cv.$$

La costante di proporzionalità, la capacità, ha unità di farad (F), e prende il nome dal fisico sperimentale inglese Michael Faraday .
Poiché la corrente è la velocità di variazione della carica, la relazione vi può essere espressa in forma differenziale o integrale.

$$i\left(t\right) = C\frac{dv\left(t\right)}{dt} \quad \text{or} \quad v\left(t\right) = \frac{1}{C}\int_{-\infty}^{t} i\left(\alpha\right)d\alpha$$

Se la tensione ai capi di un condensatore è costante, la corrente che scorre in esso è uguale a zero. In questa situazione, il condensatore è equivalente a un circuito aperto. La potenza consumata/prodotta da una tensione applicata ad un condensatore dipende dal prodotto della tensione per la sua derivata.

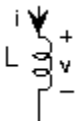
$$p(t) = Cv(t)\frac{dv(t)}{dt}$$

Questo risultato significa che il dispendio energetico totale di un condensatore fino al momento t è sinteticamente dato da

$$E(t) = \frac{1}{2}Cv^2(t)$$

Questa espressione presuppone l' assunto fondamentale della teoria dei circuiti: tutte le tensioni e le correnti in qualsiasi circuito erano pari a zero nel lontano passato (

INDUTTORE



$$v = L\frac{di(t)}{d}$$

Figura 3. Induttore.

L'induttore immagazzina il flusso magnetico, con induttori di valore maggiore in grado di immagazzinare più flusso. L'induttanza ha unità di henry (H) e prende il nome dal fisico americano Joseph Henry . Le forme differenziali e integrali della relazione vi dell'induttore sono

$$v(t) = L\frac{di(t)}{dt} \quad \text{or} \quad i(t) = \frac{1}{L}\int_{-\infty}^{t} v(\alpha)\,d\alpha$$

La potenza consumata/prodotta da un induttore dipende dal prodotto della corrente dell'induttore e della sua derivata

$$p(t) = Li(t)\frac{di(t)}{dt}$$

e il suo dispendio energetico totale fino al momento è dato da

$$E(t) = \frac{1}{2}Li^2(t)$$

SORGENTI

Figura 4. La sorgente di tensione a sinistra e la sorgente di corrente a destra sono come tutti gli elementi del circuito in quanto hanno una relazione particolare tra la tensione e la corrente definita per loro.

Per la sorgente di tensione: V= Vs

Per la sorgente di corrente: I= Is