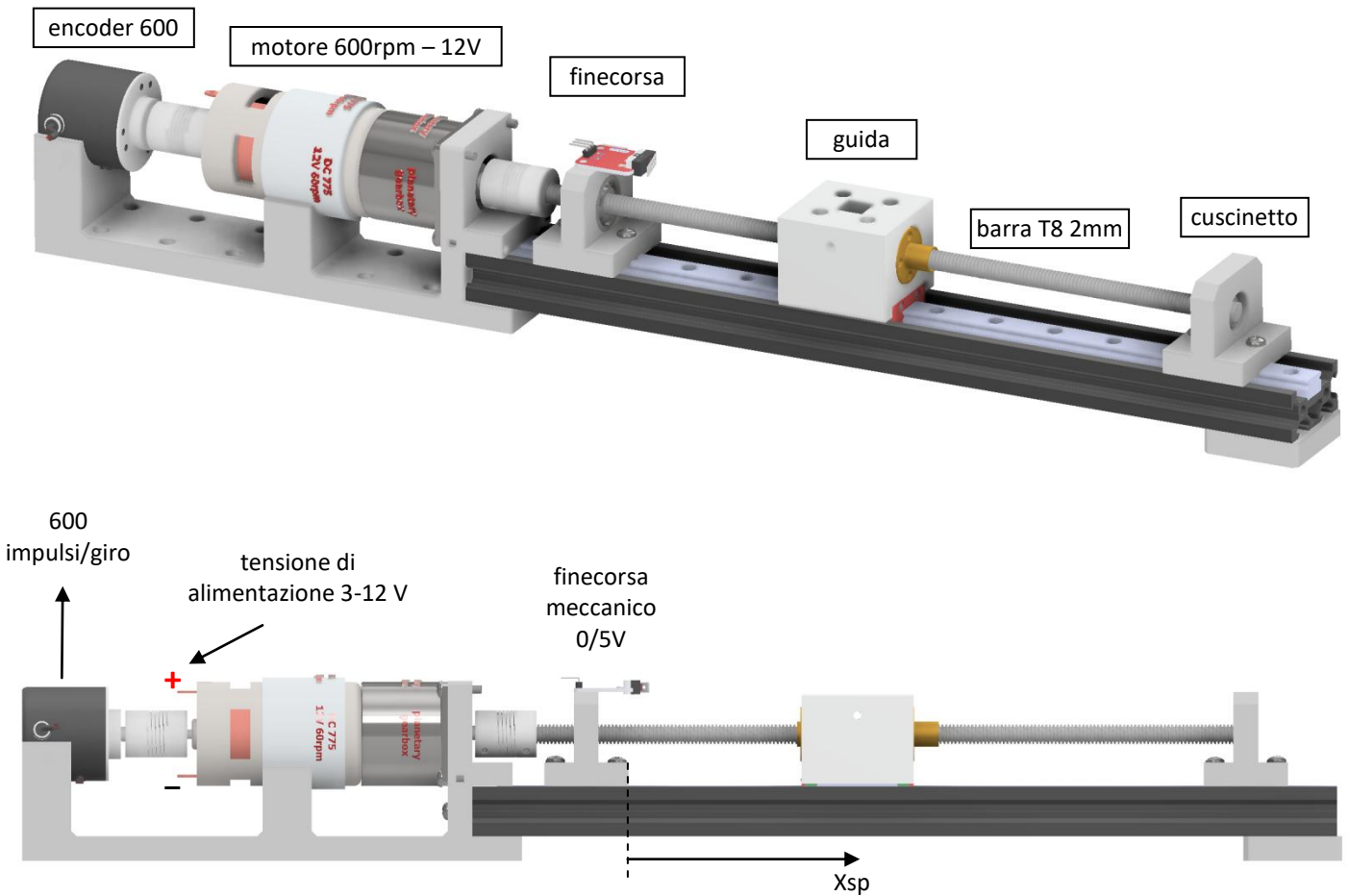


## CONTROLLO IN POSIZIONE DI UNA GUIDA LINEARE CON MOTORE C.C. E ENCODER OTTICO INCREM.

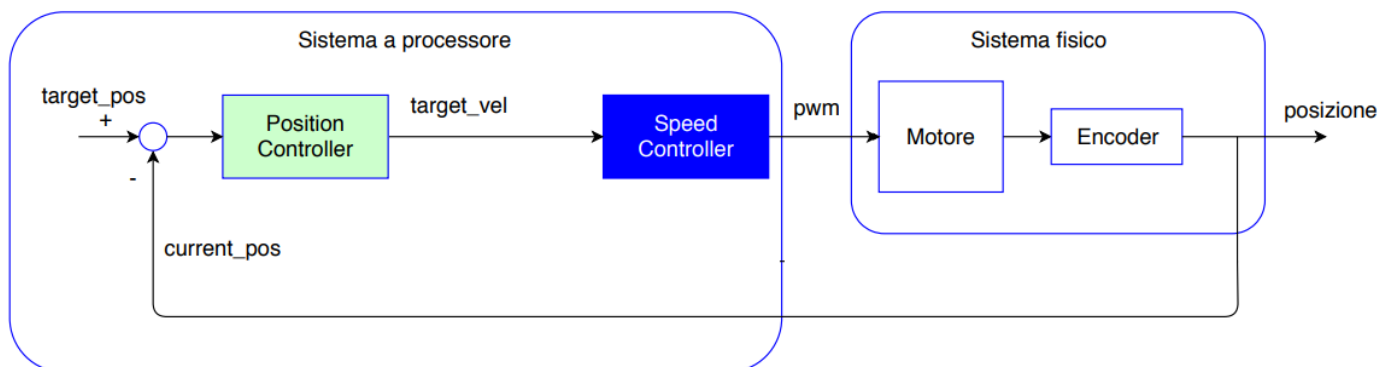
L'obiettivo è portare la guida nella posizione di SET-POINT indicata con un basso margine di errore.

Se "siamo lontani" dalla posizione  $X_{sp}$  di SET-POINT 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  $X_{sp}$  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:

$$\rightarrow v_m = K_p \cdot \text{errore} = K_p \cdot (X_{sp} - X_c) \quad K_p = \text{coeff. Proporzionale, } X_c = \text{posizione corrente}$$

In un motore C.C. la velocità è proporzionale alla tensione di alimentazione (controllata in PWM) quindi:

$$\rightarrow V_m = K_p \cdot \text{errore} = K_p \cdot (X_{sp} - X_c) \quad K_p = \text{coeff. Proporzionale, } X_c = \text{posizione corrente}$$

*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:

- $n^\circ \text{ max} = 600 \text{ rpm}$  con  $V_m = 12 \text{ V}$   $\rightarrow 10 \text{ giri/s}$   $\rightarrow a_{\text{max}} = 20 \text{ mm/s}$  (100%)
- $n^\circ \text{ min} = 50 \text{ rpm}$  con  $V_m = 1 \text{ V}$   $\rightarrow 0,834 \text{ giri/s}$   $\rightarrow a_{\text{min}} = 1,667 \text{ mm/s}$  (10%)

Curva del motore lineare:  $n^\circ = (10/12) \cdot V_m$  [giri/s]

L'encoder ottico di tipo incrementale fornisce 600 impulsi al giro.

Noto il numero di impulsi nell'intervallo di tempo di campionamento  $\Delta t$  lo spostamento della guida vale quindi:

$$s = (n_{\text{impulsi}} / 600) \cdot 2 \text{ [mm]}$$

La velocità di spostamento della guida vale:

$$v_a = s / \Delta t \text{ [mm/s]}$$

## SIMULAZIONE CON EXCEL DEL SISTEMA DI CONTROLLO P

E' necessario fissare un valore di  $K_p$  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

$X_{sp} = 30$        $K_p = 1$   
 $V_{max} = 12$        $dt = 0,1$

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

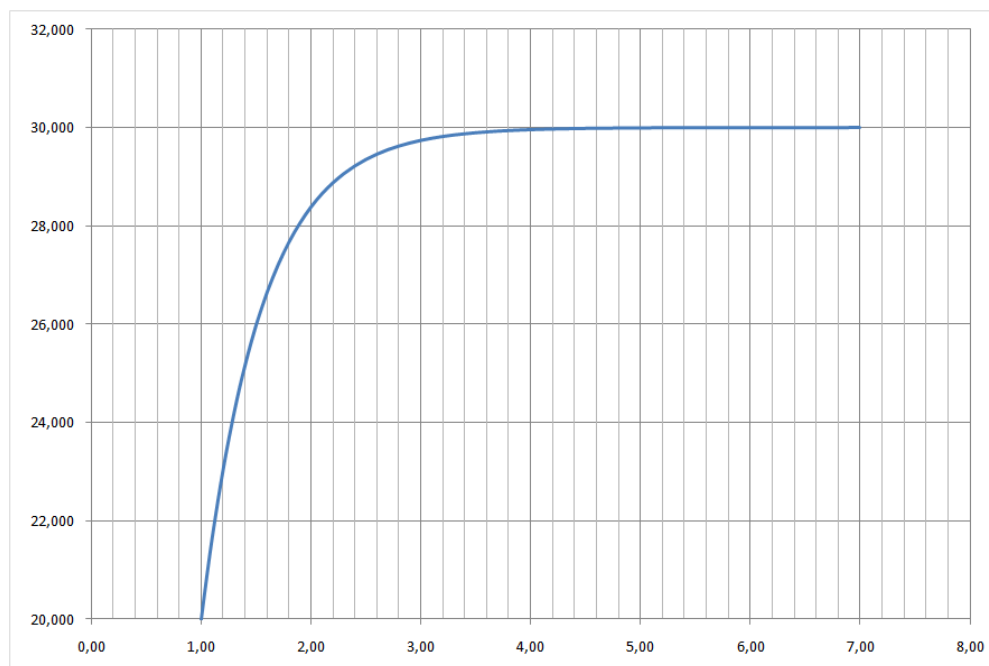
$X_{sp} = 30$        $K_p = 1$   
 $V_{max} = 12$        $dt = 0,1$

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	

$X_{sp} = 30$        $K_p = 1$   
 $V_{max} = 12$        $dt = 0,1$

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  $a = -33 \text{ mm/s}^2$ .



## SIMULAZIONE CON EXCEL DEL SISTEMA DI CONTROLLO PID

E' necessario fissare un valore di  $K_p$ ,  $K_i$  e  $K_d$  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

$X_{sp}= 30$        $K_p= 1$        $K_i= 0,01$        $K_d= 0,01$   
 $V_{max} 12$        $dt= 0,1$

t [s]	Vm [V]	n° [giri/s]	s [mm]	vel.	accel.	errore			Vm_teor.	Vm_eff [V]
						err. [mm]	integrale	derivata		
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

$X_{sp}= 30$        $K_p= 1$        $K_i= 0$        $K_d= 0$   
 $V_{max} 12$        $dt= 0,1$

t [s]	Vm [V]	n° [giri/s]	s [mm]	vel.	accel.	errore		
						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

$X_{sp}= 30$        $K_p= 1$        $K_i= 0$        $K_d= 0$   
 $V_{max} 12$        $dt= 0,1$

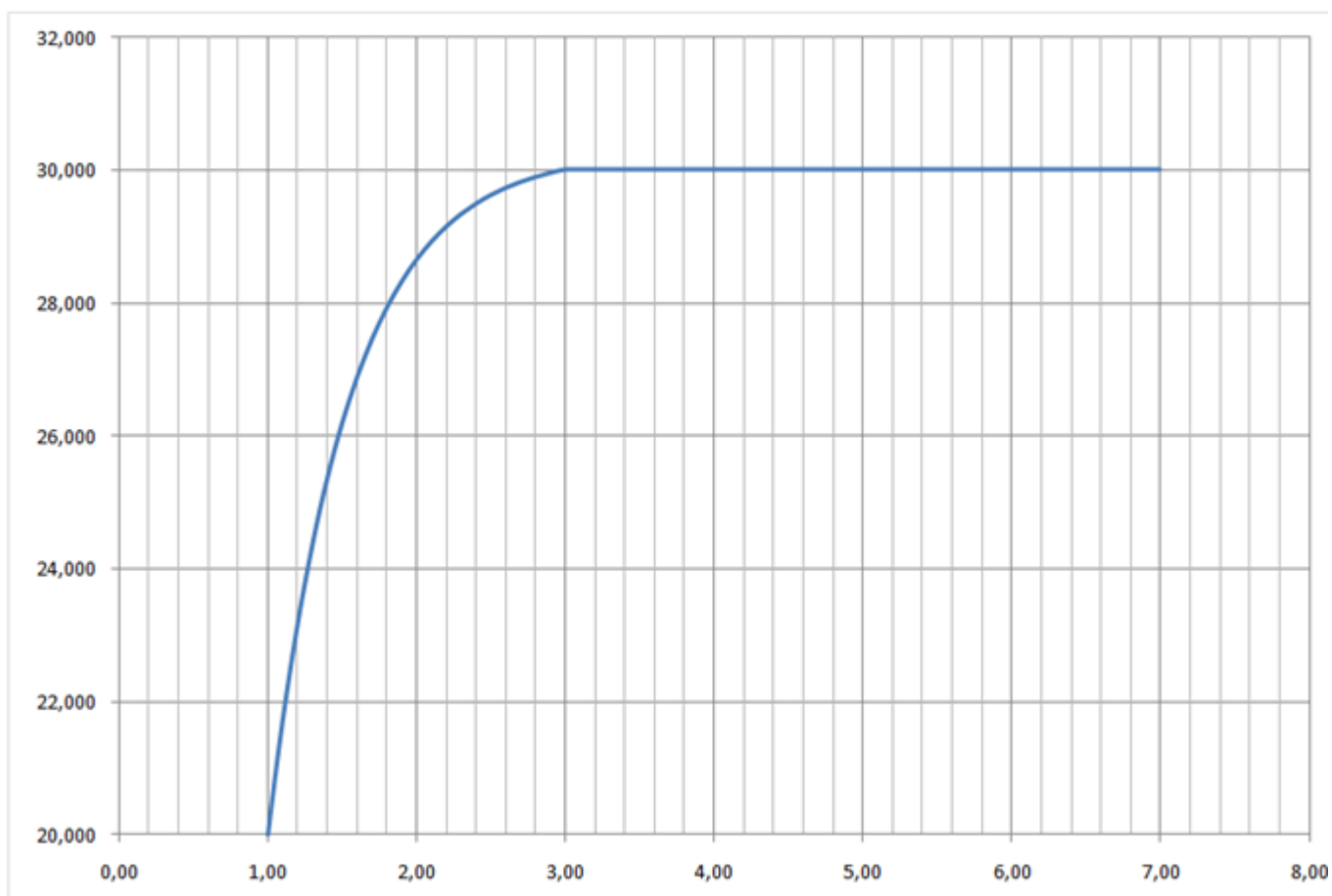
t [s]	Vm [V]	n° [giri/s]	s [mm]	vel.	accel.	errore			Vm_te
						err. [mm]	integrale	derivata	
0,00	12,00	10,00	0,000	0,0	0,0	30,000	0,000	0,000	30
0,10	12,00	10,00	2,000	20,0	200,0	28,000	3,000	=(G6-G7)/\$D\$3	

Tensione di controllo del motore

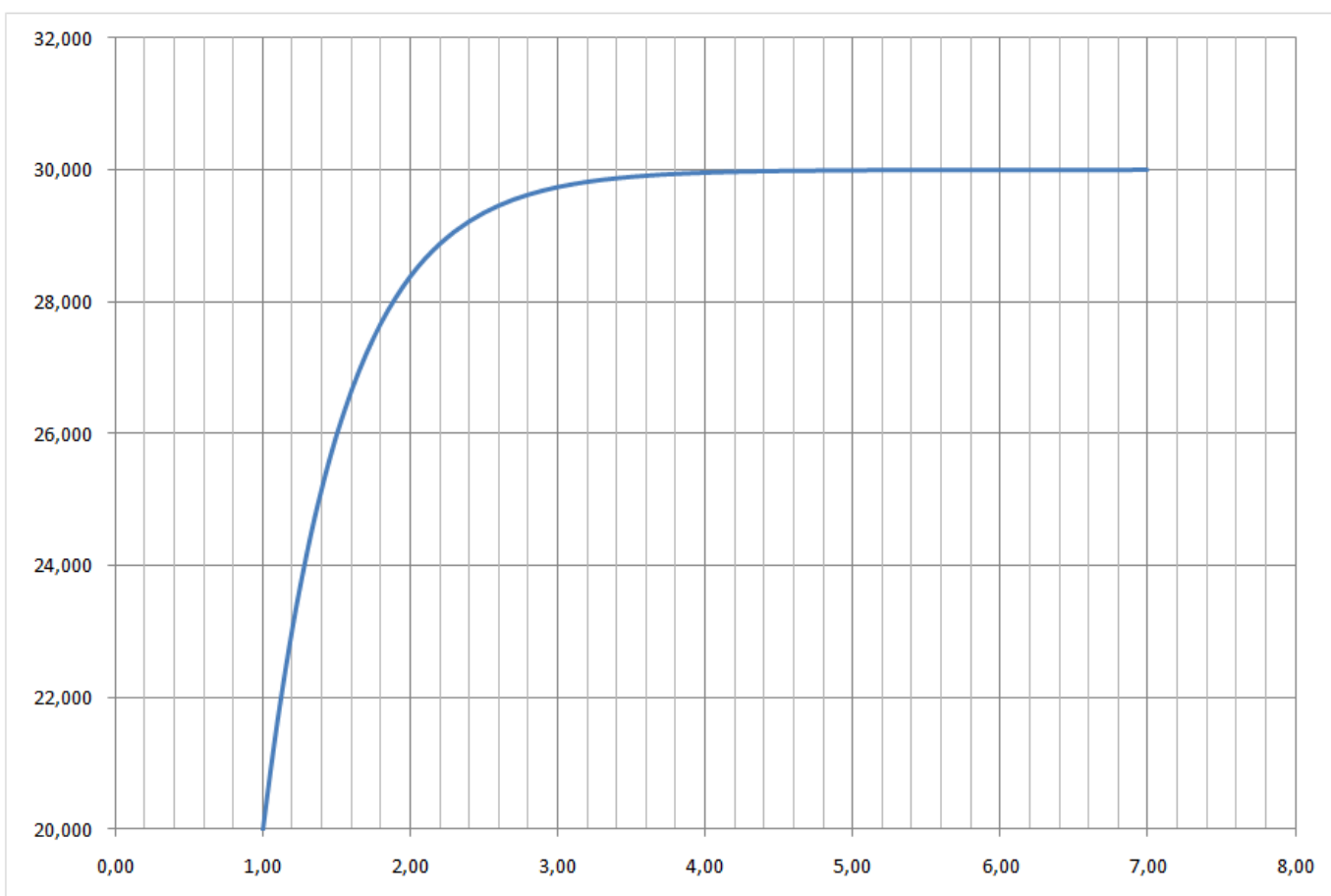
$X_{sp}= 30$        $K_p= 1$        $K_i= 0$        $K_d= 0$   
 $V_{max} 12$        $dt= 0,1$

t [s]	Vm [V]	n° [giri/s]	s [mm]	vel.	accel.	errore			Vm_teor.	Vm_eff [V]
						err. [mm]	integrale	derivata		
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	

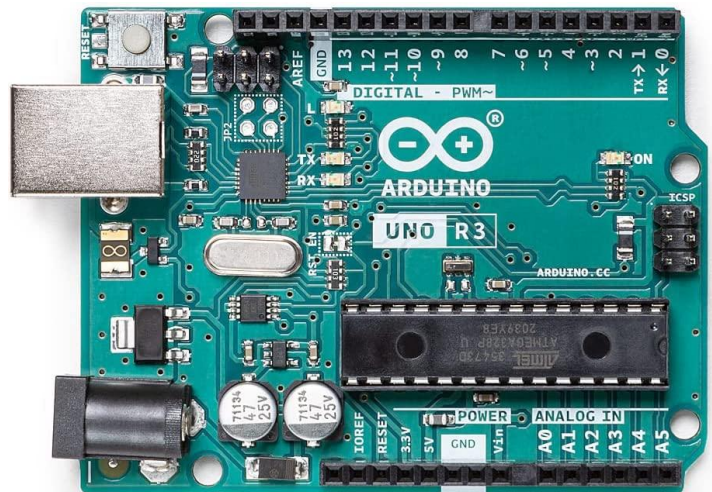
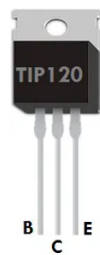
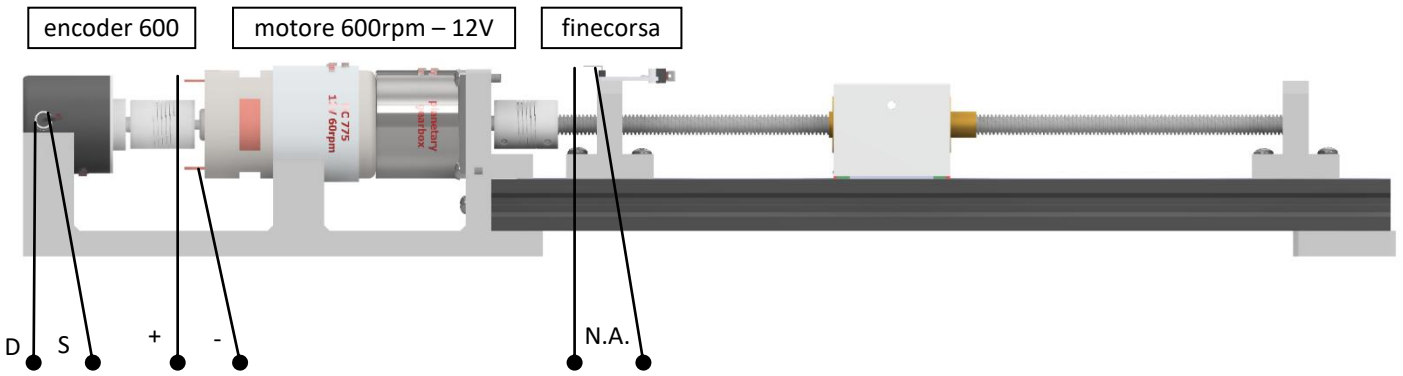
Con il PID il posizionamento si raggiunge in oltre 3 secondi con una  $a=-26 \text{ mm/s}^2$



Senza integrale e derivata dell' errore il posizionamento si raggiunge in oltre 4 secondi con una  $a=-33 \text{ mm/s}^2$

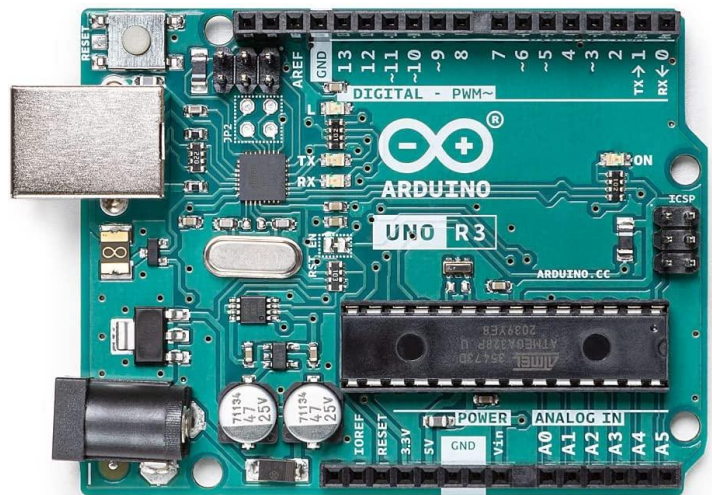
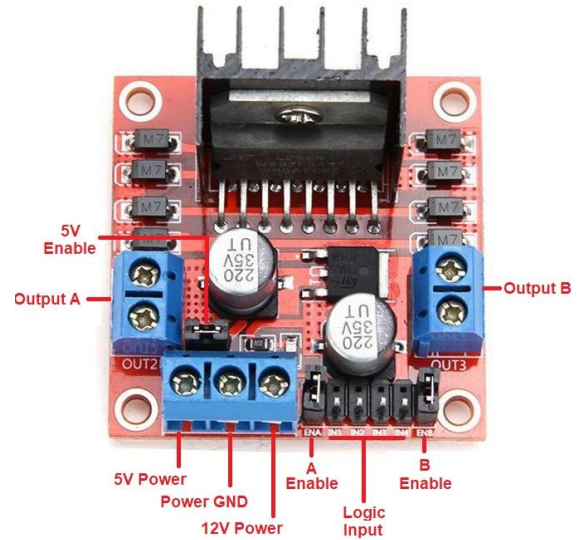
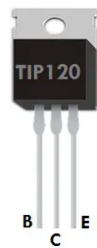
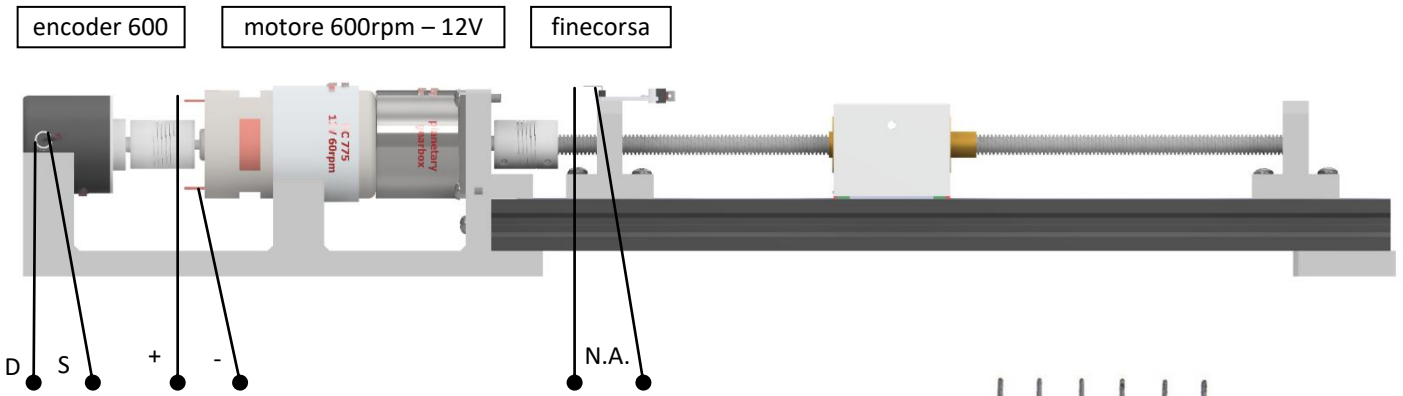


## SISTEMA DI CONTROLLO POSIZIONE CON ARDUINO E TRANSISTOR DI POTENZA TIP120



- 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).

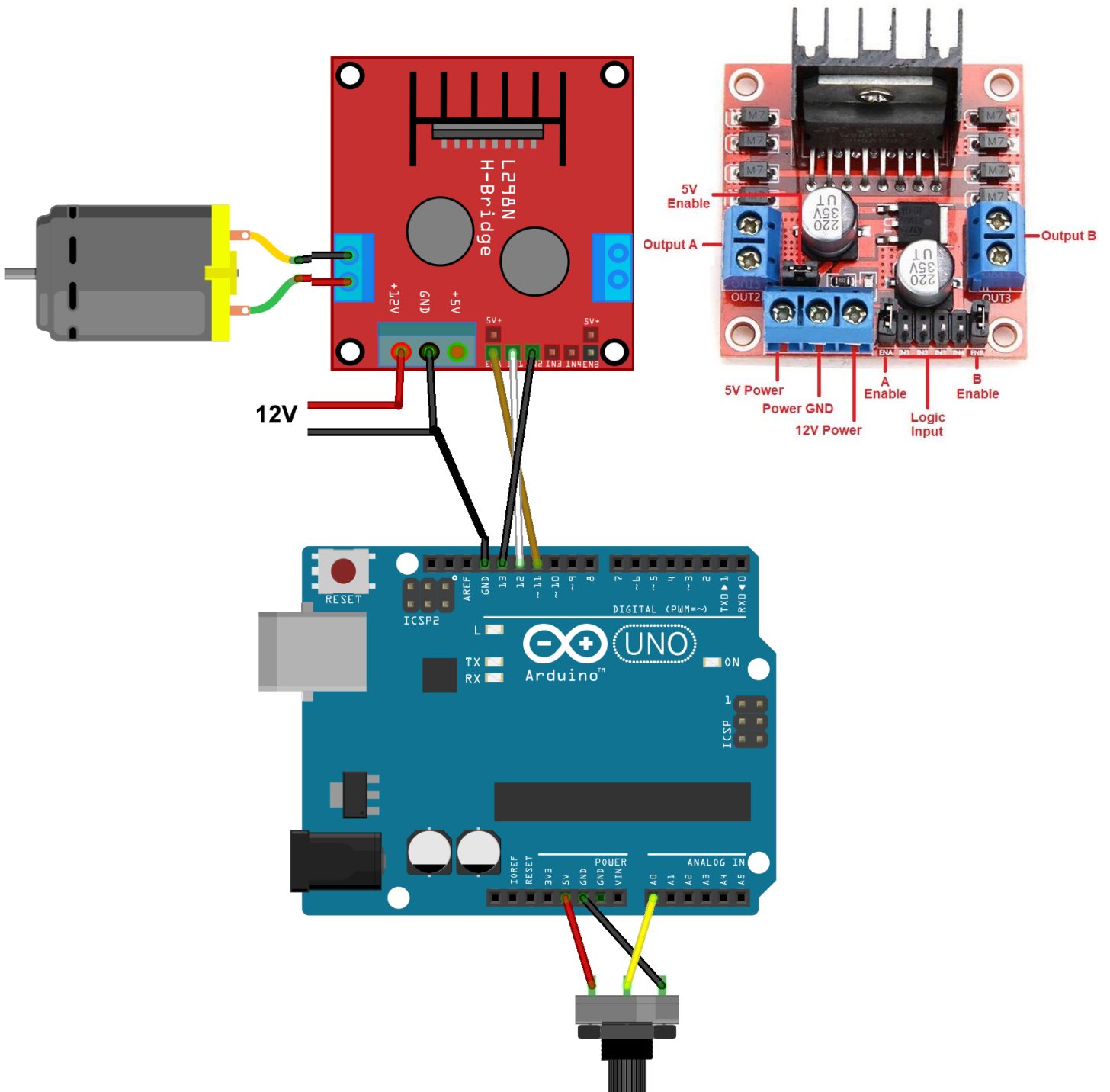
# SISTEMA DI CONTROLLO CON ARDUINO E TRANSISTOR DI POTENZA TIP120 E PONTE H L298N



- 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).

## ARDUINO E PONTE H L298N

Questa scheda controllo motori è basata sul driver Dual H-Bridge L298N e permette di pilotare con semplicità due motori DC 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.





## 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);
}
```

## CONTROLLO IN POSIZIONE E IN VELOCITA'

Lo schema usato è quello dei controllori in cascata.

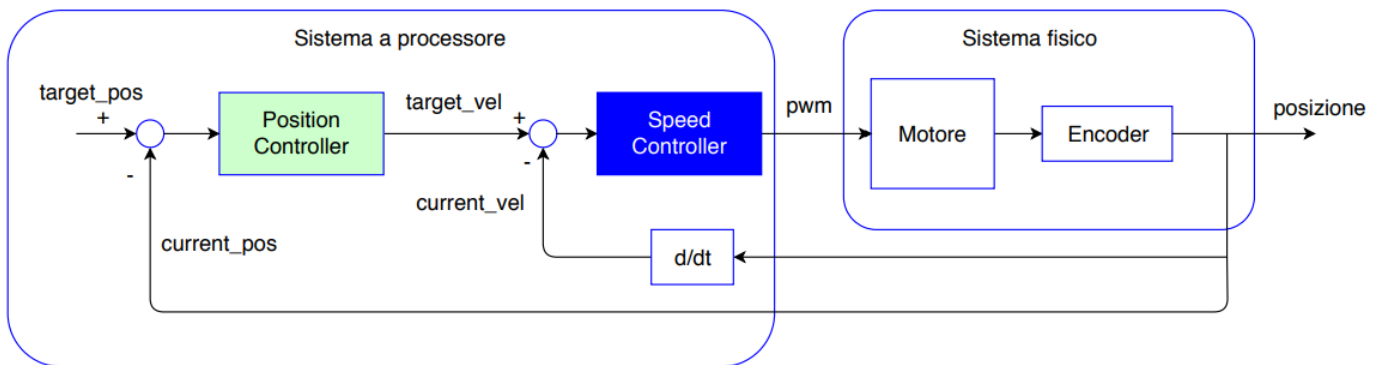
Un controllore di posizione più esterno (outer loop) fornisce il riferimento (di velocità) a un controllore di velocità interno (inner loop).

Concettualmente: se "siamo lontani" imponiamo una velocità alta che diminuisce man mano che ci si avvicina al target.

Praticamente usiamo un controllore proporzionale:

$$\text{target\_speed} = K_p * \text{error\_pos} = K_p * (\text{target\_pos} - \text{current\_pos}) \quad K_p = \text{coeff. proporzionale}$$

Tuttavia, oltre una certa velocità non potremo andare. Pertanto è necessario inserire una saturazione (limite sulla vel. massima).



Algoritmo del sistema di controllo

```
void loop(){
  current_pos = read encoder();
  current speed =  $\Delta \text{current pos} / \Delta t$  ;
  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(  $\Delta t$ );
}
```