Gestione di un motore passo-passo con Arduino

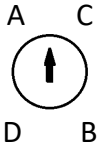
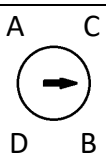
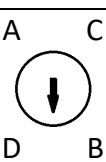
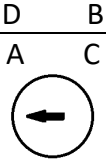
A cura del prof. Giuseppe SPALIERNO – 04/01/2012

In questa applicazione Arduino gestisce un motore passo passo a magnete permanente a 5 fili.
Scopo dell'esercitazione è fornire al motore passo-passo la sequenza di pilotaggio delle fasi A, B, C e D in funzionamento full-step. I controlli esterni sono:

- un deviatore S1 decide il verso di rotazione fornendo all'ingresso digitale 8 il potenziale 3,3V oppure 0V;
- un potenziometro R1 da 100KΩ, alimentato a 5V, fornisce all'ingresso analogico 0 un potenziale compreso tra 0 e 5V che Arduino trasforma in numero compreso tra 0 e 1023 che rappresenterà il ritardo in millisecondi tra una configurazione di pilotaggio delle fasi e la successiva.

Cenni sul principio di funzionamento del motore passo passo a magnete permanente

Supponendo, per comprensione del principio, che il motore passo passo abbia un rotore a magnete permanente con una sola coppia polare (Nord e Sud) che si orienta verso la fase alimentata, si comprende la seguente tabella operativa nota come "Full-Step" (passo intero) nella quale vengono alimentate due fasi alla volta. Il rotore si posiziona in una stato intermedio tra le due fasi alimentate. In questo caso il passo è di 90°.
Nella prima riga le fasi alimentate sono la A e la C per cui il rotore si posiziona, in figura, verso l'alto. Infatti la corrente che scorre nell'avvolgimento A crea un campo magnetico con direzione da A verso B; la corrente che scorre nell'avvolgimento C crea un campo magnetico con direzione da C verso D.
Poiché il campo magnetico è una grandezza vettoriale, la somma dei due campi magnetici ha direzione dall'alto al basso che è in grado di attrarre il rotore verso l'alto come appare nella figura della prima riga.

A	B	C	D	POSIZIONE ROTORE
1	0	1	0	
0	1	1	0	
0	1	0	1	
1	0	0	1	

Realizzando il rotore con più coppie polari sfasate di posizione tra di loro si dimostra che il passo è dato dalla relazione: **passo** = **360°**/(**n**·**p**) ove **n** è il numero delle fasi e **p** è il numero di coppie polari.
Se **n** = **4** (le fasi A, B, C e D) e **p** = **12** (coppie polari) il passo vale: **passo** = **360°**/(**4**·**12**) = **7.5°**

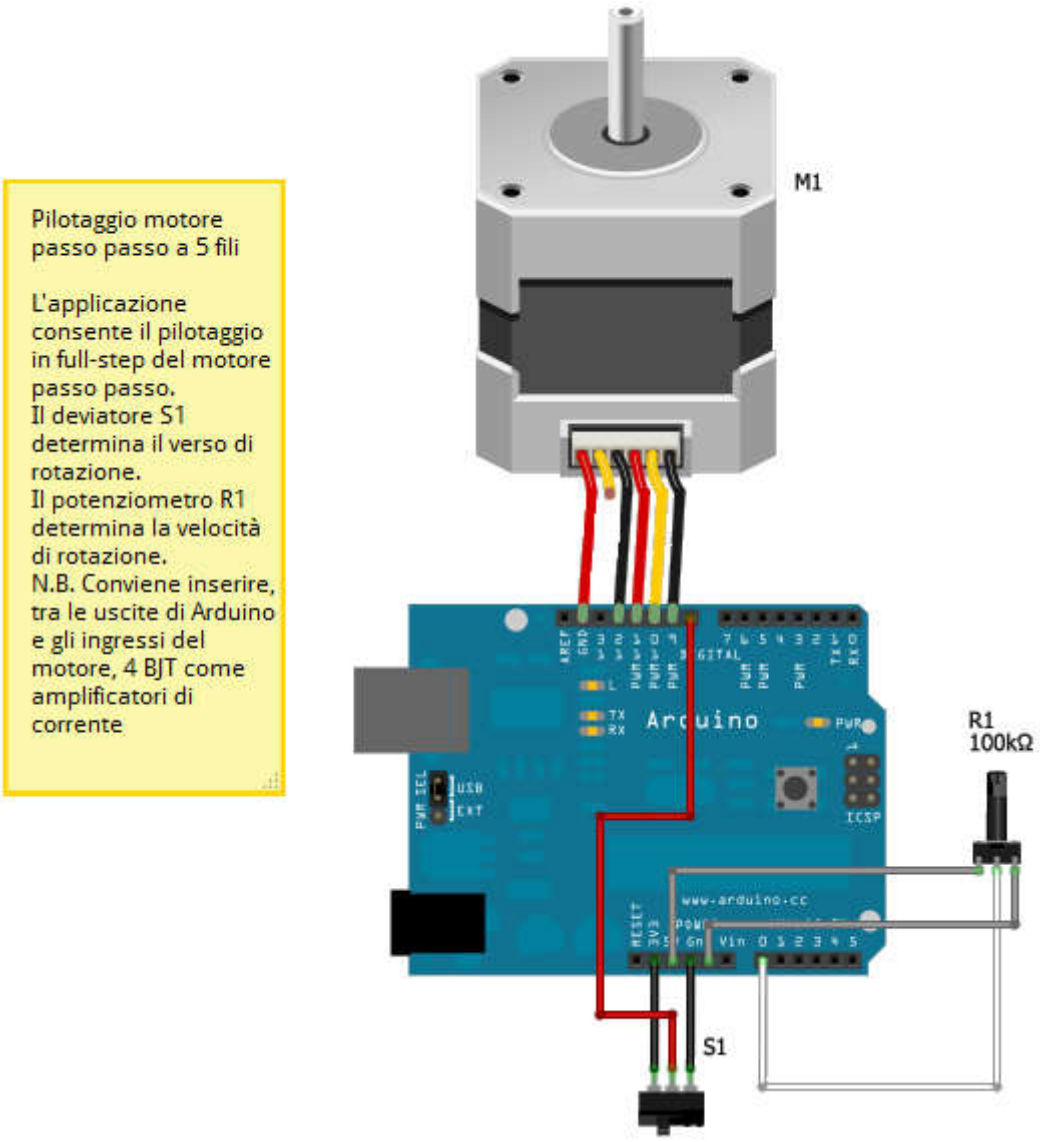
Collegamento dell'hardware

Le fasi A, B, C e D sono state collegate, rispettivamente, ai pin 12, 11, 10 e 9 delle linee "DIGITAL" di Arduino. Il cavetto comune è stato posto a massa (GND).

Il centrale di S1 è stato collegato al pin 8, definito come ingresso digitale, ed il centrale del potenziometro R1 è stato collegato all'ingresso analogico 0.

Il pin 13 digitale è stato definito di uscita in modo da segnalare, con il relativo LED presente su Arduino, se il verso di rotazione è orario (LED spento) o antiorario (LED acceso).

Il motore, il deviatore ed il potenziometro sono stati collegati come nella seguente figura:



N.B. Si suggerisce di collegare le linee digitali di uscita di Arduino pin 12, 11, 10 e 9 ad un transistor NPN come amplificatore di corrente in grado di pilotare correttamente le quattro fasi del motore.

Descrizione del software

Di seguito si riporta il listato del programma sviluppato in linguaggio C con Arduino 1.0.

Nelle prime istruzioni sono stati definiti gli indirizzi dei pin dei sensori e dalle fasi del motore.

Nel blocco "**void setup()**" sono state definite le direzioni delle linee utilizzate (OUTPUT e INPUT) e la velocità della comunicazione seriale tra Arduino ed il PC.

Le istruzioni più significative del blocco "**void loop()**" leggono il valore del potenziometro tra 0 e 1023 e lo stato dell'interruttore HIGH oppure LOW.

Tali valori sono stampati sul monitor del PC se Arduino è ad esso collegato.

Le successive istruzioni consentono di scrivere su ciascuna delle quattro linee collegate alle fasi del motore, gli stati logici corrispondenti alle sequenze da rispettare affinché il rotore del motore giri in senso orario o antiorario.

Dopo ogni sequenza si attende un ritardo, in millisecondi, stabiliti dalla istruzione "**delay(Valore Sensore)**".
Nel caso di impostazione del massimo ritardo (1023, poco più di un secondo) l'aggiornamento del prossimo ritardo impostato e del verso di rotazione avviene dopo più di quattro secondi.
Il minimo ritardo si ha impostando la variabile **ValoreSensore** al valore zero. In tal caso il ritardo tra la successione di quattro sequenze e la successiva dipende dal tempo di esecuzione del gruppo di istruzioni che compongono il ciclo che verrà ripetuto ed è comunque dell'ordine di alcune decine di microsecondi.
Sperimentalmente si è giustamente notato che dopo un certo aumento di velocità di rotazione, il motore perde il passo, non riuscendo per inerzia a restare sincronizzato, e quindi si ferma.
In tal caso potrà essere conveniente o inserire in serie al potenziometro un resistore di resistenza fissa minima che assicuri un potenziale minimo diverso da zero che consenta la rotazione del motore alla sua massima velocità (soluzione hardware) oppure controllando che il valore della variabile **ValoreSensore** sia maggiore o uguale ad valore minimo (ad esempio 20 verificato sperimentalmente). In caso negativo tale variabile sarà forzata a tale valore minimo.

Listato del programma

```
// Motore passo passo 04-01-2012

/* seleziona il pin di ingresso
per il potenziometro  */
int PinSensore = A0;

// seleziona il pin di uscita per il LED
int PinLed = 13;

/* variabile per memorizzare
il valore proveniente dal sensore */
int ValoreSensore = 0;

// indirizzi dei bit delle fasi e interruttore
int PinFaseA = 12;
int PinFaseB = 11;
int PinFaseC = 10;
int PinFaseD = 9;
int PinInterruttore = 8;

// variabile per stabilire il verso di rotazione
int ValoreInterruttore;

void setup() {
  pinMode(PinLed, OUTPUT);
/*  LED su pin 13 acceso se
il verso di rotazione è antiorario  */

  pinMode(PinFaseA, OUTPUT);
  pinMode(PinFaseB, OUTPUT);
  pinMode(PinFaseC, OUTPUT);
  pinMode(PinFaseD, OUTPUT);
  pinMode(PinInterruttore, INPUT);
  Serial.begin(9600);
  }

void loop() {
  //legge il valore tra 0 e 5V dal potenziometro
  ValoreSensore = analogRead(PinSensore);
  Serial.print("Valore letto = ");
  Serial.print(ValoreSensore);
ValoreInterruttore=digitalRead(PinInterruttore);
  if(ValoreInterruttore == HIGH){

//ROTAZIONE ANTIORARIA FULL STEP
   Serial.println("   ROTAZIONE ANTIORARIA");
   digitalWrite(PinLed, HIGH);

// 1010 (10)
   digitalWrite(PinFaseA,HIGH);
   digitalWrite(PinFaseB,LOW);
   digitalWrite(PinFaseC,HIGH);
   digitalWrite(PinFaseD,LOW);
   delay(ValoreSensore);

   // 0110 (6)
   digitalWrite(PinFaseA,LOW);
   digitalWrite(PinFaseB,HIGH);
   digitalWrite(PinFaseC,HIGH);
   digitalWrite(PinFaseD,LOW);
   delay(ValoreSensore);

    // 0101 (5)
   digitalWrite(PinFaseA,LOW);
   digitalWrite(PinFaseB,HIGH);
   digitalWrite(PinFaseC,LOW);
   digitalWrite(PinFaseD,HIGH);
   delay(ValoreSensore);

   // 1001 (9)
   digitalWrite(PinFaseA,HIGH);
   digitalWrite(PinFaseB,LOW);
   digitalWrite(PinFaseC,LOW);
   digitalWrite(PinFaseD,HIGH);
   delay(ValoreSensore);
   }
  else {

//ROTAZIONE ORARIA FULL STEP
   Serial.println("   ROTAZIONE ORARIA");
   digitalWrite(PinLed, LOW);

// 1001 (9)
   digitalWrite(PinFaseA,HIGH);
   digitalWrite(PinFaseB,LOW);
   digitalWrite(PinFaseC,LOW);
   digitalWrite(PinFaseD,HIGH);
   delay(ValoreSensore);

   // 0101 (5)
   digitalWrite(PinFaseA,LOW);
   digitalWrite(PinFaseB,HIGH);
   digitalWrite(PinFaseC,LOW);
   digitalWrite(PinFaseD,HIGH);
   delay(ValoreSensore);

   // 0110 (6)
   digitalWrite(PinFaseA,LOW);
   digitalWrite(PinFaseB,HIGH);
   digitalWrite(PinFaseC,HIGH);
   digitalWrite(PinFaseD,LOW);
   delay(ValoreSensore);

   // 1010 (10)
   digitalWrite(PinFaseA,HIGH);
   digitalWrite(PinFaseB,LOW);
   digitalWrite(PinFaseC,HIGH);
   digitalWrite(PinFaseD,LOW);
   delay(ValoreSensore);
  }

}
```