

ELS (Electronic Lead Screw) di McMax
Progetto disponibile su Meccanica & Dintorni
<http://meccanicaedintorni.morpe.it>

DESCRIZIONE DEL PROGETTO

Il sistema ELS è un automatismo che controlla l'avanzamento del carro del tornio, sia per mezzo della vite madre che per mezzo della barra scanalata, in modo che sia sincrono con il movimento del mandrino. Per mezzo di questo sistema è possibile sia far avanzare il carro ad una data velocità per la passata di tornitura, sia filettare senza dover cambiare le ruote o senza dover agire sulla scatola norton.

Il sistema è stato progettato per un tornio CASER 400, dotato sia di vite che di barra scanalata e con una scatola norton a 4 rapporti che agisce sulla sola barra. E' possibile adattare il sistema a qualsiasi tornio impostando opportunamente i parametri.

Elenco caratteristiche implementate nel software rev 1.6:

- **Impostazione parametri tra cui:**
 - Passi dell'encoder al mandrino
 - Passi del motore stepper alla vite
 - Passo della vite madre (in mm)
 - Senso di rotazione standard della vite (in base anche a come viene montato il motore stepper)
 - Ritardi di accelerazione e decelerazione del motore stepper
 - Avanzamenti per 4 rapporti di scatola norton, sia per il carro che per il trasversale (per torni dotati di avanzamento trasversale)
 - Offset di spostamento per filettatura (per compensare eventuali laschi)
 - Velocità massima ammissibile per il motore stepper
- **Avanzamento vincolato**
 - lunghezza dell'avanzamento impostabile in mm (risoluzione 1 centesimo) con limite destro e sinistro dell'avanzamento
 - Ritorno veloce
 - Impostazione della velocità di avanzamento in mm/min o cent/g modificabile in corso di esecuzione
 - Visualizzazione della velocità di rotazione del mandrino in giri al minuto
 - Visualizzazione della posizione in tempo reale ed in corso di esecuzione.
- **Avanzamento libero**
 - Impostazione della velocità di avanzamento in mm/min o cent/g modificabile in corso di esecuzione
 - Visualizzazione della velocità di rotazione del mandrino in giri al minuto
 - Visualizzazione della posizione in tempo reale ed in corso di esecuzione.
 - Ritorno veloce
- **Filettatura**
 - Impostazione del passo (mm e TPI) e dello sviluppo del filetto
 - Filettatura a misura con impostazione della lunghezza (in mm, risoluzione 0,1mm) e ritorno veloce senza svincolo della vite madre e senza inversione del mandrino
 - Filettatura con "vincolo meccanico" che ricrea l'esatta condizione che si presenta con filettatura per mezzo di ruote. La vite resta vincolata al mandrino come se ci fosse un reale vincolo meccanico.

- Movimento libero del carro con velocità proporzionale allo spostamento del joystick
- Visualizzazione della posizione angolare del mandrino con risoluzione 0,36°
- Visualizzazione della velocità di rotazione del mandrino con risoluzione 1 giro al minuto

HARDWARE

processore: [Arduino UNO](#) (ATmega328P) – 32KBflash, 2KB RAM, 1KB EEPROM – 16Mhz

display: 20x4 alfanumerico [JHD 204](#) (controller HD44780)

Interfaccia di controllo: [Sparkfun Joystick shield kit](#)

Stepper controller/driver: [2M982](#)

Motore stepper: Oriental Motors [PK299E4.5A](#) – 1.8°/passo (200 passi/giro)

Encoder: ELCIS mod. Z63-250-5-B-N-CM-R (fuori produzione)

Alimentatore: Vin 230Vac; Vout 48VDC; Iout: 7A min.

Alimentatore: Vin 230Vac; Vout 12V; Iout 500mA

NOTE SULLA SCELTA DEI COMPONENTI

Qui sopra ho indicato i componenti da me utilizzati ma ovviamente alcune parti possono essere cambiate a piacimento in base alle proprie esigenze. Vediamo in dettaglio.

PROCESSORE

Per poter replicare il progetto sfruttando il codice già scritto è necessario che il processore sia esattamente quello indicato, ovvero ARDUINO(GENUINO) UNO. E' reperibile online direttamente sul sito <http://www.arduino.cc>. Si trovano anche cloni a prezzi inferiori anche se non ne ho mai provati quindi non garantisco il corretto funzionamento.

DISPLAY

Lo stesso vale per il display. Va bene di qualsiasi marca ma è fondamentale che sia un 20x4 (20 caratteri su 4 righe) altrimenti il software risulterà inutilizzabile.

INTERFACCIA DI CONTROLLO

Questa scheda è uno "shield" da montare su ARDUINO (o su un clone) che da a disposizione un joystick analogico su 2 assi con pulsante integrato più altri 4 pulsanti aggiuntivi. Poiché 2 di questi tasti (il tasto del joystick + uno della tastiera a parte) sono stati previsti su linee I/O di ARDUINO che vengono utilizzate come interrupt per gli ingressi encoder, le rispettive piste vanno tagliate per evitare interferenze nella corretta esecuzione del programma.

In alternativa è possibile utilizzare tasti e joystick sperati avendo l'accortezza di rispettare i collegamenti che riporterò di seguito. E' chiaro che l'utente esperto può anche riconfigurare i pins nel software a piacimento avendo solo l'accortezza di utilizzare gli ingressi INTERRUPT (pins 2 e 3 di ARDUINO, rispettivamente INT0 e INT1) SOLO ED ESCLUSIVAMENTE per i segnali provenienti dall'encoder.

STEPPER CONTROLLER

Ho deciso di utilizzare un controller stepper piuttosto che farlo fare al micro in quanto le risorse di ARDUINO sono già utilizzate appieno e la parte di controllo del motore avrebbe limitato parecchio le funzionalità del sistema. Utilizzando il controller esterno, ARDUINO si preoccuperà soltanto di inviare i segnali STEP, DIR ed ENABLE.

Il controllore deve essere idoneo al motore stepper che si decide di utilizzare. Consiglio sempre di prevedere un controllore che abbia una corrente del 20/30% più alta rispetto a quella richiesta dal motore.

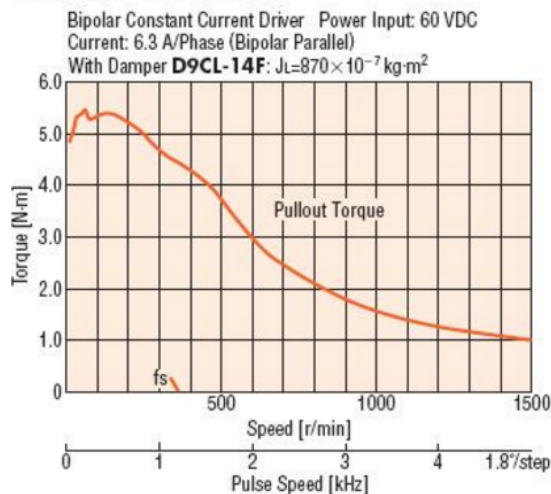
MOTORE STEPPER

Questo va previsto in modo che la coppia erogata sia in grado di muovere agevolmente la vite del tornio anche sotto sforzo.

Speed – Torque Characteristics

PK299-E4.5A/PK299-E4.5B

Bipolar (Parallel)



L'esecuzione di una filettatura con l' ELS richiede una precisione assoluta del passo, pena una filettatura non precisa e con passo non regolare..... infatti, quando il motore fa troppa fatica e la coppia erogata non è sufficiente, "perde il passo". la coppia dei motori stepper è inversamente proporzionale alla velocità a cui li si fa ruotare:

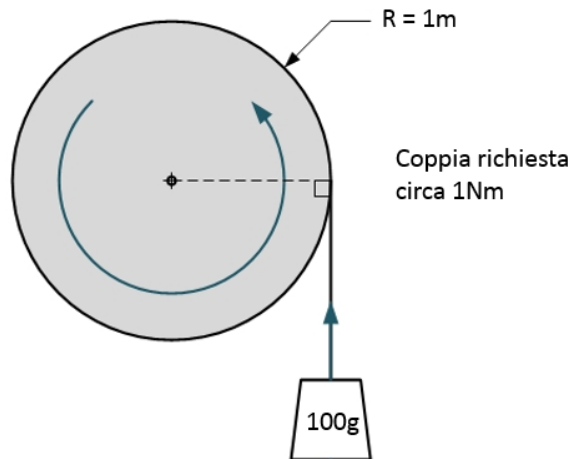
Nella figura accanto si vede la curva di coppia del motore utilizzato nel mio sistema. Si nota che la coppia massima di 5Nm è disponibile da 0 a ~250

g/min. Aumentando la velocità la coppia diminuisce fino a raggiungere un livello tale da non essere nemmeno in grado di mantenere il motore in rotazione nemmeno in assenza di carico meccanico.

Per l'operazione di filettatura la velocità di rotazione richiesta dal motore è molto bassa, sia perché l'operazione è eseguita in modo lento ma anche perché, in genere, il passo della vite è sempre abbastanza lungo (>3mm) quindi ad una singola rotazione del motore lo spostamento apprezzabile del carro è notevole. Se nel tornio però è presente anche una barra scanalata per gli avanzamenti, la velocità di rotazione richiesta al motore è più elevata in quanto la barra presenta una demoltiplica maggiore.

Per evitare problemi e sfruttare al meglio le potenzialità del sistema il consiglio è di utilizzare il motore configurato in bipolare e con fase in parallelo (dove previsto). La velocità massima consigliata non deve eccedere i 500 g/min.

La scelta del motore è fondamentale ai fini del corretto funzionamento dell'intero sistema. Se il motore stepper non ha potenza sufficiente, questi "perderà il passo" frequentemente inserendo imprecisioni tali da pregiudicare il funzionamento del sistema. E' bene scegliere un motore stepper che sia in grado di muovere il carro, sia in filettatura che avanzamento, senza problemi. Esistono formule matematiche anche complesse per il calcolo ma si può tranquillamente affidarsi al buon senso ed a prove empiriche. Non è mia intenzione spiegare questo concetto in modo dettagliato, in primis perché non posseggo le competenze per farlo e poi perché, come detto sopra, ce la si può cavare benissimo col buon senso senza dover scomodare complessi calcoli matematici. Ritengo però importante dare le basi del concetto in modo che si abbia una conoscenza sufficiente a scegliere il motore corretto per le proprie esigenze.



Per prima cosa è necessario capire qual è il parametro fondamentale di questi motori: la coppia!. Questa è espressa in Nm (Newton metro) o, se indicati con sistema "imperial", Oz-in (Once Pollice). In pratica si indica il carico che il motore riesce a spostare ad una data distanza dal centro di rotazione. Dunque, per farla semplicissima (...e spero di riuscirci), **un motore in grado di erogare una coppia di 1Nm, è in grado di spostare un carico da ca. 100g posizionato esattamente ad 1 metro dal centro di rotazione.**

La figura accanto mostra la schematizzazione di questo concetto. Se però riduciamo il raggio ed avviciniamo il carico al centro di rotazione, lo stesso

motore potrà muovere un carico maggiore, il cui valore è inversamente proporzionale alla distanza. In pratica, con raggio 1 metro abbiamo 100g, con raggio 100mm avremo 1Kg e con raggio 10mm avremo 10Kg!

Prendendo in esame il motore da me scelto, questo ha una coppia di ca. 5Nm per velocità compresa tra 0 e 250 g/min. Quindi sarà in grado di spostare un peso di ca. 500g posizionato ad 1 metro dal centro rotazione. La puleggia che utilizzo per trasmettere il moto alla vite ha però un diametro di ca. 50mm, quindi un raggio di 25mm, e ciò significa che, a quella distanza, il motore sarà in grado di muovere un peso di ca. 20Kg! Ma la puleggia agirà su una vite con diametro ca. 20mm, quindi raggio 10mm; ed a quella distanza la coppia aumenta ancora fino a 50Kg! Così, ad occhio, mi paiono sufficienti!

ENCODER

Questa è una parte importante del sistema al pari del motore stepper. Quello che ho utilizzato è di buona fattura, italiano (marca ELCIS), ma purtroppo fuori produzione. Ci sono tantissimi altri modelli idonei sul [catalogo ELCIS](#) ma hanno costi abbastanza elevati. Alternativamente si possono acquistare su ebay dalla Cina a prezzi contenuti.

Le caratteristiche che deve avere sono:

- Tipo encoder: incrementale con interfaccia in quadratura
- Numero passi: compreso tra 250 e 500.

Il sistema prevede i soli ingressi in quadratura (A e B) e non l'ingresso INDEX. Se l'encoder prevede l'uscita INDEX questa andrà lasciata scollegata.

Il software utilizza una routine di lettura dell'encoder basata sugli interrupt e moltiplica la risoluzione x4; in pratica, un encoder a 250 passi/giro, avrà una risoluzione reale di 1000 passi/giro. Il consiglio è di non esagerare con la risoluzione in quanto, più sarà elevata, più lenta sarà l'esecuzione del software e la possibilità di perdere dei passi. La risoluzione è sicuramente importante e molti saranno portati a pensare che un encoder con 600 passi/giro (che x4 diventano 2400) sia meglio di uno da 250; a livello "accademico" ciò è senz'altro vero, ma il beneficio in termini di risoluzione ha bisogno di molta più potenza di calcolo per essere gestito quindi, in caso di perdita massiccia di passi, tutti i benefici sono vanificati fino al punto in cui il sistema potrebbe addirittura diventare inutilizzabile. A conti fatti, un encoder da 250 passi/giro (quindi 1000 passi nel software) avrà una risoluzione angolare di $0,36^\circ$ che sono più che sufficienti per eseguire una filettatura di ottima qualità.

ALIMENTAZIONE

Sono necessari 2 alimentatori: uno di potenza, preferibilmente con uscita 48V o superiore, e comunque scelto in base alle specifiche del motore stepper e del controller, l'altro con uscita compresa tra 9 e 12V e corrente nell'intorno dei 500mA, utilizzato per alimentare ARDUINO e tutta la parte di controllo.

Non esistono altre regole per la scelta di questi alimentatori; io ho utilizzato 2 alimentatori switching:

- 1- per l'alimentazione di ARDUINO, un vecchio alimentatore per PC portatile con uscita a 12V. ARDUINO accetta alimentazione fino a 20V e oltre ma è consigliabile non eccedere i 12V in quanto le regolazioni interne alla scheda avvengono per mezzo di regolatori lineari che tendono a scaldare parecchio man mano che la tensione di ingresso sale.
- 2- Per l'alimentazione di potenza del motore stepper ho recuperato un alimentatore Switching open frame da 48V 10A di fattura italiana e di qualità. Sconsiglio l'utilizzo di alimentatori switching a basso costo in quanto potrebbero non essere in grado di erogare la corrente di spunto necessaria per il motore. Piuttosto è consigliabile utilizzare un alimentatore lineare con trasformatore, ponte raddrizzatore e condensatori. I controller stepper in genere accettano tensioni anche fino a 60/80V quindi non sarà necessaria un'uscita regolata. Alternativamente sono disponibili controllori stepper che integrano l'alimentazione e quindi possono essere collegati direttamente alla rete elettrica: il costo è maggiore ma si semplifica notevolmente il sistema. Sicuramente da valutare quando acquistate il controllore stepper.

Considerazioni sulla scelta delle risoluzioni: motore Stepper ed Encoder

La risoluzione scelta per il motore stepper (in termini di passi/giro) determina il passo massimo eseguibile sul tornio. La formula è la seguente:

$$\text{max. passo} = \frac{\text{passo vite madre}}{\text{passi motore stepper}} \times 800$$

I motori standard hanno risoluzione base di 200 passi/giro ma questa è con passo pieno (full-step): i controllori sono sempre in grado di gestire il mezzo passo (half-step), il quarto di passo e spesso anche i "micropassi" fino ad avere una risoluzione anche superiore ai 10000 passi/giro su motori standard da 200.

Nel progetto è stato previsto il pilotaggio del motore in "half-step" (mezzo passo) quindi con risoluzione 400 passi/giro. Il limite sulla scelta è imposto dal processore. Il numero costante 800 nella formula rappresenta la dimensione dell'array utilizzato nel codice per memorizzare i passi da eseguire.

E' possibile anche aumentare la risoluzione dello stepper ma, a quel punto, il passo massimo eseguibile sul tornio si ridurrà.

Per il mio tornio la vite è da 8 filetti per pollice (3,175mm) quindi, con stepper configurato a 400 passi/giro, il massimo passo che posso eseguire sarà di 6,35mm, ovvero esattamente il doppio rispetto al passo della vite. Se volessi ad esempio provare a scendere con la risoluzione dello stepper al "full-step", quindi 200 passi/giro, perderei la risoluzione ma aumenterei il massimo passo tornibile ed in particolare a ben 12,7mm! Se invece volessi aumentare la risoluzione ad esempio a 800 passi/giro, diminuirei il massimo passo a 3,175mm, ovvero esattamente il passo della vite.

La tabella qui di seguito chiarisce meglio il concetto, le risoluzioni ottimali sono quelle evidenziate:

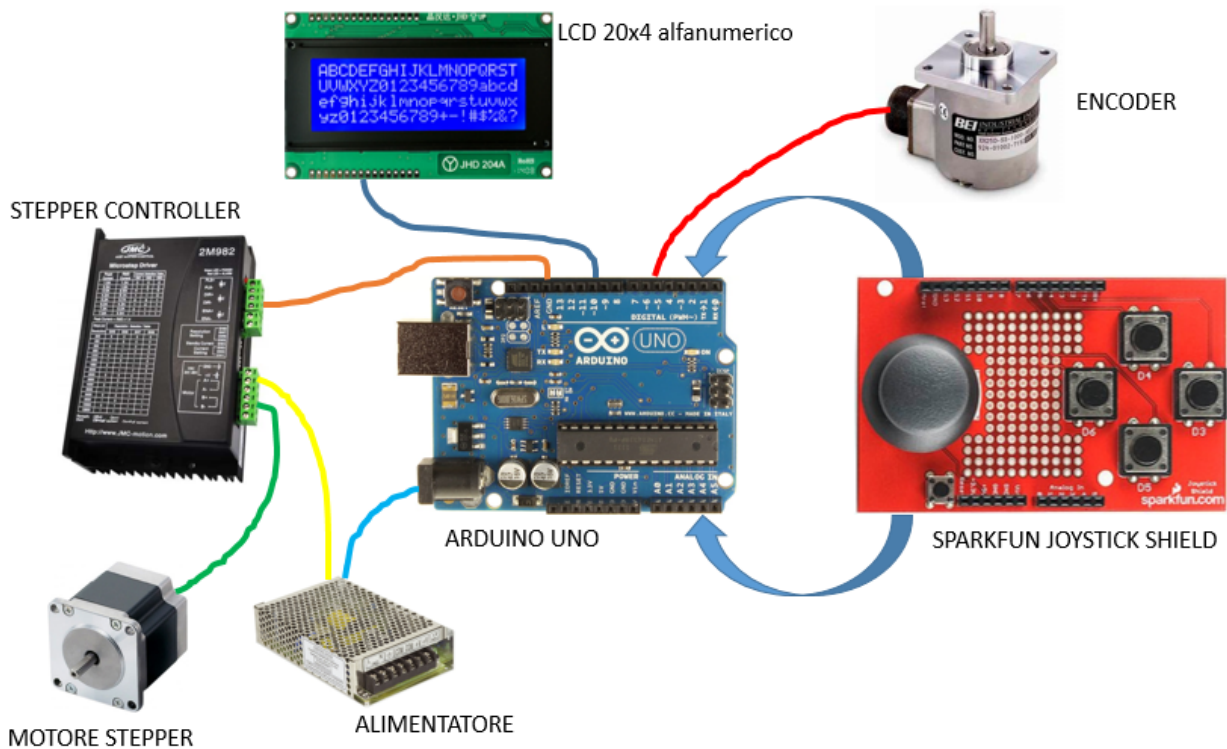
Passi/giro stepper	Max passo filettatura
200	4 x passo vite
400	2 x passo vite
800	1 x passo vite
1000	$\frac{4}{5}$ x passo vite
1600	$\frac{1}{2}$ x passo vite

Attenzione che la scelta della risoluzione del motore stepper deve essere specificata sia sul controller (in genere ci sono alcuni microswitch da impostare) che nel software. Nel manuale utente viene spiegato come cambiare la risoluzione nel software. Il parametro verrà salvato nella EEPROM interna al processore quindi non sarà necessario impostarlo ogni volta. Il valore standard previsto è di 400 passi/giro.

Il tutto è stato calcolato considerando che non ci sia alcuna demoltiplica meccanica tra lo stepper e la vite: in pratica la vite girerà esattamente come lo stepper. Nel caso in cui vogliate prevedere demoltipliche è sempre consigliabile utilizzare rapporti pari (1:2, 1:4). **La modifica del rapporto vite/stepper NON è prevista nel software** quindi chi la vuole implementare dovrà modificare il codice per adattarlo alla nuova configurazione. Posso eventualmente dare una mano in tal senso (postate la richiesta sul forum).

La scelta della risoluzione dell'encoder è invece pressoché obbligata ed in particolare sarà necessario un encoder il cui numero di passi sia maggiore o uguale a 800. Va sempre ricordato che i passi dell'encoder sono moltiplicati x4 all'interno del codice quindi, per avere 800 passi, sarà necessario un encoder da 200 passi/giro nominali. Il limite minimo è imposto sempre per via dell'array da 800 valori che viene utilizzato per scandire i passi del mandrino. L'encoder non potrà avere risoluzione minore rispetto al numero di passi stepper per giro mandino per ovvie ragioni (i passi sono sempre interi, non è possibile discriminare una frazione di passo).

SCHEMA A BLOCCHI DEL PROGETTO E ISTRUZIONI DI MONTAGGIO

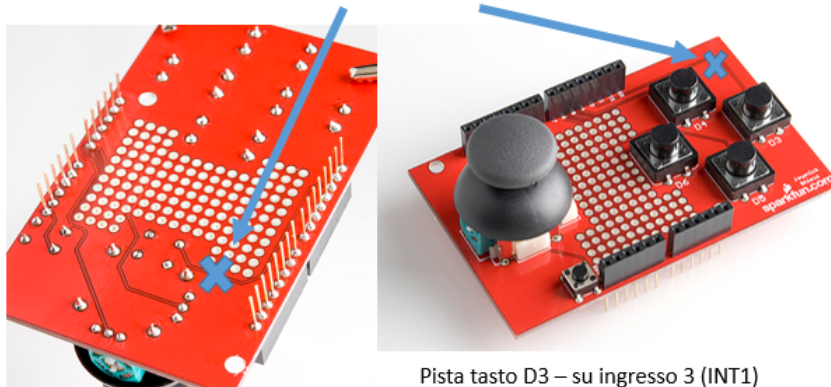


La figura mostra lo schema a blocchi del sistema completo.

Interfaccia di controllo

I controlli previsti dal software sono un Joystick a 2 assi analogico e 3 tasti (SEL, ESC, RESET). Se decidete di utilizzare il joystick shield SPARKFUN come quello in figura sarà necessario tagliare alcune piste per evitare interferenze dei tasti con gli ingressi encoder. Di seguito i dettagli:

TAGLIARE LE PISTE



Pista tasto joystick – su ingresso 2 (INT0)

Pista tasto D3 – su ingresso 3 (INT1)

Poiché questo shield viene solitamente fornito smontato, alternativamente potete evitare di montare i due tasti incriminati. Attenzione che i fori nel PCB sono metallizzati per cui anche il semplice inserimento del tasto nel PCB, senza questo venga saldato, può generare un contatto elettrico!

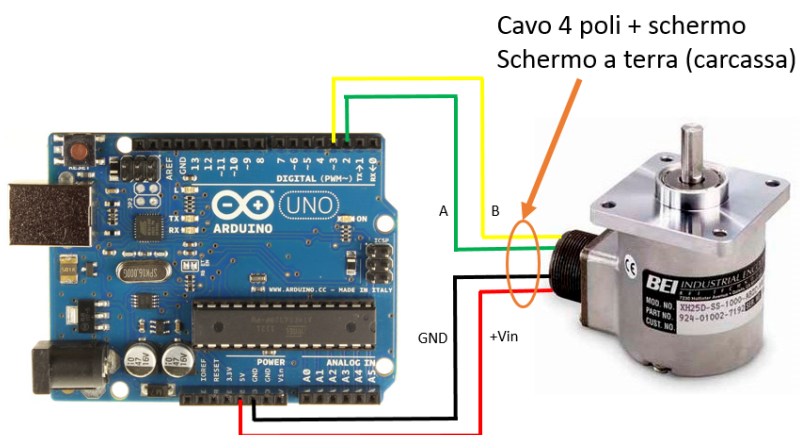
Encoder

gli encoder si presentano solitamente con 4 fili in uscita.

Vin è la tensione di alimentazione dell'encoder. Solitamente i 5V forniti da ARDUINO vanno bene ma bisogna verificare in base al tipo di encoder.

GND è chiaramente la massa.

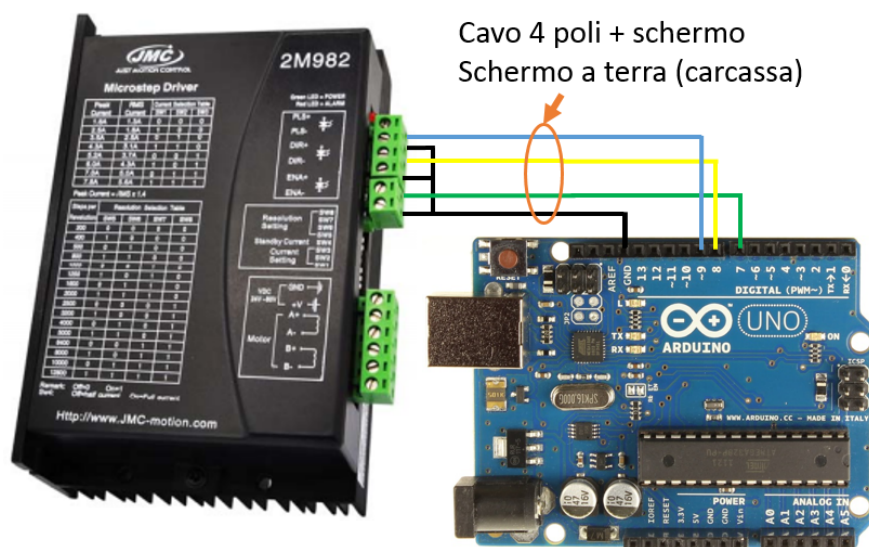
A e **B** sono i segnali in quadratura prodotti dall'encoder che andranno letti dal processore. **I colori dei cavi sono puramente indicativi.**



Supponendo che l'encoder necessiti di alimentazione a 5V, questa andrà collegata ad una delle prese a 5V presenti su ARDUINO. In caso contrario andrà prevista un'alimentazione ausiliaria per l'encoder. La massa dovrà essere in comune a quella di ARDUINO quindi, anche in caso di alimentatore ausiliario, la massa di quest'ultimo andrà collegata insieme a quella di ARDUINO. I segnali dell'encoder sono spesso soggetti a disturbi che possono causare pesanti malfunzionamenti all'intero sistema. Per questo motivo in genere i cavi in uscita dall'encoder, quando già previsti, sono schermati. Nel caso in cui il cavo non sia previsto sarà necessario recuperare un cavo schermato in grado di trasportare tutti i conduttori necessari + lo schermo in aggiunta. Lo schermo andrà collegato alla carcassa metallica più prossima che a sua volta andrà collegata alla terra dell'impianto di rete. Il consiglio è di fare in modo che tutte le carcasse metalliche del sistema (motore, dissipatore dell'alimentatore e dello stepper controller, encoder) siano elettricamente connesse al tornio il quale, a sua volta, sarà anch'esso collegato a terra.

Controllore motore stepper

Questo andrà collegato all'alimentazione di potenza con cavi di sezione adeguata. E' bene ridurre al minimo la lunghezza di questi cavi quindi è consigliabile alloggiare il controller



nella stessa scatola in cui si alloggerà l'alimentatore. Dal controller partiranno 4 cavi per il pilotaggio del motore stepper, anch'essi dovranno avere sezione adeguata viste le alte correnti in gioco. Anche per questo collegamento è bene ridurre al minimo la lunghezza del cavo. La polarità di collegamento delle fasi è poco importante in quanto influirà soltanto sul verso di rotazione. Nel caso in cui questo non sia corretto

basterà invertire la polarità di una delle fasi oppure, più semplicemente, correggere la polarità via software (spiegherò in seguito come fare).

I segnali di controllo che arrivano da ARDUINO sono 3: DIR, STEP, ENABLE.

Quasi tutti questi controllori hanno la parte di controllo isolata otticamente dal resto per evitare di avere la massa di potenza in comune con quella di segnale. In pratica ogni ingresso di controllo è un come se fosse un LED che va pilotato con una certa corrente. Poiché ARDUINO ha le uscite logiche a 5V, non sarà necessario interporre nulla tra le uscite di ARDUINO e gli ingressi del controller. Nel caso i segnali di controllo siano a tensione più alta si dovranno mettere delle resistenze in serie in modo da limitare la corrente nei LED.

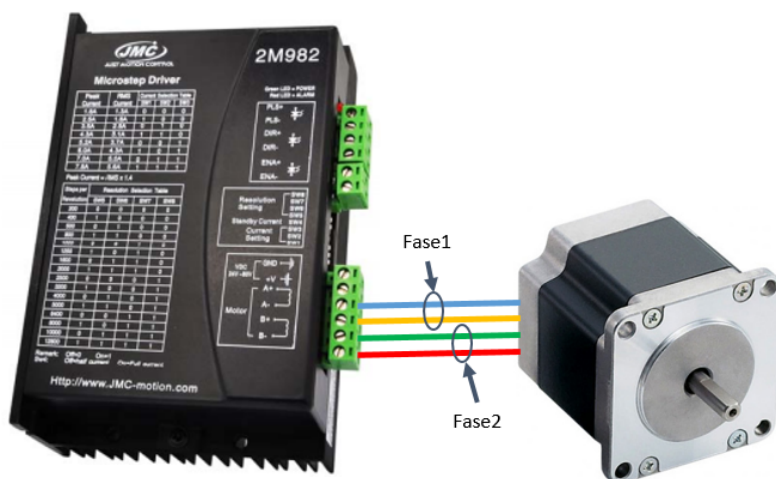
Nella figura qui sopra i dettagli del collegamento, i colori come al solito sono puramente indicativi. È consigliabile utilizzare un cavo schermato, avendo l'accortezza di mettere lo schermo a terra come già fatto per il cavo dell'encoder.

Motore Stepper

La scelta migliore è certamente su un motore a 4 fili, 2 fasi, da pilotarsi con segnale bipolare. Per bipolare si intende che la corrente può scorrere nelle fasi in entrambi i versi; in questo modo si sfruttano meglio le potenzialità del motore. I controller più comuni su questi livelli di potenza permettono già questo tipo di pilotaggio.

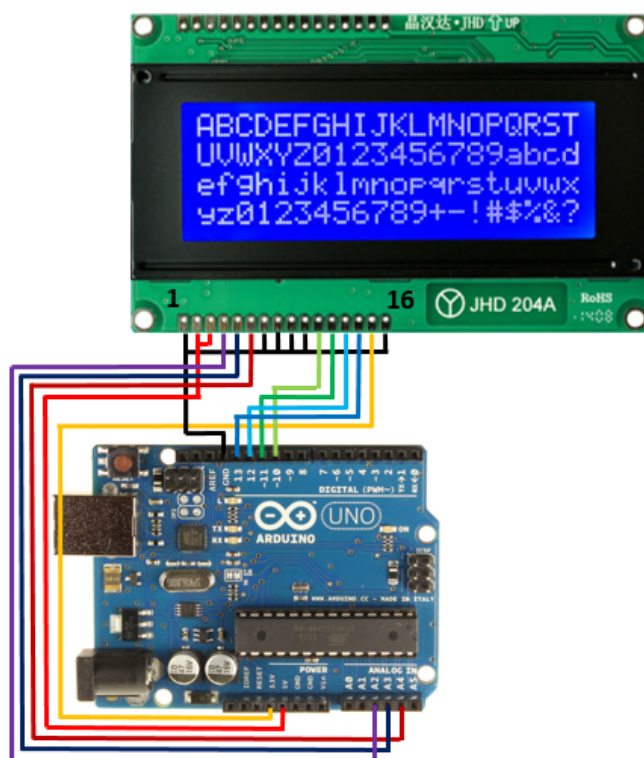
Il motore esce appunto con 4 fili, che andranno collegati direttamente alle uscite del controller. In alcuni casi i motori, soprattutto quelli di una certa potenza, escono su una morsetteria ad 8 poli in quanto prevedono 2 avvolgimenti per fase. Questo permette di configurare il motore sia con avvolgimenti in serie che in parallelo, e sfruttare quindi diversi livelli di potenza. Il motore configurato in serie svilupperà meno coppia ma richiederà anche meno corrente per funzionare. Il consiglio è quello di sfruttare al meglio il motore e di configurarlo con le fasi in parallelo.

Nella figura qui di seguito la schematizzazione del collegamento. I colori dei cavi sono puramente indicativi. Purtroppo non esiste una standardizzazione e bisogna fare riferimento al foglio tecnico del motore per i dettagli.



Display LCD

Per evitare pesanti modifiche al codice è consigliabile utilizzare un display 20x4 alfanumerico. Va bene un qualsiasi display purché abbia controller compatibile con Hitachi HD44780 (lo standard per definizione). Qui sotto lo schema dettagliato dei collegamenti e l'elenco dei pins.



PIN LCD	Funzione	PIN ARDUINO
1	VSS (GND)	GND
2	VCC (+5V)	5V
3	VEE (+5V*)	5V
4	RS	A2
5	R/W	A3
6	E	A4
7	DB0	GND**
8	DB1	GND**
9	DB2	GND**
10	DB3	GND**
11	DB4	10
12	DB5	11
13	DB6	12
14	DB7	13
15	LED+	3.3V
16	LED-	GND

(*) - Collegamento opzionale. In alcuni display è necessario un potenziometro per la regolazione del contrasto

(**) - Collegamento opzionale. Alcuni display permettono di lasciare questi 4 pins non connessi se inutilizzati

Alcuni display necessitano di un potenziometro sul pin VEE per la regolazione del contrasto. Ormai questa pratica è quasi del tutto superata e i nuovi modelli funzionano anche senza la regolazione. Nel caso in cui sia necessaria, va previsto un potenziometro da 5K o 10K ohm, i cui estremi vanno collegati tra VCC e VSS e il cursore su VEE.

Alimentazione

Per quanto riguarda l'alimentazione del controller stepper, come detto sopra, andrà previsto un alimentatore di potenza adeguata che andrà collegato direttamente al controllore.

Per quanto riguarda ARDUINO, questo risulterà alimentato dalla USB nel momento in cui lo collegheremo al PC ma andrà alimentato a parte una volta che il sistema sarà operativo e montato

sul tornio. Come detto serve un alimentatore con tensione di uscita compresa tra 9 e 12V e corrente di ca 500 mA. E' consigliabile prendere già alimentatore con la presa di alimentazione compatibile con ARDUINO: DC21MMX con positivo centrale.

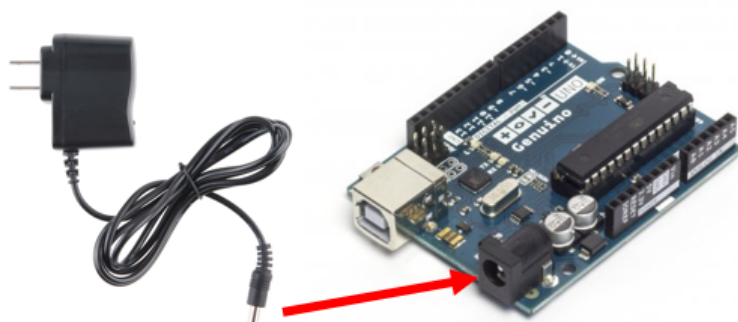


TABELLA COMPLETA DEI COLLEGAMENTI AD ARDUINO:

PIN#	Nome PIN	Direzione	Funzione assegnata	NOTE
0	RX	N/A	N/C	
1	TX	N/A	N/C	
2	INT0	INPUT	Ingresso ENCODER A	
3	INT1	INPUT	Ingresso ENCODER B	
4	Digital4	INPUT	TASTO ESC	Sparkfun Joystick shield D4
5	Digital5	INPUT	TASTO SEL	Sparkfun Joystick shield D5
6	Digital6	INPUT	TASTO RESET	Sparkfun Joystick shield D6
7	Digital7	OUTPUT	ENABLE motore stepper	
8	Digital8	OUTPUT	DIR motore stepper	
9	Digital9(OC1A)	OUTPUT	STEP motore stepper	Usato come PWM
10	Digital10	OUTPUT	LCD DB4	
11	Digital11	OUTPUT	LCD DB5	
12	Digital12	OUTPUT	LCD DB6	
13	Digital13	OUTPUT	LCD DB7	
A0	Analog0	INPUT	Joystick asse orizzontale	Sparkfun Joystick shield - asse X
A1	Analog1	INPUT	Joystick asse verticale	Sparkfun Joystick shield - asse Y
A2	DigitalA2	OUTPUT	LCD - RS	
A3	DigitalA3	OUTPUT	LCD - RW	
A4	DigitalA4	OUTPUT	LCD - E (enable)	
A5	Analog5	N/A	N/C	

Lista della spesa (Bill Of Materials)

Codice	Descrizione	q.tà	link	Note
Arduino UNO	processore ARDUINO (Genuino)	1	LINK	Consiglio acquisto versione originale (diffidate dai cloni)
SparkFun joystick	Scheda joystick/tasti ARDUINO shield	1	LINK	
JHD204A	LCD alfanumerico 20x4 retroilluminato	1	LINK	prodotto su ebay a titolo di esempio. Ce ne sono tantissimi altri
Encoder	Encoder ottico in quadratura 200-600 p/giro	1	n/a	Consiglio ricerca su ebay. Eventualmente anche usati se di qualità e funzionanti
2M982	Controller motore stepper	1	LINK	Consiglio ricerca su ebay. Eventualmente anche usati se di qualità e funzionanti
PK299E4.5A	motore stepper	1	LINK	Consiglio ricerca su ebay. Eventualmente anche usati se di qualità e funzionanti
Alimentatore 48V	Alimentatore per controller stepper	1	LINK LINK	Piuttosto che un alimentatore a basso costo consiglio trasformatore, ponte e condensatori
Alimentatore 9V	Alimentatore ARDUINO (9V 500mA)	1	n/a	va bene anche recuperato, purchè abbia il jack di uscita compatibile ARDUINO
cavo schermato	Cavo schermato 4p o 6p +schermo	QB	n/a	da rivenditori di materiale elettrico/elettronico. Va bene anche cavo USB recuperato
cavo di potenza	Cavo 4P + terra sez 2mmq	QB	n/a	da rivenditori di materiale elettrico, chiedere cavo trifase con neutro

Cronologia aggiornamenti

Novità della revisione 1.6 (Novembre 2019)

- Definiti profili configurabili nei #define (prima scheda dello sketch) per le posizioni del joystick. Modifica necessaria per far fronte ad alcuni errori di lettura del joystick in caso di imprecisioni dovute a diverse versioni dell'hardware sparkfun. Nella lista dei #define sono stati definiti 8 valori corrispondenti a 8 step di lettura analogica del joystick, corrispondenti a specifiche tensioni lette sul cursore del joystick. La formula per il calcolo del valore digitale letto in base alla tensione presente sul cursore è la seguente:
 $(1023/5) * \text{tensione_cursore}$ – risultato da arrotondare a 0 decimali
- Integrata di default la configurazione di "INPUT_PULLUP" per gli ingressi encoder. Questa impostazione è tollerata anche nel caso vengano utilizzati encoder con uscita differenziale ed sarà quindi l'unica prevista da questa revisione del firmware in avanti

Novità della revisione 1.51 (Aprile 2018)

- Sistemato bug nella funzione di lettura posizione angolare mandrino. E' ora possibile leggere la posizione in gradi fino ad una velocità massima di ca. 200rpm (con encoder da 250x4 passi). E' comunque consigliabile ruotare il mandrino a mano a bassa velocità cercando di evitare strattoni e repentini cambi di direzione.

Novità della revisione 1.5 (Marzo 2018)

- Sistemato bug sulla lettura stringhe che causava crash casuali in uscita da alcune funzioni
- Sistemate funzioni di lettura della velocità di rotazione del mandrino che riportavano errori all'aumentare della velocità.
Le funzioni:
 - "velocità mandrino"
 - "avanzamento"riportano ora valori di velocità corretti.
- Aggiunta la possibilità di aggiustare la posizione del carro muovendolo con il joystick prima di effettuare una filettatura a misura. Nella versione precedente era necessario entrare nella suddetta funzione con il carro già correttamente posizionato. Ora viene data la possibilità di regolare la posizione direttamente nella stessa funzione.

Novità della revisione 1.41 (Gennaio 2016)

- Sistemata routine di calcolo della progressione di filettatura in modo da eliminare piccoli errori presenti nel passo della versione precedente
- Aggiunta la possibilità di impostare il passo da eseguire in TPI (filetti per pollice)
- Aggiunta funzione di avanzamento libero e corretta la funzione di avanzamento vincolato.
- Aggiunta la possibilità di impostare la modalità di avanzamento: mm/min e centesimi/giro (cent/g). La modalità cent/g calcola la velocità di rotazione del motore stepper sulla base della velocità del mandrino in modo da garantire con buona approssimazione l'avanzamento richiesto in centesimi/giro. La velocità di rotazione del mandrino non viene controllata continuamente durante la passata ma solo a passata

terminata; per questo motivo è sempre necessario terminare la passata prima di modificare la velocità di rotazione del mandrino.