Réglage des Drivers pour Moteurs pas à pas Pololu A4988, DRV8825, DRV8824 et DRV4834.

IMPORTANT! Ces différents Drivers n'ont AUCUN réglage d'usine. Le Réglage des Drivers pour Moteurs pas à pas Pololu A4988, DRV8825, DRV8824 et DRV4834, est donc une étape obligatoire dans la réalisation de votre machine CNC.

Introduction

Sans entrer dans les détails de fonctionnement d'un moteur pas à pas, qui sont très bien décrits dans de nombreux documents sur Internet, rappelons simplement le besoin d'envoyer des impulsions électriques de manière précise pour les piloter. Il est parfaitement possible de créer un circuit permettant de se passer du composant "Driver" (Pilote, où Contrôleur, en Français), et de gérer directement le moteur depuis un Arduino. Pourtant, utiliser un composant/circuit dédié à cette tâche rend les chose (et le câblage) beaucoup plus simple et pratique. C'est là qu'interviennent les Drivers, et plus particulièrement les drivers Pololu, particulièrement bien adaptés aux machines CNC type fraiseuse "de bureau", Imprimantes 3D et découpe à fil chaud.

Ces drivers ont tous en commun d'être destinés à piloter des moteurs pas à pas bipolaires, et la plupart des cartes pour les machines décrites plus haut sont prévues pour ces moteurs. Si vous utilisez un CNC Shield pour Arduino, il est prévu pour fonctionner avec l'un de ces drivers.

Ce qui différencie ces drivers, c'est le voltage qu'ils peuvent accepter, voltage qui va être envoyé au moteur, et le courant qu'ils peuvent délivrer (Ampérage) par phase, donc la puissance de l'impulsion électrique fournie au moteur. Il est important de faire attention à ces caractéristiques, et d'appairer au mieux moteur et driver.

Notre choix de moteur et driver

Dans cet article, nous utilisons un moteur Nema 17, qui porte la référence 17HS19-2004S. Les caractéristiques qui nous importent ici, sont la valeur de courant par phase, la résistance par phase, et le voltage recommandé:

Caractéristiques Moteur	Quantité	Unités
Courant par Phase	2.0	Ampères
Résistance par Phase	1.4	Ohms

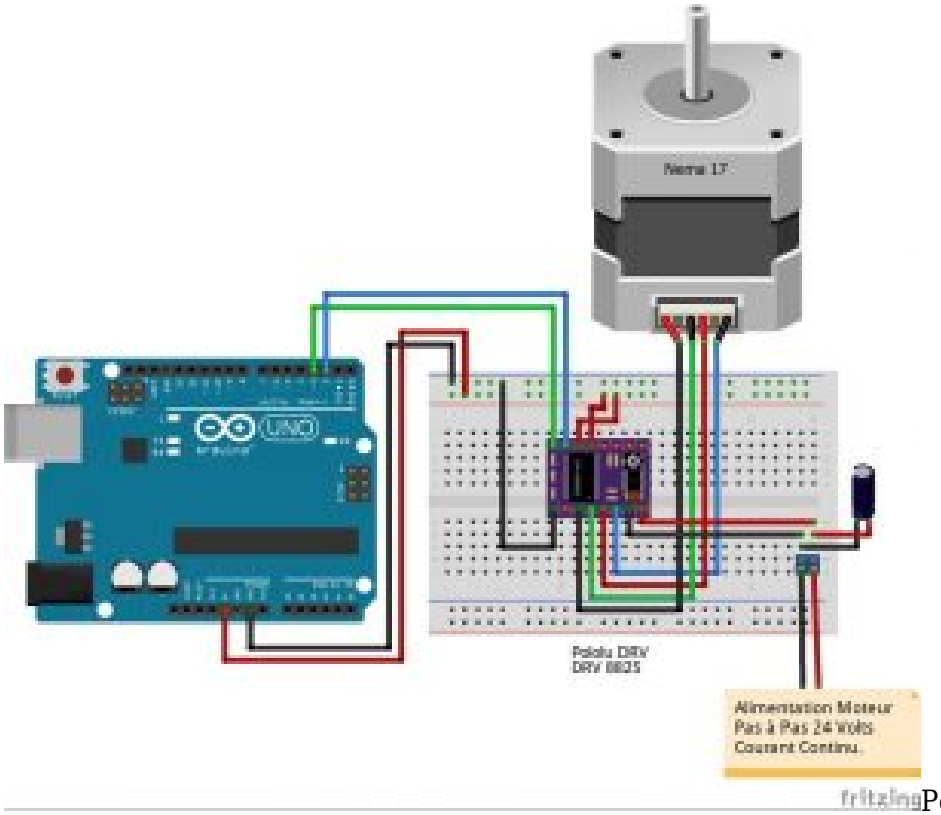
Voltage Recommandé	12-24	Volts

Le driver choisi est le Pololu DRV8825. Les caractéristiques de ce driver sont:

Caractéristiques Driver	Quantité	Unités
Courant par Phase	2.2 Max	Ampères
Voltage Recommandé	8.2 – 45	Volts

L'alimentation utilisée est de 24 Volts. On voit tout de suite qu'elle convient au driver et au moteur. On voit aussi que le driver ne va pas limiter le courant que je peux envoyer au moteur, donc je vais pouvoir en exploiter tout le couple.

Circuit simple pour piloter le moteur pas à pas depuis l'Arduino



Pour réaliser ce circuit, installez votre driver sur la platine de test, et effectuez les divers liaisons comme indiqué.

Attention, ne branchez pas l'alimentation moteur avant que tout ne soit installé et prêt, y compris les branchements avec l'Arduino. Il vaut mieux aussi que le programme de test soit déjà flashé.

Sur ce schéma, les 4 fils du moteur sont identifiés B2, B1, A1, A2. Sur votre moteur, vous avez certainement des fils avec 4 couleurs différentes. Si vous pouvez trouver la documentation exacte de votre moteur, tant mieux! Pour mon moteur, j'ai la correspondance Noir = A+, Vert = A-, Rouge = B+, Bleu = B-

Si vous n'avez pas cette information, vous pouvez déterminer quels fils sont appairés sur la même phase avec votre multimètre. Si 2 fils partagent le même bobinage, vous aurez une résistance, sinon

vous aurez une résistance infinie

L'image de droite correspond au schéma, implémenté sur une platine test, avec l'Arduino. Les alimentations de sont pas branchées. Celle du moteur, en 24 volts se branche sur le bornier bleu. Si vous ne connaissez pas Fritzing, avec lequel ce schéma a été réalisé, et qu'il vous arrive de dessiner vos propres circuits, penchez-vous sur cet excellent logiciel open source.

Mise à jour depuis la parution de cet article:

Je me suis rendu compte que platine d'essai sans soudures, condensateur et fils de connexion ne faisaient pas forcément partie de la trousse à outil de tous les lecteurs! Il sera donc beaucoup plus simple d'utiliser le shield de réglage que j'ai créé dans ce but. Même si vous avez par ailleurs un Shield CNC, où une carte pour imprimante 3D avec ces drivers montés, il sera bien plus facile de régler proprement votre driver en le montant tout seul sur ce shield. Le processus est simple, mais la manipulation ne l'est pas, le potentiomètre étant très sensible. Être à l'aise, avec l'espace dégagé, est un gage de réussite et de meilleur confort pour réaliser cette opération.

Régler sur quoi?

Notre premier réglage va s'effectuer avec le moteur pas à pas débranché. Mais **la toute première chose à faire est de déterminer sur QUELLE VALEUR** je veux régler mon driver!

Pour cela, il me faut le document technique de Pololu pour mon driver spécifique. je vais donc sur la page produit de Pololu pour le DRV8825, fait défiler vers le bas vers le chapitre "Current limiting", pour trouver la formule qui nous intéresse:

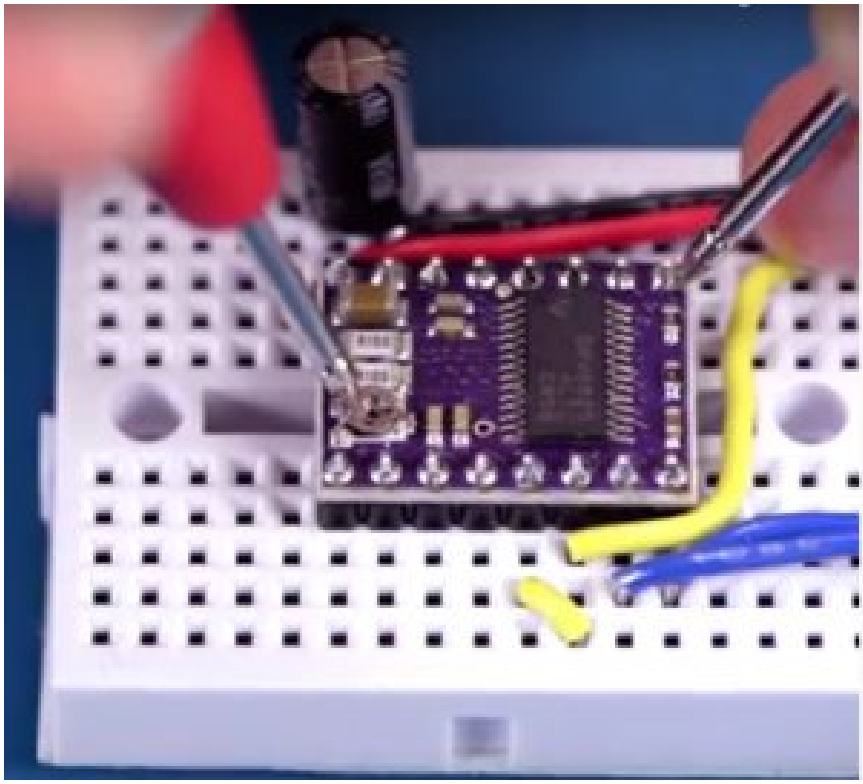
> *Another way to set the current limit is to measure the voltage on the "ref" pin and to calculate the resulting current limit (the current sense resistors are **0.100Ω**). The ref pin voltage is accessible on a via that is circled on the bottom silkscreen of the circuit board. The current limit relates to the reference voltage as follows:*

> *Current Limit = VREF × 2*
>
> *So, for example, if you have a stepper motor rated for 1 A, you can set the current limit to 1 A by setting the reference voltage to 0.5 V.*

Encore une fois, vérifiez cette formule pour VOTRE driver. Pour un A4988, par exemple, la formule devient *Current Limit = VREF × 2.5.*

Comme nous l'avons vu plus haut, mon moteur est donné pour un maximum de 2 Ampères. Je pourrais donc régler le Voltage de Référence (nous allons expliquer comment dans une minute) à 1 Volt, et je serais dans les clous. Simplement, cette valeur est une valeur Max. Pololu indique clairement qu'au-dessus de 1.5 Ampère, il est impératif de refroidir le driver, ce que j'ai l'intention de faire, mais je ne pense pas avoir besoin de tourner en permanence au maximum, d'autant que si je peux éviter de diminuer la durée de vie de mes composants, ç'est autant de gagné. Je vais donc régler le Voltage sur 0.900 Volts, de manière à être réglé sur 1.8 Amps.

Comment Régler?



Si vous le pouvez, le mieux est de clipper votre petit tournevis avec une pince alligator branchée sur le plus de votre multimètre. Faites toucher la sonde coté moins du multimètre avec la broche neutre/moins du Pololu, et mettez la pointe du tournevis sur le potentiomètre. De cette manière, vous pouvez modifier le réglage en même temps que vous lisez la valeur affichée sur le multimètre. Si vous ne pouvez pas équiper la sonde du multimètre d'une pince alligator, vous pouvez toucher le point VREF avec la sonde plus, et régler avec le tournevis en même temps, mais c'est plus acrobatique! Attention, de tout petits mouvements du potentiomètre sont suffisants. Même s'il semble qu'un tournevis cruciforme devrait être utilisé, vous serez plus précis avec un petit tournevis à bout plat. En fait, si vous ne pouvez pas directement tourner le potentiomètre en même temps que vous lisez le voltage sur votre multimètre, procédez par petites touches et mesurez entre chaque. C'est encore la meilleure méthode si vous n'avez pas 4 mains!

Conclusion

Tout ceci peut paraître bien compliqué de prime abord, mais en fait, en progressant étape par étape, vous verrez rapidement qu'il n'y a rien de bien sorcier. On peut penser que monter un circuit de test pour cet usage, c'est en faire un peu trop, pourtant, cet étape de réglage des drivers pour moteurs pas à pas Pololu A4988, DRV8825, DRV8824 et DRV4834, est la meilleure manière d'isoler un problème potentiel dans le fonctionnement de votre machine. N'oubliez pas qu'il n'y aucun réglage d'usine. Passer par cette étape est s'assurer de la durée de vie de vos moteurs, de vos drivers, et même de celle de votre Arduino.

Code utilisé pour les réglages et tests

```
/* Simple step test for Pololu stepper motor driver carriers
This code can be used with the A4988, DRV8825, DRV8824, and
DRV8834 Pololu stepper motor driver carriers.  It sends a pulse
every 500 ms to the STEP pin of a stepper motor driver that is
connected to pin 2 and changes the direction of the stepper motor
every 50 steps by toggling pin 3. */


#define STEP_PIN 2
#define DIR_PIN 3

bool dirHigh = true;

void setup()
{
        dirHigh = true;
        digitalWrite(DIR_PIN, HIGH);
        digitalWrite(STEP_PIN, LOW);
        pinMode(DIR_PIN, OUTPUT);
        pinMode(STEP_PIN, OUTPUT);
}

void loop()
{
        // Toggle the DIR pin to change direction.
        if (dirHigh)
        {
                dirHigh = false;
                digitalWrite(DIR_PIN, LOW);
        }
        else
        {
                dirHigh = true;
                digitalWrite(DIR_PIN, HIGH);
        }

        // Step the motor 50 times before changing direction again.
        for (int i = 0; i < 200; i++)
```

```
        {
                // Trigger the motor to take one step.
                digitalWrite(STEP_PIN, HIGH);
                delay(5);
                digitalWrite(STEP_PIN, LOW);
                delay(5);
        }
}
```

Quelques commentaires sur ce code:

- La boucle compte jusqu'à 200 (en fait, 199, mais en partant de zéro), avant de changer de sens. Pour un moteur faisant 1,8° par pas, il faut 200 pas pour un tour. Nous faisons donc effectuer une rotation complète dans un sens, avant de changer de sens.
- Pour varier la vitesse de rotation, modifiez la valeur de délais, en millisecondes. Mettez 1 pour aller plus vite, 30 pour aller lentement.
- Ce code fait faire des pas « pleins » au moteur, ni demi pas, 1/4 de pas où plus. Ce n'est pas nécessaire ici.