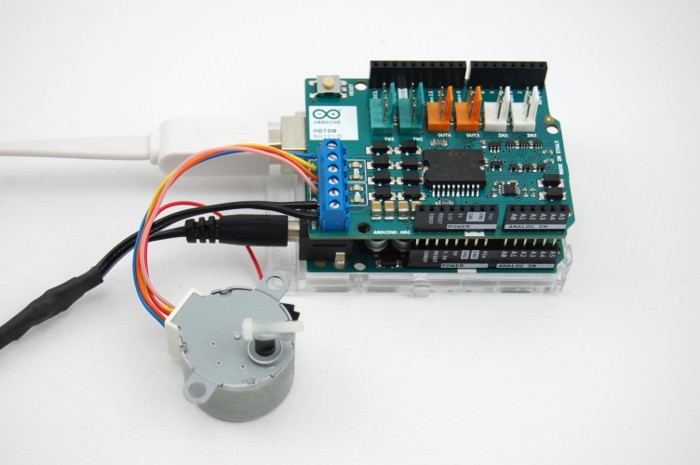
**RECHERCHE MOTEUR**

**Apprendre à contrôler un moteur pas à pas avec le shield moteur Arduino**



**Qu'est-ce qu'un moteur pas à pas ?**

Un moteur pas à pas est un type de dispositif électromagnétique dont la rotation peut être contrôlée de façon précise. Pour cela, plusieurs bobines sont organisées en "phases", et l'excitation séquentielle des phases entraîne une rotation. L'un des principaux avantages de ce type de moteur est qu'il permet de contrôler précisément sa vitesse et/ou le positionnement, et c'est pourquoi il est utilisé dans les applications demandant une haute précision, telles que les imprimantes.

**Moteurs unipolaires et bipolaires**

Il existe deux types de moteur pas à pas, unipolaire et bipolaire. La principale différence entre les deux réside dans l'organisation de leurs bobinages, car cela a un impact sur leur mode de contrôle.

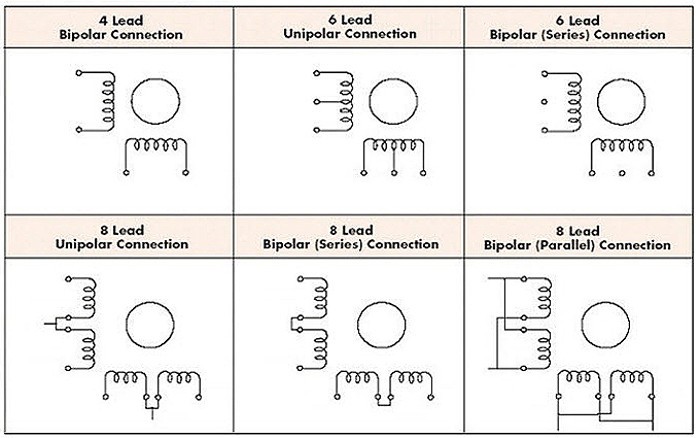
**Unipolaire**

Ce type de moteur pas à pas consiste en un bobinage unique avec une prise médiane. Chaque section du bobinage est excitée en fonction de la direction du champ magnétique voulu, ce qui permet d'inverser les pôles sans inverser la direction du courant. La prise médiane est commune et bien qu'il y ait généralement 6 fils sur un moteur pas à pas unipolaire à deux phases (3 par phase), les deux communs peuvent être reliés à l'intérieur, ce qui laisse en réalité cinq fils.

**Bipolaire**

Contrairement à un moteur pas à pas unipolaire, la version bipolaire n'a qu'un seul bobinage par phase et pas de prises médianes. Pour inverser le pôle magnétique, il faut également inverser la direction du courant, ce qui signifie que ce moteur est généralement plus compliqué à contrôler et nécessite une disposition à pont en H. Dans la mesure où il n'y a pas de prise médiane, il y a deux fils par phase et un moteur biphasé type comporte quatre fils. Si ces moteurs sont plus difficiles à contrôler, ils ont également leurs avantages, car ils utilisent mieux le bobinage et sont plus puissants que des moteurs unipolaires de la même taille. Cela est dû au fait qu'un moteur pas à pas unipolaire a une densité de fil deux fois supérieure, alors que seule la moitié de ces derniers est exploitable à un instant T et que par conséquent, ce type de moteur n'est efficace qu'à 50 %.

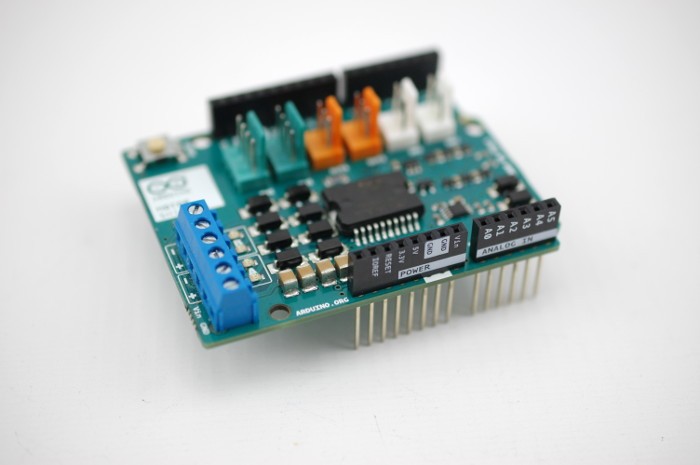
Les différentes dispositions des bobinages des moteurs unipolaires et bipolaires sont illustrées ci-dessous.



*Image source: Osmtec.com*

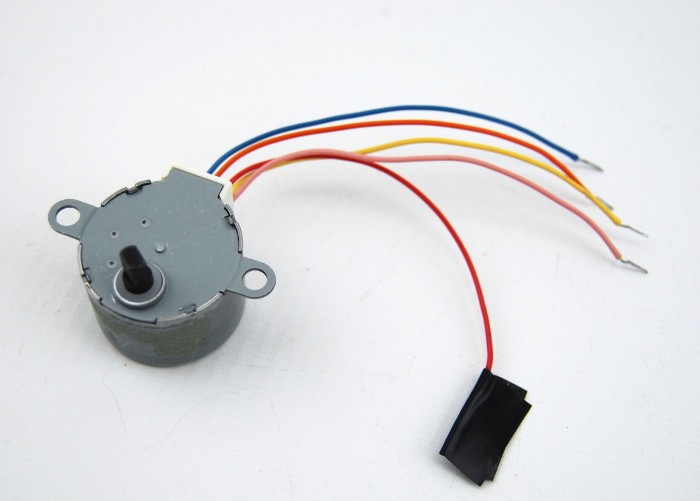
**Shield moteur Arduino**

Le shield moteur Arduino repose sur le double driver de pont complet L298 qui permet d'utiliser un moteur pas à pas unique ou deux moteurs à courant continu. Ce shield contrôle la vitesse et la direction de façon indépendante. Il permet de connecter un ou plusieurs moteurs directement sur la carte Arduino sans passer par une platine de montage ou des circuits supplémentaires, comme c'est normalement le cas lors de l'utilisation d'un module Arduino.



**Mon moteur**

Pour ce projet, j'ai choisi d'utiliser un moteur pas à pas unipolaire bipolaire, unipolaire, de Cliff Electronics, qui est un moteur pas à pas unipolaire à cinq fils. Toutefois, en ignorant le fil commun, il est possible de l'utiliser comme un moteur pas à pas bipolaire.

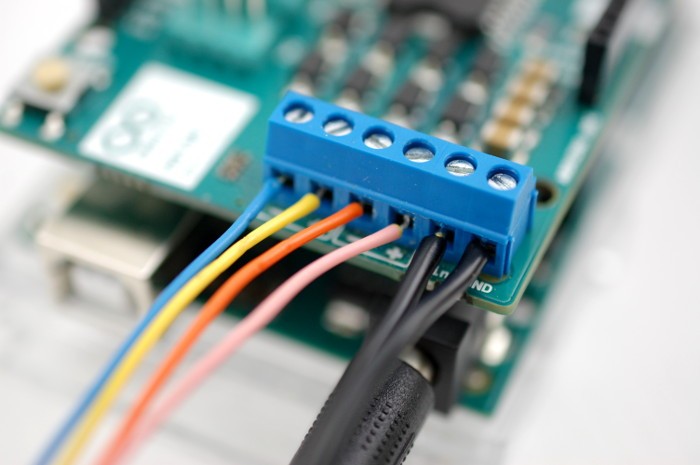


Dans la mesure où ce moteur nécessite une alimentation de 12 V, j'ai dû séparer les lignes de puissance du shield et de mon Arduino Uno, afin d'écarter tout risque de dommages, comme recommandé sur la page Web du shield moteur Arduino. Parce que les lignes de puissance devaient être séparées, je ne pouvais plus utiliser l'alimentation secteur 12 V qui se branche directement sur la carte Arduino. Il a fallu la modifier pour alimenter à la fois la carte Arduino et le shield moteur Arduino. Pour ce faire, j'ai coupé le cordon à environ 10 cm du connecteur et à peu près la même longueur de fil, que j'ai ensuite étamés et soudés pour réaliser le montage ci-dessous.



**Identification des bobines du moteur**

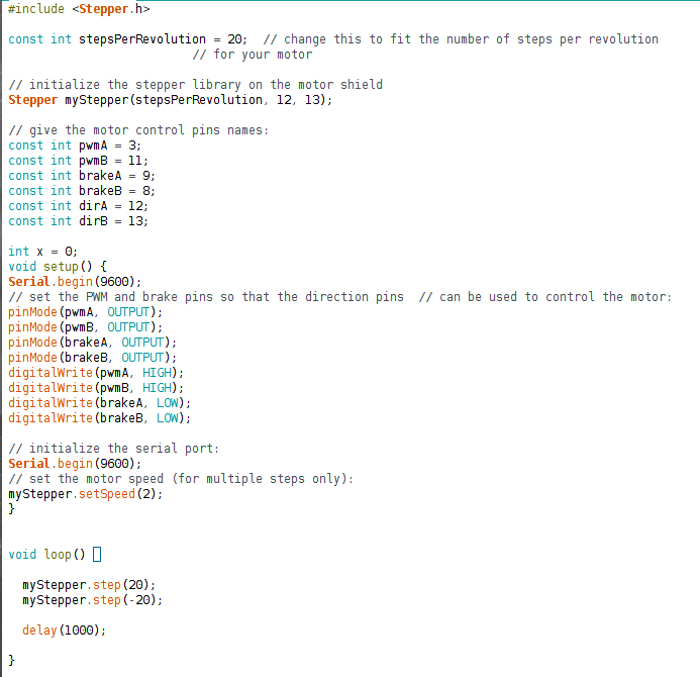
La fiche technique du fabricant du moteur que j'ai choisi était claire, et je savais que le bleu et le jaune formaient une bobine, tandis que le rose et l'orange en représentaient une autre, et que le rouge faisait référence au commun. J'ai connecté une bobine au canal A et l'autre au canal B du shield moteur, tout en laissant le commun déconnecté. Je peux ignorer le fil rouge, car il s'agit de la prise médiane des deux bobines qui n'a d'utilité que pour créer un moteur pas à pas unipolaire.



Si la fiche technique ne précise pas la relation entre les fils et les bobines, il est assez facile de la déduire avec un multimètre en mode ohmmètre. Une partie de la bobine montrera une résistance, le fil de prise médiane donnera la moitié de la résistance des extrémités. Par conséquent, vérifiez les autres fils pour vous assurer qu'une connexion médiane n'est pas confondue avec une extrémité. S'il n'y a aucune résistance, nous sommes en présence de deux bobines distinctes.

**Déplacement du moteur**

Une fois mon moteur connecté, j'ai recherché du code pouvant fonctionner avec le shield moteur, de façon à m'aider à comprendre par où commencer. J'en ai trouvé dans le [didacticiel Instructables Arduino Motor Shield](http://www.instructables.com/id/Arduino-Motor-Shield-Tutorial/step4/One-Motor/#%C3%89tape%C2%A01) qui m'a permis de faire tourner mon moteur. C'était un bon début, mais je voulais inclure la bibliothèque Stepper.h dans le programme utilisé, et cet exemple de code ne le permettait pas. Il est préférable d'utiliser des bibliothèques officielles dans la mesure du possible et j'ai donc poursuivi mes recherches et trouvé mon bonheur sur un forum Arduino. Avec ce programme, le moteur tourne sur 360° dans un sens, puis encore sur 360° dans l'autre.

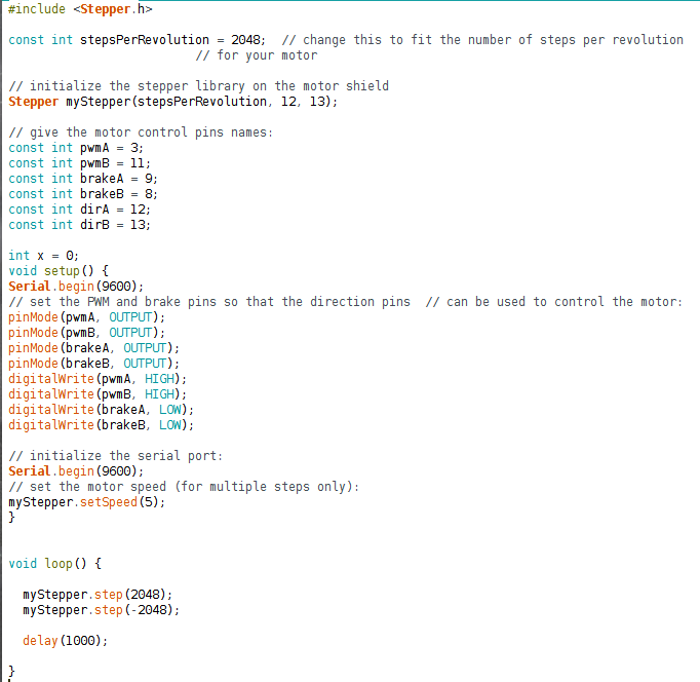


Dans ce programme, je devais spécifier le nombre de pas par révolution pour le moteur utilisé et j'ai donc dû déterminer cette valeur. La fiche technique du fabricant stipulait des incréments de 5,625 et 11,25 degrés pour le moteur et je me suis appuyé sur ces valeurs pour déterminer le nombre de pas par révolution. Pour ce faire, il faut diviser 360 par l'angle. Dans la mesure où ce moteur a également des engrenages et un rapport de transmission de 1:64, il faut multiplier le nombre de révolutions par ce rapport. Par exemple :

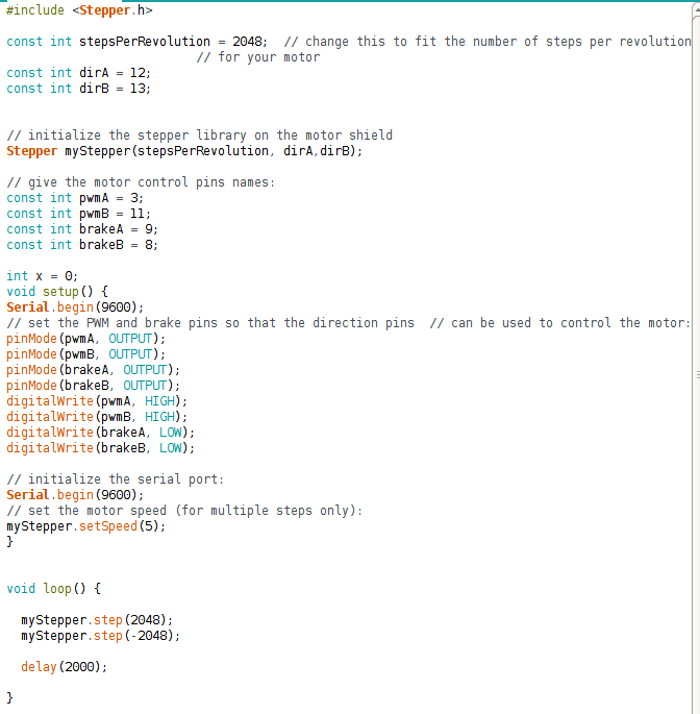
360/11,25 x 64 = 2 048

360/5,625 x 64 = 4 096

Pour commencer, j'ai utilisé 4 098 pas par révolution dans le programme, en estimant que le moteur avait un angle de 5,625 degrés. Toutefois, en faisant cela, il réalisait deux révolutions complètes dans le sens antihoraire, puis deux autres dans le sens horaire. Il était évident que j'avais choisi une valeur fausse, car il faisait deux rotations et parce que 5,625 est la moitié de 11, ce qui donne un produit deux fois supérieur après division, d'où les deux révolutions complètes. J'ai alors entré 2 048 pas par révolution et cette fois-ci le moteur a fait une révolution dans le sens horaire, puis une autre dans le sens antihoraire. J'ai également fait passer la vitesse du moteur de deux à cinq, car avec la valeur deux, le moteur semblait beaucoup trop lent.



Ce n'est qu'après avoir terminé cette opération que j'ai pu tourner mon attention vers le code utilisé, et une partie de ce dernier me paraissait étrange. Je ne comprenais pas pourquoi les broches dirA et dirB étaient déclarées, alors qu'elles n'étaient pas utilisées dans les paramètres ou la boucle. J'ai placé des marques de commentaire devant ces deux lignes et chargé le programme à nouveau, et tout a très bien fonctionné. J'en ai donc déduit que ces deux lignes étaient superflues et je les ai supprimées pour désencombrer le programme. Par la suite, j'ai remis ces lignes, mais au-dessus de la phase d'initialisation de la bibliothèque Stepper et les ai utilisées comme des définitions.



J'ai ensuite examiné le code source de la bibliothèque Stepper et me suis aperçu qu'elle ne contenait aucune ligne de code pour contrôler les broches de frein. La raison pour laquelle l'exemple inclut des lignes de contrôle des broches de frein, en les définissant comme sorties et sur un niveau logique bas, est qu'il s'agit d'une mesure de précaution, pour les empêcher de flotter.