Sami Bachir

Séverin-Marie Depasquale

Baptiste Durand

Commande Vitesse Moteur

E2 2015

ESIEE PARIS

Introduction au projet

Le projet de commande de la vitesse du moteur s'est effectué dans le cadre du projet interdisciplinaire PR2001. Ce projet s'inscrit dans la continuité du cycle préparatoire de la formation d'ingénieur à ESIEE Paris. L'interdisciplinarité du projet permet de réunir deux matière de sciences de l'ingénieur : l'électronique en premier lieu pour les nombreux circuits électriques que le projet a nécessité et l'informatique en second lieu du fait de la programmation de la carte Arduino.

Le but de ce projet est de commander la vitesse du moteur grâce à l'appui de boutons sur la carte Arduino. Il est par ailleurs demandé de pouvoir mesurer une tension aux bornes d'une résistance en série avec le moteur ainsi que la vitesse réelle du moteur grâce un montage électronique.

Ce rapport a pour objectif de vous présenter de manière précise et efficace le compte rendu de notre projet ainsi que toutes les explications est schémas qui s'en rattachent de manière à vous permettre de comprendre le plus facilement possible le fonctionnement de notre montage.

Nous vous souhaitons une agréable lecture !

Les auteurs

Table des matières

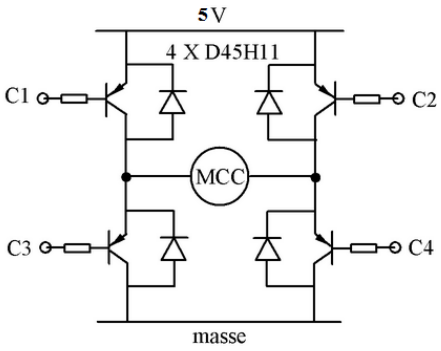
Partie I : Le Pont en H

1/ Introduction :

Pour pouvoir faire tourner le moteur dans les deux sens en fonction de l'interrupteur appuyé, nous allons avoir besoin d'un circuit électronique appelé : « pont en H ». Le pont en H est la partie qui fait office d'interface entre la carte Arduino et le moteur. On se propose ici d'étudier le pont en H, comment fonctionne le pont en H ?

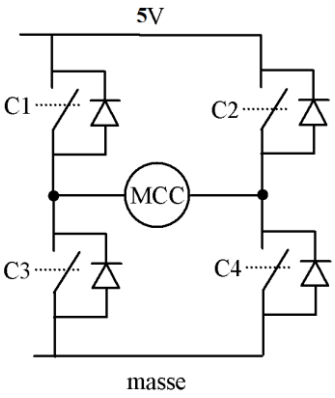
2/ Description du pont en H :

Le pont en H est le circuit électronique qui va nous servir d'interface entre la carte Arduino Uno et le moteur MCC. En effet, la carte Arduino n'est équipée que de ports PWM capables de délivrer, au mieux 40 mA, ce qui est trop faible pour pouvoir alimenter directement le moteur. C'est pourquoi, il nous faut utiliser un circuit électrique intermédiaire permettant de délivrer intensitée électrique de fonctionnement acceptable pour le moteur, tout en ayant le fonctionnement contrôlé par la carte Arduino. Ce circuit électronique est le pont en H.



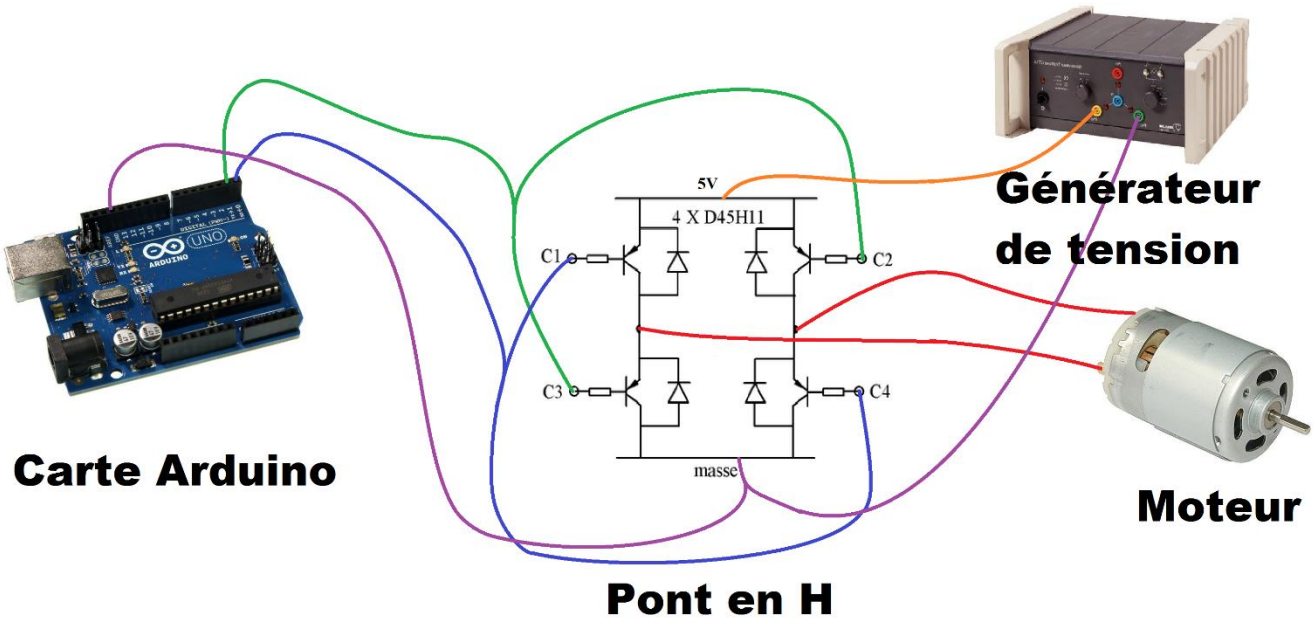
Le pont en H possède en tout quatre entrés qui le commande : C1, C2, C3 et C4. Ces entrés, sont reliés entres elles deux à deux selon le principe suivant : C1 avec C4, C2 avec C3. Elles sont directement commandées par la carte Arduino via ses ports numériques 1 et 2.

Ces entrés agissent, via des résistances de capacités 150 ohms, sur la base de transistors PNP de modèle D45H11 que l'on nommera respectivement T1, T2, T3, T4 et qui servent à bloquer ou non l'alimentation électrique de 5V d'un générateur externe de tension continue qui permet de faire tourner le moteur MCC.



Etant de type PNP, ces transistors ne laissent passer le courant entre l'émetteur et le récepteur que si une tension nulle est appliquée à leur base. Le cas inverse empêche ou du moins diminue très fortement le courant passant entre leur base et leur émetteur. Nous les en utiliserons ici en tant qu'interrupteurs (voir schéma ci-contre). Des diodes relient le collecteur et l'émetteur des transistors PNP.

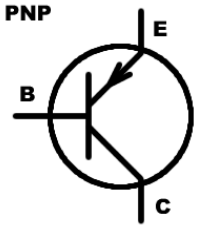
Finalement, nous obtenons le montage ci-dessous :



3/ Principe de fonctionnement :

a/ Le transistor D45H11 :

Notre pont en H utilise des transistors PNP de type D45H11 de la marque Fairchild (la datasheet se situe à : https://www.fairchildsemi.com/datasheets/D4/D45H11.pdf). Les transistors sont des composants électriques quasiment systématiquement utilisés en électronique soit en tant qu'amplificateurs de tension, soit en tant qu'interrupteurs. Ici, nous utiliserons les transistors PNP uniquement en tant qu'interrupteurs commandés par un signal électrique, c'est à dire en tant qu'interrupteurs « électronique » dont on peut faire varier le courant transitant à l'intérieur de part et d'autre. Les transistors PNP sont des composants tribroches, c'est à dire possédant trois broches : la base, le collecteur et l'émetteur.

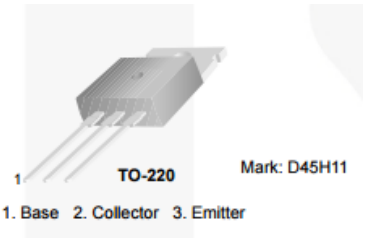


Symbole du transistor PNP

C'est un composant polarisé, c'est-à-dire que celui-ci ne peut être branché que dans le sens du courant électrique pour pouvoir bien fonctionner. Et l'émetteur et le récepteur sont des jonctions de type P comme positive, la base est une jonction de type N comme négative, d'où le nom de transistor PNP.

Il existe deux types de transistors : les transistors bipolaires et les transistors à effet de champs. Le transistor D45H11 est un transistor bipolaire.

L'avantage de l'utilisation du transistor en tant qu'interrupteur est que celui-ci possède un délai de commutation extrêmement court (de l'ordre de la nanoseconde !), ce qui lui permet d'effectuer des ouvertures et des fermetures de manière extrêmement rapide ce qui est très utile quand il s'agit de retranscrire un signal. C'est d'ailleurs la principale raison pour laquelle ce composant est systématiquement utilisé dans les ordinateurs par exemple.



La base (symbole B sur le schéma)

La base B du transistor PNP est une broche dont la tension appliquée à celle-ci détermine l'amplification du courant entre l'émetteur E et les collecteur C. Plus la tension appliquée à la base est élevé, moins le courant passera entre le collecteur et l'émetteur. Et vice versa, plus la tension appliquée à la base sera faible, plus le courant pourra passer, il pourra même être amplifié selon un facteur ε. Notre transistor peut donc être modélisé par un interrupteur variable commandé par la tension à sa base.

Le collecteur (symbole c sur le schéma)

Le collecteur est une des trois broches du transistor PNP.

L'émetteur (symbole e sur le schéma)

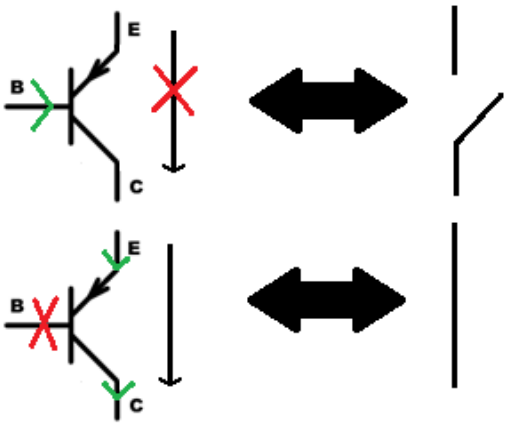
L'émetteur est une des trois broches du transistor PNP.

Fonctionnement

Le fonctionnement du transistor PNP en tant qu'interrupteur est le suivant :

Quand un signal électrique est transmis à sa base, celui-ci ne laisse pas passer le courant électrique entre le récepteur et l'émetteur : celui-ci agit comme un interrupteur ouvert.

A l'inverse, quand aucun signal électrique n'est transmis à sa base, celui-ci laisse passer le courant électrique entre le récepteur et l'émetteur : celui-ci agit comme un interrupteur fermé.



Fiche technique du transistor :

Voici les caractéristiques du transistor telles que sur la datasheet du constructeur.

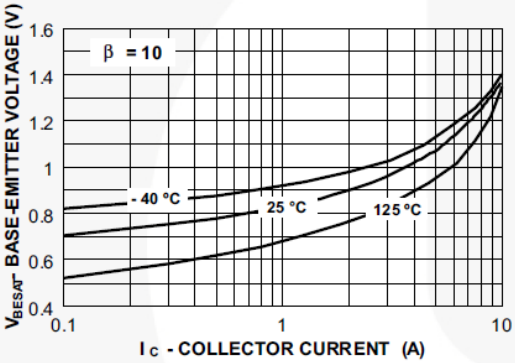


Figure 3. Base-Emitter Saturation Voltage vs. Collector Current

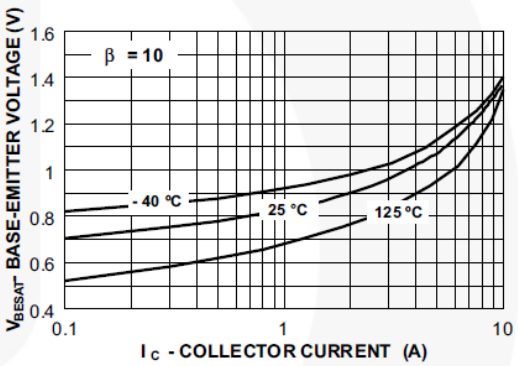


Figure 4. Base-Emitter ON Voltage vs. Collector Current

b/ Fonctionnement du moteur

Le moteur électrique

Pour notre montage, nous allons utiliser un moteur électrique. Concrètement, le moteur électrique est un dipôle (composant électrique à deux broches), non polarisé (qui fonctionne dans les deux sens), qui obéis à la loi d'Ohm (c'est-à-dire que U=RI). Ci-contre, le symbole officiel du moteur.

Le moteur est utilisé pour convertir un courant électrique en mouvement rotatif qui lui-même peut être convertis en d'autres types de mouvement (c'est de la mécanique).

Par ailleurs, ce moteur ne peut fonctionner que si les conditions d'alimentation sont respectées. Si l'intensité électrique fournie au moteur est insuffisante, alors celui-ci consommera cette intensité électrique sans réussir à se mettre en mouvement : toute la puissance électrique sera dissipée sous forme d'effet joule ! Dans le cas inverse, si cette puissance est trop élevée, cela risque d'abimer le moteur en le faisant tourner à une vitesse excessive.

Description du moteur :

Le moteur utilisé dans notre pont en H, est un moteur de type CC 3 de la marque Como Drills. Selon le site RadioSparses, ce moteur est dédié à : *une sélection de vitesses, de tensions d'entrée et de couples de sortie. Idéal pour les projets de développement et d'éducation, les bancs d'essai et de démonstration et les applications de mouvement ou de placement simple ou économique.*

La datasheet est sur : www.radiospares.fr

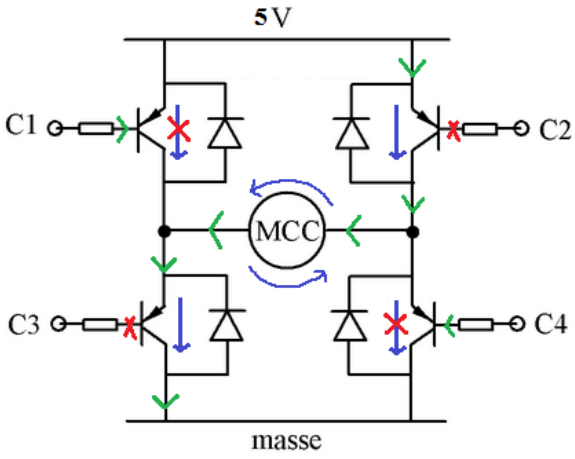
Fiche technique du moteur :

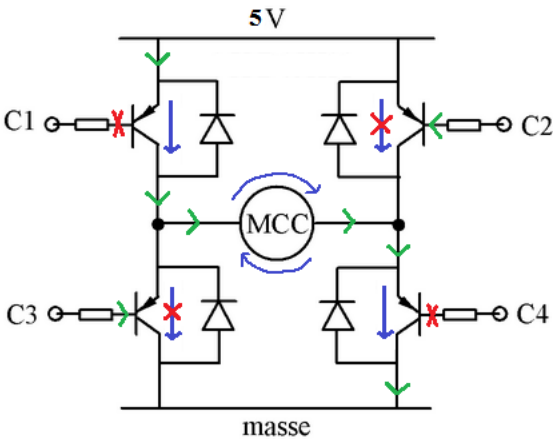
- Couple de sortie maximum : 375 gcm
- Courant : 4,41 A
- Diamètre d'arbre : 2.3mm
- Gamme de puissance : 19,68 W
- Longueur : 38mm
- Matériau du noyau : acier
- Tension d'alimentation : 3 → 7,2 V c.c.
- Vitesse de sortie maximale : 19000 tr/min

c/ Fonctionnement du circuit :

Description du fonctionnement

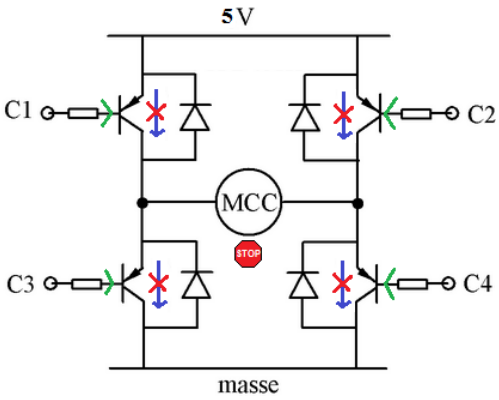
Quand le port numérique Arduino 1 qui commande les entrés C1 et C3 du pont en H est actif et que le port numérique Arduino 2 qui commande les entrés C2 et C3 du pont en H est inactif, alors les transistors T2 et T4 laissent passer la tension de 5V du générateur externe contrairement aux transistors T1 et T3 qui sont bloqués. Par conséquent, le courant aux bornes du moteur MCC circule dans un seul sens et le moteur tourne dans un premier sens.

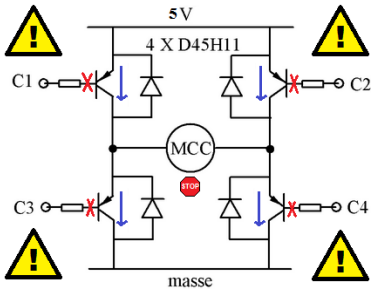




Quand le port numérique Arduino 1 qui commande les entrés C1 et C4 du pont en H est inactif et que le port numérique Arduino 2 qui commande les entrées C2 et C4 du pont en H est actif, alors les transistors T1 et T3 laissent passer la tension de 5V du générateur externe contrairement aux transistors T2 et T4 qui sont bloqués. Par conséquent, le courant aux bornes du moteur MCC circule dans un seul sens et le moteur tourne dans un second sens.

Quand le port numérique Arduino 1 qui commande les entrés C1 et C4 du pont en H est actif et que le port numérique Arduino 2 qui commande les entrés C2 et C3 du pont en H est actif, alors tous les transistors sont bloqués. Par conséquent, il n'y a aucun courant qui circule aux bornes du moteur MCC et le moteur est à l'arrêt.

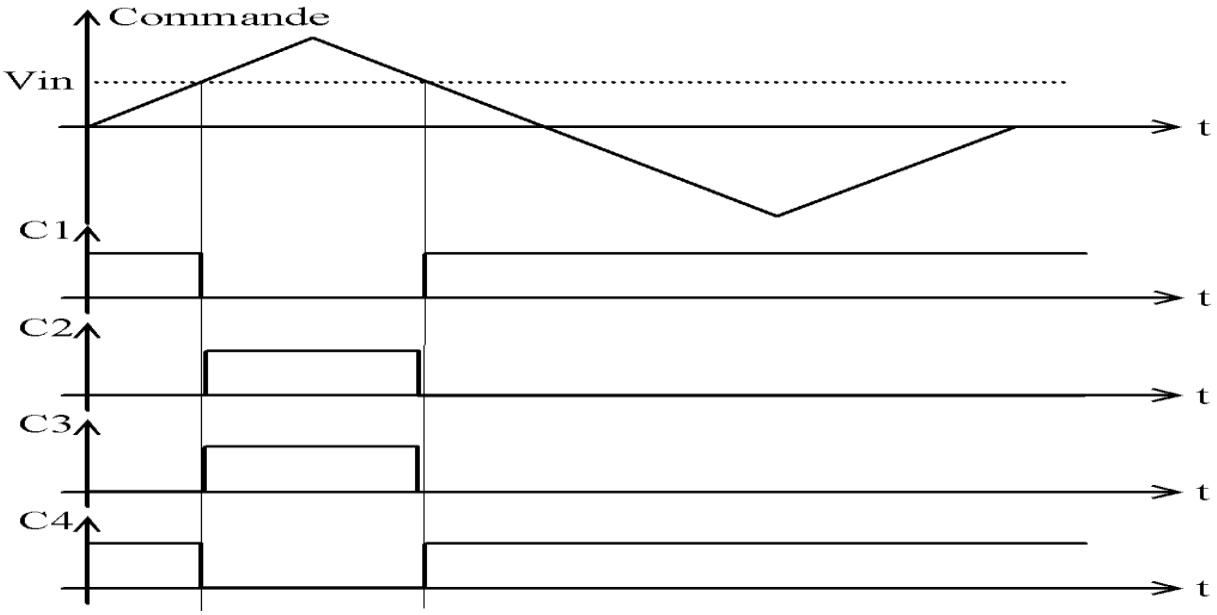




Le principal risque de ce montage est qu'il ne faut surtout pas que ces quatre transistors soient actifs en même temps. En effet, si ce cas devait arriver, en cas de mauvaise manipulation par exemple, alors on aurait un court-circuit ! Ce court-circuit pourrait abimer définitivement nos transistors, voire même causer un départ d'incendie ! Il s'agit là de la principale contrainte du pont en H. Cette contrainte nous obligera à faire en sorte d'arrêter totalement le moteur pendant un court laps de temps, à chaque changement de sens du moteur, pour éviter le cas exposé ci-dessus.

Chronogrammes :

Finalement, nous obtenons le chronogramme ci-dessous :



On retrouve exactement ce que l'on avait décrit plus haut, c'est-à-dire le fonctionnement du moteur dans un sens quand C1 et C4 sont actifs et C2 et C3 inactifs et le fonctionnement du moteur dans l'autre sens quand C2 et C3 sont actifs et C1 et C4 inactifs.

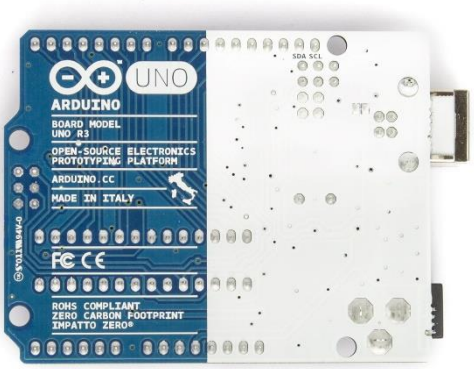
d/ Conclusion

Ainsi, en réalisant un pont en H, nous avons pu mettre en place un système de commande « indirecte » de la vitesse du moteur. Ce système a permis de d'utiliser pour commander le moteur, une carte Arduino qui malgré cela ne délivrai pas suffisamment de puissance pour faire fonctionner le moteur.

Partie II : La carte Arduino

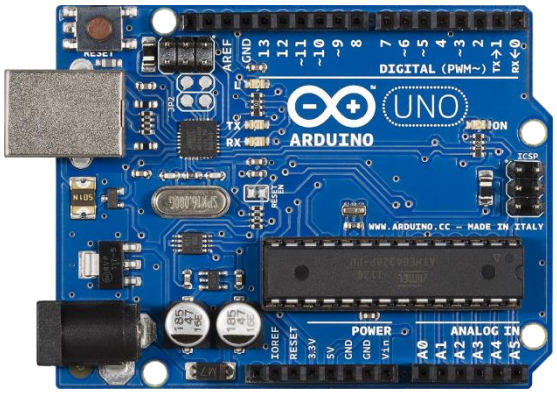
1/ Introduction

Pour contrôler le fonctionnement de notre moteur en fonction des entrés du système, nous allons avoir besoin d'une partie « contrôle » qui prendra la forme d'un circuit électronique. Le composant de cette partie est en quelques sortes le « cerveau » de notre montage car c'est lui qui analyse le signal transmis par les interrupteurs via ses broches numériques et qui détermine la vitesse et le sens de fonctionnent du moteur. Pour ce faire, nous utiliserons une Arduino de type Uno.



La carte Arduino commande indirectement la vitesse du moteur. En effet, les broches numériques de cette carte sont reliées à un circuit électronique qui sert d'intermédiaire entre la carte et le moteur : le pont en H. Pour faire varier la vitesse du moteur, la carte Arduino utilise le principe de Pulse With Modulation (PWM), c'est-à-dire qu'elle envoie de courtes pulsations périodiques d'amplitude constante aux transistors du pont en H auquel elle est reliée dans le but de contrôler la vitesse du moteur en fonction de la fréquence des pulsations.

Dans cette partie, nous ne nous intéresserons qu'au fonctionnement interne de la carte Arduino ainsi qu'à la programmation de la carte Arduino pour commander le fonctionnement du moteur. Une autre partie traitera de l'analyse de la vitesse du moteur et abordera l'analyse de la vitesse par la carte Arduino.



2/ Présentation de la carte Arduino

a/ Présentation générale

Qui n'a jamais entendu parler de la célèbre carte électronique de couleur bleue : la carte Arduino ? Cette carte est une carte très utilisée tant dans le monde des amateurs d'électroniques (elle est par exemple quasi systématiquement utilisée dans le concours Innovez de Sciences et Vie Junior), que dans le monde professionnel ou ses performances doublées d'une grande simplicité d'utilisation et d'un prix très abordable (un peu plus d'une vingtaine d'euros) ont réussies à convaincre les professionnels.

La carte Arduino a la particularité d'être en matériel libre, c'est-à-dire d'avoir ses plans de constructions en Open Source. Il n'y a aucuns droits d'auteurs. Ainsi, n'importe qui peut avoir accès, en toute légalité, aux plans de la carte qui sont donc accessibles en libre-service.

b/ Plan de fonctionnement

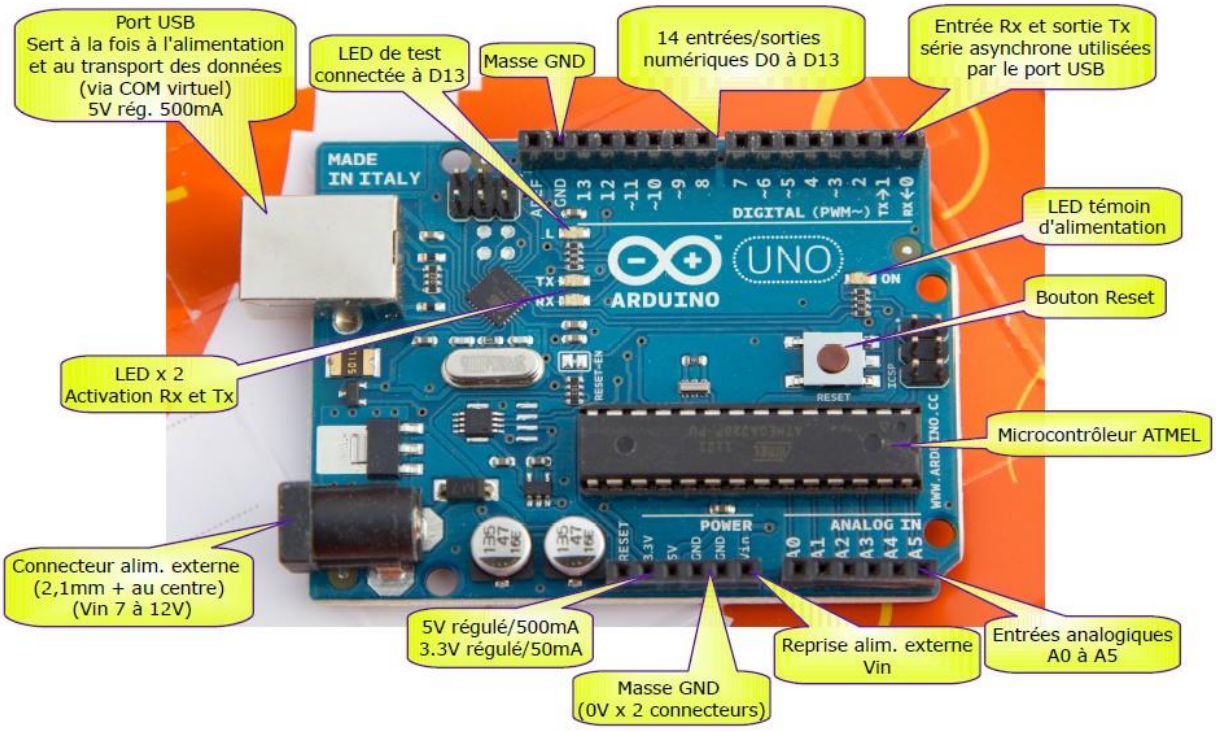


Schéma de la carte, d'après https://encrypted-tbn3.gstatic.com/images?q=tbn:ANd9GcS7vXyNxEL710hC8sOFIbQ7v6VIwIjQHreYQzHSpL0wQbvVwAkn

La carte Arduino peut être séparée en 6 parties disctinctes :

Le microcontroleur

D'une part, celle ci est dotée d'un microcontrôleur programmable de la marque ATMEL, de type Atmega328. Celui ci joue le rôle de « cerveau » de la carte Arduino, c'est à dire que c'est le microcontrôleur qui va déterminer le fonctionement de la carte. Le microcontrôleur possède une partie programmée par le fabricant qui commande le fonctionnement général interne de la carte arduino. Cette partie ne peu être programmée par l'utilisateur. Le microcontroleur possède aussi une partie programmable par l'utilisateur c'est à dire que ce dernier peut y transmetre un programme éxécutable directement via l'ordinateur via le port USB de la carte. Le programme envoyé par l'ordinateur au microprocesseur va déterminer l'interaction entre la carte arduino, et ses périphériques d'entrés (broches, boutons) et les sorties (broches, leds ...). Le programme est sauvegardé dans une mémore Flash de capacité 32 KB. Plus de détails sur http://www.atmel.com/Images/doc8161.pdf

Les broches

La carte Arduino Uno possède 14 broches numériques (donc qui ne peuvent délivrer et analyser que deux hauteurs de tensions : la tension nulle LOW et la tension HIGH). Parmi elles, 6 peuvent fonctionner en mode PWM. La carte possède aussi 6 broches d'entrée analogique (qui peuvent servir à déterminer une tension de manière précise). Elles peuvent être utilisés ausi bien en tant qu'entrés, qu'en tant que sorties. La carte Arduino possède aussi deux ports délivrant une tension fixe, l'un de 5V (mais avec une très faible intensité électrique), le second de 3.3 V. La carte possède aussi deux broches faisant office de masses (donc à 0V), elles sont marqués GND comme « ground » en Anglais qui vaut dire « sol ».

Le circuit imprimé

Le circuit imprimé est le circuit électronique de couleur bleu qui sert à relier les composants entre eux. Ce circuit électronique sert de « squelete » qui relie les composants entre eux.

Les leds

Les leds sont des sorties de la carte arduino, qui s'alument (couleur verte) quand le programme transmis par l'utilisateur, leur envoi un courant.

Les boutons

Les boutons servent d'entré à la carte Arduino. Un de ces boutons, le bouton reset, sert à réinitialiser la carte en supprimant le programme envoyé par l'utilisateur à la carte.

Les ports

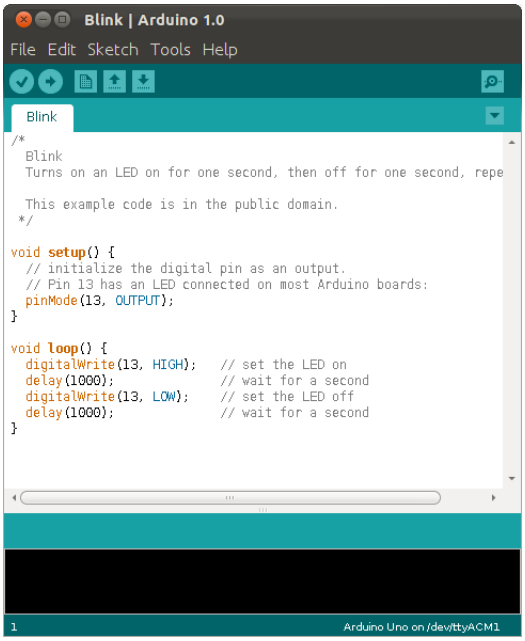
Il y en a deux. Le port USB sert à relier l'ordinateur au microprocesseur de la carte Arduino. C'est par ce port qu'est transmis le programme de l'ordinateur. Le port jack sert à relier l'alimentation électrique à la carte Arduino.

c/ Le modèle Uno

La carte Arduino existe sous de très nombreux modèles tels que Nano, Mega, Leonardo, Duemilanove, Diecimila, Fio, Esplora … Ici, nous utiliserons une carte Arduino de modèle Uno. Ce modèle est quasiment de même conception que la carte Arduino Duemilanove à la différence près que celle-ci utilisait une puce FTDI contrairement à l'Arduino Uno qui utilise une puce ATmega8U2. Comme toutes les autres cartes Arduino, la carte Uno est programmable, possède une sécurité contre les surcharges de courant.

d/ Principe de programmation de la carte Arduino

La carte Arduino est un circuit imprimé programmable, c'est-à-dire un circuit électronique auquel on peut transmettre une suite logique d'instructions à suivre et qui détermine son fonctionnement : le script. L'exécution de ce script permet d'obtenir (virtuellement) un circuit électronique possédant quasiment le même mode de fonctionnement que le circuit électronique dédié que l'on aurai utilisé si l'on avait voulu créer un circuit à part.



La programmation de la carte Arduino se fait depuis l'ordinateur, dans l'application gratuite fournie avec la carte. Le langage de programmation utilisé est un langage proche du C, adapté aux spécificités de fonctionnement de la carte Aduino. Grâce au logiciel programmé en Java fournit par Arduino, nous pouvons créer notre script en le tapant via le clavier, le compiler (c'est-à-dire transformer son langage d'écriture en un autre langage compréhensible par la machine), vérifier les erreurs de syntaxe, si il y en a, grâce au programme de vérification d'erreurs inclus dans le logiciel.

Une fois le programme compilé, on peut le transmettre au microcontroler ATMEL ATmega328 de la carte Arduino via le câble USB fourni avec celle-ci. Cette opération s'appelle le téléversement, et permet de modifier le fonctionnement du microcontrôleur de la carte. Son fonctionnement modifié, permet d'obtenir un circuit logique fonctionnant de la même manière que celle décrite dans le programme.

e/ Fiche technique de la carte Arduino

Voici la fiche technique de la carte Arduino (d'après le site http://www.arduino.cc/en/Main/ArduinoBoardUno) :

- Microcontroller : ATmega328
- Voltage Opérationnel : 5V
- Voltage d'entrée (recommandé) : 7-12V
- Voltage d'entrée(limite) : 6-20V
- Nombre de broches numériques : 14 (dont 6 fonctionnent en sortie PWM)
- Nombre de broches analogiques : 6
- Courant continu par broche : 40 mA
- Courant continu par broche 3.3V : 50 mA
- Memoire Flash : 32 KB dont 0.5 KB utilisés par le bootloader
- SRAM : 2 KB
- EEPROM : 1 KB
- Vitesse d'horloge: 16 MHz
- Longueur : 68,6 mm
- Largeur : 53,4 mm
- Poids : 25 g
- Lieu de fabrication : Italie

3/ Fonctionnement en modulation de largeur d'impulsion

a/ La carte Arduino et le PWM

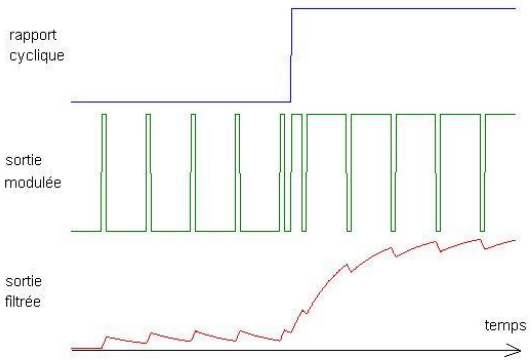
La modulation de largeur d'impulsion (en anglais, Pulse With Modulation, PWM) est une technique couramment utilisée en électronique pour générer des signaux continus à l'aide de circuit ne délivrant que des signaux discrets binaires (donc qui ne peuvent prendre que deux valeurs, le « tout » ou le « rien »). Le signal continu résultant est obtenu en faisant la moyenne des signaux sur une période donnée. Ainsi, le signal résultant peut prendre n'importe quelle valeur intermédiaire comprise entre le « tout » et le « rien » et par conséquent simuler n'importe quelle tension comprise dans cet intervalle. Dans notre cas à nous, avec la carte Arduino, cet intervalle va de 0V à 5V (la tension de sortie en PWM ne peut pas être en dehors de cet intervalle).

C'est sur ce principe, que fonctionne certains ports de la carte Arduino. En effet, l'électronique embarquée à l'intérieur ne permettant pas d'envoyer une tension analogique continue, la carte compense en délivrant des impulsions électroniques discrètes et ne pouvant prendre que deux valeurs. Les ports de la carte pouvant fonctionner en PWM sont reconnaissables par leur petite « vague » inscrite à côté de la broche (voir photo ci-contre). On s'aperçoit ainsi que les ports 3, 5, 6, 9, 10, 11 sont compatibles en mode PWM, ce qui n'est pas le cas des ports 0, 1, 2, 4, 7, 8, 12, 13. Leur fréquence de fonctionnement est d'environ 490 Hz.

L'instruction de programme permettant à la carte Arduino de délivrer une tension de sortie spécifique en PWM se fait par *AnalogWrite()* où les arguments dans la parenthèse sont la numéros de la broche concerné ainsi que le niveau de la tension à délivrer par cette même broche. L'instruction *DigitalWrite()* (mêmes arguments) permet à la carte de ne délivrer que deux tensions : la tension haute (High) et la tension basse (low). Pour fonctionner en PWM, nous n'utiliserons donc que *AnalogWrite()*.

b/ Fonctionnement

Le principe fonctionnement du PWM est celui ci :



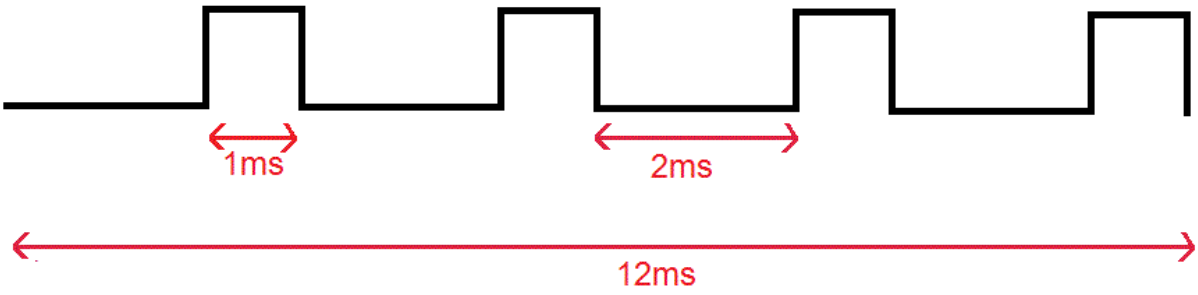
La carte Arduino génère un signal numérique valant soit 0 (tension instantanée basse), soit 1 (tension instantanée haute). Comme le signal de sortie ne peut prendre que deux types de valeurs seulement à certains moments donnés, dit qu'il s'agit d'un signal discret.

En un rapport cyclique T, la durée t durant laquelle la tension de sortie reste 1 (c'est-à-dire la période séparant deux tensions instantanées basses à 0) reste constante. Elle ne pourra varier qu'à la fin du rapport cyclique (voir figure ci-contre). Pour obtenir la tension ỹ, simulée par la carte Arduino, il faut calculer la valeur moyenne de la tension du signal sur la rapport cyclique T. Pour ce faire, on peut utiliser la formule de calcul de la moyenne sur des valeurs continues :

$$\bar{y} = \frac{1}{T}\int_0^T f(t)\,dt.$$

c/ Exemples d'utilisation du PWM

Prenons par exemple le signal suivant :



Celui-ci a une période de 12ms. Il vaut 1V, quatre fois en une période et ce, pendant 1ms à chaque fois. Il vaut 0V, quatre fois pendant une période et ce, pendant 2ms à chaque fois.

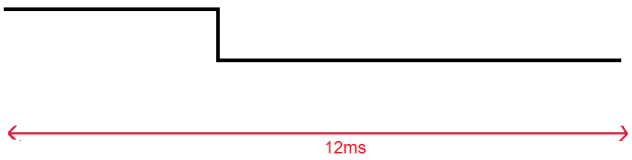
Avec toutes ces données, nous pouvons calculer la tension moyenne délivrée par la carte Arduino, dans ce cas présent, grâce à la formule de calcul de la moyenne sur des valeurs continues :

$\bar{y} = \frac{1}{T}\int_0^T f(t)\,dt.$ Ici T vaut 12ms et f(t) est la fonction définie tel que $\forall t \in \mathbb{R}, \begin{cases} f(t) = 1 \; si \; t \; est \; multiple \; de \; 3 \\ f(t) = 0 \; sinon \end{cases}$

On a donc :

$$\tilde{y} = \frac{1}{12}\int_0^{12} f(t)dt$$

Le signal peut être assimilé au signal équivalent suivant (cela ne modifiant en rien le résultat de nos calculs) :



$$\Leftrightarrow \tilde{y} = \frac{1}{12}\int_0^4 1dt$$

$$\Leftrightarrow \tilde{y} = \frac{1}{12}\int_0^4 1dt$$

$$\Leftrightarrow \tilde{y} = \frac{1}{12}[t]_0^4$$

$$\Leftrightarrow \tilde{y} = \frac{4}{12} = \frac{1}{3}$$

Donc, nous en déduisons que la tension moyenne qui sera délivrée par la carte Arduino en PWM vaut 1/3 V soit environ 0.33 V

4/ Analyse de tension électrique

a/ Introduction

La carte Arduino est un circuit intégré pouvant à la fois délivrer une tension en PWM (comme nous l'avons montré dans la partie précédente) et analyser cette tension électrique. En effet, la carte Arduino est équipée d'un convertisseur analogique-numérique (CAN) qui lui permet d'analyser une tension électrique analogique reçue par un port adapté (seulement ceux marqués comme étant « analogiques » et donc commençants par la lettre A (exemple A0, A1 …). Le CAN de la carte Arduino va convertir cette tension en tension numérique capable d'être analysée et traitée par la carte. Celui-ci possède une résolution de 10 bits.

b/ Les limites de la carte Arduino

Le CAN a, selon le site officiel Arduino, un quantum de tension de 4,9mV. C'est-à-dire que le CAN a une précision de 4,9mV et ne peut pas détecter de variation de tension inférieure à ce quantum. Par ailleurs la carte Arduino ne peut, toujours selon les mêmes sources, pas analyser plus d'une tension pour un délai de 100 microsecondes. C'est-à-dire qu'une variation de tension supérieure à ce délai ne pourra pas être analysée correctement par la carte Arduino.

c/ Lecture de tensions

La carte Arduino possède deux manières particulières pour lire une tension électrique. Celles-ci s'écrivent sous forme de lignes de codes et possèdes chacune leurs spécificités. Voici un aperçu des deux principales lignes de codes pour analyser une tension :

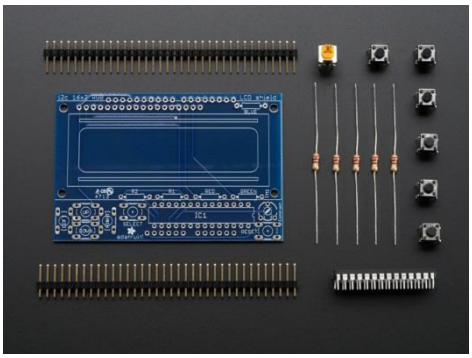
Pour pouvoir lire une tension continue, pouvant prendre de nombreuses valeurs au cours du temps, nous utiliserons plutôt la fonction *analogRead()* dont la signature prend en argument le numéro de la broche analysée. Cette fonction retourne un entier compris entre 0 et 1024. Dans le cas de notre projet, la fonction *analogRead()* sera appliquée à l'analyse de la tension aux bornes de la résistance en série avec le moteur (voir la partie Mesure de l'intensité traversant le moteur)

Pour pouvoir lire une tension binaire, c'est-à-dire ne pouvant prendre que deux valeurs, la valeur haute (High) et la valeur basse (Low) avec la valeur Low ayant une tension nulle. Dans ce cas ci, nous n'avons pas besoin d'utiliser un CAN. En effet, le CAN sert avant tout à convertir une tension analogique en tension numérique. Comme nous avons une tension numérique à la base, nous pouvons directement relier la carte Arduino à la sortie du montage délivrant la sortie binaire. Nous utiliserons alors un *digitalRead()* dont la signature prend en argument le numéro de la broche analysée. Cette fonction retourne soit 0, soit 1 en fonction du signal d'entrée. Dans notre projet, nous utiliserons un *digitalRead()* pour lire une impulsion résultant de d'un montage comparateur (voir la partie Principe de mesure de la vitesse du moteur).

5/ L'écran LCD

a/ Présentation de l'écran LCD

Pour connaitre la vitesse du moteur, ainsi que la tension délivrée à celui-ci, nous allons avoir besoin d'un écran LCD. Cet écran va nous permettre d'afficher toutes ces informations. Pour des raisons de compatibilité, nous utiliserons un écran LCD de type Adafruit I2C Controlled. En effet, ce composant électronique a été spécialement conçu pour fonctionner avec la carte Arduino Uno que nous utilisons pour commander la vitesse du moteur et analyser des mesures. La datasheet se trouve à l'adresse : http://www.adafruit.com/products/715



L'affichage sur cet écran est codé directement dans le programme que nous transmettons au microprocesseur de la carte Arduino ce qui permet une excellente compatibilité et rapidité entre la carte et l'écran.

De plus, l'écran peut se connecter directement à la carte Arduino via ses broches en se « clipsant » sur celle-ci via les broches de la carte. Les boutons présents sur cet écran peuvent être utilisés par la carte Arduino qui peut les analyser en tant qu'entrés.

b/ Spécifications techniques

- Dimensions : 2.1" x 3.2"
- Nombre de caractères affichables : 16x2
- Ecran "Plug and play" avec n'importe quelle catre Arduino 'classique' - UNO, duemilanove, diecimilla, etc as well as Arduino Mega R3.
- Uses only the I2C pins - Analog 4 & 5 on classic Arduinos, Digital 20 and 21 on Arduino Mega R3
- This board/chip uses I2C 7-bit address 0x20

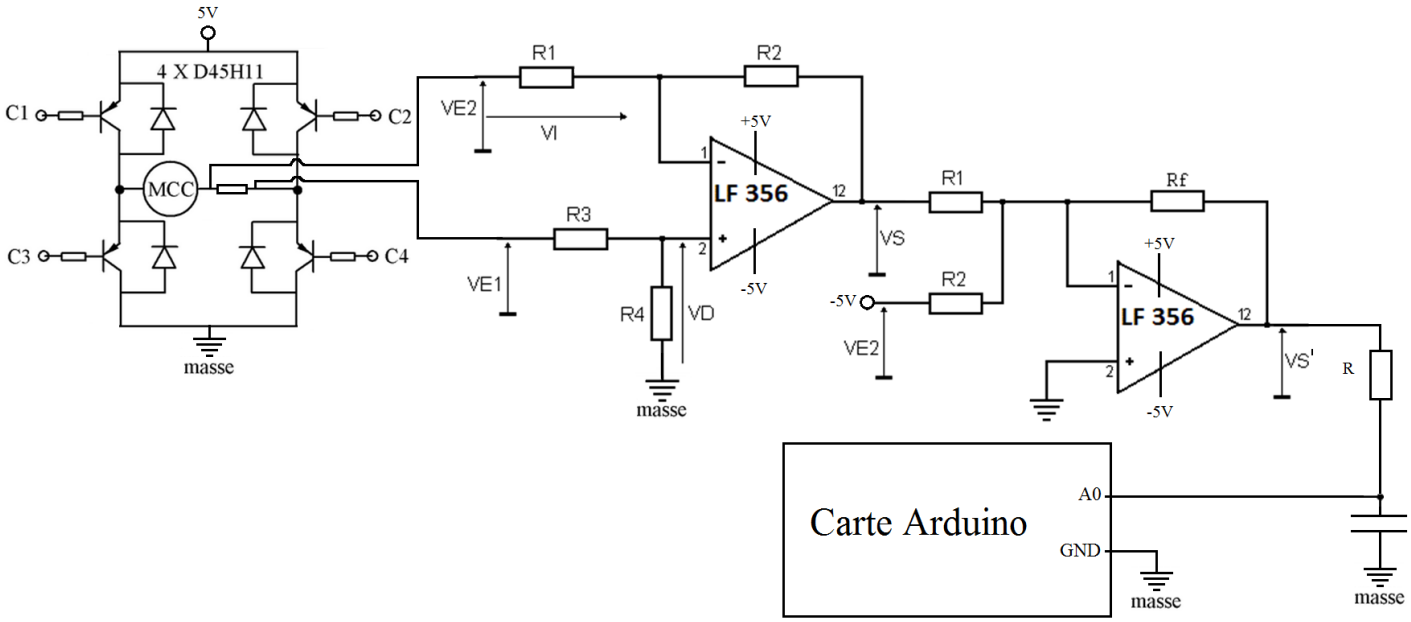
Partie III : Mesure de l’intensité traversant le moteur

1/ Introduction

Dans le cadre de notre cahier des charges, nous allons devoir mesurer la tension aux bornes d'une résistance en série avec le moteur. La mesure de la tension aux bornes de la résistance est indispensable pour connaître l'intensité électrique aux bornes du moteur et donc par conséquent, la puissance aux bornes du moteur. En effet, en plus de la valeur de cette tension, nous connaissons la valeur de la résistance aux bornes de la résistance en série avec le moteur, ce qui nous permet de disposer de toutes les valeurs nécessaires au calcul de l'intensité électrique aux bornes du moteur ainsi qu'au calcul de la puissance consommée par le moteur.

Pour pouvoir mesurer cette tension, nous allons utiliser un montage tel que celui schématisé ci-dessous où une résistance de puissance de 2,2 Ω est montée en série avec le moteur. A partir de cette résistance, nous utiliserons un amplificateur différentiel, un amplificateur sommateur, et enfin, un circuit RC.

Le schéma du circuit est exposé ci-dessous :



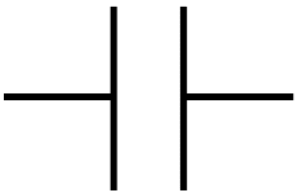
2/ Découplage des alimentations

a/ Introduction

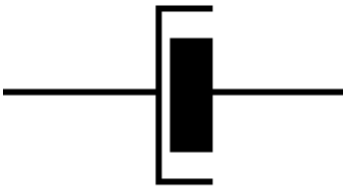
Le fait de connecter plusieurs circuits à la même alimentation peut causer problème. En effet, l'alimentation électrique reliée au circuit a tendance à envoyer de trop hautes harmoniques au circuit, ce qui peut engendrer des perturbations électriques nuisibles au bon fonctionnement du circuit. C'est pourquoi, pour assurer le bon fonctionnement de notre montage, nous utiliserons un circuit à base de condensateurs.

b/ Le condensateur

Le condensateur est le dipôle électrique qui va nous permettre de lisser le signal de sortie. Il s'agit d'un dipôle composé de deux armatures conductrices espacés par une petite distance tel que : $i(t) = C * \frac{du(t)}{dt}$ avec i, l'intensité traversant le condensateur, u la tension à ses bornes et C la capacité du condensateur (exprimée en farads) qui dépend de plusieurs facteurs tel que l'espacement entre les deux armatures où la superficie des deux armatures. En notation complexe et en courant alternatif, nous pouvons écrire l'impédance du condensateur Zc en fonction d'une pulsation ω (ω = 2πf) comme ceci :



Symbole du condensateur non polarisé



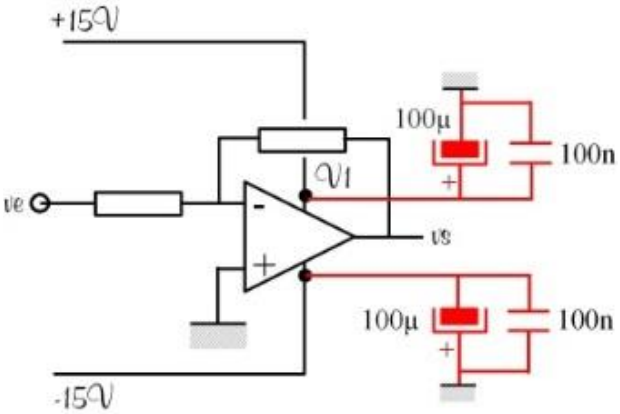
Symbole du condensateur polarisé

$$Z_C(\omega) = \frac{1}{jC\omega}$$

Si la tension d'entrée u est continue, alors sa dérivée par rapport au temps s'annule et l'intensité i est nulle est donc le courant est égal à 0.

Le condensateur peut ne pas être polarisé comme il peut l'être. Tout dépend du condensateur utilisé. La seule chose à savoir est que le condensateur polarisé ne doit pas être branché du mauvais coté, sinon, celui ci sera très sérieusement endomagé

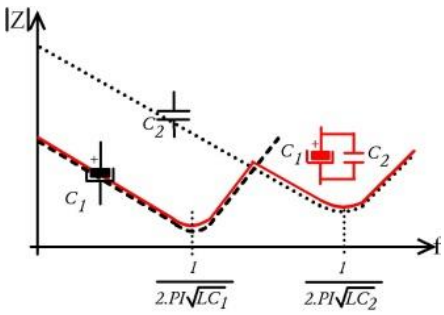
c/ Fonctionnement

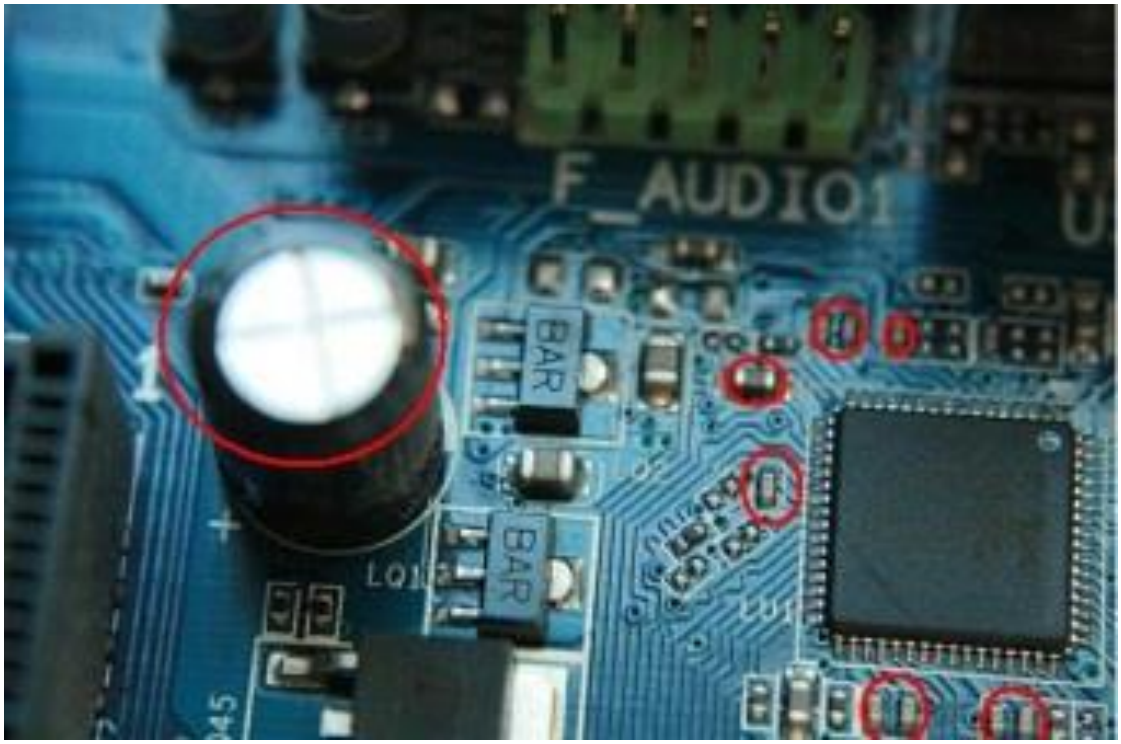


Pour éviter ce genre de nuisances, on place aux bornes des alimentations communes au circuit, plusieurs capacités. En effet, le module de l'impédance du condensateur étant sous la forme $\frac{1}{C\omega}$ avec ω = 2πf et f, la fréquence du circuit. On voit donc que plus la fréquence est élevée, plus l'impédance est faible. Ainsi, les hautes fréquences sont éliminées par le condensateur ce qui augmente par conséquent l'immunité magnétique du système.

Concrètement, le découplage des alimentations est effectué par la mise en place de deux capacités misent en parallèle. Les caractéristiques du circuit sont les suivantes :

Les capacités de découplage sont aussi très utilisées en électronique tel que dans les ordinateurs où l'on peut facilement apercevoir des capacités de découplage tel que sur la carte mère d'un PC (photo ci-contre)



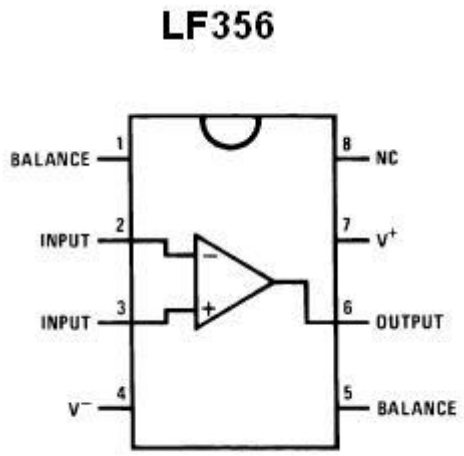


3/ L'amplificateur LF356

a/ Introduction

L'amplificateur LF356 est le composant électronique qui va nous servir à amplifier la tension de sortie VS en fonction des tensions d'entrés VE1 et VE2 dans l'amplificateur différentiel (voir partie suivante). En effet, la tension aux bornes de la résistance de puissance est trop faible pour la carte Arduino et ne peut donc pas être directement mesurée par la carte Arduino. Nous allons donc avoir besoin d'un amplificateur opérationnel : le LF356 de Texas Instrument. Le lien vers la datasheet du composant est : http://www.ti.com/lit/ds/symlink/lf357.pdf

b/ Fonctionnement



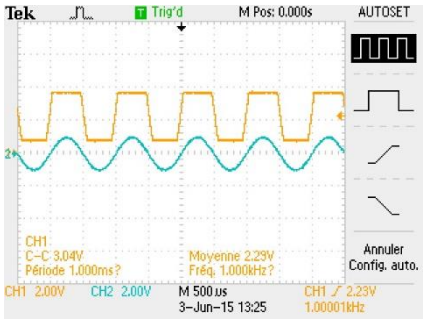
Le LF356 possède en tout 8 broches. Nous ne les utiliserons par toutes car certaines nous sont inutiles pour l'usage que nous comptons en faire. Nous n'utiliserons donc que les broches suivantes :

- INPUT 2 : La borne V-
- INPUT 3 : La borne V+
- V- 4 : L'entrée de l'alimentation - de l'ampli
- OUTPUT 6 : La sortie VS de l'ampli
- V+ 7 : L'entrée de l'alimentation + de l'ampli

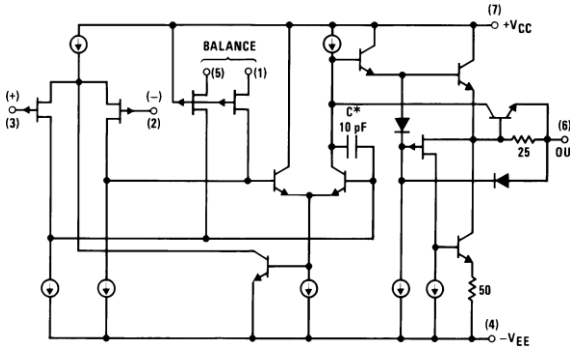
Nous ne nous servirons donc pas des broches BALANCE (1) et (5) ainsi que de la broche NC (8) qui ne nous sont pas utiles pour tous les montages que nous comptons réaliser par la suite.

c/ Fiche technique

- Plus faible courant entrant possible: 30pA
- Plus haut courant entrant possible: 3pA
- Plus grande impédance d'entrée: 1012Ω
- Plus bas bruit de courant: 0.01 pA/√Hz
- Plus grand ratio de rejection en mode: commun: 100 dB
- Plus grand gain en courant continu: 106 dB



Le LF356 sature et écrête à partir d'une certaine tension



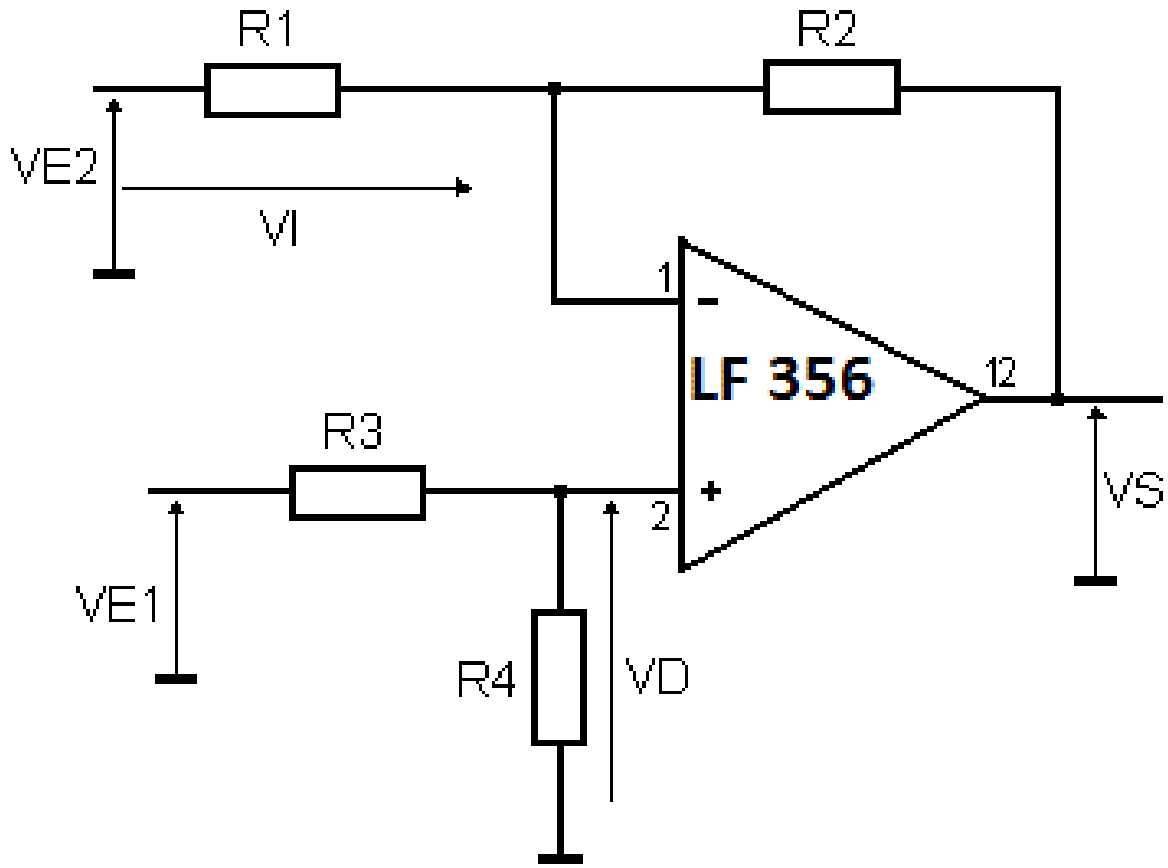
Shéma interne simplifié du LF356

4/ Principe de fonctionnement de l'amplificateur différentiel

a/ Introduction

Appelé aussi amplificateur soustracteur, l'amplificateur différentiel est un circuit électronique va être utilisé en tant qu'intermédiaire entre la résistance dont on veut mesurer la tension entre ces deux bornes, et la carte Arduino qui va analyser cette tension et l'afficher textuellement sur un écran LCD.

Ce montage à pour but d'amplifier une différence de tension entre ces deux entrés V1 et V2 pour former une tension VS augmentée, d'où le nom d'amplificateur différentiel. C'est un montage assez simple qui fait appel qui fait appel à un amplificateur opérationnel (AOP), le LF 356 dont la présentation a été des détaillé précédemment. L'amplificateur différentiel fait aussi appel à quatre résistances dont nom détermineront la résistivité dans le prochain paragraphe. Ces résistances déterminent l'amplification, et donc déterminent le gain du système. La sortie VS est directement reliée à la carte Arduino.



b/ Etude de l'amplificateur différentiel

La première chose à faire pour déterminer le fonctionnement d'un circuit faisant appel à un amplificateur opérationnel (ici le LF 356), est de regarder le branchement de sa sortie par rapport à ses entrées. Ici, on remarque que la sortie VS de l'AOP est reliée à la borne V- de l'amplificateur LF356, via la résistance R2. Cela implique donc que l'amplificateur est en mode linéaire : on a donc une contre-réaction négative. La tension V+ est ainsi égale à la tension V-. Ce résultat va nous permettre de calculer la sortie VS en fonction des entrés VE1 et VE2. On calcule :

$$V_{moins}=\frac{\frac{VE2}{R1}+\frac{VS}{R2}}{\frac{1}{R1}+\frac{1}{R2}} \Rightarrow V_{moins}=\frac{\frac{VE2\times R2+VS\times R1}{R2\times R1}}{\frac{R1+R2}{R1\times R2}} \Rightarrow V_{moins}=\frac{VE2\times R2+VS\times R1}{R1+R2}$$

$$V_{plus}=\frac{\frac{VE1}{R3}+\frac{0}{R4}}{\frac{1}{R3}+\frac{1}{R4}} \Rightarrow V_{plus}=\frac{\frac{VE1\times R4+0\times R4}{R3\times R4}}{\frac{R3+R4}{R3\times R4}} \Rightarrow V_{plus}=\frac{VE1\times R4}{R3+R4}$$

$$V_{plus}=V_{moins}$$

$$\Rightarrow \frac{VE2\times R2+VS\times R1}{R1+R2}=\frac{VE1\times R4}{R3+R4}$$

$$\Rightarrow \frac{VE2\times R2}{R1+R2}+\frac{VS\times R1}{R1+R2}=\frac{VE1\times R4}{R3+R4}$$

$$\Rightarrow \frac{VS\times R1}{R1+R2}=\frac{VE1\times R4}{R3+R4}-\frac{VE2\times R2}{R1+R2}$$

$$\Rightarrow VS=\frac{R1+R2}{R1}\times\left(\frac{VE1\times R4}{R3+R4}+\frac{(-VE2)\times R2}{R1+R2}\right)$$

$$\Rightarrow VS=VE1\times\frac{R1\times R4+R2\times R4}{R3\times R1+R4\times R1}-VE2\times\frac{R1\times R2+R2\times R2}{R1\times R1+R1\times R2}$$

$$En\ prennant\ R1=R3\ et\ R2=R4,\ on\ a:$$

$$\Rightarrow VS=(VE1-VE2)\times\left(\frac{R1\times R2+R2\times R2}{R1\times R1+R2\times R1}\right)$$

$$VS=(VE1-VE2)\times\frac{(R1\times R2)\times(1+1/R1)}{(R1\times R2)\times(1+1/R2)}$$

$$\Rightarrow VS=(VE1-VE2)\times\frac{R2}{R1}$$

Nous avons donc finalement :

$$VS=\frac{R2}{R1}*(VE1-VE2)$$

Cela rejoint bien le raisonnement que nous avons exposés dans précédemment : l'amplificateur différentiel multiplie une différence de tension, celle entre VE1 et VE2, par une tension (on l'appelle le gain) égale à une constante, pour former une tension VS. La sortie VS résulte donc bien d'une amplification d'une différence de tension d'où le nom d'amplificateur différentiel.

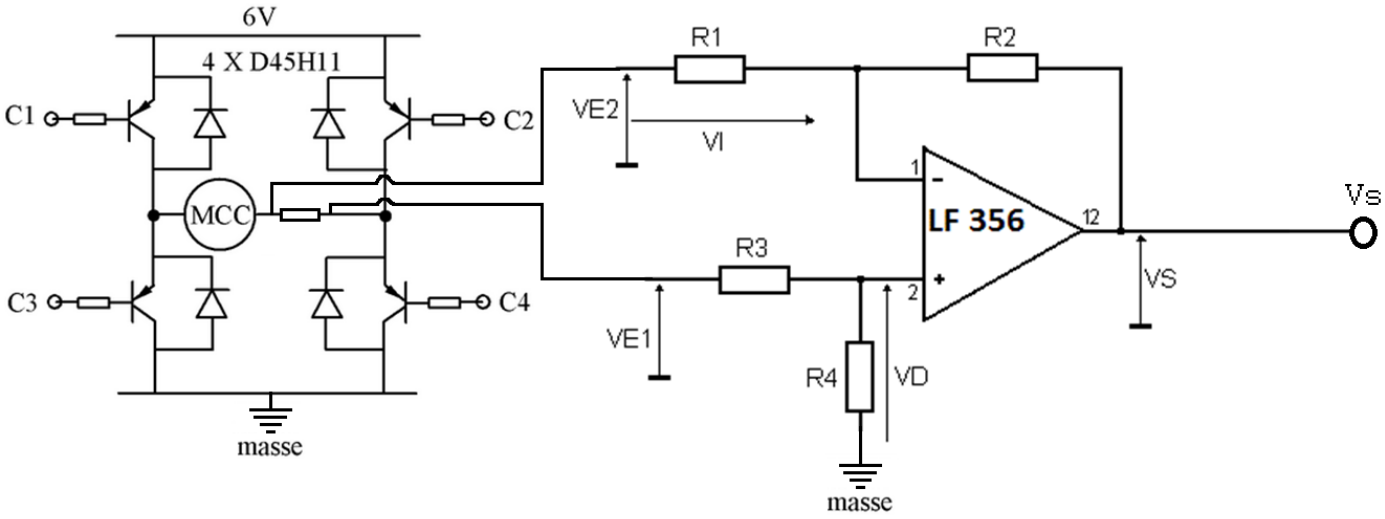
c/ Calcul des résistances nécessaires au fonctionnement de l'amplificateur

Grâce aux calculs exposés précédemment, nous pouvons donc déterminer la valeur des résistances que nous emploierons pour notre montage. Nous désirons avoir une amplification de 10, c'est à dire que VS soit 10 fois supérieur à la différence de tension entre VE1 et VE2. Comme dans notre cas R1 = R3 et R2 = R4, alors il nous suffit de calculer deux valeurs de résistances telle que R2/R1 soit égal à 10.

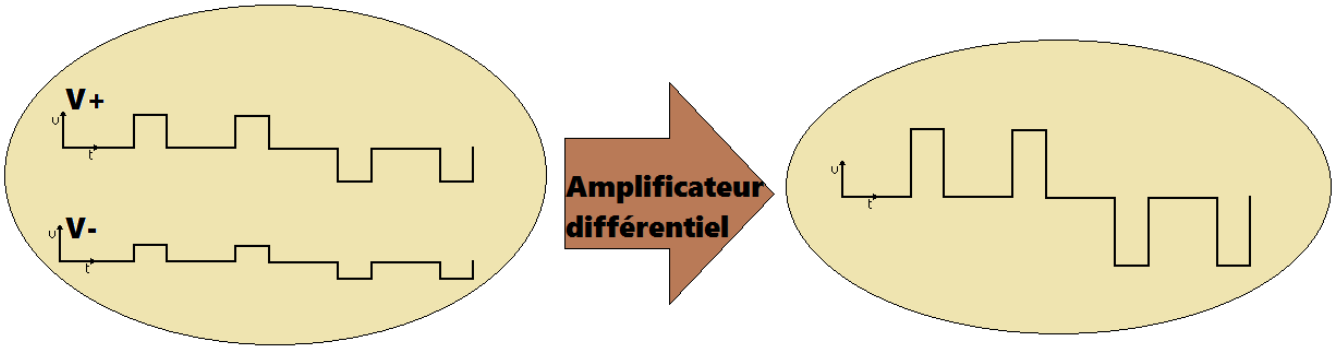
Prenons ici par exemple R1 = R3 = 470 kΩ et R2 = R4 = 4,7 MΩ

Nous avons donc bien R2/R1 = 10. La tension de sortie sera donc égale à 10 fois la différence de tension entre VE1 et VE2.

Finalement, nous obtenons le montage suivant :



Chronogramme d'entrée et de sortie de l'amplificateur différentiel :



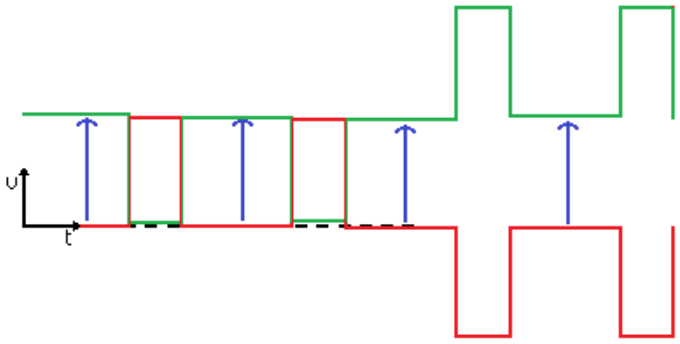
5/ Mise en place d'un offset

a/ Introduction

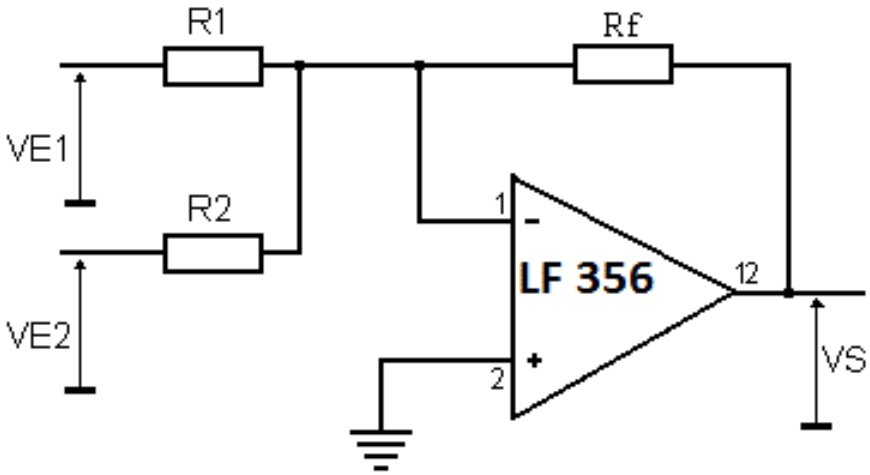
Le signe de la tension de sortie de l'amplificateur dépend de la différence entre VE1 et VE2. Ce qui veut dire en d'autres termes que celle-ci peut éventuellement être négative. Or, la carte Arduino à laquelle est reliée la sortie VS ne peut supporter que des tensions d'entrée positives. En effet, le simple fait de lui délivrer une tension négative peut potentiellement la rendre définitivement hors d'usage ! Il faut donc pouvoir s'assurer que quel que soit la différence de tension entre VE1 et VE2, la tension VS soit positive.

b/ Description du montage

Pour s'assurer que la tension arrivant à la carte Arduino soit positive, nous allons utiliser un montage permettant un offset de tension(c'est-à-dire un décalage de tension) : l'amplificateur sommateur inverseur. L'amplificateur sommateur inverseur peut être modélisé comme une fonction qui a la tension d'entrée V, fait correspondre une tension de sortie Vr telle que Vr soit égale à -V plus une constante, comme sur le diagramme ci-contre. Cette tension d'offset doit être réalisée de manière à ce que l'offset soit suffisamment grand pour s'assurer que la tension de sortie reste toujours positive.



L'amplificateur sommateur inverseur est un montage composé d'un amplificateur LF 356 comme vu précédemment dans le cas de l'amplificateur différentiel, ainsi que de plusieurs résistances. On l'appelle sommateur car cet amplificateur fait la somme des différentes tensions arrivant à sa borne V-. On l'appelle inverseur car la tension d'entrée est connecté à la borne V- de l'amplificateur opérationnel LF356 : la tension de sortie est donc égale à l'opposé de la tension d'entrée plus une constante (éventuellement nulle).



c/ Etude de l'amplificateur sommateur inverseur

A l'instar de l'amplificateur différentiel, la première chose à faire pour déterminer le fonctionnement de ce circuit est de regarder le branchement de sa sortie par rapport à ses entrées. Ici, on remarque que la sortie VS de l'AOP est reliée à la borne V- de l'amplificateur LF356, via la résistance R2. Cela implique donc que l'amplificateur est en mode linéaire : on a donc une contre-réaction négative. La tension V+ est ainsi égale à la tension V-. Ce résultat va nous permettre de calculer la sortie VS en fonction des entrés VE1 et VE2.

$$V_{moins}=\frac{\frac{VS}{Rf}+\frac{VE1}{R1}+\frac{VE2}{R2}}{\frac{1}{R1}+\frac{1}{R2}+\frac{1}{Rf}} \Rightarrow V_{moins}=\frac{\frac{VE1\times R2\times Rf+VE2\times R1\times Rf+VS\times R1\times R2}{R2\times R1\times Rf}}{\frac{R1\times R2+R2\times Rf+Rf\times R1}{R1\times R2\times Rf}}$$

$$\Rightarrow V_{moins}=\frac{VE1\times R2\times Rf+VE2\times R1\times Rf+VS\times R1\times R2}{R1\times R2+R2\times Rf+Rf\times R1}$$

$$V_{plus}=0$$

$$V_{moins}=V_{plus}$$

$$\Rightarrow -\frac{VE1\times R2\times Rf+VE2\times R1\times Rf+VS\times R1\times R2}{R1\times R2+R2\times Rf+Rf\times R1}=0$$

$$Si\ on\ considère\ que\ R1=R2\ ,\ alors\ on\ a:$$

$$-\frac{VE1\times R1\times Rf+VE2\times R1\times Rf+VS\times R1\times R1}{R1\times R1+R1\times Rf+Rf\times R1}=0$$

$$\Rightarrow -\frac{R1\times(VE1\times Rf+VE2\times Rf+VS\times R1)}{R1\times(R1+2\mathrm{Rf})}=0$$

$$\Rightarrow -\frac{VE1\times Rf+VE2\times Rf+VS\times R1}{R1+2\mathrm{Rf}}=0$$

$$\Rightarrow -\frac{VE1\times Rf+VE2\times Rf}{R1+2\mathrm{Rf}}=\frac{VS\times R1}{R1+2\mathrm{Rf}}$$

$$\Rightarrow VS=-\frac{VE1\times Rf+VE2\times Rf}{R1}$$

$$\Rightarrow VS=-\frac{Rf}{R1}\times(VE1+VE2)$$

$$VS=-\frac{Rf}{R1}*(VE1+VE2)$$

On voit donc que la tension de sotie VS est égale à l'addition des tensions VE1 et VE2, le tout multiplié par une constante négative.

On a donc bien l'addition des deux tensions d'entrée, d'où le nom d'amplificateur sommateur.

d/ Utilisation de l'amplificateur sommateur inverseur

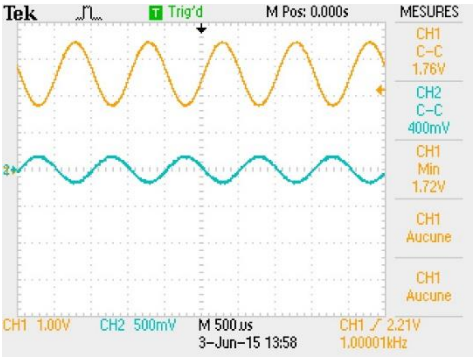
Dans le précédent calcul, nous avons pu démontrer le résultat suivant correspondant à la tension de sortie Vs de l'amplificateur sommateur par rapport aux tensions d'entrés VE1 et VE2 :

$$VS = -\frac{Rf}{R1} * (VE1 + VE2)$$

Nous allons utiliser une résistance Rf telle que Rf soit égale à moitié de la résistance R1 de manière à avoir un gain égal à -0.5 (division par -2 du signal d'entré). On a donc :

$$VS = -\frac{1}{2}(VE1 + VE2) \; car \; \frac{Rf}{R1} = \frac{1}{2}$$

D'autre part, nous allons chercher à effectuer une somme de tensions de manière à obtenir un offset rendant positive la tension de sortie. Nous allons relier le potentiel VE1 à la sortie de l'amplificateur différentiel et le potentiel VE2 à un potentiel de -5V. En effet, on sait que la tension maximale aux bornes de la résistance est toujours inférieure à 5V du fait que la tension du générateur qui alimente le moteur et la résistance est de 5V. Ainsi, la tension VE1 + VE2 sera toujours négative, tout comme la tension -0.5V. Nous nous retrouvons donc à multiplier deux termes négatifs pour calculer VS ce qui implique nécessairement que VS est négative. On voit donc ainsi que la sortie VS est elle aussi toujours positive. Ci contre, la visualisation sur oscilloscope Tektronix de la sortie de l'amplificateur sommateur en série avec l'amplificateur différentiel (en jaune) par rapport à la tension d'entrée de l'amplificateur différentiel (en bleu).



On obtient ainsi :

$$VS = -\frac{1}{2}(VE1 - 5) \; avec \; VE1 \leq 5$$

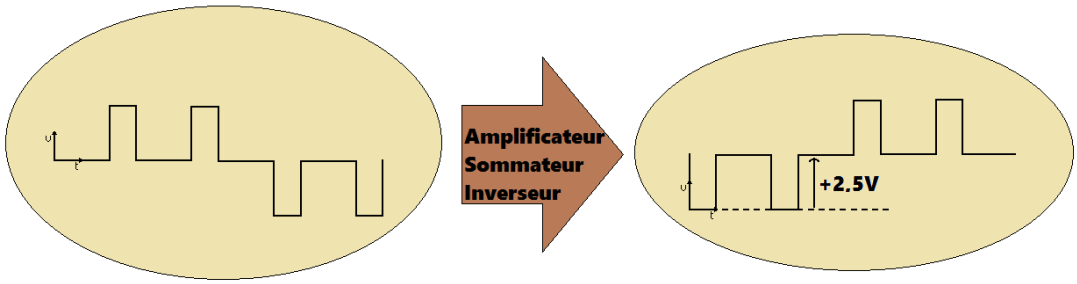
Ce qui peut s'écrire aussi :

$$VS = -0.5VE1 + 2.5$$

On remarque ainsi que l'on ajoute une constante de 2.5V à la tension d'entrée VE1 : il s'agit de l'offset de notre montage. Notre offset est donc de 2.5V. De plus, comme on multiplie VE1 par un facteur négatif (-0.5V), on a une division par deux de l'amplitude et une inversion de la tension (ce qui est normal vu que notre amplificateur sommateur est inverseur).

e/ Conclusion

Le amplificateur sommateur va donc nous servir à redresser le courant entrant en un courant de sortie uniquement positif.



6/ Lissage du circuit : un filtre passe-bas

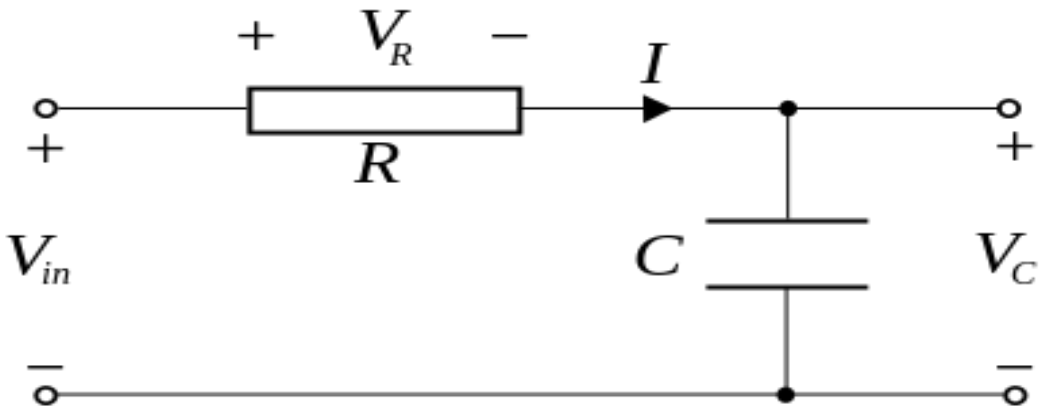
a/ Introduction

Grâce à l'amplificateur différentiel et à un pont de diode, nous avons maintenant une tension de sortie Vs égale à la valeur absolue de la différence de tension entre les bornes de la résistance, multipliée par un rapport de 10. Le dernier problème qui persiste néanmoins est que cette tension, du fait qu'elle est liée au PWM du pont en H, a encore l'allure d'une fonction créneau. Elle n'est donc pas encore mesurable par la carte Arduino. Il va donc nous falloir maintenant, lisser cette tension de sortie de manière à obtenir une tension quasi-continue au cours du temps. Pour ce faire, nous utiliserons un filtre passe bas : le circuit RC. Ce circuit composé d'une résistance et d'un condensateur va nous permettre de lisser le signal en sortie.

b/ Analyse du montage

Calcul de la transmittance :

Le circuit RC est, comme l'acronyme l'indique, un circuit composé d'une résistance et d'un condensateur. La résistance est placée en série avec le pôle Vin+ et le condensateur en parallèle, entre les branche Vc+ et Vc-.



Débutons maintenant le calcul du module de la transmittance du circuit RC :

Par pont diviseur de tension :

$$V_c(\omega)=\frac{Z_c(\omega)}{Z_c(\omega)+R}\times V_{in}(\omega)$$

$$\Rightarrow \frac{V_c(\omega)}{V_{in}(\omega)}=\frac{Z_c(\omega)}{Z_c(\omega)+R}$$

On défini la fonction de transfert $T(\omega)$ par :

$$T(\omega)=\frac{V_c(\omega)}{V_{in}(\omega)}, \text{ car } V_c(\omega)=V_{out}(\omega)$$

On a donc :

$$T(\omega)=\frac{Z_c(\omega)}{Z_c(\omega)+R}$$

De plus, $Z_c(\omega)=\frac{1}{jC\omega}$

On a donc :

$$T(\omega)=\frac{\frac{1}{jC\omega}}{\frac{1}{jC\omega}+R}$$

$$\Rightarrow T(\omega)=\frac{\frac{1}{jC\omega}}{\frac{jC\omega\times R+1}{jC\omega}}$$

$$\Rightarrow T(\omega)=\frac{1}{jC\omega R+1}$$

En posant : $RC=\frac{1}{\omega_0}$, on a $T(\omega)=\frac{1}{j\frac{\omega}{\omega_0}+1}$

Calculons maintenant le module de $T(\omega)$:

$$|T(\omega)|=\frac{1}{\sqrt{1^2+\frac{\omega^2}{{\omega_0}^2}}}$$

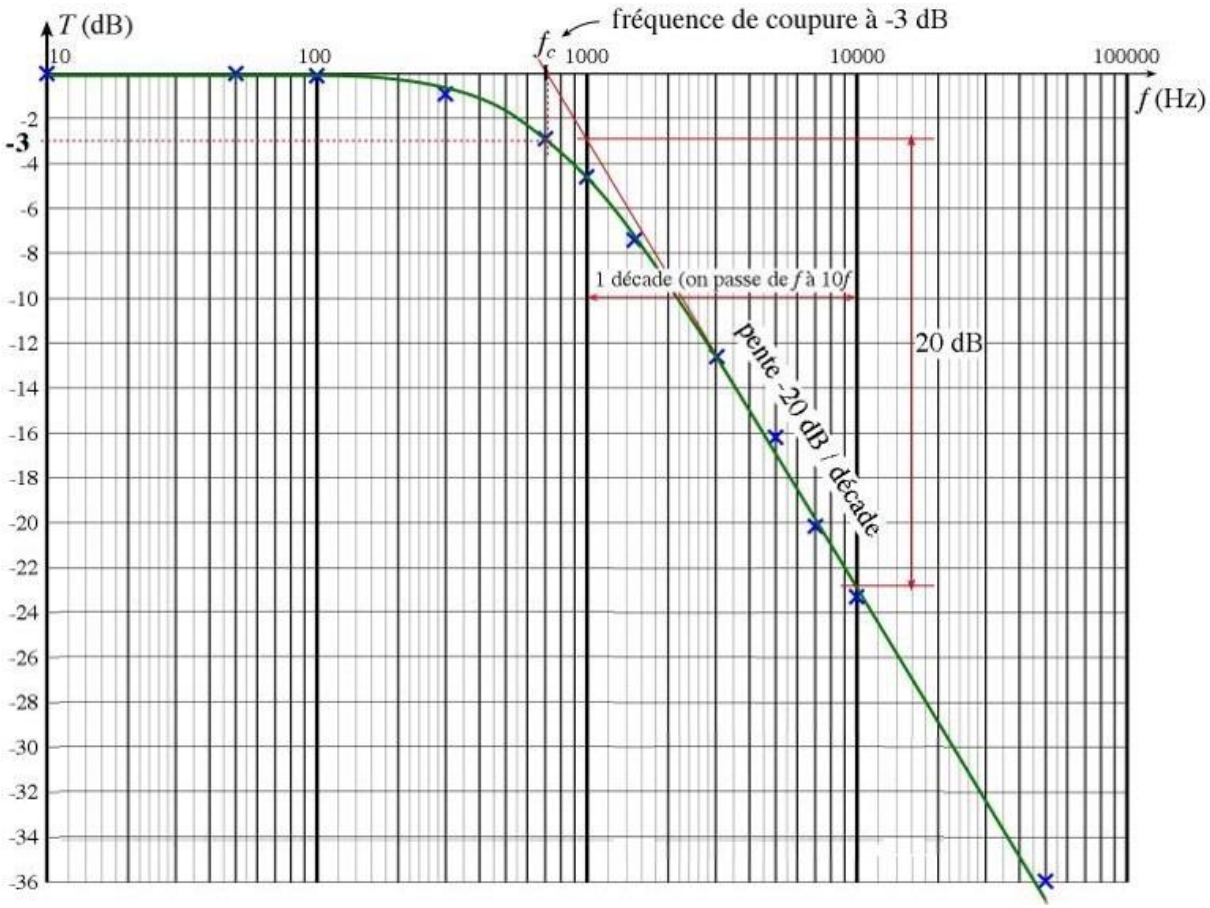
$$\lim_{\omega\to+\infty}|T(\omega)|=0$$

$$\lim_{\omega\to 0}|T(\omega)|=1$$

Allure de la courbe de transmittance :

En exprimant la fonction de transfert en décibel, on a : $T_{décibel}(\omega) = 20 \times \log|T(\omega)|$ ou encore $T_{décibel}(f) = 20 \times \log|T(2\pi f)|$

En prenant à titre d'exemple (en dehors de notre cas à nous) R = 220Ω, C = 1µF (soit une fréquence de coupure de 723 Hz), nous pouvons tracer le diagramme de Bode de cet exemple en traçant $f(\omega) = T_{décibel}(\omega)$ dans un repère logarithmique :



On a un filtre du premier ordre car le gain diminue de 20dB/décade après la fréquence de coupure.

Sur ce diagramme de Bode, on voit clairement que le gain du système diminue de manière monotone à partir de la fréquence de coupure fixée ici à 723Hz. On en déduit donc que le circuit RC est de manière générale un filtre passe-bas, c'est-à-dire un circuit ne laissant passer que les faibles fréquences.

Calcul de la fréquence de coupure

Le fréquence de coupure d'un circuit électronique est la fréquence à partir de laquelle le gain du montage H(ω) est égal au gain max divisé par $\sqrt{2}$ soit : $T(\omega_{coupure}) = \frac{T(\omega)_{max}}{\sqrt{2}}$. La fréquence étant donné par $= \frac{\omega}{2\pi}$, on a $T(2\pi f_{coupure}) = \frac{T(\omega)_{max}}{\sqrt{2}}$.

En remplaçant les termes de cette expression par ceux trouvés précédemment, on a :

$$\frac{1}{jRC2\pi fc + 1} = \frac{1}{\sqrt{2}} \; car \; \lim_{\omega \to 0} T(\omega) = T(\omega)_{max} = 1$$

Donc, cela veut dire que :

$jRC2\pi fc + 1 = \sqrt{2}$

$\rightarrow jRC2\pi fc = \sqrt{2} - 1$

$\rightarrow (RC2\pi fc)^2 = (\sqrt{2} - 1)^2$

$\rightarrow (RC2\pi fc)^2 = (\sqrt{2} - 1)^2$

$\rightarrow (fc)^2 = \frac{(\sqrt{2} - 1)^2}{(2\pi RC)^2}$

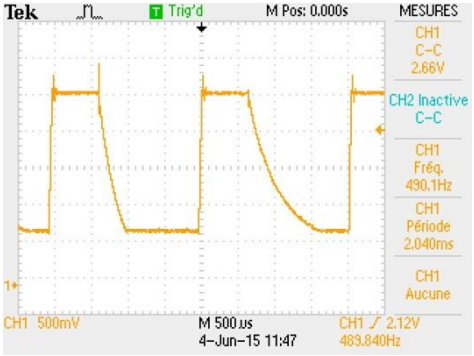
On obtient finalement : $fc = \frac{1}{2\pi RC}$

Utilisation du filtre RC pour notre montage :

Nous avons déterminés la fonction de transfert d'un filtre passe-bas du 1er ordre : le circuit RC. Il ne nous reste plus maintenant qu'à déterminer le rapport que doit avoir la valeur R de la résistance par rapport à la valeur C du condensateur pour nous permettre de lisser le signal en sortie de l'amplificateur sommateur. Il nous faut savoir cependant quelle est la fréquence de coupure de notre montage.

En effet, on sait que :

$fc = \frac{1}{2\pi RC}$ en prenant C=47nF, on a $fc = \frac{1}{2\pi R47{*}10^{-7}}$



Expérimentalement, on mesure une fréquence de 490 Hz en sortie de l'amplificateur sommateur. On souhaite faire disparaitre les pics. On va donc prendre comme fréquence de coupure : fc = 490 Hz.

Finalement, on obtient :

$$R = \frac{1}{2\pi * 490 * 47 * 10^{-7}}$$

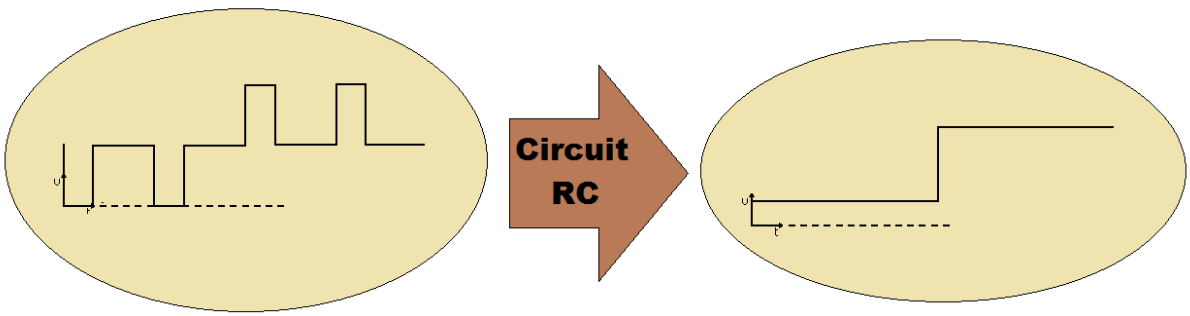
Numériquement, on obtient R = 6910.77 Ω soit 6.91077 kΩ

Pour des raisons techniques, nous arrondirons cette valeur à 6.920 kΩ.

Finalement, pour notre circuit RC, on prend : C=47nF et R=6.920 kΩ

d/ Conclusion

Ainsi, le circuit RC fait office de filtre passe-bas qui va nous permettre de lisser le signal de sortie.



7/ Mesure du courant traversant le moteur

a/ Principe

Le but final de notre chaine de traitement du signal est de permettre le calcul de l'intensité électrique traversant le moteur grâce à la connaissance de la tension de la résistance en série avec le moteur électrique. Nous avons en notre possession la valeur r de la résistance de puissance (qui est dans notre cas de 2.2 Ω) ainsi que la valeur de la tension aux bornes de cette résistance calculée par la carte Arduino.

Grâce à la loi d'Ohm qui stipule que $i(t) = \frac{u(t)}{r}$, nous pouvons donc déterminer l'intensité du courant traversant la résistance en série avec le moteur. Du fait de la loi d'unicité des intensités électriques dans une même branche, nous pouvons en déduire que cette intensité électrique est la même que celle traversant le moteur et donc, déterminer l'intensité électrique traversant le moteur.

b/ Calcul de la tension par la carte Arduino

Calcul de la vrai tension par rapport à la tension mesurée

A l'aide des calculs effectués dans les différentes parties de notre montage électronique composé de l'amplificateur différentiel (d'amplification x10), de l'amplificateur sommateur (d'amplification x0.5 plus d'un offset de -5V) et du circuit RC, nous avons pu établir le résultat suivant :

$$V_{A0} = -\frac{R2}{R1} * \frac{R4}{R3} * (VE1 - VE2) + \frac{R4}{R3} * offset$$

Avec VA0, la tension mesurée par la carte Arduino, offset, la tension rajoutée, R1, R2, R3 et R4 des résistances de notre montage et VE1 et VE2, les deux potentiels aux bornes de la résistance.

Soit :

$$\boldsymbol{V_{A0} = \frac{R4}{R3} * (-\frac{R2}{R1} * (VE1 - VE2) + offset)}$$

En posant U = VE1-VE2, c'est-à-dire la différence de potentiel entre les deux bornes de la résistance et qui est dans notre cas la tension d'entrée du système, nous pouvons exprimer U en fonction des autres termes de notre expression.

$$U = -\frac{R1}{R2}(-\frac{R3}{R4} * V_{A0} - \frac{R4}{R3} * offset)$$

Finalement, nous obtenons :

$$\boldsymbol{U = -\frac{R1}{R2} * \frac{R3}{R4} * V_{A0} + \frac{R4}{R3} * offset}$$

Calcul par la carte Arduino

Ainsi, pour déterminer, la vraie tension aux bornes de la résistance en série avec le moteur, la carte Arduino devra effectuer le même calcul que celui détaillé ci-dessus après avoir mesurée une tension électrique VA0, la tension VA0 étant obtenue après un DigitalRead du port A0 de la carte Arduino.

Une fois la valeur de la réelle tension au bornes de la résistence obtenue, la carte Arduino va pouvoir afficher cette tension sur lécran LCD.

c/ Application numérique

Numériquement, en appliquant ce résultat aux valeurs de notre montage (c'est-à-dire $\frac{R1}{R2} = 10$, $\frac{R1}{R2} = \frac{1}{2}$ et offset = -5), nous obtenons:

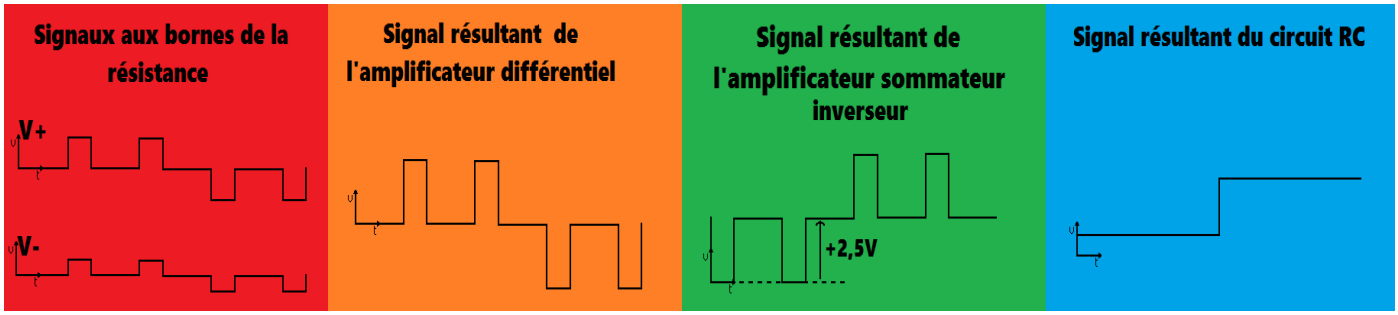
$$\boldsymbol{U = -0,2 * V_{A0} - 2,5}$$

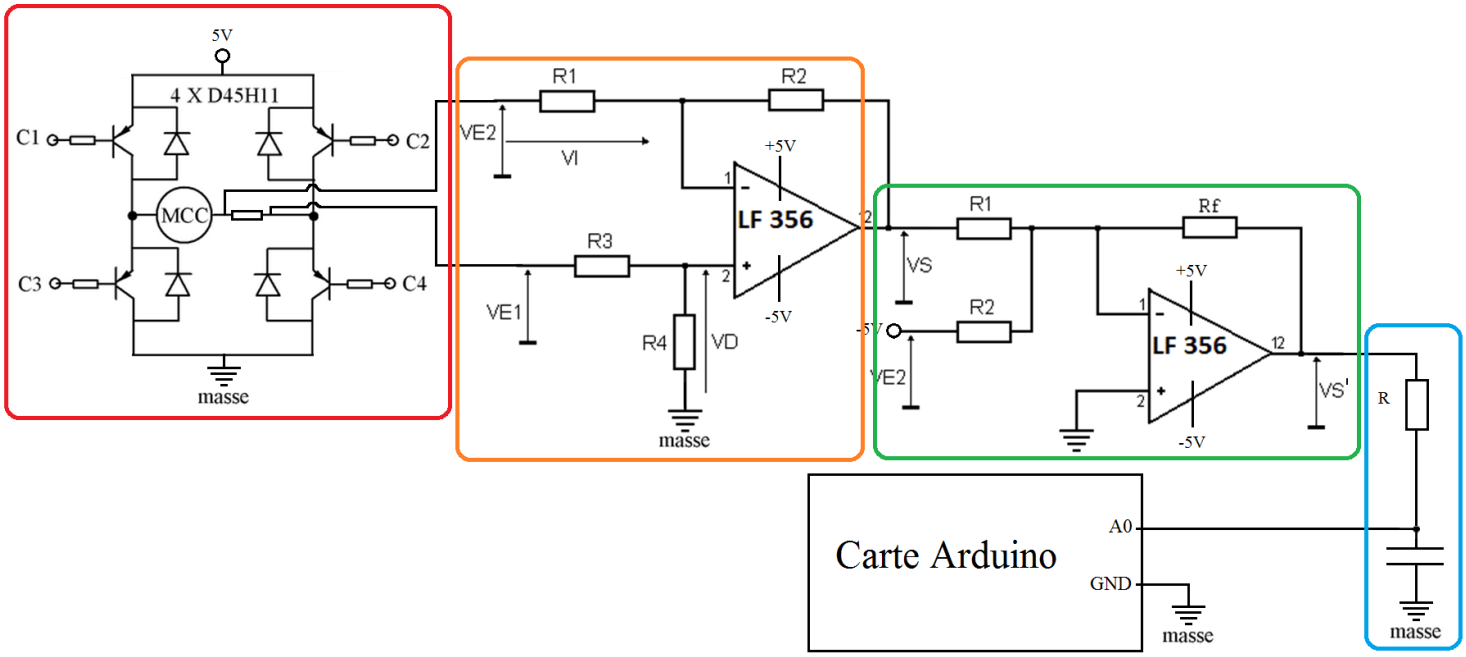
Grâce à la loi d'Ohm, nous pouvons maintenant calculer l'intensité électrique traversant le moteur. Nous obtenons donc le résultat final :

$$\boldsymbol{I = \frac{-0,2 * V_{A0} - 2,5}{2,2}}$$

8/ Récapitulatif des étapes de la mesure de la tension

Voici un récapitulatif de la chaîne de traitement que nous venons d'appliquer au signal :





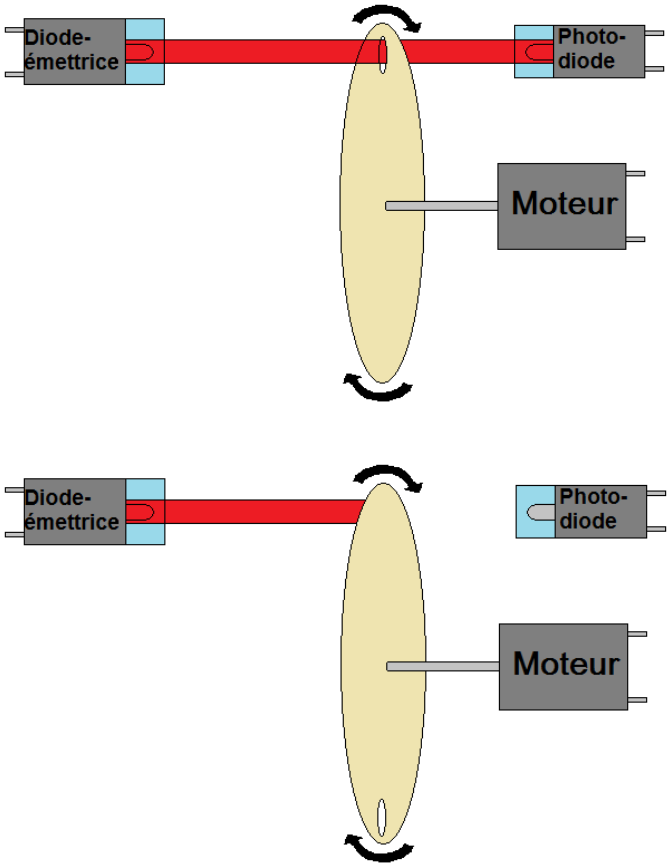
$$V_{A0} = -\frac{R2}{R1} * \frac{R4}{R3} * (VE1 - VE2) + \frac{R4}{R3} * offset$$

Partie IV : Principe de mesure de la vitesse du moteur

1/ Introduction

Nous cherchons maintenant à déterminer la vitesse de rotation du moteur de notre montage. Pour afficher cette vitesse, deux options s'offrent à nous. En premier lieu, nous pouvons afficher directement la vitesse théorique du moteur, la vitesse calculée par le programme de la carte Arduino, celle qui est convertie en tension envoyée aux transistors du pont en H (voir la partie sur le pont en H). L'ennui de cette méthode est qu'elle n'est pas très fiable du fait qu'il s'agit d'une vitesse théorique calculé par la carte Arduino grâce à des variables de son programme. Elle a cependant l'avantage d'être facile à mettre en place. La seconde option qui s'offre à nous est de mesurer la vitesse réelle du moteur grâce à un montage spécifique qui serait externe à la carte Arduino.

Ce montage est constitué d'une diode émettrice, d'une photodiode qui réceptionne la lumière de la diode émettrice, ainsi que d'un disque en carton percé et relié à la tige du moteur. Le moteur entraine donc le disque en carton en même temps que celui-ci tourne. Le disque est percé d'un trou qui laisse passer le faisceau issu de la diode émettrice contrairement aux autres parties non troués qui ne laissent pas passer le faisceau de la diode émettrice. La diode photosensible située à l'opposé du laser par rapport au disque en carton réceptionne le faisceau lumineux en fonction que celui-ci se trouve en face d'un trou ou d'une partie bouchée du disque en carton. La diode photosensible modifie la tension à ses bornes en fonction de l'intensité lumineuse perçue.



La détermination de la vitesse du moteur se fait via la carte Arduino, qui reliée à la photodiode, va analyser la fréquence de changement de potentiels de celle-ci. Plus la fréquence d'alternance de potentiel aux bornes de la photodiode est élevée (et donc plus le faisceau lumineux est coupé rapidement par le disque), plus l'on passe d'un trou du disque à une partie non trouée et ainsi plus le moteur tourne vite.

Enfin la carte Arduino va pouvoir déterminer la vitesse du moteur, en fonction de la fréquence d'alternance du potentiel aux bornes de la photodiode.

2 / Calcul de la vitesse du moteur

Principe

Nous exprimerons la vitesse du moteur en tours par minute, bien qu'il soit possible de l'exprimer d'autres façons (tours par seconde, mètres par secondes, kilomètre par heure …). Il s'agit là de la manière la plus simple de calculer la vitesse du moteur.

Le disque en carton ne comporte qu'un trou circulaire situé à bonne distance de l'arbre du moteur (environ 2cm) pour que le rayon lumineux transitant par ce trou puisse être détecté par la photodiode et que la différence de tension puisse être analysée suffisamment rapidement par la carte Arduino.

La distance séparant deux mêmes positions consécutives du trou correspond à un tour. On en déduit ainsi que comme seul le trou peut laisser passer la lumière de la photodiode, le temps séparant deux alternances consécutives de lumière correspond au temps que met le trou à effectuer un tour sur le disque entrainé sur le moteur. Ainsi, le temps de tour de moteur correspond au temps entre deux alternances lumineuses.

Application à notre montage

Dans notre montage, nous allons effectuer les mesures comme cela :

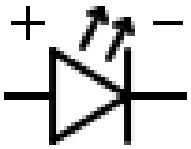
Nous allons prendre une période de mesure égale à 5 secondes. Pendant cette période, la carte Arduino va compter le nombre de d'alternance de tension électrique, donc de lumière reçues par la photodiode et ainsi déterminer le nombre de tours pour 5 secondes.

Pour convertir notre mesure en minutes nous allons multiplier le résultat par 12. En effet, une minute comporte 60 secondes et nous effectuons une mesure pendant une période de 5 secondes. Or il y a 12 périodes de 5 secondes pendant une minute. On a donc 12 fois nos mesures en une minute. Il nous faut donc multiplier notre résultat par 12 pour obtenir le nombre de tours par minute du moteur :

$$V_{moteur} = 12 * Nombre\ d'alterances\ de\ tension$$

3/ La diode électroluminescente

Pour pouvoir émettre de la lumière, nous allons utiliser une diode électroluminescente (DEL en Français ou LED en Anglais). Nous ne nous intéresserons donc qu'à la diode électroluminescente bien qu'il en existe de nombreuses variétés (diode simple, diode zener, photodiode …). Ci-contre, le symbole officiel de la DEL.

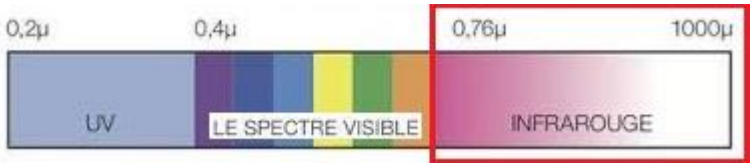


La DEL est un dipôle (c'est-à-dire un composant électrique à deux broches) qui s'avère être polarisé (dont le fonctionnement dépend du sens du courant qui la traverse). Cette jonction polarisée est sous la forme PN où P est la borne de la diode reliée à la tension supérieure à la tension de la borne N. Le sens de branchement de la DEL a donc une importance capitale dans le fonctionnement du circuit. Ce dipôle, à l'instar de n'importe quelle autre diode hormis la zener, possède en effet la particularité de ne laisser passer le courant électrique que dans un seul sens : celui correspondant à la polarisation de la DEL. Elle bloque le courant si elle est branchée en sens inverse à sa polarisation. Ces caractéristiques concernent la majorité des diodes.

De plus, la LED possède la particularité d'émettre de la lumière, d'où le qualificatif d'électroluminescente. En effet, sous l'action du courant, la DEL va produire une lumière quasi monochromatique en transformant directement le signal électrique en une onde lumineuse. Cette transformation à l'avantage non seulement de s'effectuer dans un laps de temps très cours (ce qui rend ce type de diode idéal pour la transmission rapides de données sous forme de lumière), mais aussi sans engager de grosses pertes sous formes de chaleur (effet Joule), ce qui en fait un moyen d'éclairement très économique par rapport aux autres moyens d'éclairement traditionnels tel que la lampe à filament. Enfin, la DEL est très petite et peut s'allumer avec très peu de courant ce qui rend son utilisation idéale pour des circuits intégrés tels que la carte Arduino ou encore ce montage que nous allons effectuer maintenant.

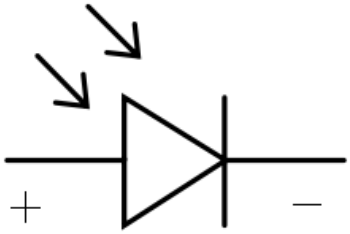
C'est donc pour ces diverses raisons que nous utiliserons donc une DEL pour émettre le faisceau lumineux nécessaire à la détermination de la vitesse du moteur.

Ici, nous utiliserons une DEL émettant dans le domaine de l'infrarouge, c'est à dire dans le domaine des longueurs d'ondes d'environ 0.78µm à 5mm. Ce rayonnement est invisible à l'œil nu du fait que le rayonnement infrarouge ne se situe pas dans le domaine visible.



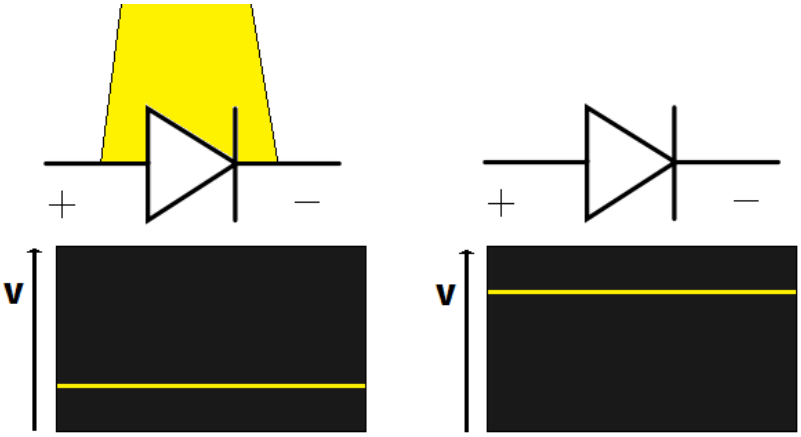
4/ La photodiode

A l'instar de la diode électroluminescente, la photodiode est un dipôle semi-conducteur, donc polarisé, qui à l'opposé de la DEL (qui transforme un signal électrique en signal lumineux), modifie un signal électrique en fonction d'un signal lumineux. C'est pourquoi la photodiode est utilisée pour détecter une intensité lumineuse Ci-contre, le symbole officiel de la photodiode, on remarque qu'en toute logique ce symbole ressemble à celui de la DEL, mais que contrairement à celle-ci, les flèches de rayonnement lumineux sont orientées en direction de la photodiode et non vers l'extérieur.



En pratique, la photodiode diminue la tension électrique en fonction de l'augmentation de l'intensité lumineuse. Ainsi, plus la lumière reçue par la photodiode est importante, plus la tension électrique transmise par la photodiode diminue et inversement, plus la lumière reçue est faible, plus la tension électrique transmise par la photodiode est élevée.

Ainsi, en connaissant la valeur de la tension aux bornes de la photodiode pour un éclairage particulier et la valeur de la tension à ses bornes pour un autre éclairage de valeur différente, nous pouvons déterminer quel est l'éclairement reçu par la photodiode.

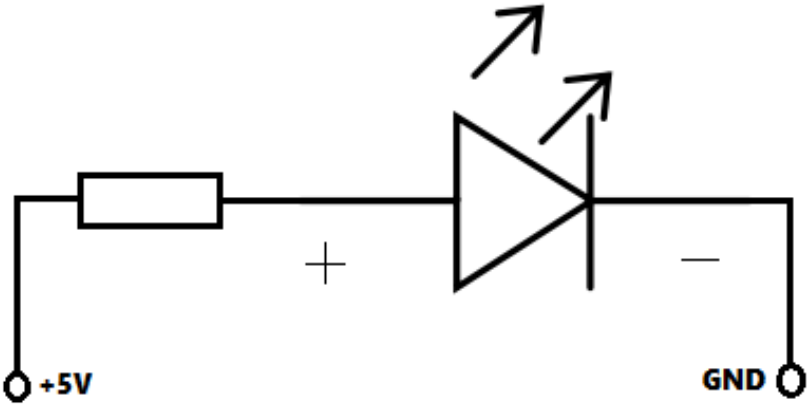


Pour les besoins de notre projet, nous n'avons pas besoin de connaitre une valeur très précise des tensions d'éclairement. Ainsi, ce qui nous importe le plus est de savoir quelle valeur de tension permet de distinguer très précisément la tension de la photodiode non éclairée à la tension de la photodiode éclairée (nous avons donc à séparer deux cas). En effet nous cherchons avant tout à savoir si la photodiode reçoit la lumière de la diode électroluminescence (auquel cas elle se situe face au trou du disque) ou pas (donc qu'elle se situe face à une partie non trouée du disque). En bref, cela nous ramène à chercher une tension binaire (qui ne peut prendre que deux valeurs) qui nous permette de comparer les deux tensions : celle de la photodiode éclairée et celle de la photodiode non éclairée.

5/ Notre montage

a/ Aux bornes de la diode électroluminescente

Pour faire fonctionner correctement notre diode électroluminescente, nous allons l'alimenter en 5V directement depuis la broche 5V de la carte Arduino. Nous relirons par ailleurs sa masse à la broche GND de la carte Arduino. Pour éviter d'endommager la DEL avec un courant trop élevé, nous allons la brancher en série avec une résistance de 10kΩ. Finalement notre montage est le suivant :

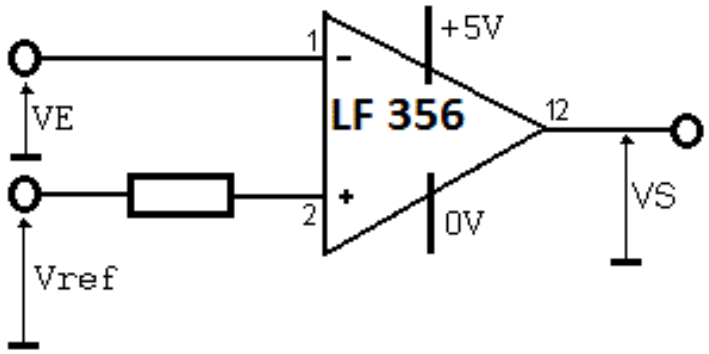


b/ Aux bornes de la photodiode

Avec la photodiode, nous allons pouvoir mesurer de manière qualitative la lumière reçue par celle-ci grâce à la tension à ses bornes. En reliant la borne plus de la photodiode à la carte Arduino, nous nous apercevons que la différence de potentiel entre la tension d'éclairement et la tension de non-éclairement est trop faible pour que la carte puisse l'analyser correctement. C'est pourquoi, nous allons utiliser un montage intermédiaire qui va nous permettre d'envoyer à la carte Arduino, une différence de potentiel suffisamment élevée entre les deux cas pour pouvoir être analysée.

Ce montage sera un amplificateur comparateur qui va, comme son nom l'indique, comparer la tension d'entrée de la photodiode à une autre tension de référence que nous aurons choisis comme indiqué dans le dernier paragraphe portant sur la photodiode. En comparant la tension à la borne plus de la photodiode avec la tension de référence, l'amplificateur comparateur va déterminer s'il faut envoyer soit 5V à la carte Arduino, soit la masse à 0V. En revanche, contrairement à ce que son nom indique, l'amplificateur comparateur ne va pas amplifier la moindre différence de potentiel.

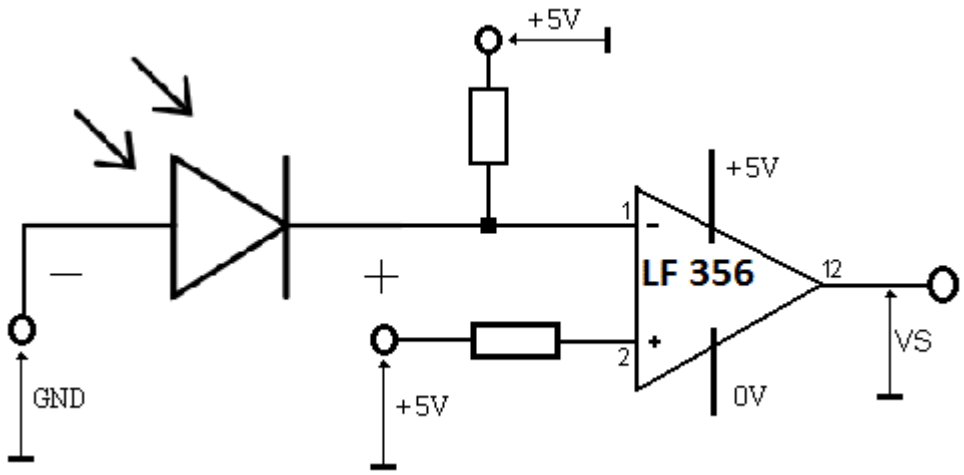
Concrètement, l'amplificateur comparateur est composé d'un amplificateur opérationnel que nous considèrerons comme parfait, composant tri-broches (on dit aussi tripôle), qui va comparer les tensions reçues par ses deux entrés V+ et V-. Pour notre montage, nous utiliserons un LF356 de Texas Instrument comme pour la partie portant sur la mesure de la tension (cf « Mesure de la puissance aux bornes du moteur »). Ci-dessus le schéma de l'amplificateur comparateur. On a donc $VS = \begin{cases} +5V \ si \ V_{ref} > VE \\ 0V \ si \ V_{ref} < VE \end{cases}$



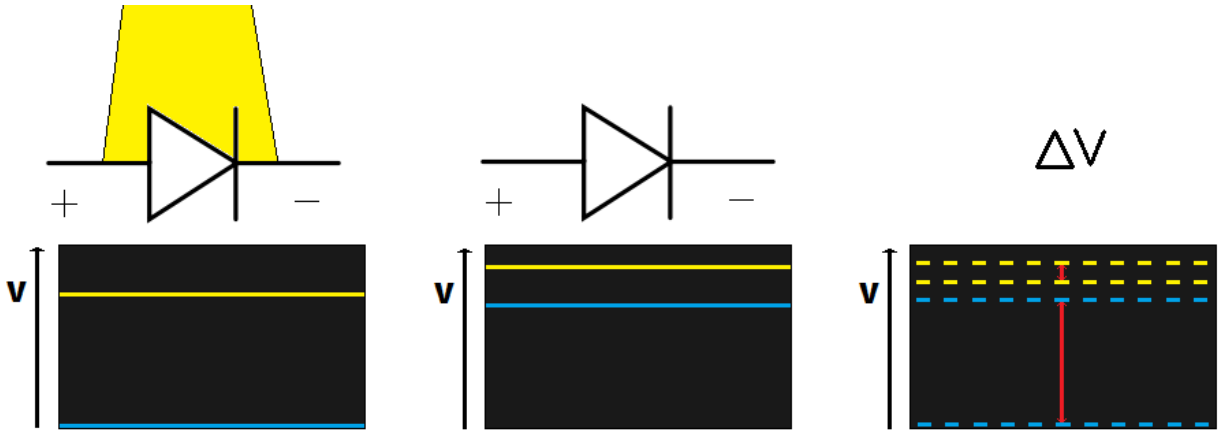
Contrairement à l'autre montage que nous avons pu faire jusqu'alors, celui-ci n'a sa sortie VS ni reliée à la borne V+, ni à la borne V-. Ce montage, contrairement au trigger de Schmitt, n'implique pas de cycle d'hystérésis, c'est-à-dire de tension de basculement différente en fonction de l'état dans lequel on se trouve. Nous n'avons en effet, pas besoin de cycle d'hystérésis.

La photodiode, contrairement à la diode électroluminescente, se branche en inverse du montage.

Finalement, nous obtenons le montage suivant :



Ainsi, l'amplificateur comparateur nous a permis de mieux comparer la différence de tension entre la photodiode éclairée et la photodiode non éclairée. Sur le schéma ci-dessous, on voit clairement que cette différence à la sortie de l'amplificateur comparateur (en bleu clair) est supérieure à celle que l'on aurait obtenue sans celui-ci (en jaune).



Partie V : Programmation de la carte Arduino

a/ Principe de commande du moteur

Le microprocesseur de la carte Arduino doit être programmé pour que grâce à celle-ci, on puisse commander la vitesse d'un petit moteur à courant continu, grâce à l'appui sur un des deux boutons poussoirs de l'écran LCD relié à celle-ci. L'écran LCD, de type de la marque Adafruit, possède plus de quatre boutons. Nous n'en utiliserons ici que trois que nous nommerons bouton « tourne droit », bouton « tourne gauche » et bouton « d'arrêt d'urgence ».

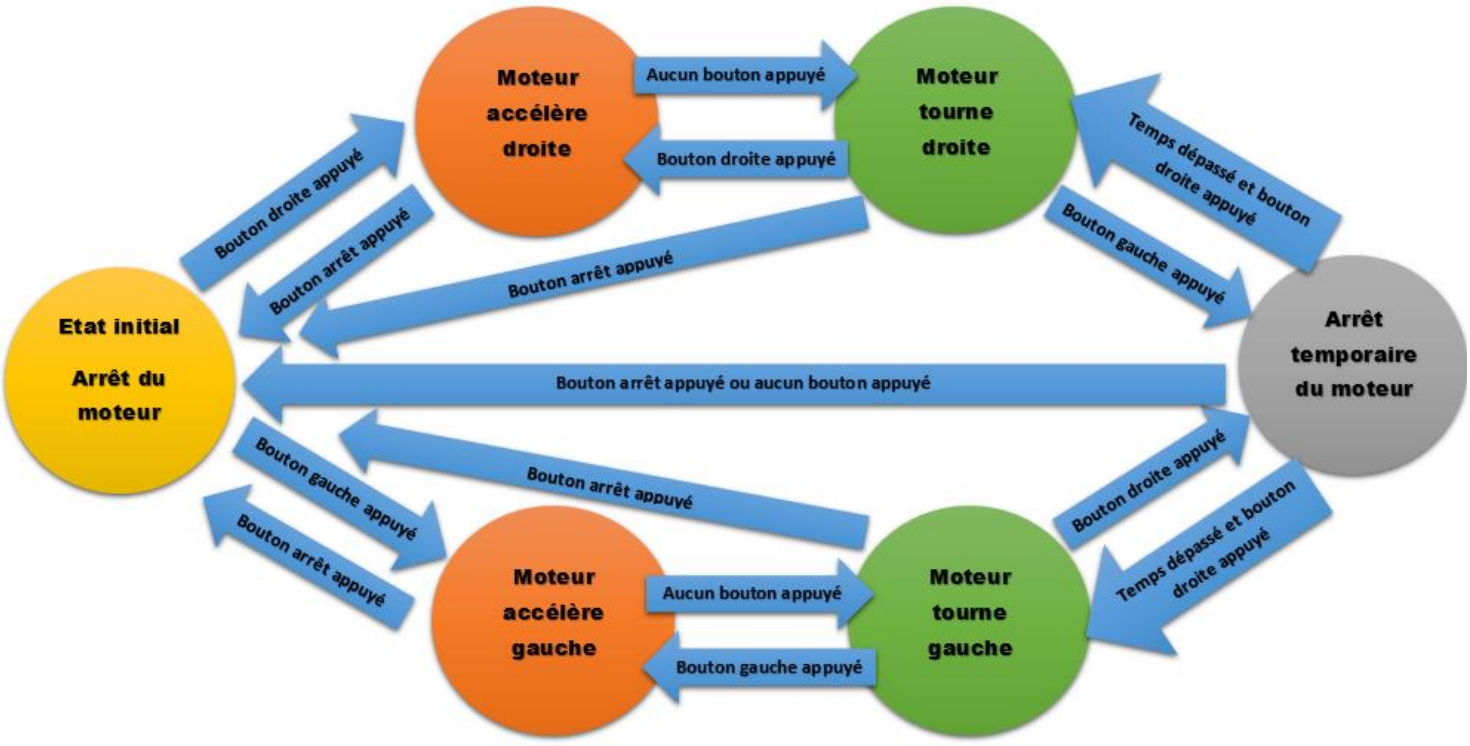
En appuyant sur un des deux boutons « tourne droite » ou « tourne gauche », le moteur tourne dans un sens et la vitesse augmente. En appuyant sur un autre bouton la vitesse du moteur décroît, se stoppe, puis le moteur tourne dans le sens contraire. Sans appuyer sur un bouton poussoir, la vitesse reste stable. Le bouton d'arrêt d'urgence permet de stopper le moteur en cas de problème.

Au regard du cahier des charges que l'on peut lire ci-dessus, nous avons pu définir une sorte de machine d'état de notre circuit. Même si les instructions de notre code Arduino sont séquentielles et par conséquent non simultanées, cette modélisation du problème permet de faciliter le traitement du problème.

Nous avons donc pus faire une synthèse des entrés et des sorties du système :

Entrées	Sorties
Bouton poussoir droite	**Broche 1**
Bouton poussoir gauche	**Broche 2**
Bouton arrêt	

Notre montage peut être modélisé par le graphe d'état ci-dessous :



b/ Graphe d'état du programme

Le programme que nous allons implémenter dans la carte Arduino peut être modélisé par le graphe suivant :

Début

- Début du programme

Déclaration

- Déclaration des constantes
- Déclaration des broches utilisées

Initialisation

- Initialisation de l'écran LCD

Déclaration

- Déclaration des entrées et sorties
- Déclaration de l'interruption qui surveille le pin 2

Lecture

- On lit l'état des boutons : A0

Calcul

- On calcule la tension aux bornes du moteur

Mesure

- On mesure la vitesse du moteur

Affichage

- On affiche la vitesse du moteur sur l'écran LCD

Appuis

- On appuis sur les différents boutons pour changer la direction de rotation du moteur

Affichage

- On affiche la tension à chaque appuis sur le bouton

En cas d'appuis

- On calcule la vitesse du moteur
- Elle est proportionnelle au temps d'appuis sur un bouton

En cas de changement

- La vitesse est remis à valeur minimale lorsque l'on change de sens

En cas d'arrêt

- La vitesse du moteur décroit progressivement juqu'à l'arrêt lorsque l'on appuie sur le bouton UP

Fin

- Le programme ne s'arrête jamais car il est principalement stocké dans la méthode loop qui est une boucle infinie

c/ Notre code :

```
// Voici le code tel que téléversé à la carte Arduino
// Début du code

/*--  Contrôle du sens de rotation d'un moteur CC à l'aide d'un bouton
poussoir :
L'appui sur un bouton poussoir déclenche l'augmentation ou la diminution de
la durée du PWM.
*/


//----  Déclaration des constantes  -----------
54ons tint C1C4 = 3 ;
54ons tint C2C3 = 5 ;
int vitesse = 8 ;
int speed = 0;
int state = 0;

54ons tint voltage = A2; // modif

// Ajout des librairies pour utilise l'écran LCD
#include <LiquidCrystal.h>

// Sélection des pins utilisés par l'écran LCD
LiquidCrystal lcd(8, 9, 4, 5, 6, 7) ;

// Définit les valeurs utilisés par les 5 boutons
int lcd_key     = 0;
int adc_key_in  = 0;
#define RIGHT  0
#define UP     1
#define DOWN   2
#define LEFT   3
#define SELECT 4
#define NONE   5


// Lecture des boutons (tous regroupés sur le Pin Analog A0)
int read_LCD_buttons()
{
adc_key_in = analogRead(0) ;      // Lit la valeur depuis le capteur sur A0
// les boutons sont centés sur vces valeurs : 0, 144, 329, 504, 741
// On ajoute environ 50 à ces valeurs et on vérifie si on est proche
if (adc_key_in > 1000) return NONE ; // On définit cela en première option
pour des raisons de vitesse car c'est l'option la plus probable
// For V1.1 us this threshold
if (adc_key_in < 50)   return RIGHT;
if (adc_key_in < 195)  return UP;
if (adc_key_in < 380)  return DOWN;
if (adc_key_in < 555)  return LEFT;
if (adc_key_in < 790)  return SELECT;
}
```

```
//---------------- Configuration des broches en entrées sorties -----------
-----
void setup ()
{
  Serial.begin(9600);//Port série
  pinMode(C1C4, OUTPUT);
  pinMode(C2C3, OUTPUT);

  pinMode(voltage, INPUT); //
//Lance la librairie (16 caractères x 2 lignes)
   lcd.begin( 16, 2 ) ;  //
   //lcd.setCursor(x,y) ; permet de positionner le curseur sur le caractère
x de la ligue y
   lcd.setCursor( 0, 0 ) ; //En haut à gauche (première colone, première
ligne)
   lcd.print(« vitesse :  « ) ;//Afficher le message vitesse en haut à
gauche de l'écran
   lcd.setCursor(0,1) ;//  En bas en début de deuxième ligne (première
colonne, deuxième ligne)
   lcd.print(« tension : ») ;
   afficheVitesse(1) ;

}


void loop ()
{
lcd_key = read_LCD_buttons();  // lire le bouton appuyé
Serial.println(analogRead(voltage*(5/2*1023)));
switch (lcd_key)      // Selon le bouton choisit on effectue une action
{
    case RIGHT:
   {
     if (state == 1){
       //Serial.println("bouton droit");
       augmenteVitesse() ;
      // Serial.println(« on augmente la vitesse »+vitesse) ;
       }
    else {
       diminueVitesse() ;
       vitesse = 8 ;

      // Serial.println(« on augmente la vitesse »+vitesse) ;
       }


       droite();
      break;
   }

   case LEFT:
   {
```

```
     if (state == 2){
       augmenteVitesse() ;
     }
      else{
        diminueVitesse() ;
        vitesse = 8 ;
        }

       gauche() ;
       break;
    }

    case UP:
     {

        // Serial.println("moteur bien arrêté");
         //if (vitesse >= 6){
           //diminueVitesse();
           //}
           stopMoteur();
            state = 0;
         break;
      }

}
}

void droite ()
{
  //Si C1 et C4 sont à l'état HAUT alors on met au niveau BAS C2 et C3

          state = 1 ;
          //Serial.println(vitesse) ;
          speed = map(vitesse,0,9,0,255) ;
          //Serial.print(« vitesse apres la map = « ) ;
          //Serial.println(vitesse);
          //Serial.print("speed = ");
          //Serial.println(speed);
          digitalWrite(C2C3, HIGH);
          //Serial.print(« valeur de C2C3 = « ) ;
          //Serial.println(C2C3);
          analogWrite(C1C4, speed);

            delay(300) ;


}

void gauche ()
{//Si C1 et C4 sont à l'état HAUT alors on met au niveau BAS C2 et C3

          state = 2;
          Serial.println(vitesse);
          speed = map(vitesse,0,9,0,255);
```

```
          digitalWrite(C1C4, HIGH);
          analogWrite(C2C3, speed) ;

            delay(300) ;

}

void stopMoteur ()
{
  while(vitesse != 9){
        vitesse ++;
              if (state==1){
                    speed = map(vitesse,0,9,0,255);
                    analogWrite(C1C4, speed) ;//On arrête le moteur si on
est en tourneDroit
                    digitalWrite(C2C3, HIGH) ;//On arrête le moteur si on
est en tourneGauche
                 }
              else {
                    speed = map(vitesse,0,9,0,255) ;
                    digitalWrite(C1C4, HIGH) ;//On arrête le moteur si on
est en tourneDroit
                    analogWrite(C2C3, speed) ;//On arrête le moteur si on
est en tourneGauche
              }
      delay(300) ;
    }
}

void augmenteVitesse ()
{
  if (vitesse != 0) //lE MOTEUR TOURNE AU MAXIMUM A 0v
      vitesse -- ;

}

void diminueVitesse(){
while(vitesse !=9){
  vitesse ++ ;
  delay(100) ;
  }
}


void afficheVitesse(int val)
{
  lcd.setCursor(9,0) ;    //Curseur sur ligne 1 (0) et position 10 (9)
  lcd.print(val);
  lcd.print(“ tr/s”);
}

//Conversion du CAN : q=Vref/2^N => val = (val*5)/1023
void afficheTension(int val)
{
```

```
  lcd.setCursor(9,1) ;
  int tension = val *(5/(1023*10)) ;  //Curseur sur ligne 2(1) et position
10(9)
  lcd.print(tension) ;
  lcd.print(«  mv ») ;
}

// Fin du code
```

Conclusion du projet

Finalement, nous avons pu grâce à la carte Arduino et au pont en H, mettre ne place un système permettant la commande de la vitesse du moteur, ainsi que sa mesure grâce à l'utilisation d'une photodiode et d'une DEL. Le circuit électrique composé des amplificateurs différentiels et sommateurs, ainsi que du circuit RC nous ont permis la mesure de tension aux bornes de la résistance en série avec le moteur et donc l'intensité traversant le moteur.

Grâce à notre montage, nous avons pu mettre en place un système complet de contrôle du moteur.

Les auteurs

Nomenclature des composants

Nom du matériel	Partie d'utilisation	Quantitée
Amplificateur LF356	Mesure électronique	3
Cable USB-Arduino	Commande moteur et analyse vitesse moteur	1
Carte Arduino Uno	Commande moteur et analyse vitesse moteur	1
Condensateur 47 µF polarisé	Analyse puissance	2
Condensateur x F	Analyse puissance	2
Diode éléctroluminescente	Analyse vitesse moteur	1
Diode photosensible	Analyse vitesse moteur	1
Disque en carton	Analyse vitesse moteur	1
Ecran LCD Adafruit	Analyse vitesse moteur	1
Fils en cuivre	Pont en H, vitesse moteur, analyse puissance	Beaucoup
Générateur de tension DC	Pont en H	1
Maquette	Pont en H, vitesse moteur, analyse puissance	4
Moteur CC3	Pont en H	1
Ordinateur	Carte Arduino : Commande moteur	1
Oscilloscope Textronik	Pont en H, vitesse moteur, analyse puissance	1
Résistance 47 kΩ	Pont en H, vitesse moteur, analyse puissance	2
Résistance 47 MΩ	Pont en H, vitesse moteur, analyse puissance	2
Résistance 56 kΩ	Analyse vitesse moteur	1
Résistance 100 kΩ	Analyse vitesse moteur	1
Résistance de puissance 2,2 Ω	Analyse puissance	1
Résistance 180 Ω	Analyse vitesse moteur	4
Résistance 1 kΩ	Analyse puissance	1
Résistance 6,8 kΩ	Analyse puissance	1
Résistance 120 Ω	Analyse puissance	1

Sources utilisées:

- http://docs-europe.electrocomponents.com/webdocs/0e8b/0900766b80e8ba21.pdf
- http://www.arduino.cc/
- http://perso.esiee.fr/~poulichp/PR201/Cmde_moteur/cmde_moteur.html
- Toutes les datasheets évoquées dans le rapport

Remerciements

Nous tenons particulièrement à remercier :

- M.Poulichet pour l'organisation du projet
- Ainsi que tous les autres intervenants pour leur aide et leur soutient