Nom : ……. Prénom : …….

TP TC

Variation de vitesse des moteurs à courant continu (Fonction hacheur)

Support : Robot moway alecop

Pré requis (l'élève doit savoir):

- Connaître le fonctionnement des éléments électriques de base (diode, transistor).
- Comprendre un schéma électrique.

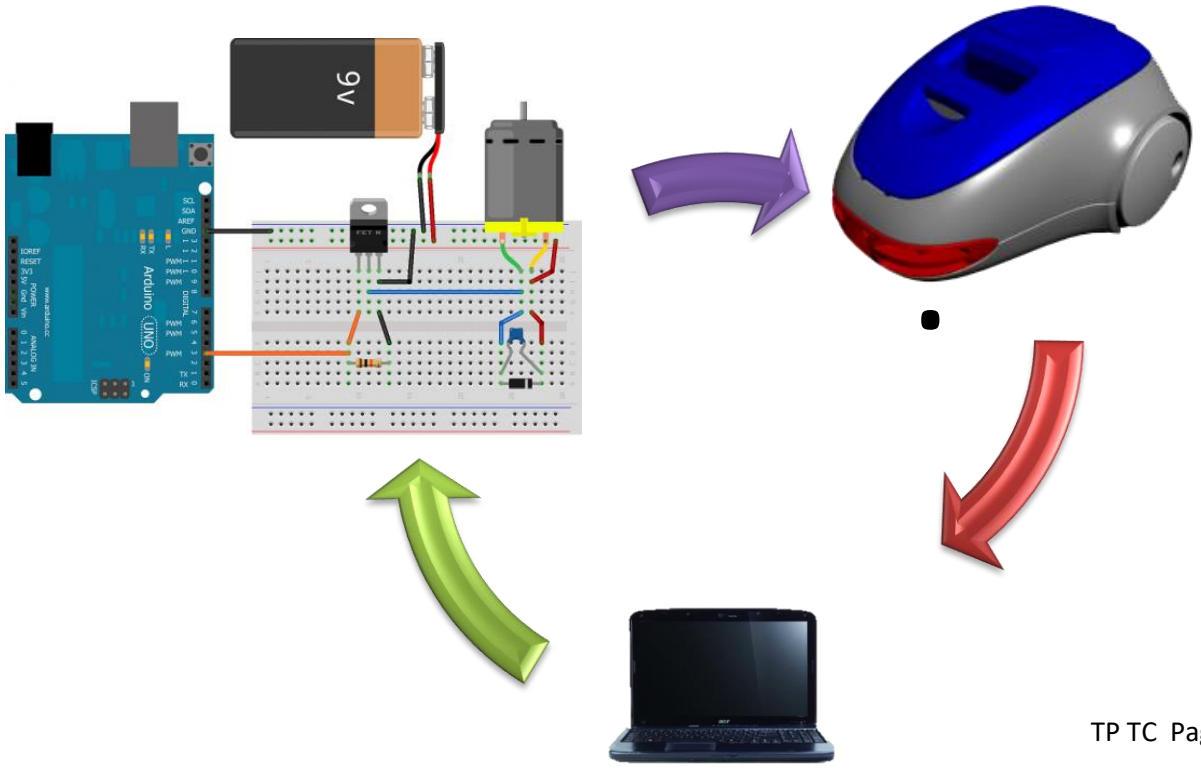
Objectif terminale :

L'élève doit être capable d'après la documentation technique des schémas électriques, d'expliquer le fonctionnement d'un hacheur.

Compétence

Matériel :

- Ordinateur
- Logiciel arduino
- Logiciel Flowcode
- Logiciel Proteus
- Carte arduino méga
- Robot moway



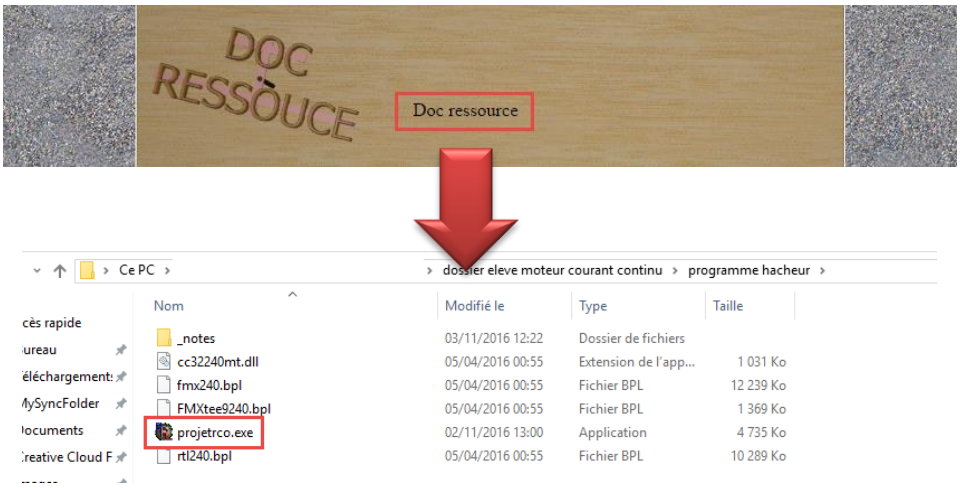
Nom : ……. Prénom : …….

1. Travail demandé

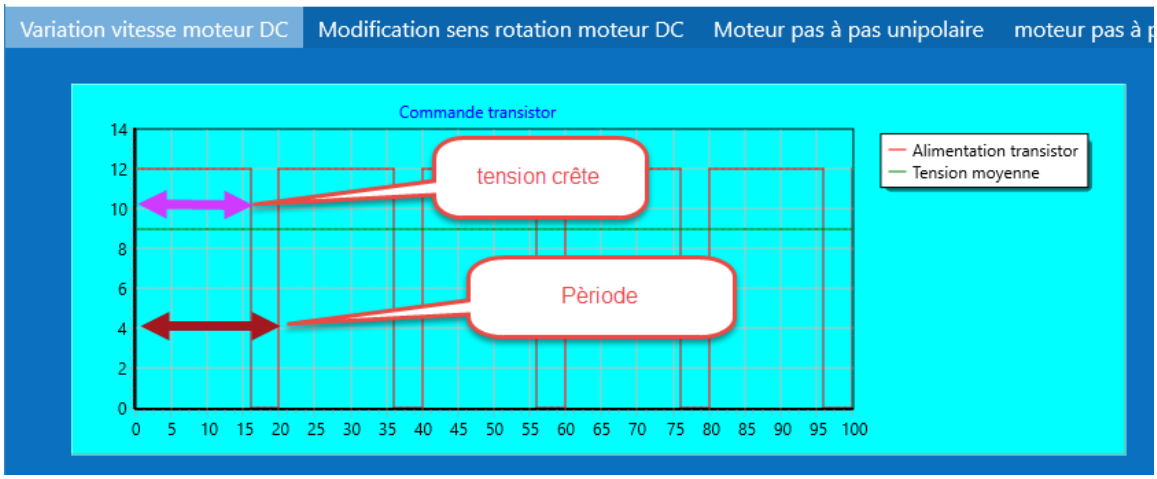
1.1 Fonction hacheur

- Récupérer le dossier « dossier eleve courant continu.zip » dans doc ressource et exécuter le programme projetrco.exe

http://www.sti2dsinhyrome.fr/doc%20cours/docchaineenergie/chaineenergiepartie1/docchaineenergiepartie1.html



- Cliquer sur « variation vitesse moteur DC » et remplir le tableau ci-dessous en fonction du pourcentage vitesse max



Pourcentage vitesse max (%)	Valeur tension moyenne	Durée tension de crête (C)	Période (T)	Rapport (C/T)
0				
20				
40				
60				
80				
100				

- Que remarquez-vous entre le rapport C/T et le pourcentage vitesse max

- Déterminer une relation entre la vitesse du moteur, sa vitesse max et le rapport C/T

Vm =

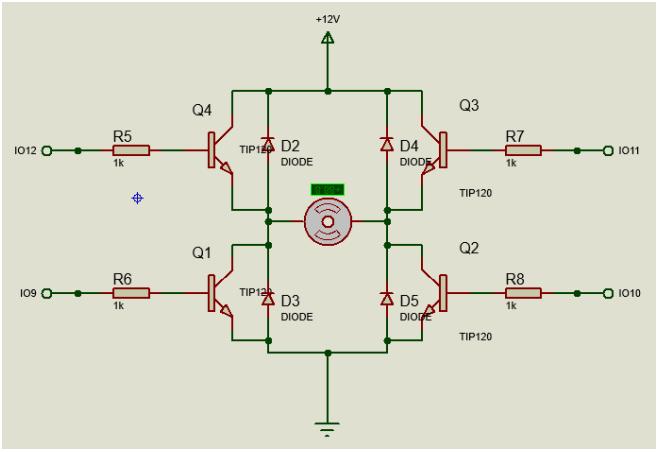
1.2. Pont en H

- Cliquer sur « Modification sens de rotation DC » et déterminer la valeur binaire des transistors pour chaque sens de rotation

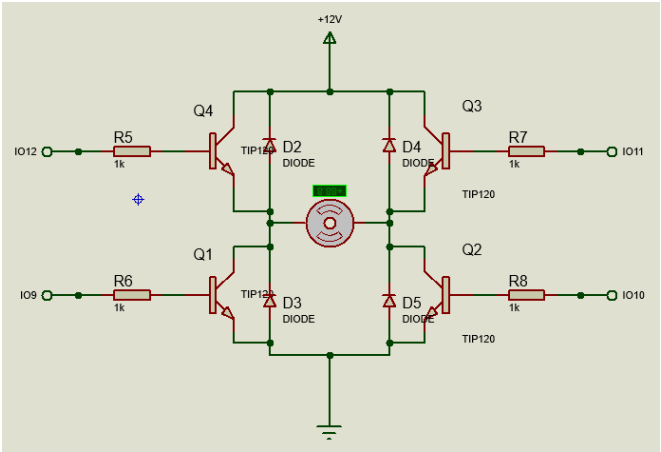
Q1 = 1, lorsqu'il est alimenté

Sens de rotation	**Q1**	**Q2**	**Q3**	**Q4**
Positif				
Négatif				

- Tracer la circulation du courant pour chaque sens et entourer les transistors passant
  - Sens positif



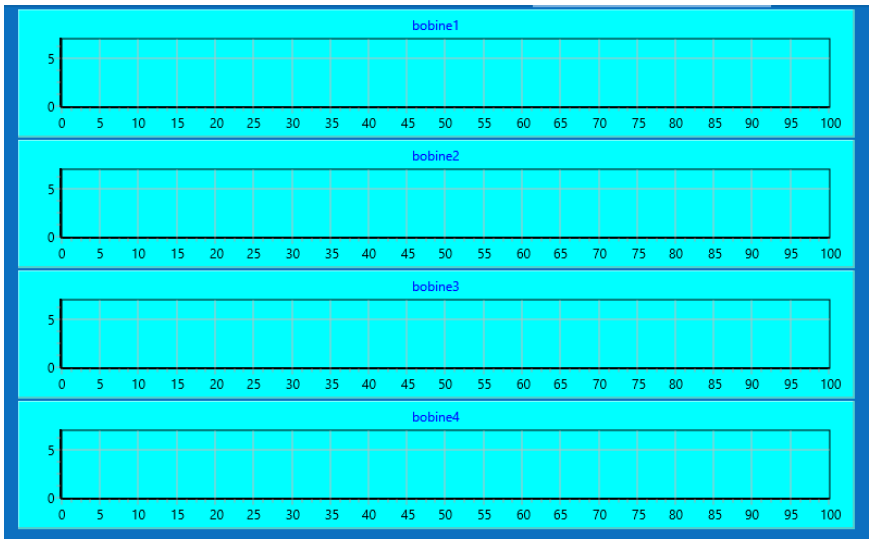
  - Sens négatif



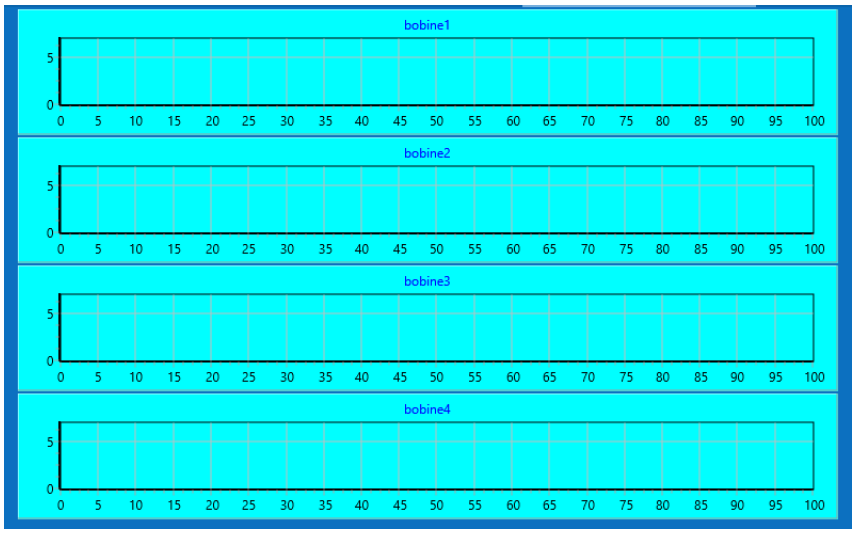
Nom : ……. Prénom : …….

1.3. Moteur pas à pas unipolaire

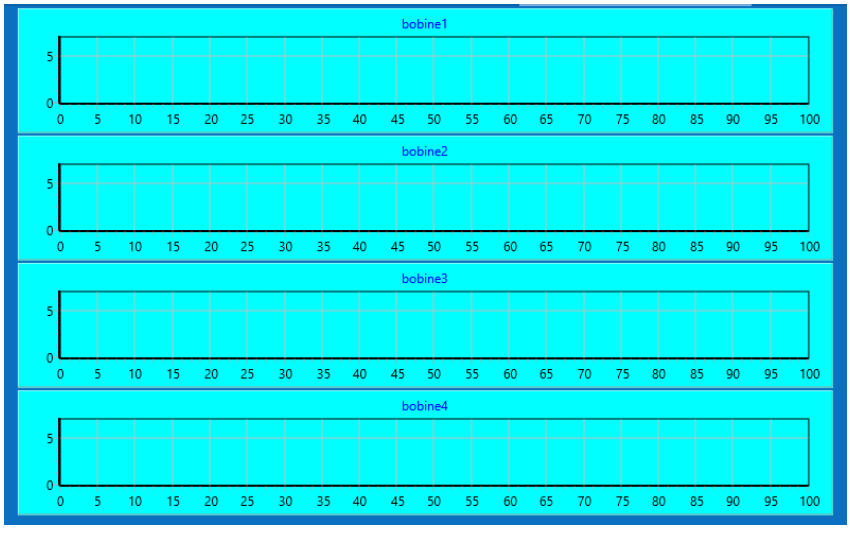
- Cliquer sur « moteur pas à pas unipolaire » et pour chaque vitesse tracer la courbe des tensions
  - Vitesse 10%



  - Vitesse 30%



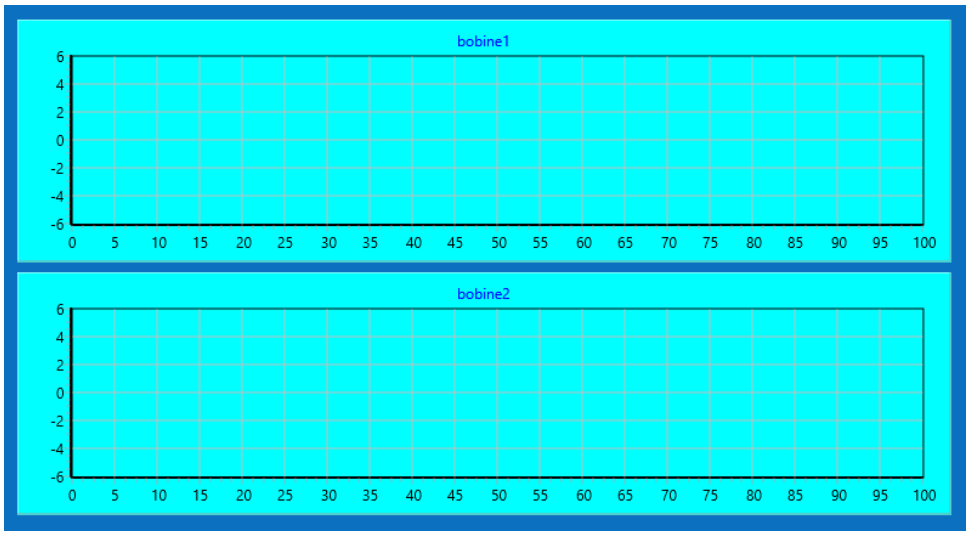
  - Vitesse 90%



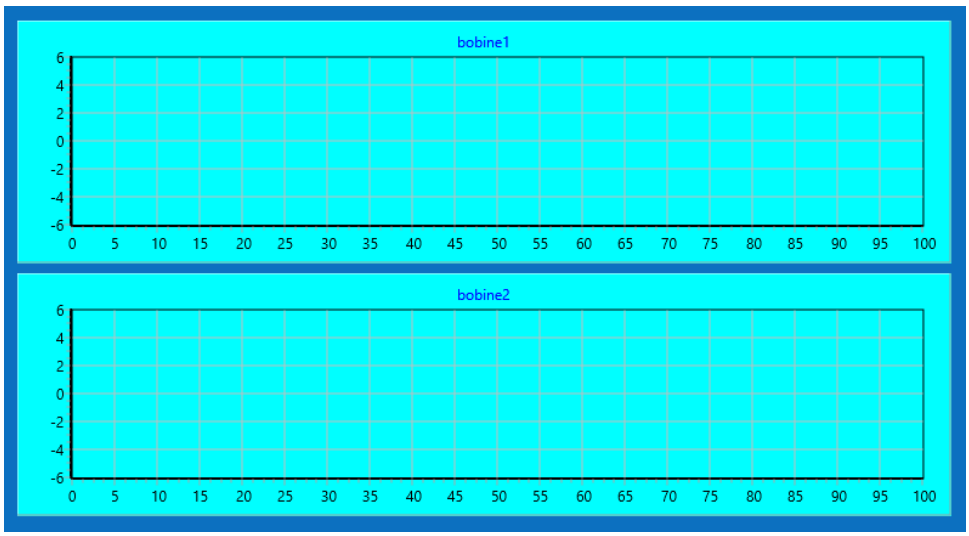
Nom : ……. Prénom : …….

Moteur pas à pas bipolaire

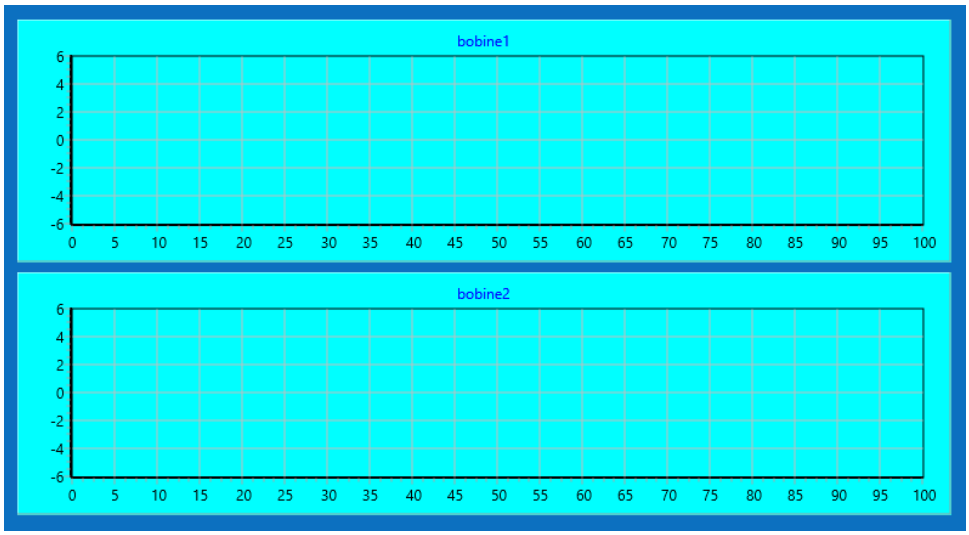
- Cliquer sur « moteur pas à pas bipolaire » et pour chaque vitesse tracer la courbe des tensions
  - Vitesse 10%



  - Vitesse 30%



  - Vitesse 90%

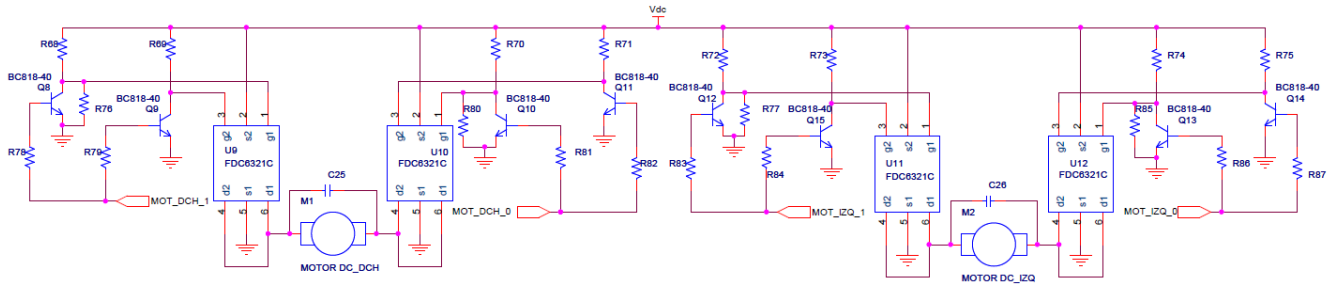


Nom : ……. Prénom : …….

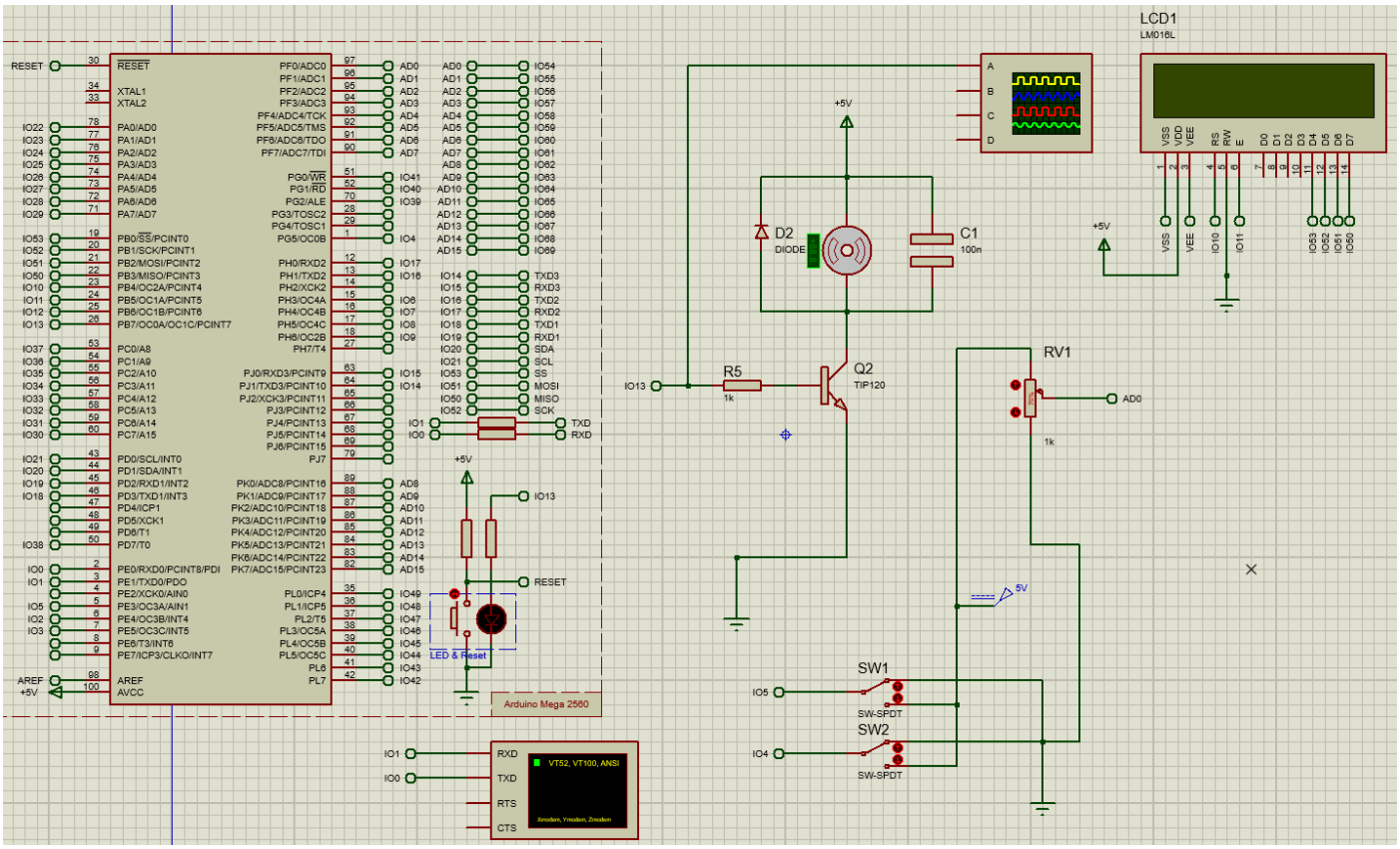
- Que remarquez-vous au niveau des courbes en fonction de la vitesse (qu'est ce qui change et ne change pas)

1.4 Etude du circuit d'alimentation des moteurs du robot moway

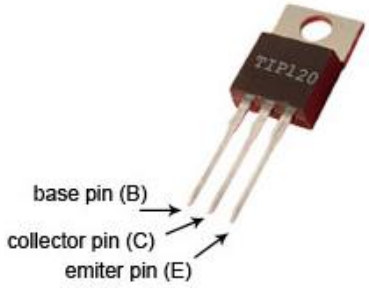
D'après le schéma ci-dessous indiquer le type d'alimentation des moteurs et l'entourer en rouge (expliquer votre réponse)

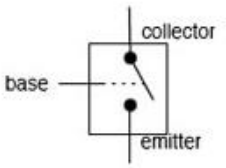


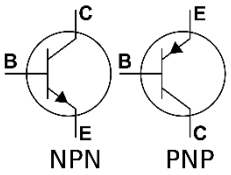
1.5. Réaliser le montage suivant sur Proteus



Nom : ……. Prénom : …….







Attention

Transistors NPN

- Pour autoriser le passage du courant du collecteur à l'émetteur, il faut appliquer une tension relativement positive à la base.
- Sur le symbole du schéma, la flèche pointe de la base vers l'émetteur et montre la direction du courant positif.
- La tension appliquée à la base doit être supérieure d'au moins 0,6 V à celle appliquée à l'émetteur.
- Le collecteur doit être plus positif que l'émetteur.

Transistors PNP

- Pour autoriser le passage du courant de l'émetteur au collecteur, il faut appliquer une tension relativement négative à la base.
- Sur le symbole du schéma, la flèche pointe de l'émetteur vers la base et montre la direction du courant positif.
- La tension appliquée à la base doit être inférieure d'au moins 0,6 V à celle appliquée à l'émetteur.
- L'émetteur doit être plus positif que le collecteur.

Réaliser les programmes sur Flowcode ou logiciel Arduino, puis tester sur Proteus

On utilisera deux boutons, Bp1 pleine vitesse, Bp2 vitesse moyenne (50%). On affichera l'information sur l'afficheur.

- Pour cela, on va utiliser la sortie 13 PMW (Digital 13 : carte Mega) et la fonction analogWrite(brochePWM, 127);

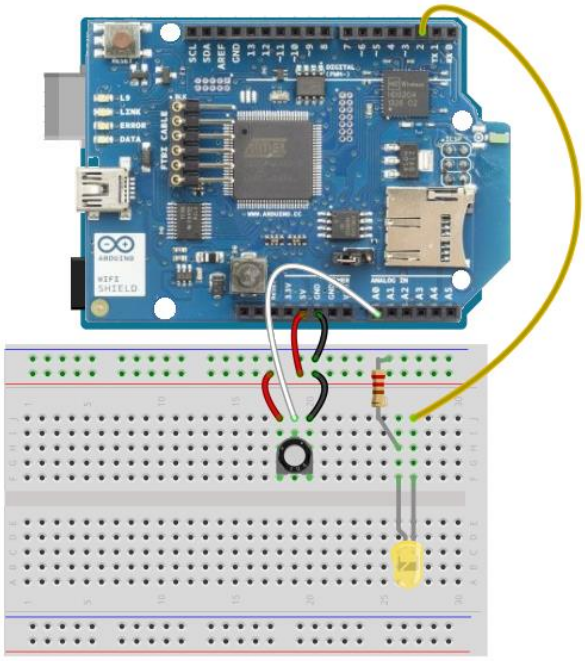
```
const int brochePWM = 13;

void setup()
{
  //configuration en sortie de la broche 13
  pinMode(brochePWM, OUTPUT);
}
```

Le rapport cyclique est défini par un nombre allant de 0 à 255. Cela signifie qu'à 0, le signal de sortie sera nul et à 255, le signal de sortie sera à l'état HAUT. Toutes les valeurs comprises entre ces deux extrêmes donneront un rapport cyclique plus ou moins grand. Dans notre cas, le moteur tourne plus ou moins vite selon si le rapport cyclique est grand ou petit. Pour savoir quel rapport cyclique correspond avec quelle valeur, il faut faire une règle de trois.

- Refaire l'exercice précédent
- On veut commander la vitesse d'un moteur en utilisant un potentiomètre

Exemple :

Nom : ……. Prénom : …….

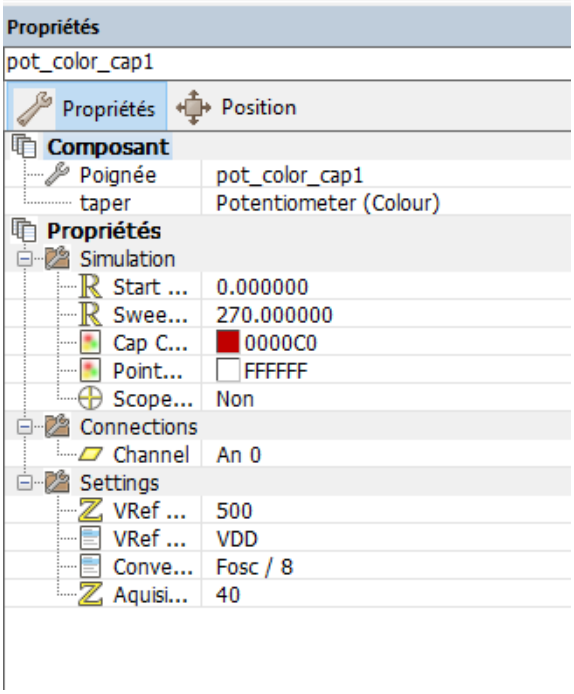


- Réaliser le programme, à fin que la vitesse du moteur soit en corrélation avec le potentiomètre.
- On affichera la valeur du potentiomètre sur l'afficheur LCD

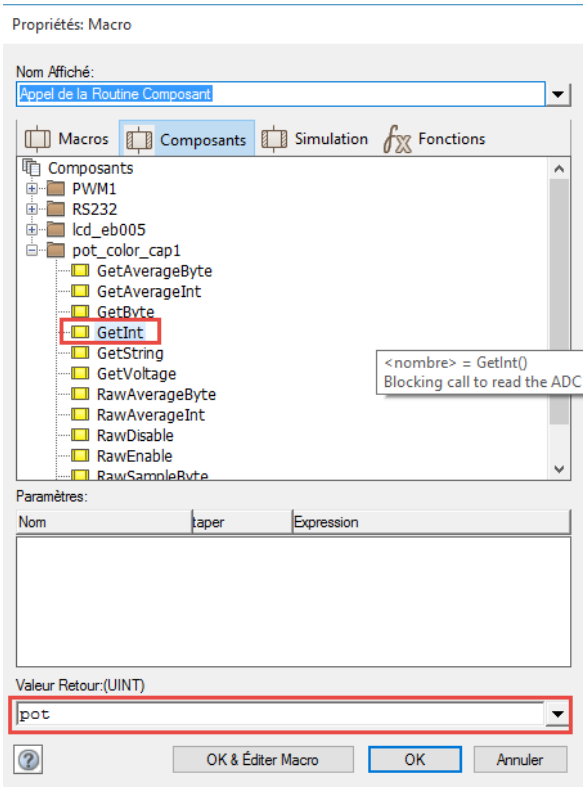
Remarque :

La lecture analogique nous renvoie une valeur entre 0 et 1023 (soit 1024 valeur possibles). Or la fonction analogWrite ne peut aller qu'entre 0 et 255 (total de 256 valeurs). On doit donc diviser par 4 la valeur lue sur le potentiomètre pour rester dans le bon intervalle, car : 4×256=1024.

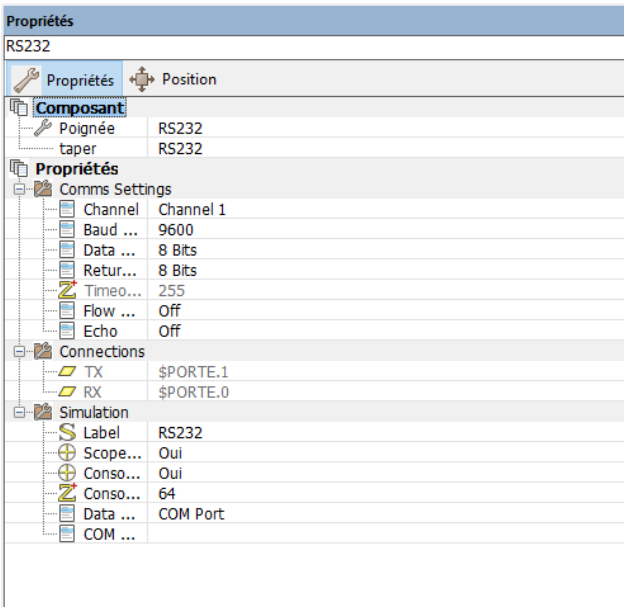
Pour flowcode, pour récupérer la valeur du potentiomètre, on utilisera la routine getInt



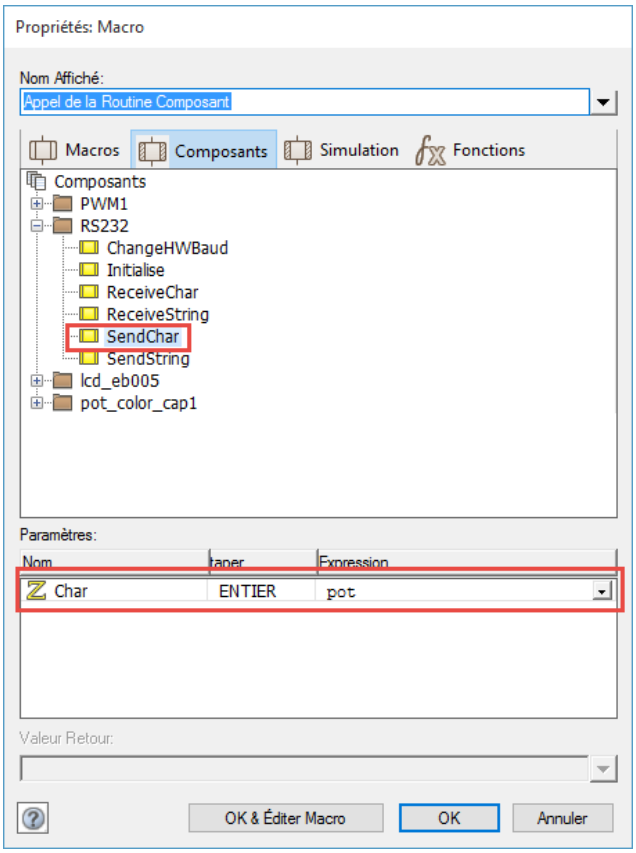
Nom : ……. Prénom : …….



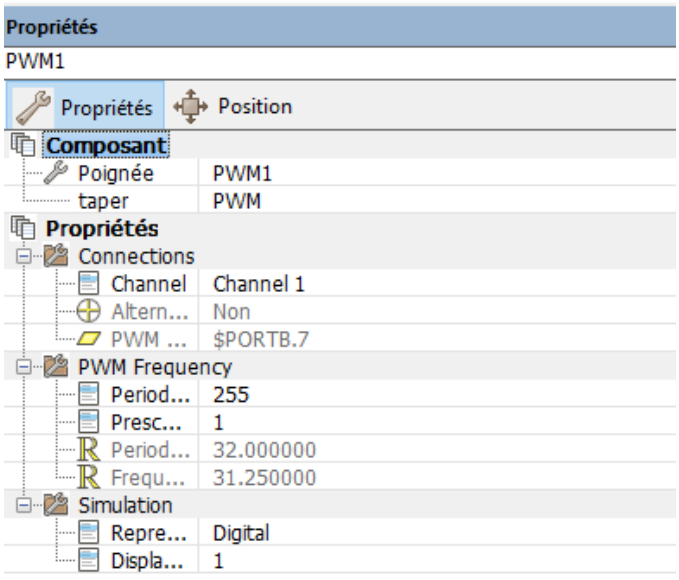
Pour envoyer la valeur sur le port série, on utilisera la routine SendChar



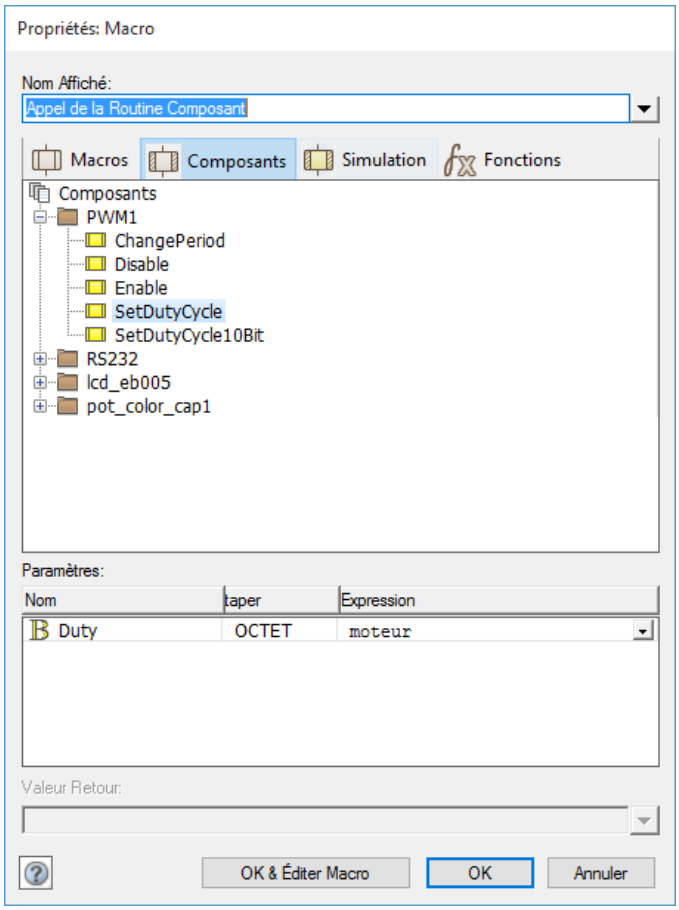
Nom : ……. Prénom : …….



Pour commander le moteur on utilisera la routine SetDutyCycle du composant PMW



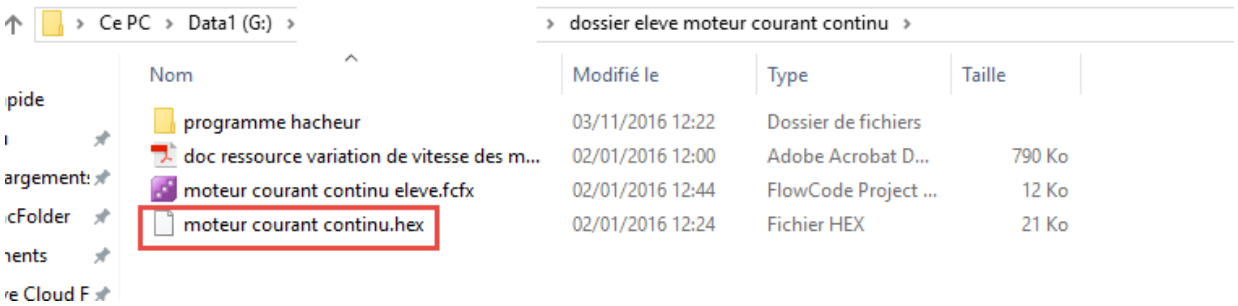
Nom : ……. Prénom : …….



Cinq routines composant sont préécrites et permettent donc d'utiliser très facilement ce composant :

- Enable (nldx(OCTET)) : Valide le canal PWM spécifié et lance son fonctionnement
- Disable (nldx(OCTET)) : Désactive le canal PWM spécifié.
- SetDutyCycle (nldx(OCTET), nDuty(OCTET)) : Fixe la valeur du rapport cyclique (nDuty) du canal PWM spécifié (nldx). Avec nDuty = 200 => le rapport cyclique = 200/250 = 80%
- SetDutyCycle10bit (nldx(OCTET), nDuty(ENTIER)) : Fixe la valeur sur 10 bits du rapport cyclique (nDuty) du canal PWM spécifié (nldx). Le rapport cyclique varie de 0% (nDuty = 0) à 100% (nDuty = 1000)
- ChangePeriod (nPeriodVal(OCTET), nPrescalerVal(ENTIER)) : Permet de modifier en cours d'exécution les valeurs de Period register et de Clock source (à utiliser avec précaution).

Vous pouvez tester le programme sur Proteus en utilisant le fichier .hex dans le document ressource



Nom : ……. Prénom : …….

- Refaire le montage sur tinkercad

https://www.tinkercad.com/#/