OZANAM

Compte rendu :

Potentiomètre et servo moteur

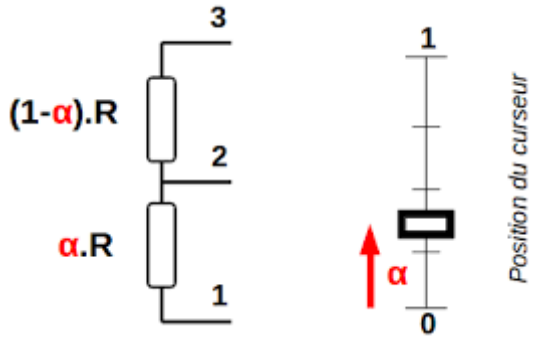
Benoit BOREK
BTS CIEL 2B
Campus Ozanam

Table des matières

Étude du potentiomètre :

Tout d'abord il fallait revoir comment fonctionnait un potentiomètre.

Dans le schéma technique ci-dessous, les bornes 3 et 1 sont l'alimentation (le sens n'est pas définie) et la borne 2 est le curseur, sa position est définie par alpha allant de 0 à 1.



En appliquant la formule où R est la résistance du potentiomètre (par exemple : 10k Ohm), on peut faire varier le courant allant vers la borne 2.

Je vais donc exploiter cette variation de courant pour que mon Arduino puisse réagir en conséquence.

A l'aide d'un multimètre j'ai relevé les différentes valeurs de la résistance approximative se trouvant dans le potentiomètre.

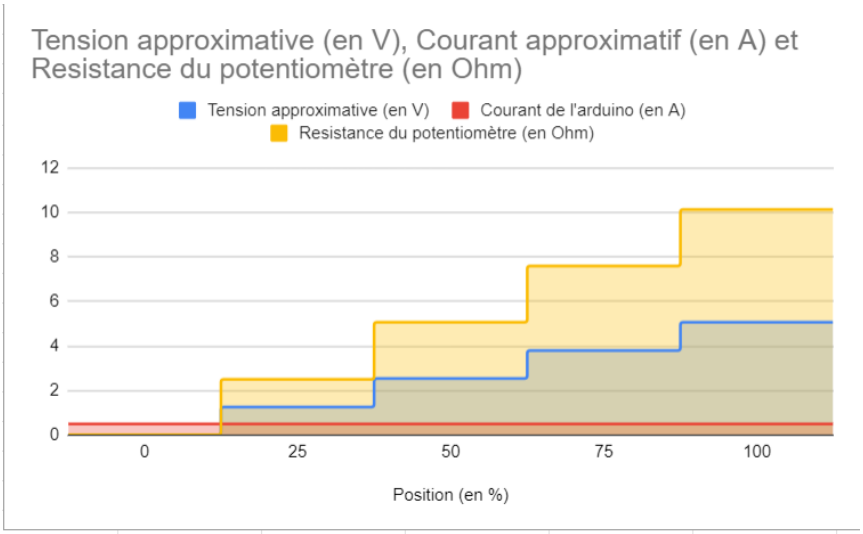


Figure 1 Valeurs approximative du potentiomètre

Étude du servo moteur :

Une fois le potentiomètre analysé j'ai étudier le fonctionnement d'un servo moteur, le modèle que j'ai utilisé est le « MG 995 ». Ce servo moteur à une plage d'angle de 0° à 180°, il ne peut pas aller au-delà de 180° ou en dessous de 0° car des sécurités sont présent mais lorsqu'il est hors tension il peut être tourner à 360°.

Le « MG 995 » à une plage de tension de +4.7V et +7.2V donc il sera alimenté via la borne +5V de l'Arduino. Il a un seuil de tension de 700mA, un Arduino Méga en délivre 500mA ce qui est donc correct.

Électriquement un servo moteur utilise une longueur d'impulsion entre +4.8V et +6V entre 1ms et 2ms sur une période de 20ms. Cette impulsion va définir l'angle dans lequel le servo moteur va se positionner, ce qui veut dire qu'il n'utilise pas de données numériques mais analogique, son MLI ira vers une borne « PWM » de l'Arduino Méga.

Câblage :

Ensuite est venu le moment de câbler le potentiomètre à l'Arduino Méga et le servo moteur à l'Arduino Méga également, il ne faut pas oublier à respecter le câblage en fonction des couleurs des fils demandé. Si jamais le code couleur du tableau ci-dessous n'est pas respecter le projet ne fonctionnera pas.

Broche du potentiomètre	Pin Arduino Méga	Couleur du fil
P1	+5V	ROUGE
P2 (curseur)	A0	VERT
P3	GND	NOIR
Broche du Servo-moteur	Pin Arduino Méga	Couleur du fil
GND	GND	Marron
VCC	+5V	Rouge
MLI	D3	Orange

Programmation :

Une fois que j'ai câbler le potentiomètre et le servo moteur à l'Arduino il ne rester plus qu'à rédiger le programme qui sera flasher dans l'Arduino pour qu'il puisse prendre les données analogiques du potentiomètre, les traités et envoyer une impulsion au servo moteur qui correspondra à la position du curseur.

Pour réaliser le code j'ai utilisé l'IDE Visual Studio Code Version 1.95.3 et l'extension « Platform IO » pour que le code puisse être flashé dans l'Arduino.

J'ai utiliser également la bibliothèque « servo.h » qui à des fonctions prédéfinis pour pouvoir contrôler le servo moteur plus facilement et alléger mon code.

```
Servo myServo;

int const potPin = A0;

int potVal;

int angle;


int const Angle_Mini = 0;
int const Angle_Max = 179;
```

Figure 2 Déclarations des bornes et variables.

Tout d'abord je renomme la bibliothèque « Servo » par « myServo » qui rajoute du confort pour le code car si je veux utiliser une fonction de la bibliothèque j'aurais juste à faire « myServo.nomDeLaFonction() ».

Je déclare le Pin du potentiomètre en A0. Je déclare un entier « potVal » qui aura la valeur analogique allant de 0 à 1023. L'entier « angle » servira à enregistrer l'angle voulu pour le potentiomètre.

Et les deux entiers « Angle_Mini » et « Angle_Max » sont les valeurs maximales dans lequel le servo moteur pourra tourner. A noter que j'ai mis 179° pour ne pas solliciter les sécurités du servo moteur.

Pour finir dans la void loop() j'ai mis des « Serial.println » pour afficher des informations utiles dans le moniteur série (il ne faut pas oublier d'initialiser la communication Arduino ←→PC avec le « Serial.begin() »).

J'utilise un « analogRead » pour lire la valeur analogique du curseur du potentiomètre, la valeur est enregistrée dans l'entier « potVal » et je l'affiche dans le moniteur série.

```
void loop() {
  Serial.println("");
  Serial.println("Nouvelle commande d'angle.");
  Serial.println("");

  potVal = analogRead(potPin);
  Serial.print("Valeur du potentiomètre : ");
  Serial.print(potVal);

  angle = map(potVal, 0, 1023, Angle_Mini, Angle_Max);
  Serial.print(" , angle : ");
  Serial.println(angle);
  myServo.write(angle);
  delay(2500);
```

Figure 3 Le code dans la void loop().

J'utilise la fonction « map() » qui permet de convertir la valeur analogique contenue dans « potVal » en un angle pour le servo moteur, on peut voir que les entiers « Angle_Mini » et « Angle_Max » sont utilisé pour la conversion et donc si je modifie l'angle max à 90° et bien la valeur dans « angle » n'ira pas au-dessus de 90.

J'affiche la valeur convertie et j'utilise la fonction « myServo.write(angle) » qui permet donc à dire au servo moteur de se positionner dans l'angle enregistrer dans l'entier « angle ».