**TP PROJET MASTER I (2022)**

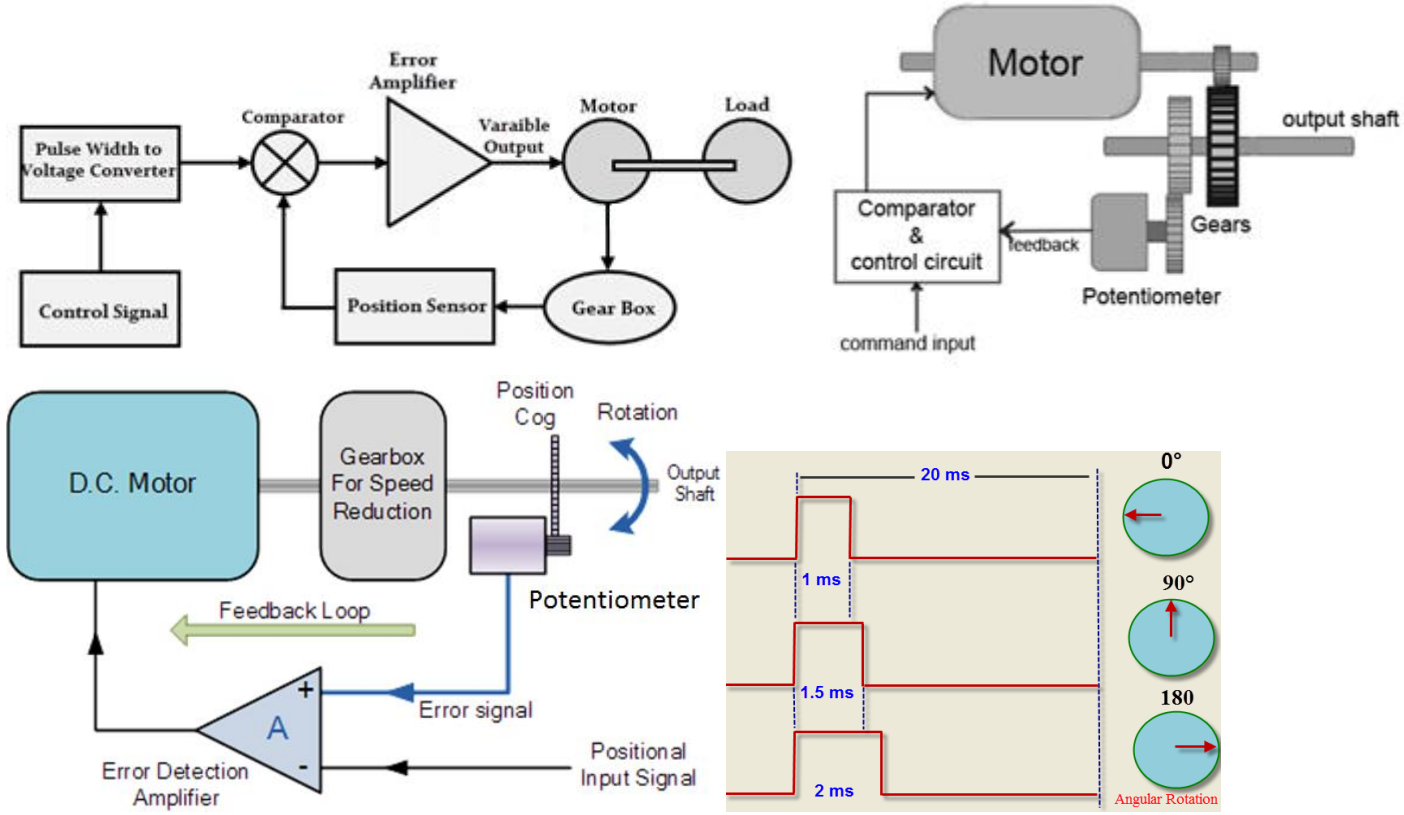
**Université de Jijel /Département Electronique**

**OBJECTIFS VISES :**

**1-Comprendre le principe de fonctionnement d'un servomoteur.**
**2-Savoir comment l'identifier et l' utiliser dans une application réelle (ex :Robotique,Radar…etc ).**
**3-Etre capable de le commander en utilisant une carte programmable telle que Arduino,PIC …etc.**

**Un moteur qui sait ou il va : le servomoteur**

## Fonctionnement




Commande en largeur d'impulsion : la largeur des impulsions, comprise en général entre 1 et 2 millisecondes, commande la position du servomoteur.

Les servomoteurs sont commandés par l'intermédiaire d'un câble électrique à trois fils qui permet d'alimenter le moteur et de lui transmettre des consignes de position sous forme d'un signal *codé en largeur d'impulsion* plus communément appelé PWM. Cela signifie que c'est la durée des impulsions qui détermine l'angle absolu de l'axe de sortie et donc la position du bras de commande du servomoteur. Le signal est répété périodiquement, en général toutes les 20 ms, ce qui permet à l'électronique de contrôler et de corriger continuellement la position angulaire de l'axe de sortie, cette dernière étant mesurée par le potentiomètre.

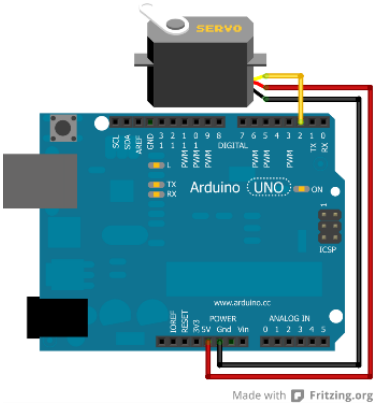
Lorsque le moteur tourne, l'axe du servomoteur change de position, ce qui modifie la résistance du potentiomètre. Le rôle de l'électronique est de commander le moteur pour que la position de l'axe de sortie soit conforme à la consigne reçue : c'est un asservissement.

### Principe

Les servomoteurs sont des actionneurs rotatifs asservis en position. Cela veut simplement dire que pour commander un servo, il suffit de lui donner l'angle qu'il doit atteindre et qu'il se débrouille pour se mettre en position et y rester. Contrairement au moteur à courant continu, on sait donc toujours où est notre moteur sans avoir à ajouter de l'électronique.
Ils sont très utilisés en modélisme et on en trouve vraiment à tous les prix. Généralement en plus de l'asservissement en position, les servos ont un fort couple. Si on regarde à l'intérieur, le couple est apporté par un moto-réducteur au dessus d'un simple moteur à courant continu. L'asservissement en position, est géré grâce à un potentiomètre sur l'axe moteur.

### Montage

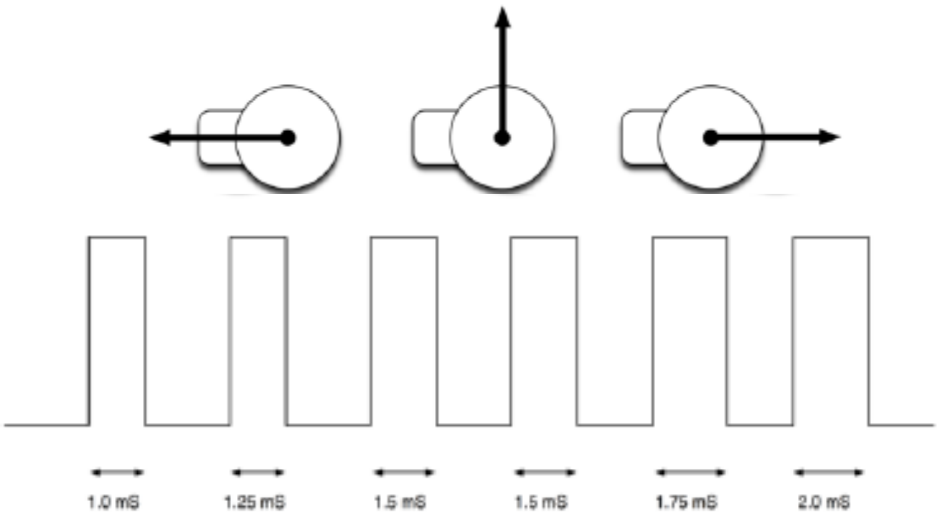
Le montage d'un servo est très simple, il y a 3 fils. Le rouge va au 5V, le noir à la masse(GND) et le jaune sur une broche digitale.




**<u>ACTIVITE I :</u> Explorer les les servo- moteurs et les instructions arduino permettant de les commander.**

## Servo Motors

The position of the servo motor is set by the length of a pulse. The servo expects to receive a pulse roughly every 20 milliseconds. If that pulse is high for 1 millisecond, then the servo angle will be zero, if it is 1.5 milliseconds, then it will be at its centre position and if it is 2 milliseconds it will be at 180 degrees.




## Sample PWM Code

```
int ledPin = 9;      // LED connected to digital pin 9
int analogPin = 0 ;  // potentiometer connected to analog pin 0
int val = 0;         // variable to store the read value

void setup()
{
   pinMode(ledPin, OUTPUT);   // sets the pin as output
}

void loop()
{
   val = analogRead(analogPin);   // read the input pin
   analogWrite(ledPin, val / 4);  // analogRead values go from 0 to 1023,
                                  // analogWrite values from 0 to 255
}
```

**1) exploiter le petit morceau de code en faisant les simulations sur Proteus, puis vérifier experimentalement le fonctionnement .Expliquer le principe de fonctionnement.**

**2)**Modifier le programme pour afficher la position du servo sur le moniteur série et su un afficheur LCD.

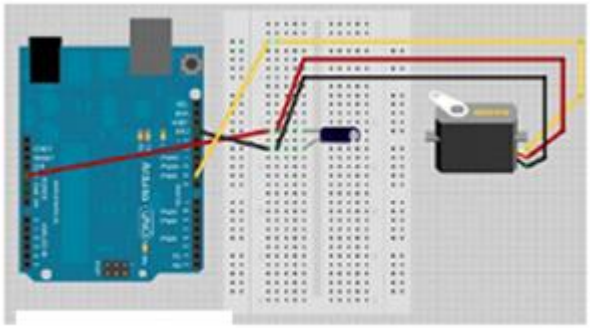
**3)**Supprimer le potentiomètre qui permet de fournir la valeur de la position (val) et utiliser le moniteur série pour envoyer la position via le clavier.

**NB** :utiliser **Serial.begin(9600); val =Serial.read();** dans le code arduino.

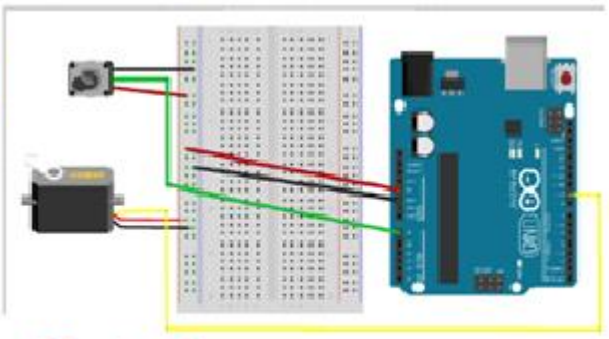
**CODE 1 :**Explorer le code suivant et interpréter les résultats obtenus.(Simulation+Réalisation) selon montage 1.

```
#include <Servo.h>

int servoPin = 9;

Servo servo;

int angle = 0;   // servo position in degrees

void setup()
{
  servo.attach(servoPin);
}
```

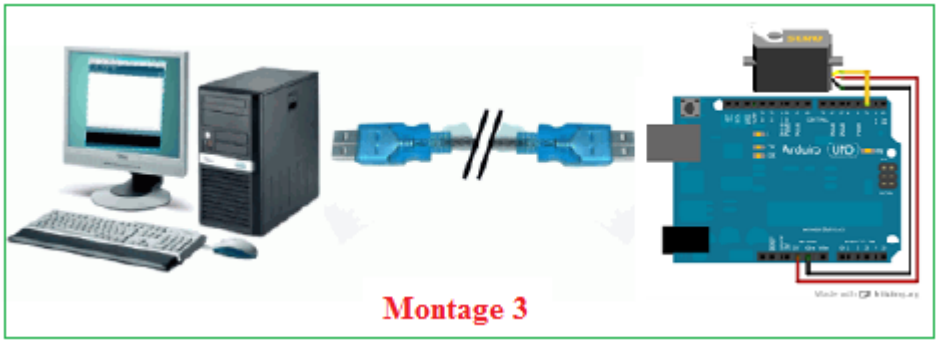
```
void loop()
{
  // scan from 0 to 180 degrees
  for(angle = 0; angle < 180; angle++)
  {
    servo.write(angle);
    delay(15);
  }
  // now scan back from 180 to 0 degrees
  for(angle = 180; angle > 0; angle--)
  {
    servo.write(angle);
    delay(15);
  }
}
```



Montage 1



Montage2



Montage 3

**CODE 2 :** Explorer le code suivant et interpréter les résultats obtenus.(Simulation+Réalisation)

```
#include <Servo.h>

int potPin = 0;
int servoPin = 9;
Servo servo;

void setup()
{
  servo.attach(servoPin);
}

void loop()
{
  int reading = analogRead(potPin);    // 0 to 1023
  int angle = reading / 6;             // 0 to 180-ish
  servo.write(angle);
}
```

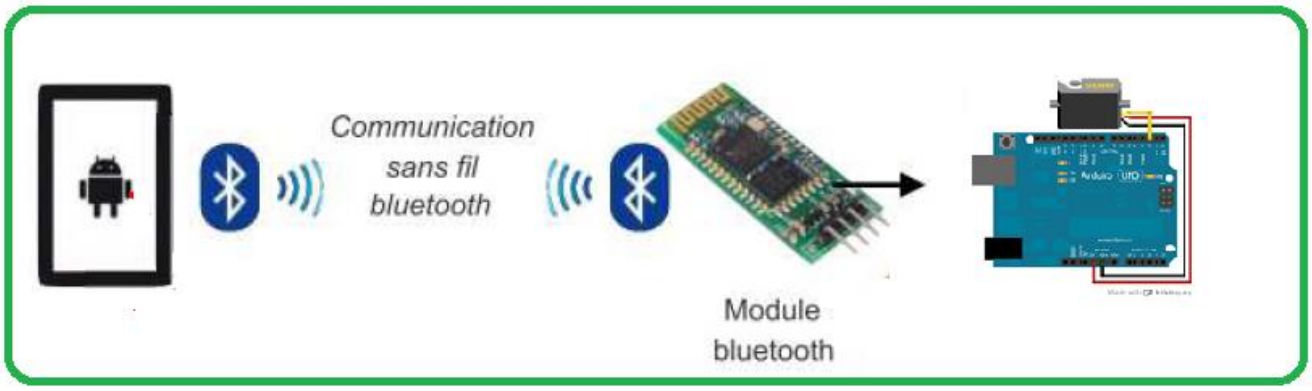
**CODE 3 :**

**1-** Explorer le code suivant et interpréter les résultats obtenus.(Simulation+Réalisation).

Il faut utiliser le moniteur série pour la commande comme illustré en **montage 3 .**

```
#include <Servo.h>
Servo monservo;
int pos = 0;

void setup()
{
Serial.begin(9600);
while (!Serial);
Serial.println("--------TP02 ESE19 PROJET-----");
Serial.println("Master 2022");
delay(1000);
Serial.println("-------------------------");
monservo.attach(9);
Serial.println("sweep");
monservo.write(0);
for(pos = 0; pos <= 180; pos ++){
monservo.write(pos);}
delay(1000);
for(pos = 179; pos >=0; pos --){
monservo.write(pos);}
monservo.write(0);
delay(1000);
Serial.println("end sweep");
Serial.println("-------------------------");
Serial.println("Entrez un engle entre 0 et 180");
Serial.println("-------------------------");
}

void loop() {

if (Serial.available())

{//*********************************
  int angle = Serial.parseInt();

if (angle >= 0 && angle <= 180)
{
  Serial.print("angle valide: ");
  Serial.println(angle);
  Serial.print("tourner le  servo à: ");
  Serial.print(angle);
  Serial.println(" degrés");
  monservo.write(angle);
  }
}//*********************************

}
```

**2-** Ajouter un module Bluetooth à votre carte arduino et Modifier le programme pour commander par liaison Bluetooth la position du servo en utilisant votre smartphone ANDROID.
La consigne est donnée par un curseur  d'une application android Bluetooth servo control.apk sur l'écran du smartphone ou par l'application android Bluetooth serial monitor.apk.Vous pouver les installer facilement depuis googleplay.com ou créer votre propre application utilisant Appinventor.



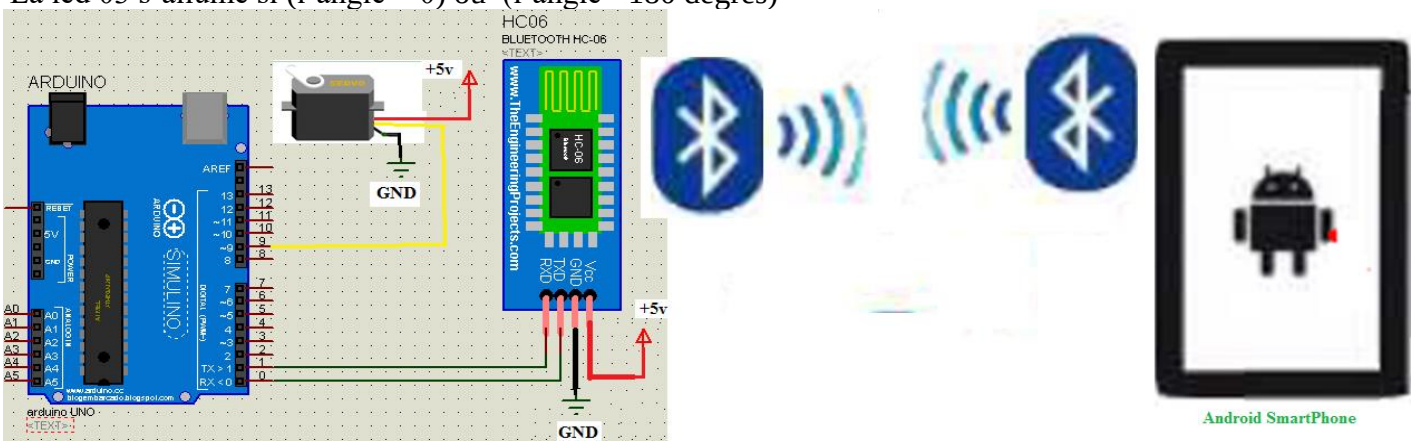
3-Que remarquez vous lorsque le moteur servo est proche des fins de course c.à.d pour les angles compris entre 0 et 10 degrés et les angles compris entre 170 et 180 degrés ?

3-Modifier le code pour allumer (03 Leds branchées aux pins 11,12,13 )comme suit :

-La led 01 s'allume si l'angle est compris entre 10 et 170 degrés

-La led 02 s'allume si (0<angle < 10) ou (170<angle < 180 degrés)

-La led 03 s'allume si (l'angle <0) ou (l'angle >180 degrés)




**ACTIVITE II : Soient les programmes Arduino suivants :**

**PROGRAM-A**

```
int led = 13;
int data;
void setup()
{
        Serial.begin(115200);
        pinMode(led, OUTPUT);
        Serial.println("C'est parti!");
}
void loop()
{
        data=Serial.read();
        if (data=='1') digitalWrite(led, HIGH);
        if (data=='0') digitalWrite(led, LOW);
}
```

**PROGRAM-B**

Tester le programme ci-dessous

```
int led = 13;
int data;
void setup()
{
 Serial.begin(115200);
 pinMode(led, OUTPUT);
 Serial.println("C'est parti!");
}
void loop()
{
 data=Serial.read();
 if (data=='1')
  {
  digitalWrite(led, HIGH);
  Serial.println("LED ON");
  }
 if (data=='0') digitalWrite(led, LOW);
}
```

**PROGRAM C**

```
#include <Servo.h>
Servo monservo;
int data;
void setup()
{   Serial.begin(115200);
    Serial.println("  TP02 ESE19 PROJET  )
}
void loop()
{   data=Serial.read();
    switch(data)
        {       case '1':  monservo.write(45);  break;
                case '2':  monservo.write(90):  break;
                case '3':  monservo.write(135); break;
                case '4':  monservo.write(170);  break;
        }
   }
```

**1)Faire les simulations nécessaires sur Proteus et bien comprendre le princpe de fonctionnement. Donner une explication détaillée et illuster avec des copies d'écran.**

**2)Mettre en oeuvre les programmes ci-dessus (A, B et C ) sur carte Arduino et bien vérifier le fonctionnement.**

**Tester le programme en envoyant le caractère 0, puis le caractère 1 et en observant la led**

**3) Rappeler le rôle des instructions suivantes :**

**Serial.begin(115200); pinMode(led, OUTPUT); Serial.println("C'est parti!"); digitalWrite(led, HIGH);**

**4)Indiquer le rôle de la nouvelle instruction: Serial.read();**

**NB:**Pour envoyer une donnée (code ascii du caractère) sur le port série de la carte Arduino, il faut entrer une valeur puis cliquer « envoyer », dans la fenêtre du moniteur série

**5)** Tester le programme **C** ci-dessus et décrire son comportement.

**6)** Retirer une instruction break ; et tester. Décrire ce qui se passe.

**ACTIVITE IV(mini-projet TP)**

Vous avez 02 potentiomètres branchés aux broches A0 et A1 du port analogique Arduino1.La carte Arduino1 est branchée par fils via les broches numériques 0 et 1 à une autre carte Arduino2 :TX (ARDUINO 1)➔RX(ARDUINO 2).

Celle-ci(La carte ARDUINO 2) reçoit les valeurs des potentiomètres Vpot1,Vpot2 et les affiche sur un LCD 2x16 de la carte arduino2 et en même temps commander la position de deux servomoteurs servo1 et servo2 branchés aux broches 3 et 9 de la carte arduino2.

1)Etablir le schéma correspondant à cette application et écrire le code arduino permettant l'envoi ,la réception et l'affichage sur LCD des positions des deux servos.

2)Faires les simulation nécessaires sur Proteus et Bien illustrer votre travail par des explications et des copies d'écran.

3)Mettre en œuvre votre programme sur le Kit Arduino et vérifier expérimentalement le fonctionnement.

4)Remplacer les fils reliant Tx et Rx par des modules de communication Bluetooth .Vérifier le fonctionnement de cette partie simulation puis pratiquement. Fournir une explication détaillée du principe de fonctionnement de l'émission et de la réception sans fils des données pour commander la position des servomoteurs.

**Vous visitez les liens suivants :**

https://community.element14.com/members-area/personalblogs/b/blog/posts/app-inventor-bluetooth-and-arduino-part-3-android-slider-controls-led-brightness

https://apkdownloadforwindows.com/fr/download/86541/1/

http://electroniqueamateur.blogspot.com/2017/06/programmer-une-appli-android-pour.html

https://fr.aptoide.com/download?app_id=42692458&store_name=aptoide-web&entry_point=appstore_appview_header_desktop