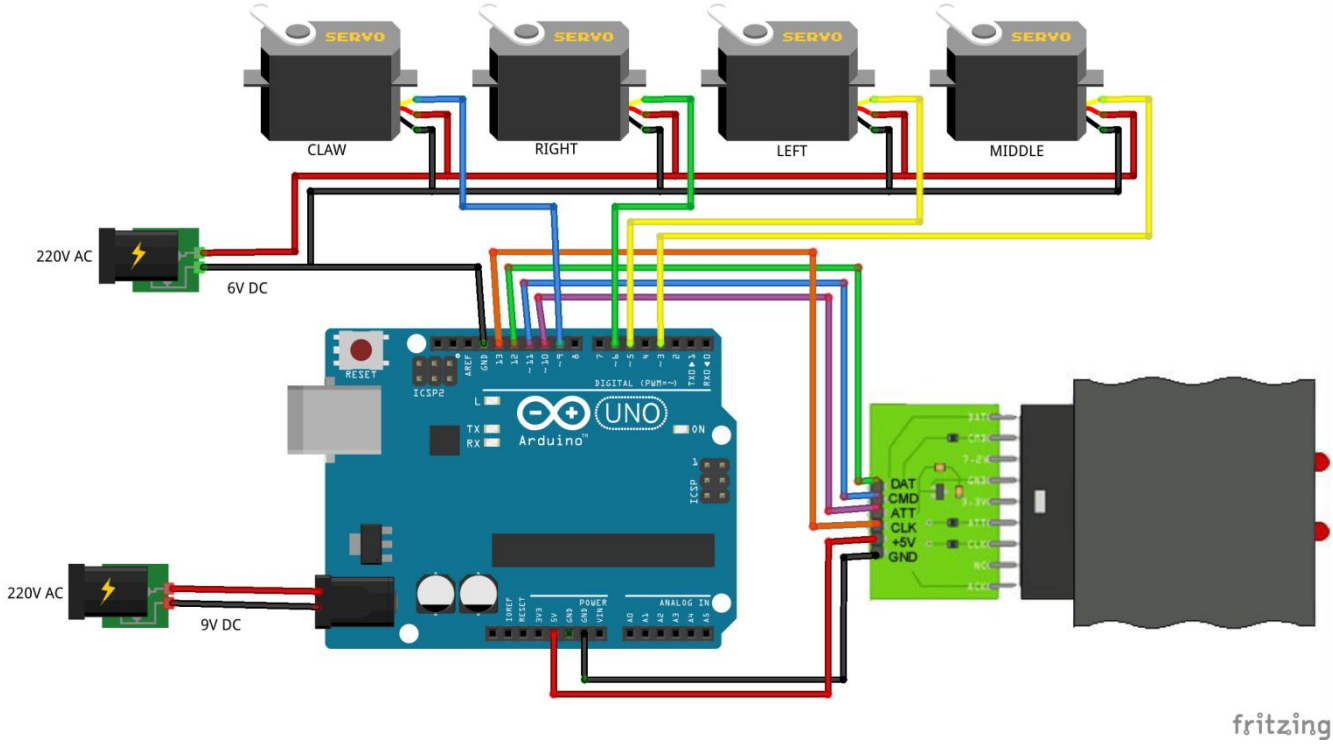
Pilotage robot MeArm avec manette de PS2

1. Câblage :

J'ai utilisé un petit robot MeArm de chez Mime Industries (https://mime.co.uk), disponible à la vente sur la plupart des sites de robotique amateur ou dont le .DXF pour découpe laser est dispo gratuitement sur http://www.thingiverse.com/thing:993759/#files. On a aussi utilisé une manette de PS2 revue et corrigée par Lynxmotion disponible sur http://www.robotshop.com/eu/fr/controleur-robot-ps2-v4-lynxmotion.html, et un microcontrôleur Arduino Genuino/Uno. Deux alimentations externes seront utilisées : une en 9V pour le microcontrôleur, une pour les servomoteurs du robot en 6V. Attention, prévoir une puissance suffisante sur cette seconde alimentation : le traditionnel boitier de piles est un peu juste, il vaut mieux lui préférer un bloc prise 200VAC / 6VCC.

Attention, au moment des branchements, il vaut mieux tout brancher sauf l'alim des servomoteurs, téléverser le code et ensuite seulement alimenter les servomoteurs.



Pin système	Signal	Description	Branchement Pin Arduino
PS2 – CLCK	Clock	Horloge de transmission de données	13
PS2 – ATTN	Attention		11
PS2 – COM	Command		10
PS2 – DATA	Data	Transmission des données de PS2	12
PS2 – PS	Alimentation		+5V
PS2 – GND	Masse		GND

Middle	Position	Position servo en degrés	3
Left	Position	Position servo en degrés	5
Right	Position	Position servo en degrés	6
Claw	Position	Position servo en degrés	9

2. Code :

Deux séries de codes : un pour la mise à zéro des servomoteurs indispensable avant le montage même du robot MeArm, l'autre pour le pilotage du robot monté ensuite. Attention à plusieurs choses : l'utilisation de la manette PS2 Lynxmotion nécessite l'utilisation pour Arduino spécifique qui est téléchargeable sur http://www.billporter.info/2010/06/05/playstation-2-controller-arduino-library-v1-0/, bien suivre le ReadMe pour savoir où coller les fichiers de la librairie. Le protocole de communication utilisé par la manette est de l'ASCII, donc pas d'inquiétude, si des caractères bizarres font leur apparition au fil du débogage du style « € » ou autre. Se reporter aux tableaux de référence complets des codages ASCII pour voir à quoi cela correspond.

Code de mise à zéro initiale des servomoteurs

```
#include <Servo.h>

Servo middle, left, right, claw ;

void setup() {
 Serial.begin(9600) ;
 middle.attach(3);
 left.attach(5);
 right.attach(6);
 claw.attach(9) ;
}

void loop() {
middle.write(90);
left.write(90);
right.write(90);
claw.write(35);
delay(300);

}
```

Code de pilotage complet

```
/*****************************************************************
MEARM ROBOT PILOT BY PS2 LYNX CONTROLLER PROGRAM
*****************************************************************
Written by Nicolas Blanchard - (CC BY-NC-SA 4.0)
*****************************************************************/

#include <PS2X_lib.h>  /*ajout PS2 Controller Library */
#include <Servo.h>  /*ajout Servo Library */

PS2X ps2x;  /* creation PS2 Controller Class*/
Servo middle, left, right, claw ;  /*creation des classes servo*/

byte Type = 0;
byte vibrate = 0;
```

```
int RX = 0 , RY = 0 , LX = 0 , LY = 0;  /* mise a zero des valeurs de 2
joysticks*/
int middlePos = 90; /*positions initiales des servomoteurs*/
int leftPos = 90;
int rightPos = 90;
int clawPos = 35;
int speedAxis = 200; /*delay en ms qui correspond à la vitesse de
deplacement des moteurs*/
int servoStep = 2; /*increment de deplacement des servo en degres*/

void setup() {
  Serial.begin(9600) ;
  ps2x.config_gamepad(13, 11, 10, 12, true, true); /*assignation des
broches d'Arduino connectees au recepteur de la PS2*/
  Type = ps2x.readType();  /*lecture du type de manette PS2*/

  middle.attach(3);  /*assignation des broches d'Arduino connectees aux
servomoteurs*/
  left.attach(5);
  right.attach(6);
  claw.attach(9) ;

  middle.write(middlePos);  /*initialisation des positions des
servomoteurs*/
  left.write(leftPos);
  right.write(rightPos);
  claw.write(clawPos);
  delay(500);
}

void loop() {

  ps2x.read_gamepad();  /*lecture des valeurs de la PS2*/

  LY = ps2x.Analog(PSS_LY);  /*ecriture de ces valeurs ASCII dans des
variables entieres entre 0 et 255 en DEC*/
  LX = ps2x.Analog(PSS_LX);
  RY = ps2x.Analog(PSS_RY);
  RX = ps2x.Analog(PSS_RX);

  if (LX == 0) {  /*pour le servo middle, increment de la position en
fonction du joystick correspondant*/
    middlePos = middlePos + servoStep;
    Serial.print("Middle :");  /*ecriture de la position du servo sur
fenetre monitor*/
    Serial.println(middlePos);
    middle.write(middlePos);
  }

  if (LX == 255) {
    middlePos = middlePos - servoStep;
    Serial.print("Middle :");
    Serial.println(middlePos);
    middle.write(middlePos);
  }

  if (LY == 0) {
    leftPos = leftPos + servoStep;
    Serial.print("Left :");
    Serial.println(leftPos);
    left.write(leftPos);
```

```
  }

  if (LY == 255) {
    leftPos = leftPos - servoStep;
    Serial.print("Left :");
    Serial.println(leftPos);
    left.write(leftPos);
  }

  if (RY == 0) {
    rightPos = rightPos + servoStep;
    Serial.print("Right :");
    Serial.println(rightPos);
    right.write(rightPos);
  }

  if (RY == 255) {
    rightPos = rightPos - servoStep;
    Serial.print("Right :");
    Serial.println(rightPos);
    right.write(rightPos);
  }

  if (RX == 0) {
    clawPos = clawPos + servoStep;
    Serial.print("Claw :");
    Serial.println(clawPos);
    claw.write(clawPos);
  }

  if (RX == 255) {
     clawPos = clawPos - servoStep;
    Serial.print("Claw :");
    Serial.println(clawPos);
    claw.write(clawPos);
  }

  delay(speedAxis);
}
```