TP INITIATION ARDUINO

L'objectif de ce TP est d'effectuer une prise en main d'une carte programmée ARDUINO.

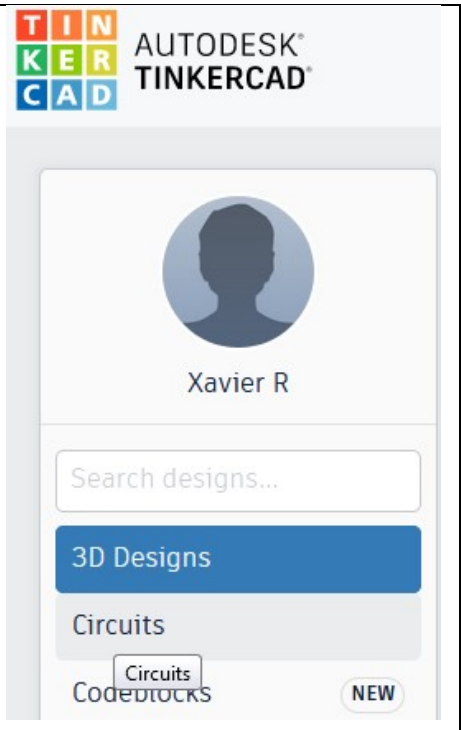
Le système Arduino est conçu autour d'une plateforme Open Source installée sur une carte programmée à microcontroleur permettant l'écriture, la compilation et le test d'un programme. Les cartes et modules Arduino sont conçus avec des entrées/sorties qui reçoivent des signaux de capteurs ou interrupteurs qui peuvent eux-mêmes commander des moteurs et éclairages par exemple. (cf https://www.arduino-france.com/tutoriels/quest-ce-que-arduino/)

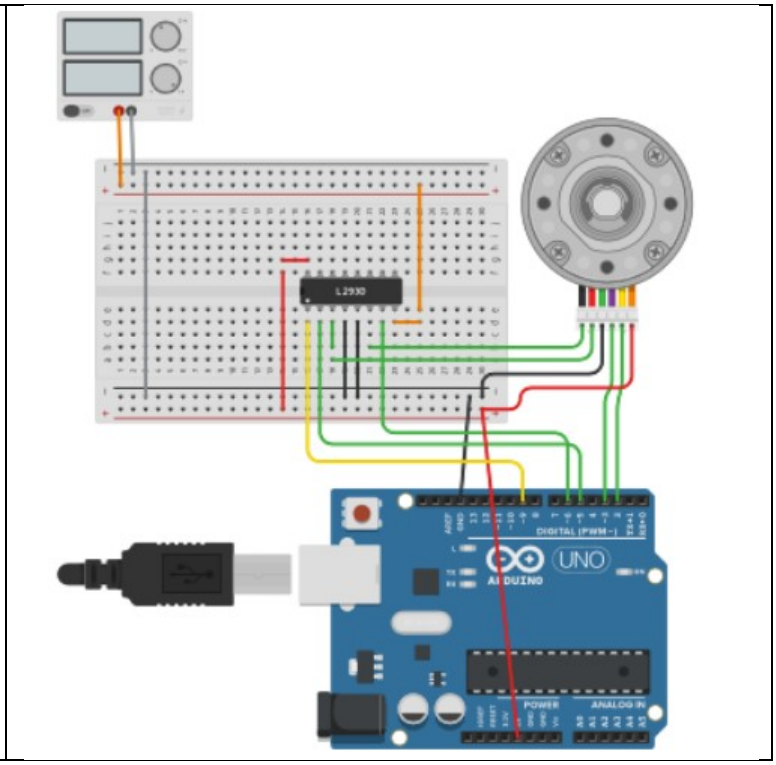
Le site https://www.tinkercad.com est un outil de modélisation 3D en ligne utilisable directement à partir d'un simple navigateur Internet. Il est conçu pour être facile à apprendre et à utiliser. Il permet également de simuler le comportement de circuit électronique, intégrant carte Arduino, moteur, codeur de position, ….

Les étapes sont les suivantes :

- Inscription sur le site tinkercad
- Réalisation du montage
- Importation d'un code source de pilotage du moteur
- Test
- Rajout d'une LED
- Affichage de la tension moteur
- Affichage signal PWM
- Rajout de l'affichage de la vitesse moteur
- Influence du rapport cyclique

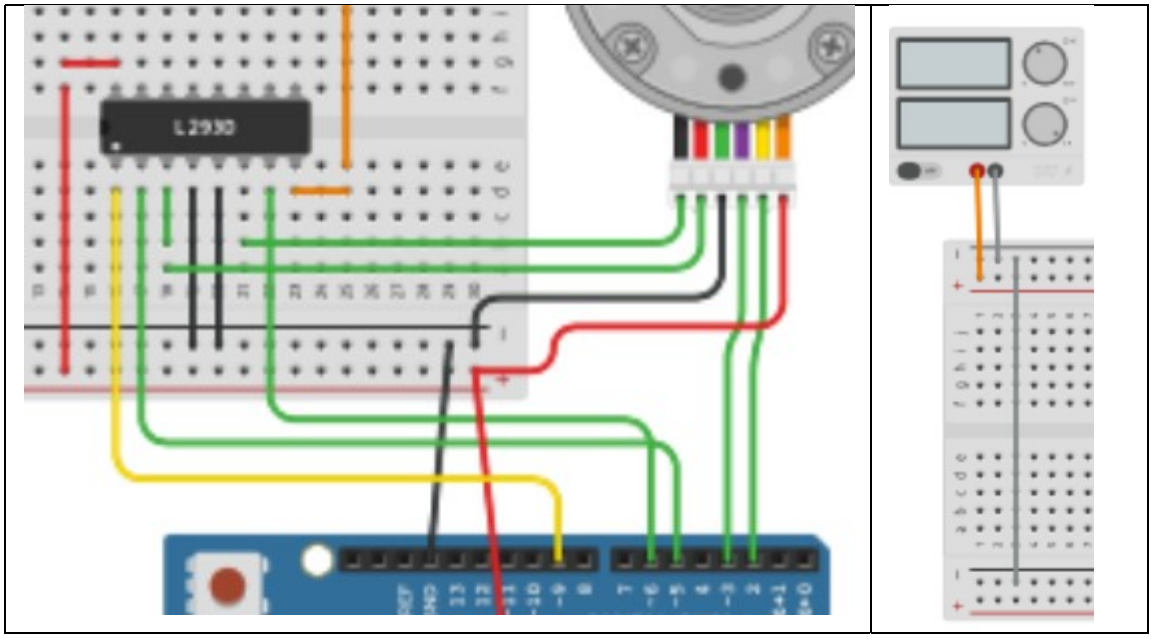
Après s'être inscrit sur le site Tinkercad (pensez à noter votre mot de passe) . Construire un nouveau circuit en respectant le câblage suivant :

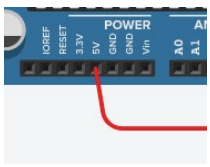




Les composants nécessaires sont les suivants :

- Arduino Uno R3
- Breadboard Small
- H-Bridge Motor Driver (L293D) : pont en H de pilotage du moteur
- DC Motor with Encoder (RPM : 45 (tr/min))
- Power Supply (Voltage 5V, Current 5V)



Ne pas oublier d'alimenter la carte ARDUINO en +5V. 

- Importer à l'aide d'un copier coller le code fourni en annexe et le coller dans la zone Code de Tinkercad.
- Consulter les différents sites référencés en début de code et expliquer rapidement sur votre compte rendu l'utilité des composants principaux.
- Lancer l'exécution du code. Vous devez observer le moteur tourner.
- De signifient les 3 nombres qui s'affichent dans la partie Serial Monitor située en bas à gauche de la fenêtre code. (cf annexe 2).
- Compléter le schéma en rajoutant une LED associée à une résistance en série de valeur 220 Ω. Consulter en autres le site : https://openclassrooms.com/fr/courses/2778161-programmez-vos-premiers-montages-avec-arduino/3285186-jeux-de-lumiere-avec-une-led-et-la-breadboard
- Modifier le code afin que la LED s'allume tant que le moteur n'a pas effectué un demi-tour. Tester votre modification.

Pour la suite, on cherche à mesurer et visualiser la tension d'alimentation délivrée par le pont en H.

- Rajouter sur votre câblage un multimeter (mesure de tension) afin de visualiser la tension d'alimentation moteur. Celle-ci doit être de valeur 4.24V en régime permanent.

- Modifier le code afin d'afficher en plus des autres valeurs la vitesse de rotation du moteur en degré/seconde. Pour cela, consulter par exemple, le site : https://arduino.blaisepascal.fr/les-codeurs-incrementaux/
- Rajouter sur votre câblage un oscilloscope (1 ms par division) afin de visualiser la tension d'alimentation moteur.
- Modifier la ligne :
  doMotor((control>=0)?HIGH:LOW, min(abs(control), 255));
  en remplaçant 255 par 100. Qu'observe t-on ?
- Expliquer le principe du pilotage par PWM.
- Rajouter sur votre câblage un oscilloscope (1 ms par division) afin de visualiser une des voies du codeur de position.
- Expliquer le principe du codeur de position.

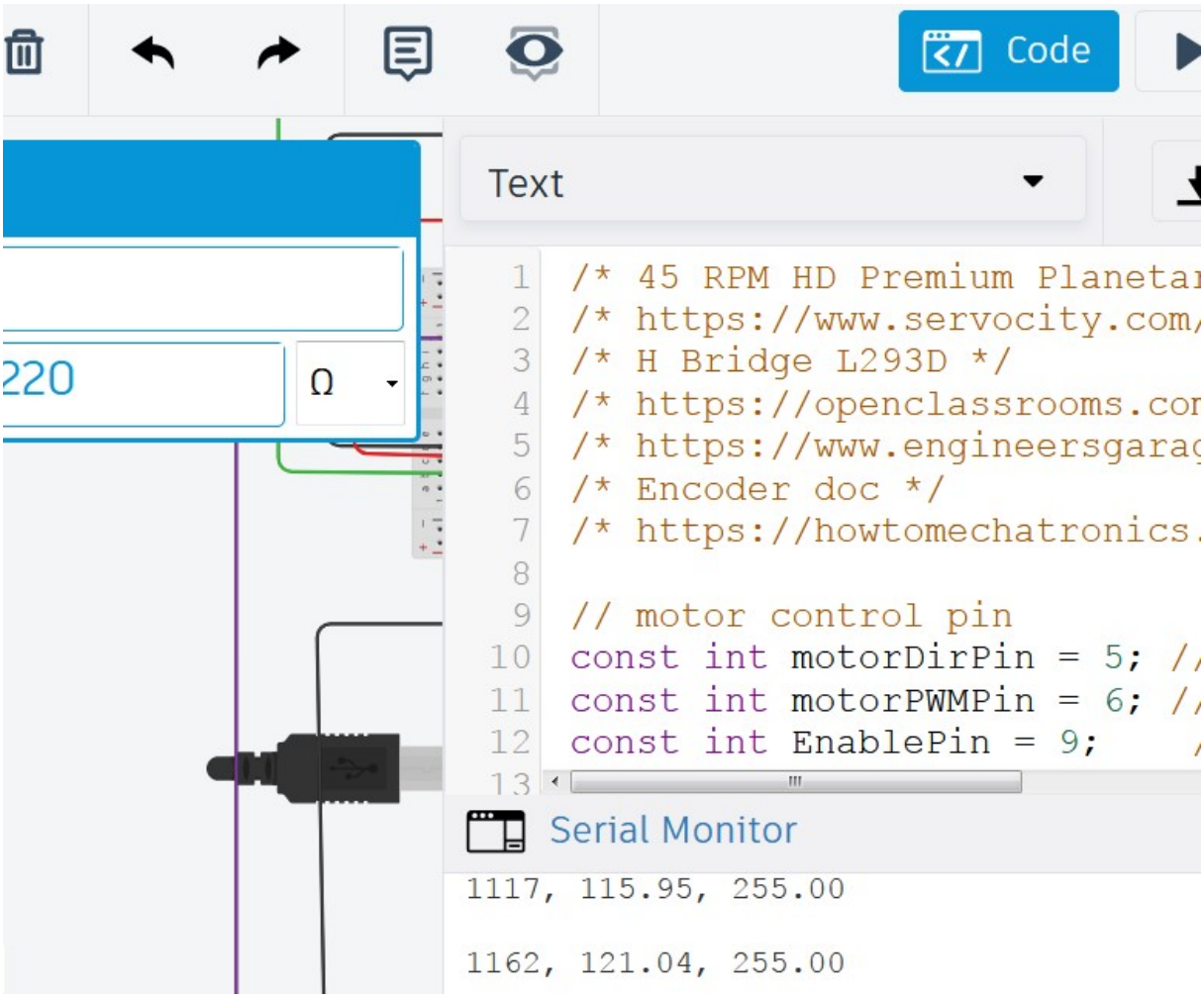
ANNEXE : code source de pilotage du moteur

```
/* 45 RPM HD Premium Planetary Gear Motor w/Encoder */
/* https://www.servocity.com/45-rpm-hd-premium-planetary-gear-motor-w-encoder */
/* H Bridge L293D */
/* https://openclassrooms.com/fr/courses/2778161-programmez-vos-premiers-montages-avec-arduino/3285355-le-moteur-a-courant-continu-partie-2-le-pont-en-h-et-les-circuits-integres */
/* https://www.engineersgarage.com/stm32/dc-motor-control-with-stm32-microcontroller/ */
/* Encoder doc */
/* https://howtomechatronics.com/tutorials/arduino/rotary-encoder-works-use-arduino/*/

// motor control pin
const int motorDirPin = 5; // Input 1
const int motorPWMPin = 6; // Input 2
const int EnablePin = 9;// Enable
// encoder pin
const int encoderPinA = 2;
const int encoderPinB = 3;
volatile int encoderPos = 0;
// encoder value change motor turn angles
const float ratio = 360./188.611/48.;
// 360. -> 1 turn
// 188.611 -> Gear Ratio
// 48. -> Encoder: Countable Events Per Revolution (Motor Shaft)

// Target value (degres)
float targetDeg = 360;

unsigned long time;

// Encoder Output A interruption function
void doEncoderA()
{
  encoderPos += (digitalRead(encoderPinA)==digitalRead(encoderPinB))?1:-1;
}

// Encoder Output B interruption function
void doEncoderB()
{
  encoderPos += (digitalRead(encoderPinA)==digitalRead(encoderPinB))?-1:1;
}

// MCC PWM function
void doMotor(bool dir, int vel)
{
  digitalWrite(motorDirPin, dir);
```

```
  analogWrite(motorPWMPin, dir?(255 - vel):vel);
}

// Setup function
void setup()
{
  Serial.begin(9600);

  pinMode(encoderPinA, INPUT_PULLUP);
  attachInterrupt(0, doEncoderA, CHANGE);

  pinMode(encoderPinB, INPUT_PULLUP);
  attachInterrupt(1, doEncoderB, CHANGE);

  pinMode(motorDirPin, OUTPUT);
  pinMode(EnablePin, OUTPUT);
}

// Main function
void loop()
{
  float motorDeg = float(encoderPos)*ratio;
  // Error
  float error = targetDeg - motorDeg;

  // Motor PWM command
  float control = (motorDeg<=targetDeg)?255:0;

  digitalWrite(EnablePin, 255);

  doMotor((control>=0)?HIGH:LOW, min(abs(control), 255));

  // Time
  time = millis();

  // Serial Monitor outputs

  //Serial.print("t (ms) : ");
  Serial.print(time);
  Serial.print(", ");
  //Serial.print("   motorDeg : ");
  Serial.print(float(encoderPos)*ratio);
  Serial.print(", ");
  //Serial.print("   error : ");
  //Serial.print(error);
  //Serial.print(", ");
  //Serial.print("    control : ");
```

```
            Serial.print(control);
            //Serial.print(", ");
            //Serial.print("    motorVel : ");
            Serial.println();

}
```

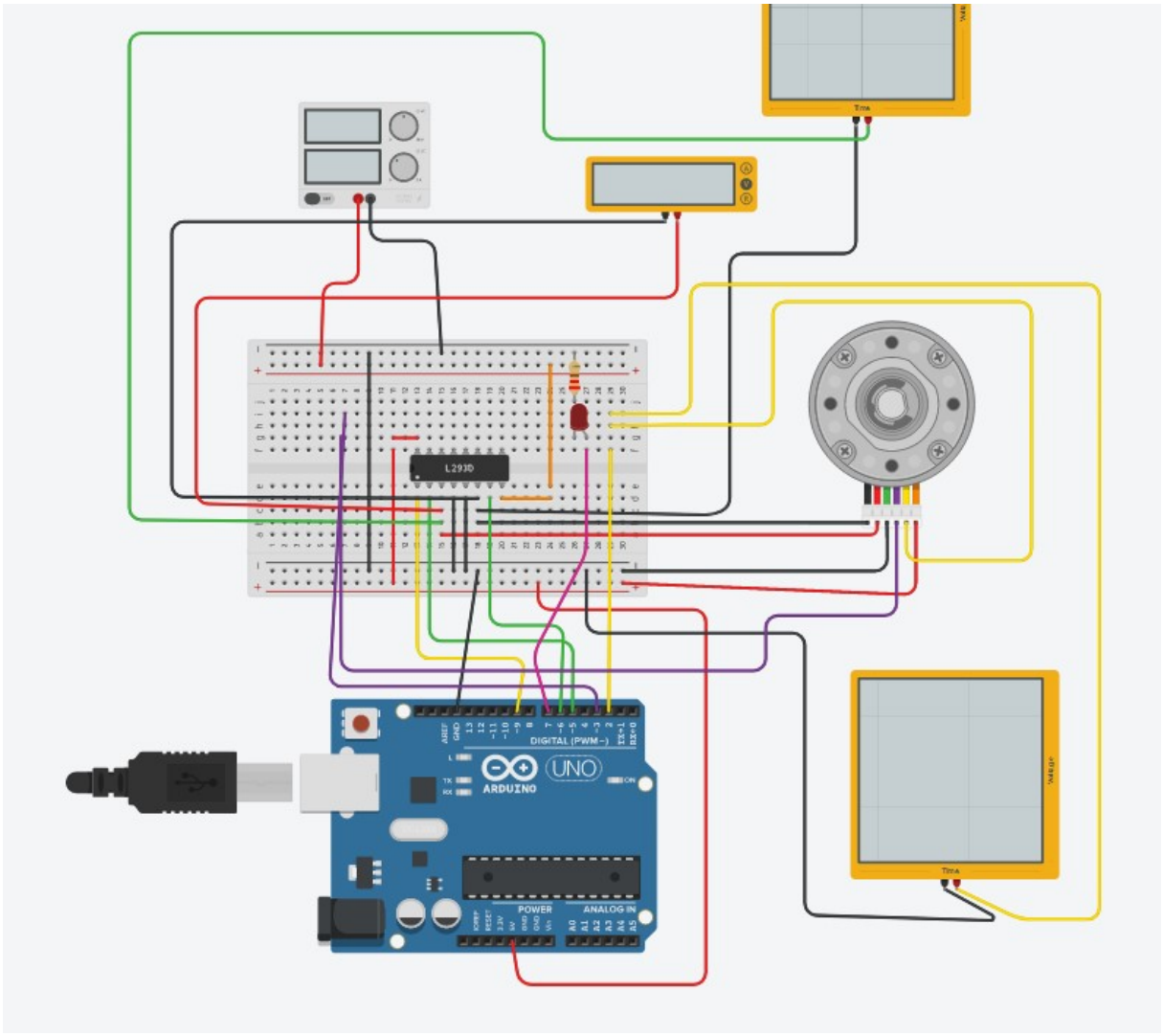
ANNEXE 2 :


Code
Text
220
Ω
1 /* 45 RPM HD Premium Planetar
2 /* https://www.servocity.com/
3 /* H Bridge L293D */
4 /* https://openclassrooms.com
5 /* https://www.engineersgarag
6 /* Encoder doc */
7 /* https://howtomechatronics.
8
9 // motor control pin
10 const int motorDirPin = 5; //
11 const int motorPWMPin = 6; //
12 const int EnablePin = 9;
13
Serial Monitor
1117, 115.95, 255.00
1162, 121.04, 255.00

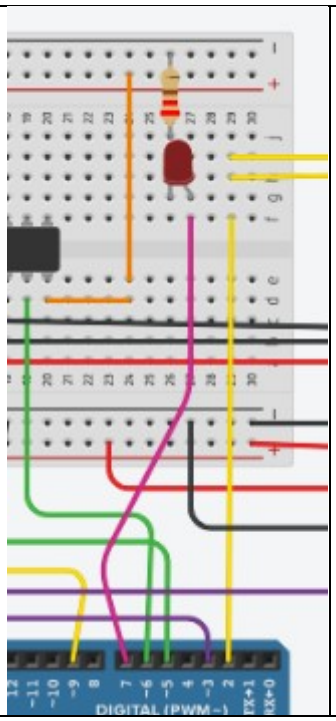
Eléments de corrigé :

Point de départ https://www.tinkercad.com/things/1jhbIgDeYnH

Le câblage final :



- Importer à l'aide d'un copier coller le code fourni en annexe et le coller dans la zone Code de Tinkercad. On observe le moteur tourner
- Consulter les différents sites référencés en début de code et expliquer rapidement sur votre compte rendu l'utilité des composants principaux. : Carte Arduino, moteur avec codeur et pont en H pour la commande PWM du moteur
- Lancer l'exécution du code. Vous devez observer le moteur tourner.
- De signifient les 3 nombres qui s'affichent dans la partie Serial Monitor située en bas à gauche de la fenêtre code. (cf annexe 2). Temps, position angulaire arbre sortie moteuren deg et paramètre de control PWM
- Compléter le schéma en rajoutant une LED associée à une résistance en série de valeur 220 Ω. Consulter en autres le site : https://openclassrooms.com/fr/courses/2778161-programmez-vos-premiers-montages-avec-arduino/3285186-jeux-de-lumiere-avec-une-led-et-la-breadboard

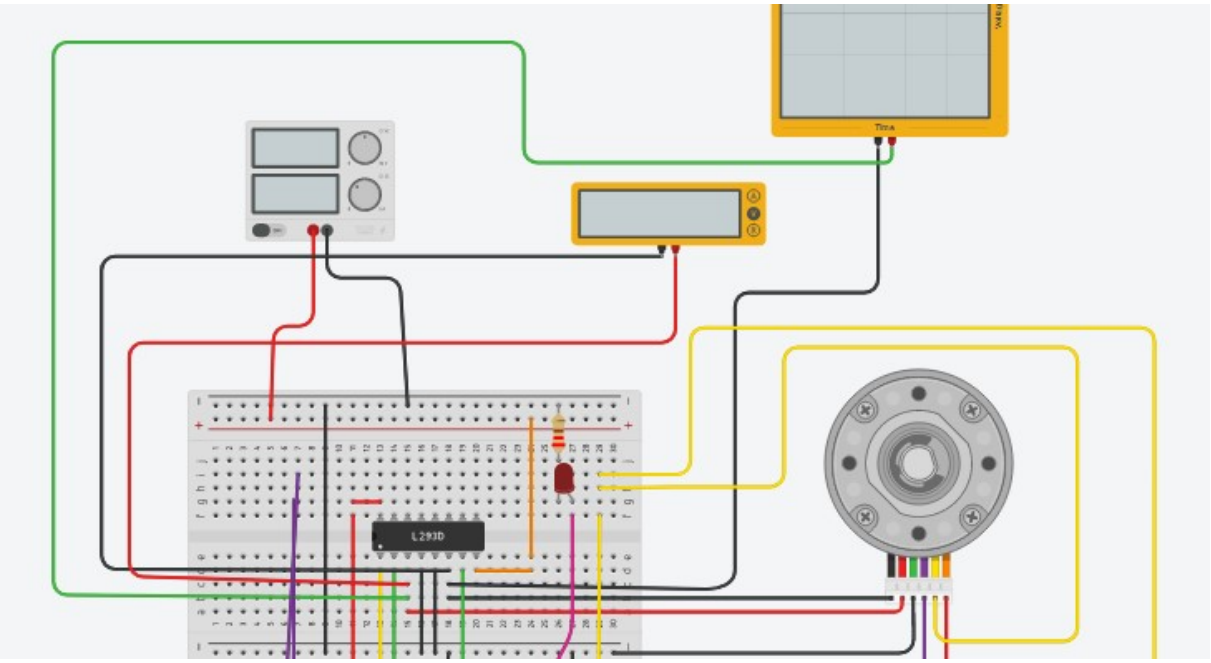



```
  pinMode(7,OUTPUT); //XR pour la LED
}


if (float(encoderPos)*ratio>180) {
  digitalWrite(7, LOW); // extinction de la LED
}
else {
  digitalWrite(7, HIGH); // allumage de la LED
}
```

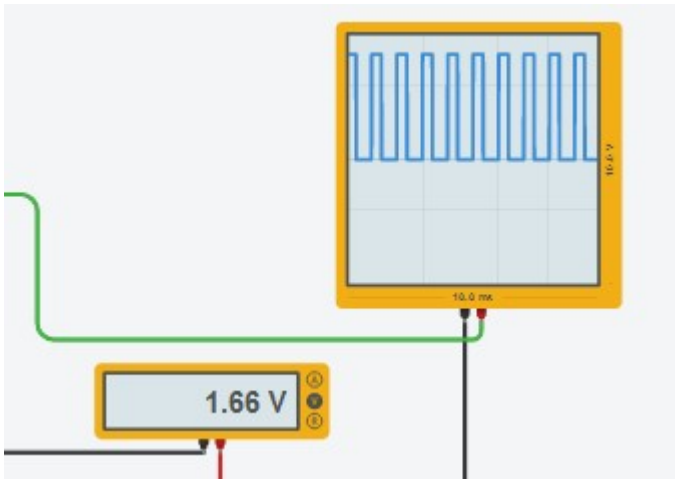
- Modifier le code afin que la LED s'allume tant que le moteur n'a pas effectué un demi-tour. Tester votre modification. Cf ci dessus

Pour la suite, on cherche à mesurer et visualiser la tension d'alimentation délivrée par le pont en H.

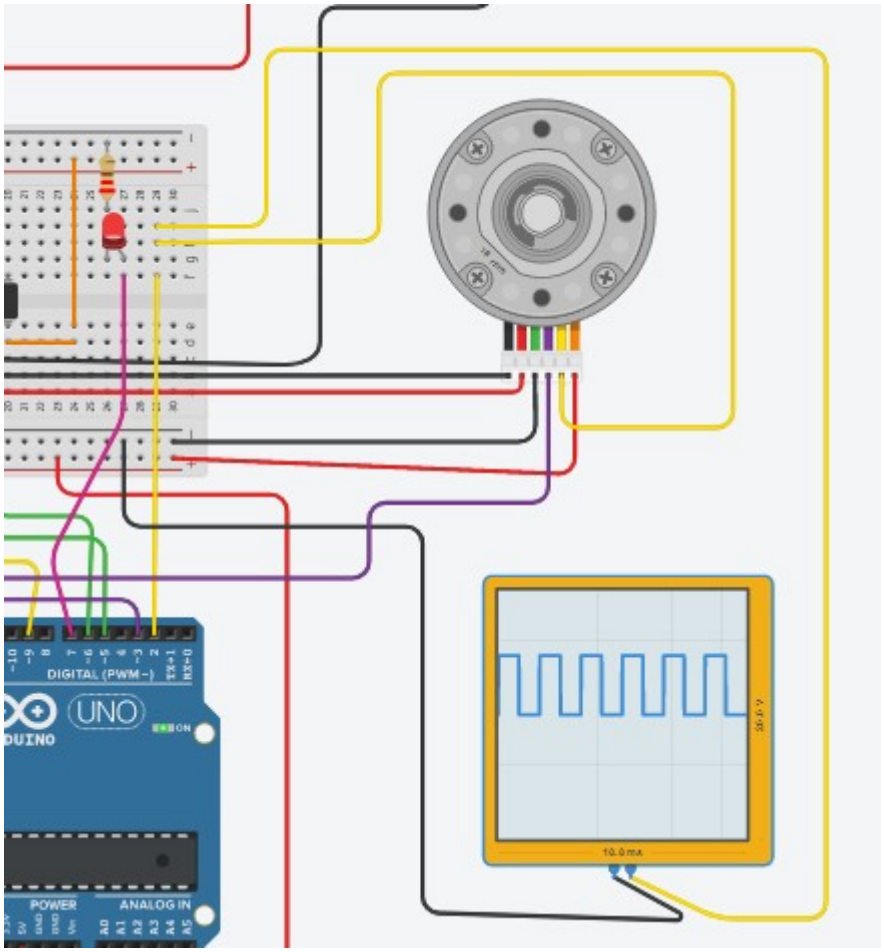
- Rajouter sur votre câblage un multimeter (mesure de tension) afin de visualiser la tension d'alimentation moteur. Celle-ci doit être de valeur 4.24V en régime permanent.
- Modifier le code afin d'afficher en plus des autres valeurs la vitesse de rotation du moteur en degré/seconde. Pour cela, consulter par exemple, le site : https://arduino.blaisepascal.fr/les-codeurs-incrementaux/
- Rajouter sur votre câblage un oscilloscope (1 ms par division) afin de visualiser la tension d'alimentation moteur.



- Modifier la ligne :
  doMotor((control>=0)?HIGH:LOW, min(abs(control), 255));
  en remplaçant 255 par 100. Qu'observe t-on ?
  control = 255 : vitesse de rotation = 96 deg/s
  control = 100 : vitesse de rotation = 37 deg/s



- Expliquer le principe du pilotage par PWM. On joue sur le rapport cyclique pour modifier la valeur moyenne de la tension d'alimentation du moteur.
- Rajouter sur votre câblage un oscilloscope (1 milliseconde par division) afin de visualiser une des voies du codeur de position.



- Expliquer le principe du codeur de position.
  Codeur 2 pistes, on détecte le passage des fronts sur les deux pistes.

Le code source du corrigé : code final pour allumer la LED, et rajouter l'affichage de la vitesse.

/* 45 RPM HD Premium Planetary Gear Motor w/Encoder */

/* https://www.servocity.com/45-rpm-hd-premium-planetary-gear-motor-w-encoder */

/* H Bridge L293D */

/* https://openclassrooms.com/fr/courses/2778161-programmez-vos-premiers-montages-avec-arduino/3285355-le-moteur-a-courant-continu-partie-2-le-pont-en-h-et-les-circuits-integres */

/* https://www.engineersgarage.com/stm32/dc-motor-control-with-stm32-microcontroller/ */

/* Encoder doc */

/* https://howtomechatronics.com/tutorials/arduino/rotary-encoder-works-use-arduino/*/

// motor control pin

```
const int motorDirPin = 5; // Input 1
const int motorPWMPin = 6; // Input 2
const int EnablePin = 9; // Enable
// encoder pin
const int encoderPinA = 2;
const int encoderPinB = 3;
volatile int encoderPos = 0;
volatile int pos = 0; // ancienne position angulaire en point codeur
volatile float motorVel = 0;  // Vitesse (en degre par seconde) du moteur
volatile unsigned long t = 0;  // temps "courant" (en millisecondes)
float ancmotorDeg = 0;  // ancienne position angulaire en degre
// encoder value change motor turn angles
const float ratio = 360./188.611/48.;
// 360. -> 1 turn
// 188.611 -> Gear Ratio
// 48. -> Encoder: Countable Events Per Revolution (Motor Shaft)

// Target value (degres)
float targetDeg = 360*2;

unsigned long time;

// Encoder Output A interruption function
void doEncoderA()
{
  encoderPos += (digitalRead(encoderPinA)==digitalRead(encoderPinB))?1:-1;
}

// Encoder Output B interruption function
```

```
void doEncoderB()
{
  encoderPos += (digitalRead(encoderPinA)==digitalRead(encoderPinB))?-1:1;
}

// MCC PWM function
void doMotor(bool dir, int vel)
{
  digitalWrite(motorDirPin, dir);
  analogWrite(motorPWMPin, dir?(255 - vel):vel);
}

// Setup function
void setup()
{
  Serial.begin(9600);

  pinMode(encoderPinA, INPUT_PULLUP);
  attachInterrupt(0, doEncoderA, CHANGE);

  pinMode(encoderPinB, INPUT_PULLUP);
  attachInterrupt(1, doEncoderB, CHANGE);

  pinMode(motorDirPin, OUTPUT);
  pinMode(EnablePin, OUTPUT);

  pinMode(7,OUTPUT); //XR pour la LED
}
```

```
// Main function
void loop()
{
  float motorDeg = float(encoderPos)*ratio;
  // Error
  float error = targetDeg - motorDeg;

  // Motor PWM command
  float control = (motorDeg<=targetDeg)?255:0;

  digitalWrite(EnablePin, 255);

  doMotor((control>=0)?HIGH:LOW, min(abs(control), 255));

  // Time

  time = millis();
  unsigned long dt = time - t;   // Temps écoulé depuis le dernier front
  t = time ; //pour stocker l'ancien temps
  if (dt > 0)   {
     motorVel = 1e3*(motorDeg-ancmotorDeg)/dt;        // Calcul de la vitesse (ici en pas par seconde)
   }
  ancmotorDeg = float(motorDeg);   // ancienne position angulaire en deg

  if (float(encoderPos)*ratio>180) {
    digitalWrite(7, LOW); // extinction de la LED
  }
  else {
    digitalWrite(7, HIGH); // allumage de la LED
```

```
}


// Serial Monitor outputs


//Serial.print("t (ms) : ");
Serial.print(time);
//Serial.print(", ");
//Serial.print("   encoderPos : ");
//Serial.print(encoderPos);
//Serial.print(", ");
//Serial.print("   targetDeg : ");
//Serial.print(targetDeg);
Serial.print(", ");
//Serial.print("   motorDeg : ");
Serial.print(float(encoderPos)*ratio);
Serial.print(", ");
Serial.print(ancmotorDeg);
Serial.print(", ");
Serial.print(motorDeg);
Serial.print(", ");
Serial.print(dt);
Serial.print(", ");
//Serial.print("   error : ");
//Serial.print(error);
//Serial.print(", ");
//Serial.print("    control : ");
Serial.print(control);
//Serial.print(", ");
Serial.print("    motorVel : ");
```

```
  Serial.print(motorVel);
  //Serial.println(min(abs(control), 255));
  Serial.println();

}
```