Moteur pas à pas bipolaire de l’axe linéaire du labo

# 

# Présentation et modèle :

Le moteur pas à pas dont nous disposons ici est un moteur 200 pas par tour, bipolaire.

Etudions le comportement d'un moteur composé d'un aimant permanent (rotor) et de 2 bobinages (phases) constitués chacun de 2 bobines. Il est possible de représenter sa structure à l’aide du schéma suivant :

**Bobinage 1**

pin1

N

S

Rotor (aimant permanent)

**Bobine 1.1**

**Bobinage 2**

pin3

pin4

**Bobine 2.1**

**Bobine 2.2**

Stator

**Bobine 1.2**

pin2

Un aimant permanent est solidaire de l'axe du moteur (rotor). Des bobines excitatrices sont placées sur la paroi du moteur (stator) et sont alimentées chronologiquement. Le rotor s'oriente suivant le champ magnétique créé par les bobines.

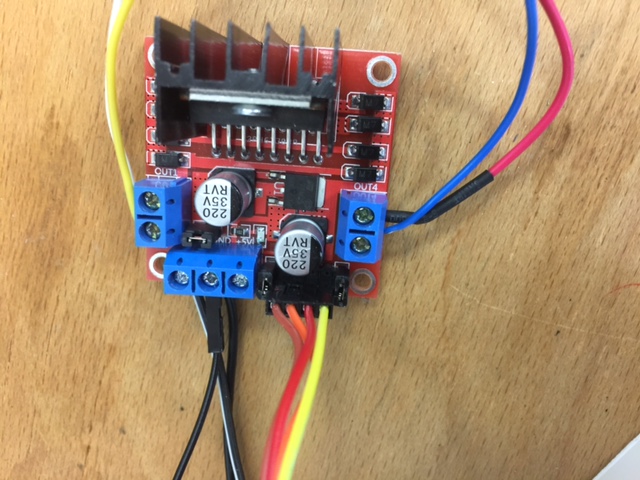
Définition des règles d'alimentation des bobines du moteur :

Nous allons ici pour ce modèle simple, établir une loi chronologique de 4 positions (4 pas) permettant de faire tourner le rotor dans le sens horaire.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| position 0 |  | |  |  |  | | --- | --- | --- | | N°pin | tension[V] | niveau logique | | 1 | 9 | 1 | | 2 | 0 | 0 | | 3 | 9 | 1 | | 4 | 0 | 0 | |
| position 1 |  | |  |  |  | | --- | --- | --- | | N°pin | tension[V] | niveau logique | | 1 | 0 | 0 | | 2 | 9 | 1 | | 3 | 9 | 1 | | 4 | 0 | 0 | |
| position 2 |  | |  |  |  | | --- | --- | --- | | N°pin | tension[V] | niveau logique | | 1 | 0 | 0 | | 2 | 9 | 1 | | 3 | 0 | 0 | | 4 | 9 | 1 | |
| position 3 |  | |  |  |  | | --- | --- | --- | | N°pin | tension[V] | niveau logique | | 1 | 9 | 1 | | 2 | 0 | 0 | | 3 | 0 | 0 | | 4 | 9 | 1 | |

# Interface de puissance : LM298

Réaliser le câblage décrit par l’image suivante :



Masse commune

+ alim

Vers bobine 1

Broche de connexion de l’alimentation externe (35V maxi)

ATTENTION à bien respecter la polarité

+ : à Gauche

Gnd : au centre

ATTENTION : il est nécessaire de relier la masse de l’arduino à la masse de la platine LM298

IN3 et IN4 commande bobine1 venant de 2 broches OUTPUT de l’arduino

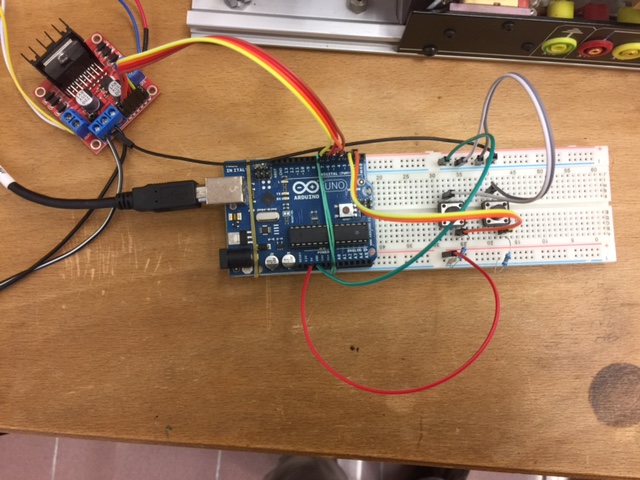
Le jumper ENB doit être en place

Vers bobine 2

IN1 et IN2 commande bobine2 venant de 2 broches OUTPUT de l’arduino

Le jumper ENA doit être en place

# Câblage commande du moteur pas à pas :



# Programme arduino :

Si pas d’appui sur le bouton marche : coulisseau à l’arret

Si appui sur le bouton marche : fonctionnement en marche avant du coulisseau

Si appui sur le bouton marche et appui sur le bouton inversion : fonctionnement en marche arrière du coulisseau

.

/\* Programme de commande de l'axe lineaire motorisé par un moteur pas à pas bipolaire

\* les deux bobines sont

\* bobine 1 : fil rouge et fil bleu

\* bobine 2 : fil jaune et fil blanc

\*

\* l'interface de puissance utilisée est un LM298

\*/

//affectation broches de commande du moteur :

// commande bobine 1 :

int IN1=4; //Connecté sur votre Arduino, Pin 3

int IN2=5; //Connecté sur votre Arduino, Pin 4

// commande bobine 2 :

int IN3=6; //Connecté sur votre Arduino, Pin 6

int IN4=7; //Connecté sur votre Arduino, Pin 7

// affectation boutons poussoirs :

int bp\_m=3; //bouton marche affecté pin3

int bp\_inv=2;// bouton inversion mouvement affecté bobine pin 2

//variable affectée au temps de communation entre bobines (ms) : t ideal = 2,3,4 ou 5 à tester

int t=2;

// procédure d'initialisation parcourue une fois

void setup() {

//Declaration en SORTIE des broches de commandes des bobines

pinMode(IN1,OUTPUT);

pinMode(IN2,OUTPUT);

pinMode(IN3,OUTPUT);

pinMode(IN4,OUTPUT);

// Déclaration en ENTREE des deux boutons poussoir de commande

pinMode(bp\_m,INPUT\_PULLUP); //bouton marche

pinMode(bp\_inv,INPUT\_PULLUP);// bouton inversion de sens

}

// boucle principale du programme parcourue infiniment

void loop(){

while(digitalRead(bp\_m)==0) { //tant que le bp marche est appuyé ...faire

if (digitalRead(bp\_inv)==0){ //Si bouton inversion appuyé alors marche arriere

marche\_ar();//utilisation procedure marche arriere

}

else { //dans le cas contraire : marche avant

marche\_av();//utilisation procedure marche avant

}

}//fin de tant que

}//fin de boucle principale

//------------------------------------------------

// defintion de la procedure marche avant

void marche\_av(){

digitalWrite(IN1,HIGH);

digitalWrite(IN2,LOW);

digitalWrite(IN3,HIGH);

digitalWrite(IN4,LOW);

delay(t);

digitalWrite(IN1,LOW);

digitalWrite(IN2,HIGH);

digitalWrite(IN3,HIGH);

digitalWrite(IN4,LOW);

delay(t);

digitalWrite(IN1,LOW);

digitalWrite(IN2,HIGH);

digitalWrite(IN3,LOW);

digitalWrite(IN4,HIGH);

delay(t);

digitalWrite(IN1,HIGH);

digitalWrite(IN2,LOW);

digitalWrite(IN3,LOW);

digitalWrite(IN4,HIGH);

delay(t);

}

//------------------------------------------------

// defintion de la procédure marche arrière

void marche\_ar(){

digitalWrite(IN1,HIGH);

digitalWrite(IN2,LOW);

digitalWrite(IN3,HIGH);

digitalWrite(IN4,LOW);

delay(t);

//position 2

digitalWrite(IN1,HIGH);

digitalWrite(IN2,LOW);

digitalWrite(IN3,LOW);

digitalWrite(IN4,HIGH);

delay(t);

//position3

digitalWrite(IN1,LOW);

digitalWrite(IN2,HIGH);

digitalWrite(IN3,LOW);

digitalWrite(IN4,HIGH);

delay(t);

//position 4

digitalWrite(IN1,LOW);

digitalWrite(IN2,HIGH);

digitalWrite(IN3,HIGH);

digitalWrite(IN4,LOW);

delay(t);

}