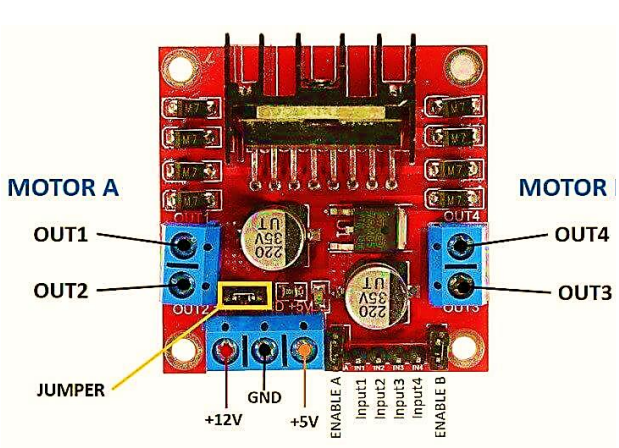
CORRECTION TD N°1 INTERFACES DE COMMUNICATION

Exercice 1

On désire commander deux moteurs dans le même sens, en utilisant la figure 1 compléter le schéma de câblage (figure2) et le programme 1.



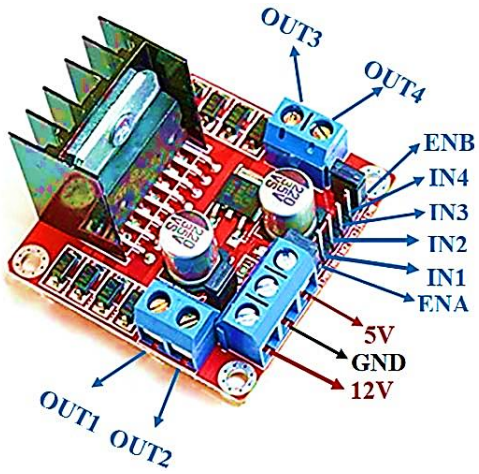


Figure 1

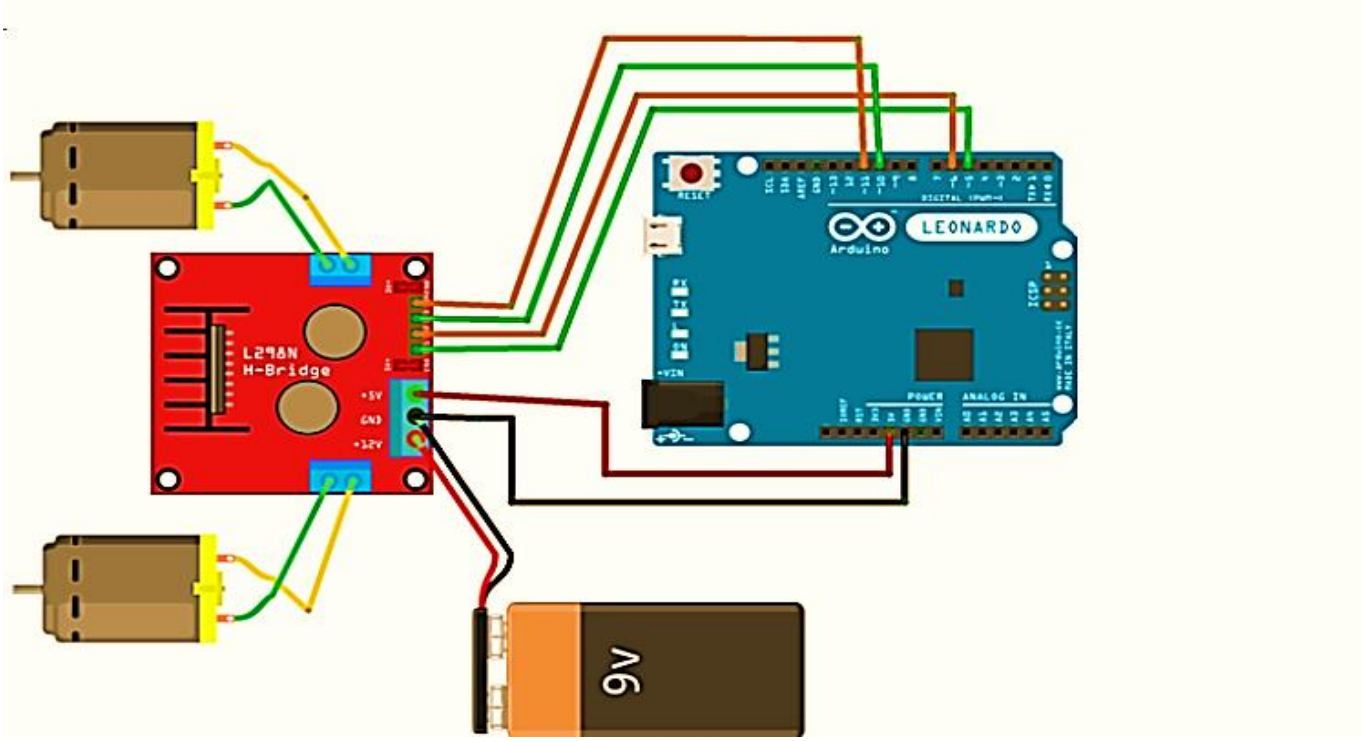


Figure 2 : Schéma de câblage

Programme 1 :

```
/ Moteur 1
int IN1 = 5;
int IN2 = 6;
// Moteur 2
int IN3 = 10;
int IN4 = 11;
void setup() {
 pinMode(IN1, OUTPUT);
 pinMode(IN2, OUTPUT);
 pinMode(IN3, OUTPUT);
 pinMode(IN4, OUTPUT);
}
void loop() {

 // Marche Avant
 digitalWrite(IN1, 1);
 digitalWrite(IN2, 0);
 digitalWrite(IN3, 1);
 digitalWrite(IN4, 0);

 // Attente 2s
 delay(2000);

 // Arrêt des moteurs
 digitalWrite(IN1, 0);
 digitalWrite(IN2, 0);
 digitalWrite(IN3, 0);
 digitalWrite(IN4, 0);

 // Attente 1s
 delay(1000);

 // Marche arrière
 digitalWrite(IN1, 0);
 digitalWrite(IN2, 1);
 digitalWrite(IN3, 0);
 digitalWrite(IN4, 1);

 // Attente 2s
 delay(2000);

 // Arrêt des moteurs
 digitalWrite(IN1, 0);
 digitalWrite(IN2, 0);
 digitalWrite(IN3, 0);
 digitalWrite(IN4, 0);

 // Attente 1s
 delay(1000);
}
```

Exercice 2

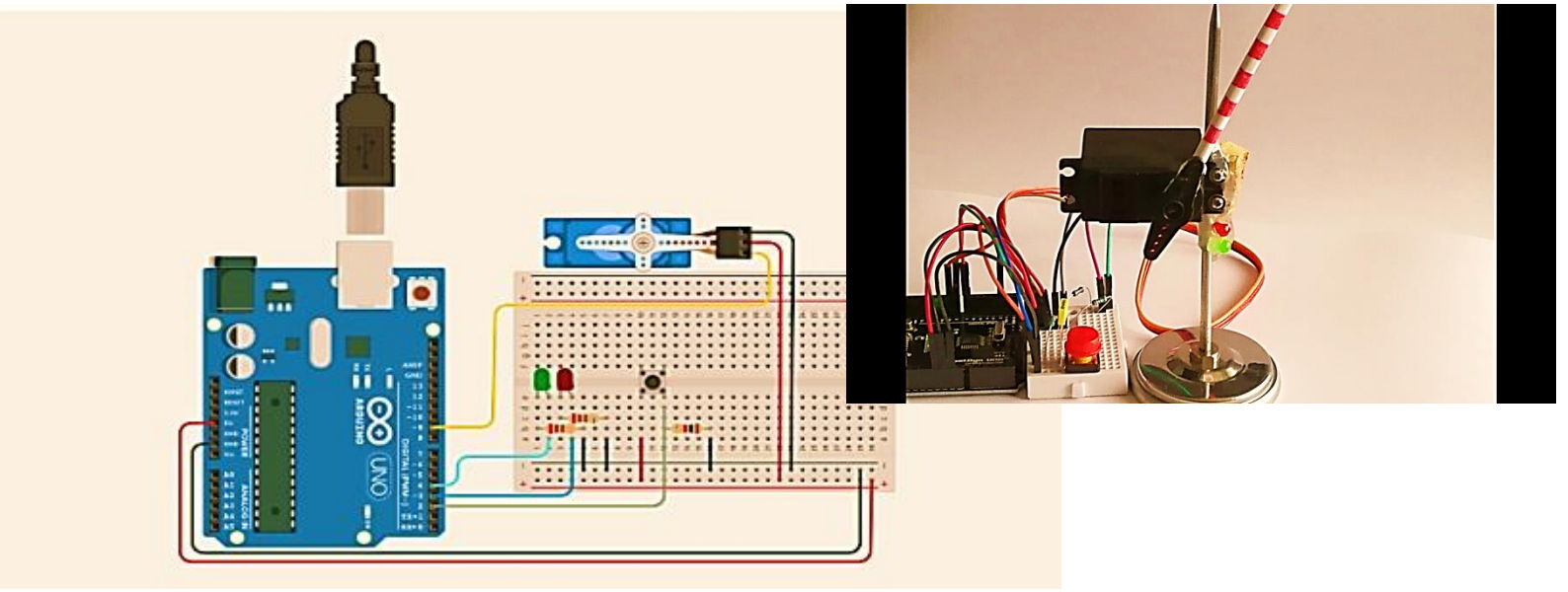


Figure3

PROGRAMME 2 :

Feu bicolore et barrière

Le montage :
* Une LED rouge sur la broche 3 en série avec une résistance de 220Ω
* Une LED verte sur la broche 4 en série avec une résistance de 220Ω

* Un servomoteur branché sur les broches 9, +5V et GND

* Bouton poussoir branché sur la broche 2 depuis +5V
* Une résistance de 1KΩ branché sur la broche 2 depuis GND

```
#include <Servo.h>

Servo servo; // création de l'objet servo


// 3 constantes
const int bouton = 2;
const int ledRouge = 3;
const int ledVerte = 4;

// 2 variables :
int etatBouton = 0;
int pos = 0;
void setup() {
// pour communiquer avec l'ordinateur
Serial.begin(9600);
```

```
// les broches des LED en sortie
pinMode(ledRouge, OUTPUT);
pinMode(ledVerte, OUTPUT);

// la broche bouton en entrée :
pinMode(bouton, INPUT);

// le servomoteur sur la broche 9
servo.attach(9);

// allume le feu rouge
digitalWrite(ledRouge, HIGH);

// barrière horizontal 0 ; verticale 90
servo.write(0);
}
// boucle infinie
void loop(){
// lire la caleur du bouton:
etatBouton = digitalRead(bouton);

// si le bouton est appuyé
if (etatBouton == HIGH) {
// alors message sur le moniteur série
Serial.print("Bouton appuye");

// et barrière à 90°
for(pos = 0; pos <= 90; pos++) {
servo.write(pos);
delay(15);
}

// Puis feu vert pendant 5 secondes
digitalWrite(ledRouge, LOW);
digitalWrite(ledVerte, HIGH);
delay(5000);

// et de nouveau le rouge
digitalWrite(ledVerte, LOW);
digitalWrite(ledRouge, HIGH);

// Enfin, barrière descend de 90° à 0°
for(pos = 90; pos>=0; pos--) {
servo.write(pos);
delay(15);
}
}
}
```

Exercice 3

Q1 : Parmi ces affirmations, cochez celles qui vous paraissent justes :

☐ Le BUS I2C est un BUS parallèle	☐ Le BUS I2C est unidirectionnel
☑ Le BUS I2C est un BUS série	☑ Le BUS I2C permet le transfert à 100kb^{s-1}
☑ Le BUS I2C est un BUS synchrone	☑ Le BUS I2C est un bus adressable
☐ Le BUS I2C est un BUS asynchrone	☑ Plusieurs octets peuvent être transmis en une seule trame
☐ N'importe quel circuit peut prendre la main sur le BUS I2C pour communiquer	☐ On ne peut relier que deux circuits sur un BUS I2C

Q2 : Combien de signaux sont utilisés sur un BUS I2C ? Donnez leur nom et leur rôle.

3 signaux sont nécessaires : SDA, SCL et GND

SDA : Signal de Données, contenant les octets transmis en Série

SCL : Signal d'horloge, permettant la synchronisation Maître/Esclave

GND : Masse Logique. Permet de fixer au même potentiel de référence les circuits

Q3 : Comment sont reliés les différents circuits I2C sur le BUS ?

Tous les composants sont câblés en parallèle sur le BUS. SDA <-> SDA et SCL <-> SCL

Q4 : Comment un circuit peut-il prendre la main sur le BUS ? Comment appelle-t-on cette opération dans la trame ?

Il faut passer le signal SDA de 1 à 0 lorsque le signal d'horloge SCL est à 1. Cette opération représente un START dans la communication.

Q5 : Comment un circuit libère-t-i le BUS ? Comment appelle-t-on cette opération dans la trame ?

Il faut passer le signal SDA de 0 à 1 lorsque le signal d'horloge SCL est à 1. Cette opération représente un STOP dans la communication.

Q6 : Quelle est la condition pour placer un bit (0 ou 1) sur le signal SDA ? A quel moment ce bit serat-il lu par l'esclave ?

Il faut obligatoirement que le signal SCL soit à 0. Le bit est lu sur SDA à chaque passage à 1 du signal d'horloge SCL

Q7 : Quel est l'élément de transmission permettant de vérifier que la communication des requêtes entre deux circuits se passe bien ?

Le maitre ou l'esclave doivent s'acquitter de la requête par un ACK (acknowledge = reconnaitre).

Q8 : Sur combien de bits est codée une adresse I2C ? Quel est le rôle du bit R/W ?

Une adresse I2C est codée sur 7 bits. L'adresse est complétée par R/W sur le poids faible.
Il correspond au mode Lecture (Read : R/W = 1) ou Ecriture (Write : R/W = 0).

Q9 : Pourquoi une communication entre Maitre / Esclave commence-t-elle toujours par une écriture ?

Le Maitre envoie en premier l'octet de COMMANDE, afin d'indiquer au circuit esclave ce qu'il attend de lui (lire une donnée, changer un paramètre, etc.). C'est donc obligatoirement une écriture de la commande avant tout envoi de données propres à chaque circuit.