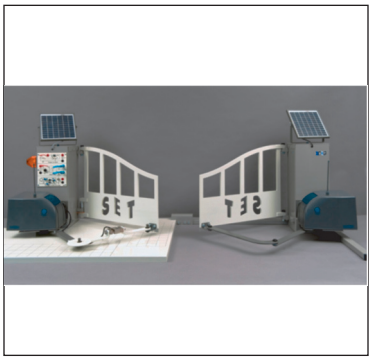
Dossier ressources : Portail automatique



1. Découverte du produit et de la problématique technique

Le système d'étude proposé est un portail destiné à la gestion du parking d'un complexe de vacances. L'accès au parking se fait à l'aide d'une télécommande remise aux clients du complexe. Dans certains cas le portail doit pouvoir être ouvert depuis l'accueil du complexe pour laisser entrer des entreprises de livraisons.

Dans le portail étudié, un automate :

- Gère l'ouverture/ fermeture du portail avec la télécommande.
- Gère la gestion des obstacles pour éviter tout accident.

Malheureusement le portail n'est pas connecté et la prise de contrôle à distance ne peut se faire que par la télécommande, ce qui contraint l'agent d'accueil à sortir de son poste pour que le signal atteigne le portail. La problématique est donc de trouver une solution pour rendre le portail connecté et pouvoir le commander via une interface web.

Un répéteur Wifi est installé et joignable depuis l'emplacement du portail

Paramètres répéteur

- **SSID répété** : Parking_Wifi_EXT
- **Mot de passe Wi-Fi répété** : Adm!n$P4ssw0rd_9Xq
- **Mode** : Repeater/Extender
- **Canal Wi-Fi** : aligné automatiquement sur le routeur
- **Sécurité** : WPA2-Personal

a. Caractéristiques

- Portail alimenté sur une batterie de 12 V elle-même alimentée par un panneau solaire (non étudiés).

b. Schéma structurel Existant

Sur le schéma fourni (Figure 2) :

- Le système d'ouverture/fermeture du vantail est représenté par un moteur à courant continu.
- La détection d'un obstacle est réalisée à l'aide d'un potentiomètre (Figure 1).

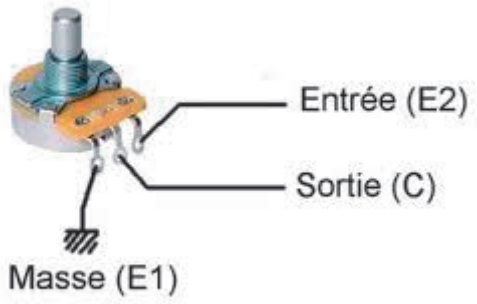


Figure1 : potentiomètre de détection d'obstacle

- L'interface web est représentée par deux boutons
- Les butées d'ouverture et de fermeture sont représentées par des boutons.
- Le feu clignotant est représenté par une LED orange.
- Un pont en H avec le composant L293D remplace le Shield moteur qui sera utilisé

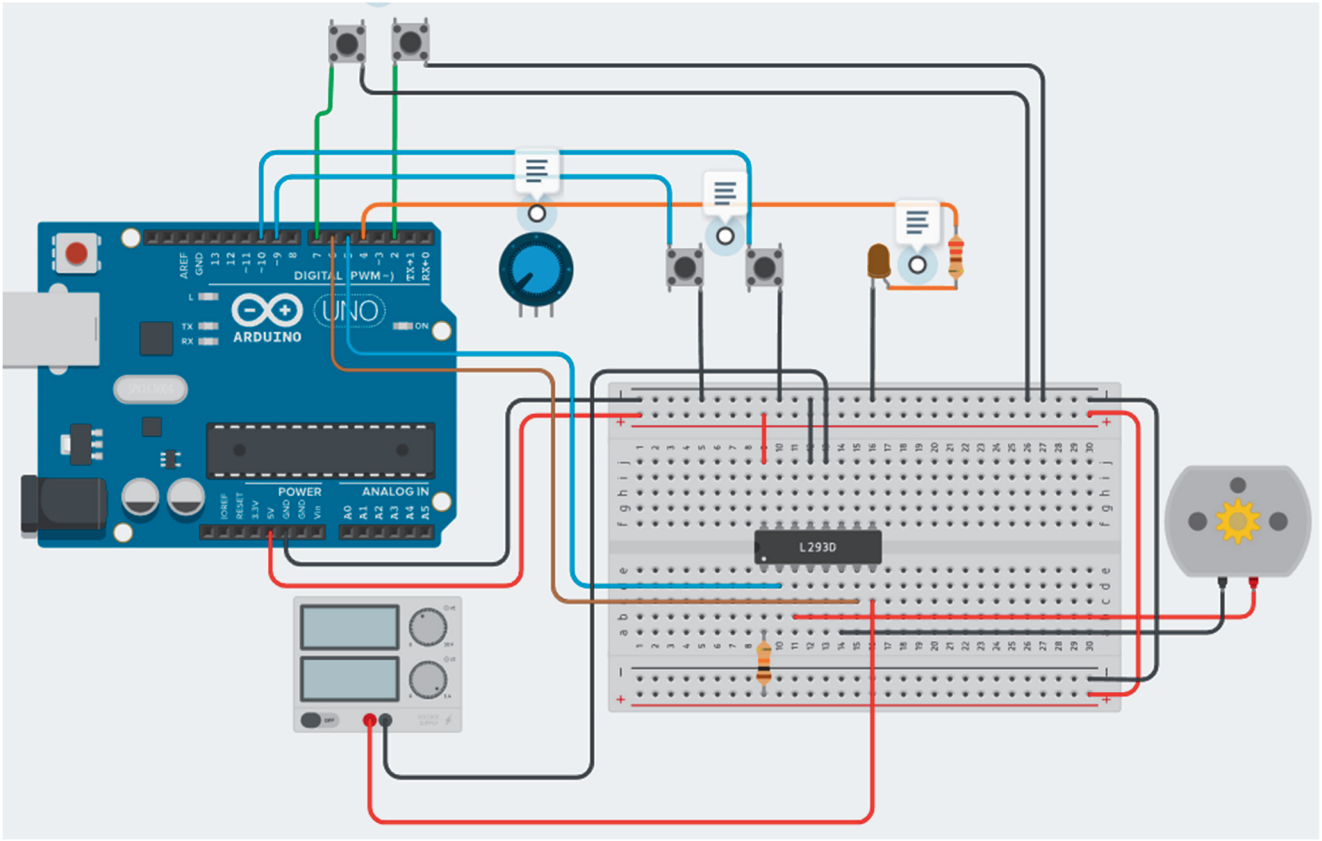


Figure 2 : schéma de fonctionnement du portail

c. Extrait du SysML – Diagramme de séquence (Figure 3)

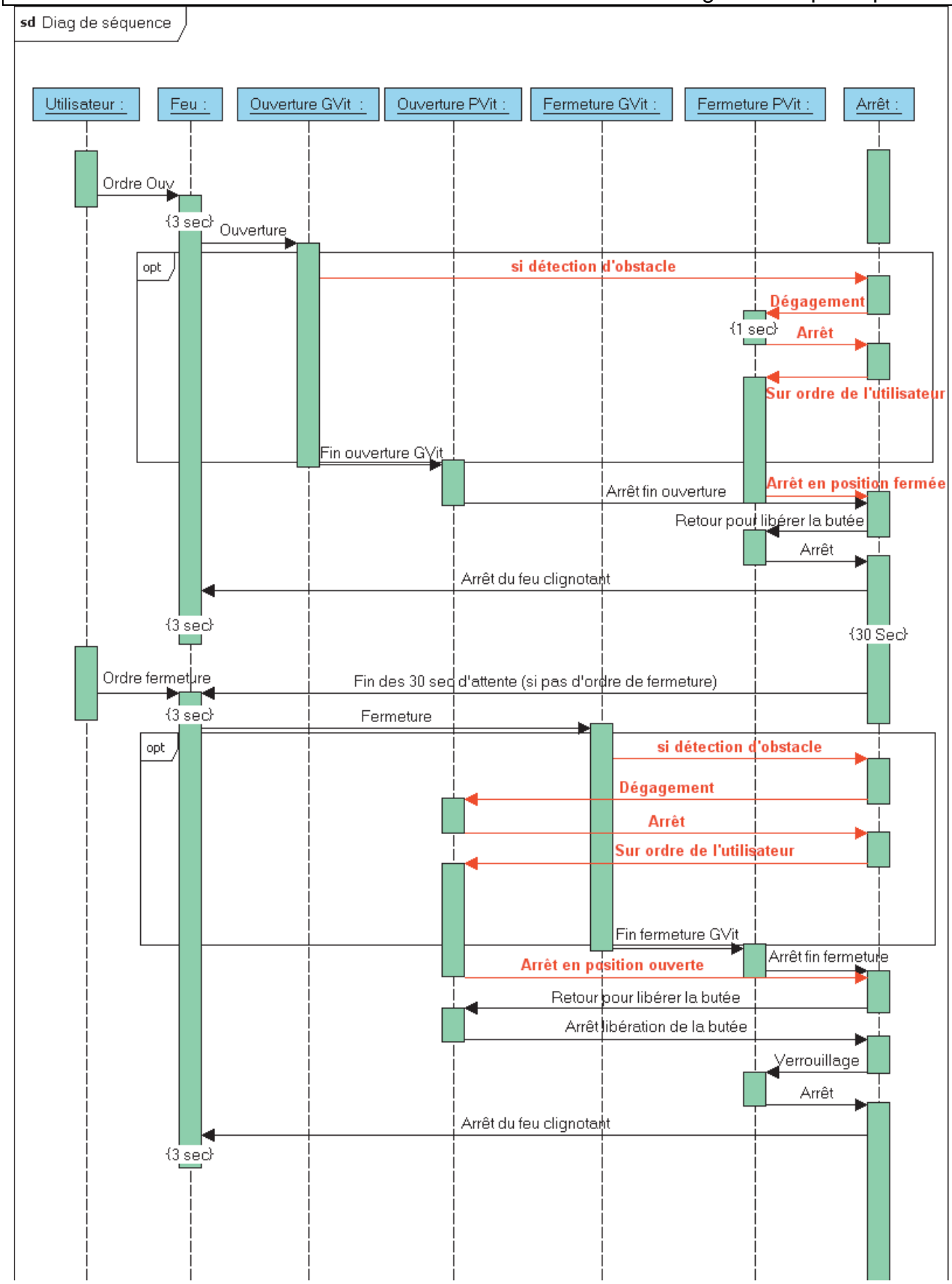
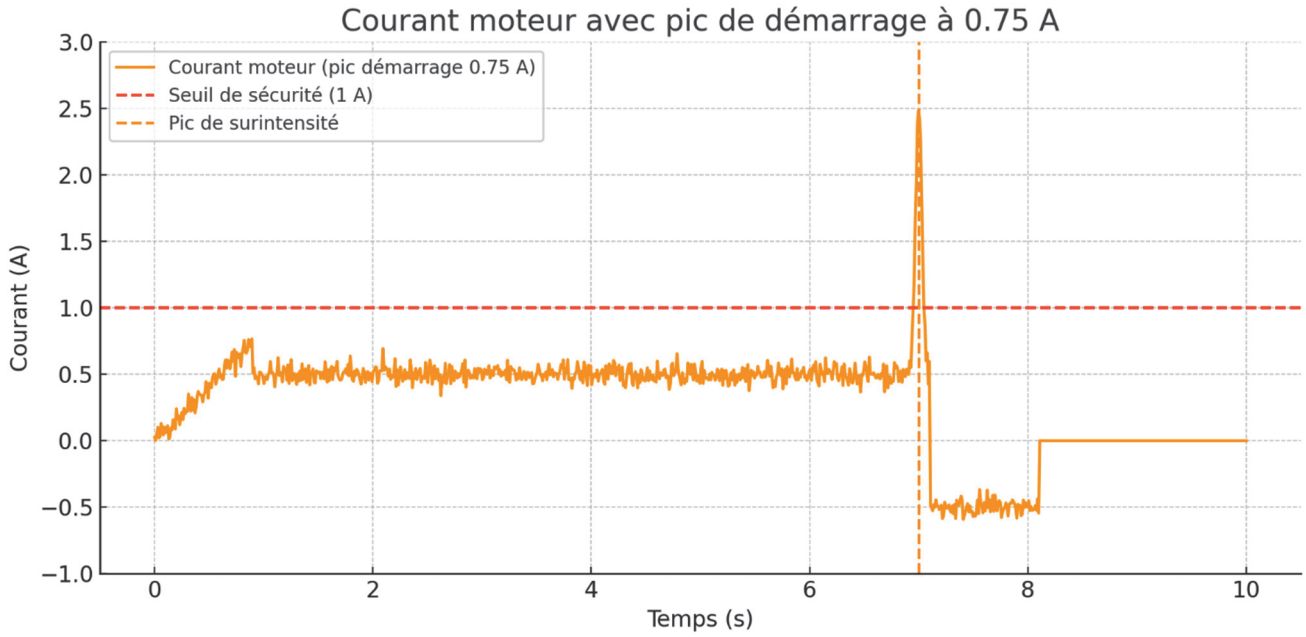


Figure 3 : diagramme de séquence

d. Détection d'un obstacle

Un obstacle est détecté par un pic de surintensité engendré par le moteur.



e. Extrait de programme

```
//Définition des constantes
#define pButtonOuvertureWeb 3
#define pButtonFermetureWeb 2
#define pButtonButeeOuverture 10
#define pButtonButeeFermeture 11
// e1 du L293 correspond à la broche 5 arduino
#define e1 5
// e2 du L293 correspond à la broche 6 arduino
#define e2 6
#define pinFeuClignotant 4
// détecteur d'obstacle
/*
 * A compléter
 */
// vitesse d'ouverture du portail
#define pVit 2
#define gVit 20

// pour la simulation on considère que le portail s'ouvre en dix secondes
#define dureeOuvertureFermeturePortail 10000

// Simulation des boutons de la page Web
int buttonOuvertureWebState = 0;
int buttonFermetureWebState = 0;
//Simulation des butées d'ouverture et de fermeture
int buttonButeeOuvertureState = 0;
int buttonButeeFermetureState = 0;
// configuration du feu clignotant
unsigned long timerFeuClignotant =0;
```

```
int delayFeuClignotant = 500;
// Etat en cours du Feu clignotant
bool etatFeuClignotant = LOW;
bool etatLedFeuClignotant = LOW;
//temps d'ouverture du portail
unsigned long timerOuvertureFermeturePortail = 0;

bool cycleOuverture = false;
bool cycleFermeture = false;

void setup()
{
  pinMode(pButtonOuvertureWeb, INPUT_PULLUP);
  pinMode(pButtonFermetureWeb, INPUT_PULLUP);
  pinMode(pButtonButeeOuverture, INPUT_PULLUP);
  pinMode(pButtonButeeFermeture, INPUT_PULLUP);
  pinMode(LED_BUILTIN, OUTPUT);
    /*
   * A compléter
   */
  pinMode(e1, OUTPUT);
  pinMode(e2, OUTPUT);
  pinMode(pinFeuClignotant, OUTPUT);
  timerFeuClignotant= millis();
  Serial.begin(9600);
}

void loop()
{
  buttonOuvertureWebState = !digitalRead(pButtonOuvertureWeb);
  buttonFermetureWebState = !digitalRead(pButtonFermetureWeb);
  buttonButeeOuvertureState = !digitalRead(pButtonButeeOuverture);
  buttonButeeFermetureState = !digitalRead(pButtonButeeFermeture);
  // Si l'un des boutons de l'interface web est appuyé on lance le cycle
correspondant.
  if (buttonOuvertureWebState == HIGH) {
    startCycleOuverture();
  }
  if (buttonFermetureWebState == HIGH){
    startCycleFermeture();
  }
}
void startCycleOuverture(){…}
void startCycleFermeture(){…}
void feuClignotantCycleInit(){…}
void gestionFeuClignotant(bool etat){…}
void feuClignotant(){…}
void Ouverture(int vit){…}
void Fermeture(int vit){…}
void Arret(){…}
// Simulation de la détection d'un obstacle
bool detectionObstacle(){
  bool obstacle= false;
  /*
   * A compléter
   */

  return obstacle;
}
```

2. Conception

Cartes programmables à disposition :

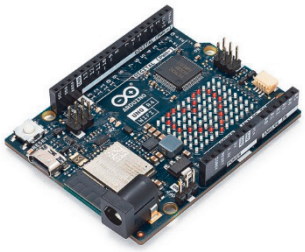
Carte Arduino UNO R3

Caractéristiques :

- Alimentation :
  - via port USB ou
  - 7 à 12 V sur connecteur alim 5,5 x 2,1 mm
- Microcontrôleur : ATMega328
- Mémoire flash : 32 kB
- Mémoire SRAM : 2 kB
- Mémoire EEPROM : 1 kB
- Interfaces :
  - 14 broches d'E/S dont 6 PWM
  - 6 entrées analogiques 10 bits
  - Bus série, I2C et SPI
- Intensité par E/S : 40 mA
- Cadencement : 16 MHz
- Gestion des interruptions
- Connecteur USB B
- Version : Rev. 3
- Dimensions : 74 x 53 x 15 mm

Carte Arduino UNO R4 Wifi

Caractéristiques :

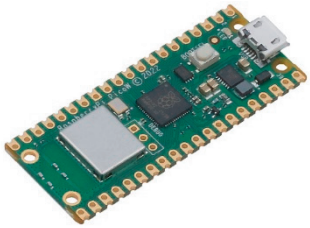


- Alimentation :
  - via le port USB Type-C
  - 6 à 24 V sur connecteur alim 5,5 x 2,1 mm
  - 6 à 24 V sur broche Vin
- Microcontrôleur : Renesas RA4M1 32 bits
- Microprocesseur : ARM Cortex-M4
- Fréquence : 48 MHz
- Mémoire Flash : 256 kB
- Mémoire RAM : 32 kB
- Circuit WiFi : ESP32-S3-Mini
- Interfaces :
  - 14 x broches d'E/S dont 6 PWM
  - 6 x entrées analogiques 10 bits
  - 1 x sortie analogique 12 bits (via un DAC : Digital-to-Analog Converter)
  - 1 x CAN (nécessite un transceiver externe)
  - 1 x bus I2C
  - 1 x liaison série UART
  - 1 x interface SPI
- Intensité par E/S : 8 mA

- Interface I2C 3,3 V sur connecteur Modulino / Qwiic
- Connecteur SWD de débogage
- Connecteur ICSP
- Module RTC
- Broche OFF
- Version : Rev. 4
- Dimensions : 68,85 x 53,34 mm

Carte Raspberry Pico H

Caractéristiques :



- Alimentation:
  - 5 Vcc via micro USB (cordon non inclus)
  - 5 Vcc via la broche VBUS
- Microcontrôleur: RP2040
- Microprocesseur: ARM Cortex-M0+ Dual Core à 133 MHz
- Mémoire SRAM: 264 KB
- Mémoire Flash: 2 MB
- 26 broches GPIO comprenant:
  - 23 x E/S digitales
  - 3 x entrées analogiques (via ADC 12 bit)
  - 2 x interfaces UART
  - 2 x bus I2C
  - 16 x sorties PWM
  - 8 x broches PIO (programmable I/O)
  - 1 x interface SWD de debug
- Interface hôte et périphérique de stockage USB 1.1 via le port micro-USB
- LED programmable sur GP25
- Température de service: -20 à 85 °C
- Dimensions: 51 x 21 x 3,9 mm
- Poids: 3 g

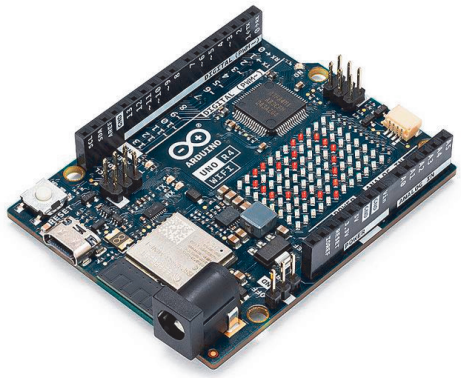
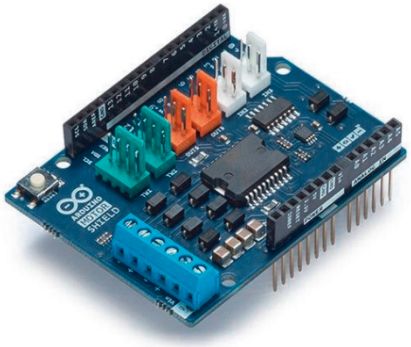
3. Simulation

Sous TinkerCAD l'accès se fait via le lien : **Simulation TinkerCAD**

4. Expérimentation

Pour effectuer l'expérimentation, le matériel suivant est à disposition :

- Une carte de développement Arduino ;
- Un shield Moteur ;
- Un moteur CC 12 V ;
- Un potentiomètre ;
- Une Alimentation ;
- Une LED ;
- Deux capteurs de fin de course ;
- Une platine de prototypage ;
- Un smartphone.

 Carte Arduino UNO R4 Wi-Fi	 Arduino Motor Shield 2 x 2 A A000079
Moteur CC Motoréducteur rapport, 1024:1 / 6-24V / Axe D4 mm (918D1024112/1) 	Potentiomètre 
Alimentation 0-30V	Capteurs de fin de course

Utiliser le programme, à compléter, donné dans le dossier ressource pour réaliser le serveur WEB.

Branchement du moteur sur le shield :