Robotique

Patrick Hainaut 2026

But de cette présentation

- Après avoir vu des notions d'électronique, de microcontrôleurs et d'impression 3D, intéressons-nous au contrôle des moteurs en robotique

Introduction

- Généralement, les robots que l'on construira comporteront un ou plusieurs moteurs

- Nous allons voir ici les différents types de moteurs que l'ont peut utiliser dans une robotique de loisir

- Nous n'aborderons pas la motorisation industrielle (ponts roulants, bras robots dans les chaînes de montage, contrôle de vannes, …)

1. LES DIFFÉRENTS MOTEURS

Types de moteurs

- Déjà, il faut différencier les moteurs DC (courant continu) des moteurs AC (courant alternatif)

- Les moteurs AC et les servomoteurs AC seront utilisés dans des robots industriels puissants et robustes tels que des ponts roulants

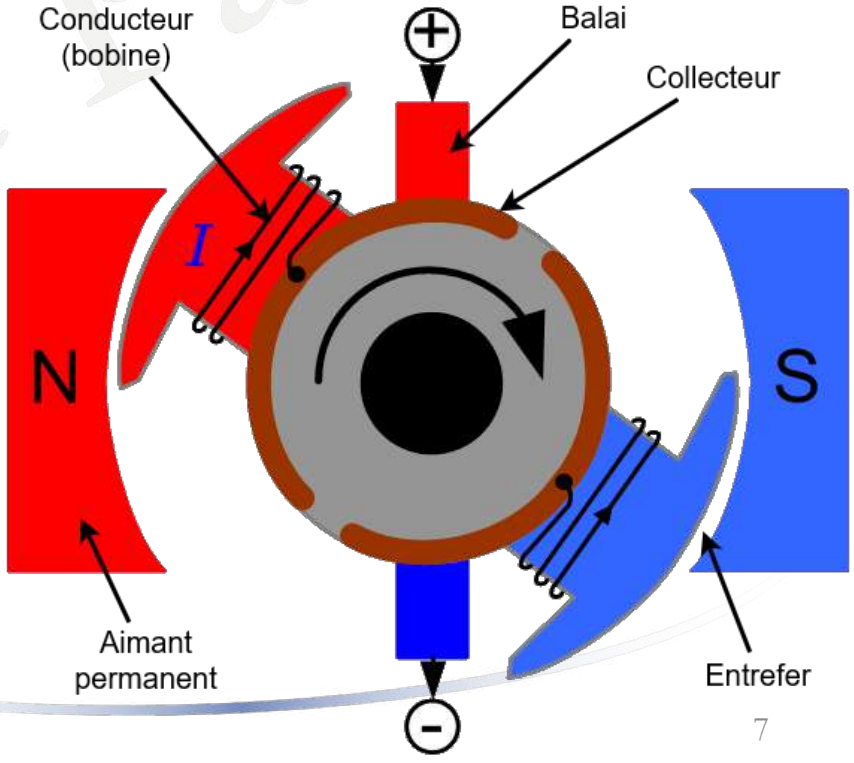
- Le reste de notre propos concerne les moteurs DC

Types de moteurs

- On va trouver:
  - les moteurs DC à balais (brushed motors): simples à mettre en œuvre et peu coûteux
  - Les moteurs DC sans balais (brushless motors): assez simples à mettre en œuvre, avec une plus grande durée de vie mais plus chers
  - Les motoréducteurs: leur vitesse de rotation est calibrée
  - Les servomoteurs: ils permettent de se déplacer un certain angle et sont utilisés dans les bras articulés
  - Les moteurs pas à pas (stepper motors): avancent d'un incrément précis et sont utilisés dans la robotique de précision, l'impression 3D par exemple

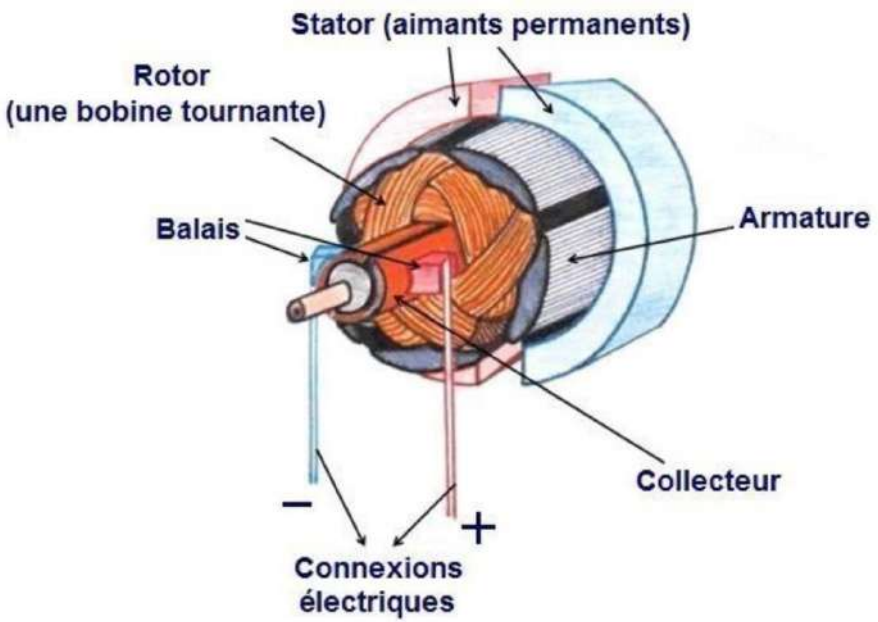
Principe de fonctionnement d'un moteur DC

- Dans un moteur DC, le stator (partie fixe) crée un champ magnétique à l'aide d'aimants permanents (ou de bobines)
- Le rotor (partie tournante) est constitué de bobines alimentées par un collecteur
- Le champ magnétique va être inversé périodiquement dans le rotor pour maintenir le rotor en rotation (les pôles de même orientation se repoussent)



Différences entre moteurs avec et sans balais

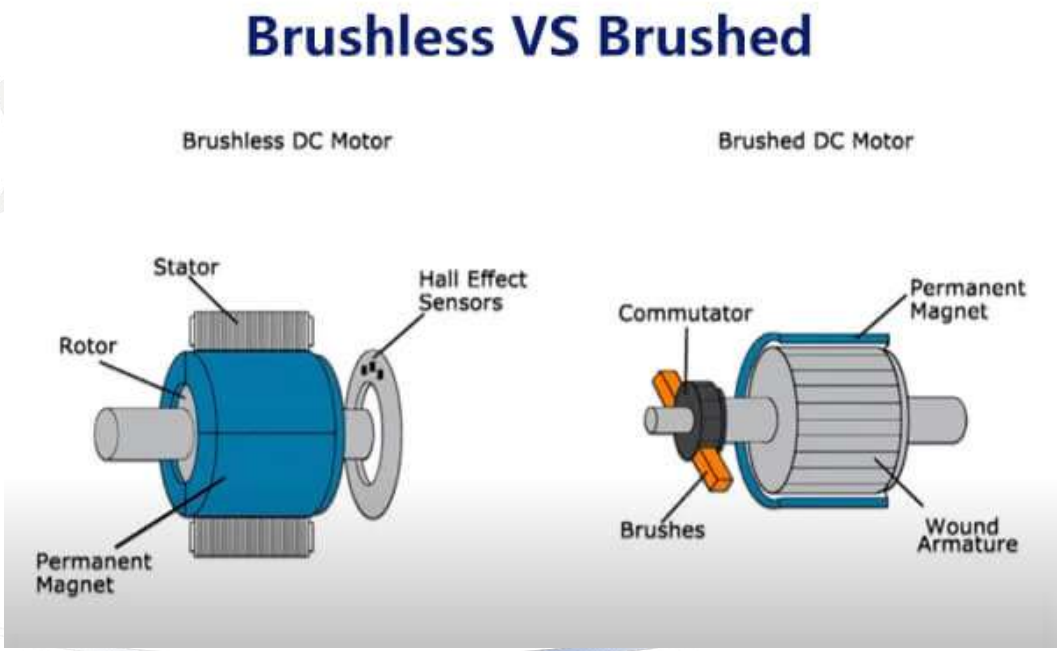
- Dans un moteur à balais, la liaison entre le collecteur et l'alimentation se fait avec des balais de graphite qui s'usent
- Cela génère du frottement et de la chaleur



Différences entre moteurs avec et sans balais

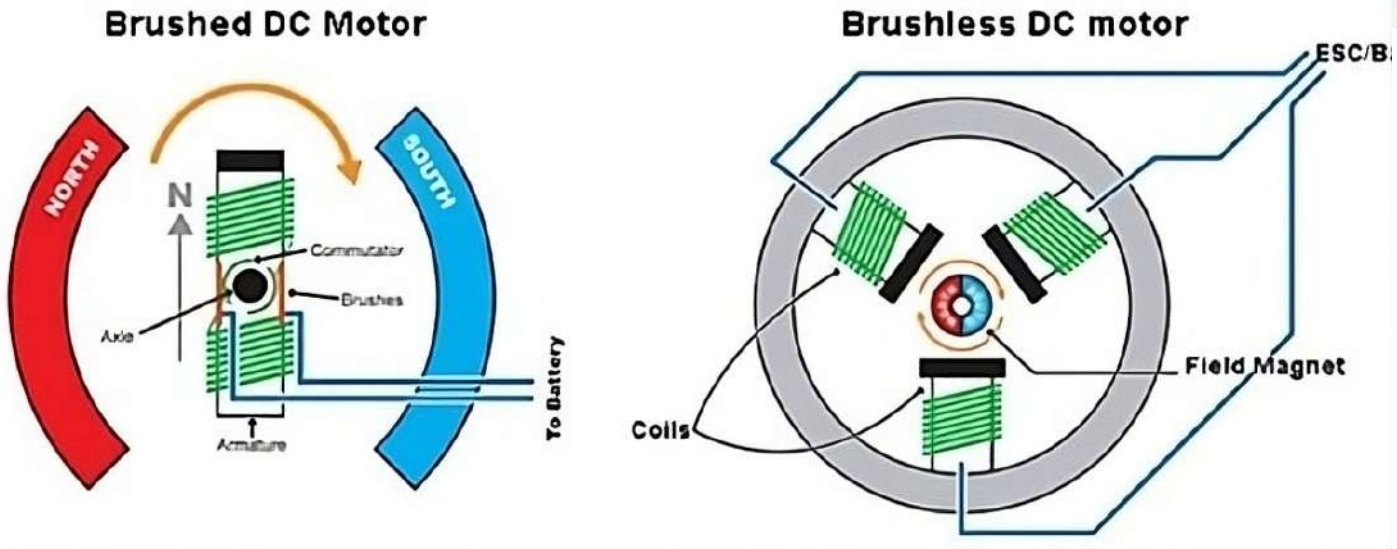
- Dans un moteur sans balais, ce sont des capteurs (à effet Hall) qui renseignent de la position du rotor et un contrôleur associé inverse la polarité au bon moment pour maintenir la rotation

- L'avantage, c'est qu'il n'y a pas de frottement et pas d'usure, l'inconvénient, c'est qu'il faut un contrôleur



Différences entre moteurs avec et sans balais

- Dans un moteur sans balais, les bobines sont au stator et les aimants permanents au rotor

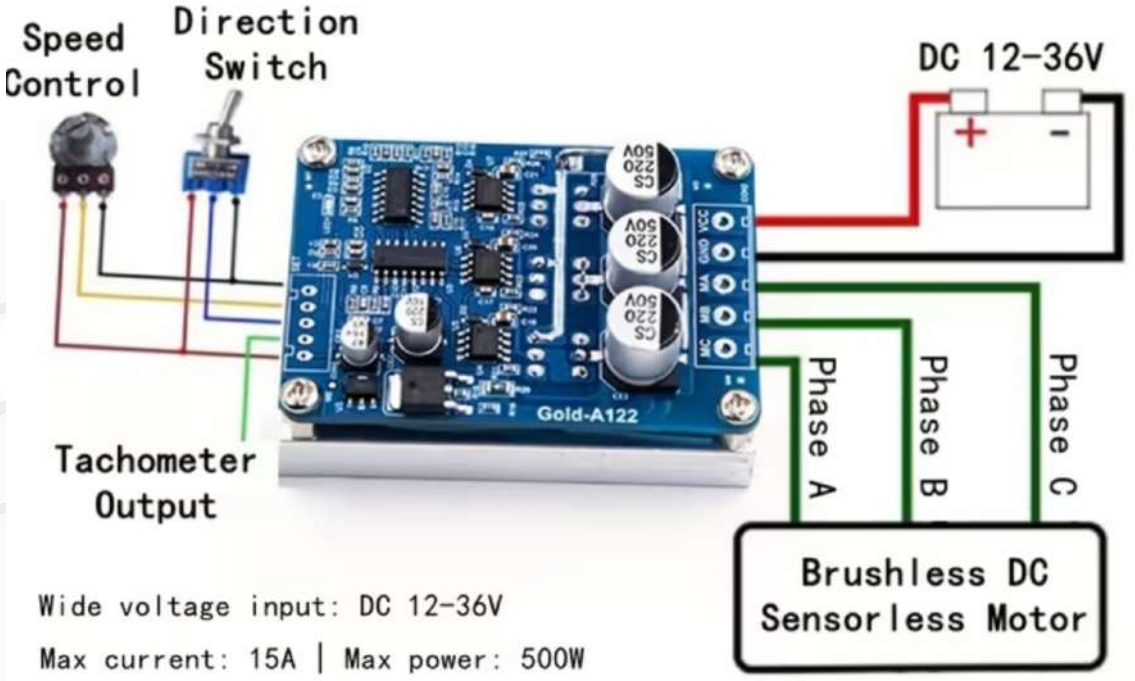


- Le moteur sans balais est en fait un moteur AC synchrone triphasé, c'est pour ça qu'il a trois fils d'alimentation, un par phase

Différences entre moteurs avec et sans balais

- Il est cependant alimenté en continu à travers son contrôleur nommé ESC en anglais (Electronic Speed Controler)

- En inversant 2 fils de phase, on inverse le sens de rotation



Grandeurs caractéristiques d'un moteur

- **Sa tension d'alimentation** en volts: si on alimente le moteur avec une tension plus faible que sa tension nominale (dans une certaine mesure), il tourne moins vite

- **Le courant qu'il consomme** en ampères: on peut en déduire la puissance du moteur et la puissance de l'alimentation nécessaire (P=UxI)

Grandeurs caractéristiques d'un moteur

- **Son couple (torque)** en Newton-mètres (Nm): c'est sa force de rotation (ce qu'il peut entrainer), proportionnelle à sa puissance et inversément proportionnelle à sa vitesse

- Pour les petits moteurs, le couple est souvent exprimé en kg/cm (parfois noté kg.cm) qui représente le nombre de kilos que le moteur peut entrainer à 1 cm de l'arbre moteur (à partir de l'axe)

- Un moteur de 3kg.cm pourra soulever 3kg à 1 cm de l'axe de l'arbre moteur et donc 0,3kg à 10 cm de cet axe

Grandeurs caractéristiques d'un moteur

- **Sa vitesse de rotation** en tours par minute (tr/min) -> rpm en anglais

- Pour un moteur brushless, la vitesse est souvent exprimée en KV (symbole sans unités)
- 1KV = 1 tr/min par volt

- Un moteur de 4500 KV, alimenté en 5V aura donc une vitesse de rotation de 4500 x 5 = 22500 tr/min
- C'est une vitesse théorique à vide ne tenant pas compte des frottements et de la charge du moteur

Moteurs DC à balais (Brushed motors)

- Disponibles en plusieurs vitesses, tensions et puissances
- La vitesse de rotation est généralement assez élevée
- Par exemple, les moteurs utilisés dans les trottinettes électriques disponibles en 12 et 24V sous différentes puissances
- Ils ont juste deux fils d'alimentation (+ et -)

Moteurs DC à balais (Brushed motors)

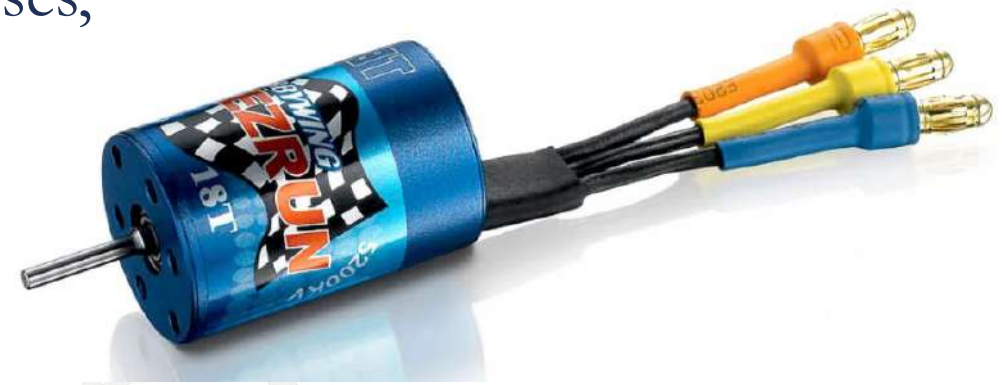
- Exemples:
  - les moteurs utilisés pour faire avancer un réplica de R2D2; MY6812: 24V – 150W – 2750 tr/min (rpm) – 0,56 Nm, environ 50€ sur Amazon

  - Un petit moteur 385: 12V – 10.000 tr/min, environ10€ pour 2 sur Amazon

  - Un petit moteur RC-300: 3-6V – 8500 tr/min, environ 5€ sur Amazon

Moteurs DC sans balais (Brushless Motors)

- Disponibles en plusieurs vitesses, tensions et puissances
- La vitesse de rotation est généralement assez élevée



- Ils ont 3 fils de connexion et se connectent obligatoirement à un contrôleur (ESC) qui est vendu à part

- Utilisé en radiomodélisme (voiture, drone, …)

Moteurs DC sans balais (Brushless Motors)

- Exemple: un moteur brushless D3542:
  7,4-15V – 1450KV, environ 30€
  sur Amazon

7,4V	**10.730 tr/min**
15V	**21.750 tr/min**

- Exemple d'ESC:
  7,4-11,1V lipo battery,
  60A maximum (environ
  30€ sur Amazon)

Moteurs DC sans balais (Brushless Motors)

- Les nouveaux moteurs de scooter électrique sont des moteurs brushless puissants alimentés sous 24V à 48V et sont parfaits pour réaliser des robots roulants de grande taille, tel qu'un R2D2

- Ils sont directement intégrés dans la roue du scooter

- On utilise également un ESC pour les contrôler

Motoréducteurs (Gear motors)

- Ce sont des moteurs DC auxquels sont associés une boite à engrenage pour diminuer la vitesse et augmenter le couple (la force du moteur)
- Ils sont très utilisés en robotique

Motoréducteurs (Gear motors)

- Exemples:
  - petit moteur à engrenages TT:
    3 – 6V, 200 tr/min – 0,15-0,6 Nm,
    quelques euros sur Amazon

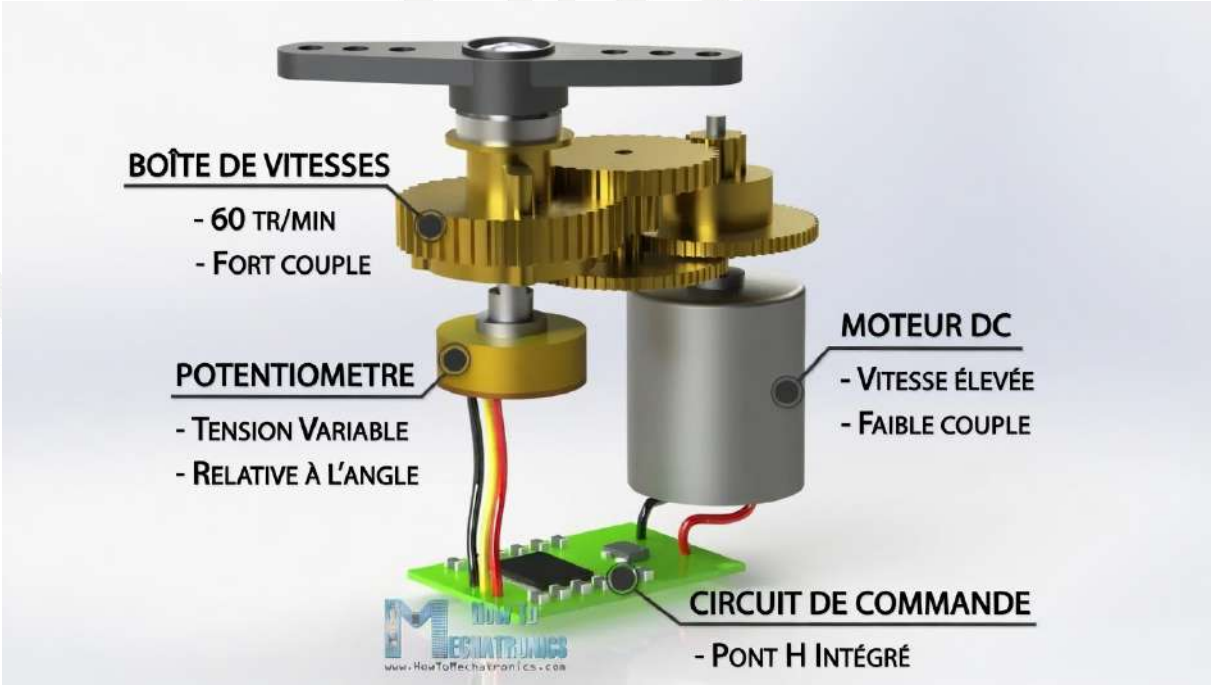
  - Moteur à engrenages 37-3530
    12-24V – 18tr/min (à 12V) – 1-1,6 kg.cm,
    environ 20€ sur Amazon

Servomoteurs (Servo motors)

- Le servomoteur contient un moteur DC + un système de contrôle de position qui va permettre au servo de rejoindre une position et de la maintenir

- Ils sont très utilisés en modélisme

  Ils ont deux fils d'alimentation et un fil signal PWM (voir section suivante)



Servomoteurs

- Exemples:
  - SG90 9g: 4,8 – 6V, 0 – 180° d'amplitude
    couple: 1,2kg/cm
    engrenages en plastique
    quelques euros sur Amazon

  - MG996R: 4,8 – 7,2V, 0 – 120° d'amplitude
    couple: 9,4kg/cm
    engrenages en métal
    environ 15€ sur Amazon

Moteurs pas à pas

- Le moteur pas à pas permet de transformer une impulsion électrique en un mouvement angulaire

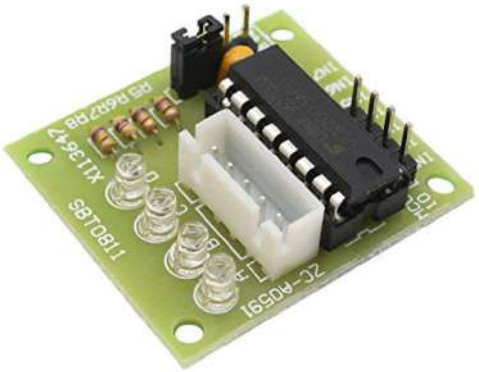
- Il fait un tour complet en un nombre de pas bien déterminé

- Le nombre de pas parcouru par le moteur est très précis, c'est pour ça qu'on les utilise dans toutes les applications de positionnement: imprimantes 3D, fraiseuses numériques (CNC), graveuses/découpeuses laser, pousse-seringue, …

Moteurs pas à pas

- Le moteur pas à pas nécessite un contrôleur (driver) spécifique qui dépend de la puissance et du type de moteur

- Exemples:
  - Carte ULN2003 pour les moteurs unipolaires 28BYJ-48 (quelques euros sur Amazon)



  - Driver DM542 pour moteurs bipolaires de 18 à 48V, comme la série NEMA (plus ou moins 20€ sur Amazon)

Moteurs pas à pas

- Le moteur pas à pas bipolaire nécessite 4 lignes de commande mais permet un positionnement plus précis, ce qui explique son utilisation dans les imprimantes 3D et autres CNC

- Le moteur pas à pas unipolaire ne nécessite que 2 lignes de commande, on le retrouve souvent dans des petits projets Arduino

- Les contrôleurs pour moteurs pas à pas unipolaire et bipolaire sont différents

Moteurs pas à pas

- Quelques exemples:
  - Petit moteur pas à pas unipolaire
    28BYJ-48: 5V – 4 phases – 300gf.cm,
    64 pas/tr en interne, 2048 pas/tr en externe,
    environ 8€ sur Amazon avec son contrôleur

  - Moteur bipolaire NEMA17 qu'on retrouve fréquemment
    dans les imprimantes 3D: 12V – 4 phases – 3,2kg.cm,
    200 pas/tr, environ 16€ sur Amazon pour ce modèle
    (des modèles avec moins de couple sont
    également en vente, ils sont moins longs)

2. CONTRÔLE DES MOTEURS

Contrôle des moteurs DC

- Si on alimente un moteur DC (ou un motoréducteur), il tourne
- Si on augmente la tension d'alimentation (sans dépasser la tension nomimale), il tourne plus vite
- Si on diminue la tension (sans aller jusqu'à la tension de décrochage), il tourne moins vite
- Si on inverse la polarité d'alimentation, le moteur tourne dans l'autre sens
- On contrôle souvent les trains électriques avec un simple variateur

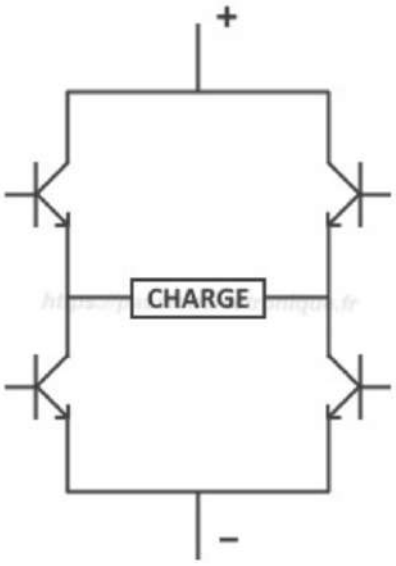
Contrôle des moteurs DC

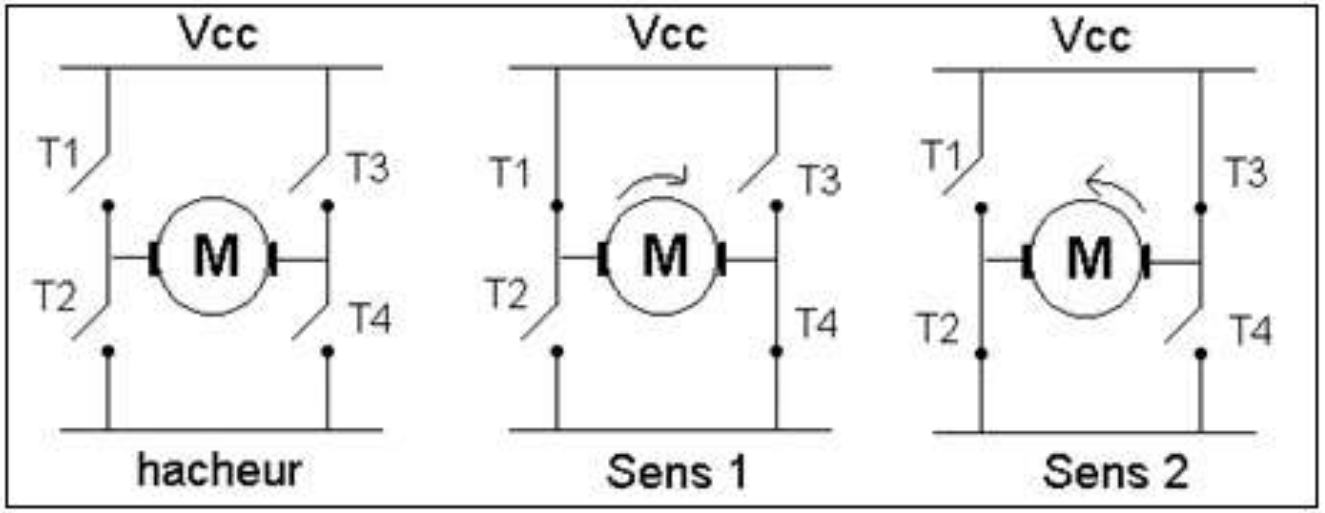
- Cependant, dans un projet de robotique, la tension d'alimentation sera fixe et on n'inversera pas les fils d'alimentation

- On utilisera donc (aussi) un contrôleur moteur composé d'une partie puissance où l'on connectera le moteur et d'une partie commande qu'on reliera à une radiocommande ou à un microcontrôleur

Contrôle des moteurs DC

- On utilise pour cela un pont en H
- Suivant les transistors activés, le moteur tournera dans un sens ou dans l'autre





- Pour varier la vitesse, on alimentera le pont en PWM

Contrôle des moteurs DC: PWM

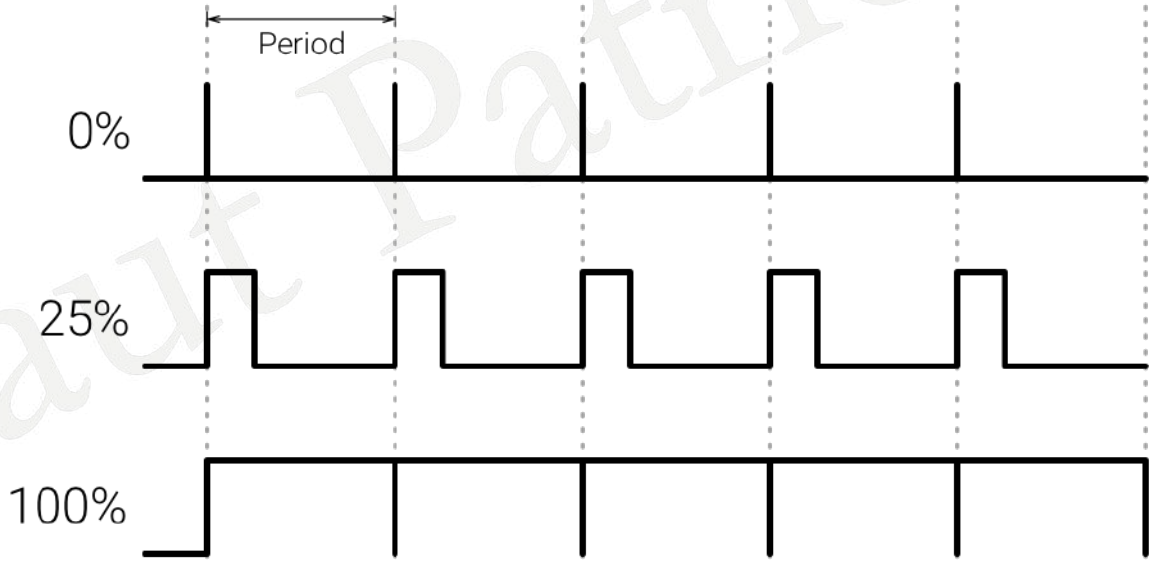
- La PWM (Pulse Width Modulation), modulation de largeur d'impulsion en français, permet de générer un signal pseudo-analogique à partir de signaux numériques tout ou rien

- On n'a toujours que deux signaux possibles:
  - 1 (Vmax) -> le moteur tourne à plein régime
  - 0 (0V) -> le moteur s'arrête

- Mais on va jouer sur le facteur temps: sur une période, on va alimenter le moteur une fraction du temps, le reste du temps, il ne sera pas alimenté -> il ne va pas s'arrêter d'un coup mais va décélérer -> sa vitesse moyenne sera plus faible

Contrôle des moteurs DC: PWM

- On va jouer sur le rapport cyclique (duty cicle en anglais)
- A 0%, la tension de sortie est continue et vaut 0V, le moteur est à l'arrêt



- A 100%, la tension de sortie est continue et au maximum, le moteur tourne à plein régime
- Entre les deux, elle est cyclique, et plus le rapport cyclique est faible, plus la tension moyenne est faible et plus le moteur tourne lentement

Contrôle des moteurs DC: PWM

- La plupart des microcontrôleurs permettent de générer des signaux PWM

- Par exemple, sur un arduino UNO, Les broches permettant la PWM sont les broches 3, 5, 6, 9, 10, 11

- La fréquence (et donc la période) du rapport cyclique est préréglé:
  - 490 Hz sur les broches 3, 9, 10 et 11
  - 980 Hz sur les broches 5 et 6

Contrôle des moteurs DC: PWM

- Pour alimenter un moteur en PWM avec un arduino, on va utiliser l'instruction:

  analogWrite(*pin*,*value***);**
  - pin: numéro de la broche
  - value: une valeur entre 0 et 255 où
    - 0 représente un rapport cyclique de 0%
    - 255 un rapport cyclique de 100%

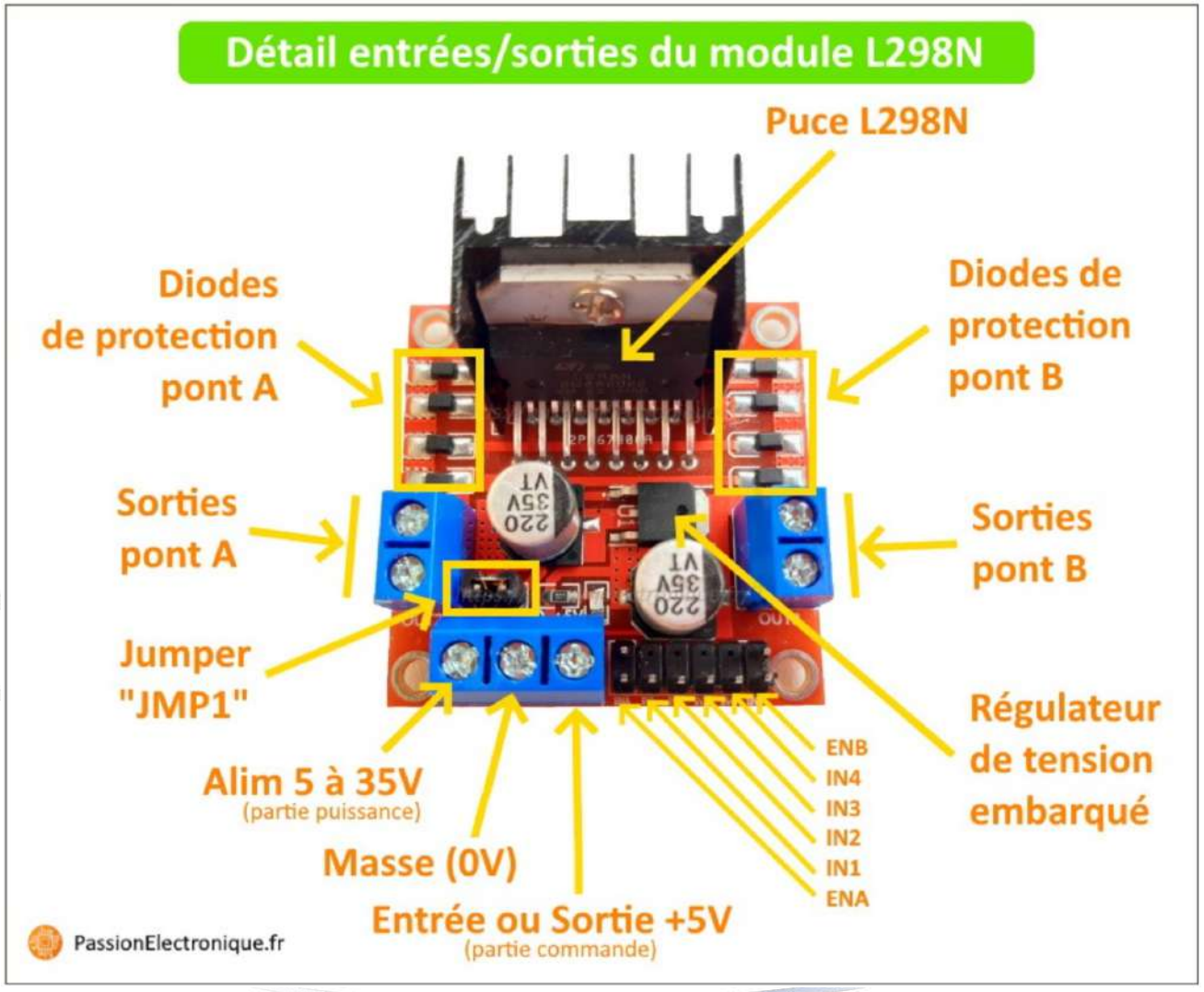
  Si on veut un rapport cyclique de 60%, il faut utiliser une règle de trois pour trouver la valeur à mettre -> 60/100*255

  Exemple: **analogWrite(**10,153**);**

Contrôle des moteurs DC: L298N

- Ce module peut piloter
  2 moteurs DC indépendamment (2A par moteur)
  ou 1 seul moteur (4A pour ce moteur)

- Prix: 5€ sur Amazon



Contrôle des moteurs DC: L298N

- Remarques:
  - Si JMP1 est placé, le +5V de commande sera généré à partir de l'alimentation de puissance, **uniquement** si elle est comprise entre 7 et 12 V
    La broche +5V est **dans ce cas** une **sortie** (pouvant alimenter l'arduino)
  - Si l'alimentation des moteurs est supérieure à 12V, JMP1 doit être enlevé et la broche +5V devient une entrée qu'il faudra alimenter (à partir d'un arduino par exemple)
  - Il y a aussi 2 jumpers placés par défaut sur les entrées ENA et ENB qui mettent ces entrées à l'état haut et donc permettent un fonctionnement des moteurs à plein régime
  - Si on veut faire varier la vitesse des moteurs (PWM), il faut donc enlever ces jumpers

Contrôle des moteurs DC: L298N

- Broches de commandes:

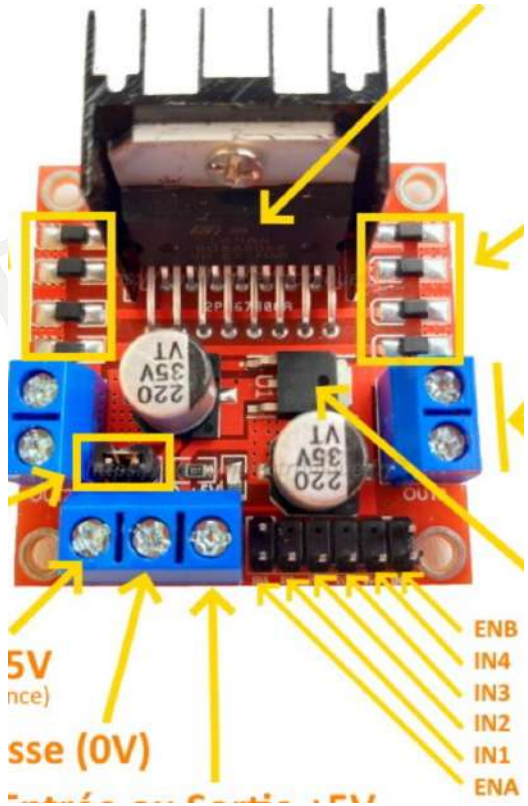
Entrées de sélection

→ 1er pont (A)
→ 2ème pont (B)

ENA	IN1	IN2	Résultat, en sortie
ENB	IN3	IN4	Résultat, en sortie
L	X	X	Moteur en roue libre (à l'arrêt, sans frein)
H	L	L	Blocage du moteur (arrêt rapide, freinage fort)
	L	H	Marche arrière
	H	L	Marche avant
	H	H	Blocage du moteur (arrêt rapide, freinage fort)

X = peu importe
L = état bas (0V, par exemple)
H = état haut (+5V, par exemple)

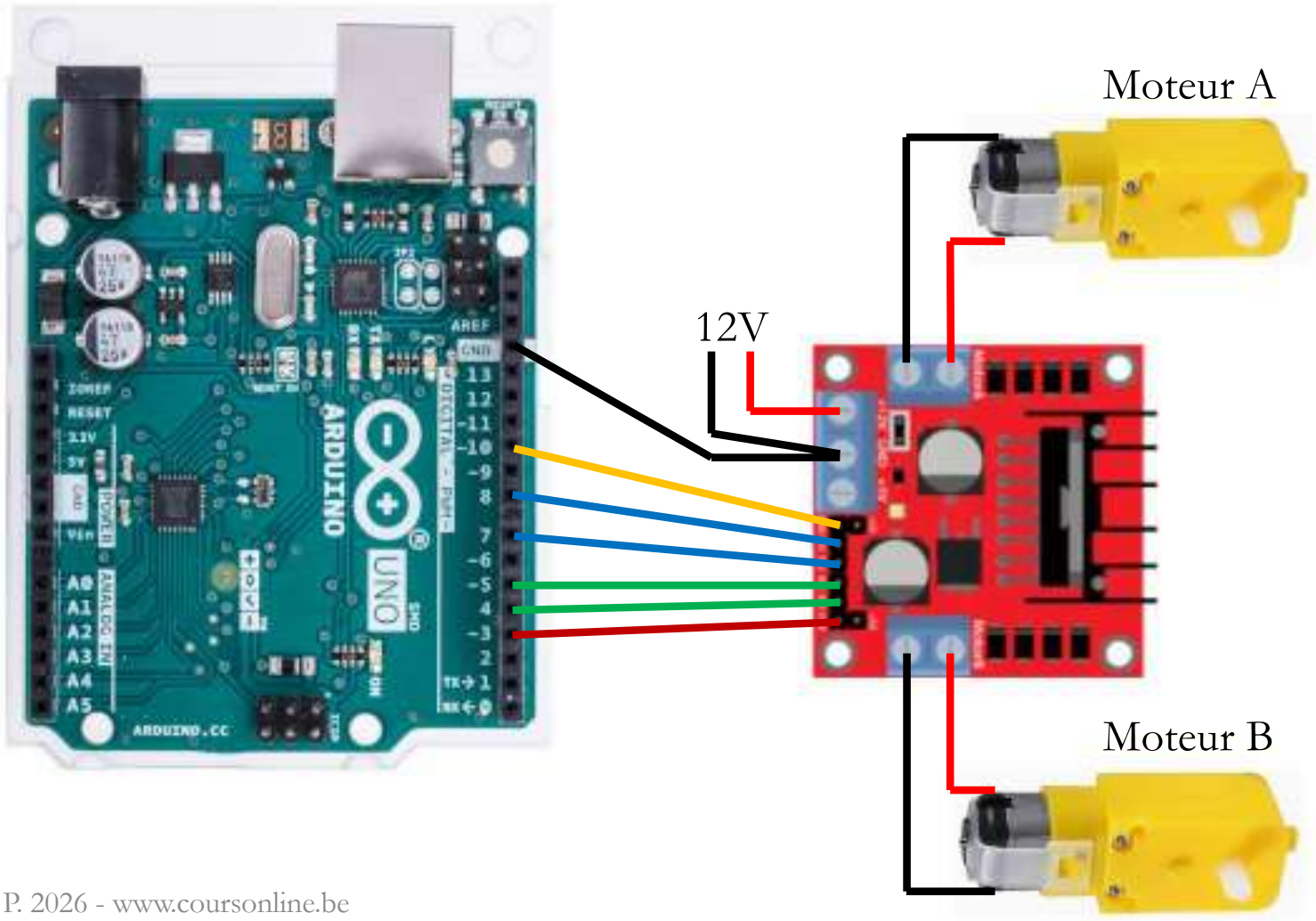
PassionElectronique.fr



Pour faire varier la vitesse du moteur, on alimentera les broches ENA et/ou ENB en PWM via un Arduino

Contrôle des moteurs DC: L298N

- Schéma complet



Contrôle des moteurs DC: L298N

- Dans ce schéma:
  - On utilise les broches 3 et 10 de l'arduino pour contrôler les entrées ENB et ENA

    Ce sont des broches PWM d'une fréquence de 490Hz (ce qui n'est pas critique dans cette application)

  - Les broches 4 et 5 contrôlent les entrées IN4 et IN3 du moteur B
  - Les broches 7 et 8 contrôlent les entrées IN2 et IN1 du moteur A
  - Une alimentation extérieure de 12V alimente les moteurs
  - On n'utilise pas la sortie +5V, l'arduino ayant sa propre alimentation, par contre, les masses doivent être communes !

Contrôle des moteurs DC: L298N

- Au niveau du programme arduino, on pourra faire tourner le moteur A à plein régime dans un sens par:

```
digitalWrite(8, HIGH); //configuration du sens de marche par IN1
digitalWrite(7, LOW); //en combinaison avec IN2
digitalWrite(10,HIGH); //activation du moteur par ENA
```

- Dans l'autre sens par:

```
digitalWrite(8, LOW);
digitalWrite(7, HIGH);
digitalWrite(10,HIGH);
```

- Les arrêter par:

```
digitalWrite(10,LOW); //désactivation du moteur
```

Contrôle des moteurs DC: L298N

- Pour faire varier la vitesse, on remplace digitalWrite par analogWrite au niveau de ENA et ENB et avec une valeur de vitesse comprise entre 0 et 255

  0: moteur arrêté
  255: moteur à plein régime

  Exemple:

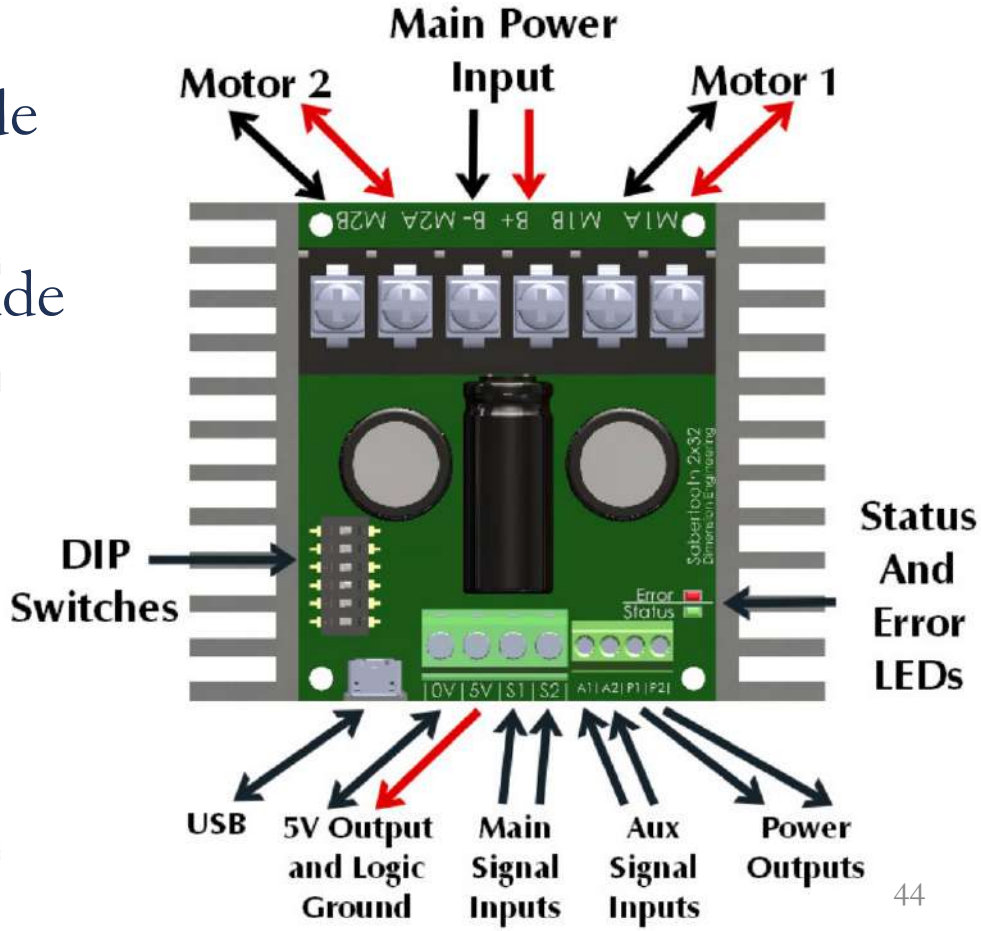
```
digitalWrite(8, HIGH); //configuration du sens de marche par IN1
digitalWrite(7, LOW); //en combinaison avec IN2
analogWrite(10, 60); // valeur PWM comprise entre 0 et 255
```

Contrôle des moteurs DC: Sabertooth

- Si on veut un contrôleur un peu plus costaud, on peut choisir un contrôleur de la série Sabertooth, par exemple, le 2x32
  - Peut piloter 2 moteurs indépendamment
  - 32A par moteur
  - 6 à 33,6V pour la partie puissance
  - Multiples modes de commande
    - Analog (joystick)
    - Radio control (radiocommande)
    - Serial (microcontrolleur)
    - USB (idem microcontrolleur)
    - User mode (un mix au choix)

Contrôle des moteurs DC: Sabertooth

- 0V: broche de masse qui doit être commune avec toutes les masses du dispositif de commande
- +5V: c'est une sortie !
- S1 et S2: broches de commande (modes Analog, R/C, Serial)
- A1 et A2: broches de commande auxiliaires
- P1 et P2: sorties utilisées dans des modes particuliers
- C'est la configuration du DIP switch qui déterminera le mode



Contrôle des moteurs DC: Sabertooth

- Le réglage du DIP switch et les branchements à effectuer dépendront du mode choisi

- Un guide détaillé existe et peut être téléchargé: https://www.dimensionengineering.com/datasheets/Sabertooth2x32.pdf

- Prix: 135€ chez Gotronic

- Ce prix peut sembler excessif mais par rapport au contrôleur L298N, la puissance et les possibilités sont bien plus élevés

Contrôle des moteurs DC: Sabertooth

- Exemple de programme basique (en mode serial):

```
#include <Sabertooth.h>
#include <SoftwareSerial.h>

// Utilisation de la pin 11 pour le TX, on connecte au S1 du Sabertooth
SoftwareSerial SWSerial(NOT_A_PIN, 11);
Sabertooth ST(128, SWSerial); // 128 est l'adresse par défaut

void setup() {
  SWSerial.begin(9600);
  ST.begin(9600);
}

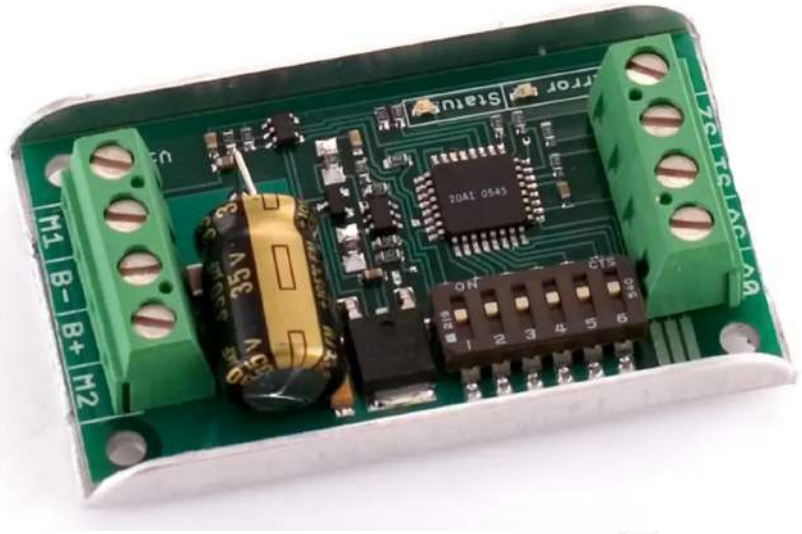
void loop() {
  ST.motor(1, 50); // Moteur 1 à 50% de puissance
  ST.motor(2, -50); // Moteur 2 à 50% de puissance inverse
  delay(2000);
  ST.motor(1, 0); // Arrêt
  ST.motor(2, 0);
  delay(2000);
}
```

- Ce programme utilise deux bibliothèques à importer

Contrôle des moteurs DC

- D'autres drivers moteurs existent, de toutes les puissances et de tous les prix, en fonction des besoins

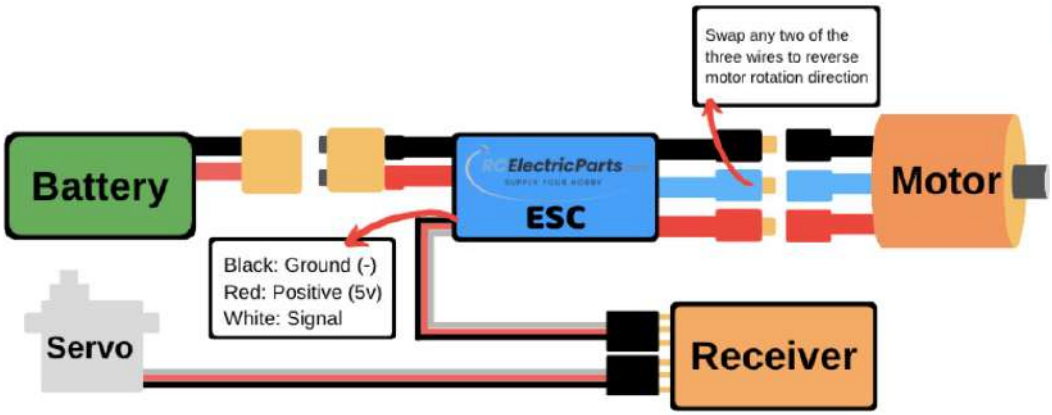
- Comme le syren 10, par exemple (environ 50€)

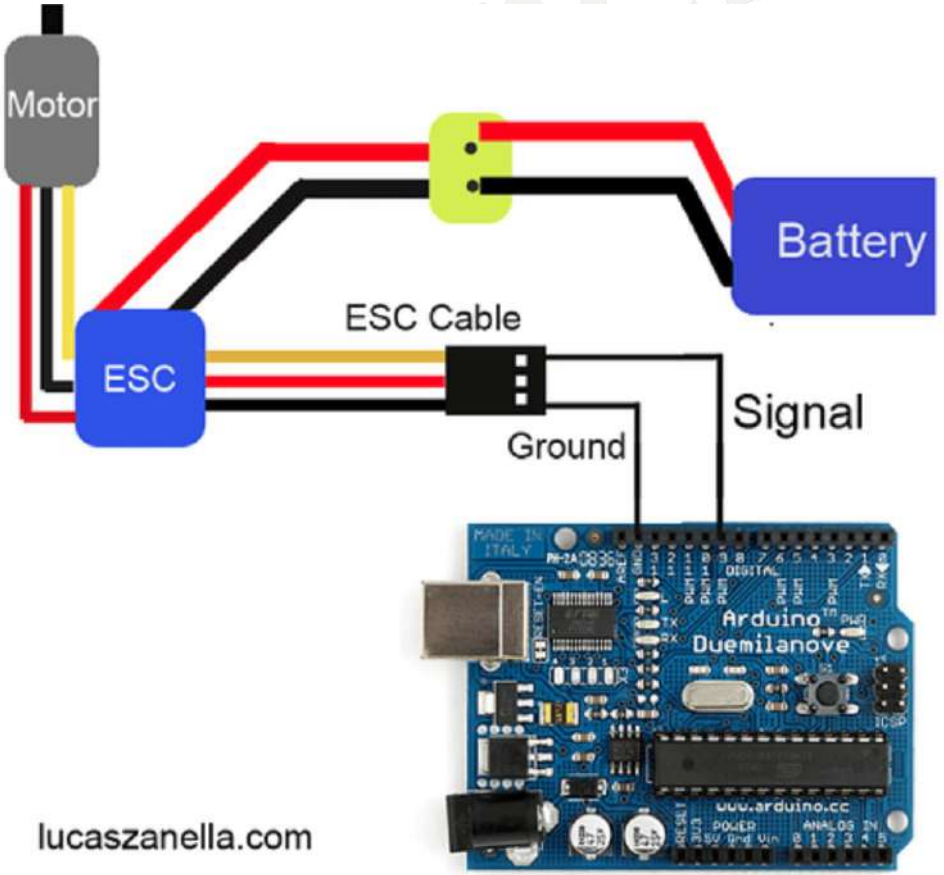


Contrôle des moteurs DC sans balais

- Les ESC sont prévus pour être connectés à un récepteur de radiocommande mais on peut aussi les connecter à un Arduino

- On connectera le fil signal à une sortie PWM





Contrôle des moteurs DC sans balais

- ESC typique:
  - Plage de tension d'alimentation de 4 à 16V
    Convient pour les batteries 2lipo ou 3 lipo
  - Vitesse interne jusqu'à 300.000 tr/min pour 2 pôles
    Vitesse externe jusqu'à 50.000 tr/min pour 12 pôles
  - Environ 9€ sur Amazon

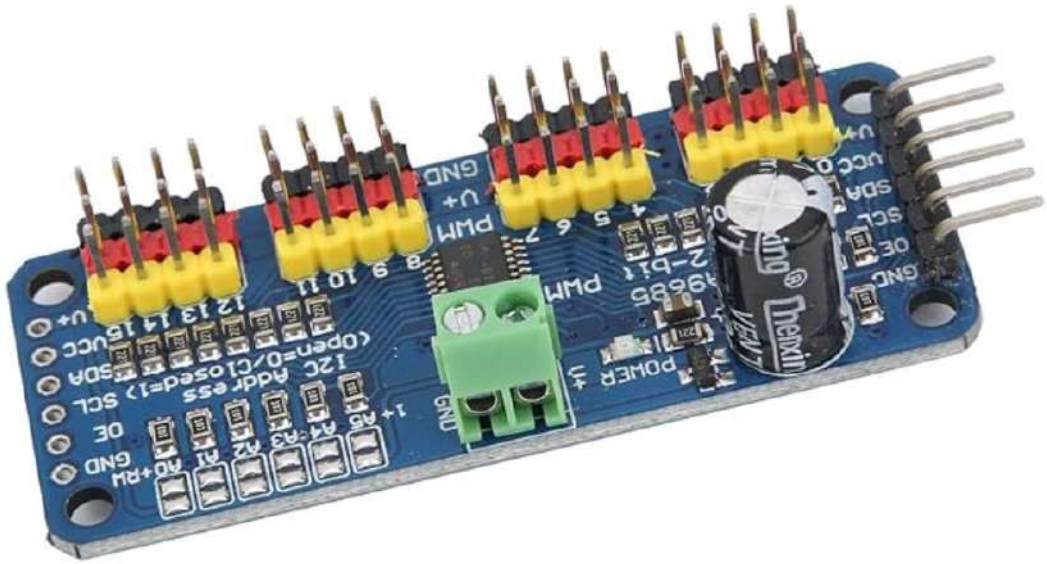
Contrôle des moteurs DC sans balais

- Exemple de programmation basique avec un potentiomètre connecté sur la broche A0 pour faire varier la vitesse:

```
#include <Servo.h>
Servo ESC; // crée une variable de type Servo
int potValue; // valeur du potentiomètre
void setup() {
// Le fil signal de l'ESC est connecté à la broche 9
ESC.attach(9,1000,2000); // (broche, largeur de pulsation min, puis max  en microsecondes)
}
void loop() {
potValue = analogRead(A0); // lit la valeur du potentiomètre (entre 0 et 1023)
potValue = map(potValue, 0, 1023, 0, 180); // le redimensionne pour l'utiliser avec la librairie Servo
ESC.write(potValue); // envoie le signal à l'ESC
}
```

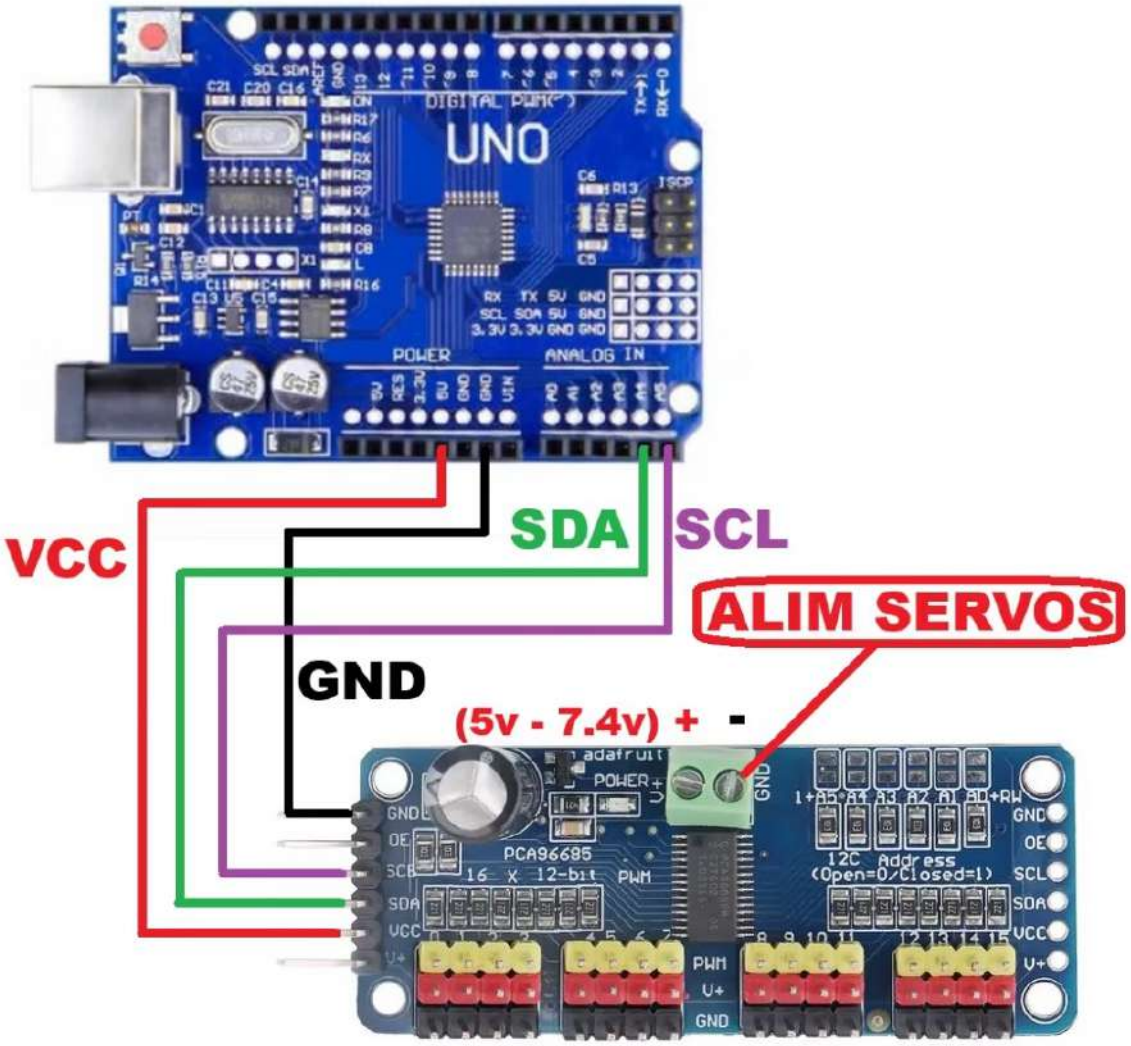
Contrôle des servomoteurs

- Ils peuvent être contrôlés directement par un Arduino
- Si on a besoin de contrôler un nombre élevé de servos, un contrôleur de servos de type PCA9685 sera très utile (environ 8€ sur Amazon)
- Il possède 16 canaux
- Il est pilotable par les lignes i²c SDA (Serial Data Line) et SCL (Serial Clock Line)



Contrôle des servomoteurs

- Schéma de câblage:



Contrôle des servomoteurs: Prg

```
#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>
Adafruit_PWMServoDriver pwm = Adafruit_PWMServoDriver( 0x40 );
// valeurs approximatives qui peuvent varier d'un servomoteur à l'autre:
#define SERVOMIN  150 // Position minimale
#define SERVOMAX  600 // Position maximale
//Création des variables Servomoteurs
uint8_t servo0 = 0; //Pour le connecteur 0 du module ou du shield PCA9685
uint8_t servo1 = 1; //Pour le connecteur 1 du module ou du shield PCA9685
int pulseLen;//Angle de positionnement du servomoteur entre 0 et 180 degrés
void setup() {
  pwm.begin(); //Initialisation de la liaison I2C
  pwm.setPWMFreq(60);//Fréquence du signal PWM
  delay(1000);
}
void loop() {
  pulseLen = map( 10, 0, 180, SERVOMIN, SERVOMAX );//Angle à 10 degrés
  pwm.setPWM(servo0, 0, pulseLen); //Positionnement du Servomoteur 0 à 10 degrés
  delay(2000);
  pulseLen = map( 123, 0, 180, SERVOMIN, SERVOMAX );//Angle à 123 degrés
  pwm.setPWM(servo1, 0, pulseLen);//Positionnement du Servomoteur 1 à 123 degrés
  delay(2000);
}
```

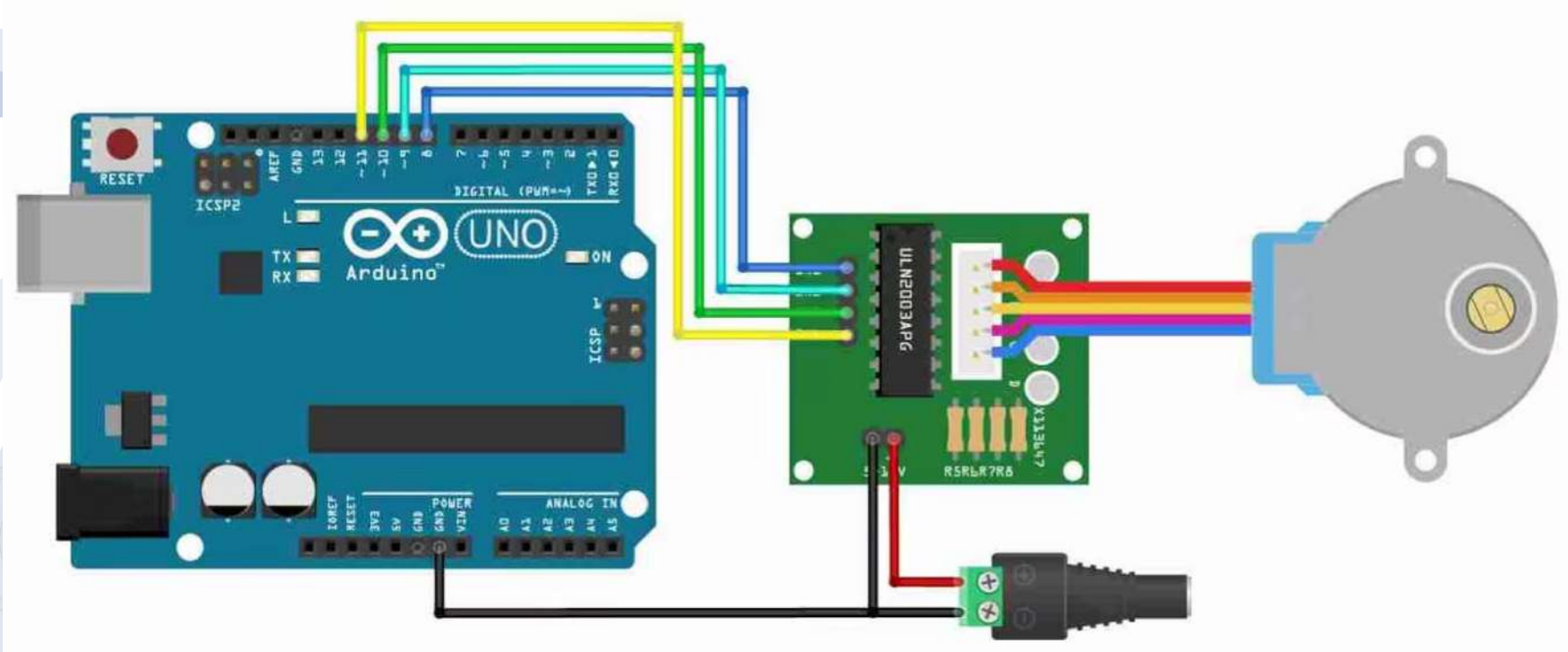
Contrôle des servomoteurs: Prg

- Autre exemple:

```
#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>
Adafruit_PWMServoDriver servoController = Adafruit_PWMServoDriver(0x40);
const uint8_t frequency = 50;    // 50Hz PWM frequency or T=20ms
const uint16_t ServoMinTicks = 102; // pulse width in ticks for 0° position
const uint16_t ServoMaxTicks = 512; // pulse width in ticks for the 180° position
void setup()
{
  servoController.begin();
  servoController.setPWMFreq(frequency );
}
void loop(){
  for (uint16_t duty = ServoMinTicks; duty < ServoMaxTicks; duty++) {
    for (uint8_t n = 0; n<16; n++) {
      servoController.setPWM(n, 0, duty);
    }
  }
  delay(1000);
  for (uint16_t duty = ServoMaxTicks; duty > ServoMinTicks; duty++) {
    for (uint8_t n = 0; n<16; n++) {
      servoController.setPWM(n, 0, duty);
    }
  }
  delay(1000);
}
```

Contrôle des moteurs pas à pas

- Choisissons, par exemple, un moteur 28BYJ-48 et un contrôleur ULN2003
- Voici un schéma de connexion possible



Contrôle des moteurs pas à pas: prg

- ```
  #include <Stepper.h>
  //définit les broches d'entrées du moteur
  #define OUTPUT1   10
  #define OUTPUT2   9
  #define OUTPUT3   8
  #define OUTPUT4   7

  // définit le nombre de pas par tour
  const int pasParTour = 2048;  // 28BYJ-48 à 2048 pas par tour en mode pas à pas complet

  // initialise le moteur pas à pas avec la séquence des broches de contrôle OUTPUT1, OUTPUT3, OUTPUT2, IN4
  // OUTPUT1 et OUTPUT3 sont connectés à une bobine et  OUTPUT2 and OUTPUT4 sont connectés à l'autre
  Stepper myStepper(pasParTour, OUTPUT1, OUTPUT3, OUTPUT2, OUTPUT4);
  void setup() {
    // définit la vitesse du moteur en RPM (à ajuster au besoin)
    myStepper.setSpeed(15);  // 15 RPM
  }

  void loop() {
    // tourne dans une direction et fait un tour en 2048 pas
    myStepper.step(pasParTour);
    delay(1000);  // délai entre les tours
    // on inverse le sens de direction en mettant un – devant le paramètre
    myStepper.step(- pasParTour);
    delay(1000);
  }
  ```
  ```

3. MICROCONTRÔLEURS

Choix du microcontrôleur

- Les exemples précédents utilisent généralement un Arduino Uno
- On sait avec le ppt struct5 qu'il n'est pas le seul modèle existant
- Les Arduino sont programmés en language C/C++ via l'IDE Arduino mais aussi, depuis récemment, pour certains modèles en micropython

- Outre les Arduino, d'autres microcontrôleurs sont intéressants dans la robotique de loisir

- En voici quelques-uns
  - raspberry Pi et Pi Pico - stm32
  - esp32 - micro:bit
  - kl25z - brique intelligente lego

Raspberry Pi 4B

- C'est un mini-ordinateur sur lequel on peut installer plusieurs OS comme Linux, l'utiliser comme centre multimédia mais aussi grâce à son GPIO, on peut en faire une console de jeu rétrogaming ou piloter des projets robotiques

- On pourra notamment le programmer en C/C++, en python et également en micropython

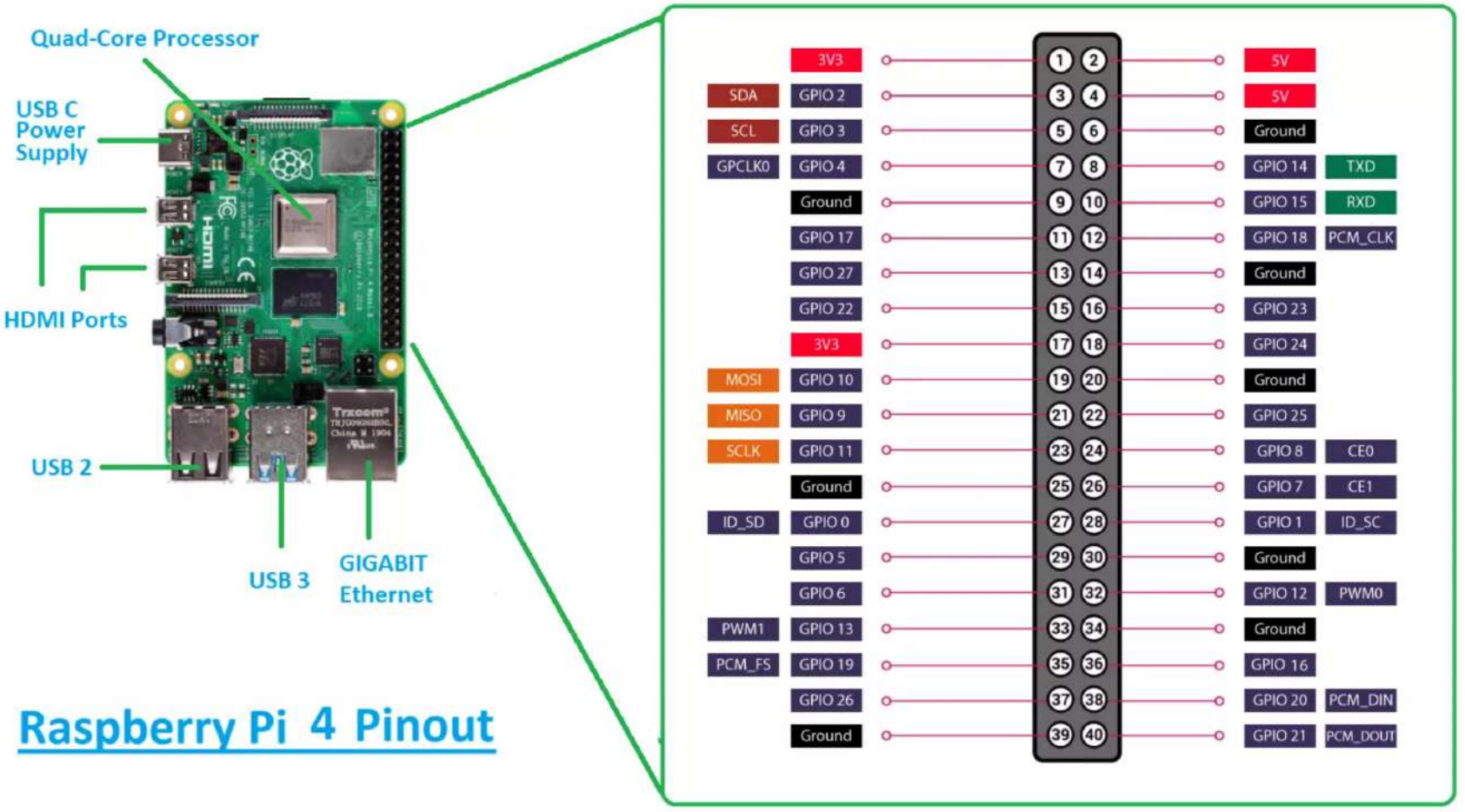
- Il existe plusieurs versions de ce mini-ordinateur, on en est à la version 5

Raspberry Pi 4B

- Caractéristiques:
  - SOC (System On Chip): Quad core Cortex-A72 (ARM v8) 64-bit
  - 1GB, 2GB, 4GB or 8GB LPDDR4-3200 SDRAM (suivant le modèle)
  - Mémoire flash sur carte microSD
  - Wifi 2.4 GHz et 5.0 GHz IEEE 802.11ac
  - Bluetooth 5.0, BLE
  - Gigabit Ethernet
  - 2 ports USB 3.0; 2 ports USB 2.0
  - GPIO 40 broches Raspberry Pi standard (entièrement compatible avec les autres versions)
  - PWM, I^2C, port série
  - 2 × ports micro-HDMI® (4kp60 supporté)
  - Prix: environ 40€ pour un 4B avec 1Gb de RAM

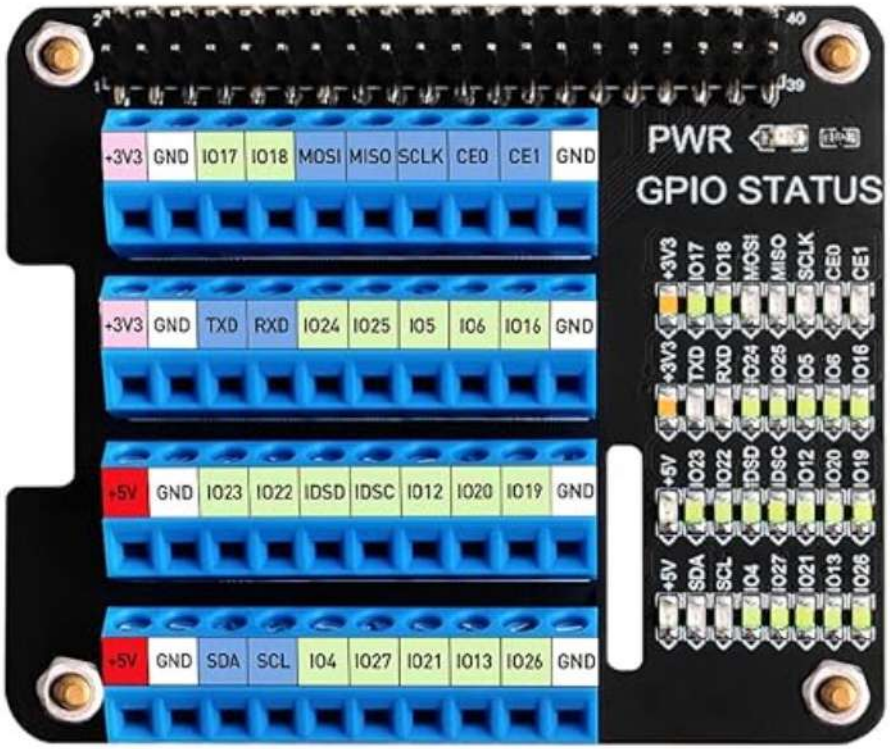
Raspberry Pi 4B

- Le brochage du GPIO reste le même pour les versions 3, 4 et 5



Raspberry Pi

- Il existe des cartes d'extension GPIO pour pouvoir connecter facilement des capteurs, des moteurs, …



Raspberry Pi

- Ce mini-ordinateur est sans aucun doute trop puissant pour servir de microcontrôleur pour un petit projet robotique

- Il servira avantageusement de centre de contrôle avec d'autres microcontrôleurs qui seront connectés à lui

- Des capteurs et actionneurs seront connectés aux microcontrôleurs satellites qui échangeront des informations avec le Raspberry Pi

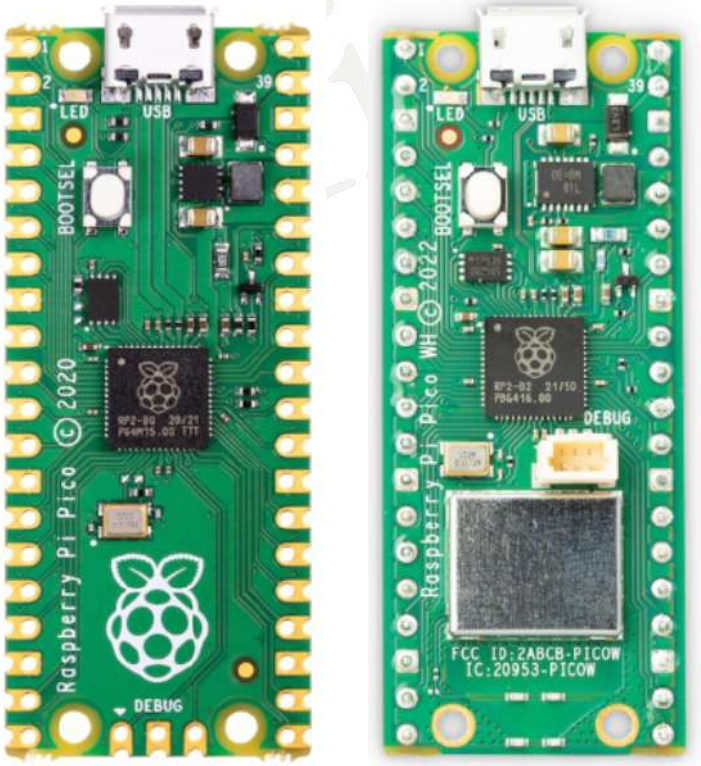
Raspberry Pi Pico et Pico 2

- Pour éviter d'utiliser un Raspberry Pi à 40€ pour piloter quelques LEDs, on peut se tourner vers le Raspberry Pi Pico à 4€
- Il existe plusieurs modèles:

	Sans fil (Wireless)	Broches soudées (Headers)
Pico	Non	Non
Pico H	Non	Oui
Pico W	Oui	Non
Pico WH	Oui	Oui
Pico 2	Non	Non
Pico 2 H	Non	Oui
Pico 2 W	Oui	Non
Pico 2 WH	Oui	Oui

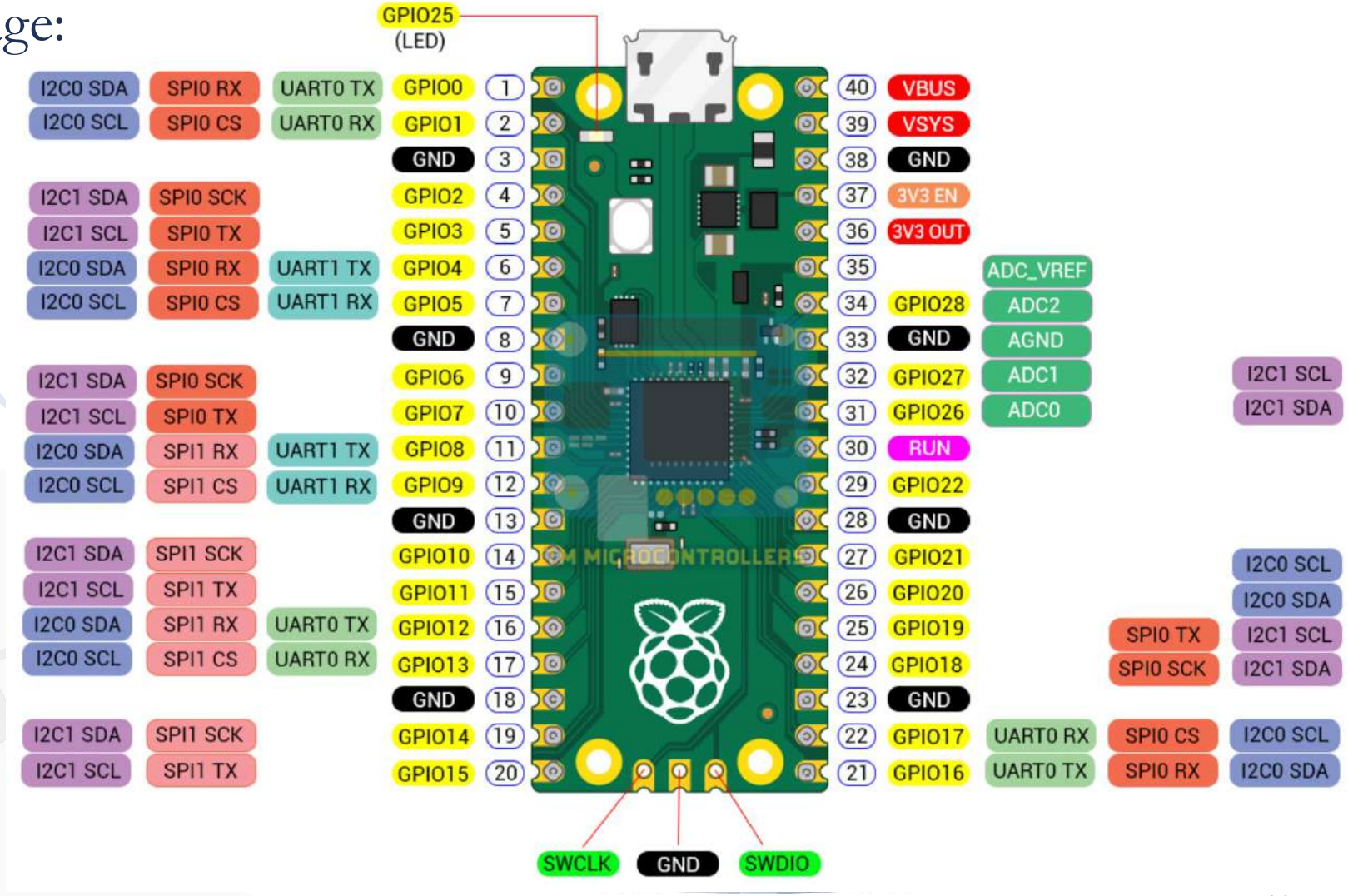
Raspberry Pi Pico

- Caractéristiques:
  - 40 broches + 3 broches de debug
  - Puce RP2040 avec processeur dual core M0+ 133 MHz
  - SRAM: 264 kb
  - Flash memory: 2 Mb
  - 26 broches I/O
  - PWM, I²C, ports série
  - Pico W(H): Wifi 802.11n 2.4 GHz
  - Pico W(H): Bluetooth v5.2
  - 3x convertisseur 12 bits analogique-digital
  - Prix: de 4 à 8 €
  - Programmation: C/C++ via le SDK natif ou via l'IDE Arduino, micropython



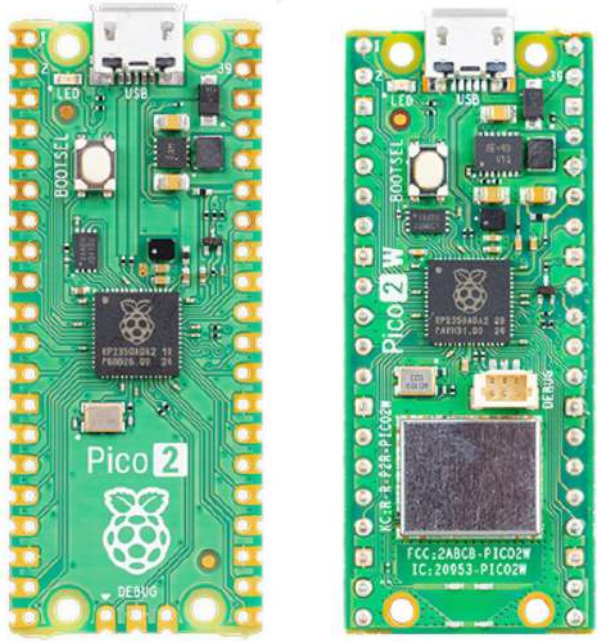
Raspberry Pi Pico

- Brochage:



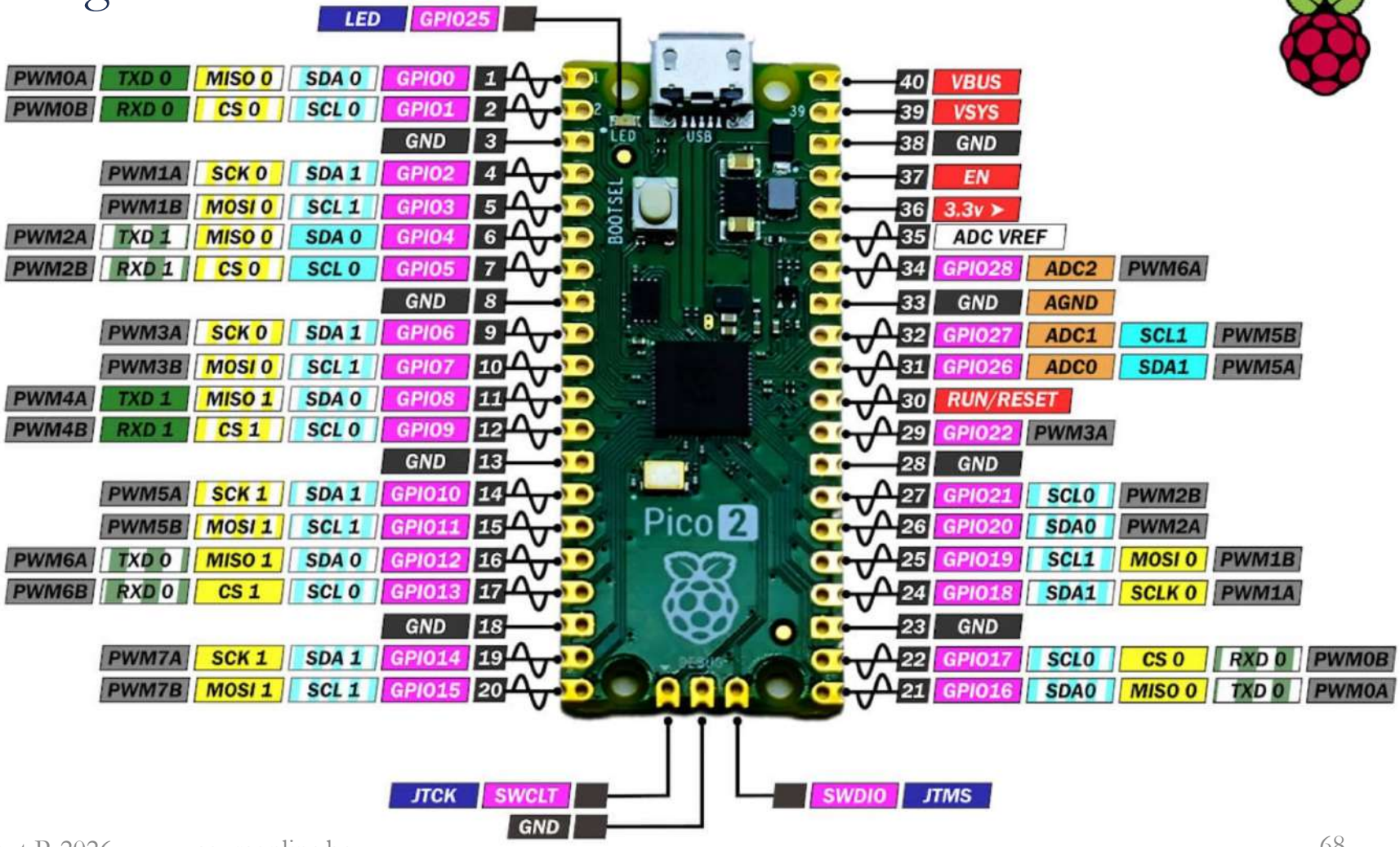
Raspberry Pi Pico 2

- Caractéristiques:
  - 40 broches + 3 broches de debug
  - Puce RP2350 avec processeur dual core Arm Cortex-M33/Hazard3 RISC-V 150 MHz
  - SRAM: 520 kb
  - Flash memory: 4 Mb
  - 26 broches I/O
  - PWM, I²C, ports série
  - Pico W(H): Wifi 802.11n 2.4 GHz
  - Pico W(H): Bluetooth v5.2
  - 3x convertisseur 12 bits analogique-digital
  - Prix: de 5 à 9 €
  - Programmation: C/C++ via le SDK natif ou via l'IDE Arduino, micropython



Raspberry Pi Pico

- Brochage:

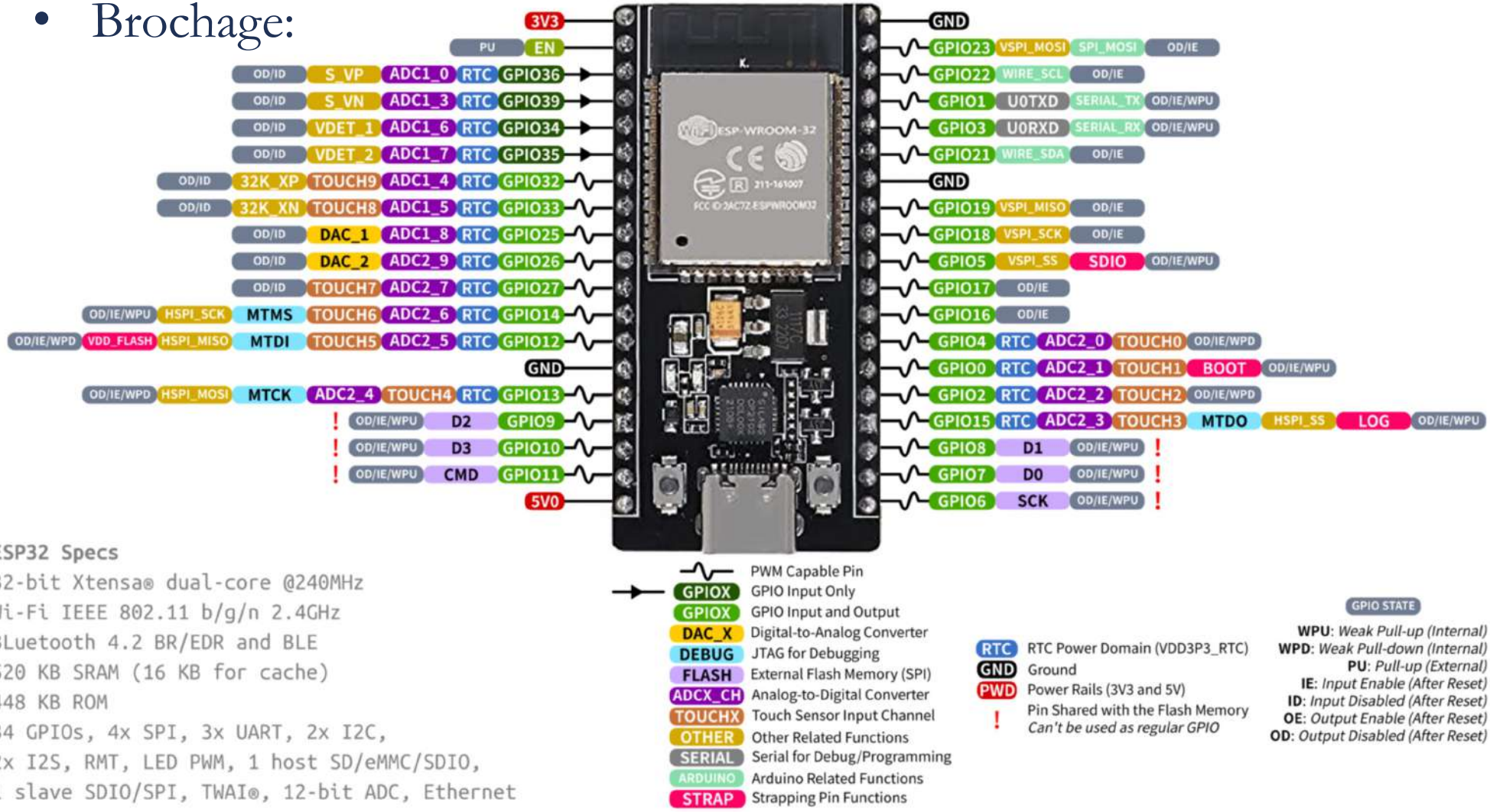


esp32

- Ce microcontrôleur présente les caractéristiques suivantes:
  - 38 broches pour la nouvelle version contre 30 pour l'ancienne
  - SOC (System On Chip): ESP32-WROOM 32 240 MHz
  - RAM: 520 kb
  - Flash memory: 4 Mb
  - 34 broches I/O
  - PWM, I^2C, port série
  - Wifi 2.4 GHz
  - Bluetooth v4.2
  - Compatible avec Arduino
  - Prix: environ 10€
  - Programmation: C/C++ via IDE Arduino et micropython

esp32

- Brochage:

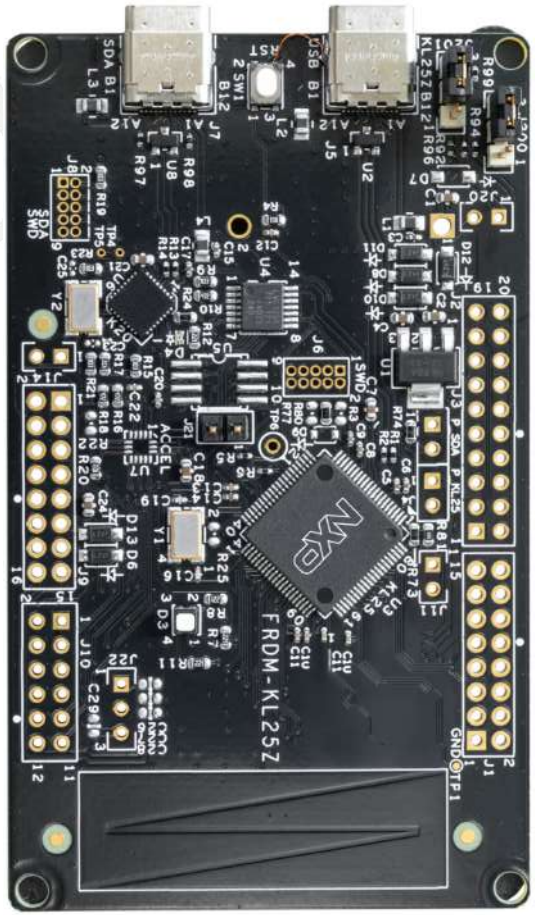


ESP32 Specs

32-bit Xtensa® dual-core @240MHz
Wi-Fi IEEE 802.11 b/g/n 2.4GHz
BLuetooth 4.2 BR/EDR and BLE
520 KB SRAM (16 KB for cache)
448 KB ROM
34 GPIOs, 4x SPI, 3x UART, 2x I2C,
2x I2S, RMT, LED PWM, 1 host SD/eMMC/SDIO,
1 slave SDIO/SPI, TWAI®, 12-bit ADC, Ethernet

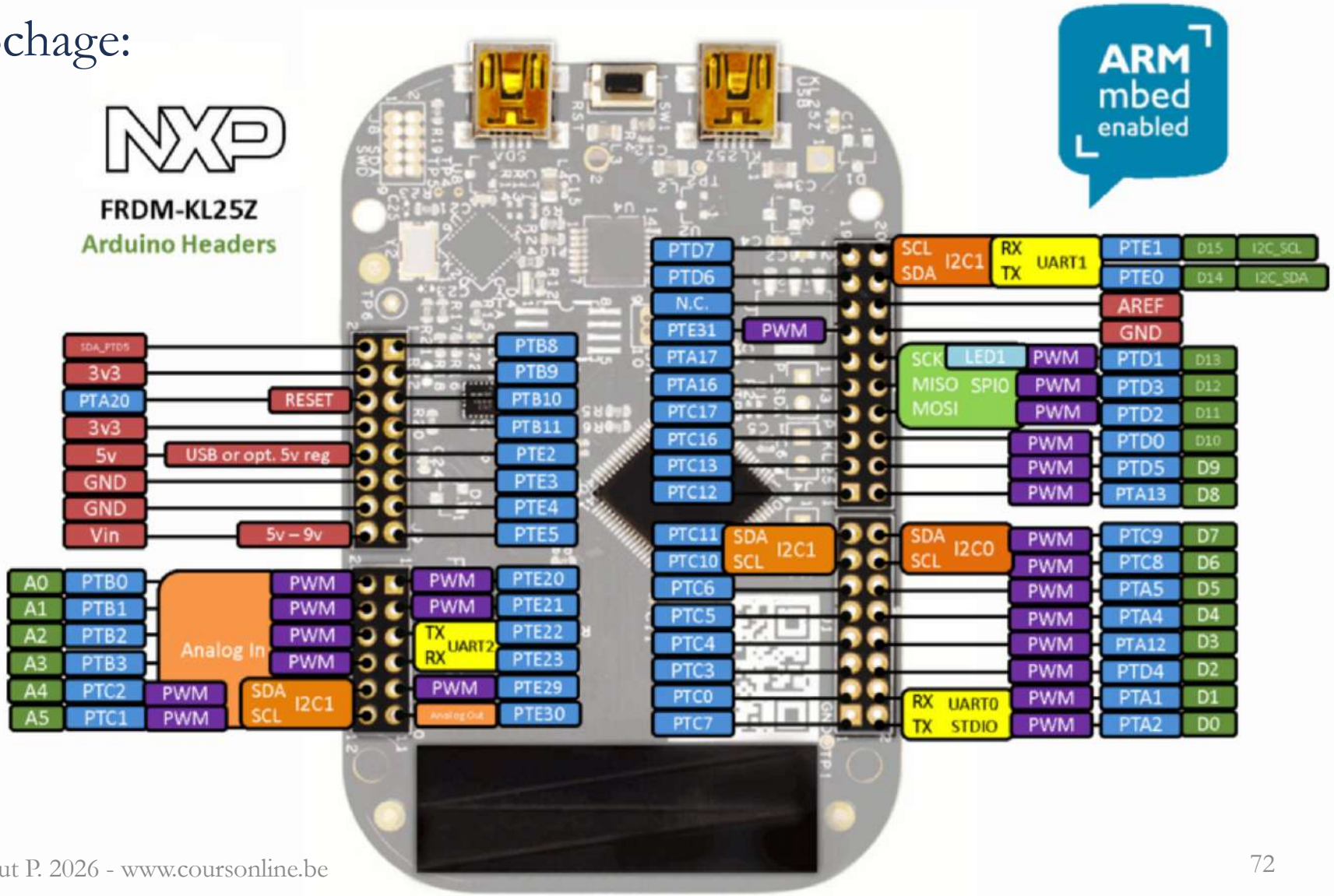
kl25z

- Ce microcontrôleur présente les caractéristiques suivantes:
  - Microcontrôleur MKL25Z128VLK4 avec processeur ARM® Cortex™-M0+ Core 48MHz
  - RAM: 16 kb
  - Flash memory: 128 kb
  - 54 broches I/O
  - PWM, I²C, port série, port USB
  - ADC 16 bits
  - DAC 12 bits
  - Touch sensor capacitif, accéléromètre
  - Compatible avec Arduino
  - Prix: environ 30€
  - Programmation: C/C++ via environnement MBED



kl25z

- Brochage:

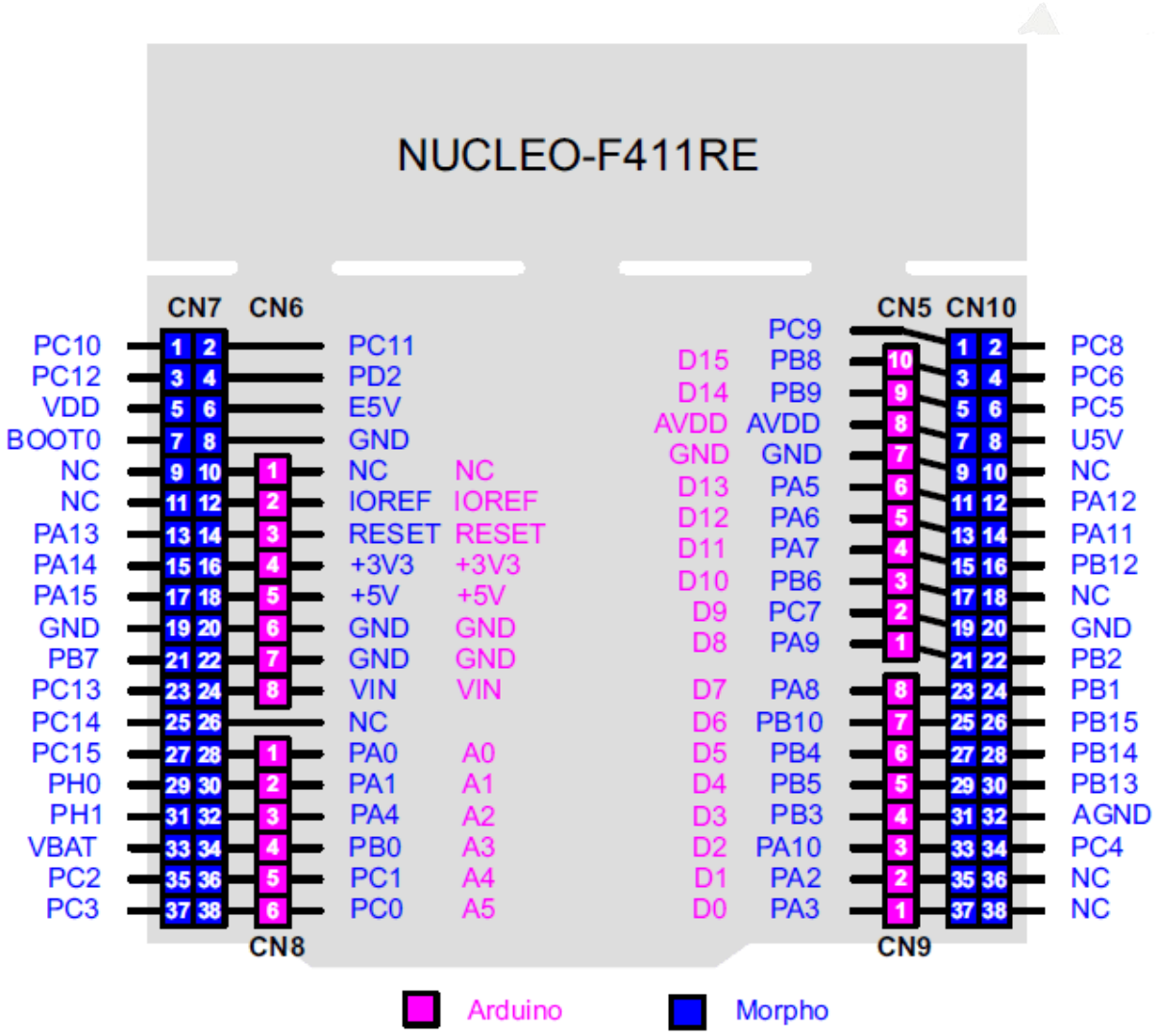


stm32 Nucleo-64

- Ce microcontrôleur présente les caractéristiques suivantes:
  - Processeur: ARM® Cortex™-M0 48MHz
  - RAM: 128 kb
  - Flash memory: 512 kb
  - 54 broches I/O
  - PWM, I²C, port série, port USB
  - ADC 16 bits
  - DAC 12 bits
  - Compatible avec Arduino
  - Prix: environ 20€
  - Programmation: C/C++ via STM32CubeMX et programmation possible en micropython pour les cartes compatibles

stm32 Nucleo-64

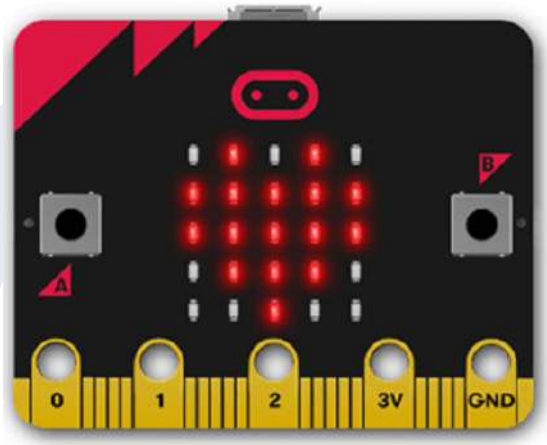
- Brochage:

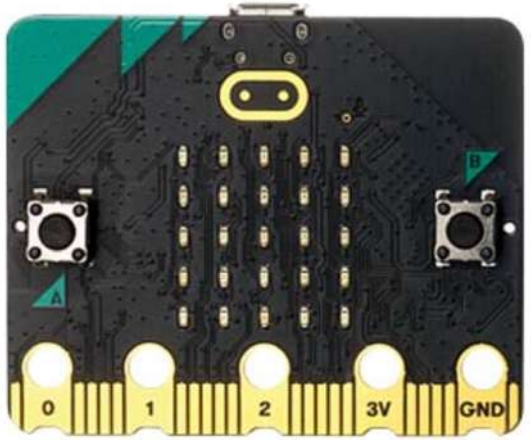


Micro:bit

- Lancé en 2015 par la BBC à destination des écoliers de 11 et 12 ans anglais pour leur apprendre les bases de la programmation

- Une version v2 a vu le jour en 2020





Micro:bit v1

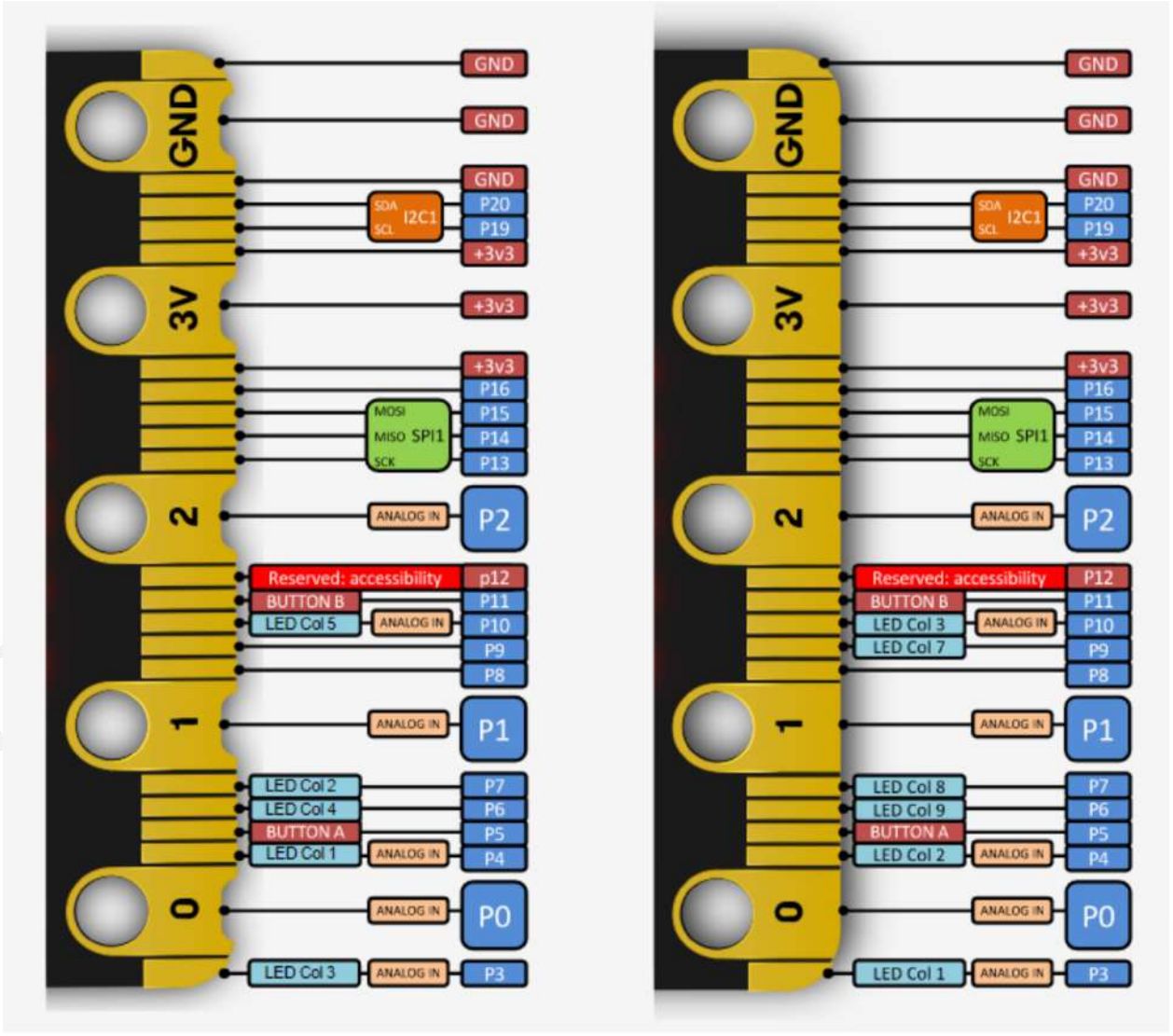
- Caractéristiques:
  - 25 broches
  - Processeur dual core Nordic nRF51822 16 MHz
  - SRAM: 16 kb
  - Flash memory: 256 kb
  - Accéléromètre, gyroscope, magnétomètre, capteur de température
  - 2 boutons programmables et 1 bouton système
  - 80 canaux radio 2.4 GHz
  - Bluetooth v4.0
  - Matrice de led 5x5
  - Prix: de 5 à 9 €
  - Programmation: en block via MakeCode, micropython

Micro:bit v2

- Caractéristiques:
  - 25 broches
  - Processeur dual core Nordic nRF52833 64 MHz
  - SRAM: 128 kb
  - Flash memory: 512 kb
  - Accéléromètre, gyroscope, magnétomètre, capteur de température
  - Micro, haut-parleur, touche sensitive
  - 2 boutons programmables et 1 bouton système
  - 80 canaux radio 2.4 GHz
  - Bluetooth v5.1
  - Matrice de led 5x5
  - Prix: de 5 à 9 €
  - Programmation: en block via MakeCode, micropython

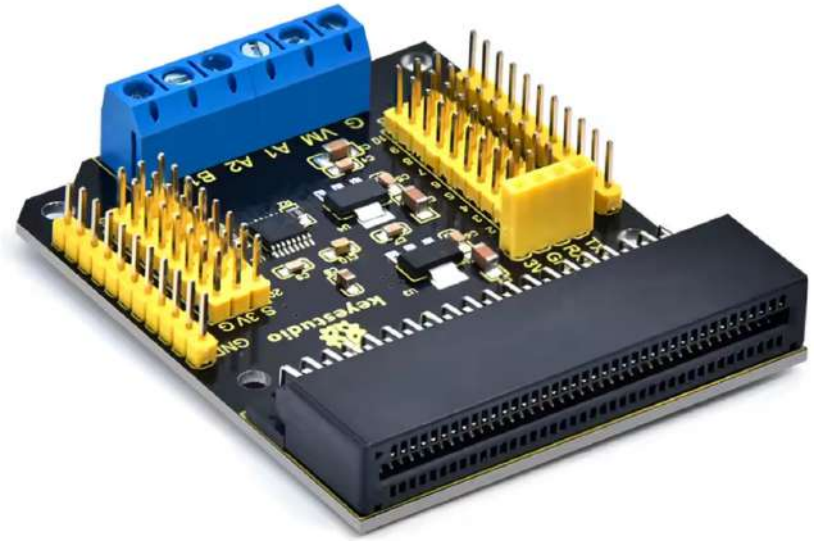
Micro:bit

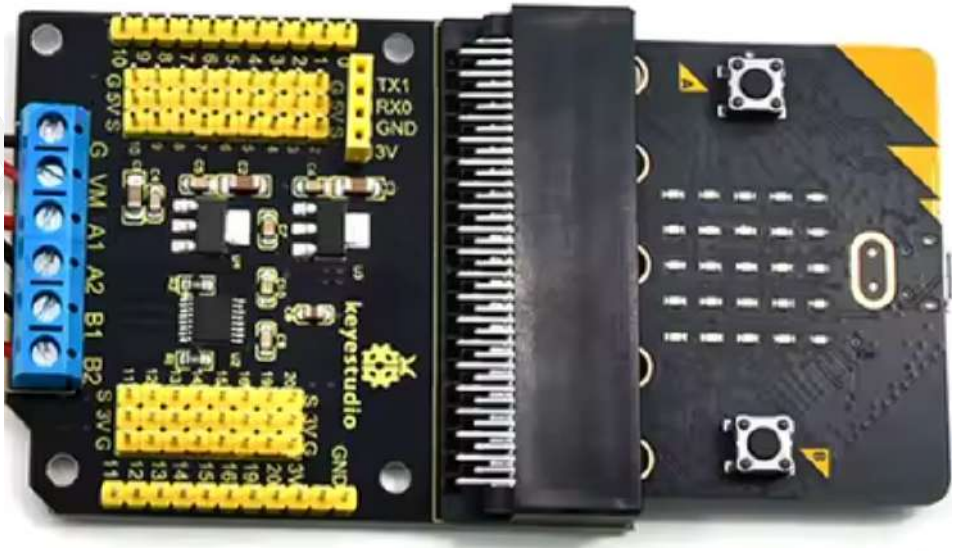
- Brochage:



Micro:bit

- Des cartes d'extension existent pour pouvoir connecter la carte micro:bit et profiter des broches I/O

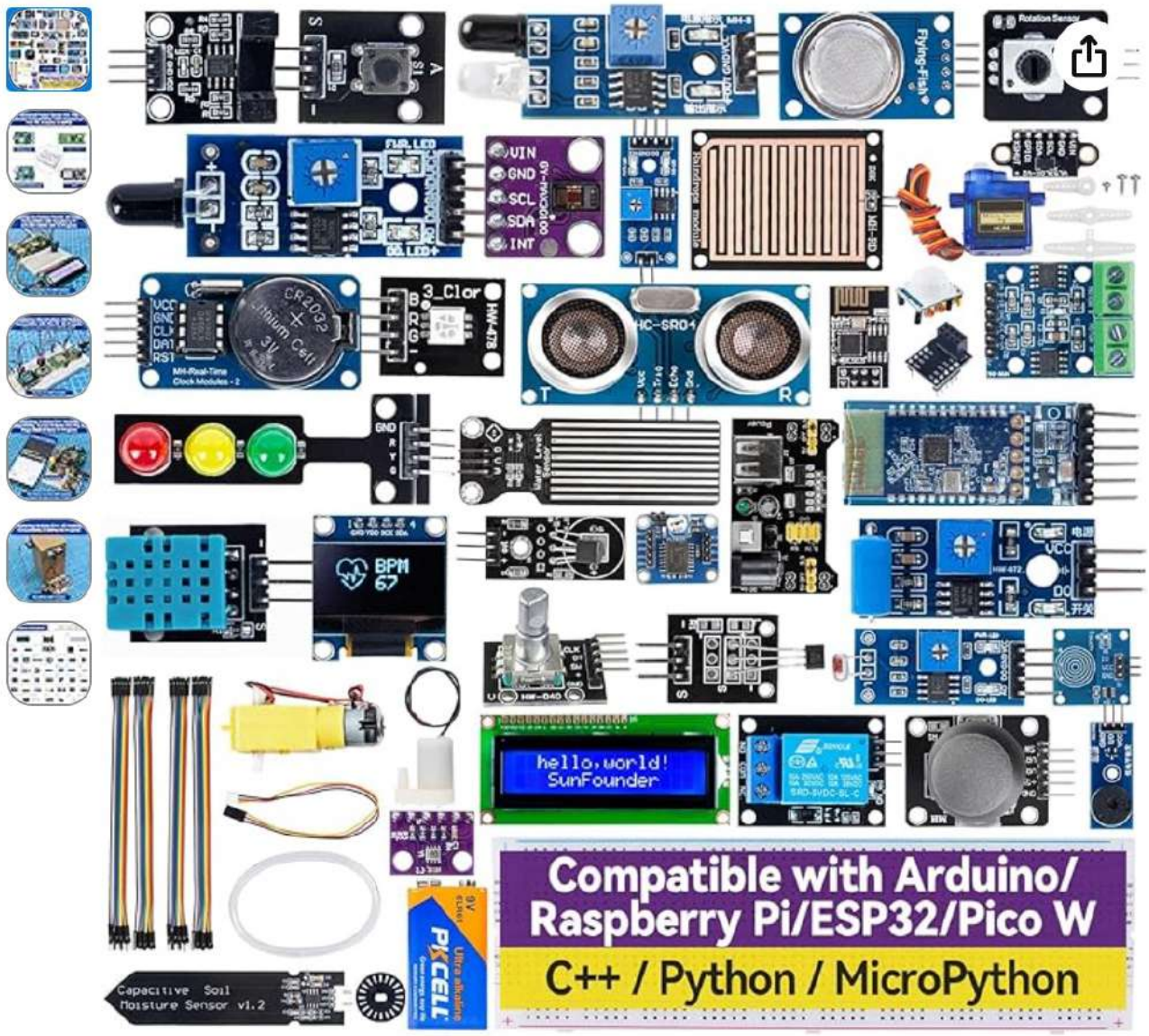




Kits de capteurs universels

- Des kits compatibles avec tous les microcontrôleurs existent et permettent de faire des expérimentations

- Prix: de 20 à 130€ suivant son contenu



Brique Lego intelligente

- Plusieurs versions se sont succédés mais la version la plus intéressante est Lego Spike
- Caractéristiques:
  - Display led blanc sous forme de matrice 5x5
  - Six ports d'entrée/sortie
  - Un gyroscope six axes (accéléromètre 3 axes et gyroscope 3 axes)
  - Navigation 3 boutons, incluant un bouton RGB
  - Speaker
  - Connectivité USB et Bluetooth
  - Programmation en blocs et en micropython

Brique Lego intelligente

- Prix: 520€ pour le kit complet (réservé aux écoles et centres de formation)

3. PILOTAGE DES ROBOTS

Commande à distance des robots

- Si le robot comporte une partie fixe (un bras robot par exemple), on pourra utiliser (et/ou fabriquer) une commande filaire

- Sinon, on aura envie de piloter à distance, sans fil, les robots construits

- Voici quelques moyens techniques pour y parvenir:
  - Radiocommande
  - Manette de jeu via un récepteur
  - Modules Bluetooth
  - Modules Wifi
  - Modules ZigBee
  - Utilisation de capteurs et robots autonomes

Radiocommande (R/C)

- Des émetteurs de radiocommande à plusieurs canaux (6 dans cet exemple) sont vendus avec le récepteur correspondant

- Ca permet de contrôler une paire de moteurs via une carte sabertooth, un autre moteur via une carte syren et de contrôler quelques servos

- Prix: ça va des modèles chinois à moins de 50€ aux modèles pro à plus de 700€

Radiocommande (R/C)

- Il existe d'autres types de radiocommande comme cette manette 8 canaux

- On pourrait en utiliser deux simultanément pour doubler le nombre de canaux

Manette de jeu via un récepteur

- Si on utilise un Arduino Mega ADK (avec un USB B intégré), on peut y connecter un récepteur de manette Xbox 360 wireless

- Une librairie Arduino a été créée pour gérer ce récepteur et piloter la manette

Manette de jeu via un récepteur

- Les 2 joysticks et les nombreux boutons permettent beaucoup de possibilités de contrôle, plus qu'avec une radiocommande
- Des drivers moteurs, une carte son, un driver servos et des capteurs peuvent connecter à l'Arduino
- C'est un système assez complet et relativement bon marché
- Ca fonctionnera avec n'importe quelle manette ayant un récepteur USB

Manette de jeu via un récepteur

- On a l'équivalent avec une manette de PS2 sans fil avec l'avantage de pouvoir utiliser un Arduino UNO

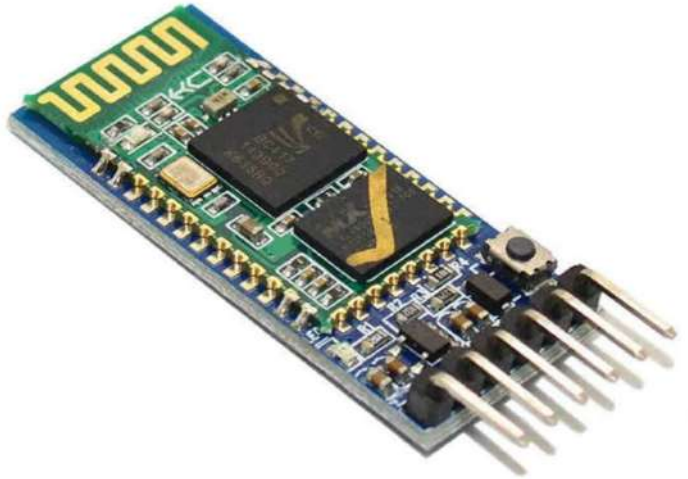
- On trouve le kit complet sur aliexpress pour moins de 40€

Modules Bluetooth

- Les modules Bluetooth HC05 peuvent être connectés à un microcontrôleur pour lui donner la connectivité Bluetooth (pour ceux qui n'en sont pas déjà doté)

- Ils peuvent s'associer pour former une paire émetteur/récepteur

- On peut alors créer une manette Bluetooth à partir de modules joystick et de boutons



Modules Bluetooth

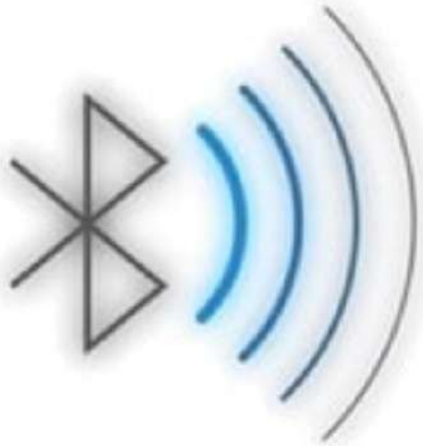
- On peut aussi utiliser des manettes de jeu Bluetooth comme celle de la xbox one (série S v2 ou série X) avec un microcontrôleur possédant la fonctionnalité Bluetooth ou via un HC05

- Toute manette ou pad Bluetooth peut faire l'affaire

- On peut, par exemple, piloter la brique Lego Spike avec une manette Bluetooth via le projet pybricks (en micropython)

Modules Bluetooth

- On peut aussi contrôler les actions du robot avec une appli sur smartphone et les ordres seront transmis par Bluetooth
- On pourra développer l'appli en python par exemple

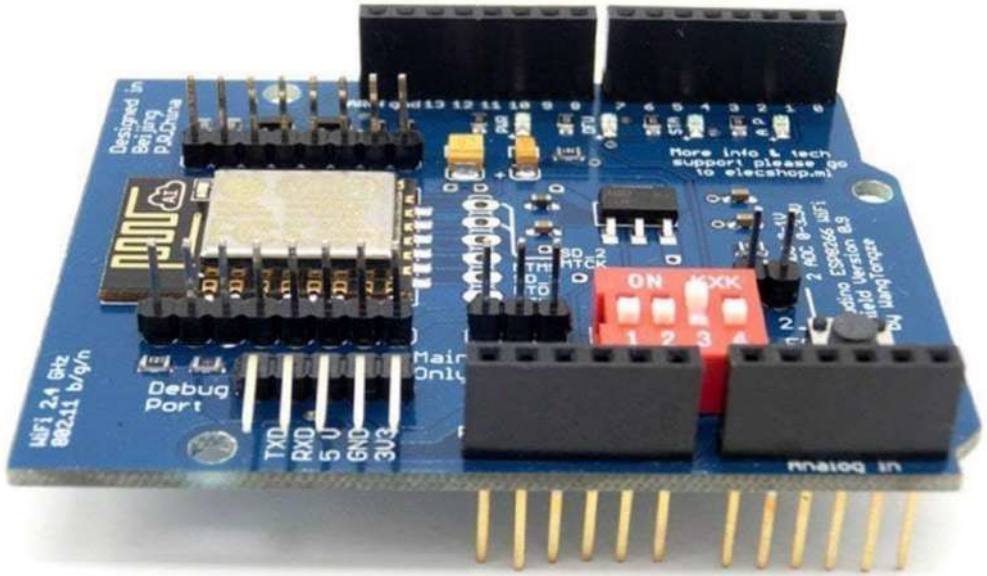


Modules Wifi

- Différents microcontrôleurs possèdent un module Wifi et des shields Wifi existent comme celui-ci de chez Dollatek

- Prix: 7€ sur Amazon

- On peut alors communiquer à travers le Wifi via un ordinateur sur lequel on peut créer un programme de commande du robot



Modules Wifi

- On peut également créer une radiocommande Wifi avec deux ESP32
- Du côté de l'émetteur, on créera une commande avec joysticks et boutons et du côté du récepteur, on connectera l'ESP32 directement aux récepteurs ou à un autre microcontrôleur
- Exemple:

Modules ZigBee

- Ils existent d'autres technologies que le Bluetooth et le Wifi
- ZigBee, par exemple, est un réseau sécurisé et peu gourmand en ressources mais avec un débit beaucoup plus faible que les versions actuelles du Bluetooth
- Ce sont des modules émetteur/récepteur

Capteurs et robots autonomes

- Tous les projets robots n'ont pas besoin d'un contrôle temps réel

- Les actions peuvent être préprogrammées en fonction d'évènements:
  - Détection d'un obstacle
  - Appui sur un bouton ou fin de course
  - Suivi de ligne
  - Modification de: la t°, la luminosité, de la pente …

Capteurs et robots autonomes

- Plusieurs capteurs correspondants existent:
  - Bouton-poussoir, fin de course
  - Détecteur d'obstacle à ultrasons, à infrarouge
  - LDR
  - Sonde de température
  - Gyroscope
  - …
- On peut ainsi préprogrammer un robot en fonction de son environnement et des tâches à effectuer

4. ALIMENTATION

Alimentation des robots créés

- Pour faire fonctionner un robot, il faut une alimentation électrique

- Si le robot comporte une partie fixe, on pourra l'alimenter par une alimentation stabilisée connectée au secteur, c'est ce qui revient le moins cher

- S'il est mobile, il devra être alimenté par des piles et/ou batteries plus ou moins puissantes suivant la consommation du robot

Alimentation stabilisée

- On en trouve de toutes les tensions secondaires, puissances, qualités (au niveau stabilité, régulation, filtrage) et donc à tous les prix

- Les plus couramment utilisées seront les alims à découpage 5V, 12V et 24V, disponibles pour
un montant de 15 à 40€
suivant le modèle

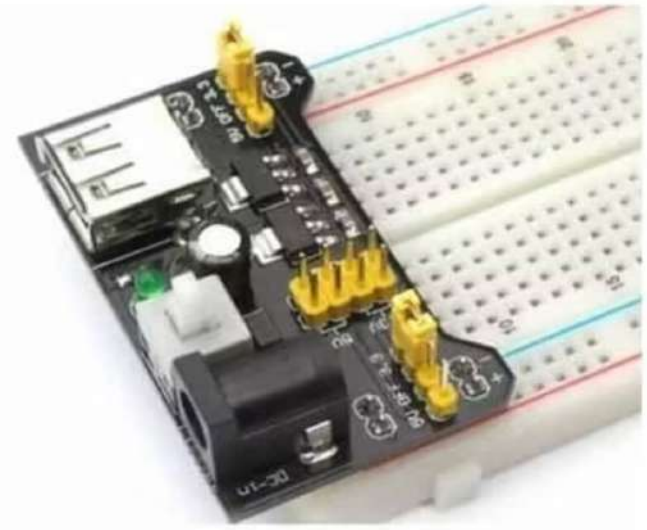
Adaptateur secteur

- Pour alimenter les microcontrôleurs et les accessoires pour les essais préliminaires, on peut utiliser un ou plusieurs adaptateurs secteurs

- Suivant le microcontrôleur à alimenter,
  il faudra choisir le voltage et l'embout de sortie

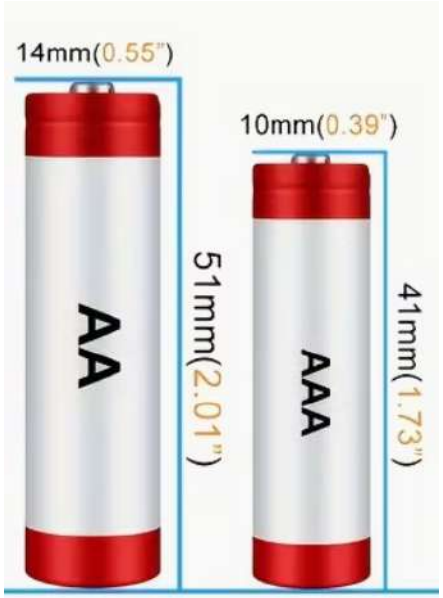
Adaptateur secteur

- Pour alimenter un breadboard, un shield alimentation existe
- Il se fixe sur le breadboard et s'alimente avec un adaptateur secteur (généralement 9V) ou en USB



Piles

- Les plus courantes sont les piles de 1,5V (attention à la taille AA ou AAA) et de 9V
- C'est pratique mais ça s'use vite et ça revient cher
- On peut leur préférer une version rechargeable en NiMH



Batteries

- Plusieurs technologies existent:
  - Plomb
  - NiCd
  - NiMH
  - Li-ion
  - LiPo

Batteries au plomb

- C'est la plus ancienne technologie, peu performante, polluante en fonction de ses composants, mais peu chère et sans danger pour une utilisation normale
- On trouvera couramment des batteries de 6, 12 et 24V de toutes les puissances possibles
- Gros avantages: pouvoir fournir un courant de crête important (démarrage d'un moteur)
- Exemple: une batterie 12V, 20Ah pour 50€ sur Amazon

Batteries NiCd

- Les batteries Nickel-Cadnium sont délaissées au profit des batteries NiMH à cause:
  - D'un effet mémoire prononcé pour les batteries NiCd (si on recharge la batterie successivement avant qu'elle ne soie déchargée, la tension chute et la batterie se vide rapidement à l'usage)
  - De la pollution générée par le Cadnium
  - D'une capacité qui peut être plus importante pour les NiMH
- Par contre, en cas de besoin d'un courant de crête, le NiCd reste supérieur au NiMH

Batteries NiMH

- Les batteries Nickel-Metal Hydrure sont plus performantes et plus écologiques que les NiCd
- On les retrouve dans la plupart des piles rechargeables
- L'inconvénient par rapport aux piles, c'est que la tension nominale est de 1,2V au lieu de 1,5V

Batteries Li-ion et LiPo

- Batteries très performantes mais très délicates à utiliser

- Elles supportent mal la surcharge et la charge doit être régulée électroniquement sous peine de détruire la batterie et de provoquer un incendie
- Elles doivent être stockées avec une charge entre 40 et 50% sinon leur chimie est abimée, elles gonflent et deviennent inutilisables et dangereuses
- Si on les perce, elles prennent feu
- Avec le temps, une accumulation de Lithium peut se former au niveau de l'anode, rejoindre la cathode et provoquer un court-circuit interne et un incendie

- A part ça tout va bien ;-)

Batteries Li-ion et LiPo

- Une cellule produit typiquement une tension de 3,7V

- On trouvera ainsi des batteries:
  - 1s, une cellule: 3,7V
  - 2s, deux cellules: 7,4V
  - 3s, trois cellules: 11,1V
  - 4s, quatres cellules: 14,8V
  - …

- Le s signifie serie, il indique le nombre de cellules mises en série

Batteries Li-ion et LiPo

- Décryptage d'une batterie LiPo
  - 5200mAh: c'est la capacité de la batterie, plus elle est élevée, plus l'autonomie sera grande
  - 50C: c'est la capacité maximum de décharge continue
    Le courant continu maximum qui peut être consommé en continu est donné par 50Cx5200mAh -> 50x5,2= 265A maximum
  - 22.2V: c'est une batterie 6s

Batteries Li-ion et LiPo

- On voit que la batterie possède un connecteur principal, un XT60 mâle (ce sont parfois des XT90 de même forme mais de dimension différente)

- Le second connecteur blanc sert pendant la charge pour équilibrer la charge des différentes cellules

Batteries Li-ion et LiPo

- Pour charger les batteries Li-ion et LiPo, des chargeurs spécifiques existent comme ce B6AC très classique à 30€ sur Amazon pour charger les batteries de 2s à 6s

Batteries Li-ion et LiPo

- Il existe des batteries chinoises avec moins de sécurité et sans équilibrage de charge possible
- Exemple: batterie 24V 20Ah, idéale pour piloter un robot de grande taille comme un R2D2 toute une journée, tout ça pour 85€ sur Amazon
- Ce n'est pas cher, ça fonctionne bien mais toutes les précautions d'usage doivent être prises !
- Depuis quelques années, dans les conventions et autres salons, il est INTERDIT de laisser les batteries charger la nuit, en raison du danger potentiel …

Batteries Li-ion et LiPo

- Pour les petits projets robotiques, il existe aussi les batteries Li-ion 18650 de 3,6V et de 1800 à 4000mAh

- Le support permet la charge des batteries in situ (sur place)

- Une version plus ancienne de cette batterie en NiMH existe également

Facteur de décharge automatique

- Si on laisse une batterie chargée sans utilisation, elle se décharge naturellement d'un certain pourcentage par mois

- Voici le pourcentage de perte par mois et la façon de conserver vos batteries

Technologie	% de perte par mois	Stockage longue durée
Plomb	5%	Chargée
NiCd	20%	Déchargée
NiMH	15%	Chargée
Li-ion	3%	40 à 50% de la charge
Li-Po	2%	40 à 50% de la charge

Régulateurs de tension

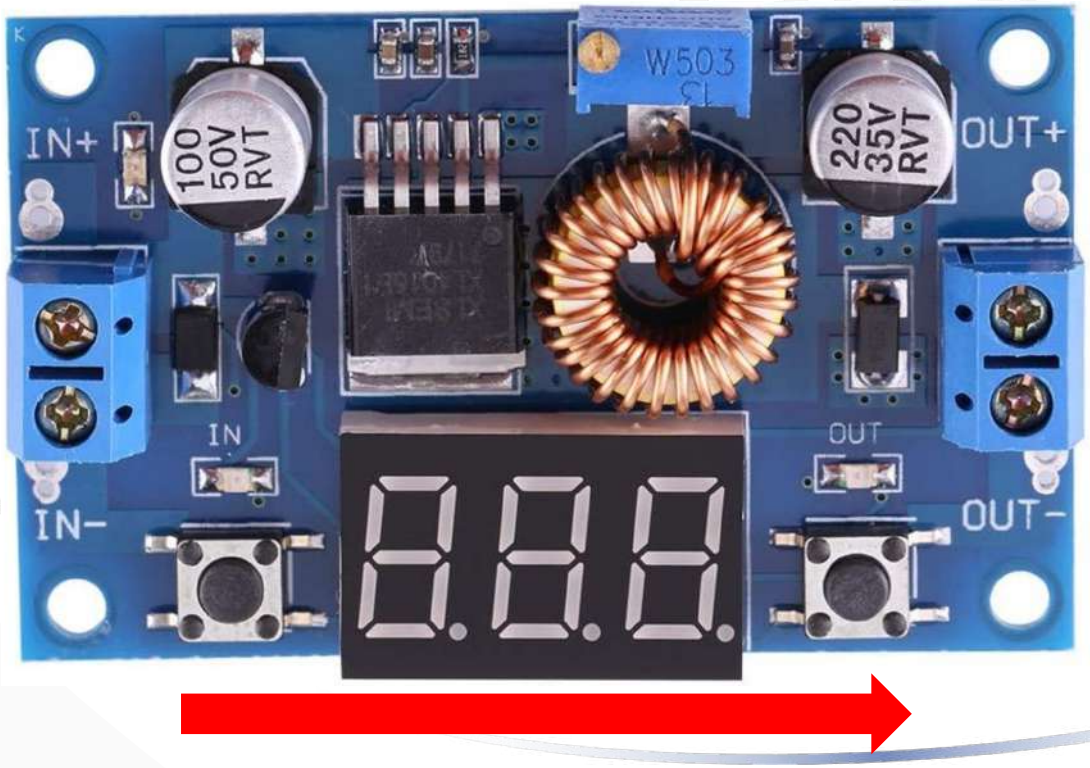
- Si dans notre projet, nous utilisons un moteur de 24V DC, un Arduino UNO sous 5V et une carte son alimentée sous 12V, nous n'allons pas multiplier les batteries

- Dans le cas de cet exemple, on utilisera une batterie de 24V qui servira à alimenter l'ensemble et on utilisera des abaisseurs de tension pour produire le 12V et le 5V

Régulateurs de tension

- Quelques modèles:
  - Abaisseur DC-DC 5A 75W à 7€ sur Amazon
    Fonctionne bien mais pas de protection contre les courts-circuits et les inversions de polarité

4 à 38V



1,25 à 36V

Régulateurs de tension

- Abaisseur DC-DC 12/24V -> 5V 15A à 19€ sur Amazon
  Produit un 5V stable, peu importe la tension d'entrée (entre 12 et 24V)

- Existe aussi en 12V et en 24V
  comme tension secondaire

Protection des circuits

- Des fausses manœuvres, des accidents, des circuits qui crament, ça n'arrive pas qu'aux autres

- On protègera donc les circuits par des fusibles placés directement après la source d'alimentation

- Dans les montages électroniques, on les trouvera principalement sous ces deux formes:

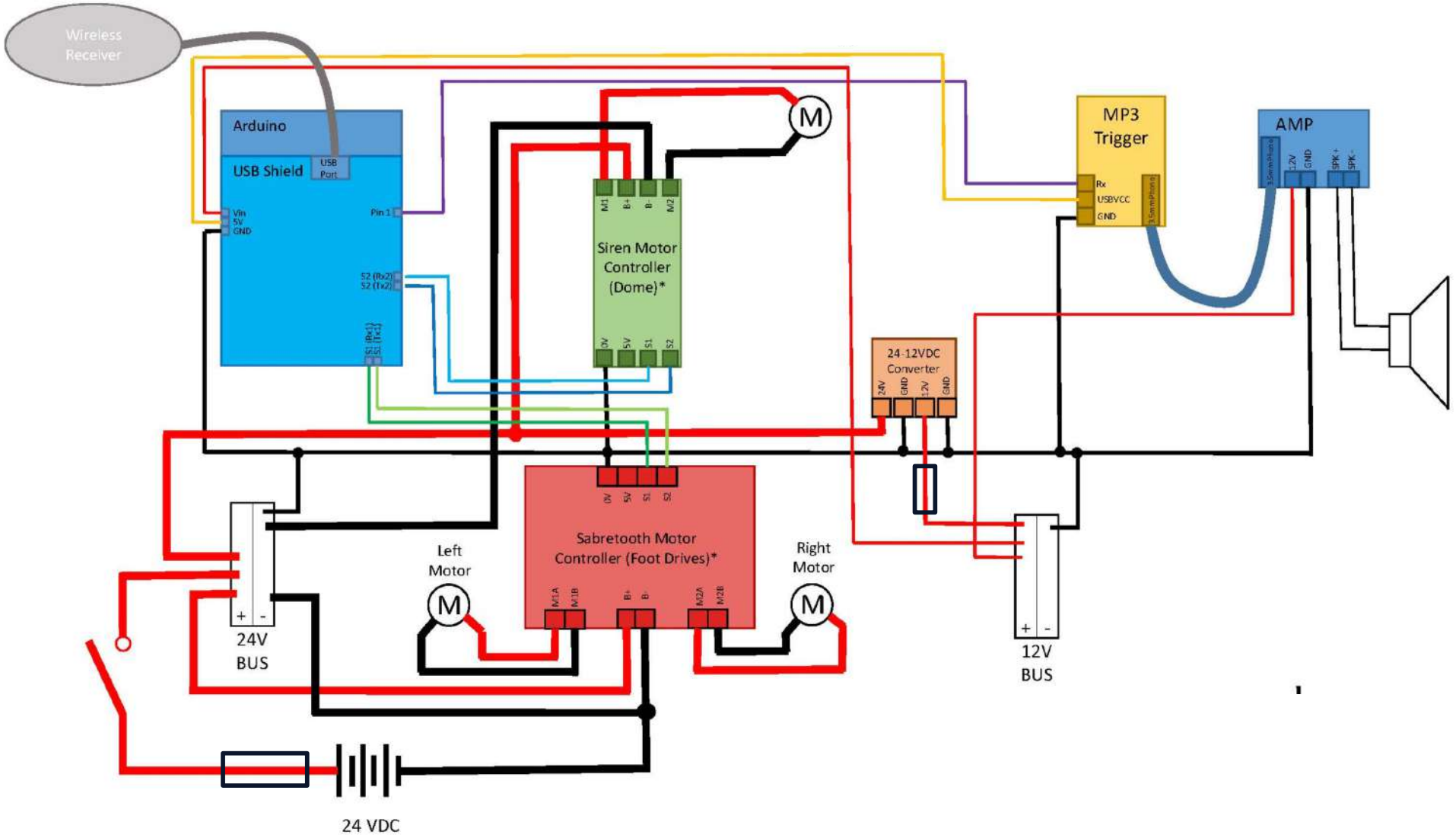
Protection des circuits

- Au niveau des portes fusibles, on trouvera ceux-ci:

Alimentation – remarque générale

- Comme chaque fois qu'on manipule du courant, on fait attention à ce qu'on fait
- Pour les alimentations secteur; on isole les broches de connexions du primaire reliées au secteur 230V
- Pour toutes les alimentations, on protège ses circuits par des fusibles calibrés
- On charge toujours les batteries avec le chargeur adéquat (pas seulement vérifier que l'embout rentre, vérifier la tension de charge)
- On évite, bien sur, les courts-circuits qui peuvent détruire alimentations, batteries, composants, … et accessoirement mettre le feu

Exemple de schéma



Conclusion

- Bon, il y a encore beaucoup à dire et beaucoup de connexions à faire avec ce qui a déjà été dit mais au moins, vous avez des pistes pour démarrer

- Expérimentez, c'est le mieux

- Vous voilà capable de contrôler le mouvement de vos créations