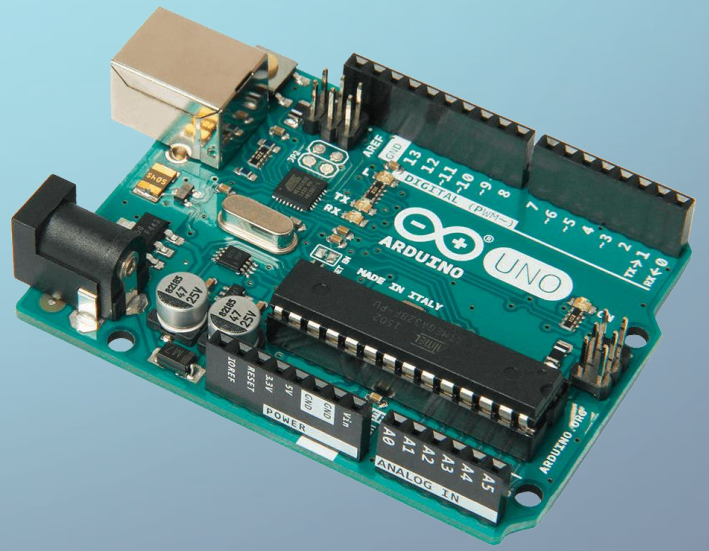
Formation préparée par Katherine Zamudio-Turcotte
PolySTAR, 2021.

POLYSTAR

PROGRAMMATION ET PROTOTYPAGE AVEC

DÉROULEMENT DE LA FORMATION

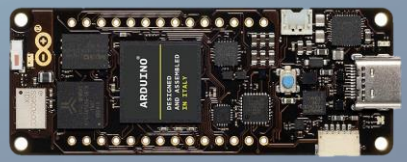
1. Introduction aux boards Arduino
2. Méthodologie de prototypage
3. Logique d'un code typique
4. Concept de PWM
5. Atelier de programmation



1. INTRODUCTION AUX BOARDS ARDUINO

Avantages :

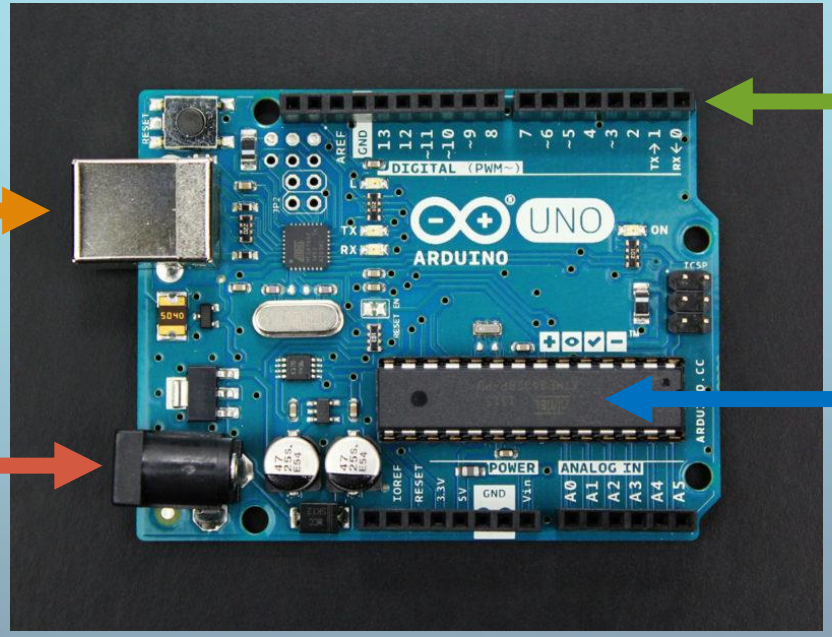
- Faciles à programmer (C++ simplifié)
- Beaucoup de produits électroniques compatibles (boards d'expansion, capteurs, motor drivers, etc.)
- Beaucoup de librairies de code open source
- Beaucoup d'exemples et de tutoriels sur le web



1. INTRODUCTION AUX BOARDS ARDUINO

Liste des modèles : https://store.arduino.cc/usa/arduino/boards-modules

Anatomie d'un Arduino Uno :



Port USB-B pour programmer le microcontrôleur

Courant entrant 7-12V (si le USB-B est déconnecté)

Plusieurs pins pour connecter des dispositifs externes

Microcontrôleur Atmel qui exécute notre code

1. INTRODUCTION AUX BOARDS ARDUINO

Où on peut se procurer des Arduino et du matériel pour Arduino :

Robotshop : https://www.robotshop.com/ca/en/

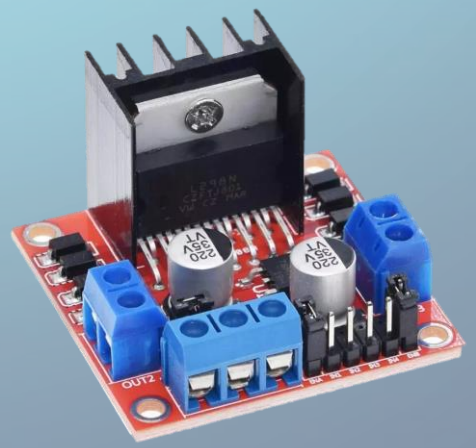
ABRA : https://abra-electronics.com/

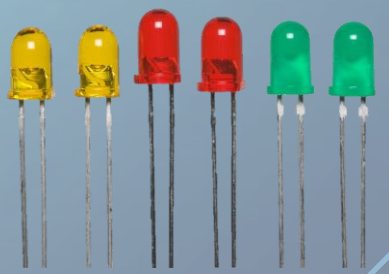
Addison : https://addison-electronique.com/en/

Amazon : https://www.amazon.ca/

Adafruit : https://www.adafruit.com/

Digikey : https://www.digikey.ca/





2. MÉTHODOLOGIE DE PROTOTYPAGE

I. Trouver des moteurs, des capteurs, des LEDs, des dispositifs à utiliser

II. Observer le matériel : “ai-je besoin de boards d’expansion additionnels pour le connecter au Arduino?”

    I. Chercher de la documentation / des tutoriels sur tout le matériel trouvé

III. Rédiger un programme en C++ sur Arduino IDE qui rassemble tout le code pour faire fonctionner les dispositifs choisis

2. MÉTHODOLOGIE DE PROTOTYPAGE

I. Trouver des moteurs, des capteurs, des LEDs, des dispositifs à utiliser

II. Observer le matériel : “ai-je besoin de boards d’expansion additionnels pour le connecter au Arduino?”

   I. Chercher de la documentation / des tutoriels sur tout le matériel trouvé

III. Rédiger un programme simple sur Arduino IDE en ajoutant des lignes une par une tranquillement

****Important****

Si on cherche à faire tourner un moteur (DC, servo, stepper)…

Souvent besoin d’un circuit spécialisé entre les moteurs et le Arduino !

On ne peut pas connecter un moteur de plus de 5V/1A directement à un Arduino, car le Arduino ne pourra pas donner assez d’ampérage et/ou de voltage!

3. LOGIQUE D'UN CODE TYPIQUE

Procédure qui s'exécute 1 fois au début du programme

Procédure qui s'exécute après setup et qui se répète tant et aussi longtemps que le Arduino reste allumé

```
void setup() {
  // put your setup code here, to run once:
  initializeTemperatureSensor();
  Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  Serial.println("temperature: %d", readSensor());
  delay(50);
}
```

Délai de 50 millisecondes entre chaque exécution de la boucle loop

3. LOGIQUE D'UN CODE TYPIQUE

Des fonctions utiles :

digitalRead(12) : lire la valeur sur la pin numérique 12 (0 ou 1)

pinMode(12, INPUT) : initialiser la pin numérique 12 comme une pin qu'on lit

digitalWrite(12, HIGH) : écrire la valeur 1 sur la pin numérique 12

pinMode(12, OUTPUT) : initialiser la pin numérique 12 comme une pin sur laquelle on écrit

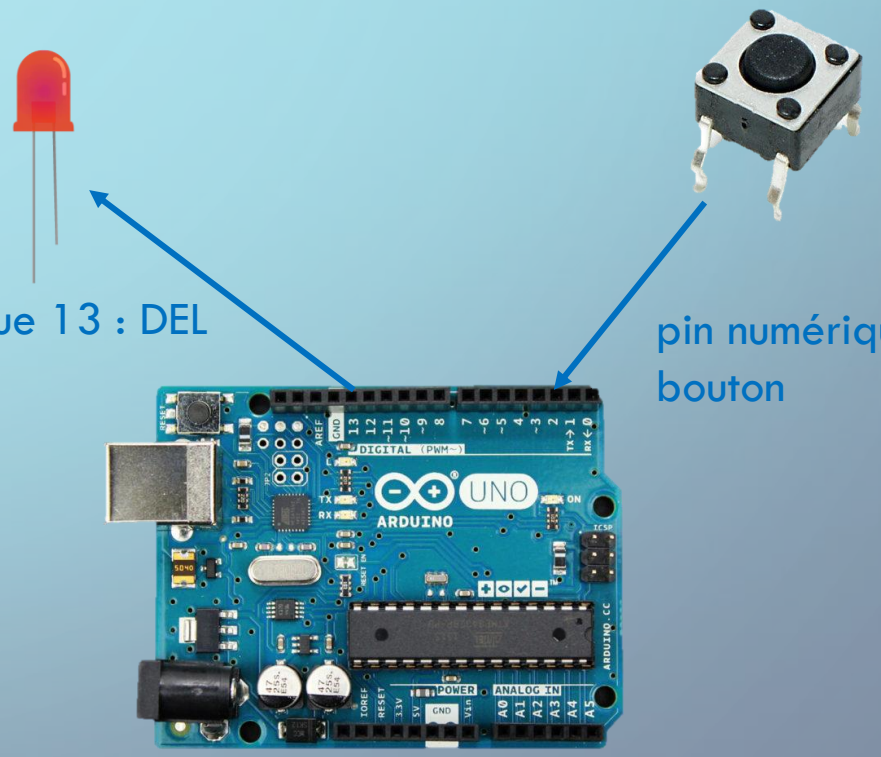
analogRead(A3) : lire la valeur sur la pin analogique A3 (entre 0 et 1023)

analogWrite(3, 2) : écrire une onde PWM avec *duty cycle* de 2
(une valeur de 8 si on prend l'échelle 0-1023)
sur la pin numérique 3

3. LOGIQUE D'UN CODE TYPIQUE

Autre exemple :

```
int etatDEL = HIGH;

void setup() {
  pinMode(2, INPUT);
  pinMode(13, OUTPUT);
  digitalWrite(13, etatDEL);
}

void loop() {
  int etatBouton = digitalRead(2);
  if (etatBouton == HIGH) {
    etatDEL = !etatDEL;
    digitalWrite(13, etatDEL);
  }
}
```



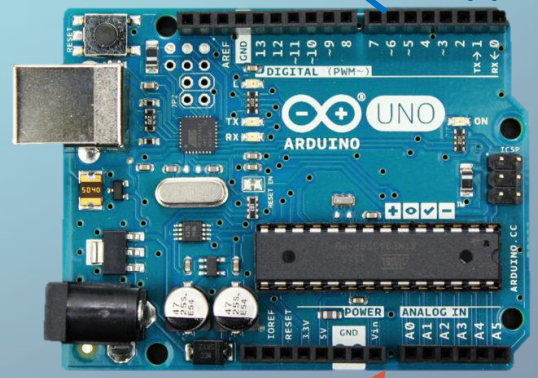
3. LOGIQUE D'UN CODE TYPIQUE

Autre exemple :

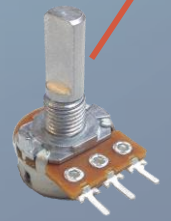
```
void setup() {
}

void loop() {
  int potentiometre = analogRead(0);
  int valeur = map(potentiometre, 0, 1023, 0, 255);
  analogWrite(6, valeur);
  delay(15);
}
```

Si on assume que les fils VCC et GND du moteur sont connectés à une batterie externe

pin numérique 6 : fil *"data"* d'un moteur *stepper*



pin analogique 0 : potentiomètre



3. LOGIQUE D'UN CODE TYPIQUE

Liste des librairies principales pour Arduino :
https://www.arduino.cc/reference/en/libraries/

Quelques propositions pour utiliser des moteurs :

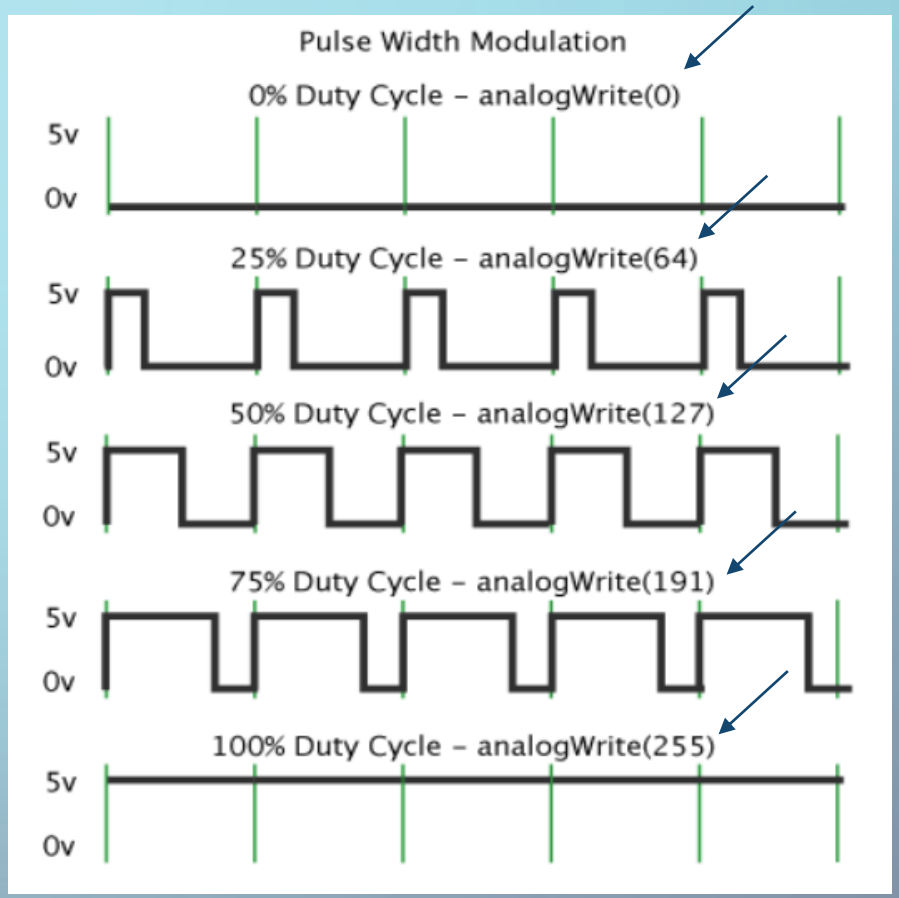
- DC → un module L298N et sa librairie
- Stepper unipolaire → un Darlington Array et la librairie *stepper*
- Stepper bipolaire → une puce SN754410NE et la librairie *stepper*
- Servo → un Darlington Array et la librairie *servo*

Comme cela a été mentionné plus tôt, si vous utilisez plus d'un moteur, ou un moteur qui demande plus de 5V ou de 1A, vous ne devriez pas simplement brancher le moteur au Arduino, cela pourrait l'endommager.

4. CONCEPT DE PWM

Vraiment utile pour contrôler des moteurs!!!

Idée générale :
Plus on augmente le *duty cycle* (%), plus la vitesse est grande.

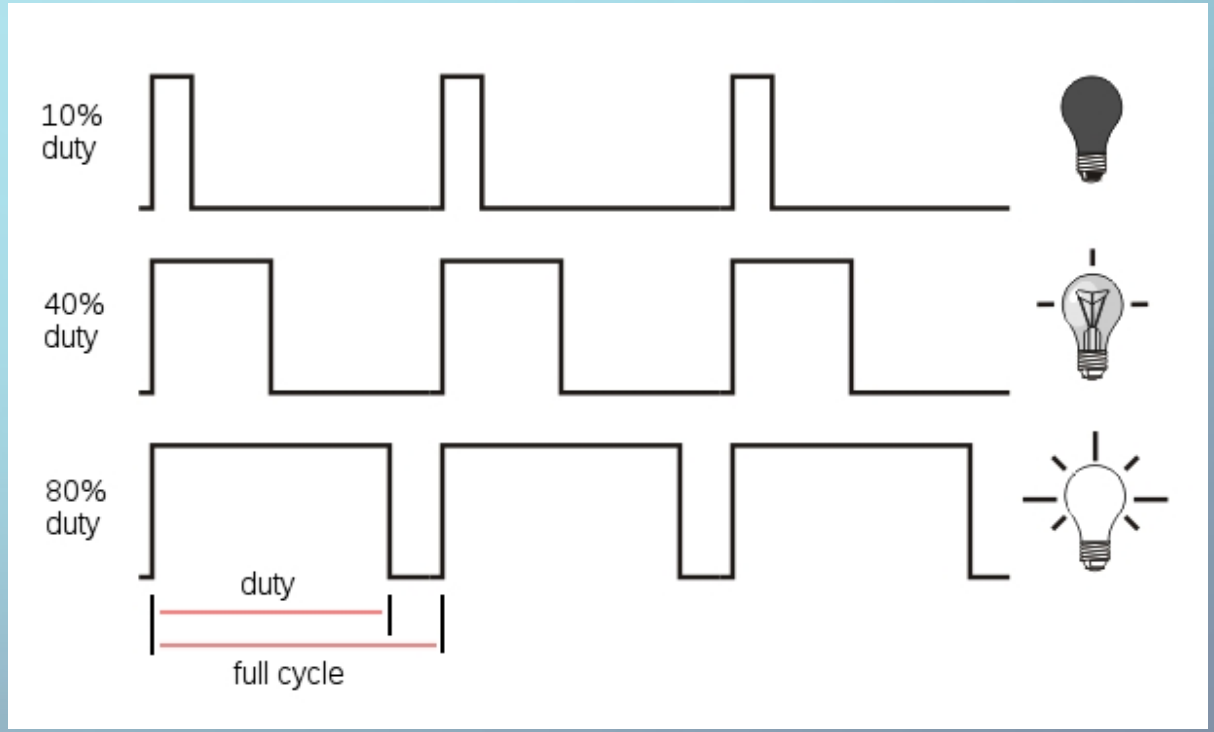


Quand l'onde est à 5V, on est en train de dire au moteur "va à ta vitesse maximale".
Quand l'onde est à 0V, on est en train de dire au moteur "ne bouge pas".
Plus on passe de temps à 5V, plus la moyenne donne une vitesse grande.

4. CONCEPT DE PWM

C'est la même chose avec une DEL :

Quand l'onde est à 5V, on est en train de dire à la DEL "soit allumée et à une intensité maximale".
Quand l'onde est à 0V, on est en train de dire à la DEL "soit éteinte".
Plus on passe de temps à 5V, plus la moyenne donne une luminosité élevée de la DEL.



4. CONCEPT DE PWM

Utilité des PWM :

Représenter une quantité “analogique” (0.00, 0.25, 0.32, 1.00) avec des données numériques (des 1 et des 0).

En numérique :

11110000111100001111000 0

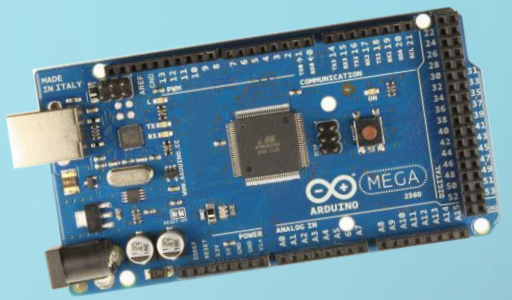
(50% du temps allumé, 50% du temps éteint)

Signification analogique :

0.5 0.5 0.5 0.5

QUESTIONS ?

5. DÉMO

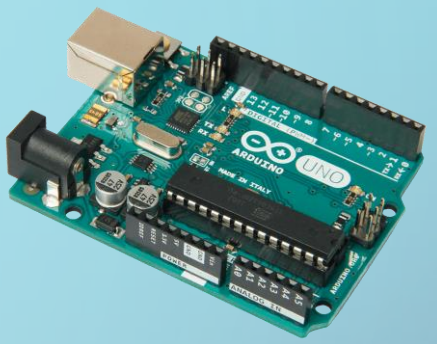


1. Mon matériel : un Arduino Mega 2560, un gearbox de 2 twin-motors Tamiya et un capteur de distance HC-SR04

2. Recherche sur le matériel : “est-ce que j’ai tout?”
    1. Type de moteurs : DC, servo, stepper?
    2. Voltage/ampérage requis par les moteurs
    3. Pins du capteur vs. connexions disponibles sur le Arduino
    4. Librairies de code disponibles pour les moteurs ou le capteur?

3. Réponse possible

MERCI DE VOTRE ÉCOUTE !!!



Quelques ressources additionnelles :

- https://www.youtube.com/playlist?list=PLGs0VKk2DiYw-L-RibttcvK-WBZm8WLEP
- https://www.tutorialspoint.com/arduino/arduino_dc_motor.htm
- https://www.arduino.cc/en/Tutorial/LibraryExamples/StepperSpeedControl