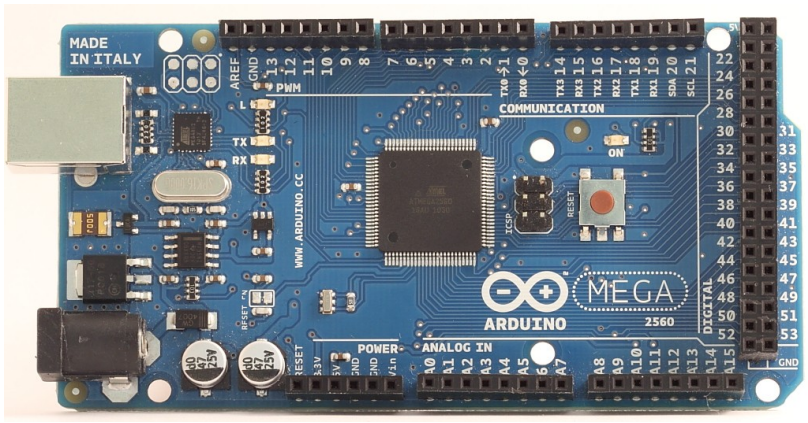
Entrées analogiques

Les connecteurs "Analogiques" :

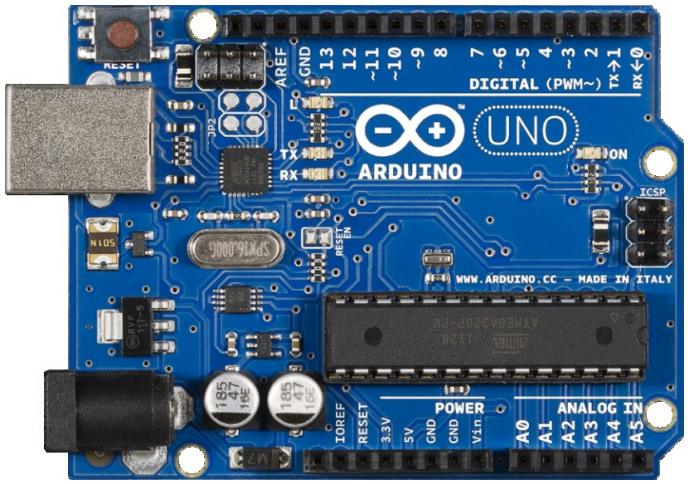
Q1. Entourer les connecteurs analogiques sur la carte Arduino MEGA 2560.

Q2. Donner leur nombre :



Q3. Entourer les connecteurs analogiques sur la carte Arduino UNO.

Q4. Donner leur nombre :



Vocabulaire utilisé en électronique :

Q5. Donner les traductions de (pour une utilisation en électronique/électricité) :

Output	
Input	
High (état logique ou niveau logique)	
Low (état logique ou niveau logique)	
Pin	
Digital	
Analog	
IN	

Rappel :

digital signifie numérique (0 ou 1) soit une tension de 0V **ou** 5V (uniquement 2 valeurs)

analog est la contraction de analogique soit une tension de 0V **à** 5V (une infinité de valeurs)

Programmation d'une carte Arduino :

Demander une carte Arduino ainsi qu'un câble USB et un potentiomètre (voir le professeur).

Lancer le logiciel Arduino , sélectionner la bonne carte (c'est comme le port salut, c'est écrit dessus) à l'aide du MD (Menu Déroulant) "Outils" puis "Type de carte". Sélectionner le port pour que la communication entre le PC et la carte puisse avoir lieu (MD "Outils" puis "Port série", prendre le 3ème COM de la liste).

I. Réalisation d'un cligotement variable :

Sélectionner dans le MD "Exemples" puis "03.analog" le fichier "AnalogInput".

Q6. Donner la valeur du potentiomètre utilisé (valeur multiple de 1 ou 2,2 ou 4,7Ω) :

Q7. Qu'est ce qu'un potentiomètre ?

Le programme "AnalogInput" est le suivant :

```
int sensorPin = A0;    // select the input pin for the potentiometer
int ledPin = 13;      // select the pin for the LED
int sensorValue = 0;  // variable to store the value coming from the sensor

void setup() {
  // declare the ledPin as an OUTPUT:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  // turn the ledPin on
  digitalWrite(ledPin, HIGH);
  // stop the program for <sensorValue> milliseconds:
  delay(sensorValue);
  // turn the ledPin off:
  digitalWrite(ledPin, LOW);
  // stop the program for for <sensorValue> milliseconds:
  delay(sensorValue);
}
```

Q8. Donner la traduction de :

Anglais	Français
Potentiometer attached to analog input 0	
* center pin of the potentiometer to the analog pin :	
* one side pin (either one) to ground :	
* the other side pin to +5V :	
sensor :	
select the input pin for the potentiometer :	
variable to store the value coming from the sensor	

Q9. Donner la couleur du fil du potentiomètre à relier au +5V :

Q10. Donner la couleur du fil du potentiomètre à relier au 0V :

Q11. Où faut-il relier l'autre fil du potentiomètre ?

Brancher le potentiomètre sur la carte Arduino (appeler le professeur pour vérification).

Téléverser le programme (bouton en dessous de la barre des MD) dans la carte Arduino. Faire varier la valeur du potentiomètre (en tournant l'arbre) et observer (sur la carte !!) ce qui se passe (vous pouvez aussi vous aider du programme) :

Fonction setup :

Q12. Donner la traduction de : pinMode(ledPin, OUTPUT);

Q13. Combien de fois est lu le contenu de la fonction setup ?

Fonction loop :

Q14. Donner la traduction de : // read the value from the sensor:

Q15. Donner la signification de l'abréviation analog :

Q16. En déduire la signification de sensorValue = analogRead(sensorPin);

Q17. Combien de fois est lu le contenu de la fonction loop ?

Q18. En déduire la signification de delay(sensorValue);

Q19. Appeler le professeur pour vérification de la saisie du circuit et le schéma sous fritzing

Rappel :

digital signifie numérique (0 ou 1) soit une tension de 0V **ou** 5V (uniquement 2 valeurs)

analog est la contraction de analogique soit une tension de 0V **à** 5V (une infinité de valeurs)

Q20. Quelles sont les tensions possibles issues du potentiomètre ?

Q21. En déduire si la tension issue du potentiomètre est une tension analogique ou numérique :

II. Affichage du résultat de la tension issue du potentiomètre :

Sélectionner dans le MD "Exemples" puis "01.basics" le fichier "AnalogReadSerial".

```
/*
  AnalogReadSerial
  Reads an analog input on pin 0, prints the result to the serial monitor.
  Attach the center pin of a potentiometer to pin A0, and the outside pins to +5V and ground.

 This example code is in the public domain.
 */

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  delay(1);        // delay in between reads for stability
}
```

Entrées analogiques

Téléverser le programme (bouton en dessous de la barre des MD) dans la carte Arduino.

Sélectionner le moniteur série (icône à droite).

Une ligne de commentaire (du programme) indique que la vitesse de communication est de 9600 bits/s.

Q22. Quelle doit être la vitesse pour le moniteur série ?

Q23. Donner la traduction de print :

Modifier le println dans votre programme en print.

Q24. En déduire la signification de println :

Donner la signification de l'abréviation ln :

Q25. Compléter le tableau ci-dessous :

Tension analogique d'entrée sur A0 en V	0	à	5
Variable numérique sensorValue affichée sur le moniteur série		à	

Q26. Compléter cette phrase :

La tension …… est convertie en ……. par un Convertisseur A…….. N…….. (CAN). On visualise le résultat de la CAN sur le moniteur série d'Arduino.

Q27. Donner l'équivalent de CAN en anglais (abréviation et signification) :

Q28. Dans quelle base de numération est la valeur affichée sur le moniteur série ?

Q29. Quelle est la valeur maximale ?

Q30. Convertir en binaire cette valeur :

On appelle "n" le nombre de bits de cette valeur.

Q31. Combien y a-t-il de bits (utiles) ?

Le nombre de bits "n" détermine la résolution du CAN. Donner la résolution du CAN de la carte Arduino :

Q32. Dans quel composant se trouve le CAN de la carte Arduino ?

Dans un CAN on appelle quantum la plus petite variation de la grandeur analogique d'entrée qui va provoquer une variation de 1'unité du code de sortie (le LSB). A chaque variation de la tension d'entrée Ve de $\pm q$, on a le nombre N en sortie du CAN qui varie de ± 1 LSB :

Exemple (avec $n = 4$) :

Pour $Ve = 0V$ $N = 0\,0\,0\,0$, Pour $Ve = 0V + q$ $N = 0\,0\,0\,1$

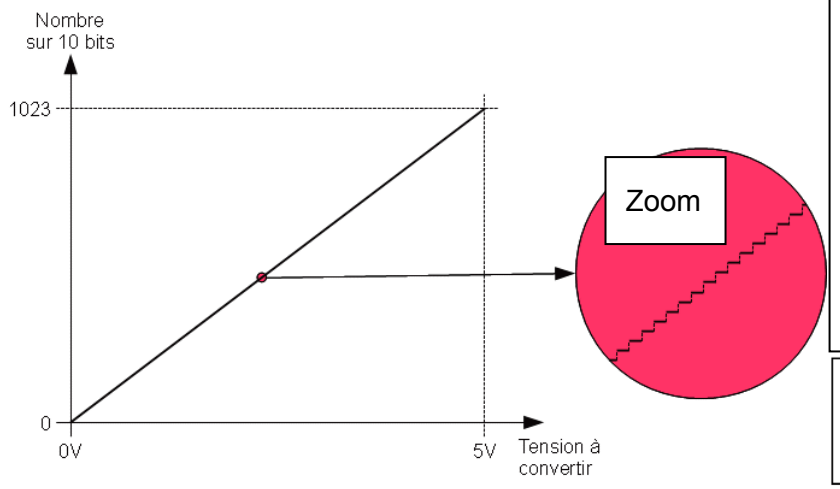
Le quantum (ou pas de progression) est donné par :

$$q_{(V)} = \frac{V_{REF}}{2^n - 1}$$

La valeur, par défaut, de V_{REF} est de 5V.

Q33. Calculer la valeur du quantum pour le CAN de la carte Arduino (préciser l'unité) :

Caractéristique $N = f(Ve)$:



V_e : tension en en entrée

N : résultat de la CAN

La formule est la suivante :

$$N = V_e \times \frac{1023}{5}$$

Pour :

$$V_e = 0 \rightarrow N = 0$$

$$V_e = 5 \rightarrow N = 1023$$

La formule est la suivante :

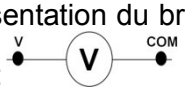
$$N = V_e \times \frac{1023}{5}$$

Q34. Placer le quantum "q" sur la figure ci-dessus.

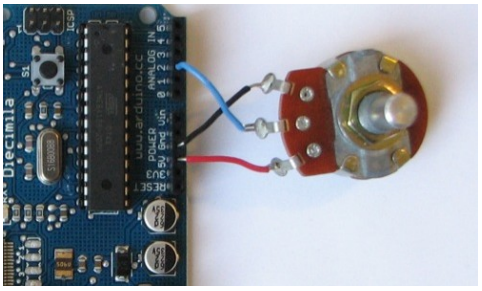
Q35. Quel appareil permet de mesurer une tension :

Pour mesurer une tension continue (selon le modèle de l'appareil) :

- faut-il se mettre en position AC ou DC :
- faut-il sélectionner le symbole de gauche ou de droite  :

La représentation du branchement de l'appareil est la suivante : 

Représenter (avec des flèches de couleurs) le branchement de l'appareil sur la photo ci-contre (on pourra se mettre directement sur le potentiomètre) pour mesurer la différence de potentiels (ddp) entre l'entrée de la tension analogique à convertir et la masse en respectant les couleurs, V en **rouge** pour le signal analogique et COM en **noir** pour la masse (GND).



Q36. Compléter le tableau ci-dessous :

Tension analogique d'entrée sur A0 en V	0	1	2	3	4	5
Variable numérique sensorValue théorique	0					1023
Variable numérique sensorValue affichée sur le moniteur série						
Variable numérique sensorValue en binaire						

III. Revenons au programme "AnalogInput":

Q37. Donner les valeurs (mini et max) lors de l'exécution de la ligne delay(sensorValue) :

Réécriture du programme "AnalogInput":

Q38. Rappeler la signification de l'abréviation int :

Q39. Rappeler la traduction de l'abréviation int :

Q40. Donner la valeur max et min d'une variable de type int

Q41. Donner la valeur max et min de la variable "sensorValue" :

Q42. Rappeler la signification de l'abréviation char :

Q43. Donner la valeur max et min d'une variable de type char :

Q44. Convertir en décimal la valeur de "sensorPin" (voir programme " AnalogInput") :

Q45. En déduire le type de variable que l'on devrait utiliser pour la variable "sensorPin" :

"sensorPin" est en fait une constante, on pourrait rajouter "const" devant la déclaration ou également utiliser un #define

Ce programme est encore modeste, seules les broches A0 et 13 sont utilisées. On pourrait imaginer qu'il y en ait d'autres et que le programme soit beaucoup plus long. Il devient alors difficile de se souvenir à quoi correspond telle ou telle broche. Il vaut mieux alors donner un nom au numéro des broches en créant une équivalence :

#define led 13 // déclaration d'une équivalence avec un nom "pertinent" (ne pas mettre de ;)

La ligne (#define led 13 // déclaration d'une équivalence avec un nom "pertinent") est à mettre après les commentaires */ et avant le void setup()

Effectuer les modifications ci-dessus (type de variable et #define), rajouter un #define pour "sensorPin", (avec le #define, le logiciel associe tous les noms led à 13 et sensorPin à A0).

De plus dans un souci de lisibilité et de compréhension, mettre les accolades des fonctions setup et loop alignées (voir le fichier présentation des cartes arduino). Vérifier le bon fonctionnement de votre programme.

Appeler le professeur pour vérifier le programme ainsi modifié : (sauvegarder le)

Réalisation d'un voltmètre :

Sélectionner dans le MD "Exemples" puis "01.Basics" le fichier "ReadAnalogVoltage".

Q46. Donner la traduction de Attach the center pin of a potentiometer to pin A0, and the outside pins to +5V and ground :

Q47. Donner la couleur du fil du potentiomètre à relier au +5V :

Q48. Donner la couleur du fil du potentiomètre à relier au 0V :

Téléverser le programme (bouton en dessous de la barre des MD) dans la carte Arduino.

Q49. Observer (sur le moniteur série) ce qui se passe :

Q50. Rappeler la signification de l'abréviation float :

Remplacer la ligne : float voltage = sensorValue * (5.0 / 1023.0); par :

float voltage = sensorValue * (5 / 1023);

Q51. Le fonctionnement est-il toujours correct ?

Remarquer la "déclaration" des variables directement dans la fonction "loop".

Proposer une modification du programme pour visualiser le résultat de la CAN (avant la tension),

Q52. Insérer ici la ligne supplémentaire :

Réécriture du programme "ReadAnalogVoltage ":

Effectuer les modifications (#define, bonne déclaration des variables, affichage du résultat de la CAN).

De plus dans un souci de lisibilité et de compréhension, mettre les accolades des fonctions setup et loop alignées (voir le fichier présentation des cartes arduino). Vérifier le bon fonctionnement de votre programme.

Appeler le professeur pour vérifier le programme ainsi modifié : (sauvegarder le)

Fading par lecture analogique :

Recopier et sauvegarder le programme ci-dessous :

```
int sensorPin = A0;    // pin that the sensor is attached to
int ledPin = 13;        // pin that the LED is attached to

// variables:
int sensorValue = 0;         // the sensor value

void setup() {
    sensorValue = analogRead(sensorPin);
  }

void loop() {
  // read the sensor:
  sensorValue = analogRead(sensorPin);

  // apply the calibration to the sensor reading
  sensorValue = map(sensorValue, 0, 1023, 0, 255);

  // fade the LED using the calibrated value:
  analogWrite(ledPin, sensorValue);
}
```

Téléverser le programme (bouton en dessous de la barre des MD) dans la carte Arduino.

Mettre en commentaire la ligne "sensorValue = map(sensorValue, 0, 1023, 0, 255) ;"

Tourner le potentiomètre (à droite ou à gauche) pour éteindre la led. Tourner doucement l'axe du potentiomètre jusqu'à l'autre butée : observer (sur la carte !!) ce qui se passe :

L'aide d'Arduino donne : map(value, fromLow, fromHigh, toLow, toHigh)

"from" signifie "**de**" et "to" "**vers**".

La fonction "map" permet une mise à l'échelle. La fonction "analogRead" a un paramètre compris entre 0 et 1023 tandis que la fonction "analogWrite" a un paramètre compris entre 0 et 255. La fonction "map" va permettre par un produit en croix de passer **de** 0 à 1023 **vers** 0 à 255 et ce proportionnellement.

Réécriture du programme "fading_analogique":

Effectuer les modifications (#define, bonne déclaration des variables).

De plus dans un souci de lisibilité et de compréhension, mettre les accolades des fonctions setup et loop alignées (voir le fichier présentation des cartes arduino). Vérifier le bon fonctionnement de votre programme.

Appeler le professeur pour vérifier le programme ainsi modifié : (sauvegarder le)