

ACQUISITION ET TRAITEMENT DES DONNÉES

TP 1

ACQUISITION



charles.meunier@u-bourgogne.fr

OBJECTIFS



- Découvrir l'environnement de simulation Wokwi
- Comprendre les bases du fonctionnement d'un microcontrôleur
- Passer de la simulation au monde réel

MAIS AVANT DE COMMENCER...

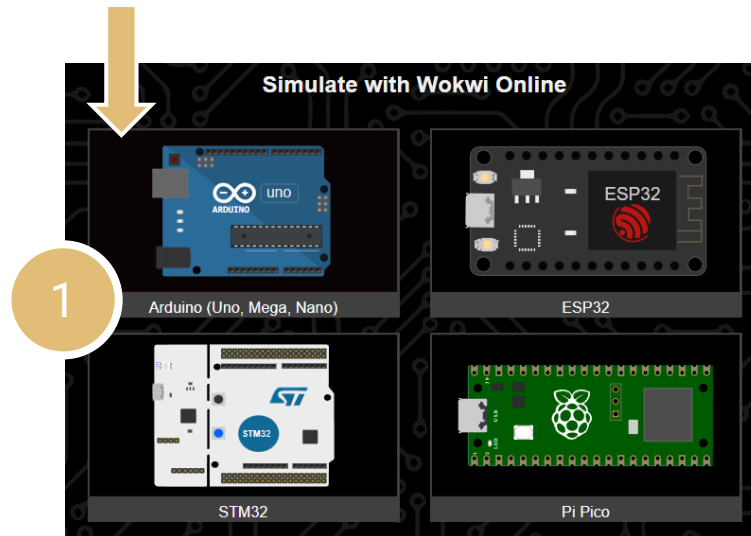
... il est important de définir quelques éléments pour la bonne compréhension de ce qui suit. Effectuez quelques recherches (oui, ChatGPT est autorisé ici) pour répondre aux questions suivantes :

- Qu'est-ce qu'un microcontrôleur ?
- Qu'est-ce qu'un programme ?
- Comment s'exécute un programme sur un microcontrôleur ?
- Qu'est-ce qu'une voie série ?

WOKWI

WOKWI est un environnement de simulation en ligne permettant de réaliser des circuits de test à base de microcontrôleur et de composants simples.

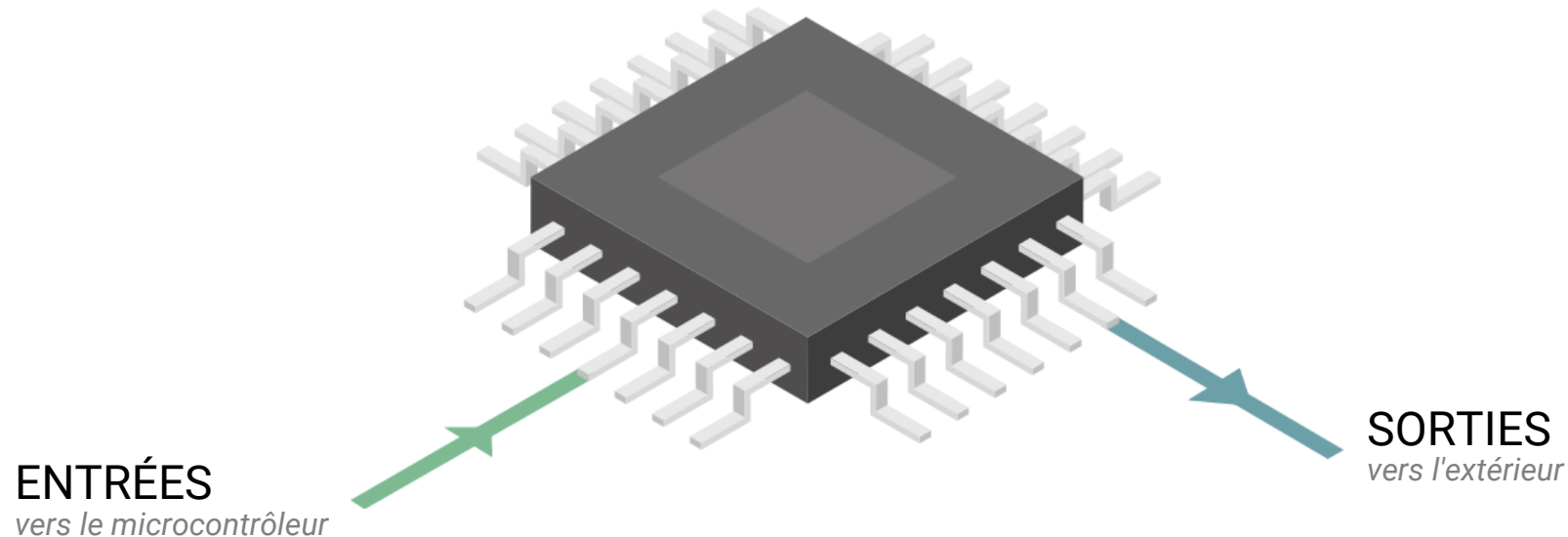
- Ouvrez un navigateur et connectez vous au site wokwi.com
- Créez un compte
- Créez un nouveau projet à base d'Arduino Nano :



ENTRÉES / SORTIES

Un microcontrôleur seul n'est pas très utile. En revanche, s'il peut communiquer avec le monde extérieur, ses possibilités deviennent presque infinies.

Pour cela, des entrées lui permettent de gérer les signaux en provenance de l'extérieur et que le microcontrôleur va pouvoir traiter. Par ailleurs, des sorties sont utilisées pour transmettre des informations ou des signaux vers d'autres composants.

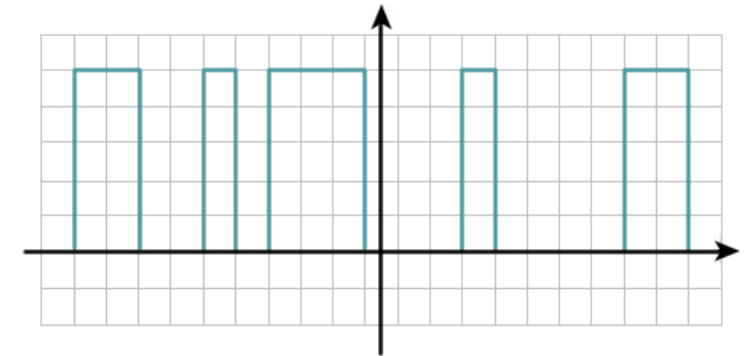


TYPES DE SIGNAUX

Deux types de signaux vont pouvoir être traités par un microcontrôleur :

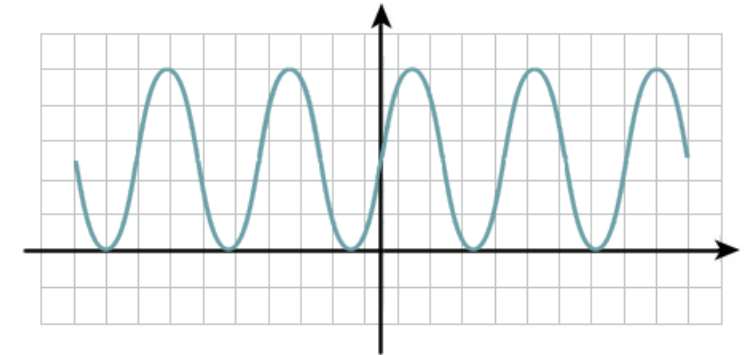
Signaux numériques

- Signal discret qui ne peut prendre que deux valeurs : HAUT (1) ou BAS (0)



Signaux analogiques

- Signal continu qui peut prendre une infinité de valeurs



- Identifiez les broches de la carte Arduino capable de gérer des signaux numériques et celles capables de gérer des signaux analogiques.

ÇA CLIGNOTE !

Vous avez, à présent, toutes les informations pour créer votre premier montage. L'objectif, ici est de faire clignoter une LED à une fréquence de 1Hz (la LED s'allume une fois par seconde).

Pour cela :

- Choisissez une broche adaptée pour connecter la LED
- Placez la broche choisie dans le bon mode (entrée ou sortie ?)
- Placez successivement la broche à l'état haut puis à l'état bas pour faire clignoter la LED

TIPS



Le courant ne peut traverser une LED que dans un sens

*Le mode de fonctionnement d'une broche se définit avec la fonction **pinMode***

*La fonction **delay(n)** permet de faire une pause de n millisecondes*

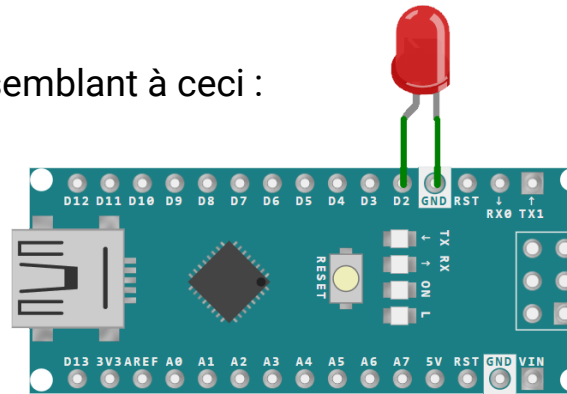


Les broches GND permettent de se raccorder à la masse du circuit

*Il est possible de changer l'état d'une broche numérique avec la fonction **digitalWrite***

PETIT RAPPEL DU LYCÉE

Vous avez certainement réalisé un schéma ressemblant à ceci :



Lorsqu'elle est alimentée dans le bon sens, la LED se comporte comme un fil avec une résistance quasi nulle.

- D'après la loi d'Ohm ($U = R \cdot I$), que pouvez-vous dire du courant qui traverse la LED dans la configuration ci-dessus ?
- Quel impact cela pourrait il avoir si le montage était réalisé hors simulation ?
- Comment peut on solutionner ce problème ?
- Appliquez la correction à votre schéma

UN BOUTON !

Nouvel objectif : allumer la LED lorsque l'utilisateur appuie sur un bouton.

Pour cela :

- Ajoutez un bouton à votre schéma.
- Connectez le bouton à une broche adaptée de la carte Arduino, dans une configuration Pull-Down.
- Modifiez votre code obtenir le comportement souhaité
- Testez le bon fonctionnement
- A quoi sert la configuration Pull-Down ?

TIPS

*La fonction **digitalRead** retourne l'état dans lequel se trouve une broche (**HIGH** ou **LOW**)*

COMMUNICATION SÉRIE

Il est parfois compliqué de savoir à quelle étape du programme se trouve le microcontrôleur. Est il bien rentré dans le test ? La bonne valeur a-t-elle été lue ? ...

Pour mieux comprendre ce qui se passe, nous pouvons demandé au microcontrôleur d'envoyer des messages sur la voie série.

- Modifiez votre programme précédent pour envoyer le message "Bouton appuyé" sur la voie série.
- Testez le bon fonctionnement.

TIPS

*La fonction **Serial.begin** permet de configurer la voie série. Une vitesse de 9600 bauds sera suffisante pour l'exercice.*

*La fonction **Serial.print** envoie une information sur la voie série*

*La fonction **Serial.println** envoie une information sur la voie série suivie d'un retour à la ligne.*

PREMIÈRES VARIABLES

Encore un nouvel objectif : afficher le nombre de fois où l'utilisateur a appuyé sur le bouton !

- Modifiez votre programme précédent pour envoyer le message "Bouton appuyé *n* fois" où *n* représentera le nombre d'appuis effectué sur le bouton.
- Testez le bon fonctionnement.

Vous allez très certainement être confronté à un problème de rebond.

- Expliquez ce phénomène
- Appliquer une solution dans votre code

TIPS

La fonction ***Serial.begin*** permet de configurer la voie série. Une vitesse de 9600 bauds sera suffisante pour l'exercice.

La fonction ***Serial.print*** envoie une information sur la voie série

La fonction ***Serial.println*** envoie une information sur la voie série suivie d'un retour à la ligne.

BONNE PRATIQUE : Prenez l'habitude de nommer vos variables avec un nom qui représente bien ce qu'elles contiennent.

PAS UN, NI DEUX MAIS TROIS BOUTONS !

Cette fois-ci on souhaite avoir trois boutons qui chacun agira différemment sur la LED :

- Le premier allumera simplement la LED comme précédemment
 - Le second fera clignoter la LED à 1Hz
 - Le troisième fera clignoter la LED à 5Hz
 - Si l'utilisateur appuie sur plusieurs boutons, rien ne se passe.
-
- Modifiez votre schémas et votre programme pour répondre à la demande ci-dessus.
 - Testez le bon fonctionnement.

TIPS

Vous pouvez affecter une touche du clavier à un bouton pour simuler l'appui de plusieurs boutons simultanément.

VOUS ÊTES PRÊT ? TOUSSER !

Nous allons utiliser une thermistance pour mesurer la température de la pièce. En fonction de la valeur mesurée, nous déclencherons le chauffage ou la climatisation.

Pour cela :

- Enregistrez votre projet et créez en un nouveau (toujours avec un Arduino Nano)
- Ajoutez une LED rouge qui symbolisera le chauffage et une LED bleue pour la climatisation
- Ajoutez un capteur de température (Analog Temperature Sensor)
- Ecrivez le programme qui envoie la température mesurée sur la voie série et qui allume la LED rouge si la température est inférieure à 18°C et qui allume la LED bleue si la température dépasse les 32°C.

TIPS

Comme son nom l'indique, la sonde de température fournit une signal analogique. Choisissez une broche adaptée.

*La fonction **analogRead** permet de convertir une valeur analogique en un entier compris entre 0 et 1023.*

Lorsque vous sélectionnez un composant, un point d'interrogation vert apparaît. En cliquant dessus vous accédez à sa documentation.

POUR DE VRAI ?

La simulation, c'est bien. Mais qu'est-ce que cela donne dans le monde réel. Et bien, il suffit d'essayer !

- Refaites le montage précédent, mais cette fois-ci sur breadboard avec un vrai Arduino, de vraies LED, de vraies résistances, une vraie sonde de température, ... vous avez compris l'idée.

Pas de panique, vos enseignants sont là pour vous aider :-)

!! WARNING !!

Les composants électroniques sont des éléments fragiles. Mauvais branchement, mauvaise tension ou courant et vous observerez un petit panache de fumée s'échapper de la petite bête. Pour éviter cela, soyez rigoureux et faites contrôler vos montages par votre enseignant.

TIPS

*L'environnement **Arduino** est installé sur les ordinateurs de l'école. Il est gratuit et vous pouvez l'installer sur votre matériel.*