Prise en main d'Arduino

1/ Pour démarrer

Le site de référence pour arduino est : http://www.arduino.cc/

On trouve beaucoup de sites traitant des arduino mais tous ne se valent pas. Essayer d'utiliser en priorité les informations du site de référence car elles sont fiables.

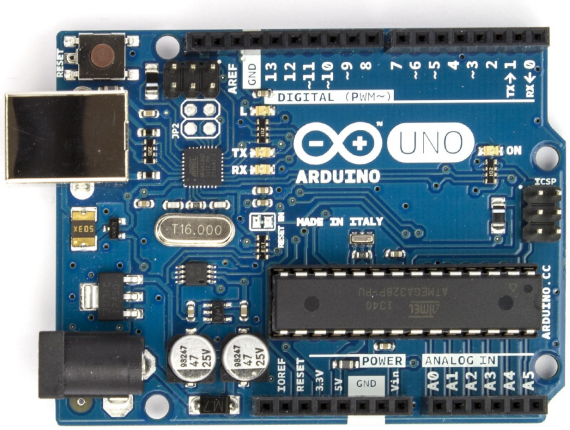
Installez l'environnement de développement :

linux : Dans un terminal $ sudo apt-get install arduino arduino-core

windows : voir http://arduino.cc/en/Guide/Windows

Si vous souhaitez un IDE plus complet vous pouvez utiliser Eclipse-Arduino :
http://www.baeyens.it/eclipse/

La carte utilisé est une arduino Uno basée sur un microcontrôleur ATMEGA328P cadencé à 16MHz.



Elle possède 32ko de mémoire flash pour vos programme, 2ko de mémoire RAM pour vos variables, 14 entrées/sorties logiques (dont deux réservées pour la communication avec le PC) et 6 entrées analogiques (également utilisable comme entrée/sortie logiques si nécessaire). **Les broches 0,1 de doivent pas être utilisées sous peine de perdre la communication USB avec le PC.**

Pour le brochage voir le document : **Arduino_uno_pinout.pdf**

2/ Faire clignoter une LED ! puis deux...

Une LED est câblée sur la broche 13 de la carte arduino Uno.

1/ Une méthode (trop) basique

Exécutez et étudiez le programme ***Exemples->01.Basics->Blink*** pour la faire clignoter. Relever à l'oscilloscope le signal sur la broche 13.

Câblez une Led externe avec la résistance de limitation du courant sur une autre broche. Faire un schéma. Calculez la valeur de la résistance pour votre LED.

Modifiez le programme pour la faire clignoter et changez le vitesse de clignotement.

Expliquez le rôle des fonctions : *setup(), loop(), pinMode(),digitalWrite()* et *delay()*

En fait l'utilisation de la fonction delay() pour des temporisations supérieures à quelques millisecondes est à éviter dans un programme réel car tout le programme est bloqué pendant la durée de l'attente. Il faut utiliser une autre méthode.

2/ La bonne façon de faire

Exécutez et étudiez le programme ***Exemples->02.Digital->BlinkWithoutDelay***

Expliquez le rôle de la fonction *millis()*

Expliquez alors l'algorithme de clignotement de la LED.

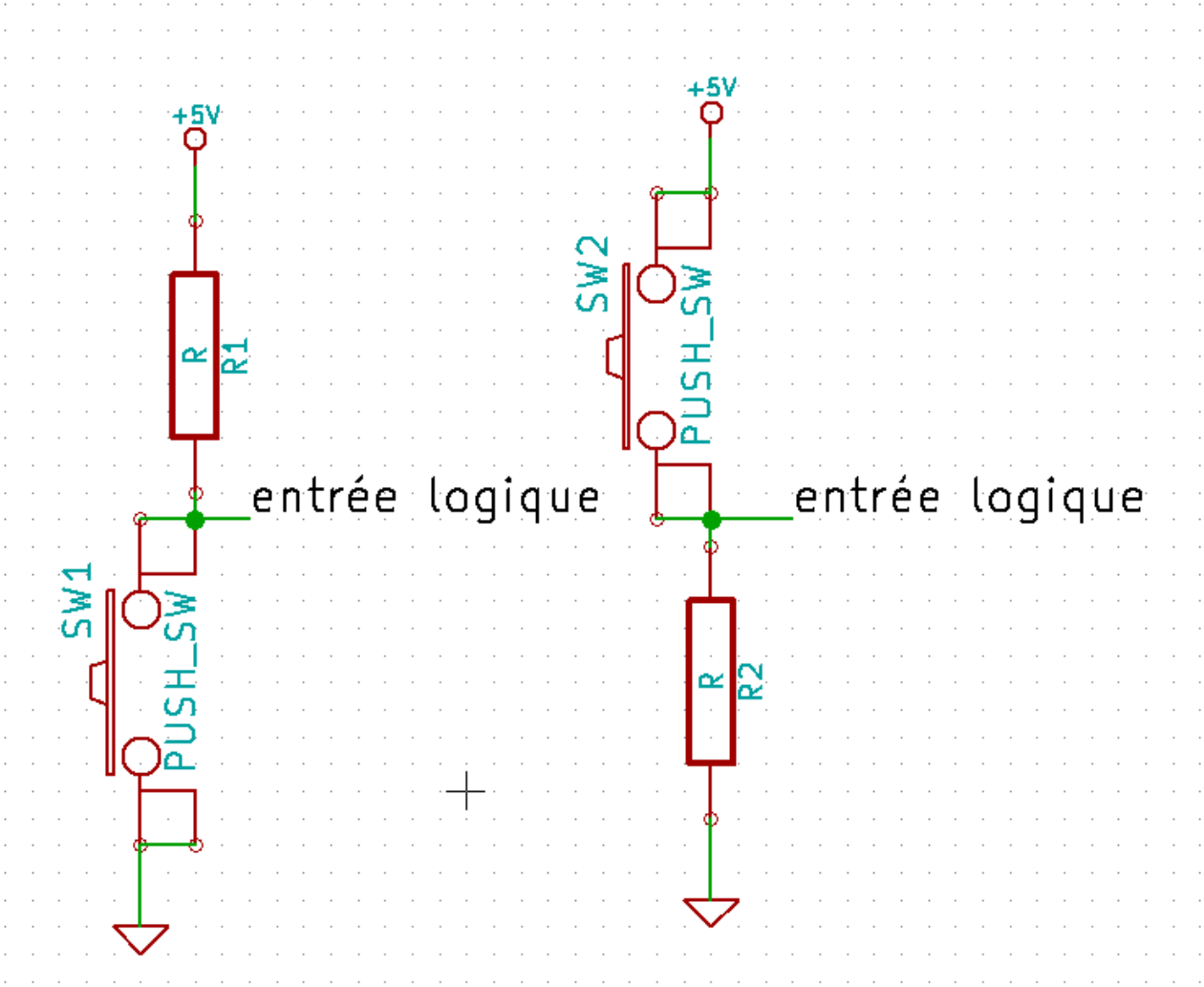
Modifier le programme pour faire clignoter deux LEDs, l'une avec un rythme de 400ms et l'autre à 3s.

On n'utilisera donc plus la fonction delay() pour temporiser des actions plus de quelques millisecondes.

(Si vous n'êtes pas convaincu essayer de faire le même programme avec des delay(). Si vous y arrivez, essayez avec trois LEDs à des rythmes différents...)

3/ Des boutons !

Il y a deux façons de câbler des boutons :



En pratique, la câblage avec mise à la masse lors de l'appui est le plus utilisé.

Pour chacun des cas donnez la valeur de l'entrée logique en fonction de l'état appuyé/relâché du bouton.

Pour le câblage de gauche, si on suppose que le courant dans l'entrée logique est au plus de 10µA, justifier le choix de R1 = 47kΩ, en considérant les deux états du bouton.

Câbler un bouton (câblage de gauche) sur une entrée logique de la carte arduino.

Exécutez et étudiez le programme ***Exemples->02.Digital->Button***

Modifier le si nécessaire pour que la Led s'allume lorsqu'on appuie sur le bouton

Expliquez le rôle de la fonction *digitalRead()*.

4/ Communication avec le PC

Pour échanger des informations avec un PC (ou un autre appareil, le microcontrôleur dispose d'un port série. Ce périphérique permet d'émettre et de de recevoir des octets. La vitesse en bits/s est fixée lors de l'initialisation. La broche d'émission TxD est reliée à la broche arduino 1 et la broche de réception RxD à la broche arduino 0. Sur la carte arduino ces signaux sont convertis au standard USB pour lequel la vitesse est bien plus élevée (plusieurs Mo/s) mais ici c'est la vitesse du port série qui limite le transfert.

Pour l'instant on étudie seulement la communication dans le sens Arduino vers PC (l'arduino émet).

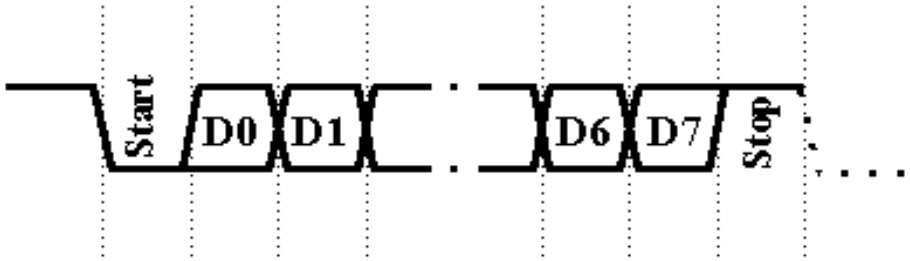
Exécutez et étudiez le programme ***Exemples->04.Communication->ASCIITable***

Expliquez le rôle des fonctions *begin(), print(),println()* et *write()* de la classe Serial.

Écrivez un programme qui émet la chaîne "bonjour" toutes les 5 secondes (avec la fonction millis()).

Quelle est la durée d'un bit à 9600 bits/s ?

L'état de repos de l'émetteur correspond à un niveau haut (+5V). Les bits sont transmis les uns après les autres (série). La transmission d'un octet démarre par un bit toujours à 0 (bit de START), puis se poursuit par les 8 bits en commençant par le poids faible (LSB first) et se termine par un bit toujours à 1 (bit de STOP). Après la transmission, l'émetteur retourne à l'état de repos 5V s'il n'y a plus d'octet à transmettre ou alors recommence un nouvelle transmission (bit de START, etc.…).



Écrivez un programme qui envoie le caractère **U** à 9600 bits/s toutes les 100ms (vous pouvez utiliser *delay*). Relevez le signal émis à l'oscilloscope et mesurez la durée d'un bit. Pourquoi avoir choisi ce caractère en particulier ?

Modifier le programme pour émettre le caractère **A**, relevez le signal et identifiez les différents bits de la transmission (START, b0-b7, STOP). Quelle est la durée totale de transmission d'un octet à 9600 bits/s ?

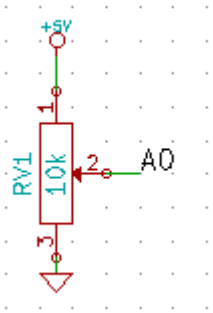
Écrivez un programme qui affiche "ON" ou "OFF" en fonction de l'état appuyé/relâché d'un bouton. Attention, il ne doit y avoir qu'un seul affichage à chaque changement d'état.

5/ Entrées analogiques

Le microcontrôleur ATMEGA328 contient un convertisseur analogique/numérique. Il peut convertir une tension analogique en une grandeur numérique qui va devenir une variable du programme.

Le programme peut ainsi prendre un compte les variations d'une tension analogique comme exemple celle fournie par un capteur de température, de pression, etc...

Câbler le point milieu potentiomètre sur l'entrée A0. Quelles sont les valeurs limites de la tension entre A0 et la masse pour les deux positions extrêmes du potentiomètre ? Quelle est la valeur de la tension lorsque le potentiomètre est en position médiane ?



Exécutez et étudiez le programme ***Exemples->03.Analog->Smothing***

Expliquez le rôle de la fonction *analogRead()*.

En relevant les valeurs retournées pour les deux positions extrêmes, en déduire la relation entre la tension analogique en Volts et la valeur numérique retournée par le convertisseur.

Modifier le programme pour afficher la valeur de la tension en Volts.

6/ Exercices

Écrivez un programme qui allume une LED si le potentiomètre est au dessus de sa valeur médiane et l'éteint au dessous. Pour éviter un clignotement de la LED autour de la position médiane on prendra un hystérésis de 10 bits.

Écrivez un programme qui fait clignoter un LED et dont la vitesse de clignotement dépend de la position d'un potentiomètre.

7/ Testez vos réflexes

Écrivez un programme qui allume une LED puis attend un appuie sur un bouton pour l'éteindre. Le programme affiche alors votre temps de réaction en millisecondes sur le terminal. Après cela un nouveau cycle repart au bout d'un temps aléatoire compris entre 2 et 5 secondes.