

Entrées analogiques : faire des mesures et utiliser des capteurs analogiques, utiliser les nombres à virgules avec la carte Arduino.



Ateliers Arduino

par X. HINAULT

www.mon-club-elec.fr



Tous droits réservés – 2012.

Ce document légèrement payant est soumis au droit d'auteur et est réservé à l'usage personnel.

Afin d'encourager la production de supports didactiques de qualité, ce document est légèrement payant.

La licence d'utilisation est attribuée pour un usage personnel uniquement, dans le cercle familial. Mise en ligne et diffusion non autorisées.

Si vous n'avez pas payé pour l'usage de ce document, soyez sympa, merci d'acheter votre exemplaire personnel ici : <https://monclubelec.dpdcart.com/>

Pour tout problème lié à l'utilisation de ce document, veuillez envoyer une copie ici : support@mon-club-elec.fr

Pour obtenir tout autres types de licence d'utilisation (enseignement, commercial, etc...), veuillez contacter l'auteur ici : support@mon-club-elec.fr

Vous avez constaté une erreur ? une coquille ? N'hésitez pas à nous le signaler à cette adresse : support@mon-club-elec.fr

Truc d'utilisation : visualiser ce document en mode diaporama dans le visionneur PDF. Navigation avec les flèches HAUT / BAS ou la souris.

En mode fenêtre, activer le panneau latéral vous facilitera la navigation dans le document. Bonne lecture !

Lancer également le logiciel Arduino et connecter votre carte Arduino afin de pouvoir tester au fur et à mesure les codes d'exemples !

1. Intro

L'objectif ici est :

- de comprendre la distinction entre analogique et numérique
- de comprendre le fonctionnement d'une broche analogique de la carte Arduino,
- d'apprendre à utiliser une résistance variable linéaire,
- d'apprendre à faire des mesures de tension avec Arduino et à calculer des tensions mesurées,
- d'apprendre à utiliser un capteur analogique linéaire de température et réaliser un thermomètre numérique,
- de découvrir le principe d'affichage graphique de courbes de mesure dans Processing
- d'utiliser un capteur analogique non linéaire : la photo-résistance.

... afin d'apprendre à faire des mesures avec une carte Arduino et à utiliser les valeurs obtenues dans des programmes.

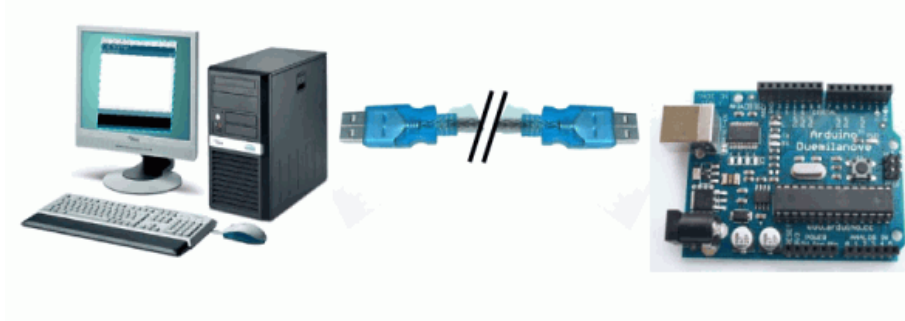


Prêt ? C'est parti !

2. Matériel nécessaire pour les ateliers Arduino

Pour cet atelier, vous aurez besoin de tout ou partie des éléments suivants pour pouvoir réaliser les exemples proposés :

De l'espace de développement Arduino

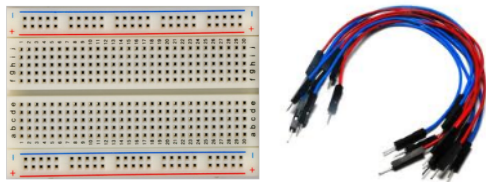


L'espace de développement Arduino associe :

- un ordinateur sous Windows, Mac Os X ou Gnu/Linux (Ubuntu)
- avec le logiciel Arduino installé (voir : <http://www.arduino.cc/>)
- un câble USB
- une carte Arduino UNO ou équivalente.

disponible chez : <http://shop.snootlab.com/> ou <http://www.gotronic.fr/>

Du nécessaire pour réaliser des montages sans soudure

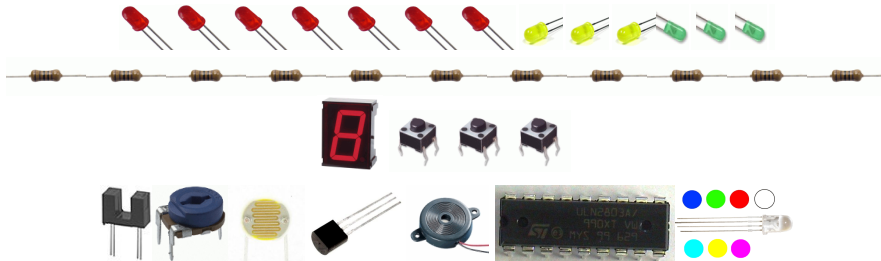


Pour réaliser des montages sans soudure, vous aurez besoin :

- d'une plaque d'essai ou breadboard moyenne (450 points)
- de quelques câbles souples (ou jumpers) mâle/mâle

disponible chez : <http://www.gotronic.fr/>

De quelques composants de base



Pour vous simplifier la vie, nous avons négocié ce kit pour vous !

Vous pouvez commander ce kit complet directement en 1 clic chez notre partenaire

<http://www.gotronic.fr/> avec le code express **701710**

GO TRONIC
ROBOTIQUE ET COMPOSANTS ÉLECTRONIQUES

Pour plus de détails, voir : http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.ATELIERS

Pour les ateliers Arduino niveau débutant, vous devrez idéalement disposer des composants suivants :

- des LEDs 5mm Rouges(x20), Vertes (x5) et 3 Jaunes (x5)
- digit à cathode commune rouge 13mm (x1)
- Résistances (1/4w - 5%) de 270 Ohms (x20), 4,7K Ohms (x1), 1K Ohms (x1)
- mini bouton-poussoir (x3)
- Opto-fourche (x 1)
- Résistance variable linéaire 10K (x 1)
- Photo-résistance 7mm (x 1)
- Capteur de température LM35DZ (-55/+150°C - 10mV/°C) (x 1)
- Capsule son piézoélectrique (x 1)
- ULN 2803A (CI amplificateur 8 voies, 500mA/ voie) (x 1)
- LED 5mm multicolore RVB cathode commune (x 1)

3. Rappel : Notion d'électronique numérique et analogique

L'électronique est une technique qui manipule les « électrons » sous forme de tension ou d'intensité. On distingue 2 types d'électronique :

- L'électronique **analogique** qui utilise des **variations continues** de la tension qui peut prendre toutes les valeurs intermédiaires (potentiomètre = variation du minimum au maximum).
- L'électronique **numérique** qui utilise des **variations « abruptes »** de la tension qui va prendre 2 niveaux (interrupteur = allumé ou éteint) : l'un dit HAUT (5V), l'un dit BAS (0V).

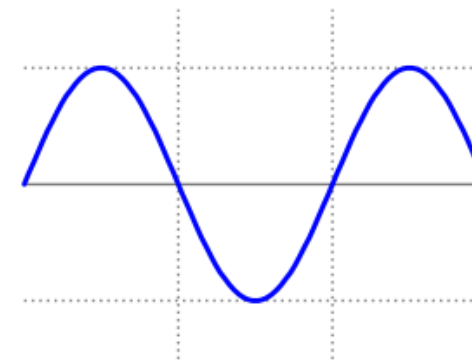
Un microprocesseur, tel que celui de la carte Arduino, est un circuit numérique qui va permettre de manipuler des niveaux HAUT/BAS.

Pour prendre une image de tuyauterie :

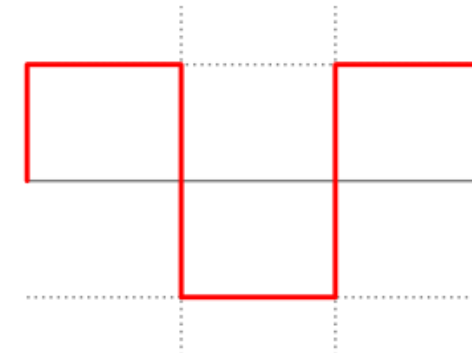
- L'électronique **analogique** est comparable à un **robinet** dont l'écoulement va pouvoir varier entre l'absence d'eau et le débit maximum.
- L'électronique **numérique** est comparable à une **vanne « tout ou rien »** qui va soit stopper l'écoulement, soit donner le débit maximum, les écoulements intermédiaires n'étant pas possibles.

L'intérêt majeur d'utiliser 2 niveaux HAUT/BAS est de permettre :

- De **compter et calculer** en combinant les niveaux HAUT et BAS entre-eux : c'est le comptage binaire, qui utilise les 0 et les 1,
- De **commander / contrôler** des dispositifs à partir de plusieurs niveaux HAUT / BAS combinés entre eux,
- De **communiquer** des informations entre 2 circuits numériques en envoyant des niveaux successifs de HAUT/BAS
- De **numériser** des signaux analogiques (conversion analogique-numérique) !
- De **coder des instructions** à exécuter par un microprocesseur.



Analogique



Numérique

4. Rappel : Une broche numérique ne peut avoir que 2 états : HAUT ou BAS, « y'a ou y'a pas » !

Une broche numérique, dans un circuit numérique est un point du circuit matérialisé par une broche métallique dans le cas d'un circuit intégré ou d'une carte électronique.

Une broche numérique va être caractérisée par son **état** ou niveau de tension : elle va pouvoir se trouver dans **2 états possibles seulement** :

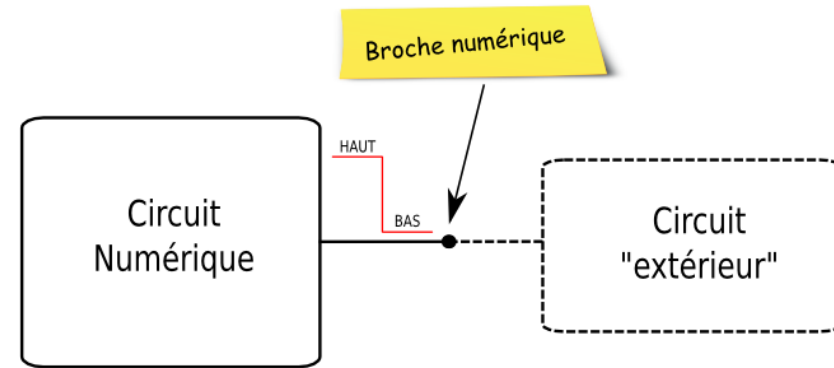
- soit au niveau **HAUT** (=5V), symbolisé par **1** ou **HIGH**
- soit au niveau **BAS** (=0V), symbolisé par **0** ou **LOW**
- noter qu'à **un instant quelconque, la broche se trouve obligatoirement dans l'un de ces 2 états.**

Pour reprendre l'image d'une vanne « tout ou rien » :

- soit il y a de l'eau qui circule
- soit il n'y en n'a pas

Pour reprendre l'image d'un « interrupteur » :

- soit il y a de la lumière,
- soit il n'y en n'a pas...



5. Rappel : Une broche numérique est caractérisée par son SENS : en SORTIE ou en ENTREE !

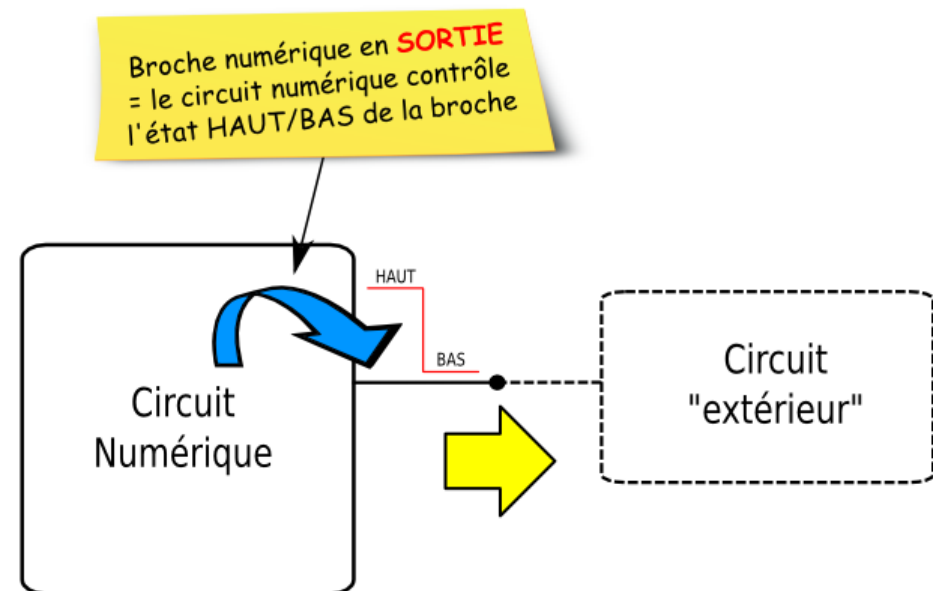
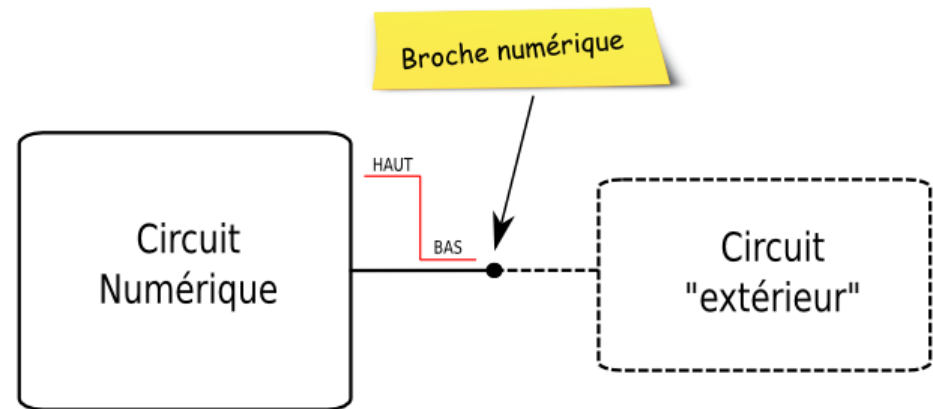
Une broche numérique est également caractérisée par son **sens** :

- la broche est dite en **sortie** d'un circuit numérique lorsque c'est **le circuit numérique qui contrôle l'état Haut/Bas de la broche**. On pourra symboliser le sens de la broche numérique en sortie par une flèche sortante.
- la broche est dite en **entrée** d'un circuit numérique lorsque le circuit numérique « reçoit » (ou « subit ») son état. **L'état Haut/Bas de la broche numérique est contrôlé par « l'extérieur »**. On pourra symboliser le sens de la broche numérique en entrée par une flèche entrante.
- noter qu'une broche numérique qui est en sortie d'un circuit numérique est en entrée du circuit extérieur auquel elle est connectée... et inversement... !

Usage avancé : en interne, dans un microprocesseur, une broche numérique est associée :

- à un bit de donnée (case unitaire mémoire) qui va permettre de fixer/lire son état
- à un bit de sens qui va définir son mode de fonctionnement

Techniquement, il existe par ailleurs plusieurs technologies de broches numériques (TTL, CMOS,..) qui ont des définitions différentes des niveaux de tension HAUT et BAS. Seules des broches compatibles entre-elles pourront être utilisées/connectées ensemble. Arduino est très souple de ce point de vue et est compatible avec la plupart des technologies E/S !

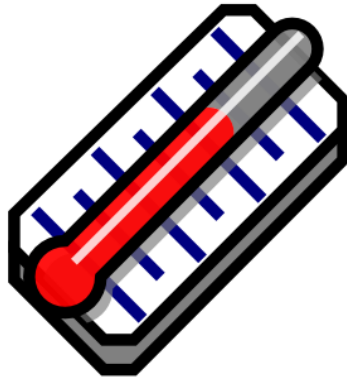


6. Le concept de « conversion analogique-numérique » (du monde physique vers le monde numérique !)

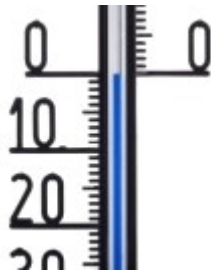
Faire une mesure, c'est quoi ?

On a tous mesuré quelque chose : une hauteur, un poids, un volume de liquide dans un verre doseur, une température avec un thermomètre, une tension avec un voltmètre, etc...

Quand on mesure, on fait quoi au juste ? Si on y réfléchit bien, **faire une mesure, c'est mettre en correspondance un état physique donné avec une graduation (ou échelle de mesure)**. Par exemple, lorsque l'on mesure une température, on associe la « hauteur » de liquide coloré dans le thermomètre avec la graduation qui est à côté.



Mais ce n'est pas tout : **pour mettre en correspondance l'état physique que l'on mesure et la graduation, on se base sur une valeur de référence** : généralement, on fixe précisément l'état physique qui correspond à la graduation 0. Par exemple, lorsque l'on mesure la longueur d'un segment, on va commencer par positionner précisément le 0 au niveau du début du segment. Pour une température, on se basera sur le 0°C pour fixer la mesure, etc...



En résumé

Si on se résume, pour faire une mesure, on a besoin :

- d'un phénomène physique à mesurer
- d'une échelle de mesure ou graduation
- d'une valeur de référence.

Faisons un rêve...

Revenons à notre Arduino : ce qui serait bien, mais alors très bien, c'est de pouvoir transformer avec notre Arduino, une tension variable appliquée sur une broche en un nombre qui serait utilisable comme une variable !

Imaginez une sorte de « **règle numérique à tension** » :

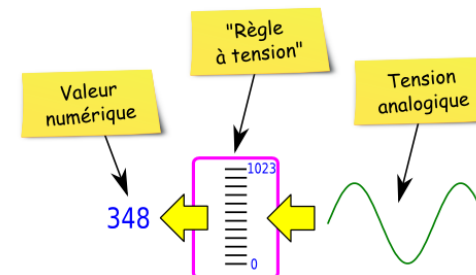
- on appliquerait une tension entre 0 et 5V par exemple sur une broche de la carte Arduino...
- et Arduino, à tout moment, pourrait nous dire quand on lui le demanderait quel est le niveau de tension sur la broche...

Mais faut pas rêver... Un truc pareil, ça doit coûter un max... ?!

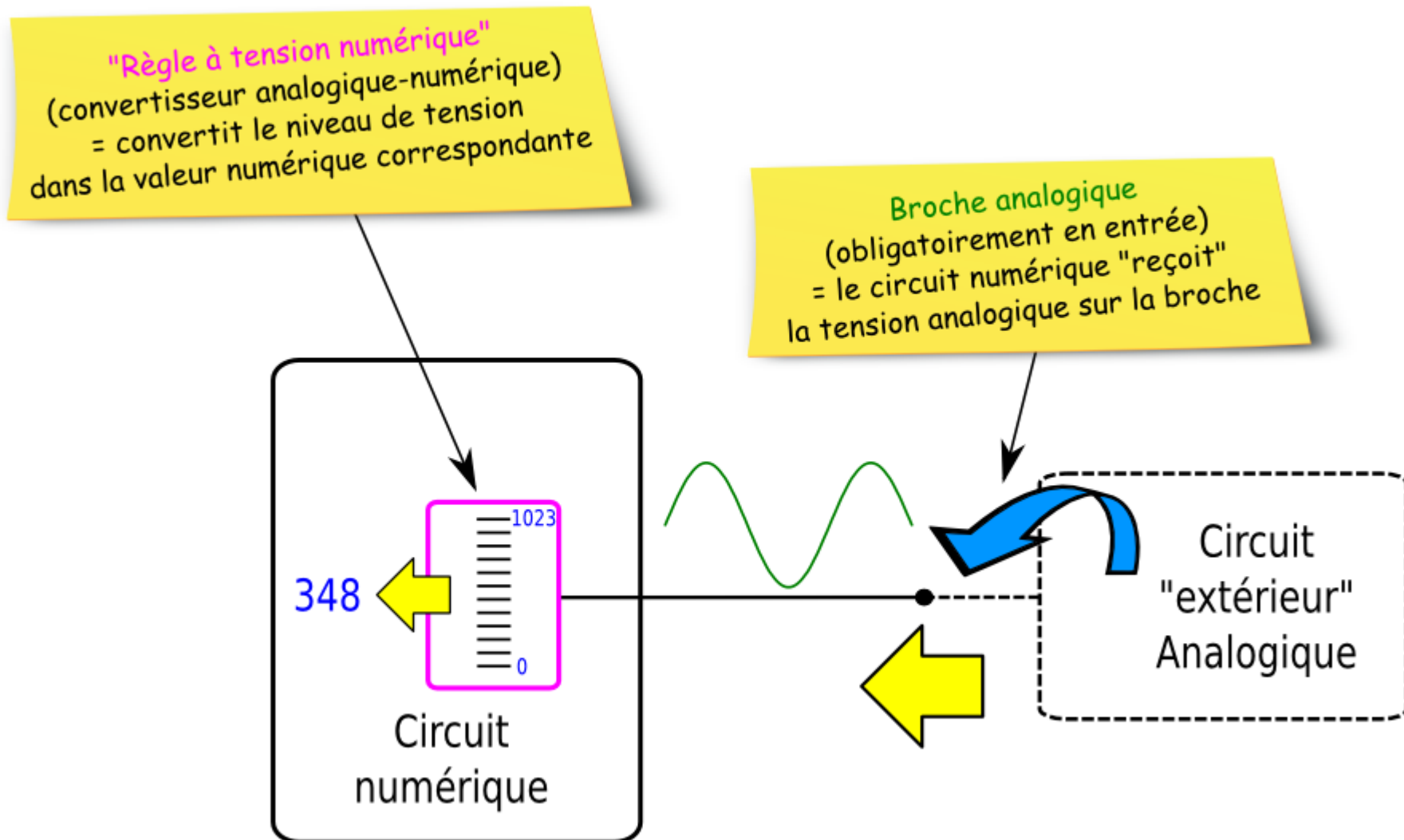
La conversion analogique-numérique...

Bon, alors là, j'ai une très bonne nouvelle : Arduino dispose en interne de ce que l'on appelle un module de « conversion analogique numérique » !

Autrement dit, une règle numérique à tension de 1024 niveaux : **il suffit simplement d'appliquer une tension entre 0 et 5V sur une des broches dite « analogique » pour obtenir une valeur entre 0 et 1023 correspondant au niveau de la tension présente sur la broche !**



7. Principe de fonctionnement d'une broche analogique



8. Les broches analogiques de la carte Arduino

La carte Arduino de base (la UNO, la Duemilanove, etc..) est une carte numérique qui possède **20 broches d'Entrée/Sortie numérique** (notées E/S) numérotées **de 0 à 19** !

Les broches 0 à 19 sont potentiellement utilisables en broches E/S !

Cependant, certaines broches ne doivent pas, dans la mesure du possible être utilisées en broches E/S :

- les **broches 0 et 1** sont utilisées par la communication USB donc, les utiliser pourrait perturber cette communication. En pratique, ne pas les utiliser.
- les **broches 14 à 19** ont un double rôle : elles peuvent également être utilisées en tant que broches analogiques pour réaliser des mesures. Donc, si possible, ne pas les utiliser en broches numériques... mais si on est obligé, on peut le faire !

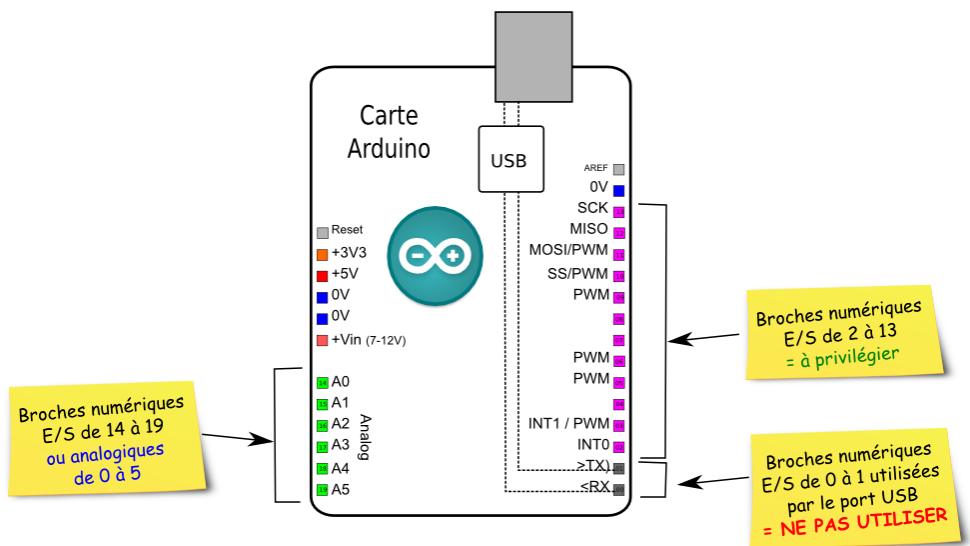
A savoir : les broches numériques 14 à 19 sont numérotées de 0 à 5 (ou désignées par A0, A1, A2, A3 et A5) lorsqu'elles sont utilisées en tant que broches analogiques.

Remarquer également que la plupart des broches numériques ont des fonctions particulières potentielles qui seront présentées au fur et à mesure de leur utilisation. *A titre indicatif, les fonctions disponibles sont la génération d'impulsion, la communication SPI, la communication I2C, les interruptions externes...*

D'un point de vue électrique, retenir que :

- chaque broche numérique E/S peut supporter 40 mA d'intensité en sortie ou en entrée
- L'ensemble des broches numériques E/S ne doit pas dépasser 200mA en entrée ou en sortie !

Usage avancé : pour des projets nécessitant de nombreuses broches E/S (= mal conçu?), la carte Arduino Mega dispose de plus d'une 50aine de broches E/S !



9. Pour info : les caractéristiques de la « règle à tension » numérique de l'Arduino

Voici quelques informations concernant la « règle à tension » numérique de la carte Arduino :

6 broches de mesure !

- disponible sur 6 broches dites analogiques : on peut donc utiliser 6 capteurs en même temps... (et même plus si on utilise un/des « multiplexeur analogique » tel que le CI 4051)

Plusieurs centaines de mesures par seconde !

- une mesure se fait en 100µs environ ce qui permet de réaliser sans aucune difficulté plusieurs centaines voir milliers de mesure par seconde (on peut transformer sa carte Arduino en un petit oscilloscope USB !)

Pleine échelle de mesure de 0 à 5V

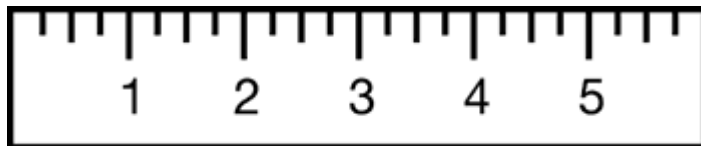
- la mesure est réalisée par défaut pour une tension comprise entre 0V et 5V. La broche ARef permet au besoin d'utiliser une autre tension de référence (2,5V par exemple), mais en pratique, c'est très peu utilisé.

Une précision de 5mV !

- la « règle à tension » dispose 1024 niveaux (ou graduations) soit une précision de 5mV environ en 5V !!

Résistance maximale de mesure = 10 KOhms

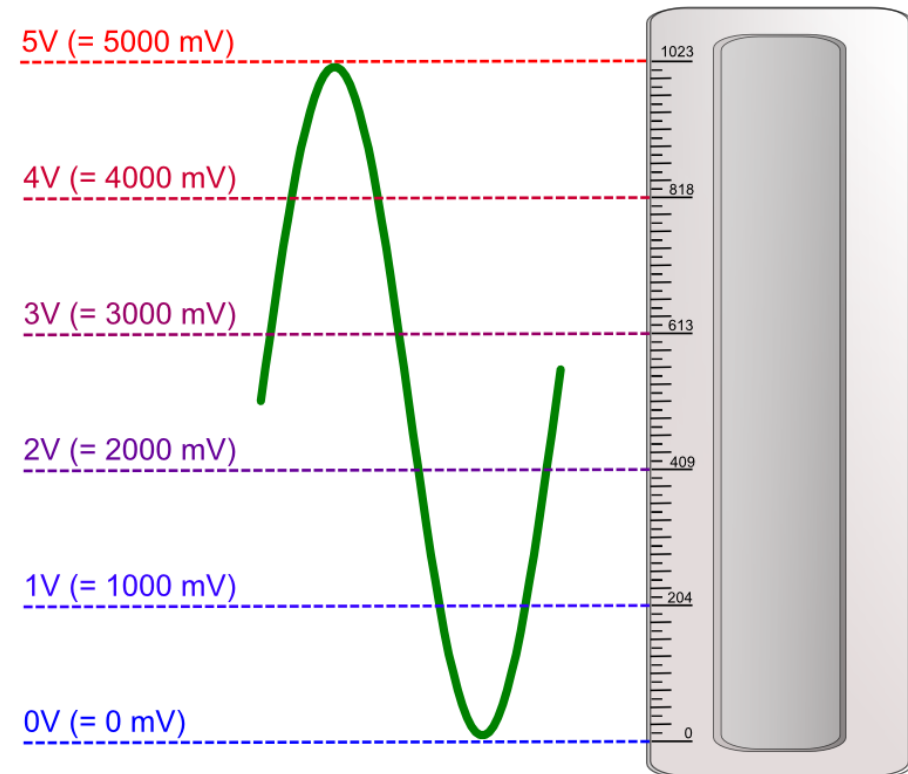
- pour éviter les erreurs de mesure, on mesurera des tensions sur des dispositifs ayant une résistance maximale de 10 KOhms . On dit que l'impédance d'entrée maximale conseillée = 10 KOhms



Correspondance entre la graduation et la tension mesurée

La « règle à tension » de l'Arduino, dispose de 1024 graduations numérotées de 0 à 1023 :

- la graduation 0 correspond au 0V ou 0mV
- la graduation 1023 correspond au 5V ou 5000mV
- la graduation 204 correspond au $204 \times 5000 / 1023 = 997\text{mV}$ soit 1V
- la graduation nnn correspond à $nnn \times 5000 / 1023 \text{ mV}$



Dans vos programmes, l'instruction `map(valeur, 0,1023,0,5000)` vous permettra de facilement réaliser la conversion.

10. Fiche composant : découvrir la résistance variable.

Description

Une résistance variable est une résistance... dont on va pouvoir faire varier la valeur. Vous utilisez déjà ce composant dans la vie de tous les jours : le bouton que l'on tourne pour régler le volume d'un appareil, c'est une résistance variable.

La résistance variable est caractérisée par sa résistance maximale, exprimée en Ohms. Elle dispose d'un axe de rotation qui permet de faire varier la résistance. A noter qu'il existe des résistances variables linéaires et logarithmiques : choisir les linéaires !

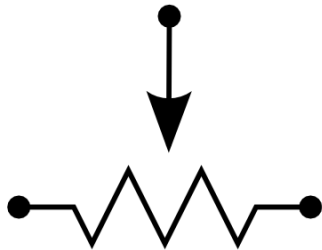
La résistance variable présente 3 broches :

- 2 broches correspondant aux broches de la résistance maximale interne, comme une résistance classique,
- 1 broche qui connectée au « curseur interne » qui permet de faire varier la résistance entre 0 Ohms et la résistance maximale.



Schéma théorique

Le schéma théorique d'une résistance variable est logiquement le suivant :

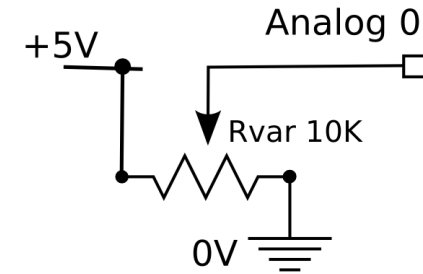


La résistance variable permet de simuler simplement un capteur !
Utiliser une valeur entre 1K et 10K Ohms avec Arduino.

Principe général d'utilisation

Le principe général d'utilisation d'une résistance variable avec une carte Arduino va consister :

- à connecter la résistance principale entre le 0V et le 5V
- à connecter la broche variable sur une entrée analogique.

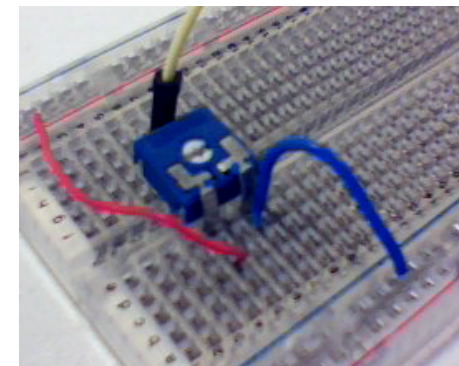


En procédant de cette façon, si on appelle R la valeur actuelle de la résistance variable, R_{max} sa valeur maximale et V_o la tension de sortie, on a la relation suivante : $V_o = 5V \times R/R_{max}$ (on a affaire à un diviseur de tension...).

Principe d'utilisation sur une plaque d'essai

Pour utiliser une résistance variable sur plaque d'essai, on la positionne à cheval sur le rail principal et on connecte :

- une des broches de la résistance max au +5V et l'autre broche de la résistance max au 0V,
- la troisième broche sera connectée à la carte Arduino sur une broche analogique.



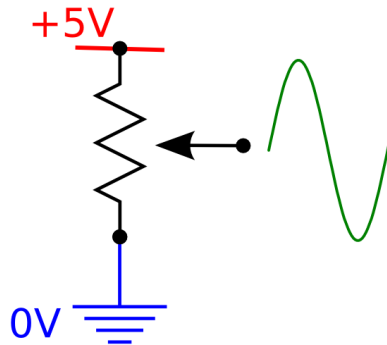
11. Affichage d'une mesure analogique dans le Terminal Serie : le montage

Principe d'utilisation

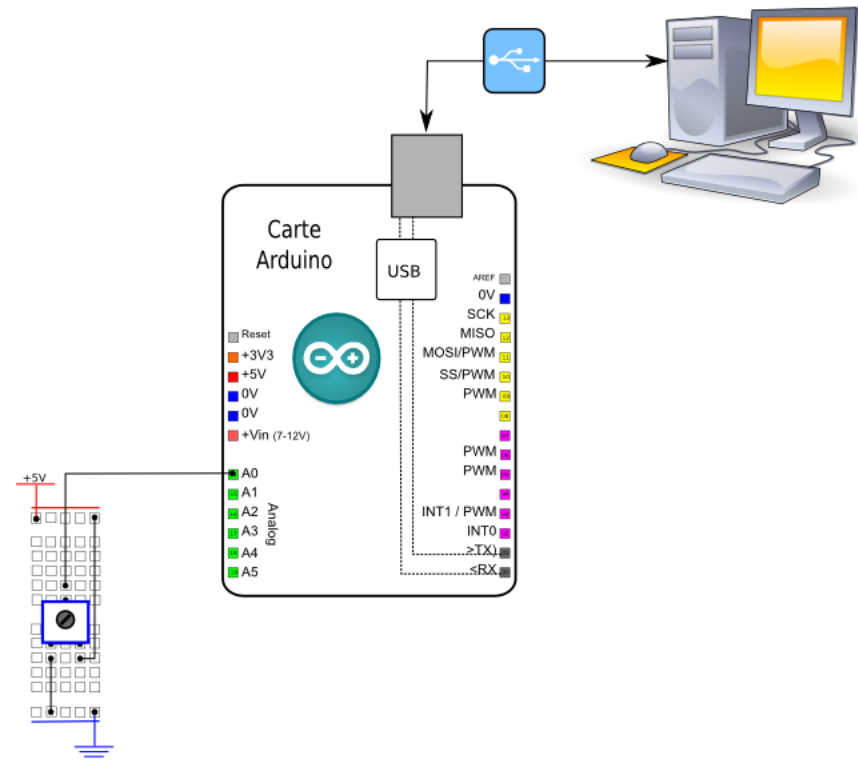
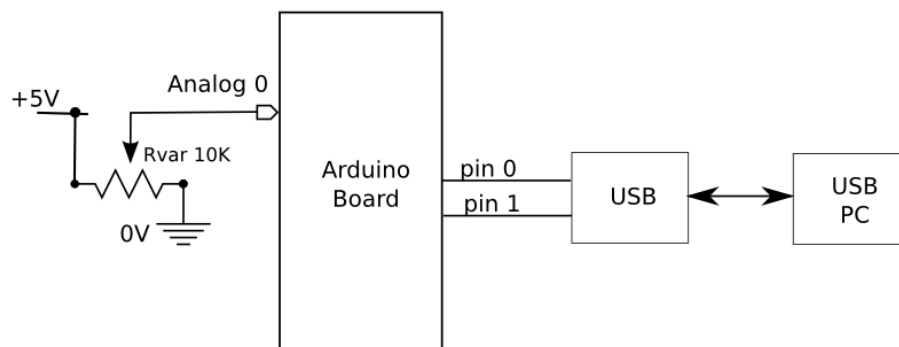
Le principe d'utilisation d'une résistance variable avec la carte Arduino est le suivant :

- on connecte les 2 broches de la résistance maximale interne au + 5V et au 0V,
- on connecte la broche du « curseur » interne (ou broche de sortie) sur une broche analogique de la carte Arduino : une des 6 broches AO à A5.

De cette façon, en faisant tourner l'axe de la résistance variable, on obtiendra une tension variant entre 0 et 5V sur la broche du « curseur » interne .



La carte Arduino, par ailleurs, sera connectée au PC via le câble USB : on affichera de cette façon les résultats de la mesure côté PC.



12. Affichage d'une mesure analogique brute dans le Terminal Serie

On va commencer par quelque chose de simple : récupérer la valeur brute (entre 0 et 1023) correspondant à la tension entre 0 et 5V présente sur une broche analogique.

Entete déclarative

On va déclarer à ce niveau :

- une constante de broche pour la voie analogique : **ATTENTION, il faut utiliser le numéro de la broche analogique (entre 0 et 5)** et pas le numéro de la broche numérique (entre 14 et 19..)
- on déclare une variable globale de type **int** pour stocker le résultat de la mesure.

Fonction **setup()**

- on initialise la communication série avec l'instruction **Serial.begin(vitesse)**. On utilisera 115200 bauds.

Fonction **loop()**

- on réalise une mesure à l'aide de la fonction **analogRead(brocheAnalogique)** : le résultat est stocké dans une variable.
- ensuite, on affiche le résultat dans le Terminal Série à l'aide de la fonction **Serial.println()**
- On réalise ensuite une pause d'une demi-seconde entre 2 mesures avec l'instruction **delay(500)**.

Fonctionnement du programme

- Ouvrir le Terminal Série (Tools > Serial Monitor) et fixer le débit à la même valeur que celle utilisée pour l'instruction **Serial.begin(vitesse)**. Ici, 115200 bauds.
- Toutes les secondes la valeur brute de la mesure s'affiche dans le Terminal Série : en faisant varier la valeur de la résistance variable, on observe que la valeur mesurée change entre 0 (à 0V) et 1023 (à 5V).

```
// --- constantes des broches ---

const int RVar=0; //declaration constante de broche analogique

// --- Déclaration des variables globales ---
int mesureBrute=0; // Variable pour acquisition résultat brut de
conversion analogique numérique

//***** FONCTION SETUP = Code d'initialisation *****

void setup() { // debut de la fonction setup()

Serial.begin(115200); // initialise connexion série à 115200 bauds
// IMPORTANT : régler le terminal côté PC avec la même valeur de transmi
ssion

} // fin de la fonction setup()

//***** FONCTION LOOP = Boucle sans fin *****

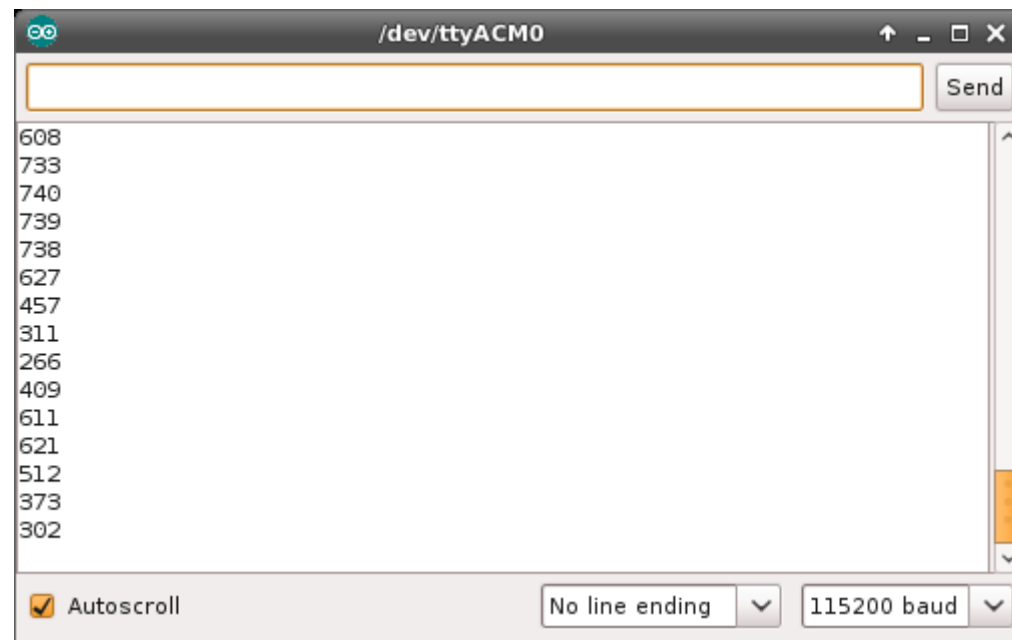
void loop(){ // debut de la fonction loop()

// acquisition conversion analogique-numerique (CAN) sur la voie analogi
que
mesureBrute=analogRead(RVar);

// affiche valeur numerique entière ou à virgule au format décimal
Serial.println(mesureBrute);

delay(500);

} // fin de la fonction loop() - le programme recommence au début de la
fonction loop sans fin
```



Purchased by Franck Ourion, franck.ourion@univ-lorraine.fr #6280170

Atelier Arduino : Entrées analogiques : faire des mesures et utiliser des capteurs analogiques, utiliser les nombres à virgules avec la carte Arduino.

13. Langage : Le type **float** pour stocker les valeurs numériques à virgule

Nous allons ici réaliser des opérations sur des nombres décimaux à virgule. Le langage Arduino dispose d'un type de variable particulier pour stocker de telles valeurs : le type **float**.

Quelques rappels au sujet des variables

En programmation, une « boîte mémoire » sur laquelle on colle une « étiquette » pour y mettre quelque chose : ça s'appelle une **variable** !

En programmation, la « taille » d'une « boîte mémoire » (ou variable) s'appelle le « **type** ». Pour dire les choses autrement, le type d'une variable, c'est sa catégorie, son genre .

Le type **int**

Vous connaissez déjà le type **int** (pour integer = entier) qui peut contenir une **valeur entière** comprise entre -32536 et + 32535 : le int est un double octet (16 cases unitaires).

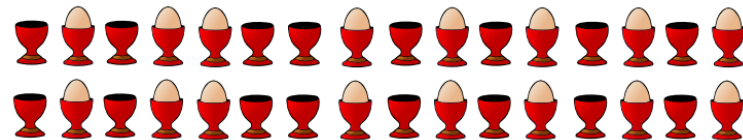
Le type int



Le type **long**

L'autre type que vous connaissez est le type **long** : le type **long** permet de stocker une **valeur entière** comprise entre -2 147 483 648 et +2 147 483 647 : le long est un quadruple octet (32 cases unitaires !)

Le type long



usage avancé : on fera précéder le type int ou long du mot clé **unsigned** si on souhaite n'utiliser que des valeurs positives. Ceci a pour effet de doubler la valeur positive utilisable (de 0 à 65535 pour un int par exemple).

Le type **float**

Le type **int** et le type **long** ne permettent de stocker que des valeurs numériques entières, pas les nombres à virgule.

Pour stocker des valeurs décimales à virgule, le langage Arduino dispose d'un type particulier appelé **float**.

Un **float** pourra contenir une valeur aussi élevées que 3.4028235E+38 et aussi basse que -3.4028235E+38.

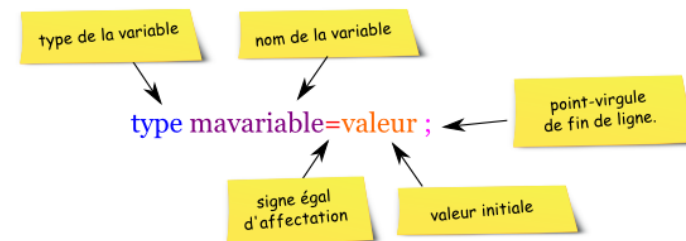
Techniquement un float est stocké sur 4 octets également.

Avec Arduino, pour des raisons techniques, la précision d'un float est tout de même limitée à l'utilisation de 6 à 7 chiffres en tout avant et après la virgule : au-delà, la précision sera perdue.

Déclaration d'une variable de type **float**

La déclaration d'une variable de type **float** se fait de la même façon que pour n'importe quel autre type de variable :

Déclaration d'une variable avec initialisation.



Par exemple :

```
float maVariable=0.0;
```

Le type **float** et les fonctions **print()** ou **println()**

Les fonctions **print()** et **println()** permettent de fixer le nombre de décimales affichées :

```
Serial.print(1.23456, 0); // affiche "1"
```

```
Serial.print(1.23456, 2); // affiche "1.23"
```

```
Serial.print(1.23456, 4); // affiche "1.2346"
```


14. Affichage d'une mesure analogique convertie en millivolts dans le Terminal Serie

A présent, on va calculer la valeur en millivolts correspondant à la valeur brute (entre 0 et 1023) correspondant à la tension entre 0 et 5V présente sur une broche analogique.

Entete déclarative

On va déclarer à ce niveau :

- une constante de broche pour la voie analogique : **ATTENTION, il faut utiliser le numéro de la broche analogique (entre 0 et 5)** et pas le numéro de la broche numérique (entre 14 et 19..)
- on déclare une variable globale de type **int** pour stocker le résultat de la mesure.
- on déclare une variable de type **float** pour stocker la valeur à virgule en millivolts.

Fonction **setup()**

- on initialise la communication série avec l'instruction **Serial.begin**(vitesse). On utilisera 115200 bauds.

Fonction **loop()**

- on réalise une mesure à l'aide de la fonction **analogRead**(brocheAnalogique) : le résultat est stocké dans une variable.
- on convertit la valeur brute en la valeur de la tension correspondante à l'aide de l'instruction **map()** qui réalise une règle de 3.
- ensuite, on affiche le résultat dans le Terminal Série à l'aide de la fonction **Serial.println()**
- On réalise ensuite une pause d'une demi-seconde entre 2 mesures avec l'instruction **delay**(500).

Fonctionnement du programme

- Ouvrir le Terminal Série (Tools > Serial Monitor) et fixer le débit à la même valeur que celle utilisée pour l'instruction **Serial.begin**(vitesse). Ici, 115200 bauds.
- Toutes les secondes un message s'affiche dans le Terminal Série : la valeur mesurée change entre 0 (à 0V) et 1023 (à 5V).

```
// --- constantes des broches ---
const int RVar=0; //declaration constante de broche analogique

// --- Déclaration des variables globales ---
int mesureBrute=0; // Variable pour acquisition résultat brut de
conversion analogique numérique
float mesure; // variable à virgule pour calcul valeur en millivolts

//***** FONCTION SETUP = Code d'initialisation *****
void setup() { // debut de la fonction setup()

Serial.begin(115200); // initialise connexion série à 115200 bauds
// IMPORTANT : régler le terminal côté PC avec la même valeur de transmi
ssion

} // fin de la fonction setup()

//***** FONCTION LOOP = Boucle sans fin *****
void loop(){ // debut de la fonction loop()

// acquisition conversion analogique-numerique (CAN) sur la voie analogi
que
mesureBrute=analogRead(RVar);

// calcule de la valeur a virgule en millivolts
// = convertit la valeur brute 0 - 1023 en valeur 0 - 5000 mV

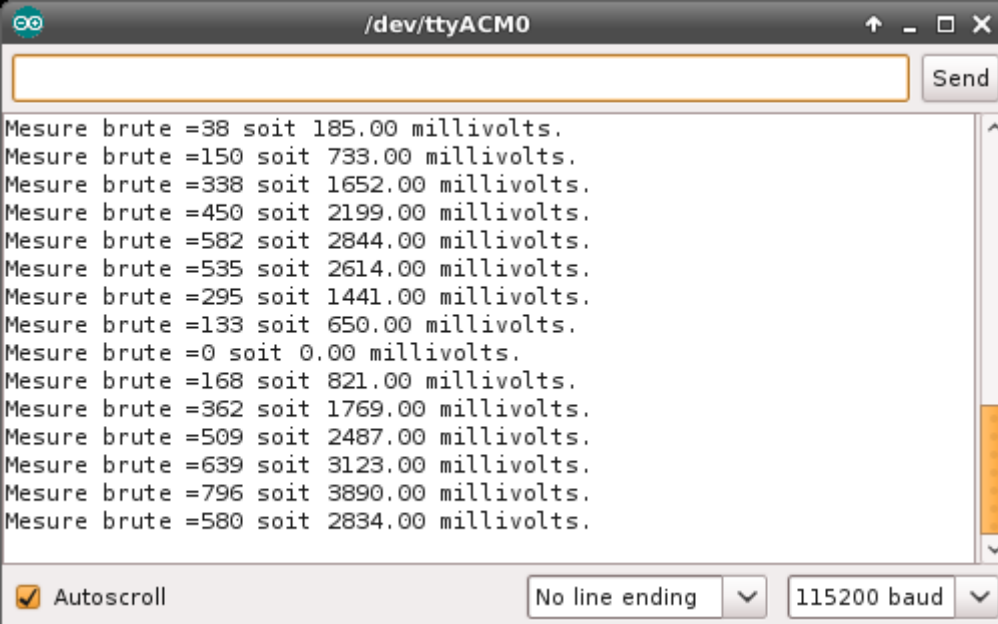
mesure=map(mesureBrute,0,1023,0.0,5000.0);

//---- calcul équivalent ----
//mesure=mesureBrute;
//mesure=mesure*5000.0;
// mesure=mesure/1023.0;

// affiche valeur numerique entière puis à virgule au format décimal
Serial.print("Mesure brute =");
Serial.print(mesureBrute);
Serial.print(" soit ");
Serial.print(mesure,2); // affichage avec 2 virgules
Serial.println(" millivolts.");

delay(500);

} // fin de la fonction loop() - le programme recommence au début de la
fonction loop sans fin
```

```
/dev/ttyACM0
Send
Mesure brute =38 soit 185.00 millivolts.
Mesure brute =150 soit 733.00 millivolts.
Mesure brute =338 soit 1652.00 millivolts.
Mesure brute =450 soit 2199.00 millivolts.
Mesure brute =582 soit 2844.00 millivolts.
Mesure brute =535 soit 2614.00 millivolts.
Mesure brute =295 soit 1441.00 millivolts.
Mesure brute =133 soit 650.00 millivolts.
Mesure brute =0 soit 0.00 millivolts.
Mesure brute =168 soit 821.00 millivolts.
Mesure brute =362 soit 1769.00 millivolts.
Mesure brute =509 soit 2487.00 millivolts.
Mesure brute =639 soit 3123.00 millivolts.
Mesure brute =796 soit 3890.00 millivolts.
Mesure brute =580 soit 2834.00 millivolts.
☒ Autoscroll
No line ending
115200 baud
```

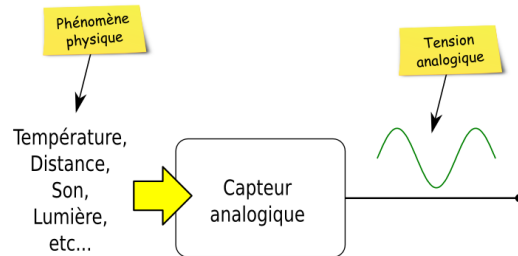
Purchased by Franck Ourion, franck.ourion@univ-lorraine.fr #6280170

Atelier Arduino : Entrées analogiques : faire des mesures et utiliser des capteurs analogiques, utiliser les nombres à virgules avec la carte Arduino.

15. Fiche technique : Principe d'utilisation d'un capteur analogique

Principe général

Un capteur analogique est un composant qui va transformer un phénomène physique (température, distance, etc...) en une tension variable fonction du phénomène physique mesuré.

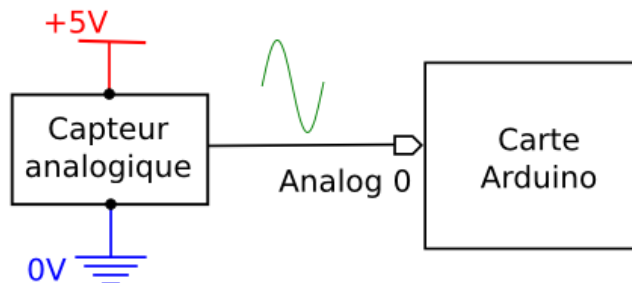


Brochage et Principe général d'utilisation avec Arduino

Un capteur analogique est un dispositif électronique et assez logiquement, il va disposer de 3 broches (à savoir : un capteur utilise que quelques mA) :

- 2 broches d'alimentation (typiquement +5V et 0V)
- une broche de sortie de la tension analogique, appelée typiquement Vo ou Vout.

Le principe général d'utilisation d'un capteur analogique est le même que celui que nous avons vu pour la résistance variable utilisée avec une carte Arduino : connexion des broches d'alimentation au 0V et au 5V et de la broche variable sur une entrée analogique.



Capteurs analogiques utilisables avec Arduino

Pour être utilisable avec une carte Arduino, il faudra :

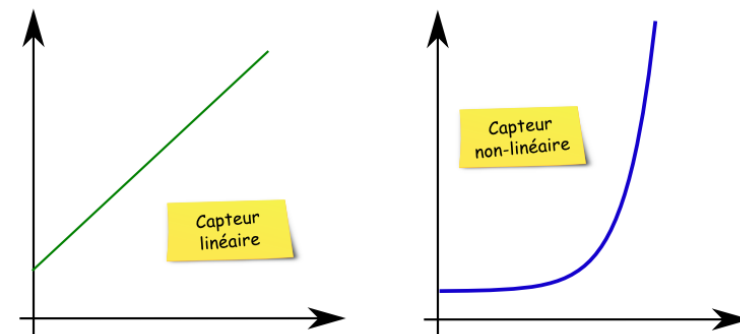
- que la tension de sortie d'un capteur analogique soit comprise entre 0 et 5V (c'est la plage de mesure de la « règle à tension » de l'Arduino)
- que la variation de tension soit grande par rapport à la résolution de 5mV/niveau (c'est la graduation minimale de la « règle à tension »)

La bonne nouvelle, c'est qu'il existe pleins de capteurs analogiques compatibles avec une carte Arduino !

Capteurs analogiques linéaires et non linéaires

Un point important à connaître : il existe 2 grands types de capteurs analogiques : les capteurs linéaires et les capteurs « non linéaires » :

- Les capteurs **linéaires** ont une tension de sortie strictement proportionnelle à la grandeur mesurée (par exemple 10mV par °C) et seront faciles à utiliser.
- Les capteurs **non-linéaires** ont une tension de sortie qui est fonction de la grandeur mesurée mais évolue selon une loi mathématique non linéaire (logarithme, exponentielle, etc...). Leur utilisation est plus compliquée et passe par un tableau d'étalonnage.



La courbe de la tension de sortie d'un capteur est fournie dans sa fiche technique : en pratique, préférer si possible des capteurs linéaires !

16. Exemples de capteurs analogiques utilisables avec une carte Arduino

Les capteurs analogiques utilisables avec une carte Arduino sont nombreux et variés ! Ils existent soit en composants individuels soit en mini-shields dédiés.

Capteurs analogiques linéaires

Voici quelques exemples :

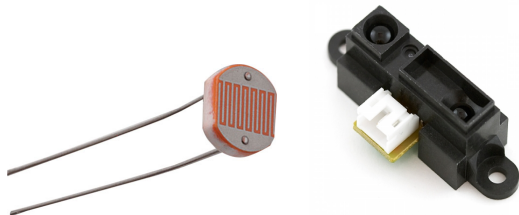
- Capteur de température (LM35)
- Capteur de luminosité ambiante
- Capteur de débit
- Capteur d'intensité
- Accéléromètre 2 ou 3 axes (mesure inclinaison)
- Capteur de niveau sonore
- etc....



Capteurs analogiques non-linéaires

Voici quelques exemples :

- Capteur de distance (GP2D12)
- Photorésistance (mesure la lumière)
- Humidistance (mesure humidité)
- etc...



De quelles informations a-t-on besoin pour utiliser un capteur analogique linéaire ?

Pour utiliser un capteur analogique linéaire, on a besoin de connaître :

- son brochage (broche +5V, 0V et Vo)
- sa plage de mesure
- la correspondance entre la grandeur mesurée et la sortie analogique en Volts
- une valeur de référence

Toutes ces informations sont fournies dans la fiche technique du capteur.

Un exemple concret

Par exemple, le capteur LM 35 est un capteur analogique linéaire de température :

- mesurant la température entre -55 et +150°C
- dont la sortie est à 0V à 0°C
- avec une variation de 10mV par °C
- alimentable en 5V

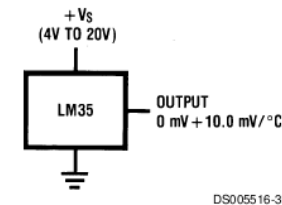


FIGURE 1. Basic Centigrade Temperature Sensor
(+2°C to +150°C)

Grâce à ces informations, on pourra utiliser précisément le capteur pour réaliser un thermomètre avec Arduino !

17. Fiche composant : le capteur analogique linéaire de température LM35

Description

Le capteur de température LM35 est un capteur de température linéaire fonctionnant en 5V.



Fiche technique : <http://www.mon-club-elec.fr/datasheet/LM35.pdf>

Brochage

3 broches : le 0V (GND), le +5V (+Vs) et la tension de sortie (Vout)



TL/H/5516-2

Order Number LM35CZ,
LM35CAZ or LM35DZ

Schéma fonctionnel

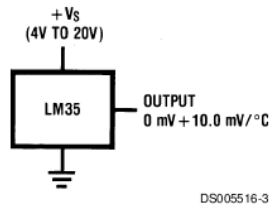


FIGURE 1. Basic Centigrade Temperature Sensor
(+2°C to +150°C)

Caractéristiques

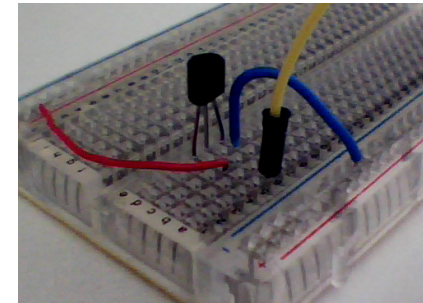
- mesurant la température entre -55 et +150°C
- dont la sortie est à 0V à 0°C
- avec une variation de 10mV par °C
- alimentable en 5V

Grâce à ces informations, on pourra utiliser précisément le capteur pour réaliser un thermomètre avec Arduino !

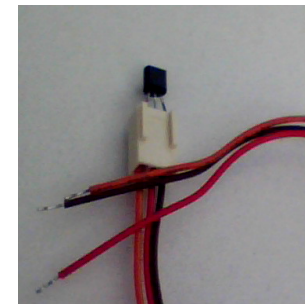
Exemple d'utilisation sur plaque d'essai avec Arduino

On connecte :

- la broche Vs au +5V
- la broche GND au 0V
- la broche Vout à une broche analogique de la carte Arduino.



Truc : on peut utiliser un connecteur 3 broches sur fils pour le montage !



18. Réaliser un simple thermomètre numérique avec affichage dans le Terminal Série : le montage

Principe d'utilisation du capteur LM 35

Le principe général d'utilisation d'un capteur analogique est le même que celui que nous avons vu pour la résistance variable utilisée avec une carte Arduino : connexion des broches d'alimentation au 0V et au 5V et de la broche variable sur une entrée analogique.

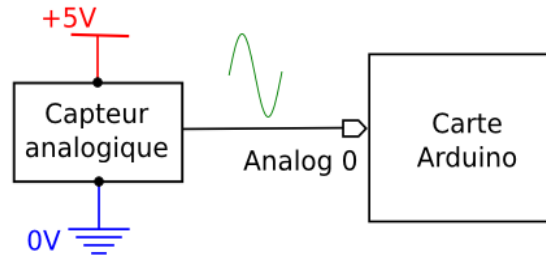
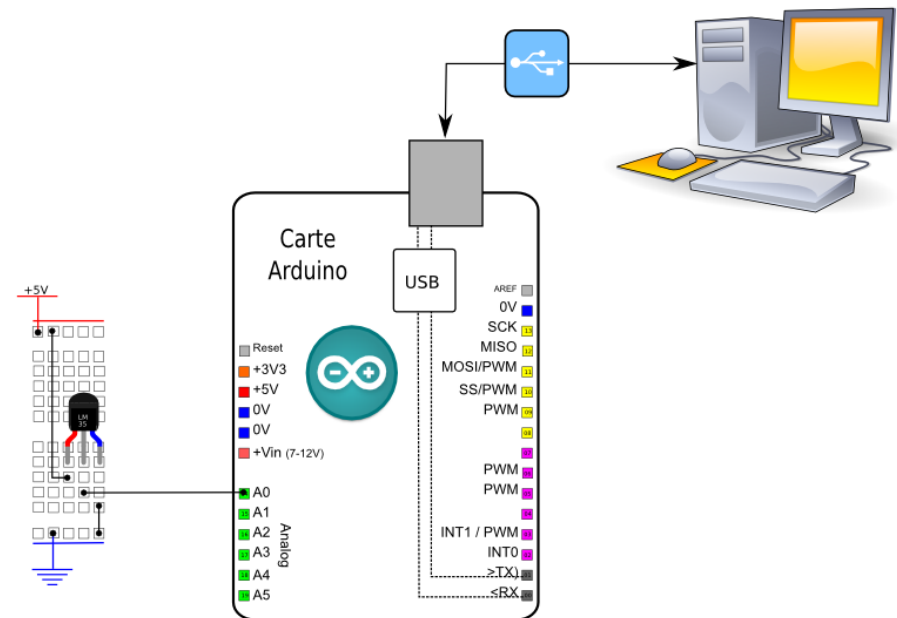
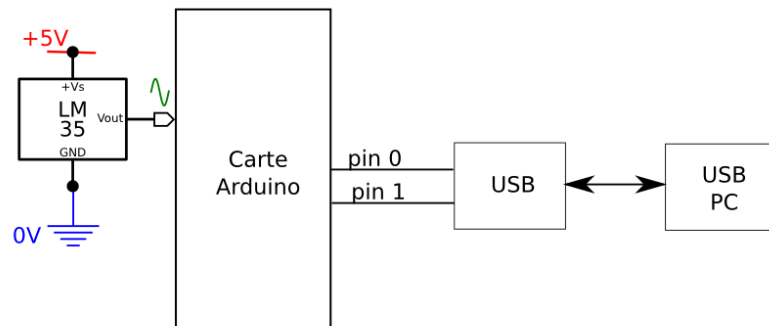


Schéma théorique du montage

On connecte :

- la broche Vs au +5V
- la broche GND au 0V
- la broche Vout à une broche analogique de la carte Arduino.

La carte Arduino, par ailleurs, sera connectée au PC via le câble USB : on affichera de cette façon les résultats de la mesure côté PC.



19. Réaliser un simple thermomètre numérique avec affichage dans le Terminal Série : le programme

On peut commencer par réutiliser les programmes déjà vus pour vérifier que les choses fonctionnent bien et afficher la mesure brute puis la mesure en millivolts.

Une fois fait, on peut convertir la valeur brute ou la tension en °C, ce qui dans le cas présent est assez simple : il suffit de prendre la tension en millivolt / 10 !

Entete déclarative

On va déclarer à ce niveau :

- une constante de broche pour la voie analogique : **ATTENTION, il faut utiliser le numéro de la broche analogique (entre 0 et 5)** et pas le numéro de la broche numérique (entre 14 et 19..)
- on déclare une variable globale de type **int** pour stocker le résultat de la mesure.
- on déclare une variable de type **float** pour stocker la valeur à virgule en millivolts puis en degrés.

Fonction **setup()**

- on initialise la communication série avec l'instruction **Serial.begin**(vitesse). On utilisera 115200 bauds.

Fonction **loop()**

- on réalise une mesure à l'aide de la fonction **analogRead**(brocheAnalogique) et on stocke dans une variable.
- on convertit la valeur brute en la valeur de la tension correspondante à l'aide de l'instruction **map()** qui réalise une règle de 3.
- on convertit la valeur en degrés par une simple division par 10 (tension de sortie = 0 +/- 10mv/°C).
- ensuite, on affiche le résultat dans le Terminal Série à l'aide de la fonction **Serial.println()**
- On réalise ensuite une pause d'une demi-seconde entre 2 mesures avec l'instruction **delay**(500).

Fonctionnement du programme

- Ouvrir le Terminal Série (Tools > Serial Monitor) et fixer le débit à la même valeur que celle utilisée pour l'instruction **Serial.begin**(vitesse). Ici, 115200 bauds.

```
// --- constantes des broches ---

const int RVar=0; //déclaration constante de broche analogique

// --- Déclaration des variables globales ---
int mesureBrute=0; // Variable pour acquisition résultat brut de conversion
analogique numérique
float mesure; // variable à virgule pour calcul valeur en millivolts

//***** FONCTION SETUP = Code d'initialisation *****

void setup() { // debut de la fonction setup()

Serial.begin(115200); // initialise connexion série à 115200 bauds
// IMPORTANT : régler le terminal côté PC avec la même valeur de transmissio
n

} // fin de la fonction setup()

//***** FONCTION LOOP = Boucle sans fin *****

void loop(){ // debut de la fonction loop()

// acquisition conversion analogique-numerique (CAN) sur la voie analogique
mesureBrute=analogRead(RVar);

// calcule de la valeur a virgule en millivolts
// = convertit la valeur brute 0 - 1023 en valeur 0 - 5000 mV

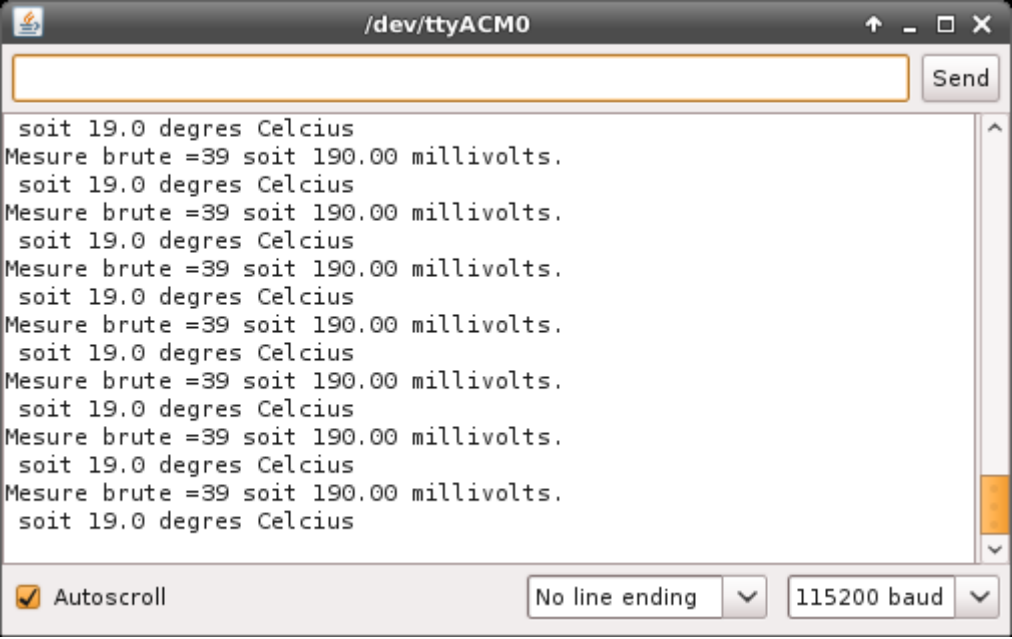
mesure=map(mesureBrute,0,1023,0.0,5000.0);

//---- calcul équivalent ----
//mesure=mesureBrute;
//mesure=mesure*5000.0;
// mesure=mesure/1023.0;

// affiche valeur numerique entière puis à virgule au format décimal
Serial.print("Mesure brute =");
Serial.print(mesureBrute);
Serial.print(" soit ");
Serial.print(mesure,2); // affichage avec 2 virgules
Serial.println(" millivolts.");
Serial.print(" soit ");
Serial.print(mesure/10,1); // affichage avec 2 virgules
Serial.println(" degres Celcius");

delay(500);

} // fin de la fonction loop() - le programme recommence au début de la
fonction loop sans fin
```



```
soit 19.0 degrees Celcius
Mesure brute =39 soit 190.00 millivolts.
soit 19.0 degrees Celcius
Mesure brute =39 soit 190.00 millivolts.
soit 19.0 degrees Celcius
Mesure brute =39 soit 190.00 millivolts.
soit 19.0 degrees Celcius
Mesure brute =39 soit 190.00 millivolts.
soit 19.0 degrees Celcius
Mesure brute =39 soit 190.00 millivolts.
soit 19.0 degrees Celcius
Mesure brute =39 soit 190.00 millivolts.
soit 19.0 degrees Celcius
Mesure brute =39 soit 190.00 millivolts.
soit 19.0 degrees Celcius
```

☒ Autoscroll No line ending 115200 baud

Purchased by Franck Ourion, franck.ourion@univ-lorraine.fr #6280170

Atelier Arduino : Entrées analogiques : faire des mesures et utiliser des capteurs analogiques, utiliser les nombres à virgules avec la carte Arduino.

20. Exemple d'affichage de résultat sous forme graphique : un mini-oscillo USB de base !

Présentation de Processing

Processing est un interface graphique programmable, libre et gratuite, écrite en Java (et donc multi-OS) qui permet de facilement réaliser des interfaces graphiques simples, de manipuler des images, de réaliser des graphiques de façon interactive à l'aide d'un langage simplifié, comme le langage Arduino.

En effet, l'interface Processing est capable d'interagir facilement avec le clavier, la souris, mais aussi la communication série USB, le réseau, etc...

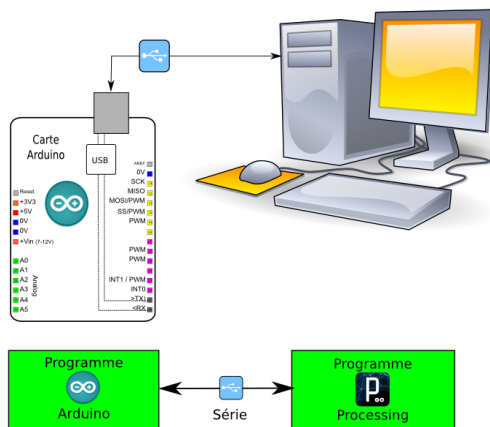
Pour installer et télécharger Processing : <http://processing.org/download/>

Vous êtes déjà familiarisé avec Processing sans le savoir : le logiciel Arduino est basé sur une interface Processing !

Principe de réalisation d'une interface graphique communiquant avec USB

Une fois que Processing est installé sur l'ordinateur, il devient possible de communiquer simplement par le port USB avec une carte Arduino elle-même programmée pour par exemple envoyer des données sur le port Série.

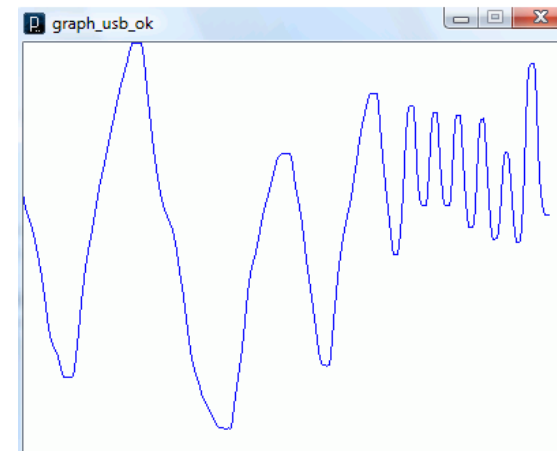
De cette façon, la carte Arduino couplée à l'interface Processing permet de réaliser simplement des applications de type oscilloscope de base ! Il suffit pour cela de programmer la carte Arduino pour envoyer le résultat de la mesure sur le port Série et d'écouter le port Série dans l'interface Processing pour afficher un point correspondant à la valeur reçue.



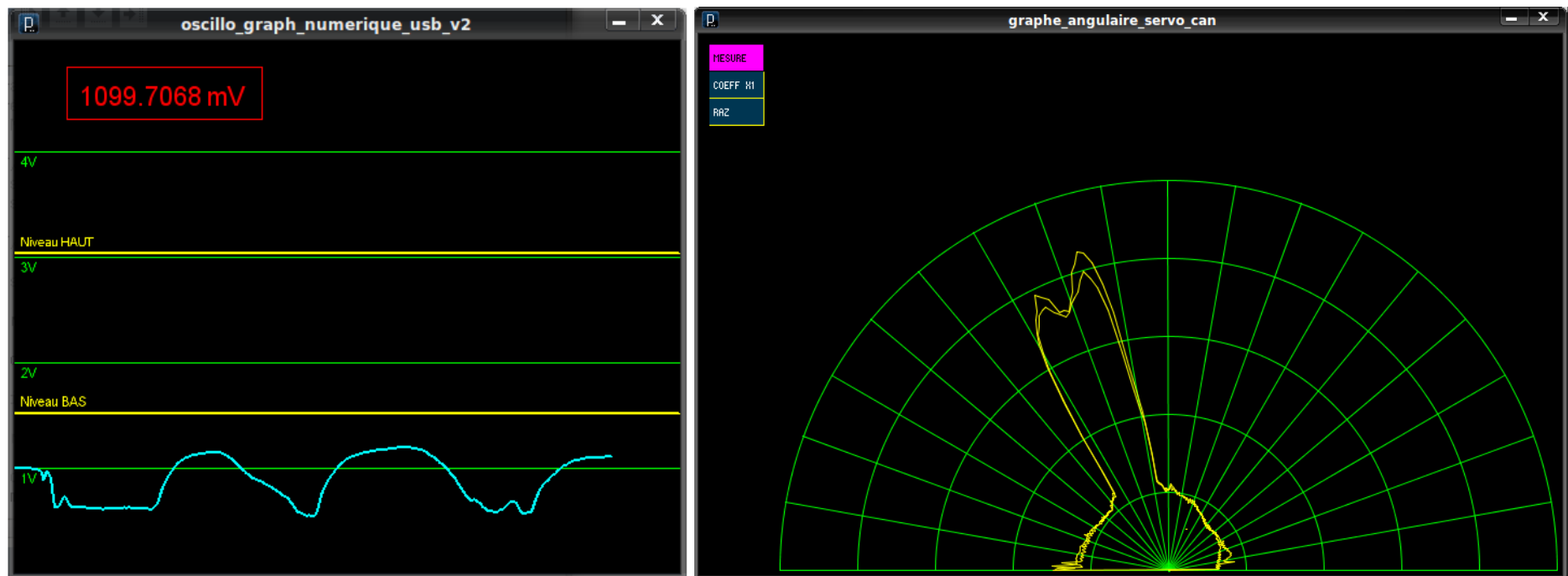
Exemple

Vous trouverez sur le site www.mon-club-elec.fr plusieurs pages proposant les codes Arduino et Processing pour réaliser facilement un petit oscilloscope USB à l'aide d'une carte Arduino. Voir :

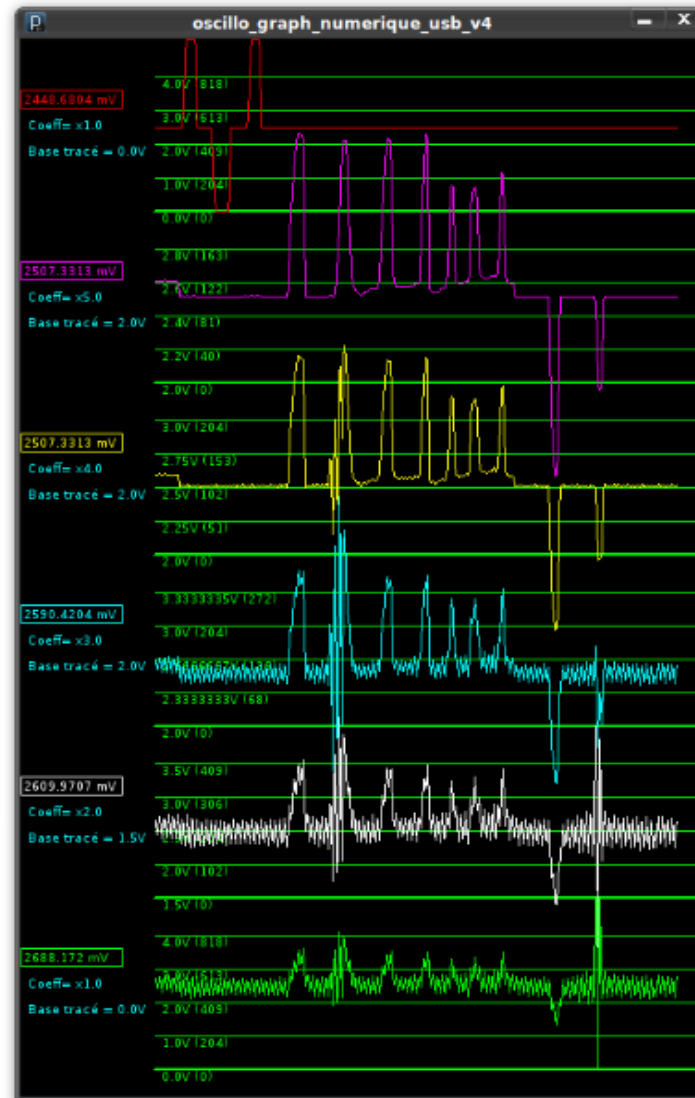
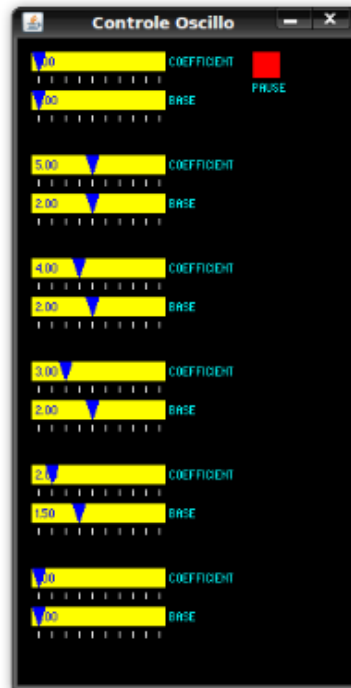
- http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.ArduinoExpertSerieCanGraphiquePC
- http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.OutilsProcessingGraphiques



L'utilisation d'une interface graphique Processing pour visualiser sous forme graphique le résultat de la mesure analogique est très utile en pratique **pour tester les capteurs analogiques** et préciser : la plage de mesure utile, la rapidité de réaction du capteur, la cinétique globale de la réponse (stable ou parasitée...), etc..



A gauche : test d'un capteur de niveau de gris. A droite, test d'un capteur de flamme monté sur servomoteur.



Exemple d'affichage des 6 voies analogiques simultanément.

Les codes Arduino et Processing sont ici :

http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.UTILSProcessingGUIControlP5OscilloMultiVoiesMultiContrôle

21. Fiche composant : la photo-résistance

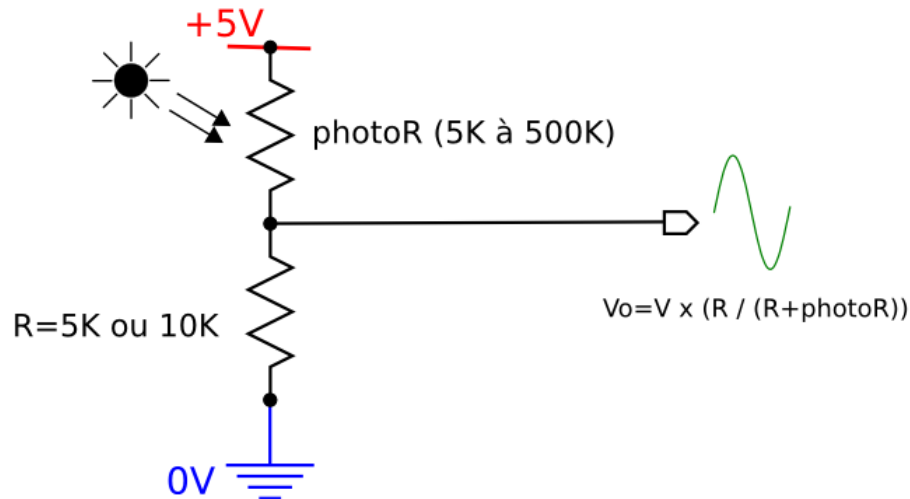
Description

Une photo-résistance est une résistance qui comme son nom l'indique, va varier en fonction de la lumière. Grosso modo :

- lorsque la photo-résistance est éclairée, sa valeur chute, de l'ordre de 10 Kohms avec une luminosité ambiante,
- lorsque la photo-résistance est dans l'obscurité, sa valeur monte (de l'ordre de 500 KOhms)

Principe d'utilisation

Pour pouvoir utiliser efficacement une photo-résistance, il faut la mettre en série avec une autre résistance fixe afin de créer un diviseur de tension : de cette façon, la variation de la luminosité et donc de la résistance de la photo-résistance se traduira par une variation de tension entre les 2 résistances.

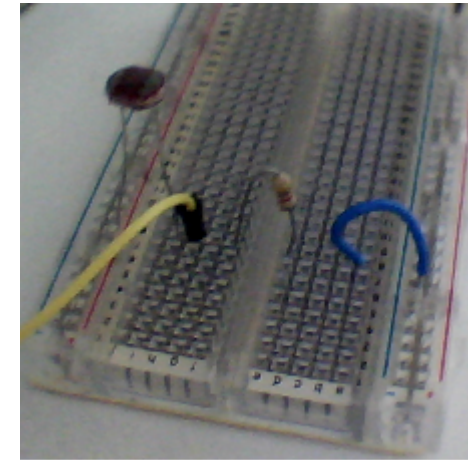


En pratique, utiliser une résistance en série ayant une valeur proche de la valeur de la photo-résistance éclairée, de l'ordre de 5K à 10K.

Exemple d'utilisation sur plaque d'essai avec Arduino

On connecte tout simplement en série la photo-résistance avec une résistance entre le 0V et le 5V.

L'entrée analogique sera connectée au point de jonction entre les 2 résistances.



Pour plus de détails, sur le diviseur de tension, voir : http://fr.wikipedia.org/wiki/Diviseur_de_tension

22. Un détecteur d'obscurité simple : le montage.

Principe d'utilisation d'une photo-résistance

Pour pouvoir utiliser efficacement une photo-résistance, il faut la mettre en série avec une autre résistance fixe afin de créer un diviseur de tension : de cette façon, la variation de la luminosité et donc de la résistance de la photo-résistance se traduira par une variation de tension entre les 2 résistances.

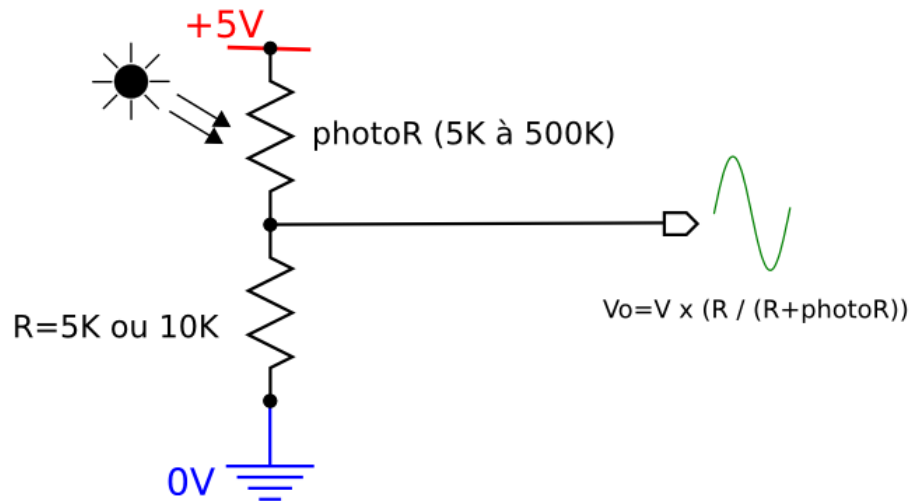
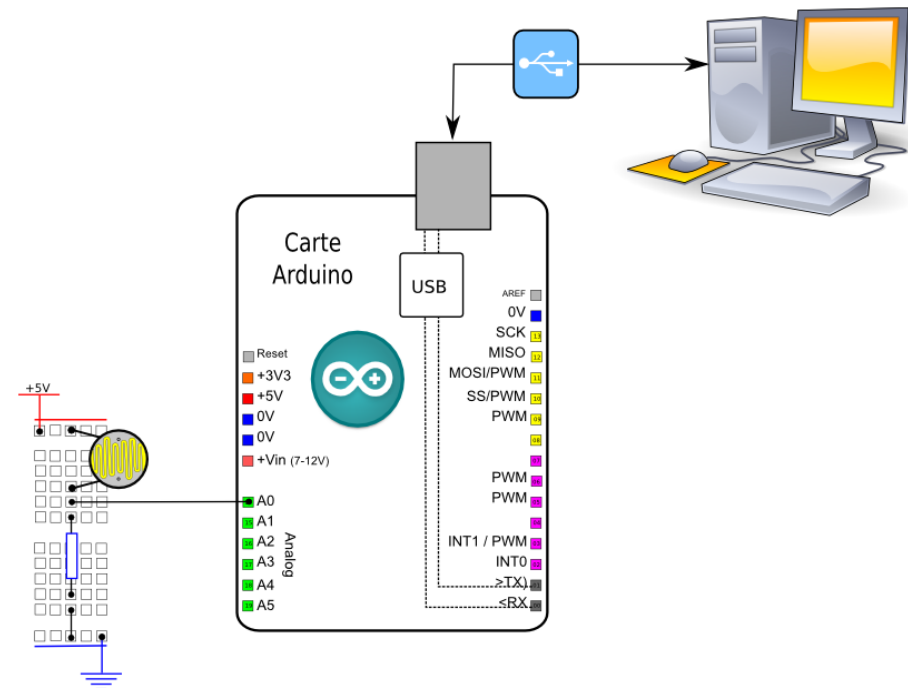


Schéma théorique du montage

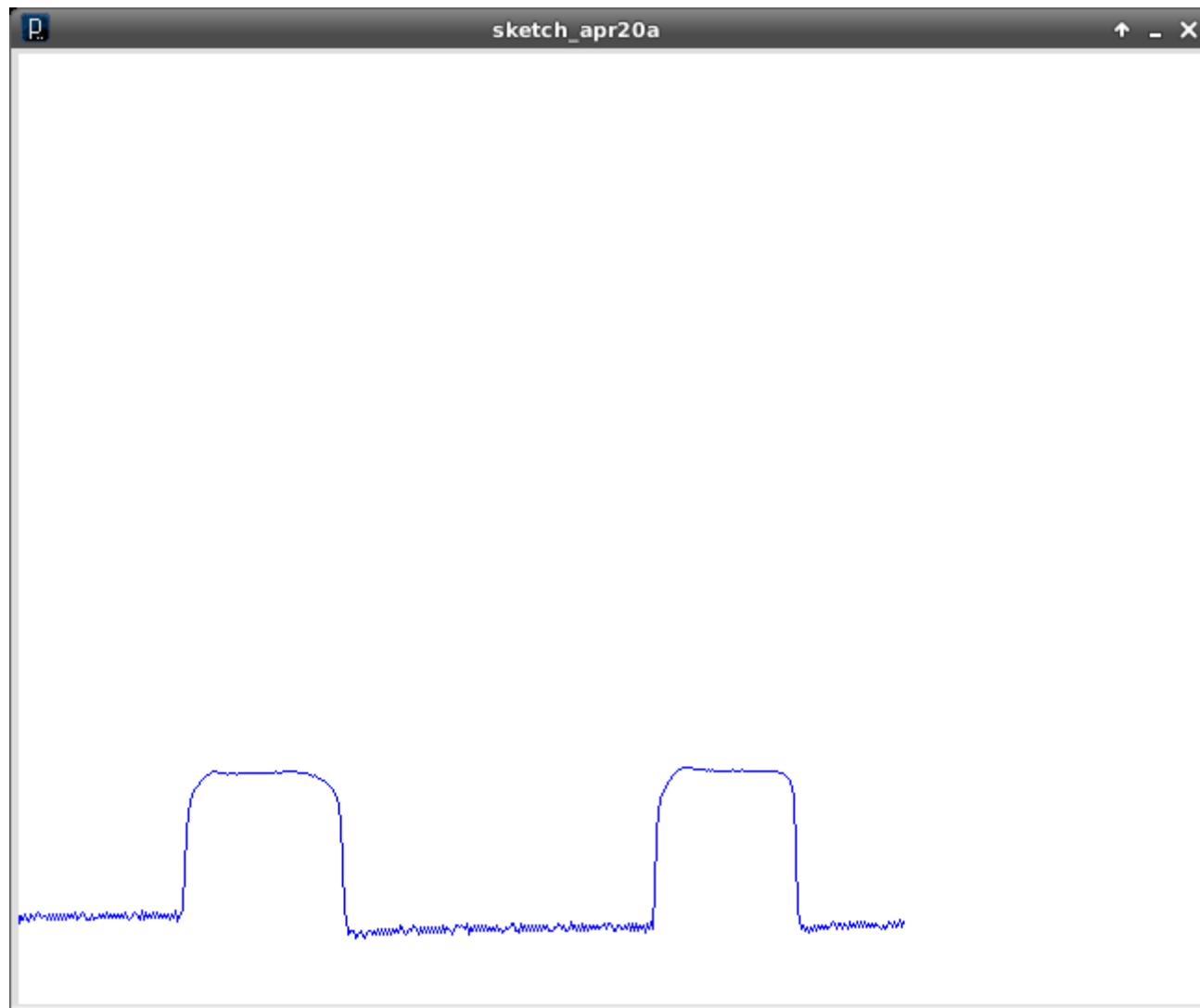
On connecte tout simplement :

- en série la photo-résistance avec une résistance entre le 0V et le 5V.
- L'entrée analogique sera connectée au point de jonction entre les 2 résistances.

La carte Arduino, par ailleurs, sera connectée au PC via le câble USB : on affichera de cette façon les résultats de la mesure côté PC.



23.



Purchased by Franck Ourion, franck.ourion@univ-lorraine.fr #6280170

Atelier Arduino : Entrées analogiques : faire des mesures et utiliser des capteurs analogiques, utiliser les nombres à virgules avec la carte Arduino.

24. Un détecteur d'obscurité simple : le programme.

On reprend la même base programme que ce que l'on a utilisé précédemment et on va utiliser un seuil pour la détection de l'obscurité.

Entete déclarative

On va déclarer à ce niveau :

- une constante de broche pour la voie analogique : **ATTENTION, il faut utiliser le numéro de la broche analogique (entre 0 et 5)** et pas le numéro de la broche numérique (entre 14 et 19..)
- on déclare une variable globale de type **int** pour stocker le résultat de la mesure.

Fonction **setup()**

- on initialise la communication série avec l'instruction **Serial.begin**(vitesse). On utilisera 115200 bauds.

Fonction **loop()**

- on réalise une mesure à l'aide de la fonction **analogRead**(brocheAnalogique) et on stocke dans une variable.
- ensuite, on affiche le résultat dans le Terminal Série à l'aide de la fonction **Serial.println()**
- A l'aide d'une condition, on teste la détection de l'obscurité.
- On réalise ensuite une pause d'une demi-seconde entre 2 mesures avec l'instruction **delay**(500).

Fonctionnement du programme

- Ouvrir le Terminal Série (Tools > Serial Monitor) et fixer le débit à la même valeur que celle utilisée pour l'instruction **Serial.begin**(vitesse). Ici, 115200 bauds.
- Lorsque l'on passe la main sur la photo-résistance, un message indique qu'elle a été détectée.

```
// --- constantes des broches ---

const int photoR=0; //declaration constante de broche analogique

// --- Déclaration des variables globales ---
int mesureBrute=0; // Variable pour acquisition résultat brut de
conversion analogique numérique

int seuil=200; // Variable fixant le seuil de détection de l'obscurité -
à adapter

//***** FONCTION SETUP = Code d'initialisation *****

void setup() { // debut de la fonction setup()

Serial.begin(115200); // initialise connexion série à 115200 bauds
// IMPORTANT : régler le terminal côté PC avec la même valeur de transmi
ssion

} // fin de la fonction setup()

//***** FONCTION LOOP = Boucle sans fin *****

void loop(){ // debut de la fonction loop()

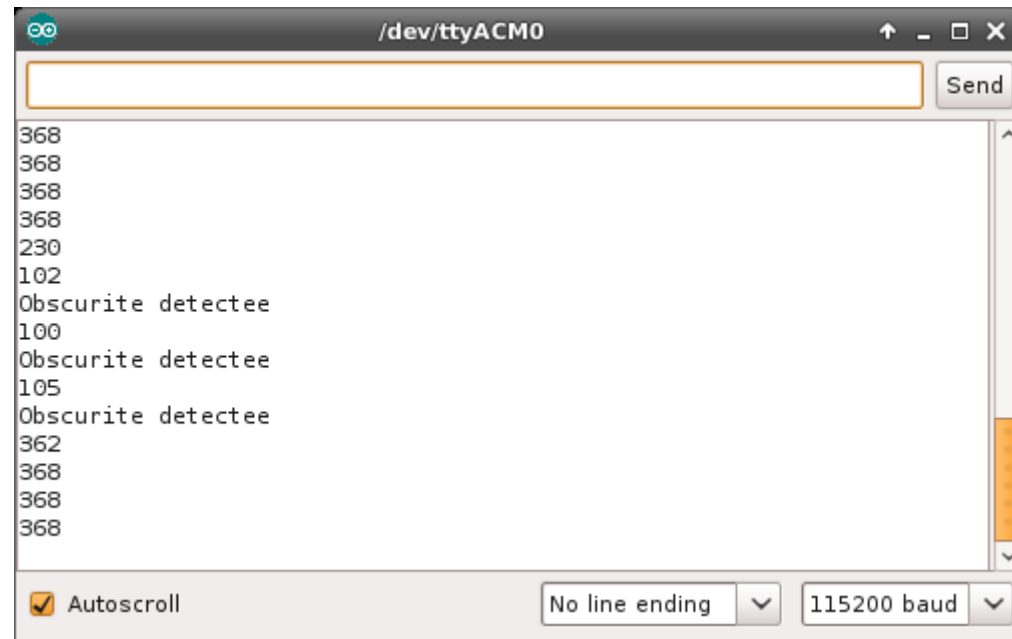
// acquisition conversion analogique-numerique (CAN) sur la voie analogi
que
mesureBrute=analogRead(photoR);

// affiche valeur numerique entière ou à virgule au format décimal
Serial.println(mesureBrute);

if (mesureBrute<seuil) Serial.println("Obscurite detectee"); // message
si obscurité détectée

delay(500);

} // fin de la fonction loop() - le programme recommence au début de la
fonction loop sans fin
```



Purchased by Franck Ourion, franck.ourion@univ-lorraine.fr #6280170

Atelier Arduino : Entrées analogiques : faire des mesures et utiliser des capteurs analogiques, utiliser les nombres à virgules avec la carte Arduino.

25. Exercices d'application

A ce stade, vous devriez être en mesure d'écrire par vous même quelques programmes utilisant les mesures. Voici quelques suggestions d'exercice à réaliser :

- Faire varier la luminosité d'une LED à l'aide d'une résistance variable
- Réalisation d'un simple « vumètre » à 10 LEDs à l'aide d'une résistance variable
- Faire varier la fréquence d'un son à l'aide d'une résistance variable

Vous trouverez par ailleurs plusieurs codes d'exemple ici : http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.ArduinoExpertCan

26. Les éléments du langage Arduino étudiés dans cet atelier

Structure

Variables et constantes

Fonctions

Types de données

- [float](#) (nombres à virgules)

Entrées analogiques

- int [analogRead](#)(broche)

Fonctions math

- [map](#)(valeur, fromLow, fromHigh, toLow, toHigh)

La documentation complète du langage Arduino en français est disponible ici :
http://www.mon-club-elec.fr/pmwiki_reference_arduino/pmwiki.php?n=Main.ReferenceMaxi

27. A présent, vous devriez être capable :

- D'utiliser les capteurs analogiques linéaires ou non, de réaliser des mesures et de calculer des grandeurs en utilisant les nombres à virgule avec une carte Arduino.

Table des matières

Broches analogiques : faire des mesures et utiliser des capteurs analogiques, utiliser les nombres à virgules avec la carte Arduino.

Intro |

Matériel nécessaire pour les ateliers Arduino |

Rappel : Notion d'électronique numérique et analogique |

Rappel : Une broche numérique ne peut avoir que 2 états : HAUT ou BAS, « y'a ou y'a pas » ! |

Rappel : Une broche numérique est caractérisée par son SENS : en SORTIE ou en ENTREE ! |

Le concept de « conversion analogique-numérique » (du monde physique vers le monde numérique !) |

Principe de fonctionnement d'une broche analogique |

Les broches analogiques de la carte Arduino |

Pour info : les caractéristiques de la « règle à tension » numérique de l'Arduino |

Fiche composant : découvrir la résistance variable. |

Affichage d'une mesure analogique dans le Terminal Serie : le montage |

Affichage d'une mesure analogique brute dans le Terminal Serie |

|

Langage : Le type float pour stocker les valeurs numériques à virgule |

Affichage d'une mesure analogique convertie en millivolts dans le Terminal Serie |

Fiche technique : Principe d'utilisation d'un capteur analogique |

Exemples de capteurs analogiques utilisables avec une carte Arduino |

Fiche composant : le capteur analogique linéaire de température LM35 |

Réaliser un simple thermomètre numérique avec affichage dans le Terminal Série : le montage |

Réaliser un simple thermomètre numérique avec affichage dans le Terminal Série : le programme |

Exemple d'affichage de résultat sous forme graphique : un mini-oscillo USB de base ! |

Fiche composant : la photo-résistance |

Un détecteur d'obscurité simple : le montage. |

|

Un détecteur d'obscurité simple : le programme. |

Exercices d'application |

Les éléments du langage Arduino étudiés dans cet atelier |

A présent, vous devriez être capable : |

Bravo !
vous avez terminé cet atelier Arduino !



Prêt pour la suite ? Retrouvez de nombreux autres thèmes d'ateliers Arduino ici :

http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.ATELIERS