# Chapitre 4.2 : Conditionnement et conversion des signaux

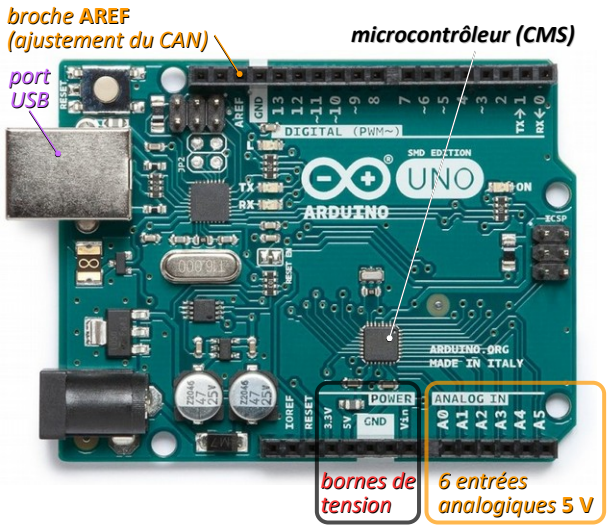
## Système : Carte à microcontrôleur Arduino — TP – 2 h

## Mise en situation

■ Un ***microcontrôleur*** est un ***circuit intégré*** qui embarque non seulement un ***microprocesseur*** – composant ***programmable*** pouvant exécuter très vite toutes sortes de ***calculs*** sur des ***données numériques*** – mais aussi des unités de ***mémoire*** (pour stocker les données) et des ***ports d'entrée/sortie*** (pour échanger des données avec son environnement). Sur une carte *Arduino Uno*, il s'agit d'un microcontrôleur *Atmel AVR ATmega328p*.

Les ports d'entrée/sortie sont accessibles via des ***connecteurs Dupont*** (fiches femelles noires) reliées par des pistes imprimées aux broches du microcontrôleur.

La carte dispose également d'un ***port USB*** assurant sa communication par liaison série avec un ***terminal de programmation*** (ordinateur) et son alimentation en tension.




■ Dans ce sujet de travaux pratique, on expérimente sur une carte *Arduino Uno R3 SMD* (cf. fig. ci-contre) le fonctionnement des ***ports d'entrées analogiques*** (bornes **A0** à **A5**) :

- ils sont associés à un **CAN *unipolaire* 0 – 5 V** à **10 bits** qui numérise en ***conversion linéaire centrée*** la tension $U_e$ présente entre une borne **A*i*** et les bornes de masse **GND** ;
- la tension $U_{max}$ du CAN est ***ajustable*** à une valeur $U_{ref}$ comprise entre 0 et 5 V en appliquant le potentiel $U_{ref}$ à la borne **AREF** (*analog reference*) de la carte.

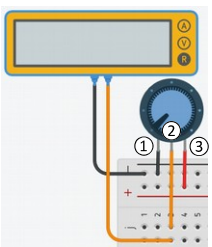
Pour limiter les risques de surtension ou de potentiels de référence flottants, on utilise comme ***source de tension*** $V_{CC}$ les bornes **5V** et **3.3V** de la carte alimentée par USB. Elles sont au potentiel éponyme par rapport à celui des bornes **GND**. On ajuste la tension $U_e$ sur la borne d'entrée **A0** via un pont diviseur de tension implémenté par un ***potentiomètre*** de résistance $R$ = 2 kΩ / 10 tours.

## Travail demandé (compléter le sujet et imprimer la feuille de calcul)

Sans danger pour les personnes, les ***courts-circuits*** en très basse tension (5 V) peuvent néanmoins provoquer des ***dégâts considérables*** sur le matériel. Une ***grande vigilance*** et le ***respect des consignes*** sont requis lors des câblages, afin qu'aucune borne de tension ne soit **jamais** reliée directement à une borne de masse.

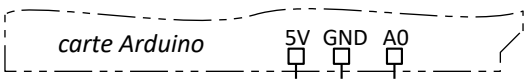
### A. Installation et paramétrage du matériel de laboratoire [40 min] ☺ 😐 ☹

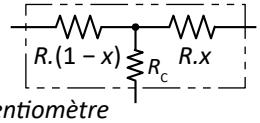
1. a) Câbler le potentiomètre sur la platine d'essai comme sur la figure ci-contre. Mesurer la ***résistance de contact du curseur*** ② $R_c$ = _____. Pourquoi est-elle négligeable ?



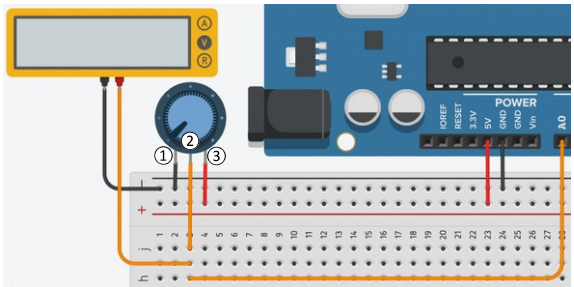

b) Commuter le multimètre en position ***voltmètre*** et puis **hors-tension**, relier à la platine d'essai les bornes **5V**, **GND** et **A0** de la carte *Arduino* selon la figure ci-dessous.

c) Ci-dessous, compléter le ***schéma électrique*** équivalent à ce montage, y représenter les tensions $U_e$ (aux bornes de $R.x$) et $V_{CC}$. Appeler l'enseignant pour validation.



potentiomètre

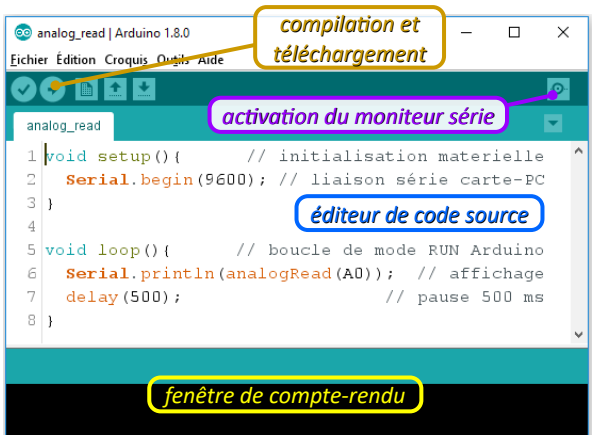


2. a) ***Raccorder la carte*** sur un port USB du terminal (PC) et ***ouvrir le programme*** fourni **analog_read.ino**. Prendre connaissance du code source.

   b) ***Activer le moniteur série*** et vérifier dans la barre inférieure que la vitesse est réglée à **9600 baud**.

   c) ***Compiler et téléverser le programme*** dans le microcontrôleur de la carte.

   d) En tournant le potentiomètre, vérifier que le moniteur série affiche un ***nombre entier naturel*** $N_s$ numérisant de `0` à `1023` la tension d'entrée $U_e$ mesurée au multimètre. Appeler l'enseignant pour validation.




## B. Mesures et interprétation [60 min] ☺ 😐 ☹

3. a) Dans la ***feuille de calcul*** `.ods`, relever dans le 1er tableau les valeurs de $N_s$ (colonne B) pour $U_e$ (colonne A) variant de **0** à **5 V** par pas de **0,5 V** environ (mais indiquer $U_e$ avec une résolution de **1 mV**).

   b) Déterminer les formules des cellules vides ($U_{max}$, $q$, $a$, etc.) puis tracer le ***graphe (X,Y)*** de ces mesures et leur ***droite de tendance*** pour vérifier la linéarité grossière du CAN. À quoi correspond la pente $a$ ?

   c) En colonne C, calculer pour chaque valeur de $N_s$ sa valeur correspondante $U_n$ en volts selon la droite de pente $a$. En déduire en colonne D l'***erreur relative*** $\varepsilon_V$ entre $U_n$ et $U_e$. Puis calculer en colonne E la valeur de $N_{th}$ pour un CAN idéal en conversion linéaire centrée (cf. cours, p. 4). En déduire l'erreur relative $\varepsilon_N$ de $N_s$.

   d) Dans le 2e tableau, relever les valeurs de seuil de $U_e$ pour les changements de valeur de $N_s$ allant de `0` à `10`. En déduire (colonne C) le pas $p$ entre chaque seuil. Que représentent ces pas (cf. cours) ? Calculer en colonne D la valeur théorique $U_{e\,th}$ pour $N_s$ puis l'écart absolu $\Delta U$ entre $U_{e\,th}$ et $U_e$.

   e) Tracer le ***graphe (X,Y)*** de ces mesures avec l'option « ligne en escalier » pour obtenir la fonction de transfert réelle du CAN. Tracer également la fonction de transfert du CAN idéal. Conclure.

## C. Ajustement du CAN [20 min] ☺ 😐 ☹

4. a) ***Commuter le fil d'alimentation*** de la platine d'essai sur la borne **3.3V** de la carte *Arduino* puis tourner le potentiomètre en fin de course dans le sens des aiguilles d'une montre. Relever la valeur $N_s$. Est-elle conforme aux données précédentes ?

   b) ***Raccorder la borne*** `AREF` (et pas `GND` 🔥) de la carte à l'alimentation de la platine d'essai (sous 3,3 V) et ajouter l'instruction `analogReference(EXTERNAL);` dans la fonction `setup` du code source. Recompiler et téléverser le programme puis relever la valeur de $N_s$ comme à la question 4.a). Expliquer le résultat.