ALGORITHMIQUE
et Initiation à la Programmation

Christophe Fond

Université de Strasbourg
Christophe.Fond@unistra.fr

Résumé

Ce cours est une modeste introduction à la programmation. Il s'agit de programmer une carte de type ARDUINO pour mesurer une température et l'afficher. Pour cela il faut programmer des boucles et des tests.

Mots clefs: ARDUINO, capteur de température, thermorésistance, langage C++.

Keywords : ARDUINO, temperature sensor, thermoresistance, C++ language.

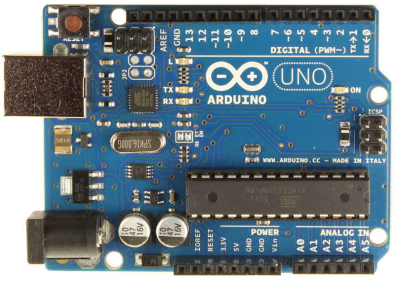

Figure 1: Carte ARDUINO UNO.

Table des matières

Préambule

Ce cours est principalement illustré avec des cartes ARDUINO UNO. Le langage utilisé est le C++. L'environnement ARDUINO permet de partir d'exemples pour ensuite les modifier.

1. Introduction

Dans le cadre d'une modeste initiation à la domotique, ce cours considère un des sous-domaines de la domotique qui consiste à communiquer avec une carte comportant un microprocesseur programmable et à le programmer.

2. Signal analogique et digital

Signal analogique

Le signal analogique peut être un voltage ou (différence de potentiel), un courant, une intensité lumineuse, etc.

Signal digital

Le signal digital est constitué d'impulsions électriques (ou lumineuses transformées en impulsions électriques). Ces impulsions doivent être décodées pour obtenir une information sous forme de caractères (chiffres ou lettres). Il faut connaître le protocole de communication (fréquences, etc.) pour savoir comment décoder l'information transmise. Le décodage de l'information est effectué à l'aide des librairies qui sont généralement fournies avec le matériel. Il convient de s'assurer que l'on aura accès à la (aux) librairie(s) d'un matériel avant de l'acheter...

Conversion analogique vers digital - ADC

ADC signifie Analogic Digital Conversion. ADC est le mot clef pour rechercher des informations sur le sujet. Le signal analogique est converti en entier codé sur un nombre n de bits. Par exemple, pour la carte UNO, il est codé sur 10 bits. En d'autres termes, un voltage compris entre 0 et $5V$ sera transformé par l'ADC de la carte en un entier compris entre 0 et 1023 comme l'explique l'Annexe A. Par exemple, si l'entier retourné vaut 618 et le voltage de référence vaut $5V$ alors le voltage mesuré vaut $U = 5.0 * 618/1023 = 3.0205V$.

La précision et de l'ordre du millième dans ce cas (1/1023). Si on considère que 617.51 est arrondi à 618 ainsi que 618.49 alors le résultat est $U = 3.0205 \pm 0.005V$.

3. Mesure d'une température avec une thermistance

Monter la thermistance MCP9700a sur la carte Arduino UNO en branchant sur l'alimentation stabilisée de la carte (5V/GND). Brancher le fil du milieu ("mesure" sur la Fig. 2, photo de droite) sur la prise (pin) A0 de la carte qui fera la mesure du courant par ADC.

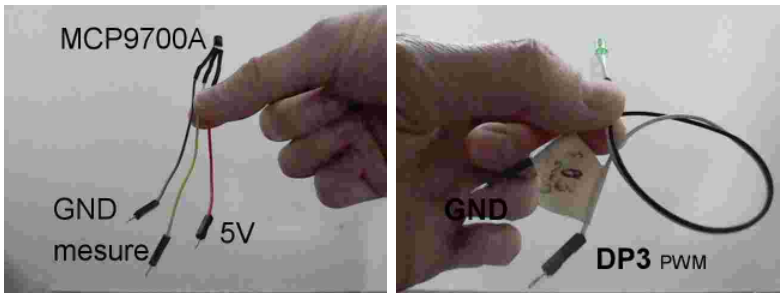

FIGURE 2: À Droite : thermistance MCP9700a et ses connexions. À gauche : diode luminescente (LED) et ses connexions.

Brancher la carte sur un port USB. Ouvrir le logiciel Arduino puis "Outils" → "Type de carte" pour choisir "Arduino UNO". Ensuite "Outils" → "Port" pour choisir le port COM correspondant au branchement de la carte.

La température T est donnée par $T \approx (U - 0.5) * 100$ (chercher avec les mots clefs "MCP9700" et "datasheet" sur internet les spécifications de la puce). On peut étalonner la thermistance avec un thermomètre de précision en ajustant dans l'équation précédente la constante valant initialement 0.5.

A partir de l'exemple "Fichier" → "Exemples" → "01.Basics" → "AnalogReadSerial", envoyer l'entier qui correspond au voltage mesuré vers le port COM ("Outils" → "Moniteur Série" pour afficher les informations reçues par le port COM). Il suffit de cliquer sur la Téléverser (flèche en haut à gauche).

Transformer cet entier en voltage puis en température et envoyer l'information sur le port COM. Éventuellement étalonner la thermistance.

4. Mesures et moyennes, précision et bruit électrique

Il conviendra de faire des organigrammes avant de passer à la programmation.

Calcul des moyennes

Faire une mesure puis la moyenne de trois mesures, puis 10 mesures et finalement 100 mesures et envoyer les quatre moyennes vers le port série (port COM) séparées par un espace. Voir des exemples utiles pour ce chapitre en Annexe B.

Affichage du résultat

A l'aide du fichier "processing_thermistance.pde" que l'on copiera dans un répertoire du même nom, afficher à l'aide de l'environnement "Processing" la graphe en temps réel des températures mesurées et moyennées ou non. A la ligne :
myPort = new Serial(this, Serial.list()[2], 9600) ;
il faudra éventuellement modifier le numéro du port "Serial.list()[votre numéro de port = 0, 1, 2, 3...]", la liste apparait dans l'ordre en partant de 0 au lancement du programme. A la ligne :
float zoomin=22.0, zoomax=24.0 ;
on fixe l'intervalle de température qui sera affiché, ici $T \in [22, 24]$. Les valeurs mesurées sont copiées dans le fichier "valeurs.txt".

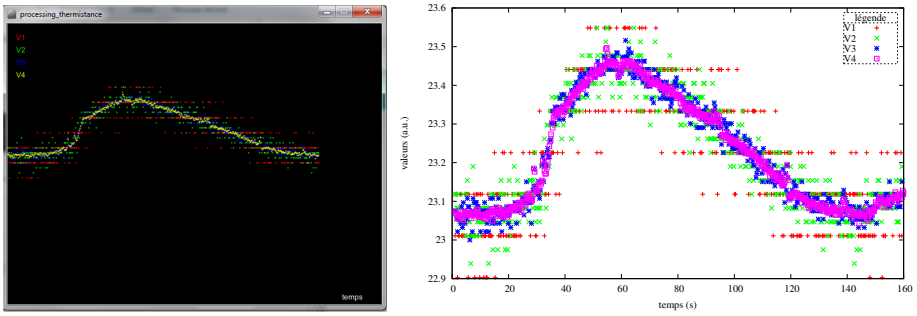

FIGURE 3: À Droite : écran affiché par Processing en temps réel. À gauche : graphe réalisé avec les valeurs enregistrées.

Que peut-on en déduire ?

5. Pilotage d'une LED

Il conviendra de faire des organigrammes avant de passer à la programmation.

A partir de l'exemple "Fichier" → "Exemples" → "01.Basics" → "Fade", faire s'allumer progressivement la LED si la température augmente, s'éteindre progressivement si la température diminue et se maintenir à luminosité constante si la température est stable (voir video_thermistance.avi). Attention, votre LED est branchée sur D3 et non sur D9, il faut donc modifier l'exemple "Fade" ("int led = 9 ;" devient "int led = 3 ;"). Les réglages en durée et limites en écart de température sont laissés à l'initiative du programmeur.

Pour réaliser ce type d'affichage progressif il faut disposer d'une conversion digital vers analogique (DAC). La prise (pin) 3 de la carte possède une fonction DAC de type PWM. Chercher sur internet le sens de cette appellation et donner les limitations.

Annexe A. Exemple de codage sur 10 bits

La table suivante fournit le principe du codage sur 10 bits en langage binaire.

registre	1	2	3	4	5	6	7	8	9	10
valeur	0 ou 1	0 ou 1	0 ou 1	0 ou 1	0 ou 1	0 ou 1	0 ou 1	0 ou 1	0 ou 1	0 ou 1
poids	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
poids	512	256	128	64	32	16	8	4	2	1

TABLE A.1: Codage binaire sur 10 bits.

Considérons un entier N codé sur 10 bits. Si tous ses registres contiennent 1 il prendra la valeur maximale possible à coder avec 10 registres binaires, ce qui s'écrit $N = 1111111111$ en binaire, $N = 1*512+1*256+1*128+1*64+1*32+1*16+1*8+1*4+1*2+1*1 = 1023$ en décimal [1]. Si $K = 5$ en décimal, puisque $5 = 0*512+0*256+0*128+0*64+0*32+0*16+0*8+1*4+0*2+1*1$, $K = 0000000101$ en binaire que l'on écrit, comme en décimal en omettant les 0 inutiles, $K = 101$.

Annexe B. Exemples de boucle et tests en C++

Voici un exemple de boucle :

```
int i;
float somme=0,valeur=1.5;
for (i = 0; i < 10; i++) {
somme = somme + valeur;
}
```

Voici des exemples de test :

```
if (somme == 0) {
Serial.println("rien n'a ete ajoute a somme");
}


if (somme > 0) {
Serial.println("somme est superieure a 0");
}
else {
Serial.println("somme est inferieure ou egale a 0");
}
```

1. on obtiendra $N = 3FF$ en hexadécimal sur le même principe...

Références

http ://www.arduino.cc
https ://processing.org