

TP4 – Mesurons la température

1. Rappel sur l'utilisation de l'environnement

- Lancer un terminal (une console).
- Tapez : `/home/shared/arduino/arduino-1.6.0/arduino` et l'environnement se lancera.

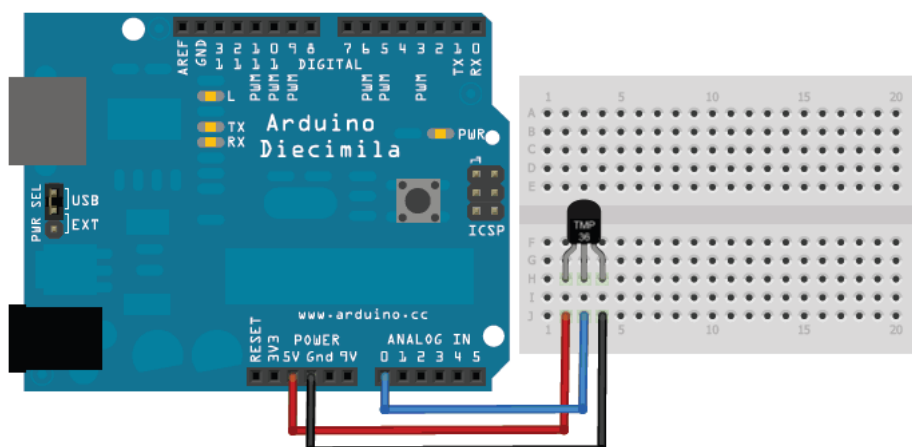
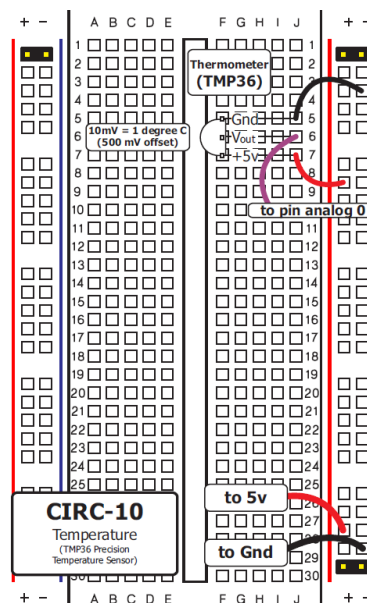
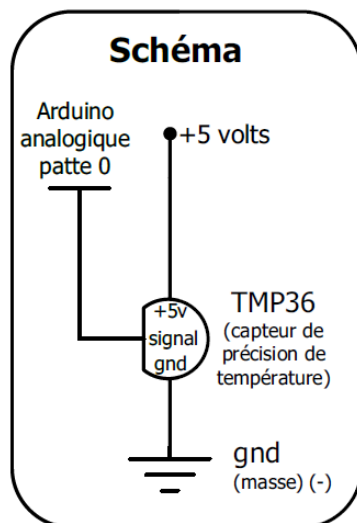
Si vous avez créé un lien symbolique depuis votre *home*, (faire `cd ~` dans le terminal), il suffira de taper `./arduino`

ATTENTION : RELIER LE 5V d'un ARDUINO À LA MASSE PEUT LE TUER (MÊME SI C'EST FAIT VIA LE BREADBOARD !!!

On rappelle que l'on a vu :

- un exemple de datalogger sur carte SD page 108 et suivantes du cours
- comment mesurer la température avec le TMP36 page 110 et suivantes du cours

Réaliser ce circuit :



Voici le code à téléverser (puis ouvrir le moniteur série) :

```
/* -----
 * | Code d'Exemple du Kit d'Expérience Arduino |
 * | CIRC-10 .: Température .: |
 * -----
 * Un programme simple pour afficher la température
 * sur la console du logiciel
 * Pour plus de détails sur ce circuit :
 */

//assignement des pattes du TMP36
int temperaturePin = 0; //la patte analogique à laquelle la patte Vout du TMP36 est connectée
                        //la resolution est de 10 mV / degré Celsius
                        //(500 mV d'offset) pour obtenir des températures négatives

void setup()
{
  Serial.begin(9600); //lance la connexion série avec l'ordinateur
                     //pour voir le résultat ouvrez la console série
                     //dernier bouton sous la barre de menu
                     //(il ressemble à rectangle avec une antenne)
}

void loop() //s'exécute à l'infini
{
  float temperature = getVoltage(temperaturePin); //obtiens la tension de sortie du
                                                  //capteur de température

  temperature = (temperature - .5) * 100; //conversion de 10mV
                                          //par degre avec 500mV d'offset en
                                          //degrés ((tension - 500mV) * 100)

  Serial.println(temperature); //affiche le résultat
  delay(1000); //attend une seconde
}

/*
 * getVoltage(int pin) - retourne la tension sur la patte analogique
 * définie par pin
 */
float getVoltage(int pin){
  return (analogRead(pin) * .004882814); //Convertie d'une plage de 0 à 1023
                                          //à une plage de 0 à 5V (Chaque pas égal environ 5 mV)
}
```

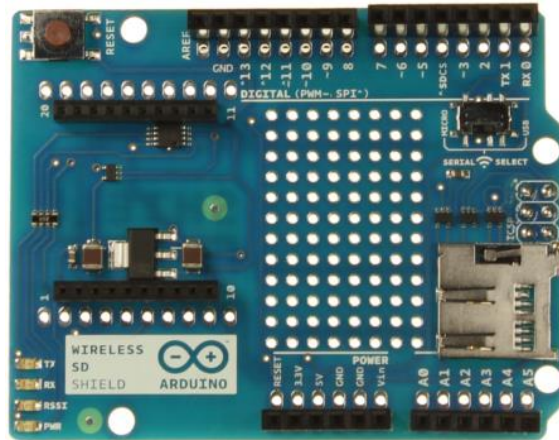
Pour améliorer le montage vous pouvez changer le `Serial.begin(9600)` par `Serial.begin(115200)` afin de communiquer 12 fois plus vite avec le PC !

Pour faire varier la température, vous pouvez souffler dessus, le rapprocher du ventilateur de l'ordinateur, etc. Soyez imaginatif mais pas de briquet et pas de cigarette ! Toucher un café ou un thé chaud est une solution. Toucher un soda froid en est une autre.

Utilisation de la carte microSD

Si vous avez un Arduino Uno classique, vous aurez besoin du shield wireless SD :

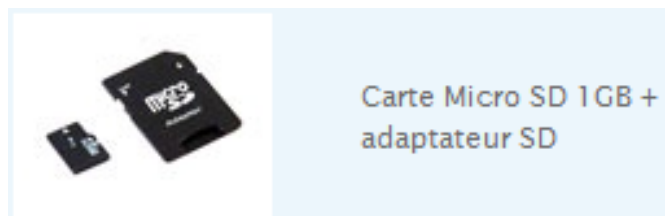
(Merci de l'enficher en douceur)



Bien que normalement, la patte 10 d'un Arduino Uno soit utilisée pour le CS, comme vous le constatez ici (<https://www.arduino.cc/en/Main/ArduinoWirelessShield>), c'est bien la patte 4 doit utilisée pour le CS (Cable Select aussi appelé SS pour Slave Select du Bus SPI permettant d'accéder à la carte microSD). Merci de laisser le commutateur de la carte sur « Micro ». Ce choix de design permet d'utiliser la patte 10 de l'Arduino UNO pour sélectionner un second périphérique éventuellement présent sur le bus SPI.

Si vous avez un Arduino Leonardo ETH, vous avez directement accès au port SD !

Placer la carte microSD à l'intérieur (sans l'adaptateur !)



Rebranchez les fils sur les bonnes pattes.

Téléversez le code suivant puis ouvrez le moniteur série :

```
#include <SD.h>
const int chipSelect = 4;

//assignement des pattes du TMP36
int temperaturePin = 0; //la patte analogique à laquelle la patte Vout du TMP36 est connectée
                        //la resolution est de 10 mV / degré Celsius
                        //(500 mV d'offset) pour obtenir des températures négatives

long timeRef=0; // variable pour temps de référence en millisecondes

void setup()
{
  // Open serial communications and wait for port to open:
  Serial.begin(9600);

  Serial.print("Initializing SD card...");
  // make sure that the default chip select pin is set to output, even if you don't use it:
  pinMode(chipSelect, OUTPUT);

  // see if the card is present and can be initialized:
  if (!SD.begin(chipSelect)) {
    Serial.println("Card failed, or not present");
    // don't do anything more:
    return;
  }
  Serial.println("card initialized.");
  timeRef=millis(); // initialisation de la référence du comptage
}

void loop()
{
  // make a string for assembling the data to log:
  String dataString = "";

  dataString += String(millis()-timeRef);
  dataString += ",";
  dataString += String(analogRead(temperaturePin));

  // open the file. note that only one file can be open at a time,
  // so you have to close this one before opening another.
  File dataFile = SD.open("datalog.csv", FILE_WRITE);

  // if the file is available, write to it:
  if (dataFile) {
    dataFile.println(dataString);
    dataFile.close();
    // print to the serial port too:
    Serial.println(dataString);
  }
  // if the file isn't open, pop up an error:
  else {
    Serial.println("error opening datalog.txt");
  }

  delay(1000); //attend une seconde
}
```

Faites varier un peu la température (avec la glace, en touchant le capteur, etc.)

Au bout de 5 minutes, récupérez la carte microSD, mettez la dans son adaptateur et branchez sur le port USB de l'ordinateur le lecteur de carte SD qu'on vous a fourni.

Insérez la carte et récupérez le fichier « datalog.csv » sur votre compte. Ouvrez « Calc » (LibreOffice/OpenOffice) et faites un diagramme illustrant les modifications de températures en fonction du temps en n'oubliant pas avant de réaliser au préalable les conversions :

$$= (VAL_ANAL * 0,004882814) - 0,5) * 100$$

Lecture du fichier CSV en Processing

1. Écrire le programme Processing qui lit un fichier CSV à récupérer sur le site du responsable de l'UE et l'affiche les données sous la forme :

temps : 470996 - temperature : 147

Il s'agit ci-dessus de la température non convertie et du temps en millisecondes.

2. Écrire un programme Processing qui lit un fichier CSV et qui calcule la température moyenne, les températures minimum et maximum et les temps auxquelles elles apparaissent.

La temperature moyenne est de xxx°C

La temperature minimum est de xxx°C et elle a eu lieu 146 secondes apres le debut de l'execution du programme

La temperature maximum est de xxx°C et elle a eu lieu 327 secondes apres le debut de l'execution du

Interaction Processing – Arduino

1. Écrire le programme Processing qui lit les données de températures envoyées sur le port série par le premier programme Arduino de cette feuille de TD et qui dessine un cercle dont le rayon est la température reçue.

Pour récupérer la température, on pourra utiliser `readStringUntil` (jusqu'à avoir un retour à la ligne – voir

https://processing.org:8443/reference/libraries/serial/Serial_readStringUntil_.html) et ensuite on pourra transformer cette `String` en `float` avec `float()`.

2. Écrire cette fois le programme Processing qui lit sur le port série les données provenant directement du convertisseur analogique numérique de l'Arduino (il faut donc modifier un peu le code précédent à mettre dans l'Arduino) et qui calcule lui-même les températures avant de les afficher sur la sortie standard.