Reporting Quatre Quadrants n°1

Projet Minuto - FISE A1

Groupe B11

15 octobre 2024

Consitution de l'équipe

Groupe/Equipe : B11
Chef de projet : Emilien WOLFF

Actualisation de la fiche à la date du : 15/10

▷ Ce que nous avions prévu de faire aujourd'hui

- Réalisation des deux montages pour la mesure de température à partir de la thermistance : diviseur de tension et **pont de Wheatstone**
- Réalisation du montage pour la mesure de température absolue à partir du capteur TMP117
- Code pour l'étalonnage de la température de la thermistance et tracés des courbes
- Montage sur breadboard et étude de la compacité
- Mise au propre des tâches dans *GanttProject* et définition des liens entre les tâches
- Recherche des coefficients de la courbe de la thermistance (coefficients de Steinhart-Hart)
- Détermination de la résistance de la thermistance R_{th}
- Soudure des fils (pont diviseur de tension entier)

▷ Ce que nous avons réalisé effectivement

- Réalisation du diviseur de tension et branchement du TMP117 sur la même carte. Essai de mesure de l'écart de température entre les deux capteurs, et essais de tracés de courbe
- Soudure des fils (pont diviseur de tension entier) au propre et mise en place de gaines thermo pour éviter les court-circuits
- Mise au propre des tâches dans *GanttProject* et définition des liens entre les tâches

▷ Ce que nous prévoyons de faire les prochains jours

- Etalonnage de la thermistance (prochaine séance de minuto)
- A partir des besoins pour la mesure finale, établissement de la structure finale du boitier.
- Tester les différentes parties de code indépendamment

▷ Problèmes rencontrés et solutions mises en oeuvre

- Changement des fils défectueux et ajout de fils supplémentaires pour ne pas que le problème réapparaisse
- Léger écart entre la température mesurée et la température supposée de la pièce → à quantifier !
- Problèmes rencontrés sur le test final avec toutes les composantes → test des différentes parties du code indépendamment

Attention : *Besoin de transportabilité à satisfaire*

Montage non transportable, nécessité de transportabilité → utilisation de la mémoire de l'Arduino, donc modification du code dans le futur.

Annexes : codes de la session

Test diviseur de tension

```
// D finir les broches
const int pinThermistor = A0;  // Entr e analogique pour lire la tension
const float Vcc = 3.3;         // Tension d'alimentation (3.3V dans ton montage)
const float R = 10000;         // R sistance en s rie (10 k )

// Caract ristiques de la thermistance
const float R0 = 10000;        // R sistance de la thermistance    25 C  (10 k
    )
const float T0 = 298.15;       // Temp rature de r f rence en Kelvin (25 C  =
     298.15K)
const float B = 3950;          // Constante B de la thermistance

void setup() {
  Serial.begin(9600);          // Initialiser la communication s rie
}

void loop() {
  int valeurBrute = analogRead(pinThermistor);  // Lire la tension (0    1023)
  float Vmes = (valeurBrute / 1023.0) * Vcc;    // Convertir en tension r elle

  // Calcul de la r sistance de la thermistance (Rth)
  float Rth = (R * Vmes) / (Vcc - Vmes);

  // Utilisation de l' quation  de Steinhart-Hart pour convertir en temp rature
  float inv_T = (1/T0) + (1/B) * log(Rth / R0);
  float temperature = (1 / inv_T) - 273.15;     // Conversion en  C

  // Afficher la temp rature et la tension
  Serial.print("Tension Vmes: ");
  Serial.print(Vmes);
  Serial.print(" V, Temp rature: ");
  Serial.print(temperature);
  Serial.println("  C ");

  delay(1000);  // Attendre 1 seconde avant la prochaine lecture
}
```

Le code est fonctionnel et on récupère des couples (V_{mes} et T).

Code pour l'étalonnage et la prise de mesure simulltanée des deux capteurs

```
#include <Wire.h>
#include <SparkFun_TMP117.h>  // Biblioth que SparkFun TMP117

// Constantes pour la thermistance
const int thermistorPin = A0;  // Broche analogique pour le pont diviseur de
    tension
const float Rref = 10000.0;    // R sistance de r f rence de 10k ohms
const float Vcc = 5.0;         // Tension d'alimentation de l'Arduino (5V)

// Coefficients de Steinhart-Hart pour la thermistance (ajustez selon votre
    thermistance)
const float A = 0.001129148;
const float B = 0.000234125;
const float C = 0.0000000876741;

// Capteur TMP117 (SparkFun)
TMP117 tmp117;

void setup() {
```

```
  Serial.begin(115200);

  // Initialisation du capteur TMP117 SparkFun
  if (tmp117.begin() == false) {
    Serial.println("Erreur : Impossible de d tecter le capteur TMP117 SparkFun !");
    while (1);
  }

  Serial.println("Capteur TMP117 SparkFun pr t.");
}

void loop() {
  // --- Lecture du capteur TMP117 SparkFun ---
  float tmp117Temperature = tmp117.readTempC();  // Lecture de la temp rature en degr s Celsius

  // --- Lecture de la thermistance en diviseur de tension ---
  int adcValue = analogRead(thermistorPin);  // Lecture de la valeur analogique
  float Vout = (adcValue * Vcc) / 16384.0;     // Conversion en tension
  float Rth = Rref * (Vcc / Vout - 1.0);     // Calcul de la r sistance de la thermistance

  // --- Calcul de la temp rature de la thermistance avec Steinhart-Hart ---
  float lnRth = log(Rth);
  float tempK = 1.0 / (A + B * lnRth + C * pow(lnRth, 3));  // Temp rature en Kelvin
  float thermistorTempC = tempK - 273.15;  // Conversion en degr s Celsius

  // --- Affichage des r sultats ---
  Serial.print("Temp rature TMP117 : ");
  Serial.print(tmp117Temperature);
  Serial.print("  C , ");

  Serial.print("Temp rature thermistance : ");
  Serial.print(thermistorTempC);
  Serial.println("  C ");

  // ---   talonnage   ---
  float calibrationOffset = tmp117Temperature - thermistorTempC;
  Serial.print("D calage pour   talonnage   (Offset) : ");
  Serial.print(calibrationOffset);
  Serial.println("  C ");

  delay(1000);  // Pause d'une seconde avant la prochaine lecture
}
```

Après passage d'un encadrant on a modifié le code pour :

1. Récupérer toutes les secondes un couple tension (diviseur de tension) / température (TMP117)
2. Enregistrer ces couples dans un tableau sur la mémoire locale de l'Arduino
3. Exporter ce tableau et tracer la courbe correspondante dans Python