TP –MESURE DE TEMPERATURE CAPTEUR NUMERIQUE - ANALOGIQUE

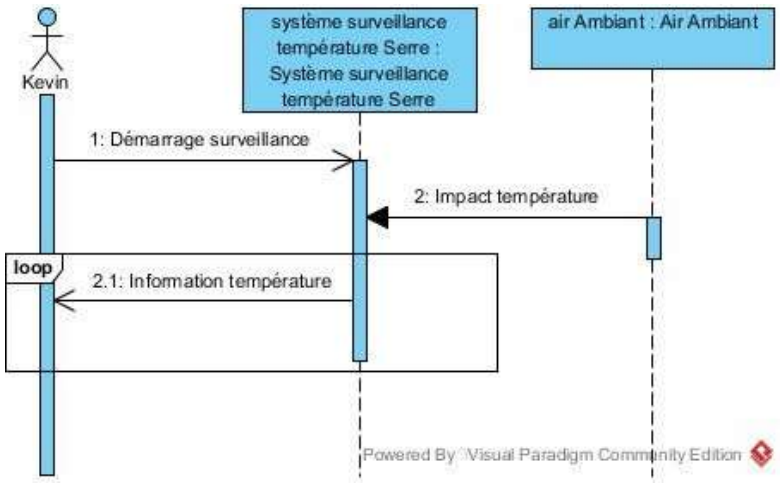
Objectifs :

- **CARACTERISER** une chaine d'information comportant comprenant une carte Arduino, des **capteurs** et des **afficheurs** d'information,
- **SPECIFIER** un comportement à l'aide du diagramme d'activité.
- **COMPARER** une information **analogique** et une information **numérique** (Bus I2C)

Situation :

Kevin trouve que les aliments qu'il achète au supermarché ne sont pas toujours de très bonne qualité gustative. Il souhaite mettre en place une serre sur son balcon, et souhaite donc réaliser rapidement un prototype de suivi de température.

Le fonctionnement du prototype de suivi de température peut être spécifié par le diagramme suivant :



Remarque : dans cette activité, nous n'étudierons pas le système de chauffage/ventilation de la serre.

Plan de l'activité :

Etape 1. Montage capteur analogique : TMP36.
Etape 2. Montage capteur numérique : TMP102.
Etape 3. Conclusion.

1 MESURE DE TEMPERATURE – CAPTEUR ANALOGIQUE

*Dans un premier temps, le choix de Kévin se porte vers une solution Arduino + Capteur analogique (**TMP36**) ; étudions alors la faisabilité et la pertinence de cette solution.*

Objectifs :

- **Utiliser** la fenêtre "**Moniteur série**" pour **lire** une valeur issue d'un **capteur de température**.
- **Décrypter** un programme et **afficher** une grandeur mesurée.

1.1 LE CAPTEUR DE TEMPERATURE

Le **TMP36** est un capteur facile à utiliser. Une fois alimenté, il va délivrer une tension analogique proportionnelle à la température.

- **Tension d'alimentation** : 2,7V à 5,5V
- **Etendu de mesure :** -40°C à 125°C
- **Précision** : ± 2°C
- **Echelle** : 10mV/°C
- **Calibration** : 750mV à 25°C
- **Compléments d'information :** TMP35_36_37.pdf

Ce produit est facilement utilisable à travers un pin analogique d'une carte Arduino.

1.2 MONTAGE DU CAPTEUR

- En fonction du **code du programme** (sur la page suivante) et des données de **la fiche technique**, relier le capteur à la carte Arduino (schéma de montage).

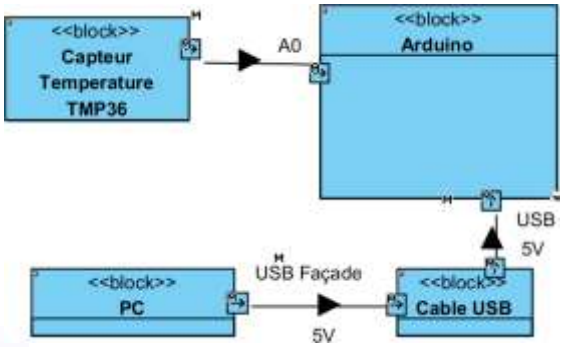
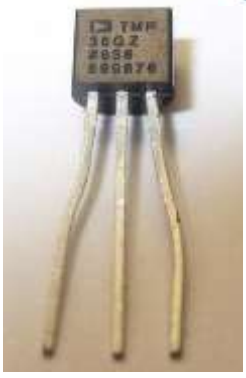


Schéma de montage :



<u>Sur le document réponse :</u>

- **TRACER** la caractéristique (voir la fiche technique) donnant la tension U (exprimée en V) en fonction de la température θ (en °C), **ECRIRE** l'équation U = f (θ) puis en **DEDUIRE** θ = f (U).
- **RECHERCHER** la relation entre les unités °C et °F.

1.3 PROGRAMMATION DE LA CARTE ARDUINO

```
void setup()

{
Serial.begin(9600);
}

void loop()

{
  float tension, degresC, degresF;
  tension = analogRead(0)* (5.0/1024.0);
  degresC = (tension - 0.5) * 100.0;
  degresF = degresC *(9.0/5.0) + 32.0;
  Serial.print("Tension : ");
  Serial.print(tension);
  Serial.print("    deg C : ");
  Serial.print(degresC);
  Serial.print("    deg F : ");
  Serial.println(degresF);
  delay(1000);
 }
```

- **FAIRE** le montage et **DEMANDER** au professeur de vérifier votre travail.
- **COPIER** le programme fourni et le **COLLER** dans le logiciel Arduino, **VERIFIER** s'il n'a pas d'erreur et le **TELEVERSER** sur la carte Arduino.

1.4 EXPLOITATION DE LA MESURE

- Dans le menu « Outils », **OUVRIR** le « Moniteur série ».

Assurez-vous que la vitesse de transmission au bas de la fenêtre est définie sur 9600.

Noter aussi que chaque fois que vous téléversez un nouveau programme, la fenêtre du moniteur de série se ferme. Elle le fait parce que le port série est également utilisé pour télécharger le code.

Lorsque le téléversement est terminé, vous pouvez ré-ouvrir la fenêtre du moniteur accessible directement par la loupe en haut à droite de la fenêtre.

Pour envoyer des données de l'Arduino à la fenêtre du moniteur série on utilise la fonction **Serial.print(),** on peut imprimer des variables ou du texte entre guillemets. **Serial.println()** permet le passage à la ligne suivante.

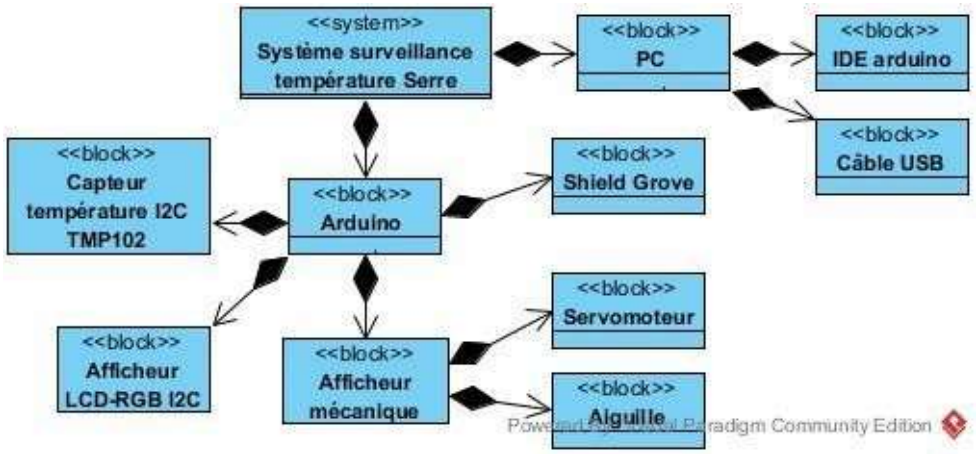
- **OBSERVER** ce qui se passe sur le moniteur série.
- Sur le document réponse, **COMMENTER** succinctement les lignes du programme.

➢ **SAUVEGARDER bien votre programme : il sera utile pour la suite...**

2 MESURE DE TEMPERATURE : **CAPTEUR ANALOGIQUE**

*Dans un second temps, le choix de Kévin se porte vers une solution **Arduino + Capteur numérique sur Bus I2C** ; étudions alors la faisabilité et la pertinence de cette solution.*

Le matériel sera composé des éléments suivants :



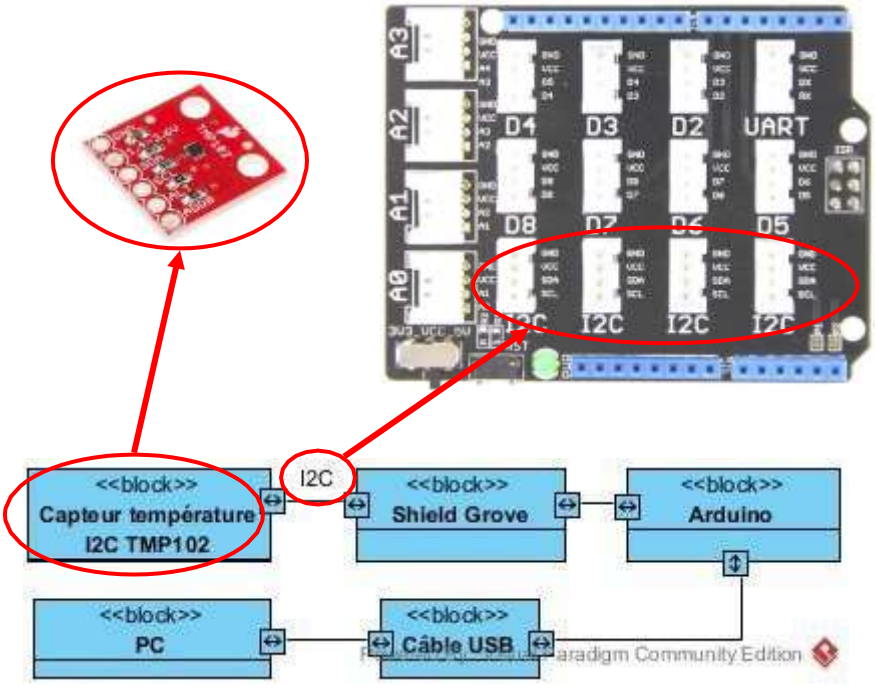
2.1 MESURE DE LA TEMPERATURE

Kevin a trouvé sur Internet un Capteur bon marché qui semblait pouvoir convenir à son besoin. L'objectif de cette partie est d'étudier ce capteur et de vérifier si ce choix peut être intéressant. Nous allons :

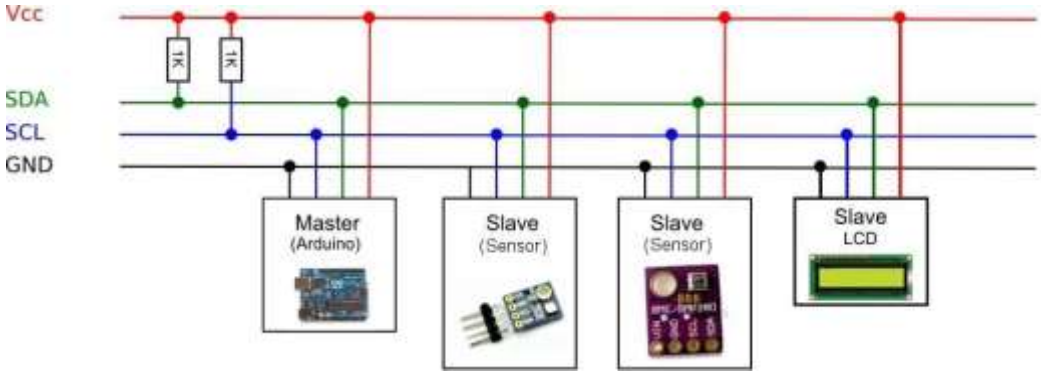
1. **Brancher le capteur TMP 102.**
2. **Détecter le capteur sur le bus I2C.**
3. **Calculer la température en fonction des données renvoyées par le capteur.**
4. **Afficher directement la température au moyen d'une bibliothèque.**

2.2 BRANCHEMENT DU CAPTEUR TMP 102

- BRANCHER le capteur de température sur n'importe quel port I2C au choix :

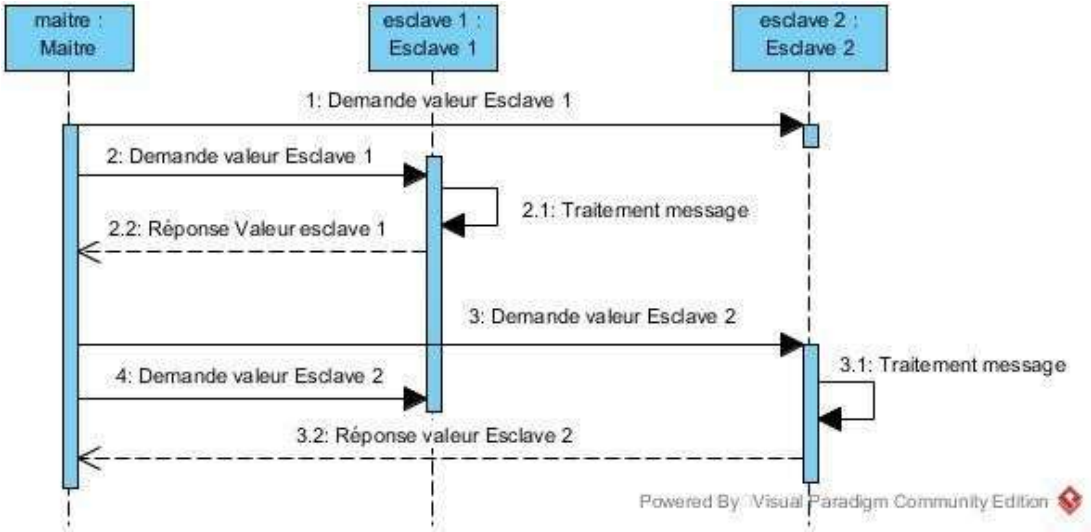


Le bus I2C permet **de connecter un grand nombre d'éléments** à la Carte Arduino. Il comporte **4 fils** : +5V ; 0V ; SDA ; SDC. Les éléments sont branchés en **dérivation** sur **ces 4 fils** :



Le Bus I2C comporte un ***Maitre*** et des ***Esclaves***. Le fonctionnement peut être décrit en première approximation comme suit :

1. Le **maître** envoie une information, ou une demande d'information à tous les **esclaves**.
2. **L'esclave** concerné récupère le message, les autres n'en tiennent pas compte.
3. **L'esclave** répond, les autres se taisent.
4. Le **maître** reprend la main sur le **Bus I2C**.



- <u>Pour le branchement présenté :</u>
  - **IDENTIFIER** le maître puis l'esclave sur le réseau **I2C** créé ?
  - **REPERER** sur la figure ci-dessous les broches d'alimentation du capteur et celles de transmission du signal.

2.3 DETECTION DU CAPTEUR I2C

Tous les éléments étant branchés en dérivation, **chacun doit avoir une adresse unique** sur le Bus **pour pouvoir communiquer** (de manière analogue à une adresse IP sur un réseau TCP/IP). **Ces adresses** sont hexadécimales, allant de **0x04** à **0x7B** (les autres sont réservées).

Pour l'instant, nous **ne connaissons pas l'adresse I2C** du capteur. Heureusement, il existe des programmes faisant office de ***Scanner I2C*** sur le bus en question !!!

- **TELEVERSER** le scanner suivant dans l'IDE Arduino puis **TELEVERSER** le dans Arduino :

```
#include<Wire.h>

void setup()
{
	Wire.begin();
	Serial.begin(9600);
	while (!Serial);
	Serial.println("\nI2C Scanner");
}
void loop()
{
	byte error, address;
	int nDevices;

	Serial.println("Scanning...");

	nDevices = 0;
	for (address = 1; address < 127; address++ )
	{
	    Wire.beginTransmission(address);
	    error = Wire.endTransmission();

	    if (error == 0)
	    {
	        Serial.print("I2C device found at address 0x");
	        if (address < 16)
	            Serial.print("0");
	         Serial.print(address, HEX);
	        Serial.println("     !");
	        nDevices++;
	    }
	    else if (error == 4)
	    {
	        Serial.print("Unknown error at address 0x");

	    if (address < 16)
	        Serial.print("0");
	    Serial.println(address, HEX);
	    }
	}

	if (nDevices == 0)
	    Serial.println("No I2C devices found\n");
	    else
	    Serial.println("done\n");
	    delay(5000);
	}
```

- **OUVRIR** la console de l'IDE Arduino (Moniteur Série) et **VERIFIER** qu'un seul capteur est branché sur le port I2C de la carte arduino.
- **Quelle est** l'adresse I2C du capteur de température ?

2.4 CALCUL DE LA TEMPERATURE MESUREE PAR LE CAPTEUR TMP102

On peut interroger le capteur I2C pour connaître la température qu'il a mesurée. Pour cela…

- **TELEVERSER** le programme suivant dans Arduino en remplaçant les caractères en rouge par l'adresse I2C identifiée du capteur :

```
#include<Wire.h>
void setup()
{
Wire.begin(); // wake up I2C bus
Serial.begin(9600);
}
void loop()
{
Wire.beginTransmission(?x??);  //  "Hey,  Message  for  you"
Wire.write(?x??); // "Move your register pointer back to 00h"
Wire.endTransmission(); // "Thanks, goodbye..."

// now get the data
Wire.requestFrom(?x??, 2); // "Send me the contents of your first two registers"
byte a = Wire.read(); // first received byte stored here
byte b = Wire.read(); // second received byte stored here

Serial.print("Octet A : ");
Serial.print(a);
Serial.print("Octet B : ");
Serial.print("\t");
Serial.print(b);
Serial.println();

delay(1000);
}
```

- **RELEVER** une ligne renvoyée par le capteur :
  - Lorsque le capteur est à **température ambiante** ;
  - Lorsque le capteur a été **tenu 30 secondes entre vos doigts**.

… à l'évidence, il va falloir la convertir dans un format plus compréhensible par l'être humain !

Lorsque l'on interroge le capteur TMP 102, il renvoie deux octets en décimal. A l'aide du convertisseur le convertir puis trouver la température :

Octet A								Octet B							
A7	A6	A5	A4	A3	A2	A1	A0	B7	B6	B5	B4	B3	B2	B1	B0

Lorsque les températures sont positives, la température mesurée se détermine ainsi :

Etape	Description	Exemple
1	On lit les 2 octets du capteur de température	Octet B : 1011 0000 Octet A : 0001 1010
2	On crée un nombre binaire en assemblant les 12 bits de A7 à B4 (en gardant cet ordre).	0001 1010 1011
3	On convertit alors ce chiffre en nombre décimal.	427
4	On multiplie ce chiffre par 0,0625 pour obtenir la température.	26,7°C

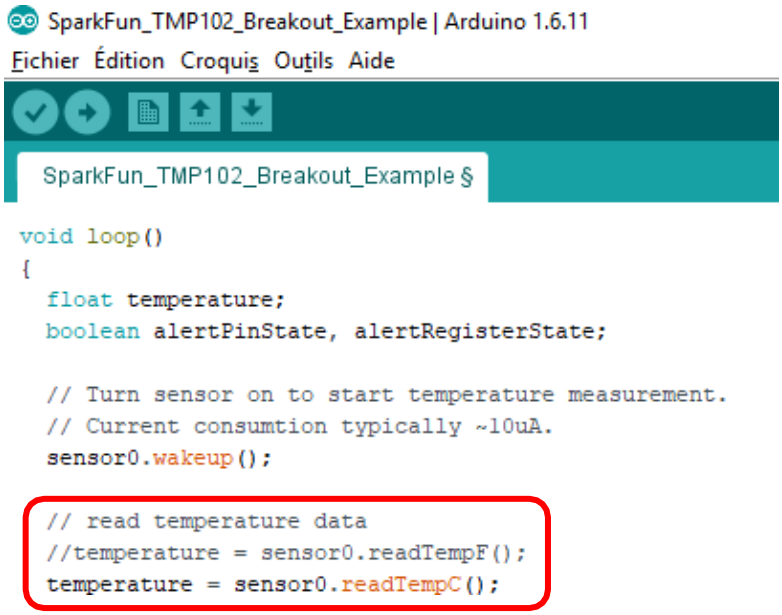
- **CALCULER** alors les températures mesurées lors de la question précédente. Les valeurs mesurées vous semblent-elles cohérentes ?

2.5 LECTURE DE LA REPONSE DU CAPTEUR TMP 102

- Il existe un moyen plus simple de lire la valeur renvoyée par le capteur : il s'agit d'utiliser la bibliothèque développée par le constructeur du capteur.

CHERCHER comment ajouter une bibliothèque au logiciel arduino

- Dans la partie "Loop", **MODIFIER** l'unité de mesure de la température : **METTRE** en commentaire la lecture de température en Fahrenheit, et **ACTIVER** la lecture de la température en degré :



- **LANCER** le programme de mesure de température en ouvrant la console.
- **Quelle est** la température ambiante mesurée ? Cela vous semblent-elles conformes à ce que l'on peut attendre ?

3 CONCLUSION - RENDU DE L'ACTIVITE

DETAILLER les points suivants :

- Présentation des deux types de capteur,
- Type d'information associé à chaque capteur,
- Traitement (matériel ou logiciel) associé à chaque capteur,

CONCLURE sur l'usage des capteurs.

NB : Vous devez étayer vos réponses à partir des cas traités dans l'activité.