

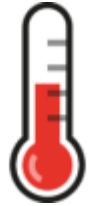


# Capteurs - Température

[Mise à jour le 28/5/2024]

## 1. Généralités

La température est une grandeur physique mesurée à l'aide d'un thermomètre et étudiée en thermométrie. Dans la vie courante, elle est reliée aux sensations de froid et de chaud, provenant du transfert thermique entre le corps humain et son environnement.



En physique, elle se définit de plusieurs manières : comme fonction croissante du degré d'agitation thermique des particules (en théorie cinétique des gaz), par l'équilibre des transferts thermiques entre plusieurs systèmes ou à partir de l'entropie (en thermodynamique et en physique statistique).

La température est une variable importante dans d'autres disciplines : météorologie et climatologie, médecine, et chimie.



- Ressource à consulter sur **Wikiversité** : [Capteur de température](#)
- Tableau comparatif des capteurs **Groove**
  - Fichier **Excel** à télécharger [ici](#)

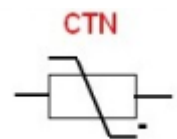
---

## 2. Capteurs analogiques

### 2.1 Thermistance

- **Généralités**

Résistance électrique dont la valeur varie rapidement en fonction de la température.

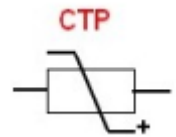


- **CTN**

Les CTN (Coefficient de Température Négatif, en anglais NTC, Negative Temperature Coefficient) sont des thermistances dont la résistance diminue de façon uniforme quand la température augmente. Leur modèle est donné ci-dessous.

$$R_{(T)} = R_0 \cdot e^{B \left[ \frac{1}{T} - \frac{1}{T_0} \right]}$$

R: Resistance in ambient temperature T (K)  
(K: absolute temperature)  
R<sub>0</sub>: Resistance in ambient temperature T<sub>0</sub> (K)  
B: B-Constant of Thermistor



- **CTP**

Les CTP (Coefficient de Température Positif, en anglais PTC, Positive Temperature Coefficient) sont des thermistances dont la résistance augmente avec la température. On distingue les thermorésistances (augmentation continue et régulière de la résistance avec la température, voir ci-dessus) des CTP dont la valeur augmente fortement avec la température dans une plage de température limitée (typiquement entre 0 °C et 100 °C).

## 2.2 CTN 10k



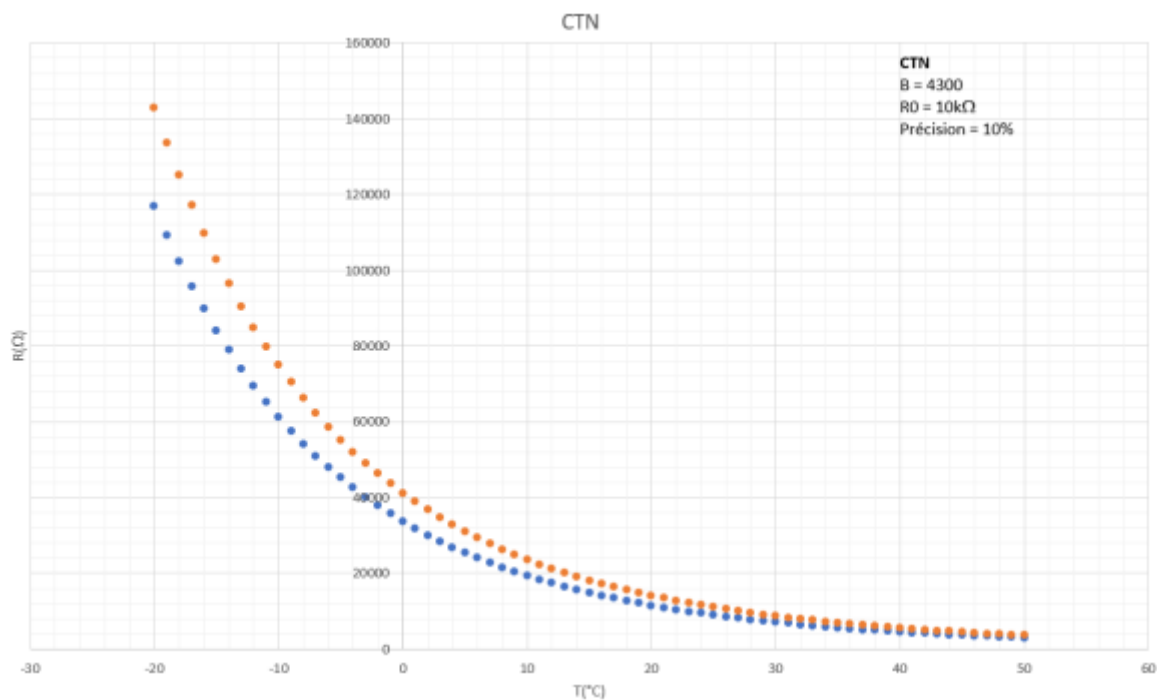
### 2.2.1 Présentation

- **Distributeur** : [Gotronic](#)
- **Caractéristiques**
  - Résistance à 25°C : 10 kΩ
  - Puissance: 0.25 W.
  - Tolérance: ±10%.
  - B=4300.




- **Documentation**
  - Fichier Acrobat Reader à télécharger [ici](#)
- **Modèle**

- Résistance à 25°C : 10 kΩ
- B=4300.
- Tolérance: ±10%.



- **Aide pour la *simulation de la chaîne de mesure***

- Les équations de la chaîne de mesure sont téléchargeables [ici](#) 
- Le modèle à simuler est téléchargeable [ici](#)

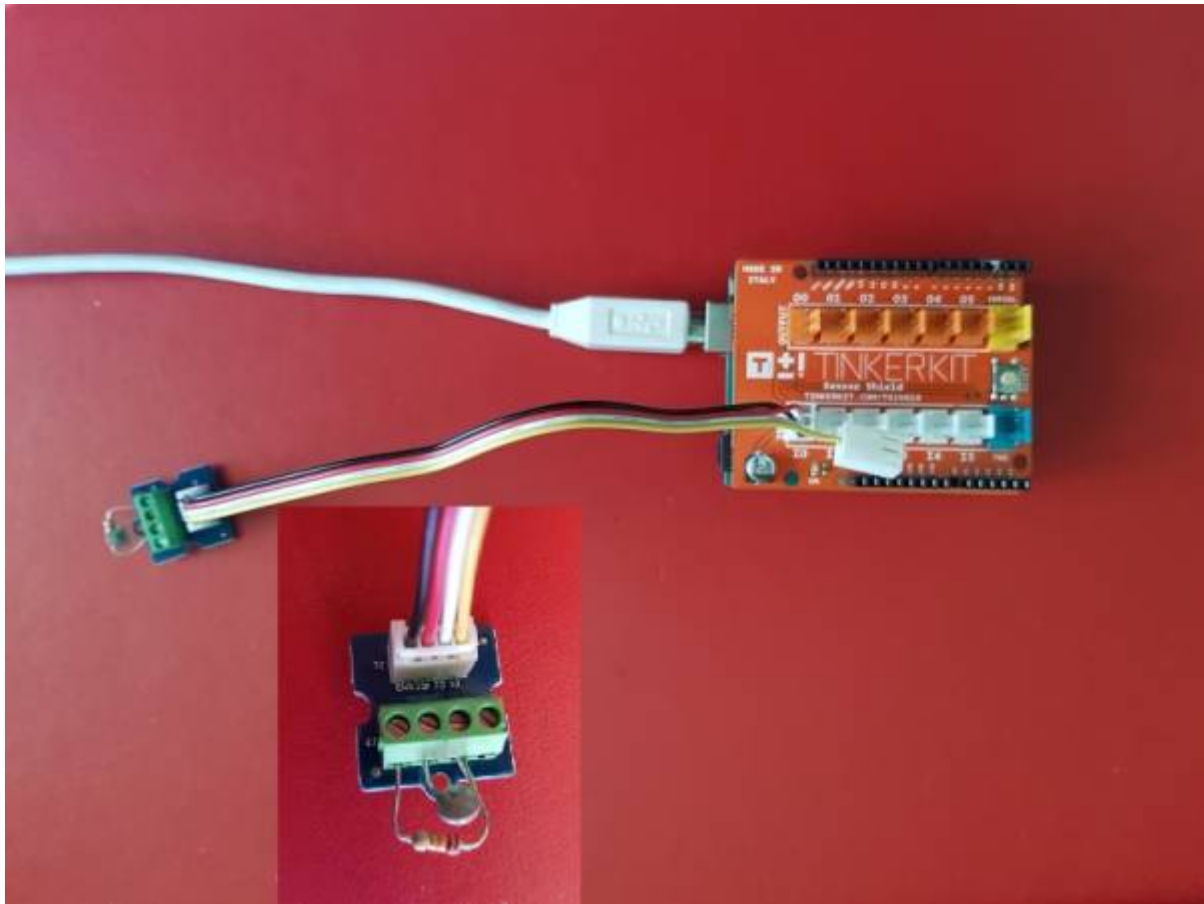
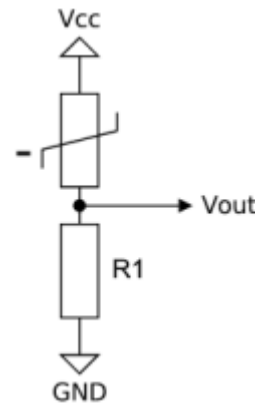
- **Algorithme**

- L'algorithme **T=f(N)** donnant la température en fonction de la sortie du convertisseur analogique / numérique est téléchargeable [ici](#)

## 2.2.2 Exemples de code

- [Arduino UNO](#)
- [ESP \(Arduino\)](#)
- [Rpi Pico \(μPython\)](#)

- **Connexion** à un shield **Tinkerkit v2** monté sur une Arduino Uno.



- **Exemple** pour tester le capteur



ctn.cpp

```
/*  
Mesure de la température ambiante avec une CTN  
Bibliothèque math.h : https://www.arduino.cc/en/math/h  
*/  
// Constantes  
//-----  
// CTN
```

```
const int Beta = 4300; // Kelvin
const float T0 = 298.15; // Kelvin (25°C)
const int R0 = 10000; // Résistance du capteur à 25°C
// Diviseur de tension
const int Vcc = 5; // Volt
const int R1 = 12000; // Ohm
// CAN
const int n = 10;
const int VPE = 5;

// Variables
//-----
int CTN = A0; // La CTN et son conditionneur sont connectés
sur la broche A0
int N = 0; // Image de la température, sortie du CAN
double temperature = 0.0; // Résultat du calcul de la température :
temperature=f(N)
// Coefficient du CAN
float kcan = pow(2, n) / VPE;
// Coefficients utilisés pour simplifier le calcul de la température
// k0, a
double k0 = kcan * Vcc * R1;
double a = R0 / exp(Beta / T0);
// k1, k2
double k1 = k0 / a;
double k2 = R1 / a;
//-----
void setup()
{
    Serial.begin(9600); // Fenêtre "serial" pour la mise au point
}
//-----
void loop()
{
    N = analogRead(CTN);
    temperature = Beta / log((k1 / N) - k2) - 273.15;
    Serial.println(temperature);
}
```

A venir  
A venir

## 2.3 Module SEN23292P



### 2.3.1 Présentation

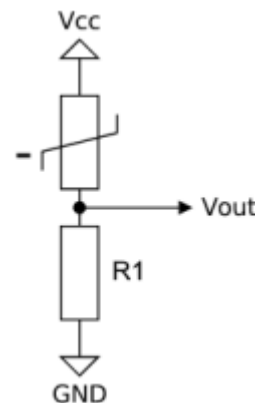
- **Source** : [wiki](#) seeed studio

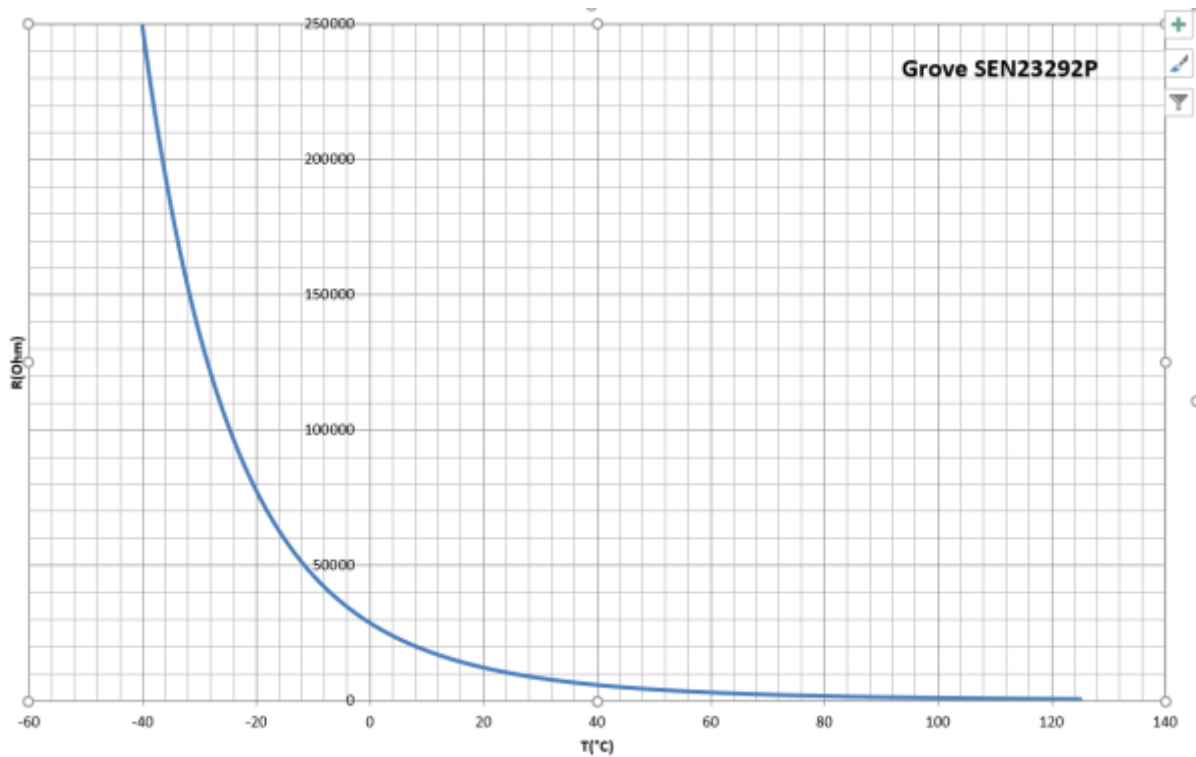
Ce capteur de température compatible Grove à CTN **NCP18WF104F03RC** délivre un signal analogique de 0 à 5 Vcc en fonction de la température mesurée.

- **Distributeur** : [Gotronic](#)
- **Caractéristiques**
  - Alimentation: 5 Vcc
  - Plage de mesure: -40 à +125 °C
  - Précision: 1,5 °C
  - Dimensions: 20 x 20 x 13 mm



- **Modèle**
  - PDF à télécharger [ici](#)





- **Aide pour la *simulation de la chaîne de mesure***

- Les équations de la chaîne de mesure sont téléchargeables [ici](#)
- Le modèle à simuler est téléchargeable [ici](#)

- **Algorithme**

- L'algorithme  **$T=f(N)$**  donnant la température en fonction de la sortie du convertisseur analogique / numérique est téléchargeable [ici](#)

### 2.3.2 Exemples de code

- [Arduino UNO](#)
- [ESP \(Arduino\)](#)
- [Rpi Pico \( \$\mu\$ Python\)](#)

- **Connexion** à un shield [Tinkerkit v2](#) monté sur une Arduino Uno.





Le capteur de température linéaire LM35 peut être utilisé pour détecter la température ambiante.

- **Distributeur** : [Gotronic](#)

- **Caractéristiques**

- Alimentation : 4 à 30 V
- Plage de mesure : -55 / +150°C
- Sensibilité : 10mV/°C
- Précision : +/-0,5°C (à 25°C)
- Boîtier : TO92



- **Documentation**

- PDF à télécharger [ici](#)

- **Modèle**

- Sensibilité : 10mV/°C



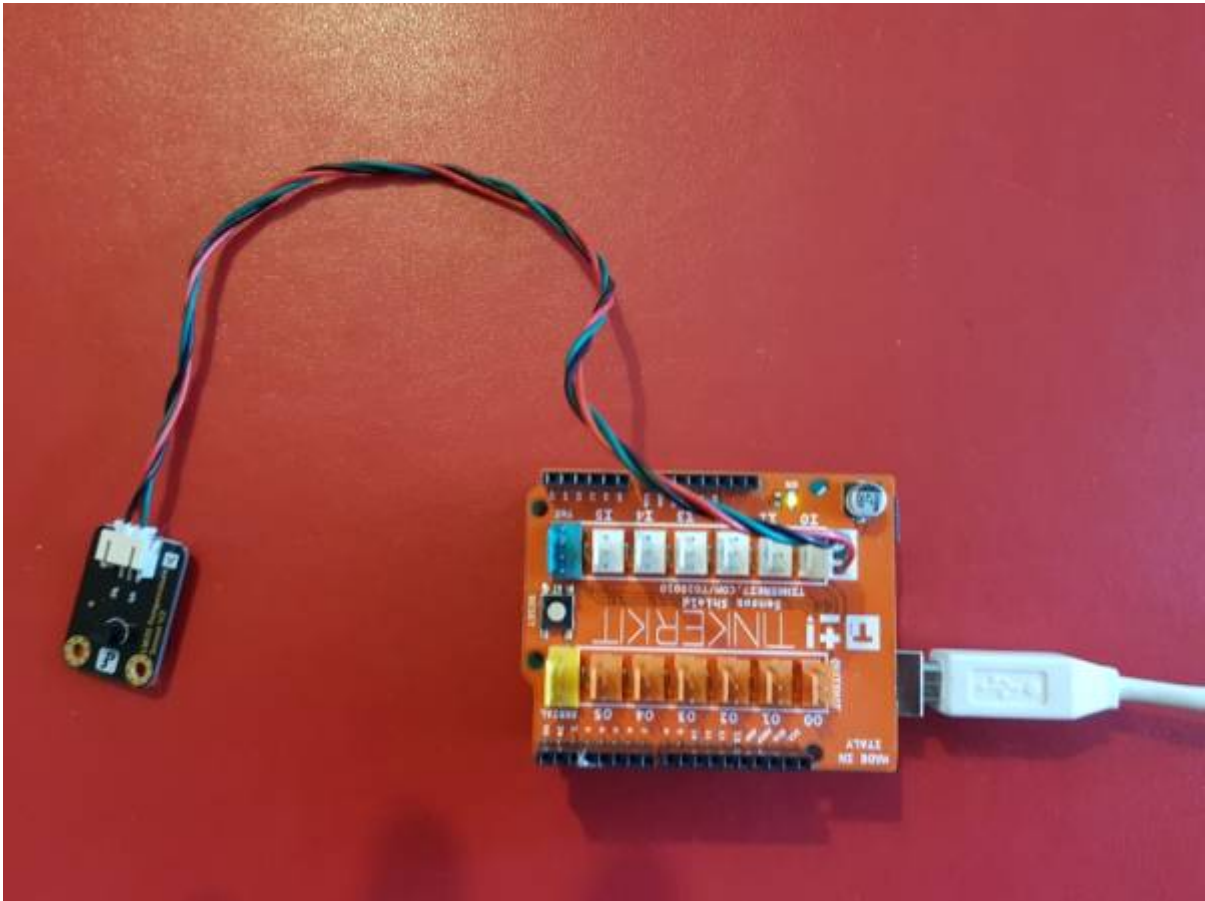
- **Aide pour la simulation** de la chaîne de mesure

- Le modèle à simuler est téléchargeable [ici](#)

### 3.1.2 Exemples de code

- [Arduino UNO](#)
- [ESP \(Arduino\)](#)
- [RPi Pico \(µPython\)](#)

- **Connexion** à un shield [Tinkerkit v2](#) monté sur une Arduino Uno



- **Exemple** pour tester le capteur



### lm35.cpp

```
void setup()
{
  Serial.begin(9600); // Débit binaire : 9600 bps
}

void loop()
{
  uint16_t N;
  double temperature;
  // Lecture
  N=analogRead(A0); //LM35 connecté à Analog 0
  // Traitement
  temperature = (double) N * (5/10.24);
  // Ecriture
  Serial.print("Température:"); // Affiche la température sur le
moniteur
  Serial.print(temperature);
  Serial.println("C");
  delay(1000);
}
```

}

### Pour aller plus loin

Mesurer une température négative ([Télécharger](#))

A venir

A venir

## 3.2 MM111



Capteur de température Velleman basé sur un amplificateur **MCP6L01T-E / LT** permettant de mesurer la **température** entre -50 et 150 °C. Il communique avec un microcontrôleur type Arduino ou compatible via une liaison analogique.

- **Distributeur** : [Gotronic](#)
- **Caractéristiques**
  - Alimentation: 5 Vcc
  - Plage de mesure: -50 à +150 °C
  - Sensibilité: 21 mV/°C
  - Précision: 0,4 °C
  - Dimensions: 22 x 22 x 5 mm



- **Documentation**
  - PDF à télécharger [ici](#)

## 4. Capteurs numériques

### 4.1 MLX90614



#### 4.1.1 Présentation

- **Source** : [wiki](#) DFROBOT

Le MLX90614 est un capteur de température infrarouge sans contact basé sur le principe de Stefan-Boltzmann.

- **Distributeur** : [Gotronic](#)
- **Caractéristiques**
  - Alimentation: 3,3 à 5 Vcc
  - Consommation: 1,2 mA
  - Plage de mesure: -70 à 382 °C
  - Résolution: 0.01 °C
  - Interface I2C ([SMBus](#)) : **SLA = 0x5A**
  - Dimensions: 32 x 18 mm

- **Schéma-blocs**

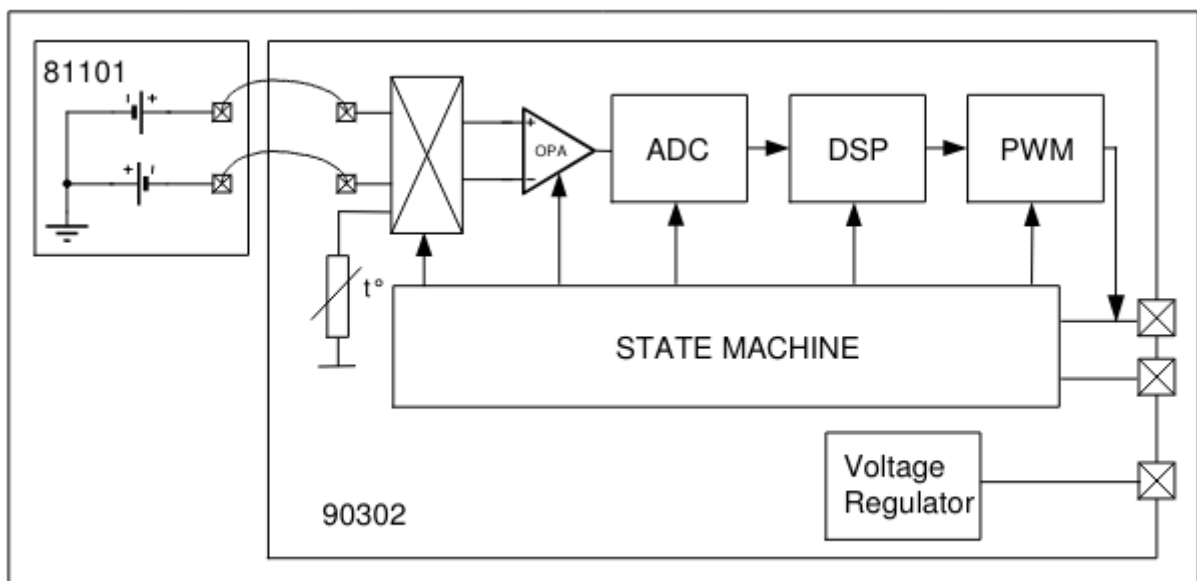


Figure 3: block diagram

- DSP : processeur du signal numérique

- **Principe de fonctionnement**

- Rayonnement Infrarouge (IR) : Tout objet émet un rayonnement infrarouge proportionnel à sa température. Le MLX90614 capte ce rayonnement.
- Détecteur Thermopile : Le module du capteur MLX90614 comprend un détecteur thermopile infrarouge. Ce détecteur collecte le rayonnement infrarouge dans son champ de vision.
- Conversion en Signal Numérique : Le signal électrique généré par le thermopile est converti en un signal numérique par l'unité de traitement du signal DSP (Digital Signal Processing).
- Communication I2C : Le capteur transmet ensuite ces lectures de température via le bus I2C au microcontrôleur.



- **Documentation**

- Datasheet à télécharger [ici](#)
- Schéma du module à télécharger [ici](#)



- **Chronogrammes**

- Relevé des signaux du bus I2C faite lors d'une mesure :  $T_A=19,55^{\circ}\text{C}$  et  $T_{\text{obj}}=-13,73^{\circ}\text{C}$ . A télécharger [ici](#).

#### 4.1.2 Bibliothèques

- [Arduino](#)
- [ESP \(Arduino\)](#)
- [Rpi Pico \(µPython\)](#)

- **Bibliothèque** à installer dans l'IDE Arduino

Adafruit MLX90614 Library par Adafruit

2.1.5 installed

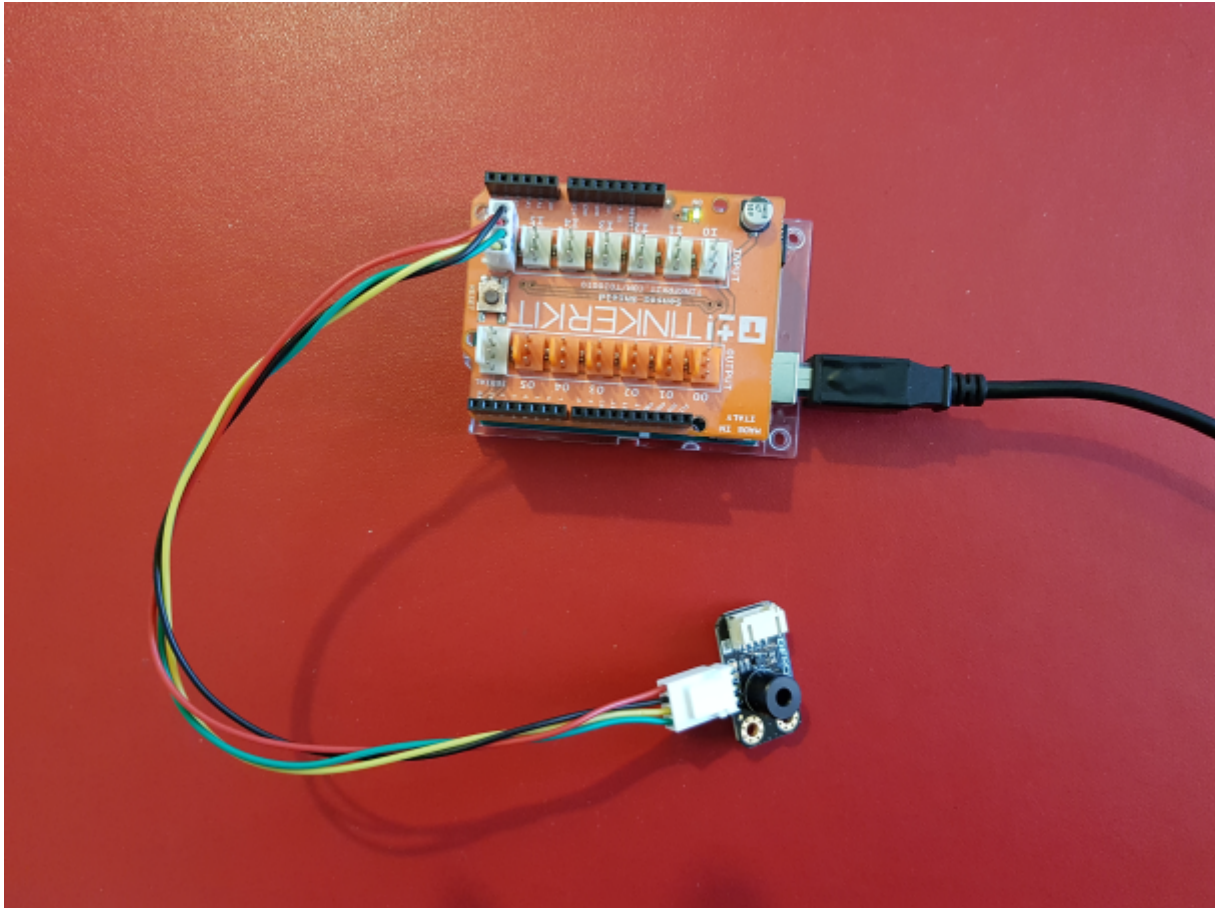
Arduino library for the MLX90614 sensors in the Adafruit shop Arduino library for the MLX90614 sensors in the Adafruit shop

A venir

A venir

#### 4.1.3 Exemples de code

- [Arduino](#)
- [ESP \(Arduino\)](#)
- [Rpi Pico \(µPython\)](#)
  
- **Ressource** : [wiki](#) DFROBOT
- **Connexion** à un shield [Tinkerkit v2](#) monté sur une Arduino Uno



- **Exemple** de l'IDE Arduino pour tester le capteur  
Dans l'IDE Arduino, sélectionner : Fichier → Exemples → Adafruit MLX90614 Library → mlxtest
- **Résultats** dans la console

```
Ambient = 23.21*C      Object = 20.65*C  
Ambient = 73.78*F     Object = 69.17*F
```

A venir  
A venir

## 4.2 TMP102



#### 4.2.1 Présentation

- **Source** : [wiki Sparkfun](#)

Le TMP102 est capable de lire des températures avec une résolution de 0,0625 °C et une précision allant jusqu'à 0,5 °C. La sortie possède des résistances intégrées de 4,7kΩ pour les communications I2C et fonctionne de 1,4V à 3,6V. La communication I2C utilise une signalisation à drain ouvert, il n'est donc pas nécessaire d'utiliser le décalage de niveau.

- **Distributeur** : [Mouser](#)
- **Caractéristiques**
  - Alimentation: 1,4 à 3,6 Vcc
  - Consommation: 10 μA maxi (1 μA en veille)
  - Plage de mesure: -40 °C à +125 °C
  - Précision: 0,5 °C (de -25 °C à +85 °C)
  - Résolution: 12bits, 0,0625 °C
  - Interface I2C : **SLA = 0x48**
  - Dimensions: 16 x 16 mm



- **Documentation**
  - PDF à télécharger : [Datasheet Résumé Datasheet](#)
  - Schéma à télécharger [ici](#)

#### 4.2.2 Bibliothèques

- [Arduino UNO](#)
- [ESP \(Arduino\)](#)
- [Rpi Pico \(μPython\)](#)
- **Ressource** : [Description de la bibliothèque sparkfun et exemple de code](#)
- **Bibliothèque** à installer dans l'IDE Arduino

**SparkFun TMP102 Breakout**  
by SparkFun Electronics  
**A library to drive the Texas Instruments TMP102 using I2C.** Communicates with the TMP102 over I2C to quickly integrate a temperature sensor into your project.  
[More info](#)

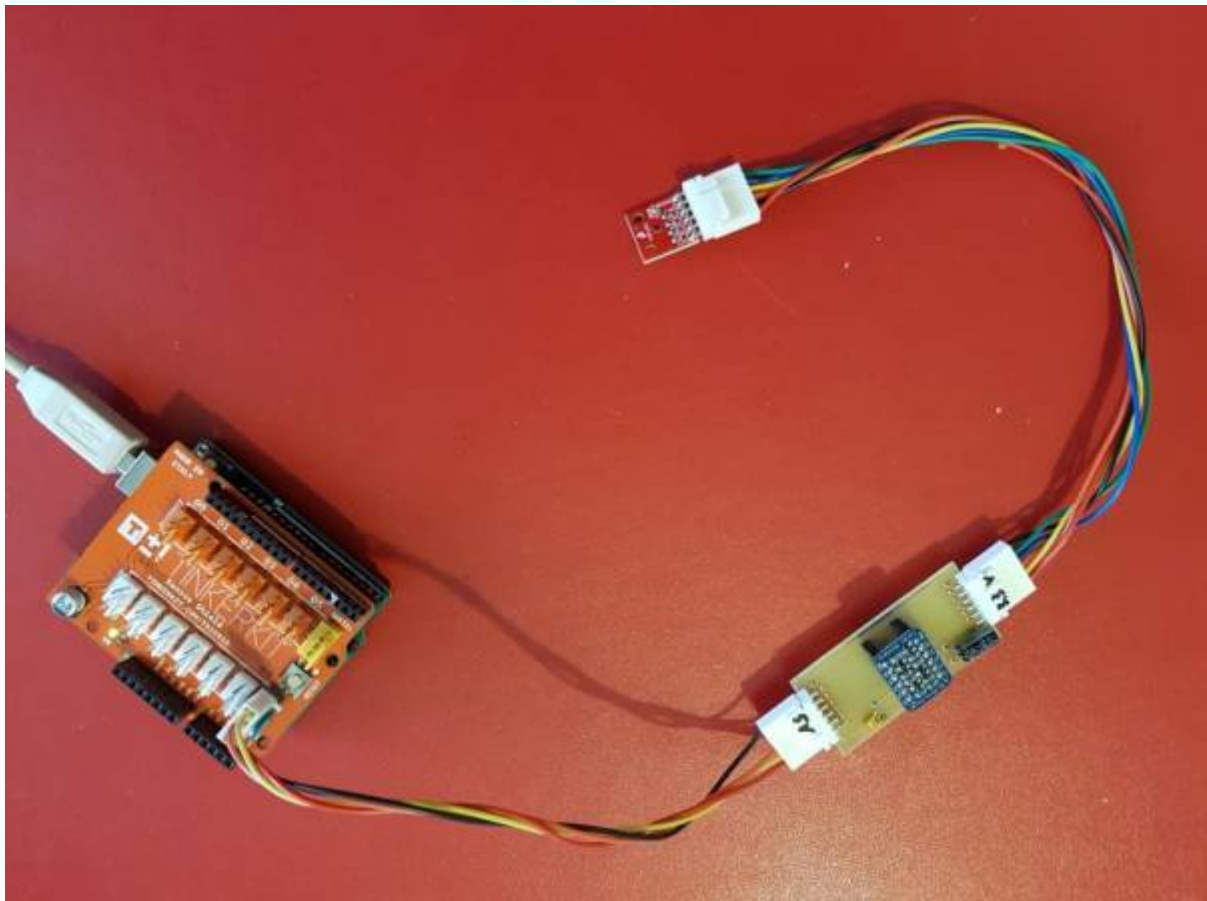
Version 1.1.2

A venir  
A venir



#### 4.2.3 Exemples de code

- [Arduino UNO](#)
- [ESP \(Arduino\)](#)
- [Rpi Pico \(µPython\)](#)
- **Connexion** à un shield [Tinkerkit v2](#) monté sur une Arduino Uno.





- **Exemple de l'IDE pour tester le capteur**  
Dans l'IDE Arduino, sélectionner : Fichiers → Exemple → SparkFun\_TMP102\_Breakout → **Example1\_Basic\_Temperature\_Readings**
- **Résultat dans la console**

```
CONSOLE DE DÉBOGAGE  SORTIE  TERMINAL
Temperature: 21.94  Alert Pin: 1
Temperature: 21.94  Alert Pin: 1
Temperature: 21.94  Alert Pin: 1
Temperature: 21.87  Alert Pin: 1
```

A venir

A venir

## 4.3 TMP117



### 4.3.1 Présentation

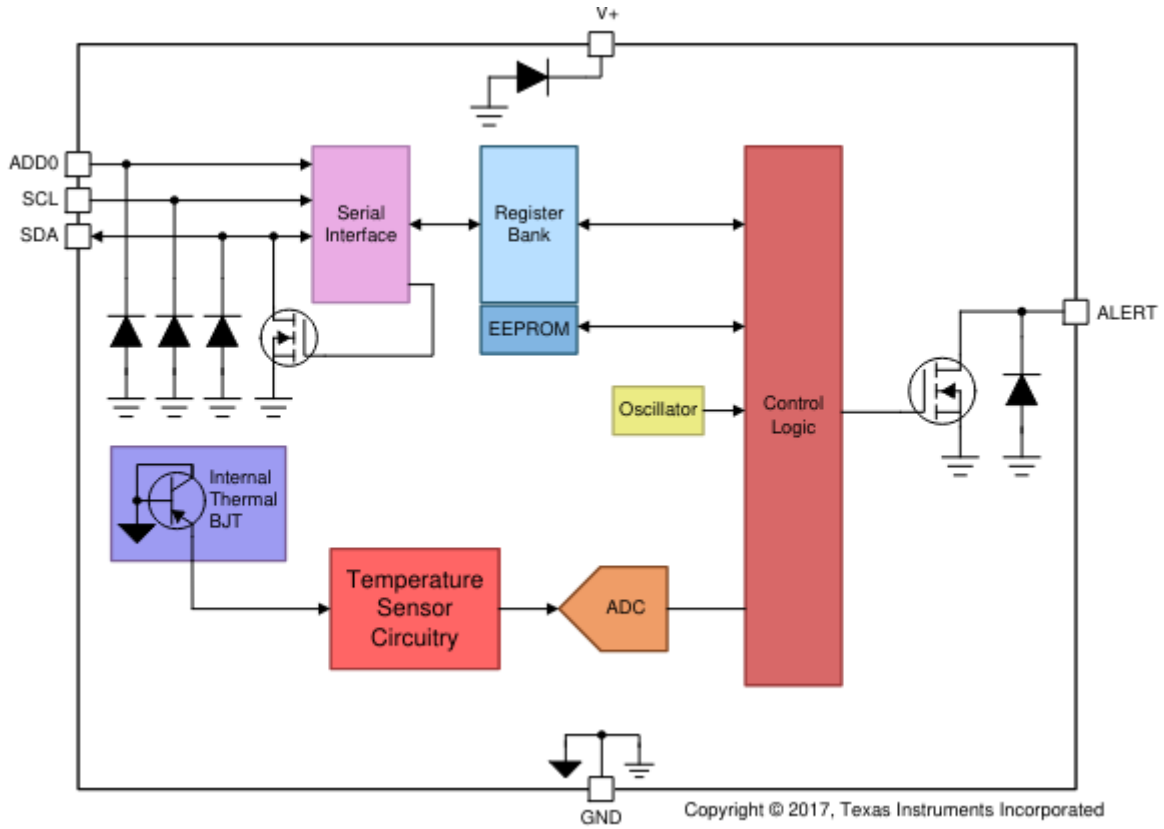
- **Source** : [wiki](#) Sparkfun

Le TMP117 fournit un résultat de mesure de la **température sur 16 bits** avec une **résolution de 0,0078 °C** et une **précision allant jusqu'à ±0,1 °C** sur la plage de température de **-20 °C à 50 °C** sans étalonnage. Le TMP117 possède une interface compatible I2C et SMBus™, une fonctionnalité d'alerte programmable, et l'appareil peut prendre en charge jusqu'à quatre appareils sur un seul bus. Une EEPROM intégrée est incluse pour la programmation de l'appareil avec une mémoire supplémentaire de 48 bits disponible pour une utilisation générale.

- **Distributeur** : [Mouser](#)
- **Caractéristiques**
  - Alimentation: 1,8 à 5,5 Vcc (3,3V avec câble Qwiic)
  - Consommation: 3,5 µA maxi (150 nA en veille)
  - Plage de mesure: -55 °C à +150 °C
  - Précision: 0,1 °C (de -20 °C à +50 °C)

- Résolution: 16bits, 0,0078 °C
- Interface série I2C : **SLA = 0x48**

• **Schéma-blocs**



• **Documentation**

- Datasheet à télécharger : [ici](#)
- Schéma du module à télécharger [ici](#)



• **Chronogrammes**

- Relevés des signaux du bus I2C : **T=18,38°C** (Tektronix MSO2012) et **T=20°C** (Analyseur IKALogic). A télécharger [ici](#).

**4.3.2 Bibliothèques**

- [Arduino UNO](#)
- [ESP \(Arduino\)](#)
- [Rpi Pico \(µPython\)](#)

- **Bibliothèques** à installer dans l'IDE Arduino

**Adafruit TMP117** par Adafruit

1.0.3 installed

Arduino library for the TMP117 sensors in the Adafruit shop Arduino library for the TMP117 sensors in the Adafruit shop

ou

**SparkFun High Precision Temperature Sensor TMP117 Qwiic** par SparkFun Electronics

1.2.5 installed

A library to drive the Texas Instruments TMP117 by I2C. Communicates with the TMP117 over I<sup>2</sup>C to quickly integrate a temperature sensor into your project. The sensor outputs temperature readings with high precision of +/- 0.1°C over the range of -20°C to +50°C with no calibration. The maximum...

- **Ressource** : [Description de la bibliothèque sparkfun et exemple de code](#)

**SparkFun High Precision Temperature Sensor TMP117 Qwiic**

by SparkFun Electronics Version 1.2.4 **INSTALLED**

**A library to drive the Texas Instruments TMP117 by I2C.** Communicates with the TMP117 over I<sup>2</sup>C to quickly integrate a temperature sensor into your project. The sensor outputs temperature readings with high precision of +/- 0.1°C over the range of -20°C to +50°C with no calibration. The maximum range is from -55°C to 150°C with a slightly lower precision of +/-0.3°C. It also has a very low power consumption which minimizes the impact of self-heating on measurement accuracy. The sensor operates from 1.8V to 5.5V.

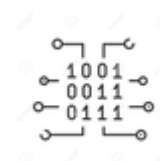
[More info](#)

Sélectionner une version ▾

Installer

- **GitHub**
  - [SparkFun\\_TMP117\\_Arduino\\_Library](#)

A venir



**4.3.3 Exemples de code**

- [Arduino UNO](#)
- [ESP \(Arduino\)](#)
- [Rpi Pico \(µPython\)](#)
- **Exemple de l'IDE Arduino pour tester le capteur.** Dans l'IDE Arduino, sélectionner :
  - **Adafruit** : Fichier → Exemples → Adafruit TMP117 → **basic\_test**
  - **SparkFun** : Fichier → Exemples → SparkFun High Precision Temperature Sensor TMP117 Qwiic → **Example1\_BasicReadings**

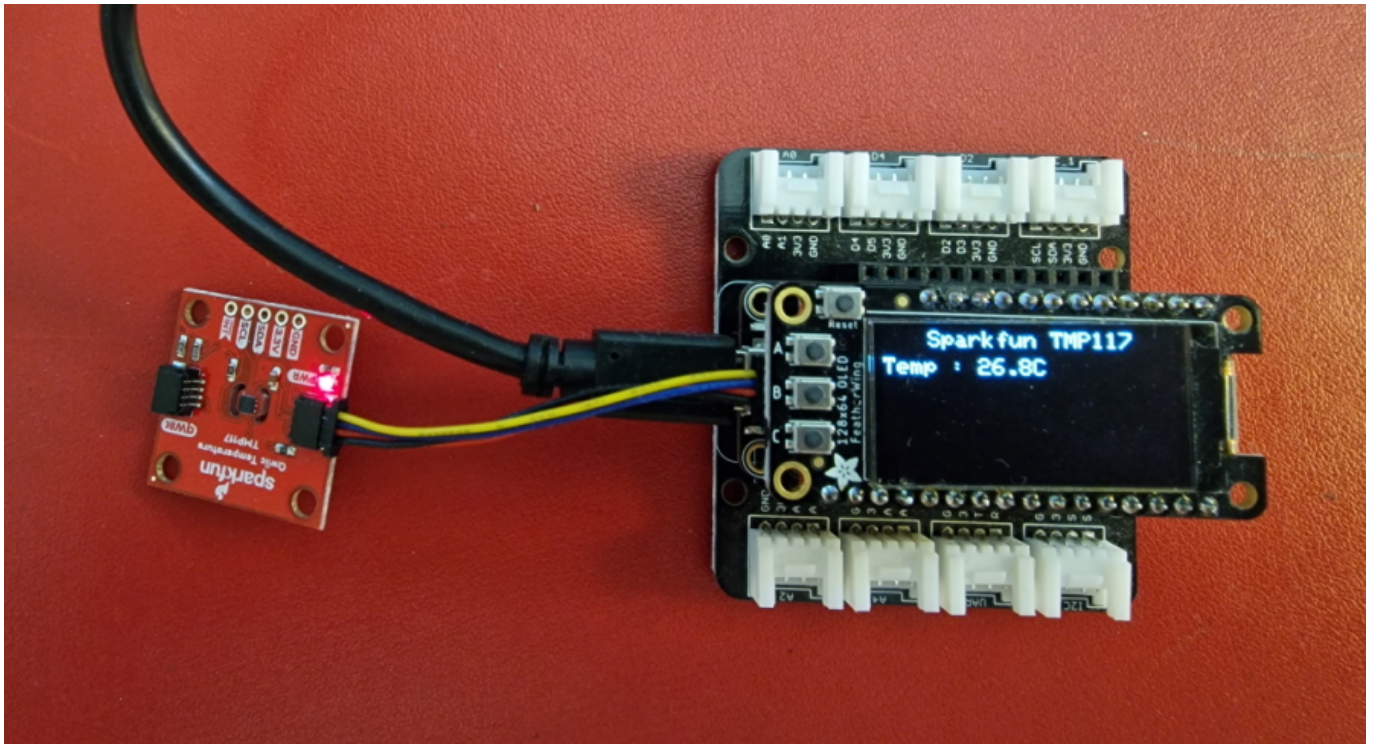
- **Un premier exemple pour tester le capteur avec l'IDE Arduino**

→ Fichier → Exemples → SparkFun\_High\_Precision\_Temperature\_Sensor\_TMP117\_Qwiic → **Example1\_Basic\_Readings.ino**



- **Mise en oeuvre avec un afficheur OLED**

- *Description* : mesure de la température à l'aide d'un capteur **Sparkfun TMP117**, test des boutons-poussoirs et affichage sur un écran Oled **Adafruit SH1107**. L'écran et le capteur sont reliés via le système **Qwiic** de Sparkfun.



- **Matériels**

- Carte à microcontrôleur : [Adafruit Feather Huzzah ESP8266 + Support Particle](#)
- Afficheur : [Adafruit OLED SH1107](#)
- Capteur de température : [Sparkfun TMP117](#)

- **Bibliothèques** à installer dans l'IDE Arduino ou dans PlatformIO (VSCode)

- Adafruit GFX Library by Adafruit [\[GitHub\]](#)
- Adafruit SH110X by Adafruit [\[GitHub\]](#)
- SparkFun TMP117 by SparkFun [\[GitHub\]](#)

- **Code**



\*.cpp

```
// Matériels : Adafruit Feather Huzzah ESP8266 + Support Particle,  
Adafruit OLED SH1107, Sparkfun TMP117, câble Qwiic  
// Logiciel : Arduino
```

```
// A ajouter
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SH110X.h>
#include <SparkFun_TMP117.h>

#define BUTTON_A 0
#define BUTTON_B 16
#define BUTTON_C 2

// Constructeurs
Adafruit_SH1107 display = Adafruit_SH1107(64, 128, &Wire);
TMP117 sensor; // L'adresse du circuit TMP117 est 0x48 = (GND) par défaut

void setup()
{
  // Bus I2C
  Wire.begin(); // Initialisation
  Wire.setClock(400000);
  display.begin(0x3C, true); // L'adresse de l'afficheur est 0x3C par défaut

  // Configuration de l'affichage
  display.setRotation(1); // Affichage horizontal
  display.setTextSize(1);
  display.setTextColor(SH110X_WHITE);
  display.clearDisplay(); // Pour ne pas afficher le logo Adafruit chargé
  // automatiquement à la mise sous tension
  // Test de la communication avec le capteur
  if (sensor.begin() == false)
  {
    display.println("DEFAULT(s)");
    display.print("1. Le capteur TMP117 ne repond pas ! ");
    display.println();
    display.print("BLOPAGE du PROGRAMME");
    display.display(); // Transfert du buffer sur l'écran
    while (1)
      delay(10); // Blocage du programme
  }

  // Connexion des boutons-poussoirs
  pinMode(BUTTON_A, INPUT_PULLUP);
  pinMode(BUTTON_B, INPUT_PULLUP);
  pinMode(BUTTON_C, INPUT_PULLUP);
}

void loop()
{
```

```
// Efface le buffer
display.clearDisplay();

// Test des boutons
display.setCursor(0, 0);

if (!digitalRead(BUTTON_A))
  display.print("[A]");
if (!digitalRead(BUTTON_B))
  display.print("[B]");
if (!digitalRead(BUTTON_C))
  display.print("[C]");

// Titre
display.setCursor(20, 0);
display.println("Sparkfun TMP117");

// Mesure et affichage
// Data Ready est un indicateur de mode de conversion - en conversion
continue, l'indicateur dataReady doit toujours être haut
if (sensor.dataReady() == true) // Affiche les valeurs de température
que lorsque les données sont prêtes
{
  display.setCursor(0, 12);
  display.print("Temp : ");
  display.print(sensor.readTempC(), 1);
  display.print("C");
  delay(500);
  display.display(); // Transfert du buffer sur l'écran
}

delay(10);
}
```



[Télécharger](#) le projet PlatformIO pour VSCode.

A venir

#### 4.4 MR312



Capteur de température Microbot basé sur un TCN75A et permettant de mesurer une température entre -40 et +125 °C. Il communique avec un microcontrôleur via le bus I2C.

- *Distributeur* : [GoTronic](#)

## A terminer

### 4.5 DS18B20



#### 4.5.1 Présentation

- **Source** : Guide de mise en marche du module ST029 [pdf](#)

Ce module, basé sur le capteur DS18B20, permet de mesurer une température ambiante de -55°C à +125°C via un BUS One Wire.

- **Distributeur** : [GoTronic](#)
- **Caractéristiques**
  - Alimentation : 3 à 5,5 V
  - Plage de température : -55 à +125 °C
  - Précision : +/- 0,5 °C (entre -10 et 85 °C)

- Communication : 1-Wire
- Résolution : 9 à 12 Bits
- Dimensions: 23 x 21 x 10 mm

• **Schéma-blocs**

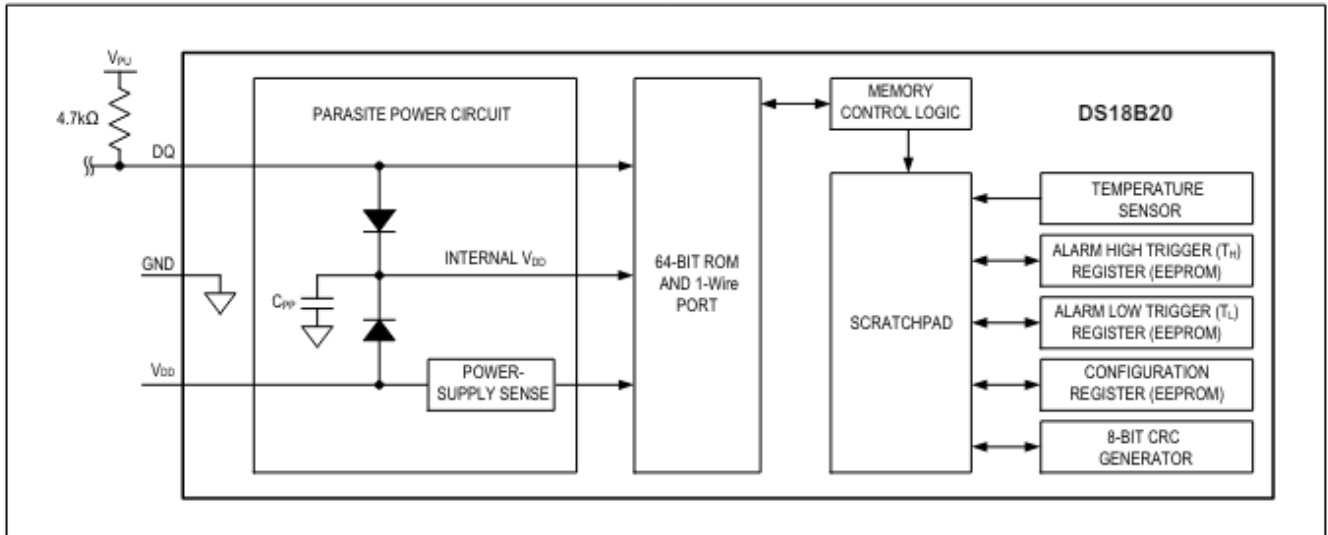


Figure 3. DS18B20 Block Diagram



• **Documentation**

- Présentation du bus **One-Wire de Dallas** (en français) à télécharger [ici](#)
- **Datasheet** du circuit 18B20 à télécharger [ici](#)



• **Chronogrammes**

- Relevé des signaux du bus One Wire de Dallas faite lors d'une mesure. A télécharger [ici](#).

4.5.2 Bibliothèques

- [Arduino UNO](#)
- [ESP \(Arduino\)](#)
- [Rpi Pico \(µPython\)](#)

• **Bibliothèques** à installer dans l'IDE Arduino



**OneWire** par Jim Studt, Tom Pollard, Robin James, Glenn Trewitt, Jason Dangel, Guillermo Lovato, Paul Stoffregen, Scott Roberts, Bertrik Sikken, Mark Tillotson, Ken Butcher, Roger Clark, Love Nystrom

2.3.7 installed

Access 1-wire temperature sensors, memory and other chips.

**DallasTemperature** par Miles Burton <miles@mnetcs.com>, Tim Newsome <nuisance@casualhacker.net>, Guil Barros <gfbarros@bappos.com>, Rob Tillaart <rob.tillaart@gmail.com>

Arduino Library for Dallas Temperature ICs Supports DS18B20, DS18520, DS1822, DS1820

A venir

A venir



#### 4.5.3 Exemples de code

- [Arduino UNO](#)
- [ESP \(Arduino\)](#)
- [Rpi Pico \(µPython\)](#)
- **Ressource** : document [pdf](#)



- **Exemple** de l'IDE Arduino pour tester le capteur

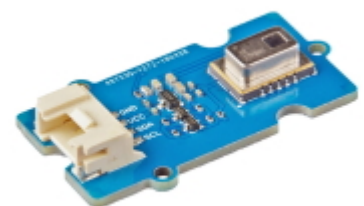
Dans l'IDE Arduino, sélectionner : Fichier → Exemples → DallasTemperature → **Simple**

A venir

A venir

## 5 Thermique

### 5.1 AMG8833



### 5.1.1 Présentation

- **Source** : [wiki](#) seed studio

Le réseau de capteurs de température infrarouge Grove (AMG8833) est un capteur de haute précision basé sur la technologie MEMS avancée. Il peut prendre en charge la détection de température d'une zone bidimensionnelle : 8 × 8 (64 pixels) et une distance de détection maximale de 7 mètres.

- **Distributeur** : [Gotronic](#)
- **Caractéristiques**
  - Alimentation: 3,3 ou 5 Vcc
  - Résolution: 8 x 8 pixels
  - Plage de mesure: 0 à 80 °C
  - Précision: ± 2,5 °C
  - Portée: ± 7 m
  - Angle de vision: 60°
  - Interface I2C : **SLA = 0x68** par défaut (0x69 via un pont à souder)



- **Documentation**
  - PDF du datasheet **AMG8833** à télécharger [ici](#)

### 5.1.2 Bibliothèques

- [Arduino UNO](#)
- [ESP \(Arduino\)](#)
- [Raspberrypi Pi \(µPython\)](#)
- **Bibliothèque** à télécharger et à installer dans l'IDE : [Seeed\\_AMG8833](#)



Pour utiliser la bibliothèque `Seeed_AMG8833` avec un ESP, mettre la déclaration des types de données en commentaire.

A venir  
A venir



### 5.1.3 Exemples de code

- [Arduino UNO](#)
- [ESP \(Arduino\)](#)
- [Raspberry Pi \(µPython\)](#)

- **Exemple** de l'IDE Arduino pour tester le capteur

Dans l'IDE Arduino, sélectionner : Fichier → Exemples → Grove IR Matrix Temperature sensor AMG8833 → **basic\_demo**

- **Résultat** dans la console



```
COM4

Temperature for 8X8 matrix are::
22.00  21.75  21.25  21.25  21.25  22.00  20.75  22.25
22.00  21.75  21.25  21.00  20.75  21.50  21.75  21.75
21.75  21.75  21.25  21.25  21.50  21.25  21.25  21.75
21.75  21.75  21.25  21.25  21.00  21.25  20.75  22.00
21.25  20.50  21.00  20.50  21.00  21.50  21.50  21.25
21.00  21.25  21.00  20.75  20.50  20.75  21.50  22.00
20.50  20.50  21.00  20.75  21.25  20.75  21.25  21.25
20.50  20.25  20.75  21.00  21.50  21.25  21.50  21.50
```

- **Mise en oeuvre** du capteur avec un afficheur OLED

- Ressource : [wiki](#) seed studio
- Matériels
  - Carte à microcontrôleur : [ESP8266 Feather Huzzah](#)



[Télécharger](#) le projet PlatformIO pour VSCode.

A venir

A venir

Last update: 2024/05/28 06:51  
materiels:capteurs:temperature:temperature <https://webge.fr/dokuwiki/doku.php?id=materiels:capteurs:temperature:temperature&rev=1716871876>

---

From:  
<https://webge.fr/dokuwiki/> - **WEBGE Wikis**

Permanent link:  
<https://webge.fr/dokuwiki/doku.php?id=materiels:capteurs:temperature:temperature&rev=1716871876>

Last update: **2024/05/28 06:51**

