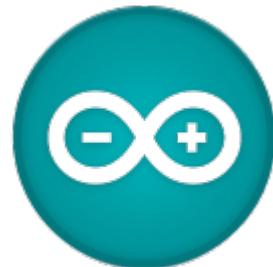




Bibliothèques - Sparkfun TMP102, TMP117

Mise à jour le [18/8/2023]



Ressources

- **Description** en français sur Github [lien](#)
- **Guide** sur le site Sparkfun [lien](#).

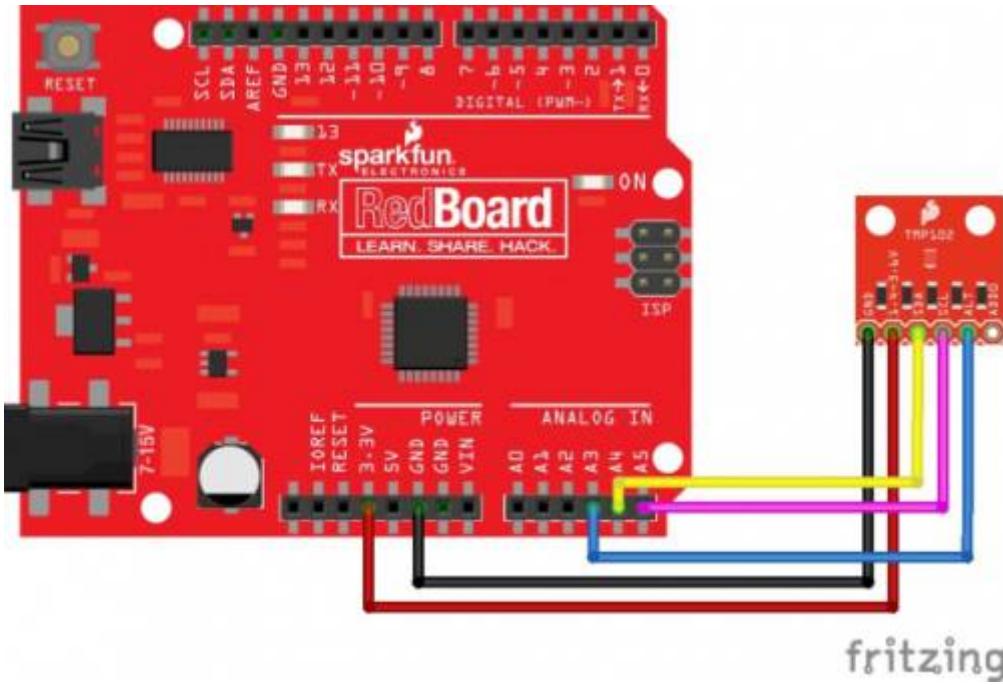
Description

Le **TMP102** de Texas Instruments est un **capteur de température numérique**. Il transmet la température mesurée par un bus **I2C**. Sa **Résolution** est égale à **0,0625°C** et sa **Précision** **0,5°C**. Le module intègre des résistances de tirage de 4,7 kΩ et fonctionne entre 1,4 V et 3,6 V.

Connexions

- VCC → 3.3V
- GND → GND
- SDA → SDA/A4
- SCL → SCL/A5
- ALT → A3

Exemple avec une ancien modèle de carte



fritzing

- ADD0 est utilisée pour changer l'adresse du TMP102.
 - VCC → 0x49
 - SDA → 0x4A
 - SCL → 0x4B

La classe TMP102

Constructeur

	Syntaxe	Description
📦	TM102(byte adresse); // Exemple TM102 sensorTemp(0x48);	Initialise un objet TM102 avec l'adresse passée en paramètre

Méthodes

	Syntaxe	Description
Initialisation		
📦	void begin(void); // Exemple sensorTemp.begin();	Connecte le capteur au bus I2C.
Lecture de la température		
📦	float readTempC(void); // Exemple float temperature = sensorTemp.readTempC();	Retourne la température en °C.
📦	float readTempF(void); // Exemple float temperature = sensorTemp.readTempF();	Retourne la température en °F.
📦	float readLowTempC(void); // Exemple float temperature = sensorTemp.readLowTempC();	Retourne le contenu du registre T_LOW en °C.

	Syntaxe	Description
◆	float readHighTempC(void);	Retourne le contenu du registre T_HIGH en °C. // Exemple float temperature = sensorTemp.readHighTempC();
◆	float readLowTempF(void);	Retourne le contenu du registre T_LOW en °F. // Exemple float temperature = sensorTemp.readLowTempF();
◆	float readHighTempF(void);	Retourne le contenu du registre T_HIGH en °F. // Exemple float temperature = sensorTemp.readHighTempF();

Consommation, seuils de température et alerte

◆	void sleep(void);	Passe le capteur en mode faible consommation (<0,5µA). // Exemple sensorTemp.sleep();
◆	void wakeup(void);	Réveille le capteur et le ramène en mode consommation normale (~10µA). // Exemple sensorTemp.wakeup();
◆	bool alert(void);	Retourne le contenu du registre d'alerte. La valeur est identique à celle la broche ALT. // Exemple bool state = sensorTemp.alert();
◆	void setLowTempC(float temperature);	Ecrit le seuil d'alerte bas (en °C) dans le registre T_LOW. // Exemple sensorTemp.setLowTempC(20.0);
◆	void setHighTempC(float temperature);	Ecrit le seuil d'alerte haut (en °C) dans le registre T_High. // Exemple sensorTemp.setHighTempC(22.0);
◆	void setLowTempF(float temperature);	Ecrit le seuil d'alerte bas (en °F) dans le registre T_LOW. // Exemple sensorTemp.setLowTempF(84.0);
◆	void setHighTempF(float temperature);	Ecrit le seuil d'alerte haut (en °F) dans le registre T_High. // Exemple sensorTemp.setHighTempC(85.0);

Utilitaires

◆	void setConversionRate(byte rate);	Règle le taux de conversion (0-3). 0 → 0,25Hz 1 → 1Hz 2 → 4Hz (par défaut) 3 → 8Hz // Exemple sensorTemp.setConversionRate(3);
◆	void setExtendedMode(bool mode);	Active ou désactive le mode étendu. false → désactive (-55°C à +128°C) true → active (-55°C à +150°C) // Exemple sensorTemp.setExtendedMode(false);
◆	void setAlertPolarity(bool polarity);	Règle la polarité de l'alerte. false → active à l'état BAS true → active à l'état HAUT // Exemple <code cpp> sensorTemp.setAlertPolarity(false);

	Syntaxe	Description
hexagon icon	void setFault(byte faultSetting);	Règle le nombre d'erreurs consécutives. 0 → 1 erreur 1 → 2 erreurs 2 → 4 erreurs 3 → 6 erreurs // Exemple <code>sensorTemp.setFault(1);</code>
hexagon icon	void setAlertMode(bool mode);	Règle le type d'alerte. false → Mode comparateur: Actif si temp > T_HIGH ou temp < T_LOW true → Mode thermostat: actif quand temp > T_HIGH jusqu'à ce qu'une opération de lecture se produise // Exemple <code cpp> <code>sensorTemp.setAlertMode(false);</code>

Exemple à télécharger

[temp1.cpp](#)

```
// Réglage de l'intelliSense (pour la classe TMP102)
// "C_Cpp.intelliSenseEngine": "Default" ou
// "C_Cpp.intelliSenseEngine": "Tag Parser" ne doit pas se trouver dans
setting.json
// Fermer puis rouvrir éventuellement VSCode après la modification
#include <Wire.h>
#include <SparkFunTMP102.h>

TMP102 cptTemp(0x48); // Construction d'un objet TMP102 avec l'adresse
I2C 0x48

void setup()
{
    Serial.begin(115200); // Débit binaire par défaut du moniteur série
    cptTemp.begin(); // Connexion du capteur au bus I2C
}

void loop() {
    float temperature;
    temperature = cptTemp.readTempC(); // Lecture de la température
    Serial.print("Temperature: ");
    Serial.print(temperature);
    Serial.println();
    delay(1000);
}
```

From:

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

Permanent link:

<https://webge.fr/dokuwiki/doku.php?id=materiels:capteurs:temperature:libtmp102&rev=1692370096>

Last update: **2023/08/18 16:48**

