

Rapport chaine de mesure avec capteur

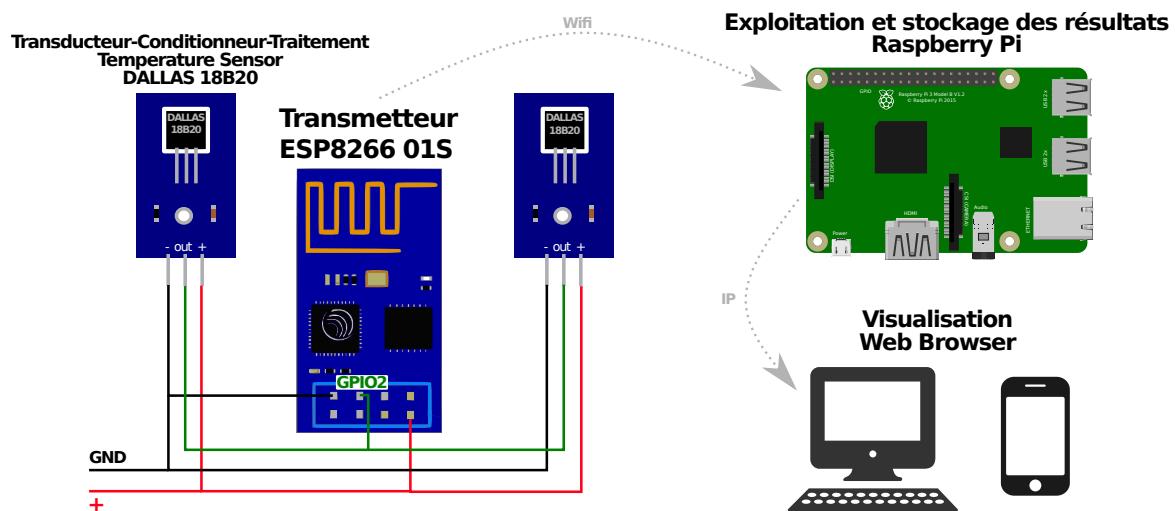
Ulysse COUTAUD p2112506

June 6, 2022

Contents

1	Vue d'ensembe de la chaine de mesure	1
2	Le capteur Dallas 18B20	1
2.1	Caractérisques du capteur 18B20:	2
2.2	Fonctionnalités supplémentaires:	2
3	Mise en oeuvre de la chainde mesure avec capteurs 18B20	3
3.1	Acquisition de la mesure via un micro-controleur ESP8266-01S	3
3.2	Transmission en UDP-IP	4
3.3	Exploitation	4
3.3.1	Stockage des données: UDP Data Logger	4
3.3.2	Visualisation des données: page web sur serveur Apache	5
4	Code source	5
5	Système en production	5

1 Vue d'ensembe de la chaine de mesure



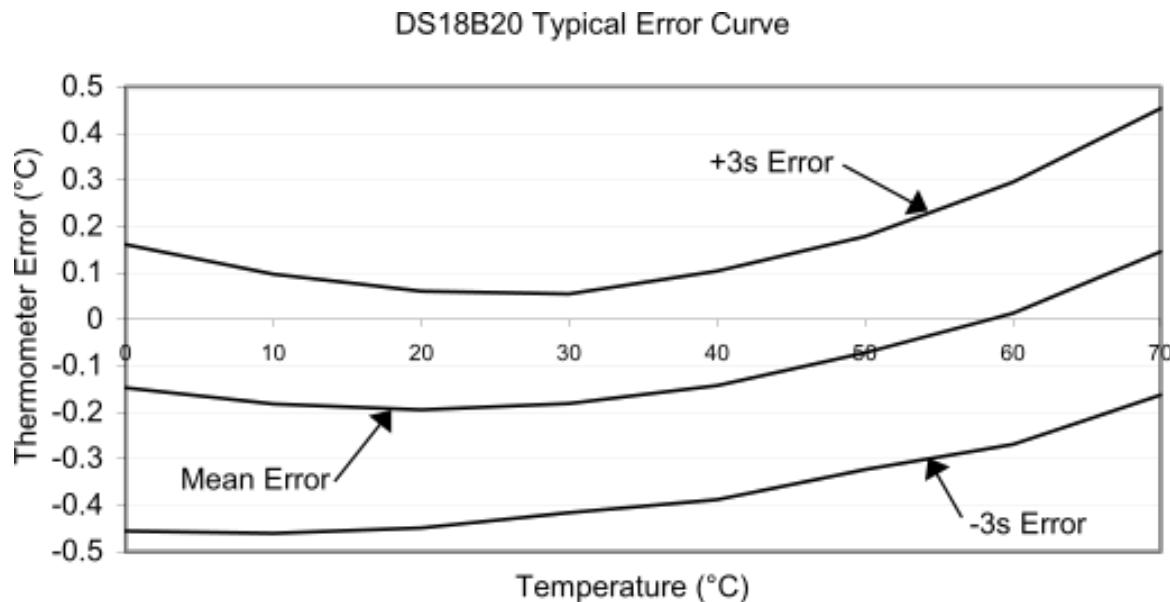
2 Le capteur Dallas 18B20

- J'ai choisi d'utiliser des capteurs 18B20 de *Dallas Semiconductor*.

- Le capteur Dallas 18B20 incorpore les fonctionnalités de transducteur, conditionneur et de traitement de la mesure.

2.1 Caractéristiques du capteur 18B20:

- Plage de mesure: $-55^{\circ}\text{C}/+125^{\circ}\text{C}$.
- Précision: $+/- 0.5^{\circ}\text{C}$ sur la plage $-10^{\circ}\text{C}/+85^{\circ}\text{C}$.
- Résolution: configurable jusqu'à 0.0625°C .
- La figure ci-dessous donne l'erreur typique du capteur décrite par le fabricant:



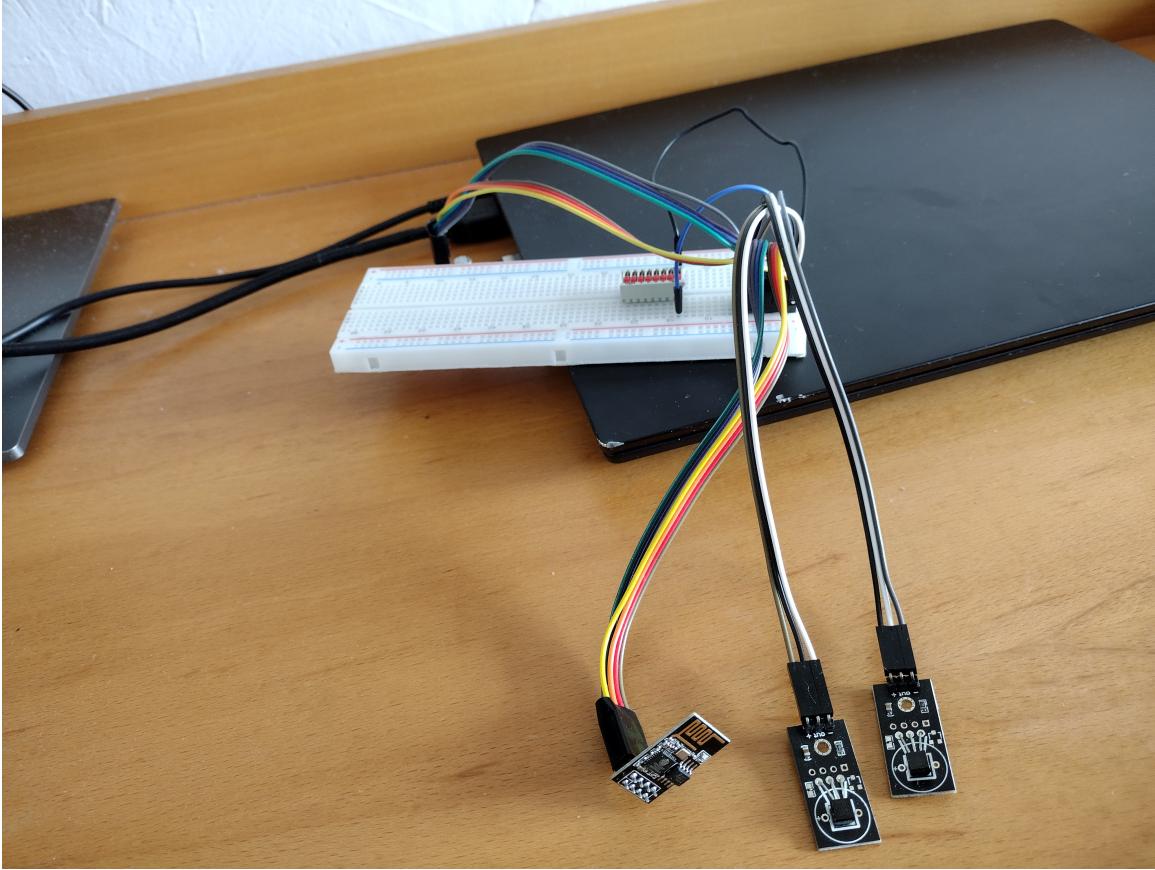
- Le capteur 18B20 utilise une mémoire interne et permet une lecture digitale de la température mesurée, sans nécessité ni d'amplifier le signal, ni de convertir la mesure.
- La lecture de la mesure se fait via un protocole de type bus (1-Wire).

2.2 Fonctionnalités supplémentaires:

- Le capteur 18B20 offre également des fonctionnalités supplémentaires telles que :
 - Un numéro d'identification unique, qui donne la possibilité d'utiliser plusieurs capteurs sur une seule pin GPIO du micro-processeur pilotant la mesure.
 - Un mode de fonctionnement *parasite* pour alimenter le capteur via la pin de données.
 - Un mode alarme pour détecter la sortie des bornes de températures programmées.
 - Une fonction de conversion température-digital de résolution paramétrable à 9, 10, 11 ou 12 bits. NB: le temps

3 Mise en oeuvre de la chaine de mesure avec capteurs 18B20

- La photo ci-dessous présente le banc de mesure.



3.1 Acquisition de la mesure via un micro-controleur ESP8266-01S

- Deux capteurs 18B20 sont connectés sur le port GPIO2 du microcontrôleur ESP8266.
- Le mode alimentation *parasite* n'est pas utilisé. C'est à dire que les bornes + et - du capteur sont alimentés à 3.3v.
- L'utilisation de 2 capteurs me permettra donc de comparer les résolutions à 9bits et 12 bits.
- Les capteurs sont initialisés par la fonction si dessous:

```
int initDallas18B20Sensor(DallasTemperature* sensors, DeviceAddress* sensorsAddresses) {  
    int nbCapteurs;  
    sensors->begin();  
    nbCapteurs = sensors->getDeviceCount();  
  
    Serial.print("Capteurs branchés: ");  
    Serial.println(nbCapteurs, DEC);  
    for (int i=0; i<nbCapteurs; i++){  
        if(!sensors->getAddress(sensorsAddresses[i], i)) {  
            Serial.println("Echec à la détection du capteur.");  
            return -1;  
    }  
}
```

```

        }
        Serial.print("Adresse OneWire: [");
        printAddress(sensorsAddresses[i]);
        Serial.println("]");
    }
    sensors->setResolution(sensorsAddresses[0], 12);
    sensors->setResolution(sensorsAddresses[1], 9);
    return 0;
}

```

- L'acquisition de la mesure est ensuite effectuées toutes les 30s par la fonction ci-dessous:

```

void readFromSensors(DallasTemperature* sensors){
    sensors->requestTemperatures();
    while (!sensors->isConversionComplete()){
        //Serial.println("Acquisition en cours.");
    }
    while (!sensors->isConversionComplete());
    return;
}

```

3.2 Transmission en UDP-IP

- Le micro-controlleur ESP8266 se connecte au réseau WIFI pré-provisionné dans le code source.
- Les mesures des 2 capteurs sont concaténées et transmise en Wifi par une communication UDP-IP vers un serveur de données en écoute.

```

getDataFromSensor(&my18B20Sensors, mySensorsAddresses, 0, buffer);
Serial.println(buffer);
buffer += strlen(buffer);
getDataFromSensor(&my18B20Sensors, mySensorsAddresses, 1, buffer);
Serial.println(buffer);
buffer = packet;
sendDataUdp(buffer);

```

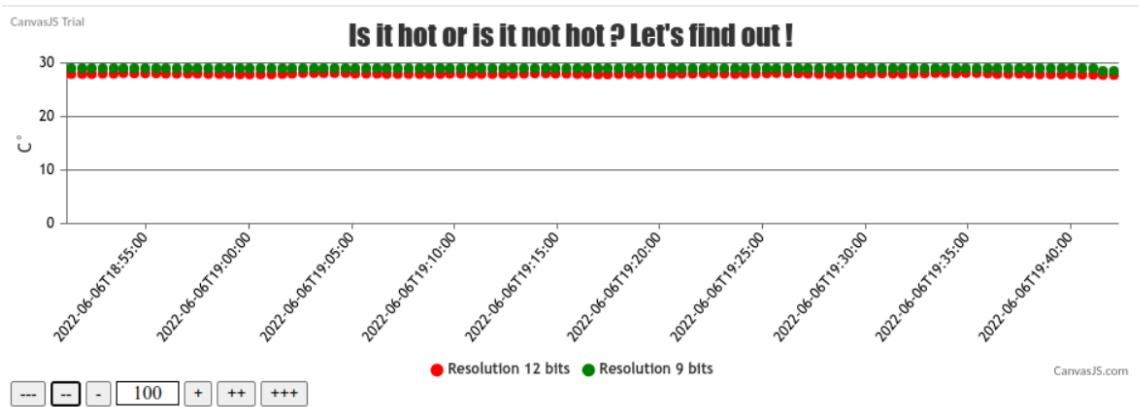
3.3 Exploitation

3.3.1 Stockage des données: UDP Data Logger

- Le serveur de données est développé en C et pour une base Linux.
- Il consiste en un serveur UDP qui effectue:
 - une boucle d'écoute sur un port,
 - ajoute un timestamp au paquet reçu,
 - enregistre les données de manière cumulative dans un fichier au format CSV.

3.3.2 Visualisation des données: page web sur serveur Apache

- L'exploitation des données consiste en une visualisation sur une page web.
- Un serveur Apache avec un module Javascript est installé sur la même machine que le serveur de données.
- Le serveur web est paramétré pour répondre sur le port 32770.
- Une page Web utilisant un script de lecture dynamique du fichier de mesures permet une visualisation en direct de la mesure, sans besoin de rafraîchir la page et en réduisant la bande passante.
- Une fonction de tracé graphique de Canvas JS permet la visualisation graphique des mesures stockées dans le fichier CSV.
- Les boutons en bas à droite du graphe permettent de paramétriser le nombre de données à afficher et ainsi observer des mesures plus anciennes.
- La capture d'écran ci-dessous montre la page web de visualisation:



4 Code source

Le code source est accessible sur github <https://github.com/coutaudu/ChaineDeMesureDallas18B20>

5 Système en production

- Une instance de la chaîne de mesure est accessible sur <http://coutaudu.freeboxos.fr:32770/>
- Les mesures en résolution 12 bits offrent une meilleure granularité que les mesures en résolution 9bits.
- Les mesures des 2 capteurs ne sont pas égales mais la différence est inférieure à 1°C ce qui reste cohérent avec la précision annoncée par le fabricant.