Utilisation d'un capteur de Turbidité et d'un capteur de Température Aquatique et envoi des données via LoRa

DeltaLab

Espace Maison Milon
2 Place E.Colongin
84600 Grillon

deltalabprototype.fr

Qu’est-ce que DeltaLab ?

DeltaLab est une association ‘loi 1901’ d’intérêt général, dont l’objectif est la création d’un espace dédié à l’innovation, à la production numérique au prototypage et à l’«expression artistique».

Le principe fondateur de l’associtation est **d’apprendre à faire soi-même, pour passer de l’dée à l’objet.**

Deltalab se spécialise en **Objets Connectés**, et est en train de créer un vaste «écosystème digital» entre Drôme et Vaucluse, pour répondre à des besoins non-couverts, mettre à disposition ressources et équipements pour usage professionnels et instaurer des partenariats avec les autres structures et initiatives numériques existantes.

Deltalab est aussi un **FabLab** (*Fabrication Laboratory* / *Laboratoire de Fabrication*), un tiers-lieu de type makerspace ou se trouve un atelier qui dispose de machines de fabrication comme des Imprimantes 3D ou des découpeuses Laser.

Deltalab se veut ouvert à tous publics : étudiants, professionnels, associations, inventeurs, designers,artistes, ...

Contexte de cette Documentation

Ce projet présente l'utilisation de capteurs de turbidité (à quel point un liquide est trouble) et de température de liquide. Ces capteurs sont utilisés dans beaucoup de machines (lave-linges , lave-vaisselle,...) et dans de nombreux projets autour de l'Eau (monitoring d'une rivière, d'un chateau d'eau,...).

A DeltaLab, un projet d'Aquaponie est en projet, et ces capteurs seront évidemment utilisés dès le début.

Table des matières

I - Introduction

Ce projet a pour but de présenter l'utilisation de capteurs de turbidité et de température aquatique, dans le but de les intégrer à un système plus grand.

Le projet se décompose en 2 parties distinctes : les capteurs et la carte TTGO associée, et le serveur Node-RED.

Les fonctionnalités proposées par ce projet sont les suivantes :

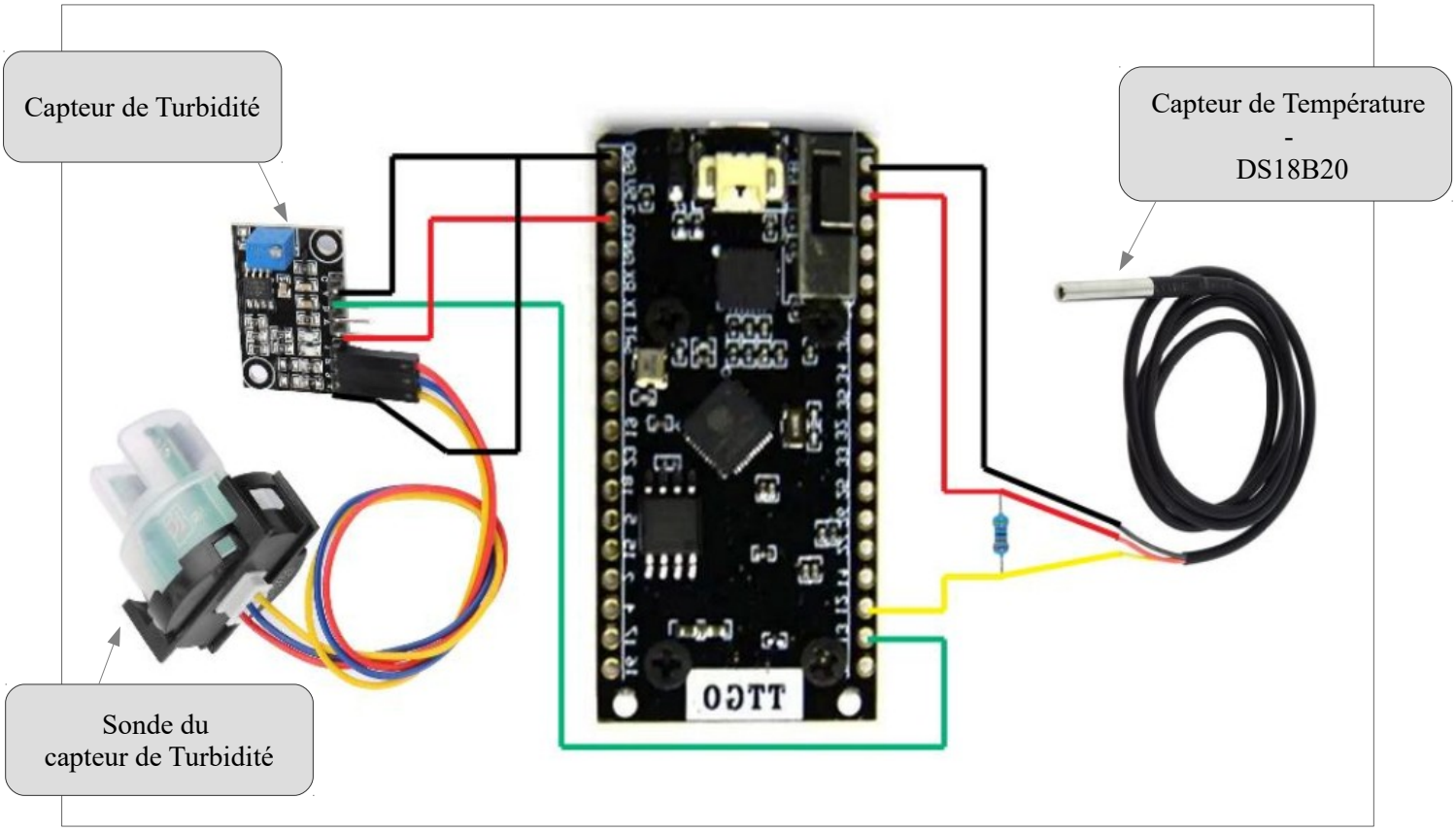
- Récupération des données des capteurs et affichage dans Node-RED, en passant par TTN.
- Possible stockage dans une BdD.

Logiciels :

- Arduino IDE **:** téléchargeable à : https://www.arduino.cc/en/main/software
- Node-RED : à deltalab , entrez l'@ip ***192.168.1.45:1880*** dans un navigateur
- TTGO : ***Fichier > Préférences > URL du gestionnaire de cartes***. Entrez l'URL https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json
  Puis ***Outils> Type de Carte > Gestionnaire de cartes*** , rechercher **ESP32** et installer. Choisissez ensuite **TTGO_LoRa_32** comme Type de Carte.
- Librairies :
  - **AdafruitBME280** : pour le capteur
    - dans Arduino : ***croquis > insérer une bibliothèque > gérer les bibliothèques***
      entrez **BME280** et installez la librairie **Adafruit_BMP280_Library**
  - **MCCI_LoRaWAN_LMIC** : pour LoRaWAN
    - dans Arduino : ***insérer une bibliothèque > gérer les bibliothèques***
      entrez **LMIC** et installez la librairie **MCCI_LoRaWAN_LMIC_Library**
    - **important** : allez dans votre dossier arduino (défaut : ***Documents\Arduino***) et allez dans la configuration de la librairie pour activer la bonne fréquence (***libraries > MCCI_LoRaWAN_LMIC_Library > project_config > lmic_project_config.h***). Dans ce fichier , décommentez (enlevez le ' **//** ') la ligne **#define CFG_eu868 1** (fréquence européenne) et commentez (mettez un ' **//** ') la ligne qui était décommentée (par défaut **#define CFG_us915 1** , fréquence américaine)

II - Présentation du Circuit

Circuit :



Connexions :

Capteur de Turbidité	TTGO	Capteur de Température	Couleur
V	3,3V	-	Rouge
A (mode analogique) ou **D** (mode digital)	12	-	Vert
G	GND	-	Noir
4	GND	-	Noir
-	5V	Rouge	Rouge
-	13	Jaune	Jaune
-	GND	Noir	Noir

Explication :

- Le capteur de Température DS18B20 est entièrement Water-Proof. Il en existe de différentes longueurs, les plus longs mesurant plusieurs mètres. La longueur n'est pas problématique pour la qualité des données grâce au support et au protocole utilisé. Il utilise la librairie OneWire, qui permet d'avoir plusieurs capteurs indentiques sur une seule Pin et de pouvoir récupérer leurs données individuellement. Il renvoie une tempéraure en degré celsius et fonctionnne entre -50 et 80°C.

- Le capteur de Turbidité se compose d'un circuit et d'une sonde. **Le circuit n'est pas Water-Proof, la partie opaque en haut de la sonde non plus. Seule la partie inférieure translucide de la sonde doit donc être immergée.** La sonde est reliée au circuit via 3 cables :

  - ***rouge*** : alimentaion positive , pin #1
  - ***bleu*** : alimentation négative , pin#2,
  - ***jaune*** : données , pin #3

  La pin #4 doit être reliée à une pin Ground de votre TTGO.
  Le capteur peut fonctionner de deux manières différentes :

  - ***Analogique*** : vous devez brancher un cable sur la pin A. Le capteur renverra alors une valeur entre 0 et 1023, de telle sorte que plus le liquide est trouble, plus la valeur est basse.
  - ***Digital*** : vous devez brancher un cable sur la pin D. Le capteur renverra 0 (LOW) par défaut et passera à 1 (HIGH) une fois un seuil de turbidité franchi. Ce seuil est réglable grâce à la vis sur la partie bleue du circuit. La tourner dans le sens horaire augmentera ce seuil et inversement.

- La TTGO récupère les données et les transmet à Node-RED via LoRaWAN.

III - Codes Arduino

Librairies utilisées :

- **MCCI_LoRaWAN_LMIC - version 1.4.6**
- **AdafruitBMP280 - version 2.7.0**

Code :

```
#include <OneWire.h>
#include <DallasTemperature.h>
#include <lmic.h>
#include <hal/hal.h>

#define pin_turbi 12
#define pin_tmp 13

double turb;
double tmp;
OneWire Wir(pin_tmp);
DallasTemperature sensor(&Wir);

typedef union {
  float f[2];
  unsigned char bytes[8];
} donnees;
donnees datas;

static const u1_t PROGMEM APPEUI[8]={0x9A, 0x2F, 0x03, 0xD0, 0x7E, 0xD5, 0xB3,
0x70 };
void os_getArtEui (u1_t* buf) { memcpy_P(buf, APPEUI, 8);}

static const u1_t PROGMEM DEVEUI[8]={0x55, 0x49, 0x30, 0x8C, 0xEF, 0x46, 0x16, 0x00};
void os_getDevEui (u1_t* buf) { memcpy_P(buf, DEVEUI, 8);}

static const u1_t PROGMEM APPKEY[16] = { 0x98, 0x91, 0x66, 0x2A, 0x92, 0xD5, 0xCF,
0x40, 0x5B, 0xBE, 0x90, 0x27, 0xC4, 0xB4, 0x91, 0x0A };
void os_getDevKey (u1_t* buf) {  memcpy_P(buf, APPKEY, 16);}

const unsigned TX_INTERVAL = 20 ;
osjob_t sendjob;
```

```
// Pin Mapping
const lmic_pinmap lmic_pins = {
  .nss = 18,
  .rxtx = LMIC_UNUSED_PIN,
  .rst = 14,
  .dio = {26, 33, 32},
};


// Gestionnaire des évenements de TTN
void onEvent (ev_t ev) {
   switch(ev) {
      case EV_JOINING:
           Serial.println(F("EV_JOINING - antenne cherchée"));
      break;
      case EV_JOINED:
           Serial.println(F("EV_JOINED - antenne connectée"));
           LMIC_setLinkCheckMode(0);
      break;
      case EV_TXCOMPLETE:
           Serial.println(F("EV_TXCOMPLETE - message envoyé"));
           os_setTimedCallback(&sendjob, os_getTime()+sec2osticks(TX_INTERVAL), Send);
      break;
      default:
           Serial.print(F("Unknown event: "));
      break;
    }
}

// Récupération des donneés des capteurs
void getData(){
  turb = 100.0-(analogRead(pin_turbi)*(100.0/1000.0));
  Serial.println(analogRead(pin_turbi));
  Serial.println(turb);
  sensor.requestTemperatures();
  tmp = sensor.getTempCByIndex(0);
  Serial.println(sensor.getTempCByIndex(0));
}

// Envoi des données à TTN
void Send(osjob_t* j){
       getData();
       datas.f[0] = turb;       //turbidité
       datas.f[1] = tmp ;       //température
       LMIC_setTxData2(1, datas.bytes, 8, 0);
       Serial.println(F("Prêt à l'envoi"));
    }
}
```

```
void setup() {
    Serial.begin(115200);
    Serial.println(F("Starting"));
    pinMode(pin_tmp , INPUT_PULLUP);
    sensor.begin();
    pinMode(pin_turbi , INPUT);
    os_init();
    LMIC_reset();
    LMIC_setClockError(MAX_CLOCK_ERROR * 1/100);
    Send(&sendjob);
}

void loop() {
    os_runloop_once();
}
```

Dans arduino, pour choisir la bonne carte , allez dans **Outils > Type de carte**
On utilise ici une TTGO_LORA32_OLED_V1. Vérifiez le Port et la vitesse d'écriture

Pour téléverser le code sur la carte, allez dans **croquis > Téléverser** , ou cliquez sur la flèche.

IV - The Things Network

1. Ajout du device

- Sur **TTN (thethingsnetwork.org) ,** connectez-vous (ou créez un compte) , allez dans la **console**, choisissez **Applications** et **Ajoutez** une nouvelle Appli. Reinseignez un ID (nom) et une descritpion (facultatif).

- Dans cette Application , ajoutez un nouveau **Device** (appareil). Renseignez un nom et une description, et cliquez une fois sur les flèches du champs EUI pour qu'il soit auto-généré.

- Dans l'**overview de votre Device** , vous avez accès aux **EUIs** nécéssaires dans le programme. Appuyer sur les flèches pour le rendre lisible , puis sur les crochets pour le changer en **lsb**. Vous pouvez les copier / coller dans votre code.

- Dans l'**overview de l'application** , vous avez accès à l'**App_Key**. Copiez-Collez la telle qu'elle.

2. Décodage du payload

Votre Application > Payload Format. Collez cette fonction dans la zone '**Decoder**'

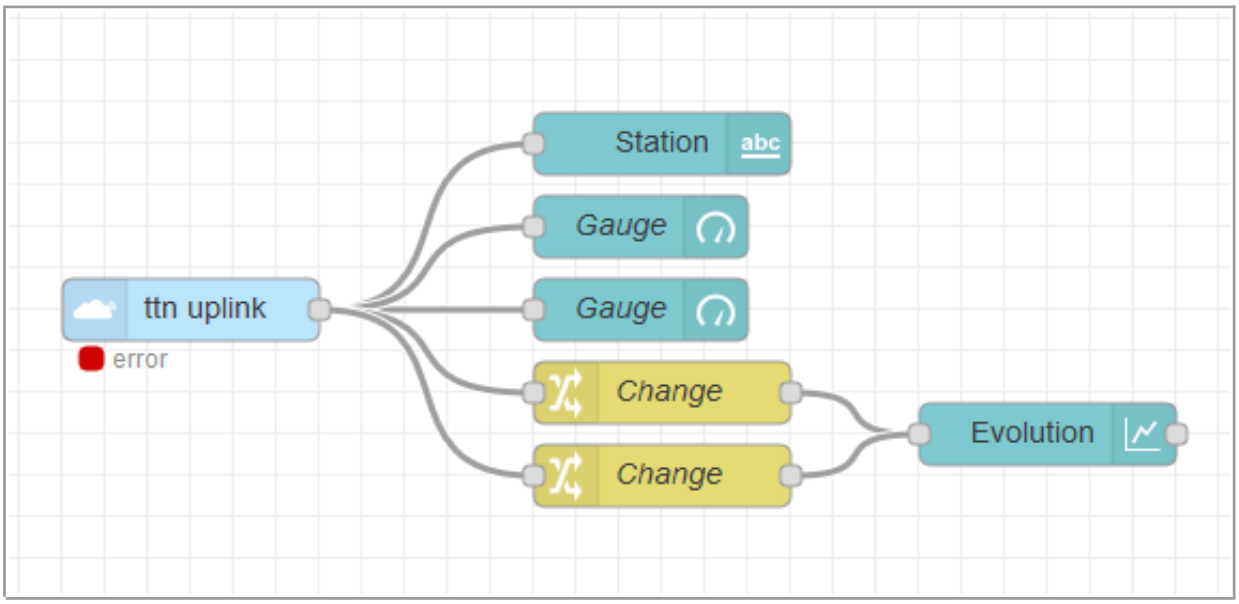
```
function B2F32(bytes) {
        var sign = (bytes & 0x80000000) ? -1 : 1;
        var exponent = ((bytes >> 23) & 0xFF) - 127;
        var significand = (bytes & ~(-1 << 23));
        if (exponent == 128)
                return sign * ((significand) ? Number.NaN : Number.POSITIVE_INFINITY);
        if (exponent == -127) {
                if (significand === 0) return sign * 0.0;
                exponent = -126;
                significand /= (1 << 22);
        } else significand = (significand | (1 << 23)) / (1 << 23);
                return sign * significand * Math.pow(2, exponent);
}

function Decoder(bytes, port) {
        var turb = bytes[3] << 24 | bytes[2] << 16 | bytes[1] << 8 | bytes[0];
        var tmp = bytes[7] << 24 | bytes[6] << 16 | bytes[5] << 8 | bytes[4];

        return{
                turb: B2F32(turb) , tmp: B2F32(tmp)
        };  }
```

IV - Serveur Node-RED

- **Flow :**



- **Explications :**

  - Le capteur envoie ses données et TTN les récupère. Node-RED les lit ensuite grâce à la node TTN-Uplink

  - Les données sont ensuite récupérées par les nodes de dashboard qui permettent de les afficher : des jauges pour la turbidité et la température , ainsi qu'un graphe pour l'évolution de ces deux valeurs.

  - On peut y associer une node 'file' pour enregistrer les données dans un fichier sur le serveur, ou une node de base de données pour y enregistrer les données choisies.

- **Pour le refaire :**

  1. **TTN_Uplink**

     - Application :
       - id : l'id de votre application sur TTN
       - acces key : l'application_key de votre application sur TTN

     - Device (dev_id) : l'id de votre ttgo , si vous voulez filtrer les messages entrants

2. **Gauges (Turbidité, Température)**

   - UI group : créez un nouveau group pour la première et mettez les autre dedans
     - ui_tab : créez une nouvelle
     - width (largeur) : comme vous voulez
     - vous pouvez rendre son nom visible ou non dans le dashboards

   - Type : gauge

   - Label :
     - pour la température : <i class=”fa fa-thermometer-3 fa-2x”></i> (icone thermomètre)
     - pour la turbidité : <i class=”fa fa-cloud fa-2x”></i> (icone nuage noire)

   - Value Format :
     - pour la température : {{msg.payload.tmp.toFixed(2)}}
     - pour la turbidité : {{msg.payload.turbi.toFixed(2)}}

   - Echelle
     - pour la température : -50 à 80, unité : °C ou degrés
     - pour la turbidité : 0 à 100 , unité : % ou pourcents

3. **Changes**

   - première règle :
     - SET
     - champs : msg.topic
     - valeur : tmp ou température / turb ou turbi

   - deuxième règle :
     - MOVE
     - champs : msg.payload.tmp / msg.payload.turbi
     - destination : msg.payload

4. **Chart**

   - Ui_group : le même que les gauges
   - Type : line
   - axe Y : min = -50 , max = 100
   - Réglez le nombre de points max ou la durée représentée comme vous le souhaitez

Dashboard :

Pour ce résultat , changez le thème à **sombre** (**panneau de droite > onglet dashboard - icone de graphe > theme > dark(sombre)**) et réglez la width du groupe et le positionnement des éléments (**panneau de droite > dashboard > votre ui tab > layout**) Ici , le groupe à une largeur de **18** , chaque texte à une largeur de **6** et une hauteur de **1**, chaque gauge à une largeur de **6** et une hauteur de **4** et le graphe à une largeur de **18** et une hauteur de **8**.