



2023
2024

Rapport SAE 4.02.ESE

LOUNES DJERRAH & AMEL AGA

UPEC | Génie électrique et informatique industrielle parcours ESE
Pr. Mr. FRIZIEL

Table des matières

INTRODUCTION	2
1. COMPOSANTS.....	3
1.1. SHT31.....	3
1.2. NUCLEO-L432KC	4
2. SCHÉMATIQUES	5
3. STM32.....	10
4. PASSERELLE NEMEUS	11
5. The Things Network	13
6. TAGOIO.....	0
CONCLUSION	1

INTRODUCTION

Ce document constitue le rapport de projet de la SAE 4.02.ESE que nous avons réalisé durant le 4^{ème} semestre de notre formation BUT génie électrique et informatique industrielle.

Pour ce projet notre groupe été constitué de 2 personnes, Lounes DJERRAH et Amel AGA.

Monsieur FRIZIEL est l'enseignant qui nous accompagnera durant ce projet et nous le remercions pour son aide et ses conseils.

L'objectif de ce projet est de mettre en œuvre un système électronique / systèmes embarqués communicant sans fil en intégrant un traitement numérique des données.

Nous allons devoir pour cela réaliser une « station météo » numérique sur laquelle nous devrons afficher en temps réel la température et l'humidité d'une zone.

Afin de mener ce projet à bien nous a été fournis des composants, des logiciels ainsi que de la documentation constructeur que nous verrons plus en détail dans la suite de ce rapport.

La technologie que nous allons utiliser afin de communiquer nos données sera la technologie LoraWan, nous la verrons également plus en détails par la suite.

Compétences visées

- Assurer le maintien en condition opérationnelle d'un système
- Implanter un système matériel ou logiciel
- Concevoir la partie GEII d'un système
- Vérifier la partie GEII d'un système

Objectifs et problématique professionnelle L'objectif de la SAÉ sera :

- Mise en œuvre modulaire pour la communication d'informations d'un système à un autre
- Choix du protocole sans fil (RF/HF ou optique) choix des modules matériels d'émission et/ou de réception sur le système embarqué
- Configuration des modules de communication
- Mise en place du transfert de données entre l'émetteur et le récepteur.

1. COMPOSANTS

Afin de réaliser le système électronique / système embarqué nécessaire, nous avons besoins spécifiques.

Tout d'abord il nous faut un capteur permettant de mesurer nos 2 degré : la température et l'humidité.

1.1. SHT31

Le composant permettant de mesurer la température ambiante ainsi que l'humidité qui nous est fournis est le *SHT31*.

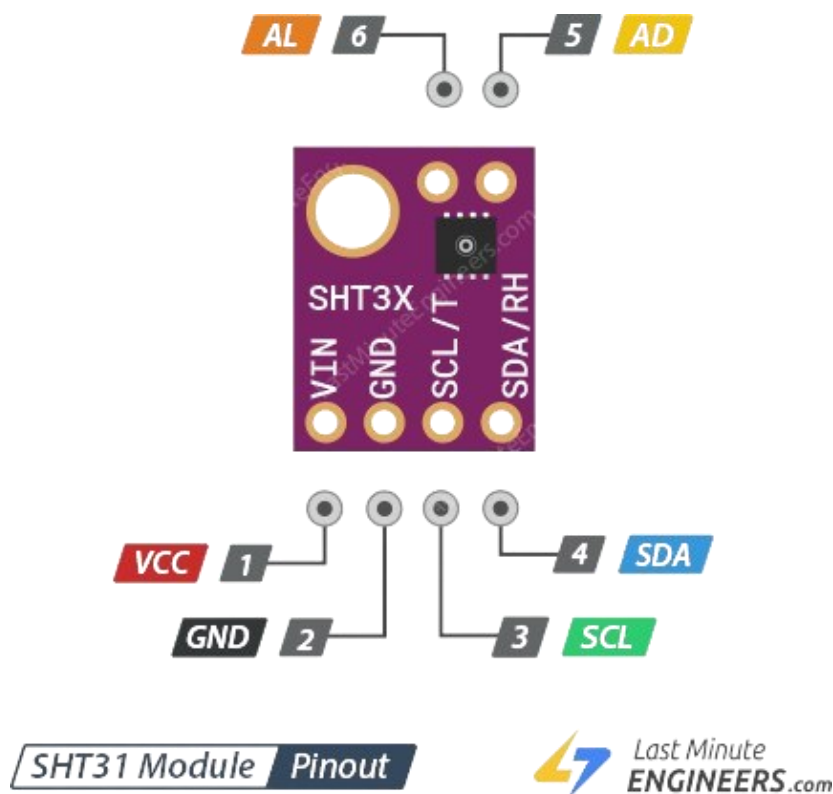


Schéma du SHT31

En observant la datasheet de ce module nous remarquons qu'il possède une précision de plus ou moins 0.1 °C, ce qui est totalement suffisant pour l'utilisation qui nous est demandé.

Pour ce qui est du câblage électronique, pour notre SAE notre module SHT31 est directement implémenté dans une carte électronique connecté avec le microcontrôleur que nous allons utiliser.

1.2. NUCLEO-L432KC

Le microcontrôleur que nous allons utiliser pour ce projet est une carte de développement de la série Nucleo-32 de STMicroelectronics, le *NUCLEO-L432KC*.

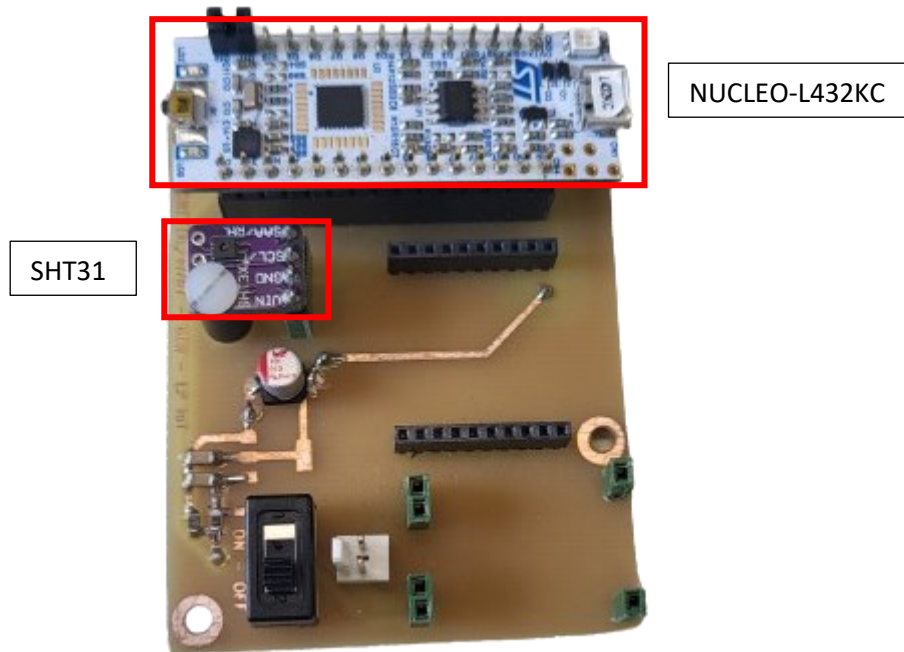


Illustration de notre carte électronique complète

Comme on peut le constater sur l'image précédente, dans notre carte électronique final se trouve le NUCLEO-L432KC couplé à notre module SHT31.

La carte NUCLEO va nous permettre de traité les données de mesures générées par le SHT31 et donc de pouvoir les exploiter.

2. SCHÉMATIQUES

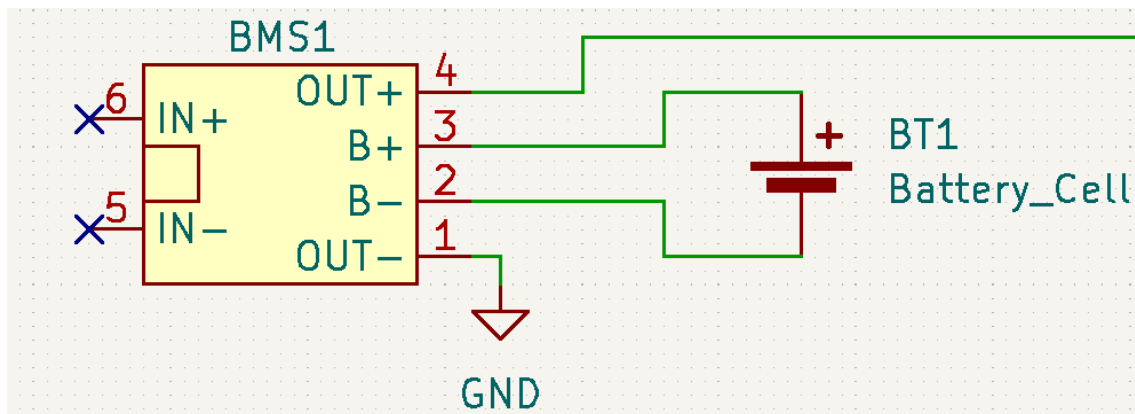
L'on réalise la carte électronique qui est composé :

- Carte NUCLEO L432KC
- Capteur SHT31
- Module XM001
- Module BMS
- Module ADP_3338
- Switch
- Batterie 3.7 V
- Pont diviseur de tension ($R1 = 10\text{kohm}$, $R2 = 4.3\text{ko}$)
- 2 Condensateur ($C1$ et $C2 = 1\text{uf}$)

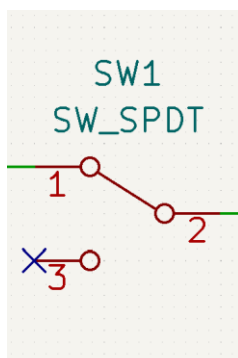
Pour la conception du schéma électrique de la carte électronique nous avons utilisé Kicad.

Lors du montage nous avons créé les symboles pour chacun de nos composants en cherchant sur internet leur taille. On les connecte entre eux selon les spécifications du projet.

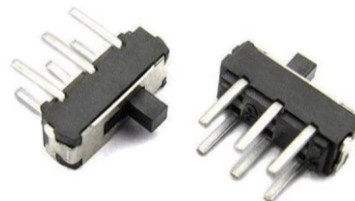
Le BMS est un composant qui permet la gestion de la batterie.



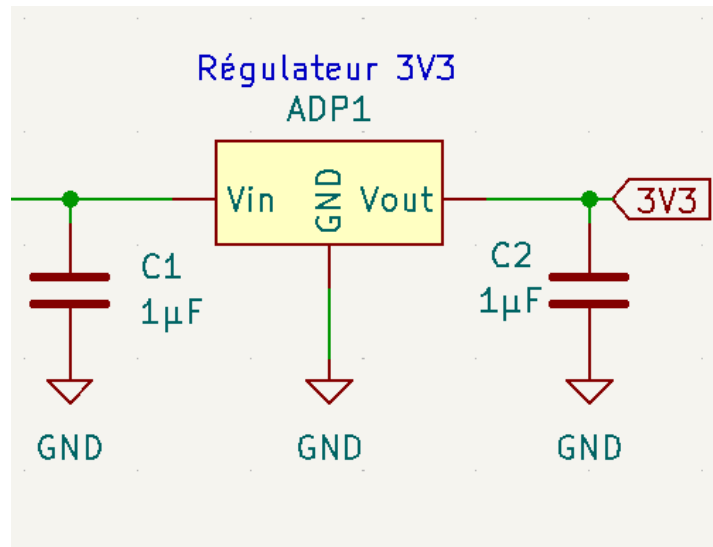
Le switch branché dans le pin out+ du composant permet de déconnecter et de connecter la batterie.



Nous utiliserons un interrupteur de ce type :



Le composant ADP est un régulateur low-drop 3V3 l'on utilise un régulateur car il a une chute de tension de 190mv, ce qui signifie que la tension d'entrée et de sortie peuvent être proche, on met une tension de 3V7 qui va être régulé en 3V3 donc la carte nucléo sera alimenté jusqu'à 3V5 de la batterie. Il est composé de deux condensateur qui serve au découplage.



Le pont diviseur de tension va permettre de récupérer la tension de la batterie. On utilise un pont diviseur de tension car la carte Nucléo ne peut pas recevoir une tension de 4V mais de 3V alors on va diviser la tension.

Pour passer d'une tension maximale de 4V à 3V en utilisant un pont diviseur de tension, nous devons choisir correctement les valeurs des résistances R1 et R2.

Ce pont diviseur servira à fournir à la carte une tension de 3V variable que l'on utilisera pour déterminer la charge restante de la batterie de notre système.

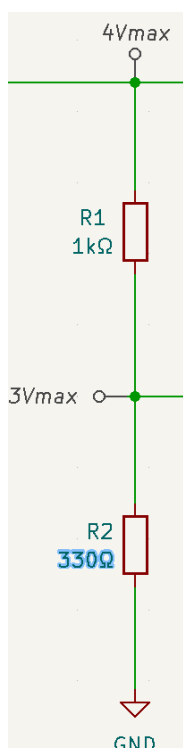


Schéma et calcul du pont diviseur de tension :

$$V_{out} = V_{in} \cdot \frac{R_2}{R_1 + R_2}$$

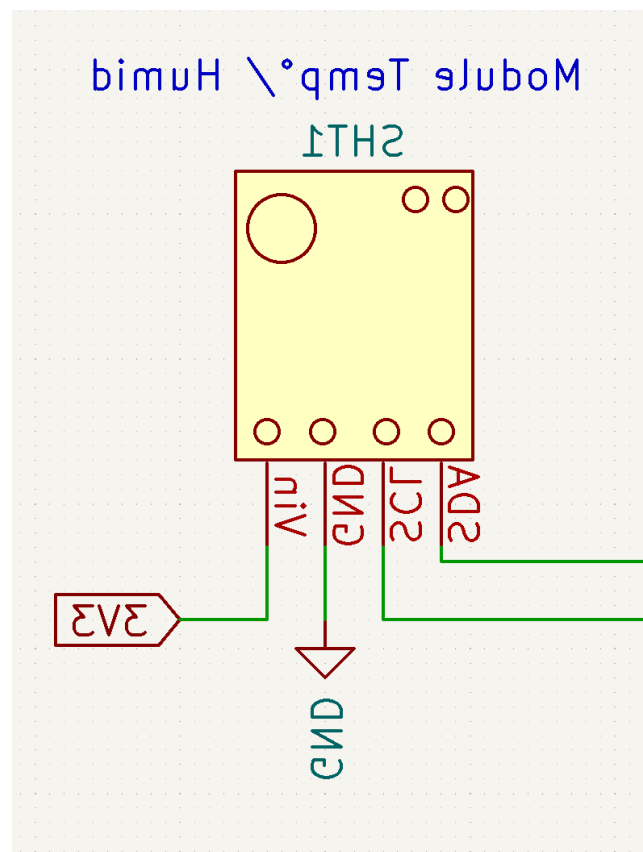
$$3V = 4V \cdot \frac{1000 \Omega}{330 \Omega + 1000 \Omega}$$

$$\frac{1000}{330 + 1000} = \frac{1000}{1330} \approx 0.7519$$

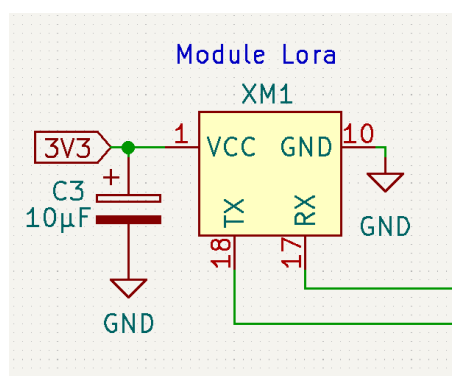
$$V_{out} = 4V \cdot 0.7519 \approx 3.0076V$$

Nous voyons que la sortie serait légèrement supérieure à 3V. Cette approximation est très proche mais pas exacte, car nous devons utiliser des résistances normalisées E24 et nous jugeons que le résultat est suffisant dans notre application.

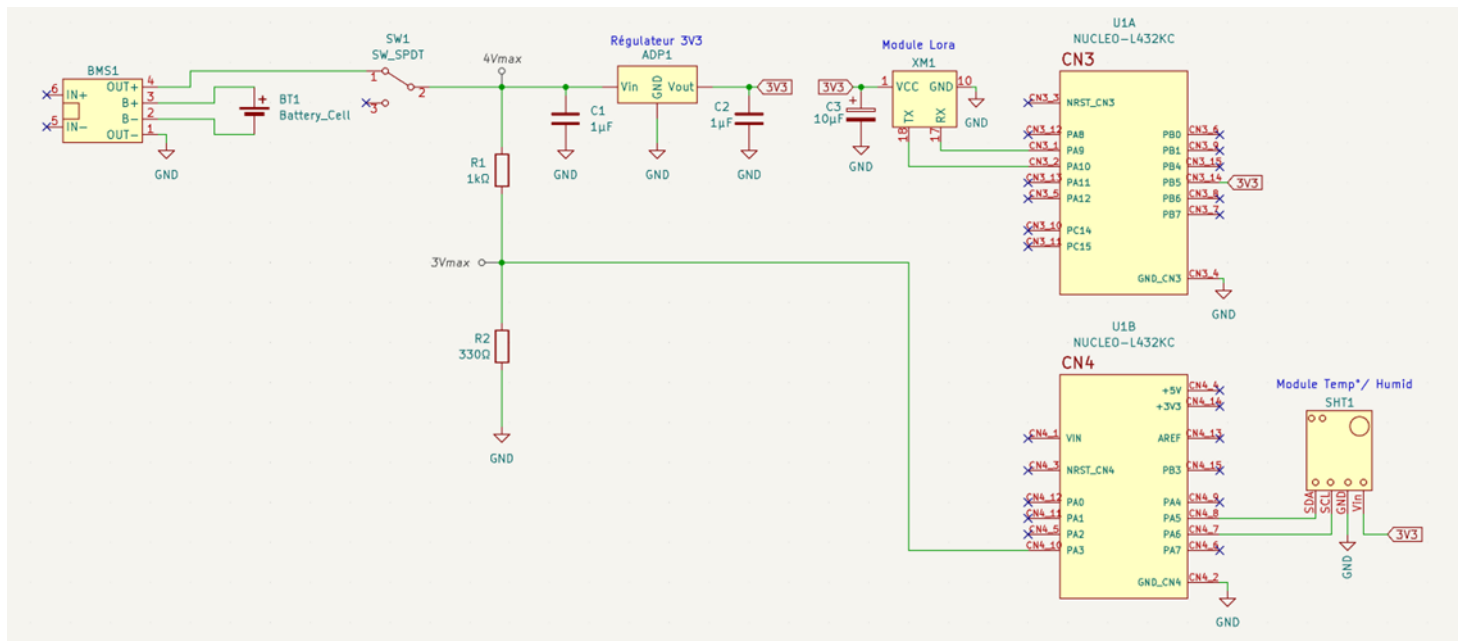
Le composant SHT31 représente le capteur de température et d'humidité. La broche SDA (Serial Data Line) permet aux appareils d'envoyer et de recevoir les informations qu'ils se partagent et la broche SCL (Serial Clock Line) représente l'horloge qui va permettre de synchroniser la transmission des données sur la broche SDA.



Le composant XM001 permet la communication entre Lora et STM32 avec le pin Rx qui va recevoir les données et le pin Tx qui va transmettre les données.

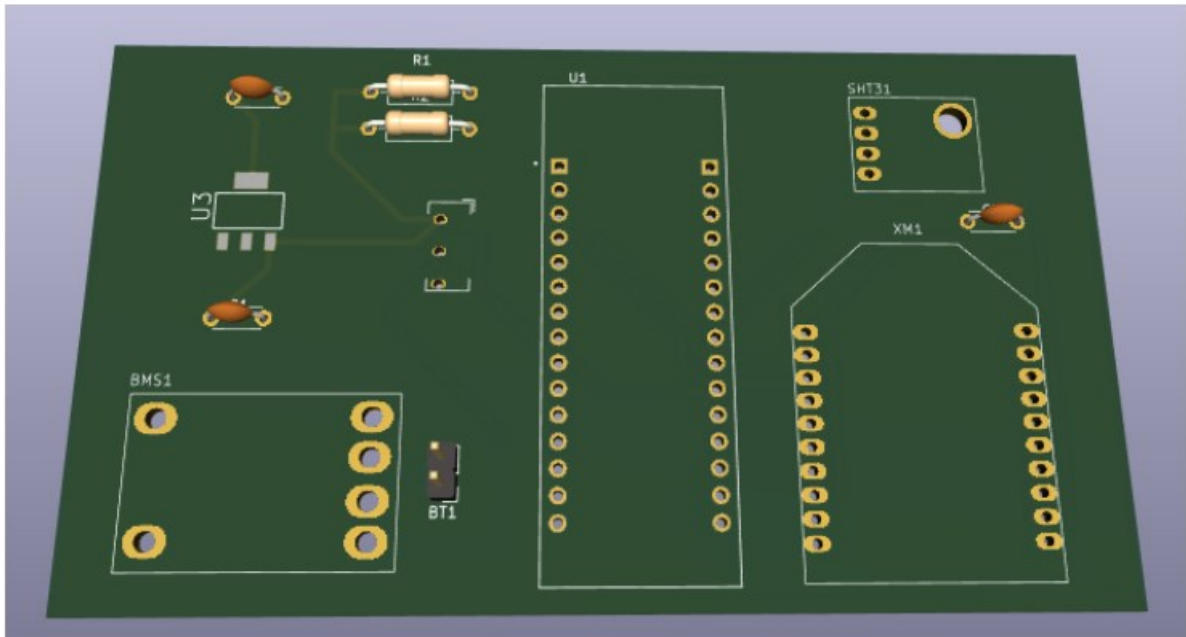


La carte NUCLEO L432KC permet de récupérer les valeurs de température et d'humidité donné par le capteur SHT31 pour les transmettre au module XM001 qui permet d'envoyer les données au serveur.

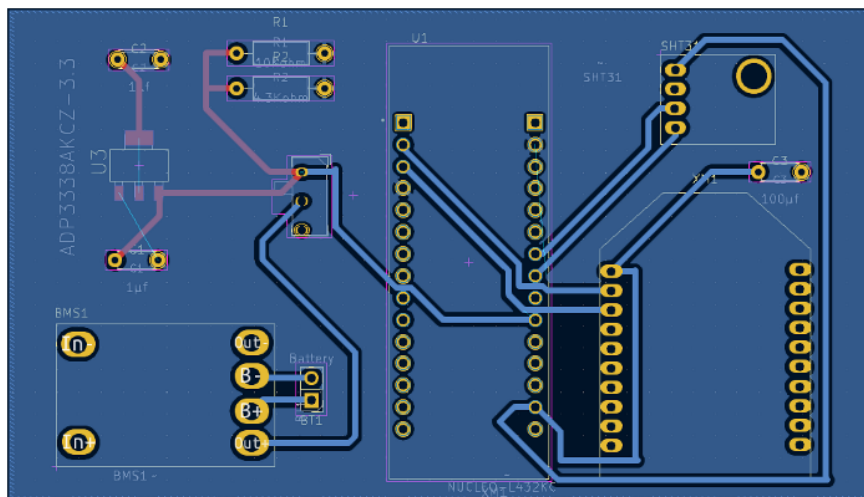


Routage

Après avoir réalisé le schéma électronique sur Kicad, nous sommes passés à l'étape du routage. Le routage consiste à connecter logiquement les composants électroniques en respectant les bonnes dimensions de chaque composant et en optimisant le cheminement des pistes de connexion. Dans notre cas, le routage a été effectué en deux couches : la couche supérieure, représentée par les fils rouges, et la couche inférieure, représentée par les fils bleus.



Le routage a pour objectif de garantir plusieurs aspects essentiels au bon fonctionnement du circuit électronique. En résumé, le routage est une étape cruciale dans la conception électronique, car il transforme le schéma théorique en un circuit logique fonctionnel et permet la création du fichier gerber.



3. STM32

Pour la programmation de cette carte nous allons nous baser sur l'environnement de développement STM32CubeIDE, compatible avec notre carte.

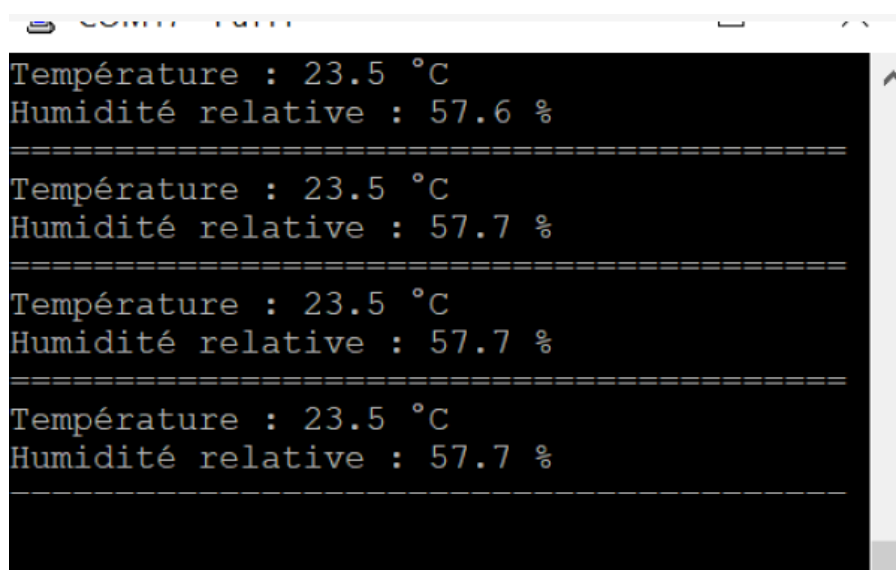
Nous avons donc besoin d'un programme qui puisse récupérer les données du SHT31, les convertir en données numérique (au départ les données de sorties du SHT31 sont des données analogiques) afin qu'elle soit exploitable et enfin les afficher sur un émulateur, en l'occurrence ici nous utiliserons *TeraTerm*.

Voici donc notre programme :

```
while (1)
{
    sht3x_read_temperature_and_humidity(&handle,&temperature,&humidity);
    sprintf(buffer_temperature,"%4.2f\n\r",temperature);
    HAL_UART_Transmit(&huart2,(uint8_t*)buffer_temperature,sizeof(buffer_temperature)/sizeof(buffer_temperature[0])-1,100);
    HAL_Delay(1000);
    sprintf(buffer_humidity,"%4.2f\n\r",humidity);
    HAL_UART_Transmit(&huart2,(uint8_t*)buffer_humidity,sizeof(buffer_humidity)/sizeof(buffer_humidity[0])-1,100);
    HAL_Delay(1000);

    sprintf(buffer,"AT+MAC=SNDBIN,%4.2f,0,2,1\r\n",humidity);
    HAL_UART_Transmit(&huart1, (uint8_t*)buffer, sizeof(buffer)/sizeof(buffer[0])-1,100);
    HAL_UART_Transmit(&huart2, (uint8_t*)buffer, sizeof(buffer)/sizeof(buffer[0])-1,100);
}
```

Grâce à ce programme nous obtenons la sortie sur TeraTerm suivante :

A screenshot of a TeraTerm terminal window. The terminal has a black background with white text. It displays four lines of sensor data, each preceded by a separator line of equals signs. The data shows a constant temperature of 23.5 °C and a relative humidity that fluctuates slightly between 57.6% and 57.7%. The window title bar at the top shows 'COM1 - TeraTerm' and standard window controls.

```
=====
Température : 23.5 °C
Humidité relative : 57.6 %
=====
Température : 23.5 °C
Humidité relative : 57.7 %
=====
Température : 23.5 °C
Humidité relative : 57.7 %
=====
Température : 23.5 °C
Humidité relative : 57.7 %
=====
```

Sortie de TeraTerm

4. PASSERELLE NEMEUS

La passerelle NEMEUS est une Gateway (passerelle réseau) avec laquelle nous devons communiquer les données de mesures enregistrées par notre module SHT31.

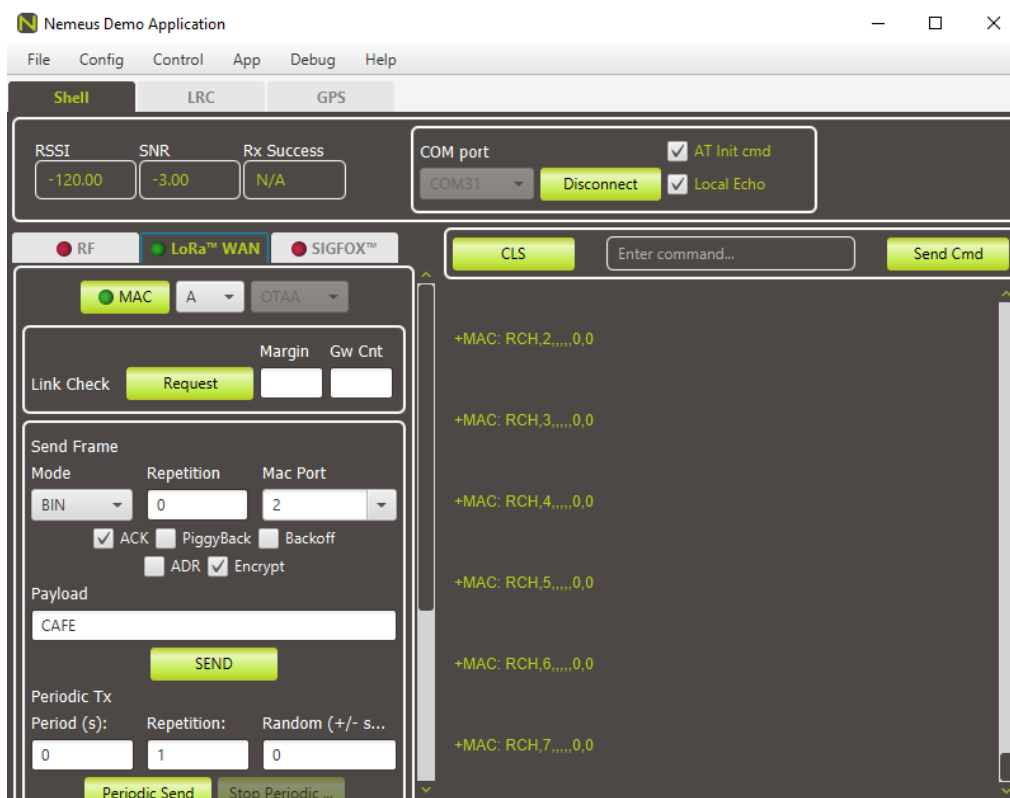
Ensuite, le site internet TTN (The Things Network) sur le compte de communication qui nous a été attribué (donc un adressage spécifique), nous allons pouvoir normalement lire les messages envoyés.

Pour réaliser cela nous avons à notre disposition un logiciel nommé « NEMEUS » qui permet donc de réaliser les configurations nécessaires pour communiquer avec une Gateway grâce à la communication radio LoRa WAN.

LoRaWAN (Long Range Wide Area Network) est un protocole sans fil pour l'Internet des Objets (IoT) offrant une longue portée de plusieurs kilomètres et une faible consommation d'énergie, permettant une durée de vie des batteries de plusieurs années.

Pour notre cas ce protocole est idéal pour la réalisation de la communication de nos données et donc de la création de notre station météo.

Nous paramétrons alors notre configuration dans le logiciel NEMEUS et allons procéder à un premier test de communication. Pour cela nous allons envoyer le message « CAFE » qui est une chaine de caractères en langage ASCII, ce mot est utilisé par convention.



Interface NEMEUS

Comme on peut le constater sur l'image précédente, nous utilisons bien le protocole LoRa WAN, nous sommes bien placés sur le port série où est branché notre module XM001 et nous envoyons bien le message « CAFE » à notre Gateway.

Si tout s'est bien passé nous devons recevoir le message envoyé sur notre Gateway.

5. The Things Network

Nous avons à présent envoyé un message à notre Gateway grâce à notre module XM001 et grâce au logiciel NEMEUS depuis notre PC.

Mais pour nous assurer que la communication a bien été établie nous devons vérifier si le message est correctement reçu dans la Gateway.

Pour cela, nous a été mis à disposition un compte commun pour toute la classe de communication entre la Gateway et un serveur internet sur lequel nous pouvons visualiser les messages envoyés, ce site est The Things Network (TTN).

The Things Network (TTN) est une initiative mondiale qui offre une infrastructure de réseau LoRa WAN ouverte et communautaire, facilitant la création et le déploiement de solutions IoT à faible coût et faible consommation d'énergie.

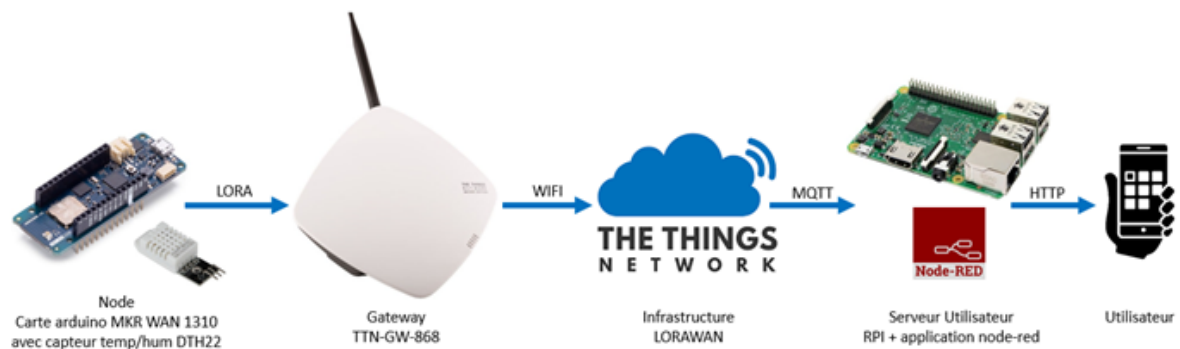


Schéma explicative du fonctionnement de TTN

Nous nous connectons donc au compte commun de notre classe sur TTN et analysons les résultats.

app-liot-2021a

Overview

End devices

Live data

Payload formatters

Integrations

Collaborators

API keys

General settings

14:11:47 device-lpi... Fail to send webhook

End devices (12)

Search

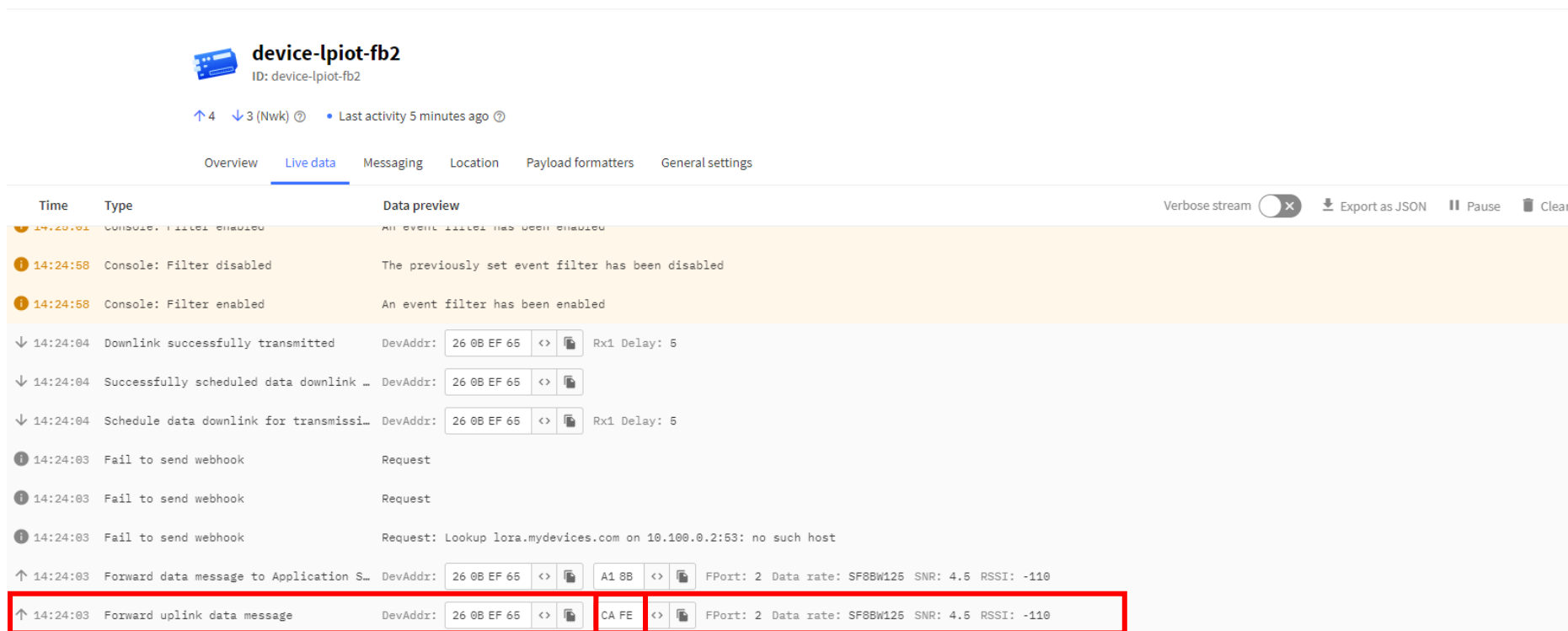
Import end devices

Register end device

ID	Name	DevEUI	JoinEUI	Last activity
eui-70b3d53260008fbc		70 B3 D5 32 60 00 8F BC	70 B3 D5 32 60 00 8F BC	Never
device-lpiot-fb5	device-lpiot-fb5	70 B3 D5 32 60 00 8F B5	70 B3 D5 32 60 00 01 00	Never
device-lpiot-fb7	device-lpiot-fb7	70 B3 D5 32 60 00 8F B7	70 B3 D5 32 60 00 01 00	27 days ago
device-lpiot-fb0	device-lpiot-fb0	70 B3 D5 32 60 00 8F B0	70 B3 D5 32 60 00 01 00	Apr 4, 2023
device-lpiot-fb8	device-lpiot-fb8	70 B3 D5 32 60 00 8F B8	70 B3 D5 32 60 00 01 00	7 days ago
device-lpiot-fb9	device-lpiot-fb9	70 B3 D5 32 60 00 8F B9	70 B3 D5 32 60 00 01 00	Apr 4, 2024
device-lpiot-fb3	device-lpiot-fb3	70 B3 D5 32 60 00 8F B3	70 B3 D5 32 60 00 01 00	7 days ago
device-lpiot-fb4	device-lpiot-fb4	70 B3 D5 32 60 00 8F B4	70 B3 D5 32 60 00 01 00	Apr 4, 2024
device-lpiot-fb1	device-lpiot-fb1	70 B3 D5 32 60 00 8F B1	70 B3 D5 32 60 00 01 00	Apr 5, 2024
device-lpiot-fba	device-lpiot-fba	70 B3 D5 32 60 00 8F BA	70 B3 D5 32 60 00 01 00	Never
device-lpiot-fb2	device-lpiot-fb2	70 B3 D5 32 60 00 8F B2	70 B3 D5 32 60 00 01 00	6 sec. ago
device-lpiot-fb6	device-lpiot-fb6	70 B3 D5 32 60 00 8F B6	70 B3 D5 32 60 00 01 00	Apr 30, 2024

Hide sidebar

Comme on peut le constater sur l'image précédente, nous savons que nous sommes le device-iplot-fb2 et nous voyons bien que notre message a bien été reçu par la Gateway il y a 6 secondes.



device-iplot-fb2
ID: device-iplot-fb2

↑ 4 ↓ 3 (Nwk) ⓘ • Last activity 5 minutes ago ⓘ

Overview **Live data** Messaging Location Payload formatters General settings

Time	Type	Data preview
14:24:01	Console: Filter enabled	An event filter has been enabled
14:24:58	Console: Filter disabled	The previously set event filter has been disabled
14:24:58	Console: Filter enabled	An event filter has been enabled
14:24:04	Downlink successfully transmitted	DevAddr: 26 0B EF 65 <> Rx1 Delay: 5
14:24:04	Successfully scheduled data downlink ...	DevAddr: 26 0B EF 65 <>
14:24:04	Schedule data downlink for transmissi...	DevAddr: 26 0B EF 65 <> Rx1 Delay: 5
14:24:03	Fail to send webhook	Request
14:24:03	Fail to send webhook	Request
14:24:03	Fail to send webhook	Request: Lookup lora.mydevices.com on 10.100.0.2:53: no such host
14:24:03	Forward data message to Application S...	DevAddr: 26 0B EF 65 <> A1 8B <> FPort: 2 Data rate: SF8BW125 SNR: 4.5 RSSI: -110
14:24:03	Forward uplink data message	DevAddr: 26 0B EF 65 <> CA FE <> FPort: 2 Data rate: SF8BW125 SNR: 4.5 RSSI: -110

Si l'on regarde dans les détails de ce message on observe bien que c'est le message en ASCII « CAFE » qui a été reçu par la Gateway.

Suite à cette vérification on peut affirmer que notre communication avec la Gateway est correctement établie.

6. TAGOIO

Nous pouvons maintenant communiquer avec notre Gateway à l'aide de la passerelle NEMEUS, l'objectif est maintenant de pouvoir communiquer nos données de mesures enregistrées par le module SHT31 en direct et de pouvoir les exploiter en les visualisant.

Pour cela nous avons le site internet TagoIO qui nous permet de réaliser cela.

TagoIO est une plateforme IoT qui fournit des outils pour la collecte, la visualisation, l'analyse et l'intégration des données IoT. Elle permet aux utilisateurs de connecter des appareils, de créer des tableaux de bord interactifs, de configurer des alertes et des automatisations, et d'intégrer facilement avec d'autres services et API.



Exemple d'utilisation de TagoIO

Malheureusement nous n'avons pas pu arriver jusqu'à cette étape car nous n'avons pas réussi à exploiter les données que nous recevons sur le site internet.

CONCLUSION

Durant cette SAE nous avons été en capacité de :

- Utiliser le SHT31 en exploitant ses données de mesures grâce à notre carte NULCEO L432KC.
- Établir une connexion / communication entre notre PC et la Gateway NEMEUS grâce au logiciel NEMEUS et notre module XM001 utilisant le protocole LoRa WAN.
- Vérifier le bon fonctionnement de la communication entre notre PC et la passerelle réseau grâce au site internet TTN.

Cela nous a permis :

- Apprendre à exploiter un module de monitoring grâce à une carte NUCLEO.
- Apprendre la fonction d'un protocole de communication radio, ici LoRa WAN, ainsi que ses particularités et ses qualités
- Apprendre à mettre en relation une Gateway et un site comme TTN.
- Apprendre à concevoir entièrement une application de surveillance.

Cette SAE nous a permis d'apprendre énormément de choses malgré que l'on n'ait pas réussi la dernière étape de ce projet qui était d'afficher visuellement nos résultats

