

## CAHIER DES CHARGES

**Surveillance Température-Humidité  
pour réfrigérateur****N° projet Diplôme 2409****Mandataire**

Entreprise/Client:	ETML-ES – Ph. Bovey	Département:	SLO
Demandé par (Prénom, Nom):	Ph. Bovey	Date:	14.08.2025

**1 Objectif - Cahier des charges**

Le but de ce travail de diplôme est de réaliser un système de surveillance pour réfrigérateur, mesurant la température et l'humidité à l'intérieur de celui-ci (surveillance de médicaments ou d'autres denrées périssables).

Le diplômant devra étudier le système d'alimentation : pile (ex : bouton style CR2032) ou superCap, pour permettre à ce que le système de surveillance puisse fonctionner sur une durée d'un à deux mois minimums.

Une communication Wifi, à l'aide d'un **ESP32**<sup>1</sup> ou module équivalent, devra être implémentée pour transmettre une alarme, si un seuil ou un écart de température et/ou d'humidité est détecté, ou si la pile/superCap arrive en fin de vie.

L'utilisateur devra pouvoir se connecter l'ESP32 (celui-ci sera vu comme un web serveur) et transmettre des informations de configuration (écart min-max, seuil limite – éviter la congélation).

Un affiche type ePaper sera implanté sur le système permettant d'afficher des valeurs de température et d'humidité (actuel, min, max, ...). L'affichage sera mis à jour s'il y a des variations de +/- 1° Celsius ou +/- 5% d'humidité.

Le système développé devra être fixé sur la porte du réfrigérateur, pour cela prévoir des magnets (petits aimants) en guise de pieds de fixation.

Pour simuler le réfrigérateur, une glacière portative fera office de Frigo.

**Partie Hardware**

Le diplômant devra concevoir une carte électronique basé sur un microcontrôleur de son choix ; cette carte doit contenir au minimum :

- Un microcontrôleur
  - Alimentation
  - Capteurs : humidité – température
- ATTENTION** : les capteurs seront déportés
- Affichage type ePaper

<sup>1</sup> ESP32 : <https://www.espressif.com/en/products/modules>

- Module RF ESP32 pour transmission de données
- ...

Il devra soit choisir un boîtier du commerce et l'adapter (emplacement affichage – câbles capteurs, accus, etc...) ou le designer lui-même.

### **Partie Firmware**

L'étudiant réalisera un Firmware (partie microcontrôleur) permettant de faire au minimum les tâches suivantes :

- Lecture Capteurs
  - Température
  - Humidité
  - ...
- Communication
  - uC et module Wifi (ESP ou équivalent)
    - envoi et réception de datas
- Communication UART (extérieur)
  - Configuration module ESP
- Affichage
  - Datas (température – humidité – seuil - ...)
- Algorithme pour alarme
- ...

Le diplômant devra aussi configurer le module ESP32 :

- Wifi – mode web serveur

### **Partie Software**

Le diplômant devra garantir la réception et la transmission de donnée sur :

- Module ESP (Wifi – mode web serveur - ...)
- Page Web simple (html)
- Configuration d'un routeur Wifi

## 1.1 Données en lien avec l'objectif – les grandes lignes

- Recherche et implémentation de solutions **Hardware** :
  - Microcontrôleur (préférence fabriquant Microchip ou fabriquant ST Electronics)
  - Alimentation – Etude de cas / recherche de solution :
    - Par pile Bouton
    - Par superCap
    - Pour les deux solutions :
      - Tension et ampérage adéquat
      - Durée de vie
      - ...
- Attention :** les tensions subsidiaires au montage (interne) devront être réalisées par des alimentations à découpage DC-DC
- Capteur Température
  - Plage de mesure – Résolution
  - Communication avec le uC
  - Connexion : Le capteur doit pouvoir être déporté
  - ...
- Capteur d'humidité
  - Plage de mesure – Résolution
  - Communication avec le uC
  - Connexion : Le capteur doit pouvoir être déporté
  - ...
- Module de communication ESP32 ou équivalent
  - Chip ou module
  - Paramétrage du module (boot)
  - Communication – Configuration – UART
    - Avec le uC
    - Extérieur – configuration
  - ...
- Affichage
  - Type Epaper – Monochrome
  - Communication avec le uC
  - ...
- Recherche de composants électroniques autour des composants décrits ci-dessus : AOP, résistances, condensateur, self, autres
- Réalisation de schématique(s) complet(s) et d'un PCB - à réaliser sous ALTIUM de préférence
- Recherche (achat) ou réalisation d'un boîtier sous SolidWorks
- Recherche et implémentation de solutions au niveau **Firmware**
  - Lecture de capteurs
    - Température
      - Protocole de communication (SPI, I2C, UART, ...)
      - Conversion en °C
      - ...
    - Humidité
      - Protocole de communication (SPI, I2C, UART, ...)
      - Conversion en °C
      - ...
    - ...
  - Communication entre le uC et l'ESP32 ou équivalent

- Choix protocole de communication (SPI, I2C, UART, ...)
  - Configuration trames
    - Longueur trame – début et fin trame – taux de transfert
    - Datas (alarme – seuils température – humidité - ...)
    - ...
  - Communication entre le uC et l'affichage (ePaper)
    - Choix protocole de communication (SPI, I2C, UART, ...)
    - Affichage Datas : seuils - température – humidité - ...
    - ...
  - Algorithme Alarme
    - Comparaison seuil/écart et valeurs capteurs
      - Température
      - Humidité
      - ...
    - Batterie
    - ...
  - Configuration module ESP32 ou équivalent :
    - Wifi
      - SSID
      - Password
      - Protocole communication
      - ...
    - Serveur Web
      - Datas (alarme – seuils température – humidité - ...)
      - ...
  - ...
- Recherche et implémentation de solutions au niveau **Software**
    - Configuration serveur WEB sur ESP32
      - Wifi
      - Configuration page html simple
      - Réception et émission de datas (alarme – seuils température – humidité - ...)
      - ...
  - Démontrer par différentes simulations et mesures que les parties Hardware, Firmware et Software ont été bien implémentées.

## 2 A l'issue du projet de diplôme, l'étudiant fournira (liste non exhaustive) :

- Fichiers sources de CAO électronique du PCB réalisé (ALTIUM) + configuration logiciel (version utilisées)
- Tout le nécessaire pour fabriquer un exemplaire hardware : fichiers de fabrication (GERBER) / liste de pièces avec références pour commande (BOM) / implantation (prototype) / modifications, etc
- Fichiers sources de programmation microcontrôleur (.c / .h) + configuration logiciel (version utilisées)
- Fichiers sources d'application externe au système électronique + configuration logiciel (version utilisées)
- Tout le nécessaire pour programmer le microcontrôleur (logiciel ou fichiers .hex), sous format numérique => utilisation de la structure de projet fourni par l'ES
- Tout le nécessaire pour programmer des applications (logiciel ou fichiers .hex) sous format numérique => utilisation de la structure de projet fourni par l'ES
- Tout le nécessaire à l'installation de programmes sur PC ou autres environnements utilisés durant le travail de diplômes
- Un rapport de diplôme contenant :
  - Les concepts du design **Hardware** :
    - Études des différents systèmes à implémenter (explications)
    - Choix des composants et dimensionnement de ceux-ci (calculs / simulation / autre)
    - Réalisation schématique / PCB / boitier / montage de la carte
  - Les concepts du design **Firmware**
    - Structogramme / flowchart / Pseudocode
    - Explication des algorithmes mise en place
    - Démonstration par calculs / outils de debug / des résultats obtenus
    - Validation des concepts mis en place
  - Les concepts du design **Software**
    - Structogramme / flowchart / Pseudocode
    - Explication des algorithmes mise en place
    - Démonstration par calculs / outils de debug / des résultats obtenus
    - Validation des concepts mis en place
  - Tests & Validation :
    - Méthodologie de tests
    - Mesure(s)
    - Validation des résultats
    - Correction(s) apportée(s) au design (Hardware / Firmware / Software)
  - Estimation des coûts pour le design développé (un prototype)
  - Etat d'avancement & problème(s) rencontré(s)
  - Conclusion
  - Bibliographie / webographie / autre sources
- Les annexes :
  - Calculs détaillés des concepts
  - Listings complets des parties Firmware & Hardware que vous avez implémenté
  - Schématique + plan d'implémentation complète du PCB
  - Dessin / schématique du boitier
  - Mesures

- Pages utilisée des différents datasheets ou documentations exploités
  - Mode d'emploi du système développé pendant le diplôme
  - Journal de travail
  - PV de séances hebdomadaires
  - Un prototype

### **3 Autres demandes / contraintes / conseils**

- **Planifier** dans le détail les travaux demandés.
- Se référer au planning régulièrement, **vérifier son avancement**, rédiger son **journal de projet** quotidiennement.
- Commencer à **rédiger le rapport de diplôme le plus tôt possible**, et régulièrement tout au long du travail de diplôme.
- Prendre du temps, préparer sa réflexion, rechercher des apports théoriques et des exemples pratiques, **envisager plusieurs possibilités** avant de finaliser une solution.
- **Numéroter et dater tous les documents**
- En cas de **problème** (retard, objectif à revoir, difficulté rencontrée, etc.), se référer à l'enseignant et au mandataire au plus vite.
- Toutes les **décisions importantes**, tant au niveau technique qu'organisationnel, doivent être posées **par écrit** dans le PV de séance, le rapport de diplôme et /ou figurer dans le journal de projet, après discussion avec l'enseignant / le mandataire.

### **4 Documents de références**

#### **Projet ES**

- **Prj 2312 – Badge pour place de travail**  
K:\ES\PROJETS\SLO\ 2312\_BadgePourPlaceTravail

#### **Vidéo**

- **Envoi de donnée via ESP32 (wifi)**  
<https://www.youtube.com/watch?v=13Crtvr85IE>
- **ESP32 – Web serveur**  
<https://www.youtube.com/watch?v=z-l-r3PX2IU>

#### **Pages web :**

- **Articles**

Accès Point (AP) et ESP32

<https://randomnerdtutorials.com/esp32-access-point-ap-web-server/>

Réseau Wifi et ESP32 :

[https://www.wikidebrouillard.org/wiki/Configurez\\_le\\_r%C3%A9seau\\_Wifi\\_sur\\_un\\_ESP](https://www.wikidebrouillard.org/wiki/Configurez_le_r%C3%A9seau_Wifi_sur_un_ESP)

Web Serveur – ESP32 :

<http://emery.claude.free.fr/esp32-serveur-web-simple.html>

Epaper – fournisseur :

[https://www.mouser.ch/c/optoelectronics/displays/electronic-paper-displays-epaper/?srsltid=AfmBOoqdtHntuGWUgNRkvcWUS7Bdp4\\_K-tzP2bIPH2\\_8cU8vCf7sjlFV&\\_gl=1\\*tfnc0\\* ga\\*MjA1MTI3MzQ3Ny4xNzlzNjQwOTEz\\* ga 15W4STQT4T\\*MTcyMzY0MDkxMy4xLjAuMTcyMzY0MDkxNC41OS4wLjA.](https://www.mouser.ch/c/optoelectronics/displays/electronic-paper-displays-epaper/?srsltid=AfmBOoqdtHntuGWUgNRkvcWUS7Bdp4_K-tzP2bIPH2_8cU8vCf7sjlFV&_gl=1*tfnc0* ga*MjA1MTI3MzQ3Ny4xNzlzNjQwOTEz* ga 15W4STQT4T*MTcyMzY0MDkxMy4xLjAuMTcyMzY0MDkxNC41OS4wLjA.)

➤ **Documentation sur l'ESP32**

Wifi :

[https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/network/esp\\_wifi.html](https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/network/esp_wifi.html)

➤ **Git**

Espressif : <https://github.com/espressif>

# Annexe B

	No semaine projet	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
	Date																																									
	Tâches																																									
Théorique	Pré-étude																																									
Réelle																																										
Théorique	Schématique																																									
Réelle																																										
Théorique	PCB (routage)																																									
Réelle																																										
	Boitier																																									
Théorique	Montage																																									
Réelle																																										
Théorique	Software																																									
Réelle																																										
Théorique	Tests et mise au point																																									
Réelle																																										
Théorique	Rédaction du rapport																																									
Réelle																																										
Théorique	Affiche travail de diplôme																																									
Réelle																																										
Théorique	Résumé travail de diplôme																																									
Réelle																																										
Théorique	Présentation finale																																									
Réelle																																										
		19.08.2024	20.08.2024	21.08.2024	22.08.2024	23.08.2024	24.08.2024	25.08.2024	26.08.2024	27.08.2024	28.08.2024	29.08.2024	30.08.2024	31.08.2024	01.09.2024	02.09.2024	03.09.2024	04.09.2024	05.09.2024	06.09.2024	07.09.2024	08.09.2024	09.09.2024	10.09.2024	11.09.2024	12.09.2024	13.09.2024	14.09.2024	15.09.2024	16.09.2024	17.09.2024	18.09.2024	19.09.2024	20.09.2024	21.09.2024	22.09.2024	23.09.2024	24.09.2024	25.09.2024	04.10.2024	03.10.2024	02.10.2024

Légende
F = Février
R = Rendu
DL = DeadLine (dernier délai)
P = Présentation finale

# Annexe C

Tâches effectuées	Dates	Temps alloué	Problèmes rencontrés	Solutions aux problèmes	Informations et personnes de contact
Présentation des travaux de diplômes	19 août 2024	30 min			
Préparation des fichiers (Rapport final, planification, journal de travail)	19 août 2024	1h	-	-	
Faire le schéma général et le schéma bloc + description de tous les blocs	19 août 2024	2h			
Comprendre le fonctionnement de l'ESP32	19 août 2024	2h			
Recherche de composant pour la pré-étude + écriture du rapport	19 août 2024	4h			
Recherche des composants + comprendre le fonctionnement du e-paper +écriture du rapport	20 août 2024	10h			
Ecriture suite pré-étude + estimation de consommation et coût du PCB	21 août 2024	4h			
Commencement de la partie schéma + écriture du rapport	21 août 2024	2h			
Avancement sur le schéma + écriture du rapport	22 août 2024	6h			
Réunion avec M.Bovey	22 août 2024	1h	J'avais un problème au niveau du pinout de l'E-paper	M.Bovey m'a conseillé de rajouter une résistance ou nous étions pas sur du pinout pour relié ou non la piste si besoin	M.bovey : philippe.bovey@eduvaud.ch
Ecriture du PV	22 août 2024	1h			
Correction du PV	23 août 2024	10 min			
Terminer le schéma	23 août 2024	6h	Le pinout concernant le e-paper était compliqué à comprendre car les pins utilisent différentes terminaisons entre eux, ce qui m'a fait perdre un peu de temps	J'ai simplement regardé le datasheet du STM32 en écrivant sur une feuille que le " DIN " indiqué correspondait au SDI de la pin du e-paper, etc..	
Revue de schéma avec M.Bovey	23 août 2024	2h	Pas de problème spécialement, mais demande d'ajout d'un quartz	J'ai repris le même quartz qu'utilisé lors de mon précédent projet, c'est un 8Mhz allant jusqu'à du 48Mhz si besoin	M.bovey : philippe.bovey@eduvaud.ch
Rajout d'un quartz et changement de la diode pour l'alimentation sur le schéma + correction des noms de composants	24 août 2024	1h			
Relecture pré-étude et commencement écriture de la partie schématique	25 août 2024	4h			
Revue de schéma avec M.Feliciano	26 août 2024	2h			cyril.feliciano@eduvaud.ch
Vérifier les composants en stock à l'ETML-ES + mettre les footprints sur chaque composants sur schéma	26 août 2024	5h			
Etablir la taille du pcb + placement des composants pour le routage	26 août 2024	4h			
Commande des composants	27 août 2024	1h			
Routage du PCB	27 août 2024	6h			cyril.feliciano@eduvaud.ch
Revue du PCB avec M.Feliciano et M.Bignens	28 août 2024	1h			
Modification du PCB pour mettre les points tests au bottom pour les signaux	28 août 2024	1h30			
Vérification du PCB sur Eurocircuits et aider M.Feliciano à faire un panel	28 août 2024	1h			
Commencement du boîtier sur Solidworks + écriture du rapport	28 août 2024	4h			
Ecriture du rapport	29 août 2024	2h40			
Rencontre de M.Déglon et explication du système	29 août 2024	20 min			
Continuer le boîtier sur solidoworks et créer un deuxième pour la carte SparkFun	29 août 2024	4h			
Réunion avec M.Bovey	29 août 2024	20min			
Terminer le boîtier contenant le PCB SparkFun + changement des ouvertures qui ne convenait pas	30 août 2024	2h			
Ecriture du rapport	30 août 2024	5h			
Ecriture du rapport	31 août 2024	1h			
Création page HTML	1 septembre 2024	5h			
Ecriture du rapport	2 septembre 2024	4h			
Générer les mises en plan solidworks	2 septembre 2024	1h			

Tâches effectuées	Dates	Temps alloué	Problèmes rencontrés	Solutions aux problèmes	Informations et personnes de contact
Aider un collègue avec des problèmes Altium	2 septembre 2024	1h			
Création du projet sur CubeMxIDE avec assignation des pins	3 septembre 2024	1h			
Montage du PCB	3 septembre 2024	5h			
Montage du PCB	4 septembre 2024	8h			
Nettoyage + contrôle du PCB	5 septembre 2024	3h			
Firmware	5 septembre 2024	5h			
Firmware	6 septembre 2024	6h			
Réunion avec M.Bovey	6 septembre 2024	1h			
Firmware (e-paper)	7 septembre 2024	8h	Je n'arrivais pas à faire fonctionner le e-paper	Le problème était du au fait que j'avais plusieurs fois des fonctions pour le SPI, ce qui causait des problèmes, j'ai donc recommencé un projet de zéro en implémentant petit à petit les fichiers dont j'avais besoin en effectuant un build à chaque fois pour vérifier qu'il y avait pas d'erreurs.	
Software ( création fichier PHP pour le serveur)	8 septembre 2024	12h			
Décoder trame I2C + faire les deux fiches pour le contrôle du Checksum	9 septembre 2024	2h	Les trames étaient fausses	Ceci était dû au fait que je n'avais pas fait attention aux seuils de tension des lignes (elles étaient à 3,4[V] alors que j'aurais dû les mettre à 1,65[V] pour avoir la moitié de 3,3[V])	
Installation xampp et test du fonctionnement du serveur à l'école + flash l'ESP32 + tester un code exemple avec Arduino IDE	9 septembre 2024	9h	Ce qui m'a fait perdre le plus de temps était que l'ESP n'arrivait pas à rentrer en mode Download	Il suffisait de rebrancher le câble, cependant sur Arduino IDE, j'avais le problème qu'il voulait pas mettre le code, c'est parce qu'il faut appuyer encore sur SW2 et SW3 et attendre le bon moment (voir fichier mode d'emploi pour les détails)	
Faire le mode d'emploi pour xampp et arduino IDE	10 septembre 2024	3h			
Test de communication entre le serveur et l'ESP + écriture rapport	10 septembre 2024	4h			
Firmware (Essaie de communiquer le STM32 et ESP32)	11 septembre 2024	7h50	Conflit lorsque je lance mon code STM32, mon ESP32 s'éteint, cependant il est bien alimenté	La pin "ESP-EN" était à l'état low dans STMCubeMx, il fallait la mettre à high	
Réunion avec M.Bovey	11 septembre 2024	1h10			
Firmware (ESP32)	12 septembre 2024	10h	La librairie arduino n'a pas l'air d'avoir les bons pinouts pour mon ESP32	Essayer de recopier la librairie "HardwareSerial.h" et modifier ces deux lignes : #elif CONFIG_IDF_TARGET_ESP32C3 #define TX1 (gpio_num_t)19 en modifiant 19 par 6  Si pas possible -> passer ne AT command	
Firmware (ESP32)	13 septembre 2024	2h	Essaie de trouver comment faire la propre librairie	Changement de méthode -> j'ai utilisé un exemple trouvé sur github qui n'utilisait pas la librairie "HardwareSerial.h"	
Ecriture du rapport	13 septembre 2024	4h			
Firmware ( envoie de ESP32 au STM32)	14 septembre 2024	17h			
Firmware ( envoie de ESP32 au STM32)	15 septembre 2024	17h			
Ecriture du rapport	15 septembre 2024	4h			
Firmware (envoie du STM32 à l'ESP32)	16 septembre 2024	17h			
Test du code à l'ETML	17 septembre 2024	4h	Le pare-feu de l'ETML m'empêchait de recevoir correctement mes requêtes depuis le site web	Activer le mode "Bridge*" dans la VM sinon ça fonctionnait pas à cause des pare-feux	
Test trame UART	17 septembre 2024	2h	J'avais du mal à comprendre ma trame UART, comme j'utilise un double dans ma trame, celle-ci m'envoyait des valeurs étranges.	J'ai utilisé un convertisseur en ligne pour décoder ma trame : <a href="https://baseconvert.com/ieee-754-floating-point">https://baseconvert.com/ieee-754-floating-point</a>	
Ecriture du rapport	17 septembre 2024	2h			
Réunion avec M.Bovey	18 septembre 2024	1h			

Tâches effectuées	Dates	Temps alloué	Problèmes rencontrés	Solutions aux problèmes	Informations et personnes de contact
Ecriture du rapport	18 septembre 2024	6h			
Tests des différentes communications (UART, SPI, I2C)	19 septembre 2024	2h			
Ecriture du rapport	19 septembre 2024	9h			
Software ( ajout du webhook discord )	19 septembre 2024	2h			
Ecriture du rapport	20 septembre 2024	9h			
Ecriture du rapport	21 septembre 2024	12h			
Ecriture du rapport	22 septembre 2024	12h			
Ecriture du rapport	23 septembre 2024	15h			
Ecriture du rapport + annexes	24 septembre 2024	4h			

# PV de séance

## Procès-verbal du 22.08.2024

### Présents

- Mme Perret (MPT)
- M. Bovey (PBY)

### État des lieux

- Avancement phase pré-étude
  - Proposition de schéma bloc (schéma général et schéma bloc global)
  - Sélection des composants principaux : microcontrôleur, affichage e-paper, capteur de température et humidité
  - Recherche concernant l'alimentation (choix entre pile bouton ou supercondensateur)
- Rédaction rapport :
  - Chapitres traités : Introduction et pré-étude
- Planification et journal de travail
  - Planification cohérente
  - Journal de travail à jour

### Problèmes rencontrés

- Alimentation :
  - Le choix d'une pile bouton ou supercondensateur n'est pas approprié pour ce circuit à cause de la durée de vie souhaitée (min un mois) et des courants de pic trop élevés pour les piles boutons
- E-paper :
  - Le schéma proposé dans le datasheet V2 n'est pas le même que dans la V4, sachant que la V2 est indiquée pour un e-paper noir et blanc, tandis que la V4 est pour tous les types d'écrans (couleur compris)
- ESP32 :
  - Le mode serveur continu n'est pas viable sans alimentation continue (USB-C), car l'ESP32 consomme environ 82[mA] en mode réception. Si nous gardons l'alimentation par pile, la durée de vie du système serait d'environ 14 heures

### Solutions proposées

- Alimentation
  - Utilisation de piles AAA avec une autonomie d'environ 1200 mAh (trouvable en magasin)

- E-paper :
  - Utiliser le schéma indiqué dans le datasheet de la V2 au lieu de la V4, car la V2 semblait pour les e-paper blanc et noir et la V4 pour toutes sortes de e-paper
- ESP32 :
  - Activation du serveur via un bouton et déconnexion avec celui-ci ou alimentation via USB-C

### Décisions prises

---

- Alimentation
  - Alimentation à l'aide de trois piles AAA en série
- E-paper
  - PBY m'a conseillé de rajouter une résistance 0[ohm] sur la liaison pas très claire, ce qui permettrait de relier la piste au GND si nécessaire.
- ESP32 :
  - Pas de mode serveur continue, mais une page HTML en local sur un pc avec un point d'accès où l'ESP pourra récupérer les informations

### Suite du projet / objectifs - jusqu'au 30 août

---

- Finaliser la pré-étude
  - Rédaction
- Réaliser le schéma électrique de la carte
- Choisir et commander tous les composants nécessaires au système

### Prochaine réunion :

---

23.08.2024, 16h00, Salle R112 ETML-ES

Revue du schéma électrique

28.08.2024, 11h00, Salle R110 ETML-ES

Séance classique

### Destinataires de ce PV

---

Aurélie Cuagnier, Secrétaire

Philippe Bovey, Maître de diplôme

Lausanne le 28.08.2024

Mélissa Perret

# PV de séance

## **Procès-verbal du 29.08.2024**

### **Présents**

- Mme Perret (MPT)
- M. Bovey (PBY)

### **État des lieux**

- Avancement schématique et conception :
  - Vérification schématique (le 23 août 2024)
  - Commande du PCB (le 28 août 2024)
- Boitier :
  - Boitier PCB terminé
  - Boitier capteur de température et humidité à terminer
- Rédaction rapport :
  - Chapitres traités : schématique
- Planification et journal de travail :
  - Planification cohérente
  - Journal de travail à jour

### **Problèmes rencontrés**

*Pas de problèmes rencontrés*

### **Solutions proposées**

-

### **Décisions prises**

- Schématique :
  - Changement de la diode sur l'alimentation -> mettre une diode Schottky avec une tension directe de 0.3[V] et non 0.7[V]
  - Rajout d'un quartz externe pour les signaux
- Conception :
  - Mettre les points de tests pour les signaux sur le bottom, ce qui permettra de faciliter les tests, cependant les tests points pour l'alimentation peuvent rester sur le top

- Software :
  - Déterminer l'ordre des importances au niveau du code
    - Lecture du capteur de température et d'humidité
    - Communication entre le microcontrôleur et l'affichage e-paper
    - Configuration du module ESP32
    - Configuration des trames UART (pour la communication de l'ESP au STM32)
    - Création et configuration de la page HTML

### ***Suite du projet / objectifs - jusqu'au 30 août***

---

- Ecriture du rapport
  - Finaliser les parties suivantes : schématique - conception - boitier
- Terminer le boitier pour le capteur de température et humidité
- Montage de la carte
- Commencement de la partie Software

### ***Prochaine réunion :***

---

05.09.2024, 17h00, Salle R110 ETML-ES

Séance classique

### ***Destinataires de ce PV***

---

Aurélie Cuagnier, Secrétaire

Philippe Bovey, Maître de diplôme

Lausanne le 30.08.2024

Mélissa Perret

# PV de séance

## **Procès-verbal du 06.09.2024**

### **Présents**

- Mme Perret (MPT)
- M. Bovey (PBY)

### **État des lieux**

- Rédaction rapport :
  - Chapitres traités : Montage, Hardware, Software
- Planification et journal de travail :
  - Planification cohérente
  - Journal de travail à jour
- Software :
  - Lecture du capteur de température/humidité (réception de la température et l'humidité, mais manque de checksum)
  - E-paper ne fonctionne pas correctement (impossible d'afficher autre chose que des pixels noirs)

### **Problèmes rencontrés**

- Software :
  - L'e-paper affiche seulement un écran noir, suite à l'utilisation de la librairie de Waveshare qui n'utilise pas le même microcontrôleur

### **Solutions proposées**

- Software :
  - Ajout de « define » pour les pins posant problèmes, et changement de l'optimisation du code (level 1).

### **Décisions prises**

- Software :
  - S'assurer que la librairie soit compatible avec le STM32F0 au lieu de F1, configuration des pins sur STM32CubeMX pour les signaux RST, DC, CS, BUSY
  - Regarder que le SPI communique avec le STM32, même si les signaux ne sont pas corrects
  - Après trois jours, si l'e-paper ne fonctionne toujours pas, passer au point configuration de l'ESP32

---

## **Suite du projet / objectifs - jusqu'au 11 septembre**

---

- Software :
  - Essayer d'afficher quelque chose sur l'e-paper
  - Effectuer toutes les mesures possibles pour vérifier que la communication SPI s'établit entre le microcontrôleur et l'e-paper

### **Prochaine réunion :**

---

11.09.2024, 10h00, Salle R110 ETML-ES

Séance classique

### **Destinataires de ce PV**

---

Aurélie Cuagnier, Secrétaire  
Philippe Bovey, Maître de diplôme

Lausanne le 06.09.2024

Mélissa Perret

# PV de séance

## **Procès-verbal du 11.09.2024**

### **Présents**

- Mme Perret (MPT)
- M. Bovey (PBY)

### **État des lieux**

- Rédaction rapport :
  - Chapitres traités : Firmware, mode d'emploi
- Planification et journal de travail :
  - Planification cohérente
  - Journal de travail à jour
- Software :
  - Lecture de capteur -> 100%
  - E-paper -> affichage ok, mais récupération des valeurs depuis l'esp non ok -> 90%
  - Configuration ESP -> 0%
  - Serveur web -> 100%

### **Problèmes rencontrés**

*Pas de problèmes rencontrés*

### **Solutions proposées**

-

### **Décisions prises**

- Software :
  - Communication entre ESP32 et STM32 prioritaire
  - Récupération des valeurs du site web et mise à jour de l'e-paper

---

**Suite du projet / objectifs - jusqu'au 18 septembre**

---

- Ecriture du rapport
- Communication entre ESP32 et STM32 (envoie trame UART)

---

**Prochaine réunion :**

---

18.09.2024, Salle R110 ETML-ES

Séance classique

---

**Destinataires de ce PV**

---

Aurélie Cuagnier, Secrétaire  
Philippe Bovey, Maître de diplôme

Lausanne le 13.09.2024

Mélissa Perret

# PV de séance

## Procès-verbal du 18.09.2024

### Présents

- Mme Perret (MPT)
- M. Bovey (PBY)

### État des lieux

- Rédaction rapport :
  - Chapitres traités : Firmware
- Planification et journal de travail :
  - Planification cohérente
  - Journal de travail à jour
- Boitier
  - Boitier imprimé
- Software :
  - Lecture de capteur -> 100%
  - E-paper -> affichage avec récupération des valeurs depuis le site -> 100%
  - Configuration ESP -> 100%
  - Serveur web -> 100%
- Démo
  - Démo du système en fonctionnement

### Problèmes rencontrés

- Hardware :
  - MOSFET Q1 défectueux -> pas de tension sur le drain, mais tension correcte sur la source et la gate
- Questions posées :
  - 1. Dois-je mettre le listing de tous les codes utilisées depuis github en annexe ?
  - 2. Puis-je utiliser un convertisseur en ligne pour décoder le type double de mon UART ?

## Solutions proposées

---

- Hardware :
  - Faire le pont avec le jumper J2 pour simuler l'activation du transistor, ce qui me permet d'envoyer les alarmes (seuil et tension piles)
- Réponses aux questions :
  - 1. Lister les fonctions utilisées et les lignes modifiées
  - 2. Oui

## Décisions prises

---

- Hardware :
  - Laisser le pont sur le jumper J2 jusqu'à la présentation afin tout problème avec la carte vu qu'elle est fonctionnelle

## Suite du projet / objectifs - jusqu'au 24 septembre

---

- Ecriture du rapport
  - Finaliser le rapport

## Prochaine réunion :

---

## Destinataires de ce PV

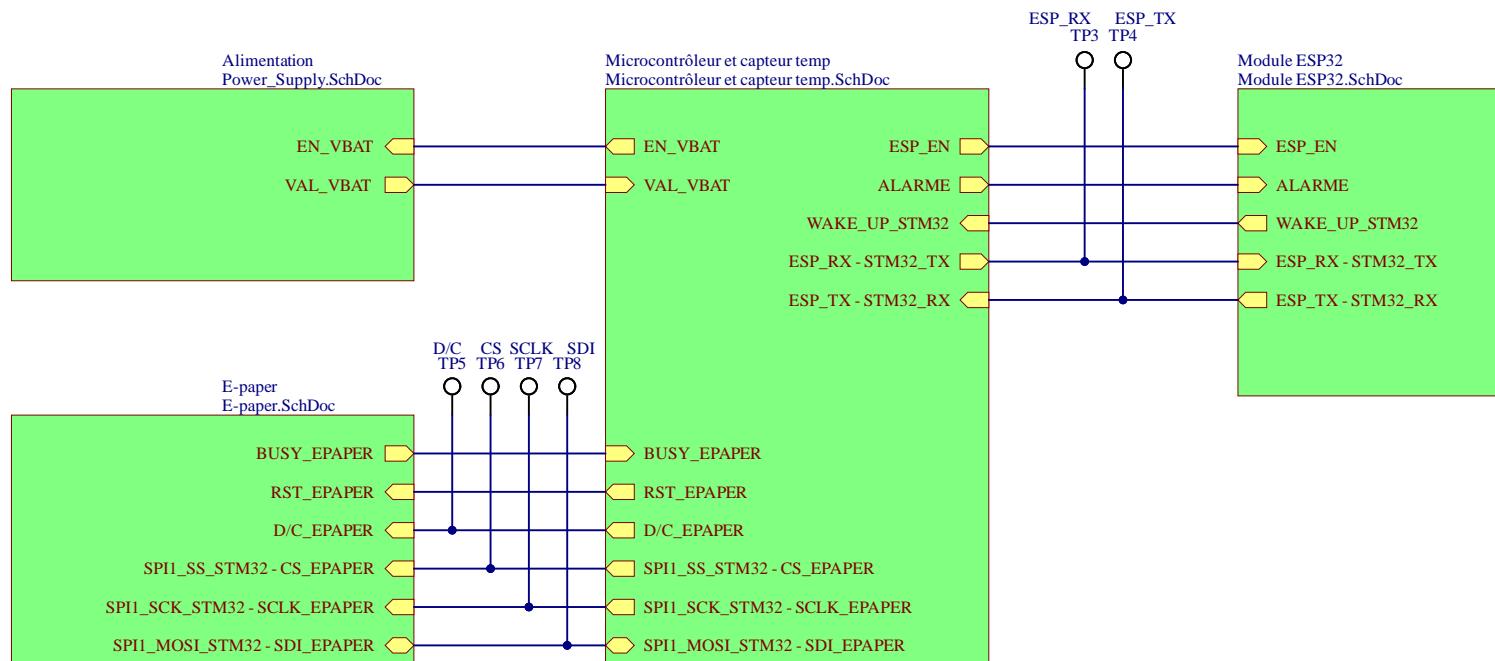
---

Aurélie Cuagnier, Secrétaire  
Philippe Bovey, Maître de diplôme

Lausanne le 19.09.2024

Mélissa Perret

# Annexe E

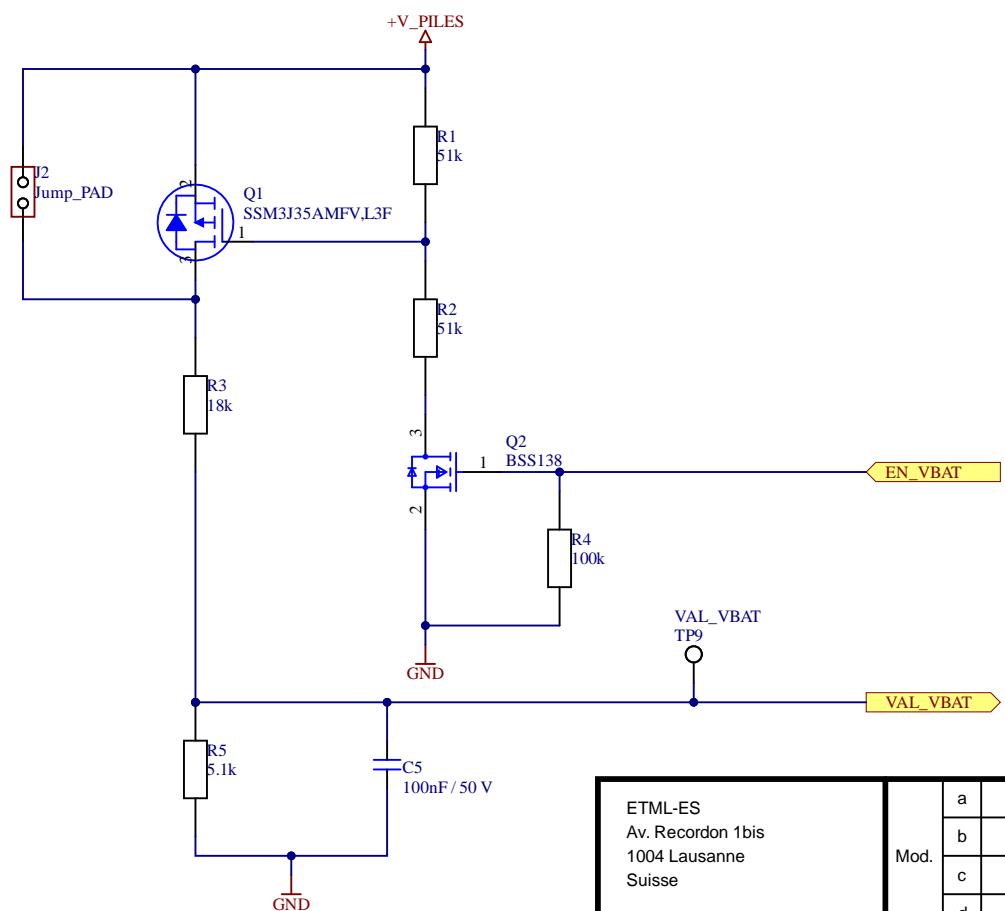


ETML-ES Av. Recordon 1bis 1004 Lausanne Suisse *	Mod. a b c d		Dessiné	MPT	21.08.2024	A4 Nb. feuillets 5	Echelle :  Feuille n° 1			
			Contrôlé	CFO	26.08.2024					
			Remplace							
		Nom du projet <b>MesureTH</b>								
		No <b>2409</b>								

## Entrée alimentation par piles

### Régulateur +3V3

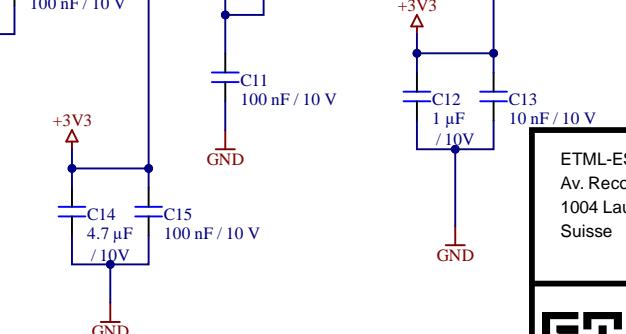
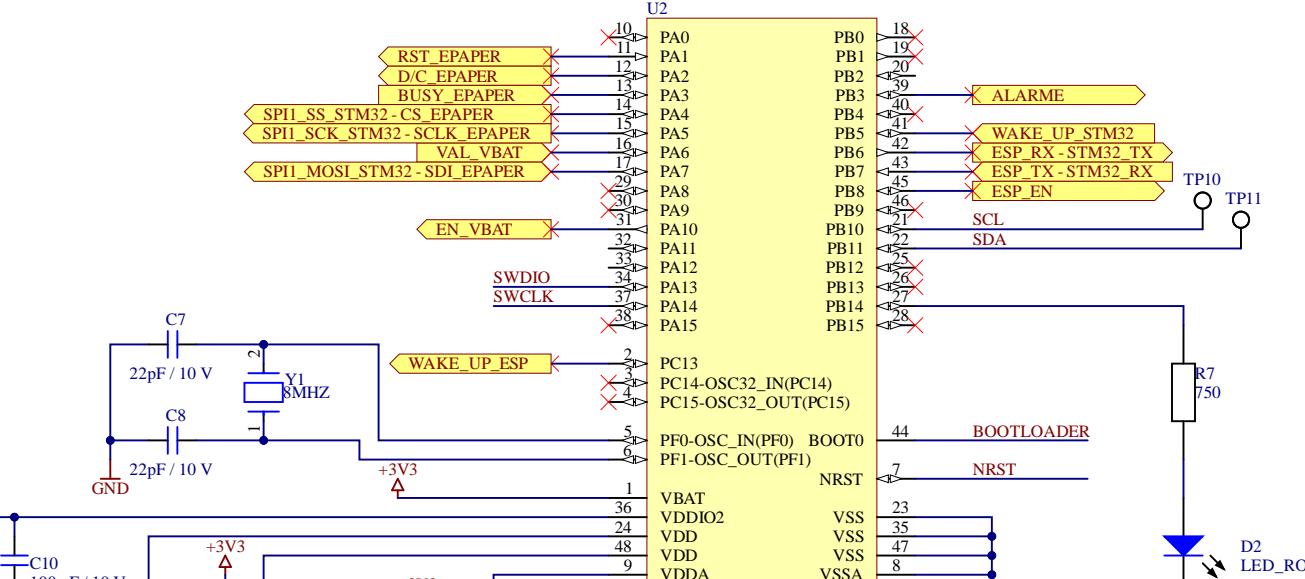
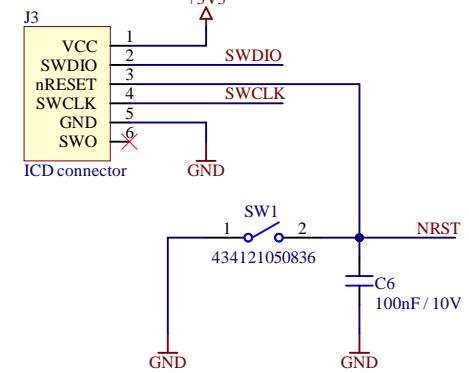
### Contrôle état de la pile



Mod.	a	Dessiné	MPT	21.08.2024	A4	Nb. feilles 5	Feuille n° 2	Echelle : —
	b	Contrôlé	CFO	26.08.2024				
	c							
	d	Remplace						
<b>ETML-ES</b>		Nom du projet <b>MesureTH</b>						No <b>2409</b>

# RESET ET DEBUG

# Capteur SparkFun (T et H)

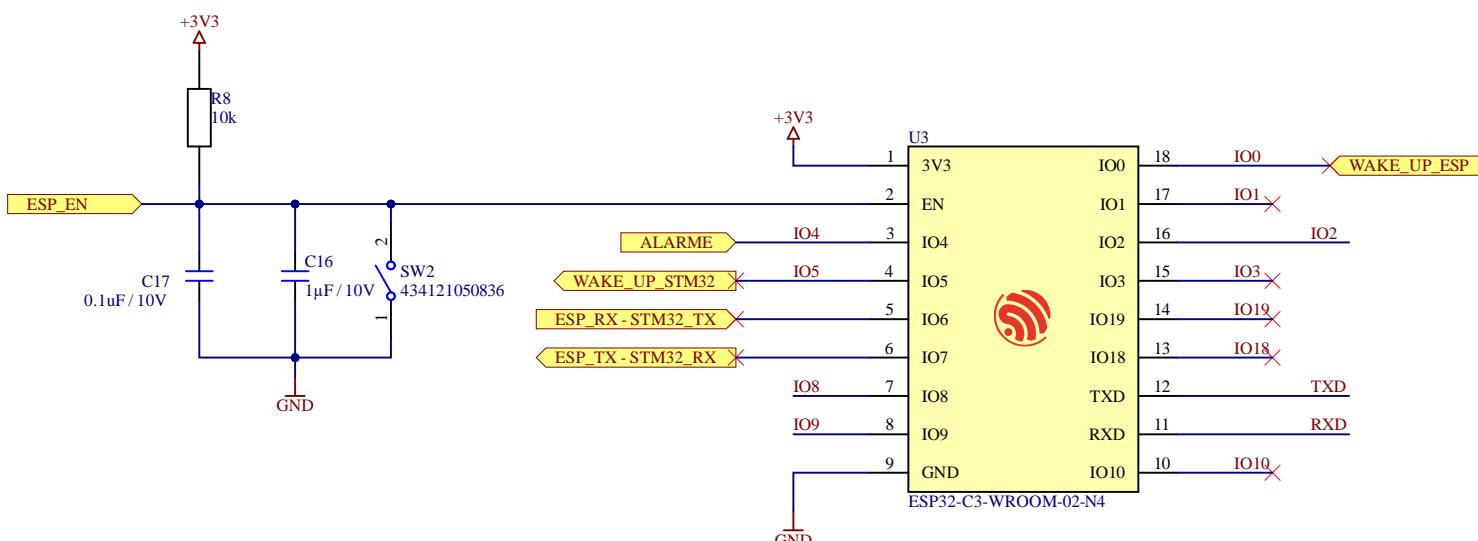


ETML-ES  
Av. Recordon 1bis  
1004 Lausanne  
Suisse

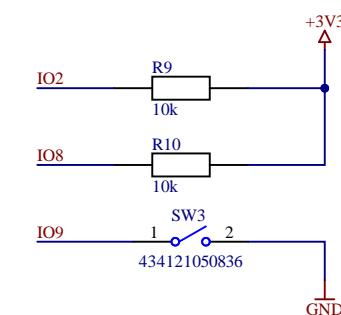
Nom du projet  
**MesureTH**

a	Dessiné	MPT	21.08.2024	<b>A4</b>		Echelle :
b	Contrôlé	CFO	26.08.2024			
c						
d	Remplace					
Nb. feilles	5	Feuille n°	3			
No	2409					

## Bouton de redémarrage

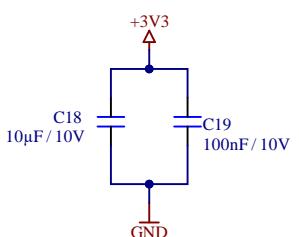


## Selection de mode

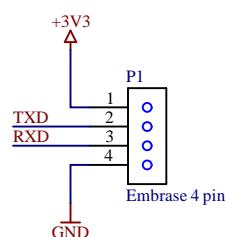


Pour lancer le mode "Download"  
 1) Garder appuyé sur ce bouton (SW3)  
 2) Appuyer sur le bouton de redémarrage (SW2)  
 3) Les deux boutons peuvent ensuite être relâchés

## Découplage

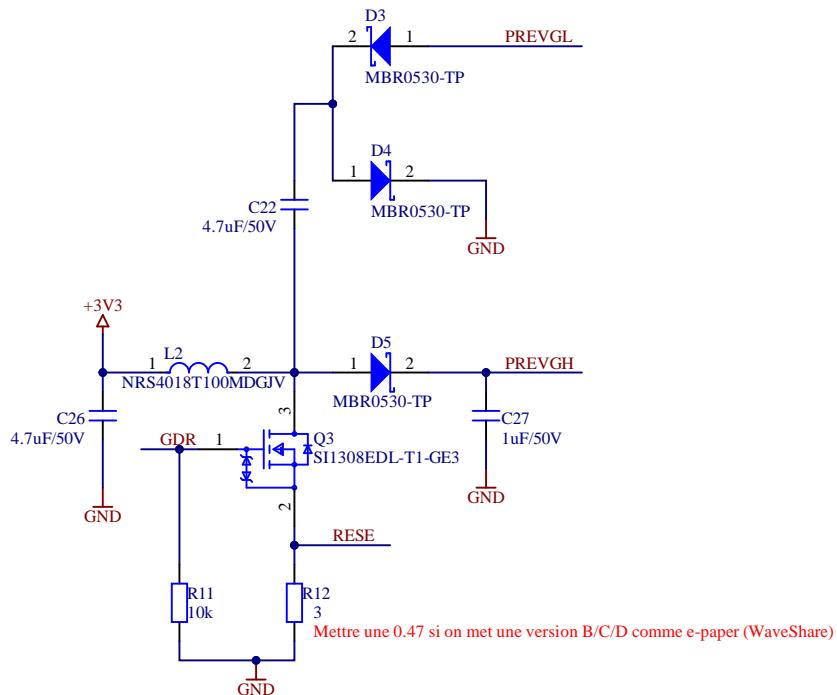


## Port de programmation

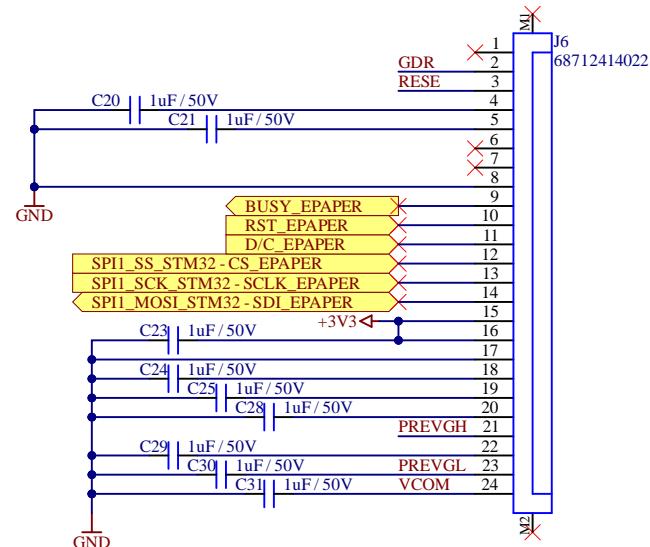


Mod.	a	Dessiné	MPT	21.08.2024	A4	Nb. feuillets 5	Feuille n° 4	Echelle : —	
	b	Contrôlé	CFO	26.08.2024					
	c								
	d	Remplace							
<b>ETML-ES</b> Av. Recordon 1bis 1004 Lausanne Suisse						No <b>2409</b>			
Nom du projet <b>MesureTH</b>									

## A Contrôle E-paper



## A Connecteur E-paper



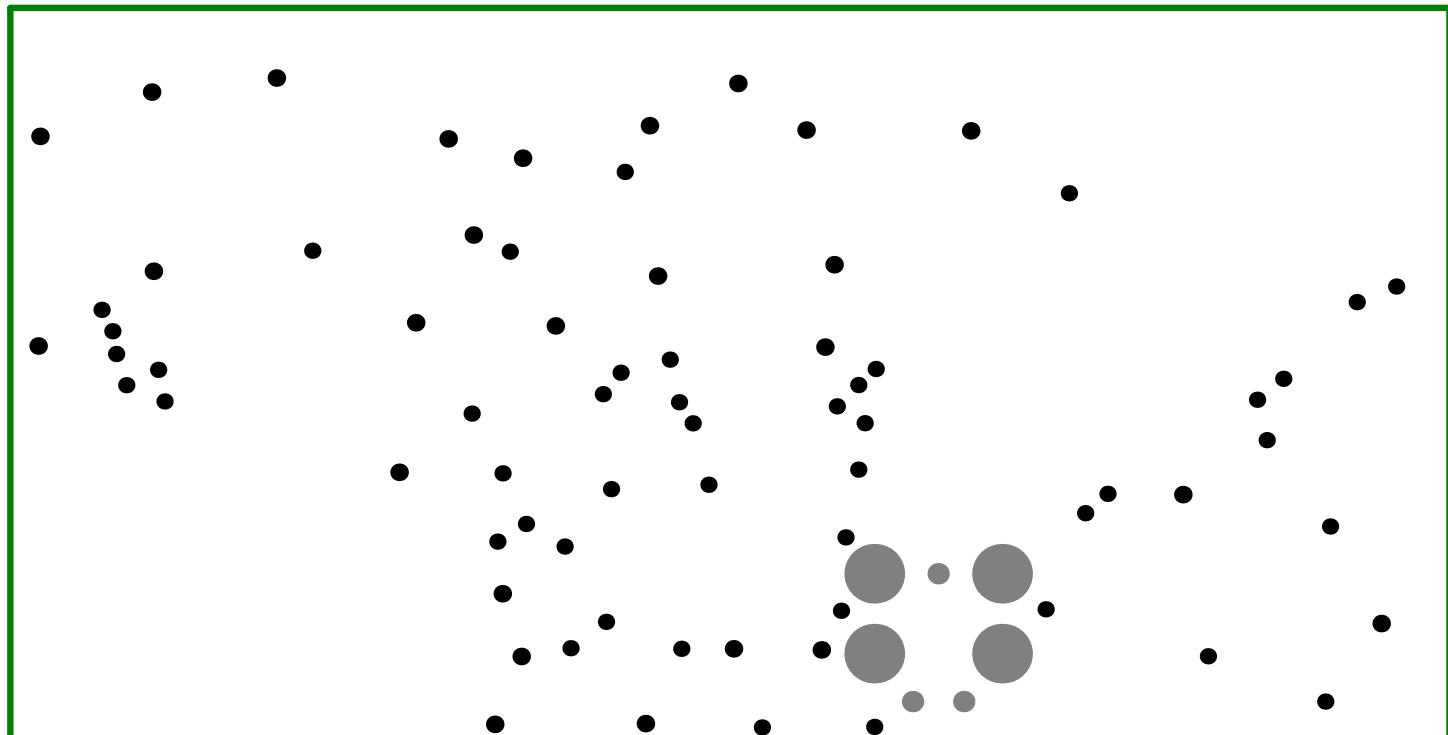
Mod.	a	Dessiné	MPT	21.08.2024	A4	Nb. feuillets 5	Feuille n° 5	Echelle : —	
	b	Contrôlé	CFO	26.08.2024					
	c								
	d	Remplace							
ETML-ES Av. Recordon 1bis 1004 Lausanne Suisse			Nom du projet <b>MesureTH</b>						
No <b>2409</b>									

# Annexe F.1

D2  
D1

2049\_MEASURETH\_V1

MPT



GND



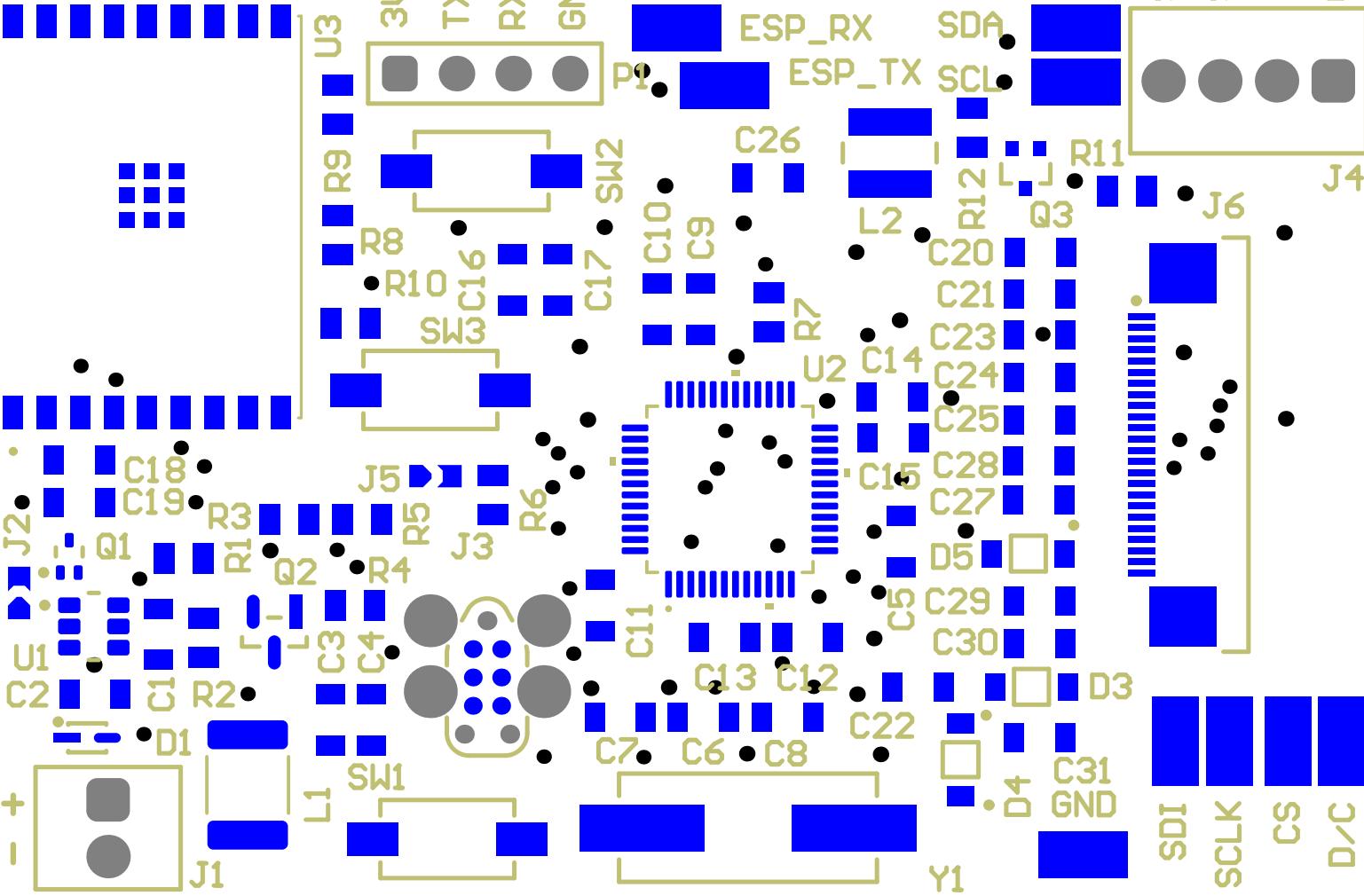
+3V3



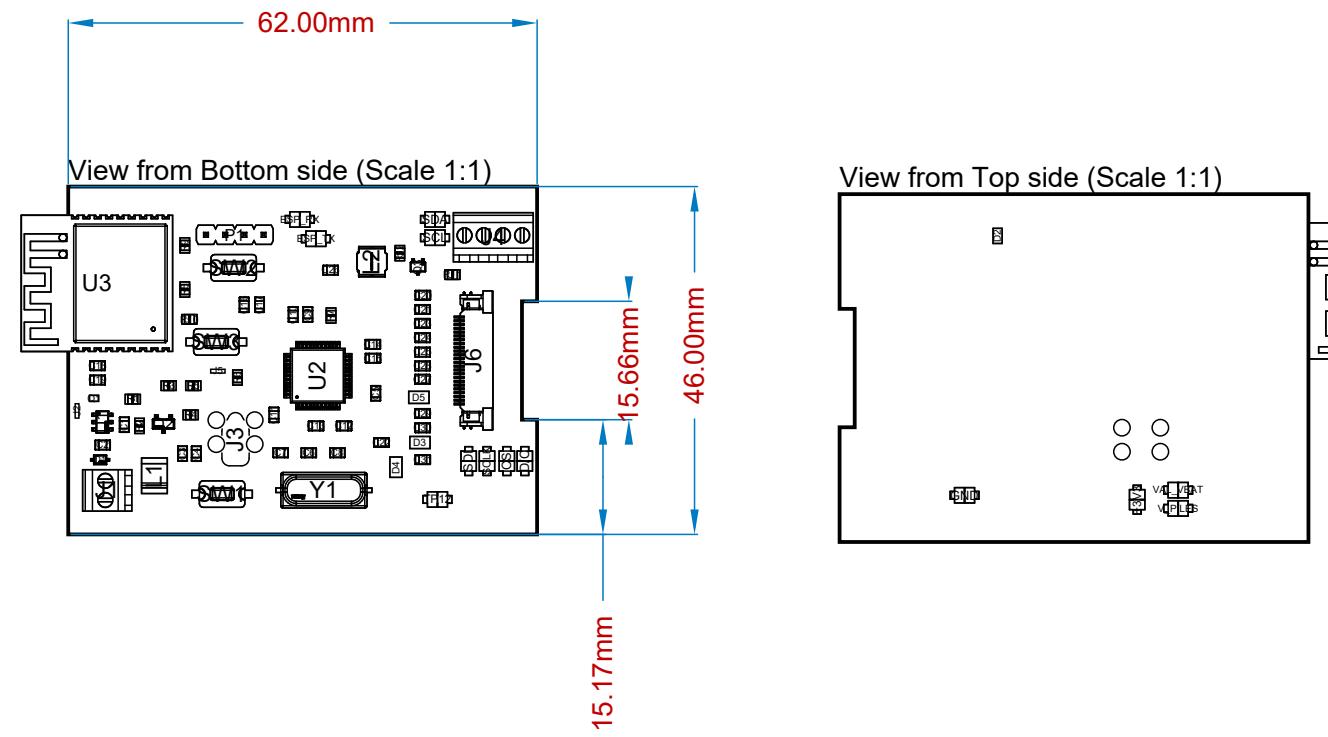
VAL\_VBAT

V\_PILES





# Annexe F.2



ETML-ES Av. Recordon 1bis 1004 Lausanne Suisse www.etml.ch	Mod.	a	.	Dessiné	MPT	28.08.24	A4	Nb. feuillets 3	Echelle : 1:1
		b	.	Contrôlé					
		c	.						
		d	.	Remplace	.				
		Nom du projet <b>ETML-ES</b> MesureTH				No 2409			

A

B

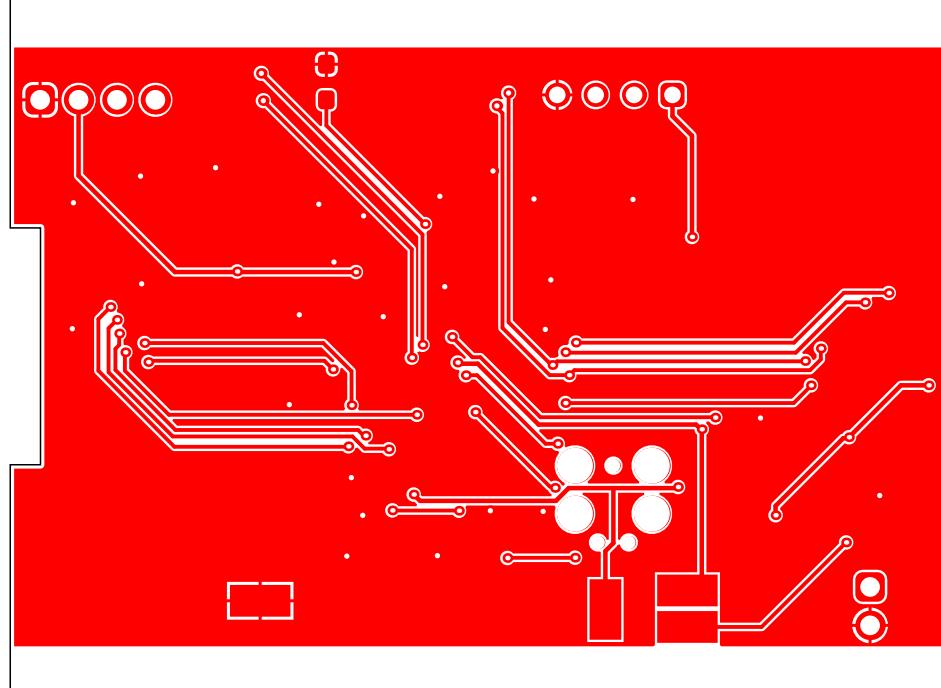
C

D

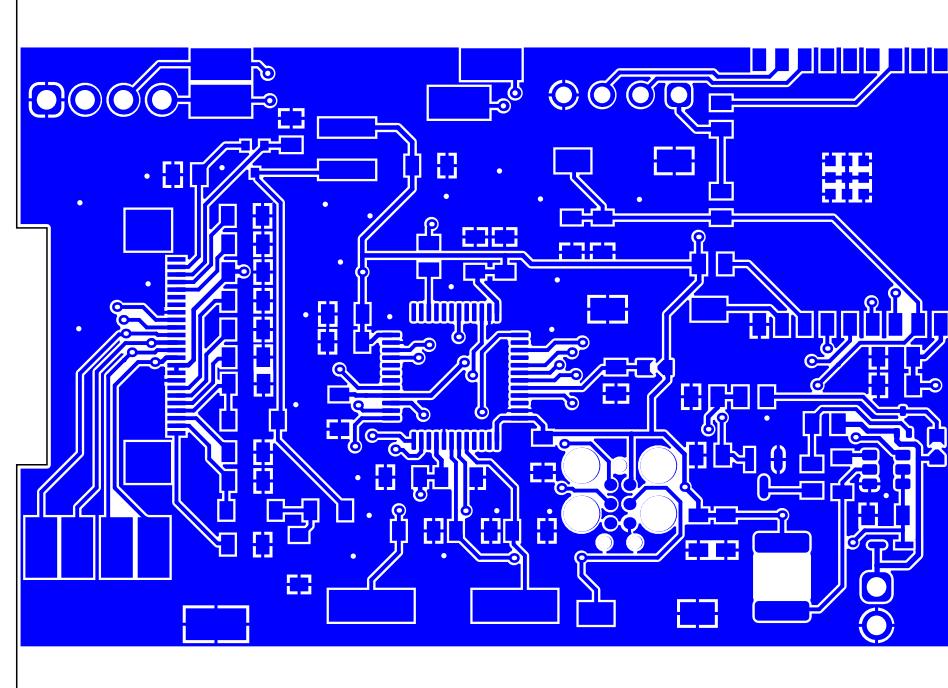
E

F

Top Layer (Scale 2:1)



Bottom Layer (Scale 2:1)



ETML-ES  
Av. Recordon 1bis  
1004 Lausanne  
Suisse  
[www.etml.ch](http://www.etml.ch)

**ETML-ES**

Mod.

a .

b .

c .

d .

Dessiné

Contrôlé

Remplace

MPT

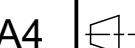
.

28.08.24

.

A4

Nb. feuillets  
3



Feuille n°  
2

Echelle :  
:1

Nom du projet  
**MesureTH**

No  
**2409**

A

B

C

D

E

F

A

B

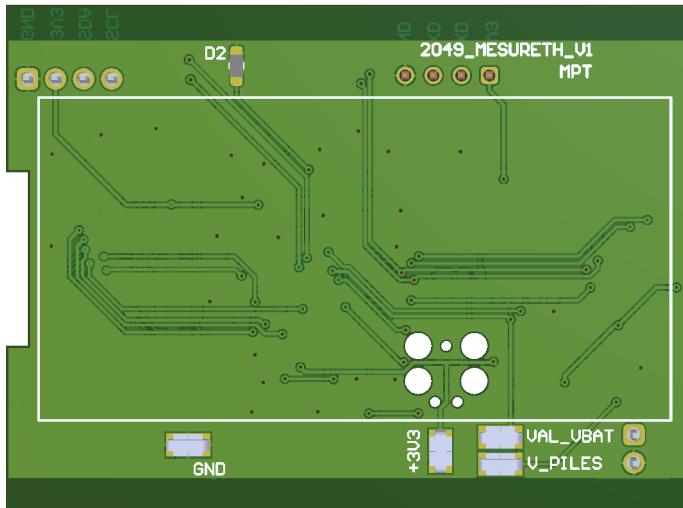
C

D

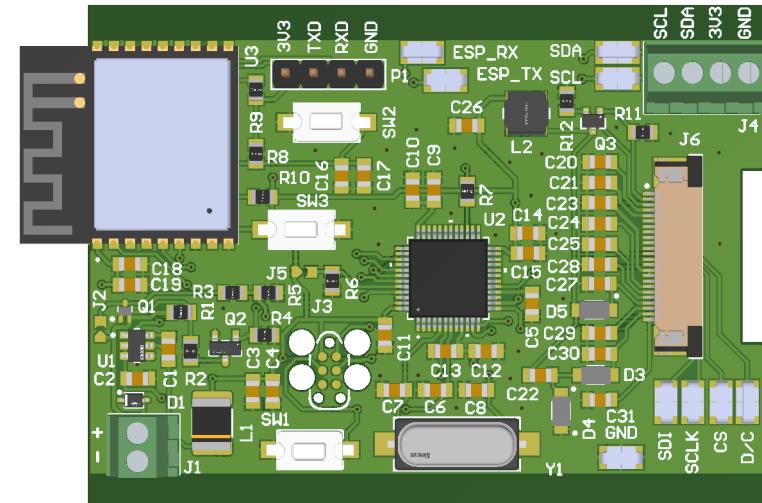
E

F

Realistic View



Realistic View



ETML-ES  
Av. Recordon 1bis  
1004 Lausanne  
Suisse  
www.etml.ch

**ETML-ES**

Nom du projet  
**MesureTH**

Mod.

a .

b .

c .

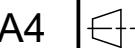
d .

Dessiné

MPT

28.08.24

A4



Echelle :  
1:1

Nb. feuillets

3

Feuille n°

3

No

2409

A

B

C

D

E

F

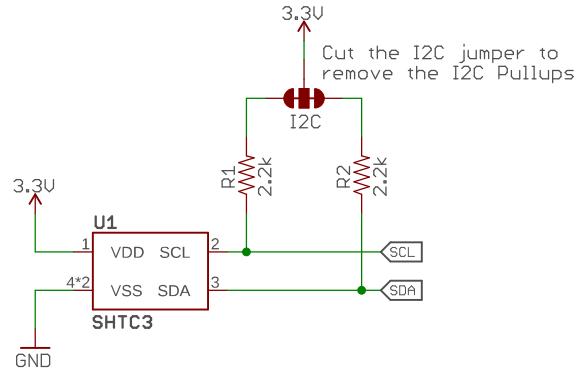
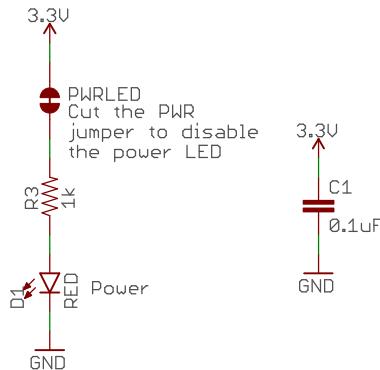
# Annexe G

BOM Cost		PROJET:	2409_MesureTH_RefrigerateurCongelateur_V1.PrjPcb							ETML-ES	Rue de Sébeillon 12 1004 Lausanne	
Date:	28.08.2024	Variante:	None							Quantité de production (nb PCBs) :		1
Par:	MPT	Source:	2409_MesureTH_RefrigerateurCongelateur_V1.BomDoc							Devise :		CHF
Line #	Designator	Value	Name	Manufacturer 1	Manufacturer Part Number 1	Quantity	Supplier 1	Supplier Part Number 1	Supplier Order Qty 1	Supplier Unit Price 1	Supplier Subtotal 1	
C17		0.1uF / 10V	Kyocera AVX	0805ZC104JAT2A	1	Mouser	581-0805ZC104JAT2A	1		0.2062	0.21	
C12		1 μF / 10V	Kyocera AVX	0805ZC105KAZ2A	1	Digikey	478-3557-1-ND	1		0.5121	0.51	
C16		1μF / 10V	Kyocera AVX	0805ZC105KAZ2A	1	Digikey	478-3557-1-ND	1		0.5121	0.51	
C20, C21, C23, C24, C25, C28, C29, C30, C31		1uF / 50V	Vishay Vitramon	VJ0805Y104KFAAT	9	Mouser	77-VJ0805Y104KFAAT	10		0.2977	2.98	
C27		1uF/50V	Kyocera AVX	08055C105JAT2A	1	Mouser	581-08055C105JAT2A	1		1.0300	1.03	
R12		3	Vishay Dale	RCS08053R00FKEA	1	Digikey	541-10538-1-ND	1		0.2193	0.22	
L1		3.9uF	TDK EPCOS	B82432T1392K000	1	Digikey	495-B82432T1392K000CT-ND	1		0.7337	0.73	
C9, C14		4.7 μF / 10V	KEMET	C0805C475K8RAC7210	2	Mouser	80-C0805C475K8R7210	2		0.2614	0.52	
C22, C26		4.7uF/50V	Kyocera AVX	08055C474KAT4A	2	Mouser	581-08055C474KAT4A	2		0.3879	0.78	
R5		5.1k	Yageo	RC0805FR-075K1L	1	Digikey	311-5.10KCRCT-ND	1		0.0843	0.08	
C13		10 nF / 10 V	Kyocera AVX	0805ZC103MAT2A	1	Mouser	581-0805ZC103MAT2A	1		0.2193	0.22	
C18		10μF / 10V	Kyocera AVX	0805ZD106MAT4A	1	Digikey	478-KGM21AR51AT06MLCT-ND	1		0.3795	0.38	
C2		10μF / 25V	KEMET	C0805C106M3PAC7210	1	Mouser	80-C0805C106M3PACLR	1		0.3542	0.35	
R6, R8, R9, R10		10k	Yageo	RC0805FR-0710KL	4	Distrelec	30213971	4		0.0097	0.04	
R11		10k	Yageo	RC0805FR-0710KL	1	Distrelec	30213971	1		0.0097	0.01	
R3		18k	Yageo	RC0805FR-0718K2L	1	Digikey	311-18.2KCRCT-ND	1		0.0843	0.08	
C3, C4		22μF / 25V	Vishay Vitramon	VJ0805Y224KXXMT	2	Mouser	77-VJ0805Y224KXXMT	2		0.3879	0.78	
C7, C8		22pF / 10 V	Kyocera AVX	0805ZA220JAT2A	2	Mouser	581-0805ZA220JAT2A	2		0.1855	0.37	
R1, R2		51k	Yageo	RC0805JR-0751KL	2	Newark	49AK3520	2		0.0042	0.01	
C10, C11, C15		100 nF / 10 V	Kyocera AVX	0805ZC104JAZ2A	3	Mouser	581-0805ZC104JAZ2A	3		0.2867	0.86	
R4		100k	Yageo	RC0805FR-07100KL	1	Digikey	311-100KCRCT-ND	1		0.0843	0.08	
C1, C6, C19		100nF / 10V	Kyocera AVX	0805ZC104JAT2A	3	Mouser	581-0805ZC104JAT2A	3		0.2062	0.62	
C5		100nF / 50 V	Kyocera AVX	08055C104M4T4A	1	Mouser	581-08055C104M4T4A	1		0.2024	0.20	
R7		750	Panasonic	ERA-6AEB751V	1	Digikey	P750DACT-ND	1		0.1096	0.11	
J6		68712414022	Wurth Electronics	68712414022	1	Digikey	732-3515-1-ND	1		1.2300	1.23	
SW1, SW2, SW3		434121050836	Wurth Electronics	434121050836	3	Digikey	732-7049-1-ND	3		0.3458	1.04	
J1		691210910002	Wurth Electronics	691210910002	1	Mouser	710-691210910002	1		0.8518	0.85	
J4		691210910004	Wurth Electronics	691210910004	1	Digikey	732-691210910004-ND	1		1.6400	1.64	
Y1		ABLS-8.000MHZ-B2-T	Abraccon	ABLS-8.000MHZ-B2-T	1	Digikey	535-9089-1-ND	1		0.2361	0.24	
U1		AP63203WU-7	Diodes	AP63203WU-7	1	Mouser	621-AP63203WU-7	1		0.7337	0.73	

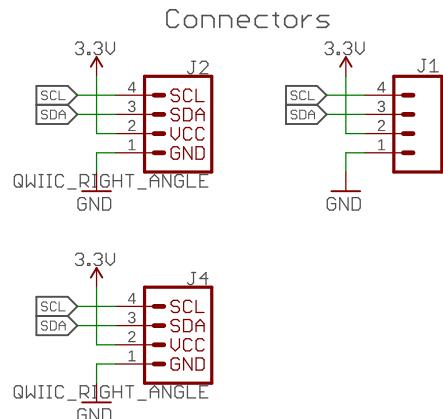
Line #	Designator	Value	Name	Manufacturer 1	Manufacturer Part Number 1	Quantity	Supplier 1	Supplier Part Number 1	Supplier Order Qty 1	Supplier Unit Price 1	Supplier Subtotal 1
D1		BAT60AE6327HTSA1	Infineon	BAT60AE6327HTSA1	1	Newark	85X4127	1	0.0911	0.09	
Q2		BSS138	ON Semiconductor / Fairchild	BSS138	1	Mouser	512-BSS138	1	0.2277	0.23	
P1		Embrace 4 pins	Wurth Electronics	61300411121	1	Digikey	732-5317-ND	1	0.1602	0.16	
U3		ESP32-C3-WROOM-02-N4	Espressif Systems	ESP32-C3-WROOM-02-N4	1	Digikey	1965-ESP32-C3-WROOM-02-N4CT-ND	1	1.6000	1.60	
J2, J5		Jump_PAD			2						
D2		LED RED	SunLED	XZCM2CRK54WA-1VF	1	Digikey	1497-XZCM2CRK54WA-1VFCT-ND	1	0.5229	0.52	
D3, D4, D5		MBR0530-TP	MCC	MBR0530-TP	3	Digikey	MBR0530TPMSCT-ND	3	0.1771	0.53	
L2		NRS4018T100MDGJV	Taiyo Yuden	NRS4018T100MDGJV	1	Digikey	587-3573-1-ND	1	0.2867	0.29	
		PCB	Eurocircuits	2409	1	Eurocircuits	2409	1	40.0000	40.00	
Q3		SI1308EDL-T1-GE3	Vishay Siliconix	SI1308EDL-T1-GE3	1	Mouser	78-SI1308EDL-T1-GE3	1	0.3711	0.37	
Q1		SSM3J35AMFV,L3F	Toshiba	SSM3J35AMFV,L3F	1	Mouser	757-SSM3J35AMFVL3F	1	0.1771	0.18	
U2		STM32F072C8T6TR	STMicroelectronics	STM32F072C8T6TR	1	Mouser	511-STM32F072C8T6TR	1	3.4200	3.42	
TP1, TP2, TP3, TP4, TP5, TP6, TP7, TP8, TP9, TP10, TP11, TP12,		Test_Point_SMD	Keystone Electronics	5019	13	Newark	59Y8500	13	0.2724	3.54	CHF 68.35
<b>Approved Notes</b>											

# Annexe H

Humidity Sensor  
VIN: 1.62-3.6V



7-bit unshifted I2C address: 0x70



Released under the Creative Commons  
Attribution Share-Alike 4.0 License  
<https://creativecommons.org/licenses/by-sa/4.0/>

**TITLE:** SHTC3 Breakout

Design by: Bryan Hoff  
Revision By: Andy England

**REV:**  
v10

Date: 5/6/2020 5:37 PM

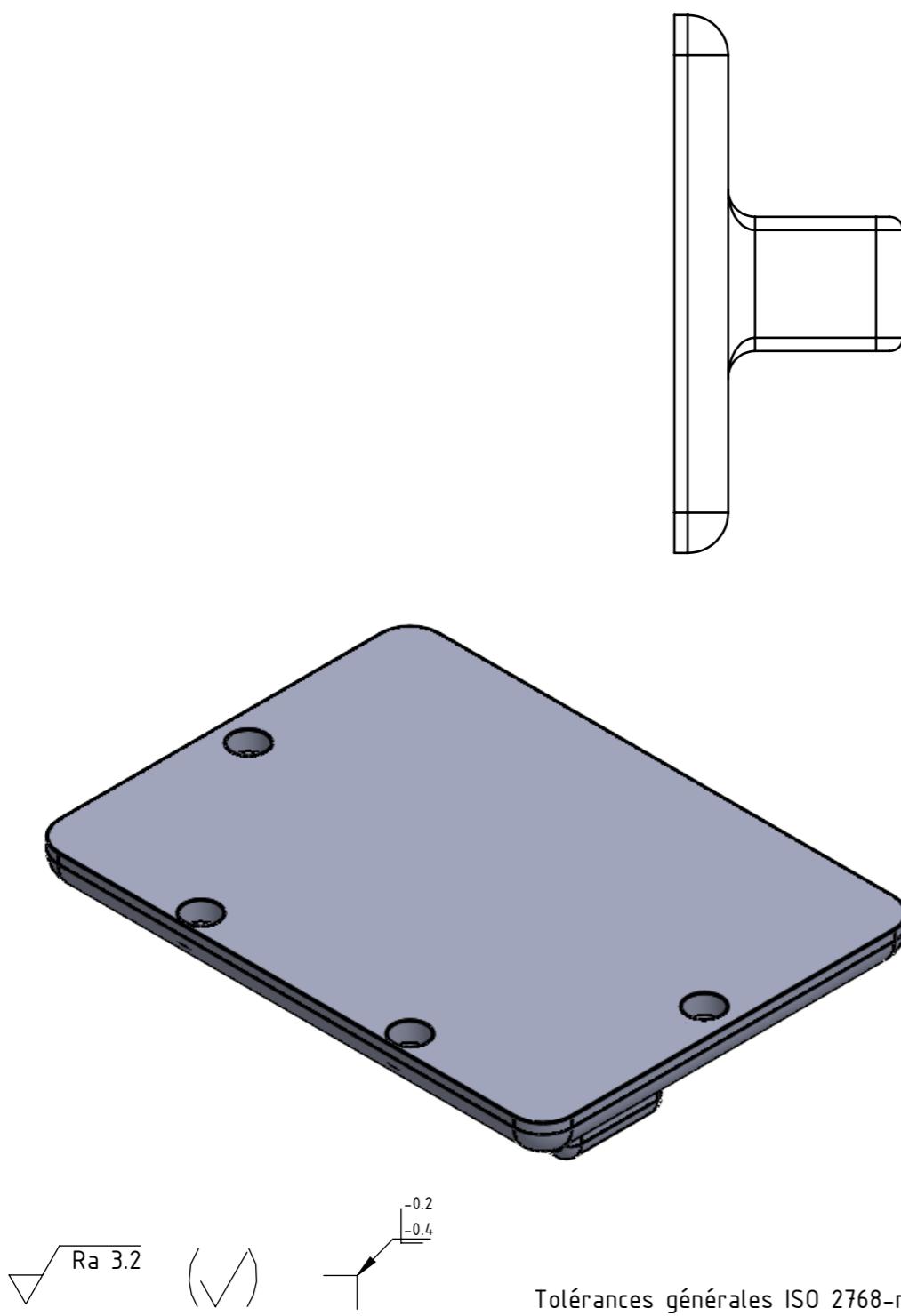
Sheet: 1/1



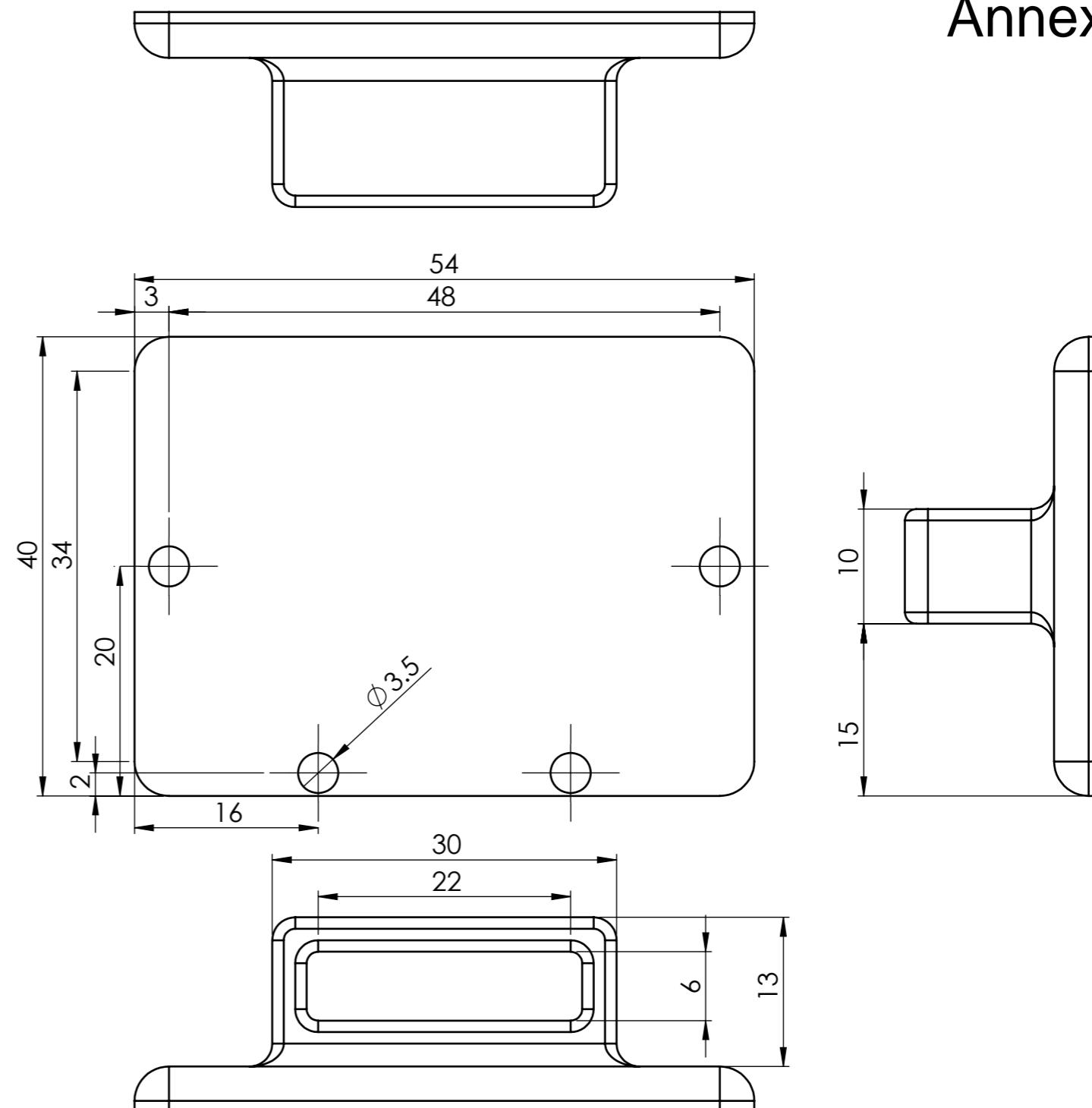
# Annexe I

<b>ETML-ES</b>			<b>Liste de contrôle PCB</b>	Page 1/1
Dessiné par	MPT		Date dessin	2024-08-26
Contrôlé par	CFO		Date contrôle	2024-08-28
N° document	2409		Nom du projet	MesureTH_V1
Dessinateur	Contrôleur	Pas applicable	Contrôle	Critères / Remarques
<b>Schéma électrique (SCH)</b>				
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Cartouche	Cartouche complet avec numérotation des pages
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Repérage	Lettres repères conforme aux normes CEI 81346
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Symboles	Symboles conformes aux normes CEI 60617
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Textes	Tous les textes lisibles avec deux sens de lecture
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Connexions	Fils verticaux ou horizontaux sur une grille de 2.5 mm
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Exhaustivité	Toutes les connexions nécessaires sont présentes
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Compilation	Le projet compile sans erreurs
<b>Tracé du circuit imprimé (PCB)</b>				
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Sérigraphie	Tous les textes lisibles avec deux sens de lecture
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Graphique / mécanique	Retirer la référence de la sérigraphie
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Dimensions	Coter le contour mécanique du PCB
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Découplage	Capacités de découplages au plus près des composants
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Alimentation	Plans ou topologie en étoile, éviter les boucles, GND et VCCs proches, largeur adaptée
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Routage	Angles de 45°, largeur appropriées, plan de masse sous les composants critiques (ex. quartz)
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Tables	Table des perçages, numérotation des couches cuivre
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Règles de conception	Contrôle des règles (DRC) sans erreur
<b>Pochoir (stencil, inclus dans le PCB)</b>				
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Calculs	Calculs des ratios IPC-7525B sur les pastilles critiques, sélection de l'épaisseur du pochoir
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Réduction	50 µm ou 10 % ou règle 50/50 pour le fine-pitch
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Arrondi	Ouvertures arrondies, pas d'angles droits
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Grandes pastilles	Réduction ou découpage en carrés (50 % surface)
<b>Assemblage (ASY)</b>				
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Cartouche	Cartouche complet avec numérotation des pages
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Textes	Tous les textes lisibles avec deux sens de lecture
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Valeurs	La valeur des composants est indiquée sur le dessin
<b>Liste de pièces (BOM)</b>				
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Cartouche	Cartouche complet avec numérotation des pages
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Contenu	Liste à jour, aux normes ETML, groupée valeur, boîtier
<b>Fichier de placement (PAP)</b>				
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Fichier de placement	Placement à jour, aux normes ETML
<b>Dossier de fabrication (Manufacturing)</b>				
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Dossier Titre_date.zip	Couches format Gerber RS-274X et perçages NC Drill

# Annexe J.1

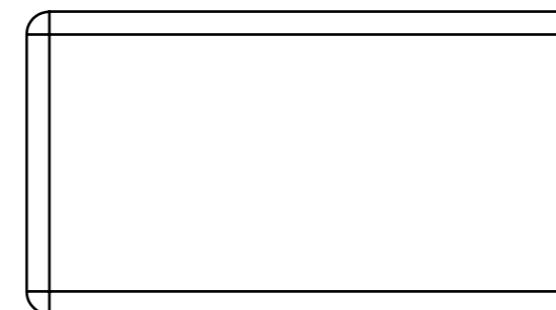
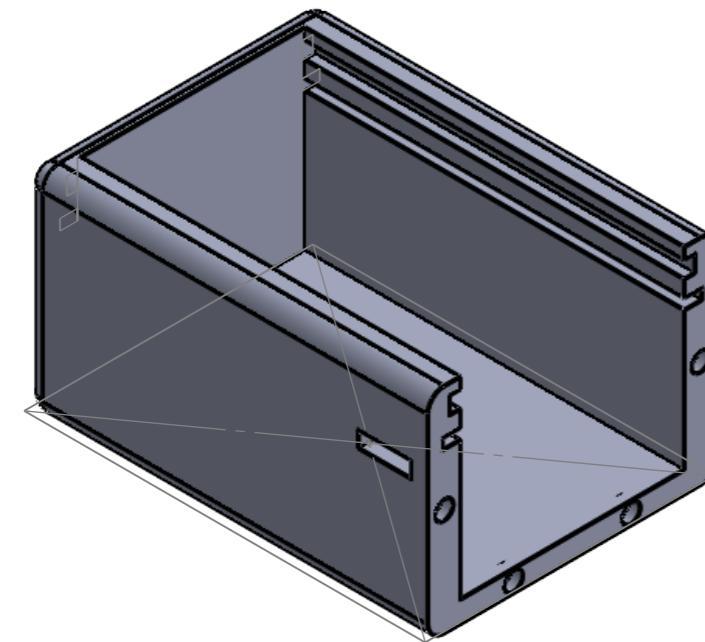
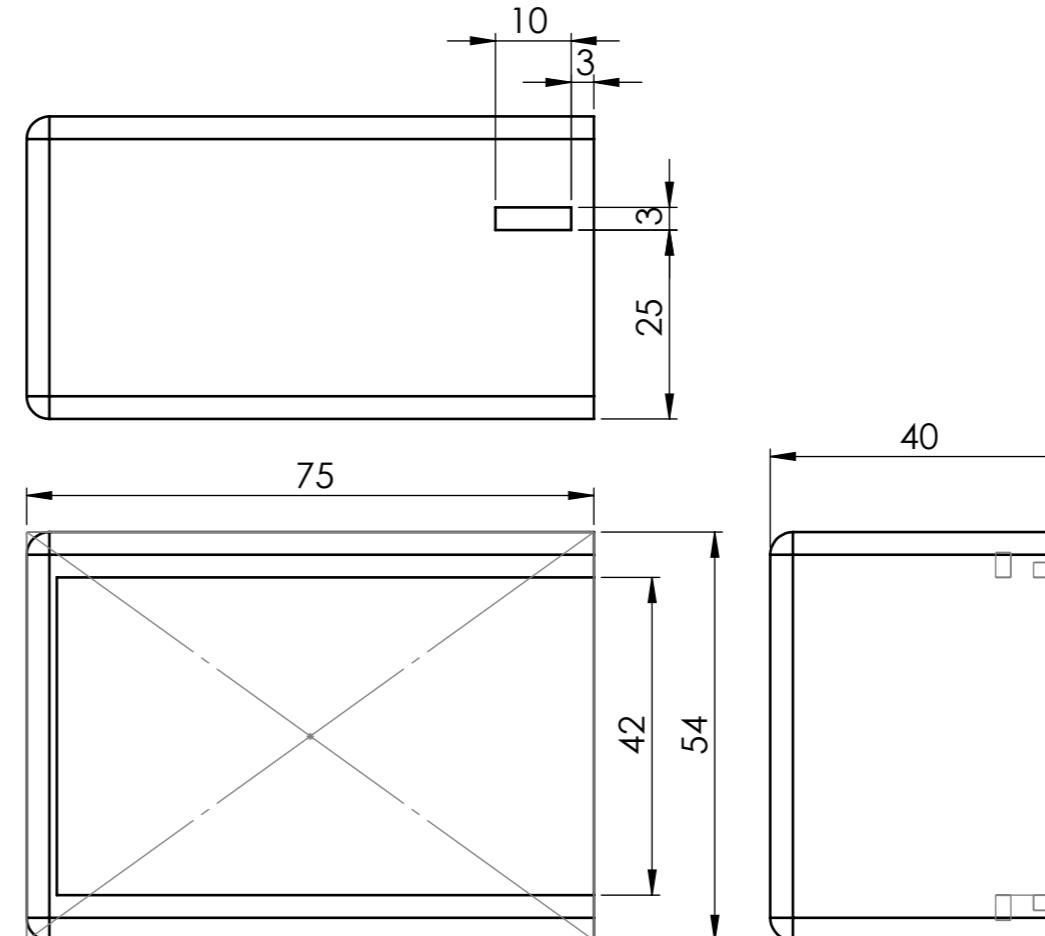
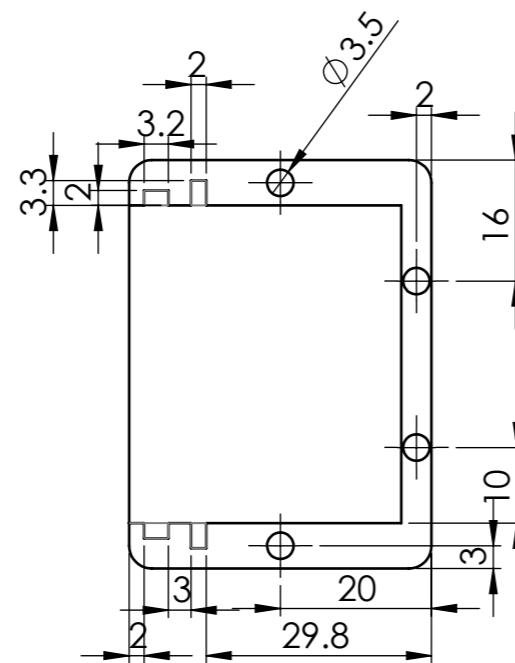


Tolérances générales ISO 2768-m (moyenne)



Numéro d'identification				Dénomination / Caractéristiques				Echelle 2:1	
Pos.	Quant.	Un.		A	E	F	G		
Mod.				B	Mod.	H			
				C					
				D					
<input type="checkbox"/> Sans nomenclature séparée		<input type="checkbox"/> N° de cmde		<input type="checkbox"/> Nomcl. sép. de même N°		<input type="checkbox"/> Origine		Feuille N° 1	
<input type="checkbox"/> Nomcl. sép. de N° diff.		<input type="checkbox"/> N° d'ident.		<input type="checkbox"/> Remplace					
<b>CTNL</b> Projet Dénomin. 2409_Couvercle_BoitierPCB_V1				N° de dessin					

## Annexe J.2

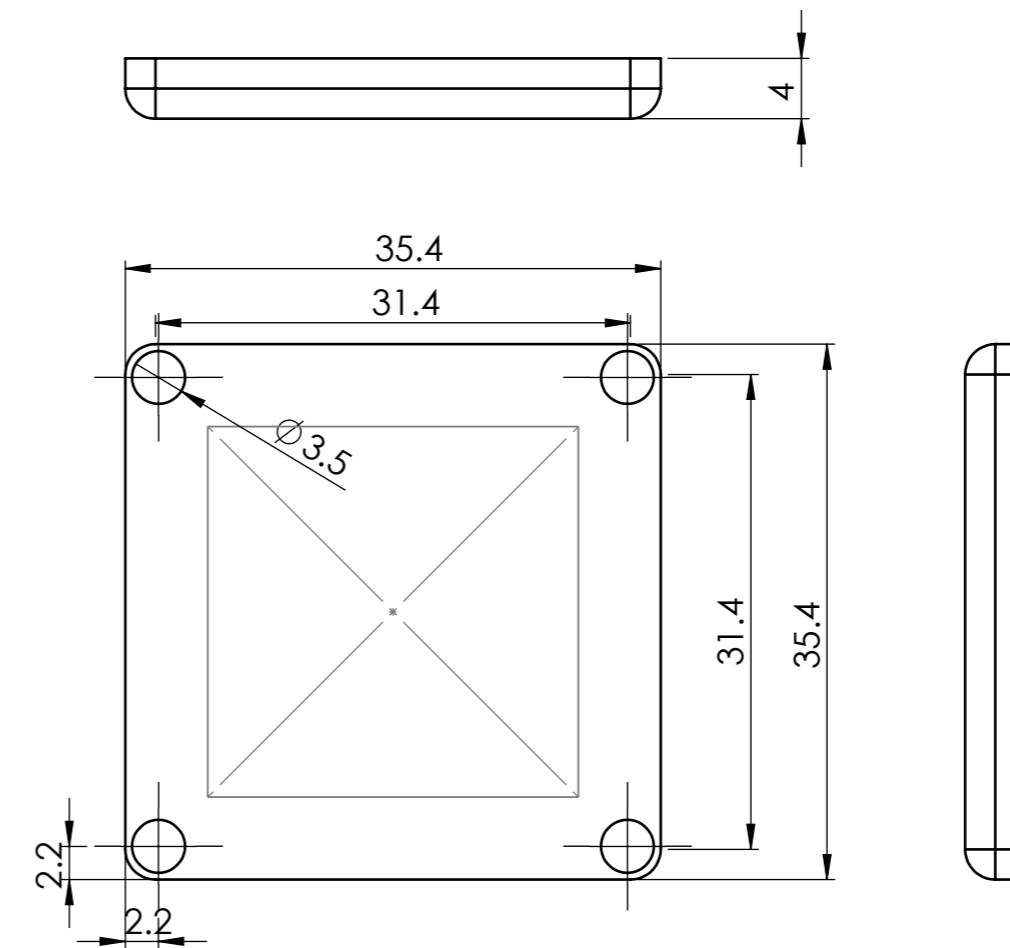
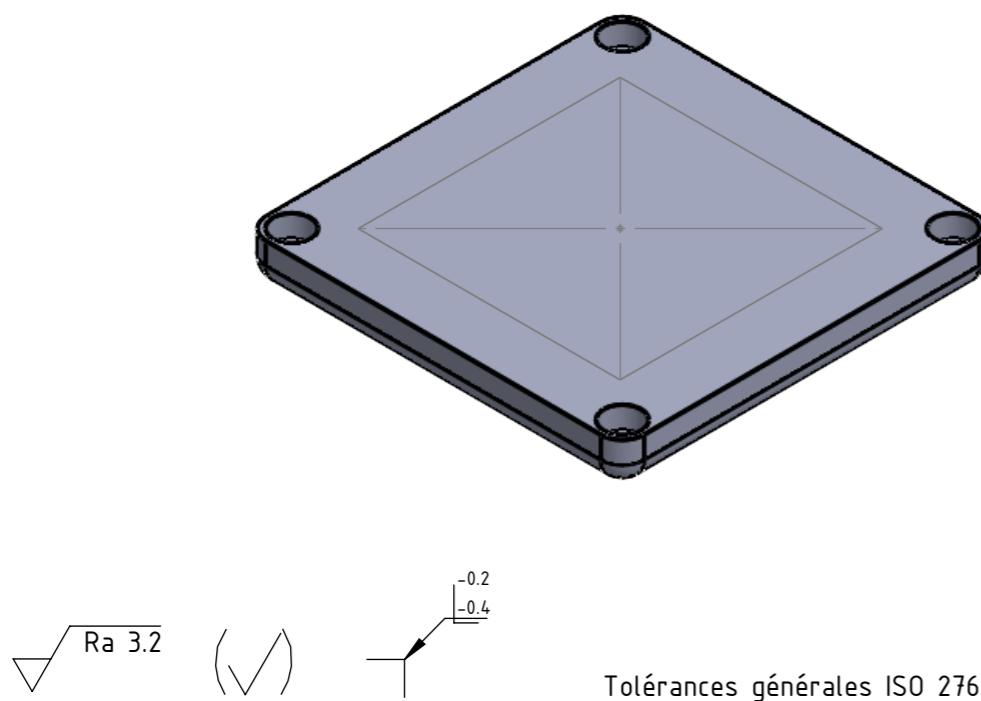


$\sqrt{Ra} 3.2$  (✓)  $-0.2$   $-0.4$

Tolérances générales ISO 2768-m (moyenne)

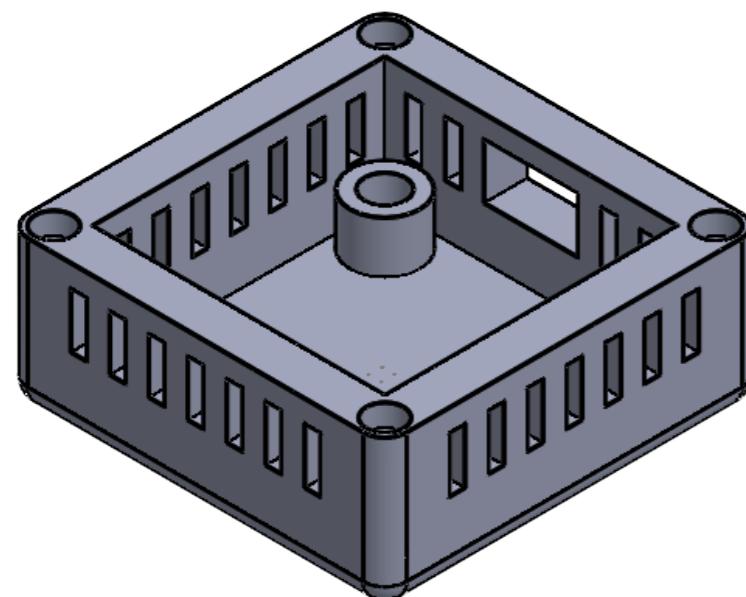
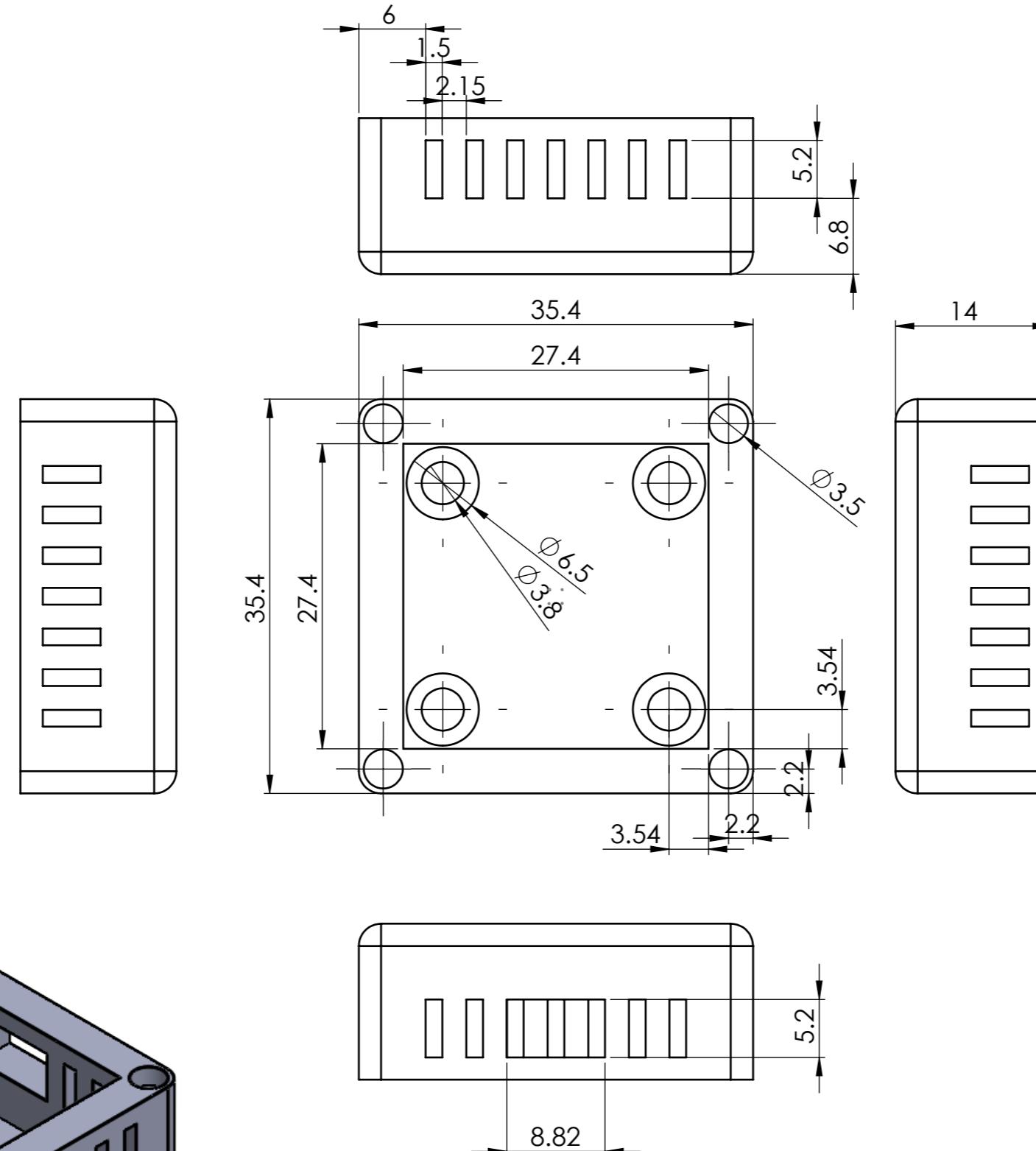
Pos.	Quant.	Un.	Numéro d'identification		Dénomination / Caractéristiques				
Mod.	A		Mod.	E	Dessiné	20.09.2024		Echelle 1:1	
	B			F	Contrôlé				
	C			G	Conf. aux normes				
	D			H	Bon pour exec.				
<input type="checkbox"/> Sans nomenclature séparée					N° de cmdé				
<input type="checkbox"/> Nomencl. sép. de même N°					Origine			Feuille N° 1	
<input type="checkbox"/> Nomencl. sép. de N° diff.					N° d'ident.	Remplace			
		Projet			N° de dessin				
		Dénomin. 2409_BoitierPCB_Base_V1							

## Annexe J.3



Pos.	Quant.	Un.	Numéro d'identification		Dénomination / Caractéristiques					
Mod.	A		Mod.	E	Dessiné	20.09.2024		Echelle 2:1		
	B			F	Contrôlé					
	C			G	Conf. aux normes					
	D			H	Bon pour exec.					
Sans nomenclature séparée <input type="checkbox"/>					N° de cmde					
Nomcl. sép. de même N° <input type="checkbox"/>					Origine			Feuille N° 1		
Nomcl. sép. de N° diff. <input type="checkbox"/>					N° d'ident.	Remplace				
<b>CTNL</b>		Projet			N° de dessin					
		Dénom. 2409_Couvercle_CapteurTemperature_V1								

## Annexe J.4



$\sqrt{Ra} 3.2$  (✓)  $-0.2$   $-0.4$

Tolérances générales ISO 2768-m (moyenne)

Numéro d'identification				Dénomination / Caractéristiques				Echelle 2:1
Pos.	Quant.	Un.	Mod.	E	F	G	H	
A					Dessiné	20.09.2024		
B					Contrôlé			
C					Conf. aux normes			
D					Bon pour exec.			
Sans nomenclature séparée <input type="checkbox"/>					N° de cmdé			
Nomcl. sép. de même N° <input type="checkbox"/>					Origine			
Nomcl. sép. de N° diff. <input type="checkbox"/>				N° d'ident.	Remplace			
<b>CTNL</b>				Projet Dénom. 2409_Base_CapteurTemperature V1				
				N° de dessin				

## Annexe K Liste de matériel

Désignation	Marque	Type	Caractéristique	N° d'inventaire
G1	GwINSTEK	GPS-3303	Alimentation	ES. SLO2. 00.00.26
P1	GwINSTEK	GDM-396	Multimètre	ES. SLO2.00.00.93
P2	Rohde & Schwarz	RTB2004	Oscilloscope	ES. SLO2.05.01.06
X1	-	-	PCB 2409	2409
X2	DELL	-	Ordinateur	SLO-R110-M306
-	Cisco	WRVS4400N V2	Routeur	-

## Annexe L

### L.1 UART

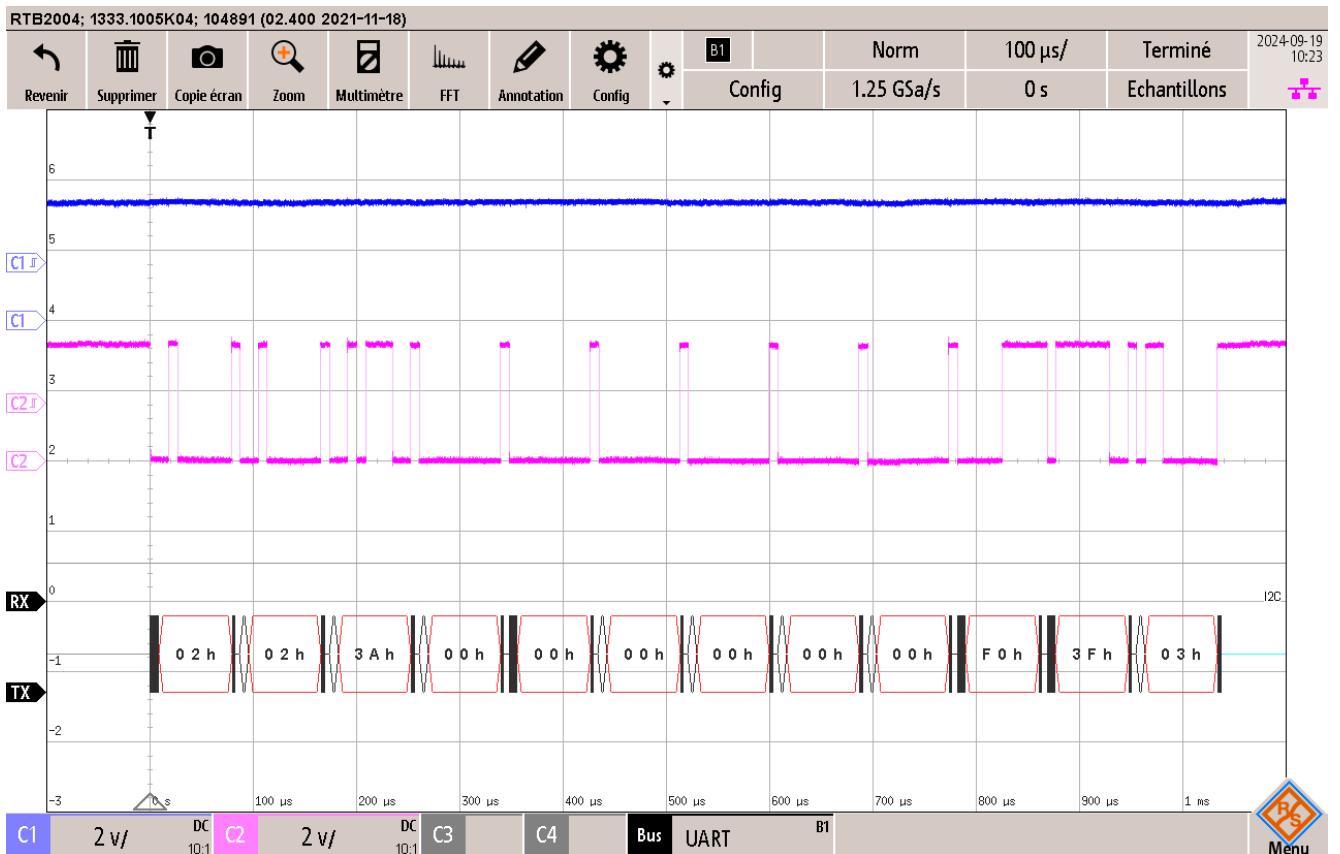
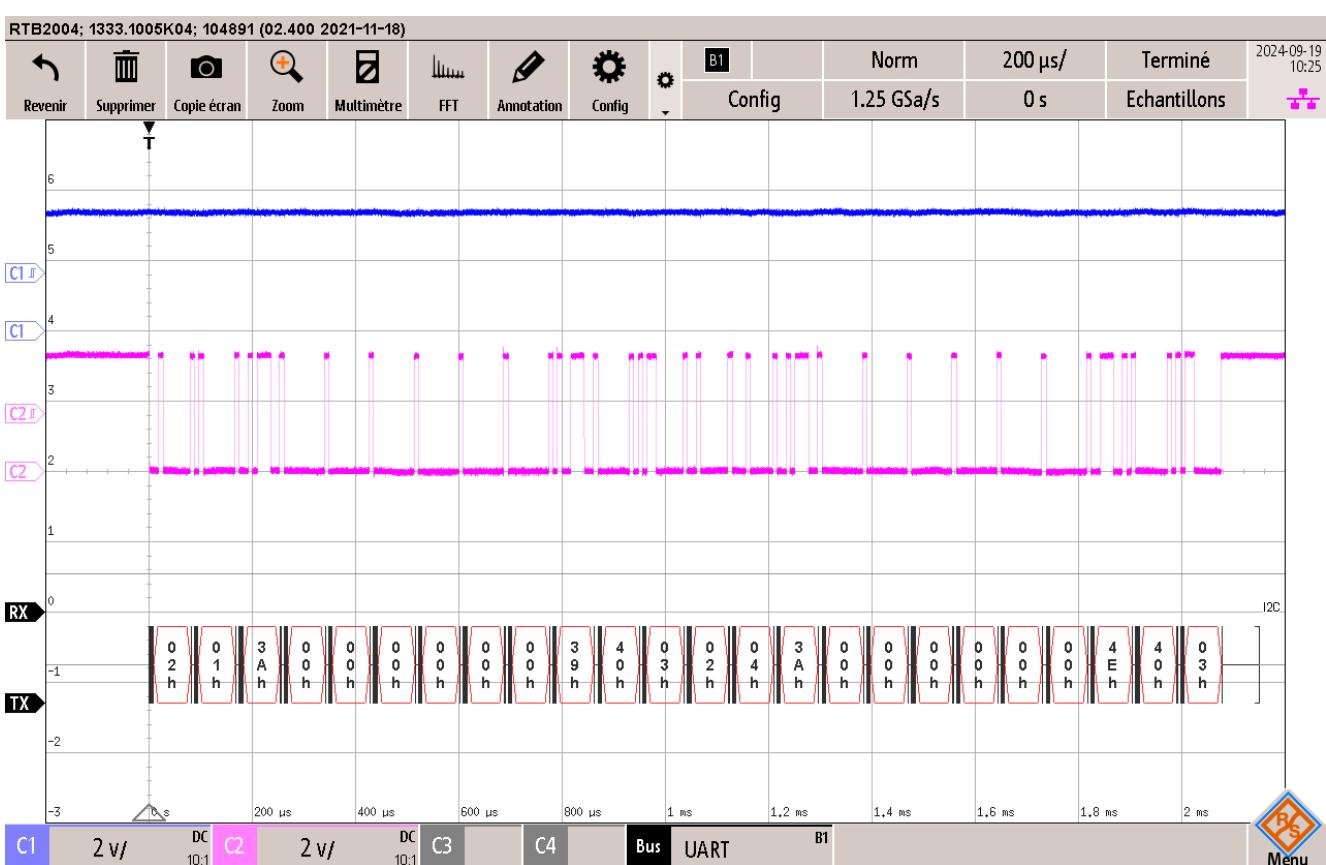
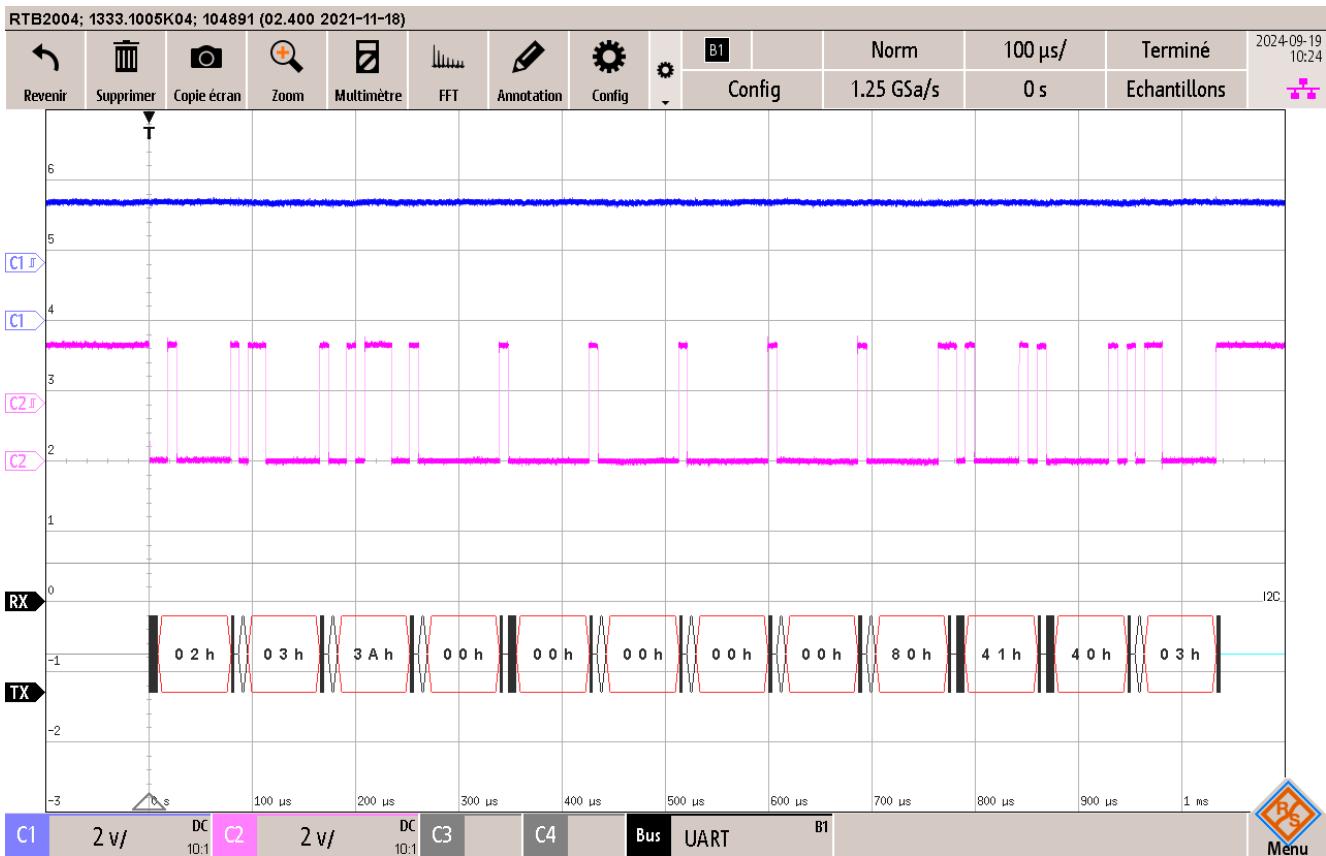
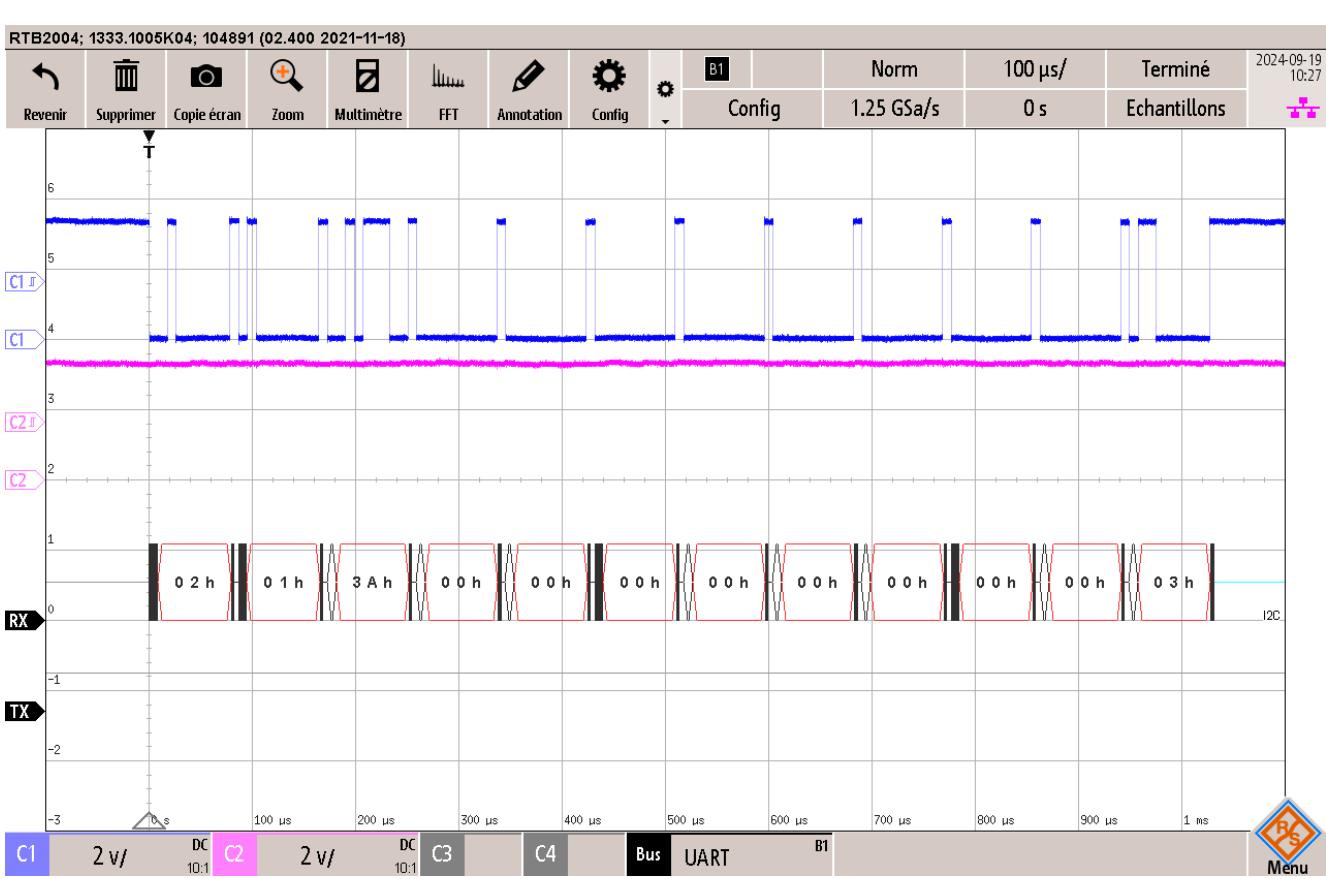
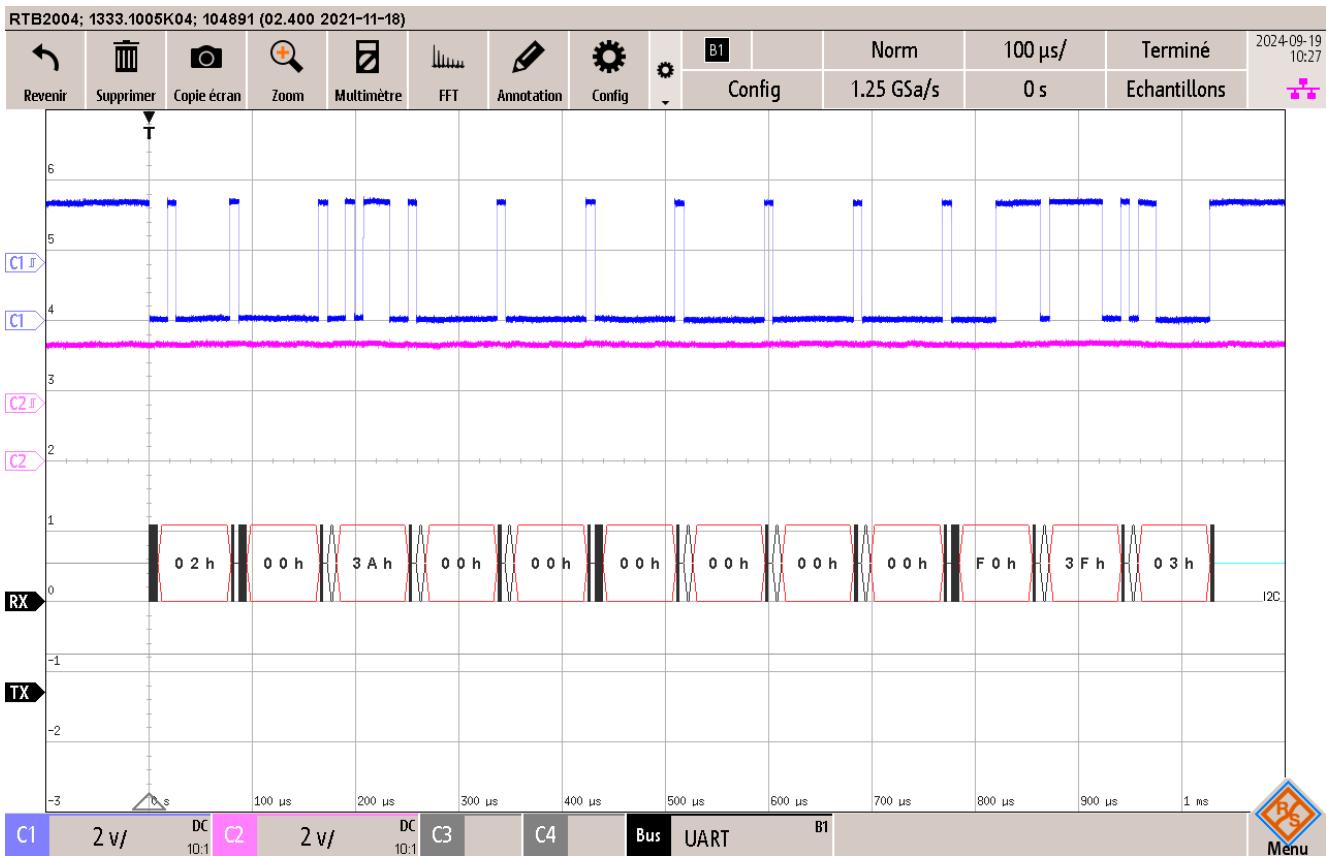


Figure 1 Envoie trame du serveur au STM32, modification index 2 à 1[°C]





## L.2 SPI

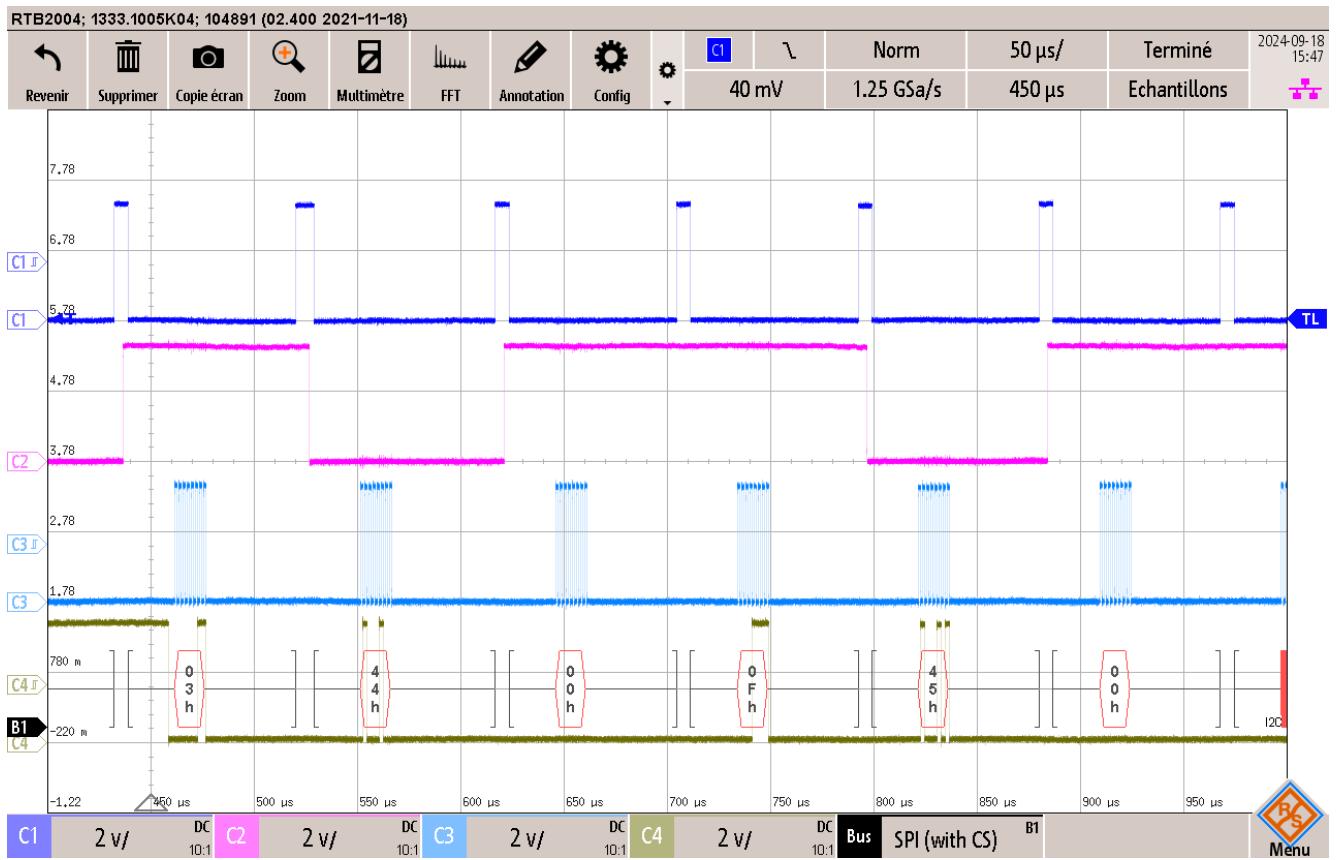


Figure 6 Deuxième trame d'initialisation avec le baudrate à 500[kBits/s]

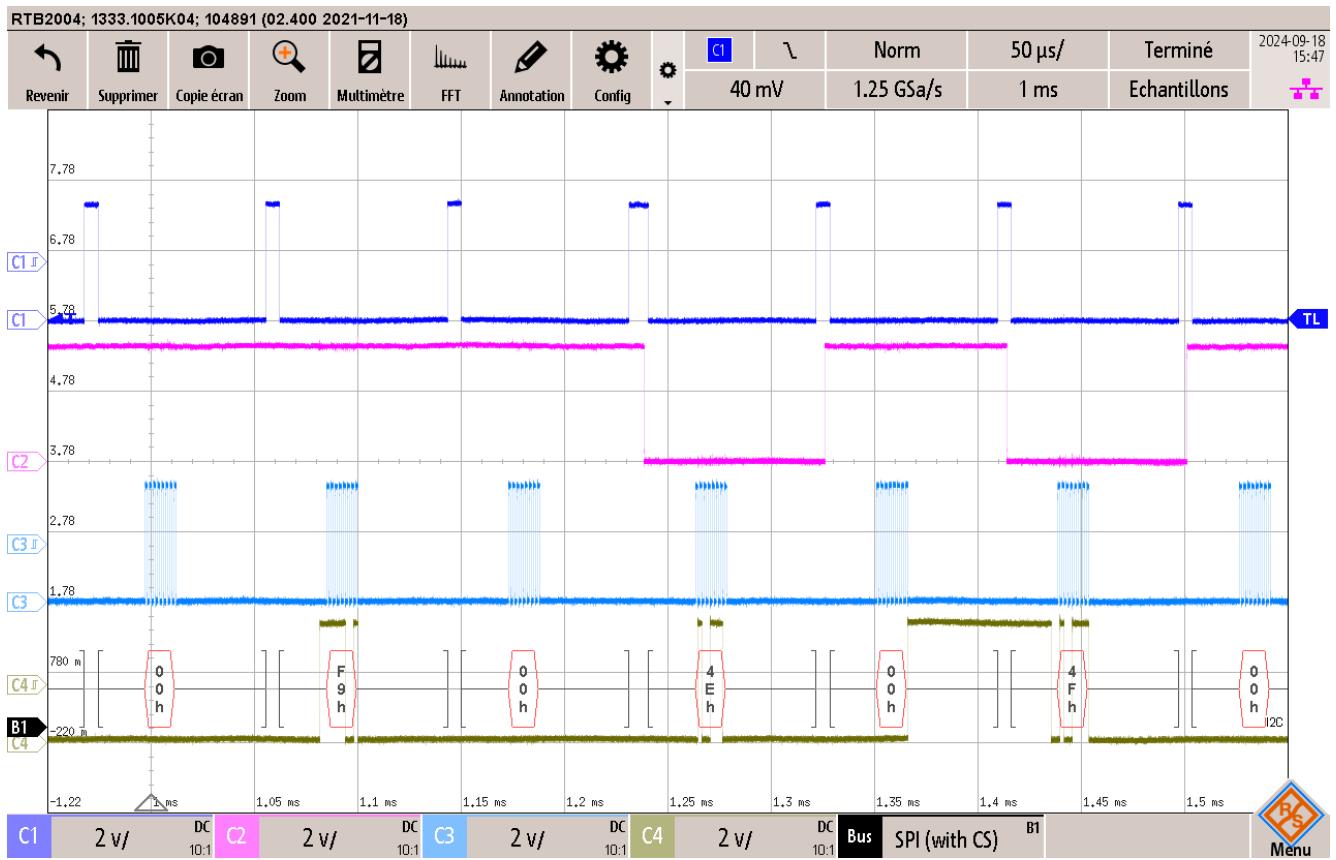


Figure 7 Troisième trame d'initialisation avec le baudrate à 500[kBits/s]

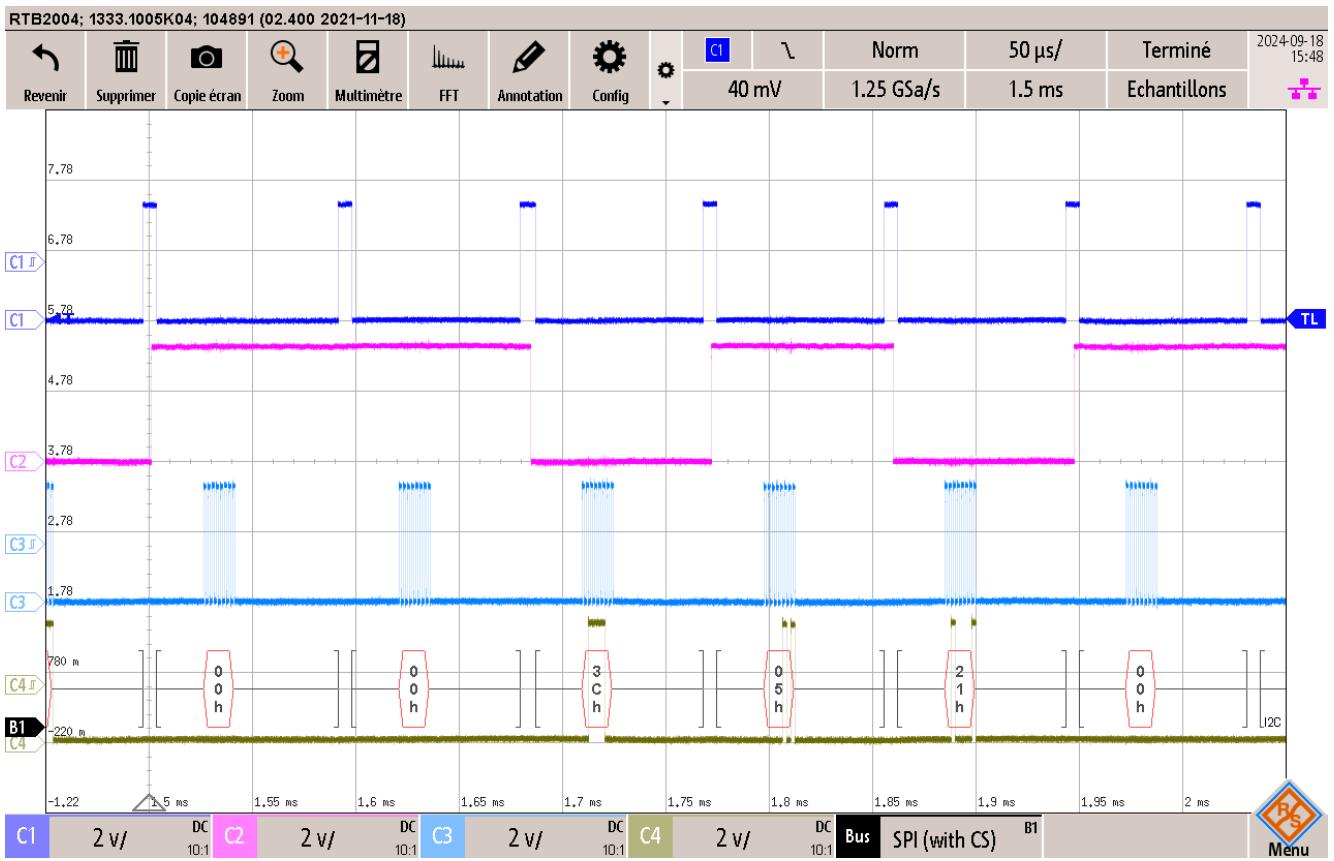


Figure 8 Quatrième trame d'initialisation avec le baudrate à 500[kBits/s]

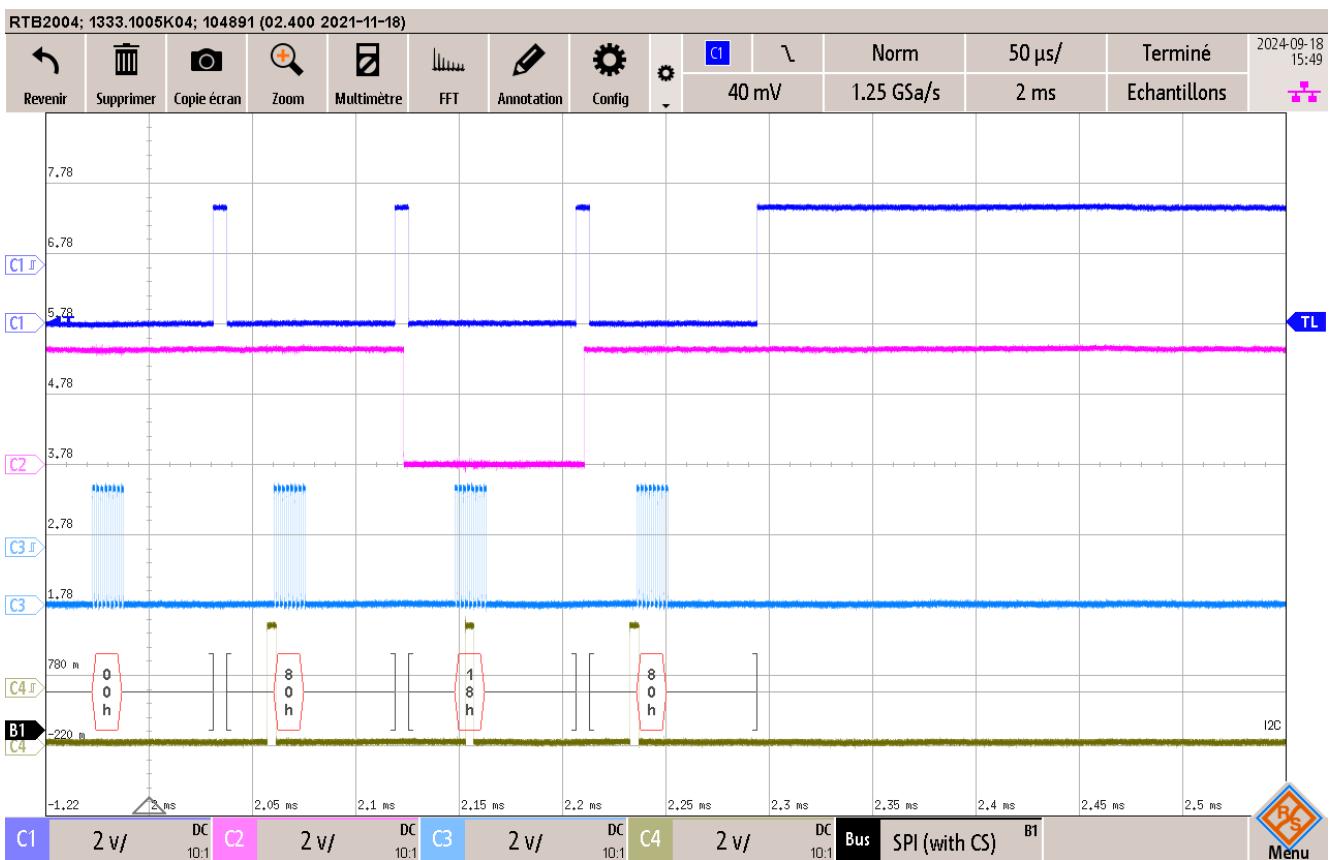


Figure 9 Dernière trame d'initialisation avec le baudrate à 500[kBits/s]

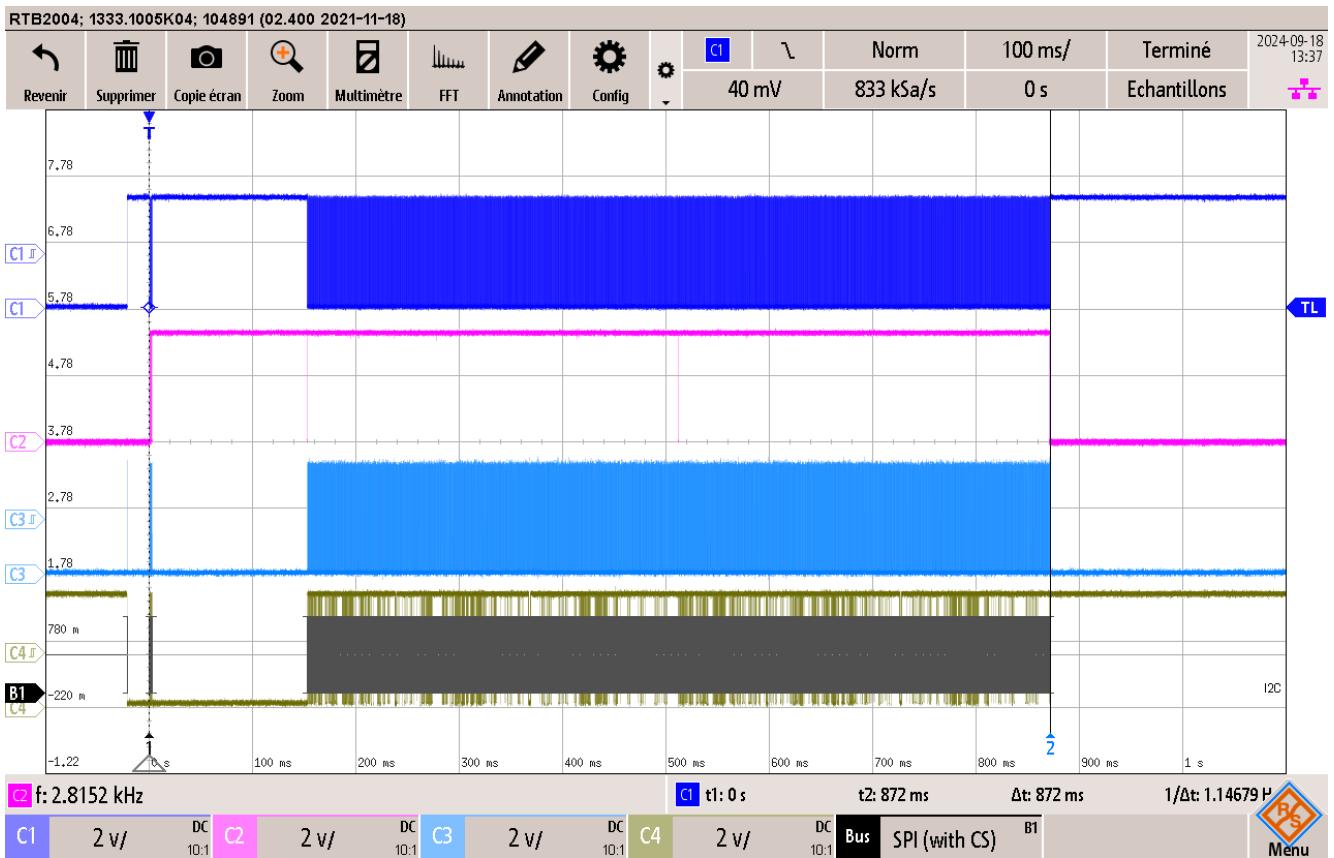


Figure 10 Durée d'une trame SPI

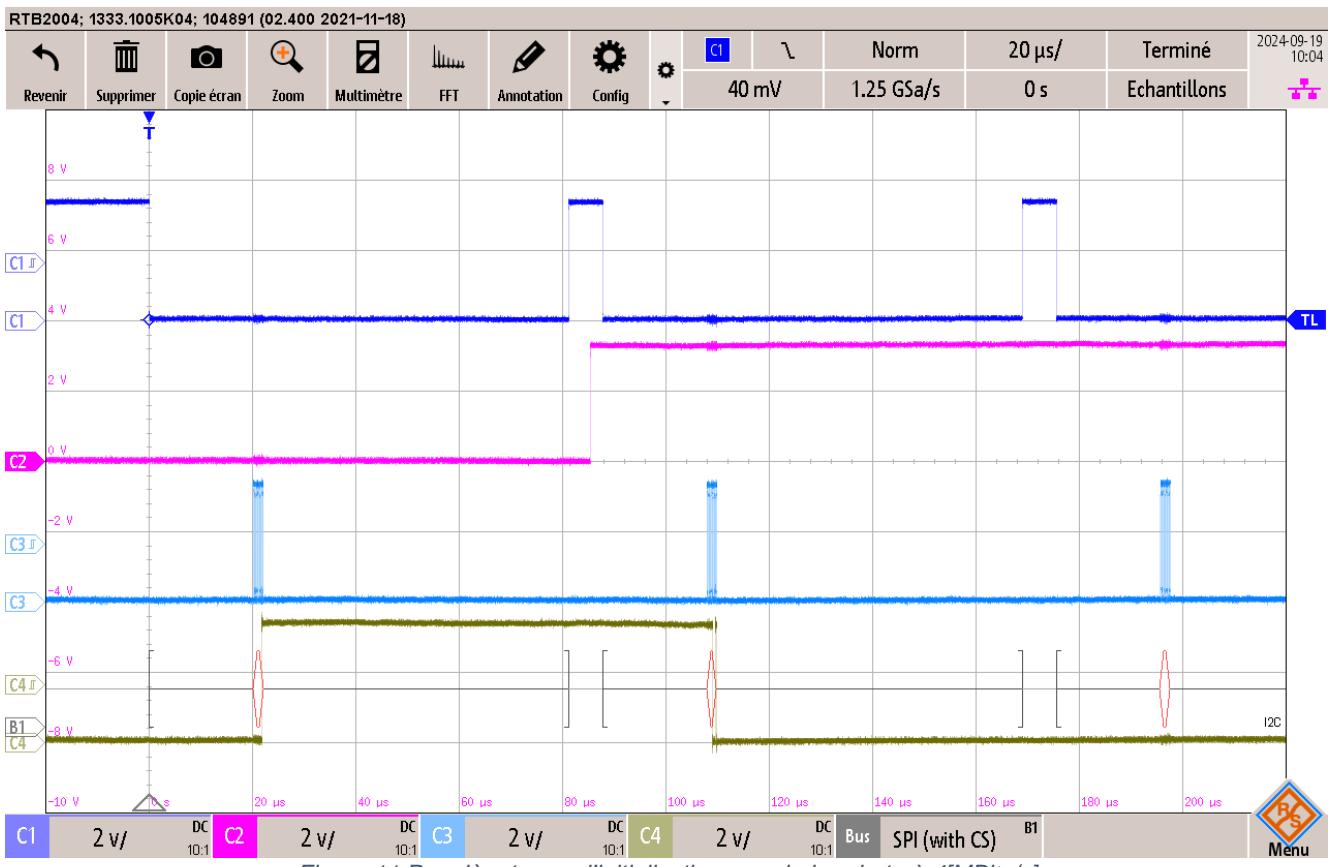


Figure 11 Première trame d'initialisation avec le baudrate à 4[MBits/s]

## Annexe M Librairies GitHub utilisées

Nom de la bibliothèque	Rôle dans le projet	Lien GitHub	Auteur	Licence
Waveshare e-Paper STM32	Gestion de l'affichage e-paper dans le projet	<a href="https://github.com/waveshareteam/e-Paper/tree/master/STM32">https://github.com/waveshareteam/e-Paper/tree/master/STM32</a>	Waveshare	Non indiquée*
Stm32-hal-libraries	Gestion capteur de température et humidité	<a href="https://github.com/belyalov/stm32-hal-libraries/tree/master">https://github.com/belyalov/stm32-hal-libraries/tree/master</a>	belyalov	MIT**

\*Remarque : WaveShare n'indique pas de licence explicite pour l'ensemble du projet dans le dépôt principal. Cependant dans la plupart des fichiers, source ou header, un texte de permission est inclus, permettant l'utilisation, la modification et la distribution du code sous certaines conditions :

```

46      * Permission is hereby granted, free of charge, to any person obtaining a copy
47      * of this software and associated documentation files (the "Software"), to deal
48      * in the Software without restriction, including without limitation the rights
49      * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
50      * copies of the Software, and to permit persons to whom the Software is
51      * furnished to do so, subject to the following conditions:
52      *
53      * The above copyright notice and this permission notice shall be included in
54      * all copies or substantial portions of the Software.
55      *

```

Figure 12 Permission WaveShare

Certains fichiers contiennent une permission différente :

```

11      * <h2><center>© 2014 STMicroelectronics</center></h2>
12      *
13      * Redistribution and use in source and binary forms, with or without modification,
14      * are permitted provided that the following conditions are met:
15      *   1. Redistributions of source code must retain the above copyright notice,
16      *      this list of conditions and the following disclaimer.
17      *   2. Redistributions in binary form must reproduce the above copyright notice,
18      *      this list of conditions and the following disclaimer in the documentation
19      *      and/or other materials provided with the distribution.
20      *   3. Neither the name of STMicroelectronics nor the names of its contributors
21      *      may be used to endorse or promote products derived from this software
22      *      without specific prior written permission.
23      *

```

Figure 13 Permission différente

\*\* La licence MIT (Massachusetts Institute of Technology License), accorde un droit illimité d'utiliser, copier, modifier et distribuer le logiciel, à condition de conserver les mentions de copyright.

## Annexe N Installations liées à Arduino IDE

Nom	Rôle dans le projet	Auteur	Ou
Esp32	Prise en charge de l'ESP32	Espressif Systems	Tools -> Board -> Manage librairies
Arduino_JSON	Manipulation et traitement des données reçues depuis le serveur	Arduino	Sketch -> Include Library-> Manage librairies
Usini_discord_webHook	Envoi de notification Discord lors d'un seuil dépassé	Usini	Sketch -> Include Library-> Manage librairies

## Projet de diplôme

---

**Technicienne ES en génie électrique,  
spécialisation électronique**

---

## Mode d'emploi 2409\_Surveillance Température- Humidité pour réfrigérateur

---

**Réalisé par :**

Mélissa Perret

**A l'attention de :**

M. Bovey  
M. Déglon  
M. Jacot-Guillarmod

**Date de création :**

22 septembre 2024

## Table des matières

1.	Mode d'emploi pour le projet principal .....	3
1.1.	Branchemet pour le fonctionnement du système .....	3
1.2.	Changement valeurs sur le site internet .....	3
1.3.	Alarme sur le site .....	4
2.	Mode d'emploi pour la mise du serveur (sur XAMPP) .....	4
3.	Configuration du routeur Cisco .....	9
4.	Paramétrer Arduino IDE .....	10

## 1. Mode d'emploi pour le projet principal

### 1.1. Branchement pour le fonctionnement du système

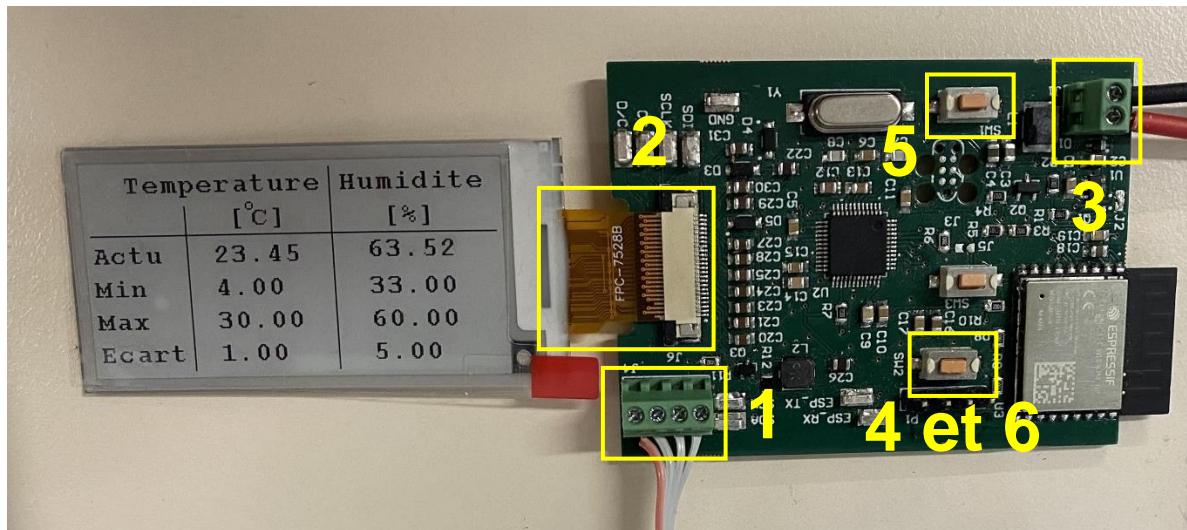


Figure 1 Fonctionnement global

Mise en marche :

1. Brancher le câble du capteur de température (le câble rouge indique le GND)
2. Brancher l'e-paper comme sur la figure 1
3. Brancher l'alimentation (piles AAA)
4. Maintenir le bouton SW2
5. Appuyer sur le bouton SW1
6. Relâcher le bouton SW1, puis SW2

Les étapes 4 à 6 sont importantes dans cet ordre (seulement lors d'un changement de piles), car cela permet au STM32 de démarrer avant l'ESP32.

### 1.2. Changement valeurs sur le site internet

1. Effectuer les étapes pour mettre en place XAMPP et activer les serveurs (suivre le point 2)
2. Une fois les étapes du point 2 réalisées, modifier l'adresse IP et le port de cet URL, puis collez le sur internet de la machine virtuelle : <http://192.168.1.102:8080/MesureTH/home.php>

Non sécurisé | 192.168.1.102:8080/MesureTH/home.php

Figure 2 Exemple de lien pour accéder au site internet

3. Normalement la page suivant doit s'afficher :

## Page pour le réglages des seuils et écarts pour le projet 2409

Seuil température min :  [°C]  
Seuil température max :  [°C]

Ecart température :  [°C]

Seuil humidité min :  [%]  
Seuil humidité max :  [%]

Ecart humidité :  [%]

Figure 3 Interface site internet

4. Modifier les valeurs, puis cliquer sur Envoyer

5. Vous devez arriver sur cette page :

Valeurs modifiées !

Figure 4 Valeurs envoyées au serveur

6. Attendre environ une seconde avant de cliquer sur le bouton retour, afin de s'assurer que les données aient bien été transmises au serveur
7. L'affichage e-paper devrait ensuite se mettre à jour (cela peut mettre un peu de temps, entre 1 et 5[s])

### 1.3. Alarme sur le site

Si la tension des piles est trop faible, le message « Pile faibles ! » apparaittra sur le site web. Tandis que si un dépassement de seuils à lieu, le message « Alarme ! » sera affiché.

Alarme !  
Piles faibles !

Figure 5 Messages d'alarme

## 2. Mode d'emploi pour la mise du serveur (sur XAMPP)

1. Installer XAMPP sur une machine virtuelle, ou réutiliser celle se trouvant dans K:\ES\PROJETS\SLO\2409\_MesureTH\_RefrigerateurCongelateur\soft
2. Lancer XAMPP, toujours en mode administrateur

3. J'ai fait en sorte que tous les appareils connectés au même Wi-Fi puissent accéder à la page web, il faut donc effectuer les modifications suivantes :

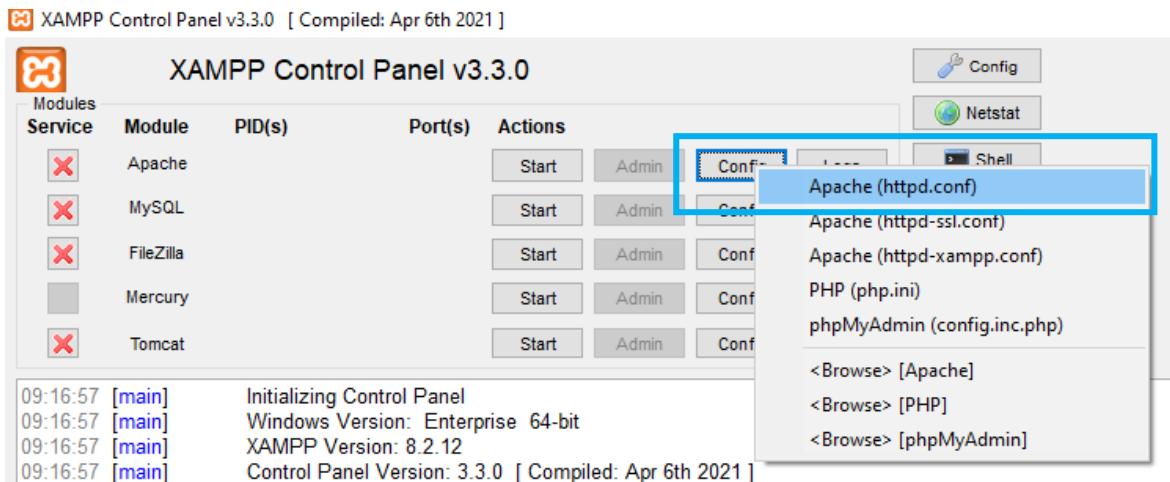


Figure 6 Interface XAMPP

4. Chercher « listen 80 » dans ce fichier, puis modifier en 8080, comme ceci :

```
#  
#Listen 12.34.56.78:80  
Listen 8080
```

Figure 7 Changement à effectuée

5. Sauvegarder le fichier  
6. Chercher l'adresse IP dans l'invite de commande Windows avec la commande « ipconfig »

```
C:\Users\ETML>ipconfig  
  
Configuration IP de Windows  
  
Carte Ethernet Ethernet0 :  
  
    Suffixe DNS propre à la connexion. . . . . : etmlnet.local  
    Adresse IPv6 de liaison locale. . . . . : fe80::7193:70c1:813:9500%12  
    Adresse IPv4. . . . . : 192.168.1.102  
    Masque de sous-réseau. . . . . : 255.255.255.0  
    Passerelle par défaut. . . . . : 192.168.1.1
```

Figure 8 Invite de commande, adresse IP

7. Retourner dans le même fichier que la figure 6  
8. Chercher « ServerName localhost :80 » et remplacer le « localhost :80 » par l'adresse IP suivie de :8080

```
ServerName 192.168.16.130:8080
```

Figure 9 Changement à effectuer

9. Sauvegarder le fichier

10. Ouvrir le fichier suivant :



Figure 10 Interface XAMPP

11. Chercher ce paragraphe :

```
Alias /phpmyadmin "C:/xampp/phpMyAdmin/"
<Directory "C:/xampp/phpMyAdmin">
    AllowOverride AuthConfig
    Require local
    ErrorDocument 403 /error/XAMPP_FORBIDDEN.html.var
</Directory>
```

Figure 11 Paragraphe à modifier

12. Remplacer le local par « all granted » comme ceci :

```
Alias /phpmyadmin "C:/xampp/phpMyAdmin/"
<Directory "C:/xampp/phpMyAdmin">
    AllowOverride AuthConfig
    Require all granted
    ErrorDocument 403 /error/XAMPP_FORBIDDEN.html.var
</Directory>
```

Figure 12 Modification effectuée

13. Sauvegarder le fichier

14. Aller ensuite dans configuration :



Figure 13 Configuration XAMPP

15. Aller dans « Service and Port Settings »

16. Changer le port par 8080 dans l'onglet Apache :

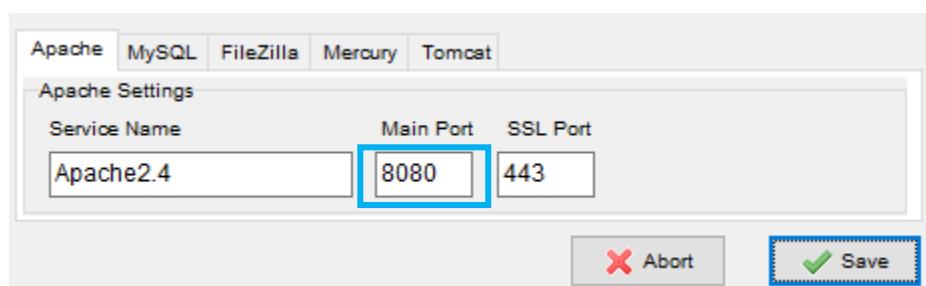
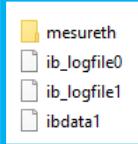


Figure 14 Changement du port dans « Service and Port Settings »

17. Sauvegarder

Si la page HTML et le serveur MySQL ne contient pas la base de données « mesureth » effectuer les étapes suivantes :

1. Dans l'interface XAMPP, cliquer sur le bouton « Explorer »
2. Aller dans le dossier « htdocs »
3. Ajouter le dossier « MesureTH » (se trouvant dans le chemin suivant : K:\ES\PROJETS\SLO\2409\_MesureTH\_RefrigerateurCongelateur\soft\Software)
4. Revenir au dossier parent
5. Aller dans le dossier « mysql », puis « data »
6. Ajouter les fichiers suivants (se trouvant également dans le chemin suivant : K:\ES\PROJETS\SLO\2409\_MesureTH\_RefrigerateurCongelateur\soft\Software) :



mesureth	08.09.2024 17:43	Dossier de fichiers
ib_logfile0	08.09.2024 17:41	Fichier 5 120 Ko
ib_logfile1	21.10.2019 14:19	Fichier 5 120 Ko
ibdata1	08.09.2024 17:41	Fichier 12 288 Ko

Figure 15 fichiers à copier dans mysql

7. Ensuite quitter puis relancer le serveur Apache et MySQL en appuyant sur Start

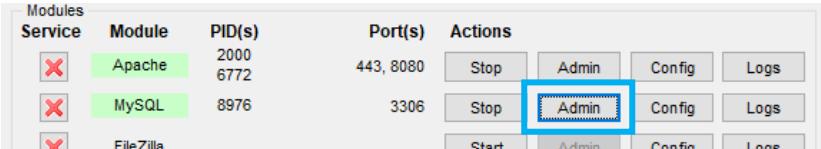
REMARQUE : il se peut que le serveur MySQL affiche le message d'erreur suivant :

```
13:58:49 [mysql] Error: MySQL shutdown unexpectedly.
13:58:49 [mysql] This may be due to a blocked port, missing dependencies,
13:58:49 [mysql] improper privileges, a crash, or a shutdown by another method.
13:58:49 [mysql] Press the Logs button to view error logs and check
13:58:49 [mysql] the Windows Event Viewer for more clues
13:58:49 [mysql] If you need more help, copy and post this
13:58:49 [mysql] entire log window on the forums
```

Figure 16 Message d'erreur

Si ce message apparaît, aller dans Explorer -> mysql -> data et supprimer le fichier commençant par *aria\_log.00000*. Ce fichier peut parfois créer des erreurs empêchant le démarrage de MySQL.

8. Pour vérifier les valeurs contenues dans la base de données :



Modules	Service	Module	PID(s)	Port(s)	Actions			
		Apache	2000 6772	443, 8080	Stop	Admin	Config	Logs
		MySQL	8976	3306	Stop	Admin	Config	Logs
		FileZilla			Start	Admin	Config	Logs

[Netstat](#)  
[Shell](#)  
[Explorer](#)  
[Services](#)

Figure 17 Interface XAMPP

9. Sur la page phpMyAdmin dans la hiérarchie vous devriez avoir : *mesureth*
10. Cliquer sur + à gauche de *mesureth* pour afficher le sous-menu déroulant
11. Cliquer sur le bouton *valeurs*

C'est dans cette page que l'on peut voir les valeurs stockées dans la base de données, qui devraient correspondre à celles affichées dans la page HTML.

A screenshot of the phpMyAdmin interface. The left sidebar shows a tree view of databases: Nouvelle base de données, information\_schema, mesureth (selected), Nouvelle table, valeurs, mysql, performance\_schema, phpmyadmin, and test. The main area shows the 'valeurs' table under the 'mesureth' database. The table has the following structure:

seuilTemperatureMin	seuilTemperatureMax	ecartTemperature	seuilHumiditeMin	seuilHumiditeMax	ecartHumidite	alarme	pilesFaibles
87	25	35	4	5	6	1	0

12. Figure 18 Valeurs stockées dans la base de données

### 3. Configuration du routeur Cisco

1. Ouvrir la machine virtuelle se trouvant dans K:\ES\PROJETS\SLO\2409\_MesureTH\_RefrigerateurCongelateur\soft
2. Rentrer sur internet l'adresse IP suivante : 192.168.1.1
3. Ensuite suivre le procédé suivant :



Figure 19 Ajout d'un SSID et d'un mot de passe

4. Sauvegarder
5. Ajouter le port 8080 ici :

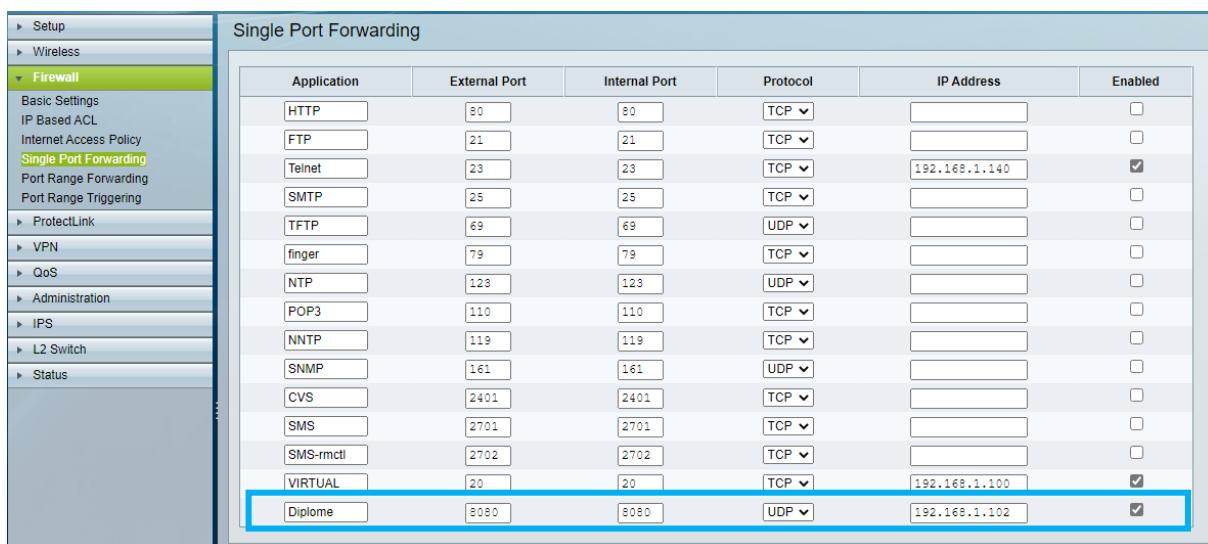


Figure 20 Ouverture du port 8080

## 4. Paramétrer Arduino IDE

1. Installer Arduino IDE
2. File -> préférences

Ajouter le lien suivant : [https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package\\_esp32\\_index.json](https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json) dans « Additional boards manager URLs », celui-ci permet d'ajouter les modules complémentaires EST dans l'IDE d'arduino

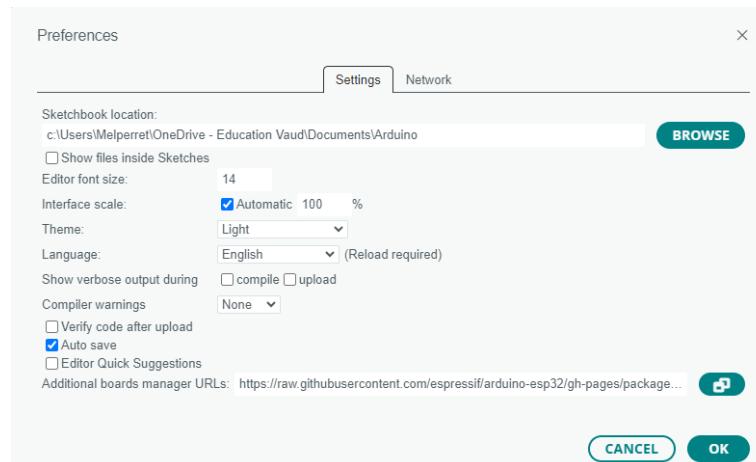


Figure 21 Paramètres Arduino

3. Installer le driver pour la carte Espressif -> Tools -> Board -> Boards Managers

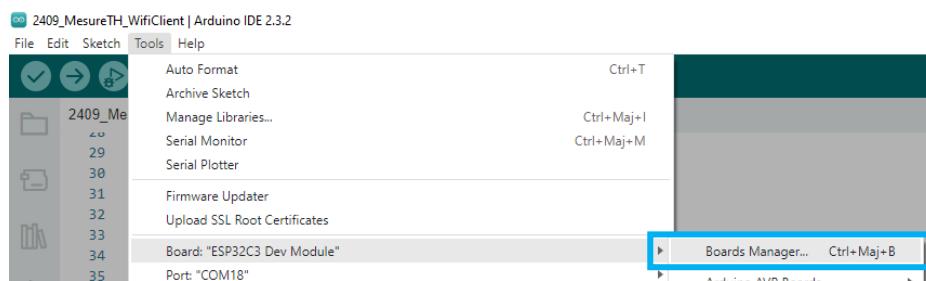


Figure 22 Installation du paquet ESP32 dans IDE Arduino

4. Installer esp32 by Espressif Systems :

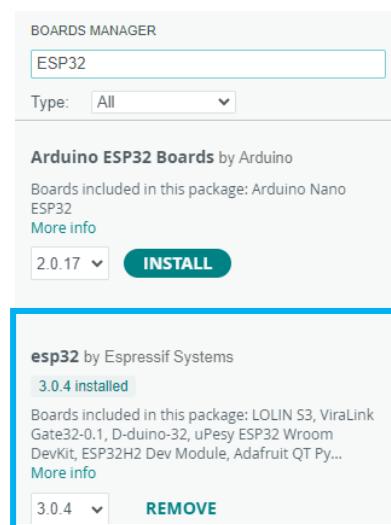


Figure 23 Installation des paquets ESP32

## 5. Installer la librairie JSON : dans Sketch -> Include library -> Manages Librairies

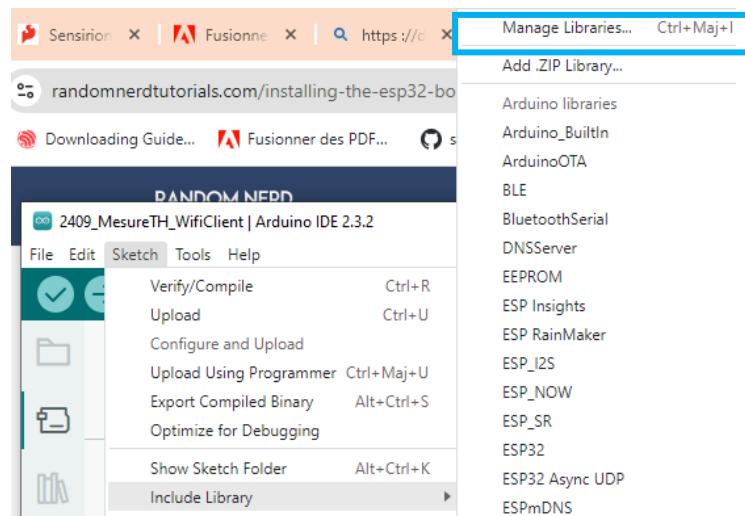


Figure 24 Installation librairie JSON

## 6. Installer la librairie Arduino\_JSON

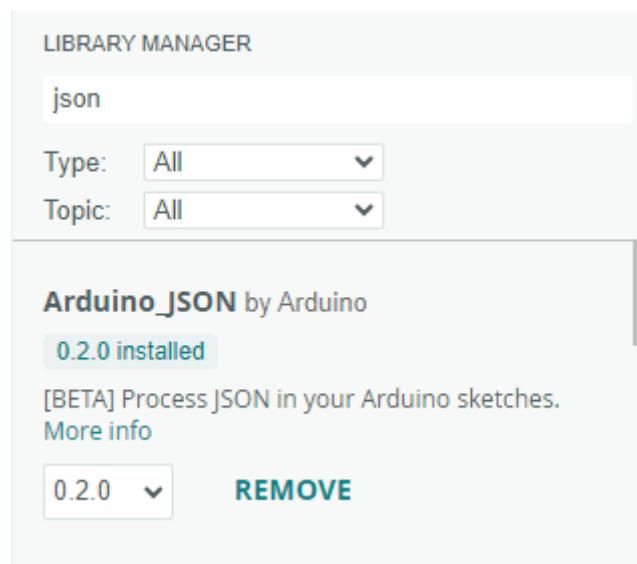


Figure 25 Librairie JSON à installer

7. Une fois cela effectué, sélectionner « ESP32C3 Dev Module » dans Tools -> Board -> esp32 -> ESP32C3 Dev Module

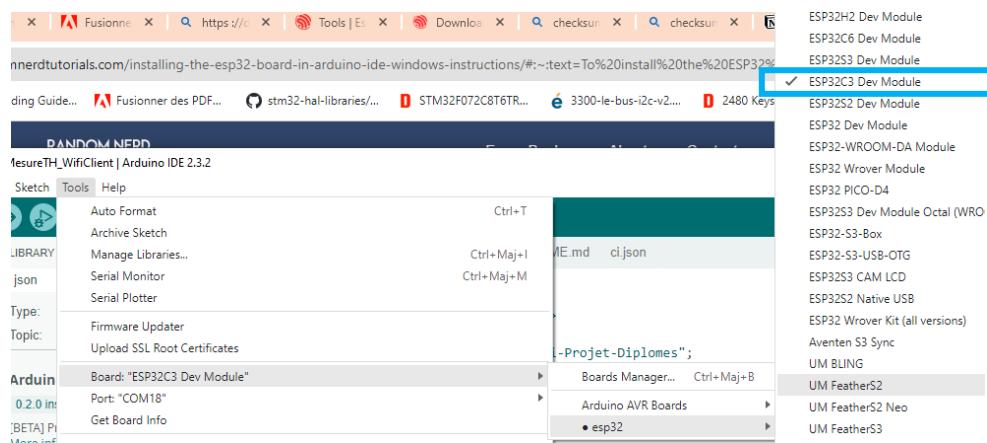


Figure 26 Sélection de la carte

8. Aller dans le gestionnaire de périphériques Windows et trouver le port COM correspondant au câble FTDI :

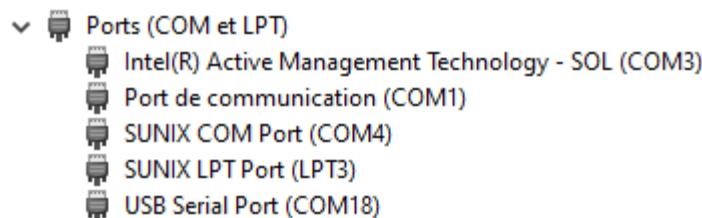


Figure 27 Liste des différents ports

9. Sélectionner le port com correspond dans Arduino IDE :

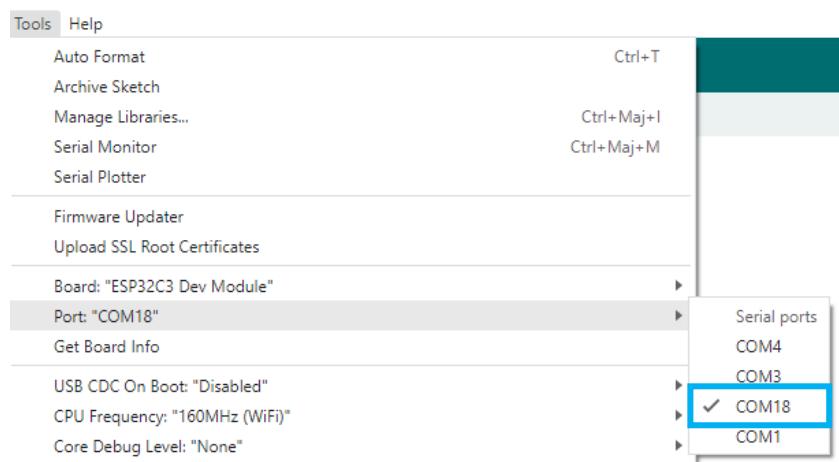


Figure 28 Sélection du port COM

10. Ensuite dans Tools, modifier les paramètres « JTAG Adapter » et « Upload Speed » :

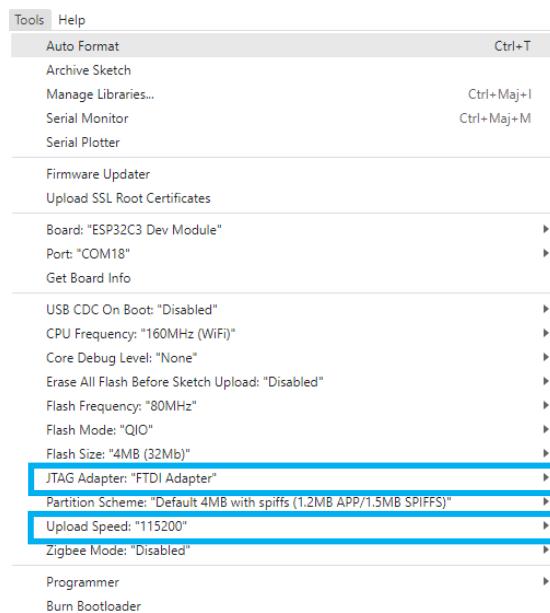


Figure 29 Changement de paramètres

11. Vérifier le code avec le bouton « ✓ »,



Figure 30 Icônes de programmation dans Arduino IDE

12. Activer le moniteur série dans Tools ->Serial Monitor

13. Passer l'ESP en mode download

- Maintenir le bouton SW3
- Appuyer sur bouton SW2
- Relâcher le bouton SW2, puis SW3

14. Si le message suivant s'affiche dans le moniteur série « waiting for download » c'est que l'ESP32 est prêt à être programmé

15. « Programmer l'ESP32 avec le bouton en forme de flèche « → »

16. Une fois l'ESP32 programmé, appuyé sur le bouton SW2 pour démarrer l'exécution

Remarque : Si l'erreur « A fatal error occurred: Failed to connect to ESP32-C3: No serial data received », cela signifie que le mode download n'a pas été activé correctement.

Lausanne, le 23 septembre 2024

Perret Mélissa

```

1 Objectif : valider le Checksum pour l'humidité
2
3 CRC (0X752A) = 0x90 = 10010000
4
5 Polynome 0x31 = 00110001 (d'après datasheet du capteur SHTC3, point 5.10, p.9)
6
7 0X752A = 01110101 00101010
8
9 Afin de valider le Checksum, il faudra retrouver la valeur 0x90
10
11 Initialisation = 0xFF = 11111111 (d'après datasheet du capteur SHTC3, point 5.10,
12 p.9)
13
14 Le calcul du CRC se fait en traitant chaque octet bit par bit.
15 A chaque étape, il faudra faire un décalage d'un bit vers la gauche.
16 Lorsque le bit MSB vaut "1", on applique un XOR avec le polynome (0x31 pour ce
17 capteur).
18 Si le MSB vaut "0", on continue simplement le décalage.
19
20 Le symbole "^" désigne un XOR
21 Rappel : Si les bits sont identiques, cela correspond à un "0"
22 Si les bits sont différents, cela correspond à un "1"
23
24 Pour commencer, il nous faut le point de départ, pour cela, on fait un XOR avec
25 l'initialisation et le premier octet obtenu :
26
27      11111111 (0xFF)
28      ^ 01110101 (0X75)
29      -----
30      10001010    -> point de départ
31
32 Comme le MSB contient un "1", on effectue l'opération XOR :
33
34 1.  00010100
35  ^  00110001 (0x31)
36  -----
37  00100101
38
39 2.  01001010    -> pas besoin de XOR, car MSB = 0
40
41 3.  10010100    -> pas besoin de XOR, car MSB = 0
42
43 4.  00101000    -> XOR, car au point 3, le MSB avait un "1" au MSB
44  ^  00110001 (0x31)
45  -----
46  00011001
47
48 5.  00110010    -> pas besoin de XOR, car MSB = 0
49
50 6.  01100100    -> pas besoin de XOR, car MSB = 0
51
52 7.  11001000    -> pas besoin de XOR, car MSB = 0
53
54 8.  10010000    -> XOR, car au point 7, le MSB avait un "1" au MSB
55  ^  00110001 (0x31)
56  -----
57  10100001
58
59 Pour le calcul du deuxième octet, le processus est le même.
60
61 Calcul pour le 2ème octet (0x2A) :
62 On reprend le résultat du premier octet et on fait un XOR avec le deuxième octet pour
63 avoir le point de départ
64
65      10100001
66      ^  00101010 (0X2A)
67  -----
68 1.  00010110
69  ^  00110001 (0x31)

```

```
70 -----  
71      00100111  
72  
73 2.  01001110  
74  
75  
76 3.  10011100  
77  
78 4.  00111000  
79 ^   00110001 (0x31)  
80 -----  
81      00001001  
82  
83 5.  00010010  
84  
85 6.  00100100  
86  
87 7.  01001000  
88  
89 8.  10010000    -> 0x90 -> Checksum OK  
90
```

```

1 Objectif : valider le Checksum pour la température
2
3 CRC (0X688A) = 0x42 = 00100010
4
5 Polynome 0x31 = 00110001 (d'après datasheet du capteur SHTC3, point 5.10, p.9)
6
7 0X688A = 01101000 10001010
8
9 Afin de valider le Checksum, il faudra retrouver la valeur 0x90
10
11 Initialisation = 0xFF = 11111111 (d'après datasheet du capteur SHTC3, point 5.10,
12 p.9)
13
14 Le calcul du CRC se fait en traitant chaque octet bit par bit.
15 A chaque étape, il faudra faire un décalage d'un bit vers la gauche.
16 Lorsque le bit MSB vaut "1", on applique un XOR avec le polynome (0x31 pour ce
17 capteur).
18 Si le MSB vaut "0", on continue simplement le décalage.
19
20 Le symbole "^" désigne un XOR
21 Rappel : Si les bits sont identiques, cela correspond à un "0"
22 Si les bits sont différents, cela correspond à un "1"
23
24 Pour commencer, il nous faut le point de départ, pour cela, on fait un XOR avec
25 l'initialisation et le premier octet obtenu :
26
27      11111111 (0xFF)
28      ^ 01101000 (0X68)
29      -----
30      10010111    -> point de départ
31
32 Comme le MSB contient un "1", on effectue l'opération XOR :
33
34 1.  00101110
35  ^  00110001 (0x31)
36  -----
37  00011111
38
39 2.  00111110    -> pas besoin de XOR, car MSB = 0
40
41 3.  01111100    -> pas besoin de XOR, car MSB = 0
42
43 4.  11111000    -> pas besoin de XOR, car MSB = 0
44
45 5.  11110000    -> XOR, car au point 7, le MSB avait un "1" au MSB
46  ^  00110001 (0x31)
47  -----
48  11000001
49
50 6.  10000010    -> XOR, car au point 7, le MSB avait un "1" au MSB
51  ^  00110001 (0x31)
52  -----
53  10110011
54
55 7.  01100110    -> XOR, car au point 7, le MSB avait un "1" au MSB
56  ^  00110001 (0x31)
57  -----
58  01010111
59
60 Pour le calcul du deuxième octet, le processus est le même.
61
62 Calcul pour le 2ème octet (0x8A) :
63 On reprend le résultat du premier octet et on fait un XOR avec le deuxième octet pour
64 avoir le point de départ
65
66      10101110
67  ^  10001010 (0X8A)
68  -----
69  00100100    -> point de départ

```

```
70
71 1. 01001000
72
73
74 2. 10010000
75
76
77 3. 00100000
78 ^ 00110001 (0x31)
79 -----
80      00010001
81
82 4. 00100010
83
84 5. 01000100
85
86 6. 10001000
87
88 7. 00010000
89 ^ 00110001 (0x31)
90 -----
91      00100001
92
93 8. 01000010    -> 0x42 -> Checksum OK
94
```

## Annexe Q Calculs utilisés pour l'estimation de consommation

Délais entre deux réveils : 60[min]  
Total heures mensuelle : 720[heures]  
Durée réveil microcontrôleur : 5[s]  
Nombre de rafraîchissement e-paper par mois : 10  
Durée rafraîchissement : 3[s]

### Calcul temps de réveil par mois en heures :

$$\frac{\text{Durée réveil microcontrôleur} \cdot 60}{\text{Délais entre deux réveils}} \cdot \frac{\text{Total heures mensuelle}}{3600}$$

### Calcul temps de sommeil par mois en heures :

$$\text{Total heures mensuelle} - \text{temps de réveil}$$

### Consommation moyenne en mA :

$$\frac{(\text{Conso mode sommeil du composant} \cdot \text{Temps de sommeil par mois}) + (\text{Conso mode réveil du composant} \cdot \text{Temps de réveil par mois})}{\text{Délais entre deux réveils}}$$

### Autonomie :

$$\frac{\text{Capacité en mA}}{\text{Conso moyenne en mA} \cdot 24}$$

## Annexe R Flash ESP32 pour AT commandes

### 1. Instructions pour flash l'ESP32 pour les AT commandes

#### 1.1 Téléchargement des drivers

1. Télécharger le programme ESP-AT<sup>1</sup> (« v3.3.0.0 [ESP32-C3-MINI-1-AT-V3.3.0.0.zip](https://docs.espressif.com/projects/esp-at/en/latest/esp32c3/AT_Binary_Lists/esp_at_binaries.html) » ou la version recommandée), puis le décompresser
2. Télécharger l'outil de programmation :

Flash Download Tools					<a href="#">Expand all</a>	<a href="#"></a> Download selected
Title	Platform	Version	Release Date	Download		
Flash Download Tools	Windows PC	V3.9.7	2024.06.07	<a href="#"></a>		

Figure 1 Logiciel de programmation<sup>2</sup>

#### 1.2 Schéma de programmation

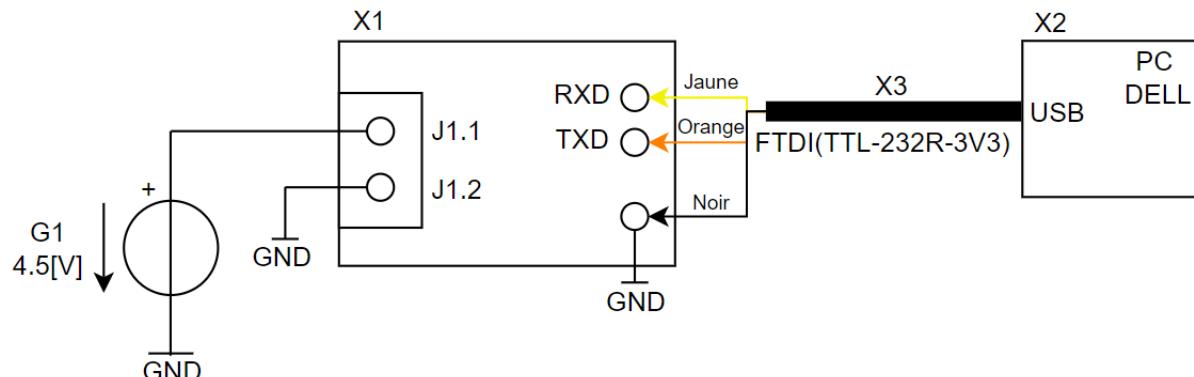


Figure 2 Schéma pour programmer l'ESP32

#### 1.3 ESP en mode téléchargement

1. Brancher la carte selon le schéma de la figure 2
2. Ouvrir le gestionnaire de périphériques Windows
3. Récupérer le nom du port COM
4. Ouvrir PUTTY

<sup>1</sup> Espressif. « Released Firmware », [en ligne], (consulté le 22 septembre 2024), master(latest), , ESP32-C3-MINI-1 Series. URL : [https://docs.espressif.com/projects/esp-at/en/latest/esp32c3/AT\\_Binary\\_Lists/esp\\_at\\_binaries.html](https://docs.espressif.com/projects/esp-at/en/latest/esp32c3/AT_Binary_Lists/esp_at_binaries.html)

<sup>2</sup> Espressif. « Download », [en ligne], (consulté le 22 septembre 2024), master(latest), Flash Download Tools. URL: <https://www.espressif.com/en/support/download/other-tools>

5. Dans PuTTY, sous Connection -> Serial, configurer les paramètres suivants :

- COM18 ou celui relevé dans le gestionnaire de périphérique
- Speed: 115200
- Data bits: 8
- Stop bits: 1
- Parity: None
- Flow Control: XON/XOFF

6. Cliquer sur le bouton Open

7. Maintenir le bouton SW3

8. Appuyer sur le bouton SW2

9. Le message suivant devrait apparaître :

```
ESP-ROM:esp32c3-apil-20210207
Build:Feb 7 2021
rst:0xl (POWERON),boot:0x7 (DOWNLOAD(USB/UART0/1))
waiting for download
```

*Figure 3 Message indiquant que l'ESP32 est passé en mode download*

10. Fermer PuTTY

#### 1.4 Programmation de l'ESP32

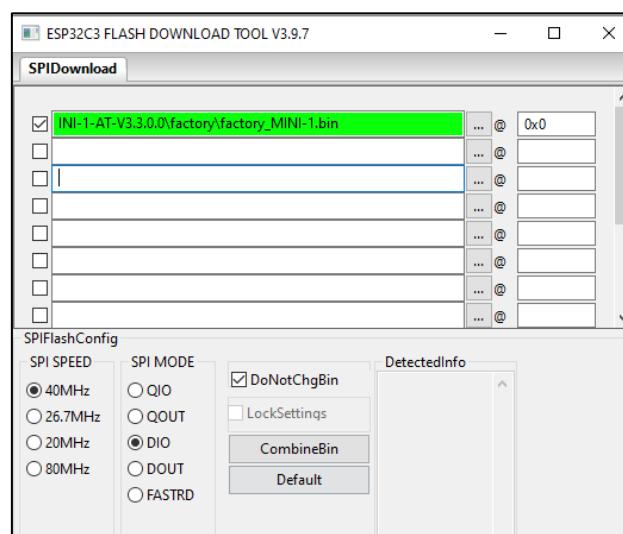
1. Ouvrir l'outil de programmation téléchargé précédemment (figure 1)

2. Sélectionner dans la configuration :

- ChipType: ESP32-C3
- WorkMode: Develop
- LoadMode: UART

3. Appuyer sur le bon OK

4. Une fois ouvert, sélectionner la configuration suivante :



*Figure 4 Configuration du logiciel*

3. Cocher la première ligne comme la figure 4, puis dans les « ... », ajouter le fichier nommé « factory\_Mini-1.bin » dans le dossier « factory » du dossier « ESP32-C3-MINI-1-AT-V3.3.0.0 »
4. Mettre la valeur à 0x0 dans la colonne de droite
5. Cocher « DoNotChgBin »
6. Mettre le port COM
7. Mettre 115200 dans BAUD
8. Appuyer sur START
9. Attendre l'affichage suivant :

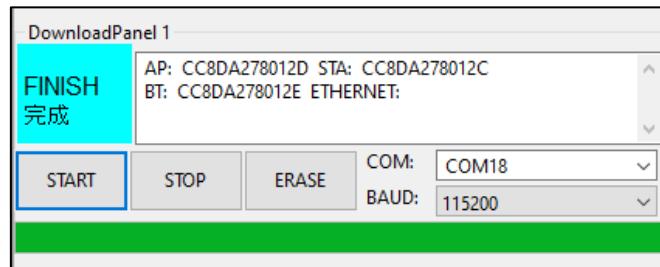


Figure 5 fin de programmation

## 1.5 Vérification de la programmation

1. Ouvrir PuTTY et régler les mêmes paramètres que précédemment (voir point 1.3, point 5)
2. Cliquer sur Open
3. Appuyer sur le bouton SW2
4. Si l'ESP32 est bien programmé, le message suivant apparaît :

```
COM18 - PuTTY
ESP-RO:esp32c3-april-20210207
Build:Feb 7 2021
rst:0x1 (POWERON),boot:0xf (SPI_FAST_FLASH_BOOT)
SPIWP:0xee
mode:DIO, clock div:2
load:0x3fc5820, len:0x16a8
load:0x403cc710, len:0x93c
load:0x403ce710, len:0x2eb8
entry 0x403cc710
I (31) boot: ESP-IDF v5.0.6-dirty 2nd stage bootloader
I (31) boot: compile time 08:51:32
I (31) boot: chip revision: v0.4
I (34) boot.esp32c3: SPI Speed      : 40MHz
I (39) boot.esp32c3: SPI Mode       : DIO
I (43) boot.esp32c3: SPI Flash Size : 4MB
I (48) boot: Enabling RNG early entropy source...
I (54) boot: Partition Table:
I (57) boot: ## Label           Usage         Type ST Offset  Length
I (64) boot: 0 ota_data        OTA data     01 00 0000d000 00002000
I (72) boot: 1 phy_init        RF data      01 01 0000f000 00001000
I (79) boot: 2 nvs             WiFi data   01 02 00010000 0000e000
I (87) boot: 3 at_customize    unknown      40 00 0001e000 00042000
I (94) boot: 4 ota_0          OTA app     00 10 00060000 001d0000
I (102) boot: 5 ota_1          OTA app     00 11 00230000 001d0000
I (109) boot: End of partition table
I (114) boot: No factory image, trying OTA 0
I (118) esp_image: segment 0: paddr=00060020 vaddr=3c160020 size=31a40h (203328) map
I (170) esp_image: segment 1: paddr=00091a68 vaddr=3fc96200 size=0396ch ( 14700) load
I (173) esp_image: segment 2: paddr=000953dc vaddr=40380000 size=0ac3ch ( 44092) load
I (186) esp_image: segment 3: paddr=000a0020 vaddr=42000020 size=15e318h (1434352) map
I (486) esp_image: segment 4: paddr=001fe340 vaddr=4038ac3c size=0b458h ( 46168) load
I (498) esp_image: segment 5: paddr=002097a0 vaddr=50000000 size=00018h (    24) load
I (505) boot: Loaded app from partition at offset 0x60000
I (546) boot: Set actual ota_seq=1 in ota_data[0]
I (546) boot: Disabling RNG early entropy source...
no external 32k oscillator, disable it now.
at param mode: 1
AT cmd port:uart1 tx:7 rx:6 cts:5 rts:4 baudrate:115200
module_name: MINI-1
max tx power=78, ret=0
v3.3.0.0
```

Figure 6 Vérification de programmation de l'ESP

```

1 // Copyright (c) Konstantin Belyalov. All rights reserved.
2 // Licensed under the MIT license.
3
4 #ifndef __SHTC3_H
5 #define __SHTC3_H
6
7 #include "main.h"
8
9 #ifdef __cplusplus
10 #define EXPORT extern "C"
11 #else
12 #define EXPORT
13 #endif
14
15
16 // The shtc3 provides a serial number individualized for each device
17 // Params:
18 // - `hi2c` I2C bus
19 // Returns device id or 0 in case of error.
20 EXPORT uint16_t shtc3_read_id(I2C_HandleTypeDef *hi2c);
21
22 // Put sensor into sleep mode
23 // Params:
24 // - `hi2c` I2C bus
25 // Returns device id or 0 in case of error.
26 EXPORT uint32_t shtc3_sleep(I2C_HandleTypeDef *hi2c);
27
28 // Wake up sensor.
29 // You must wait for 240us to let sensor enter into IDLE mode.
30 // Params:
31 // - `hi2c` I2C bus
32 // Returns zero in case of error
33 EXPORT uint32_t shtc3_wakeup(I2C_HandleTypeDef *hi2c);
34
35 // Performs full cycle: starts temperature/humidity measurements using "clock stretch" method.
36 // Params:
37 // - `hi2c` I2C bus
38 // - `temp` measured temperature, in C multiplied by 100 (e.g. 24.1C -> 2410)
39 // - `hum` measured relative humidity, in percents
40 // Returns zero in case of error
41 EXPORT uint32_t shtc3_perform_measurements(I2C_HandleTypeDef *hi2c, int32_t* temp, int32_t* hum);
42
43 // Start temperature/humidity measurements using "clock stretch" approach, in low power mode.
44 // After completed - values can be obtained by shtc3_read_measurements()
45 // Params:
46 // - `hi2c` I2C bus
47 // - `temp` measured temperature, in C multiplied by 100 (e.g. 24.1C -> 2410)
48 // - `hum` measured relative humidity, in percents
49 // Returns zero in case of error
50 EXPORT uint32_t shtc3_perform_measurements_low_power(I2C_HandleTypeDef *hi2c, int32_t* out_temp,
int32_t* out_hum);
51
52 #endif
53

```

## Annexe S.1

## Annexe S.2

```
1 #include "shtc3.h"
2
3 #define SHTC3_ADDRESS_READ          (0x70 << 1) | 0x01
4 #define SHTC3_ADDRESS_WRITE         (0x70 << 1)
5
6 #define SHTC3_PRODUCT_CODE_MASK    0x083F
7 #define SHTC3_SENSOR_ID_MASK       0xF7C0
8
9 // NOTE: all commands are "byte swapped" (ntohs), meaning:
10 // 0x3517 -> 0x1735
11
12 #define SHTC3_CMD_WAKEUP           0x1735
13 #define SHTC3_CMD_SLEEP             0x98B0
14 #define SHTC3_CMD_SOFT_RESET        0x5D80
15 #define SHTC3_CMD_READ_ID           0xC8EF
16
17 // Clock stretching based commands
18 #define SHTC3_CMD_CLK_STRETCH_READ_HUM_FIRST      0x245C
19 #define SHTC3_CMD_CLK_STRETCH_READ_HUM_FIRST_LOW_POWER 0xDE44
20
21 // Polling commands
22 #define SHTC3_CMD_POLL_HUM_FIRST           0xE058
23 #define SHTC3_CMD_POLL_HUM_FIRST_LOW_POWER 0x1A40
24
25
26 uint16_t shtc3_read_id(I2C_HandleTypeDef *hi2c)
27 {
28     uint8_t data[2];
29     uint16_t command = SHTC3_CMD_READ_ID;
30
31     uint32_t res = HAL_I2C_Master_Transmit(hi2c, SHTC3_ADDRESS_WRITE, (uint8_t*)&command, 2, 100);
32     if (res != HAL_OK) {
33         return 0;
34     }
35     res = HAL_I2C_Master_Receive(hi2c, SHTC3_ADDRESS_READ, (uint8_t*)data, 2, 100);
36     if (res != HAL_OK) {
37         return 0;
38     }
39
40     // SHTC3 16 bit ID encoded as:
41     // xxxx xxxx xx00 0111
42     // where "x" are actual, sensor ID, while the rest
43     // sensor product code (unchangeable)
44     uint16_t id = data[0] << 8 | data[1];
45     uint16_t code = id & SHTC3_PRODUCT_CODE_MASK;
46     if (code == 0x807) {
47         // Sensor preset, return actual ID
48         return id & SHTC3_SENSOR_ID_MASK;
49     }
50
51     return 0;
52 }
53
54 uint32_t shtc3_sleep(I2C_HandleTypeDef *hi2c)
55 {
56     uint16_t command = SHTC3_CMD_SLEEP;
57     uint32_t res = HAL_I2C_Master_Transmit(hi2c, SHTC3_ADDRESS_WRITE, (uint8_t*)&command, 2, 100);
58
59     return res == HAL_OK;
60 }
61
62 uint32_t shtc3_wakeup(I2C_HandleTypeDef *hi2c)
63 {
64     uint16_t command = SHTC3_CMD_WAKEUP;
65     uint32_t res = HAL_I2C_Master_Transmit(hi2c, SHTC3_ADDRESS_WRITE, (uint8_t*)&command, 2, 100);
66
67     return res == HAL_OK;
68 }
69
70
71 static uint32_t checkCRC(uint16_t value, uint8_t expected)
72 {
73     uint8_t data[2] = {value >> 8, value & 0xFF};
74     uint8_t crc = 0xFF;
75     uint8_t poly = 0x31;
76
77     for (uint8_t indi = 0; indi < 2; indi++) {
```

```

78     crc ^= data[indi];
79     for (uint8_t indj = 0; indj < 8; indj++) {
80         if (crc & 0x80) {
81             crc = (uint8_t)((crc << 1) ^ poly);
82         } else {
83             crc <= 1;
84         }
85     }
86 }
87
88 if (expected ^ crc) {
89     return 0;
90 }
91 return 1;
92 }
93
94 static uint32_t _read_values(uint8_t* data, int32_t* out_temp, int32_t* out_hum)
95 {
96     // Check CRC
97     uint32_t raw_hum = data[0] << 8 | data[1];
98     uint32_t raw_temp = data[3] << 8 | data[4];
99
100    if (!checkCRC(raw_hum, data[2])) {
101        return 0;
102    }
103    if (!checkCRC(raw_temp, data[5])) {
104        return 0;
105    }
106
107    // Convert values
108    if (out_hum) {
109        *out_hum = raw_hum * 10000 / 65535;
110    }
111    if (out_temp) {
112        *out_temp = raw_temp * 17500 / 65535 - 4500;
113    }
114
115    return 1;
116 }
117
118 static uint32_t _perform_measurements(I2C_HandleTypeDef *hi2c, uint16_t command, int32_t* out_temp,
119 int32_t* out_hum)
120 {
121     uint8_t result[6];
122
123     uint32_t res = HAL_I2C_Master_Transmit(hi2c, SHTC3_ADDRESS_WRITE, (uint8_t*)&command, 2, 100);
124     if (res != HAL_OK) {
125         return 0;
126     }
127
128     res = HAL_I2C_Master_Receive(hi2c, SHTC3_ADDRESS_READ, result, 6, 100);
129     if (res != HAL_OK) {
130         return 0;
131     }
132
133     return _read_values(result, out_temp, out_hum);
134 }
135
136 uint32_t shtc3_perform_measurements(I2C_HandleTypeDef *hi2c, int32_t* out_temp, int32_t* out_hum)
137 {
138     return _perform_measurements(hi2c, SHTC3_CMD_CLK_STRETCH_READ_HUM_FIRST, out_temp, out_hum);
139 }
140
141 uint32_t shtc3_perform_measurements_low_power(I2C_HandleTypeDef *hi2c, int32_t* out_temp, int32_t*
142 out_hum)
143 {
144     return _perform_measurements(hi2c, SHTC3_CMD_CLK_STRETCH_READ_HUM_FIRST_LOW_POWER, out_temp,
145 out_hum);
146 }
147

```

```

1 // affichage.c
2 //
3 // Description : fonctions liée à l'affichage E-paper
4 // Auteur : Perret Mélissa
5 // Création : 16/09/2024
6 // Modifications : --
7
8 // Version : V1.0
9 /*-----*/
10
11
12 #include "affichage.h"
13 #include "EPD_2in13_V4.h" // pour EPD_2in13_V4_Init etc.
14 #include "GUI_Paint.h" // pour les fonctions Paint (Paint_NewImage, Paint_DrawString_EN etc.)
15 #include <stdlib.h> // pour malloc et free
16
17
18 /////////////////////////////////////////////////////////////////// Fonction AffichageEPaper (affichage des données sur l'écran E-paper)
19 /////////////////////////////////////////////////////////////////// Description: affiche les données sur l'écran E-paper (valeurs provenant du serveur, valeurs mesurées avec la sonde)
20 ////////////////////////////////////////////////////////////////// Entrées: Pointeur:Mesures mesures (données mesurées), Tableau:DefinitionValeur valeursServeur (valeurs provenant du serveur)
21 ////////////////////////////////////////////////////////////////// Sorties: uint16_t (-1 si problème, 0 si affiche réussi)
22 int AffichageEPaper(Mesures* mesures, DefinitionValeur valeursServeur[])
23 {
24     // Initialisation du module de développement
25     if(DEV_Module_Init() != 0)
26     {
27         return -1; // Initialisation échouée
28     }
29
30     // Créer une nouvelle image pour le cache
31     UBYTE *Image;
32     UWORD Imagesize = ((EPD_2in13_V4_WIDTH % 8 == 0)? (EPD_2in13_V4_WIDTH / 8) : (EPD_2in13_V4_WIDTH / 8 + 1)) * EPD_2in13_V4_HEIGHT; //instruction ternaire ?
33     if((Image = (UBYTE *)malloc(Imagesize)) == NULL)
34     {
35         return -1; // Allocation impossible. Problème de HEAP_SIZE insuffisante ?
36     }
37
38     // Mettre à jour les données considérées comme affichées
39     mesures->temperatureAffichee = mesures->temperatureActuelle;
40     mesures->humiditeAffichee = mesures->humiditeActuelle;
41
42     EPD_2in13_V4_Init(); // Initialisiation E-paper
43
44     // Initialisation de l'image avec des pixels blancs
45     Paint_NewImage(Image, EPD_2in13_V4_WIDTH, EPD_2in13_V4_HEIGHT, 90, WHITE);
46     Paint_SelectImage(Image);
47     Paint_Clear(WHITE);
48
49     // Première colonne
50     Paint_DrawString_EN(0, 45, "Actu", &Font16, WHITE, BLACK);
51     Paint_DrawString_EN(0, 65, "Min", &Font16, WHITE, BLACK);
52     Paint_DrawString_EN(0, 85, "Max", &Font16, WHITE, BLACK);

```

## Annexe S.3

```
53     Paint_DrawString_EN(0, 105, "Ecart", &Font16, WHITE, BLACK);
54
55     Paint_DrawLine(60, 23, 60, 122, BLACK, DOT_PIXEL_1X1, LINE_STYLE_SOLID); // Ligne verticale pour séparer les colonnes
56
57     // Seconde colonne (température)
58     Paint_DrawString_EN(23, 3, "Temperature", &Font16, WHITE, BLACK); // Titre colonne
59     Paint_DrawString_EN(85, 23, "[", &Font16, WHITE, BLACK); // Unité colonne
60     Paint_DrawCircle(100, 23, 2, BLACK, DOT_PIXEL_1X1, DRAW_FILL_EMPTY); // Unité colonne (symbole degré °)
61     Paint_DrawString_EN(100, 23, "C]", &Font16, WHITE, BLACK); // Unité colonne
62
63     Paint_DrawNumDecimals(75, 45, mesures->temperatureActuelle, &Font16, NOMBRE_DECIMALES_MESURES, BLACK, WHITE); // Température actuelle
64     Paint_DrawNumDecimals(75, 65, valeursServeur[SEUIL_TEMPERATURE_MIN].valeur, &Font16, NOMBRE_DECIMALES_MESURES, BLACK, WHITE); // Température
min
65     Paint_DrawNumDecimals(75, 85, valeursServeur[SEUIL_TEMPERATURE_MAX].valeur, &Font16, NOMBRE_DECIMALES_MESURES, BLACK, WHITE); // Température max
66     Paint_DrawNumDecimals(75, 105, valeursServeur[ECART_TEMPERATURE].valeur, &Font16, NOMBRE_DECIMALES_MESURES, BLACK, WHITE); // Ecart température
67
68     Paint_DrawLine(150, 0, 150, 122, BLACK, DOT_PIXEL_1X1, LINE_STYLE_SOLID); // Ligne verticale pour séparer les colonnes
69
70     // Troisième colonne (humidité)
71     Paint_DrawString_EN(155, 3, "Humidite", &Font16, WHITE, BLACK); // Titre colonne
72     Paint_DrawString_EN(180, 23, "[%]", &Font16, WHITE, BLACK); // Unité colonne
73     Paint_DrawNumDecimals(170, 43, mesures->humiditeActuelle, &Font16, NOMBRE_DECIMALES_MESURES, BLACK, WHITE); // Humidité actuelle
74     Paint_DrawNumDecimals(170, 65, valeursServeur[SEUIL_HUMIDITE_MIN].valeur, &Font16, NOMBRE_DECIMALES_MESURES, BLACK, WHITE); // Humidité min
75     Paint_DrawNumDecimals(170, 85, valeursServeur[SEUIL_HUMIDITE_MAX].valeur, &Font16, NOMBRE_DECIMALES_MESURES, BLACK, WHITE); // Humidité Max
76     Paint_DrawNumDecimals(170, 105, valeursServeur[ECART_HUMIDITE].valeur, &Font16, NOMBRE_DECIMALES_MESURES, BLACK, WHITE); // Ecart humidité
77
78     Paint_DrawLine(0, 40, 250, 40, BLACK, DOT_PIXEL_1X1, LINE_STYLE_SOLID); // Ligne horizontale (séparation)
79
80     EPD_2in13_V4_Display_Base(Image); // Affichage
81
82     // Libérer la mémoire allouée pour l'image
83     free(Image);
84     Image = NULL;
85
86     DEV_Module_Exit(); // Fermeture
87
88     return 0;
89 }
90 }
```

# Annexe S.4

```
1  /* **** */
2  /** Descriptive File Name
3
4  @Company
5      ETML-ES
6
7  @File Name
8      affichage.h
9
10 @Auteurs
11     - Perret Mélissa
12
13 @Description
14     Fonctions liée à l'affichage E-paper
15 */
16 /* **** */
17
18
19
20 #ifndef _AFFICHAGE_H_
21 #define _AFFICHAGE_H_
22
23
24 /* **** */
25 /* **** */
26 /* Section: Included Files */
27 /* **** */
28 /* **** */
29
30 /* This section lists the other files that are included in this file.
31 */
32
33 #include "mesures.h" // pour Mesures
34
35
36 #ifdef __cplusplus
37 extern "C" {
38 #endif
39
40 // ****
41 // ****
42 // Section: Global Data
43 // ****
44 // ****
45
46 /* **** */
47 /* **** */
48 /* Section: Constants */
49 /* **** */
50 /* **** */
51
52 #define NOMBRE_DECIMALES_MESURES 2 // Nombre de décimales affichées sur l'écran pour les valeurs
liées au mesure
53
54
55 // ****
56 // ****
57 // Section: Prototypes
58 // ****
59 // ****
60 //-----
```

---

```
62 int AffichageEPaper(Mesures* mesures, DefinitionValeur valeursServeur[]);
63
64 #ifdef __cplusplus
65 }
66 #endif
68 #endif /* _AFFICHAGE_H_ */
```

# Annexe S.5

```

1 //  gestionBatterie.c
2 //
3 //  Description : fonctions liées à la gestion de la batterie
4 //  Auteur : Perret Mélissa
5 //  Création : 09/06/2024
6 //  Modifications : --
7
8 //  Version    : V1.0
9 /*-----*/
10
11
12 #include "main.h"
13 #include "stdbool.h"
14 #include "fonctionsADC.h"
15 #include "gestionBatterie.h"
16
17
18 /////////////////////////////////////////////////////////////////// Fonction GestionCheckBatterie (regarder si une nouvelle vérification de la batterie est nécessaire)
19 /////////////////////////////////////////////////////////////////// Description: décrémente le compteur lié à la vérification de la batterie, déclenche le lancement de la vérification batterie si besoin
20 ////////////////////////////////////////////////////////////////// Entrées: Pointeur:ADC_HandleTypeDef hadc, Pointeur:TIM_HandleTypeDef htim, Pointeur:InfoBatterie infoBatterie
21 ////////////////////////////////////////////////////////////////// Sorties: bool (true si besoin de mesurer l'état de la batterie)
22 bool GestionCheckBatterie(ADC_HandleTypeDef *hadc, TIM_HandleTypeDef *htim, InfoBatterie *infoBatterie)
23 {
24     if(infoBatterie->compteurCheckBatterie > 0)
25     {
26         infoBatterie->compteurCheckBatterie--; // Décrementer le compteur à chaque réveil
27     }
28
29     if(infoBatterie->compteurCheckBatterie == 0)
30     {
31         // Si le compteur vaut 0, il est temps de procéder à une nouvelle vérification de l'état de la batterie
32         LancerCheckBatterie(htim, infoBatterie);
33         return true;
34     }
35     else
36     {
37         return false; // Pas de vérification batterie nécessaire pour le moment
38     }
39 }
40
41 ////////////////////////////////////////////////////////////////// Fonction MesurerEtatBatterie (détermine l'état de la batterie)
42 ////////////////////////////////////////////////////////////////// Description: utilise ControleEtatChargeBatterie pour récupérer l'état de la batterie et ArreterLectureTensionBatterie pour terminer
43 ////////////////////////////////////////////////////////////////// l'opération
44 ////////////////////////////////////////////////////////////////// Entrées: Pointeur:ADC_HandleTypeDef hadc, Pointeur:TIM_HandleTypeDef htim, Pointeur:InfoBatterie infoBatterie
45 ////////////////////////////////////////////////////////////////// Sorties: Etat (état de la batterie)
46 Etat MesurerEtatBatterie(ADC_HandleTypeDef *hadc, TIM_HandleTypeDef *htim, InfoBatterie *infoBatterie)
47 {
48     Etat nouvelEtatBatterie = ControleEtatChargeBatterie(hadc, infoBatterie);
49     ArreterLectureTensionBatterie(htim, infoBatterie);
50     return nouvelEtatBatterie;
51 }
52

```

```

53 ///////////////////////////////////////////////////////////////////
54 // Fonction LancerCheckBatterie (étape préliminaire avant de pouvoir mesurer l'état de la batterie)
55 // Description: enclenche le transitor nécessaire pour pouvoir lire l'état de la batterie, met en place le timer pour le délai avant mesure
56 // Entrées: Pointeur:TIM_HandleTypeDef htim, Pointeur:InfoBatterie infoBatterie
57 // Sorties: -
58 void LancerCheckBatterie(TIM_HandleTypeDef* htim, InfoBatterie* infoBatterie)
59 {
60     // Enclenche le transistor nécessaire pour pouvoir lire l'état de la batterie
61     HAL_GPIO_WritePin(EN_VBAT_GPIO_Port, EN_VBAT_Pin, GPIO_PIN_SET);
62     // Activation de l'interruption timer pour pouvoir attendre un certain délai (DELAI_ACTIVATION_TRANSISTOR_MS) avant de lire l'état de la batterie
63     HAL_TIM_Base_Start_IT(htim);
64     // Compteur cadencé sur l'interruption timer
65     // Nombre de fois que l'on doit décrémenter le compteur dans l'interruption timer avant de procéder à la mesure
66     infoBatterie->compteurCheckBatterie = DELAI_ACTIVATION_TRANSISTOR_MS / DELAI_TIMER6_MS;
67 }
68
69 ///////////////////////////////////////////////////////////////////
70 // Fonction ControleEtatChargeBatterie
71 ///////////////////////////////////////////////////////////////////
72 // Description: Contrôle l'état de charge de la batterie (allume une LED en cas de problème détecté)
73 // Entrées: Pointeur:ADC_HandleTypeDef hadc, Pointeur:InfoBatterie infoBatterie
74 // Sorties: Etat (état de la batterie)
75 Etat ControleEtatChargeBatterie(ADC_HandleTypeDef* hadc, InfoBatterie* infoBatterie)
76 {
77     float valeurBruteADC_VBAT = 0;
78     float valeurVBat = 0;
79
80     // Lecture de valeur du ADC et sauvegarde
81     valeurBruteADC_VBAT = LectureValeurAdcBrute(hadc, ADC_CHANNEL_6);
82
83     // Convertie la valeur brute de l'ADC en volt
84     valeurVBat = ConversionValeurAdcEnVolt(valeurBruteADC_VBAT, R3_THEORIQUE, R6_THEORIQUE);
85
86     Etat nouvelEtatBatterie = INDEFINI;
87
88     // Enbranchement en fonction de la tension des piles
89     // Pour changer la tension de la charge, modifier la valeur de VALEUR_BATTERIE_VOLT dans gestionBatterie.h
90     if (VALEUR_BATTERIE_VOLT == VALEUR_3PILES_VOLT)
91     {
92         if((valeurVBat > VALEUR_BATTERIE_4_5V_MAX_VOLT ) || (valeurVBat < VALEUR_BATTERIE_4_5V_MIN_VOLT ))
93         {
94             // Problème détecté (mauvaise tension de batterie ou batterie déchargée)
95             HAL_GPIO_WritePin(LED_RED_GPIO_Port, LED_RED_Pin, GPIO_PIN_SET); // Allume la LED rouge
96             nouvelEtatBatterie = ALARME;
97         }
98     else
99     {
100        // Aucun problème (batterie avec la bonne tension et chargée)
101        HAL_GPIO_WritePin(LED_RED_GPIO_Port, LED_RED_Pin, GPIO_PIN_RESET); // Eteint la LED rouge
102        nouvelEtatBatterie = OK;
103    }
104 }
105

```

```

C:\ES\PROJETS\SLO\2409_MesureTH_RefrigerateurCongelateur\soft\Firmware\STM32\2409_MesureTH_V1\2409_MesureTH_V1\ControlePiles\gestionBatterie.c
106     if((valeurVBat > VALEUR_BATTERIE_6V_MAX_VOLT ) || (valeurVBat < VALEUR_BATTERIE_6V_MIN_VOLT ))
107     {
108         // Problème détecté (mauvaise tension de batterie ou batterie déchargée)
109         HAL_GPIO_WritePin(LED_RED_GPIO_Port, LED_RED_Pin, GPIO_PIN_SET); // Allume la LED rouge
110         nouvelEtatBatterie = ALARME;
111     }
112 else
113 {
114     // Aucun problème (batterie avec la bonne tension et chargée)
115     HAL_GPIO_WritePin(LED_RED_GPIO_Port, LED_RED_Pin, GPIO_PIN_RESET); // Eteint la LED rouge
116     nouvelEtatBatterie = OK;
117 }
118 }
119
120 // Code de test qui change l'état de la batterie à chaque réveil
121 // Permet de tester l'activation de la LED et l'envoi des trames UART vers l'ESP
122 if(DEBUG_ALARME_BATTERIE)
123 {
124     if(infoBatterie->etatBatterie == ALARME) // Si l'état était ALARME on passe en OK
125     {
126         nouvelEtatBatterie = OK;
127         HAL_GPIO_WritePin(LED_RED_GPIO_Port, LED_RED_Pin, GPIO_PIN_RESET); // Eteint la LED rouge
128     }
129     else // Si l'état n'était pas ALARME on passe en ALARME
130     {
131         nouvelEtatBatterie = ALARME;
132         HAL_GPIO_WritePin(LED_RED_GPIO_Port, LED_RED_Pin, GPIO_PIN_SET); // Allume la LED rouge
133     }
134 }
135
136     return nouvelEtatBatterie;
137 }
138
139
140 ///////////////////////////////////////////////////////////////////
141 /////////////////////////////////////////////////////////////////// Fonction ArreterLectureTensionBatterie
142 /////////////////////////////////////////////////////////////////// Description: Réinitialise le transistor utilisé pour lire l'état de la batterie, défini le compteur pour le délai avant la prochaine vérification
143 /////////////////////////////////////////////////////////////////// Entrées: Pointeur:TIM_HandleTypeDef htim, Pointeur:InfoBatterie infoBatterie
144 /////////////////////////////////////////////////////////////////// Sorties: -
145 void ArreterLectureTensionBatterie(TIM_HandleTypeDef* htim, InfoBatterie* infoBatterie)
146 {
147     // Réinitialiser le transistor utilisé pour lire l'état de la batterie
148     HAL_GPIO_WritePin(EN_VBAT_GPIO_Port, EN_VBAT_Pin, GPIO_PIN_RESET);
149
150     // Arrêt de l'interruption timer utilisée pour attendre entre l'enclenchement du transistor et la mesure de l'état de la batterie
151     HAL_TIM_Base_Stop_IT(htim);
152
153     // Compteur cadencé sur la fréquence de réveil du STM
154     // Nombre de fois que le STM doit se réveiller avant de procéder à une nouvelle vérification de l'état de la batterie
155     infoBatterie->compteurCheckBatterie = DELAI_VERIFICATION_BATTERIE_MS / DELAI_ENTRE_DEUX_REVEILS_MS;
156 }

```

# Annexe S.6

```
1  /* **** */
2  /** Descriptive File Name
3
4  @Company
5      ETML-ES
6
7  @File Name
8      gestionBatterie.h
9
10 @Auteurs
11     - Perret Mélissa
12
13 @Description
14     Fonctions liées à la gestion de la batterie
15 */
16 /* **** */
17
18
19
20 #ifndef _GESTION_BATTERIE_H
21 #define _GESTION_BATTERIE_H
22
23
24 /* **** */
25 /* **** */
26 /* Section: Included Files
27 /* **** */
28 /* **** */
29
30 /* This section lists the other files that are included in this file.
31 */
32
33 #include "main.h" // pour Etat
34
35 #ifdef __cplusplus
36 extern "C" {
37 #endif
38
39 // ****
40 // ****
41 // Section: Global Data
42 // ****
43 // ****
44
45 // Structure pour stocker les informations liée à la batterie
46 typedef struct {
47     uint32_t compteurCheckBatterie;
48     Etat etatBatterie;
49 } InfoBatterie;
50
51 // Valeurs des resistances utilisées pour le pont diviseur de tension
52 #define R3_THEORIQUE 18000 // Valeur théorique de la résistance
53 #define R6_THEORIQUE 5100 // Valeur théorique de la résistance
54
55 #define VALEUR_BATTERIE_VOLT 4.5 // Valeur batterie (4.5 ou 6.0)
56
57 #define VALEUR_3PILES_VOLT 4.5 // Tension pour 3 piles
58 #define VALEUR_BATTERIE_4_5V_MIN_VOLT 4.0 // Valeur min acceptable pour que la batterie 4.5V soit
59 // considérée comme chargée
60 #define VALEUR_BATTERIE_4_5V_MAX_VOLT 4.8 // Valeur max acceptable pour que la batterie 4.5V soit
61 // considérée comme chargée
62
63 #define VALEUR_4PILES_VOLT 6.0 // Tension pour 4 piles
64 #define VALEUR_BATTERIE_6V_MIN_VOLT 4.5 // Valeur min acceptable pour que la batterie 6V soit
65 // considérée comme chargée
66 #define VALEUR_BATTERIE_6V_MAX_VOLT 6.4 // Valeur max acceptable pour que la batterie 6V soit
67 // considérée comme chargée
68
69 // Attention: DELAI_ENTRE_DEUX_REVEILS_MS doit coincider avec la valeur DUREE_SOMMEIL_MS du côté de
70 // l'ESP (Sommeil.h)
71 // #define DELAI_ENTRE_DEUX_REVEILS_MS 1*3600*1000 // Délai entre deux réveils du STM32
72 // (1*3600*1000 = 1h).
73 #define DELAI_ENTRE_DEUX_REVEILS_MS 5000 // Délai test entre deux réveils du STM32
74
75 // #define DELAI_VERIFICATION_BATTERIE_MS 4*3600*1000 // Délai entre deux vérifications de batterie
76 // (4*3600*1000 = 4h)
77 #define DELAI_VERIFICATION_BATTERIE_MS 5000 // Délai test entre deux vérifications de
```

```

K:\ES\PROJETS\SLO\2409_MesureTH_RefrigerateurCongelateur\soft\Firmware\STM32\2409_MesureTH_V1\2409_MesureTH

    batterie (faible délai pour tester le bon fonctionnement du code)
71
72 #define DELAI_ACTIVATION_TRANSISTOR_MS 10 // Délai entre l'enclenchement du transistor et la mesure
73 de l'état de la batterie
74 #define DELAI_TIMER6_MS 10 // Délai entre 2 interruptions du timer 6
75 #define DEBUG_ALARME_BATTERIE false // Quand vrai, change l'état de la batterie à chaque réveil
76 (permet de tester l'envoi des trames)
77
78 /* **** */
79 /* **** */
80 /* Section: Constants */
81 /* **** */
82 /* **** */
83
84 // ****
85 // ****
86 // Section: Prototypes
87 // ****
88 // ****
89 //-----
```

90

```

91 bool GestionCheckBatterie(ADC_HandleTypeDef *hadc, TIM_HandleTypeDef *htim, InfoBatterie
*infoBatterie);
92 Etat MesurerEtatBatterie(ADC_HandleTypeDef *hadc, TIM_HandleTypeDef *htim, InfoBatterie
*infoBatterie);
93 void LancerCheckBatterie(TIM_HandleTypeDef* htim, InfoBatterie *infoBatterie);
94 Etat ControleEtatChargeBatterie(ADC_HandleTypeDef* hadc, InfoBatterie* infoBatterie);
95 void ArreterLectureTensionBatterie(TIM_HandleTypeDef* htim, InfoBatterie* infoBatterie);
96
97
98 #ifdef __cplusplus
99 }
100#endif
101
102#endif /* _GESTION_BATTERIE_H */
```

103

# Annexe S.7

```

1 // fonctionsADC.c
2 //
3 // Description : fonctions liée à l'ADC
4 // Auteur : Perret Mélissa
5 // Création : 09/05/2024
6 // Modifications : --
7
8 // Version : V1.0
9 /*-----*/
10
11 //-----CONFIGURATION UART HARMONY-----//
12 /*
13 * Activer le UART -> CubeMX
14 * Régler le BaudRate @ 9600 (comme signalé dans le datasheet)
15 * Activer 8 bits de data, aucune parité et 1 bit de stop
16 * Paramétriser l'over Sampling à 16
17 * Activer le control de flux
18 */
19
20 #include "fonctionsADC.h"
21
22 ///////////////////////////////////////////////////////////////////
23 /////////////////////////////////////////////////////////////////// Fonction LectureValeurAdcBrute (lecture de registre du ADC)
24 /////////////////////////////////////////////////////////////////// Description: lire la valeur brute de l'adc en fonction du channel
25 ////////////////////////////////////////////////////////////////// Entrées: Pointeur:ADC_HandleTypeDef hadc, uint8_t channel (numéro de channel à utiliser pour l'ADC)
26 ////////////////////////////////////////////////////////////////// Sorties: uint16_t (valeur brute de l'ADC, valeur comprise entre 0 à 4095)
27 uint16_t LectureValeurAdcBrute(ADC_HandleTypeDef* hadc, uint8_t channel)
28 {
29     HAL_ADC_Stop(hadc); // Arrêt de l'ADC
30
31     HAL_StatusTypeDef calibrationStatus = HAL_ADCEx_Calibration_Start(hadc); // Nouvelle calibration par précaution
32
33     hadc->Instance->CHSELR = 1 << channel; // Sélection du bon channel
34
35     HAL_StatusTypeDef adcStatus = HAL_ADC_Start(hadc); // Démarrage de l'ADC
36
37     // Réalise une mesure du registre de l'ADC, sort de la fonction après un certain temps si pas possible (TIMEOUT_LECTURE_ADC_MS)
38     while (HAL_ADC_PollForConversion(hadc, TIMEOUT_LECTURE_ADC_MS) != HAL_OK) // Attente fin de conversion
39     {
40     }
41
42     uint32_t mesure = 0; // Déclaration variable locale (valeur mesurée)
43     mesure = HAL_ADC_GetValue(hadc); // Sauvegarde de la mesure
44
45     HAL_ADC_Stop(hadc); // Arrêt de l'ADC
46
47     return mesure; // Retourne la valeur brute mesurée
48 }
49 /////////////////////////////////////////////////////////////////// Fonction ConversionValeurAdcEnVolt: conversion d'une valeur brute ADC en une tension
50 /////////////////////////////////////////////////////////////////// Description: conversion d'une valeur brute ADC (0 à 4095) en une tension (en Volt)
51 ////////////////////////////////////////////////////////////////// Entrées: uint16_t valeurADC (valeur brute ADC à convertir),
52 ////////////////////////////////////////////////////////////////// uint32_t pontDiviseurResistanceHaute (valeur de la resistance en haut dans le pont diviseur de tension)
53 ////////////////////////////////////////////////////////////////// uint32_t pontDiviseurResistanceBasse (valeur de la resistance en bas dans le pont diviseur de tension)

```

```
54 ////////////////////////////////////////////////////////////////////  
55 // Sorties: float (valeur brute de l'ADC convertie en valeur tension)  
56 float ConversionValeurAdcEnVolt(uint16_t valeurADC, uint32_t pontDiviseurResistanceHaute, uint32_t pontDiviseurResistanceBasse)  
57 {  
58     // Rappels ponts diviseur de tension:  
59     // Vout = (Vin * R2) / (R1 + R2)  
60     // Vin = (Vout / R2) * (R1 + R2)  
61  
62     uint32_t R1 = pontDiviseurResistanceHaute; // Plus simple pour lire le code qui suit (formules)  
63     uint32_t R2 = pontDiviseurResistanceBasse; // Plus simple pour lire le code qui suit (formules)  
64  
65     float valeurAdcEnVolt = valeurADC * VREF_ADC_V / VAL_ADC_MAX; // Correspond au Vout du pont diviseur de tension  
66     return valeurAdcEnVolt * (R1 + R2) / R2; // Prise en compte du pont diviseur de tension pour retrouver le Vin  
67 }
```

# Annexe S.8

```
1  /* **** */
2  /** Descriptive File Name
3
4  @Company
5      ETML-ES
6
7  @File Name
8      fonctionsADC.h
9
10 @Auteurs
11     - Perret Mélissa
12
13 @Description
14     Fonctions liée à l'ADC
15 */
16 /* **** */
17
18
19
20 #ifndef _FONCTIONS_H
21 #define _FONCTIONS_H
22
23
24 /* **** */
25 /* **** */
26 /* Section: Included Files */
27 /* **** */
28 /* **** */
29
30 /* This section lists the other files that are included in this file.
31 */
32
33 #include "stm32f0xx_hal.h" // pour uint16_t et ADC_HandleTypeDef
34
35 #ifdef __cplusplus
36 extern "C" {
37 #endif
38
39 // ****
40 // ****
41 // Section: Global Data
42 // ****
43 // ****
44
45 /* **** */
46 /* **** */
47 /* Section: Constants */
48 /* **** */
49 /* **** */
50
51 #define VAL_ADC_MAX 4095 // Valeur max de l'ADC suivant sa resolution 12 bit
52 #define VREF_ADC_V 3.3 // Valeur de la tension max de l'adc
53
54 #define TIMEOUT_LECTURE_ADC_MS 10000 // Timeout lors de la mesure du registre du ADC
55
56
57 // ****
58 // ****
59 // Section: Prototypes
60 // ****
61 // ****
62 //-----
```

---

```
64 uint16_t LectureValeurAdcBrute(ADC_HandleTypeDef* hadc, uint8_t channel);
65 float ConversionValeurAdcEnVolt(uint16_t valeurADC, uint32_t pontDiviseurResistanceHaute, uint32_t
pontDiviseurResistanceBasse);
66 uint16_t ConversionVoltEnDac(float volt);
67
68
69 #ifdef __cplusplus
70 }
71 #endif
72
73 #endif /* _FONCTIONS_H */
```

```

1  /* **** */
2  /** Descriptive File Name
3
4  @Company
5      ETML-ES
6
7  @File Name
8      communication.h
9
10 @Auteurs
11     - Perret Mélissa
12
13 @Description
14     Fonctions liées à la communication ESP/STM
15 */
16 /* **** */
17
18
19
20 #ifndef _COMMUNICATION_H
21 #define _COMMUNICATION_H
22
23
24 /* **** */
25 /* **** */
26 /* Section: Included Files */
27 /* **** */
28 /* **** */
29
30 /* This section lists the other files that are included in this file.
31 */
32
33 #include "main.h" // pour DefinitionValeur
34
35
36 #ifdef __cplusplus
37 extern "C" {
38 #endif
39
40 // ****
41 // ****
42 // Section: Global Data
43 // ****
44 // ****
45
46 #define TAILLE_BUFFER 50 // Taille du tableau utilisé pour stocker les données UART reçues
47 #define UART_RECEPTION_TIMEOUT_MS 3000 // Timeout lors de la réception UART
48 #define DELAI_ENVOIE_TRAME_MS 1000 // Délai entre le réveil de l'ESP et l'envoie de trame pour
        s'assurer que l'ESP est prêt à recevoir les trames
49
50
51 /* **** */
52 /* **** */
53 /* Section: Constants */
54 /* **** */
55 /* **** */
56
57 // ****
58 // ****
59 // Section: Prototypes
60 // ****
61 // ****
62 //-----
63
64 void ReceptionnerTrameUART(UART_HandleTypeDef* huart, DefinitionValeur valeursServeur[]);
65 void EnvoyerTrameUART(UART_HandleTypeDef* huart, uint8_t index, double valeur);
66
67
68 #ifdef __cplusplus
69 }
70 #endif
71
72 #endif /* _COMMUNICATION_H */
73

```

# Annexe S.9

# Annexe S.10

```

1 // communication.c
2 //
3 // Description : fonctions liées à la communication ESP/STM
4 // Auteur : Perret Mélissa
5 // Création : 16/09/2024
6 // Modifications : --
7
8 // Version : V1.0
9 /*-----*/
10
11
12 #include "communication.h"
13 #include <string.h> // pour memset et memcpy
14
15
16 uint8_t receptionTramesUART[TAILLE_BUFFER]; // Tableau pour stocker les trames UART reçues
17
18 // Important: sizeof ne renvoie pas les bonnes valeurs quand utilisé pour définir la valeurs de variables déclarées en const
19 uint8_t OCTET_DEBUT = 0x02; // STX (Start of Text)
20 uint8_t OCTET_FIN = 0x03; // ETX (End of Text)
21 uint8_t TAILLE_TRAME = sizeof(OCTET_DEBUT) + sizeof(int8_t) + sizeof(char) + sizeof(double) + sizeof(OCTET_FIN);
22
23
24 ///// Fonction ReceptionnerTrameUART (réception analyse et traitement des trames UART provenant de l'ESP)
25 ///// Description: réception des données UART
26 /////           analyse des données pour localiser les trames (caractères de début et de fin)
27 /////           analyse des trames reçues pour extraire et stocker les données reçues
28 ///// Entrées: Pointeur:UART_HandleTypeDef huart, Tableau:DefinitionValeur valeursServeur (tableau pour identifier et stocker les valeurs reçues)
29 ///// Sorties: -
30 void ReceptionnerTrameUART(UART_HandleTypeDef* huart, DefinitionValeur valeursServeur[])
31 {
32     int uartStatut; // Déclaration variable locale (valeur de retour de la lecture UART)
33
34     do
35     {
36         // Réception données UART
37         uartStatut = HAL_UART_Receive(huart, (uint8_t*) receptionTramesUART, TAILLE_TRAME, UART_RECEPTION_TIMEOUT_MS);
38         if(uartStatut == HAL_OK)
39         {
40             // On parcourt le buffer pour trouver toutes les trames (identifiées par OCTET_DEBUT)
41             for (int indexDebutTrame = 0; indexDebutTrame < TAILLE_BUFFER; indexDebutTrame++)
42             {
43                 if (receptionTramesUART[indexDebutTrame] == OCTET_DEBUT) // Début de trame trouvé
44                 {
45                     int indexFinTrame = indexDebutTrame + TAILLE_TRAME - 1;
46                     if(indexFinTrame < TAILLE_BUFFER && receptionTramesUART[indexFinTrame] == OCTET_FIN) // Si on a bien une fin de trame comme attendu
47                     {
48                         uint8_t index = receptionTramesUART[indexDebutTrame + 1]; // Deuxième octet de la trame correspond à l'identifiant de la
49                         // Troisième octet correspond à un séparateur ':' pour facilier la lecture des trames lors du debuggage
50                         // Exemple de trame
51                         // 0:12
52

```

```

53     // Effet de la trame: mettre la valeur de SEUIL_TEMPERATURE_MIN à 12
54     // valeursServeur[0].valeur = 12
55
56     // Explications pour memcpy : https://en.cppreference.com/w/cpp/string/byte	memcpy
57     // Les 8 octets suivants (octets 4 à 11) correspondent à la valeur de la trame
58     // On copie la valeur de la trame dans la variable qu'on utilise pour stocker les valeurs du serveur
59     memcpy(&valeursServeur[index].valeur, &receptionTramesUART[indexDebutTrame + 3], sizeof(double));
60
61     rafraichirEPaper = true; // Une nouvelle valeur reçue implique qu'un rafraîchissement de l'écran sera nécessaire
62 }
63 }
64 }
65 }
66 } while (uartStatut == HAL_OK); // Continuer tant qu'on reçoit quelque chose
67 }
68
69 //////////////////////////////////////////////////////////////////// Fonction EnvoyerTrameUART (construire et transmettre une trame UART à l'ESP)
70 //////////////////////////////////////////////////////////////////// Description: construire et transmettre la trame UART à l'ESP
71 //////////////////////////////////////////////////////////////////// Entrées: Pointeur:UART_HandleTypeDef huart, uint8_t index, double valeur
72 //////////////////////////////////////////////////////////////////// Sorties: -
73 void EnvoyerTrameUART(UART_HandleTypeDef* huart, uint8_t index, double valeur)
74 {
75     HAL_GPIO_WritePin(ALARME_GPIO_Port, ALARME_Pin, GPIO_PIN_SET); // Réveiller l'ESP
76     HAL_Delay(DELAI_ENVOIE_TRADE_MS); // Délai pour être sûr que l'ESP soit réveillé et prêt à recevoir les trames
77
78     uint8_t trame[TAILLE_TRADE]; // Buffer pour stocker la trame à envoyer
79     char séparateur = ':'; // Séparateur entre l'index et la valeur dans les trames, pour faciliter le débogage
80
81     // Explications memset : https://cplusplus.com/reference/cstring/memset/
82     memset(trame, 0, sizeof(trame)); // Réinitialisation de la trame en mettant tous les bits à 0 (par précautions)
83
84     // Explications memcpy : https://en.cppreference.com/w/cpp/string/byte	memcpy
85     memcpy(&trame[0], &OCTET_DEBUT, sizeof(OCTET_DEBUT)); // OCTET_DEBUT
86     memcpy(&trame[sizeof(OCTET_DEBUT)], &index, sizeof(index)); // index
87     memcpy(&trame[sizeof(OCTET_DEBUT) + sizeof(index)], &séparateur, sizeof(séparateur)); // ':'
88     memcpy(&trame[sizeof(OCTET_DEBUT) + sizeof(index) + sizeof(séparateur)], &valeur, sizeof(double)); // valeur
89     memcpy(&trame[sizeof(OCTET_DEBUT) + sizeof(index) + sizeof(séparateur) + sizeof(valeur)], &OCTET_FIN, sizeof(OCTET_FIN)); // OCTET_FIN
90
91     HAL_UART_Transmit(huart, trame, sizeof(trame), 1000); // Transmission de la trame à l'ESP
92 }
93

```

# Annexe S.11

```

1 //  mesures.c
2 //
3 // Description : fonctions liées aux mesures de la sonde (température et humidité)
4 // Auteur : Perret Mélissa
5 // Création : 16/09/2024
6 // Modifications : --
7
8 // Version : V1.0
9 /*-----*/
10
11
12 #include "mesures.h"
13 #include "shtc3.h" // pour utiliser la sonde
14
15
16 /////////////////////////////////////////////////////////////////// Fonction EffectuerMesuresSonde (mesure température et humidité)
17 /////////////////////////////////////////////////////////////////// Description: mesure température et humidité
18 /////////////////////////////////////////////////////////////////// vérification dépassement des seuils
19 /////////////////////////////////////////////////////////////////// en cas de changement d'état, demande rafraîchissement écran et l'envoie de la trame UART
20 ////////////////////////////////////////////////////////////////// Entrées: Pointeur:I2C_HandleTypeDef hi2c2, Pointeur:Mesures mesures, Tableau:DefinitionValeur valeursServeur
21 ////////////////////////////////////////////////////////////////// Sorties: Etat (état de l'alarme liée au dépassement des seuils)
22 Etat EffectuerMesuresSonde(I2C_HandleTypeDef* hi2c2, Mesures* mesures, DefinitionValeur valeursServeur[])
23 {
24     if (shtc3_read_id(hi2c2)) // Si on détecte la sonde
25     {
26         // Effectuer les mesures
27         if (shtc3_perform_measurements(hi2c2, &mesures->temperatureEntierActuelle, &mesures->humiditeEntierActuelle)) // Si la mesure est réussie
28         {
29             // Les résultats de la sonde sont transmis sous forme d'entier
30             // Il s'agit en réalité de la valeur avec deux décimales, remis sous forme d'entier en étant multiplié par 100
31             // On effectue l'opération inverse (division par 100) pour récupérer la valeur décimale
32             mesures->temperatureActuelle = (float)mesures->temperatureEntierActuelle/100;
33             mesures->humiditeActuelle = (float)mesures->humiditeEntierActuelle/100;
34
35             // Pour éviter de rafraîchir l'écran E-paper trop souvent, on mesure l'écart entre la nouvelle valeur et la dernière valeur affichée
36             // Si l'écart est supérieur à l'écart minimal on demandera le rafraîchissement de l'écran
37             // Les valeurs des écarts sont transmises par le serveur
38
39             // Ecart température
40             double ecartTemperatureMinimal = valeursServeur[ECART_TEMPERATURE].valeur; // Ecart minimal de température pour effectuer un
rafaichissement du E-paper
41             float ecartTemperature = mesures->temperatureActuelle - mesures->temperatureAffichee; // Ecart entre la température mesurée et la
température affichée
42             if(ecartTemperature >= ecartTemperatureMinimal || ecartTemperature <= -ecartTemperatureMinimal) // Si l'écart est plus important que
l'écart minimal
43             {
44                 rafraichirEPaper = true; // Indiquer qu'un rafraîchissement de l'écran est nécessaire
45             }
46
47             // Ecart humidité
48             double ecartHumiditeMinimal= valeursServeur[ECART_HUMIDITE].valeur; // Ecart minimal d'humidité pour effectuer un rafraîchissement du E-paper
49             float ecartHumidite = mesures->humiditeActuelle - mesures->humiditeAffichee; // Ecart entre l'humidité mesurée et l'humidité affichée
50             if(ecartHumidite >= ecartHumiditeMinimal || ecartHumidite <= -ecartHumiditeMinimal) // Si l'écart est plus important que l'écart minimal

```

```
51     {
52         rafraichirEPaper = true; // Indiquer qu'un rafraîchissement de l'écran est nécessaire
53     }
54
55     // Vérification dépassement des seuils (pour définir le nouvel état de l'alarme dépassement des seuils)
56     bool depassementTemperature = mesures->temperatureActuelle < valeursServeur[SEUIL_TEMPERATURE_MIN].valeur || mesures->temperatureActuelle >
57     valeursServeur[SEUIL_TEMPERATURE_MAX].valeur;
58     bool depassementHumidite = mesures->humiditeActuelle < valeursServeur[SEUIL_HUMIDITE_MIN].valeur || mesures->humiditeActuelle >
59     valeursServeur[SEUIL_HUMIDITE_MAX].valeur;
60     Etat nouvelEtatSeuils = INDEFINI;
61     if(depassementTemperature || depassementHumidite)
62     {
63         nouvelEtatSeuils = ALARME; // Dépassement des seuils min/max (température et/ou humidité)
64     }
65     else
66     {
67         nouvelEtatSeuils = OK; // Pas de dépassement des seuils min/max (température et humidité)
68     }
69
70     // Code de test qui change l'état de l'alarme dépassement des seuils à chaque réveil
71     // Permet de tester l'envoi des trames vers l'ESP
72     if(DEBUG_ALARME_SEUILS)
73     {
74         if(mesures->etatSeuils == ALARME) // Si l'état était ALARME on passe en OK
75         {
76             nouvelEtatSeuils = OK;
77         }
78         else
79         {
80             nouvelEtatSeuils = ALARME; // Si l'état n'était pas ALARME on passe en ALARME
81         }
82     }
83
84     return nouvelEtatSeuils;
85 }
86 else
87 {
88     return mesures->etatSeuils; // Mesure impossible, pas de changement d'état
89 }
90 else
91 {
92     return mesures->etatSeuils; // Mesure impossible, pas de changement d'état
93 }
94 }
```

```

1  /* **** */
2  /** Descriptive File Name
3
4  @Company
5      ETML-ES
6
7  @File Name
8      mesures.h
9
10 @Auteurs
11     - Perret Mélissa
12
13 @Description
14     Fonctions liées aux mesures de la sonde (température et humidité)
15 */
16 /* **** */
17
18
19
20 #ifndef _MESURES_H
21 #define _MESURES_H
22
23
24 /* **** */
25 /* **** */
26 /* Section: Included Files */
27 /* **** */
28 /* **** */
29
30 /* This section lists the other files that are included in this file.
31 */
32
33 #include "main.h" // pour Etat et DefinitionValeur
34
35
36 #ifdef __cplusplus
37 extern "C" {
38 #endif
39
40 // ****
41 // ****
42 // Section: Global Data
43 // ****
44 // ****
45
46 // Structure pour stocker les informations liées aux mesures de la température et de l'humidité
47 typedef struct {
48     int32_t temperatureEntierActuelle;
49     float temperatureActuelle;
50     float temperatureAffichee;
51
52     int32_t humiditeEntierActuelle;
53     float humiditeActuelle;
54     float humiditeAffichee;
55
56     Etat etatSeuils;
57 } Mesures;
58
59
60 /* **** */
61 /* **** */
62 /* Section: Constants */
63 /* **** */
64 /* **** */
65
66 #define DEBUGALARME_SEUILS false // Quand vrai, change l'état des seuils (alarme) à chaque réveil
67 (permet de tester l'envoi des trames)
68
69 // ****
70 // ****
71 // Section: Prototypes
72 // ****
73 // ****
74 //-----
75
76 Etat EffectuerMesuresSonde(I2C_HandleTypeDef* hi2c2, Mesures* mesures, DefinitionValeur

```

# Annexe S.12

K:\ES\PROJETS\SLO\2409\_MesureTH\_RefrigerateurCongelateur\soft\Firmware\STM32\2409\_MesureTH\_V1\2409\_MesureTH

```
77 valeursServeur[]);  
78  
79 #ifdef __cplusplus  
80 }  
81 #endif  
82  
83 #endif /* _MESURES_H */  
84
```

```
1 // sommeil.c
2 //
3 // Description : fonctions liées à l'entrée et sortie du mode sommeil
4 // Auteur : Perret Mélissa
5 // Création : 21/09/2024
6 // Modifications : --
7
8 // Version : V1.0
9 /*-----*/
10
11 #include "sommeil.h"
12 #include "main.h" // pour GPIO, HAL, REVEIL_Pin et REVEIL_GPIO_Port
13
14
15 ///// Fonction InitialiserGestionSommeil ()
16 ///// Description:
17 ///// Entrées: -
18 ///// Sorties: -
19 void InitialiserGestionSommeil()
20 {
21     // Initialisation de la pin utilisée pour pouvoir être reveillé par l'ESP
22     GPIO_InitTypeDef GPIO_InitStruct = {0};
23     GPIO_InitStruct.Pin = REVEIL_Pin;
24     GPIO_InitStruct.Mode = GPIO_MODE_IT_RISING;
25     GPIO_InitStruct.Pull = GPIO_PULLUP | GPIO_PULLDOWN;
26     HAL_GPIO_Init(REVEIL_GPIO_Port, &GPIO_InitStruct);
27
28     // Activation de l'interruption pour pouvoir être reveillé par l'ESP
29     HAL_NVIC_SetPriority(EXTI4_15_IRQn, 2, 0);
30     HAL_NVIC_EnableIRQ(EXTI4_15_IRQn);
31
32     // Initialisation de la pin utilisée pour réveiller l'ESP (pour pouvoir le réveiller plus tard)
33     HAL_GPIO_WritePin(ALARME_GPIO_Port, ALARME_Pin, GPIO_PIN_RESET);
34 }
35
36 ///// Fonction EntrerModeSommeil (mettre le STM en mode stop)
37 ///// Description: logique pour mettre le STM en mode stop et remise en route après réveil
38 ///// Entrées: -
39 ///// Sorties: -
40 void EntrerModeSommeil()
41 {
42     HAL_SuspendTick(); // Désactiver le timer SysTick (pour éviter qu'il provoque un réveil)
43
44     HAL_PWR_EnterSTOPMode(PWR_LOWPOWERREGULATOR_ON, PWR_STOPENTRY_WFI); // Entrer en mode stop
45     // Exécution du programme en pause tant que le STM n'est pas reveillé par l'ESP
46
47     // Reprise de l'exécution du programme ici après le réveil
48     HAL_ResumeTick(); // Réactiver le timer SysTick
49     HAL_GPIO_WritePin(ALARME_GPIO_Port, ALARME_Pin, GPIO_PIN_RESET); // Initialiser pin pour pouvoir réveiller ESP plus tard
50 }
51
```

## Annexe S.13

```

1  /* **** */
2  /** Descriptive File Name
3
4  @Company
5      ETML-ES
6
7  @File Name
8      sommeil.h
9
10 @Auteurs
11     - Perret Mélissa
12
13 @Description
14     Fonctions liées à l'entrée et sortie du mode sommeil
15 */
16 /* **** */
17
18
19
20 #ifndef _SOMMEIL_H
21 #define _SOMMEIL_H
22
23
24 /* **** */
25 /* **** */
26 /* Section: Included Files */
27 /* **** */
28 /* **** */
29
30 /* This section lists the other files that are included in this file.
31 */
32
33 #ifdef __cplusplus
34 extern "C" {
35 #endif
36
37 // ****
38 // ****
39 // Section: Global Data
40 // ****
41 // ****
42
43 /* **** */
44 /* **** */
45 /* Section: Constants */
46 /* **** */
47 /* **** */
48
49 // ****
50 // ****
51 // Section: Prototypes
52 // ****
53 // ****
54 //-----
55
56 void InitialiserGestionSommeil(void);
57 void EntrerModeSommeil(void);
58
59
60 #ifdef __cplusplus
61 }
62 #endif
63
64 #endif /* _SOMMEIL_H */
65

```

# Annexe S.14

# Annexe S.15

```

1  /* USER CODE BEGIN Header */
2  /**
3   * @file          : main.c
4   * @brief         : Main program body
5   * @attention
6   *
7   * Copyright (c) 2024 STMicroelectronics.
8   * All rights reserved.
9   *
10  * This software is licensed under terms that can be found in the LICENSE file
11  * in the root directory of this software component.
12  * If no LICENSE file comes with this software, it is provided AS-IS.
13  *
14  */
15  /**
16  */
17  /*
18  /* USER CODE END Header */
19  /* Includes -----*/
20 #include "main.h"
21
22 /* Private includes -----*/
23 /* USER CODE BEGIN Includes */
24 #include <stdbool.h>
25 // #include "stm32f0xx.h"
26 #include "mesures.h"
27 #include "affichage.h"
28 #include "gestionBatterie.h"
29 #include "communication.h"
30 #include "sommel.h"
31 #include "EPD_2in13_V4.h" // TEMP
32 /* USER CODE END Includes */
33
34 /* Private typedef -----*/
35 /* USER CODE BEGIN PTD */
36
37 /* USER CODE END PTD */
38
39 /* Private define -----*/
40 /* USER CODE BEGIN PD */
41
42 /* USER CODE END PD */
43
44 /* Private macro -----*/
45 /* USER CODE BEGIN PM */
46
47 /* USER CODE END PM */
48
49 /* Private variables -----*/
50 ADC_HandleTypeDef hadc;
51
52 I2C_HandleTypeDef hi2c2;
53
54 RTC_HandleTypeDef hrtc;
55
56 SPI_HandleTypeDef hspil;
57
58 TIM_HandleTypeDef htim6;
59
60 UART_HandleTypeDef huart1;
61
62 /* USER CODE BEGIN PV */
63
64 EtatSTM etatSTM = INITIALISATION;
65
66 // Variables mesures
67 Mesures mesures;
68
69 // https://www.arduino.cc/reference/en/language/variables/data-types/array/
70 DefinitionValeur valeursServeur[] = {
71     { "seuilTemperatureMin", 2 },
72     { "seuilTemperatureMax", 8 },
73     { "ecartTemperature", 1 },
74     { "seuilHumiditeMin", 30 },
75     { "seuilHumiditeMax", 60 },
76     { "ecartHumidite", 5},
77 };

```

```
78 // Variables affichage
79 bool rafraichirEPaper = false;
80 bool receptionTramesEspTermine = false;
82
83 // Variables batterie
84 InfoBatterie infoBatterie;
85
86 /* USER CODE END PV */
87
88 /* Private function prototypes -----*/
89 void SystemClock_Config(void);
90 static void MX_GPIO_Init(void);
91 static void MX_ADC_Init(void);
92 static void MX_I2C2_Init(void);
93 static void MX_SPI1_Init(void);
94 static void MX_USART1_UART_Init(void);
95 static void MX_TIM6_Init(void);
96 static void MX_RTC_Init(void);
97 /* USER CODE BEGIN PFP */
98
99 /* USER CODE END PFP */
100
101 /* Private user code -----*/
102 /* USER CODE BEGIN 0 */
103
104
105
106 /* USER CODE END 0 */
107
108 /**
109  * @brief  The application entry point.
110  * @retval int
111  */
112 int main(void)
113 {
114
115     /* USER CODE BEGIN 1 */
116
117     /* USER CODE END 1 */
118
119     /* MCU Configuration-----*/
120
121     /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
122     HAL_Init();
123
124     /* USER CODE BEGIN Init */
125
126     /* USER CODE END Init */
127
128     /* Configure the system clock */
129     SystemClock_Config();
130
131     /* USER CODE BEGIN SysInit */
132
133     /* USER CODE END SysInit */
134
135     /* Initialize all configured peripherals */
136     MX_GPIO_Init();
137     MX_ADC_Init();
138     MX_I2C2_Init();
139     MX_SPI1_Init();
140     MX_USART1_UART_Init();
141     MX_TIM6_Init();
142     MX_RTC_Init();
143     /* USER CODE BEGIN 2 */
144
145     /* USER CODE END 2 */
146
147     /* Infinite loop */
148     /* USER CODE BEGIN WHILE */
149     while (1)
150     {
151         /* USER CODE END WHILE */
152
153         /* USER CODE BEGIN 3 */
154
```

```

155     switch(etatSTM)
156     {
157         case INITIALISATION:
158             Initialisation(); // Initialisation au lancement du STM
159             etatSTM = RECEPTION; // Pret à recevoir les données de la part de l'ESP
160             break;
161         case RECEPTION:
162
163             // Note: la fréquence d'exécution de l'état RECEPTION est lié au timeout de la réception
164             UART (define UART_TIMEOUT)
165             ReceptionnerTrameUART(&huart1, valeursServeur);
166
167             // On vérifie que receptionTramesEspTermine pour être sûr qu'on a bien été réveillé par l'ESP
168             // On vérifie que HAL_GPIO_ReadPin(REVEIL_GPIO_Port, REVEIL_Pin) est GPIO_PIN_RESET pour
169             // être sûr que l'ESP à fini d'envoyer ses trames UART
170             if(receptionTramesEspTermine == true && HAL_GPIO_ReadPin(REVEIL_GPIO_Port, REVEIL_Pin) ==
171             GPIO_PIN_RESET)
172             {
173                 receptionTramesEspTermine = false;
174                 etatSTM = VERIFIER_BATTERIE; // Passage à l'étape suivante
175             }
176             break;
177         case VERIFIER_BATTERIE:
178         {
179             bool besoinMesurerEtatBatterie = GestionCheckBatterie(&hadc, &htim6, &infoBatterie);
180             if(besoinMesurerEtatBatterie == true)
181             {
182                 etatSTM = MESURER_BATTERIE; // Une mesure de l'état de la batterie est nécessaire
183             }
184             else
185             {
186                 etatSTM = MESURER SONDE;
187             }
188             break;
189         }
190         case MESURER_BATTERIE:
191         {
192             // Attente que le compteur décrémenté par l'interruption timer atteigne 0 (délai entre
193             activation transistor et mesure)
194             if(infoBatterie.compteurCheckBatterie <= 0)
195             {
196                 Etat nouvelEtatBatterie = MesurerEtatBatterie(&hadc, &htim6, &infoBatterie); // Procéder à
197                 la mesure de l'état de la batterie
198                 if(nouvelEtatBatterie != infoBatterie.etatBatterie)
199                 {
200                     infoBatterie.etatBatterie = nouvelEtatBatterie;
201                     EnvoyerTrameUART(&huart1, ETAT_BATTERIE, infoBatterie.etatBatterie); // Indiquer à l'ESP
202                     que l'état de la batterie à changé
203                 }
204                 etatSTM = MESURER SONDE;
205             }
206             break;
207         }
208         case MESURER SONDE:
209         {
210             Etat nouvelEtatSeuils = EffectuerMesuresSonde(&hi2c2, &mesures, valeursServeur); // Procéder
211             aux mesures de la température et de l'humidité
212             if(nouvelEtatSeuils != mesures.etatSeuils)
213             {
214                 mesures.etatSeuils = nouvelEtatSeuils;
215                 EnvoyerTrameUART(&huart1, ETAT_SEUILS, mesures.etatSeuils); // Indiquer à l'ESP que l'état
216                 de l'alarme à changé
217             }
218
219             // Indiquer à l'ESP qu'on a fini d'envoyer les trames UART pour qu'il puisse se rendormir
220             (si ne l'est pas déjà)
221             HAL_GPIO_WritePin(ALARME_GPIO_Port, ALARME_Pin, GPIO_PIN_RESET);
222
223             // Si un rafraîchissement de l'écran est nécessaire
224             if(rafraichirEPaper == true)
225             {
226                 etatSTM = RAFRAICHER_EPAPER;
227             }
228             else
229             {
230                 etatSTM = SOMMEIL;
231             }
232         }

```

```

K:\ES\PROJETS\SLO\2409_MesureTH_RefrigerateurCongelateur\soft\Firmware\STM32\2409_MesureTH_V1\2409_MesureTH

223         break;
224     }
225     case RAFRAICHIR_EPAPER:
226         rafraichirEPaper = false;
227         AffichageEPaper(&mesures, valeursServeur); // Procéder au rafraîchissement de l'écran
228         etatSTM = SOMMEIL;
229         break;
230     case SOMMEIL:
231         EntrerModeSommeil();
232         etatSTM = REVEIL;
233         break;
234     case REVEIL:
235         etatSTM = RECEPTION;
236         break;
237     }
238 }
239 /* USER CODE END 3 */
240 }
241 }

243 /**
244 * @brief System Clock Configuration
245 * @retval None
246 */
247 void SystemClock_Config(void)
248 {
249     RCC_OscInitTypeDef RCC_OscInitStruct = {0};
250     RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
251     RCC_PeriphCLKInitTypeDef PeriphClkInit = {0};

252     /** Initializes the RCC Oscillators according to the specified parameters
253      * in the RCC_OscInitTypeDef structure.
254     */
255     RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI14|RCC_OSCILLATORTYPE_LSI
256                                         |RCC_OSCILLATORTYPE_HSE;
257     RCC_OscInitStruct.HSEState = RCC_HSE_ON;
258     RCC_OscInitStruct.HSI14State = RCC_HSI14_ON;
259     RCC_OscInitStruct.HSI14CalibrationValue = 16;
260     RCC_OscInitStruct.LSIState = RCC_LSI_ON;
261     RCC_OscInitStruct.PLL.PLLState = RCC_PLL_NONE;
262     if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
263     {
264         Error_Handler();
265     }
266 }

268     /** Initializes the CPU, AHB and APB buses clocks
269     */
270     RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
271                                         |RCC_CLOCKTYPE_PCLK1;
272     RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_HSE;
273     RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
274     RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
275

276     if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_0) != HAL_OK)
277     {
278         Error_Handler();
279     }
280     PeriphClkInit.PeriphClockSelection = RCC_PERIPHCLK_USART1|RCC_PERIPHCLK_RTC;
281     PeriphClkInit.Usart1ClockSelection = RCC_USART1CLKSOURCE_PCLK1;
282     PeriphClkInit.RTCClockSelection = RCC_RTCCLKSOURCE_LSI;
283     if (HAL_RCCEx_PeriphCLKConfig(&PeriphClkInit) != HAL_OK)
284     {
285         Error_Handler();
286     }
287

288     /** Enables the Clock Security System
289     */
290     HAL_RCC_EnableCSS();
291 }

293 /**
294 * @brief ADC Initialization Function
295 * @param None
296 * @retval None
297 */
298 static void MX_ADC_Init(void)
299 {

```

```

C:\ES\PROJETS\SLO\2409_MesureTH_RefrigerateurCongelateur\soft\Firmware\STM32\2409_MesureTH_V1\2409_MesureTH

300
301     /* USER CODE BEGIN ADC_Init 0 */
302
303     /* USER CODE END ADC_Init 0 */
304
305     ADC_ChannelConfTypeDef sConfig = {0};
306
307     /* USER CODE BEGIN ADC_Init 1 */
308
309     /* USER CODE END ADC_Init 1 */
310
311     /** Configure the global features of the ADC (Clock, Resolution, Data Alignment and number of
312     conversion)
313     */
314     hadc.Instance = ADC1;
315     hadc.Init.ClockPrescaler = ADC_CLOCK_ASYNC_DIV1;
316     hadc.Init.Resolution = ADC_RESOLUTION_12B;
317     hadc.Init.DataAlign = ADC_DATAALIGN_RIGHT;
318     hadc.Init.ScanConvMode = ADC_SCAN_DIRECTION_FORWARD;
319     hadc.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
320     hadc.Init.LowPowerAutoWait = DISABLE;
321     hadc.Init.LowPowerAutoPowerOff = DISABLE;
322     hadc.Init.ContinuousConvMode = DISABLE;
323     hadc.Init.DiscontinuousConvMode = DISABLE;
324     hadc.Init.ExternalTrigConv = ADC_SOFTWARE_START;
325     hadc.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;
326     hadc.Init.DMAContinuousRequests = DISABLE;
327     hadc.Init.Overrun = ADC_OVR_DATA_PRESERVED;
328     if (HAL_ADC_Init(&hadc) != HAL_OK)
329     {
330         Error_Handler();
331     }
332
333     /** Configure for the selected ADC regular channel to be converted.
334     */
335     sConfig.Channel = ADC_CHANNEL_6;
336     sConfig.Rank = ADC_RANK_CHANNEL_NUMBER;
337     sConfig.SamplingTime = ADC_SAMPLETIME_1CYCLE_5;
338     if (HAL_ADC_ConfigChannel(&hadc, &sConfig) != HAL_OK)
339     {
340         Error_Handler();
341     }
342     /* USER CODE BEGIN ADC_Init 2 */
343
344     /* USER CODE END ADC_Init 2 */
345 }
346
347 /**
348 * @brief I2C2 Initialization Function
349 * @param None
350 * @retval None
351 */
352 static void MX_I2C2_Init(void)
353 {
354
355     /* USER CODE BEGIN I2C2_Init 0 */
356
357     /* USER CODE END I2C2_Init 0 */
358
359     /* USER CODE BEGIN I2C2_Init 1 */
360
361     /* USER CODE END I2C2_Init 1 */
362     hi2c2.Instance = I2C2;
363     hi2c2.Init.Timing = 0x2000090E;
364     hi2c2.Init.OwnAddress1 = 0;
365     hi2c2.Init.AddressingMode = I2C_ADDRESSINGMODE_7BIT;
366     hi2c2.Init.DualAddressMode = I2C_DUALADDRESS_DISABLE;
367     hi2c2.Init.OwnAddress2 = 0;
368     hi2c2.Init.OwnAddress2Masks = I2C_OA2_NOMASK;
369     hi2c2.Init.GeneralCallMode = I2C_GENERALCALL_DISABLE;
370     hi2c2.Init.NoStretchMode = I2C_NOSTRETCH_DISABLE;
371     if (HAL_I2C_Init(&hi2c2) != HAL_OK)
372     {
373         Error_Handler();
374     }
375

```

```

K:\ES\PROJETS\SLO\2409_MesureTH_RefrigerateurCongelateur\soft\Firmware\STM32\2409_MesureTH_V1\2409_MesureTH

376     /** Configure Analogue filter
377     */
378     if (HAL_I2CEx_ConfigAnalogFilter(&hi2c2, I2C_ANALOGFILTER_ENABLE) != HAL_OK)
379     {
380         Error_Handler();
381     }
382
383     /** Configure Digital filter
384     */
385     if (HAL_I2CEx_ConfigDigitalFilter(&hi2c2, 0) != HAL_OK)
386     {
387         Error_Handler();
388     }
389     /* USER CODE BEGIN I2C2_Init 2 */
390
391     /* USER CODE END I2C2_Init 2 */
392
393 }
394
395 /**
396  * @brief RTC Initialization Function
397  * @param None
398  * @retval None
399  */
400 static void MX_RTC_Init(void)
401 {
402
403     /* USER CODE BEGIN RTC_Init 0 */
404
405     /* USER CODE END RTC_Init 0 */
406
407     /* USER CODE BEGIN RTC_Init 1 */
408
409     /* USER CODE END RTC_Init 1 */
410
411     /** Initialize RTC Only
412     */
413     hrtc.Instance = RTC;
414     hrtc.Init.HourFormat = RTC_HOURFORMAT_24;
415     hrtc.Init.AsynchPrediv = 127;
416     hrtc.Init.SynchPrediv = 255;
417     hrtc.Init.OutPut = RTC_OUTPUT_DISABLE;
418     hrtc.Init.OutPutPolarity = RTC_OUTPUT_POLARITY_HIGH;
419     hrtc.Init.OutPutType = RTC_OUTPUT_TYPE_OPENDRAIN;
420     if (HAL_RTC_Init(&hrtc) != HAL_OK)
421     {
422         Error_Handler();
423     }
424     /* USER CODE BEGIN RTC_Init 2 */
425
426     /* USER CODE END RTC_Init 2 */
427
428 }
429
430 /**
431  * @brief SPI1 Initialization Function
432  * @param None
433  * @retval None
434  */
435 static void MX_SPI1_Init(void)
436 {
437
438     /* USER CODE BEGIN SPI1_Init 0 */
439
440     /* USER CODE END SPI1_Init 0 */
441
442     /* USER CODE BEGIN SPI1_Init 1 */
443
444     /* USER CODE END SPI1_Init 1 */
445     /* SPI1 parameter configuration*/
446     hspi1.Instance = SPI1;
447     hspi1.Init.Mode = SPI_MODE_MASTER;
448     hspi1.Init.Direction = SPI_DIRECTION_2LINES;
449     hspi1.Init.DataSize = SPI_DATASIZE_8BIT;
450     hspi1.Init.CLKPolarity = SPI_POLARITY_LOW;
451     hspi1.Init.CLKPhase = SPI_PHASE_1EDGE;
452     hspi1.Init.NSS = SPI_NSS_SOFT;

```

```

453     hspi1.Init.BaudRatePrescaler = SPI_BAUDRATEPRESCALER_2;
454     hspi1.Init.FirstBit = SPI_FIRSTBIT_MSB;
455     hspi1.Init.TIMode = SPI_TIMODE_DISABLE;
456     hspi1.Init.CRCCalculation = SPI_CRCALCULATION_DISABLE;
457     hspi1.Init.CRCPolynomial = 7;
458     hspi1.Init.CRCLength = SPI_CRC_LENGTH_DATASIZE;
459     hspi1.Init.NSSPMode = SPI_NSS_PULSE_ENABLE;
460     if (HAL_SPI_Init(&hspi1) != HAL_OK)
461     {
462         Error_Handler();
463     }
464     /* USER CODE BEGIN SPI1_Init 2 */
465
466     /* USER CODE END SPI1_Init 2 */
467
468 }
469
470 /**
471 * @brief TIM6 Initialization Function
472 * @param None
473 * @retval None
474 */
475 static void MX_TIM6_Init(void)
476 {
477
478     /* USER CODE BEGIN TIM6_Init 0 */
479
480     /* USER CODE END TIM6_Init 0 */
481
482     TIM_MasterConfigTypeDef sMasterConfig = {0};
483
484     /* USER CODE BEGIN TIM6_Init 1 */
485
486     /* USER CODE END TIM6_Init 1 */
487     htim6.Instance = TIM6;
488     htim6.Init.Prescaler = 1;
489     htim6.Init.CounterMode = TIM_COUNTERMODE_UP;
490     htim6.Init.Period = 39999;
491     htim6.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
492     if (HAL_TIM_Base_Init(&htim6) != HAL_OK)
493     {
494         Error_Handler();
495     }
496     sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
497     sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
498     if (HAL_TIMEx_MasterConfigSynchronization(&htim6, &sMasterConfig) != HAL_OK)
499     {
500         Error_Handler();
501     }
502     /* USER CODE BEGIN TIM6_Init 2 */
503
504     /* USER CODE END TIM6_Init 2 */
505
506 }
507
508 /**
509 * @brief USART1 Initialization Function
510 * @param None
511 * @retval None
512 */
513 static void MX_USART1_UART_Init(void)
514 {
515
516     /* USER CODE BEGIN USART1_Init 0 */
517
518     /* USER CODE END USART1_Init 0 */
519
520     /* USER CODE BEGIN USART1_Init 1 */
521
522     /* USER CODE END USART1_Init 1 */
523     huart1.Instance = USART1;
524     huart1.Init.BaudRate = 115200;
525     huart1.Init.WordLength = UART_WORDLENGTH_8B;
526     huart1.Init.StopBits = UART_STOPBITS_1;
527     huart1.Init.Parity = UART_PARITY_NONE;
528     huart1.Init.Mode = UART_MODE_TX_RX;
529     huart1.Init.HwFlowCtl = UART_HWCONTROL_NONE;

```

```

530     huart1.Init.OverSampling = UART_OVERSAMPLING_16;
531     huart1.Init.OneBitSampling = UART_ONE_BIT_SAMPLE_DISABLE;
532     huart1.AdvancedInit.AdvFeatureInit = UART_ADVFEATURE_NO_INIT;
533     if (HAL_UART_Init(&huart1) != HAL_OK)
534     {
535         Error_Handler();
536     }
537     /* USER CODE BEGIN USART1_Init_2 */
538
539     /* USER CODE END USART1_Init_2 */
540
541 }
542
543 /**
544  * @brief GPIO Initialization Function
545  * @param None
546  * @retval None
547  */
548 static void MX_GPIO_Init(void)
549 {
550     GPIO_InitTypeDef GPIO_InitStruct = {0};
551     /* USER CODE BEGIN MX_GPIO_Init_1 */
552     /* USER CODE END MX_GPIO_Init_1 */
553
554     /* GPIO Ports Clock Enable */
555     __HAL_RCC_GPIOC_CLK_ENABLE();
556     __HAL_RCC_GPIOF_CLK_ENABLE();
557     __HAL_RCC_GPIOA_CLK_ENABLE();
558     __HAL_RCC_GPIOB_CLK_ENABLE();
559
560     /*Configure GPIO pin Output Level */
561     HAL_GPIO_WritePin(WKUP_ESP_GPIO_Port, WKUP_ESP_Pin, GPIO_PIN_RESET);
562
563     /*Configure GPIO pin Output Level */
564     HAL_GPIO_WritePin(GPIOA, RST_Pin|DC_Pin|EN_VBAT_Pin, GPIO_PIN_RESET);
565
566     /*Configure GPIO pin Output Level */
567     HAL_GPIO_WritePin(SPI_CS_GPIO_Port, SPI_CS_Pin, GPIO_PIN_SET);
568
569     /*Configure GPIO pin Output Level */
570     HAL_GPIO_WritePin(GPIOB, LED_RED_Pin|ALARME_Pin, GPIO_PIN_RESET);
571
572     /*Configure GPIO pin Output Level */
573     HAL_GPIO_WritePin(ESP_EN_GPIO_Port, ESP_EN_Pin, GPIO_PIN_SET);
574
575     /*Configure GPIO pin : WKUP_ESP_Pin */
576     GPIO_InitStruct.Pin = WKUP_ESP_Pin;
577     GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
578     GPIO_InitStruct.Pull = GPIO_NOPULL;
579     GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
580     HAL_GPIO_Init(WKUP_ESP_GPIO_Port, &GPIO_InitStruct);
581
582     /*Configure GPIO pins : RST_Pin DC_Pin SPI_CS_Pin EN_VBAT_Pin */
583     GPIO_InitStruct.Pin = RST_Pin|DC_Pin|SPI_CS_Pin|EN_VBAT_Pin;
584     GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
585     GPIO_InitStruct.Pull = GPIO_NOPULL;
586     GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
587     HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
588
589     /*Configure GPIO pin : BUSY_Pin */
590     GPIO_InitStruct.Pin = BUSY_Pin;
591     GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
592     GPIO_InitStruct.Pull = GPIO_NOPULL;
593     HAL_GPIO_Init(BUSY_GPIO_Port, &GPIO_InitStruct);
594
595     /*Configure GPIO pins : LED_RED_Pin ALARME_Pin ESP_EN_Pin */
596     GPIO_InitStruct.Pin = LED_RED_Pin|ALARME_Pin|ESP_EN_Pin;
597     GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
598     GPIO_InitStruct.Pull = GPIO_NOPULL;
599     GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
600     HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);
601
602     /* USER CODE BEGIN MX_GPIO_Init_2 */
603     /* USER CODE END MX_GPIO_Init_2 */
604 }
605
606 /* USER CODE BEGIN 4 */

```

```

K:\ES\PROJETS\SLO\2409_MesureTH_RefrigerateurCongelateur\soft\Firmware\STM32\2409_MesureTH_V1\2409_MesureTH

607
608 //-----
609
610     //// Fonction Initialisation
611     //// Description: Logique utilisée pour initialiser le système (initialisation des variables,
612     // calibration ADC, initialisation de la gestion du mode sommeil)
613     //// Entrées: -
614     //// Sorties: -
615     void Initialisation()
616     {
617         mesures.etatSeuils = INDEFINI; // Pour s'assurer qu'un changement d'état de seuils aura bien lieu
618         au lancement du programme
619
620         infoBatterie.etatBatterie = INDEFINI; // Pour s'assurer qu'un changement d'état de batterie aura
621         bien lieu au lancement du programme
622         infoBatterie.compteurCheckBatterie = 0; // Pour s'assurer qu'une vérification de l'état de la
623         batterie ait lieu au lancement du programme
624
625         HAL_ADCEx_Calibration_Start(&hadc); // Calibration ADC (utilisé pour vérifier l'état de la batterie)
626
627         InitialiserGestionSommeil();
628
629
630     //// Fonction HAL_TIM_PeriodElapsedCallback: interruption Timer
631     //// Description: fonction d'interruptioin du timer (délai entre activation transistor et mesure de
632     // l'état de la batterie)
633     //// Entrées: Pointeur:TIM_HandleTypeDef htim
634     //// Sorties: -
635     void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
636     {
637         // Decompte du compteur utilisé pour le délai entre l'activation du transistor et la mesure de
638         l'état de la batterie
639         if(etatSTM == MESURER_BATTERIE)
640         {
641             if(infoBatterie.compteurCheckBatterie > 0)
642             {
643                 infoBatterie.compteurCheckBatterie--;
644
645             // Quand le compteur vaudra 0
646             // Ce sera le moment de mesurer la tension de la batterie (à la prochaine exécution de la boucle
647             main)
648         }
649     }
650
651
652     //// Fonction EXTI4_15_IRQHandler: interruption pins GPIO
653     //// Description: fonction d'interruptioin de la pin utilisée par l'ESP pour réveiller le STM
654     //// Entrées: -
655     //// Sorties: -
656     void EXTI4_15_IRQHandler(void)
657     {
658         if(HAL_GPIO_ReadPin(REVEIL_GPIO_Port, REVEIL_Pin) == GPIO_PIN_SET)
659         {
660             receptionTramesEspTermine = true; // Indique que l'ESP à fini d'envoyer ses éventuelles trames
661             UART
662
663             // Pour s'assurer que l'état de l'interruption soit bien réinitialisé
664             HAL_NVIC_ClearPendingIRQ(EXTI4_15_IRQn);
665             __HAL_GPIO_EXTI_CLEAR_FLAG(GPIO_PIN_5);
666         }
667
668     /* USER CODE END 4 */
669
670     /**
671      * @brief This function is executed in case of error occurrence.
672      * @retval None
673      */
674     void Error_Handler(void)
675     {
676         /* USER CODE BEGIN Error_Handler_Debug */
677         /* User can add his own implementation to report the HAL error return state */
678         __disable_irq();
679         while (1)
680         {
681         /* USER CODE END Error_Handler_Debug */

```

```
C:\ES\PROJETS\SLO\2409_MesureTH_RefrigerateurCongelateur\soft\Firmware\STM32\2409_MesureTH_V1\2409_MesureTH

676 }
677
678 #ifdef USE_FULL_ASSERT
679 /**
680  * @brief Reports the name of the source file and the source line number
681  * where the assert_param error has occurred.
682  * @param file: pointer to the source file name
683  * @param line: assert_param error line source number
684  * @retval None
685 */
686 void assert_failed(uint8_t *file, uint32_t line)
687 {
688  /* USER CODE BEGIN 6 */
689  /* User can add his own implementation to report the file name and line number,
690  * ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
691  /* USER CODE END 6 */
692 }
693#endif /* USE_FULL_ASSERT */
694
```

```

1  /* USER CODE BEGIN Header */
2  /**
3   * @file          : main.h
4   * @brief         : Header for main.c file.
5   *                   This file contains the common defines of the application.
6   *
7   * @attention
8   *
9   * Copyright (c) 2024 STMicroelectronics.
10  * All rights reserved.
11  *
12  * This software is licensed under terms that can be found in the LICENSE file
13  * in the root directory of this software component.
14  * If no LICENSE file comes with this software, it is provided AS-IS.
15  *
16  */
17  ****
18  */
19  /* USER CODE END Header */
20
21 /* Define to prevent recursive inclusion -----*/
22 #ifndef __MAIN_H
23 #define __MAIN_H
24
25 #ifdef __cplusplus
26 extern "C" {
27 #endif
28
29 /* Includes -----*/
30 #include "stm32f0xx_hal.h"
31
32 /* Private includes -----*/
33 /* USER CODE BEGIN Includes */
34 #include <stdbool.h>
35 /* USER CODE END Includes */
36
37 /* Exported types -----*/
38 /* USER CODE BEGIN ET */
39
40 typedef struct {
41     char* nom;
42     double valeur;
43 } DefinitionValeur;
44
45 typedef enum {
46     SEUIL_TEMPERATURE_MIN,
47     SEUIL_TEMPERATURE_MAX,
48     ECART_TEMPERATURE,
49     SEUIL_HUMIDITE_MIN,
50     SEUIL_HUMIDITE_MAX,
51     ECART_HUMIDITE,
52 } ValeursServeur;
53
54 typedef enum {
55     ETAT_SEUILS,
56     ETAT_BATTERIE,
57 } ValeursSTM;
58
59 typedef enum {
60     OK,           // 0
61     ALARME,        // 1
62     INDEFINI,      // 2
63 } Etab;
64
65 typedef enum {
66     INITIALISATION,
67     VERIFIER_BATTERIE,
68     MESURER_BATTERIE,
69     RECEPTION,
70     MESURER_SONDE,
71     RAFRAICHE_EPAPER,
72     SOMMEIL,
73     REVEIL,
74 } EtabSTM;
75
76 extern bool rafraichirEPaper;
77

```

## Annexe S.16

```

78  /* USER CODE END ET */
79
80  /* Exported constants -----*/
81  /* USER CODE BEGIN EC */
82
83  /* USER CODE END EC */
84
85  /* Exported macro -----*/
86  /* USER CODE BEGIN EM */
87
88  /* USER CODE END EM */
89
90  /* Exported functions prototypes -----*/
91  void Error_Handler(void);
92
93  /* USER CODE BEGIN EFP */
94  void Initialisation(void);
95
96  /* USER CODE END EFP */
97
98  /* Private defines -----*/
99  #define WKUP_ESP_Pin GPIO_PIN_13
100 #define WKUP_ESP_GPIO_Port GPIOC
101 #define RST_Pin GPIO_PIN_1
102 #define RST_GPIO_Port GPIOA
103 #define DC_Pin GPIO_PIN_2
104 #define DC_GPIO_Port GPIOA
105 #define BUSY_Pin GPIO_PIN_3
106 #define BUSY_GPIO_Port GPIOA
107 #define SPI_CS_Pin GPIO_PIN_4
108 #define SPI_CS_GPIO_Port GPIOA
109 #define VAL_VBAT_Pin GPIO_PIN_6
110 #define VAL_VBAT_GPIO_Port GPIOA
111 #define LED_RED_Pin GPIO_PIN_14
112 #define LED_RED_GPIO_Port GPIOB
113 #define EN_VBAT_Pin GPIO_PIN_10
114 #define EN_VBAT_GPIO_Port GPIOA
115 #define ALARME_Pin GPIO_PIN_3
116 #define ALARME_GPIO_Port GPIOB
117 #define ESP_EN_Pin GPIO_PIN_8
118 #define ESP_EN_GPIO_Port GPIOB
119
120  /* USER CODE BEGIN Private defines */
121  #define DIN_Pin GPIO_PIN_7
122  #define DIN_GPIO_Port GPIOA
123  #define SCK_Pin GPIO_PIN_5
124  #define SCK_GPIO_Port GPIOA
125
126  #define REVEIL_Pin GPIO_PIN_5
127  #define REVEIL_GPIO_Port GPIOB
128
129  /* USER CODE END Private defines */
130
131  #ifdef __cplusplus
132  }
133  #endif
134
135  #endif /* __MAIN_H */
136

```

# Annexe T.1

```
1 // Notes
2 // Comment afficher des variables de type String dans des printf:
https://stackoverflow.com/questions/66555803/serial-printf-with-string-object-of-which-length-is-11-or-more
3 // Autres fichiers: les autres fichiers .ino sont intégrés par ordre alphabétique.
4 Utilisation de .h pour avoir plus de contrôles pour les variables et les defines
4 // Les prototypes ne doivent pas être déclarés dans les .h, cela crée des conflits
4 avec les prototypes générés automatiquement par Arduino IDE
5
6
7 // Les fichiers .h créés à part doivent absolument être ajoutés dans le fichier
7 principal !
8 // https://forum.arduino.cc/t/variable-or-field-declared-void/687410/11
9 #include "CommunicationServeur.h"
10 #include "CommunicationSTM.h"
11 #include "Discord.h"
12 #include "Execution.h"
13 #include "Sommeil.h"
14 #include "Wifi.h"
15
16
17 // Quand le mode debug est activé, affiche de nombreux messages dans le moniteur
17 série pour pouvoir débugger plus facilement
18 // Attention: les printf sont également envoyés avec les données UART au STM, ce qui
18 empêche le STM de bien recevoir les vraies trames UART envoyées
19 #define MODE_DEBUG false
20
21
22 ///// Fonction setup: point d'entrée du code (fonction créée automatiquement et appelé
22 au lancement ou au réveil de l'ESP)
23 ///// Description: configuration de la communication UAR, configuration du Wifi, appel
23 de la fonction GestionReveil
24 ///// Entrées: -
25 ///// Sorties: -
26 void setup() {
27
28     Serial.begin(115200); // Démarre le serial
29
30     ConfigurerUART();
31
32     ConfigurerWifi();
33
34     GestionReveil();
35 }
36
37
38 ///// Fonction loop: fonction d'execution (fonction créée automatiquement et appelée
38 en continu lors de l'exécution)
39 ///// Description: fonction vide, car la logique d'exécution (fonctions
39 ExecutionReveilTimer ou ExecutionReveilSTM) ne s'exécute qu'une seule fois avant de
39 repasser en mode sommeil profond
40 ///// Entrées: -
41 ///// Sorties: -
42 void loop() {
43 }
```

## Annexe T.2

```
1  /* **** */
2  /** Descriptive File Name
3
4      @Company
5          ETML-ES
6
7      @File Name
8          CommunicationServeur.h
9
10     @Auteurs
11         - Perret Mélissa
12
13     @Description
14         Fonctions liées à la communication entre l'ESP et le serveur
15     */
16  /* **** */
17
18
19 #ifndef _COMMUNICATION_SERVEUR_H
20 #define _COMMUNICATION_SERVEUR_H
21
22 #define NOMBRE_TENTATIVES_REQUTES 10 // Combien de fois on essaye d'effectuer les
23 requêtes HTTP (car des fois on obtient un read timeout sur le réseau de l'école)
24 #define HTTP_TIMEOUT_MS 10000 // Combien de fois on essaye d'effectuer les requêtes
25 HTTP (car des fois on obtient un read timeout sur le réseau de l'école)
26 #define DELAI_NOUVELLE_TENTATIVE_HTTP_MS 2000 // Délai avant de réessayer une requête
27 HTTP
28
29 // Tableau ASCII: https://ss64.com/ascii.html
30 const uint8_t OCTET_DEBUT = 0x02; // Caractère STX (Start of Text) utilisé pour
31 identifier les débuts de trames
32 const uint8_t OCTET_FIN = 0x03; // Caractère ETX (End of Text) utilisé pour
33 identifier les fins de trames
34
35 // Trame: OCTET_DEBUT + index + ':' + valeur + OCTET_FIN
36 const uint8_t TAILLE_TRAME = sizeof(OCTET_DEBUT) + sizeof(uint8_t) + sizeof(char) +
37 sizeof(double) + sizeof(OCTET_FIN);
38
39 const char separateur = ':'; // Séparateur entre l'index et la valeur dans les
40 trames, pour faciliter le debuggage
41
42 // Rappel format des trames :
43 // trame[0] = OCTET_DEBUT;
44 // trame[1] = index;
45 // trame[2] = ':';
46 // trame[3] à trame[10] = valeur;
47 // trame[11] = OCTET_FIN;
48 uint8_t trame[TAILLE_TRAME]; // Tableau pour envoyer les trames
49
50
51 // https://www.arduino.cc/reference/en/language/variables/data-types/array/
52 // Utilisation d'un tableau de String et d'un tableau de double séparés (au lieu
53 d'utiliser un tableau de DefinitionValeur) car les valeurs String ne peuvent pas être
54 sauvegardées dans la RTC
55 String nomValeursServeur[6]{
56     "seuilTemperatureMin",
57     "seuilTemperatureMax",
58     "ecartTemperature",
59     "seuilHumiditeMin",
60     "seuilHumiditeMax",
61     "ecartHumidite",
62 };
63
64
65 // Note : ne pas définir les valeurs initiales directement ici pour éviter de
66 // réinitialiser les valeurs à chaque réveil
67 // les valeurs sont initialisées dans la fonction GestionReveil uniquement
68 lorsqu'on vient de démarrer l'ESP
69 RTC_DATA_ATTR double valeursServeur[6];
70
71
72 #endif /* _COMMUNICATION_SERVEUR_H */
```

## Annexe T.3

```
1 // CommunicationServeur.ino
2 //
3 // Description : fonctions liées à la communication entre l'ESP et le serveur
4 // Auteur : Perret Mélissa
5 // Création : 22/09/2024
6 // Modifications : --
7
8 // Version      : V1.0
9 /*-----*/
10
11
12 #include "CommunicationServeur.h"
13 #include <Arduino_JSON.h> // Librairie pour manipuler et traiter les données JSON
14 // reçues par le serveur
15 #include <HTTPClient.h>    // pour HTTPClient
16
17 ///// Fonction ReceptionServeur: logique pour recevoir les données de la part du STM
18 ///// Description: envoie d'une requête HTTP Get au serveur
19 /////           réception de la réponse, analyse de la réponse pour extraire les
20 /////           données reçues (seuils min/max et écarts pour température et humidité)
21 /////           appel de la fonction TraiterValeurServeur pour traiter les données
22 /////           reçues
23 ///// Entrées: Pointeur:HTTPClient http (client http utilisé pour effectuer les
24 /////           requêtes auprès du serveur)
25 ///// Sorties: -
26 void ReceptionServeur(HTTPClient* http) {
27
28     if (MODE_DEBUG) {
29         printf("[HTTP] Requete GET...\n");
30     }
31
32     http->setTimeout(HTTP_TIMEOUT_MS);
33
34     do {
35
36         if(nombreTentativesHTTP > 0)
37         {
38             delay(DELAIS_NOUVELLE_TENTATIVE_HTTP_MS); // S'il s'agit d'une nouvelle
39             // tentative, attendre un peu avant de réessayer
40         }
41
42         nombreTentativesHTTP++;
43
44         // Démarrer la connexion et envoyer la requête GET
45         int httpCode = http->GET();
46
47         // HttpStatusCode est négatif en cas d'erreur
48         if (httpCode > 0) {
49             // Le header HTTP à été envoyé et une réponse du serveur à été reçue
50             if (MODE_DEBUG) {
51                 printf("[HTTP] Requete GET... code: %d\n", httpCode);
52             }
53
54             if (httpCode == HTTP_CODE_OK) {
55                 String reponseServeur = http->getString(); // Obtenir la réponse du serveur
56
57                 if (MODE_DEBUG) {
58                     printf("[HTTP] Requete GET reponse serveur: %s\n", reponseServeur.c_str());
59                 }
60
61                 // Parsing JSON
62                 JSONVar myObject = JSON.parse(reponseServeur);
63                 JSONVar keys = myObject.keys();
64
65                 // Traitement des valeurs
66                 for (int i = 0; i < keys.length(); i++) {
67                     String nom = keys[i];
68                     double valeur = std::atof(myObject[keys[i]]); // utilisation de la
69                     // fonction atof pour convertir de string en double
```

```

68         TraiterValeurServeur(nom, valeur);
69     }
70 }
71 } else {
72     printf("[HTTP] GET... echec, erreur: %s\n", http->errorToString(httpCode).c_str());
73 }
74 } while (httpCode <= 0 && nombreTentativesHTTP < NOMBRE_TENTATIVES_REQUESTS);
75 }
76
77 // Fonction TraiterValeurServeur: logique pour traiter une donnée reçue par le
78 // serveur
79 // Description: vérifier si la valeur à changé
80 //                mettre à jour la valeur stockée dans la RTC
81 //                construire et transmettre la trame UART au STM
82 // Entrées: String nom (nom du paramètre modifié)
83 //           double valeur (valeur du paramètre modifié)
84 // Sorties: -
85 void TraiterValeurServeur(String nom, double valeur) {
86     int8_t index = IndexValeurServeur(nom); // Récupérer index du paramètre modifié
87     (-1 si non trouvé)
88     if (index == -1) {
89         printf("Nouvelle valeur provenant du serveur. Impossible de trouver le nom %s
90         dans nomValeursServeur\n", nom.c_str());
91     } else if (valeur != valeursServeur[index]) { // Si la valeur n'a pas changé,
92         inutile d'en informer le STM
93
94         if (MODE_DEBUG) {
95             printf("Nouvelle valeur provenant du serveur. Nom:%s Avant:%lf Maintenant:%lf\n"
96                   , nomValeursServeur[index].c_str(), valeursServeur[index], valeur);
97         }
98
99         valeursServeur[index] = valeur; // Mise à jour de la valeur stockée dans la RTC
100
101        // Rappel format des trames :
102        // double trame[] = {0,0,0,0,0} // 40 octets
103        // trame[0] = OCTET_DEBUT;
104        // trame[1] = index
105        // trame[2] = ':';
106        // trame[3] à trame[10] = valeur;
107        // trame[11] = OCTET_FIN;
108
109        // Explications memset : https://cplusplus.com/reference/cstring/memset/
110        memset(trame, 0, sizeof(trame)); // Réinitialisation de la trame en mettant tous
111        les bits à 0 (par précautions)
112
113        // Explications memcpy : https://en.cppreference.com/w/cpp/string/byte	memcpy
114        memcpy(&trame[0], &OCTET_DEBUT, sizeof(OCTET_DEBUT));
115
116        // OCTET_DEBUT
117        memcpy(&trame[sizeof(OCTET_DEBUT)], &index, sizeof(index)); // index
118
119        // séparateur
120        // ':'
121        memcpy(&trame[sizeof(OCTET_DEBUT) + sizeof(index)], &separateur, sizeof(separateur));
122        // valeur
123        memcpy(&trame[sizeof(OCTET_DEBUT) + sizeof(index) + sizeof(separateur)], &valeur,
124        sizeof(double)); // valeur
125        // OCTET_FIN
126        memcpy(&trame[sizeof(OCTET_DEBUT) + sizeof(index) + sizeof(separateur) + sizeof(
127        valeur)], &OCTET_FIN, sizeof(OCTET_FIN)); // OCTET_FIN
128
129        int writeResult = uart_write_bytes(uart_num, trame, sizeof(trame)); // //
130        Transmission de la trame au STM
131
132        if (MODE_DEBUG) {
133            printf("\nEnvoie de %d octets en UART\n", writeResult);
134        }
135    }
136
137    // Fonction IndexValeurServeur: logique pour retrouver depuis le nom d'un paramètre
138    // son index dans le tableau des valeurs
139    // Description: parcourir le tableau nomValeursServeur pour trouver l'index où la
140    // valeur correspond au nom indiqué en paramètre d'entrée
141    // Entrées: String nom (nom du paramètre recherché)

```

```
125 ///// Sorties: int8_t (index du paramètre dans le tableau des valeurs stockées, -1 si
126 non trouvé)
127 int8_t IndexValeurServeur(String nom) {
128     int tailleTableau = sizeof(valeursServeur) / sizeof(valeursServeur[0]); // On
129     recalcule la taille du tableau (pour plus d'informations sur la fonction sizeof :
130     https://www.arduino.cc/reference/en/language/variables/utilities/sizeof/)
131     for (int i = 0; i < tailleTableau; i++) {
132         if (nomValeursServeur[i] == nom) {
133             return i; // Nom trouvé
134         }
135     }
136     return -1; // Nom introuvable
137 }
```

# Annexe T.4

```
1  /* **** Descriptive File Name **** */
2  /** Descriptive File Name
3
4      @Company
5          EML-ES
6
7
8      @File Name
9          CommunicationSTM.h
10
11     @Auteurs
12         - Perret Mélissa
13
14     @Description
15         Fonctions liées à la communication entre l'ESP et le STM
16     */
17  /* **** Descriptive File Name **** */
18
19
20 #ifndef _COMMUNICATION_STM_H
21 #define _COMMUNICATION_STM_H
22
23
24 #include "driver/uart.h"    // pour UART_NUM_0
25 #include <HTTPClient.h>    // pour HTTPClient
26
27
28 #define TXD_PIN (GPIO_NUM_7)    // PIN utilisée pour écrire les trames UART
29 #define RXD_PIN (GPIO_NUM_6)    // PIN utilisée pour lire les trames UART
30
31 #define PIN_POUR_REVEILLER_STM GPIO_NUM_5    // PIN utilisée pour réveiller le STM
32      (état haut) et pour indiquer quand l'ESP à fini de transmettre ses trames UART (état bas)
33
34 #define UART_RECEPTION_TIMEOUT_MS 100    // Timeout lors de la réception UART
35
36 const uart_port_t uart_num = UART_NUM_0;        // UART utilisé
37 const size_t TAILLE_BUFFER = 50;                // Taille du tableau utilisé pour
38 stocker les données UART reçues
39 uint8_t receptionTramesUART[TAILLE_BUFFER];    // Tableau pour stocker les données UART
40      reçues
41
42 // Enumération pour représenter le type de valeur reçue par le STM et améliorer la
43 lisibilité du code
44 typedef enum {
45     ETAT_SEUILS,
46     ETAT_BATTERIE,
47 } ValeursSTM;
48
49 // Enumération pour représenter les états reçus par le STM (alarme seuils ou
50 batterie) et améliorer la lisibilité du code
51 typedef enum {
52     FONCTIONEL,    // 0
53     ALARME,        // 1
54     IDENTIQUE,     // 2
55 } Etat;
56
57 // Structure pour stocker les valeurs liées au STM (alarme seuils et etat batterie)
58 typedef struct {
59     String nom;
60     double valeur;
61 } ValeurSTM;
62
63 // Valeurs liées au STM (alarme seuils et etat batterie)
64 ValeurSTM valeursSTM[] = {
65     { "alarme", -9999 },
66     { "pilesFaibles", -9999 },
67 };
68
69
70 #endif /* _COMMUNICATION_STM_H */
```

# Annexe T.5

```
1 // CommunicationSTM.ino
2 //
3 // Description : fonctions liées à la communication entre l'ESP et le STM
4 // Auteur : Perret Mélissa
5 // Création : 22/09/2024
6 // Modifications : --
7
8 // Version      : V1.0
9 /*-----*/
10
11
12 #include "CommunicationSTM.h"
13 #include "CommunicationServeur.h" // pour OCTET_DEBUT et OCTET_FIN
14 #include <HTTPClient.h>          // pour HTTPClient
15
16
17 ///// Fonction ConfigurerUART: configuration de la transmission UART
18 ///// Description: défini les pins utilisées pour l'écriture et la lecture UART
19 ///// Entrées: -
20 ///// Sorties: -
21 void ConfigurerUART() {
22     uart_config_t uart_config = {
23         .baud_rate = 115200,
24         .data_bits = UART_DATA_8_BITS,
25         .parity = UART_PARITY_DISABLE,
26         .stop_bits = UART_STOP_BITS_1,
27         .flow_ctrl = UART_HW_FLOWCTRL_DISABLE,
28     };
29
30     ESP_ERROR_CHECK(uart_param_config(uart_num, &uart_config));           // Configuration UART
31     ESP_ERROR_CHECK(uart_set_pin(uart_num, TXD_PIN, RXD_PIN, -1, -1)); // Indiquer les pins utilisées pour l'écriture et la lecture UART
32 }
33
34 ///// Fonction ReveillerSTM: logique pour réveiller le STM
35 ///// Description: change l'état de la pin utilisée pour réveiller le STM
36 ///// Entrées: -
37 ///// Sorties: -
38 void ReveillerSTM() {
39     pinMode(PIN_POUR_REVEILLER_STM, OUTPUT); // Important de mettre la pin en mode OUTPUT pour pouvoir modifier sa valeur
40     digitalWrite(PIN_POUR_REVEILLER_STM, HIGH);
41
42     if (MODE_DEBUG) {
43         printf("Reveil du STM en mettant la pin %d à l'état %d\n", PIN_POUR_REVEILLER_STM,
44             digitalRead(PIN_POUR_REVEILLER_STM));
45     }
46 }
47
48 ///// Fonction EnvoyerSignalFinSTM: logique pour indiquer au STM que l'ESP a fini de lui transmettre des trames UART
49 ///// Description: met la pin utilisée pour réveiller le STM à l'état bas (la pin à l'état haut réveille le STM, et l'état bas lui indique qu'il a reçu toutes les trames nécessaires de la part de l'ESP)
50 ///// Entrées: -
51 ///// Sorties: -
52 void EnvoyerSignalFinSTM() {
53     pinMode(PIN_POUR_REVEILLER_STM, OUTPUT); // Important de mettre la pin en mode OUTPUT pour pouvoir modifier sa valeur
54     digitalWrite(PIN_POUR_REVEILLER_STM, LOW);
55
56     if (MODE_DEBUG) {
57         printf("Envoie signal fin transmission au STM en mettant la pin %d à l'état %d\n",
58             PIN_POUR_REVEILLER_STM, digitalRead(PIN_POUR_REVEILLER_STM));
59     }
60
61 ///// Fonction ReceptionSTM: logique pour recevoir les trames UART de la part du STM
62 ///// Description: réception des données UART
63 /////               analyse des données pour localiser les trames (caractères de début et de fin)
64 /////               analyse des trames reçues pour extraire et traiter les données
```

```

réçues
64      /////
65          envoie de requête HTTP PUT pour transmettre les nouvelles données
66          au serveur (états alarme seuils et batterie)
67  ///////////////////////////////////////////////////////////////////
68  // Entrées: Pointeur:HTTPClient http (client http utilisé pour effectuer les
69  // requêtes auprès du serveur)
70  ///////////////////////////////////////////////////////////////////
71  ///////////////////////////////////////////////////////////////////
72  ///////////////////////////////////////////////////////////////////
73 ///////////////////////////////////////////////////////////////////
74  ///////////////////////////////////////////////////////////////////
75 ///////////////////////////////////////////////////////////////////
76  ///////////////////////////////////////////////////////////////////
77  ///////////////////////////////////////////////////////////////////
78  ///////////////////////////////////////////////////////////////////
79 ///////////////////////////////////////////////////////////////////
80  ///////////////////////////////////////////////////////////////////
81  ///////////////////////////////////////////////////////////////////
82  ///////////////////////////////////////////////////////////////////
83  ///////////////////////////////////////////////////////////////////
84  ///////////////////////////////////////////////////////////////////
85  ///////////////////////////////////////////////////////////////////
86  ///////////////////////////////////////////////////////////////////
87  ///////////////////////////////////////////////////////////////////
88  ///////////////////////////////////////////////////////////////////
89  ///////////////////////////////////////////////////////////////////
90  ///////////////////////////////////////////////////////////////////
91  ///////////////////////////////////////////////////////////////////
92 ///////////////////////////////////////////////////////////////////
93  ///////////////////////////////////////////////////////////////////
94  ///////////////////////////////////////////////////////////////////
95  ///////////////////////////////////////////////////////////////////
96  ///////////////////////////////////////////////////////////////////
97  ///////////////////////////////////////////////////////////////////
98  ///////////////////////////////////////////////////////////////////
99  ///////////////////////////////////////////////////////////////////
100 ///////////////////////////////////////////////////////////////////
101 ///////////////////////////////////////////////////////////////////
102 ///////////////////////////////////////////////////////////////////
103 ///////////////////////////////////////////////////////////////////
104 ///////////////////////////////////////////////////////////////////
105 ///////////////////////////////////////////////////////////////////
106 ///////////////////////////////////////////////////////////////////
107 ///////////////////////////////////////////////////////////////////
108 ///////////////////////////////////////////////////////////////////
109 ///////////////////////////////////////////////////////////////////
110 ///////////////////////////////////////////////////////////////////
111 ///////////////////////////////////////////////////////////////////
112 ///////////////////////////////////////////////////////////////////
113 ///////////////////////////////////////////////////////////////////
114 ///////////////////////////////////////////////////////////////////
115 ///////////////////////////////////////////////////////////////////
116 ///////////////////////////////////////////////////////////////////
117 ///////////////////////////////////////////////////////////////////
118 ///////////////////////////////////////////////////////////////////
119 ///////////////////////////////////////////////////////////////////
120 ///////////////////////////////////////////////////////////////////

```

// Réception UART

```

ESP_ERROR_CHECK(uart_get_buffered_data_len(uart_num, (size_t*)&length));
length = uart_read_bytes(uart_num, receptionTramesUART, length,
                          UART_RECEPTION_TIMEOUT_MS);

if (length > 0) { // Si on a reçu quelque chose

    // On parcourt le buffer pour trouver toutes les trames (identifiées par
    OCTET_DEBUT)

    for (int indexDebutTrame = 0; indexDebutTrame < length; indexDebutTrame++) {
        if (receptionTramesUART[indexDebutTrame] == OCTET_DEBUT) { // Début de trame
            trouvé

            int indexFinTrame = indexDebutTrame + TAILLE_TRAME - 1;
            if (indexFinTrame < length && receptionTramesUART[indexFinTrame] ==
                OCTET_FIN) // Si on a bien une fin de trame comme attendu
            {
                uint8_t index = receptionTramesUART[indexDebutTrame + 1]; // Deuxième
                octet de la trame correspond à l'identifiant de la trame
                // Troisième octet correspond à un séparateur ':' pour faciliter la
                lecture des trames lors du débogage

                // Explications pour memcpy :
                https://en.cppreference.com/w/cpp/string/byte/memcpy
                // Les 8 octets suivants (octets 4 à 11) correspondent à la valeur de la
                trame
                double valeur;
                memcpy(&valeur, &receptionTramesUART[indexDebutTrame + 3], sizeof(double));
                // Copier les 8 octets de la trame représentant la valeur dans la
                variable valeur

                if (MODE_DEBUG) {
                    printf("Nouvelle valeur provenant du STM. Nom:%s Avant:%lf
                           Maintenant:%lf\n", valeursSTM[index].nom.c_str(), valeursSTM[index].
                           valeur, valeur);
                }

                valeursSTM[index].valeur = valeur; // Mise à jour de la valeur

                // Envoie de la requête PUT pour transmettre la nouvelle valeur au serveur
                String putString = valeursSTM[index].nom + "=" + valeur;

                http->setTimeout(HTTP_TIMEOUT_MS);

                int httpCode;
                int nombreTentativesHTTP = 0;
                do {

                    if(nombreTentativesHTTP > 0)
                    {
                        delay(DELAI_NOUVELLE_TENTATIVE_HTTP_MS); // S'il s'agit d'une
                        nouvelle tentative, attendre un peu avant de réessayer
                    }

                    nombreTentativesHTTP++;

                    httpCode = http->PUT(putString);

                    // httpCode est négatif en cas d'erreur
                    if (httpCode > 0) {
                        // Le header HTTP a été envoyé et une réponse du serveur a été reçue

                        if (MODE_DEBUG) {
                            printf("[HTTP] Requête PUT... code: %d\n", httpCode);

```

```
121     }
122
123     if (httpCode == HTTP_CODE_OK) {
124         String reponseServeur = http->getString();
125         if (MODE_DEBUG) {
126             printf("[HTTP] Requete PUT reponse serveur: %s\n", reponseServeur.
127                   c_str());
128         }
129     } else {
130         printf("[HTTP] Requete PUT... echec, erreur: %s\n", http->
131               errorToString(httpCode).c_str());
132     }
133     } while (httpCode <= 0 && nombreTentativesHTTP <
134           NOMBRE_TENTATIVES_REQUTES);
135   }
136 }
137 } while (length > 0); // Continuer tant qu'on reçoit quelque chose
138 }
```

# Annexe T.6

```
1  /* **** */
2  /** Descriptive File Name
3
4  @Company
5      ETML-ES
6
7  @File Name
8      Discord.h
9
10 @Auteurs
11     - Perret Mélissa
12
13 @Description
14     Fonctions liées à l'envoi de notifications Discoprd
15 */
16 /* **** */
17
18
19 #ifndef _DISCORD_H
20 #define _DISCORD_H
21
22
23 // URL pour envoyer les notifications Discord
24 // Comment récupérer l'URL du webhook:
25 https://support.discord.com/hc/en-us/articles/228383668-Intro-to-Webhooks
26 #define DISCORD_WEBHOOK_URL
27 "https://discord.com/api/webhooks/1287209194017128478/7BPCUgn1GDFLxNcpws1uXw25Ld2WCcd67s2fNQfORzKD3JV0i-wap5MRjXm4DbqWoZMb"
28 #endif /* _DISCORD_H */
```

# Annexe T.7

```
1 // Discord.ino
2 //
3 // Description : fonctions liées à l'envoi de notifications Discoprd
4 // Auteur : Perret Mélissa
5 // Création : 22/09/2024
6 // Modifications : --
7
8 // Version      : V1.0
9 /*-----*/
10
11 #include "Discord.h"
12 #include <WiFiClientSecure.h> // pour envoyer notifications
13 #include <HTTPClient.h>       // pour HTTPClient
14
15
16 // Code provenant initialement de la librairie Discord_WebHook : //
17 // https://github.com/usini/usini_discord_webhook/tree/main
18 // La librairie ne fonctionnait pas (erreur de compilation lié à un mauvais #include)
19 // Pour régler le problème, le code principal de la librairie a été directement
20 // intégré dans le projet
21 // Les fichiers originaux de la librairie (ainsi que la licence) sont inclus dans le
22 // dossier du projet.
23
24
25 ///// Fonction EnvoyerNotificationsDiscord: gère l'envoie des notifications Discord
26 // (état batterie et état seuils alarme)
27 ///// Description: gère l'envoie des notifications Discord (état batterie et état
28 // seuils alarme)
29 ///// Entrées: Pointeur:HTTPClient http
30 ///// Sorties: -
31 void EnvoyerNotificationsDiscord(HTTPClient* http) {
32
33     // Traitement état batterie
34     switch ((Etat)valeursSTM[ETAT_BATTERIE].valeur) {
35         case FONCTIONNEL:
36             EnvoyerNotificationDiscord(http, false, "Charge OK");
37             break;
38         case ALARME:
39             EnvoyerNotificationDiscord(http, false, "Charge faible !");
40             break;
41         case IDENTIQUE:
42             break;
43     }
44
45     // Traitement alarme seuils
46     switch ((Etat)valeursSTM[ETAT_SEUILS].valeur) {
47         case FONCTIONNEL:
48             EnvoyerNotificationDiscord(http, false, "Seuils OK");
49             break;
50         case ALARME:
51             EnvoyerNotificationDiscord(http, false, "Alarme seuils !");
52             break;
53         case IDENTIQUE:
54             break;
55     }
56
57 ///// Fonction EnvoyerNotificationDiscord: gère l'envoie d'une notifications Discord
58 // (text passé en paramètre d'entrée)
59 ///// Description: gère l'envoie des notifications Discord (état batterie et état
60 // seuils alarme)
61 ///// Entrées: Pointeur:HTTPClient http (client http utilisé pour effectuer la requête)
62 /////           bool syntheseVocale (indique si la synthèse vocale pour le message est
63 // activé, pour permettre aux utilisés d'utiliser la synthèse vocale d'entendre le
64 // message).
65 /////           String message (text à envoyer dans Discord)
66 ///// Sorties: bool (indique si la notification a bien été envoyée)
67 bool EnvoyerNotificationDiscord(HTTPClient* http, bool syntheseVocale, String message)
68 {
69
70     String discord_tts = "false";
71     if (syntheseVocale) {
72         discord_tts = "true";
73
74     }
```

```

64 }
65
66 // Utilisation de WiFiClientSecure pour pouvoir créer une requête HTTPS
67 WiFiClientSecure* client = new WiFiClientSecure;
68 bool ok = false;
69 if (client) {
70     client->setInsecure(); // Désactiver la vérification certificat SSL
71
72     if (MODE_DEBUG) {
73         printf("[HTTP] Connexion à Discord...\n");
74         printf("[HTTP] Message: %s\n", message.c_str());
75         printf("[HTTP] Synthèse Vocale: %s\n", discord_tts.c_str());
76     }
77
78     // Débuter requête HTTPS
79     if (http->begin(*client, DISCORD_WEBHOOK_URL)) { // pour indiquer le début de la
80     communication HTTP
81         http->addHeader("Content-Type", "application/json"); // Définir la requête
82         sous forme de JSON
83
84         // Envoyer la requête POST
85         int httpCode = http->POST("{\"content\":\"" + message + "\",\"tts\":\"" +
86         discord_tts + "\"}");
87         if (httpCode > 0) { // Si un code HTTP code est renvoyé
88             if (httpCode == HTTP_CODE_OK || httpCode == HTTP_CODE_MOVED_PERMANENTLY ||
89             httpCode == HTTP_CODE_NO_CONTENT) {
90                 // https://support.discord.com/hc/en-us/articles/228383668-Intro-to-Webhooks
91                 if (MODE_DEBUG) {
92                     printf("[HTTP] OK\n");
93                 }
94                 ok = true;
95             } else {
96                 if (MODE_DEBUG) {
97                     // Erreur
98                     printf("[HTTP] ERREUR: %s\n", http->getString().c_str());
99                     ok = false;
100                }
101            }
102        } else {
103            if (MODE_DEBUG) {
104                // Serveur non accessible
105                printf("[HTTP] ERREUR: %s\n", http->errorToString(httpCode).c_str());
106                ok = false;
107            }
108        }
109    } else {
110        if (MODE_DEBUG) {
111            // Echec de la requête
112            printf("[HTTP] Connexion impossible\n");
113            ok = false;
114        }
115    }
116    http->end(); // pour indiquer la fin de la communication HTTP
117 } else {
118     if (MODE_DEBUG) {
119         printf("[HTTP] Impossible de creer client\n");
120         ok = false;
121     }
122 }
123 delete client;
124 return ok;
125 }
```

## Annexe T.8

```
1 // Execution.ino
2 //
3 // Description : fonctions liées à l'exécution principale de la logique de l'ESP
4 // Auteur : Perret Mélissa
5 // Création : 22/09/2024
6 // Modifications : --
7
8 // Version      : V1.0
9 /*-----*/
10
11
12 #include "Execution.h"
13 #include "Wifi.h"      // pour wifiMulti
14 #include "Sommeil.h"   // pour PIN_REVEIL_PAR_STM
15
16
17 ///// Fonction ExecutionReveilTimer: logique principale de l'ESP lorsqu'il est
18 ///// réveillé par le timer
19 ///// Description: connexion Wifi, réveiller le STM, récupération et transmissions
20 ///// des valeurs du serveur, envoie notification Discord, retour en mode sommeil
21 ///// Entrées: -
22 ///// Sorties: -
23 void ExecutionReveilTimer() {
24
25     // connexion WiFi
26     if ((wifiMulti.run() == WL_CONNECTED)) {
27
28         HTTPClient http;
29
30         if (MODE_DEBUG) {
31             printf("[HTTP] begin...\n");
32         }
33
34         ReveillerSTM();
35         delay(DELAI_TRANSMISSION_STM_MS); // pour s'assurer que le STM soit bien prêt à
36         // recevoir les trames UART
37
38         ReceptionServeur(&http); // réception et traitement des données du serveur
39
40         EnvoyerSignalFinSTM(); // indiquer au STM que l'ESP a fini d'envoyer ses
41         // éventuelles trames UART
42
43         // considérer les valeurs du STM comme non changées avant de recevoir les
44         // éventuels changements de la part du STM
45         valeursSTM[ETAT_SEUILS].valeur = IDENTIQUE;
46         valeursSTM[ETAT_BATTERIE].valeur = IDENTIQUE;
47         ReceptionSTM(&http);
48
49         EnvoyerNotificationsDiscord(&http); // envoyer les éventuelles notifications
50         // Discord en cas de changements de valeurs de la part du STM
51
52         http.end(); // pour indiquer la fin de la communication HTTP
53     }
54
55     delay(DELAI_MODE_SOMMEIL_MS); // délai éventuel avant de passer en mode sommeil
56     // (par précautions)
57
58     EntréeModeSommeilProfond(); // entrer en mode sommeil
59 }
60
61 ///// Fonction ExecutionReveilSTM: logique principale de l'ESP lorsqu'il est réveillé
62 ///// par le STM
63 ///// Description: réception trames STM, envoie notification Discord, retour en mode
64 ///// sommeil
65 ///// Entrées: -
66 ///// Sorties: -
67 void ExecutionReveilSTM() {
68     // connexion WiFi
69     if ((wifiMulti.run() == WL_CONNECTED)) {
```

```

64     HTTPClient http;
65
66     if (MODE_DEBUG) {
67         printf("[HTTP] begin...\n");
68     }
69
70     http.begin(SERVEUR_ADDRESSE_HTTP); // pour indiquer le début de la communication
71     // marquer les valeurs du STM comme non changées avant de recevoir les éventuels
72     // changements de la part du STM
73     valeursSTM[ETAT_SEUILS].valeur = IDENTIQUE;
74     valeursSTM[ETAT_BATTERIE].valeur = IDENTIQUE;
75
76     // procéder à la réception des données UART tant que le STM ne nous indique pas
77     // qu'il a terminé d'envoyer d'éventuelles trames UART
78     do {
79         ReceptionSTM(&http);
80         delay(DELAI_RECEPTION_STM_MS); // Délai entre deux essais
81         // pour recevoir les trames UART attendues de la part du STM
82     } while (digitalRead(PIN_REVEIL_PAR_STM) == HIGH); // Attendre que le STM nous
83     // signale qu'il a terminé d'envoyer les trames UART
84
85     ReceptionSTM(&http); // Important de refaire une nouvelle réception de trames
86     // UART au cas où le STM aurait envoyé des trames juste avant d'indiquer le signal
87     // de fin de communication
88
89     EnvoyerNotificationsDiscord(&http); // envoyer les éventuelles notifications
90     // Discord en cas de changements de valeurs de la part du STM
91
92     http.end(); // pour indiquer la fin de la communication HTTP
93 }
94
95     delay(DELAI_MODE_SOMMEIL_MS); // délai avant de passer en mode sommeil (par
96     // précautions)
97
98     EntréeModeSommeilProfond();
99 }
```

# Annexe T.9

```
1  /* **** */
2  /** Descriptive File Name
3
4      @Company
5          ETML-ES
6
7      @File Name
8          Execution.h
9
10     @Auteurs
11         - Perret Mélissa
12
13     @Description
14         Fonctions liées à l'exécution principale de la logique de l'ESP
15     */
16  /* **** */
17
18
19 #ifndef _EXECUTION_H
20 #define _EXECUTION_H
21
22 #define DELAI_TRANSMISSION_STM_MS 1000 // Délai entre le réveil du STM et l'envoi
23 des trames UART pour s'assurer que le STM soit bien prêt à recevoir les trames UART
24 #define DELAI_RECEPTION_STM_MS 1000 // Délai entre deux essais pour recevoir les
25 trames UART attendues de la part du STM
26 #define DELAI_MODE_SOMMEIL_MS 1000 // Délai avant de passer en mode sommeil (par
27 précautions)
28
29 #endif /* _EXECUTION_H */
```

# Annexe T.10

```
1  /* **** */
2  /** Descriptive File Name
3
4  @Company
5      ETML-ES
6
7  @File Name
8      Sommeil.h
9
10 @Auteurs
11     - Perret Mélissa
12
13 @Description
14     Fonctions liées à l'entrée et la sortie du mode sommeil
15 */
16 /* **** */
17
18 #ifndef _SOMMEIL_H
19 #define _SOMMEIL_H
20
21
22 // Attention: DUREE_SOMMEIL_MS doit coincider avec la valeur de
23 DELAI_ENTRE_DEUX_REVEILS_MS du côté du STM (gestionBatterie.h)
24 #define DUREE_SOMMEIL_MS 5000    // Durée du mode sommeil de l'ESP en ms (quand il
n'est pas réveillé par le STM).
25 #define CONVERSION_MS_EN_uS 1000 // Facteur pour convertir des millisecondes en
microsecondes
26
27
28 #endif /* _SOMMEIL_H */
```

# Annexe T.11

```
1 // Sommeil.ino
2 //
3 // Description : fonctions liées à l'entrée et la sortie du mode sommeil
4 // Auteur : Perret Mélissa
5 // Création : 22/09/2024
6 // Modifications : --
7
8 // Version      : V1.0
9 /*-----*/
10
11 #include "Sommeil.h"
12
13
14
15 ///// Fonction EntréeModeSommeilProfond: logique pour entrer en mode sommeil profond
16 ///// Description: active les conditions de réveil (pin pour permettre le réveil
17 // depuis le STM, et timer pour le prochain réveil) et rentre en mode sommeil profond
18 ///// Entrées: -
19 ///// Sorties: -
20 void EntréeModeSommeilProfond() {
21
22     // Activation réveil possible via GPIO
23     const gpio_config_t config = {
24         .pin_bit_mask = BIT(PIN_REVEIL_PAR_STM),
25         .mode = GPIO_MODE_INPUT,
26     };
27     ESP_ERROR_CHECK(gpio_config(&config)); // Pour configurer la pin de réveil
28     ESP_ERROR_CHECK(esp_deep_sleep_enable_gpio_wakeup(BIT(PIN_REVEIL_PAR_STM),
29             ESP_GPIO_WAKEUP_GPIO_HIGH)); // Pour indiquer la pin permettant de sortir du mode
30             // sommeil (Permet au STM de réveiller l'ESP si besoin)
31
32     if (MODE_DEBUG) {
33         printf("Entrée en mode sommeil pendant %d ms (ou via réveil STM pin GPIO%d)\n",
34             DUREE_SOMMEIL_MS, PIN_REVEIL_PAR_STM);
35     }
36
37     // https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/system/sleep_modes.html
38     //esp_deep_sleep_pd_config(ESP_PD_DOMAIN_RTC_PERIPH, ESP_PD_OPTION_OFF);
39     esp_deep_sleep_start(); // Pour entrer en mode sommeil profond
40 }
41
42 ///// Fonction GestionRéveil: logique exécutée au moment du réveil
43 ///// Description: détermine la raison du réveil (lancement, STM ou timer), lance
44 // l'exécution de la logique correspondante au type de réveil
45 ///// Entrées: -
46 ///// Sorties: -
47 void GestionRéveil() {
48     esp_sleep_wakeup_cause_t raisonRéveil; // Déclaration variable locale
49     raisonRéveil = esp_sleep_get_wakeup_cause(); // Pour obtenir la raison du réveil
50     if (MODE_DEBUG) {
51         printf("\nRaison du réveil: %d\n", raisonRéveil);
52     }
53
54     // Si la raison du réveil est inconnue, c'est qu'il s'agit du premier démarrage de
55     // l'ESP
56     if (raisonRéveil == ESP_SLEEP_WAKEUP_UNDEFINED) {
57         if (MODE_DEBUG) {
58             printf("Initialisation au premier réveil\n");
59         }
60
61         // Initialisation au premier réveil
62         // On met des valeurs incohérentes pour s'assurer que les premières valeurs qui
63         // seront reçues du serveur seront considérées comme différentes
64         valeursServeur[0] = -9999;
65         valeursServeur[1] = -9999;
```

```
63     valeursServeur[2] = -9999;
64     valeursServeur[3] = -9999;
65     valeursServeur[4] = -9999;
66     valeursServeur[5] = -9999;
67
68     ExecutionReveilTimer();
69 } else {
70     // Si la raison du réveil provient de la pin GPIO, c'est que l'ESP a été réveillé
71     // par le STM
72     if (raisonReveil == ESP_SLEEP_WAKEUP_GPIO) {
73         if (MODE_DEBUG) {
74             printf("Réveil causé par le STM (trames UART à réceptionner)\n");
75
76             ExecutionReveilSTM();
77         } else {
78             // ESP réveillé par le timer
79             if (MODE_DEBUG) {
80                 printf("Réveil causé par le timer\n");
81             }
82
83             ExecutionReveilTimer();
84         }
85     }
86 }
```

# Annexe T.12

```
1  /* **** */
2  /** Descriptive File Name
3
4      @Company
5          ETML-ES
6
7      @File Name
8          Wifi.h
9
10     @Auteurs
11         - Perret Mélissa
12
13     @Description
14         Fonctions liées au Wifi
15     */
16  /* **** */
17
18
19 #ifndef _WIFI_H
20 #define _WIFI_H
21
22
23 #include <WiFiMulti.h> // pour WiFiMulti
24
25
26 // Wifi école
27 //#define WIFI_SSID "Wifi-Projet-Diplomes"
28 //#define WIFI_MOT_DE_PASSE "DiplomeSLO2"
29 //#define SERVEUR_ADDRESSE_HTTP "http://192.168.1.102:8080/MesureTH/api.php"
30
31 // Wifi perso
32 #define WIFI_SSID "SFR_7CD8"
33 #define WIFI_MOT_DE_PASSE "26dkjf2mjs26vyyk6pwg"
34 #define SERVEUR_ADDRESSE_HTTP "http://192.168.1.41:9090/MesureTH/api.php"
35
36 // Wifi test
37 //#define ADDRESSE_HTTP "https://www.howsmyssl.com/a/check"
38
39
40 WiFiMulti wifiMulti;
41
42
43 #endif /* _WIFI_H */
```

# Annexe T.13

```
1 // Wifi.ino
2 //
3 // Description : fonctions liées au Wifi
4 // Auteur : Perret Mélissa
5 // Création : 22/09/2024
6 // Modifications : --
7
8 // Version      : V1.0
9 /*-----*/
10
11
12 #include "Wifi.h"
13
14
15 /////////////////////////////////////////////////////////////////// Fonction ConfigurerWifi: configuration de la connexion Wifi
16 /////////////////////////////////////////////////////////////////// Description: initie la connexion au réseau Wifi en utilisant le SSID et le mot
17 /////////////////////////////////////////////////////////////////// de passe
18 /////////////////////////////////////////////////////////////////// Entrées: -
19 /////////////////////////////////////////////////////////////////// Sorties: -
20 void ConfigurerWifi() {
21     wifiMulti.addAP(WIFI_SSID, WIFI_MOT_DE_PASSE); // Connexion au réseau Wifi
}
```

# Annexe U.1

```
1  <!DOCTYPE html> <!-- Indispensable, indique qu'il s'agit d'une page HTML -->
2  <html lang="fr"> <!-- Pas obligatoire, mais indique la langue du site -->
3  <head>
4      <meta charset="utf-8"> <!-- Encodage des caractères : accents, idéogrammes
5          chinois et japonais, etc... -->
6      <title>Projet 2409 Mesure TH</title> <!-- Titre du site -->
7  </head>
8  <body>
9      <?php
10
11         // Important : ne pas oublier de Start MySQL dans XAMPP pour que ça
12         fonctionne
13         $connexionSQL = mysqli_connect("localhost", "root", "", "measureth"); // //
14         Connexion base de données
15
16         // Vérification connexion
17         if($connexionSQL === false) // Si la connexion a échouée
18         {
19             die("ERREUR: Connexion impossible. " . mysqli_connect_error());
20
21         $requeteSQL = "SELECT * FROM valeurs";
22         $resultatRequeteSQL = mysqli_query($connexionSQL, $requeteSQL);
23
24         if($resultatRequeteSQL) // Si requête effectuée avec succès
25         {
26             $colonnes = $resultatRequeteSQL->fetch_assoc(); // Récupérer les résultats
27         }
28         else
29         {
30             echo "ERREUR: Impossible d'executer la requete SQL suivante: $sql. " .
31             mysqli_error($connexionSQL);
32         }
33
34         mysqli_close($connexionSQL); // Fermer la connexion
35
36     ?>
37
38     <!-- Formulaire -->
39     <!-- valeurs initiales provenant de la base de données en utilisant le php -->
40     <form method="post" action="form.php"> <!-- Executon de form.php lorsque le
41     formulaire est envoyé avec le bouton Envoyer -->
42         <h1>Page pour le réglages des seuils et écarts pour le projet 2409 </h1>
43         <!-- H1 pour titre -->
44
45         <!-- step pour définir la précision des valeurs -->
46         <label>Seuil température min :</label>
47         <input type="number" step="0.01" name="SeuilTemperatureMin" id=
48             "SeuilTemperatureMin" value="<?php echo $colonnes["seuilTemperatureMin"]; ?>"> [°C]<br>
49
50         <label>Seuil température max :</label>
51         <input type="number" step="0.01" name="SeuilTemperatureMax" id=
52             "SeuilTemperatureMax" value="<?php echo $colonnes["seuilTemperatureMax"]; ?>"> [°C]<br>
53
54         <label for="EcartTempMax">Ecart température :</label>
55         <input type="number" step="0.01" min="0" name="EcartTemperature" id=
56             "EcartTemperature" value="<?php echo $colonnes["ecartTemperature"]; ?>">
57             [°C]<br>
58
59         <label>Seuil humidité min :</label>
60         <input type="number" min="0" max="100" name="SeuilHumiditeMin" id=
61             "SeuilHumiditeMin" value="<?php echo $colonnes["seuilHumiditeMin"]; ?>">
62             [%]<br>
63
64         <label>Seuil humidité max :</label>
65         <input type="number" min="0" max="100" name="SeuilHumiditeMax" id=
66             "SeuilHumiditeMax" value="<?php echo $colonnes["seuilHumiditeMax"]; ?>">
67             [%]<br>
```

```
57 <br>
58
59 <label>Ecart humidité :</label>
60 <input type="number" min="0" max="100" name="EcartHumidite" id=
61 "EcartHumidite" value="<?php echo $colonnes["ecartHumidite"]; ?>"> [%]<br>
62 <br>
63 <input type="submit" value="Envoyer"> <!-- Bouton pour envoyer le
64 formulaire -->
65 </form>
66
67 <br><br>
68
69 <?php
70 // Si l'alarme vaut vrai dans la base de données (valeur transmise par le STM
71 // à l'ESP, puis par l'ESP à la base de données)
72 if($colonnes["alarme"] == true)
73 {
74     echo "<div style='color:red; text-align: center; font-size:200px'>Alarme
75         !</div>"; // Afficher un message d'alerte sur la page
76 }
77
78 // Si pilesFaibles vaut vrai dans la base de données (valeur transmise par le
79 // STM à l'ESP, puis par l'ESP à la base de données)
80 if($colonnes["pilesFaibles"] == true)
81 {
82     echo "<div style='color:red; text-align: center; font-size:200px'>Piles
83         faibles !</div>"; // Afficher un message d'alerte sur la page
84 }
85
86 ?>
87 </body>
88
89 </html>
```

# Annexe U.2

```
1 <?php
2
3 // Fichier php utilisé pour traiter l'envoi de formulaire depuis le site HTTP
4 // (modifier les valeurs dans la base de données)
5
6 // Important : ne pas oublier de Start MySQL dans XAMPP pour que ça
7 // fonctionne
8 $connexionSQL = mysqli_connect("localhost", "root", "", "measureth"); // Connexion
9 // à la base de données
10
11 // Vérification connexion
12 if($connexionSQL === false) // Si la connexion a échouée
13 {
14     die("ERREUR: Connexion impossible. " . mysqli_connect_error());
15
16 // On stocke chaque valeur du formulaire dans une variable
17 // Utilisation de "mysqli_real_escape_string" recommandée pour gérer les
18 // éventuels caractères spéciaux (par précautions)
19 // Pour plus de détails :
20 https://www.lephpfacile.com/manuel-php/mysqli.real-escape-string.php
21 $seuil_temperature_min = mysqli_real_escape_string($connexionSQL, $_POST[
22     'SeuilTemperatureMin']);
23 $seuil_temperature_max = mysqli_real_escape_string($connexionSQL, $_POST[
24     'SeuilTemperatureMax']);
25 $ecart_temperature = mysqli_real_escape_string($connexionSQL, $_POST[
26     'EcartTemperature']);
27
28 $seuil_humidite_min = mysqli_real_escape_string($connexionSQL, $_POST[
29     'SeuilHumiditeMin']);
30 $seuil_humidite_max = mysqli_real_escape_string($connexionSQL, $_POST[
31     'SeuilHumiditeMax']);
32 $ecart_humidite = mysqli_real_escape_string($connexionSQL, $_POST['EcartHumidite']);
33
34 // Requête SQL UPDATE pour modifier les valeurs dans la base de données
35 $requeteSQL = "UPDATE valeurs SET seuilTemperatureMin='$seuil_temperature_min',
36     seuilTemperatureMax='$seuil_temperature_max', ecartTemperature='$ecart_temperature',
37     seuilHumiditeMin='$seuil_humidite_min', seuilHumiditeMax='$seuil_humidite_max',
38     ecartHumidite='$ecart_humidite'";
39
40 // Exécution de la requête SQL
41 if(mysqli_query($connexionSQL, $requeteSQL)) // Si requête exécutée avec succès
42 {
43     echo "Valeurs modifiées !"; // Affichage d'un message d'information sur la
44     // page
45 }
46 else
47 {
48     echo "ERREUR: Impossible d'exécuter la requête SQL. $sql. " . mysqli_error(
49         $connexionSQL); // Affichage d'un message d'erreur sur la page
50 }
51
52 echo "<br><br><form><input type='button' value='Retour'
53 onclick='history.back()'></form>"; // Bouton pour revenir à la page précédante
54
55 mysqli_close($connexionSQL); // Fermer la connexion
56 ?>
```

# Annexe U.3

```

1 <?php
2
3 // Fichier php utilisé pour traiter les requêtes HTTP depuis l'ESP
4 // (recevoir/modifier les données de la base de données)
5
6
7 // Important : ne pas oublier de Start MySQL dans XAMPP pour que ça
8 // fonctionne
9 $connexionSQL = mysqli_connect("localhost", "root", "", "measureth"); // Connexion
10 // à la base de données
11
12 // Vérification connexion
13 if($connexionSQL === false) // Si la connexion a échouée
14 {
15     die("ERREUR: Connexion impossible. " . mysqli_connect_error());
16 }
17
18 $typeDeRequete = $_SERVER['REQUEST_METHOD'];
19
20 switch ($typeDeRequete) {
21     case 'GET':
22         TraiterRequeteGet($connexionSQL);
23         break;
24     case 'PUT':
25         TraiterRequetePut($connexionSQL);
26         break;
27     default:
28         echo json_encode(['message' => 'Type de requête invalide']);
29         break;
30 }
31
32 mysqli_close($connexionSQL); // Fermer la connexion
33
34
35 ///// Fonction TraiterRequeteGet (traitement des requêtes HTTP de type GET)
36 ///// Description:
37 ///// Entrées: connexionSQL
38 ///// Sorties: -
39 function TraiterRequeteGet($connexionSQL)
40 {
41     // Requête SQL pour récupérer les données de la base de données
42     $requeteSQL = "SELECT
43     seuilTemperatureMin, seuilTemperatureMax, ecartTemperature, seuilHumiditeMin, seuil
44     HumiditeMax, ecartHumidite FROM valeurs";
45
46     $resultatRequeteSQL = mysqli_query($connexionSQL, $requeteSQL); // Exécution
47     // de la requête
48     if($resultatRequeteSQL) // Si requête exécutée avec succès
49     {
50         $colonnes = $resultatRequeteSQL->fetch_assoc(); // Récupérer les
51         résultats de la requête
52         echo json_encode($colonnes); // Encoder les résultats en JSON avec
53         json_encode et transmettre la réponse avec echo
54     }
55     else // Echec lors de l'exécution de la requête
56     {
57         echo "ERREUR: Impossible d'exécuter la requête SQL: $requeteSQL. " .
58         mysqli_error($connexionSQL);
59     }
60 }
61
62 ///// Fonction TraiterRequetePut (traitement des requêtes HTTP de type PUT)
63 ///// Description:
64 ///// Entrées: connexionSQL
65 ///// Sorties: -
66 function TraiterRequetePut($connexionSQL)
67 {
68     parse_str(file_get_contents("php://input"), $putData); // Récupérer les
69     données passées dans la requête et les stocker dans la variable $putData

```

```
62
63     if(count($putData) == 0) // Si la requête HTTP ne contient aucune données
64     {
65         // Pas de paramètres à modifier
66         echo "Pas de param";
67         return;
68     }
69
70     // Construction de la requête SQL
71     $requeteSQL = "UPDATE valeurs SET";
72
73     // Ajout des paramètres (nom et nouvelle valeur) indiqués dans la requête PUT
74     foreach ($putData as $nom => $valeur)
75     {
76         // On ajoute chaque données reçues (nom/valeur) à la requête SQL
77         $requeteSQL .= " $nom='$valeur',"; // .= pour ajouter à la suite du text
78         existant (+= ne marche pas)
79     }
80
81
82
83     $requeteSQL = trim($requeteSQL, ','); // Pour enlever la virgule en trop à la
84     fin
85
86
87     //echo $requeteSQL; // Pour debugguer la requête SQL créée
88     //return;
89
90
91     // Exécution de la requête
92     if(mysqli_query($connexionSQL, $requeteSQL)) // Si requête exécutée avec succès
93     {
94         echo "OK!";
95     }
96     else // Echec lors de l'exécution de la requête
97     {
98         echo "ERREUR: Impossible d'exécuter la requête SQL: $requeteSQL. ";
99         mysqli_error($connexionSQL);
100    }
101}
102?>
```

## Annexe V Flowchart STM32

### 1.1 Fonctions dans le fichier main.c

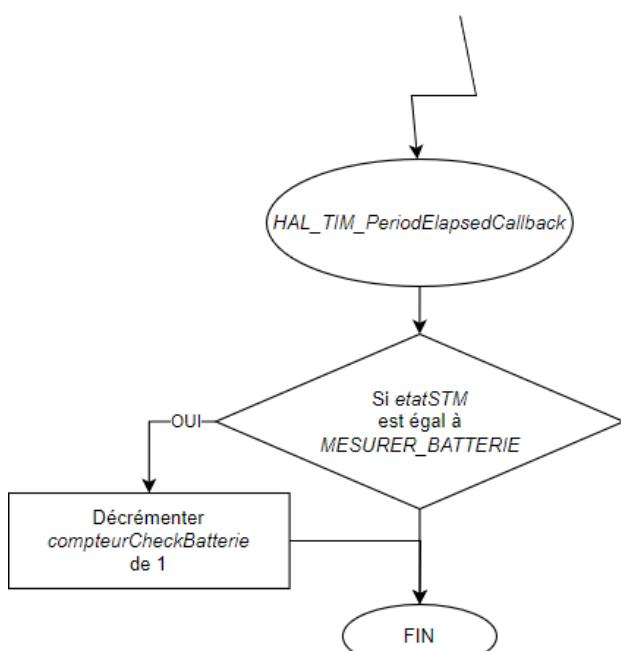


Figure 1 Fonction `HAL_TIM_PeriodElapsedCallback`

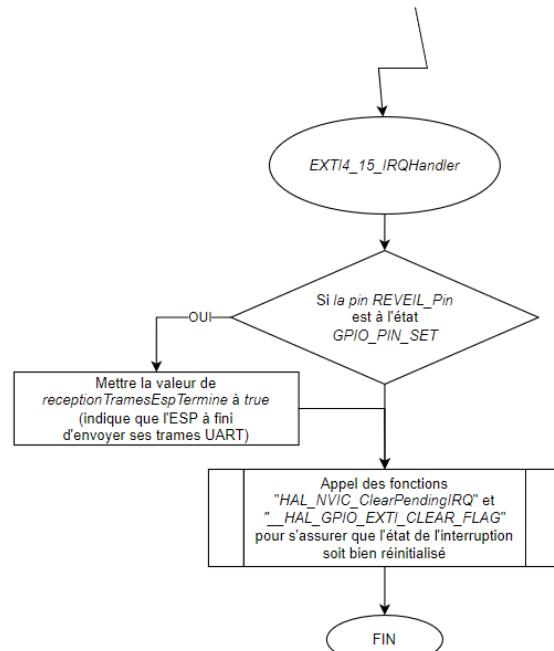


Figure 2 Fonction `EXTI4_15_IRQHandler`

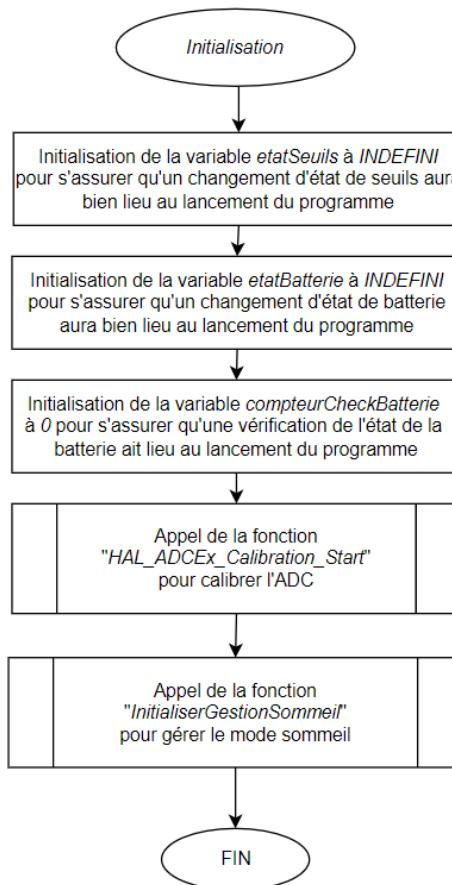


Figure 3 Fonction `Initialisation`

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
HAL_TIM_PeriodElapsedCallback	Pointeur TIM_HandleTypeDef htim	-	Fonction d'interruption du timer (délai entre activation transistor et mesure de l'état de la batterie)
EXTI4_15_IRQHandler	-	-	Fonction d'interruption de la pin utilisée par l'ESP pour réveiller le STM
Initialisation	-	-	Logique utilisée pour initialiser le système (initialisation des variables, calibration ADC, initialisation de la gestion du mode sommeil)

Tableau 1 Détail des fonctions dans main.c

## 1.2 Fonctions dans le fichier affichage.c

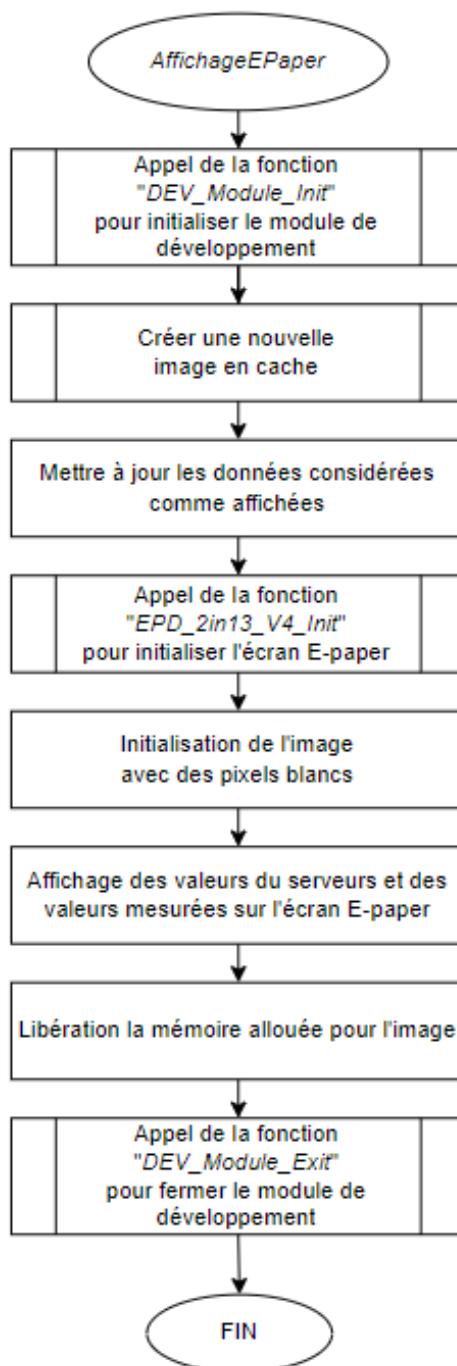


Figure 4 Fonction AffichageEPaper

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
AffichageEPaper	Pointeur Mesures mesures  Tableau DefinitionValeur valeursServeur	Entier 16 bits non signé	Affiche les données sur l'écran E-paper (valeurs provenant du serveur, valeurs mesurées avec la sonde)

Tableau 2 Détails des fonctions dans affichage.c

### 1.3 Fonctions dans le fichier communication.c

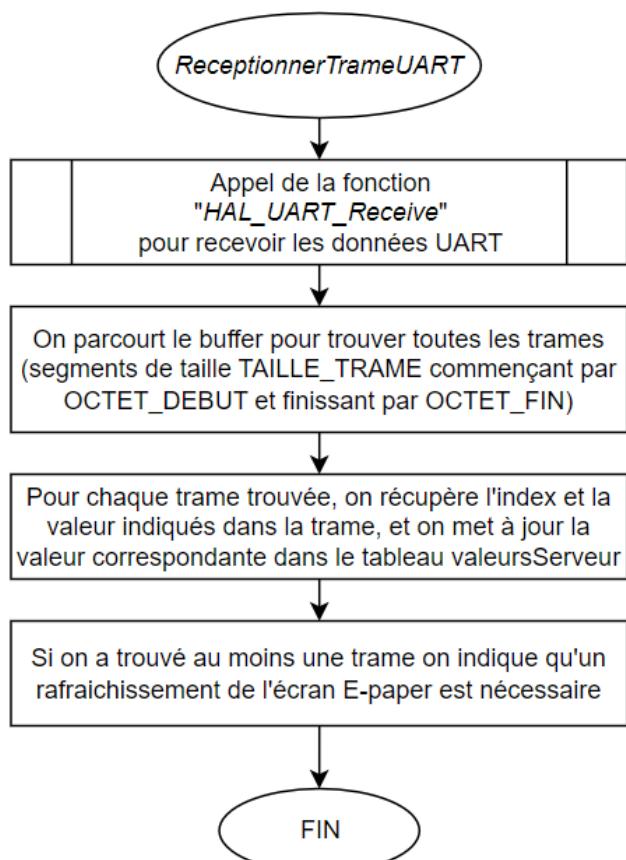


Figure 5 Fonction ReceptionnerTrameUART

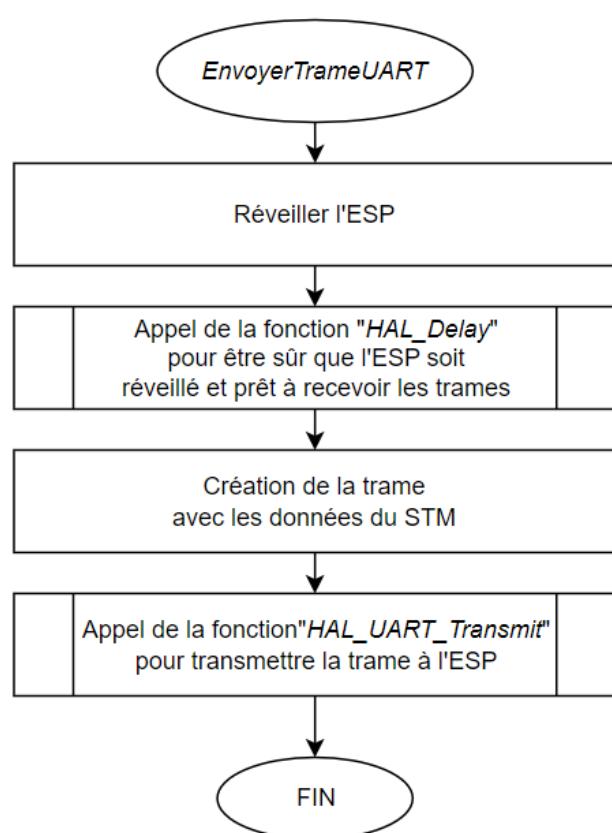


Figure 6 FonctionEnvoyerTrameUART

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
ReceptionnerTrameUART	Pointeur UART_HandleTypeDef huart  Tableau DefinitionValeur valeursServeur	-	Réception des données UART Analyse des données pour localiser les trames (caractères de début et de fin) Analyse des trames reçues pour extraire et stocker les données reçues
EnvoyerTrameUART	Pointeur UART_HandleTypeDef huart  Entier 8 bits non signé index  double valeur	-	Construire et transmettre la trame UART à l'ESP

Tableau 3 Détails des fonctions dans communication.c

## 1.4 Fonctions dans le fichier fonctionADC.c

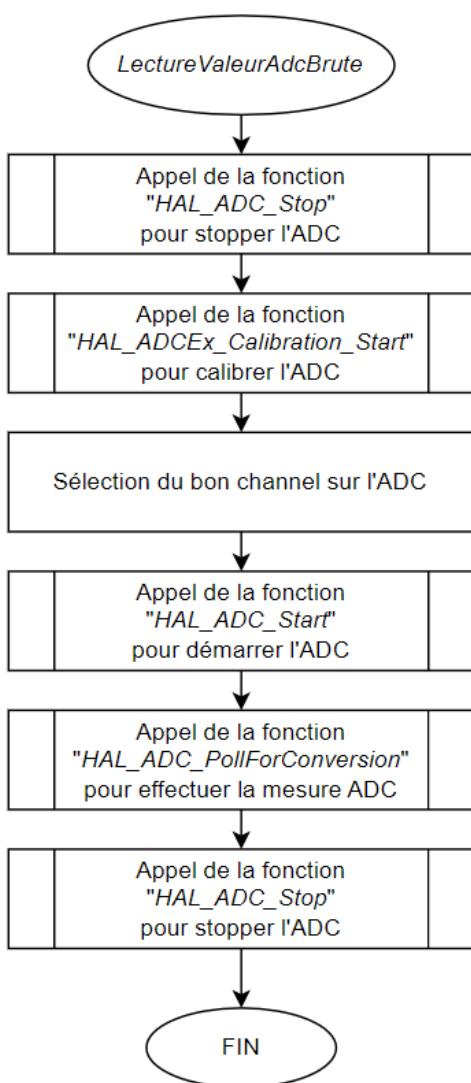


Figure 7 Fonction LectureValeurAdcBrute

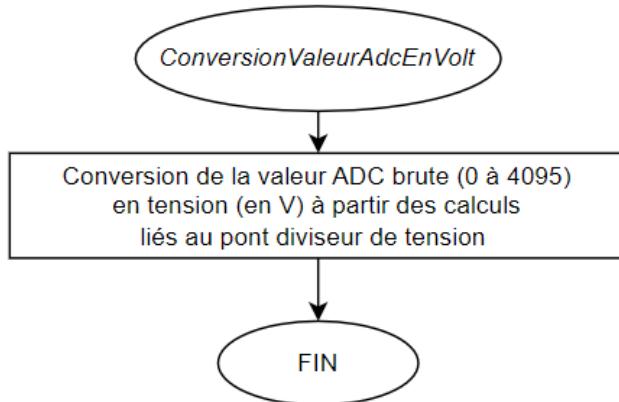


Figure 8 Fonction ConversionValeurAdcEnVolt

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
LectureValeurAdcBrute	Pointeur ADC_HandleTypeDef hadc Entier 8 bits non signé channel	Entier 16 bits non signé	Lire la valeur brute de l'ADC en fonction du channel
ConversionValeurAdcEnVolt	Entier 16 bits non signé valeurADC Entier 32 bits non signé pontDiviseurResistanceHaute Entier 32 bits non signé pontDiviseurResistanceBasse	float	Conversion d'une valeur brute ADC (0 à 4095) en une tension (en Volt)

Tableau 4 Détails des fonctions dans affichageADC.c

## 1.5 Fonctions dans le fichier gestionBatterie.c

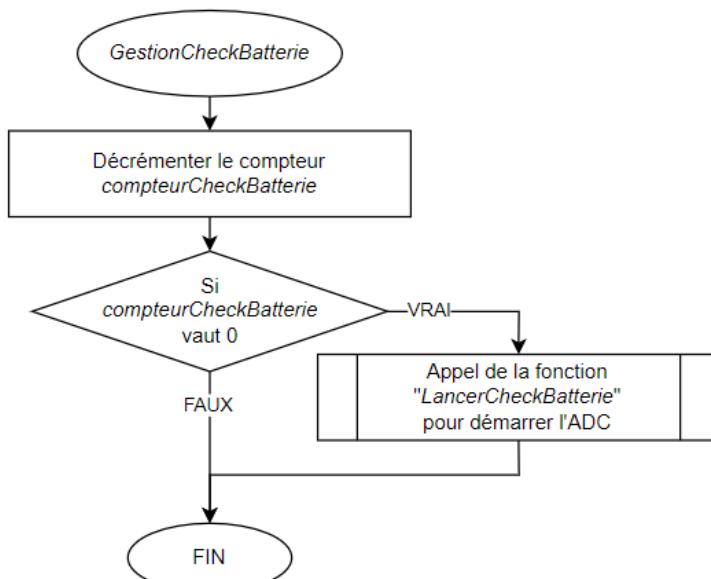


Figure 9 Fonction GestionCheckBatterie

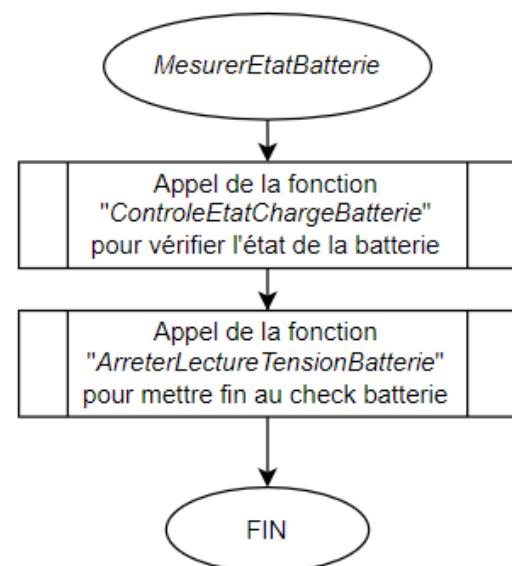


Figure 10 Fonction MesurerEtatBatterie

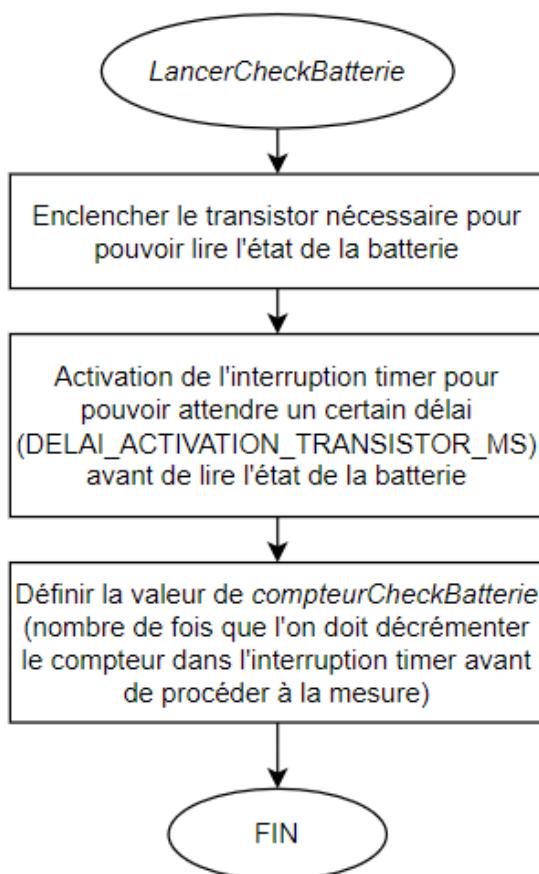


Figure 11 Fonction LancerCheckBatterie

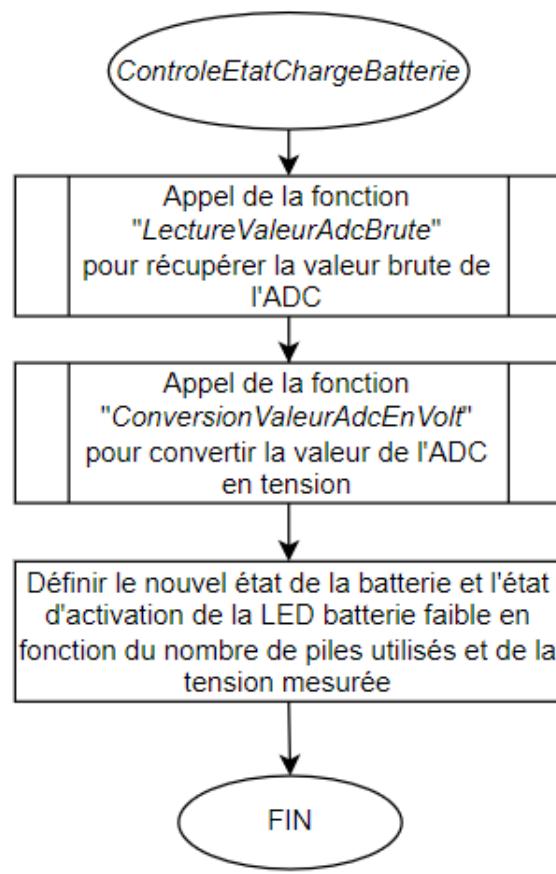


Figure 12 Fonction ControleEtatChargeBatterie

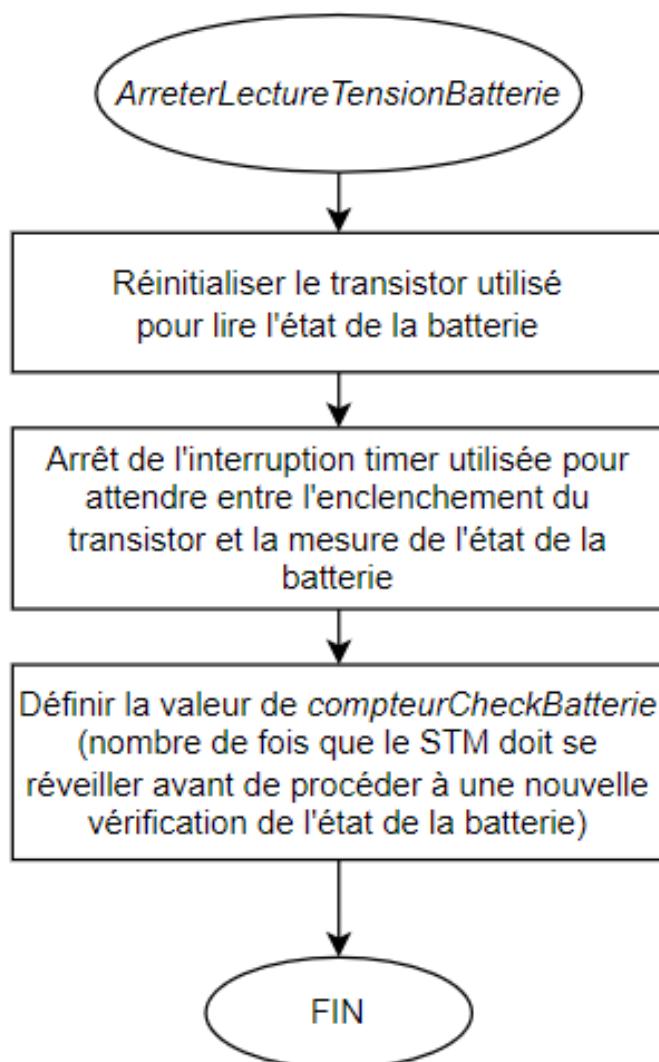


Figure 13 Fonction ArreterLectureTensionBatterie

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
GestionCheckBatterie	Pointeur ADC_HandleTypeDef hadc Pointeur TIM_HandleTypeDef htim Pointeur InfoBatterie infoBatterie	Booléen	Décrémente le compteur lié à la vérification de la batterie, déclenche le lancement de la vérification batterie si besoin
MesurerEtatBatterie	Pointeur ADC_HandleTypeDef hadc Pointeur TIM_HandleTypeDef htim Pointeur InfoBatterie infoBatterie	Etat	Utilise ControleEtatChargeBatterie pour récupérer l'état de la batterie et ArreterLectureTensionBatterie pour terminer l'opération
LancerCheckBatterie	Pointeur TIM_HandleTypeDef htim Pointeur InfoBatterie infoBatterie	-	Enclenche le transistor nécessaire pour pouvoir lire l'état de la batterie, met en place le timer pour le délai avant mesure
ControleEtatChargeBatterie	Pointeur ADC_HandleTypeDef hadc Pointeur InfoBatterie infoBatterie	Etat	Contrôle l'état de charge de la batterie (allume une LED en cas de problème détecté)
ArreterLectureTensionBatterie	Pointeur TIM_HandleTypeDef htim Pointeur InfoBatterie infoBatterie	-	Réinitialise le transistor utilisé pour lire l'état de la batterie, défini le compteur pour le délai avant la prochaine vérification

Figure 14 Détails des fonctions dans gestionBatterie.c

## 1.6 Fonctions dans le fichier mesures.c

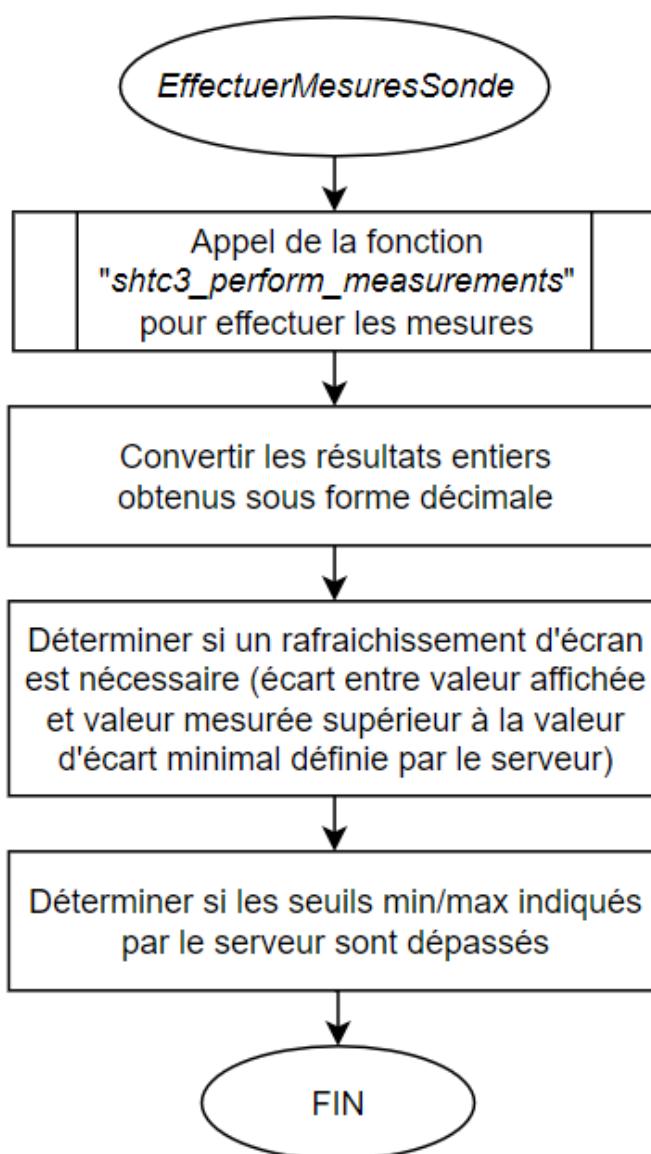


Figure 15 Fonction EffectuerMesuresSonde

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
EffectuerMesuresSonde	Pointeur I2C_HandleTypeDef hi2c2 Pointeur MesuresDef mesures Tableau DefinitionValeurDef valeursServeur	Etat	Mesure température et humidité

Tableau 5 Détails des fonctions dans mesures.c

## 1.7 Fonctions dans le fichier sommeil.c

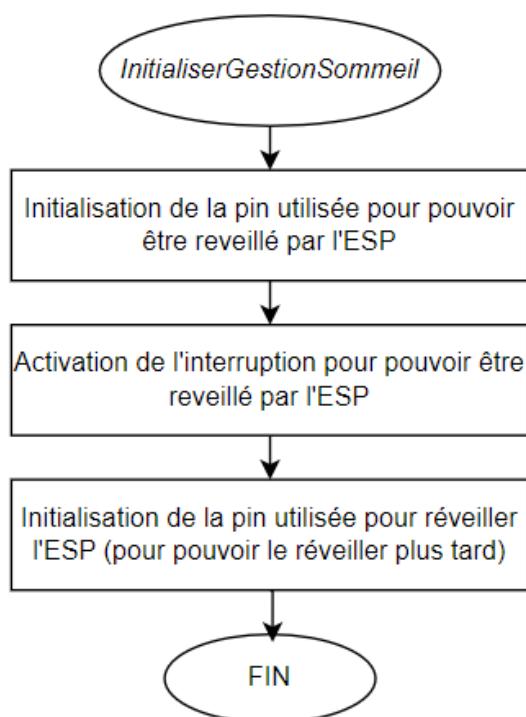


Figure 16 Fonction `InitialiserGestionSommeil`

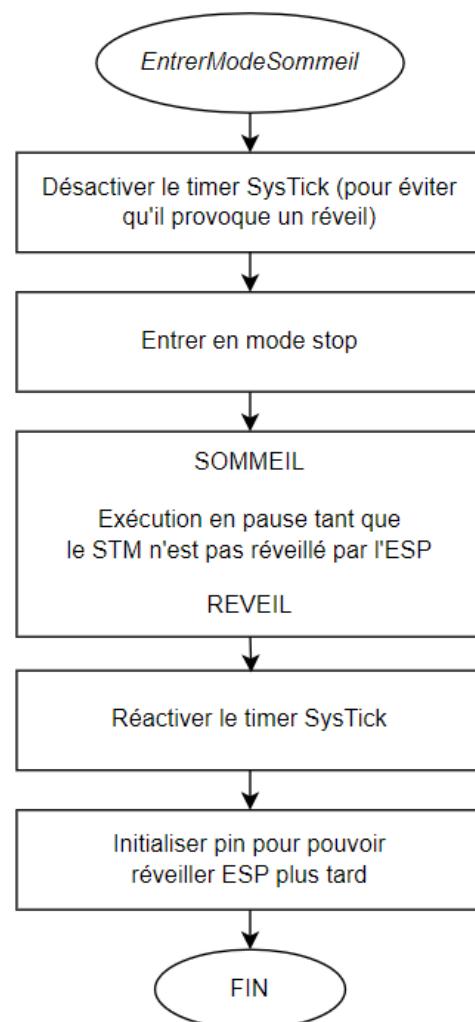


Figure 17 Fonction `EntrerModeSommeil`

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
InitialiserGestionSommeil	-	-	Configuration de la pin pour pouvoir être réveillé par l'ESP
EntrerModeSommeil	-	-	Logique pour mettre le STM en mode stop et remise en route après réveil

Tableau 6 Détails des fonctions dans `sommeil.c`

## Annexe W Flowchart ESP32

### 1.8 Fonctions dans le fichier 2409\_Esp\_V1.ino

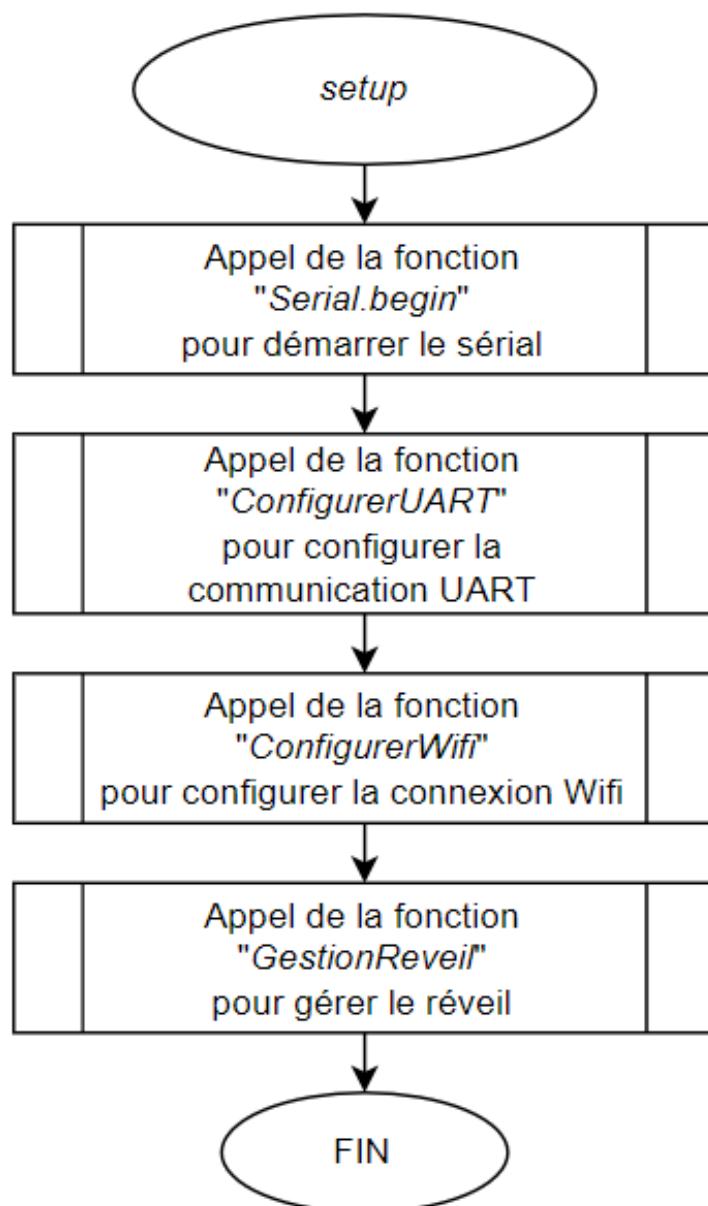


Figure 18 Fonction setup

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
setup	-	-	Configuration de la communication UAR, configuration du Wi-Fi, appel de la fonction GestionReveil

Tableau 7 Détails de la fonction dans `setup.ino`

## 1.9 Fonctions dans le fichier CommunicationSTM.ino

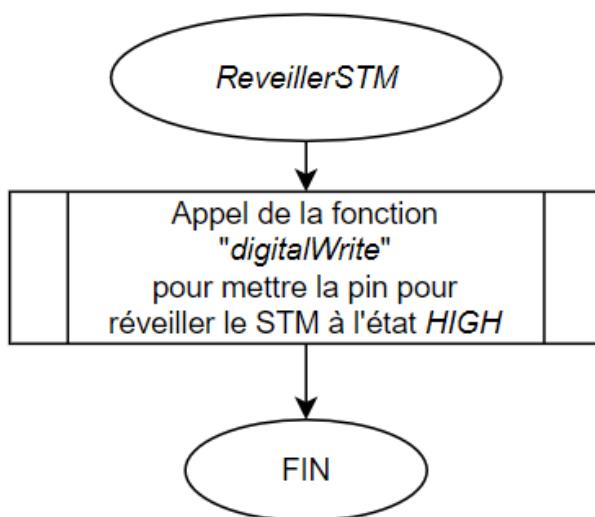


Figure 19 Fonction ReveillerSTM

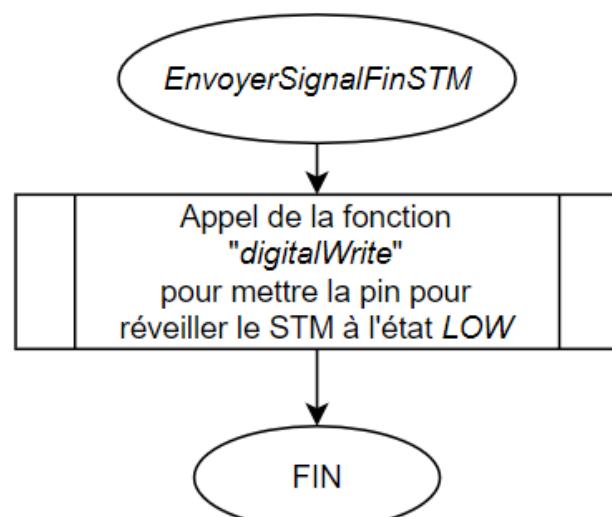


Figure 20 Fonction EnvoyerSignalFinSTM

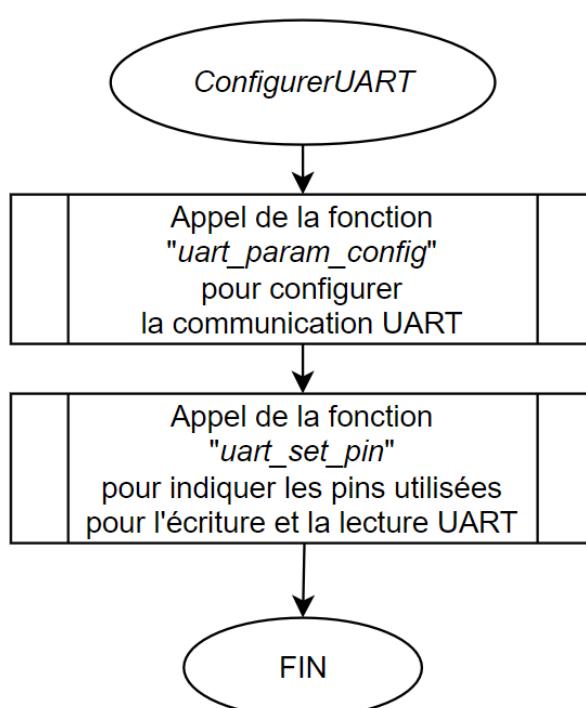


Figure 21 Fonction ConfigurerUART

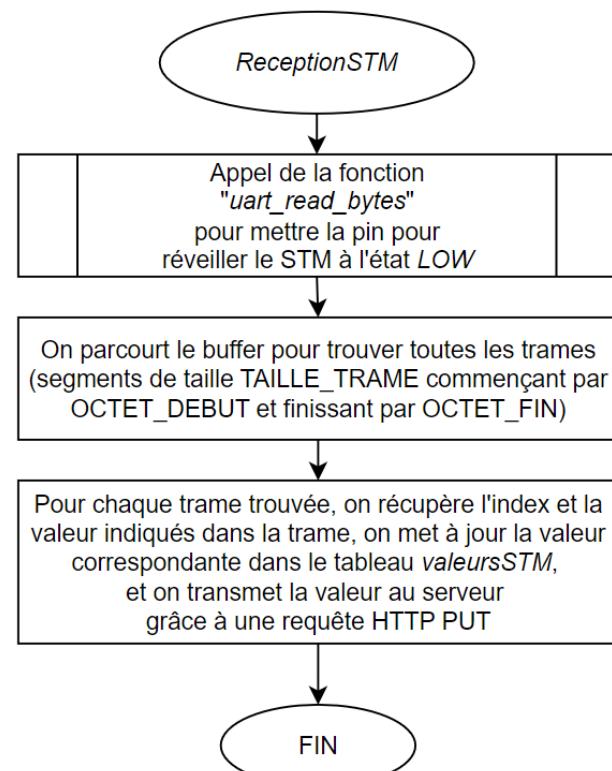


Figure 22 Fonction ReceptionSTM

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
ReveillerSTM	-	-	Change l'état de la pin utilisée pour réveiller le STM
EnvoyerSignalFinSTM	-	-	Met la pin utilisée pour réveiller le STM à l'état bas (la pin à l'état haut réveille le STM, et l'état bas lui indique qu'il a reçu toutes les trames nécessaires de la part de l'ESP)
ConfigurerUART	-	-	Défini les pins utilisées pour l'écriture et la lecture UART
ReceptionSTM	Pointeur HTTPClient http	-	Réception des données UART Analyse des données pour localiser les trames (caractères de début et de fin) Analyse des trames reçues pour extraire et traiter les données reçues Envoi de requête HTTP PUT pour transmettre les nouvelles données au serveur (états alarme seuils et batterie)

Tableau 8 Détails des fonctions dans CommunicationSTM.ino

## 1.10 Fonctions dans le fichier CommunicationServeur.ino

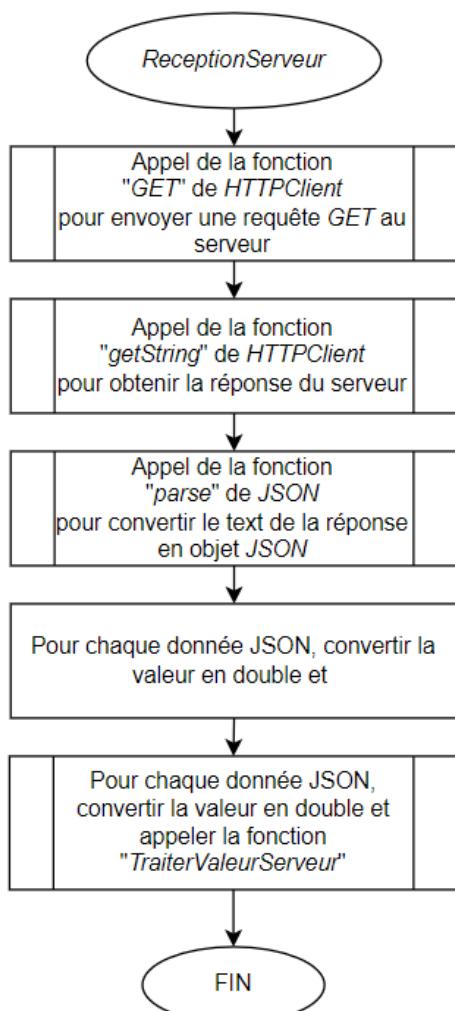


Figure 23 Fonction ReceptionServeur

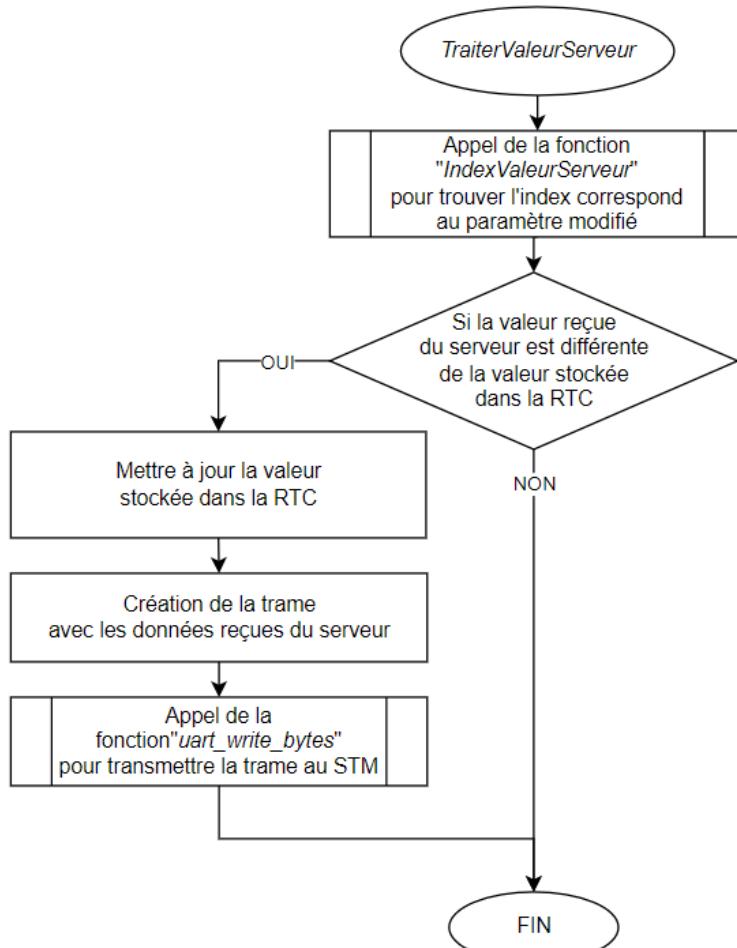


Figure 24 Fonction TraiterValeurServeur

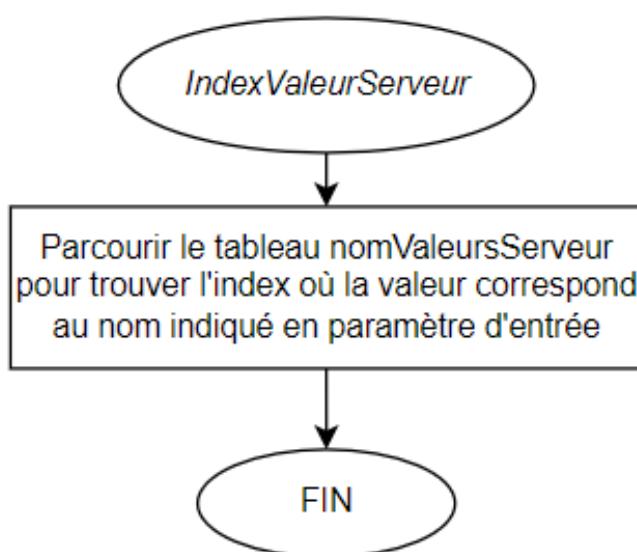


Figure 25 Fonction IndexValeurServeur

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
ReceptionServeur	Pointeur HTTPClient http	-	Envoi d'une requête HTTP Get au serveur Réception de la réponse, analyse de la réponse pour extraire les données reçues (seuils min/max et écarts pour température et humidité) Appel de la fonction TraiterValeurServeur pour traiter les données reçues
TraiterValeurServeur	String nom  double valeur	-	Vérifier si la valeur a changé Mettre à jour la valeur stockée dans la RTC Construire et transmettre la trame UART au STM
IndexValeurServeur	String nom	Entier 8 bits non signé	Parcourir le tableau nomValeursServeur pour trouver l'index où la valeur correspond au nom indiqué en paramètre d'entrée

Tableau 9 Détails des fonctions dans CommunicationServeur.ino

## 1.11 Fonctions dans le fichier Discord.ino

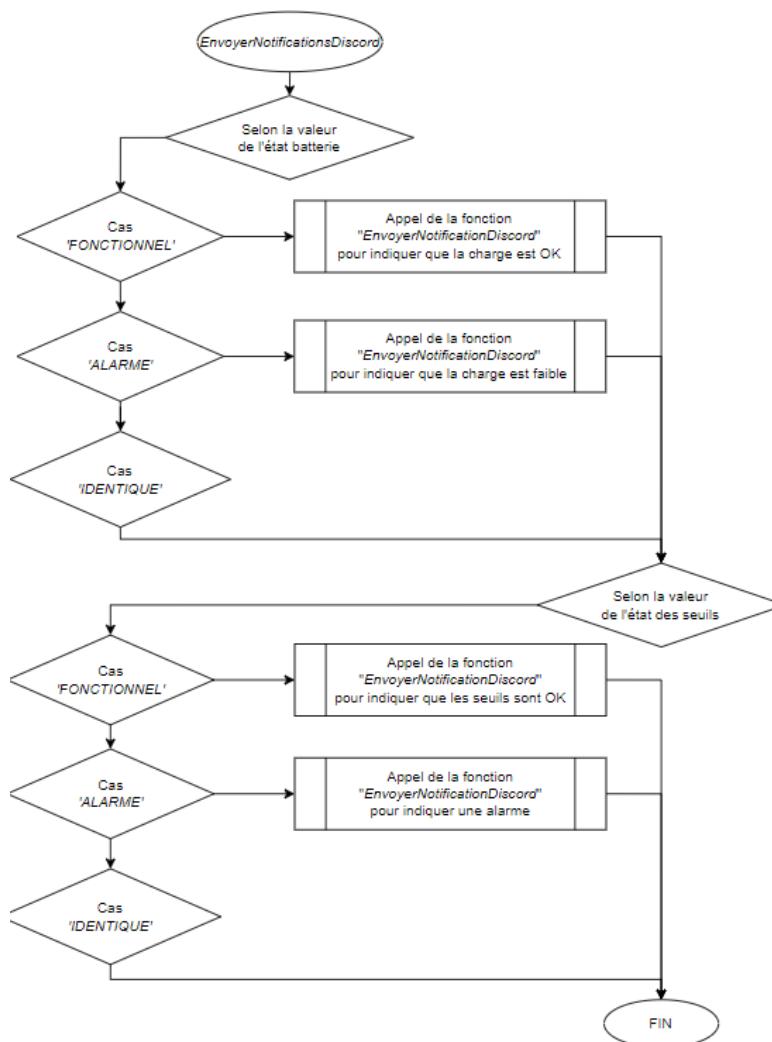


Figure 26 Fonction EnvoyerNotificationsDiscord

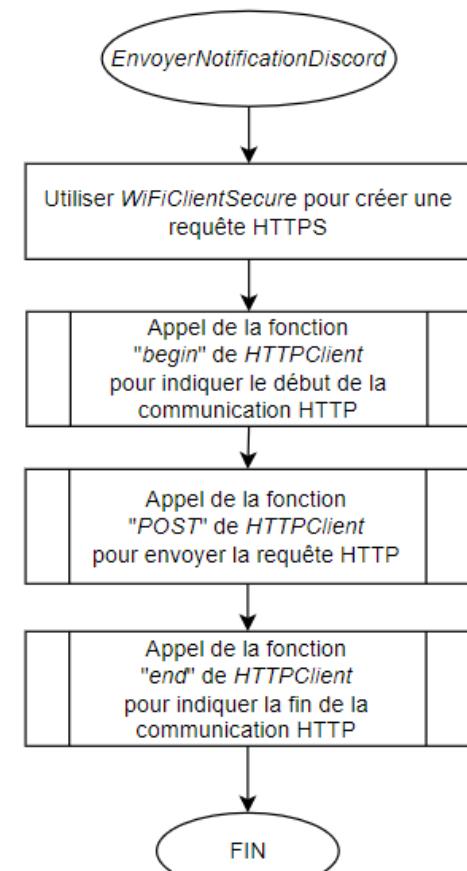


Figure 27 Fonction EnvoyerNotificationDiscord

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
EnvoyerNotificationsDiscord	Pointeur HTTPClient http	-	Gère l'envoie des notifications Discord (état batterie et état seuils alarme)
EnvoyerNotificationDiscord	Pointeur HTTPClient http  Booléen syntheseVocale  String message	Booléen	Gère l'envoie d'une notifications Discord (texte passé en paramètre d'entrée)

Tableau 10 Détails des fonctions dans Discord.ino

## 1.12 Fonctions dans le fichier Execution.ino

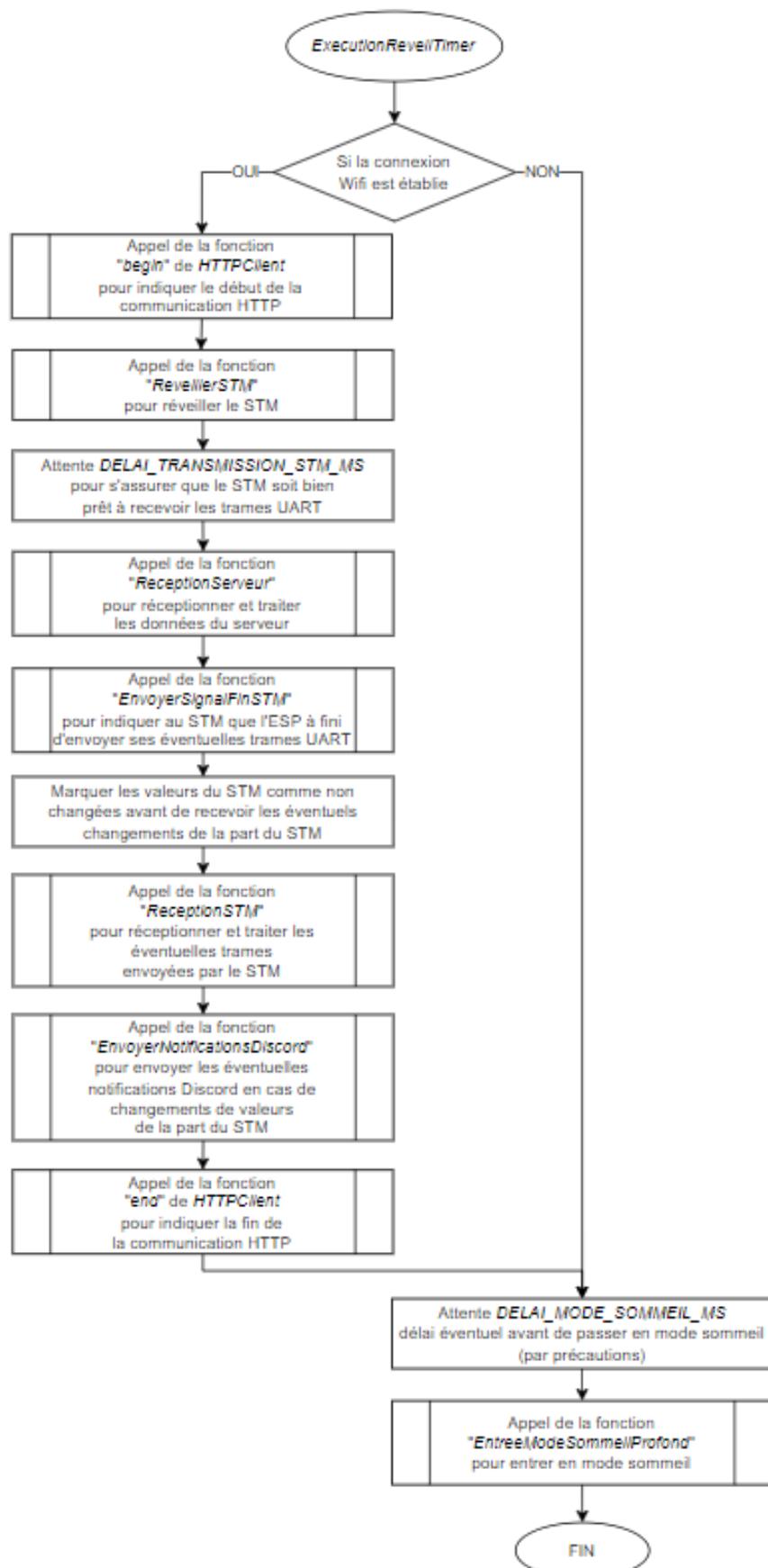
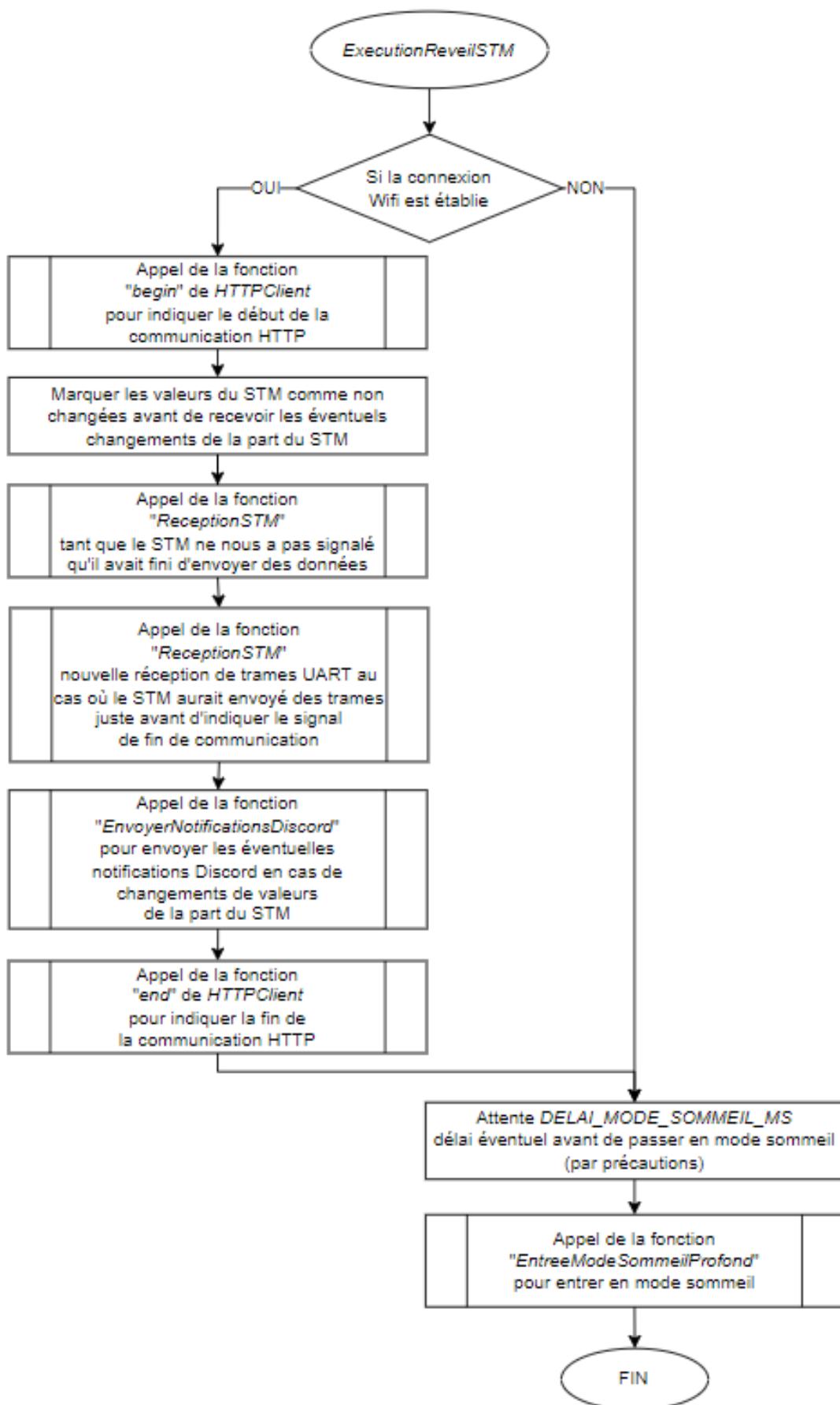


Figure 28 Fonction `ExecutionReveilTimer`

Figure 29 Fonction `ExecutionReveilSTM`

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
ExecutionReveilTimer	-	-	Connexion Wifi, réveiller le STM, récupération et transmissions des valeurs du serveur, envoie notification Discord, retour en mode sommeil
ExecutionReveilSTM	-	-	Réception trames STM, envoie notification Discord, retour en mode sommeil

Tableau 11 Détails des fonctions dans Execution.ino

### 1.13 Fonctions dans le fichier Sommeil.ino

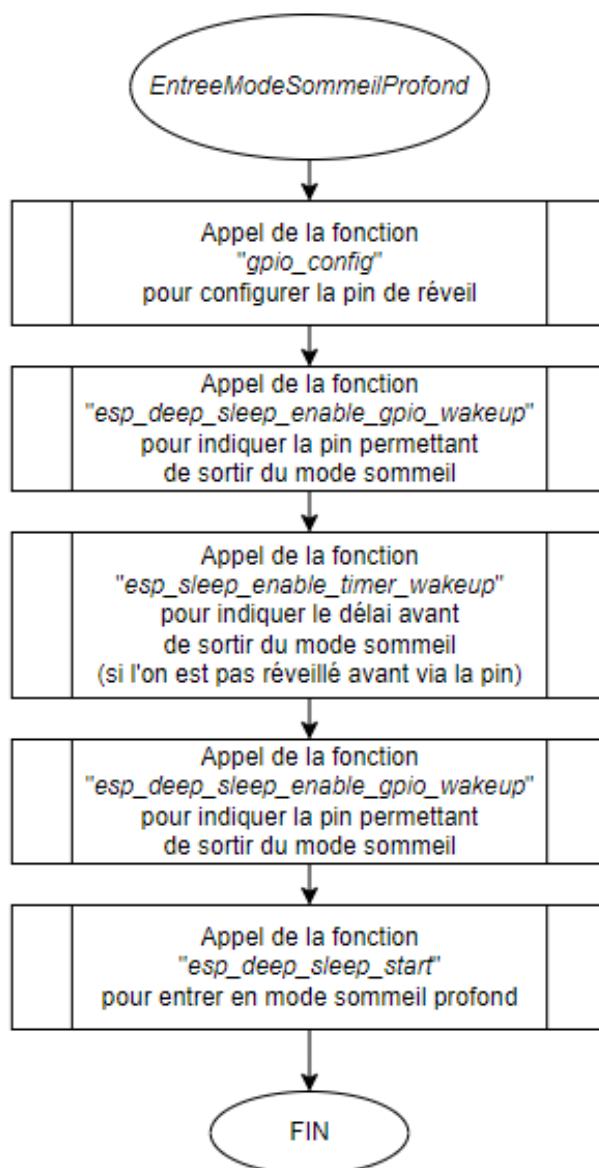


Figure 30 Fonction EntreeModeSommeilProfond

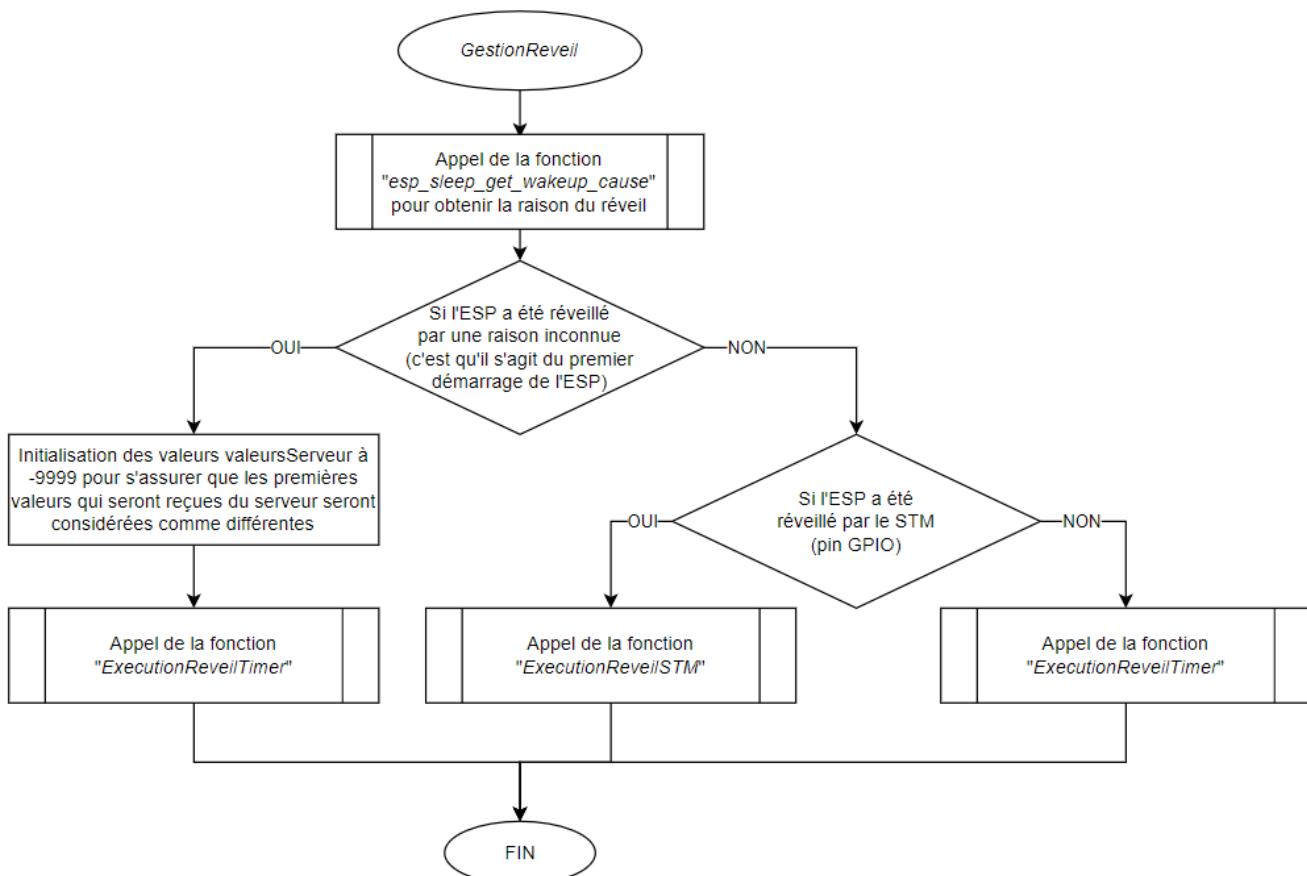


Figure 31 Fonction GestionReveil

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
EntreeModeSommeilProfond	-	-	Active les conditions de réveil (pin pour permettre le réveil depuis le STM, et timer pour le prochain réveil) et rentre en mode sommeil profond
GestionReveil	-	-	Détermine la raison du réveil (lancement, STM ou timer), lance l'exécution de la logique correspondante au type de réveil

Tableau 12 Détails des fonctions dans Sommeil.ino

### 1.14 Fonctions dans le fichier Wifi.ino

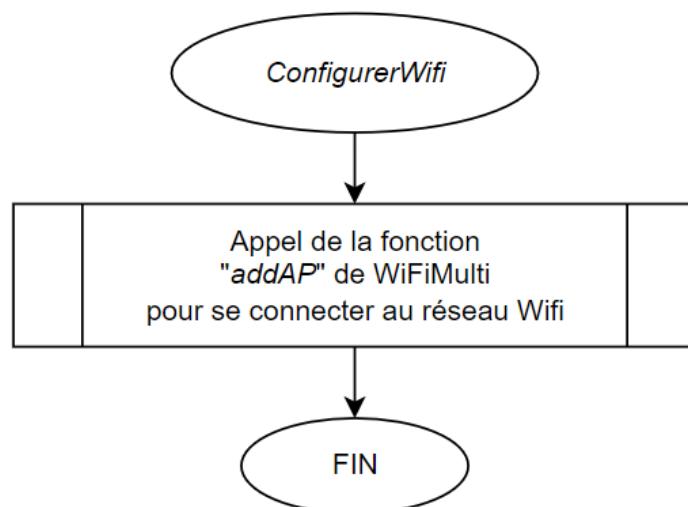


Figure 32 Fonction `ConfigurerWifi`

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
ConfigurerWifi	-	-	Initie la connexion au réseau Wifi en utilisant le SSID et le mot de passe

Tableau 13 Détails des fonctions dans `Wifi.ino`

## Annexe X Flowchart WEB

### 1.15 Fonctions dans le fichier form.php

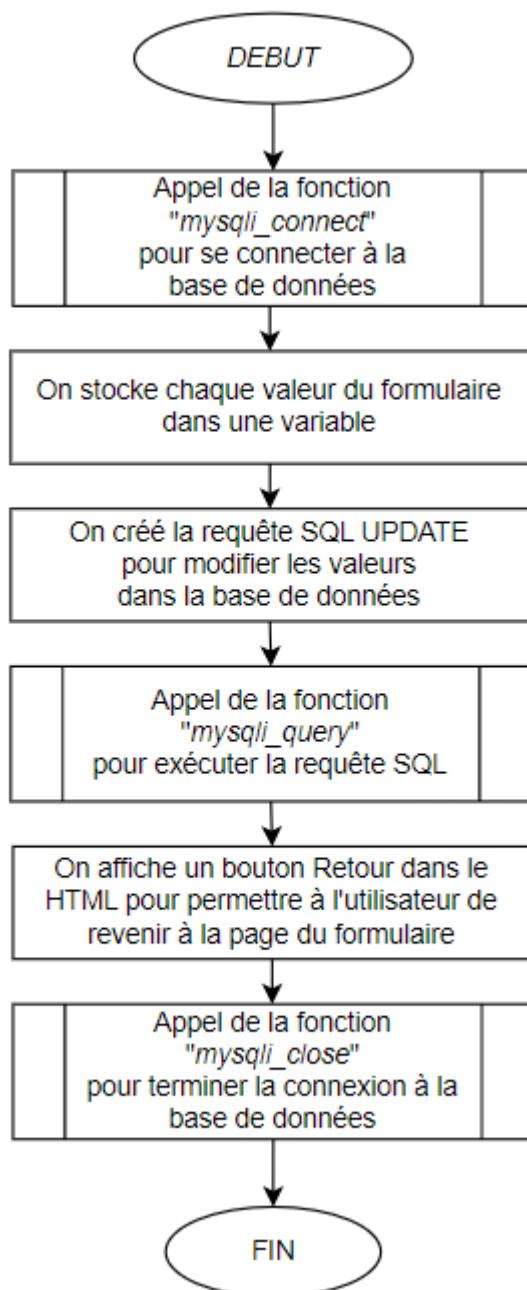


Figure 33 Fonction home

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
form	-	-	Transmet les valeurs du formulaire à la base de données

Tableau 14 Détails des fonctions dans form.php

### 1.16 Fonctions dans le fichier api.php

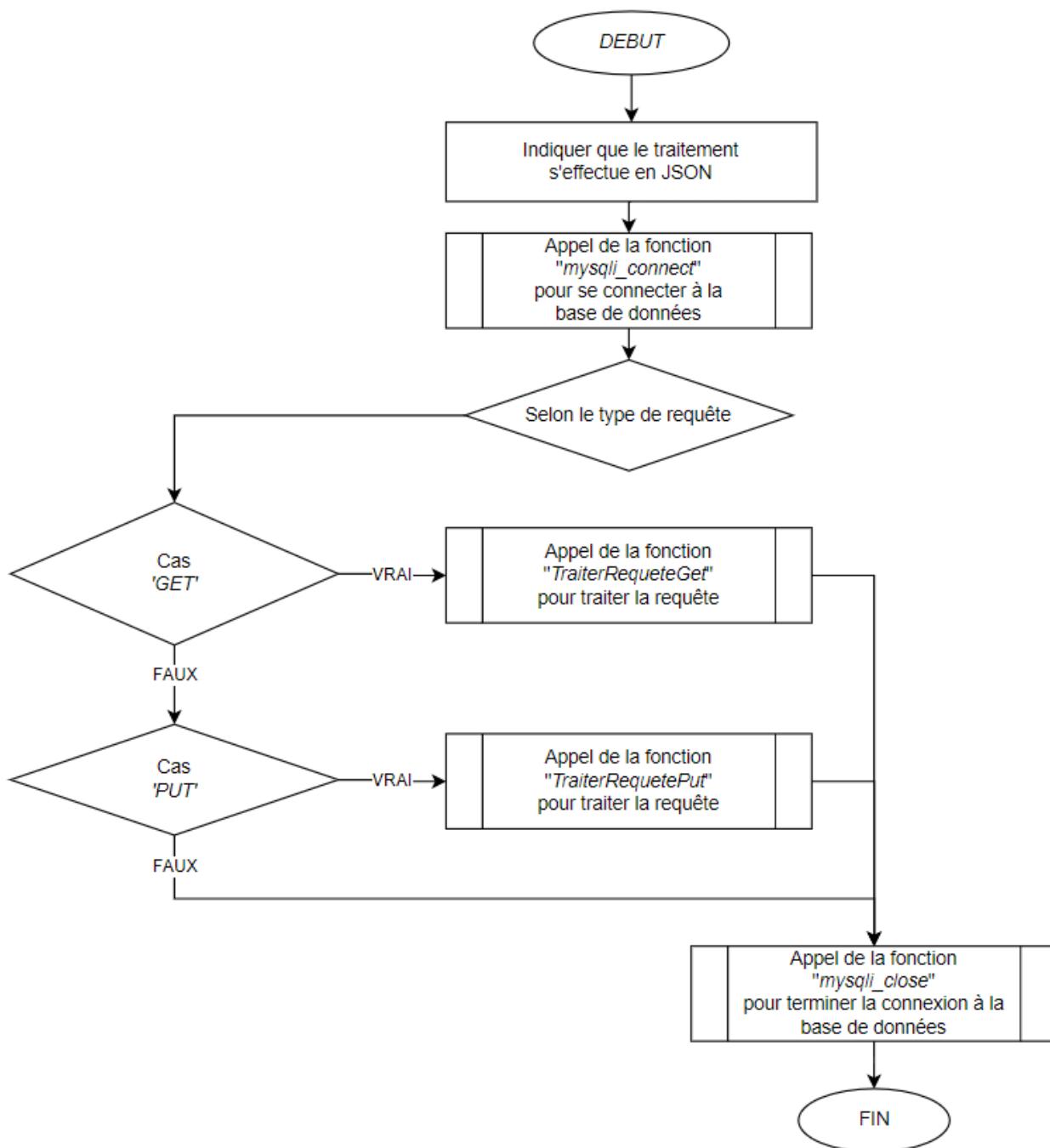


Figure 34 Fonction api

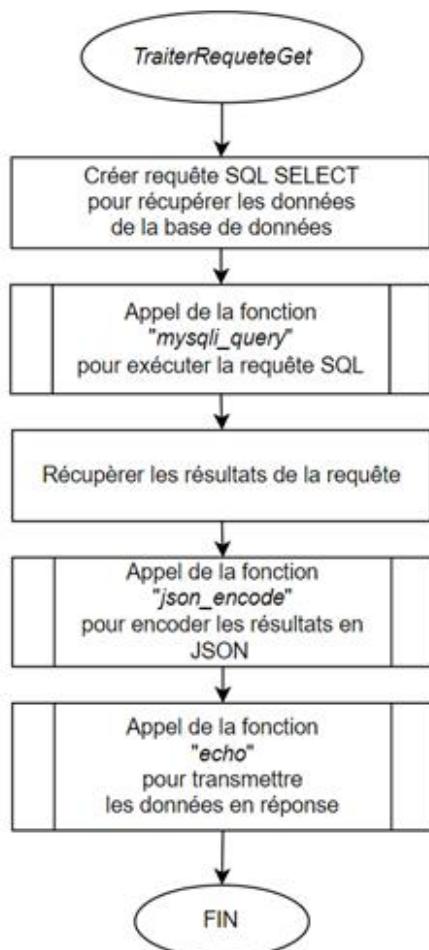


Figure 35 Fonction TraiterRequeteGet

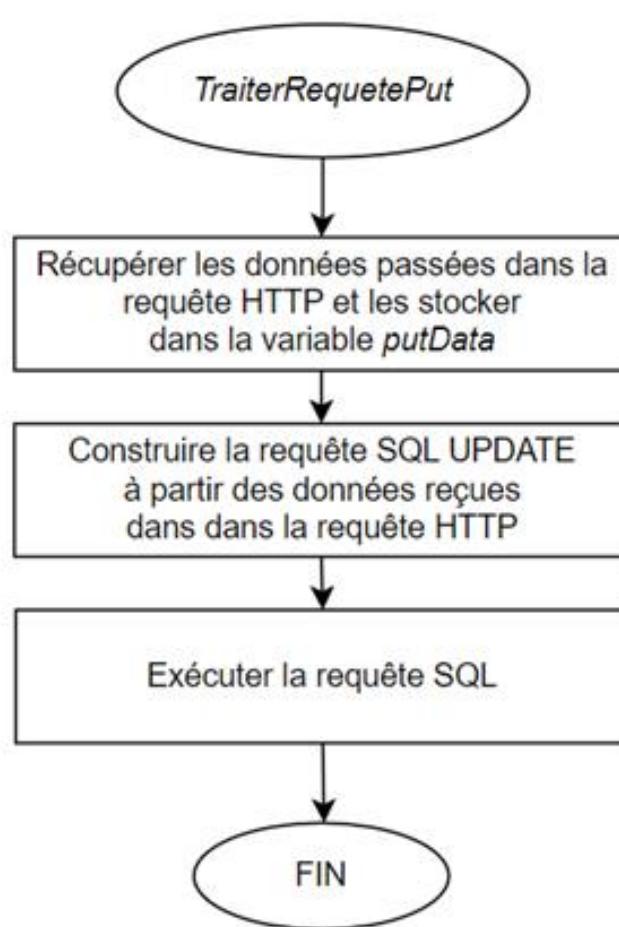


Figure 36 Fonction TraiterRequetePut

Nom de la fonction	Paramètres d'entrée	Paramètres de sortie	Description
api	-	-	Traitement des requêtes HTTP
TraiterRequeteGet	connexionSQL	-	Retourne les valeurs stockées dans la base de données du serveur
TraiterRequetePut	connexionSQL	-	Transmet les valeurs de la requête à la base de données

Tableau 15 Détails des fonctions dans api.php

# Projet ETML-ES – Modification

<b>PROJET:</b>	2409_MesureTH_RefrigerateurCongelateur		
<b>Entreprise/Client :</b>	Phillipe Bovey	<b>Département :</b>	loisir
<b>Demandé par (Prénom, Nom) :</b>	Phillipe Bovey	<b>Date :</b>	23.09.2024
<b>Objet (No ou réf, pièce, PCB...)</b>	Projet numéro 2409		
<b>Version à modifier:</b>	Version 1 (Version A)		

<b>Auteur (ETML-ES):</b>	Mélissa Perret	<b>Filière:</b>	SLO
<b>Nouvelle version:</b>	Version 1 (version A)	<b>Date:</b>	23.09.2024

## 1 Description ou justification

Changement du MOSFET Q1 après la présentation pour éviter tout problème et réaliser quelques petits ajustements pour la partie mécanique et la partie hardware.

## 2 Référence conception

Vous pouvez retrouver les dossiers contenant mon projet à l'endroit suivant :

K:\ES\PROJETS\SLO\2409\_MesureTH\_RefrigerateurCongelateur

## 3 Détail des modifications

### Partie Hardware

#	Description	Fait	Approuvé
1	Changement du MOSFET Q1	NOK	
2	Retirer la liaison PWRLED du capteur Sparkfun	NOK	

### Partie Mécanique

#	Description	Fait	Approuvé
1	Ajout des aimants pour la fermeture du boitier	NOK	
2	Coller le support de pile	NOK	

## 4 Remarques

Le changement de Q1 sera effectué après la présentation de diplôme

## 5 Convention de nommage et liens

- **2409\_MesureTH\_RefrigerateurCongelateur-MOD-v1.PDF**

### 5.1 Stockage du fichier

Ce fichier (2409\_MesureTH\_RefrigerateurCongelateur-MOD-v1.PDF) sera stocké à la racine du dossier **/doc** de ce projet