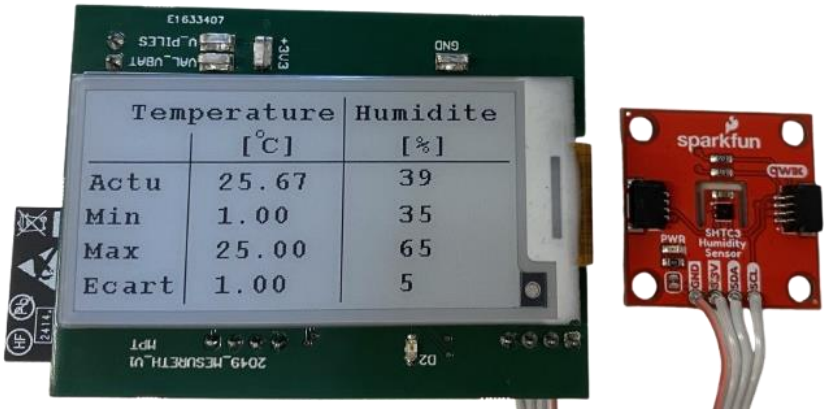
Projet de diplôme

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

2409_Surveillance Température-Humidité pour réfrigérateur_v1



Réalisé par :

Mélissa Perret

A l'attention de :

M. Bovey
M. Déglon
M. Jacot-Guillarmod

Date de début	Date de fin
19 août 2024	24 septembre 2024

Table des matières

1 Introduction 6
1.1 Contexte 6
1.2 But du projet 6
1.3 Lexique 7
2 Pré-étude 7
2.1 Schéma général du système 7
2.2 Schéma bloc global 7
2.2.1 Alimentation 7
2.2.2 Etat de la pile 8
Ce bloc permettra de surveiller l'état de charge de la pile. Il sera possible de signaler si celle-ci est faible. 8
2.2.3 STM32 8
2.2.4 LED de vie 8
2.2.5 Module ESP32 8
2.2.6 E-paper 8
2.2.7 Capteur de température et d'humidité 8
3 Choix des composants 8
3.1 Microcontrôleur 8
3.2 Affichage e-paper 8
3.3 Pile ou supercondensateur 9
3.3.1 Piles boutons 9
3.3.2 Supercondensateur 9
3.3.3 Piles AAA 11
3.4 Capteur température et humidité 11
3.5 ESP32 11
3.6 Boitier 12
4 Estimation du coût 12
5 Schématique 13
5.1 Schéma bloc 13
5.2 Alimentation 13
5.2.1 Entrée alimentation V_Piles et régulateur de tension 13
5.2.2 Contrôle état de la pile 15
5.3 Microcontrôleur et capteur température/humidité 16
5.3.1 Reset et debug 16
5.3.2 STM32 17
5.3.3 Chargeur d'amorçage (Bootloader) 18
5.3.4 Capteur température et humidité 19
5.4 ESP-32 20
5.4.1 UART0 et UART1 20
5.4.2 Pin ESP_EN 21
5.4.3 Port de programmation 21
5.4.4 Sélection de mode 22
5.5 E-paper 22
6 Hardware 24
6.1 Dimension du PCB 24
6.2 Zone interdite (keepout) 24
6.3 Placement des composants 25
6.4 Plan de masse 26
6.5 Attention particulière sur les composants 26
6.5.1 ESP32 26
6.5.2 Quartz externe 27
6.6 Largeur des pistes 27

6.6.1 DRC et Eurocircuit 28
6.7 Boitier 28
6.7.1 Boitier PCB 28
6.7.2 Boitier pour le capteur SparkFun 29
7 Coût total 30
8 Montage du PCB 30
9 Firmware 30
9.1 STM32 30
9.1.1 Configuration des pins sur le STM32 30
9.1.2 Configuration des GPIOs 31
9.1.3 Configuration du Timer 6 31
9.1.4 Configuration de l'ADC 32
9.1.5 Configuration UART 32
9.1.6 Configuration SPI 33
9.1.7 Configuration I2C 33
9.1.8 Configuration quartz externe 34
9.1.9 Configuration heap size et stack size 34
9.2 Fonctionnement globale du système 35
9.2.1 Fonctions dans le fichier main.c 35
10 ESP32-C3 35
10.1 Schéma branchement câble FTDI 35
10.2 Etapes pour la programmation 36
10.3 Fonctionnement global du système 36
10.3.1 Remarque supplémentaire 37
11 Création de la page HTML 37
12 Serveur 37
12.1 Principe de fonctionnement 37
12.2 Création serveur 38
12.3 Création base de données sur phpMyAdmin 38
12.4 Option supplémentaire : alarme sur serveur Discord 39
13 Tests 39
13.1 I2C 39
13.1.1 Schéma de mesure 39
13.1.2 Mesures 40
13.2 SPI 42
13.2.1 Schéma de mesure 42
13.2.2 Mesure premier test (e-paper non fonctionnel) 42
13.2.3 Mesures deuxième test (e-paper fonctionnel) 42
13.3 UART 44
13.3.1 Schéma de mesure 44
13.3.2 Mesures 44
13.4 Contrôle batterie 45
13.5 Consommation courant total du circuit 45
13.5.1 Schéma de mesure 45
14 Problèmes Rencontrés 46
14.1 MOSFET Q1 46
14.2 E-paper 46
14.3 ESP32-C3 46
14.3.1 Conflit avec STM32 47
14.4 Librairie Arduino 47
14.5 Problème connexion Wi-Fi 47
14.6 Problème réveil ESP32 47
15 Etat d'avancement 48

15.1 Tâches effectuées 48
15.2 Tâches restantes 48
16 Commentaires sur le déroulement du diplôme, Auto-analyse 49
16.1 Acquis du travail de diplôme 49
16.2 Prise en compte de l'environnement social 49
16.3 Prise en compte de l'environnement naturel 49
16.4 Leadership et développement personnel 49
17 Améliorations possibles 50
18 Conclusion 50
19 Bibliographie 52
20 Webographie 52
21 Logiciels utilisés 56
22 Figures 56
23 Tableaux 57
24 Projets références 57
Annexe A Cahier des charges 58
Annexe B Planification 58
Annexe C Journal de travail 58
Annexe D Procès-verbaux 58
Annexe E Schéma complet carte principale 58
Annexe F Fichier de fabrication 58
F.1 Assembly 58
F.2 Draftman 58
Annexe G Liste de pièces 58
Annexe H Schéma carte capteur de température 58
Annexe I Checklist revue de schéma et PCB 58
Annexe J Plan d'assemblage boitiers 58
J.1 Plan couvercle PCB V1 58
J.2 Plan boitier base PCB V1 58
J.3 Plan couvercle capteur température/humidité V1 58
J.4 Plan boitier base capteur température/humidité V1 58
Annexe K Liste de matériel 58
Annexe L Mesures 58
L.1 UART 58
L.2 SPI 58
Annexe M Librairies GitHub utilisées 58
Annexe N Installations liées à Arduino IDE 58
Annexe O Mode d'emploi 58
Annexe P Vérification des checksum I2C 58
P.1 Checksum humidité 58
P.2 Checksum température 58
Annexe Q Calculs utilisés pour l'estimation de consommation 58
Annexe R Flash ESP32 pour AT commandes 58
Annexe S Listing code STM32 58

S.1 shtc3.c 59
S.2 shtc3.h 59
S.3 affichage.c 59
S.4 affichage.h 59
S.5 gestionBatterie.c 59
S.6 gestionBatterie.h 59
S.7 fonctionsADC.c 59
S.8 fonctionsADC.h 59
S.9 communication.c 59
S.10 communication.h 59
S.11 mesures.c 59
S.12 mesures.h 59
S.13 sommeil.c 59
S.14 sommeil.h 59
S.15 main.c 59
S.16 main.h 59
S.17 librairie e-paper lien github 59
Annexe T Listing code ESP32 59
T.1 2409_ESP_V1 ino 59
T.2 CommunicationServeur.h 59
T.3 CommunicationServeur.ino 59
T.4 CommunicationSTM.h 59
T.5 CommunicationSTM.ino 59
T.6 Discord.h 59
T.7 Discord.ino 59
T.8 Execution.ino 59
T.9 Execution.h 59
T.10 Sommeil.h 59
T.11 Sommeil.ino 60
T.12 Wifi.h 60
T.13 Wifi.ino 60
Annexe U Listing code web 60
U.1 Home.php 60
U.2 Form.php 60
U.3 Api.php 60
Annexe V Flowchart STM32 60
Annexe W Flowachat ESP32 60
Annexe X Flowchart web 60
Annexe Y Fiche de modification 60

1 Introduction

1.1 Contexte

Ce travail de diplôme est réalisé dans le cadre de fin d'études à l'ETML-ES en génie électronique. Il permet de valider les compétences ainsi que les connaissances acquises tout au long de la formation. La réussite de ce travail de diplôme conduit à l'obtention du diplôme de technicien en génie électrique (section électronique).

Ce projet a été réalisé sur une durée de cinq semaines. M. Bovey a été mon maître de diplôme, avec qui j'ai pu effectuer un suivi hebdomadaire des tâches accomplies. Ce dernier, ainsi que les experts mandatés par l'école. M. Déglon et M. Jacot-Guillarmod, réaliseront l'évaluation finale de ce rapport et de la défense orale.

1.2 But du projet

Le but du projet est de mettre en place un système mesurant la température et l'humidité dans un réfrigérateur, avec un affichage e-paper placé à l'avant du PCB pour avoir toutes les informations visibles. Ce projet me permet de travailler avec des communications I2C, SPI et UART.

Le cahier des charges initial prévoyait l'utilisation d'un ESP32 en mode serveur web. Cependant, après analyse, cela impliquait une consommation de courant trop importante. Cette option s'est donc révélée non viable dans le cadre d'une alimentation par pile bouton ou supercondensateur. Avec l'accord du client, nous avons à la place décidé de créer un serveur en local, hébergé sur une machine virtuelle, pour modifier les seuils ou écarts pour la température et l'humidité.

L'ESP32 récupèrera les informations du serveur et les transmettra au STM32 en cas de changement de valeur. Il pourra également envoyer des alertes si un seuil est dépassé ou si l'alimentation est faible. Le STM32 mettra à jour l'e-paper en fonction des changements ou des variations détectés.

Après analyse approfondie, j'ai discuté avec le client pour modifier l'alimentation. La pile bouton n'étant pas adaptée et le supercondensateur, bien que souhaité dans le cahier des charges, ne garantissant pas une autonomie d'un mois minimum, le client a validé l'option d'alimentation par piles AAA.

J'ai également ajouté une fonctionnalité supplémentaire que je trouvais pertinente : en cas d'alarme, un message sera envoyé sur un serveur Discord, permettant ainsi au client de recevoir les alertes sur une plateforme qu'il utilise fréquemment.

Le cahier des charges au complet se trouve en annexe A.
Les procès-verbaux indiquant les modifications se trouvent en annexe D.

1.3 Lexique

Abréviation	Description
E-ink / e-paper	Encre électronique / Papier électronique
ESP32	Module Wi-Fi
LED	Diode Emettrice de Lumière (Light-Emitting Diode)
STM32	Microcontrôleur ARM 32 Bit
ETML	Ecole Technique et des Métiers de Lausanne
ES	Ecole Supérieure
CDC	Cahier Des Charges
PCB	Circuit imprimé (Printed Circuit Board)
DRC	Vérification des règles de conception sur Altium Designer
PLA	Filament d'impression 3D en Acide Polylactique
ABS	Filament d'impression 3D en Acrylonitrile Butadiène Styrène
PETG	Filament d'impression 3D en PolyEthylene Terephthalate Glycol
GET	Méthode de requête HTTP pour obtenir des données sur un serveur
PUT	Méthode de requête HTTP pour envoyer des données vers un serveur
JSON	JavaScript Object Notation (format de données structurées en texte)
PHP	Hypertext Preprocessor (langage de programmation côté serveur)
HTTP	Hypertext Transfer Protocol (protocole de communication client-serveur)

2 Pré-étude

2.1 Schéma général du système

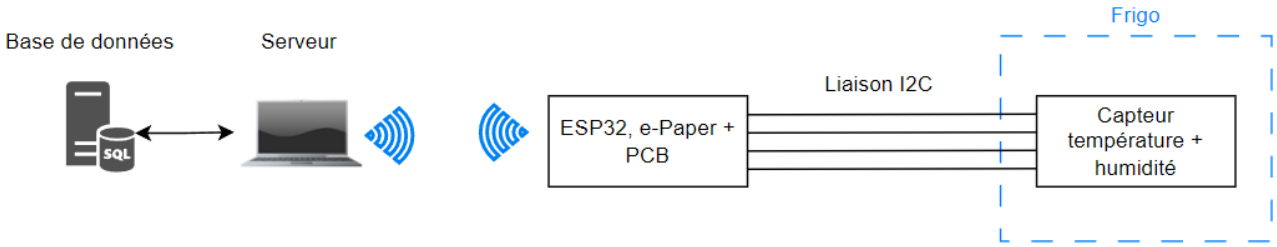


Figure 1 Schéma général

2.2 Schéma bloc global

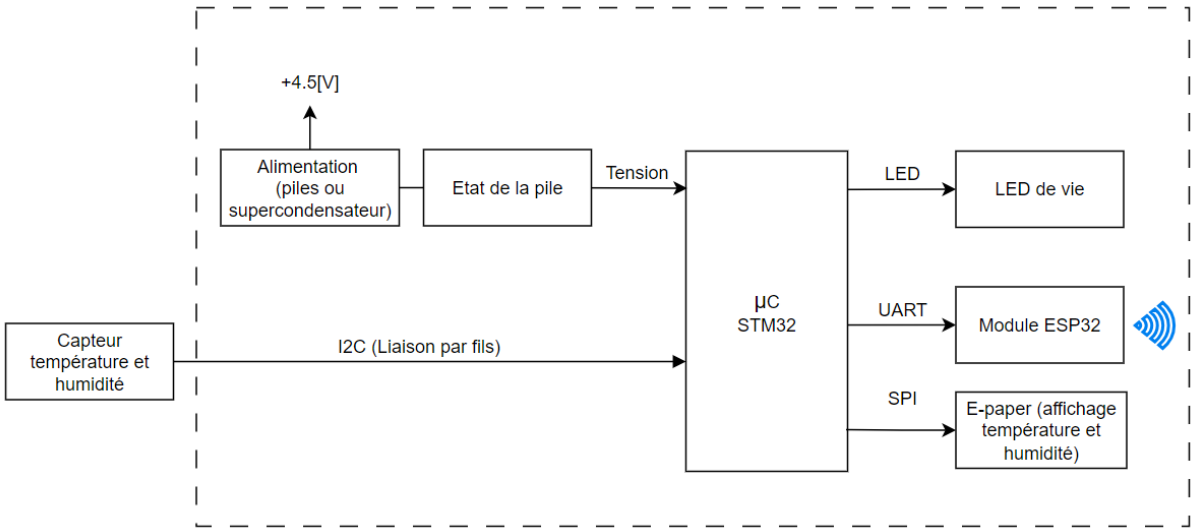


Figure 2 Schéma bloc global

2.2.1 Alimentation

Le PCB sera alimenté à l'aide d'une pile bouton ou d'un supercondensateur. Une autonomie d'un ou deux mois minimum est souhaitée.

2.2.2 Etat de la pile

Ce bloc permettra de surveiller l'état de charge de la pile. Il sera possible de signaler si celle-ci est faible.

2.2.3 STM32

Le choix du microcontrôleur est libre, j'ai opté pour un STM32, étant plus à l'aise avec celui-ci. Il sera utilisé avec une sonde pour récupérer la température et l'humidité. Il me permettra également de recevoir ou envoyer des données au module ESP32 via une communication UART.

2.2.4 LED de vie

La LED indiquera lorsqu'il sera nécessaire de remplacer la pile ou de recharger le circuit. Elle restera allumée jusqu'à ce que l'action soit effectuée. Elle sera de couleur rouge.

2.2.5 Module ESP32

Le module ESP32 permettra la gestion du serveur, ainsi que de recevoir et émettre des données.

2.2.6 E-paper

L'écran e-paper permettra d'afficher les valeurs actuelles de la température et de l'humidité, ainsi que les écarts et les seuils limites.

2.2.7 Capteur de température et d'humidité

Le capteur sera placé à l'intérieur du réfrigérateur et enverra les données au STM32 via une communication I2C.

3 Choix des composants

Le but ici est de faire un choix concernant les composants importants qui seront utilisés pour le projet.

3.1 Microcontrôleur

Concernant le choix du microcontrôleur, je reprendrai le STM32 que j'avais utilisé lors de mon précédent projet, il s'agit du « *STM32F072C8T6TR ».* N'ayant pas accès aux licences payantes du logiciel « Keil µVision » et ayant déjà utilisé celui-ci, je sais qu'il est compatible avec la version gratuite.

J'ai créé un tableau pour estimer approximativement le nombre de pins minimum nécessaires :

Pin analogique	Uart	SPI	I2C	Pin numérique
Etat de la batterie	Com RX (ESP)	SPI_SCLK	SDA	LED rouge
	Com TX (ESP)	SPI_MOSI	SCL	ALARME (ESP)
		SPI_CS		Réveil STM32 depuis ESP
1 pin	2 pins	3 pins	2 pins	3 pins
Total de pins minimum	**11 pins**			

Tableau 1 Pins minimum requises

3.2 Affichage e-paper

Concernant l'affichage, il m'est demandé d'utiliser un écran à encre électronique. N'ayant pas de contrainte de taille, je suis partie sur le modèle « Waveshare 2.13 pouces », ayant une résolution de 250x122[pixels]. J'ai choisi ce modèle car une librairie sur GitHub de chez Waveshare est disponible avec les STM32. La taille de cet écran e-paper devrait être suffisante pour afficher les informations importantes.

Lien de la librairie GitHub : https://github.com/waveshareteam/e-Paper/tree/master/STM32

E-paper choisi : https://www.digitec.ch/fr/s1/product/waveshare-250x122-213inch-e-ink-raw-noir-blanc-ecran-e-paper-carte-de-developpement-kit-24963573

3.3 Pile ou supercondensateur

3.3.1 Piles boutons

Initialement, je souhaitais mettre deux piles de 3[V] en série afin d'obtenir 6[V]. Cependant, à la suite de mes recherches, il s'est avéré que les piles boutons ne permettaient pas de garantir le bon fonctionnement du module ESP32. En effet, d'après le datasheet de la pile CR2477, le courant maximum que la pile peut supporter est de 15[mA][1]. Sachant que l'ESP32 consomme un courant entre 280 et 345[mA] lorsqu'il est actif[2], la pile ne pourrait donc pas fournir assez de courant pour qu'il fonctionne correctement. Suite à cela, j'ai décidé d'étudier le cas d'une alimentation avec un supercondensateur.

3.3.2 Supercondensateur

Afin de savoir si mon supercondensateur pourrait tenir au minimum un mois, j'ai estimé le mode de fonctionnement du circuit pour calculer sa consommation. Pour cela, j'ai établi les données suivantes :

Supercondensateur : 4[V] / 1200[F]

ESP32 : allumé toutes les heures pendant 5[s] (estimation) pour vérifier si les consignes ont été changées sur la page web. En cas de changement, envoie des valeurs au STM32. Réveille le STM32 avant de s'éteindre.

Remarque : il n'est pas possible d'utiliser l'ESP32 comme serveur, cela impliquerait de garder le Wi-Fi constamment allumé ce qui réduirait considérablement l'autonomie. Pour éviter cela, nous utiliserons un ordinateur comme point d'accès avec lequel l'ESP32 communiquera à chaque réveil.

Remarque : l'envoi de requête pour signaler un problème (de température ou d'humidité) est considéré comme suffisamment exceptionnel pour être négligé dans les calculs.

STM32 : allumé toutes les heures pendant 5[s] (estimation) pour effectuer les mesures (température, humidité, batterie). Comparaison des valeurs avec les consignes. En cas de consigne non respectée, envoie un signalement à l'ESP32 pour qu'il puisse transmettre l'alerte au serveur.

Remarque : j'ai estimé la durée d'activation de l'ESP32 et du STM32 à 5[s], je m'attends à ce que la durée soit plus faible dans la réalité, mais je souhaitais inclure une marge de sécurité dans mes calculs.

E-paper : la consommation du e-paper est plus élevée lors des rafraichissements d'écrans (changement de valeurs à afficher). D'après le datasheet de la V4[3], un rafraichissement dure environ 3[s] avec une consommation de 3.5[mA]. Etant donné la faible fréquence des mesures, et la prise en compte d'un écart minimal à respecter pour rafraichir l'écran, j'ai estimé qu'il pourrait y avoir en moyenne 10 rafraichissements par mois.

Autres composants : estimé à 500[µA]

A partir de ces données, j'ai réalisé un tableau Excel calculant la consommation moyenne du circuit (moyenne pondérée prenant en compte le temps en mode réveil et le mode sommeil). Cela permet de modifier facilement les paramètres d'entrée (comme par exemple le délai entre deux réveils) pour directement voir l'impact sur l'autonomie de la batterie.

[1] Omnergy, « SPECIFICATION FOR LITHIUM BATTERY : MODEL CR2477 », [en ligne], (consulté le 20 août 2024), tableau Maximum current, p.10. URL : https://www.farnell.com/datasheets/1496886.pdf
[2] Espressif Systems, « ESP32-C3-WROOM-02, ESP32-C3-WROOM-02-U : Datasheet », [en ligne], (consulté le 20 août 2024), version 1.3, table 9, p.15. URL : https://www.espressif.com/sites/default/files/documentation/esp32-c3-wroom-02_datasheet_en.pdf
[3] Waveshare. « 2.13inch e-Paper », [en ligne], consulté le 9 août 2024, version 4, point 6.2 (Typical operating current) et 8 (T update), p. 9 et 24. URL : https://files.waveshare.com/upload/4/4e/2.13inch_e-Paper_V4_Specification.pdf

Délai entre deux réveils [minutes]	Total heures mensuelles [heures / mois]	Durée réveil microcontrôleurs [secondes]	Rafraichissements e-paper [nombre / mois]	Durée rafraichissement e-paper [secondes]
60	720	5	10	3

Composants	Consommation en mode sommeil [mA]	Consommation en mode réveil [mA]	Sommeil / mois [heures]	Réveil / mois [heures]	Consommation moyenne [mA]
STM32	0,90	4,75	719	1	0,905
ESP32	0,005	82	719	1	0,119
E-paper	0,02	3,50	719,992	0,008	0,020
Autres	0,50	0,50	719	1	0,500
Total	1,425	90,75			**1,544**

Tableau 2 Estimation réaliste de la consommation pondérée du circuit par mois

L'ensemble des calculs nécessaires pour ce tableau, ainsi que la consommation de chaque composant, se trouvent en annexe Q.
Comme indiqué sur le tableau 2, la consommation moyenne devrait être d'environ 1,544[mA]. Pour déterminer l'autonomie du condensateur avant que celui-ci soit déchargé, j'ai d'abord calculé sa capacité en mAh.

Pour ce faire, j'ai effectué les calculs suivants :

Calcul de l'énergie en joules [4]:

$$W[J] = \frac{1}{2} \cdot C \cdot U^2 = \frac{1}{2} \cdot 1200 \cdot 4^2 = 9600\ [J]$$

Calcul pour avoir le résultat en Wh[5] :

$$W[Wh] = \frac{E[J]}{3600} = \frac{9600}{3600} = 2.67[Wh]$$

Conversion en mAh :
J'ai ensuite utilisé la formule tirée du site Digikey[6] pour convertir les wattheures en ampères-heures :

$$Ah = \frac{Wh}{V} = \frac{2.67}{4} = 0.6675\ [Ah] = 667.5\ [mAh]$$

Enfin, j'ai utilisé la formule d'autonomie de batterie tirée du même site :

$$Autonomie\ (h) = \frac{mAh}{mA} = \frac{667.5}{1.544} = 432.319\ [heures] = 18.01\ [jours]$$

On constate que la durée souhaitée d'un mois d'autonomie n'est pas non plus possible avec un supercondensateur.

[4] Rosset. Claude. et *al.* « FORMULAIRE : ELECTRONIQUE – TRANSMISSION », LEP Loisirs et Pédagogie SA, Le Mont-sur-Lausanne, 2018, p.21
[5] *Ibid., p.14*
[6] Digikey. « Calculateur d'autonomie des batteries », [en ligne], (consulté le 20 août 2024). URL : https://www.digikey.fr/fr/resources/conversion-calculators/conversion-calculator-battery-life

3.3.3 Piles AAA

Finalement, après discussions avec le client, nous avons décidé de partir sur un système de piles AAA trouvables en magasin. Avec une capacité d'environ 1200[mAh], cela permet d'atteindre l'autonomie souhaitée d'environ un mois.

$$Autonomie\ (h) = \frac{mAh}{mA} = \frac{1200}{1.544} = 777.202\ [heures] = 32.38\ [jours]$$

Par précautions, j'ai également réalisé une estimation en prenant des marges plus élevées en utilisant mon tableau Excel. Malgré une durée de réveil plus élevée pour les microcontrôleurs, avec des rafraîchissements plus longs et plus nombreux pour l'écran e-paper, on constate que cela n'impacte que très peu la consommation moyenne finale. Cela s'explique notamment par le fait que la durée à l'état réveillé reste très faible par rapport à la durée totale de fonctionnement. L'objectif d'un mois d'autonomie reste atteint (30 jours d'autonomie).

Délai entre deux réveils [minutes]	Total heures mensuelles [heures / mois]	Durée réveil microcontrôleurs [secondes]	Rafraichissements e-paper [nombre / mois]	Durée rafraichissement e-paper [secondes]
60	720	10	100	6

Composants	Consommation en mode sommeil [mA]	Consommation en mode réveil [mA]	Sommeil / mois [heures]	Réveil / mois [heures]	Consommation moyenne [mA]
STM32	0,90	4,75	718	2	0,911
ESP32	0,005	82	718	2	0,233
E-paper	0,02	3,50	719,833	0,167	0,021
Autres	0,50	0,50	718	2	0,500
Total	1,425	90,75			**1,664**

Alimentation	Capacité [mAh]	Autonomie [jours]
Supercondensateur	667,5	**16,71**
Piles AAA	1200	**30,04**

Tableau 3 Estimation de la consommation pondérée du circuit par mois (marges élevées)

Remarque : l'esp32 n'est pas viable en mode serveur, celui-ci consommant en permanence 82[mA][7], cela réduirait l'autonomie à environ 14[heures]. En discutant avec le client nous avons décidé de créer un serveur en mode local (voir PV 29_08_24).

3.4 Capteur température et humidité

Le capteur étant déporté, j'ai pris une carte de chez SparkFun, la « SEN-16467 » faisant directement office de capteur de température et d'humidité, avec l'accord du client.

Lien du composant : https://www.sparkfun.com/products/16467

3.5 ESP32

J'ai repris un ESP32 déjà utilisé pour plusieurs projets à l'ETML-ES, il s'agit du « ESP32-C3-WROOM-02-N4 »

Lien du composant : https://www.digikey.ch/fr/products/detail/espressif-systems/ESP32-C3-WROOM-02-N4/14553031

[7] Espressif Systems. « ESP32-C3-WROOM-02, ESP32-C3-WROOM-02-U : Datasheet », [en ligne], (consulté le 20 août 2024), version 1.3, Table 12, p.21. URL : https://www.espressif.com/sites/default/files/documentation/esp32-c3-wroom-02_datasheet_en.pdf

3.6 Boitier

Dans mon CDC, il est demandé soit de choisir un boitier, soit de le réaliser soi-même. J'ai préféré réaliser mes boitiers sur Solidworks (boitier pour le PCB et boitier pour le capteur de température/humidité). Cette méthode me permet de concevoir un boitier adapté sans besoin d'usinage. La charge de réalisation sera compensée par le gain de temps d'usinage d'un boitier standard.

Cependant, il est important de noter que certains des filaments 3D absorbent de l'humidité, ce qui pourrait être un problème avec mon capteur. Les trois matériaux principalement utilisés, sont le PLA, l'ABS et le PETG. Le PLA ne conviendrait pas à cette application, car il peut absorber l'humidité[8], ce qui pourrait fausser les mesures de mon capteur d'humidité. L'ABS quant à lui, peut contenir des substances toxiques[9]. Dans le cas où mon capteur serait pour un réfrigérateur contenant des médicaments ou aliments, ceci n'est absolument pas recommandé. Le PETG serait le filament à privilégier dans ce cas, il est résistant à l'humidité[10] et peut-être utilisé pour de l'alimentaire[11].

Malheureusement, l'ETML-ES dispose simplement de filament ABS, pour le prototype j'utiliserai donc celui-ci, mais s'il faut un jour en produire, il faudra privilégier le PETG.

Les informations concernant le boitier, se trouveront au point 6.7, p.28.

4 Estimation du coût

Composants	Quantité	Numéro de référence	Prix unitaire (par composant) [CHF]	Total (par composant) [CHF]
PCB	1	-	40	40
Microcontrôleur	1	STM32F072C8T6TR	4.26	4.26
E-paper	1	24963573	16.9	16.9
ESP32	1	ESP32-C3-WROOM-02-N4	1.69	1.69
Sparkfun-16467	1	SEN-16467	9.53	9.53
Composants divers	30	-	1.2	36
Prix total			**73.58**	**108.38**

Tableau 4 Estimation du coût

[8] Robert. Karl-Emerik. « Le Filament PLA et l'Humidité : Comprendre et prévenir les Problèmes. », [en ligne], (consulté le 21 août 2024), publié le 18 juin. URL : https://www.make3dprinting.com/post/le-filament-pla-et-l-humidit%C3%A9-comprendre-et-pr%C3%A9venir-les-probl%C3%A8mes

[9] Sculpteo. « Compatibilité alimentaire des impressions 3D : Quels matériaux choisir ? », [en ligne], (consulté le 21 août 2024). URL : https://www.sculpteo.com/fr/centre-apprentissage/choisissez-le-meilleur-materiau-impression-3d/compatibilite-alimentaire-impression-3d/

[10] Angelini. Oliver. « Le filament PETG résiste-t-il à l'humidité ? », [en ligne], expert-scan3d.fr, (consulté le 21 août 2024). URL : https://www.expert-scan3d.fr/le-filament-petg-resiste-t-il-a-lhumidite/

[11] Robert. Karl-Emerik. « Guide Complet sur le Filament PETG : Propriétés, Utilisations et Avantages pour l'Impression 3D », [en ligne], (consulté le 21 août 2024), publié le 6 mai. URL : https://www.machine3d.fr/post/guide-complet-sur-le-filament-petg-propri%C3%A9t%C3%A9s-utilisations-et-avantages-pour-l-impression-3d

5 Schématique

5.1 Schéma bloc

Durant la conception de mon schéma, j'ai découpé celui-ci en plusieurs feuilles et établi une hiérarchie à l'aide d'un schéma bloc. Cette approche facilite la compréhension rapide du fonctionnement global de mon système.

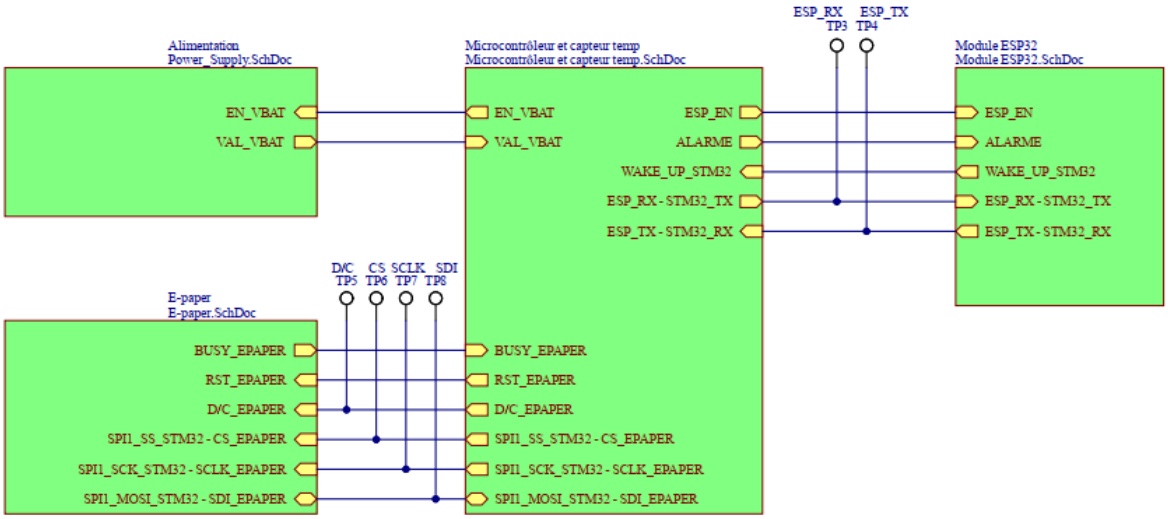


Figure 3 Schéma bloc sous Altium

J'ai rajouté les noms des signaux, ce qui sera également le cas sur le PCB. Cela permet de faciliter la compréhension et les tests.

5.2 Alimentation

5.2.1 Entrée alimentation V_Piles et régulateur de tension

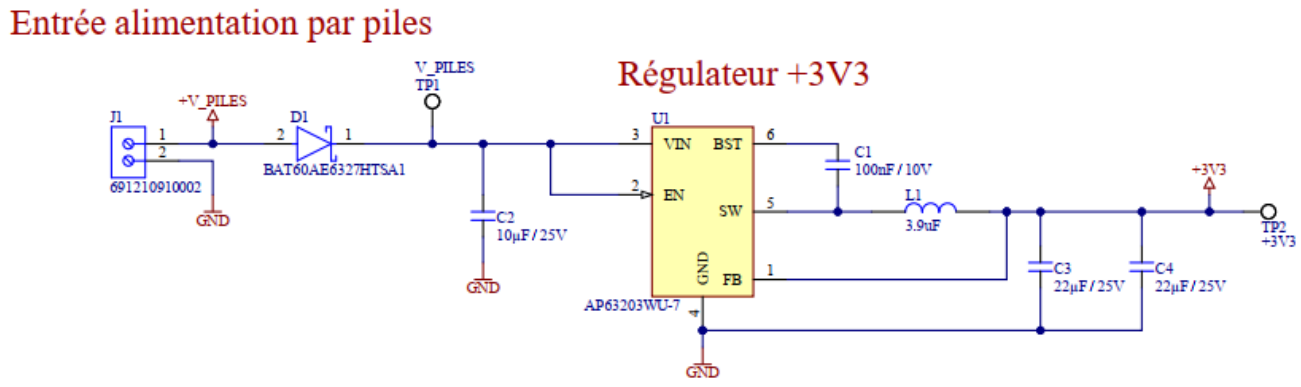


Figure 4 Schéma représentant l'entrée V_PILES ainsi que le régulateur de tension

Pour l'alimentation, j'ai pris une alimentation à découpage, comme souhaité dans le CDC. J'ai choisi de réutiliser le même régulateur que j'avais déjà utilisé dans un précédent projet (2320) et qui fonctionnait correctement.

Pour commencer, j'ai pris une diode ayant une très faible tension directe de l'ordre de 0.3[V] max[12]. Cela me permet de garantir une tension supérieure à 3,8[V][13] sur le pin VIN de mon régulateur si j'ai une tension d'alimentation (V_PILES) de 4.5[V].

Concernant le montage, le datasheet indiquait un montage pour une application typique :

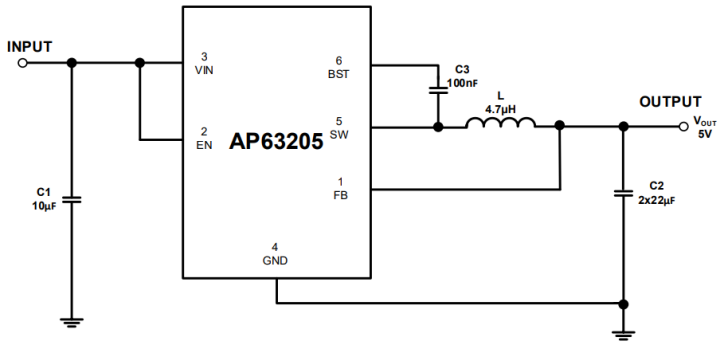


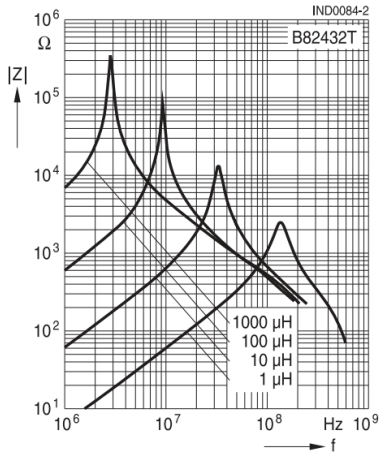
Figure 5 Application typique du circuit[14]

En référence à la figure 5, la valeur des composants à utiliser est directement indiquée dans le datasheet comme suit :

AP63203				
Output Voltage (V)	**L (µH)**	**C1 (µF)**	**C2 (µF)**	**C3 (nF)**
3.3	3.9	10	2 x 22	100

Figure 6 Valeur des composants recommandés[15]

Pour l'inductance que j'ai choisie, j'ai vérifié que sa fréquence soit bien dans la gamme de celle de mon régulateur de tension, qui est de l'ordre de 1100[kHz][16].



A l'aide de la figure 7, je peux constater que mon impédance sera inférieure à 10[Ω] en prenant la courbe de 10[µH], ce qui garantit que mon système sera bel et bien stable au niveau de la tension de sortie.

Figure 7 Courbe impédance en fonction de la fréquence à +20[°C][17]

[12] Infineon. « BAT60A… », [en ligne], (consulté le 20 août 2024), datasheet du 2007-04-19, DC Characteristics (Forward Voltage), p. 2. URL : https://www.infineon.com/dgdl/bat60aseries.pdf?folderId=db3a304313d846880113def5812204a1&fileId=db3a304313d846880113def70c9304a9

[13] Diodes incorporated, « AP63200/AP63201/AP63203/AP63205 », [en ligne], (consulté le 20 août 2024), révision 2-2, Recommanded Operating Conditions (Vin), p.4. URL : https://www.diodes.com/assets/Datasheets/AP63200-AP63201-AP63203-AP63205.pdf

[14] *Ibid.,* p.2

[15] *Ibid.,* table 1, p.13

[16] *Ibid*., Electrical Characteristics (FSS), p.5

[17] TDK, « STM inductors », [en ligne], (consulté le 20 août 2024), datasheet de juin 2019, Impedance |Z| versus frequency f, p. 6. URL : https://www.tdk-electronics.tdk.com/inf/30/db/ind_2008/b82432t.pdf

5.2.2 Contrôle état de la pile

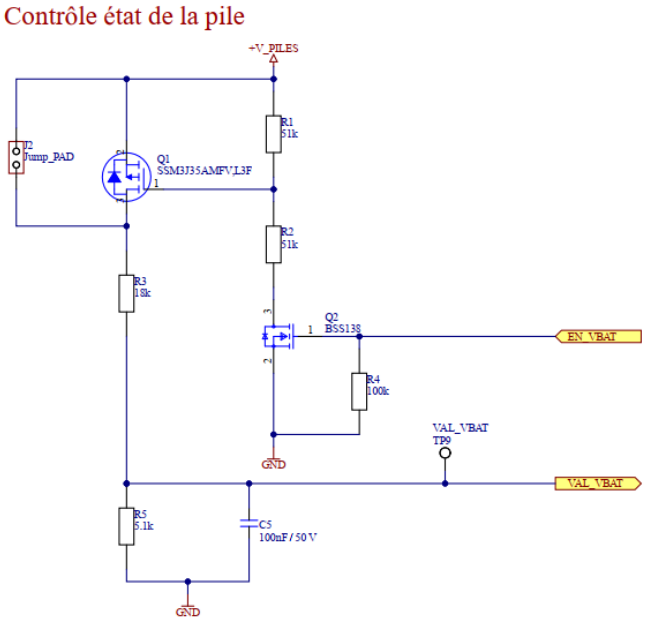


Figure 8 Schéma représentant le contrôle de la pile

J'ai réutilisé le schéma de mon ancien projet (2320) pour mesurer l'état de la pile. Pour cela, la pin « EN_VBAT » du microcontrôleur applique une tension 3.3[V] à la grille de Q2 (BSS138), activant ainsi le MOSFET. Cela abaisse la tension sur la grille de Q1 (SSM3J35AMFV,L3F), permettant son activation.

La résistance R1 agit comme une pull-up, garantissant que Q1 reste bloqué tant que Q2 n'est pas en conduction, tandis que R4 assure un état bas lorsque le microcontrôleur est dans un état indéfini.

Les résistances R3 et R5, ont été dimensionnées pour garantir un fonctionnement dans la plage de tension de l'ADC (0 à 3.6[V])[18] du STM32.

$$VAL_{VBAT} = \frac{V_{PILES}}{R_5 + R_3} \cdot R_5 = \frac{4.5}{5.1 \cdot 10^3 + 18 \cdot 10^3} \cdot 5.1 \cdot 10^3 = 0.99[V]$$

De plus, si on décide de mettre une pile supplémentaire pour augmenter la tension à 6[V], en reprenant le même calcul, nous aurons une tension de 1.32[V] sur « VAL_VBAT », ce qui reste dans la plage de l'ADC.

Enfin, j'ai vérifié que les MOSFETs conduisent correctement. Pour Q2, la tension VGS typique est de 1.3[V][19], donc avec une tension de 3,3[V] appliquée par la pin « EN_VBAT », je garantis qu'il conduit correctement.

Pour le MOSFET Q1, j'ai effectué un calcul pour vérifier si la tension VGS était négative et si elle correspondait aux conditions d'activation spécifiées dans le datasheet. Pour cela j'ai utilisé la formule suivante :

$$V_{GS} = V_G - V_{PILES} = 2.25 - 4.5 = -2.25[V]$$

[18] *Ibid.*, p.1
[19] ONSEMI, « N-Channel Logic Level Enhancement Mode Field Effect Transistor : BSS138 », *op. cit*.

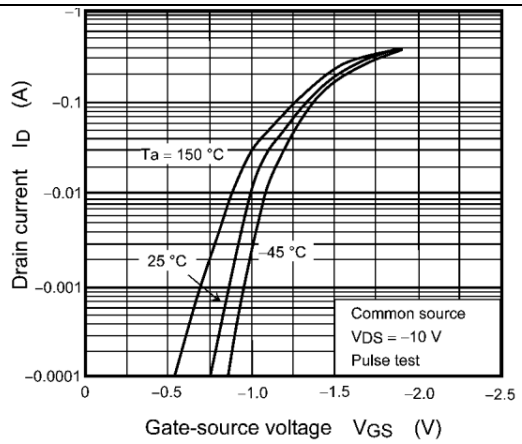


Figure 9 Tension VGS en fonction du courant ID

La courbe du datasheet[20] montre qu'avec une tension de -2.25[V], je permets au MOSFET de conduire correctement.

5.3 Microcontrôleur et capteur température/humidité

5.3.1 Reset et debug

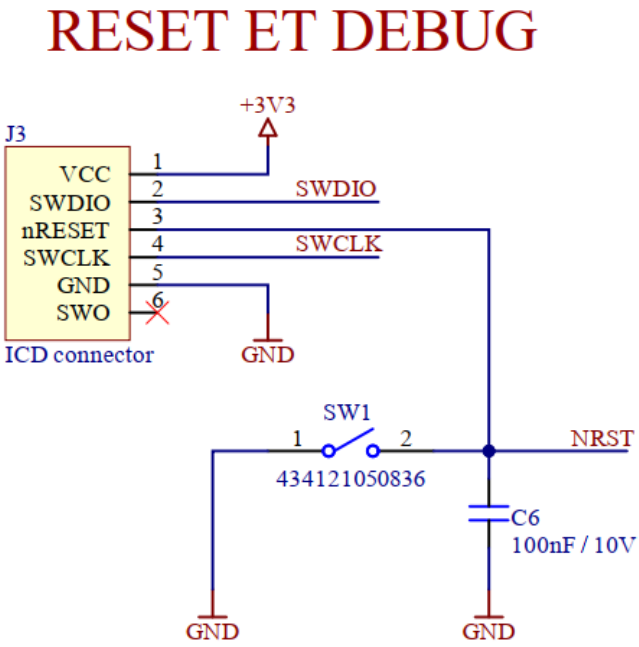


Figure 10 Schéma reset et debug

J'ai opté pour l'utilisation d'un « tag connector », que j'avais également utilisé lors de mon précédent projet (2320).

Il est pratique en raison de sa petite taille et contient les pins nécessaires à la programmation.

Pour déterminer les pins de connexion, j'ai utilisé le lien suivant : https://www.tag-connect.com/wp-content/uploads/bsk-pdf-manager/TC2030-CTX_1.pdf

Concernant la connexion du bouton, j'ai suivi la figure 25 se trouvant dans le datasheet du STM32[21].

[20] TOSHIBA, « SSM3J35AFV » [en ligne], (consulté le 20 août 2024), datasheet du 22 juin 2017, révision 2.0, figure 7.2, p.5. URL: https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/754/SSM3J35AMFV.pdf
[21] ST, « STM32F072x8 STM32F072xB », *op. cit*. p.84

5.3.2 STM32

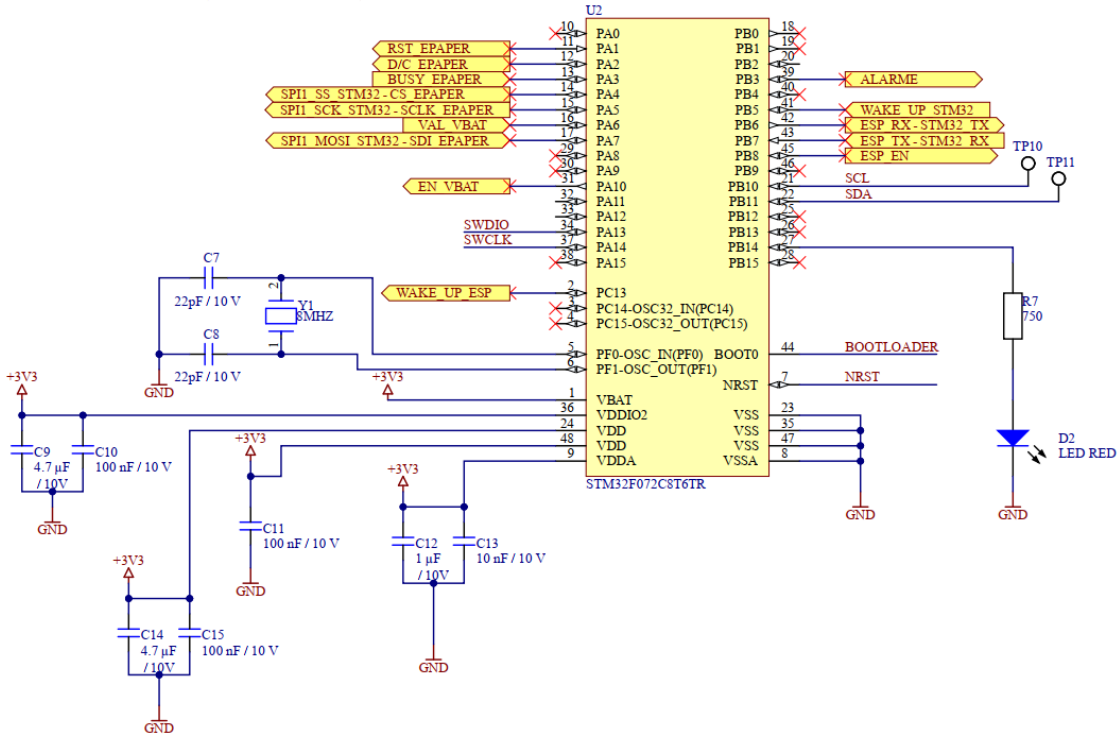


Figure 11 Schéma STM32

Etant donné que j'utilise des communications UART, SPI et I2C, j'ai ajouté un quartz externe pour obtenir une meilleure précision par rapport à celle du quartz interne du STM32.
Les valeurs des condensateurs pour le quartz ont été calculées avec : CL = 10 [pF] et C0 = 7[pF][22]

$$C_{21} = C_{22} = 2 \cdot (18 \cdot 10^{-12} - 7 \cdot 10^{-12}) = 22[pF]\ (E6)$$

Équation 1 Calcul pour définir les condensateurs de découplage du quartz[23]

La LED rouge joue le rôle d'indicateur de vie, comme expliqué au point 2.2.4 p.8.

Pour le dimensionnement de la résistance R7, j'ai utilisé la formule suivante :

$$R_{LED\_ROUGE} = \frac{U_{\mu C} - U_{typ_{LED}}}{I_F} = \frac{3.3 - 1.8}{2 \cdot 10^{-3}} = 750[\Omega]\ (E24)$$

La valeur typique pour la tension de la LED se trouve dans son datasheet[24].

[22] ABLS. « HC/49US SMD LOW PROFILE CRYSTAL », [en ligne], (consulté le 20 août 2024). URL : https://abracon.com/Resonators/ABLS.pdf, Standard Specifications (Shunt capacitance & Load Capacitance), p.1
[23] MICROCHIP. « Calculating crystal load capacitor », [en ligne], (consulté le 20 août 2024), publié le 11 septembre 2020. URL : https://microchip.my.site.com/s/article/Calculating-crystal-load-capacitor
[24] SunLED. « XZCM2CRK54WA-1VF », [en ligne], (consulté le 20 août 2024), Operating Characteristics (Forward Voltage Typ), p.1. URL : https://www.sunledusa.com/products/spec/XZCM2CRK54WA-1VF.pdf

Pour la configuration des pins, le fabricant « Waveshare » indiquait la configuration des pins entre le STM32 et le e-paper de la manière suivante :

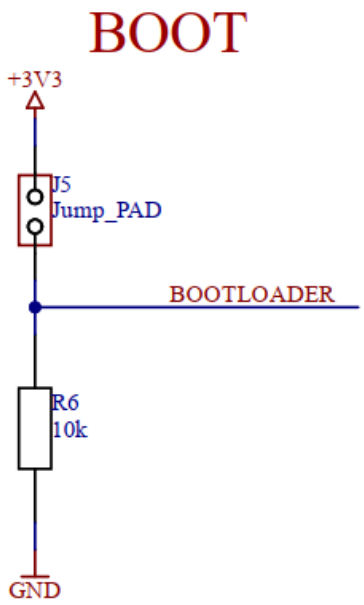
STM32 Pinout

e-Paper	STM32
VCC	3.3V
GND	GND
DIN	PA7
CLK	PA5
CS	PA4
DC	PA2
RST	PA1
BUSY	PA3

Figure 12 Configuration des pins entre le STM32 et l'e-paper[25]

Pour terminer, tous les condensateurs de découplage utilisés sont indiqués dans son datasheet[26].

5.3.3 Chargeur d'amorçage (Bootloader)



Le bootloader est prévu pour une phase de production, il ne sera pas utilisé pour le prototype

Figure 13 Schéma du bootloader pour le STM32

Le STM32 dispose d'une broche appelée « BOOT0 » qui est utilisée pour sélectionner le mode de démarrage.

J'ai tout de même implémenté le boot0 même s'il ne sera pas utilisé pour le prototype. Au cas où le client ait envie d'effectuer différents tests, cette option sera disponible.

J'ai consulté une page de stm32world qui explique qu'il est essentiel de connecter la broche BOOT0 à la masse lors du reset pour assurer un fonctionnement normal.

La page recommande l'utilisation d'une résistance entre 10[kΩ] et 1 [MΩ], j'ai donc repris celle de mon ancien projet (2320).

Dans le schéma, à la place du jumper pad, il y avait un switch, cependant celui-ci est facultatif. J'ai donc préféré utiliser un pad SMD qui économise de la place sur le PCB.

[25] WAVESHARE. « 2.13inch e-Paper HAT Manual », [en ligne], (consulté le 20 août 2024), Working With STM32 (Hardware Connection). URL : https://www.waveshare.com/wiki/2.13inch_e-Paper_HAT_Manual#Working_With_STM32

[26] ST, « STM32F072x8 STM32F072xB », *op. cit*. p.49

5.3.4 Capteur température et humidité

Capteur SparkFun (T et H)

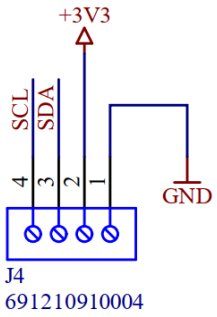


Figure 14 Schéma du capteur de température et humidité

Comme indiqué lors de ma pré-étude, j'ai pris un kit de SparkFun comme capteur.

Vu que le PCB de SparkFun sera déporté par câble plat, j'ai décidé de mettre un bornier fixe, ce qui permettra d'assurer une meilleure connexion et de retirer plus facilement les câbles lors de la mise en boitier.

La carte indique qu'elle contient déjà des résistances de pull-up d'une valeur de 2.2[kΩ][27] pour les signaux I2C, je n'en ai donc pas implémenté dans mon schéma.

Pour savoir quel mode d'I2C je peux utiliser entre le mode standard, rapide ou rapide plus, j'ai calculé la capacité que j'aurai pour un temps de montée de 500[ns]. N'ayant pas accès au temps de montée minimum pour le mode standard, j'ai pris une marge par rapport au temps maximum du mode rapide qui est de 300[ns][28].

Parameter	Symbol	Conditions	Standard-mode		Fast-mode		Fast-mode Plus		Units
			Min.	Max.	Min.	Max.	Min.	Max.	
SCL clock frequency	f_{SCL}	-	0	100	0	400	0	1000	kHz
Hold time (repeated) START condition	$t_{HD;STA}$	After this period, the first clock pulse is generated	4.0	-	0.6	-	0.26	-	µs
LOW period of the SCL clock	t_{LOW}	-	4.7	-	1.3	-	0.5	-	µs
HIGH period of the SCL clock	t_{HIGH}	-	4.0	-	0.6	-	0.26	-	µs
Set-up time for a repeated START condition	$t_{SU;STA}$	-	4.7	-	0.6	-	0.26	-	µs
SDA hold time	$t_{HD;DAT}$	-	0	-	0	-	0	-	µs
SDA set-up time	$t_{SU;DAT}$	-	250	-	100	-	50	-	ns
SCL/SDA rise time	t_R	-	-	1000	20	300	-	120	ns
SCL/SDA fall time	t_F	-	-	300	20 x (V_{DD} / 5.5V)	300	20 x (V_{DD} / 5.5V	120	ns
SDA valid time	$t_{VD;DAT}$	-	-	3.45	-	0.9	-	0.45	µs
Set-up time for STOP condition	$t_{SU;STO}$	-	4.0	-	0.6	-	0.26	-	µs
Capacitive load on bus line	C_B	-	-	400	-	400	-	550	pF

Figure 15 Temps de communication pour le capteur SHTC3[29]

J'ai ensuite calculé ma capacité à l'aide de la formule tirée du site de Texas Instruments[30] :

$$R_{pmax} = \frac{t_r}{(0.8473 \cdot C_b)} => C_b = \frac{t_r}{0.8473 \cdot R_{pmax}} = \frac{500 \cdot 10^{-9}}{0.8473 \cdot 2.2 \cdot 10^3} = 268.23[pF]$$

Comme je me situe en dessous de 400[pF], cela signifie que je peux utiliser n'importe quel mode parmi les trois proposés sur la figure 15.

[27] SparkFun. « SparkFun Humidity Sensor Breakout - SHTC3 (Qwiic) Hookup Guide », [en ligne], (consulté le 20 août 2024), chapitre jumpers. URL : https://learn.sparkfun.com/tutorials/sparkfun-humidity-sensor-breakout---shtc3-qwiic-hookup-guide#hardware-overview

[28] SENSIRION. « Datasheet SHTC3 : Humidity and Temperature Sensor IC », [en ligne], (consulté le 29 août 2024), version 2 – juin 2019, table 6, SCL/SDA rise time (tr : Fast-mode max), p.5. URL : https://cdn.sparkfun.com/assets/1/1/f/3/b/Sensirion_Humidity_Sensors_SHTC3_Datasheet.pdf

[29] *Ibid.*, p.5

[30] TEXAS INSTRUMENTS. « I2C Bus Pullup Resistor Calculation », [en ligne], (consulté le 29 août 2024), Pullup Resistor Calculation (point 5), p.2. URL : https://www.ti.com/lit/an/slva689/slva689.pdf?ts=1724585444033

Remarque : il serait intéressant de connaitre la longueur maximale de mon câble, cependant, je n'ai pas trouvé de norme définissant la longueur maximale d'une liaison I2C. Toutefois, il est important de noter que plus la vitesse de communication est élevée, plus le câble doit être court afin d'éviter des perturbations.

5.4 ESP-32

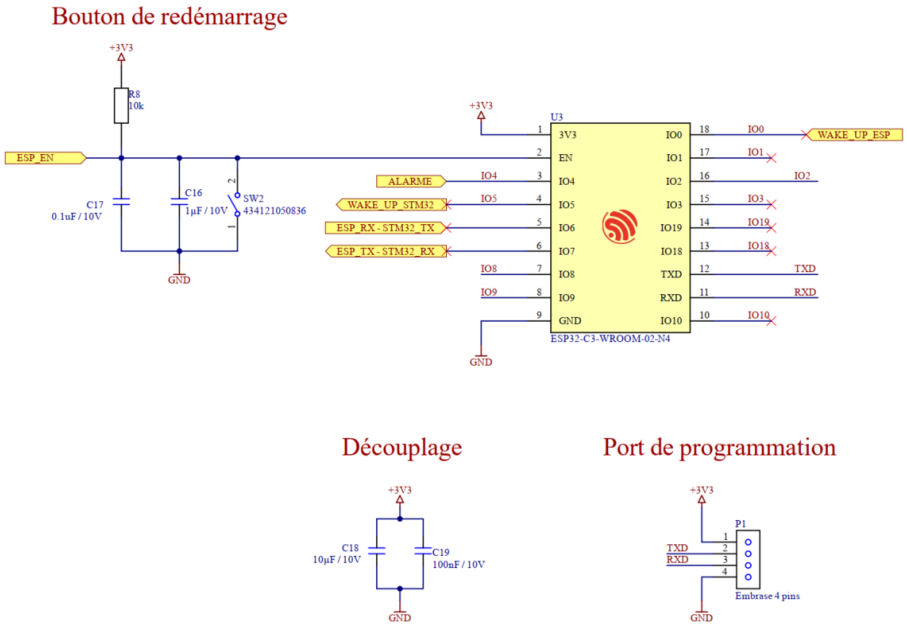


Figure 16 Schéma pour l'ESP32

Pour le schéma de l'ESP32, je me suis basée sur le schéma typique indiqué dans le datasheet[31].

La pin ALARME servira à envoyer un « 1 » si une valeur mesurée (température ou humidité) se situe au-delà des seuils souhaités. J'ai également prévu une pin « WAKE_UP_STM32 » pour permettre à l'ESP32 de réveiller le STM32, ainsi qu'une pin « WAKE_UP_ESP322 » en sens inverse au cas où le STM32 aurait besoin de réveiller l'ESP32.

5.4.1 UART0 et UART1

L'ESP32 contient deux UART : le premier, UART0, est utilisé pour programmer l'ESP32 avec un PC. Le second, UART1, est utilisé pour la communication entre l'ESP32 et le microcontrôleur afin d'envoyer ou recevoir des données.
Les informations pour la configuration des pins des UART viennent du site d'Espressif :

Function of Connection	ESP32-C3 Board or Module Pins	Other Device Pins
Download/Log output [1]	UART0 • GPIO20 (RX) • GPIO21 (TX)	PC • TX • RX
AT command/response [2]	UART1 • GPIO6 (RX) • GPIO7 (TX) • GPIO5 (CTS) • GPIO4 (RTS)	USB to serial converter • TX • RX • RTS • CTS

Figure 17 Configuration des pins UART de l'ESP32[32]

[31] Espressif Systems. « ESP32-C3-WROOM-02, ESP32-C3-WROOM-02-U : Datasheet », [en ligne], (consulté le 20 août 2024), version 1.3, figure 7, p.25. URL : https://www.espressif.com/sites/default/files/documentation/esp32-c3-wroom-02_datasheet_en.pdf
[32] Espressif. « Hardware Connection », [en ligne], (consulté le 20 août 2024), master(latest), ESP32-C3 Series Hardware Connection Pinout. URL : https://docs.espressif.com/projects/esp-at/en/latest/esp32c3/Get_Started/Hardware_connection.html

Remarque : les pins CTS et RTS sont optionnelles, comme l'indique la note 2 située en dessous du tableau « *Series Hardware Connection Pinout* » trouvé dans le datasheet.

En comparant le tableau de la figure 17 et celui du datasheet[33], j'ai pu obtenir la configuration des pins présentées dans le tableau ci-dessous :

Nom	N° pin	Type	Fonction
GPIO6	5	I/O/T	UART1 (RX)
GPIO7	6	I/O/T	UART1 (TX)
GPIO20	11	I/O/T	UART0 (RX)
GPIO21	12	I/O/T	UART0 (TX)

Tableau 5 Définition des pins UART0 et UART1

Note: I : Input, O : Output, T = high impedance.

5.4.2 Pin ESP_EN

La pin 2 « ESP_EN » est la pin d'activation du composant. Lorsqu'elle est à l'état haut, l'ESP32 s'active, et à l'état bas, il se désactive.
Le datasheet recommande l'ajout d'un filtre RC pour garantir une alimentation stable lors du démarrage, en indiquant les valeurs de la résistance et du condensateur (R8 et C16 dans mon schéma) :

> To ensure that the power supply to the ESP32-C3 chip is stable during power-up, it is advised to add an RC delay circuit at the EN pin. The recommended setting for the RC delay circuit is usually R = 10 kΩ and C = 1 μF. However, specific parameters should be adjusted based on the power-up timing of the module and the power-up and reset sequence timing of the chip. For ESP32-C3's power-up and reset sequence timing diagram, please refer to *ESP32-C3 Series Datasheet* > Section *Power Scheme*.

Figure 18 Information réinitialisation filtre RC[34]

5.4.3 Port de programmation

Pour programmer mon ESP32, on m'a fourni le câble de programmation suivant : TTL-232R-3V3 du fabricant FTDI. Ce câble possède six pins femelles, comme le montre la figure ci-dessous, cependant, je n'ai dans mon cas besoin que des pins suivantes : TXD, RXD et GND.

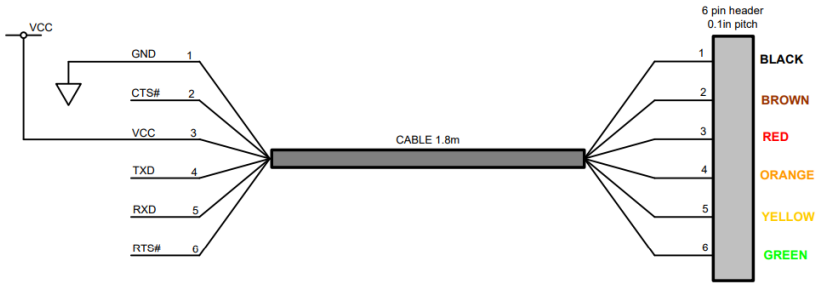


Figure 19 Liaison du câble TTL-232R-3V3[35]

Remarque : il est important de ne pas connecter le VCC au connecteur de programmation, car cela risquerait de fournir deux alimentations à l'ESP32, ce qui pourrait empêcher la programmation.

[33] Espressif Systems. « ESP32-C3-WROOM-02, ESP32-C3-WROOM-02-U: Datasheet », *op. cit*. table 3, p10-11
[34] *Ibid.*, figure 7, p. 25
[35] FTDI. TTL-232R TTL TO USB SERIAL CONVERTER RANGE OF CABLES Datasheet », [en ligne], (consulté le 29 août 2024), version 2.05, figure 4.1, p.10. URL : https://ftdichip.com/wp-content/uploads/2023/07/DS_TTL-232R_CABLES.pdf

5.4.4 Sélection de mode

Selon le datasheet, les GPIO2, GPIO8 et GIPO9 activent le mode « Download Boot » pour programmer l'ESP32. J'ai ajouté les résistances de pull-up sur les GPIO2 et GPIO8 pour les maintenir à l'état haut (leurs valeurs sont également indiquées sur le schéma typique mentionné au point précédent). La pin GPIO9 ayant une pull-up interne, elle nécessite un switch (SW2) relié à la masse pour passer à l'état bas et respecter les conditions de la figure suivante :

Booting Mode [1]			
Pin	Default	SPI Boot	Download Boot
GPIO2	N/A	1	1
GPIO8	N/A	Don't care	1
GPIO9	Internal weak pull-up	1	0

Figure 20 Mode de démarrage pour l'ESP32[36]

Il suffit donc avoir les pins GPIO 2 et 8 à l'état haut lors du démarrage, comme indique la figure 20.

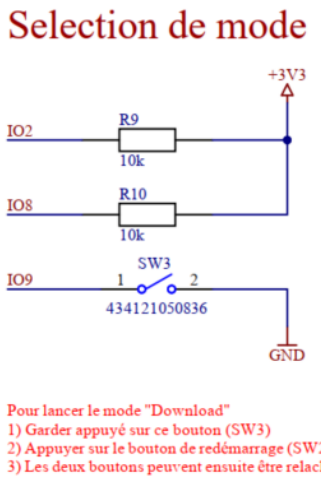


Figure 21 Schéma sélection du mode de l'ESP32

C'est avec ce circuit que nous pourrons lancer le mode de téléchargement du firmware. Pour cela, il faut maintenir le bouton SW3 et redémarrer le module avec le bouton SW2 de la figure 21.

Les broches IO2 et IO8 peuvent être à l'état haut en utilisant les résistances de pull-up de 10[kΩ], comme indiqué sur le schéma du datasheet[37].

Les détails complets pour la programmation se trouveront dans la partie Firmware, au point 9 p.30.

5.5 E-paper

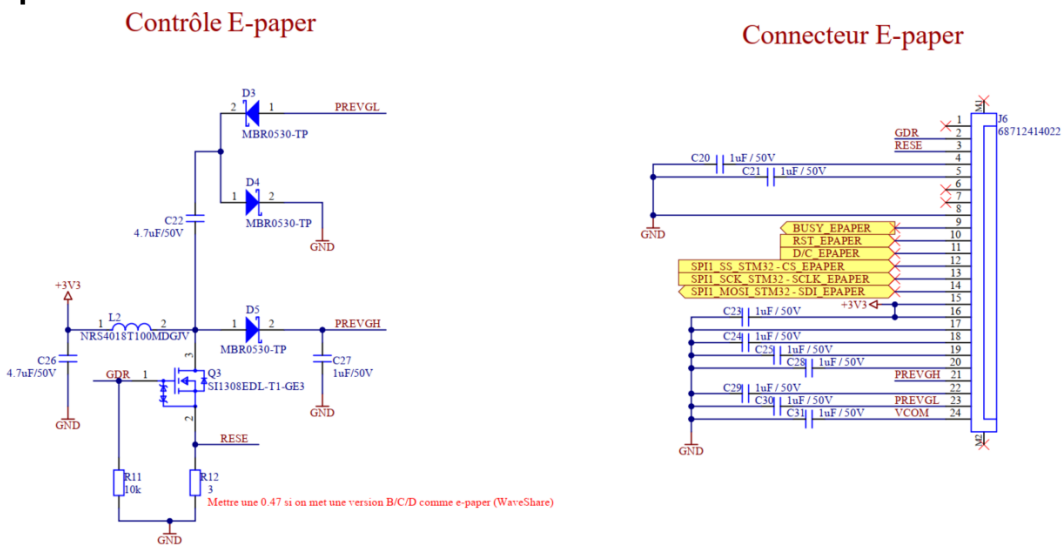


Figure 22 Schéma pour le e-paper

J'ai réalisé ce schéma à l'aide de deux datasheets différents. La première étant la version 2, dédiée aux écrans noir et blanc, tandis que la version V4 est destinée à tous les types d'e-paper.

[36] Espressif Systems. « ESP32-C3-WROOM-02, ESP32-C3-WROOM-02-U : Datasheet », *op. cit*. table 4, p. 12
[37] *Ibid*., figure 7, p.25

La V2 contient des indications supplémentaires sous le schéma de référence[38] contrairement à la V4.

Les connexions concernant le connecteur n'étaient pas les mêmes pour les deux versions. J'ai rajouté le condensateur C20 sur la pin4 du connecteur selon les indications de la V4[39], qui indiquait cela sur le schéma de référence. En revanche, la V2 indiquait que la pin 4 du connecteur était non connectée. Ne sachant pas vraiment si ce condensateur serait nécessaire pour mon e-paper, ou seulement utile pour les autres versions B/C/D, j'en ai discuté avec le client, qui m'a conseillé de prévoir une empreinte sur le PCB au cas où nous en aurions besoin.

La pin 8 était également différente sur les deux schémas de référence. Cependant, une note dans chaque datasheet indiquait que la pin servait à définir, l'interface SPI. Un bus à quatre lignes si la pin est reliée à la masse, ou un bus en mode trois lignes si celle-ci est à l'état haut.

BS1 State	MCU Interface
L	4-lines serial peripheral interface(SPI) - 8 bits SPI
H	3- lines serial peripheral interface(SPI) - 9 bits SPI

Figure 23 Indication concernant la pin 8[40]

En utilisant l'option à quatre lignes, j'utilise une pin supplémentaire sur mon microcontrôleur, mais cela me permet de distinguer les commandes de données. En mode trois lignes, on économise une pin du microcontrôleur, mais la distinction doit être gérée différemment. J'ai opté pour la version quatre lignes qui me paraissait plus simple à utiliser, et aussi plus simple à comprendre lors de l'analyse des signaux sur un oscilloscope.

Concernant le montage de gauche sur la figure 22, les deux datasheets indiquaient les mêmes composants à l'exception de l'inductance, où la V2 indiquait 10[µH] tandis que la V4 indiquait une inductance de 68 [µH]. Après discussion avec le client, nous avons préféré mettre une inductance de 10[µH] comme indiqué dans la V2 du datasheet, car il concernait principalement les e-paper noir et blanc. De plus, un ancien projet à l'ETML-ES (2017) avec un e-paper utilisait également ce schéma avec une inductance de 10[µH]. Cette différence ne devrait pas avoir d'influence sur le fonctionnement du circuit. Si nécessaire, nous pourrons toujours remplacer l'inductance par une 68[µH].

Concernant la résistance R12, j'ai pris la résistance indiquée pour ma version, soit la A.
A titre informatif, la version B correspond aux modèles de e-paper pouvant afficher du rouge en plus du noir et blanc. Tandis que la version C peut aussi afficher du jaune et la version D est un e-paper flexible[41].

Le schéma complet du projet se trouve en annexe E

[38] WAVESHARE. « 2.13inch e-Paper », [en ligne], (consulté le 20 août 2024), version 2, Reference Circuit (note), p.8. URL : https://files.waveshare.com/upload/d/d5/2.13inch_e-Paper_Specification.pdf
[39] WAVESHARE. « 2.13inch e-Paper », [en ligne], (consulté le 20 août 2024), version 4, Reference Circuit, p.28. URL : https://files.waveshare.com/upload/4/4e/2.13inch_e-Paper_V4_Specification.pdf
[40] *Ibid*., note 5-5, p.8
[41] WAVESHARE. « 250x122, 2.13inch E-Ink raw display panel », [en ligne], (consulté le 20 août 2024), Selection Guide (2.13inch e.Paper). URL : https://www.waveshare.com/2.13inch-e-paper.htm

6 Hardware

6.1 Dimension du PCB

N'ayant pas de contrainte au niveau de la taille de PCB, j'ai décidé de l'adapter par rapport à mon e-paper, mais également à mon support de pile, qui sera placé en dessous du PCB. Le e-paper mesurant 59.20x29.20[mm][42], j'ai ajouté une marge pour la longueur passant ainsi de 59.20 à 62[mm]. Concernant la largeur, mon support de pile faisant du 52.7x37.5[mm][43], j'ai mis une marge pour avoir 40[mm] en largeur.
J'ai également découpé une zone du PCB en laissant une marge pour y placer le connecteur de mon e-paper. Cela me permet de minimiser le dépassement du câble hors de la carte, car le e-paper sera posé sur la face avant de mon PCB et le connecteur à l'arrière.

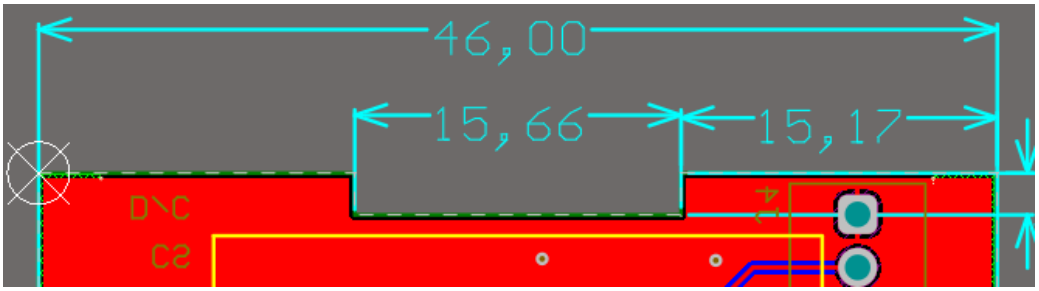


Figure 24 Coupure du PCB pour le câble du e-paper

Ayant mis 46[mm] pour la largeur, j'ai adapté au mieux ma coupure pour que mon e-paper soit bien centré au milieu du PCB.

6.2 Zone interdite (keepout)

J'ai ajouté deux zones de keepout de 62x3[mm] pour permettre au PCB de glisser dans son boitier.

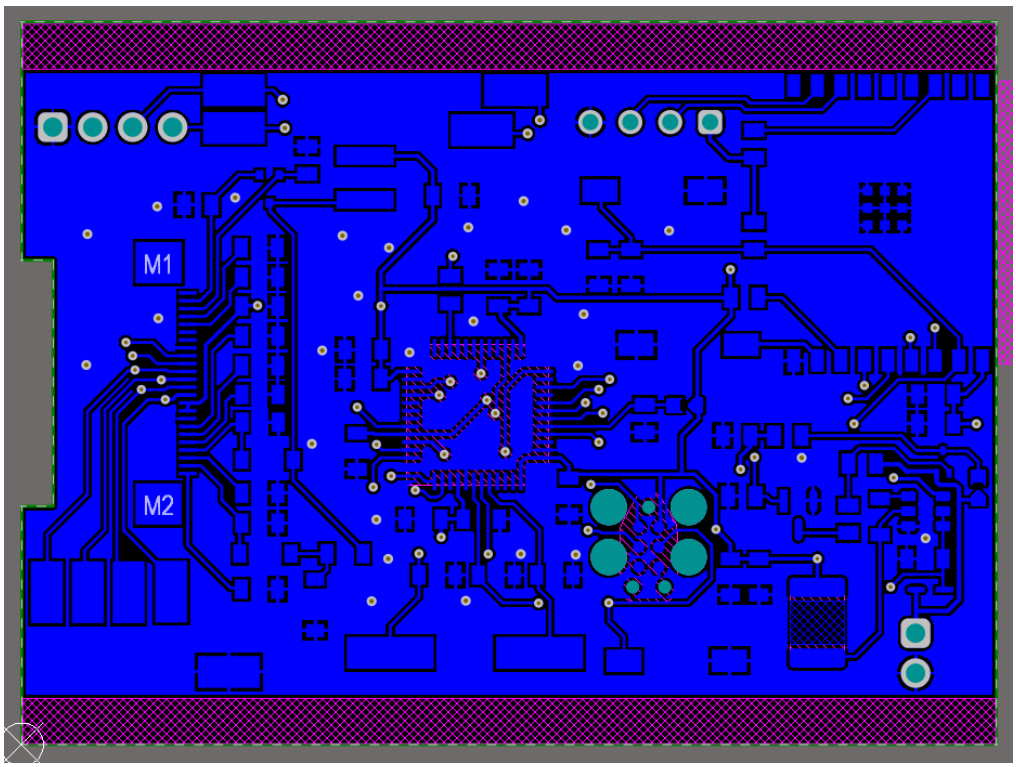


Figure 25 Zone de keepout réservée pour insérer le PCB dans le boîtier

[42] WAVESHARE. « 2.13inch e-Paper », version 4, *op. cit*. p.6
[43] KEYSTONE. « ECONOMICAL PLASTIC BATTERY HOLDERS », [en ligne], (consulté le 26 août 2024), Three (3) "AAA" Cells - In Series, p.1. URL : https://www.keyelco.com/userAssets/file/M65p28.pdf

6.3 Placement des composants

J'ai placé mes composants de façon à faciliter le plus possible mon routage. J'ai tout simplement suivi mon schéma en essayant de regrouper les blocs ensembles tout en ayant un routage simple. Cependant, les points de tests concernant les signaux ont été placés à l'arrière du PCB pour faciliter les tests durant la phase de programmation. J'ai laissé les points de tests d'alimentation côté avant, car ils ne seront pas tout le temps utilisé. De plus, j'ai également placé ma LED rouge sur le côté avant pour avoir un visuel dessus au cas où les piles seraient à changer.

Concernant l'ESP32, il y avait cependant une contrainte, puisque le fabricant recommande un placement de l'antenne en bordure de PCB, afin qu'elle soit éloignée de toute interférence avec les autres composants[44]. Cela permet de garantir une performance optimale de l'antenne en minimisant les risques d'interférences. Par conséquent, j'ai disposé le module comme le recommande le fabricant :

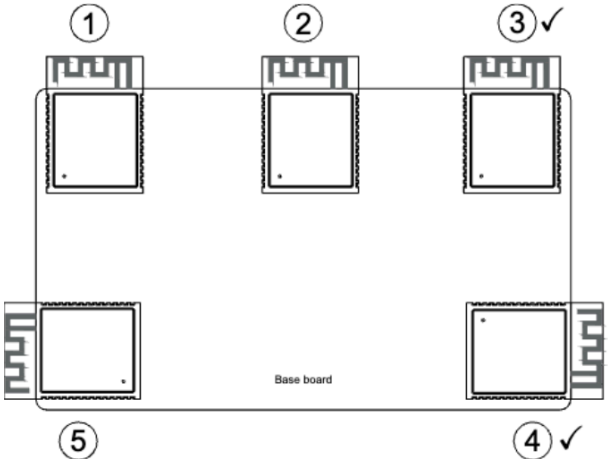


Figure 26 Recommandation de placement pour l'ESP32

Il était également indiqué que si l'espace ne permet pas de placer le module au bord du circuit, il est essentiel de découper le PCB afin d'éviter la présence de cuivre autour de l'antenne[45]. Cela permet de créer une zone de dégagement minimisant les interférences avec l'antenne. Je n'ai pas eu besoin de cette découpe dans mon cas, mais je tenais à inclure cette information au cas où quelqu'un souhaiterait à l'avenir reprendre le projet et refaire le PCB de manière différente.

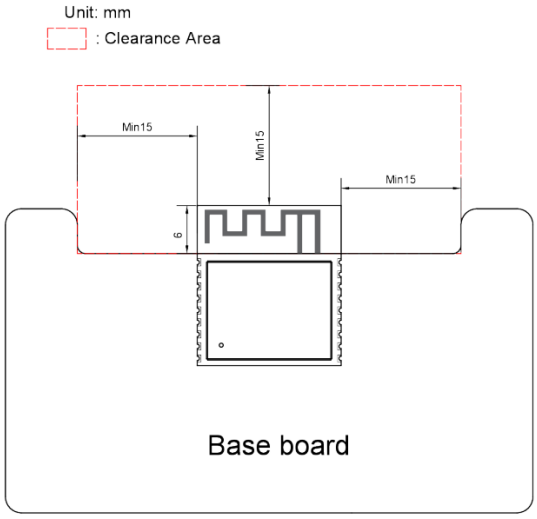


Figure 27 Recommandation de coupe du PCB pour l'antenne de l'ESP32

[44] Espressif Systems, « ESP32-C3 : Hardware Design Guidelines », [en ligne], (consulté le 26 août 2024), datasheet du 29 août 2024, figure 2, p. 20. URL : https://docs.espressif.com/projects/esp-hardware-design-guidelines/en/latest/esp32c3/esp-hardware-design-guidelines-en-master-esp32c3.pdf
[45] *Ibid.*, figure 4, p.21

6.4 Plan de masse

J'ai mis un plan de masse sur les deux côtés de mon PCB, ce qui réduit la résistance du chemin de retour et prévient les boucles de masse, minimisant ainsi les interférences.

Pour les connexions entre le plan de masse et les composants, j'ai laissé la règle par défaut dans Altium. De cette manière, les pads sont connectés avec un « relief connect », ce qui facilite le brasage manuel. Cette méthode permet de focaliser la dissipation thermique principalement sur le pad, plutôt que de la répartir sur tout le plan de masse. J'ai simplement modifié la condition pour les vias, en mettant une connexion directe pour minimiser la résistance.

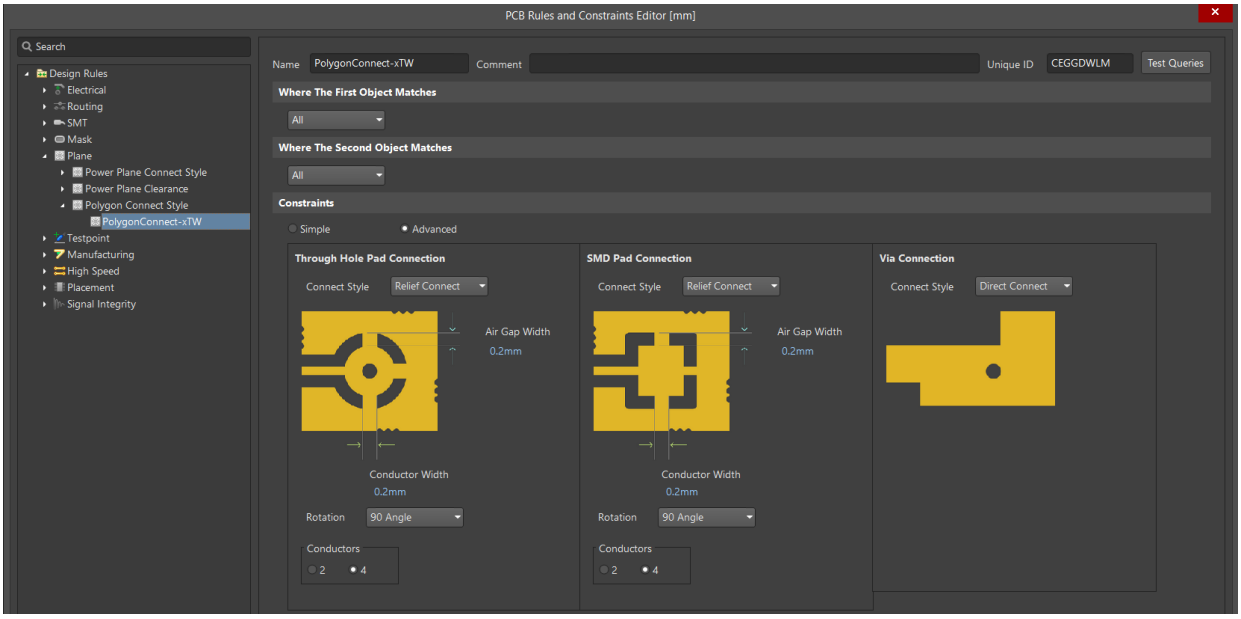


Figure 28 Règle de connexion pour les plans de masse

J'ai également effectué un « stitching » pour assurer une connexion optimale entre les plans de masse, garantissant un chemin de retour aussi court que possible et réduisant le risque de boucles de masse, ce qui diminue les interférences. Concernant la distance « grid » j'ai simplement laissé celle par défaut qui était de 8[mm], car le calcul précis pour les emplacements exact peut s'avérer compliqué. J'ai également rajouté quelques vias manuellement car le stitching m'avait mis seulement neuf vias, ce qui était insuffisant pour relier correctement les masses entrent-elles.

6.5 Attention particulière sur les composants

6.5.1 ESP32

Figure 29 Face arrière de l'ESP32

L'ESP32 dispose d'EPAD sous le composant. Dans le datasheet, il est indiqué que ceux-ci sont optionnels, mais peuvent aider à la dissipation de la chaleur de l'ESP32[46].

Dans mon cas, s'agissant d'un prototype, je ne souderai pas les pads dans cette version du projet.

[46] Espressif Systems. « ESP32-C3-WROOM-02, ESP32-C3-WROOM-02-U: Datasheet », *op. cit*. p.25

6.5.2 Quartz externe

Il est important qu'aucune piste ne passe sous le composant afin d'assurer le fonctionnement optimal de minimiser le risque d'interférences susceptibles de perturber d'autres signaux sur le circuit.

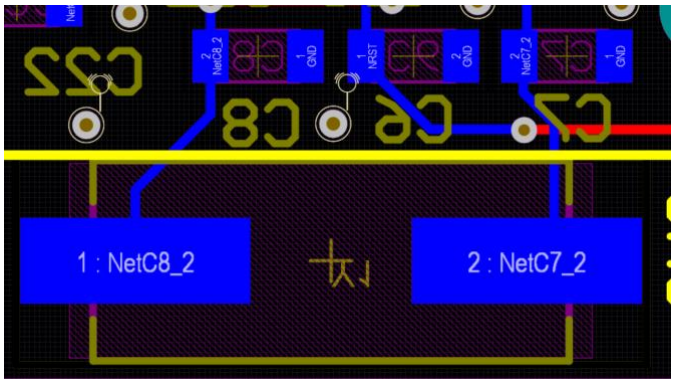


Figure 30 Vue du quartz du STM32

6.6 Largeur des pistes

Sachant que le courant maximal que mon circuit consommera sera d'environ 400[mA] (marges comprises) à cause du mode transmission de l'ESP32, j'ai regardé à l'aide du logiciel *PCB Toolkit version 8.23* si des pistes standard de 0.245[mm] étaient suffisantes.
Avant de commencer, j'ai modifié les paramètres dans *Tools -> Program Options -> IPC-2152 without modifiers*, car par défaut c'est le premier choix « *with modifiers* » qui est sélectionné. C'est également recommandé par le logiciel lui-même d'utiliser cette norme pour la plupart des applications[47].
Sur la figure ci-dessous on peut constater que le courant maximal que mes pistes pourront supporter est de 0.8952[A].

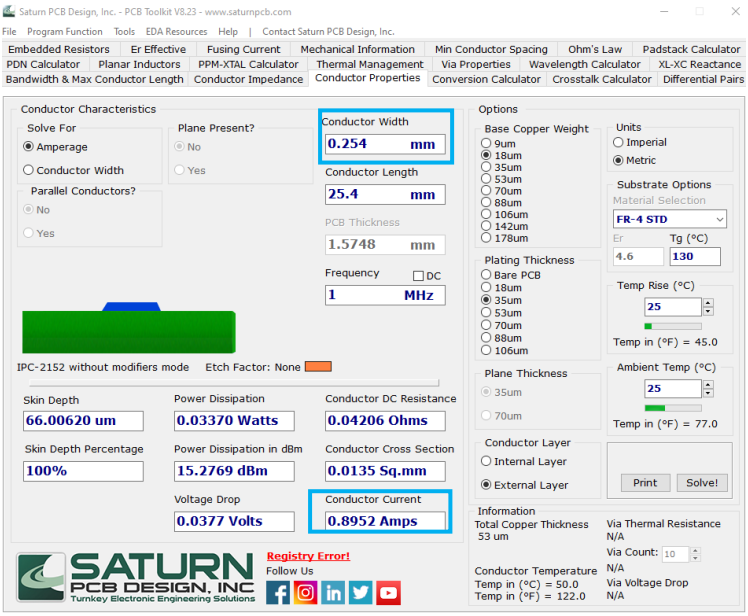


Figure 31 Dimensionnement de largeur de piste pour les signaux

Concernant les pistes pour le V_PILES (alimentation), j'ai opté pour une taille de 0.35[mm] et pour les pistes 3.3[V] une taille de 0.3[mm]. J'aurai pu mettre toutes les pistes à la même taille (0.254[mm]) mais je trouve cela plus pratique d'utiliser des tailles différentes pour mieux différentier les pistes.

Largeur de pistes [mm]	Courant max [A]
0.30	1.0100
0.35	1.1294

Tableau 6 Courant max en fonction de la largeur des pistes

[47] SATURN PCB DESIGN, INC. « Saturn PCB Toolkit Help », [en ligne], (consulté le 02 septembre 2024), Program Options (IPC Version). URL : https://saturnpcb.com/saturn-pcb-toolkit-help-htm/

6.6.1 DRC et Eurocircuit

Une fois mon PCB terminé et contrôlé, j'ai effectué un DRC tout en décochant la case « Board Outline Clearance », comme sur l'image suivante :

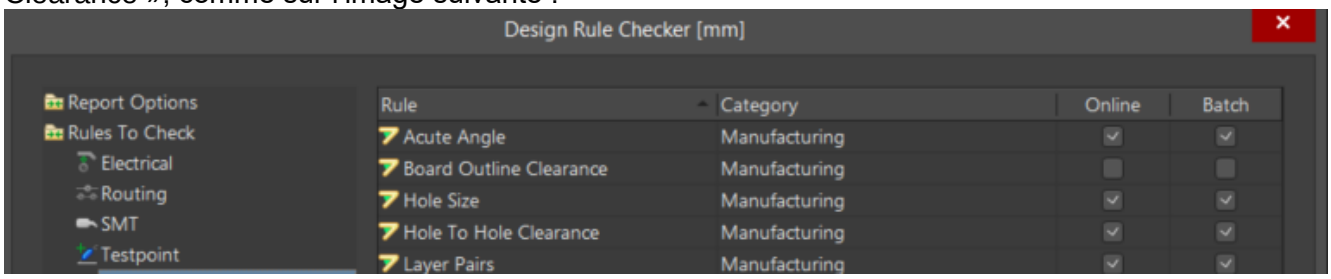


Figure 32 Sélection des règles concernant le DRC

J'ai pris cette décision car sans cela, j'avais une erreur à cause de l'antenne de l'ESP32 qui était hors du PCB. Sachant que ce n'était pas un vrai problème, l'antenne devant bel et bien être en dehors, j'ai simplement désactivé ce contrôle. Le dernier DRC ne m'indiquant aucune erreur, j'ai pu à partir de là effectuer une vérification sur Eurocircuits.

Eurocircuits, me garantit que mon PCB est classe 6C, qui est la préférence de l'ES en termes de spécification. J'ai ensuite pu commander mon PCB.

Les fichiers complets concernant la conception, ainsi que le draftsman, se trouveront en annexe F.

6.7 Boitier

6.7.1 Boitier PCB

Comme mentionné au point 3.6, p.12, j'ai opté pour un boitier fait sur mesure, ce qui me permettait de créer des encoches pour glisser mon PCB. De plus, je voulais que l'e-paper soit visible, donc mettre un plexiglass sur le dessus. Pour cela j'ai prévu des encoches de 3.3[mm] de chaque côté pour le PCB et de 2[mm] pour le plexiglass, j'ai à chaque fois rajouté des marges, car avec l'impression 3D, il peut y avoir des imprécisions.

Pour les dimensions du boitier en lui-même j'ai pris du 75 x 54 x 40[mm], car j'ai rajouté une épaisseur pour que le boitier soit moins cassant. J'ai également prévu une marge pour le PCB à l'intérieur en prenant en compte l'antenne (celle-ci faisant 6[mm][48]), de plus comme j'avais les encoches, j'ai élargi l'épaisseur sur les côtés. Il y a également un trou permettant de passer le câble de la communication I2C pour le capteur de température. Concernant la hauteur, j'ai procédé comme suit :

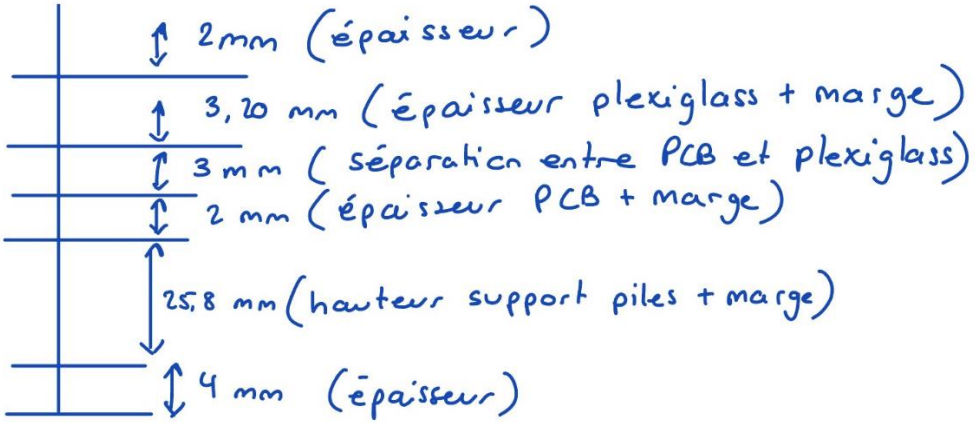


Figure 33 Estimation hauteur du boitier

J'ai choisi de fermer le boitier à l'aide de petits aimants, tout en mettant une sorte de poignée pour pouvoir ouvrir et fermer facilement le boitier.

[48] Espressif Systems, « ESP32-C3-WROOM-02, ESP32-C3-WROOM-02-U : Datasheet », *op. cit*. figure 10, p.27

Finalement, mon boitier devrait ressembler à ceci :

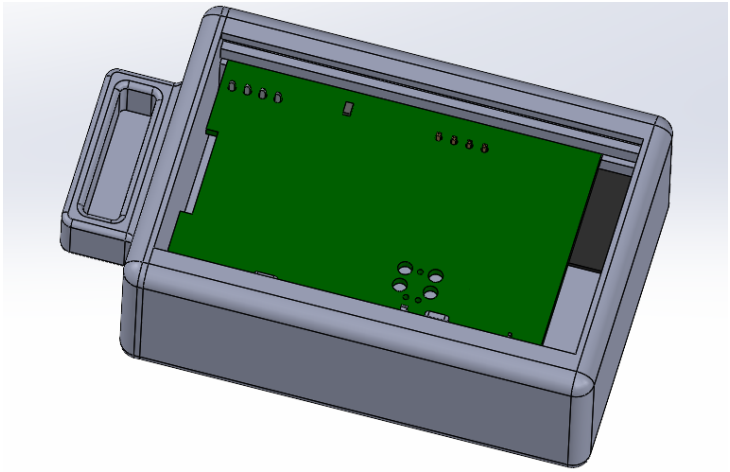


Figure 34 Vue 3D de l'assemblage du boitier avec PCB

ATTENTION : il faudra faire attention à ce que l'antenne de l'ESP32 soit à l'opposé des aimants pour éviter tout risque d'interférences.

Les mises en plan se trouvent en annexe J.1 et J.2.

6.7.2 Boitier pour le capteur SparkFun

Le client voulait un boitier pour le capteur de température, j'ai donc créé un boitier en prenant les dimensions du PCB qui était du 1x1[inches][49]. Sachant que 1[inch] correspond 25.4[mm], j'ai effectué la conversion en [mm] pour obtenir le résultat suivant : 25,4x25,4[mm].
J'ai ajouté à cela les 4[mm] d'épaisseur de chaque côté et un peu de marge à l'intérieur du boitier pour pouvoir facilement positionner mon PCB.

Finalement, j'ai un boitier avec les dimensions suivantes : 34.5 x 34.5 x 14[mm]. Etant donné que j'ai choisi de mettre des inserts filetés, j'ai pris une hauteur de 14[mm] pour laisser suffisamment de marge, tant pour les inserts que pour le PCB.

Etant donné que le diamètre des trous sur le PCB n'était pas indiqué, j'ai mesuré directement le diamètre sur le PCB que j'avais à disposition, et j'ai pu voir qu'il s'agissait de trous pour vis M3.

J'ai également fait des ouvertures sur chaque côté pour que le capteur de température et d'humidité ne soit pas isolé entièrement dans le boitier, et puisse donc mesurer les valeurs de températures et humidité du réfrigérateur, comme on peut le voir sur la figure ci-dessous :

Figure 35 Vue 3D du boitier fermé avec PCB

Les mises en plan se trouvent en annexe J.3 et J.4.

[49] SparkFun. « Dimensions in inches », [en ligne], (consulté le 29 août 2024). URL : https://cdn.sparkfun.com/assets/learn_tutorials/1/1/6/9/SHTC3_Dimensions.png

7 Coût total

Coût estimé [CHF]	Coût réel [CHF]
108.38	94.78

Tableau 7 Tableau de comparaison entre coût estimé et réel

Pour le coût estimé, j'ai repris celui du point 4, p.12.
Le coût réel contient : coût du PCB, le capteur de température SparkFun et le e-paper.

8 Montage du PCB

J'ai commencé par monter mon alimentation pour vérifier que le régulateur 3.3[V] fonctionnait correctement. Puis, j'ai vérifié la tension comme suit :

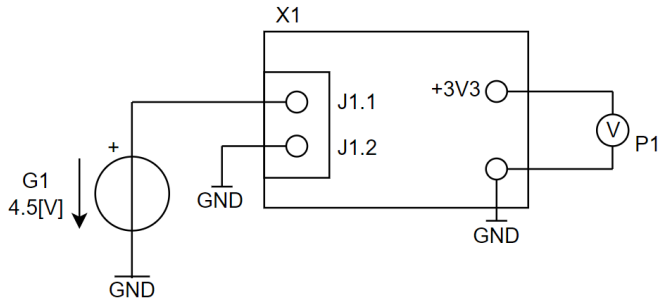


Figure 36 Schéma de mesure

Méthode de mesure :

1) Alimenter le circuit avec du 4.5[V] à l'aide de G1.
2) Brancher le voltmètre
3) Relever la mesure

Tension souhaitée [V]	Tension obtenue [V]	Validation
3.3	3.33	OK

Tableau 8 Tension relevée

Par la suite, j'ai monté tout le reste de mes composants.

9 Firmware

9.1 STM32

9.1.1 Configuration des pins sur le STM32

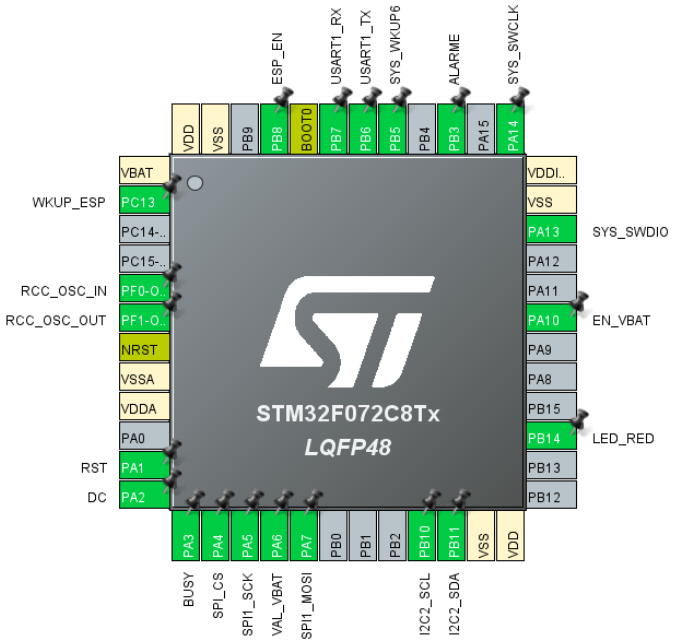


Figure 37 Configuration des pins

J'ai réalisé la configuration des pins en fonction du schéma confectionné au point 5.3.2, p.17

9.1.2 Configuration des GPIOs

Pin Name	Signal on Pin	GPIO output l...	GPIO mode	GPIO Pull-up...	Maximum out...	Fast Mode	User Label	Modified
PA1	n/a	Low	Output Push ...	No pull-up an...	Low	n/a	RST	☑
PA2	n/a	Low	Output Push ...	No pull-up an...	Low	n/a	DC	☑
PA3	n/a	n/a	Input mode	No pull-up an...	n/a	n/a	BUSY	☑
PA4	n/a	High	Output Push ...	No pull-up an...	Low	n/a	SPI_CS	☑
PA10	n/a	Low	Output Push ...	No pull-up an...	Low	n/a	EN_VBAT	☑
PB3	n/a	Low	Output Push ...	No pull-up an...	Low	n/a	ALARME	☑
PB8	n/a	High	Output Push ...	No pull-up an...	Low	Disable	ESP_EN	☑
PB14	n/a	Low	Output Push ...	No pull-up an...	Low	n/a	LED_RED	☑
PC13	n/a	Low	Output Push ...	No pull-up an...	Low	n/a	WKUP_ESP	☑

Figure 38 Configuration GPIOs

Il est important de configurer la pin « ESP_EN » (PB8) à l'état haut. Sinon, lors de la programmation du STM32, l'ESP32 passera en mode réinitialisation.

9.1.3 Configuration du Timer 6

Le timer6 sera utilisé pour le contrôle de l'état de la pile, qui servira pour définir le délai entre l'activation du transistor et la mesure. J'ai pris un temps de 10[ms].

Procédure de calcul pour le timer6 :

$$T_{périodesignal} = \frac{1}{f_{QUARTZ}} = \frac{1}{8 \cdot 10^6} = 125[ns]$$

Ensuite j'ai calculé la valeur jusqu'à laquelle le compteur doit aller pour avoir une interruption toutes les 10[ms] de la manière suivante :

$$N_{compteur} = \frac{T_{voulu}}{T_{périodesignal}} = \frac{10 \cdot 10^{-3}}{125 \cdot 10^{-9}} = 80'000$$

Je constate que la valeur obtenue est supérieure à celle que le microcontrôleur peut compter. En effet, la valeur maximale de celui-ci est de 65'536 (2^{16})[50]. Pour cette raison, j'utiliserai un prescaler qui me permettra de diviser la fréquence du microcontrôleur.

$$Prescaler = \frac{N_{compteur}}{2^{16}} = \frac{80'000}{65536} = 1.22 => 2$$

J'ai donc utilisé un prescaler de 2 et recalculé le nombre de comptages avec celui-ci:

$$N_{compteur} = \frac{N_{compteur}}{Prescaler} - 1 = \frac{80'000}{2} - 1 = 39'999$$

J'ai soustrait un, car la valeur « 0 » est prise en compte.

Finalement, j'ai également activé l'interruption du timer dans « NVIC Setting » en cochant la case « TIM6 Global Interrupt ».

Le Timer6 sera démarré juste après avoir activé le transistor, et sera arrêté après avoir effectué la mesure pour déterminer l'état de la pile.

[50] ST, « STM32F072x8 STM32F072xB », [en ligne], (consulté le 20 août 2024), datasheet de septembre 2019, révision 6, table7 (TIM6), p.21. URL : https://www.st.com/content/ccc/resource/technical/document/datasheet/cd/46/43/83/22/d3/40/c8/DM00090510.pdf/files/DM00090510.pdf/jcr:content/translations/en.DM00090510.pdf

9.1.4 Configuration de l'ADC

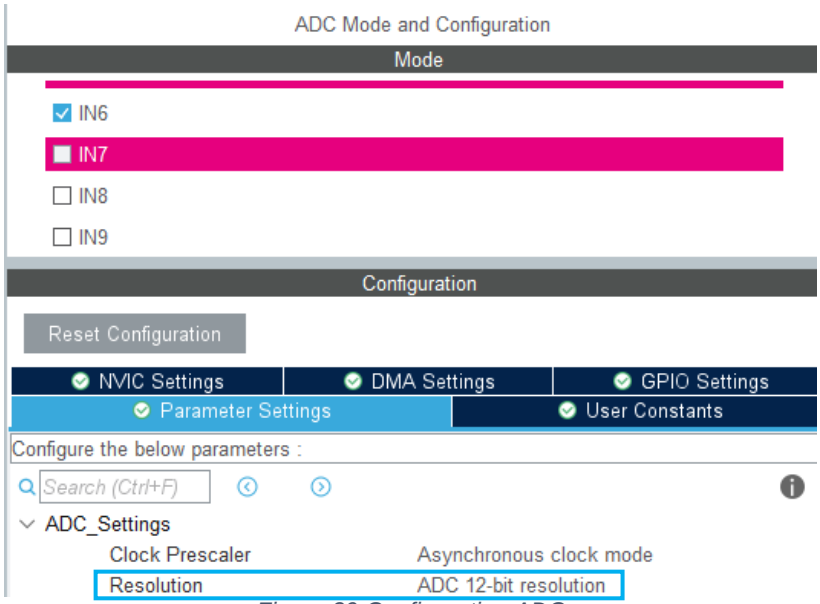


Figure 39 Configuration ADC

J'ai modifié la résolution en 12 bits afin d'avoir une meilleure précision pour les valeurs mesurées.

Le reste de la configuration reste par défaut.

Remarque : lors de l'initialisation, il est important de démarrer la calibration de l'ADC.

9.1.5 Configuration UART

J'ai repris la configuration indiquée sur le site d'Espressif[51] :

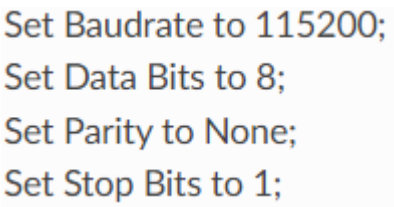


Figure 40 Configuration de l'UART1 (Espressif)

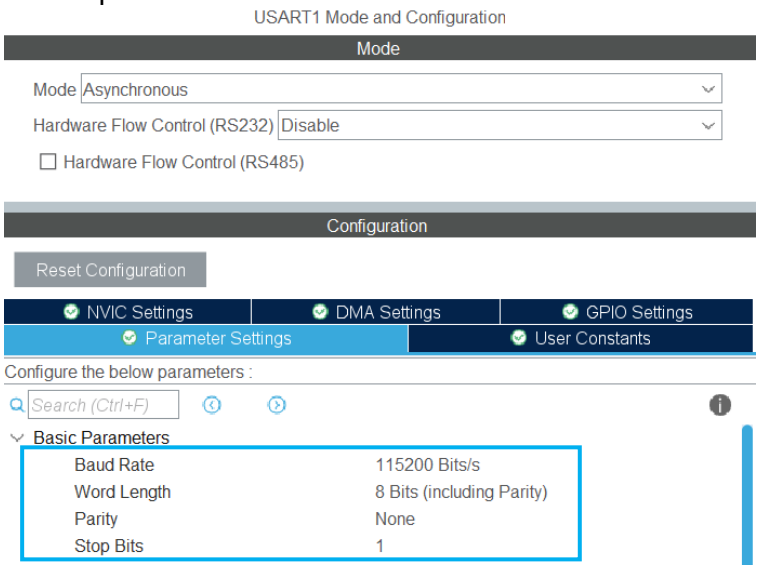


Figure 41 Configuration de l'UART1 (STM32)

[51] Espressif. « ESP32-C3 : Downloading Guide », [en ligne], (consulté le 9 septembre 2024), master(latest), Check Whether AT Works. URL : https://docs.espressif.com/projects/esp-at/en/latest/esp32c3/Get_Started/Downloading_guide.html

9.1.6 Configuration SPI

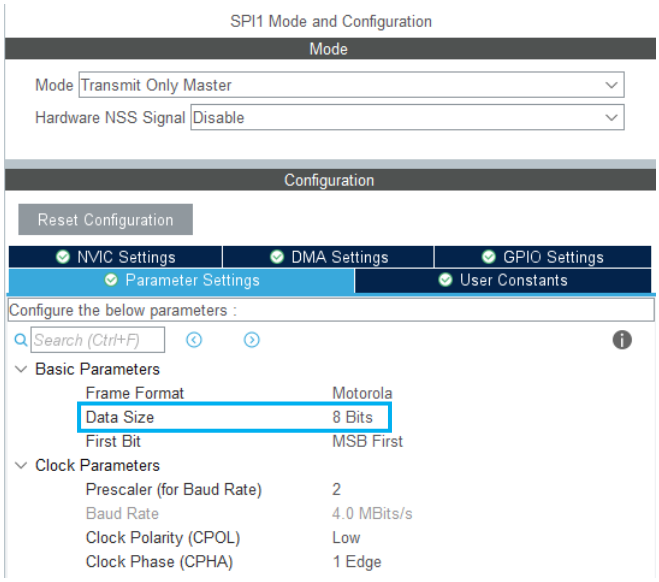


Figure 42 Configuration SPI

J'ai mis le SPI en mode « Transmit Only Master » puisque le e-paper ne renvoie pas d'informations.

J'ai laissé les autres paramètres par défaut, sauf « Data Size » à 8[Bits] comme indiqué dans le datasheet[52].

Toutefois, j'ai quand même vérifié la fréquence maximum supportée par l'e-paper, qui est de 20[MHz] pour le mode écriture[53]. Le baudrate de 4[Mbits/s] est loin de la limite maximum, cela me permet donc d'envoyer des données plus rapidement, tout en étant en-dessous de la limite.

9.1.7 Configuration I2C

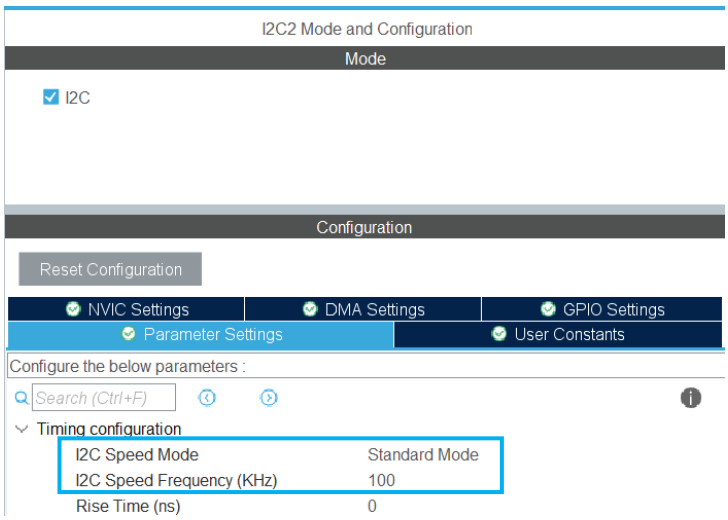


Figure 43 Configuration I2C

Après le calcul réalisé au point 5.3.4, p.19, j'ai décidé de prendre le mode standard pour l'I2C.

De plus, ayant un câble allant dans un réfrigérateur, les lignes seront moins sensibles aux interférences électromagnétiques et j'aurai moins de risque de perdre des données.

[52] Waveshare. « 2.13inch e-Paper », [en ligne], consulté le 9 août 2024, version 4, point 5, note 5-5, p.8. URL : https://files.waveshare.com/upload/4/4e/2.13inch_e-Paper_V4_Specification.pdf
[53] *Ibid.,* point 6.3.4, write mode (f_{scl}), p.12

9.1.8 Configuration quartz externe

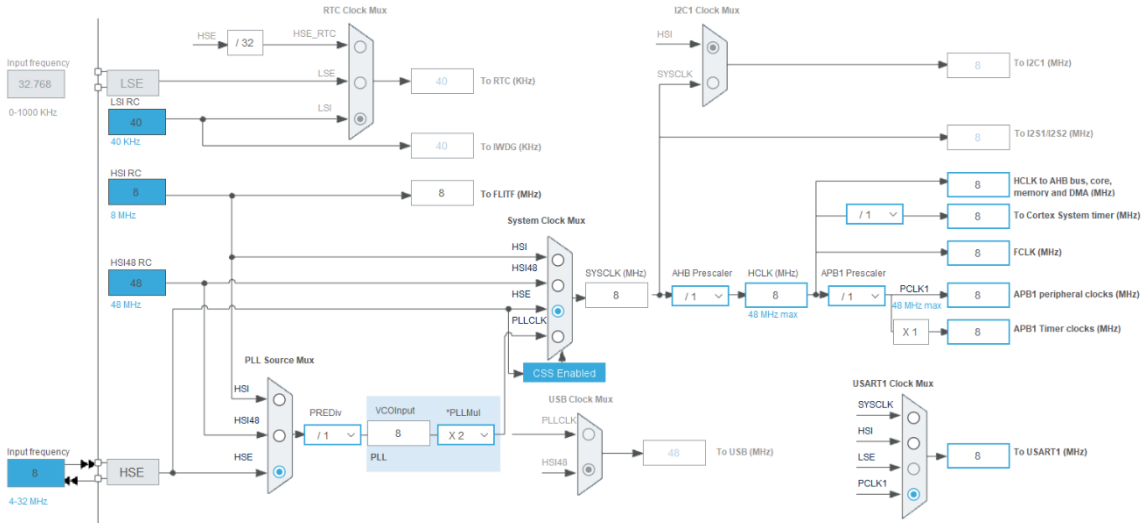


Figure 44 Configuration du quartz

Le cadencement de mon STM32 se fait via le quartz externe. J'ai donc modifié le « Clock Configuration » sur STM32CubeMx pour qu'il prenne en compte celui-ci au lieu de l'interne. J'ai également activé l'option « Enabled CSS » pour permettre au système de basculer sur l'oscillateur interne au cas où l'externe rencontrerait un problème.

9.1.9 Configuration heap size et stack size

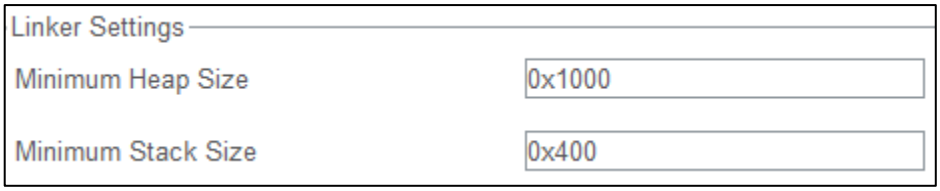


Figure 45 Modification du heap size

J'ai dû augmenter la taille de la heap size (mémoire dynamique) à cause de l'utilisation de malloc() dans le code d'affichage du e-paper. En effet, le code alloue une zone mémoire supplémentaire pendant l'exécution du programme dont la taille est liée à la résolution de l'écran e-paper. Il s'agit d'un tableau qui permet de déterminer la couleur (noir ou blanc) de chaque pixel de l'écran. Sans cette augmentation de la heap size, l'écran ne fonctionnait pas car il n'y avait tout simplement pas assez de mémoire dynamique disponible.

Attention, le STM32 étant bien évidemment limité en termes d'espace mémoire, cela doit être fait avec précaution. Augmenter le heap size de manière trop importante peut être source d'instabilité et de bugs. J'ai donc augmenté la heap size à la valeur tout juste suffisante pour permettre l'exécution de mon programme. Il faut également bien veiller à libérer la mémoire allouée dynamiquement pour éviter les fuites mémoires. Il est donc important de bien gérer les allocations dynamiques afin de garantir la bonne stabilité du programme.

9.2 Fonctionnement globale du système

9.2.1 Fonctions dans le fichier main.c

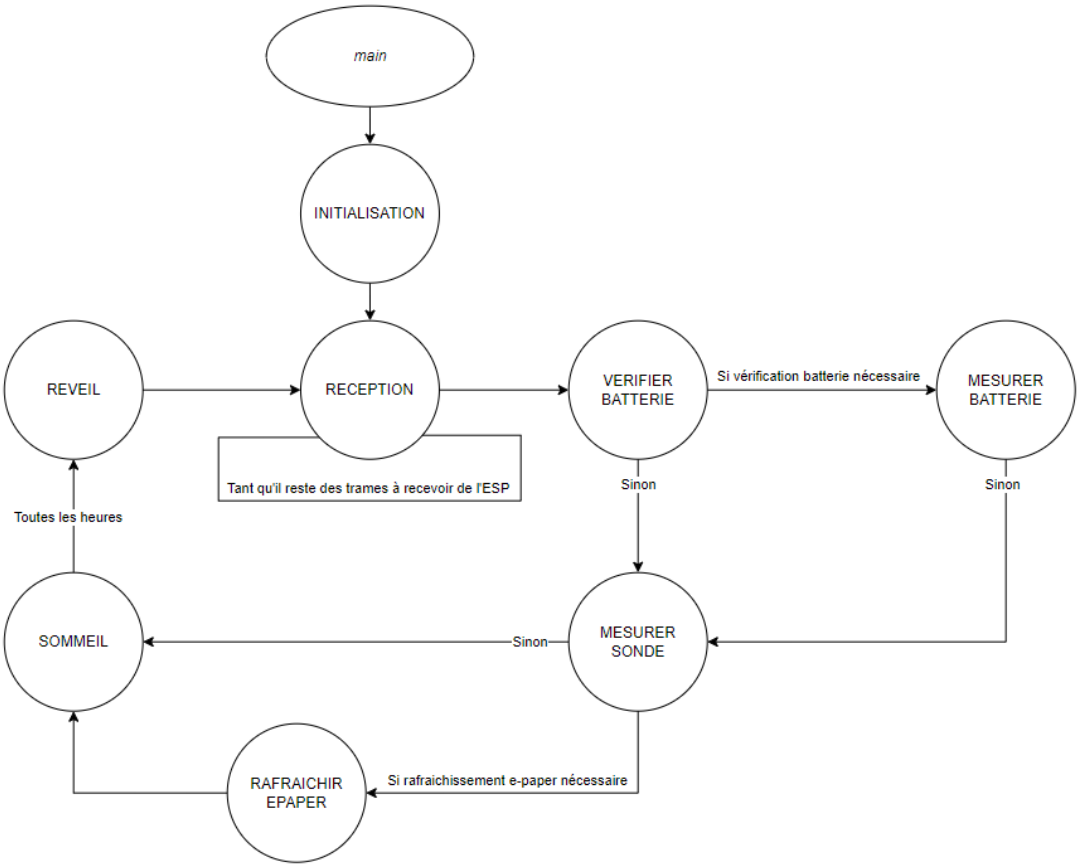


Figure 46 Machine d'état principale

Initialisation : initialisation des variables, calibration ADC, initialisation de la gestion du mode sommeil
Réception : réceptionne les trames UART que l'ESP envoie
Vérification batterie : vérifie si une nouvelle mesure de l'état de la batterie est nécessaire
Mesurer batterie : mesure de la tension d'alimentation (tous les quatre heures)
Mesurer sonde : récupère la température actuelle et regarde s'il faut rafraichir l'e-paper ou non
Rafraichir e-paper : rafraichissement de l'écran e-paper
Sommeil : met le STM32 en mode sommeil
Réveil : toutes les heures (réveillé par l'ESP)

10 ESP32-C3

10.1 Schéma branchement câble FTDI

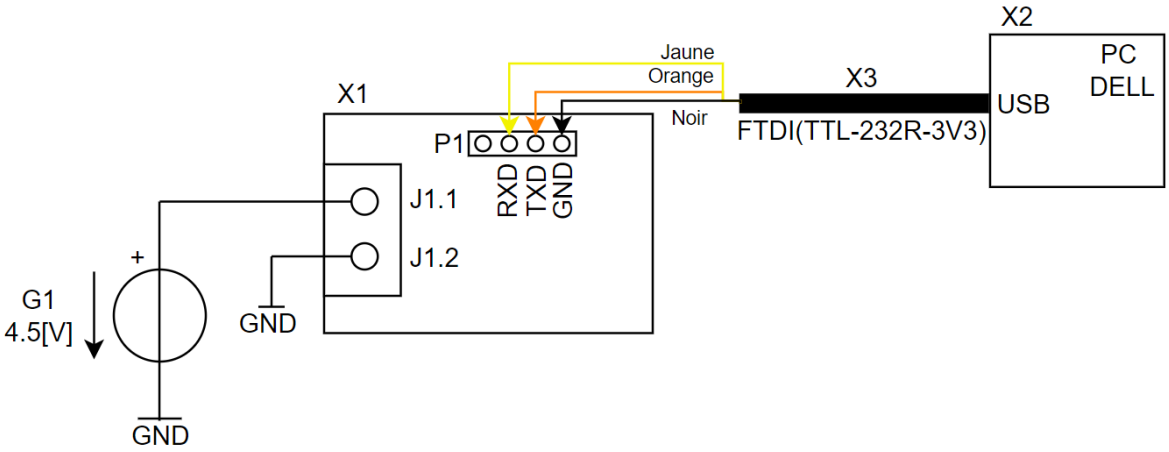


Figure 47 Schéma pour programmer l'ESP32

10.2 Etapes pour la programmation

1. Installer ArduinoIDE et suivre les instructions jusqu'au point 10 du mode d'emploi se trouvant en annexe O
2. Lancer le fichier nommé « 2409_ESP _V1.ino » se trouvant à l'endroit suivant : *K:\ES\PROJETS\SLO\2409_MesureTH_RefrigerateurCongelateur\soft\Firmware\ESP32\2409_ESP_V1*
3. Effectuer la programmation comme indiqué au point 10 dans le mode d'emploi (annexe O)

10.3 Fonctionnement global du système

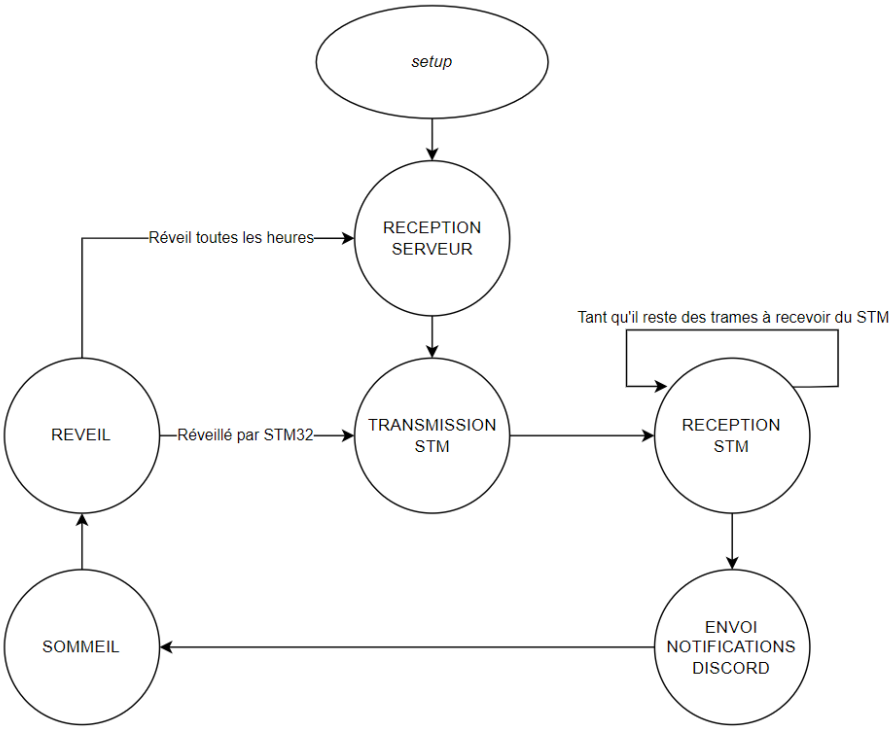


Figure 48 Fonctionnement global

Figure 49 Fonction setup

10.3.1 Remarque supplémentaire

J'avais initialement flashé l'ESP32 pour utiliser les commandes AT, ce qui m'aurait permis de communiquer via UART en envoyant des commandes textuelles, sans avoir à programmer l'ESP32. Je pensais les intégrer dans le projet, mais finalement, je ne les ai pas utilisées car j'ai décidé de programmer l'ESP via Arduino IDE pour qu'il puisse comparer les valeurs avec celles sur serveur, afin que l'ESP reste allumé le moins longtemps possible. L'annexe R, montre les étapes réalisées pour le flash.

11 Création de la page HTML

1. Ouvrir le logiciel Notepad++
2. Sélectionner le langage HTML dans « langage -> H -> HTML »
3. Sauvegarder le fichier en mettant l'extension .html

Ensuite j'ai suivi le tutoriel pour créer une site web sur le site *openclassrooms*[54].

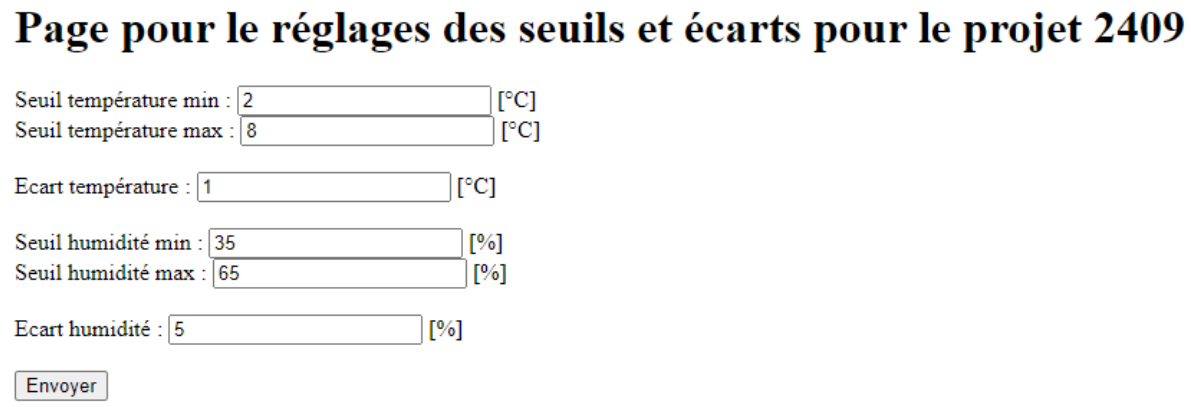


Figure 50 Résultat final du site internet

Le bouton « Envoyer » servira à stocker les valeurs sur le serveur pour plus tard

12 Serveur

12.1 Principe de fonctionnement

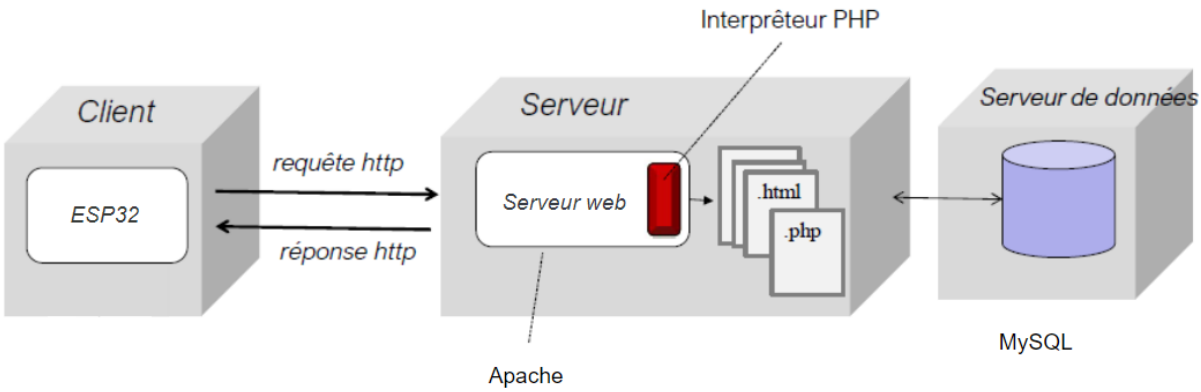


Figure 51 Fonctionnement globale du système[55]

[54] OpenClassrooms. « Créez votre site web avec HTML5 et CSS3 », [en ligne], (consulté le 1er septembre 2024), mis à jour le 18/06/2024. URL : https://openclassrooms.com/fr/courses/1603881-creez-votre-site-web-avec-html5-et-css3

[55] ETML-ES. « Programmation web : 00 Introduction », slide 6

L'ESP32 enverra des requêtes afin de récupérer les valeurs du serveur et constater si celles-ci ont été modifiées. Il transmettra également au serveur les changements d'état concernant la pile ou les dépassement des seuils de température et humidité.

12.2 Création serveur

1. Mise en place serveur Apache dans XAMPP
2. Mise en place MySQL dans XAMPP
3. Configuration du routeur Cisco utilisé

Pour la mise en place de tout le système, voir le mode d'emploi en annexe O.

Le serveur « Apache » servira à transformer mon ordinateur en point d'accès pour héberger mon site internet, tandis que « MySQL » servira à stocker les dernières données entrées sur la page internet.

L'utilisation du langage « PHP » étant nécessaire pour communiquer avec la base de données, j'ai dû renommer mon fichier .html (mentionné au point 11) en .php afin de pouvoir y inclure du code PHP.

12.3 Création base de données sur phpMyAdmin

Pour accéder à la base de données, il faut démarrer le serveur « Apache », puis « MySQL ». Ensuite cliquer sur « Admin » sur la même ligne que « MySQL ».

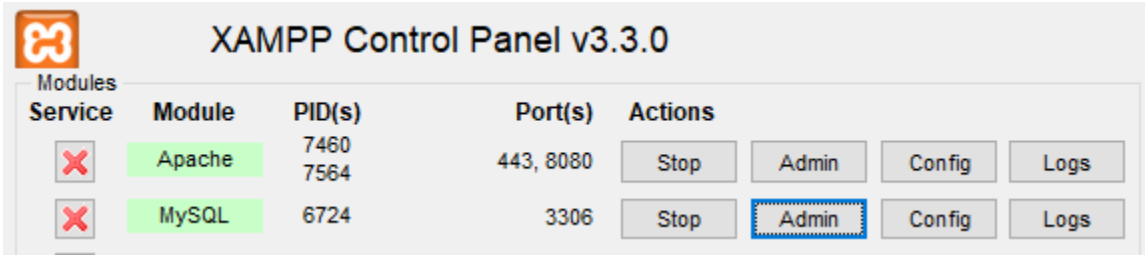


Figure 52 Interface XAMPP avec Apache et MySQL activés

Ensuite j'ai procédé comme cela :

1. Création de la base de données
    a. Cliquer sur le bouton « Nouvelle base de données » dans le menu latéral gauche
    b. Indiquer le nom de la base de données
    c. Cliquer sur le bouton Créer
2. Création d'une nouvelle table
    a. Sélectionner la base de données créée dans le menu latéral gauche
    b. Cliquer sur le bouton « Nouvelle table » dans le menu latéral gauche
    c. Indiquer le nom et type de chaque valeur qui seront contenues dans la base de données
    d. Cliquer sur le bouton « Enregistrer »

A la suite à cela, j'ai suivi plusieurs tutoriels qui m'ont permis de réaliser les fichiers api.php et form.php et de mettre en place tout le système. Tous les liens se trouvent dans la webographie.

12.4 Option supplémentaire : alarme sur serveur Discord

Pour configurer cela, j'ai créé un webHook de la façon suivante :

1. Se connecter à son compte Discord
2. Aller sur ajouter un serveur
3. Créer le mien -> pour mes amis et moi
4. Choisir un nom de serveur pour ce projet : Projet 2409_MesureTH
5. Puis créer
6. Créer un salon
7. Nommer le salon comme voulu, dans ce cas : notification-alarmes
8. Créer un salon
9. Modifier le salon
10. Intégrations -> Créer un webhook -> nouveau webhook
11. Nom : ESP_ALARME salon : #notification-alarmes
12. Copier le lien du webhook

J'ai créé cette option en partant de la réflexion suivante : que se passe-t-il si une alarme est détectée en pleine journée, mais que l'on ne regarde l'écran e-paper ou le site qu'en rentrant chez soi le soir venu ? Si la température et l'humidité sont depuis revenus entre les seuils, il serait impossible de savoir qu'une alarme a eu lieu pendant notre absence, ni de savoir pendant combien de temps les seuils n'ont pas été respectés. Avec l'envoi de notification Discord, cela permet d'avoir un historique des changement d'états, pour savoir précisément à quel moment les alarmes ont eu lieu et combien de temps elles ont duré. De plus, Discord est également une plateforme utilisée par mon client, ce qui rend la solution d'autant plus pratique.

13 Tests

13.1 I2C

13.1.1 Schéma de mesure

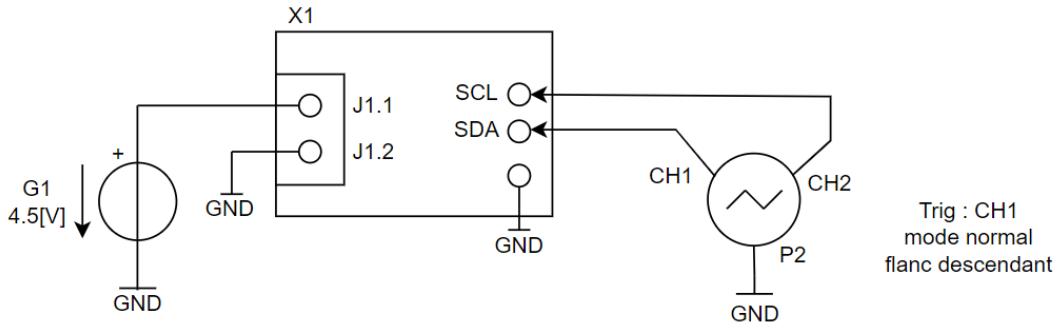


Figure 53 Schéma de mesure I2C

Méthode de mesure :

1) Alimenter le circuit avec du 4.5[V] à l'aide de G1
2) Brancher la sonde CH1 sur « SDA » et CH2 sur « SCL »
3) Appuyer sur le bouton « Protocol », changer le type bus en I2C, puis dans configuration mettre les seuils à 1.65[V]
4) Toujours dans la configuration, mettre C2 sur « SCL » et C1 sur « SDA »
5) Relever l'oscillogramme

13.1.2 Mesures

Le code que j'ai utilisé provient d'une librairie trouvée sur GitHub (voir annexe M). J'ai vérifié que les trames correspondaient correctement à ce qui est indiqué sur la figure 7[56] du datasheet, montrant que la séquence de communication devrait se dérouler comme suit :

1. Envoi de l'adresse I2C avec la commande d'écriture pour accéder au registre « Read ID register [57]» du capteur
2. Lecture de l'adresse I2C (accusé de réception) pour confirmer qu'il a reconnu l'adresse et est prêt à recevoir des commandes
3. Envoi des commandes de mesure MSB et LSB pour lancer la mesure d'humidité et température
4. Lecture des données d'humidité
5. Lecture des données de température

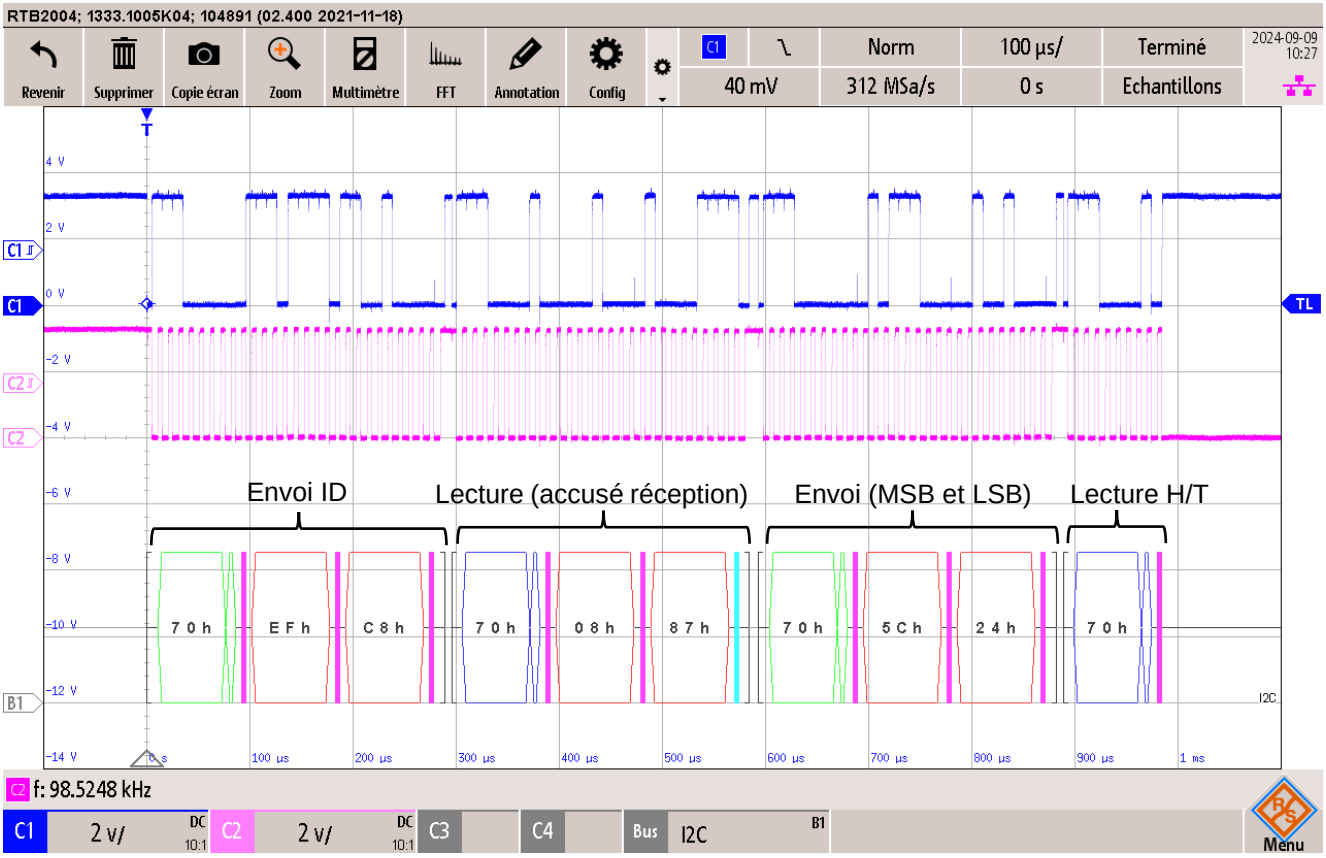


Figure 54 Début trame I2C (envoi, lecture et commande MSB et LSB)

La première trame décodée est cohérente avec ce qui est attendu d'après la figure 7 du datasheet.

[56] SENSIRION. « Datasheet SHTC3 : Humidity and Temperature Sensor IC », [en ligne], (consulté le 29 août 2024), version 2 – juin 2019, figure 7, p.7. URL : https://cdn.sparkfun.com/assets/1/1/f/3/b/Sensirion_Humidity_Sensors_SHTC3_Datasheet.pdf
[57] *Ibid*., point 5.9, p.8

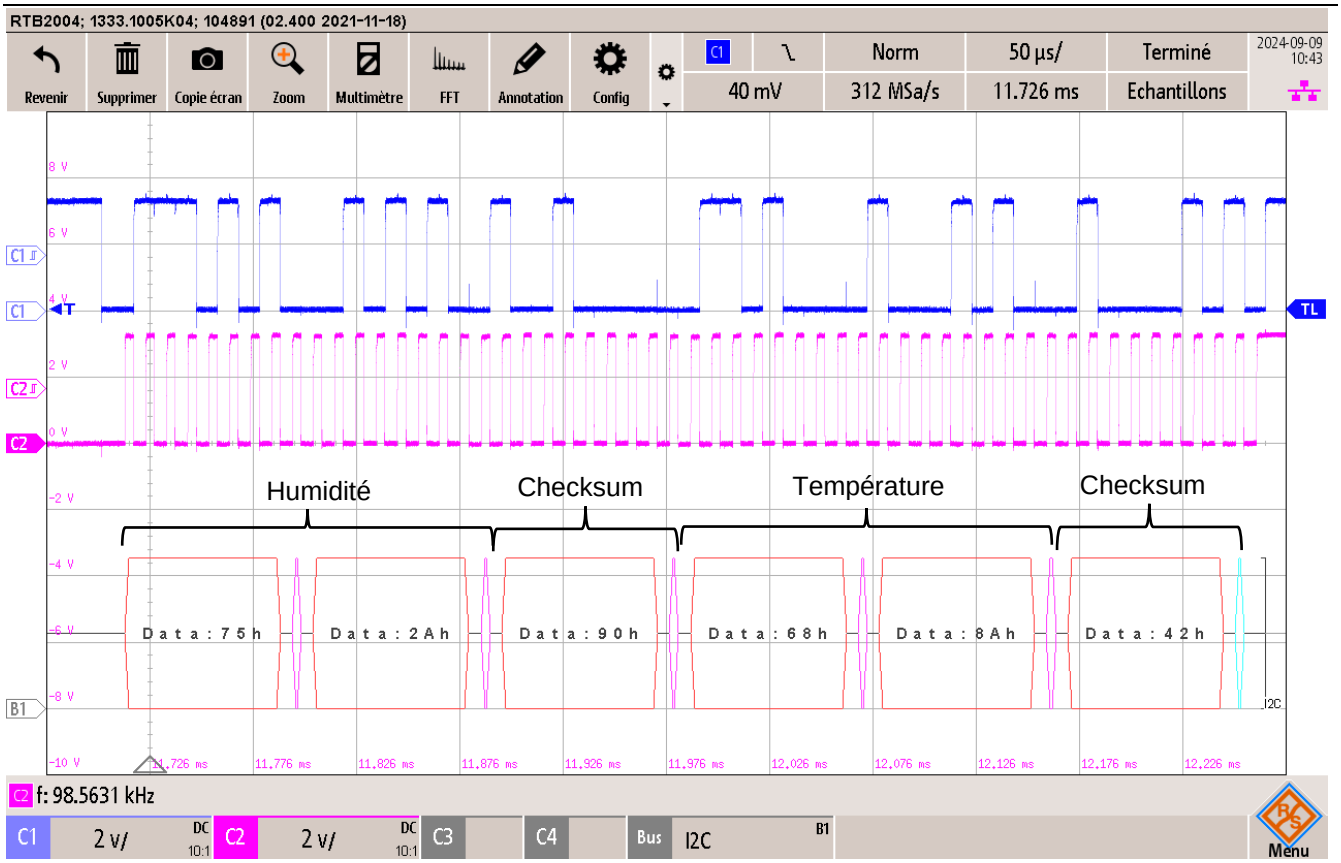


Figure 55 Fin trame I2C (humidité et température)

En utilisant les formules pour le calcul de l'humidité et de la température :

Relative humidity conversion formula (result in %RH):

$$RH = 100 \cdot \frac{S_{RH}}{2^{16}}$$

Temperature conversion formula (result in °C):

$$T = -45 + 175 \cdot \frac{S_T}{2^{16}}$$

Figure 56 Formule pour obtention de l'humidité et température[58]

J'ai comparé les valeurs obtenues avec celles affichées en mode debug pour vérifier qu'elles étaient correctement reçues.

	Valeurs affichées mode debug	Valeurs obtenues depuis la trame
Humidité	45.76	45.76
Température	26.46	26.46

Tableau 9 Valeurs relevées

Les valeurs sont donc correctes.
Afin de vérifier que les sommes de contrôle (checksums) soient correctes, j'ai effectué les étapes de décodage détaillées en annexe P.

Remarque : j'ai décidé de ne pas activer le mode sommeil de la sonde afin que le capteur maintienne des mesures stables lors de la récupération des valeurs par le STM32. Après un réveil, il pourrait prendre entre 5 et 30[s] pour ajuster la température et environ 8[s] pour l'humidité, afin d'atteindre 63[%] [59]de leurs valeurs, ce qui n'est pas souhaité dans mon cas.

[58] *Ibid*. point 5.11, p.9
[59] Ibid., Table 1-2, p.2

13.2 SPI

13.2.1 Schéma de mesure

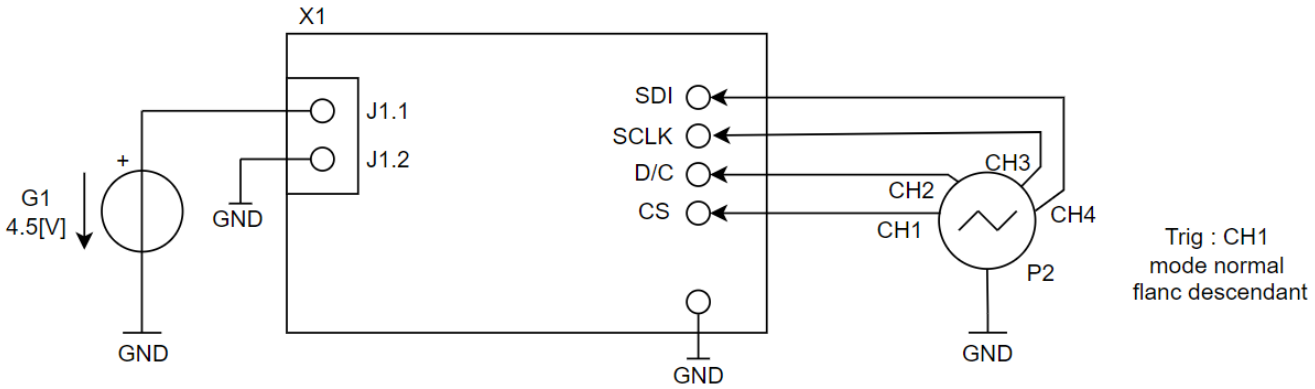


Figure 57 Schéma de mesure SPI

Méthode de mesure :

1) Alimenter le circuit avec du 4.5[V] à l'aide de G1
2) Brancher la sonde CH1 sur « CS », CH2 sur « D/C », CH3 sur « SCLK » et CH4 sur « SDI »
3) Appuyer sur le bouton « Protocol », changer le type bus en SPI (with CS), puis dans configuration mettre les seuils à 1.65[V]
4) Mettre C1 sur « Sélection Chip », C3 sur « Clock » et C4 sur « MOSI »
5) Relever l'oscillogramme

13.2.2 Mesure premier test (e-paper non fonctionnel)

Comme indiqué au point 3.2, p.8 j'ai utilisé la librairie provenant de GitHub, cependant lors de la première implémentation du code, j'ai obtenu un écran noir bruité comme présenté sur la figure ci-dessous :

Figure 58 Ecran e-paper bruité

Cependant les trames SPI étaient cohérentes avec ce qui était attendu dans le datasheet[60].
Les trames se trouvent en annexe L.2.

13.2.3 Mesures deuxième test (e-paper fonctionnel)

Le problème précédent venait probablement du fait que j'avais recopié tout le code depuis GitHub, ce qui pouvait créer des conflits entre les fichiers. J'ai donc décidé de tout recommencer depuis zéro, en ajoutant les fichiers nécessaires un par un, et en effectuant une compilation après chaque ajout pour vérifier s'il y avait des erreurs. Bien que longue et fastidieuse, cette procédure s'est révélée payante puisque cela a permis de faire fonctionner l'écran e-paper.
La mesure étant trop longue pour être complétement décodée avec mon oscilloscope, il m'est impossible de vérifier l'intégralité des informations. Cependant, j'ai pu confirmer que la trame

[60] WAVESHARE. « 2.13inch e-Paper », version 4, *op. cit*. figure 6-2, p11

d'initialisation correspondait bien aux valeurs du code de la librairie. Pour démontrer que la communication se fait correctement, j'ai temporairement réduit le baudrate à 500[kBits/s] dans STM32CubeMx. A ma vitesse de base, la trame était trop longue, ce qui aurait nécessité trop de captures.

D'après le datasheet, la trame reçue doit correspondre à celle-ci :

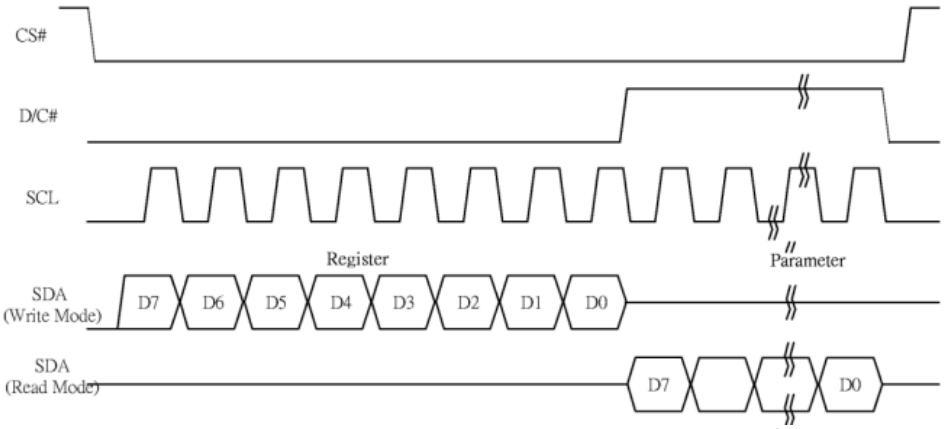


Figure 59 Format de trame souhaité d'après le datasheet[61]

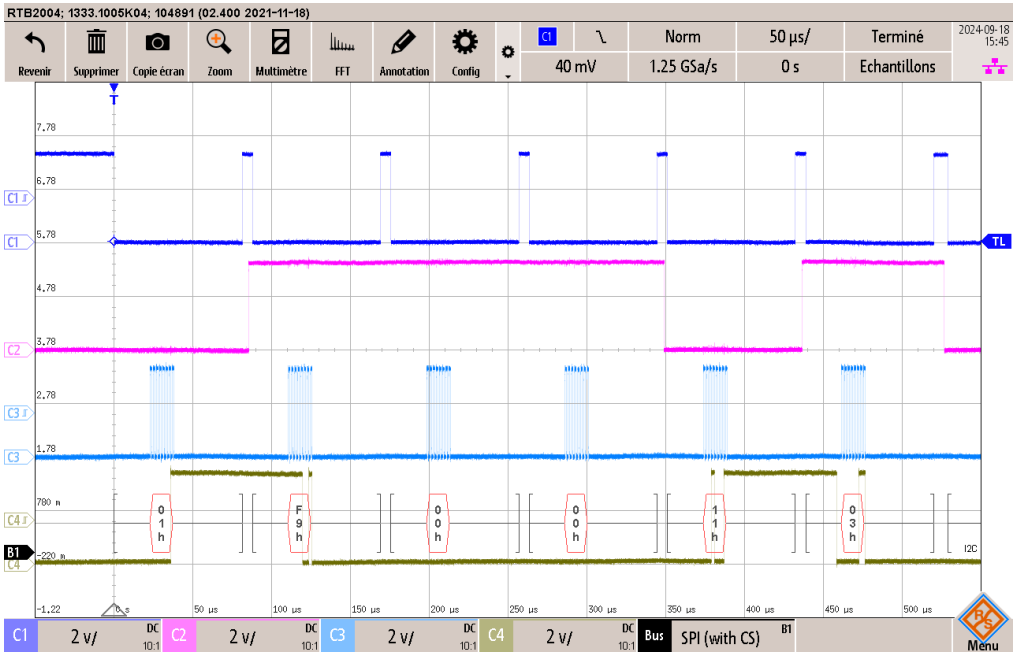


Figure 60 Début initialisation avec le baudrate à 500[kBits/s]

Le format de trame correspond bien à celui du datasheet. De plus, j'ai comparé les valeurs avec celles indiquées dans la fonction d'initialisation du code disponible sur Github[62].

Le reste des captures se trouvent en annexe L.2.

[61] *Ibid.*
[62] Waveshare, « e-Paper/EPD_2in13_V4.c », GitHub, [en ligne], (consulté le 21 septembre 2024), lignes 160 à 189. URL : https://github.com/waveshareteam/e-Paper/blob/master/STM32/STM32-F103ZET6/User/e-Paper/EPD_2in13_V4.c

13.3 UART

13.3.1 Schéma de mesure

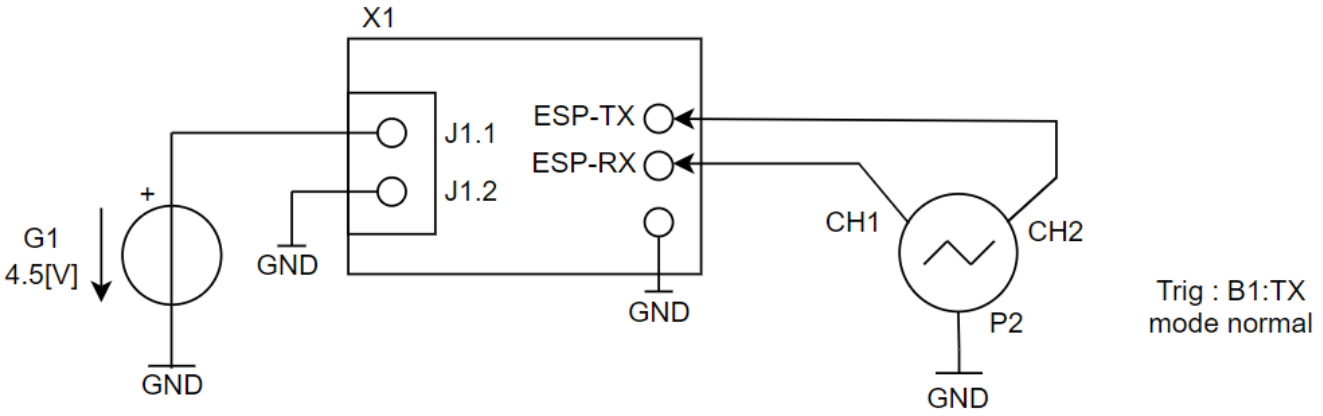


Figure 61 Schéma de mesure UART

Méthode de mesure :

1) Alimenter le circuit avec du 4.5[V] à l'aide de G1
2) Brancher la sonde CH1 sur « ESP-RX » et CH2 sur « ESP-TX »
3) Appuyer sur le bouton « Protocol », changer le type bus en UART, puis dans configuration mettre les seuils à 1.65[V]
4) Mettre C1 sur « RX » et C2 sur « TX »
5) Mettre le trigger sur B1 : UART, TX ou B1 : UART, RX selon la trame souhaitée
6) Relever les oscillogrammes

13.3.2 Mesures

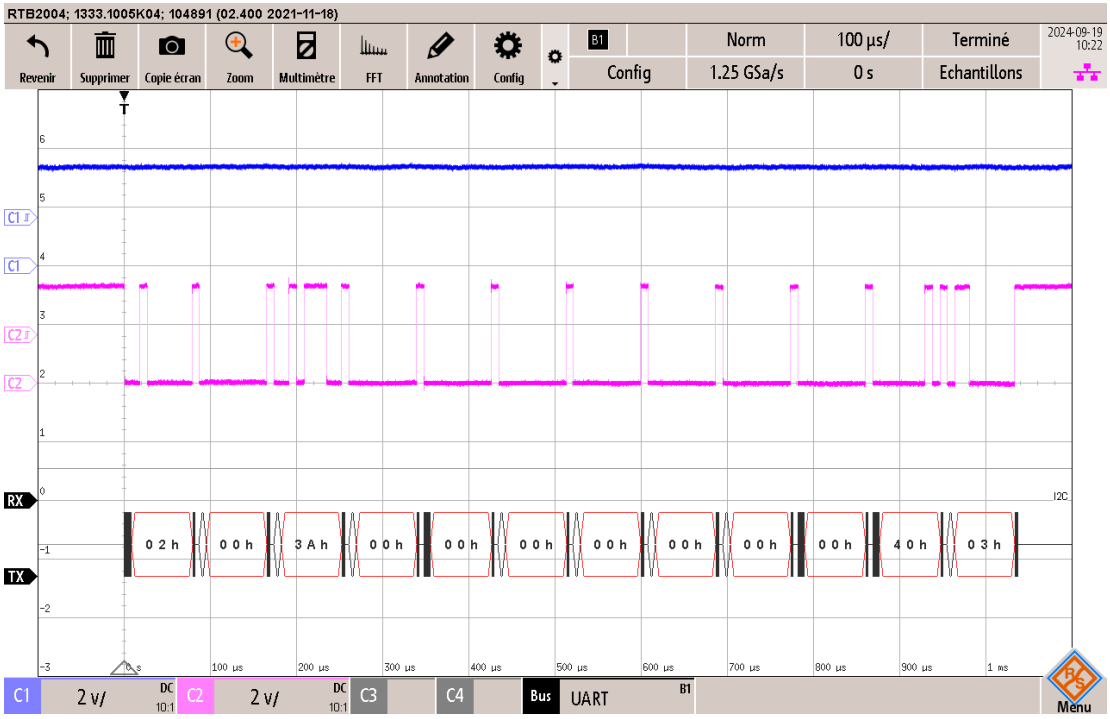


Figure 62 Envoie trame du serveur au STM32, modification index 0 à 5[°C]

Pour décoder plus rapidement ma trame, j'ai utilisé le format suivant :

« STX index : valeur ETX »

- STX (Start of Text) est un caractère utilisé pour identifier les débuts de trames
- index est un entier pour identifier le type de valeur communiquée
- : est un caractère pour séparer et faciliter la lecture des trames pendant les phases de tests
- valeur est un double correspondant à la valeur communiquée
- ETX (End of Text) est un caractère utilisé pour identifier les fins de trames

Concrètement, ce format comprend : 1 octet pour le début de la trame (STX)[63], 1 octet pour l'identifiant de la valeur, 1 octet pour les deux points, 8 octets pour la valeur communiquée, puis 1 octet pour la fin de trame (ETX). Ce qui signifie que chaque trame correspond à 12 octets.

Sur la mesure de la figure 76, j'ai modifié la valeur de « Seuil température min » à 5 [°C] depuis mon site internet. Cette valeur correspond à l'index 0 (voir figure 64, p.37).

J'ai bien la trame cohérente, cependant pour la valeur (type double), j'ai utilisé un convertisseur en ligne[64], car le décodage manuel était trop complexe (norme IEEE 754). En entrant la valeur 5 dans le convertisseur, celui-ci m'a donné la valeur : 4014000000000000 en hexadécimal, correspondant aux 8 derniers octets de ma trame avant la fin. Cependant ces octets apparaissent inversés par rapport au convertisseur, probablement à cause du sens de lecture de l'oscilloscope.

D'autres tests de trames sont disponibles en annexe L.1, avec le même processus de décodage.

13.4 Contrôle batterie

Tension alimentation [V]	Affichage souhaité sur le site	Affichage obtenu sur le site
4.5	Rien	Rien
4.0	Piles faibles !	Piles faibles !
5	Piles faibles !	Piles faibles !

Tableau 10 Résultat contrôle batterie

Les trames indiquant l'envoi d'une alarme ou non, se trouveront en annexe L.1

13.5 Consommation courant total du circuit

13.5.1 Schéma de mesure

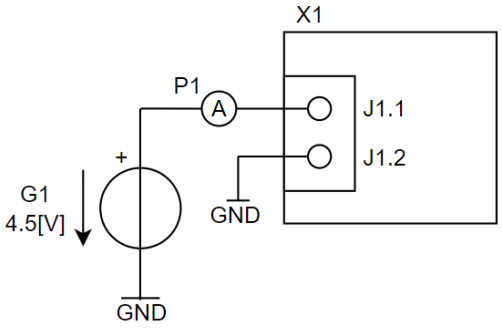


Figure 63 Schéma de mesure du courant

Méthode de mesure :

4) Brancher l'ampèremètre P1 en série
5) Alimenter le circuit avec du 4.5[V] à l'aide de G1.
6) Relever la mesure

[63] ASCII-Code.com. « Table des codes ASCII », [en ligne], (consulté le 21 septembre 2024), URL : https://www.ascii-code.com/fr
[64] BaseConvert. « IEEE 754 Floating Point », [en ligne], (consulté le 21 septembre 2024), URL : https://baseconvert.com/ieee-754-floating-point

Courant souhaité [mA]	Courant obtenu [mA]	Validation
1.425	1.1	OK*

Tableau 11 Comparaison consommation avec l'estimation

*La consommation est inférieure à celle estimée dans la pré-étude (voir tableau 2 point 3.3.2, p.10), ce qui garantit le bon fonctionnement du système et une meilleure autonomie.

Lors de mes mesures, j'ai laissé la LED de vie du capteur de température allumée, ce qui augmente la consommation du circuit. J'ai également réalisé des mesures en laissant la LED rouge indiquant que la batterie est faible allumée, ce qui là encore augmente d'autant plus la consommation du circuit :

	Courant souhaité [mA]	Courant obtenu [mA]
LED SparkFun Allumée	1.425	2.6
LED de vie + LED capteur	1.425	4.4

Tableau 12 Valeurs relevées avec les leds

Ces courants sont normaux pour les tests, ils ne resteront pas lors de la version définitive.

La LED d'alimentation du capteur SparkFun peut-être désactivée via le jumper PWRLED (voir schéma en annexe G). Cependant, pour la démonstration et les tests, j'ai choisi de la laisser activée.

La LED indiquant que les piles sont déchargées n'a pas été prise en compte dans l'estimation de la consommation, car cette LED n'est censée s'allumer que lorsque les piles arrivent en fin de vie ce qui est en dehors de l'état de fonctionnement normal du système.

14 Problèmes Rencontrés

14.1 MOSFET Q1

Lors du contrôle de tension des piles, je n'ai relevé aucune valeur sur mon ADC. J'ai donc contrôlé les tensions sur la gate et la source du MOSFET avec un multimètre pour vérifier si la condition de la figure 9, p.16 était correcte. Bien que celle-ci soit respectée, le drain affiche une tension très faible (0.1[mV]), ce qui me fait déduire que le MOSFET est défectueux.

Etant donnée la petite taille du composant, je ne le remplacerai pas avant la présentation. Le jumper J2 permet de simuler l'activation, confirmant que si Q1 fonctionnait, le contrôle de batterie serait opérationnel, surtout que Q2 semble fonctionner correctement.

14.2 E-paper

Problème de bruit, expliqué au point 13.2.2, p.42.

14.3 ESP32-C3

J'ai perdu un certain temps pour lancer la programmation de l'ESP32. En effet, j'avais l'erreur suivante :

```
A fatal error occurred: Failed to connect to ESP32-C3: No serial data received.
For troubleshooting steps visit: https://docs.espressif.com/projects/esptool/en/latest/troubleshooting.html
Failed uploading: uploading error: exit status 2
```

Figure 64 Erreur sur Arduino concernant l'ESP32

J'ai ensuite trouvé sur un forum[65], qu'il fallait à chaque modification de ligne, repasser l'ESP32 en mode download boot. J'ai indiqué cela dans le mode d'emploi se trouvant en annexe O, car il est nécessaire de le faire à chaque fois.

[65] Stackoverflow. « No serial data received: ESP32 CAM», [en ligne], (consulté le 9 septembre 2024), URL : https://stackoverflow.com/questions/76021459/no-serial-data-received-esp32-cam

14.3.1 Conflit avec STM32

Lors du démarrage du STM32, mon ESP32 cessait de fonctionner. Après un long moment à chercher l'origine du problème, j'ai fini par comprendre que cela était dû à la configuration du pin par défaut dans STM32CubeMx, comme mentionné au point 9.1.2, p.31.

14.4 Librairie Arduino

Au départ, j'avais utilisé un exemple de code[66] utilisant la librairie « HardwareSerial.h » pour permettre à l'ESP de communiquer en UART avec le STM. Malheureusement, cela ne fonctionnait pas. Après de nombreuses tentatives infructueuses pour faire fonctionner ce code, j'ai décidé de trouver d'autres exemples plus fonctionnels[67]. J'ai également dû modifier les pins utilisés dans le code en veillant à entrer les numéros des GPIO plutôt que les numéros de pin.

14.5 Problème connexion Wi-Fi

A l'ETML-ES, le réseau est protégé comme la montre cette figure :

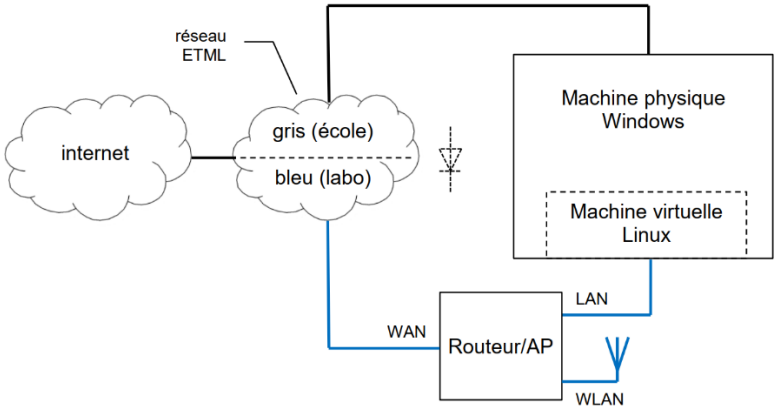


Figure 65 Structure réseau ETML

La diode représente le pare-feu de l'ETML-ES, empêchant les appareils non autorisés d'accéder au réseau. Pour que mon ESP32 puisse se connecter au Wi-Fi, j'ai configuré la machine virtuelle que j'utilise en mode « bridged », permettant d'utiliser la deuxième carte réseau sans contrainte de port ou autres.
Les informations pour connecter correctement l'ESP32 au Wi-Fi se trouvent dans le mode d'emploi, au point : Connexion ESP au Wi-Fi.

14.6 Problème réveil ESP32

Au cours du développement, je me suis rendu compte que cela pouvait en effet être utile de réveiller l'ESP32 depuis le STM32. En effet, cela permet à l'ESP32 de se mettre en mode sommeil après avoir réveillé le STM32, sans avoir à attendre que le STM32 ait fini ses tâches (économie de batterie). Dans les rares cas où le STM32 aurait à envoyer des trames UART à l'ESP32 (dépassement de seuils ou batterie faible), le STM32 s'assure alors que l'ESP32 soit bien réveillé et prêt à recevoir les informations avant de transmettre les trames.

Malheureusement, au cours du développement du projet je n'ai pas réussi à ce que le STM32 puisse réveiller l'ESP32 grâce à la pin « WAKE_UP_ESP32 ». En revanche, cela fonctionnait en utilisant la pin « ALARME ». J'ai donc finalement décidé d'utiliser la pin « ALARME » non pas pour communiquer l'état d'alarme mais pour réveiller l'ESP. L'état de l'alarme étant transmis par communication UART.

[66] Attari. « ESP32 UART Communication Explained with Example », Microcontrollers Lab, [en ligne], (consulté le 12 septembre 2024), publié le 20 août 2022. URL : https://microcontrollerslab.com/esp32-uart-communication-pins-example/
[67] Alb3rtov, « uart-communication-stm32-esp32 », GitHub, [en ligne], (consulté le 12 septembre 2024). URL : https://github.com/alb3rtov/uart-communication-stm32-esp32/tree/master

15 Etat d'avancement

15.1 Tâches effectuées

Voici les tâches effectuées selon le CDC durant les cinq semaines :

Hardware :

- Pré-étude avec choix des composants et analyse pile bouton / supercondensateur
- Conception du schéma électrique
- Conception du PCB
- Assemblage du PCB

Mécanique :

- Réalisation des boitiers (PCB et capteur de température / humidité)
- Impression des boitiers en PETG

Firmware :

STM32

- Communication entre l'ESP32 et le STM32
  - Réveil par l'ESP32
  - Réveil de l'ESP32
  - Réception et envoi trame UART
- Contrôle état de la batterie
- Lecture du capteur température / humidité
- Détection des dépassements de seuils
- Communication du STM32 au e-paper
  - Affichage des données sur le e-paper
  - Rafraichissement en cas de changement de valeurs
- Mode sommeil (sleep mode)

ESP32

- Connexion au Wi-Fi avec un SSID et un password
- Communication avec le serveur
  - Récupération des valeurs sur le serveur
  - Détection des changements de valeurs
  - Envoie des changement d'états (dépassement seuils et pile faible)
- Communication entre l'ESP32 et le STM32
  - Réveil par le STM32
  - Réception et envoi trame UART
  - Réveil du STM32
- Envoi notifications Discord
- Mode sommeil profond (Deep Sleep)

Software :

- Création page HTML simple -> passée en PHP
- Création base de données sur serveur
- Mise en place d'un serveur local

15.2 Tâches restantes

Mécanique :

- Coller les aimants dans les trous prévus à cet effet sur le boitier
- Coller le support de piles

Hardware :

- Retirer la liaison de la LED PWRLED du capteur SparkFun
- Changer le MOSFET Q1

16 Commentaires sur le déroulement du diplôme, Auto-analyse

16.1 Acquis du travail de diplôme

Ce travail de diplôme m'a permis de mettre en pratique mes acquis tout en développant de nouvelles compétences. L'ESP32, l'e-paper et la mise en service d'un serveur étaient des éléments nouveaux pour moi. J'ai aussi eu l'occasion de collaborer avec les informaticiens de l'ES, notamment pour découvrir des outils comme « XAMPP », que je n'aurais probablement pas connu sans eux.
Ce projet m'a particulièrement intéressée, notamment pour l'e-paper, une technologie d'avenir que je voulais apprendre à programmer. Mais au-delà de ça, j'ai également acquis des compétences concernant la mise en place d'un serveur, la configuration du mode sommeil d'un microcontrôleur, et bien plus. Des compétences qui me seront, j'en suis sûre, utiles tant personnellement que professionnellement.

16.2 Prise en compte de l'environnement social

Pouvoir surveiller la température et l'humidité d'un lieu tel qu'un réfrigérateur peut se révéler vitale. Que ce soit pour l'alimentation ou le stockage de médicaments, il est important de respecter la chaîne du froid pour éviter des problèmes de santé. A titre d'exemple, la durée maximale de rupture de la chaine du froid pour le vaccin contre la tuberculose est de deux heures[68].
Grâce à mon système d'alarme, le client peut être au courant des problèmes et réagir plus rapidement pour potentiellement sauver son médicament. Cette sécurité est également rassurante et devrait pouvoir apporter une certaine tranquillité d'esprit à son utilisateur.

16.3 Prise en compte de l'environnement naturel

L'utilisation d'un écran e-paper permet de minimiser la consommation d'énergie. En combinant cela avec la mise en veille et l'activation de l'ESP32 et du STM32 seulement toutes les heures, la durée de vie du circuit est largement prolongée. L'ajout d'un bornier permet également une flexibilité au niveau de l'alimentation.
Le PCB étant compact, cela réduit l'utilisation de ressource comme le cuivre. Cependant, cela complique également les dépannages liés aux problèmes éventuels. Il serait donc préférable de créer d'abord un prototype plus grand, puis de le réduire lors d'une production en série.
Il serait également judicieux d'envisager des alternatives aux composants susceptible d'être en rupture de stock, notamment l'ESP32 ou le STM32. De plus, produire le PCB en Suisse pourrait réduire l'empreinte carbone.
Enfin, pour optimiser d'avantage le système et réduire son impact environnemental, on pourrait doubler sa durée de vie en le maintenant en mode standby la plupart du temps, comme suggéré au point des améliorations possibles (voir point 17, p.50).

16.4 Leadership et développement personnel

En mettant en place une planification, j'ai pu suivre mon avancement et savoir si j'étais en avance ou non, ce qui m'a permis de travailler sans ressentir de réelle pression.
Toutefois, pour expliquer les erreurs rencontrées au cours du projet j'ai dû relire l'ensemble de mes notes écrites pour me souvenir en détail de ce qui s'était passé. Cela m'a fait perdre un temps précieux. Pour éviter cela à l'avenir, je m'assurerai de documenter au fur et à mesure les erreurs rencontrées.

La relation hiérarchique avec le client était constructive et fluide. Lors de nos échanges, il écoutait attentivement mes propositions et me donnait un retour clair. De plus, il me suggérait des solutions lorsque je rencontrais des blocages, ce qui a enrichi notre collaboration.

[68] Hôpitaux Universitaires Genève. « QUE FAIRE DES MÉDICAMENTS EN CAS DE RUPTURE DE LA CHAINE DU FROID ? », [en ligne], (consulté le 22 septembre 2024 », publié le 31.01.03, révision du 20.07.2023, p.2. URL : https://www.hug.ch/pharmacie/recommandations/document/réfrigérateur_contenu

En réfléchissent à la répartition des tâches, j'ai pris conscience que le projet comportait une part d'électronique, mais également d'information. Idéalement, j'imagine que la création et la gestion du côté serveur aurait pu être confiée à des informaticiens. En tant que cheffe de projet, j'aurais plutôt scindé ce travail de diplôme en deux parties : la conception du PCB et le développement du firmware (ESP32 et STM32) pour un technicien en électronique, et le développement web à un technicien en informatique. Cela aurait favorisé une collaboration d'équipe plutôt qu'un travail individuel. De plus, cela aurait permis un gain de temps considérable, car j'ai perdu beaucoup de temps sur des choses liées au serveur web. Ce qui me prenait parfois trois jours aurait j'imagine pris seulement une journée à un informaticien.

17 Améliorations possibles

1. Garantir la bonne précision de la température et de l'humidité relevées, par exemple en comparant avec celle d'un capteur haute précision.
2. Améliorer les informations concernant les dépassements de seuils. Le type de seuil dépassé pourrait être indiqué, pour distinguer s'il l'alarme provient d'un problème de température ou d'humidité. Il pourrait également être intéressant de connaître les valeurs mesurées au moment du dépassement pour se rendre compte de l'ampleur du dépassement.
3. Recevoir les notifications par SMS lors d'un dépassement d'alarme
4. Utiliser le mode « Standby » pour mettre en sommeil le STM32. Cela consomme beaucoup moins, et permettrait d'augmenter l'autonomie du système. Toutefois, cela implique une remise à zéro de la mémoire du STM. Il faudrait trouver un moyen efficace pour que l'ESP puisse conserver ou retrouver sa mémoire (valeurs affichées etc.). A noter qu'en mode standby la LED rouge indiquant que la pile est faible ne pourrait pas être gardée allumée. Cela signifie que le mode standby ne pourrait être utilisé que lorsque la LED n'a pas besoin d'être allumée.
5. Amélioration du design du site

18 Conclusion

Ce projet m'a permis d'améliorer mes compétences en programmation, mais également d'élargir mes connaissances à de nouveaux composants (ESP32 et e-paper). Au niveau hardware, le projet n'était pas excessivement complexe, cependant la partie firmware et software ont été de loin les plus compliquées. Le fait de mettre en place un serveur en mode local était assez complexe, heureusement l'aide des informaticiens m'a fait gagner plusieurs heures de recherches.

Durant la phase de pré-étude, j'ai passé un certain temps à réfléchir aux différentes pistes concernant l'alimentation. A partir du moment où les calculs ont été posés et effectués, cela a été beaucoup plus simple pour choisir la meilleure direction à prendre et j'ai été contente de pouvoir trouver une solution avec le client afin de pouvoir avancer sur le projet.

La phase de schéma a été assez rapide, car j'ai réutilisé certains schémas que j'avais déjà utilisé auparavant, sachant que ceux-ci fonctionnaient correctement. A l'inverse, la partie qui m'a pris le plus de temps était la partie e-paper, qui comportait deux versions différentes de datasheet.
Concernant la phase PCB, je n'ai pas eu de problème particulier.

Durant la partie firmware, j'ai utilisé deux librairies différentes pour m'aider à avancer plus rapidement. Celle pour le capteur de température et humidité et celle pour le e-paper. Pour la première, je n'ai pas eu de problème avec, cependant pour le e-paper, j'ai rencontré un problème de bruitage, dû à une mauvaise importation du code. Ce problème a été réglé en intégrant au fur et à mesure les fichiers nécessaire et en compilant à chaque fois pour vérifier qu'il n'y ait pas d'erreur.

Les parties les plus compliquées ont concerné la création du serveur et la programmation de l'ESP32. A noter qu'avec le routeur de l'ETML-ES, j'ai parfois des problèmes de connexion réseau, empêchant parfois mon ESP32 de se connecter au serveur pour effectuer les requêtes HTTP. N'ayant pas rencontré ce problème depuis chez moi, j'en ai déduit que cela était lié au réseau de l'ETML, potentiellement un problème de pare-feu qui pourrait empêcher les connexions.

De la même manière, l'envoi des notifications Discord ne fonctionne pas depuis le Wi-Fi de l'école, la connexion étant refusée. Cela est un peu frustrant car je ne peux malheureusement pas remédier à ce genre de problèmes.

Au final, je suis satisfaite d'avoir réussi à terminer un projet de cette ampleur en si peu de temps. Le projet est fonctionnel, et j'ai même eu le temps d'intégrer une option supplémentaire (envoi de notifications Discord lors d'une alarme).

J'ai apprécié participer à ce projet, il m'a permis de découvrir de nouvelles technologies (e-paper, ESP32, serveur local). De plus, je n'avais jamais eu l'occasion de créer un serveur en mode local, mais cela pourrait m'être utile dans le futur. Ce projet m'a permis de m'améliorer en programmation et je suis satisfaite de mon travail global.

Lausanne, le 24 septembre 2024 Perret Mélissa

19 Bibliographie

Rosset. Claude. et al. « FORMULAIRE : ELECTRONIQUE – TRANSMISSION », LEP Loisirs et Pédagogie SA, Le Mont-sur-Lausanne, 2018, p.21.

20 Webographie

ABLS. « HC/49US SMD LOW PROFILE CRYSTAL », [en ligne], (consulté le 20 août 2024). URL : https://abracon.com/Resonators/ABLS.pdf

Alb3rtov, « uart-communication-stm32-esp32 », GitHub, [en ligne], (consulté le 12 septembre 2024). URL : https://github.com/alb3rtov/uart-communication-stm32-esp32/tree/master

Angelini. Oliver. « Le filament PETG résiste-t-il à l'humidité ? », [en ligne], expert-scan3d.fr, (consulté le 21 août 2024). URL : https://www.expert-scan3d.fr/le-filament-petg-resiste-t-il-a-lhumidite/

Attari. « ESP32 UART Communication Explained with Example », Microcontrollers Lab, [en ligne], (consulté le 12 septembre 2024), publié le 20 août 2022. URL : https://microcontrollerslab.com/esp32-uart-communication-pins-example/

ASCII-Code.com. « Table des codes ASCII », [en ligne], (consulté le 21 septembre 2024), URL : https://www.ascii-code.com/fr

BaseConvert. « IEEE 754 Floating Point », [en ligne], (consulté le 21 septembre 2024), URL : https://baseconvert.com/ieee-754-floating-point

Computer Hope. « How to create an HTML back button », [en ligne], (consulté 15 septembre 2024), mise à jour le 31.12.2017. URL : https://www.computerhope.com/issues/ch000317.htm

Controllerstech. « Low Power Modes in STM32 », [en ligne], (consulté le 15 septembre 2024). URL : https://controllerstech.com/low-power-modes-in-stm32/

Digikey. « Calculateur d'autonomie des batteries », [en ligne], (consulté le 20 août 2024). URL : https://www.digikey.fr/fr/resources/conversion-calculators/conversion-calculator-battery-life

Diodes incorporated, « AP63200/AP63201/AP63203/AP63205 », [en ligne], (consulté le 20 août 2024), révision 2-2. URL : https://www.diodes.com/assets/Datasheets/AP63200-AP63201-AP63203-AP63205.pdf

Electrical Enginneering. « ESP32-C3 Deep Sleep Wake-Up », [en ligne], (consulté le 15 septembre 2024). URL : https://electronics.stackexchange.com/questions/677205/esp32-c3-deep-sleep-wake-up

Espressif Systems. « Arduino IDE Tools Menu » [en ligne], (consulté le 15 septembre). URL : https://docs.espressif.com/projects/arduino-esp32/en/latest/guides/tools_menu.html#core-debug-level

Espressif Systems, « Basic Use of the Extension », Github, [en ligne], (consulté le 15 septembre 2024). URL : https://github.com/espressif/vscode-esp-idf-extension/blob/master/docs/tutorial/basic_use.md

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

Espressif Systems. « ESP32-C3 : Downloading Guide », [en ligne], (consulté le 9 septembre 2024), master(latest). URL : https://docs.espressif.com/projects/esp-at/en/latest/esp32c3/Get_Started/Downloading_guide.html

Espressif Systems, « ESP32-C3 : Hardware Design Guidelines », [en ligne], (consulté le 26 août 2024), datasheet du 29 août 2024. URL : https://docs.espressif.com/projects/esp-hardware-design-guidelines/en/latest/esp32c3/esp-hardware-design-guidelines-en-master-esp32c3.pdf

Espressif Systems. « ESP32-C3-WROOM-02, ESP32-C3-WROOM-02-U : Datasheet », [en ligne], (consulté le 20 août 2024), version 1.3. URL : https://www.espressif.com/sites/default/files/documentation/esp32-c3-wroom-02_datasheet_en.pdf

Espressif Systems, « examples/system/deep_sleep/main/gpio_wakeup.c », Github, [en ligne], (consulté le 15 septembre 2024). URL : https://github.com/espressif/esp-idf/blob/master/examples/system/deep_sleep/main/gpio_wakeup.c

Espressif Systems, « Hardware Connection », [en ligne], (consulté le 20 août 2024), master(latest). URL : https://docs.espressif.com/projects/esp-at/en/latest/esp32c3/Get_Started/Hardware_connection.html

Espressif Systems, « Installation », GitHub, [en ligne], (consulté le 15 septembre 2024). URL : https://github.com/espressif/vscode-esp-idf-extension/blob/master/docs/tutorial/install.md

Espressif Systems, « libraries/ESP32/examples/DeepSleep/ExternalWakeUp/ExternalWakeUp.ino », [en ligne], GitHub, (consulté le 15 septembre 2024). URL : https://github.com/espressif/arduino-esp32/blob/master/libraries/ESP32/examples/DeepSleep/ExternalWakeUp/ExternalWakeUp.ino

Espressif Systems. « Released Firmware », [en ligne], (consulté le 22 septembre 2024), master(latest). URL : https://docs.espressif.com/projects/esp-at/en/latest/esp32c3/AT_Binary_Lists/esp_at_binaries.html

Espressif Systems, « Sleep Modes », [en ligne], (consulté le 15 septembre 2024). URL : https://docs.espressif.com/projects/esp-idf/en/latest/esp32c3/api-reference/system/sleep_modes.html

ETML-ES, « Programmation web : 00 Introduction », slide 6.
FTDI. TTL-232R TTL TO USB SERIAL CONVERTER RANGE OF CABLES Datasheet », [en ligne], (consulté le 29 août 2024), version 2.05. URL : https://ftdichip.com/wp-content/uploads/2023/07/DS_TTL-232R_CABLES.pdf

Hôpitaux Universitaires Genève, « QUE FAIRE DES MÉDICAMENTS EN CAS DE RUPTURE DE LA CHAINE DU FROID ? », [en ligne], (consulté le 22 septembre 2024 », publié le 31.01.03, révision du 20.07.2023, URL : https://www.hug.ch/pharmacie/recommandations/document/frigo_contenu

Hutscape.com, « External wakeup with Arduino on ESP32-C3 », [en ligne], (consulté le 15 septembre 2024). URL : https://hutscape.com/tutorials/external-wakeup-arduino-esp32c3

Infineon, « BAT60A…», [en ligne], (consulté le 20 août 2024), datasheet du 2007-04-19. URL : https://www.infineon.com/dgdl/bat60aseries.pdf?folderId=db3a304313d846880113def5812204a1&fileId=db3a304313d846880113def70c9304a9

Ionos Digital Guide, « Tutoriel XAMPP : Installation et premiers pas », [en ligne], (consulté le 08 september 2024), publié le 01.03.2023. URL : https://www.ionos.fr/digitalguide/serveur/outils/tutoriel-xampp-creer-un-serveur-de-test-local/

KEYSTONE, « ECONOMICAL PLASTIC BATTERY HOLDERS », [en ligne], (consulté le 26 août 2024). URL : https://www.keyelco.com/userAssets/file/M65p28.pdf

Medium, « PHP REST API », [en ligne], (consulté le 15 septembre 2024), 28.06.2024. URL : https://medium.com/@dharshithasrimal/php-rest-api-7441197312d7

MICROCHIP, « Calculating crystal load capacitor », [en ligne], (consulté le 20 août 2024), publié le 11 septembre 2020. URL : https://microchip.my.site.com/s/article/Calculating-crystal-load-capacitor

Omnergy, « SPECIFICATION FOR LITHIUM BATTERY : MODEL CR2477 », [en ligne], (consulté le 20 août 2024), URL : https://www.farnell.com/datasheets/1496886.pdf

OpenClassrooms, « Créez votre site web avec HTML5 et CSS3 », [en ligne], (consulté le 1er septembre 2024), mis à jour le 18/06/2024. URL : https://openclassrooms.com/fr/courses/1603881-creez-votre-site-web-avec-html5-et-css3

Random Nerd Tutorial, « ESP32 External Wake Up from Deep Sleep », [en ligne], (consulté le 15 septembre). URL : https://randomnerdtutorials.com/esp32-external-wake-up-deep-sleep/

Random Nerd Tutorial, « ESP32 HTTP GET and HTTP POST with Arduino IDE (JSON, URL Encoded, Text) », [en ligne], (consulté le 15 septembre). URL : https://randomnerdtutorials.com/esp32-http-get-post-arduino/

Random Nerd Tutorial, « ESP32 Timer Wake Up from Deep Sleep », [en ligne], (consulté le 15 septembre). URL : https://randomnerdtutorials.com/esp32-timer-wake-up-deep-sleep/

Random Nerd Tutorial, « Installing the Esp32 Board in Arduino IDE (Windows, Mac OS X, Linux) », [en ligne], (consulté le 08 septembre 2024). URL : https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/#:~:text=To%20install%20the%20ESP32%20board%20in%20your%20Arduino,It%20should%20be%20installed%20after%20a%20few%20seconds

Ravi Thanki, « Accessing localhost (xampp) from another computer over LAN network », [en ligne], (consulté le 15 septembre 2024), publié le 21.02.2019. URL : https://ravithanki.wordpress.com/2019/02/21/accessing-localhost-xampp-from-another-computer-over-lan-network/

Robert. Karl-Emerik, « Guide Complet sur le Filament PETG : Propriétés, Utilisations et Avantages pour l'Impression 3D », [en ligne], (consulté le 21 août 2024), publié le 6 mai. URL : https://www.machine3d.fr/post/guide-complet-sur-le-filament-petg-propri%C3%A9t%C3%A9s-utilisations-et-avantages-pour-l-impression-3d

Robert. Karl-Emerik, « Le Filament PLA et l'Humidité : Comprendre et prévenir les Problèmes. », [en ligne], (consulté le 21 août 2024), publié le 18 juin. URL : https://www.make3dprinting.com/post/le-filament-pla-et-l-humidit%C3%A9-comprendre-et-pr%C3%A9venir-les-probl%C3%A8mes

SATURN PCB DESIGN, INC, « Saturn PCB Toolkit Help », [en ligne], (consulté le 02 septembre 2024). URL : https://saturnpcb.com/saturn-pcb-toolkit-help-htm/

Sculpteo, « Compatibilité alimentaire des impressions 3D : Quels matériaux choisir ? », [en ligne], (consulté le 21 août 2024). URL : https://www.sculpteo.com/fr/centre-apprentissage/choisissez-le-meilleur-materiau-impression-3d/compatibilite-alimentaire-impression-3d/

SENSIRION, « Datasheet SHTC3 : Humidity and Temperature Sensor IC », [en ligne], (consulté le 29 août 2024), version 2 – juin 2019. URL : https://cdn.sparkfun.com/assets/1/1/f/3/b/Sensirion_Humidity_Sensors_SHTC3_Datasheet.pdf

SparkFun, « Dimensions in inches », [en ligne], (consulté le 29 août 2024). URL : https://cdn.sparkfun.com/assets/learn_tutorials/1/1/6/9/SHTC3_Dimensions.png

SparkFun, « SparkFun Humidity Sensor Breakout - SHTC3 (Qwiic) Hookup Guide », [en ligne], (consulté le 20 août 2024). URL : https://learn.sparkfun.com/tutorials/sparkfun-humidity-sensor-breakout---shtc3-qwiic-hookup-guide#hardware-overview, chapitre jumpers

ST, « STM32F072x8 STM32F072xB », [en ligne], (consulté le 20 août 2024), datasheet de septembre 2019, révision 6. URL : https://www.st.com/content/ccc/resource/technical/document/datasheet/cd/46/43/83/22/d3/40/c8/DM00090510.pdf/files/DM00090510.pdf/jcr:content/translations/en.DM00090510.pdf

Stackoverflow, « Could not open mysql.plugin table. Some plugins may be not loaded », [en ligne], (consulté 15 septembre 2024). URL : https://stackoverflow.com/questions/34198735/could-not-open-mysql-plugin-table-some-plugins-may-be-not-loaded

Stackoverflow, « How get size of $ GET », [en ligne], (consulté le 15 septembre 2024). URL : https://stackoverflow.com/questions/25068290/how-get-size-of-get

Stackoverflow, « Get and insert Data into MySQL database using PHP with xampp on localhost », [en ligne], (consulté le 15 septembre 2024). URL : https://stackoverflow.com/questions/38812682/get-and-insert-data-into-mysql-database-using-php-with-xampp-on-localhost

Stackoverflow, « loop through $ GET results », [en ligne], (consulté 15 septembre 2024). URL : https://stackoverflow.com/questions/1222244/loop-through-get-results

Stackoverflow, « No serial data received: ESP32 CAM», [en ligne], (consulté le 9 septembre 2024), URL : https://stackoverflow.com/questions/76021459/no-serial-data-received-esp32-cam

Stackoverflow, « Why Xampp control panel shows an error access denied upon launch [closed] », [en ligne], (consulté le 15 septembre 2024). URL : https://stackoverflow.com/questions/38676374/why-xampp-control-panel-shows-an-error-access-denied-upon-launch

SunLED, « XZCM2CRK54WA-1VF », [en ligne], (consulté le 20 août 2024). URL : https://www.sunledusa.com/products/spec/XZCM2CRK54WA-1VF.pdf
Tag-connect. URL : https://www.tag-connect.com/wp-content/uploads/bsk-pdf-manager/TC2030-CTX_1.pdf

TDK, « STM inductors », [en ligne], (consulté le 20 août 2024), datasheet de juin 2019. URL : https://www.tdk-electronics.tdk.com/inf/30/db/ind_2008/b82432t.pdf

Techtutorialsx, « ESP32: Parsing JSON », [en ligne], consulté le 15 septembre). URL : https://techtutorialsx.com/2017/04/26/esp32-parsing-json/

TEXAS INSTRUMENTS, « I2C Bus Pullup Resistor Calculation », [en ligne], (consulté le 29 août 2024). URL : https://www.ti.com/lit/an/slva689/slva689.pdf?ts=1724585444033

TOSHIBA, « SSM3J35AFV » [en ligne], (consulté le 20 août 2024), datasheet du 22 juin 2017, révision 2.0. URL : https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/754/SSM3J35AMFV.pdf

Waveshare, « e-Paper/EPD_2in13_V4.c », GitHub, [en ligne], (consulté le 21 septembre 2024). URL : https://github.com/waveshareteam/e-Paper/blob/master/STM32/STM32-F103ZET6/User/e-Paper/EPD_2in13_V4.c

Waveshare, « 2.13inch e-Paper », [en ligne], consulté le 9 août 2024, version 4. URL : https://files.waveshare.com/upload/4/4e/2.13inch_e-Paper_V4_Specification.pdf

WAVESHARE, « 2.13inch e-Paper », [en ligne], (consulté le 20 août 2024), version 2. URL : https://files.waveshare.com/upload/d/d5/2.13inch_e-Paper_Specification.pdf

Waveshare, « 2.13inch e-Paper HAT Manual », [en ligne], (consulté le 20 août 2024). URL : https://www.waveshare.com/wiki/2.13inch_e-Paper_HAT_Manual#Working_With_STM32

Waveshare, « 250x122, 2.13inch E-Ink raw display panel », [en ligne], (consulté le 20 août 2024). URL : https://www.waveshare.com/2.13inch-e-paper.htm

Waveshare, « waveshareteam / e-Paper », Github, [en ligne], (consulté le 15 septembre 2024). URL : https://github.com/waveshareteam/e-Paper

W3schools.com, « Select Data From a MySQL Database », [en ligne], (consulté le 15 septembre 2024). URL : https://www.w3schools.com/php/php_mysql_select.asp

W3schools.com, « Update Data In a MySQL Table Using MySQLi and PDO », [en ligne], (consulté le 15 septembre 2024). URL : https://www.w3schools.com/pHp/php_mysql_update.asp

21 Logiciels utilisés

Nom logiciel	Description	Version
Altium Designer	Conception du PCB	23.6.0
SolidWorks	Modélisation 3D	2023 SP3.0
STM32CubeMX	Configurateur graphique STM32	6.11.1
µVision	IDE de programmation (STM32)	5.39.0.0
µVision	Compilateur ARM intégré à µVision	5
ArduinoIDE	IDE de programmation (ESP32)	2.3.2
ArduinoIDE	Interface ligne de commande (CLI) Arduino	0.35.3
XAMPP	Logiciel pour serveur web local	3.3.0
Notepad++	Editeur de texte	8.6.7

Tableau 13 Logiciels utilisés

22 Figures

Figure 1 Schéma général 7
Figure 2 Schéma bloc global 7
Figure 3 Schéma bloc sous Altium 13
Figure 4 Schéma représentant l'entrée V_PILES ainsi que le régulateur de tension 13
Figure 5 Application typique du circuit 14
Figure 6 Valeur des composants recommandés 14
Figure 7 Courbe impédance en fonction de la fréquence à +20[°C] 14
Figure 8 Schéma représentant le contrôle de la pile 15
Figure 9 Tension VGS en fonction du courant ID 16
Figure 10 Schéma reset et debug 16
Figure 11 Schéma STM32 17
Figure 12 Configuration des pins entre le STM32 et l'e-paper 18
Figure 13 Schéma du bootloader pour le STM32 18
Figure 14 Schéma du capteur de température et humidité 19
Figure 15 Temps de communication pour le capteur SHTC3 19
Figure 16 Schéma pour l'ESP32 20
Figure 17 Configuration des pins UART de l'ESP32 20
Figure 18 Information réinitialisation filtre RC 21
Figure 19 Liaison du câble TTL-232R-3V3 21
Figure 20 Mode de démarrage pour l'ESP32 22
Figure 21 Schéma sélection du mode de l'ESP32 22
Figure 22 Schéma pour le e-paper 22
Figure 23 Indication concernant la pin 8 23
Figure 24 Coupure du PCB pour le câble du e-paper 24
Figure 25 Zone de keepout réservée pour insérer le PCB dans le boîtier 24
Figure 26 Recommandation de placement pour l'ESP32 25
Figure 27 Recommandation de coupe du PCB pour l'antenne de l'ESP32 25
Figure 28 Règle de connexion pour les plans de masse 26
Figure 29 Face arrière de l'ESP32 26
Figure 30 Vue du quartz du STM32 27
Figure 31 Dimensionnement de largeur de piste pour les signaux 27
Figure 32 Sélection des règles concernant le DRC 28

Figure 33 Estimation hauteur du boitier....28
Figure 34 Vue 3D de l'assemblage du boitier avec PCB....29
Figure 35 Vue 3D du boitier fermé avec PCB....29
Figure 36 Schéma de mesure....30
Figure 37 Configuration des pins....30
Figure 38 Configuration GPIOs....31
Figure 39 Configuration ADC....32
Figure 40 Configuration de l'UART1 (Espressif)....32
Figure 41 Configuration de l'UART1 (STM32)....32
Figure 42 Configuration SPI....33
Figure 43 Configuration I2C....33
Figure 44 Configuration du quartz....34
Figure 45 Modification du heap size....34
Figure 46 Machine d'état principale....35
Figure 47 Schéma pour programmer l'ESP32....36
Figure 48 Fonctionnement global....36
Figure 49 Fonction setup....37
Figure 64 Résultat final du site internet....37
Figure 65 Fonctionnement globale du système....37
Figure 66 Interface XAMPP avec Apache et MySQL activés....38
Figure 67 Schéma de mesure I2C....39
Figure 68 Début trame I2C (envoi, lecture et commande MSB et LSB)....40
Figure 69 Fin trame I2C (humidité et température)....41
Figure 70 Formule pour obtention de l'humidité et température....41
Figure 71 Schéma de mesure SPI....42
Figure 72 Ecran e-paper bruité....42
Figure 73 Format de trame souhaité d'après le datasheet....43
Figure 74 Début initialisation avec le baudrate à 500[kBits/s]....43
Figure 75 Schéma de mesure UART....44
Figure 76 Envoie trame du serveur au STM32, modification index 0 à 5[°C]....44
Figure 77 Schéma de mesure du courant....45
Figure 78 Erreur sur Arduino concernant l'ESP32....46
Figure 79 Structure réseau ETML....47
Figure 80 liste des fichiers utilisés de la librairie....59

23 Tableaux

Tableau 1 Pins minimum requises....8
Tableau 2 Estimation réaliste de la consommation pondérée du circuit par mois....10
Tableau 3 Estimation de la consommation pondérée du circuit par mois (marges élevées)....11
Tableau 4 Estimation du coût....12
Tableau 5 Définition des pins UART0 et UART1....21
Tableau 6 Courant max en fonction de la largeur des pistes....27
Tableau 7 Tableau de comparaison entre coût estimé et réel....30
Tableau 8 Tension relevée....30
Tableau 9 Valeurs relevées....41
Tableau 10 Résultat contrôle batterie....45
Tableau 11 Comparaison consommation avec l'estimation....46
Tableau 12 Valeurs relevées avec les leds....46
Tableau 13 Logiciels utilisés....56

24 Projets références

2320_DeclencheurADistancePiegePhoto
2407_CommandeOuvertureFrigoRFID

Annexe A Cahier des charges

Annexe B Planification

Annexe C Journal de travail

Annexe D Procès-verbaux

Annexe E Schéma complet carte principale

Annexe F Fichier de fabrication

- **F.1 Assembly**
- **F.2 Draftman**

Annexe G Liste de pièces

Annexe H Schéma carte capteur de température

Annexe I Checklist revue de schéma et PCB

Annexe J Plan d'assemblage boitiers

- **J.1 Plan couvercle PCB V1**
- **J.2 Plan boitier base PCB V1**
- **J.3 Plan couvercle capteur température/humidité V1**
- **J.4 Plan boitier base capteur température/humidité V1**

Annexe K Liste de matériel

Annexe L Mesures

- **L.1 UART**
- **L.2 SPI**

Annexe M Librairies GitHub utilisées

Annexe N Installations liées à Arduino IDE

Annexe O Mode d'emploi

Annexe P Vérification des checksum I2C

- **P.1 Checksum humidité**
- **P.2 Checksum température**

Annexe Q Calculs utilisés pour l'estimation de consommation

Annexe R Flash ESP32 pour AT commandes

Annexe S Listing code STM32

S.1 shtc3.c

S.2 shtc3.h

S.3 affichage.c

S.4 affichage.h

S.5 gestionBatterie.c

S.6 gestionBatterie.h

S.7 fonctionsADC.c

S.8 fonctionsADC.h

S.9 communication.c

S.10 communication.h

S.11 mesures.c

S.12 mesures.h

S.13 sommeil.c

S.14 sommeil.h

S.15 main.c

S.16 main.h

S.17 librairie e-paper lien github

https://github.com/waveshareteam/e-Paper/tree/master/STM32
fichiers utilisés:

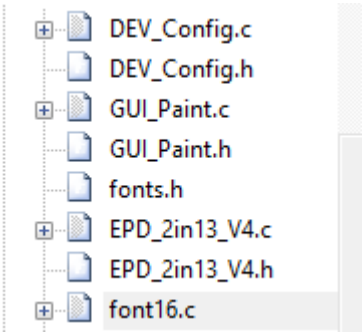


Figure 66 liste des fichiers utilisés de la librairie

Annexe T Listing code ESP32

T.1 2409_ESP_V1 ino

T.2 CommunicationServeur.h

T.3 CommunicationServeur.ino

T.4 CommunicationSTM.h

T.5 CommunicationSTM.ino

T.6 Discord.h

T.7 Discord.ino

T.8 Execution.ino

T.9 Execution.h

T.10 Sommeil.h

T.11 Sommeil.ino

T.12 Wifi.h

T.13 Wifi.ino

Annexe U Listing code web

U.1 Home.php

U.2 Form.php

U.3 Api.php

Annexe V Flowchart STM32

Annexe W Flowachat ESP32

Annexe X Flowchart web

Annexe Y Fiche de modification