UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

COMME EXIGENCE PARTIELLE
DE LA MAITRISE EN GÉNIE ÉLECTRIQUE

PAR
FRÉDÉRICK CHARTIER-BEAULNE

CONCEPTION D'UNE PASSERELLE RFID À WI-FI

AVRIL 2017

Résumé

Ce projet a comme objectif la création d'une passerelle RFID à Wi-Fi.

Cette passerelle est développée dans le but de fabriquer une plate-forme de développement pour la création de produits voulant utiliser la technologie de celle-ci. Afin d'accomplir cet objectif, un module RFID et un module Wi-Fi ont été choisis ainsi qu'un contrôleur qui vient agir comme l'intelligence de la passerelle. Le tout a été assemblé sur un circuit imprimé conçu et fabriqué durant le projet.

La programmation actuelle de la passerelle donne une preuve de concept de celle-ci en effectuant une lecture multiple d'étiquettes RFID. Après cette lecture, le résultat est envoyé par Wi-Fi à un ordinateur qui, par la suite, affiche les étiquettes à l'utilisateur.

Dans ce document, nous allons dans le détail des modules utilisés. Premièrement, en expliquant le choix et ensuite en exposant leur implémentation dans le projet. Puis, nous entrons dans le désign et les fonctionnalités de la passerelle incluant les composantes matérielles et logicielles.

Remerciements

Je voudrais d'abord remercier mon directeur et co-directeur, Frédéric Domingue et Adel Omar Dahmane, pour leur support autant financier que technique durant le projet. Sans ce support, ce projet n'aurait pas pu être mené à bien. Je voudrais aussi les remercier de m'avoir présenté ce projet que j'ai trouvé motivant et aussi une continuation naturelle du développement de mes aptitudes et de mon intérêt pour les systèmes embarqués que j'ai développés au baccalauréat.

Je tiens aussi à remercier les autres étudiants du LSMT pour leur accueil chaleureux au sein de leur équipe. Leur présence fut source d'inspiration et de motivation tout au long de ma maîtrise.

Table des matières

Liste des tableaux

Liste des figures

Liste des symboles et abréviations

V : Volt

W : Watt

A : Ampère

VDC : Volt courant continue

kB : kilo octet

dBm : décibel par milliwatts

UART : Universal Asynchronous Receiver Transmitter

SMT : Surface Mount Technology

MCU : Microcontroller Unit

CRC : Cycle Redundancy Check

RFID : Radio Frequency Identification Device

ISR : Interrupt Service Routine

PCB : Printed Circuit Board

GPIO : General Purpose Inputs Outputs

BMP : Bitmap

IP : Internet Protocol

FAT : File Allocation System

Chapitre 1 - Introduction

1.1 Problématique

L'intérêt pour la radio identification (RFID) a augmenté considérablement dans les dernières années. La technologie a rendu possible l'intégration de l'intelligence dans des objets physiques. Il est maintenant possible d'obtenir des solutions RFID à bas prix pour des applications commerciales et industrielles.

Les technologies RFID ont plusieurs utilisations dans la traçabilité d'inventaire, des biens, personnes, etc. Un des avantages de celles-ci est la possibilité de lire des centaines d'étiquettes en même temps. Donc, il est possible de surveiller plusieurs items étiquetés par RFID à l'aide d'un lecteur.

Un autre mouvement technologique qui a commencé à gagner de l'intérêt est l'"Internet of things" qui est une vision selon laquelle une grande majorité des systèmes embarqués seront capables d'opérer à travers l'internet. Le but du projet étant de faire une passerelle entre la RFID et le Wi-Fi, celui-ci s'intègre bien dans cette vision du futur, car il va permettre de relier les lecteurs d'étiquettes à la structure internet.

Avec la création d'une passerelle qui permet de faire un développement flexible pour de

multiples applications, il sera facile de faire des prototypes fonctionnels d'équipement utilisant les deux technologies. Par exemple, le module pourrait être utilisé pour fabriquer un réfrigérateur intelligent où les aliments sont étiquetés RFID et lus par le module. Le module pourrait ensuite envoyer les informations du contenu du réfrigérateur et possiblement d'autres informations comme sa température par le réseau internet pour les rendre accessibles au propriétaire.

1.2 Objectif

L'objectif de ce projet est de créer une passerelle qui nous permettra de lire des étiquettes RFID et d'envoyer l'information à travers le réseau. Cette passerelle servira d'outil de développement pour de projets futurs qui pourraient bénéficier de cette technologie. Afin de réaliser ce projet, un contrôleur doit être choisi ainsi que des modules qui permettront d'effectuer les tâches de lecture d'étiquettes et d'envoi par Wi-Fi. Un code qui dicte au contrôleur comment communiquer avec ces modules doit être mis au point pour atteindre notre but. Finalement, une carte qui réunit les dispositifs et le contrôleur devra être fabriquée.

1.3 Méthodologie

La méthodologie de conception commence par le choix du contrôleur ainsi que des modules à utiliser pour la lecture et le transfert de l'information des étiquettes RFID sur le réseau. À noter que le module RFID a été fourni lorsque j'ai commencé le projet et donc très peu de considération supplémentaire a été apportée à son choix.

Une fois sélectionné, les modules ont été testés séparément avec le microcontrôleur afin

d'assurer le bon fonctionnement de ceux-ci et afin de commencer à écrire du code qui pourra être utilisé plus tard. Une fois que ce code rudimentaire, mais fonctionnel a été écrit, les modules ont été joints ensemble pour faire une preuve de concept.

Finalement, une carte électronique qui peut accueillir les modules a été mise au point. Ceci nous donne un bon outil de développement pour une utilisation de cette technologie de passerelle dans le futur et conclura le projet.

1.4 Organisation du mémoire

Ce mémoire est organisé en cinq chapitres, le premier chapitre étant l'introduction et le dernier la conclusion. Le sujet du chapitre deux est le module RFID utilisé. Dans ce chapitre des détails sur le module seront présentés tels que ses spécifications et son intégration au circuit imprimé. De plus, les commandes utilisées pour la communication entre ce module et le microcontrôleur seront présentées ainsi que l'algorithme qui a été développé afin d'utiliser le module.

Dans le chapitre trois, le sujet est le dispositif Wi-Fi. Dans la première section, celui-ci sera présenté de la même manière que le module RFID. Par la suite, nous étalerons les configurations du module disponibles et celles appliquées dans le projet ainsi que le code développé pour les enregistrer dans le module. Finalement, nous détaillerons la méthodologie d'envoi en regardant le protocole utilisé et les algorithmes d'envoi et de réception.

L'avant-dernier chapitre portera sur la passerelle entre les deux modules. La première section détaillera le choix de technologie pour relier les dispositifs. Puis, nous enchaînerons sur le design et la fabrication du circuit imprimé en survolant les circuits d'alimentation,

de débogage et d'interface humaine. Puis, nous irons en détail dans chaque section de code de l'interface humaine sur la passerelle. Finalement, le chapitre conclura sur les résultats obtenus.

Chapitre 2 - Module RFID

2.1 Le module et ses spécifications

2.1.1 Module utilisé

Le module RFID utilisé provient de la série de produits Mercury de ThingMagic qui sont des circuits embarqués que l'on peut intégrer à d'autres systèmes. Plus spécifiquement, le modèle utilisé est le M5e-C ou le M5e-Compact. Le modèle compact nécessite moins de puissance pour fonctionner et a une plus grande plage de tension opérationnelle. De plus, il est près de la moitié de la dimension de sa version non compacte. La table 2.1, prise directement du manuel du développeur dans les références [1], compare les caractéristiques des deux versions du M5e.

Les étiquettes RFID lues par le module sont des étiquette EPC Gen 2. Ces étiquettes ont des banques mémoires qui peuvent aller jusqu'à 496 bits. Nous pouvons donc créer des étiquettes avec de l'information dans ces plages de mémoire et les lire. Cette information pourrait être simplement des caractères qui nous aident à identifier l'objet auquel l'étiquette sera liée. Par exemple, on pourrait créer des étiquettes qui identifient les objets qui sortent d'une ligne de production avec leur date de production, numéro de série, usine de fabrication, etc. Ou encore on pourrait créer des étiquettes pour des produits alimentaires afin de les identifier avec

leur date d'expiration et leur positionnement dans le magasin où ils sont vendus.

2.1.2 Spécifications du module

Le module M5e-C a une dimension de 56mm par 35.6mm avec des trous de 100 mils pour pouvoir fixer celui-ci à l'aide de vis. La figure 2.1 montre une photo du module utilisé. Ses pattes de sortie sont reliées à un connecteur pour câble plat à 12 pins qui a un pas de 1mm. Dans le projet, j'ai utilisé une tension de 5V pour alimenter le module. Selon le tableau de comparaison 2.1, nous pouvons atteindre une puissance maximale de 2.7W et donc le PCB doit pouvoir fournir au moins 0.54A au M5e-C. Le module supporte différentes régions en utilisant une table interne de fréquences pour la région sélectionnée. La communication UART du module opère à un baud rate de 9600 par défaut et peut être augmenté après la séquence de démarrage.

2.1.3 Intégration au PCB final

Une espace selon les dimensions du module a été allouée sur la couche inférieure pour celui-ci. Des trous de 100 mils selon les chartes de clairance de vis ont été placés aux bons endroits afin de fixer les supports du module. Les entretoises choisies ont une hauteur de 3/8 de pouces et accueillent des vis 2-56. La figure 2.2 montre une photo du montage du module M5e-C sur la carte mère.

Pour faire la connexion du module au board, j'ai utilisé le câble plat 12 positions qui venait avec le kit de développement du ThingMagic. Pour le connecteur que l'on peut voir à la figure 2.3, j'ai retrouvé celui utilisé par le kit. Vu l'emplacement du module, le connecteur avec les

Item	M5E	M5e-Compact
Processor	Atmel AT91SAM7S-256	Atmel AT91SAM7S-256
Flash memory	256kB	256kB
On-chip RAM	64 kB	64 kB
RF Architecture	ASIC Impinj Indy1000 with ThingMagic front end for improved sensitivity	ASIC Impinj Indy1000 with ThingMagic front end for improved sensitivity
Input Power Requirements	+5VDC +/-4% 7.5W max (6.5 typical) 25mV max peak-peak ripple no spectral spike greater than 5mVpp in any 1kHz band	+3 to +5.5VDC 2.7W max (2.6 typical) 25mV max peak-peak ripple no spectral spike greater than 5mVpp in any 1kHz band
Protocols supported	GEN2, ISO 18000-6C	GEN2, ISO 18000-6C
Dimensions :	82mm L x 54mm W x 5mm H	56mm L x 35.6mm W x 5mm H
Regions supported	NA, EU (3 variations), Korea,India, China	NA, EU (3 variations), Korea, India, China
Electrostatic Discharge	• IEC-61000-4-2 discharge direct to operational antenna port tolerates max 300 Volt Pulse • MIL-883 3015.7 discharge direct to operational antenna port tolerates max 1200 Volt Pulse Note : Survival level varies with antenna return loss and antenna characteristics. See Appendix E : Environmental Considerations for methods to increase ESD tolerances.	

TABLE 2.1 – Comparaison des fonctionnalités entre le M5e et le M5e-Compact

FIGURE 2.1 – Photo du module ThingMagic compact M5e-C utilisé

contacts sur le bas a été choisi.

Pour acheminer le courant au M5e-C j'ai utilisé un adaptateur mural qui fournit une tension de 5V et un courant jusqu'à 1.5A, ce qui convient parfaitement aux besoins de toutes les composantes du circuit. La figure 2.4 donne une image de l'adapteur. Son connecteur est de type barrel plug SMT. Tous ces pièces peuvent être trouvé à partir de leur numéro Digikey dans la table 4.1.

Finalement, la sortie et l'entrée UART sont reliées du connecteur à câble plat jusqu'aux pattes désignées à cette tâche sur le MCU.

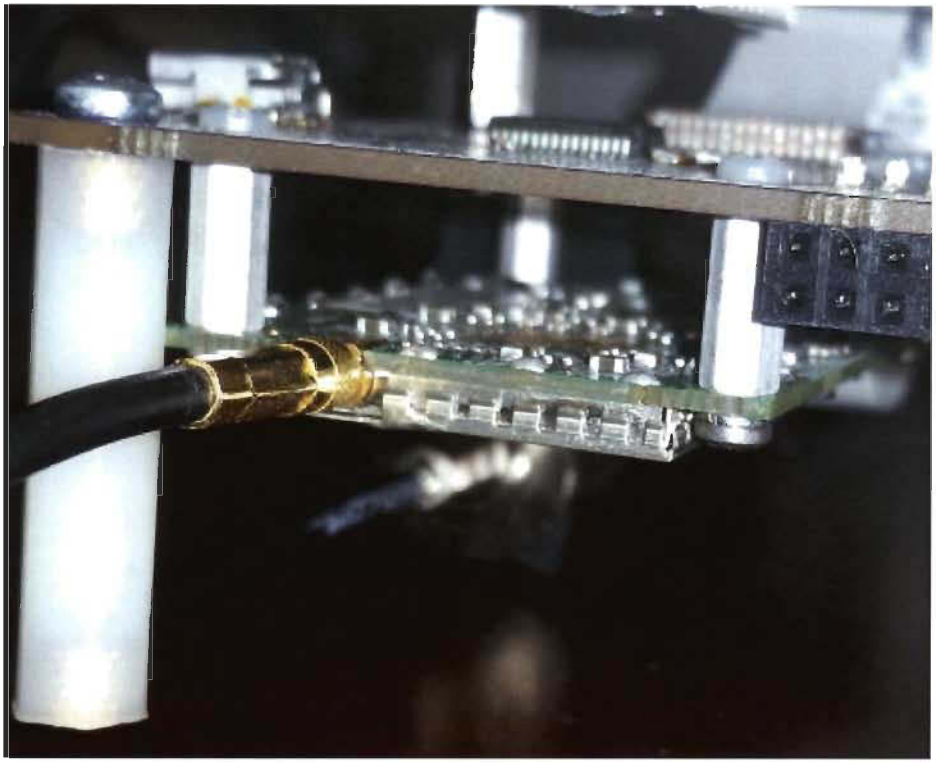

FIGURE 2.2 – Photo du module ThingMagic vue de côté

FIGURE 2.3 – Connecteur du câble utilisé pour le module avec les contacts sur le haut. Image provenant de site de Digikey

2.2 Communication avec le module

Comme indiqué précédemment, la communication avec le module se fait par UART. Le protocole de communication du Thingmagic utilise la forme de transmission que l'on voit à

FIGURE 2.4 – Image du type d'adaptateur utilisé. Image provenant de site de Digikey

la table 2.2. Cette forme pour des communications de l'hôte vers le lecteur commence par un octet d'entête qui est habituellement 255. Celui-ci est suivi d'un octet qui indique le nombre d'octets dans la section Data. Puis, nous avons un octet qui indique la commande envoyée. Ensuite, nous avons les octets de données et finalement le message se termine sur un check-sum CRC-16 calculé sur tous les octets de la commande à l'exception de l'octet d'entête. La communication du lecteur à l'hôte est similaire, mais deux octets de statut sont ajoutés après l'octet de commande.

Header	Data Lenght	Command	Data	CRC-16
1 octet	1 octet	1 octet	0 à N octets	2 octets

TABLE 2.2 – Structure d'un message échangé entre l'hôte et le module

La fonction utilisée pour calculer le CRC provient du document intitulé M5e Family Dev Guide. Elle est appelée CRC_calcCrc8 et elle prend en argument un pointeur vers le registre

du CRC, le polynombre utilisé qui est 0x1021 dans notre cas, ainsi que le l'octet de donnée à ajouter au calcul de CRC. Cette fonction est encapsulée dans CRCcalc qui l'utilise en boucle pour passer à travers un tableau de données d'une longueur définie avec une possibilité d'offset pour sauter par-dessus l'octet header ou un autre message. La fonction de CRC n'est présentement pas utilisée dans le code de la passerelle malgré sa présence dans le fichier ThingMagicReader.c. Par contre, le code du calcul est utile à avoir si jamais un programmeur voulait modifier ou ajouter un message. Ceci nécessiterait de faire un calcul d'un nouveau CRC. Le programmeur pourrait aussi vouloir l'implémenter pour faire la vérification des messages reçus.

2.2.1 Commandes de démarrage

Afin de démarrer le lecteur RFID certaines commandes doivent lui être envoyées. Pour l'usage de la passerelle, cinq commandes de démarrage sont effectuées après l'alimentation de la passerelle. Les commandes sont identifiées par un code unique envoyé lors de l'envoi du message. Le démarrage s'effectue à un baud rate de 9600, il faut donc s'assurer que le baud rate est adéquat avant d'envoyer les commandes.

La première est la commande pour démarrer le firmware du ThingMagic. Elle dit au bootloader du lecteur de partir l'image firmware en mémoire. Si le démarrage s'effectue sans problème, le lecteur renvoie une réponse avec le même code sinon il envoie un code d'erreur. Le temps maximal pour le démarrage du firmware est de 650ms, il est donc important d'attendre que le ThingMagic réponde à la commande avant de procéder à la prochaine étape. Cette commande n'a pas de paramètre modifiable et un exemple est fourni à la table 2.3.

Les trois prochaines commandes (tables 2.4 à 2.6) nous permettent d'envoyer les réglages

Header	Data Lenght	Command	Data	CRC-16
FF	00	04	-	1D 0B

TABLE 2.3 – Commande de démarrage

nécessaires au lecteur. La première est la région utilisée. Ceci permet au ThingMagic d'utiliser les bonnes configurations régionales. Ensuite, nous avons la commande de protocole. Cette commande nous permet de choisir entre les différents protocoles RFID. Étant donné que le compact M5e peut seulement utiliser le protocole GEN2, nous réglons cette configuration à ce protocole. La dernière commande de cette série est pour la configuration de la lecture. On peut choisir une longueur d'EPC soit 496 bits ou 96 bits et on peut choisir si on veut utiliser le module en mode haute performance ou en mode d'économie d'énergie. Les arguments de ces trois commandes n'ont pas besoin d'être modifiés pour ce qui est de la portée du projet.

Header	Data Lenght	Command	Data	CRC-16
FF	01	97	01	4B BC

TABLE 2.4 – Commande pour la sélection de région

Header	Data Lenght	Command	Data	CRC-16
FF	02	93	00 05	51 7D

TABLE 2.5 – Commande pour la sélection de protocole

Header	Data Lenght	Command	Data	CRC-16
FF	02	9A	00 03	C0 52

TABLE 2.6 – Commande pour la configuration de la lecture

Finalement, dans mon code j'utilise la commande pour ajuster le baud rate à 115200 (table 2.7) car au redémarrage, le baud rate du module est réinitialisé à 9600. Il est évidemment

possible d'utiliser d'autres baud rates. Le baud rate est situé dans les quatre octets de données de la commande en hexadécimale du plus haut octet au plus bas. Dans le guide du développeur, il y a une table pour les différents baud rate et leur équivalent hexadécimal pour les insérer facilement dans la commande à envoyer.

Header	Data Lenght	Command	Data	CRC-16
FF	04	06	00 01 C2 00	A4 06

TABLE 2.7 – Commande pour la sélection du baud rate

Une fois que ces cinq commandes sont envoyées, nous sommes prêt à utiliser le module afin de faire de la lecture d'étiquette RFID.

2.2.2 Commandes de lecture

Pour effectuer une lecture d'étiquettes, une chaîne de commandes doit être envoyée comme pour les commandes de démarrage. Quatre commandes sont nécessaires pour le fonctionnement de base de la passerelle. La première commande (table 2.8) est tout simplement d'effectuer une lecture d'étiquettes multiple. Cette fonction qui permet d'identifier de multiples étiquettes en une lecture est un des avantages de ce lecteur RFID. Le timeout utilisé dans la commande est celui fourni dans les exemples.

Header	Data Lenght	Command	Data	CRC-16
FF	04	22	00 01 03 E8	3F 8E

TABLE 2.8 – Commande de lecture multiple

Ensuite, il est important de savoir que le ThingMagic opère avec un tampon interne où il emmagasine les étiquettes après une lecture. Avant d'envoyer les étiquettes reçues à notre

microcontrôleur hôte, une commande (table 2.10) peut nous dire le nombre d'étiquettes présentes dans le tampon. Ceci nous permet de savoir combien d'entrées, dans le tampon nous devons lire. La commande pour le nombre d'étiquettes est similaire à celle pour l'envoi (table 2.9), mais elle n'a pas de paramètre. Celle pour l'envoi contient un paramètre qui indique le nombre d'étiquettes à retourner. Présentement dans la passerelle, le code donne la commande pour lire seulement une étiquette à la fois et répète cette opération jusqu'à ce que le contrôleur ait lu le nombre d'étiquettes présentes dans le module, mais ceci pourrait être changé. Il y a aussi une possibilité de lire le tampon d'un index de la table à l'intérieur du tampon jusqu'à un autre index.

Header	Data Lenght	Command	Data	CRC-16
FF	02	29	00 01	57 E8

TABLE 2.9 – Commande pour obtenir une étiquette du tampon du module

Header	Data Lenght	Command	Data	CRC-16
FF	00	29	-	1D 26

TABLE 2.10 – Commande pour obtenir le nombre d'étiquettes dans le tampon du module

La dernière commande utilisée dans la lecture (table 2.11) est celle pour vider le tampon interne. Dans le code, celle-ci est utilisée au début de chaque lecture. Cette fonction n'a pas de paramètre.

Header	Data Lenght	Command	Data	CRC-16
FF	00	2A	-	1D 25

TABLE 2.11 – Commande pour vider le tampon du module

2.3 Algorithme d'envoi de commande

L'algorithme pour envoyer des commandes consiste de façon brève à envoyer la commande par UART, attendre la réponse et la mettre dans un tampon. Si la réponse est bonne, on peut vider le tampon. Lorsqu'une commande est envoyée par la fonction d'envoi, celle-ci met une variable témoin à vrai et met aussi le numéro de la commande en tampon. Dans le ISR de la réception, si nous avons une commande d'active, la fonction met en tampon chaque octet reçu. Sachant le nombre d'octets de données à partir du deuxième reçu, on peut compter le nombre exact d'octets à recevoir et une fois que la réception du message est terminée on peut mettre notre variable témoin à faux, ce qui nous indique que nous pouvons passer à la prochaine commande. Une représentation visuelle de l'algorithme pour la réception d'un message provenant du module est fournie à la figure 2.5.

2.3.1 Algorithme de démarrage

L'algorithme de démarrage utilise ce qui a été mentionné précédemment pour faire la séquence nécessaire au bon fonctionnement du module. La première étape est de configurer notre UART pour un baud rate de 9600 et de mettre une valeur dans une variable qui nous sert de condition d'arrêt si jamais le démarrage prend trop de temps. Le code envoie ensuite la commande de démarrage firmware, suivi du mode de fonctionnement par rapport à la région, au protocole et à la lecture d'étiquette. Finalement, l'algorithme envoie la commande pour configurer le UART à un baud rate plus élevé et termine en ajustant le baud rate de l'hôte à la même vitesse. Ceci est représenté à la figure 2.6.

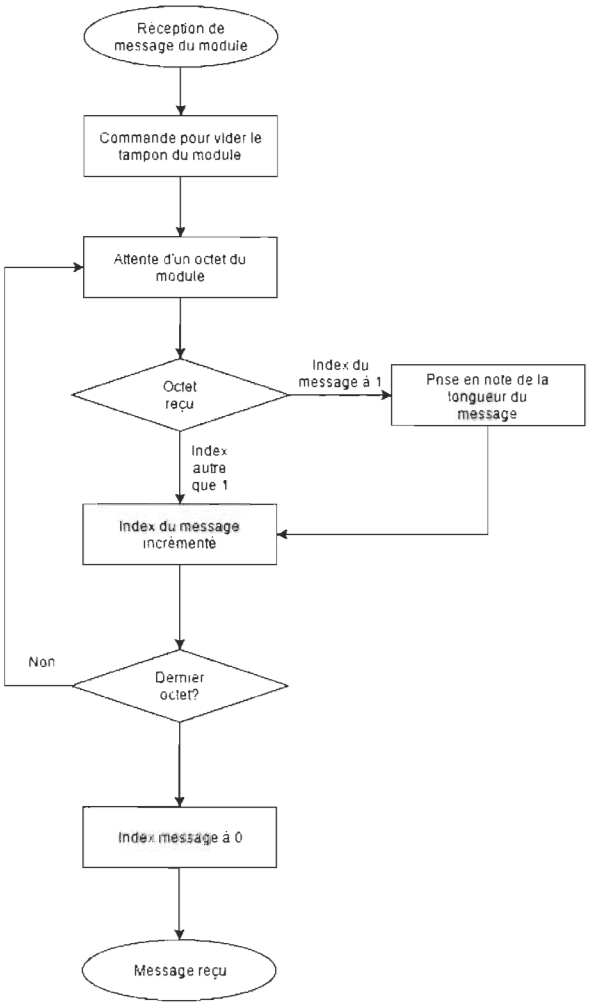

FIGURE 2.5 – Algorithme de réception de message du module

2.3.2 Algorithme de lecture

La séquence de lecture d'étiquette est similaire au démarrage (figure 2.7). Elle utilise aussi l'algorithme d'envoi de commande et une variable compteur comme condition d'arrêt. Un tampon de réception d'étiquette est utilisé pour emmagasiner ceux-ci sur la passerelle. L'algorithme commence par la commande pour vider le tampon interne du lecteur. Ceci est suivi de la commande pour effectuer la lecture d'étiquette RFID. Une fois la lecture terminée, on va chercher par commande le nombre d'étiquettes lues et on vide le tampon de la passerelle. La dernière étape de la séquence est de transférer les données du tampon du lecteur à celui

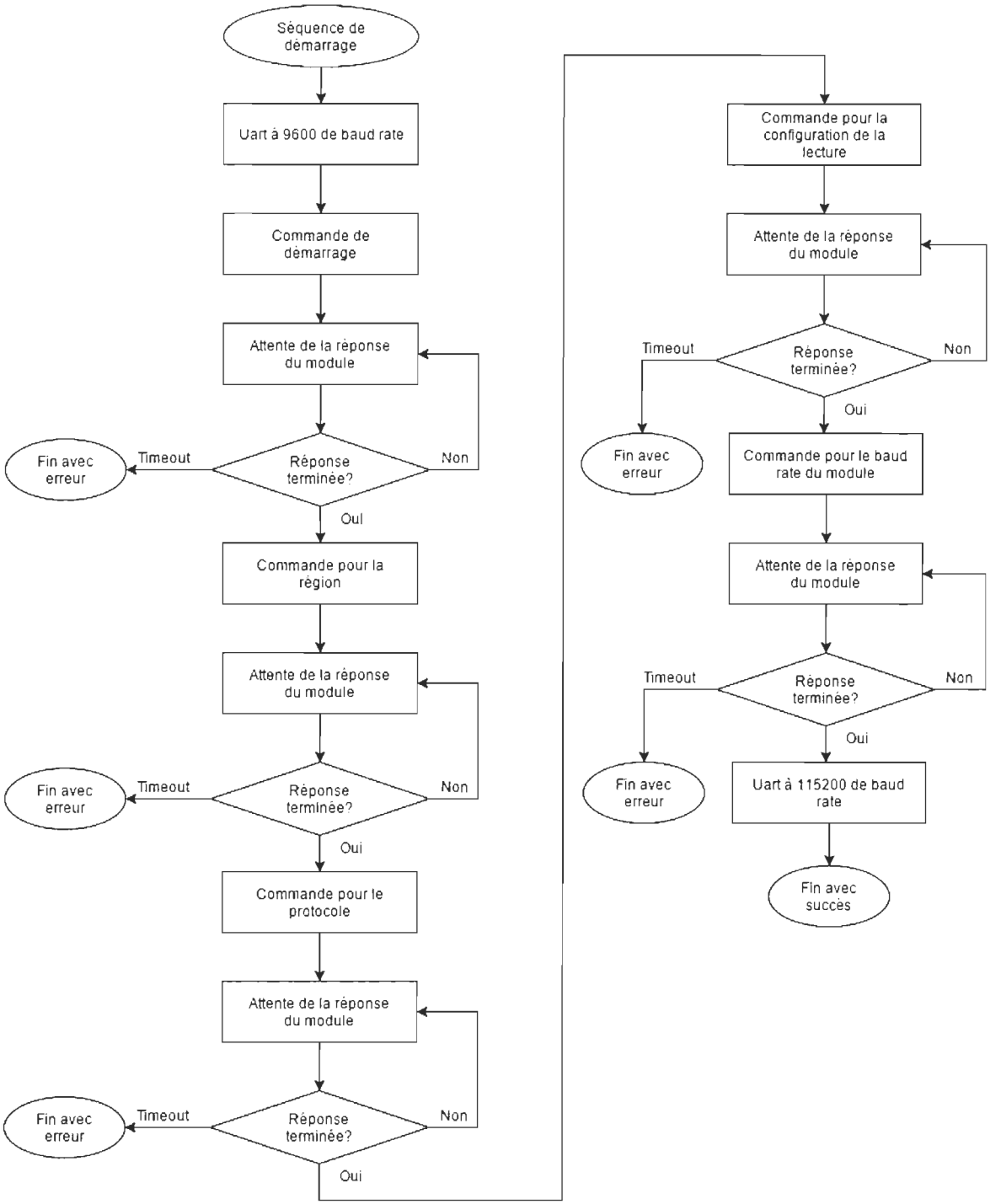

FIGURE 2.6 – Algorithme de démarrage du module RFID

de la passerelle. Cette partie est accomplie en envoyant les étiquettes une à une jusqu'à ce qu'elles soient toutes dans le tampon de la passerelle.

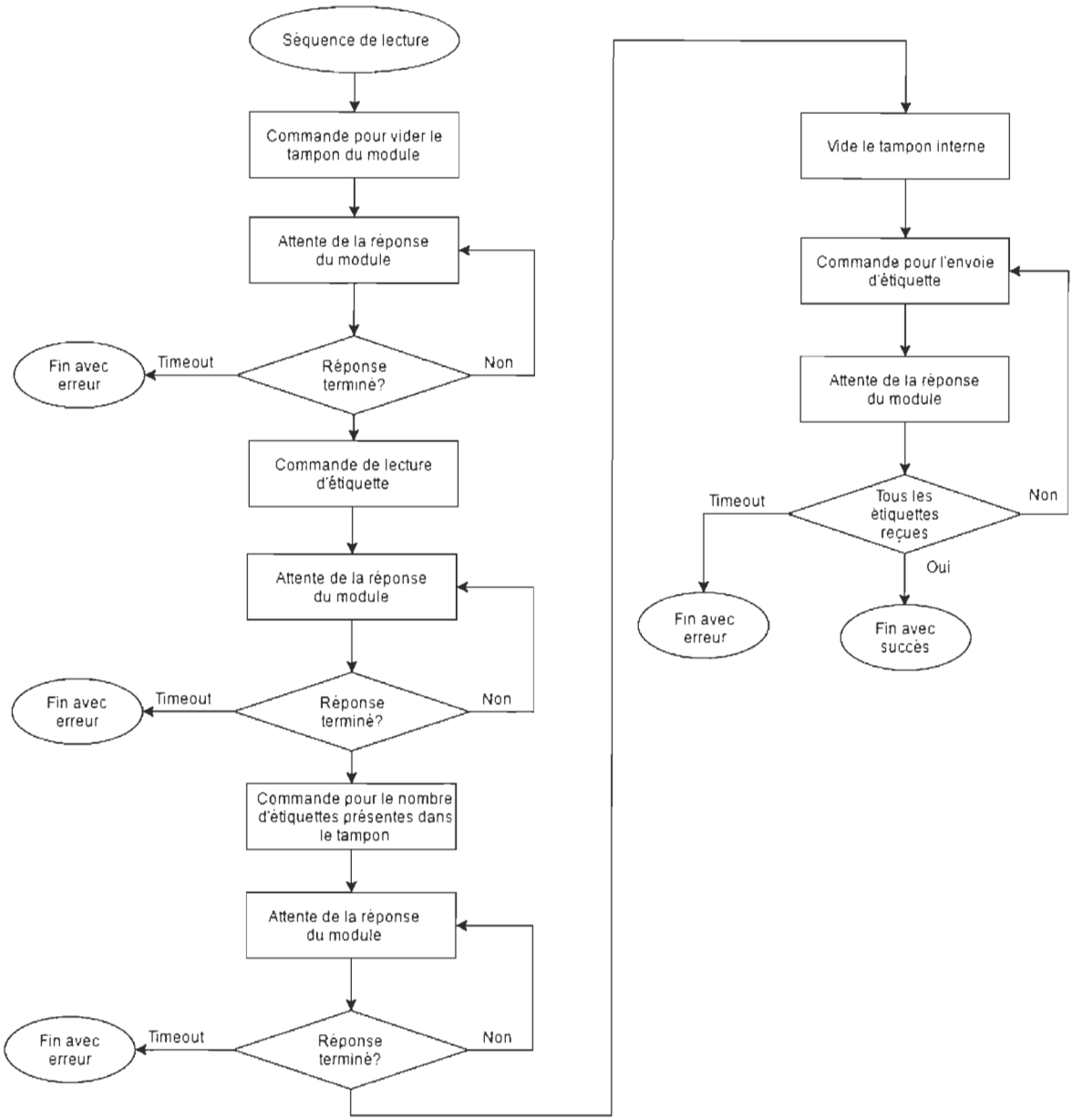

FIGURE 2.7 – Algorithme de la séquence de lecture

2.4 Conclusion

Pour conclure, le guide du développeur et la plate-forme de développement de ThingMagic ont aidé à faire une transition rapide vers une passerelle fonctionnelle. Grâce à la plate-forme de développement, j'ai pu trouver une façon d'intégrer physiquement le module à mon

circuit imprimé. J'ai retrouvé les dimensions et les composants nécessaires pour recréer les connexions et le montage présent sur la plate-forme. Pour ce qui est du code et du fonctionnement du firmware, le guide du développeur de ThingMagic était assez détaillé et les algorithmes créés rendent son opération simple pour quelqu'un qui voudrait partir un projet avec ce module. Il serait extrêmement suggéré à un programmeur voulant utiliser le code et les branchements électriques de ce projet de lire le guide du développeur. Finalement, le lecteur RFID ThingMagic M5e compact est un des deux gros blocs de ce projet et le prochain se situe dans le chapitre suivant.

Chapitre 3 - Module Wi-Fi

3.1 Le module et ses spécifications

Le module Wi-Fi utilisé dans ce projet est le RN171 de Microchip. Plus spécifiquement, le RN171-XV qui est la version montée sur un circuit imprimé avec une antenne filaire que l'on peut voir à la figure 3.1.

FIGURE 3.1 – Image du module RN171 utilisé. Image provenant du site de Digikey

3.1.1 Choix du module

Il y avait quelques choix pour un module Wi-Fi. Initialement le module proposé était le MRF24WG0MA qui était présent sur une plateforme de développement de Digilent. Par contre, ce module n'avait pas la pile TCP/IP d'intégrées et ceci devait donc être intégrée du côté de la passerelle. La solution trouvée pour contourner ce problème fût de se diriger vers le module Wi-Fi RN171 qui lui avait la pile TCP/IP à l'intérieur de son firmware ce qui a pu sauver de la puissance de traitement sur le côté passerelle ainsi que du temps de développement. Il existe d'autres modules Wi-Fi qui pourraient être utilisés à la place du RN171 comme le CC3200 et le WL1837 de Texas Instrument. De plus, il y a le ESP8266 de Espressif. Le choix est tombé sur le RN171 vu sa documentation détaillée et aussi pour rester avec le même fabricant que le module proposé au début du projet, mais les autres options pourraient être explorées.

3.1.2 Spécifications du module

Le RN171 est un module Wi-Fi 2.4 GHz IEEE 802.11b/g de 26.67mm par 17.78 mm. Il a 14 GPIOs et 8 entrées analogues. On utilise des commandes ASCII pour le configurer. Côté réseau, il supporte les modes infrastructure et SoftAp. Il supporte les protocoles TCP, UDP, DFCP, DNS, ICMP, ARP, client HTTP et client FTP. Il a le stack TCP/IP intégré ainsi qu'une adresse MAC unique. Il supporte les protocoles de sécurité WEP-128, WPA-PSK et WPA2-PSK. Pour ce qui est de la consommation de courant, il possède selon la fiche technique un mode veille à 4uA. De plus, en mode réception on a une consommation de 40mA et en mode de transmission un courant de 120mA à 0 dBm. Le module marche à 3.3V à une température entre -40C et 85C. Les caractéristiques RF sont une fréquence d'entre 2.412 et 2.462 Ghz et

une modulation qui a une sensibilité de -83dBm et une sortie de 0 à 12dBm. Finalement, la vitesse de transfert est de 1 à 11 Mbps pour le 802.11b et de 6 à 54 Mps pour le 802.11g. Le module communique par UART à son hôte avec une vitesse de base de 9600 qui peut être augmentée en la configurant.

3.1.3 Intégration au PCB final

Pour intégrer le module au board une version du RN171 montée sur un circuit imprimé avec une antenne filaire et avec des pins de 2mm de pas pour les connexions aux sorties du module a été achetée. On peut voir une image du montage à la figure 3.2 Le nom du module avec le PCB est le RN171XVW-I/RM. Les connecteurs femelles pour accueillir le RN171 ont été trouvés sur Digikey. Le numéro de ces connecteurs est 952-1354-5-ND. Le module est situé sur le côté qui fait face au sol à côté du lecteur RFID et du connecteur d'alimentation comme nous pouvons l'observer à la figure 3.2. L'empreinte sur le PCB est basée sur l'empreinte 802.15.4 qui est utilisée pour plusieurs dispositifs RF comme les Zigbees.

Les pattes reliées au RN171 sont celles pour le UART ainsi que le CTS et le RTS pour que le module ait la possibilité d'opérer à un baud rate au-delà de 115200. Finalement, la patte pour réinitialiser le dispositif est aussi reliée au microcontrôleur.

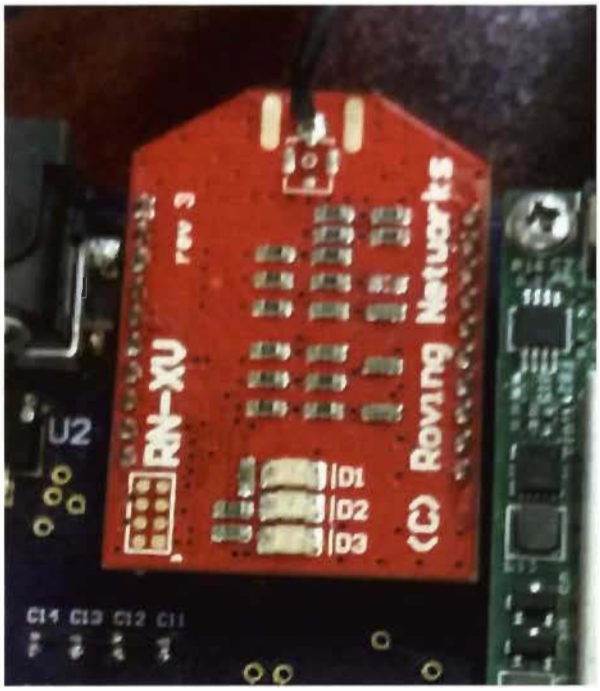

FIGURE 3.2 – RN171 sur la passerelle à côté du module RFID

3.2 Configuration du module

3.2.1 Configurations disponibles

Il y existe quelques manières différentes d'utiliser le RN171. Premièrement, on peut l'utiliser comme point d'accès. Dans ce mode on peut configurer le nom du réseau et son mot de passe. Le module à aussi la capacité d'aller cherché une adresse IP du serveur DHCP d'un routeur. Cette capacité est intégré dans le module et se réalise si l'option est sélectionné. On peut aussi utiliser le RN171 en client FTP. Le module va se connecter au serveur et peut créer et aller chercher des fichiers. Ces fichiers vont être transférés à travers le UART. Un autre mode de configuration est le client HTML qui permet au module de se connecter à un serveur Web et d'inscrire des données sur celui-ci. Finalement, la façon que le RN171 est utilisé dans ce projet est de se connecter à un hôte externe.

Le protocole UDP est utilisé pour envoyer des paquets à un ordinateur à travers un réseau préexistant. Le protocole TCP aurait aussi pu être utilisé, mais l'UDP a été choisi pour sa simplicité. Le désavantage d'utiliser UDP est qu'il serait possible d'avoir de la corruption ou de la perte de paquet, car il n'y a pas de validation de réception durant la communication. Ce genre de problèmes n'a pas été observé en expérimentation, mais la méthodologie de communication définie dans ce chapitre (section 3.3.2) offre la possibilité d'implémenter un CRC afin de vérifier la validité des messages reçus. Le nombre d'octets par message est aussi inclue dans ceux-ci donc, il serait possible d'implémenter un délai d'attente qui déduit que le message n'est pas complet et que certains octets ont été perdus.

Pour accéder au réseau, le module à besoin du nom du réseau et de son mot de passe. La communication requiert donc l'IP de l'hôte et le port utilisé pour le transfert. Une fois que le RN171 est mis en mode UDP, on peut utiliser seulement trois commandes pour le changer de réseau si nous ne changeons pas le port. Une commande pour l'ip de l'hôte, une pour le nom du réseau et une pour le mot de passe. Une autre commande utile est celle qui récupère les détails de la connexion du RN171 à l'hôte. Avec cette commande, nous pouvons savoir si la connexion a été établie entre le réseau et le module.

Les commandes de base utilisées sont écrites dans le fichier header de RN171cmd.c. Les commandes pour sauvegarder les nouvelles configurations sont situées dans leurs fonctions respectives. Pour avoir une référence aux diverses commandes, on peut consulter le manuel de référence des commandes WiFly dont le numéro de document Microchip est 50002230B.

3.2.2 Algorithme pour la configuration à partir d'une interface

Pour mettre le module en mode configuration, on envoie '$$$'. Une fois reçu, le module est prêt à recevoir des commandes. Les commandes sont des phrases ASCII envoyées par le UART. Par exemple, "set wlan ssid test \r\n'est une commande qui met à jour le nom du réseau dans le module à test. Il faut toujours enregistrer les modifications effectuées avec la commande "save\r\n" avant de revenir en mode de communication. Pour sortir du mode de commande, il faut envoyer "exit\r\n".

L'algorithme utilise trois structures tampons pour enregistrer les configurations localement sur le microcontrôleur. Deux de ces structures, présentes dans le fichier RN171_comm.h, contiennent les données qui sont envoyées par le module après les deux commandes pour aller chercher les informations présentes dans le RN171. La première commande nous renvoie l'IP de l'hôte et l'autre le nom et mot de passe du réseau. Par contre, ces commandes envoient aussi d'autres informations sur les configurations du module, donc la troisième structure présente dans DefnType.h est utilisée pour garder seulement les trois configurations énumérées précédemment qui seront modifiées par l'usager.

Lors du démarrage, l'algorithme attend que le module soit opérationnel et ensuite fait une requête pour l'information enregistrer dans le RN171 afin de remplir les tableaux de caractères situés dans leurs structures respectives. Cette séquence est représentée à la figure 3.3. Une fois les données enregistrées, le reste des étapes s'effectue lors d'une requête de l'usager. Pour changer une configuration, l'usager doit entrer dans le menu WiFi. Lors de l'affichage de ce menu, un test de connexion est effectué pour pouvoir montrer à l'utilisateur l'état de celle-ci. Ce qui apparaît à l'utilisateur sont les configurations copiées du tampon du réseau courant. Les nouvelles configurations mises par l'utilisateur ne sont pas mises à jour dans le tampon du réseau courant tant que celui-ci n'a pas appuyé le bouton pour télécharger

les configurations vers le RN171. Lorsque ce bouton est appuyé, l'algorithme envoie les trois commandes pour mettre à jour les configurations au module et il met aussi à jour le tampon des configurations actuelles du côté du microcontrôleur. Ceci est représenté par la figure 3.4.

3.3 Protocole et méthodologie d'envoi utilisé

3.3.1 Protocole utilisé

Le protocole utilisé dans ce projet est l'UDP. C'est un protocole simple, mais rapide. La passerelle se connecte à un hôte UDP sur le même port et elle peut ensuite envoyer ses données. Le logiciel PC agit comme l'hôte UDP et écoute sur le port choisi pour les messages de la passerelle. Le protocole UDP nous permet d'envoyer des octets, mais pour envoyer un message plus complexe, il faut se donner une structure.

3.3.2 Méthodologie de communication

La méthodologie utilisée a été établie après avoir mis en place le code du lecteur RFID. Pour garder de la cohérence entre les deux modules, la structure des messages du lecteur a été répliquée pour le module Wi-Fi. Donc, nous avons un octet de tête suivi d'un octet représentant la longueur du message. Puis, nous avons le code de la commande qui est utilisé et ensuite les octets de données. Le message fini par deux octets pour un CRC. Ce CRC n'a pas été implémenté pour la démonstration de fonctionnement, mais ces deux octets sont réservés pour une possibilité d'ajouter cette fonctionnalité dans le futur.

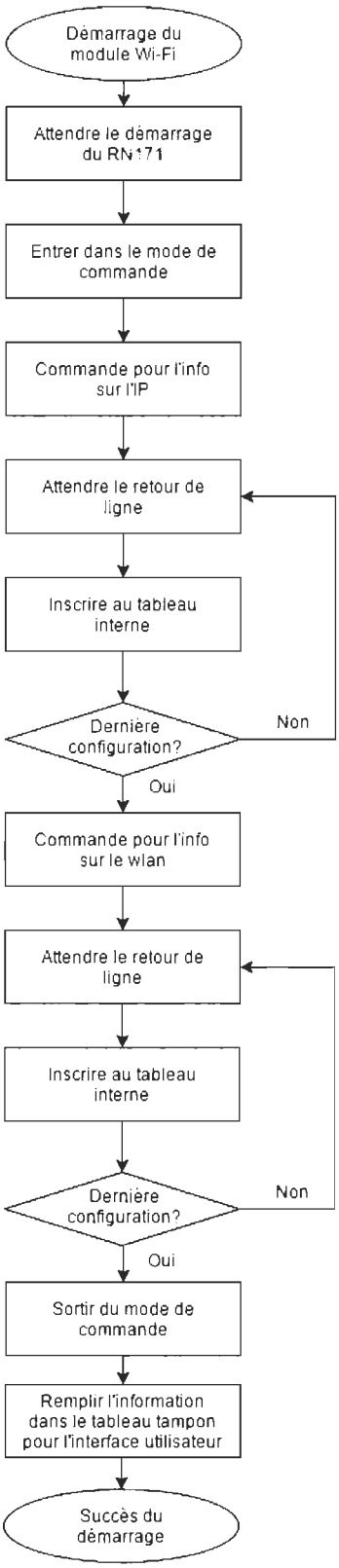

FIGURE 3.3 – Algorithme de démarrage du module Wi-Fi

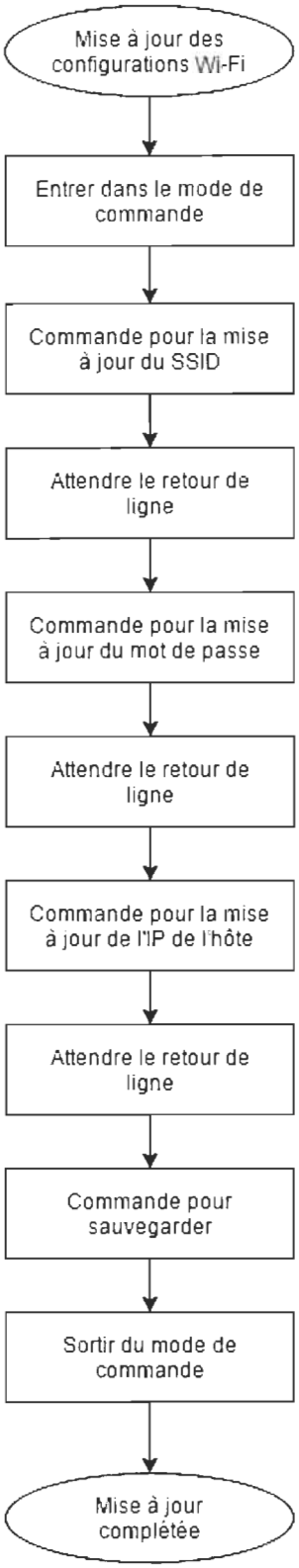

FIGURE 3.4 – Algorithme pour la mise à jour des configurations Wi-Fi

Header	Data Lenght	Command	Data	CRC-16
FF	00	02	-	00 00

TABLE 3.1 – Commande du PC vers la passerelle pour une lecture et envoi

Présentement, la seule implémentation de cette structure est une commande pour recevoir toutes les étiquettes présentes dans le tampon de la passerelle (table 3.1) et une commande pour faire une demande de lecture avant d'envoyer les étiquettes (table 3.2). Une fois que la passerelle a reçu la commande, elle envoie un par un les étiquettes lues en utilisant la structure sur chaque étiquette envoyée. L'octet de longueur pour les messages de la passerelle au PC a été omis.

Header	Data Lenght	Command	Data	CRC-16
FF	03	01	64 octets	2 octets de crc

TABLE 3.2 – Un message de réponse de la passerelle à la commande de la table 3.1

3.3.3 Algorithme d'envoi

Cet algorithme représenté à la figure 3.5, commence par une requête de l'hôte UDP. La passerelle attend de recevoir une des deux commandes implémentées. Lorsque le microcontrôleur reçoit l'octet de tête et que son index de lecteur est à zéro, il se prépare à recevoir un nouveau message. Sur les prochaines réceptions d'octets, l'index incrémente pour savoir où nous sommes rendus dans le message. Lorsqu'on est rendu à la fin du message l'index se remet à zéro et on peut lire la commande envoyée. Si la commande pour la lecture est reçue, l'algorithme d'envoi utilise l'algorithme de lecture vu dans le chapitre précédant (figure 2.7) pour effectuer une lecture avec le module RFID. Une fois la lecture terminée ou si nous avons simplement reçu une commande pour avoir le contenu du tampon, la passerelle envoie les étiquettes RFID qu'elle a en mémoire. Chaque étiquette est envoyée sous la forme d'un

message donc, si nous avons trois étiquettes, nous envoyons trois messages.

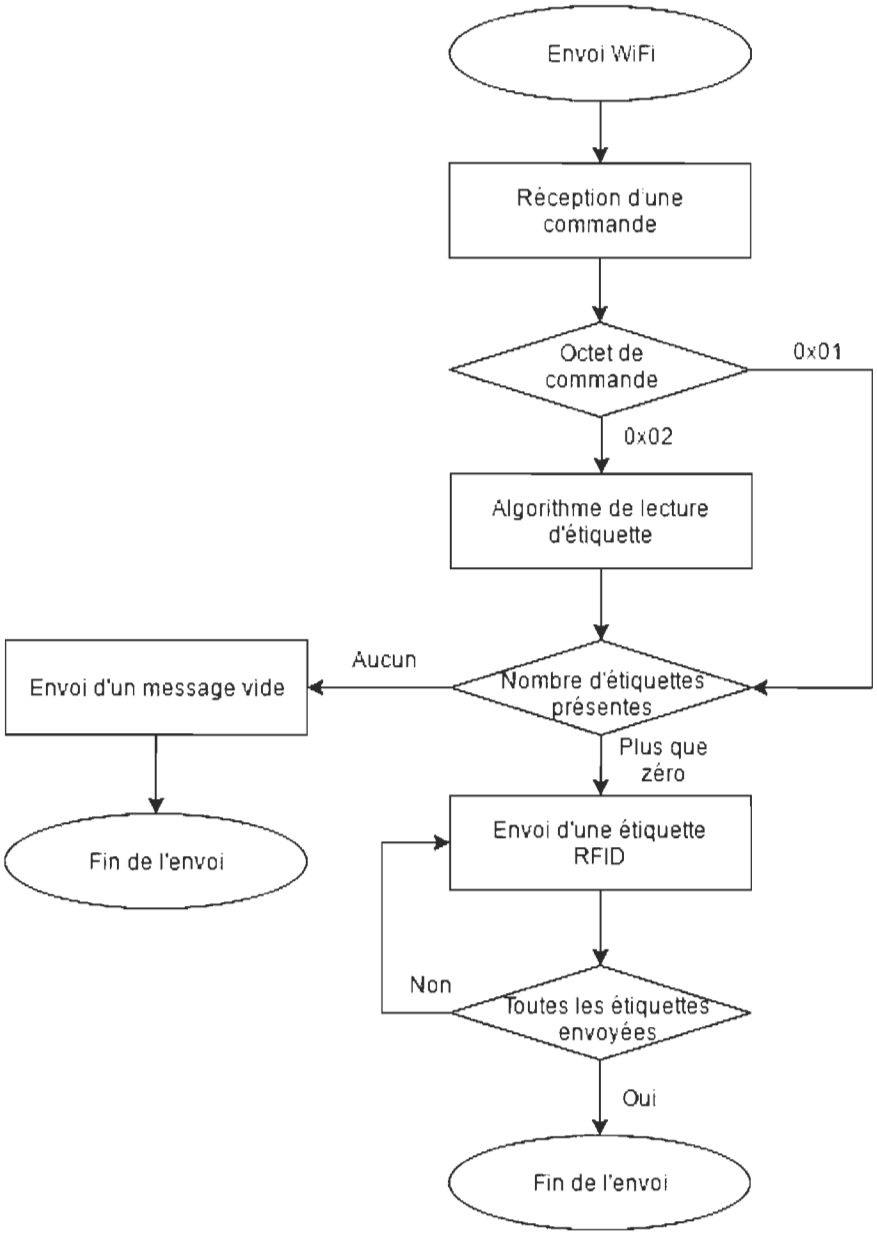

FIGURE 3.5 – Algorithme pour l'envoi par Wi-Fi

3.3.4 Algorithme de réception sur PC

Lorsque l'utilisateur sur le logiciel PC appuie sur le bouton de connexion, l'algorithme commence par prendre l'IP écrit dans l'interface, ouvre un socket UDP et part un fils d'exé-

cution séparé afin de gérer la réception. À chaque 10 secondes, elle fait une requête pour les étiquettes de la passerelle. Nous pouvons aussi demander au logiciel d'effectuer une lecture avec le module RFID au lieu de seulement lire le tampon de la passerelle. Quand le PC reçoit un message, il regarde le nombre total d'étiquettes ainsi que le numéro de l'étiquette reçu. Avec cette information, il ajoute la nouvelle étiquette à son tableau interne et il active une valeur booléenne pour faire une mise à jour de l'interface. L'outil pour afficher les étiquettes est une listview. Pour faire la mise à jour, l'algorithme vide la liste et ajoute les étiquettes en tant qu'item dans la liste. Ceci est représenté à la figure 3.6.

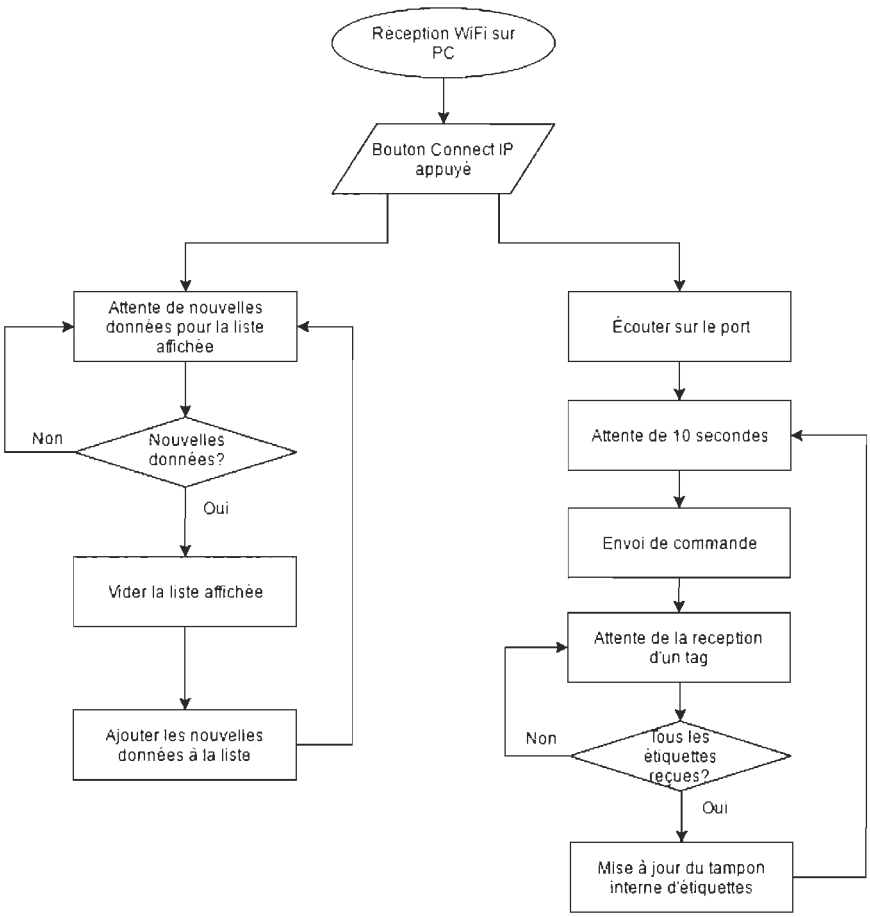

FIGURE 3.6 – Algorithme de réception sur PC

3.4 Conclusion

Afin de résumer, dans ce chapitre, nous avons présenté le module Wi-Fi utilisé et son intégration au projet. Le choix du module est tombé sur le RN171 de Microchip. D'autres modules ont été considérés et nous voyons de plus en plus de dispositifs qui peuvent accomplir ce que le module de Microchip peut faire. Il se pourrait que dans un futur proche un autre module puisse être nettement supérieur à celui utilisé, mais les algorithmes créés lors du projet pourraient être réutilisés avec un autre module. Seulement la configuration devrait changer. Pour ce qui est des configurations présentes dans le module, le projet a utilisé une infrastructure réseau conventionnelle pour relier des messages d'un point à un autre avec le protocole UDP. Le mode client HTTP pourrait être intéressant à regarder dans le contexte d'une mise en oeuvre dans un produit. Finalement, le logiciel sur ordinateur est présentement très simple. Il sert à illustrer le fonctionnement de la passerelle et pourrait être agrandi pour envoyer des informations présentes dans la passerelle. La structure des messages mis en place permet d'ajouter facilement des commandes pour ajouter de la fonctionnalité. Il faudrait seulement prendre en considération les nouvelles commandes dans les algorithmes réception sur PC et ceux d'envoi de la passerelle.

Chapitre 4 - Passerelle RFID à Wi-Fi

Ce chapitre discute des éléments de la passerelle qui ne sont pas reliés aux deux modules utilisés. C'est-à-dire le choix du contrôleur, l'alimentation du PCB, l'interaction avec l'utilisateur et celle avec le développeur ainsi que la fabrication du circuit imprimé. Pour les parties du circuit qui n'entrent pas dans les catégories énumérées, nous avons le connecteur pour la programmation du microcontrôleur ainsi qu'un connecteur à douze places qui ont été connectées sur des GPIOs libres du microcontrôleur pour offrir la possibilité d'ajouter des périphériques externes à la passerelle ou d'effectuer des tests de fonctionnalité du microcontrôleur s'il y a des problèmes avec la communication UART.

4.1 Choix de technologie

4.1.1 Choix de contrôleur

Au début du projet deux choix de type de contrôleur ont fait surface. Le premier était un contrôleur dans le style du Raspberry Pi (figure 4.1). Le Pi est un petit ordinateur embarqué qui supporte une variété de distributions Linux ainsi que Windows 10 IoT Core. Comme module Wi-Fi, un dongle Wi-Fi aurait suffi pour rendre le Pi compatible avec ce standard. Les

avantages de cette solution sont, qu'il serait possible, de développer un logiciel plus élaboré sur celui-ci à cause de sa puissance et de sa mémoire interne. Par contre, il y a plusieurs fonctionnalités du Pi qui ne seraient pas utilisés et dans le cas où nous voudrions réduire les coûts de la passerelle, les microcontrôleurs nous offrent plus de flexibilité dans les périphériques internes disponibles. De plus, ils nous offrent aussi plus d'options pour les périphériques externes que nous voulons utiliser, car nous sommes responsables du design du PCB. Bien sûr, nous avons moins de puissance de calcul sur un microcontrôleur, mais cette puissance risque de ne pas être exploitée à son plein potentiel.

FIGURE 4.1 – Image d'un Raspberry Pi. Image provenant de Wikipédia

4.1.2 Choix de microcontrôleur

Un choix de microcontrôleur se fait en deux parties. La première est de choisir un manufacturier. Il existe plusieurs manufacturiers tels que ST microelectronics, Texas Instrument, Microchip, Atmel, NXP Semiconductors, etc. La majorité des microcontrôleurs actuelle utilisent le processeur ARM dans leur circuit électronique. Microchip est une des rares exceptions à cette tendance. Un des avantages d'utiliser leurs produits est leurs fiches techniques et

exemples d'applications bien documentés et facilement accessibles. De plus, l'expérience que j'ai acquise lors de divers projets avec leurs produits a rendu le développement plus rapide.

Pour ce qui est du choix du modèle de microcontrôleur, nous avons choisi un des nouveaux modèles. Le PIC32MZ est dans les produits les plus récents de Microchip et un des plus puissants pour sa catégorie. Nous avons préféré utiliser un microcontrôleur puissant pour créer la plate-forme de développement afin de permettre la création de fonctionnalités diverses sans trop de limitations matérielles. Ces fonctionnalités pourraient toujours être enlevées si elles étaient jugées non essentielles afin d'optimiser les coûts du produit créer à partir de la passerelle.

4.2 Conception et fabrication du PCB

Le PCB dans ce projet a été conçu sur Altium et envoyé chez OSHpark pour sa fabrication. Le PCB à cinq sous-systèmes. Il y a l'alimentation qui est le premier point de cette section. Ensuite, il y a les modules de communication qui ont été discutés lors des deux derniers chapitres. Finalement, il y a l'interface humaine qui comprend un écran tactile graphique ainsi qu'une carte SD qui sert présentement au stockage d'image pour la construction des menus.

4.2.1 Alimentation

Il y a deux sources d'alimentation pour la passerelle. Par contre, à cause du courant nécessaire pour alimenter le lecteur RFID seulement l'alimentation de la prise murale est utilisée pour ce module. L'alimentation du module RFID se fait avec un adaptateur mural de Triad Ma-

gnetics qui fournit jusqu'à 1.5A à 5V. Ceci couvre nos demandes de courant pour le système au complet. L'autre source d'alimentation est la prise USB utilisée pour la communication avec le PC lors du débogage. Les ports USB 2.0 d'un ordinateur fournissent une tension de $5^{+0.25}_{-0.60}V$ et les ports USB 3.0 fournissent une tension de $5^{+0.25}_{-0.55}V$. Une alimentation USB provenant d'un ordinateur en général jusqu'à 500mA à un périphérique. Bien sûr, si un chargeur de téléphone USB est utilisé ce courant est plus élevé, mais c'est à cause du courant fourni par les ordinateurs que l'alimentation USB n'est pas utilisée pour le ThingMagic.

Le 5V mural et l'USB passent chacun dans une diode de protection afin que dans l'éventualité qu'une des deux sources ait un court-circuit l'autre ne soit pas affectée. La chute de tension dans la diode est de 310mV à 1A. Cette chute ne nous affecte pas, car après la diode la tension est ajustée à 3.3V pour le reste des composantes du circuit. Le régulateur utilisé est linéaire et a un dropout de 0.19V à 1A ce qui ne sera pas un problème, car la tension la plus basse attendue est le 5V de l'USB 2.0 qui peut théoriquement chuter à 4.45V. Finalement, le circuit du régulateur est basé sur les recommandations de la fiche technique.

4.2.2 Communication externe pour débogage

Pour aider au débogage du circuit, un port UART a été utilisé afin d'envoyer de l'information au PC du programmeur. Le UART du microcontrôleur passe par un FT232 qui agit comme passerelle entre le UART et l'USB. Deux LEDs ont été ajoutés sur les sorties du FT232 pour indiquer le fonctionnement de la communication, sinon le circuit est basé sur les recommandations de la fiche technique. Un port mini USB est utilisé pour se connecter aux entrées du périphérique. Le FT232 apparaît sur l'ordinateur comme un port sériel et il est relativement simple de se faire un programme rapide en C# afin de lire les données envoyées ou encore d'envoyer des instructions au microcontrôleur.

Cet outil a été extrêmement utile lors du développement de la plate-forme. Par exemple, pour tester les commandes du ThingMagic, mon code attendait de recevoir un caractère spécifique à travers le UART de dégoguague. Une fois le caractère reçu, il envoyait la commander. Dès que le contrôleur recevait l'information du module il la transférait à l'ordinateur pour que le développeur confirme la validité de l'information reçue. Cette méthode était applicable à tous les systèmes de la passerelle une fois que la communication entre le microcontrôleur et l'ordinateur était fonctionnelle.

4.2.3 Interface humaine

La passerelle offre aussi une interface humaine afin de modifier les configurations du module et aussi dans le but de donner de l'information visuelle sur l'état du programme. L'écran (figure 4.2) est un TFT LCD de 3.2 pouces avec une résolution de 320 par 240 pixels. Cette interface est un circuit imprimé qui inclut un écran LCD sous un écran tactile, un connecteur de carte SD et un contrôleur d'écran LCD. Le tout est sorti sur un connecteur mâle de 40 positions. Le contrôleur LCD prend une entrée parallèle de seize bits. Les entrées sont connectées de façon à utiliser le PORT B des GPIOs du microcontrôleur au complet afin de simplifier le code. L'écran tactile ainsi que la carte SD communique par SPI et sont branchés sur des pattes capables d'utiliser les périphériques SPI du microcontrôleur, car les périphériques du PIC32MZ ne sont pas interchangeables sur toutes les pattes de la chip. Une des positions sur le connecteur de l'écran contrôle le rétroéclairage de l'écran. Elle est reliée à un MOSFet PNP qui permet au microcontrôleur de fermer l'éclairage utilisant le firmware. Ce MOSFet a un seuil de grille de 1V à 250 uA ce qui nous convient pour pouvoir contrôler son état avec une sortie de microcontrôleur.

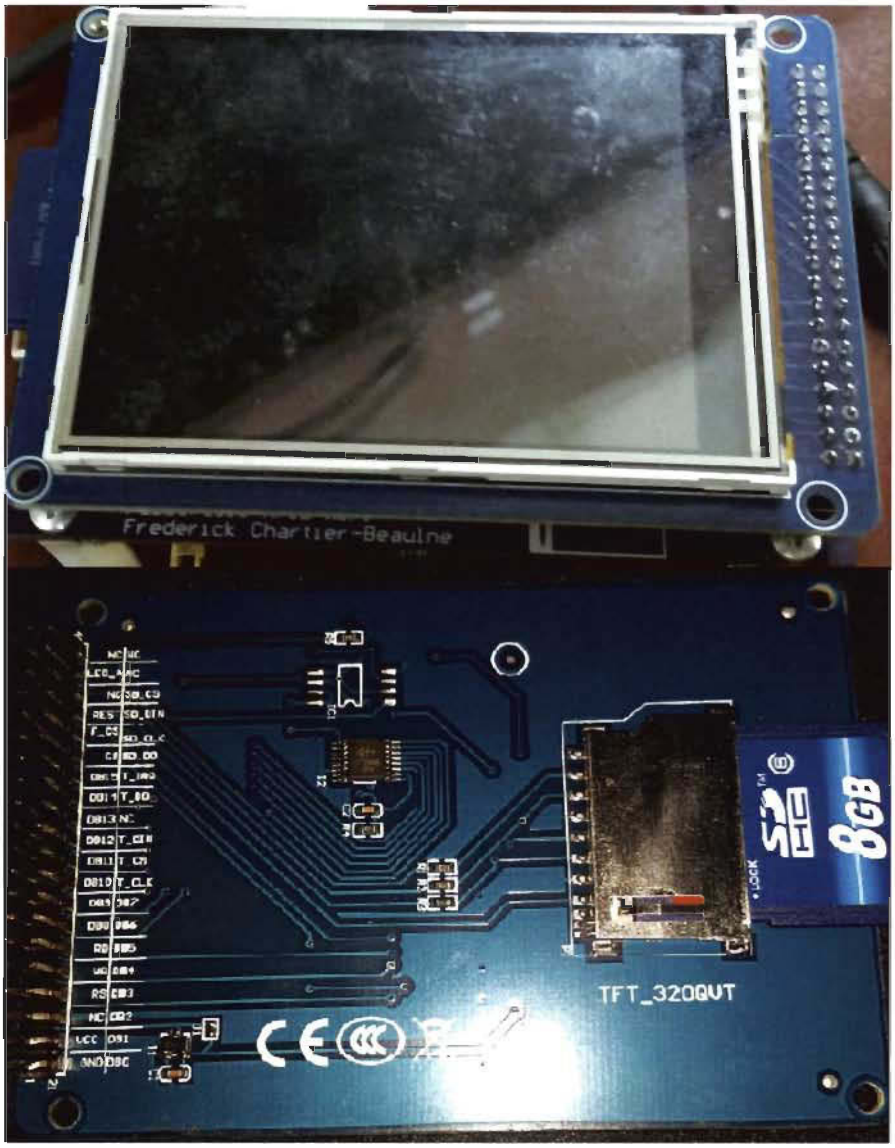

FIGURE 4.2 – Écran utilisé

4.2.4 Fabrication du PCB

Le PCB deux couches a été fabriqué chez OSHpark au prix de 5$ du pouce carré. Ce qui a donné un prix de 50.55 $. Le circuit imprimé est arrivé en trois exemplaires et a été assemblé au laboratoire. L'attente pour la réception du produit a été d'environ 3 semaines. Trois entretoises de 32mm supportent le circuit et laissent assez de place pour monter les modules Wi-Fi et RFID. La quatrième entretoise est un support fabriqué avec l'imprimante

3D de l'UQTR. Ce support a été conçu dans l'objectif d'insérer le connecteur de l'antenne du module RFID. Excluant les modules, il y a pour environ 80-90$ de pièces mise sur le PCB. Les figures 4.3 à 4.5 montrent la passerelle assemblée. La table 4.1 montre les pièces commandées sur digikey excluant le module WiFi, le module RFID, l'écran et les composantes passives que le laboratoire avait déjà en inventaire.

FIGURE 4.3 – La passerelle assemblé vue du haut

4.3 Interface Humaine sur la passerelle

L'interface humaine sur la passerelle sert présentement à donner une indication visuelle de son bon fonctionnement et aussi à changer les paramètres concernant le réseau Wi-Fi. Pour afficher des menus, des images bitmaps sont stockées dans une carte SD. Ces images sont lues lors d'un changement de menu ou de certaines actions utilisateurs. Pour accomplir ceci, du code pour faire l'initialisation de la carte SD ainsi que pour la lecture de celle-ci a été écrit. La carte mémoire est formatée en FAT32 et donc les algorithmes de lecture et écriture ont été

Part Number	Quantité	Descriptif
237-1417-ND	1	AC/DC WALL MOUNT ADAPTER 5V 8W
36-1797D-ND	4	HEX STANDOFF 2-56 ALUMINUM 3/8"
486-2443-ND	3	HEX STANDOFF M3 NYLON 32MM
952-1487-ND	1	HEX STANDOFF M3 BRASS 12MM
S6104-ND	1	CONN HEADER FEM 40POS .1" DL TIN
36-24338-ND	3	HEX STANDOFF M3 ALUMINUM 12MM
952-1354-5-ND	2	CONN RECEPT 2MM VERT AU 10POS
ZXMP3A16GCT-ND	1	MOSFET P-CH 30V 4.6A SOT223
609-1900-1-ND	1	CONN FFC BOTTOM 12POS 1.00MM R/A
S5520-ND	1	CONN FEMALE 12POS DL .1" R/A TIN
CP-002AHPJCT-ND	1	CONN PWR JACK 2X5.5MM SOLDER
PIC32MZ2048ECH100-I/PT-ND	1	IC MCU 32BIT 2MB FLASH 100TQFP
952-2247-ND	1	SIL HORIZ PC TAIL PIN HEADER
160-1183-1-ND	1	LED GREEN CLEAR 0603 SMD
160-1181-1-ND	1	LED RED CLEAR 0603 SMD
768-1007-1-ND	1	IC USB FS SERIAL UART 28-SSOP
ADP3338AKCZ-3.3RL7CT-ND	1	IC REG LDO 3.3V 1A SOT223-3
WM17116CT-ND	1	CONN RECEPT MINIUSB R/A 5POS SMD
SS14-E3/61TGICT-ND	1	DIODE SCHOTTKY 40V 1A DO214AC
445-5932-1-ND	1	CAP CER 0.1UF 50V X7R 0402

TABLE 4.1 – Table des pièces commandées sur Digikey

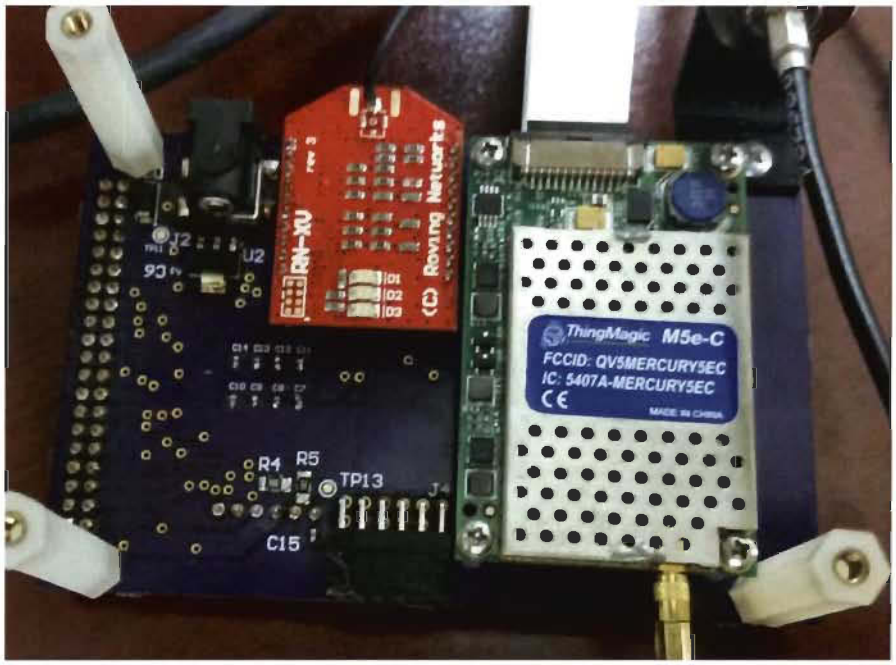

FIGURE 4.4 – La passerelle assemblée vue du bas

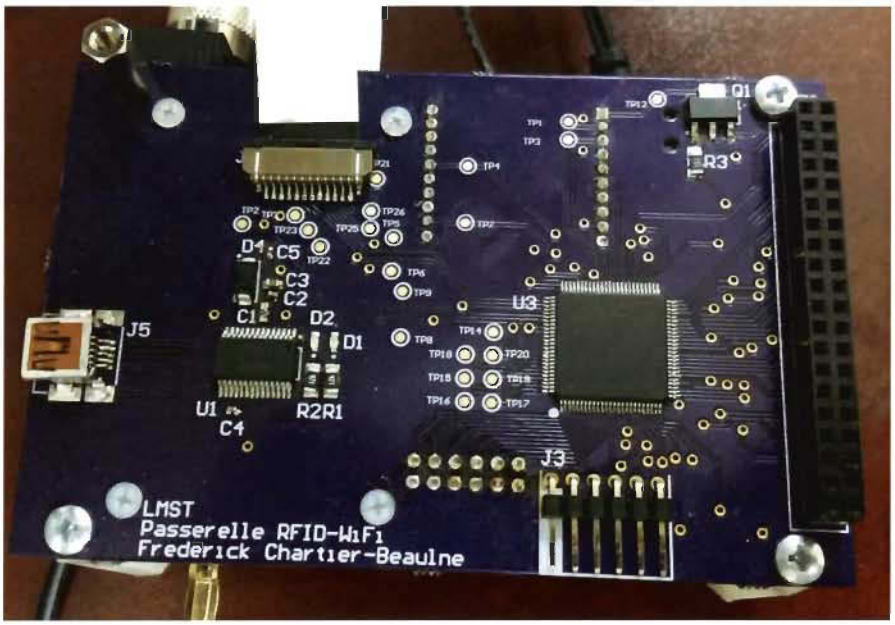

FIGURE 4.5 – La passerelle sans l'écran

développés en prévision de ce format. Finalement. l'utilisateur utilise l'écran tactile afin de choisir les options présentées par l'interface.

4.3.1 Menus

Le programme est divisé en quatre menus distincts. Le premier est le menu principal (figure 4.6) où la passerelle opère de façon normale. Le deuxième menu est le menu de configuration Wi-Fi ou on peut trouver et modifier le nom et mot de passe réseau ainsi l'IP de l'hôte. On peut aussi voir l'état de la connexion avec le réseau et l'IP de la passerelle si présent. Les deux autres menus sont des menus claviers. Il y a un clavier pour l'écriture des noms de réseau et de mot de passe. Puis, il y a un clavier numérique pour l'écriture de l'IP de l'hôte. Ces claviers fonctionnent tous avec l'écran tactile et sont des images prises des claviers d'appareils cellulaires.

FIGURE 4.6 – Menu Principal

Dans le menu principal, on voit si la passerelle a bien effectué son démarrage. Nous avons

un bouton pour accéder aux configurations Wi-Fi ainsi qu'un bouton qui sert à réinitialiser le module Wi-Fi. Le menu a beaucoup de place pour ajouter des fonctionnalités si nécessaire. C'est dans ce menu que le code qui autorise une lecture et un envoi vers le logiciel externe est présent. Si une demande est en attente de traitement et que la passerelle n'est pas en cours de lecture, nous traitons la demande. C'est aussi dans ce menu que l'on trouve l'algorithme pour le démarrage du module RFID. L'algorithme est utilisé la première fois que le menu apparaît après un démarrage sans problème de la passerelle.

À la figure 4.7, nous avons une représentation de l'algorithme du menu principal. Comme on peut le constater, l'algorithme fini seulement que lorsque nous appuyons sur le menu Wi-Fi. Les algorithmes de menu sont le corps de la boucle du microcontrôleur.

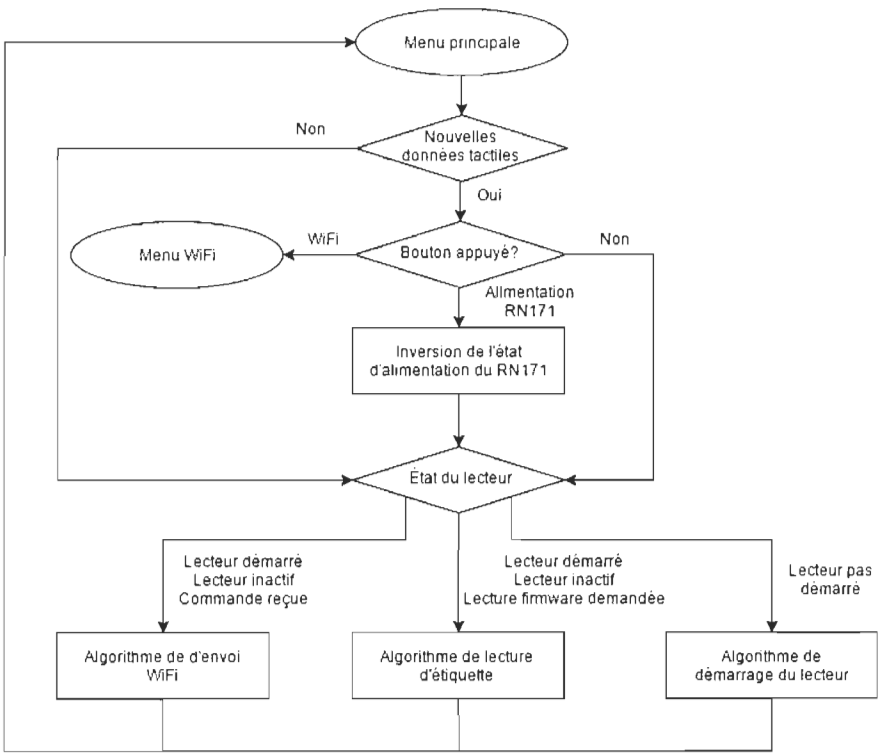

FIGURE 4.7 – Algorithme du menu principal

Le menu WiFi (figure 4.8) est composé de trois boutons et de trois boîtes de texte. Nous avons un bouton qui permet de fermer le menu, un bouton qui permet d'envoyer les données

au module Wi-Fi et un bouton qui permet de rafraîchir l'IP de la passerelle. Le bouton d'envoi utilise l'algorithme dédié à cette tâche expliqué dans le chapitre trois. Pour le rafraîchissement, le module fait un test de connexion et si elle est fonctionnelle, il affiche l'IP du module. Sur ce menu nous pouvons aussi appuyer sur les trois boîtes de texte pour modifier l'IP d'hôte, le nom du réseau ou son mot de passe. Cette action nous amène au menu clavier approprié. Il est important de noter que si l'utilisateur veut appliquer les modifications apportées aux boîtes de texte, il doit appuyer sur le bouton pour les envoyer au RN171. La figure 4.9 montre la représentation de l'algorithme du menu WiFi.

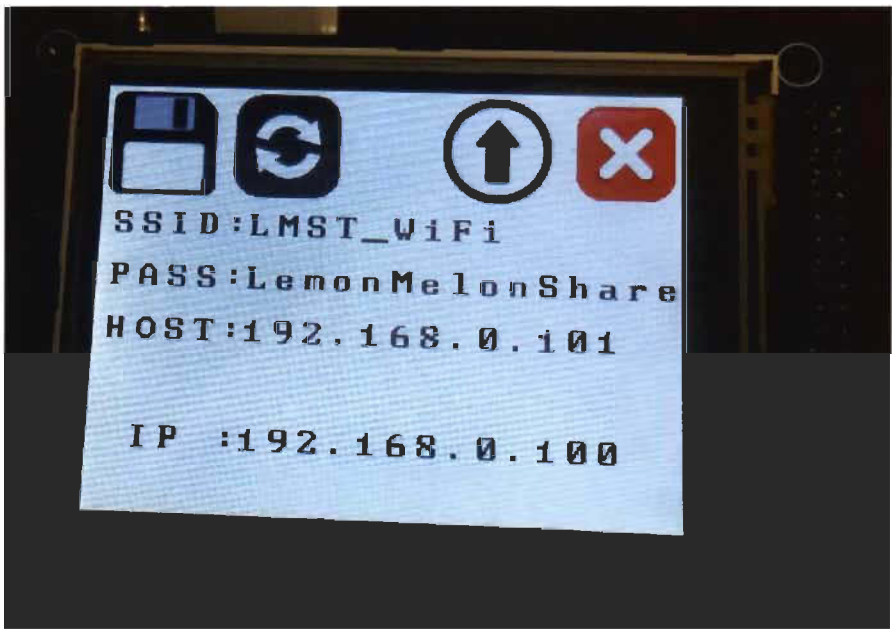

FIGURE 4.8 – Menu pour la configuration du Wi-Fi

Les menus claviers (figures 4.10 et 4.11) sont assez simples, ils prennent la chaîne de caractères et sa longueur et l'affiche à l'utilisateur. Tous les emplacements de touche sont en note dans le code et ils fonctionnent comme un clavier normal. Le clavier standard a deux versions, une normale et une autre pour les caractères spéciaux.

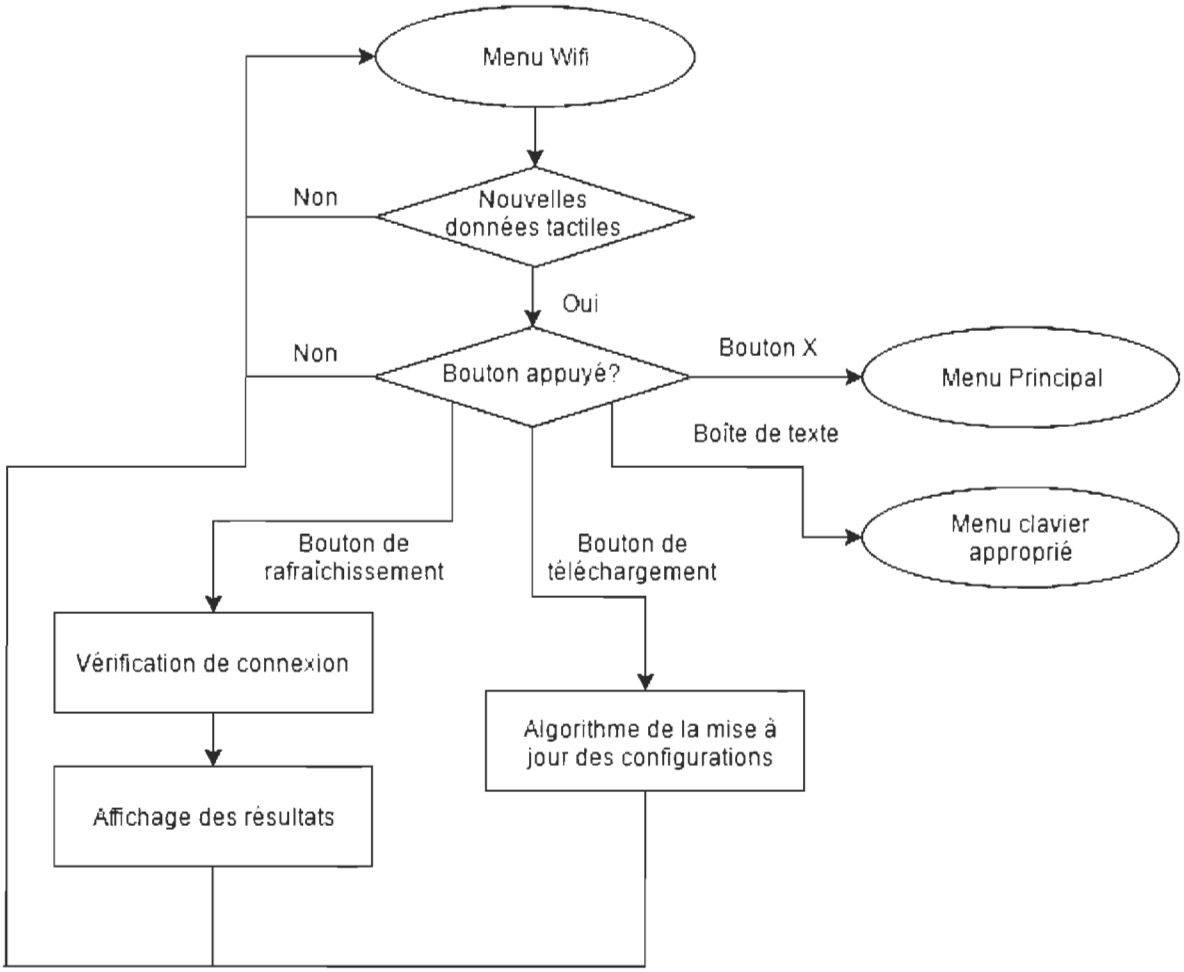

FIGURE 4.9 – Algorithme du menu Wi-Fi

4.3.2 Algorithme d'initialisation de la carte SD

La carte SD dans le projet est utilisé pour sauvegarder les images présentes sur le menu. Pour avoir accès à la mémoire sur la carte, il faut d'abord l'initialiser. Il y a plusieurs étapes à suivre pour faire ceci. La première étape de l'initialisation est d'ajuster la vitesse du SPI à 400kHz. Ensuite, nous devons mettre la patte de chip select au niveau haut et envoyer au moins 74 coups d'horloge. Dans le code du projet, 80 coups d'horloge sont envoyés. Puis, on donne un niveau bas au CS et on envoie la commande CMD0. Une représentation visuelle est fournie à la figure 4.12.

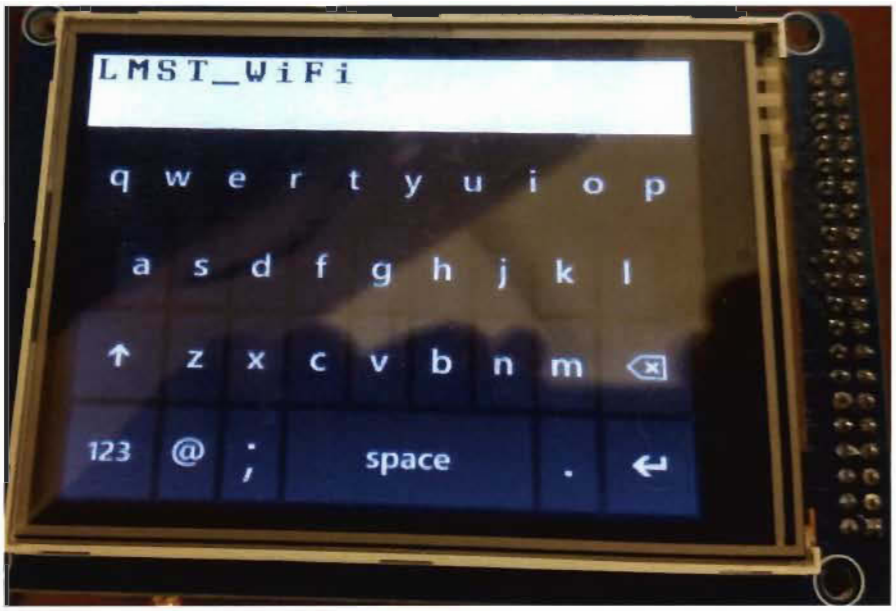

FIGURE 4.10 – Clavier standard

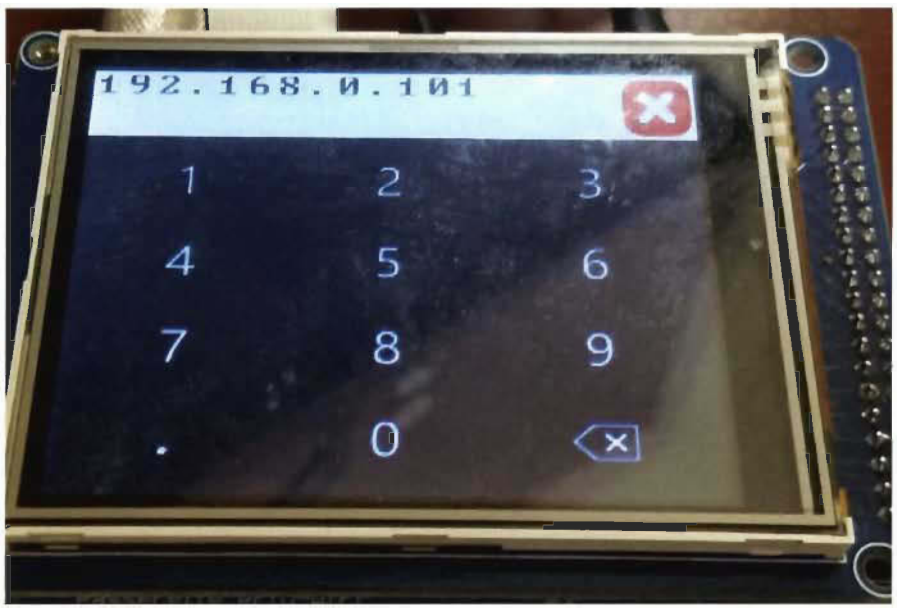

FIGURE 4.11 – Clavier numérique

Les commandes des cartes SD sont toujours de six octets. Dans le premier on retrouve le numéro de la commande. Plus spécifiquement dans les six bits bas de l'octet. Nous avons 01 dans les deux bits hauts. Les quatre prochains octets sont utilisés pour envoyer les arguments nécessaires pour les commandes. Le dernier octet est utilisé pour le CRC. Par contre, une fois initialisé en SPI, le CRC n'est pas utilisé et donc nous pouvons envoyer un octet vide. Le

CMD	Abréviation	Descriptif
CMD0	GO_IDLE_STATE	Change le mode d'opération du mode SD en mode SPI
CMD17	READ_SINGLE_BLOCK	Lie un bloc mémoire (512 octets pour une SDHC)
CMD24	WRITE_BLOCK	Écrie un bloc mémoire (512 octets pour une SDHC)
ACMD41	SD_APP_OP_COND	Utilisé pour démarré la couche applicative de la carte
CMD55	APP_CMD	Utilisé pour démarré la couche applicative de la carte
CMD58	SEND_IF_COND	Renvoie le type de carte (SDSC ou SDHC)

TABLE 4.2 – Commandes de carte SD utilisées

détail des commandes peut être trouvé sur le site de la SD Card Association dans le document sur les spécifications de la couche physique.

Les commandes utilisées sont disponibles à la table 4.2. Une séquence d'initialisation consiste à envoyer une série de commande. La commande CMD0 réinitialise la carte et la met en état de repos. Ensuite la commande CMD58 est envoyée pour vérifier le type de carte (étape optionnel). Cette commande retourne aussi de l'information sur le voltage de la carte. Une fois la réponse reçue, la commande CMD55 et la commande ACMD41 sont envoyées en boucle jusqu'à ce que le bit de repos soit zéro. Ceci termine l'initialisation de la carte et nous pouvons accélérer la vitesse de communication au niveau que nous voulons. Dans le projet le SPI opère à une vitesse de 12.5MHz ce qui est la vitesse maximale que la communication peut atteindre avant d'avoir des problèmes.

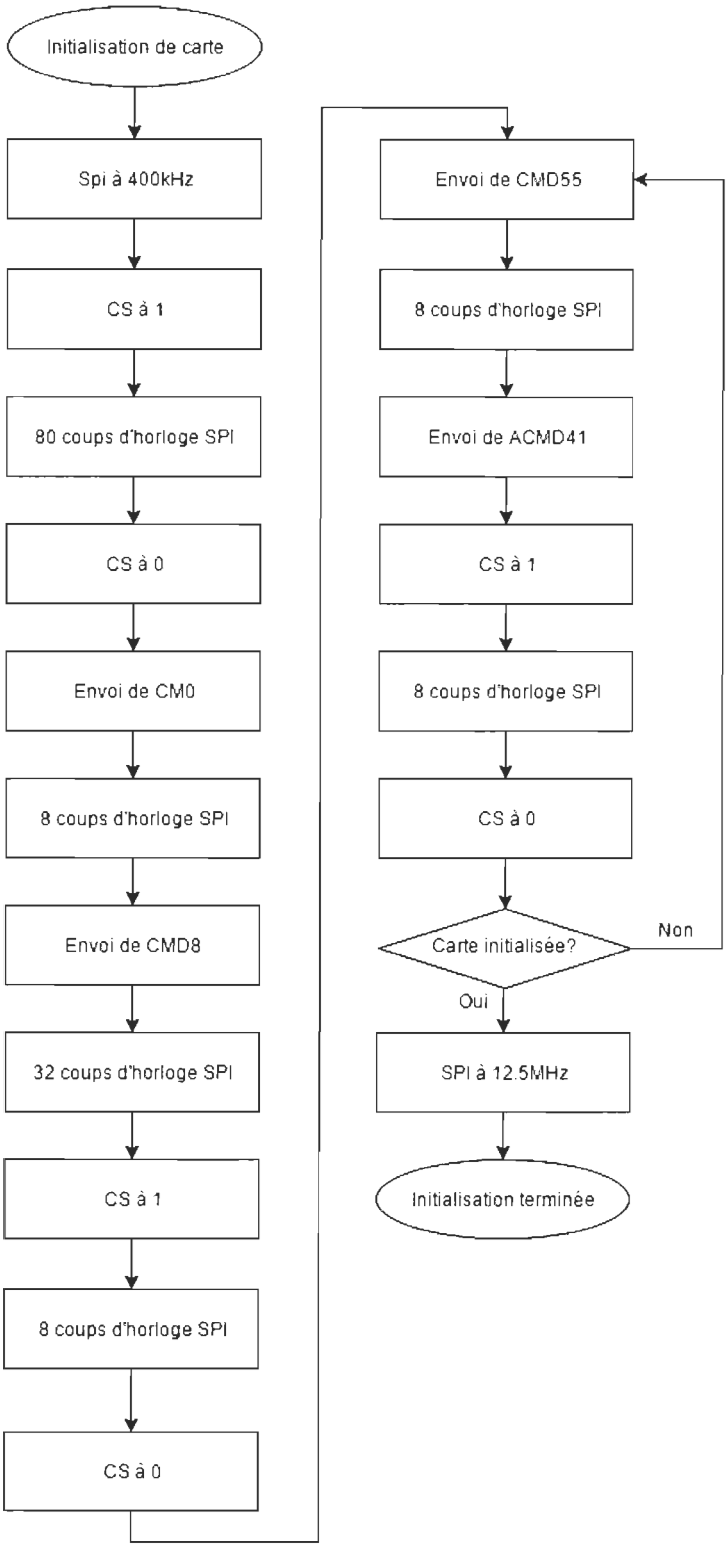

FIGURE 4.12 – Algorithme de l'initialisation de la carte

4.3.3 Le FAT32

Le FAT32 est un système de fichier pour organiser la mémoire morte d'un périphérique qui a pour but de stocker de l'information. Ce système organise la distribution des pages de mémoires d'un fichier. Les pages ou secteurs de mémoire sont un bloc de mémoire continue de longueur fixe. Dans notre cas, la carte a des secteurs fixes de 512 octets. On peut aussi voir le terme cluster quand on fait des recherches sur le FAT32. Un cluster est simplement un regroupement de pages. Le nombre de secteurs par cluster varie selon le formatage et il est important de savoir combien de secteurs par cluster il y a pour faire une lecture précise, mais ce chiffre peut être retrouvé dans le boot sector de la partition utilisée de la carte. Ce secteur peut être trouvé en vérifiant le premier secteur de la carte qui est souvent nommé le master boot sector ou encore le master boot record.

La seule information que nous cherchons réellement dans le MBR est le secteur ou la première partition commence. On peut trouver cette information de l'offset 454 à 457. Avec ces quatre octets, nous pouvons reconstruire le numéro du secteur où se trouve le boot record de la partition. Le projet assume que la carte a seulement une partition pour simplifier le code. Pour un exemple de MBR, allez voir la référence [5] de la bibliographie.

Le boot record de partition (exemple à la figure 4.13) varie un peu en fonction du FAT utilisé, mais nous allons détailler ces points importants pour le FAT32 seulement, car c'est le format utilisé dans le projet. Ces points sont le nombre de secteurs par cluster à l'offset 13, le nombre de FAT à l'offset 16, le nombre de secteurs par FAT de l'offset 36 à 39, le nombre de secteurs réservés de l'offset 14 à 15. Avec ces informations, nous pouvons trouver les secteurs où sont situés le début des FATs et le début du dossier root de la partition. Le début du premier FAT est trouvé en additionnant le secteur du boot record au nombre de secteurs réservés. Pour trouver le premier secteur où se situe le dossier root de la partition, nous additionnons

le secteur du boot record avec le nombre de secteurs réservés et avec le nombre de FATs multiplié au nombre de secteurs par FAT. À la figure 4.13 nous avons un exemple de secteur de partition avec les éléments mentionnés surligné en couleur. Par exemple, nous avons le nombre de secteurs réservés surligné en bleu.

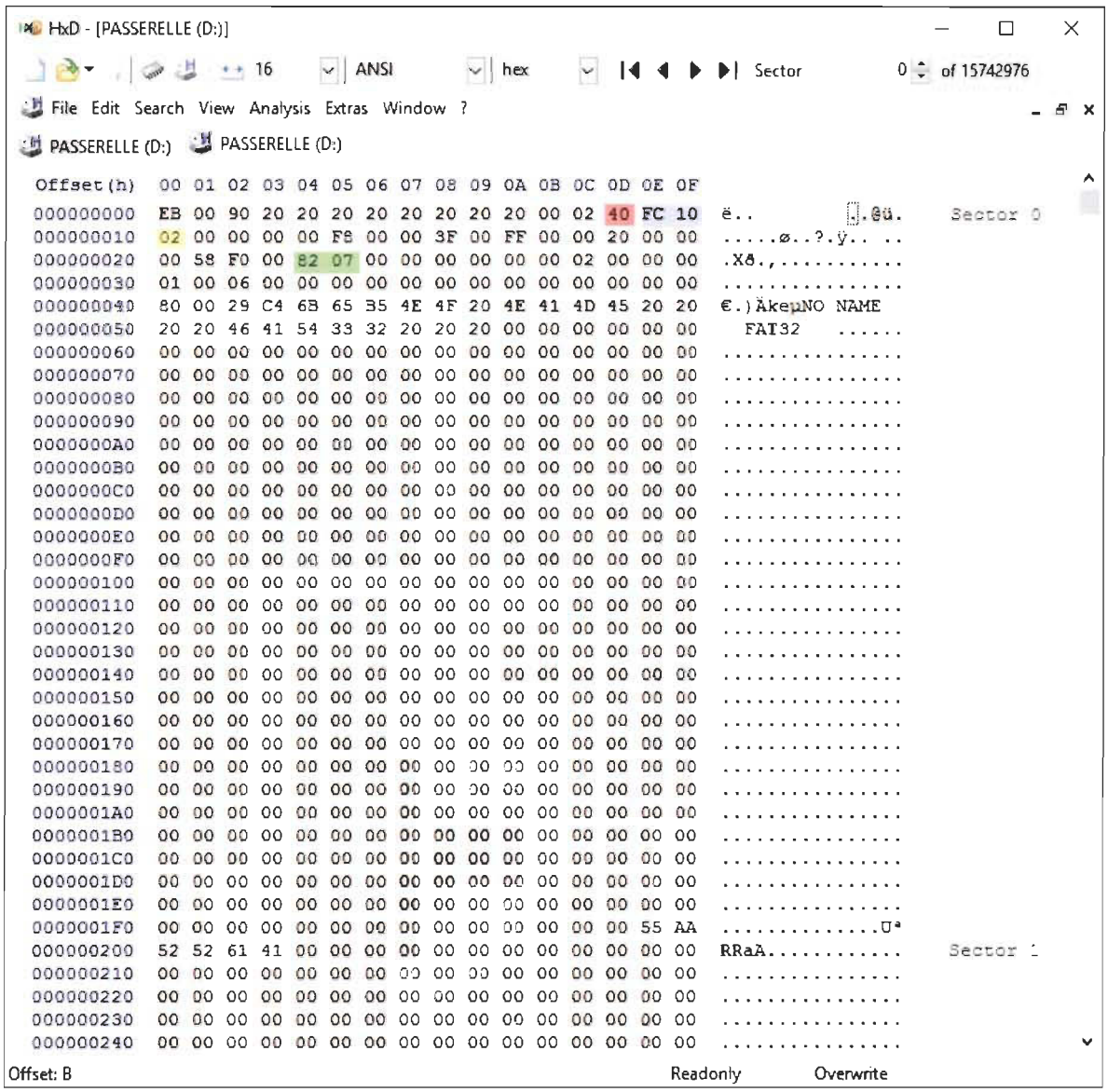

FIGURE 4.13 – Boot record de la partition

Nous avons fait allusion au FAT dans le paragraphe précédent sans l'expliquer. Le FAT est le File Allocation Table. C'est un tableau gigantesque dans lequel tous les clusters de données sont représentés. Chaque cluster est attribué quatre octets (32 bits) dans le FAT, d'où

l'épellation FAT32. Les deux premiers clusters (le cluster 0 et le cluster 1) de la table sont réservés donc, le premier cluster qui contient des données se situe à la troisième position dans le FAT soit à l'offset 8. Dans les quatre octets qui représentent le cluster dans la table se trouve le numéro du cluster qui le suit. Si aucun cluster ne le suit, l'entrée sera soit 0x0F 0xFF 0xFF 0xFF ou simplement 0xFF 0xFF 0xFF 0xFF. À la figure 4.14 nous avons le premier secteur d'un FAT. Les octets surlignés en rouge et en jaune sont les clusters 0 et 1 qui sont réservés et les octets surlignés en vert représentent le premier cluster de données. Les octets surlignés en bleu représente le cluster 9 qui est le début d'une chaîne de clusters. Le cluster 9 pointe vers le cluster 10 qui est en rose. Puis le cluster 10 pointe sur le prochain jusqu'au cluster 14 en orange qui marque la fin du fichier. À noter que les clusters dans un FAT ne sont pas nécessairement continus comme dans cet exemple. Le cluster 9 aurait aussi bien pu pointer sur le cluster 200.

La dernière particularité du FAT32 à expliquer est les clusters de dossier. Les clusters pour les dossiers sont considérés comme des clusters de données et donc sont présents dans le FAT avec les mêmes règles qui s'appliquent aux clusters qui entreposent de l'information de fichier. Les secteurs dans ces clusters sont organisés de façon particulière. Ils possèdent des entrées de 32 octets. Il y a deux types d'entrée, les entrées pour les longs noms de fichier ou dossier et les entrées pour les informations sur fichiers et dossiers. Dans le projet des noms de huit caractères et moins ont été utilisés afin d'optimiser le nombre d'entrées à chercher. Les entrées contenant les informations sur les dossiers et fichiers commencent par huit octets qui indiquent leur nom. Si le premier octet est 0xE5, il s'agit d'une entrée effacée. Étant donné que l'ordinateur préfère écrire dans de nouvelles entrées ou le premier octet est vide, il est possible de retrouver de façon très rudimentaire des fichiers effacés ayant une grosseur inférieure à un cluster en regardant les informations des entrées effacées. En fait, une des tâches de la défragmentation est justement de compresser les entrées de fichier et de dossier. Après les octets pour le nom, nous avons trois octets pour l'extension du fichier. Puis, nous

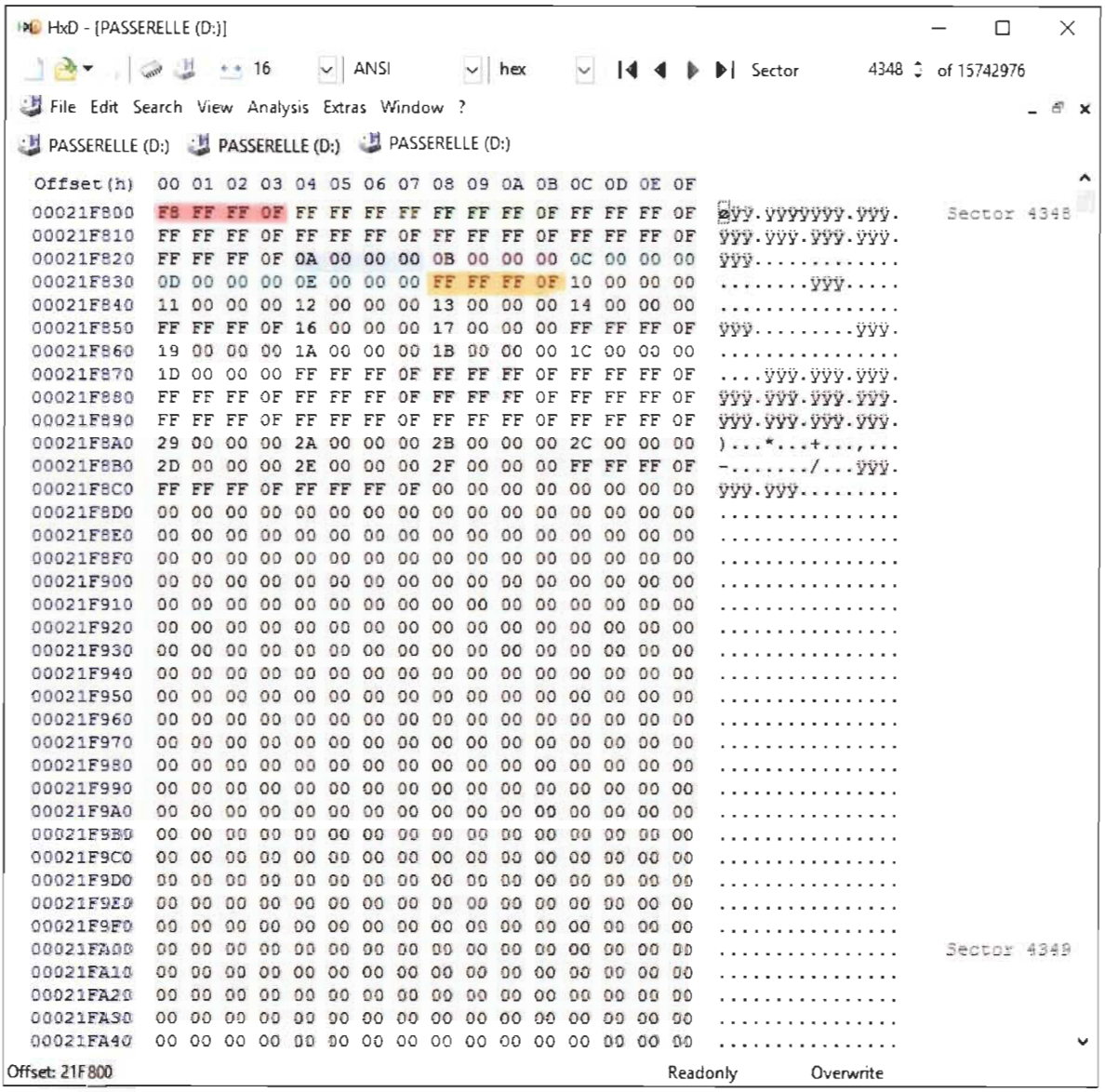

FIGURE 4.14 – Le premier secteur du premier FAT

avons un octet qui indique si l'entrée est un dossier ou un fichier. Il y a huit autres octets qui nous intéressent. Les deux octets qui indiquent les octets hauts du premier cluster sont à l'offset 0x14 et 0x15, leur complément sont à 0x1A et 0x1B. Finalement, les quatre octets qui indiquent la grosseur du fichier sont les quatre derniers de l'entrée. À la figure 4.15 nous avons un exemple d'un secteur de dossier. Les octets surlignés en bleu sont une entrée complète de fichier dans le dossier. Les autres octets surlignés représentent les différents éléments énumérés précédemment.

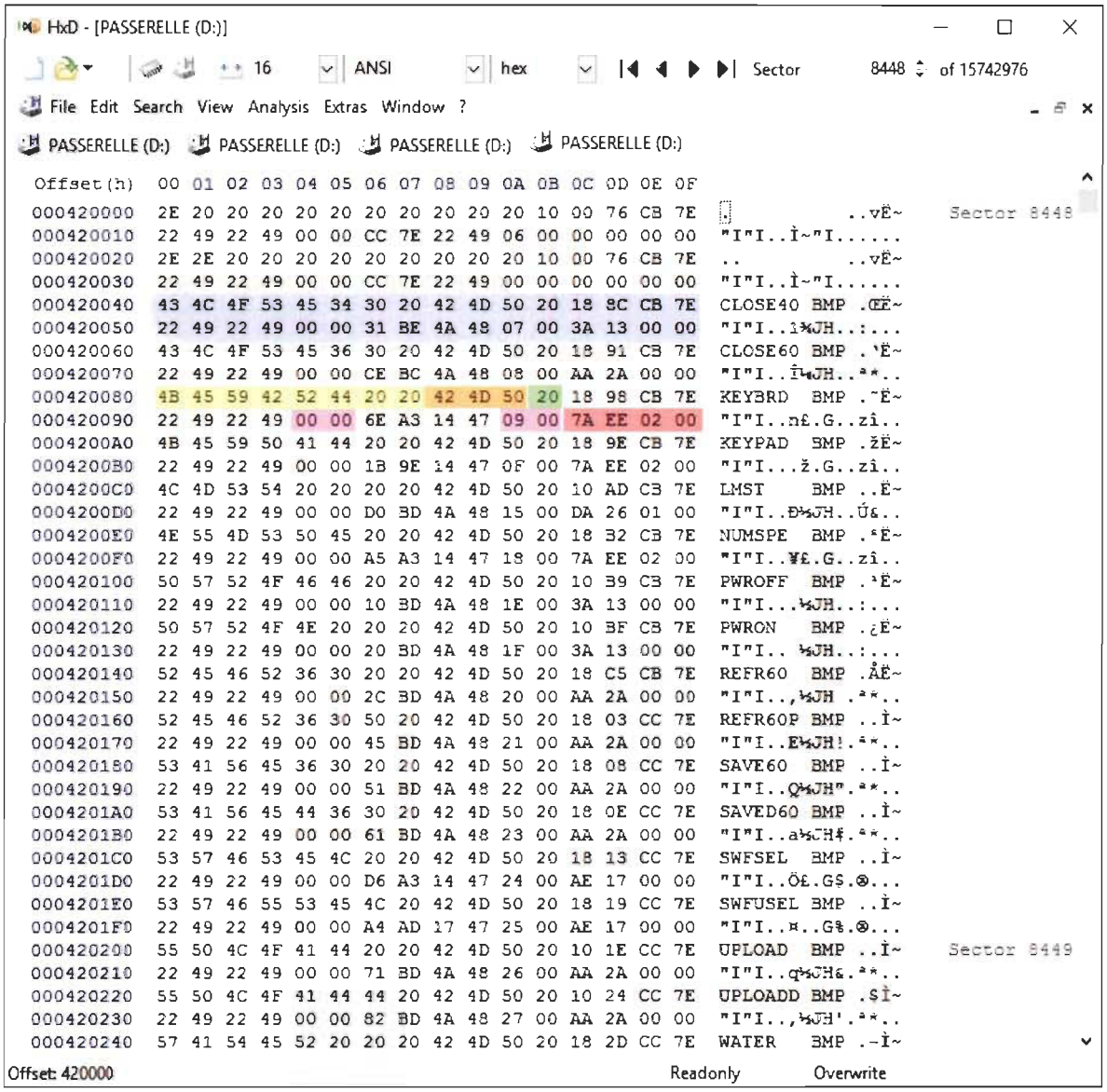

FIGURE 4.15 – Un secteur d'un dossier en FAT32

En résumé, pour faire une lecture d'un fichier dans un système de fichier FAT32 nous devons trouver les clusters où son information est entreposé. Pour faire ceci, nous devons nous référer à son entrée dans les clusters de son dossier afin de trouver son premier cluster et nous devons aussi nous référer au FAT pour savoir si ce fichier est réparti en plusieurs clusters. Pour trouver le dossier, nous devons faire la même chose jusqu'à ce qu'on remonte au dossier root. Pour trouver le dossier root nous devons regarder dans le boot record de la partition dont le numéro de secteur est indiqué dans le premier secteur de la carte SD. Les

fonctions développées rendent ce processus invisible, mais il est important de comprendre comment il fonctionne dans le cas où un problème subviendrait ou qu'une modification serait souhaitée.

4.3.4 Algorithme de lecture de la carte SD

Les algorithmes pour la lecture de fichier sur la carte SD (figure 4.16) se basent sur les notions du chapitre précédent. Pour la lecture de carte, il y a deux algorithmes qui sont importants. Le premier est un algorithme pour effectuer une lecture d'octets présents dans la carte. Il est utilisé à plusieurs reprises dans les algorithmes qui gèrent la structure du FAT et la lecture de fichier, mais il est très simple. Une chose importante à savoir lorsqu'on utilise les commandes de lecture de la carte SD est que les cartes qui sont indiquées comme HC ou High Capacity fonctionnent par adressage de secteur. Donc, pour lire, nous devons donner le numéro de secteur à la commande. Le minimum d'octets que nous pouvons lire est le nombre d'octets dans une page. Ceci est différent pour les cartes plus anciennes. Dans les vieilles cartes, l'adressage est fait à partir de l'octet lui-même et nous pouvions lire seulement un octet. Le nombre d'octets présents dans les cartes modernes empêche l'utilisation de ce type d'adressage. Pour faire la lecture, nous utilisons donc le numéro du secteur et nous envoyons la commande CMD17 avec le secteur en argument. Ensuite nous attendons de recevoir 0xFE qui indique le début de la réception de données. Finalement, nous lisons tous les données que la carte nous envoie. Il y existe aussi des commandes pour faire de multiples lectures de secteurs continus à la fois, mais ils ne sont pas utilisés dans le projet.

Le deuxième algorithme important pour cette section et le projet est celui qui vient trouvé le dossier contenant les images pour les différents menus. Il a deux parties à cet algorithme qui est représenté à la figure 4.17. La première est de faire la lecture du MBR et du Boot

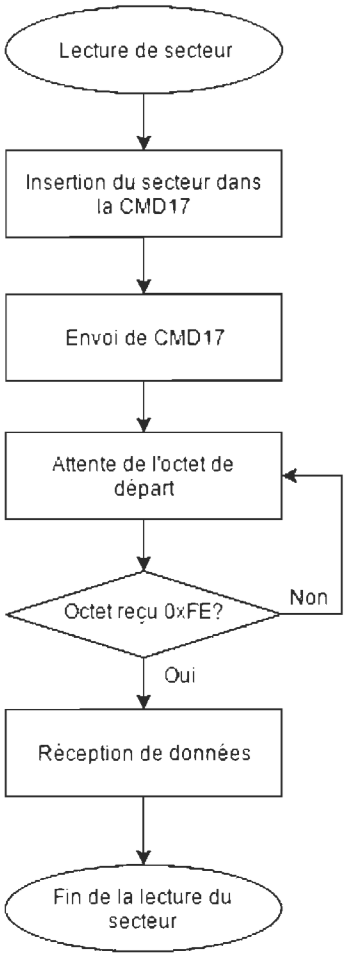

FIGURE 4.16 – Algorithme de lecture d'un secteur de la carte SD

record de la partition pour savoir ou se trouve le FAT et le dossier root. Pour faire ceci, on lit le secteur zéro de la carte et on regard les octets contenant le secteur ou commence le boot record de la partition. Ensuite, nous lisons le secteur trouvé et on cherche les valeurs qui nous intéresse afin de trouver les secteurs ou se situe le dossier root et le secteur ou commence le premier FAT. La deuxième partie est de trouver le dossier contenant les images. L'information importante à trouver pour atteindre cet objectif est le cluster de départ du dossier. Avec cette information, nous pouvons trouver les clusters suivants s'il y a lieu et nous pouvons calculer le secteur où se situe l'information du dossier. Il suffit d'additionner le secteur du dossier root assumant que celui-ci se situe dans le premier cluster de données, au numéro de cluster qui est soustrait par deux à cause des clusters réservés et multiplier par le nombre de secteurs par cluster. Le processus pour trouver notre dossier est le même que pour trouver un fichier, donc

cet algorithme s'applique dans les deux cas. On commence par lire les secteurs du dossier qui contiennent ce que l'on recherche et nous passons à travers les entrées de 32 octets en comparant les octets contenant le nom du dossier ou fichier au nom de l'entrée que l'on recherche. Ceci peut prendre de multiple lectures de secteur si notre entrée est loin du premier secteur et peut même nécessiter que l'on change de cluster. Pour changer de cluster, il faut aller lire le FAT à la position du cluster que nous avons fini de lire et prendre en note le numéro du prochain pour se positionner sur le premier secteur de ce nouveau cluster. Une fois que nous avons trouvé la bonne entrée, nous vérifions si c'est bien un fichier ou un dossier à l'offset 8 puis nous prenons en note le cluster ou commence le fichier. Ceci complète notre recherche pour un dossier ou fichier. Le processus pour lire un fichier est le même que celui que nous avons utilisé pour parcourir le dossier contenant ce que nous cherchions. Bien sûr un fichier ne contient pas d'entrée de fichier ou de dossier, mais des informations qui sont structurés en fonction de son format.

4.3.5 Algorithme d'affichage d'image bitmap

Le format d'image bitmap est un format simple qui a été utilisé afin de créer le menu. Lorsqu'un menu est affiché sur l'écran, le programme lit les images appropriées dans la carte SD et reconstruit ceux-ci aux endroits prédéterminés. Les pixels d'une image bitmap sont stockés de gauche à droite du bas jusqu'au haut. Donc le premier pixel lu est celui en bas à droite. Le fichier contient aussi un en-tête où des informations sur le fichier sont situées. Les informations qui nous intéressent sont la largeur et la hauteur de l'image ainsi que l'adresse où commence les données de pixel. Il est aussi important de savoir que lorsqu'on lit un bitmap, des octets de remplissage sont ajoutés pour rendre le nombre d'octets par rangée divisible par quatre. Par exemple, si nous avons une rangée de 31 pixels et que nous avons 24 bits par pixel, ceci nous donne une rangée de 93 octets. Donc, 3 octets de remplissage ont été ajoutés à la

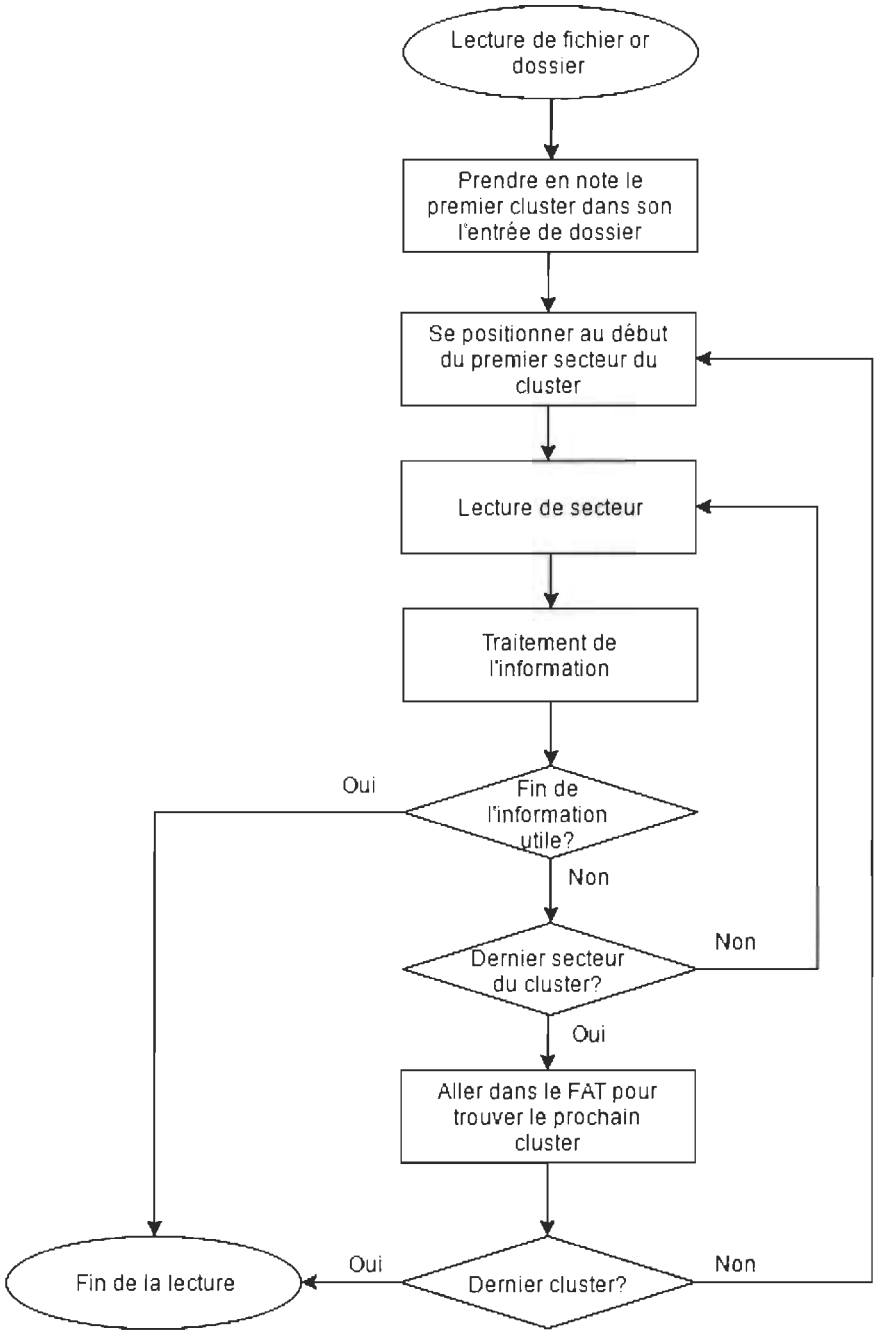

FIGURE 4.17 – Algorithme de lecture d'un fichier ou d'un dossier

fin de la rangée pour la rendre divisible par quatre. Si cette particularité n'est pas tenue, en compte des pixels noirs sont ajoutés, lorsque l'image est dessinée et le résultat est une image ou tous les pixels sont décalés. Les bitmaps utilisés dans le projet sont en format 24 bits donc un octet de chaque couleur commençant par le bleu, suivi du vert et finalement suivi par le rouge.

La façon que l'algorithme (figure 4.18) affiche une image est qu'il lit une rangée de pixels, il l'affiche et ensuite il lit la prochaine. Le cycle continue jusqu'à ce que la dernière rangée du haut soit affichée. Donc, au début le code lit le premier secteur de l'image dans la carte et prend en note la largeur, la hauteur et la position du premier octet. Puis, il lit le reste du secteur. Puisque le LCD utilise des pixels 16 bits, l'algorithme doit déconstruire les pixels 24 bits qu'il a lus de l'image et les reconstruire en 16 bits. Ce processus s'effectue dès que l'algorithme détecte qu'il a lu un pixel complet. Ce pixel 16 bits est ajouté à un tampon de rangée. Si le code arrive à une fin de rangée incluant les octets de rembourrage, il envoie la rangée au LCD pour qu'il l'affiche. Lorsque nous sommes à la fin du secteur, le prochain secteur est lu et si nous sommes à la fin du cluster, le prochain cluster est trouvé et on se positionne sur son premier secteur. Ce processus est répété jusqu'à ce que toute l'image soit lue et affichée.

Pour l'affichage ligne par ligne, avec la vitesse du PIC32MZ, il est difficile de voir le processus à l'oeil. Par contre avec un PIC24F, par exemple, nous pouvons voir l'algorithme tracer l'image. Présentement, la lecture s'effectue en même temps que l'affichage ce qui contribue à l'effet de traçage que l'on peut observer sur le PIC24F et un peu moins sur le PIC32MZ, car la lecture sérielle de la carte représente un bottleneck pour la rapidité d'affichage.

4.3.6 Algorithme relatif à l'écran tactile

L'écran tactile utilisé dans le projet est résistif. Pour savoir lorsqu'il y a une interaction, une interruption de patte est générée par le microcontrôleur provenant de la patte désigné pour cette tâche sur l'écran. Dans la fonction qui sert l'interruption, une lecture est effectuée sur l'écran. Cette lecture retourne deux valeurs entre 0 et 4095, une pour la valeur en x lu et une pour la valeur en y. Ces valeurs doivent être converties avant de pouvoir les utiliser, mais

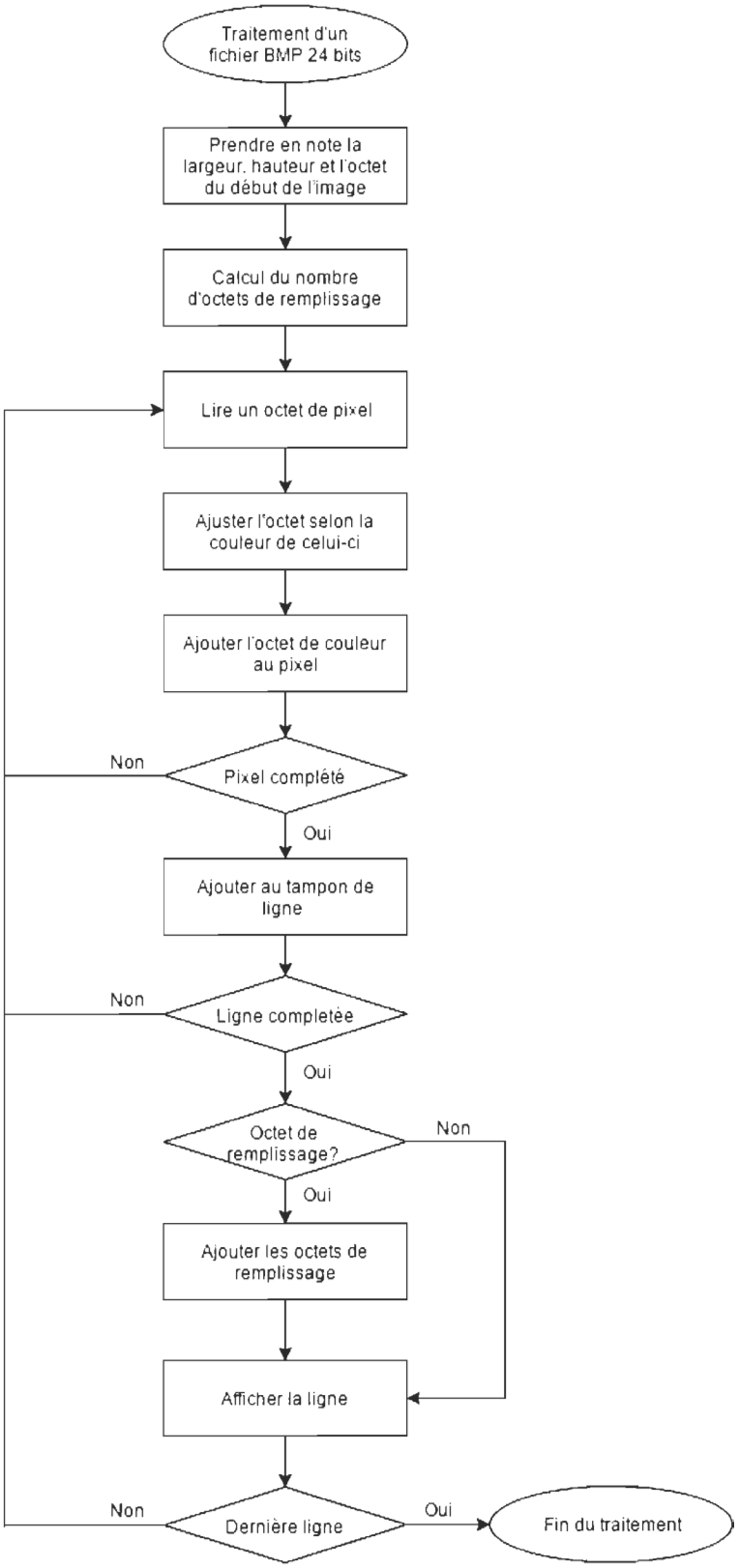

FIGURE 4.18 – Algorithme d'affichage d'un fichier bitmap

ceci n'est pas fait dans la fonction qui sert l'interruption. Dans les fonctions de menu, il y a toujours une vérification qui est faite au début de la boucle infinie pour vérifier si nous avons une nouvelle donnée tactile. L'interruption met une valeur booléenne à vrai lorsqu'elle est appelée. Lors de la vérification, si la valeur booléenne est vraie, la conversion est appliquée. Cette conversion prend la valeur brute que nous avons reçue de l'écran tactile et la convertie dans un intervalle qui respecte la résolution de l'écran LCD. La calibration vient d'un fichier de code trouvé en ligne. Ce fichier effectue un calcul matriciel avec trois points qui sont prédéterminés. Le résultat du calcul donne une matrice de coefficients qui, une fois multipliée avec les coordonnées brutes, nous retourne un résultat dans l'intervalle que nous recherchons. Cette méthode est plus complexe que de simplement faire une conversion linéaire. Par contre, elle donne des résultats beaucoup plus précis, car les valeurs brutes de l'écran tactile ne sont pas linéaires par rapport à la longueur et largeur de celui-ci. Quand une donnée tactile est convertie, elle passe à travers une fonction de vérification unique au menu dans lequel l'utilisateur est présent. Cette fonction retourne un caractère qui représente ce qui a été appuyé. Puis la fonction du menu en cours gère l'action à effectuer. La représentation de l'algorithme se trouve à la figure 4.19.

4.4 Résultats

La passerelle a été testée avec le programme PC pour valider sa fonctionnalité. Afin d'effectuer le test ont doit d'abord brancher la passerelle dans une source d'alimentation et partir le programme PC. Dans le logiciel, nous devons entrer l'adresse IP (figure 4.20) de la passerelle que l'on peut trouver dans le menu Wi-Fi de la passerelle (figure 4.21). Dans la passerelle, nous devons entrer l'adresse IP de l'ordinateur. Cette adresse peut être trouver en exécutant «ipconfig» dans le terminal windows (figure 4.22). Une fois les configurations sur la passerelle terminées, il faut appuyer sur le bouton qui permet d'envoyer la nouvelle configuration à

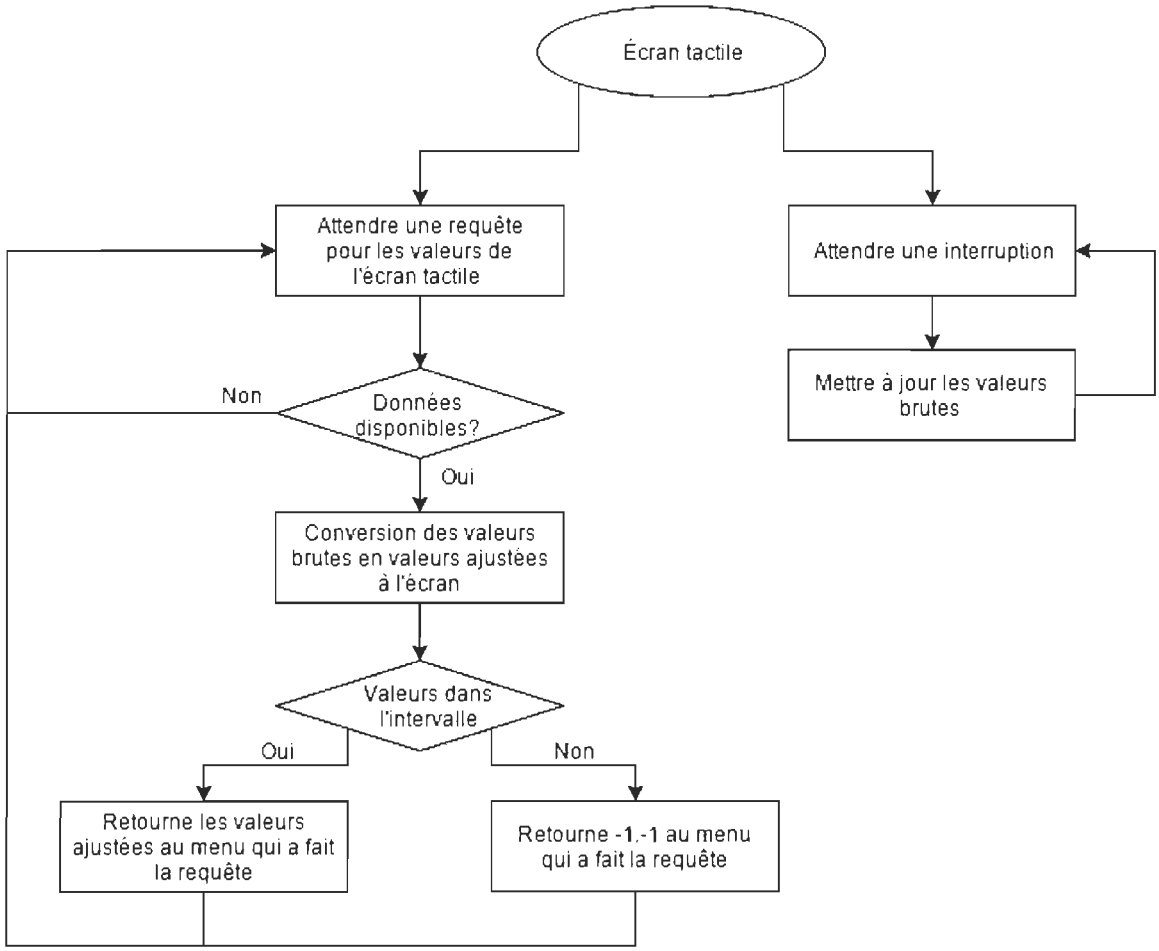

FIGURE 4.19 – Algorithme du traitement de l'information de l'écran tactile

la passerelle s'il y a lieu (figure 4.23). Puis, nous pouvons appuyer sur le bouton «connect» du logiciel PC. Afin d'effectuer des lectures sur demande on peut cocher la case «Read Tag on command». Les lectures sont effectués à tous les 10 secondes. La passerelle devrait maintenant effectuer des lectures et envoyer les résultats au PC. Si plusieurs étiquettes sont présentes, ceux-ci apparaîtront dans le tableau. La figure 4.24 montre un exemple de lecture de plusieurs étiquettes.

Lors de l'expérimentation, plusieurs étiquettes ont été testées et elles n'avaient pas tous la même portée. Donc, la conception de l'étiquette à un impact sur la portée de lecture. À la figure 4.25, nous avons trois étiquettes à une distance d'environ 120 cm de l'antenne. Par contre, seulement l'étiquette de coureur et l'étiquette blanche ont été captées. Pour le Wi-Fi, la portée

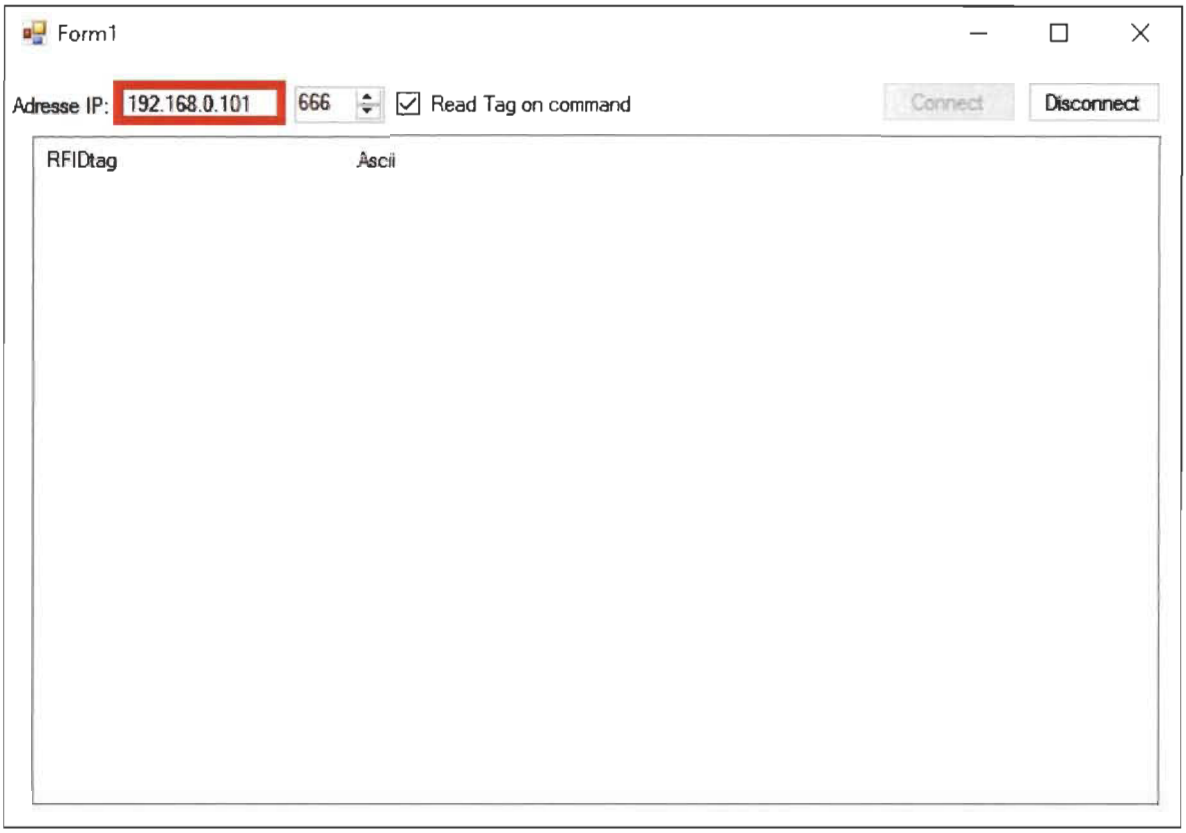

FIGURE 4.20 – Logiciel PC, champs pour l'adresse de la passerelle encadré en rouge

dépend du router utilisé, de l'antenne sur le module et de la géométrie de l'emplacement. Dans le laboratoire, le signal était capté partout dans la pièce.

Le résultat final est une plate-forme de développement pour une passerelle RFID à Wi-Fi. Nous avons aussi un programme PC qui démontre la fonctionnalité de la passerelle. Celle-ci possède un écran graphique LCD utile si un développeur veut créer une interface pour un utilisateur ou pour simplement déboguer. Avec le programme PC la passerelle fait présentement une lecture aux dix secondes et envoie le résultat pour l'afficher sur l'écran de l'ordinateur. L'interface de la passerelle permet à l'utilisateur de changer le nom et le mot de passe du réseau ainsi que l'IP de l'hôte. La passerelle est une bonne preuve de concept que ce type de périphérique est possible et est un bon point de départ pour un produit qui voudrait utiliser cette technologie.

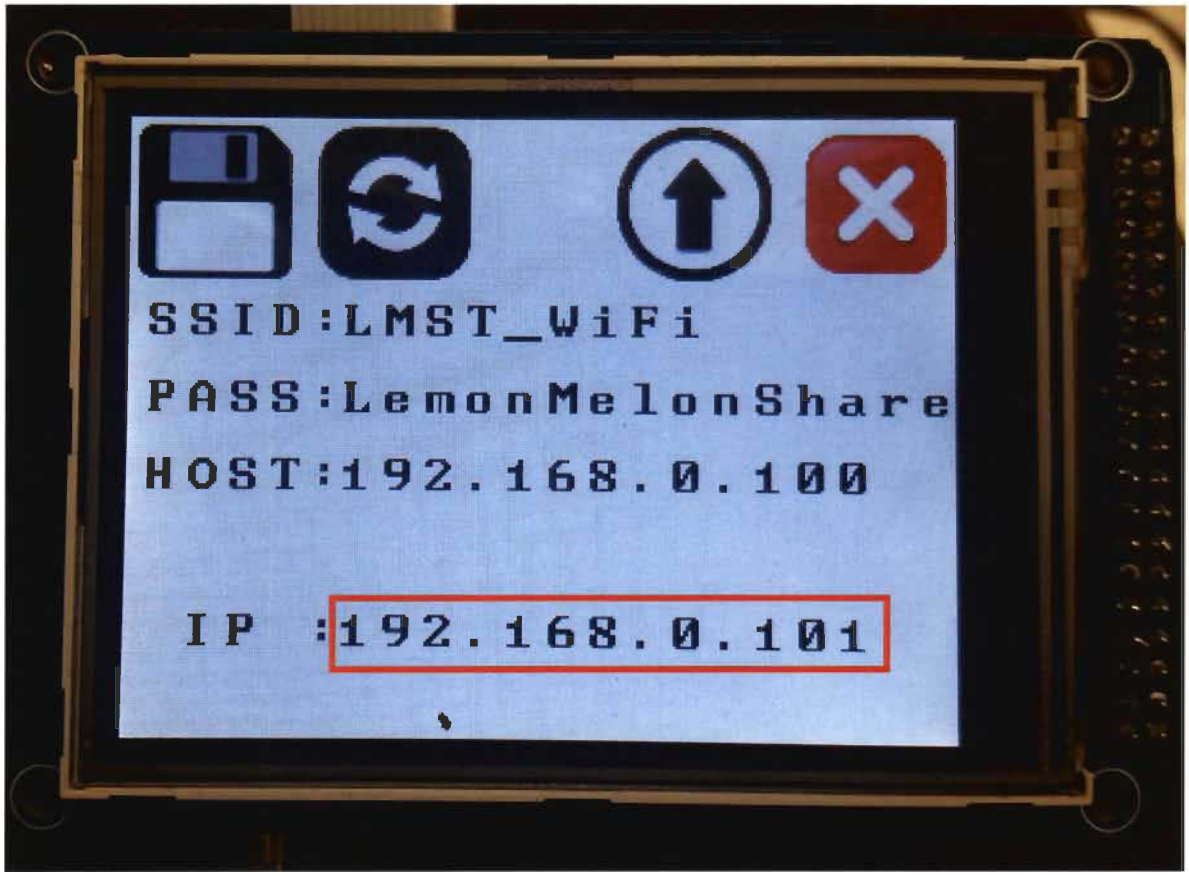

FIGURE 4.21 – Menu Wi-Fi, adresse de la passerelle encadrée en rouge

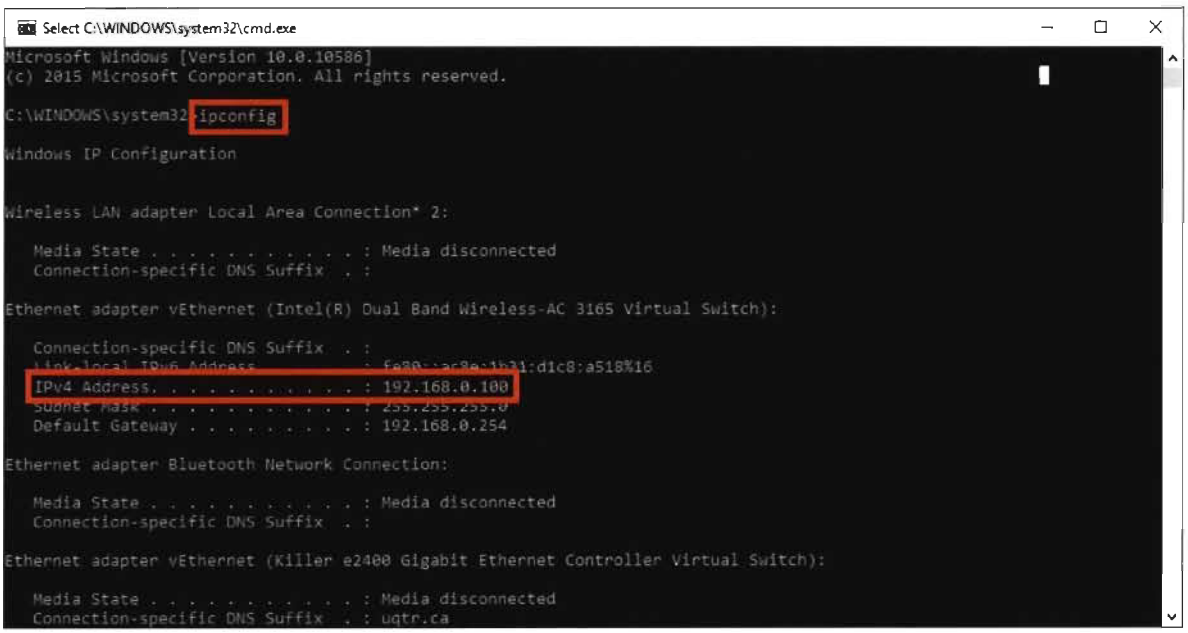

FIGURE 4.22 – Commande ip config dans le terminal, adresse du PC encadrée en rouge

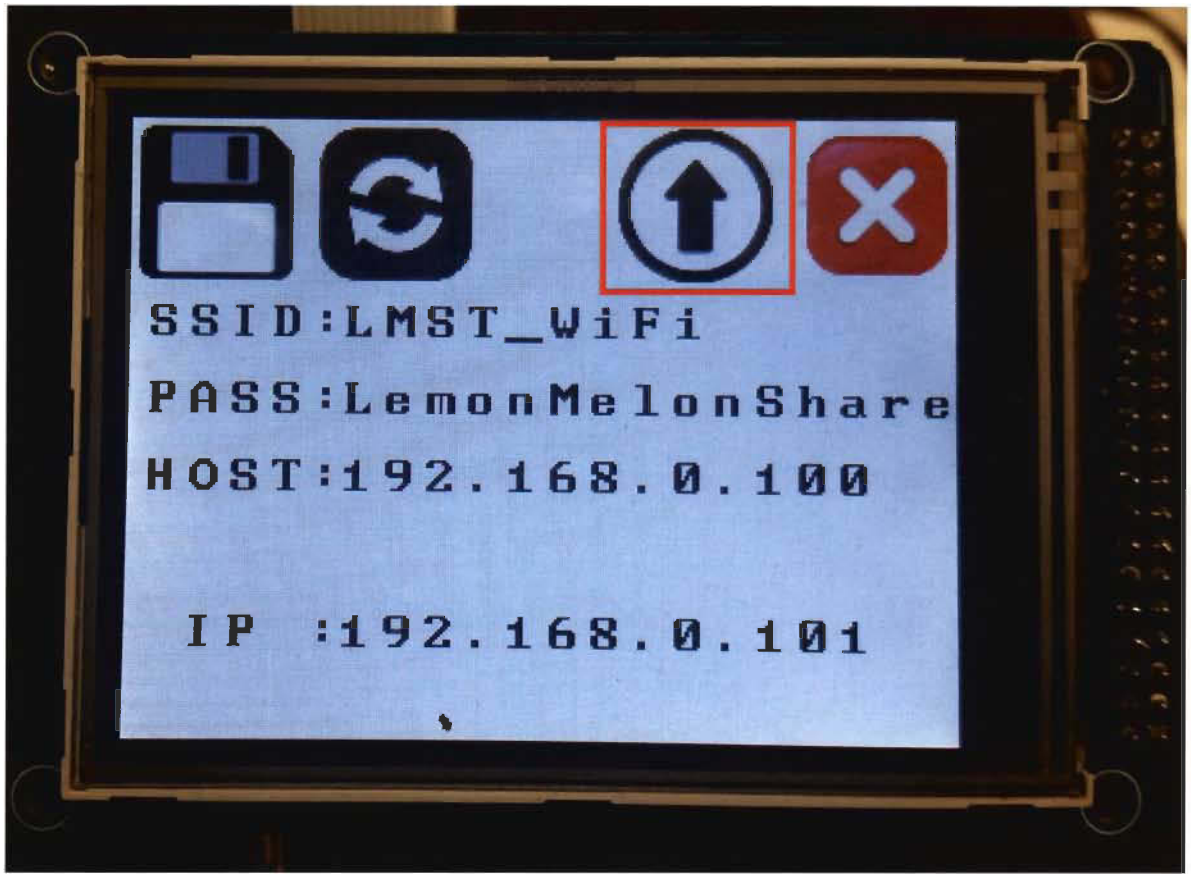

FIGURE 4.23 – Menu Wi-Fi, bouton qui permet d'envoyer la configuration encadré en rouge

4.5 Conclusion

Dans ce chapitre, les éléments de la passerelle autre que les modules de communication ont été détaillés. Le choix de technologie a dicté le reste des éléments dans ce chapitre. Avec un microcontrôleur nous avons dû faire le design et fabriquer un circuit imprimé afin d'atteindre nos objectifs. Ce PCB nous a permis de placer tous les éléments de la passerelle dans un format compact et pratique. Dans le design, nous avons opté pour un adaptateur mural qui nous fournissait le courant nécessaire pour le lecteur RFID. Nous avons aussi inséré un moyen d'avoir une communication directe avec un ordinateur pour le déboguer dans le cas où l'interface humaine ne serait pas adéquate pour cette tâche. L'interface humaine de la passerelle peut aussi aider au développement d'un produit fini. Elle possède la capacité d'utiliser

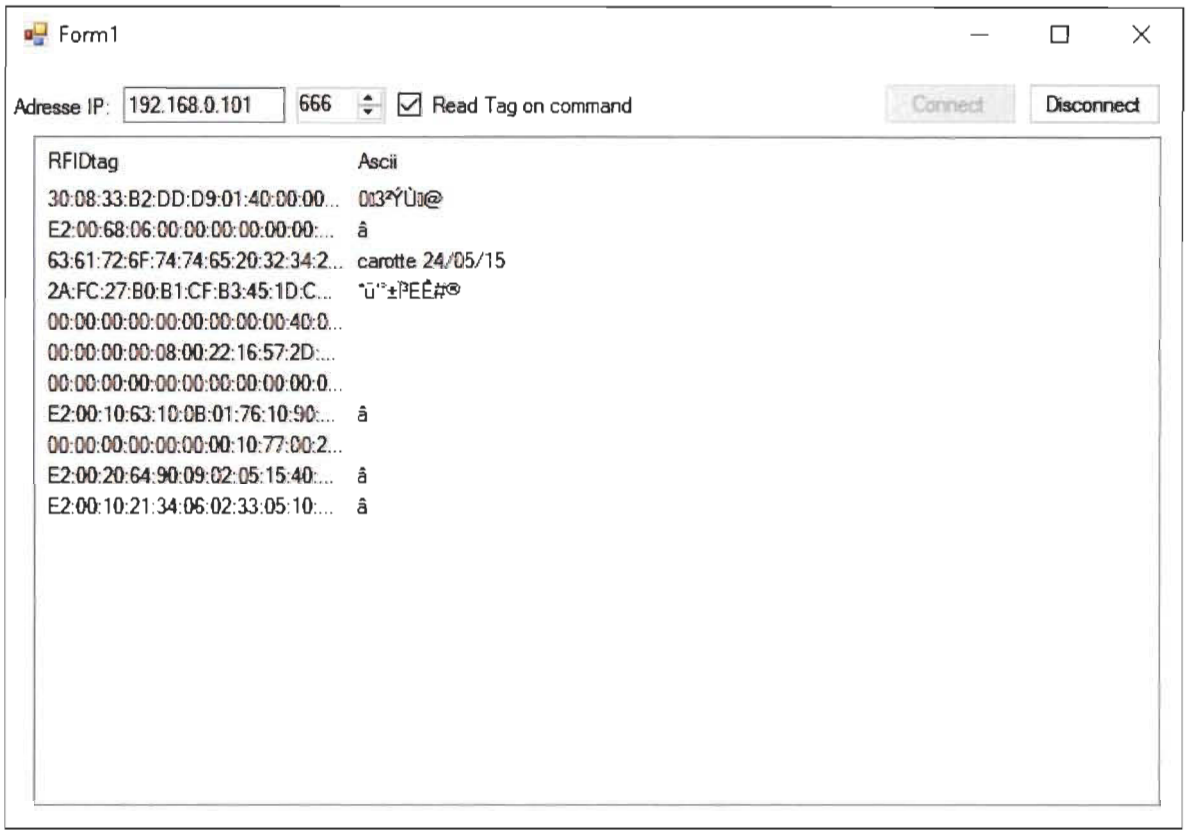

FIGURE 4.24 – Logiciel PC, détection de plusieurs étiquettes

une carte SD pour garder en mémoire des images qui composent un menu ou encore quelqu'un pourrait utiliser la carte pour stocker des données sur l'utilisation de la passerelle de façon locale. L'écran tactile de l'interface permet aussi une interaction avec l'usager et ouvre plusieurs possibilités de fonctionnalité. Au final, le PCB créé chez OSHpark a été assemblé et puis programmé afin de bien démontrer le concept et les possibilités de la passerelle.

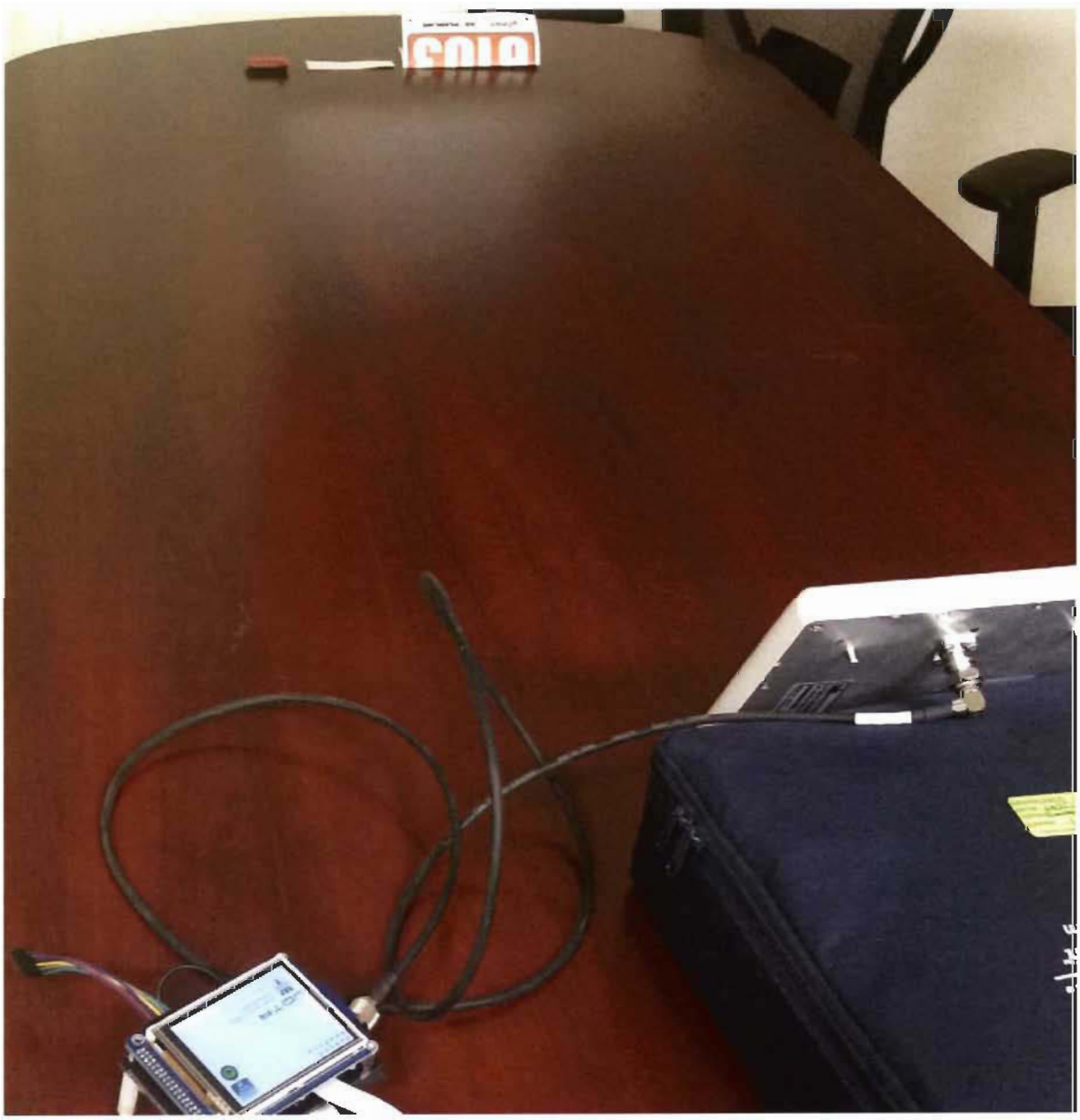

FIGURE 4.25 – Test de portée des étiquettes

Chapitre 5 - Conclusion

Ce mémoire a présenté les divers éléments de la conception de la passerelle RFID à Wi-Fi développer durant ce projet de maîtrise.

Le module RFID utilisé le ThingMagic compact M5E. Ce module a la caractéristique de pouvoir lire plusieurs étiquettes RFID par lecteur ce qui engendre plusieurs possibilités d'applications que des lecteurs conventionnels ne pourraient pas avoir vu la limitation d'une étiquette par lecture. Le module a pu être monté sur le circuit électrique avec des entretoises. Il a été connecté avec le câble plat de la plate-forme de développement du lecteur ainsi qu'avec le connecteur présent sur le module. Le module communique UART avec le microcontrôleur et des commandes spécifiques sont utilisées pour démarrer et effectuer des lectures d'étiquettes. Ces commandes ont été insérées dans le code créé sur la passerelle afin d'accomplir les objectifs établis pour le lecteur RFID.

Le module Wi-Fi est le RN171 de Microchip, il existe d'autres modules Wi-Fi qui auraient pu accomplir la tâche de communication par ce standard, mais pour les raisons présentées dans le chapitre 3 ce module a été choisi. La version soudée sur un circuit électrique a été utilisée dans le projet pour faciliter son intégration au PCB final et pouvoir le changer, car le format du PCB du module en est un qui est standard pour beaucoup d'autres modules de communication. Le protocole utilisé est l'UDP qui est un protocole simple et efficace pour faire de

la communication d'un point à un autre à travers une infrastructure réseau conventionnelle. Finalement, les algorithmes de communication ont été créés afin de complémenter le module RFID utilisé.

La passerelle a été conçue afin de donner le plus d'outils à un développeur potentiel. Elle a un port USB qui sert de communication directe avec un ordinateur. Elle a un écran graphique tactile pour faire du débogage ou pour développer une interface utilisateur. Une carte SD peut être utilisée pour garder en mémoire les différents éléments des menus de l'interface et elle pourrait être utilisée pour stocker des informations sur la passerelle ou sur le logiciel que le développeur voudrait créer. Finalement, cette passerelle remplit les objectifs qui ont été établis au début de ce document et ouvre la porte à d'autres projets qui voudraient tirer avantage de la technologie présente sur celle-ci.

Comme applications possibles, la passerelle pourrait être développé dans le but de faire de l'inventaire dans un espace confiné comme une pièce de laboratoire ou un réfrigérateur. Afin de poussé les fonctionnalité de la passerelle plus loin, des solutions infonuagiques comme Ayla, Electric Imp ou Nabto pourraient être utilisées. Ceux-ci remplaceraient le module de Microchip. Une passerelle minimaliste pourrait aussi être conçu afin de réduire le coût. C'est-à-dire une passerelle ayant seulement le module RFID, un module WiFi et un microcontrôleur avec moins de capacité de traitement. Il y a donc quelques directions qu'un nouveau projet pourrait prendre s'il veut se basé sur celui-ci.

Bibliographie

[1] "Mercury5e and M5e-Compact Developer's Guide", ThingMagic, 2012.

[2] "2.4 GHz IEE Std. 802.11b/g Wireless LAN Module", Microchip, 2014.

[3] "WiFly Command Reference Manuel", Microchip, 2015.

[4] SD Card Association Technical Committee, "SD Specifications Part 1 Physical Layer Simplified Specification", SD Group, 2006

[5] "file allocation table - 16bit", 2016. [En ligne]. "http ://www.beginningtoseethelight.org/fat16/index.htm". [8 Septembre 2016].

[6] "FAT32 Dick Structure Information", 2016. [En ligne]. "http ://www.easeus.com/resource/fat32-disk-structure.htm". [8 Septembre 2016].

[7] "Triad Magnetics WSU050-1500-R", 2016. [En ligne]. "http ://www.digikey.ca/product-search/en ?keywords=237-1418-ND". [8 Septembre 2016].

[8] "Amphenol FCI SFW12R-1STE1LF", 2016. [En ligne]. "http ://www.digikey.ca/product-search/en ?keywords=609-1900-1-N". [8 Septembre 2016].

[9] "Microchip Technology RN171XVW-I/RM", 2016. [En ligne]. "http ://www.digikey.ca/product-detail/en/microchip-technology/RN171XVW-I-RM /740-1044-ND/2673186". [8 Septembre 2016].

[10] "Raspberry Pi", Wikipedia, 2016. [En ligne]. "https ://en.wikipedia.org/wiki/Raspberry_Pi". [8 Septembre 2016].