Rapport de projet de fin d'études
Création d'un système domotique sans fil

Soutenance de mi-projet

Thomas Maurice
Benoit Maliar

École Polytechnique Universitaire de Lille
Département IMA

Thomas.Maurice@polytech-lille.net
Benoit.Maliar@polytech-lille.net

17 décembre 2014

Table des matières

Introduction

Dans le cadre de notre projet de fin d'étude de cinquième année en Informatique, Micro-électronique, Automatique, nous avons été amené à travailler sur la création d'un système domotique sans fil.

L'objectif est de créer ce système à l'aide d'une raspberry pi et de petits systèmes embarquées à base de microprocesseur et d'antenne pour permettre une production *low cost*, reproductible chez soi et modifiable à souhait.

Ce projet se découpe en deux parties faisant appel à nos compétences en Informatique et en Electronique. La première se focalise sur la construction de la carte électronique et la deuxième sur la création et l'implémentation du système d'exploitation dans le microprocesseur et de l'interface d'administration des données.

Dans un premier temps, nous expliquerons le cahier des charges et les réflexions qui ont permis de dresser la liste du matériel nécessaire. Dans un second temps, nous préciserons l'avancé des travaux sur les deux parties et ce qu'il reste à faire. Enfin, nous terminerons sur un planning détaillant le futur du projet jusqu'à la soutenance finale.

1 Cahier des charges

1.1 Idée générale

Le but du projet est de réaliser un réseau de capteurs distribués à base de microprocesseurs MSP430 qui feront remonter des informations fournies par différents senseurs (gaz, fumée, température, luminosité) à une Raspberry Pi. Cette dernière traitera ces données et permettra à un utilisateur de les consulter. Le but est que l'importe quel capteur puisse être branché à la carte et qu'elle s'y adapte.

1.2 Partie capteur

Chaque carte doit permettre l'envoi de données à l'aide de composants radios sur différentes bande de fréquences afin de pouvoir utiliser une autre bande si celle actuellement utilisée est surchargée ou parasitée.

Nous avons choisi d'utiliser pour chaque carte un CC1101 et un CC2520. Le premier permet de communiquer sur la bande 868/915 MHz et le second sur la bande 2.4 GHz (Wifi). La bande la plus basse offre l'avantage d'une meilleure portée, au détriment du débit d'information.

Ces deux modules radios seront connectés à un microprocesseur MSP430 sur lequel sera installé un système d'exploitation pour connecter la carte sur un réseau (Contiki). La carte doit être capable d'identifier les différents senseurs qui lui seront branchés, afin de se reflasher de manière dynamique des parties du MSP430 de manière à ce qu'il dispose des drivers nécessaires à piloter le nouveau senseur. Les drivers seraient stockés sur la Raspberry Pi.

1.3 Partie serveur

Les données envoyées par les capteurs seront récupérées par une carte identique aux autres mais reliée à une Raspberry Pi en SPI qui collectera toutes les informations dans une base de données PostgreSQL et qui offrira, via une interface de visualisation/administration, la possibilité d'interagir avec elles.

2 Matériel utilisé

La liste globale du matériel utilisé pour une carte (et le matériel unique) se trouve en annexe 1.

2.1 Matériel imposé

Les éléments imposés par le cahier des charges sont :
- Le **MSP430xx** pour la partie microcontrolleur, choisi pour sa compatibilité avec Contiki et l'expérience engrangé sur d'autres projets.
- Le **CC1101** pour la partie radio 868 MHz, choisi pour sa bande de fréquence.
- Le **CC 2xxx** pour la partie radio 2.4 GHz, pour sa bande de fréquence.
- **Contiki** pour le système d'exploitation du microcontrolleur, choisi pour l'interaction qu'il apporte au microcontrolleur, notamment pour le flashage à distance et la réalisation de pins à chaud.
- Une **Raspberry Pi** faisant office de serveur récoltant les données. Elle a été choisie pour sa taille et sa consommation énergétique tout en permettant d'effectuer sans problèmes.
- Une LED, indice visuel d'allumage à distance.

2.2 Matériel choisi

2.2.1 Général

Nous avons sélectionné le MSP **MSP430F5418A**
- *16 Bit Ultra-Low Power Microcontroller*
- *128 Kb Flash*
- *16 Kb RAM*

pour ses caractéristiques : Contiki tenant sur 10 Kb de RAM, nous avons pris une petite marge (6 Kb) pour le code à implémenter (i.e la couche MAC pour le CC1101). La justification des 128 Kb de Flash vient du fait qu'il n'existe pas de composant ayant moins de Flash et assez de RAM.

Pour le module radio 2.4 GHz, nous avons sélectionné le CC2520 et non le CC2420 qui propose les mêmes caractéristiques car le CC2520 est le module recommandé pour les nouveaux design.

2.2.2 Filtre des antennes

Afin de réaliser le filtre pour les deux antennes (868 MHz et 2.4 GHz), nous avions le choix entre construire le filtre à l'aide de capacités, inductances et résistances ou d'utiliser des baluns déjà conçus. Nous avons choisi les baluns car le faible prix, le besoin de moins de composant (gain de place) nous a permis de nous concentrer sur les autres éléments de la carte à calculer en sachant que le balun serait fiable.

2.2.3 Connecteurs HE10

Pour connecter les senseurs (température, luminosité, etc) à une carte capteur, nous avons décidé d'utiliser des connecteurs HE10. Ces derniers nous permettent de configurer l'emplacement des capteurs d'une manière générique. En effet, ces derniers peuvent être de types différents (SPI, tout ou rien, analogique) et nous avons décidé de créer une interface permettant de ne pas s'occuper du type du capteur (qui sera géré directement par le MSP). Il nous a donc fallu un connecteur permettant d'accueillir tous les bus possibles en une seule fois.

Ces connecteurs sont au nombre de 4 par carte permettant de connecter jusqu'à 4 capteurs par carte.

2.2.4 Flashage du MSP

Dans un premier temps, afin de pouvoir reflasher le MSP à distance, nous avons décidé de mettre un connecteur Mini-USB sur la carte pour pouvoir le flasher en premier lieu tout en ne prenant pas trop d'espace sur la carte.

2.2.5 Liaison capteur/Raspberry Pi

Nous avons choisi de connecter la Raspberry Pi à la carte via une liaison SPI afin de faciliter l'interfaçage avec le microcontrolleur et de laisser libre de le port série de la Raspberry pour du *debug*.

3 Partie Electronique

3.1 Analyse

Dans un premier temps, après établissement des besoins de la carte, nous avons du nous pencher sur les *datasheets* et *référence design* des différents composants (notamment les modules radios pour la RF).

3.2 Bibliothèque Eagle

Pour créer la carte et le PCB, nous avons utilisé le logiciel **Eagle.** Dans ce cadre, nous avons du créer nos propres bibliothèques de composants quand celle ci n'étaient pas disponibles comme par exemple pour les CC, les quartz et les baluns.

3.3 Calcul des éléments du circuit

Une partie du travail sur la partie électronique a été de calculer toutes les valeurs de composants nécessaire aux éléments du circuits, ceci incluant :

- les capacités de découplage de l'alimentation
- les ferrites pour l'alimentation
- les capacités de découplages des quartz
- les capacités pour la RF
- Autres capacités et inductances dans le circuit

3.4 Partie RF

Après discussion avec A. Boé, nous avons optimisé la partie RF en rapprochant au maximum les éléments critiques (comme les capacités de découplages) des modules radios et en réfléchissant au positionnement des éléments pour obtenir le moins d'interférences possible.

En outre, un plan de masse homogène a été placé sur toute la carte (avec l'aide de *vias*) sauf sur la partie RF pour permettre l'émission correcte des signaux.

3.5 Schematic

L'étape suivante de la création des cartes a été l'élaboration des différents *shematic*.

3.5.1 Carte principale

Le *schematic* de la carte principale a été crée sous Eagle 7 et comprend les éléments nécessaires aux cartes "capteur" et "Raspberry". Ce *schematic* est visible en annexe 2.

3.5.2 Carte senseurs

Pour chaque type de senseur (luminosité et température pour le moment), un schematic a été crée. Ces *schematics* sont visibles en annexe 3.

3.6 PCB

Une fois les schematics terminés et validés, l'étape suivante a été la création du PCB et le routage des cartes. Différentes vérifications et validation ont été effectuées avec les différents professeurs et intervenant avant l'aboutissement à un PCB valide.

3.6.1 Carte principale

Le PCB de la carte avait pour objectif d'être terminé pour la mi-novembre. Il a été terminé pour début décembre.

Ce PCB est visible en annexe 4.

3.6.2 Carte senseurs

Les différents PCB des cartes "senseurs" sont visibles en annexe 5

3.7 Communication avec les cartes filles

Pour communiquer avec les vartes filles, nous sommes partis sur un bus universel en 9 signaux, comprenant alimentation, masse, port série, deux chips select, une entrée tout ou rien et une entrée analogique. De cette sorte la carte mère sera capable de traiter tout type de capteur.

4 Partie Informatique

4.1 Choix technologiques

Le SGBD choisi est PostgreSQL, puisqu'il est entièrement Open Source (par opposition à MySQL propriété d'Oracle). Le serveur applicatif pour lequel nous avons opté est Node.js, qui permet d'exécuter du code JavaScript côté serveur.

Node.js est basé sur le moteur V8 de Google (utilisé notamment par Google Chrome) et permet une exécution rapide et asynchrone de code Javascript. C'est notamment pour cette capacité à exécuter le code de manière asynchrone, et donc optimisée d'un point de vue temps de traitement, que nous avons opté pour Node.js. En effet, la Raspberry Pi étant monocoeur, nous ne pouvions pas opter pour une solution basée par exemple sur PHP, et allouer plusieurs processus à un PHP-FPM pour en augmenter les performances.

Un autre avantage de Node.js est qu'un très grand nombre de modules est disponible via le gestionnaire de paquetages *npm* et permet de simplifier de manière drastique le développement en fournissant des modules pour gérer les identifications, les templates, les connexions à la base de données... La contrainte étant d'adopter un style de programmation clair et strict et de ne pas céder à la tentation de faire n'importe quoi, comme la syntaxe de JavaScript le permet.

Pour prendre en charge la partie graphique et design, nous avons opté pour le toolkit graphique Twitter Bootstrap, qui est objectivement beau et facile à intégrer et à prendre en main. Il se distingue également des autres (INK) par son poids très léger et par la qualité de sa documentation.

Au niveau du système de construction de l'application, nous avons opté pour la suite classique en développement Node.js : npm, bower et Grunt en tant que système de build.

4.2 Fonctionnalités de l'application Web

L'application sur la Raspberry Pi doit être en mesure de permettre à l'utilisateur de visualiser les données des capteurs de manière simple et intuitive. Elle doit également permettre de restreindre l'accès aux fonctionnalités du sites aux utilisateurs identifiés et de proposer des fonctions basiques de gestion de compte.

Le rafraîchissement de la page de présentation est prévu pour se faire automatiquement, ou a la demande de l'utilisateur. La partie automatique sera gérée par jQuery et des appels à une API REST qui nous permettra d'aller récupérer les valeurs des différents capteurs de la base de donnée. De cette manière on a une interface dynamique et une facilité d'utilisation accrue pour l'utilisateur. Il est également possible à l'utilisateur d'afficher les détails d'un capteur individuellement. On obtient ainsi de jolis graphs permettant de se rendre compte de l'évolution de telle ou telle métrique.

A terme l'application pourra proposer des graphs des différentes métriques remontées en fonction du temps, éventuellement des statistiques, et pourquoi pas proposer un système d'alerting par email lorsque certaines valeurs atteignent des niveaux réellement anormaux. Cependant ces fonctionnalités sont optionnelles, le plus important étant de pouvoir consulter les données

à un instant t.

4.3 Interface et ergonomie

Nous avons commencé à établir un design de ce à quoi peut ressembler l'interface. Notez que les données de la figure 1 ci jointe sont *générées aléatoirement en base* pour le moment et servent ici à titre d'illustration. Le point important est qu'au login l'utilisateur peut d'un simple balayage de l'oeil vérifier que tout va bien ou pas à un instant t. Dans un second temps, si il le souhaite il peut descendre dans le détail plus finement en dépliant le bloc correspondant à la pièce qui l'intéresse et en affichant le graphique qui l'intéresse.

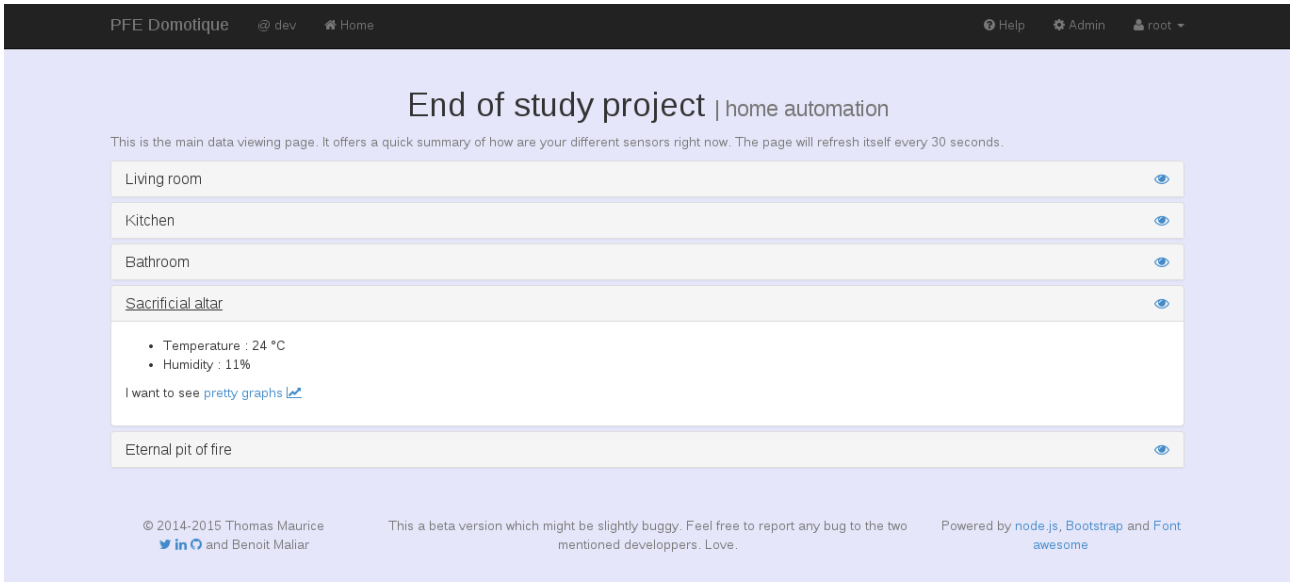


Figure 1 – Aperçu de l'interface

4.4 Fonctionalités disponibles le jour de la soutenance

Au jour de la soutenance, les fonctionnalités suivantes sont implémentées :
— Gestion des comptes utilisateurs et administrateur
— Ajout/suppression de capteurs
— Affichage des valeurs récentes en page d'accueil
— Rafraîchissement automatique de ces valeurs
— Affichage de graphiques par capteurs et par métrique
— L'interface est jolie

4.5 Méthode de développement

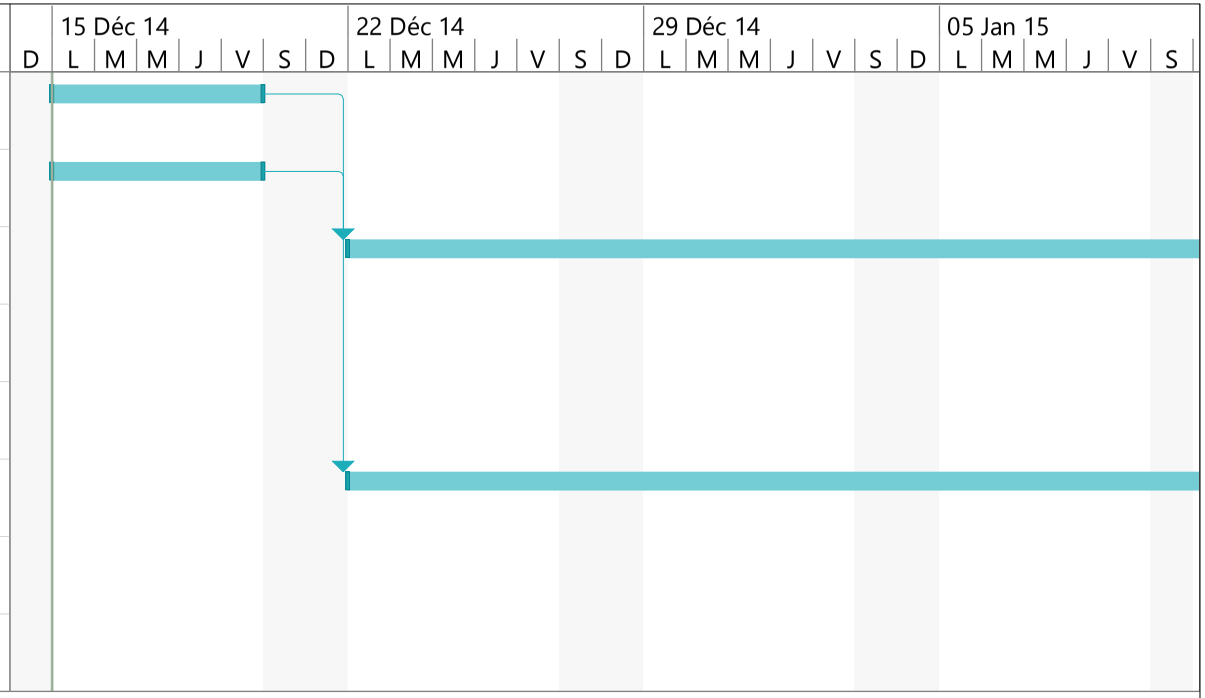
Pour développer l'application Web, comme nous avons fait pour la carte, ainsi que pour le présent rapport d'ailleurs, nous utilisons le système de versionnement Git afin de synchroniser notre progression l'un avec l'autre. Le dépôt est hébergé sur l'un de nos serveurs.

Le développement s'articule autour d'un liste de fonctionalités que nous traitons de manière séquencielle, et nous réalisons des tests fonctionnels à chaque étape du développement, de manière à nous assurer que chaque fonctionnalité fonctionne comme elle le doit, avant de s'attaquer à la suivante. Cela nous assure un code relativement robuste et un peu éprouvé.

5 Calendrier

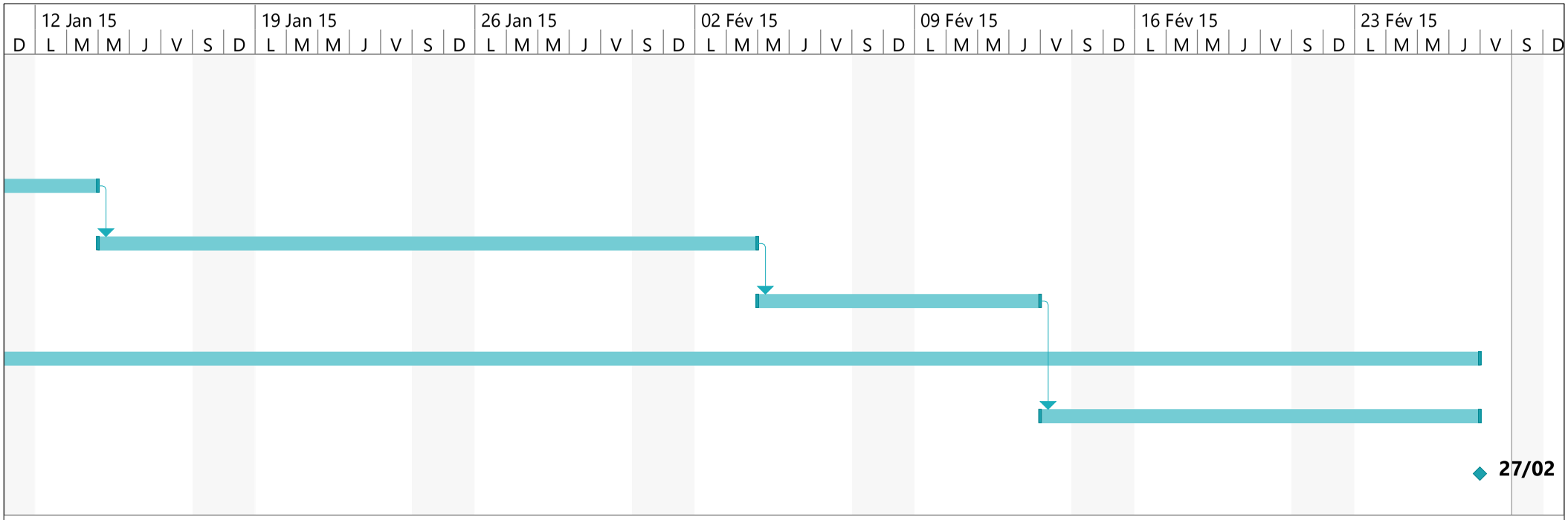
Le diagramme de Gantt suivant donne le prévisionnel des tâches restantes.

N°	Nom de la tâche	Durée	Début	Fin
1	Production de la carte	5 jours	Lun 15/12/14	Ven 19/12/14
2	Application Web fonctionnelle	5 jours	Lun 15/12/14	Ven 19/12/14
3	Soudure des composants	17 jours	Lun 22/12/14	Mar 13/01/15
4	Avancée sur Contiki	15 jours	Mer 14/01/15	Mar 03/02/15
5	Interfaçage MSP/Raspberry	7 jours	Mer 04/02/15	Jeu 12/02/15
6	Amélioration application Web	49 jours	Lun 22/12/14	Jeu 26/02/15
7	Réglage Contiki	10 jours	Ven 13/02/15	Jeu 26/02/15
8	Soutenance projet	0 jour	Ven 27/02/15	Ven 27/02/15



Projet : PFE
Date : Lun 15/12/14

Tâche
Fractionnement
Jalon
Récapitulative
Récapitulatif du projet
Tâche inactive
Jalon inactif
Récapitulatif inactif
Tâche manuelle
Durée uniquement
Report récapitulatif manuel
Récapitulatif manuel
Début uniquement
Fin uniquement
Tâches externes
Jalons externes
Échéance
Avancement
Progression manuelle



Projet : PFE
Date : Lun 15/12/14

Tâche		Récapitulatif inactif		Tâches externes	
Fractionnement		Tâche manuelle		Jalons externes	
Jalon		Durée uniquement		Échéance	
Récapitulative		Report récapitulatif manuel		Avancement	
Récapitulatif du projet		Récapitulatif manuel		Progression manuelle	
Tâche inactive		Début uniquement			
Jalon inactif		Fin uniquement			

Conclusion

Pour le moment, le projet se déroule bien, malgré un démarrage un peu lent sur la partie électronique. Le fait que la carte soit sur le point d'être tirée nous permet d'être assez optimistes quand à la suite du projet.

L'application Web est déjà bien avancée, la priorité est donc mise sur l'intégration du système d'exploitation Contiki au projet, une fois que les cartes seront soudées.

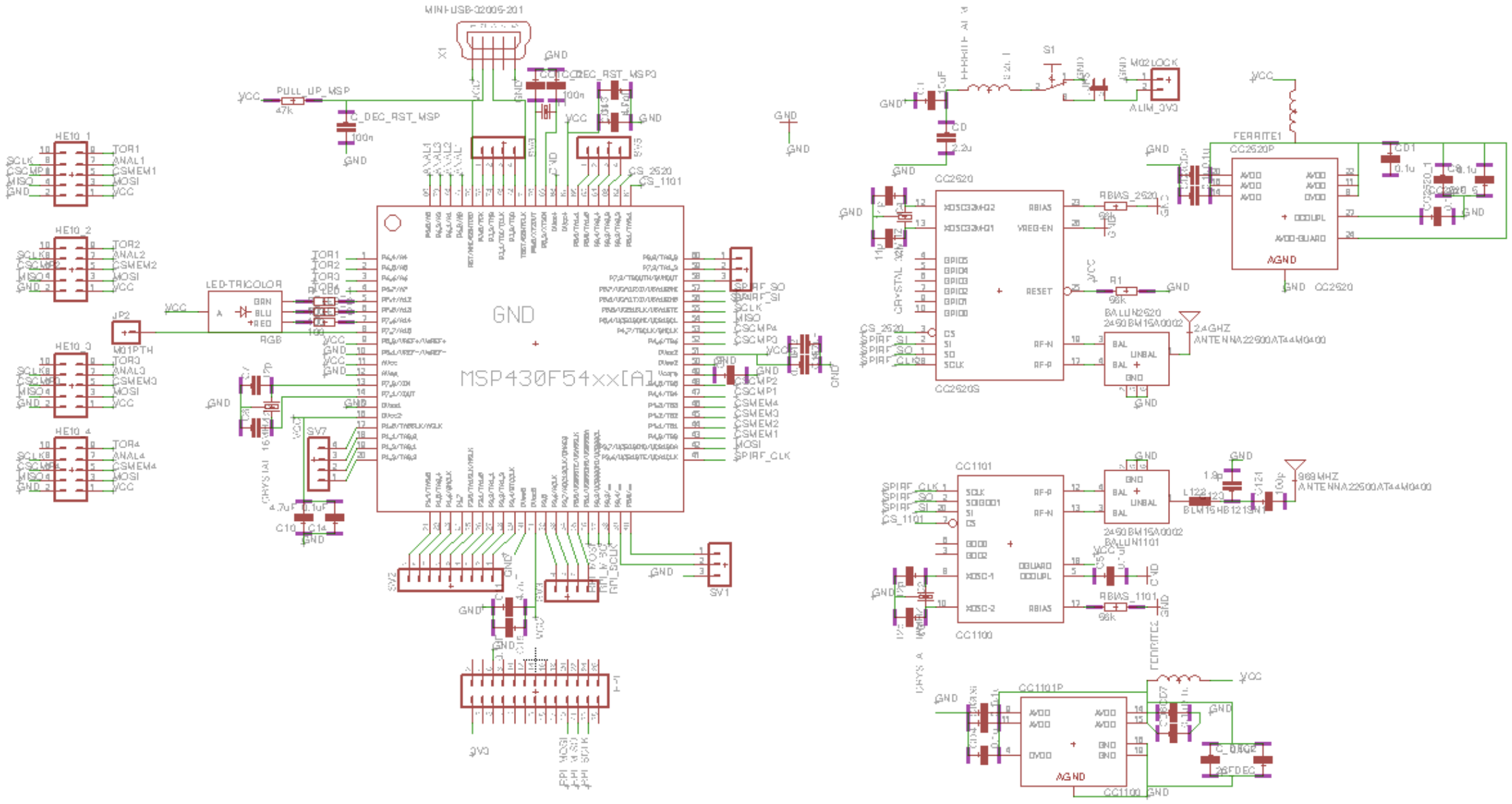
Annexes

Type composant	Destination	Value	Package
Balun	CC2520	2.4 GHz	0603
Balun	CC1101	868/915MHz	0805
Mini USB connector	Flash MSP		
Connecteur HE10 2x5 points droits	Connexion capteurs		
Boitier piles	Alim		
LED RGB	LED		1.6 mm x 1.6 mm x 0.7 mm
Launchpad	Launchpad		
Quartz	CC1101	16MHz	2.5 mm W x 3.2 mm L x 0.6 mm H
Capa équilibre Quartz	CC1101 & MSP	12 pF	0603
Quartz	CC2520	32MHz	1.6 mm W x 2 mm L x 0.5 mm H
Capa équilibre Quartz	CC2520	14pF	0402
RBIAS2520+ pull up2520+RBIAS1101	CCxxxx	56Kohms	0805
C1[4/5/6]01 + C271+ découpl CCxxxx 0.1uF	Alim CC2520	0.1uF	0402
C2	Alim CC2520	2.2uF	0402
Ferrite	Alim		0402
L 122	CC1101	5.6nH	0402
C123	CC1101	1.8pF	0402
C124	CC1101	100pF	0402
C51	CC1101	0.1uF	0402
Résistance	LED	100ohms	0603
Résistance	Pull-up alim MSP	47Kohms	0603
C découplage reset	MSP	0.1uF	0402
Antenne	CC2520	2.4GHz	7 mm L x 2 mm W x 2 mm H
Antenne	CC1101	868MHz	7 mm L x 2 mm W x 0.8 mm H
Quartz	MSP	24MHz	2.5 mm W x 3.2 mm L x 0.6 mm H
Flash	Capteurs	4Mb	SO-8
Capteur température	Capteurs		TO-92-3
R pull up	MSP	47Kohms	0402
Quartz bis MSP 32Mhz	MSP	32MHz	2.5 mm W x 3.2 mm L x 0.8 mm H

Annexe 1: Liste du matériel

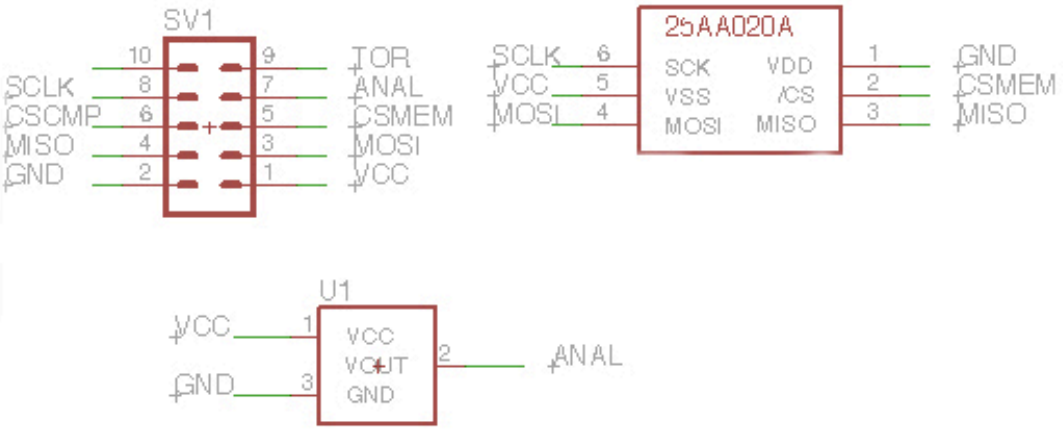
Quartz bis MSP 16Mhz	MSP	16MHz	2.5 mm W x 3.2 mm L x 0.6 mm H
Quartz bis MSP 12Mhz	MSP	12MHz	2.5 mm W x 3.2 mm L x 0.7 mm H
Capa découplage alim CCxxxx 2pF	CCxxxx	2pF	0402
Capa alim switch	All	10uF	0603
Inductance alim switch	All	2.2uH	0603
Buck boost	Alim		WSON-10
Capa Vcore	MSP	4700pF	0603
Capa dvcc	MSP	4.7uF	0603
Self filtrage DC	Filtrage des CC		0603
MSP430F5418A	MSP		
CC1101	CC1101		
CC2520	CC2520		

Annexe 1: Liste du matériel

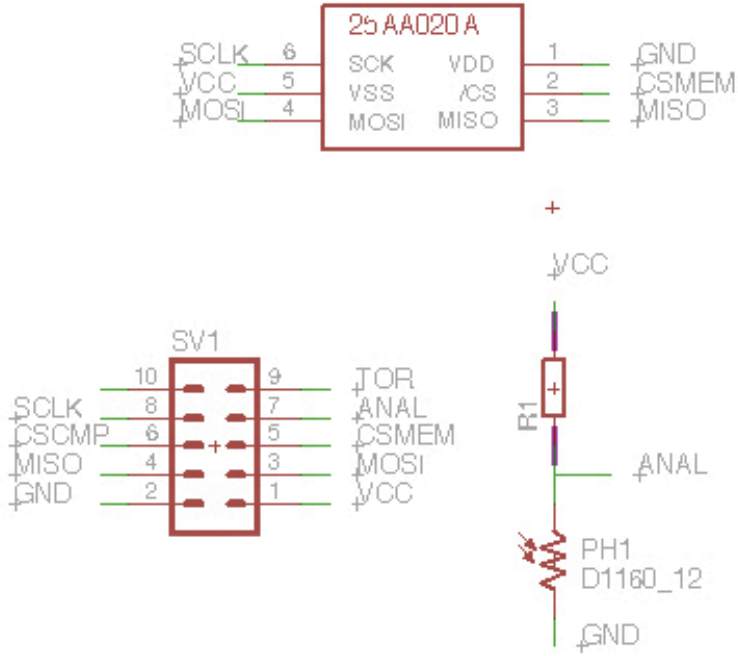


Annexe 1: Schematic de la carte principale

Luminosité

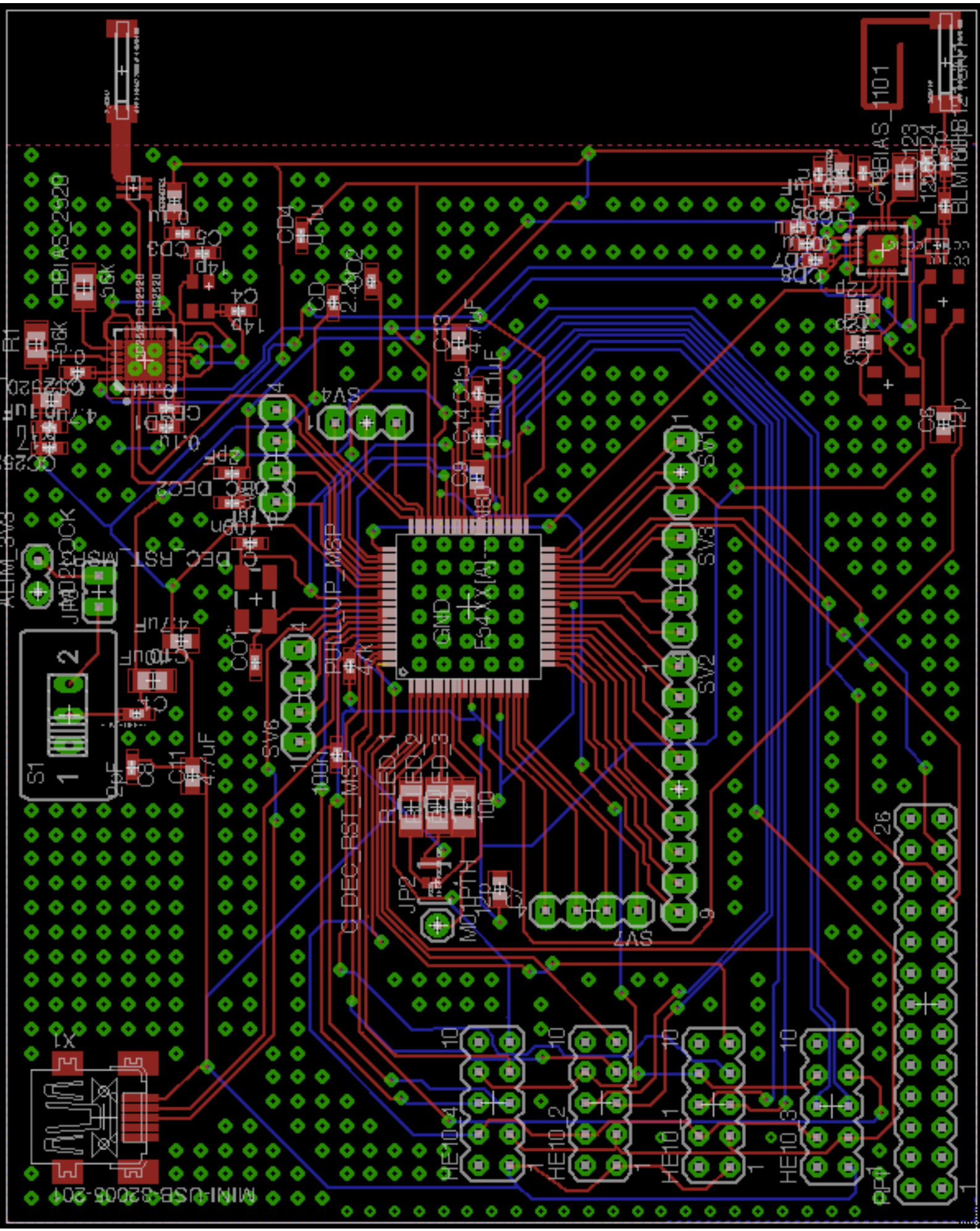


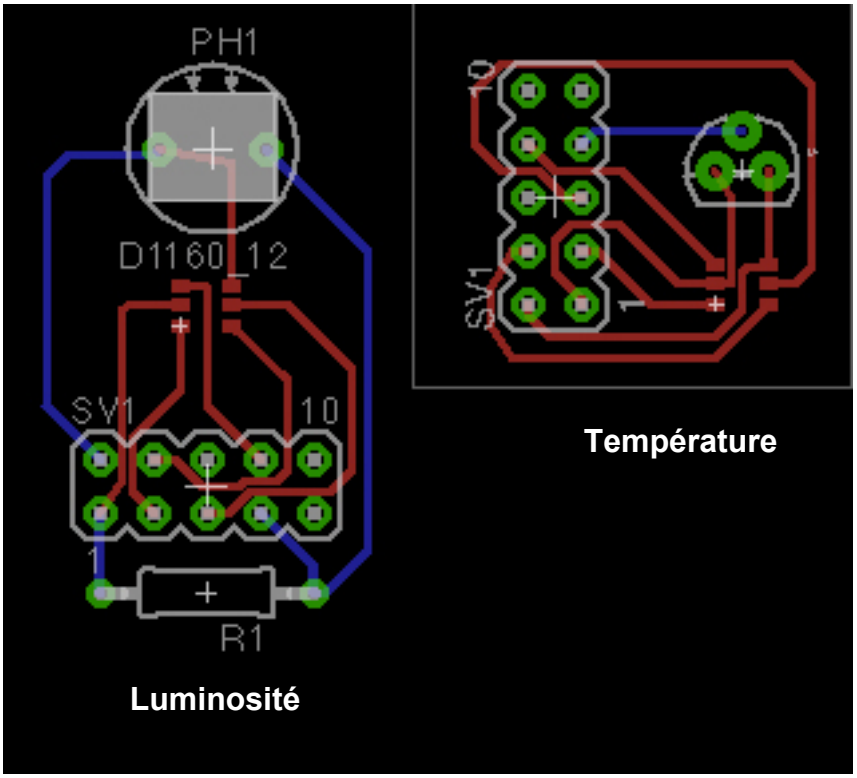
Température



Annexe 2: Schematics des cartes "senseurs"

Annexe 3: PCB de la carte principale





Annexe 5: PCB des cartes senseurs