



Détecteur de drone par
radio-goniométrie

D'Acremont Antoine
Cotten Guillaume
Legay Kevin
Kennan Aya
Shehade Mohammed
Rigaud Michaël

Table des matières

Table des matières	1
Introduction	2
1 Contexte	3
1.1 Nature du besoin	3
1.2 Etat d'avancement du projet	3
2 Ingénierie Système	4
2.1 Analyse Fonctionnelle	4
3 Tests unitaires	6
3.1 Raspberry Pi	6
3.2 PIC	9
4 Réalisation	12
4.1 Radiogoniomètre Doppler	12
4.2 Application Android : S.M.A.R.T Comm Center	12
Conclusion	15
A Documentation Technique du Raspberry Pi B+	17
B Documentation Technique à Raspbian	19
Table des figures	20

Introduction

Contexte

1.1 Nature du besoin

Les drones sont de plus en plus présents dans le monde moderne et font maintenant partie intégrante du paysage urbains. Il est en effet possible d'acheter pour 50€ un drone miniature dans n'importe quel rayon de jouet de grandes surfaces, ... Mais son usage ne s'arrête pas au loisir puisque l'actualité a montré que l'intrusion de drones dans des sites sécurisés représentaient un risque de sécurité majeur. Le risque de sécurité que représentent ces drones peut aussi s'étendre à d'autres lieux, moins sensibles, mais où leur intrusion peut avoir des conséquences désastreuses comme un aérodrome de campagne ou au dessus d'un terrain de sport pendant une compétition.

Notre projet, SMART (System with Multi Antennas to Reorient a Target), doit répondre à ce problème en permettant de détecter ces drones.

1.2 Etat d'avancement du projet

L'état de l'art a permis de déterminer plusieurs méthodes pour détecter un drone aérien. Elles sont principalement acoustiques, optiques ou électromagnétiques. Compte tenu du budget et de la complexité des différents systèmes observés l'équipe a opté pour une solution entièrement électromagnétique. La solution envisagée est un système passif de radio-goniométrie qui réceptionne les ondes émises par le drone puis utilise l'effet Doppler pour obtenir la direction d'émission par rapport à un système d'antennes fixe. Un dispositif muni de deux systèmes d'antennes sera alors en mesure d'obtenir la position approximative du drone à détecter. La connaissance de la position du drone pourra servir au développement de systèmes de brouillage ou de piratage du drone pour le neutraliser définitivement.

CHAPITRE 2

Ingénierie Système

2.1 Analyse Fonctionnelle

Diagramme Pieuvre

Le diagramme pieuvre permet de mettre en évidence rapidement la fonction principale du système et les principales contraintes qui s'appliquent sur le système. Ce dernier est représenté par l'ovale central et l'ensemble des éléments extérieurs ayant une influence sont matérialisés tout autour. Les différentes relations sont appelées les fonctions de contraintes qui naissent d'une contrainte imposée par un élément extérieur « météo », de l'existence d'un produit déjà existant « un autre drone émettant des ondes » ou encore d'une exigence particulière de l'utilisateur voire de la présence de normes et de législations, de limitations lié au budget ou du type d'alimentation énergétique nécessaire.

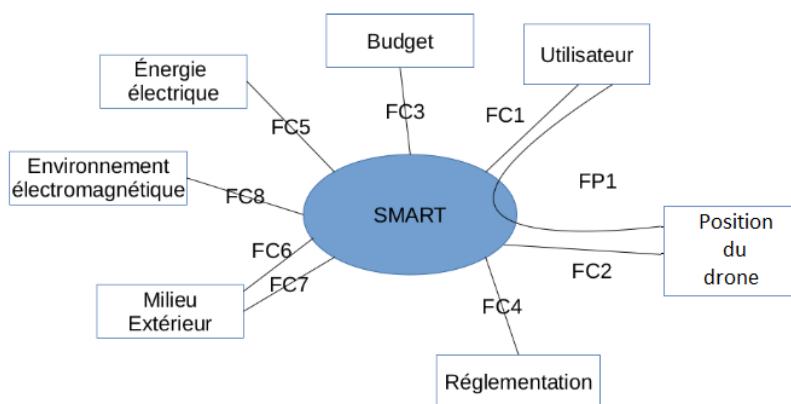


FIGURE 2.1 – Diagramme pieuvre

Diagramme FAST

Ce diagramme présente la manière de penser et d'agir. Le diagramme FAST se construit de gauche à droite, dans une logique du pourquoi au comment. On développe les fonctions de service du système en fonctions techniques. On choisit des solutions pour construire finalement notre système. On a mentionné les fonctions techniques chacune à part pour trouver la solution convenable qui nous permet à la fin la réalisation finale du système. En utilisant des outils et méthodes déjà existant, on a trouvé des solutions qui satisfont les fonctions demandés. L'antenne goniomètre était l'une des solutions les moins chères pour la détection du drone, à condition d'avoir au minimum deux antennes pour préciser la position et la vitesse du drone. Dès la détection du drone, il sera alors possible de déterminer sa position, d'enregistrer cette position via un logiciel dédié (MATLAB)

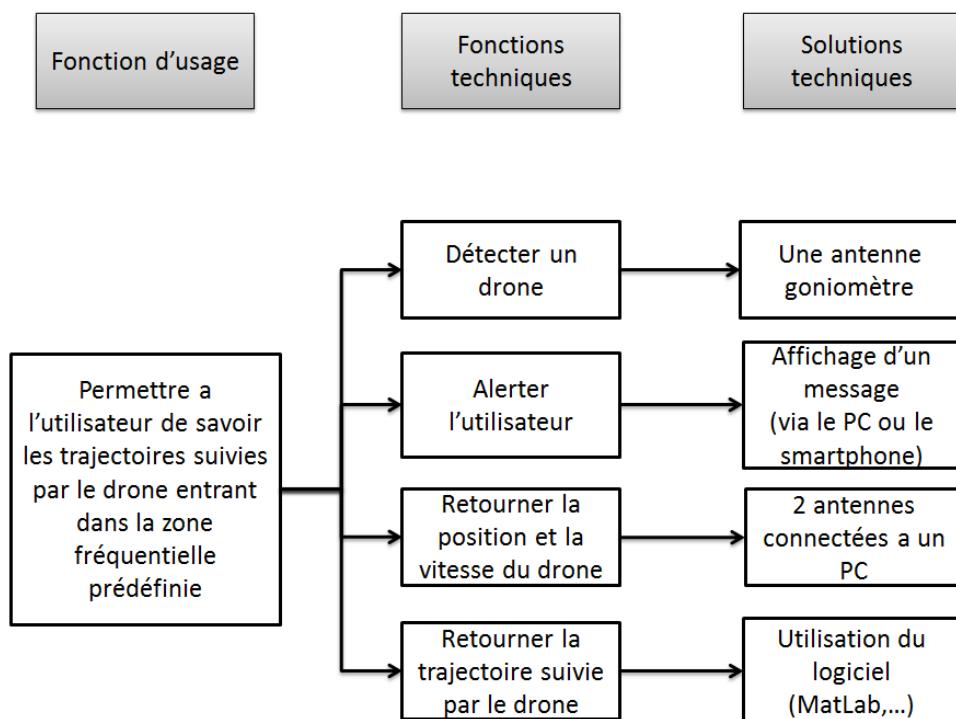


FIGURE 2.2 – Diagramme pieuvre

CHAPITRE 3

Tests unitaires

3.1 Raspberry Pi

La documentation technique lié a notre Raspberry Pi es situé en annexe à la page 17

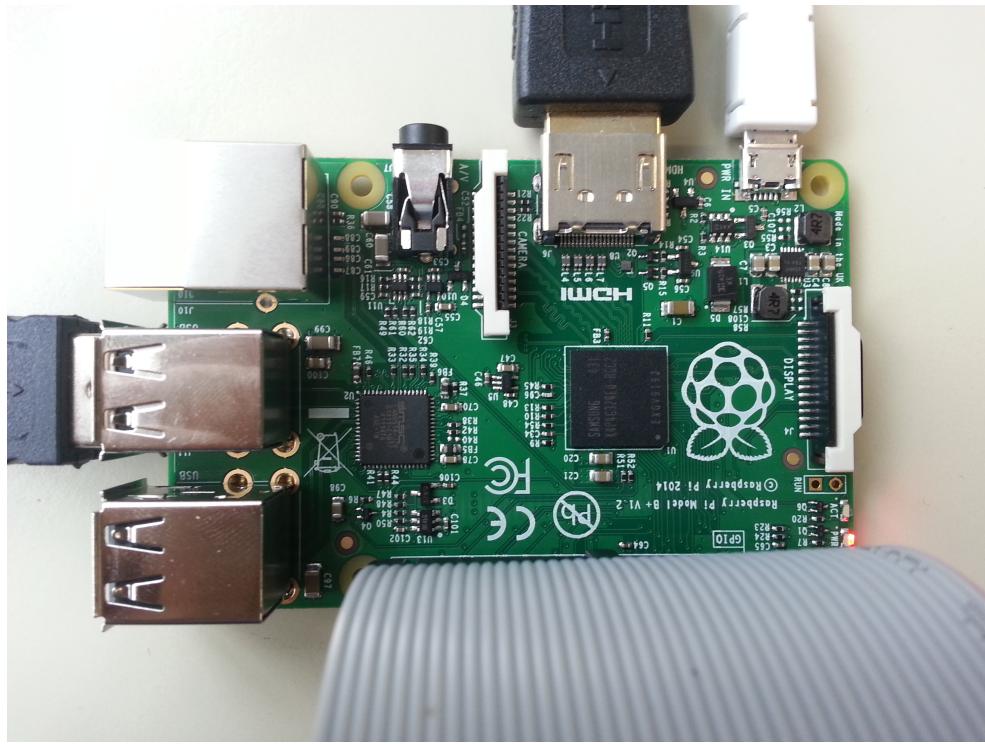


FIGURE 3.1 – Notre Raspberry Pi B+

Pour s'assurer que notre Raspberry Pi répond aux spécifications fonction-

nelles et qu'il fonctionne correctement en toutes circonstances pour notre projet, nous y avons réalisé des tests unitaires.

Après avoir enfin installé le système d'exploitation Raspbian¹ sur notre Raspberry Pi B+, nous avons tenté de tester les ports GPIO. Pour cela, dans un premier temps, nous avons allumé des LED grâce à un script python à travers différents ports GPIO. Sur la figure 3.2, on peut observer que nous avons allumé une LED grâce au port 22.

Dans notre projet le Raspberry Pi sera placé entre le radio-goniomètre à effet Doppler et l'utilisateur. Il aura deux tâches, corrélérer les données entre tous les dispositifs pour obtenir la position du drone et afficher le résultat à l'utilisateur. Pour cela il doit récupérer la direction qui est donné par le Montréal 3v2. Cette position est donnée à travers des LED (voir figure 3.3). Nous allons donc placer le Raspberry Pi au niveau des LED pour obtenir les informations délivré par le Montréal 3v2.

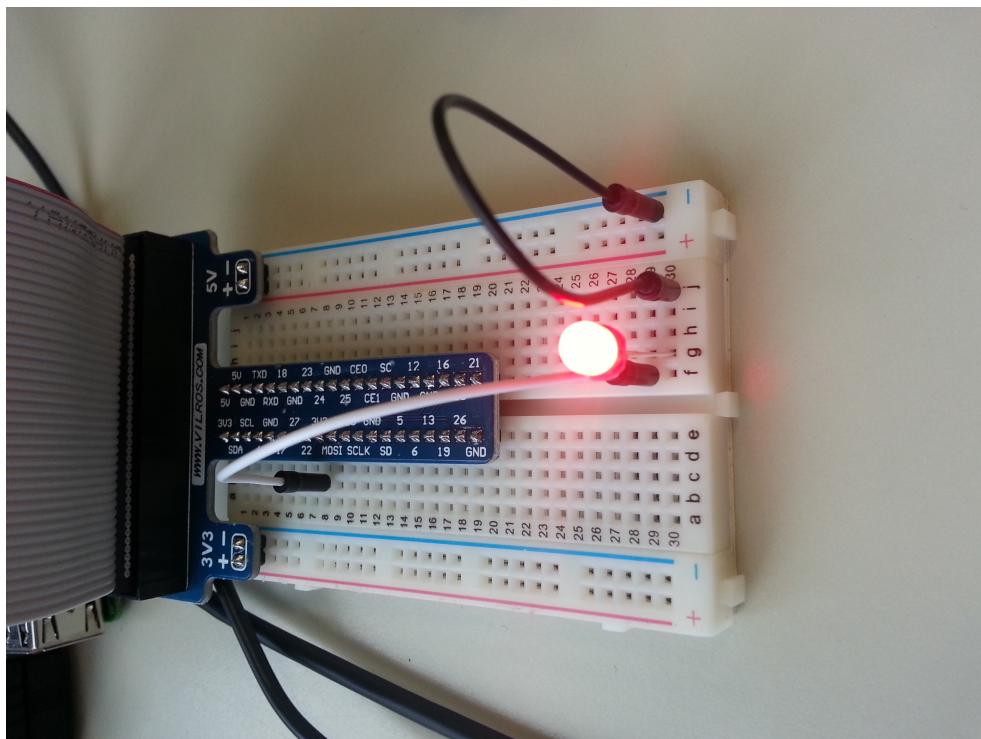


FIGURE 3.2 – Allumage d'une LED par Raspberry Pi

La sortie du Montréal 3v2 est décrite à la figure 3.3. On constate que le pic qui permet l'affichage à 12 sortie qui lui permet de gérer 24 LED. Pour connaître quelle LED est allumé, il faut savoir laquelle

1. La documentation lié à Raspbian est situé en annexe à la page 19

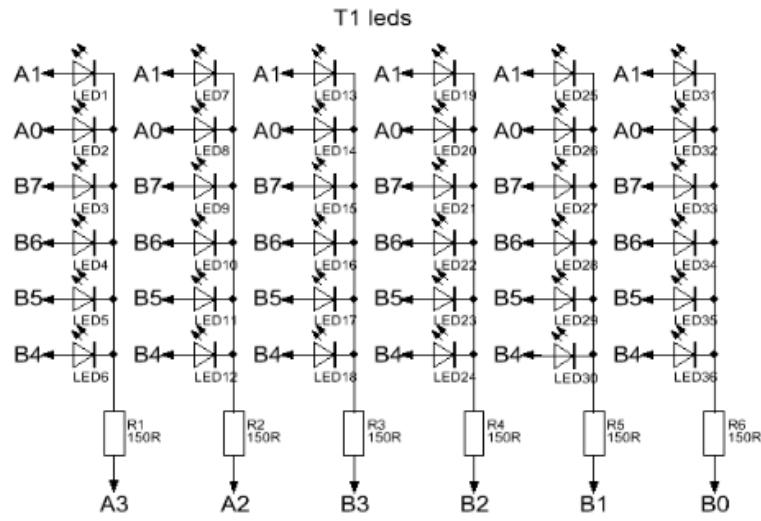


FIGURE 3.3 – Méthode de connexion des leds dans le Montréal

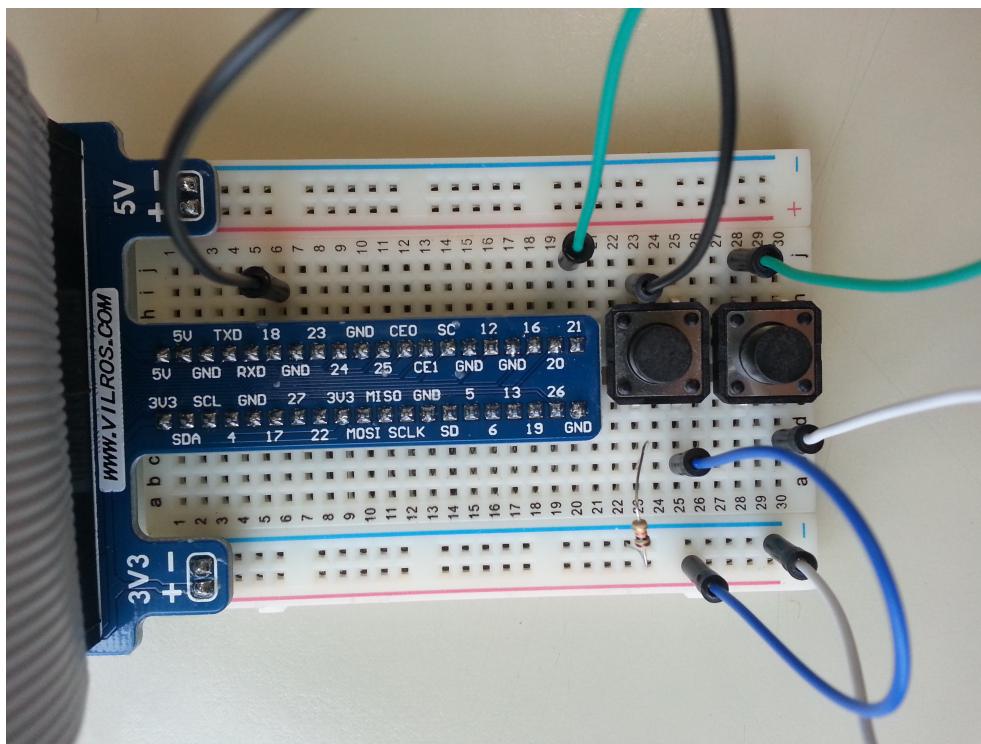


FIGURE 3.4 – Système modélisant une LED

des entré A1,A0,B7,B6,B5,B4 à un front montant et laquelle des entrées B0,B1,B2,B3,A3,A2 à un front descendant.

Pour modéliser une LED en entré du Raspberry Pi, nous avons positionné 2 boutons poussoirs (voir figure 3.4). Le premier permet de réaliser le front montant et le second le front descendant. Ainsi en positionnant ces boutons au bon endroit par rapport au port GPIO du raspberry il est possible de connaître quelle LED on a simuler.

Nous avons réalisé un script python qui lié les entrées du raspberry avec les sortie du pic. Puis nous avons tester en simulant une LED comme décrit précédemment.

On peut constater que l'expérience est un succès car le raspberry pi nous renvoie bien le numéro de la LED que nous voulions tester.

3.2 PIC

Dans le schéma du Montréal 3v2 nous avons pu constater qu'il y avait 3 PIC programmés. Nous avons commandé les PIC programmés au près de l'entreprise F1LVT [?].

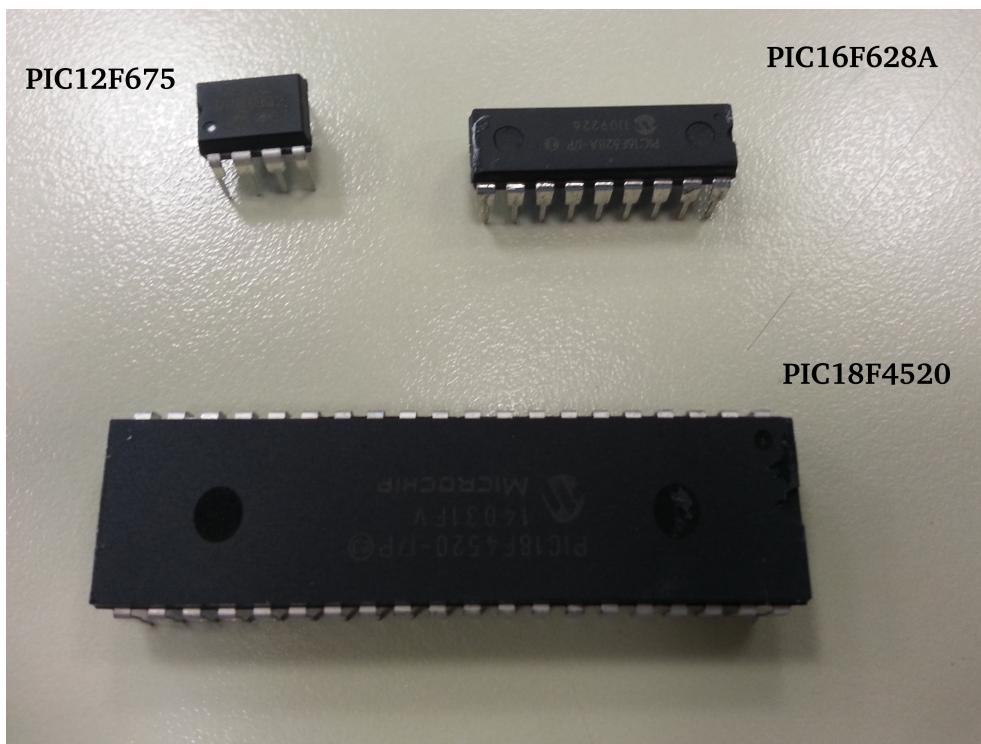


FIGURE 3.5 – 3 PIC programmés

Nous avons ensuite imaginé et réalisé des tests unitaires sur chacun des PIC pour vérifier qu'ils ont bien été programmés et qu'aucune erreur n'est

apparu sur ce système de décision critique pour le système.

PIC16F628A

Ce PIC sert à réaliser l'affichage sur les LED. Pour tester ce PIC, nous avons réalisé le montage de la figure 3.6. On peut voir à la figure 3.7 le schéma de montage du PIC sur le Montréal 3v2.

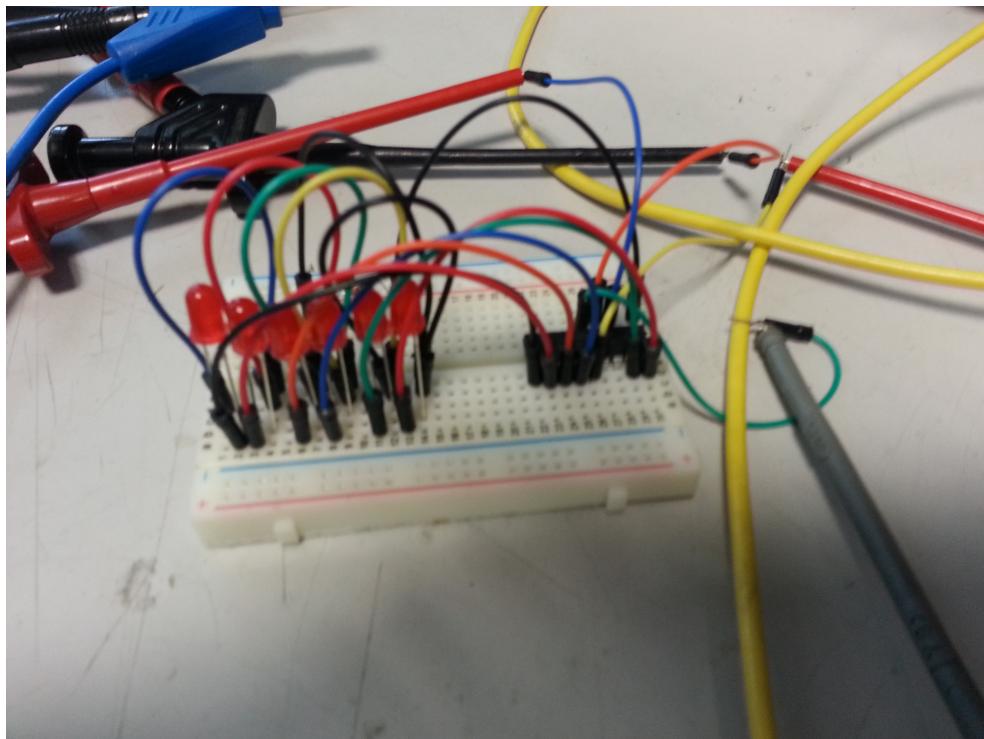


FIGURE 3.6 – Schéma électrique du test unitaire

Pour tester ce composant, nous avons donc choisi de monter une partie des LED situés en sorti, de configurer le « clock » sur un signal carré de fréquence 1 MHz, et de faire varier la fréquence de l'entrée « data ».

Malheureusement nous n'avons pas pu observer de LED s'allumer pendant notre expérience.

PIC12F675

PIC18F4520

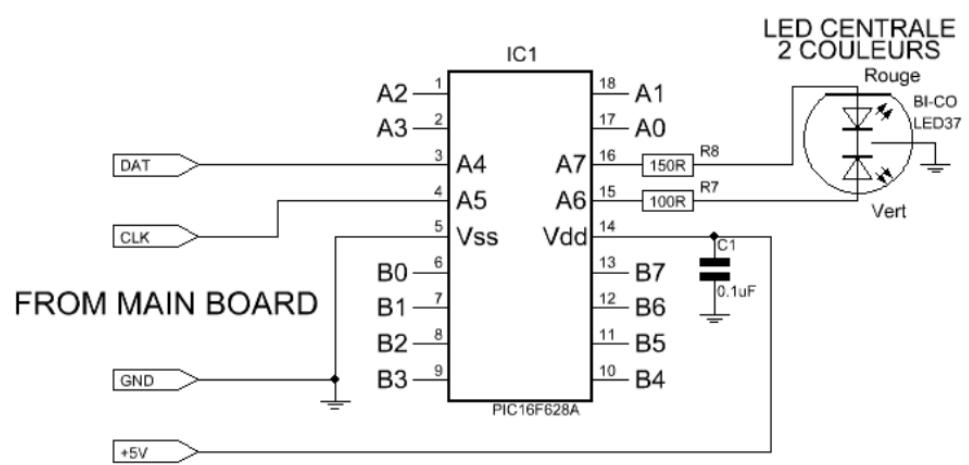


FIGURE 3.7 – Schéma de montage du pic sur le Montréal 3v2

CHAPITRE 4

Réalisation

4.1 Radiogoniomètre Doppler

4.2 Application Android : S.M.A.R.T Comm Center

Présentation Générale

Le système SMART se base sur un ensemble de capteurs reliés à une station centrale dont les données sont accessibles depuis internet. Cela permet d'accéder aux paramètres du système à distance depuis un autre appareil relié à internet de préférence un ordinateur. Toutefois, le système, étant donné son coût réduit se destine à être utilisé au sein de structures de petites taille ou le personnel en charge de la sécurité du site doit parfois se déplacer en fonction de ses autres obligations et ne peut être constamment en train de surveiller l'état du S.M.A.R.T depuis un ordinateur.

Pour pallier à ce manquement, il semblait intéressant de proposer une solution sur téléphone mobile qui permettrait d'avertir l'utilisateur final où qu'il se trouve. Deux options existent :

- Une version mobile de l'interface web
- Une application dédiée

Ces deux utiliseraient les APIs des différentes plateformes mobiles existantes (Windows phone, Android, iOS) pour avertir l'utilisateur en utilisant les fonctions vibrer ou la sonnerie du téléphone. Toutefois, les capacités du site mobile sont assez limitées car de nombreux éléments de sécurité peuvent restreindre l'accès à certaines fonctionnalités du téléphone comme l'accès au vibrer ou la possibilité de s'exécuter en tache de fond. De plus ces restrictions varient en fonction de la plateforme mobile visée.

L'application mobile a donc l'avantage d'offrir plus de latitude au développeur et d'implémenter plus facilement différents moyens d'alerte pour

l'utilisateur. Il faut cependant garder à l'esprit le fait que ce choix de développement implique de réaliser une application par système d'exploitation mobile existant.

Le choix final pour la version actuelle du système S.M.A.R.T à donc été celui de l'application mobile. Compte tenu des équipements dont nous disposons le système sur lequel l'application a été développé est Android. En effet, une grande partie du code source est sous licence GPL et le codage des application se fait en Java dans sa version 1.7. De plus, ce système d'exploitation mobile représente en Janvier 2016 64 % du parc mobile français.

Fonctionnement de l'application

Vue d'ensemble

Le système S.M.A.R.T utilise un serveur web pour gérer l'affichage des données à destination de l'utilisateur final, il est en mesure de créer des connexions vers plusieurs appareils distants. L'application jouera le rôle de client et recevra du serveur les information de position du drone en cas d'intrusion. Dans l'état actuel la réception des données de position par l'application se fait en mode "pull". Cela signifie que c'est l'utilisateur qui lance la demande d'information et le serveur répond ensuite à la requête.

Un mode automatique, avec un rafraîchissement régulier de l'information, a été codé et implémenté mais n'a pas été utilisé dans la version finale de l'application. Il est aussi possible de passer par un mode "push", où le serveur envoie l'information de lui même vers le client dès que celle-ci est mise à jour. Les raisons de ces choix technologiques seront détaillées par la suite.

Choix du niveau d'API

La version actuelle de S.M.A.R.T Comm Center a été développé avec un niveau d'API Android minimum de 19¹. Le choix d'un niveau aussi élevé d'API a été déterminé par trois éléments importants : la gestion des "threads", des tâches asynchrones et des socket. En effet depuis l'API 19 Google, qui édite et maintient le code d'Android, a modifié la façon dont les connexions réseau étaient gérés sous Android et le fonctionnement actuel qui sera détaillé par la suite nous convenait mieux.

Il faut cependant noter qu'un tel choix limite le nombre de smartphones qui seront en mesure de lancer l'application. L'API 22 par exemple représente seulement 34 % du nombre total d'appareils Android activés. Dans un souci de faciliter la réalisation de l'application nous avons maintenu ce choix, sachant que l'ensemble des téléphones pouvant exécuter du code écrit avec un niveau d'API de 19 représente en Janvier 2016 75.6% du nombre total des téléphones

1. L'API correspond à la version d'Android ciblée et détermine donc les fonctionnalités disponibles pour le développeur. le niveau 19 correspond à Android 4.4 KitKat.

Android activés dans le monde. Ce choix n'est donc pas si restrictif au vu du nombre d'appareils touchés (plus d'un milliard).

Achitecture du client

Les applications Android se basent sur un système d'"activités" et une activité correspond à une fenêtre visible par l'utilisateur. L'application en possède deux : La fenêtre d'accueil et la fenêtre d'affichage des données de localisation du drone.

- La fenêtre d'accueil ne comprends qu'un bouton permettant de lancer l'activité de localisation et donc le client web et un bouton pour quitter l'application, accompagnés du logo du projet. Un menu déroulant a été implémenté de façon à pouvoir modifier certains paramètres comme l'adresse et le port du serveur visé.
- La fenêtre de localisation est un peu plus complexe. Pour l'utilisateur, les trois éléments importants sont : les informations concernant l'état du système, le bouton "rafraîchir" et le bouton "retour".

Conclusion

Annexe

ANNEXE A

Documentation Technique du Raspberry Pi B+

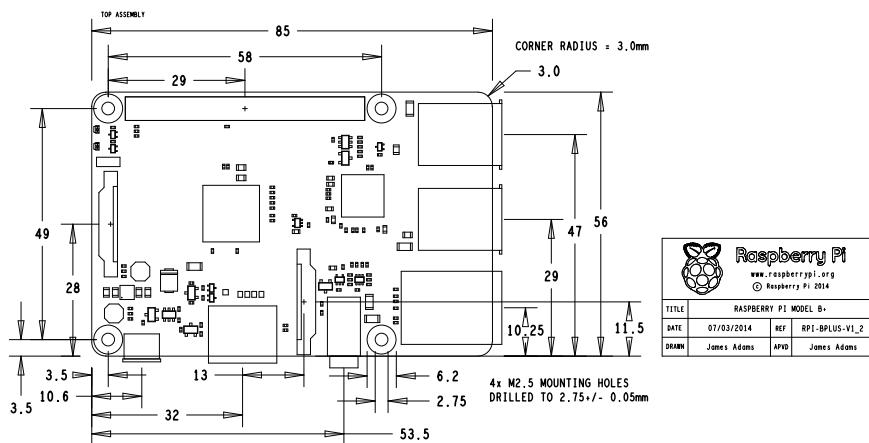


FIGURE A.1 – Dessin mecanique

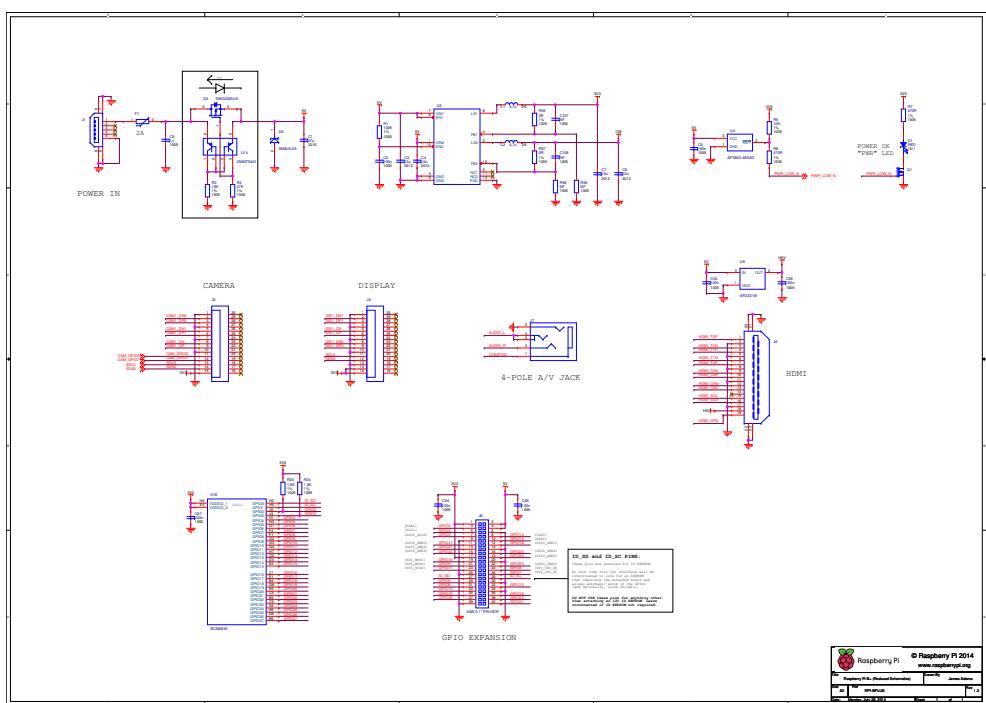


FIGURE A.2 – Schéma technique

Documentation Technique à Raspbian

Raspbian (recommended for Raspberry Pi 1) – is maintained independently of the Foundation ; based on the Debian ARM hard-float (armhf) architecture port originally designed for ARMv7 and later processors (with Jazelle RCT/ThumbEE and VFPv3), compiled for the more limited ARMv6 instruction set of the Raspberry Pi 1. A minimum size of 4 GB SD card is required for the Raspbian images provided by the Raspberry Pi Foundation. There is a Pi Store for exchanging programs.

The Raspbian Server Edition is a stripped version with fewer software packages bundled as compared to the usual desktop computer oriented Raspbian.

The Wayland display server protocol enables efficient use of the GPU for hardware accelerated GUI drawing functions.[104] On 16 April 2014, a GUI shell for Weston called Maynard was released.

PiBang Linux – is derived from Raspbian.

Raspbian for Robots – is a fork of Raspbian for robotics projects with Lego, Grove, and Arduino.

Table des figures

2.1	Diagramme pieuvre	4
2.2	Diagramme pieuvre	5
3.1	Notre Raspberry Pi B+	6
3.2	Allumage d'une LED par Raspberry Pi	7
3.3	Méthode de connexion des leds dans le Montréal	8
3.4	Système modélisant une LED	8
3.5	3 PIC programmés	9
3.6	Schéma électrique du test unitaire	10
3.7	Schéma de montage du pic sur le Montréal 3v2	11
A.1	Dessin mecanique	17
A.2	Schéma technique	18