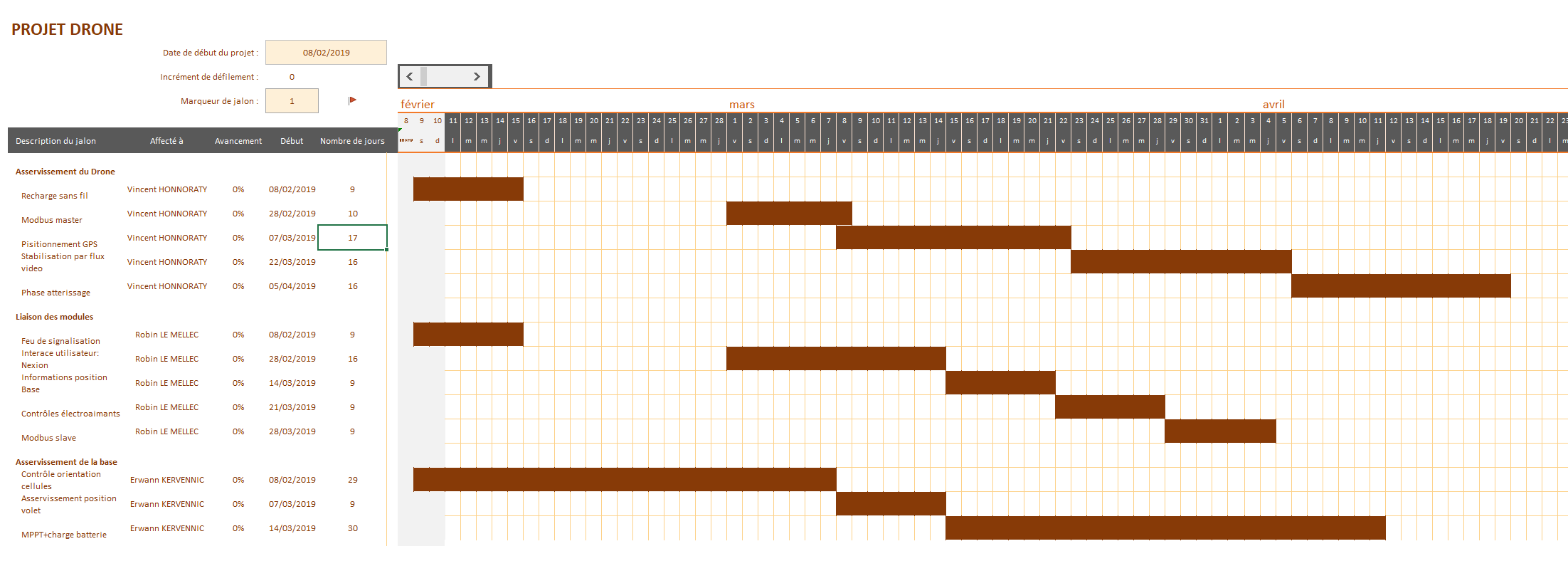
Attribution des taches : Projet drone de cartographie

# Distribution taches

|  |  |  |
| --- | --- | --- |
| Robin | Erwann | Vincent |
| Feu de signalisation | Contrôle orientation cellules | Recharge sans fil |
| Interface utilisateur : Nexion | Asservissement position volet | Positionnement : GPS |
| Informations position Base | **MPPT + charge batterie** | Stabilisation par flux vidéo |
| Contrôles électroaimants |  | Phase atterrissage |
| Modbus slave |  | Modbus master |



# Estimation du temps :

## Feu de signalisation :

|  |  |  |
| --- | --- | --- |
| **Tache** | | **Temps** |
| Programmation | | 3h |
| Routage PCB | | 4h |
| Cable-managing | | 2h |
|  | **Total :** | 9h |

### Description :

6 LEDs RGB signalant la position du drone, illuminées par défaut : 2 vertes vers l’avant, 2 rouges vers l’arrière, une bleue de chaque côté.  
Enregistrer différent thème sélectionnable depuis la base Programme fonctionnant sur un microcontrôleur, qui contrôle un driver de LED, communique via protocole série avec le Raspberry pour la sélection du mode.

## Interface utilisateur :

|  |  |  |
| --- | --- | --- |
| **Tache** | | **Temps** |
| Design graphique | | 4h |
| Programmation | | 5h |
| Routage | | 3h |
| Tests | | 2h |
|  | **Total :** | 14h |

### Description :

Interface graphique basée sur un écran Nexion (téléchargé logiciel propriétaire gratuit pour dessiner, designer l’interface graphique). L’utilisateur sélectionnera le mode de signalement du drone, ajoutera des points « d’intérêt » au format de coordonnées GPS pris sur par exemple sur google map + altitude relative au sol.

Le microcontrôleur lié à celui-ci fera l’interface Modbus en contenant dans ses registres :

* Nombre de points d’intérêt
* Coordonnées pour chaque au format (latitude, longitude, altitude)
* Registre d’état en lecture seule : exemple 0 drone en attente, 1 drone doit aller au point courant, 2 le drone doit revenir à la base
* Registre d’état du drone en écriture seule : exemple 0 drone en attente, 1 drone est en train d’aller au point, 2 le drone est au point, 3 le drone est à la base
* Registre de position courante écriture seule : le drone renvoie à quel point de la liste il est

Ce microcontrôleur gèrera le GPS et le capteur de pression atmosphérique de la base pour déterminer la position relative de la base par rapport au drone

* Registres de position de la base uniquement par GPS (latitude, longitude, altitude)
* Registre d’altitude

## Informations position base

|  |  |  |
| --- | --- | --- |
| **Tache** | | **Temps** |
| Programmation | | 3h |
| Routage | | 3h |
|  | **Total :** | 6h |

### Description :

Même microcontrôleur que celui de d’interface graphique, MODBUS avec les registres définis suivant :

* Registres de position de la base uniquement par GPS (latitude, longitude, altitude)
* Registre d’altitude

## Contrôle électroaimants

|  |  |  |
| --- | --- | --- |
| **Tache** | | **Temps** |
| Programmation | | 3h |
| Routage | | 4h |
|  | **Total :** | 7h |

### Description :

Microcontrôleur gérant l’activation de MOSFET pour alimenter les électroaimants en fonction de l’ordre courant reçus par MODBUS, registres :

* Registre état courant (lecture/écriture) : 0 éteint, 1 activé, 2 plus de batterie : stop

Mesure de la tension des batterie, stop l’alimentation si tension inférieure à 22v.

## Modbus slave

|  |  |  |
| --- | --- | --- |
| **Tache** | | **Temps** |
| Programmation | | 6h |
| Routage | | 3h |
|  | **Total :** | 9h |

### Description :

Module de la base relié par MODBUS, utilisation du protocole, mais implémentation hardware revisité, les modules reçoivent toutes les mêmes informations du maitre via le récepteur Lo-Ra, mais écrivent sur une ligne 3 états.

Chaque module a un étage d’adaptation 3 états (0v, 5v, high Z) avec un 4066, puisqu’ils écrivent en différé étant donné que c’est le maitre qui donne les ordres d’écriture sur le bus pour récupérer l’état des registre. Donc quand le microcontrôleur devra écrire, passera la ligne en état définie via l’activation du 4066 qui lui est propre. (Un seule 4066 activer à la fois.)

## Contrôle orientation cellules

|  |  |  |
| --- | --- | --- |
| **Tache** | | **Temps** |
| Mathématique : algèbre | | 6h |
| Programmation | | 8h |
| Routage | | 3h |
|  | **Total :** | 17h |

### Description :

Déterminer les équations définissant l’orientation des volets pour qu’il soit toujours parallèle à un même plan (donnée par une volet ayant le capteur de position solaire).

Implémentation de l’équation, puis implémenter le tracker solaire.

Monter le tracker solaire (souder photorésistance).

Devra être commander par MODBUS :

* Registre d’état en écriture : 0 doit être en mode repos (volet vers le bas), 1 mode limité (quand le drone est sur la base), 2 trackages complet du soleil
* Registre d’état courant réel : 0 doit être en mode repos (volet vers le bas), 1 mode limité (quand le drone est sur la base), 2 trackages complet du soleil

(Ce microcontrôleur contiendra le programme pour asservir les volets en position)

## Asservissement position volet

|  |  |  |
| --- | --- | --- |
| **Tache** | | **Temps** |
| Programmation | | 3h |
| Routage | | 4h |
|  | **Total :** | 7h |

### Description :

Asservir en position les motoréducteurs des volets avec potentiomètre : utilisation de PID, tous dans un seul microcontrôleur, celui qui s’occupe du contrôle des cellules.

Souder les L298 avec les fils des moteurs, d’alimentation et qui vont au microcontrôleur.

Minimiser le poids !!

## MPPT + charge batterie

|  |  |  |
| --- | --- | --- |
| **Tache** | | **Temps** |
| Se renseigner sur internet | | 4h |
| Tester sur breadboard l’effet boost | | 4h |
| Programme pour tester | | 3h |
| Programme finale | | 4h |
| Routage | | 3h |
| Test de fonctionnement | | 2h |
| Déterminer rendement | | 1h |
|  | **Total :** | 21h |

### Description :

Module très important pour optimiser les cellules solaires, et très sensible car directement branché au batterie Li-Po.

Comprendre la recharge de batterie Li-Po et ses DANGERS !!

Comprendre comment tirer le plus de puissance d’un panneau solaire : courbe caractéristique I-V, cadrant 4.

(BMS pour équilibrage de la charge déjà en place sur la base, protection sur-courant à 21A en utilisation prolongée.)

Il devra charger à courant constant (et maximiser) les batteries via un convertisseur boost.

18 cellules solaires de 2.8W, 0.5V, soit ~9v

Batterie Li-Po 6S soit 22.2V nominal, 25.2V MAXIMUN, 5.800 mAh x 2

## Recharge sans fil

|  |  |  |
| --- | --- | --- |
| **Tache** | | **Temps** |
| Expérimentation sur breadboard | | 6h |
| Programme | | 3h |
| Routage | | 2h |
| Bobinage | | 1h |
|  | **Total :** | 12h |

### Description :

Expérimenter la transmission sans fil d’énergie avec émetteur/récepteur LC :

* Choisir topologie émettrice : classe C, pont complet, demi pont
* Calculer capacité complémentaire pour la fréquence choisie
* Design du récepteur : diode rapide + redressage + convertisseur pour recharge des batteries Li-Po

Déterminer rendement en fonction de la distance (dessiner courbe : P\_emise/P\_reçue).

Programmer sur microcontrôleur la gestion de l’émetteur et du récepteur, un sur la base, l’autre dans le drone : mesure courant/tension, arrêt si charge de batterie trop faible, gérer par MODBUS.

Bobiner dans les tubes PVC du drone et de la base les bobines optimisées trouvé expérimentalement.

## Positionnement : GPS

|  |  |  |
| --- | --- | --- |
| **Tache** | | **Temps** |
| Etude théorique | | 3h |
| Programmation | | 3h |
| Testes | | 2h |
|  | **Total :** | 8h |

### Description :

Asservissement en position du drone via coordonnée GPS de destination et coordonnée GPS réel.

Sera programmé en python sur le raspberry pi, et communiquera avec le Naze 32 via liaison série.

3 PID : longitude, latitude et altitude pour corriger la position du drone.

(Déplacement en 2 phase : déplacement grossier via GPS, puis une fois proche du point voulu, stabilisation par flux vidéo.)

## Stabilisation par flux vidéo

|  |  |  |
| --- | --- | --- |
| **Tache** | | **Temps** |
| Etude théorique | | 3h |
| Programmation | | 4h |
| Testes | | 5h |
|  | **Total :** | 12h |

### Description :

Correction des micromouvements, permettant une stabilisation 2D améliorée par rapport à un drone simplement stabilisé par gyroscope/accéléromètre.

Caméra sous le drone prend des images à intervalle régulier (dizaine par seconde) du sol, à l’aide d’une librairie de traitement d’image, on extrait le mouvement moyen entre chaque image.

Le mouvement moyen est ensuite séparé en 2 composantes et injecter dans 2 PID, dont leur résultat est envoyé au Naze 32 pour corriger le mouvement.

Pour l’altitude, l’asservissement est fait par un capteur de pression atmosphérique qui contrôle la puissance totale des moteur (poussé globale) via un PID.

## Phase atterrissage

|  |  |  |
| --- | --- | --- |
| **Tache** | | **Temps** |
| Etude théorique | | 3h |
| Programmation | | 4h |
| Testes | | 5h |
|  | **Total :** | 12h |

### Description :

Objectif que le drone reconnaisse un dessin (une croix) sur la base, s’aligne dessus, et se pose correctement dessus.

Phase 1 : Le drone connait les coordonnée GPS de la base, il s’en rapproche alors grossièrement via coordonnée GPS.

Il connait aussi sont altitude relative par les capteurs de pression atmosphérique présent sur la base et le drone.

Phase 2 : Caméra sous le drone prend des images à intervalle régulier (dizaine par seconde) du sol, à l’aide d’une librairie de traitement d’image, on recherche la présence du dessin reconnaissable.

Une fois trouvé le drone corrige ça position pour s’aligner sur la croix en corrigeant en 2D, cette correction peu être faite aussi des PID qui compare la position courante de la croix avec la position finale (centré au milieu de l’image par exemple).

Phase 3 : Une fois aligné, le drone descente toute en continuant à viser la croix via le même algorithme que la première phase.

Phase 4 : Electroaimant activé, et arrêt complet des moteurs.

## Modbus master

|  |  |  |
| --- | --- | --- |
| **Tache** | | **Temps** |
| Programmation | | 3h |
| Routage | | 3h |
|  | **Total :** | 6h |

### Description :

Communication entre les modules, établir l’adressage de tous les modules, écrire fonction de lecture et écriture de registre des carte esclaves. Mettre en place communication sans fil avec module Lo-Ra.

# Estimation temps totaux :

|  |  |  |  |
| --- | --- | --- | --- |
|  | Robin | Erwann | Vincent |
| Temps totaux : | 45h | 45h | 50h |

## Nombre de séances disponibles :

**15 séances de 6h -> 90h**