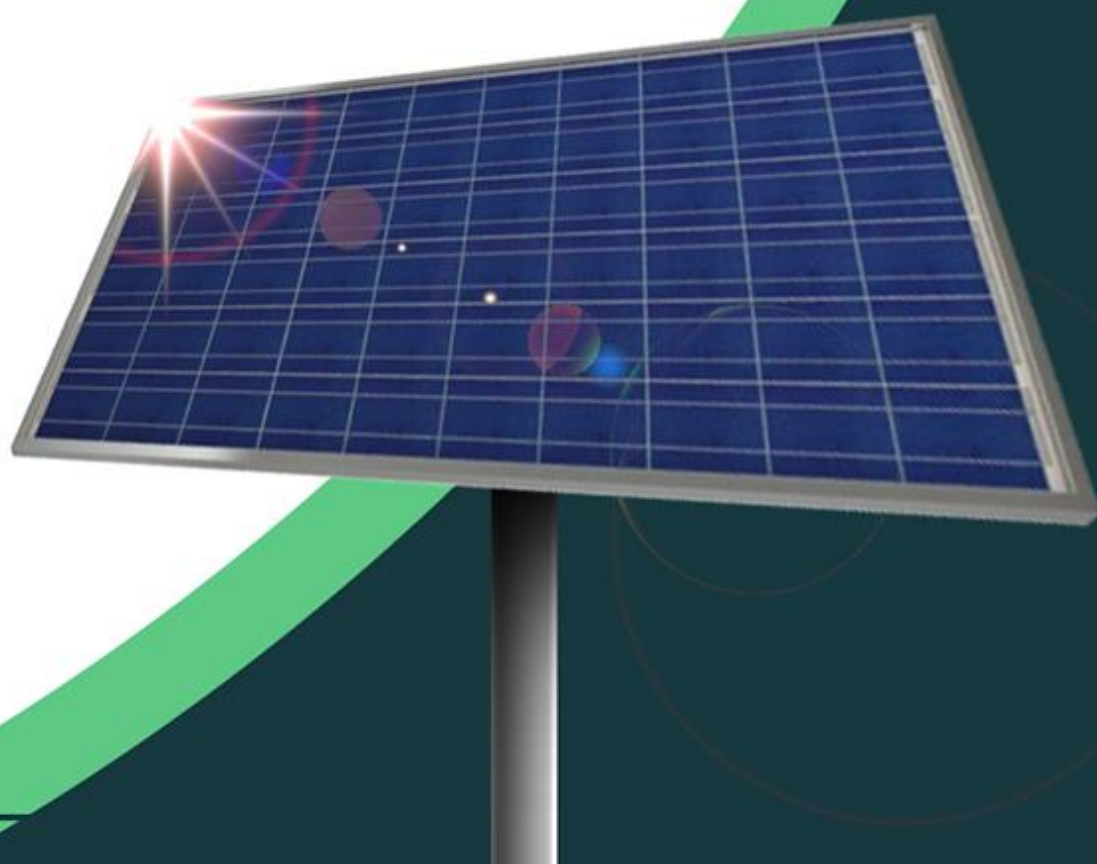


# TRACKER SOLAIRE

Licence Professionnelle Ecologie Industrielle

2018-2019

Professeur tuteuré: M. Larramendy



BEAUDOUIN Pierce  
BOUGHENOUT Joan  
CITERIN Lucas  
HANQUEZ Florian

# Table des matières

Remerciements .....	3
Résumé .....	4
Abstract .....	5
Introduction .....	6
Présentation générale du projet.....	7
Analyse de l'existant : .....	7
Objectifs à réaliser : .....	8
Fonctionnement d'une installation photovoltaïque en site isolé : .....	9
Fonctionnement d'un panneau photovoltaïque : .....	10
Fonctionnement sans électronique de puissance : .....	10
Fonctionnement avec électronique de puissance : .....	11
Partie électrique du projet.....	12
Etude du circuit de mesure .....	12
Fonctionnement d'une carte de mesure : .....	14
Test mesures : .....	18
Récupération et traitement des données : .....	19
Communication avec le Raspberry : .....	21
Supervision .....	23
Partie mécanique du projet.....	26
Conception du boîtier .....	26
Annexes .....	33
Lexique .....	33
Diagramme de Gant.....	34
.....	34
Tableaux de structuration des données des carte de mesures .....	35
Tutoriel Pyscada .....	37
Méthode pour modifier un script : .....	37
Méthode pour créer un graphique : .....	38
Méthode pour retrouver l'adresse d'une carte : .....	41
Guide de montage SolidWorks .....	42
Numérotation des plaques pour le sens du montage.....	42
Liste des vis et écrous pour le guide de montage .....	43
Composants du tracker solaire .....	54
Langage de communication Python .....	55

## Remerciements

Le projet arrivant à son terme nous souhaitons exprimer notre gratitude à toutes les personnes qui nous ont apporté leur aide et qui ont contribué à la réussite de ce projet.

Nous remercions également l'IUT de Bayonne et des pays de l'Adour qui nous a permis de travailler dans les conditions optimales pour réussir ce projet ainsi que pour nous avoir fourni le matériel nécessaire et nous avoir facilité l'accès à la salle où se situe le tracker.

Pour finir, nous adressons nos remerciements à notre tuteur Monsieur Larramendy pour nous avoir guidé tout au long du projet et apporté son aide pour nous aider à mener ce projet à son terme.

## Résumé

Notre groupe de projet composé de : Pierce Beaudouin, Joan Boughenout, Lucas Citérin et Florian Hanquez, avait en charge le projet du tracker photovoltaïque qui était une reprise des projets des autres années. Durant ces 7 mois de projet, nous avons pu mettre en pratique nos compétences théoriques sur le fonctionnement d'un panneau photovoltaïque ainsi que nos compétences sur SolidWorks pour la conception du boîtier du tracker.

Nos objectifs étaient les suivants :

- Concevoir un nouveau boîtier capable de contenir les différents éléments nécessaires au fonctionnement du panneau. Ce nouveau boîtier permettra de supprimer l'armoire électrique afin de réduire le nombre de câbles et de permettre la mobilité du panneau.
- Étudier les cartes de mesure et définir leur emplacement afin de pouvoir récolter les données nécessaires pour avoir un suivi du point optimal de fonctionnement du panneau.
- Réaliser une interface web sur le logiciel PyScada affichant les données en temps réel du panneau.

Pour conclure, nous avons pu développer et améliorer nos compétences techniques, acquérir de nouvelles compétences et connaissances notamment dans le domaine du photovoltaïque. Cela nous a permis d'appliquer nos compétences acquises lors de notre cursus.

## Abstract

Our project group: Beaudouin, Joan Boughenout, Lucas Citérin and Florian Hanquez, Pierce had supported the project of the solar tracker which was a cover of the projects from other years. During these 7 months of project, we were able to put into practice our theoretical skills on operating a photovoltaic panel and our skills on SolidWorks for the design of the housing of the tracker.

Our objectives were to:

- develop a new housing able to contain the different elements necessary for the functioning of the Panel. This new box will remove the electrical cabinet to reduce the number of cables and allow the mobility of the Panel.
- Study cards and define their location in order to collect the data needed to have a follow-up to the optimal operating point of the Panel.
- Make a web interface on the software PyScada displaying the data in real time to the control panel.

To conclude, we were able to develop and improve our technical skills, acquire new skills and knowledge, including in the field of photovoltaics. This allowed us to apply our skills acquired during our course.

# Introduction

Le Département GIM dispose d'un panneau photovoltaïque monté sur un support orientable sur deux axes (angle d'élévation et angle d'azimut).

Ce dispositif permet l'alignement de la normale du panneau photovoltaïque avec le soleil afin d'optimiser la production d'énergie de l'installation.

Le tracker s'oriente selon la position du soleil grâce à deux moteurs, l'un contrôlant la rotation selon l'axe horizontal l'autre contrôlant la rotation selon l'axe vertical. Les moteurs sont programmés pour suivre la position du soleil quel que soit l'heure ou la saison grâce à un micro-ordinateur.

Notre groupe de projet composé de : Pierce BEAUDOUIN, Joan BOUGHENOUT, Lucas CITÉRIN et Florian HANQUEZ, nous avons pour objectif durant ce projet tutoré de poursuivre le travail déjà effectué sur le tracker photovoltaïque. L'objectif final du projet était de pouvoir placer le tracker solaire sur le toit de l'IUT afin que les étudiants du département GIM puisse réaliser des travaux pratiques.

Comme nous l'avons dit précédemment le projet a été entamé par les groupes des années précédentes. Le groupe de l'année dernière a finalisé la partie concernant le suivi du soleil par le tracker solaire. Cette année, nos objectifs étaient de concevoir un nouveau boîtier mécano-soudé qui permettrait de rendre le tracker plus mobile. Mais aussi de réaliser l'installation de la partie mesure et le développement d'une supervision pour tracker photovoltaïque.

Après avoir pris connaissance du sujet et de nos principaux objectifs lors de la première séance de projet, nous avons mis en place un emploi du temps prévisionnel des différentes tâches que nous allions devoir réaliser au cours du projet. Nous avons donc réalisé un diagramme de Gant (**voir annexes**) sur le logiciel Project 2016 pour organiser au mieux le travail que nous allions devoir réaliser au cours du projet. De plus, nous avons également créé une adresse mail ([18.19ptlpei.trackerpv@gmail.com](mailto:18.19ptlpei.trackerpv@gmail.com)) pour le groupe nous permettant de partager, entre nous sur un drive, tous les documents nécessaires à l'avancement du projet ainsi que de communiquer avec notre tuteur Monsieur Larramendy en cas de question ou d'un besoin d'un document. Nous avons également attribué le rôle de chacun dans le groupe : Lucas serait le chef de projet et Joan le responsable de communication.

## Présentation générale du projet

Comme nous l'avons évoqué précédemment, notre groupe reprend le travail d'anciens groupes de projet pour poursuivre l'objectif final du projet qui est de placer le tracker solaire sur le toit du bâtiment de l'IUT afin que les élèves du DUT GIM puissent faire des travaux pratiques.

### Analyse de l'existant :

Le projet a été débuté lors de l'année universitaire 2013-2014, de nombreuses tâches ont déjà été réalisées par les anciens groupes de projet notamment sur la partie électrique. Ils ont réalisé :

- L'équation solaire qui permet au tracker de suivre la position du soleil à n'importe quel moment de la journée. Cela permet d'augmenter le rendement du panneau. (Début 2013-2014 fin 2017-2018)
- La conception d'un boîtier d'étanchéité pour les servomoteurs. (2017-2018)
- L'armoire électrique indépendante du panneau. (2013-2014)
- La mise en place du Raspberry qui est un micro-ordinateur qui nous permet de faire l'acquisition des mesures à travers des cartes et de communiquer à distance grâce à une supervision. (2014-2015)
- La programmation en Python. (2014-2015)
- Une interface WEB qui a été abandonnée car le langage de programmation était laborieux à apprendre et le temps passé pour réaliser une fonction était trop élevé pour être satisfaisant. (2015-2016)

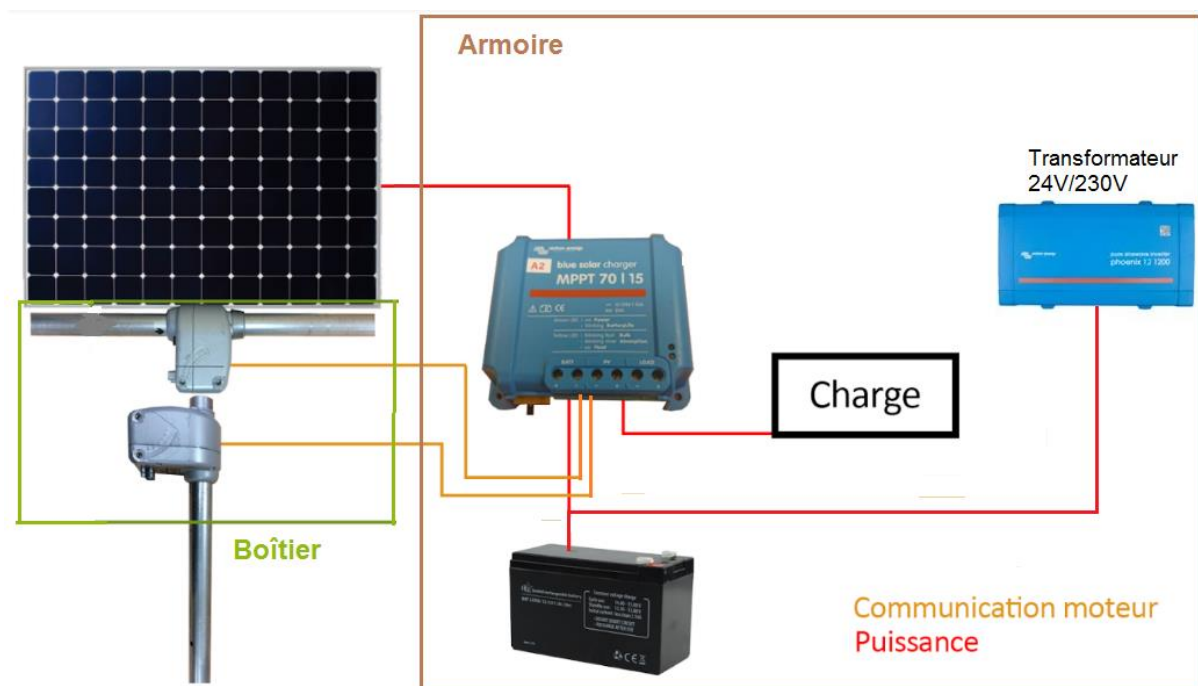


Figure 1 : ancienne installation

## Objectifs à réaliser :

Après la première séance de projet nous avons défini nos objectifs principaux pour le projet :

- Concevoir un nouveau boîtier plus étanche qui doit contenir les deux servomoteurs, le Raspberry et le régulateur de charge (qui se trouvaient initialement dans l'armoire électrique) et nous permettre de supprimer cette dernière.
- Cette suppression doit nous permettre de diminuer le nombre de câbles sortants du panneau, de pouvoir se connecter en WI-FI grâce au Raspberry et rendre le panneau mobile.
- Définir les emplacements des cartes de mesures et les installer afin de récolter des données du tracker.
- Réaliser une interface de supervision, affichant les données nécessaires et établir des commandes afin de prendre en main en mode manuel le tracker à distance.

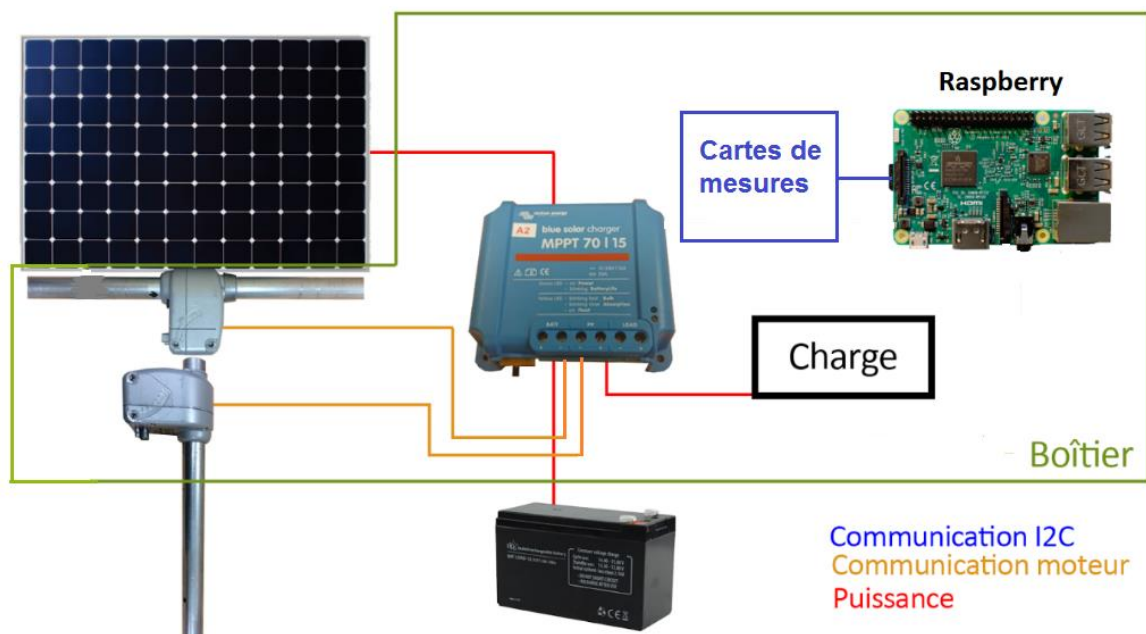


Figure 2 : nouvelle installation

Pour réaliser ces objectifs principaux nous devons auparavant réaliser différentes tâches :

- D'un point de vue mécanique, afin de supprimer l'armoire électrique nous devons dans un premier temps déterminer l'encombrement des différents éléments que devra contenir le boîtier en les modélisant sur SolidWorks. Pour la conception du boîtier nous devons penser à différentes contraintes comme l'étanchéité du boîtier, veiller à ce que le boîtier ne gêne pas la rotation du panneau lorsque celui-ci s'incline complètement à la verticale. Pour finir, Monsieur Larramendy nous a informé que le boîtier de l'année dernière était trop grand, c'est pourquoi nous devons concevoir notre boîtier le plus ergonomiquement possible. Nous devons également penser à une conception simple pour monter et démonter le panneau dans le cas d'une éventuelle maintenance.



- D'un point de vue électrique, nous allons devoir étudier le fonctionnement du circuit de mesure pour bien comprendre celui-ci, puis nous allons devoir définir l'emplacement des points de mesure utiles. De plus, nous devons réaliser l'interface de supervision qui affichera les données comme la puissance, l'énergie du panneau en temps réel.

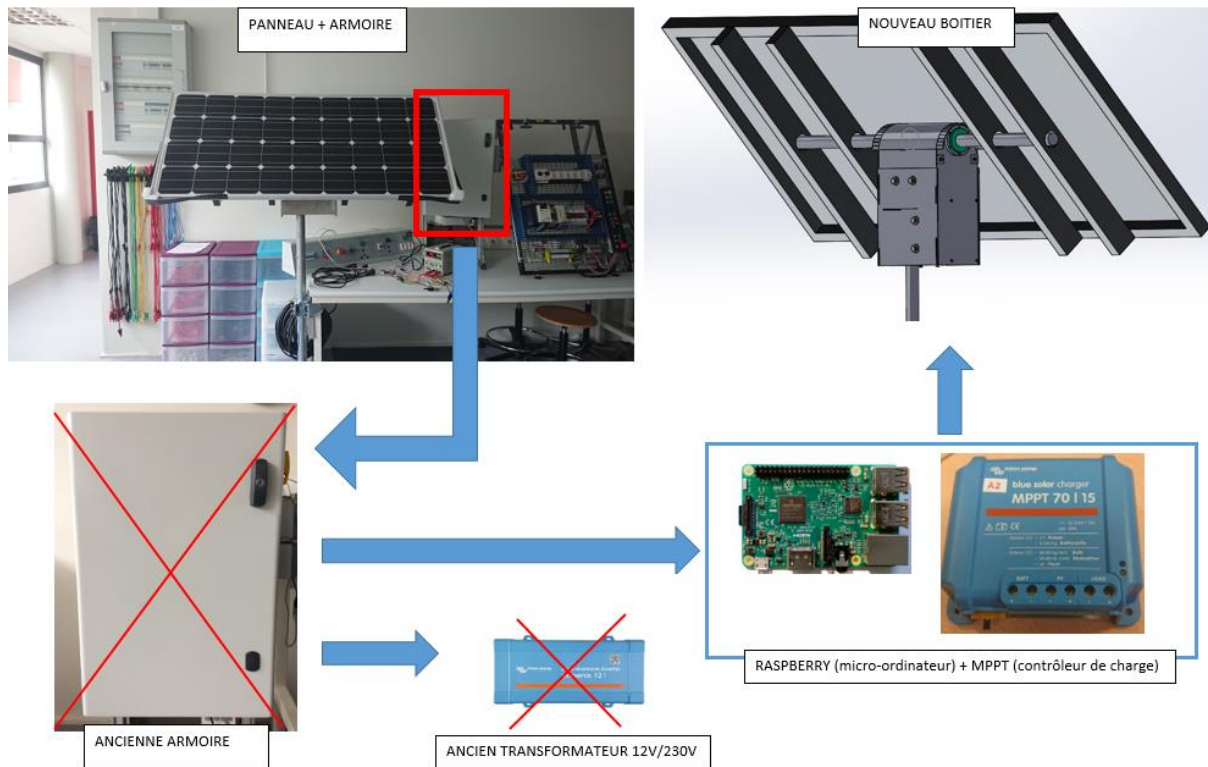


Figure 3 : passage de l'ancienne installation à la nouvelle installation

## Fonctionnement d'une installation photovoltaïque en site isolé :

Une installation photovoltaïque en site isolé est composée principalement de 4 parties :

- Le panneau solaire, son rôle est la production d'électricité il transforme l'énergie rayonnante reçue en énergie électrique.
- Le régulateur de charge, son rôle est de maîtriser la charge et la décharge de la batterie et d'utiliser le panneau au maximum de sa puissance.
- La batterie, son rôle est de stocker l'énergie reçue par le panneau et de la restituer lorsque le panneau ne peut pas subvenir au besoin.
- L'onduleur, son rôle est de convertir le courant continu en alternatif pour pouvoir être utilisé dans les différents appareils électriques.

## Fonctionnement d'un panneau photovoltaïque :

Il existe différents types de panneaux photovoltaïques, monocristallin, polycristallin et amorphe, le plus intéressant étant le monocristallin avec un rendement pouvant aller jusqu'à 16% (en moyenne 12%).

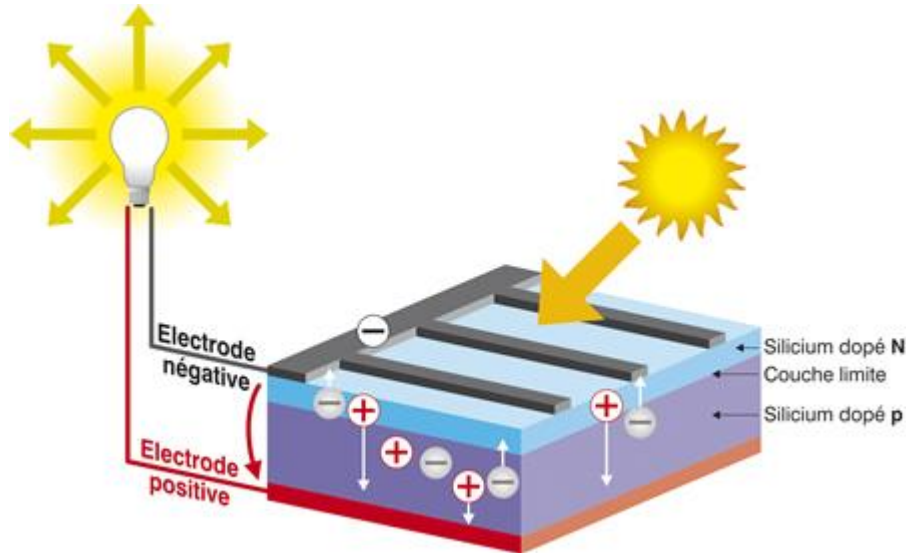


Figure 4 : schéma de fonctionnement panneau photovoltaïque

Le panneau est composé de deux couches de silicium : une couche dopée P (chargée positivement) et une couche dopée N (chargée négativement).

Lorsque le panneau va recevoir les photons solaires cela aura pour effet de décrocher un électron à un atome de silicium. L'ion silicium chargé positivement dans la zone N se dirige vers la zone P et l'électron reste dans la zone N : cela crée une différence de potentiel.

## Fonctionnement sans électronique de puissance :

Le panneau photovoltaïque est installé avec une batterie qui lui impose sa tension et lui permet de stocker l'énergie capturée, il y a aussi une diode pour éviter qu'il puisse recevoir du courant. Cependant le courant et la puissance varient en fonction de la météo et de l'orientation du panneau, ce qui rend le rendement du panneau très faible.

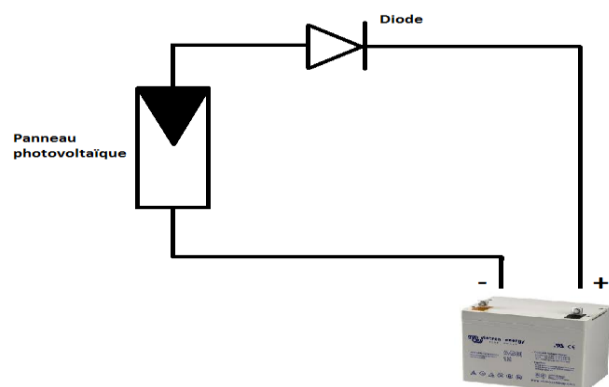


Figure 5 : schéma de fonctionnement sans électronique de puissance

## Fonctionnement avec électronique de puissance :

L'électronique de puissance (dans notre cas le **MPPT**, description en annexe), permettant d'optimiser le rendement du panneau photovoltaïque (orientation du panneau et adaptation à la météo). Il permet de trouver en temps réel, le point optimal de fonctionnement.

Le point optimal de fonctionnement du panneau photovoltaïque, n'est atteint que dans une certaine configuration, et pour l'atteindre il faut pouvoir faire varier la tension de la batterie, afin d'obtenir la puissance maximale car en courant continu la puissance est le produit du courant et de la tension ( $P=UI$ ).

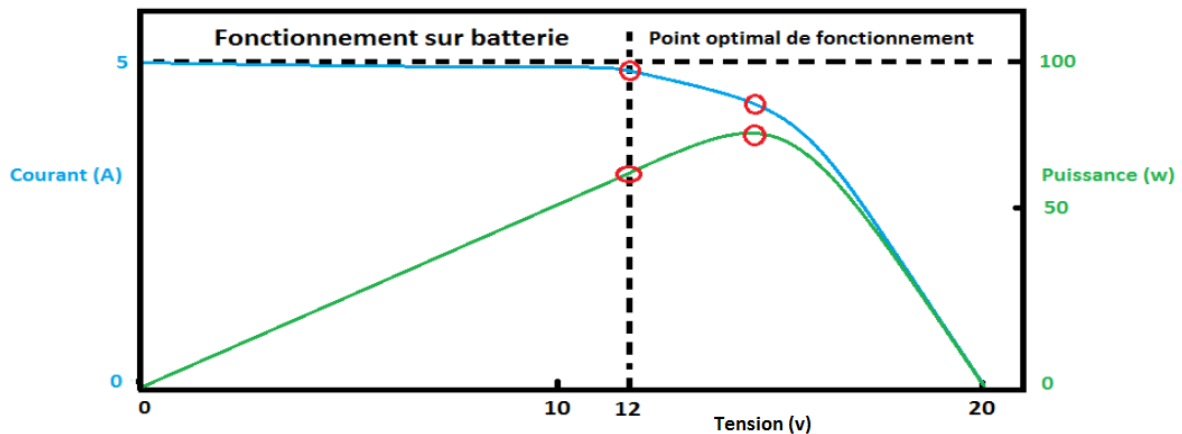


Figure 6 : recherche point optimal de fonctionnement

Maintenant nous avons besoin d'intégrer un algorithme (MPPT), pour calculer, en temps réel, le meilleur coefficient pour que notre installation fonctionne à son point optimal de fonctionnement.

## Partie électrique du projet

### Etude du circuit de mesure

Pour un meilleur suivi des consommations, nous avons besoin d'effectuer plusieurs mesures de tension, courant, puissance, en différents points du circuit, en particulier entre les principaux composants (panneau, MPPT, batterie...), ces mesures seront ensuite remontées et traitées par la supervision. En tout pour le projet, cinq cartes de mesures seront nécessaires, nous pourrons grâce à la loi des nœuds retrouver par calcul la 6<sup>ème</sup> valeur.

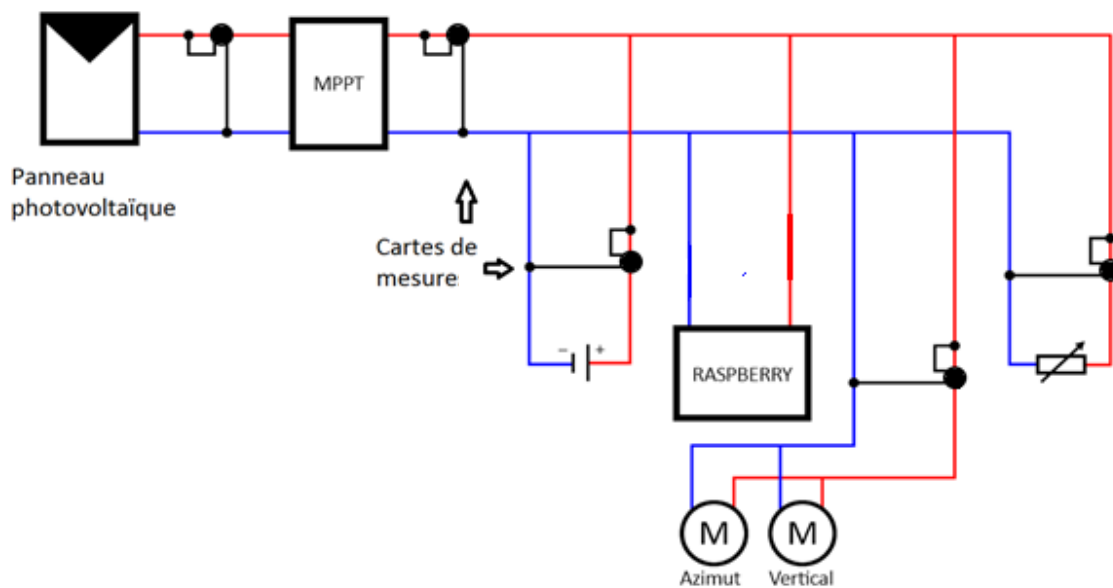


Figure 7 : placement des points de mesures

Points de mesure :

Pour effectuer ces mesures, nous utilisons plusieurs cartes de mesure identiques, placées en différents points et toutes reliées au Raspberry : **Linear Technology, LTC 2946**

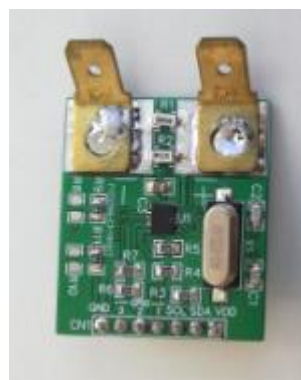


Figure 8 : carte de mesure LTC 2946

Ces cartes vont communiquer, avec le Raspberry, en remontant différentes informations (courant, tension, énergie...) permettant de les archiver dans la supervision.

Elles vont communiquer avec le Raspberry grâce au Bus I2C, nous permettant ainsi de mettre jusqu'à 127 cartes de mesures différentes.

Les cartes ne doivent pas avoir la même adresse, nous pouvons modifier l'adresse d'une carte en changeant **quatre résistances** servant de shunt (0 ohm) (R8, R9, R10, R11), ce qui permet de varier l'adressage, que nous pouvons identifier grâce à un tableau fourni avec la notice de la carte de mesure.

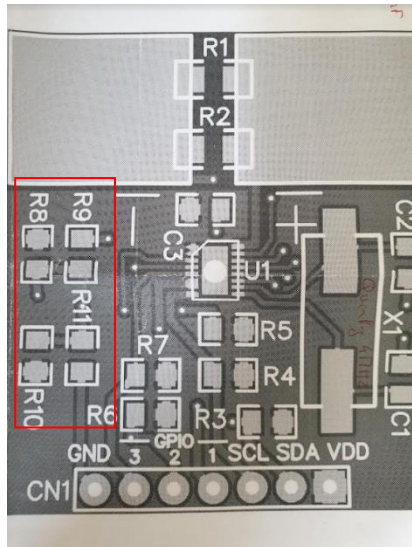


Figure 9 : schéma de rootage de la carte de mesure

### Bus I2C (Inter Integrated Circuit):

- But : faire communiquer entre eux des composants électroniques grâce à seulement trois fils.
- Caractéristiques :
  1. Un signal de données (SDA), un signal d'horloge (SCL) et un signal de référence électrique (masse).
  2. Une adresse pour chaque périphérique
  3. Bus multi-maître
  4. Filtrage intégré
  5. Nombre de circuits uniquement limité par la capacitance maximale du bus : 400pF.

Version schématisée d'une des cartes de mesure utilisée dans notre projet :

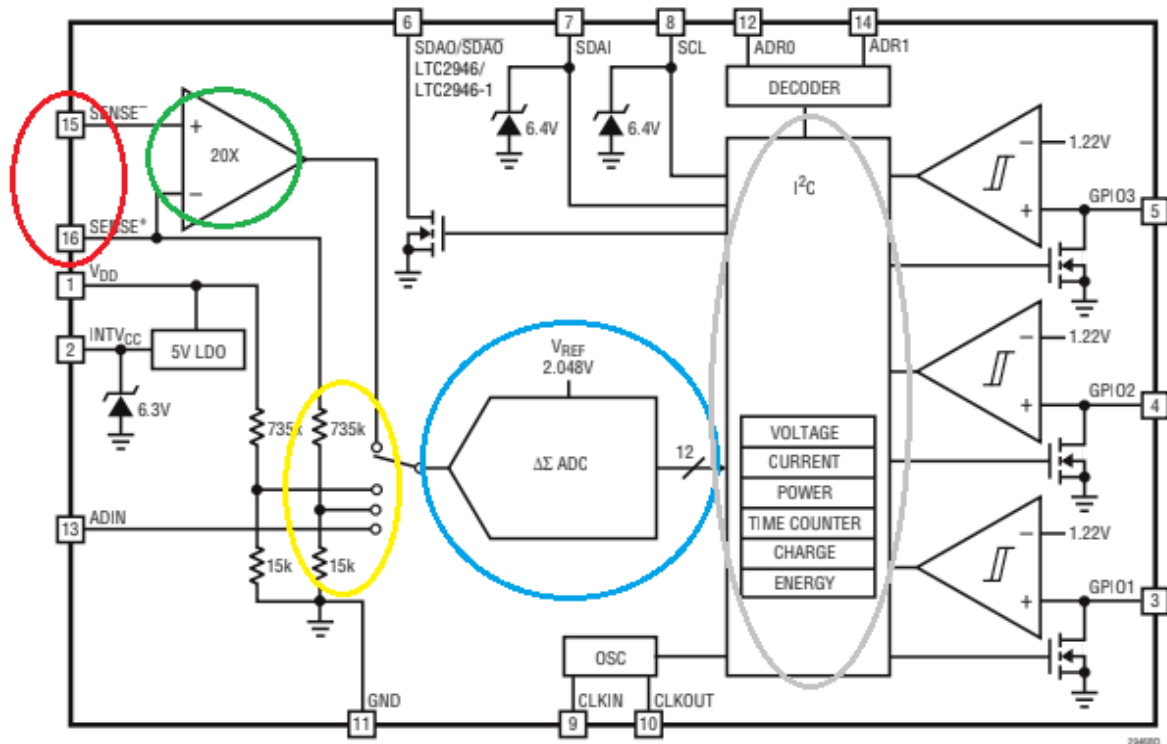


Figure 10 : schéma électrique d'une carte de mesure

## Fonctionnement d'une carte de mesure :

-En jaune : le pont diviseur permet de calculer la tension du circuit, nous pouvons les retrouver grâce aux 2 octets nommés Vin MSB et Vin LSB, explication plus en détails en aval.

-En rouge : entre les bornes 15 (sense-) et 16 (sense+), une tension est mesurée aux bornes de deux résistances très faibles ( $R_1=200\mu\Omega$  et  $R_2=200\mu\Omega$ ) branchées en parallèle, ce qui nous donne une résistance équivalente  $((R_1+R_2) / (R_1 \cdot R_2))$  de  $0.01\Omega$ . C'est à l'aide de ces deux bornes que nous allons effectuer la mesure de tension ( $\Delta SENSE$ ), permettant de calculer le courant par la loi d'Ohm, explication plus en détails en aval.

-En vert : ce composant multiplie la valeur de tension par 20 (précision).

-En bleu : ce bloc (ADC) est un convertisseur analogique  $\rightarrow$  numérique, il nous donne un rapport de transformation, le constructeur nous dit que 2.048v correspond à une valeur de 4096 (numérique), cette valeur de 4096 est liée à la façon dont les deux octets sont utilisés, le premier est totalement utilisé (donc 8 bits ce qui correspond à  $2^8$  en valeur décimale) et le deuxième est utilisé à moitié seulement (donc 4 bits ce qui correspond à  $2^4$  en valeur décimale), si on multiplie les deux en valeur décimale cela nous donne bien 4096.

Ce bloc s'adapte grâce à un interrupteur, au différent style de mesure souhaitée (entre deux points du circuit, ou entre un point et la masse).

-En gris : c'est le bloc où les calculs sont effectués (tension, courant, puissance...), grâce en parti à la loi d'OHMS, ( $U=R*I$ ).

Pour la mesure de **tension** (entre un point du circuit et la masse) :

La borne 16 (SENSE+) nous permet une mesure de tension entre un point du circuit et la masse, le constructeur nous limite l'entrée à 102.4V.

-En **jaune** : la tension (SENSE+) recueillie, passe par un pont diviseur, d'où le calcul s'écrit :  $UR1=(SENSE+) * (R1/(R1+R2))$  avec  $R1=15K\Omega$  et  $R2=735K\Omega$ .

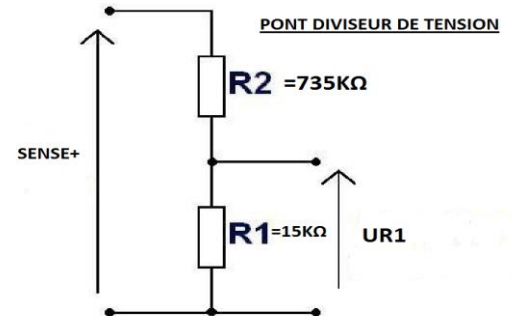


Figure 11 : pont diviseur de tension

La tension réduite est ensuite récupérée par le bloc ADC, qui la transforme d'après le rapport de transformation de 0.025 (car pour 102.4v nous avons que 2.048v en sortie), puis en valeur numérique, comme déjà précisé en amont, avec cette fois ci 2 octets nommés Vin MSB et Vin LSB.

Le reste des valeurs est, dans les deux cas, calculé par le bloc Gris I².

Mesure du **courant** :

Nous savons que le courant mesuré se situe entre les deux bornes (15 et 16), mais la valeur que nous récupérons est sous forme numérique, c'est pourquoi il nous faut trouver un facteur pour la transformer en valeur exploitable, avec la loi d'ohm  $U=R*I$  qui nous donne  $I=U/R$ , nous avons  $U_{max}=2.048/20=0.1024V$  (on le multiplie par 20 pour avoir la valeur d'entrée) et  $R_{eq}=0.01\Omega$  (en théorie), ce qui nous donne un  $I_{max}$  théorique de 10.24A, sachant que cette valeur correspond à 4095 en numérique, avec un simple produit en croix on trouve un facteur 0.0025A/LSB. Différents des données constructeur (Courant =  $25\mu V/LSB/RSNS= 0.00125A/LSB$  soit 5.12A max), car la résistance de mesure du constructeur est de  $0.02\Omega$ , alors que la nôtre est de  $0.01\Omega$ .

Donc chaque valeur d'ampérage mesurée devra être multipliée par ce facteur.

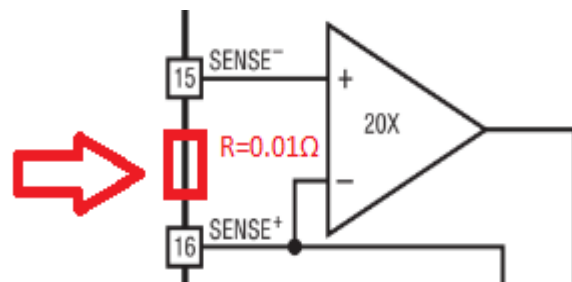


Figure 12 : résistance pour la mesure

Calcul de la puissance :

On sait d'après nos précédents calculs que le facteur de transformation de la tension est de 0,025 V/LSB et que le rapport de transformation du courant est de 2.5m A/LSB. Nous pouvons à partir de ces deux rapports de transformation retrouver celui de la puissance en multipliant l'un par l'autre, ce qui nous donne un rapport de transformation pour la puissance de 62.5μW/LSB. Pour retrouver notre valeur de puissance nous devons donc multiplier la valeur décimale affichée par la Raspberry par notre rapport de transformation.

(Puissance constructeur = 1.25mA/LSB • 25mV/LSB= 31.25μW/LSB soit 0.128W max) différentes car résistance de mesures de 0.02Ω.

Calcul de l'énergie :

Pour calculer le facteur de transformation pour l'énergie, nous avons utilisé la documentation du constructeur qui nous donne comme formule : (facteur de transformation pour la puissance) \*65536\*(16,39543\*10<sup>-3</sup>ms). Le constructeur nous donne un facteur de puissance de 3,125\*10<sup>-5</sup> J/LSB. Or nous avons calculé précédemment un rapport égal à 62.5μW/LSB ce qui nous donne une valeur de facteur de transformation pour l'énergie de 67,3157mJ/LSB, cette différence peut être expliquée par le fait que le constructeur nous indique qu'une résistance de 0.02 Ω est placée entre les points SENSE + et SENSE -.

(Energie constructeur=31.25μW • 65536 • 16.39543ms= 33.578mJ/LSB)

- Caractéristiques d'une carte de mesure :

Mesures	Valeur Maximum	Précision	Nos relevés, Rshunt(0.01Ω)
Tension	102.4V	+/- 0.4%	0.025V/LSB Sur deux octets, non utilisé en entier (2 <sup>12</sup> )
Ampérage	10.24A	+/-0.6%	2.5mA/LSB Sur deux octets, non utilisé en entier (2 <sup>12</sup> )
Puissance	1048.576 W	+/-1%	62.5μW/LSB Sur trois octets (2 <sup>24</sup> )
Energie	1 126 707.495 J 313 Wh	+/-1%	67.157mJ/LSB Sur trois octets (2 <sup>24</sup> )



MODULE		FU 150 P
Standard Test Conditions (STC): 1000 W/sqm - AM 1.5 - 25 °C - 1		
Module power (Pmax)	W	150
Module efficiency	%	15.1
Maximum power voltage (Vmpp)	V	17.9
Maximum power current (Impp)	A	8.38
Open circuit voltage (Voc)	V	22.2
Short circuit current (Isc)	A	8.9

Figure 13 : données constructeur

En comparant les données du panneau et celles de la carte de mesure, nous pouvons remarquer que la tension maximale de la carte ainsi que le courant maximal sont supérieures à la tension et au courant du panneau. Nous pouvons donc utiliser ces cartes de mesures à la sortie du panneau sans risque de détérioration de la carte de mesure.

## Test mesures :

Sur le schéma d'installation des cartes de mesure vu précédemment, on peut voir cinq points de mesure où il serait intéressant de relever les informations.

Afin de tester notre système nous avons réalisé notre installation sur un petit panneau photovoltaïque et trois cartes de mesure. Les trois cartes de mesures ont été soudées par nos soins en suivant le schéma d'une carte. Cependant le panneau délivrant une tension variable nous avons remplacé le panneau par une alimentation stable pour être sûr de nos résultats.

Notre première carte (déjà soudée) ayant déjà deux résistances installées nous avons soudé une carte sans résistance et une autre avec une seule résistance afin d'être sûr d'avoir trois adresses différentes. La méthode pour retrouver leur adressage sera expliquée plus tard.

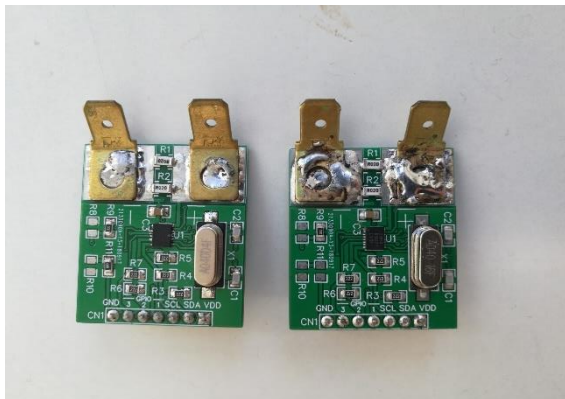


Figure 14 : deux cartes de mesure pour test

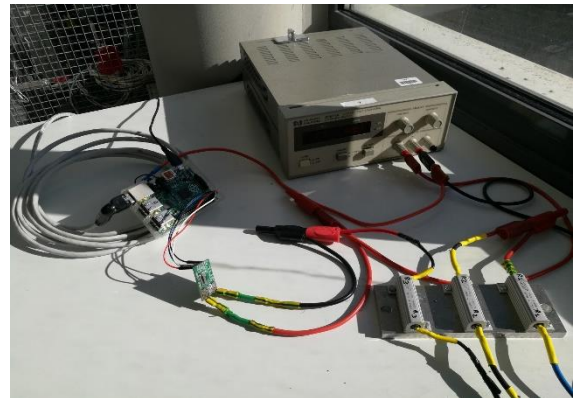


Figure 15 : installation test mesures

## Récupération et traitement des données :

Afin de réaliser une interface de supervision nous avons besoin de rapatrier les données des cartes de mesure. Cependant les données reçues depuis les cartes ne peuvent pas être exploitées directement. On peut voir ci-dessous un extrait de la documentation de la carte de mesure, comprenant un tableau qui regroupe toute les données et leur propre adressage, nous permettant de les récupérer à travers le Raspberry.

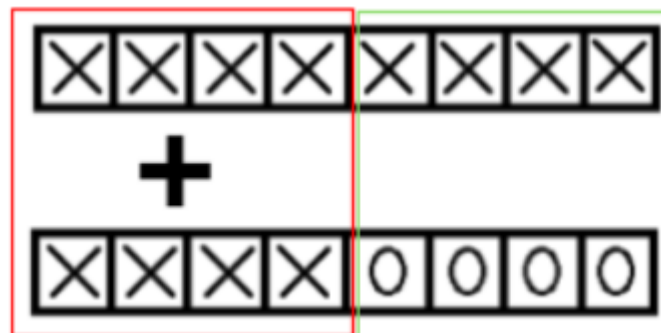
On peut voir que pour la donnée de tension  $V_{IN}$ , l'information est envoyée sur deux octets, un octet de poids fort  $V_{IN}$  MSB et un octet de poids faible  $V_{IN}$  LSB. L'octet de poids fort est rangé sous la forme XXh il utilise ses 8 bits, l'octet de poids faible est rangé sous la forme X0h il utilise seulement les 4 dernier bits.

1Eh	$V_{IN}$ MSB	R/W	ADC $V_{IN}$ MSB Data	XXh
1Fh	$V_{IN}$ LSB	R/W	ADC $V_{IN}$ LSB Data	X0h

Figure 16 : rangement bits

Pour pouvoir lire l'information il faut ajouter les deux octets, cependant on ne peut pas les ajouter directement. En rouge, on ne peut pas faire une addition de bit, cela changerait l'état du bit, pour conserver le même état il faut leur ajouter la valeur 0 comme ici les bits encadrés en vert.

Récupération des données  $\Delta V_{IN}$  (variable de tension) :



Nous allons montrer comment, avec un exemple, on peut modifier des octets afin de récupérer les valeurs.

La mesure  $\Delta V_{IN}$  est composée de deux octets :

- $\Delta V_{IN}$  MSB (le poids fort) rangé par le constructeur en XXh (l'octet est utilisé en entier).

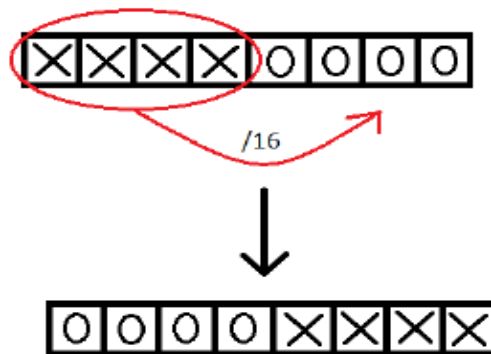


- $\Delta V_{IN}$  LSB (le poids faible) rangé par le constructeur en X0h (seulement les 4 bits les plus forts sont utilisés).

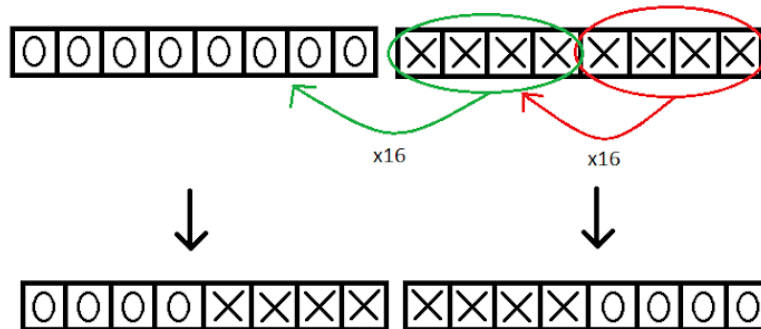


Pour pouvoir traiter la valeur il nous faut additionner les deux octets, pour cela nous devons les réorganiser :

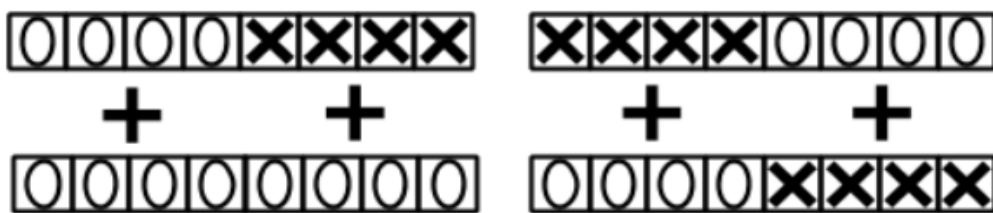
$\Delta V_{in}$  LSB doit être réorganisé en décalant les bits sur ceux de poids faible en divisant par  $2^4 = 16$  (car pour le décaler d'un bit d'une case vers la droite il faut diviser par 2).



$\Delta V_{in}$  MSB doit aussi être réorganisé pour laisser cette fois ci les bits de poids faible libre, en multipliant cette fois ci par  $2^4 = 16$  (car il faut le décaler vers la gauche d'une case il faut le multiplier par 2).



Ensuite il nous suffit de les additionner :



Chaque bit va donc pouvoir être ajouté à une valeur de 0 ce qui nous donne.



Nous pouvons maintenant traiter les octets et ressortir une valeur décimale.

Exemple de calcul :

-Pour  $SENSE+ = 7V$ , cela nous donne :

- $UR1 = (SENSE+) * (R1 / (R1 + R2))$  avec  $R1 = 15 \text{ k}\Omega$  et  $R2 = 735 \text{ k}\Omega$ , donc  $UR1 = 7 * 0.02 = 0.14$
- Rapport de transformation :  $(0.14 * 4096) / 2.048 = 280$  en valeur décimale

Donc pour 7V en entrée nous récupérerons la valeur 280 dans la mémoire de la carte.

## Communication avec le Raspberry :

Pour communiquer avec le Raspberry on utilise le langage Python en inscrivant l'adresse IP du Raspberry, une fois la page texte ouvert, entrer le login : pi et le mot de passe : Raspberry puis valider.

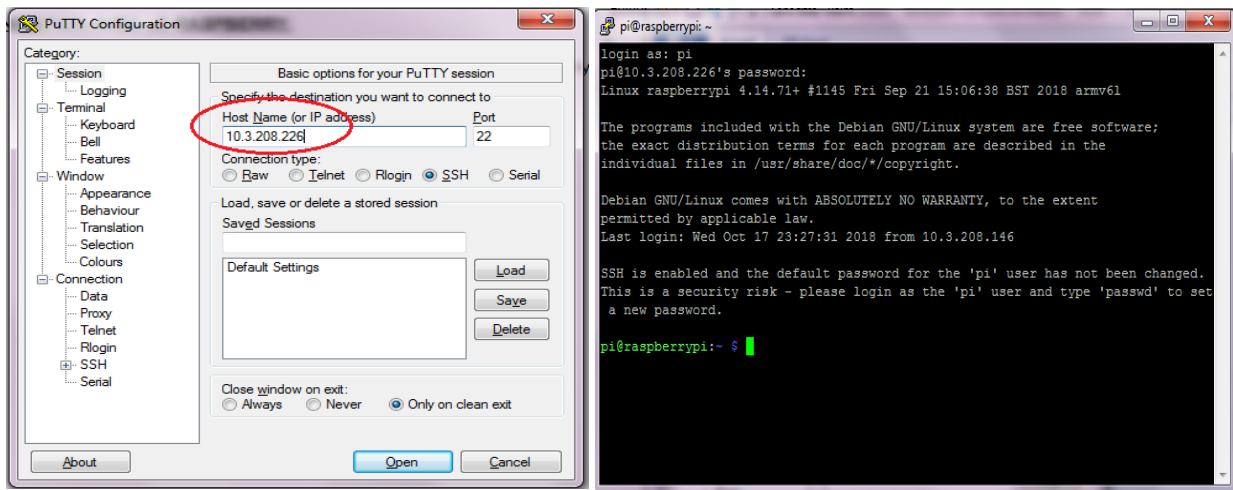


Figure 17 : première étape de communication avec Raspberry

Ensuite entrer "python" pour rentrer dans le programme :

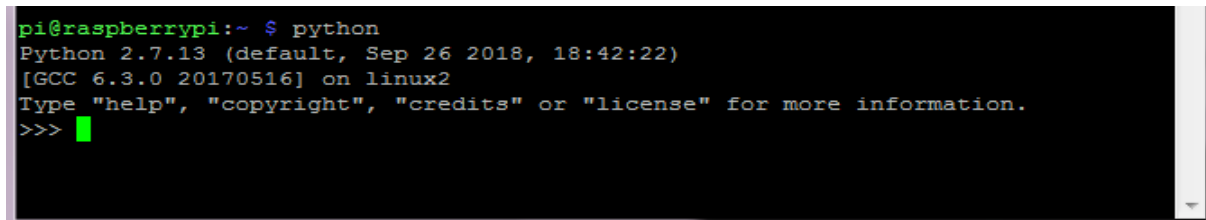


Figure 18 : entrée dans le programme

Exemple de communication de base du python :

**bus.read\_byte\_data(address, 0x27)** est une commande qui nous permet en communiquant avec la Raspberry (bus) de lire une donnée en octet (read\_byte\_data). Pour cela, il faut préciser l'adresse de la carte de mesure que l'on veut interroger (address) ainsi que le registre que l'on veut lire qui est codé en valeur hexadécimale (0x27). (Voir annexe tableau...)

**bus.write\_byte\_data(address, 0x27, 0xff)** est une commande fonctionnant exactement comme la commande ci-dessus, elle sert à écrire une valeur dans un registre.

Ici nous pouvons retranscrire l'exemple cité plus tôt en programme :

```
pi@raspberrypi:~$ python
Python 2.7.13 (default, Sep 26 2018, 18:42:22)
[GCC 6.3.0 20170516] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import smbus
>>> bus = smbus.SMBus(1)
>>> address = 0x6f
>>> VinMSB=(bus.read_byte_data(address, 0x1e))*16
>>> VinLSB=(bus.read_byte_data(address, 0x1f))/16
>>> VinMSB+VinLSB
270
>>>
```

Connexion avec le bus

Créer une variable VinMSB et lui donner la valeur de l'adresse correspondante multiplié par 16

Créer une variable VinLSB et lui donner la valeur de l'adresse correspondante divisé par 16

Ajouter les deux variables

Figure 19 : connexion avec le bus

Nous trouvons une valeur de 270 ce qui fait 6.75 V.

Il nous suffit maintenant de réaliser la même chose pour les autres données que nous voulons relever.

# Supervision

Récupérer ces données nous permet ensuite de pouvoir les traiter grâce à un logiciel de supervision nommé **PyScada**. Avec ce logiciel, nous pouvons récupérer les valeurs de nos différentes cartes de mesure, puis les afficher en graphiques (ou autres), les enregistrer, et y avoir accès à tout moment et n'importe où.

- Méthode de connexion à Pyscada :
  1. Commencez par tester la connexion en effectuant un “ping” dans l’onglet commande, à l’adresse IP : 10.3.208.226.
  2. Ensuite ouvrir une page web, et dans la barre de recherche tapez l’adresse IP : 10.3.208.226.
  3. Une page s’affiche demandant un login et un password  
Login : pi  
Password : trackeriut

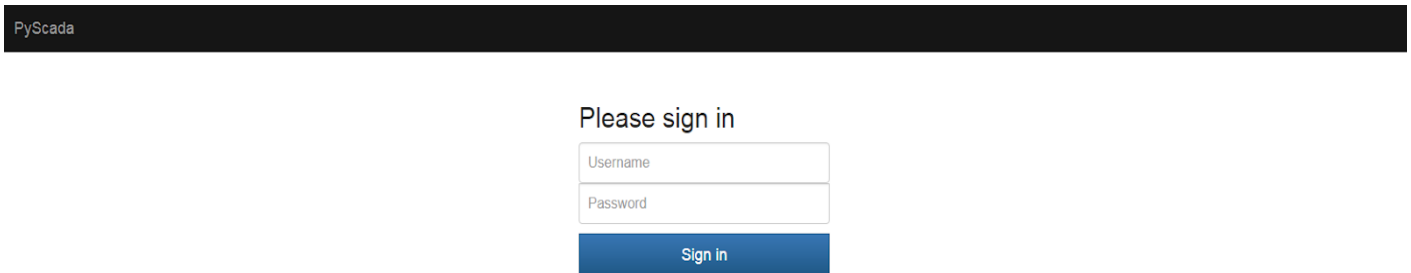


Figure 20 : page d'entrée Pyscada

Une fois connecté nous pouvons créer des vues et afficher des graphiques par exemple. Avant de les créer nous devons tout d’abord écrire le script qui permettra de réaliser les conversions entre les valeurs en octets et les valeurs en V, A... Que l’on affichera sur les graphiques (voir annexe tutoriel du logiciel PyScada).

Pour réaliser notre interface de supervision nous avons installé une carte de mesure à la sortie d’un petit panneau photovoltaïque (tension max 17.5 V, courant max 1.14A, puissance max 20 W). Avant de commencer à créer les graphiques nous avons dû déterminer l’adresse de la carte que nous allons utiliser pour mesurer les données (voir annexe ...). Nous avons également modifié le script afin de pouvoir tester nos différentes cartes. Pour cela nous avons utilisé la fonction « try » qui permet de tester quelle carte est mis en place sur le circuit sans risque de dysfonctionnement de l’installation.

Après avoir réalisé ces deux tâches, nous avons créé nos relevés affichant les quatre données essentielles pour avoir un suivi sur le rendement optimal du panneau ; c’est-à-dire la tension, le courant, la puissance et l’énergie.

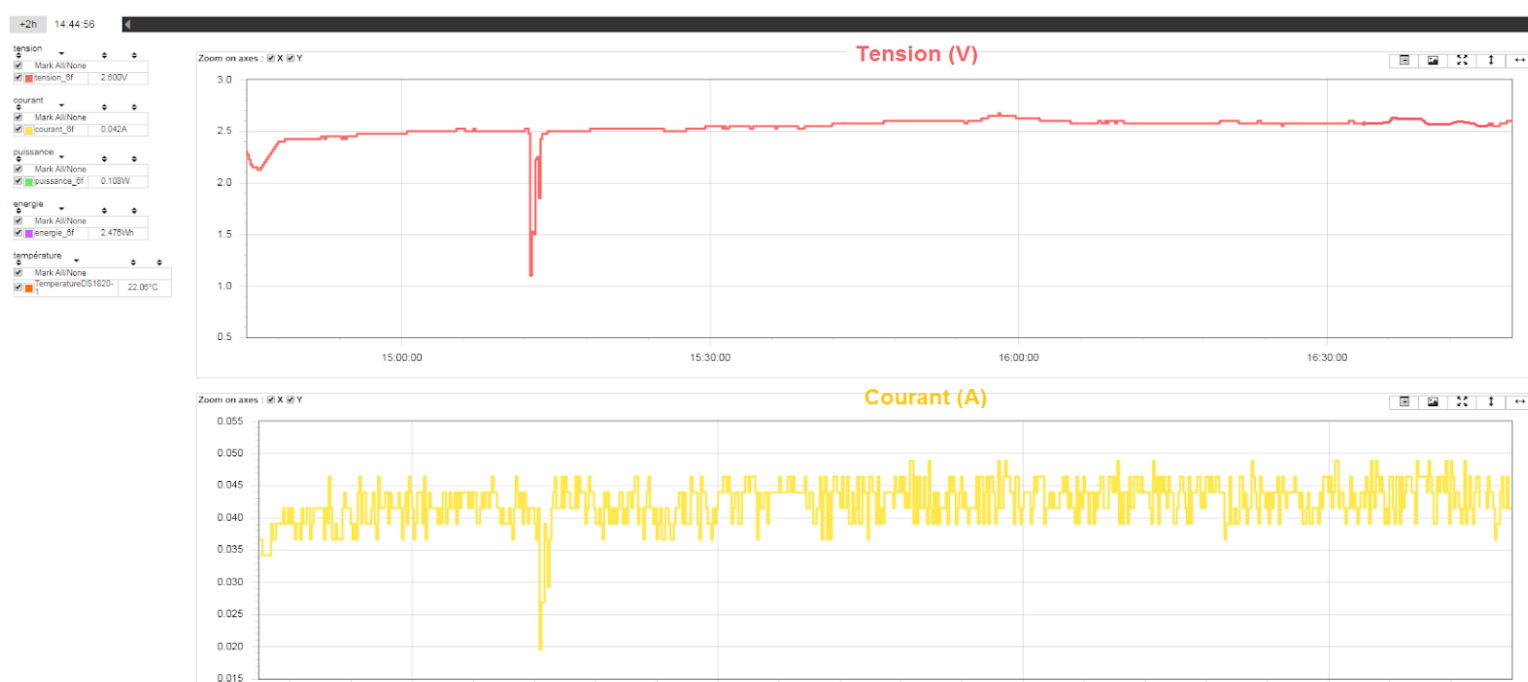


Figure 21 : relevé tension et courant en fonction du temps

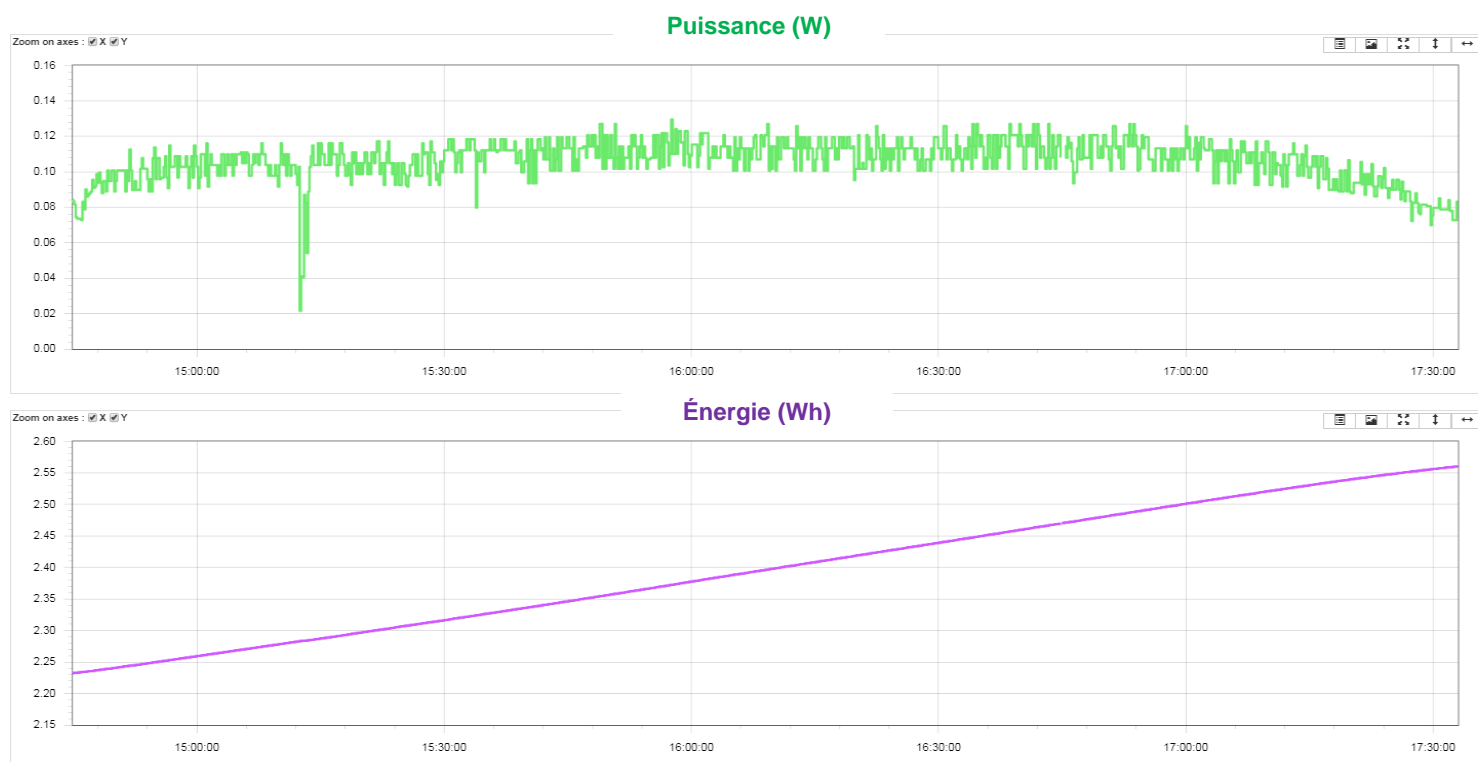


Figure 22 : relevé puissance et énergie en fonction du temps



Nous avons également rajouté une sonde de température mesurant la température dans la salle et nous avons créé le graphe de l'évolution de la température en temps réel.

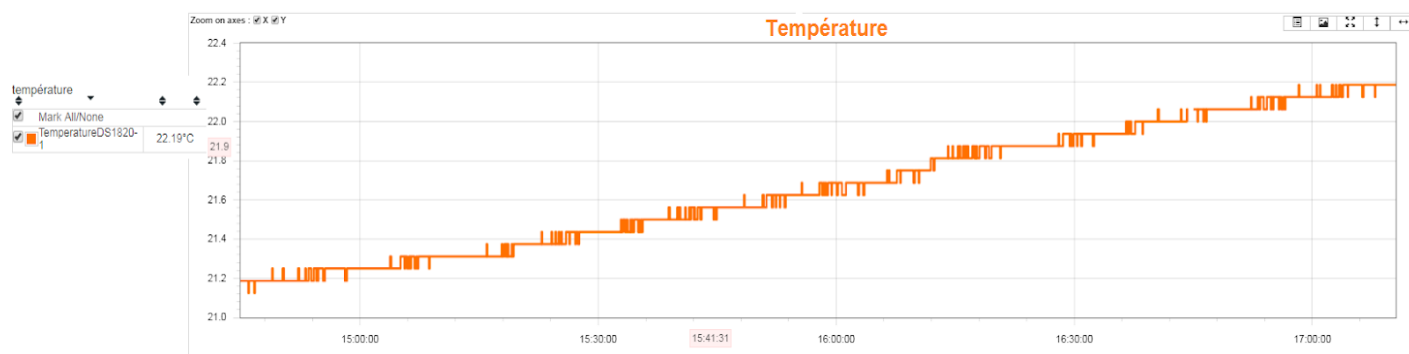


Figure 23 : relevé de la température en fonction du temps

## Partie mécanique du projet

### Conception du boîtier

L'un des objectifs principaux de notre projet était de supprimer l'armoire électrique extérieure au panneau photovoltaïque, en concevant un seul et même boîtier capable d'accueillir tous les composants nécessaires au fonctionnement du tracker : les deux moteurs (axe horizontal et axe vertical), le microordinateur Raspberry Pi et le régulateur de charge MPPT 70/15. Le boîtier doit être étanche, lors de sa conception nous allons devoir penser à une solution empêchant l'eau de s'infiltrer.

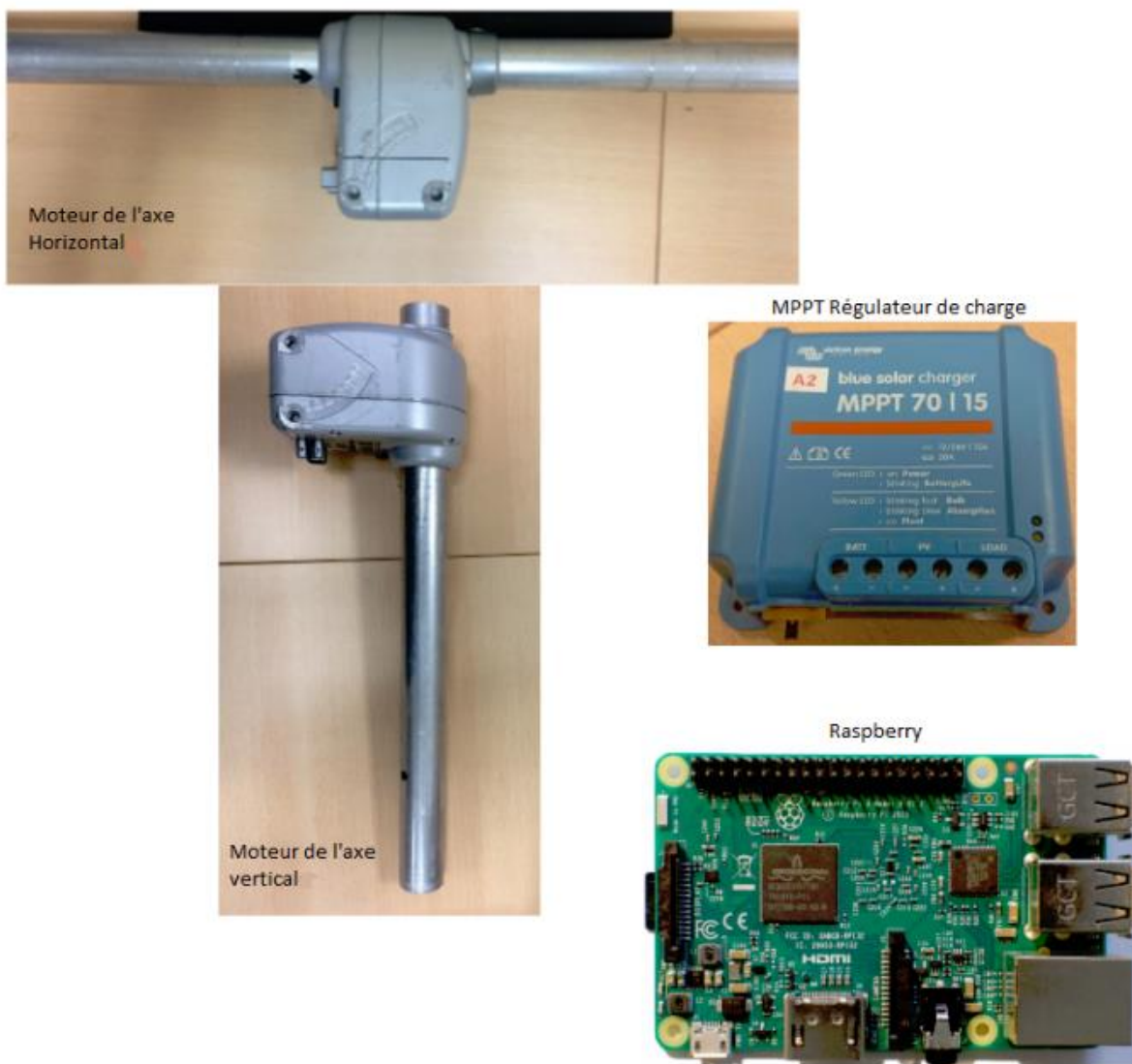


Figure 24: composants du boîtier

L'ancienne armoire électrique accueillait le régulateur de charge, le Raspberry Pi ainsi qu'un transformateur qu'on ne gardera pas pour la suite.



Figure 26 : ancienne armoire électrique



Figure 25 : Raspberry



Figure 27 : régulateur de charge MPPT 70/15

Le fait de supprimer cette armoire électrique réduit le nombre de câbles sortants du panneau en laissant seulement celui de la batterie. Les deux moteurs étant placés dans un boîtier à l'arrière du panneau, afin d'assurer sa rotation sur l'axe azimut et vertical, et l'armoire étant extérieure au panneau, il fallait alors relier les moteurs à l'armoire et donc au régulateur de charge et au Raspberry. Les câbles de liaison devaient passer à l'intérieur de l'axe du moteur vertical, ce qui était contraignant en cas de maintenance et pour la mobilité du panneau. Alors qu'en plaçant tous les composants dans un seul et même boîtier, ce problème disparaîtrait. Ci-dessous, nous avons un modèle de boîtier pouvant répondre à l'objectif que nous nous sommes fixé :

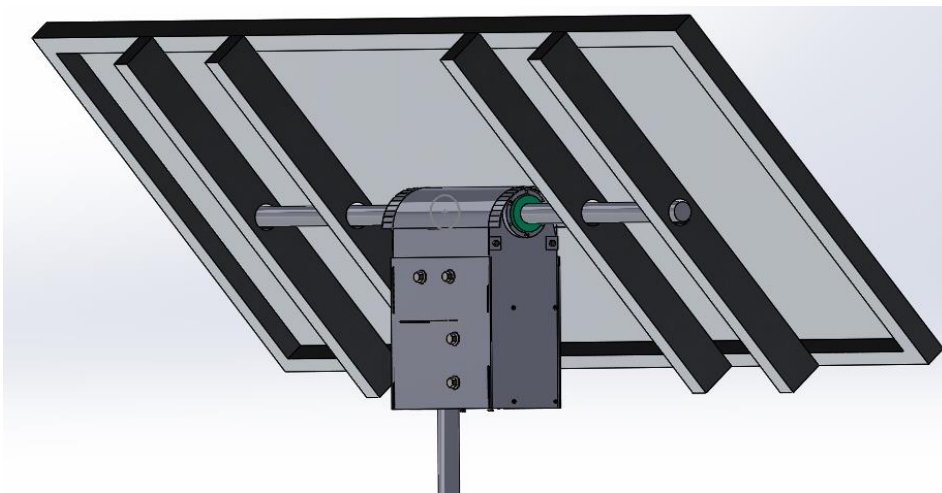


Figure 28 : objectif final

Avant de le réaliser nous avons tout d'abord pris les mesures des moteurs, du régulateur de charge et du boîtier d'accueil du nano ordinateur Raspberry Pi. Nous avons ensuite pensé à l'emplacement de ces éléments dans le boîtier afin de le dimensionner.

Après avoir passé ces étapes nous sommes passé à la modélisation des différentes pièces grâce au logiciel de simulation 3D SolidWorks :

- Les premières à modéliser étaient les quatre plaques. L'ensemble doit être mécano-soudé en imbriquant les plaques entre elles par l'intermédiaire d'encoches (Cf. [flèches bleues](#)) dans le but de faciliter les soudures une fois le boîtier monté.

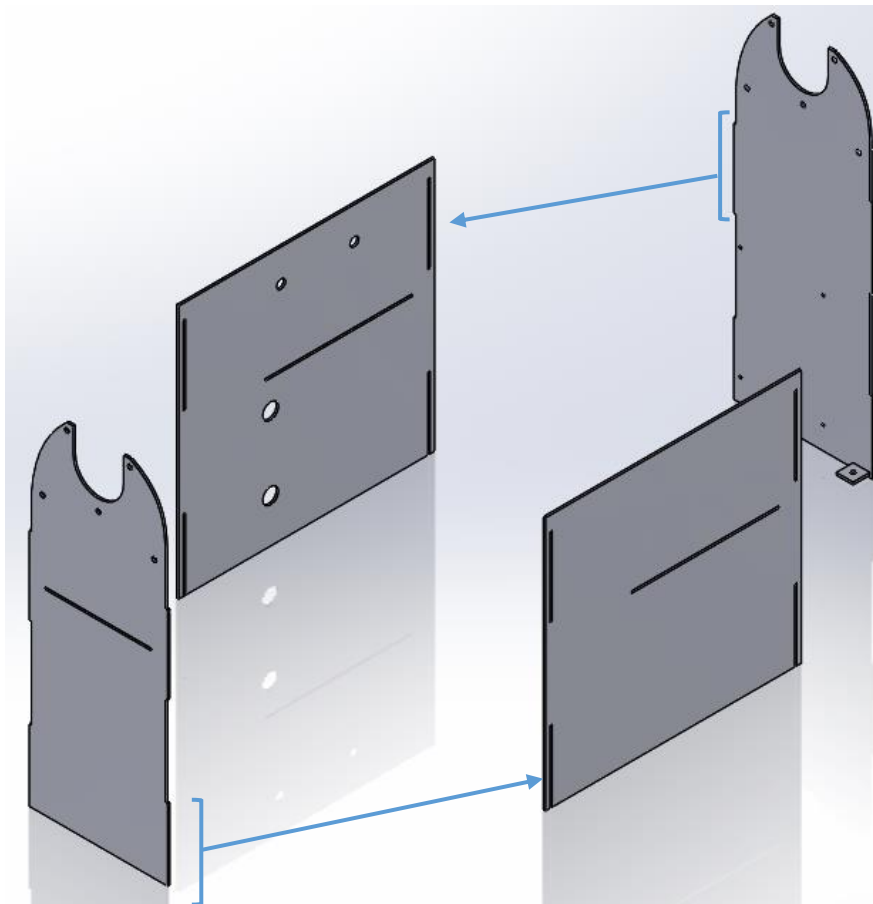


Figure 29 : 1<sup>ère</sup> étape de la modélisation 3D

- La deuxième étape de la modélisation du boîtier était d'insérer les deux moteurs (**1** et **2**) dans l'assemblage.

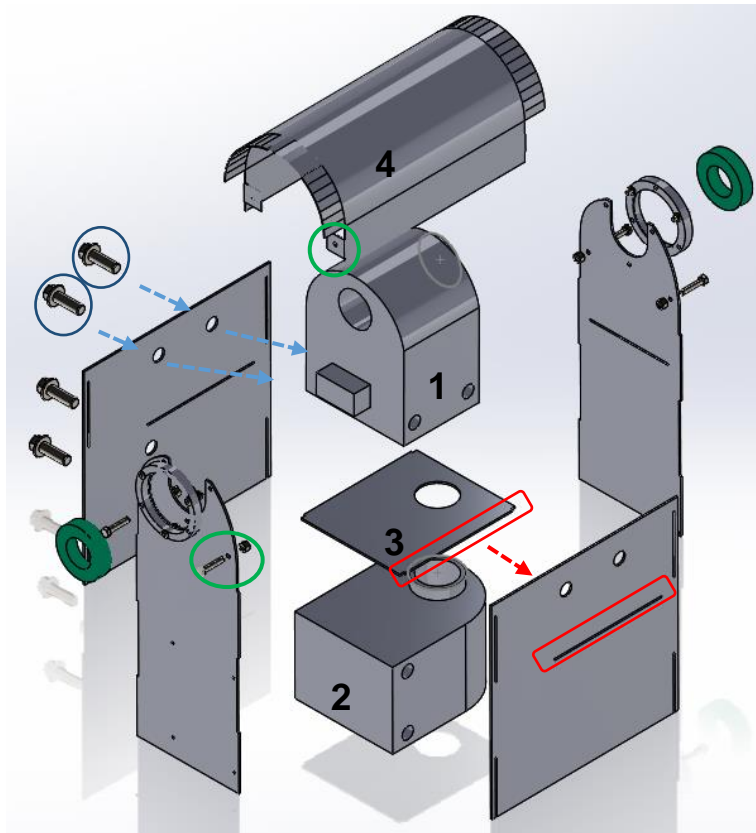


Figure 30 : 2ème étape de la modélisation 3D

Les moteurs sont fixés au boîtier par l'intermédiaire de quatre vis (Cf. [flèches et cercles bleus](#)).

Concernant Le moteur **2**, en plus d'être vissé au boîtier, est relié à la plaque **3** par l'intermédiaire de son axe et d'un roulement qui sera monté sur la plaque. Elle est solidaire au boîtier grâce à un **système d'encoche** afin de les imbriquer puis de les souder.

Dans la mesure où il faut protéger les moteurs les roulements et l'intérieur du boîtier d'éventuelles intempéries, nous avons modélisé un couvercle (**4**). Il sera vissé par l'intermédiaire d'un système vis-écrou avec les écrous soudés à l'intérieur du boîtier. (Cf. [cercles verts](#)).

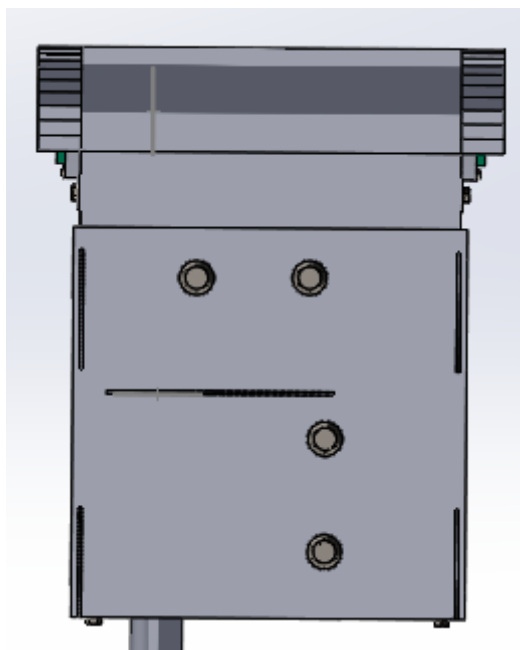


Figure 31 : boîtier

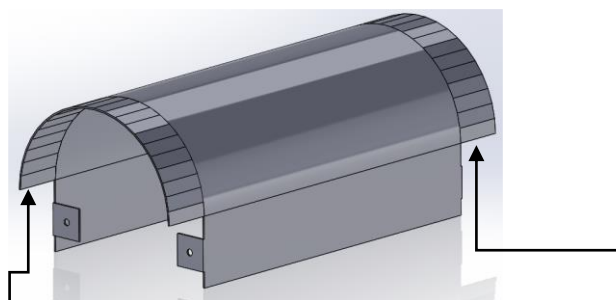


Figure 32 : couvercle

Le couvercle est plus long de 30mm de chaque côté afin de protéger les roulements de la pluie.

Nous avons aussi utilisé le système vis-écrou pour fixer les deux accueils roulement et donc les roulements qui vont assurer la rotation du moteur à axe horizontal.

- La troisième étape consistait à assembler le régulateur de charge MPPT et le Raspberry Pi. Il fallait aussi créer une plaque permettant de fermer le dessous du boîtier.

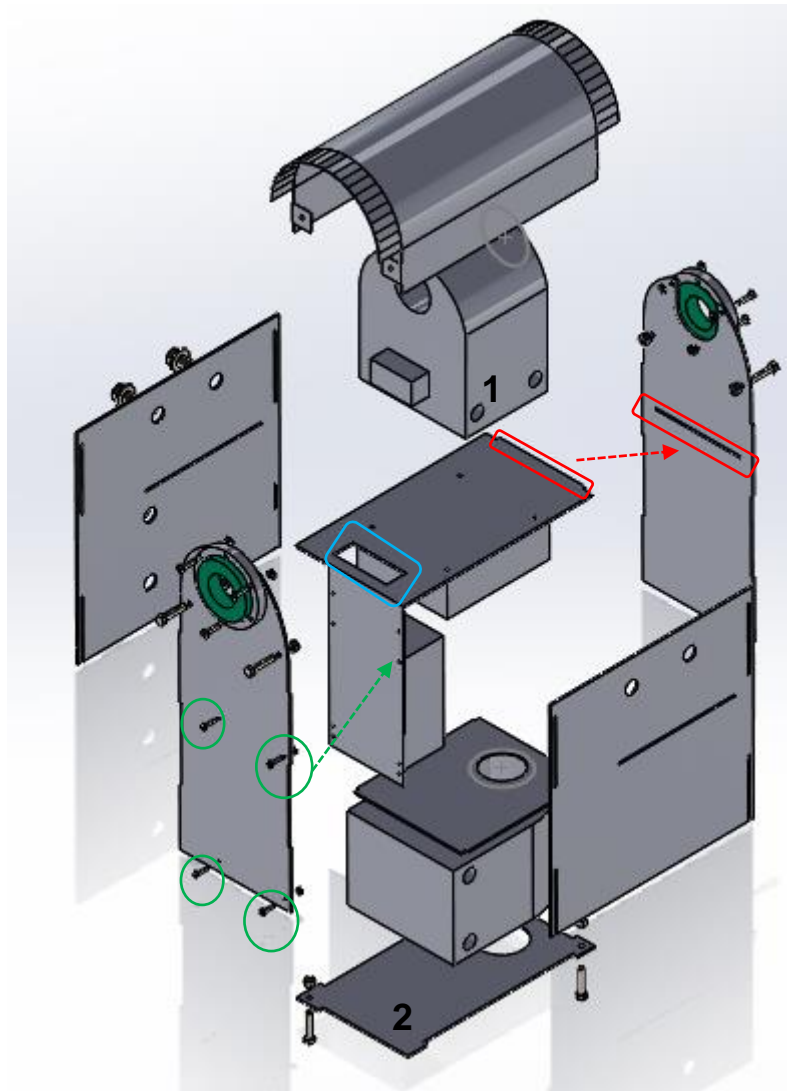


Figure 33 : 3ème étape de la modélisation

Nous avons choisi de visser nos composants (MPPT et Raspberry) sur une plaque à deux faces qui sera elle-même vissée (**système de vis-écrou**). Cette solution facilitera le montage et rendra accessible le MPPT et le Raspberry en cas d'éventuelles opérations de maintenance. Un bout de la plaque est imbriqué dans une encoche et ensuite soudé au boîtier ce qui le rendra plus solide. (**Cf. rectangles et flèche rouges**). Une **ouverture** est aussi prévue pour laisser passer les câbles de liaison du moteur (**1**).

Nous avons modélisé un couvercle (**2**) capable de fermer le bas du boîtier grâce à deux vis et deux écrous. Le fait de rajouter cette plaque ne doit pas empêcher l'eau, créée par condensation, de s'écouler par le bas. Nous avons donc modélisé un profil de plaque avec des ouvertures afin d'évacuer toutes sources d'humidité. Elles vont aussi permettre, avec l'espace entre le couvercle du haut, de créer un courant d'air et de maintenir l'intérieur du boîtier à bonne température les composants.

## Conclusion

Au cours de notre projet nous devions réaliser plusieurs objectifs évoqués dans l'introduction. Nous avons pu y répondre d'un point de vue réalisation, en revanche d'un point de vue organisationnel nous avons réalisé nos objectifs avec du retard.

Ce projet nous a permis de d'acquérir de nouvelles connaissances en termes d'utilisation du Raspberry PI et de programmation en langage Python en apprenant seulement les bases. Nous avons pu développer une interface de supervision énergétique du panneau qui pourra être utilisée par les élèves lors de futures travaux pratiques.

Nous avons aussi pu développer nos compétences en matière de modélisation 3D grâce au logiciel SolidWorks car nous devions repenser la conception du boîtier. Il fallait le rendre le plus compact et ergonomique possible pour la maintenance et sa mobilité, tout en s'assurant de son étanchéité et la ventilation des composants du boîtier. La solution proposée est viable mais afin de valider complètement ce modèle (épaisseur des plaques, résistance des matériaux) il est nécessaire d'effectuer le test de vent pour simuler les forces exercées par le vent sur le boîtier. Nous n'avons pas réussi à le réaliser entièrement, cependant nous sommes confiants quant aux résultats obtenus lors de nos tentatives.

Concernant les pistes d'améliorations et les tâches à réaliser pour les années à venir :

- Effectuer entièrement le test de vent pour valider l'épaisseur des plaques du boîtier.
- Usiner les pièces du boîtier.
- Le monter puis effectuer le câblage dans le but de l'installer sur le toit de l'IUT.
- Ajouter des cartes de mesure.
- Développer l'interface de supervision en ajoutant plus de variables.

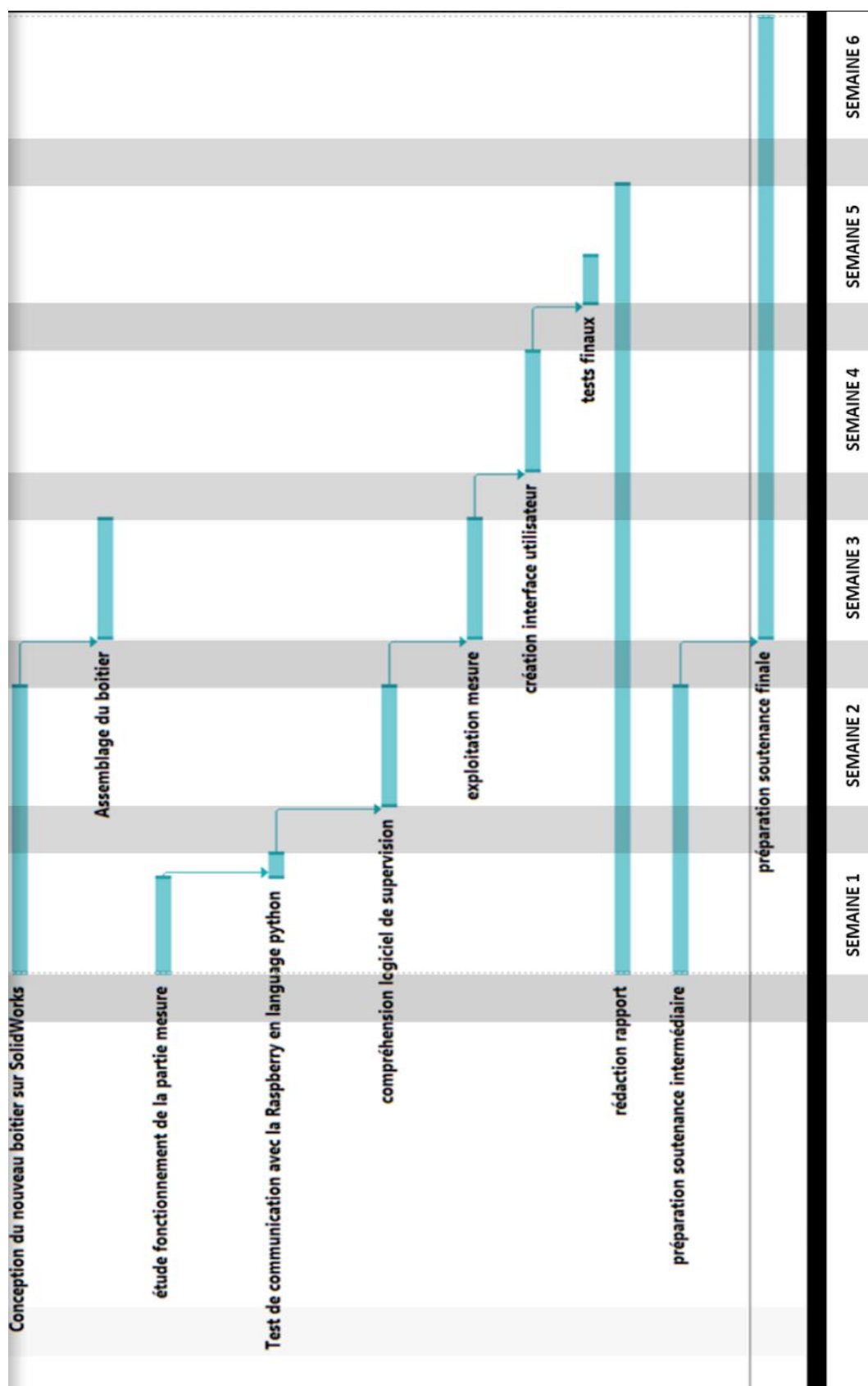


# Annexes

## Lexique

- Panneau monocristallin : Panneau solaire composé de cellules qui sont elle-même composées d'un seul cristal
- Panneau polycristallin : Panneau solaire composé de cellules composées de plusieurs cristaux
- Panneau amorphe : Les panneaux solaires amorphes ont un taux de rendement deux fois moins important que le polycristallin, mais beaucoup plus actifs dans le diffus et beaucoup moins cher.
- Silicium : Le silicium est un élément de chimie (Si). Il est présent sur l'ensemble de la terre et notamment dans les minéraux.
- MPPT : Un MPPT (Maximum Power Point Tracking) est un principe permettant de suivre, le point de puissance maximal d'un générateur électrique non linéaire.
- Raspberry : Le Raspberry est un microordinateur équipé du langage de programmation Python.
- Python : Le python est un langage de programmation, c'est un langage facile est très puissant utilisé partout dans le monde.
- PuTTY : PuTTY est logiciel qui nous permet de se connecter avec le Raspberry.
- Pyscada : Pyscada est un logiciel permettant de développer des interfaces de supervision.
- SolidWorks : SolidWorks est un logiciel de conception 3D utiliser dans l'industrie permettant de créer des pièces et des assemblages.

## Diagramme de Gant



## Tableaux de structuration des données des carte de mesures

Table 1. LTC2946 Device Addressing

DESCRIPTION	HEX DEVICE ADDRESS	BINARY DEVICE ADDRESS								LTC2946 ADDRESS PINS	
		a6	a5	a4	a3	a2	a1	a0	R/W	ADR1	ADR0
Mass Write	CC	1	1	0	0	1	1	0	0	X	X
Alert Response	19	0	0	0	1	1	0	0	1	X	X
0	CE	1	1	0	0	1	1	1	X	H	L
1	D0	1	1	0	1	0	0	0	X	NC	H
2	D2	1	1	0	1	0	0	1	X	H	H
3	D4	1	1	0	1	0	1	0	X	NC	NC
4	D6	1	1	0	1	0	1	1	X	NC	L
5	D8	1	1	0	1	1	0	0	X	L	H
6	DA	1	1	0	1	1	0	1	X	H	NC
7	DC	1	1	0	1	1	1	0	X	L	NC
8	DE	1	1	0	1	1	1	1	X	L	L

Table 2. LTC2946 Register Addresses and Contents

REGISTER ADDR	REGISTER NAME	READ/WRITE	DESCRIPTION	DEFAULT
00h	CTRLA	R/W	Operation Control Register A	18h
01h	CTRLB	R/W	Operation Control Register B	00h
02h	ALERT1	R/W	Selects Which Primary Faults Generate Alerts	00h
03h	STATUS1	R	Primary Status Information	00h
04h	FAULT1	R/W	Primary Fault Log	00h
05h	POWER MSB2	R/W	Power MSB2 Data	XXh
06h	POWER MSB1	R/W	Power MSB1 Data	XXh
07h	POWER LSB	R/W	Power LSB Data	XXh
08h	MAX POWER MSB2	R/W	Maximum Power MSB2 Data	00h
09h	MAX POWER MSB1	R/W	Maximum Power MSB1 Data	00h
0Ah	MAX POWER LSB	R/W	Maximum Power LSB Data	00h
0Bh	MIN POWER MSB2	R/W	Minimum Power MSB2 Data	FFh
0Ch	MIN POWER MSB1	R/W	Minimum Power MSB1 Data	FFh
0Dh	MIN POWER LSB	R/W	Minimum Power LSB Data	FFh
0Eh	MAX POWER THRESHOLD MSB2	R/W	Maximum POWER Threshold MSB2 to Generate Alert	FFh
0Fh	MAX POWER THRESHOLD MSB1	R/W	Maximum POWER Threshold MSB1 to Generate Alert	FFh
10h	MAX POWER THRESHOLD LSB	R/W	Maximum POWER Threshold LSB to Generate Alert	FFh
11h	MIN POWER THRESHOLD MSB2	R/W	Minimum POWER Threshold MSB2 to Generate Alert	00h
12h	MIN POWER THRESHOLD MSB1	R/W	Minimum POWER Threshold MSB1 to Generate Alert	00h
13h	MIN POWER THRESHOLD LSB	R/W	Minimum POWER Threshold LSB to Generate Alert	00h
14h	ΔSENSE MSB	R/W	ΔSENSE MSB Data	XXh
15h	ΔSENSE LSB	R/W	ΔSENSE LSB Data	XXh
16h	MAX ΔSENSE MSB	R/W	Maximum ΔSENSE MSB Data	00h
17h	MAX ΔSENSE LSB	R/W	Maximum ΔSENSE LSB Data	00h
18h	MIN ΔSENSE MSB	R/W	Minimum ΔSENSE MSB Data	FFh
19h	MIN ΔSENSE LSB	R/W	Minimum ΔSENSE LSB Data	FFh
1Ah	MAX ΔSENSE THRESHOLD MSB	R/W	Maximum ΔSENSE Threshold MSB to Generate Alert	FFh

1Bh	MAX $\Delta$ SENSE THRESHOLD LSB	R/W	Maximum $\Delta$ SENSE Threshold LSB to Generate Alert	F0h
1Ch	MIN $\Delta$ SENSE THRESHOLD MSB	R/W	Minimum $\Delta$ SENSE Threshold MSB to Generate Alert	00h
1Dh	MIN $\Delta$ SENSE THRESHOLD LSB	R/W	Minimum $\Delta$ SENSE Threshold LSB to Generate Alert	00h
1Eh	V <sub>IN</sub> MSB	R/W	ADC V <sub>IN</sub> MSB Data	XXh
1Fh	V <sub>IN</sub> LSB	R/W	ADC V <sub>IN</sub> LSB Data	X0h
20h	MAX V <sub>IN</sub> MSB	R/W	Maximum V <sub>IN</sub> MSB Data	00h
21h	MAX V <sub>IN</sub> LSB	R/W	Maximum V <sub>IN</sub> LSB Data	00h
22h	MIN V <sub>IN</sub> MSB	R/W	Minimum V <sub>IN</sub> MSB Data	FFh
23h	MIN V <sub>IN</sub> LSB	R/W	Minimum V <sub>IN</sub> LSB Data	F0h
24h	MAX V <sub>IN</sub> THRESHOLD MSB	R/W	Maximum V <sub>IN</sub> Threshold MSB to Generate Alert	FFh
25h	MAX V <sub>IN</sub> THRESHOLD LSB	R/W	Maximum V <sub>IN</sub> Threshold LSB to Generate Alert	F0h
26h	MIN V <sub>IN</sub> THRESHOLD MSB	R/W	Minimum V <sub>IN</sub> Threshold MSB to Generate Alert	00h
27h	MIN V <sub>IN</sub> THRESHOLD LSB	R/W	Minimum V <sub>IN</sub> Threshold LSB to Generate Alert	00h
28h	ADIN MSB	R/W	ADIN MSB Data	XXh
29h	ADIN LSB	R/W	ADIN LSB Data	X0h
2Ah	MAX ADIN MSB	R/W	Maximum ADIN MSB Data	00h
2Bh	MAX ADIN LSB	R/W	Maximum ADIN LSB Data	00h
2Ch	MIN ADIN MSB	R/W	Minimum ADIN MSB Data	FFh
2Dh	MIN ADIN LSB	R/W	Minimum ADIN LSB Data	F0h
2Eh	MAX ADIN THRESHOLD MSB	R/W	Maximum ADIN Threshold MSB to Generate Alert	FFh
2Fh	MAX ADIN THRESHOLD LSB	R/W	Maximum ADIN Threshold LSB to Generate Alert	F0h
30h	MIN ADIN THRESHOLD MSB	R/W	Minimum ADIN Threshold MSB to Generate Alert	00h
31h	MIN ADIN THRESHOLD LSB	R/W	Minimum ADIN Threshold LSB to Generate Alert	00h
32h	ALERT2	R/W	Selects Which Secondary Faults Generate Alerts	00h
33h	GPIO_CFG	R/W	GPIO Configuration	00h
34h	TIME COUNTER MSB3	R/W	Time Counter MSB Data3	XXh
35h	TIME COUNTER MSB2	R/W	Time Counter MSB Data2	XXh
36h	TIME COUNTER MSB1	R/W	Time Counter MSB Data1	XXh
37h	TIME COUNTER LSB	R/W	Time Counter LSB Data	XXh
38h	CHARGE MSB3	R/W	Charge MSB Data3	XXh
39h	CHARGE MSB2	R/W	Charge MSB Data2	XXh
3Ah	CHARGE MSB1	R/W	Charge MSB Data1	XXh
3Bh	CHARGE LSB	R/W	Charge LSB Data	XXh
3Ch	ENERGY MSB3	R/W	Energy MSB Data3	XXh
3Dh	ENERGY MSB2	R/W	Energy MSB Data2	XXh
3Eh	ENERGY MSB1	R/W	Energy MSB Data1	XXh
3Fh	ENERGY LSB	R/W	Energy LSB Data	XXh
40h	STATUS2	R	Secondary Status Information	00h
41h	FAULT2	R/W	Secondary Fault Log	00h
42h	GPIO3_CTRL	R/W	GPIO3 Control Command	00h
43h	CLK_DIV	R/W	Clock Divider Command	04h
E7h	MFR_SPECIAL_ID MSB	R	Manufacturer Special ID MSB Data	60h
E8h	MFR_SPECIAL_ID LSB	R	Manufacturer Special ID LSB Data	01h

## Tutoriel Pyscada

### Méthode pour modifier un script :

1. Cliquez sur l'onglet script qui se trouve dans la page administrateur
2. Copiez le chemin du script (encadré rouge)

### Change script

Label:	<input type="text" value="tracker"/>
<input checked="" type="checkbox"/> Active	
Interval:	<input type="text" value="1 Second"/>
Script file:	<input type="text" value="/usr/local/lib/python3.5/dist-packages/pysc"/>
<div>Delete</div>	

3. Collez le lien dans PuTTY après avoir taper la commande "sudo nano" pour rentrer en tant qu'administrateur.

```
login as: pi
pi@10.3.208.226's password:
Linux raspberrypi 4.14.98+ #1200 Tue Feb 12 20:11:02 GMT 2019 armv6l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

pi@raspberrypi:~ $ sudo nano /usr/local/lib/python3.5/dist-packages/pyscada/laborem/scripts/tracker.py
```

4. Pour modifier la partie du script que l'on veut changer on utilise les raccourcis de commande situés en bas de l'écran.

```
try:
    address = 0x6f          #adresse I2C circuit

    if (RAZ):
        logger.debug("RAZ 6f")
        bus.write_byte_data(address, 0x01, 0x02) #Remise a zero t, q et W
        bus.write_byte_data(address, 0x01, 0x00)
        #self.write_variable_property('tracker','RAZ',0,value_class='boolean')

    if i % 10 == 0:
        VMSB=bus.read_byte_data(address, 0x1E)      #Lecture de la tension
        VLSB=bus.read_byte_data(address, 0x1F)
        tension=( (VMSB*16)+(VLSB/16)) *0.025      #Calcul de la tension
```

^G Get Help   ^O Write Out   ^W Where Is   ^R Cut Text   ^J Justify  
^X Exit   ^R Read File   ^\ Replace   ^U Uncut Text   ^I To Linter

### Méthode pour créer un graphique :

1. Créer une page (qui viendra se placer dans la vue)

Home > PyScada HMI > Pages > Add page

Add page

Title:	<input type="text"/>
Link title:	<input type="text"/>
Position:	<input type="text" value="0"/>

2. Créer la vue : étape (admin - views- additionner- (titre, description, link title=titre, save)).

Home > PyScada HMI > Views > Add view

### Add view

**Title:**

**Description:**

**Link title:**

**Pages:**

Available pages ⓘ

Q Filter

Panneau  
Charge

Choose all ⓘ

Chosen pages ⓘ

Remove all ⓘ

- Créer un graphique (charts) qui viendra se placer dans un widget : étapes (chart - titre (courant/tension) - variables déjà créées (6f représente n°carte de mesure, (adresse module)) - choix variables (ID\*)).

Home > PyScada HMI > Charts > Add chart

### Add chart

Title:

X axis label:

X axis ticks:

Y axis label:

Y axis min:

Y axis max:

Variables:

Available variables ?

Filter

- temps\_6f( seconde )
- courant\_6f( Ampère )
- tension\_6f( Volt )
- puissance\_6f( Watt )
- energie\_6f( Watt-heure )
- tracker()
- courant\_6a( Ampère )
- temps\_6a( seconde )
- tension\_6a( Volt )
- puissance\_6a( Watt )
- energie\_6a( Watt-heure )
- TemperatureDS1820-1( celsius )

Choose all ?

Chosen variables ?



+ Remove all

- Créer un widget : étapes (titre-choisir la page- row et col (ne pas toucher) - size (largeur de la page) - content (chart 4)).

Home > PyScada HMI > Widgets > Add widget

### Add widget

Title:

Page:   

Row:

Col:

Size:

☒ Visible

Content:

- Rafraîchir la page principale, attendre l'apparition des graphiques.



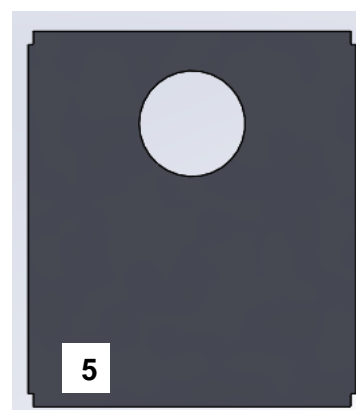
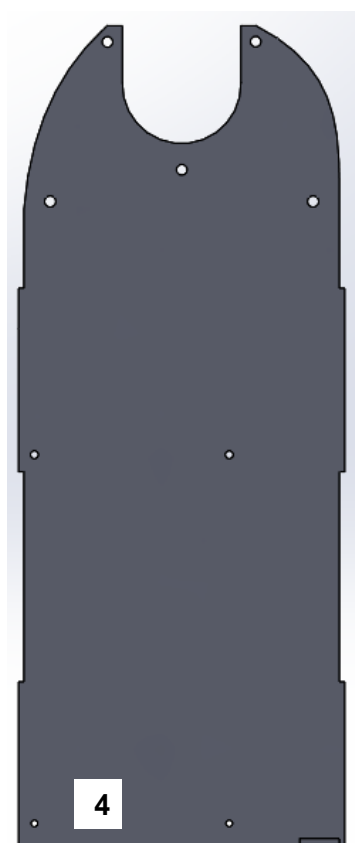
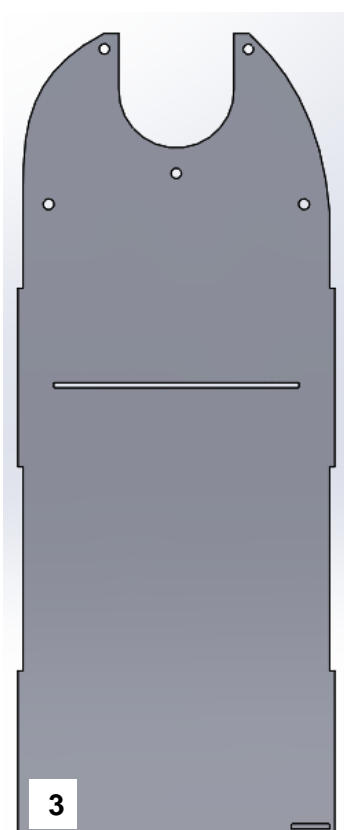
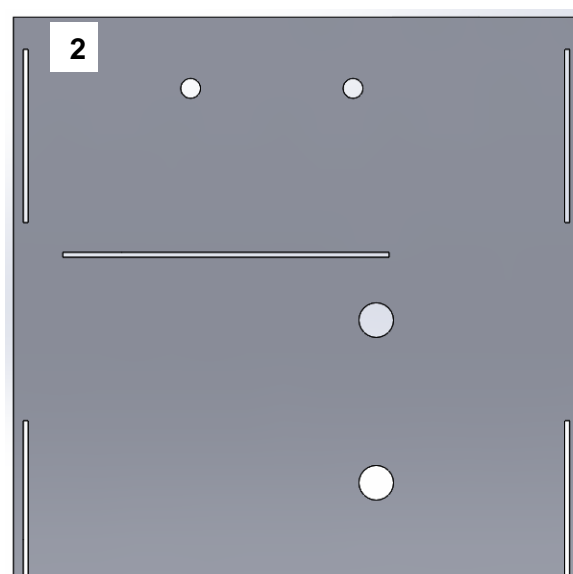
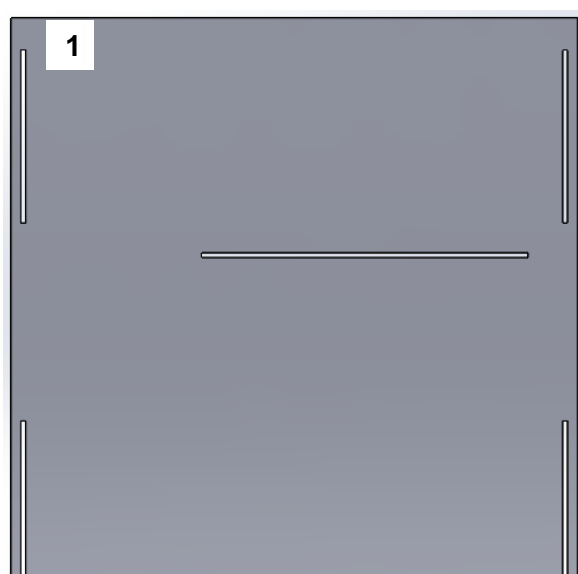
Méthode pour retrouver l'adresse d'une carte :

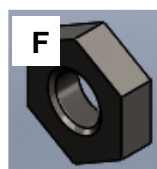
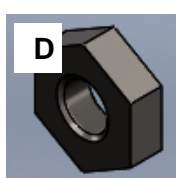
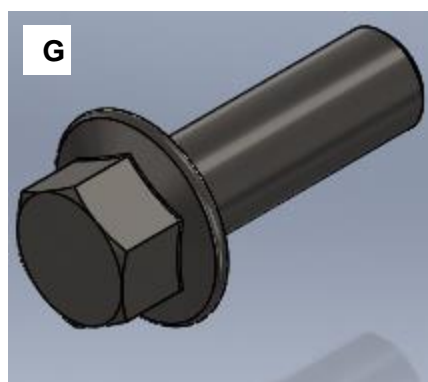
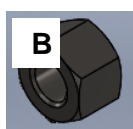
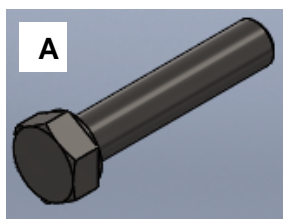
```
pi@raspberrypi:~ $ sudo i2cdetect 1
WARNING! This program can confuse your I2C bus, cause data loss and worse!
I will probe file /dev/i2c-1.
I will probe address range 0x03-0x77.
Continue? [Y/n] y
    0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
10:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
20:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
30:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
40:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
50:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
60:  -- -- -- -- -- -- 66 -- -- -- -- -- -- -- 6f
70:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
pi@raspberrypi:~ $
```

Taper la commande « sudo i2cdetect 1 » dans le logiciel PuTTY pour trouver l'adresse d'une carte.

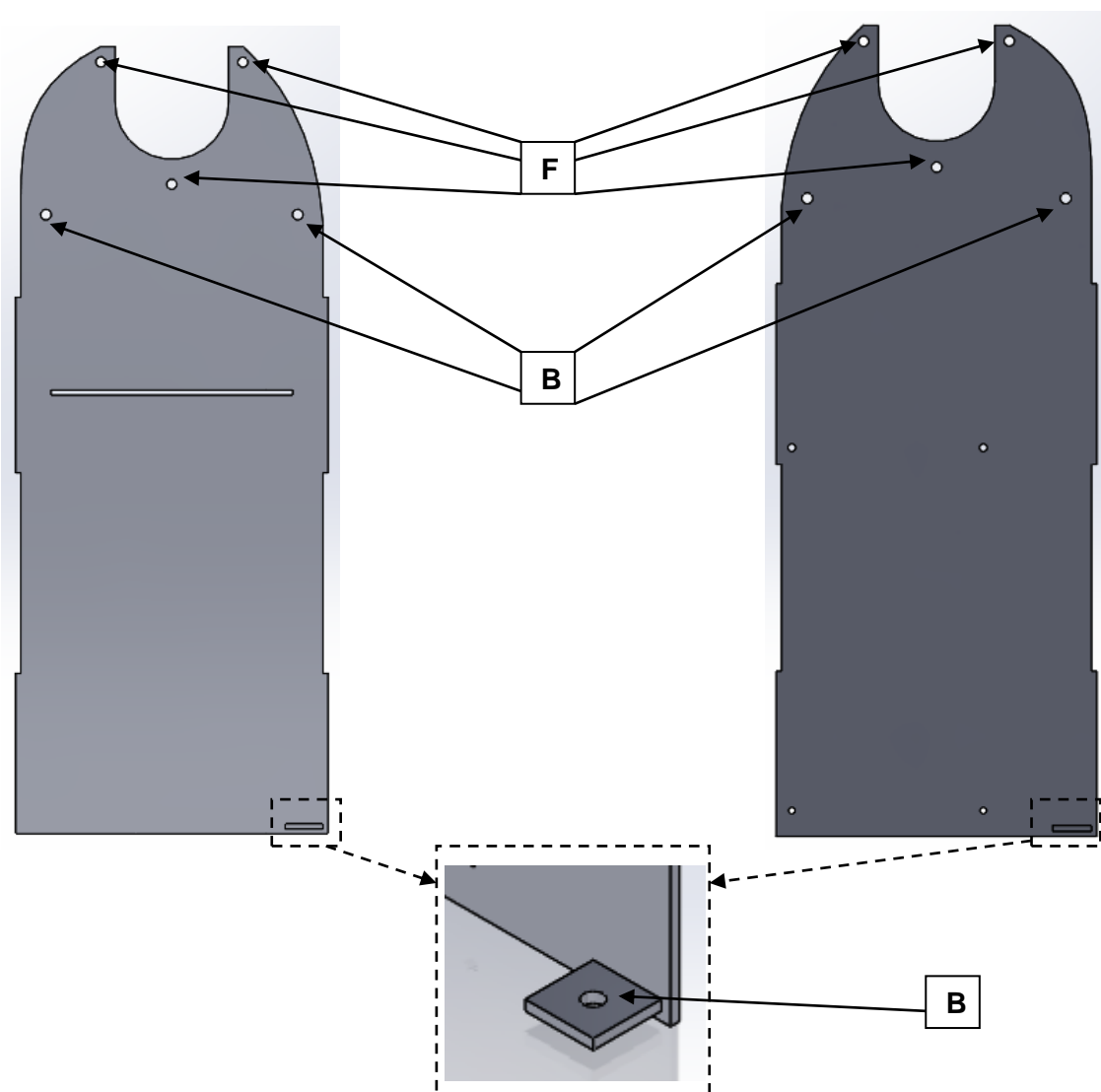
## Guide de montage SolidWorks

### Numérotation des plaques pour le sens du montage



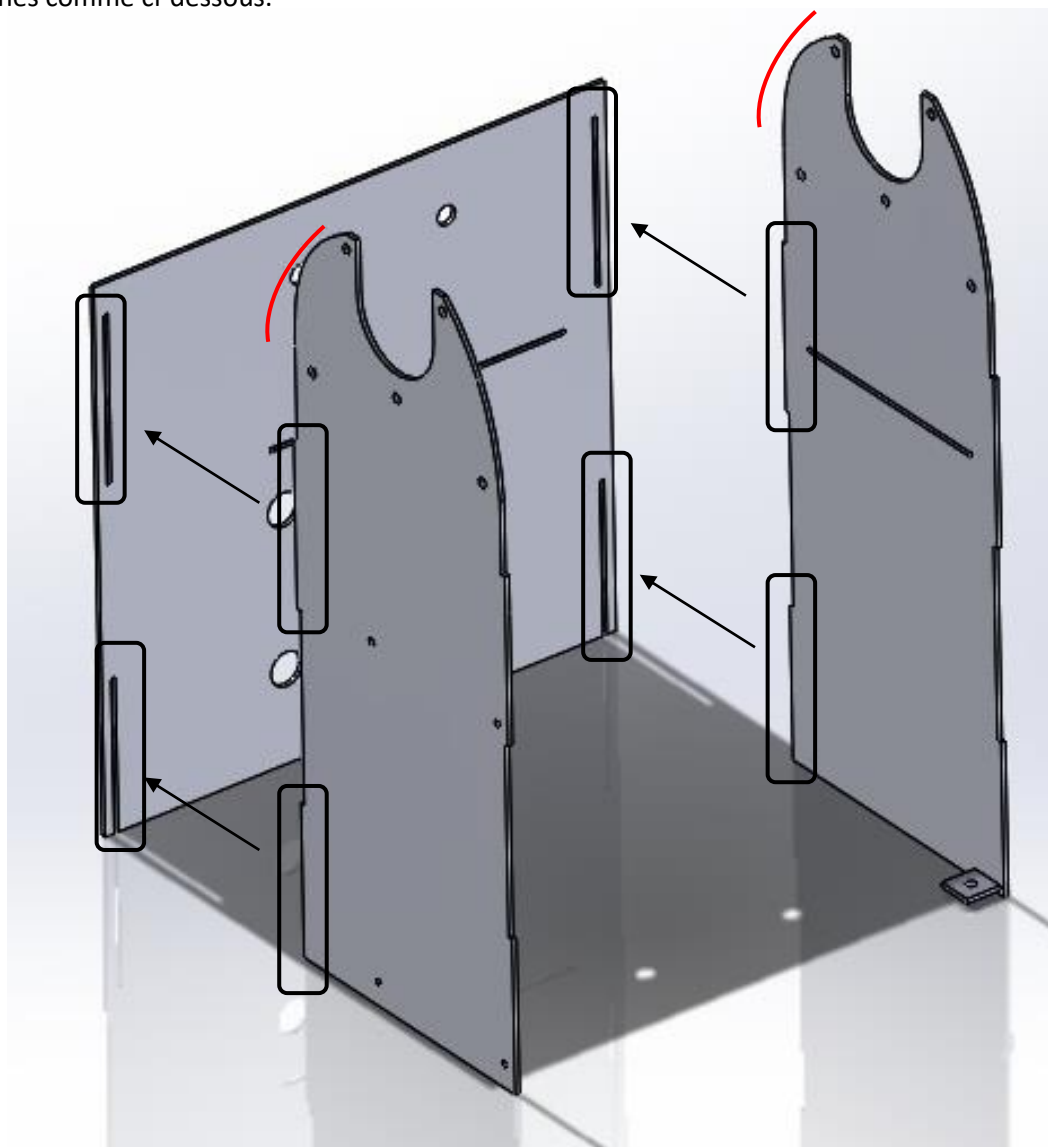
Liste des vis et écrous pour le guide de montage

- Souder tous les écrous qui, une fois le boîtier monté, seront à l'intérieur

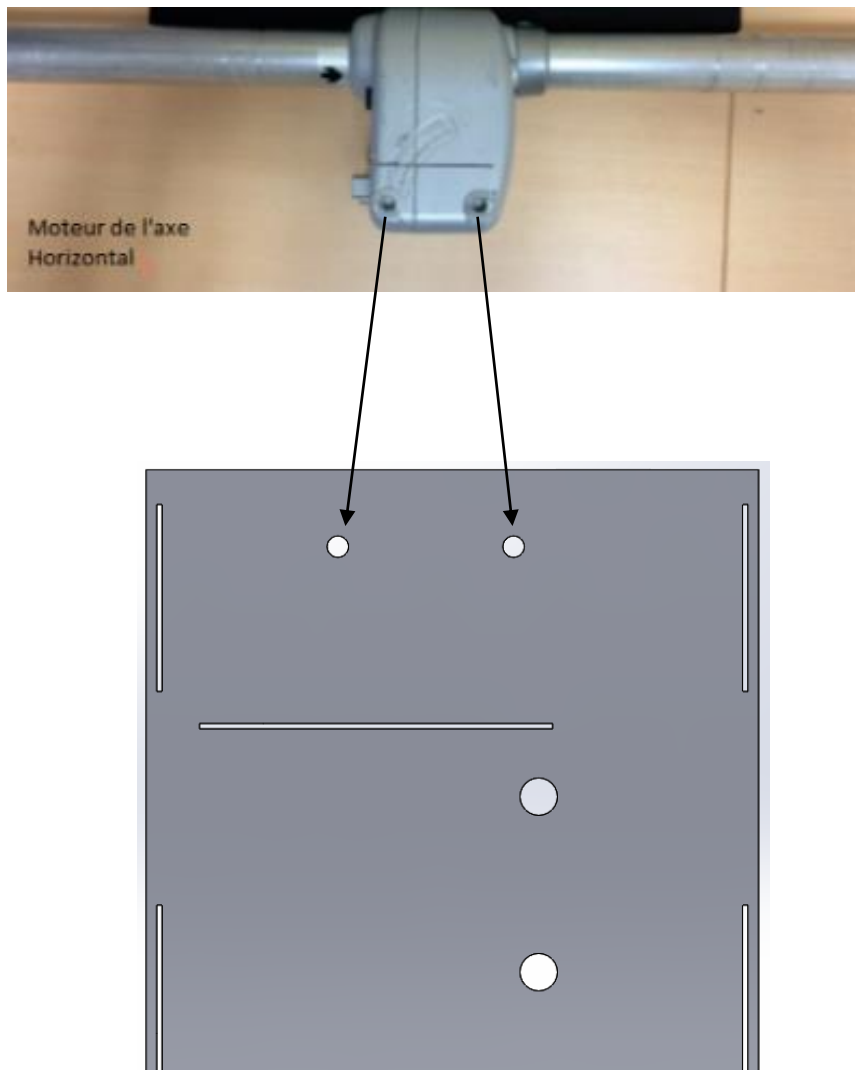


- Assembler les plaques 2, 3 et 4 puis souder au niveau des encoches sur les faces extérieures.

Attention au sens de montage, **la partie la plus bombée** des plaques 3 et 4 doit être face aux encoches comme ci-dessous.

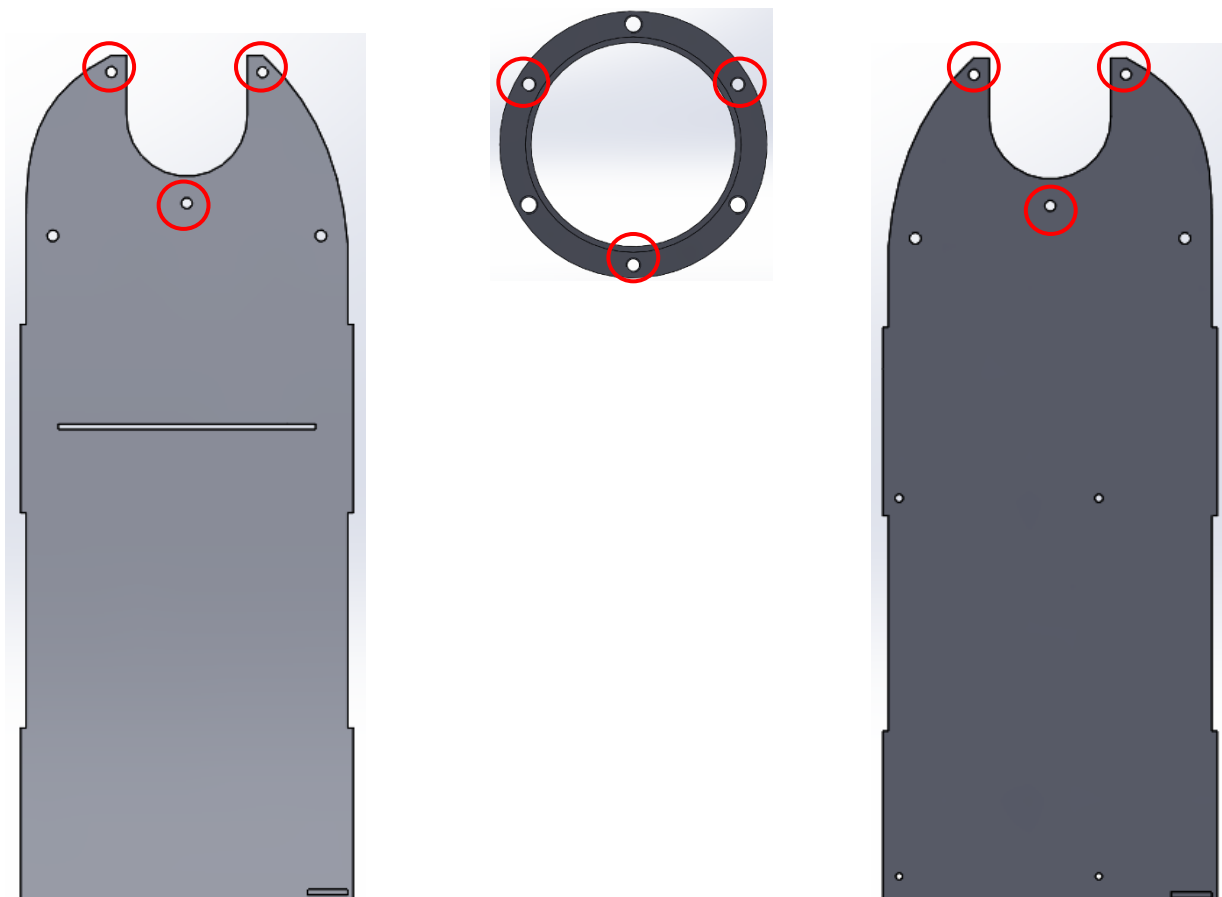


- Visser le moteur à axe horizontal à la plaque 2 avec deux vis G

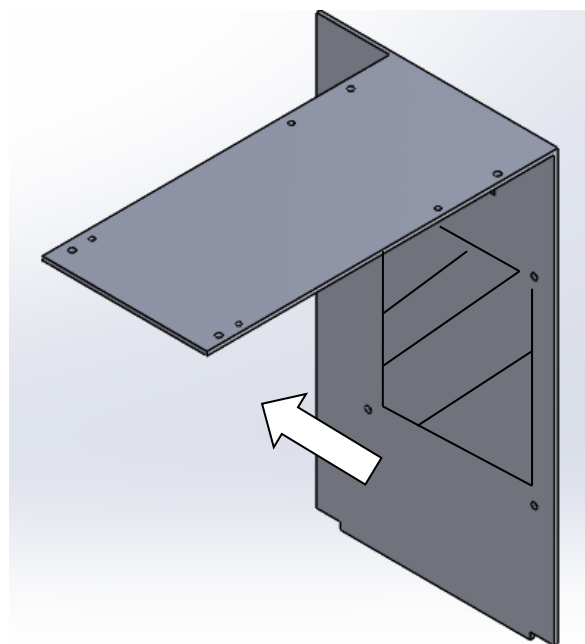
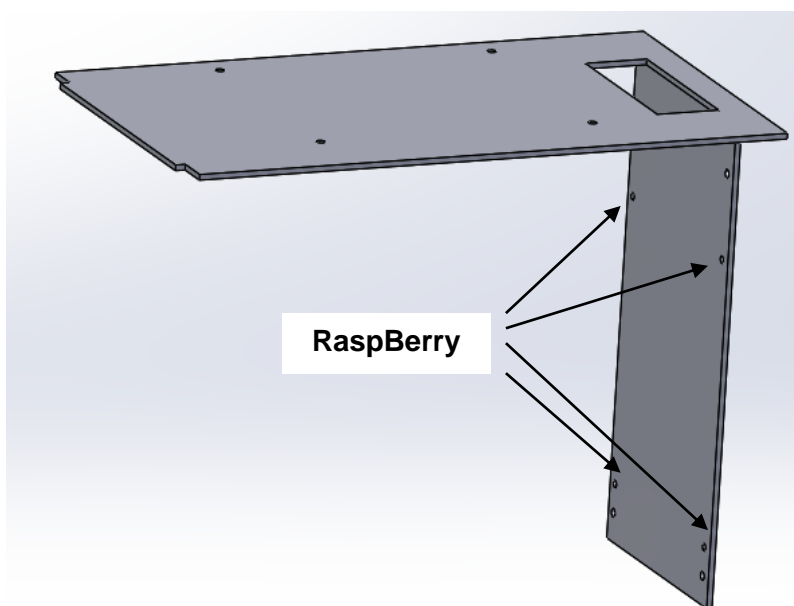


- Visser les cages roulements avec les vis E et insérer les roulements

La partie creusée doit être orientée vers l'extérieur du boîtier.



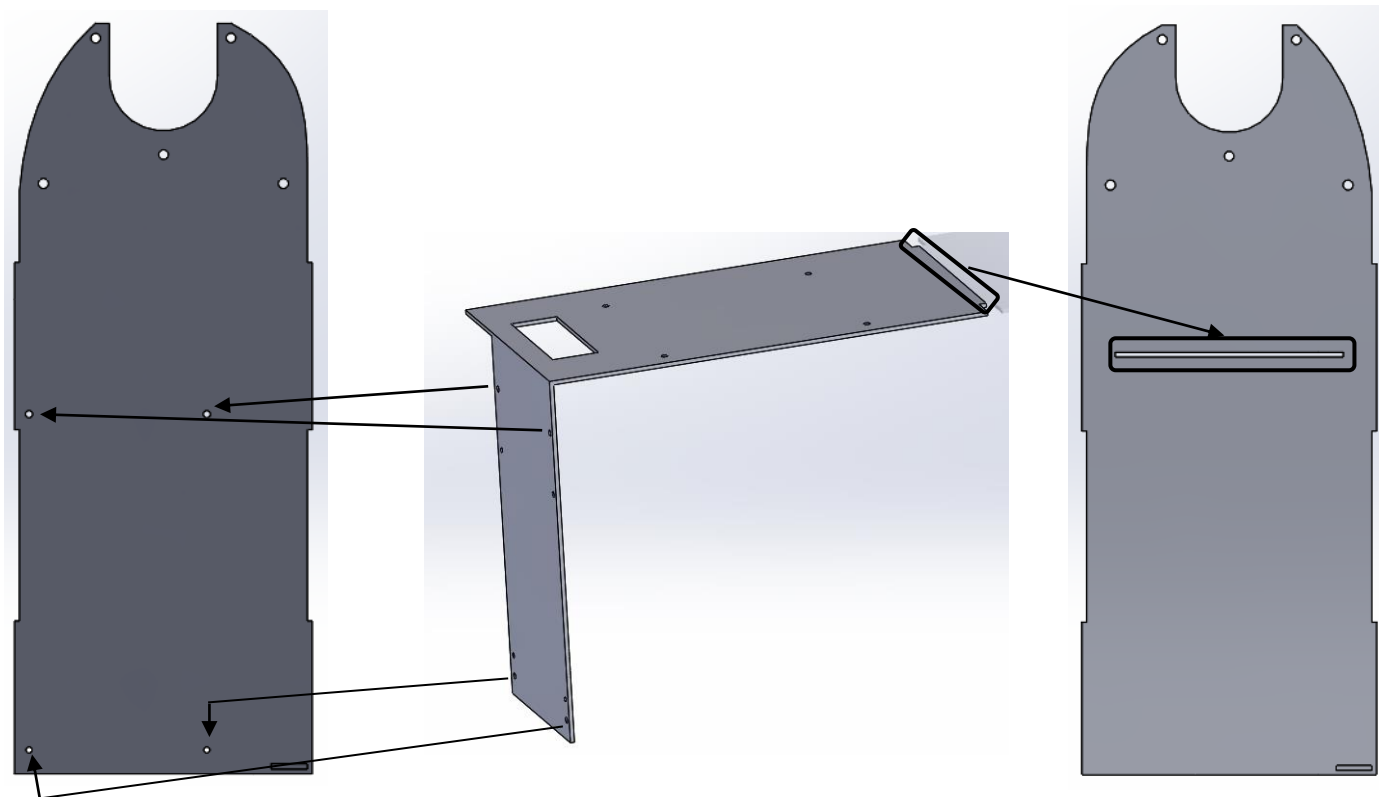
- Visser le boîtier du Raspberry et le MPPT sur la plaque à deux faces



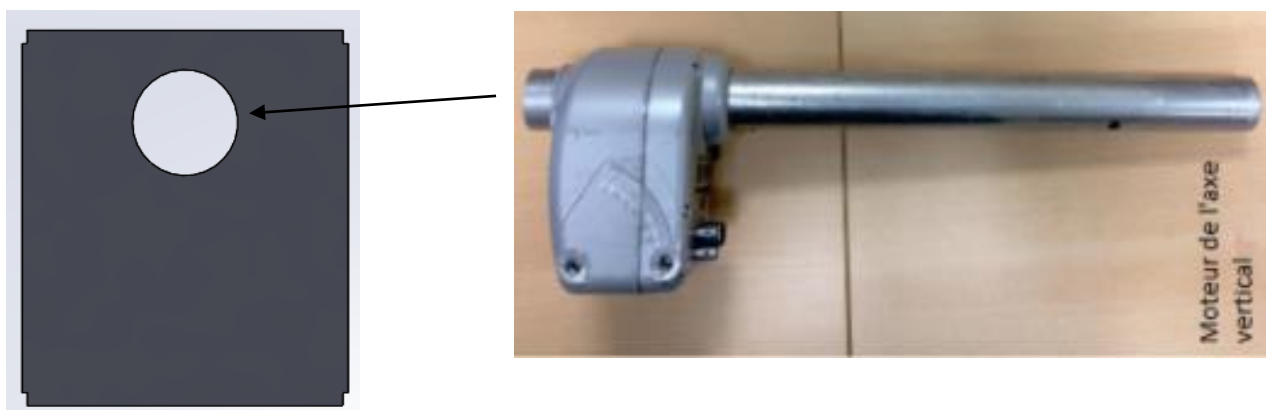
Placer les connections dans le sens de la flèche et la face métallique du MPPT contre la partie hachurée. Les deux plaques métalliques vont servir d'échangeur thermique pour refroidir le MPPT.



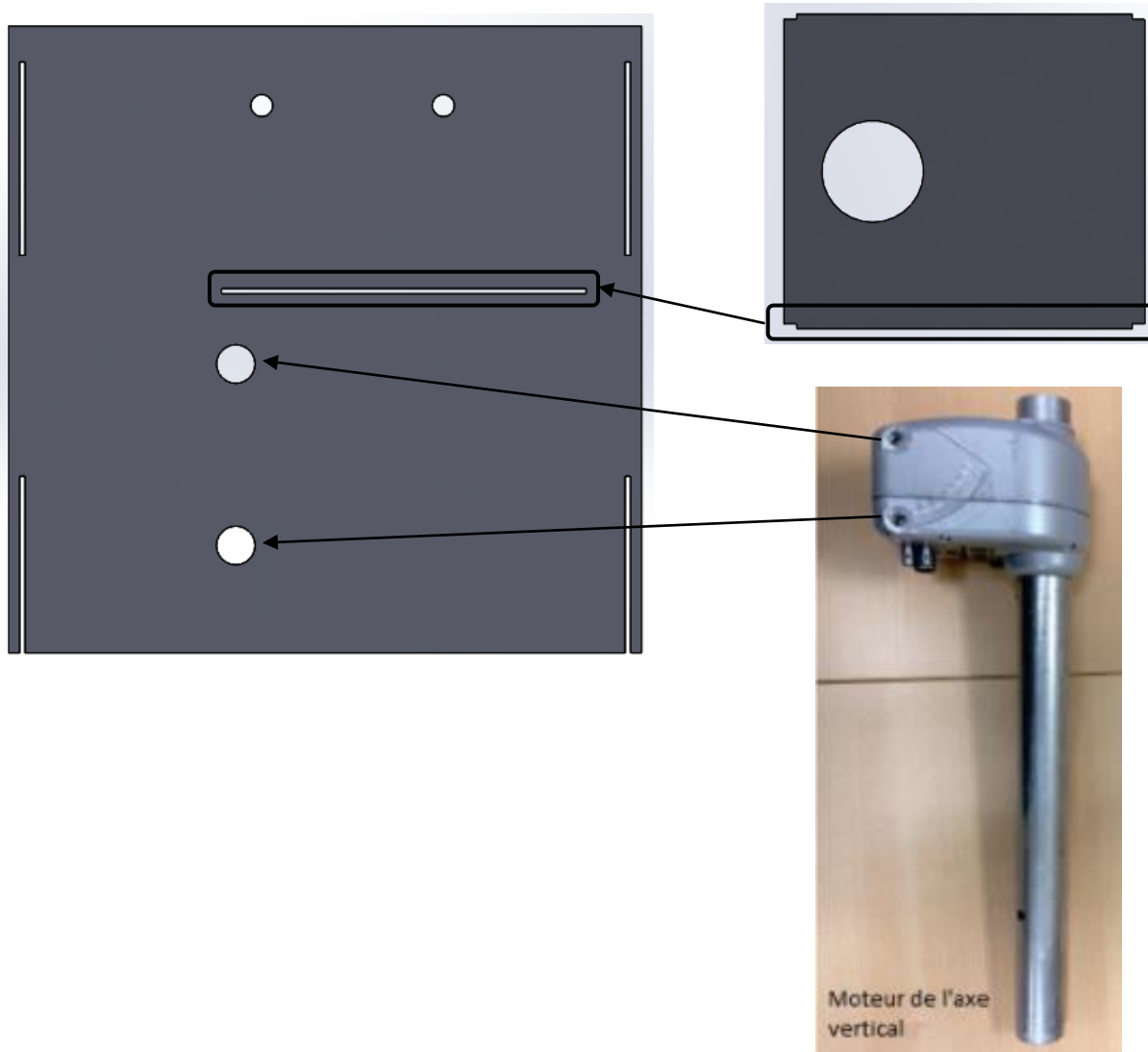
- Visser la plaque avec le MPPT et le Raspberry à la plaque 4. Imbriquée aussi la plaque 3 avec la plaque à deux faces puis souder.



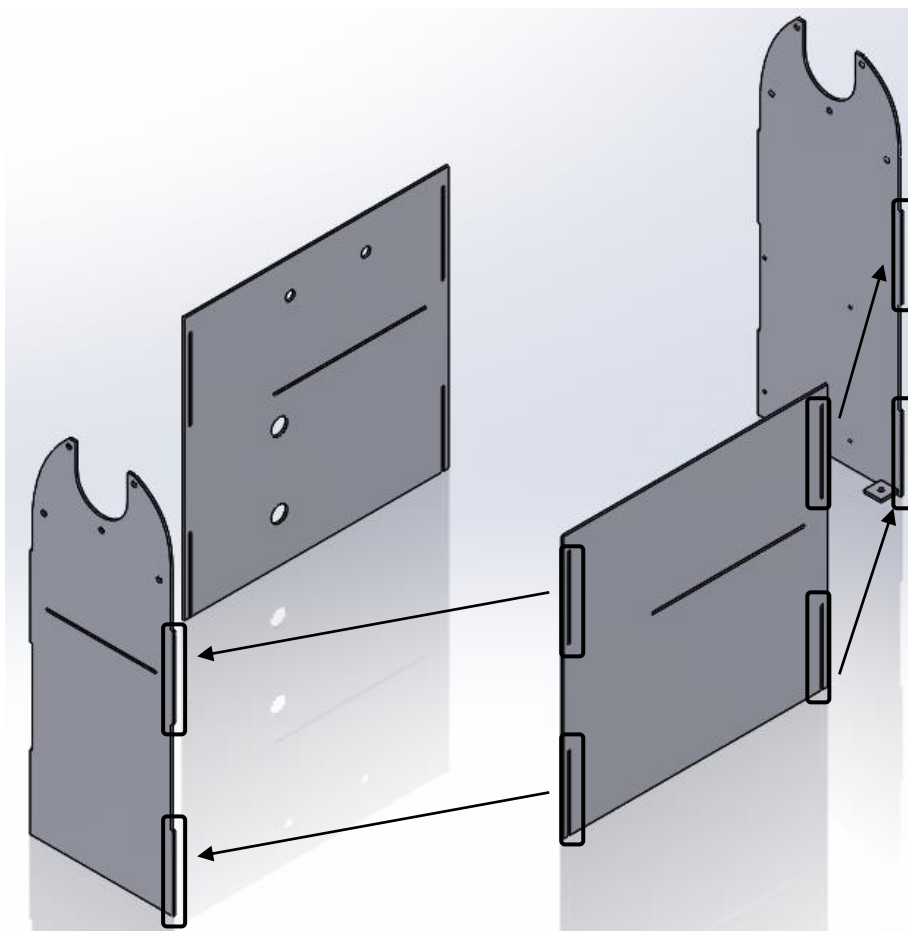
- Fixer le moteur à axe vertical à la plaque 5



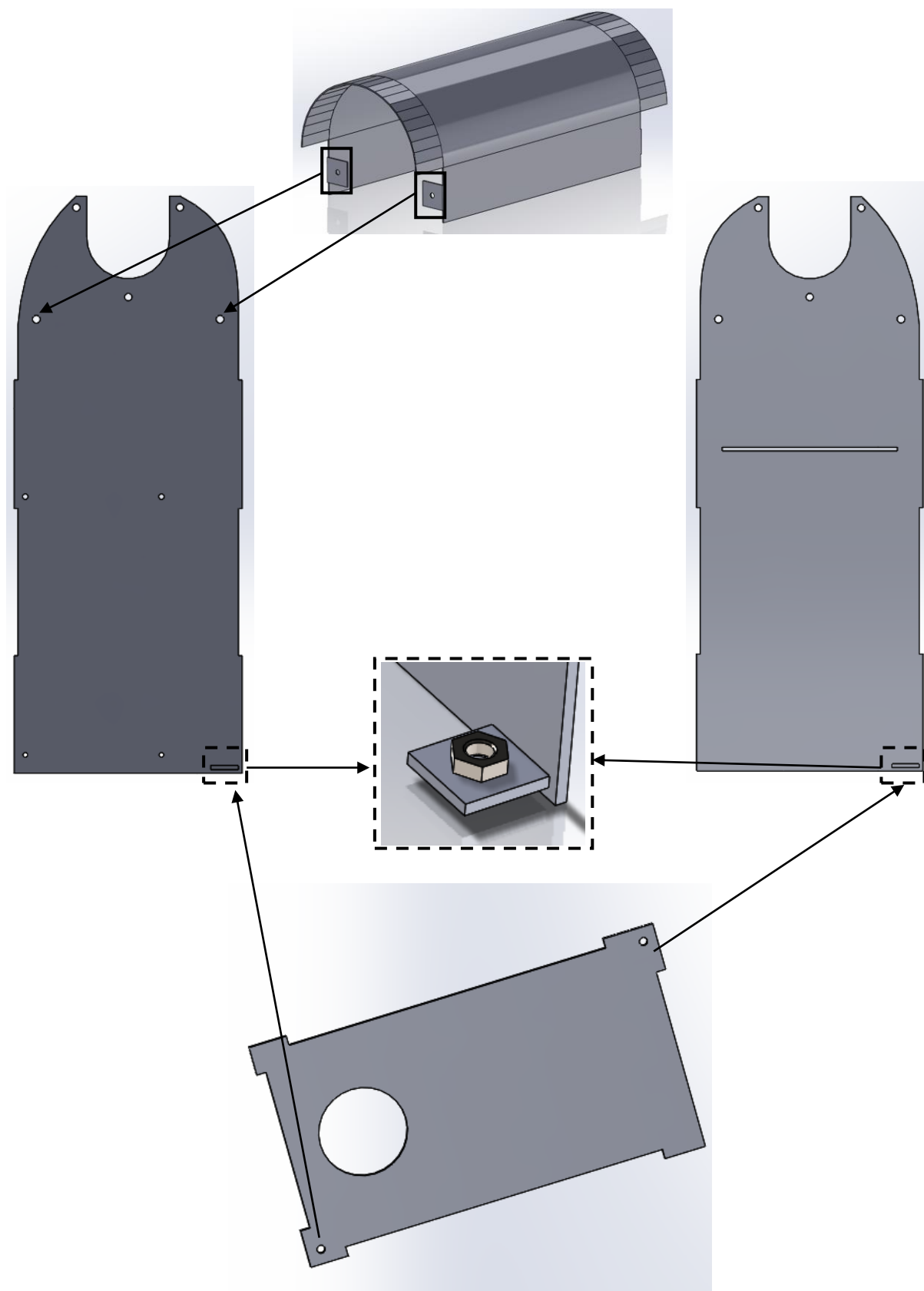
- Placer cet assemblage à l'intérieur du boîtier puis souder au niveau des encoches. Visser ensuite le moteur à la plaque 2 avec les vis G



- Fixer la plaque 1 au niveau des encoches puis souder

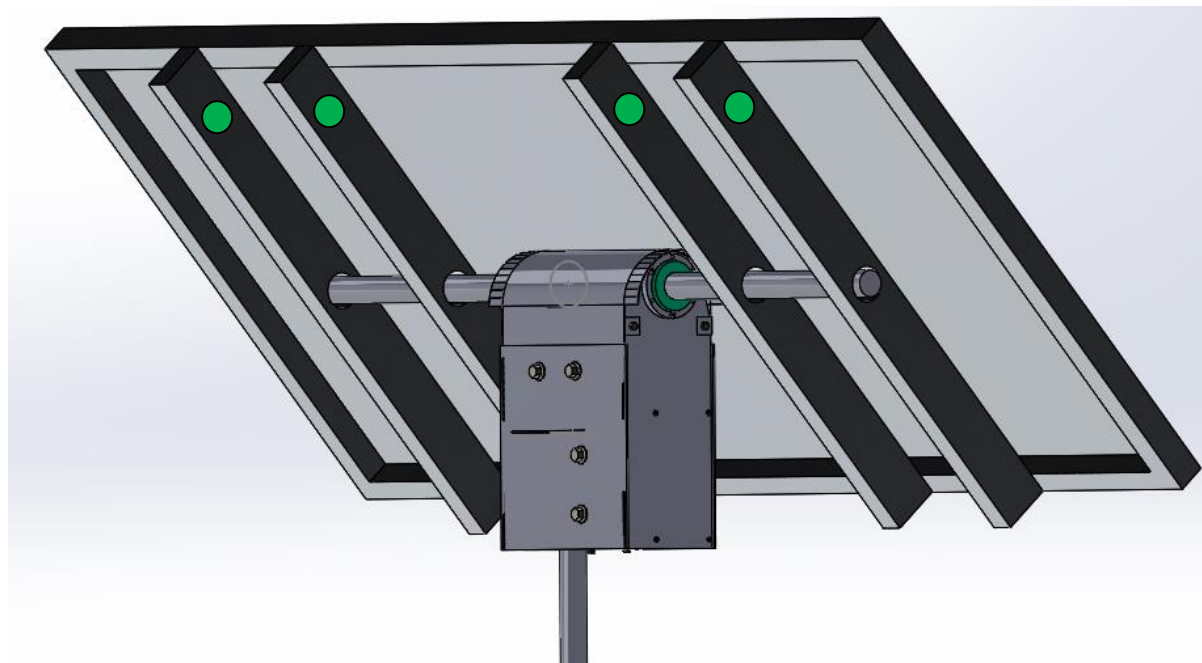


- Visser le couvercle du bas avec les vis A à la plaque 3 et 4. Visser ensuite celui du haut avec les vis A aux plaques 3 et 4.



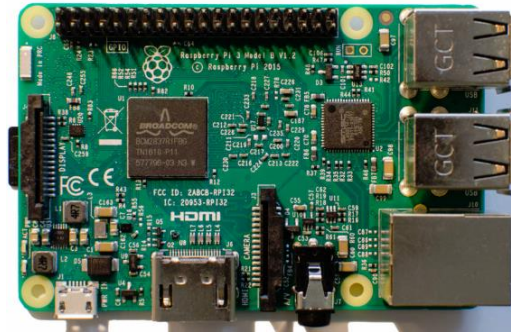
- Monter le panneau sur l'axe du moteur horizontal

Placer les quatre **fixations** sur l'axe horizontal puis monter le panneau dessus.



## Composants du tracker solaire

### Raspberry Pi :



Le Raspberry est un micro-ordinateur, créé par David Braben, de la taille d'une carte de crédit. Il est utile dans le domaine de la programmation de logiciel / jeux vidéo et le développement de page web. Il fonctionne sous le système d'exploitation GNU/Linux.

Le Raspberry que nous utilisons dispose de :

- 1 prise d'alimentation micro-USB (700mA)
- 4 ports USB
- 1 port réseau Ethernet (10 à 100 Mbits/seconde)
- 2 sorties vidéos (HDMI et prise RCA)
- 1 lecteur de carte micro SD
- 1 mémoire vive (RAM) de 512 Mo
- 40 broches GPIO (entrées/sorties) alimentées en 3.3 Vcc

Dans le cadre du projet, la Raspberry Pi va nous servir à :

- Stocker les données
- Connecter et communiquer avec les cartes de mesure.
- Communiquer avec la supervision

### Panneau à 2 axes :

Le tracker solaire est monté sur 2 axes motorisés (azimut et vertical) qui vont nous permettre d'optimiser son rendement, en suivant la course du soleil, car il nous sert de source d'énergie électrique continue. Le panneau a une surface de 2m<sup>2</sup> et peut résister à des vents allant jusqu'à 130 km/h.

### Servomoteurs :

Comme dit précédemment, ils vont incliner automatiquement le panneau en exploitant au maximum le rayonnement du soleil grâce à un positionneur intégré dans le moteur et aux 2 axes (1 horizontal et 1 vertical). Les moteurs solaires sont alimentés par une source externe continue 12 Vcc.

## Langage de communication Python

Python :



Le Python a été créé par Guido van Rossum en 1990 est un langage de programmation facile à prendre en main et souvent utilisé dans l'apprentissage de la programmation et dans le développement d'application web.

Le but du langage python est :

- D'être simple d'utilisation et intuitif mais également très puissant
- D'être open source pour que tout le monde puisse contribuer à son développement
- D'être adapté aux tâches quotidiennes, et permet des courts délais de développement
- D'être aussi compréhensible qu'un texte en anglais.

Dans notre projet le python nous servira à récupérer des informations envoyées sur le Raspberry depuis différents capteurs.