

|  |  |  |
| --- | --- | --- |
| **Auteur :** | **Encadrant :** | **Année:** |
| Aymeric Maurice Vincent Desnos | Pierre Gaucher | DII3 |

Rapport du projet électronique

Ce document caractérise le déroulement de notre projet électronique

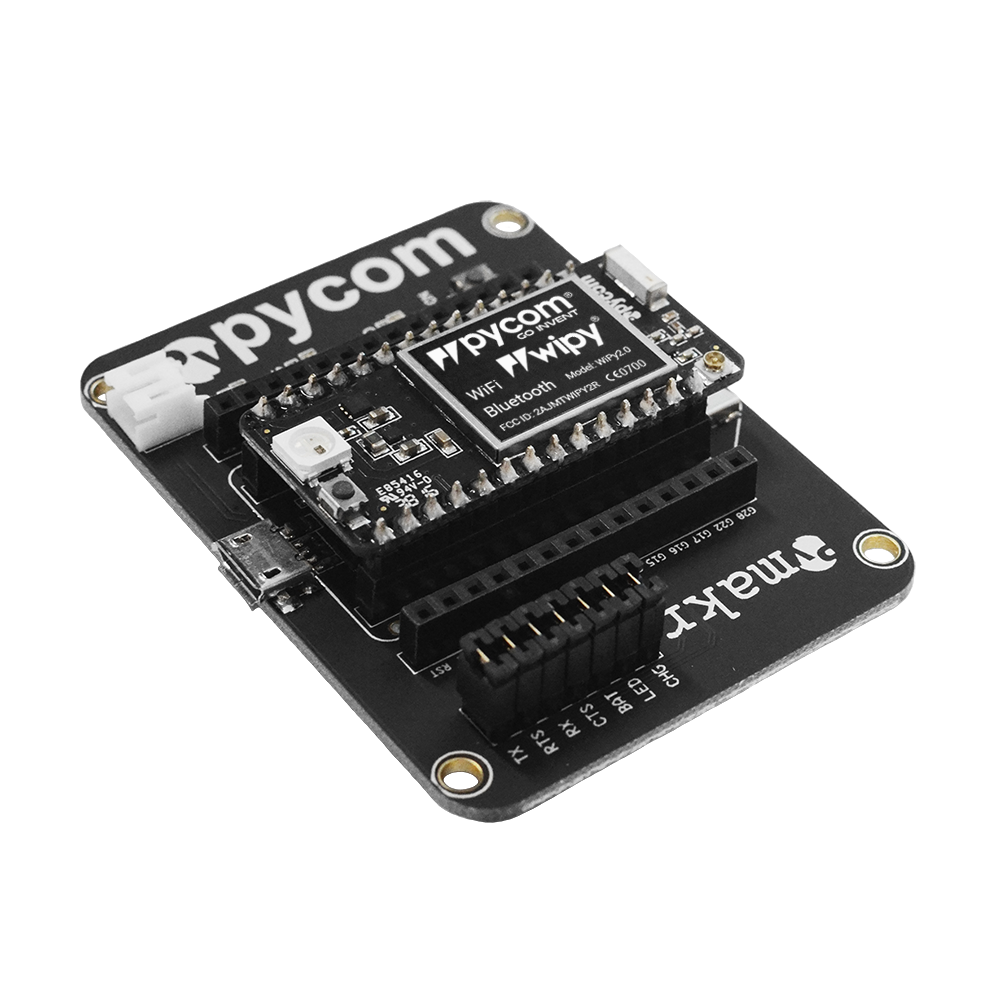


Table des matières

[Introduction 4](#_Toc517295012)

[Contexte 4](#_Toc517295013)

[DDC 5](#_Toc517295014)

[1. Identifiants du produit 5](#_Toc517295015)

[2. Nature du document 5](#_Toc517295016)

[3. Présentation du produit à développer 6](#_Toc517295017)

[4. Conception préliminaire du produit 6](#_Toc517295018)

[Description des fonctions et des signaux 7](#_Toc517295019)

[5. Conception détaillée du produit 7](#_Toc517295020)

[5.1. Captage de la distance 7](#_Toc517295021)

[5.2. Conditionnement numérique 8](#_Toc517295022)

[5.3. Captage vitesse rotation moteur 9](#_Toc517295023)

[5.4. Interfaçage I2C 10](#_Toc517295024)

[5.5. Commande en puissance 11](#_Toc517295025)

[5.6. Moteurs 11](#_Toc517295026)

[5.7. Stockage donnée sur la carte SD 12](#_Toc517295027)

[5.8. Interfaçage plug & play 13](#_Toc517295028)

[5.9. Traitement numériques 14](#_Toc517295029)

[5.10. Alimentation 14](#_Toc517295030)

[5.10.1. Signalisation présence alimentation 15](#_Toc517295031)

[5.10.2. Gestion recharge batterie 15](#_Toc517295032)

[6. Conception mécanique châssis du robot 15](#_Toc517295033)

[7. Planning 16](#_Toc517295034)

[8. Coût 17](#_Toc517295035)

[DDF 19](#_Toc517295036)

[9. Document de fabrication du produit 19](#_Toc517295037)

[9.1. Schéma électrique 19](#_Toc517295038)

[9.2. Nomenclature 20](#_Toc517295039)

[9.3. Fichier Gerber 20](#_Toc517295040)

[9.4. Plan perçage 21](#_Toc517295041)

[9.5. Schéma implantation 21](#_Toc517295042)

[9.6. Châssis robot 22](#_Toc517295043)

[DDV 23](#_Toc517295044)

[1. Nature du document 23](#_Toc517295045)

[2. Vérification du produit développé 23](#_Toc517295046)

[2.1. Mesure robot 23](#_Toc517295047)

[2.2. Autonomie batterie 23](#_Toc517295048)

[2.3. Test puissance moteur 24](#_Toc517295049)

[2.4. Test bus de communication I2C 25](#_Toc517295050)

[10. Matrice de conformité du produit 25](#_Toc517295051)

[Comparatif et description des outils de développement 27](#_Toc517295052)

[Electronique 27](#_Toc517295053)

[Mécanique 27](#_Toc517295054)

# Introduction

Nous avons choisi ce projet dans le but de découvrir un nouveau microcontrôleur, cela présente un opportunité d’élargir notre spectre de connaissance. Le python est un langage en pleine croissance, par sa simplicité et son indentation bien structuré. Il permet d’acquérir de très bon automatisme. Pour nous se former sur du python présente un atout majeur dans notre formation , car ce langage est apprécié par de plus en plus en entreprise et montre une certaine polyvalence.

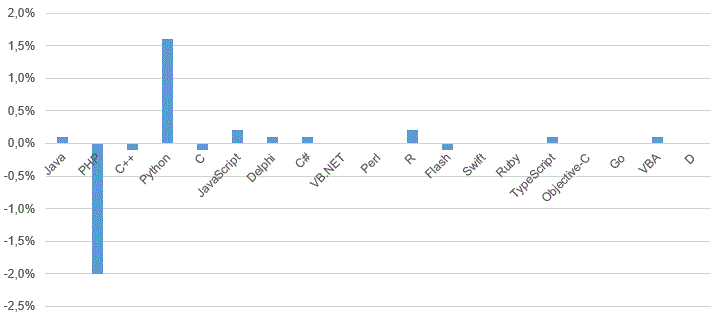


Figure 1:taux de fréquentation des rubriques associées aux différents langages de programmation en 2015 et 2016

L’intention de ce projet est aussi de nous faire découvrir le “Micropython” : il s’agit une implémentation logicielle du langage de programmation Python 3 , écrit en C , optimisé pour fonctionner sur un microcontrôleur. MicroPython est un compilateur Python complet et une exécution qui s'exécute sur le matériel du micro-contrôleur. Avec ceci, nous verrons aussi les différentes librairies du micro-contrôleur (I2C, ADC, GPIO, Accéléromètre ...), ainsi que les ressources internes disponibles sur ce type de carte. Le micro-contrôleur que l’on va utiliser est la “Pyboard”.

# Contexte

Ce robot aura pour objectif de faire découvrir à un public jeune, avec un langage simple et intuitif, la robotique d’un point de vue hardware et software. Aussi le partage d’une passion, et permettre de faire découvrir ce que l’on peut réaliser dans des filières comme en DII ou DI à Polytech Tours. Pour nous, ce projet s’inscrit dans notre cursus et représente une partie conséquente de notre UE Informatique. Il constitue aussi un moyen de travailler dans un domaine qui nous correspond plus que sur les matières classiques. Ce premier projet est en quasi total autonomie pour nous en apprendre un peu plus sur la gestion de projet et nous faire comprendre l’importance des outils et compétences nécessaires à mettre en œuvre pour réussir le projet. Notre client, Pierre Gaucher, nous laisse une grande liberté pour le coté rédactionnel du projet. Cependant, ceci doit suivre une démarche ingénieur comme on l’a vu en cours ( mais aussi ce que nous n’avons pas encore vu).

# DDC

# Identifiants du produit

**Nom du produit :** Programmation microcontrôleur en Python

**Référence du produit** : Robot\_Python

**Projet :** Programmation microcontrôleur en Python

**Client :** Pierre GAUCHER

# Nature du document

Ce document est un cahier des charges et a pour but de décrire l'ensemble des exigences client relatives au développement du produit.

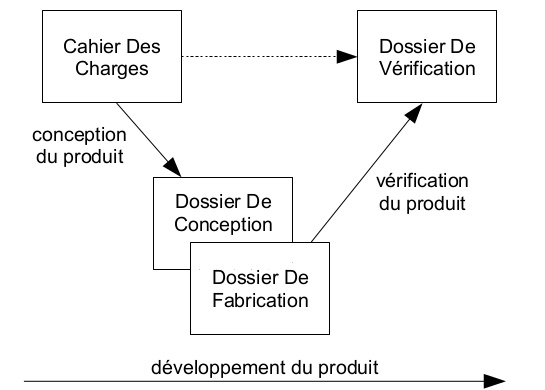


Figure 1: Arborescence documentaire

La figure ci-dessus fournit une vision d'ensemble de l'arborescence documentaire du projet*.* Ceci permet ainsi de mieux comprendre la nature de ce document et son positionnement dans le développement en *V* du produit.

# Présentation du produit à développer

Le produit à développer est une base robotique pédagogique programmé en Python avec un microcontrôleur WiPy 2.0. Ce robot devra être Plug&Play c'est-à-dire qu’il sera possible de changer, remplacer ou même ajouter différents composants et périphériques sur le robot. Un shield devra être conçu pour pouvoir accueillir certain type de composants explicités dans ce cahier des charges. Ce robot aura pour objectif de faire découvrir à un public jeune, avec un langage simple et intuitif, la robotique d’un point de vue hardware et software.

# Conception préliminaire du produit

Ce chapitre décrit l'architecture fonctionnelle du produit. Il apporte les premiers éléments de preuve de la faisabilité du produit vis-à-vis des exigences client.

**Référence de préconception:** **PRC01**

**Exigences client vérifiées : EXIG\_ROBOT\_DIMENSIONS, EXIG\_ROBOT\_ROUE, EXIG\_ROBOT\_PLUG&PLAY, EXIG\_ROBOT\_ENERGIE, EXIG\_ROBOT\_RECHARGE, EXIG\_ROBOT\_BUS, EXIG\_ROBOT\_ENCODEUR, EXIG\_ROBOT\_µc, EXIG\_ROBOT\_STOCKAGE, EXIG\_ROBOT\_MOTEUR, EXIG\_ROBOT\_PUISSANCE, EXIG\_ROBOT\_INDICATEUR**

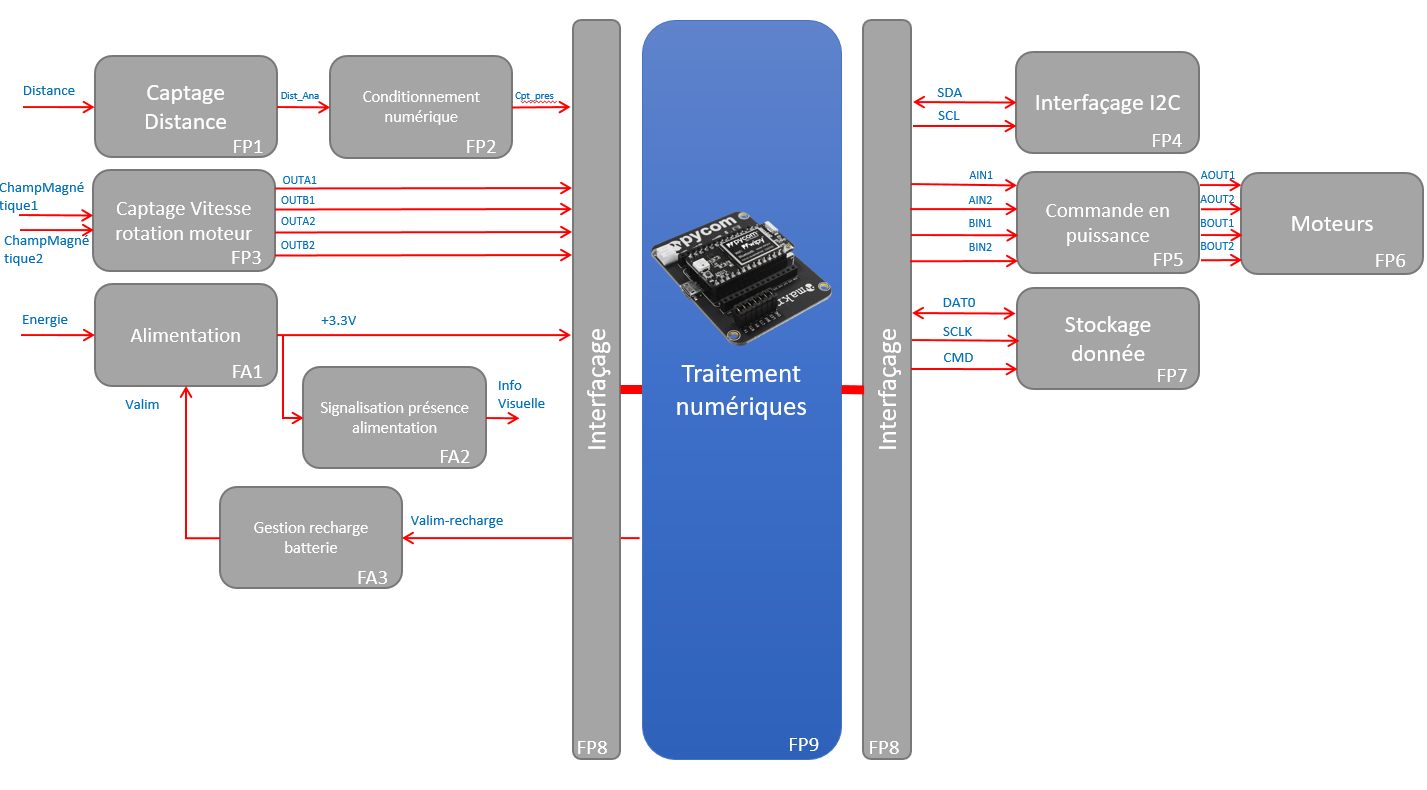
****

Figure 2: SF1D

Outre une alimentation autonome (FA) imposée par l'exigence **EXIG\_ROBOT\_ENERGIE**, l'architecture du produit s'articule en sept blocs fonctionnels :

### Description des fonctions et des signaux

Voir le tableau de descriptions des signaux disponible dans le répertoire : **Robot\_Python/Electronique/Tableaux de description des fonctions et des signaux**

# Conception détaillée du produit

Ce chapitre détaille la conception du produit développé. Il constitue une preuve de la conformité du produit.

Chaque paragraphe de cette étude fait donc clairement référence aux exigences client issues du [CDC].

Pour nos choix technique nous avons accès à une liste site : Mouser, Farnell, Roboshop.

## Captage de la distance

**Référence de conception:** CCPT01

**Exigences client vérifiées : EXIG\_ROBOT\_BUS, EXIG\_ROBOT\_PLUG&PLAY**

Le captage à distance doit répondre à plusieurs critères :

* Compatible avec une tension de 3,7 V
* Capter à une distance d’au moins 10 cm
* Être compatible avec le µC
* Il faut 3 à 4 capteurs
* Facile à remplacer

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Capteur | Aperçu | Tension alim | Distance max | Compatible avec le µC ? | Consommation | Type de sortie |
| **GP2Y0D815Z0F IR Sharp** | **pololu-carrier-sharp-gp2y0d815z0f-ir-range-sensor-05-15cm** | **2.7V - 6.2V** | **15 cm** | **oui** | **5 mA** | **Numérique** |
| Sharp GP2Y0A51SK0F | sharp-gp2y0a51sk0f-analog-distance-sensor-2cm-15cm | 4.5V - 5.5V | 15cm | non | 12 mA | Analogique |
| Sharp GP2Y0A60SZLF | ar-module-gp2y0a60szlf-3v-10-150cm-22267 | 2,7V - 3,6V | 150cm | oui | 33 mA | Analogique |

Parmi les 3 capteurs qui ont été retenu, le **GP2Y0D815Z0F** est le plus adapté à notre besoin : malgré sa faible portée, il est facile à utiliser car c’est un capteur numérique (TOR), sa consommation est beaucoup moins élevée que les autres, on prolonge l’autonomie du robot. On pourra en utiliser 4 sans consommer plus de 20 mA sur la batterie. Enfin, il est compatible avec la tension d’alimentation de la carte qui est de 3,7V.

Les 3 capteurs respectent exigences “plug and play” car on peut les brancher et les débrancher grâce aux pins fournis avec.

## Conditionnement numérique

**Référence de conception:** CCPT02

**Exigences client vérifiées : EXIG\_ROBOT\_BUS, EXIG\_ROBOT\_µc**

Pour s’assurer du bon conditionnement de nos capteurs de distances avec le microcontrôleur, on doit s’assurer que que les caractéristiques d’entrées et sorties soient compatible:

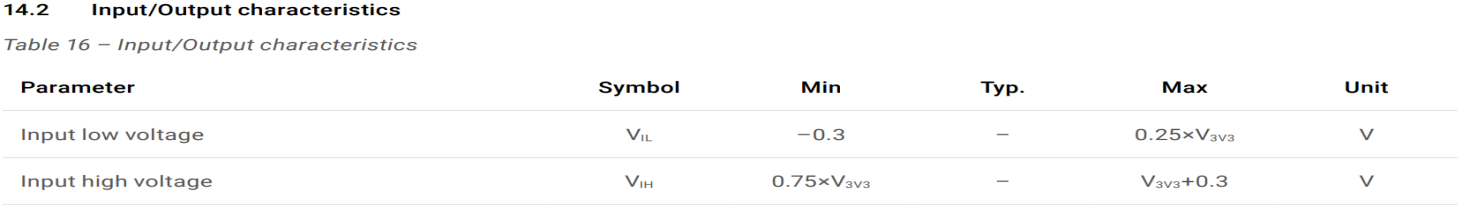
****

Figure 3: Caractéristique entrée Wipy 3.0

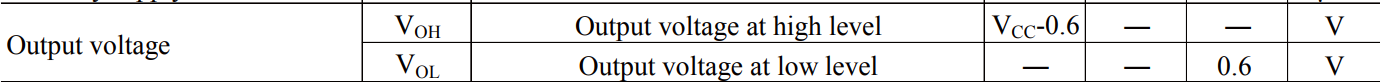


Figure 4: Caractéristique sortie capteur distance **GP2Y0D815Z0F**

Avec l’aide des tableaux ci-dessus on observe les plages de tension suivantes:

|  |
| --- |
| **Wipy 3.0**  **GP2Y0D815Z0F**  VOH = 2,7V  VOL = 0,6V |
| Vihmax= 3,6V  Vihmin= 2,475V |
| Zone de transition |
| Vilmax= 0,825V  Vilmin = -0,3V |

Le conditionnement entre le capteur numérique et le µC est **conforme** car les tensions du capteur de distance VOH et VOL sont comprises dans les intervalles du µC.

## Captage vitesse rotation moteur

**Référence de conception:** CCPT03

**Exigences client vérifiées : EXIG\_ROBOT\_ENCODEUR, EXIG\_ROBOT\_PLUG&PLAY**

Afin de connaitre la vitesse de rotation des moteurs, des encodeurs seront prévus à cet effet, ils doivent :

* Être compatible à une tension d’alimentation 3,7V
* Mesurer le nombre de tours des moteurs

La compatibilité 3,7V réduit beaucoup le choix:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Encodeur | Aperçu | Tension alim | Compatible avec le µC ? | Consommation | Type de sortie |
| **SKU:SEN0116** | **/media/vincent/Data/Ubuntu/Documents/projetDev/encodeur.jpgencodeur** | **3,3(short R4) ou 5 V** | **oui** | **4mA** | **Numérique** |
| Encodeur RS034 | encodeurPourri | 3 à 24 Vcc | oui | 4mA | Numérique |

Le SKU:SEN0116 est plus adapté car il est plus robuste que le le RS034, de plus un code exemple sont disponible : <https://www.dfrobot.com/wiki/index.php/Mini_Robot_chassis_Encoder_(SKU:SEN0116)>

Cependant c’est un code en C basé sur Arduino, il est nécessaire de le convertir en Micropython pour pouvoir l’exploiter :

1. """
2. The sample code for driving one way motor encoder.
3. link:
4. A pin -- digital 2
5. B pin -- digital 4
6. See the link for detail:
7. http://docs.micropython.org/en/v1.9.3/esp8266/esp8266/tutorial/pins.html
8. """
10. **import** time
11. **from** machine **import** Pin
13. encoder0pinA = 0 # A pin -> the interrupt pin
14. encoder0pinB = 2 # B pin -> the digital pin
15. encoder0PinALast = None
16. duration = None
17. direction = None
19. **def** encoderInit():
20. direction = True
21. pA = Pin(encoder0pinA, Pin.IN)
22. pB = Pin(encoder0pinB, Pin.IN)
23. p0 = Pin(0, Pin.IN)
24. p0.irq(trigger=Pin.IRQ\_RISING | Pin.IRQ\_FALLING, handler=wheelSpeed)
26. **def** wheelSpeed():
27. Lstate = pA.value()
28. **if**((encoder0PinALast == 0) **and** Lstate== 1):
29. val = pB.value()
30. **if**(val == 0 **and** direction):
31. direction = False
32. **elif**(val == 1 **and** **not** direction):
33. direction = True
34. encoder0PinALast = Lstate
35. **if** **not** direction:
36. duration += 1
37. **else**
38. duration -= 1
40. **while** True:
41. **print**("Pulse: " + duration)
42. duration = 0
43. time.sleep\_ms(100)

La fonction wheelSpeed permet de récupérer le nombre de pulse sachant que 12 pulse équivaut à 1 révolution donc un tour de moteur. Le schéma de câblage est également disponible :

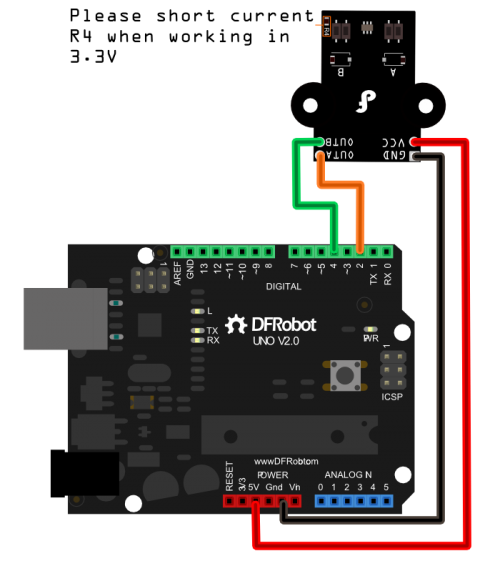


Figure 5: Câblage SEN0116

## Interfaçage I2C

**Référence de conception:** CCPT04

**Exigences client vérifiées : EXIG\_ROBOT\_BUS, EXIG\_ROBOT\_PLUG&PLAY**

Comme on peut le voir sur le schéma MultiSim, le µC possède un bus I2C, il sera exploité dans ce projet pour interfacer les différents périphériques I2C présent ou non sur la carte. Pour répondre à l’exigence plug and play, on utilisera des connecteurs 4 pin classique comme ci-dessous (mais découpé) car ce sont les plus communs, et ils sont imposés par le client:



Figure 6: Connecteur mâle

Il y aura au total 5 périphériques I2C.

## Commande en puissance

**Référence de conception:** CCPT05

**Exigences client vérifiées : EXIG\_ROBOT\_PUISSANCE, EXIG\_ROBOT\_PLUG&PLAY**

La commande en puissance de nos moteurs, sera réalisée par un driver de moteur qui doit respecter les contraintes suivantes :

* Commander 2 moteurs CC avec une tension de 3,7V dans les deux sens de rotation
* Être compatible avec une tension 3,7V

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Driver | Aperçu | Tension alim | Compatible avec le µC ? | Courant par moteur | Type communication |
| **Adafruit DRV8833** | **/home/vincent/Images/ori-driver-moteur-pas-a-pas-ada3297-25543.jpgori-driver-moteur-pas-a-pas-ada3297-25543** | **2,7 à 10,8 Vcc** | **oui** | **1,2A** | **Pont en H classique** |
| ROB-13911 | /home/vincent/Images/driveri2c.jpgdriveri2c | 3,3 Vcc | oui | 1,2A | I2C |

Encore une fois la contrainte d’une tension d’alimentation 3,7V réduit pas mal les possibilités. Deux drivers de moteurs ont été retenus, un que l’on peut contrôler par I2C, ce qui divise par 2 le nombre de pin à allouer. Et l’autre qui nécessite 4 pins sur la carte mais qui est plus simple à utiliser et surtout moins chère. Nôtre choix s’est porté sur le DRV8833 pour ces deux raisons. Les signaux sont détaillez sur la SF1D, et sur le tableau de description des signaux.

## Moteurs

**Référence de conception:** CCPT06

**Exigences client vérifiées : EXIG\_ROBOT\_MOTEUR, EXIG\_ROBOT\_ROUE**

Les moteurs choisis doivent respecter les contraintes suivantes :

* Être compatible avec une tension 3,7V
* Les roues doivent être compatibles avec le moteur

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Capteur | Aperçu | Tension alim | Compatible avec le µC ? | Courant consommé par moteur (bloqué) | Vitesse | Roue |
| **Jeu de Roues et Moteurs MiniQ** | **/home/vincent/Images/miniq-motor-wheel-set-encoder_1.jpgminiq-motor-wheel-set-encoder_1** | **3 à 6 Vcc** | **oui** | **1,5A sous 3 Vcc** | **260 tr/mn à 6 Vcc** | **Diamètre des roues: 42 mm** |
| Dagu Kit Moteur | /home/vincent/Images/ori-kit-moteurs-roues-encodeur-rs034-21383.jpgori-kit-moteurs-roues-encodeur-rs034-21383 | 3 à 6 Vcc | oui | 1,5A sous 3 Vcc | 135 tr/mn à 6 Vcc | Diamètre: 65 mm. |

Les moteurs MiniQ ont été retenus car ils sont plus robuste (engrenages métallique), plus petit. De plus, les encodeurs sont de meilleures qualités. Un autre plus est la vitesse de rotation qui doublée, ce qui signifie que a 3,7V le moteur tournera à une vitesse correcte.

## Stockage donnée sur la carte SD

**Référence de conception:** CCPT07

**Exigences client vérifiées : EXIG\_ROBOT\_STOCKAGE**

La carte SD ayant déjà été fournie, nous n’avons pas besoin de faire de comparatif. Pour savoir comment stocker des données sur cette carte.

Au niveau des connexions nécessaire les pins P8: DAT0, P23: SCLK et P4: CMD, seront réserver à l’utilisation de la carte SD.

Comme on possède une carte d’extension, on a juste à insérer la carte SD puis lancer un script.

**Remarque :** **La carte SD doit être formatée en FAT32 ou FAT16**

Un code exemple est déjà disponible sur le projet développement réalisé en amont, voici ce qui va nous permettre de valider la faisabilité de cette exigence:

1. **from** machine **import** SD
2. **import** os
4. sd = SD()
5. os.mount(sd, '/sd')
7. # check the content
8. os.listdir('/sd')
10. # try some standard file operations
11. f = open('/sd/test.txt', 'w')
12. f.write('Testing SD card write operations')
13. f.close()
14. f = open('/sd/test.txt', 'r')
15. f.readall()
16. f.close()

Sachant que le CDC ne spécifie pas ce que l’on doit stocker dans la carte ce code est suffisant.

## Interfaçage plug & play

Besoin d’ajouter un bouton de reset

**Référence de conception:** CCPT08

**Exigences client vérifiées : EXIG\_ROBOT\_PLUG&PLAY**

L’interfaçage plug and play de tous les composants sera géré par un PCB sur mesure que l’on va réaliser avec le schéma ci-dessous :

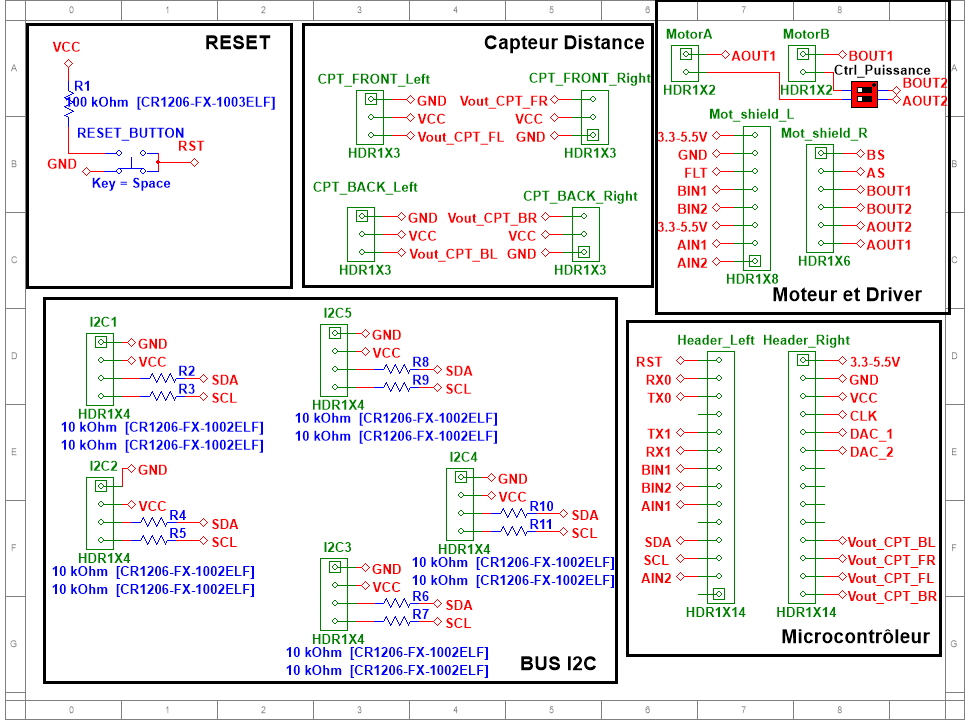


Figure 7: Schéma électronique

Ce dernier est uniquement composé de connecteurs classiques mâles (voir figure 6), qui sont reliés par ce PCB.

En plus de cela elle dispose d’un bouton de reset du µC car celui étant branché sous la carte PCB, le bouton de reset interne n’est pas accessible. Le schéma du circuit de reset est dans le bloc “RESET” ci-dessus

## Traitement numériques

**Référence de conception:** CCPT09

**Exigences client vérifiées : EXIG\_ROBOT\_PLUG&PLAY, EXIG\_ROBOT\_BUS, EXIG\_ROBOT\_µc, EXIG\_ROBOT\_PUISSANCE**

Pour le traitement numérique le microcontrôleur est imposé par le client : la Wipy 3.0 sera utilisée pour le projet électronique. Le tableau va valider toute les exigences de cette fonction:

|  |  |  |  |
| --- | --- | --- | --- |
| Wipy 3.0  wipysideTransTemp | **Besoin en ressource** | | **Disponible ?** |
| - 4 sorties numériques pour capteur de distance  - 4 sorties numérique pour les encodeurs  - 4 sorties numérique pour le driver de moteur | | Oui |
| **Total** | 12 |
| - 1 bus de communication I2C | | Oui |
| - 3 sorties spécifiques pour la carte micro SD | | Oui |
| - Communication Wifi | | Oui |

La Wipy 3.0 répond à tous les besoins, elle est donc **conforme** au CDC.

## Alimentation

**Référence de conception:** CCPT010

**Exigences client vérifiées : EXIG\_ROBOT\_ENERGIE, EXIG\_ROBOT\_RECHARGE, EXIG\_ROBOT\_INDICATEUR**

### Signalisation présence alimentation

L’ “Expansion Board 3v.0” permet de signaler la présence de l’alimentation grâce à la LED présente sur la carte comme on peut la voir sur le schéma ci-dessous:

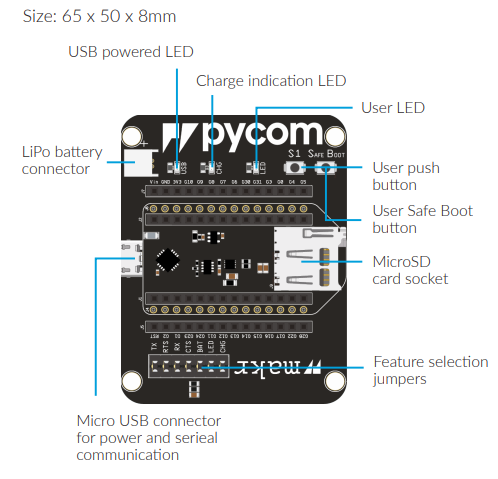
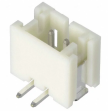


Figure 8: Caractéristique carte extension 3.0

### Gestion recharge batterie

L’ “Expansion Board 3v.0” permet de gérer la gestion de la batterie avec le chargeur de batterie Lipo (BQ24040), avec deux possibilités de courant de charge : 100mA ou 450mA. Cette gestion se fait grâce à l’ADC de la Wipy. De plus cette carte est protégée d’un inversement de polarité avec le TPS2115A.

**Remarque :** Le connecteur de la batterie est de type JST: 

## Conception mécanique châssis du robot

**Référence de conception:** CCPT012

**Exigences client vérifiées : EXIG\_ROBOT\_CHASSIS**

Plusieurs possibilités s’offrent à nous pour réaliser ce châssis :

1. Acheter un châssis standard directement sur site marchand
2. Réaliser soi-même le châssis avec les ressources à disposition à Polytech (via la découpe laser ou l’impression 3D)
   1. En pensant le robot entièrement nous même
   2. En se servant d’un châssis déjà existant au bon format et en l’adaptant à notre besoin.

Nous avons choisis de faire la 2.b), car la plus rentable, en temps, mais aussi cela permet de fixer le PCB sur mesure, les moteurs, les capteurs. De plus, on utilise les machines directement sur notre site.

Pour réaliser le châssis du robot, nous nous sommes inspiré d’une base sur Thingiverse, site internet qui partage des objets en 3D ou en découpe laser :

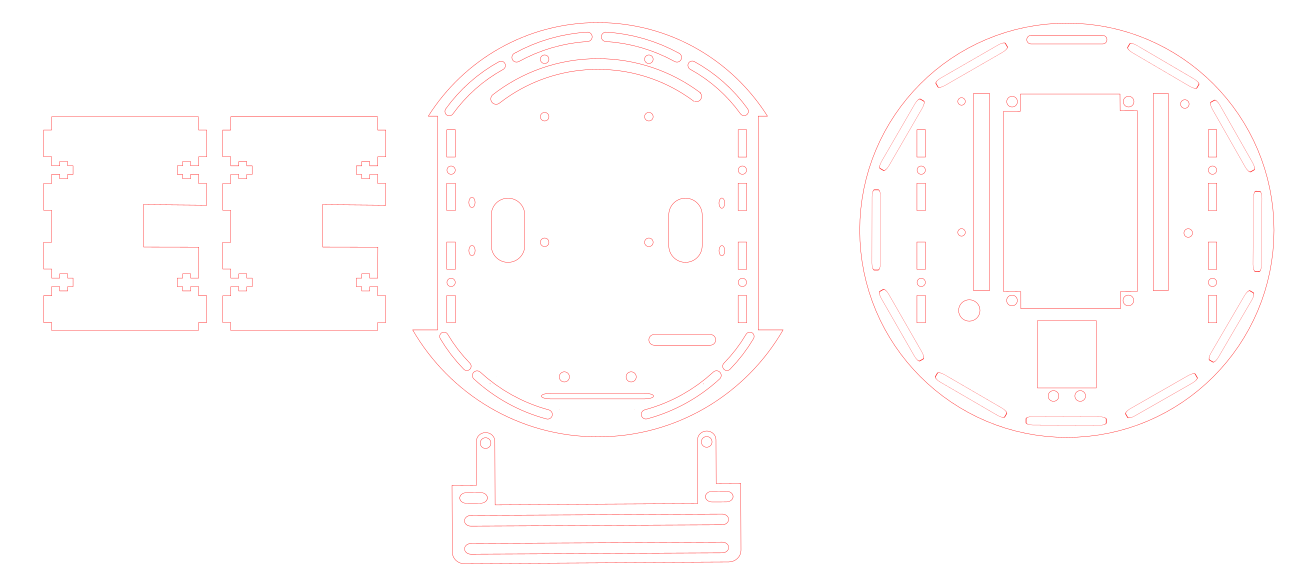


Figure 9 : Schéma châssis du robot

Le châssis a ensuite été adapté à nos composants, il y a donc eu diverse modification :

* Fixation et ouverture pour les moteur DC
* Fixation pour la roue avant
* Emplacement pour fixer la carte avec des vis
* Trou pour passer les nappes de câbles
* Divers trou de fixation ou d’attache pour plusieurs capteurs selon les besoins.

Pour pouvoir réaliser la découpe laser, nous avons rempli un formulaire disponible directement sur Celene puis nous avons pris contact avec un professeur. Enfin nous nous sommes rendus en DMS et il nous a imprimé les pièces en 4 min. Le service a été très efficace.

## Planning

**Référence de conception:** CCPT011

**Exigences client vérifiées : EXIG\_ROBOT\_DELAI**

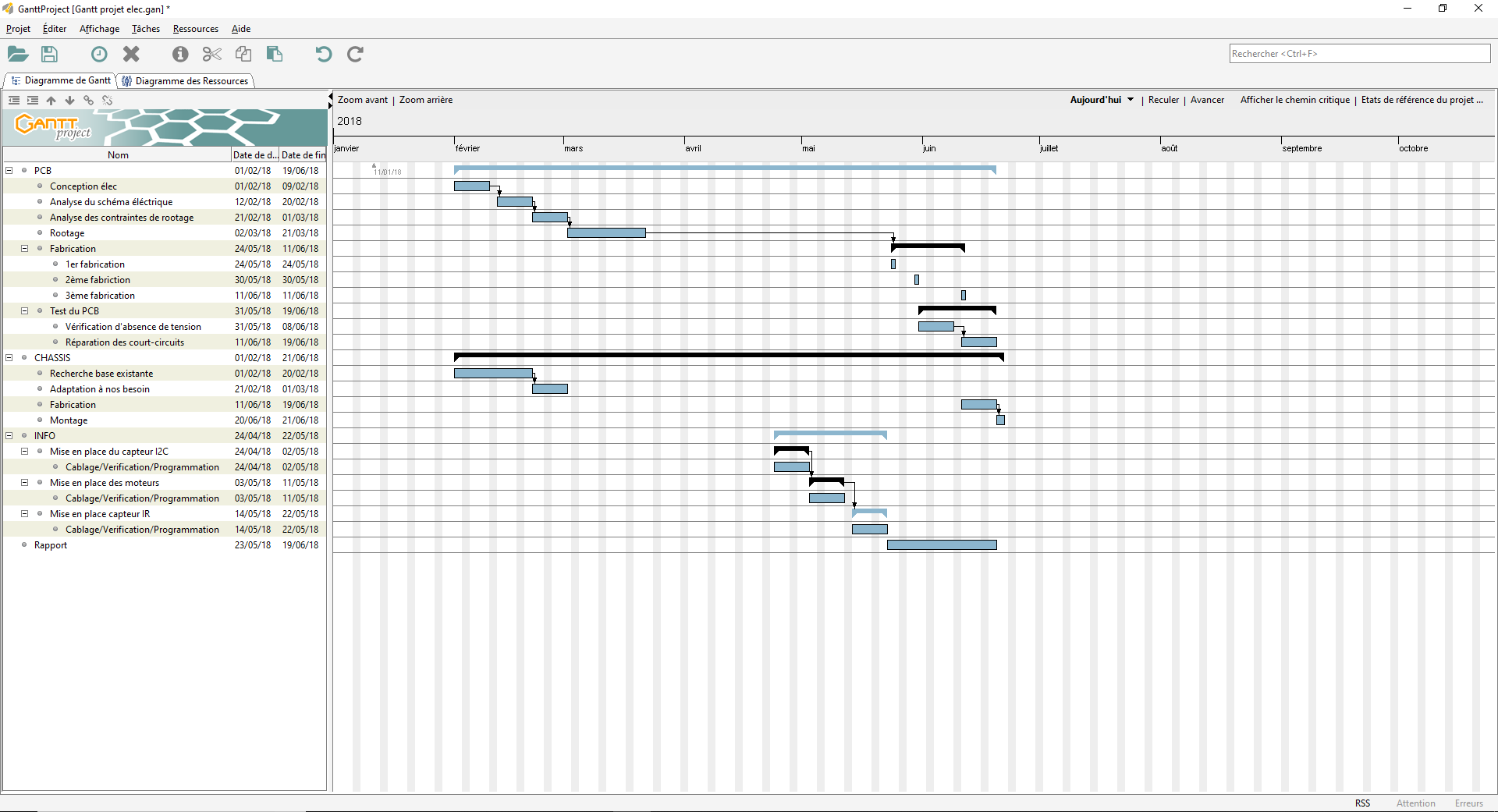


Figure 10: Planning du projet

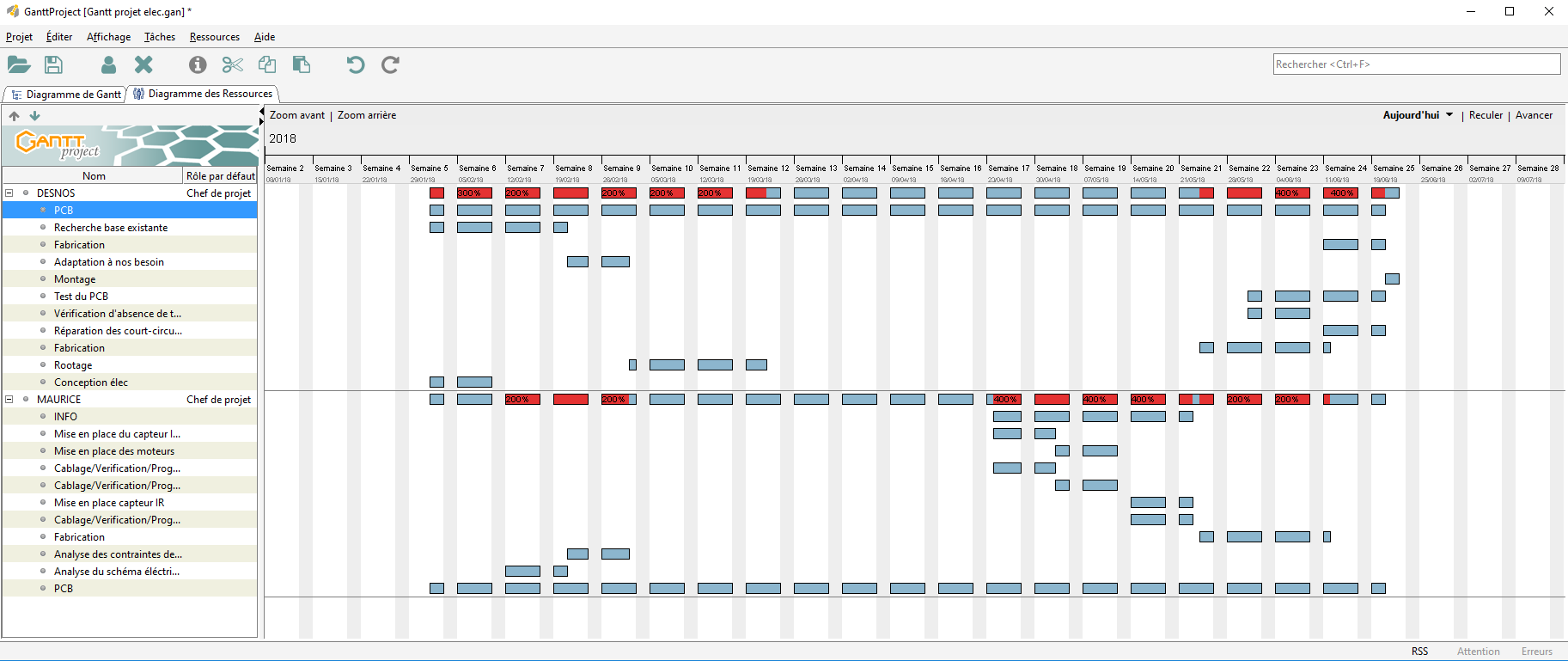


Figure 11 : Allocation des ressources humaines

## Coût

**Référence de conception:** CCPT013

**Exigences client vérifiées : EXIG\_ROBOT\_COUT**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **N° nom.** | **Références** | **Désignation produits, prestations** | **Montant unitaire H.T.** | **Qté** | **Montant H.T.** | **Montant T.V.A.** | **Montant T.T.C.** |
| 1 | <https://www.gotronic.fr/art-driver-moteur-pas-a-pas-ada3297-25543.htm> | Driver moteur pas-à-pas ADA3297 | 4,83 € | 1 | 4,83 € | 0,97 € | 5,80 € |
| 2 | <https://www.robotshop.com/ca/fr/jeu-roues-moteurs-miniq-encodeurs.html> | Jeu de Roues et Moteurs MiniQ avec Encodeurs | 30,49 € | 1 | 30,49 € | 6,10 € | 36,59 € |
| 3 | <https://www.robotshop.com/ca/fr/batterie-lipo-37v-2000mah-5c.html> | Batterie LiPo 3.7V, 2000mAh 5C | 7,41 € | 1 | 7,41 € | 1,48 € | 8,89 € |
| 4 | [https://www.gotronic.fr/art-capteur-de-luminosite-tsl2561-19569.htm (2.7 à 3.6V)](https://www.gotronic.fr/art-capteur-de-luminosite-tsl2561-19569.htm) | Capteur de luminosité TSL2561 | 5,67 € | 1 | 5,67 € | 1,13 € | 6,80 € |
| 5 | https://www.gotronic.fr/art-module-wipy-3-0-25373.htm | Wipy 3.0 | 23,95€ | 1 | 23,95€ | 4,79€ | 28,74€ |
| 6 | <https://www.gotronic.fr/art-carte-d-extension-expb-wp-24915.htm> | Extension Board pour Wipy 3.0 | 34,48 € | 1 | 34,48 € | 6,90 € | 41,38 € |
| 7 | <https://www.robotshop.com/ca/fr/interface-pololu-capteur-distance-gp2y0d815z0f-ir-sharp-05-15-cm.html> | Interface Pololu avec Capteur de Distance GP2Y0D815Z0F IR Sharp - 0.5 à 15 cm | 7,37 € | 4 | 29,48 € | 5,90 € | 35,38 € |
| 8 | <https://www.gotronic.fr/art-connecteur-ard008-18113.htm> | connecteur 8 pins traversant | 0,42 € | 2 | 0,84 € | 0,17 € | 1,01 € |
| 9 | <https://www.gotronic.fr/art-connecteur-ard006-18112.htm> | connecteur 6 pins traversant | 0,42 € | 2 | 0,84 € | 0,17 € | 1,01 € |
| 10 | <http://fr.farnell.com/bourns/cr1206-fx-1002elf/res-couche-epaisse-10k-1-0-25w/dp/2008395?st=CR1206-FX-1002ELF> | resistance CR1206 10kohm | 0,05 € | 10 | 0,50 € | 0,10 € | 0,60 € |
| 11 | Polytech | PCB | 2,5€ | 2 |  |  | 5€ |
| 12 | Polytech | Chassis robot | 15€ | 1 |  |  | 15€ |
| **138,49 €** | **27,70 €** | **186,19 €** |

# DDF

# Document de fabrication du produit

Nous avons pris soin d'archiver les fichiers de conception associés au projet. Les documents de fabrication du produit peuvent donc être exploités ou consultés en cas de besoin pendant ou après le développement du produit.

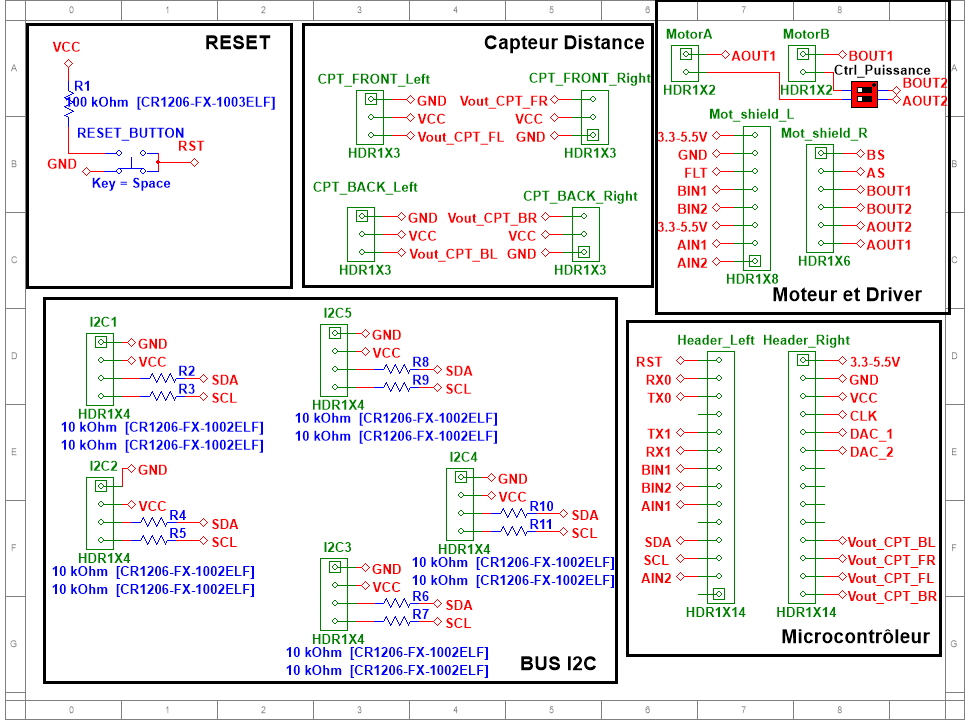
L'ensemble des fichiers est disponible dans le dossier : Robot\_Python

## Schéma électrique

**Référence de conception:** FAB01 (schéma électrique)

**Exigences client vérifiées : EXIG\_ROBOT\_BUS, EXIG\_ROBOT\_PLUG&PLAY**

**Fichier :** Robot\_Python/Electronique/PCB\_robot.ms14



## Nomenclature

**Référence de conception:** FAB02

**Exigences client vérifiées : EXIG\_ROBOT\_BUS, EXIG\_ROBOT\_µc**

**Fichier :** Robot\_Python/Electronique/Nomenclature

Ce fichier permet de connaître les composants à acheter et à placer sur le PCB.

## Fichier Gerber

**Référence de conception:** FAB03

**Exigences client vérifiées : EXIG\_ROBOT\_PLUG&PLAY**

**Fichier :** Robot\_Python/Electronique/Fichier Gerber

Dans ce dossier il y a tous les fichiers gerber non compressé de la dernière version compatible CNC.

De plus il y a deux fichiers .zip : “robotPython\_CNC.zip” pour utiliser la graveuse de Polytech CNC, et “robotPythonPro.zip”, pour **sous-traiter à un professionnel uniquement car les pistes sont plutôt fines et ne sont donc pas exploitable avec la CNC de Polytech**.

## Plan perçage

**Référence de conception:** FAB04

**Exigences client vérifiées : EXIG\_ROBOT\_BUS, EXIG\_ROBOT\_PLUG&PLAY**

**Fichier :** Robot\_Python/Electronique/Fichier Gerber/PCB\_robot\_ulti\_1 - Copper Top-Copper Bottom.drl

Ce fichier permet de percer les trous de la carte directement lors de l’usinage de la carte

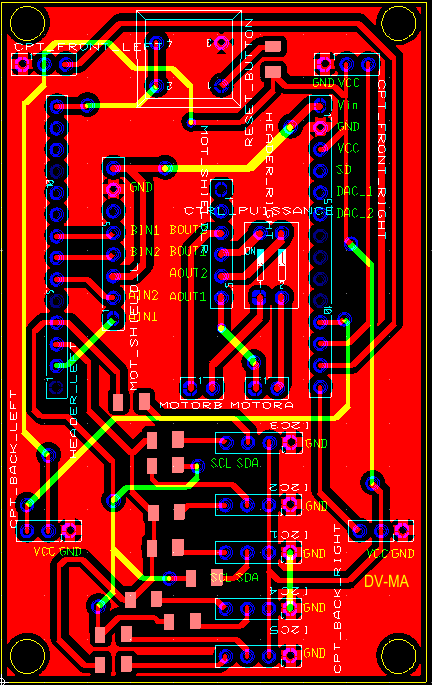
## Schéma implantation

**Référence de conception:** FAB05

**Exigences client vérifiées : EXIG\_ROBOT\_PUISSANCE, EXIG\_ROBOT\_PLUG&PLAY**

**Fichier :** Robot\_Python/Electronique/Images/PCB.svg

Ce fichier permet de placer les composants sur la carte après avoir usiné cette dernière.



## Châssis robot

**Référence de conception:** FAB06

**Exigences client vérifiées : EXIG\_ROBOT\_CHASSIS**

**Fichier :** Robot\_Python/Mécanique/chassis\_RobotPython/files/ ROBOT\_CHASSIS.svg

Pour réaliser le châssis du robot, nous nous sommes inspiré d’une base sur Thingiverse, site internet qui partage des objets en 3D ou en découpe laser.

# DDV

# Nature du document

Ce document est un dossier de vérification et a pour but de décrire les essais et les résultats de vérification. Il apporte les preuves de la conformité du produit développé vis-à-vis des exigences client. Le paragraphe 3 du [CDC] décrit de façon plus détaillée la nature et le positionnement de ce document dans l'arborescence documentaire du projet.

# Vérification du produit développé

Ce chapitre détaille la vérification par essais du produit développé. Il constitue une preuve de la conformité du produit. Chaque paragraphe d'essai fait donc clairement référence aux exigences client issues du Cahier des Charges.

## Mesure robot

**Référence de l'essai :** ESS01

**Exigences client vérifiées :** EXIG\_ROBOT\_DIMENSIONS

**But de l'essai : Vérifier la taille du PCB électronique**

**Moyens utilisés :**

* Règle

**Procédure d'essai:**

Mesure de la carte

**Résultats attendus :**

Des dimensions égales à 50mm (+/-20mm) enlargeur, 80mm (+/-20mm) en longueur.

**Résultats obtenus :**

55 mm en largeur et 80 mm en longueur

**Statut de l'essai :** Conforme

**Problèmes rencontrés :**

La carte a été découpée un peu en dessous du trait de découpe donnée par le schéma électrique car il a été fait à vue. Il sera donc difficile de percer les trous de fixation sur ce PCB, cependant cela ne pose pas de problème pour la production série.

## Autonomie batterie

**Référence de l'essai :** ESS02

**Exigences client vérifiées :** EXIG\_ROBOT\_ENERGIE

**But de l'essai : Vérifier l’autonomie du robot**

**Moyens utilisés :**

* Ampèremètre
* Alimentation stabilisée
* Driver moteurs
* Moteur DC

**Procédure d'essai:**

Branchement série de l’ampèremètre et de l’alimentation avec le driver branché aux moteurs

Chaque moteur consomme environ 600 mA à vides soit 1,2 A de consommation instantanée. On va arrondir à 2 A lorsque les moteurs doivent entrainer le robot.

Donc Autonomie =

**Résultats attendus :**

Une autonomie de 30-45 min

**Résultats obtenus :**

Une autonomie de 1h

**Statut de l'essai :** Conforme

**Problèmes rencontrés :** Aucun

## Test puissance moteur

**Référence de l'essai :** ESS03

**Exigences client vérifiées :** EXIG\_ROBOT\_PUISSANCE, EXIG\_ROBOT\_ROUE, EXIG\_ROBOT\_CHASSIS, EXIG\_ROBOT\_MOTEUR,

**But de l'essai : Vérifier la puissance des moteurs et le fonctionnement du driver**

**Moyens utilisés :**

* Moteurs DC
* Driver de moteurs
* Robot + charge

**Procédure d'essai:**

Programmer le robot pour tester la puissance des moteurs, le sens de rotation des moteurs, avec la PWM.

**Résultats attendus :**

Le robot comporte deux ponts en H pour pouvoir actionner les moteurs dans les deux sens de rotations.

**Résultats obtenus :**

Le driver contrôle les 2 moteurs dans les 2 sens de rotation avec une vitesse proportionnel à la PWM envoyée. La puissance max est compatible pour une utilisation pédagogique.

**Statut de l'essai :** Conforme

**Problèmes rencontrés :** Aucun

## Test bus de communication I2C

**Référence de l'essai :** ESS04

**Exigences client vérifiées :** EXIG\_ROBOT\_BUS, EXIG\_ROBOT\_PLUG&PLAY

**But de l'essai : Vérification du fonctionnement des bus I2C**

**Moyens utilisés :**

* Périphérique I2C, capteur de luminosité
* Moteurs
* Robot avec PCB
* Wipy 3.0

**Procédure d'essai:**

Programmer le robot pour communiquer avec le périphérique I2C de luminosité de manière à faire avancer le robot lorsqu’il est éclairé. Essayer le périphérique sur chaque connecteur I2C.

**Résultats attendus :**

Tous les bus I2C sont exploitables

**Résultats obtenus :**

Tous les bus I2C sont exploitables

**Statut de l'essai :** Conforme

**Problèmes rencontrés :**

Erreur sur les résistances, on a mis des résistances de 100 kOhms alors qu’il fallait mettre des résistances de 330 Ohms, du coup comme les résistances de 330 Ohms n’étaient pas disponibles à temps, nous avons choisi de les remplacer par des fils.

# Matrice de conformité du produit

Ce chapitre synthétise par l'intermédiaire d'un tableau la conformité du produit développé par rapport aux exigences issues du Cahier des Charges.

|  |  |  |  |
| --- | --- | --- | --- |
| **Référence de l'exigence client** | **Méthodes de vérification** | **Élément vérifiant l’exigence** | **Statut** |
| EXIG\_ROBOT\_DIMENSIONS | par inspection documentaire | PRC01 | Conf. |
| par mesure | ESS01 | Conf. |
| EXIG\_ROBOT\_INDICATEUR | par observation visuelle |  | Conf |
| Par analyse | PRC01, CCPT010 | Conf. |
| EXIG\_ROBOT\_ENERGIE | par analyse et calculs | PRC01, CCPT010, | Conf. |
| par essai | ESS02 | Conf. |
| EXIG\_ROBOT\_PUISSANCE | par essai | ESS03 | Conf. |
| par analyse | PRC01, CCPT09, CCPT05, FAB05, | Conf. |
| EXIG\_ROBOT\_COUT | par analyse | CCPT012 | **Non Conf.** |
| EXIG\_ROBOT\_DELAI | par analyse | CCPT011 | Conf. |
| EXIG\_ROBOT\_STOCKAGE | par analyse | PRC01, CCPT07 | Conf. |
| par essai | Non testé | **Non Conf**. |
| EXIG\_ROBOT\_MOTEUR | par observation visuelle | ESS03 | Conf. |
| par analyse | PRC01, CCPT06 | Conf. |
| EXIG\_ROBOT\_µC | par observation visuelle |  | Conf |
| par analyse | PRC01, CCPT09, CCPT02 | Conf. |
| EXIG\_ROBOT\_ENCODEUR | par essai | Non testé | **Non Conf.** |
| par analyse | PRC01, CCPT03 | Conf. |
| EXIG\_ROBOT\_BUS | par essai | ESS04 | Conf. |
| par analyse | PRC01, CCPT09, CCPT04, CCPT02, CCPT01, FAB04 | Conf. |
| EXIG\_ROBOT\_RECHARGE | par observation visuelle | ESS02 | Conf. |
| par analyse | PRC01, CCPT010 | Conf. |
| EXIG\_ROBOT\_PLUG&PLAY | par observation visuelle | ESS04 | Conf. |
| par analyse | PRC01, CCPT09, CCPT08, CCPT05, CCPT04, CCPT03, CCPT01, FAB01, FAB02, FAB03, FAB04, FAB05 | Conf. |
| EXIG\_ROBOT\_ROUE | par essai | ESS03 | Conf. |
| par analyse | PRC01, CCPT06 | Conf. |
| EXIG\_ROBOT\_CHASSIS | par observation visuelle | ESS03 | Conf. |
| Par analyse | CCPT012, FAB06 | Conf. |

# Comparatif et description des outils de développement

## Electronique

Pour la partie électronique du projet nous avons choisi d’utiliser la suite **MultiSim** et **UltiBoard**, car des licences ont été payé par Polytech et il est souhaité que nous utilisions ce logiciel dans nos réalisations de schéma ou carte électronique. Une alternative gratuite possible est Kicad.

## Mécanique

Pour réaliser le châssis du robot, le logiciel **Inkscape** nous a été fortement recommandé pour réaliser le fichier .svg de découpe laser. Il est à la fois gratuit et plutôt facile à prendre en main.