République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Abderrahmane Mira - Bejaia
Faculté de Technologie

جامعة بجاية
Tasdawit n Bgayet
Université de Béjaïa

Département d'Automatique, de Télécommunication et d'Electronique

Mémoire de fin de cycle

En vue de l'obtention du diplôme de Master en Electronique
Option : Automatique

THEME :

Etude et Réalisation d'une Carte de Commande Pour un Robot à Quatre ddl

Réalisé par :
Mr : ALLOU Hichem

Encadré par :
Mr : MENDIL Boubekeur

Promotion : 2015/2016

Remerciements

Je remercie Mr Boubekeur MENDIL, Professeur à l'université de Bejaia, pour avoir accepté d'encadrer ce projet de master.

Je tiens également à remercier les membres du Jury qui ont bien voulu accepter d'examiner ce travail.

Mes remerciements vont aussi aux membres du club scientifique génie électrique et énergies renouvelables pour le soutien moral qu'ils ont apporté.

A tous ceux qui nous ont aidé de près ou de loin à l'accomplissement de ce travail.

Dédicaces

Je dédie ce modeste travail

Aux êtres les plus chères à mon coeur,

Mes chers parents

D'avoir oeuvré pour ma réussite, votre encouragement et votre soutien, tous les sacrifices consentis et vos précieux conseils.

A mes frères SAMIL et AMIR

A tous mes amis et tous ceux qui sont chers pour moi

Hichem

Sommaire

Introduction générale 1

Chapitre I **Étude du bras de robot** .....

I.1. Introduction 2

I.2. Structure mécanique 3

I.2.1. Dimensions des articulations 3

I.2.2. Réducteurs de vitesse 4

I.2.3. Espace de travail du manipulateur 5

I.3. Identification 5

I.3.1. Les actionneurs 5

I.3.2. Les capteurs 7

I.4. Conclusion 8

Chapitre II **Étude et Réalisation** .....

II.1. Introduction 9

II.2. Microcontrôleurs utilisés 9

II.2.1. Le microcontrôleur PIC18f2550 9

II.2.1.1. Caractéristiques générales 10

II.2.1.2. Schéma fonctionnel de PIC18f2550 11

II.2.2. Le microcontrôleur PIC16f887 12

II.2.2.1. Caractéristiques générales 12

II.2.2.2. Schéma fonctionnel de PIC16f887 13

II.3. Fréquence de fonctionnement des microcontrôleurs 14

II.4. Entrées/Sorties analogiques 15

II.4.1. Entrées analogiques (A/D) 15

II.4.2. Sorties analogiques (PWM) 16

II.5. Protocoles de communication 17

II.5.1. Communication USB 17

II.5.2. Bus I²C 17

II.6. Etage d'alimentation 19

II.6.1. Transformateur 19

II.6.3. Régulateurs de tension 20

II.6.4. Réalisation d'Alimentation 21

II.7. Partie de puissance 22
II.7.1 Pont H 22
II.8. Acquisition de données 24
II.9. Coté commande 24
II.10. Schéma électrique 25
II.11. Réalisation de typon 27
II.12. Réalisation de la carte électronique 27
II.12.1. Insolation 28
II.12.2. Révélation 29
II.12.3. Graveur 29
II.12.4 Circuit finalisé 30
II.13 Conclusion 32

Chapitre III **Outils informatique et programmation** .....
III.1. Introduction 33
III.2. Outils informatique utilisés 33
III.2.1. logiciel de conception 33
III.2.1.1. Proteus 33
III.2.1.2. CINEMA 4D 35
III.2.2. Logiciels de programmation 36
III.2.2.1. Microsoft visual studio 36
III.2.2.2. Mikroc 37
III.2.2.3. USB-PicProg 37
III.3 Programmations 38
III.3.1. Interface utilisateur 38
III.3.1.1. USB 39
III.3.1.2. Commande 40
III.3.1.2. Simulateur 3D 41
III.3.2. Programmation des microcontrôleurs 41
III.3.2.1. Microcontrôleur Maitre 42
III.3.2.1.1 Configuration 44
III.3.2.1.2 Acquisitions et envoi de données 46
III.3.2.1.3 Génération des commandes 47
III.3.2.1.4 Réinitialisation des données et envoi aux esclaves 49

III.3.2.2. Programmation des Pics en mode esclave.....50
III.4 Conclusion.....51
Conclusion générale et perspective 52

Acronymes

d.l.l	Degré de liberté.
EEPROM	Electrically Erasable Programmable Read-Only Memory.
PWM	Pulse Width Modulation.
MSSP	Master Synchronous Serial Port.
USART	Universal Synchronous Asynchronous Receiver Transmitter.
PIC	Programmable Interface Controller.
FPGA	Field Programmable Gate Array
CPU	Central Processing Unit
USB	Universal Serial Bus.
SPI	Serial Protocol Interface.
I^2C	Inter-Integrated Circuit
HID	Human Interface Device
VID	Vendor ID
PID	Product PD

Liste des figures

Chapitre I

Figure I.1. Image représentent du projet

Figure I.2. Schématisation du bras de robot : vue de face

Figure I.3. Schématisation du bras de robot : vue de haut

Figure I.4. Image représentante d'un réducteur de vitesse

Figure I.5. Les sections des moteurs

Figure I.6. Les capteurs potentiométriques encodeur optique

Chapitre II

Figure II.1. Schéma fonctionnel du PIC18f2550

Figure II.2. Schéma fonctionnel de PIC16f887

Figure II.3. Représentation du signale d'horloge

Figure II.4. Image représentante du quartz

Figure II.5. Branchement de l'oscillateur avec un PIC

Figure II.6. Représentation du convertisseur analogique/numérique

Figure II.7. Représentation du rapport cyclique

Figure II.8. Commande d'un moteur en utilisant un signal PWM

Figure II.9. Deferent type de connecter USB

Figure II.10. Branchement entre maitre et esclaves

Figure II.11. Transformateur

Figure II.12. Représentation d'un redresseur

Figure II.13. Régulateurs de tension

Figure II.14. Schéma de l'alimentation 12v et 5v

Figure II.15. Image représentative de l'alimentation 12v et 5v

Figure II.16. Branchement d'un moteur avec un PIC

Figure II.17. Représentation du Pont H

Figure II.18. Pont en H transistor MOS

Figure II.19. Pont en H : L293D

Figure II.20. Acquisition de données et filtrage

Figure II.21. Les microcontrôleurs maitre et esclaves

Figure II.22. Schéma électrique sur isis

Figure II.23. Schéma électrique

Figure II.24. Le logiciel *ARES*

Figure II.25. L'Insoleuse

Figure II.26. Image du révélateur solide

Figure II.27. Le circuit final

Figure II.28. La carte finale

Chapitre III

Figure III.1. Vue du logiciel *ISIS*

Figure III.2. Vue du logiciel *ARES*

Figure III.3. Vue en 3D

Figure III.4. Vue du logiciel *CINEMA 4D*

Figure III.5. Vue du logiciel *Microsoft Visual*

Figure IV.6. Vue du logiciel *Microc*

Figure III.7. Vue du logiciel *Usb-PicProg et le Matériel*

Figure III.8. Organigramme de l'interface utilisateur

Figure III.9. L'éditeur de commandes

Figure III.10. Simulateur 3D

Figure III.11. Aperçu final de l'interface utilisateur

Figure III.12. Organigramme maitre

Figure III.13. USB HID terminal

Figure III.14. Fenêtre de configuration

Figure III.15. Organigramme PID

Introduction générale

La robotique est devenue une discipline extrêmement populaire. Alliant un grand intérêt pédagogique et industriel, elle demande beaucoup de créativité et de connaissances pluridisciplinaires. Un robot est constitué en général d'un système mécanique articulé et de la partie électronique permettant sa commande. Il peut aussi intégrer une variété de capteurs, selon les tâches à réaliser. La programmation du système de commande exige des connaissances en informatique, automatique, en traitement temps-réel et même en des domaines avancés tels que la vision artificielle, l'intelligence artificielle, etc.

L'objectif de notre travail est la conception et la réalisation de cartes électroniques pour la commande d'un bras manipulateur à 4 degrés de liberté et une interface utilisateur pour l'envoi des commandes.

Pour effectuer les tests on a récupéré un bras robotique à 4 degrés de liberté qui se trouve au niveau du laboratoire de robotique de l'université de Bejaia. La carte de commande réalisée est à base de microcontrôleurs PIC programmé avec le *langage C* à l'aide de l'environnement de développement *MikroC*. L'interface utilisateur qui permet l'interaction avec le robot et programmée avec le *langage C#* sous l'environnement de développement *visual studio 2012*.

Le mémoire est organisé en trois chapitres. Le premier décrit le bras de robot étudié. Le deuxième chapitre est consacré à la présentation des différents modules des microcontrôleurs utilisés. La dernière partie présente les détails de réalisation et le fonctionnement de l'étage d'alimentation et les autres cartes électroniques. Le chapitre trois englobe les logiciels utilisés dans notre projet et la partie programmation des microcontrôleurs et la conception de l'interface utilisateur.

Chapitre I

Étude du bras de robot

I.1. Introduction

De nos jours, la robotique est devenue une profession assez connue. Elle est présente un peu partout : l'automobile, l'aéronautique, les appareilles domestiques et aussi la médecine. Qui dit robotique dit mécanique, électronique, informatique et automatique. L'union de toutes ces disciplines fait de la robotique un domaine prometteur et passionnant.

Notre travail consiste à concevoir et à réaliser la partie électronique pour un bras robotique récupéré ayant 4 articulations. Le problème avec ce robot est qu'aucune documentation le concernant n'est trouvée. Il faudra donc faire le dimensionnement de sa structure mécanique, identifier le type d'actionneurs utilisés, ainsi que ces capteurs.

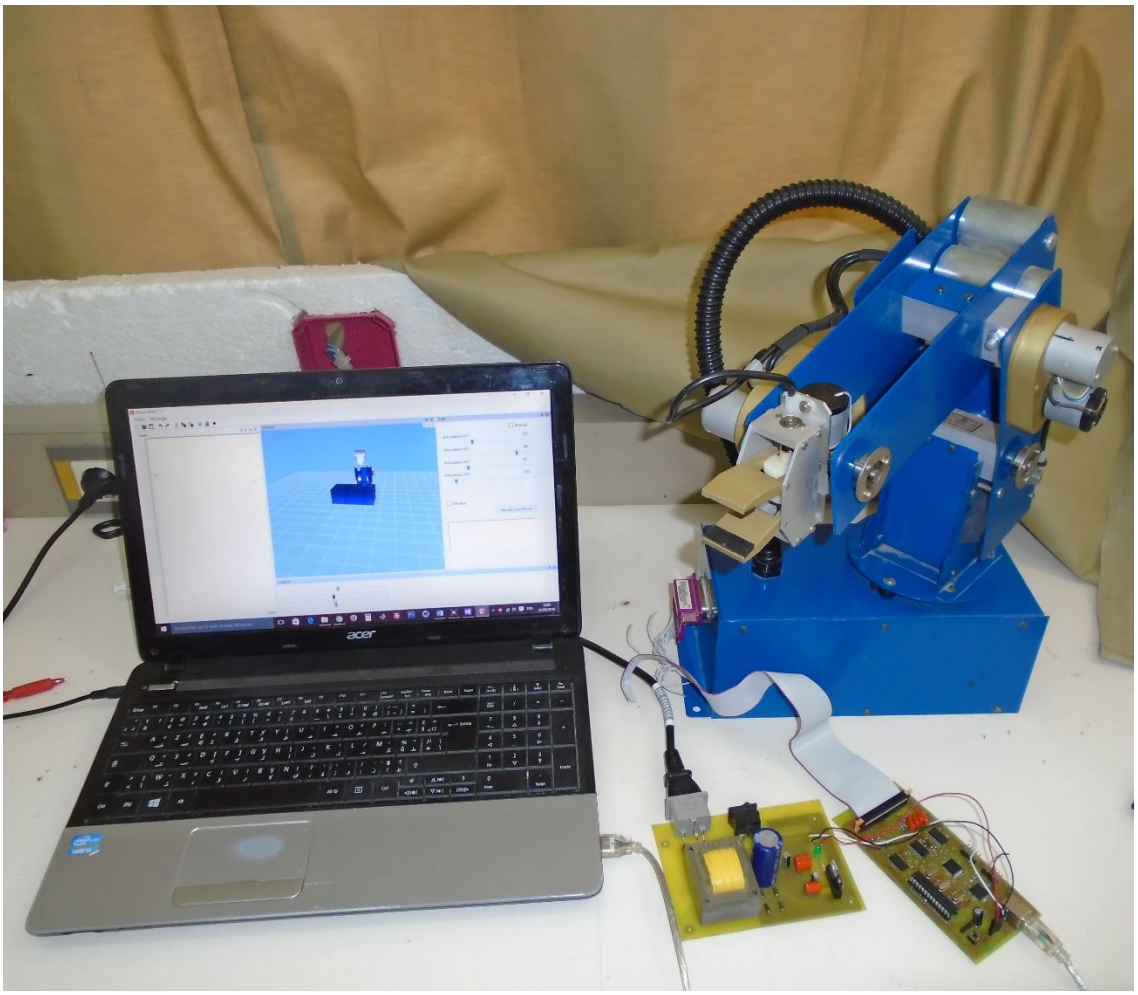


Figure I.1. Image représentent du projet

I.2. Structure mécanique

On parle des pièces qui constituent le bras. D'une manière générale, la structure du robot est fabriquée avec des surfaces métalliques. On retrouve aussi du plastique avec lequel sont fabriqués quelques engrenages.

I.2.1. Dimensions des articulations

Afin de pouvoir commander un bras robotique, il est essentiel de bien connaitre les dimensions de chaque articulation. Comme on ne dispose d'aucune information concernant le robot, on a dû prendre des mesures de chaque articulation à l'aide d'un pied à coulisse et établir les schémas ci-dessous.

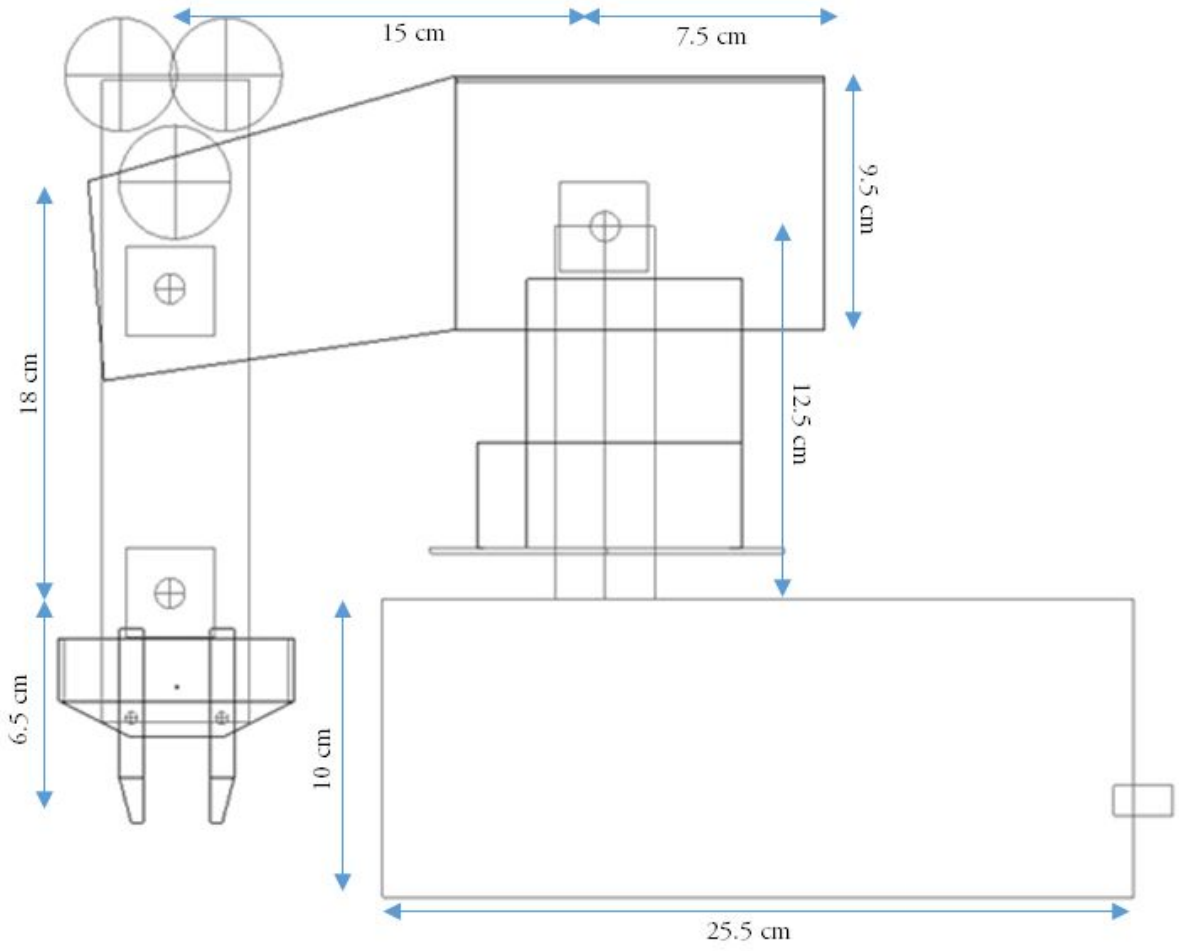


Figure I.2. Schématisation du bras de robot : vue de face

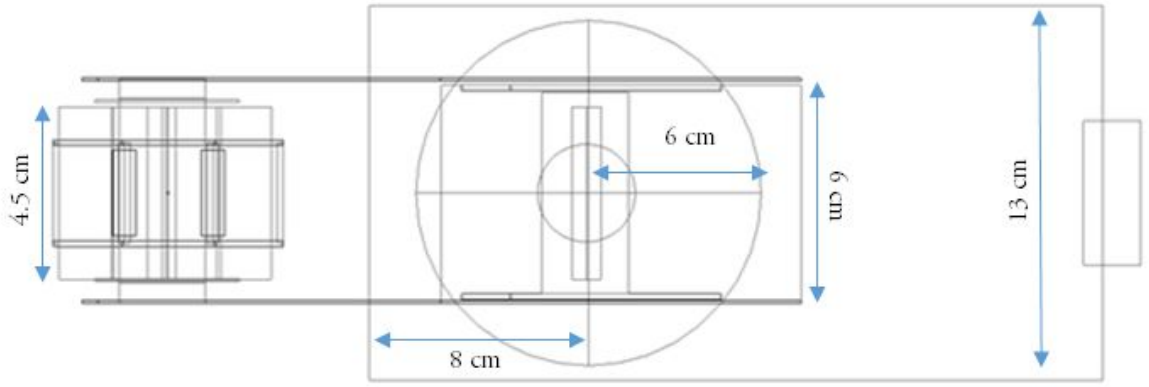


Figure I.3. Schématisation du bras de robot : vue de haut

I.2.2. Réducteurs de vitesse

Comme son nom l'indique, un réducteur de vitesse sert à réduire la vitesse du moteur, afin de pouvoir augmenter son couple mécanique. Son principe de fonctionnement ce résume au rapport entre le nombre de dent des engrenages qui le constituent. A chaque fois que le nombre de dents de l'engrenage associé à l'arbre du moteur est moins élevé que celui du deuxième engrenage, la vitesse diminue.

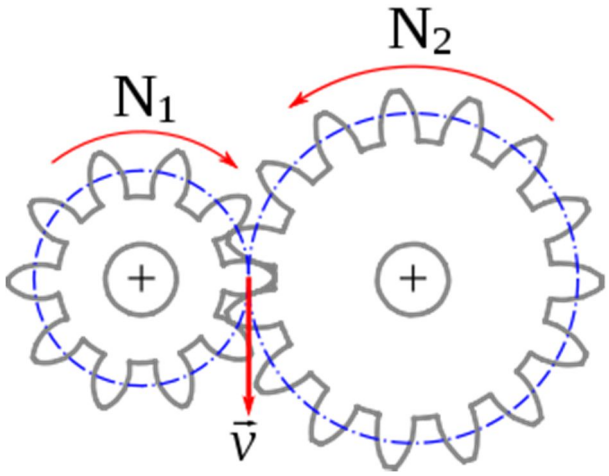


Figure I.4. Image représentante d'un réducteur de vitesse

I.2.3. Espace de travail du manipulateur

L'espace de travail représente le volume atteignable par l'effecteur. Il dépend de la structure mécanique elle-même, le tableau ci-dessous représente la totalité des angles que chaque articulation peut atteindre.

Table I.1. Plages de variations des articulations

Articulations	Angles max	Angle min	Type
Articulation 1	260	25	Rotatif
Articulation 2	160	90	Rotatif
Articulation 3	225	25	Rotatif
Articulation 4	80	250	Rotatif
Effecteur	/	/	

I.3. Identification

I.3.1. Les actionneurs

Les actionneurs représentent la partie la plus importante de la structure mécanique d'un robot. Le choix des actionneurs dépend de la structure elle-même et aussi l'environnement où le robot est censé exécuter des taches.

Les actionneurs utilisés dans notre robot sont des moteurs à courant continu sous la référence **F 2130** de chez *maxonmotor*. On a pu retrouver les informations essentialles pour le bon fonctionnement de ces moteurs, à savoir les valeurs nominales de la tension, du courant et de la vitesse ainsi que quelques informations supplémentaires **[1]**.

Table I.2. Caractéristiques des moteurs

Tension nominale (v)	12
Vitesse nominale (rmp)	3050
Courant nominale (A)	0.204
Courant de démarrage (A)	0.376
Résistance interne (Ω)	31.9
Inertie du rotor (gcm^2)	4.02

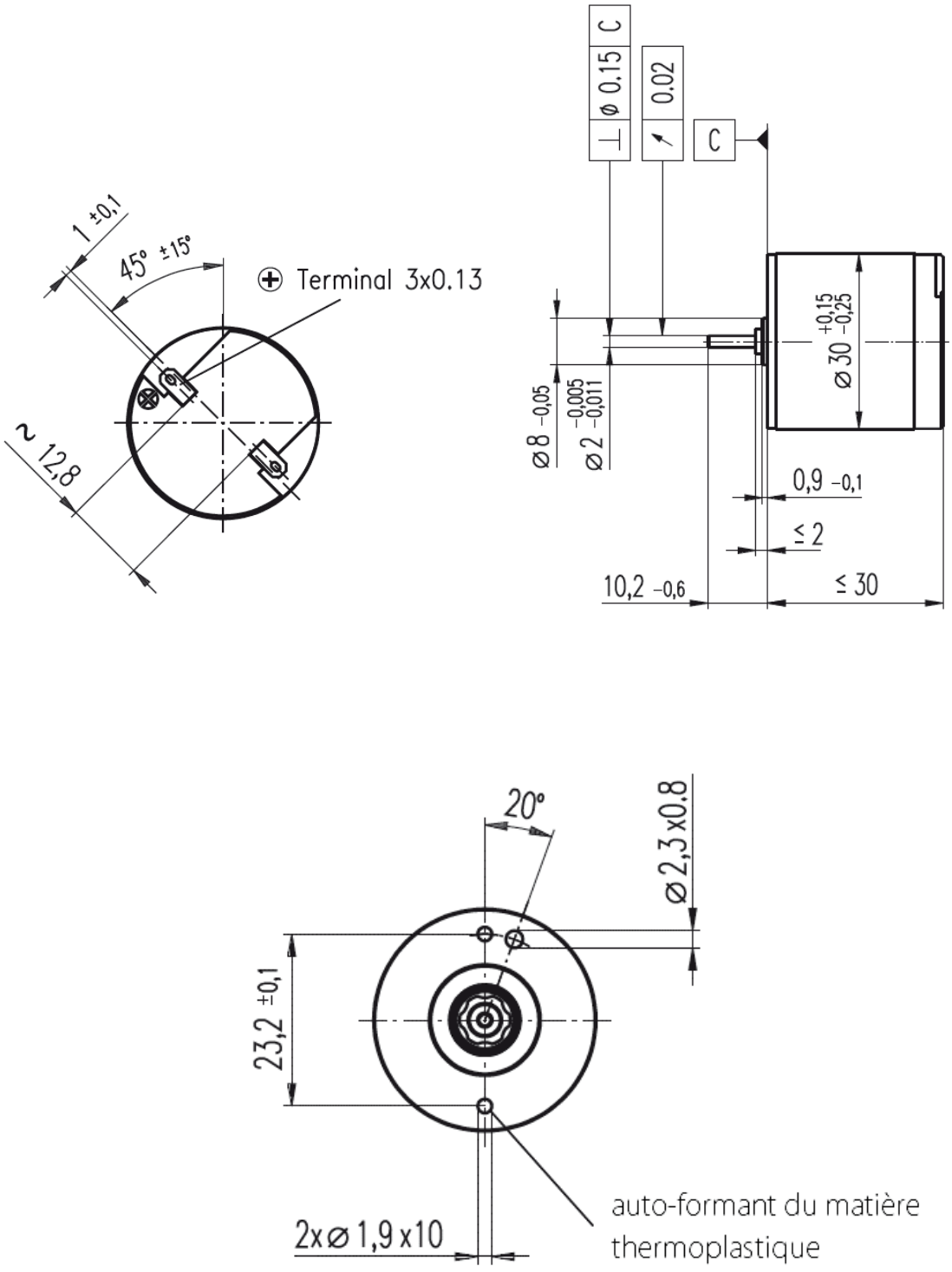


Figure I.5. Les sections des moteurs

I.3.2. Les capteurs

Dans la robotique, plusieurs types de capteurs peuvent être utilisés pour avoir l'état du robot de façon explicite, à savoir : la position, la vitesse, l'accélération et la détection des obstacles.

Les capteurs les plus utilisés dans les bras robotique sont des capteurs potentiométriques ou des encodeurs optiques, vue leur simplicité et leur coût.

Figure I.6. Les capteurs potentiométriques encodeur optique

Notre robot comporte 4 capteurs potentiométriques de 5kΩ reliés aux moteurs grâce à un système d'engrenages rédacteur. Leur axe rotatif et relié directement à l'articulation. Ce qui permet de savoir la valeur de l'angle sans avoir recours à des calculs supplémentaires, à part celui du rapport degré/tension.

I.4. Conclusion

Dans ce chapitre on a exposé quelques informations utiles sur la structure et le fonctionnement du bras de robot utilisé. Néanmoins, vue son ancienneté, on n'a pu trouver que quelques bribes d'information. Le reste des chapitres concerne la conception et la réalisation de la partie commande constituée des cartes électroniques et de l'algorithmique associée.

Chapitre II

Étude et Réalisation

II.1. Introduction

L'électronique a révolutionné le monde notamment avec la sortie des composantes programmables, à savoir les microprocesseurs, les FPGA. Mais, le composant qui nous intéresse le plus c'est le microcontrôleur. Car, il est le cœur de notre projet.

La carte de commande est réalisée sur une seule plaque double face et comporte 3 microcontrôleurs : un qui fonction en maître et les deux autres en esclaves. Le maître est le responsable de la communication entre la carte et l'interface de contrôle. On exposera dans ce chapitre les différents composants utilisés, les protocoles de communications et aussi le schéma électronique.

II.2. Microcontrôleurs utilisés

Les microcontrôleurs ont contribué à la miniaturisation des systèmes électroniques grâce à leur petite taille qui ne dépasse pas les quelques centimètres, mais aussi au périphériques qu'ils intègrent. On peut trouver des ports d'entrée/sortie et des périphériques de communications (USB, UART, I²C, SPI …etc.).

Il existe plusieurs types de microcontrôleurs. On s'est intéressé aux microcontrôleurs de la famille Microchip, plus exactement au PIC18f2550 et au PIC16f887. Le choix de ces deux n'est pas au hasard, mais par ce qu'ils conviennent le mieux à notre projet, le PIC18f2550 pour son périphérique USB et le PIC16f887 pour sa facilité de configuration.

On a utilisé 3 microcontrôleurs en relation maître-esclave (un maître et deux esclaves) pour générer la sortie PWM. Chaque microcontrôleur ne possède que deux sorties de ce type, alors que on a besoin de six.

II.2.1. Le microcontrôleur PIC18f2550

Le choix de ce microcontrôleur est déjà précisé. Mais, il n'empêche qu'il a d'autres avantages par rapport aux autres microcontrôleurs. Il dispose d'une vitesse de traitement de 16 bits, alors que la vitesse de traitement du PIC16f887 est de 14 bits. Il possède aussi une plage de fréquence très importante (0 Mhz -48 Mhz).

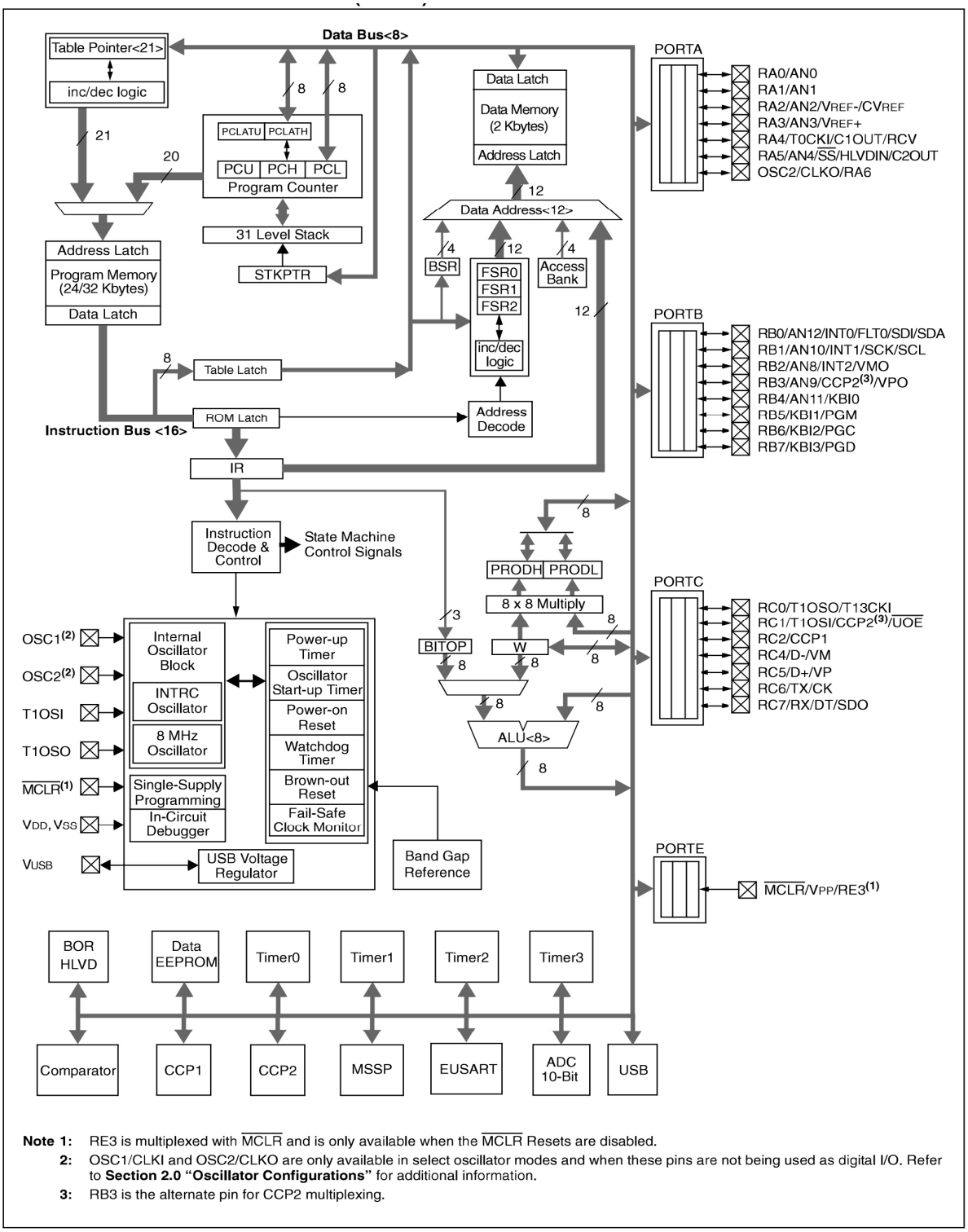
II.2.1.1. Caractéristiques générales

Les caractéristiques et les périphériques principaux englobés dans ce microcontrôleur sont :

- Tension d'alimentation : 2V - 5.5V
- Fréquence de fonctionnement : 48 Mhz.
- Mémoire de programme : 32768Octets.
- Mémoire de données : 2048Octets.
- Mémoire EEPROM : 255 Octets.
- Ports E/S: Ports A, B, C.
- Timers : 4.
- Capture/Compare/ Modules PWM : 2
- Communications série : Oui.
- USB : Oui.
- Convertisseur analogique-numérique : 10 Entries à 10 bits.
- Comparateurs : 2.
- Programmable basse tension : Oui.
- Programmable Brown sans Réinitialiser : Oui.
- Jeu d'instructions : 75 Instructions de service, 83 avec jeu d'instructions étendu activée.
- Boîtier : 28Pin PDIP, 28Pin SOIC.

II.2.1.2 Schéma fonctionnel de PIC18f2550

La CPU du PIC18f2550 est entouré d'un nombre important de modules parmi eux, trois PORT entrée/sortie, quatre Timer, un module EUSART pour la communication série asynchrone et le module MSSP pour la communication série synchrone, un convertisseur analogique/numérique, deux module PWM, et le plus important le périphérique USB



Note 1: RE3 is multiplexed with MCLR and is only available when the MCLR Resets are disabled.
2: OSC1/CLKI and OSC2/CLKO are only available in select oscillator modes and when these pins are not being used as digital I/O. Refer to **Section 2.0 "Oscillator Configurations"** for additional information.
3: RB3 is the alternate pin for CCP2 multiplexing.

Figure II.1. Schéma fonctionnel du PIC18f2550

II.2.2. Le microcontrôleur PIC16f887

La différence de ce microcontrôleur est importante par rapport au PIC18f2550. On remarque qu'il a moins de mémoire de programme, une fréquence de fonctionnement de 20Mhz, alors que le PIC18f2550 fonctionne avec une fréquence de 48Mhz.

II.2.2.1. Caractéristiques générales

- Tension d'alimentation : 2V - 5.5V
- Fréquence de fonctionnement : 20 Mhz.
- Mémoire de programme : 8192 Mots.
- Mémoire de données : 2048Octets.
- Mémoire EEPROM : 255 Octets.
- Ports E/S: Ports A, B, C, D, E.
- Timers : 3.
- Modules PWM : 2
- Communications série : Oui.
- USB : Non.
- Convertisseur analogique-numérique : 14 Entries à 10 bits.
- Comparateurs : 2.
- Programmable basse tension : Oui.
- Programmable Brown sans Réinitialiser : Oui.
- Jeu d'instructions : 35 Instructions.
- Boîtier : 40Pin PDIP, 44Pin QFN, 44Pin TQFP.

II.2.2.2 Schéma fonctionnel de PIC16f887

De même la CPU du PIC16f887 entouré d'un nombre important de module. On voit clairement que le nombre de PORT entrée/sortie et plus importante par rapport au PIC18f2550 qui peut être un avantage dans certaine application, on remarque l'absence de périphérique USB dans ce PIC.

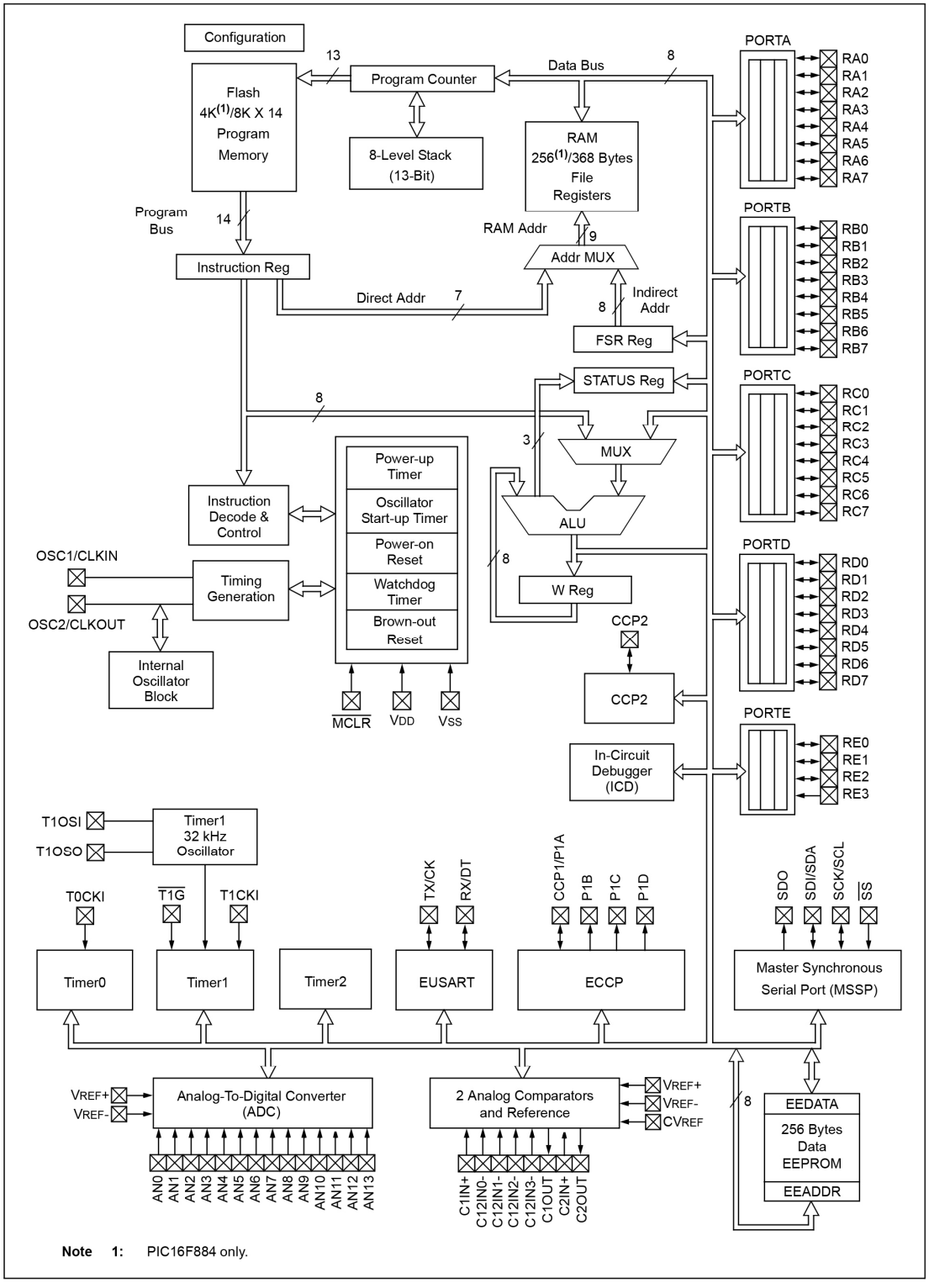


Figure II.2. Schéma fonctionnel de PIC16f887

II.3. Fréquence de fonctionnement des microcontrôleurs

Les microcontrôleurs ont besoins d'un signal de synchronisation permettant le bon déroulement des programmes stockés dans la mémoire de programme et aussi de quelques périphériques, à savoir : les timers, les périphériques de communications, etc.

Ce signal appelé aussi signal d'horloge est soit généré directement à l'intérieur du microcontrôleur (l'oscillateur est sous forme de périphérique) ou bien à l'extérieur.

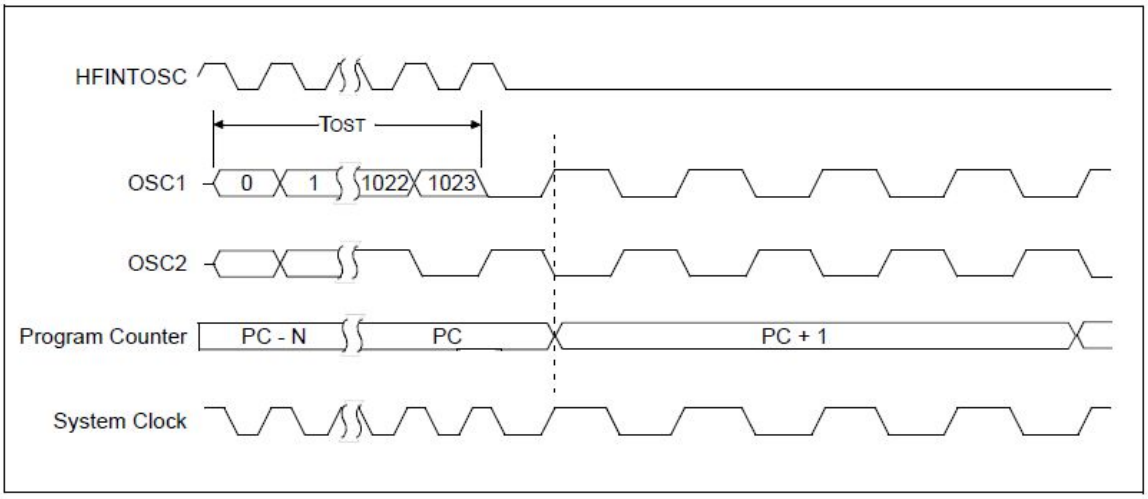


Figure II.3. Représentation du signale d'horloge

Les oscillateurs utilisés dans notre projet sont à base de quartz. Le choix de ce type d'oscillateurs est dû à sa grande fréquence de 20Mhz pour les trois microcontrôleurs.

Figure II.4. Image représentante du quartz

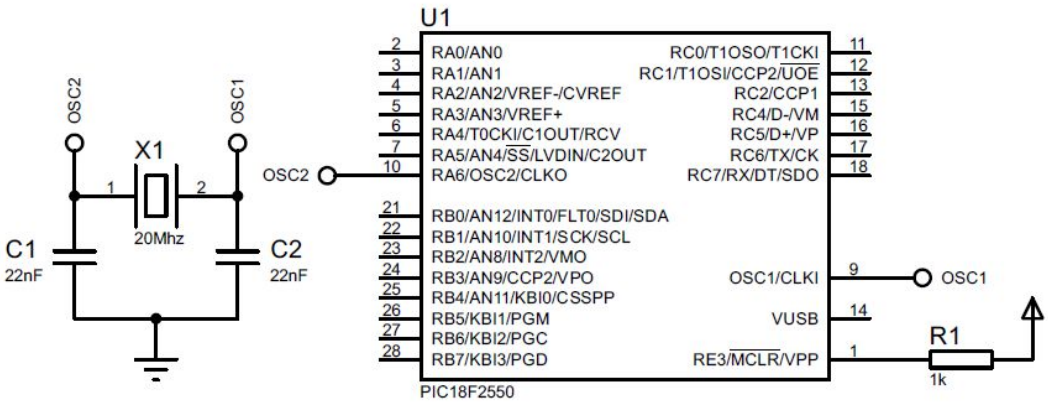


Figure II.5. Branchement de l'oscillateur avec un PIC

II.4. Entrées/Sorties analogiques

Afin de pouvoir commander le bras robotique, il est essentiel de récupérer les états des articulations qui sont fournies par des capteurs sous forme de signaux analogiques et aussi de générer les tensions d'alimentation variables pour les moteurs.

II.4.1. Entrées analogiques (A/D)

Il s'agit en fait des convertisseurs analogiques numériques. Nous avons besoin de six entrées pour six capteurs. Grâce aux trois microcontrôleurs, on a un total de 38 entrées, ce qui largement suffisant.

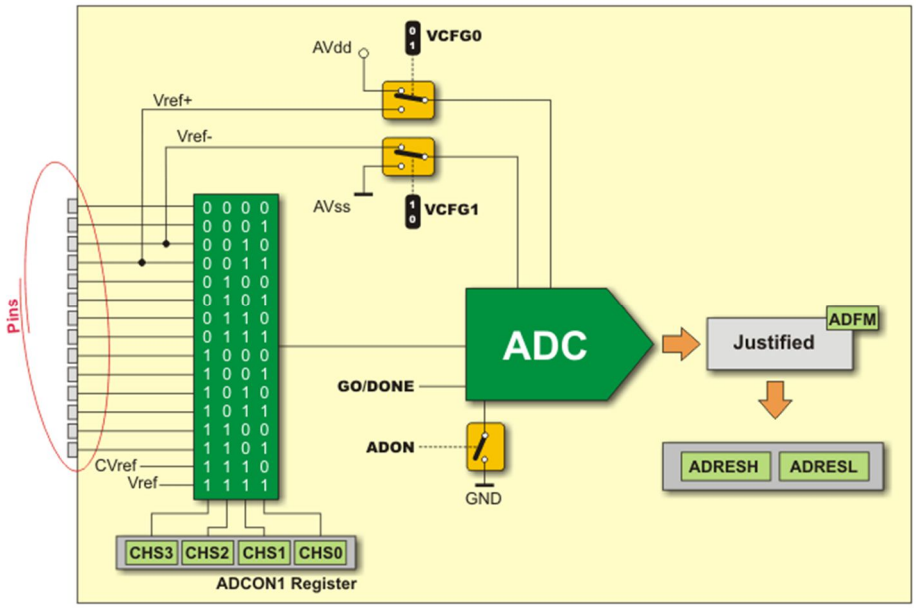


Figure II.6. Représentation du convertisseur analogique/numérique

II.4.2. Sorties analogiques (PWM)

Le périphérique PWM (Pulse Width Modulation) ou MLI (Modulation de Largeur d'Impulsions), comme son nom l'indique, est capable de générer des impulsions de fréquence et de rapport cyclique donnés, la variation du rapport cyclique fait en sorte que la valeur moyenne soit variable elle aussi.

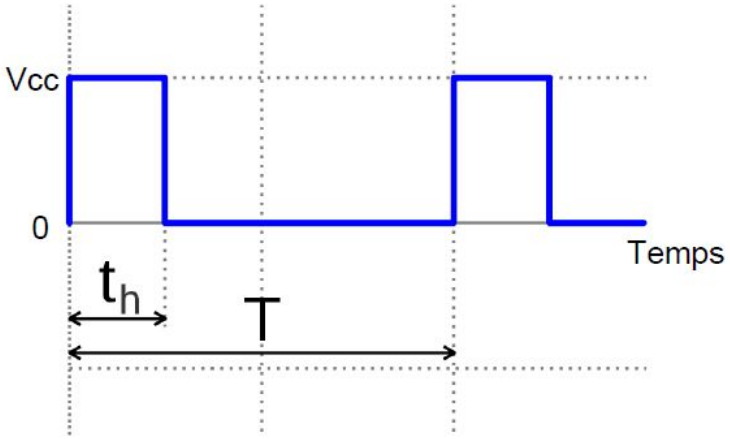


Figure II.7. Représentation du rapport cyclique

$$V_{moy} = \frac{t_h V_{cc}}{T}$$

Chaque microcontrôleur possède deux sorties PWM. Donc, on dispose d'un total de six sorties analogiques pour commander la vitesse de rotation des moteurs.

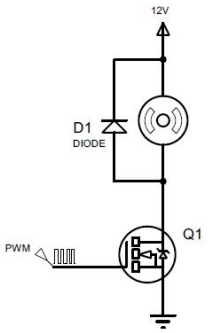


Figure II.8. Commande d'un moteur en utilisant un signal PWM

II.5. Protocoles de communication

Une communication entre le maitre et les esclaves est nécessaire afin d'assurer le fonctionnement de toutes les articulations du robot, mais aussi pour recevoir les actions à effecteur par l'utilisateur à travers une interface sur ordinateur. Pour cela, on a choisi l'interface de communication USB avec l'ordinateur et le bus I²C pour communication entre le maitre et les esclaves

II.5.1. Communication USB

Les avantages du port USB sont nombreux : faible coût de l'interface, alimentation possible des dispositifs via le câble, indépendance vis à vis des machines hôtes, Hot Plug & Play (c'est à dire branchement et débranchement sans avoir besoin d'arrêter le PC), jusqu'à 127 périphériques possibles, fiabilité et sécurité (détection et correction d'erreurs), plusieurs vitesses possibles et 4 types de transferts **[2]**.

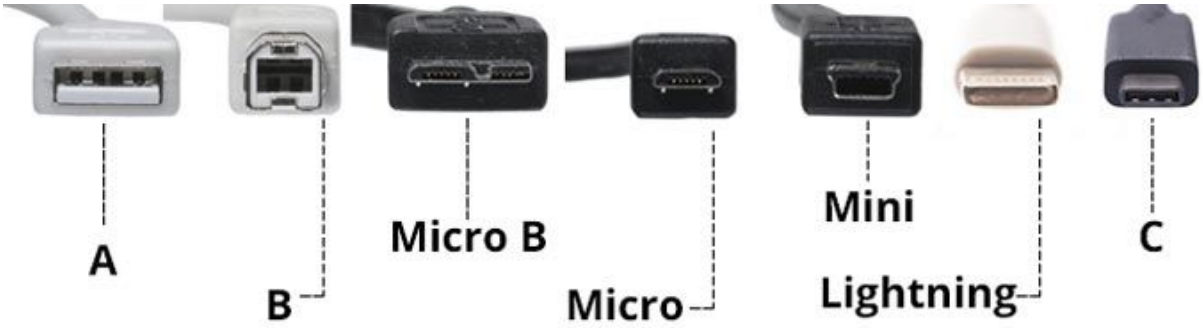


Figure II.9. Deferent type de connecter USB

II.5.2. Bus I²C

Le bus série I²C permet de transmettre des informations de façon synchrone entre divers circuits connectés sur le bus. Le protocole de la liaison est du type maitre /esclave. Chaque circuit est reconnu par son adresse et peut être soit transmetteur soit receveur de l'information. Ces circuits peuvent être : Un ordinateur, un microcontrôleur, un microprocesseur, une mémoire, un périphérique (clavier, écran,...) etc.

Dans le protocole du bus I2C le circuit maître est celui qui demande un transfert d'information sur le bus et qui génère le signal d'horloge qui permet le transfert. Ainsi un circuit adressé est considéré comme un esclave. Le bus I2C est un bus multi maître. Cela signifie que plusieurs circuits peuvent contrôler le bus.

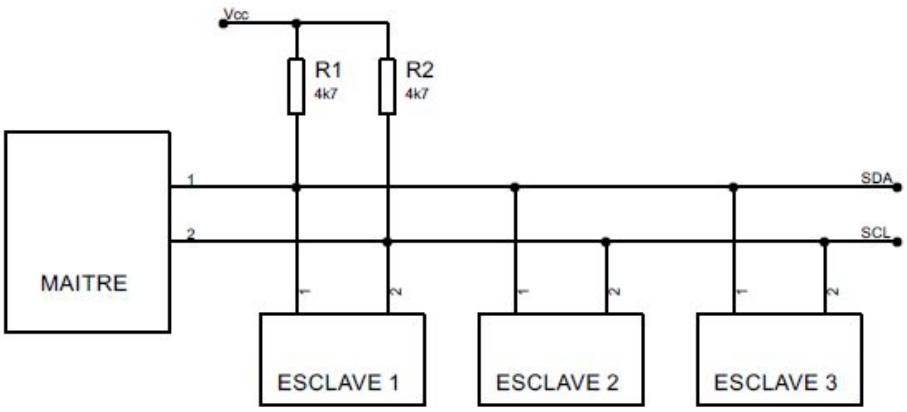


Figure II.10. Branchement entre maitre et esclaves

Les lignes *SDA* et *SCL* sont bidirectionnelles, connectées à plus VCC par l'intermédiaire de deux résistances de tirage. Quand le bus est libre, c'est à dire quand il n'y a pas de transfert de données, les deux lignes sont à l'état haut **[3]**.

II.6. Etage d'alimentation

Toute système électronique à besoins d'une alimentation qui fournit une tension continu lui permettent de fonctionner, alors on a opté pour une alimentation linéaire classique (transformateur, redresseur, régulateur).

II.6.1. Transformateur

Le transformateur est l'élément clé pour les alimentations traditionnelle, ou il joue le rôle d'abaisseur de tension alternative pour pouvoir la redresser et avoir une tension continu en suite, il comporte deux bobines isolées l'une de l'autre et fixer sur un support ferromagnétique pour diminuer les pertes d'énergie

Figure II.11. Transformateur

La transformation se fait selon des lois théoriques ou on considère le transformateur parfaite. $m = \frac{U_1}{U_2} = \frac{I_1}{I_2} = \frac{N_1}{N_2}$

Table II.1. Les notations

m	le rapport de transformation
U_1	la tension au primer
U_2	la tension au secondaire
I_1	le courant au primer
I_2	le courant au secondaire
N_1	nombre de spire au primer
N_2	nombre de spire au secondaire

II.6.2. Redresseurs

Les montages redresseurs, souvent appelés simplement redresseurs, sont les convertisseurs de l'électronique de puissance qui assurent directement la conversion alternatif-continu. Alimentés par une source de tension alternative monophasée, ils permettent d'alimenter en courant continu le récepteur branché à leur sortie.

Les diodes sont utilisées pour un redresseur non commandé, soit une seule (redresseur mono alternance) ou bien un pont (redresseur double alternance) ce qu'est notre cas.

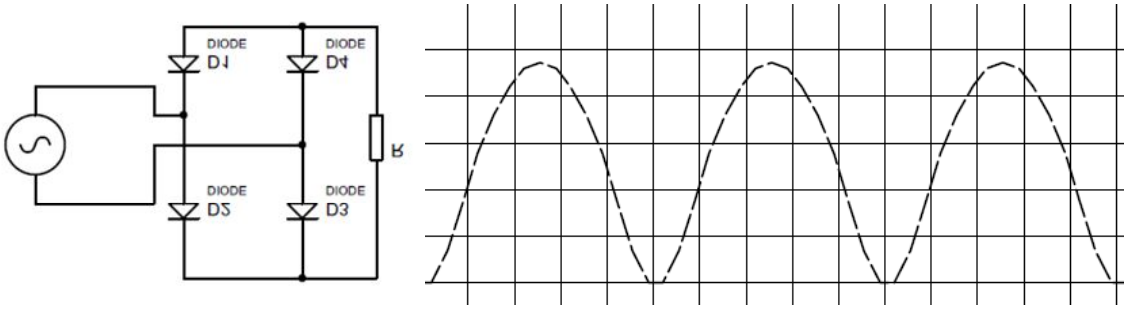


Figure II.12. Représentation d'un redresseur

II.6.3. Régulateurs de tension

Les montages électroniques sont sensibles à la variation de la tension. Et pour parvenir à ce besoin, on utilise un régulateur de tension, ce dernier peut être composé d'un ensemble de composants classiques (résistances, diodes zener et transistor par exemple), mais il peut aussi être de type "intégré" et contenir tout ce qu'il faut dans un seul et même boitier, pour faciliter son usage, pour notre projet on a utilisé deux régulateurs 12V et 5V à savoir le 7812 et 7805.

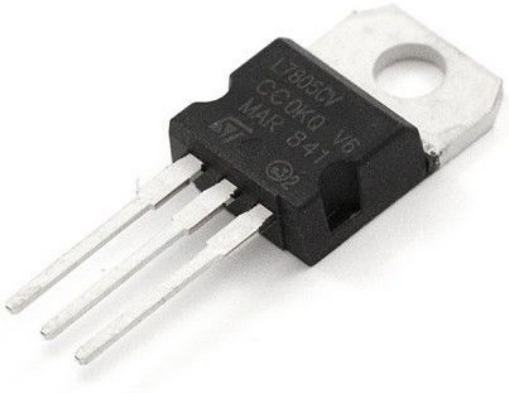


Figure II.13. Régulateurs de tension

II.6.4. Réalisation d'Alimentation

On se contentera de réaliser une alimentation classique vu qu'on ne dispose pas des composantes nécessaires pour la réalisation d'une alimentation à découpage.

L'alimentation fournit deux sorties, une de 12V pour alimenter les moteurs et l'autre de 5V qui va servir pour l'alimentation de la parte logique.

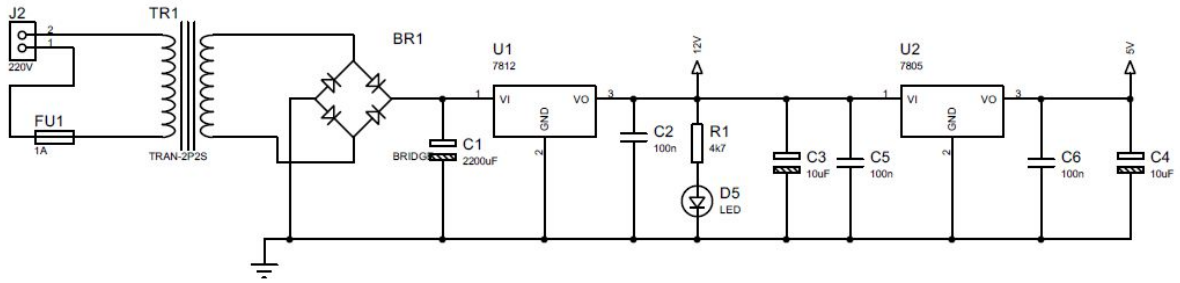


Figure II.14. Schéma de l'alimentation 12v et 5v

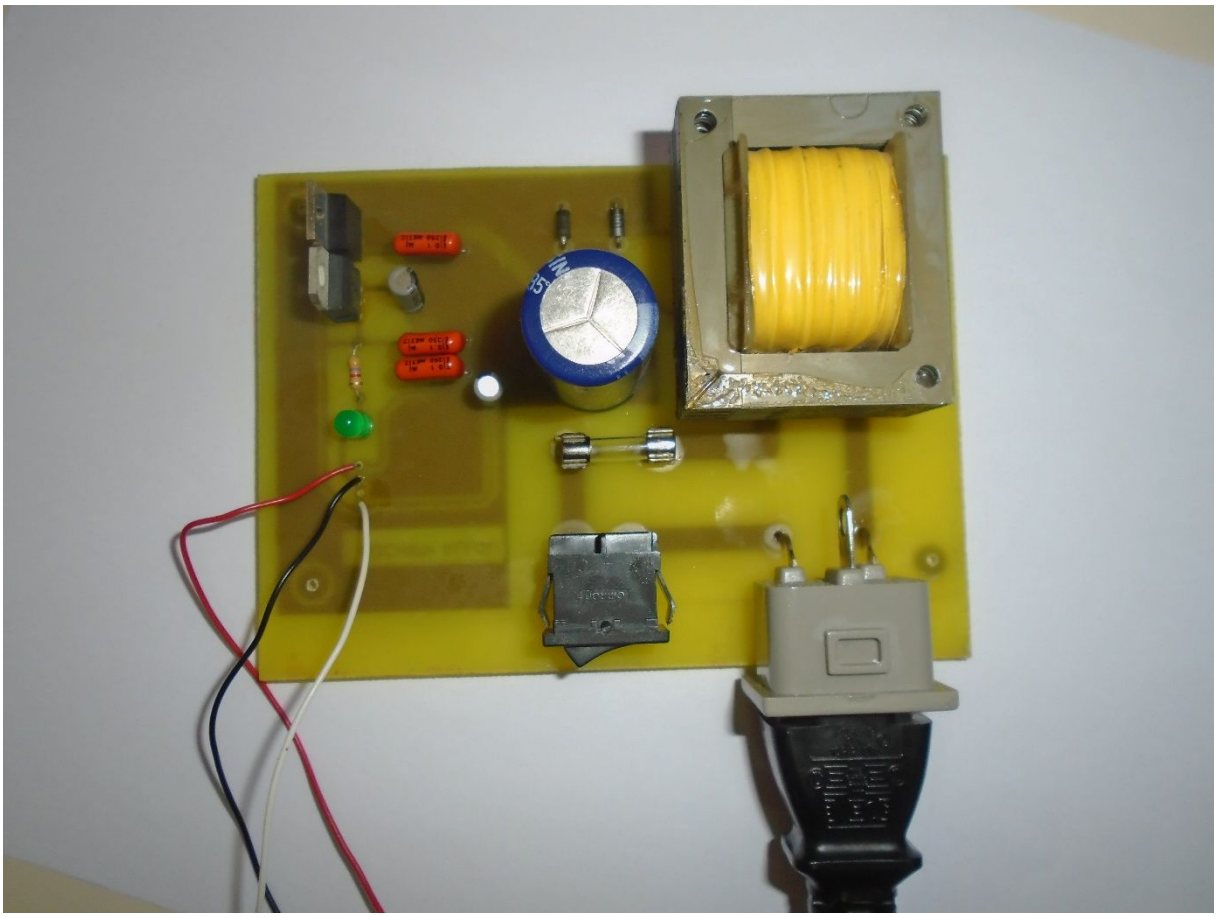


Figure II.15. Image représentative de l'alimentation 12v et 5v

II.7. Partie puissance

Il est impossible de connecter directement les sorties du microcontrôleur avec un moteur, vue la puissance mise en jeu. On utilise alors des éléments de l'électronique de puissance pour alimenter les moteurs sans endommager les microcontrôleurs.

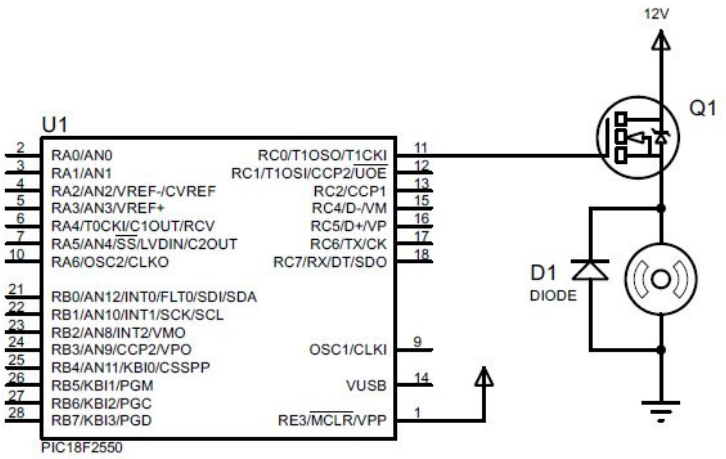


Figure II.16. Branchement d'un moteur avec un PIC

II.7.1. Pont H

Une articulation d'un robot exige deux sens de rotation. Pour piloter le moteur dans les deux sens, il est nécessaire de passer à la structure en H. selon la paire d'interrupteur activée (T1-T4 ou T3-T2), le courant passe dans un sens ou dans l'autre.

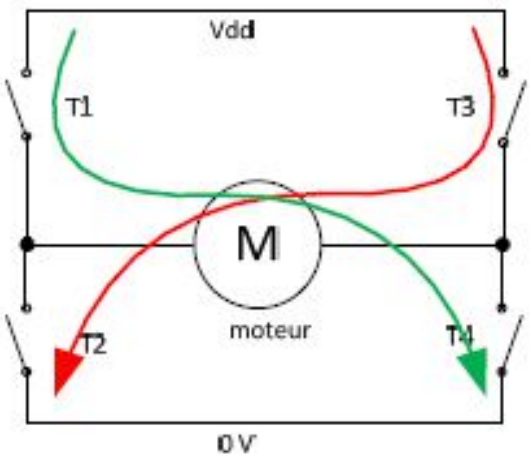


Figure II.17. Représentation du Pont H

D'un point de vue électronique, les interrupteurs sont réalisés par des transistors MOS discrets. Mais, ils existent aussi des circuits intégrés en demi-pont ou pont-entier incluant MOS ou SENSEFET logique de commande, sécurité diverses. Car, certaines technologies modernes le permettent pour des tensions de pont allant jusqu'à 60V et des courants de quelques dizaines d'ampères.

L'intérêt des MOS, si bien choisis, est qu'ils ont une résistance R_{on}, à l'état passant, très faible. Ce qui limite les pertes de conduction au minimum. Par ailleurs, ils ont une diode intrinsèque interne que l'on peut utiliser en roue libre **[5]**.

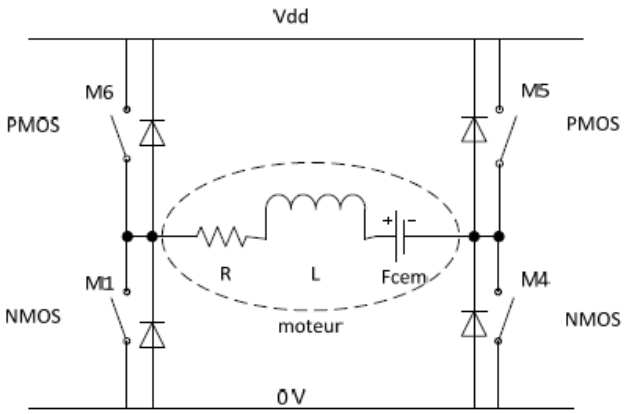


Figure II.18. Pont en H transistor MOS

Le L293D est l'un des circuits intégrés en pont H. Il possède deux ponts. Trois entrées de commande sont prévues pour chaque pont : deux entrées pour le sens de rotation et une pour la validation qui sera utilisé pour le signal PWM.

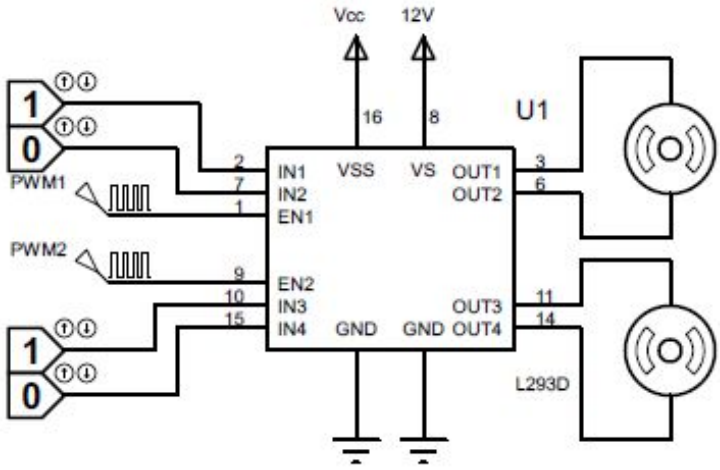


Figure II.19. Pont en H : L293D

II.8. Acquisition de données

La carte est équipée de 6 entrées destinées pour l'acquisition de données concernent les états des articulations, les entrées sont repartitionnées de *CAPIN1* jusque *CAPIN6*, les signaux passent par un filtre passe-bas afin d'avoir une meilleure lecture, pour qu'il soit envoyé aux microcontrôleurs.

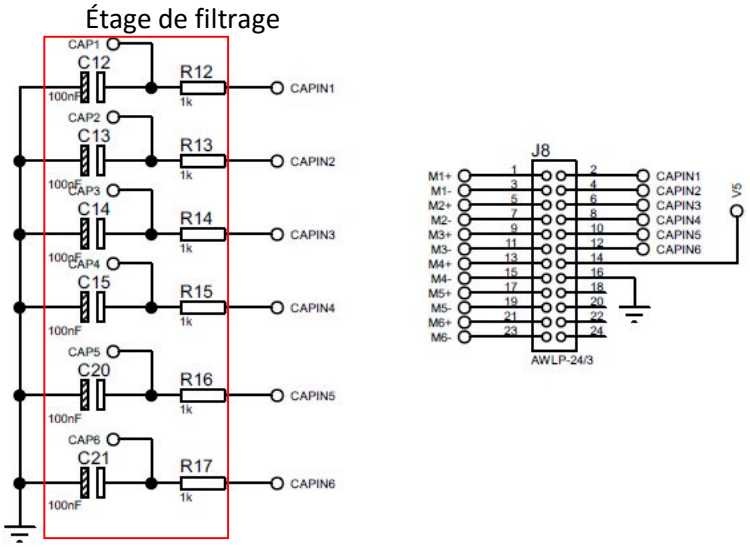


Figure II.20. Acquisition de données et filtrage

II.9. Coté commande

L'ensemble des commandes sont générées et envoyées aux *ponts H* grâce à des programmes implémentés sur trois microcontrôleurs, on a utilisé les *Ports A* pour l'acquisition des données et le *Port B* coté maitre et les Ports D coté esclaves pour la génération des commande des *Pont H*.

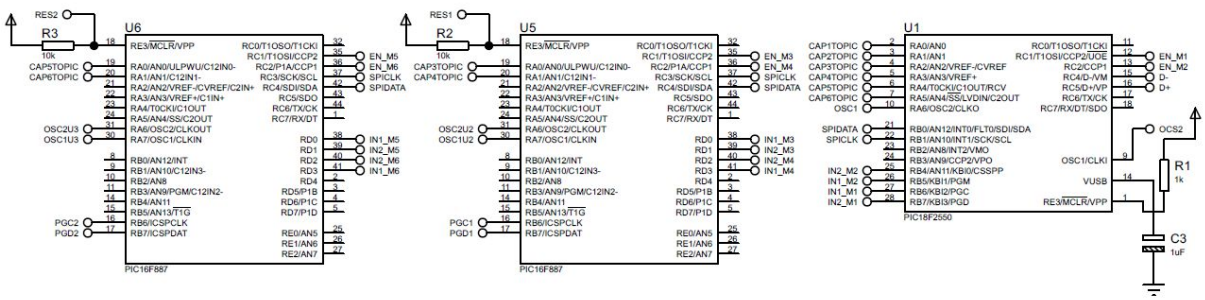


Figure II.21. Les microcontrôleurs maitre et esclaves

II.10. Schéma électrique

Pour la réalisation du circuit imprimé, un schéma électrique est d'abord élaboré à l'aide du logiciel Isis Proteus. La majorité des composants sont soudés sur surface. Cela permet d'avoir une carte plus petite.

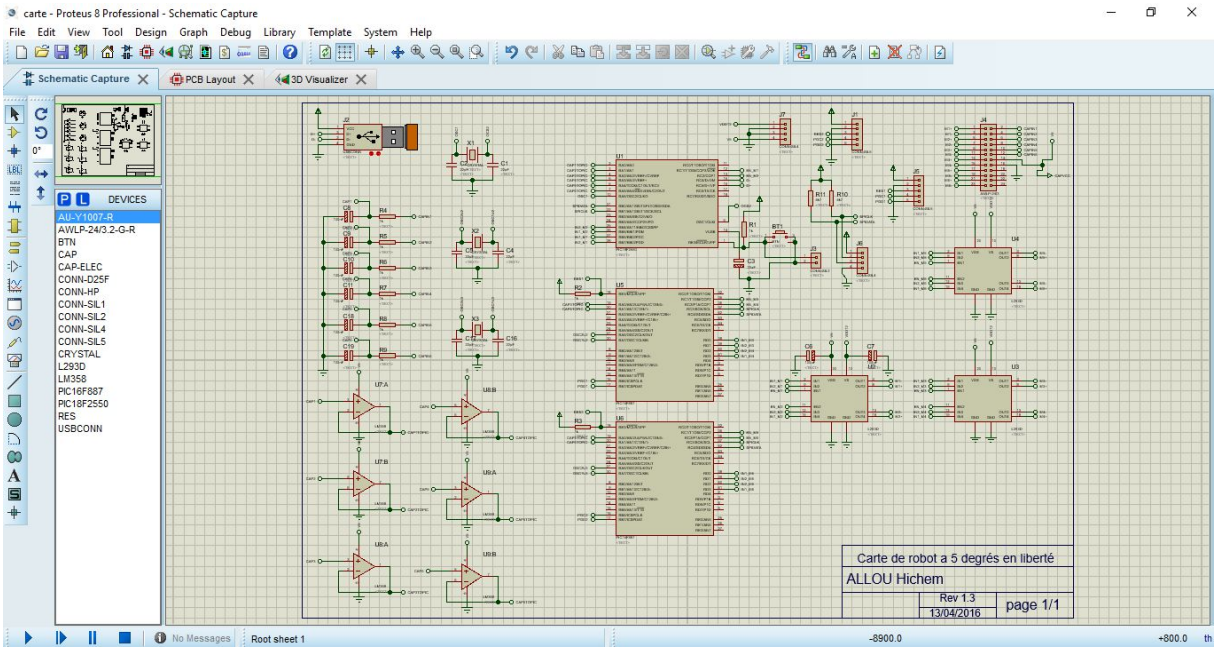


Figure II.22. Le schéma électrique sur isis

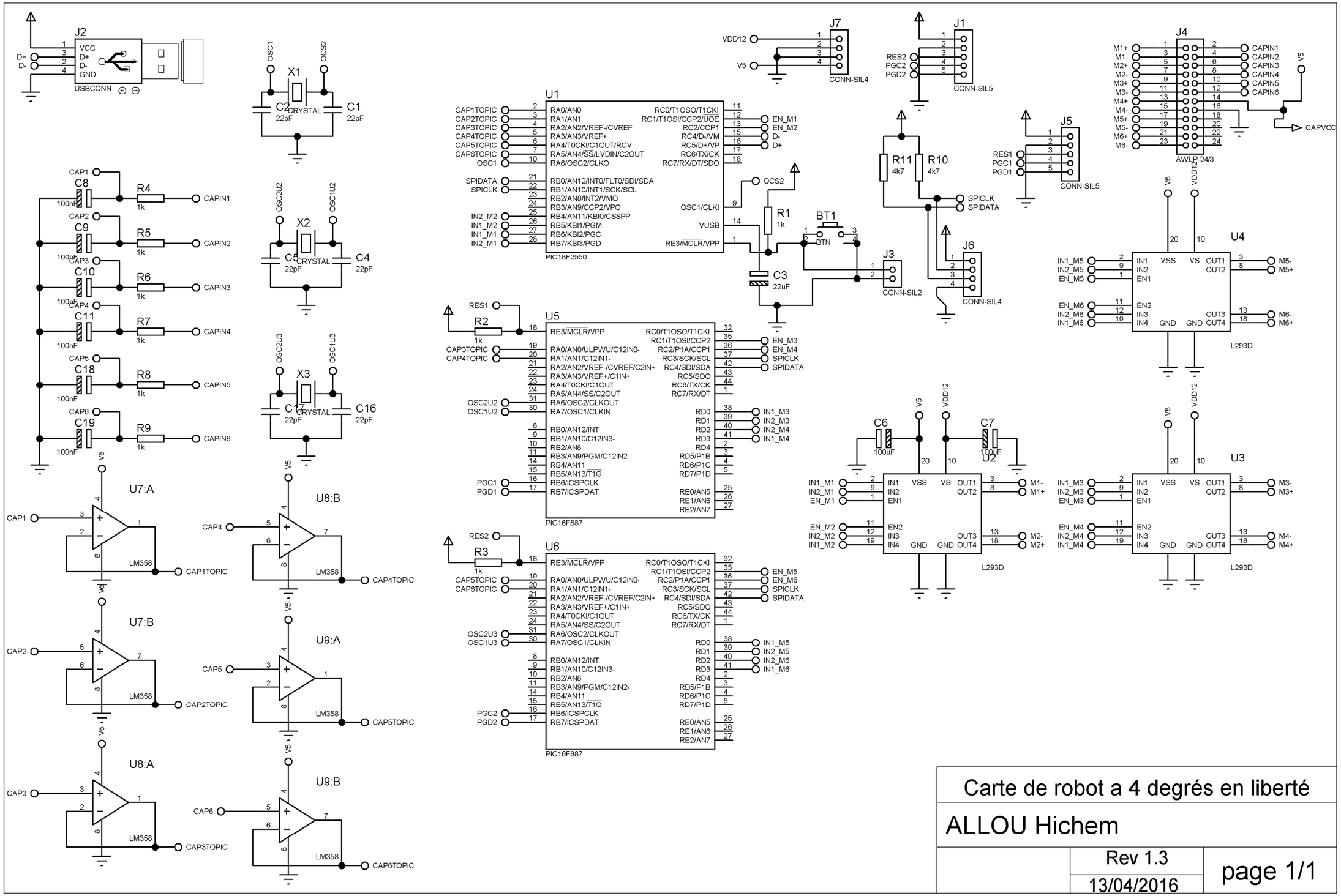


Figure II.23. Schéma électrique

II.11. Réalisation de typon

Le typon est le dessin qui sera utilisé pour développer la carte électronique. Il a été réalisé avec le logiciel *ARES*.

Pour avoir un bon typon, il faut bien une meilleure disposition des composants. Il faut aussi remplir le vide d'une masse pour réduire le bruit analogique.

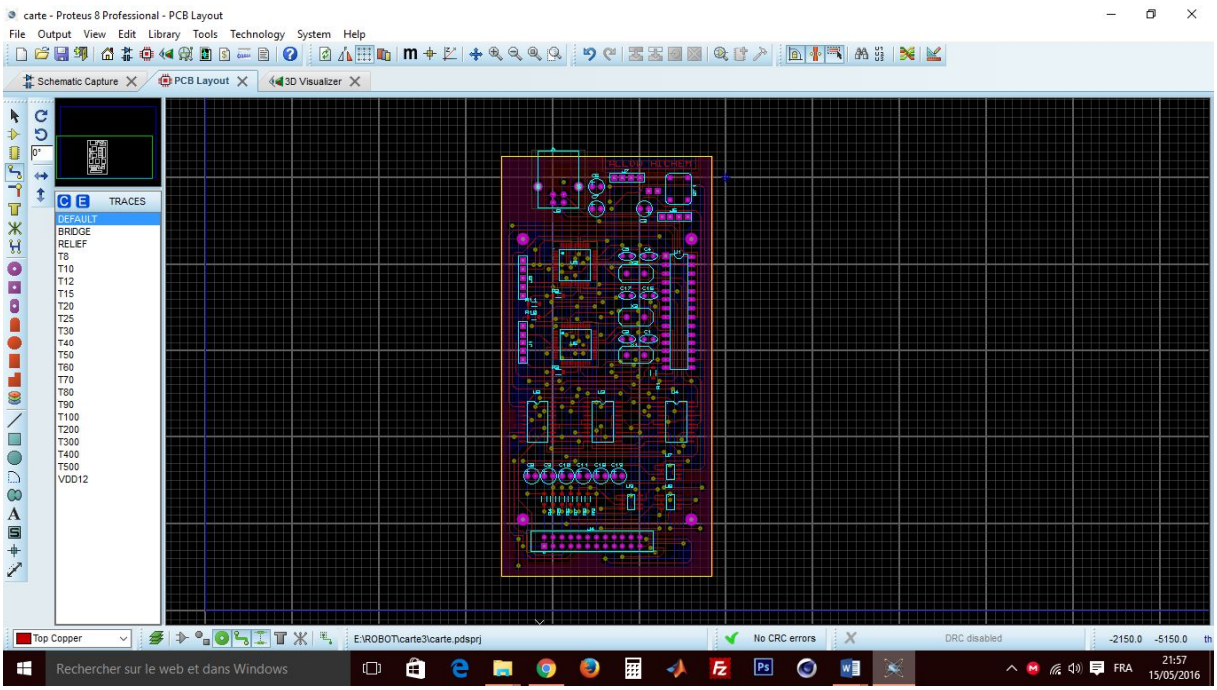


Figure II.24. Le logiciel **ARES**

Une fois fini, on exporte les deux faces du typon au format PDF pour pouvoir les imprimer sur un papier calque, afin de commencer le développement du circuit.

II.12. Réalisation de la carte électronique

La réalisation de la carte passe par trois étapes : l'insolation, la révélation et la gravure. Le dévalement du circuit se fait sur une plaque d'époxy pré sensibilisée double face.

II.12.1. Insolation

Durant l'insolation, on expose la plaque d'époxy au rayonnement ultraviolet. La résine qui couvre la plaque et qui n'est pas protégée par le typon, entre en réaction au contact des UV. L'exposition se fait pendent trois minutes.

Remarque : les manouvres avant le début de l'insolation se font sous une Lumière rouge, afin de ne pas endommager la plaque d'époxy care elle est trop sensible.

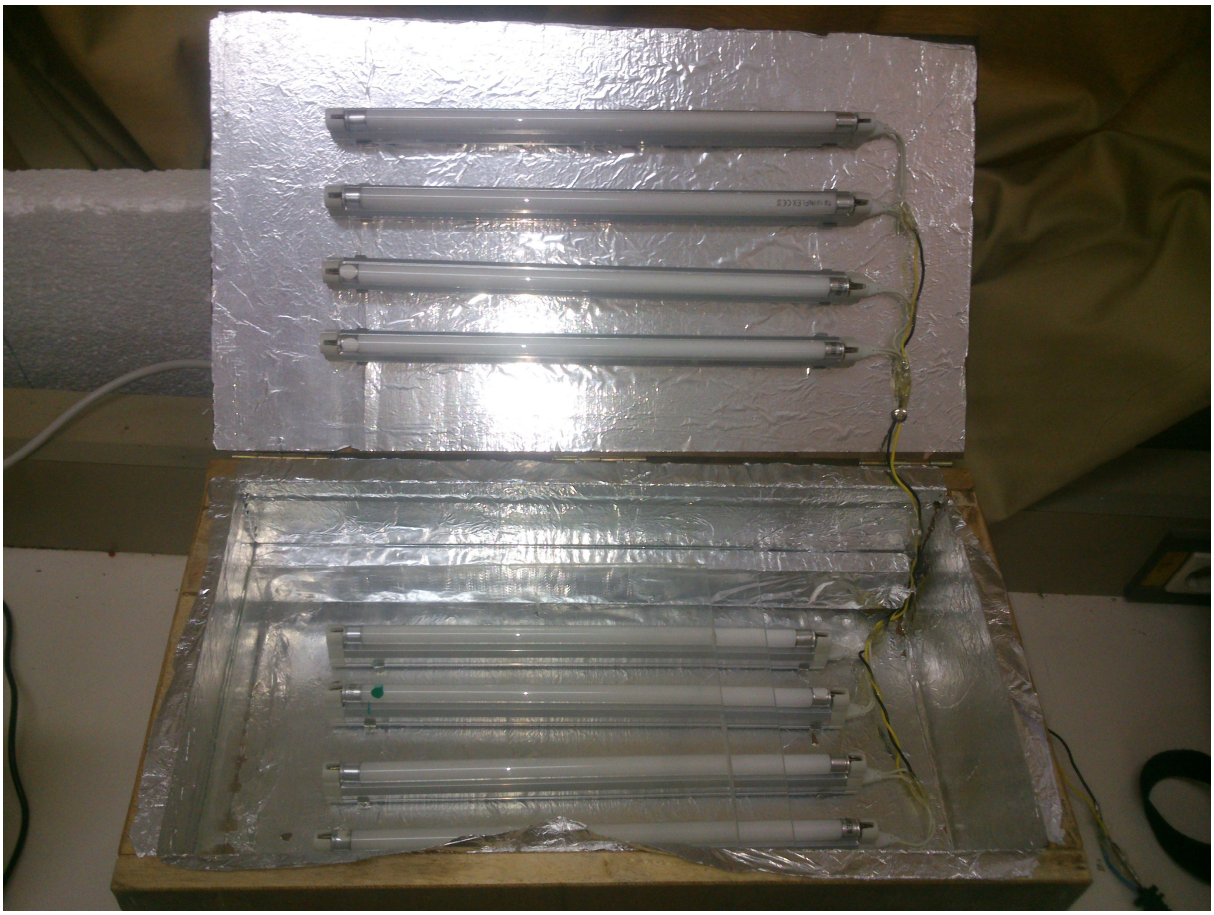


Figure II.25. L'Insoleuse

II.12.2. Révélation

La révélation est la deuxième étape où on trempe la carte dans une base forte. La solution attaque la partie insolée. Ainsi, il ne reste que le circuit électronique. Le processus prend quelques secondes.

Figure II.26. Image du révélateur solide

II.12.3. Graveur

C'est l'étape finale qui consiste à dessouder tout le cuivre non protégé par la résine photosensible. Généralement on utilise l'acide perchlorure de fer($FeCl_3$). Mais, il existe d'autres produits qui peuvent être utilisés. Une surveillance du processus est obligatoire pour s'assurer d'un bon résultat. Si on expose la carte plus que il le faut, on risque d'endommager la carte et tout est à refaire. Il reste qu'à enlever la résine photosensible, en utilisant de l'acétone. Enfin, on teste la continuité des pistes et soude les points via.

Note : Il est très important d'utiliser des gants pour manipuler les produits chimiques.

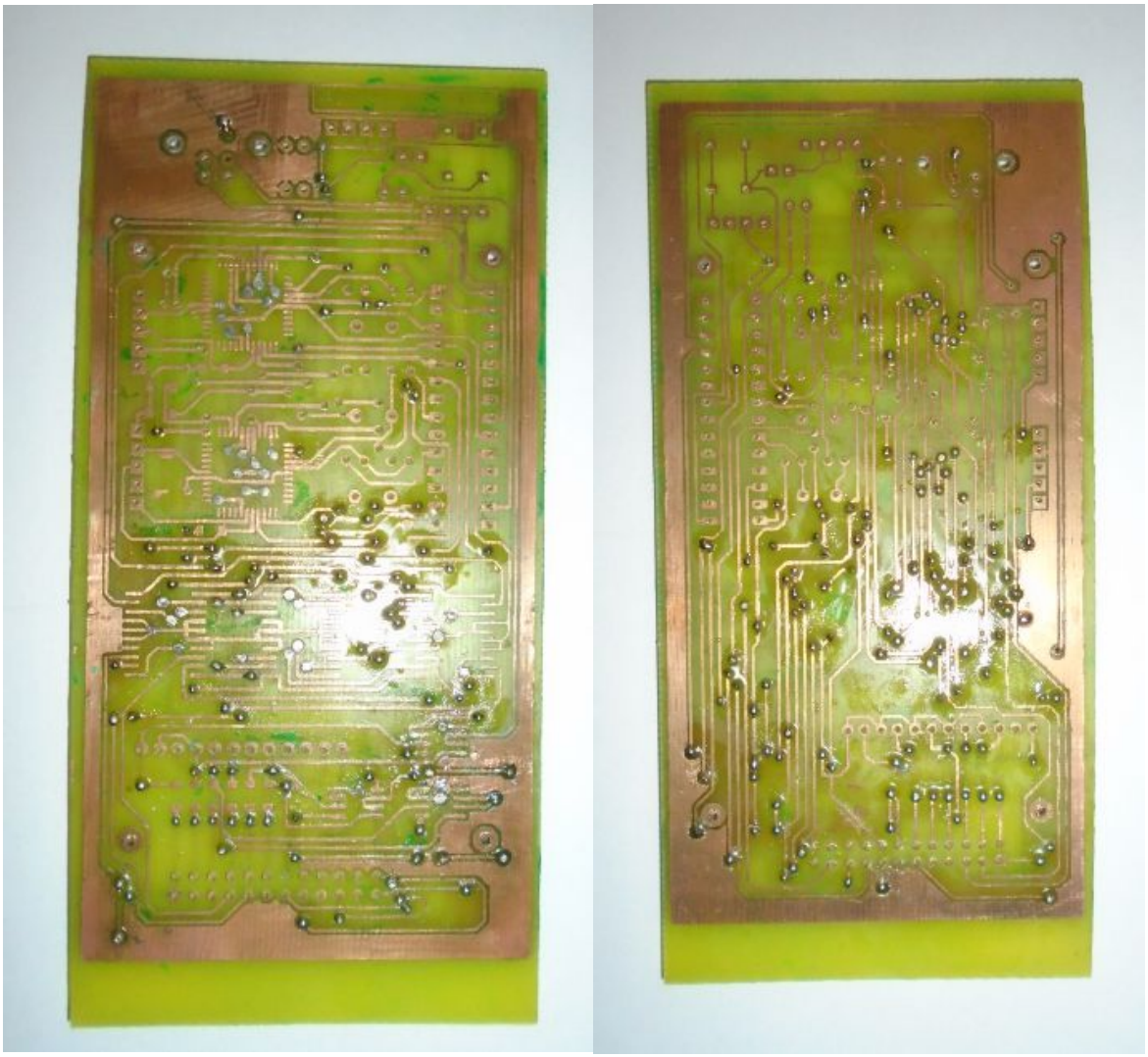


Figure II.27. Le circuit final

II.12.4. Circuit finalisé

Une fois la carte finalisée, il est temps de souder les composants électroniques et on teste la présence de courts-circuits.

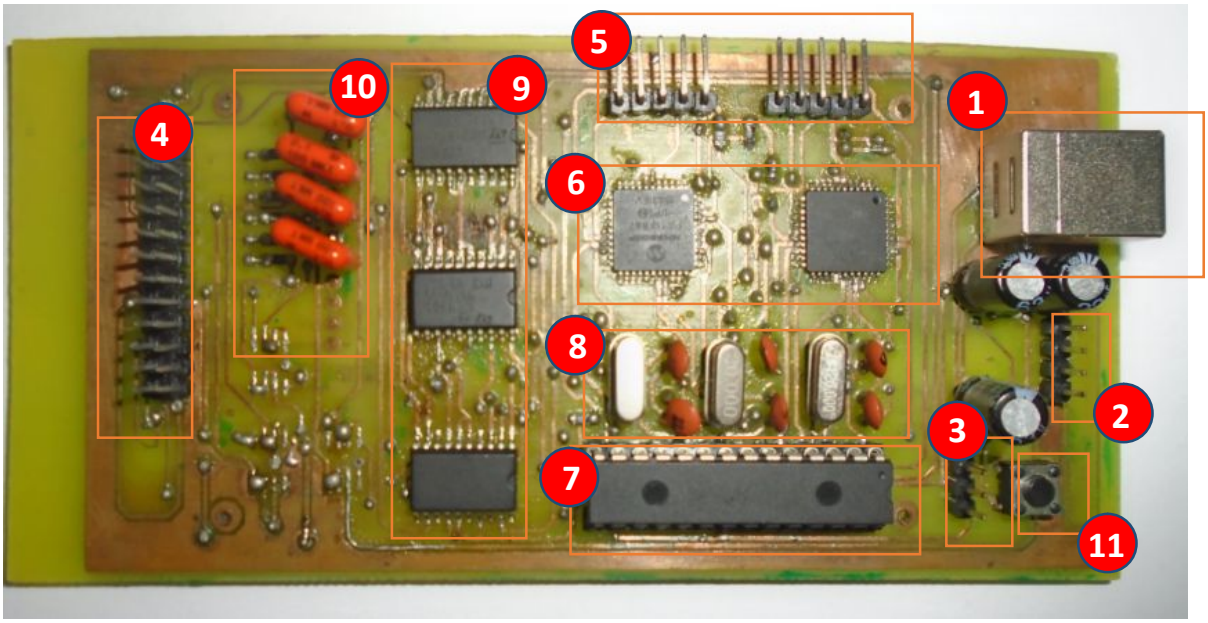


Figure II.28. La carte finale

La liste ci-dessous représente les différents éléments de la carte.

1. Le connecteur USB.
2. Connecteur prévu pour l'alimentation.
3. Connecteur prévu pour l'extension en utilisant le bus I²C.
4. Connecteur prévu pour l'interaction avec le bras robotique (récolte des données et envoi des commandes).
5. Deux connecteurs sont prévus pour l'injection des programmes aux deux microcontrôleurs esclaves (PIC16f887).
6. Deux microcontrôleurs esclaves (PIC16f887).
7. Microcontrôleur maitre (PIC18f2550).
8. Oscillateurs à quartz.
9. Ponts H (L293DD).
10. Etage de filtrage passe bas.
11. Bouton reset.

II.13. Conclusion

Ce chapitre donne une présentation des microcontrôleurs utilisés, en particulier, les protocoles de communication avec l'ordinateur ou en configuration maitre-esclave. Puis, on a exposé la carte de puissance. La dernière partie est consacrée aux détails de réalisation pratique des cartes.

Pour réaliser une carte électronique, un nombre de connaissances est requis afin d'avoir un résultat satisfaisant, notamment le choix des composants à utiliser, en fonction des besoins ou des résultats finaux du projet. Il faut aussi avoir les connaissances de base en électronique (les schémas de base, test des composantes, soudure…).

Chapitre III

Outils informatiques et programmation

III.1. Introduction

De nos jours, l'informatique est indispensable quel que soit le demain. Elle rend l'exécution des tâches et l'acquisition des données plus faciles. Le domaine de l'électronique a beaucoup bénéficié de ce que l'informatique a apporté. Il est plus facile de réaliser des systèmes complexes à faible coût grâce aux composants programmables. Ce chapitre présente les outils informatiques utilisés ainsi que la partie programmation de notre projet.

III.2. Outils informatique utilisés

Afin de parvenir à réaliser un système programmable, des connaissances en informatique sont requises, à savoir l'algorithmique, la programmation, utilisation des logiciel de conception de circuits électroniques, ainsi que les logiciels de design 3D pour la simulation du robot et l'interface utilisateur.

III.2.1. logiciel de conception

Il n'est pas toujours facile de faire les schémas des cartes électroniques à la main. Il faut impérativement maîtriser au moins un logiciel destiné à ce type d'applications.

III.2.1.1. Proteus

Proteus est une suite logicielle destinée à l'électronique, développé par la société *Labcenter Electronics*. Il possède trois logicielles :

- **ISIS** : c'est le logiciel principal. Il est utilisé pour la schématisation des circuits électroniques, mais aussi pour la simulation.
- **ARES** : un logiciel de routage qui complète parfaitement ISIS. Un schéma électrique réalisé sur ISIS peut alors être importé facilement sur ARES pour réaliser le PCB de la carte électronique.
- **3D Viewer** : permet d'avoir une vue en 3d du projet avent la réalisation.

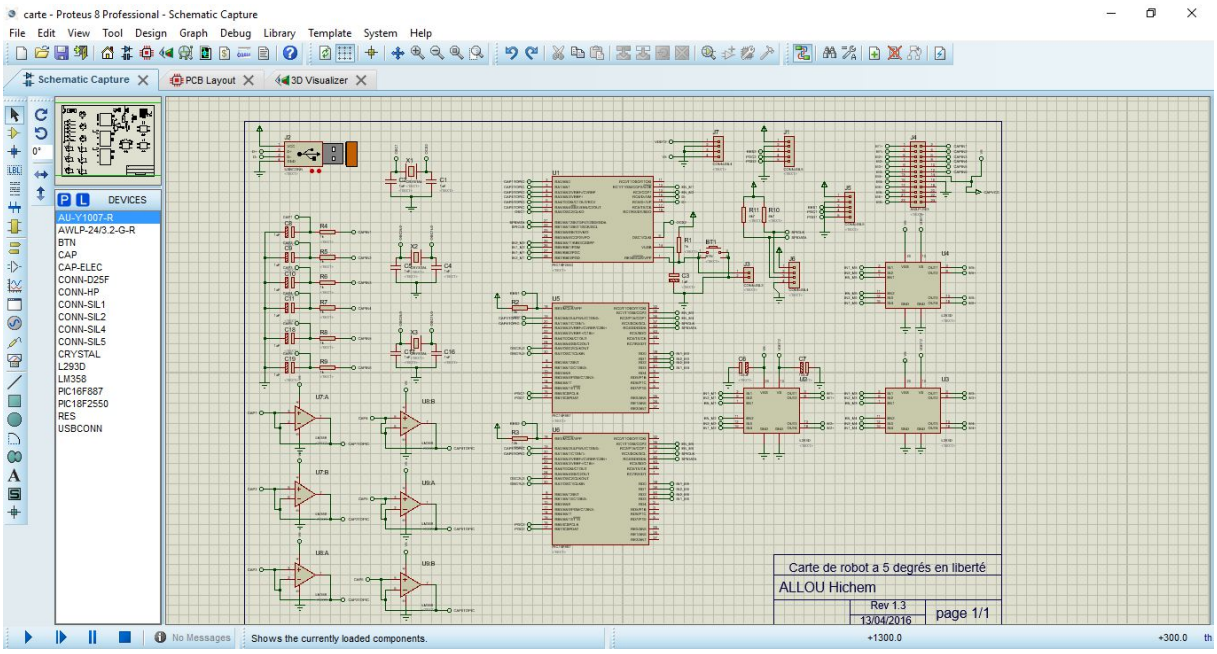


Figure III.1. Vue du logiciel *ISIS*

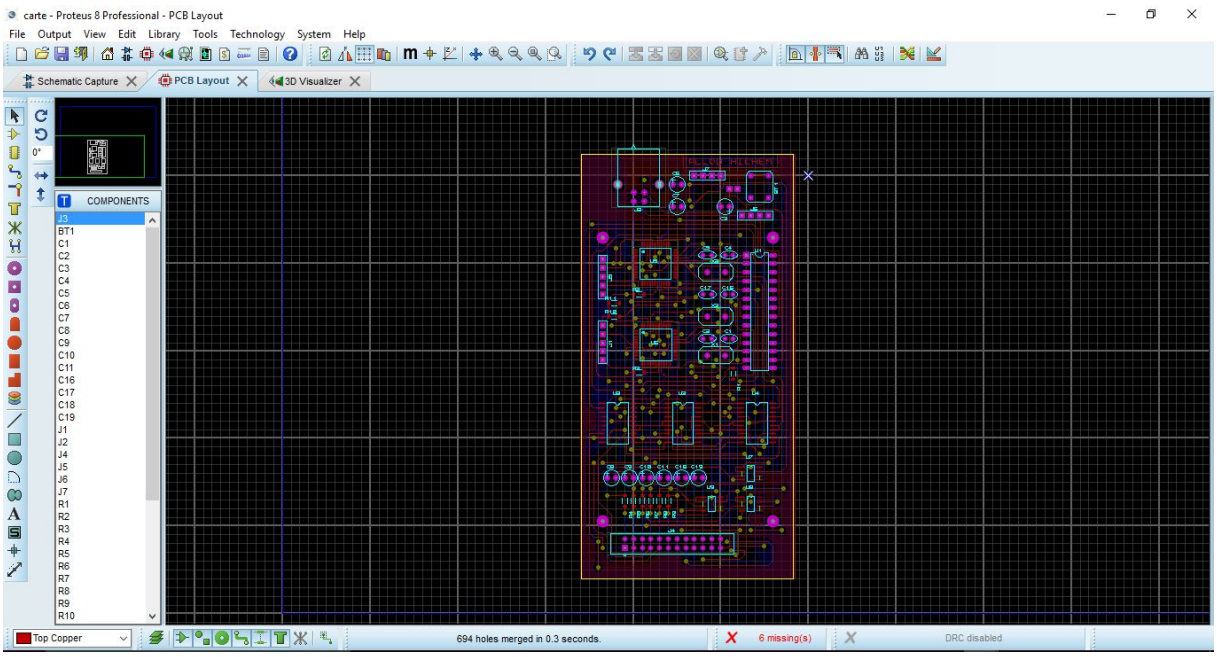


Figure III.2. Vue du logiciel *ARES*

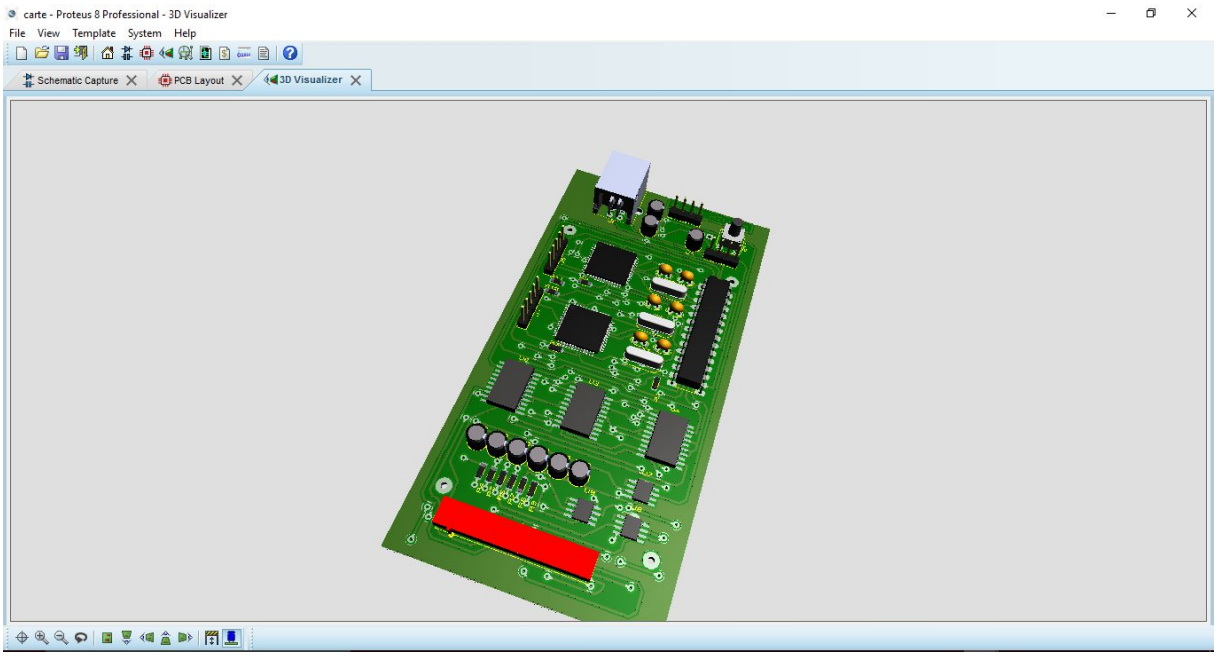


Figure III.3. Vue en 3D

III.2.1.2. CINEMA 4D

Dans le but d'avoir une visitation 3D sur l'interface utilisateur, on a au recours un logiciel spécialisé (CINEMA 4D). C'est un logiciel de création 3D développé par la société allemande *Maxon*.

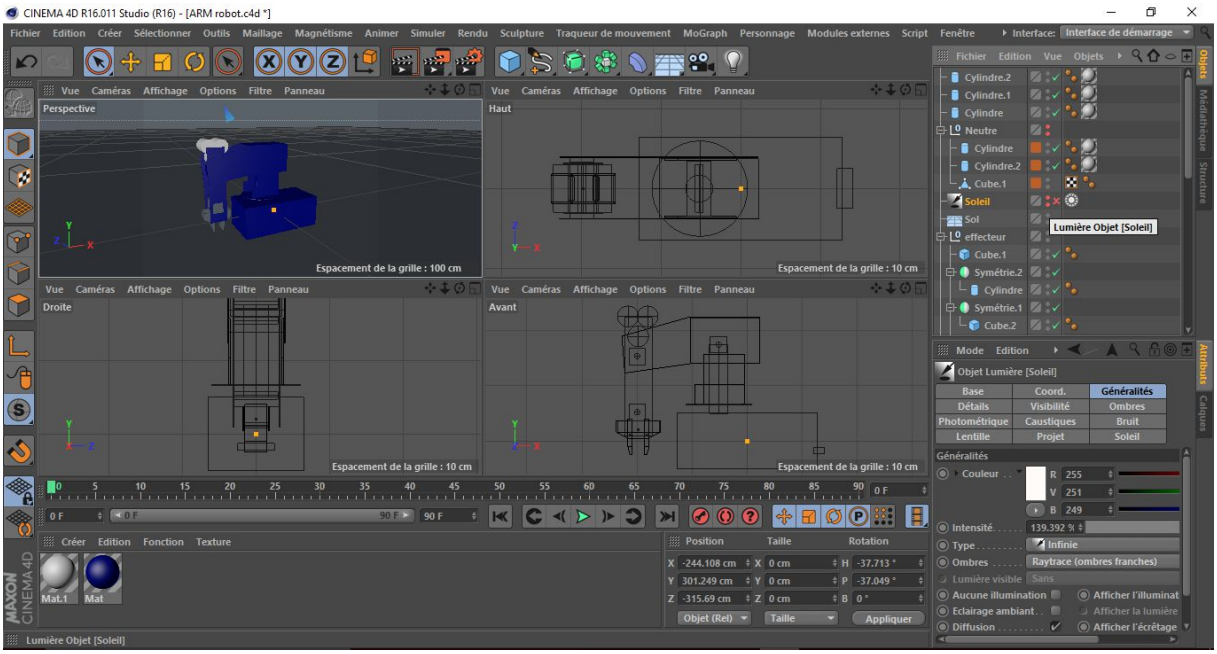


Figure III.4. Vue du logiciel *CINEMA 4D*

III.2.2. Logiciels de programmation

III.2.2.1. Microsoft visual studio

Microsoft Visual Studio est une suite d'environnement de développement et d'exécution parmi eux le C#. Sa syntaxe est très proche de celle de Java, mais il existe des différences subtiles entre les deux langages. Elle est également voisine de celle du C et de celle du C++**[5]**.

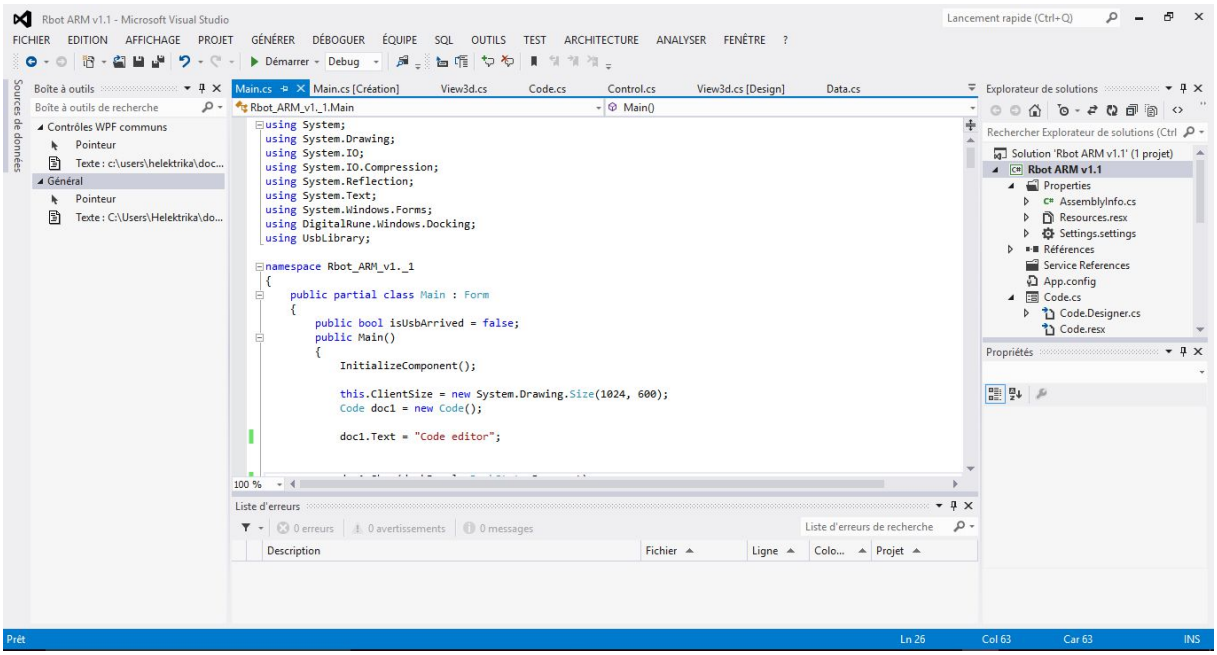


Figure III.5. Vue du logiciel *Microsoft Visual*

III.2.2.2. Mikroc

L'environnement de développement *Mikroc* est conçu pour programmer les microcontrôleurs Pic, en utilisent le langage C. Il est doté d'une large bibliothèque permettant à l'utilisateur d'exploiter les microcontrôleurs sans être vraiment un spécialiste.

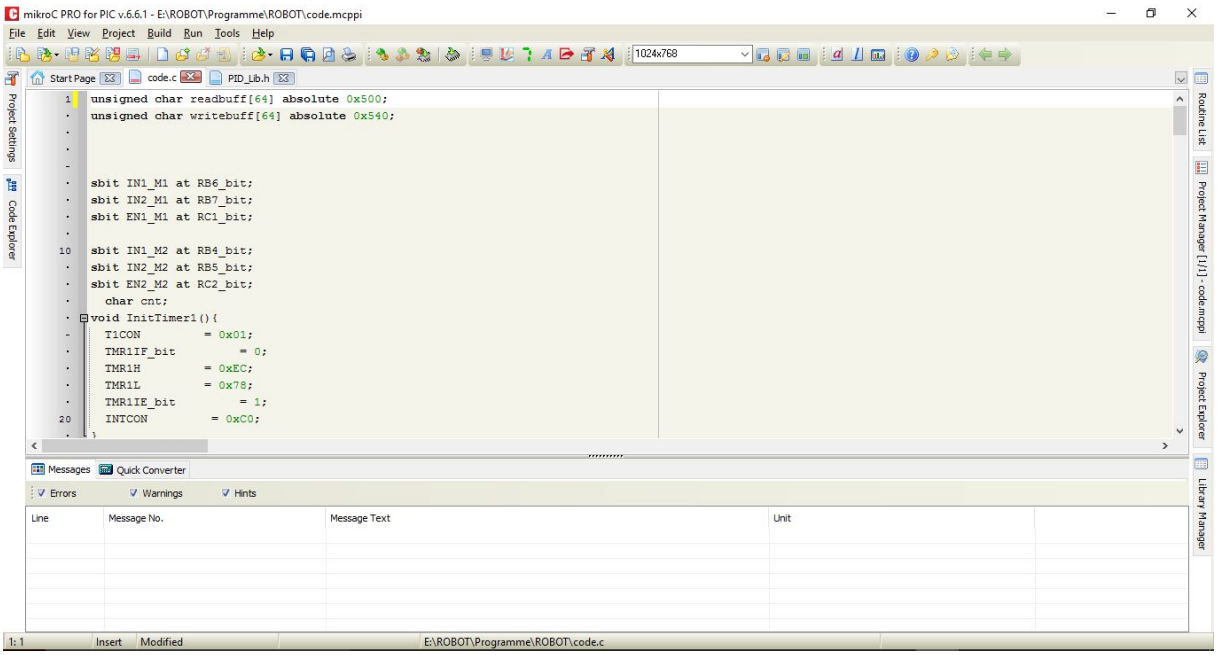


Figure IV.6. Vue du logiciel *Microc*

III.2.2.3. USB-PicProg

Il ne s'agit pas d'un langage de programmation, mais d'un outil qui permet de charger les programmes dans les microcontrôleurs. C'est un programmateur open source.

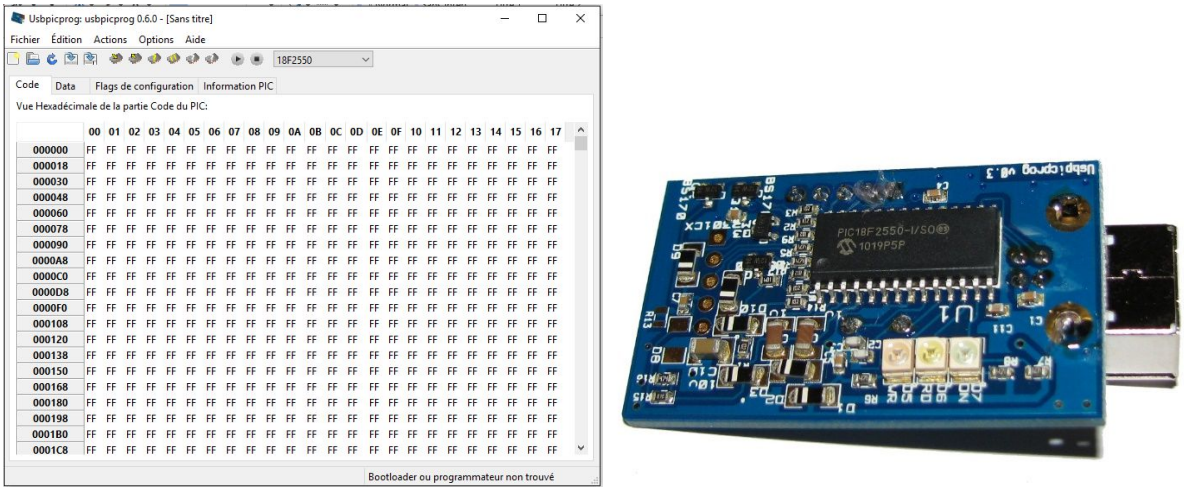


Figure III.7. Vue du logiciel *Usb-PicProg et le Matériel*

III.3 Programmations

Une application d'interaction entre le robot et l'utilisateur est très utile pour faciliter l'envoi de commandes et exécuter des tâches. Pour cela, on a conçu une application Windows qui se chargera de recevoir les données à partir de la carte de commande et aussi une visualisation en temps réel des états du robot.

Afin que la carte électronique puisse recevoir les commandes de l'interface utilisateur ou bien envoyer l'état du robot à l'interface, des programmes ont été développés.

III.3.1. Interface utilisateur

L'organigramme ci-dessus représente d'une manière générale l'approche opté pour l'interface utilisateur que on va détailler par la suite.

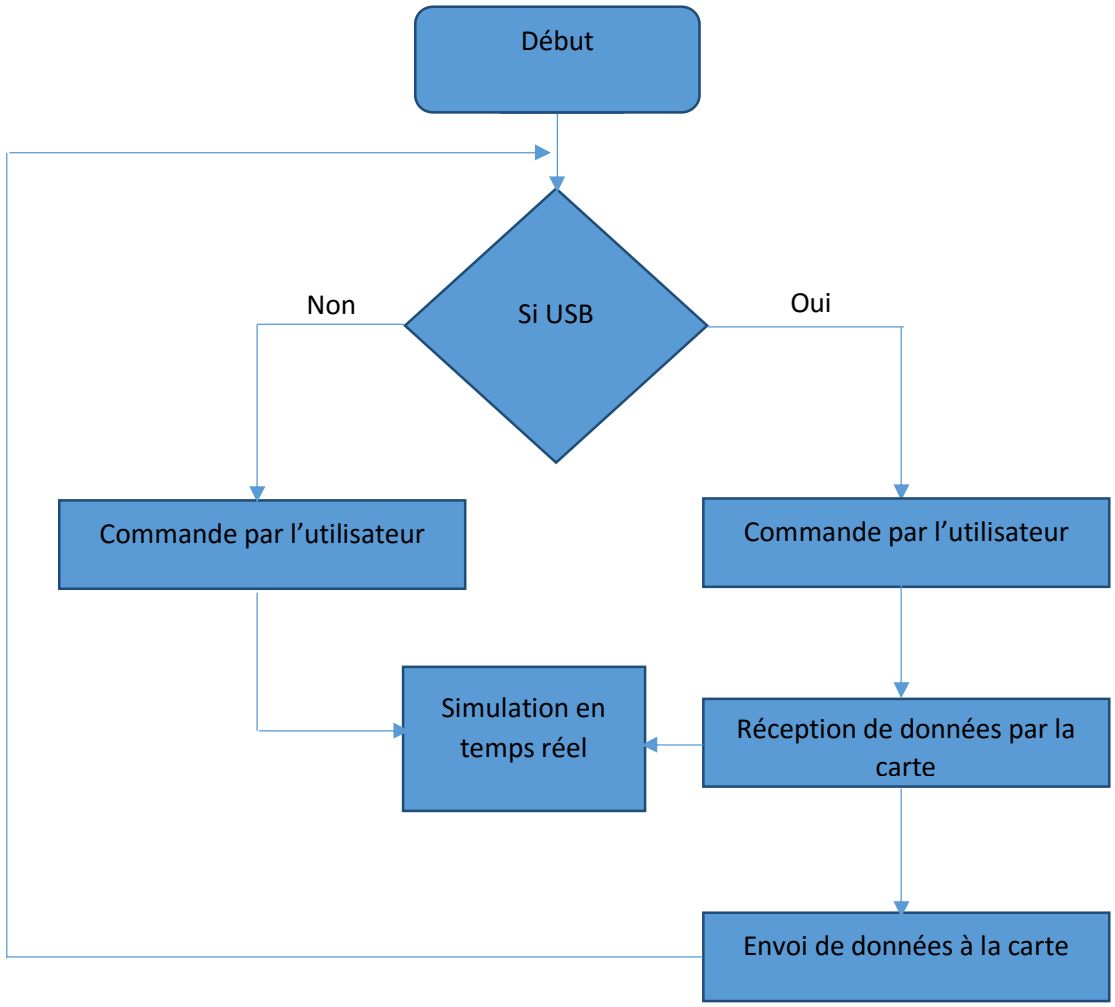


Figure III.8. Organigramme de l'interface utilisateur

III.3.1.1. USB

La connexion entre la carte électronique et l'interface utilisateur se fait via une liaison USB. Le côté logiciel ce fait à l'aide d'une bibliothèque pour le langage C#, appelée *UsbLibrary.dll*. Afin de pouvoir utiliser cette Library, des fonctions prédéfinies sont obligatoires pour fonctionnement correcte.

-La fonction *OnHandleCreatedsert* permet de creer l'objet USB pour pouvoir l'utiliser en suite. Elle est définie comme suit :

```
protectedoverridevoid OnHandleCreated(EventArgs e)
{
      base.OnHandleCreated(e);
      usb.RegisterHandle(Handle);
}
```

-La fonction `WndProcest` est utilisée pour détecter les événements du port USB.

```
protectedoverridevoid WndProc(refMessage m)
{
     usb.ParseMessages(ref m);
     base.WndProc(ref m);
}
```

-La fonction `usb_OnDataRecieved` permet de recevoir les données envoyées par la carte électronique écrit sur 64 octets pour ensuite les sauvegarder dans un vecteur `UsbData`.

```
byte[] UsbData = newbyte[65];
privatevoid usb_OnDataRecieved(object sender,
UsbLibrary.DataRecievedEventArgs
args)
{

if (InvokeRequired)
{
   Try
          {
          Invoke(newDataRecievedEventHandler(usb_OnDataRecieved), new
          object[] { sender, args });
   }
   catch (Exception ex)
   {
          Console.WriteLine(ex.ToString());
   }
    }else UsbData = args.data;

}
```

Afin d'envoyer des données à la carte, une fonction est aussi mise en place. Les entrées de cette fonction doivent être mises sous forme d'un vecteur de type *byte*. Si la carte est détectée comme un périphérique USB par l'ordinateur alors les données seront envoyées.

```
byte[] data = newbyte[65];
if (this.usb.SpecifiedDevice != null)
{

     this.usb.SpecifiedDevice.SendData(data);

}
```

III.3.1.2. Commande

Les instructions effectuées par l'utilisateur sont introduites par un éditeur de texte. Elles doivent être écrites d'une manière bien déterminée afin qu'elles soient traitées par le traducteur mis en place.

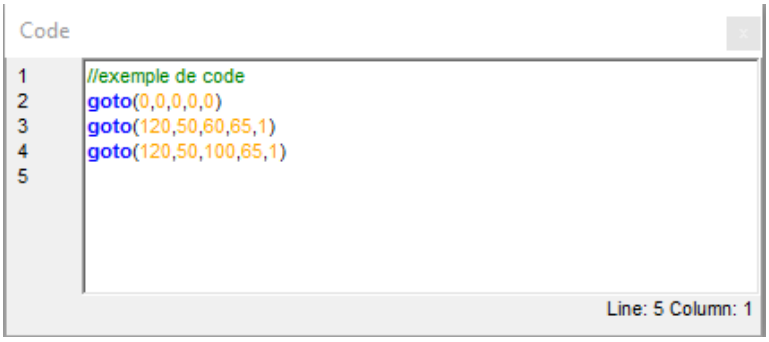


Figure III.9. L'éditeur de commandes

Le mot clé *goto* représente une fonction prédéfinie de notre application qui prend en entrée l'angle de chaque articulation du robot ainsi que l'état de l'effecteur (ouvert ou fermé). Chaque ligne est décodée et sauvegardée dans une liste nommée *GotoData*. Le décodage se fait à l'aide des expressions régulières, en mode interpréteur. Chacune d'elles permet de convertir le texte en des donnée exploitables.

```
   Regex regKeywords = newRegex(@"[^/^$^(^)]\bgoto( )*\([-]{0,1}[0-
9]{1,3}(,[-     ]{0,1}[0-9]{1,3}){4}\)", RegexOptions.IgnoreCase |
RegexOptions.Compiled);
Match regMatch;
if (Data.compiler_flag)
{
Data.GotoData.Clear();
for (regMatch = regKeywords.Match(codetext.Text); regMatch.Success;
          regMatch = regMatch.NextMatch())
{
Data.GotoData.Add(regMatch.ToString());
}

}
```

III.3.1.2. Simulateur 3D

Un simulateur 3D du bras robotique est mis en place, afin de permettre à l'utilisateur de garder un œil sur le bras. Les différentes parties du robot ont été créées séparément pour faciliter la manipulation de chacune d'entre elles.

La visualisation 3D est assurée par la bibliothèque *Opengl* (*Open Graphics Library*) et plus exactement la sous-librairie `Tao.OpenGl.dll`.

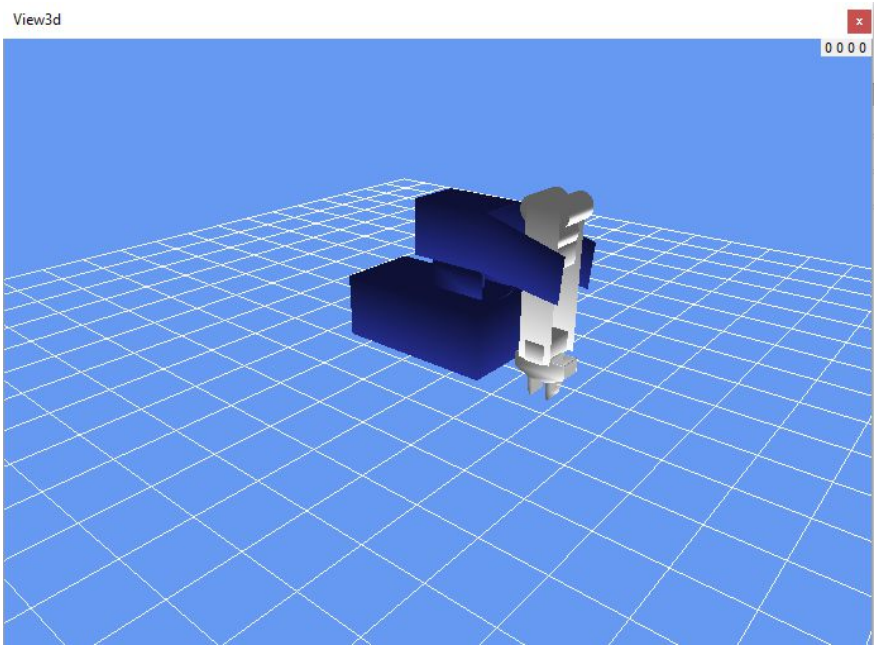


Figure III.10. Simulateur 3D

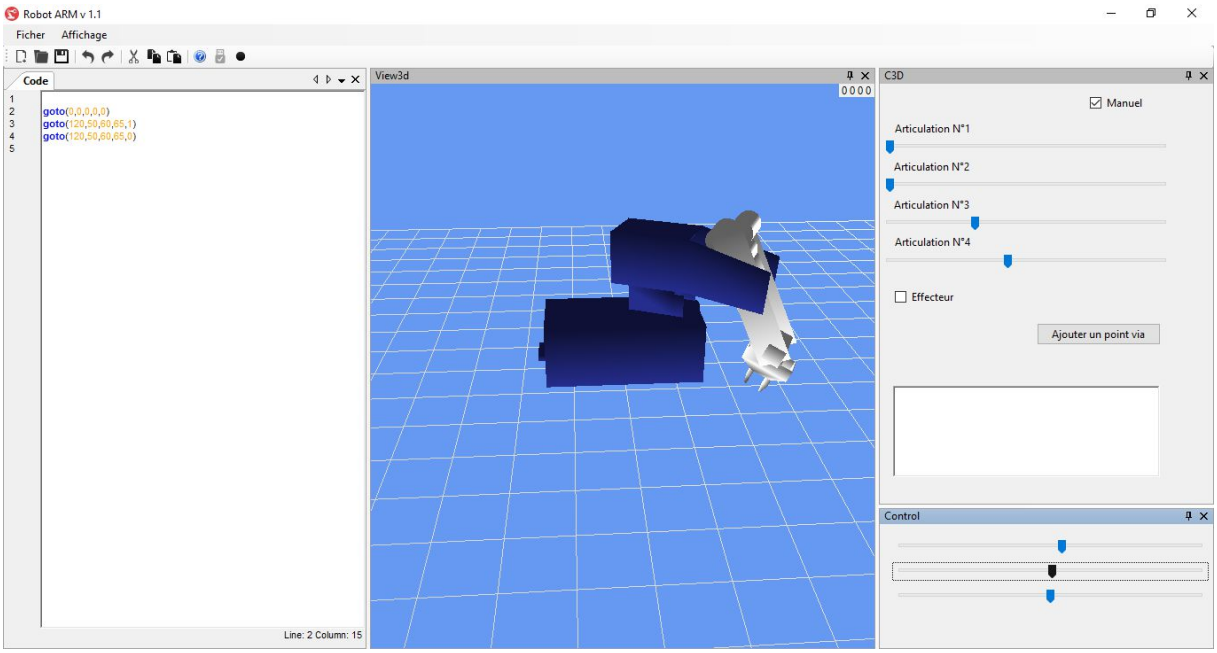
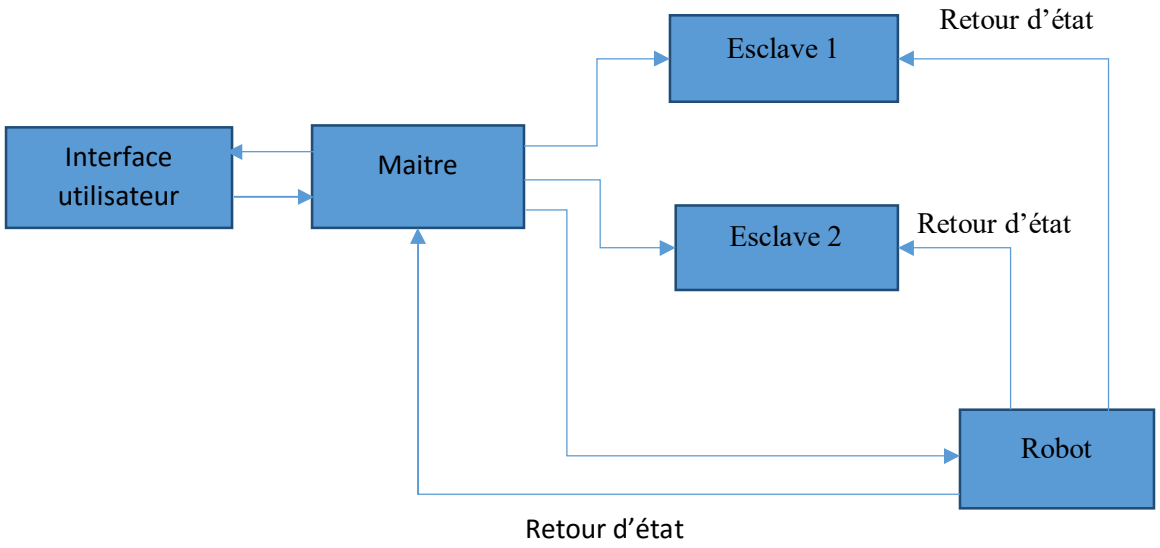


Figure III.11. Aperçu final de l'interface utilisateur

III.3.2. Programmation des microcontrôleurs

Le traitement de données envoyé par l'interface utilisateur à la carte électronique se fait au niveau des microcontrôleurs (maitre et esclaves). Les données sont directement reçues par le maitre qui les transmet aux deux esclaves pour commander le robot.

Le robot reçoit deux commandes par le maitre, deux par le premier esclave et une seule par le deuxième esclave. Ce qui fait qu'on a une commande non utilisée. Tous les états du robot sont envoyés au maitre. Mais, les esclaves reçoivent juste les états des moteurs qu'ils commandent.



III.3.2.1. Microcontrôleur Maitre

La majorité du traitement de données et des calculs se fait au niveau du maitre, l'acquisition des données concernent l'état du robot, l'envoi de données aux esclaves, génération des commandes et renvoi de données à l'interface utilisateur.

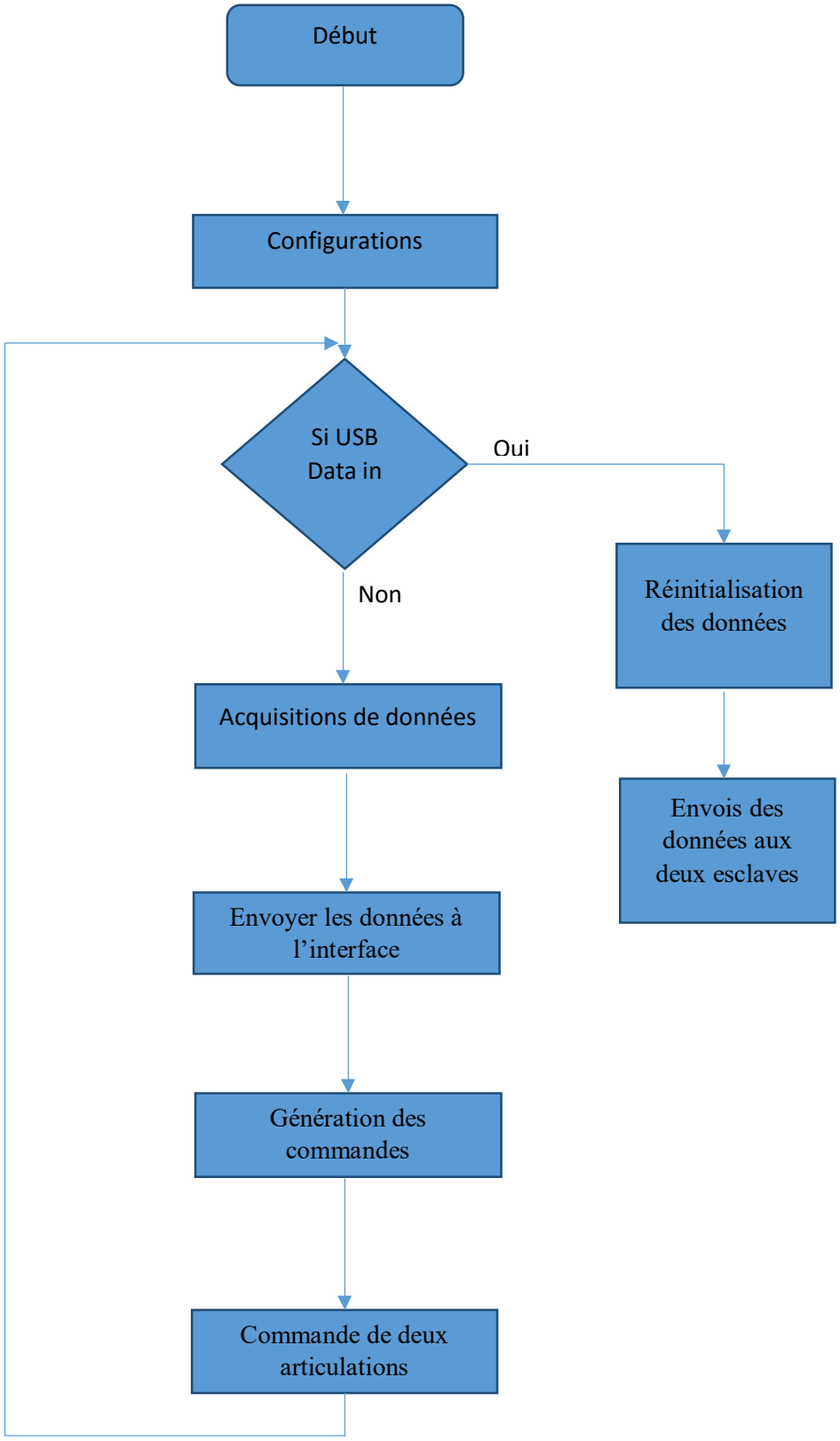


Figure III.12. Organigramme maitre

III.3.2.1.1. Configuration

La configuration est très importante. C'est là où on définit les variables à utiliser dans le programme et aussi où on initialise les périphériques et les fonctions, sachant qu'il faut bien choisir le type des variables selon le besoin, car la mémoire est très limitée.

La configuration USB

Le périphérique USB a une configuration particulière et le non-respect de cette configuration implique un disfonctionnement de ce dernier. Dans le menu *Tools* du logiciel de programmation Mikroc, on trouve un outil appelé *HID Terminal,* l'onglet *Descriptor* qui définit la configuration essentielle, à savoir : VID (vendor number), PID (product number), Vendor Name et Product Name. Le reste est configuré par défaut.

Figure III.13. USB HID terminal

Une deuxième partie de la configuration est nécessaire pour que le périphérique soit fonctionnel. Dans le menu *Project*, on ouvre *EditProject* et on reproduit la même configuration que celle de la Figure III.14.

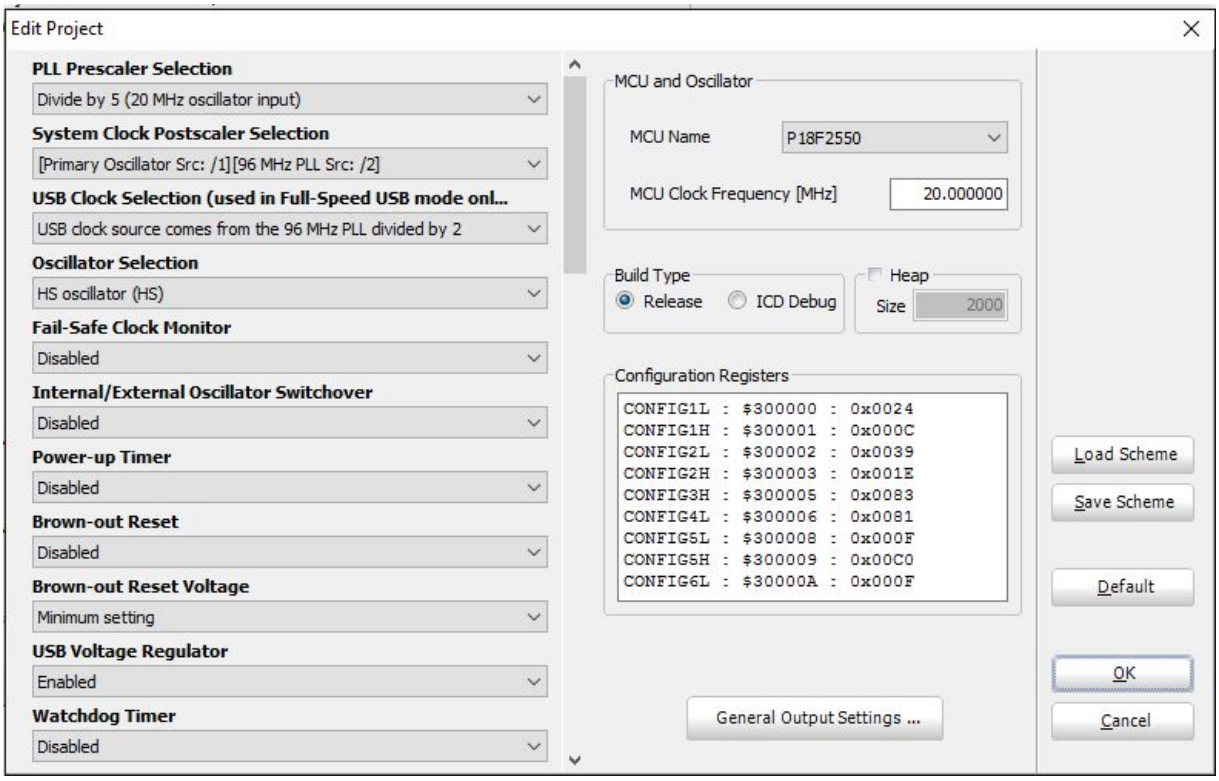


Figure III.14. Fenêtre de configuration

Deux vecteurs de 64 octets sont prévus pour s'assurer d'avoir une mémoire volatile suffisante pour effecteur le transfert de données.

```
unsigned char readbuff[64] absolute 0x500;
unsigned char writebuff[64] absolute 0x540;
```

A chaque fois que le périphérique est utilisé, une interruption du programme se fait pour permettre une synchronisation avec l'interface.

```
void interrupt(){
USB_Interrupt_Proc();
}
```

La fonction *HID_Enable* est prévue pour activer le périphérique USB et doit être utilisé dans la routine principale *main*.

```
void main(void){
HID_Enable(&readbuff,&writebuff);
}
```

Les deux fonctions *HID_Read()* et *HID_Write(&writebuff,64)* sont faites respectivement pour lire et écrire sur le bus USB

Les variables

La déclaration des variables se fait de deux façons : à l'extérieur des routines dites variables globales et à l'intérieur, on parle des variables locales. Les variables globales peuvent être utilisées par toutes les routines. Mais ce n'est pas le cas des variables locales.

On peut donner le même nom pour les variables locales sans risque de chevauchement ou d'écrasement des données.

```
char var1;        //variable global
void subroutine(){
char var2; // variable local
char var3; // variable local
var3 = var1 ;
}

void main(){
char var2; // variable local
}
```

Initialisation

Afin d'utiliser certaines fonctions, comme les fonctions prédéfinies ou bien des fonctions qu'on a créé nous-mêmes, des initialisations sont obligatoires.

- Initialisation du protocole de communication I²C : I2C1_Init(100000);
- Initialisation des Modules PWM : PWM1_Init (500000); et PWM2_Init (500000);
- Initialisation de la commande PID : Init_PID(kp, ki, kd,min,max);

III.3.2.1.2. Acquisitions et envoi de données

L'acquisition des sonnées se fait via les entrées analogiques, en utilisant une fonction prédéfinie du Mikroc. Les valeurs lues sont écrites sur 32 bit. Une division sur 4 est effectuée pour faciliter l'envoi de données.

```
analog1= ADC_Read(0)/4;
analog2= ADC_Read(1)/4;
analog3= ADC_Read(2)/4;
analog4= ADC_Read(3)/4;
writebuff[0]=analog1;
writebuff[1]=analog2;
writebuff[2]=analog3;
writebuff[3]=analog4;
```

III.3.2.1.3. Génération des commandes

La commande utilisée est la commande PID. L'organigramme suivant représente la fonction `PID_Calculate(VD, VI)`

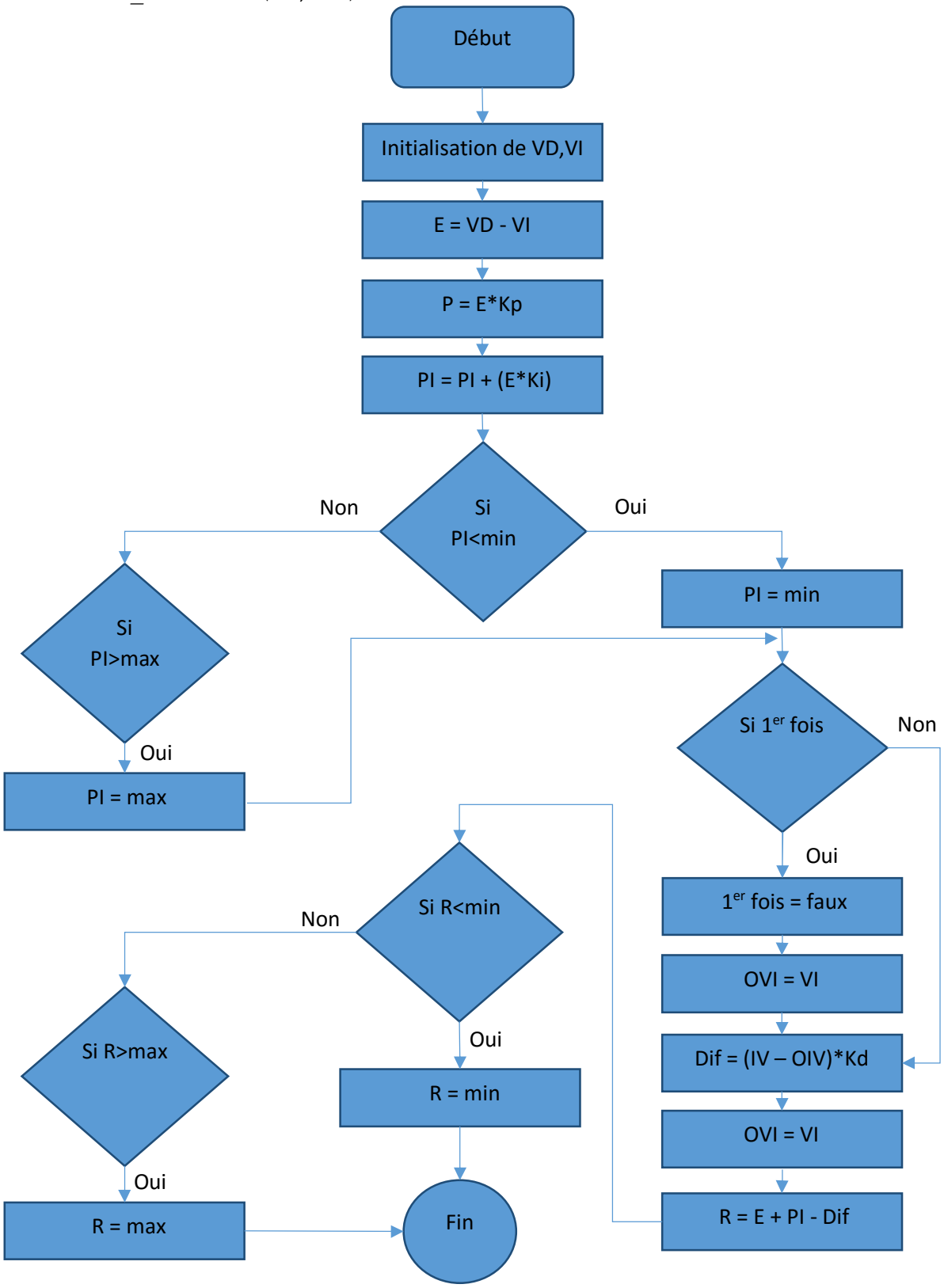


Figure III.15. Organigramme PID

Après la génération des commandes, il reste à les envoyer aux ponts H qui à leurs tours vont donner les ordres aux moteurs.

Avent de commander les ponts, il faut déclarer les branche d'activation. Cette configuration se fait au début du programme.

```
//  activation du 1er Pont
sbit IN1_M1 at RB6_bit;
sbit IN2_M1 at RB7_bit;
//  activation du 2eme Pont
sbit IN1_M2 at RB4_bit;
sbit IN2_M2 at RB5_bit;
```

Ces commandes sont écrites à l'intérieur de la routine principale.

```
// Les commande reçu via USB
VD1 =  readbuff[0];
VD2 =  readbuff[1];

Reset_PID();
PID1=PID_Calculate(VD1,analog1);
PWM1_Set_Duty(abs(PID1));

// Commande du 1er moteur
if(PID1<0){

    IN2_M1=0 ;
    IN1_M1=1;

 } else{

     IN1_M1=0 ;
     IN2_M1=1;
 }

 Reset_PID();
 PID2=PID_Calculate(VD2,analog2);
PWM2_Set_Duty(abs(PID2));
 // Commande du 2eme moteur
 if( temp2<0){

    IN2_M2=0 ;
    IN1_M2=1;

 } else{

     IN1_M2=0;
     IN2_M2=1;
 }
```

III.3.2.1.4. Réinitialisation des données et envoi aux esclaves

Une fois les données reçues une réinitialisation s'effectue pour pouvoir générer les nouvelles commandes. Une partie de ces données est envoyée aux esclaves, afin de générer à leurs tours des commandes.

L'envoi de données aux esclaves se fait via un bus I²C. Cela est assuré par la fonction *I2C_Sender* qui a comme entrée l'adresse de stockage, l'adresse de l'esclave et les données.

```
void I2C_Sender(char stockage, unsigned int address,
 unsigned int send_data)
{
    I2C1_Start();
    I2C1_Wr(slaive);
    I2C1_Wr(address);
    I2C1_Wr(send_data);
    I2C1_Stop();
    delay_ms(10);
}
```

III.3.2.2. Programmation des Pics en mode esclave

La génération des commandes se fait avec un microcontrôleur maitre entouré de deux microcontrôleurs esclaves. Il est important d'aborder la configuration permettant de définir ces microcontrôleurs en esclaves ainsi la manière de récupération des données.

```
    INTCON = 0b00000000;
    SSPCON = 0b00111110;     // Mode adresse esclave 7 bits.
    SSPCON2 = 0x00;
    SSPADD = 0xB0;           //Adresse du périphérique
    SSPSTAT=0x00;
    SSPCON.SSPEN = 1;
    SSPSTAT.SMP = 1;
    SSPSTAT.CKE = 1;
    PIR1.SSPIF = 0;
    PIR2.BCLIF=0;

    PIE1.SSPIE = 1;          // Active le module d'interruption SSP
    INTCON.PEIE = 1;         // Active les interruptions de périphériques
    INTCON.GIE = 1;          // Active les interruptions
```

Une interruption est mise en place afin de détecter si il y a transfert de données via le bus I²C.

```
void interrupt(){
    if (PIR1.SSPIF){        //Vérifiez si l'interruption est I2C
       PIR1.SSPIF = 0;
       if(!SSPSTAT.R_W & SSPSTAT.D_A & SSPSTAT.BF ){
           a++;

           if (a==1){
              address=SSPBUF;  //Sauvegarder l'adresse de stockage
           }
           if (a==2){
              rxbuffer=SSPBUF;//Sauvegarder les données reçues
              a=0;
           }
           PIR1.SSPIF = 0;
       }

       if( !SSPSTAT.R_W & !SSPSTAT.D_A & SSPSTAT.BF ){
           buff = SSPBUF;
           PIR1.SSPIF = 0;
       }
    }
}
```

On a comme sorties, les variables *address* et *rxbuffer* qui seront utilisées par la suite pour la génération des commandes.

III.4. Conclusion

Ce chapitre présente les outils informatiques utilisés, tels que les logiciels de conception de cartes électroniques, les logiciels de programmation (MS Visual Studio, MikroC, USB-PicProg) et les outils de simulation 3D. La dernière partie expose la programmation des microcontrôleurs et les différentes configurations matérielles.

Conclusion générale et perspective

Le travail réalisé dans ce mémoire de master concerne la conception et la réalisation de certaines cartes électroniques, destinées à la commande d'un bras manipulateur à 4 degrés de liberté, ainsi qu'une interface utilisateur prévue pour le contrôle et la simulation en temps réel du robot.

Une étude préliminaire effectuée sur le bras robotique utilisé, nous a permis d'avoir une certaine information sur les caractéristiques techniques de ses composants, tels que les actionneurs et les capteurs. Cette information est primordiale pour la conception et le dimensionnement des cartes réalisées.

Le travail principal consiste à réaliser la partie électronique, qui peut être utilisée même pour la commande d'autres robots, tout cela en respectant la limite en courant et en tension des composants utilisés. Pour mener à bien ce projet, des connaissances dans plusieurs domaines sont requises. Ce qui nous a poussés à perfectionner nos connaissances dans le domaine informatique, électronique et automatique.

La carte électronique répond suffisamment aux besoins exigés. Cependant, des améliorations peuvent être apportées, notamment en utilisant des calculateurs plus performants on peut aussi augmenter le nombre de sorties de commande ou ajouter des entrées pour d'éventuels capteurs. Enfin, du point de vue logiciel, on peut envisager une implémentation des algorithmes de planification de trajectoires et l'amélioration de l'interface utilisateur.

Références Bibliographiques

[1] Fiche technique du moteur F300, document de Maxonmotor, 2013.

[2] X. Fenard "*Le bus USB* ", Edition Dunod, 2001.

[3] D. Paret "*LE BUS I2C* ", Edition Dunod, 1999.

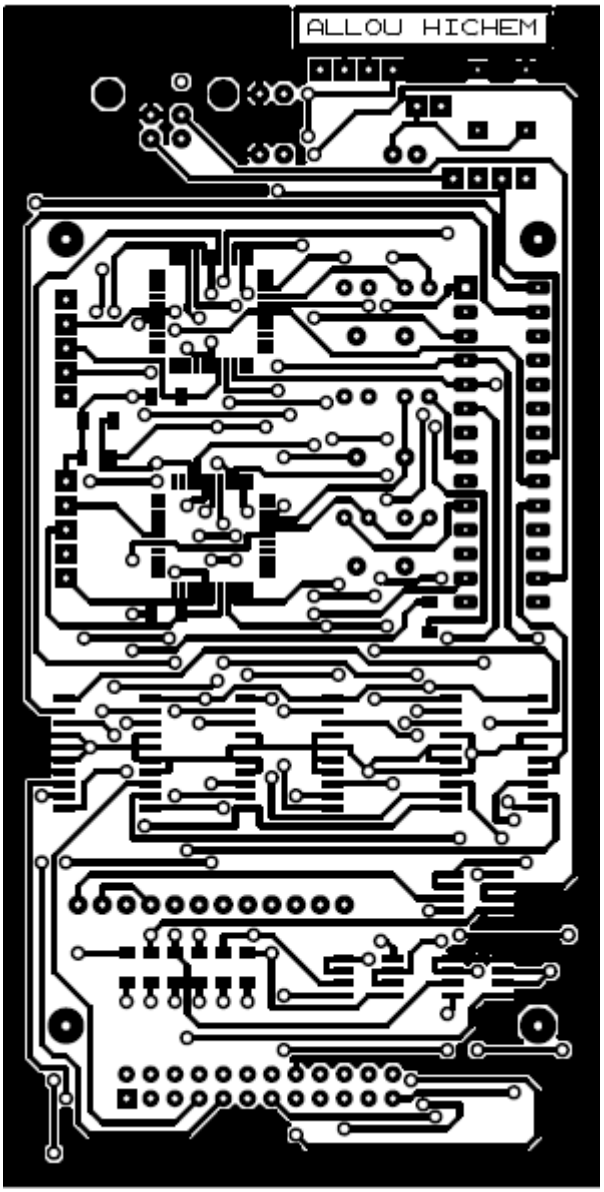
[4] P. Dondon "*Cour sur les ponts en H* ", École nationale d'électronique, Bordeaux, 2013.

[5] L. Gervais "*Apprendre la programmation Orientée Objet avec le langage C#*", Edition ENI, 2013.

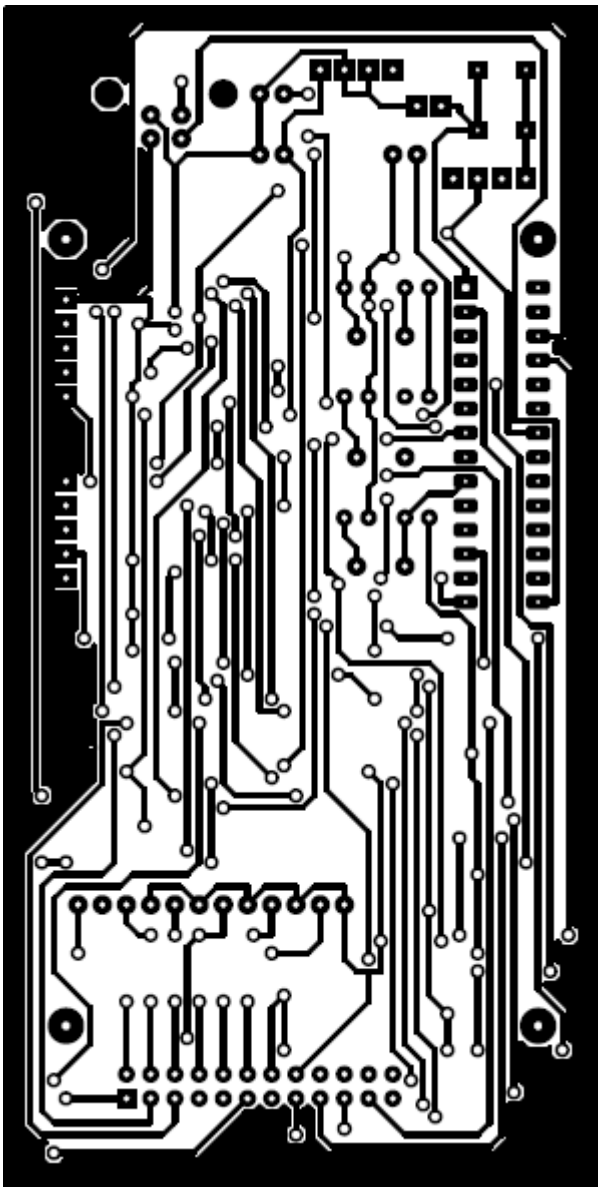
Annexe

Annexe 1

Typon de la carte de commande : couche supérieure



Typon de la carte de commande : couche inférieure



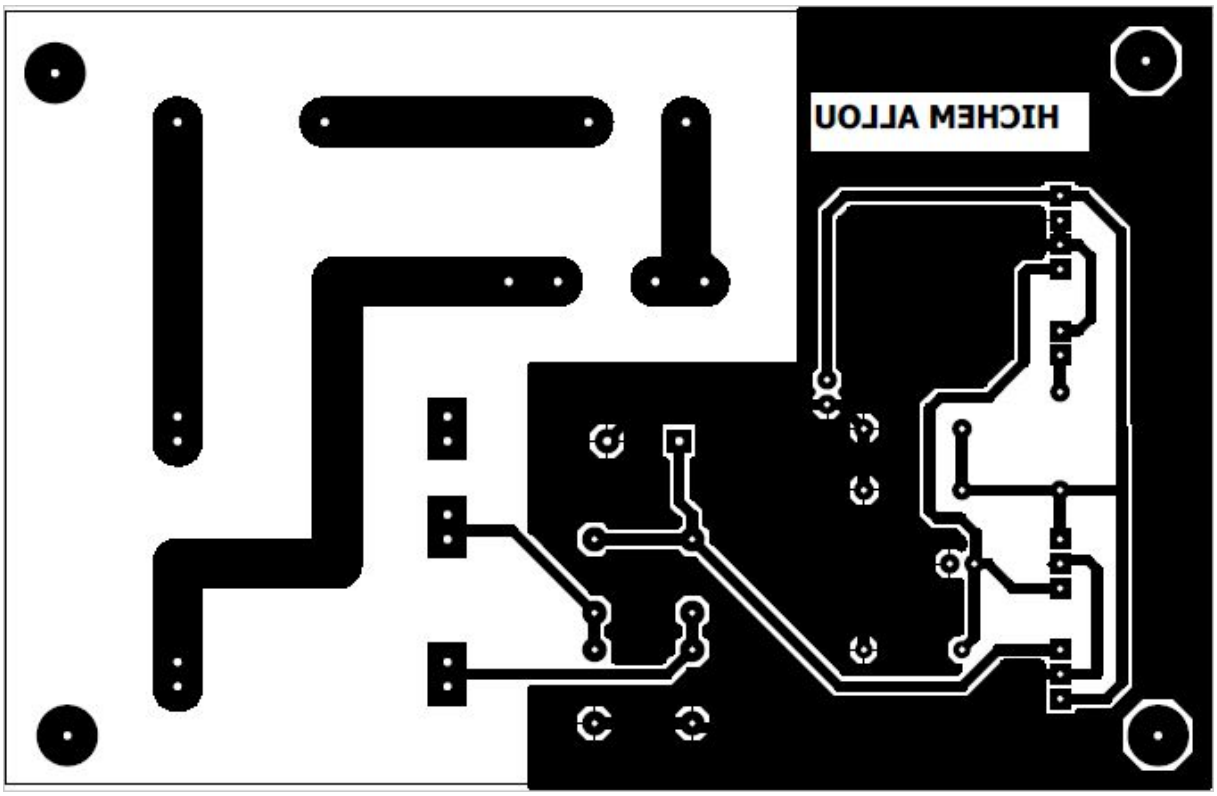
Annexe 2

Liste des composants de la carte de commande :

Référence	Quantité
PIC18f2550	1
PIC16f887	2
Quartz 20 Mhz	3
L293DD	3
Connecteur USB	1
Connecteur AWLP-24	1
Connecteur CONN-SIL4	2
Connecteur CONN-SIL5	2
Résistance 1KΩ	9
Résistance 4.7KΩ	2
Capacité 100nF	6
Capacité 100uF	2
Capacité 22uF	1
Capacité 22pF	6

Annexe 3

Typon d'étage d'alimentation



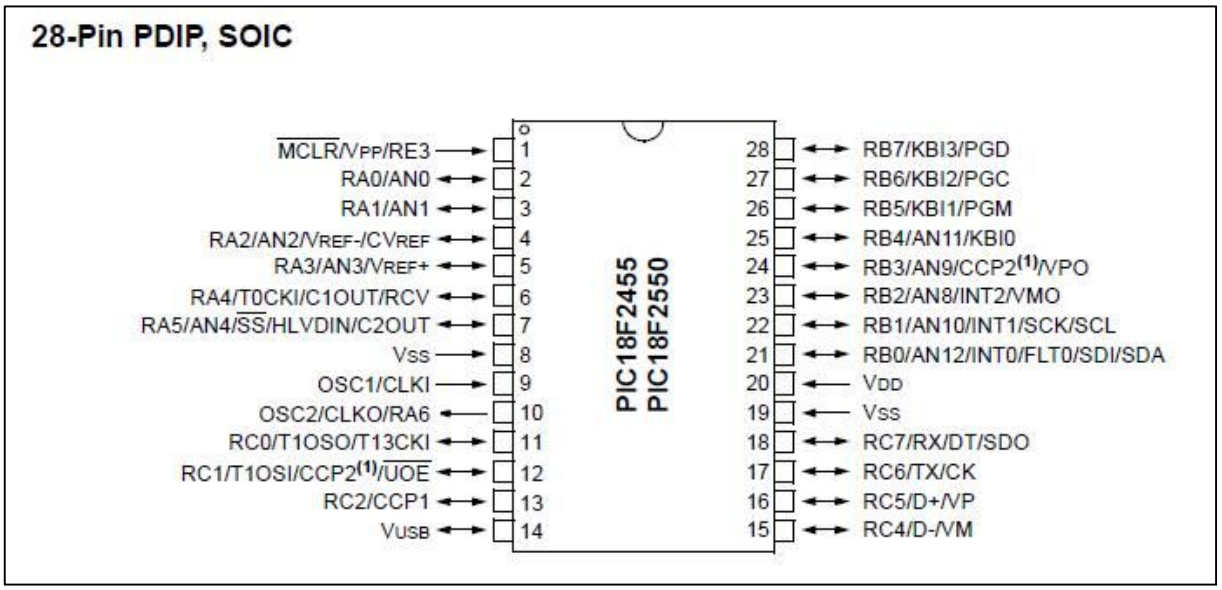
Annexe 4

Liste des composants d'étage d'alimentation :

Référence	Quantité
Transformateur 220v/12v	1
Pont de diode	1
7812	1
7805	3
Capacité 100nF	3
Capacité 10uF	2
Capacité 2200uF	1
Résistance 4.7KΩ	1
LED	1

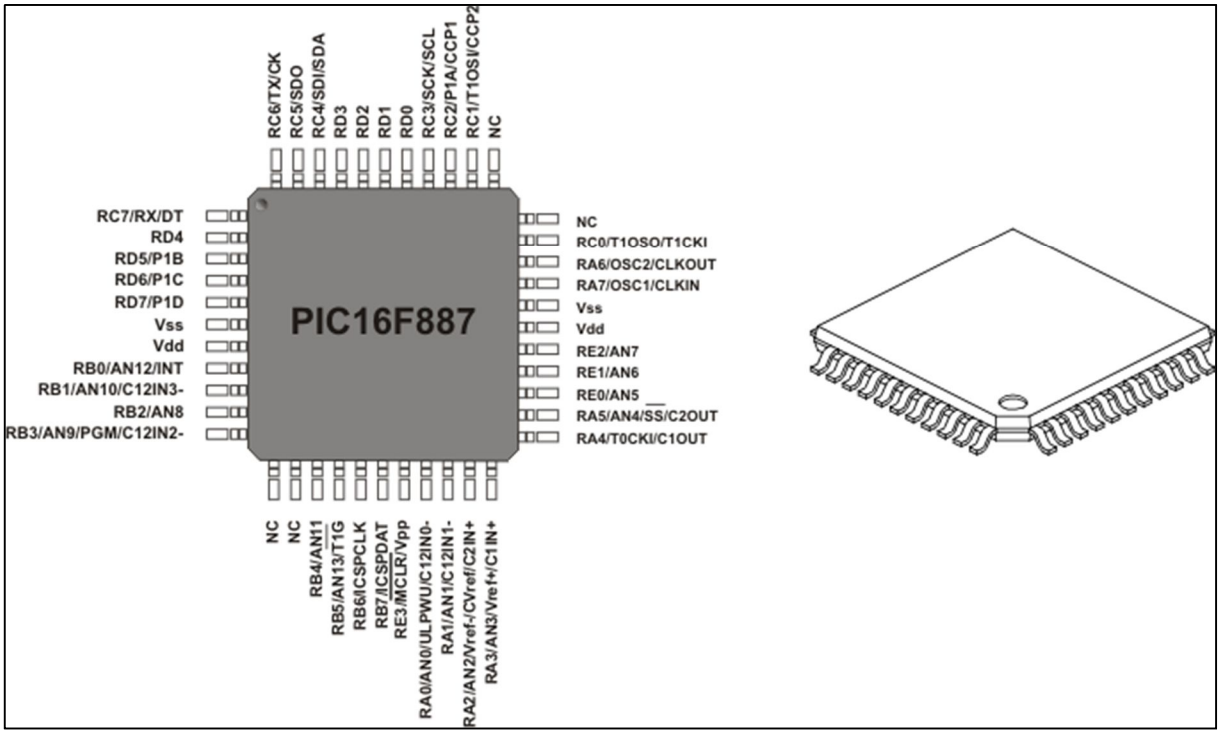
Annexe 5

Broche de PIC18f2550 :



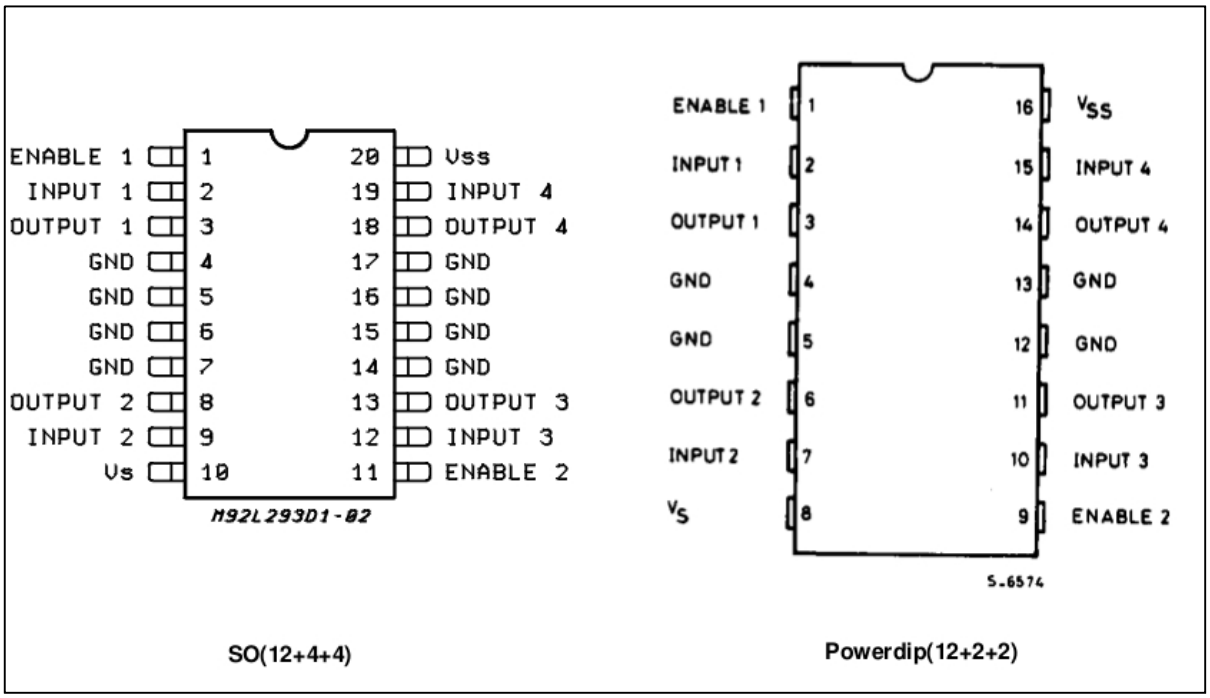
Annexe 6

Broche de PIC16f887 :



Annexe 7

Broche du L293D :



Annexe 8

Datasheet PIC18f2550

Annexe 9

Datasheet PIC16f887

Annexe 10

Datasheet L293D

Résumé :

Le travail réalisé dans ce mémoire est la conception et la réalisation d'une carte électronique pour un bras robotique. Un bras robotique a été récupéré afin de tester le bon fonctionnement de la carte, avant de commencer la conception, une étude sur la structure mécanique et les caractéristique du robot sont faite. Une étude d'une manière générale sur les composantes utilisées, notamment les périphériques des microcontrôleurs et les ponts H est très utile afin de développer les programmes. Une alimentation pour assurer les tensions de fonctionnement est mise en place. Les microcontrôleurs sont programmés en utilisant le langage C, on a conçu une application Windows pour faciliter a l'utilisateur de manipuler le robot, l'application est programmer avec le langage C#.

Mots Clés : Robotique, PIC, *MikroC*, *Protues*, *Visual studio*, C#, *Cinema 4D*.

Abstract:

The work in this dissertation is the design and implementation of an electronic card for a robotic arm. Before starting the design, a study of the mechanical structure and the characteristic of the robot are made. A generally study on the components used, including peripherals microcontrollers and H bridges is very useful in order to develop programs. A power supply to provide the operating voltages is established. Microcontrollers are programmed using the C language, a Windows application is designed to facilitate for the user to manipulate the robot, and the application is programmed with the C #language.

Keywords: Robotics, PIC, *MikroC*, *Protues*, *Visual studio*, C#, *Cinema 4D*.