République **A**lgérienne **D**émocratique et **P**opulaire
Ministère de L'**E**nseignement **S**upérieur et de la **R**echerche **S**cientifique

Université **M**ouloud **M**ammeri De **T**izi-**O**uzou

Faculté De **G**énie **E**lectrique Et D'informatique
DEPARTEMENT D'AUTOMATIQUE

Mémoire de Fin d'Etude
de MASTER PROFESSIONNEL
Spécialité : **Automatique et informatique industrielles**

Présenté par
Amar SI LOUNIS
Yazid TAHMI

Mémoire dirigé par **TOUAT M.A.** et co-dirigé par **GOUDJIL M.**

Thème

Conception et réalisation d'une solution de régulation de température à base Microcontrôleur pour un système de test de composants électroniques

Mémoire soutenu publiquement le 14 septembre 2014 devant le jury composé de :

M LAGHROUCHE M.

M TOUAT M.A.

M GOUDJIL M.

M ALLAD M.

Mme BOUDJEMAA F.

Remerciements:

Nous remercions, avant toute chose, dieu pour ses bien faits, pour sa miséricorde et pour nous avoir permis d'acquérir ce savoir et d'enrichir nos connaissances.

Nous tenons par la présente à adresser nos remerciements les plus sincères à toute personne ayant pris part de près ou de loin à notre formation, à tous ceux qui nous ont un jour donné un conseil et à tous ceux qui nous ont érudit. Particulièrement Mr. HOUCINI L., qui a été d’une aide certaine pour l’aboutissement de ce stage.

Nous tenons aussi à exprimer nos vifs remerciements à Mr. TOUAT M.A et Mr. GOUDJIL M., respectivement en qualité de promoteur et d'encadreur, pour les conseils avisés qu'ils ont su nous donner, le temps qu'ils nous ont accordé, leurs disponibilités sans faille et la rigueur dans l’accomplissement d’un travail bien fait.

Nous tenons aussi à remercier les jurés d'avoir accepté de juger ce modeste travail et de lui accorder l'attention nécessaire.

Dédicaces

Je tiens, en premier lieu, à dédier ce travail à l'intégralité de ma famille que je chéris, que dieu vous garde. Je dédie ce travail et remercie particulièrement mes chers parents, qui ont toujours su m'encourager, me motiver et me pousser à m'améliorer.

Je tiens aussi à dédier ce travail à mon ami et binôme, à tous mes amis, et à tous ceux qui m'ont soutenu.

A vous qui rythmez mon quotidien et à toute la promotion d'Automatique Professionnelle et Académique 2014.

Yazid

Dédicaces

Je dédie ce modeste travail :

- A mes parents pour leur soutien et leur encouragement tout au long de mon cursus scolaire. A mes frères et à ma grande sœur.
- A toute ma famille
- A mes amis.
- A mon binôme avec qui j'ai partagé ces quatre dernières années.
- A tous mes amis de la promotion master professionnel et académique 2014.

Amar

Liste d'indices

G_S	:	Gain statique.
θ	:	Constante de temps.
τ	:	Temps de retard.
X (t)	:	Mesure.
W(t)	:	Consigne.
Y(t)	:	Commande.
Ti	:	Temps d'intégration.
Td	:	Temps de dérivation.
Kc	:	Action proportionnelle.
W_n	:	Pulsation propre.
z	:	Facteur d'amortissement.
V_e	:	Tension d'entrée.
V_S	:	Tension de sortie.

Carte de commande (figure III.6)

- R1:10k
- R2, R3, R4, R5, R7: 100
- R6: 220
- RL1: RELAI 5V
- C1, C2, C3, C4: 1uF
- C5, C6: 22pF
- C7: 10nF
- C8: 1uF
- D1, D2: 1N4148
- J1: connecteur sil-6
- J2: connecteur dil-8
- J3: connecteur sil-2
- J4: connecteur sil-7
- J5: connecteur sil-7
- J6: connecteur sil-1
- J7: connecteur sil-1
- J9: connecteur Tblock-i2
- j_com: connecteur DB9_femelle
- j_usb : connecteur USB femelle type B
- U1:PIC18F2550
- U2:MAX232

Carte de puissance (figure III.13)

- RL1, RL2 : relai 12v double contact
- R1, R4, R8, R7: 10k
- R9, R10, R11: 220
- R2, R5: 100
- R3, R6: 3.3k
- Q1, Q3, Q5, Q6 : TIP122
- D1, D2: 1N4148

Sommaire

Sommaire

Chapitre 2 : Description du projet et des composants utilisés

Chapitre 3 : Conception et simulation

Sommaire

Chapitre 4 : Réalisation et Identification

Introduction Générale

Introduction Générale

L'essor de l'industrie et de la production de masse a poussé l'homme au développement des technologies modernes, ces dernières ont permis l'évolution des sciences tout en imposant l'exploration des domaines théoriques, dans le but d'avoir une productivité maximale. Parmi ces sciences en pleine expansion, on compte l'automatique, cette dernière fait partie des sciences de l'ingénieur, elle traite entre autres de la régulation des procédé c'est-à-dire, obtenir d'un système un comportement voulu, dans un environnement susceptible de présentant des variations paramétriques et des perturbations.

L'objectif de la régulation est de maintenir le plus près possible une grandeur commandée, ou une grandeur réglée à une valeur prédéfinie par le cahier des charges.

Notre travail se fait sous l'encadrement de l'équipe fiabilité des composants à semi-conducteur de la Division microélectronique et nanotechnologie du Centre de Développement des Technologies Avancées (CDTA). Il consiste en la conception et la réalisation d'une solution de régulation de température avec une plateforme de commande sous logiciel LabVIEW, avec une interface de commande à base PIC18f2550 destiné pour un système de teste de composants électroniques.

Le logiciel de programmation LabVIEW dispose d'un nombre important de fonctions graphiques, ce qui nous permet de programmer facilement des applications pour acquérir des données, de les traiter et d'afficher les résultats.

Pour mener à terme notre travail, nous avons organisé notre mémoire de la façon suivante :

Le premier chapitre consiste à faire des rappels sur l'identification et la régulation des systèmes et de donner un exemple sur la supervision et des méthodes de communication qu'on pourrait utiliser pour la réalisation du procédé.

En suite Le but du projet et les outils mis à notre disposition pour la réalisation du procédé finale seront énumérés dans le deuxième chapitre.

Dans le troisième chapitre, nous mettrons en lumière la stratégie de programmation et à la conception des programmes et des cartes qui vont être mis au point pour la réalisation du système de régulation.

Le dernier chapitre traitera de la réalisation, de l'identification, et de l'identification du régulateur et de ses paramètres, puis le tester dans les conditions d'utilisation.

Nous terminerons notre travail par une conclusion générale discutant des résultats obtenus et donnant une perspective pour des travaux futurs.

Chapitre 1

Généralités sur l'identification et la régulation

I. Introduction :

Il est commun dans la littérature que la modélisation d'un procédé physique par des équations différentielles représente une tâche difficile pour le concepteur, cependant nous pouvons concevoir un modèle pouvant approcher ce procédé en déterminant les paramètres de ce dernier à partir de tests expérimentaux, de façon à ce que le comportement du modèle soit équivalent à celui du procédé.

Ce modèle peut être utilisé pour la conception d'une loi de commande afin de forcer le système à avoir un fonctionnement optimal.

II. Modélisation :

Le développement d'un modèle pour un système physique peut être réalisé pour différentes raisons :

- Avoir une meilleure compréhension des phénomènes.
- Dimensionnement d'une installation.
- Formation des opérateurs.
- La mise au point de la stratégie de commande.
- Conception de la loi de commande et son réglage.
- Conception de capteurs logiciels ou estimateur d'état du système.

Les modèles utilisés en automatique sont les modèles de connaissances et les modèles de représentation, mais le modèle de connaissance est rarement utilisé car les équations physiques régissant les processus ne sont pas toujours facile à obtenir, contrairement au modèle de représentation qui est basé sur les données expérimentale des entrées/sorties. Pour ce faire nous utilisons les plusieurs méthodes d'identification afin d'élaborer ce modèle.

II.1. Différents types de modèles :

Les modèles généralement utilisés sont dynamiques, ils permettent de représenter l'évolution d'un système dans le temps, il est important d'en distinguer trois types de modèles, modèle de connaissance, modèle de comportement et modèle intermédiaire, qui ont pour objectif de décrire le système avec plus ou moins de détails.

- **Modèles de connaissance ‘’boite blanche” :**

Ils sont élaborés à partir des lois de la physique. Ils ont pour objectif d'expliquer un phénomène par une relation mathématique.

- **Modèles de comportement ‘’boite noire” :**

Ce sont des modèles linéaires, dont la validité reste limitée à de petites variations autour du point de fonctionnement. Les petites variations de l'entrée autour d'un point de fonctionnement peuvent être reliées à de petites variations de la sortie par un modèle dynamique linéaire.

- **Modèles intermédiaires ‘’boite grise” :**

Ils constituent un hybride entre les deux méthodes précédentes. On peut les considérer comme des modèles de connaissance simplifiés.

III. Identification des procèdes :

L'identification d'un système c'est la détermination de son modèle mathématique sur la base des observations expérimentales entrées/sorties.

Le traitement mathématique des réponses graphiques du système est appelé IDENTIFICATION. Le modèle obtenu est dit de conduite ou de représentation. [1]

III.1. Modèle du premier ordre :

La réponse d'un système du premier ordre soumis à un échelon, est donnée comme suit :

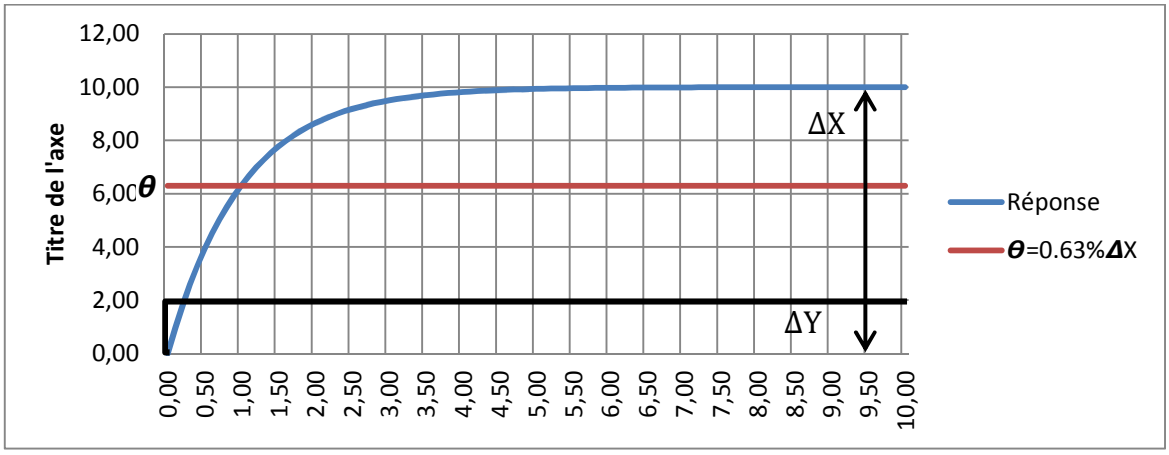


Figure I.1 : Système du premier ordre

Le modèle du premier ordre est de la forme : $$G(p) = \frac{Gs}{(1+\theta P)} \quad (1.1)$$

La détermination des paramètres de modèle se fait comme suit :

- Le gain statique est mesuré directement par : $$Gs = \frac{\Delta x}{\Delta y} \quad (1.2)$$

- La constante de temps ***θ***: on trace conjointement la droite d'ordonnée (0.63 ***Δx***) parallèle à l'axe des abscisses.

III.2. Modèle du second ordre :

La réponse d'un système du second ordre soumis à un échelon, est donnée comme suit :

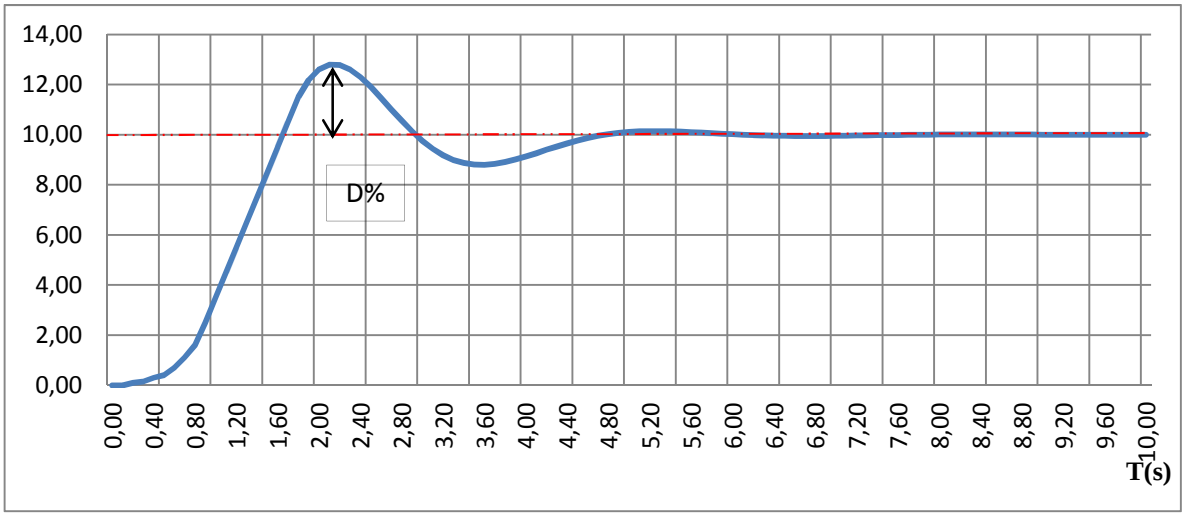


Figure I.2 Réponse d'un système du deuxième ordre à un échelon pure

Sa forme canonique est : $$G(p) = \frac{Gs}{1+\frac{2h}{W_n}P+\frac{P^2}{{W_n}^2}} \quad (1.3)$$

- Le facteur d'amortissement **h** se détermine à l'aide de la mesure du dépassement :

$$D\% = 100e^{\frac{-\pi h}{\sqrt{1-h^2}}} \quad (1.4)$$

- La pulsation propre W_n : $$Tpic = \frac{\pi}{W_n\sqrt{1-h^2}} \quad (1.5)$$

III.3. Méthode de Broïda :

Broida a estimé qu'en passant d'un ordre n a un 1er ordre que la tangente au point d'inflexion était une source d'erreur importante et que la durée des essais pouvait être longue sur les systèmes lents avec le risque d'avoir une entrée qui varie pendant l'essai. [1]

Le modèle proposé pour approcher le comportement du système est un premier ordre avec retard, sa fonction de transfert est : $G(p) = \frac{Gs.e^{-\tau p}}{(1+\theta P)}$ (1.6)

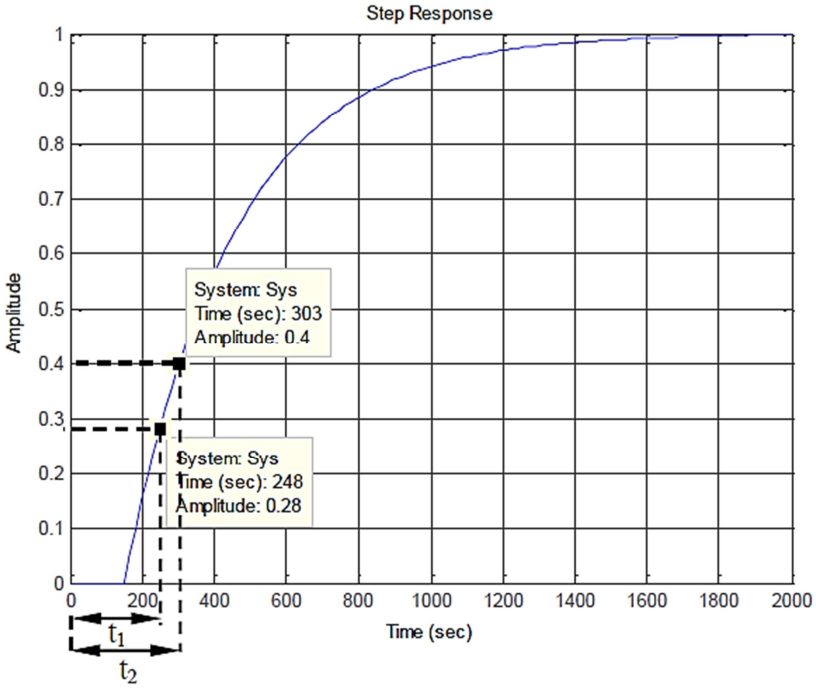
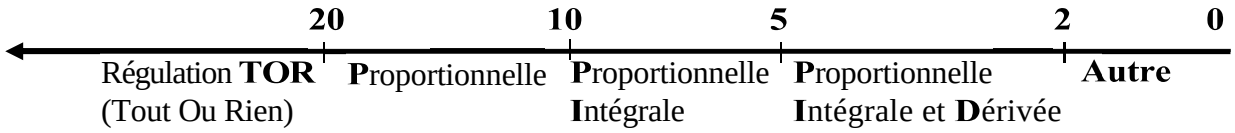


Figure I.3 : Système sans dépassement avec un point d'inflexion

Ces calculs montrent que pour obtenir les temps t_1 et t_2, on prend respectivement pour Y(t) les valeurs **0.28Δx** et **0.40Δx**

- Le gain statique Gs : $Gs = \frac{\Delta x}{\Delta y}$ (1.8)
- La constante de temps (***θ***) : $P = 5.5 * (t40\% - t28\%)$ (1.9)
- Le retard (τ) : $\tau = (2.8\, t28\% - 1.8\, t\ 40\%)$ (1.10)
- **Choix de la régulation – indice de réglabilité**

La difficulté de réguler un procédé est d'autant plus grande, pour une constante θ donnée, que le retard τ est grand. Il est donc naturel de mettre en œuvre un régulateur d'autant plus riche en action que le procédé comporte un rapport $\frac{\theta}{\tau}$ petit. Le modèle de Broïda ait été établi par calcul ou par identification expérimentale, le graphe suivant guide sur le choix de la régulation en fonction du rapport $\frac{\theta}{\tau}$



- **Réglage du régulateur :** Une fois la régulation choisie, le tableau ci-dessous conduit au réglage du régulateur à appliquer selon Broïda.

	P	PI série	PI parallèle	PID série	PID parallèle	PID mixte
Kc	$\frac{0,78}{Gs}.\frac{\theta}{\tau}$	$\frac{0,78}{Gs}.\frac{\theta}{\tau}$	$\frac{0,78}{Gs}.\frac{\theta}{\tau}$	$\frac{0,83}{Gs}.\frac{\theta}{\tau}$	$\frac{0,83}{Gs}.(\frac{\theta}{\tau}+0,4)$	$\frac{0,83}{Gs}.(\frac{\theta}{\tau}+0,4)$
Ti	sans	$\boldsymbol{\theta}$	$\frac{\tau.Gs}{0,78}$	$\boldsymbol{\theta}$	$\frac{\tau.Gs}{0,75}$	$\boldsymbol{\theta}$+0,4τ
Td	0	0	0	0,42τ	$\frac{0,35.\theta}{Gs}$	$\frac{\theta.\tau}{\tau+2.5\theta}$

Tableau I.1 : Réglage du régulaeur

III.4. Modèle d'ordre supérieur ''Méthode de Strejc" :

Cette méthode permet l'identification d'un processus dont la réponse à l'échelon n'a pas de dépassement. [1]

La méthode de Strejc consiste à caractériser le procédé par un modèle de la forme :

$$G(p) = \frac{Gs.e^{-\tau p}}{(1+\theta P)^n} \qquad (1.11)$$

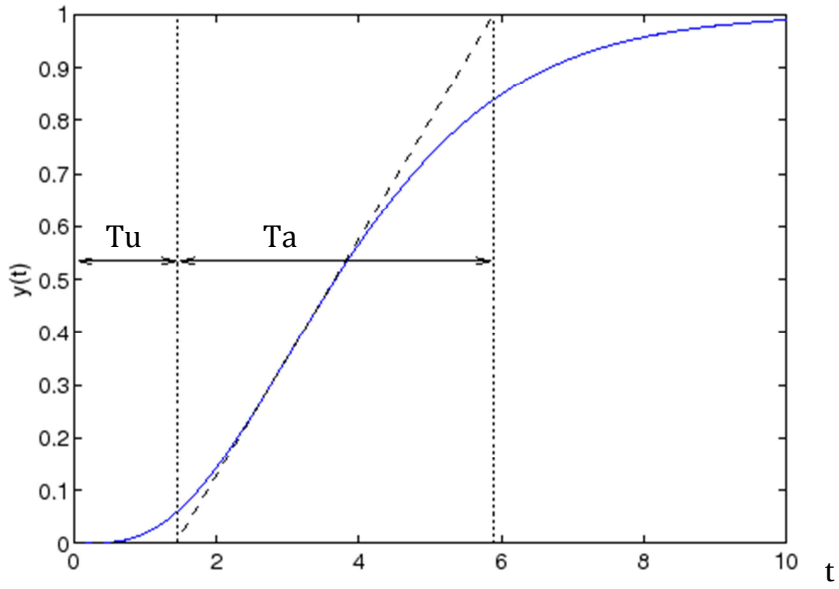


Figure 1.4 : Réponse d'un système sans dépassement

- **Le principe de la méthode de Strejc :**

Quatre éléments sont essentiels pour la détermination d'un le modèle selon Strejc

o Tracer la tangente au point d'inflexion, qui permet de définir les deux grandeurs (Tu) et (Ta), comme dans la Figure (1.4).

o On calcule le rapport η : $$\eta = \frac{\text{Tu}}{\text{Ta}} \quad (1.12)$$

o Dans le tableau (1.1) on cherche la valeur immédiatement inférieure à la valeur (η) dans la colonne Tu/Ta, en suivant la ligne correspondante, on obtient l'ordre (n) du modèle, mais aussi la 3éme colonne nous permet de calculer la constante de temps $\boldsymbol{\theta}$.

o Le retard est égale à : $\tau = Tu - \frac{Tu}{Ta}$

$\frac{Tu}{\text{Ta}}$	Ordre n	$\frac{\tau}{Ta}$
0.000	1	1.000
0.104	2	0.368
0.218	3	0.271
0.319	4	0.224
0.410	5	0.195
0.493	6	0.175
0.570	7	0.161
0.642	8	0.149

Tableau I.2 : Coefficients de la méthode de Strejc

IV. La Régulation :

Réguler une grandeur, c'est obtenir d'elle un comportement voulu. La régulation regroupe l'ensemble des techniques utilisées visant à contrôler une grandeur physique (débit, pression, température...) dans le but d'obtenir une valeur désirée (consigne), malgré les perturbations.

Pour effectuer des tâches de régulation, il est important de comprendre les relations de cause à effet entre les systèmes entrant dans la boucle de régulation. [2]

IV.1. Objectifs de la régulation :

- Stabiliser les systèmes instables.
- Corriger rapidement les perturbations influant sur le système
- Assurer l'écart le plus faible possible entre la consigne et la mesure

- Maitriser la qualité de production

IV.2. Principe de régulation :

Une boucle de régulation comprend généralement :

- Une grandeur réglée : c'est la grandeur physique que l'on désir réglée, elle donne son nom à la régulation.
- Une consigne : c'est la valeur que doit prendre la grandeur réglée.
- Une grandeur réglante : c'est la grandeur physique qui a été choisie pour le contrôle de la grandeur réglée. Généralement elle n'est pas de même nature que la grandeur réglée.
- Les perturbations : ce sont les grandeurs physiques qui influencent sur la grandeur regelée. Généralement elles ne sont pas de même nature que la grandeur réglée.
- Un organe réglant : c'est l'élément qui agit sur la grandeur réglante.

Il faut donc mesurer les principales grandeurs servant au contrôle du processus, puis récupérer et comparer les valeurs de ces grandeurs aux valeurs souhaitées (consigne). En cas de non-concordance entre ces valeurs, l'organe de régulation envoie un signal de commande à l'organe de contrôle (actionneur), afin que celui-ci agisse sur le processus. Les paramètres qui régissent le processus sont ainsi stabilisés en permanence.

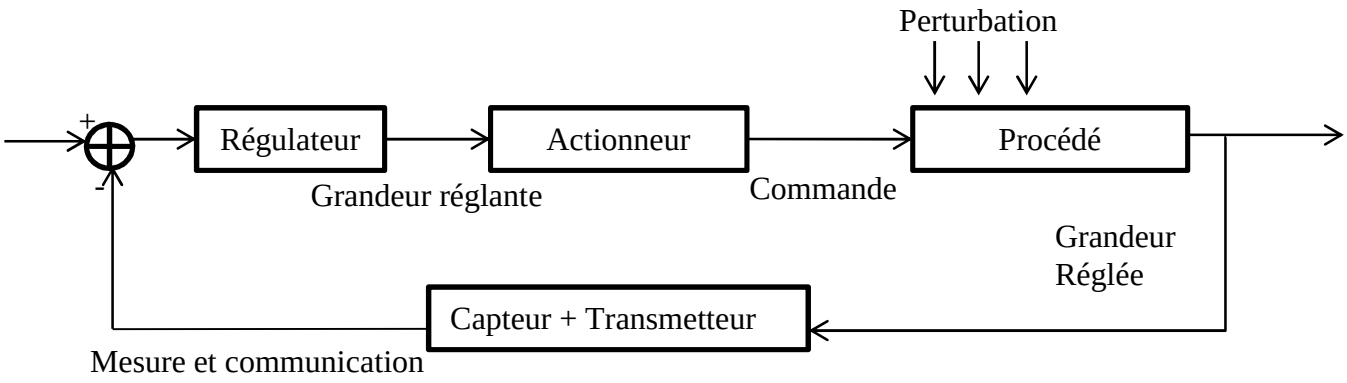


Figure 1.5 : Schéma de principe d'une régulation en boucle fermée

IV.3. Formes fondamentales de régulation :

IV.3.1 Régulation tout ou rien :

Cette régulation a deux états possibles pour la commande ; Celui qui correspond à la commande maximale (100%) et celui qui correspond à la commande minimale (0%), avec un seuil de commutation.[1]

Le réglage du régulateur se fait à l'aide de deux paramètres : La consigne W, et le seuil DIFF.

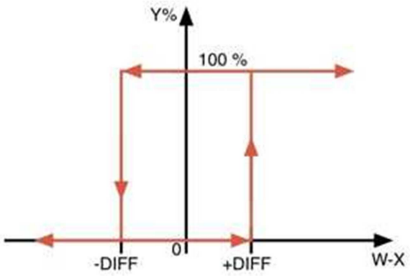


Figure I.6: Caractéristique d'un régulateur TOR

La grandeur réglée oscille autour du point de fonctionnement (figure I-7 et figure I-8). À chaque dépassement des seuils de commutation, la sortie du régulateur change d'état. Compte tenu de l'inertie du système, la valeur absolue de l'erreur |{E}| peut dépasser le seuil DIFF. Sauf exception, la mesure ne peut pas être constante dans ce type de régulation. Le système est en régime d'instabilité entretenue.

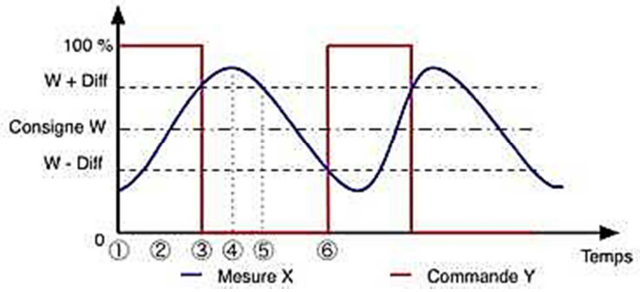


Figure I.7 : Evolution des grandeurs dans le temps

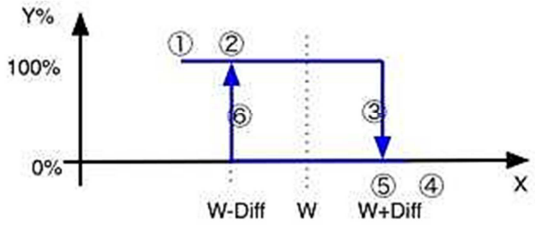


Figure I.8 : Evolution des grandeurs dans le plan XY

IV.3.2. Régulation en boucle ouverte :

En boucle ouverte, l'organe de contrôle ne réagit pas, vu qu'il n'y a pas de retour sur la mesure, donc c'est l'opérateur qui tient le rôle de l'organe de réglage, cela nous oblige à connaitre la corrélation entre la valeur mesurée et la grandeur réglante. [1]

Ce genre d'asservissement permet d'anticiper les phénomènes et d'obtenir un temps de réponse très court. Lorsqu'il n'y a pas de contrôle final possible, l'asservissement en boucle ouverte est la seul solution envisageable, Par contre, la régulation en boucle ouverte ne compense pas les facteurs perturbateurs et on doit connaitre les lois régissant le fonctionnement du processus.

Figure I.9 : Schéma fonctionnel en boucle ouverte

IV.3.3. Régulation en boucle fermée :

La régulation en boucle fermée offre beaucoup d'avantages, vu que les facteurs perturbateurs sont en général automatiquement compensés par la contre-réaction à travers le procédé, mais aussi, il n'est pas nécessaire de connaître avec précision les lois ou le comportement des différents composants de la boucle, bien que pour le choix des composants, la connaissance des allures statistiques et dynamiques des divers phénomènes rencontrés soit obligatoire, car un mauvais choix de ces composants peut amener la boucle à entrer en oscillation, du fait que le comportement dynamique de la boucle dépend de leurs caractéristiques, notamment de celles du processus, il faut citer aussi que la précision et la fidélité de la régulation dépend de la fidélité et de la précision sur les valeurs mesurées et sur la consigne. [1]

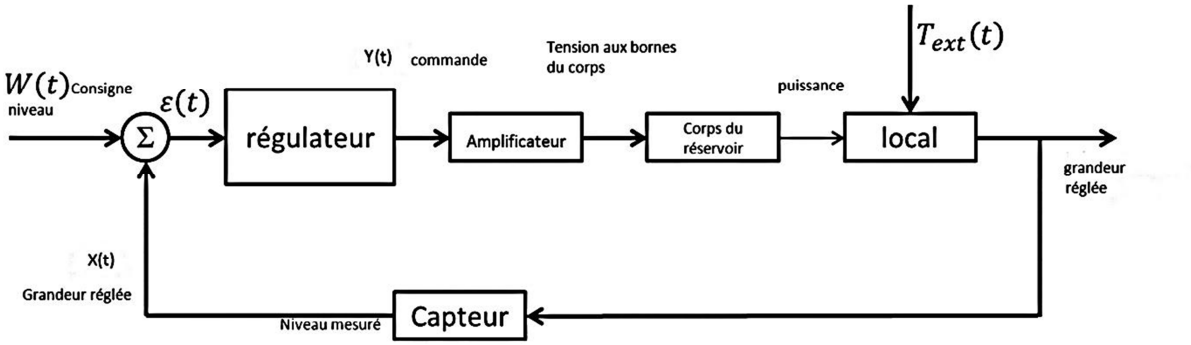


Figure 1.10 : Exemple fonctionnel en boucle fermée

IV.3.3.1. Rôle des actions dans la boucle fermée :

- **Rôle de l'action proportionnelle (P)** :

Le rôle de l'action proportionnelle est d'accélérer la réponse de la mesure, dans le but réduire l'écart entre la mesure et la consigne.

L'étude de l'action proportionnelle sur un système naturellement stable en boucle fermée, montre que lors d'un changement de consigne, le régime permanent atteint un écart résiduel.

Une augmentation du Gain Gs, accélère la réponse du procédé, provoque une diminution de l'écart résiduel, mais rend la mesure de plus en plus oscillatoire.

La valeur optimale de Gs est celle qui donne la réponse la plus rapide, avec un bon amortissement

(ne dépassant pas 15 %). L'étude de l'action proportionnelle sur un procédé instable (aussi appelé intégrateur), montre que lors d'une variation de consigne, la mesure rejoint la consigne dans tous les cas, mais en cas de perturbation, la mesure s'écarte de la consigne, la régulation proportionnelle tend à la ramener tout en laissant subsister un écart résiduel, lorsque le régime permanent est atteint. [2]

- **Rôle de l'action Intégrale (I) :**

Le rôle de l'action intégrale est d'annuler l'écart entre la mesure et la consigne. Le signal de sortie du régulateur en intégrateur seul est proportionnel à l'intégrale de l'écart mesure-consigne. L'action intégrale est généralement associée à l'action proportionnelle.

Une augmentation excessive de l'action intégrale (diminution de Ti) peut être source d'instabilité.

Le comportement de l'action intégrale sur un procédé instable, est sensiblement le même que sur un procédé stable. Il faut noter que l'action intégrale est nécessaire pour annuler l'écart, suite à des perturbations. Lors de changement de consigne, son intérêt est moindre car l'écart s'annule naturellement du fait que le procédé est lui-même intégrateur. Dans ce cas l'action intégrale donne une réponse plus rapide qu'en régulation à action proportionnelle seule. [2]

- **Rôle de l'action dérivée (D) :**

Cette action compense les effets du temps mort (retard) du procédé. Elle a un effet stabilisateur mais une valeur excessive peut entraîner l'instabilité.

Son rôle est identique quelle que soit la nature du procédé. La sortie du dérivateur est proportionnelle à la vitesse de variation de l'écart.

L'étude de l'action dérivée sur un système stable est donnée par les figures suivantes pour un test en asservissement et un autre en régulation.

Dans le cas d'un signal de mesure bruité, la dérivée amplifie le bruit, ce qui rend son utilisation impossible.

La solution à ce problème consiste, soit à filtrer le signal de mesure, soit à utiliser un module de dérivée filtrée avec un gain transitoire réglable.

Dans tous les algorithmes PID, la dérivée est filtrée, mais la valeur du filtre (gain transitoire) est rarement réglable sur les régulateurs monoblocs ; elle l'est parfois, sur les PID des systèmes numériques. [2]

IV.3.3.2. Méthodes de réglage des actions :

Avant de commencer les réglages d'une boucle de régulation, il faut s'assurer que le sens d'action du régulateur est correct, quelle que soit la méthode de réglage utilisée, les réglages ne sont adaptés qu'au point de fonctionnement.

Il existe différentes méthodes de réglage des actions d'un régulateur P.I.D. suivant le type de procédé et les contraintes de fabrication. [2]

- **Méthode par approches successives :**

Elle consiste à modifier les actions du régulateur et à observer les effets sur la mesure enregistrée, jusqu'à obtenir la réponse optimale. On règle l'action proportionnelle, puis l'action dérivée et l'intégrale.

Cette technique est simple et utilisable sur n'importe quel type de système. Néanmoins du fait de son caractère itératif, son application devient longue sur des procédés à grande inertie.

- **Méthode nécessitant l'identification du procédé :**

Il est possible de calculer rapidement les paramètres de réglage, si l'on connaît les paramètres du procédé, suite à une modélisation de sa fonction de transfert réglante, et si l'on est en possession de la structure du régulateur. Il est alors possible d'affiner ces suite à des essais, afin d'obtenir la réponse souhaitée.

Cette méthode nécessite un enregistreur à déroulement rapide. Elle est de préférence utilisée sur des procédés à grande inertie.

- **Méthode de Ziegler et Nichols :**

Elle nécessite l'observation de la réponse du procédé et la connaissance de la structure du régulateur. C'est une méthode qui permet le calcul des actions, sans la détermination des paramètres du procédé.

IV.3.4. Régulation cascade :

La régulation cascade sert à améliorer la boucle fermée simple sur les procédés à grande inertie, en diminuant les effets d'une ou plusieurs grandeurs perturbatrices qui agissent :

- soit sur la grandeur réglante.
- soit sur une autre grandeur appelée grandeur intermédiaire.

Ceci est obtenu en rajoutant une boucle rapide, ce qui conduit généralement à deux boucles fermées imbriquées, l'une interne, l'autre externe.

Sur ce type de régulation, on trouve en général deux points de mesure, deux régulateurs et un organe de réglage, et pour la mettre au point on doit suivre les étapes suivantes :

- Détermination du sens d'action des régulateurs
- Réglage de la boucle interne (régulateur asservi)
- Mise en service du régulateur asservi (passage de consigne interne en consigne externe sans a coups)
- Réglage de la boucle externe (régulateur pilote)

On distingue donc deux types de régulations cascade :

- **Cascade sur grandeur réglante :** elle est efficace uniquement sur les perturbations affectant la grandeur réglante.
- **Cascade sur la grandeur intermédiaire** : la boucle interne régule une grandeur intermédiaire de même nature que la grandeur réglée et elle est en partie soumise aux mêmes perturbations.

Pour que la cascade soit justifiée, il faut que la boucle interne soit beaucoup plus rapide que la boucle externe. [1]

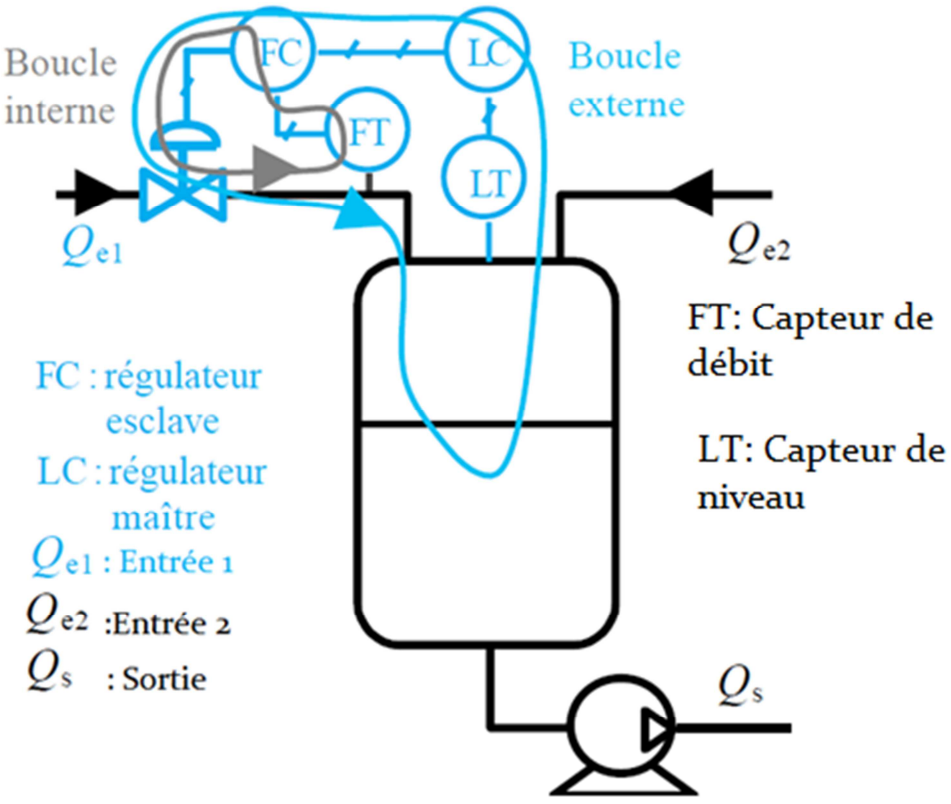
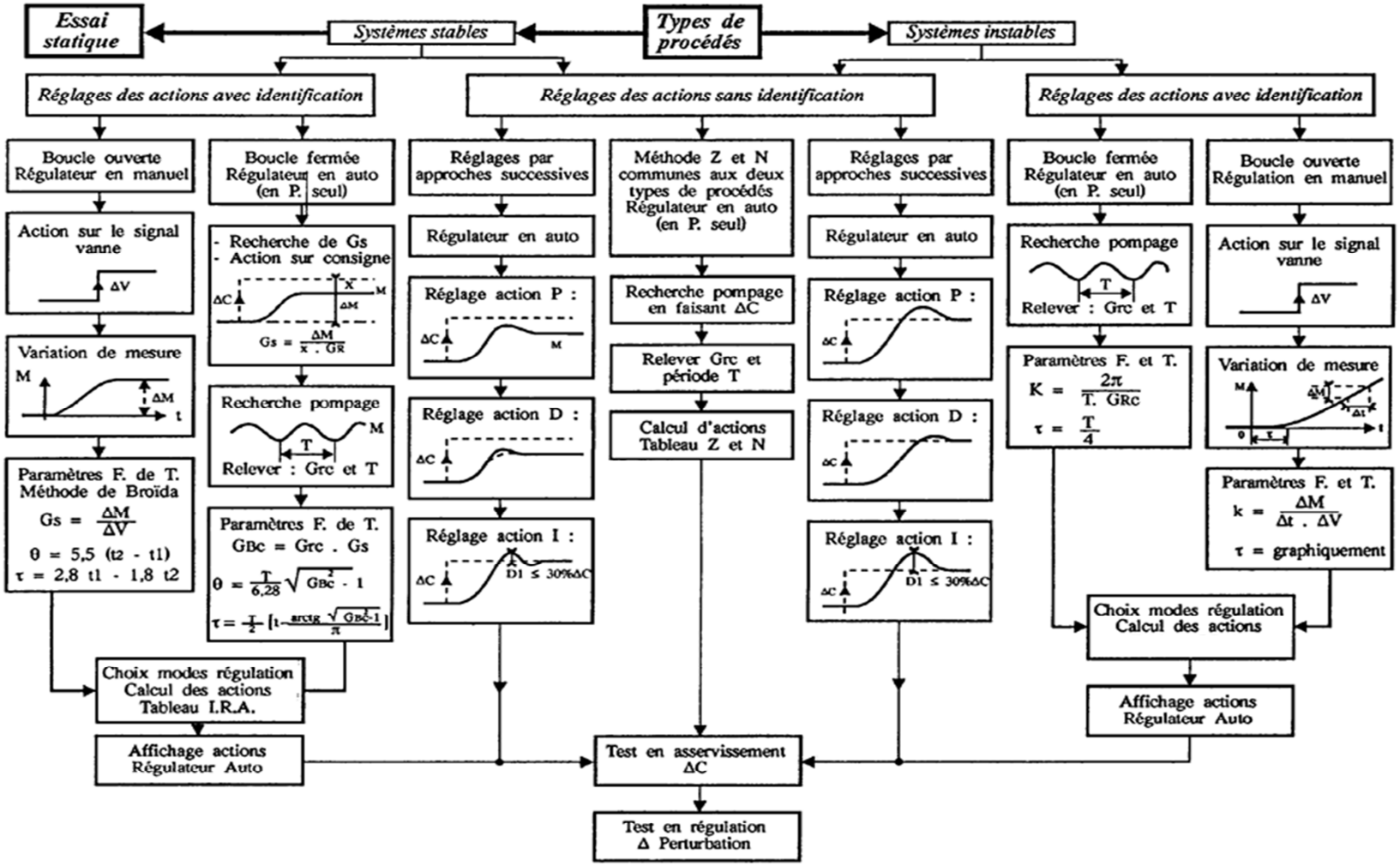


Figure I.11 : Régulation cascade

Résumé de la mise au point d'une boucle de régulation :



V. Les constituants d'un système de régulation de température :

V.1. Les capteurs :

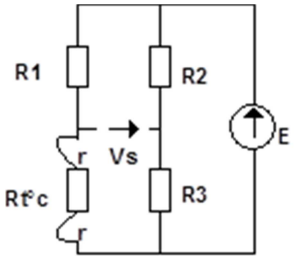
- **Les sondes résistives :**

La résistance que présentent les conducteurs électriques vis à vis d'un courant est en fonction de la température, si le rapport est prévisible, régulier et stable, ce phénomène peut être utilisé comme moyen de mesure de température. [9]

La relation approximative liant la résistance à la température est définie par :

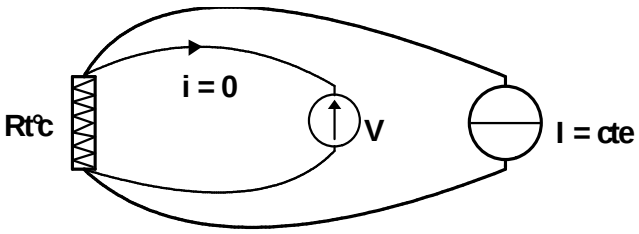
$Rt\,℃ = R0 * (1 + aT)$

Alors Vs = 0v, Donc Vs après étalonnage sera proportionnelle a à Rt°c donc à la température. Pour s'affranchir de 2.r dans l'autre branche du pont.



Autre procédé de mesure :

Il faut utiliser un générateur de courant et un voltmètre à très haute impédance d'entrée, (ie = 0) : $Rt°C = \frac{V}{I}$



V.2. L'élément chauffant :

- **La résistance blindée :**

Loi de Joule : $W = R * I^2 * t$

W : Energie thermique (J) I : Intensité du courant (A)

t : Temps de passage du courant (s) R : Valeur de la résistance (Ω)

Domaine d'activité : Chauffage domestique, Sidérurgie, Four de cuisson, ...

Présentation : Thermoplongeur, collier chauffant, résistance à ailettes, ...

Avantages : Rendement voisin de 1, Facilités de mise en œuvre et de régulation, Aucune pollution.

- **Les Effets Thermoélectriques (L'effet Peltier) :**

L'effet Peltier est un effet thermoélectrique découvert en 1834 par le physicien Jean-Charles Peltier. Il décrit le phénomène de déplacement de chaleur en présence d'un courant électrique à la jonction de deux métaux différents. En effet, lorsqu'un courant électrique passe à travers la jonction reliant deux conducteurs, on observe un dégagement de chaleur ou une absorption de chaleur selon le sens du courant.

Les modules Peltier sont nommés ainsi car ils mettent en œuvre l'effet Peltier. Ce module est alimenté par un courant et présente deux faces, l'une dite froide et l'autre chaude. L'objet à refroidir doit se mettre sur la face froide, tandis qu'il est nécessaire d'avoir un mécanisme d'évacuation de la chaleur de l'autre côté (Ventilateur...). [4]

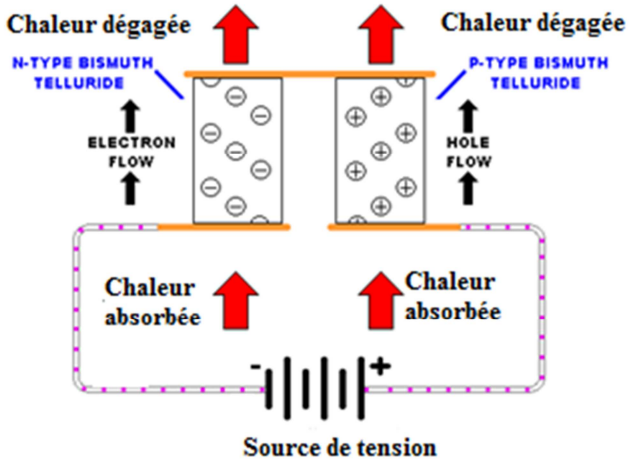


Figure I.12 : fonctionnement d'un Peltier

V.2.1 Détermination de la valeur ohmique de l'élément chauffant :

L'énergie thermique dissipée dépend de l'élément à chauffer et des contraintes de température.

$$W = M * c * (\theta 2 - \theta 1) = V * \rho * c * (\theta 2 - \theta 1)$$

W : Energie thermique en Joule

M : Masse à chauffer en kg

c : Capacité thermique massique en J / kg . °c

V : Volume à chauffer en dm^3

ρ : Masse volumique en kg / dm^3

θ2 : Température finale

θ1 : Température initiale

La puissance à installer est fonction de l'énergie thermique et du temps de chauffe : $P = \frac{W}{t}$

On impose la tension efficace U aux bornes de la résistance : $P = \frac{U^2}{R}$

- **Calcul de la puissance à installer :**

Exemple :

La puissance à fournir pour chauffer un volume V de liquide dans un temps T désiré est donnée par la relation : $P = \frac{V*\rho*c*(\theta_2-\theta_1)}{3600*T}$

P : Puissance en KW. V : Volume à chauffer en litre ou dm^3

ρ : Masse volumique kg/dm^3 c : Capacité thermique massique J/kg.°c.

T : Temps de chauffe en heures. θ_1 : Température initiale.

θ_2 : Température finale.

- **Influence des déperditions de chaleur.**

Le volume étant porté à la température désirée, il faut tenir compte ensuite des déperditions de chaleur.

Ces pertes thermiques dépendent du type de calorifugeage de la cuve et du temps durant lequel le volume de liquide doit être maintenu à la température désirée. $P1 = K * P$

VI. Simulation en situation réelle:

La simulation en situation réelle ou HIL « Hardware in the loop » consiste à tester un procédé final dans un environnement aussi proche que possible de l'environnement réel.

Le produit à tester est ainsi placé dans une boucle matérielle liant les commandes aux contrôles et simulant les liens et interactions qui existent physiquement entre eux.

Cette approche a pour objectif de palier aux inconvénients de l'approche mathématique sans pour autant passer au lancement de l'utilisation finale du procédé, Le processus à contrôler (actionneurs, système physique, et ses capteurs) peut être composé soit d'éléments simulés, soit d'éléments réels. En général, les actionneurs sont réels, et le système physique ainsi que les capteurs sont simulés.

Cela s'explique par le fait que les actionneurs et système de contrôle font, la plus part du temps, partie du même sous-ensemble, ou bien, que les actionneurs sont difficilement « simulable » en temps-réel. [10]

Exemple :

Pour un système de régulation de température, la loi de commande implémentée sur l'organe de régulation ne peut pas prendre en compte le temps de propagation de la chaleur dans l'enceint où l'action se fait, alors qu'en théorie, les tests unitaires effectués sur la carte de commande et l'algorithme de régulation aurait dû parfaitement fonctionner.

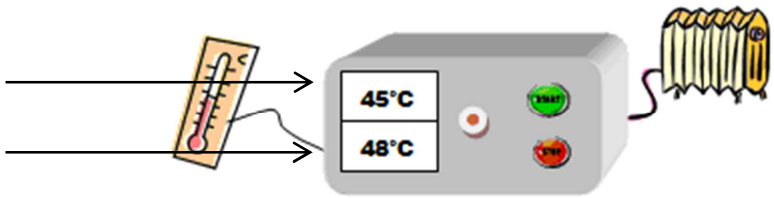


Figure I.13 : Exemple d'un système en simulation

La simulation en HILS intervient dans ce contexte, le système étant toujours sous surveillance, l'utilisateur pourra toujours agir sur le système pour pallier.

VII. Type de transmission :

Toute régulation ou traitement de donnée qui soit, doit être transmis, il y a des manières et des protocoles différents selon les besoins, on peut citer la transmission parallèle, série, sans file…

En industrie, les plus utilisés sont la transmission RS232, USB, PROFIBUS/PROFINET, et TCP/IP, dans notre projet on a utilisé la transmission USB et RS232

VII.1. E/S SERIE ASYNCHRONE:

La liaison série asynchrone est couramment utilisé en générale pour la commander de procédés pour traiter tout problème de transmission.

Pour transporter l'information, on utilise la tension (RS 232, RS422-liaison multipoint, RS485-liaison multipoint bidirectionnelle) ou le courant (boucle de courant 20mA) selon la norme EIA 232.

VII.1.1. Norme RS232 :

La norme RS232 définit des caractéristiques électriques, mécaniques et fonctionnelles. Les valeurs des tensions que les interfaces séries doivent fournir aux matériels connectés sont les suivantes (logique négative) :

- Un 0 logique est reconnu pour une tension allant de +3 à +25V.
- Un 1 logique est reconnu pour une tension allant de -3 à -25V.

Généralement, les signaux envoyés sont compris entre -15 et + 15 V (tensions d'alimentation des A-Op). Sur une liaison série au repos, on doit observer un 1 logique (voir schéma ci-dessous) :

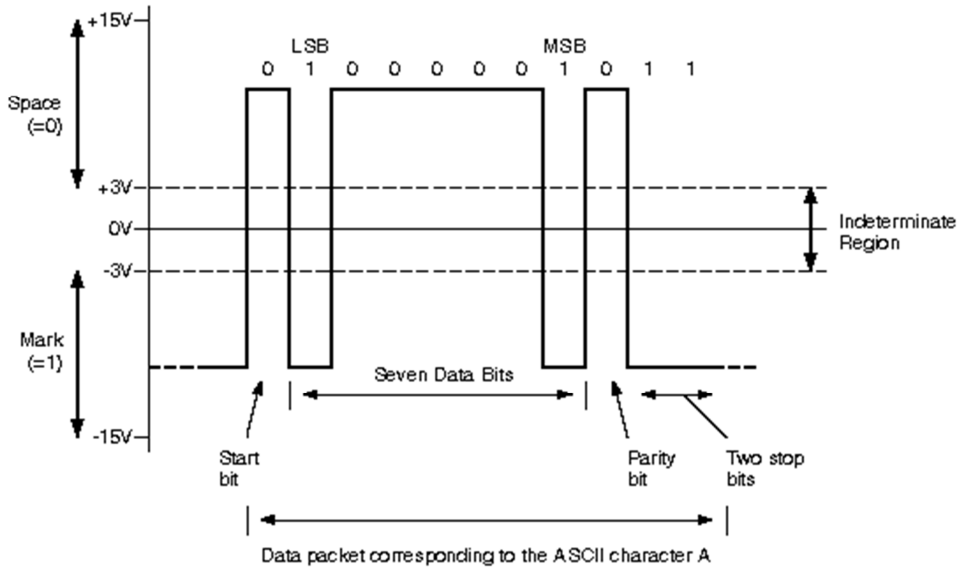


Figure I.14 : Exemple de données envoyées par port RS232

VII.1.2. Brochage du port série standard :

Le port série permet de connecter un grand nombre de périphériques tels qu'un modem, une imprimante, un autre ordinateur ou une carte d'acquisition de données.

La plupart des périphériques nécessitant une connexion bilatérale pour communiquer avec un PC utilisent un port série standard RS232.

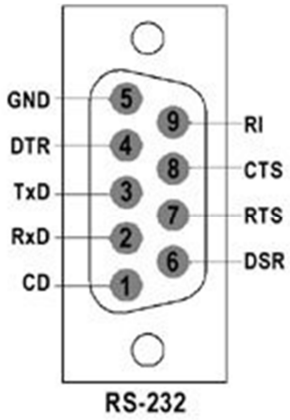


Figure I.15 : Forme du port RS232 DB9

VII.1.3. Description des signaux :

Numéro	Nom	Désignation
1	CDC - Carrier Detect	Détection de porteuse
2	RXD - Receive Data	Réception de données
3	TXD - Transmit Data	Transmission de données
4	DTR - Data Terminal Ready	Terminal prêt
5	GND - Signal Ground	Masse logique
6	DSR - Data Set Ready	Données prêtes
7	RTS - Request To Send	Demande d'émission
8	CTS - Clear To Send	Prêt à émettre
9	RI - Ring Indicator	Indicateur de sonnerie
10	Shield	Blindage

Tableau I.3 : Signaux utilisés par le RS232

VII.1.4. Modalités de transmission (simplex-duplex) :

Pour réaliser une interface, il faut 2 éléments : un DTE (Data Terminal Equipment, exemple PC, écrans) et un DCE (Data Communication Equipment, exemple modem). La transmission entre le DTE et le DCE peut se faire de plusieurs manières :

- Dans une seule direction (DTE vers DCE ou DCE vers DTE). On parle de liaison simplex (voir figure ci-dessous). Dans ce cas, 2 fils suffisent : l'un pour transporter les données, l'autre comme référence (la masse en général).

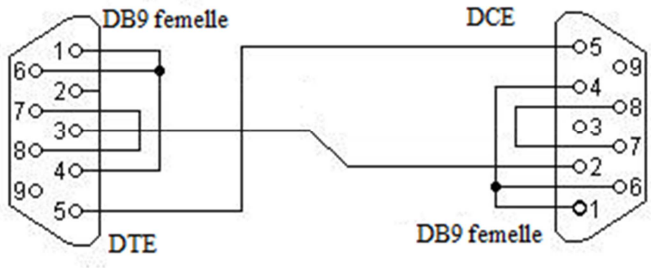


Figure I.16 : Liaison simplex

- En mode bidirectionnel : les dispositifs DCE et DTE doivent pouvoir émettre et recevoir des données. On parle de liaison duplex (voir figure ci-dessous).Si la communication est alternée : half-duplex, simultanée : full-duplex.

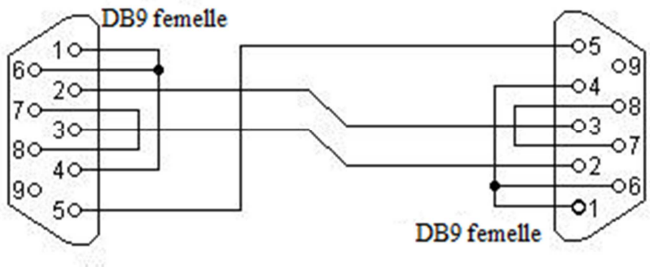


Figure I.17 : Liaison full duplex DB9

VII.1.5. Différents protocoles

Lorsqu'un DTE émet plus de données que ne peut accepter un DCE ou pour déterminer si le DTE ou si le DCE est en service, on utilise les signaux de protocole d'accord «Handshake».

Les signaux correspondants permettent de surveiller l'état d'un autre et de répondre en conséquence.

Ils indiquent la façon par laquelle le flot des données passant dans l'interface est régulé et commandé.

Selon la nature (DTE ou DCE) des appareils connectés, le type de connexion et le protocole d'échange pourra être différent. Il existe 2 grandes familles de protocole d'accord :

- Matériel : au niveau physique (fils, tensions). Utilisable seulement si les appareils concernés peuvent être connectés par câbles. DTR-DSR et RTS-CTS.
- Logiciel : au niveau du contenu des données. Ces dernières contiennent des caractères spéciaux (de contrôle). Xon-Xoff.

VII.1.6. Liaison DTE-DTE sans contrôle de flux

Pour connecter 2 DTE entre eux par une liaison série RS232, il faut au minimum 3 fils :

- Un pour les données qui circulent dans un sens.
- Un pour les données qui circulent dans l'autre sens.
- Un pour la masse électrique des signaux.

Si on veut initier une liaison full-duplex, l'émission de l'un doit correspondre à la réception de l'autre. On doit donc **croiser** les connexions (soit directement sur l'équipement, soit sur le câble). Cette liaison à 3 fils est minimum, (ce qui est utilisé dans notre projet)

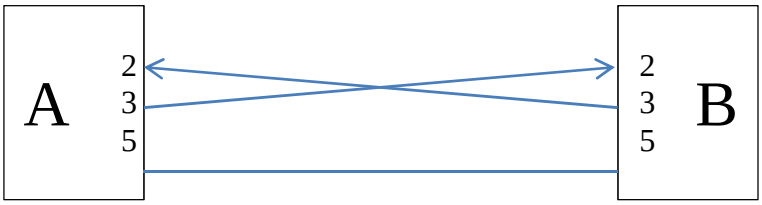


Figure I.18 : Schéma bloque transmission RS-232 DB9

En effet personne ne vérifie si son correspondant est prêt à émettre, prêt à recevoir ou tout simplement sous tension, ce qui exige l'application d'un protocole de synchronisation par programme. Exemple :

- Retour chariot
- Nombre de caractères transmis
- Transmission en code ASCII
- Utilisation d'une interruption de réception

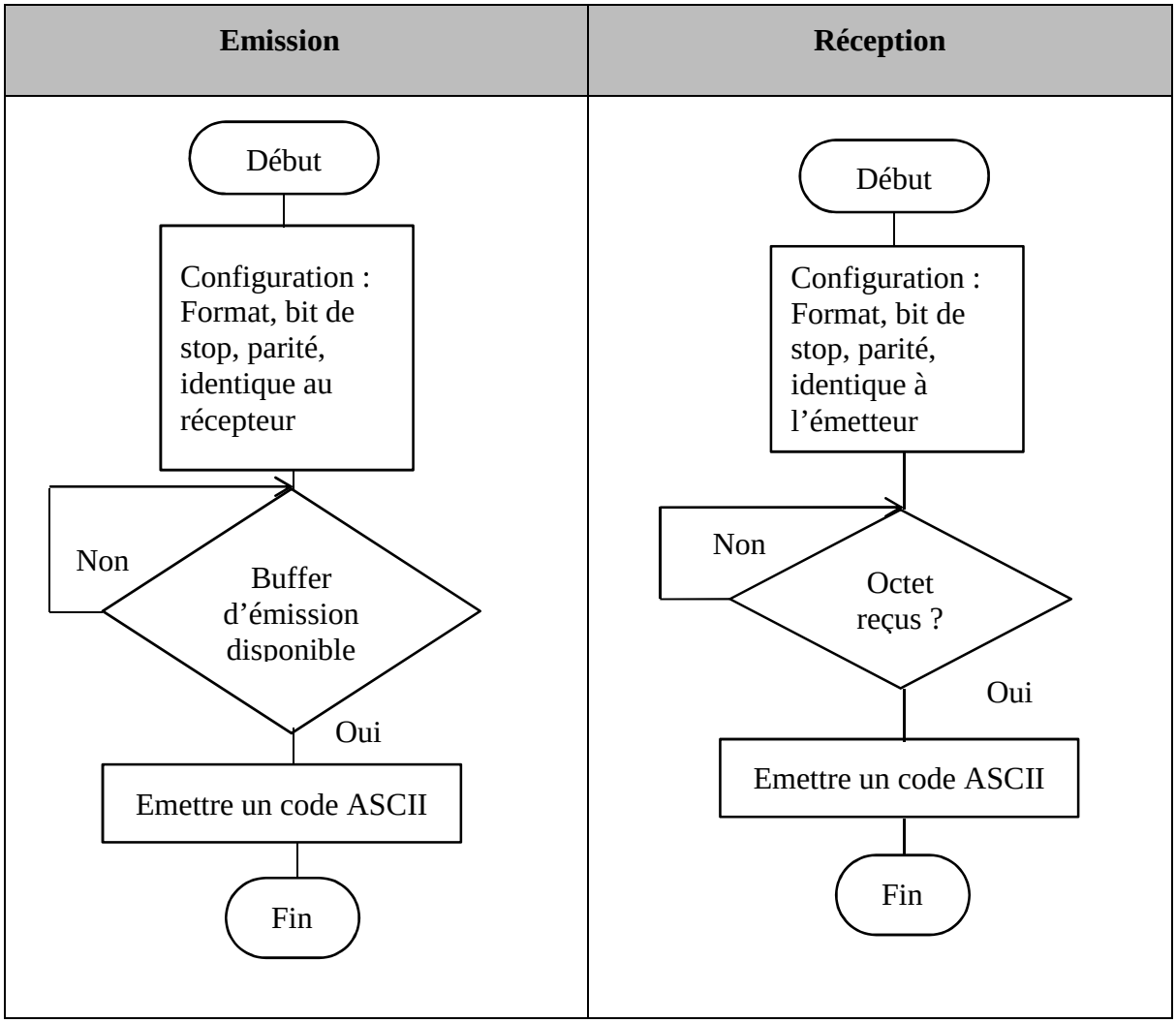


Figure I.19: organigramme de communication RS232

VII.2. USB :

La Vulgarisation d'appareils numériques grand public nécessitant des taux de transfert soutenus, a obligé les constructeurs à développer des interfaces plus rapides, plus universelles tout en pouvant y raccorder un maximum de périphériques.

Les ports série / parallèle, avec leurs caractéristiques désuètes, voient leur utilité décroître au profit de récentes interfaces comme l'USB (**U**niversal **S**erial **B**us), le FireWire (domaine grand public) ou d'autres plus anciennes mais qui ne cessent d'être améliorée comme le SCSI (domaine professionnel).

La première version de l´Universal Serial Bus est l´interface USB 1.1, elle permet de connecter à un micro-ordinateur jusqu´à 127 (adressage sur 7 bits) périphériques pour un débit maximal de 12 Mbits/s, (soit 1.5 Mo/s).

L´USB 2.0 (12/2000) s´enrichit d´une troisième vitesse de connexion entre le Host (le « Maître ») et les périphériques USB 2.0 : il est maintenant possible de relier des disques durs externes, imprimantes, scanners et autres lecteurs à des vitesses frôlant les 480 Mbits/s (soit 60 Mo/s). Le gain en vitesse a été notamment possible grâce à la réduction du voltage des signaux transmis dans les câbles, ceux-ci passant de 3.3V à 0.4V. [3]

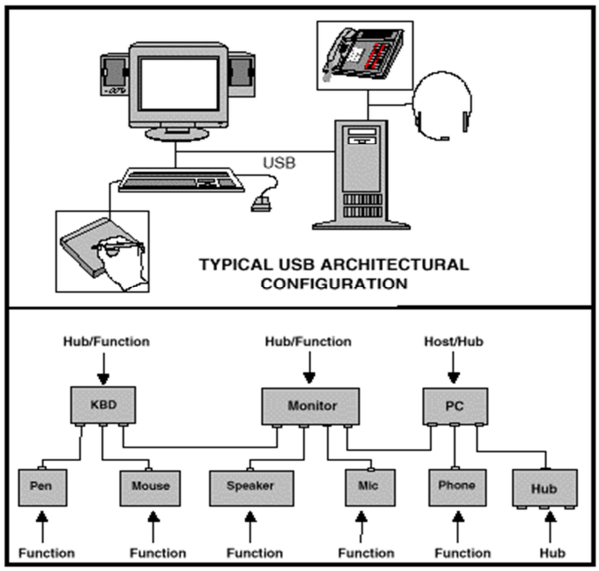


Figure I.20 : Topologie physique de connexions USB

Les principaux critères du développement de ce bus ont été les suivants :

- 3 débits cohabitent faible, pleine et haute vitesse (Low Speed 1.5 Mbit/s, Full (Medium) Speed 12 Mbit/s et High Speed 480 Mbit/s pour USB 2.0)

- Facilité d'emploi et d'utilisation : hot plug&play (détection du changement de la tension entre D+ et D-), hot swapping.
- Une seule IRQ pour tous les périphériques connectés au port (fin aux problèmes de conflits).
- Energie (faible : 100 mA) distribuée par le câble (moins de prises).
- Bas-prix pour les débits « faibles » (12 Mbit/s = 1.5Mo/s).
- Transmission en temps réel (voix, audio, vidéo comprimée).
- Protocole souple pour différents modes :
    - Transmission de paquets.
    - Mode isochrone.
    - Messages courts.
    - Séparation des commandes et des données.
- Adaptation à différentes configuration sur le terrain.
- Adaptation aisée à de nouveaux types de périphériques (adressage dynamique).
- Support architecture CTI (Computer Telephony Integration).
- Topologie en étoile (via des Hubs) ou en bus (chainés les uns aux autres).
- Nombre maximum de périphériques élevé (127). Récemment, carte multi-ports gérée par 2 ou + contrôleurs.
- principalement half-duplex. [3]

VII.2.1. Câbles

Le bus USB nécessite une connectivité composée de connecteurs de deux types, Type A disposés aux entrées du contrôleur maître et des HUB.

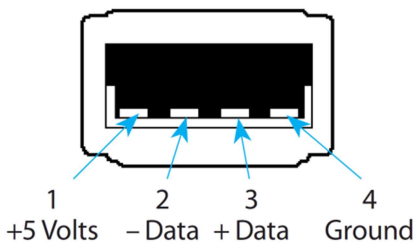


Figure I.21: USB type A

Type B : il est disposé sur les périphériques (imprimantes, caméscope,…).

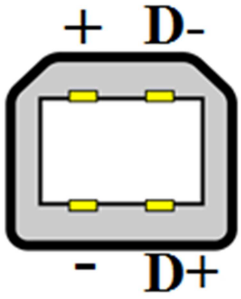


Figure I.22: USB type B

Le câble USB se compose de 4 fils, une paire torsadée pour le transfert des données, un fil au potentiel de +5V qui permet d´alimenter les périphériques USB si nécessaire et enfin la masse (télé alimentation). Il peut être blindé ou non, le mode basse vitesse de 1.5 Mbits/s ayant une tolérance supérieure aux perturbations électromagnétiques. Un blindage est fortement recommandé pour une utilisation pleine vitesse à 12 Mbits/s (longueur max de 5 mètres de câble entre 2 éléments).

VII.2.2. Transactions & Transferts

L'USB permet deux modes de transfert de données :

- mode **asynchrone** : comme un port série classique
- mode **isochrone** : permet une communication périodique et continue entre le contrôleur maître et les périphériques.

3 phases de **transactions** (protocoles) :

- Paquet Jeton (TOKEN) : Un jeton est émis par le host (obligatoire), puis un PID (Packet ID) identifie le type de transaction
- Paquet Données (DATA) : donnée utile (payload), optionnel
- Paquet d'Etat : (HANDSHAKE) : validation transactions, erreurs.

4 modes de **transferts** des paquets (terminaisons):

- Control Transfers : (paquets de contrôle: Message) : configuration, états, initialisation des périphériques.
- Bulk Data Transfers : (paquets en bloc, en rafale de données imprimantes, scanners.
- Interrupts Transfers : (paquets d´interruption) : message court émis de temps en temps (l'USB ne supporte pas les interruptions)
- Isochronous Transfers : (paquets isochrones) : transmission à intervalles de temps réguliers et à bande passante garantie.

VIII. Conclusion :

Dans ce chapitre nous avons pu constater que la mise au point d'un système de régulation englobe plusieurs étapes dont la communication et la supervision, mais la plus importante est l'identification, car l'identification d'un système dynamique réel revient à caractériser son modèle. Ainsi, le choix de la méthode d'identification adéquate est primordiale pour assurer avec efficacité la mise au point des boucles de régulations.

Chapitre 2

Description du Projet et des composants utilisés

I. Introduction

La citation des notions théoriques au chapitre précédent a pour but de les appliquer dans notre projet, ce dernier a été proposé et encadré par l'équipe fiabilité des composants à semi-conducteur de la Division microélectronique et nanotechnologie du Centre de Développement des Technologies Avancées(CDTA).

Ce centreest un établissement public à caractère scientifique et technologique (EPST). Il a pour mission de mener des actions de recherche scientifique, d'innovation technologique, devalorisation et de formation dans les domaines des sciences et des technologies, del'industrie et de la robotique, des dépôts et des traitements des matériaux, ses actions s'articulent autour des axes suivants :

- La microélectronique et la nanotechnologie
- L'architecture des systèmes et le multimédia.
- La productique et la robotique.
- Le dépôt de couches minces.
- Les milieux ionisés.

II. Description du projet :

Il consiste en la réalisation, la régulation et la commande d'un système de teste de composants électroniques, ces derniers sont soumis à un stresse électrique puis testés sous de différentes températures dans le but de déterminer leurs caractéristiques et de tester leur résistance dans un environnement équivalent aux environnements de fonctionnement.

III. Cahier des charges :

Plusieurs approches sont possibles pour réguler ce système, mais aussi, il faut respecter un cahier des charges bien spécifique selon l'utilisation finale des composants de la boucle de régulation.

Les principaux objectifs du projet sont :

- Programmation d'une plateforme de commande et de régulation sous logiciel LabVIEW.
- Commande et Identification des réactions du système et conceptiond'une solution adéquate pouvant faire fonctionner le système de manière optimale.
- Changements de températures de l'environnement doivent se faire grâce à un Peltier à base semi-conducteur.
- Conception et réalisation d'une carte de commande à base PIC18f2550, servant de pont entre le système et l'ordinateur.

- Récupération de la température de l'environnement de teste, la convertir et la transférer vers l'ordinateur afin de générer un signale régulé basé sur un PID numérique, contrôlant les actions du Peltier à travers la génération d'un signal MLI à partir du signal régulé.
- Transmission des données de la carte de commande vers l'ordinateur via connexion série (RS232).

IV. Description des composants utilisés :

Le matériel et logiciels utilisés pour mener à bien ce travail ont étaient mis à notre disposition par le département d'accueil, que nous décrivons comme suit :

IV.1. La chambre de teste:

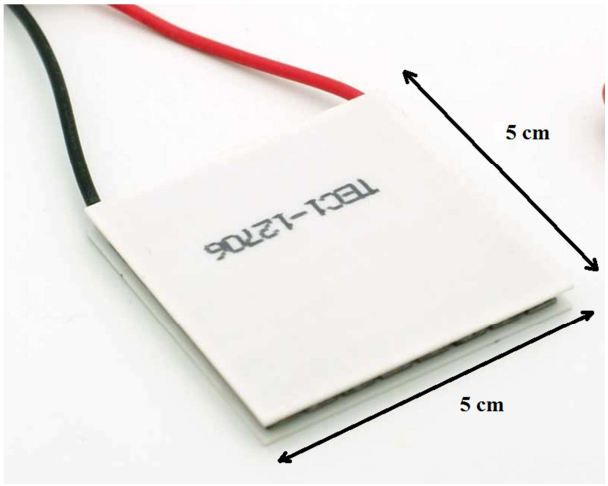
C'est une boite de 15cm de longueur, 10cm de largeur, 7 cm de hauteur contenant une petite chambre isolée thermiquement, cette chambre est l'environnement réel de test.

Elle englobe l'élément de refroidissement/réchauffement (Peltier), le capteur de température (sonde PT100), et une connexion électrique où sera branché l'élément à tester.

IV.2. Le Peltier :

Le module Peltier utilisé a une taille appropriée à la taille de la chambre de teste qui est de 6cm de longueur et de largeur, avec une hauteur 4cm, vu qu'il sera placé à la base de cette dernière.

Pour son fonctionnement, ce dernier consomme un courant de 10A avec une alimentation 24Ven continue, ce qui fournit une puissance de 240W.[4]



Vue II.1 : Les Dimension du Peltier utiliser.

IV.3. Sonde PT100 :

La sonde Pt 100 est un capteur de température a coefficient positifs (CTP) qui est utilisé dans le domaine industriel (agroalimentaire, chimie, raffinerie…). Ce capteur est constitué d'une résistance en Platine. La valeur initiale du Pt100 est de 100 ohms correspondant à une température de 0°C.**[9]**

Vue II.2 : Sonde PT100

Principe de fonctionnement :

$$RT = R_0(1 + AT + BT^2) \qquad (2.1)$$

Avec:

RT = résistance du thermomètre à la température T.

R_0 = résistance du thermomètre à 0°C.

T = la température en °C.

A = $3.9083*10^{-3}$ et B = $-5.775*10^{-7}$ (A et B sont des coefficients d'étalonnage)

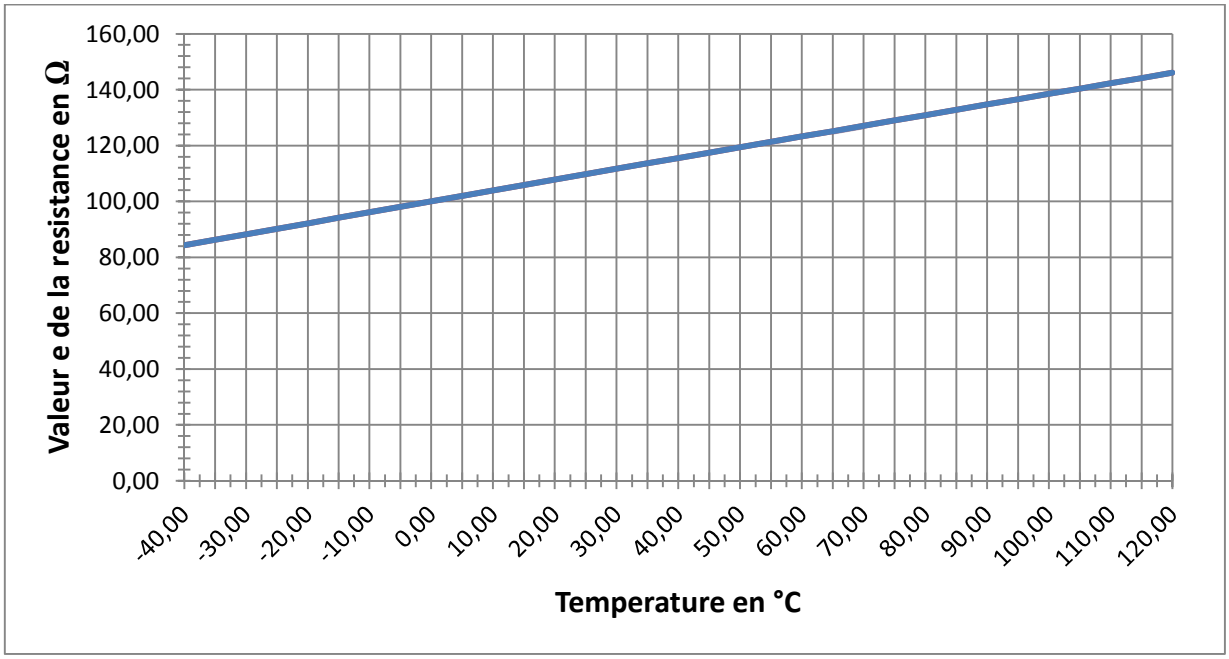


Figure II.1 : caractéristique de la Pt100

Nous constatant que la caractéristique de cette thermistance est linaire ce qui nous facilitera son utilisation.

IV.4. L'interface de contrôle :

Cette interface convertie et transmet les données émises par le capteur vers le régulateur implémenté sur l'ordinateur (logiciel LABVIEW, ou autre), et après traitement de ces données, elle retransmet le signal de commande du régulateur vers l'actionneur sous forme de signal MLI.

Elle est scindée en deux cartes :

IV.4.1. Carte de commande

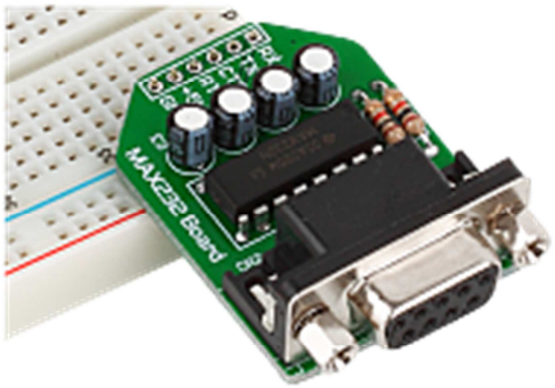
Elle est basée sur un PIC conçue pour être un pont de communication entre le système régulé et le régulateur.

Les composants principaux de cette carte, sont le PIC18f2550, MAX232 :

IV.4.1.a. Le MAX232 :

Le MAX232 est un circuit intégré, créé en 1987 par Maxim IntegratedProducts, qui convertit les signaux à partir d'un port série RS-232 en signaux compatibles TTL. Le MAX232 est un émetteur/récepteur double et convertit généralement les signaux RX, TX, CTS et les signaux RTX.

Les pilotes fournissent RS-232 sorties de niveau de tension (env. ± 7,5 V) à partir d'un seul + 5 V via des pompes sur puce charge et condensateurs externes. Ceci le rend utile pour la mise en œuvre RS-232 dans des dispositifs qui ne sont pas autrement pas besoin des tensions à l'extérieur du 0 V à + 5 V gamme, comme la conception d'alimentation ne doit pas être plus compliqué juste pour entraîner le RS-232 dans ce cas . [6]



Vue II.3 : Max232

IV.4.1.b. Le PIC18F2550 :

IV.4.1.b.1. Caractéristiques du PIC 18F2550

Le 18F2550 est un microcontrôleur en boîtier DIL 28 pattes possédant jusqu'à 24 E/S sur 3 ports (A, B, C, et éventuellement RE3).

Alimenté de 2 à 6 V continu, il est compatible avec le mode de programmation ICSP (In Chip Serial Programming), et dispose d'un oscillateur interne pouvant monter jusqu'à 8 Mhz. Cependant Sur oscillateur externe, le 18F2550 peut monter jusqu'à 48 MHz.

Au niveau mémoire ce PIC dispose de 16 kbit de mémoire flash pour le firmware, de 2 kbit de RAM et de 256 octets d'EEPROM.

Outre les E/S numériques classiques, ce PIC dispose de 10 CAN, de 2 comparateurs, de 2 PWM 10 bits, d'1 module de communication série synchrone et asynchrone, d'un module de communication USB, de 4 timers. [6]

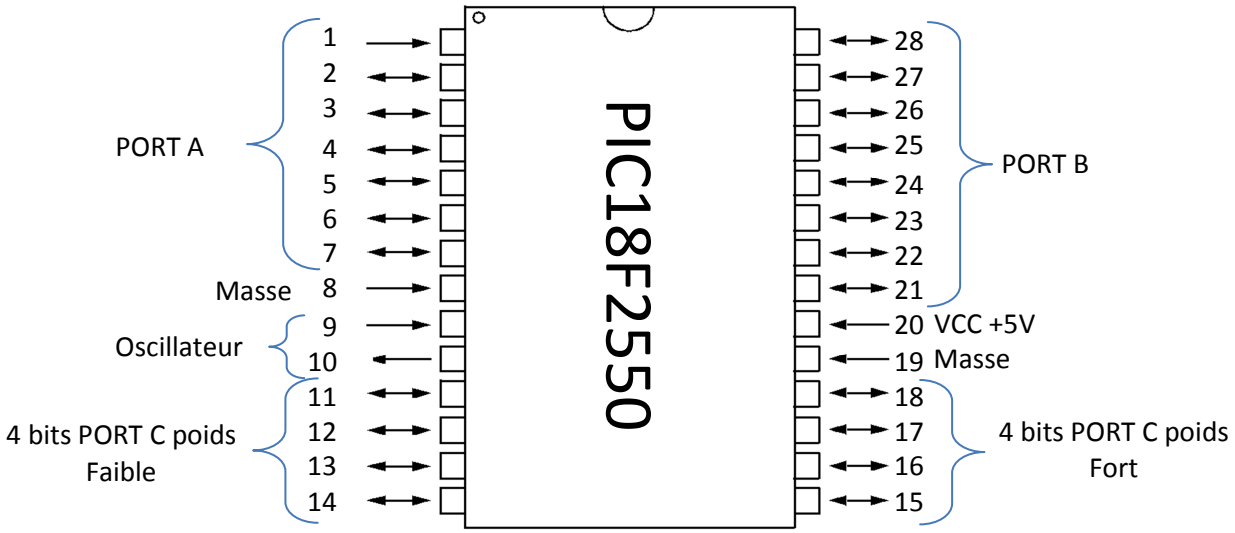


Figure II.2 : Brochages du PIC18F2550/2455

IV.4.1.b.2. Les entrées/sorties

Dans cette partie, nous allons désigner chaque patte par leur numéro. A noter, qu'une seule fonction est disponible à la fois, par patte.

- 1: E numérique RE3, reset
- 2: E/S numérique RA0, CAN0
- 3: E/S numérique RA1, CAN1
- 4: E/S numérique RA2, CAN2, tension de référence du comparateur, référence basse CAN

- 5: E/S numérique RA3, CAN3, référence haute CAN
- 6: E/S numérique RA4, entrée comptage timer 0, sortie comparateur 1, E module USB externe
- 7: E/S numérique RA5, CAN4, sélection de périphérique synchrone, sortie de comparateur 2
- 8: Masse
- 9: Entrée oscillateur, entrée horloge
- 10: Entrée oscillateur, sortie horloge, numérique RA6 PATTE
- 11: E/S numérique RC0, sortie horloge timer1
- 12: E/S numérique RC1, entrée horloge timer 1, sortie PWM 2, S module USB externe
- 13: E/S numérique RC2, sortie PWM 1
- 14: Référence de tension USB 3,3V (condensateur de 470 nF)
- 15: E/S numérique RC4, E/S — USB, E module USB externe VM
- 16: E/S numérique RC5, E/S + USB, E module USB externe VP
- 17: E/S numérique RC6, S RS232, S horlogesynchrone
- 18: E/S numérique RC7, E RS232, E donnée synchrone, sortie donnée SPI
- 19: Masse
- 20: Alimentation positive +5V
- 21: E/S numérique RB0, CAN12, interruption externe 0, entrée d'erreur PWM, entrée SPI, E/S I^2C
- 22: E/S numérique RB1, CAN10, interruption externe 1, horloge SPI, horloge I^2C
- 23: E/S numérique RB2, CANE, interruption externe 2, S module USB externe VM
- 24: E/S numériqueRB3, CAN9, S PWM 2, S module USB externe VP
- 25: E/S numérique RB4, CAN11, interruption de changement 0
- 26: E/S numérique RB5, interruption de changement 1
- 27: E/S numérique RB6, interruption de changement 2
- 28: E/S numérique RB7, interruption de changement 3

Les E/S se configurent via TRISA, TRISB, TRISC, TRISE

IV.4.1.b.3. Fonctionnement du PIC 18F2550

Dans la figure suivante est représenté le diagramme d'organisation des pins du PIC18f2550

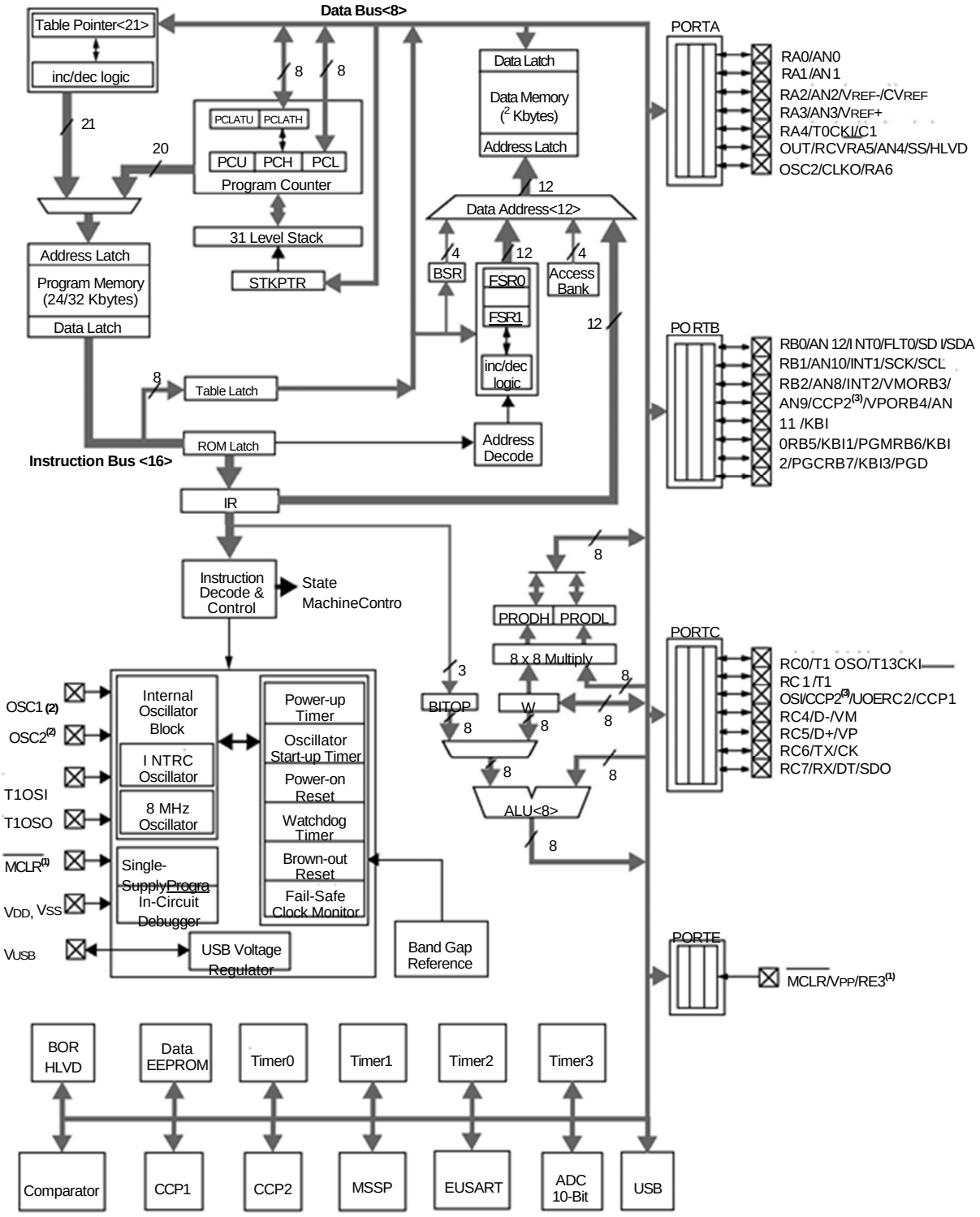


Figure II.3 : Organisation interne du PIC18F2550

IV.4.1.b.3. Les modes d'horloge

Dans cette partie, nous allons voir les différents modes d'horloge. Dans le cas d'un quartz (mode1), vous pourrez choisir horloge XT (jusqu'à 4 Mhz), ou HS (jusqu'à 20 MHz).

MODE 1 : XT, quartz ou résonateur céramique jusqu'à 4 MHz

MODE 2 : XTPLL, quartz avec PLL

MODE 3 : HS, quartz ou résonateur céramique jusqu'à 20 MHz MODE 4 : HSPLL, quartz avec PLL

MODE 5 : EC, horloge externe avec sortie F/4 sur RA6

MODE 6 : ECIO, horloge externe avec E/S sur RA6

MODE 7 : ECPLL, idem mode 5 mais avec PLL

MODE 8 : ECPIO, idem mode 6, mais avec PLL

MODE 9 : INTHS, oscillateur interne pour le PIC, quartz HS pour l'USB MODE 10 : INTXT, oscillateur interne pour le PIC, quartz XT pour l'USB

MODE 11 : INTIO, oscillateur interne pour le PIC, EC pour l'USB, E/S sur RA6

MODE 12 : INTCKO, oscillateur interne pour le PIC, EC pour l'USB, F/4 sur RA6

IV.4.1.b.4. Les Interruptions

A chaque source d'interruption activée, correspond un drapeau, un "flag", permettant de savoir si l'interruption a eu lieu.Chaque drapeau, une fois passé à "1" est à remettre à zéro de manière logicielle.

Ainsi pour savoir si une interruption a eu lieu, il suffit de surveiller le flag de l'interruption concernée.

Les registres concernés sont INTCON, INTCON2, INTCON3, PIR1, PIR2, PIE1, PIE2, IPR1, IPR2 et RCON (le seul bit de flag c'est RCON7). Autre chose à savoir: l'USB possède ses propres registres d'interruptions : UIR, UIE, UEIR, UEIE.

IV.4.1.b.5. Les TIMERS

Ils sont au nombre de 3. Un timer permet de compter le temps, ou des impulsions. Chaque timer possède des spécificités précises. La compréhension à 100% des timers est assez difficile. Cependant, la plupart des compilateurs nous permettent de nous en servir de manière relativement transparente.

- **TIMER 0**

Ce timer est le timer de base, et se configure via le registre T0CON.

Timer 8 ou 16 bits, le timer 0 est un compteur pouvant être initialisé à une valeur donnée, via le registre TMRO (8 bits) ou TMROL & TMROH (charger la valeur désirée dans ce(s) registre(s)). Lorsque l'interruption du timer 0 est activée, le passage de 0xFF à 0x00 du timer provoque l'activation du flag du timer0.

- **TIMER 1**

Second timer, il est sur 16 bits. Ce registre se configure via T1 CON. Tout comme le timer 0, le timer 1 est initialisable en chargeant la valeur désirée dans les 2 registres dédiés: TMR1H pour l'octet supérieur, et TMR1 L pour l'octet inférieur. A noter que l'entrée timer 1 (autre que T1 CKI), est en fait une porte logique inverseuse.

- **TIMER 2**

Troisième timer du PIC, 8 bits, il possède, en plus d'un prédiviseur, un postdiviseur. Il est, lui, initialisable via le registre TMR2. La période du timer 2 peut être configurée via PR2 (1 octet).

- **TIMER 3**

Quatrième timer du PIC, 16 bits, le timer 3 est notamment utilisé pour générer la PWM. Il est initialisable via le registre TMR3H et TMR3L.

IV.4.1.b.6. L'EUSART, SPI, USB

Comme pour les autres fonctions, elles sont transparentes ou émulées avec un compilateur. De fait, nous ferons que nommer ces derniers.

a) USART/EUSART:

L'USART/EUSART (PIC18f2550/4550), sur les PICs est généralement utilisé pour les communications série asynchrone, c'est-à-dire que les communications circulent sans horloge de synchronisation, un peu en "électrons libres".

Les registres utilisés sont TXSTA, TXREG, RCSTA, RCREG, BAUDCON, SPBRGH et SPBRG (pour définir le débit).

- **TXSTA:**

CSRC	TX9	TXEN	SYNC	SNEDB	BRGH	TRMT	TX9D

CSRC**:** source de l'horloge (1 = Mode maître, 0 = Mode esclave)

TX9**:** 1 = transmission sur 9-bits, 0 = transmission sur 8-bit

TXEN**:** 1/0 = activation/désactivation de la transmission.

SYNC**:** EUSART 1 = mode Synchrone, 0 = mode Asynchrone

SENDB**:** envoi d'un Caractère d'arrêt en mode Asynchrone:

1 = envoi un Sync Break dans la prochaine transmission (nettoyé après la fin)

0 = fin de transmission du sync break

BRGH**:** fréquence de transmission en mode Asynchrone: (1 = Haute0 = basse)

TRMT**:** 1 = TSR vide, 0 = TSR complet

TX9D**:** 9ème bit de transmission de données (une adresse ou bit de parité).

- **RCSTA** (pour la configuration de la réception)

RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D

SPEN : mise en service de l'USART.

RX9 : réception sur 8 ou sur 9 bits (1 = 9 bits, 0 = 8 bits)

SREN : validation de réception unique, en mode synchrone Maître :

1 = Active la réception unique, 0 = Désactive la réception unique, ce bit est effacé après que la réception soit achevée

CREN : lance la réception continue.

ADDEN : permet de filtrer la réception.

FERR : indique une erreur de trame.

OERR : indique une erreur de type overflow (dépassement de capacité).

RX9D : contient le 9ème bit de votre donnée reçue.

En plus de ces deux registres nous comptons trois registres utilisés dans la transmission :

- **SPBRGH/SBPRG** : registre HAUT/BAS de la fréquence de transmission.
- **TXREG** : Le registre TXREG contient la valeur sous 8bits à envoyer
- **RCREG** : Le registre RCREG contient la valeur sous 8bits reçue

b) SPI

LE SPI (Serial Peripheral Interface) est un mode de communication série synchrone 3 fils, fonctionnant en mode maître-esclave. Plusieurs esclaves sont possibles, grâce à une patte dédiée sur le périphérique esclave SPI permettant la sélection de l'esclave avec lequel le maître désire dialoguer.

Le maître est seul capable de commencer un dialogue.

La différence du SPI par rapport à l'I2C est qu'il possède deux fils de données. Un pour les données entrantes, et l'autre pour les sortantes (comme pour le RS232). Ainsi, l'échange des données est plus rapide.

Ce protocole utilise donc 3 fils, voir 4 si l'on possède plusieurs esclaves:

- CS\ ou SS\: patte de sélection
- CMD ou SDI: entrée du périphérique
- CLK ou SCK: horloge
- DATO ou SDO: sortie du périphérique

c) USB:

A priori compliqué à programmer, impression justifié, l'utilisation de l'USB via un compilateur devient relativement facile. En effet, une fois de plus, tout est totalement transparent. De fait, on ne cherche pas à comment le port USB fonctionne précisément, mais plutôt comment utiliser l'USB avec le compilateur.

Les registres permettant d'initialiser l'USB sont UCON, UCFG, USTAT, UADDR, UEPn, UFRMH ET UFMRL.

A noter cependant, que l'USB possède ses propres interruptions avec des registres dédiés.

IV.4.1.b.7. LeMLI (PWM)

La PWM pour Pulse Width Modulation, ou en français MLI pour Modulation en Largeur d'Impulsion fait partie d'un bloc interne du PIC: le CCP; ou Capture/Compare/PWM.

Elle permet de gérer l'énergie transmise à l'extérieur. En effet, si un signal continu correspond à 100% d'énergie, un signal carré dont la durée d'état haut égale celle d'état bas correspond à 50 % d'énergie. Le pourcentage d'énergie transmis se calcule en faisant le rapport de la durée d'état haut sur la durée de la période (Ton/Toff).

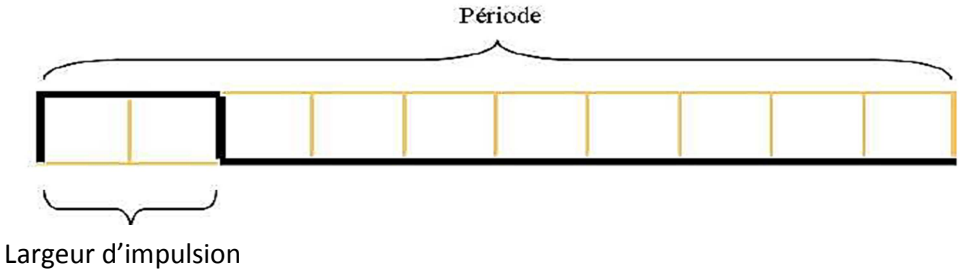


Figure II.4 : illustration d'un signale MLI(PWM)

Deux formules permettent de calculer la durée de la période, et la largeur de l'impulsion.

PERIODE=(PR2+1)*4*Tosc*(prédiviseurtimer 2)

LARGEUR=(10 bits)*Tosc*(prédiviseurtimer 2)

- Tosc : la période de l'oscillateur servant d'horloge au PIC.
- Les 10 bits correspondent à l'octet de CCPR1L et des 2 bits de DC1Bx. Cependant cette résolution de 10 bits est la valeur maximale. En effet, selon la fréquence désirée, cette résolution peut changer. Pour connaître la résolution maximale, on utilise la formule suivante:

  RESOLUTION= (log (Fosc/Fpwm))/log (2) bits.

Par exemple, avec le prédiviseurtimer 2 à 16, et un PR2 à 255 (OXFF), nous avons une résolution maximale de 10 bits, et une fréquence de PWM de 1,22 Khz.

Autre exemple, avec un prédiviseur à 1, et un PR2 à 31, la résolution maximale est alors de 7 bits, et une fréquence de PWM de 156,3 KHz.

Ici, les registres utiles sont le CCP1CON, CCP2CON, PR2, CCPR1 L, CCPR2L. De plus, comme précisé précédemment, c'est le timer 2 qui est utilisé ici. De fait, il faudra également utiliser T2CON, afin d'activer le timer 2. Cependant, la plupart des compilateurs rendent toutes ces actions transparentes.

- **CCP1CON**

-	-	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0

DC1B1-0: contient les 2 bits de poids faibles de la période (PWM 10 bits)

Les 8 autres bits doivent être chargés dans le registre CCPR1L.

CCP1M3-0: Pour la PWM, égal à 11xx

- **CCP2CON**

-	-	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0

DC2B1-0: contient les 2 bits de poids faibles de la période (PWM 10 bits)

Les 8 autres bits doivent être chargés dans le registre CCPR2L.

CCP2M3-0: Pour la PWM, égal à 11xx

IV.4.1.b.8. Les CAN (Convertisseurs analogiques numériques)

Le PIC dispose de 10entrées CAN, répartis sur ses entrées.

3 registres permettent la configuration de ces convertisseurs, et 2 autres registres permettent de récupérer les résultats (sur 10 bits).

- **ADCONO**

-	-	CHS3	CHS2	CHS1	CHS0	GO-DONE\	ADON

CHS3-0: sélection du CAN (0000 pour le 0 à 1100 pour le 12, le 5,6,7, ne sont disponibles que sur le 4550)

GO-DONE\: à 1, la conversion est en cours, à 0, elle est finie

ADON: activation (1) ou non des CAN

- **ADCONI**

-	-	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0

VCFG1: définie la source de référence négative: Vref- (1) ou Vss (0)

VCFG0: définie la source de référence positive: Vref1 (1) ou Vdd (0)

PCFG3-0: défini les CAN utilisés, et ceux inutilisés

Pour la sélection des entrées analogiques à utiliser,nous agissons sur les bits PCFG3--PCFG0, suivant le code appliqué. Nous obtenons le tableau suivant :

PCFG3: PCFG0	AN12	AN11	AN10	AN9	AN8	AN7(2)	AN6(2)	AN5(2)	AN4	AN3	AN2	AN1	AN0
0000(1)	A	A	A	A	A	A	A	A	A	A	A	A	A
0001	A	A	A	A	A	A	A	A	A	A	A	A	A
0010	A	A	A	A	A	A	A	A	A	A	A	A	A
0011	D	A	A	A	A	A	A	A	A	A	A	A	A
0100	D	D	A	A	A	A	A	A	A	A	A	A	A
0101	D	D	D	A	A	A	A	A	A	A	A	A	A
0110	D	D	D	D	A	A	A	A	A	A	A	A	A
0111(1)	D	D	D	D	D	A	A	A	A	A	A	A	A
1000	D	D	D	D	D	D	A	A	A	A	A	A	A
1001	D	D	D	D	D	D	D	A	A	A	A	A	A
1010	D	D	D	D	D	D	D	D	A	A	A	A	A
1011	D	D	D	D	D	D	D	D	D	A	A	A	A
1100	D	D	D	D	D	D	D	D	D	D	A	A	A
1101	D	D	D	D	D	D	D	D	D	D	D	A	A
1110	D	D	D	D	D	D	D	D	D	D	D	D	A
1111	D	D	D	D	D	D	D	D	D	D	D	D	D

Tableau II.1 : de configuration des entrées analogiques.

A= Entrées Analogiques D=E/S numériques

- **ADCON2**

ADFM	-	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0

ADFM: justification à gauche (0) ou à droite

ACQT2-0: définie le temps d'acquisition

ADCS2-0: définie l'horloge pour l'échantillonnage

Deux autres registres permettent de récupérer la valeur de la conversion: ADRESH et ADRESL. Pour des raisons de commodités de traitement de la conversion, 2 cas sont possibles selon la valeur d'ADFM (ADRESH en haut)

IV.4.2. Carte de puissance Commandant l'actionneur :

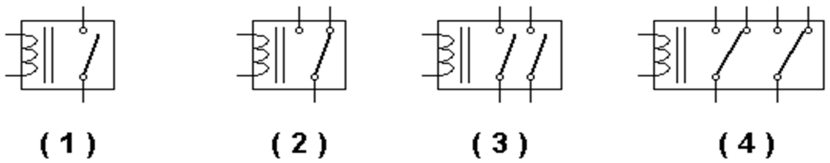
Elle se compose principalement de relais et de transistors de puissance :

IV.4.2.1Les relais

Lesrelaismécaniquessontdescommutateurs àélectroaimantpossédantunouplusieurscontacteursquipeuventêtreuniquementnormalement(ouvertaurepos) ou (fermer au repos).

Ilssontcomposésdedeuxparties:

Unepartiecommandequiestlabobineetunepartiepuissancequiestle contacteur



Vue II.4 : les relais mécaniques

V. Supervision et régulation :

Le logicielimposé pour mener ce modeste travail à bien est le LABVIEW celui-ci englobe toutes les opérations de supervision, de commande et de régulation.

En effet, nous avons à notre disposition une interface graphique et ludique, permettant de commander les actions agissant sur le procédé :

- Démarrage, choix et configuration de la PWM à utiliser.
- Choix de la température à appliquer sur le composant.
- Normalisation des signaux d'acquisition.
- Régulation du signal de commande en fonction des données acquises et de la consigne souhaité.

V.1. Présentation du logiciel LabVIEW

LelangagedeprogrammationLabVIEW(LaboratoryVirtualInstrumentEngineeringWorkbench) est un environnement de programmation à caractère universel. C'est un programme dont le but estde contrôler des processus allant du simple capteur ou de l'actionneur, en passant parune chaîne de fabrication complète.

LabVIEWdisposed'unnombreimportantde fonctionsgraphiquesquipermettentfacilementd'acquérirdesdonnées,delestraiteretd'afficher les résultats. [7]

V.1.1.Définition

Contrairementauxlangagesdeprogrammationtextuelsoùsesontlesinstructionsquidéterminent l'ordre d'exécution du programme, LabVIEW utilise la programmation par flux avec des icônes aulieu des lignes de textede données : c'est le flux de données transitant par les nœuds sur le diagramme qui déterminel'ordre d'exécution des VIs (Virtual Instruments) et des fonctions.

Nous parlerons d'instrumentsvirtuels car leur apparence et leur fonctionnement sont semblables à ceux d'instruments réels.[7]

Le LabVIEW est un outil d'acquisition, d'analyse et de présentation de données comme il estillustré dans le tableau suivant :

Acquisition	Analyse	Présentation
Contrôle d'instruments	Traitement numérique	Affichage de données
-GPIB IEEE 488 -RS 232 -VXI	-Génération de signaux -Filtrage, fenêtrage -Analyse fréquentielle	-Interface interactive -Graphiques et courbes
Acquisition de données	Traitement statistique	Stockage des données
-E/S analogiques -E/S numériques	-Régression, lissage -Moyenne, écart type	-archivage -impression

Tableau 2.2: Fonctionnalités de LabVIEW.

GPIB: generalpurpose interface bus

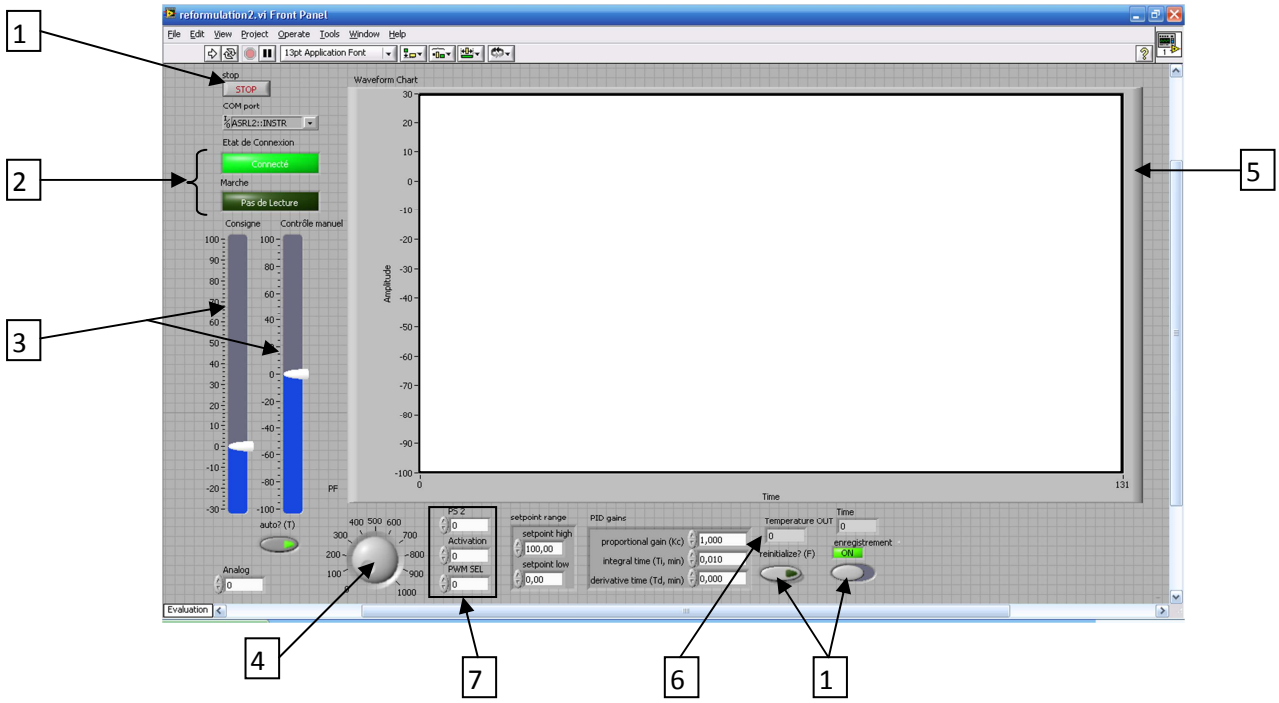
VXI : bus informatique industriel destiné à l'instrumentation

V.1.2.Définition d'une VI (Virtual Instruments)

LesVissecomposentdetroiséléments principaux: Interface d'affichage, diagramme et icône, connecteur

V.1.3. Interface d'affichage (Front Panel)

L'Interface d'affichage**(**face avant)réceptionne les données d'entrées par l'utilisateur et affiche celles fournies en sortie par le programme. Cette face peutcontenir différents types d'instruments virtuels tels que des boutons poussoirs, des interrupteurs,des graphes et d'autre sorte de commande et d'indicateurs.

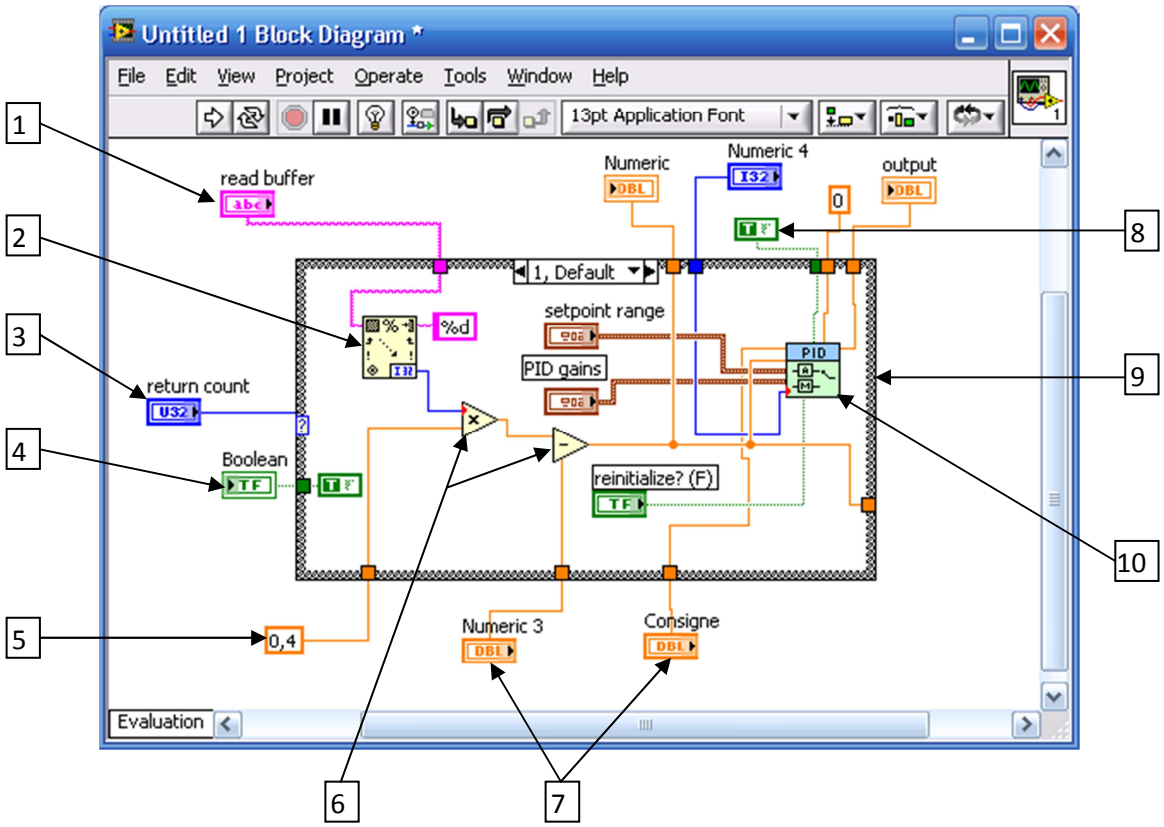


Vue II.5: Exemple de la face avant d'un VI

1. Contrôleur buttons (booléenne)
2. Indicateur led (booléenne)
3. Contrôleur sous forme de bar de défilement (numérique)
4. Contrôleur boutons rotatif (numérique)
5. Affichage graphe
6. Indicateur numérique sous forme zone texte
7. Contrôleur numérique sous forme zone texte

V.1.4.Diagramme (Block Diagram)

Le diagramme contient le code source graphique des instruments virtuels. On programme leVI pour contrôler et remplir les fonctions sur les entrées et sorties crées dans la face avant.Le diagramme peut contenir des fonctions et des structures issues des bibliothèques de Visintégrées à LabVIEW. Il peut aussi contenir des terminaux associées à des commandes et àdes indicateurs crées dans la face avant.

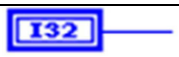


VueII.6: exempled'une régulation sous forme de diagramme

1. Représentation d'une zone de texte (chaine de caractères) dans le Diagramme
2. Fonction de conversion
3. Indicateur booléenne
4. Indicateur booléenne
5. Constante réel
6. Operateur mathématique
7. Contrôleur réel
8. Constante booléenne
9. Structure de condition case (case structure)
10. Fonction PID (appelle d'une VI)

V.1.5. Structures de donnéesVariables

Afin de les identifier plus facilement dans les diagrammes, chaque type de variables possèdeune couleur Existantes :

Entiers	Réels	Booléen	Chaînes de caractères
			

V.1.6.Structures de programmation dans LabVIEW

Lesstructuresdeprogrammationsontdesreprésentationsgraphiquesdebouclesetdeconditions.

Elles sont utilisées pour répéter des blocs décode et exécuter le code de manière conditionnelle ou dans un ordre spécifique. Cesstructures se divisent en trois types : séquence, condition, itération.

- **Structure séquence**

Les structures de séquence sont un moyen pour imposer l'ordre d'exécution des taches, leurs cadres ressemblent à des diapositives placées lesunes derrières les autres ; l'exécution commence par le code contenu dans la première (n°0)puis continue dans l'ordre 1, 2,ect.

- **Structure Condition**

La structure conditionnelle peut correspondre à unif … then … elsedans les langagestextuels.

LabVIEW représente l'alternative à l'aide de plusieurs diagrammes alternatifs

- **Structures d'itération**
  - **La boucle while**

La boucle while (tant que) permet de répéter une ou plusieurs instructions N fois (N nombre connu).

Al'intérieurdelabouclewhilesetrouveunterminald'entréelocalgénérantl'entierindiquant l'indice d'itération de la boucle. Un terminal de sortie de type booléen permetd'arrêterla boucle lorsque la valeur du stop soit égale à 1.

V.1.7.Traitements de données dans LabVIEW

- **Les fonctions prédéfinies**

Commetouslesautreslangagesdeprogrammation,LabVIEWpossède desfonctionsprédéfinies qui permettent de traiter les différents types de données.

Ainsi, on trouve des fonctions liée aux variables numériques (entiers, réels et complexes),aux variables booléennes, aux chaînes de caractères et aux tableaux. Ajoutant à cela desfonctions liées à la comparaison.

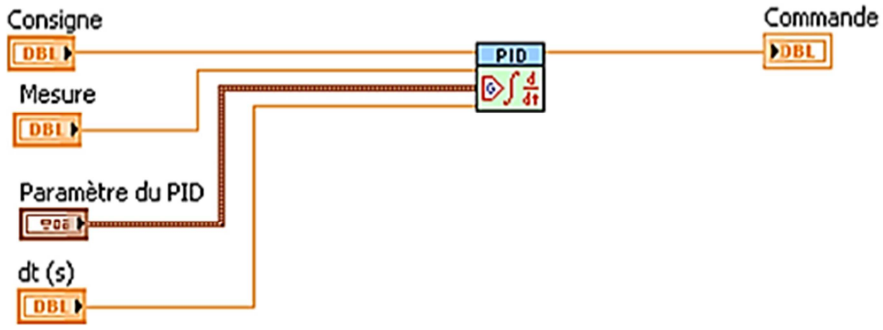
- **Boîtes à outils mathématique**

LabVIEW contient aussi des boites de calcul mathématiques qui servent à introduire descommandes complexes telles que (, ode45, sin, cos, linspace, ...)

- **La bibliothèque de contrôle PID**

Il ajoute aux fonctions standards de LabVIEW un ensemble d'algorithmes de régulation P, PI,PDetPID,avectouteslesoptionsquel'ontrouvegénéralementdanslessystèmesde régulation complets.

Tous les régulateurs dans cette bibliothèque ont une structure parallèle qui estreprésenté comme suit:



Vue II.7: Régulateur PID

V.1.8.Graphes dans LabVIEW

Les graphes sont accessibles dans le module ➔contrôle ➔Graphe indicator. Les graphes inscrivent les pointsen donnant une abscisse initiale et un incrément à chaque nouveau point.LabVIEW propose 3 principaux types de graphes :

- **Graphe déroulant**

Le graphe déroulantest une sortie 2D représentant une courbe (ou plusieurs courbes) dont lespoints sont donnés point par point

- **Graphe XY**

Le graphe XY permet de tracer des courbes paramétriques, en annulant la base de temps.Pour ce faire, il faut envoyer un clustercontenant les points pour X, et les points pour Y surchaque composante du clusteren entrée du graph

V.1.9.Transmission des données :

Pour contrôler, réguler ouagirefficacementsurunprocessusphysique,chimiqueoubiologique,naturelouindustriel, il faut avoir un système de transmission tel que le RS232.

LabVIEW dispose de VI spéciales appelés VISA RS232, elle permet de transmettre le signal acquis via port série en code ASCII de l'ordinateur vers l'interface de commande pour un traitement ou une visualisation.[7]

Vue II.8: Exemple de connexion RS232 dans LabVIEW

Dans Cet exemple on peut voir que la transmission rs232 se fait avec des VI bien spécifiques, tel que l'ouverture se fait avec par un VISA OPEN, et fermé par un VISA CLOSE, configuré avec un VISA CONTROL SERIAL PORT, on envoi des données via la VISA WRITE, et on peut lire avec un VISA READ.

VI. Conclusion :

De façon générale, la réalisation d'un travail passe d'abord par la description des outils utilisés, Ce chapitre nous a permis de mettre en lumière les caractéristiques de chaque composant et le rôle qu'ils doivent jouer dans la réalisation du système.

Après présentation de ces outils, nous pouvons élaborer une stratégie de conception matériel et logiciel que nous aborderons dans le chapitre suivant.

Chapitre 3

Conception et simulation

I. Introduction :

La conception d'un système de régulation numérique exige la disposition d'un calculateur tel qu'un ordinateur, un automate programmable ou un microcontrôleur.

Dans notre cas, la récupération de données sera effectuée par un ordinateur via une carte d'acquisition à base de PIC18F2550, cette dernière joue le rôle d'un CAN (Convertisseur Analogique Numérique) et d'un générateur PWM « MLI » (Modulateur de Largeur d'Impulsion). Nous l'utilisons pour récupérer la température de la chambre de teste et actionner le Peltier en chauffage ou refroidissement en réponse aux exigences du PID implanté dans le PC.

II. Organisation et fonctionnement

La disposition des éléments de la boucle est de la manière suivante:

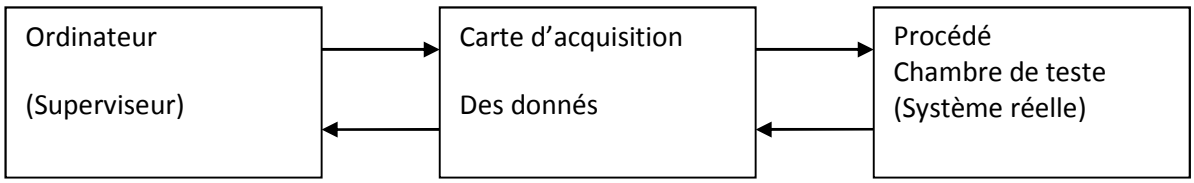


Figure III.1 : diagramme du système à concevoir

II.1. Analogie à une boucle de régulation traditionnelle:

Comme nous l'avons cité au début, notre travail consiste en la régulation de température avec un régulateur numérique et l'utilisation d'un ordinateur et un PIC. Pour ce faire, nous avons opté pour la disposition suivante :

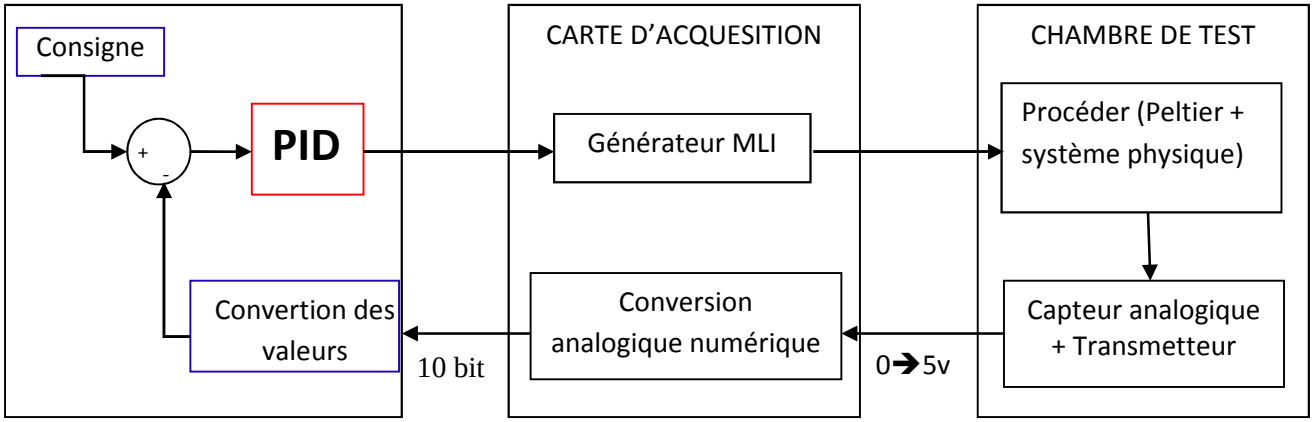


Figure III.2 : boucle de régulation dans le système

Vu que nous disposant de logiciels (LABVEIW, DELPHI, Matlab…) qui peuvent être préprogrammé à la régulation, nous allons placer le PID dans l'ordinateur pour faciliter la programmation de celui-ci, et permettre la supervision et la commande sur le pc.

Le générateur MLI et le convertisseur analogique numérique étant intégré dans le microcontrôleur donc ils sont automatiquement dans la carte d'acquisition.

Le capteur de température et l'actionneur (PELTIER) seront implanter dans la chambre de teste qui est illustrée sur la figure suivante.

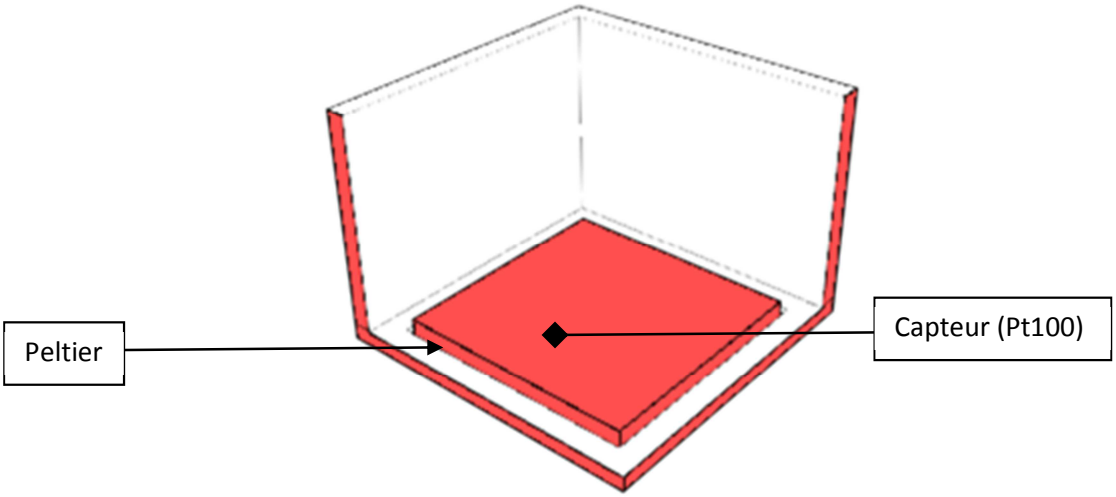


Figure III.3 : Disposition des éléments dans la chambre de teste

II.2. Fonctionnement du système :

Le capteur de température se trouvant dans la chambre de teste est relié a un circuit de conditionnement qui est brancher à l'une des entrés analogiques de la carte d'acquisition, cette dernière convertie la grandeur analogique (la température) en numérique puis, elle la transfère vers l'ordinateur afin de générer une réaction par le régulateur PID implanté dans celui-ci.

Après traitement de la grandeur, la carte transforme les commandes du PID numérique en signaux MLI, qui sont envoyés vers le Peltier (l'actionneur de refroidissement et de chauffage).

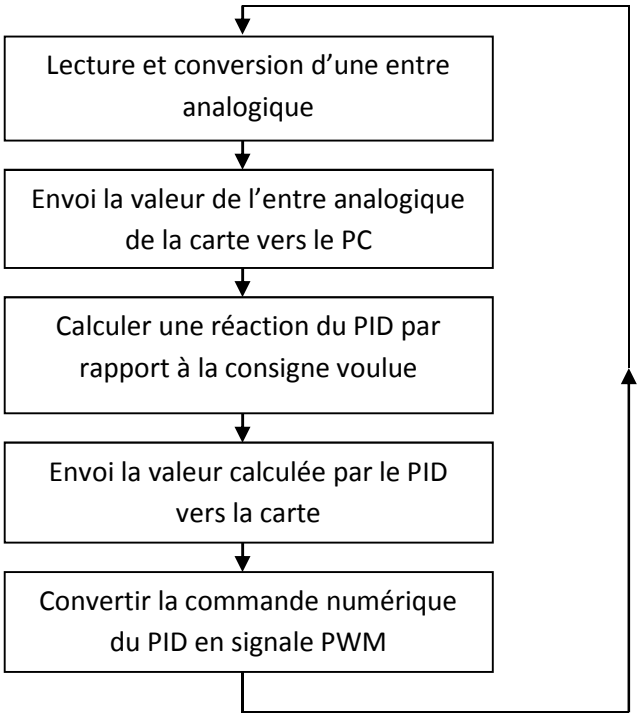


Figure III.4 : Etapes du fonctionnement général du système

II.3. Approche programme :

Pour obtenir le fonctionnement voulu, il nous faudra connaitre le mode de transmission entre le PIC et le PC.

La transmission se fait en série asynchrone full-duplex, donc une mise au point de quelques mots de commande s'impose pour parvenir à coordonner la communication des données entre la carte et l'ordinateur.

Il faut configurer le module USART du pic en premier, afin d'avoir les mêmes fréquences de transmission entre le module et le programme de l'ordinateur, sinon les données ne serons pas reconnues par les deux appareils.

Pour ce faire on suit les étapes suivantes :

Pour programmer l'USART du PIC, nous utiliserons les registres suivants :

TXSTA : registre contrôlons la transmission.
RXSTA : registre contrôlons la réception.
SBPRG : registre contenant la valeur configurant la fréquence de l'USART.
TRISC : registre permettons de contrôler la direction du PORTC.

Configuration de l'USART :

1. TRISC = 0x80

   RC7 « RX, pate 18» en entre

   RC6 « TX, pate 17» en sortie

2. TXSTA = 0x24

   Activer la transmission en mode asynchrone avec 8 bits de donnés et **BRGH=16.**

3. RCSTA = 0x90

   Activer l'USART réserver « RC7 et RC6 » pour « TX et RX » et la réception avec 8 bits de donner

4. SPBRG = 129 pour une fréquence de 9600 Baud (bits/s) avec BRGH=16

   Le calcule de SBPRG se fait avec une formule qui est en rapporte aves la fréquence du quartz

SPBRG = (FSOSC/(BRGH*Baud désiré))-1

   SPBRGH = (20 000 000/(16*9600))-1 = 129.2

La vérification se fait avec :

V Transmission= FSOSC/(BRGH*(SPBRG+1))

20 000 000/(16(129+1)) = 9615.13

(15.13*100/9600) = 0.157%

Cette erreur approche les 0.16% qui est assez faible, donc nous gardons la valeur de SPBRG = 129 pour une fréquence de transmission de 9600 Baud.

Après configuration de la vitesse de transmission, nous devons passer à la mise au point des mots de commande.

II.3.1 Les mots de commande

- **AN XX** Lire l'entrée analogique numéro XX

  Le retour est la valeur numérique de l'entrée XX.

  Exemple : AN 00 Lire l'entrée analogique numéro 00 sur la broche RA0

- **PWSXXX** Configure la fréquence de la PWM pas de retour de valeur

  S Configure Le présalaire de la PWM

  S peut prendre les valeurs 1, 2, 4

  1 ➔ 1 horloge Système /1

  2 ➔ 4 horloge Système /4

  4 ➔ 16 horloge Système /16

  Exemple : PF2100 ➔ F=100*10=1KHZ

  S=2 ➔ fosc/4 si 20MHZ donc 20/4=5MHZ

- **PTEXXX** Changer le cycle de la PWM E en XXX %

  Exemple PT1050➔PWM1=50% de F

- **PAEX** Activer la PWM X si E=1 désactiver si E=0

  Exemple PA12 ➔ Activer PWM2

  PA02 ➔ Désactiver PWM2

II.3.2 Fonctionnement du système

Les programmes fonctionneront de la manière suivante :

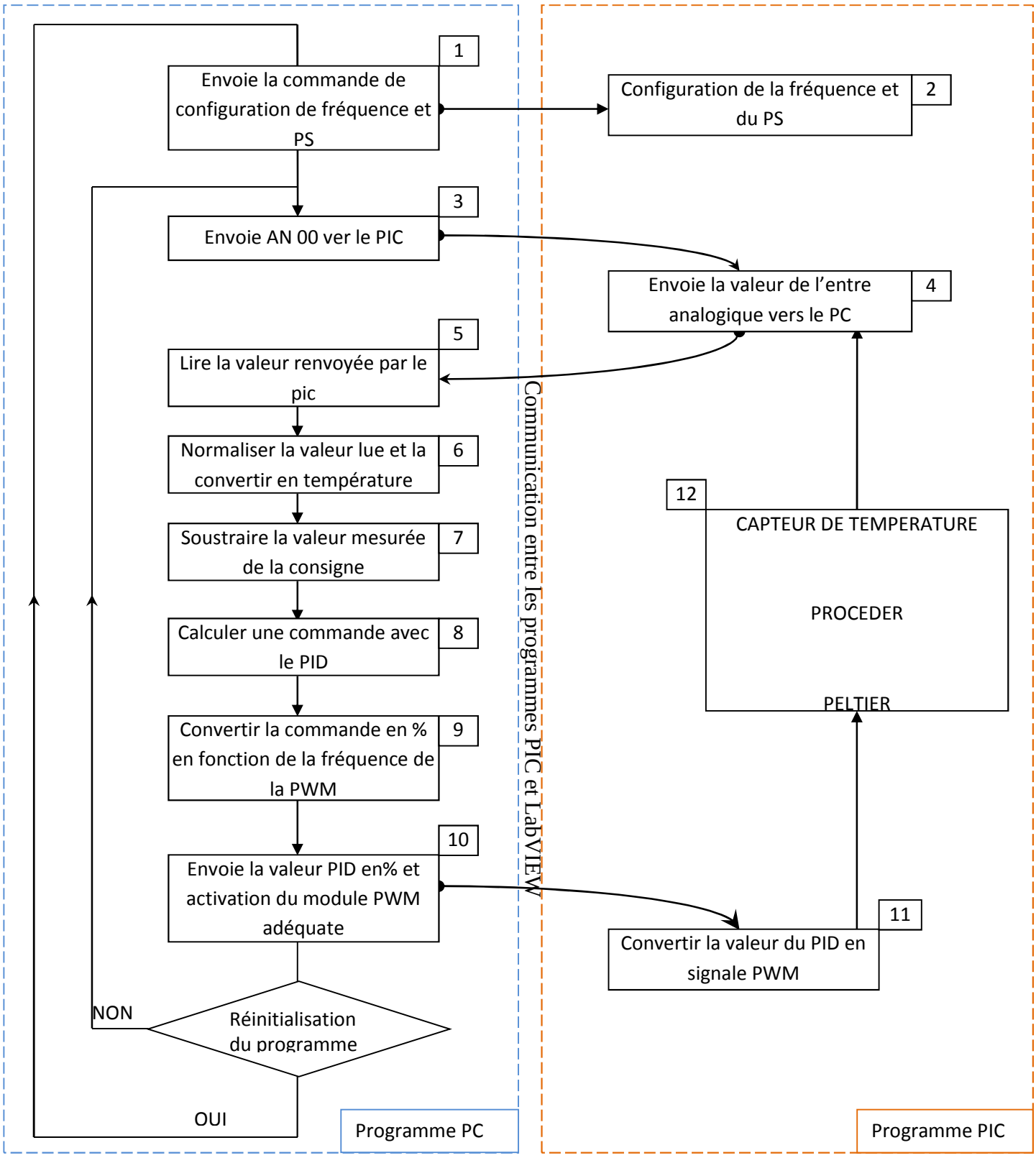


Figure III.5 : description du fonctionnement du procédé

II.3.3.Description de l'organigramme:

1. Envoi de la commande « **PWSXXX** » vers le PIC afin de configurer le présalaire et la fréquence de la PWM.
2. Application de la commande de l'étape « 1 » (configuration de la fréquence et du PS)
3. Le programme de l'ordinateur envoi les commande « **ANXX** » pour lire l'entrée analogique, puis attend la réponse du PIC.
4. Le PIC ayant reçu le message du programme du PC lance la convection analogique de l'entre XX, puis renvoi la valeur de celle-ci.
5. Cette étape est lancer réellement juste après l'étape (3), elle restera en attente jusqu'à réception la réponse du PIC, avec un délai d'attente, une fois que celui-ci est dépassé, un message d'erreur s'affichera.
6. Sachant que la valeur de l'entrée analogique est comprise entre [0..1023], nous devons la normaliser pour la convertir en température suivant le capteur à utiliser.
7. Cette étape joue le rôle d'un soustracteur entre la consigne et la température reçue.
8. Calcule de la réaction adéquate du PID implanté dans l'ordinateur pour la transférer au PIC.
9. Avant de transmettre la commande du PID vers le PIC, nous devons d'abord la transformé en pourcentage (%) pour activer la PWM.
10. Envoi des commandes « **PTEXXX% PA1E** » vers le pic dans le but de réaliser les commandes suivante :

    a. **PTEXXX%** : envoyer la valeur de la réaction du PID en %.
    b. **PA1E** : activer le module PWM concerné par l'action voulue.

11. Convertir la valeur envoyée par l'ordinateur en un signale MLI (PWM), qui sera interprété par le Peltier en chauffage ou refroidissement.
12. Cette étape ne concerne pas le programme, mais elle symbolise la réaction du système aux variations de température générées par l'actionneur, qui sont recueillies par le capteur de température.

III. Conception Electronique

III.1. Choix des éléments

Les éléments sont choisis selon le cahier des charges, suivant des critères bien déterminés.

- La vitesse de réaction
- La plage de fonctionnement
- Le volume
- La flexibilité de programmation

- Les composants thermoélectriques (Peltiers) :

  Ce sont des composants thermoélectriques à base semi-conducteurs, fonctionnant en deux modes :

  - Alimentation directe ➔ fonction chauffage
  - Alimentation indirecte ➔ fonction refroidissement

- La PT100 :

  C'est une thermistance a coefficient positif (CTP), sa valeur à 0° =100Ω, ayant une plage de linéarité adéquate à la plage de température désiré -20°C ➔ 80°C

- Le PIC18F2550 :

  C'est un microcontrôleur de MICROCHIP possédant plusieurs périphériques interne utile à notre projet qui sont :

  - CAN
  - USART
  - Une paire de module PWM (Pulse Wide Modulate)
  - Une Horloge de 20 MHZ
  - Module USB

- L'ordinateur :

  C'est un élément incontournable dans ce projet vu les facilites qu'il offre :

  - Programmation rapide avec plusieurs langages et une infinité d'outils.
  - Elément d'affichage et de supervision.

III.2. La disposition des éléments

Pour une flexibilité optimale nous avons opté pour une dissociation en deux parties de l'interface de commande (la carte de commande, la carte de puissance).

III.2.1. Carte de commande :

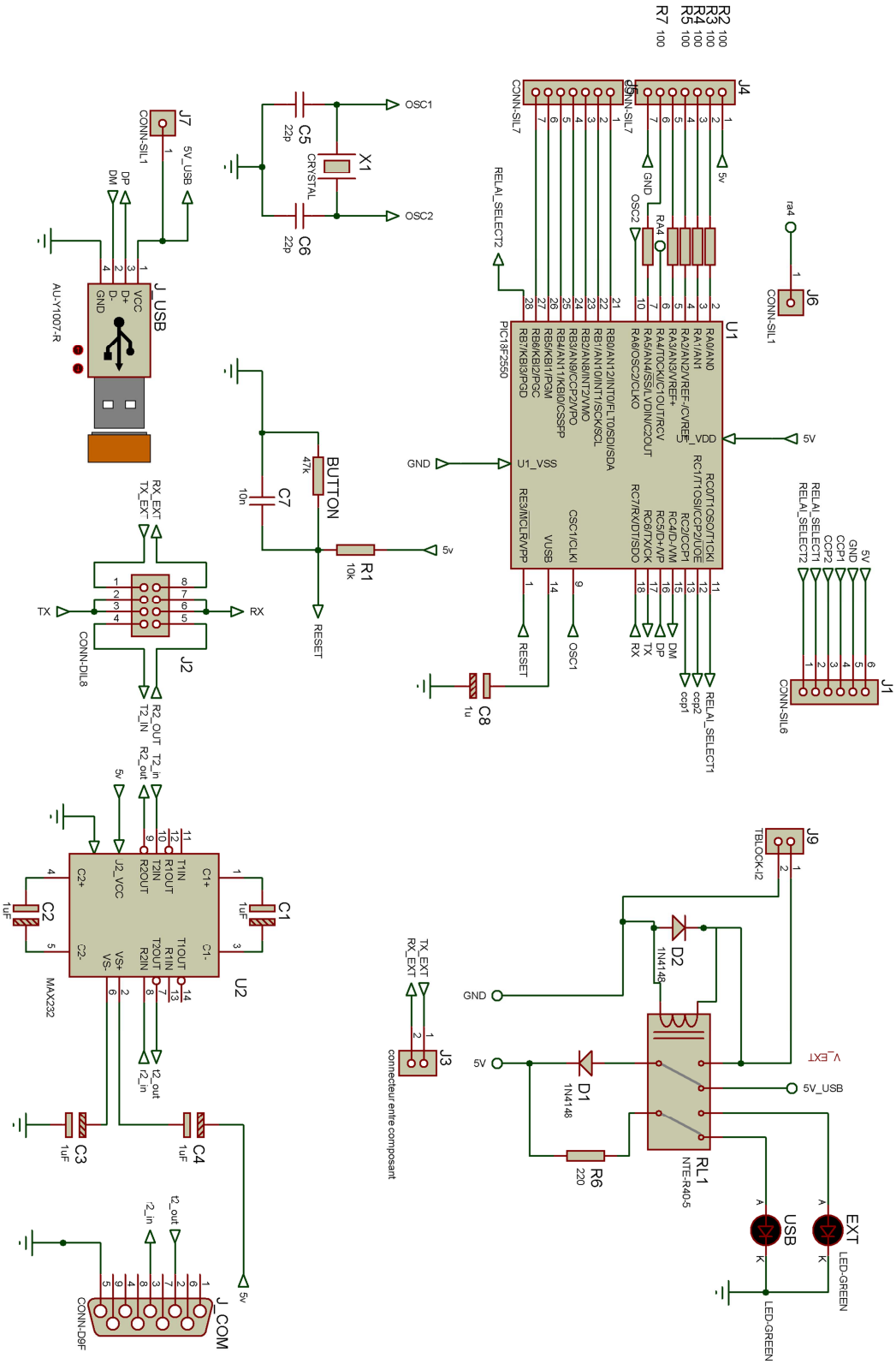


Figure III.6 : Schémas de la carte de commande

Cette carte joue le rôle d'interface de communication entre la carte de puissance et l'ordinateur, mais aussi de convertisseur analogique numérique A/N et de générateur PWM.

Composé de trois parties distinctes alimentation, microcontrôleur, communication :

III.2.1.1. L'alimentation :

Son objectif est d'assurer l'alimentation des composants de la carte avec 5 volts, soit par une prise externe « J9 » ou bien par le connecteur USB, pour se faire nous avons conçus un système d'auto-basculement entre les deux sources grâce à un relai de 5v qui permet la détection d'une prise externe.

Son fonctionnement est basé sur l'utilisation d'un relai RL1 de 5v.

Au repos, il se place sur l'alimentation de l'USB 5V_USB, mais si une alimentation externe est connectée sur « J9 » celle-ci excite le relai RL1 qui passe à son tour sur l'entre V_EXT pour assure l'alimentation de la carte à travers la diode D1.

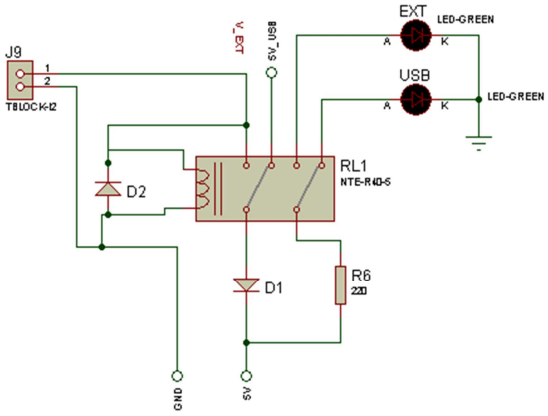


Figure III.7 : Bloque de commutation D'alimentation

III.2.1.2. Partie microcontrôleur

Le principal élément de ce bloque est le PIC18F2550 qui contiendra le programme de communication, conversion A/N et Générateur MLI (PWM).

Un connecteur « J4 » est relié aux entrées analogiques à travers des résistances de protection de 100Ω, qui servira à relier le ou les capteurs de températures analogiques.

Un autre connecteur « J1 » sera relié aux sorties logiques et PWM du PIC, qui commanderons la carte de Puissance.

Enfin un dernier connecteur « J5 » auquel nous assignons toutes les E/S non utilisées pour une utilisation future.

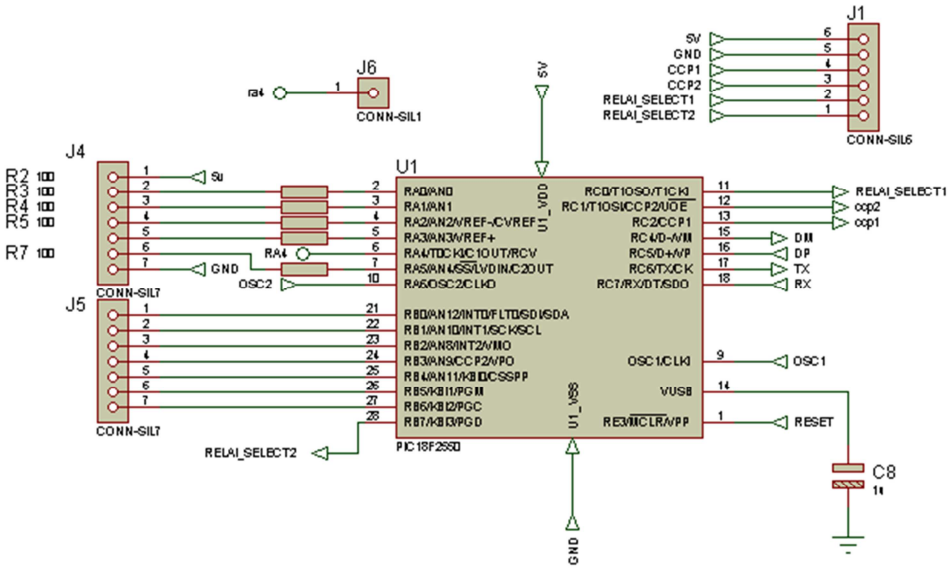
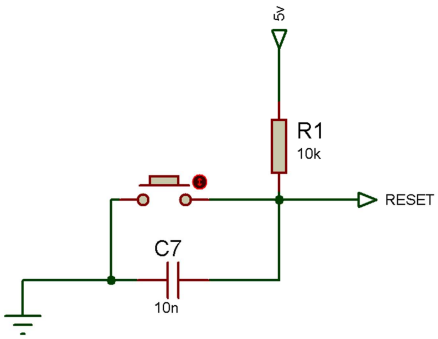


Figure III.8 : Bloque Microcontrôleur

- Une horloge à base de quartz 20 MHZ et deux capacités de 22pF est nécessaire au fonctionnement du PIC, relier aux broches « OSC1, OSC2 pates 9, 10». (Figure IV.10)
- Un circuit de réinitialisation (RESET) connecté à $\overline{\text{MCLR}}$ permet de réinitialiser (redémarrer) le système en cas de besoin. (Figure IV.9)



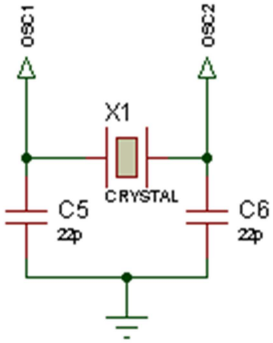


Figure III.9 : RESET

Figure III.10 : Horloge

III.2.1.3. Partie communication :

Cette partie est scindée en deux, la transmission RS-232 et USB, cela nous mène à concevoir de circuits de communication :

- **RS232 :**

Ce mode exige l'installation d'un circuit intègre nommé MAX 232 pour adapter les signaux normalisé de -25v à +25v du PC, vers des signaux TTL et inverser la logique de ces signaux -25V pour 1, +25 pour 0 adaptés au PIC.

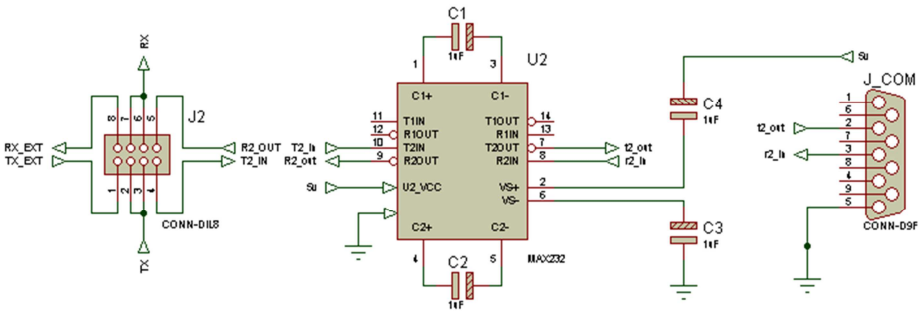


Figure III.11 : Circuit de transmission RS-232

- **USB :**

La mise en place d'une connexion USB avec le PIC18F2550 est très simple, le module USB est intégré dans ce dernier, donc il suffit juste de connecter les pates D+ et D- aux pins adéquates (D+ pin 16, D- pin 15).

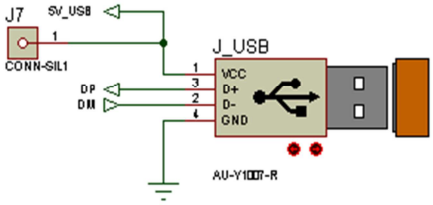


Figure III.12 : Connecteur USB

III.2.2. Carte de Puissance :

Elle effectue le rôle d'un amplificateur de puissance, et d'inverseur de sens du courant entrant dans le Peltier afin d'effectuer les actions chaud et froid.

Cette carte est composée de deux parties identiques comportant un relai et trois transistors pour chaque partie

- RL1-RL2 : Relai a deux commutateurs NO1 et NF1. NO2 et NF2.
- Q1- Q2 : Transistors TIP41 commandant le relai de commutation de sens du courant.
- Q1- Q3 : Transistor Darlington TIP122 afin de contrôler le transistor de puissance qui a son tour contrôle le Peltier en réaction au signale PWM.
- Q5-Q6 : Transistor de puissance 2N3773 qui a pour but la transformation des signaux PWM applique sure le Peltier.

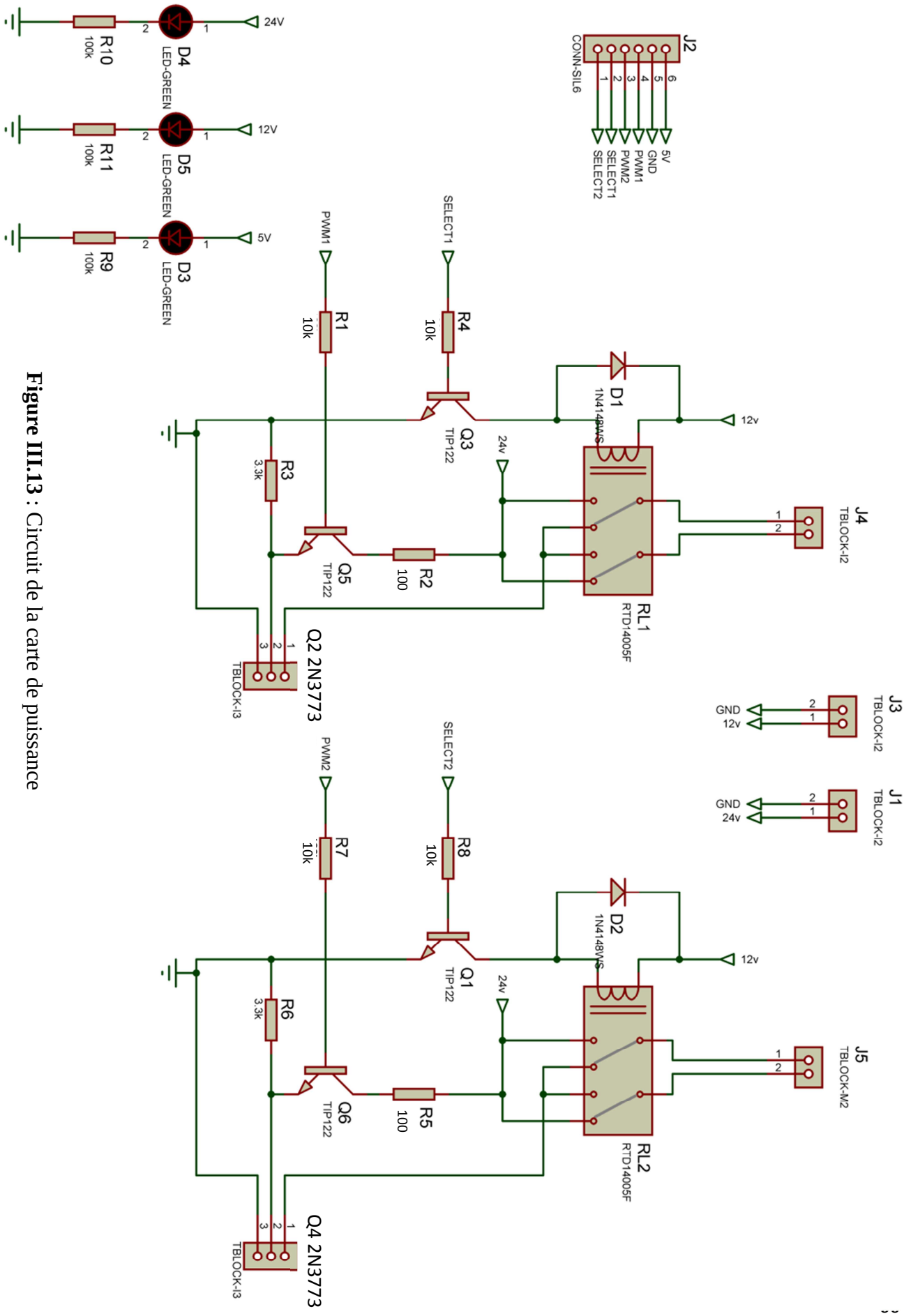


Figure III.13 : Circuit de la carte de puissance

La carte de puissance sera alimentée par deux tensions afin de commander le relai en 12V et le Peltier en 24V et des entrées de commande, pour se faire on a doté notre carte de quelque connecteur qui sont définit comme suit :

- J1: Alimentation de 24V pour alimenter les deux Peltier en puissance.
- J3: Alimentation de 12V pour les deux relais de commutation.
- J2: Connecteur raccordant cette carte à la carte de commande, avec une natte de 6 files :

1. SELECT2 : sélection du sens du courant dans le Peltier N2 en actionnant le relai RL2 Par le transistor Q1.

   0V sens directe « Chauffage ».

   5V sens inverse « refroidissement ».

2. SELECT1 : sélection du sens du courant dans le Peltier N1 en actionnant le relai RL1 Par le transistor Q3.

   0V sens directe « Chauffage ».

   5V sens inverse « refroidissement ».

3. PWM2 : entrée PWM2 dans la carte actionnant le Peltier 2.
4. PWM1 : entrée PWM2 dans la carte actionnant le Peltier 1.
5. GND : masse commune avec la carte de commande.
6. 5V : permet d'alimenter la LED D3 qui indique la connexion à la carte de commande.

- J4: connecteur permettant de relier le Peltier N1à la sortie de puissance.
- J5: connecteur permettant de relier le Peltier N2 à la sortie de puissance.

Nous ajoutons à cela, trois diodes lumineuses permettant d'indiquer l'alimentation en tension 24v et 12v et la connexion à la carte le commande :

- D3: indique la connexion à la carte de commande.
- D4: indique la connexion de la tension 24V.
- D5: indique la connexion de la tension 12V.

III.2.3. Conditionnement du capteur de température :

Pour l'utilisation du capteur de température, un circuit de conditionnement est nécessaire pour abaisser l'intensité du courant le traversant dans le but de réduire l'échauffement par effet Joule et de récupérer les données de température de manière précise.

Le circuit de conditionnement est un amplificateur opérationnel en mode « amplificateur non-inverseur » comme le montre la figure IV.5

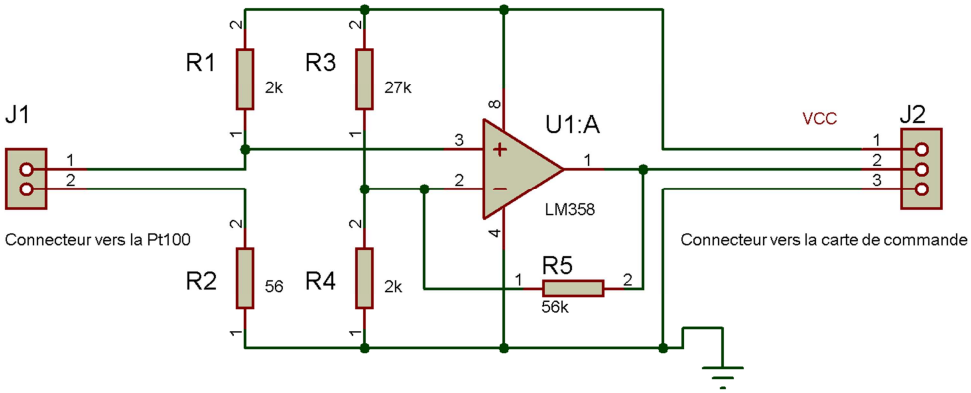


Figure III.14: Circuit de conditionnement du capteur de température Pt100.

Une foi le circuit de conditionnement réalisé, on calcule le gain de l'amplificateur :

Calcule de e^- :

Premièrement l'isolation du circuit (R3, R4, R5)

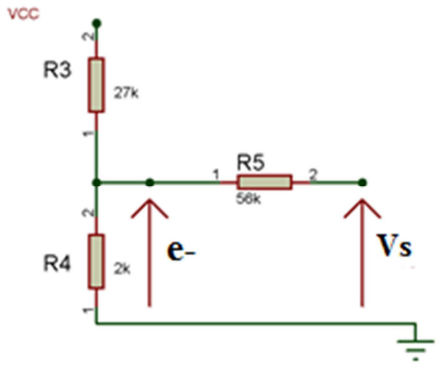


Figure III.15: isolation de l'entre e^-

$$e^- = \frac{Vs*Rtv}{Rtv+R5} + \frac{Vth*R5}{Rtv+R5} \qquad (4.1)$$

Avec :

$$\mathrm{Vth} = \frac{\mathrm{Vcc}*\mathrm{R4}}{\mathrm{R3}+\mathrm{R4}} \tag{4.2}$$

$$\mathrm{Rth} = \frac{\mathrm{R3}*\mathrm{R4}}{\mathrm{R3}+\mathrm{R4}} \tag{4.3}$$

Calcule de e^+:

Premièrement l'isolation du circuit (R3, R4, R5)

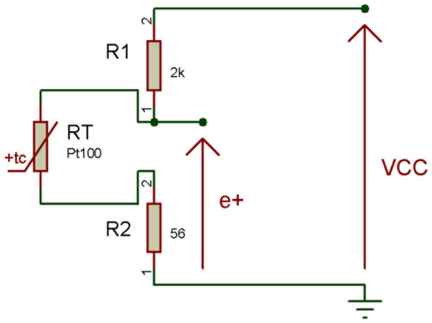


Figure III.16: isolation de l'entre e^+

$$e^+ = \mathrm{Vcc} * \frac{\mathrm{R2}*\mathrm{RT}}{\mathrm{R1}+\mathrm{R2}+\mathrm{RT}} \tag{4.4}$$

Avec :

RT: Le capteur de température Pt100

Vcc : La tension d'alimentation

En posante$^- = e^+$ nous obtenons:

$$\mathrm{Vs} = \mathrm{Vcc} * \frac{\frac{\mathrm{R2}+\mathrm{RT}}{\mathrm{R1}+\mathrm{R2}+\mathrm{RT}} - \mathrm{C1}}{\mathrm{C2}} \tag{4.5}$$

Avec :

$$\mathrm{C1} = \frac{\mathrm{R4}}{\mathrm{R3}+\mathrm{R4}} * \frac{\mathrm{R5}}{\frac{\mathrm{R3}*\mathrm{R4}}{\mathrm{R3}+\mathrm{R4}}+\mathrm{R5}} \tag{4.6}$$

$$\mathrm{C2} = \frac{\frac{\mathrm{R3}*\mathrm{R4}}{\mathrm{R3}+\mathrm{R4}}}{\frac{\mathrm{R3}*\mathrm{R4}}{\mathrm{R3}+\mathrm{R4}}+\mathrm{R5}} \tag{4.7}$$

Des expressions 4.5, 4.6, 4.7 nous obtenons :

$$Vs = Vcc * \frac{\frac{56+RT}{2056+RT} - 0{,}067}{0{,}032} \qquad (4.8)$$

En utilisant la fonction 4.8, nous obtenons la courbe définissant la variation de la tension Vs en fonction de la résistance RT :

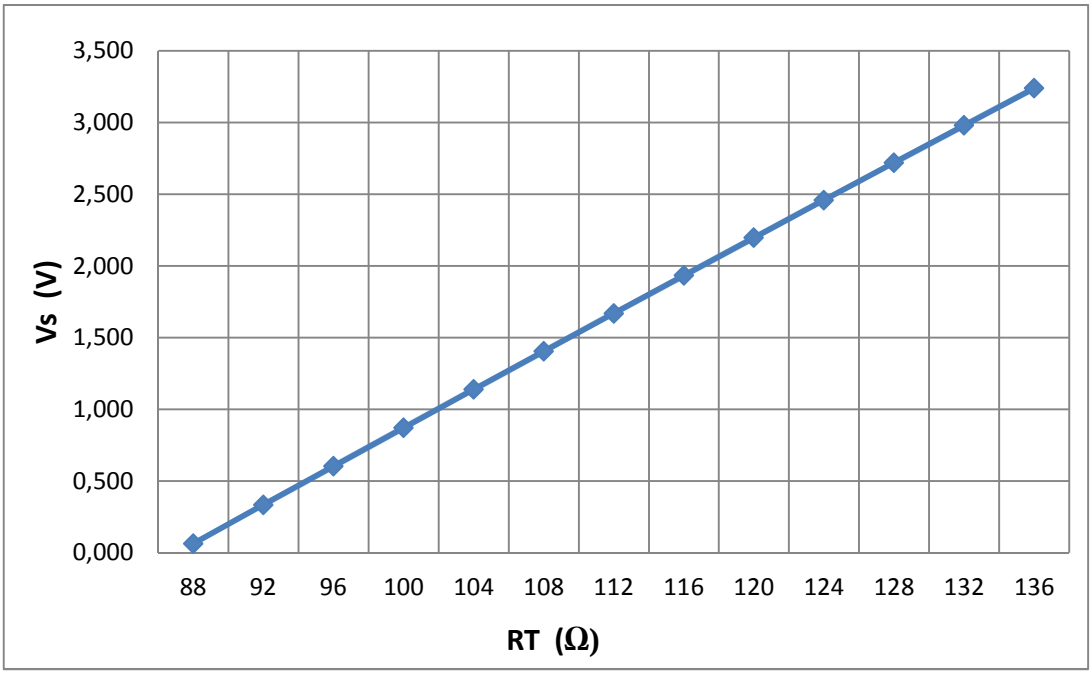


Figure III.17: Vs en fonction de RT

Nous observons sur la figure précédente que l'évolution du signale de sortie en fonction de la résistance du capteur est linéaire, et nous savons que la caractéristique de la résistance Pt100 en fonction de la température est elle aussi linéaire (voir chapitre II, figure II.1), que nous pouvons approchez par l'équation suivante:

$$T = 2{,}591(RT - 92{,}160) - 20 \qquad (4.9)$$

Nous remplaçons les valeurs de la résistance RT par les valeurs de la température obtenue par l'équation (4.9) dans la figure IV.8 et nous obtenant l'allure de la figure suivante :

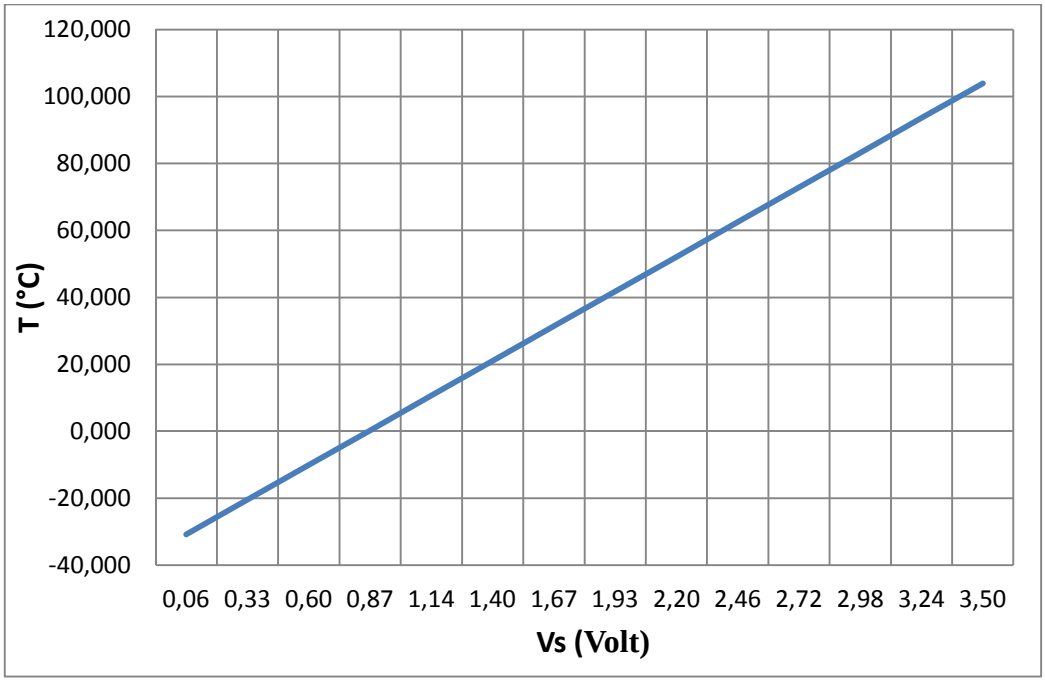


Figure III.18: température en fonction de Vs

De la figure précédente nous pouvons approcher la fonction de la température en fonction de la tension de sortie du conditionneur, donc l'équation T(VS) est comme suit :

$$T = 39{,}258 * (Vs - 0{,}065) - 30{,}779 \qquad (4.10)$$

Après remplacement des tensions par leurs valeurs numériques adaptées dans l'équation précédent (4.9), la fonction devient :

$$T = 0{,}192 * (digit - 13) - 30{,}78 \qquad (4.11)$$

III.2.3. Disposition globale des parties du projet

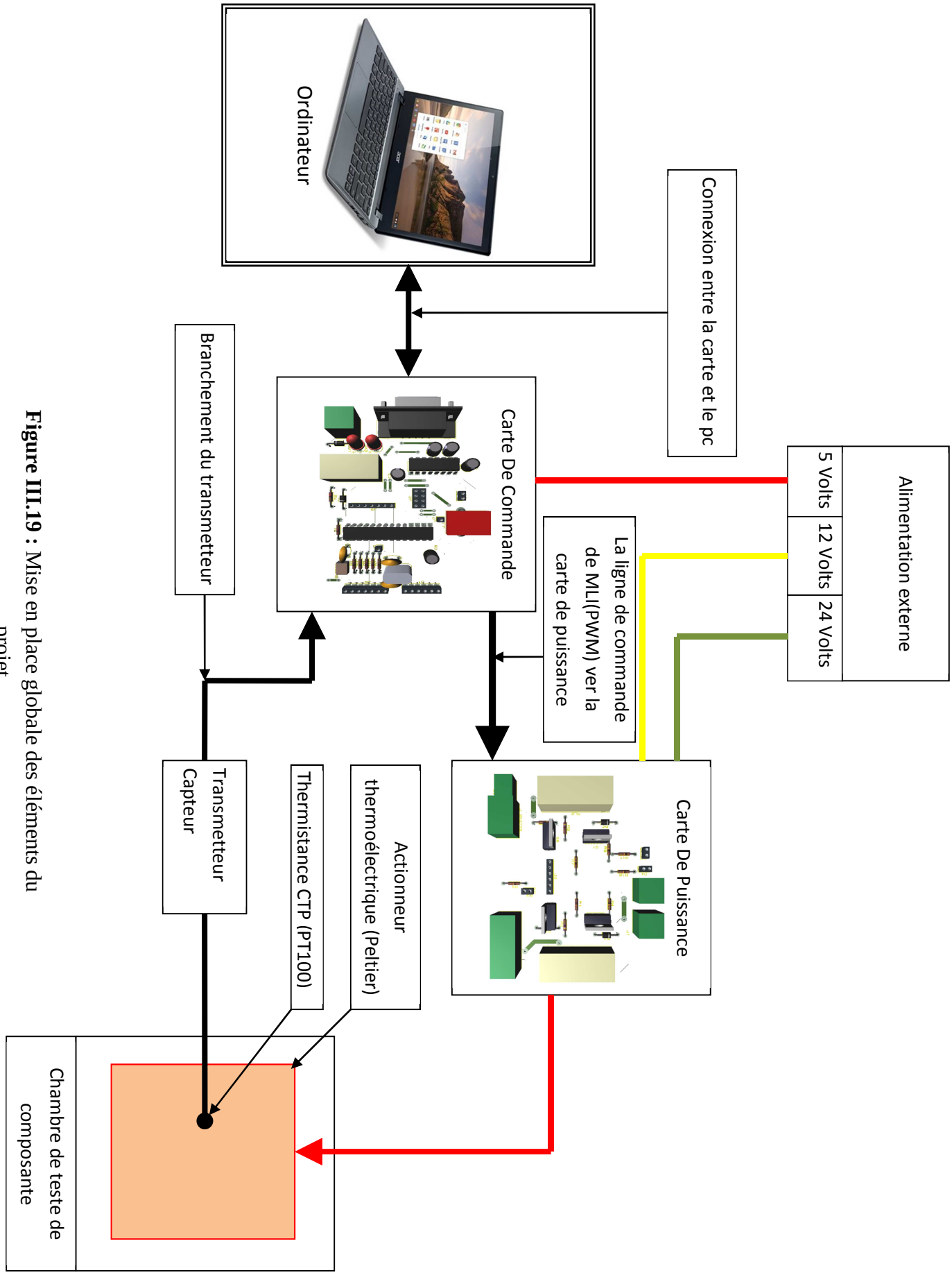


Figure III.19 : Mise en place globale des éléments du projet

IV. Simulation du fonctionnement de la transmission et la conversion A/N

Après la conception des cartes et avant d'entamer la réalisation, nous devons simuler les fonctions de la carte de commande, la transmission série et la conversion analogique numérique.

La simulation se fait avec le logicielle LABCENTER ELECTRONIQUE ISIS PROTEUSE.

Afin de teste la liaison et la configuration du PIC, nous avons conçu un programme qui recueille la chaine de caractères « AN 00 » et renvoi la valeur de l'entre analogique sous forme de chaine de caractères formaté sur 6 caractères.

IV.1. Programme de teste:

```
void main (void)
{                   TRISB  =      0xff          ;
                    TRISB  =      0             ;
                    TRISA  =      0xff          ;
                    TRISC  =      0b10000000    ; //PORTC(0-6) en sortie RC7 en entre
                    PORTA  =      0             ;
                    PORTB  =      0             ;
                    SPBRG =       129           ; // (129==> 9600 bits/s)
                    TXSTA =       0x24          ; //8 bits pas de bit erreur
                    RCSTA =       0x90          ; // active l'USART
      WHILE (1)
      {
            GETSUSART ((char *) ch, 6);      lire la commande reçue
            DELAY10TCYX (10)                 ;      marqué un temps d'arrêt
if (((ch[0]=='A')|(ch[0]=='a'))&((ch[1]=='N')|(ch[1]=='n'))&(ch[2]==' '))
            {     A      =      0                    ; initialisé A à 0
                  A      =      get_adc(ADC_CH0)     ; récupérer la valeur de AN0
                  PRINTF        ("%06d", A)          ; envoyer la valeur de AN0
            }     ch[0]=0x00;
      }
}
int get_adc(char chanel)
{
OpenADC( ADC_FOSC_32 & ADC_RIGHT_JUST &ADC_5ANA,chanel &ADC_INT_OFF, 15 );
//ouvrir le module ADC avec 5 entres analogique int adc off
      SETCHANADC   (chanel)     ; //sélectionner l'entré a convertir
      DELAY10TCYX (5)           ; //délai d'acquisition
      CONVERTADC   ()           ; //lancer la conversion
      WHILE        (BUSYADC ()); //attendre la fin de la conversion
      RETURN (READADC ()); //renvoyer le résultat de la conversion
      CLOSEADC (0)        ; //fermer le module de conversion analogique
}
```

Une fois le fichier .Hex généré, ce dernier sera chargé dans le PIC pour la simulation sous PROTEUS.

IV.2. Circuit de teste:

Nous commençons par l'insertion du composant COMPIM qui permet de simuler une liaison série, puis le PIC18F2550 et un rhéostat en cliquant sur P et de saisir le nom a la zone de recherche.

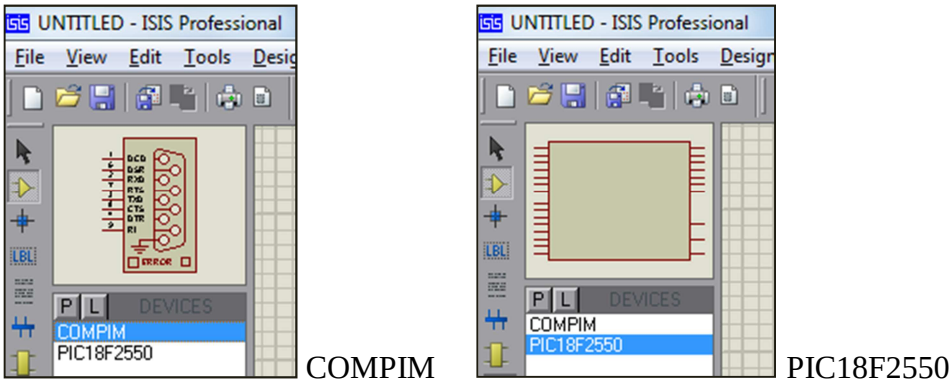


Figure III.20 : Composants sous PROTEUS

COMPIM : c'est un composant qui permet de relier le circuit à simuler au PORT COM avec une liaison virtuelle, et de ce PORT à un autre ordinateur, ce qui fait passer l'ordinateur ayant le PROTEUS pour une carte électronique.

Sa configuration se fait après double clic sur le composant est comme suit :

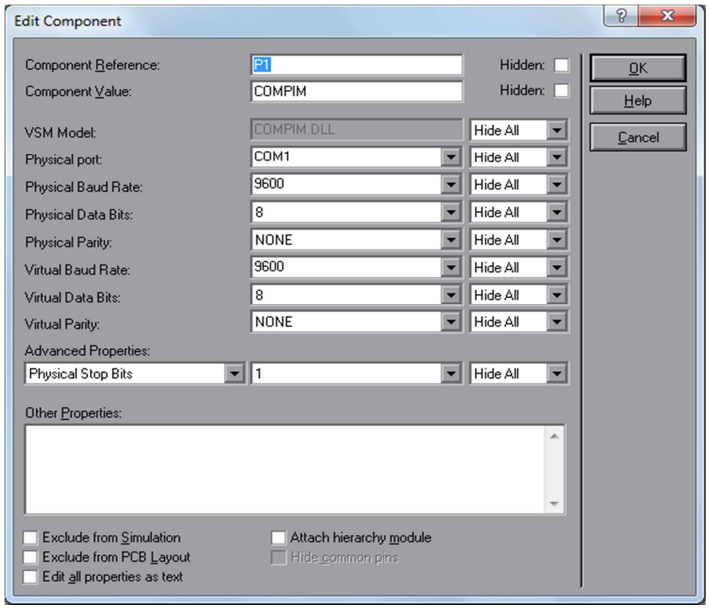


Figure III.21 : Configuration de COMPIM

Description de la fiche :

- **Physical port**: choix du port physique a travers le quel se connecter.
  COM1.
- **Physical baud rate** : choix de la vitesse physique de transmission
  9600.
- **Physical data bits** : choix du nombre de bits de donnés
  8 bits
- **Physical parity** : choisir la parité qui servira au contrôle du flux.
  NONE (aucune)
- **Virtuel baud rate** : choix de la vitesse virtuel de transmission
  9600.
- **Virtuel data bits** : choix du nombre de bits de donnes virtuels
  8 bits
- **Virtuel parity** : choix de la parité virtuelle.
  NONE (aucune)

Le rhéostat : Est une résistance variable a trois pates qui nous permet de réalise un deviseur de tension afin de faire varier la tension de l'entre analogique AN0 sur RA0 de 0v a 5v pour simuler un capteur de température ceci en réalisant le circuit suivant :

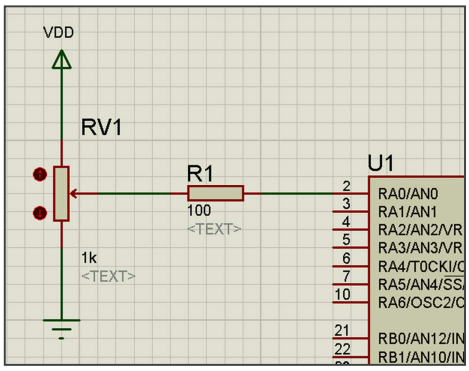


Figure III.22 : simulation d'un capteur analogique

PIC18F2550 : C'est la représentation du pic dans le PROTEUS, ce dernier nous permet de charger le programme du pic et de le configurer comme suit :

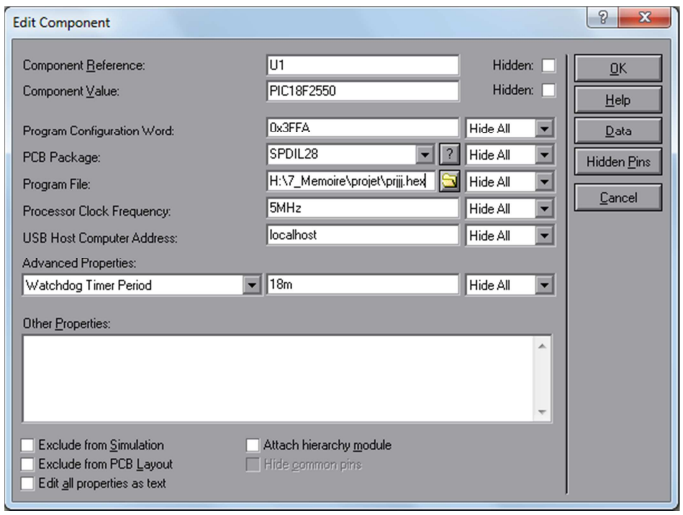


Figure III.23 : paramétrage du PIC18F2550

Dans cette fenêtre nous allons nous intéresser à quelques champs, Programme File et Processor Clock frequency :

Programme File : ce champ nous permet de charger le programme du pic sous forme de fichier Hexadécimal (.HEX).

Processor Clock frequency : ce champ nous permet de modifier la fréquence de l'horloge du **PIC.**

Une fois que tous les composants sont insères, nous les connectons de la manier suivante :

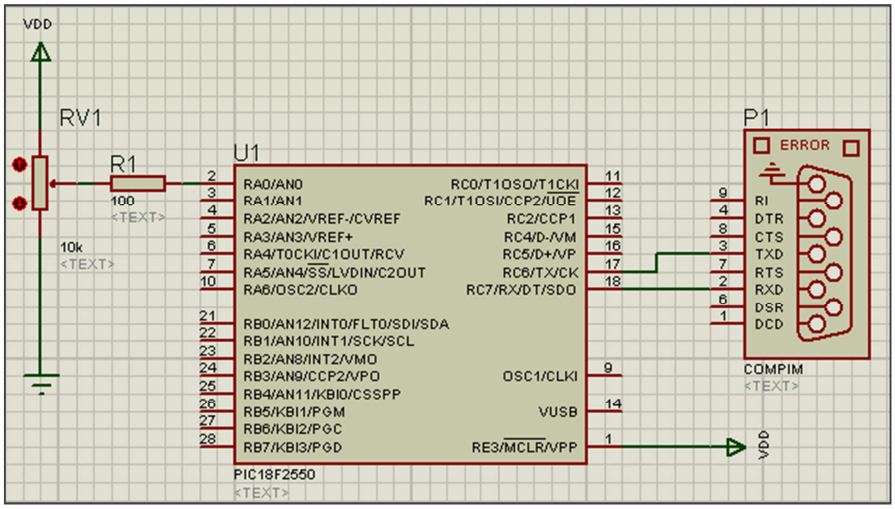
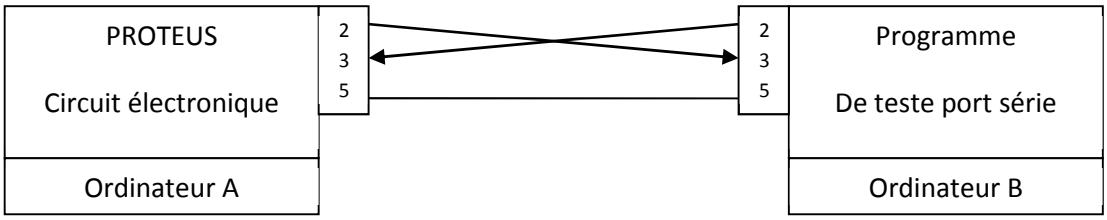


Figure III.24 : Circuit de simulation

Maintenant il ne reste qu'à relier les deux PC par un câble RS-232, croisé de la manière à avoir une connexion full-duplex, comme dans la figure suivante.



Après la mise en place du câble, il ne reste qu'à lancé la simulation de PROTEUSE dans l'ordinateur A, et en même temps, exécuter l'application de test de la liaison RS-232 dans l'ordinateur B.

On aura la fenêtre suivante après exécution de l'application de teste du port COM :

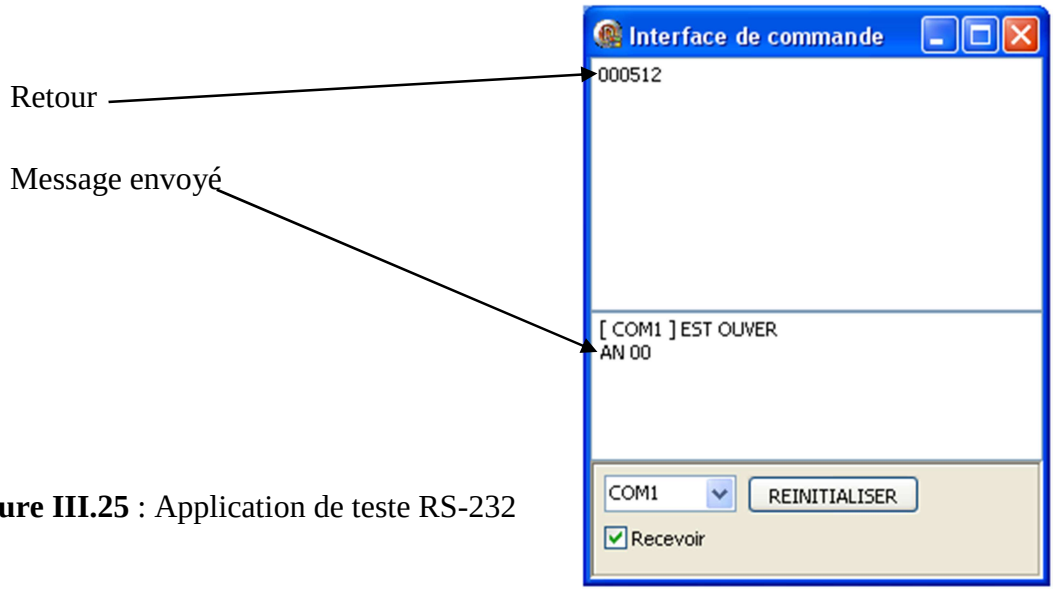


Figure III.25 : Application de teste RS-232

Nous constatons que lors de l'envoi du message « AN 00 », nous obtenons la valeur de l'entre analogique AN0, qui est égale à 512, cela signifie que la démarche adopté pour programme de transmission est fonctionnelle, de même pour le programme de conversion analogique numérique.

V. Conclusion :

Dans ce chapitre nous avons présenté la stratégie choisie pour réaliser le système de régulation, nous avons mis en lumière l'organisation du des programmes et des cartes, puis nous les avons testés pour aboutir à la mise en œuvre finale du procédé que nous traiterons dans le chapitre suivant.

Chapitre 4

Réalisation et identification du système

I. Introduction

Ce chapitre est dédier à la réalisation du système, en effet après conception et simulation du système, il ne reste qu'à entamer sa réalisation, où nous mettrons en application les différentes méthodes et programmes des chapitres précédents.

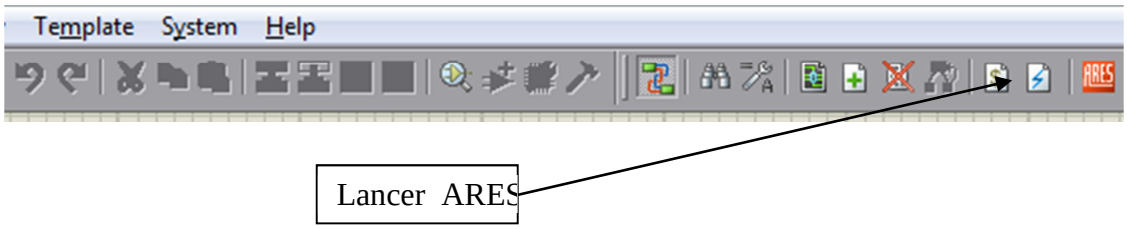
Il est divisé en deux parties, la première partie est consacrée à la réalisation du système, cartes et programmes (hardware et software), et la seconde, à l'identification des paramètres du régulateur PID à appliquer au procédé.

II. Partie réalisation et programmation

II.1 Réalisation des cartes

La réalisation de ces cartes se fait avec LABCENTER ELECTRONIQUE ARES PROTEUSE.

ARES est une application qui permet de modéliser un typon à partir du circuit déjà conçue avec ISIS.



Une fois ARES lancé, nous introduisons les composant et les relions d'une manière esthétique et moins encombrante afin de minimiser le plus possible la taille des cartes, et nous obtenons ces schémas :

II.1.1. Carte de command

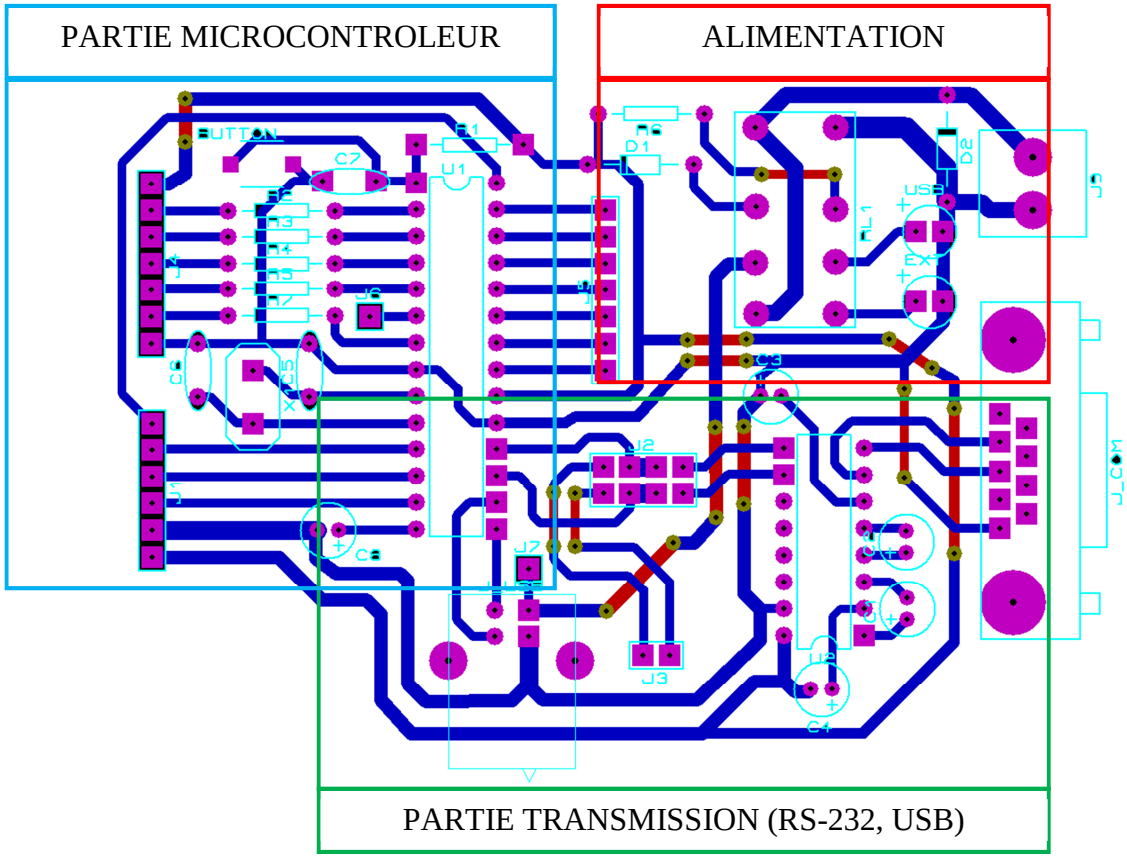


Figure IV.1 : circuit de la carte de commande

Nous retrouvons les trois parties de la conception ; le rectangle rouge représente l'alimentation, le rectangle vert le circuit de transmission RS-232 et USB et enfin en bleu la partie microcontrôleur.

Nous pouvons remarquer la proximité des fils, cela n'est pas fortuit mais obéit à des considérations esthétiques et économiques ; puisque cette carte ne fonctionnera pas avec de grandes puissances, il n'est pas nécessaire d'utiliser de grosses pistes, ce qui nous permet de minimiser sa taille et le coup de la réalisation.

Typon de la carte de commande :

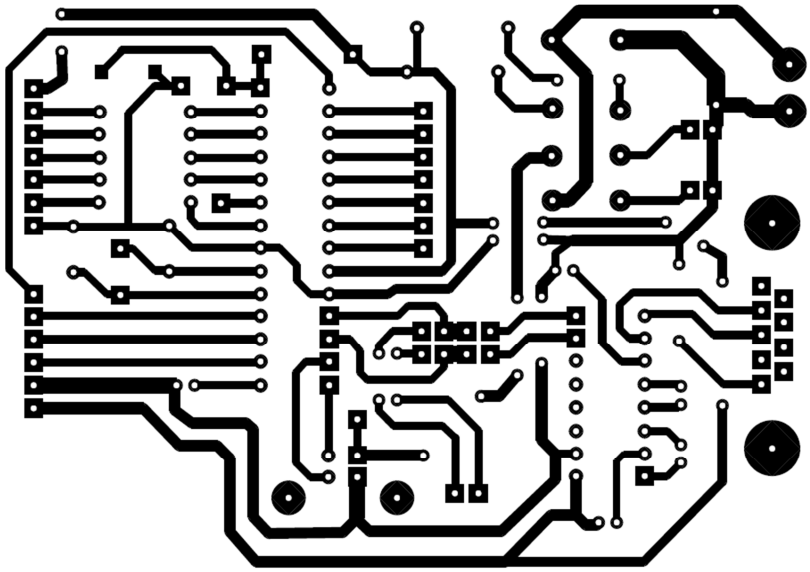
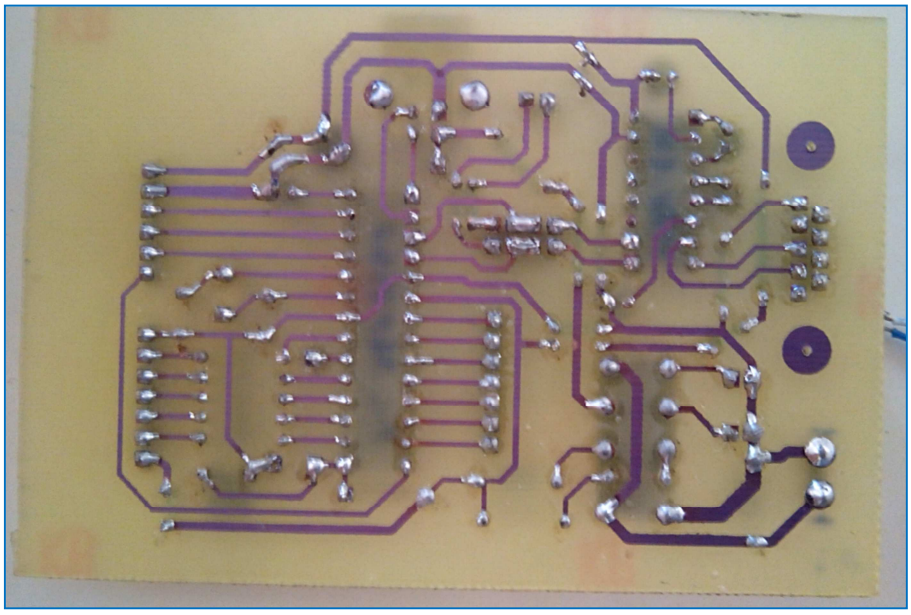


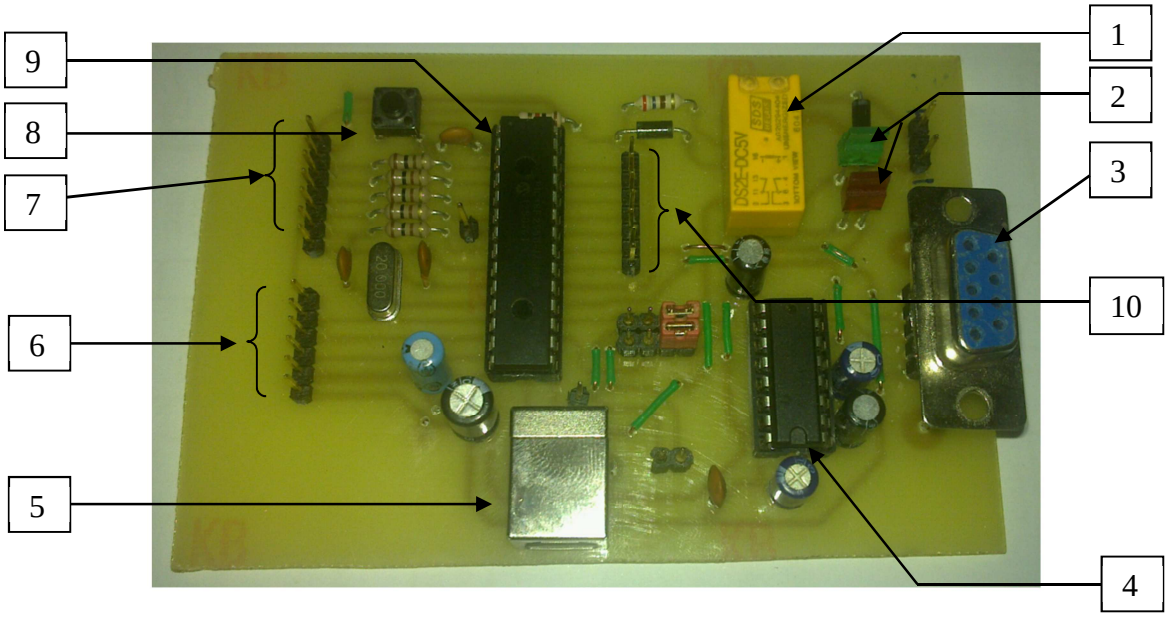
Figure IV.2: typon carte de commande

Une fois les typons imprimés sur un support transparent donnés sur les figures précédentes, vient la réalisation des deux cartes comme le présente les vues IV.1, IV.2, IV.3 IV.4.



Vue IV.1: carte imprimé du circuit de commande

Après cela, nous placerons les composants et nous les soudons à leur place.



Vue IV.2: carte du circuit de commande finalisé

Eléments de la carte de commande :

1- Relais 5V

2- LEDs d'alimentation

3- Connecteur RS232

4- MAX232

5- Connecteur USB B

6- Connexions vers 2nd carte

7- Entrées Analogiques

8- Bouton de réinitialisation

9- PIC18f2550

10- Autres Connexions non-utilisés

II.1.2. Carte de puissance

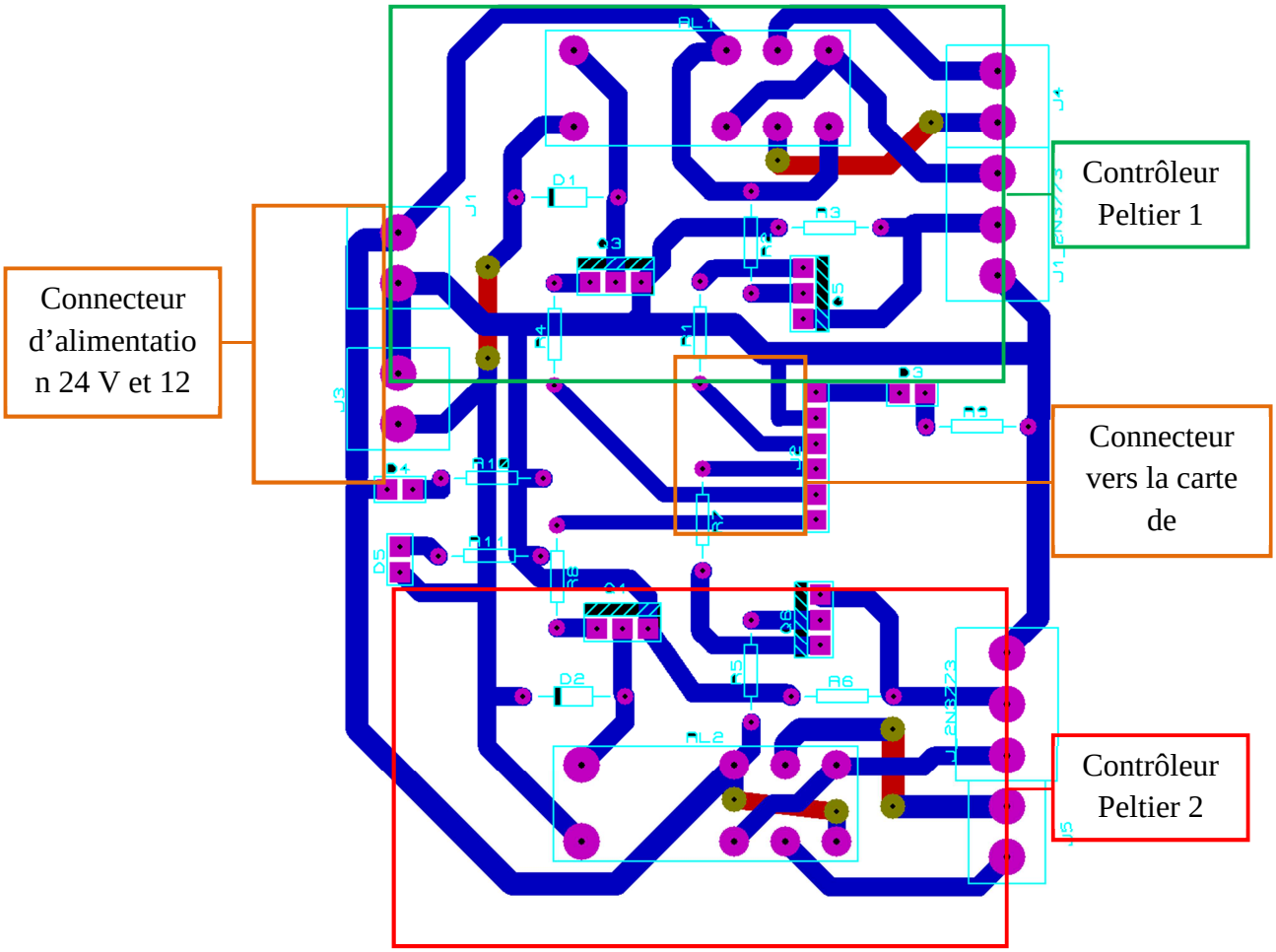


Figure IV.3 : circuit de la carte de puissance

Contrairement à la première carte, celle-ci devra supporter une puissance allant jusqu'à 240W (24V/10A), cela nous ramène à utiliser de plus grosses pistes sur le schéma et des files de renforcement sur la carte finale.

Typon de la carte de puissance

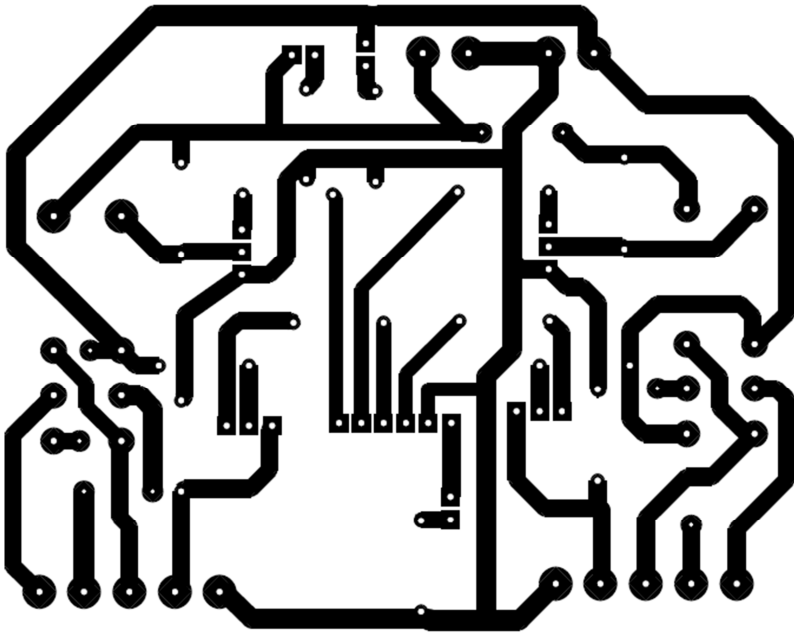
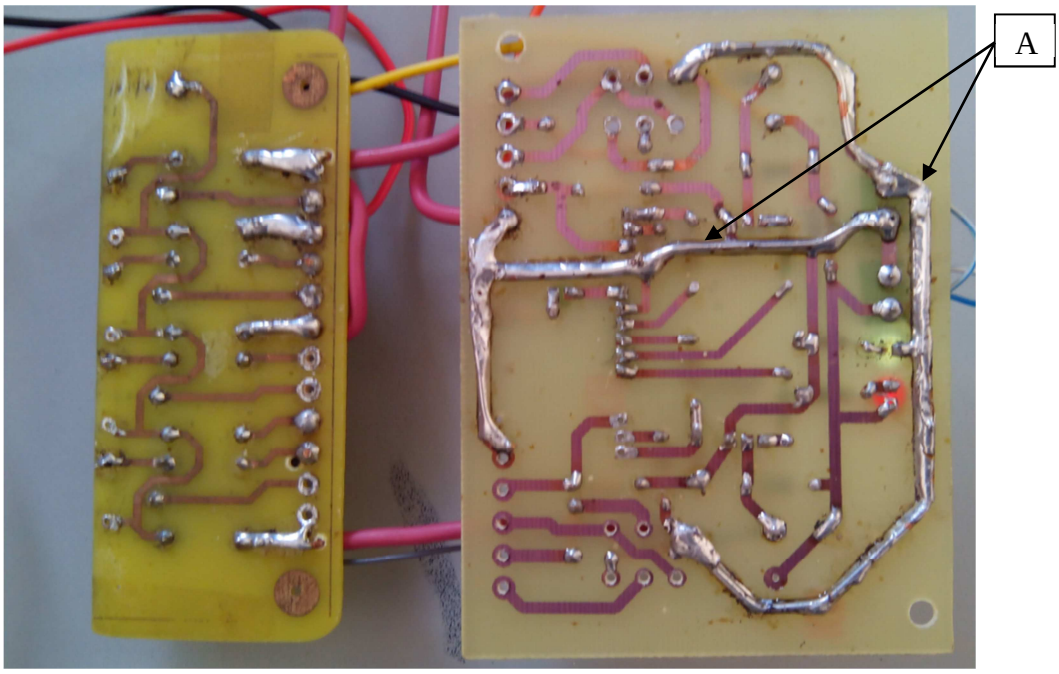
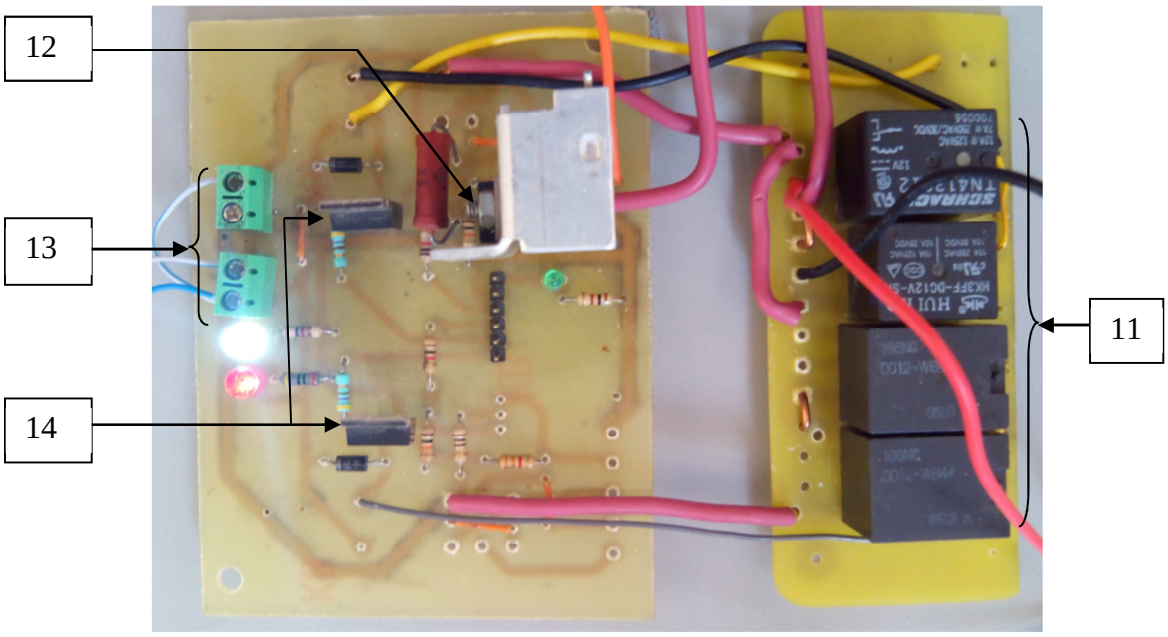


Figure IV.4: typon carte de puissance



Vue IV.3: carte imprimé de puissance

La flèche (A) dans la vue précédente, indique les pistes renforcées avec une file 1,5 mm rigide dans le but de supporter la puissance consommée par le Peltier.



Vue IV.4: carte de puissance finalise

Composants de la carte de puissance **:**

11- Relais de commutation du sens de courant.

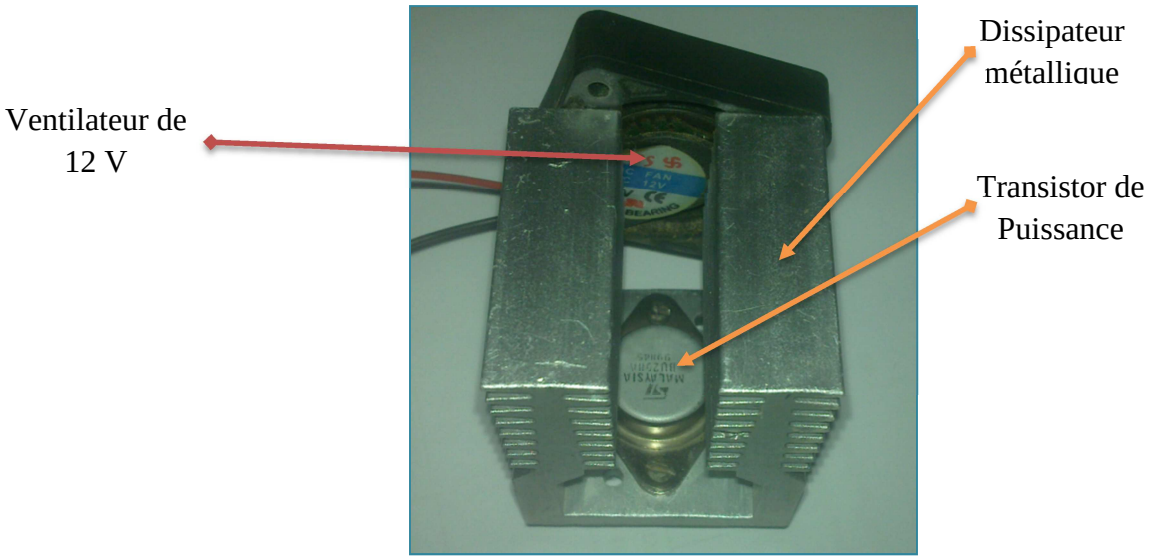
12- Transistor commandant le transistor de puissance.

13- Alimentations 24V et 12V.

14- Transistors de commutation des relais.

II.1.3. Transistor de puissance 2n3773

Le Peltier fonctionnant avec une puissance de 240W, nous placerons un transistor pouvant supporter cette puissance, il est mis sur un dissipateur de chaleur, pour éviter la surchauffe et la perturbation du système.



Vue IV.5: Transistor de puissance avec son dissipateur et le ventilateur

II.2 Réalisation des programmes

III.2.1. Programme du microcontrôleur

La programmation du PIC se fait avec le logiciel MPLAB avec un compilateur MCC18, de ce fait, le programme sera rédigé avec le code assembleur du PIC, et le langage « C ». Le programme implémenté sur le PIC fait office de programme de communication, programme d'échantillonnage de signal analogique et d'un générateur de tensions en signal modulé.

Il faut toutefois prendre en compte que ce programme sera en total synchronisation avec le programme sous LabVIEW, donc il faudra respecter une certaine structure de communication (envoi➔ réception➔renvoi d'un accusé de réception), et cette structure est appliquée par les deux programmes en marche.

Le programme fonctionne comme suite :

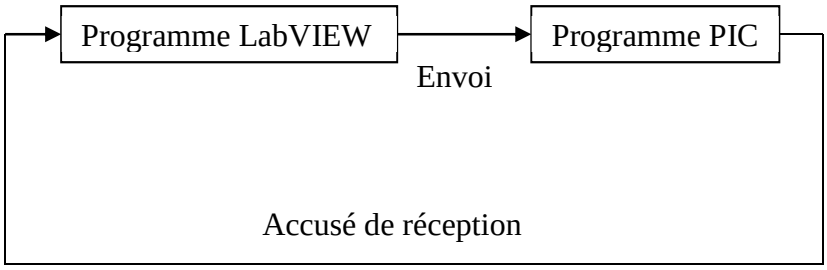
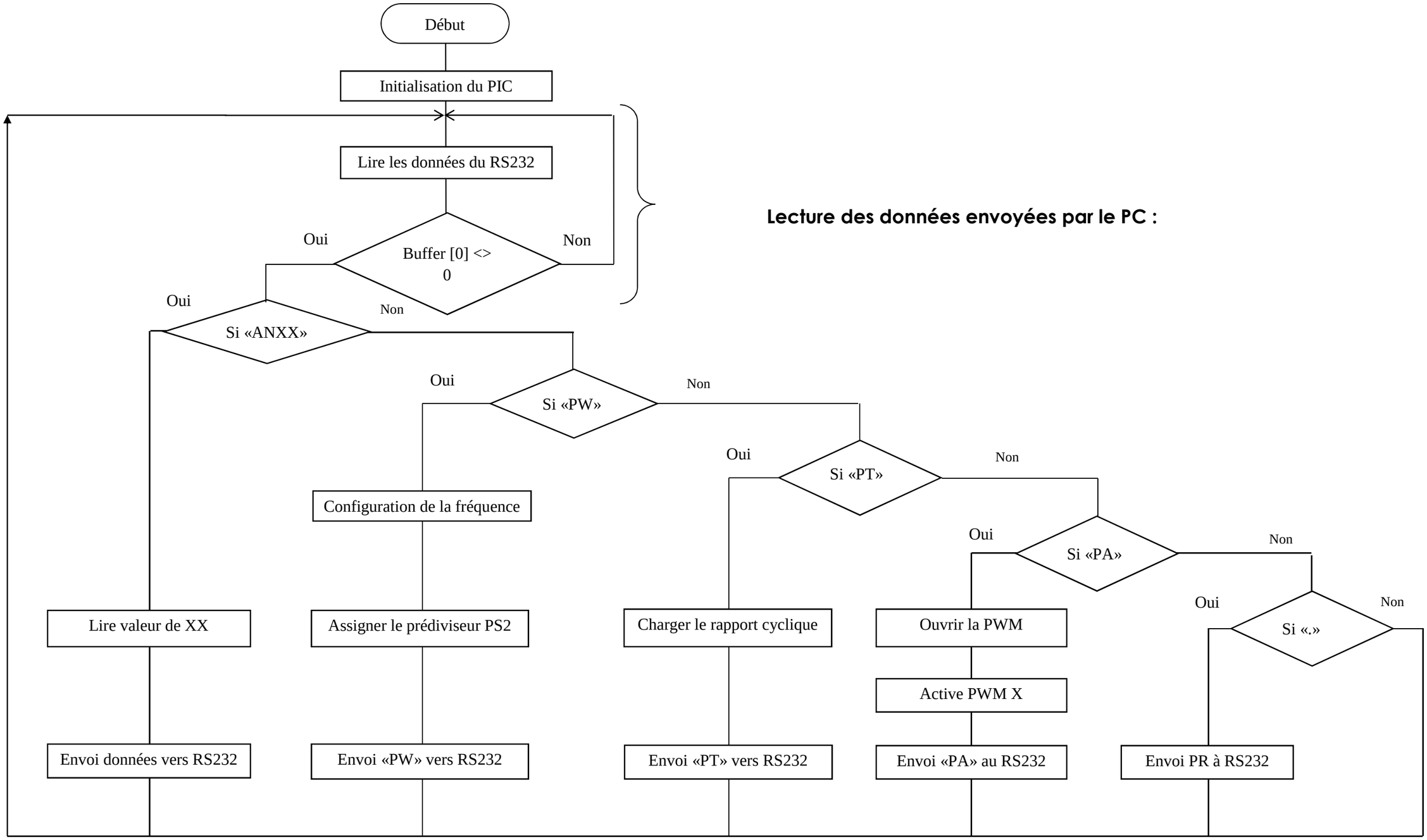


Figure IV.5 : illustration d'un message et de son accusé entre le programme LabVIEW et le Programme PIC

Organigramme du programme du microcontrôleur



Lecture des données envoyées par le PC :

Figure IV.6 : fonctionnement du programme du microcontrôleur

Liste des commandes et leurs procédures

Initialisation des paramètres du PIC :

Dans cette partie, nous avons configuré les E/S et mis en place les multiples variables devant être utilisés :

```
TRISB=0xff;
TRISA=0xff;
TRISC=0b10000000;
PORTA=0;
PORTB=0x80;
a=100;
ps=0;
i=0;
SPBRG =  129        ;
TXSTA = 0x24        ;
RCSTA = 0x90        ;
```

Lecture des données envoyées par le PC :

Cette étape lit les données envoyée par le PC, ensuite les compare aux commandes préprogrammées.

Si la condition est atteinte, le programme passe à la vérification des dites commandes préprogrammées, puis exécute les procédures adéquates à ces commandes.

```
while (1)
        {       ch[0]=0;
                //ch=ReadUSART();
                getsUSART((char *)ch,6 );
if (ch[0]!=0)   {
                //Sous programmes …
                }
        }
```

1. **Vérification de connexion :** ce message vérifie s'il y a connexion entre la carte et le PC :

```
if (ch[0]=='.')
printf ("PR");// renvoi « PR » en cas de connexion
```

2. **Lecture d'une entrée analogique :** Si la commande ANXX est lue, le PIC lit l'entrée analogique précisée par les deux octets XX et renvoi la valeur de c'celle-ci:

```
if ((ch[0]=='A')&(ch[1]=='N'))
{
b=0;
b=(ch[2]-0x30)*10+(ch[3]-0x30);
a=0;
        switch (b)
        {
        case 0:{a=get_adc(ADC_CH0); printf ("%06d",a); break; }
        case 1:{a=get_adc(ADC_CH1); printf ("%06d",a); break; }
        case 2:{a=get_adc(ADC_CH2); printf ("%06d",a); break; }
        case 3:{a=get_adc(ADC_CH3); printf ("%06d",a); break; }
        case 4:{a=get_adc(ADC_CH4); printf ("%06d",a); break; }
        }
}
```

- La procédure **get_adc(chanel)**

```
intget_adc(char chanel)
{

OpenADC( ADC_FOSC_32& ADC_RIGHT_JUST &ADC_6_TAD & ADC_5ANA, Chanel & ADC_INT_OFF, 5  );
SetChanADC(chanel);
Delay10TCYx(100
ConvertADC();
while(BusyADC() );
return(ReadADC());
closeADC(0);

}
```

3. **Configuration de la fréquence de la PWM et du prédiviseur du TIMER2 :**

Si la commande PWXXXS le pic assigne la fréquence et le prédiviseur PS2 et renvoi « PW »au PC:

```
if ((ch[0]=='P')&(ch[1]=='W'))
        {
            psc=1;
            if (ch[5]=='0'){    psc=1  ;    T2CON=0b00000100;}
            if (ch[5]=='1'){    psc=4  ;    T2CON=0b00000101;}
            if (ch[5]=='2'){    psc=16;     T2CON=0b00000110;}
a=(int)(ch[2]-0x30)*(int)1000+(int)(ch[3]-0x30)*(int)100+(int)(ch[4]-0x30)*(int)10;
            freq_1=((char)((500000/(a*psc))))-1;
            printf (''PW'');
        }
```

4. Validation du rapport cyclique :

La réception du message PTXXXX permet de valider le rapport cyclique du module PWM en suivant la valeur accompagnant ce message, et renvoi le message «PT » :

```
if ((ch[0]=='P')&(ch[1]=='T'))
{
    if (ch[2]=='-')
    {       PORTCbits.RC0=1;        ch[2]='0';      }
    else    {       PORTCbits.RC0=0;                }
    D_Cycl=(int)(ch[2]-0x30)*(int)1000+(int)(ch[3]-0x30)*(int)100+(int)(ch[4]-
    0x30)*(int)10+(int)(ch[5]-0x30);
printf (''PT'');
}
```

5. **Activation du module PWM :**

Le message PAXE permet l'activation d'un ou des deux modules des modules

PWM du PIC l'octet « X » permet le choix du module, l'octet « E » permet l'activation ou la désactivation du module choisi par « X »

II.2.2. Programme sous labVIEW

Organigramme du programme LabVIEW

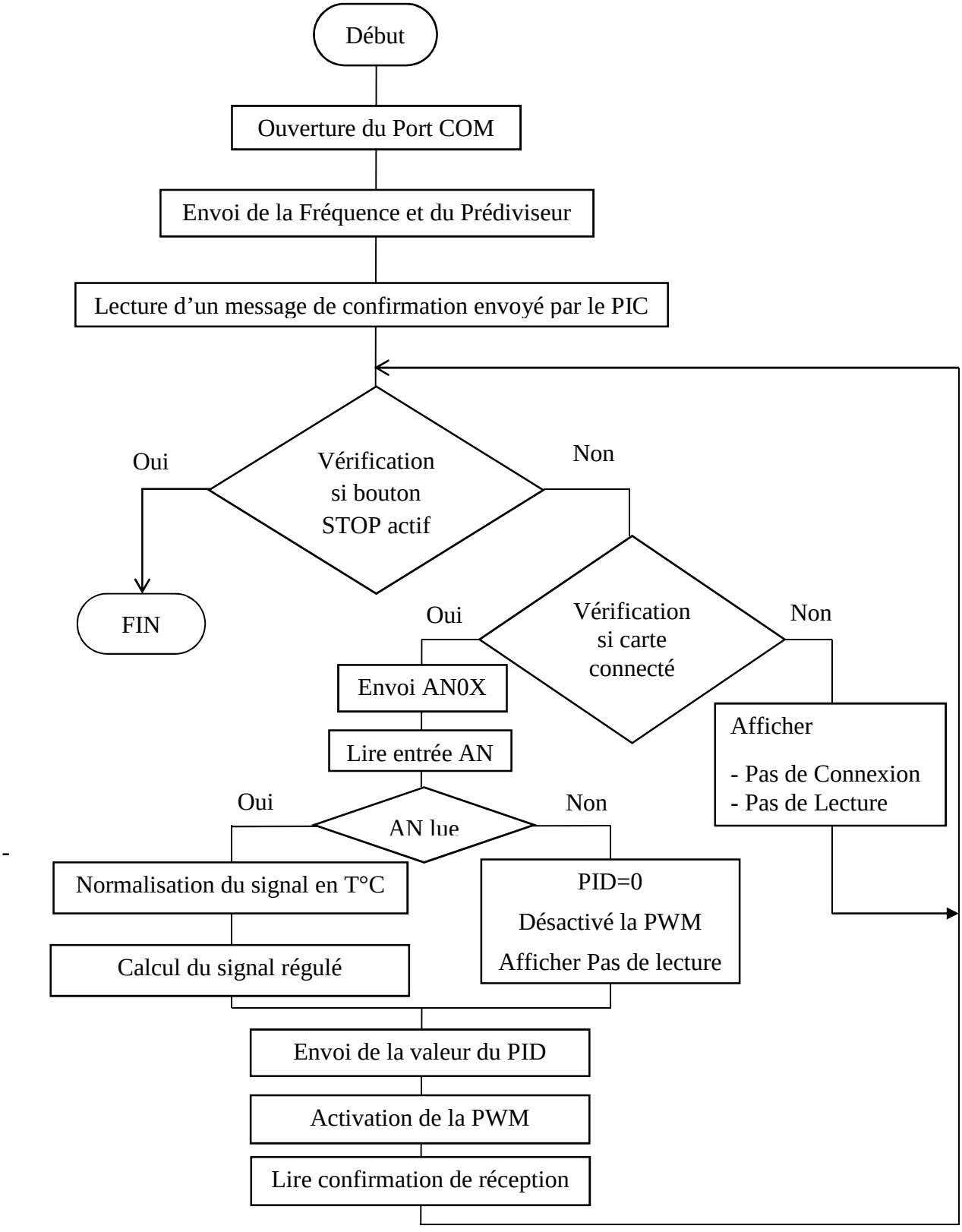


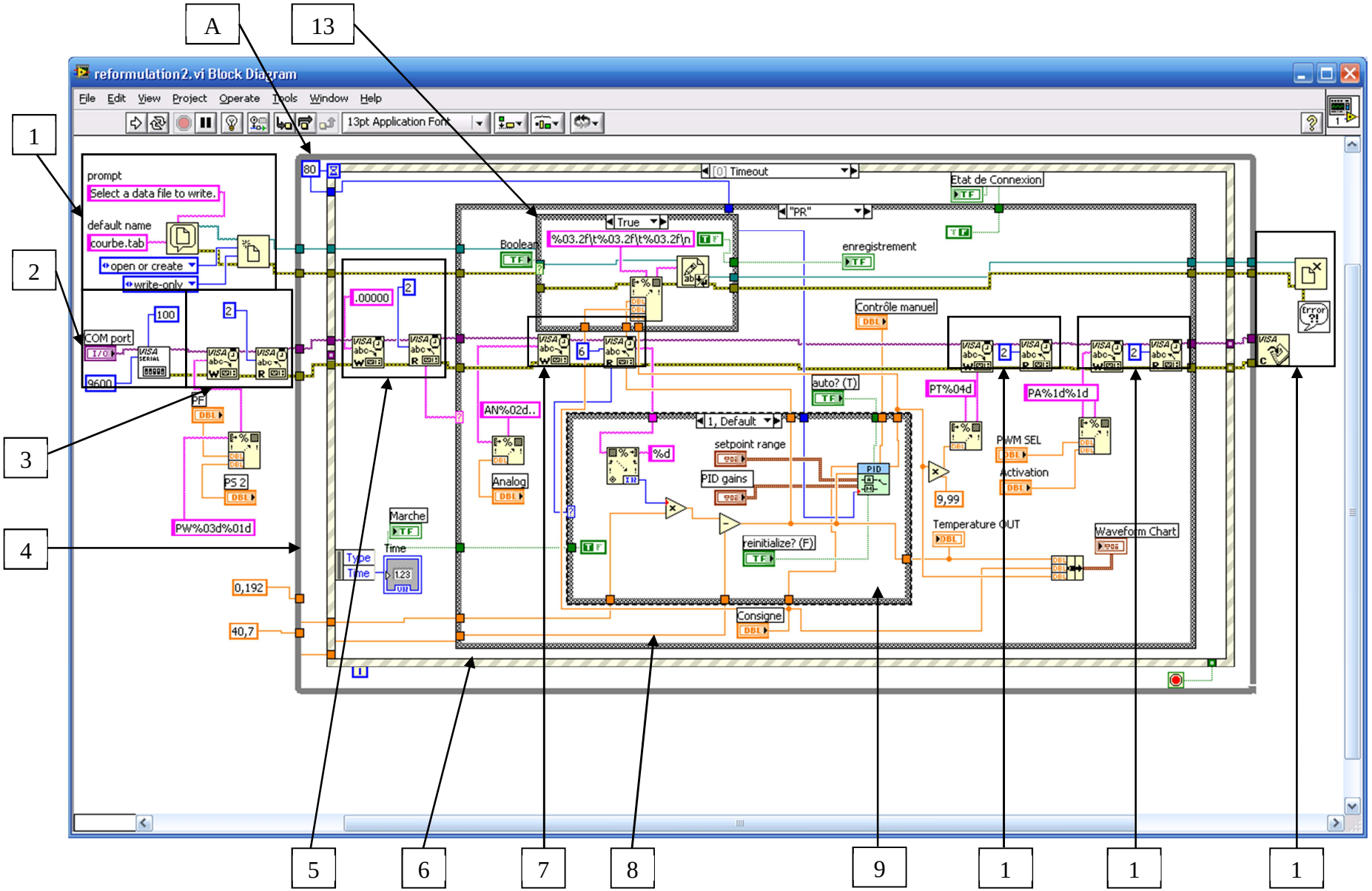
Figure IV.7 : fonctionnement du programme de labVIEW

Explication de l'organigramme :

Pour avoir une réception continue des données, le programme sous LabVIEW est synchronisé avec le programme du PIC, tel que le nombre d'octets envoyé est limité à 6 caractères, à cela s'ajoute la lecture d'une donnée de confirmation envoyée par le PIC, et pour cela nous avons choisi de coordonner le programme comme suite :

- La communication est faite via port COM, le programme commence par l'ouverture de ce port, tout en configurant ses paramètres.
- L'envoie des valeurs voulues de la fréquence et le prédiviseur au PIC, tout en lisant une confirmation de réception.
- L'envoie d'un autre message dans le but de vérifier la connexion entre la carte et le PIC (un point «.» suivit de 5 zéros pour atteindre 6 caractères).
- La lecture d'une réponse envoyée par le PIC, le programme allumera/éteindra les LEDs « Marche » et « état de connexion » et affichera « Pas de connexion » et « Pas de lecture » tant que le programme n'aura pas reçu le message de confirmation « PR »
- la connexion est confirmée, nous choisissons l'entrée analogique à lire, après il faut commencer à lire les données envoyées par le PIC. Cependant Si aucune donnée n'est pas reçue, le programme affichera « Pas de lecture » et enverra un 0 vers le PIC. Dans le cas contraire, les données reçues en ASCII seront traité pour avoir une température, cette dernière sera injectée dans le régulateur PID pour générer un signal régulé.
- Le signal régulé est envoyé vers le PIC précédé de «PT» pour lui signifier qu'il s'agit du signal régulé.
- Après confirmation de réception du signal, le programme fini par choisir la PWM à activer, ce dernier l'active, et reçois un message de confirmation d'activation.

Tout cela est continuellement exécuté en boucle tant que le bouton STOP n'est pas activé.



Vue IV.6 : programme sous LabVIEW

1. Ouverture d'un fichier en écriture pour l'enregistrement d'un fichier de données ;
2. Ouverture du port série et configuration des paramètres (fréquence, timeout, nom du Port) ;
3. Envoi de la fréquence et du présalaire de la (PWM) et lecture de l'accusé de réception, en même temps vérification de la connexion ;
4. Rectangle représentant la boucle while (tant que) ;
5. Re-vérification de la connexion dans la boucle while (tant que) en envoyant le message « .00000 » et lire « PR » ;
6. Rectangle représentant une exécution d'événements, dans notre cas nous avons deux événements :
    a. Timeout (délai d'attenante) :

        Exécute les procédures contenue ans le rectangle après un temps prédéfinie dans la case indique par (A) dans la figure précédent.

    b. Appui sur le bouton stop :

        Cet événement permet de sortir de la boucle while donc, arrêter le programme.
7. Lecture de l'entré analogique choisie en envoyant le message « AN0X » et lire la valeur de l'entré X choisie ;
8. Rectangle de la structure CASE, l'entrée de celle-ci est reliée à l'accusé de réception « pas 5 » :
    - Si la lecture est « PR » alors la structure continue l'exécution
    - Sinon afficher pas de connexion.
9. Une case structure qui vérifie si la valeur de l'entre analogique :
    - Si y a une lecture correcte exécutée le programme du régulateur ;
    - Sinon afficher pas de connexion.
10. Envoi de la valeur calculée par le régulateur sous forme de signale (PWM) et lire l'accusé ;
11. Choix et activation du module (PWM) ;
12. Fermeture du fichier et du port COM (série) déjà ouvert dans 1 et 2 ;
13. Exécution de la structure d'enregistrement de fichier en cliquant sur un bouton dédié à cet effet.

III. Partie Identification et configuration du régulateur

Dans cette deuxième partie, nous traiterons du système à identifier, en effet la mise en place finale de ce dernier ne se fera qu'après la détermination des paramètres du régulateur, et l'implémentation de ces derniers dans le programme.

Cela signifie qu'il faut passer par l'identification du système, en commençant d'abords par la définition l'approche à utiliser (en boucle ouverte, ou boucle fermée), puis le choix de la méthode (expérimentale ou analytique), enfin vient l'application de cette dernière pour déterminer le régulateur à utiliser (P, PI, PID, ou autre) et ses paramètres à appliquer sur le procédé.

III.1. Identification

En appliquant un signal échelon sur le système non-régulé (en boucle ouverte), nous constatons l'évolution du système (voir Figure IV.8).

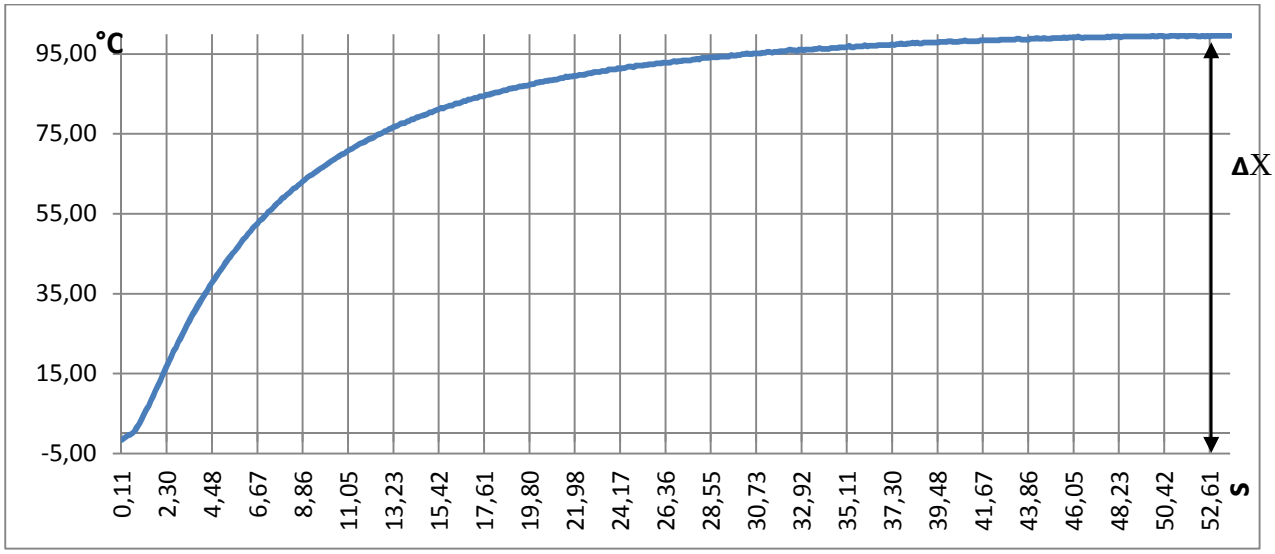


Figure IV.8 : Courbe de réponse du système en boucle ouverte à un échelon de 100%

X : La température T : Le temps en seconde

Vu l'allure sur la figure précédente, nous pouvons déduire que notre système est un premier ordre avec un léger retard au démarrage. Parmi les méthodes d'identification pouvant être utilisé, la Méthode de Broïda, plus exactement, celle qui concerne les procédés autorégulant sera celle que nous utiliserons.

En appliquant l'identification selon la méthode de Broïda sur l'allure de la figure Figure (IV.8), nous obtenons les relevés expérimentaux suivants :

28% de **Δ**X=27,594°C donc t1 correspond à 3 secondes.

40% de **Δ**X=39,622°C donc t2 correspond à 4,397 secondes.

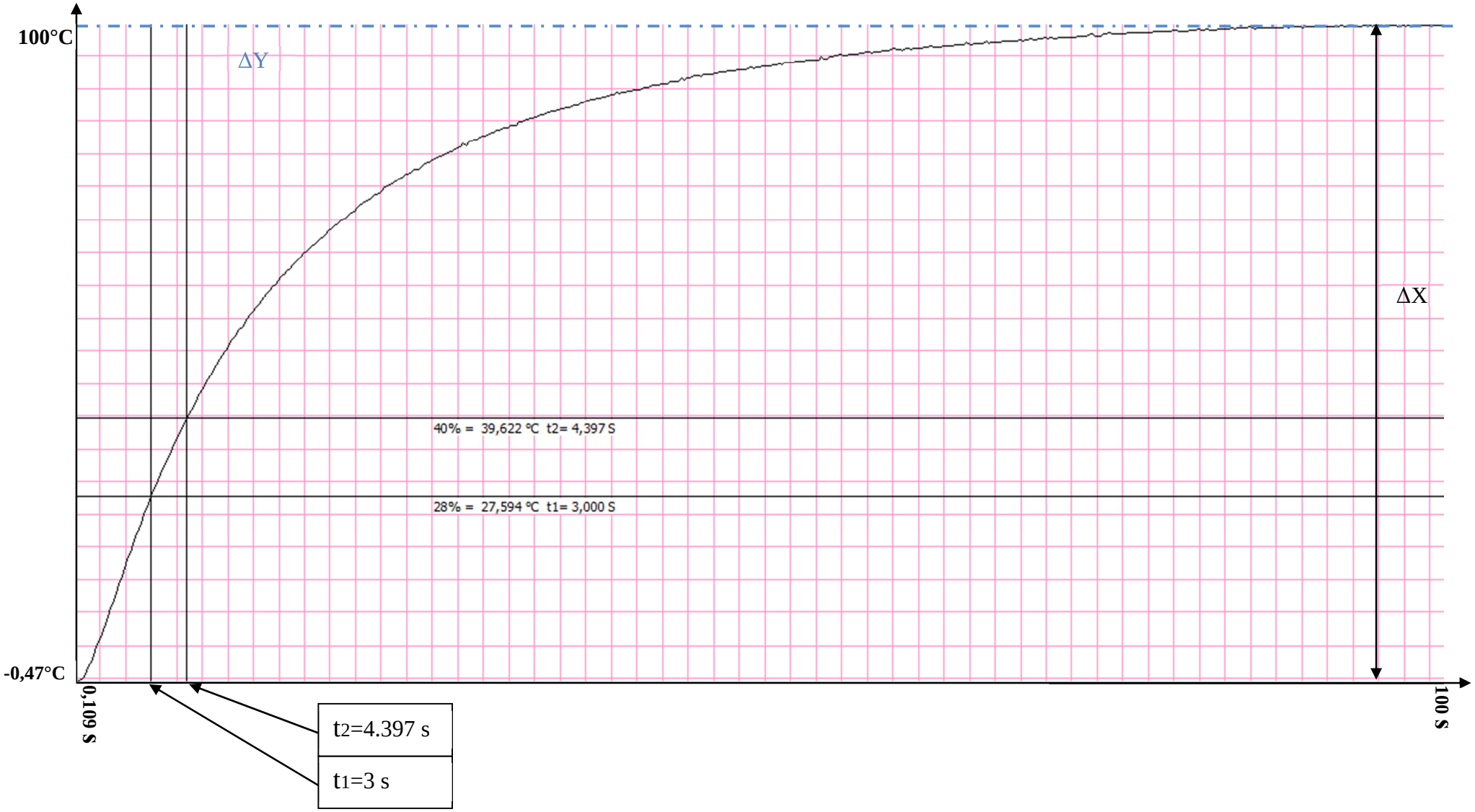


Figure IV.9 : Identification Avec Méthode De Broïda Procèdes Autorégulant

III.2. Détermination du modèle de Broïda

Le modèle de Broïda est de forme : $H(p) = \frac{X(p)}{Y(p)} = \frac{Gs.e^{-\tau p}}{(1+\theta P)}$ (4.12)

Le Gain $Gs = \frac{\Delta X}{\Delta Y} = \frac{100,23}{100} = 1,002.$ (4.13)

La constante de temps $\theta = 5,5 * (t2 - t1) = 7,681\ s.$ (4.14)

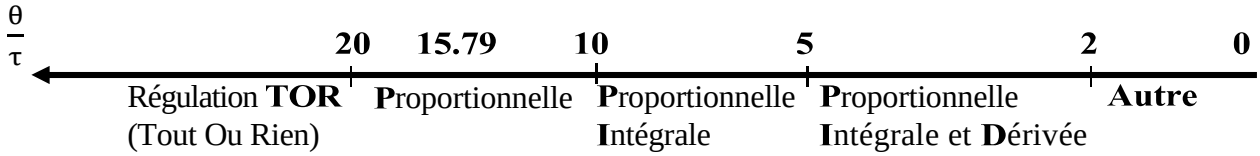
Le retard $\tau = (2,8 * t1) - (1,8 * t2) = 0,486\ s.$ (4.15)

On obtient la fonction $H(p) = \frac{1,002.e^{-0,486p}}{(1+7,681P)}$ (4.16)

III.3. Régulation à appliquer

Après cela vient le choix du régulateur à appliquer. Broïda a mis en place une suite d'étapes à suivre pour trouver la régulation à appliquer :

- l'indice de réglabilité : $\frac{\theta}{\tau} = \frac{7.681}{0.486} = 15.790$ (4.17)



- Cette valeur nous indique que le régulateur approprié à notre système est un régulateur proportionnel qui se calcule suivant le tableau de réglage du régulateur PID selon Broïda :

	P	PI série	PI parallèle	PID série	PID parallèle	PID mixte
Kc	$\frac{0,78}{Gs}.\frac{\theta}{\tau}$	$\frac{0,78}{Gs}.\frac{\theta}{\tau}$	$\frac{0,78}{Gs}.\frac{\theta}{\tau}$	$\frac{0,83}{Gs}.\frac{\theta}{\tau}$	$\frac{0,83}{Gs}.(\frac{\theta}{\tau}+0,4)$	$\frac{0,83}{Gs}.(\frac{\theta}{\tau}+0,4)$
Ti	sans	$\boldsymbol{\theta}$	$\frac{\tau.Gs}{0,78}$	$\boldsymbol{\theta}$	$\frac{\tau.Gs}{0,75}$	$\boldsymbol{\theta}$+0,4τ
Td	0	0	0	0,42τ	$\frac{0,35.\theta}{Gs}$	$\frac{\theta.\tau}{\tau+2.5\theta}$

Ce qui donne un gain K_C =12,288. Kc dans le programme $= (12,288 * 1023)/100 = 125,70$

Après application des paramètres, nous passons à l'application des paramètres de la régulation à appliquer, et nous avons obtenu les résultats illustrés sur la figure suivante :

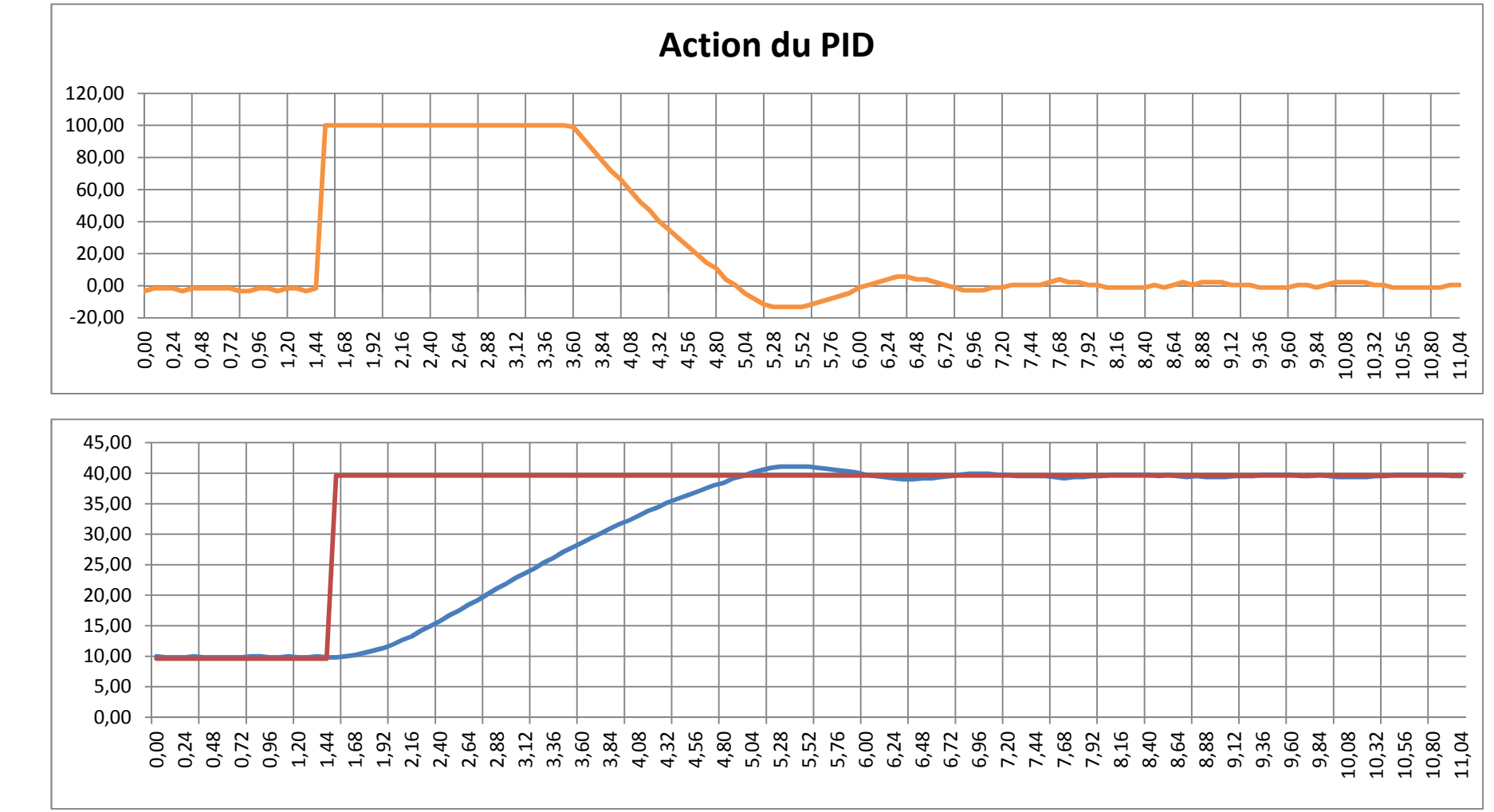


Figure IV.10 : réponse du système et du régulateur à une consigne

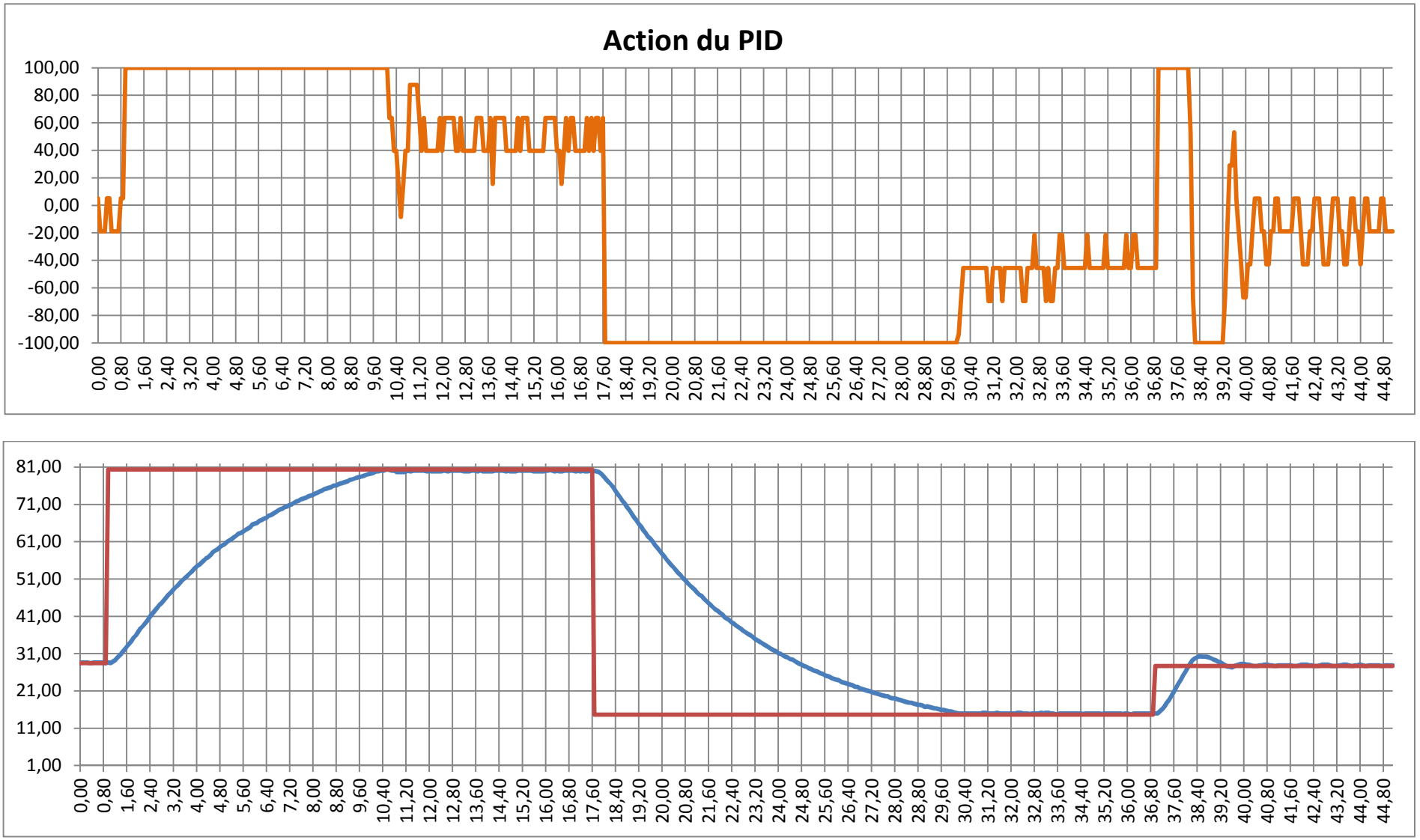


Figure IV.11 : Test de changement de consigne

Nous pouvons remarquer sur les graphs de la figure IV.10 qu'en réponse à la consigne, le régulateur se met en marche, et génère une réponse commandant l'actionneur pour régler la température, mais aussi, il corrige tout dépassement ou perturbation affectant cette dernière.

Dans le but de tester les actions du régulateur et les réactions du Peltier (figure IV.11), nous donnons plusieurs consignes à des instants différents, celles-ci, varient entre de basses, et de hautes températures, en passant par la température ambiante.

Nous constatons que pour les températures très éloignés de la température ambiante, le système mets du temps à atteindre la consigne voulue, contrairement aux températures l'avoisinant.

IV. Conclusion

La mise en pratique des énoncés des précédents chapitres prouve que la régulation d'un procédé repose sur la détermination des paramètres du régulateur.

Pour le cas de notre procédé, nous avons déterminé que le régulateur P est suffisant pour assurer un bon fonctionnement de la chambre de teste, et répondre aux exigences voulus qui sont :

1. Ramener la mesure à la consigne désirée.
2. Maintenir une température voulue.
3. Maintenir un écart le plus faible possible.

A partir des résultats obtenus, nous pouvons constater que LabVIEW associé à une carte à base PIC est un excellent outil pour la conception des applications en temps réel, destinées au contrôle et commande des processus.

Conclusion Générale

Conclusion Générale :

L'objectif de notre travail est la réalisation d'une plateforme de commande et de régulation de température pour un système de teste de composants électroniques, conformément au cahier de charge imposé par l'organisme d'accueil.

La conception et la réalisation complète des cartes (commande et puissance), nous a permis de mettre en vigueur les connaissances acquises durant notre cursus pour traiter des problèmes d'ordre pratique.

Nous avons aussi touché à plusieurs branches (électronique, informatique, automatique) pendant la réalisation de ce modeste travail.

Après la réalisation, nous sommes passé à la programmation du PIC18f2550 en utilisant l'assembleur et le C. l'interface de control, de communication, et de régulation a été programmée sous National Instrument LabVIEW.

Avant la réalisation des cartes précitées, nous avons optés pour le test de ce programme en utilisant la méthode SIL (Software In The Loop).

Durant ce travail nous avons pu acquérir une certaine expérience compte à la conception et à la réalisation des différentes composantes d'un système de régulation.

Les réactions du système que nous avons conçue n'est pas aussi performant que ceux vendus sur le marché, mais il l'est assez pour l'utilisation décrite dans le cahier des charges.

En perspective, du fait que le régulateur soit numérique, il sera très intéressant d'adapter des méthodes de régulation avancées, tel que la méthode dite gain programmé (GAIN Schudling) qui consiste à avoir des paramètres dynamique du régulateur, ou bien d'introduire un algorithme d'adaptation des paramètres suivant le procédé régulé.

Notons aussi que ce régulateur peut être utilisé sur d'autres systèmes autre que la régulation de température, à condition d'adapter les capteurs, les actionneurs, et les paramètres PID adéquats.

Le travail qu'on a réalisé peut aussi servir de base solide pour d'éventuels travaux qui auront comme objet la conception de cartes de commande à base des microcontrôleurs destinés pour des procédés à réguler.

Référence bibliographiques :

[1]: PROUVOST Patrick, *« Instrumentation et régulation en 30 Fiches »*, Edition DUNOD, 2010, 156p

[2]: René PRIGENT, Mathieu AUCLERC, « *Régulation et automatisme des systèmes frigorifiques 2e Edition* », Edition DUNOD, 2013, 207p.

[3] : Microchip technology incorporated, « 390_USB », 160p, 2008.

[4]: J.G STOCKHOLM « Thermoélectricité, application et perspectives », revue Sciences, 95.3, juillet 1995

[5]: Ibrahim DOGAN *« advanced pic microcontroller projects in C »*, Edition ELSVIER. 532p

[6]: « Datasheet PIC18f2550, MAX232, 2n3773, TIP122 »

[7]: LabVIEW, :http://www.ni.com/labview/

[8]: Utilisation des structures de LabVIEW, http://www.ni.com/white-paper/5189/en/pdf

[9]: fiche PT100, http://www2.ac-lyon.fr/lyc69/descartes/IMG/pdf/Affiche_Pt100v3.pdf

[10]: Dufour, C. Abourida, S. Belanger, J. « Hardware-In-the-Loop Simulation of

Power Drives with RT-LAB » PEDS 28-01 Nov. 2005,pp. 1646-1651.

Logiciels Utilisés

LabVIEW 7.5 express

MPLAB C18

ISIS Proteus 8

Annexes

Registres du PIC18f2550 :

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Details on page
TOSU	—	—	—	Top-of-Stack Upper Byte (TOS<20:16>)					---0 0000	53, 60
TOSH	Top-of-Stack High Byte (TOS<15:8>)								0000 0000	53, 60
TOSL	Top-of-Stack Low Byte (TOS<7:0>)								0000 0000	53, 60
STKPTR	STKFUL	STKUNF	—	SP4	SP3	SP2	SP1	SP0	00-0 0000	53, 61
PCLATU	—	—	—	Holding Register for PC<20:16>					---0 0000	53, 60
PCLATH	Holding Register for PC<15:8>								0000 0000	53, 60
PCL	PC Low Byte (PC<7:0>)								0000 0000	53, 60
TBLPTRU	—	—	bit 21[1]	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					--00 0000	53, 84
TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								0000 0000	53, 84
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								0000 0000	53, 84
TABLAT	Program Memory Table Latch								0000 0000	53, 84
PRODH	Product Register High Byte								xxxx xxxx	53, 97
PRODL	Product Register Low Byte								xxxx xxxx	53, 97
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	53, 101
INTCON2	RBPU	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP	1111 -1-1	53, 102
INTCON3	INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF	11-0 0-00	53, 103
INDF0	Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register)								N/A	53, 75
POSTINC0	Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register)								N/A	53, 76
POSTDEC0	Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register)								N/A	53, 76
PREINC0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register)								N/A	53, 76
PLUSW0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register) – value of FSR0 offset by W								N/A	53, 76
FSR0H	—	—	—	—	Indirect Data Memory Address Pointer 0 High Byte				---- 0000	53, 75
FSR0L	Indirect Data Memory Address Pointer 0 Low Byte								xxxx xxxx	53, 75
WREG	Working Register								xxxx xxxx	53
INDF1	Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register)								N/A	53, 75
POSTINC1	Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register)								N/A	53, 76
POSTDEC1	Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register)								N/A	53, 76
PREINC1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register)								N/A	53, 76
PLUSW1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register) – value of FSR1 offset by W								N/A	53, 76
FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High Byte				---- 0000	53, 75
FSR1L	Indirect Data Memory Address Pointer 1 Low Byte								xxxx xxxx	53, 75
BSR	—	—	—	—	Bank Select Register				---- 0000	54, 65
INDF2	Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register)								N/A	54, 75
POSTINC2	Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register)								N/A	54, 76
POSTDEC2	Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register)								N/A	54, 76
PREINC2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register)								N/A	54, 76
PLUSW2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register) – value of FSR2 offset by W								N/A	54, 76
FSR2H	—	—	—	—	Indirect Data Memory Address Pointer 2 High Byte				---- 0000	54, 75
FSR2L	Indirect Data Memory Address Pointer 2 Low Byte								xxxx xxxx	54, 75
STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx	54, 73
TMR0H	Timer0 Register High Byte								0000 0000	54, 129
TMR0L	Timer0 Register Low Byte								xxxx xxxx	54, 129
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	54, 127
OSCCON	IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0	0100 q000	54, 33

HLVDCON	VDIRMAG	—	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0	0-00 0101	54, 285
WDTCON	—	—	—	—	—	—	—	SWDTEN	--- ---0	54, 304
RCON	IPEN	SBOREN(2)	—	RI	TO	PD	POR	BOR	0q-1 11q0	54, 46
TMR1H	Timer1 Register High Byte								xxxx xxxx	54, 136
TMR1L	Timer1 Register Low Byte								xxxx xxxx	54, 136
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	0000 0000	54, 131
TMR2	Timer2 Register								0000 0000	54, 138
PR2	Timer2 Period Register								1111 1111	54, 138
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	54, 137
SSPBUF	MSSP Receive Buffer/Transmit Register								xxxx xxxx	54, 198, 207
SSPADD	MSSP Address Register in I^2C™ Slave mode. MSSP Baud Rate Reload Register in I^2C™ Master mode.								0000 0000	54, 207
SSPSTAT	SMP	CKE	D/A	P	S	R/W	UA	BF	0000 0000	54, 198, 208
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	54, 199, 209
SSPCON2	GCEN	ACKSTAT	ACKDT/ ADMSK5(7)	ACKEN/ ADMSK4(7)	RCEN/ ADMSK3(7)	PEN/ ADMSK2(7)	RSEN/ ADMSK1(7)	SEN	0000 0000	54, 210
ADRESH	A/D Result Register High Byte								xxxx xxxx	54, 274
ADRESL	A/D Result Register Low Byte								xxxx xxxx	54, 274
ADCON0	—	—	CHS3	CHS2	CHS1	CHS0	GO/DONE	ADON	--00 0000	54, 265
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	--00 0qqq	54, 266
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	0-00 0000	54, 267
CCPR1H	Capture/Compare/PWM Register 1 High Byte								xxxx xxxx	55, 144
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								xxxx xxxx	55, 144
CCP1CON	P1M1(3)	P1M0(3)	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	0000 0000	55, 143, 151
CCPR2H	Capture/Compare/PWM Register 2 High Byte								xxxx xxxx	55, 144
CCPR2L	Capture/Compare/PWM Register 2 Low Byte								xxxx xxxx	55, 144
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	55, 143
BAUDCON	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	0100 0-00	55, 246
ECCP1DEL	PRSEN	PDC6(3)	PDC5(3)	PDC4(3)	PDC3(3)	PDC2(3)	PDC1(3)	PDC0(3)	0000 0000	55, 160
ECCP1AS	ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1(3)	PSSBD0(3)	0000 0000	55, 161
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000 0000	55, 281
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	55, 275
TMR3H	Timer3 Register High Byte								xxxx xxxx	55, 141
TMR3L	Timer3 Register Low Byte								xxxx xxxx	55, 141
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	T3SYNC	TMR3CS	TMR3ON	0000 0000	55, 139
SPBRGH	EUSART Baud Rate Generator Register High Byte								0000 0000	55, 247
SPBRG	EUSART Baud Rate Generator Register Low Byte								0000 0000	55, 247
RCREG	EUSART Receive Register								0000 0000	55, 256
TXREG	EUSART Transmit Register								0000 0000	55, 253
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	55, 244
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	55, 245

EEADR	EEPROM Address Register								0000 0000	55, 91
EEDATA	EEPROM Data Register								0000 0000	55, 91
EECON2	EEPROM Control Register 2 (not a physical register)								0000 0000	55, 82
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	xx-0 x000	55, 83
IPR2	OSCFIP	CMIP	USBIP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	1111 1111	56, 109
PIR2	OSCFIF	CMIF	USBIF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	0000 0000	56, 105
PIE2	OSCFIE	CMIE	USBIE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	0000 0000	56, 107
IPR1	SPPIP(3)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	1111 1111	56, 108
PIR1	SPPIF(3)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	56, 104
PIE1	SPPIE(3)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	56, 106
OSCTUNE	INTSRC	—	—	TUN4	TUN3	TUN2	TUN1	TUN0	0--0 0000	56, 28
TRISE(3)	—	—	—	—	—	TRISE2	TRISE1	TRISE0	---- -111	56, 126
TRISD(3)	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	1111 1111	56, 124
TRISC	TRISC7	TRISC6	—	—	—	TRISC2	TRISC1	TRISC0	11-- -111	56, 121
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	56, 118
TRISA	—	TRISA6(4)	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	-111 1111	56, 115
LATE(3)	—	—	—	—	—	LATE2	LATE1	LATE0	---- -xxx	56, 126

LATD(3)	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	xxxx xxxx	56, 124
LATC	LATC7	LATC6	—	—	—	LATC2	LATC1	LATC0	xx-- -xxx	56, 121
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxx xxxx	56, 118
LATA	—	LATA6(4)	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	-xxx xxxx	56, 115
PORTE	RDPU(3)	—	—	—	RE3(5)	RE2(3)	RE1(3)	RE0(3)	0--- x000	56, 125
PORTD(3)	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxx xxxx	56, 124
PORTC	RC7	RC6	RC5(6)	RC4(6)	—	RC2	RC1	RC0	xxxx -xxx	56, 121
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	56, 118
PORTA	—	RA6(4)	RA5	RA4	RA3	RA2	RA1	RA0	-x0x 0000	56, 115
UEP15	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP14	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP13	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP12	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP11	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP10	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP9	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP8	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP7	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP6	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP5	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP4	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP3	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP2	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP1	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UEP0	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	57, 172
UCFG	UTEYE	UOEM	—	UPUEN	UTRDIS	FSEN	PPB1	PPB0	00-0 0000	57, 168
UADDR	—	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0	-000 0000	57, 173
UCON	—	PPBRS	SE0	PKTDIS	USBEN	RESUME	SUSPND	—	-0x0 000-	57, 166
USTAT	—	ENDP3	ENDP2	ENDP1	ENDP0	DIR	PPBI	—	-xxx xxx-	57, 171
UEIE	BTSEE	—	—	BTOEE	DFN8EE	CRC16EE	CRC5EE	PIDEE	0--0 0000	57, 185
UEIR	BTSEF	—	—	BTOEF	DFN8EF	CRC16EF	CRC5EF	PIDEF	0--0 0000	57, 184
UIE	—	SOFIE	STALLI	IDLEIE	TRNIE	ACTVIE	UERRIE	URSTIE	-000 0000	57, 183
UIR	—	SOFIF	STALLI	IDLEIF	TRNIF	ACTVIF	UERRIF	URSTIF	-000 0000	57, 181
UFRMH	—	—	—	—	—	FRM10	FRM9	FRM8	---- -xxx	57, 173
UFRML	FRM7	FRM6	FRM5	FRM4	FRM3	FRM2	FRM1	FRM0	xxxx xxxx	57, 173
SPPCON(3)	—	—	—	—	—	—	SPPOWN	SPPEN	---- --00	57, 191
SPPEPS(3)	RDSPP	WRSPP	—	SPPBUSY	ADDR3	ADDR2	ADDR1	ADDR0	00-0 0000	57, 195
SPPCFG(3)	CLKCF	CLKCF	CSEN	CLK1EN	WS3	WS2	WS1	WS0	0000 0000	57, 192
SPPDATA(3)	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0	0000 0000	57, 196

Legend: x= inconnu, u= inchangé, -= non implémenté, q= valeurs dépends de conditions. Shaded cells are unimplemented, read as '0'.

Note:

1: Bit 21 de TBLPTRU permet l'accès au bit de traduction configuration de l'appareil.

2: Le bit SBOREN est utilisable que quand BOREN<1:0> = 01; sinon, il est lu comme '0'.

3: les registres ayant ce chiffre ne sont utilisables que dans les pics avec 40/44-pines.

4: RA6 est utilisable comme port de configuration des oscillateurs primaires. Quand cette pin est non utilisée, toutes les autres pins qui y sont associé sont égales à 0.

5: RE3 est considéré comme pin de port seulement si le bit de configuration de MCLRE est à « 0 ».

6: RC5 and RC4 sont considérés comme pin de port seulement si le bit 3 de UCON est à « 0 »

7: I^2C™ en mode esclave uniquement.

2N3773

NPN Power Transistors

The 2N3773 is a PowerBase power transistor designed for high power audio, disk head positioners and other linear applications. This device can also be used in power switching circuits such as relay or solenoid drivers, DC DC converters or inverters.

Features

- High Safe Operating Area (100% Tested) 150 W @ 100 V
- Completely Characterized for Linear Operation
- High DC Current Gain and Low Saturation Voltage
  h_{FE} = 15 (Min) @ 8.0 A, 4.0 V
  $V_{CE(sat)}$ = 1.4 V (Max) @ I_C = 8.0 A, I_B = 0.8 A
- For Low Distortion Complementary Designs
- This is a Pb Free Device

MAXIMUM RATINGS (Note 1)

Rating	Symbol	Value	Unit
Collector Emitter Voltage	V_{CEO}	140	Vdc
Collector Emitter Voltage	V_{CEX}	160	Vdc
Collector Base Voltage	V_{CBO}	160	Vdc
Emitter Base Voltage	V_{EBO}	7	Vdc
Collector Current Continuous	I_C	 16 30	Adc
Base Current Continuous	I_B	 4 15	Adc
Total Power Dissipation @ T_A = 25C Derate above 25C	P_D	150 0.855<	W W/C
Operating and Storage Junction Temperature Range	T_J, T_{stg}	65 to +200	C

Stresses exceeding Maximum Ratings may damage the device. Maximum Ratings are stress ratings only. Functional operation above the Recommended Operating Conditions is not implied. Extended exposure to stresses above the Recommended Operating Conditions may affect device reliability.

1. Indicates JEDEC Registered Data.
2. Pulse Test: Pulse Width = 5 ms, Duty Cycle <— 10%.

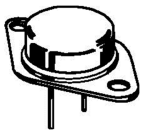
THERMAL CHARACTERISTICS

Characteristic	Symbol	Max	Unit
Thermal Resistance, Junction to Case	R_{0JC}	1.17	C/W

*For additional information on our Pb Free strategy and soldering details, please download the ON Semiconductor Soldering and Mounting Techniques Reference Manual, SOLDERRM/D.

16 A NPN
POWER TRANSISTORS
140 V, 150 W

MARKING DIAGRAM



TO 204
CASE 1 07

A = Assembly Location

YY = Year

WW = Work Week

ORDERING INFORMATION

See detailed ordering and shipping information in the package dimensions section on page 2 of this data sheet.

ELECTRICAL CHARACTERISTICS (T_c = 25°C unless otherwise noted)

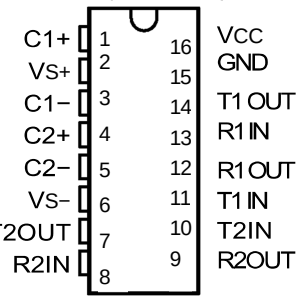
Characteristic	Symbol	Min	Max	Unit
Collector−Emitter Breakdown Voltage (Note 4) (I_C = 0.2 Adc, I_B = 0)	$V_{CEO(sus)}$	140	–	Vdc
Collector−Emitter Sustaining Voltage (Note 4) (I_C = 0.1 Adc, $V_{BE(off)}$ = 1.5 Vdc, R_{BE} = 100 Ohms)	$V_{CEX(sus)}$	160	–	Vdc
Collector−Emitter Sustaining Voltage (I_C = 0.2 Adc, R_{BE} = 100 Ohms)	$V_{CER(sus)}$	150	–	Vdc
Collector Cutoff Current (Note 4) (V_{CE} = 120 Vdc, I_B = 0)	I_{CEO}	–	10	mAdc
Collector Cutoff Current (Note 4) (V_{CE} = 140 Vdc, $V_{BE(off)}$ = 1.5 Vdc) (V_{CE} = 140 Vdc, $V_{BE(off)}$ = 1.5 Vdc, T_C = 150°C)	I_{CEX}	 – –	 2 10	mAdc
Collector Cutoff Current (V_{CB} = 140 Vdc, I_E = 0)	I_{CBO}	–	2	mAdc
Emitter Cutoff Current (Note 4) NBE = 7Vdc, IC=0)	I_{EBO}	–	5	mAdc

DUAL EIA-232 DRIVERS/RECEIVERS

SLLS047L − FEBRUARY 1989 − REVISED MARCH 2004

- **Meets or Exceeds TIA/EIA-232-F and ITU Recommendation V.28**
- **Operates From a Single 5-V Power Supply With 1.0-μF Charge-Pump Capacitors**
- **Operates Up To 120 kbit/s**
- **Two Drivers and Two Receivers**
- **±30-V Input Levels**
- **Low Supply Current ... 8 mA Typical**
- **ESD Protection Exceeds JESD 22 − 2000-V Human-Body Model (A114-A)**
- **Upgrade With Improved ESD (15-kV HBM) and 0.1-μF Charge-Pump Capacitors is Available With the MAX202**
- **Applications**
  - **TIA/EIA-232-F, Battery-Powered Systems, Terminals, Modems, and Computers**

MAX232 ... D, DW, N, OR NS PACKAGE
MAX232I ... D, DW, OR N PACKAGE
(TOP VIEW)



description/ordering information

The MAX232 is a dual driver/receiver that includes a capacitive voltage generator to supply TIA/EIA-232-F voltage levels from a single 5-V supply. Each receiver converts TIA/EIA-232-F inputs to 5-V TTL/CMOS levels. These receivers have a typical threshold of 1.3 V, a typical hysteresis of 0.5 V, and can accept ±30-V inputs. Each driver converts TTL/CMOS input levels into TIA/EIA-232-F levels. The driver, receiver, and voltage-generator functions are available as cells in the Texas Instruments LinASIC™ library.

ORDERING INFORMATION

TA	PACKAGE†		ORDERABLE PART NUMBER	TOP-SIDE MARKING
0°C to 70°C	PDIP (N)	Tube of 25	MAX232N	MAX232N
	SOIC (D)	Tube of 40	MAX232D	MAX232
		Reel of 2500	MAX232DR	
	SOIC (DW)	Tube of 40	MAX232DW	MAX232
		Reel of 2000	MAX232DWR	
	SOP (NS)	Reel of 2000	MAX232NSR	MAX232
−40°C to 85°C	PDIP (N)	Tube of 25	MAX232IN	MAX232IN
	SOIC (D)	Tube of 40	MAX232ID	MAX232I
		Reel of 2500	MAX232IDR	
	SOIC (DW)	Tube of 40	MAX232IDW	MAX232I
		Reel of 2000	MAX232IDWR	

MAX232, MAX232I

DUAL EIA-232 DRIVERSIRECEIVERS

SLLS047L − FEBRUARY 1989 − REVISED MARCH 2004

Function Tables

EACH DRIVER

INPUT TIN	OUTPUT TOUT
L	H
H	L

H = high level, L = low level

EACH RECEIVER

INPUT RIN	OUTPUT ROUT
L	H
H	L

H = high level, L = low level

logic diagram (positive logic)