

الجمهورية الجزائرية الديمقراطية الشعبية  
وزارة التعليم العالي و البحث العلمي

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE  
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE  
SCIENTIFIQUE

جامعة سعد دحلب البليدة  
Université SAAD DAHLEB de BLIDA



Faculté des sciences de  
كلية الهندسة  
d'électronique

l'ingénieur

Département  
دائرة الإلكترونيك

Mémoire de projet de fin d'études  
Pour l'obtention du diplôme de MASTER en électronique  
option automatique

# Commande adaptative d'un four électrique

Présenté par :

- LAREF Rachid
- BOUSLIMANI Billel

Proposé par :

**B.KAZED**

Session : Septembre 2011

## SOMMAIRE

### REMERCIEMENTS

### RESUMES

### LISTES DES FIGURES ET TABLEAUX

INTRODUCTION GENERALE.....	1
----------------------------	---

#### Chapitre I : Généralités

I.1 Introduction.....	2
I.2 Incubation artificielle des œufs.....	2
I.2.1 Les différents types d'incubateur.....	2
I.2.1.1 Incubateur statique.....	3
I.2.1.2 Incubateur dynamique.....	3
I.2.2 Les étapes d'incubation artificielle des œufs.....	4
I.2.2.1 Réception des œufs et stockage.....	4
I.2.2.2 Préchauffage des œufs :.....	5
I.2.2.3 Incubation des œufs et éclosion.....	5
I.3 Conclusion.....	6

#### Chapitre II : La carte d'interface

II.1 Introduction.....	7
II.2 La liaison RS-232.....	7
II.2.1 Signification des signaux.....	7
II.2.2 Le protocole série RS-232 :.....	8
II.3 Le bus I2C.....	9
II.3.1 Caractéristiques.....	9
II.3.2 Le protocole I2C.....	10
II.3.2.2 La transmission d'un octet.....	11
II.4 Le capteur de la température DS1621.....	12
II.4.1 Caractéristiques techniques.....	13
II.4.2 Principe de fonctionnement.....	13
II.4.3 Instructions de mesure de température.....	15
II.5 microcontrôleur pic 16F877.....	16

II.5.1 Le cœur du PIC : le microprocesseur.....	16
II.5.2 La mémoire .....	16
II.5.2.1 La mémoire FLASH.....	16
II.5.2.2 La mémoire RAM .....	17
II.5.2.2 La mémoire EEPROM .....	17
II.5.3 Les Ports d'Entrées/Sorties .....	17
II.5.4 Le Port Parallèle Esclave PSP « Parallel Slave Port ».....	17
II.5.5 Le circuit de génération d'horloge.....	17
II.5.6 Le convertisseur Analogique – numérique .....	18
II.5.7 USART .....	18
II.5.8 Le SSP.....	18
II.5.9 Les Interruptions .....	19
II.6 Le max232 .....	19
II.7 La réalisation de la carte d'interface.....	20
II.7.1 partie matérielle .....	20
II.7.2 partie programmation.....	20
II.8 La carte de puissance .....	21
II.8.1 La réalisation de la carte de puissance .....	21
II.9 Conclusion.....	21

### **Chapitre III : Identification du système**

III.1 Introduction.....	22
III.2 Généralité sur la modélisation.....	22
III.2.1 Modèles non paramétriques.....	22
III.2.1.1 Analyse fréquentielle .....	22
III.2.1.2 Réponse impulsionnelle .....	23
III.2.2 Modèles paramétriques .....	23
III.2.2.1 Les méthodes paramétriques graphiques (déterministes) .....	23
III.2.2.2 Les méthodes paramétriques statique .....	24
III.2.2.3 L'algorithme des moindres carrées .....	25
III.2.3 Les séquences binaires pseudo aléatoires (S.B.P.A) .....	29
III.2.3.1 choix des paramètres du SBPA.....	29
III.2.4 Choix du modèle.....	30
III.3 Etudes expérimentale .....	31

III.3.2 Identification des paramètres .....	32
III.3.3 Validation du modèle.....	36
III.3.3.1 analyse résiduelle .....	36
III.3.4 La réponse indicielle du modèle.....	37
III.4 Conclusion.....	38

## **Chapitre IV : La commande numérique**

IV.1 Introduction.....	39
IV.2 La régulation .....	40
IV.2.1 Le correcteur PID .....	41
IV.2.1.1 Description des différentes actions du PID.....	41
IV.2.1.2 Synthèse de contrôleurs PID.....	43
IV.2.1.3 la correction PID du modèle .....	45
IV.2.1.4 Définition de PWM.....	49
IV.2.1.4.1 Principe de PWM.....	49
IV.2.1.4.2 Les types de PWM.....	49
IV.2.1.5 le correcteur PID en pratique .....	51
IV.3 Commande adaptative.....	55
IV.3.1 Commande adaptatif à modèle de référence .....	55
IV.3.1.1 La règle du MIT (méthode de gradient).....	56
IV.3.1.1.1 L'application de la règle du MIT .....	56
IV.4 Conclusion .....	60
CONCLUSION GENERALE .....	61
BIBLIOGRAPHIE .....	62
ANNEXE	

# Remerciement

*Nous tenons à exprimer nos vifs remerciements à notre promoteur Dr B. KAZED pour nous avoir encadrés et diriger durant ce projet de fin d'études.*

*Nous remercions chaleureusement les membres du jury pour l'honneur qu'ils nous ont fait en acceptant d'évaluer notre travail.*

*Nous souhaitons également remercier tous les enseignants de l'université de BLIDA et en particulier, l'équipe d'Automatique qui nous ont encadré dans notre formation en Master.*

*Nous tenons à exprimer notre très grande gratitude, et notre profonde affection à nos chers parents pour leurs encouragements, leur patience et leur grand soutien, durant toutes ces années d'études.*

*Nous remercions également tous nos amis et camarades, pour leurs encouragements et leur précieux soutien.*

## المخلص

إن العمل المقدم في هذه المذكرة يهتم أساسا بالتحكم في درجة حرارة فرن. و هذا باستعمال بطاقة واجهة لاكتساب معطيات المستشعر الحراري نحو الحاسوب و كذلك لإرسال التحكم نحو مقاومات التدفئة. هذه المعطيات تحدد معالم النظام بفضل طرق دقيقة. معالم النظام تعطي نموذج رياضي يمكننا من التحكم في درجة الحرارة على أساس مصحح PID كلاسيكي و التحكم المكيف بالنموذج المرجعي.

## Résumé

Le travail présenté dans ce mémoire s'intéresse principalement à la commande de la température d'un four .ceci en utilisant une carte d'interface pour acquérir les données de capteur du température vers le PC ainsi que pour envoyer la commande vers les résistances de chauffage. Ces données permettent d'identifier les paramètres de notre système grâce à des méthodes bien précises. Les paramètres de notre système donnent un modèle mathématique qui permet de réguler la température à base d'un correcteur PID classique et la commande adaptative à modèle de référence.

## Summary

the work presented in this thesis is mainly concerned with controlling the temperature of an oven. This by using an interface board to acquire data from temperature sensor to the PC and to send the command to the heating resistors. These data are used to identify the parameters of our system through MATLAB identification toolbox. The parameters of our system give a mathematical model that helps regulate the temperature based on a conventional PID controller and adaptive control with reference model.

## INTRODUCTION GENERALE

Dans un monde en perpétuel changements, les sociétés ont besoin d'être réactives pour rester compétitives et pour conquérir de nouveaux marchés. Pour y arriver, ils sont contraints d'améliorer leurs productions pour bien réagir face aux problèmes rencontrés.

Ceci les conduit à modifier et adapter leurs moyens de production. De nos jours, grâce aux développements de l'électronique et de l'informatique, les systèmes sont devenus automatisés et la plupart des lois de commande sont implémentées sur des microcontrôleurs ou processeurs numériques.

La régulation de la température est indispensable pour plusieurs domaines de l'industrie comme l'industrie pharmaceutique ou agroalimentaire.

Notre travail se place dans ce contexte, il consiste à régler la température d'un four électrique.

Le système est un four muni des deux résistances chauffantes, la mesure de la température se fait par un capteur numérique, les mesures sont transmises vers le PC à travers une carte d'interface et la commande de ces résistances se fait par un relai.

L'objectif de notre projet est de trouver un modèle mathématique du four qui facilite l'obtention d'une loi de commande afin de commander ce processus, à savoir maintenir la température à l'intérieur du four, à une valeur prédéfinie.

Ce travail est composé de deux parties :

Partie matérielle consiste à réaliser la carte d'interface et la carte de puissance.

Partie programmation consiste à l'identification et la commande.

Le mémoire est composé de quatre chapitres :

Dans le premier chapitre nous donnerons des généralités sur l'incubateur artificiel des œufs et les étapes d'incubation en montrant l'importance du suivi de la température. Le deuxième chapitre est consacré à interfacer le four avec le PC.

L'obtention du modèle mathématique du système est une condition nécessaire pour faire les études sur ce système, pour cela le chapitre trois est dédié à l'identification de notre système.

Le quatrième chapitre est consacré à la commande du processus, en appliquant la commande PID classique et la commande adaptative avec un modèle de référence.

## Liste des figures et tableaux

Figure II.1 RS232 8, n, 1 .....	8
Figure II.2 RS232 8, n, 2.....	8
Figure II.3 RS232 8, p, 1 .....	9
Figure II.4 le bus I2C .....	10
Figure II.5 les conditions de départ du bus I2C .....	10
Figure II.6 transmission d'une adresse.....	12
Figure II.7 Synoptique des blocs fonctionnels intégrés dans le DS1621 .....	14
<a href="#">Figure</a> II.8 le circuit max232.....	20
Figure III.1 Organigramme de différents types de modélisation .....	28
Figure III.2 génération de signal pseudo aléatoire. ....	29
Figure III.3 une séquence pseudo aléatoire.....	30
Figure III.4 Réponse indicielle avec pas d'échantillonnage $T_e=5s$ . ....	31
Figure III.5 la réponse du système à une séquence SPBA .....	33
Figure III.6 la tool box de l'identification du système.....	34
Figure III.7 la sortie réelle et la sortie estimée .....	36
Figure III.8 l'analyse résiduelle.....	37
Figure III .9 La réponse indicielle du modèle .....	38
Figure IV.1 La régulation en général .....	39
Figure IV.2 régulation du système en boucle fermée.....	40
Figure IV.3 action proportionnelle.....	41
Figure IV.4 action intégrale .....	42
Figure IV.5 action dérivée.....	42
Figure IV.6 réponse indicielle du système 1 ordre avec retard .....	43
Figure IV.7 Le schéma bloc de la régulation du modèle .....	46
Figure IV.8 signal de sortie, la consigne $U_c=10$ . ....	47
Figure IV.9 signal de sortie à la consigne $U_c=20$ . ....	47
Figure IV.10 signal de commande à la consigne $U_c=10$ .....	48
Figure IV.11 signal de commande, consigne $U_c=20$ . ....	48
Figure IV.12 Principe de PWM.....	49
Figure IV.13 signal PWM asymétrique.....	50
Figure IV.14 signal PWM symétrique.....	50
Figure IV.15 l'organigramme du programme. ....	51
Figure IV.16 Le schéma bloc du modèle discret.....	51
Figure IV.17 Menu des paramètres PID du MATLAB.....	52
Figure IV.18 signal de sortie, $U_c=38$ . ....	52
Figure IV.19 signal de commande, $U_c=38$ .....	52

Figure IV.20 le signal de sortie réelle, $U_c=38$ .....	53
Figure IV.21 le signal de commande réelle, $U_c=38$ .....	53
Figure IV.22 le signal de sortie réelle, $U_c=38$ .....	54
Figure IV.23 le signal de sortie réelle, $U_c=38$ .....	54
Figure IV.24 Schéma bloc d'un système de contrôle à modèle référence. ....	56
Figure IV.25 Schéma bloc de simulink.....	59
Figure IV.26 la simulation du sortie du système et la sortie du modèle. ....	60
Figure IV.27 la commande appliqué au système .....	60
Figure IV.28 la sortie du système réel et la sortie du modèle de référence .....	61
Figure IV.29 la commande appliqué au système réel .....	61
Tableau II.1 Correspondance température/données .....	13
Tableau II.2 Instructions de mesure de température .....	15
Tableau IV.1 Paramètres PID obtenus à partir de Broïda. ....	43
Tableau IV.2 Paramètres PID obtenus à partir d'une réponse indicielle .....	44
Tableau IV.3 Paramètres PID obtenus à partir de Ziegler-Nichols.....	45

## **I.1 Introduction**

La régulation de la température est indispensable dans tous les domaines industriels.

Plusieurs secteurs exigent un contrôle de cette grandeur physique avec une grande précision, par exemple le domaine agricole qui nécessite un maintien de cette grandeur :

Les chambres froides pour la conservation des produits agroalimentaire, les serres pour la production artificielle des fruits et légumes aussi le secteur d'incubation artificielle des œufs qu'on va présenter dans ce chapitre.

## **I.2 Incubation artificielle des œufs**

La couveuse artificielle ou "incubateur" ne date pas d'aujourd'hui. Depuis plusieurs siècles, les Chinois et les Égyptiens construisaient ces machines. En Égypte ancienne ils se nommaient "Mammals". C'étaient de véritables constructions en maçonnerie comportant des fours chauffés à la bouse de chameau au dessus desquels se trouvaient les chambres d'incubation. Un seul homme s'occupait d'un Mammal pouvant incuber plusieurs milliers d'œufs. Il n'était pas rétribué mais tirait son bénéfice de sa production en livrant 80 poussins pour 100 œufs mis à couver.

Actuellement presque tous les incubateurs fonctionnent à l'électricité, bien qu'il existe encore quelques modèles à pétrole ou à gaz

Le principe de l'incubateur est toujours le même : une enceinte étanche, souvent doublée de matériaux isolants, dans laquelle se trouvent principalement une source de chaleur régulée par un thermostat et, accessoirement, une source d'humidité et des orifices permettant le renouvellement progressif de l'air. On peut également y trouver des organes mécaniques facilitant le retournement des œufs.

### **I.2.1 Les différents types d'incubateur**

L'incubateur peut être "statique", c'est à dire que l'air contenu à l'intérieur ne se déplacera et renouvellera qu'en fonction de sa température et du réglage des ouvertures d'échange avec l'extérieur. Il peut également être "dynamique" ce qui signifie que l'air intérieur est brassé par un système de ventilation, ce qui n'exclut pas la nécessité de le renouveler progressivement par apport d'air extérieur. Selon qu'il s'agira de l'un ou de l'autre modèle, les réglages seront différents.

### **I.2.1.1 Incubateur statique**

Les incubateurs statiques sont les couveuses traditionnelles où les œufs sont disposés sur un seul plateau. Le chauffage s'effectue en général par le haut. Sur les modèles électriques, il s'agit le plus souvent d'une résistance tapissant la face inférieure du plafond, mais il existe aussi des modèles chauffés par de classiques lampes à incandescence ou par des lampes à filament de carbone. Sur les modèles à gaz ou à pétrole, l'air intérieur est réchauffé soit directement sous l'action d'une flamme permanente (dans ce cas la régulation de température est effectuée en évacuant les gaz trop chauds par ouverture d'une soupape, commandée par un thermostat mécanique, au sommet de la couveuse), soit par l'intermédiaire d'un réservoir d'eau chaude. Les modèles à réservoir d'eau chaude sont un peu plus complexes, donc souvent plus chers, mais les variations de température y sont beaucoup plus lentes, leur inertie étant importante.

Dans les couveuses statiques la température est toujours beaucoup plus élevée vers le haut de l'appareil que vers le bas, d'une part parce que la source de chaleur est presque toujours en haut, d'autre part parce que l'air chaud se rassemble toujours vers le sommet d'une enceinte fermée. La température est donc de plus en plus basse en allant du haut vers le bas.

En moyenne, et quel que soit le modèle, elle descend d'un degré par centimètre, bien que cela puisse légèrement varier. Pour bénéficier de la même température, les œufs doivent donc tous être placés au même niveau et être de même grosseur. La température se mesure au sommet des œufs et doit être de 39° centigrades.

Un œuf normal ayant 4 à 5 centimètres d'épaisseur, la face inférieure est à 34 ou 35°C si le sommet est à 39°C.

Le retournement des œufs au moins deux fois par jour est donc particulièrement nécessaire. Si l'on place en même temps des œufs de différentes grosseurs, les plus petits ne seront pas chauffés, un œuf de 3 centimètres d'épaisseur sera entre 35 et 37°C).

On peut pallier à cet inconvénient en posant les œufs les plus petits sur une planchette compensant leur manque d'épaisseur mais ils seront alors un peu plus chauds que les gros œufs, entre 36 et 39 °C et se développeront légèrement plus vite.

### **I.2.1.2 Incubateur dynamique**

Dans un incubateur dynamique, l'air intérieur est brassé en permanence. Le plus souvent c'est un ventilateur mis directement dans la chambre d'incubation qui remplit ce rôle. Il doit être placé suffisamment loin des œufs ou protégé par une grille pour que les poussins naissants ne

risquent pas de s'y blesser. D'autres modèles comportent une soufflerie qui prélève l'air en haut ou en bas de la chambre d'incubation et le réinjecte à l'autre extrémité après réchauffage et filtration.

L'avantage des modèles dynamiques est que la température est, théoriquement, uniforme dans l'ensemble de l'appareil. Les œufs sont à température égale sur toutes leurs faces. Des œufs de différentes grosseurs peuvent donc être placés ensemble. Si la ventilation est puissante, on peut y mettre plusieurs plateaux superposés contenant des œufs, pourvu que le fond de ces plateaux soit grillagé pour permettre la circulation de l'air. Le plus souvent il y a trois plateaux permettant de mettre en incubation tous les sept jours. Un quatrième plateau peut servir d'éclosion à moins que l'on utilise une seconde couveuse pour cet usage. Dans les modèles dynamiques la température doit être réglée à 37,8 ou 38 °C, toujours mesurée au sommet des œufs, bien que, en théorie, elle soit partout identique. Les modèles dynamiques sont toujours plus chers à l'achat que les modèles statiques. A capacité et isolation égales, malgré le moteur de ventilation, ils consomment un peu moins de courant que les statiques car la température y est moins élevée : dans les modèles statiques, si la température est réglée à 39° au sommet des œufs, elle peut être beaucoup plus élevée au sommet de l'enceinte.

## **I.2.2 Les étapes d'incubation artificielle des œufs**

Le succès de l'incubation commence par la collecte des œufs dans les nids.

Immédiatement après la collecte, le lavage des œufs à incuber n'est pas recommandé, en revanche les œufs les plus sales doivent être brossés dans l'eau à 40°C (la température de l'eau doit toujours être supérieure à la température de l'œuf).

### **I.2.2.1 Réception des œufs et stockage**

Immédiatement après la collecte et le nettoyage des œufs, il faut pratiquer une fumigation sur place avant leur acheminement vers le couvoir. Cette fumigation immédiate est capitale. Plus on attend et moins elle est efficace, car les germes *Aspergillus* et *Salmonella* peuvent pénétrer la coquille.

Les œufs doivent alors être disposés en une seule couche, pour être directement accessibles aux radiations solaires. Il est important de faire ce traitement dans un endroit propre, afin que les œufs ne récoltent pas plus de bactéries que le soleil n'en détruit.

Il faut bien entendu contrôler la durée de l'exposition pour ne pas risquer une surchauffe des œufs.

### **I.2.2.2 Préchauffage des œufs :**

Après la fumigation, les œufs d'oies doivent être stockés en attendant l'incubation. Il est recommandé de les entreposer à une température comprise entre 10 et 15°C avec une hygrométrie de 70 à 75 pour cent. Les œufs doivent subir six rotations par jour avec une amplitude de 90 degrés pendant la période de stockage. Le rythme d'une incubation par semaine entraîne des durées de conservation des œufs de sept jours maximum, ce qui n'est pas préjudiciable à la production des poussins. Pour constituer des grandes bandes de poussins, on est parfois amené à espacer les incubations, ce qui entraîne des stockages de 15 jours ou plus. En pratique, les résultats obtenus sur des œufs conservés deux semaines sont légèrement hypothéqués mais convenables. Au delà de 15 jours de conservation, ces résultats deviennent aléatoires. Des travaux récents sur le préchauffage des œufs de poule en période de stockage indiquent que le préchauffage des œufs conservés pendant une longue durée améliore leur éclosabilité.

### **I.2.2.3 Incubation des œufs et éclosion**

Pendant la durée d'incubation les conditions habituelles indiquent une température de 37,7°C et une hygrométrie de 50 à 55 pour cent pour la période qui concerne les 18 premiers jours, c'est-à-dire avant le transfert en éclosoir. Les œufs sont placés en position horizontale dans l'incubateur et retournés sur 90 degrés toutes les 2 heures. Si on place les œufs en position verticale, le rythme de rotation reste le même, mais l'amplitude de la rotation passe à 180 degrés.

Les œufs doivent être refroidis chaque jour en ouvrant les portes de l'appareil pendant 15 minutes. Avant de refermer l'incubateur, il faut les brumasser avec de l'eau, ou mieux, avec une solution bactéricide. La température de l'eau doit être comprise entre 20 et 25°C. Au septième jour de l'incubation, il faut mirer les œufs au moyen d'une source lumineuse et contrôler par transparence que l'embryon se développe normalement. Les œufs clairs et les embryons morts doivent être retirés, car ils sont des sources de contaminations potentielles. Lors du passage en éclosoir le 18e jour, il faut renouveler l'opération et extraire les embryons qui sont morts entre temps. Dans l'éclosoir, les œufs ne sont plus refroidis ni humidifiés, la température est de 37,5°C et l'hygrométrie de 75 pour cent. L'éclosion débute le 20e jour, il

faut ouvrir les trappes de l'éclosoir pour poussins sèchent, même si on ne les sort effectivement que le 21e jour.

### **I.3 Conclusion**

Le respect de la température recommandée pendant la collection des œufs jusqu'à l'éclosion est le facteur le plus important dans la production des poussins en revanche l'erreur de réglage de la température explique la plupart des échecs.

## **Chapitre II**

### **La Carte d'interface**

#### **II.1 Introduction**

L'acquisition d'une donnée physique telle que la température nécessite l'emploi d'une carte d'interface reliant le capteur au PC. Ce dernier a pour rôle de traiter, stocker et afficher les données reçues. Pour notre projet il s'agit d'acquérir la température à l'intérieur d'un four électrique, utilisant un capteur numérique délivrant la température sous forme de donnée série au format I2C. Afin de transmettre cette donnée au PC. Nous allons exploiter un microcontrôleur qui aura pour tâche de transformer cette donnée au format série RS-232 pour être ensuite transmise au PC.

#### **II.2 La liaison RS-232**

La liaison RS-232 sert à relier la carte d'interface « PIC » avec le PC par des bits « signal », et le protocole RS-232 permet d'interpréter ces bits des deux côtés « PC-PIC ».

La RS-232 est une liaison série asynchrone dissymétrique full duplex normalisée avec une trame d'un caractère de longueur. Selon la norme, elle comporte 22 fils - 23 dans sa déclinaison RS-232C. Ces fils sont placés de manière standard sur une broche DB25. Lorsque IBM créa son PC en 1981, Intel lui fournissait le circuit (8250) capable de ne gérer directement que 9 de ces fils. À partir de cela les PC n'implémentent en standard qu'une sous-version de la norme complète, limitée à ces 9 fils. Pour des soucis de simplicité, d'économie et d'encombrement, ces 9 fils furent positionnés non plus sur une DB25, mais sur une DB9.

## II.2.1 Signification des signaux

**GND** : (Ground) la masse. Référence nécessaire à toute mesure de tension.

**Rx** : (Received Data) Données reçues.

**Tx** : (Transmitted Data) Données émises.

**CTS** : (Clear to Send) Indique que l'appareil est prêt à recevoir.

**RTS** : (Request to Send) Indique que le PC est prêt à recevoir.

**DSR** : (Data Set Ready) Indique que l'appareil s'occupe de sa liaison RS-232.

**DTR** : (Data Terminal Ready) Indique que le PC s'occupe de la liaison RS-232.

**DCD** : (Data Carrier Detect, aussi nommée RLSD : Receive Line Signal Detect) Indique au PC qu'une porteuse de données valide a été détectée.

**RI** : (Ring Indicator) Indique au PC que la ligne téléphonique à laquelle l'appareil est relié est en train de sonner.

## II.2.2 Le protocole série RS-232 :

La transmission des données peut se faire suivant plusieurs formats (7 ou 8 bits) avec ou sans contrôle de parité pouvant être gérée comme paire ou impaire); une trame commence par 1 bit de *start* « "0" logique » et se termine par 1 ou 2 bits de *stop* « "1" logique ».

Parité paire : Par = 1 si les données [7:0] a un nombre pair de "1".

Parité impaire : Par.= 1 si données [7:0] a un nombre impair de "1".

Les figures suivantes présentent l'allure de quelques formats :

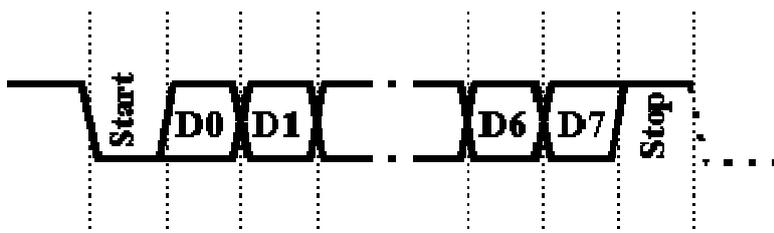


Figure II.1 RS232 8, n, 1 : 8 bits de données, pas de parité, 1 bit de stop.

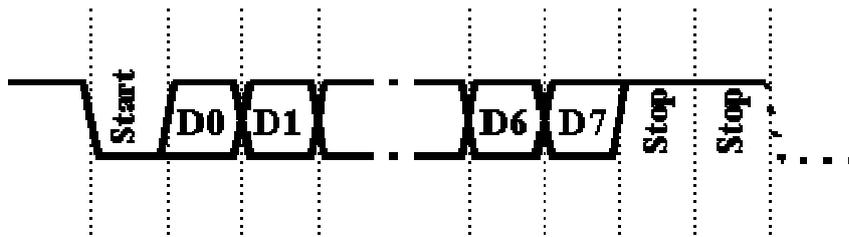


Figure II.2 RS232 8, n, 2 : 8 bits de données, pas de parité, 2 bits de stop.

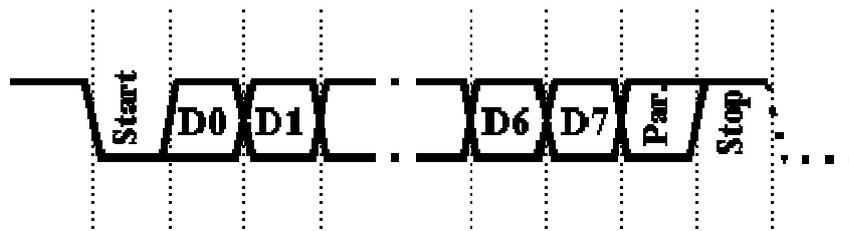


Figure II.3 RS232 8, p, 1 : 8 bits de données, avec parité, 1 bit de stop.

Les circuits d'interface RS-232 à l'heure actuelle émettent à +12V et -12V, car le récepteur, probablement très proche, a de bonnes chances de recevoir une tension non altérée ; il faut donc que l'émetteur reste dans la zone accessible au récepteur. De plus, ces deux tensions sont souvent présentes dans les appareils numériques, et en tous cas fournis par toutes les alimentations de PC. Notons aussi, que ce soit en émission ou en réception, un certain nombre de limites à la tension pouvant représenter un bit. L'absence d'information « no man's land » autour de 0V a deux raisons : d'abord, cette zone inaccessible évite de confondre un 0 avec un 1, et d'autre part, le fait que le 0V soit hors limite permet une détection sommaire de pannes : un câble débranché, coupé, détérioré ou un faux contact ne peut plus être pris pour l'émission constante de l'un des deux bits.

### II.3 Le bus I2C

Pour que le PIC communique avec le capteur, envoyer ou recevoir des données on utilise le bus I2C. L'utilisation d'un nombre réduit de fils et un grand nombre de capture d'une manière rapide par un seul bus rend le bus I2C efficace. Le bus I<sup>2</sup>C (Inter Integrated Circuit) fait partie des bus série, deux fils pour faire tout passer. Il a été développé au début des années 1980, par Philips pour minimiser les liaisons entre les circuits intégrés. Aujourd'hui, Philips a dans son catalogue plus de 150 circuits intégrés CMOS et bipolaires qui sont compatibles I<sup>2</sup>C. D'autres fabricants ont aussi développé des circuits intégrés qui peuvent être connectés au bus I<sup>2</sup>C. Le protocole utilisé est simple et rapide.

### II.3.1 Caractéristiques

Le bus I<sup>2</sup>C permet de faire communiquer entre eux des composants électroniques très divers grâce à seulement trois fils : un signal de données (SDA), un signal d'horloge (SCL), et un signal de référence électrique (masse). Il s'agit d'une liaison en mode série, ce qui signifie que la vitesse de transfert sera plus faible qu'avec un bus de type parallèle. Le bus I<sup>2</sup>C permet cependant des échanges à la vitesse de 100 kbits par seconde. Certes, la vitesse de transfert du bus I<sup>2</sup>C n'est pas fulgurante, mais dans bien des cas, la vitesse n'est pas l'élément prédominant. L'utilisation d'un bus I<sup>2</sup>C permet de réduire la complexité des circuits imprimés à réaliser. Par exemple, pour connecter une EEPROM ou une RAM à un microcontrôleur classique, il faut relier entre eux les bits de données et les bits d'adresses des différents composants, et, en plus, il faut bien souvent ajouter une logique de sélection. Avec des composants prévus pour se connecter au bus I<sup>2</sup>C, il suffit de les relier par deux pistes seulement.

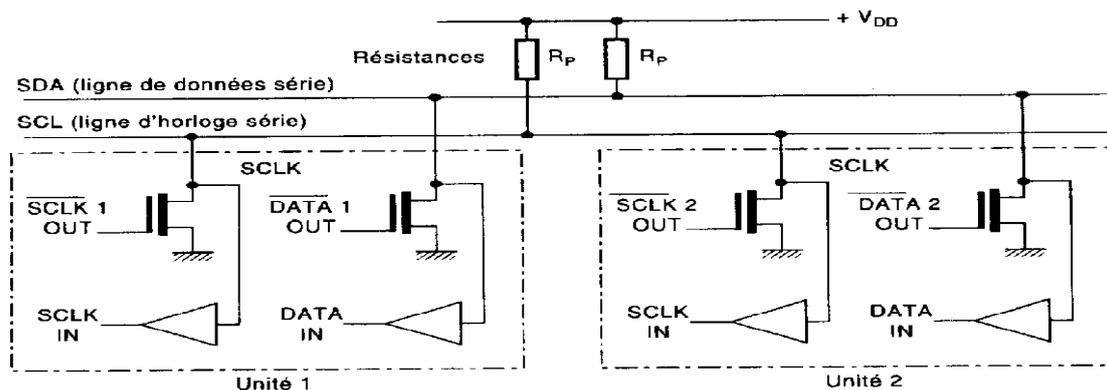


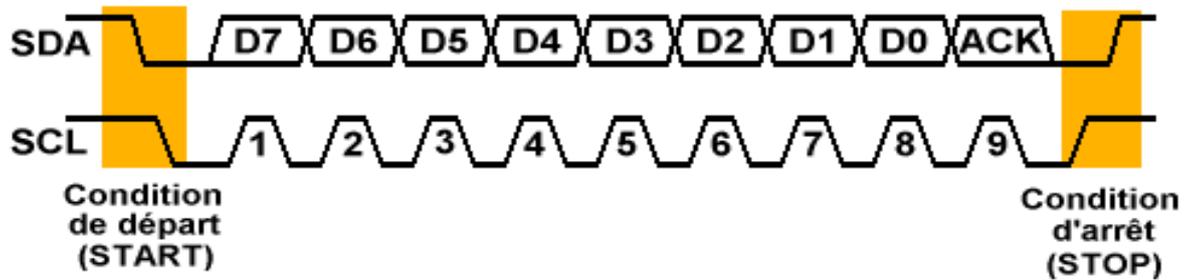
Figure II.4 le bus I<sup>2</sup>C

### II.3.2 Le protocole I<sup>2</sup>C

Le protocole du bus I<sup>2</sup>C définit la succession des états possibles sur les lignes SDA et SCL, et comment doivent réagir les circuits en cas de conflit.

#### II.3.2.1 La prise de contrôle du bus

Pour transmettre des données sur le bus I<sup>2</sup>C, il faut surveiller deux conditions particulières : la condition de départ et la condition d'arrêt.



**Figure II.5 les conditions de départ du bus I2C**

Avant de tenter de prendre le contrôle du bus, un circuit doit vérifier que les lignes SDA et SCL sont au repos, c'est-à-dire à l'état haut. Si c'est le cas, le circuit indique qu'il prend le contrôle du bus en mettant la ligne SDA à 0.

A partir de ce moment là, les autres circuits savent que le bus est occupé et ils ne devraient pas tenter d'en prendre contrôle. Le circuit qui vient de prendre le contrôle du bus en devient le maître .C'est lui qui génère le signal d'horloge, quel que soit le sens du transfert.

### **II.3.2.2 La transmission d'un octet**

Avant de placer les bits qui forment l'octet à transmettre sur le bus, le maître doit placer la ligne d'horloge SCL à 0. Tant que la ligne SCL est au niveau haut, la ligne SDA ne doit pas changer d'état, sinon cette condition sera interprétée comme la condition d'arrêt. La condition arrêt peut survenir même au milieu de la transmission d'un octet, pour abandonner la transmission et libérer le bus pour les autres circuits. Pour transmettre correctement les bits sur la ligne SDA, le maître doit donc tout d'abord placer la ligne SCL à 0. Ensuite, le maître peut placer la ligne SDA au niveau correspondant au bit à transmettre et replacer la ligne SCL au niveau 1 pour indiquer que le bit est présent sur la ligne SDA. La même opération va se répéter autant de fois que nécessaire pour transmettre les 8 bits de données.

### **II.3.2.3 La transmission d'une adresse**

Le nombre de composants qu'il est possible de connecter sur un bus I<sup>2</sup>C étant largement supérieur à deux, le maître doit pouvoir choisir quel esclave est censé recevoir les données. Dans ce but, le premier octet que transmet le maître n'est pas une

donnée mais une adresse. Le format de l'octet d'adresse est un peu particulier puisque le bit D0 est réservé pour indiquer si le maître demande une lecture à l'esclave ou bien au contraire si le maître impose une écriture à l'esclave.

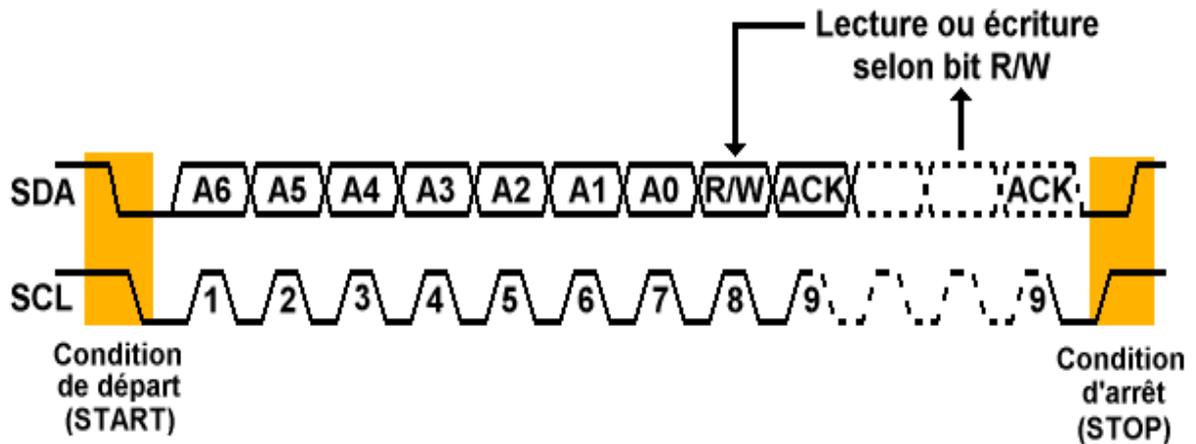


Figure II.6 transmission d'une adresse

Chaque circuit connecté au bus I<sup>2</sup>C possède une adresse, qui doit être unique. Une fois l'adresse envoyée sur le bus, l'esclave concerné doit répondre en plaçant le bit ACK à 0. Si le bit ACK vaut 1, le maître comprend qu'il y a une erreur de sélection et il génère la condition arrêt. En revanche, si le bit ACK vaut 0, le maître peut continuer les opérations.

#### II.4 Le capteur de la température DS1621

Dans de nombreux domaines ayant trait au quotidien et à la technique, la mesure de température est une tâche de mesure très importante. En concurrence avec les processus analogiques classiques, les capteurs numériques « intelligents » capables de faire plus que de mesurer uniquement la température, prennent une place de plus en plus importante.

Le DS1621 de Dallas Semi-conducteur en est l'exemple-type. La tâche première d'un capteur de température numérique moderne est restée, bien entendu, toujours de mesurer la température instantanée. Pour ce faire, la puce mesure sa propre température qui est pratiquement celle de son environnement direct, la convertit en une valeur numérique et l'envoie, en série vers un microcontrôleur. L'utilisateur n'a pas à se soucier de toute l'électronique requise pour l'étalonnage, l'amplification des tensions de température à

base d'amplificateurs opérationnels, la linéarisation des courbes et tout le reste vu que la puce s'en charge elle-même.

#### II.4.1 Caractéristiques techniques

- Plage de mesure de température allant de  $-55$  à  $+125$  °C.
- Résolution de  $0,5$  °C.
- Valeur de mesure de température à 9 bits signée et valeur absolue.
- Durée de conversion de 1 s.
- Fonction de thermostat programmable par l'utilisateur.
- Interface sérieuse bifilaire (I2C) avec connexion en collecteur ouvert.

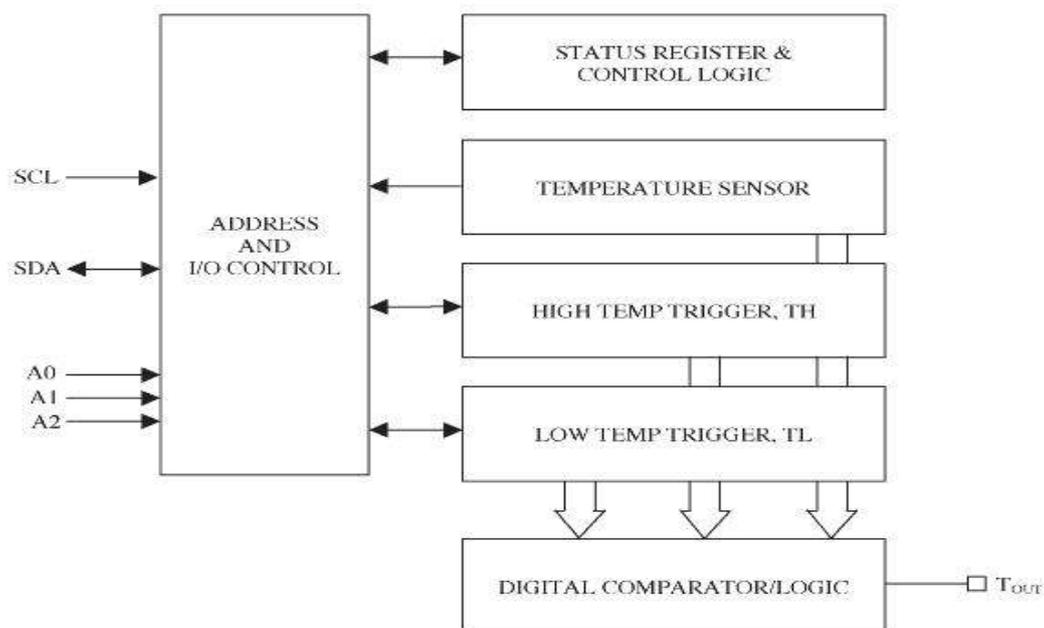
Température	Sortie(Binaire)	Sortie(Hexadécimal)
+125°C	01111101 00000000	7D00h
+25°C	00011001 00000000	1900h
+½°C	00000000 10000000	0080h
+0C	00000000 00000000	0000h
-125°C	11111111 10000000	FF80h
-25°C	11100111 00000000	E700h
-55°C	11001001 00000000	C900h

**Tab II.1 Correspondance température/données**

#### II.4.2 Principe de fonctionnement

La fonction primaire du DS1621 est la mesure de température. On retrouve, en figure II.7, le synoptique des blocs de fonctions disponibles dans ce circuit. Le DS1621 mesure la température par comptage, la limite d'une fenêtre de temps, du nombre de cycles d'horloge d'un oscillateur à coefficient de température faible. La largeur de ladite fenêtre est, à son

tour, définie à l'aide d'un oscillateur à coefficient de température élevé. Le compteur est calibré pour une valeur correspondant à une température de  $-55\text{ }^{\circ}\text{C}$ . Lorsque le compteur atteint, à l'intérieur de la fenêtre la valeur 0, le registre de température, qui lui aussi se trouve à une valeur de  $-55\text{ }^{\circ}\text{C}$ , est incrémenté pour indiquer que la température dépasse ces  $-55\text{ }^{\circ}\text{C}$  de référence. Le compteur décompte, à la fréquence de l'horloge, jusqu'à ce qu'il arrive à zéro pour ensuite recommencer au début au cas où la limite de la porte de temps ne serait pas encore atteinte. Une différence de température de  $1\text{ }^{\circ}\text{C}$  ne correspond pas toujours au même nombre de cycles d'horloge de la porte de temps. En effet, le compteur se voit appliquer, par l'accumulateur de pente (SLOPE ACCUMULATOR), un offset (différent) pour chaque différence de température de  $1\text{ }^{\circ}\text{C}$ , ceci en vue de compenser le comportement non linéaire de l'oscillateur, cette correction automatique permettant d'atteindre une résolution en température de  $0,5\text{ }^{\circ}\text{C}$ . La mesure de température est stockée dans le registre de température sous la forme d'une valeur à 9 bits. Comme nous le montre le **tableau II.1**, la plage de température du DS1621 va de  $-55$  à  $+125\text{ }^{\circ}\text{C}$  par pas de  $0,5^{\circ}\text{C}$ . On peut recueillir la valeur de température momentanée par le biais de l'instruction READ TEMPERATURE, cette information transitant par l'interface bifilaire sous la forme soit d'un seul octet, la température ayant alors une résolution de  $1^{\circ}\text{C}$ , soit sous forme de 2 octets pour une résolution de  $0,5^{\circ}\text{C}$  l'octet de poids fort (MSB) en premier, l'octet de poids faible (LSB) déterminant quant : lui le bit de poids faible ( $0,5\text{ }^{\circ}$ ). Les 7 bits restants du LSB (*Least Significant Byte* = octet de poids faible dans ce cas-là) sont mis à zéro.



**Figure II.7 Synoptique des blocs fonctionnels intégrés dans le DS1621.**

### II.4.3 Instructions de mesure de température

Instruction	Description	Valeur (Hex)	Ordre des octets
Read Temperature	Extrait du registre de température la valeur de température la plus récente	AA	2 octets
Read Counter	Extrait le reste du compteur dans la fenêtre de comptage	A8	1 octet
Read Slope	Extrait la valeur de l'accumulateur de pente	A9	1 octet
Start Convert T	Lance une conversion de température	EE	SANS
Stop Convert T	Arrête une conversion de température	22	SANS
<i>Access Config</i>	<i>Lit/Écrit des données du/dans le registre de configuration</i>	<i>AC</i>	<i>Lit/Écrit 1 octet de données</i>

**Tab II.2 Instructions de mesure de température**

Chaque communication requiert une condition de début et se termine par un bit d'arrêt. Le DS1621 signale la réception de chacun des octets par un acquittement (Acknowledge). Le bit de début est suivi d'un octet d'adresse prenant la forme de 4 bits mis à 1001 les 3 suivants se trouvant aux niveaux correspondant aux entrées d'adresse A2, A1 et A0. Le dernier bit paramètre la direction de la communication (R/W= Lecture/Écriture).

L'octet d'adresse est suivi par un octet d'instruction. Le DS1621 connaît les instructions de lecture et d'écriture ainsi que leurs codes respectifs. Les deux instructions de conversion ne comportent qu'un octet d'adresse et d'instruction.

Les instructions d'écriture sont suivies par un ou deux octets de données. Dans le cas des instructions de lecture la structure se complique quelque peu, vu que le circuit intégré doit, à nouveau, recevoir un octet d'adresse, ceci en vue de positionner le bit R/W sur Read (lecture). Ce n'est qu'après cette opération que pourra se faire la lecture des deux octets de donnée.

## **II.5 microcontrôleur pic 16F877**

Le 16F877 est un microcontrôleur de la série « MID RANGE» qui se prête particulièrement bien à la programmation en C. Les PIC de la série inférieure sont un peu justes en performance et en capacité mémoire pour accueillir un programme issu d'un compilateur C ; mieux vaut les programmer en assembleur. Les gammes supérieures (16 ou 32 bits) supportent sans problème la programmation en C, mais comme se sont des circuits plus complexes, le 16F877 a une mémoire programme de taille suffisante (8K), nombreux périphériques intégrés, fréquence de fonctionnement allant jusqu'à 20 MHz.

### **II.5.1 Le cœur du PIC : le microprocesseur**

Un microcontrôleur, c'est avant tout un microprocesseur, une unité de traitement logique qui effectue l'une après l'autre les opérations contenues dans un microprogramme stocké en mémoire (la mémoire FLASH).

Il est essentiellement composé de l'ALU « Unité Arithmétique et Logique » qui effectue les opérations sur les données, le registre de travail « W reg. », le multiplexeur « MUX », le registre de statut « status reg », le registre « FSR reg » utilisé pour l'adressage indirect (en assembleur), le multiplexeur d'adresse « Addr mux », le compteur programme « Program Counter » qui pointe les instructions à exécuter, la pile à 8 niveaux, le registre d'instructions « Instruction reg », ainsi que les différents bus qui relient tous ces éléments entre eux.

### **II.5.2 La mémoire**

Il existe trois types de mémoire.

#### **II.5.2.1 La mémoire FLASH**

La Mémoire Programme de type FLASH sur le 16F877 a une Capacité de 8Koctets.

C'est dans celle-ci qu'est stocké le programme du PIC. Après compilation de code, le compilateur génère un fichier « .hex », une suite de codes hexadécimaux. Celui-ci est transféré ensuite dans la mémoire programme du PIC à l'aide du programmeur.

Cette mémoire n'est pas reliée au bus de données (DATA Bus), sa vocation est de stocker le programme du PIC, mais pas les variables de programme.

Le gros avantage de la mémoire FLASH c'est que nous pouvons la réécrire, donc implémenter un nouveau programme dans le PIC.

### **II.5.2.2 La mémoire RAM**

La Mémoire RAM fait partie de la zone d'adressage des données. Elle comprend tous les registres spéciaux permettant de contrôler le cœur du PIC ainsi que ses périphériques. Elle contient également des cases mémoires à usage générique dans lesquelles pourront être stockées les variables de nos futurs programmes.

### **II.5.2.3 La mémoire EEPROM**

L'EEPROM est plutôt une mémoire de stockage de données à long terme, alors que la RAM est utilisée pour les variables du programme. Sur le PIC 16F877, on a 256 octets d'EEPROM disponible. Les mémoires de type EEPROM sont limitées en nombre de cycles d'effacement / écriture. Ce nombre de cycle est tout de même de l'ordre du million pour le PIC, mais si on l'utilisait pour stocker des variables modifiées plusieurs milliers de fois par secondes, cette limite pourrait être atteinte plus vite qu'on ne le croit.

### **II.5.3 Les Ports d'Entrées/Sorties**

Le PIC 16F877 est doté de 5 ports, les broches du PORTA servent également au convertisseur Analogique/Numérique, les broches du PORTD au Port Parallèle Esclave. Il faudra faire des choix au moment de la conception du schéma électronique. On voit également que les ports B, C et D ont 8 lignes d'entrée/sortie, alors que le port A n'en a que 6 et le port E que 3. Ils sont tous connectés au bus de donnée (DATA BUS), on pourra donc librement les adresser pour y lire ou écrire des données, On peut configurer les entrées/sorties de chaque port en entrée ou en sortie, grâce à un registre spécial dédié à chaque port.

### **II.5.4 Le Port Parallèle Esclave PSP « Parallel Slave Port »**

Le Port Parallèle Esclave est un port 8 bits permettant d'interfacer le PIC avec, par exemple, un autre microprocesseur. Les données transitent via les lignes PSP0 à PSP7, qui physiquement utilisent les mêmes broches que le PORTD. Le flux de données est contrôlé par les lignes RD, WR et CS qui correspondent aux broches du PORTE.

### **II.5.5 Le circuit de génération d'horloge**

Pour faire fonctionner notre PIC, on a besoin d'un circuit de génération d'horloge. Avec les PIC «MID RANGE», plusieurs options sont possibles : circuit RC (résistance + condensateur), ou circuit résonateur céramique, ou oscillateur à quartz... Le mode de fonctionnement est déterminé au moment de la programmation du PIC par des « bits de configuration » qui sont en fait des emplacements de mémoire non-volatile. Pour générer un « cycle d'instruction », il faut 4 cycles d'horloge. Autrement dit, si notre quartz oscille à 20 MHz, les instructions du programme s'exécutent à une cadence de 5 MHz.

### **II.5.6 Le convertisseur Analogique – numérique**

Le convertisseur A/D convertit le signal analogique présent sur une de ses 8 entrées en son équivalent numérique, codé sur 10 bits. Les broches AN2 et AN3 peuvent être utilisées comme références de tension ou comme entrées analogiques standard, les références de tension étant dans ce dernier cas prises sur les tensions d'alimentations du PIC.

### **II.5.7 USART**

L'USART ou « Universal Synchronous Asynchronous Receiver Transmitter » est l'un des deux modules de communication série du PIC ou SCI « Serial Communication Interface ».

Comme son nom l'indique, elle peut établir une liaison synchrone ou asynchrone, recevoir et transmettre des données, selon la manière dont elle est configurée.

L'USART permet de communiquer avec le reste du monde : un ordinateur ou tout autre matériel équipé d'une interface série RS232, des circuits intégrés convertisseurs Numérique/Analogique ou Analogique/Numérique, des EEPROMs série...etc.

L'USART peut être configurée selon 3 modes :

- Asynchrone (Full Duplex)
- Synchrone Maître (Half Duplex)
- Synchrone Esclave (Half Duplex)

### **II.5.8 Le SSP**

Le module SSP « Synchronous Serial Port » est la deuxième interface de communication série du PIC. En fait, il s'agit d'un port Maître, donc l'appellation exacte est plutôt MSSP « Master Synchronous Serial Port ».

Il est utile pour communiquer avec d'autres modules ou microcontrôleurs, des EEPROMs séries, des registres à décalage, des afficheurs, des convertisseurs A/N ou N/A...

Il peut fonctionner dans deux modes de communication :

- Le mode SPI (Serial Peripheral Interface)
- Le mode I<sup>2</sup>C (Inter-Integrated Circuit)

## **II.5.9 Les Interruptions**

Un certain nombre d'événements sont susceptible de générer des interruptions : fin de conversion de signal analogique, écriture en mémoire EEPROM terminée, débordement de Timer, USART, SSP, changement d'état d'une entrée de port...

Si votre programme comprend une routine de gestion de cette interruption, le déroulement normal s'interrompt et la routine de gestion de l'interruption est exécutée.

Le gros avantage de ce mécanisme, c'est que le microcontrôleur peut exécuter différentes tâches sans être obligé de scruter en permanence si telle ou telle action a eu lieu.

## **II.6 Le max232**

Les cartes électroniques à base de microcontrôleurs fonctionnent très souvent avec des niveaux TTL soit 0-5Volt, 0V pour le niveau 0 et 5Volt pour le niveau 1. Brancher donc directement une ligne RS232 sur un microcontrôleur n'aurait donc aucun sens et pourrait aussi endommager le système en imposant des tensions de 25volt. Pour rendre compatible une ligne RS232 avec une carte de ce type il existe un composant très simple, le max232. En regardant son schéma interne dans la figure (2.7), nous constatons directement qu'il est premièrement doté d'un convertisseur de tension, au travers des capacités C1 et C3 il génère une tension de 10Volt depuis les 5Volt (doubleur de tension), et au moyen des capacités C2 et C4 il génère une tension de -10Volt à partir de la tension de 10Volt. Il est bien sur évident que la puce est munie de tout un système, avec un oscillateur, des diodes et ... afin d'intégrer ce convertisseur DC-DC.

La connexion avec le port série se fait via la broche Rx (réception) et Tx (émission) et RTS qui donne la possibilité d'émission. Les broches 6, 1 et 4 seront reliées ensemble, Et nous n'oublierons pas de brancher la masse (broche 5) avec la masse du circuit. Les broches 7 et 9 ne sont pas nécessaires dans notre cas.

TOP VIEW

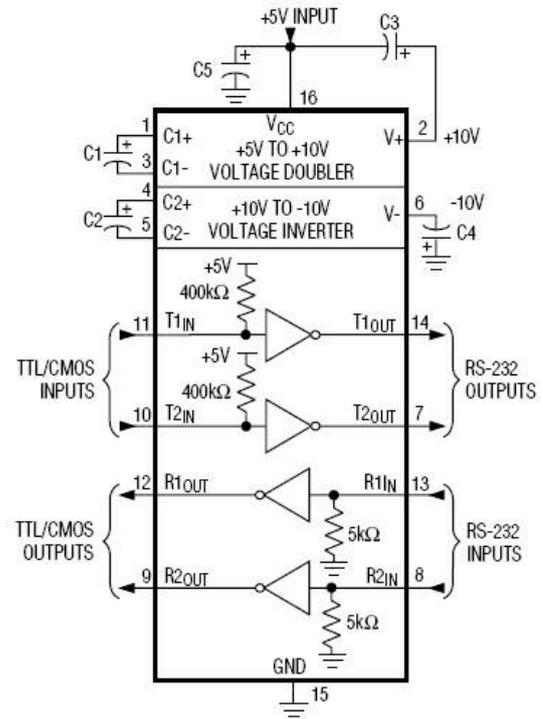
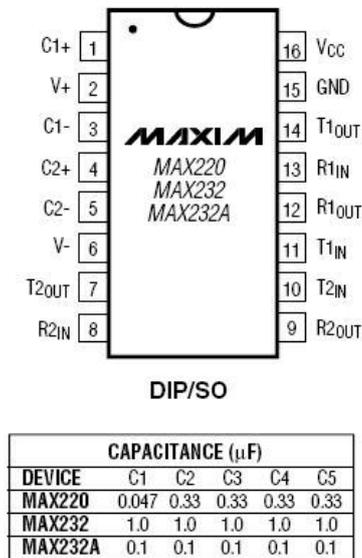


Figure II.8 le circuit max232

## II.7 La réalisation de la carte d'interface

Pour que la carte d'interface fonctionne correctement il faut que les liaisons entre les entrées sorties de différents éléments de la carte d'interface « PIC, MAX232 » soient correctes des deux parties de la carte.

### II.7.1 partie matérielle

Pour réaliser la carte d'interface on doit concevoir son schéma électrique, pour cela nous avons exploité le logiciel « PROTEUS », ce dernier permet aussi d'effectuer la simulation de notre montage.

« Voir l'annexe chapitre II ».

### II.7.2 partie programmation

Cette partie consiste à écrire le programme du microcontrôleur « PIC » qui permet de communiquer avec PC via RS-232 et avec le capteur via I2C.

En utilisant le compilateur « CCS C COMPILER » nous avons développé ce programme en langage C

## **II.8 La carte de puissance**

La carte de puissance consiste à contrôler l'alimentation du four, en choisissant le type et la valeur de la tension, les valeurs des résistances et l'élément de contrôle.

### **II.8.1 La réalisation de la carte de puissance**

Pour contrôler la tension on utilise une tension continue, on obtient cette tension à partir d'une source alternative de 150V, le redressement double alternance et le filtrage. On applique cette tension sur les deux résistances du four en série par un relai de 5V, ce Relai est commandé par des signaux PWM envoyé par le PC.

« Voir l'annexe chapitre II ».

## **II.9 Conclusion**

Dans ce chapitre nous avons décrit la carte d'interface permettant de connecter le capteur de la température au PC à travers une carte basé sur un microcontrôleur PIC16F877. l'envoi de la commande vers les résistances de chauffage s'effectue grâce à un étage de puissance à base d'un relai permettant la commutation de l'alimentation des résistances.

## **Chapitre III**

### **Identification du système**

#### **III.1 Introduction**

L'Automatique consiste en l'étude des systèmes réels des différentes disciplines scientifiques (Electronique, mécanique, thermique, chimie, écologie, biologie, économie, physique, cosmologie...), en vue de l'analyse, de la prédiction, de la surveillance, de la commande, et de l'optimisation des systèmes.

La condition nécessaire pour cela est l'obtention d'un modèle mathématique du système réel.

Un système est un objet dans lequel des variables de différent types interagissent et produisent des signaux observables. Lorsque le modèle du système n'est pas connu, il est nécessaire de procéder à son identification.

## **III.2 Généralité sur la modélisation**

La modélisation d'un processus consiste à déterminer la structure des équations qui régissent le comportement de ce système, et aussi, a fixer, a priori les valeurs de ses paramètres (température, masse, inertie, capacité, résistance, frottement...).

La notion de modèle mathématique d'un système, d'un processus ou d'un phénomène, est un concept fondamental. Il existe une multitude de types de modèles, chacun étant destiné à une application particulière. Nous pouvons les décliner en deux grandes catégories :

- Les modèles de connaissance (basés sur les lois de la physique, de la chimie...), donnent une description complète des systèmes et sont utilisés pour la simulation et la conception des procédés. Ce sont souvent des modèles complexes.

Si le modèle de connaissance s'avère trop peu précis ou impossible à obtenir, on peut alors chercher un modèle de comportement entrée - sortie. On parle alors de modèles dynamiques de commande.

- Les modèles dynamiques de commande donnent la relation entre les variations d'entrée d'un système et les variations de la sortie, ils sont de deux sortes :
  - Modèles non paramétriques (réponse fréquentielle, réponse à un échelon)
  - Modèles paramétriques (fonction de transfert, équations différentielles).

### **III .2.1 Modèles non paramétriques**

#### **III .2.1.1 Analyse fréquentielle**

Les essais expérimentaux harmoniques sont rarement employés, car ils sont longs et fastidieux. Des essais réalisés avec une pulsation comprise entre «  $0.1 \cdot \omega_c$  » et «  $10 \cdot \omega_c$  », avec  $\omega_c$  est la fréquence de coupure.

Avec une grande constante de temps pour le processus, cela peut durer des heures, d'autant plus qu'il est parfois nécessaire de recommencer deux a trois fois l'expérimentation pour chaque pulsation (on doit attendre le temps d'établissement pour chaque mesure). Les outils classiques de l'automatique sont utilisés pour l'analyse de la réponse en fréquence du système à identifier : Bode, Nyquist, Black...

#### **III .2.1.2 Réponse impulsionnelle :**

Un système linéaire invariant est entièrement caractérisé par sa réponse à une impulsion de Dirac, dont la transformée de Fourier est constante sur tout le spectre. La transformée de Fourier de la réponse impulsionnelle du système est sa fonction de transfert. Un Dirac « réel », peut être approché, du point de vue de ses effets, par une impulsion brève, de durée

finie et d'énergie limitée. La réponse du système, est en conséquence faible et bruitée, donc difficilement exploitable. Ainsi, pour atteindre la réponse impulsionnelle  $h(t)$  par voie indirecte, il est possible d'appliquer les techniques de corrélation. Le processus est alors excité par une Séquence Binaire Pseudo Aléatoire (S.B.P.A.).

**Remarque :**

Les diverses méthodes non paramétriques (analyse fréquentielle, réponse indicielle, impulsionnelle...) peuvent aider à choisir la structure du modèle. Néanmoins, les méthodes qui prennent en compte un modèle paramétrique du processus, sont préférées, car elles sont plus simples à mettre en œuvre et à utiliser.

**III.2.2 Modèles paramétriques :**

Il existe deux principales classes de méthodes paramétriques :

- Les méthodes paramétriques graphiques ou déterministes.
- Les méthodes paramétriques statistiques (études stochastiques).

**III.2.2.1 Les méthodes paramétriques graphiques (déterministes) :**

L'objectif est d'obtenir des modèles paramétriques à partir de modèles non paramétriques, type « réponse à l'échelon » ou « réponse en fréquence ».

Les méthodes graphiques sont simples, en revanche :

- Elles sont limitées dans le choix des modèles,
- Elles sont sensibles aux perturbations (et il n'est pas possible ici de modéliser les perturbations).
- Il n'y a pas de procédures explicites de validation des modèles
- Les procédures d'identification graphiques sont aussi imprécises.

**III.2.2.2 Les méthodes paramétriques statistiques :**

Il s'agit de mettre en place des algorithmes d'estimation automatique des paramètres des modèles échantillonnés des procédés en utilisant un calculateur numérique. Le traitement des données entrées / sorties peut se faire à l'aide d'algorithmes non récursifs (Traitement en bloc des fichiers de données obtenues sur un horizon de temps) ou récursifs (Traitement pas à pas des données) le second type d'algorithme, présente l'avantage d'être appliqué en « temps réel » et nécessite moins de ressources mémoire en terme de calculs.

L'identification paramétrique statistique peut être décomposée en quatre étapes :

- Acquisition des entrées-sorties
- Estimation (choix) de la structure du modèle (complexité)
- Estimation des paramètres du modèle
- Validation du Modèle identifié

L'identification paramétrique statistique est généralement basée sur l'erreur de prédiction, qui est l'erreur entre la sortie du procédé  $y(t)$ , et la sortie  $\hat{y}(t)$  prédite par le modèle. Cette erreur de prédiction est utilisée par un Algorithme d'Adaptation Paramétrique (A.A.P.), qui à chaque instant d'échantillonnage, va modifier les paramètres du modèle afin d'en minimiser l'erreur.

L'entrée appliquée au système est généralement une Séquence Binaire Pseudo Aléatoire (S.B.P.A.). Cette approche moderne de l'identification :

- élimine les défauts des autres méthodes mentionnées précédemment
- permet de modéliser (et supprimer si besoin) les perturbations et bruits de capteurs, et permet un traitement aisé des signaux (analyse spectrale des signaux).

Il existe plusieurs algorithmes de l'identification statistique parmi ces algorithmes :

l'algorithme des moindres carrées.

### III.2.2.3 L'algorithme des moindres carrées

La méthode des moindres carrés a été introduite par Karl Gauss en 1809. Elle a été à la base de toutes les méthodes d'identification et d'estimation des paramètres, cette méthode est basée sur la minimisation d'une fonction quadratique (J) définie comme :

$$J = \sum_{k=1}^N (e(k))^2 \quad (\text{III-1})$$

Où  $e(k)$  représente l'erreur de prédiction commise sur l'estimation.

#### III.2.2.3.1 Identification du modèle ARMA

La modélisation auto-régressive à moyenne ajustée d'ordre (n, m) noté ARMA (n, m) est définie par l'équation aux différences suivante :

$$\sum_{i=0}^N a_i y(t-i) = \sum_{i=0}^M b_i u(t-i) + e(t) \quad (\text{III-2})$$

Avec  $a_0 = 1$

$u(t)$  : Entrée du système

$y(t)$  : Sortie du système

$e(t)$  : Bruit blanc.

En développant l'équation (III-2), on obtient :

$$y(t) = -a_1 y(t-1) - a_2 y(t-2) - \dots - a_n y(t-n) + b_0 u(t) + \dots + b_m u(t-m) + e(t) \quad (\text{III-3})$$

- **Détermination des paramètres**

Nous pouvons écrire l'équation (III.3), sous une forme matricielle :

$$y(t) = \theta^T \varphi(t) + e(t) \quad (\text{III-4})$$

Posons pour la commodité des notations :

$$\theta^T = [a_0 \ a_1 \ a_2 \ \dots \ a_n \ ; \ b_0 \ b_1 \ b_2 \ \dots \ b_m]$$

$$\varphi^T(t) = [ [-y(t-1) \ \dots \ y(t-n) \ ; \ u(t) \ \dots \ u(t-m) ] ]$$

$\theta^T$  : vecteur des paramètres à identifier

$\varphi^T(t)$  : vecteur des données.

$y(t)$  : sortie du système

$e(t)$  : bruit blanc affecté au système

T : désigne la transpose d'un vecteur ou d'une matrice.

On définit l'erreur de prédiction comme étant la différence entre la sortie du système et la sortie du modèle :

$$e(t) = y(t) - \hat{y}(t) \quad (\text{III-5})$$

Sachant que :

$$\hat{y}(t) = \theta^{*T} \varphi(t) \quad (\text{III-6})$$

avec  $\theta^{*T}$  représente les paramètres estimés.

La méthode des moindres carrés est basée sur la détermination des meilleurs paramètres, c'est-à-dire ceux qui minimiseront un certain critère d'optimalité. Ce critère représente la somme des carrés des erreurs de prédictions :

$$J_N(\theta) = \frac{1}{N} \sum_{t=1}^N (e(t))^2 \quad (\text{III-7})$$

tel que N : nombre d'échantillons.

La minimisation du critère  $J_N(\theta)$  consiste à trouver :

$$\left[ \frac{\delta J_N}{\delta \theta} \right]_{\theta = \hat{\theta}^{(N)}} = 0 \quad (\text{III-8})$$

De l'équation (III.7), on déduit la solution optimale au sens des moindres carrés de la forme suivante :

$$\hat{\theta}^N = \left[ \sum_{t=1}^N \varphi(t) \varphi^T(t) \right]^{-1} \sum_{t=1}^N \varphi(t) y(t) \quad (\text{III-9})$$

Nous constatons que la matrice  $\varphi(t) \varphi^T(t)$  est grande, si le nombre d'échantillons N est important, d'où le calcul de son inverse n'est pas conseillé, pour cela on utilise l'estimation récursive du moindre carré.

- **Mise en œuvre de l'algorithme des moindres carrés récursif**

Pour la mise en œuvre de l'algorithme récursif, on pose :

$$R(t) = \sum_{k=1}^t \varphi(k) \varphi^T(k) = R(t-1) + \varphi(t) \varphi^T(t) \quad (\text{III-10})$$

D'après les équations (III.9) et (III.10), on a :

$$\hat{\theta}^{tT} = R^{-1}(t) \sum_{k=1}^t \varphi(k) y(k) \quad (\text{III-11})$$

$$\hat{\theta}^{tT} = R^{-1}(t) \left[ \sum_{k=1}^{t-1} \varphi(k) y(k) + \varphi(t) y(t) \right] \quad (\text{III-12})$$

$$\hat{\theta}^{tT} = R^{-1}(t) \left[ R(t-1) \hat{\theta}^{(t-1)T} - \varphi(t) \varphi^T(t) \hat{\theta}^{(t-1)T} + \varphi(t) y(t) \right] \quad (\text{III-13})$$

$$\hat{\theta}^{tT} = \hat{\theta}^{(t-1)T} + R^{-1}(t) \varphi(t) \left[ y(t) - \hat{\theta}^{(t-1)T} \varphi(t) \right] \quad (\text{III-14})$$

Un optimum, c'est à dire de calculer la dérivée : D'après cette dernière équation (III .18), on remarque que la solution des moindres carrés récursive contient le terme  $R^{-1}(t)$  qui nécessite une inversion matricielle à chaque instant  $t$ . Donc, on utilise le lemme d'inversion matricielle qui se présente sous la forme suivante [2] :

$$[B - CD^T]^{-1} = B^{-1} - B^{-1}CD^T B^{-1} [1 + D^T B^{-1} C]^{-1} \quad (\text{III-15})$$

On pose :

$$B=R(t-1) ; C= \varphi(t) ; D=\varphi^T,$$

D'où :

$$R^{-1}(t)=[R(t - 1)\varphi(t)\varphi^T (t)]^{-1} \quad (\text{III-16})$$

L'application du lemme d'inversion matricielle sur (III-16)

On obtient :

$$R^{-1}(t) = R^{-1}(t - 1) - R^{-1}(t - 1)\varphi(t)\varphi^T (t)R^{-1}(t - 1) \frac{1}{1+\varphi^T (t)R^{-1}(t-1)\varphi(t)} \quad (\text{III-17})$$

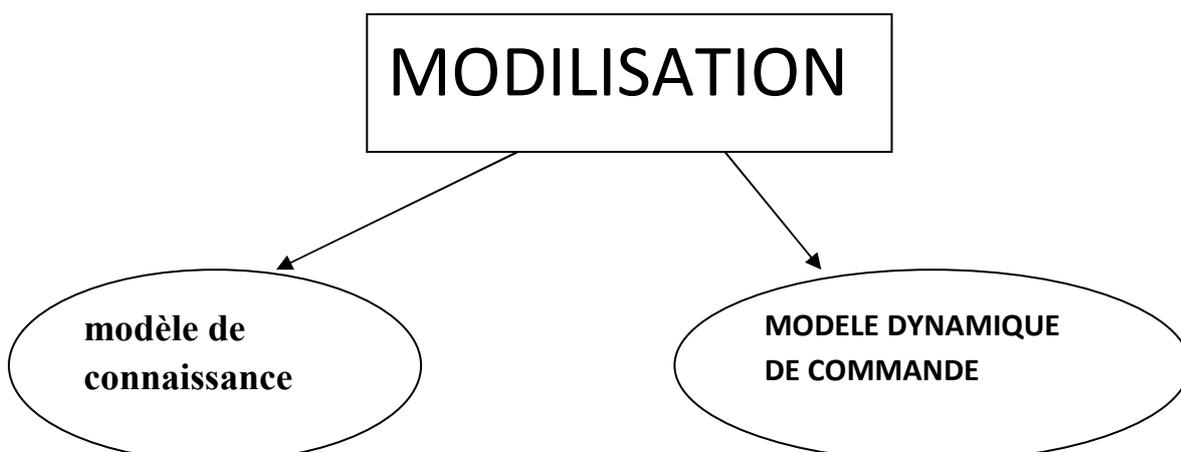
L'introduction de la matrice du gain d'adaptation  $P(t) = R^{-1}(t)$  , permet la mise en œuvre de l'algorithme des moindres carrés récursif MCR de la forme suivante :

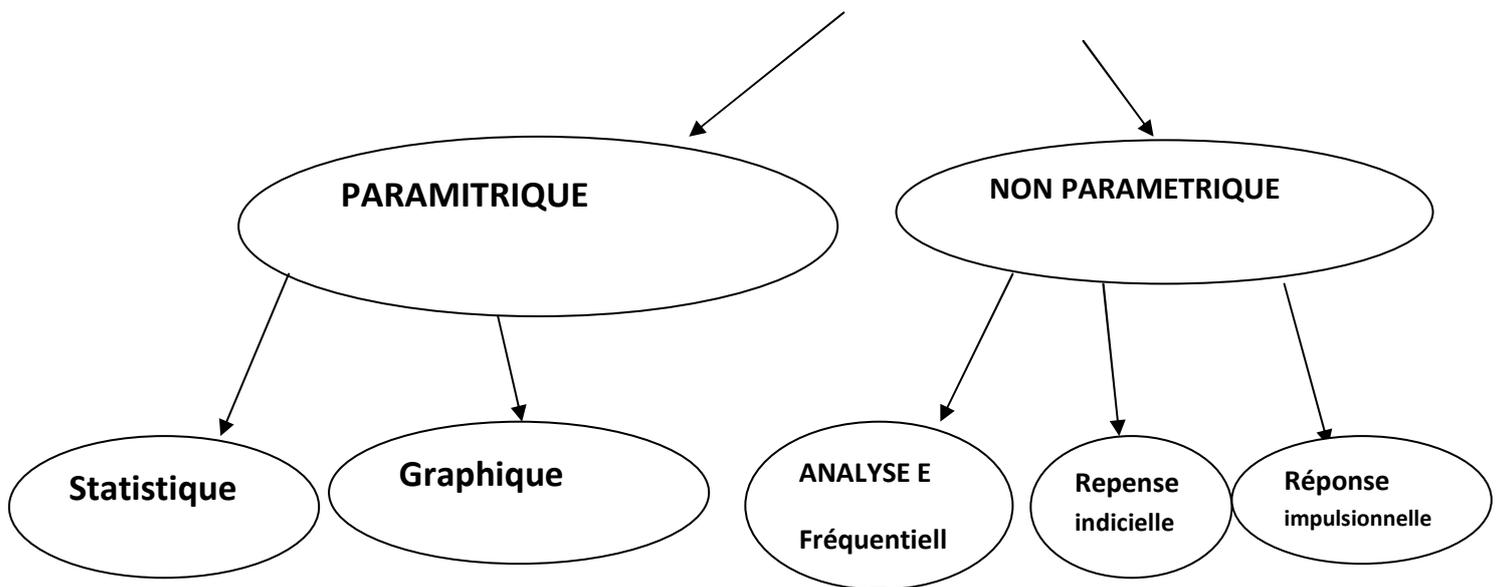
$$\theta(t) = \theta(t - 1) + p(t)\varphi(t)\varphi^T (t)[y(t) - \theta^T(t - 1)\varphi(t)] \quad (\text{III-18})$$

avec :

$$P(t) = P(t - 1) - P(t - 1)\varphi(t)\varphi^T (t)P(t - 1) \frac{1}{1 + \varphi^T (t)P(t - 1)\varphi(t)} \quad (\text{III - 19})$$

On peut résumer tous les approches de la modélisation par l'organigramme de la figure III.1





**Figure III.1 Organigramme de différents types de modélisation**

### **III .2.3 Les séquences binaires pseudo aléatoires (S.B.P.A)**

Le signal d'excitation doit être riche en fréquences pour balayer toutes les fréquences.

Ce signal nous permet de voir toutes les variations des systèmes et en conséquence une bonne identification des paramètres, ces caractéristiques peuvent se trouver dans un bruit blanc, en revanche on ne peut pas générer ce dernier comme signal d'excitation.

L'un des moyens de réaliser un signal aléatoire «bruit blanc » est la mise en œuvre d'une séquence Binaire Pseudo Aléatoires (en anglais PRBS : Pseudo Random Binary Sequence).

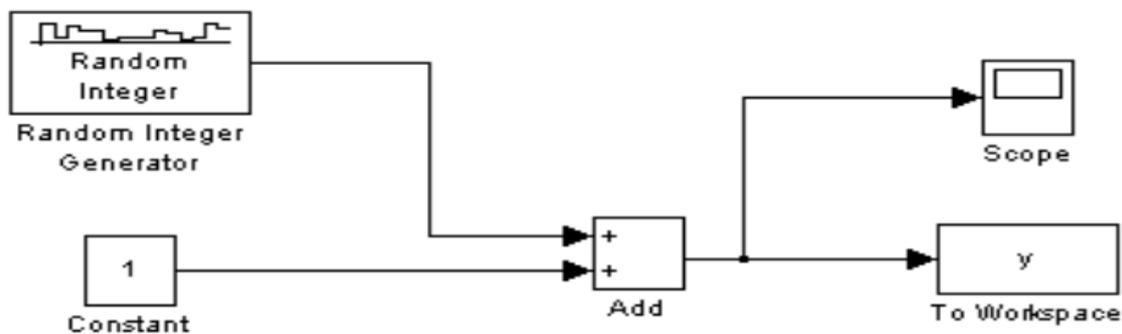
La largeur SBPA est un signal formé d'impulsions rectangulaires modulées aléatoirement en longueur, qui approxime un bruit blanc.

#### **III.2.3.1 choix des paramètres du SBPA**

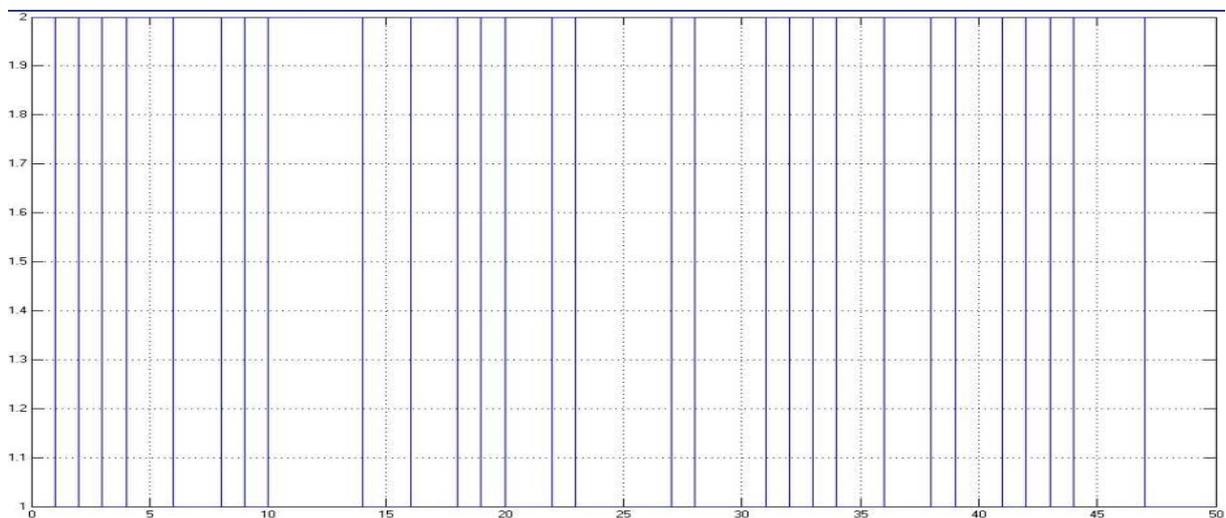
Pour bien identifier le gain statique de la procédé, il faut que la durée d'au moins une des impulsions (par exemple l'impulsion de durée maximale) soit supérieure au temps de montée du procédé.

D'autre part, pour balayer tout le spectre de fréquences, il faut que la longueur d'un essai soit au moins égale à la longueur de la séquence

L'amplitude de la SBPA doit être supérieure au niveau du bruit résiduel. Si le rapport signal/bruit est trop faible, il faut allonger la longueur de l'essai pour pouvoir obtenir une bonne estimation des paramètres. Notons que, dans de nombreuses applications l'augmentation significative du niveau de la SBPA n'est pas souhaitable à cause du caractère non linéaire des procédés à identifier (dans notre cas nous nous intéresserons à l'identification d'un modèle linéaire). Le bloque simulink illustré dans la figure(III.2) permet d'obtenir une SBPA.



**Figure III.2** génération de signal pseudo aléatoire.



**Figure III .3** une séquence pseudo aléatoire

### III.2.4 Choix du modèle

Le principe d'une identification paramétrique est d'extraire un modèle mathématique à partir d'observations. Le modèle doit permettre de calculer la sortie du procédé  $y$  à n'importe quel

instant  $t$  si les conditions initiales du système sont connues. Pour cela on peut se servir des valeurs des entrées aux instants présents et précédents ( $u(t)$ ,  $u(t - 1)$ , ...) et des valeurs précédentes de la sortie ( $y(t - 1)$ ,  $y(t - 2)$ , ...) dans le cas d'un modèle régressif.

Il est tout de même important d'avoir des connaissances basiques du système pour choisir un type de modèle adapté :

- Modèle possédant une entrée/une sortie (SISO) ou plusieurs entrées et plusieurs sorties (MIMO).
- Modèle linéaire ou non-linéaire (dans ce cas, qu'est-ce qui est non-linéaire en fonction de quoi).
- Modèle continu ou discret.
- Modèle régressif ou indépendant : pour un modèle régressif, la sortie à un instant  $t$ ,  $y(t)$ , dépend des instants précédents ( $y(t - i)$ ).
- Modèle stochastique ou déterministe.

En général, le modèle est représenté sous forme de fonction de transfert utilisant la Transformée en Z. L'identification nécessite une structure de modèle connu à priori pour venir identifier dans cette structure différents paramètres.

### **III.3 Etudes expérimentale**

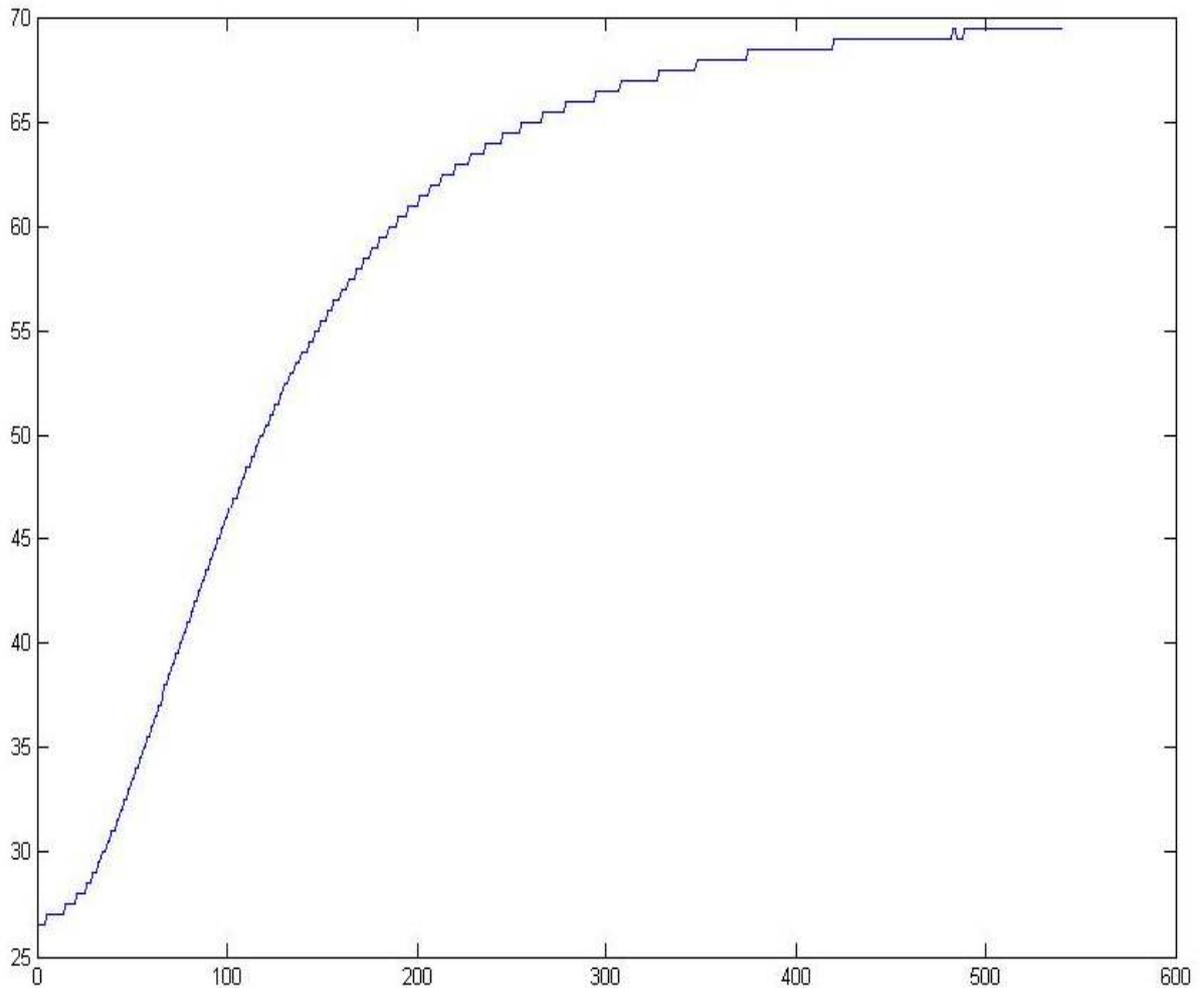
On commence cette étude par une modélisation non paramétrique en faisant la réponse indicielle qui nous permet de déterminer l'ordre du modèle et d'avoir une connaissance approximative sur les grandeurs du modèle pour passer ensuite à l'identification paramétrique pour identifier les paramètres du modèle déjà choisi et enfin valider notre modèle.

#### **III.3.1 La réponse indicielle**

On applique un échelon sur notre processus (une tension de 150V) et on reçoit à chaque seconde les mesures du capteur.

On constate que notre système est lent et l'information fournie par le capteur se répète plusieurs fois alors il faut augmenter la période d'échantillonnage

La figure III.4 représente la réponse indicielle avec un pas d'échantillonnage de 5 secondes



**Figure III. 4 Réponse indicielle avec pas d'échantillonnage  $T_e=5s$ .**

D'après la Figure III. 4 on peut modéliser le système par une fonction de transfert du premier ordre avec retard pure :

$$G(s) = \frac{k}{1+T_p s} e^{-Tr.s} \quad (\text{III.20})$$

Avec :

- $k$  : le gain statique
- $T_p$  : la constant du temps.
- $Tr$  : le temps de retard.
- Avec un temps de montée égale à 20.66mn

### **III.3.2 Identification des paramètres**

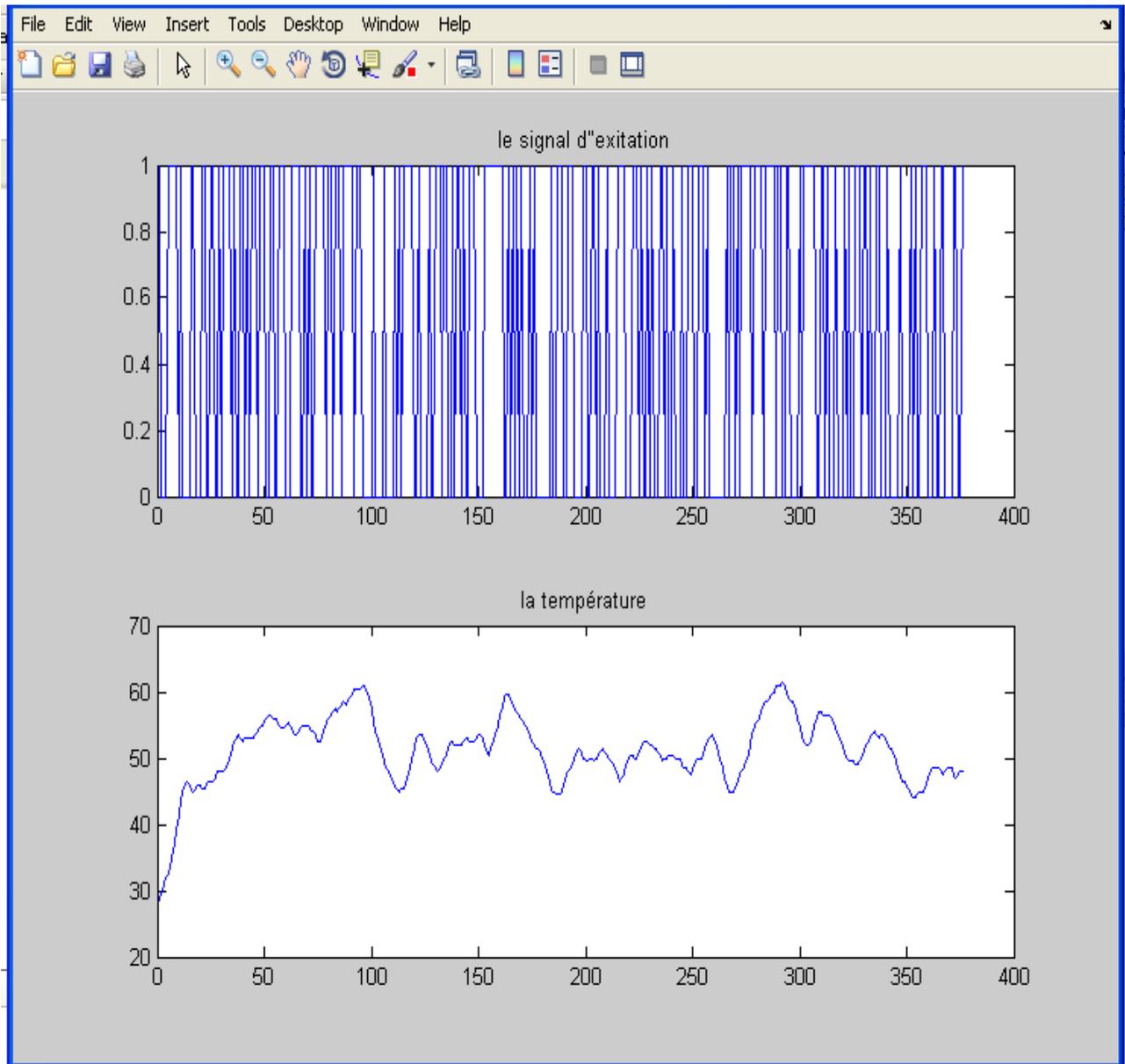
Passons maintenant à l'identification paramétrique du processus

Pour cela on a besoin d'un signal d'excitation qui permet de balayer toute les composantes fréquentielles, on met comme entrée une séquence binaire pseudo aléatoire.

Pour avoir le maximum de variation du système il faut bien choisir les paramètres de la SBPA.

La durée de l'expérience sera longue donc il faut que la SPBA soit aussi importante environ 500 échantillons, chaque échantillon va durer 1 min.

Le programme « voir annexe chapitre.III » permet d'envoyer le signal d'excitation et recevoir les données du capteur, cette opération va durer environ 8 heures afin d'avoir un nombre de données assez suffisant pour l'identification (environ 500 échantillons).



**Figure III.5 la réponse du système à une séquence SPBA**

La Figure III. 5 montre les variations de la température en fonction du signal d'excitation. On constate qu'il y a plusieurs variations sur la température qui va nous permettre de bien identifier les paramètres du modèle.

Pour effectuer l'identification des paramètres du modèle il existe dans matlab un toolbox qui facilite l'identification et la validation du modèle (System Identification Toolbox).

- On tape sur le workspace « ident » :

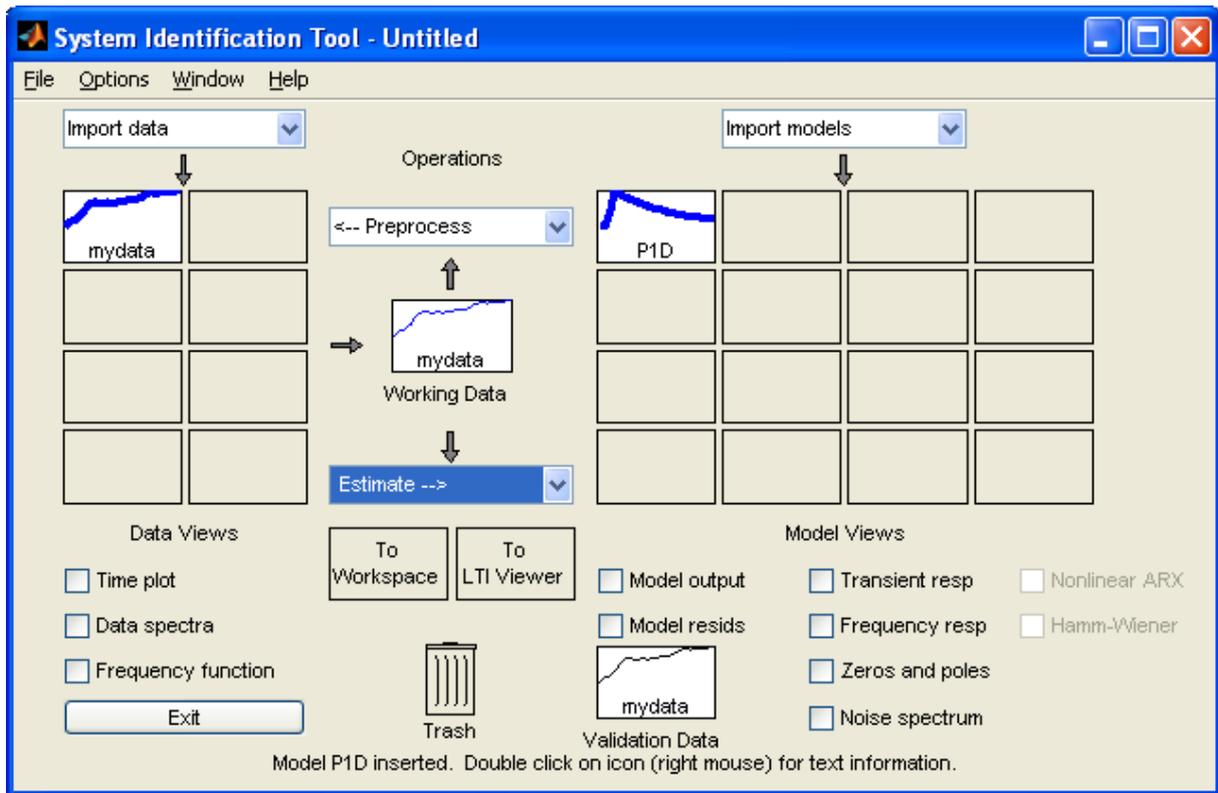
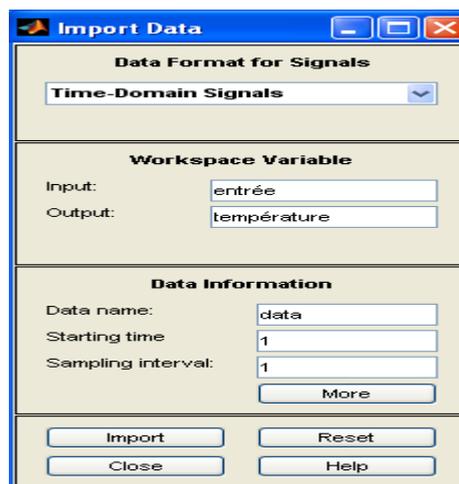
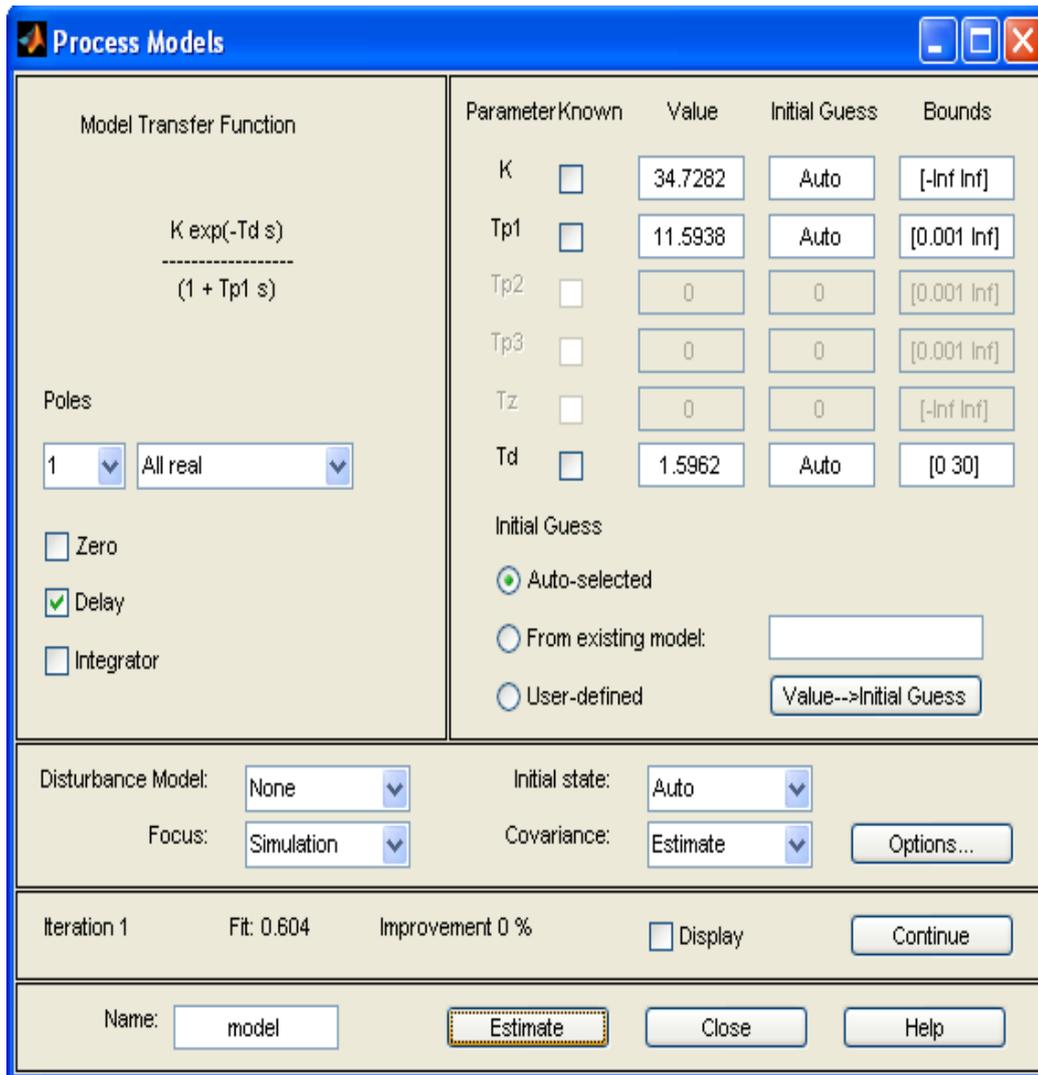


Figure III.6 la tool box de l'identification du système

- On importe les mesures qui se trouvent dans le workspace :



- On choisit le modèle du processus et on démarre l'estimation :

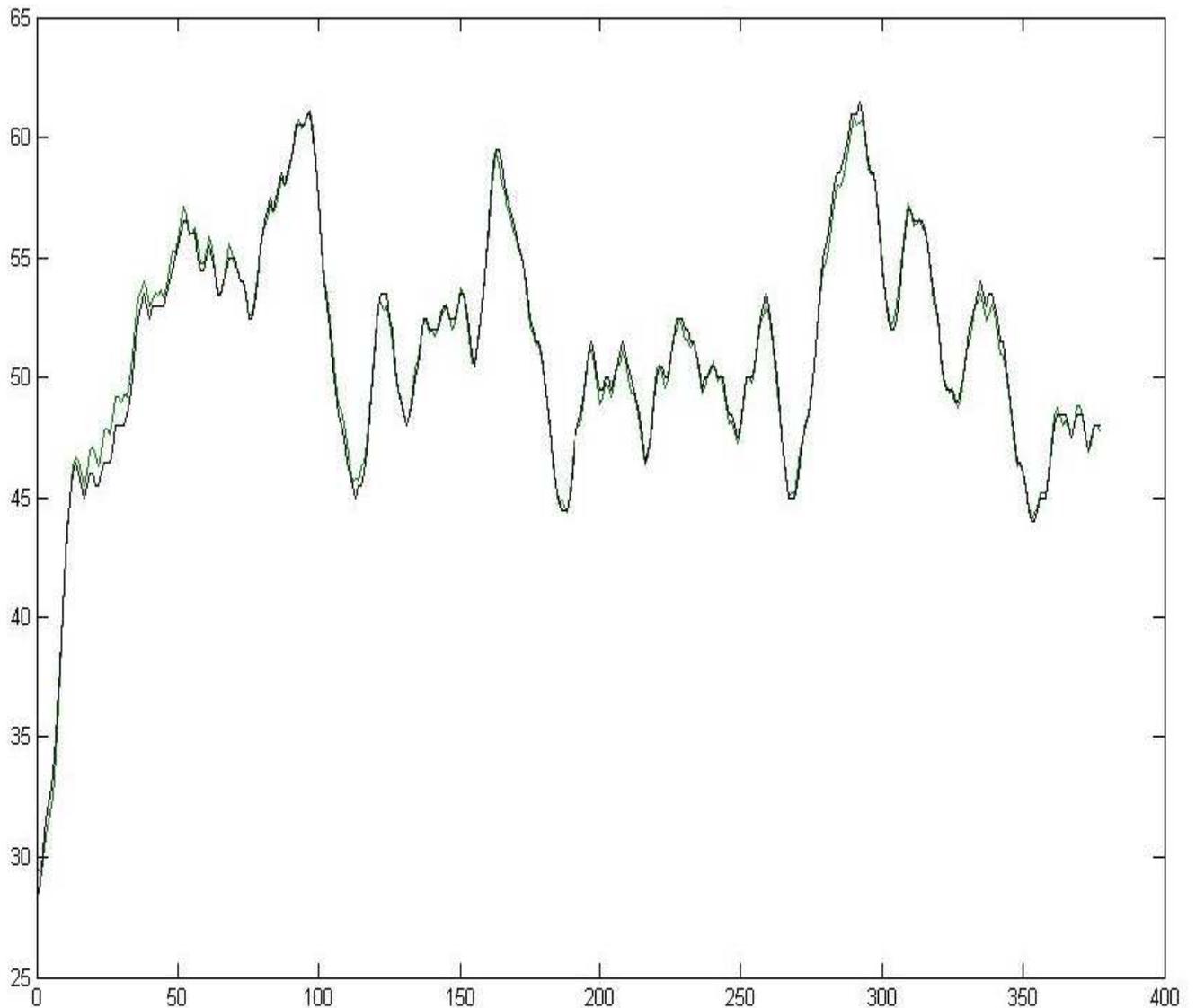


➤ La tool box donne le modèle de l'estimation suivant :

$$G(s) = \frac{k}{1 + T_p s} e^{-T_r s}$$

avec :

- $k = 34.728 \pm 0.027829$
- $T_p = 11.594 \pm 0.15677$
- $T_r = 1.5962 \pm 0.027412$



**Figure III.7 la sortie réelle et la sortie estimée**

### **III.3.3 Validation du modèle**

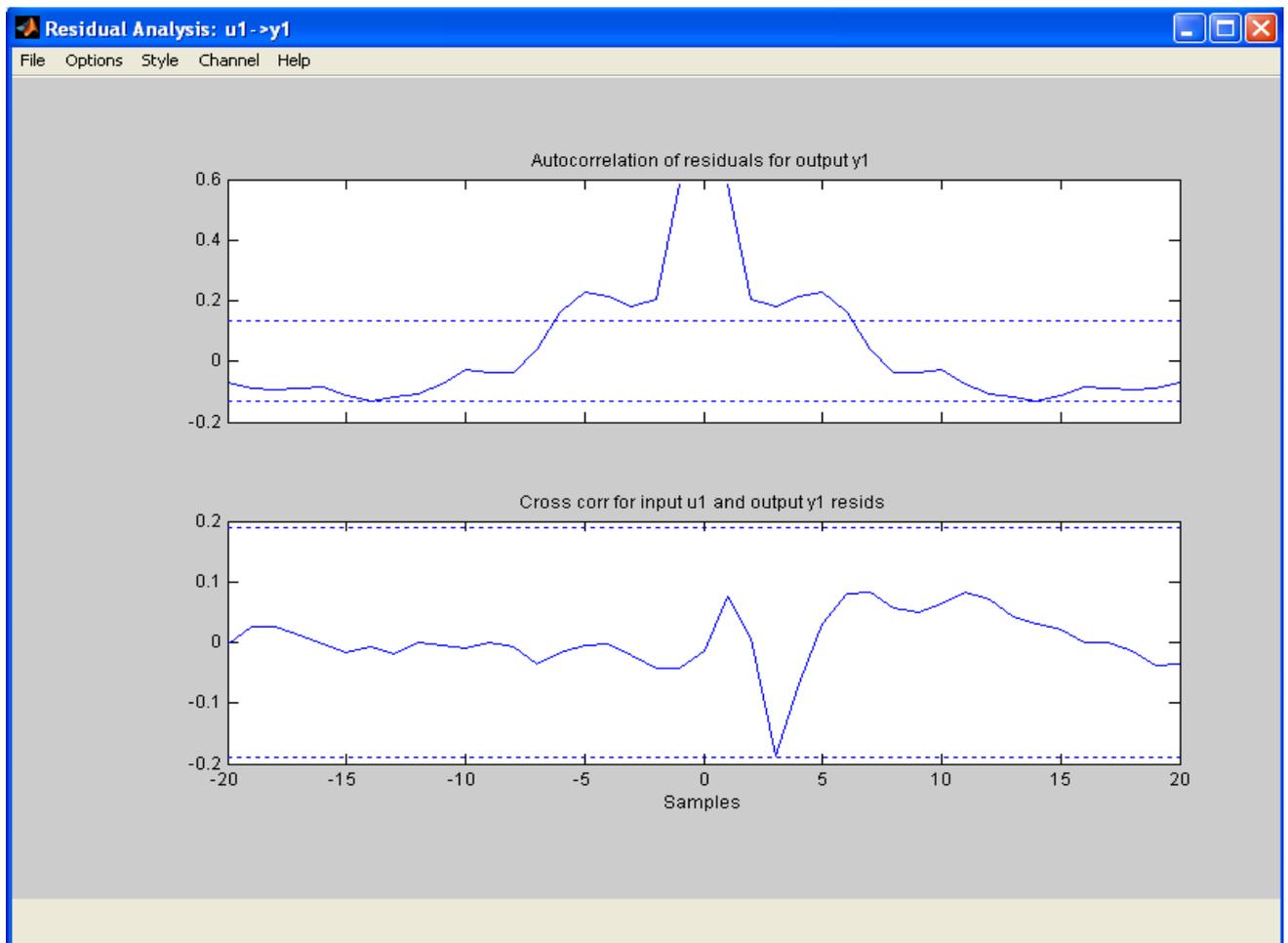
#### **III.3.3.1 analyse résiduelle**

Le résidu est la différence entre la sortie du modèle et les mesures réelles du processus donc le résidu représente la portion non concernée par le modèle.

La tool box « ident » fournit une analyse résiduelle qui nous permet de tester la validité du modèle, elle est composée de deux tests :

- test de témoignage : ce test consiste à calculer l'autocorrélation du résidu, pour un bon modèle l'autocorrélation de son résidu se trouve dans l'intervalle de confiance motionné sur la figure III.8, ceci indique que le résidu est incorréolé ou se comporte comme un bruit blanc.

- test d'indépendance ce test consiste à calculer l'intercorrélation entre l'entrée et le résidu, pour un bon modèle l'intercorrélation entre l'entrée et les résiduel doit s'approcher de Zéro dans l'intervalle de confiance, ceci indique que ce sont incorrelés .une corrélation indique que le model ne décrit pas comment une partie de la sortie est reliée a leur entrée correspondante par exemple un pic à l'extérieur de l'intervalle de confiance implique qu'une sortie qui correspond à l'entrée ou se trouve ce pic n'est pas décrite par le modèle.



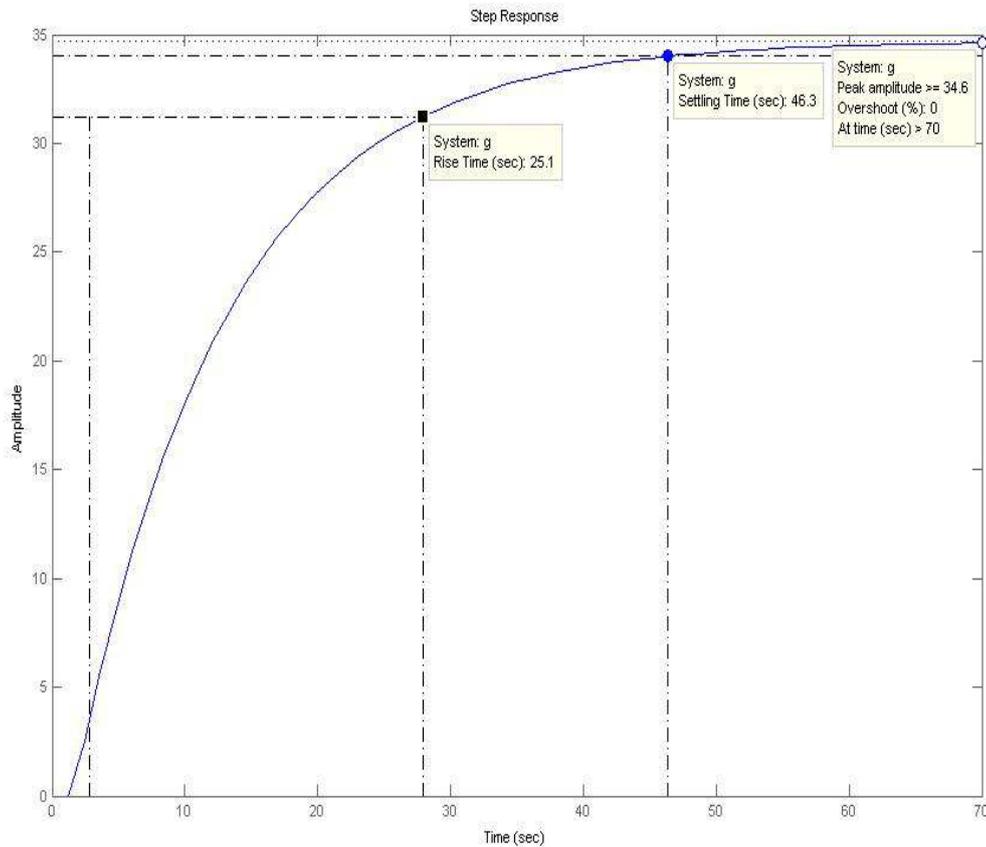
**Figure III.8 l'analyse résiduelle**

D'après la figure III.8 on remarque que l'autocorrelation de résidu se trouve dans l'intervalle de confiance, aussi pour l'intercorrélation entre l'entrée et le résidu.

### III.3.2 La réponse indicielle du modèle

Les caractéristique de la réponse indicielle du modèle est très proche de notre processus

Ceci est illustré par la figure III.9



**Figure III .9 La réponse indicielle du modèle**

### **III.4. Conclusion**

Un bon choix du model avec un nombre d'échantillons assez suffisants conduit a une bonne estimation des paramètres.

## **Chapitre IV**

### **La commande numérique**

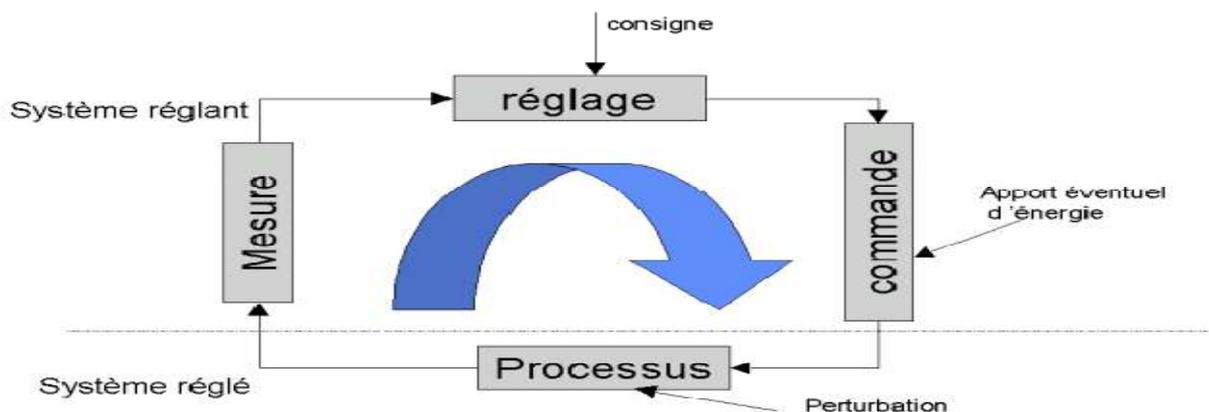
#### **IV.1 Introduction**

La régulation est très importante dans notre vie quotidienne, elle apparait dans la plupart des actions, par exemple, conduire une voiture, régler un four de cuisine...

La régulation ou bien l'asservissement consiste à faire une action sur le système pour que la mesure soit égale la consigne.

Si l'on cherche à atteindre une consigne (de position ou de température), on parlera de poursuite ou asservissement ; si l'on cherche à éliminer des perturbations pour qu'une valeur reste constante (ex : garder la température intérieure de la voiture constante quelle que soit la température extérieure), on parlera de régulation. L'industrie utilise à foison des systèmes d'asservissement ou de régulation : que ce soit pour gérer le débit d'un fluide dans une conduite, la température d'un produit, la hauteur d'un niveau de cuve...

Historiquement, les régulateurs n'étaient pas intégrés dans l'unité centrale des Automates Programmables Industriels, mais se présentaient sous forme de modules autonomes gérant leur environnement propre (acquisition, calcul, commande...). Les régulateurs permettent ainsi de lier plus simplement les parties séquentielle et continue du procédé. La régulation fait partie intégrante de la qualité de production : c'est donc un point non négligeable de la chaîne de valeurs d'une installation.



**Figure. IV.1 La régulation en général**

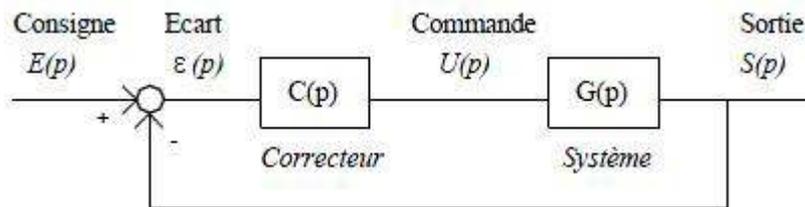
Le contrôle de procédé met en œuvre une chaîne d'outils à plusieurs niveaux. Schématiquement, le lien entre le capteur et l'actionneur peut se représenter comme ci-dessus. Le but étant de maîtriser le procédé et, soit de le piloter, soit de l'empêcher d'évoluer en raison de perturbations.

## **IV.2 La régulation**

Les systèmes asservis peuvent présenter des défauts, une précision insuffisante, une stabilité trop relative (voire une instabilité), un temps de réaction trop lent, un dépassement trop important. Il est donc souvent nécessaire d'intégrer dans le système asservis un réseau

correcteur dont l'objectif est d'améliorer un ou plusieurs de ces différents paramètres sans bien sûr le faire au détriment des autres.

Alors après avoir identifié le modèle de comportement du procédé, il faut boucler le système à l'aide d'un correcteur qu'il faut régler.



**Figure IV.2 régulation du système en boucle fermée.**

Les paramètres du système à régler :

- la vitesse à laquelle la valeur finale sera atteinte. C'est le temps de réponse.
- la stabilité. Un système est stable si la sortie tend vers une valeur finie. Si elle oscille, l'asservissement est instable.
- le dépassement. Souvent exprimé en pourcent. Même lorsqu'un système est stable, il arrive que la sortie dépasse la consigne avant de se stabiliser.
- la précision, c'est à dire la capacité de l'asservissement à atteindre la consigne. On l'appelle l'écart de positionnement à l'infinie.

### **IV.2.1 Le correcteur PID**

Les correcteurs, usuellement utilisés en pratique sont les régulateurs à effet proportionnel, intégral et dérivé « P.I.D ». Les correcteurs P.I.D. permettent d'engendrer à partir de la sortie du comparateur (c'est-à-dire l'écart existant entre la consigne et la grandeur à réguler) un signal proportionnel à l'erreur et à sa dérivée d'une part, et à son intégrale d'autre part.

Sa description temporelle est la suivante :

$$u(t) = K_p(e(t) + \frac{1}{T_i} \int_0^t e(t) + T_d \frac{de(t)}{dt})$$

Avec l'écart défini comme suit :

$$e(t) = uc - y(t)$$

#### IV.2.1.1 Description des différentes actions du PID

- **L'action Proportionnelle**

Cette action est la plus classique. Elle consiste à appliquer une correction en rapport avec la différence instantanée entre la mesure et la consigne courante, le rapport Erreur/Commande s'appelle le gain.

Dans le cas d'un correcteur proportionnel, la loi de commande corrigée  $u(t)$  est proportionnelle à l'écart  $e(t)$  :

$$u(t) = K_p \cdot e(t)$$

La limite de cette commande arrive à partir du moment où l'erreur devient petite ou si la consigne n'est pas constante ; en effet l'annulation de l'écart considère que la consigne à l'instant  $t+1$  sera le même qu'à l'instant  $t$ .

L'effet d'une augmentation du gain entraîne une diminution de l'erreur statique, rend le système plus rapide mais augmente l'instabilité du système.

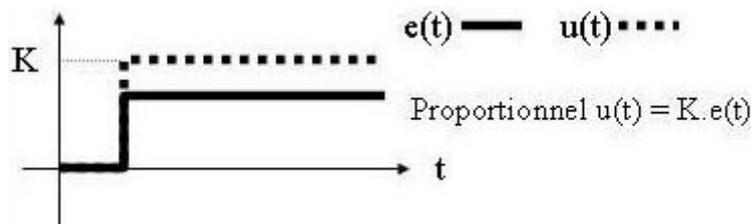


Figure IV.3 action proportionnelle

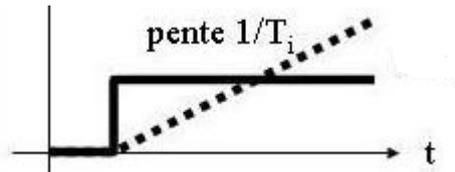
- **L'action intégrale**

L'action intégrale complète l'action proportionnelle. Elle permet d'éliminer l'erreur résiduelle en régime permanent. Afin de rendre le système plus dynamique (diminuer le temps de réponse), on diminue l'action intégrale mais, ceci provoque l'augmentation du déphasage ce qui provoque l'instabilité en état fermé.

L'action intégrale est utilisée lorsqu'on désire avoir, en régime permanent, une précision parfaite, en outre, elle permet de filtrer la variable à régler d'où l'utilité pour le réglage des variables bruitées.

Pour un intégrateur pur la loi de commande  $u(t)$  est de la forme :

$$u(t) = \frac{1}{T_i} \int_0^t e(t) dt$$



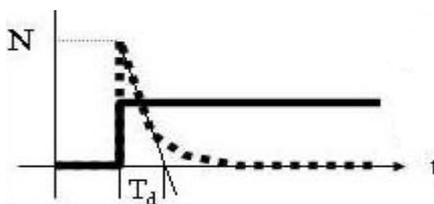
**Figure IV.4 action intégrale**

- **L'action dérivée**

L'action Dérivée, en compensant les inerties dues au temps mort, accélère la réponse du système et améliore la stabilité de la boucle, en permettant notamment un amortissement rapide des oscillations dues à l'apparition d'une perturbation ou à une variation subite de la consigne.

L'intérêt principal de ce type de correcteur est donc d'améliorer la précision, il introduit malheureusement un déphasage de  $-90^\circ$  et risque de rendre le système instable (diminution de la marge de phase).

$$u(t) = T_d \cdot \frac{de(t)}{dt}$$



**Figure IV.5 action dérivée**

L'intérêt du correcteur PID est d'intégrer les effets positifs des trois correcteurs précédents. La détermination des coefficients  $K_p$ ,  $T_i$ ,  $T_d$  du correcteur PID permet d'améliorer à la fois la précision ( $T_d$  et  $K_p$ ) la stabilité ( $T_d$ ) et la rapidité ( $T_d$ ,  $K_p$ ).

#### **IV.2.1.2 Synthèse de contrôleurs PID**

Le réglage d'un PID est en général assez complexe, des méthodes pratiques de réglage permettent d'obtenir des bons résultats.

### 1. méthode de Broïda

Si l'on procède à des tests, l'un des plus simples à appliquer sur un procédé stable est la réponse indicielle. On provoque un échelon sur la commande du procédé, préalablement mis en un état stable, et on analyse l'évolution de la mesure.

En général, le système se comporte comme une fonction de transfert de Premier Ordre à retard pure.

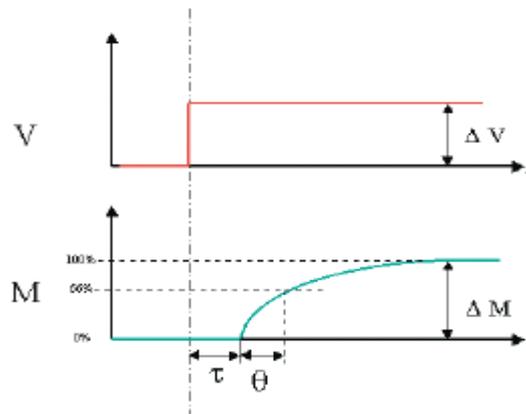


Figure IV.6 réponse indicielle du système 1 ordre avec retard

$\theta/\tau$	$10 < \theta/\tau < 20$	$5 < \theta/\tau < 10$	$2 < \theta/\tau < 5$
Type de régulation	P	PI	PID
$k_p$	$\frac{0.8 \cdot \theta}{GS \cdot \tau}$	$\frac{0.8 \cdot \theta}{GS \cdot \tau}$	$\frac{\theta}{\tau} + 0.4$ $\frac{1.2 \cdot GS}{GS}$
$T_i$	-	$\frac{GS \cdot \tau}{0.8}$	$\theta + 0.4 \cdot \tau$
$T_d$	-	-	$\frac{\theta/\tau}{2.5 + \tau}$

TAB IV.1 Paramètres PID obtenus à partir de Broïda.

### 2. Méthode de Ziegler et Nichols

#### 2.1 Méthode de la réponse indicielle

Pour obtenir les paramètres du régulateur PID, il suffit d'enregistrer la réponse indicielle du processus seul (c'est-à-dire sans le régulateur), puis de tracer la tangente au point d'inflexion

de la courbe. On mesure ensuite sa pente  $P$ , le retard apparent  $L$  correspondant au point d'intersection de la tangente avec l'abscisse et le gain  $K$ . On peut alors calculer les coefficients du régulateur choisi à l'aide du tableau IV.1.

type	$K_p$	$T_i$	$T_d$
P	$1/P.L.K$		
PI	$0.9/P.L.K$	$3.L$	
PID	$1.2/P.L.K$	$2.L$	$0.5.L$

**TAB IV.2 Paramètres PID obtenus à partir d'une réponse indicielle « Ziegler et Nichols ».**

## 2.2 Méthode du point critique

Méthode du point critique est basée sur l'analyse de la stabilité du système à contrôler. Elle consiste à analyser le système en présence d'un contrôleur proportionnel pur, et à déterminer le gain de ce contrôleur qui met le système à la limite de stabilité, en assurant une oscillation entretenue. On note le gain de ce contrôleur « gain ultime  $K_p$  » et la période de l'oscillation (fréquence critique). Remarquons le parallèle avec l'analyse de Bode en boucle ouverte : la fréquence critique correspond à un déphasage de  $-180^\circ$ , et la stabilité à cette fréquence exige que le gain de la boucle ouverte soit unitaire. La présence d'un contrôleur purement proportionnel ne change pas le déphasage, mais son gain vient multiplier celui des autres sous systèmes de la boucle.

La méthode de Ziegler-Nichols consiste donc à :

1. déterminer le diagramme de Bode du système en boucle ouverte. On trace les graphes « Rapport d'amplitude » et « Déphasage » en fonction de la fréquence (Diagramme de Bode). Dans le système analysé, le contrôleur implante l'algorithme proportionnel, avec un gain unitaire  $K_c=1$ .
2. déterminer la fréquence critique  $\omega_c$  ainsi que le rapport d'amplitude à cette fréquence  $|G(\omega_c)|$ .

Calculer le gain du contrôleur proportionnel qui mettrait le système avec rétroaction à la limite de l'instabilité. Comme cet état est caractérisé par une valeur unitaire du rapport d'amplitude  $|G(\omega_c)|$  et que la valeur de  $K_c$  n'a pas d'effet sur la fréquence critique, le gain ultime à la fréquence critique est donc égal à l'inverse du rapport d'amplitude déterminé au point « 2 », donc  $K_u = 1 / |G(\omega_c)|$ .

3. Ce gain qui amène le système à la marge de stabilité est appelé gain ultime, et la période d'oscillation à cette limite de stabilité est appelée période ultime  $P_u$ . Elle s'exprime en unités de temps :  $P_u = 2\pi / \omega_c$ .

4. On calcule les paramètres de réglage du contrôleur à partir des corrélations de Ziegler-Nichols rappelés dans le tableau suivant.

Type de contrôleur	Kc	Ti	Td
P	$K_u / 2$	-	-
PI	$K_u / 2.2$	$P_u / 1.2$	-
PID	$K_u / 1.7$	$P_u / 2$	$P_u / 8$

**TAB IV.3 Paramètres PID obtenus à partir de Ziegler-Nichols.**

#### IV.2.1.3 la correction PID du modèle

On utilise les résultats de l'identification :

La fonction de transfert du système s'écrit sous la forme d'une fonction du premier ordre avec retard :

$$G(s) = \frac{K}{1 + T_{p1} s} e^{-Tr.s}$$

Avec :

$$K = 34.728$$

$$T_{p1} = 11.594$$

$$Tr = 1.5962$$

Les méthodes de synthèse de PID peuvent nous aider pour avoir un intervalle de différents éléments du PID «  $K_p, T_i, T_d$  ». On commence donc par installer un contrôleur P de gain obtenu par la méthode de **Broïda**, en prenant plusieurs consignes.

Remarquons que ce gain est acceptable pour plusieurs consignes ou pour une perturbation extérieure. Puis on ajoute l'action intégrale pour que le système soit plus précis, et on varie les paramètres «  $K_p$  et  $T_i$  » pour avoir des résultats qu'on veut atteindre.

- Alors avant de passer en pratique, on simule le système avec le PID en boucle fermée.
- Puisque notre système a un retard on utilise le bloc «Transport Delay» en fonction de transfert continue et «Unit Delay» dans le discret.
- On applique une saturation «Saturation» entre un et zéro sur la commande puisque notre commande réel est en tout ou rien.

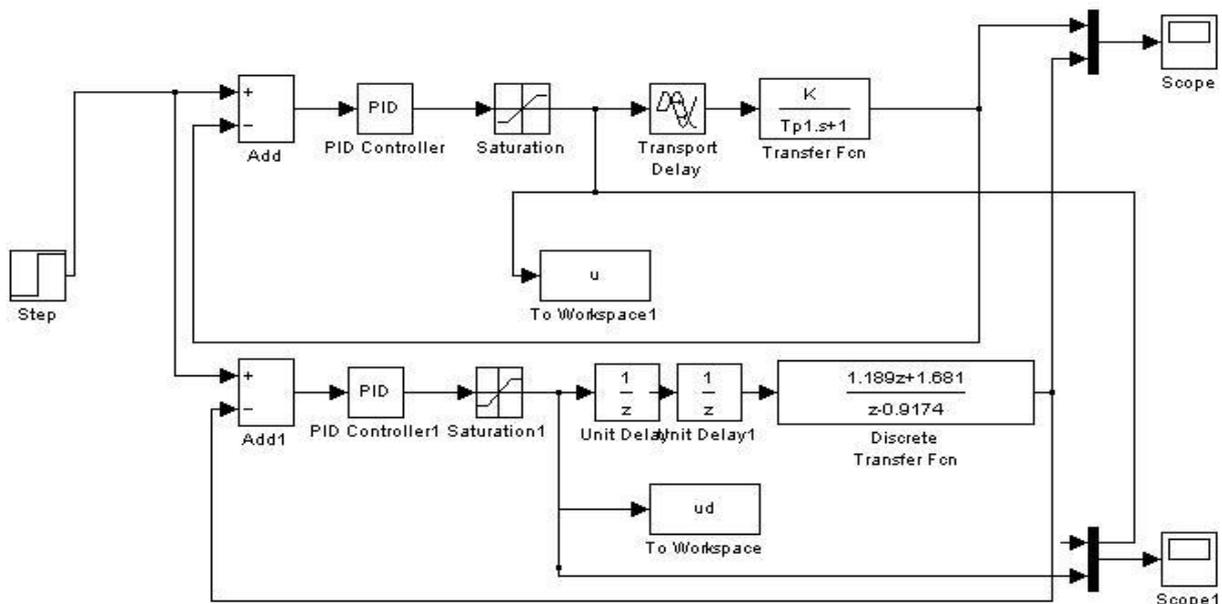
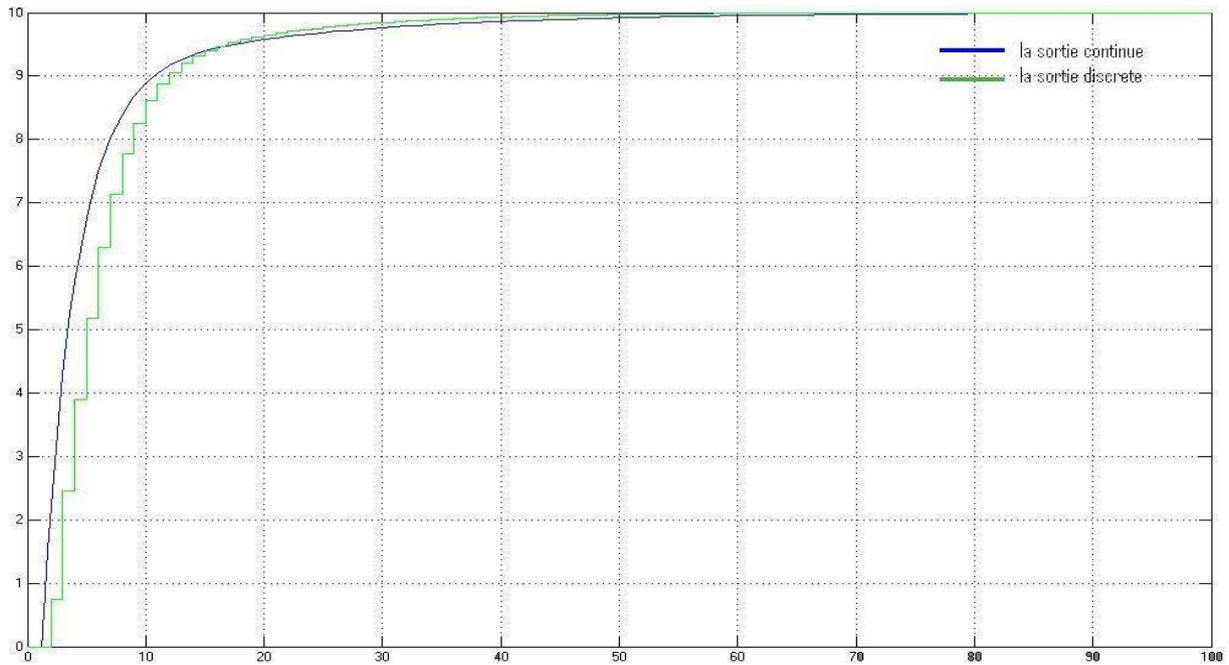
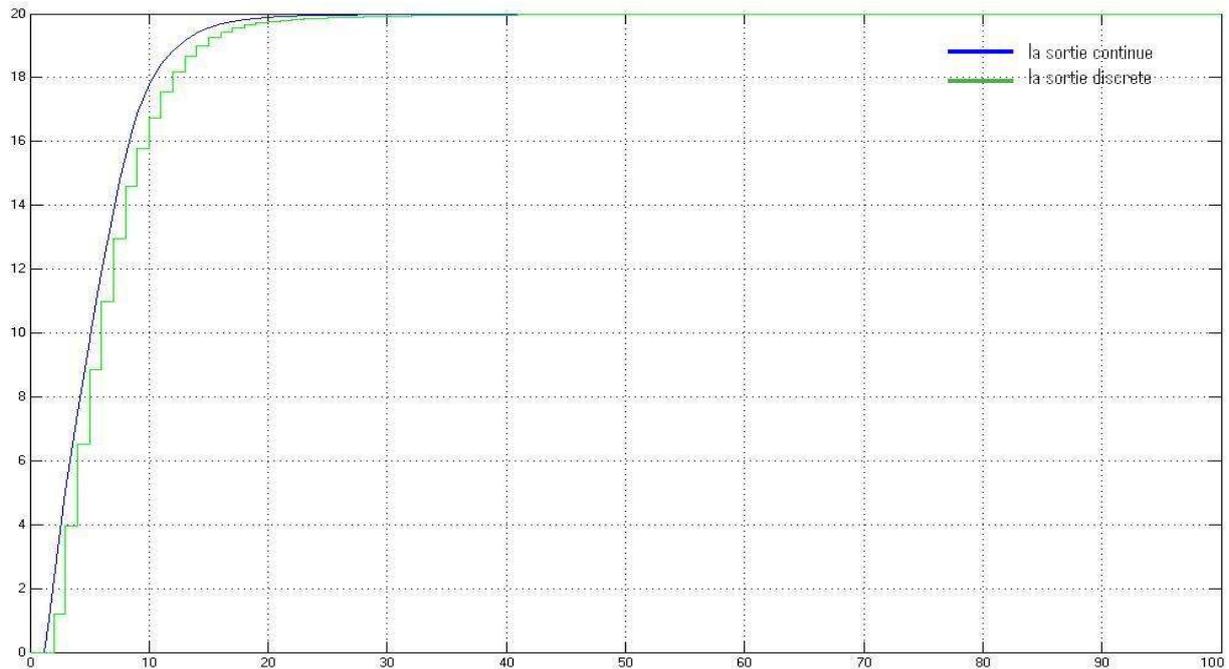


Figure IV.7 le schéma bloc de la régulation du modèle



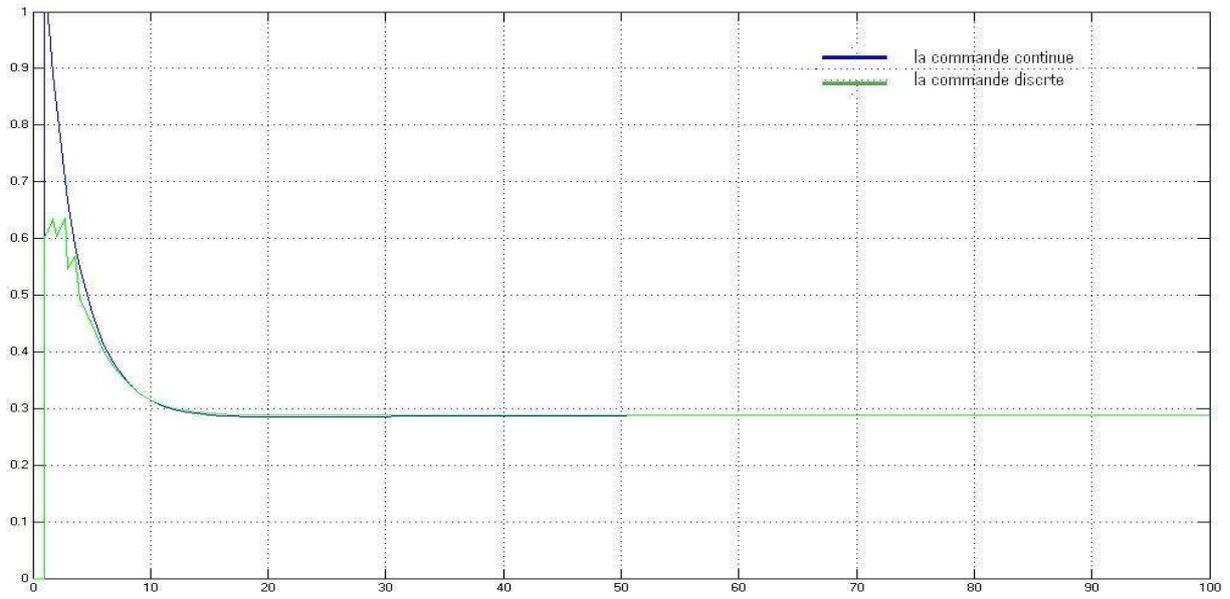
**La figure IV.8 signal de sortie, la consigne  $U_c=10$ . Continue :  $K_p=0.1$  et  $K_i=0.006$ . Discret :  $K_p=0.06$  et  $T_i=0.0048$**



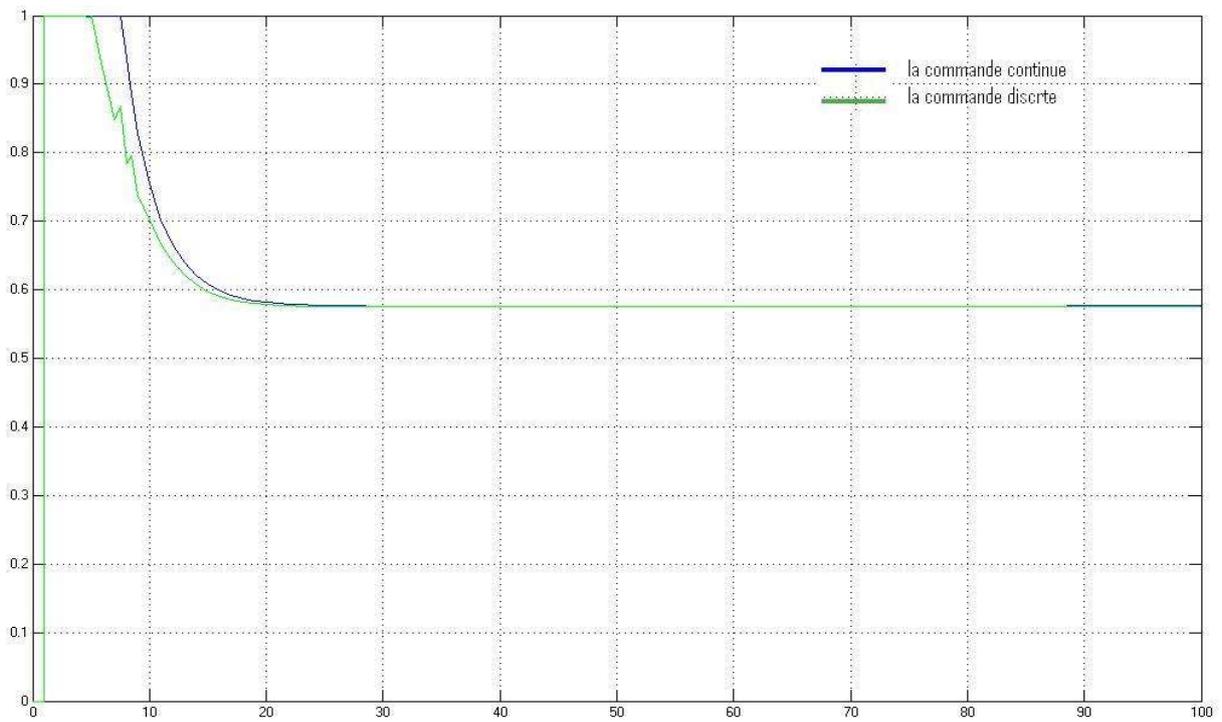
**La figure IV.9 signal de sortie à la consigne  $U_c=20$ .**

**Continue :  $K_p=0.1$  et  $K_i=0.06$ . Discret :  $K_p=0.06$  et  $K_i=0.0048$**

Les deux figures précédentes « figure IV.8 et figure IV.9 » montrent que le PI appliqué au système est convenable puisque la sortie est stable et converge vers la consigne désirée.



**La figure IV.10 signal de commande à la consigne  $U_c=10$ .**



**La figure IV.11 signal de commande, consigne  $U_c=20$ .**

Remarque :  $K_i = 1/T_i$ .

#### IV.2.1.4 Définition de PWM

Le PWM « Pulse Width Modulation » ou bien Modulation de largeurs d'impulsions « MLI », consiste à alterner rapidement entre deux états distincts du système afin d'obtenir en moyenne un signal analogique.

##### IV.2.1.4.1 Principe de PWM

Le signal PWM peut être obtenu en comparant un signal avec un autre signal triangulaire. En effet, si le signal comparé est au-dessus du signal triangulaire, la sortie est à l'état bas. Dans le cas contraire, la sortie est mise à l'état haut.

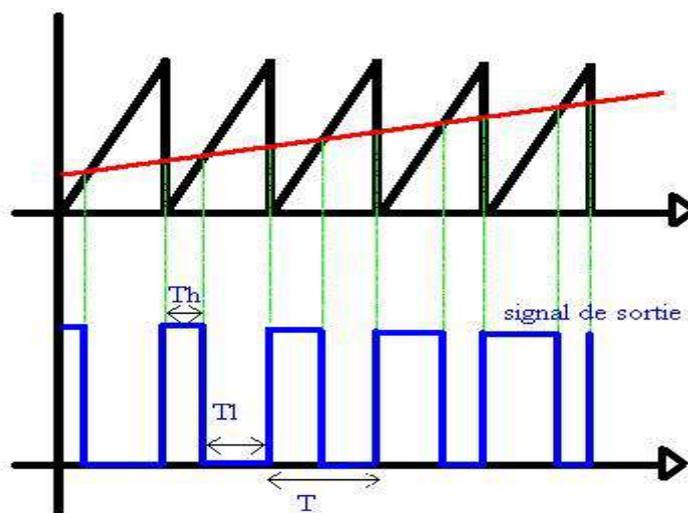
Le signal continu généré à partir des deux valeurs distinctes prend la valeur moyenne sur les intervalles de temps.

On précise que dans la figure IV.12 :

- $T_h$  représente la période de temps pendant laquelle le système est à l'état haut.
- $T_l$  représente la période de temps pendant laquelle le système est à l'état bas.

Le PWM se caractérise par :

- la période PWM :  $T_{pwm} = T_h + T_l$
- le rapport cyclique d'impulsion  $\alpha = \frac{T_h}{T_{pwm}}$



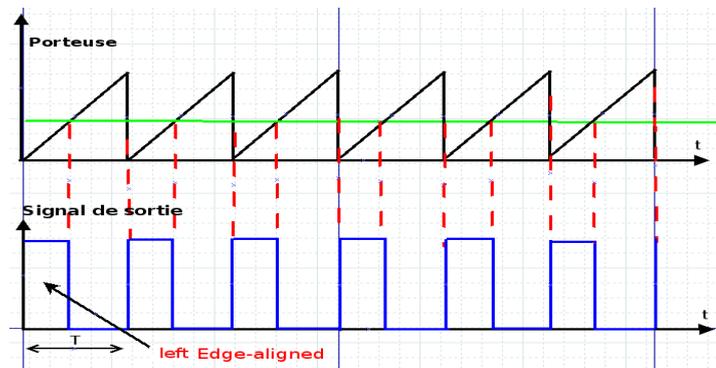
La figure.12 Principe de PWM.

La fréquence PWM est toujours constante par contre le rapport cyclique peut varier. De plus, le signal PWM est un signal rectangulaire qui varie entre deux valeurs extrêmes mais les systèmes sur lesquels on applique du PWM sont en général des filtres passe bas (moteurs, œil humain, . . .), ils ne ressentent que la valeur moyenne de la PWM.

#### IV.2.1.4.2 les types de PWM

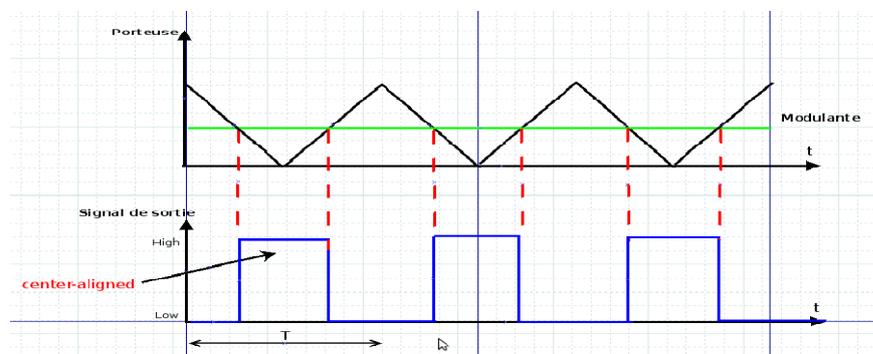
Il existe deux modes de PWM :

- edge-aligned : Le signal est asymétrique par rapport au début de la période. Ce signal s'obtient en comparant un signal constant avec un signal triangulaire possédant une seule pente. Nous trouvons le mode left edge-aligned (à chaque début de période, on est à l'état haut) et le mode right edge-aligned (à chaque fin de période, on est à l'état haut).



La figure.13 signal PWM asymétrique.

- center-aligned : Le signal est une fois à l'état haut, et à la période suivante à l'état bas. Ce signal est donc symétrique par rapport au début d'une période. Ce signal s'obtient en comparant un signal constant avec un signal triangulaire à deux pentes.

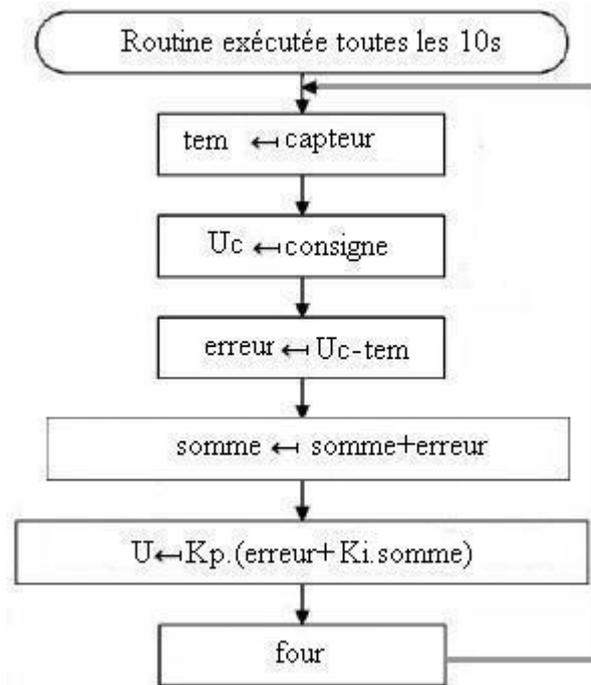


La figure.14 signal PWM symétrique.

### IV.2.1.5 le correcteur PID en pratique

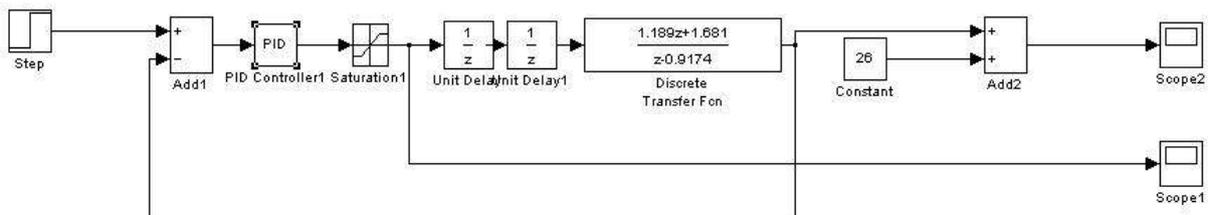
Dans cette partie pratique on applique les résultats qu'on a trouvé en simulation. Le programme écrit sous MATLAB permet d'accueillir les mesures réelles de la température, calculer la commande et l'envoyer sous forme PWM.

L'organigramme de la figure IV.15 explique les différentes étapes traitées dans le programme.

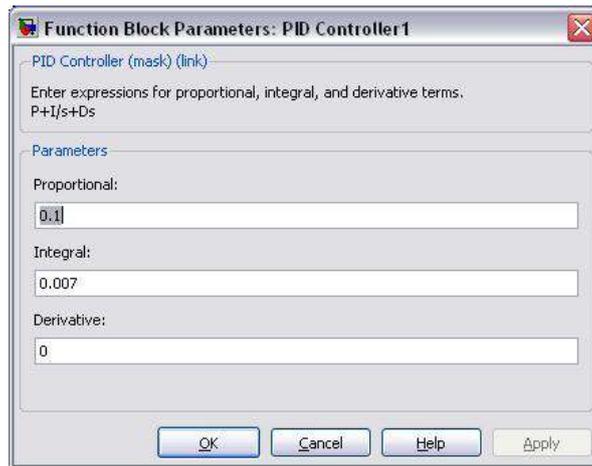


La figure IV.15 l'organigramme du programme.

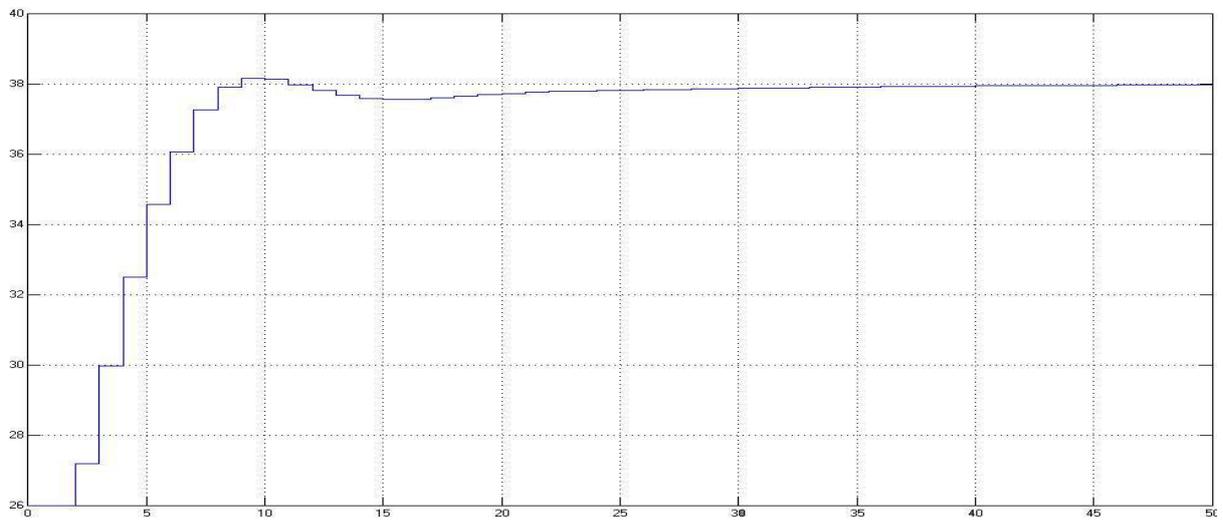
- Si on applique les résultats du PI de la partie simulation «  $K_p=0.06$  et  $K_i=0.0048$  » le système converge vers la température désirée très lentement.
- Alors on a varié les paramètres du PI pour rendre le système plus rapide mais avec un dépassement acceptable.  $K_p=0.1$  et  $K_i=0.07$ .



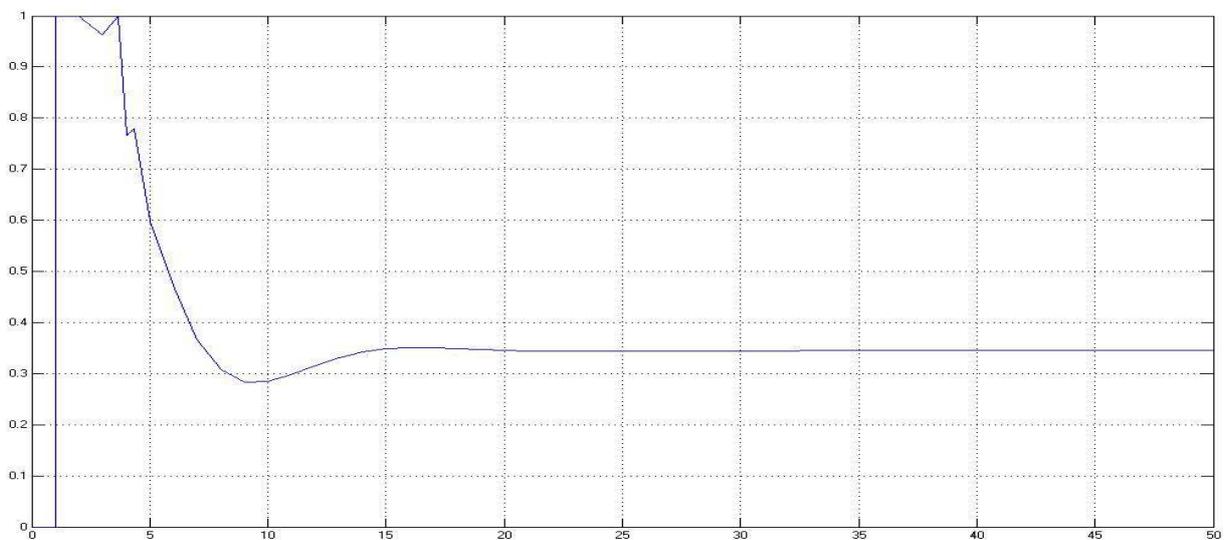
La figure IV.16 le schéma bloc de la régulation du modèle discret.



**La figure IV.17 Menu des paramètres PID du MATLAB**

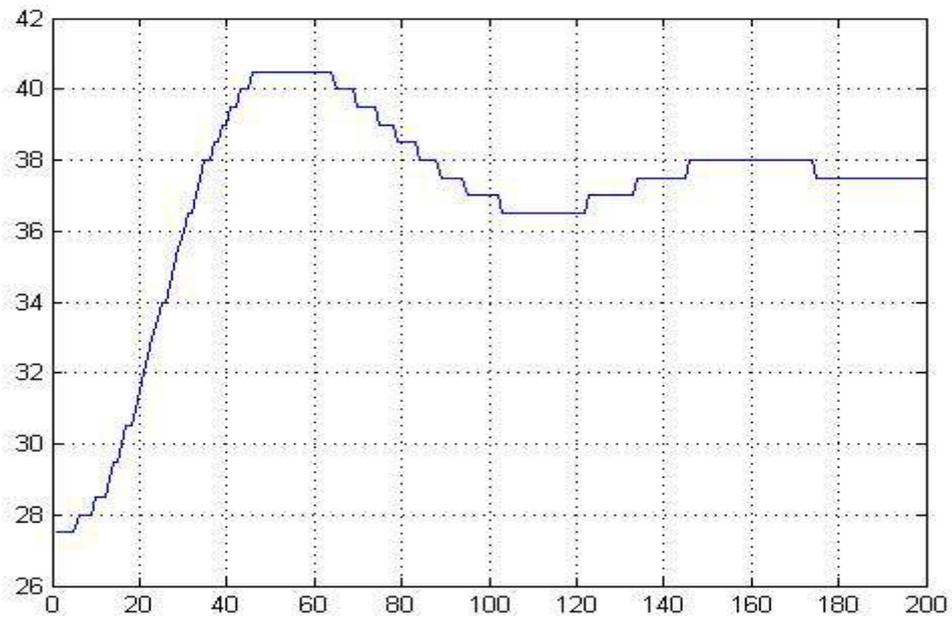


**La figure IV.18 signal de sortie,  $U_c=38$ .**

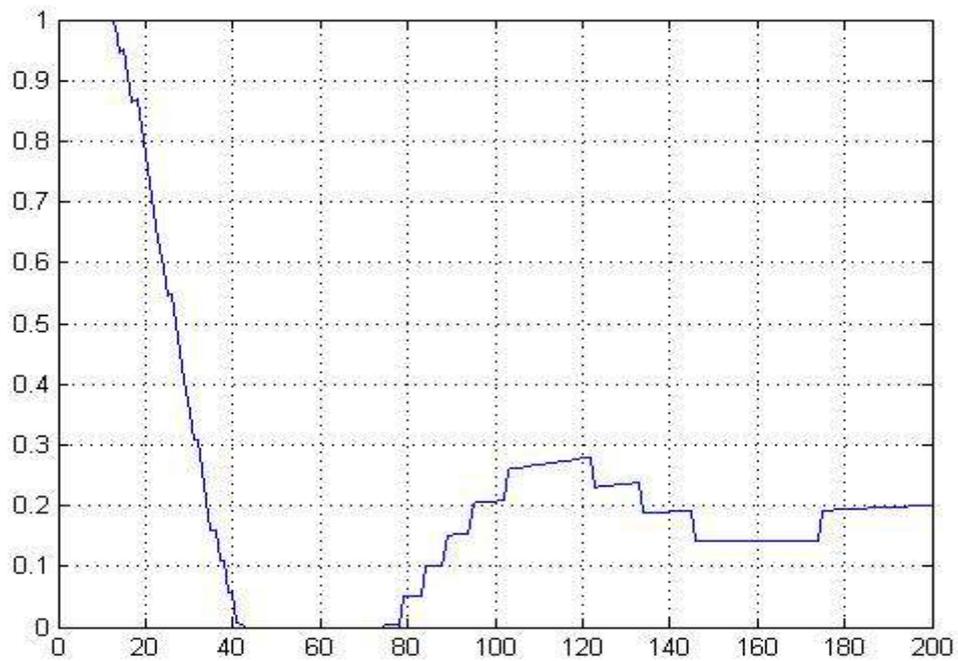


**La figure IV.19 signal de commande,  $U_c=38$ .**

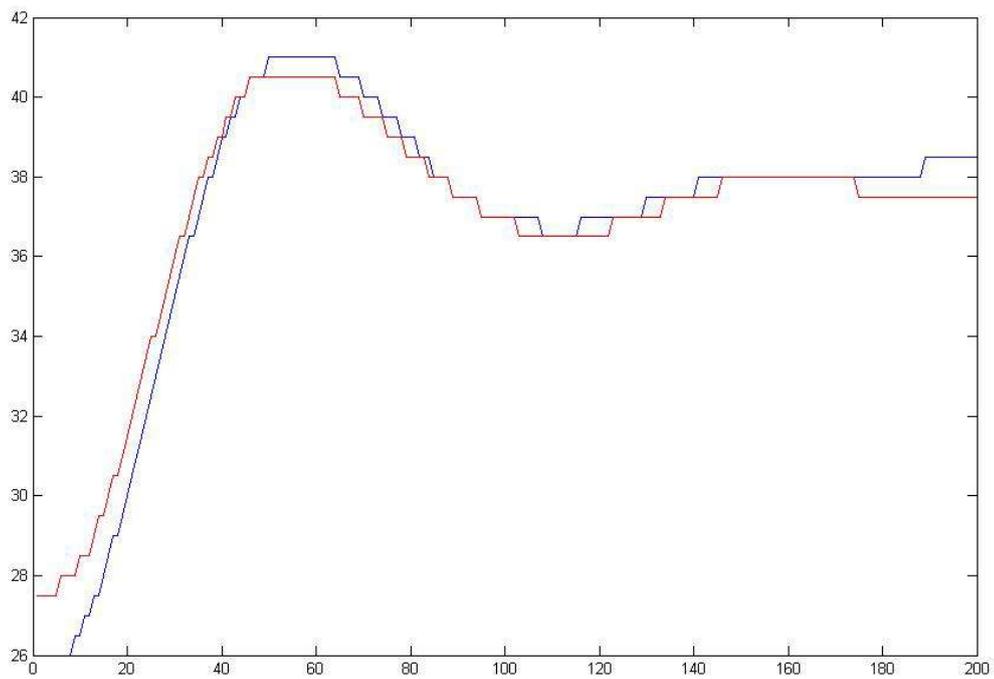
- Si on utilise ces paramètres du PID sur le système réel « four » on obtient ces résultats :



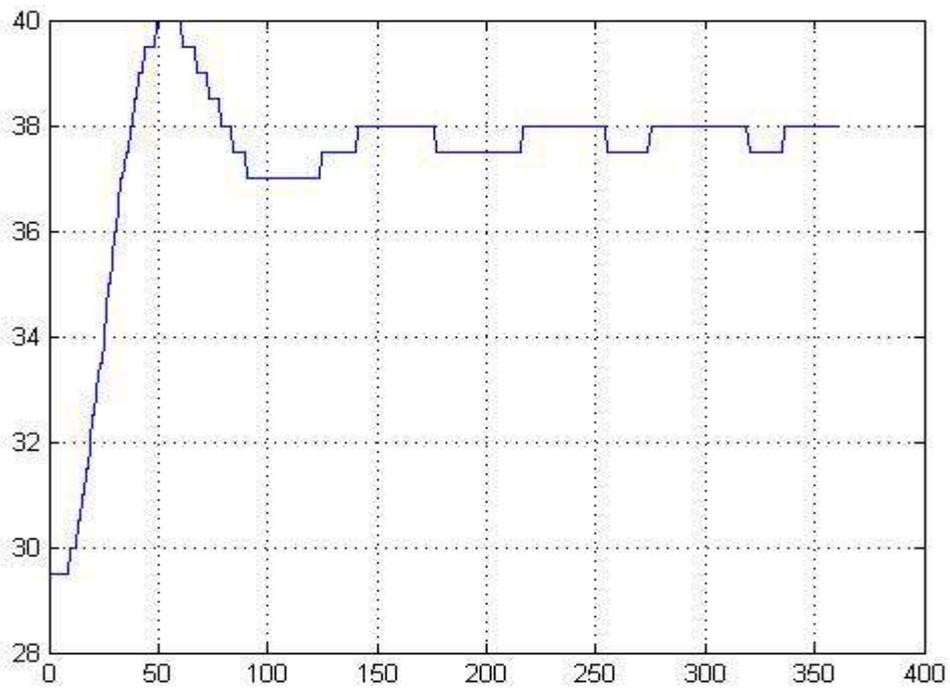
**La figure IV.20 le signal de sortie réelle,  $U_c=38$   $T_e=10s$**



**La figure IV.21 le signal de commande réelle,  $U_c=38$   $T_e=10s$**



**La figure IV.22 le signal de sortie réelle,  $U_c=38$   $T_e=10s$  température ambiante  $U_e=26$  et  $U_e=27.5$**



**La figure IV.23 le signal de sortie réelle,  $U_c=38$   $T_e=10s$  après 1 heure.**

À partir de ces résultats obtenus en pratique, on déduit que :

1. Les paramètres choisis du PID sont convenables au système réel.
2. Les résultats obtenus dans la partie de simulation et la partie pratique sont proches, alors la fonction de transfert obtenu de l'identification décrit notre système d'une manière convenable.
3. Le changement de la température externe n'influe pas sur la régulation.

La température du système est égale à la consigne et reste constante après 25min.

### **IV.3 Commande adaptative**

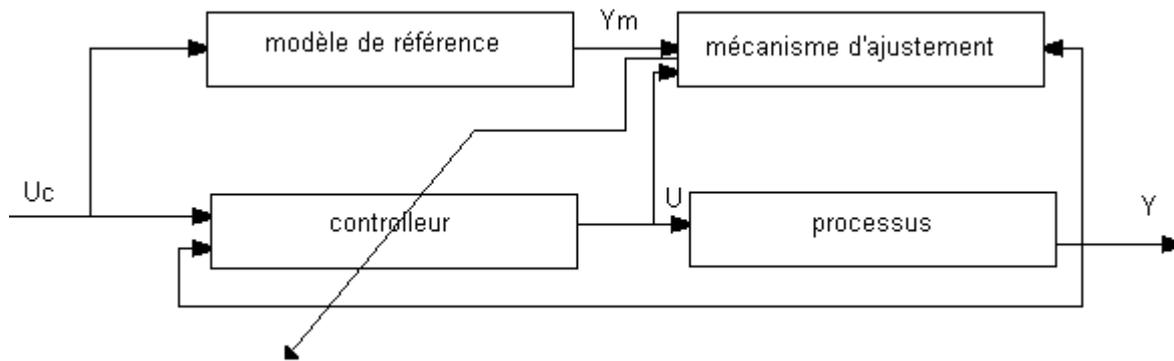
L'une des techniques de commande qui a connu une grande notoriété, est la commande adaptative, qui comme son nom l'indique, consiste à adapter le régulateur en ligne aux variations du procédé régulé pour assurer une qualité constante de performances. Les processus commandés sont totalement inconnus (systèmes boîtes noires) ou partiellement inconnus, ou des systèmes dont les paramètres varient dans le temps.

Un travail de recherche considérable sur la commande adaptative a été effectué depuis le début des années 1950. La première motivation fut la commande des avions de hautes performances qui opéraient sur une large plage de vitesses et d'altitudes. Cependant, seules des méthodes basées sur le domaine fréquentiel étaient disponibles alors, et ils ne suffisaient pas à résoudre tous les problèmes de commande adaptative.

#### **IV.3.1 Commande adaptatif à modèle de référence**

un système adaptatif à modèle de référence est conçu de manière à ce qu' la réponse à un signal de commande soit confondu avec celle d'un modèle de référence qui donnerait la réponse désirée au même signal de commande un schéma bloc d'un tel système est illustré par la figure IV.24 .

Le système est composé d'une boucle de retour ordinaire comprenant le processus et le contrôleur ainsi qu'une deuxième boucle de retour dont le rôle est de varier les paramètres du contrôleur en se basant sur l'erreur calculé à partir de la différence entre le signal de sortie du processus et celui généré par le modèle de référence.



La figure IV.24 Schéma bloc d'un système de contrôle à modèle référence

### IV.3.1.1 La règle du MIT (méthode de gradient)

Le règle de MIT est considéré comme étant a la base de développement du contrôleur adaptatifs à modèle de référence. son nous découle se fait qu'elle a été proposée par le laboratoire d'instrumentations au MIT.

Pour présenter cette approche on considère un système en boucle fermée dans lequel le contrôleur a un seul paramètre ajustable  $\theta$ , la sortie désirée du système est celle délivrée par le modèle de référence «  $y_m$  ». Soit l'erreur entre la sortie «  $y$  » de processus et «  $y_m$  ».

Une possibilité d'ajustement de paramètre  $\theta$  consiste à minimiser la fonction de cout  $J(\theta)$  définie :

$$J(\theta) = \frac{1}{2} e^2 \quad (\text{IV-1})$$

Pour avoir un «  $J(\theta)$  » faible il est donc raisonnable de proposer le faire varier  $\theta$  la direction opposés du gradient de  $J(\theta)$  :

$$\frac{d\theta}{dt} = -\gamma \cdot e \cdot \frac{de}{d\theta} \quad (\text{IV-2})$$

Cette équation exprime le fait que si  $J$  augmente ( $\frac{dJ}{d\theta} > 0$ ) il faut faire diminuer  $\theta$  ( $\frac{d\theta}{dt} < 0$ )

Le but étant de faire en sorte que  $J$  diminue.

### IV.3.1.1.1 L'application de la règle (MIT)

La fonction de transfert :

$$G(s) = \frac{k}{s + k_2} * e^{(-td*s)} \quad (IV-3)$$

Avec:

$$k = k_p/t_p$$

$$k_2 = 1/t_p$$

On choisit comme lois de commande

$$u = (\theta_1 U_C - \theta_2 y) \quad (IV-4)$$

D'ou

$$y = G(s) * u$$

D'où :

$$y = \frac{k}{s + k_2} * e^{(-td*s)} * (\theta_1 U_C - \theta_2 y) \quad (IV-5)$$

$$y(1 + \frac{\theta_2 * k * e^{(-1.6*s)}}{s + k_2}) = (\frac{k}{s + k_2} * e^{(-td*s)} * \theta_1) U_C \quad (IV-6)$$

$$y = (\frac{k}{s + k_2 + \theta_2 * k * e^{(-1.6*s)}} * e^{(-td*s)} * \theta_1) U_C \quad (IV-7)$$

$$y_m = \frac{km}{s + k_{2m}} e^{(-tdm*s)} * u_C \quad (IV-8)$$

Donc on a un bon suivi si :

$$k\theta_1 = km$$

$$k_2 + \theta_2 * k * e^{(-td*s)} = k_{2m}$$

$$e = y - y_m \quad (IV-9)$$

$$\frac{d\theta_1}{dt} = -\gamma * e * \frac{de}{d\theta_1} \quad (IV-10)$$

$$\frac{d\theta_2}{dt} = -\gamma \cdot e \cdot \frac{de}{d\theta_2} \quad (\text{IV-11})$$

$$\frac{de}{d\theta_1} = \frac{-ke^{(-td*s)}uc}{(s + k_2 + \theta_2 * k * e^{(-td*s)})} \quad (\text{IV-12})$$

$$\frac{d\theta_1}{dt} = -\gamma \cdot e \cdot \frac{ke^{(-td*s)}uc}{(s + k_2 + \theta_2 * k * e^{(-td*s)})} \quad (\text{IV-13})$$

$$\frac{de}{d\theta_2} = \frac{(k * e^{(-td*s)})^2 uc}{(s + k_2 + \theta_2 * k * e^{(-td*s)})^2} \quad (\text{IV-14})$$

$$\frac{d\theta_2}{dt} = -\gamma \cdot e \cdot \frac{(k * e^{(-td*s)})^2 uc}{(s + k_2 + \theta_2 * k * e^{(-td*s)})^2} \quad (\text{IV-15})$$

Si on prend :

$$k_{2m} = k_2 + \theta_2 * k * e^{(-td*s)} \quad e^{2t} \quad k\theta_1 = km \quad \text{et } td = tdm$$

on a:

$$\frac{d\theta_1}{dt} = \gamma' \cdot e \cdot \frac{uc}{(s + k_{2m})} \quad (\text{IV-16})$$

16)

et

$$\frac{d\theta_2}{dt} = -\gamma' \cdot e \cdot \frac{y}{(s + k_{2m})} \quad (\text{IV-17})$$

Tel que :

$$\gamma' = \gamma * ke^{(-td*s)}$$

On peut faire la simulation sous MATLAB pour choisir le paramètre ( $\gamma'$ ) qui convient en choisissant les paramètres  $k$  et  $k_2$  proches des valeurs du modèle.

- La simulation de la commande adaptative :

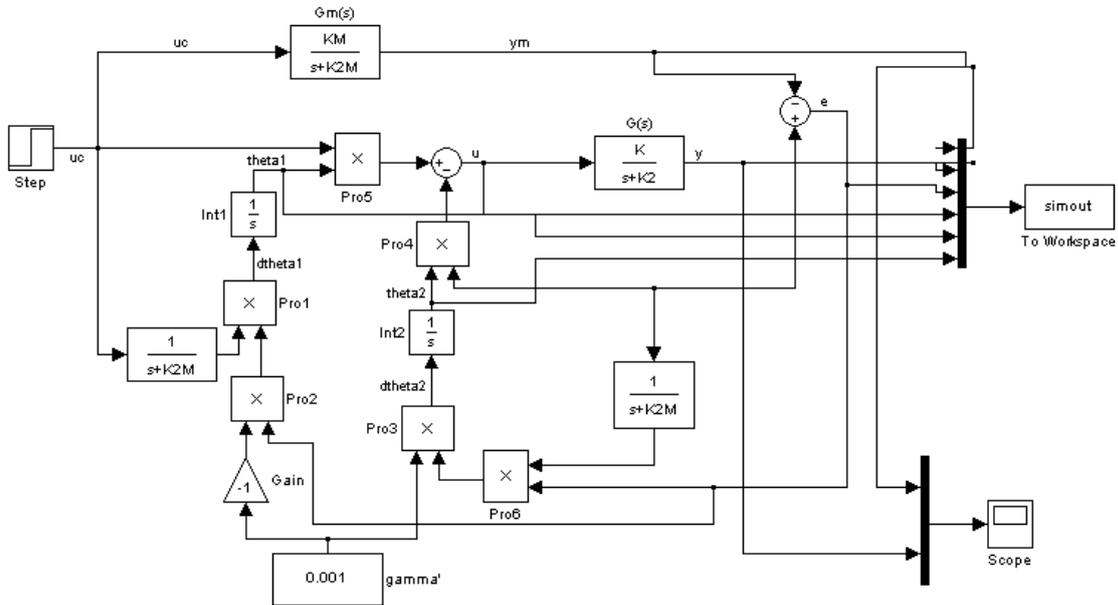


Figure IV.25 Schéma bloc de simulink

Le résultat de simulation montre que le le signal de sortie du système converge ver celui du modèle après un temps ceci est illustré par la figure IV.26

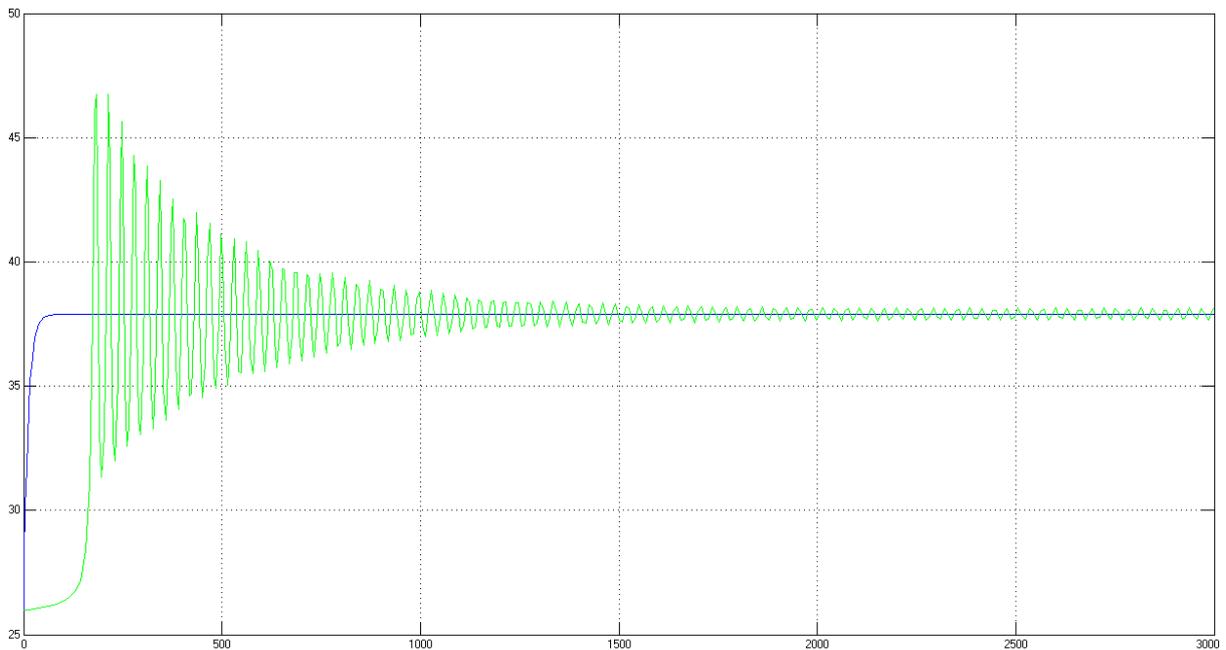
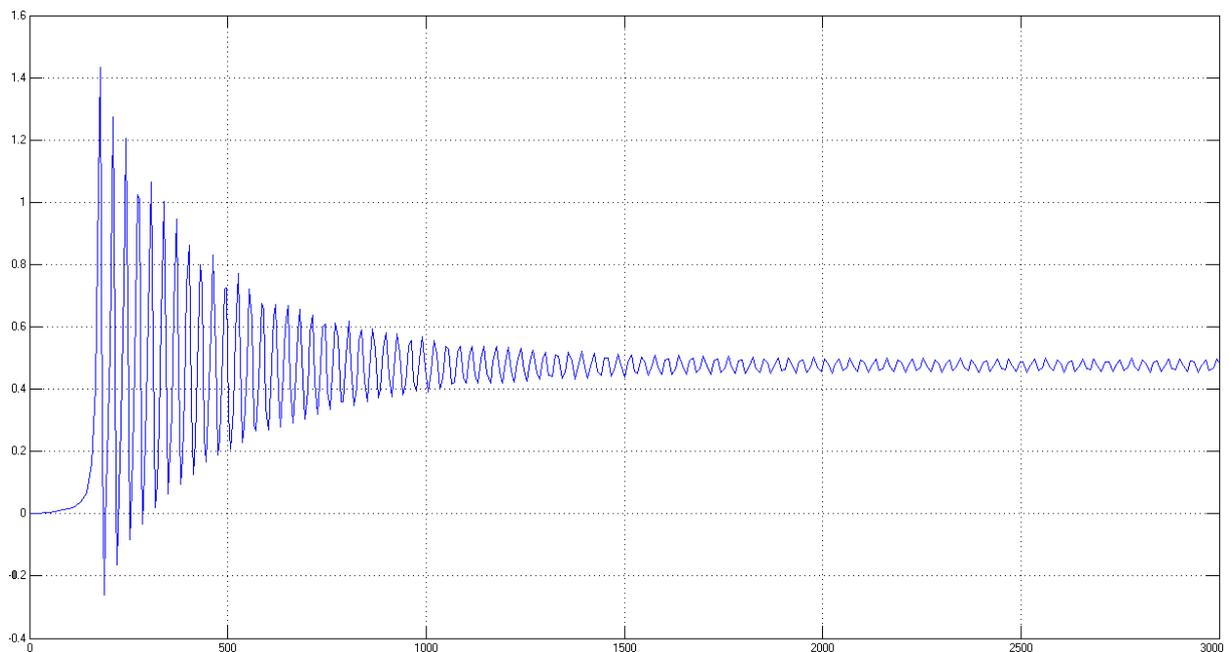


Figure IV.26 la simulation du sortie du système et la sortie du modèle.

La figure IV.26 montre que le système a des oscillations importantes dans les 1000 premières secondes, ensuite elles ont diminué jusqu'à devenir petites après un temps de 1500 secondes.

Ces remarques sont notées aussi sur la commande qui est illustrée par la figure IV.27



**Figure IV.27 la simulation de la commande du système.**

#### **IV.4 Conclusion**

Le correcteur PID classique est très utilisé dans la régulation car il est facile à appliquer, ainsi pour trouver ses paramètres P, I et D., mais ces paramètres ils n'adaptent pas avec le grand changement dans les conditions externes alors que la commande adaptative à un modèle de référence adapte le système à ces variations

## CONCLUSION GENERALE

Dans ce travail, nous avons étudié et réalisé une commande numérique par PC via un microcontrôleur pour contrôler la température d'un four électrique.

En premier lieu nous avons fait des généralités sur le domaine d'incubation des œufs qui était nécessaire pour avoir des informations sur la régulation de la température et extraire la problématique de notre étude.

On a commencé notre étude par réaliser une carte d'interface constituée essentiellement d'un microcontrôleur adapté au port série du PC. Grâce à cette étape on a acquis des notions de base sur la programmation des PIC et les différents types de protocoles de communication RS232 et I2C

Le logiciel de simulation « proteus » permet de simuler la carte avant de la réaliser ce qui conduit à gagner énormément de temps.

On a rencontré des problèmes dans l'identification du système qui a été faite dans le troisième chapitre, ces problèmes sont dus à la température ambiante qu'on n'a pas pu la considérer comme une source de bruits car sa grandeur est proche du signal lui-même c'est-à-dire que le rapport signal sur bruit est très faible. On a contourné ce problème par l'ajout d'un gain constant aux valeurs de signal d'excitation, ce gain représente la température ambiante. Les tests effectués dans ce chapitre ont permis de dire que le modèle est très bon et représente bien le système. Ce modèle nous a aidé pour trouver la loi commande adéquate pour notre objectif.

Les méthodes appliquées dans le chapitre quatre sert à obtenir les valeurs du correcteur PI « Broïda, Ziegler et Nichols » donne un intervalle sur la grandeur de ces paramètres, ces valeurs sont corrigées par la simulation. Les valeurs du PI de simulation sont appliquées sur le système réel et elles ont donné des bons résultats ce qui confirme une autre fois que le modèle mathématique représente bien le système.

D'autre par les résultats d'une correction PI sont satisfaisants, le maintien de la température est atteint après 18 minutes.

Les résultats de simulation la commande adaptative ont montré que le système exige un temps assez long pour éliminer les oscillations et avoir un bon suivi du signal de modèle de référence (environ 10000 secondes)

## Bibliographie

1. Article sur le site : [http://www. Gallinette®](http://www.Gallinette®) Elevage de poules et d'animaux de basse-cour - L'incubation artificielle .fr.
2. Nico van Wageningen, « L'incubation des œufs par les poules et en couveuse »,2004.
3. D.REY « port série », interface PCN°9, ISSN 02434911, juin 2001.
4. B.KAINKA, « petite expérience d'électronique avec PC », EDITION PUBLITRONIQUE ELEKTOR ,2001.
5. P.BORNE, « analyse et régulation des processus industriels », tome I, régulation continue, Edition Technip, Paris ,1993.
6. T.L.Floyd, « Electronique fondamentals ».
7. LOUCIO DI JASIO, «programming 16 bit microcontroller in c », NEWNES.
8. JIVANS.PARAB, «exploring C for microcontrollers », SPRINGER.
9. « PIC Sample Code in C », sur le site suivant: <http://www.microchip.com>.
10. Bigonoff, « La programmation des PIC », Cours sur le site Web : <http://www.Bigonoff.com>.
11. Cours sur le site : <http://www-timc.imag.fr/Norbert.Noury/LPRO-regulation/correction PID.html>.
12. Cours sur le site : <http://www.licn.sciences.univ.metz.fr>.
13. A.Maddi , «Etude de la méthode des moindres carrée récursive et application au signal de parole»,IEEE ,2005 .
14. D. Landau, « Identification des systèmes », Hermes, Paris, 1998.
15. LANDAUID, « Identification et commande des systèmes » ,deuxième édition revue et augmentée, (ISBN 2-86601-365-4), 1993
16. M. Kunt, « traitement numérique des signaux »,Dunod.
17. BOURLES .H « Système linéaire de la modélisation à la commande », Hermès (15mars 2006), Pages : 279. ISBN.
18. LANDAU « commande adaptative : aspect pratique et théorique », 1986.
19. KAZED.B, « support de cours commande adaptative », UNIVERSITE DE BLIDA, 2010 /2011.

## Annexe chapitre II

### Circuit électrique de la carte de commande

Les éléments de la carte :

- PIC 16f877A.
- Quartz 4MHz.
- 2 Capacités 22 $\mu$ F.
- Résistance 1k $\Omega$ .
- DB-9.
- Max232.
- 5 Capacités 1nF.
- 2 résistances 1k $\Omega$  « pull-up ».

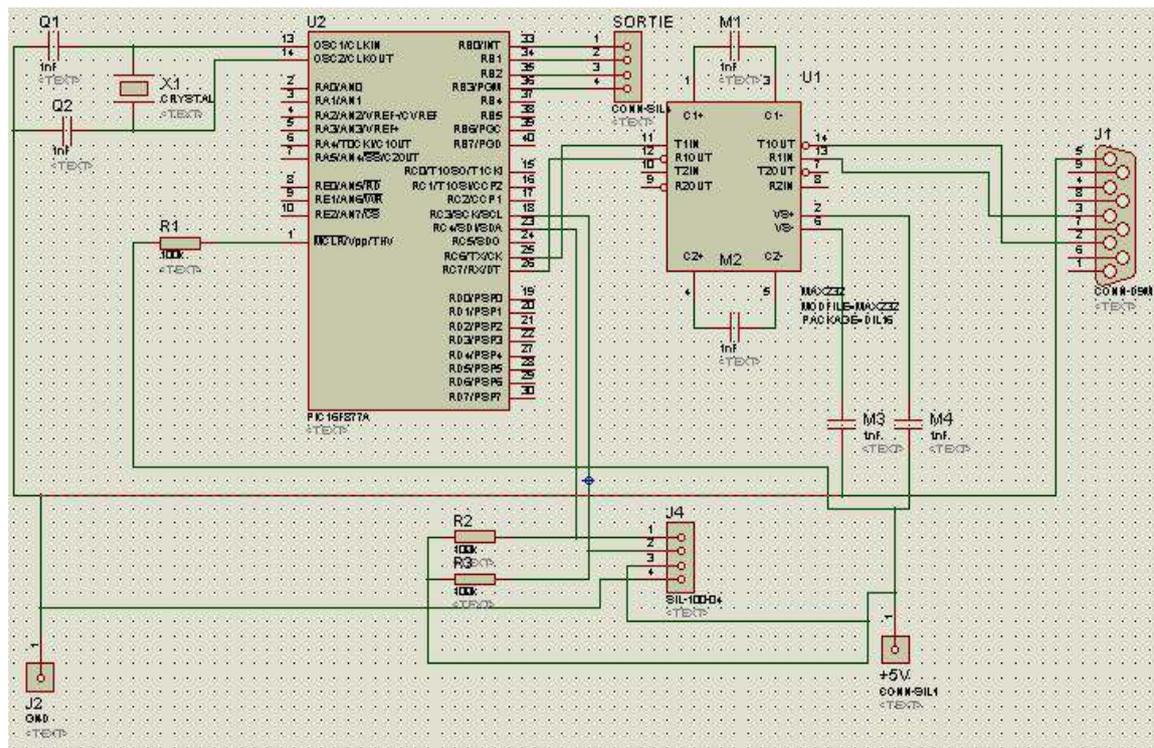


Figure II.9 le schéma électrique de la carte d'interface.

## Circuit électrique de la carte de puissance

Les éléments de la carte :

- Transformateur 220V/110V.
- 4 diodes IN540.
- 1 Condensateur 400V/220 $\mu$ f.
- 1 Relai 5V.

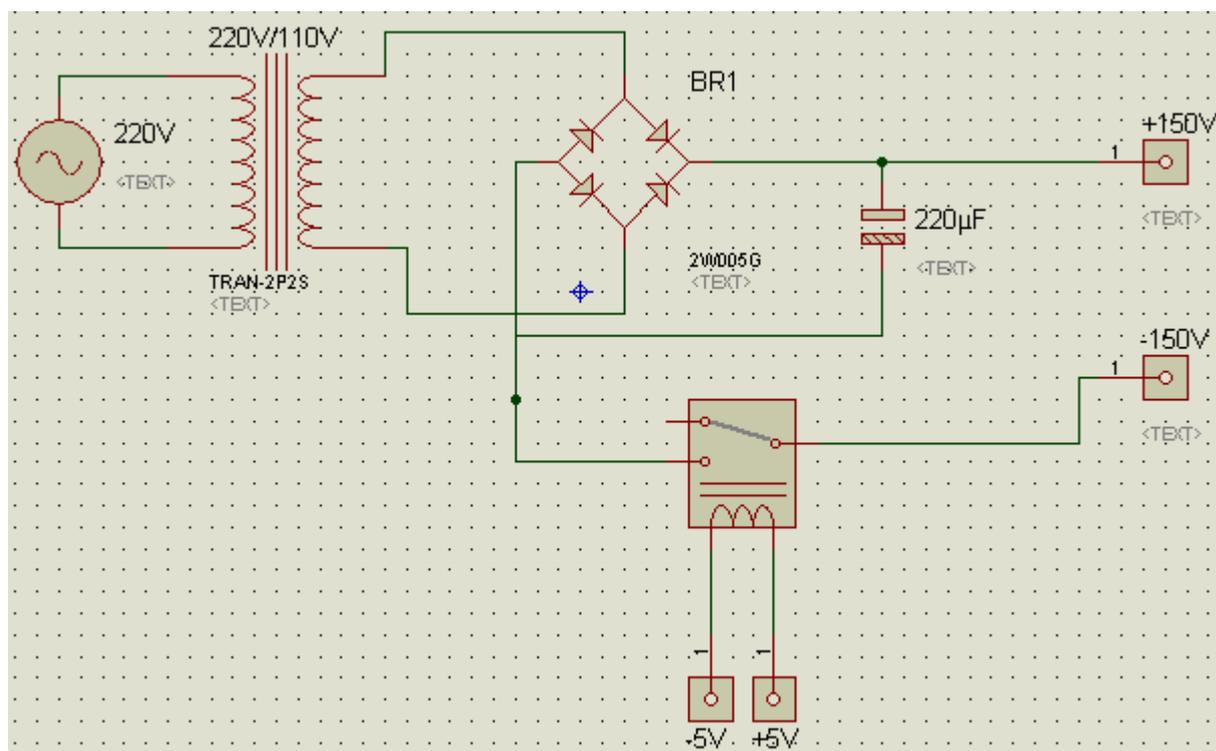


Figure II.10 le schéma électrique de la carte de puissance.