RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique

ÉCOLE NATIONALE POLYTECHNIQUE
DÉPARTEMENT D'AUTOMATIQUE

Laboratoire de Commande des Processus

PROJET DE FIN D'ÉTUDE EN VUE DE L'OBTENTION DU DIPLÔME
D'INGÉNIEUR D'ÉTAT EN AUTOMATIQUE

THÈME :

Commande et supervision de la station CE117 Process Trainer avec l'automate s7 300

Réalisé Par :
BOUALBANI MOHAMED EL SEDDIK

Encadreur :
Pr.El Madjid
BERKOUK.

Juin 2015

Ecole Nationale Polytechnique – Rue des frères Oudek, Hacen Badi, El Harrach, ALGER 16200.

العنوان : التحكم والإشراف على محطة CE 117 وذلك بإستعمال المبرمج S7 300.

ملخص : محطة CE 117 هي المحطة محل الدراسة في هذا العمل. لقد قمنا في بداية الأمر بالتعرف على وظائف النقل المختلفة وذلك بإستخدام طرق مختلفة. ثم قمنا بتحديد إعدادات مختلف المنظمين بإستخدام طريقة زغلر نيكلز.
لقد قمنا بالتحكم عن طريق المبرمج Siemens S7 300 وذلك بوساطة مشروع Step7-Wincc. قمنا بالتحكم في : المستوى، التدفق، الضغط و درجة الحرارة.
الكلمات المفتاحية : CE 117 Process Trainer, Siemens S7-300, Wincc, Step7

Résumé : L'objet d'étude de ce mémoire est la station CE117 Process Trainer. Nous avons procédé dans un premier temps à l'identification des différentes fonctions de transfert à l'aide des méthodes d'identification (Broida, Moments simples), puis nous avons déterminé les paramètres des régulateurs par la méthode de Ziegler-Nichols.
La commande est réalisée par l'automate Siemens S7-300 par le biais d'un projet STEP7-Wincc, nous avons procéder à la commande du niveau, du débit, de la pression, et de la température.
Mots clés : CE 117 Process Trainer, Siemens S7-300, Wincc, Step7.

Control and supervision of the CE 117 Process Trainer station with the PLC Siemens S7 300.

Abstract : The study object of this thesis is the station CE117 Process Trainer. We conducted initially in the identification of different transfer functions using the methods of identification (Broida, Simple Moments), then we determined the parameters of regulators by the method of Ziegler-Nichols.
Control is achieved by the Siemens S7-300 PLC through a STEP7 Wincc project, we proceed to the control of level, flow, pressure, and temperature.

Remerciements

Je dois exprimer ma gratitude envers le bon ***Dieu*** *« Allah » le tout puissant, qui m'a donné la sagesse, la patience, le courage et la volonté pour qu'on puisse terminer ce travail.*

J'exprime mes remerciements à mon prometteur Monsieur Em.Berkouk pour l'assistance qu'il m'a témoigné tout le long de ce travail, qu'il trouve ici l'expression de ma gratitude pour ses conseils.

Je remercie l'ensemble de mes enseignants de l'Ecole Nationale polytechnique qui ont contribué à ma formation tout le long de mon cursus universitaire, en particulier les enseignants d'Automatique.

Je dois absolument adresser une pensée personnelle et très profonde à toute ma ***famille****, pour leurs soutien sans faille, leurs présence émotionnelle ainsi que les nombreux conseils qu'elles m'ont prodigué, et qui m'ont indéniablement permis de mener à bien ce travail.*

Dédicace

Je dédie ce travail à:

Mon père qui a toujours été là pour moi.

Ma mère qui a dédié toute sa vie pour ma réussite.

Mes chères frères.

Mes précieuses sœurs.

Tout mes amis de polytech et de Bouraoui.

Tout le reste de mes amis.

Table des matières

Table des figures

Liste des tableaux

Introduction Générale

La compétitivité industrielle exige de maintenir un procédé le plus près possible de son optimum de fonctionnement prédéfini par un cahier des charges qui regroupe les conditions et les performances imposées, telles que la qualité des produits fabriqués, la flexibilité de la production, la sécurité du personnel et des installations, l'économie de l'énergie et le respect de l'environnement.

Pendant les dernières décennies, grâces aux progrès technologiques liés principalement à la rapidité de traitement des données et les grandes capacités de stockage de l'information, les industries, toutes catégories confondues, ont considérablement évolué grâce aux moyens de haute technologie appliquée aux domaines de la gestion de production, de supervision et la surveillance. Tous les procédés industriels aujourd'hui sont presque automatisés.

Le travail présenté dans ce memoire concerne l'identification des paramètres du CEE 117 process trainer puis à la commande de ce circuit hydraulique équipé de plusieurs instruments et équipements de mesures. Ce travail comporte de nombreux volets et touche à plusieurs disciplines en même temps. Il comporte de plus des parties expérimentales fortes en enseignements et qui permettent, non seulement de voir concrètement l'aboutissement et la finalité de l'étude, mais aussi de faire ressortir les problèmes cruciaux de mise en œuvre.

Afin de bien mener la commande de notre station, et pour réaliser un asservissement idéal, il est impératif pour nous de procéder à l'identification qui va être l'opération de détermination des différentes fonctions de transfert du système.

Après l'identification, vient la détermination des paramètres des régulateurs qui sont nécessaires pour mener une régulation satisfaisante.

Une fois tout ces paramètres sont déterminés, la commande sera réalisée avec un projet Step7-Wincc et nous aurons un très bon asservissement vu qu'on a commandé le système en se basant sur des paramètres qui appartiennent aux système lui-même.

Ce mémoire est organisé en trois chapitres détaillés ci-après. Au Chapitre 1, nous exposons la présentation générale des automates, définitions, types et fonctions.

Le chapitre 2 fait l'objet de presentation du logiciel STEP7 avec toutes ces fonctionnalités et interfaces.

Notre contribution réelle réside dans le chapitre 3, ou nous présentons notre stratégie d'identification des fonctions de transfert du système, la détermination des paramètres des régulateurs ainsi que la commande du système, programmée dans le STEP 7 et chargée dans notre automate.

Vers la fin du troisième chapitre , nous présentons une validation expérimentale de la commande, aussi des différents résultats expérimentaux obtenus, qui valident les théories et la stratégie développée de la commande.

Cette memoire s'achèvera par une conclusion générale résumant les principaux résultats obtenus ainsi que les perspectives ouvertes par ce travail.

Chapitre 1

Automates Programmables Industriels

1.1 introduction

Dans ce chapitre, on va présenter les automates programmables et l'architecture de ses automates ainsi que les langages de programmation et par la suite on va terminer par l'illustration du traitement de l'information dans l'automate.

1.2 Généralités sur les automates programmables industriels

1.2.1 Domaine d'emploi des automates

On utilise les API dans tous les secteurs industriels pour la commande des machines (convoyage, emballage...) ou des chaines de production (automobile, agroalimentaire...) il peut également assurer des fonctions de régulation de processus (métallurgie, chimie...). Il est de plus en plus utilise dans le domaine du bâtiment (tertiaire et industriel) pour le contrôle du chauffage, de l'éclairage, de la sécurité ou des alarmes.

1.2.2 Architecture générale

En général un automate programmable se constitue essentiellement d'une unité centrale, un module d'entrées/sorties, un module d'alimentation, un module de communication et des auxiliaires.

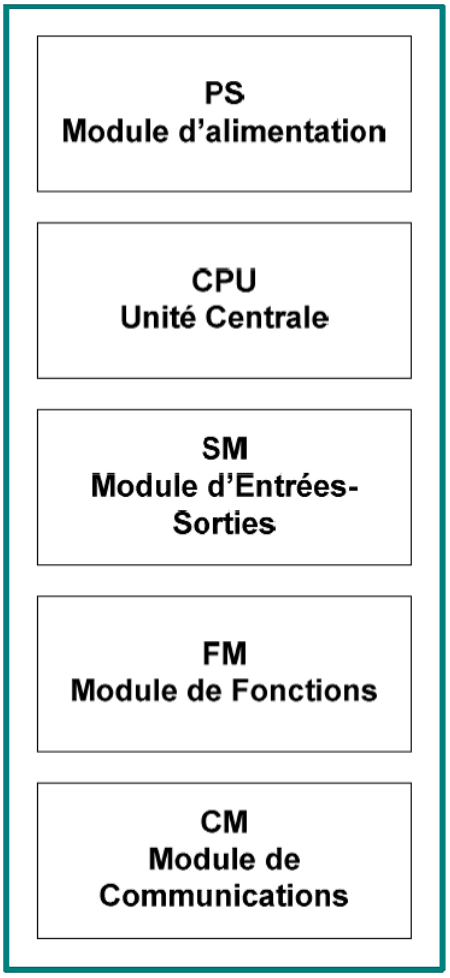


FIGURE 1.1: Architecture d'un automate programmable

1.2.2.1 Le module d'alimentation "PS"

Il est composé de blocs qui permettent de fournir à l'automate l'énergie nécessaire à son fonctionnement, il convertit la tension du réseau (AC 220 V) en tension de service (DC 24V, 12V ou 5V) et assure l'alimentation de l'automate ainsi que circuits de charge.
Un voyant est positionné en générale sur la façade pour indiquer la mise sous tension de l'automate.[1]

1.2.2.2 L'unité centrale "CPU"

La CPU est une carte électronique bâtie autour d'un ou plusieurs processeurs, elle comprend aussi des moyens de stockage, qui sert à sauvegarder les programmes et les données. [3]

1.2.2.3 Le processeur

Le processeur est chargé d'exécuter le programme utilisateur, il doit assurer des opérations logiques et arithmétiques ainsi que des fonctions de temporisation et du comptage. Il peut être issu de la technologie câblée ou de la technologie à microprocesseur. En général un microprocesseur est composé d'une Unité Arithmétique et Logique (UAL), deux registres, un Décodeur d'Instructions, un Compteur Programme et une horloge [2].

1.2.2.4 La mémoire

La mémoire est l'élément fonctionnel qui peut recevoir, conserver et restituer l'information. Elle est découpée en zones :
- une zone mémoire programme ;
- une zone mémoire donnée ;
- une autre pour les variables internes.

Pour un automate, il faut connaître la capacité mémoire minimale utile et la capacité maximale que l'on peut obtenir par diverses extensions.

1.2.2.5 Le module d'entrées/sorties "SM" [3]

Le module E/S assure le rôle d'interface pour la partie commande, qui distingue la partie opérative (les sorties), où les actionneurs agissent physiquement sur le processus, et la partie d'acquisitions (les entrées) récupérant les informations sur l'état de ce processus et coordonnant en conséquence les actions pour atteindre les objectifs prescrits (matérialisés par des consignes). [3] En plus d'assurer la communication entre la CPU et les organes externes, le module d'E/S doit garantir une protection contre les parasites électriques, c'est pour quoi la plus part des modules E/S font appel au découplage optoélectronique. Il existe deux types d'interface E/S :

Le module E/S Tout Ou Rien (TOR)

Permet de raccordé l'automate à des capteurs TOR (boutons poussoirs, fins de course, capteurs de proximité, capteurs photoélectriques ...) ou à des pré-actionneurs (vannes, contacteurs, voyant pneumatique, électrovannes, relais de puissance, LED....). L'état de chaque entrée ou sortie est visualisé par une diode électroluminescente. Le nombre d'entrées sur une carte est de : 4, 8, 16, 32.

Le module E/S analogique

Permet de traiter les signaux analogiques. Il est muni d'un convertisseur analogique/-numérique pour les entrées et un autre numérique/analogique pour les sorties. Il existe des modules à 2, 4, 8 voies.[1]

1.2.2.6 Le module de fonction "FM" (Les cartes spécialisés)

Le module de fonction ou ≪Function Module ≫ est un module additionnel ou des cartes spécialisées peuvent être connectés. Ces cartes comportent un processeur spécifique ou une carte électronique spécialisée, elles assurent non seulement la liaison avec le monde extérieur mais aussi une partie du traitement pour soulager le processeur. On peut citer : les cartes d'axe, les concentrateurs de communication, les cartes E/S déportées, les cartes de comptage rapide, les cartes de pesage, les cartes de régulations PID...

1.2.2.7 Le module de communication "CM"

Le module de communication comprend les consoles et les boîtiers de tests.

1.2.2.8 Les consoles

Les consoles permettent la programmation, le paramétrage et les relevés d'informations, ils peuvent également afficher le résultat de l'autotest comprenant l'état des modules d'entrées et de sorties, l'état de la mémoire, de la batterie, etc. Ils sont équipés (pour la plupart) d'un écran à cristaux liquides. Pendant la phase de programmation les consoles permettent : l'écriture, la modification, l'effacement et le transfert d'un programme dans la mémoire de l'automate ou dans une mémoire EPROM. Pendant la phase de réglage et d'exploitation elles permettent : de visualiser ou d'exécuter le programme pas à pas, de forcer ou de modifier les données (les entrées, les sorties, les bits internes, les registres de temporisation, les compteurs...). Certaines consoles ne peuvent être utilisées que connectées à un automate car c'est ce dernier qui leurs fournit l'alimentation et la mémoire de travail, c'est les consoles de programmation Online, avec ces consoles le programme introduit par l'utilisateur est directement mémorisé dans l'automate.

D'autres consoles peuvent fonctionner de manière autonome grâce à leurs mémoires interne et à leurs alimentations, c'est les consoles de programmation Offline, elles offrent un plus grand confort, le programme écrit de cette façon est appelé source, il est compilé par la console puis transféré dans la mémoire de l'automate. [1]

1.2.2.9 Les boîtiers de tests

Les boîtiers de testes quand a eux sont destinées aux personnels d'entretien, ils permettent de visualiser le programme ou les valeurs des paramètres (affichage de la ligne de programme à contrôler, visualisation de l'état des entrées et des sorties...) [1]

1.2.2.10 Les auxiliaires

Il s'agit principalement de :
•Un support mécanique (un rack) : l'automate se présentant alors sous forme d'un ensemble de cartes, d'une armoire, d'une grille et des fixations correspondantes.
•Un ventilateur : il est indispensable dans les châssis comportant de nombreux modules ou dans le cas où la température ambiante est susceptible de devenir assez élevée (plus de 40C).
•Un indicateurs d'état : il indique la présence de tension, l'exécution du programme (mode RUN), la charge de la batterie, le bon fonctionnement des coupleurs.

1.2.2.11 Nature des informations traitées par l'automate [5]

Les informations peuvent être de type :
1. Tout ou rien (T.O.R.) :
l'information ne peut prendre que deux états (vrai/faux, 0 ou 1...). C'est le type d'information délivrée par un détecteur, un bouton poussoir... etc.
2. Analogique : l'information est continue et peut prendre une valeur comprise dans une plage bien déterminée. C'est le type d'information délivrée par un capteur (pression, température ..).
3. Numérique : l'information est contenue dans des mots codes sous forme binaire ou bien hexadécimale. C'est le type d'information délivrée par un ordinateur ou un module intelligent.

1.3 Language de programmation

1.3.1 Introduction

La norme IEC 1131-3 (la Commission Électrotechnique Internationale) définit cinq langages qui peuvent être utilisés pour la programmation des automates programmables industriels. Ces langages peuvent être divisés en deux grandes catégories :

Langages graphiques :

- SFC ≪ Sequential Funiculite Chart ≫ ou GRAFCET
- LD ≪ Ladder Diagram ≫ou schéma à relais
- FBD ≪ Function Block Diagram ≫ou schéma par bloc

Langages textuels :

ST ≪ structured text ≫ ou texte structuré
- IL ≪ Instruction List ≫ ou Liste d'instructions

1.3.2 Les langages graphiques

Le GRAFCET

L'acronyme GRAFCET signifie : GRAphe Fonctionnel de Commande Etape Transition (SFC Sequential Function Chart). C'est une méthode de représentation graphique permettant de décrire le cahier de charge d'un automatisme. Il est adapté aux systèmes à évolution séquentielle, il est défini par un ensemble d'éléments graphiques de base traduisant le comportement de la partie commande vis-à-vis de ses entrées et de ses sorties.

Un programme GRAFCET décrit un procédé comme une suite d'étapes, reliées entre elles par des transitions. À chaque transition est associée une réceptivité, celle-ci est une condition logique qui doit être vraie pour franchir la transition et passer à l'étape suivante. Des actions sont associées aux étapes du programme.

Le format graphique d'un programme GRAFCET est le suivant :

Une étape représentée par un carré qui a un numéro identificateur et les actions associées sont indiquées dans un rectangle relié à la partie droite du carré ; (l'étape initiale est représentée par un carré double).
Une liaison orientée représentée par une ligne, parcourue par défaut de haut en bas ou de gauche à droite. Une transition entre deux étapes à qui est associé une réceptivité inscrite a ça droite, est représentée par une barre perpendiculaire aux liaisons orientées qui relient ces étapes.

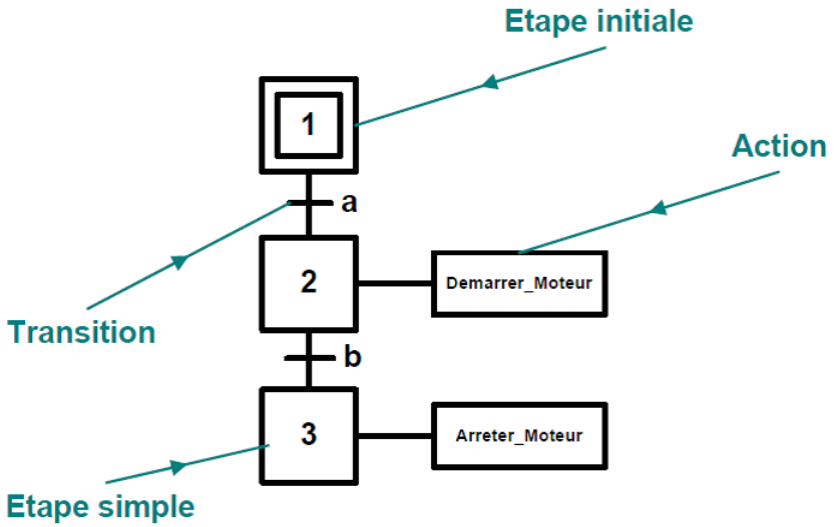


FIGURE 1.2: Le GRAFCET

Ladder Diagram :

Le LD est une représentation graphique qui traduit directement des équations booléennes en un circuit électrique en combinant des contacts et des relais a l'aide des connexions horizontales et verticales, les contacts représentent les entrées (contact normalement ouvert, contact normalement fermé,...) et les relais représentent les sorties (relais directs, relais inversés,...), les diagrammes LD sont limités sur la gauche par une barre d'alimentation et par la masse sur la droite.
Par exemple pour réaliser la fonction logique $x = (a + b)(\bar{c} + \bar{d}\ e)$

voici le programme LD :

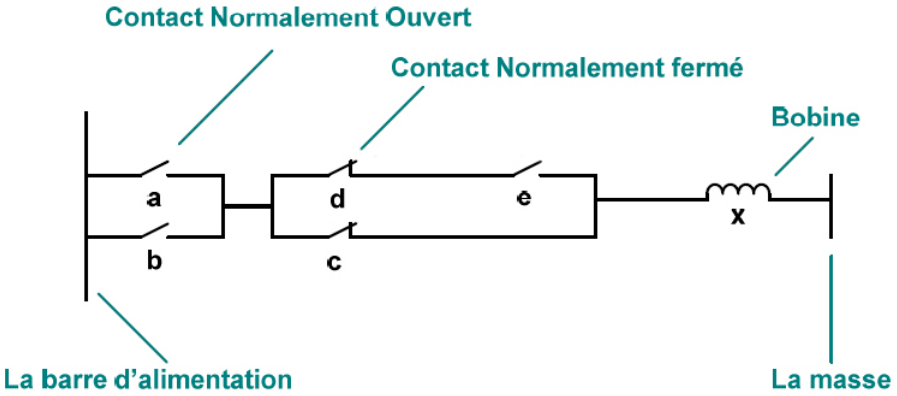


FIGURE 1.3: Exemple d'un programme LADDER

Le langage LD propose d'autre type de fonction tel que les fonctions de comptages et de temporisations, les fonctions arithmétiques et logiques, les fonctions de comparaison et de transfert. Par exemple pour réaliser la fonction

$Z = (X \geq Y)$

On utilise directement la fonction déjà disponible.

Bloc de Fonction

C'est un langage graphique qui permet la construction d'équations complexes à partir des opérateurs standard, ou de blocs fonctionnels, il se compose de réseaux de fonctions préprogrammées ou non, représentées par des rectangles qui sont connectés entre eux par des lignes.
La programmation avec le FBD est très souple et facile à apprendre, la plupart des fonctions nécessaires (les fonctions arithmétique et logique, les fonctions de temporisation, des blocs fonctionnels PID...) sont déjà disponible dans la bibliothèque, il suffit juste de les connecter et bien paramétrer les entrées et les sorties, c'est-à-dire respecter le type des variables lors de la connexion.

1.3.3 Les langages textuels

Texte Structuré

Le langage ST (Structured Text) est un langage de programmation textuel de haut niveau dédié aux applications d'automatisation, il est utilisé principalement pour décrire les procédures complexes et difficilement modélisables avec les langages graphiques, il peut aussi être utilisé entant que sous programme avec d'autre langage de programmation.
Il utilise les même énoncés que les langages de programmation de haut niveau (Pascal, C, C++...) comme : les assignations, les appels de fonction, les énoncés de contrôle (IF, THEN, ELSE, CASE) ou d'itération (FOR, WHILE, REPEAT) en plus des opérations arithmétiques et logiques.

Liste d'instructions

Le langage IL est un langage textuel de bas niveau (proche du langage machine), qui utilise un jeu d'instruction simple, il trouve sa puissance dans les applications de petites tailles, et dans la création de sous programme ou procédure, car il permet un contrôle

totale et une optimisation parfaite du code, par contre dans les grandes applications il est très difficile de programmer avec le IL, les programmes dans ce langage peuvent être traduit ou déduit des autres langages.
Le IL a la même structure que l'assembleur, il utilise un ou plusieurs registres de travail. Les valeurs intermédiaires nécessaires pour l'exécution d'une instruction donnée seront mémorisées dans ces registres le temps de leurs utilisations et il possède un jeu d'instruction d'assez riche pour décrire toutes les opérations arithmétiques et logiques, les opérations de comptage et temporisation, la comparaison et le transfert...

1.4 Traitement du programme automate

Tous les automates fonctionnent selon le même mode opératoire :

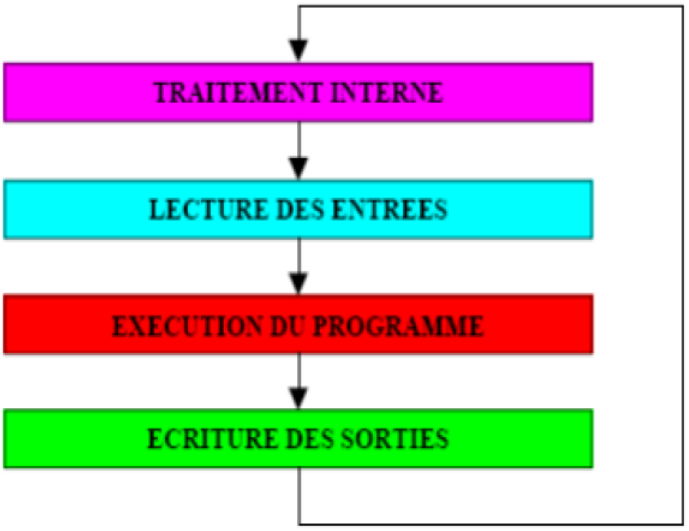


FIGURE 1.4: Traitement du programme automate

1.4.1 Traitement interne

L'automate exécute des opérations de contrôle et met a jour certains paramètres systèmes (détection des passages en RUN/STOP, mises a jour des valeurs de l'horodateur,...)

1.4.2 Lecture des entrées

L'automate lit les entrées (de façon synchrone) et les recopie dans la mémoire image des entrées.

1.4.3 Exécution du programme

L'automate exécute le programme instruction par instruction et écrit les sorties dans la mémoire image des sorties.

1.4.4 Écriture des sorties

L'automate bascule les différentes sorties (de façon synchrone) aux positions définies dans la mémoire image des sorties. Ces quatre opérations sont effectuées continuellement par l'automate (fonctionnement cyclique). On appelle scrutation l'ensemble des quatre opérations réalisées par l'automate et le temps de scrutation est le temps mis par l'automate pour traiter la même partie de programme. Ce temps est de l'ordre de la dizaine de millisecondes pour les applications standards.

1.4.5 Le temps de réponse total (TRT)

Le temps de réponse total (TRT) est le temps qui s'écoule entre le changement d'état d'une entée et le changement d'état de la sortie correspondante :
1. Le temps de réponse total est au plus égal à deux fois le temps de scrutation.
2. Le temps de scrutation est directement lié au programme implanté. Ce temps peut être fixé à une valeur précise (fonctionnement périodique), le système indiquera alors tout dépassement de période.
Dans certains cas, on ne peut admettre un temps de réponse aussi long pour certaines entrées : ces entrées pourront alors être traitées par l'automate comme des évènements (traitement événementiel) et prises en compte en priorité (exemples : problème de sécurité, coupure d'alimentation ...)
Certains automates sont également pourvues d'entrées rapides qui sont prises en compte avant le traitement séquentiel mais le traitement évènementiel reste prioritaire.

1.4.6 Sécurité

Les systèmes automatisés sont, par nature, source de nombreux dangers (tensions utilisées, déplacements mécaniques, jets de matière sous pression ...). Placé au cœur du système automatisé, l'automate se doit d'être un élément fiable car :
- un dysfonctionnement de celui-ci pourrait avoir de graves répercussions sur la sécurité des personnes.
- les couts de réparation de l'outil de production sont généralement très élevé.
- un arrêt de la production peut avoir de lourdes conséquences sur le plan financier.

Aussi, l'automate fait l'objet de nombreuses dispositions pour assurer la sécurité :
- Contraintes extérieures : l'automate est conçu pour supporter les différentes contraintes du monde industriel et à fait l'objet de nombreux tests normalisés (tenue aux vibrations, CEM ...)
- Coupures d'alimentation : l'automate est conçu pour supporter les coupures d'alimentation et permet, par programme, d'assurer un fonctionnement correct lors de la réalimentation (reprises à froid ou à chaud).
- Mode RUN/STOP :Seul un technicien peut mettre en marche ou arrêter un auto- mate et la remise en marche se fait par une procédure d'initialisation (programmée).
- Contrôles cycliques :
- Procédures d'autocontrôle des mémoires, de l'horloges, de la batterie, de la tension d'alimentation et des entrées/sorties.
- Vérification du temps de scrutation à chaque cycle appelée Watchdog (chien de garde), et enclenchement d'une procédure d'alarme en cas de dépassement de celui-ci (règle par l'utilisateur).
- Visualisation : Les automates offrent un écran de visualisation ou l'on peut voir l'évolution des entrées/sorties. La défaillance d'un automate programmable pouvant avoir de graves répercussions en matière de sécurité, les normes interdisent la gestion des arrêts d'urgence par l'automate ; celle-ci doit être réalisée en technologie câblée. On peut également ajouter des modules de sécurité à l'automate (sécurité des machines).
Il existe enfin des automates dits de sécurité (APIdS) qui intègrent des fonctions de surveillance et de redondance accrues et garantissent la sécurité des matériels.

1.5 Conclusion

La plus part des grands constructeurs d'automates programmables, fournissent des logiciels de configuration et de programmation munis des langages SFC, LD, FBD, ST et IL. Le choix d'un langage s'appuie sur la complexité de l'application et de la tache de commande. Il est préférable d'utiliser les langages graphiques (SFC, LD et FBD) pour la réalisation des programmes de commande séquentielle. Le SFC est la réalisation direct d'un GRAFCET de commande, les langages LD et FBD sont plus utiles pour les opérations combinatoires sur bits ou mots.
Les langages textuels sont beaucoup plus performants pour le traitement de variables continues ou analogiques ainsi que pour la commande des systèmes continus. Les programmes en IL sont un peu fastidieux à mettre en œuvre, mais connaissent une optimisation parfaite pour le temps de traitement et l'occupation de la mémoire. Le ST est le langage par excellence, très utile pour des utilisateurs ayant des connaissances en langages évolués tels que PASCAL.

Chapitre 2

step7

2.1 introduction

Le step7 est un logiciel de programmation des automates simens , dans ce chapitre on va faire une présentation du logiciel STEP7 ainsi que les étapes de création d'un projet Step7 pour finaliser par une simulation avec le logiciel PLSIM.

2.2 Description du STEP7 [9]

STEP7 est le logiciel de base pour la programmation et la configuration de systèmes d'automatisation SIMATIC. Le progiciel de base STEP 7 existe en plusieurs versions :

- STEP 7-Micro/DOS et STEP 7-Micro/Win pour des applications autonomes simples sur SIMATIC S7 - 200.
- STEP 7 pour des applications sur SIMATIC S7-300/400, SIMATIC M7-300/400 et SIMATIC C7.

2.3 Fonctions du logiciel step7 [9]

Le logiciel step7 nous assiste dans toutes les phases du processus de création de nos solutions d'automatisation, comme par exemple :

- La création et la gestion de projets.
- La configuration et le paramétrage du matériel et de la communication,
- La gestion des mnémoniques,

- La création de programmes, par exemple pour les systèmes cibles S7,
- Le chargement des programmes dans des systèmes cibles,
- Le test de l'installation d'automatisation,
- Le diagnostic lors de perturbations de l'installation.
- Introduction sur le produit et installation.

2.4 Applications disponibles

Il inclut 6 applications :

1 **Gestionnaire de projets SIMATIC :**
il gère toutes les données relatives à un projet d'automatisation. Il démarre automatiquement les applications requises pour le traitement des données sélectionnées.

2 **Editeur de mnémoniques :**

il permet de gérer toutes les variables globales. C'est à dire la définition de désignations symboliques et de commentaires pour les signaux du processus (entrées/sorties), mémentos et blocs, l'importation et l'exportation avec d'autres programmes Windows. [9]

3 **Diagnostic du matériel :**
il fournit un aperçu de l'état du système d'automatisation. Dans une représentation d'ensemble, un symbole permet de préciser pour chaque module, s'il est défaillant ou pas. De plus permet l'affichage d'informations générales sur le module et son état, l'affichage d'erreurs sur les modules de la périphérie centrale et des esclaves DP et l'affichage des messages de la mémoire tampon de diagnostic.

4 **Langages de programmation :**

trois langages de programmation sont inclus dans le logiciel de base : CONT (LD Ladder Diagram), LIST (IL Instruction List) et LOG (FBD Function Bloc Diagram), d'autre langage de programmation peuvent être procurés sous forme de logiciel additionnel : le SCL (ST Structured Text) et le GRAPH (GRAFCET). [7]

5 **Configuration matérielle :**

il permet de configurer et paramétrer le matériel d'un projet d'automatisation. Il suffit juste de sélectionner le châssis (Rack) dans un catalogue électronique et leurs affecter les modules sélectionnés aux emplacements souhaités dans les racks (CPU, SM, FM...). De plus il permet le paramétrage de la CPU (comportement

à la mise en route, surveillance du temps de cycle), des modules fonctionnels (FM) et de processeurs de communication (CP).

6 **NetPro :**

il permet le transfert de données via MPI tout en offrant les possibilités de choisir les participants à la communication et de définir les liaisons de communication.

2.5 conception d'une structure programme complète

2.5.1 Création du projet SIMATIC Step7

Un projet comprend deux données essentielles, les programmes et la configuration du matériel, on peut commencer par définir l'une ou l'autre, mais tout d'abord il faut démarrer le programme SIMATIC Manager. Ce programme est l'interface graphique qui permet la manipulation du projet et l'accès aux autres programmes de STEP7.

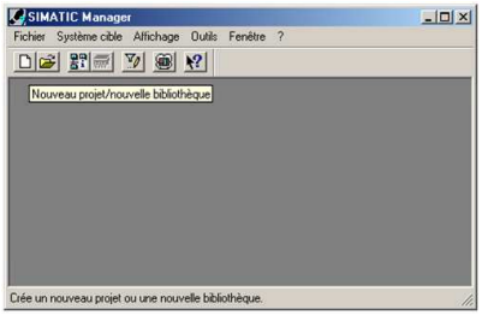


FIGURE 2.1: création d'un nouveau projet

Pour en créer un nouveau, il suffit de cliquer sur le bouton 'Nouveau projet', attribuer un nom et valider. Ensuite il faut choisir une station de travail. Une station SIMATIC représente une configuration matérielle S7 comportant un ou plusieurs modules programmables. Il existe différents types :

- SIMATIC 400 :
  Automate à performances extrêmes, adapté à l'exécution de programme de lourds calculs.
- SIMATIC 300 :
  Automate à extensibilité modulaire.
- SIMATIC H :
  Automate insensible aux défaillances, il se compose de 2 CPU du même type, en cas de problème elle commute de l'une vers l'autre sans perte de données.

- SIMATIC PC :
  ou Station PC, représente un PC ou une station OS contenant des composants SIMATIC : des applications (WinCC, par ex.), un automate logiciel ou une carte CPU enfichée dans le PC.
- Autres stations :
  se sont soit des appareils d'autres fabricants ou bien des stations de SIMATIC S7 contenus dans un autre projet.
- SIMATIC S5 : liaison vers un projet S5.
- PG/PC :
  Outils de programmation pour contrôleurs SIMATIC, c'est une console de programmation compatible avec le milieu industriel.

Par exemple on va choisir une station SIMATIC 300

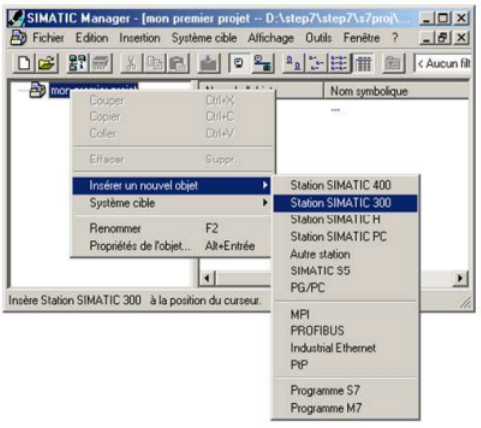


FIGURE 2.2: choix de la station de travail

Pour commencer, le plus simple est de configurer le matériel, d'éditer les programmes puis les charger dans la CPU. Avec un double clic sur « Matériel », on démarre l'application HW Config.

2.5.2 Configuration du matériel

Pour configurer le matériel, il suffit de faire glisser des éléments du catalogue dans l'emplacement approprié, on choisit le Rack, l'alimentation, la CPU et les E/S... Dans le catalogue on trouve les modules qu'on peut affecter à chaque type de station, on distingue :

- C7 :
  Système intégré compact qui regroupe automate programmable et interface homme machine (pupitre opérateur) pour la réalisation de commandes de machines sous encombrement réduit.

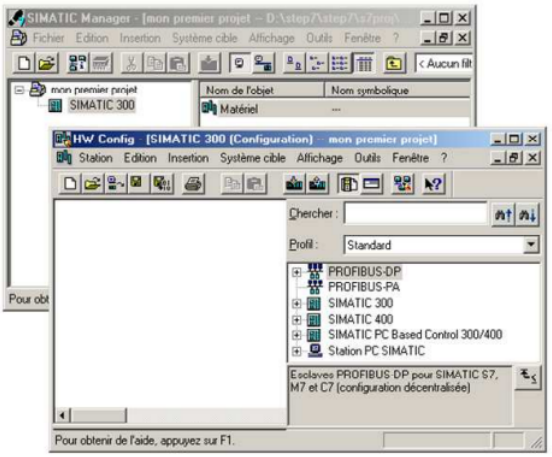


FIGURE 2.3: Configuration matériels

- CP :
  Communication Processor, module de communication (PROFIBUS, Industriel Ethernet,....).
- FM :
  (Function Module), il regroupe les modules de fonctions (régulation, comptage...).
- IM :
  Coupleurs d'extension, il permet l'ajout d'autres modules.
- M7 :
  Modules d'extension et cartouches interface pour SIMATIC M7. • PS : Module d'alimentation.
- Rack : Support mécanique.
- Routeur : Relie Industriel Ethernet à PROFIBUS.
- SM :
  Signal Module, c'est le module d'E/S, il contient le AI module d'entrées analogiques, le AO module de sorties analogiques, le DI module d'entrées TOR et le DO module de sorties TOR.
- CPU : L'Unité Centrale, noté CPU xxx a b.
  xxx est la famille de la CPU.
  a, b sont les propriétés de la CPU (éléments additionnels, port de communication...). Par exemple :

  - C : compact, la CPU intègre des modules E/S ainsi que des fonctions spécialisées.
  - PtP : Peer to Peer, la CPU intègre un port de communication Point to Point.
  - H : Fault-tolerant, des unités de traitements insensibles aux défaillances

- DP : Decentralized Periphery, la CPU intègre un port de communication PROFIBUS.

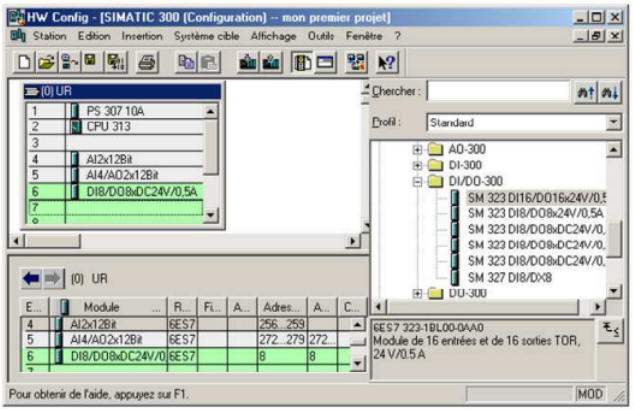


FIGURE 2.4: sélection du module

Si l'insertion de l'élément choisi est possible dans le Rack, la case appropriée devient verte. Une fois le matériel choisi on sauvegarde, on compile et on charge dans la CPU.

2.5.3 Définition des mnémoniques

De retour dans le SIMATIC Manager, on trouve de nouveaux éléments. On commence par créer les mnémoniques dans la section programmes.

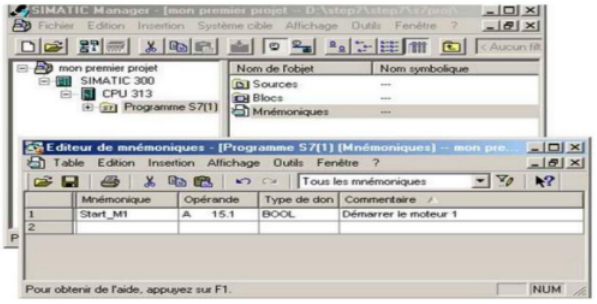


FIGURE 2.5: Edition des Mnémoniques

En affectant des noms symboliques aux adresses absolues, les programmes deviennent plus lisible, faciles à corriger et à mettre à jour. Il y a quatre différents types d'opérande : le bit, l'octet, le mot et le double mot. Ces types définissent l'accès à une zone mémoire. Pour chaque opérande un certain type de données est permis :

- Pour le bit : BOOL : variable booléenne (True ou False, 1 ou 0).
- Pour l'octet : deux types de données sont possibles :
  1. BYTE : nombre hexadécimal de B♯16♯0 à B♯16♯FF.
  2. CHAR : Caractère ASCII, 'A', 'B'...

- Pour le mot : quatre types de données sont possible :
  1. WORD : nombre hexadécimal de W♯16♯0 à W♯16♯FFFF.
  2. INT : nombre entier de -32768 à 32767.
  3. S5TIME : Durée S7 en pas de 10 ms (valeur par défaut), de S5T♯0H-0M-0S-10MS à S5T♯2H-46M-30S-0MS.
  4. DATE : Date en incréments de 1 jour, de D♯1990-1-1 à D♯2168-12-31.
- Pour le double mot : cinq types de données :
  1. DWORD : nombre hexadécimal de DW♯16♯0000-0000 à DW♯16♯FFFF-FFFF.
  2. DINT : nombre entier de L♯-2147483648 à L♯2147483647.
  3. REAL : nombre à virgule flottante, Limite supérieure : 3.402823e+38 Limite inférieure : 1.175 495e-38.
  4. TIME : Durée en incréments de 1 ms, de -T♯24D-20H-31M-23S-648MS à T♯24D-20H-31M-23S-647MS.
  5. TIME-OF-DAY : Heure en pas de 1ms, de TOD♯0 :0 :0.0 à TOD♯23 :59 :59.999

2.5.4 Édition des programmes

Dans la section 'bloc' du SIMATIC Manager, on trouve par défaut le bloc d'organisation 1 'OB1'qui représente le programme cyclique. On peut rajouter d'autres blocs à tout moment par un clic droit dans la section Bloc de SIMATIC Manager.

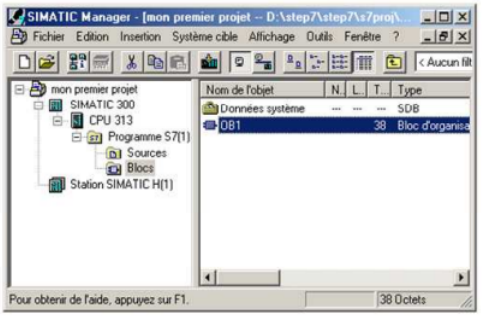


FIGURE 2.6: Édition des programmes

Deux programmes différents s'exécutent dans la CPU : le système d'exploitation et le programme utilisateur. Le système d'exploitation, organise toutes les fonctions et procédures dans la CPU qui ne sont pas liées à une tâche d'automatisation spécifique. Il gère le déroulement du démarrage à chaud et du redémarrage, l'actualisation de la mémoire image des entrées et l'émission de la mémoire image des sorties, l'appel du programme utilisateur, la gestion des zones de mémoire l'enregistrement des alarmes et l'appel des OB d'alarme...

Le programme utilisateur contient toutes les fonctions nécessaires au traitement des tâches d'automatisation spécifique. Ce programme doit être créé et chargé dans la CPU par l'utilisateur. Il détermine les conditions pour le démarrage à chaud et le redémarrage de la CPU (par exemple, initialiser des signaux), il traite les données du processus (par exemple, combiner des signaux binaires, lire et exploiter des valeurs analogiques), il doit réagir aux alarmes et traiter les perturbations dans le déroulement normal du programme.

Le STEP7 permet de structurer le programme utilisateur en le subdivisant en différentes parties autonomes ou dépendantes. Ceci permet d'écrire des programmes importants mais clairs, simples à tester et à modifier.

2.5.5 Simulation de modules Avec le logiciel PLCSIM [4]

S7-PLCSIM dispose d'une interface simple permettant de visualiser et de forcer les différents paramètres utilisés par le programme (comme, par exemple, d'activer ou de désactiver des entrées). En outre, S7-PLCSIM possède les fonctions suivantes :

- On peut créer des « fenêtres » dans lesquelles on a la possibilité d'accéder aux zones de mémoire d'entrée et de sortie, aux accumulateurs ainsi qu'aux registres de la CPU de simulation. On peut également accéder à la mémoire par adressage symbolique (il faut juste charger la table des mnémoniques dans 'options', puis sur 'outils' 'insérer mnémoniques').

- On peut sélectionner l'exécution automatique des temporisations ou encore les définir et les réinitialiser manuellement.
- On a la possibilité de changer l'état de fonctionnement de la CPU (STOP, RUN et RUNP) comme pour une CPU réelle. De plus, on dispose d'une fonction de pause qui permet d'interrompre momentanément la CPU, sans affecter l'état du programme.

2.5.6 Chargement du programme dans la CPU

Une fois la configuration, le paramétrage et la création du programme terminés, on peut transférer le programme utilisateur dans le système cible. La CPU contient déjà le système d'exploitation.

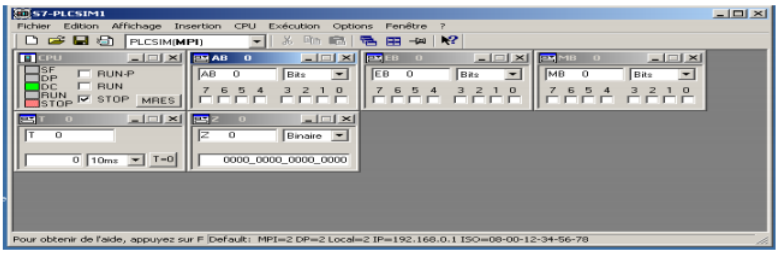


FIGURE 2.7: logiciel de simulation PLC-SIM [4]

2.5.7 Surveillance du fonctionnement et diagnostic du matériel [6]

La détermination des causes d'un défaut dans le déroulement d'un programme utilisateur se fait à l'aide de la « Mémoire tampon de diagnostic », accessible depuis le SIMATIC Manager.

Chapitre 3

Identification et commande a l'aide d'un projet Step7-WinCC

3.1 introducion

Dans ce chapitre, on va décrire les méthodes d'identification utilisées dans ce travail.par la suite on va décrire la station CE117 process trainer ainsi que les sous systèmes et on finira par les résultats de régulation et le schéma de supervision.

3.2 Méthodes d'identification

3.2.1 Méthode de BROIDA

La méthode de Broïda consiste à assimiler le procédé régulé à un système du **premier ordre avec retard**, dont la **fonction de transfert** est : $H(p) = \dfrac{Ke^{-Tp}}{1+\tau p}$

(p : variable de Laplace)

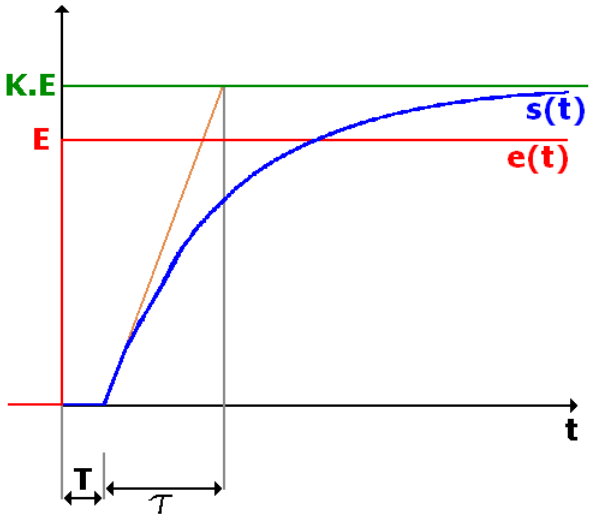
Il s'agit de déterminer les coefficients :
✓K : gain statique du procédé ;
✓T : temps mort d'identification ;
✓τ : constante de temps.

La transformée de Laplace permet de ramener la résolution des équations différentielles linéaires à coefficients constants à la résolution d'équations simples

•l'intégration est transformée en division par p ;

•la dérivation est transformée en multiplication par p.

L'essai d'identification s'effectue en imposant un échelon de commande au procédé.



$$H(p) = \frac{Ke^{-Tp}}{1+\tau p}$$

La détermination directe de τ et T n'est pas toujours aisée sur une courbe réelle : on préfère mesurer t_2 et t_1 et calculer τ et T

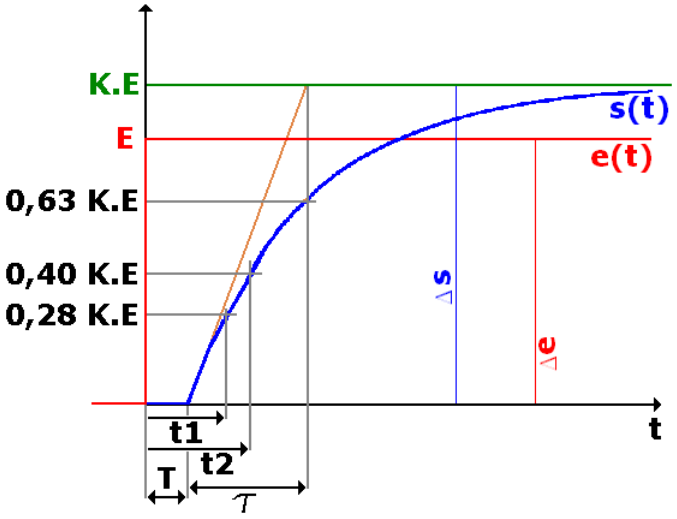


FIGURE 3.1: Modèle optimal si $\frac{T}{\tau} < 0,25$

$$\tau = 5,5(t_2 - t_1)$$

$$T = 2,8t_1 - 1,8t_1$$

$$K = \frac{\Delta s}{\Delta e}$$

3.2.2 Méthode des Moments

Le processus d'identication consiste à estimer les paramétres inconnus de la dynamique du systéme. En conséquence, détermination de la structure du systéme est d'une grande importance dans le processus d'identification du systéme.

3.2.2.1 Calcul des moments

On appelle moment d'ordre k d'une fonction f(t) la quantité :

$$M_k = \frac{(-1)^k}{k!} \int_0^\infty t^k f(t) dt, \quad k = 0, 1, 2, \quad ... \tag{3.1}$$

On d'apres les properties de la transformation de Laplace :

$$[t^k f(t)] = (-1)^k \frac{d^k F(s)}{ds^k} = t^k f(t) e^{-st} dt \tag{3.2}$$

$$(-1)^k (-1)^k \frac{d^k F(s)}{ds^k} = (-1)^k \int_0^\infty t^k f(t) e^{-st} dt \tag{3.3}$$

$$\frac{1}{k!} \frac{d^k F(s)}{ds^k} = \frac{(-1)^k}{k!} \int_0^\infty t^k f(t) e^{-st} dt \tag{3.4}$$

$$\frac{(-1)^k}{k!} \int_0^\infty t^k f(t) e^{-0t} dt = \frac{1}{k!} \frac{d^k F(s)}{ds^k}|_{s=0} \tag{3.5}$$

donc :

$$M_k = \frac{1}{k!} \frac{d^k F(s)}{ds^k}|_{s=0} \tag{3.6}$$

3.2.2.2 Modèle second ordre

Soit un modéle du second ordre tel que sa fonction transfert :

$$H(s) = \frac{k}{1 + a_1 s + a_2 s^2} \tag{3.7}$$

On veut calculer les paramétres de ce modéle avec la méthode des moments,on applique l'équation 3.6, on aura besoin des 3 premiers moments M0 ,M1 et M2 :

$$M_k = \frac{1}{k!}\frac{d^k F(s)}{ds^k}|_{s=0} \quad \Rightarrow \quad \begin{cases} k = M_0 \\ a_1 = -\frac{M_1}{M_0} \\ a_2 = (\frac{M_1}{M_0})^2 - \frac{M_2}{M_0} \end{cases} \tag{3.8}$$

En se basant sur ces deux méthodes en a crée un code *matlab* et une interface graphique sous *matlab* (voir fig 3.2)

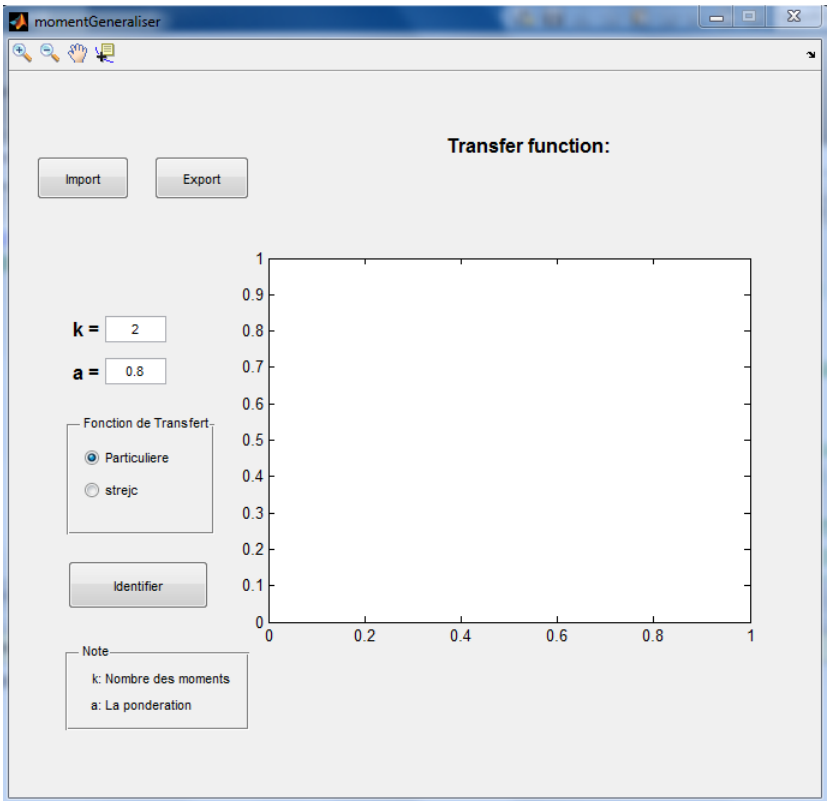


FIGURE 3.2: Interface graphique de l'identification

Pour faire l'identification, on doit créer un fichier de valeurs (entrés/sorties) et temps d'échantillonnage, pour pouvoir ensuite l'importer sur l'interface qui va nous calculer la fonction de transfert et les deux réponses réelle et identifiée.

3.3 Présentation du CEE 117 Process Trainer

Le système CE117 Process Trainer est un appareil de contrôle de processus autonome entièrement intégré. Les équipements de cet appareil permettent de contrôler :
le niveau, le débit, la pression et la température. Notre procédé se compose de trois

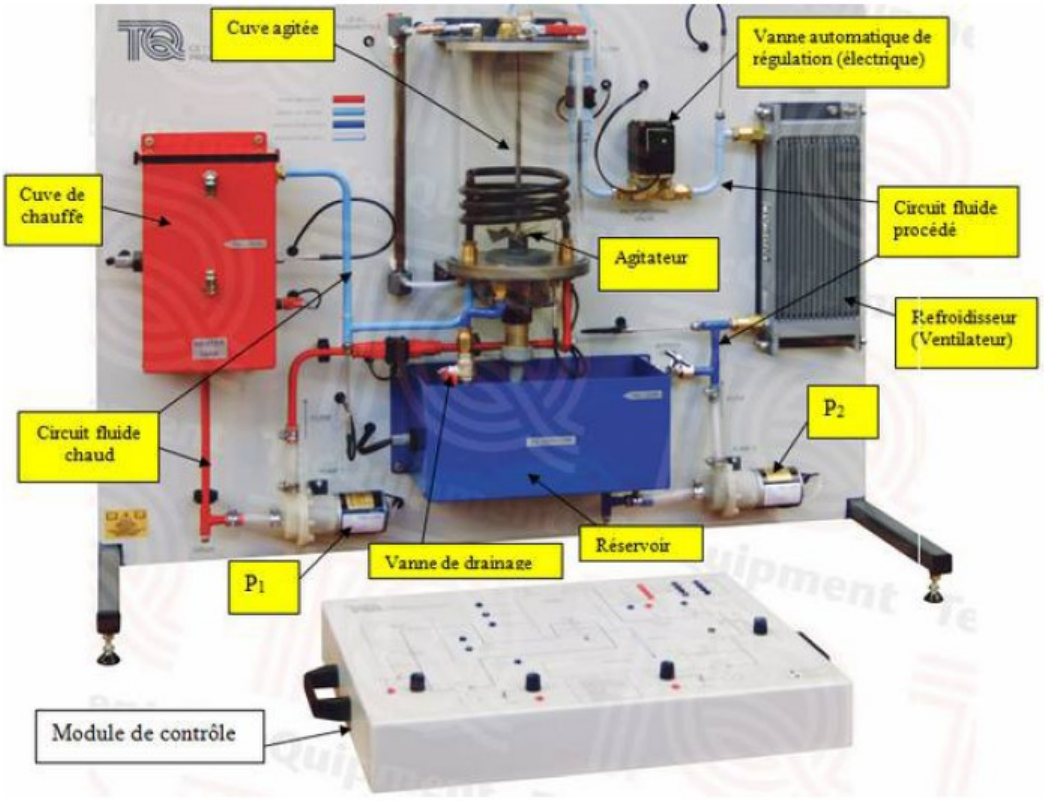


FIGURE 3.3: Présentation du CEE 117 Process Trainer

parties : une cuve agitée, un circuit de fluide procédé et un circuit de fluide chaud. Pour des raisons de sécurité, l'eau est utilisée comme fluide de travail.

3.3.1 Le module de contrôle

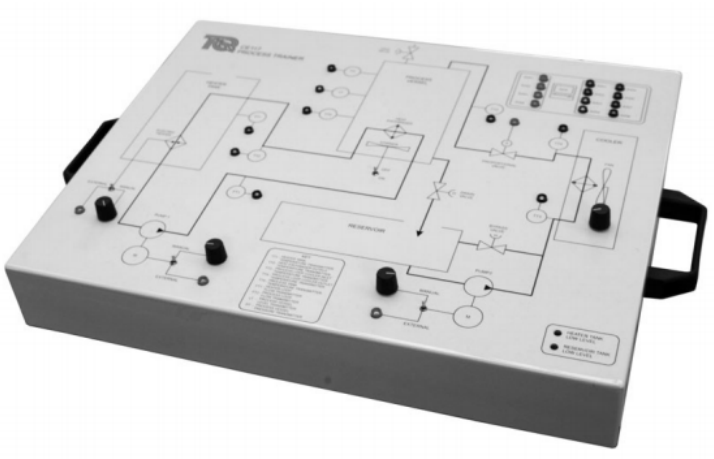


FIGURE 3.4: Description du module de contrôle du process trainer CE117

Notre process trainer comporte un pupitre qui représente un module de contrôle. Ce module regroupe l'ensemble des circuits de cablage des actionneurs et des capteurs-transmetteurs au sein de notre processus ainsi que l'interface avec le procédé. On trouve sur ce module un schema du procédé à étudier ainsi que les connexions des differents capteurs-transmetteurs et actionneurs avec l'interface ADA du logiciel CE2000 qui est lié au module par le biais d'une liaison série..

Les capteurs-transmetteurs fournit 8 entrées à l'interface ADA de notre module de contrôle. Ces entrées de nature analogique seront converties en numérique. L'interface comporte 4 sorties converties en analogique destinées aux actionneurs (voir fig 3.5).

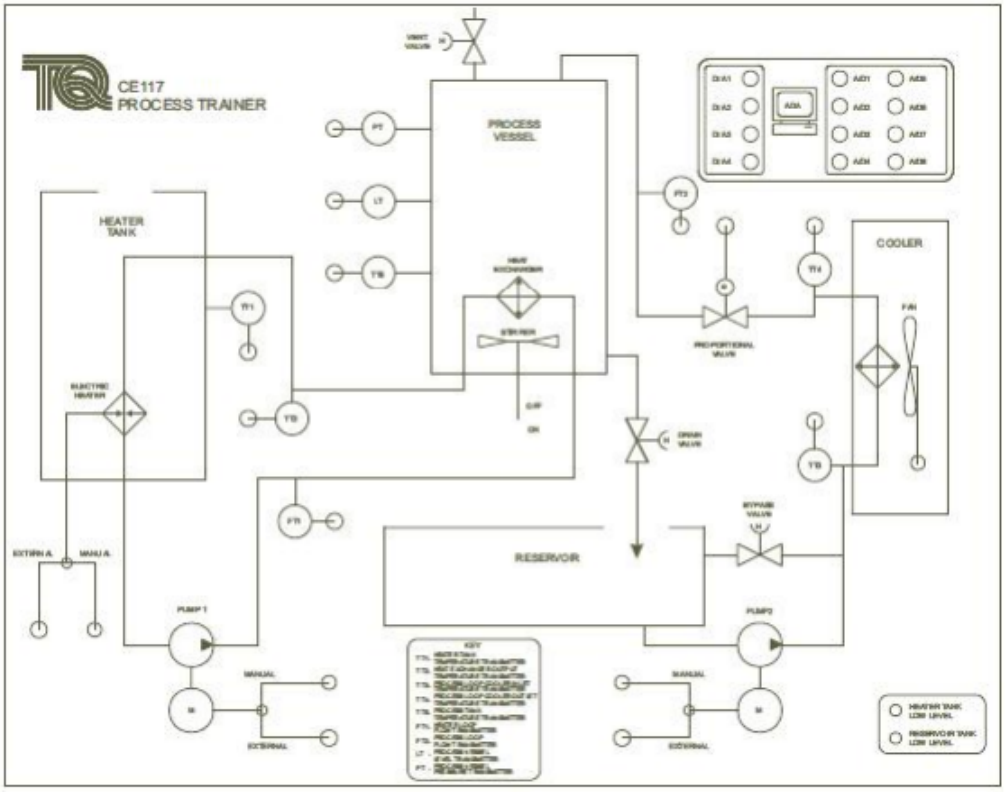


FIGURE 3.5: Schéma du module de contrôle du process trainer CE117

le tableau suivant résume la nature des entrées et des sorties dans le module de contrôle :

article		Signal Analogique	Détail de la conversion
PRT Température Transmetteurs (résistance platinuim Thermomètre)	TT1 TT2 TT3 TT4 TT5	Sortie 0-10V linéaire	$10°C$ par Volt 0V= $0°C$ 10V= $100°C$
Transmetteur de débit	FT1 FT2	Sortie 0-10V linéaire	1L/min par Volt 0V=pas de débit
Transmetteur de niveau	LT	Sortie 0-10V non linéaire	0V=récipiant vide 10V=Niceau Maximum
Transmetteur de pression	PT	Sortie 0-10V linéaire	100mbar par volt 0V=0mbar (jauge)
chauffage électrique		Sortie 0-10V	75W par volt 0V=chauffage OFF 10V=750W puissance max(Nominale)
Vanne Proportionnelle	S	Sortie 0-10V	0V=fermée 10V=ouverte
Pompe 1 Pompe 2		Sortie 0-10V	0V=pas de débit 10V=débit Maximal

TABLE 3.1: nature des entrées et des sorties du module de contrôle

3.4 Description du système utilisé pour contrôler le débit et le niveau

Pour réguler le débit et le niveau de notre cuve agitée, nous allons utiliser la partie droite de notre système, c'est-à-dire le circuit de fluide procédé.
La figure suivante nous montre cette partie en nommant ces différents composants :

Dans ce circuit, le fluide se dirige depuis le réservoir vers la cuve agitée par le biais de la pompe P2, et cela en traversant le refroidisseur, ensuite le fluide revient au réservoir par la gravité lorsque la vanne de drainage est ouverte.

Notre système comporte deux capteurs de température avant et après le refroidisseur et un troisième capteur de débit, ainsi qu'une vanne By-pass qui représente la perturbation de notre système.

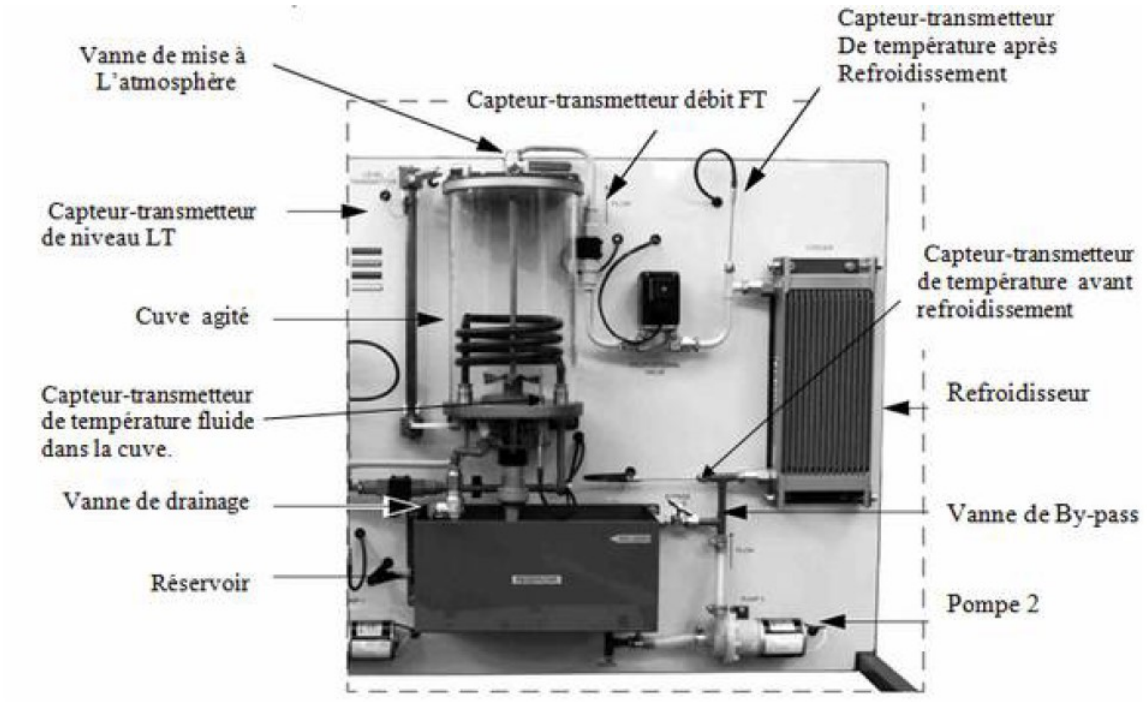


FIGURE 3.6: Description du système utilisé pour contrôler le débit et le niveau

Les deux éléments clés du système sont la pompe P2 et la vanne proportionnelle qui se situe entre le refroidisseur et la cuve agitée.

Dans tous le travail qui suit on a utilisé le logiciel CE2000 pour injecter des entrées sur le système ainsi pour récupérer les données par l'extraction des données sous format text donc on injecte une entrée et en récupère la sortie avec un temps d'échantillonnage de notre choix.

3.5 Régulation de débit

Dans notre système, le débit peut être contrôlé par l'intermédiaire de deux types de commande :

- La régulation du débit en utilisant la vanne proportionnelle comme contrôleur.
- La régulation du débit en utilisant la pompe comme contrôleur.

3.5.1 Système pompe-débit

Le fonctionnement de la station est géré par l'automate, l'opérateur peut intervenir a l'aide d'une écran de supervision ou il peut charger sa configuration et visualiser l'état

de la station .

pour la régulation du débit avec la pompe comme actionneur on a fait un programme du réglage PID ou l'operateur entre la valeur de la consigne sur l'écran de supervision et le système va automatiquement se réguler a l'aide de l'automate qui va commander la pompe a travers une commande calculée par le régulateur PID et pour faire ce programme on a choisit l'adressage des entrées et de sortie comme le montre le tableau suivant :

PEW 128	débit
PEW 130	Niveau
PEW 132	Température dans la cuve
PEW 134	Ouverture de la vanne
PAW 128	l'entrée de la pompe

TABLE 3.2: Réglage du débit -pompe- (Manip1)

tel que l'indice PEW et PAW représentent respectivement les entrées et les sorties analogique.

3.5.1.1 Identification du système pompe-débit

Pour l'identification, on utilise la méthode de Broïda, injectant un échelon de (80%) à l'entrée de la pompe et en introduit la réponse obtenue a l'aide du logiciel CE2000 comme un vecteur (Ys,t). La figure 3.7 représente les résultats de l'identification :

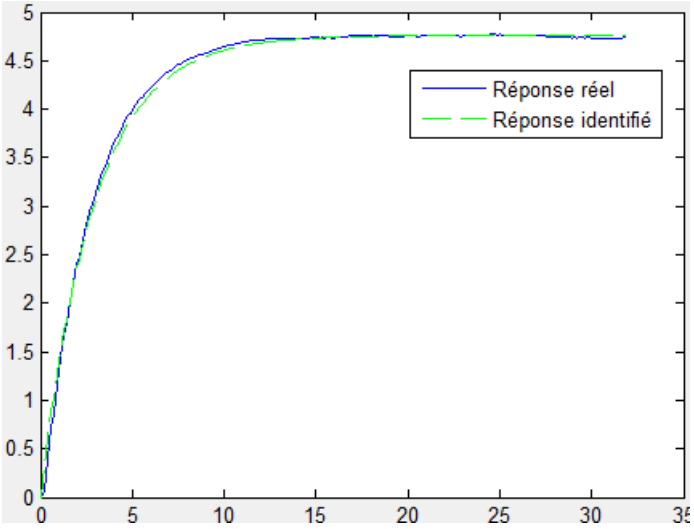


FIGURE 3.7: Réponse réelle et identifiée du système pompe-débit en boucle ouverte

- **La fonction de transfert en boucle ouverte du système pompe-débit**

La fonction de transfert est la suivante (eq :3.9) :

$$F(s) = \frac{0.2127}{s + 0.3593} \tag{3.9}$$

3.5.1.2 Synthèse du régulateur pour le système pompe-débit

Pour le calcule des parcmètres du régulateur pour ce système, plusieurs méthodes peuvent être appliquées, a savoir :

- le calcule analytique des paramètres du régulateur PI.
- l'utilisation de la méthode de Ziegler-Nichols pour trouver les paramètres du régulateur PID.

Notre méthode consiste a fixer les paramètres Kp et Kd du régulateur a zero (Kp=Kd=0) puis faire augmenter le gain de l'action intégrale Ki jusqu'à l'obtention d'un dépassement de 15 à 25 % de la consigne, une fois c'est fait on va augmenter le gain de l'action proportionnel Kp pour faire disparaître ce dépassement, et pour finir on va aussi augmenter le gain de l'action dérivée Kd pour avoir une réponse plus rapide.

les paramètres du régulateur trouvés sont les suivants :

- Kp=5.2.
- KI=0.7.
- Kd=0.

le schéma bloc en boucle fermée de l'ensemble régulateur système pompe-débit est représenté sur la figure suivante :

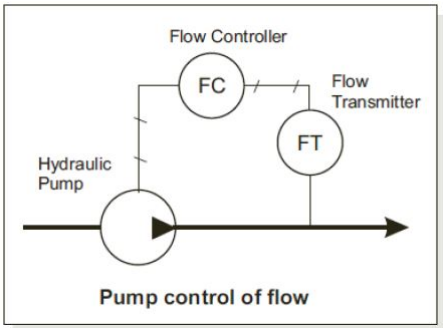


FIGURE 3.8: l'ensemble régulateur système pompe-débit en boucle fermée

3.5.1.3 Réponse en boucle fermée du système (sous WinCC)

Après avoir implémenté les paramètres du régulateur PID dans le programme de l'automate, on a obtenu la réponse de ce système en boucle fermé representée sur la figure 3.9

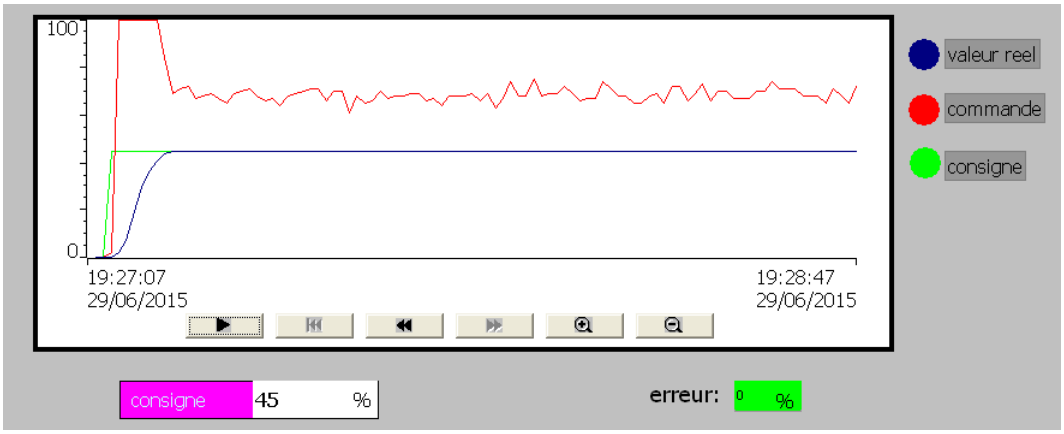


FIGURE 3.9: Réponse réelle du système Pompe-débit en boucle fermée

On remarque bien que pour une consigne de 45% la commande appliquée a assurée la régulation.

3.5.2 Système Vanne-débit

pour le systeme vanne-debit on a suit le meme chaiher de charge sauf que l'automate va commander la vanne au lieu de la pompe avec une commande calculée par le regulateur PID qui est programmé dans l'automate avec la possibilté d'introduire la consigne par l'operateur et on a choisit l'adressage suivant :

PEW 128	débit
PEW 130	Niveau
PEW 132	Température dans la cuve
PEW 134	Température dans les tubes
PAW 128	l'entrée de la vanne

TABLE 3.3: Réglage du débit -vanne- (Manip1)

3.5.2.1 Identification du système Vanne-débit

On utilise la même méthode que la régulation par pompe, les résultats sont montrés sur la figure 3.10

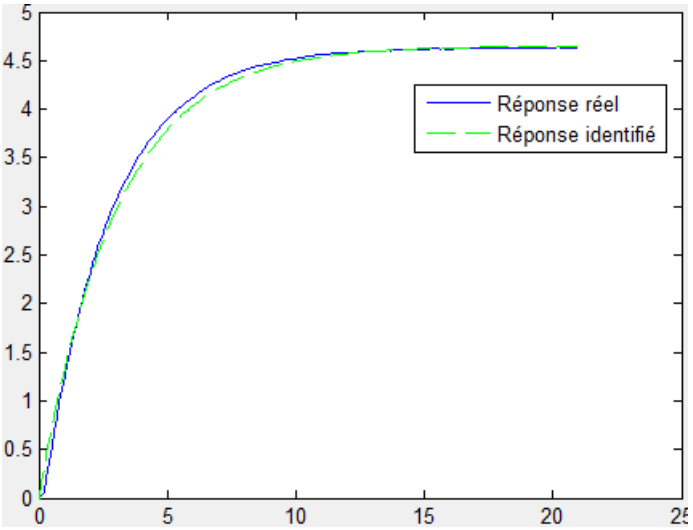


FIGURE 3.10: Réponse l'ensemble régulateur système Vanne-débit en boucle fermée

- **La fonction de transfert en boucle ouverte du système Vanne-débit**

La fonction de transfert est la suivante (eq :3.10) :

$$F(s) = \frac{0.2846}{s + 0.333} \tag{3.10}$$

3.5.2.2 Synthèse du régulateur pour le système Vanne-débit

Comme nous l'avons mentionnés précédemment, plusieurs méthodes peuvent être appliquées pour le calcule des parcmètres du régulateur.

Notre méthode est la même considérée pour le système pompe-débit, elle consiste a fixer les paramètres Kp et Kd du régulateur a zero (Kp=Kd=0) puis faire augmenter le gain de l'action intégrale Ki jusqu'à l'obtention d'un dépassement de 15 à 25 % de la consigne, une fois c'est fait on va augmenter le gain de l'action proportionnel Kp pour faire disparaître ce dépassement, et pour finir on va aussi augmenter le gain de l'action dérivée Kd pour avoir une réponse plus rapide.

les paramètres du régulateur trouvés sont les suivant :

- Kp=5.6.
- KI=0.5.

- Kd=0.

le schéma bloc en boucle fermée de l'ensemble régulateur système pompe-débit est représenté sur la figure suivante :

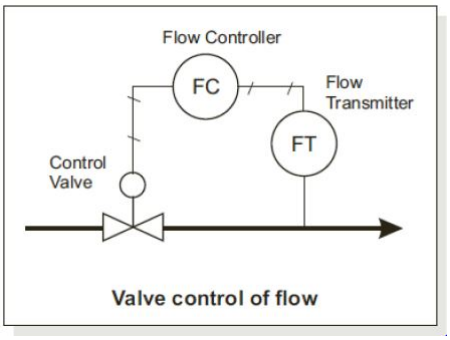


FIGURE 3.11: l'ensemble régulateur système Vanne-débit en boucle fermée

3.5.2.3 Réponse en boucle fermée du système (sous WinCC)

On introduit les valeurs du régulateur PI dans notre programme, la figure suivante (3.12) représente la réponse du système en Boucle fermée.

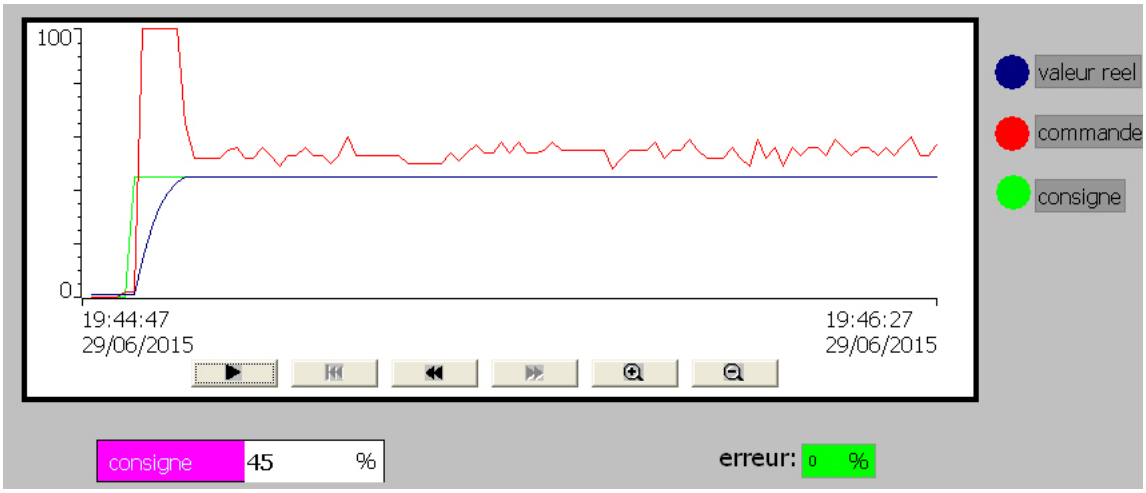


FIGURE 3.12: Réponse réelle du système Vanne-débit en Boucle fermée.

3.6 Régulation de niveau

Sur cet appareil, nous pouvons procéder à la régulation par deux façons différentes : on peut choisir la pompe P 2 comme actionneur lors de la régulation comme on peut réguler le niveau par le biais de la vanne proportionnelle.

3.6.1 système pompe-niveau

dans ce manip l'automate commande le niveau, le programme chargé dans l'automate a pour but de régler du niveau de l'eau dans la cuve, et pour assurer cette régulation on a fait un programme de régulateur PID et un programme WinCC ou l'operateur peut introduire la valeur de la consigne . Dans ce programme ona choisit l'adressage des entrées et de sortie comme le montre le tableau suivant :

PEW 128	Niveau
PEW 130	débit
PEW 132	Température dans la cuve
PEW 134	pression
PAW 128	l'entrée de la pompe

TABLE 3.4: Réglage du Niveau (Manip2)

3.6.1.1 Identification du système pompe-niveau

Notre système est assimilé à celui du deuxième ordre, ayant comme fonction de transfert en boucle ouverte :

$$F(s) = \frac{K}{a_0 s^2 + a_1 s + a_2} \tag{3.11}$$

En utilisant la méthode des moments simples, nous procédons à l'identification en faisons entrer les valeurs de notre entrée et de notre sortie dans l'interface graphique, nous obtenons les résultats suivants (fig 3.13) :

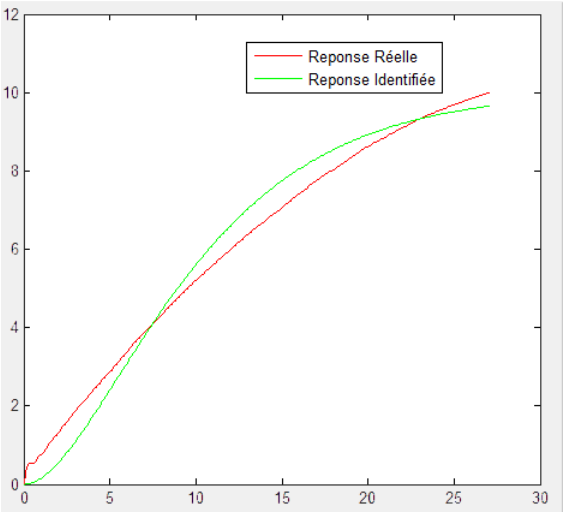


FIGURE 3.13: Résultats obtenues de l'interface graphique

On trouve que la fonction de transfert est la suivante :

$$F(s) \quad = \quad \frac{1,01}{(27,8336)s^2 + (10,4933S)s + 1} \tag{3.12}$$

3.6.1.2 Synthèse du régulateur

Afin d'éliminer l'erreur statique obtenue, nous allons utiliser un régulateur de type PI, ayant comme fonction de transfert :

$$R(s) \quad = \quad K_p + \frac{K_I}{s} \tag{3.13}$$

En intégrant ce régulateur, nous bouclons notre système et nous calculons ses paramètres en boucle fermée avec la methode de Ziegler-Nicholz on trouve : kp=3.9 KI=0.03.

3.6.1.3 Réponse en boucle fermée du système (sous WinCC)

Après avoir faire un programme du régulation PID avec les paramètres trouvés précédemment, on obtient les résultats montrés sur la figure suivante 3.14.

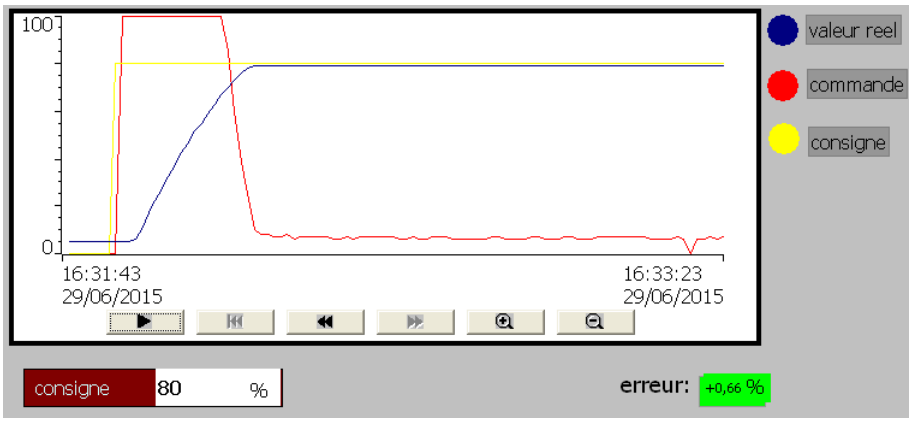


FIGURE 3.14: Réponse réelle du système Niveau-Pompe en Boucle fermée

On remarque que le système suit la référence avec une erreur statique négligeable de l'ordre de 0.66%.

3.6.2 système niveau-vanne

pour faire cette régulation on a suit un simple cahier de charge, si la valeur donnée par le capteur de niveau on ouvre la vanne proportionnelle a 90% si la valeur de la consigne est égale a la valeur réelle on ferme la vanne et toujours l'operateur peut entrer la valeur de la consigne a travers l'ecran de supervision. En se basent sur le cahier de charge on a crée un projet Step7-WinCC avec l'adressage suivant :

PEW 128	Niveau
PEW 130	débit
PEW 132	Température dans la cuve
PEW 134	pression
PAW 128	l'entrée de la pompe

TABLE 3.5: Réglage du Niveau (Manip2)

3.6.2.1 Identification du système niveau-vanne

De la même manière qu'avec la pompe, nous procédons à l'identification en utilisant la méthode des moments simples, nous obtenons les résultats suivants (fig 3.15) :

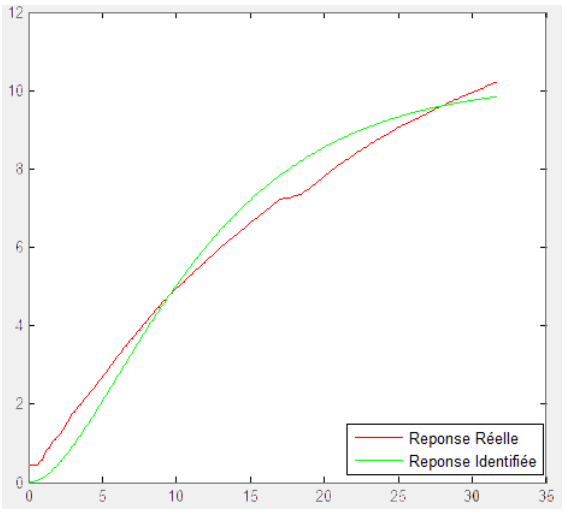


FIGURE 3.15: Résultats obtenues de l'interface graphique

Après identification, voici les résultats des paramètres de la fonction de transfert :

$$F(s) \quad = \quad \frac{1,01}{(33,7979)s^2 + (11,8325)s + 1} \tag{3.14}$$

Pour cette partie on va appliquer une commande de type tout ou rien (TOR).

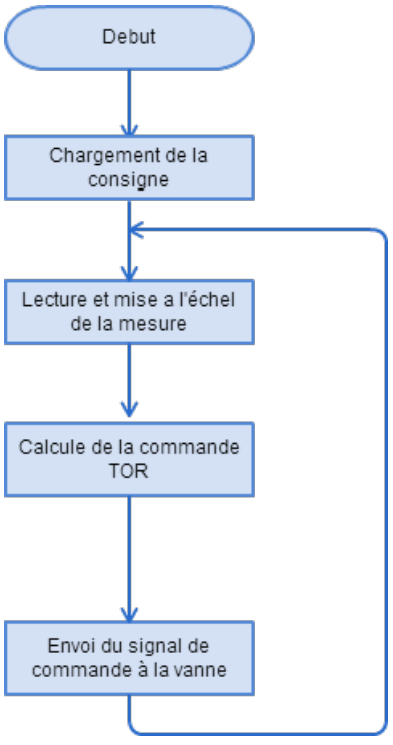


FIGURE 3.16: Organigramme de la régulation de Niveau.

En se basant sur l'organigramme ci-dessus on a développé un programme qui a pour objectif de faire la régulation. l'application de ce programme nous a donnée les résultats montrée sur la figure suivante 3.17

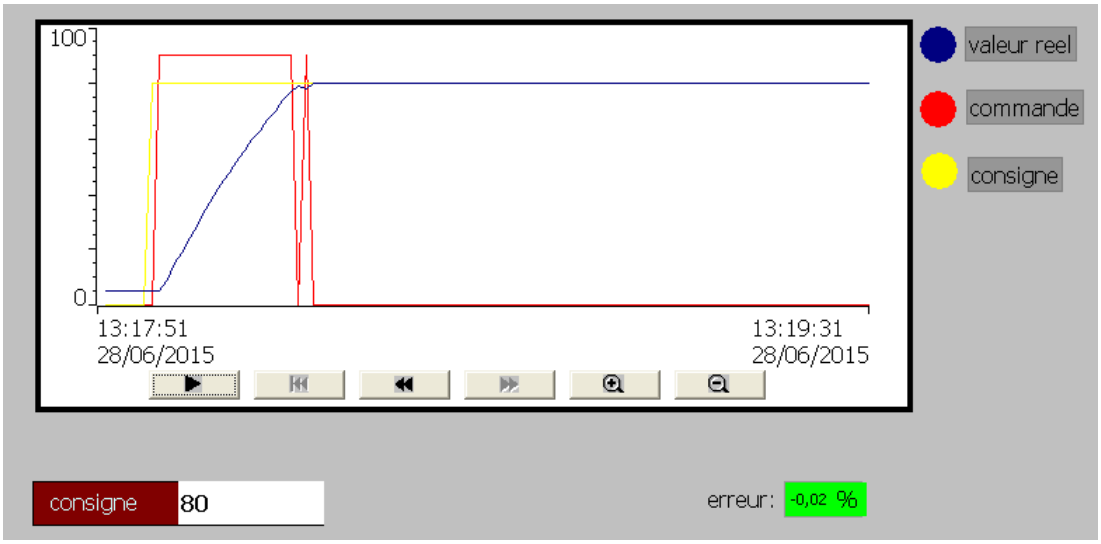


FIGURE 3.17: Réponse réelle du système Niveau-vanne en BF.

On remarque que le système est bien régulé, et que la réponse suit la référence.

3.7 Réglage de la température

Notre but dans cette approche est de faire commander la température dans la cuve à une température de référence et de la faire maintenir autour de cette dernière. Le schéma ci-dessous décrit la méthodologie de régulation de la température via l'algorithme de réglage ON-OFF(régulateur à hystérésis).

La régulation à hystérésis à été largement utilisé dans la littérature. Ce succès est du à la simplicité de sa mise en œuvre et sa robustesse vis-à-vis des perturbations externe.

La technique de régulation à hystérésis consiste a amener la trajectoire d'état(le pont de régulation) du système à étudier vers une bande de surface et de la faire commuter à l'aide d'une logique de commutation appropriés autour de point désiré, cette logique de commutation dans le but borner l'erreur entre la consigne et la mesure dans un intervalle (-a,+a) est établi en présence deux commande max et celle min.

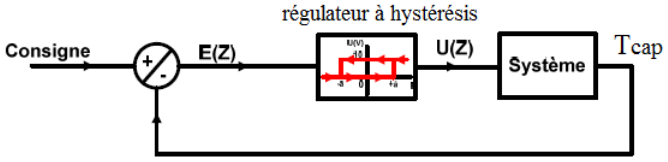


FIGURE 3.18: la régulation à hystérésis

Ou :

E(z) : représente l'erreur de réglage numérique entre la température capté par le capture et celle de référence.

U(z) : La commande délivré par le bloc d'hystérésis. La commande U peut prendre 3 valeurs et ceci dans un intervalle de temps bien déterminé : On représente la commande à l'instant k comme suit :

$$U_k = \begin{cases} 90\% \quad si \quad e(k) \succeq 1 \\ U_{k-1} \quad si \quad -1 < e(k) < +1 \\ 0\% \quad si \quad e(k) < -1 \end{cases} \tag{3.15}$$

pour suivre le cahier de charge on a fait un programme Step7 aves l'adressage suivant :

PEW 128	Température du *Heater*
PEW 130	débit
PEW 132	Température de retour
PEW 134	Température dans la cuve

TABLE 3.6: Réglage de temperature (Manip4)

3.7.1 L'organigramme de l'algorithme

L'expression de la commande dépend énormément de la comparaison de l'erreur à l'instant k avec les borne d'hystérésis (+1 ou –1), l'organigramme définissant le chemin de la commande étudié pour le réglage de la température est donné comme suit :

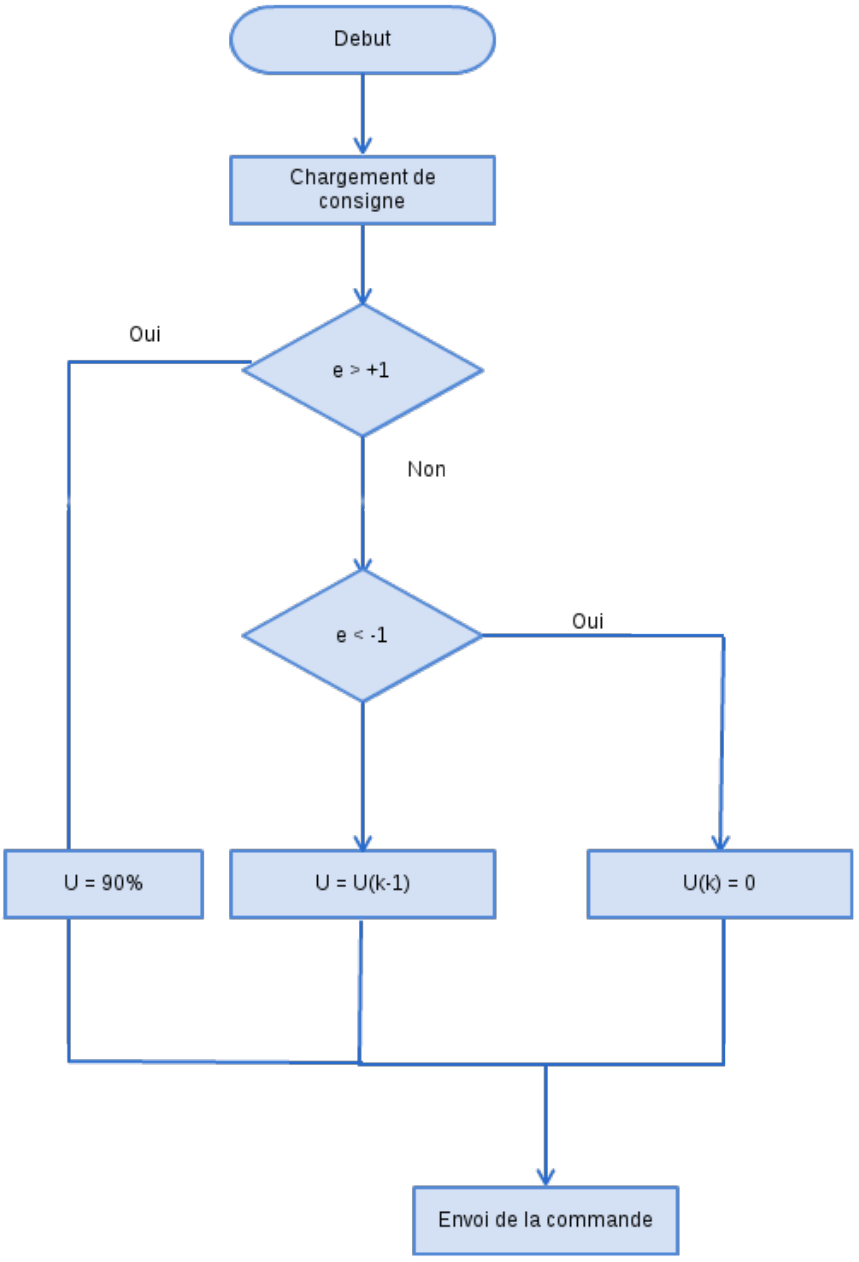


FIGURE 3.19: L'organigramme de réglage.

3.7.2 La synthèse de la commande

Notre objectif décrit dans cette partie est d'appliquer une commande TOR le but de réguler et de faire maintenir la température du chauffage à une valeur de référence fixé. Pour ce faire Pour mieux comprendre le phénomènes et la méthodologie de réglage,

3.8 Réglage de la Pression

Régulation de pression :
La pression dans la cuve est une image du niveau de l'eau : quand le niveau augmente la pression augmente. Pour commander la pression dans la cuve on a développé un projet step 7 . Description du projet :

Le prjet step7 permet de contrôler la valeur de pression dans la cuve en utilisant la vanne proportionnelle comme actionneur, et on va suivre dans ce programme les règles suivantes :

- si la valeur de la consigne est supérieure à celle de la pression de sorte que la différence entre elles est supérieure à 7%, on va appliquer une commande de 90% sur la vanne.
- si la différence entre la consigne et la pression mesurée est entre 2 et 7%, on va appliquer une commande de 60% sur la vanne.
- si la différence est inferieure à 2%, on va appliquer une commande de 40% sur la vanne.
- si la différence est nulle, la commande sur la vanne doit être nulle.

avec l'adressage suivant :

PEW 128	Pression
PEW 130	Niveau
PEW 132	débit
PEW 134	Température dans la cuve
PAW 128	l'entrée de la vanne

TABLE 3.7: Réglage de Pression (Manip3)

L'organigramme ci-dessous (fig 3.20) représente le programme précédent :

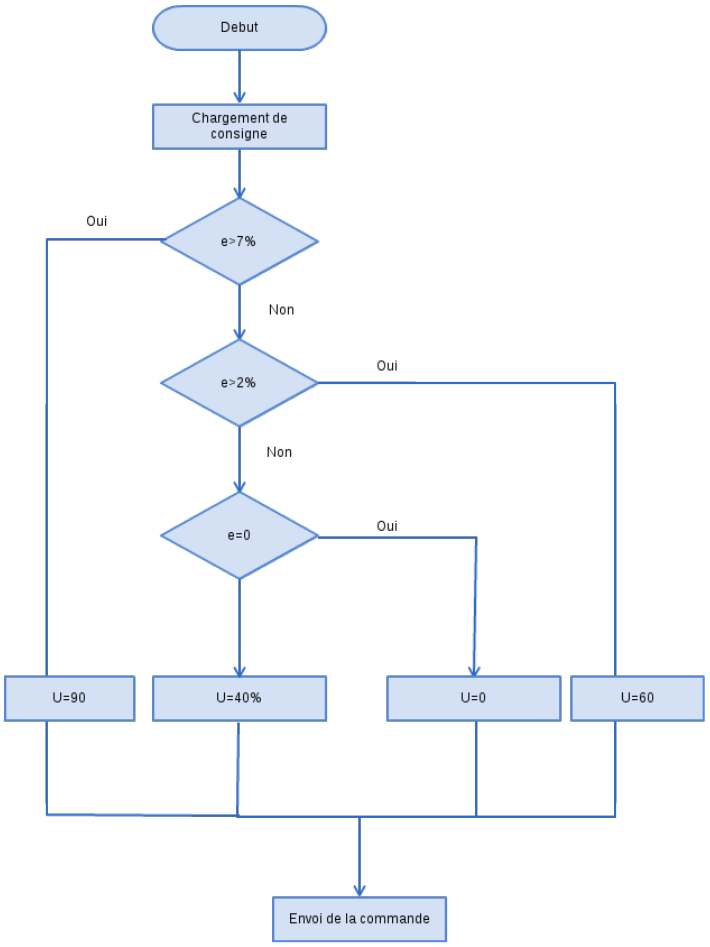


FIGURE 3.20: L'organigramme de réglage.

3.8.0.1 Réponse en boucle fermée du système (sous WinCC)

Après avoir implémenté le programme sur Step7 et la création d'un projet de supervision WinCC on a pu visualiser la réponse du système imposant une référence de 40%. les résultats sont illustrés dans la figure 3.21.

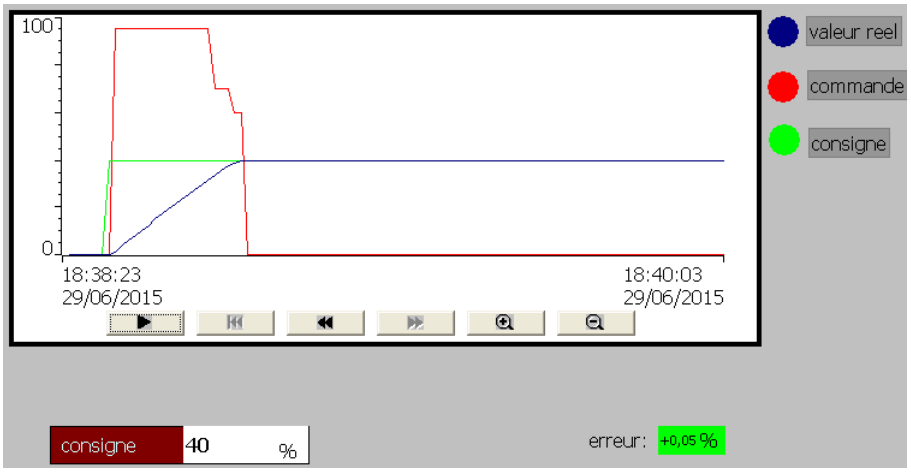


FIGURE 3.21: Réponse réelle de la pression en boucle fermée.

On remarque bien que le système suit la référence d'une façon quasi parfaite avec une erreur de 0.05%

3.9 Schéma de supervision sous WinCC

Pour la supervision et la commande de la station CE117, on a crée une interface homme machine par l'intermédiaire du logiciel WinCC.

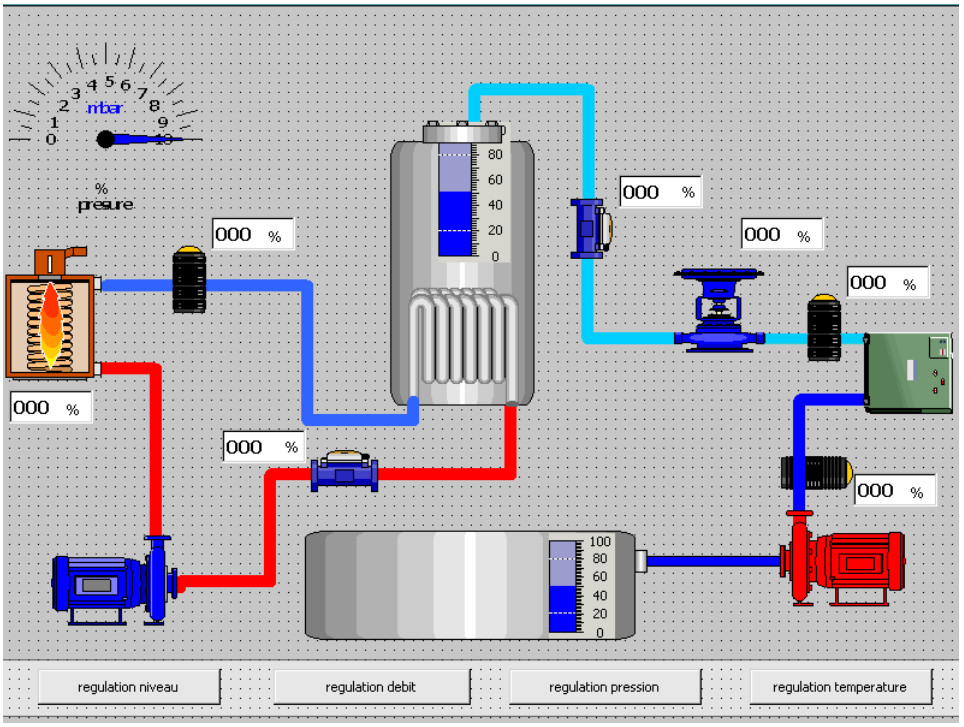


FIGURE 3.22: Schéma de supervision sous WinCC.

Conclusion générale

Nous avons étudié dans ce memoire l'apport des techniques avancées de l'automatique appliquées à l'identification et la commande du CEE117.

Le premier et le deuxième chapitre avaient pour but de présenter quelques généralités sur les automates et le logiciel de simulation utilisées dans notre approche.

La suite de ce mémoire a été consacrée à la présentation de la méthode d'identification des paramètres du CEE 117, ainsi que l'application de la commande des actionneurs a l'aide de l'automate s7 300 et le projet crée sur le STEP 7.

A la fin, une présentation des résultats expérimentaux de la simulation du programme de commande, qui illustre et confirme sa fonctionnalité et sa robustesse.

A la lumière des observations et des résultats présentés dans ce memoire des perspectives intéressantes pouvant contribuer à l'amélioration du fonctionnement de notre dispositif expérimental, du point de vue de la commande, d'autres techniques plus intelligentes peuvent être testées pour commander notre circuit.on site la commande multivariable ou on peut commander plusieurs états ou bien on peut faire une régulation en cascade par exemple regulation debit et niveau en cascade.

Annexe A

Annexe A

A.1 Le Bloc OB35 [8]

L'OB35 est un bloc dit *d'alarme cyclique*, son utilisation assure une fréquence d'appel constante du bloc SFB41 (régulateur PID). Ceci est primordial pour que le régulateur puisse être optimisé grâce au réglage de ses paramètres KP, TN et TV. Ceci serait impossible si l'on utilisait l'OB1 qui offre une fréquence d'appel incertaine.

Dans le programme de configuration matérielle, il est possible de choisir la fréquence d'exécution de l'OB35 en allant dans les propriétés de la CPU sous l'onglet, *Alarmes cycliques*. Ce temps ne doit pas être trop court.

On doit s'assurer que le reste du programme et notamment l'OB1 ait le temps de s'exécuter entre deux appels de l'OB35.

A.2 Le Bloc FB41

Le bloc FB 41 ≪ CONT-C ≫ (continuous controller) sert à réguler des processus industriels à grandeurs d'entrée et de sortie continues sur les automates programmables SIMATIC S7. Le paramétrage nous permet d'activer ou de désactiver des fonctions partielles du régulateur PID et donc d'adapter ce dernier au système régulé.

A.2.1 Utilisation

On pout utiliser le régulateur comme régulateur PID de maintien autonome mais aussi comme régulateur en cascade, de mélange ou de rapport dans des régulations à plusieurs boucles. Sa méthode de travail se base sur l'algorithme PID du régulateur à échantillonnage à sortie analogique, complété le cas échéant par un étage conformateur d'impulsions assurant la formation des signaux de sortie à modulation de largeur d'impulsions pour régulations à deux ou trois échelons avec organes de réglage proportionnels.

A.2.2 Description

En plus des fonctions traitant la consigne et la mesure, le FB réalise un régulateur PID prêt à l'emploi avec sortie continue de la grandeur de réglage et possibilité d'influencer à la main la valeur de réglage. Selon le type de CPU, il sera mis en œuvre grâce au FB41 (pour les CPU 3xx-2DP) ou au SFB41 (pour les CPU sans interface Profibus). Il propose les fonctions partielles suivantes :

- Branche de consigne.
- Branche de mesure.
- Formation du signal d'erreur.
- Algorithme PID.
- Traitement de la valeur de réglage manuelle.
- Traitement de la valeur de réglage.
- Action.

A.2.3 Anticipation États de fonctionnement

Démarrage et redémarrage.Le bloc FB41 *CONT-C* dispose d'un sous-programme de démarrage qui est exécuté quand le paramètre d'entrée COM-RST = TRUE. A la mise en route, l'intégrateur est positionné de façon interne sur la valeur d'initialisation I-ITVAL. En cas d'appel dans un niveau d'alarme d'horloge, il continue à travailler à partir de cette valeur. Toutes les autres sorties sont positionnées sur leurs valeurs par défaut.

A.2.4 Informations d'erreur

Le bloc ne procède pas à un contrôle interne d'erreur. Le mot indicateur d'erreur RET-VAL n'est pas employé.

A.2.5 Mise en œuvre d'un régulateur PID avec Step 7 grâce au (S)FB41 *CONT-C*

La programmation d'un SIMATIC S7-300 en tant que régulateur PID se fait avec le logiciel STEP 7, outil unique de programmation de tous vos types d'application. Ici n'est mentionné que l'essentiel, pour plus d'informations vous pouvez vous référer à la documentation spécifique à STEP7.

Le paramétrage du régulateur PID se fait grâce à l'outil : ≪ Paramétrage de la régulation PID ≫, les paramètres choisis seront sauvegardés dans le DB d'instance local associé à l'appel du (S)FB 41.

A.3 Le Logiciel CE2000

Le logiciel CE2000 est un logiciel de commande avec de nombreuses fonctionnalités. Il est livré en standard avec le CE117 process Trainer objet de notre étude. Le CE2000 permet à l'étudiants et aux ingénieurs de développer et de tester une large sélection de contrôleurs et de filtres. Le logiciel nous permet de créer un système de commande tout en combinant le design du contrôleur et l'implémentation dans un processus logique.

Pour créer un système sur notre logiciel, on utilise les icônes de logiciels et on les relient ensemble sur écran, tout comme l'élaboration d'un système de commande sur un morceau de papier. Les icônes sont les parties importantes des contrôleurs, des générateurs de signaux, des signaux contrôlés manuellement et les tensions, et des instruments virtuels.

Le CE2000 nous permet d'enregistrer d'importantes variables, tracer les résultats dans un tableau et exporter les données pour une utilisation dans d'autres programmes. On peut aussi créer un ou plusieurs types de contrôleurs et de simuler les réponses théoriques.

En raison de sa structure ouverte et flexible, le CE2000 peut également modèliser, simuler et exécuter tout autre système compatible.

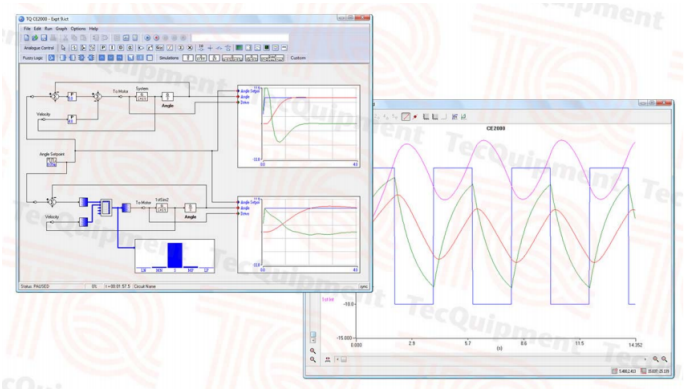


FIGURE A.1: Exemple de system de control sur le logiciel ce2000.

A.4 La fonction FC105

La conversion de la valeur numérique entière (plage nominale : entre 0 et 27648) de l'entrée analogique en valeur normée (normalisation) est réalisée par le bloc fonctionnel standard « Mise à l'échelle » (SCALE) FC105. La fonction FC 105 est fournie par STEP 7 dans la bibliothèque « Standard Library » dans le programme S7 « TI-S7 Converting Blocks », le schémas suivant montre son paramétrage

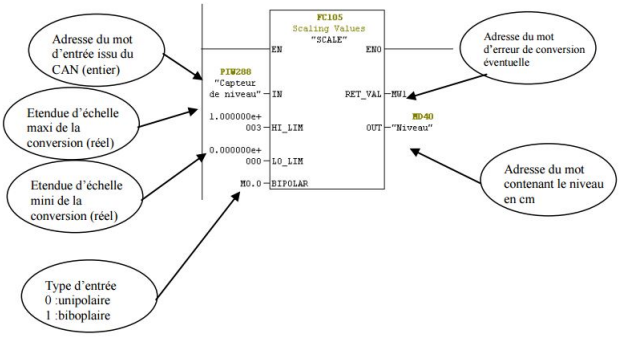


FIGURE A.2: La fonction scale.

A.5 La fonction FC106

De même le passage d'une valeur normée en une valeur numérique entière (entre 0 et 27648) pour la sortie analogique est réalisé par le bloc fonctionnel standard « Annuler la mise à l'échelle » (UNSCALE) FC106. La fonction FC 106 est fournie par STEP 7 dans la bibliothèque « Standard Library » dans le programme S7 « TI-S7 Converting Blocks », le schémas suivant montre son paramétrage

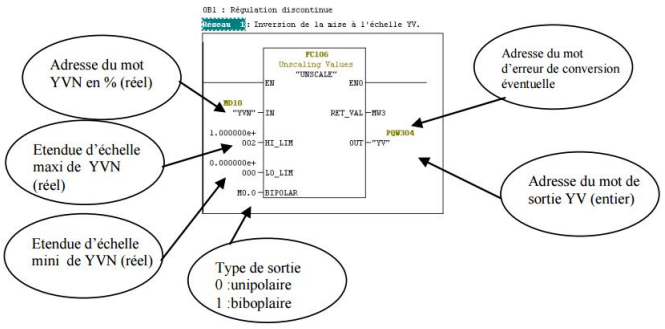


FIGURE A.3: La fonction nscale.

Bibliographie

[1] *GILLES MICHEL Architecture et application des automates programmable, DUNOD, Paris 1988.*

[2] *Jack HUGH Automating Manufacturing Systems with PLC's. 2005.*

[3] *Michel BERTRAND Automates programmables Industriels. Ecole Nationale superieure d'Arts et Métiers ENSAM, Centre d'Enseignement et de Recherche de Lille.*

[4] *Siemens,S7-PLCSIM. version 5.4, SIMATIC, 07/2011.*

[5] *www.sitelec.org, consulté en Juillet 2015.*

[6] A.ABRICHE. *Réalisation et gestion d'un prototype de station de pompage à base d'automates programmables industriels SIEMENS*, juin 2007.

[7] P.JARGOT. Langages de programmation pour api. norme iec 1131-3 vol. s 8030. *Techniques de l'ingénieur.*

[8] Siemens. *Techniques de régulation avec STEP7 :Document de formation pour une solution complète d'automatisation Totally Integrated Automation (TIA)*, juin 2005.

[9] Siemens. *Programmer avec STEP 7*, Mai 2010.