République Algérienne Démocratique et Populaire

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE ABOU BEKR BELKAID DE TLEMCEN
FACULTE DE TECHNOLOGIE
DEPARTEMENT DE GENIE ELECTRIQUE ET ELECTRONIQUE

MEMOIRE

Présenté pour l'obtention du diplôme

Master en Electrotechnique

Spécialité : Commandes Electriques

Intitulé :

Régulation de Position d'un MCC via un API S7-300 destiné pour un module de Stockage / Destockage de Marque Festo

Présenté par :

Mr. ZAHRAOUI ISSAM EDDINE

Mr. SAIDI FAYSSAL

Soutenu le 30 Juin 2021 devant le jury :

Mr. BENHABIB Mohamed Choukri	**MCA**	**Président**
Mr. BRIXI NIGASSA Mohammed El Amine		**Examinateur**
Mr. MELIANI Sidi Mohammed.	**Pr**	**Encadreur**

Universitaire : 2020– 2021

Dédicace

Je dédie ce mémoire A mes chers parents ma mère

Et mon père Pour leur patience, leur amour,

Leur soutien et leurs encouragements. A mes frères.

A mes amis et mes camarades. Sans oublier tout

Les professeurs que ce soit du primaire, du moyen,

Du secondaire ou de l'enseignement supérieur.

ZAHRAOUI ISSAM EDDINE

Tlemcen, juin 2021

Dédicace

Je dédie ce travail à :

Mes chers parents pour leurs amours,

Leurs encouragements, leurs soutiens et leurs

Sacrifices durant toutes mes années d'études

Dont les mots sont insuffisants pour exprimer

Ma reconnaissance et mon profond amour,

Dieu leurs donne une longue et parfaite vie.

- mes frères.
- Ma sœur.
- Mes amis sans exception.
- Tous ceux que j'aime et qui m'aiment.

SAIDI FAYSSAL

Tlemcen, juin 2021

Remerciement

En premier lieu, nous tenons à manifester notre louange à Dieu par aisance et excellence, veuille-t-il-nous guider toujours dans le droit chemin.

Au terme de ce modeste travail, nous tenons à exprimer nos vifs remercîments à notre encadreur Mr. MELIANI Sidi Mohamed pour sa disponibilité, son aide, son orientation et ses conseils précieux. Nos remercîments s'adressent également aux membres du jury Mr BENHABIB Mohamed Choukri et Mr BRIXI NIGASSA Mohammed El Amine pour avoir eu le temps d'expertiser ce modeste travail. Nos remerciements s'adressent aussi à tous les enseignants de l'université Abou Bekr Belkaid de Tlemcen, Faculté de Technologie, et particulièrement ceux du département de Génie Electrique et Electronique, pour les efforts qu'ils ont déployé durant notre formation.

Nous remercions évidement nos familles, nos amis pour leur irremplaçable et inconditionnel soutien. Ils ont été présents pour écarter les doutes, soigner les blessures et partager les joies.

ZAHRAOUI ISSAM EDDINE

Et SAIDI FAYSSAL

Tlemcen, juin 2021

Table des matières

Listes des figures

Liste des Abréviation

U Tension du réseau [V]

i_a Courant de l'induit [A]

C Couple moteur [Nm].

Cr : Couple resistant

Cem : Couple électromagnétique

E Force contre électromotrice [V]

R_a Résistance de l'induit [Ω]

L Inductance de l'induit [H]

K Gain [N.m/I]

Ω Vitesse angulaire [tr/s]

J L'inertie [kg m^2]

Introduction Générale

La technologie moderne a permis le développement des sciences tout en imposant l'exploration de domaines théoriques de plus en plus complexes. Parmi ces sciences en pleine expansion et intégrant rapidement l'apport des technologies modernes, on compte l'automatique. Le substantif « automatique » a été utilisé pour la première fois en 1914 dans un article « Essai sur l'Automatique » publié dans une revue scientifique.

De nos jours, l'automatique fait partie des sciences de l'ingénieur. Cette discipline traite de la modélisation, de l'analyse, de la commande et de la régulation des systèmes dynamiques. [35]

Le but de notre travail est de faire l'étude de régulation de position du moteur à courant continu et relie avec API S7-300 est superviseur HMI. [24]

Ce mémoire est divisé en trois chapitres.

Dans le premier chapitre, nous allons présenter d'une manière générale ce qu'est la régulation industrielle de la position.

Dans le deuxième chapitre, nous allons faire le principe détaillé de la régulation et nous Allos jeter un œil à TIA portal et Présentation Matériel Et Logiciel et expliqué les deux partie (hardware et software).

Enfin, le troisième chapitre, nous allons vous présenter en détail le développement d'un simulateur virtuel qui va nous permettre de faire la régulation de position d'un MCC sans utilisation du matériel.33

Enfin notre travail se termine par une conclusion générale.[13]

Chapitre I

Généralité sur la régulation

De position d’un MCC

I.1 Introduction :

Dans ce premier chapitre qui s'articule autour de Cinq parties principales, nous nous proposons en premier lieu d'introduire notre travail en établissant le contexte général du sujet.

La régulation de position est plus importante dans l'industrie et appliquer dans plusieurs domaines comme (drones, lecteur CD, l'ascenseur, Robot, Vérin pour parabole, antenne), Pour contrôler et avoir une grande précision sur la position il faut opter pour une technique de commande et dans notre projet on va utiliser un API S7-300 qui permet faire cette régulation.

Les moteurs électriques sont aujourd'hui présents dans toutes les branches de l'industrie L'intérêt grandissant envers les moteurs électriques est justifié par le besoin des processus industriels à la position et la vitesse variable, il est très utilisé en industrie est nécessite une régulation précise de la position de rotation avec le réducteur sert à réduire la vitesse d'un moteur avec transmission de la puissance motrice vers une machine réceptrice en absorbant le moins d'énergie.

Le pont H est utilisé pour l'entrainement d'une machine à courant continu dans un sens de rotation pour le fonctionnement en moteur avec freinage par récupération dans l'autre sens de rotation

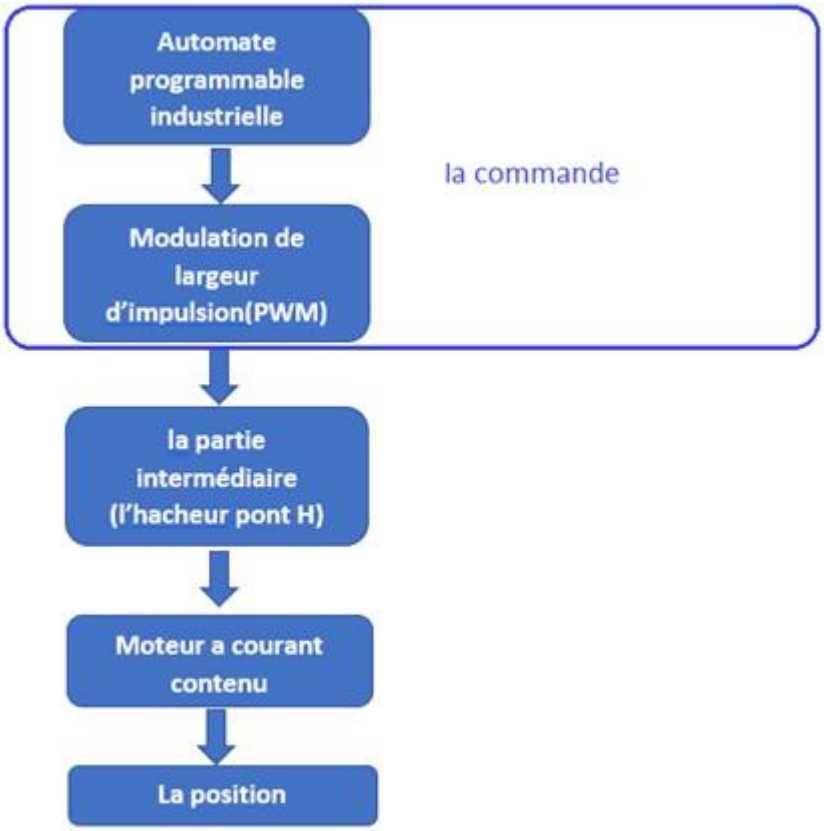


Figure I. 1 : schéma bloc de système de la commande de position

I.2 Exemples des processus utilisant la régulation de position :

I.2.1 Drone :

Les drones (du mot anglais signifiant faux bourdon1) sont des aéronefs sans équipage dont le pilotage est automatique ou télécommandé, à usage civil ou au profit des forces armées ou de sécurité — police, douane — d'un État. En fonction des capacités recherchées, leur masse varie de quelques grammes à plusieurs tonnes. Leur autonomie peut atteindre jusqu'à plusieurs dizaines d'heures (à comparer aux deux heures typiques d'autonomie d'un chasseur).[29]

Figure I. 2 : drone [29]

I.2.2 Lecteur de CD :

Le lecteur de CD (appelé communément « lecteur CD ») est un lecteur de disque optique qui lit au moyen d'une diode laser les disques optiques appelés disques compacts ou CD, qu'il s'agisse de CD audio ou de CD-ROM informatiques.[30]

Figure I. 3 lecteur CD [30]

I.2.3 Ascenseur :

Un ascenseur est un transport vertical assurant le déplacement en hauteur. Les dimensions, la construction et le contrôle en temps réel pendant l'usage des ascenseurs permettent l'accès sécurisé des personnes. L'ensemble du dispositif des guides, moteur, mécanique et câbles est installé le plus souvent dans une trémie ou gaine rectangulaire verticale

fermée ou parfois semi-fermée située en général à l'intérieur de l'édifice, dans laquelle la cabine et le contrepoids gravitent.[31]

Figure I. 4 : ascenseur [31]

I.2.4 Robot :

Un robot est un dispositif mécatronique (alliant mécanique, électronique et informatique) conçu pour accomplir automatiquement des tâches imitant ou reproduisant, dans un domaine précis, des actions humaines. La conception de ces systèmes est l'objet d'une discipline scientifique, branche de l'automatisme nommé robotique.[32]

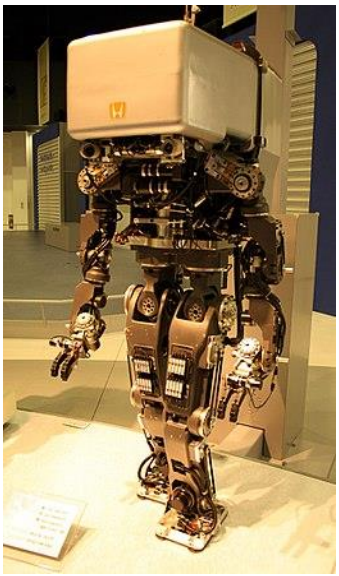


Figure I. 5 : Robot [32]

I.2.5 Imprimante :

Une imprimante est un engin permettant d'obtenir un document sur papier à partir d'un modèle informatique du document. Par exemple, un texte écrit via un logiciel de traitement de texte sur ordinateur pourra être imprimé pour en obtenir une version papier (c'est un changement du support d'information). Les imprimantes ont été conçues dès l'apparition des premiers ordinateurs, pour permettre la consultation et la conservation sur support papier des résultats produits par les programmes informatiques. En effet, à l'époque des premiers calculateurs, les

écrans n’existaient pas encore et les méthodes de stockage de l’information étaient très rudimentaires et très coûteuses.[33]

Figure I. 6 : Imprimante [33]

I.2.6 Bras manipulateur :

Un bras manipulateur est le bras d'un robot généralement programmable, avec des fonctions similaires à un bras humain. Les liens de ce manipulateur sont reliés par des axes permettant, soit de mouvement de rotation (comme dans un robot articulé) et/ou de translation (linéaire) de déplacement.[34]

Figure I. 7 : manipulateur [4]

I.3 Principe de la régulation de position :

Pour certain le terme régulation s’applique lorsque que le but est constant et ils parlent d’asservissement lorsque ce but à atteindre varie dans le temps. Ce n’est pas très important, en fait et pour nous ce sera des synonymes. D’autres parlent de pilotage pour l’ensemble des entités en interaction, pour nous ce terme « pilotage » sera restreint à la fonction du conducteur dans un véhicule par exemple.

Mais ce qui nous intéresse ici, ce sont les « règles » de la régulation et les fonctions des mouvements plus que la machine elle-même.

Pour ce faire, voici un système qui va nous permettre de visualiser un déplacement ; considérons un petit chariot qui se déplace lorsque le moteur tourne. Notre ami, avec son drapeau vert, nous indique si l'on est loin ou près du mur, et suivant ses indications, on appuie ou on relâche la pédale des gaz ! Le but est de stopper notre chariot « juste au mur ».

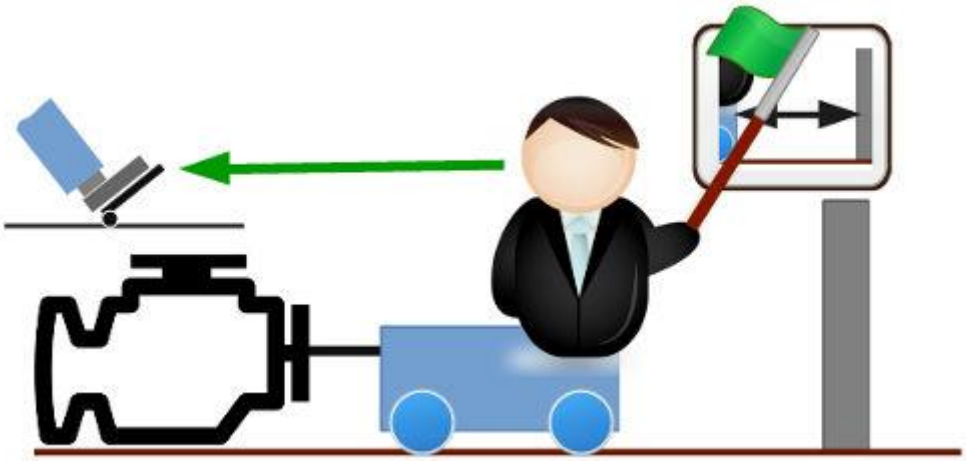


Figure I. 8 : système de visualisation un déplacement

Pour arriver à s'arrêter « contre le mur » sans abîmer notre chariot, il est nécessaire d'avoir un retour de la distance entre le chariot et le mur. Ici on doit de suite moduler ce propos, certaines machines ne fonctionnent pas selon ce principe, nous y reviendrais un peu plus loin.

Comme vous régulez l'avance de votre chariot en fonction de la distance restante, c'est donc une régulation de position dont il s'agit.

Pour piloter correctement notre chariot, nous devons bien connaître divers éléments. Par exemple, si vous êtes à 1, 10 ou 100 mètres du mur vous n'aurez pas la même vitesse maximum lors de votre déplacement. Généralement (mais là c'est vous qui voyez !) plus vous vous approchez du mur, plus diminuez votre vitesse, jusqu'à avoir une vitesse nulle au moment du contact avec le mur. Les paramètres que vous gérez lors de ce déplacement sont principalement :

-La distance à parcourir jusqu'au mur

-La vitesse et l'accélération maximale admissible (surtout par votre ami !)

-Le mode d'arrivée

-Que signifie ce paramètre « mode d'arrivée » ? Lorsque vous vous déplacez avec votre véhicule, vous avez une succession de distance à parcourir entre chaque arrêt, jusqu'au prochain stop, par exemple. Habituellement vous admettez votre prochain point d'arrivée sur la « ligne du stop ». Si vous êtes un peu avant ou un peu après, ce n'est pas important, car vous avez

toujours la possibilité de corriger votre position après votre arrivée. Autre exemple ou le dépassement de la consigne (point d'arrivée) est admis : le réglage de la température de l'eau de votre douche. Un peu chaud, un peu froid, « re » un peu chaud, « re » un peu froid, etc, et enfin la douche est agréable ! Dans ces « modes d'arrivée », on admet que la position à atteindre (la consigne) peut être dépassée. Mas vous comprendrez bien que pour notre chariot et le mur, il est impératif de ne pas dépasser la distance à parcourir, donc que le point d'arrivée (la position à atteindre, la consigne) est à ne jamais excéder ! Dans le cas d'axes de machines-outils, c'est identique, le dépassement de consigne est prohibé !

I.3.1 Régulation de position d'un mcc :

Schématiquement, la régulation de position d'axe machine présente cet aspect, notez que c'est général :

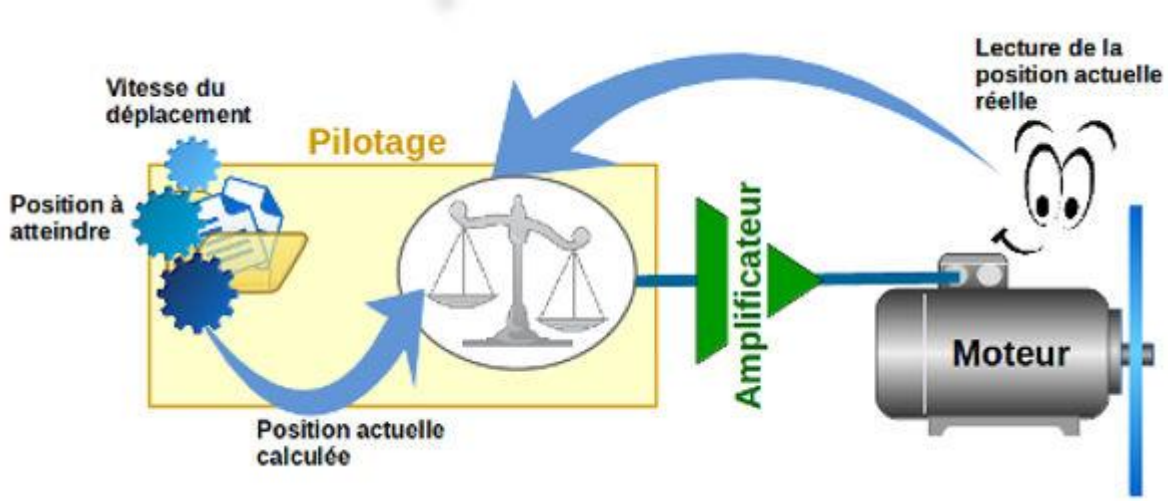


Figure I. 9**:** schéma de la régulation d'un axe **[3]**

Ce schéma, qui résume la régulation d'un axe de machine, demande quelques explications.

On a séparé la régulation en quatre entités principales (pilotage, amplificateur, moteur, lecture position), qui sont les fonctions clés de la régulation de position des axes.

D'abord un côté pratique, la lecture de la position actuelle réelle, telle que dessinée, suggère que l'on ne contrôle pas directement la position du chariot, mais que l'on compte le nombre de tours moteur+. C'est une méthode possible, car par déduction (pas de la vis) on trouve de la position du chariot. Ce n'est qu'une solution parmi beaucoup d'autres pour suivre la position du chariot.

Si l'on veut visualiser le passage de la consigne au déplacement de l'axe, on peut le résumer comme suit

Si l'on veut visualiser le passage de la consigne au déplacement de l'axe, on peut le résumer comme suit :

Consigne → pilotage → amplificateur → moteur → position

Sur ce montage photos vous avez tout de suite repéré la photo qui représente notre ami avec le drapeau, celle « du retour d'information » (position réelle). Et enlevez le casque on vous a reconnu !

Figure I. 10 : Schéma de la régulation d'un axe sur mode plus technique [3**]**

Le même schéma sur un mode plus technique (c'est juste pour faire peur) :

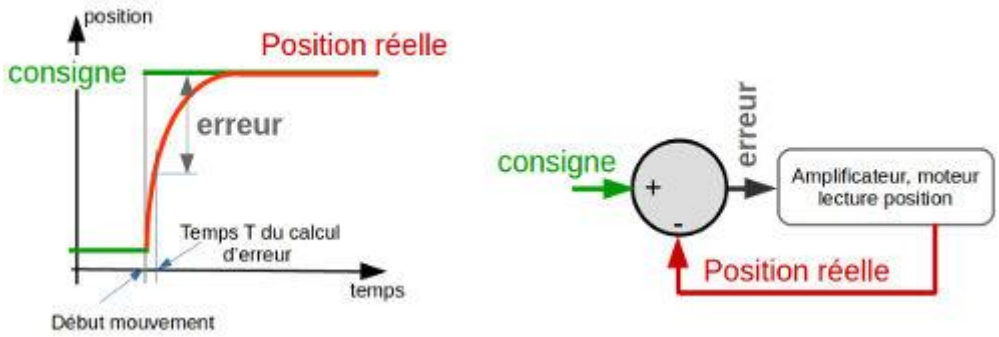


Figure I. 11 : schéma d'erreur entre consigne et position réelle **[3]**

C'est dans les grandes lignes le principe des régulations de machines. Dans la réalité, pour les machines-outils, on contrôle, aussi lire « retour d'information », d'autres paramètres (la vitesse, le couple, l'accélération, le courant, etc.), il n'y a pas seulement la position du chariot. Mais le but est toujours le même : essayer de suivre au plus près la consigne du système de pilotage, l'atteindre le plus vite, le plus régulièrement et surtout NE PAS LA DÉPASSER. Si l'on se réfère au schéma ci-dessus : toujours minimiser au maximum l'erreur !

Certaines machines, robots ou axes de machine ne sont pas régulés en position mais en force (ou couple) ou encore en vitesse mais nous ne considérons pas ces types de régulation. Bien que le principe soit un peu similaire, un paramètre principal du mouvement est particulièrement contrôlé et le plus suivi. Notez encore que dans certains cas il n'y a pas de déplacement principal, car le but est l'application d'une force ou d'une pression par exemple.

I.3.2 Pilotage :

Son rôle principal est de définir la position de l'axe en tout moment et d'envoyé une consigne (une erreur) « accélérer ou ralentir » à l'amplificateur suivant la position réelle du chariot.

I.3.3 Mouvement axe machine

Cette régulation se résume à un déplacement du chariot selon le souhait d'un utilisateur. Le système de régulation essaye donc de répondre à cette demande. On peut synthétiser le mouvement du chariot comme suit :

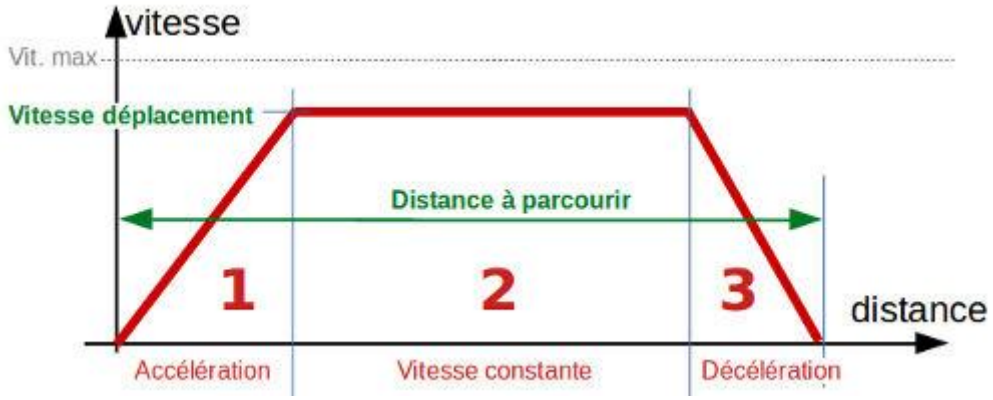


Figure I. 12 : schéma de la variation de vitesse para port la distance[3]

Sur ce graphique, on remarque tout de suite qu'en fait deux éléments sont surveillés, la vitesse et la distance, le souhait est concrètement un déplacement à une certaine vitesse. On parle bien de régulation de position, mais en fait, c'est une régulation de vitesse et de position qui est utilisée. Il faut cependant considérer une prédominance de la consigne de position sur celle de la vitesse et ce fait justifie entièrement de ne parler que de régulation de position. En effet pour les mouvements des axes machines, on ne cherche pas systématiquement la vitesse maximum des axes, c'est pourquoi on fixe une vitesse pour chaque déplacement donné. En fait, on essaye, espère de réaliser un certain travail lors des déplacements d'axes, donc la maîtrise de la vitesse est assez compréhensible.

Le déplacement total est réalisé en 3 phases, l'accélération (1), la vitesse constante (2) et le freinage (3) ou aussi dénommé décélération ou encore l'accélération négative. L'utilisateur fixe deux paramètres : la distance et la vitesse, le système de régulation, lui, calcule la position du chariot en fonction des phases du déplacement.

On analysera que le cas général ou le déplacement demandé permet d'atteindre la vitesse souhaitée. Ce n'est pas réducteur, car vous pouvez très facilement imaginer que le paramètre « distance » est plus important à respecter que celui de la vitesse qui lui sera redéfini. Autre point, dans le parcours présenté la vitesse finale est nulle, mais ce n'est pas toujours le cas. Là aussi le but sera redéfini avec le but « vitesse à cette distance ». Le principe global restant le même.[3]

I.4 Moteur a courant contenu

I.4.1 Définition d'un MCC :

Les moteurs à aimants permanents comportent des aimants permanents plutôt que des enroulements inducteurs pour produire le champ magnétique du stator. La figure I. 13 représente le schéma d'un tel moteur.

Ces aimants assurent une intensité de champ constante, ce qui amène des caractéristiques similaires à celles des moteurs à excitation en dérivation.

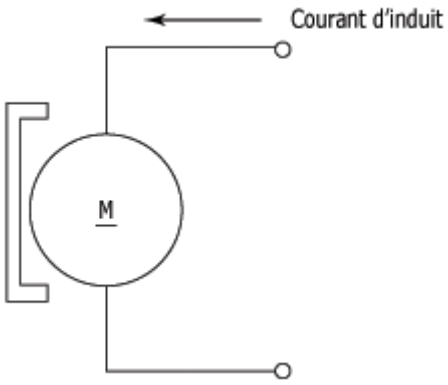


Figure I. 13 : Schéma d'un moteur à aimants permanents [15]

On se sert des moteurs à aimants permanents pour des applications de faible et moyenne puissance, en particulier pour les appareils alimentés par pile. Il est également très utilisé en robotique. [15]

I.4.2 Modèle électrique :

1)Equations Electromécanique du moteur à courant continu en régime dynamique :

On a donc deux relations de proportionnalité entre la f.é.m. E et la vitesse du rotor :

$$\Rightarrow E = K_e * \Omega\,(\mathrm{t}) \qquad (1.1)$$

Et un moment du couple électromagnétique directement proportionnel au courant d'induit :

$$\mathrm{Cem(t)} = K_m * I_a(t) \qquad (1.2)$$

a) Equations électriques : La tension d'induit (en convention récepteur) :

$$U(t) = R_a . i_a(t) + L_a \frac{di_a}{dt}(t) + e(t) \quad (1.3)$$

b) Equation mécaniques :

Le principe fondamental de la dynamique (PFD)nous permet d'écrire

$$J\frac{d\Omega}{dt} = Cu - Cr \text{ avec } Cu = Cem - Cp \quad (1.4)$$

On suppose que le moment du couple de perte est de la forme : Cp = f. Ω

f: coefficient de frottement visqueux

$$J\frac{d\Omega}{dt} = Cem - f\Omega - Cr \quad (1.5)$$

Pour une régulation de position, la plupart du temps Cr(t) très faible ou négligeable mais cst.

Sachant que $\mathrm{Cem(t)} = K_m * I_a$ avec $K_m = K_e \Longrightarrow \mathrm{Cem(t)} = K_e * I_a$

$$J\frac{d\Omega(\mathrm{t})}{dt} = Cem - f\Omega(\mathrm{t}) \quad (1.6)$$

I.5 Réducteur :

I.5.1 Présentation :

Les réducteurs sont des composants importants de la chaîne d'énergie, ils ont pour rôle de modifier les caractéristiques du mouvement de rotation produit par un moteur, afin de l'adapter au mieux aux demandes du récepteur.

Le réducteur est caractérisé par son rapport de transmission : R

R < 1 nous avons à faire à un réducteur

R > 1 nous avons à faire à un multiplicateur de vitesse $\mathrm{R} = \frac{\Omega \text{ sortie}}{\Omega \text{ entrée}}$

Souvent, le réducteur est réversible, c'est à dire que la sortie peut jouer le rôle d'entrée et réciproquement.

I.5.2 Place du réducteur dans la chaine d'énergie :

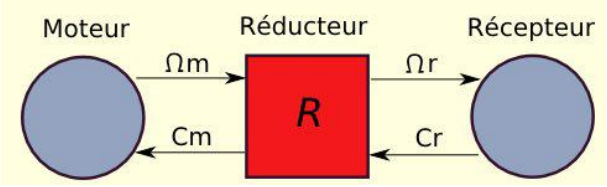


Figure I. 14 : Schéma du moteur avec réducteur [18]

Le réducteur s'interpose entre le moteur et le récepteur. Il modifie les deux valeurs caractéristiques du mouvement de rotation :

- La vitesse de rotation (Ω)
- Le couple transmis (C)
- Prise en compte du rendement du réducteur :
  - La puissance de sortie est donnée par : Ps = Cr. Ωr-
  - La puissance d'entrée est donnée par: Pe = Cm. Ωm
  - Le rendement est donné par: η = Ps / Pe = (Cr. Ωr) / (Cm. Ωm)
- Ωr = R. Ωm (par définition)
- η Cm = R Cr

On peut noter que le rendement n'intervient pas sur la vitesse de rotation et qu'un réducteur de vitesse est un multiplicateur de couple.[18].

I.6 Pont en h (hacheur) :

I.6.1 Principe :

Hacheur alimente le bobinage disposé sur l'induit mobile (rotor). Ce bobinage est placé dans un champ magnétique, permanent ou non, produit par l'inducteur (stator), notamment lorsque l'inducteur est constitué d'aimants. Le courant circulant dans les spires de l'induit du moteur, des forces électriques lui sont appliquées et, grâce un dispositif adapté (balais et collecteur), les forces s'additionnent pour participer à la rotation. Pour modifier le sens de rotation d'un moteur à courant continu, il suffit d'inverser l'alimentation à ses bornes (induit). La structure permettant de réaliser cette inversion est appelée « pont en H ».

I.6.2 Pont en H :

Lorsqu'on veut commander le sens de rotation d'un moteur (à courant continu) on est souvent obligé d'inverser la polarité. De plus il est généralement préférable de pouvoir faire varier la vitesse du moteur. La solution est d'utiliser le pont en H. Lorsqu'on arrête le moteur, et qu'il continue à tourner avec l'inertie, il se comporte comme une génératrice. Pour éviter d'avoir des courants dans les transistors on monte des diodes de roues libres. [17]

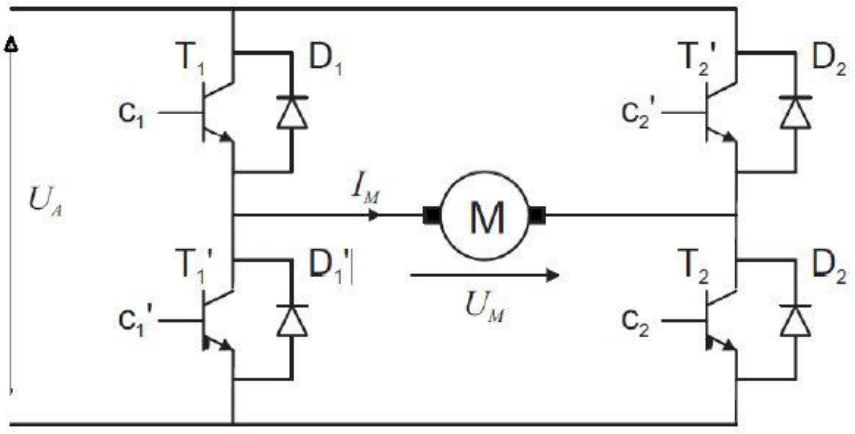


Figure I. 15 : Schéma d'un circuit de pont en H [13]

I.7 MLI (PWM) :

I.7.1 Stratégies de commande :

La fonction MLI (Modulation en Largeur d'Impulsion) ou PWM (Pulse Width Modulation) est une technique de pilotage pour les convertisseurs statiques utilisée Pour la conversion de l'énergie, ayant ses bases dans le domaine des Télécommunications (traitement du signal).

C'est une manière simple et efficace de générer une tension analogique avec un microcontrôleur. Peu d'entre eux sont en effet équipés d'un convertisseur numérique analogique. Les niveaux de tensions générés par le microcontrôleur sont celles de son alimentation (en général 0 ~ 5V), mais la fréquence et le rapport cycliques sont configurables/variables.

Loin d'être un élément accessoire dans la chaîne de variation de vitesse (variateur électrique associé à une machine électrique), l'étage MLI joue le rôle d'interface entre la partie commande d'un variateur de vitesse et la machine électrique associée. Cette commande joue un rôle essentiel avec des conséquences sur toutes les performances du système.

I.7.2 Schéma de Principe :

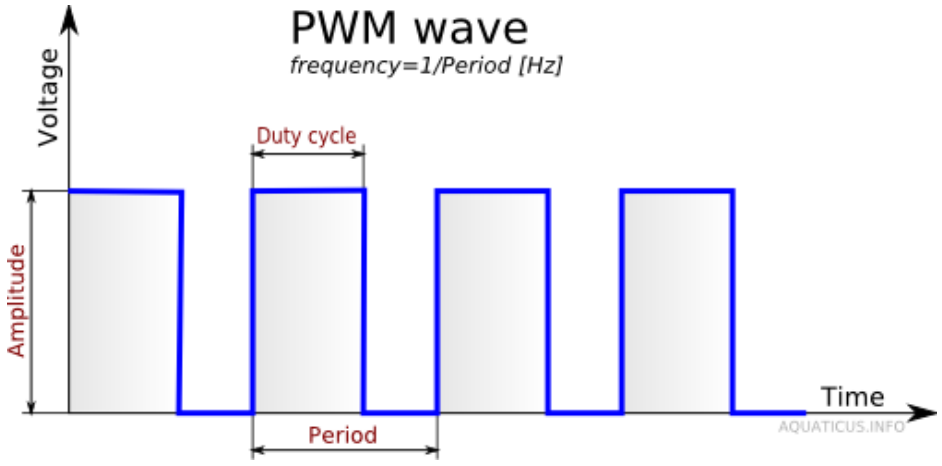


Figure I. 16 : Schéma de principe de la commande nature **[17]**

La variation de vitesse Ω d'un moteur à courant continu est obtenue en faisant varier la valeur moyenne de la tension d'alimentation de l'induit U. Cette dernière est obtenue en appliquant une tension rectangulaire d'amplitude Û constante et de rapport cyclique α=th / T variable. Dans ce cas, la valeur moyenne <U>=Umoy* α [17]

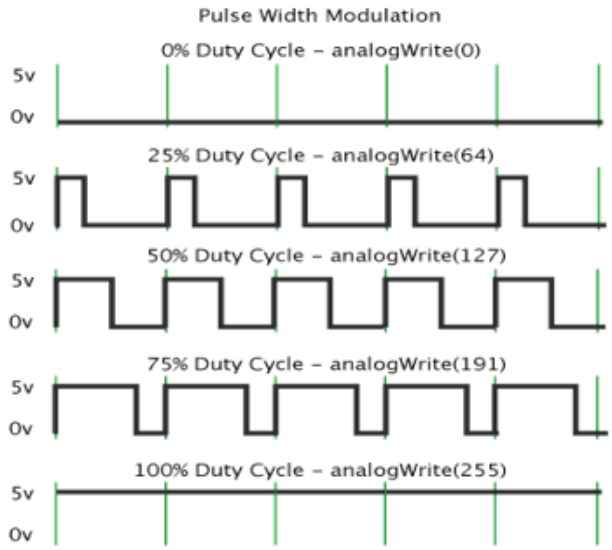


Figure I. 17 : Schéma de variation de duty cycle [17]

I.8 Organe de commande API :

I.8.1 Définition d'un Automate Programmable :

L'automate programmable est un produit de technologie électronique dont le fonctionnement est défini par un programme ; c'est un organe de traitement de l'information comme toutes les familles de constituants pneumatiques à relais ou électronique. [20]

I.8.2 La définition de Win CC :

WinCC (**Win**dows **C**ontrol **C**enter) flexible est un système IHM (**I**nterface- **H**omme-**M**achine) très performant développé par SIEMENS. C'est un outil flexible qui s'intègre parfaitement dans les solutions d'automatisation et de techniques de l'information et qui est destiné à la configuration des systèmes de supervision.

WinCC flexible permet la saisie, l'affichage et l'archivage des données tout en facilitant les tâchesde conduite et de surveillance aux exploitants. Il est compatible avec Windows et comporte des objets graphiques prédéfinis tels que : Affichage numérique, bibliothèque complète de symboles IHM, affichage de texte et courbes, champs d'édition de valeurs du processus…etc.[28]

I.9 Conclusion :

Dans ce chapitre nous avons vu la notion de système et leur performance, ensuite ont vu la définition de système et les exemples de régulateurs industriels, poursuite on définir la modélisation d'un mcc et on a parlé sur le pré-actionneur (pont h).

Ce système est contrôlé par le signal PWM qui est envoyé par l'API

Chapitre II

Présentation Matériel Et Logiciel pour la régulation industrielle

II.1 Introduction :

Dans ce chapitre nous allons voir le principe détaillé de la régulation de position pi qui sera avec l'organe de commande qui l'API, au niveau de l'API en va ajouter un objet technologie qui est le PID.

On applique les équations électromécaniques dans le domaine de Laplace pour faire Schéma fonctionnel du moteur à courant continu

On va faire l'étude sur la régulation en tension d'un moteur à courant continu

II.2 Principe de la régulation :

Le principe de fonctionnement d'une régulation est le suivant. Pour réguler un système physique, il faut :

Mesurer la grandeur réglée avec un capteur.

Analyser : c'est la fonction du régulateur. Le régulateur compare la grandeur réglée avec la consigne et élabore le signal de commande.

Agir sur la grandeur réglant par l'intermédiaire d'un organe de réglage.

Structure de principe d'un régulateur :

Le régulateur compare la mesure et la consigne pour générer le signal de commande.

Le signal de mesure est l'image de la grandeur réglée, provenant d'un capteur et transmetteur et transmise sous forme d'un signal électrique ou pneumatique.

La consigne peut être interne (fournie en local par l'opérateur) ou externe (via une supervision).

L'affichage de la commande se fait en % et généralement en unités physiques pour la consigne et la mesure. [22.]

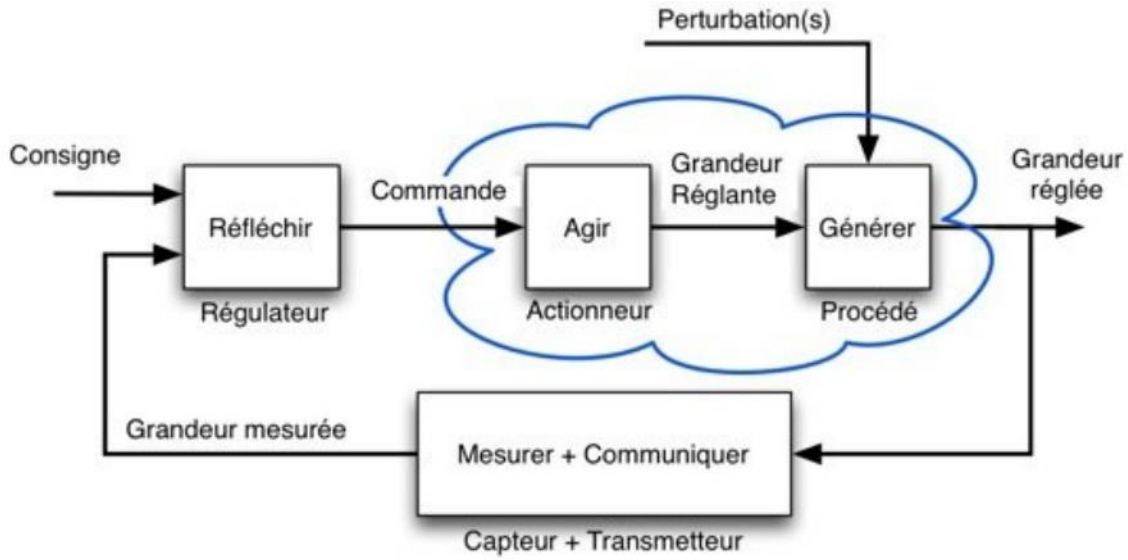


Figure II. 18 : schéma de principe de la régulation (la boucle de régulation) [21]

Capteur c'est un dispositif qui capte un phénomène physique et le restitue sous forme signal en utilise de position

Régulateur :il sert à maintenir la régularité de fonctionnement d'un mécanisme

Actionneur qui va capteur transforme l'Energie qui lui est transmise en un travail

II.3 Equations électromécaniques dans le domaine de Laplace :

D'après les équations des moteurs de chapitre 1

La transformée de Laplace de l'équation (1.3) est **:**

$$U(p) = R_a * I_a(p) + p * L_a * I_a(p) + K_e * \Omega(\mathrm{p}) \qquad (2.1)$$

La transformée de Laplace de l'équation (1.1) Est :

$$E(p) = K_e * \Omega(p) \qquad \text{(gain2)} \qquad (2.2)$$

La transformée de Laplace de l'équation (1.4) Est :

$$Jp\Omega(p) = K_m * I_a(p) - f * \Omega(p) - Cr(p) \qquad (2.3)$$

Soit : $$\Omega(p) = \frac{K_m * I_a(p) - Cr(p)}{J * p + f} \qquad (2.4)$$

Fonction de transfert du moteur :

On suppose que le moment du couple de pertes (qui est vu comme une perturbation) est négligeable devant le moment du couple électromagnétique ce qui donne :

$$\Omega(p) = \frac{K_m * I_a(p)}{J * p + f} \qquad (2.5)$$

Le courant I devient : $$I_a = \frac{(J * p + f)\Omega(p)}{K_m} \qquad (2.6)$$

Et en remplaçant cette nouvelle expression de I(p) dans l'équation (2.1)

On obtient :

$$U(p) = R_a * \frac{(J * p + f)\Omega(p)}{K_e} + L_a \frac{(J * p + f) * \Omega(p)}{K_e} p(p) + K_e\Omega$$

$$U = \frac{R_a * J * \Omega * p + R_a * f * \Omega + L_a * J * \Omega * p^2 + L_a f * \Omega * p + {K_e}^2\Omega}{K_e}$$

$$\frac{U}{\Omega} = \frac{L_a * J * p^2 + (R_a * J + L_a * f) * p + R_a * f + {K_e}^2}{K_e} \qquad (2.7)$$

On peut maintenant exprimer la fonction de transfert en boucle fermée la vitesse de sortie par rapport à la tension d'entrée :

$$G_{u\omega}(p) = \frac{\Omega(\mathrm{p})}{U(p)} = \frac{K_e}{L_a J * p^2 + (R_a * J + L_a * f) * p + R_a * f + {K_e}^2} \qquad (2.8)$$

On peut écrire aussi sous la forme canonique d'une fonction de transfert de second ordre :

$$G_{u\omega}(p) = \frac{\Omega(\mathrm{p})}{U(p)} = \frac{K_o}{\tau * \tau e * p^2 + (\tau + \alpha * \tau e) * p + 1} \qquad (2.9)$$

Avec : $\tau e = \frac{L_a}{R_a}$ et $\tau = \frac{R_a J}{R_a f + {K_e}^2}$ et $\alpha = \frac{R_a f}{R_a f + {K_e}^2}$ on peut négliger α car $K_e I_a \gg f\Omega$

Idem pour E : $E = K_e * I_a \gg R_a * I_a$

On aura une nouvelle écriture de la fonction de transfert :

$$G_{u\omega}(p) = \frac{\Omega}{U} = \frac{K_o}{\tau * \tau e * p^2 + \tau * p + 1} \qquad (2.10)$$

Et si l'on a $\boldsymbol{\tau e \ll \tau}$, c'est souvent le cas : la constante de temps électrique est négligeable devant la constante de temps électromécanique, on peut alors réécrire une nouvelle fois la fonction de transfert en factorisant son dénominateur : ($\boldsymbol{\tau + \tau e \# \tau}$ on rajoute une quantité négligeable ($\boldsymbol{\tau e}$ à $\boldsymbol{\tau}$)

$$G_{u\omega}(p) = \frac{\Omega}{U} = \frac{K_o}{\tau * \tau e * p^2 + (\tau + \tau e) * p + 1} \qquad (2.11)$$

Et on factorise à nouveau :

$$G_{u\omega}(p) = \frac{\Omega}{U} = \frac{K_o}{(1 + \tau e * p)(1 + \tau * p)} \qquad (2.12)$$

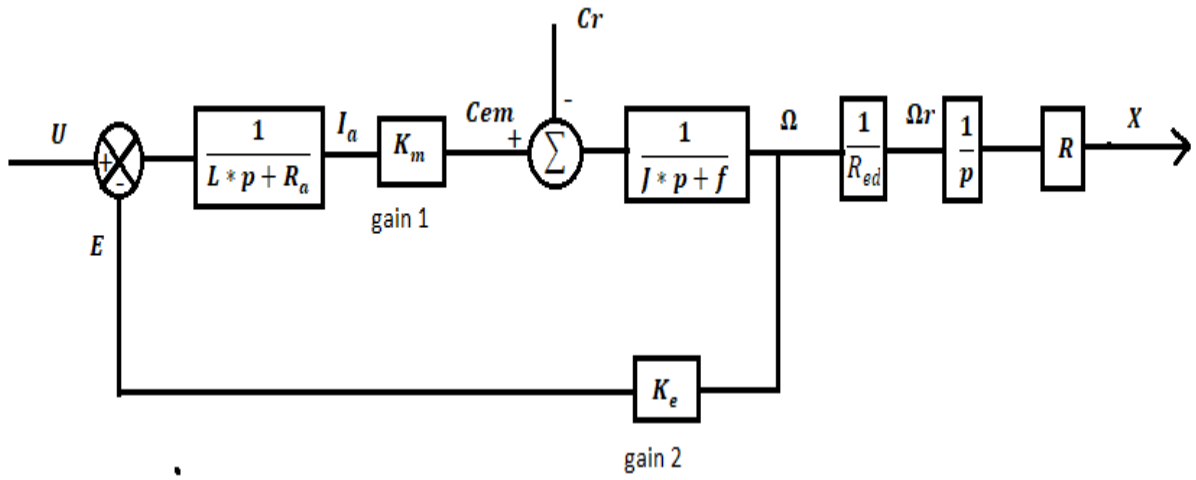
Explication :

D'après l'équation (2.1)

$$I_a = \frac{U - E}{L_a * p + R_a} \qquad (2.13)$$

La transformée de Laplace de l'équation (1.2) est :

$$Cem = K_e * I_a(\mathrm{p}) \qquad (\mathrm{gain1})$$



D'après l'équation (1.4)

$$J * p * \Omega(\mathrm{p}) = Cem - f\Omega - Cr \Rightarrow Cem - Cr = \Omega(\mathrm{p})(J * p + f)$$

$$\Omega(\mathrm{p}) = \frac{Cem - Cr}{J * p + f} \tag{2.14}$$

$$E(p) = K_e * \Omega(\mathrm{p})$$

Sachant que $$\Omega r(p) = \frac{1}{R_{ed}} * \Omega(p) \tag{2.15}$$

Pour régler la position en intégrer la vitesse angulaire :

$$v(t) = \frac{dx(t)}{dt} = R * \Omega(\mathrm{t}) \Longrightarrow V(t)_{red} = R * \frac{1}{R_{ed}} * \Omega(t)$$

$$x_{red}(t) = \int v_{red}(t) * dt \tag{2.16}$$

Figure II. 19 : Schéma fonctionnel du moteur à courant continu avec réducteur

La transformée de Laplace de l'équation (2.16)

$$X_{Red}(p) = \frac{1}{p} * R * \Omega r(p) = \frac{1}{p} * R * \frac{1}{R_{ed}} * \Omega(p) \tag{2.17}$$

II.4 Régulation du moteur à courant continu :

II.4.1 Pont en H avec PWM :

II.4.1.1 Principe Générale :

L'intérêt du PWM est de pouvoir générer un signal continu à partir d'un système fonctionnant en tout ou rien. Le principe est d'utiliser le filtre passe-bas naturel des systèmes, par exemple l'inertie d'un moteur. Ainsi, à partir d'une succession discrète pendant un temps T, on peut obtenir une valeur continue moyenne pendant ce temps T.

II.4.1.2 Modélisation d'un pont en H avec PWM :

La figure II.20 représente le schéma fonctionnel du variateur avec la commande PWM

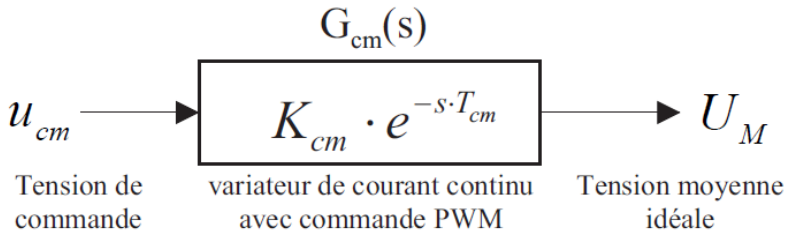


Figure II. 20 : Schéma fonctionnel du variateur avec commande PWM [23]

Avec $$K_{cm} = \frac{U_A}{\hat{U}_{cm}} \quad (2.18)$$

Si la modulation PWM est analogique :

$$T_{cm} = \frac{T_P}{3} \dots \frac{T_p}{2} \quad (2.19)$$

Si la modulation PWM est numérique : $T_{cm} = T_P \quad et \; T_{cycle} = T_P$ (2.20)

II.4.2 Régulation en tension moteur à courant continu :

Pour un petit moteur, on peut poser l'hypothèse : $\boldsymbol{\tau e \ll \tau}$

La tension d'alimentation du moteur est déterminée par le régulateur de vitesse et est amplifiée par le variateur.

Dans la Figure II.21, on représentera le schéma fonctionnel d'une régulation en tension d'un moteur à courant continu avec les fonctions de transfert du moteur et sa charge.

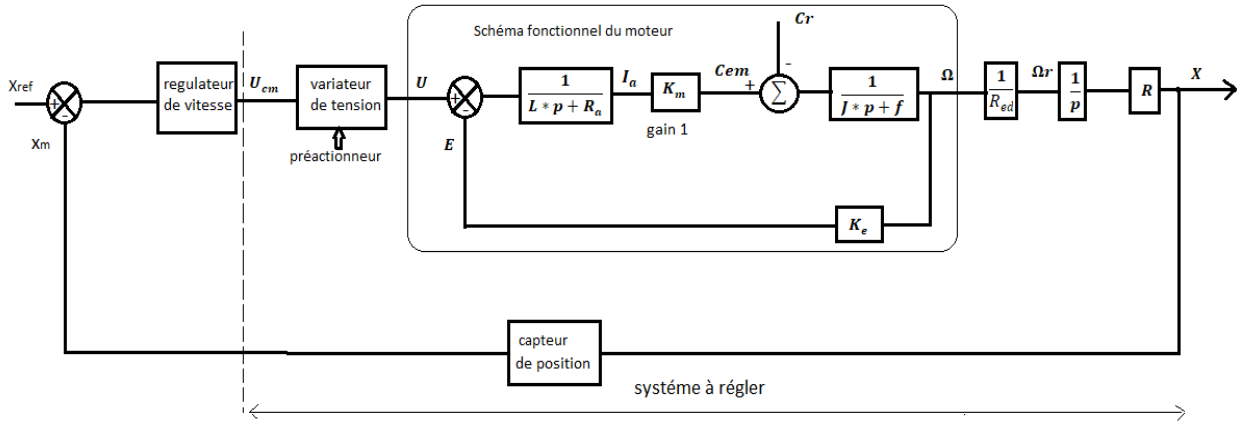


Figure II. 21 : Schéma fonctionnel d'une régulation en tension d'un MCC [23]

II.4.2.1 Fonction de transfert du système à régler :

L'équation (2.21) exprime la fonction de transfert du système à régler

$$G_a(p) = \frac{K_{cm}}{(1+p\tau_{cm})}\frac{1}{K}\frac{1}{(1+p\tau)\cdot(1+p\tau_e)}\frac{K_{m\omega}}{(1+p\tau_{m\omega})} \qquad (2.21)$$

Avec: $K_{a\omega} = \frac{K_{cm}*K_{m\omega}}{K}$

On a la fonction de transfert du système à régler exprimé par l'Eq (2.22) :

$$G_a(p) = K_{a\omega}\frac{1}{(1+p\tau_{cm})}\frac{1}{(1+p\tau)\cdot(1+p\tau_e)}\frac{1}{(1+p\tau_{m\omega})} \qquad (2.22)$$

II.4.2.2 Choix du régulateur :

Un régulateur P ne suffit pas car le couple perturbateur provoque une erreur statique.

Un régulateur PI annule cette erreur statique.

Un régulateur PID n'est pas souhaitable.

-On pourrait compenser l'un des pôles correspondants aux petites constantes de temps

-Mais : le résultat serait une amplification des bruits, ou l'annulation de l'effet d'un filtre de mesure.

Règle des entraînements réglés

Éviter les régulateurs à comportement dérivateur !

Réglage en tension avec un régulateur PI :

L'équation (2.23) représente la fonction de transfert du système avec un régulateur PI :

$$G_a(p) = \mathrm{K_{p\omega}}\frac{(1+\mathrm{p}\tau_{\mathrm{i}\omega})}{\mathrm{p}\tau_{\mathrm{i}\omega}}K_{a\omega}\frac{1}{(1+p\tau_{cm})}\frac{1}{(1+p\tau)(1+p\tau_e)}\frac{1}{(1+p\tau_{m\omega})} \qquad (2.23)$$

$\mathrm{K_{p\omega}}\frac{(1+\mathrm{p}\tau_{\mathrm{i}\omega})}{\mathrm{p}\tau_{\mathrm{i}\omega}}$: fonction de transfert du régulateur PI

$et\ K_{a\omega}\frac{1}{(1+p\tau_{cm})}\frac{1}{(1+p\tau)(1+p\tau_e)}\frac{1}{(1+p\tau_{m\omega})}$: fonction de transfert du système à régler

On ajuste d'abord la constante de temps du régulateur par compensation du pôle dominant

$\tau_{\mathrm{i}\omega} = \tau$. Mais la constante de temps mécanique est très variable : soit en fonction de lacharge entraîner (inertie) ou en fonction de la température du moteur (R)

II.4.3 Constats :

- La performance du régulateur de vitesse dépend assez fortement des variations de charge (Jch) Et de la température du moteur (R)
- Le schéma fonctionnel mais aussi les équations du moteur, donnent le lien entre le courant et la vitesse pour un couple résistant nul :

$$\Omega\,(p) = \frac{C_{em}}{Jp + f} = \frac{K_m * I_a}{Jp + f} \qquad (2.24)$$

-La fonction de transfert qui lie un courant à la consigne de vitesse vaut donc

$$G_{\omega i}(p) = \frac{I_a(p)}{U\omega c(p)} = \frac{I_a(p)}{\Omega\,(p)} \frac{\Omega\,(p)}{U\omega c(p)} = \frac{Jp + f}{K_m} G_{u\omega}(p) \qquad (2.25)$$

-Pour un saut de consigne de vitesse, la réponse indicielle du courant

(Jch et R nominal, max min,) est donnée par la Figure II. 22:

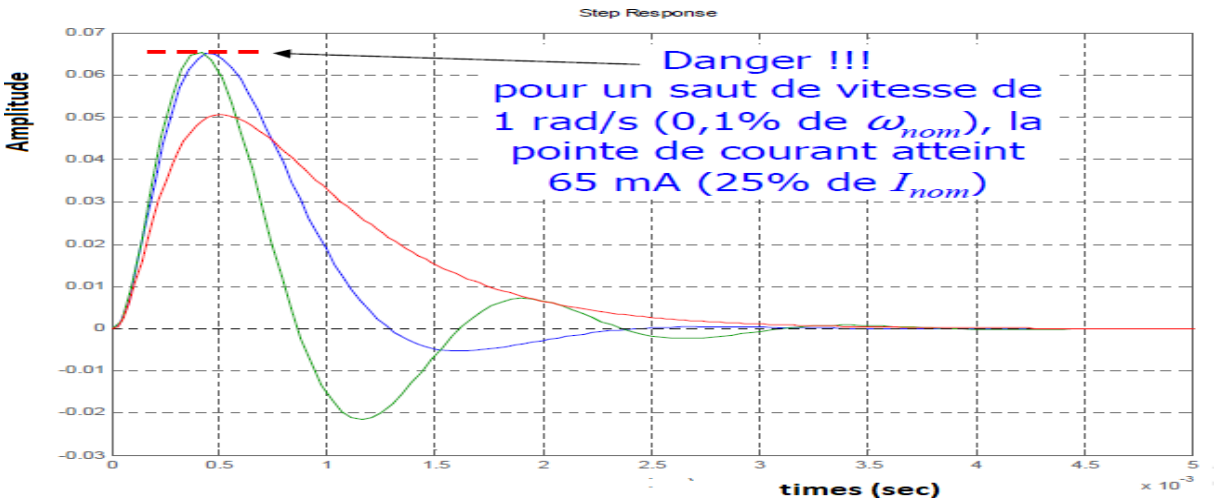


Figure II. 22 : Diagramme de la réponse indicielle [23]

II.5 Architecture d'un API :

La structure matérielle interne d'un API obéit au schéma donné sur la figure II.24

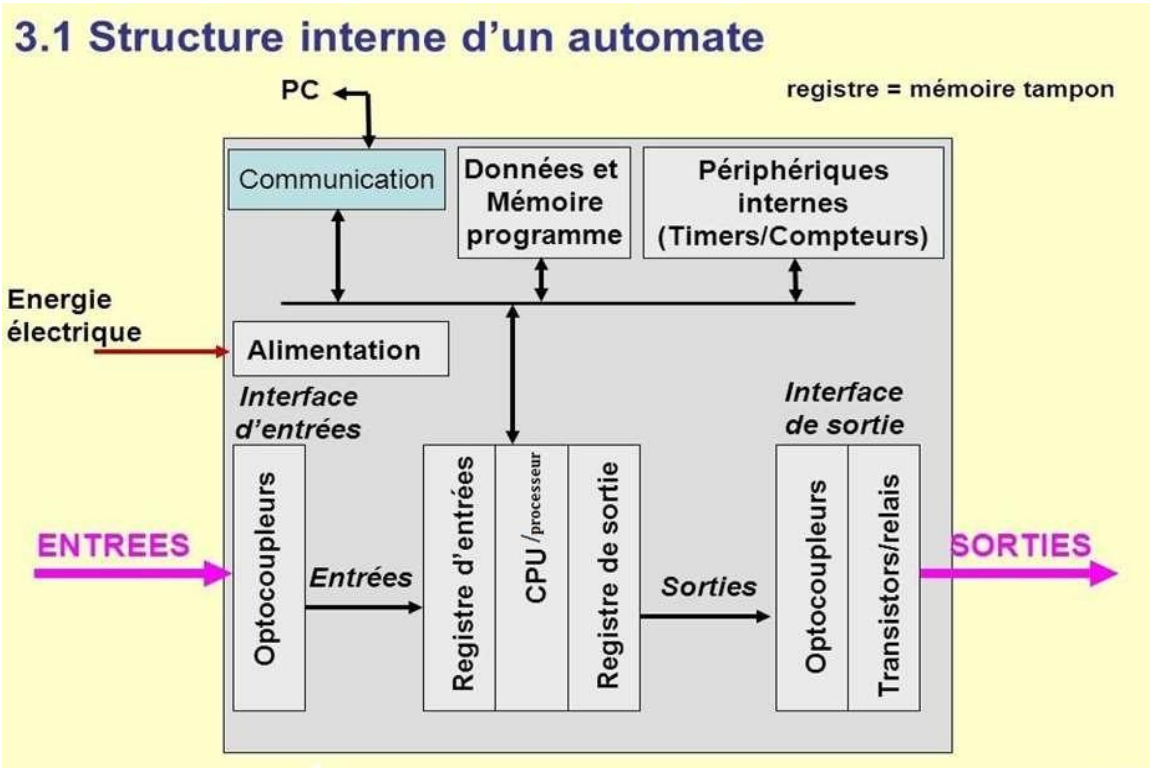


Figure II. 23 La structure matérielle interne d'un API [24]

Détaillons successivement chacun des composants qui apparaissent sur ce schéma

II.5.1 Unité centrale (UC) :

L'unité centrale représente le cœur de la machine, et comprend le regroupement du processeur et de la mémoire centrale. Elle commande l'interprétation et l'exécution des instructions programme. Les instructions sont effectuées les unes après les autres, séquencées parune horloge.

II.5.2. Microprocesseur :

Le microprocesseur réalise toutes les fonctions logiques ET, OU, les fonctions de temporisation, de comptage, de calcul... à partir d'un programme contenu dans sa mémoire.

Il est connecté aux autres éléments (mémoire et interface E/S) par des liaisons parallèles appelées 'BUS' qui véhiculent les informations sous forme binaire.

II.5.3 La zone mémoires :

La Zone mémoire va permettre :

De recevoir les informations issues des capteurs d'entrées· De recevoir les informations générées par le processeur et destinées à la commande des sorties (valeur des compteurs, des temporisations, …) · De recevoir et conserver le programme du système.

Action possible sur une mémoire :

ECRIRE : pour modifier le contenu d'un programme·

EFFACER : pour faire disparaître les informations qui ne sont plus nécessaire·

LIRE : pour lire le contenu d'un programme sans le modifie

Technologie des mémoires :

RAM (Random Accès Memory): mémoire vive dans laquelle on peut lire, écrire eteffacer (contient le programme)-

ROM (Read Only Memory): mémoire morte dans laquelle on ne peut que lire.-

EPROM : mémoires mortes reprogrammables effaçables aux rayons ultra-violets. -

EEPROM : mémoires mortes reprogrammables effaçables électriquement.

Remarque :

Exemple :

Soit une mémoire de 8 Koctets = 8 x 1024 x 8 = 65 536 BITS. Cette mémoire peut contenir 65536 informations binaires

II.5.4. Les interfaces d'entrées/sorties :

Les entrées reçoivent des informations en provenance des éléments de détection et dupupitre opérateur.

Les sorties transmettent des informations aux pré-actionneurs et aux éléments de signalisation du pupitre.

II.5.5 Le module d'entrées :

Les cartes d'entrées logiques :

Les cartes d'entrées logiques (cartes d'entrées tout ou rien) permettent de raccorder àl'automate les différents capteurs logiques tels que :

Boutons poussoirs

Fin de course

Capteurs de proximité inductifs ou capacitifs

Capteurs photoélectriques

Elles assurent l'adaptation, l'isolement, le filtrage et la mise en forme des signaux électriques.

Une diode électroluminescente située sur la carte donne l'état de chaque entrée.

Les cartes d'entrées analogiques :

Les cartes d'entrées analogiques permettent de gérer des grandeurs analogiques en faisant varier un code numérique au sein du module.

Les entrées analogiques disposent d'un seul convertisseur analogique /numérique, elles sont scrutées les unes à la suite des autres par un multiplexeur.

II.5.6 Le module de sorties :

Les cartes de sorties logiques :

Les cartes de sorties logiques (tout ou rien) permettent de raccorder à l'automate les différents pré-actionneurs tels que :

Les contacteurs

Les voyants

Les distributeurs

Les afficheurs...

Ces cartes possèdent soit des relais, soit des triacs, soit des transistors. L'état de chaque sortie est visualisé par une diode électroluminescente.

Les cartes de sortie analogiques :

Les cartes de sortie analogiques permettent de gérer des grandeurs analogiques en faisant varier un code numérique au sein du module. Il existe deux grands types de cartes de sorties :

Haut niveau avec une résolution de 8 bits en tension 0/10 V ou en intensité, 0/20 mA ou 4/20 mA ;

Haut niveau avec une résolution de 12 bits en tension 0/10V, 0/5V, ±5V, ±10V ou en intensité 0/20mA ou 4/20mA.

Ces modules assurent la conversion numérique/analogique. L'intensité ou la tension est proportionnelle à la valeur numérique. Avec les résolutions 8 bits il y a 256 valeurs numériques possibles, tandis qu'avec les résolutions de 12 bits il y en a 4096.

II.5.7 Fonctionnement de l'interface d'entrées/sorties :

Interfaces d'entrées :

Elles sont destinées à :

Recevoir l'information en provenance des capteurs

Traiter le signal en le mettant en forme, en éliminant les parasites et en isolant électriquement l'unité de commande de la partie opérative.

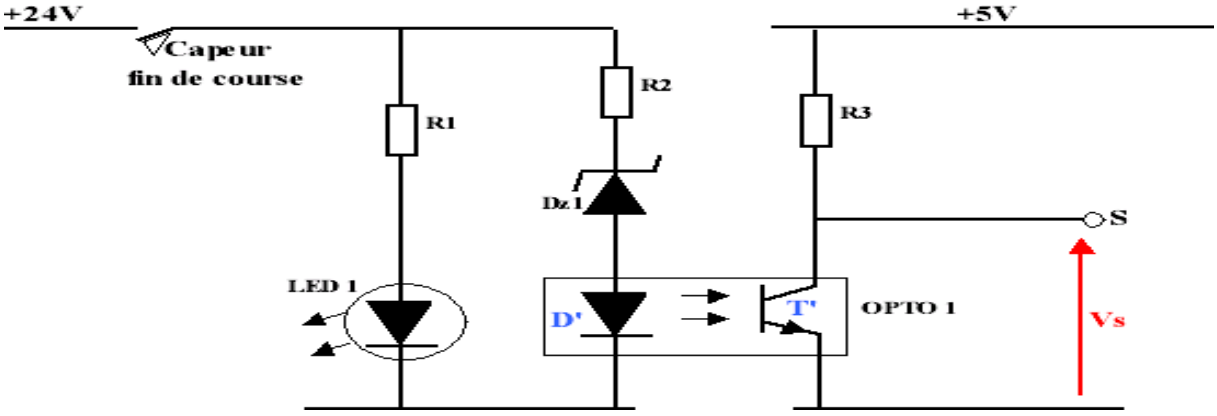


Figure II. 24 Exemple de circuit électronique d'interface d'entréeLors de la fermeture du capteur [24]

LED1 signale que l'entrée automate est actionnée

La LED de l'optocoupleur s'éclaire

La photo transistor de l'optocoupleur devient passante

La tension Vs=OV

Donc lors de l'activation d'une entrée automate, l'interface d'entrée envoie un O logique à l'unité de traitement et une 1 logique lors de l'ouverture du contact du capteur (entrée non actionnée).

Interfaces de sorties :

Elles sont destinées à :

Commander les pré-actionneurs et éléments des signalisations du système

Adapter les niveaux de tension de l'unité de commande à celle de la partie opérative dusystème en garantissant une isolation galvanique entre ces dernières

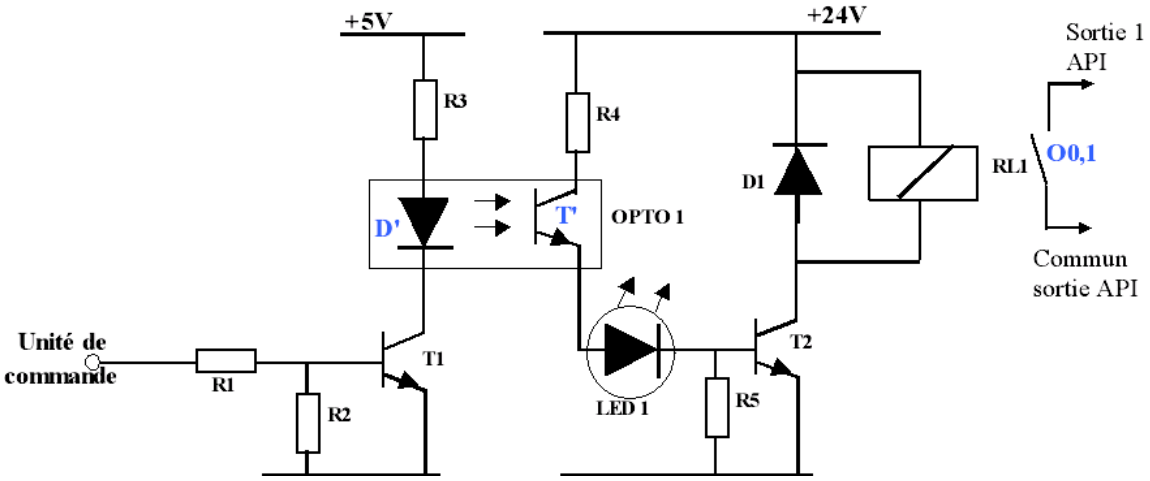


Figure II. 25 Exemple de circuit électronique d'interface de sortie [24]

L'unité de commande envoie une 1 logique (5V)

T1 devient passant, donc D' s'éclaire

La photo transistor T' de l'optocoupleur devient passant

LED 1 s'éclaire et nous informe de la commande de la sortie O0,1

T2 devient passant

La bobine RL 1 devient sous tension et commande la fermeture du contact de la sortie O0, 1 Donc pour commander une sortie automate l'unité de commande doit envoyer :

Une 1 logique pour actionner une sortie API

Une 0 logique pour stopper la commande d'une sortie API

Le module d'alimentation :

Composé de blocs qui permettent de fournir à l'automate l'énergie nécessaire à son fonctionnement. A partir d'une alimentation en 220 volt alternative, ces blocs délivrent des sources de tension dont l'automate a besoin : 24V, 12V ou 5V en continu. En règle générale, un voyant positionné sur la façade indique la mise sous tension de l'automate.

Le module de communication :

Comprend les consoles, les boîtiers de tests et les unités de dialogue en ligne :

Les consoles :

Il existe deux types de consoles. L'une permet le paramétrage et les relevés d'informations (modification des valeurs, et visualisation), l'autre permet en plus la programmation, le réglage et

L'exploitation. Cette dernière dans la phase de programmation effectue :

L'écriture

La modification

L'effacement

Le transfert d'un programme dans la mémoire de l'automate ou dans une mémoire (REPROM).

La console peut 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. Les consoles sont équipées souvent d'un écran à cristaux liquides. Certaines consoles ne peuvent être utilisées que connectées à un automate, d'autres peuvent fonctionner de manière autonome grâce à la mémoire interne et à leur alimentation.

Les boîtiers de tests :

Destinées aux personnels d'entretien, ils permettent de visualiser le programme ou les valeurs des paramètres. Par exemple :

Affichage de la ligne de programme à contrôler

Visualisation de l'instruction (code opératoire et adresse de l'opérande)

Visualisation de l'état des entrées

Visualisation de l'état des sorties.

II.5.8 Le Bus :

C'est un ensemble de conducteurs qui réalisent la liaison entre les différents éléments de l'automate. Dans un automate modulaire, il se présente sous forme d'un circuit imprimé situé au fond du bac et supporte des connecteurs sur lesquels viennent s'enficher les différents modules : processeur, extension mémoire, interfaces et coupleurs.

Le bus est organisé en plusieurs sous-ensembles destinés chacun à véhiculer un type bien défini d'informations :

Bus de données.

Bus d'adresses.

Bus de contrôle pour les signaux de service tels que tops de synchronisation, sens des échanges, contrôle de validité des échanges, etc…

Bus de distribution des tensions issues du bloc d'alimentation.[24]

II.6 L'objet technologique PID_Compact:

L'objet technologique PID_Compact permet de disposer d'un régulateur PID avec optimisation intégrée pour organes de réglage à action proportionnelle.

Les modes suivants sont disponibles :

Inactif

Optimisation préalable

Optimisation fine

Mode automatique

Mode manuel

Valeur de sortie de remplacement avec surveillance d'erreurs

A cette étape, ce régulateur doit être paramétré, paramétré et mis en service pour le mode automatique.

Lors la mise en service, nous utiliserons les algorithmes d'optimisation intégrés et nous enregistrerons le comportement de régulation du système réglé.

L'appel de l'objet technologique PID_Compact s'effectue toujours dans un OB d'alarme cyclique, dont le temps de cycle est défini de manière fixe à 50 ms.

La présélection de la consigne de vitesse s'effectue sous la forme d'une constante à l'entrée "Setpoint" de l'objet technologie PID_Compact exprimée en tours par minute (plage : +/- 50 tr/min). Le type de données est en virgule flottante 32 bits (Real).

La mesure (valeur réelle) de vitesse -B8 (Capteur de mesure de la vitesse réelle du moteur +/-10V correspondent à +/- 50 tr/min) est saisie sur l'entrée "Input_PER".

La sortie du régulateur "Output_PER" est directement paramétrée au signal -U1 (valeur de réglage de la vitesse du moteur dans les deux directions +/-10V correspondent à +/- 50 tr/min).

Le régulateur ne doit être activé aussi longtemps que la sortie -Q3 (Moteur du convoyeur -M1 vitesse variable) est commandée. Si celle-ci n'est pas commandée, alors le régulateur doit être mis en mode inactif au travers de l'entrée "Reset".

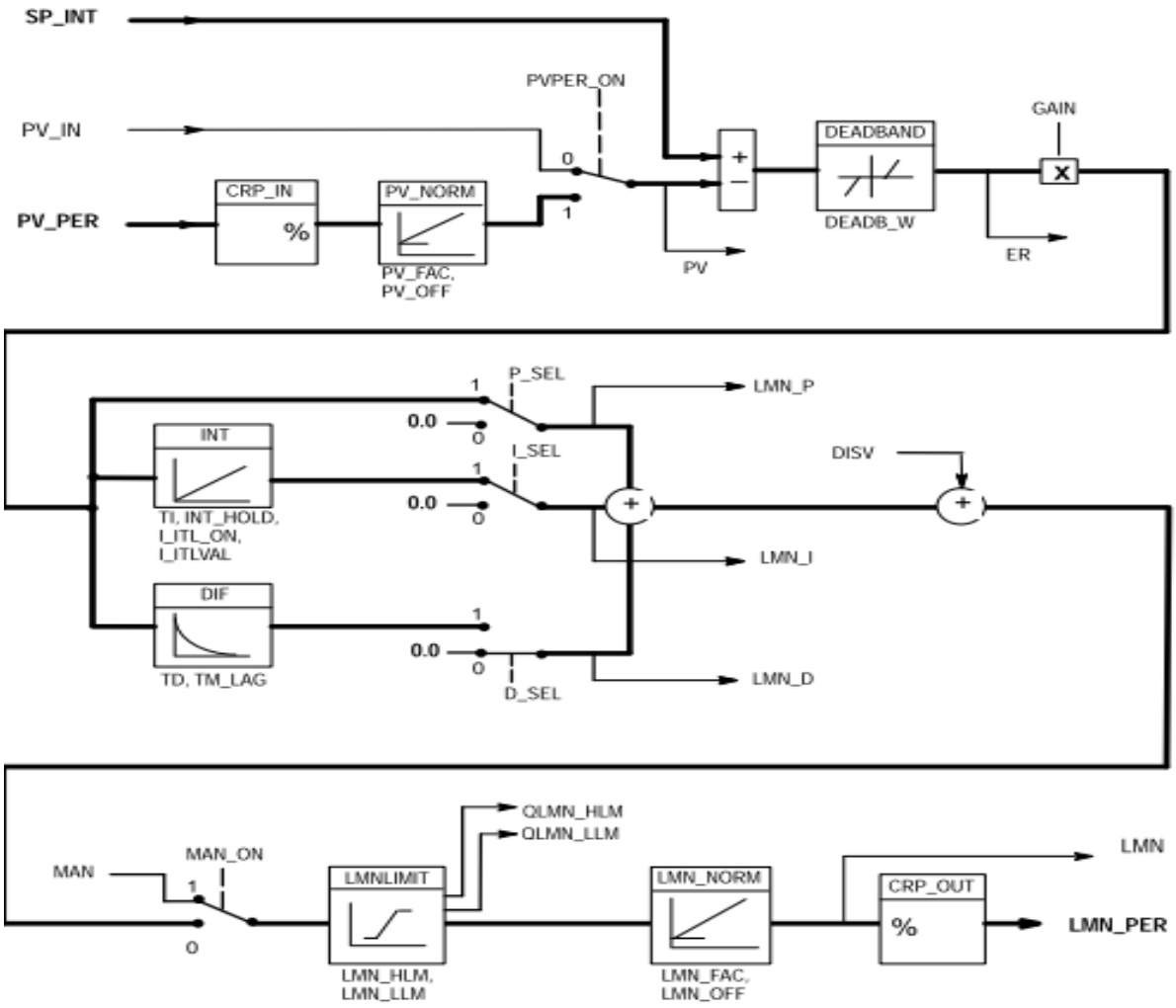


Figure II. 26 Schéma fonctionnel de CONT_C [36]

II.7 Présentation générale de l'automate S7-300 :

Le système d'automatisation SIMATIC S7-300 est un automate modulaire de milieu de Gamme SIMATIC, S7 désigne un produit de la société SIEMENS, il est synonyme de la gamme des Automates programmable.

Les automates programmables SIEMENS sont des appareils fabriqués en série conçus Indépendamment d'une tâche précise. Tous les éléments logiques, fonctions de mémoire, Temporisations, compteurs etc., nécessaires à l'automatisation sont prévus par le fabricant et Intégrés à l'automate. Ils se distinguent principalement par le nombre des

Entrées et sorties ;

Compteurs ;

Temporisations ;

Mémentos.

II.8 Caractéristique de l'automate S7-300 :

L'automate S7-300 offre les caractéristiques suivantes :

Gamme diversifiée de CPU ;

Gamme complète de modules ;

Possibilité d'extension jusqu'à 32 modules ;

Bus de fond de panier intégré au module ;

Possibilité de mise en réseau avec MPL, PROFIBUS ou INDUSTRIAL ETHERNET ;

Raccordement central de la PG avec accès à tous les modules ;

Liberté de montage aux différents emplacements ;

La vitesse du travail ;

Configuration et paramétrage à l'aide de l'outil configuration matérielle ;

Programmation libre ;

II.9 Constitution de l'automate S7-300 :

L'automate programmable S7-300 est un système d'automatisation modulaire offrant la gamme demodules suivant :

Module d'alimentation (PS) 2A ,5A, 10A.

Unité centrale (CPU)

Module de signaux (SM) pour entrées et sorties TOR et analogiques.

Module d'extension (IM) pour configuration multi rangée du S7-300.

Module de fonction (FM) pour les fonctions spéciales.

Processus de communication (CP) pour la connexion au réseau.

II.9.1 La configuration matérielle de notre CPU est donnée comme suit :

Le module d'alimentation : PS 307 5A avec Alimentation externe 120/230 V CA à l'entrée et24V CC/5 A à la sortie. Ref : 6ES7 307-1EA01-0AA0. [24]

II.9.2 La CPU 314C-2 DP avec :

Mémoire de travail 192 Ko ;

0,6 ms / 1000 instructions ;

DI24 / DO16 ; AI5 / AO2 intégré ;

4 sorties d'impulsions (2,5 kHz) ;

4 canaux de comptage et de mesure avec codeurs incrémentaux 24 V (60 kHz) ;

Fonction de positionnement intégrée ;

Interface PROFINET et 2 ports ;

MRP ; PROFINET CBA ;

Proxy PROFINET CBA ;

Protocole de transport TCP / IP ;

Interface combinée MPI / DP (maître MPI ou DP ou esclave DP) ;

Configuration multi-niveau jusqu'à 31 modules ;

Capable d'envoyer et de recevoir en échange direct de données ;

Temps de cycle de bus constant ;

Routage ;

firmware V3.3 [25]

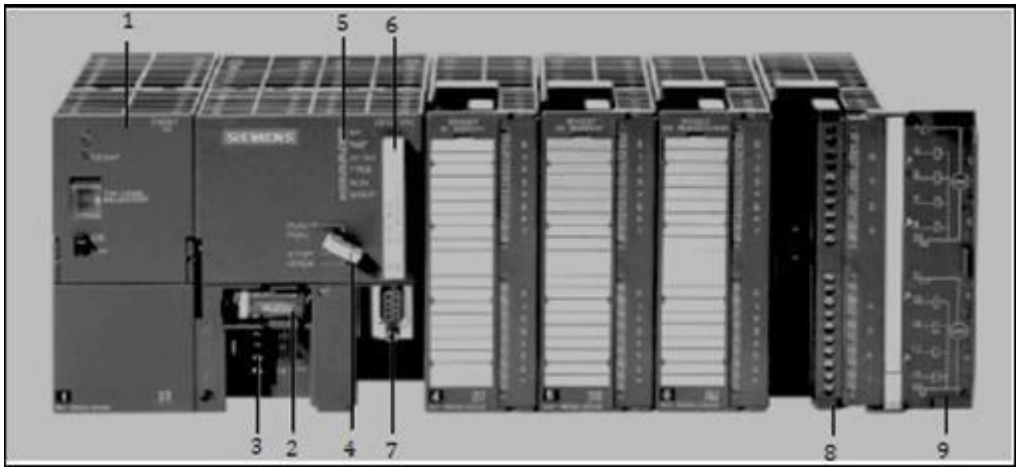


Figure II. 27 *: API* S7-300 [9]

Module d’alimentation

Pile de sauvegarde

Connexion au *24Vcc*

Commutateur de mode

LED de signalisation d'état et de défauts

Carte mémoire

Interface multipoint *(MPI)*

Connexion frontale

Volet en face avant [26]

II.10 TIA Portal (Totally Integrated Automation):

En réponse à la pression internationale croissante de la concurrence, il est aujourd'hui plus que jamais important d'exploiter à fond tous les potentiels d'optimisation sur l'ensemble du cycle de vue d'une machine ou d'une installation. Des processus optimisés permettent de réduire le coût total de possession, de réduire le temps entre la conception et la commercialisation et d'améliorer la qualité. Cet équilibre parfait entre qualité, temps et coûts et plus que jamais le facteur décisif de la réussite industrielle. Totally Integrated Automation apporte une réponse optimale à toutes les exigences et offre un concept ouvert vis à vis des normes internationales et de systèmes tiers. Avec ses six principaux caractéristiques systèmes et robustesse, Le TIA Portal accompagne l'ensemble du Cycle de vie d'une machine ou d'une installation. L'architecture système complète offre des solutions complètes pour chaque segment d'automatisation sur la base d'une gamme de produits complète.

II.10.1 Description du logiciel TIA Portal :

La plateforme « Totally Intergrated Automation Portal » est le nouvel environnement de travail Siemens qui permet de mettre en oeuvre des solutions d'automatisation avec un système d'ingénierie intègre comprenant les logiciels SIMATIC Step7 et SIMATIC WinnCC.

II.10.2 Les avantage du logiciel TIA portal :

Programmation intuitive et rapide : avec des éditeurs de programmation nouvellement développés SCL, CONT, LOG, LIST et GRAPH

Efficacité accrue grâce aux innovations linguistiques de STEP 7 : programmation symbolique uniforme, Calculate Box, ajout de blocs durant le fonctionnement, et bien plus encore ;

Performance augmentée grâce à des fonctions intégrées : simulation avec PLCSIM, télémaintenance avec TeleService et diagnostic système cohérent ;

Technologie flexible : Fonctionnalité motion control évolutive et efficace pour les automates S7-1500 et S7-1200 ;

Sécurité accrue avec Security Integrated : Protection du savoir-faire, protectioncontre la copie, protection d'accès et protection contre la falsification ;

Environnement de configuration commun avec pupitres IHM et entraînements dans l'environnement d'ingénierie TIA Portal

II.10.3 SIMATIC STEP 7 :

SIMATIC STEP 7, intégré à TIA Portal, est le logiciel de configuration, programmation, vérification et diagnostic de tous les automates SIMATIC. Doté d'un grand nombre de fonctions conviviales, SIMATIC STEP 7 garantit une efficacité nettement supérieure pour toutes les tâches d'automatisation, qu'il s'agisse de la programmation, de la simulation, de la mise en service ou de la maintenance.

II.10.4 Vue du portal et vue du projet :

Lorsqu'on lance TIA Portal, l'environnement de travail se décompose de deux types de vue :

Vue du portal : elle est axée sur les tâches à exécuter et sa prise en main est très rapide ;

Vue du projet : elle comporte une arborescence avec les différents éléments du projet, les éditeurs requis s'ouvrent en fonction des tâches à réaliser. Données, paramètres et éditeurs peuvent être visualisés dans une seule et même vue

II.10.4.1 Vue du portal :

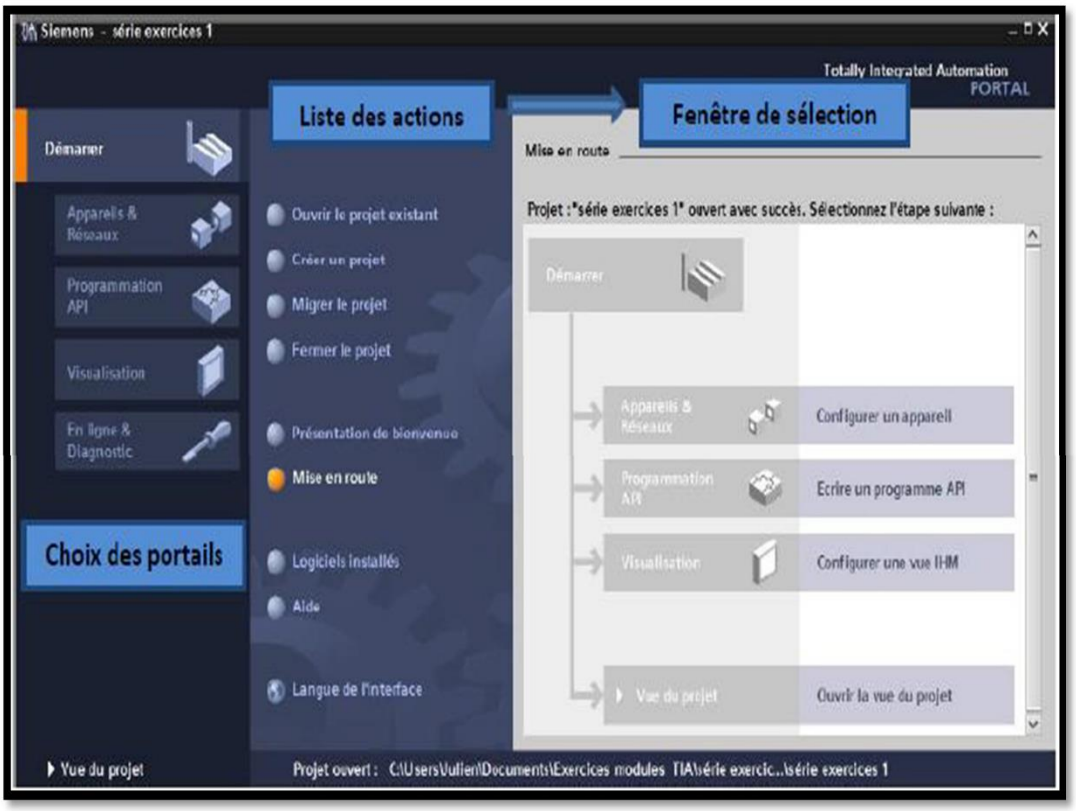


Figure II. 28 *:* Vue du portal.

Chaque portail permet de traiter une catégorie de tâche (action) la fenêtre affiche la liste des actions peuvent être réalisées pour la tâche sélectionnée

II.10.4.2 Vue du projet :

L'élément « Projet » contient l'ensemble des éléments et des données nécessaires pour mettre en œuvre la solution d'automatisation souhaitée.

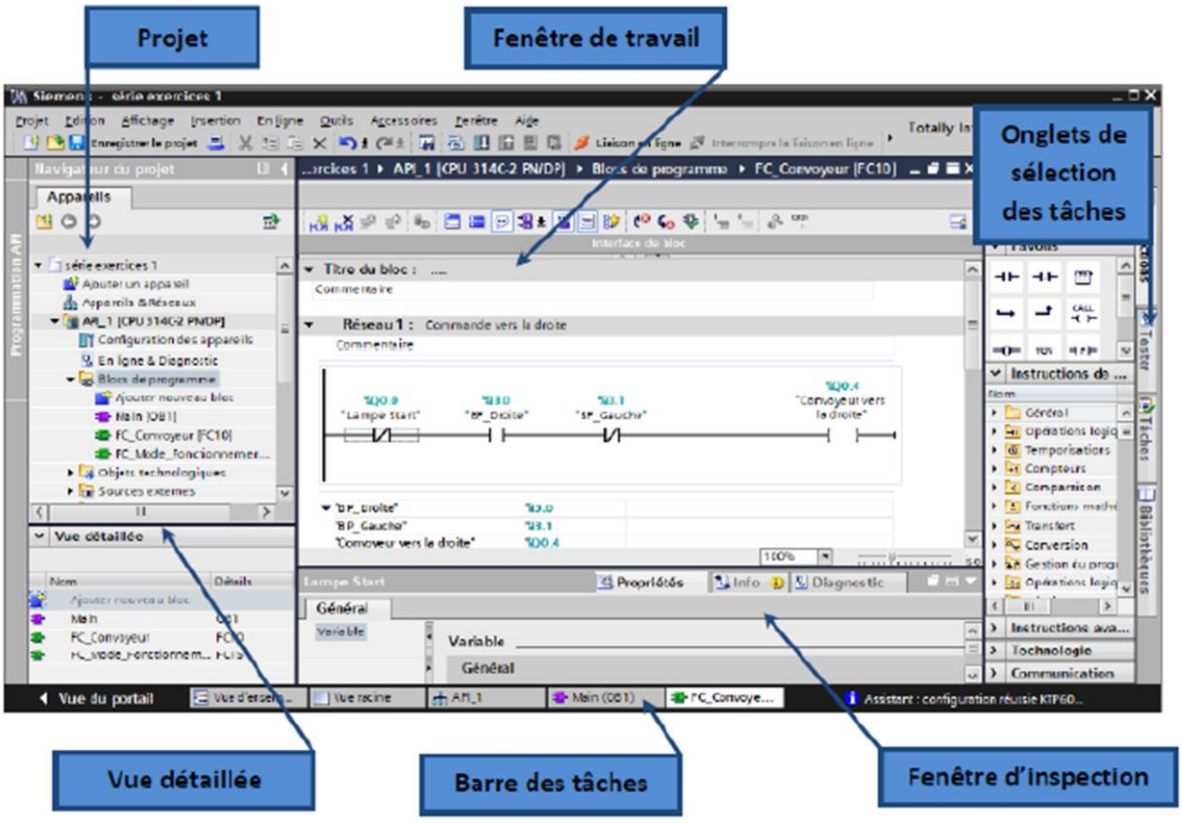


Figure II. 29 : Vue du projet.

La fenêtre de travail : permet de visualiser les objets sélectionnés dans le projet pour être traités. Il peut s'agir des composants matériels, des blocs de programme, des tables des variables, des HMI… ;

La fenêtre d'inspection : permet de visualiser des informations complémentaires sur un objet sélectionné ou sur les actions en cours d'exécution (propriété du matériel sélectionné, messages d'erreurs lors de la compilation des blocs de programme…) ;

Les onglets de sélection de tâches ont un contenu qui varie en fonction de l'objet sélectionné (configuration matérielle, bibliothèques des composants, bloc de programme, instructions de programmation). Cet environnement de travail contient énormément de données. Il est possible de masquer ou réduire certaines de ces fenêtres lorsque l'on ne les utilise pas. Il est également possible de redimensionner, réorganiser, désancrer les différentes fenêtres. [27]

II.10.5 Blocks de programme :

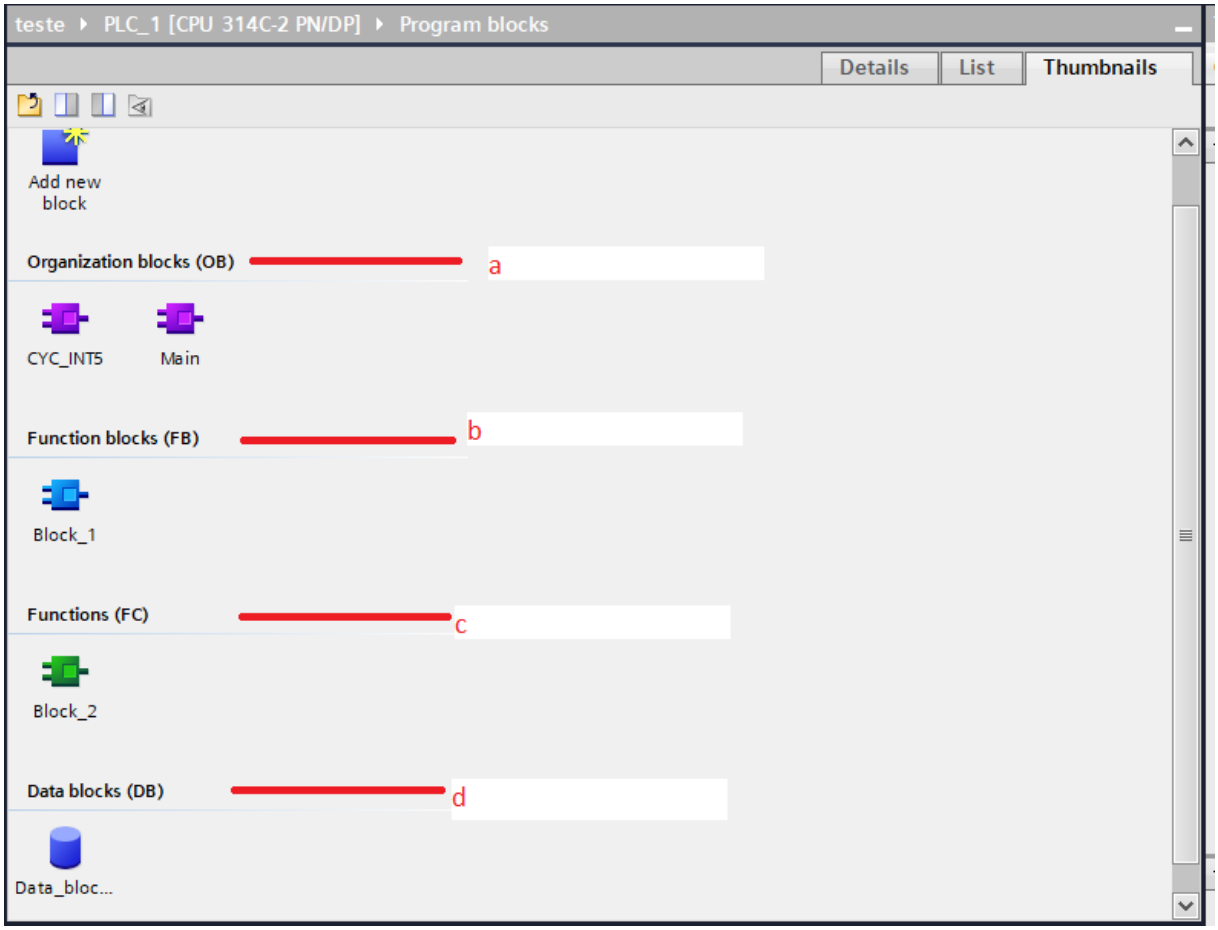


Figure II. 30 Blocks de programme

a) Bloc d'organisation (OB)

Un OB est appelé cycliquement par le système d'exploitation et constitue donc une interface entre le programme utilisateur et le système d'exploitation.

L'OB contient des instructions d'appel de blocs indiquant à l'unité de commande de l'automatel'ordre dans lequel il doit traiter les blocs.

Nous avons utilisé l'OB35 pour la programmation des régulateurs, et dans l'OB1 les différents appels des fonctions.

b) Bloc fonctionnel (FB)

Un bloc fonctionnel contient un programme qui est exécuté dès son appel par un autre bloc de code. Ilfacilite la programmation de fonction complexe.

c) Fonction (FC)

Les fonctions font partie des opérations que le concepteur programme. Elles ne possèdent pas de mémoires. Les variables temporaires d'une fonction sont sauvegardées dans la pile de données sont perdues après exécution de la fonction. Les fonctions peuvent faire appel à des

blocs de données globaux pour la sauvegarde de données. Une fonction contient un programme qui est exécuté lorsqu'elle est appelée par un autre bloc de code.

d) Bloc de données (DB) :

Les DB sont utilisés pour la mise à disposition de l'espace mémoire pour des variables de type données. Tous les FB, FC, OB peuvent lire les données contenues dans un DB global ou écrire des données dans un DB global. Ces données sont conservées dans le bloc de données même lorsqu'on quitte DB.

La vérification du bon fonctionnement du programme élaboré est faite par l'outil de simulationPLCSIM de TIAPORTAL (STEP7) et le RUNTIME de WinCC] [28]

II.10.6 Adressage des E/S :

Pour connaitre l'adressage des entrées et sorties présentes dans la configuration matérielle, il faut aller dans « Appareil et réseau » dans le navigateur du projet. Dans la fenêtre de travail, on doit s'assurer d'être dans l'onglet « Vue des appareils » et de sélectionner l'appareil voulu.

On sélectionne la CPU puis à l'aide des deux petites flèches (voir figure), on fait apparaitre l'onglet « **Vue d'ensembles des appareils ».** Les adresses des entrées et sorties apparaisse. On peut les modifier en entrant une nouvelle valeur dans la case correspondante.

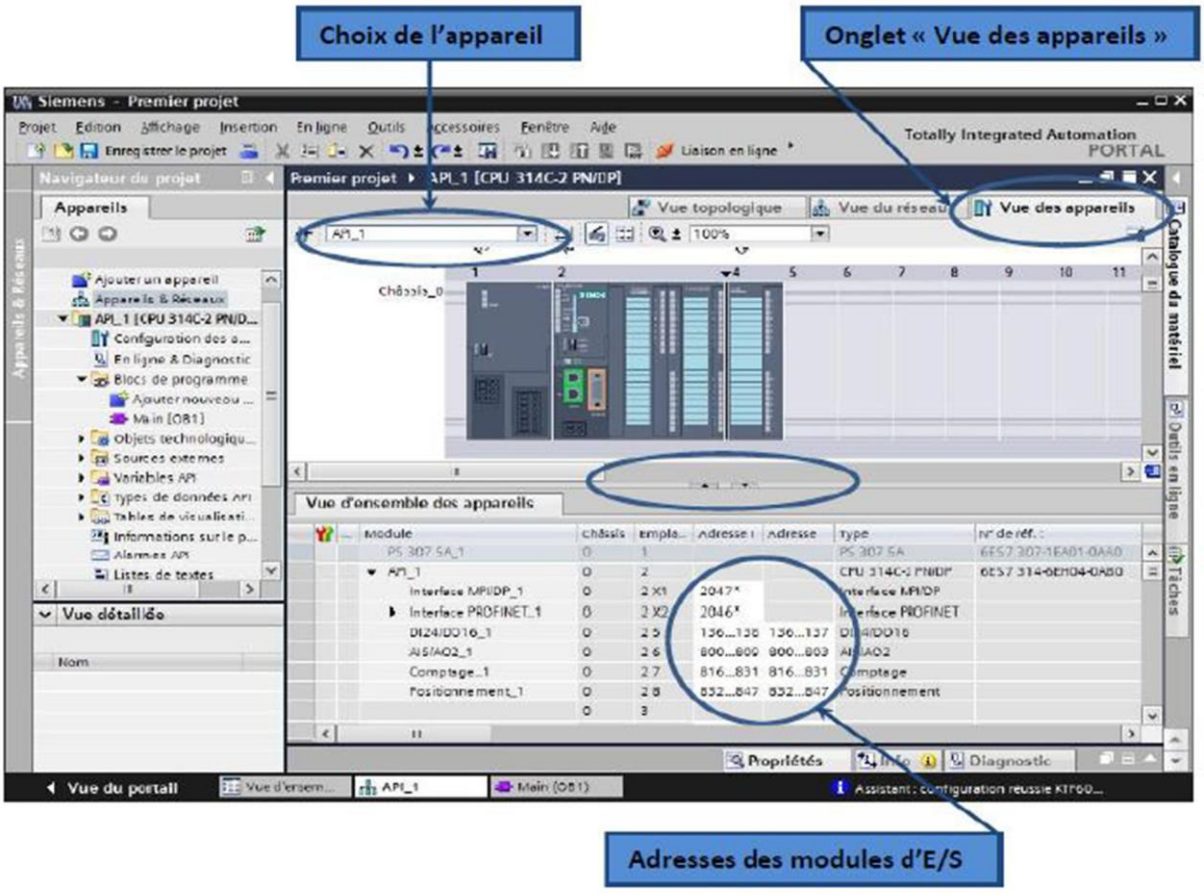


Figure II. 31 Adressage des E/S.

II.10.7 Les variables API :

II.10.7.1 Adresses symbolique et absolue :

Dans TIA portal, toutes les variables globales (entrées, sorties, mémentos,…) possèdent une adresse symbolique et une adresse absolue.

L'adresse absolue : représente l'identificateur d'opérande (I, Q, M,…) et son adresse et numéro de bit.

Adresse symbolique correspond au nom que l'utilisateur a donné à la variable (ex : bouton marche). Le lien entre les adresses symbolique et absolue se fait dans la table des variables API.

Lors de la programmation, on peut choisir d'afficher les adresses absolues, symboliques ou encore les deux simultanément.

II.10.7.2 Table des variables API :

C'est dans la table des variables API que l'on va pouvoir déclarer toutes les variableset les constantes utilisées dans le programme. Lorsque l'on définit une variable API,

Il faut définir :

Un nom : c'est l'adressage symbolique de la variable ;

Le type de donnée : BOOL, INT,… ;

L'adresse absolue : par exemple Q 1.5.

On peut également insérer un commentaire qui nous renseigne sur cette variable. Le commentaire peut être visible dans chaque réseau utilisant cette variable

II.10.8 Liaison avec l'automate :

Il faut maintenant charger la configuration de l'automate dans celui-ci. Pour cela, il faut tout d'abord connecter l'automate au PC en utilisant l'interface SIMATIC S7 PC USB adapté. Ensuite, après avoir sélectionné la vue « En ligne et diagnostique », sélectionnez les options suivantes :

Mode : MPI ;

Interface PG /PC : pc Adapter ;

Adresse Ethernet de la CPU :

Toujours dans les propriétés de la CPU, il est possible de définir son adresse Ethernet. Un double clic sur l'icône Ethernet de la station fait apparaitre la fenêtre d'inspection permettant de définir ses propriétés.

Pour établir une liaison entre la CPU et la console de programmation, il faut affecter aux deux appareils des adresses appartenant au même sous réseau. L'adresse utilisée est **192.168.0.2** de l'automate.

II.10.9 WinCC sur TIA portal :

Le SIMATIC WinCC dans le TIA portal fait partie d'un nouveau concept d'ingénierie intégré qui offre un environnement d'ingénierie homogène pour la programmation et la configuration de solution de commande, de visualisation d'entrainement, c'est le logiciel pour toutes les applications IHM allant de solutions de commande simples avec basic panels aux applications SCADA pour système multipostes basé sur PC. [27]

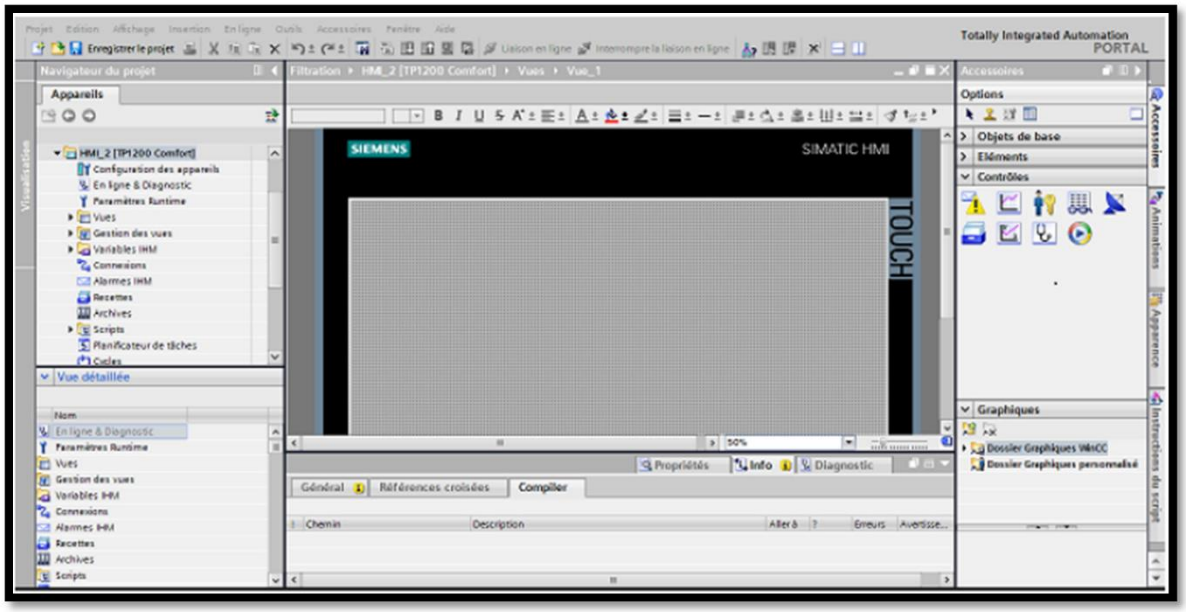


Figure II. 32 : Vue SIMATIC HMI [27.]

II.11 Conclusion :

En utilise la régulation en tension Parce qu'elle commander la vitesse, autant que la tension diminué la vitesse diminue et le moteur s'arrête, à partir de ça nous obtenons la position.

Le logiciel « TIA PORTAL V 15.1», a pour but d'intégrer un nouveau programme sous l'automate programmable industriel « S7-300 », pour augmenter les performances, améliorer la sécurité de l'opérateur, éliminer l'effort physique, augmenter la précision et la rapidité de la tâche réalisée, et minimiser l'erreur.

CHAPITRE.III

Développement et validation de notre simulateur

III.1 Introduction :

Dans ce chapitre, nous allons vous présenter en détail le développement d'un simulateur virtuel qui va nous permettre de faire la régulation de position d'un MCC sans utilisation du matériel. C'est la nouvelle tendance et nouveau moyen dans le monde industriel appelée le « Digital Twain ».

Pour concrétiser notre idée, ce chapitre est composé de deux grandes parties :

- Validation du modèle,
- Implémentation d'une régulation de position sur simulateur.

Pour valider les résultats de notre projet qui est la régulation en position utilisant le cerveau de commande qui est l'API S7-300, nous avons trouvé judicieux de les comparer avec les résultats trouvés sous Matlab/SimPower System qui est un outil incontournable dans notre domaine en général et dans notre cursus en particulier. Pour cela, tous les résultats seront comparés que ce soit en boucle ouverte (validation du modèle) ou en boucle fermé (but principal de notre travail qui la régulation de position). Notre simulateur sera développé sur le logiciel WinCC sous Tia Portal v15.1. Par contre, la régulation et l'animation du simulateur sera programmé en utilisant le logiciel Step7 qui se trouve aussi sous Tia Portal v15.1. Tous les résultats trouvés seront discutés et commentés.

Il faut savoir que la validation du modèle et tous les résultats trouvés dans ce chapitre sont basés sur les paramètres suivants du moteur qui sont obtenus à partir de la bibliothèque Simulink SimPower System.

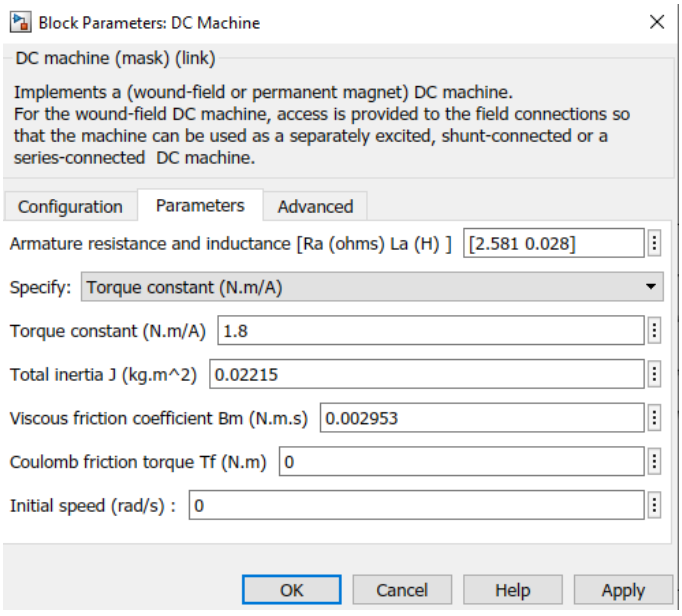


Figure III. 33 : Paramètres de moteur de la bibliothèque SimPower System.

III.2 Validation du modèle :

Pour valider notre modèle, nous avons appliqué la méthodologie suivie durant notre cursus à savoir :

- Régulation en boucle ouverte utilisant les toolBox de Matlab Simulink pour vérifier tout d'abord si notre processus est stable. Cette étape va nous permettre aussi d'obtenir les différents graphes nécessaires pour la validation
- Validation du modèle d'état de la MCC sous Matlab/Script en obtenant les différentes réponses qui seront comparés à ceux obtenus dans Simulink
- Implémentation du même modèle au niveau de notre API S7-300

III.2.1 Simulation du modèle en boucle ouvert sous Matlab/SimPower System :

Le schéma suivant représente le bloc Simulink utilisé pour la validation du modèle. C'est une régulation en boucle ouverte. A partir de cette simulation, nous avons obtenu les graphes suivants.

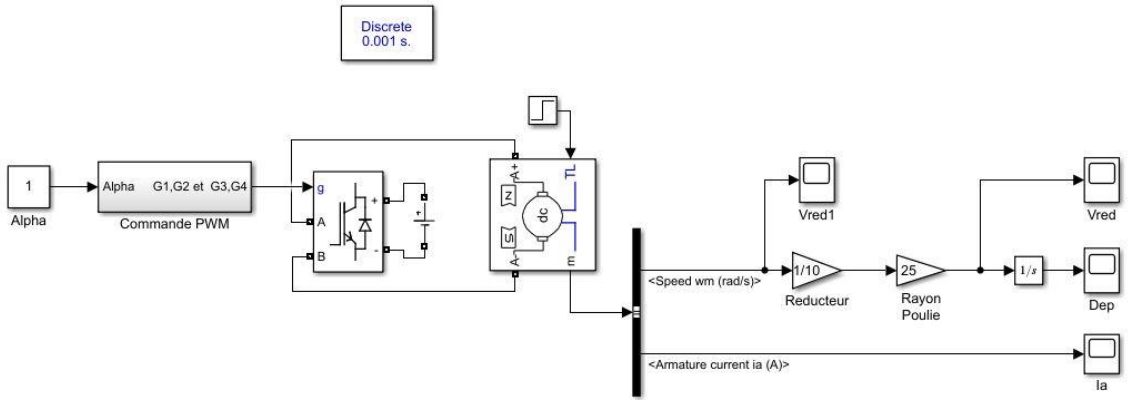


Figure III. 34 : Schéma bloc en boucle fermé de la régulation en position.

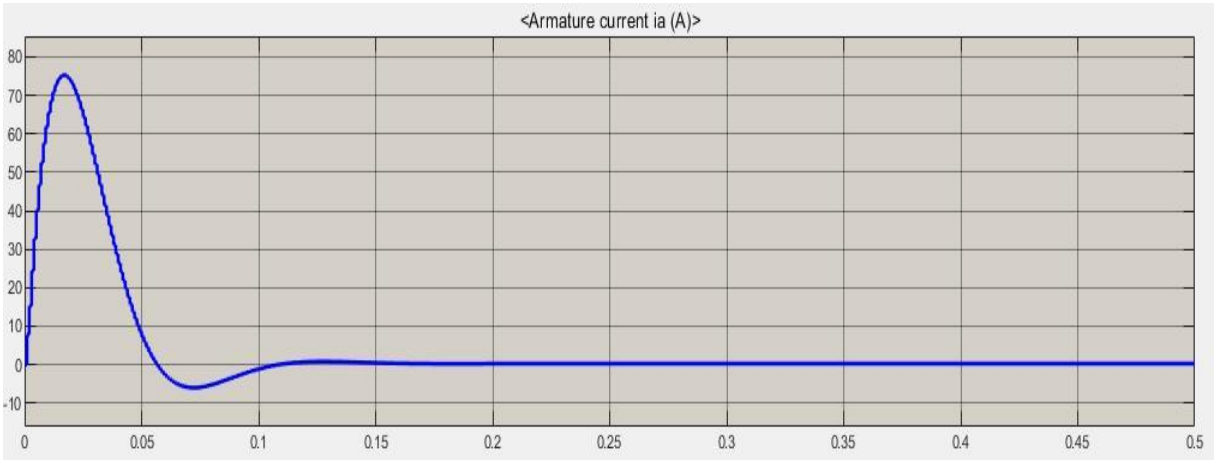


Figure III. 35 : Allure du courant.

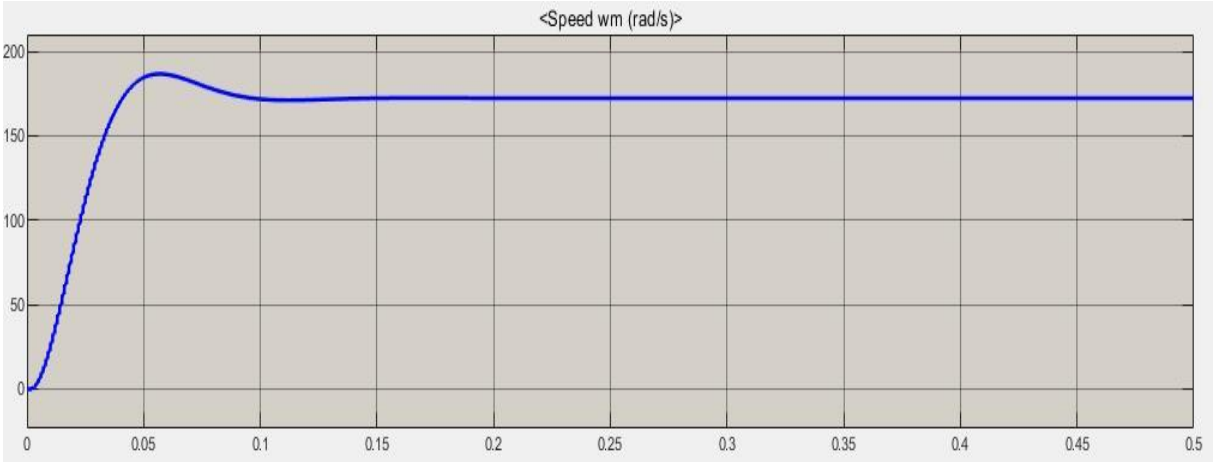


Figure III. 36 : Allure de la vitesse angulaire.

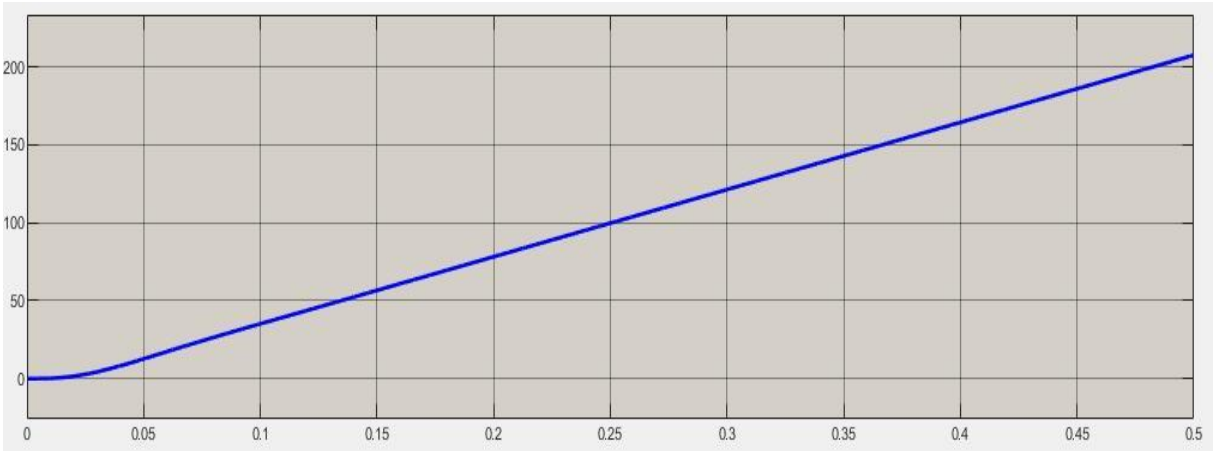


Figure III. 37 : Allure du déplacement.

III.2.2 Validation du modèle par Matlab/Script :

On a passé à la programmation du modèle d'état du MCC sous Script pour validation et ce sera le même modèle qui sera implémenter dans l'API.

III.2.2.1 Equations mathématique du MCC dans le domaine discret :

D'après l'équation électrique de la MCC donnée par l'équation suivante :

$$u(t) = Ri_a + L\frac{di_a}{dt} + e(t) \tag{3.1}$$

D'après l'équation (3.1) et en remplaçant *e(t)* par son équation $e(t) = K_e\Omega(\mathrm{t})$, on obtient l'équation suivante :

$$\frac{di_a}{dt}(t) = -\frac{R_a}{L_a}.i_a(t) - \frac{K_e}{L_a} * \Omega(\mathrm{t}) + \frac{1}{L_a}\,u(t) \tag{3.2}$$

u(t) est la tension appliquée au moteur. En d'autres termes, c'est la commande de notre processus qui dépend essentiellement du rapport cyclique de notre pré-actionneur (qui est dans

notre cas u hacheur en pont) et de la tension du bus continu Vd_0 : $u(t) = \alpha\, Vd_0$. Par conséquent, l'équation (3.2) devient :

$$\frac{di_a}{dt}(t) = -\frac{R_a}{L_a}.\, i_a(t) - \frac{K_e}{L_a} * \Omega(\mathrm{t}) + \frac{Vd_0}{L_a}\, \alpha(t) \tag{3.3}$$

L'équation mécanique après transformation nous donne l'équation suivante :

$$\frac{d\Omega(\mathrm{t})}{dt} = \frac{K_m}{J} \times I_a\ - \frac{f}{J}\Omega(\mathrm{t}) \tag{3.4}$$

Dans le domaine discret :

$$\frac{di_a}{dt}(t) = \frac{I_a(k+1) - I_a(k)}{T_s} \quad (3.5) \quad ; \quad \frac{d\Omega(\mathrm{t})}{dt} = \frac{\Omega(\mathrm{k}+1) - \Omega(\mathrm{k})}{T_s} \quad (3.6)$$

Remplacer l'équation (3.5) dans l'équation (3.3), on obtient

$$I_a(k+1) = \left(1 - \frac{R_a}{L_a} * T_s\right) I_a(k) - \frac{K_e}{L_a} * T_s * \Omega(\mathrm{k})\ + \frac{Ts}{L_a}\, Vd_0 * \alpha(k) \tag{3.7}$$

Remplacer l'équation (3.6) dans l'équation (3.4), l'équation obtenue sera comme suit :

$$\Omega(\mathrm{k}+1) = \left(1 - \frac{f}{J} * T_s\right) \Omega(\mathrm{k}) + \frac{K_m}{J} * T_s * I_a\,(k) \tag{3.8}$$

Pour obtenir le déplacement, le passage de la vitesse angulaire à la vitesse linéaire est obligatoire, il suffit de multiplier la vitesse angulaire par le rayon *R* :

$$v(k) = R * \Omega(\mathrm{k}) \iff \Omega(\mathrm{k}) = \frac{v(k)}{R} \tag{3.9}$$

En remplacer l'équation (3.9) dans les équations (3.7) et (3.8), les équations finales seront comme suit :

$$I_a(k+1) = \left(1 - \frac{R_a}{L_a} * T_s\right) * I_a(k) - \frac{K_e}{L_a} * T_s * \frac{v(k)}{R}\ + \frac{Vd_0}{L_a} * T_s\ * \alpha(k) \tag{3.10}$$

$$v(k+1) = R * \frac{K_m}{J} * T_s * I_a\,(k) + \left(1 - \frac{f}{J} * T_s\right) * v(k) \tag{3.11}$$

Il faut savoir que la vitesse de ce genre de moteur est très élevée ($\Omega(\mathrm{k}) = 170\ rad/s$). Par conséquent, l'utilisation d'un réducteur ($v(k) = R_{ed} * v_{Red}(k)$) ($R_{ed}$ coefficient de réduction) est primordiale, les équations (3.10) et (3.11) deviennent :

$$I_a(k+1) = \left(1 - \frac{R_a}{L_a} * T_s\right) * I_a(k) - \frac{K_e}{L_a} * T_s * \frac{R_{ed}}{R} * v_{Red}(k) + \frac{Vd_0}{L_a} * T_s\ * \alpha(k) \tag{3.12}$$

$$v_{Red}(k+1) = \frac{R}{R_{ed}} * \frac{K_m}{J} * T_s * I_a(k) + \left(1 - \frac{f}{J} * T_s\right) * v_{Red}(k) \qquad (3.13)$$

Pour finaliser notre modèle d'état, il nous reste à introduire l'équation du déplacement. Sachant que l'équation de la vitesse dans le domaine discret est comme suit :

$$v_{Red}(t) = \frac{Dep(k+1) - Dep(k)}{Ts} \qquad (3.14)$$

En ajoutant l'équation (14) avec les deux équations (12) et (13), on obtient notre modèle final suivant :

$$\begin{cases} I_a(k+1) = \left(1 - \frac{R_a}{L_a} * T_s\right) * I_a(k) - \frac{K_e}{L_a} * T_s * \frac{R_{ed}}{R} * v_{Red}(k) + 0 * Dep(k) + \frac{Vd_0}{L_a} * T_s * \alpha(k) \\ v_{Red}(k+1) = \frac{R}{R_{ed}} * \frac{K_m}{J} * T_s * I_a(k) + \left(1 - \frac{f}{J} * T_s\right) * v_{Red}(k) + 0 * Dep(k) \\ Dep(k+1) = 0 * I_a(k) + T_s * v_{Red}(k) + Dep(k) \end{cases} \qquad (3.15)$$

III.2.2.2 Exécution du modèle mathématique du MCC sous Matlab :

Comme le montre la figure suivante où est représentée le modèle d'état exécuté sous Matlab/Script, les graphes (Figure III.40, 41et 42) obtenus (comparés avec ceux de Simulink Figure III.36, 37 et 38 respectivement) montrent que notre modèle est exact et peut être maintenant implémenté au niveau de notre API.

MCC_BO_Vitesse.m

```
clc; clear all; close all;
Ts=0.001; Vd0 = 311;
Ra = 2.581; La = 0.028; % Ra(ohm): Resistance Induit , La(H): Inductance induit
Ke = 1.8;               % Ke(V/(rad/s)): Constante electrique  Ke = 0.1885 V/rpm  (rpm = tr/mn)
Km = Ke; J = 0.02215;   % Km(kg.m^2/(s*A): Constante mecanique , J(kg.m^2): moment d'inertie
f = 0.002953;           % f(N.M.s):Coefficient de frottement
alpha = 1;  % Rapport cyclique
u = alpha;  % Commande de notre processus
R = 25;     % Rayon poulie (mm)
Red = 10;   % Rapport de reduction
x=[0;0;0];  % x=[Ia;Vr;Dep]    Ia(A): Courant induit, Vred(mm/s): Vitesse linéaire réduite , Dep(mm): Deplacement
A = [1-(Ra/La)*Ts, (-Ke/La)*(Red/R)*Ts, 0; (R/Red)*(Km/J)*Ts, 1+(f/J)*Ts , 0; 0, Ts, 1];
B = [(Vd0/La)*Ts;0;0];
C = [1 0 0;0 1 0;0 0 1]; D = 0;
t_simu=1;       % t_simu(s): Temps de simulation
nb=t_simu/Ts;   % nb: Nombre d'iteration
yv=[];
tv=[];
```

```
for i=1:nb
        t = (i-1)*Ts;
        x = A*x+B*u;
       Ia = C(1,:)*x;
     Vred = C(2,:)*x;
      Dep = C(3,:)*x;
        y = [Ia;Vred;Dep];
       yv = [yv y];
       tv = [tv; t];
end
figure(1)
    plot(tv,yv(1,:))
figure (2)
    plot(tv,yv(2,:))
figure (3)
    plot(tv,yv(3,:))
```

Figure III. 38 : Modèle d'état développé sous Matlab/Script.

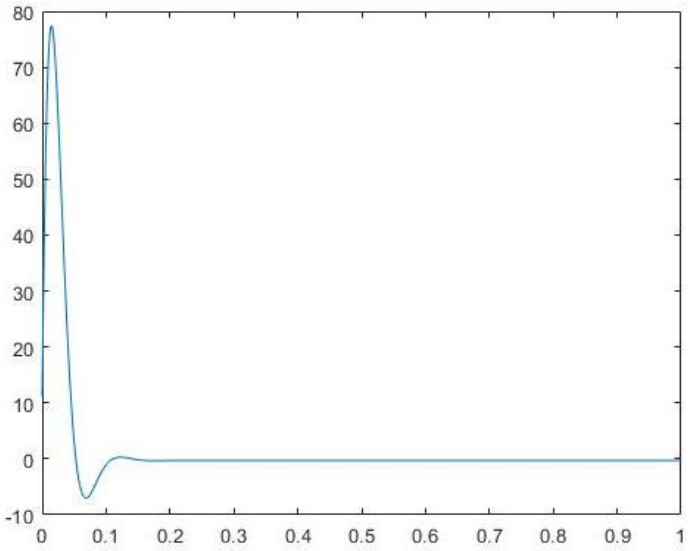


Figure III. 39 : Allure du courant.

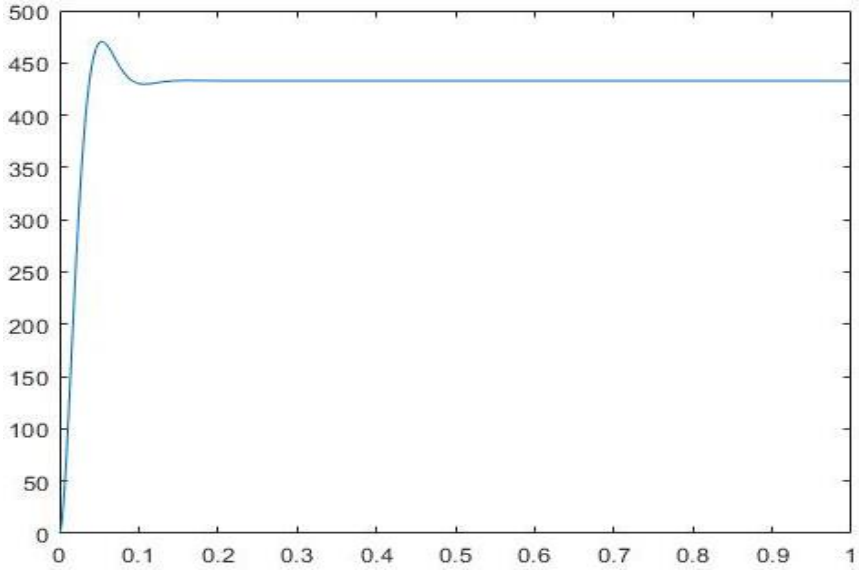


Figure III. 40 : Allure de la vitesse linéaire réduite

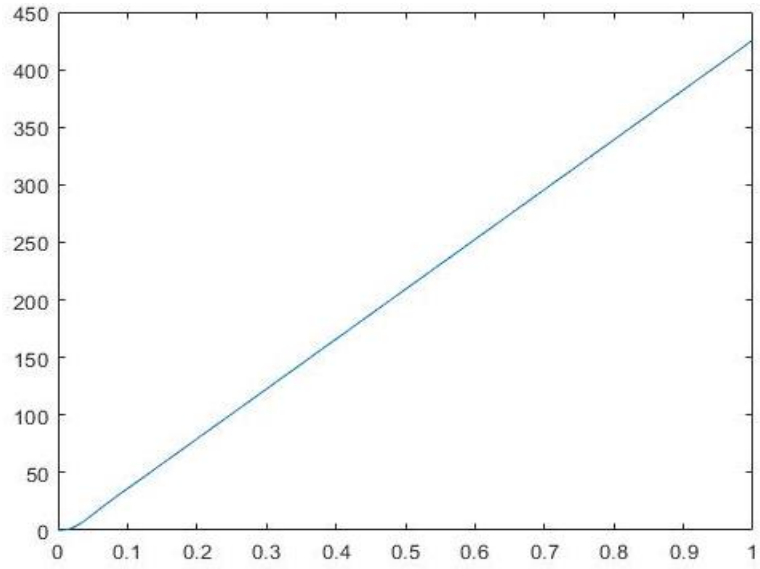


Figure III. 41 : Allure du déplacement

III.2.3 Implémentation du modèle au niveau de notre API

Sachant que c'est nouveau pour nous et pour nos camarades, nous avons voulu partager nos connaissances acquises en montrant toutes les étapes suivies jusqu'à la validation du modèle en s'inspirant directement des différentes captures d'écrans.

Pour arriver à valider notre modèle, il nous a fallu passer par trois étapes principales qui sont :

- **Etape principale 1** : Création et attribution d'un nom au projet, Ajout d'appareils et établissement de communication,
- **Etape principale 2** : Création d'IHM ou en d'autres termes de notre simulateur sous WinCC. Dans cette étape, on représente l'IHM destinée pour la validation du modèle,
- **Etape principale 3** : Développement de notre programme d'API pour la commande et d'animation de notre simulateur sous Step 7.

Il faut souligner que chaque étape principale comporte un ensemble d'étape représentée sous forme de capture (image).

III.2.3.1 Description détaillée des différentes étapes principales

- Etape principale 1 : Création et attribution d'un nom au projet, Ajout d'appareils et établissement de communication,

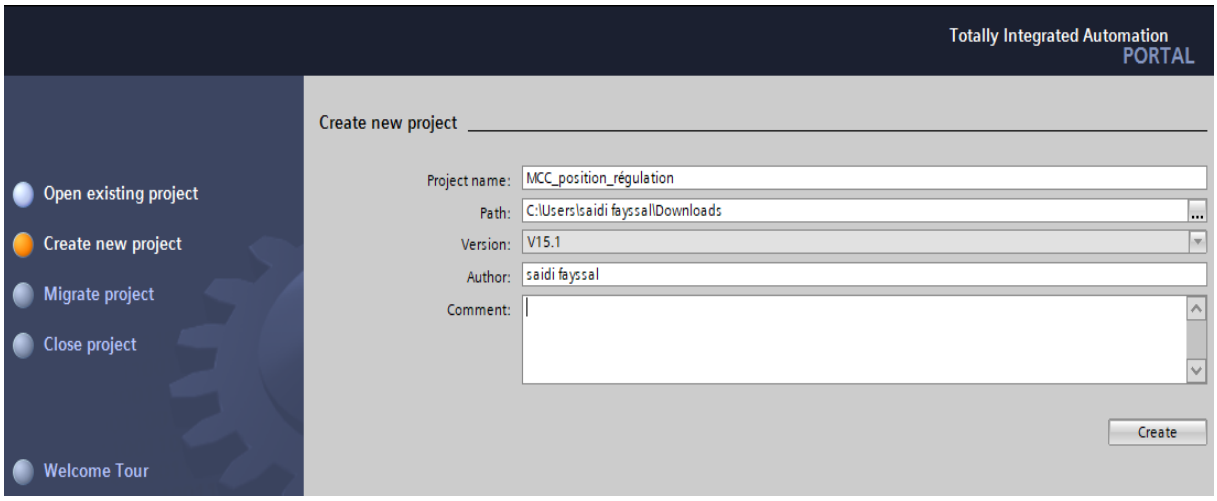


Figure III. 42 : Etape1 : Création et attribution du nom au projet.

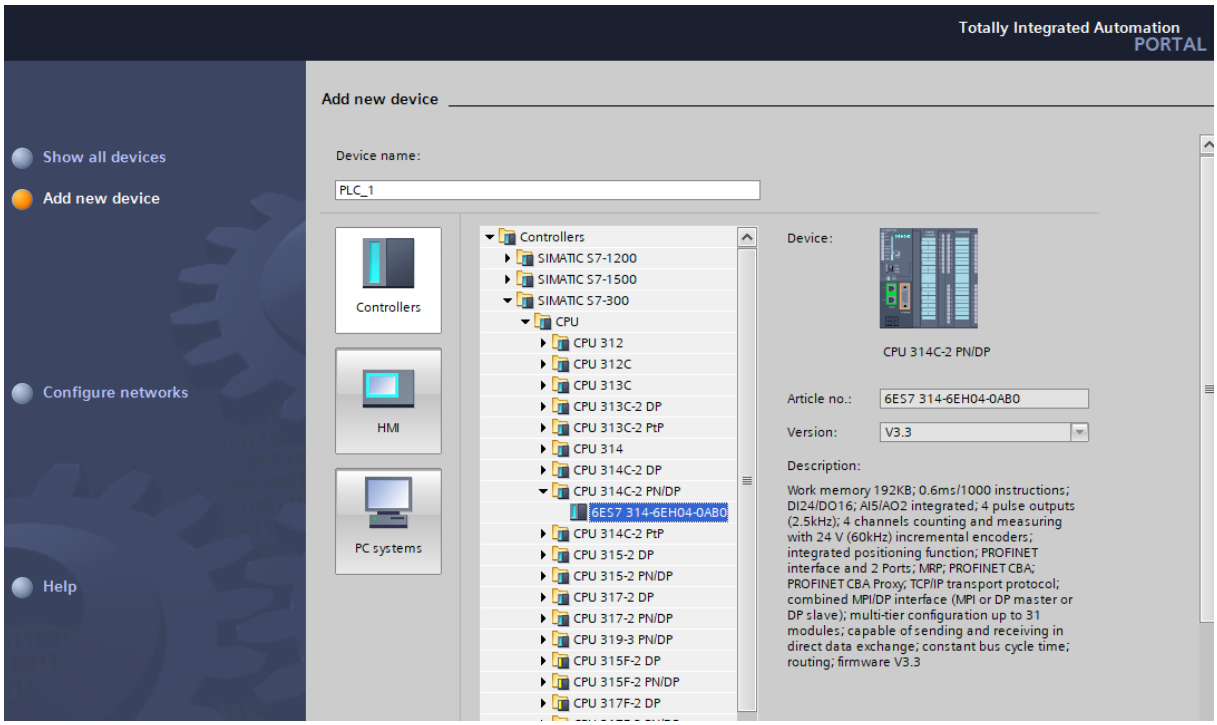


Figure III. 43 : Etape2 : Ajout d'appareil : Contrôleur API S7-300 avec sa CPU 314C-2PN/DP.

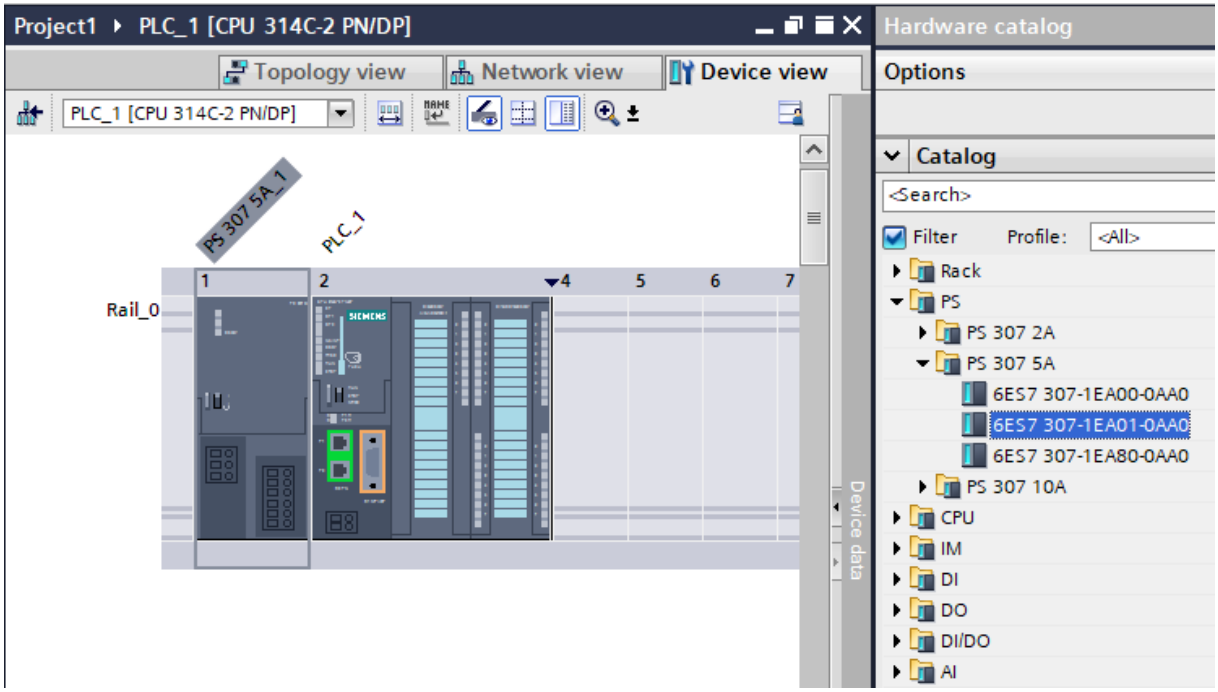


Figure III. 44 : Etape3 : Ajout d'alimentation pour API (PS : 307 5A) 6ES7 3-1EA01-0AA0.

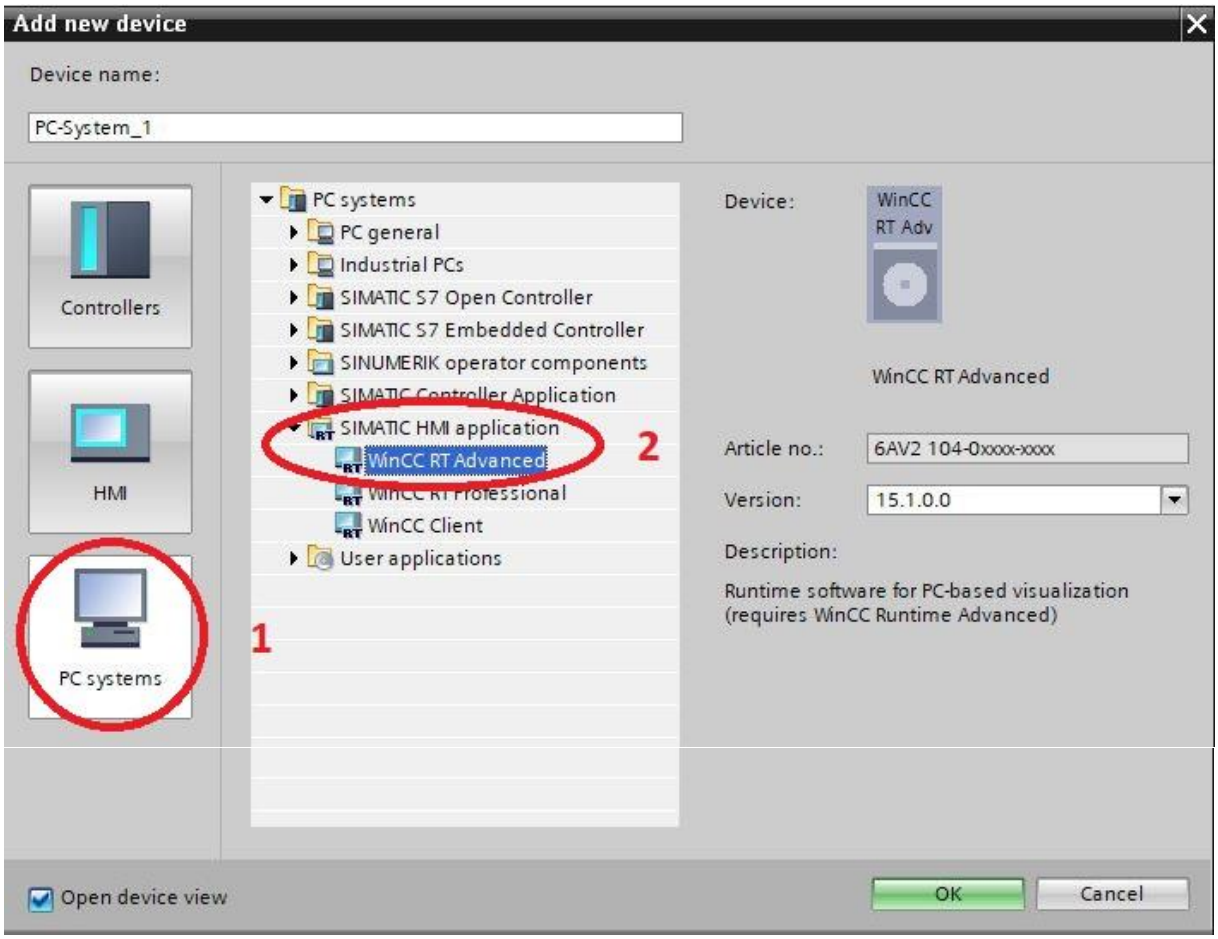


Figure III. 45 : Etape4 : Ajout du PC system pour la création d'IHM avec WinCC RT Advanced.

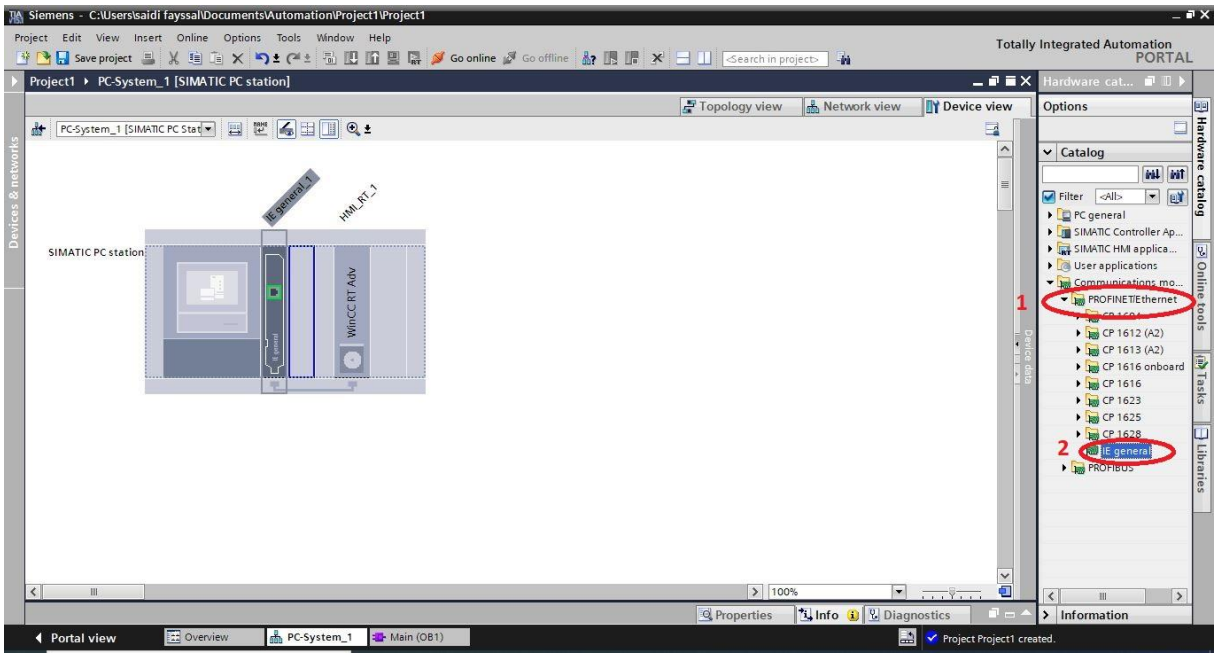


Figure III. 46 : Etape5 : Ajout d'une carte réseau à notre PC system.

Après établissement de la communication entre le Pc-Système et contrôleur, il faut mieux afficher les adresses IP pour vérification comme le montre la figure ci-dessous. Dans notre cas, c'est le réseau Profinet qui est adopté.

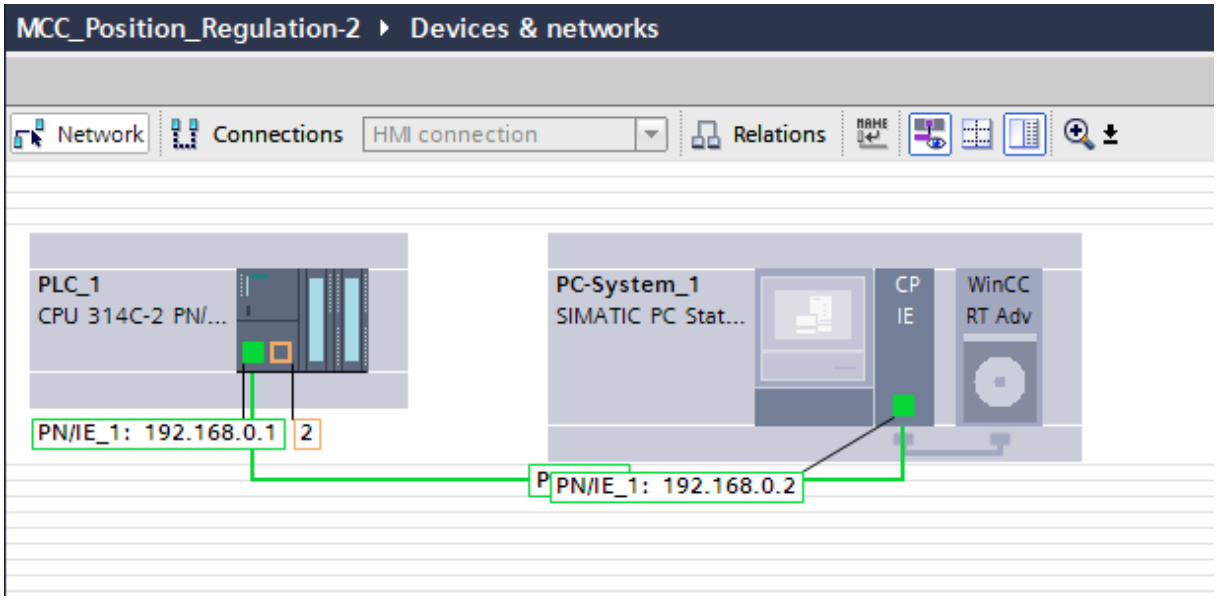


Figure III. 47 : Affichage des adresses IP de chaque appareil pour vérifier et éminier tout conflit IP.

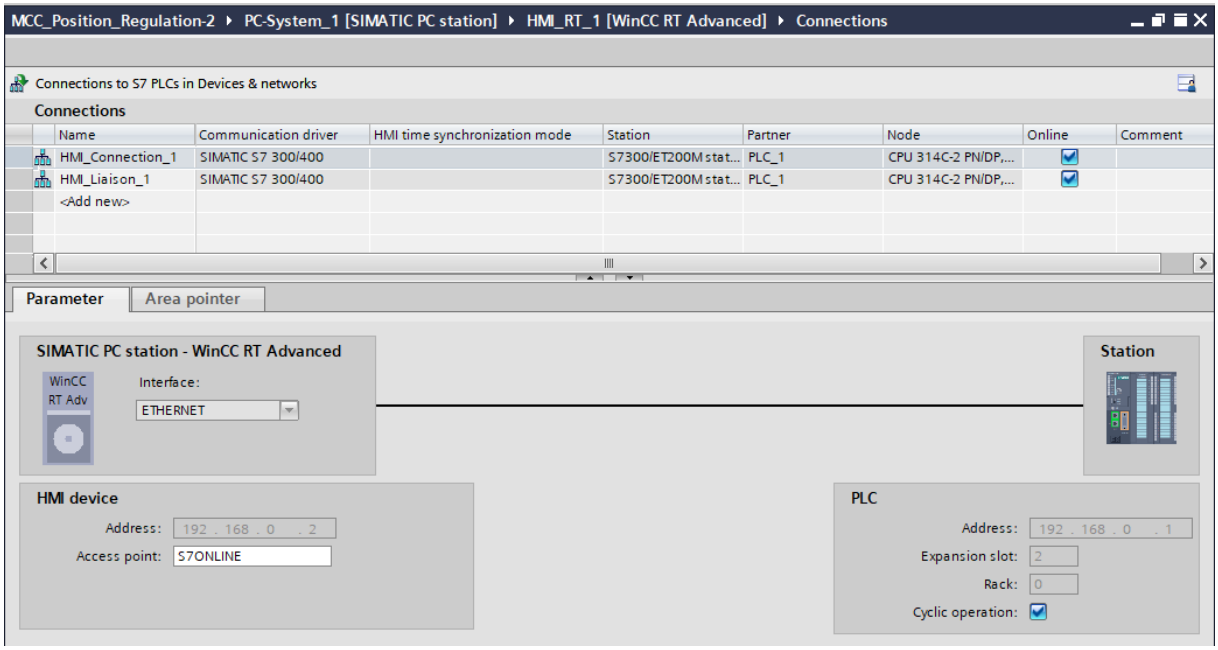


Figure III. 48 : Etape6 : Etablissement de la communication Profinet entre les 02 appareils.

- **Etape principale 2 : Création d'IHM sous WinCC.**

Le logiciel WinCC de la firme Siemens est un logiciel très puissant et très utilisé pour la création d'IHM pour les systèmes industriels. Il compotre une bibliothèque très riche en objet graphique comme par exemple : bouton, afficheur, slider, trace de figure. La figure ci-dessous comporte les principaux objets utilisés pour la conception de notre simulateur.

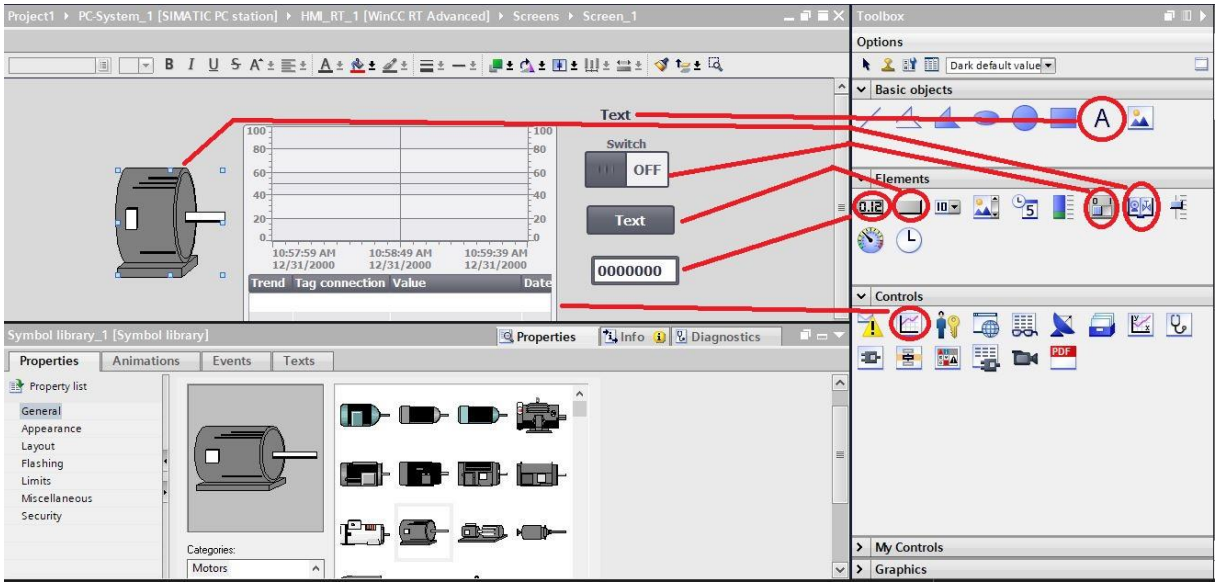


Figure III. 49 : Etape 1 : Dépôt des différents objets.

Il faut souligner que chaque objet doit avoir un attribut. La plus part du temps, les objets insérés sont liés aux variables d'API qui se trouvent au niveau de la table des mnémoniques (on discutera de ça dans l'étape principale 3).

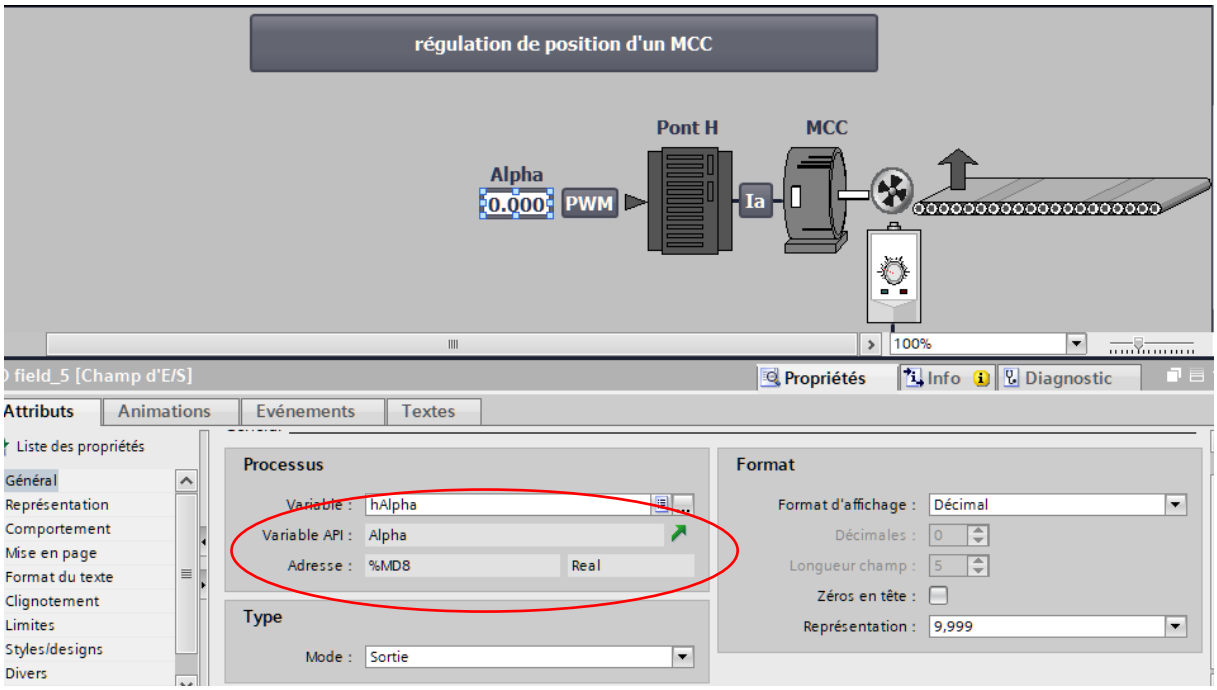
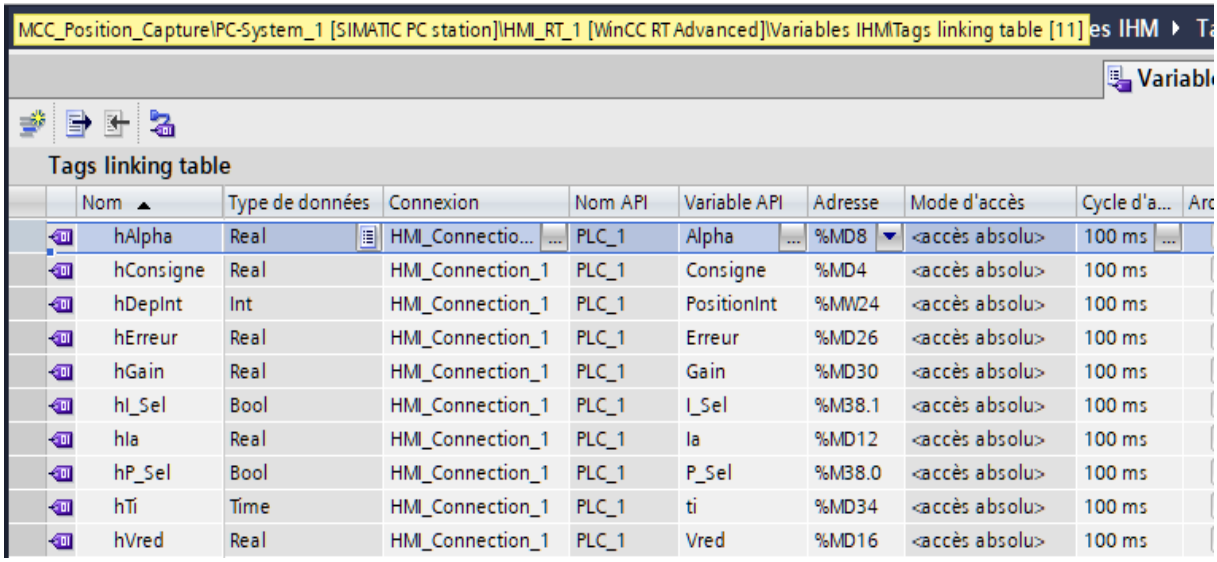


Figure III. 50 : Etape 2 : Liaison entre les variables IHM « Nom » avec celles d'API « Variable API ».

Pour le cas de notre simulateur et dans le cas du WinCC, les objets liés aux variables sont représentés dans une table spéciale, donné par la figure ci-dessous.



Nom	Type de données	Connexion	Nom API	Variable API	Adresse	Mode d'accès	Cycle d'a...
hAlpha	Real	HMI_Connectio...	PLC_1	Alpha	%MD8	<accès absolu>	100 ms
hConsigne	Real	HMI_Connection_1	PLC_1	Consigne	%MD4	<accès absolu>	100 ms
hDepInt	Int	HMI_Connection_1	PLC_1	PositionInt	%MW24	<accès absolu>	100 ms
hErreur	Real	HMI_Connection_1	PLC_1	Erreur	%MD26	<accès absolu>	100 ms
hGain	Real	HMI_Connection_1	PLC_1	Gain	%MD30	<accès absolu>	100 ms
hI_Sel	Bool	HMI_Connection_1	PLC_1	I_Sel	%M38.1	<accès absolu>	100 ms
hIa	Real	HMI_Connection_1	PLC_1	Ia	%MD12	<accès absolu>	100 ms
hP_Sel	Bool	HMI_Connection_1	PLC_1	P_Sel	%M38.0	<accès absolu>	100 ms
hTi	Time	HMI_Connection_1	PLC_1	ti	%MD34	<accès absolu>	100 ms
hVred	Real	HMI_Connection_1	PLC_1	Vred	%MD16	<accès absolu>	100 ms

Figure III. 51 : Table des variables de notre simulateur.

Le but principal de notre simulateur de base donné par la figure ci-dessous est de nous permettre de visualiser le déplacement de la flèche qui se trouve sur le convoyeur. En fur et à mesure que le déplacement s'effectue, l'évolution dans le temps des 03 variables d'état s'affichent sur les graphes au niveau du simulateur.

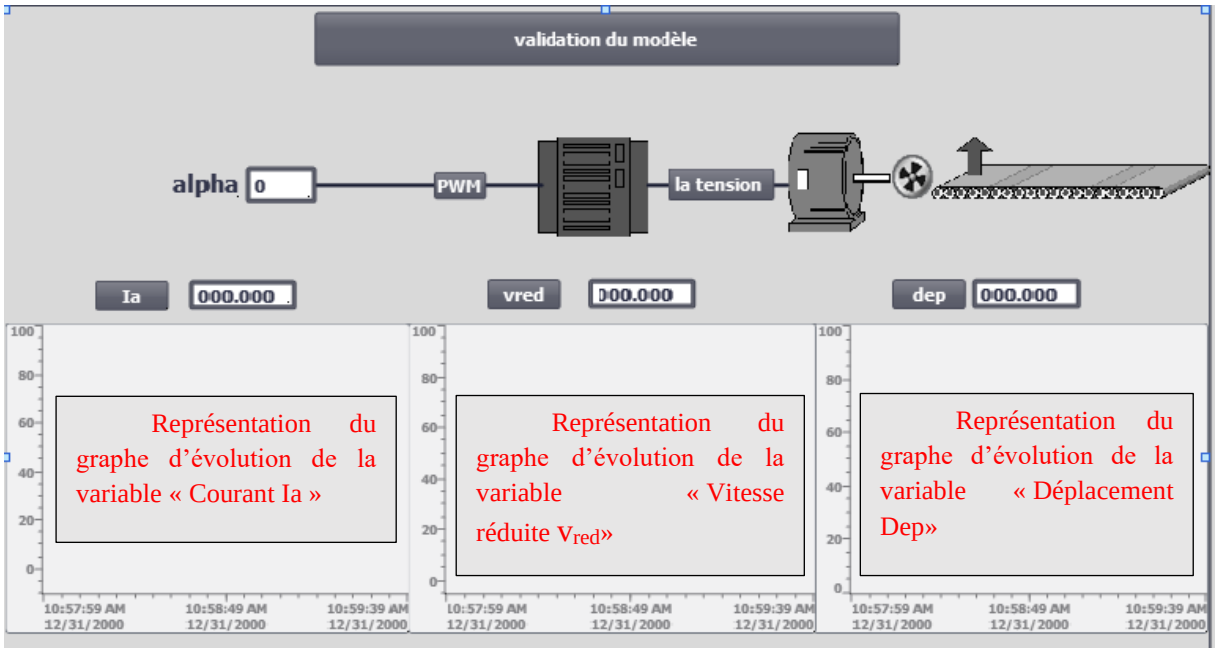
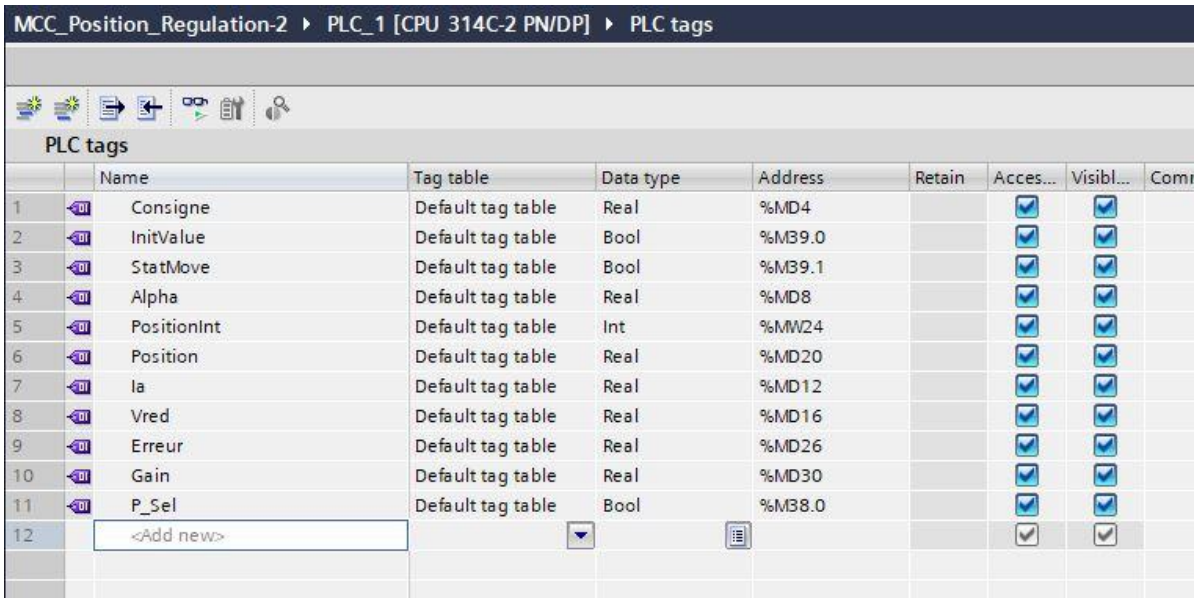


Figure III. 52 : Etape 2 : Réarrangement des différents objets suivant le simulateur désiré.

- **Etape principale 3 : Développement de notre programme d'API pour la commande et animation de notre simulateur sous Step7.**

Avant de commencer à développer notre programme sur le logiciel Step 7, la première étape est le remplissage de la table des mnémoniques (table des variable API). Dans notre cas, tous les variables utilisées sont représentés dans la figure ci-dessous qui correspond à notre table des mnémoniques.



	Name	Tag table	Data type	Address	Retain	Acces...	Visibl...	Comr
1	Consigne	Default tag table	Real	%MD4		☑	☑	
2	InitValue	Default tag table	Bool	%M39.0		☑	☑	
3	StatMove	Default tag table	Bool	%M39.1		☑	☑	
4	Alpha	Default tag table	Real	%MD8		☑	☑	
5	PositionInt	Default tag table	Int	%MW24		☑	☑	
6	Position	Default tag table	Real	%MD20		☑	☑	
7	Ia	Default tag table	Real	%MD12		☑	☑	
8	Vred	Default tag table	Real	%MD16		☑	☑	
9	Erreur	Default tag table	Real	%MD26		☑	☑	
10	Gain	Default tag table	Real	%MD30		☑	☑	
11	P_Sel	Default tag table	Bool	%M38.0		☑	☑	
12	<Add new>					☑	☑	

Figure III. 53 : Etape 1 : Remplissage de la table des mnémoniques.

L'étape suivante est l'introduction des fonctions pour développer notre programme. En intégrant une fonction, vous devez sélectionner le langage de programmation. Dans notre cas, on a utilisé le langage évolué SCL (Figure III.53) pour implémenter notre programme (Figure III.54) et le langage « Ladder » pour lancer et initialiser (Figure III.55) notre simulateur.

SCL (Structured Control Language) est un langage de programmation évolué apparenté au langage PASCAL qui permet une programmation structurée.

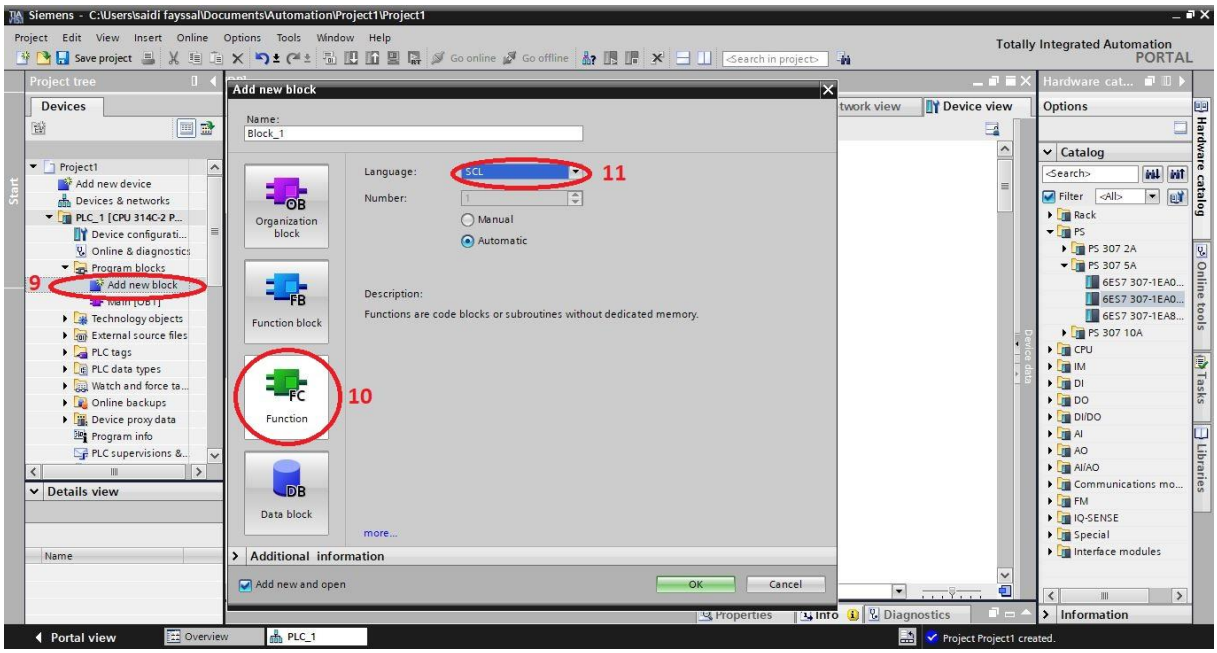


Figure III. 54 : Etape 2 : Ajout des différentes fonctions.

```
IF...  CASE... OF...  FOR... TO DO..  WHILE.. DO...  (*...*)  REGION

1  #Ia := (1-(#Ra/#La)*#Ts)*#Ia + ((-#Ke/#La)*(#Red/#R)*#Ts)*#Vred + ((#Vd0/#La)*#Ts)*1;
2  #Vred := ((#R/#Red)*(#Km/#J)*#Ts)*#Ia + (1-(#f/#J)*#Ts)*#Vred;
3  #Dep := #Ts * #Vred + #Dep;
4  #DepInt := REAL_TO_INT(#Dep);
```

Figure III. 55 : Etape 3 : Implémentation de notre modèle d'état avec le langage « SCL » .

MCC_Modelling

	Name	Data type	Offset	Default value	Comment
1	Input				
2	alpha	Real			
3	Output				
4	DepInt	Int			
5	InOut				
6	Ia	Real			
7	Vred	Real			
8	Dep	Real			
9	Temp				
10	<Add new>				
11	Constant				
12	Ts	Real		0.001	
13	Vd0	Int		311	
14	Ra	Real		2.581	
15	La	Real		0.028	
16	Ke	Real		1.8	
17	Km	Real		1.8	
18	J	Real		0.02215	
19	f	Real		0.002953	
20	R	Real		25.0	
21	Red	Real		10.0	
22	Return				
23	MCC_Modelling	Void			

Figure III. 56 : Table des paramètres de la fonction « modèle d'état du MCC ».

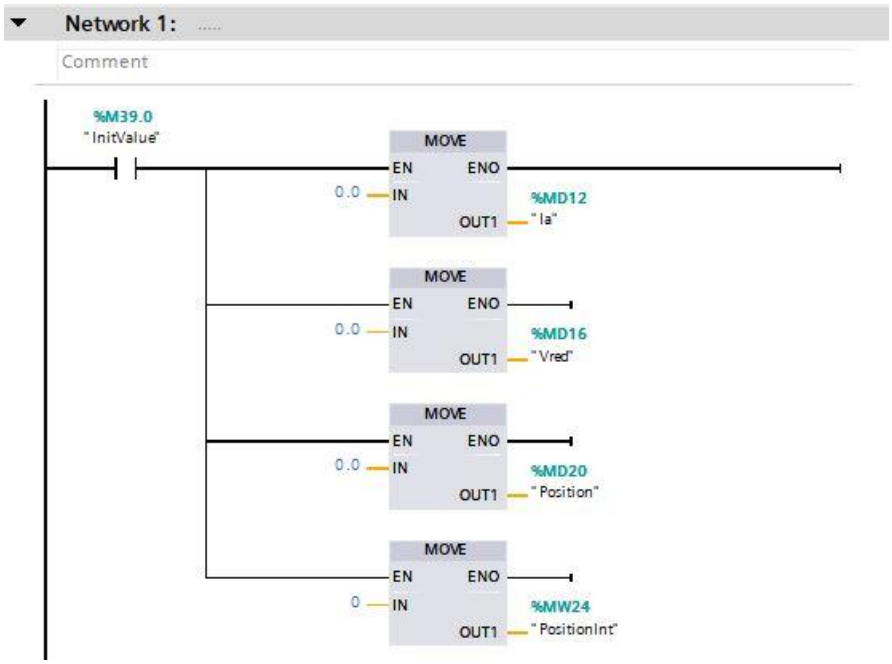


Figure III. 57 : Etape 4 : Initialisation des variables d'état. Programme développé par le langage « Ladder ».

L'étape suivante est cruciale qui consiste à l'exécution de ces fonctions. Il faut que chaque fonction doit être déposé dans un « OB » (Bloc Organisationnel) pour être exécutée.

Il faut souligner qu'en créant un projet et en insérant un contrôleur, le Tia Portal crée automatiquement un OB « main ». Ce bloc sera utilisé généralement pour lancer, initialiser et calculer des valeurs qui ne nécessite pas une exécution ou un calcul périodique.

Par contre, si la détermination des valeurs d'une variable demande un calcul période (détermination des valeurs de nos variables d'état sont périodiques « Ts : temps d'échantillonnage»), nous serons contraints d'utiliser un « OB cyclique ». Pour notre cas, on a déposé notre fonction « Modèle d'état du MCC » dans un OB35 (Figure III) où son temps d'exécution périodique est de 100ms.

Arriver à ce niveau, on peut estimer que notre programme est achevé et les différents graphes peuvent obtenu au niveau de notre IHM (Figure III.52).

Après exécution du programme au niveau du PLC-Sim, les graphes obtenus (Figue III.59, 60et 61) prouvent que l'implémentation du modèle dans l'API donnent des résultats très satisfaisantes et identiques à ceux obtenus par l'outil de calcul et de simulation très puissant qui est le Matlab. Nous sommes fiers de dire que notre objectif est atteint et on peut continuer à progresser dans ce contexte en ajoutant la régulation industrielle et particulièrement l'objet technologique fourni par le contrôleur API S7-300.

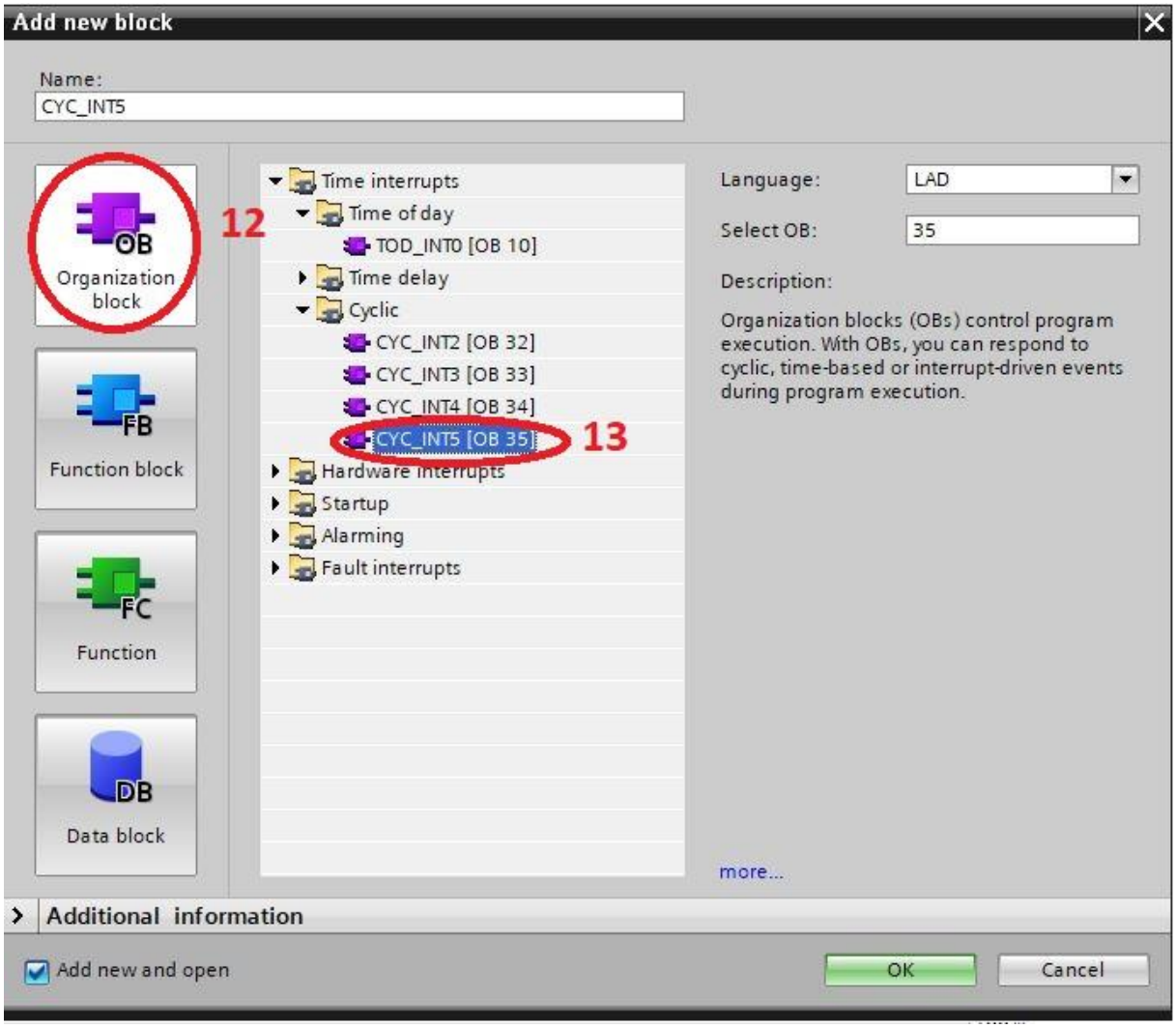


Figure III. 58 : Etape 5 : Ajout d'un OB Cyclique « OB 35 » permettant l'exécution de notre fonction.

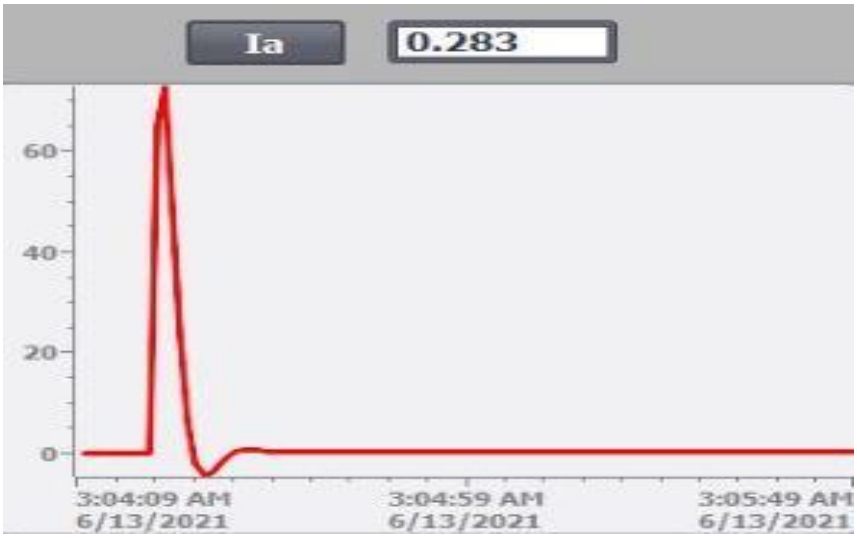


Figure III. 59 : Evolution du courant obtenu au niveau d'API.

Figure III. 60 : Evolution de la variable vitesse réduite obtenu au niveau d'API.

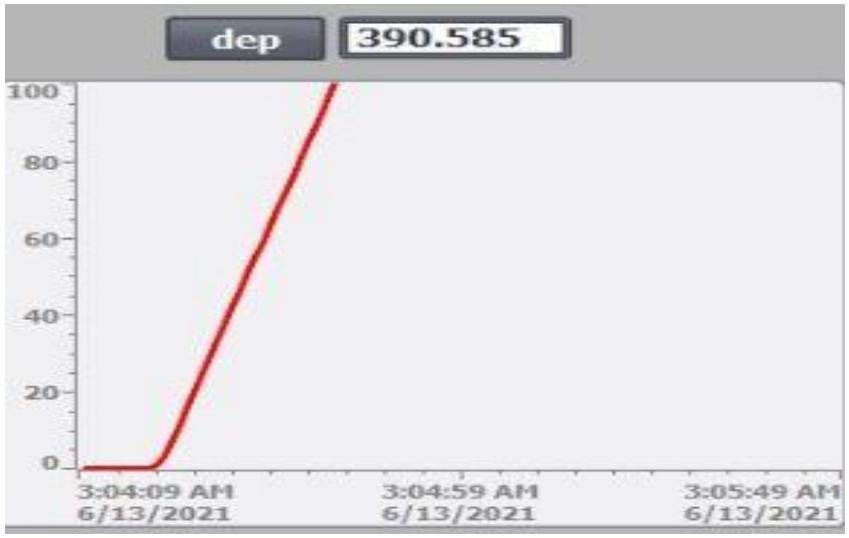


Figure III. 61 : Evolution de la variable déplacement obtenu au niveau d'API.

Remarque : avant de sauvegarder n'importe quel graphe, il faut inialiser le système puis le lancer. En validant par l'activation de l'initialisation, la valeur zéro est transférer à chaque variable d'état.

III.3 Régulation de position :

Pour continuer dans le même contexte et faciliter la compréhension de notre manuscrit, nous allons commencer par l'implémentation de la régulation sous API en intégrant un objet technologique « PID Control », puis on finira par la validation des résultats obtenus en les comparant à ceux obtenus par Matlab.

III.3.1 Régulation sous API avec intégration d'objet technologique « PID control »

- **Développement de notre programme d'API**

L'objet technologique est ajouté à notre projet comme le montre la figure III.62. Un description détaillée sera représentée au niveau de l'annexe.

Il faut savoir que ce type d'objet doit être déposé au niveau d'un OB cyclique. Pour notre cas, on l'a déposé dans le même OB cyclique OB 35 où est exécuté notre fonction « Modèle d'état du MCC » (Figure III.64).

Pour le cas de notre objet technologique, les trois paramètres importants sont : « Gain, Ti , TD et trois grandeurs : la consigne, la position (position actuelle) et la sortie de PID Control (LMN) qui correspond au rapport cyclique « Alpha » (Figure III.63)

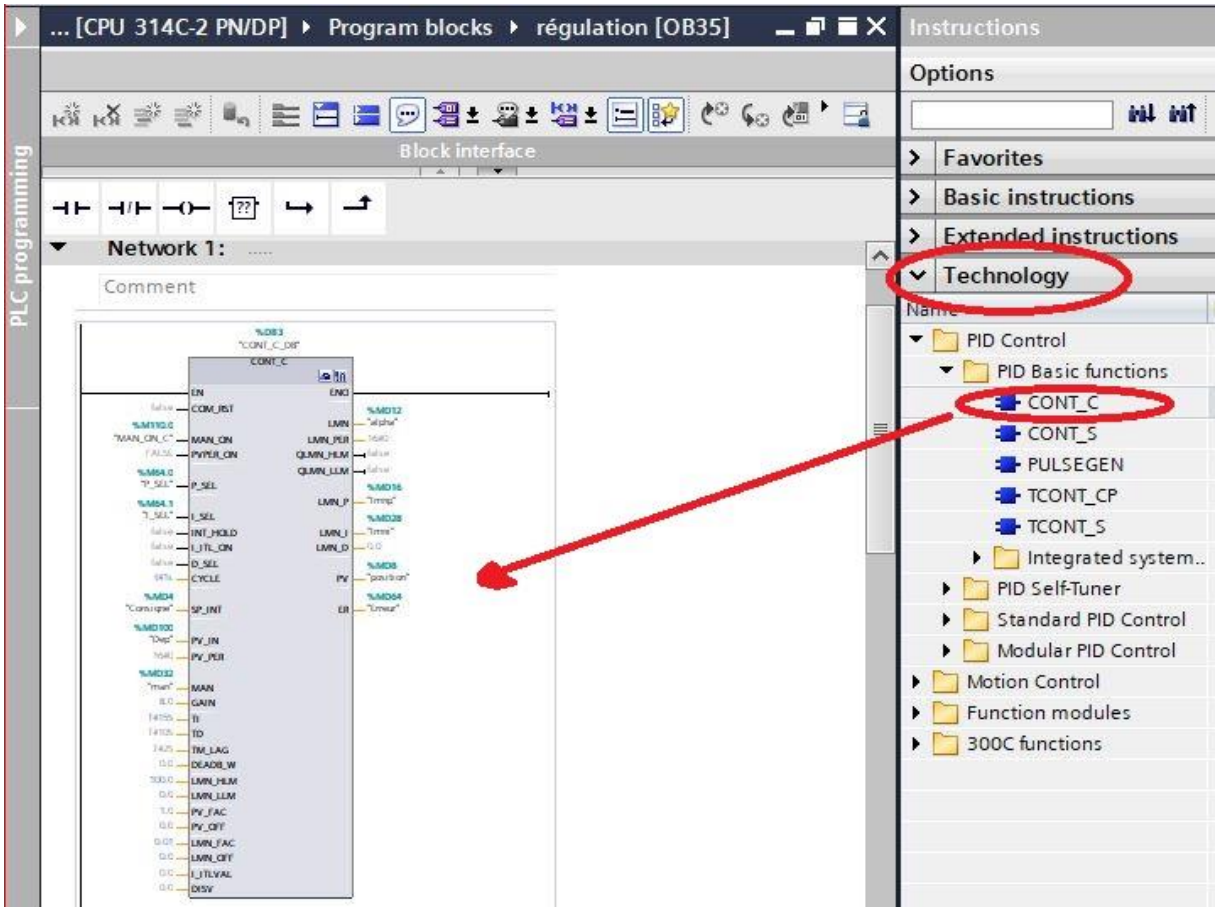


Figure III. 62 : Ajout et dépôt au niveau d'OB cyclique OB35 d'objet technologique PID Control.

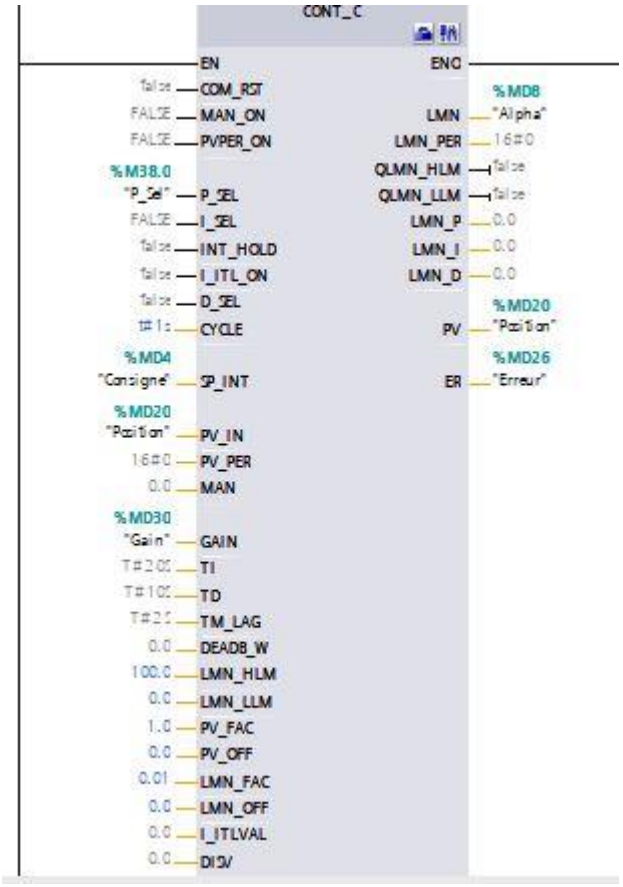


Figure III. 63 : Vue d'ensemble d'objet technologique avec différents paramètres et grandeurs.

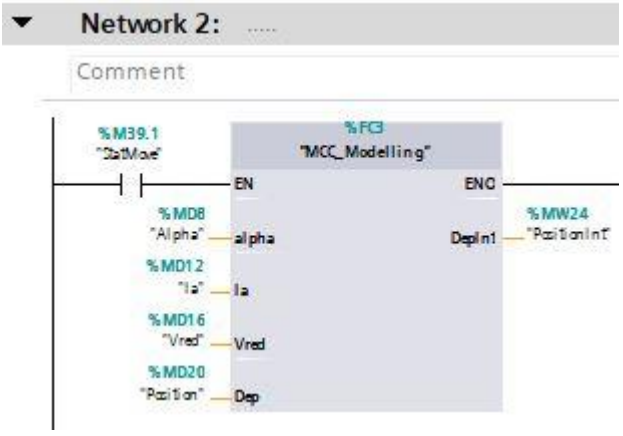


Figure III. 64 : Dépôt de la fonction 'Modèle d'état du MCC' au niveau de l'OB 35.

- **Développement et finalisation de notre simulateur**

Notre simulateur final est représenté par la figure III.66 qui correspond ou qui est identique à une boucle fermée pour la régulation de position. Notre processus correspond à la flèche qui se déplace sur un convoyeur. Le but est de régler les paramètres de note objet technologique de tel façon que la flèche atteint la position désirée (consigne) sans oscillation.

Après initialisation et lancement de notre simulateur, l'évolution dans le temps de nos variables d'état est représentée dans les figures III.67, 68, 69, 70. Ces graphes seront comparés à ceux obtenus par Matlab pour valider nos résultats et prouver l'exactitude de notre simulateur.

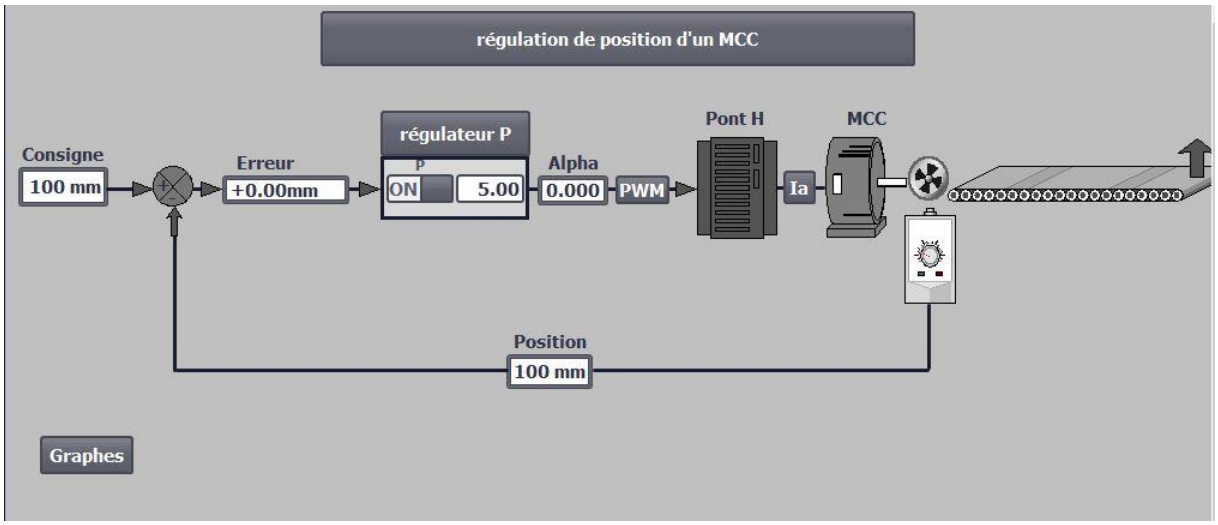


Figure III. 65 : Simulateur complet représentant la régulation de position d'un MCC

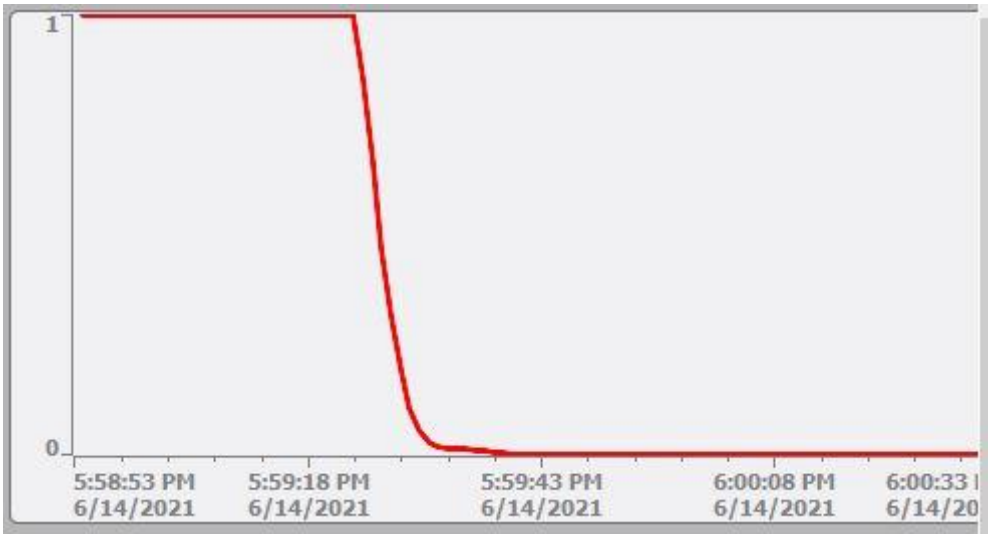


Figure III. 66 : Evolution dans le temps de la valeur du rapport cyclique 'Alpha'.

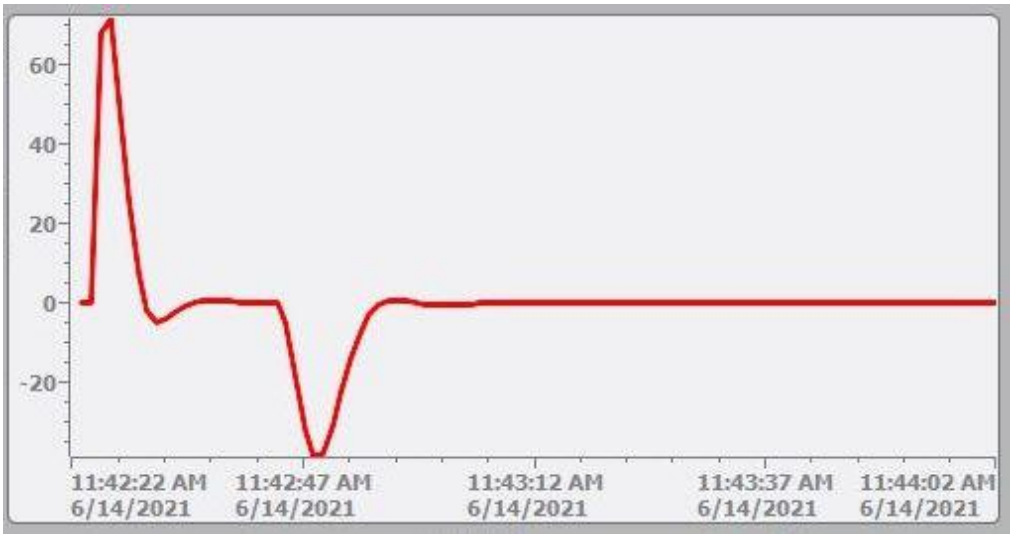


Figure III. 67 : Evolution dans le temps du courant Ia.

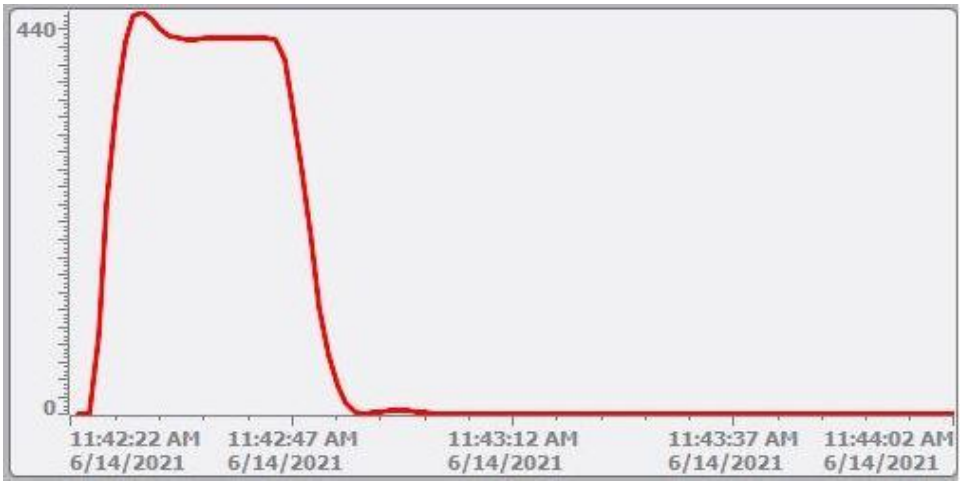


Figure III. 68 : Evolution dans le temps de la vitesse réduite.

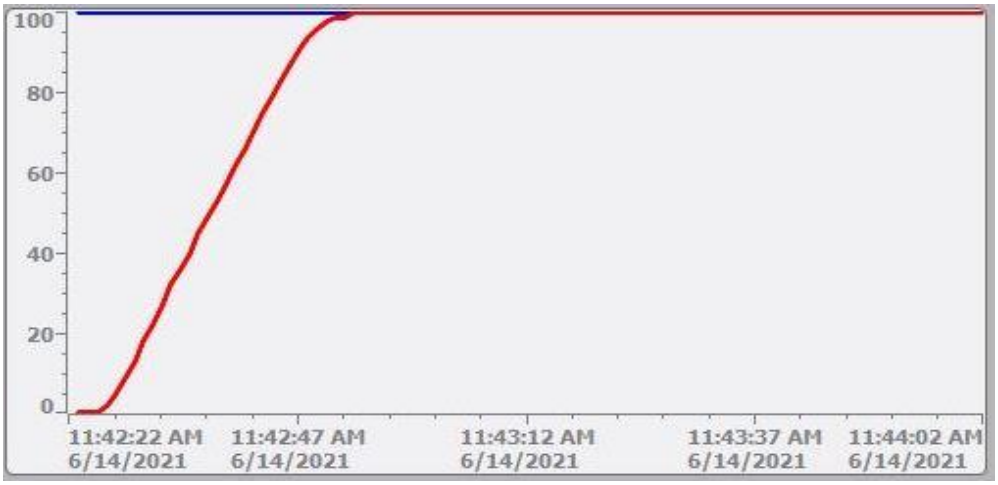


Figure III. 69 : Evolution dans le temps de la position.

III.3.2 Régulation de position sous Matlab/SimPower System et validation de nos résultats

La régulation par Simulink va nous permettre de valider nos résultats obtenus par l'intégration de notre objet technologique sous l'API en les comparant à ceux obtenus par Matlab/SimPower system.

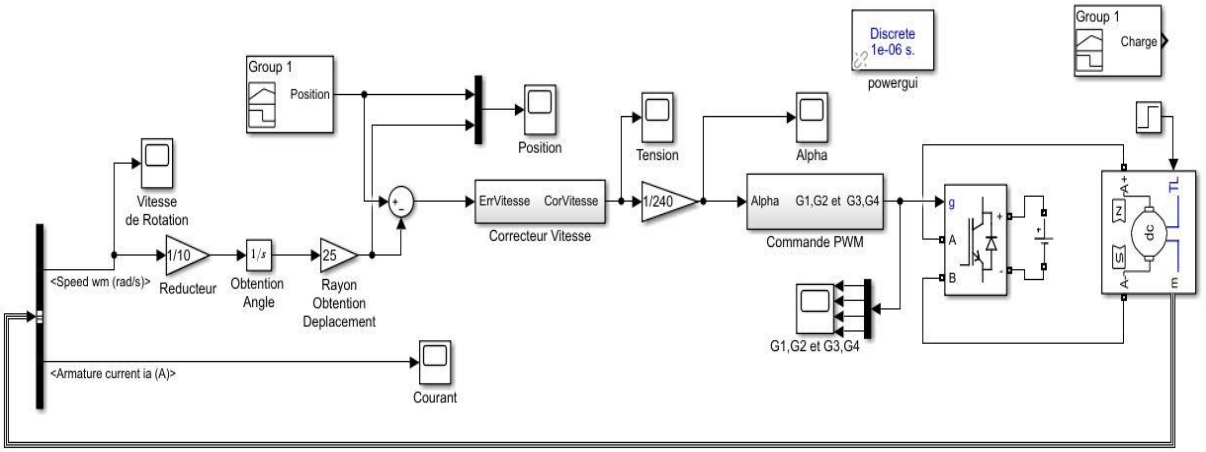


Figure III. 70 : Schéma bloc en boucle fermé de la régulation en position sous Matlab/SimPower System

Le schéma est divisé en deux parties, la 1 er partie est le passage de la vitesse angulaire à la vitesse linéaire, puis le passage de la vitesse linéaire au déplacement en un intégrant la vitesse linéaire. Connaissant maintenant la consigne qui est la position, en déterminant l'erreur entre la consigne et la position actuelle qui sera l'entrée de notre régulateur. La sortie de régulateur va nous permettre de fournir la valeur du rapport cyclique. Sachant que le pré-actionneur est un pont en H, donc la vitesse (ou tension) sera proportionnelle à la valeur du rapport cyclique 'Alpha'. En fur et à mesure le processus s'approche de la position désirée, la vitesse diminue jusqu'à l'arrêt complet du processus (Alpha =0).

Après simulation, les graphes trouvés (Figure III.72, 73, 74 et 75) montrent que les résultats obtenus par l'objet technologique sous API sont identiques ou mieux que ceux obtenus par Matlab.

On a constaté que la meilleure régulation est obtenue pour le cas proportionnel sans intégrateur. LA raison est que l'intégration de la vitesse pour l'obtention du déplacement prend la place de l'intégrateur de notre régulateur. Le courant négatif est légitime en raison de l'inertie du moteur choisi.

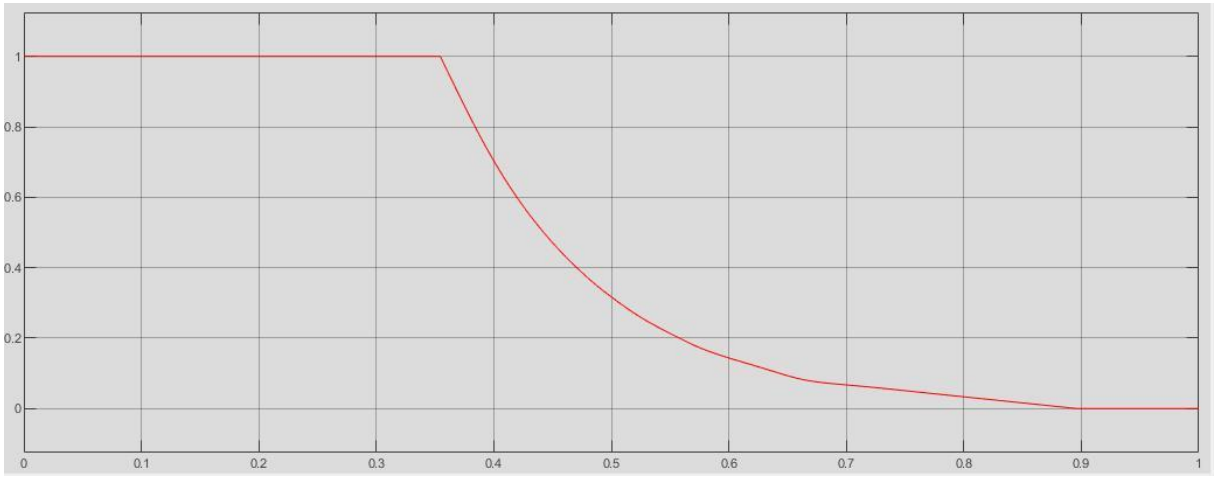


Figure III. 71 : Allure de la commande ‘Alpha’.

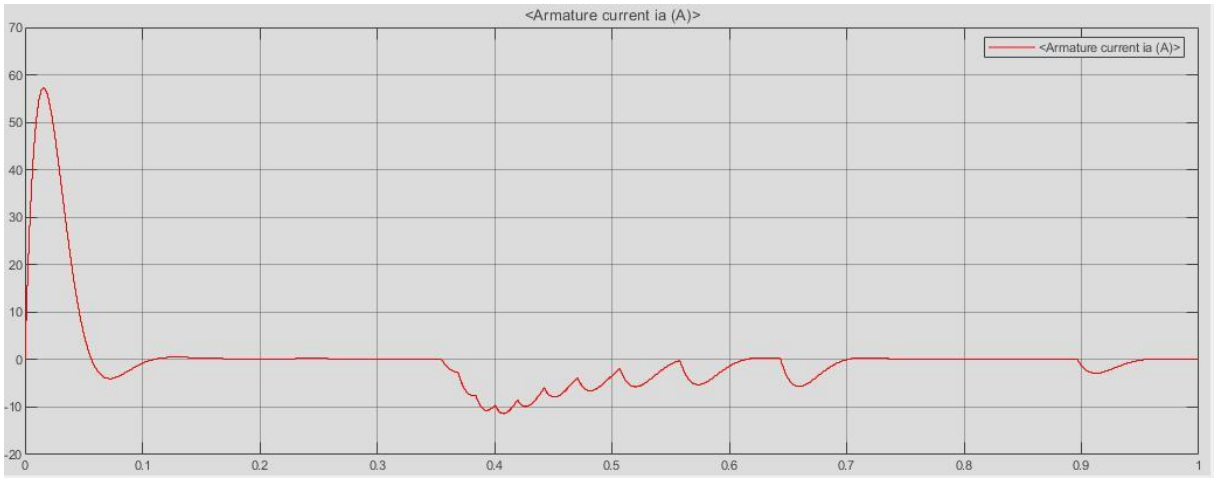


Figure III. 72 : Allure de courant.

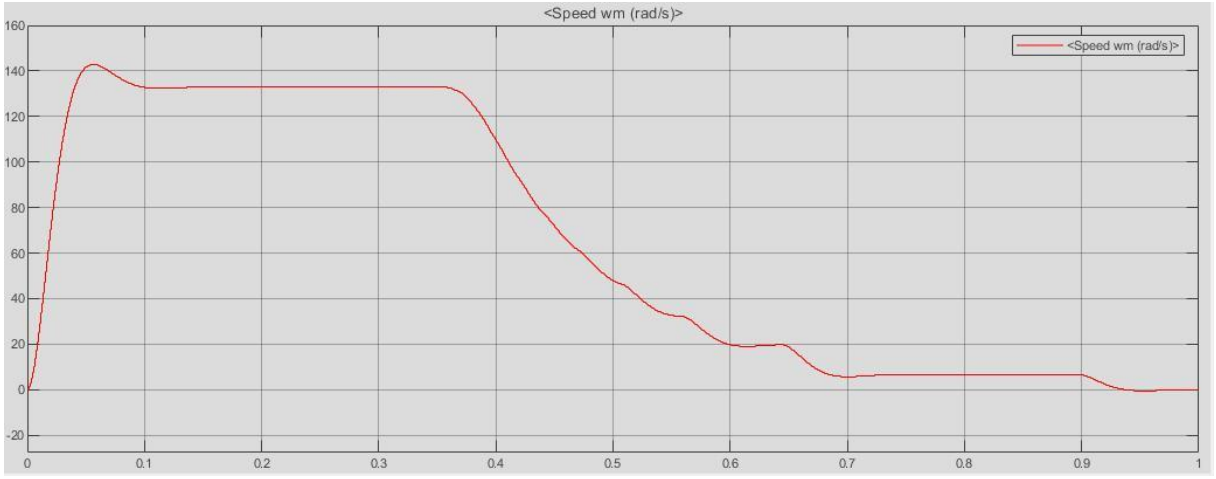


Figure III. 73 : Allure de la vitesse de rotation.

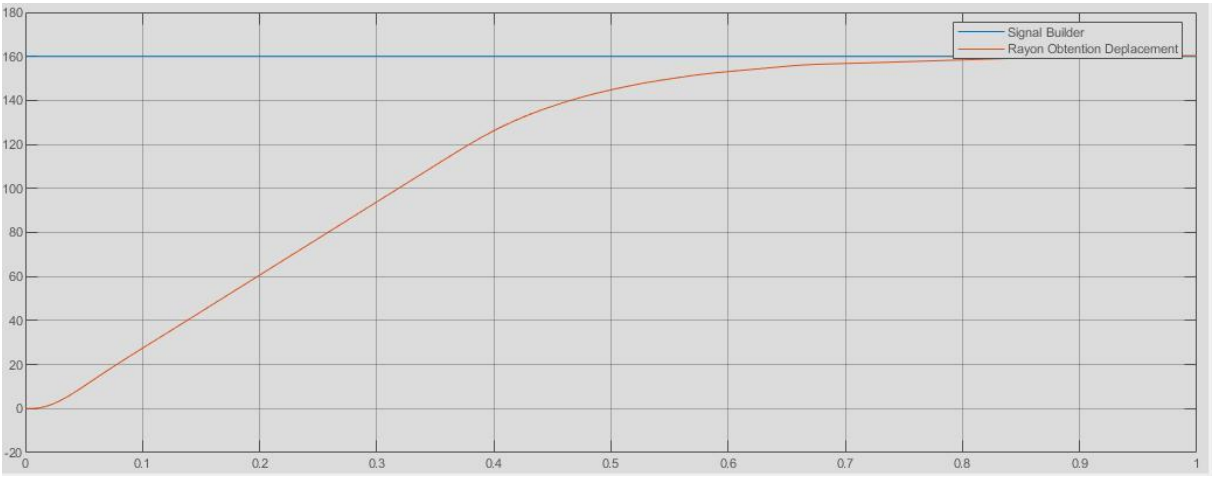


Figure III. 74 : Allure de la position.

III.4 Conclusion

A travers ce chapitre, nous avons montré que notre simulateur donne de très bon résultat et sera un outil puissant pour montrer à nos étudiants le rôle de la régulation industrielle par l'utilisation su contrôleur API S7-300. Il faut savoir, c'est ce contrôleur qui se trouve presque dans tous les processus industriels et dans les machines automatiques en particulier. Il faut souligner qu'avec ce type de simulateur, on n'a pas besoin de matériel. C'est la nouvelle. Ce type de technologie s'appelle 'Twin digital'. Il fait gagner beaucoup d'argent, réduit les incidents, améliore la rentabilité en faisant plusieurs tests sans bloquer ni la production.

Conclusion Générale

L’automatisation s’est généralisée à l’ensemble des activités de production, tant dans l’industrie, que dans les activités de services. Quel que soit son domaine d’application et les techniques auxquelles elle fait appel. L’automatisation s’est constamment développée dans l’unique but de réduire la pénibilité du travail humain et d’améliorer la productivité du travail.

Le travail que nous avons réalisé s’inscrit dans le cadre de l’étude de la régulation de position du moteur courant continu relie avec API S7-300 avec un superviseur HMI.

Nous avons développé un programme dans le logiciel STEP7 qui sera chargé dans l'automate programmable en vue de réguler la position d’un moteur à courant continu

Au terme de ce travail, nous avons acquis une très bonne expérience comme une nouvelle connaissance. Cette connaissance concernant le logiciel de supervision WinCC, la programmation des automates S7-300 et la régulation de position d’un moteur à courant continu.

L’objectif de notre travail consistait à utiliser le langage de programmation TIA Portal pour faire la régulation de position du moteur à courant continu

Le Matlab est un outil puissant et qu’on connait bien et qu’on l’a utilisé durant notre cursus. C’est pour cela que nous l’avons utilisé durant beaucoup car c’est le seul moyen ou c'est le meilleur moyen pour concrétiser et valider le résultat trouvé.

Enfin, nous sommes arrivés à développer et valider un outil qu’on trouve sera très utile pour les étudiants qui est le simulateur développé durant ce modeste travail utilisant comme cerveau de commande l’API S7-300 et WinCC comme logiciel de conception graphique. Les résultats sont très satisfaisants et prometteurs

Références bibliographiques

[4] N. GUEZGUEZ A. BEN ISMAIL N.BEN SlIMAN « Asservissement de position et de vitesse d'une articulation Robotique » mini projet 2012

[13] R. F. RAMAMONJISOA « MODELISATION ET REGULATION D'UN MOTEUR A COURANT CONTINU APPLIQUEE A UNE SCIE » mémoire PFE 25 Septembre 2019

[17] A. BENSAYAH« Contrôle de vitesse d'un moteur à courant continu » mémoire PFE Juin 2016

[20] S. BENSIZERAR « Commande d'un système pneumatique didactique par automate programmable SIMATIC S7-1200 » mémoire PFE 2018

[24] S. OUGRINE « Commande de moteur asynchrone par variateur de vitesse SEW relié avec API S7-300 et superviseur HMI » mémoire PFE 2017/2018

[26] Y. OUABBAS « AUTOMATISATION DES CYCLES DE TRANFERT DU SUCRE VERS LES TREMIES DE CONDITIONEMENT AU SEIN DE L'ENTREPRISE CEVITAL SPA-BEJAIA » mémoire PFE 2018

[27] S. BOUTIBA « Etude et miss en œuvre de la commande automatique de l'ensacheuse rotative et la supervision » mémoire PFE 2017/2018

[28] Mme HADJEM « Automatisation de l'autoclave AAV612 de l'unité verre feuilleté de MFG par S7-300 » mémoire PFE 27/ 09/ 2014

Web graphiques :

[3] ''Régulation de position (CNC-1)'' 04/2021 {site web} disponible sure : Régulation de position d'axes de machine, principes généraux (michel.re).

[15] https://www.maxicours.com/se/cours/types-de-moteur-a-courant-continu/.

[18] http://www.gecif.net/articles/genie_electrique/ressources/RessourcesSI/BSite/GM/meca-Reducteur.html.

[19] https://www.academia.edu/10869507/Modelisation_moteur_a_courant_continu.

[21] https://vdocuments.mx/representation-symbolique-de-la-regulation.html(photo de principe de regulation).

[22] https://www.se.com/fr/fr/faqs/FA138378/(principe de regulation).

Bibliographie

[23] Régulation du moteur DC " pdf _Michel Girardin et Bernard Schneider.

[25] help de tia portal.

[29] https://fr.wikipedia.org/wiki/Drone.

[30] https://fr.wikipedia.org/wiki/Lecteur_de_CD.

[31] https://fr.wikipedia.org/wiki/Ascenseur.

[32] https://fr.wikipedia.org/wiki/Robot.

[33] https://fr.wikipedia.org/wiki/Imprimante.

[34] https://fr.wikipedia.org/wiki/Bras_manipulateur

[36] pdf « Regulation PID Logiciel de base pour S7300-400 ».

ملخص:

الأتمتة ضرورة. في أنظمة الاجتماعات التي تهدف إلى التحكم في العمليات الفيزيائية

الهدف من هذه الأطروحة هو دراسة التحكم في موضع محرك DC ، لقد طورنا برنامجًا في برنامج STEP7 ليتم تحميله في PLC للتحكم في المحرك الحالي وانتهينا من عملنا من خلال إدخال نظام إشراف لضمان الإنسان / الآلة واجهة والتأكد من التحكم في العملية ومراقبتها.

في نهاية هذا العمل، اكتسبنا خبرة جيدة جدًا. هذه المعرفة فيما يتعلق ببرنامج الإشراف WinCC، وبرمجة S7-300 PLCsوالتحكم في الموقع

أخيرًا، نأمل أن يكون مشروعنا مرحلة تعليمية عملية ومفيدة لمن خلفنا لجميع أولئك الذين سيتعاملون مع نفس الموضوع

كلمات مفتاحية: محرك التيار المستمر، التنظيم، النمذجة، الموضع.

Abstract

Automation is a necessity. In meeting systems whose purpose is to control physical processes

The objective of this thesis is to study the position control of a DC motor, we developed a program in STEP7 software to be loaded into the PLC to control a current motor and we finished our work by introducing a supervision system to ensure the Man/Machine interface and ensure the control and monitoring of the process.

At the end of this work, we acquired a very good experience. This knowledge concerning the supervision software WinCC, the programming of S7-300 PLCs and the position control

Finally, we hope that our project will be a practical and useful learning phase for our successors to all those who will deal with the same subject.

Keys Words: Direct Current Motor, regulation, modeling, position.

Résumé

L’automatisation est une nécessité. En rencontre des systèmes dont le but est de contrôle des procédés physiques

L’objectif de ce mémoire est étude de la régulation de position d’un moteur à courant continu, on a développé un programme dans le logiciel STEP7 qui sera chargé dans l'automate programmable en vue de commander un moteur a courant et on a terminé notre travail par l'introduction d'un système de supervision pour garantir l'interface Homme/Machine et assurer le contrôle et la surveillance du procédé.

Au terme de ce travail, on a acquis une très bonne expérience. Cette connaissance concernant le logiciel de supervision WinCC, la programmation des automates S7-300 et la régulation de position

Enfin, nous espérons que notre projet sera une phase d'apprentissage pratique et utile pour nos successeurs à tous ceux qui traiteront du même sujet.

Mots clés : Moteur à Courant Continu, régulation, modélisation, position.