الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne démocratique et populaire

وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

جامعة سعد دحلب البليدة
Université SAAD DAHLAB de BLIDA

كلية التكنولوجيا
Faculté de Technologie

قسم الإلكترونيك
Département d'Électronique

Mémoire de Master

Mention Électronique
Spécialité Signaux en Ingénierie des Systèmes, Informatique industriel (SISII)

présenté par

RAHALI IDRIS.

&

MAROUF YOUCEF.

STABILISATION D'UN PENDULE INVERSE PAR API

Proposé par : BENNILA NOUREDDINE

Année Universitaire 2016-2017

Remerciements

Remerciements

Nous remercions avant tout, Allah notre Dieu de nous avoir donné la force et le courage de nous mener à terme.

Nous tenons à exprimer nos vifs remerciements à no promoteur Mr BENNILA Noureddine, pour nous avoir proposé ce sujet, leurs précieux conseils tout au long de notre travail, leur aide et leur confiance.

Nous remercions très chaleureusement les membres du jury pour

L'honneur qu'ils nous ont fait en acceptant d'examiner notre travail.

Nous exprimons nos profondes reconnaissances à tous les enseignants

De notre département d'électronique, et en particulier nos enseignants de SISII

Pour leur disponibilité et leur gentillesse durant ces deux dernièresAnnées.

Nous aimerions adresser du fond du cœur nos plus fervents

Remerciements à nos parents, car nul autres qu'eux se sont plus sacrifiés pour

Notre bien et l'accomplissement de nos projets .Il sont fait de nous ce que

Nous somme sa aujourd'hui.

Enfin, tous nos remerciements à toutes les personnes qui ont contribué

de près ou de loin à l'aboutissement de ce modeste travail. Nous leurs

Sommes très reconnaissants.

ملخص:لقدسطرناكهدفمنهذاالمشروع،دراسةوتحقيقاستقرارنواسمقلوبباستعمالمبرمجالتحكمالالي، PIDومعاينتهباستعمالبرمجيةTIA PORTAL ،يعملهذاالبرنامجكمتحكمفيوحدةتصحيح API التصحيحيكونعلىمستوىالتغييرفيزاويةالنواسلكينصلالىنقطةتحقيقتوازنهذاالنواسالذييكونفيهذهالحالةعموديااوذالكباختيارا لاعداداتالمناسبةللمصحح.

كلماتالمفاتيح:نواس ; TIA PORTAL; API; PID،

Résumé : Nous nous sommes assigné comme objectif l'étude et la réalisation d'un pendule inversée par un API programmé et supervise à travers un logiciel appelé TIA PORTAL ce programme joue le rôle d'un correcteur PID, dont la correction est au niveau de l'angle du pendule pour arriver à un point d'équilibre ou le pendule est verticale en jouant sur les paramètres de régulation.

Mots clés :TIA PORTAL ;API ;PID; pendule inversée.

Abstract : In this work, our main goal is the study and the realization of an inverted pendulum using an API and a software named TIA PORTAL as a supervisor, This program play the role of PID corrector,whose correction is at the angle of the pendulum, to arrive at equilibrium point or the pendulum is vertical, by playing on the regulation parameters.

Keywords:TIA PORTAL;API;PID; inverted pendulum.

Listes des acronymes et abréviations

API : Automate Programmable Industriel.

PLC: Programmable logic Controller.

TIA : Totally Integrated Automation.

MPI : l'interface Multipoint.

CPU : Unité centrale de l’automate (Central Processing Unit).

OB : Organisation Bloc.

TOR : tout ou rien.

E/S : Entrées/Sorties.

IHM : interface Homme Machine.

RT: RUNTIME.

Table des matières

Liste des figures

Liste des tableaux

Utiliser cette liste si vous avez des tableaux dans votre manuscrit.

Introduction générale

Plusieurs inventions imminente ont commencé par des idées simples, les chercheurs utilisent généralement des exemples classiques pour leurs recherches.

L'un de ces exemples dans le domaine d'automatisme est le pendule inversé.

Le développement massif de la technologie des API a donné un élan très important au secteur industriel, qui se traduit par l'augmentation de la productivité avec toute sa diversité.

Avant, les API n'étaient réservés qu'au séquencement des taches au niveau des machines de production. Les systèmes à base d'automate programmable ne peuvent pas répondre aux exigences temporelles de certains procédés de fabrication industrielle vue la lenteur des CPU, des automates programmables et l'exécution cyclique des programmes. Les constructeurs des API se sont investis sur la technologie des CPU pour pouvoir contourner les exigences des systèmes à temps réel.

C'est dans ce contexte bien précis que nous allons développer notre projet, et pour application nous avons choisi la stabilisation du pendule inversé, qui a fait l'objet de plusieurs études vue son exigence très pointu en temps de réponse des équipements le constituant.

L'objectif de ce projet est de réalisé une commande linéaire à base d'un automate programmable industriel, cette réalisation nous permet de stabiliser notre pendule inversé, tout en assurant le transfert des données vers un superviseur.
Un pendule est essentiellement une tige ayant un poids collé en dessous, on parle d'un pendule inversé lorsque cette dernière est inversé, c'est une tige suspendu en l'air instable grâce à un point situé en bas de la tige appelé liaison pivot.
L'idée est d'appliqué une action sur la liaison pivot pour garder l'équilibre.

Probablement nous avons à un certain moment tous fait ce genre d'expérience dans notre vie en utilisant un bâton de bois et essayer de le maintenir en équilibre sur notre doigt, afin d'éviter la chute de celui-ci, nous devons déplacer notre main pour garder l'équilibre de ce bâton c'est exactement le même système qu'on veut analyser et réaliser dans ce projet.

Le contexte de notre système n'est pas l'équilibre du bâton de bois, mais l'équilibre d'un pendule inversé qui est maintenu par un chariot mobile.

D'abord, on passe par une analyse générale du système, et observer comment fonctionne le système réellement.

Le chariot a une accélération commandé par un moteur, on peut prévoir quelques perturbations appliquées par l'environnement.

Donc, en observant le système on remarque qu'il y a deux entrées, une entrée qui représente les perturbations externes qu'on suppose incontrôlable ,et l'autre représente l'accélération extérieure applicable ou chariot, cette accélération remplace le mouvement de la main dans le cas d'équilibre du bâton en bois, puis à travers la dynamique du système qui est provoquée par l'influence de l'angle de la tige ,ce qui fait que la sortie du système est l'angle lui-même.

Alors, l'idée est de contrôler l'équilibre du système en choisissant une accélération qui permet de stabiliser l'angle à zéro en utilisant une boucle de régulation fermée avec une rétroaction autour du système grâce une mesure de l'angle traites par un correcteur approprié.

On a divisé notre mémoire en quatre chapitres ou le chapitre 1 est réservé à la description du système puis le chapitre 2 qui parle du schéma fonctionnel et interface, ensuite le chapitre 3 pour la description du programme utilisée dans l'API et l'interface, en fin le chapitre 4 dont on trouve les expériences et les résultats qu'on a élaboré.

Chapitre 1 Description du système

1.1 Introduction

Dans ce chapitre on va décrire la plateforme du système et établir sa modélisation mathématique afin de trouver des solutions pour stabiliser le système.

1.2 Plateforme expérimentale

Dans cette partie on va discuter le coté mécanique de notre conception (plateforme expérimentale).

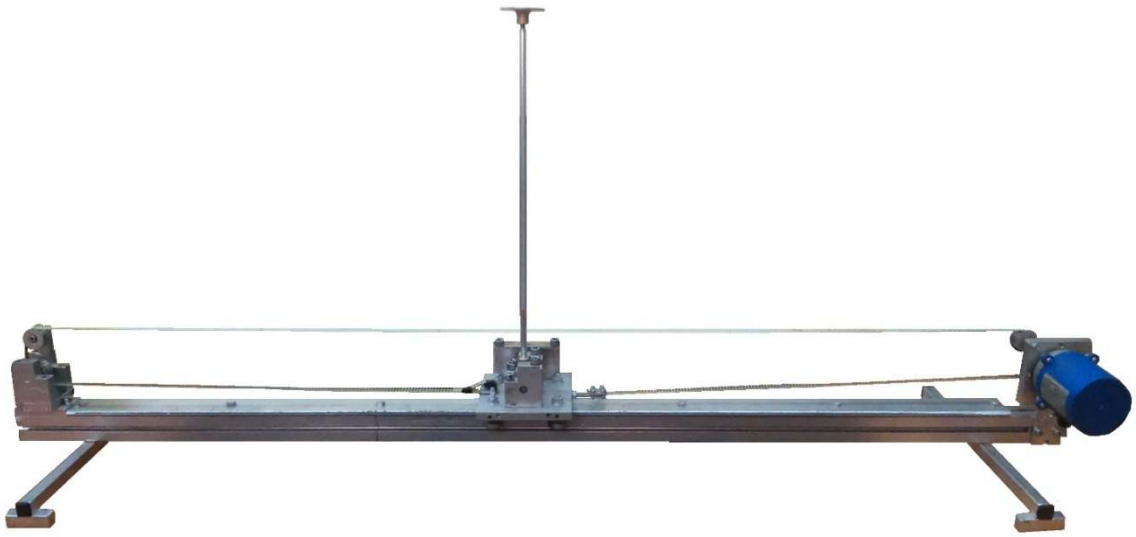


Figure1.1 : vue d'ensemble du pendule inversé.

Le système de pendule inversé est composé d'une table et d'un chariot mobile comportant une tige dont on doit garder son équilibre.

1.2.1 Chariot

Repose sur quatre roues glissantes actionnées par un moteur à travers une courroie dentée relié à quatre poulies dentées pour un mouvement de translation, et un système d'articulation (point pivot) portant une tige, ainsi un emplacement pour un potentiomètre. (Figure 1.2).

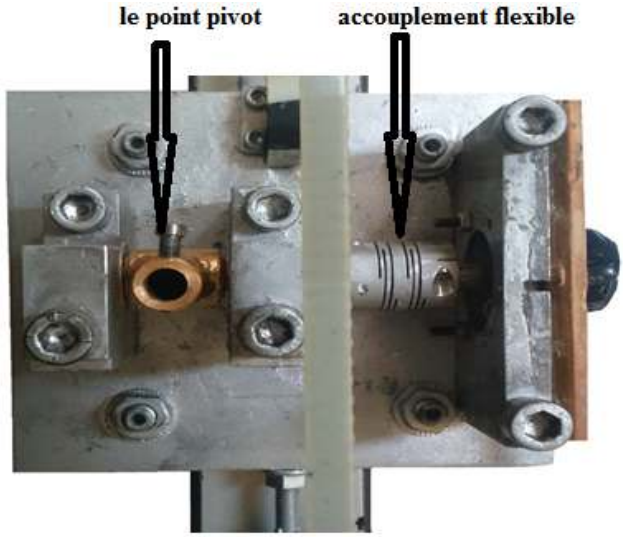


Figure 1.2 : le chariot

1.2.2 Tige

Est un élément essentiel dans ce système où le but est de la stabilisée à un angle de 0°, la tige est de forme cylindrique comporte en haut un support pour un poids et en bas fixée à l'articulation. . (Figure 1.3).

Figure 1.3 : la tige.

1.2.3 Liaison pivot

Fixée par deux petite plaques chacune contient un roulement et une liaison entre eux par un axe adapté aux dimensions, ce qui permet un mouvement oscillatoire du support de la tige. (Figure 1.4).

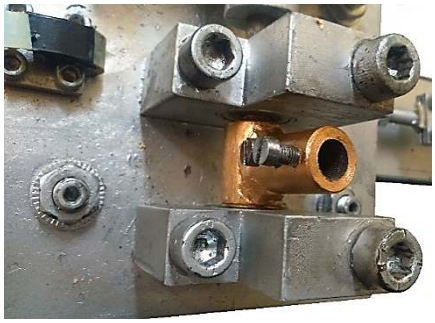


Figure 1.4 : le pivot qui supporte la tige.

1.2.4 Support système

Une table composée de deux rails pour le déplacement du chariot à travers les roues glissantes pour chaque rail deux roues, et d'un emplacement pour le moteur qui fait tourner les poulies, la table est limite à gauche et à droite par deux capteurs de fin de course. (Figure1.5).

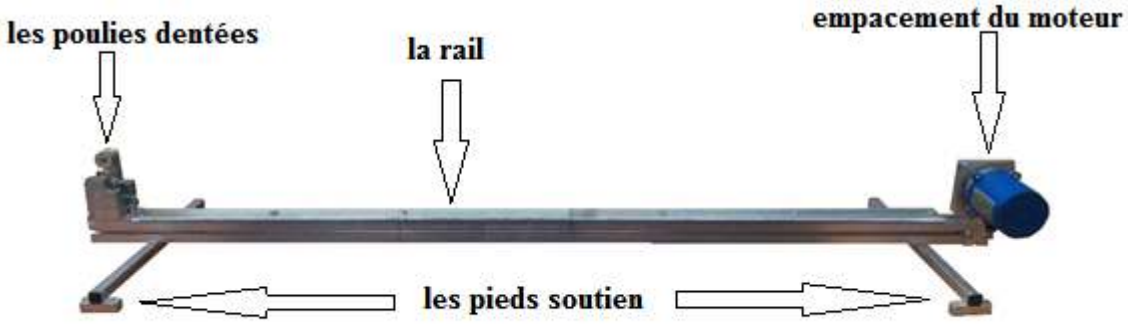


Figure 1.5 : la table du système**.**

1.2.5 Les roues

Fixé au chariot permettant son mouvement avec souplesse tout le long de la rail. (Figure1.6).

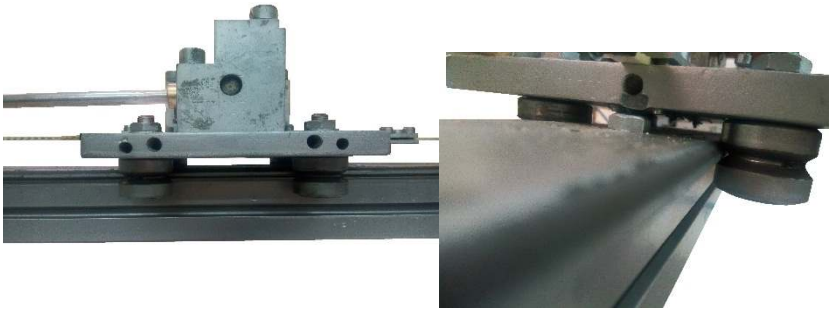


Figure 1.6 : les roues.

1.2.6 Courroie

Est une courroie dentée avec une largeur de 1cm et une longueur de 3.5m, la distance entre ces dents [2mm]. Ainsi deux petites plaques de fixation l'une se situe dans le chariot et l'autre à 3 cm du chariot. (Figure 1.7).

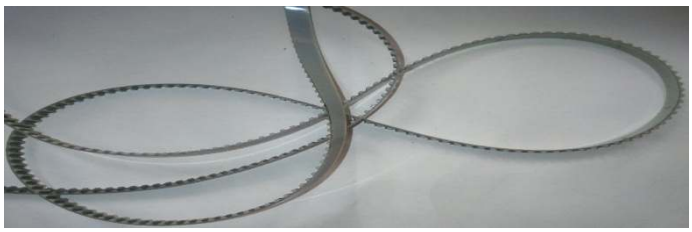


Figure 1.7 : la courroie dentée**.**

1.2.7 Poulies

Le système comporte quatre poulies dentées composées de 20 dents. (Figures 1.8 & 1.9). À travers la courroie leurs mouvements circulaires se transforment en déplacement horizontal du chariot.

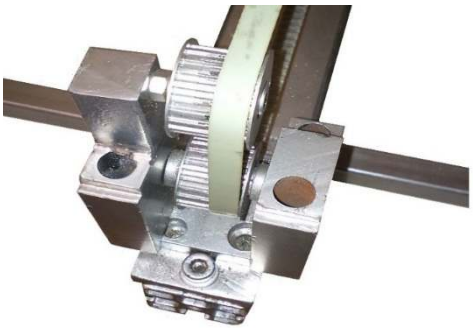


Figure 1.8 Figure1.9

Figure 1.8 et figure 1.9 : représente les poulies dentées du système.

1.3 MODELISATION MATHEMATIQUE

Le but de cette partie est l'obtention de l'équation d'accélération du chariot pour pouvoir stabiliser le système, ce système consiste à mesurer l'angle de la tige, en choisissons l'accélération du chariot en fonction de cette angle.

Le système que l'on veut étudier se compose de deux entrées, l'une de ses entrées non commandée représente les perturbations externes, et l'autre commandé représente l'accélération du chariot, la sortie doit donner l'angle.

Pour stabiliser le système on place une contre réaction autour de lui, grâce à une mesure de la sortie par un capteur approprié traité avec un contrôleur adéquat en utilisant ce dernier pour réglée l'accélération du chariot.

Si on choisit correctement Le régulateur on peut obtenir un système stable, même si le système en boucle ouverte est instable.

Premièrement, nous allons analyser le système en boucle ouverte, puis voir comment le stabiliser à l'aide du contrôleur.

Nous effectuons une analyse du système à boucle ouverte avec plusieurs choix possibles de contre réaction à explorer.

L'équation du pendule inversé peut être décrite par une équation différentielle du second ordre linéaire à des coefficients constants.
À travers la transformée de Laplace on trouve deux pôles réel, l'un dans la moitié gauche du plan S et l'autre dans la moitié droite, d'après ce critère le système est instable.

Comme premier choix de correction, on peut choisir tout simplement l'accélération proportionnelle à l'angle mesuré du chariot$a(t) = k_1\theta(t)$, la fonction du système résultante à deux pôles dépond du gain (retour) k_1 s'il est supérieur à zéro les deux pôles sont proche de l'origine ensuite se déplaceront à l'axe de l'imaginaire, La présence d'un pôle positif indique que même avec une très faible valeur mesuré (petit déplacement de la tige), l'angle de la tige augmente de façon exponentielle. Avec les pôles sur l'axe imaginaire tout déplacement entraine un comportement oscillatoire de la tige, en conséquence le système reste instable pour toutes les valeurs du gain de retour.

Le deuxième choix, l'accélération proportionnelle au dérivé du déplacement angulaire $a(t) = k_2 \frac{d\theta(t)}{dt}$alors si le gain de retour k_2 est supérieur à zéro, le système est alors plus stable mais pas complètement.

Enfin, nous proposant un contrôleur avec deux actions proportionnelles et dérivateur $a(t) = k_1\theta(t) + k_2 \frac{d\theta(t)}{dt}$, dans le cas où les deux facteurs de gain k_1>0&k_2>0 le système peut devenir stable.

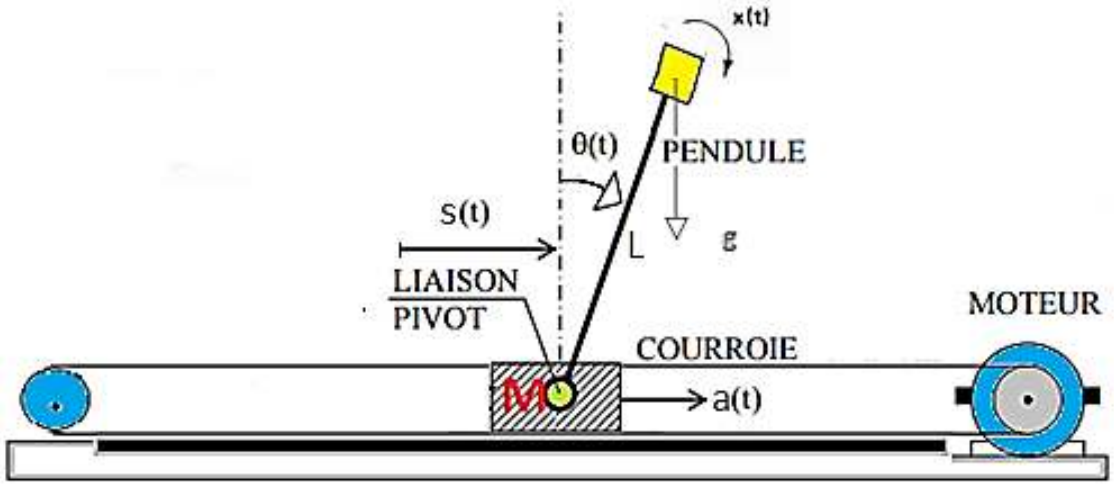


Figure 1.10 : représentation des forces.

Les notations sont les suivantes :

_ x(t) : l'accélération angulaire.

_ g : l'accélération de la pesanteur.

_ a(t) : l'accélération du chariot.

_ L : la longueur de la tige.

_ Θ(t):l'angle mesuré.

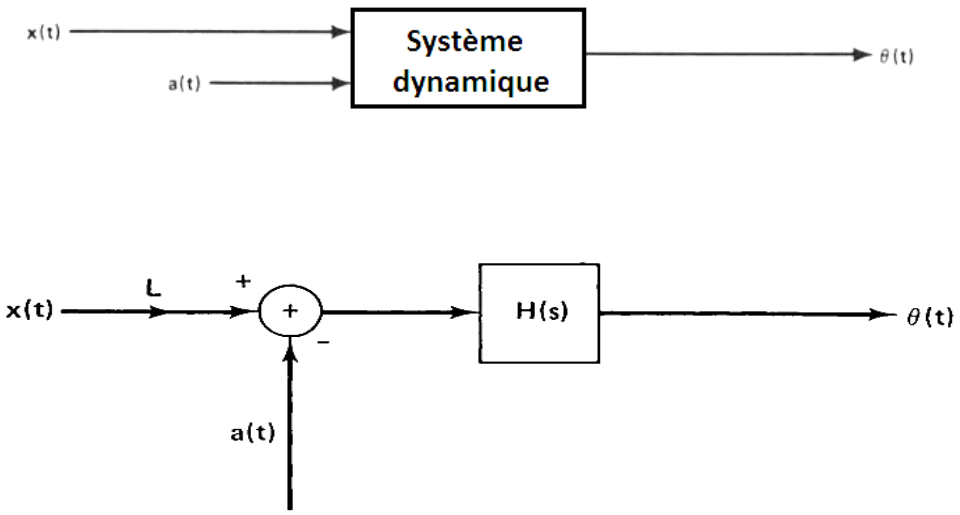


Figure 1.11 : Représentation de Système à boucle ouverte.

Dans un premier temps, l'analyse de système boucle ouverte sans retour, (figure1.11), prouve qu'il est instable, ensuite en utilisant la contre réaction pour parvenir à la stabilité.

Comme nous l'avons indiqué, la variable est l'angle mesuré qui représente la sortie, l'accélération angulaire due à des perturbations extérieures, ceux-ci indique qu'il faut appliquer une accélération sur le chariot, il y'a aussi les variables comme l qui représente la longueur de la tige et g qui représente l'accélération de la pesanteur avec S(t) qui est la position du chariot.

A partir de la deuxième loi de la dynamique on obtient l'équation en termes d'équilibre :

$$l\,\frac{\partial^2\theta(t)}{\partial t^2} = g\sin\theta(t) + lx(t) - a(t)\cos\theta(t).$$

Lorsque le système est en équilibre stable, L'accélération angulaire de la tige $\frac{\partial^2\theta(t)}{\partial t^2}$ dépend de trois paramètres, l'accélération angulaire $x(t)$ qui représente les perturbations externes et la pesanteur de la tige, aussi l'accélération du chariot qui rattrape l'équilibre en jouant le rôle de la force opposée à la déstabilisation.

$$\frac{\partial^2\theta(t)}{\partial t^2} = \frac{g\sin\theta(t) + lx(t) - a(t)\cos\theta(t)}{l}$$

On remarque que la longueur de la tige et l'accélération angulaire sont de relation inversement proportionnelle tel que plus la longueur est grande plus l'équilibre est facile à contrôler.

Maintenant, cette équation est non linéaire. Ce que nous voulons faire est linéariser l'équation. Nous proposant alors une hypothèse selon laquelle l'angle est très faible ce qui veut dire proche de 0°.En d'autres termes, on suppose que nous sommes capables de Maintenir la tige relativement verticale.

Selon le développement limité nous trouvons que le sinus est approximativement égal à l'angle et cosinus approximativement égal à 1.

$$\sin\theta(t) \approx \theta$$

$$\cos\theta(t) \approx 1$$

Nous obtenons cette équation

$$l\,\frac{\partial^2\theta(t)}{\partial t^2} - g\theta(t) = lx(t) - a(t).$$

Ce que l'on voit sur le côté droit de l'équation est le côté d'entrée avec $\frac{\partial^2\theta(t)}{\partial t^2}$l'accélération angulaire de la tige et $\theta(t)$le changement d'angle. Sur le côté gauche de l'équation, nous avons l'accélération du chariot a(t) et la vitesse angulaire x(t).

Si on applique la transformée de Laplace l'équation devient :

$$\Theta(s) = \frac{1}{Ls^2 - g}[LX(s) - A(s)]$$

Il y a deux pôles, parce qu'elle est de second ordre :(figure1.12)

$$S^2 = \frac{g}{L}$$

$$\boldsymbol{s = \pm\sqrt{\frac{g}{L}}}$$

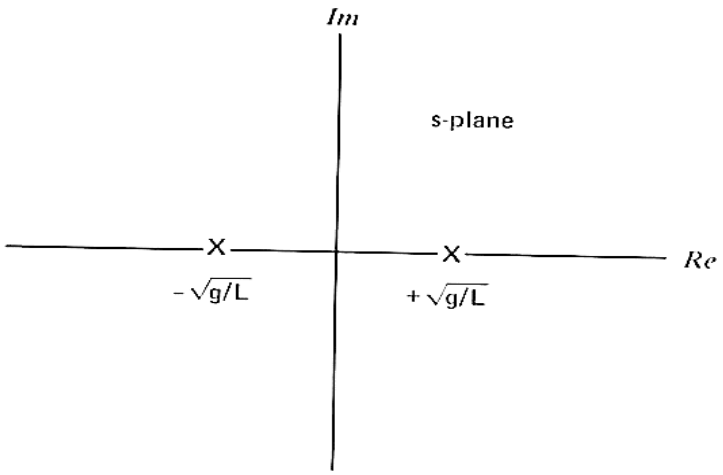


Figure 1.12 : les pôles de système en boucle ouverte.

Observation importante : le pôle négatif représente un pôle stable, et le pôle de la moitié droite représente le pôle instable. Ce système n'est pas stable à cause du pôle dans la moitié droite du plan S.

Maintenant, on entame la boucle fermée et comme première étape on a choisi le retour d'état $G(s) = k_1$, dans lequel l'angle mesuré est renvoyé par le choix approprié de la contre réaction. Comme le montre la figure 1.13 :

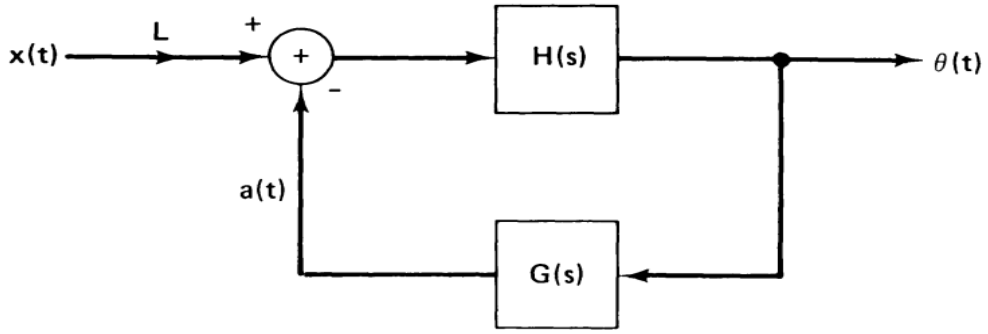


Figure 1.13 : Représentation de Système à boucle fermé.

Pour avoir la fonction de transfert du système on applique la transformée de Laplace :

$$\Theta(s) = \frac{LH(s)}{1 + G(s)H(s)} X(s)$$

L'équation de retour d'état:

$$G(s) = k_1$$

L'équation temporaire de l'accélération :

$$a(t) = k_1 \theta(t)$$

En remplaçant $H(s)$ et $G(s)$ par leurs valeurs nous trouvons

$$\Theta(s) = \frac{1}{s^2 - \left(\frac{g - k_1}{L}\right)} X(s)$$

$$\boldsymbol{s = \pm \sqrt{\frac{g - k_1}{L}}}$$

D’abord, on fixe K1 à 0, ce qui implique l’absence de réaction est identique au système en boucle ouverte, avec les pôles obtenu précédemment.

Comme deuxième étape on prend K1<0, plus la valeur absolu de K1 est plus grande le pole qui se trouve dans la moitie gauche s’éloigne beaucoup plus dans la partie gauche par contre celui qui se trouve dans la moitie droite se perd dans le partie droite, (figure1.14)
Donc le système devient instable beaucoup plus.

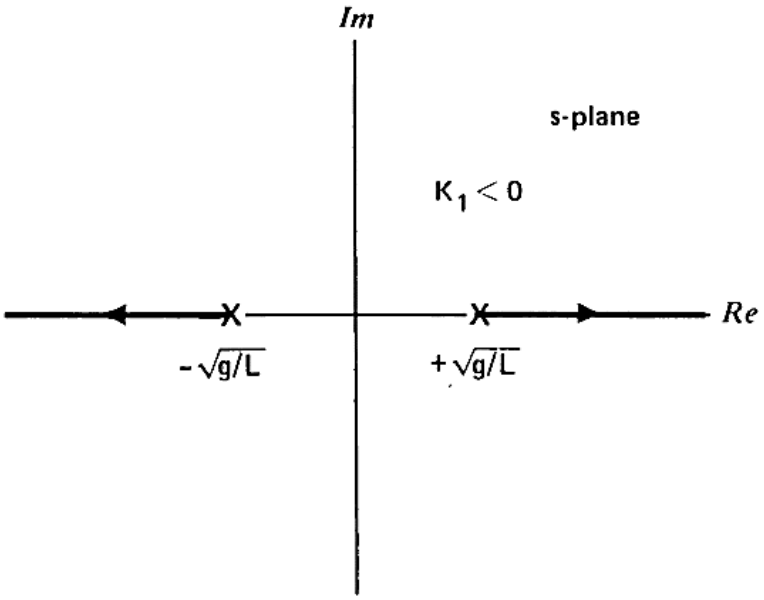


Figure1.14 : représentation des pôles dans le cas de K1<0.

Au lieu de prendre K1<0, nous allons prendre K1>0 et étudier les conséquences, (figure1.15),dans ce cas les deux pôles se rapproche de l’origine et plus |k1|est grande plus les deux pôles s’étalent sur l’axe de l’imaginaire, parconséquent la tige est en mouvement Oscillatoire, le chariot se déplaçant en avant et en arrière dans la direction opposée de l’angle de tige pour compenser le risque d’erreur.
Le système devient alors asymptotiquement stable, ce qui nous oblige à placer les deux pôles dans la moitié gauche du plan S pour aboutir à une stabilisation.

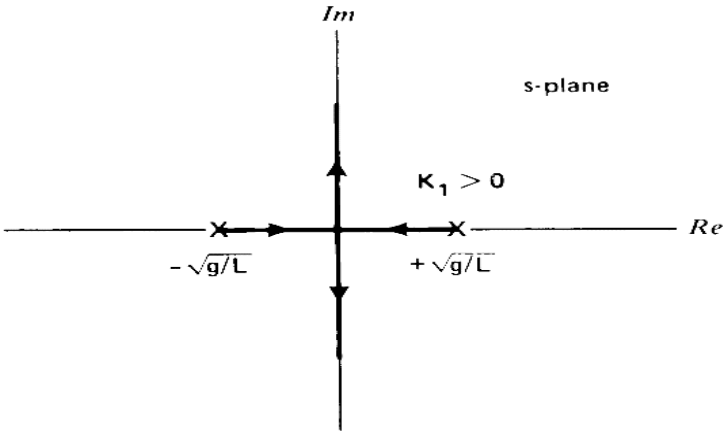


Figure1.15 : représentation des pôles dans le cas de K1>0.

Dans ce qu'on a dit précédemment on voit clairement que c'est impossible à stabiliser le système avec le paramètre K1 de la commande proportionnelle, ce qui nous oblige à utiliser un autre paramètre,

Donc on doit agir sur la contre réaction en insérant un dérivateur et on élimine l'action proportionnelle pour voir son influence sur le système, afin qu'on puisse étudier la possibilité d'utiliser un dérivateur. Ce que nous devons faire est d'analyser l'accélération proportionnelle à la dérivée de l'angle. Au lieu d'être proportionnelle à l'angle, comme dans le cas précédent,

L'équation de retour d'état :

$$G(s) = k_2 s$$

L'équation de l'accélération :

$$a(t) = k_2 \frac{d\theta(t)}{dt}$$

On applique la transformée de Laplace :

$$\Theta(s) = \frac{1}{s^2 - \frac{k_2}{L}s - \frac{g}{L}} X(s)$$

On passe à l'étudie de la variation de ces deux pôles en fonction de K_2 , on répète les mêmes étapes utilisées pour k1.

$$s = -\frac{K_2}{2L} \pm \sqrt{\left(\frac{K_2}{2L}\right)^2 + \frac{g}{L}}$$

Dans la première étape K2 est égale à 0, donc aucune réaction dans le système.

Si K2 <0, plus|k2|>0 plus les deux pôles se déplacent vers la droite de l'axe des réels, (figure1.16), donc le système est de plus en plus instable.

Finalement Lorsque K2 négative, le système n'est pas stable.

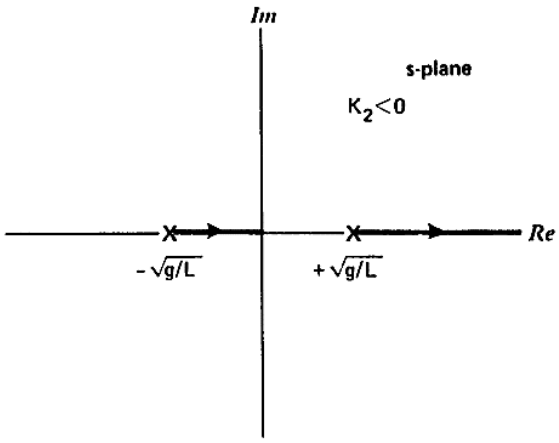


Figure1.16 : représentation des pôles dans le cas de K2<0.

On prend K2>>0, plus|K2|>>0 plus les deux pôles se déplacent vers la gauche du plan S, ce qui alors, le pole qui se trouve dans la partie droite se rapproche de l'origine et il reste sur l'axe imaginaire (figure.1.17), donc c'est convenable Pour augmenter la stabilité mais Pas jusqu'au point d'équilibre.

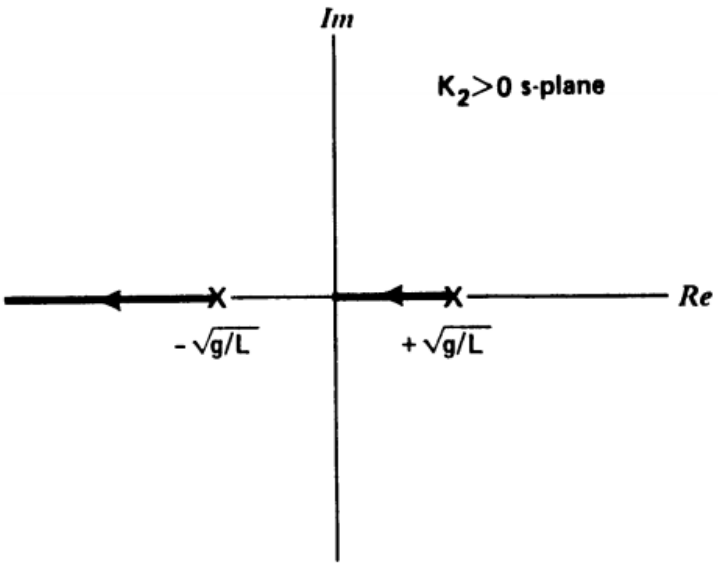


Figure1.17 : représentation des pôles dans le cas de K2>0.

Comme troisième choix pour le système en boucle fermée est de voir si nous pouvons le stabilisé et parvenir à avoir l'angle et en même temps cet angle change de façon rapide, pour que le chariot se déplace rapidement en sens opposé de l'angle, rattrapant l'équilibre de la tige ce qui permet la stabilisation du pendule.

Maintenant ce que nous voulons étudier les deux premier choix mais dans la même boucle, donc La fonction de l'accélération du chariot comporte deux paramètres. Le paramètre K2 qui est proportionnelle à la dérivée de l'angle, et la constante K1 qui est proportionnel à l'angle.

$$\boldsymbol{a(t) = k_1\theta(t) + k_2\frac{d\theta(t)}{dt}}$$

La transforme de Laplace :

$$\boldsymbol{G(s) = k_1 + k_2 s}$$

Les racines de cette équation :

$$\boldsymbol{s = -\frac{K_2}{2L} \pm \sqrt{\left(\frac{K_2}{2L}\right)^2 - \left(\frac{K_1 - g}{L}\right)}}$$

Dans ce cas k1 et k2 supérieur à 0, on remarque que les pôles se déplacent ensembles vers la gauche, en approchant l'un de l'autre jusqu'à un certain point négatif ou les pôles se déplacent en sens opposés sur l'axe imaginaire. (figure1.18).

Donc le choix de k1 et k2 est approprié pour rendre le système très proche de la stabilité. Ceci explique qu'on doit utiliser les deux paramètres ensembles.

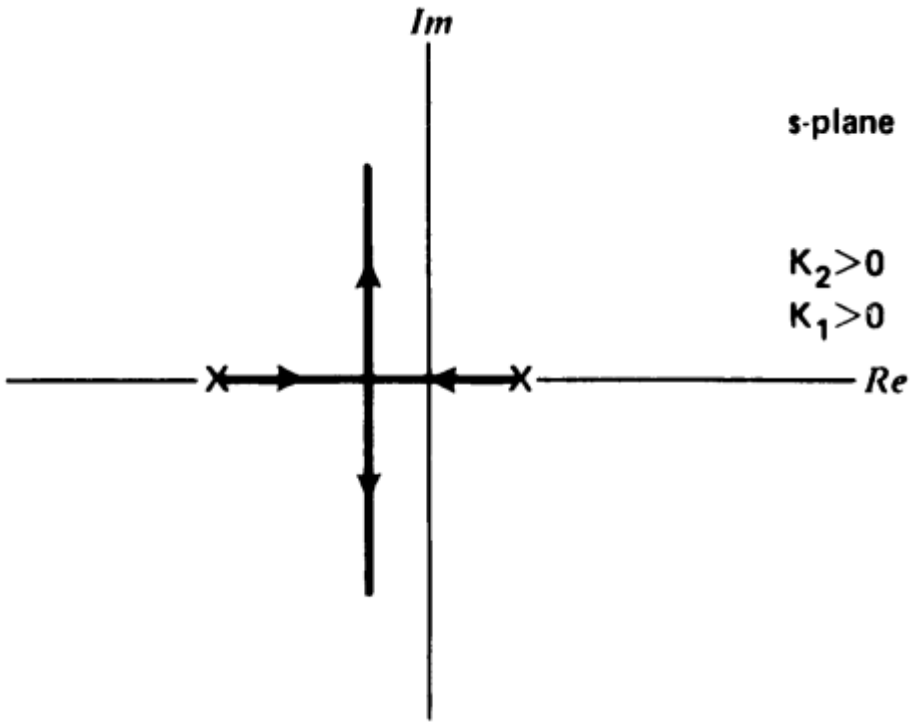


Figure1.18 : représentation des pôles dans le cas de K1>0 et k2>0.

1.3 Conclusion

Nous avons développé la partie mécanique du système et on a pu voire de façon générale l'environnement de notre système et ses constituants ensuite on a fait la modélisation du système ou on est arrivé à une solution pour stabiliser l'équilibre de ce dernier.

Chapitre 2 Développement du schéma fonctionnel sur API

2.1 introduction

Le but de ce chapitre de présentétous les capteurs et actionneurs utilisé dans notre projet, par un schéma bien définie.

2.2 Capteur et actionneur nécessaire au schéma fonctionnel

Pour le développement et l'application de ce schéma, on a commencé par une brève description ainsi que le principe de fonctionnement des capteurs et des actionneurs que nous avons utilisé.

2.2.1 Le capteur

Un capteur est un dispositif qui soumis à l'action d'une grandeur physique, fournit un signal pour la partie commande.

Dans quelques cas ce signal est pneumatique mais dans la grande majorité des cas, cette information se fait par l'intermédiaire d'un signal électrique. [1]

a Codeur incrémental

Dans notre cas, on a essayé un codeur optique de type incrémental ou générateur d'impulsion comme un capteur de position angulaire.

Figure 2.1 : codeur incrémental.

Il se compose d'un disque qui comporte au maximum trois pistes, deux pistes divisé en 250 intervalles d'angles égaux, l'autre pour la fixation du zéro, lorsque le codeur fait un tour complet, il délivre 250 impulsions sur la voie A et 250 sur la voie B.

Figures 2.2 et 2.3

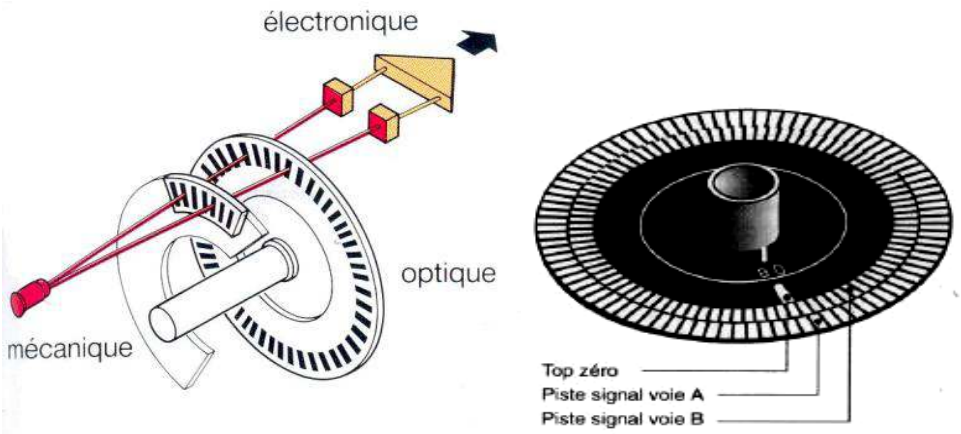


Figure 2.2 : principe de fonctionnement---**Figure 2.3 :** les disques du codeur.

Le déphasage de 90° électrique des signaux A et B permet de déterminer le sens de rotation :

- Dans un sens pendant le front montant du signal A, le signal B est à zéro.
- Dans l'autre sens pendant le front montant du signal A, le signal B est à un. [2]

Cette figure clarifie le fonctionnement :

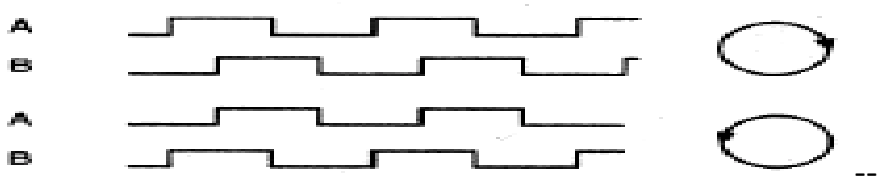


Figure 2.4 : les signaux A et B du codeur incrémental.

Pour un codeur de 250 impulsions par tour, l'erreur est ±1.44° donc ce codeur fournit une faible précision ce qui nous obligent à l'augmentée, afin de la doublée, tout cela en mettant en compte les fronts montants et les fronts descendants du signal A dans ce cas-là on obtient la moitié d'erreur et si on veut quadrupler cette précision, en met en compte aussi les fronts du signal B, on obtient dans ce cas une erreur de ± 0.36°. Figure 2.5.

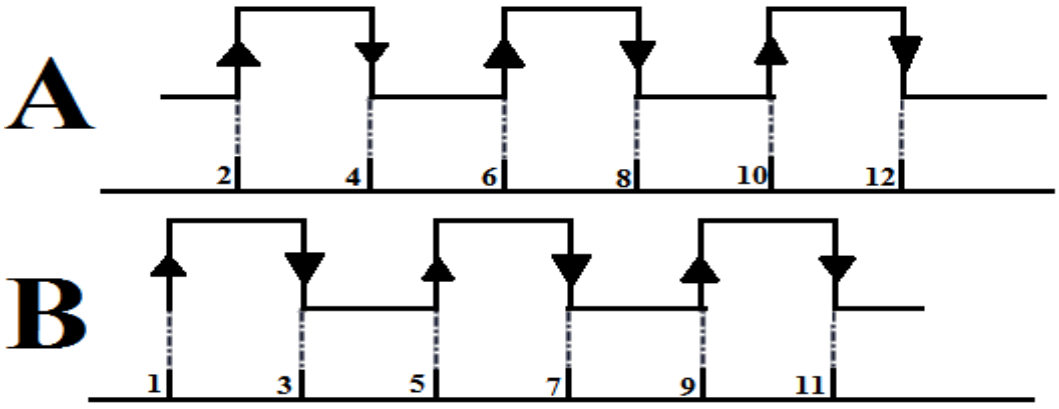


Figure 2.5 : comptage des fronts montants et descendants du A & B**.**

On peut parler ici d'un décomptage si le signal B est capté en premier, dans le cas d'un comptage le signal A sera capté en premier

Par rapport à l'encodeur cité précédemment le potentiomètre et mieux adapté dans notre application, due à la linéarité de ces variations en fonction de déplacement angulaire de la tige.

b Potentiomètre

Le potentiomètre est une résistance variable utilisés pour diviser une tension, contrôler un niveau, compenser des erreurs…etc. [3] .

L'utilisation du potentiomètre c'est pour déterminer la déviation de l'angle de la tige, on a trouvé que le potentiomètre est convenable, parce que qu'il varie d'une manière précise.

Le potentiomètre qu'on a utilisé est linéaire de 220 Ω ohms en un seul tour, appliquée pour une plage de variation de la tension entre 0 à 5v. . (Figure **2.6)**.

Figure 2.6 : le potentiomètre.

c Détecteur photoélectrique

Un détecteur photoélectrique est constitué d'un émetteur (qui est généralement une diode électroluminescente) associe à un récepteur de lumière (généralement un phototransistor).

Il délivre une information chaque fois que le faisceau lumineux issu de l'émetteur est interrompu par un obstacle (objet opaque). Le récepteur détecte la coupure de ce faisceau.

Si l'émetteur et le récepteur son dans le même boitier, le faisceau lumineux doit être renvoyé par un réflecteur ou par l'objet à détecter (objet réfléchissant).

Afin de rendre le dispositif insensible à la lumière ambiante, l'émission de lumière se fait à fréquence fixe (infrarouge par exemple). [4]

Nous avons utilisé deux photocellule émettrice et réceptrice sans objet réfléchissant, (Figure 2.7 et 2.8) positionné aux frontière de la table pour donner un signal d'arrêt à l'arrivé du chariot devant la photocellule.

Figure 2.7 :photocellule OMRON----- **Figure 2.8 :** photocellule télémécanique.

Elles sont composées de trois fils :

-Deux fils d'alimentation.

-Un fil pour l'information vers l'entrée de l'API.

Le faisceau lumineux peut être réglé selon la distance qu'on veut utiliser.

Le type des photocellule est NPN.

2.2.2 ACTIONNEUR

a Le pré-actionneur

Le variateur de vitesse est l'organe incontournable des applications industrielles ou la maitrise de la vitesse de rotation d'un moteur asynchrone est essentielle. (Figure2.9)

Figure 2.9 : variateur de vitesse ATV12.

Ces constituants électroniques regroupent en un seul appareil toutes les fonctions nécessaires à la commande du moteur :

- Démarrage (avec contrôle de l'accélération)
- Inversion du sens de rotation
- Freinage (avec contrôle de la décélération)
- Choix de plusieurs vitesses de rotation
- Variation de vitesse avec consigne analogique
- Surveillance du moteur (courant moteur, échauffement…)
- Contrôle du couple moteur (contrôle vectoriel de flux)

Les variateurs de vitesse électroniques utilisent la technique de modulation de largeur d'impulsion (MLI) ou le contrôle vectoriel de flux. [Guide des automatismes]

On a utilisé un variateur de vitesse Schneider ATV12 avec une alimentation monophasée de 220 V AC et une fréquence de 50 HZ, la puissance du moteur utilisé est de 0.0.9KW

a .Fonction de l'afficheur et des touches

La figure **2.10** décrit la configuration matérielle du variateur ATV12

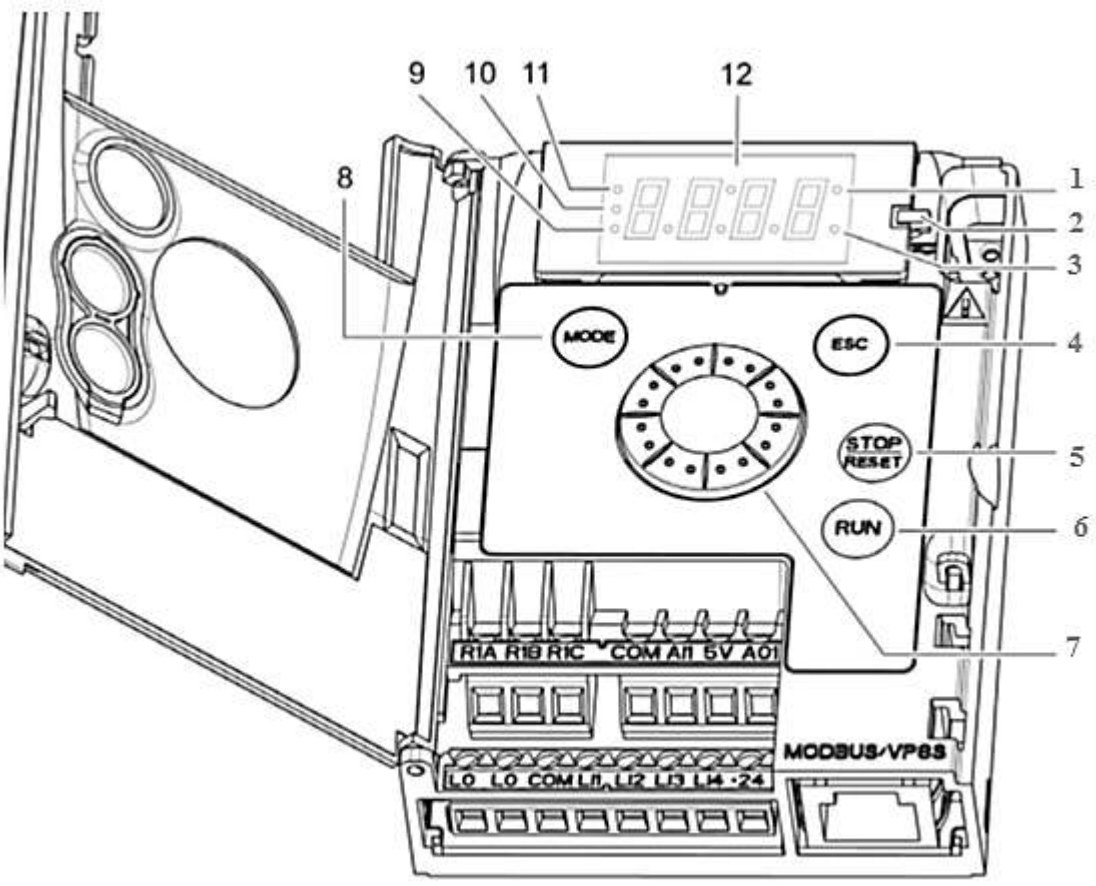


Figure 2.10 : les composants du variateur.

1. DEL de valeur (a) (b).
2. DEL de charge
3. DEL d'unité (c)

Avec (a), (b) et (c) sont :

a)Si elle est allumée, elle indique qu'une **valeur** est affichée, par exemple, 05 s'affiche pour «0,5 ».

(b)Lorsque vous changez une valeur, la DEL du mode Configuration et la DEL de valeur sont constamment allumées.

(c) Si elle est allumée, elle indique qu'une unité est affichée, par exemple, AMP s'affiche pour « Ampères ».

4. Bouton ESC : permet de quitter un menu ou un paramètre ou de mettre fin à la valeur affichée afin de revenir à la valeur précédente se trouvant en mémoire. En configuration locale, une pression de 2 s sur le bouton ESC commune entre les modes contrôle et programmation.

5. Bouton STOP/RESET : arrête le moteur (peut être dissimulé par la porte si la fonction est désactivée)**,** permet de réarmer un défaut.

6. Bouton RUN : lance l'exécution en configuration locale et en configuration à distance si la fonction est configurée (peut être dissimulé par la porte si la fonction est désactivée).

7. Bouton de navigation

- Agit comme un potentiomètre en configuration locale et en configuration à distance si la fonction est configurée.
- Molette servant à la navigation lorsqu'elle est tournée dans le sens horaire ou anti-horaire.
- et à la sélection / validation par simple pression.

8. Bouton MODE

Permet de passer d'un des modes Référence, Surveillance ou Programmation à un autre. Une pression de 3 s sur le bouton MODE commune entre les configurations locale et à distance.

9. DEL du mode CONFIGURATION (b)

10. DEL du mode SURVEILLANCE

11. DEL du mode REFERENCE

12. Afficheur 4 fois 7 segments

Note: En configuration locale, les trois Leds 9, 10, 11 clignotent simultanément en mode programmation et fonctionnent comme un chenillard à Led en mode contrôle.[5]

D'abord, il faut voir comment fonctionne le variateur de vitesse :

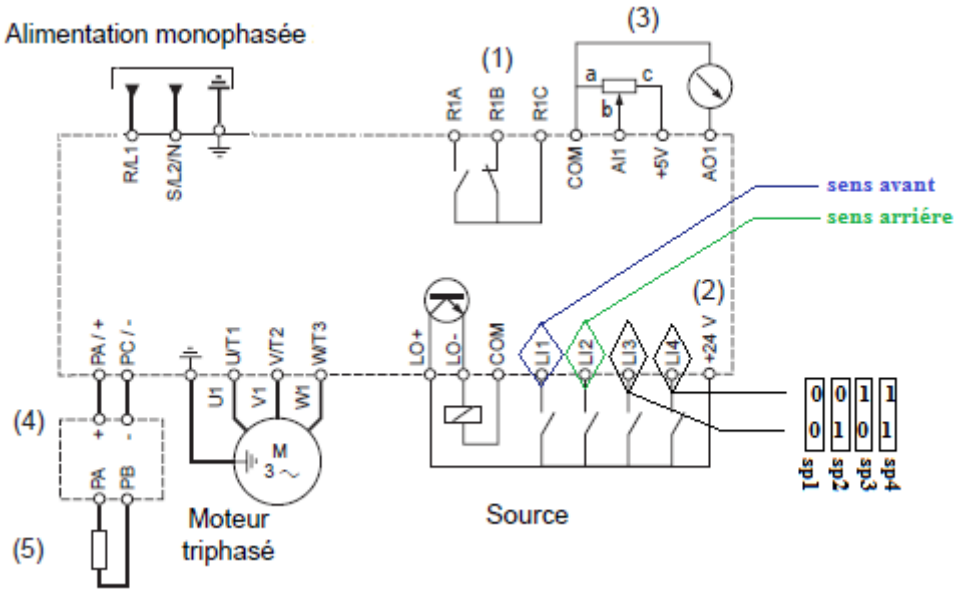


Figure 2.11 : schéma électrique du variateur.

(1) deux Contacts de relais R1, le contact R1A est normalement ouvert (NO), et le contact R1B est normalement fermé (NC) avec la borne R1C qui représente la commune du relais, l'utilité de ces contacts signifie la signalisation à distance de l'état du variateur par exemple en cas ou le variateur en charge indique un défaut, le contacteur R1A se ferme pour donner une alarme.il y'a aussi une autre sortie logique 24V a transistor, il se compose d'un émetteur (LO) et d'un collecteur (LO+), cette sortie pour les actions de tension de 24V par exemple API.

(2) Interne + 24 V DC. Si une source externe est utilisée (+ 30 V DC au maximum), connectez le 0V de la source sur la borne COM.

(3) l'entrée sortie analogique, l'entré analogique AI1 pour le potentiomètre, on peut alimenter directement avec le 5V fournit par le variateur, sa résistance ne dépasse pas 10 KΩ, AO1 est une sortie analogique.

(4) Module de freinage.

(5) Résistance de freinage optionnelle VW3A7 ou une résistance acceptable.

L'injection de la borne LI1 avec 24V pour démarrer le moteur en marche avant, et l'injection de la borne LI2 pour démarrer le moteur en marche arrière, la combinaison des sélections LI3 et LI4 donne la vitesse du moteur :

La combinaison sp1= $\binom{0}{0}$ permet de contrôlée la vitesse par le potentiomètre ou bien par la molette et les autres combinaisons fixées par le paramètre PSS qui se trouve dans le menu FUN du variateur.

b Moteur asynchrone

La machine asynchrone, connue également sous le terme anglo-saxon de machine à induction, est une machine électrique à courant alternatif sans connexion entre le stator et le rotor.

Comme les autres machines électriques (machine à courant continu, machine synchrone), la machine asynchrone est un convertisseur électromécanique basé sur l'électromagnétisme permettant la conversion bidirectionnelle d'énergie entre une installation électrique parcourue par un courant électrique et un dispositif mécanique.

Cette machine est réversible et susceptible de se comporter, selon la source d'énergie, soit en moteur soit en générateur. [6]

Le moteur asynchrone triphasé est largement utilisé dans l'industrie, sa simplicité de construction en fait un matériel très fiable et qui demande peu d'entretien. Il est constitué d'une partie fixe, le stator qui comporte le bobinage, et d'une partie rotative, le rotor qui est bobiné en cage d'écureuil. Les circuits magnétiques du rotor et du stator sont constitués d'un empilage de fines tôles métalliques pour éviter la circulation de courants de Foucault. [7]

Le moteur qu'on a utilisé à un courant nominal de 0.78A sous une tension de 230v, il à un très bon couple et une puissance de 0.09 KW, sa vitesse de rotation atteint à 1380 rpm (tr/min) et une fréquence de 50HZ. (Voir la figure **2.12**).

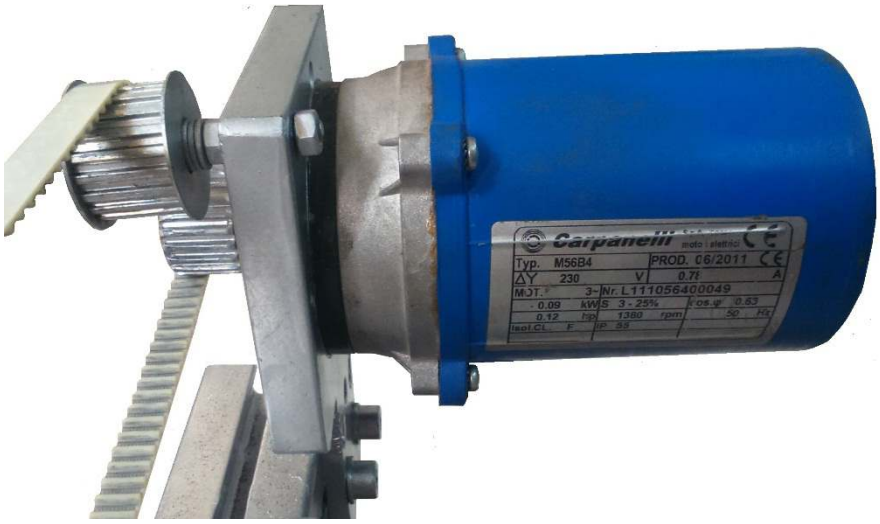


Figure 2.12 : moteur asynchrone.

Le changement de sens de rotation du moteur doit être sous l'influence de l'API sachant que le potentiomètre il détermine la position angulaire et l'actionneur essaye de rattrapé la tige à un angle bien déterminé.

2.3 Développement du schéma fonctionnel sur API et IHM

2.3.1 La Marquette d'essai

Est un dispositif qui remplace l'interface d'entrée-sortie de l'API, afin d'exécuter des opérations à travers des buttons poussoirs et des interrupteurs de manière facile et rapide. Cette interface est composée de deux faces, intérieur et extérieur.

a La face extérieure

Est la face principale ou les opérations sont effectuées, cette dernière dispose de divers éléments, la figure 2 .13, schématise cette face.

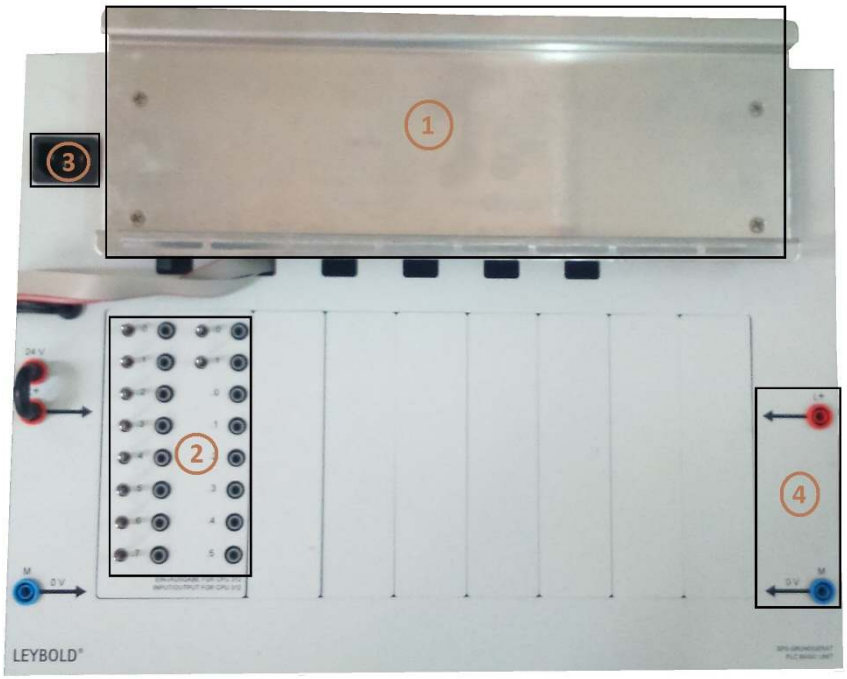


Figure 2.13 : Les composants de la maquette.

①. Le rack : est un support pour l'API et ses modules.

②. Raccordement des d'entrée-sortie de la maquette.

③. Entrée pour l'alimentation 220v AC.

④. Sortie 24v DC.

b La face intérieure

Comporte la carte électronique raccordée à l'interface d E\S de l'API. Figure 2.14.

Figure 2.14 : Raccordement des E/S de la maquette.

. le raccordement de la carte avec l'API.

2.3.2 L'API

Un automate programmable industriel [API] ou [PLC] est une forme particulière d'automate à base de micro-processeur qui se fonde sur une mémoire programmable pour enregistrer les instructions et mettre en œuvre des fonctions, qu'elles soient logiques, de séquencement, de temporisation, de comptage ou arithmétique, pour contrôler des machines et des processus.

Il est conçu pour être manipulé par des ingénieurs ayant potentiellement une connaissance limitée en informatique et en langages de programmation.

En générale, un système API est constitué des composants fonctionnels des bases suivant

- Une unité de traitement et de la mémoire.
- Une unité d'alimentation des interfaces d'entrée-sortie.
- Une interface de communication.
- Une unité de programmation.

Pour fonctionner, le système API doit pouvoir accéder aux données à traiter et aux instructions c'est à dire le programme, ceci indique comment traiter les données.

Ces informations sont enregistrées dans la mémoire de l'API afin d'y accéder au cours du traitement.

Les canaux d'entrées-sorties assurent une fonction d'isolation et du traitement du signal pour que les capteurs et les actionneurs puissent être le plus souvent connectées directement aux entrées-sorties sans nécessité d'autres circuits électroniques.

Les sorties peuvent être a relais, a transistor ou a triacs.

L'interface de communication est utilisée pour recevoir et transmettre des données.

Les API existent principalement sous deux formes :

- Un boitier unique (compact).
- Un système modulaire (rack).

Le cycle d'exécution d'un API :

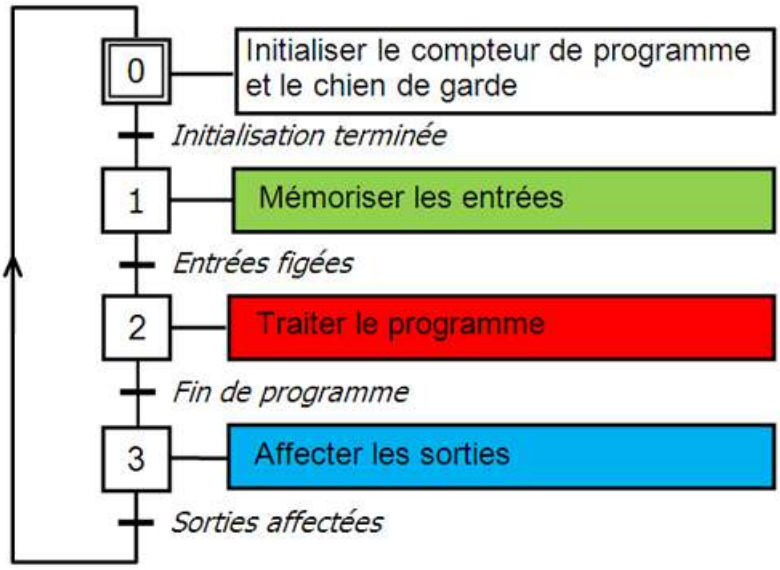


Figure 2.15 : Le cycle d'exécution de l'API.

Choix du matériel :

Système d'automatisation SIMATIC S7-300 :

L'automate utilisé dans notre projet appartient à la gamme SIMATIC S7 de SIEMENS, un automate modulaire avec possibilité d'extensions jusqu'à 8 modules, et une mise en réseau par l'interface Multipoint(MPI).

L'automate S7 est constitué d'une alimentation, d'une CPU et de modules d'entrées ou de sorties (Modules E/S). A ceux-ci peuvent s'ajouter des processeurs de communication et des modules de fonction qui se chargeront de fonctions spéciales, telles que la commande d'un moteur pas à pas par exemple, dans notre cas on a utilisé les modules suivant.

CPU 312c :

Le CPU 312c se caractérise de :

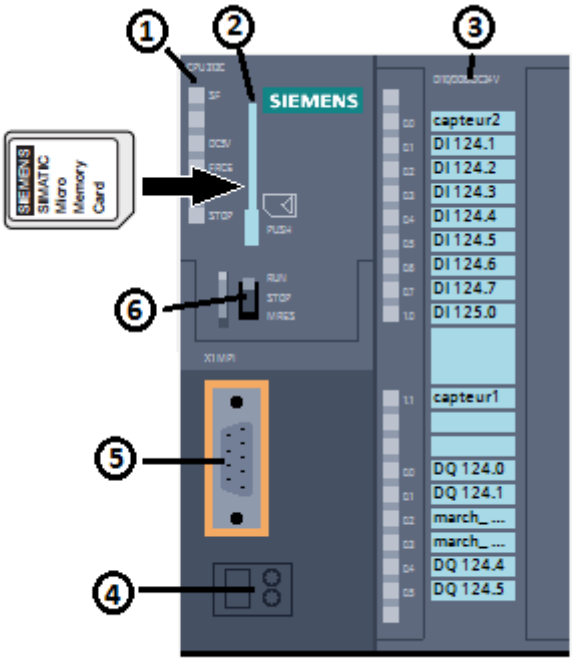


Figure 2.16 :CPU312c et ses éléments.

Les éléments de commande et de signalisation de la CPU indique ci-dessus dans la figure sont :

1. ① Indicateurs d'état et d'erreur
2. ② Logement de la microcarte mémoire SIMATIC avec éjecteur
3. ③ module d’entrées et sorties intégrées.

Ce module se compose de 10 entrées et 6 sorties digital (TOR) intégrées, figure de ses 6 sorties on trouve 2 sorties qu’ont une impulsion de 2.5 KHZ, aussi deux voies de comptage rapide 10 KHZ et de mesure avec codeurs incrémentaux 24v chaque voie contient doux entrée l’un est utilisée pour le comptage\décomptage, et l’autre pour déterminer le sens.

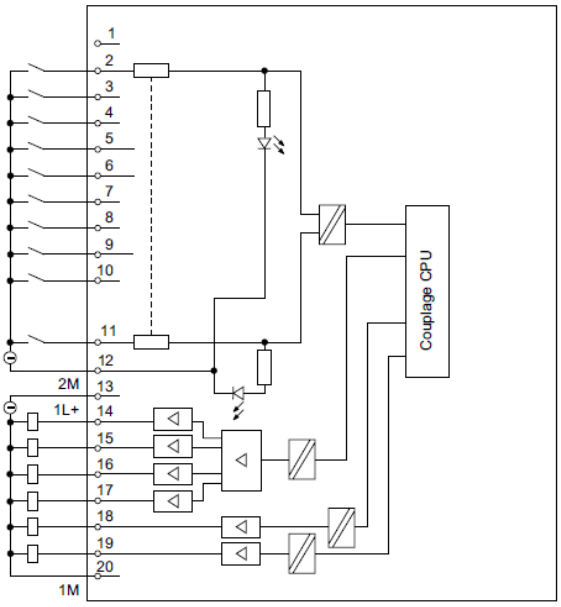


Figure 2.17: Schéma de connexion de la périphérie TOR intégrée.

④ Raccordement de la tension d'alimentation

⑤ Interface (MPI) L'interface multipoint (MPI) est l'interface de la CPU avec un PG/OP ou pour la communication dans un sous-réseau MPI. (**Figure 2.18**)

Figure 2.18 : adaptateur MPI.

Ce système de bus a été principalement développé comme interface programmable. MPI sert toutefois aussi à la communication avec les composants mis en place pour 'Servir & Visualiser' ainsi qu'à une communication homogène entre les automates.

L'interface multipoint MPI (Multipoint Interface) est une des interfaces de communication intégrée SIMATIC S7 dans de nombreux appareils, connectés simultanément à plusieurs outils de programmation/ PC avec STEP 7, les systèmes HMI (Operator Panel/Operator Station), S7-300, M7-300, S7-400 et M7-400. Elle peut être mise en place pour de simples mises en réseau.

La vitesse de transmission par défaut pour toutes les CPU est de 187,5 kbauds. Pour la communication avec un S7-200, on peut également régler 19,2 kbauds. Des vitesses de transmission allant jusqu'à 12 Mbauds au maximum sont possibles pour les

CPU 315-2 PN/DP, CPU 317-2 et pour la CPU 319-3 PN/DP

La CPU envoie automatiquement à l'interface MPI ses paramètres de bus réglés. Ainsi, une console de programmation peut, par exemple, avoir les bons paramètres et se connecter automatiquement à un sous-réseau MPI.

⑥ Commutateur de mode de fonctionnement**:** le réglage de mode de fonctionnement actuel de la CPU via le commutateur de mode de fonctionnement.

Dans la CPU 312c, il y'a trois positions de commutations, Position de fonctionnement RUN pour commencer de traiter le programme, et position de fonctionnement STOP pour que le CPU arrête le traitement, il y'a aussi une position de fonctionnement MRES pour l'effacement générale de la CPU.

2.3.3 Les modules analogiques

Les valeurs analogiques pour toutes les plages de mesure ou de sorties pouvant être utilisées avec les modules analogiques sont représentées dans cette partie.
La CPU ne traite que des valeurs analogiques.

a Le module d'entrée analogique

Le module d'entrées analogiques convertit le signal de process analogique en un signal numérique.
Le type du module qu'on a utilisé **SM 331 AI 2*12 bits. (Figure 2.19)**

Figure 2.19 : Représentation du module.

a.1 Caractéristiques

● 2 entrées formant 1 groupe de voies

● Type de mesure réglable pour chaque groupe de voies :

– Tension

– Courant

– Résistance

– Température

● Résolution réglable par groupe de voies (9/ 12/ 14 bits + signe)

● Sélection de la plage de mesure au choix par groupe de voies

● Diagnostic paramétrable et alarme de diagnostic

● Surveillance de limite paramétrable pour une voie

● Alarme de processus paramétrable pour alarme de limite

● Séparation galvanique par rapport à la CPU et à la tension de charge.

a.2 Résolution

La résolution de la valeur de mesure dépend directement de la période d'intégration sélectionnée. Elle est d'autant plus précise que la période d'intégration choisie pour une voie d'entrées analogiques est longue.

a.3 Brochage

Les figures suivantes montrent des exemples de branchement. Les résistances d'entrée dépendent de la plage de mesure sélectionnée.

Raccordement : mesure de tension

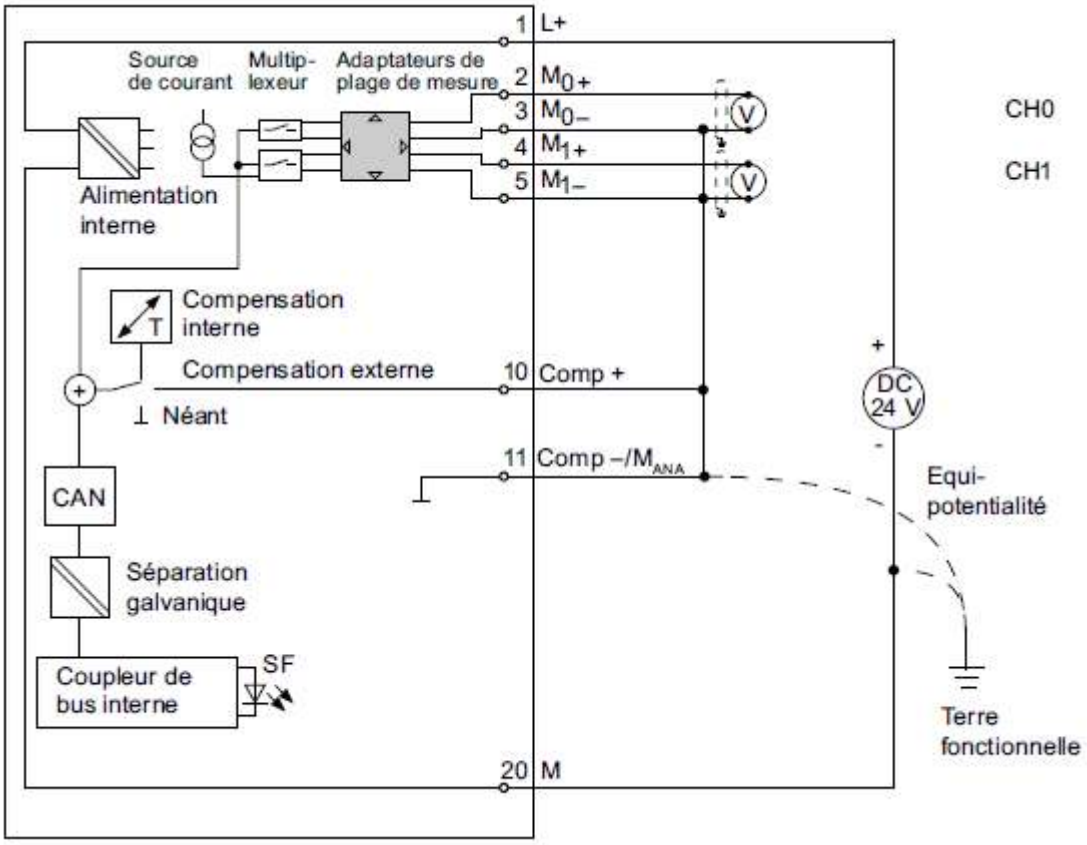


Figure 2.20 : Schéma de branchement et de principe.

Réglage de l'adaptateur de plage de mesure

L'adaptateur contient quatre positions, deux A et B pour la tension et les autres C et D pour le courant

Remarque : chaque position de l'adaptateur représente une plage de mesure.

Plage de mesure	Position de l'adaptateur de plage de mesure
± 80 mV ± 250 mV ± 500 mV ± 1000 mV	A
± 2,5 V ± 5 V de 1 à 5 V ± 10 V	B

Tableau 2.1 : plages des mesures de la tension.

Plage de mesure		Position de l'adaptateur de plage de mesure
Transducteur de mesure 2 fils	de 4 à 20 mA	D
Transducteur de mesure 4 fils	± 3,2 mA ± 10 mA de 0 à 20 mA de 4 à 20 mA ± 20 mA	C

Tableau 2.2 : plages des mesures de courant.

a.4 Types et plages de mesure

Le module SM 331; AI 2 x 12 bits dispose d'un adaptateur de plage de mesure (**Figure 2.21)**. Les types et les plages de mesure sont configurés via l'adaptateur et par le paramètre "Type de mesure" dans TIA PORTAL. Le module est réglé par défaut sur le type de mesure "Tension" et la plage de mesure ± 10 V. on peut utiliser ce type de mesure avec cette plage de mesure sans paramétrer le SM 331 ; AI 2 x 12 bits avec TIA PORTAL.

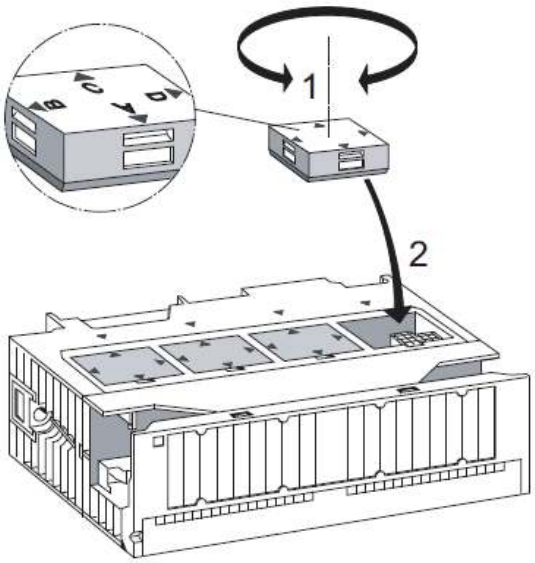


Figure 2.21 : Enfichage de l'adaptateur dans le module d'entrées analogiques.

Pour modifier le type de mesure et la plage de mesure, il faut le cas échéant modifier la position de l'adaptateur .En outre, les réglages nécessaires sont sérigraphies sur le module. Sur la porte avant, marquez la position de l'adaptateur de plage de mesure (**Figure 2.22)**.

Pour réglé la plage de mesure en tension (±10V) il suffit de positionné l'adaptateur en position B.

b Le module de sortie analogique

Un module de sorties analogiques convertit un signal de sorties numériques en un signal analogique.

Le type du module qu'on a utilisé **SM 332 AO 2*12 bits.** (**Figure 2.22)**.

Figure 2.22 : représentation du module.

b.1 caractéristiques

● 2 sorties dans un groupe

● Les sorties sont sélectionnables voie par voie en tant que

– Sortie de tension

– sortie de courant

● Résolution 12 bits

● Diagnostic paramétrable et alarme de diagnostic

● Avec séparation galvanique par rapport au couplage de bus interne et à la tension d'alimentation

● prend en charge la fonction reparamétrage en MARCHE

b.2 Brochage

Les **figures** 2.23 et 2.24 montrent des exemples de branchement.

Connecteur : montage 2 et 4 fils pour sortie de tension

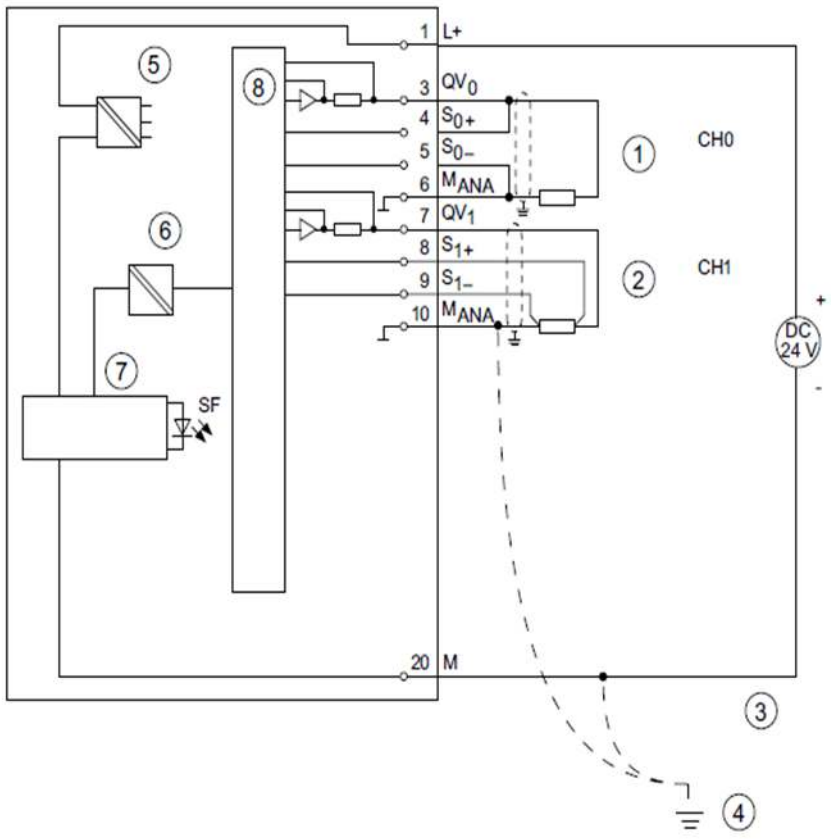


Figure 2.23 : Schéma de branchement et de principe

Avec :

① Montage 2 fils: sans compensation de la résistance de câble

② Montage 4 fils: avec compensation de la résistance de câble

③ Equipotentialité

④ Terre fonctionnelle

⑤ Alimentation interne

⑥ Séparation de potentiel

⑦ Coupleur de bus interne

⑧ Convertisseur numérique-analogique

Connecteur pour sortie de courant

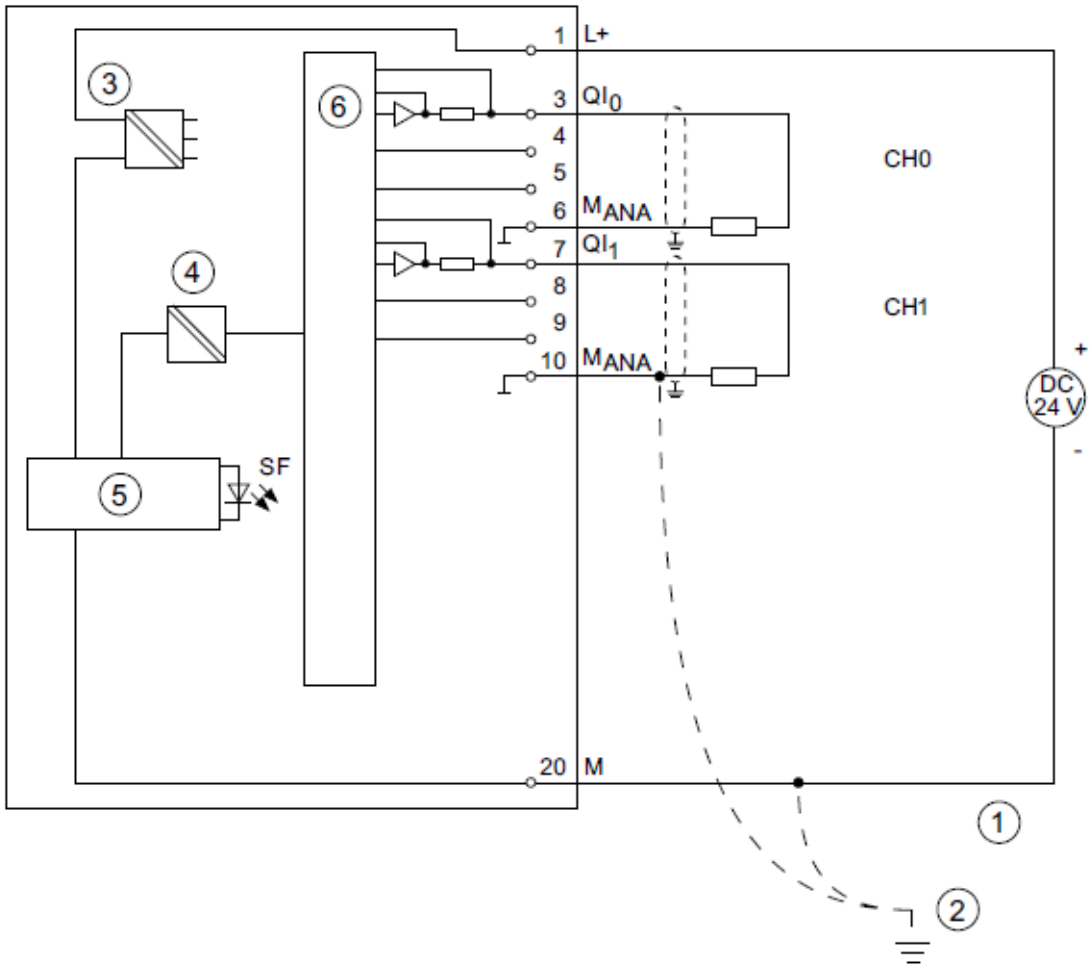


Figure 2.24 : Schéma de branchement et de principe

Avec :

① Equipotentialité

② Terre fonctionnelle

③ Alimentationb interne

④ Séparation de potentiel

⑤ Coupleur de bus interne

⑥ Convertisseur numérique-analogique

Plages de sortie du module SM 332 ; AO 2 x 12 bits

Le paramétrage de sortie peut être comme sorties de tension ou de courant ou désactivé. Le paramètre "type de sortie" dans TIA PORTAL donne les choix de l'utilisation de la sortie.

Les plages de sortie autorisées pour les sorties de tension et de courant sont réglées au moyen de TIA PORTAL.[8]

Type de sortie sélectionné	Plage de sortie
tension	de 1 à 5 V de 0 à 10 V ± 10 V
courant	de 0 à 20 mA de 4 à 20 mA ± 20 mA

Tableau2.3 : Plages de sortie du module SM 332 ; AO 2 x 12 bits.

2.3.4 Schéma de câblage

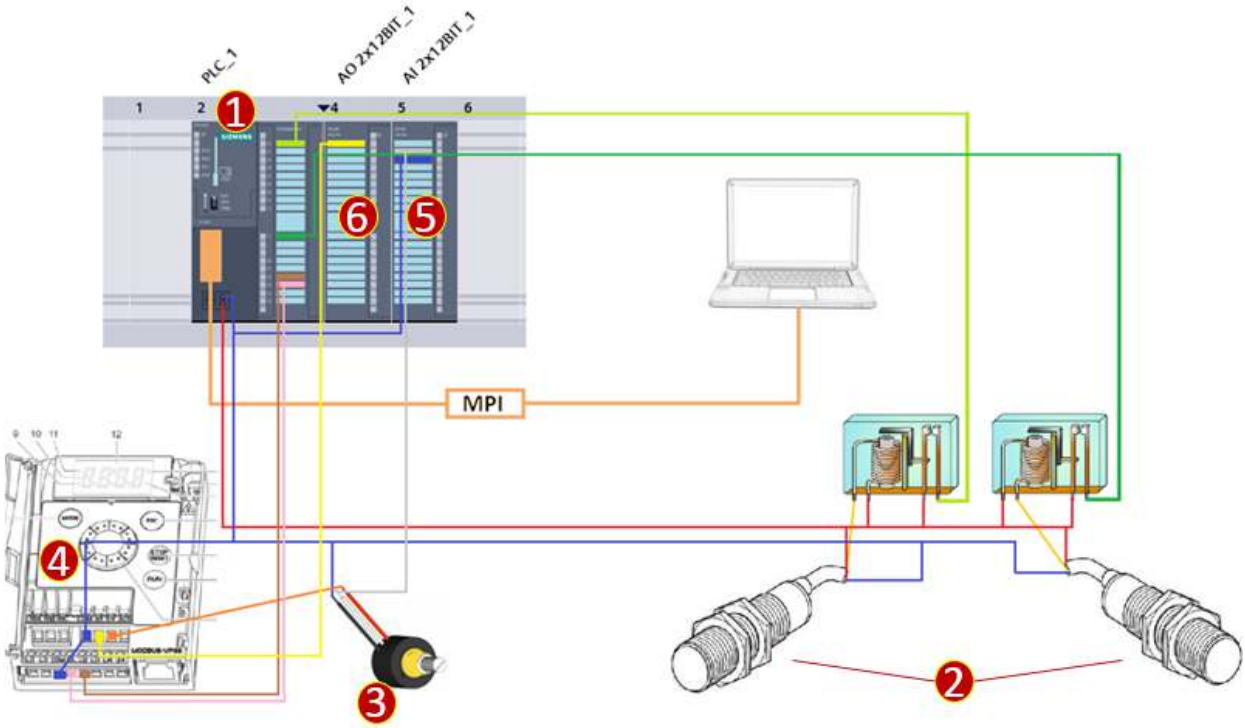


Figure 2.25 : schéma de câblage.

On commence d'abord, par les modules de l'API qui sont mis en place de la façon suivante :

①**.API** : Le module CPU vient en première position parce qu'on n'a pas besoin d'un module d'alimentation placée sur un rack et connectée aux deux modules entrées et sorties analogiques qui sont interconnectée respectivement voir figure 2.25, et alimentée à travers le module CPU.

②**. Photocellules** : les deux fils d'alimentation 24v sont relié au transformateur et le fil d'information passe par un relais reliée qui donne une tension de sortie 0v ce

qui donne un 1 et 24v pour un 0 reliée à une entrée TOR située dans le module CPU qui est de logique négative.

③**. Potentiomètre**: est alimenté par 5v ce qui représente le seuil max du potentiomètre à travers le variateur de fréquence et deux fils d'information vers le module d'entrée analogique, l'un et relié à la borne 2 et l'autre à la borne 3 vers la masse.

④. **variateur de vitesse** : alimenté en monophasé 230 v et alimente le moteur en 230v triphasé par les sorties U/T1 V/T2 W/T3 GND du variateur vers les bornes du moteur respectivement U V W et GND couplé en étoile, on commande la vitesse du moteur en reliant l'entrée analogiques du variateur AI vers le module sortie analogique, et les deux entrée LI1 et LI2 logique du variateur sont utilisé pour contrôlé le sens de rotation du moteur relié à deux sortie TOR du module CPU.

⑤**. le module d'entrée analogique** : on a paramétré le module AI 2*12 bits en tension entre 1 et 5V, les borne 2 et 3 réservé au potentiomètre connecté avec le module de sortie analogique par un coupleur de 20 points**. Figure 2.26.**

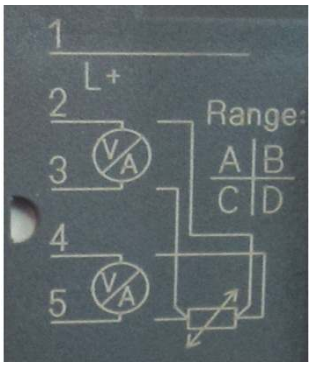


Figure 2.26 : schéma de fonctionnement de AI.

⑥**. le module de sortie analogique** : on a paramètre le module AO 2*12 bits en tension entre 0 et 10V, les borne 3 et 6 réservé au variateur de vitesse avant de les brancher on fait un court-circuit entre la borne 3 et 4 et aussi entre 5 et 6 pour compenser la résistance de câble c'est q dire diminuer l'erreur. **Figure 2.27.**

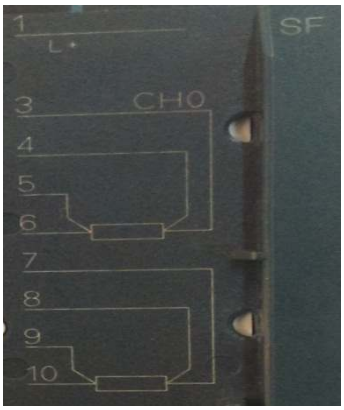


Figure 2.27 : schéma de fonctionnement de AO.

2.3.5 SIMATIC HMI

C'est une technologie de siemens pour l'interface homme machine qui est conçu pour répondre aux besoins des processus de plus en plus complexes des machines et des systèmes. SIMATIC HMI est optimisé pour satisfaire les besoins spécifiques en matière d'interface homme machine utilisant des interfaces ouvertes et standardisées dans le matériel et les logiciels, pour une intégration efficace de système d'automatisation.

a Appareils de commande

Pourvu de nombreuses fonctions et disponible dans différentes classes de performance. Chaque SIMATIC panel permet une conduite et supervision efficaces de l'installation au pied de la machine.

b Logiciel runtime basés sur pc

L'exécutif (runtime) est embarqué sur les pupitres operateur SIMATICHMI. Les fonctionnalités HMI et les capacités fonctionnelles dépendent de la configuration matérielle de l'appareil.

Deux visions de SIMATIC WinCC ruuntime sont disponibles pour les plateformes PC :

- SIMATIC WinCC Runtime Advanced.
- SIMATIC WinCC Runtime Professional.

c Logiciel de visualisation SIMATIC WinCC Runtime Advanced

Solution de conduite et supervision sur PC pour systèmes monopostes au pied de machines.

1) Pack de base pour la visualisation, signalisation et journalisation, gestion des utilisateurs; extensible de façon flexible par des scripts VB
2) Pack de base extensible par l'utilisation des packs optionnels
3) Intégrable dans des automatismes basés sur des réseaux TCP/IP
4. Concepts étendus de maintenance avec fonctions de conduite, diagnostic, administration à distance via Internet/Intranet en liaison avec une communication par courrier électronique

d Logiciel de visualisation SIMATIC WinCC Runtime Professional

Système de conduite et supervision sur base PC pour procédé, opérations de fabrication, machines et installations de tous les secteurs - des systèmes monopostes simples aux systèmes multipostes avec serveurs redondants et solutions réparties multi-sites avec clients Web.

- Fonctions industrielles pour la signalisation et l'acquittement d'événements, l'archivage de messages et de valeurs de mesure, la journalisation des données du procédé et de configuration, la gestion des utilisateurs ; extensible de façon flexible par des scripts VB et C
- Pack de base extensible par l'utilisation des packs optionnels
- Des API pour l'exécutif permettant l'utilisation des interfaces ouvertes de programmation sont également comprises.[9]

e Solution PC :

On utilise la solution PC comme un écran de supervision à travers des logiciels runtime Advanced ou bien Professional, la solution PC possède plusieurs avantages, parmi ces avantages l'utilisation d'une webcam pour visualiser un système en temps réel, la supervision de plusieurs machines simultanément.

2.3.5 Montage de l'automate :

Le montage se fait en 3 parties :

• correctement monter et mettre le profilé support à la terre.

• monter les modules sur le profilé support.

• mettre en place le connecteur frontal.

a Montage et mise à la terre du profilé support :

D'abord on fixe le profilé support ou le rack sur l'embase en utilisant deux vis de taille M6, après on Relie le profilé support au conducteur de protection figure 2.28.

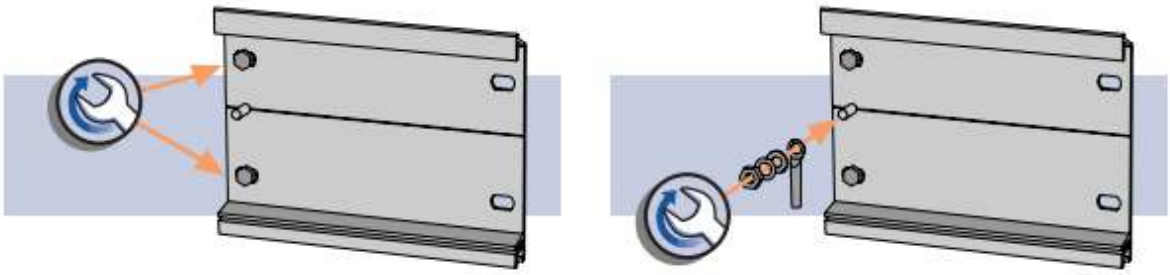


Figure 2.28 : fixation et mise à la terre du rack.

bMontage des modules sur le rack :

En général on Monte l'alimentation dans le premier emplacement mais dans notre cas l'alimentation est remplacer par un transformateur, ensuite on fixe la CPU 312C dans l'emplacement N°2 et aussi on fixe de la même manière les modules dE/S, figure 2.29.

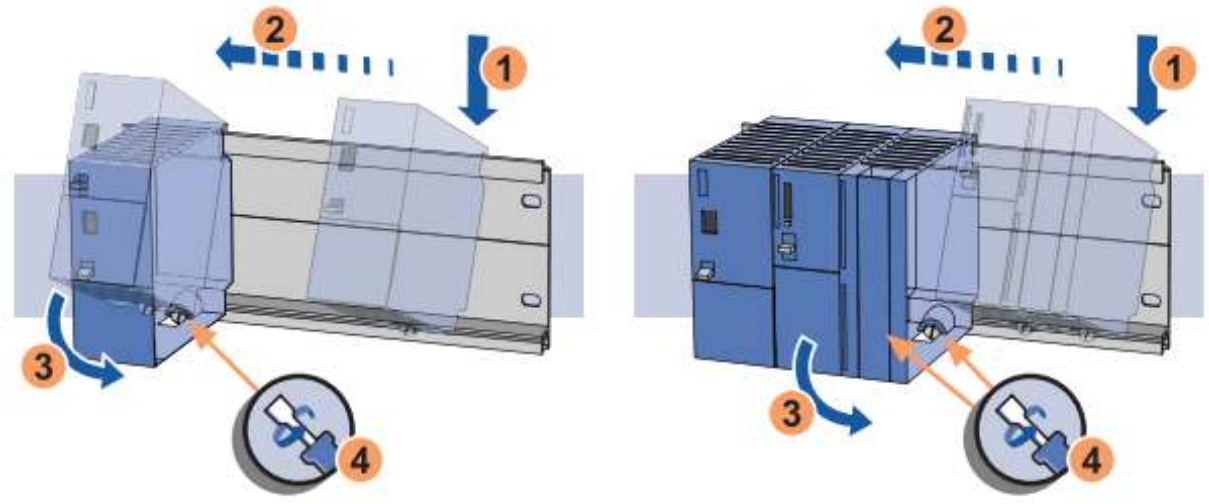


Figure 2.29 : montage des modules.

c insertion du connecteur frontale :.

On insère le connecteur frontal dans les modules DE/S Jusqu'à ce qu'il s'enclenche. Dans cette position de câblage, le connecteur frontal ressort du module et n'a pas de contact avec ce dernier.

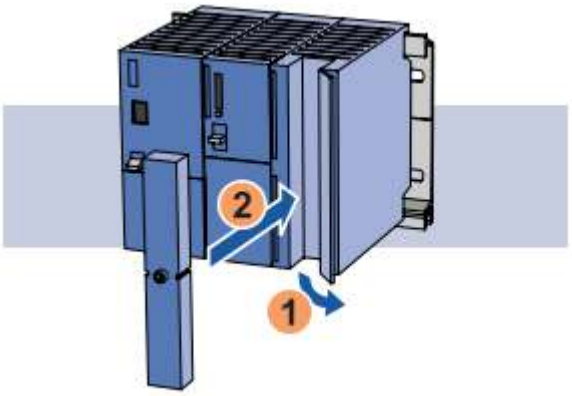


Figure 2.30 : insertion du connecteur frontale.

2.4 Conclusion

On a pu voir dans ce chapitre les différents composants électriques et électroniques un par un et leurs câblages.

Chapitre 3 développement du schéma fonctionnel sur API et IHM

3.1 Introduction

Le présent chapitre illustre les instructions suivi pour élaborer le programme et la supervision de notre projet par l'intermédiaire du logiciel TIA PORTAL.

3.2 Présentation du TIA PORTAL

Le TIA Portal est la clé ouvrant l'accès au potentiel intégral de la Totally Integrated Automation. Le logiciel optimise l'ensemble des procédures au niveau planification, machine et processus. Son interface utilisateur intuitive, ses fonctions simples et sa transparence totale des données le rendent extrêmement convivial. Des données et projets déjà existants peuvent être intégrés aisément, ce qui garantit la sécurité de l'investissement.[10]

3.3 Création d'un projet

La création d'un projet commence toujours par la configuration du matériel, elle revient à lister tous les modules présents dans le projet. Par exemple la CPU, les entrées-sorties, les modules de communications, etc…

Tous ces éléments se trouvent dans la bibliothèque du projet.

Pour ouvrir le logiciel TIA PORTAL. On suit la trajectoire suivante :

On clique sur Démarrer > Programmes > Siemens Automation > TIA-Portal V13, ou bien sur leur raccourci. **Figure3.1**

Figure3.1 : logo de TIA PORTAL

Pour créer un nouveau projet il faut d'abord Cliquer sur « Créer un projet », on définit le nom du projet et l'endroit où il doit être sauvé en fin on clique sur « créer ». Figure3.2

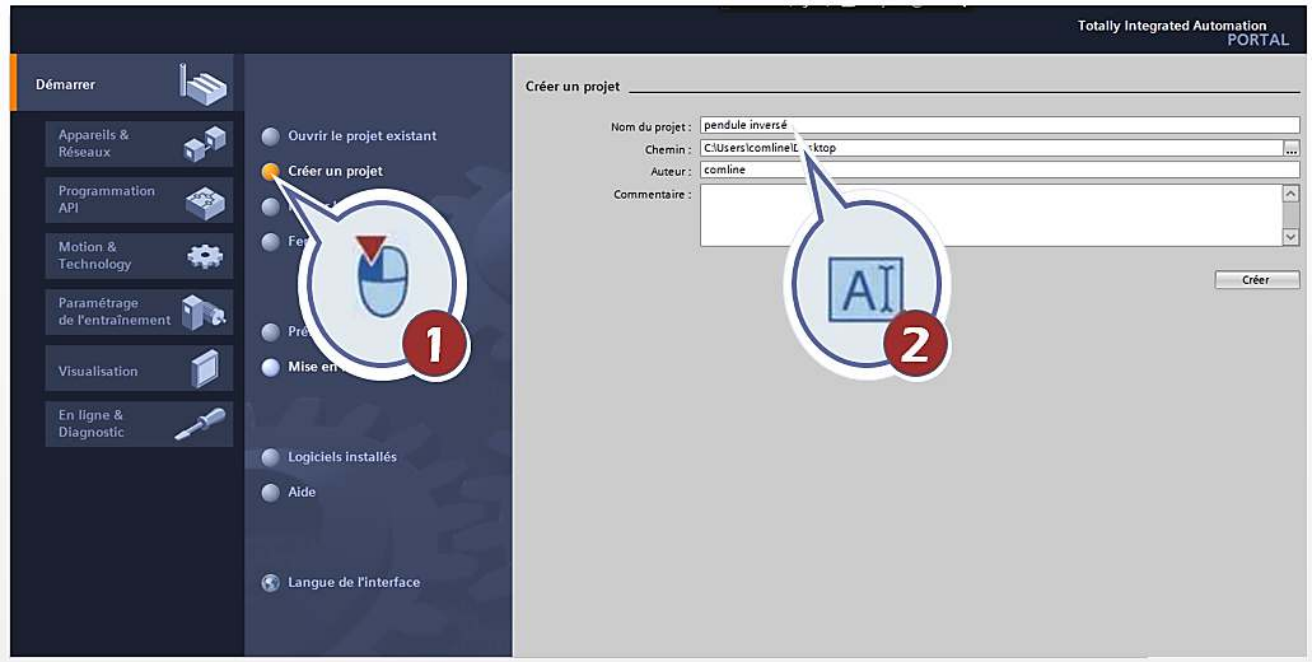


Figure3.2 : Création d'un nouveau projet

Dans notre cas on a créé un projet sous le nom de «pendule inversé».

Lorsque le projet est créé, on clique sur « Vue du projet » pour développer le programme.

3.4Configuration de l'automate

Pour insérer la CPU, on clique sur "ajouter un appareil" et le renommer, on Sélectionne « Contrôleur » et la CPU adéquate. Figure3.3

Figure3.3 : Configuration de l'automate

Pour notre projet on a choisi la CPU312C vue sa disponibilité, leur référence est de 6ES7 312-5BF04-0AB0 qui contient un Module D'E/S TOR intégrer, les CPU les plus développée sont beaucoup plus adapté à ce type de réalisation. Voir figure3.3.

Apres avoir validé on trouve automatiquement la CPU dans l'emplacement Nº2 du châssis (le rack) et son module D'E/S dans l'emplacement Nº3, "figure 3.4", mais pour les autres emplacements la configuration des modules se fait manuellement, on a enfiché un module d'entrée analogique AI 2x12BIT dans l'emplacement Nº5 et un module de sortie analogique AO 2x12BIT_1 dans l'emplacement Nº4, figure3.5

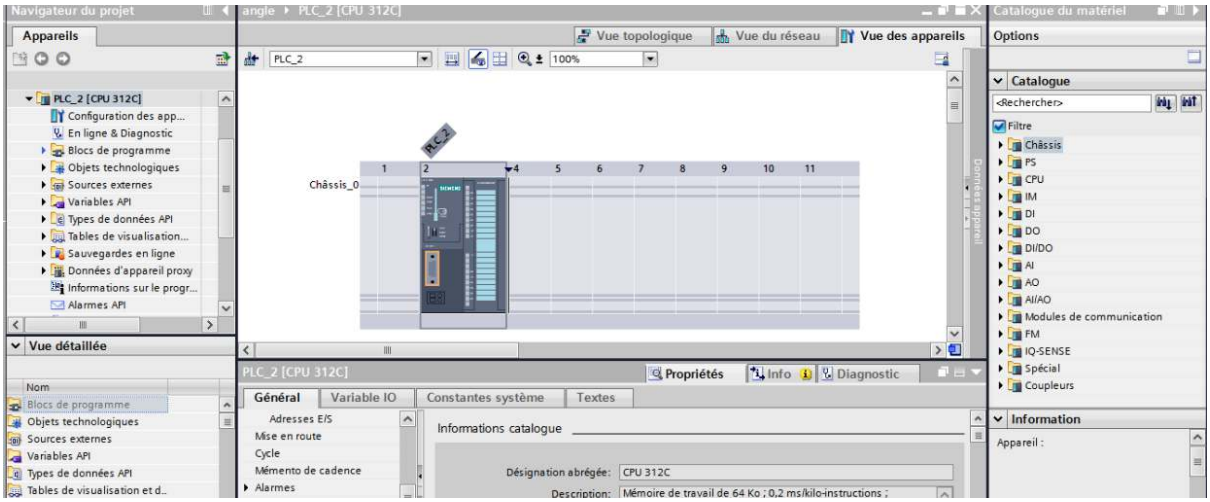


Figure 3.4 :l'emplacement de l'API sur le châssis.

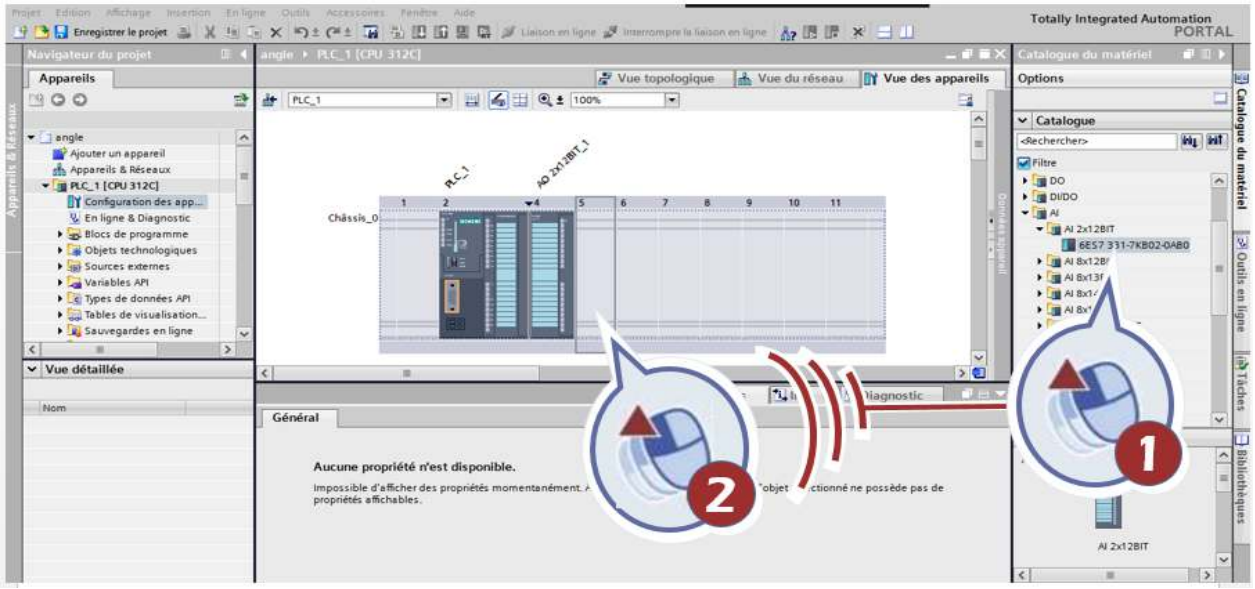


Figure3.5 :insertion des modules externes.

3.5 Table de mnémoniques

Avant de commencer à programmer, il faut démarrer par la table des mnémoniques, après on saisit les variables qu'on aura besoin dans le programme. figure3.6

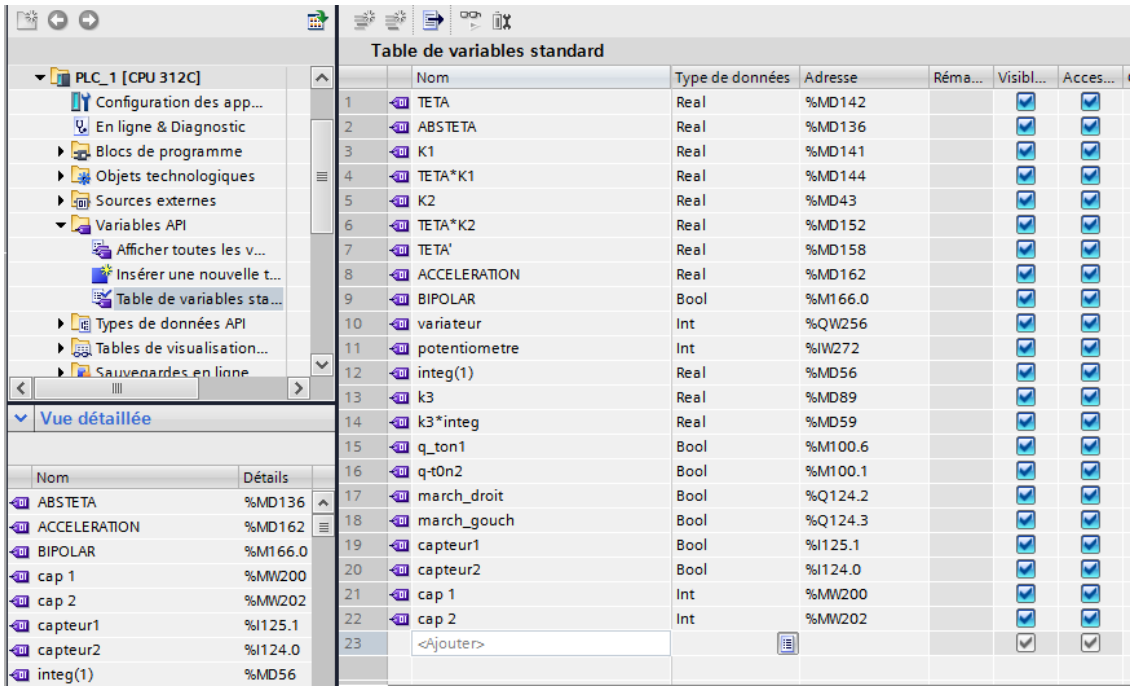


Table de variables standard

	Nom	Type de données	Adresse	Réma...	Visibl...	Acces...
1	TETA	Real	%MD142		☑	☑
2	ABSTETA	Real	%MD136		☑	☑
3	K1	Real	%MD141		☑	☑
4	TETA*K1	Real	%MD144		☑	☑
5	K2	Real	%MD43		☑	☑
6	TETA*K2	Real	%MD152		☑	☑
7	TETA'	Real	%MD158		☑	☑
8	ACCELERATION	Real	%MD162		☑	☑
9	BIPOLAR	Bool	%M166.0		☑	☑
10	variateur	Int	%QW256		☑	☑
11	potentiometre	Int	%IW272		☑	☑
12	integ(1)	Real	%MD56		☑	☑
13	k3	Real	%MD89		☑	☑
14	k3*integ	Real	%MD59		☑	☑
15	q_ton1	Bool	%M100.6		☑	☑
16	q-t0n2	Bool	%M100.1		☑	☑
17	march_droit	Bool	%Q124.2		☑	☑
18	march_gouch	Bool	%Q124.3		☑	☑
19	capteur1	Bool	%I125.1		☑	☑
20	capteur2	Bool	%I124.0		☑	☑
21	cap 1	Int	%MW200		☑	☑
22	cap 2	Int	%MW202		☑	☑
23	<Ajouter>				☑	☑

figure3.6: Table des variables de l'API.

Les types de variables qu'on a utilisés sont décrits dans les tableaux suivant :

Type symbolique	*Type*	*Valeur max*	*Mémoire requise*
bo	BOOL	TRUE ou « 1 »	1 bit
by	BYTE	255	8 bits
wo	WORD	65535	16 bits
dw	DWORD	4294967295	32 bits

Tableau 3.1 : types des variables.

Genre de variable	*Type*	*Exemple*
I	Entrée Physique	%I 10.0
Q	Sortie physique	%Q 4.7
M	Mémento	%M 50.2
T	Temporisation	%T
C	Compteur	%C1
MB	Byte,Char	%MB204
ID	DInt,DWord,Real	%ID90
MW	Word	%MW168
IW	Date,Int,S5Time	%IW67
MD	Real	%MD156
QW	Sortie analogique	%QW256
IW	Entrée analogique	%IW272

Tableau 3.2 : genre de variable.

Dans les systèmes automatisés, la notion de bit, byte, Word ou double Word est souvent utilisée.la signification de ces termes est :

Bit : Un bit correspond à une position ou un caractère binaire. Il s'agit de la plus petite unité en matière d'information et elle n'admet que l'un des deux états suivants : "0" ou "1". Dans notre cas le bit représente une entrée ou sortie physique (TOR) ou bien un memento.

Byte (ou octet): Un byte est une unité de 8 bits. Il est utilisé par exemple pour regrouper les états logiques de 8 entrées ou de 8 sorties pour une entrée ou sortie analogique.

Word (ou Mot) : Un Word se compose de 2 bytes, soit 16 bits. Un mot permet de regrouper 16 entrées ou 16 sorties. Leur utilisation permet d'augmenter la précision

Double Word (ou double mot) : Un double Word se compose de 2 mots, ou 4 bytes, ou 32 bits. Un double mot est la plus grande unité qu'un automate soit capable de traiter. Le

Remarque :

Pour écrire ou lire les valeurs des modules externes il faut ajouter la lettre P dans la syntaxe des adressages physiques. Figure 3.7

Figure 3.7 :la lettre P des addresses E/S externes.

3.6 Développement du programme

On trouve automatiquement le bloc d'organisation OB1 après la configuration matérielle, en cliquant deux fois sur OB1, on commence à programmer en (CONT), Figure3.8

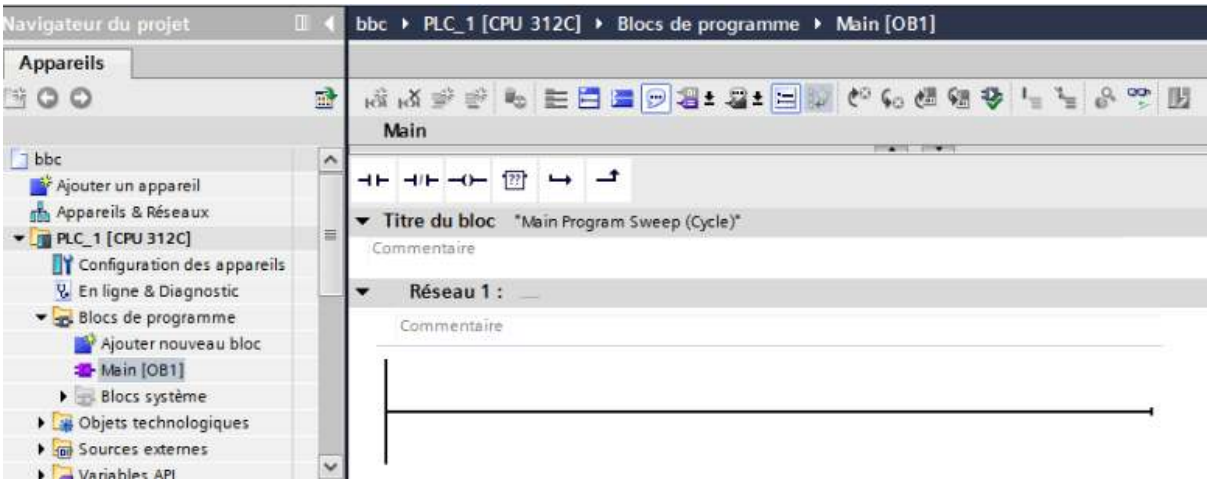
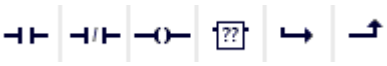


Figure3.8: le bloc d'organisation OB1.

On a Utilisé les icones suivantes pour programmer 

On appuie sur « nouveau réseau »  pour créer un nouveau.

3.6.1 Structure du programme

Le programme est structuré autour de 16 réseaux.

Réseau 1 : la valeur renvoyé par le potentiomètre.

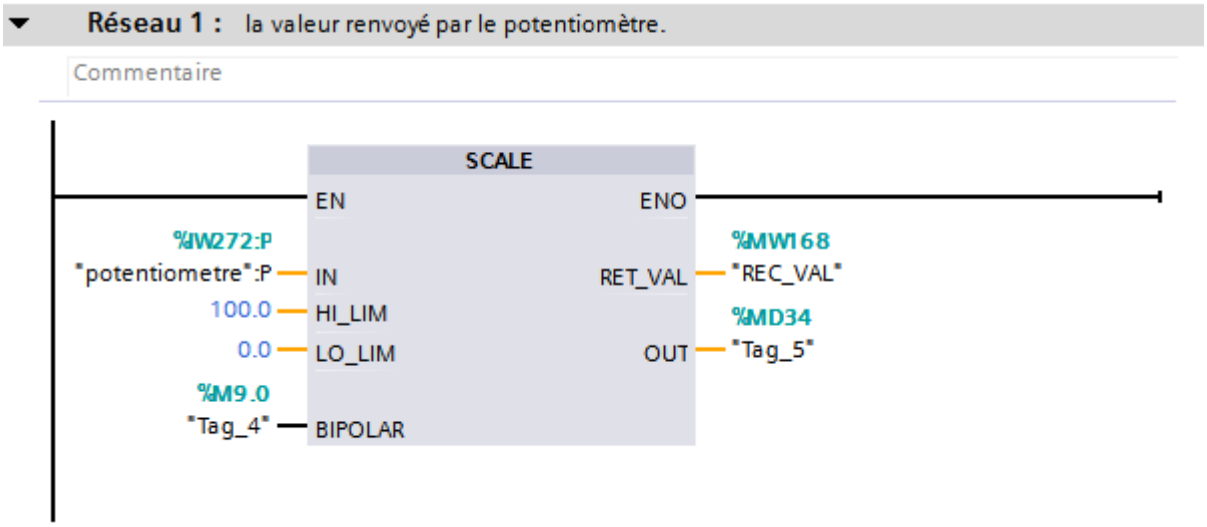


Figure 3.9 : réseau 1 de bloc OB1

IW272 : P : recoie la valeur du potentiomètre.

Paramètres du bloc scale :

Le tableau suivant montre les paramètres de l'instruction "UNSCALE"

Paramètre	Déclaration	Type de données	Zone de mémoire	Description
EN	Input	BOOL	I, Q, M, D, L	Entrée de validation
ENO	Output	BOOL	I, Q, M, D, L	Sortie de validation
IN	Input	INT	I, Q, M, D, L, P ou constante	Valeur d'entrée à mettre à l'échelle
HI_LIM	Input	REAL	I, Q, M, D, L, P ou constante	Valeur limite supérieure
LO_LIM	Input	REAL	I, Q, M, D, L, P ou constante	Valeur limite inférieure
BIPOLAR	Input	BOOL	I, Q, M, D, L ou constante	Indique si la valeur du paramètre IN est interprétée comme bipolaire ou unipolaire. Ce paramètre peut prendre les valeurs suivantes : 1: bipolaire 0: unipolaire
OUT	Output	REAL	I, Q, M, D, L, P	Résultat de l'instruction
RET_VAL	Output	WORD	I, Q, M, D, L, P	Informations d'erreur

Tableau 3.3: Paramètres du bloc scale.

Paramètre RET_VAL

La signification des valeurs du paramètre RET_VAL :

Le code d'erreur est en hexadécimal **(W#16#...).**

Le nombre 0000 indique qu'il n y 'a aucune erreur, et le nombre 0008 indique que la valeur IN est supérieure à la valeur de la constante K2 ou inferieur a la valeur de la constante K1.

Le fonctionnement du bloc scale :

L'instruction scale ou "Mise à l'échelle" permet de convertir le nombre entier du paramètre IN en un nombre à virgule flottante mis à l'échelle en unités physiques comprises entre une valeur limite inférieure et une valeur limite supérieure. On définit la valeur limite inférieure et supérieure de la plage de valeurs sur laquelle la valeur d'entrée est mise à l'échelle par le biais des paramètres LO_LIM et HI_LIM. Le résultat de l'instruction est inscrit dans le paramètre OUT.

L'instruction "Mise à l'échelle" utilise l'équation suivante :

$$OUT = [((FLOAT\ (IN) - K1)/\ (K2 - K1)) * (HI\_LIM - LO\_LIM)] + LO\_LIM$$

Les valeurs des constantes "K1" et "K2" sont déterminées par l'état logique du paramètre BIPOLAR.

Le paramètre BIPOLAR peut prendre les états logiques suivants :

- Etat logique "1" : On suppose que la valeur dans le paramètre IN est bipolaire et se situe dans une plage de valeurs allant de -27 648 à 27 648. Dans ce cas, la constante "K1" a la valeur "-27 648,0" et la constante "K2" la valeur "+27 648,0".

- État logique "0" : On suppose que la valeur dans le paramètre IN est unipolaire et se situe dans une plage de valeurs allant de 0 à 27 648. Dans ce cas, la constante "K1" a la valeur "0,0" et la constante "K2" la valeur "+27 648,0".

Lorsque la valeur au paramètre IN est supérieure à celle de la constante "K2", le résultat de l'instruction prend la valeur de la limite supérieure (HI_LIM) et une erreur est signalée.

Lorsque la valeur au paramètre IN est inférieure à celle de la constante "K1", le résultat de l'instruction prend la valeur de la limite inférieure (LO_LIM) et une erreur est signalée.

Lorsque la valeur limite inférieure indiquée est supérieure à la valeur limite supérieure (LO_LIM > HI_LIM), le résultat est mis à l'échelle de manière inversement proportionnelle à la valeur d'entrée

On a utilisé le bloc scale pour convertir la tension du potentiomètre en nombre real |binaire de 12 bits|.[11]

Réseau 2 :l'angle en fonction de la valeur du potentiomètre

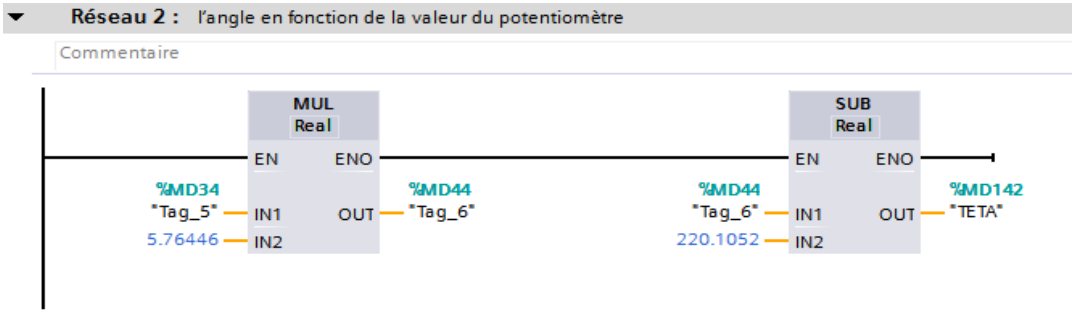
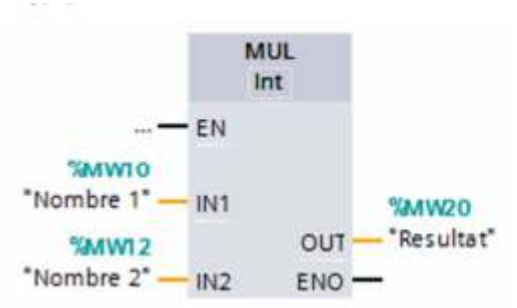


Figure 3.10: réseau 2 de bloc OB1.

Les opérations mathématiques :

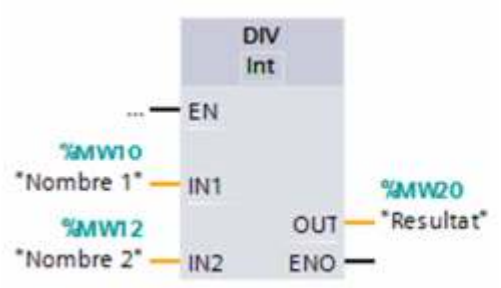
a. Multiplication : Mul



Résultat = Nombre 1 x Nombre 2

Figure 3.11 : opération de multiplication.

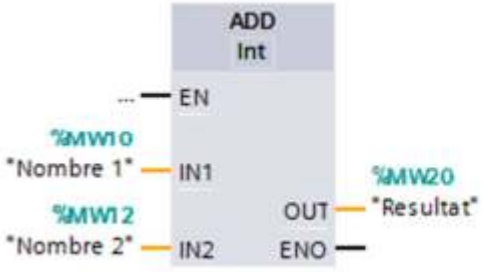
b.Division : Div



Résultat = Nombre 1 : Nombre 2

Figure 3.12: opération de la division.

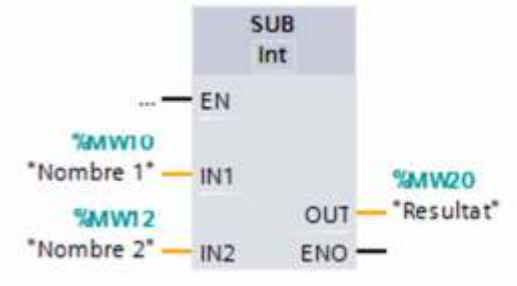
c.Addition : ADD



Résultat = Nombre 1 + Nombre 2

figure 3.13.: opération d'addition.

d.Soustraction : SUB



Résultat = Nombre 1 - Nombre 2

Figure 3.14: opération de soustraction.

Après avoir choisi l'opération de base, le format du nombre doit être déterminé :

• « Int » pour un entier.

• « DInt » pour un double entier.

• « Real» pour un real.

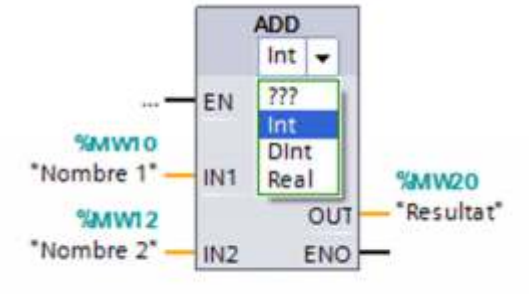


Figure 3.15: le choix de type d'addition.

Dans le Réseau 2 on a une équation de l'angle en fonction de la valeur du potentiomètre $\Theta = a \times IN + b$ tel que a=5,76446 et b=220,1052.

Avec : tag 6= tag 5*5.76446 et téta=tag 6*220.1052.

Le but de cette équation est de fixé l'angle vertical en 0.

Réseau 3 : sens droit du moteur.

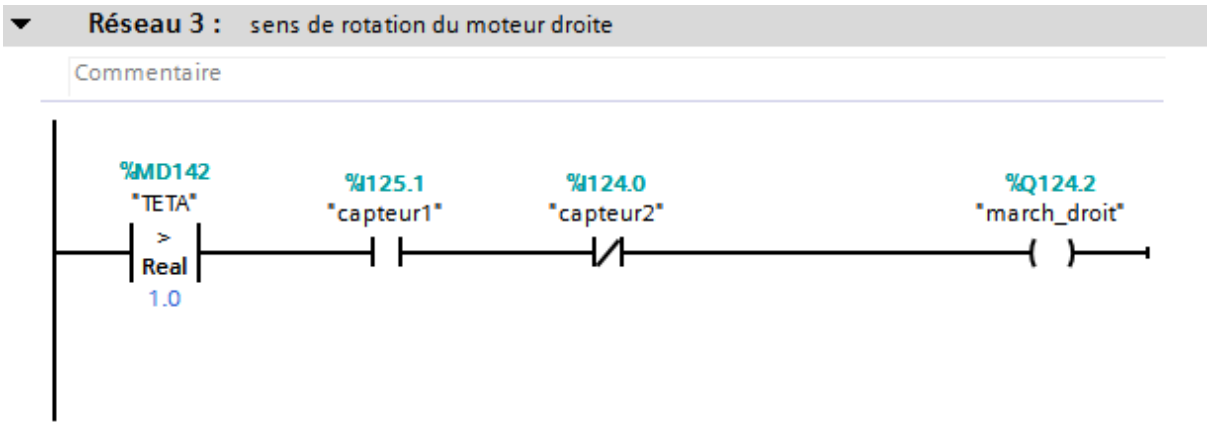


Figure 3.16 :réseau 3 de bloc OB1.

L'action du chariot vers la droite sous trois conditions, la première c'est de la divination de l'angle doit être >1, la deuxième du capteur 1 qu'elle s'active lors de la détection l'autre il faut être en état de repos.

Les opérations de comparaison se font entre les nombres entier ou double entier ou real, Le résultat est binaire (0 ou 1).

Exemple : Si « TETA » > « 1.0 » la sortie « Etat » passe à « 1 », sinon elle reste à « 0 ».

Les opérations de comparaison possible :

CMP ==	Egal à
CMP <>	Différent de
CMP >=	Supérieur ou égal à
CMP <=	Inférieur ou égal à
CMP >	Supérieur à
CMP <	Inférieur à

Figure 3.17 : table de comparaison

Dans ce réseau on commande le sens du moteur à partir de l'entrée LI1 du variateur de vitesse reliée avec la sortie logique dont l'adresse est de %Q124.2 réservée au sens droit.

Réseau 4 : sens gauche du moteur

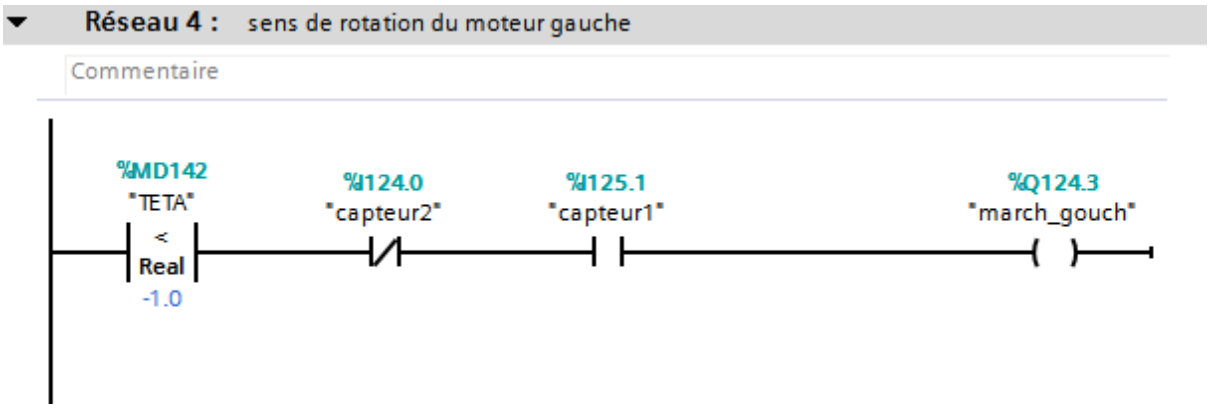


Figure 3.18 :réseau 4 de bloc OB1.

Dans ce réseau on commande le sens du moteur à partir de l'entrée LI2 du variateur de vitesse reliée avec la sortie logique dont l'adresse %Q124.3 réservée au sens gauche.

Réseau 5 : relation de k1 et k2 avec téta.

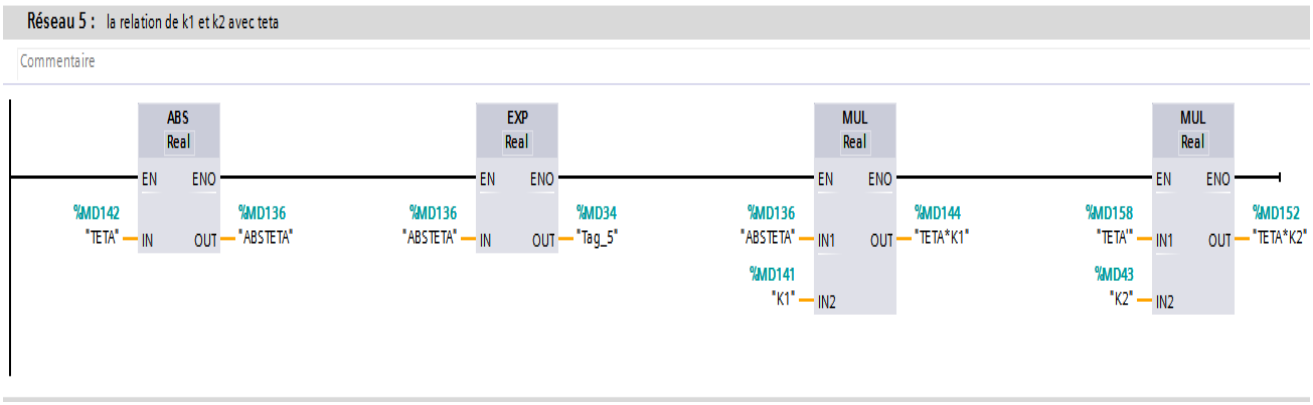


Figure3.19 : réseau 5 de bloc OB1.

Ce réseau est réservé aux opérations mathématiques dont on a besoin pour l'équation finale de l'accélération avec correcteur PID qui se trouve dans le réseau 11.

Réseau 6 :calcul du dérivé de l'angle

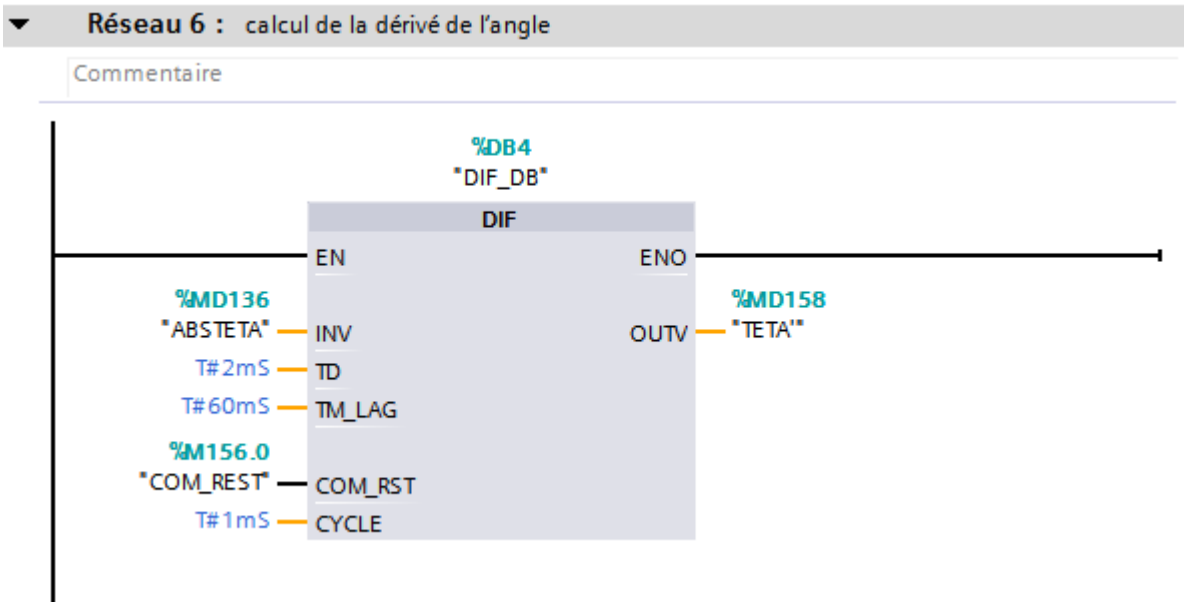


Figure 3.20 :réseau 6 de bloc OB1.

TD Temps de dérivation : 2ms

TM_LAG Temps de retard : 1s

INV0 Hauteur d'échelon de l'entrée

INV Grandeur d'entrée, dans notre cas on a dérivée téta (l'angle)

OUTV Grandeur de sortie : téta'

Description du dérivateur DIF :

Les grandeurs du processus sont dérivées de manière dynamique. Il est ainsi possible de calculer la vitesse à partir du chemin parcouru. Le dérivateur s'utilise pour la compensation de perturbation, la commande pilote et la configuration d'un régulateur.

Fonctionnement :

Cette instruction dérive la grandeur d'entrée selon le temps, lisse le signal avec un élément de retard du 1er ordre et réalise la fonction de transfert suivante dans le domaine de Laplace:

- OUTV(s) / INV(s) = TD / (1+TM_LAG*s)

La réponse temporelle est déterminée par le temps de dérivation TD et le temps de retard TM_LAG. La figure 3.2 montrent la réponse indicielle de DIF.

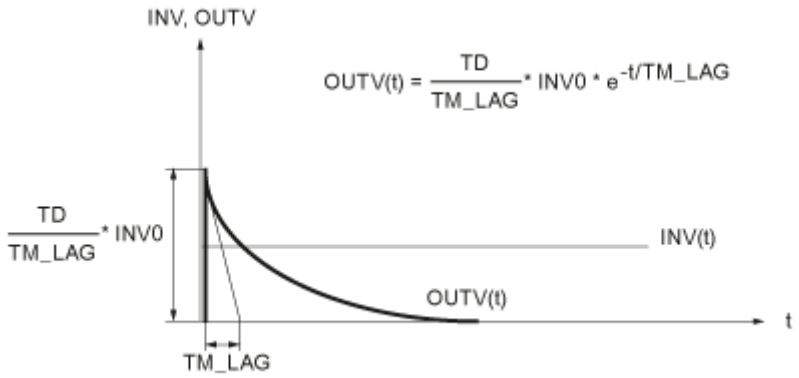


Figure 3.21 :la réponse indicielle de DIF.

DIF fonctionne sans retard si on a paramétré TM_LAG ≤CYCLE/2. Un échelon de l'entrée est appliqué à la sortie avec le facteur TD**/**CYCLE. Après un cycle, la sortie revient à 0.Figure3.22.[12]

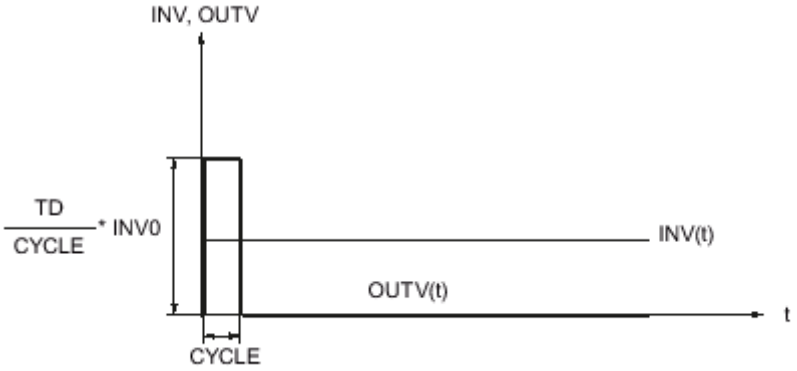


Figure 3.22 : la réponse si TM_LAG ≤CYCLE/2.

Réseau 7/8 : horloge de l'intégrateur de l'angle

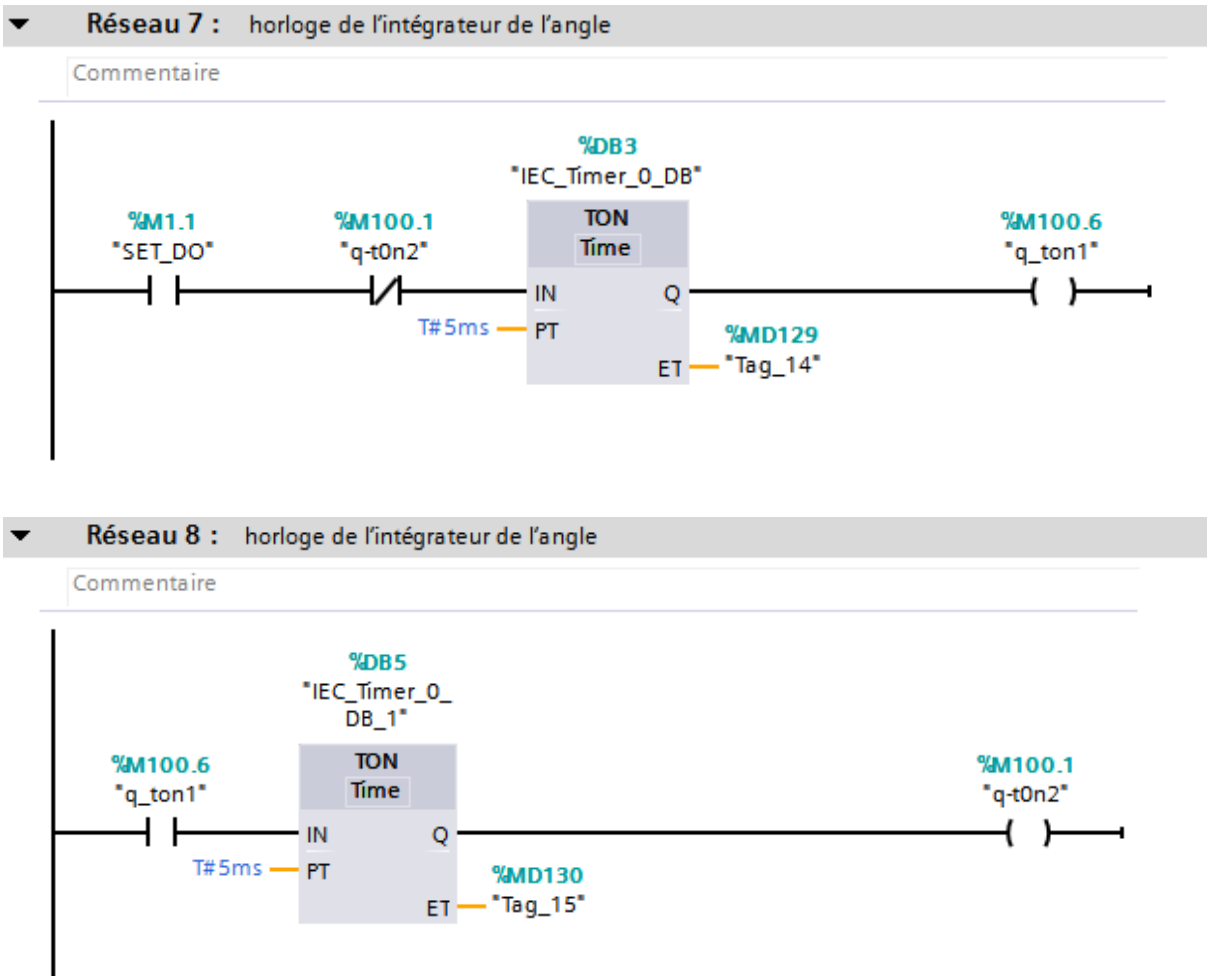


Figure 3.23 : les réseaux 7 et 8 du bloc OB1.

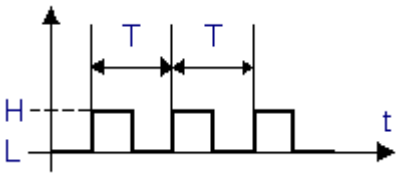


Figure3.24 : le signal d'horloge.

Le signal d'horloge de l'intégrateur est forme par les deux temporisateurs périodiquement avec une période T de 10ms et H=1, L=0.Figure3.24.

temporisation TON Retard à la montée:

L'instruction "**TON**" vous permet de retarder la mise à 1 de la sortie Q de la durée programmée PT. L'instruction est démarrée lorsque le résultat logique (RLO) à l'entrée IN passe de "0" à "1" (front montant). La durée PT programmée débute au démarrage de l'instruction. Une fois la durée PT écoulée, la sortie Q fournit l'état logique "1". La sortie Q reste à 1 tant que l'entrée de démarrage fournit "1". Lorsque l'état logique à l'entrée de démarrage passe de "1" à "0", la sortie Q est remise à 0. La fonction de temporisation est redémarrée lorsqu'un nouveau front montant est détecté à l'entrée de démarrage.

La valeur de temps actuelle peut être demandée à la sortie ET. La valeur de temps débute à T#0s et se termine lorsque la durée PT est atteinte. La sortie ET est remise à 0 dès que l'état logique à l'entrée IN passe à "0".

Il faut associer à chaque appel de l'instruction "Retard à la montée" une temporisation CEI dans laquelle les données de l'instruction sont stockées.[13]

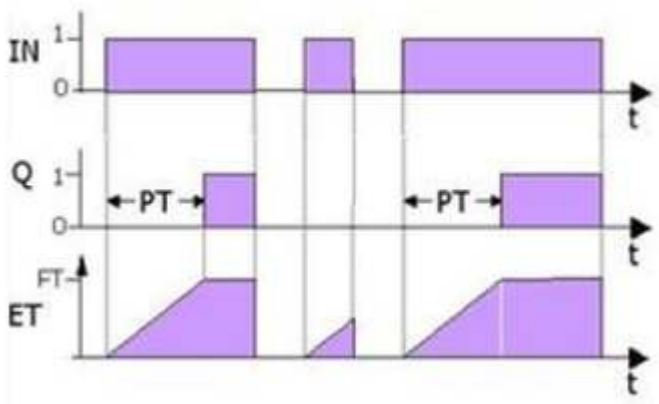


Figure 3.25 : le fonctionnement de temporisateur TON

Paramètres du temporisateur TON :
Le tableau suivant montre les paramètres de l'instruction "Retard à la montée" :

Paramètres	**Déclaration**	**Type de données**	**Zone de mémoire**	**Description**
IN	Input	BOOL	I, Q, M, D, L	Entrée de démarrage
PT	Input	TIME	I, Q, M, D, L ou constante	Durée d'un retard à la montée La valeur du paramètre PT doit être positive.
Q	Output	BOOL	I, Q, M, D, L	Sortie mise à 1 après écoulement de la durée PT.
ET	Output	TIME	I, Q, M, D, L	Valeur de temps actuelle

Tableau 3.4 : les Paramètres du temporisateur TON.

Réseau 9 : calcul de l'intégral de l'angle

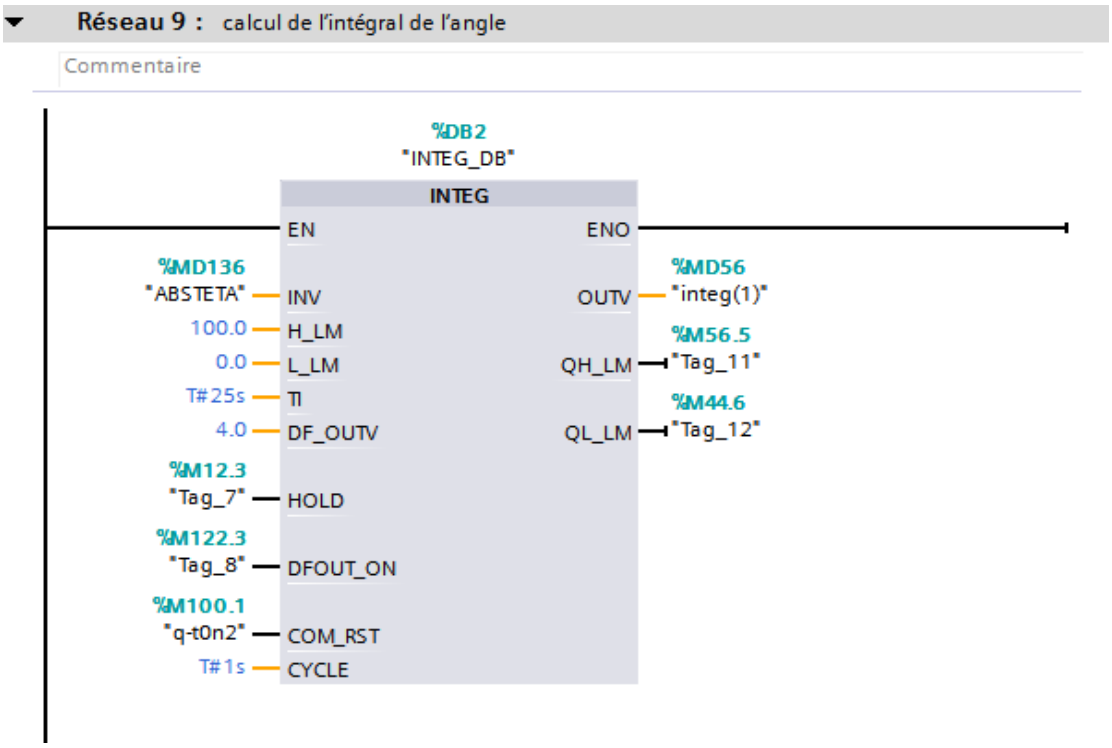


Figure 3.26 : réseau 9 du blocOB1.

T1 Temps d'intégration : 25s

INV0: Hauteur d'échelon de l'entrée

t: Temps

INV Grandeur d'entrée : on a choisi l'entrée comme une valeur absolue de teta.

OUTV: Grandeur de sortie, tout comme la mémoire de l'action I sont limitées dans les paramètres H_LM =100 et L_LM=0. Si la sortie se situe hors de la plage L_LM...H_LM, les bits de signalisation QH_LM ou QL_LM prennent la valeur TRUE.

Figer l'intégrateur

Avec HOLD = TRUE, l'action I reste sur sa valeur de sortie momentanée OUTV. En cas de retour à HOLD = FALSE, l'intégration reprend à partir de la valeur de sortie momentanée OUTV.

Description de INTEG :

Les grandeurs du processus sont intégrées de manière dynamique. Il est ainsi possible de calculer le chemin parcouru à partir de la vitesse. L'instruction s'utilise pour la configuration d'un régulateur à action I.

Fonctionnement

Cette instruction intègre la grandeur d'entrée selon le temps et limite l'intégrale à une valeur inférieure et supérieure prédéterminée. La limitation de la grandeur de sortie est indiquée par des bits de signalisation. Instruction fonctionne comme suit :

Fonctions	DFOUT_ON	HOLD
Intégration	FALSE	FALSE
Figer l'action I	FALSE	TRUE
Paramétrer par défaut la sortie	TRUE	Indifférente

Tableau 3.5 : les instructions de l'intégrateur.

- **Intégration**

Pour l'intégration, l'instruction réalise la fonction de transfert suivante dans le domaine de Laplace :

OUTV(s) / INV(s) = 1 / (TI*s)

La réponse temporelle de l'action I est déterminée par le temps d'intégration TI. La réponse indicielle correspondante est présentée dans la figure suivante.[14]

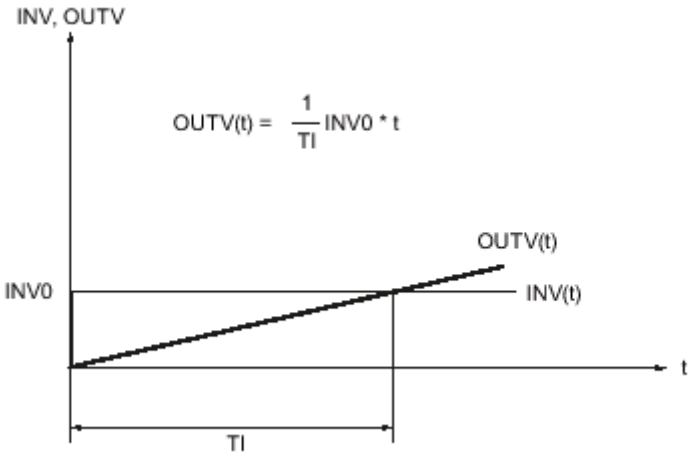


Figure 3.27 : La réponse indicielle.

Réseau 10 action de l'intégrateur

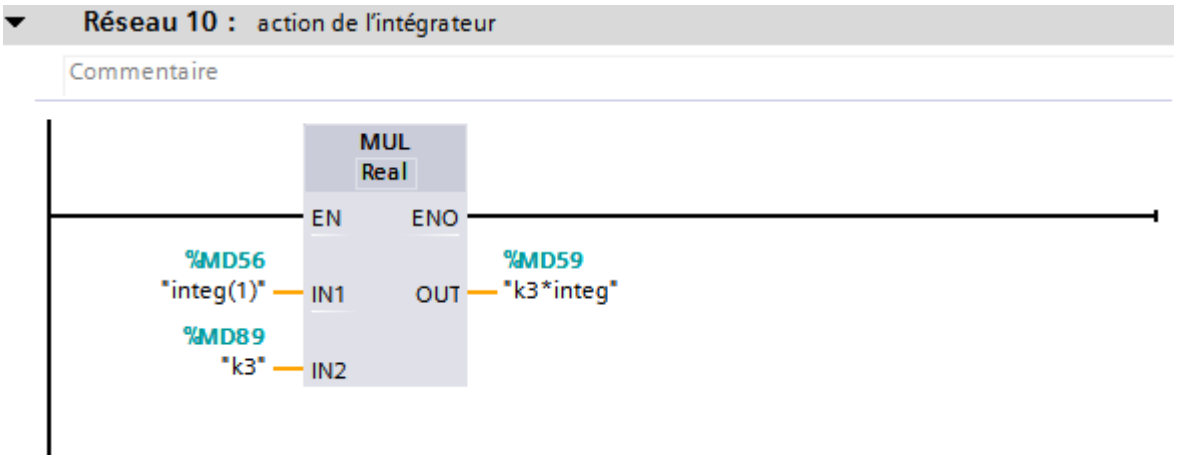


Figure 3.28: réseau 10 du bloc OB1.

La dernière valeur de l'intégrateur est multipliée par k3 (KI) pour obtenir l'équation finale du bloc intégrateur qui représente l'action d'intégrateur sur l'accélération.

Réseau 11 fonction de l'accélération

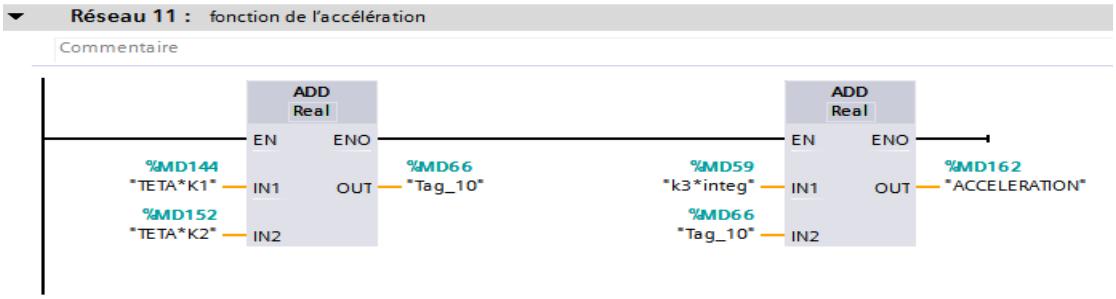


Figure 3.29 : réseau 11 du bloc OB1.

Dans ce réseau on cherche la valeur de l'accélération (régulateur PID) en fonction des paramètres K1*teta+K2*teta'+K3*$\int teta$, avec K1= Kp, K2=Kd, K3=Ki.

Ki , Kp , Kd sont les paramètres de bloc PID.

Réseau 12 : la valeur absolue de l'accélération

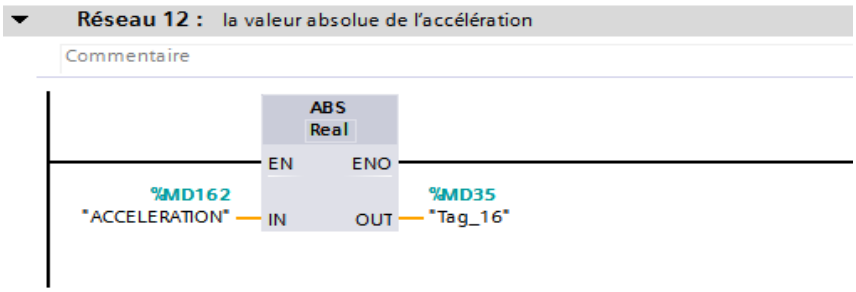


Figure 3.30 : réseau 12 du bloc OB1.

Réseau 12 spécifié pour la valeur absolue de l'accélération.

Réseau 13 la commande du moteur

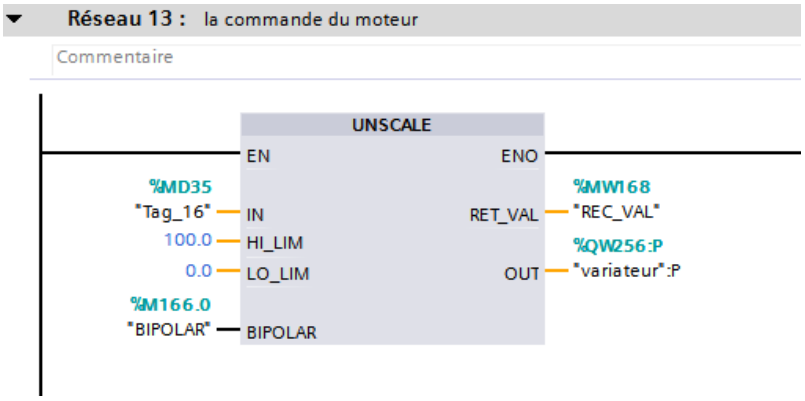


Figure 3.31 : réseau 13 du bloc OB1.

Paramètres

Le tableau suivant montre les paramètres de l'instruction "Annuler mise à l'échelle" :

Paramètre	Déclaration	Type de données	Zone de mémoire	Description
EN	Input	BOOL	I, Q, M, D, L	Entrée de validation
ENO	Input	BOOL	I, Q, M, D, L	Sortie de validation
IN	Input	REAL	I, Q, M, D, L, P ou constante	Valeur d'entrée dont la mise à l'échelle doit être annulée pour obtenir une valeur entière.
HI_LIM	Input	REAL	I, Q, M, D,L,P ou constante	Valeur limite supérieure
LO_LIM	Input	REAL	I, Q, M, D, L, P ou constante	Valeur limite inférieure
BIPOLAR	Input	BOOL	I, Q, M, D, L ou constante	Indique si la valeur du paramètre IN est interprétée comme bipolaire ou unipolaire. Ce paramètre peut prendre les valeurs suivantes : 1: bipolaire0: unipolaire
OUT	Output	INT	I, Q, M, D, L, P	Résultat de l'instruction
RET_VAL	Output	WORD	I, Q, M, D, L, P	Informations d'erreur

Tableau3.5: les Paramètres de UNSCALE

Paramètre RET_VAL:

La signification des valeurs du paramètre RET_VAL :

Le code d'erreur est en hexadécimal **(W#16#...),** le nombre 0000 indique qu'il y 'a aucune erreur et le nombre 0008 indique que La valeur du paramètre IN est supérieure à la valeur de la limite supérieure (HI_LIM) ou inférieure à la valeur de la limite inférieure (LO_LIM).

UNSCALE : Annuler mise à l'échelle:

Description

Avec l'instruction "**UNSCALE** ", permet d'annulez la mise à l'échelle en unités physiques entre une valeur limite inférieure et une valeur limite supérieure du nombre à virgule flottante au paramètre IN et vous le convertissez en nombre entier. Vous définissez la valeur limite inférieure et supérieure de la plage de valeurs sur laquelle la mise à l'échelle de la valeur d'entrée est annulée par le biais des paramètres LO_LIM et HI_LIM. Le résultat de l'instruction est fourni au paramètre OUT.

L'instruction "Annuler mise à l'échelle" utilise l'équation suivante :

$OUT = [((IN – LO\_LIM) / (HI\_LIM – LO\_LIM)) * (K2 – K1)] + K1$

Les valeurs des constantes "K1" et "K2" sont déterminées par l'état logique du paramètre BIPOLAR. Le paramètre BIPOLAR peut prendre les états logiques suivants :

- Etat logique "1" : on suppose que la valeur du paramètre IN est bipolaire et se situe dans une plage de valeurs allant de -27648 à 27648. Dans ce cas, la constante "K1" a la valeur "-27648,0" et la constante "K2" la valeur "+27648,0".
- Etat logique "0" : on suppose que la valeur du paramètre IN est unipolaire et se situe dans une plage de valeurs allant de 0 à 27648. Dans ce cas, la constante "K1" a la valeur "0,0" et la constante "K2" la valeur "+27648,0".

Lorsque la valeur dans le paramètre IN est supérieure à la valeur de la limite supérieure ("HI_LIM"), le résultat de l'instruction prend la valeur de la constante "K2" et une erreur est renvoyée.

Lorsque la valeur dans le paramètre IN est inférieure à la valeur de la constante de la limite inférieure (LO_LIM), le résultat de l'instruction prend la valeur de la constante "K1" et une erreur est émise.[15]

Réseau 14/15 extrémités de la table

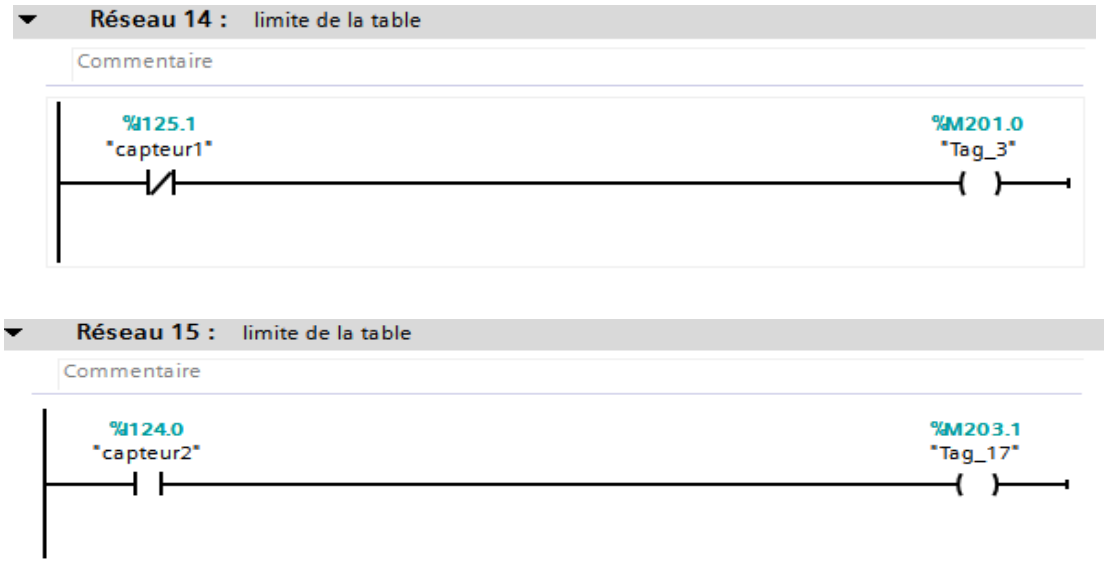


Figure 3.32 :réseau 13 du bloc OB1.

Les réseaux 14 et 15 réservé pour les photocellules, l'un s'occupe l'adresse physique I124.0 et l'autre dans l'adresse i125.1

Réseau 16 l'état du moteur

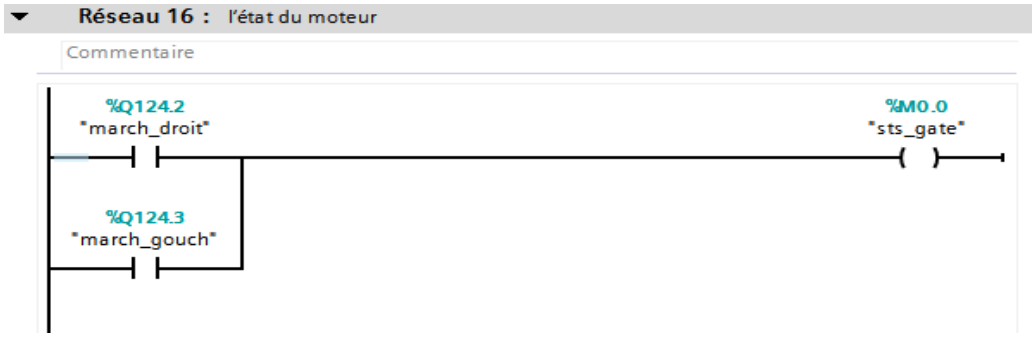


Figure 3.33 : réseau 16 du bloc OB1.

On réserve le réseau 16 pour afficher l'état du moteur en supervision.

3.6.2 Test du programme par simulation

Pour simuler le programme, on clique sur « Démarrer la simulation »  La fenêtre du simulateur s'ouvre : on doit configurer l'API avec les cartes et éventuellement des

zones mémoires. Pour ajouter des cartes d'E/S, cliquez sur les icônes

On change les adresses d'E/S pour correspondre à notre projet.

Par exemple :

On clique sur Charger pour passer à la simulation, charger le programme dans le PLC virtuel et lancer la simulation.

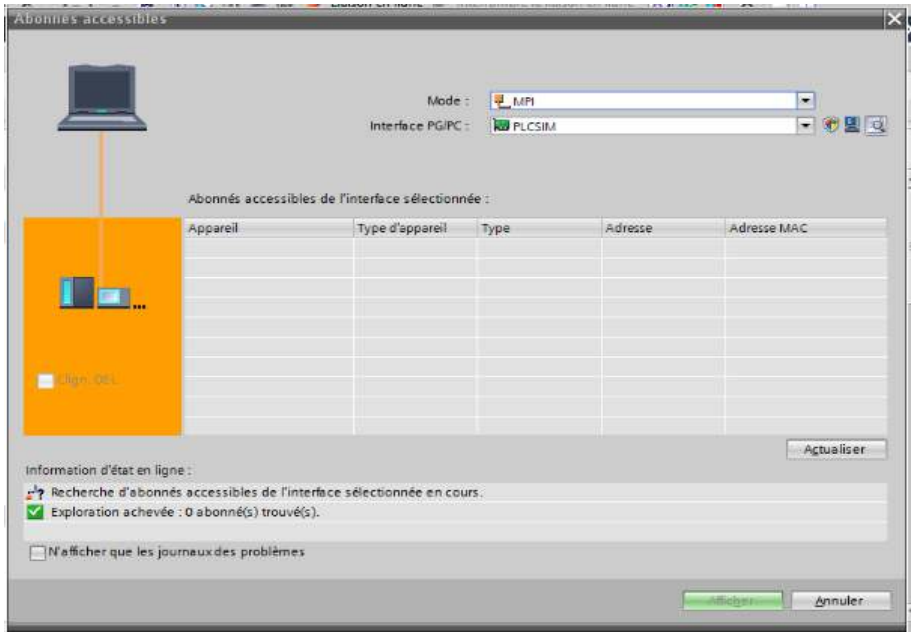


Figure3.34 : la liaison entre le PLCsim et le PC

Apres avoir chargé le programme dans le simulateur on peut tester notre programme facilement et utiliser les fenêtres d'E/S pour lire et écrire les valeurs analogique et logique comme si on a un API réel

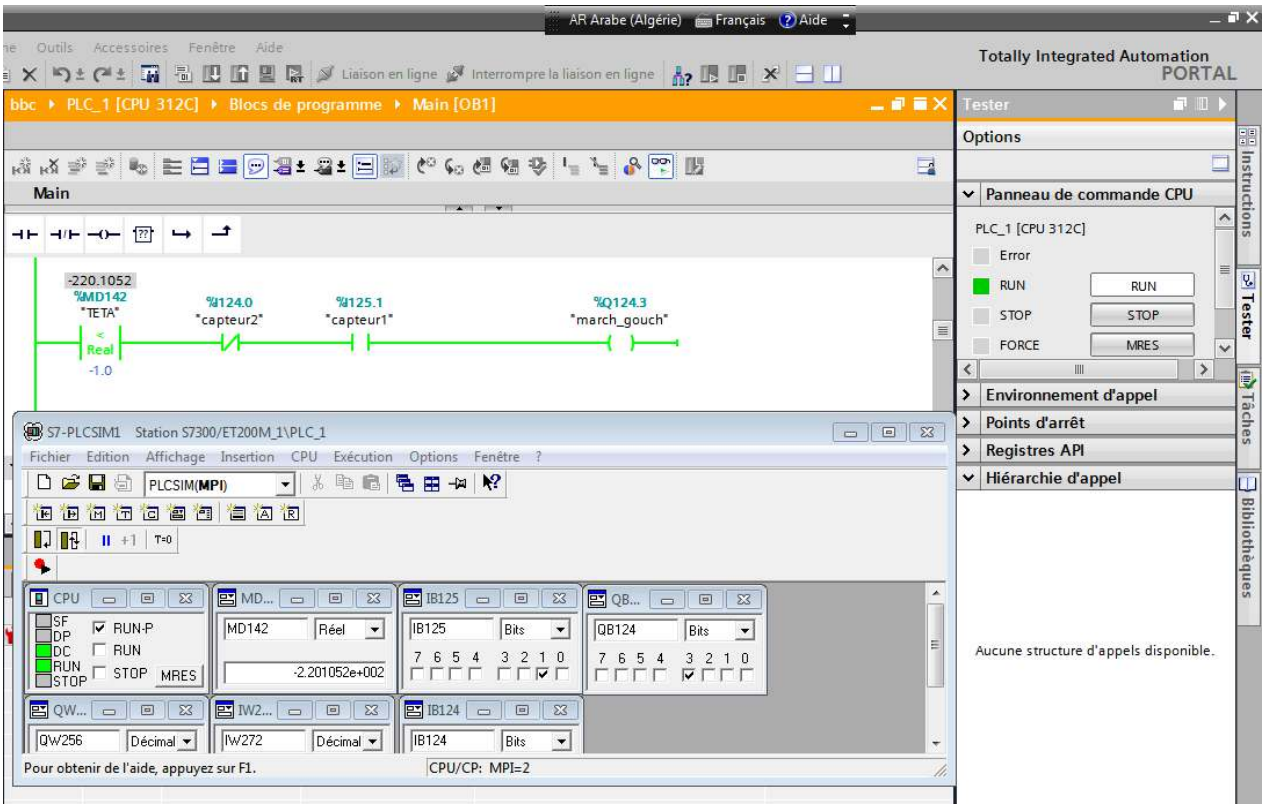


Figure 3.35 : la simulation du programme en PLCsim.

3.6.3 Test sur Plc réel (s7 300 cpu312c)

On transfère le programme et la configuration matérielle dans la mémoire de l'API.

On Place l'automate en RUN, avec le bouton situé sur l'unité centrale.

3.7 Programme sur le runtime advanced

3.7.1 Configuration de l'appareil

Pour insérer l'écran de supervision, on clique sur "ajouter un appareil" et le renommer, on Sélectionne « Systèmes PC » et on double clic sur" wincc runtime advanced ". Figure3.36.

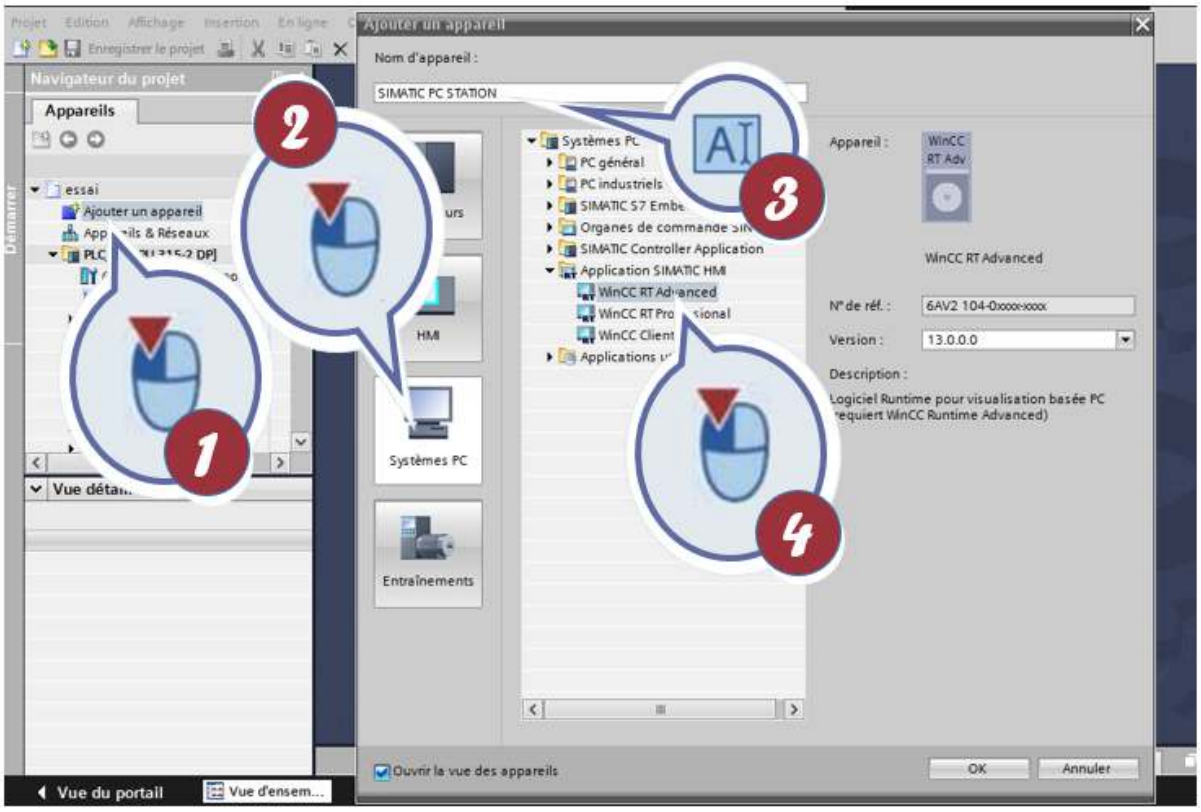


Figure 3.36 : la config matérielle de WINCC.

Apres avoir validé on trouve automatiquement l'écran HMI à l'emplacement Nº1 de la "station PC SIMATIC", on ajoute un module de communication qui contient un Interface MPI, pour faire une liaison entre l'écran de supervision et l'API réel qu'on a utilisée.figure3.37.

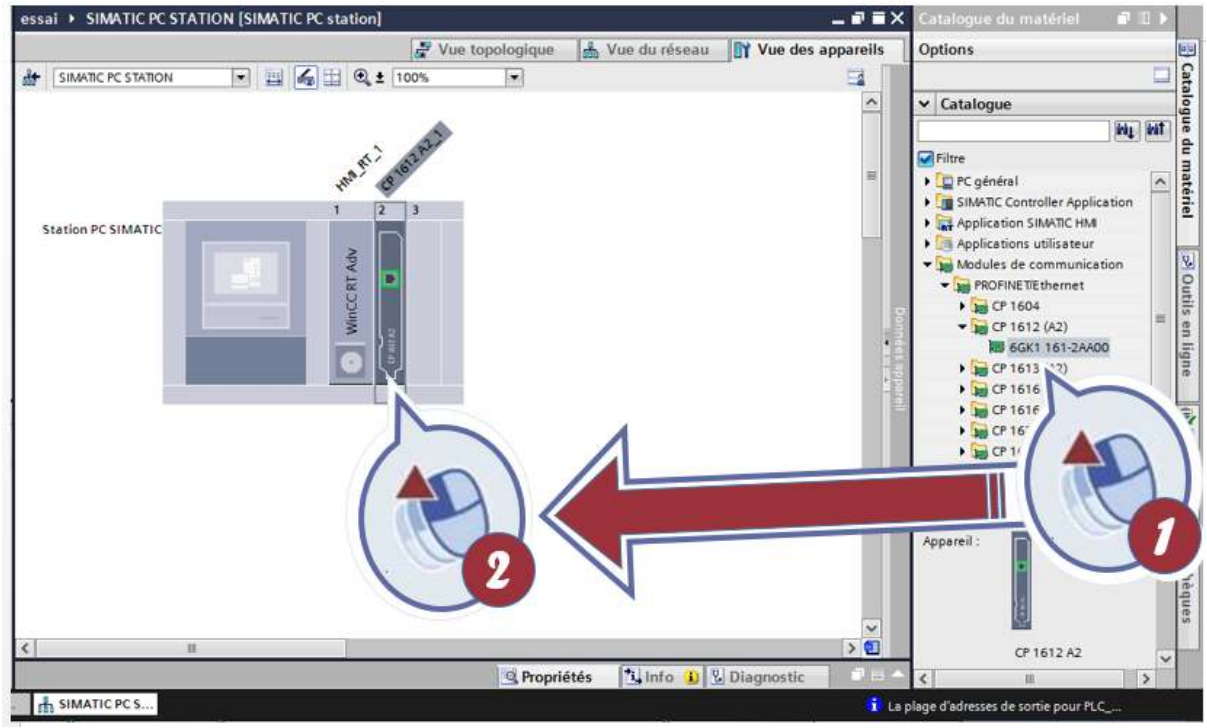


Figure 3.37 :insertion du module de communication.

Apres l'insertion du module de communication, on passe à " vue réseau" pour faire une liaison ente HMI et API, figure 3.398

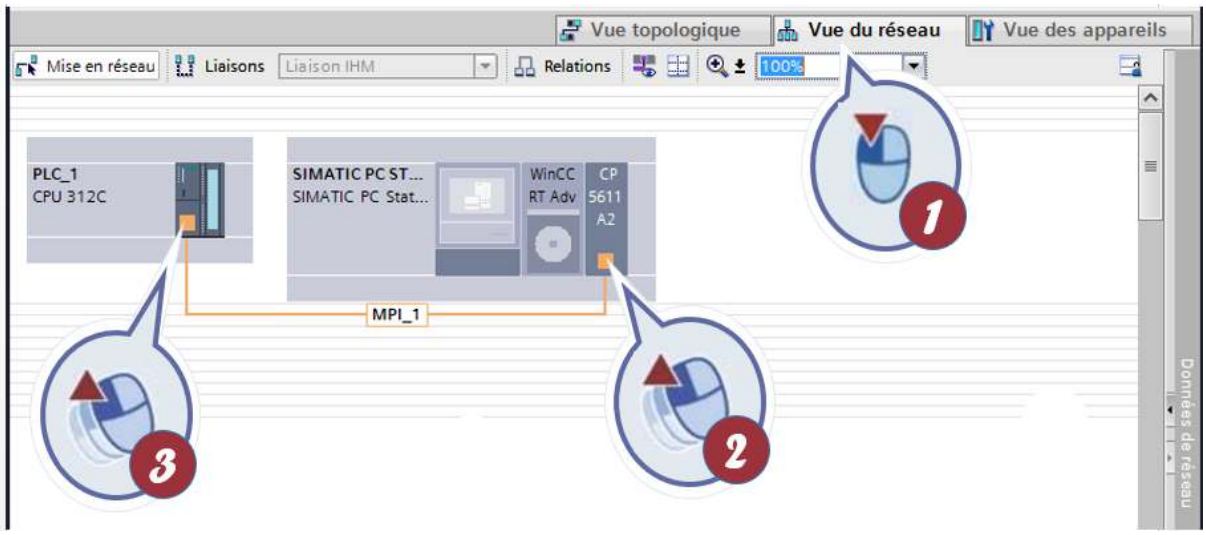


Figure 3.38 : la mise en réseau entre LAPI et WINCC RT Advanced.

3.7.2 Table de mnémoniques

Pour le wincc, on établir une la table de variables standard de l'HMI, après on saisit les variables qu'on aura besoin dans le programme de la supervision

a	Table de variables standard	Real	HMI_Liaison...	PLC_1	a	%MD88
ABSTETA	Table de variables standard	Real	HMI_Liaison_1	PLC_1	ABSTETA	%MD136
ACCELERATION	Table de variables standard	Real	HMI_Liaison_1	PLC_1	ACCELERATION	%MD162
b	Table de variables standard	Real	HMI_Liaison_1	PLC_1	b	%MD84
cap 1	Table de variables standard	Int	HMI_Liaison_1	PLC_1	"cap 1"	%MW200
cap 2	Table de variables standard	Int	HMI_Liaison_1	PLC_1	"cap 2"	%MW202
capteur1	Table de variables standard	Bool	HMI_Liaison_1	PLC_1	capteur1	%I125.1
capteur2	Table de variables standard	Bool	HMI_Liaison_1	PLC_1	capteur2	%I124.0
K1	Table de variables standard	Real	HMI_Liaison_1	PLC_1	K1	%MD141
K2	Table de variables standard	Real	HMI_Liaison_1	PLC_1	K2	%MD43
k3	Table de variables standard	Real	HMI_Liaison_1	PLC_1	k3	%MD89
march_droit	Table de variables standard	Bool	HMI_Liaison_1	PLC_1	march_droit	%Q124.2
march_gouch	Table de variables standard	Bool	HMI_Liaison_1	PLC_1	march_gouch	%Q124.3
sts_gate	Table de variables standard	Bool	HMI_Liaison_1	PLC_1	sts_gate	%M0.0
Tag_16	Table de variables standard	Real	HMI_Liaison_1	PLC_1	Tag_16	%MD35
TETA	Table de variables standard	Real	HMI_Liaison_1	PLC_1	TETA	%MD142
<Add new>						

Figure 3.39 : la table de variables standard de l'HMI.

3.7.3 Structure du programme HMI

Pour superviser le système il faut ajouter des vues. Figure 3.40.

Dans notre cas on a trois vue de supervision

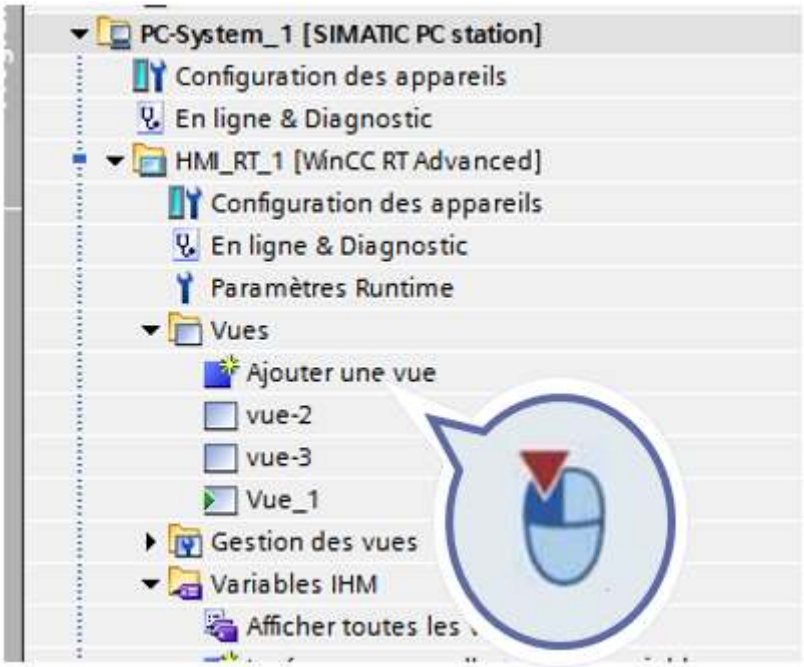


Figure 3.40 : ajouter une vue.

Dans un écran il y a plusieurs vues qui représente les interfaces de supervision, l'avantage de Runtime Advanced c'est qu'on peut paramétrées l'écran de supervision et on peut ajouter un nombre important des actions.figure3.41.

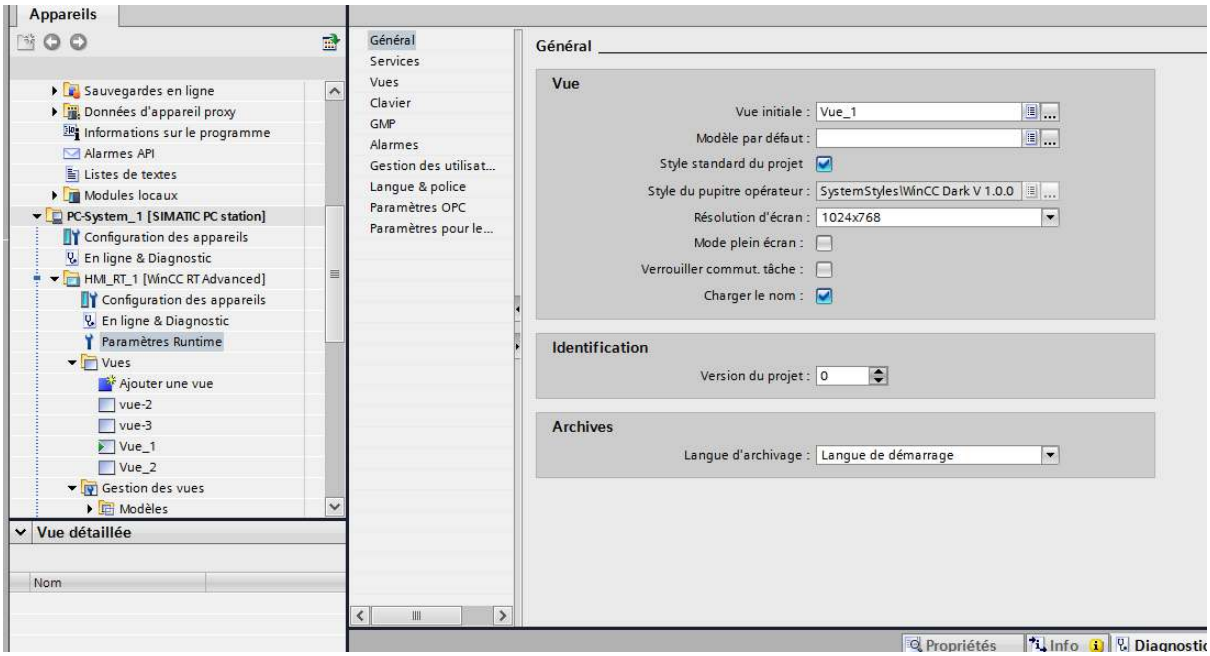


Figure 3.41 : paramètres Runtime.

L'interface de la vue-1 :

La première vue c'est pour saisie les valeurs de PID et voir la variation de l'angle, puisque on a dans le système PC, on a ajouté une petite fenêtre de camera surveillance pour voir la variation de l'angle en temps réel. Figure 3.42.

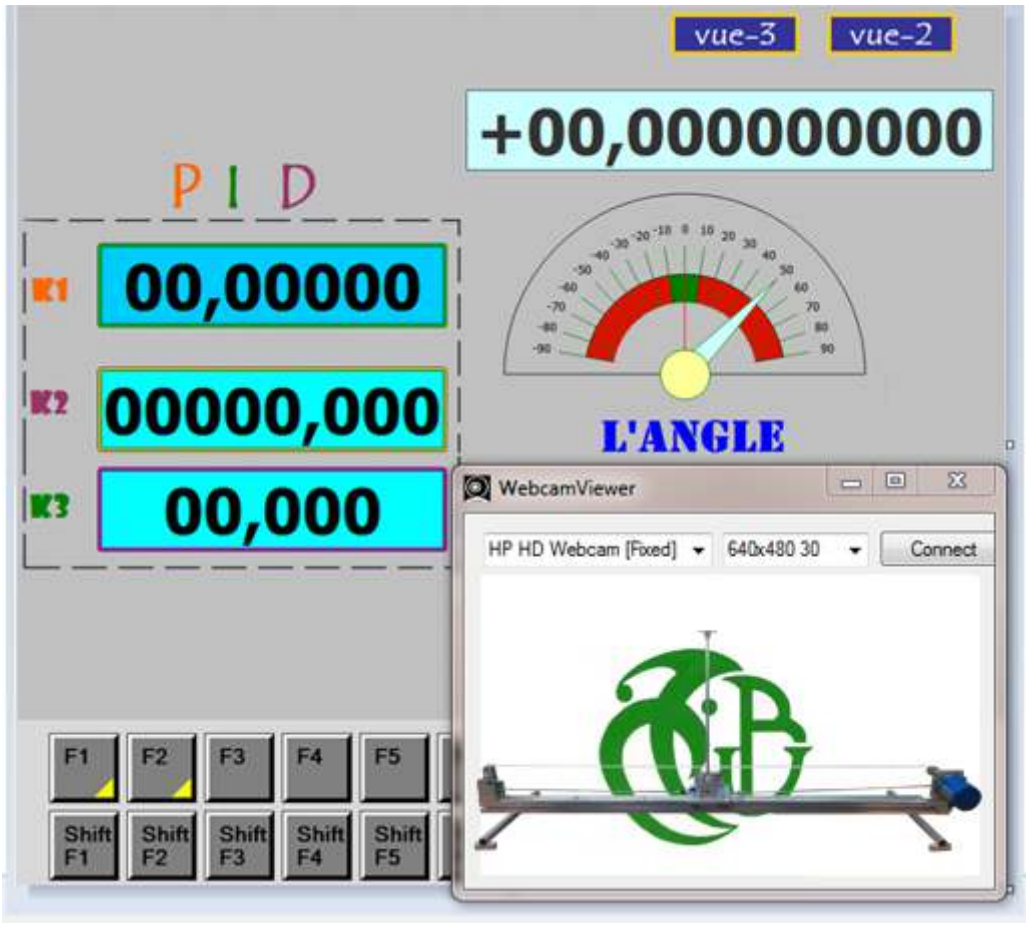


Figure 3.42 : vue-1.

On utilise les boutons 

pour deplacer de vue a une autre.On utilise aussi 

pour quiter la supervision.

L'interface de la vue-2 :

On peut visualiser les états momentanément des capteurs et le moteur dans cette vue

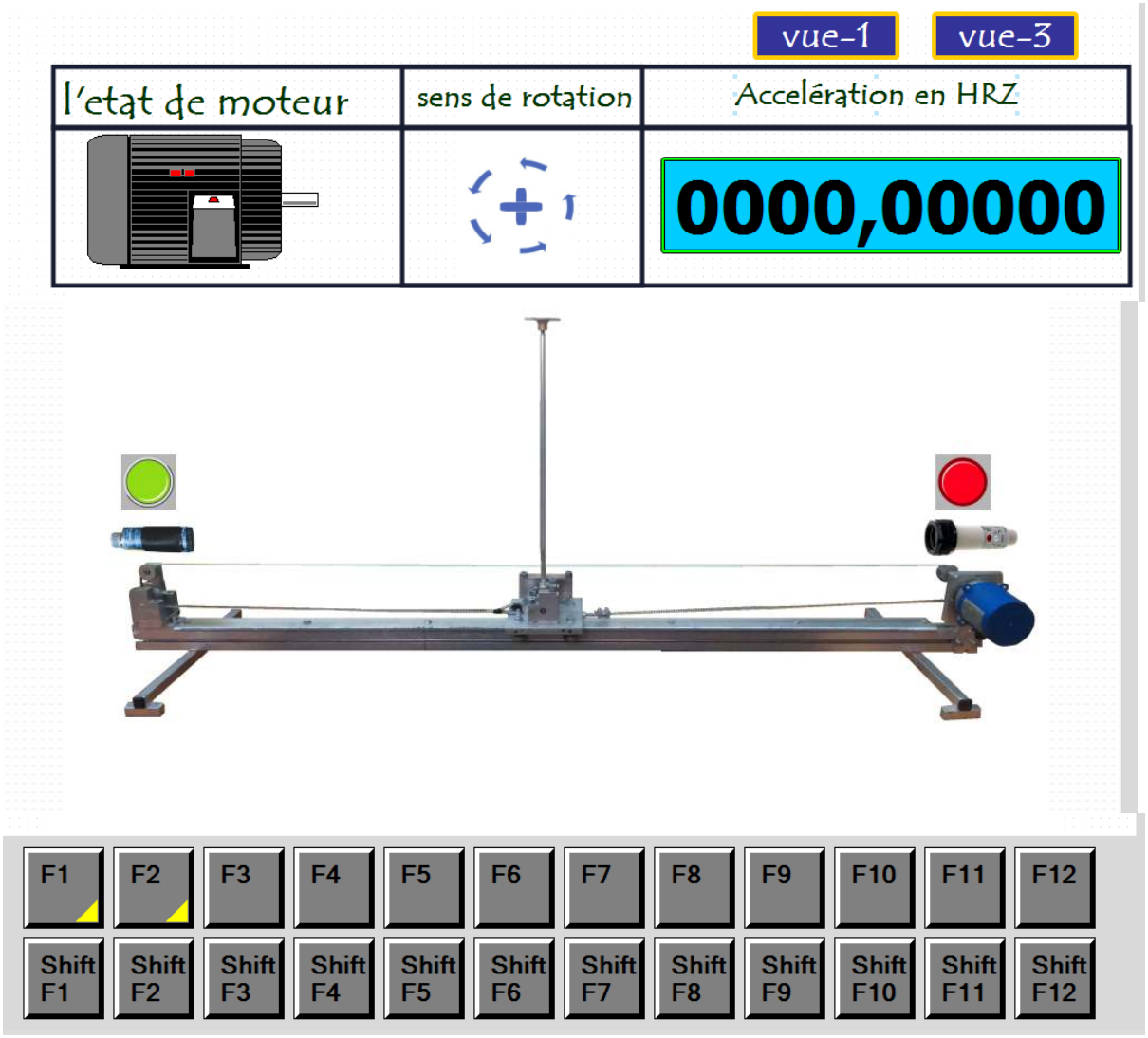


Figure 3.43 : vue-2.

Les capteurs en état du repos représenté par un feu vert , le feu rouge indique quil ya un objet devant le capteur.

Le clignotement du symbol du moteur ndique que le moteur est en etat de marche, sinon le moteur est en etat darret

Le symbole  signifie la rotation droite du moteur, et le symbole  signifie rotation gauche.

L'interface de la vue-2 :

Dans cette vue on voir le graphe de la variation de teta en fonction du temps et il y a aussi l'historique des alarmes

Figure 3.44 : vue-3.

Les alarmes :

Pour afficher les alarmes dans toutes les vues, on ajoute une vue globale et mettant la fenêtre des alarmes, on paramétrées cette fenêtre de façon que s'affiche lors de ladétection d'une alarme, et il s'interrompe jusque ou le problème se règle.

Figure3.45.

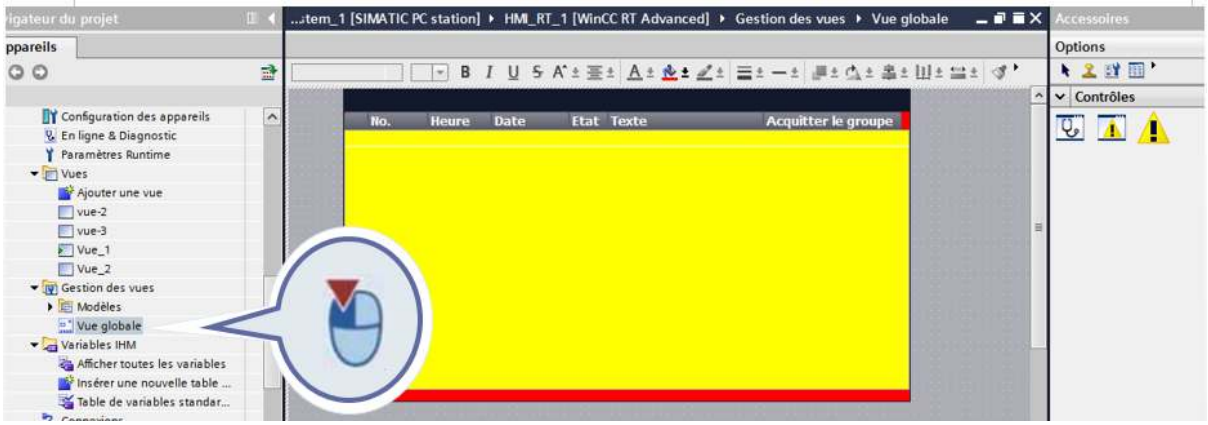


Figure 3.45 : vue global des alarmes.

Dans notre projet on a deux types des alarmes, alarmes de bit pour les variables booléens sont des capteurs (figure 3.46) et des alarmes analogiques se sont les extrémités de l'angle (figure 3.47).

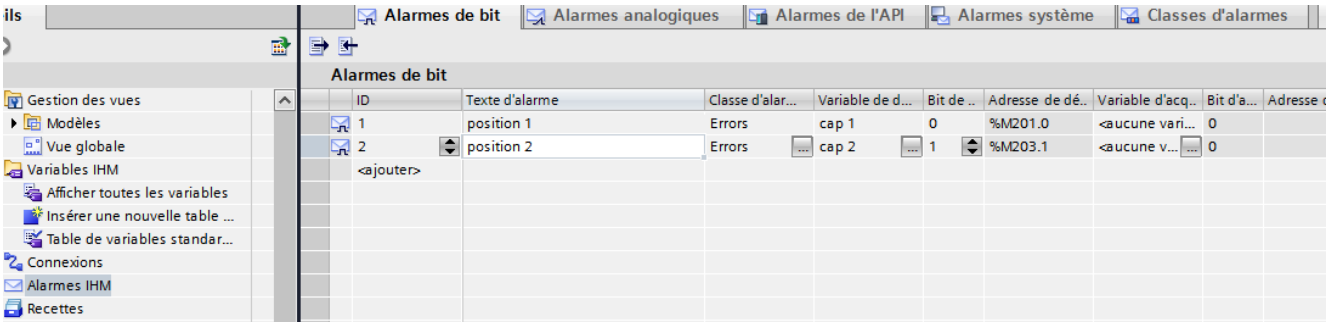


Figure 3.46 : Alarmes de bit.

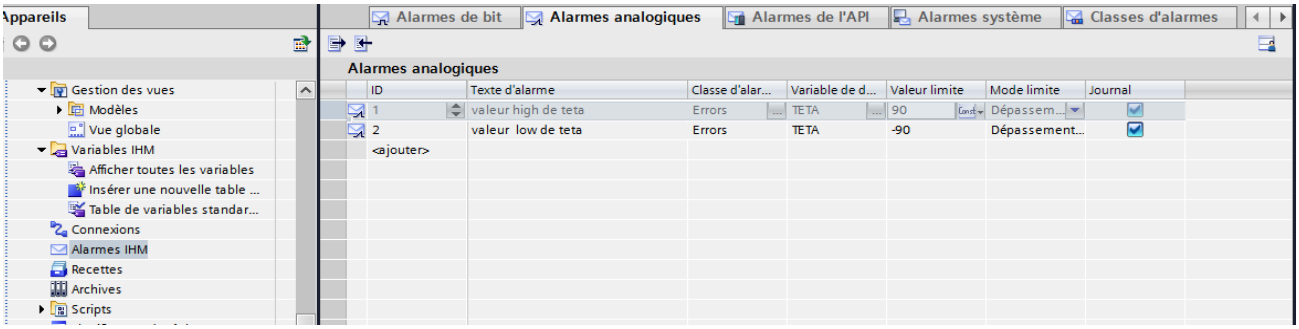


Figure 3.47 : Alarmes analogiques.

3.8 Conclusion

Le logiciel TIA PORTAL nous à faciliter La programmation de l'API à travers ses blocs qui ont plusieurs fonctions, il y a une possibilité de créer des nouveaux blocs pour des trucks spéciaux, dans notre programme on a utilisé que les blocs qui existent déjà en TIA PORTAL, par exemple UNSCALE et SCALE.

Il y a plusieurs avantages de Runtime Advanced parce qu'il est basé à PC, donc comme si on a plusieurs écrans d HMI standard dans une même station, la solution PC permet de faire des chausses qui n'existent pas dans l HMI , comme l'utilisation de caméra de surveillance dans notre projet.

Chapitre 4 RESULTATS EXPERIMENTAUX

4.1 Introduction

Dans ce chapitre on va voir les expériences qu'on a élaborées au fur et à mesure, selon les problèmes qu'on a rencontrés.

4.2 Expérience N°1

La CPU312c été le seul matériel disponible au labo donc on a fait les essais sans modules analogiques, donc on a mis en place un codeur incrémentale connecté a une voie a comptage rapide 10 KHZ, et la commande du variateur est faite par un bloc générateur d'impulsion appelé "PULSGEN.DB " qui utilise l'angle renvoyer par le codeur c'est à dire que la commande dépend de l'angle mesurée.

On a testé la sortie qui est bronché au variateur représentant le signal pwm de la commande par un multimètre on trouve toujours une variation de tension même si l'angle est fixe ce qui est un problème permanent.

4.3 Expérience N°2

Pour régler le problème précèdent on a intégré un module de sortie analogique, ce qui nous a donné un signal de commande stable et précis, mais quand même on a trouvé un problème au niveau du codeur qui ne peux pas mesurer avec précision en cas de petite variation répétitive.

4.4 Expérience N°3

Pour régler le problème du codeur on a choisi un potentiomètre parce qu'il utilise une plage de tension converti en angle dans notre cas donc les petites variations ne pose pas problème, on a ajouté un module d'entrée analogique pour faire la conversion tension/angle ainsi la lecture de l'angle.

4.5 Expérience N°4

Même avec ces améliorations on n'arrive pas à stabiliser le système parce qu'il y a plusieurs perturbation externes dont la tige et son point pivot donc on a effectué un changement du point pivot avec un nouveau qui est mieux centrer pour diminuer l'erreur de la mesure d'angle, aussi on a essayé plusieurs tiges pour joue sur le paramètre de la pesanteur.

4.6 Conclusion

Apres toutes ces expériences on est arrivé au point de contrôler la tige mais pas complètement, à cause des frottements au niveau du chariot et des poulies et aussi le type de moteur utilisée qui est lourd pour notre application.

Conclusion générale

Dans cette expérience, nous nous sommes intéressés à la stabilisation d'un pendule inversé à l'aide d'un API, ainsi que la supervision de ce système à travers un ordinateur qui remplace dans notre cas un écran programmable de supervision HMI.

Notre système est essentiellement composé d'un chariot mobile, un moteur asynchrone, une tige et deux capteurs. L'API joueici le rôle d'une unité de commande à l'aide du logiciel TIA PORTAL.

Au cours de la réalisation de cette expérience, on a été confronté à certain nombre de difficulté concernant le coté programmation, l'encodeur qu'on a utilisé ne représenté pas le bon choix parce que ce dernier n'adapté pas l'angle de la tige de façon optimale, on avait aussi face à nous des problèmes de nature mécanique comme par exemple, le problème que posent les frottements au niveau du chariot et les poulies dentés, le problème du choix concernant lemoteur. Si nous n'avons pas pu les résoudre tous, on a pu trouver une manière de régler ces difficultés, plus particulièrement en jouent sur les différents composants du système, pour notre part, l'expérience a été enrichissante puisqu' elle nous a fait découvrir de façon concrète la manière de résoudre certains problèmes qui nous on confronter lors des commandes du système.

Dans l'état actuel,notre table fonctionne de façon satisfaisante Nous n'avons pas pu réaliser l'expérience parfaitement avec la commande du chariot par le moteur asynchrone Nous pensons que cela est dû en particulier à l'insuffisance et l'incapacité du moteur dont nous disposons.

Ce problème se règlera par l'utilisation d'un moteur à courant continu ou pas a pas pour s'approcher du comportement stable du système.

Ce travail nous a montré que les commandes API peuvent aussi contribuer dans divers autre domaine que l'industrie, par exemple la robotique.

Le logiciel TIA PORTAL à démontrer son importance dans le cas des commandes des systèmes de supervision , ce dernier nous permet de convertir l'ordinateur en plusieurs écrans de supervision programmable de tels sorte qu'on peut bénéficier des avantages qui sont à la bases absentes dans les écrans HMI, par exemple l'utilisation de caméra de surveillance.

Bibliographie

[1],[4] Guide des automatismes, Copyright © 2002-2007 Thierry Schanen & POS Industry.

[2] D.GRIDAINE: sts Maintenance Industrielle, Codeurs rotatifs industriels.

[3] tayeb chrif, livreélectronique de base

[5] schneider-electric, Guide d'exploitation, Altivar 12Variateurs de vitesse pourmoteurs asynchrones, 2010.

[6]Wikipédia, Moteur asynchrone

[7]Auteur : alain.charbonnel@ac-caen.fr, Le moteur asynchrone triphasé, Mis à jour le 27 11 2010

[8] siemens manuel caractéristiques des modules,02 2013.

[9] siemens.html,Technique d'automatisation-siemens

[10] http://www.industry.siemens.com/topics/global/fr/tia-portal/pages/default.aspx.

[11],[12],[13],[14],[15],[16] l'aide du logiciel TIA PORTAL V13.

Remarque : on a développé la modélisation d'après le cours [lecture 26, Feedback Exemple : The inverted Pendulum] de professeur Alan V. Oppenheim.1978.