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

جـامعة سعد دحلب البليدة
Université SAAD DAHLAB de BLIDA

كلية التكنولوجيا
Faculté de Technologie

قسم الالكترونيك
Département d'Électronique

CDTA

Mémoire de Master

Filière Télécommunications

Spécialité Réseaux et

Télécommunications

Présenté par

ROUABHA ILHEM

&

MESBAH ROUMEISSA

Réalisation d'une interface LabVIEW de commande et de contrôle de température des huiles via le RS-485 Modbus RTU et un PC

Proposé par : Dr A. KEDADRA & Pr A. AISSAT

Année Universitaire 2020.2021

Remerciements

Nous tenons profondément à remercier Dieu de nous avoir éclairci les chemins du savoir et ceux de la vie, à nos familles respectives pour leurs soutiens morals et surtout financier.

Nous tenons à remercier notre encadreur le Maître de Recherche CDTA Mr. KEDADRA Abdelkrim pour son encadrement sans faille, son soutien moral, sa rigueur au travail, ses multiples conseils, ses orientations et sa disponibilité malgré ses multiples occupations.

Nos remerciements vont aussi à notre chère Professeur Mr. AISSAT Abdelkader qui nous a offert cette formidable expérience qui a enrichie nos connaissances et va marquer le début de notre carrière. Pour avoir accepté de diriger ce travail, Son soutien, ses compétences et sa clairvoyance nous ont été d'une aide inestimable.

Nous tenons à remercier sincèrement les membres du jury qui nous font le grand honneur pour d'évaluer ce travail et de nous accorder leur précieux temps.

Nous profitons aussi de ce mémoire pour exprimer nos plus vifs remerciements envers tous les enseignants qui nous ont apportés du soutien durant nos études et envers tous nos amis qui ont été toujours près de nous avec leurs encouragements, critiques et conseils.
Enfin, Nous voudrions associer à nos remerciements toutes les personnes qui ont contribué de près ou de loin à l'aboutissement de ce travail.

ملخص:

اي تغير غير مرغوب في درجة الحرارة يمكن ان يسبب بعض السلبيات التي لا ينبغي التغاضي عنها، ولهذا السبب فان درجة الحرارة عامل مهم للغاية في العملية الصناعية. من المهم ان تعرف قيمتها بدقة لتوفير الوقت وتحسين الانتاجية ما يستلزم التنظيم التلقائي هذا العمل مساهمة في مشروع CDTA للتحكم في درجة حرارة الزيوت الغذائية المستخدمة وتنظيمها من اجل اتخاذ قرار بشأن تدهورها. للقيان بذلك يتم تنفيذ العمل النظري على الصناعة التلقائية واعداد تجريبي يتضمن ذلك تطوير واجهة اكتساب والتحكم في منظمEMKO eco PID عبر
محول USB / RS-485 بواسطة جهار كمبيوتر تحت واجهة رسومية LabVIEW.

كلمات المفاتيح: التنظيم التلقائي؛ تنظيم درجة الحرارة؛ القيادة والاستحواذ والسيطرة.

Résumé :

Tout changement de température indésirable peut engendrer certains inconvénients qu'il ne faut pas négliger, car la température représente un paramètre très important dans le processus industriel. Sa valeur doit être connue avec précision pour gagner du temps et avoir une meilleure productivité, la régulation automatique s'impose. Ce travail consiste à contribuer dans un projet du CDTA pour commander et assurer la régulation de la température des huiles agroalimentaires utilisés afin de décider sur leur dégradation. Pour ce faire, un travail théorique sur l'automatisme industriel, et un montage expérimental sont réalisés. Il s'agit de développer une interface d'acquisition et de contrôle d'un régulateur EMKO eco PID via le convertisseur adaptateur USB/RS-485 par un PC sous l'interface graphique LabVIEW.

Mots clés : la régulation automatique ; régulation de température ; commande, acquisition et contrôle.

Abstract :

An unsolicited temperature change can cause some drawbacks that should not be overlooked. That's why the temperature is a very important parameter in an industrial process. Its value must be known precisely, to save time and have better productivity, automatic regulation is required. This work is a contribution to CDTA project to achieve the temperature control of used food oils in order to decide on their degradation. A theoretical work on industrial automation and an experimental set-up are carry out. This includes developing an acquisition interface and controlling an EMKO eco PID regulator via the USB / RS-485 adapter converter by a PC under the LabVIEW graphical interface.

Keywords : automatic regulation ; temperature regulation ; command, acquisition and control.

Listes des acronymes et abréviations :

ASCII : American Standard Code for Information Interchange.

CRC : Contrôle de Redondance Cyclique.

CDTA : Centre de Développement des Technologies Avancées.

CIM : Computer Integrated Manufecturing.

CAN : Controller Convertisseur Analogique Numérique.

DCS : Distributed Control System.

ICS : Industrial Control System.

IP : Internet Protocol.

LabVIEW : Laboratory Virtual Instrument Engineering Workbench.

PLC : Programmable Logic Controllers.

PID : Proportionnel Intégral Dérivé.

PV : Process Value.

PTC : Coefficient de Température Positif.

PLC : Contrôleur Logique Programmable.

PID : Proportionnel Intégral Dérivé.

RTU : Remote Terminal Unit (Unité Terminale Distante).

SCADA : Supervisoy Control And Data Acquisition.

SV : Set Point.

TTL : Transistor-Transistor Logic.

TOR : Tout Ou Rien.

TCP : Transmission Control Protocol.

VI : Virtual Instrument.

Liste des symboles :

C(t) : la commande générée du régulateur.

ε(t) : Erreur instantané.

Kp : le gain proportionnel.

Ki : le gain intégral.

Kd : le gain dérivé.

Ti : la constante du temps d'action intégrale (en seconds).

Td : La constante du temps d'action dérivée (en seconde).

Table des matières

Chapitre 02 : Etudes de la partie matérielle

Chapitre 03 : Etudes de la partie logicielle.

Chapitre 04 : Implémentation et résultats.

Liste des figures

Liste des tableaux

Introduction générale

Depuis des années, l'humanité connaît une évolution exponentielle qui ne fait que s'accroître avec l'essor des nouvelles technologies. Ces technologies sont significatives de progrès, permettant ainsi une évolution de la société vers un futur désirable qui a pour but d'améliorer le quotidien et les conditions de vie [1].

Parmi ces technologies on peut citer l'automatique qui est généralement définie comme la science qui traite des ensembles qui se suffissent à eux-mêmes et où l'intervention humaine est limitée à l'alimentation en énergie et en matière première.

L'objectif de l'automatique est de remplacer l'homme dans la plupart des tâches. Les systèmes automatiques peuvent opérer en boucle ouverte à partir d'un seul signal de commande, n'ayant aucune information sur la sortie la correction est impossible. C'est uniquement avec une boucle fermée (contre réaction) qu'on peut stabiliser, améliorer les performances et de compenser l'effet des perturbations. La commande de ces systèmes est faite par des régulateurs.

Un régulateur compare la consigne à une mesure qui est effectué par des capteurs puis élabore une commande qui sera transmise aux actionneurs (vannes, moteurs, etc.), afin de corriger les erreurs et conduire la sortie du système vers la consigne. A titre indicatif le régulateur PID (proportionnel intégral dérivé) est bien adapté à la plupart des processus industriel grâce à la simplicité de sa structure, le nombre restreint de paramètres à régler et il permet d'obtenir une régulation optimale si les paramètres sont bien choisis (Rapidité, précision, stabilité et robustesse) [2].

Dans la réalité industrielle, la complexité des systèmes, ainsi que celle des traitements à réaliser, nécessite souvent le recours à des outils numériques de traitement appelé calculateurs numériques. L'utilisation d'un calculateur à la place d'un correcteur analogique est remarquable, parmi les avantages de la commande numérique :

- Mise au point souple.
- Meilleurs résultats en termes de performances.
- Capacité de mémoire élevée.

Dans notre projet de fin d'étude de master est demandé de faire la commande et l'implémentation d'un processus de régulation de température. Notre travail épuise d'un projet de recherche de CDTA pour la caractérisation des huiles.

Notre réalisation concerne le développement d'une interface d'opérateur pour la commande et le contrôle (lecture et écriture) sous LabVIEW d'un régulateur de température de type EMCO Eco PID. La commande et le contrôle est assurer par une l'interface à développer sous LabVIEW ou on peut afficher la valeur de la consigne et établir la valeur de processus.

Notre projet est par conséquent organisé de la manière suivante :

- Le premier chapitre s'étalera sur des généralités concernant le système du contrôle et de régulation industriel en ciblant le régulateur PID.
- Le deuxième chapitre traitera le régulateur de température EMKO Eco PID. Nous allons présenter quelques protocoles de communications industriels, en particulier ceux utilisés pour établir la connexion entre le maitre et l'esclave. Nous traitons surtout le protocole Modbus RTU et la configuration physique du dispositif dans ses parties matérielles.
- Le troisième chapitre portera essentiellement sur le logiciel LabVIEW utilisé pour effectuer l'acquisition et le contrôle de température.
- Le dernier chapitre sera consacré à la mise en œuvre de notre système ainsi que les résultats obtenus et leurs interprétations.

On terminera avec une conclusion générale et les perspectives.

I. Chapitre 1 Contrôle et régulation industriel

1.1 Introduction :

L'évolution des technologies a engendré un accroissement du niveau d'automatisation et, en conséquence, une augmentation de la complexité des procédés et de la probabilité de défaillances. L'efficacité et la sûreté des systèmes industriels sont ainsi devenues indissociables et la présence des opérateurs humains reste encore indispensable pour prendre en charge la surveillance et le fonctionnement du procédé [3].

Afin d'atteindre une production performante et fiable des machines industrielles les systèmes de contrôle nous offrent une régulation bien précise d'un certain nombre de grandeur physique pour obtenir des valeurs désirées telle que la température dans notre cas.

1.2 Les systèmes de contrôle industriel :

Le système de contrôle industriel (ICS : industial control system) englobe plusieurs types des systèmes de contrôle, y compris :

- Les systèmes de contrôle de supervision et d'acquisition des données (SCADA : Supervisory Control And Data Acquisition) ;
- Les systèmes de contrôle distribués (DCS : Distributed Control Systems) ;
- D'autres configurations de système de contrôle telles que les contrôleurs logiques programmables (PLC : Programmable Logic Controllers) qu'on trouve souvent dans les secteurs industriels et les infrastructures critiques.

Un ICS regroupe un ensemble des combinaisons de composants de commande (électriques, mécaniques, hydrauliques) qui agissent pour réaliser une fonction objet dans l'industrie tell que la fabrication en série. La partie processus concerne principalement les résultats de la production automatisée. Le contrôle peut être

entièrement automatisé ou peut inclure intervention humaine dans la boucle de commande [4].

1.2.1 Hiérarchie des systèmes de contrôle :

La hiérarchie de système de contrôle comprend l'ensemble des dispositifs (hardware) et logiciels (software) organisés dans une arborescence hiérarchique.

Ils peuvent être hiérarchisés selon deux axes. La première, qui peut être considérée comme « verticale », s'appuie sur le concept de la pyramide CIM (Computer Integrated Manufacturing) et propose 4 niveaux, chacun assurant un niveau de décision et disposant d'une visibilité adaptée à sa place dans la hiérarchie **(Voir figure 1.1)** :

- Niveau 0 : ensemble des équipements de terrain (capteur, actionneur…) permettant l'action et la prise d'informations sur le procédé. C'est le niveau de l'instrumentation.
- Niveau 1 : ensemble des équipements réalisent les fonctions d'acquisition et les fonctions dites « réflexes », c'est-à-dire de traitements d'automatismes logiques et séquentiels, de régulation, de protection et de restitution des informations logiques et analogiques, en lien avec le niveau 01.
- Niveau 2 : ensemble des équipements permettent de réaliser les fonctions de conduite et de supervision de l'unité de production, en lien direct avec le niveau 1.
- Niveau 3 : ensemble des équipements qui apportent une aide à l'opérateur pour optimiser l'exploitation de son installation.

Figure (1.1) les 4 niveaux hiérarchiques d'un système de contrôle industriel [6].

Le second axe privilégie la hiérarchie des fonctions de contrôle-commande selon leur criticité vis-à-vis du processus [5].

1.3 Régulation et systèmes asservis :

Afin d'assurer le contrôle des grandeurs permettent le fonctionnement du processus, un certain nombre de moyens et une « intelligence » adaptée doivent être mis en œuvre soit par une régulation industrielle soit par un asservissement de système.

1.3.1 La régulation industrielle :

La régulation industrielle occupe une place importante dans le monde moderne, en raison des performances de plus en plus élevées. La régulation regroupe l'ensemble des techniques utilisées visent à contrôler une grandeur physique soumise à des perturbations. Cette grandeur est appelée "grandeur réglée".

La régulation permet de maintenir une grandeur physique à une valeur constante quelque soit les perturbations extérieures. L'objectif global de la régulation peut se résumer par ces trois mots clefs : Mesurer, Comparer et Corriger.

1.3.2 Les systèmes asservis :

L'asservissement impose à la grandeur de sortie (grandeur à régler) du système de suivre rapidement les variations de la consigne avec une certaine précision (la grandeur de sortie est identique ou proportionnelle à une grandeur d'entrée) Un système asservi est un système dit suiveur, c'est la consigne qui varie.

Avec la représentation par schéma bloc **(Voir figure 1.2)**, on voit qu'une commande en boucle fermée est composée de deux chaînes : La chaîne directe et la chaîne inverse.

- La chaîne directe se compose de comparateur, correcteur, l'amplificateur et de l'actionneur. Elle permet de corriger les effets d'une perturbation sur le système.
- La chaîne inverse (ou boucle de rétroaction) se compose du capteur et du transmetteur. Elle « surveille » en permanence l'état de la sortie pour informer le régulateur des modifications à apporter sur la chaîne directe [7].

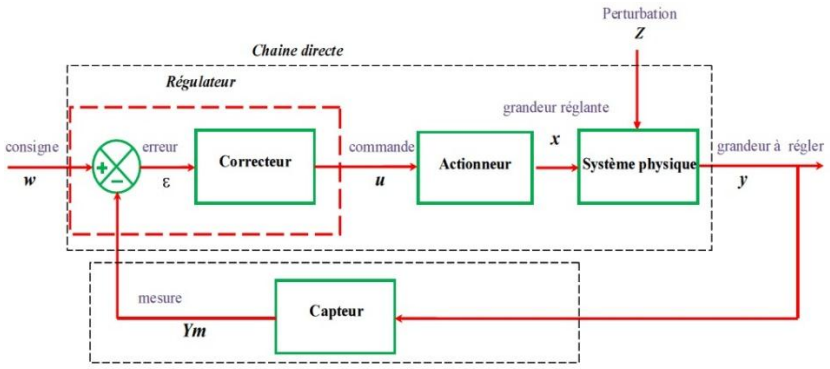


Figure (1.2) chaine direct (chaine d'action) et chaine inverse (chaine de réaction) d'un système asservis [8].

Il est bien de faire la distinction entre la boucle de régulation et la boucle d'asservissement. Les deux fonctionnent sur le même principe, mais leur finalité diffère sensiblement [9] :

- La régulation impose à la grandeur de sortie d'atteindre une valeur de consigne et d'y rester quelles que soient les perturbations éventuelles.
- l'asservissement consiste à maintenir une grandeur de sortie identique ou proportionnelle à une grandeur d'entrée [10].

1.4 Les différents types de régulation :

Le régulateur génère le signal de commande à partir de l'écart entre mesure et consigne. Le signal de mesure est l'image de la grandeur réglée, provenant d'un capteur et d'un transmetteur, et transmise sous forme d'un signal électrique ou pneumatique. La consigne peut être interne (fournie en local par l'opérateur) ou externe. L'affichage de la commande se fait en (%) et généralement en unités physique pour la consigne et la mesure.

Si un régulateur est en automatique, sa sortie dépend de la mesure et de la consigne. Ce n'est pas le cas s'il est en manuel [10].

Il existe différents types de régulation :

- Régulation en boucle ouvert
- -Régulation en boucle ferme
- -Régulation mixte
- -Régulation en cascade
- -Régulation split range
- -Régulation TOUT ou RIEN (TOR)
- -Régulation PID

1.4.1 La régulation en boucle ouvert :

On parle de régulation en boucle ouverte quand c'est l'opérateur qui contrôle l'organe de réglage. Une régulation en boucle ouverte ne peut être mise en œuvre que si l'on connaît la loi régissant le fonctionnement du processus. Un asservissement en boucle ouverte permet d'anticiper les phénomènes et d'obtenir des temps de réponse très courts. Parmi les inconvénients de l'asservissement, c'est qu'il n'y a aucun moyen de contrôler à plus forte raison, ou de compenser : les erreurs, les dérives, les accidents qui peuvent intervenir à l'intérieur de la boucle. En général, la régulation en boucle ouverte ne compense pas les facteurs perturbateurs. **(Voir figure 1.3)**

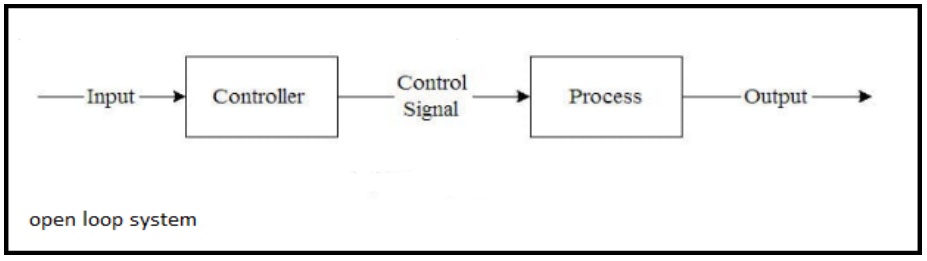


Figure (1.3) un système en boucle ouvert [10].

1.4.2 La régulation en boucle fermée :

La variable de sortie (grandeur réglant) de la chaîne de régulation, exerce une influence sur la valeur de la variable d'entrée (contrôlée), pour la maintenir dans des limites définies on utilise une régulation ou un asservissement en boucle fermée.

Le principe dans ce type de régulation est que l'action correctrice s'effectue après que les effets des grandeurs perturbatrices aient produit un écart entre la mesure et la consigne. Cet écart peut être également provoqué par un changement de consigne. Dans les deux cas, le rôle de la boucle fermée est d'annuler l'écart. **(Voir figure 1.4)**

Parmi les inconvénients d'une régulation en boucle fermée, le comportement dynamique de la boucle dépend des caractéristiques des différents composants de la boucle, et notamment du processus, un mauvais choix de certains composants peut amener la boucle à entrer en oscillations dit phénomène pompage [11].

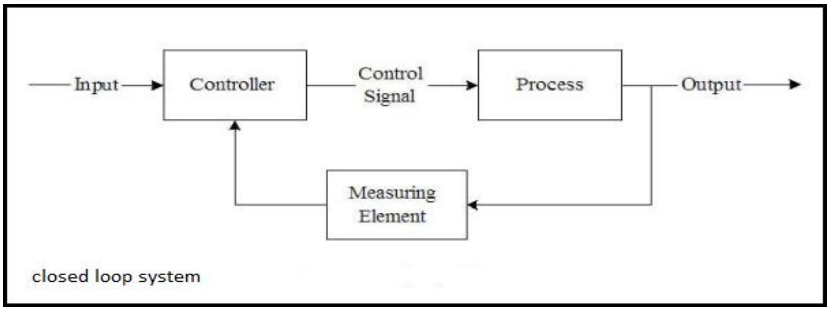


Figure (1.4) un système en boucle fermée [10].

1.4.3 La régulation mixte :

Ce type de régulation est l'association d'une régulation en boucle fermée et d'une régulation en boucle ouverte. Ce type de régulation est à mettre en œuvre lorsqu'une perturbation affecte directement la grandeur à régler.

1.4.4 La régulation en cascade :

Le but de la régulation en cascade est de prévoir une boucle interne rapide afin d'anticiper les perturbations, avant que celles-ci n'aient atteint la sortie de la boucle principale. Bien entendu, la régulation en cascade est inefficace si la perturbation survient en aval de la mesure intermédiaire.

1.4.5 La régulation split range :

La régulation split range est un montage particulier utilisant au minimum deux organes de réglage commandés par le même signal. On a également recours à ce type de régulation lorsqu'il est nécessaire d'utiliser deux grandeurs réglées ayant des effets opposés ou complémentaires sur le processus à contrôler.

1.4.6 La régulation TOUT ou RIEN (TOR) :

Le fonctionnement TOR se caractérise par deux états possibles pour la commande. Celui qui correspond à la commande maximale (100 %) et celui qui correspond à la commande minimale (0 %). Un seuil limite la fréquence de commutation du système.

1.4.7 La régulation PID :

Le régulateur standard le plus utilisé dans l'industrie est le régulateur PID (proportionnel Intégral dérivée), car il permet de régler à l'aide de ses trois paramètres les performances d'une régulation d'un processus [11].

Un régulateur PID remplit essentiellement trois fonctions :

- Il fournit un signal de commande
- Il élimine l'erreur statique ε(t) grâce au terme intégrateur.
- Il anticipe les variations de la sortie grâce au terme dérivateur.

1.5 Les Régulateurs PID :

Les systèmes peuvent présenter une précision insuffisante, une instabilité, un temps

de réponse trop lent, une sensibilité aux perturbations etc. alors il est nécessaire de corriger leurs comportements en insérant dans la boucle de régulation un dispositif visant à corriger les défauts des systèmes [9]. Les contrôleurs PID sont utilisés dans une large gamme d'applications pour le contrôle de processus industriel. Environ 95% des opérations en boucle fermée du secteur de l'automatisation industrielle utilisent des contrôleurs PID (PID : Proportional-Integral-Derivative). Ces trois contrôleurs sont combinés de manière à produire un signal de contrôle [12].

1.5.1 Le fonctionnement des Régulateurs PID :

Le contrôleur PID maintient la sortie de sorte qu'il n'y ait aucune erreur entre la variable de processus et le point de consigne / la sortie souhaitée lors d'opérations en boucle fermée. Le PID utilise trois comportements de contrôle P, I et D de base [12].

Il existe différentes possibilités de les associer qui sont expliqués Dans la suite.

a. L'action proportionnelle :

Le signale de la sortie du régulateur proportionnel est directement (ou inversement, suivant le sens d'action) proportionnel à l'écart entre la consigne C(t) et la mesure M(t), noté ε(t).

L'équation de régulateur proportionnel en fonction de temps est donnée par :

$$S(t) = k_p . \varepsilon(t) + u(t) = k_p . (C(t) - M(t)) + u(t) \qquad (1.1)$$

S(t) est la sortie du régulateur

k_p le gain du régulateur

C(t) est la consigne

M(t) est la mesure de la variable à régler

u(t), valeur du signal de sortie du régulateur lorsque l'écart **ε(t) = 0**, cette grandeur est accessible sur certains régulateurs, elle est appelée centrage de bande ou intégrale manuelle.

- **Bande proportionnelle (BP) :**

Le gain peut être défini par la spécification d'une autre variable, la bande proportionnelle.

$$BP(\%) = 100\% / K_p \qquad (1.2)$$

Elle est définie comme étant la variation, en pourcentage, de l'erreur à appliquer à l'entrée du régulateur pour que sa sortie varie de 100%.

Avec une régulation proportionnelle sur un procédé stable, la mesure ne rejoint pas la

consigne, Il subsiste toujours un écart résiduel **ε(t)**. Le rôle de l'action proportionnelle est de minimiser cet écart entre la consigne et la mesure, elle permet également d'accélérer le comportement global de la boucle fermée (la pente de la mesure augmente) en augmentant le gain. Mais il faut trouver un compromis entre la rapidité et la stabilité, car l'augmentation provoque des oscillations de la mesure.

b. L'action intégral :

En général, le régulateur ne fonctionne pas en action intégrale pure (trop instable), car son effet ne devient sensible que lorsque l'erreur dure depuis un certain temps. Pour obtenir une réponse initiale plus rapide, on l'utilise avec un régulateur proportionnel.

$$S(t) = ki. \int \varepsilon(\tau).\, d\tau = 1/Ti. \int \varepsilon(\tau).\, d\tau = 1/Ti. \int (\, M(t) - C(t)\,).\, d\tau \qquad (1.3)$$

c. L'action dérivée :

Elle est une action qui tient compte de la vitesse de la variation de l'écart entre la consigne et la mesure. Contrairement à l'action intégrale, l'action dérivée joue un rôle stabilisateur.

En effet, elle délivre une sortie variant proportionnellement à la dérivée de l'écart **ε(t)**.

$$\mathbf{S(t)} = kd.\,(d\,\varepsilon(t)/dt) = Td.\,(d\,\varepsilon(t)\,/\,dt) = Td\,.\,d(\,M(t) - C(t)\,)/dt \qquad (1.4)$$

Les figures ci-dessus **(Voir figure 1.5,6)** résumes les différents types de régulateur PID et ces différentes structures :

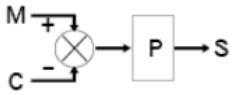
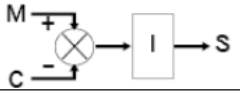
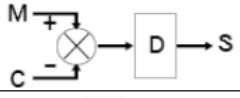
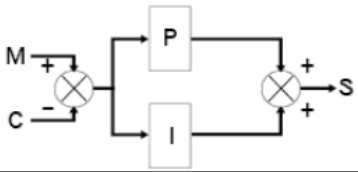
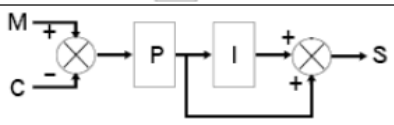
Type	Schéma	Signal de commande
P		$S_P(t) = K\varepsilon = K(M - C)$
I		$S_i(t) = \frac{1}{T_i}\int \varepsilon\, dt = \frac{1}{T_i}\int (M - C)\, dt$
D		$S_d(t) = T_d \frac{d\varepsilon}{dt} = T_d \frac{d(M - C)}{dt}$
PI parallèle		$S(t) = K\varepsilon + \frac{1}{T_i}\int \varepsilon\, dt$
PI série		$S(t) = K\varepsilon + \frac{K}{T_i}\int \varepsilon\, dt$

Figure (1.5) les différents types de régulateur PID [10].

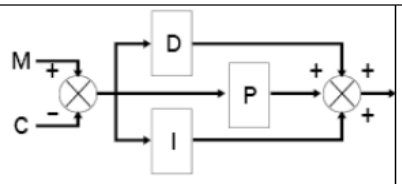
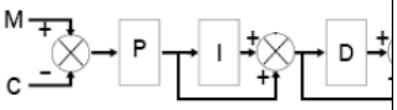
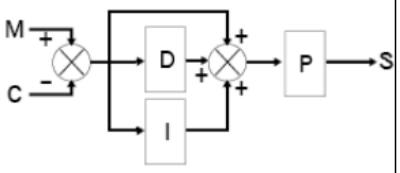
PID parallèle		$S(t) = K\varepsilon + T_d\frac{d\varepsilon}{dt} + \frac{1}{T_i}\int \varepsilon\, dt$
PID série		$S(t) = K(\frac{T_i + T_d}{T_i})\varepsilon + KT_d\frac{d\varepsilon}{dt} + \frac{K}{T_i}\int \varepsilon\, dt$
PID mixte		$S(t) = K\varepsilon + KT_d\frac{d\varepsilon}{dt} + \frac{K}{T_i}\int \varepsilon\, dt$

Figure (1.6) les différentes structures d'un régulateur PID [10].

1.5.2 Méthodes de réglage des actions :

Avant de commencer les réglages d'une boucle de régulation, il faut s'assurer que le sens d'action du régulateur est correct.

Il existe différentes méthodes de réglage des actions d'un régulateur PID suivant le type de procédé et les contraintes de fabrication on choisira l'une des méthodes.

a. Méthode par approches successives :

Elle consiste à modifier les actions du régulateur et à observer les effets sur la mesure enregistrée, jusqu'à obtenir la réponse optimale. On règle l'action proportionnelle, puis l'action dérivée et l'intégrale. Cette technique présente l'intérêt d'être simple et utilisable sur n'importe quel type de système. Néanmoins du fait de son caractère itératif, son application devient longue sur des procédés à grande inertie.

b. Méthode de l'identification du procédé :

Si l'on connaît les paramètres du procédé, suite à une modélisation de sa fonction de transfert réglant, et si l'on est en possession de la structure du régulateur. Il est alors possible de calculer rapidement les paramètres de réglage qu'on pourra affiner suite à des essais, afin d'obtenir la réponse souhaitée. Cette méthode nécessite un enregistreur à déroulement rapide. Elle est de préférence utilisée sur des procédés à grande inertie. [13]

c. Méthode de Ziegler et Nichols :

Zeigler-Nichols a proposé des méthodes en boucle fermée pour le réglage du contrôleur PID. Celles-ci sont, une méthode de cycle continu et une méthode

d’oscillation amortie. Les procédures pour les deux méthodes sont identiques mais le comportement en oscillation est différent. Pour cela, nous devons d’abord définir la constante du contrôleur p, Kp, sur une valeur particulière, tandis que les valeurs de Ki et Kd sont égales à zéro.

Le gain proportionnel est augmenté jusqu'à ce que le système oscille à une amplitude constante. Le gain pour lequel le système produit des oscillations constantes est appelé gain ultime (Ku) et la période d'oscillations est appelée période ultime (Pc). Une fois atteint, nous pouvons entrer les valeurs de P, I et D dans le contrôleur PID grâce au tableau de Zeigler-Nichols qui dépend du contrôleur utilisé, tel que P, PI ou PID [12].

1.6 Conclusion :

Dans la production industrielle, le contrôle électrique adopte généralement des nouvelles technologies développées pour améliorer l'ensemble du système industriel. Qu'il soit basé sur une structure hiérarchique ou générique, le système de contrôle industriel s'applique à toutes sortes d'industries, il offre des hautes performances, de la sécurité et de la fiabilité et de la Gestion électronique efficace des tâches.

Dans ce chapitre on a exploré les différents régulateurs industriels utilisées dans les system de contrôle. On a surtout insisté sur les régulateurs par PID qui reste, sous différentes versions, à la base de la majorité des équipements de contrôle-commande de l’industrie. A la suite nous allons concentre sur les parties matérielles et les parties logicielle du notre projet.

Chapitre 2 Etudes de la partie matérielle

2.1 Introduction :

Avant d'entamer un projet de réalisation pratique, une recherche et une analyse sont nécessaires concernant le matériel requis. On doit d'abord commencer par identifier les dispositifs dont on a besoin pour réaliser le projet qui assure le fonctionnement (la tache) souhaité.

Dans notre projet de fin d'étude de master est demandé de faire la commande et l'implémentation d'un processus de régulation de température objet d'un projet de recherche de CDTA pour la caractérisation des huiles.

L'objectif de ce chapitre est d'expliquer le réglage manuel de la valeur de consigne concernant le régulateur. Ensuite nous allons présenter quelques protocoles de communications industriels et en particulier ceux utilisés pour établir la connexion entre le maitre et l'esclave. Nous abordons surtouts la configuration physique du dispositif dans ses parties hardware et software.

2.2 Compositions matérielles :

Le prototype expérimental développé au CDTA est constitué de (**voir figure 2.1**) **:**

- Deux bacs dotés de transducteurs ultrasonores (quelque MHz).
- Sondes de température.
- Résistances chauffantes.
- Système de régulation de température à base du régulateur EMKO eco PID.

Ce dispositif permet la caractérisation de la dégradation des huiles usées.

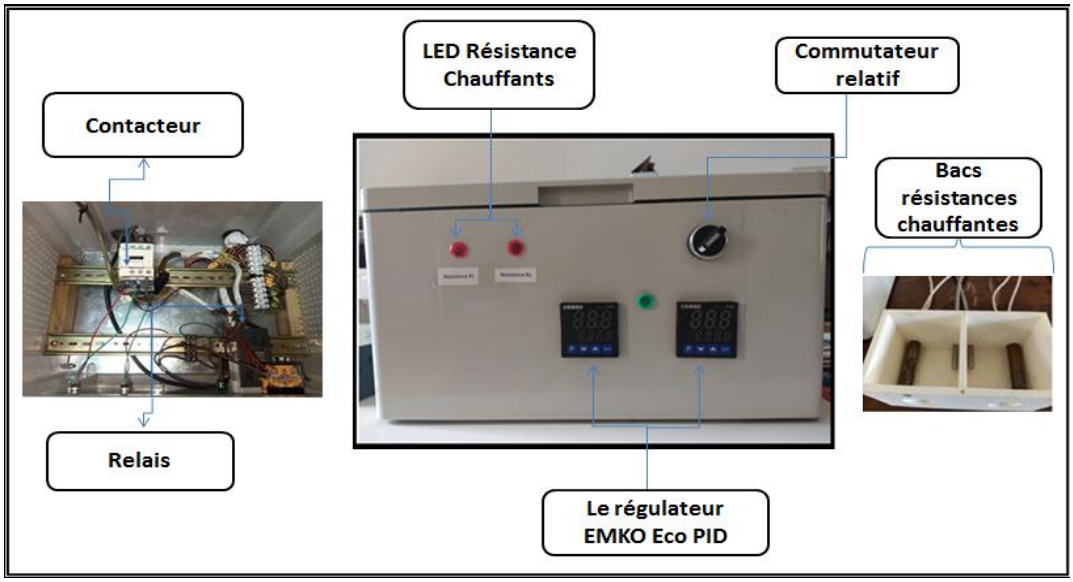


Figure (2.1) Dispositif expérimental du CDTA.

Ce dispositif se constitue de :

- ✓ Trois LED (deux pour les résistances de chauffage R1 et R2 et l'autre indique la mise en marche du dispositif ON/OFF).
- ✓ Un commutateur rotatif pour le mode MARCHE/ARRET.
- ✓ Deux régulateurs de température à deux étages de type EMKO eco PID. Ce dernier utilise une sonde de type PTC Pour prélevé la température d'huile.
- ✓ Deux capteurs de température qui transforme la grandeur mesurée (en dégrée Celsius ou en Fahrenheit) en un signal électrique qui sera traité et affiché sur le premier afficheur.
- ✓ Deux résistances chauffantes qui servent à augmenter la température de l'huile dans chaque bac lors du refroidissement.

Afin de garder la température désirée Le contrôle se fait grâce à un relais (Switch Interrupteur output1) permettant d'activer un contacteur électrique de coupure de phase. Le dispositif permet d'être contrôlé par un PC à travers une liaison série intégrée au régulateur de type RS-485 Modbus RTU.

2.2.1 Les Régulateurs EMKO :

Les régulateurs de température EMKO sont conçus pour mesurer et régler les valeurs d'un processus. Ils peuvent être utilisés dans de nombreuses applications, grâce à leurs fonctions basées sur les entrées de mesure TC et RTD, leurs sorties de contrôle multifonction, leurs fonctions d'alarme sélectionnables. Il inclut un contrôle avancé

par : les fonctions ON-OFF, P, PI, PD, PID et réglage adaptatif PID [14].

2.2.2 Le Régulateur EMKO eco PID :

Le régulateur EMKO de la série eco PID **(Voir figure 2.2)** contient un contrôle de température PID sensible à haute résolution, des entrées TC (types d'entrée J, K, R, S, T) et RTD sélectionnables par paramètre, faible consommation d'énergie, économie d'énergie et respect de l'environnement avec 2VA. De plus il offre un sauvegarde et récupération des paramètres utilisateur, un retour aux paramètres d'usine, une option de communication RS-485 Modbus (RTU), un affichage de température Processus (PV) à 3 chiffres et la consigne Set (SV) à 4 chiffres.

Figure (2.2) Le régulateur de température EMKO eco PID.

De plus, il a des entrées processus (TC, RTD), programmable ON-OFF, contrôle PID, et avec une adaptation des coefficients PID pour le système avec fonctionnement d'autoréglage (Step Response Tuning), une fonction de chauffage et de refroidissement sélectionnable, une sélection du type de fonctionnement avec hystérésis, un réglage du temps de réaction minimum pour les sorties de contrôle, une protection par mot de passe pour le mode de programmation **(Voir figure 2.3)** [14].

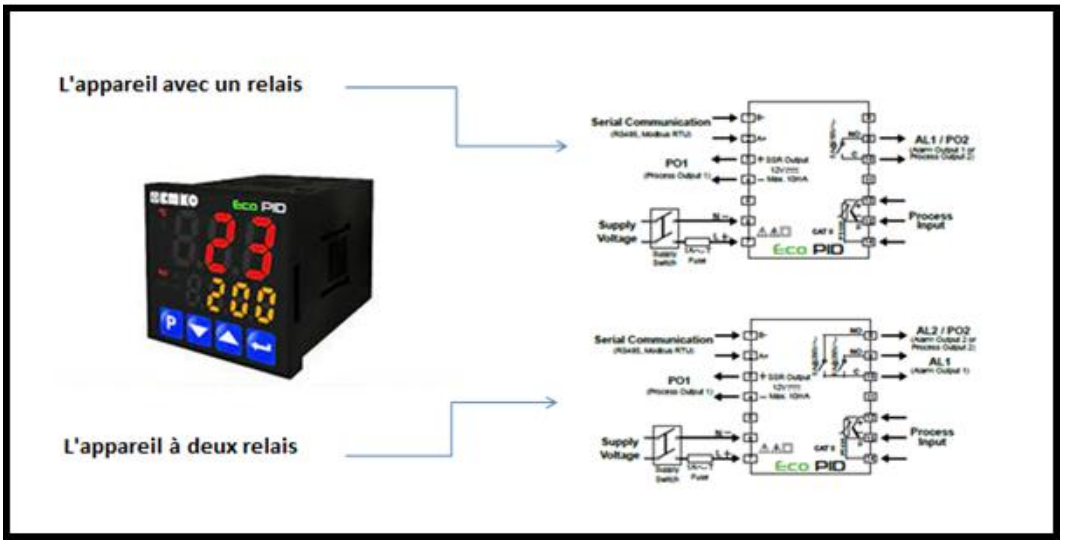


Figure (2.3) Schéma électrique du régulateur de température EMKO eco PID.

a. Réglage manuel du régulateur EMKO eco PID :

Le régulateur EMKO eco PID contient 4 touches différentes permettant de défiler ses différents paramètres. La **figure (2.4)** expliquera la fonctionnalité de chaque touche :

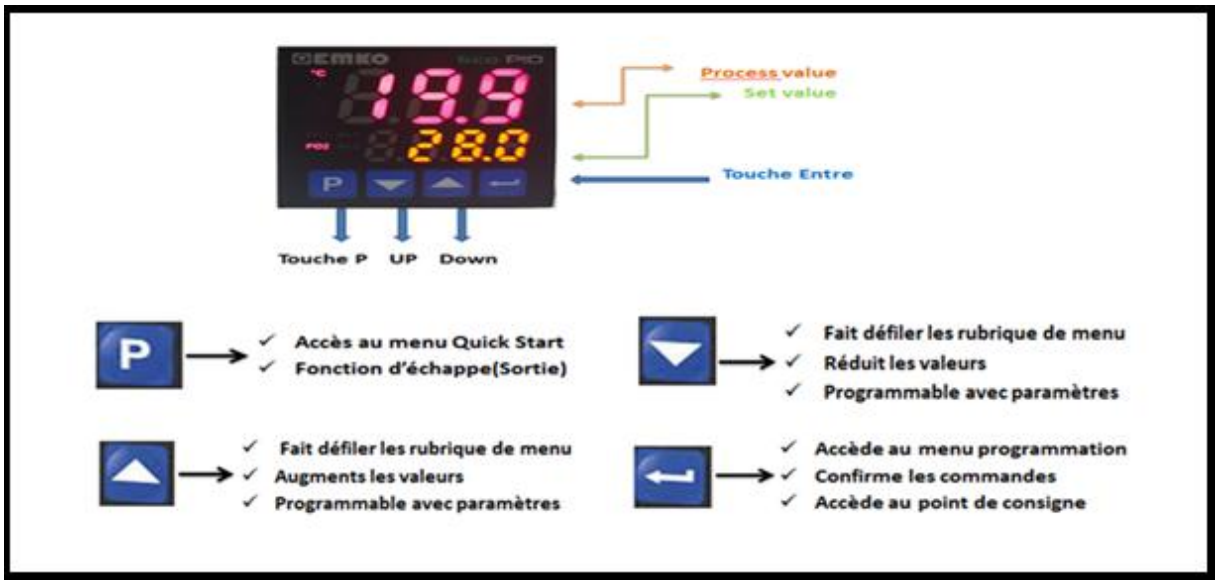


Figure (2.4) Touches composants l'écran du régulateur EMKO eco PID.

- **Procédure de réglage :**

1. Accédez à la section Programmation
2. Sélectionnez, paramètre dans le menu. Appuyez sur le bouton ASET/OK pour enregistrer le paramètre et revenir à l'écran principal Opération.
3. Le paramètre TUN clignote dans l'écran SET.
   - **Réglage du chauffage :** La valeur du Processus (PV) doit être inférieure à la valeur de la consigne (SP), au moins 5% de la pleine échelle.
   - **Réglage du refroidissement :** La valeur du Processus doit être supérieure à la valeur de la consigne, au moins 5% de l'échelle totale. Si cette condition n'est pas satisfaite, clignote pendant 10 secondes.
4. Le réglage Adaptatif est annulé dans les conditions suivantes :
   - ✓ Si le capteur se déclenche
   - ✓ Si le réglage Adaptatif n'est pas terminé en huit heures
   - ✓ Si la valeur Processus est supérieure à la valeur de consigne pendant le réglage Adaptatif du chauffage

- ✓ Si la valeur Processus est inférieure à la valeur de consigne pendant le réglage Adaptatif du refroidissement
- ✓ Si l'utilisateur modifie la valeur de consigne pendant le réglage Adaptatif Le réglage Adaptatif est alors annulé et le dispositif utilise les paramètres PID précédents sans les modifier [14].

- **Réglage manuel du point de consigne :**

Le régulateur EMKO eco PID fonctionne selon une seule valeur de consigne set1 à programmer. Pour se faire, nous devons suivre les étapes suivantes :

1. Appuyer et relâcher sur la touche « PSET » correspondante à la page initiale de l'écran.
2. L'afficheur de la valeur mesurée visualise l'étiquette SET1, tandis que l'afficheur de la consigne montre la valeur actuelle du point de consigne.
3. Les touches « up » et « down » permettent de modifier la valeur de consigne affichée sur l'écran SV.
4. Appuyez sur le bouton ASET/OK pour enregistrer la nouvelle valeur de la consigne et revenir à l'écran principal, Appuyez sur le bouton ASET/OK.
5. Appuyez sur le bouton Haut ou Bas pour modifier la valeur de consigne Alarme 1. Appuyez sur le bouton ASET/OK pour enregistrer la nouvelle valeur de consigne Alarme et revenir à l'écran principal.

Pour contrôler et commander le dispositif, il est nécessaire d'utiliser les bus de transmission et les protocoles de communication. Ces derniers sont détaillés dans les paragraphes qui suivent.

2.3 Modes de transmission :

Selon le sens d'échanges, on distingue 3 modes de transmission [15] :

- **Mode simplex ou unidirectionnel** : il caractérise une liaison dans laquelle les données circulent dans un seul sens, c'est-à-dire de l'émetteur vers le récepteur.
- **Mode half duplex ou bidirectionnel alterné** : les données circulent dans un sens ou dans l'autre mais pas les deux en même temps.
- **Mode full duplex ou duplex intégral** : dans ce cas Chaque extrémité de la ligne peut émettre et recevoir en même temps.

2.3.1 Techniques de transmission :

Afin de transférer des données entre un dispositif et un pc, il y'a deux méthodes, à savoir la transmission série et la transmission parallèle**.**

- **Transmission parallèle :** elle est caractérisée par un transfert simultané de tous les bits d'un même mot, elle utilisé pour de courtes distances.
- **Transmission en série** : Cette transmission permet de transmettre les données sur un seul support physique. La transmission se fait en émettant les bits des données l'une après l'autre de manière synchrone ou asynchrone.

  Les informations peuvent être transmises de façon irrégulière. Cependant l'intervalle de Temps entre 2 bits est fixe. Des bits de synchronisation (START, STOP) encadrent les informations des données [15].

2.4 Le port série :

Un port série est une interface de communication sur laquelle peut être mise en place une communication série. Sur les ordinateurs compatibles IBM, ils sont généralement appelés ports COM. Ils permettent de connecter à un ordinateur des périphériques externes transmettant des données séries et d'établir une communication bidirectionnelle entre l'ordinateur et le périphérique [16].

2.5 La liaison RS-485 :

L'interface série la plus utilisée dans l'industrie est le RS-485 ou le protocole EIA-485, qui présente un avantage majeur par rapport aux interfaces RS-232. Grâce à sa technologie multipoints, plusieurs émetteurs et récepteurs peuvent être connectés simultanément. La transmission des données s'effectue à l'aide des signaux différentiels pour une meilleure fiabilité [16].

- **Deux types de communications RS-485 sont possibles :**

✓ Les interfaces RS-485 à 2 contacts fonctionnent en mode half-duplex, uniquement capables d'envoyer ou de recevoir des données.

✓ Les interfaces RS-485 à 4 contacts, qui peuvent être utilisées en mode full-duplex pour permettre l'envoi et la réception simultanés des données [17].

2.5.1 Adaptation de la liaison RS-485 avec un PC :

En général les PCs grand public ne comportent pas une liaison directe RS-485, pour établir la connexion RS-485 avec le port COM. Il est alors recommandé d'utiliser le Port USB, généralisé actuellement sur tous les ordinateurs, donc il faut utiliser un

convertisseur adaptateur USB/RS-485 [18].

2.5.2 Convertisseur adaptateur USB/RS-485 :

Le module d'interface adaptateur USB/RS-485 est un modèle simple et pratique qui peut convertir différents formats de port série, son rôle est d'établir la connexion aux systèmes de télégestion télévise ou Modbus. Parmi ces caractéristiques on citera :

- Adaptateur de série compact : Il permet de convertir différents formats de port série. Modèle simple et pratique, idéal pour les intégrateurs de systèmes, l'électronique, etc.
- Convertisseur USB vers TTL RS-485.
- Connecteur USB : Type B-femelle.
- Connecteur série RS-485 : bornier à 3 broches.
- Taille : 77 x 19 x 18 mm [16].

Le convertisseur série USB vers RS-485 **(Voir figure 2.5)** utilise un port COM série virtuel pour convertir les signaux de communication série bidirectionnelle entre RS-485 et le port USB d'un ordinateur personnel. Il est autoalimenté via USB, utilise FT232RL et peut contenir jusqu'à 32 périphériques sur le bus.

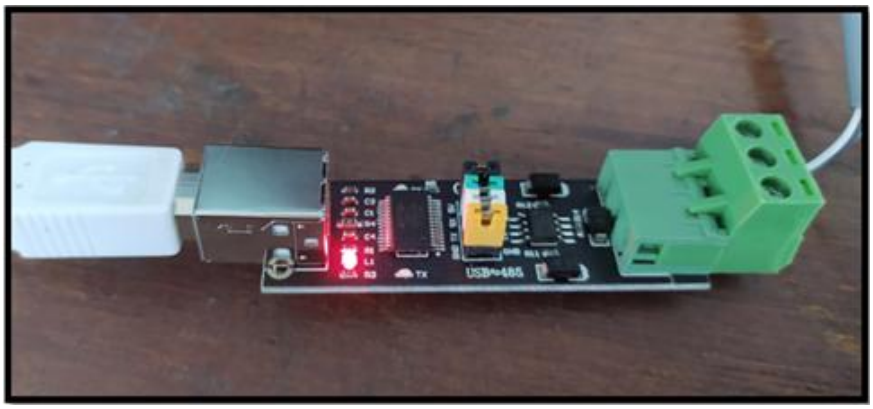


Figure (2.5) USB/RS-485 adaptateur.

2.6 Protocoles de communications :

Les besoins en communication sont très diversifiés selon les équipements connectés et les applications qu'ils supportent ce qui explique la variété des réseaux locaux industriels. Le trafic au niveau des réseaux de terrain n'est pas le même que celui des réseaux d'atelier ou d'usine. Les besoins varient selon la taille des données à transmettre et les contraintes de temps associées. Les flux d'échanges des données provenant du bas des architectures ne cessent de croître. Ceci nécessite d'augmenter les capacités de traitement des composants d'automatisme.

Dans le monde industriel, on rencontre plusieurs protocoles de communication par

lesquels on peut citer [20] :

- ✓ Le modbus ASCII
- ✓ Le modbus RTU
- ✓ Le Modbus TCP/IP
- ✓ Le CAN
- ✓ EtherNet/IP.

Parmi les protocoles suscités, notre étude concerne l'utilisation du protocole Modbus car le régulateur EMKO intègre le RS-485 Modbus pour communiquer.

2.6.1 Protocole Modbus :

Modbus est une norme de protocole industriel qui a été créé par < Modicon, maintenant Schneider Electric>, à la fin des années 1970 pour la communication entre les automates programmables (PLC).

Le protocole Modbus est défini comme un protocole maître / esclave, ce qui signifie qu'un appareil fonctionnant en tant que maître fera contrôler un ou plusieurs appareils fonctionnant en tant qu'esclave.

Le maître écrira des données dans les registres d'un appareil esclave et lire des données à partir des registres d'un appareil esclave. Une adresse de registre ou une référence de registre se trouve toujours dans le contexte des registres de l'esclave[21].

2.6.2 Principe de fonctionnement du Modbus :

Dans le protocole MODBUS **(Voir figure 2.7)**, c'est le maître entame la communication par une requête pour demander une action à accomplir par l'esclave ou par tous les esclaves. Il n'y a que le maître seul qui a le droit de lancer la communication avec tous les esclaves. Les esclaves qui sont toujours à l'écoute interceptent la totalité des échanges de messages sur le bus à qu'ils sont connectés. L'esclave est reconnu par une adresse propre et distinctive (un esclave n'a pas le droit d'avoir une même adresse qu'un autre déjà connecté au BUS). Celui-ci répondra seulement si le message lui est destiné. Le maître envoie une demande et attend une réponse ; Deux esclaves ne peuvent dialoguer ensemble [15].

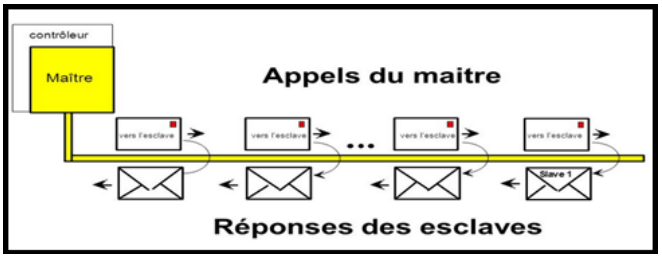


Figure (2.7) Principe de fonctionnement du protocole Modbus.

2.6.3 Types de données Modbus :

Il existe deux types de données Modbus, on trouve les bobines et les registres :

- **Les bobines :** ce sont des variables booléenne (bits) qui peuvent être activés (1) ou désactivés (0). Certaines bobines représentent des entées, ce qui signifie qu'elles contiennent l'état de certaines bobines entrées physique discrètes (tout ou rien) ou signifie sortie, c'est-à-dire qu'ils maintiennent l'état du signal de sortie physique.
- **Les registres :** Ce ne sont que des cases mémoire des données non signées de 16-bits. La valeur du registre peut être de 0 à 65535 (de 0 à FFFF en hexadécimal). Il n'y a pas de représentation pour les valeurs négatives ni pour les valeurs supérieures à 65535 et c'est pareil pour les données réelles. Les registres sont divisés en registre d'entrée et registre de stockage. Comme les bobines d'entrée, les registres d'entré indiquent l'état de l'entrée externe [19].

2.6.4 Modes de transmission en Modbus :

Il existe deux modes de transmission dans le protocole Modbus : RTU et ASCII, qui déterminent la manière dont les messages Modbus sont codés. Ces deux modes sont conçus pour être utilisés avec les périphériques séries supportant les protocoles RS-232, RS-485 et RS-422.

a. **Modbus ASCII :**

Lorsque Le contrôleur est configuré pour les communications ASCII sur un réseau Modbus, chaque octet de 8 bits du message est transmis sous forme de deux caractères ASCII. Le principal avantage de ce mode est qu'il nous permet d'avoir un intervalle de temps jusqu'à 1 seconde entre les caractères sans erreur.

b. **Modbus RTU :**

Lorsque le contrôleur est configuré pour communiquer sur le réseau Modbus en utilisant RTU, chaque octet de 8 bits du message contient : deux 4-bits caractères hexadécimaux.

2.6.5 Trame d'échange requête / réponse pour Modbus RTU :

La trame Modbus RTU se compose d'une série des caractères Hexadécimal et contient les informations montrées sur la figure ci-dessous :

	Adresse esclave	Code Requête	Données	contrôle	
START	Adresse	Fonction	Données	CRC	END
Silence	1 octet	1 octet	n octets	2 octets	Silence

Figure (2.9) La trame Modbus RTU.

Afin de communiquer avec l'appareil esclave, le maitre envoie un message contenant [15] :

- Un bit de Start (silence).
- L'adresse de l'esclave : est un nombre compris entre 0 et 247. Les messages envoyés à l'adresse 0 (message de diffusion) peut être reçus par n'importe quel appareil esclave, mais les numéros 1 à 247 sont des adresses spécifiques. À l'exception des messages diffusés, l'esclave répond toujours au message Modbus afin que le maitre sache qu'un message a été reçu.
- Le code de fonction : il définit les commandes que l'esclave doit exécuter, telles que la lecture de données, la réception de donnée, l'état du rapport, etc. la plage des codes des fonctions varie de 1 à 255.

2.6.6 Types de registres Modbus :

Les informations peuvent être stockées dans l'appareil esclave selon deux types différents des données [19] :

Valeurs on/off ou entiers. Ils disposent d'options de lecture seule et de lecture-écriture. La spécification Modbus ne définit pas l'utilisation des registres. Tous les types des registres ne peuvent pas être donc pris en charge par l'esclave **(Voir figure 2.8)**.

	Type d'objet	Accès	Exemples
Discret Inputs	Bit	R	- entrées TOR - Fin de course - Contact auxiliaire de disjoncteur
Coils	Bit	R/W	- Sorties TOR - Bit interne - RAZ d'un compteur d'énergie
Input Registers	Mot	R	- Entrées analogiques - Lecture d'un capteur
Holding Registers	Mot	R/W	- Sorties analogiques - Variable d'un programme (ex: temporisation, opérande d'un calcul...) - Valeur de paramétrage d'un équipement (ex : consigne de vitesse d'un variateur...)

Figure (2.8) Types de registres Modbus.

Comme les nombres décimaux ne sont pas pris en charge, les valeurs contenant une partie décimale sont souvent converties en des valeurs entières.

- **CODE FONCTION :**

Le code fonction définit la requête du maître pour l'esclave. Il existe 19 codes fonctions (**Voir tableau 2.1**) Modbus RS485. Les principales sont les suivantes [18] :

01- Lecture coli statuts.

02 -Lecture input statuts.

03- lecture holding register.

04- lecture input register.

05- écriture d'un coil status.

06 -écriture d'un register.

15- écriture de plusieurs coils.

16- écriture de plusieurs register.

Fonction	Discrétion
01	Lire les bobines
02	Lire les entre discrète
03	Lire le registre interne
04	Lire les registres d'entrée
05	Ecrire une bobine unique
06	Ecrire un registre unique
15	Ecrire plusieurs bobines
16	Ecrire plusieurs registres

Tableau (2.1) Codes des fonctions Modbus RTU.

2.7 Zone mémoire :

Chaque appareil Modbus dispose une mémoire pour stocker les données variables. Modbus définit comment récupérer les données et quels types des données peuvent être récupérés. Cette mémoire est organisée en 4 groupes [21]. **(Voir le tableau 2.2)**

Le type	Plage d'adresse	Accès
Bobine	00001 à 09999	Lire-écrire
Entrée discrète	10001 à 19999	Lire seulement
Registre d'entrée	30001 à 39999	Lire seulement
Registre général	40001 à 49999	Lire-écrire

Tableau (2.2) Zones mémoires des appareils Modbus.

- **Zone des bobines (colis)** :la zone de stockage est organisée en bits (de 00001 à 09999), qui contient la valeur de la sortie TOR discrète.
- **Zone des entrée discrète (discret input)** : la zone est organisée en bits, c'est la valeur de l'entrée TOR, la plage d'adresses est comprise entre 10001 et 19999.
- **Zone des registres d'entrée (input registres) :** Est un registre 16-bits contient la valeur de l'entrée analogique, qui occupe les adresses 30001 à 39999.
- **Zone des registres généraux (Holding registres) :** organisé en 16 bits et occupe des adresses comprises entre 40001 et 49999 où se trouve la valeur de sortie.

2.8 Conclusion :

Ce chapitre a été consacré à la présentation des dispositifs de l'application objet de notre projet de fin d'étude de master basé sur un démonstrateur développée et réalisée au CDTA. Notre étude a concerné l'implémentation d'un module sous le logiciel LABVIEW pour la commande et le contrôle de régulateur de température en utilisant le protocole Modbus RTU, dans le chapitre suivant nous allons présenter la partie logicielle.

Chapitre 3 Etudes de la partie logicielle

3.1 Introduction :

Ce troisième chapitre portera essentiellement sur le logiciel LabVIEW utilisé pour effectuer l'acquisition et le contrôle de température. Nous donnerons les détails et les discrétions concernant la structuration des modules nécessaires pour faire le développement software sous le logiciel.

3.2 Définition du Logiciel LabVIEW :

LabVIEW (Laboratory Virtual Instrument Engineering Workbench) est un logiciel de développement d'applications mis au point par la société américaine **National Instruments**.

Ce logiciel est utilisé dans un grand nombre des domaines plus particulièrement ceux destinés à l'acquisition des données et au traitement du signal. En effet, il offre des larges possibilités de communication entre l'ordinateur et le monde physique par cartes d'acquisitions. Il fournit aussi des multitudes d'outils et des bibliothèques permettent de réaliser des divers traitements sur les signaux mesurés. Son approche totalement graphique offre une souplesse et une arrangement intuitive inégalée. Comparativement aux langages textuels, il offre la même puissance de programmation, mais sans le côté abstrait et complexe lié à la syntaxe [22].

LabVIEW est un logiciel permettant de faire l'acquisition et la génération des données ; l'analyses et les traitements des données ; l'exploitation par la présentation et le sauvegarde des données et enfin permettant de faire la publication et l'échanges des données traites.

3.2.1 L'environnement du logiciel LabVIEW :

LabVIEW est centré autour le principe d'instrument virtuel (Virtual Instrument ou encore VI). Il se décompose en trois parties [22] :

- La face avant.
- Le diagramme.

- Icône/connecteur

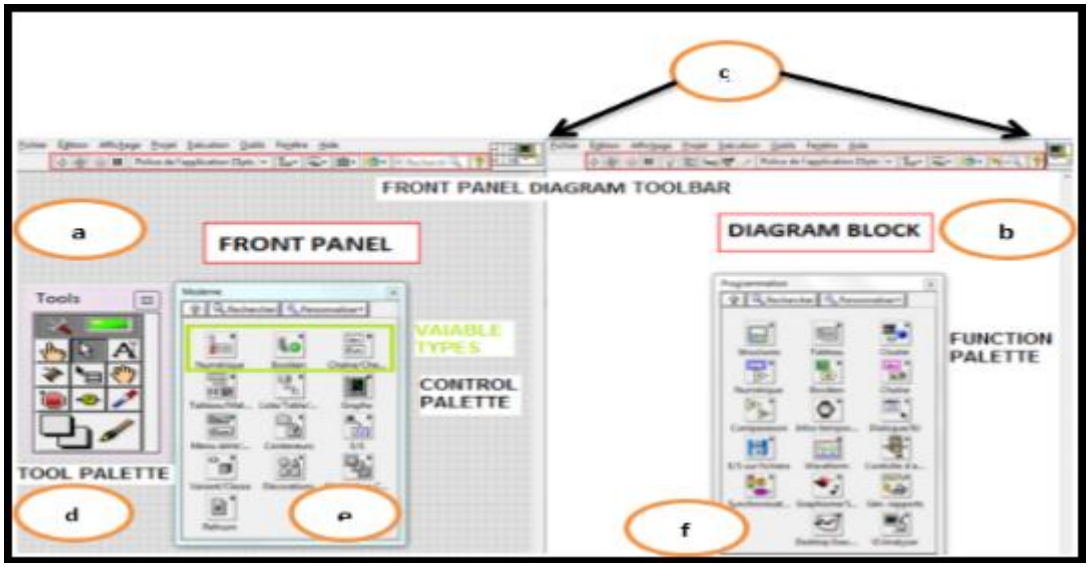


Figure (3.1) les différentes parties du logiciel LabVIEW.

a. La face avant

La face avant **(Voir figure 3.2)** est l'interface utilisateur, elle comporte [23] :

- Les contrôles
- Les indicateurs

Les contrôles simulent les entrées des instruments virtuels et fournissent les données au diagramme. Les indicateurs simulent la réponse des instruments et affichent les données acquises ou engendrées par les VIs.

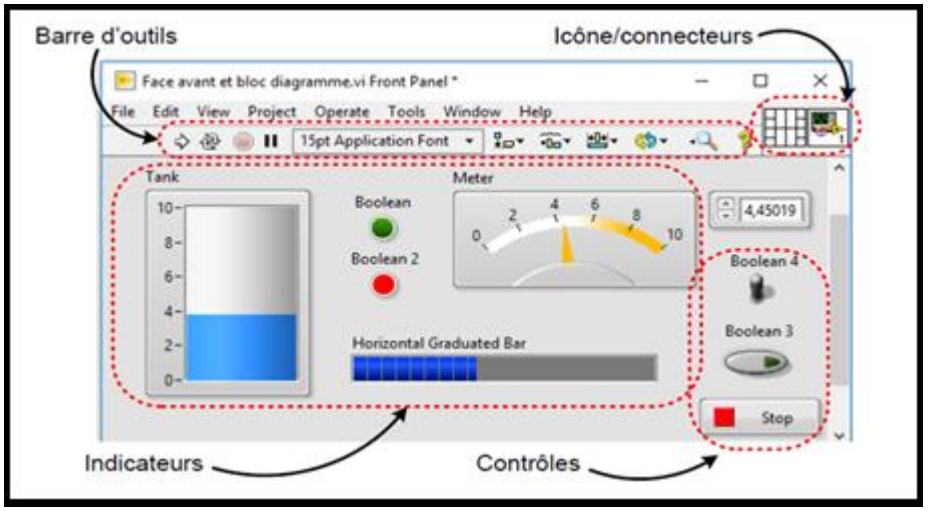


Figure (3.2) Fenêtre face Avant [22].

b. Le diagramme

Le diagramme est chargé de mettre en relation les entrées et les sorties ainsi que les différents éléments fonctionnels du programme **(Voir figure 3.3)**, Il se compose de [23] :

- **Nœuds**

Les nœuds sont des objets sur le diagramme. Ils possèdent des entrées et/ou des

sorties. Ils sont équivalents à des fonctions dans les langages textuels.

- **Terminaux**

Les objets situés dans la face avant apparaissent comme des terminaux dans le diagramme. Les terminaux sont des ports de communication entre la face avant et le diagramme. Ils sont équivalents aux paramètres et aux constantes dans les langages textuels.

- **Fils**

L'ensemble des terminaux et des nœuds sont mis en relation par des fils pour transférer les données dans le diagramme. Les fils sont analogues aux variables dans les langages textuels.

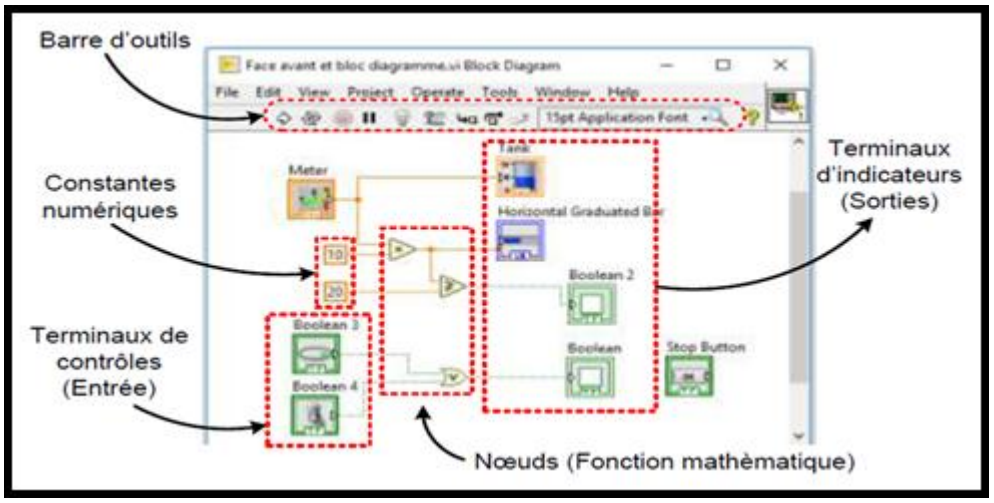


Figure (3.3) Fenêtre diagramme [22].

c. *Création des sous-VIs*

Un sous VI est un VI qui peut être utilisé dans un autre VI de plus haut niveau, comme un sous-programme appelé par un programme principal dans un logiciel de langages de programmation classique [24].

Son utilisation présente les avantages suivants :

- ⇨ Modularité : création des blocs de base réutilisables pour diverses applications pour un gain de productivité.
- ⇨ Facilite le « débogage ».
- ⇨ Nécessite une seule création de code.

d. *La palette d'outil*

La palette d'outil existe sur le diagramme et sur la face avant. Elle permet de modifier des valeurs, des couleurs, mais aussi de câbler les entrées et les sorties des icônes

entre elles, de poser des points d'arrêt, des sondes… [23].

e. La palette de commande

La palette de commande est uniquement accessible depuis la face avant elle permet de créer toutes les commandes et l'indicateurs quelques soient leurs types, Dans cette palette nous trouverons tous les éléments (commandes et indicateurs) nécessaires à la création de la face avant d'une application. Elle est disponible à partir de la fenêtre face avant par un clic droit avec la souris ou dans la barre des menus [23] :

. Nombres
. Booléens
. Chaînes de caractères
. Tableaux
. Clusters
. Graphiques
. Types spéciaux

f. La palette de fonctions

La palette de fonctions est seulement accessible sur le diagramme, elle permet de créer toutes les fonctions du LabVIEW. Dans cette palette nous trouverons tous les éléments (fonctions de base, VI Express, etc.) nécessaires à la création du code graphique dans la fenêtre diagramme. Elle est disponible à partir de la fenêtre diagramme par un clic droit avec la souris ou dans la barre des menus [23] :

. Structures
. Opérations numériques
. Opérateurs logiques
. Comparaisons
. Fonctions temporelles
. Entrées/Sorties
. Acquisition
. Traitement du signal

3.2.2 La palette de structures :

Les boucles permettent l'exécution d'un programme, d'un sous-programme ou d'une partie de programme jusqu'à une action ou une valeur définie par l'opérateur : La boucle FOR et La boucle WHILE.

Les structures permettent d'organiser, de séquencer ou de conditionner les éléments d'un V.I : Timed structure, Case structure et Event Structure [25].

a. La boucle FOR :

La boucle FOR **(Voir figure 3.4)** répète une partie du code diagramme un nombre déterminé de fois, ce nombre étant définissable par l'opérateur. Elle est définie par 2 terminaux [25] :

- Terminal de comptage.
- Terminal d'itération.

La valeur dans la borne de comptage « N » (une borne d'entrée) indique combien de fois répéter le sous-diagramme. Définisse le nombre explicitement en câblant une valeur de l'extérieur de la boucle vers le côté gauche ou supérieur du terminal de comptage, ou définissez le nombre implicitement avec l'auto-indexation. L'indexation

Le terminal d'itération «i» (un terminal de sortie) contient le nombre d'itérations terminées. Le nombre d'itérations commence toujours à zéro. Lors de la première itération, le terminal d'itération renvoie 0.

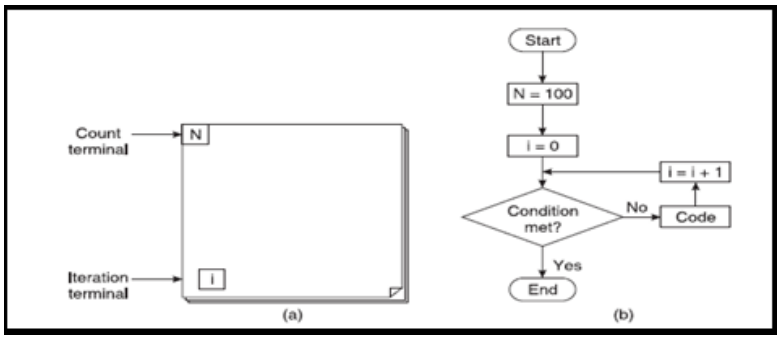


Figure (3.4) : (a) Boucle For dans LabVIEW et (b) Organigramme équivalent à la boucle For [25].

b. la boucle WHILE :

La boucle WHILE **(Voir figure 3.5)** répète le code diagramme contenu à l'intérieur de la boucle jusqu'à un changement d'état de la variable booléenne associé au terminal conditionnel. Elle est définie par 2 terminaux [25] :

- Terminal condition booléen
- Terminal d'itération

Elle répond au codage algorithmique, Faire exécuter le diagramme contenu à l'intérieur de la boucle Tant que la condition est vraie et elle présente un Terminal d'itération « i » qui est une sortie, il contient le nombre de fois que la boucle s'est exécutée, sa valeur initiale est zéro et vaudra zéro si la boucle ne s'exécute qu'une seule fois. Elle présente également un terminal conditionnel qui est une entrée, cette entrée est testée à la fin de chaque itération. Lorsque ce terminal est de couleur verte, la boucle continue à s'exécuter tant que la valeur du fil arrivant au terminal est « vraie ou True ». La boucle s'exécute donc toujours au moins une fois. On peut changer l'état de cette condition et passer en rouge en signifiant que la boucle continuera à s'exécuter tant que la condition sera fausse.

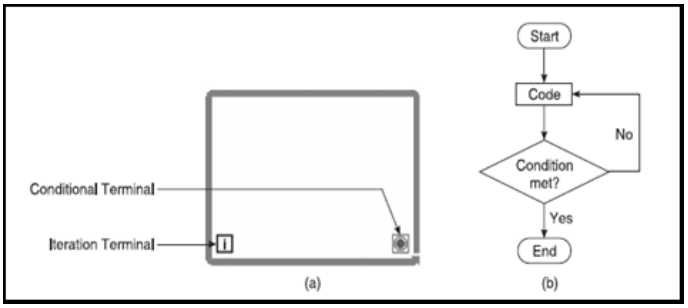


Figure (3.5) : (a) Boucle While dans LabVIEW et (b) Organigramme équivalent à la boucle While[25].

c. La structure condition :

La structure Case **(Voir figure 3.6)** est une structure de contrôle de programmation permettant d'effectuer plusieurs choix [26].

La structure CONDITION est organisée sous forme de fenêtres associées :

- ⇨ Une seule case est visible à la fois et elle correspond à l'un des cas de figures du terminal de sélection.
- ⇨ Chaque case contiendra un sous-diagramme et 1 seul cas s'exécute à la fois selon le terminal de sélection.

Figure (3.6) Case structure.

d. La structure de séquence :

La structure de « séquence » **(Voir figure 3.7)** permet de spécifier l'ordre d'exécution de flots de données. Cette structure se présente sous la forme d'un cadre et a le statut d'un nœud [26].

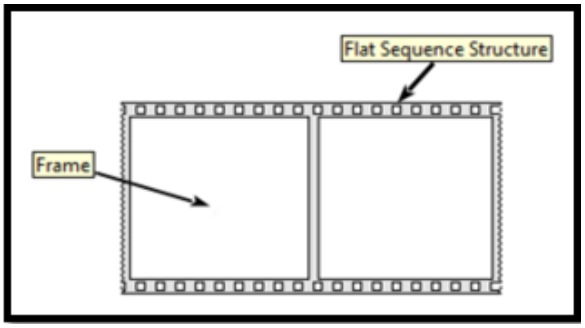


Figure (3.7) structure de séquence.

3.3 Conclusion :

Ce troisième chapitre a porté essentiellement sur le logiciel LabVIEW utilisé pour effectuer l'acquisition et le contrôle de température. Nous avons donné les détails et les descriptions concernant la structuration des modules nécessaire pour faire le développement software sous le logiciel.

Nous avons vue dans ce chapitre les principales caractéristiques du logiciel LabVIEW et les fonctions permettant de réaliser des programmes sous LabVIEW.

Chapitre 4 Implémentation et résultats

4.1 Introduction :

Ce dernier chapitre sera dédié aux résultats obtenus suite à l'implémentation d'un programme de contrôle et de commande d'un régulateur de température EMKO de type ECO PID, complément à un travail de PFE réalisé l'année passer au niveau de CDTA et qui a concerné le développement d'un programme d'interface uniquement pour la lecture sous le logiciel LabVIEW via le PC à travers un port COM avec un adaptateur convertisseur USB/RS-485 et en utilisant un régulateur de température EW4822.

Notre réalisation concerne le développement d'une interface d'opérateur pour la commande et le contrôle (lecture et écriture) sous LabVIEW pour un régulateur de température de type EMKO Eco PID. Le dispositif regroupe deux contenaires pour mettre les huiles alimentaires de test, qui sont chauffés par deux résistances chauffantes de 100 watt. La température de l'huile est contrôlée et commandée via un adaptateur USB/RS-485 par un PC. La commande et le contrôle sont assurés par l'interface développée sous LabVIEW ou on peut afficher la valeur de la consigne et établir la valeur de processus.

Notre travail consiste essentiellement de faire le câblage de dispositif ensuite le développement software de l'interface sous LabVIEW qui est détaillé par la suite. Puisque le travail est duel, une seule interface sous LabVIEW est développée.

4.2 Câblage du dispositif de test :

Avant de passer à la réalisation globale pour le projet on doit assurer le câblage et les connexions des E/S du système régulateur de température et le PC via le convertisseur adaptateur USB /RS-485 (**Voir figure 4.1)** présentant l'assemblage des éléments constituant le dispositif de test :

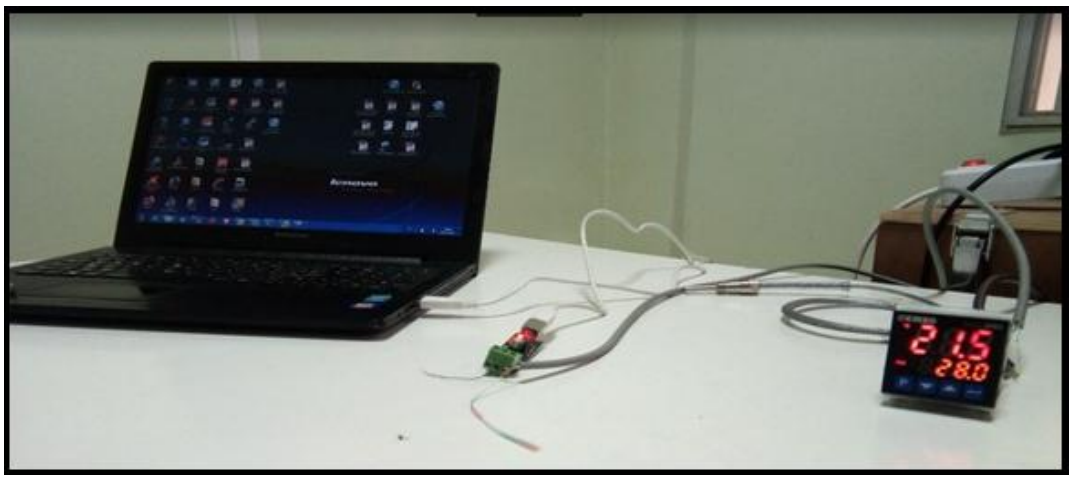


Figure (4.1) Dispositif de test.

Le standard de transmission du BUS RS-485 se fait par deux fils (Half-duplex). Le régulateur EMKO et le convertisseur adaptateur RS485/USB sont censés être câblés (+) à (+) et (-) à (-). **(Voir figure 4.2)**

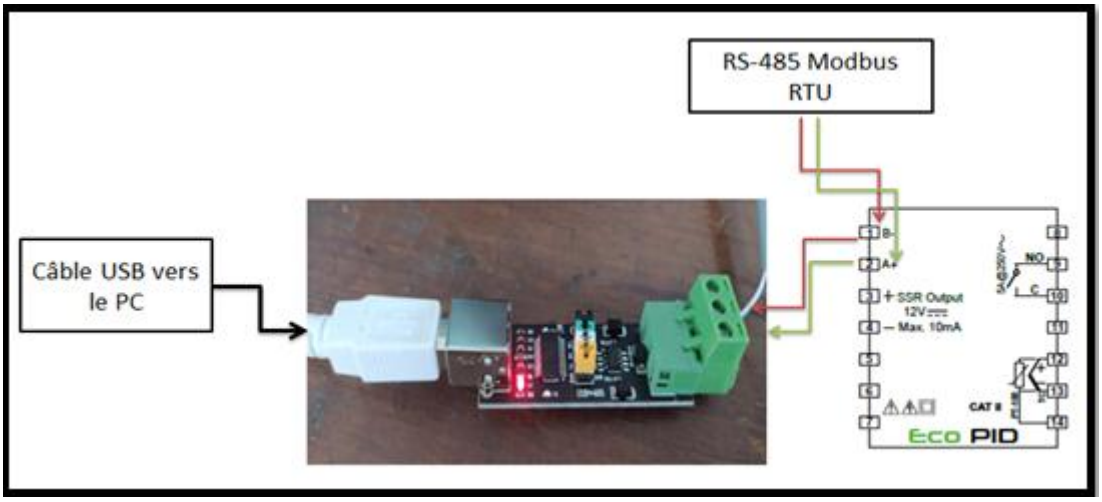


Figure (4.2) Schéma de standard de transmission RS-485.

4.3 Réglage des éléments du dispositif :

Dans cette partie nous allons citer tous les paramètres qu'on doit régler pour établir une connexion en lecture/écriture selon le tableau suivant **(Voir tableau 4.3)** :

Paramètre	EMKO Eco PID	Port COM	LabVIEW
Adresse d'esclave	01	Aucun réglage	01
Débit en baud	9600	9600	9600
Protocol	MODBUS	MODBUS RTU	MODBUS RTU
Bit d'arrête	1 Bit	1 Bit	1 Bit
Control de flux	Aucun réglage	Aucun	Aucun
Bit de parité	Aucun	Aucun	Aucun

Tableau (4.1) Réglages des paramètres dans les 3 niveaux.

4.3.1 Réglage des paramètres du régulateur EMKO eco PID :

Comme nous l'avons décrit dans le deuxième chapitre, notre régulateur de température EMKO eco PID possède pour accéder aux différents paramètres un menu (paramètre de menu dispositif, menu de contrôle, menu de PID, menu de communication, menu de l'alarme, menu de protection).

Afin de régler ce qui est nécessaire pour établir la communication entre l'instrument et le PC, on commence par entrée manuellement les différents paramètres concernant le régulateur en utilisant le menu <<Set>> **(Voir figure 4.3).**

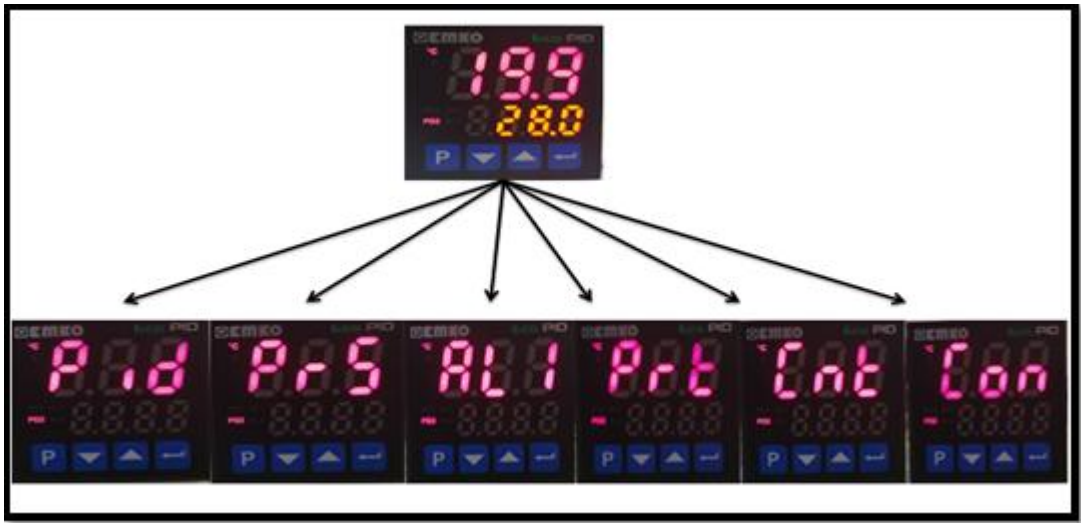


Figure (4.3) Différents paramètres du régulateur de température EMKO eco PID.

4.3.2 Réglages du port COM :

Après avoir connecté le convertisseur USB/RS-485 avec le régulateur EMKO eco PID et s'assuré que le câblage est bien fait, ainsi que le branchement du convertisseur USB/RS-485 au PC, ce dernier reconnait le convertisseur USB/RS-485 comme un nouveau périphérique. Nous aurons par la suite installé le driver du port COM <<virtuel>> de Windows. Ce port COM a besoin d'un numéro (COM°n) qui va être utilisé pour le port COM du maitre LabVIEW, pour savoir sur quel port COM le convertisseur est installé ensuite on règle les paramètres conformément au **Tableau 4.3** ci-dessus.

L'acheminement suivant indique séquentiellement la démarche suivante **(Voir figure 4.4)** :

Démarrer ⟹ Gestion de l'ordinateur ⟹ Gestionnaire de périphérique ⟹ Ports (COM et LPT).

Pour accéder au réglage des paramètres du port COM, on fait un clic droit sur ⟹ <<USB Serial Port (COM8) ⟹ propriétés ⟹ paramètres de USB serial port ⟹ Avancé

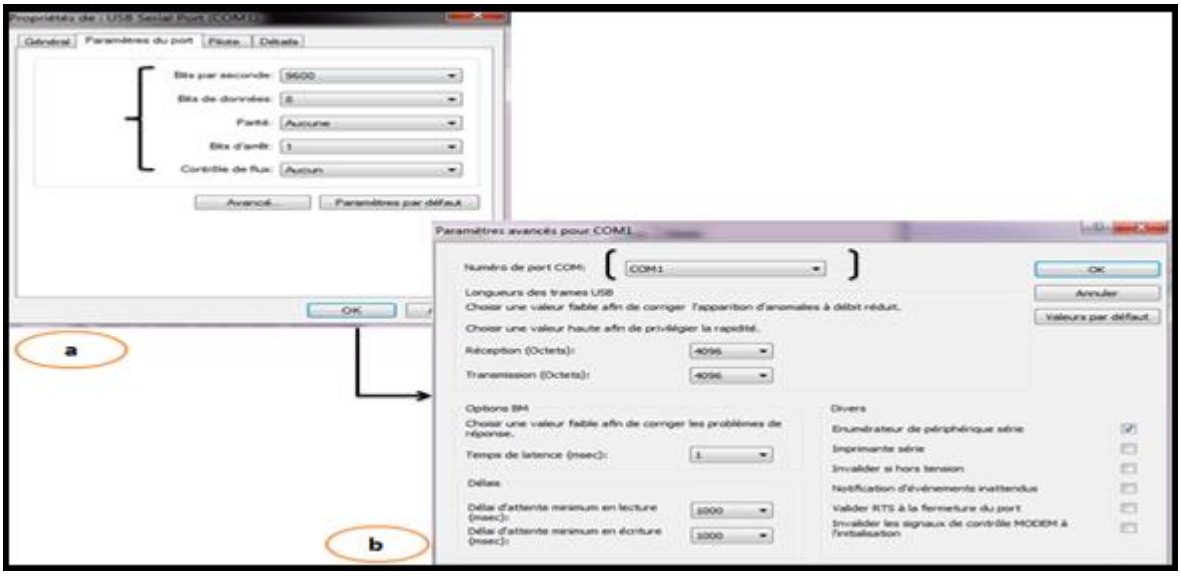


Figure (4.4) Réglages des paramètres du Port COM.

La **figure (4.3.a)** indique des paramètres à régler, à savoir :

- Bits/ seconde : 9600.
- Bits de données : 8.
- Parité : Aucune.
- Bits d'arrêt : 1.
- Contrôle de flux : Aucun.

La **figure (4.3.b)** indique des paramètres à régler, à savoir :

- Numéro du port COM : COM1
- Longueurs des trames USB : on a pris les valeurs maximales pour la réception et la transmission 4096 Octets pour privilégier la rapidité.
- Temps de latence : on a pris la valeur la plus faible 1 ms
- Le délai d'attente en lecture/écriture : 1000 ms
- Options diverses

4.4 Test de connectivité et activation du MODBUS :

Les tests de connectivité peuvent s'effectués grâce à plusieurs logiciels de test qui peuvent être payants ou gratuits. Parmi eux, on peut citer le logiciel Modbus

tester(lecteur) et le logiciel Modbus/Jbus (lecteur, écriture) tester qui seront décrits dans les sections suivantes.

4.4.1 Logiciel Modbus tester :

Le logiciel Modbus tester **(Voir figure 4.5)** est un logiciel gratuit qui permet de tester la communication entre le maitre et les machines esclaves. Nous pouvons lire les registres 4XXXX / 3XXXX (Holding / Input), les entrées 1XXXX, les sorties 0XXXX. Les données peuvent être de formats (décimale, hexadécimal, binaire).

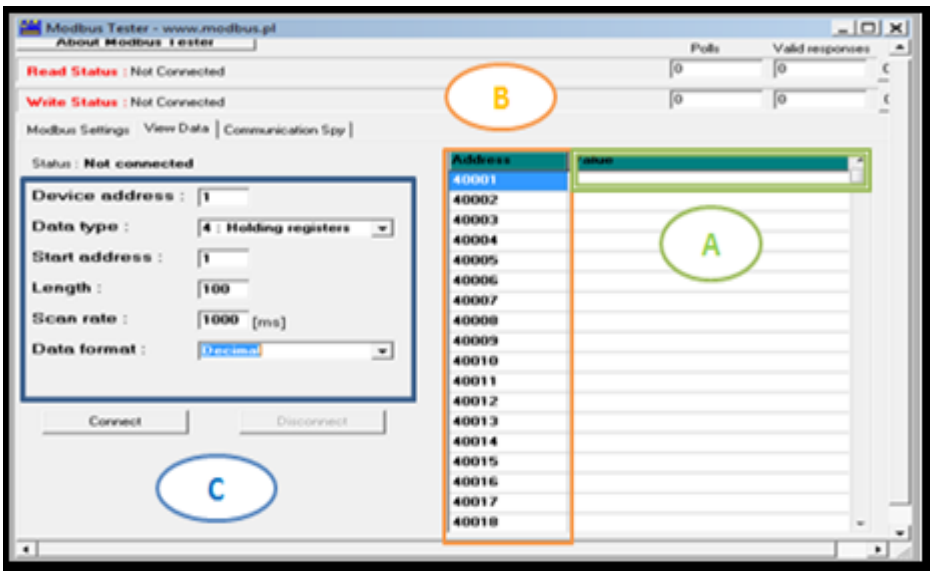


Figure (4.5) Interface du logiciel Modbus tester.

Description des trois parties :

- **Panneau A** : permet de d'afficher les valeurs du paramètre dans le régulateur.
- **Panneau B** : permet d'afficher les adresses.
- **Panneau C** : permet de choisir le format des données (HEX, Décimal, binaire...etc.) et les types des données (Input, Coil, Input register, Holding register).

4.4.2 Logiciel Modbus/Jbus tester :

Modbus/Jbus tester est un logiciel simple de lecture et l'écriture qui permet de lire à partir d'un registre et d'écrire dans un registre d'un appareil esclave Modbus, La **figure (4.6)** ci-dessous détaille chaque bloc utilisé.

- **Panneau A** : permet de d'afficher les valeurs du paramètre dans le régulateur.
- **Panneau B** : permet d'afficher les adresses.

• **Panneau C** : permet de choisir le format des données (HEX, Décimal, binaire...etc.) et les types des données (Input, Coil, Input register, Holding register).

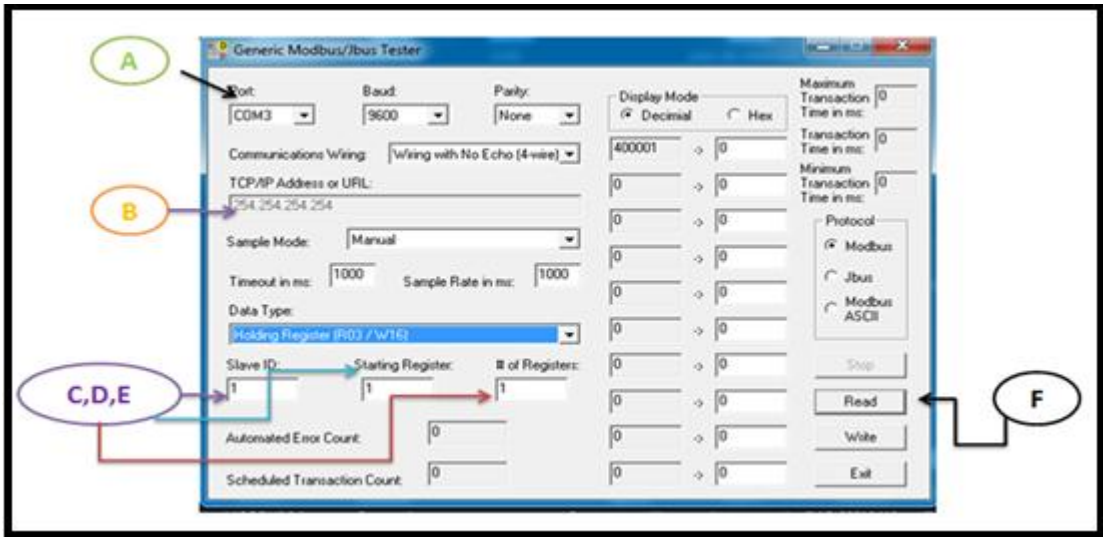


Figure (4.6) Interface du logiciel Modbus/jbus tester.

Description des parties :

- **Panneau A** : le numéro du port COM.
- **Panneau B** : l'adresse IP d'une passerelle Ethernet.
- **Panneau C :** l'adresse d'esclave**.**
- **Panneau D :** le registre de départ que nous voulons lire.
- **Panneau E** : le nombre de registres que nous souhaitons lire dans un ordre séquentiel en commençant par le registre de départ.
- **Panneau F :** lorsque tous les paramètres sont configurés, on va cliquer sur <Read>.

4.5 Travail expérimental et résultats :

Dans cette section, on va discuter les résultats des tests d'une lecture/écriture à partir de notre instrument. Parmi les menues et les paramètres dont on s'est intéressé sont indiqués dans les **tableaux (4.2,3,4,5,6).**

Menu	discrétion	adresse	action	Par défaut
Modbus	Valeur de température affichée	30000	Lecture	℃
	Etat des LEDs	30001	Lecture	9 pour ℃
	Etat de l'appareil	30002	Lecture	0
	Point de consigne	40000	Lecture/écriture	℃

Tableau 4.2 Les paramètres de menu Modbus du régulateur EMKO eco PID.

Menu	paramètre	discrétion	adresse	action	Par défaut
prs	pin	Type de processus	40004	Lecture/écriture	0
	uni	l'unité (°C ou F)	40005	Lecture/écriture	°C
	plo	Echelle minimal	40006	Lecture/écriture	-199
	pup	Echelle maximal	40007	Lecture/écriture	900
	sul	consigne minimal	40008	Lecture/écriture	-199
	suu	Consigne maximal	40009	Lecture/écriture	900
	pof	Valeur de décalage	40010	Lecture/écriture	0
	ift	Temps de filtrage	40011	Lecture/écriture	1.0

Tableau 4.3 Les paramètres de menu processus du régulateur EMKO eco PID.

Menu	paramètre	discrétion	adresse	action	Par défaut
cnt	out	Type de sortie	40015	Lecture/écriture	SSR
	prs	Type de dispositif	40016	Lecture/écriture	Chauffer
	cns	Type de contrôle	40017	Lecture/écriture	3
	sbo	Valeur de sortie	40019	Lecture/écriture	0
	sbd	Type de capteur	40020	Lecture/écriture	sbr
	sst	Consigne réglée	40021	Lecture/écriture	no
	sco	Sortie de contrôle	40022	Lecture/écriture	10.0
	sct	Temps de contrôle	40023	Lecture/écriture	1.0

Tableau 4.4 Les paramètres de menu contrôle du régulateur EMKO eco PID.

Menu	paramètre	discrétion	adresse	action	Par défaut
PID	tun	calcule PID automatiquement	40027	Lecture/écriture	no
	prb	Bande proportionnel	40028	Lecture/écriture	10.0
	tin	Temps de l'intégral	40029	Lecture/écriture	100
	tde	Temps de dérivation	40030	Lecture/écriture	25.0
	tco	Période Contrôle de sortie	40031	Lecture/écriture	1.0
	sof	Décalage de la consigne	40032	Lecture/écriture	0

Tableau 4.5 Les paramètres de menu PID du régulateur EMKO eco PID.

Menu	paramètre	discrétion	adresse	action	Par défaut
con	adr	Accès de l'adresse de l'appareil	40056	Lecture/écriture	1
	bau	Débit en bauds	40057	Lecture/écriture	3
	par	Parité	40058	Lecture/écriture	0
	stb	Bit de stop	40059	Lecture/écriture	0

Tableau 4.6 Les paramètres de menu communication du régulateur EMKO eco PID.

4.5.1 Application sous le logiciel Modbus tester :

A titre de test, nous présentons la lecture du point de consigne. Cependant En ce qui concerne les menus Prs (process menu) les paramètres PID sont rapporter dans **l'annexe 1a**.

- **Lecture du point de consigne :**

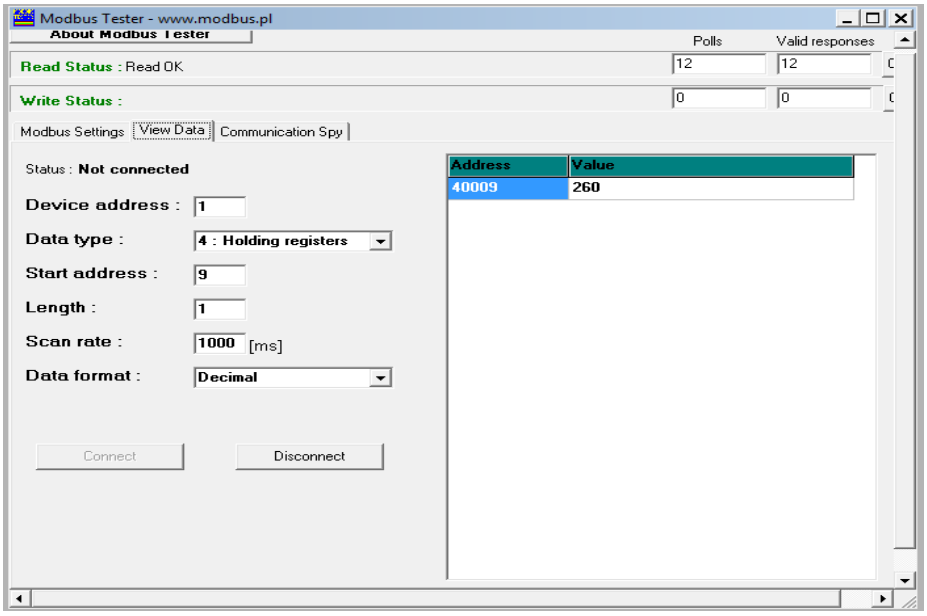


Figure (4.7) Résultat de lecture du point de consigne sur Modbus tester.

Un autre logiciel de test Modbus/Jbus a été utiliser pour la lecture des paramètres de menu Prs ainsi que l'écriture du paramètre PuP avec le réglage des paramètres du port série. Les démonstrations sont dans **l'annexe 1b**.

4.6 Implémentation du contrôle et de la commande du régulateur de température sous logiciel LabVIEW :

Dans cette partie nous entamons notre application concernant le contrôle et la commande de régulateur de température par l'implémentation de deux programmes sous LabVIEW, le premier programme pour la lecture et le deuxième programme pour l'écriture à l'aide de la bibliothèque ModBus. Pour ce faire nous commençons par la vérification de la communication et l'utilisation de Protocole Modbus.

4.6.1 Vérification de la communication et l'utilisation de Protocole Modbus :

La bibliothèque MODBUS est un ensemble d'instruments virtuels (VI) téléchargeables gratuitement à partir du NI Package manager, qui peuvent fournir une communication Modbus via un port série standard ou n'importe quel port Ethernet.

La **figure (4.12)** présente les VI's composant notre block diagramme :

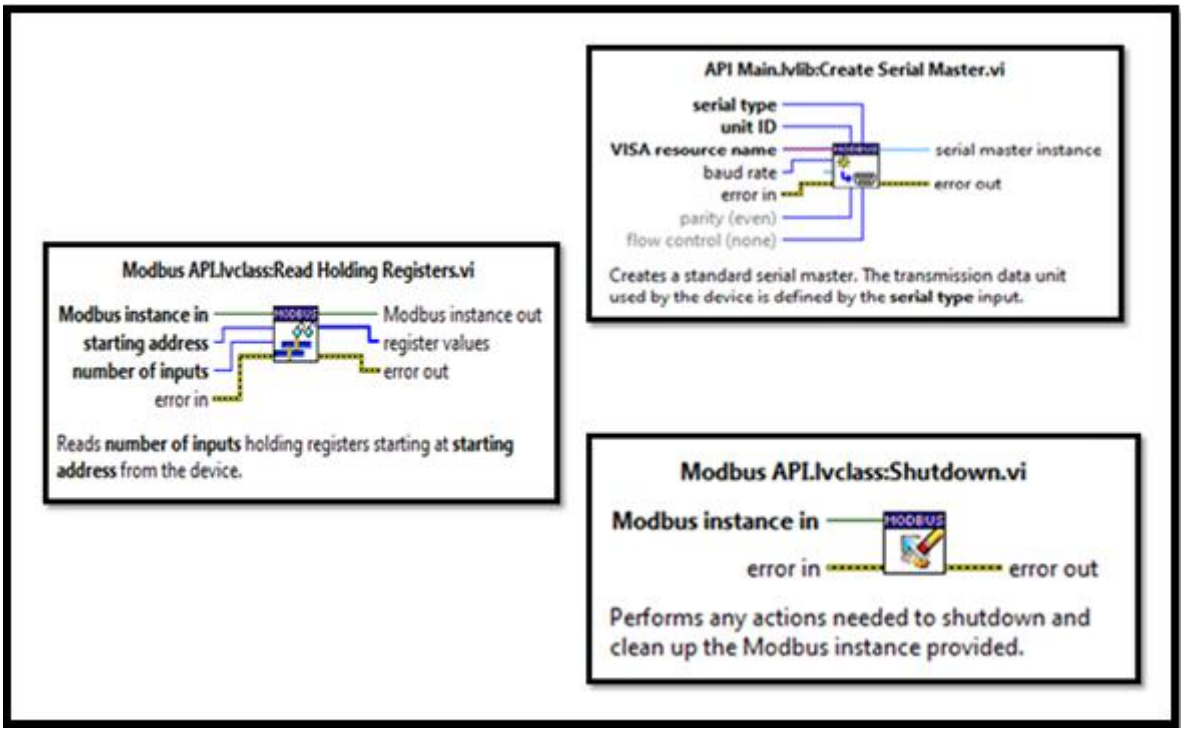


Figure (4.8) Instruments virtuels nécessaires pour l'implémentation de lecture sur LabVIEW utilisant Modbus Library.

En connectant l'ensemble des VI's indiqués sur **la figure (4.8)**, nous obtiendrons le VI final **(Voir figure 4.9)** qui nous permet de lire les valeurs des quelques registres choisis précédemment à partir des **tableaux (4.5)** ci-dessus.

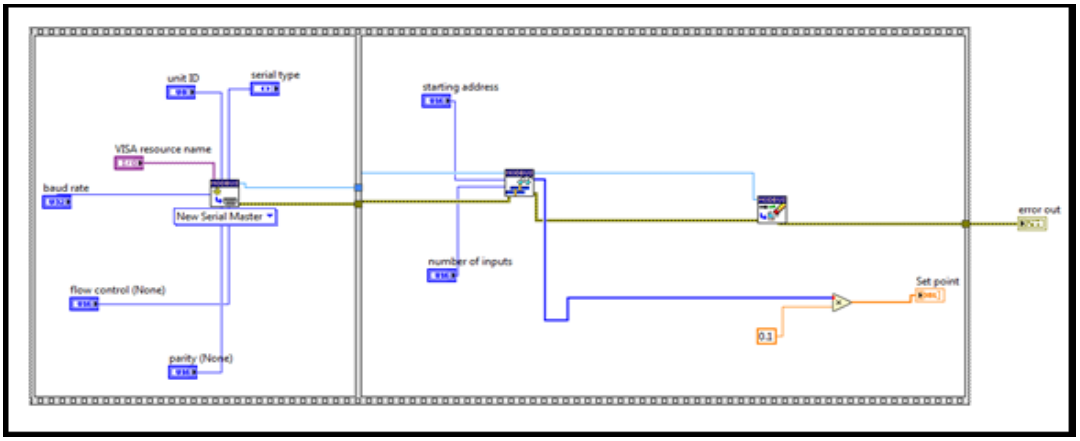


Figure (4.9) bloc diagramme de la lecture des registres du régulateur EMKO eco Pid.

4.6.2 Procédure de lecture en Modbus RTU sous LabVIEW :

La procédure de lecture est réalisée conformément aux schémas copie écran **(Voir**

Annexe 2a).

- **Lecture du registre de la valeur de consigne :**

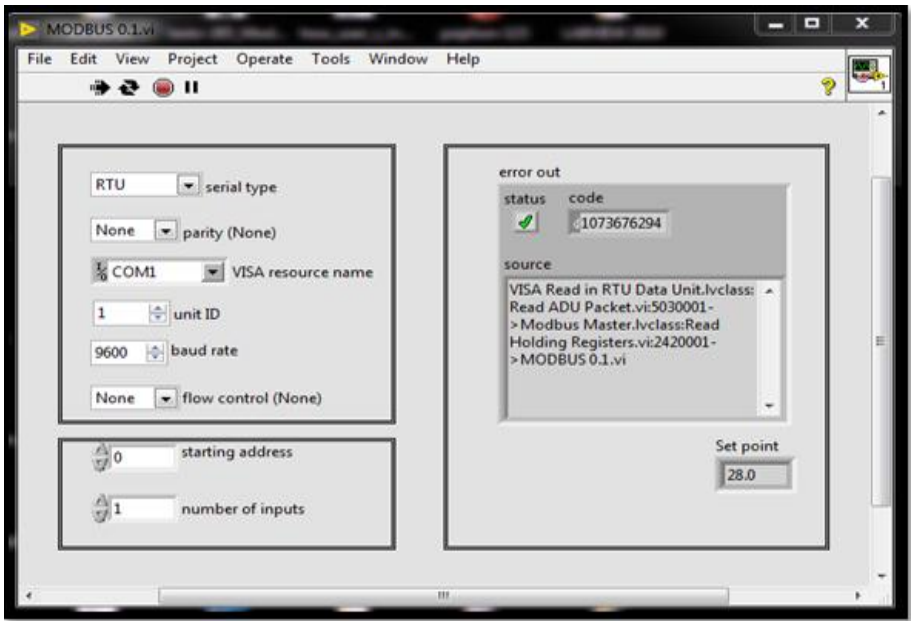


Figure (4.10) résultat de la lecture du registre du point de consigne sur la face-avant du logiciel LabVIEW.

4.6.3 Procédure de l'écriture en Modbus RTU sous LabVIEW :

Maintenant en va faire l'écriture des quelques paramètres de régulateur et ensuite on va donner les résultats de l'écriture de chaque registre. La figure ci-dessous présente les VI's composants virtuels nécessaires pour l'implémentation de l'écriture sur LabVIEW en utilisant Modbus Library.

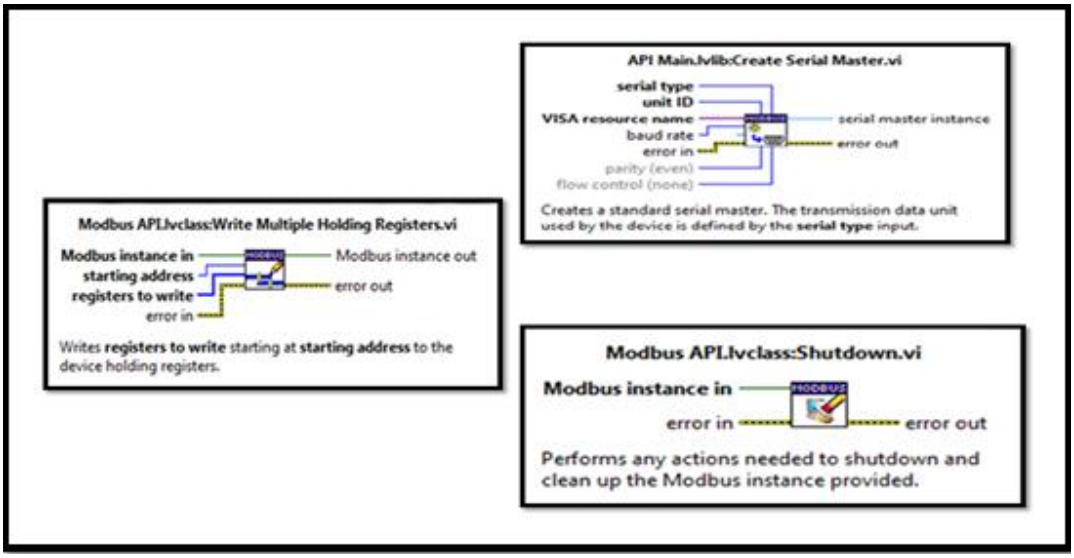


Figure (4.11) Instruments virtuels nécessaire pour l'implémentation de l'écriture sur LabVIEW utilisant Modbus Library.

L'implémentation de ce VI indiqué ci-dessus en séquences, donnera le résultat de

chaque écriture respective des registres adressés sur la face-avant (**voir tableaux 4.2 ,3,4,5,6**) :

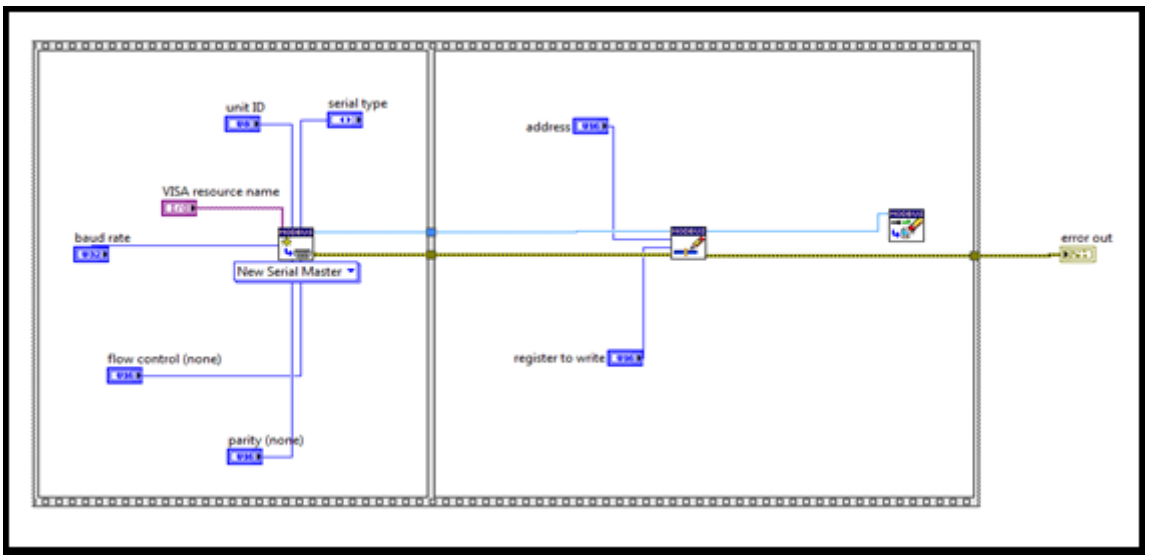


Figure (4.12) Bloc diagramme de l'écriture du registre du régulateur EMKO.

La procédure de l'écriture est réalisée de la même façon qu'en écriture par des schémas copie écran **(Voir Annexe 2b)**.

- **L'écriture du la consigne :**

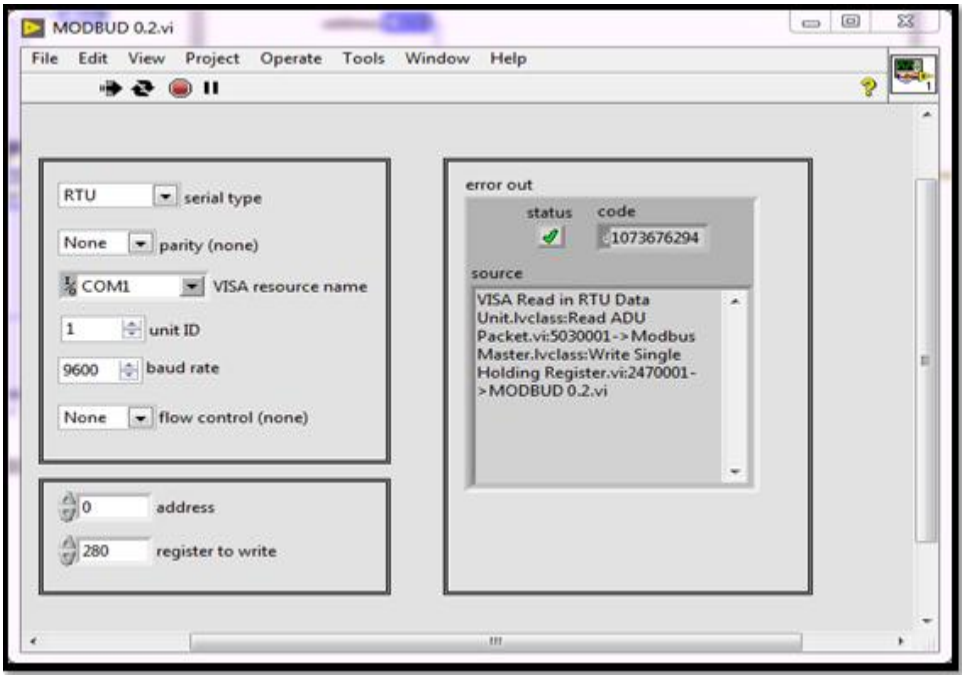


Figure (4.13) résultat de l'écriture du point de consigne sur la face-avant du logiciel LabVIEW.

4.7 Développement et implémentation d'interface opérateur :

Dans cette partie on va détailler les différentes étapes de développement et l'implémentation. Il s'agit d'un software sous logicielle LabVIEW pour automatiser la procédure manuelle de réglage et d'exploitation du régulateur EMKO eco PID (selon le

data sheet) par l'opérateur. Le travail est réalisé on utilisons le modèle type « Producer - consumer » disponible sous LabVIEW et qui nous permet de créer une interface utilisateur basée sur deux boucles fonctionnant en parallèle.

La boucle « Producer », répond immédiatement aux interactions utilisateur telles que les clicks de bouton et les mouvements de souris, ensuite envoi des commandes via une file d'attente à la boucle « consumer » qui effectue les tâches requises [28].

La division de la représentation d'état (state machine) en deux boucles permet à l'interface utilisateur de rester réactive si une tâche consommateur nécessite un temps inhabituel ou doit attendre jusqu'à une ressource partagée soit disponible.

Les deux boucles citées ci-dessus se référencent chacune à un fil d'attende commun appelé « Queue ». Dont la présentation sous LabVIEW est sous forme de différentes fonctions **(Voir figures annexe 2c).** Ces dernières sont utilisées pour implémenter notre programme et rapportées dans l'annexe 2c.

a. Boucle de producteur :

La structure de cette boucle gère les clics sur les boutons qui activent l'état de la boucle consommateur qui lui est propre sous la forme d'une chaîne "state : parameters" dans laquelle l'option "parameters" **(Voir figure 4.16)** fournit des paramètres spécifiques à l'état. (**Voir figure 4.14) :**

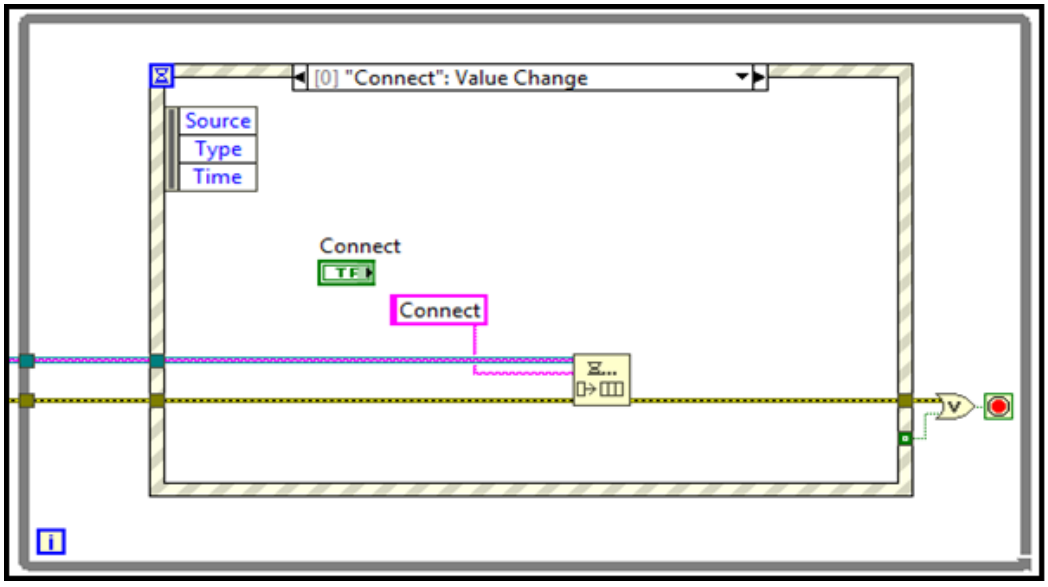


Figure (4.14) bloc diagramme de l'état boucle producteur pour le bouton « connect ».

Afin d'empêcher les utilisateurs de fermer le front-panel on va utiliser un état de

« **panel Closed? Filter event »** qui capture et rejeté l'action de fermeture du panneau en câblant une constante Vrai à la fonction « **Discard? Filter node »** sur le côté droit de la structure event où se trouve le bouton connect. (**Voir figure 4.15)** :

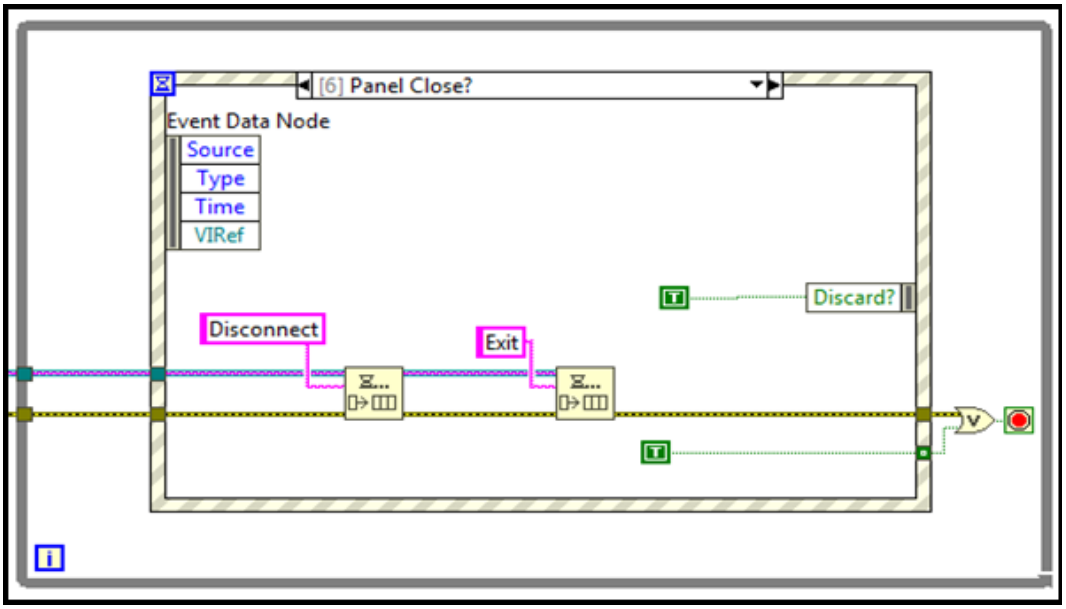


Figure 4.15 bloc diagramme d'état « **panel Closed? Filter event** ».

b. Boucle de consommateur :

Cette boucle utilise (Queue state machine) qui est basée sur (Queue) commune.

Le « clear error.vi », inspecte le cluster d'erreurs à chaque itération de la boucle par :

- Pas d'erreur – à l'aide de la fonction « Dequeue » la boucle consommateur exécute la commande ordonnée à partir de la boucle de production pour exécuté le programme parallèle.
- En cas d'erreur « Erreur » - l'ordre passera directement à l'état « shutdown ».

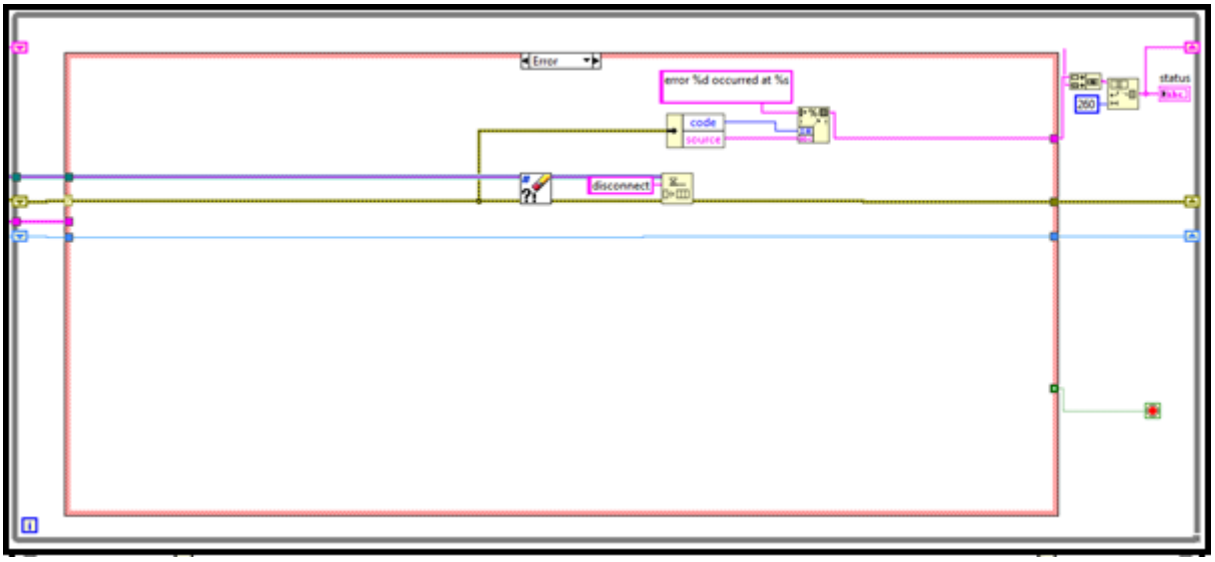


Figure 4.16 bloc diagramme de l'état d'erreurs de la partie consommateur.

En cas ou « Pas Erreur », La structure de case « case structure » décode la

« Dequeue » de la boucle du producteur pour exécute les commandes suivant :

- **« Default » :** La première chose à faire à l'intérieur de nos programmes et de crée un état par défaut, cette état gère toutes les variables entrantes possibles livrées par « Dequeue element ». (**Voir figure 4.17)**

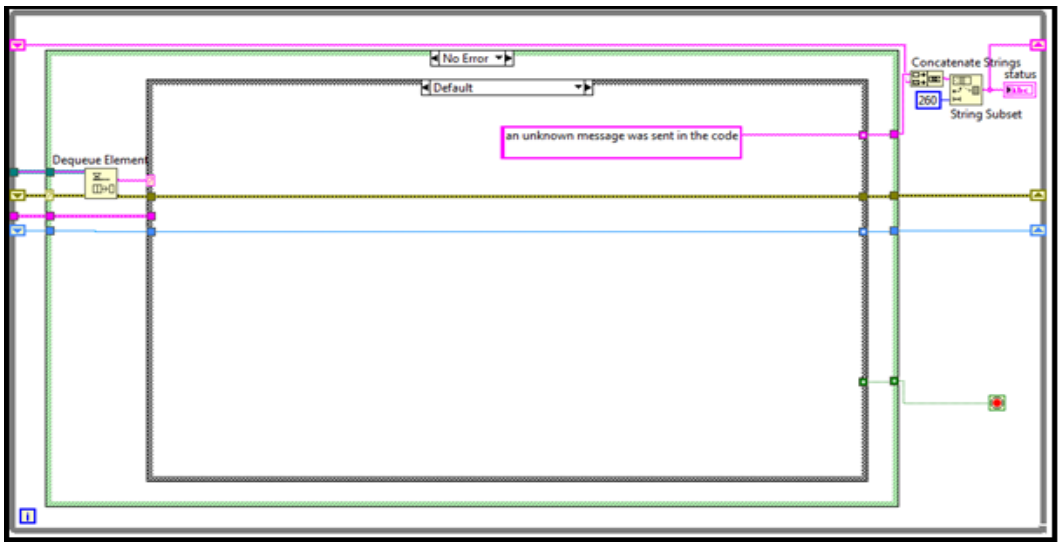


Figure 4.17 bloc diagramme de l'état « Default ».

En cas d'ordre « connect », ce dernier s'exécute comme suit :

- **« Connect »** : crée le point de terminaison pour canaliser le flux réseau (données, informations…). Désactiver le bouton « connect » et activer les boutons du contrôleur système. **(Voir figure 4.18)**

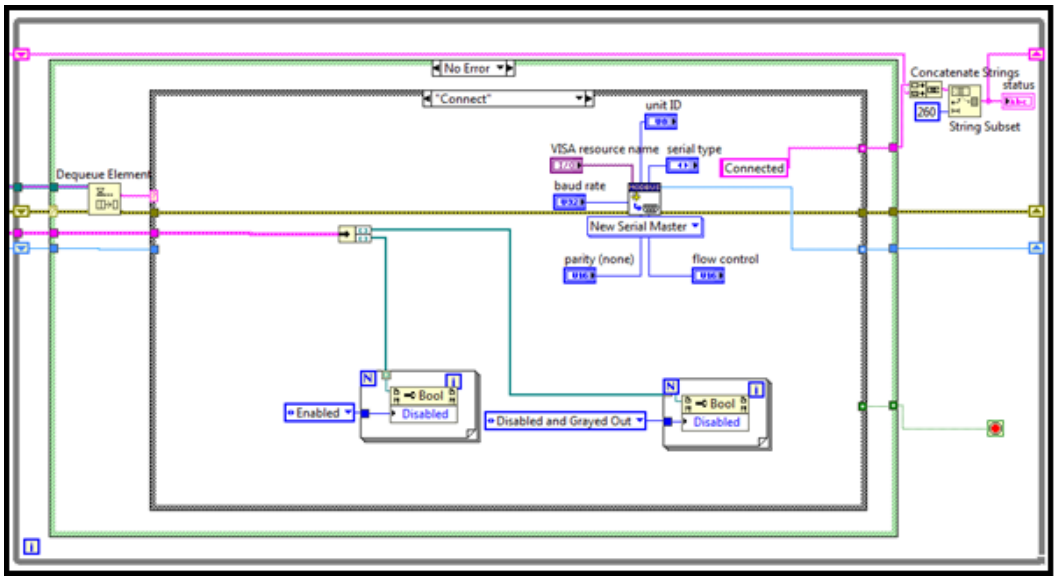


Figure 4.18 bloc diagramme d'état « connect ».

En cas de déconnexion :

- **« Disconnect »** : envoyer le message de commande "Disconnect" pour déconnecter et ôter le point de terminaison du flux réseau, puis remettre les boutons à leurs états par défaut. **(Voir figure 4.19)**

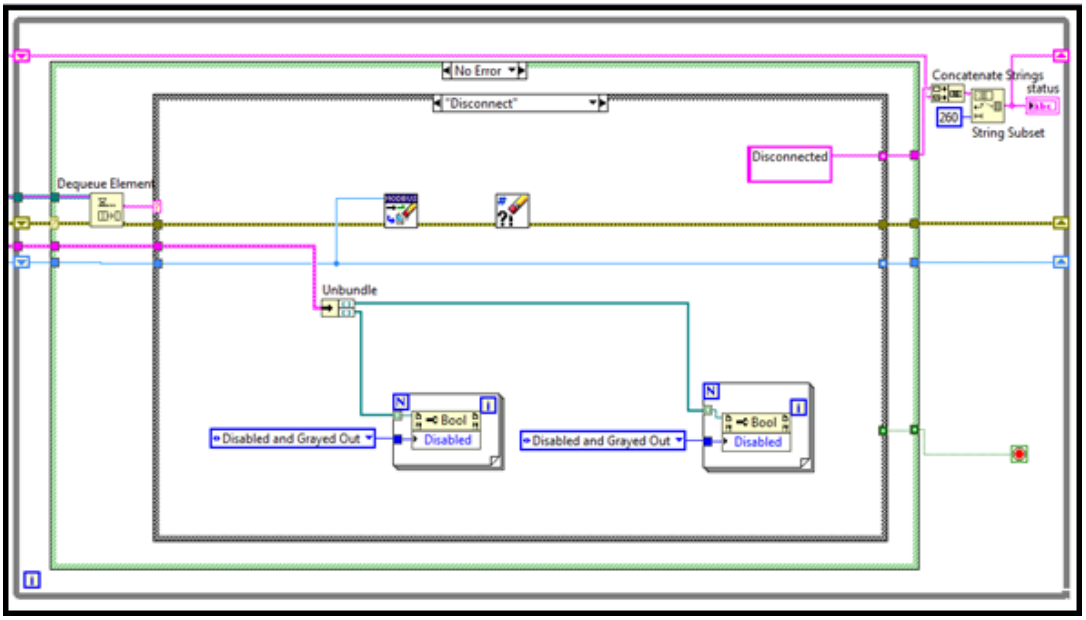


Figure 4.19 bloc diagramme d'état « Disconnect ».

Lorsque on cherche à exécuter un sous-programme dans le programme principal et afin d'empêcher que l'opérateur répète constamment l'exécution de ce dernier pour une tache operateur on crée un état d'initialisation pour le programme comme suit **(Voir figure 4.20)** :

- **« Init »** : dans cette état on a utilisé « Property Nodes » pour désactiver les boutons qui ne doivent être utilisés qu'une fois la connexion maitre-esclave est établie.

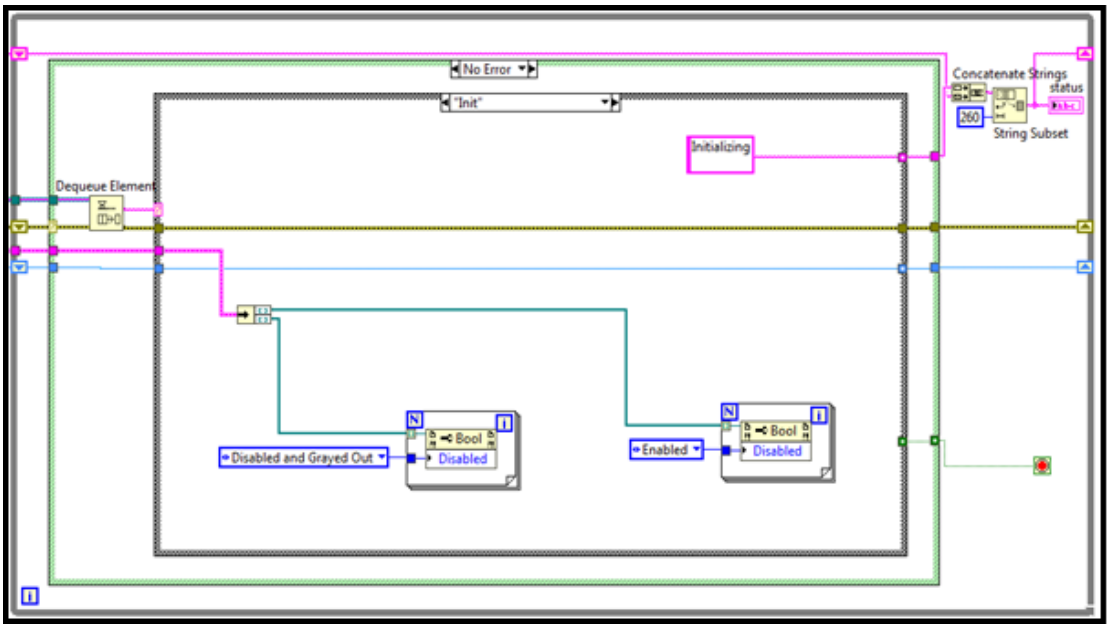


Figure 4.20 bloc diagramme d'état « Init ».

Afin de s'assurer qu'on a connecté l'instrument souhaitée à notre PC et à travers le bus RS-485/USB adaptateur, nous utilisons le port virtuel appelé « VISA resource name » crée par NI pour facilite la connexion. Pour confirmer l'adresse de notre l'instrument

l'instruction « Verify ID » nous permet de savoir son adresse (dans notre cas cette adresse = 01). Cette adresse se trouvant dans le registre (Add Registre = 56) est lue à l'aide d'un programme **subvi** et le résultat est comparé avec l'adresse de l'instrument **(Voir figure 4.21)**.

NB :

- ⇨ S'il est vrai, **true** (machine connecter) on aura un message de connectivité utilisant la fonction **One button dialog.**
- ⇨ S'il est faux**, faulse** (machine non connecter) on aura un message d'erreur avec la fonction **Display message to user**.

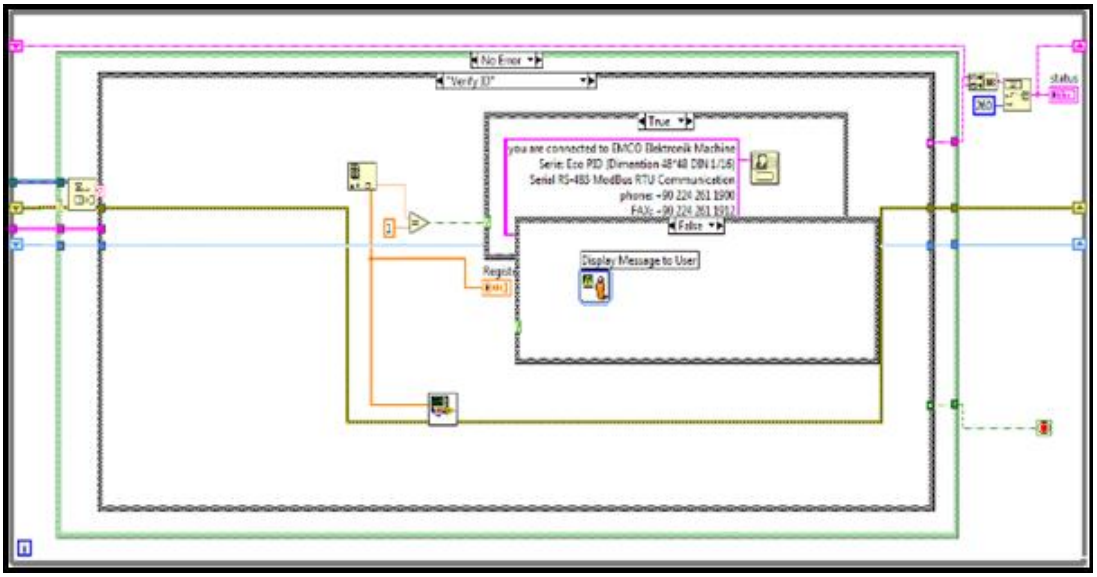


Figure 4.21 bloc diagramme d'état « Verify ID ».

L'opération de lecture est réalisée par l'utilisation des menus disponibles dans Holding Register et l'input Register ou le **« Fetch Holding Register, Fetch Input Register »** permettent de lire tous les paramètres disponibles de l'instrument. **(Voir figures 4.22)**

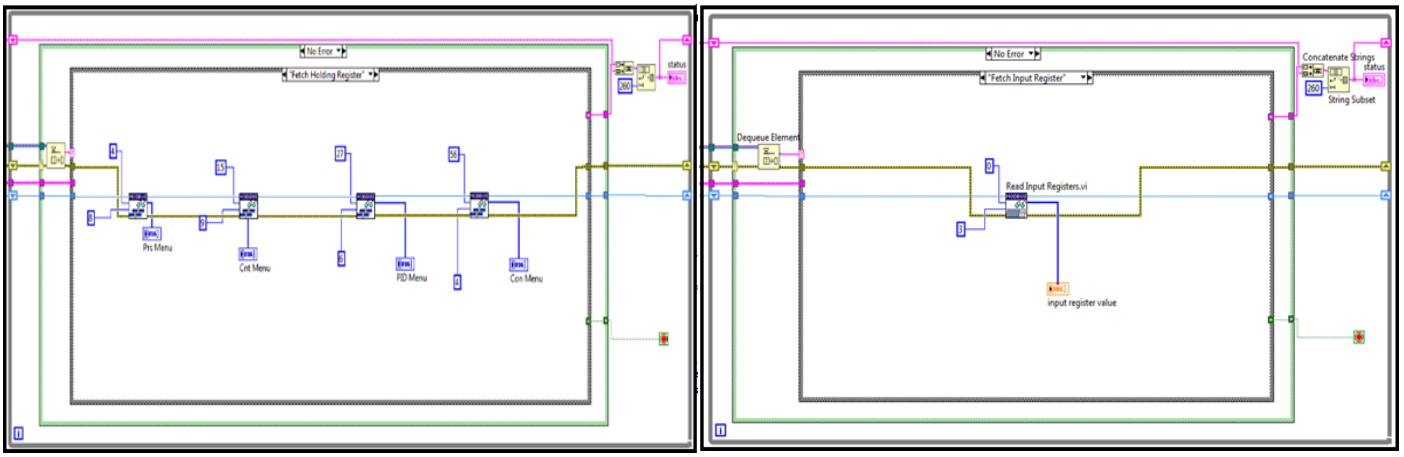


Figure 4.22 bloc diagramme des états « Fetch Holding Register » « Fetch Input Register ».

L'opération d'écriture (write) est réalisée à l'aide des **subvi** sous LabVIEW et permettent à l'opérateur de commander par PC l'instrument régulateur de température. Pour ce faire, les différents menus **(voire tableaux 4.2,3,4,5,6)** jouent un rôle important dans le contrôle de notre appareil. La procédure d'écriture se concrétisé comme suit :

- **« Set Prs Holding Register » :** cette état est destinée à l'écriture des holding register inclus dans le menu « **Process menu »** qui contient les différents paramètres liés au réglage initial de la température à l'aide d'un programme **subvi (Voir figure 4.23)**.

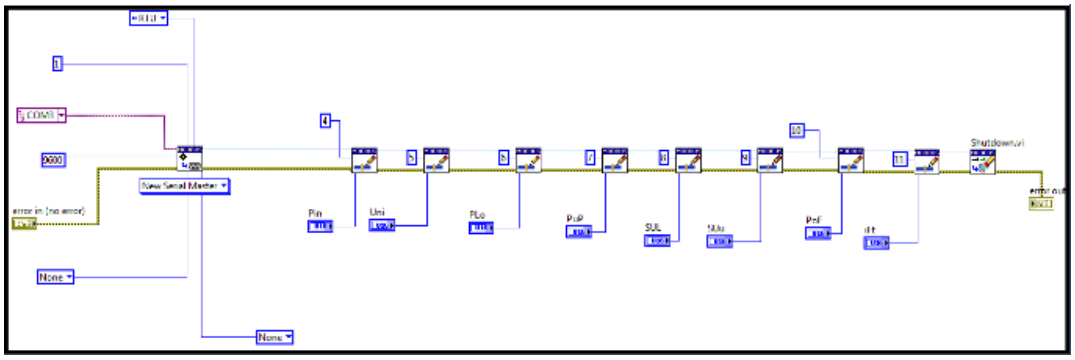


Figure (4.23) bloc diagramme de SubVI utilise dans l'état « Set Prs Holding Register ».

- **« Set Con Holding Register » :** afin d'établir la liaison entre le maitre et l'appareil esclave, cet état accèdera aux paramètres série dans le menus « **con menu » (Voir tableau 4.6)** par un programme **subvi** qui facilite l'écriture de menu Communication. **(Voir figure 4.24)**

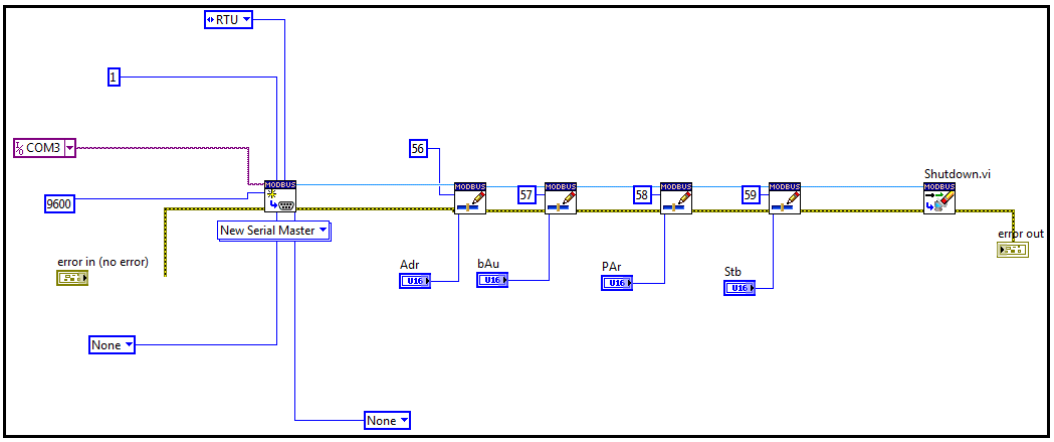


Figure (4.24) bloc diagramme de SubVI utilise dans l'état **« Set Con Holding Register »**.

Pour réaliser l'étape du contrôle PID qui est demandée comme travail dans notre PFE afin de stabiliser la température du régulateur **EMKO eco PID** avec précision, on utilise la fonction **« Set PID Holding Register »** qui permet de changer la méthode de

contrôle des paramètres du PID **(manuel, autotuning or automatic)** de notre appareil esclave à l'aide d'un programme **subvi** qui fait l'écriture de **menu PID**. **(Voir figure 4.25)**

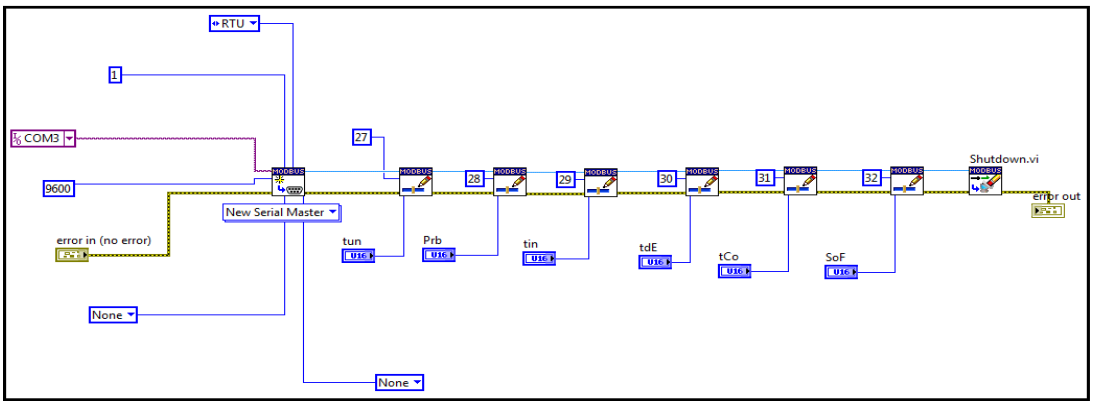


Figure (4.25) bloc diagramme de SubVI utilise dans l'état « Set PID Holding Register ».

Nous connectons toutes les **subvi (4.23,24,25)** mentionnés précédemment, nous obtiendrons notre block diagramme pour le « **read / write** » avec les paramètres fixe dans les tableaux **(4.2,3,4,5),** prévues pour notre instrument **(Voir figure 4.26)** :

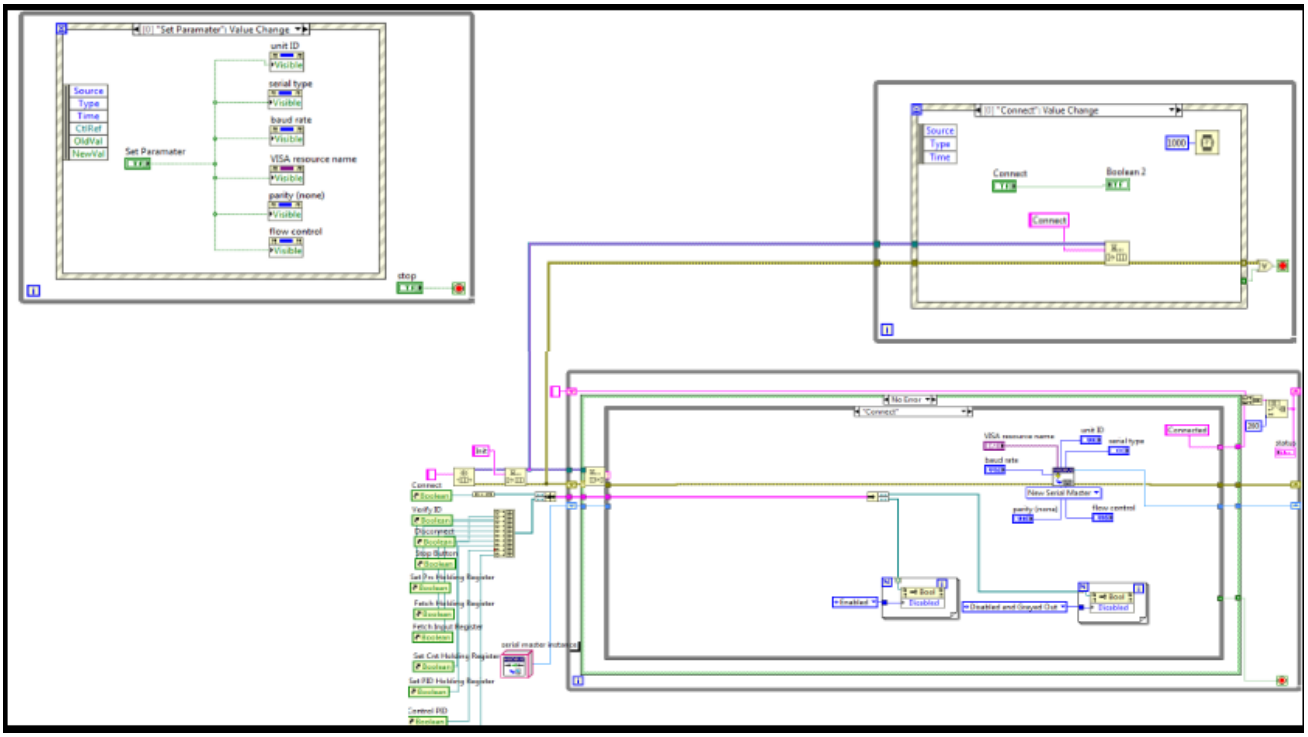


Figure (4.26) block diagramme pour la lecture et l'écriture des registres du régulateur de température **EMKO eco PID.**

Dans le but de vérifie le bon fonctionnement de notre programme on va faire la lecture et l'écriture du quelques paramètres des menus **(voir tableaux 4.2,3,4,5,6)** essentiels dont on a besoin pour contrôler notre régulateur. **(Voir l'annexe 2d)**

Une fois implémentée et exécuté l'algorithme répondant au modèle basée sur les deux boucles référençant chacune une queue commune, nous développons par la suite notre interface exploitable par l'opérateur.

4.7.1 Développement de l'Interface exploitable par l'opérateur :

Afin de réaliser une interface software simple à exploitation et utiliser par les techniciens, nous basons notre travaille sur la procédure technique rapportée dans le data sheet de l'instrument régulateur EMKO eco PID.

Dans la figure qui suive **(Voir figure 4.27)** nous montrons cette interface opérateur réalisée sous LabVIEW :

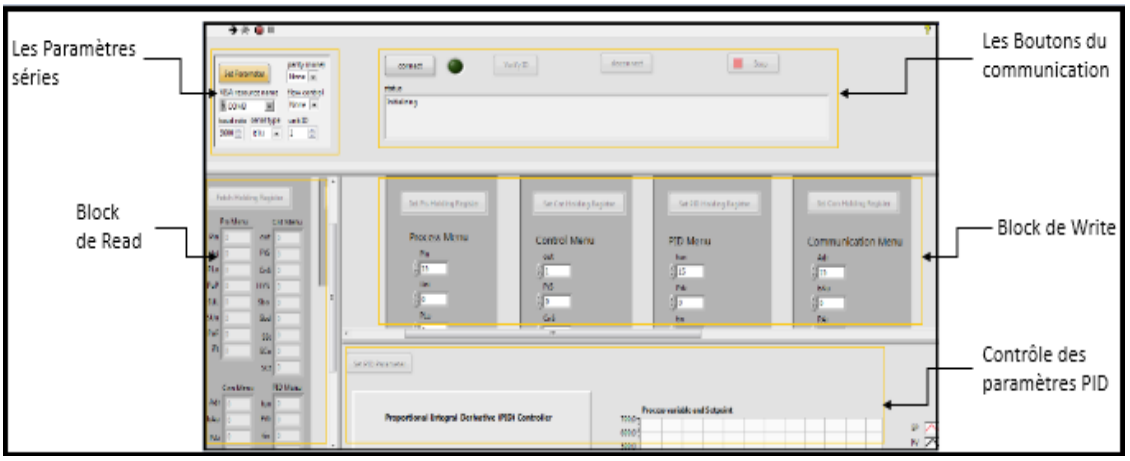


Figure (4.27) interface pour l'opérateur concernant le régulateur de température **EMKO eco PID** développée sous LabVIEW.

4.7.2 Interface du contrôle PID :

Dans cette partie, le but est de :

- Afficher et régler les paramètres essentiaux de régulateur **EMKO eco PID** (P, I et D : Proportionnel-Integral-Derivative) ;
- Observer l'évolution de la valeur mesurée de la température ;
- Contrôler la valeur de la température pour rester proche de la valeur désirer (la valeur de consigne).

L'interface du contrôle PID se fera de la manière suivante :

- **« Control PID » :** qui aidera le choix de réglage du PID (manuel, autotuning,

automatique) et ces paramètres inclus dans le menus « **PID menu »**, et à la fois permet d'examiner la progression de la température mesure (PV) par rapport à la valeur du consigne (SP) a l'aide du .**vi** (**PID Advanced Autotuning.vi) (Voir figure 4.28)** qui facilitera la lecture du la consigne et la valeur mesure.

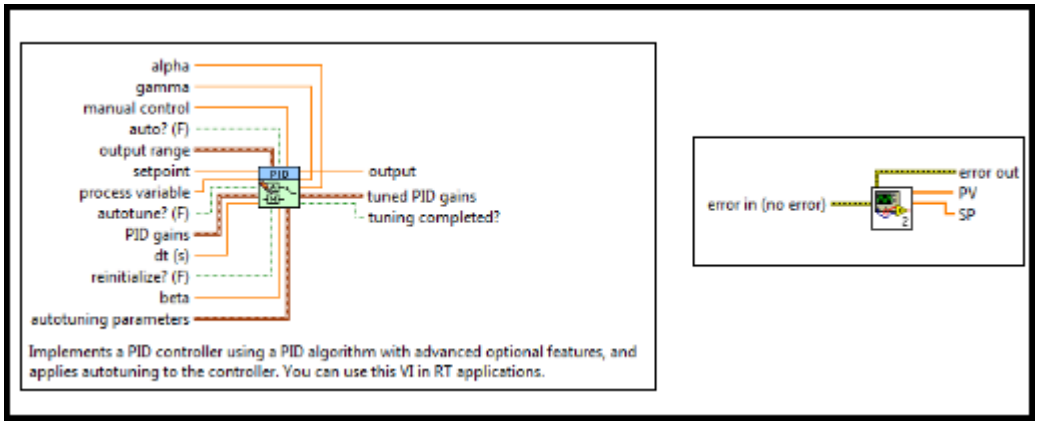


Figure (4.28) les .VI utilises pour l'implémentation sur LabVIEW.

Le block diagramme adopter pour notre instrument EMKO eco PID, utilise Le regroupement de ces deux VI sus' indiqués et la forme de block digramme qui nous permettra de réaliser notre objectif, est rapportée selon la figure suivante :

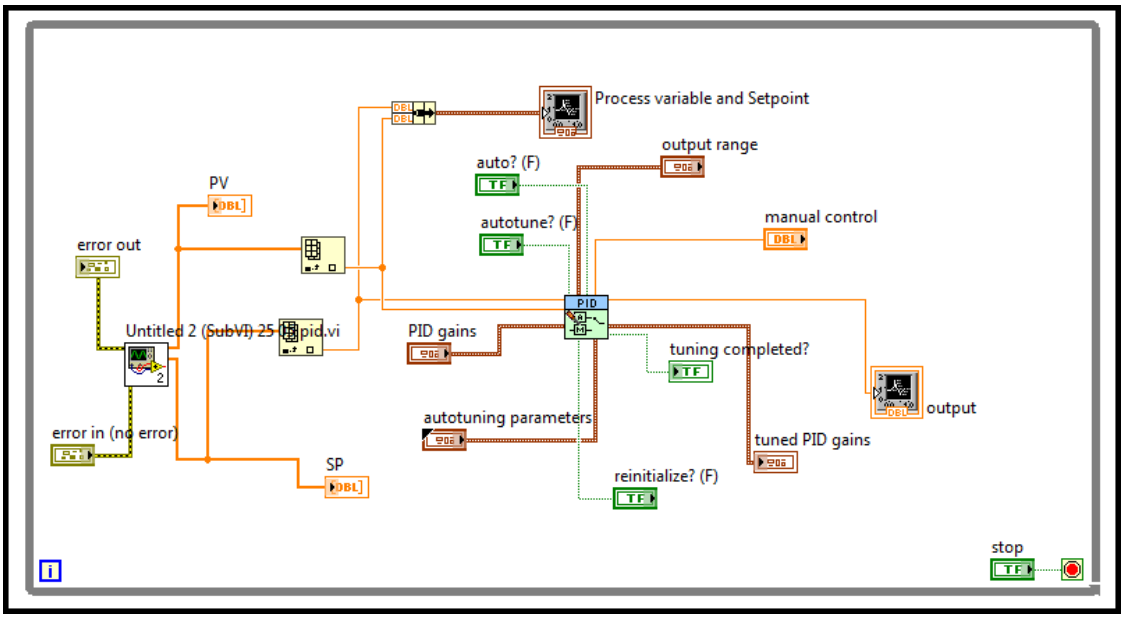


Figure (4.29) Bloc diagramme de contrôle PID de régulateur **EMKO eco PID**.

En ce qui concerne la face-avant du logiciel test implémenter pour le contrôle PID sous logiciel LabVIEW se présente comme suit :

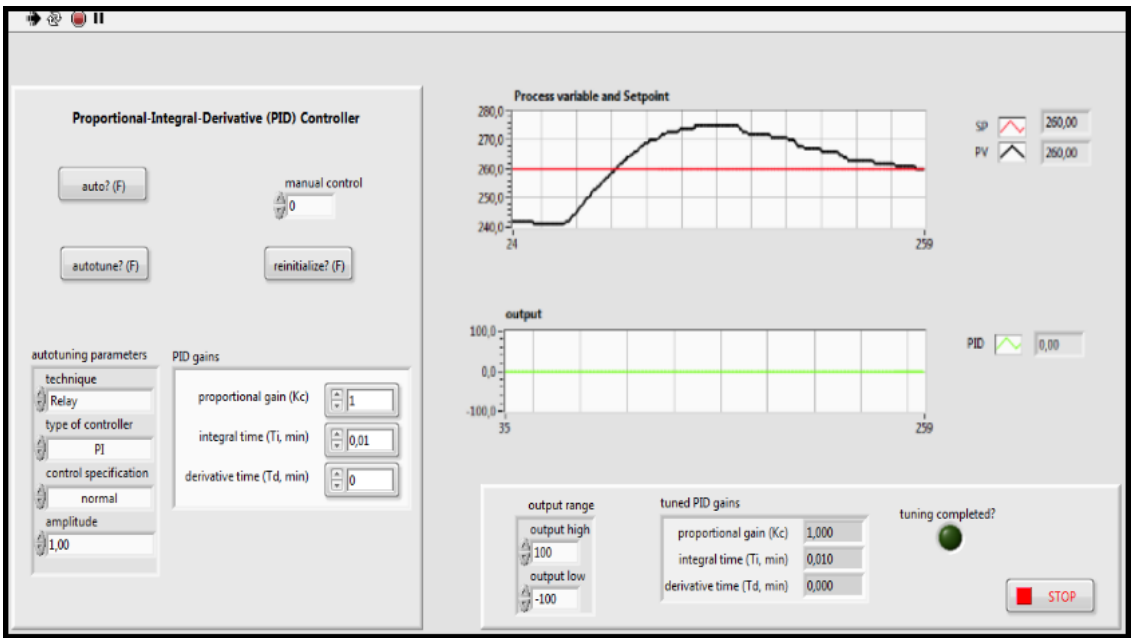


Figure (4.30) les résultats du contrôle PID.

4.8 Conclusion :

Ce dernier chapitre exploite toutes les étapes que nous avons franchies au cours du processus d'acquisition. Nous avons testé et confirmé la connectivité Modbus avec deux simulateurs de test différents pour s'assurer le bon déroulement de la configuration matérielle avant de passer à l'implémentation sur le logiciel LabVIEW. A la suite nous avons exploitées le bon fonctionnement de notre interface d'opérateur par une série des tests du lecteur et l'écriture des différents paramètres essentiels pour le contrôle de notre régulateur de température **EMKO eco PID.** A la fin nous avons développés des interfaces du contrôle d'opérateur globale et particulièrement pour le contrôle de paramètres du PID de régulateur **EMKO eco PID** objet de notre projet de fin d'étude de master 2.

Conclusion générale

En définitive nous pouvons dire que nous avons atteint l'objectif souhaité, qui consiste à la réalisation d'une interface d'opérateur pour le contrôle et le commande d'un régulateur de température via le bus RS-485 Modbus et un PC.

Pour y parvenir nous avons commencé par donné quelques notions et principes sur la régulation aves des généralités concernant le système du contrôle et de régulation industriel tout en insistant sur le régulateur PID. Ce dernier ne pouvant être réalisé d'une manière simple et pratique qu'en l'implémentant sur un calculateur numérique.

Nous avons traité dans ce projet le régulateur de température **EMKO Eco PID**, avec l'étude de ces caractéristiques et son fonctionnement. Ceci nous a permis de voir de près sa flexibilité et sa facilité pour l'acquisition, la transmission de données et la commande.

Pour ce faire, nous avons également testé et confirmé la connectivité Modbus-régulateur avec deux simulateurs de test différents pour s'assurer du bon déroulement de la configuration matérielle. Nous avons mis en œuvre une implémentation de commande et de contrôle de ce régulateur **EMKO eco PID** par l'utilisation du logiciel graphique LabVIEW. Le développement de l'interface de contrôle d'opérateur globale pour les paramètres de PID du régulateur **EMKO eco PID** objet de notre projet de fin d'étude de master 2 est fait aussi sous LabVIEW.

En final, les résultats obtenus répondent au problématique du sujet proposé. Néanmoins, vu le temps alloué pour notre PFE, l'optimisation de l'interface développée et sa mise sous un protocole de tests réels méritent une prise en charge sous la forme d'un projet de fin d'étude master 2 en perspective.

Bibliographie

[1] Antoine Masset, « le meilleur des mondes des technologies : un progrès pour les hommes », 2020.

[2] Enseignements en electronique, « l'automatique les automatismes séquentiels les asservissements », cours génie électrique, 2018.

[3] Mohamed Djamel, Mouss, « Diagnostic et conduite des systèmes de production par approche à base de connaissance », thèse doctorat, laboratoire d'automatique et de productique, Université du Batna, 2006.

[4] Keith Stouffer, Suzanne Lightman, Victoria Pillitteri, Marshall Abrams, Adam Hahn, « Guide to Industrial Control Systems (ICS) Security » **NIST** National Institute of Standards and Technologie U.S. Department of Commerce Special Publication 800-82, Revision 2, May 2014.

[5] Michel Grout, Patrick Salaun, « Spécification et installation des capteurs et vannes de régulation », chapitre 1 pp 23-30, 4^{em} Édition DUNOD 2015.

[6] Lionnel Ponchon, « pilotage du procède MES SCADA », Lyon France, 20 décembre 2018.

[7] Hubert Faigner, « Structure général d'un système automatisé », Support cours asservis BTS MI, chapitre pp 1-27, 5 mai 2021.

[8] Amir Mohamed, « Introduction aux systèmes asservis mode de compatibilité » Support de cours, pp 1-20.

[9] Bouakkaz Messaoud, « Régulation industriel » Support de cours, CRI Electromécanique, chapitre pp 1-25, Elearning univ-annaba, 5 avril 2020.

[10] Hichem ZAYANI, « Support de cours : Régulation et contrôle des systèmes de climatisation » Support de cours, chapitre 3 pp 30-66, l'ISET de Sfax A.U, 2015.

[11] Melah.R, « développement d'une plat forme de régulation du niveau on-line de la station UCP-EDIBON sous environnement LabVIEW » mémoire master, Université Mouloud Mammeri, Tizi-Ouzou, 2011.

[12] Marc Correvon, « Les régulateurs standards » Support cours, chapitre 7 pp 57-95, Haute Ecole Spécialisée de Suisse Occidentale, 25 septembre 2007.

[13] Hechmi Khaterchi, « boucles de régulation » Support cours, automatisme et informatique industrielle, chapitre 8 77-107, 22 novembre 2017.

[14] Gokcen Bircan Degerliyurt, « ENG-EcoPID » Data sheet de regulateur EMKO eco PID, pdf, pp 1-2, Bursa tukiye 2019.

[15] Ali lafioune Sid, « Etude et réalisation d'un banc de démonstration pour la supervision industrielle en utilisant le protocole Modbus RTU », mémoire de master 2, Université de Mohamed Seddik Ben Yahia, jijel,2019.

[16] Terry King, Jun Peng, Your Duino, « USB TO TTL/RS-485 Interface », datasheet, pp 1, USA 2021.

[17] Weis Olga, « Virtual Serial Port », *Eltima publishing*, United Arabe Emirates, ''Consulté le mai 18, 2021 '', 23 octobre 2020.

[18] Cablematic, datasheet, « adaptateur-série », Espagne, aout 2020.

[19] NBS System, « transmission en série vs transmission en parallèle »,USA , aout 2020.

[20] Develle, Ulysse, « protocole de communication industrielle » Article de programmation industriel, pp 1-3, 26 mai 2020.

[21] National Instrument, « LabVIEW Modbus API » publication, USA 04 aout,2014.

[22] Ahmed Hanafi, « initiation à la programmation graphique » Support de cours, élément de module 5.3 pp 2-5, Université sidi mohamed ben abdellah de fess, Maroc 2019.

[23] Zaidi Mouloud, Maouche Med Amine, « Développement d'une interface pour capteur CND à courants de Foucault », mémoire de master 2, université mouloud mammeri, tizi-ouzzou, 2013.

[24] FREELANCE « en Système Industriel et Scientifique LabVIEW, TestStand, LabWindows/CVI », école national des ingénieurs, France, 2019.

[25] Jerome jovita, « Virtual instrumentation using LABVIEW », estern economy edition 110001, new Delhi India 2010.

[26] Sakli MOUADH, « commande et régulation de niveau d'eau avec LabVIEW » mémoire master 2, École Nationale d'Ingénieurs de GABÈS, TUNISIE,2007.

[27] The Interface Solution Experts publication, « Using MODBUS for Process Control and Automation », PDF, pp 1-2, USA, 2011.

[28] Ed Doering, National Instrument, « Producer-Consumer-state-machine », article, Rose-Hulman Institue of Technology, 21 january 2018.

Annexe 1a

a. Lecture de paramètre PrB (proportional band) du menu PID :

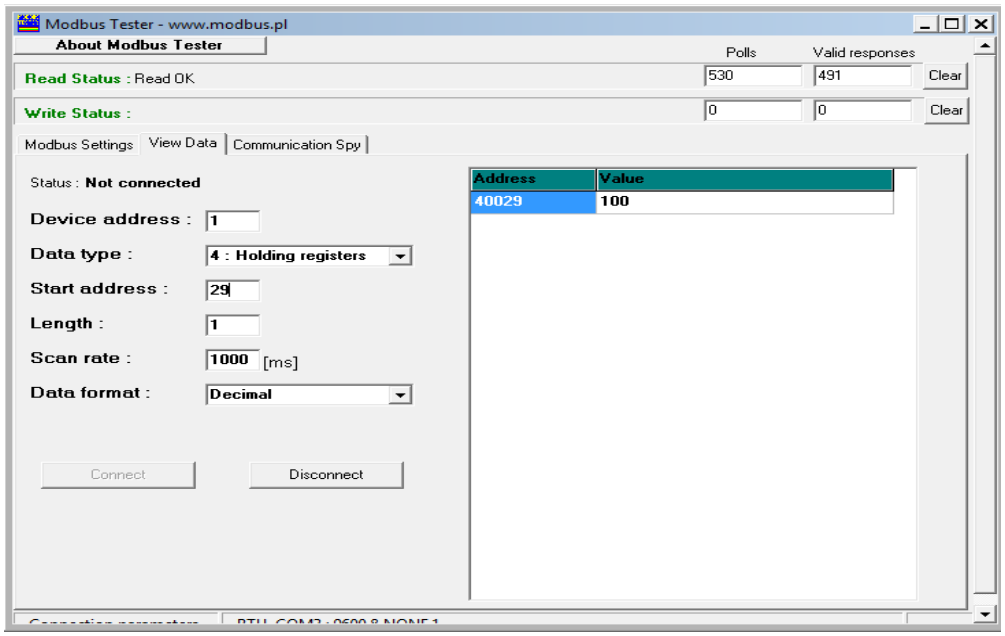


Figure a1.1 Résultat de lecture du registre PrB de menu PID sur Modbus tester.

b. Lecture de paramètre Uni (unit) du menu Prs :

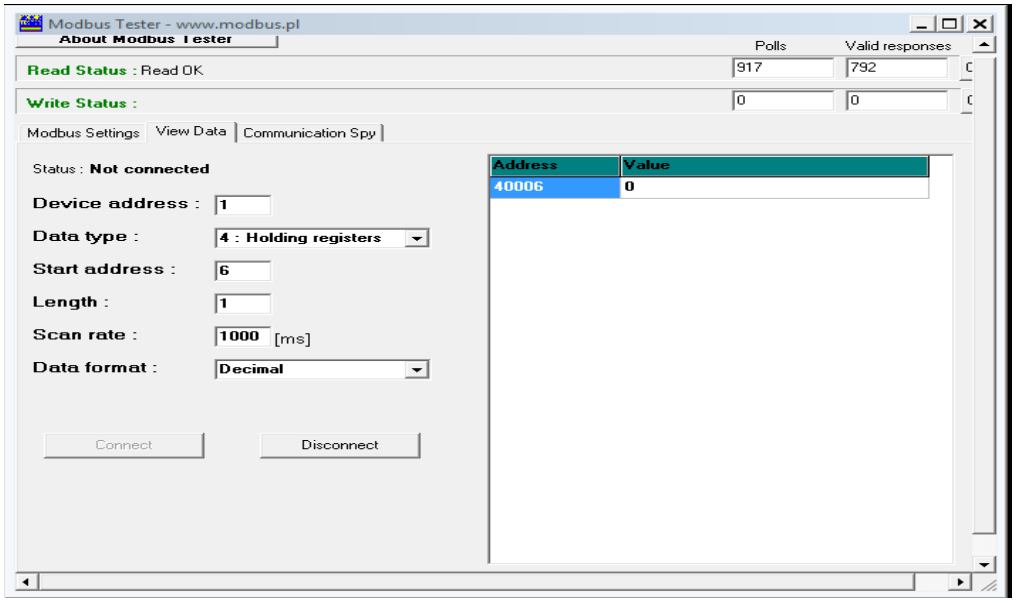


Figure (1a.2) Résultat de lecture du registre Uni de menu PRS sur Modbus tester.

Annexe 1b

a. Lecteur du paramètre de menu Prs :

Réglages du paramètre série :

- Port : COM3
- Baud rate : 9600
- Parity bit : None
- Communication wiring : wiring with no echo 4 wire
- Sample mode : Manual
- Data type : Holding Register R03/W16
- Slave ID : 1
- Starting Register : 5
- Number of Registers : 8
- Display Mode : Decimial
- Protocol : Modbus

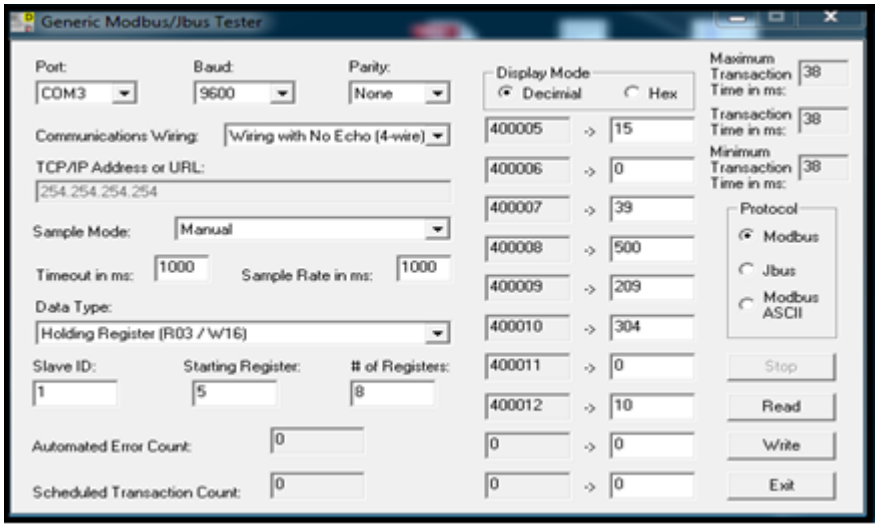


Figure (1b.1) Résultat de lecture de menu Prs sur Modbus/Jbus.

b. L'écriture du paramètre PuP de menu Prs :

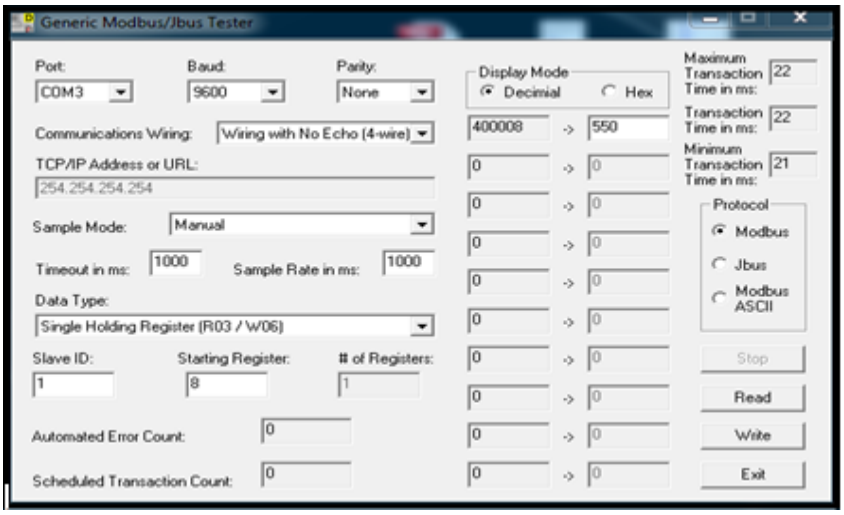


Figure 1b.2 Résultat de l'écriture du registre PUP de menu PRS sur Modbus/Jbus.

Figure 1b.3 Résultat de l'ecriture du registre PuP de menu PrS dans le régulateur.

Annexe 2a

a. **Lecture du registre PrB (proportional band) de menu PID :**

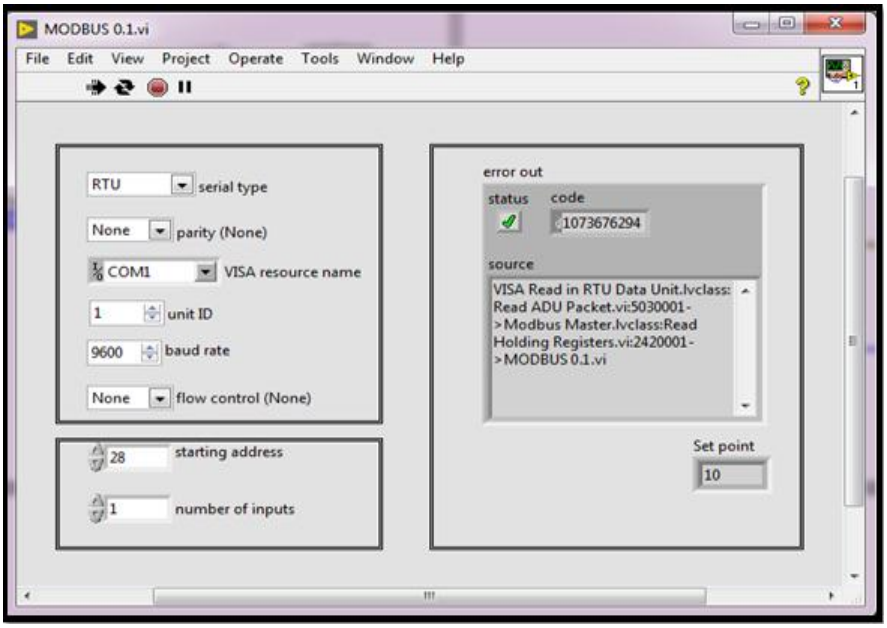


Figure (2a.1) résultat de la lecture du registre PrB sur la face-avant du logiciel LabVIEW.

b. **Lecture du registre PrS (process type selection) de menu Cnt :**

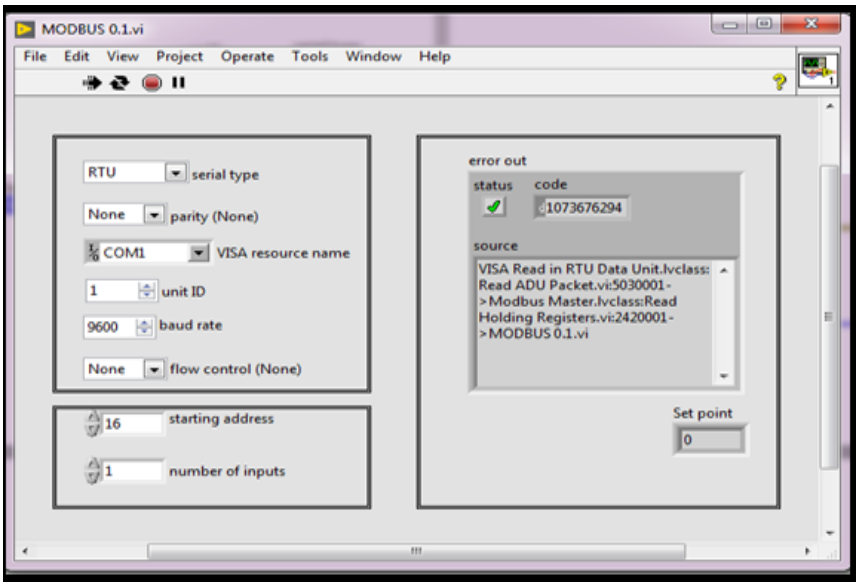


Figure (2a.2) résultat de la lecture du registre PrS sur la face-avant du logiciel LabVIEW.

c. **Lecture du registre Par (parity selection) de menu Con :**

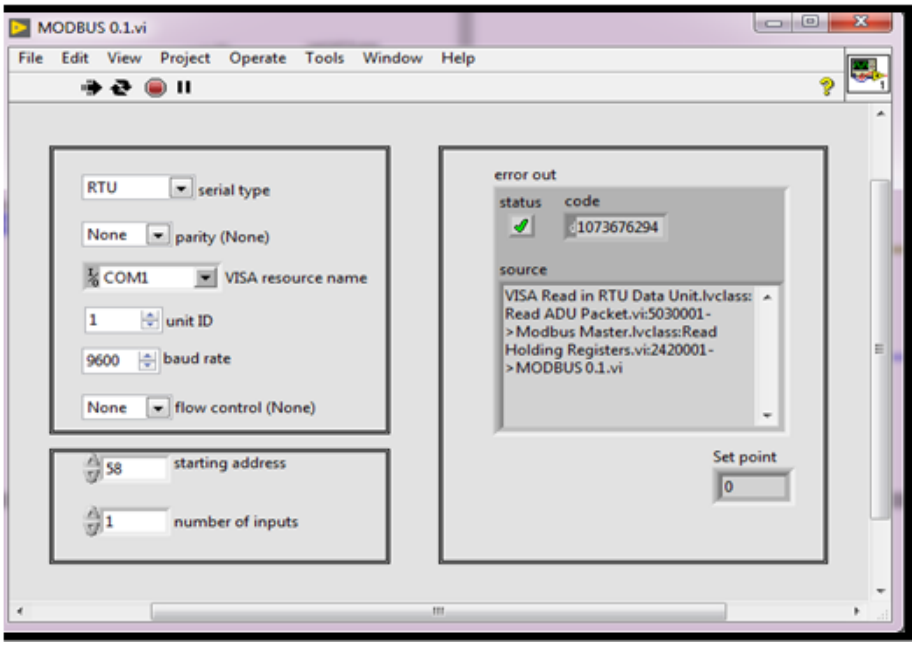


Figure (2a.3) résultat de la lecture du registre PAr sur la face-avant du logiciel LabVIEW.

d. **Lecture du registre tdE(derivative time) de menu PID :**

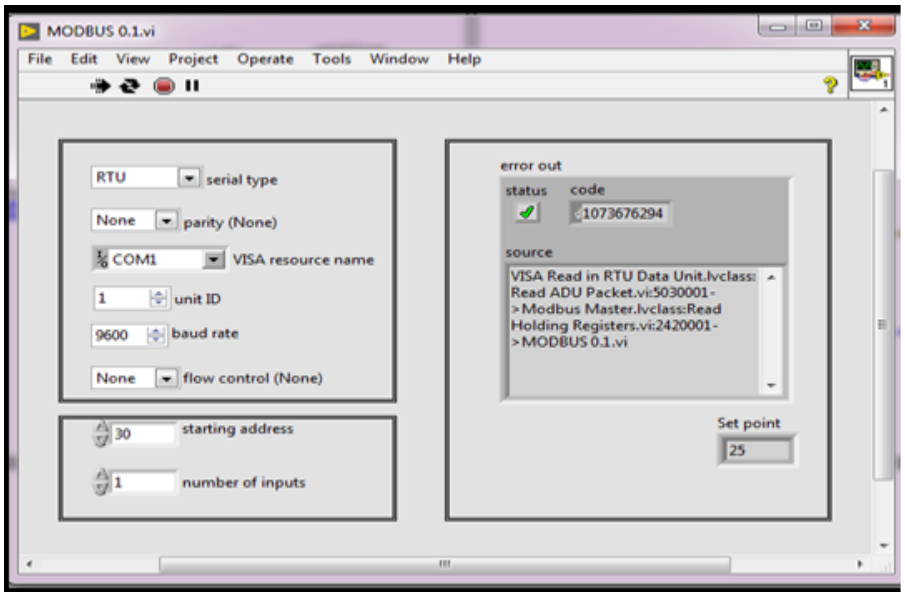


Figure (2a.4) résultat de la lecture du registre tdE sur la face-avant du logiciel LabVIEW.

Annexe 2b

a. L'écriture du registre de PrB :

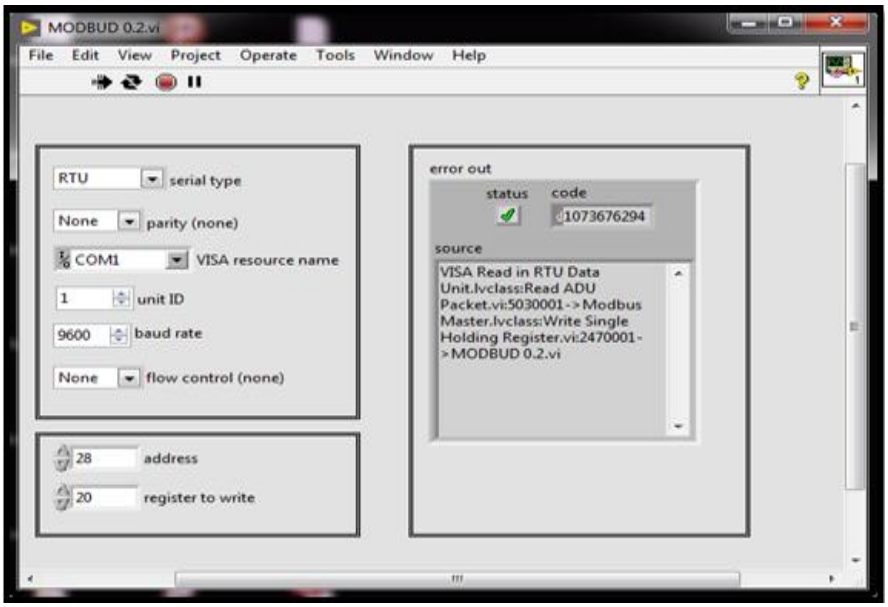


Figure (1b.1) résultat de l'écriture de paramètre PrB sur la face-avant du logiciel LabVIEW.

b. L'écriture du registre de tdE :

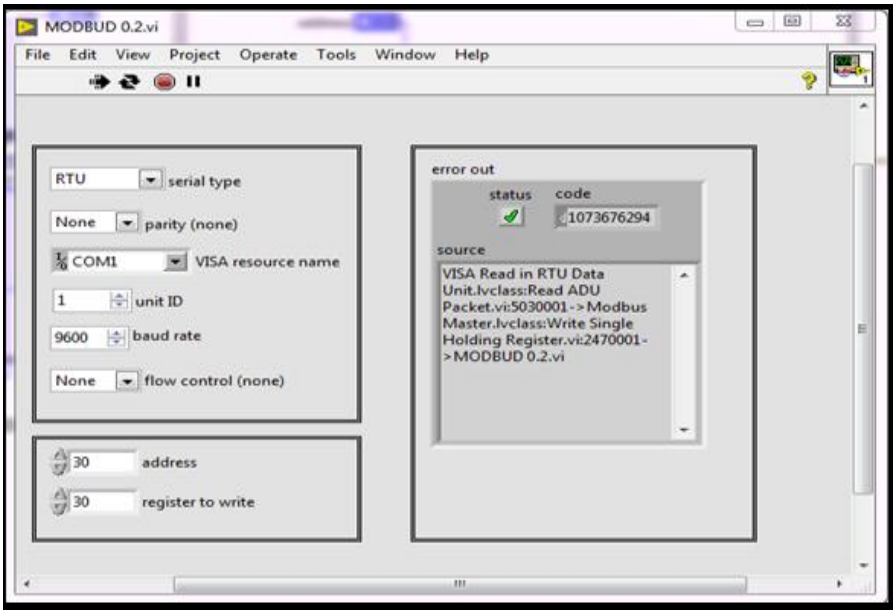


Figure (1b.2) résultat de l'écriture de paramètre TDE sur la face-avant du logiciel LabVIEW.

Annexe 2c

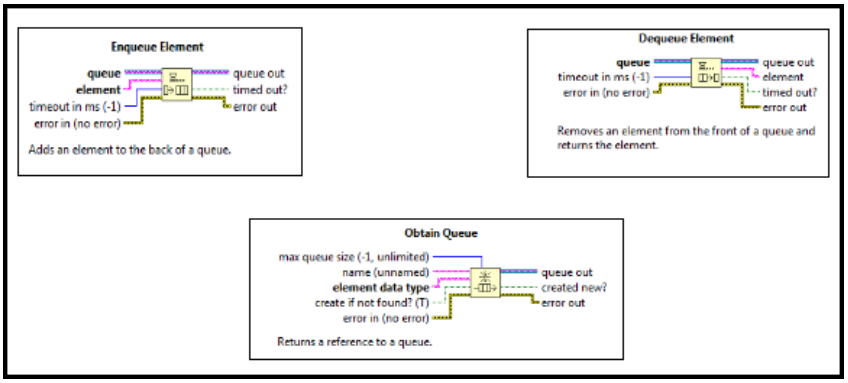


Figure (2c.1) les fonctions de la palettent « Queue operation subpalet ».

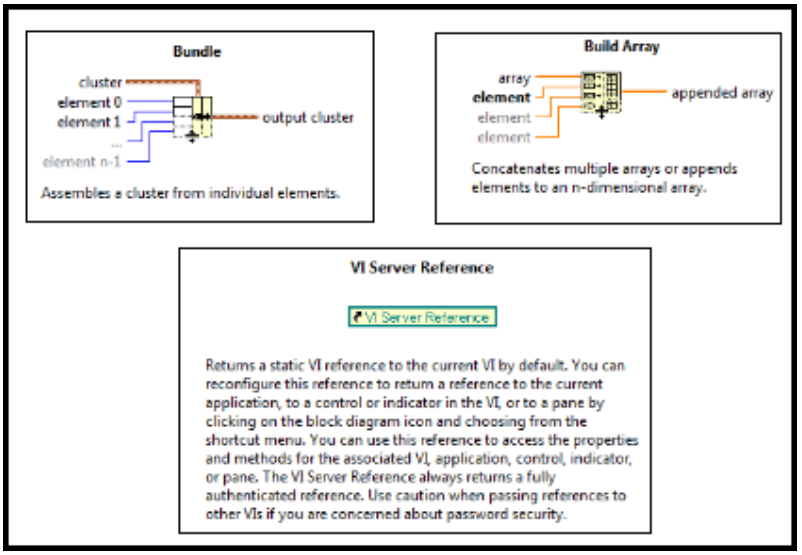


Figure (2c.2) les éléments utilisées dans la partie consommateur.

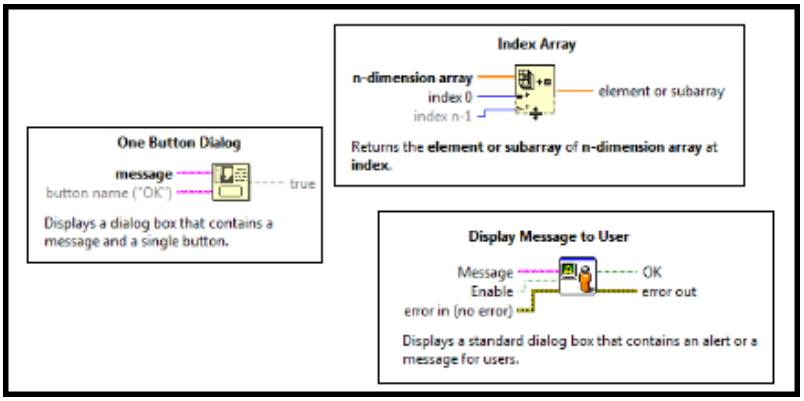


Figure (2c.3) les fonctions du palette « Dialog And User Interface »

Annexe 2d

➢ **Démarche pour la lecture des paramètres dans les menus du Holding et Input registres :**

- Réglage des paramètres série :
- VISA ressource name : COM8.
- Baude rate : 9600.
- Seriel type : RTU.
- Parity(non) : Non.
- Flux control : Non.
- Unit ID : 1.

Figure (2d.1) résultats du lecteur des Registres sur la face avant du logiciel LabVIEW.

- **Démarche pour l'écriture des paramètres dans les menus du Holding et Input registres :**

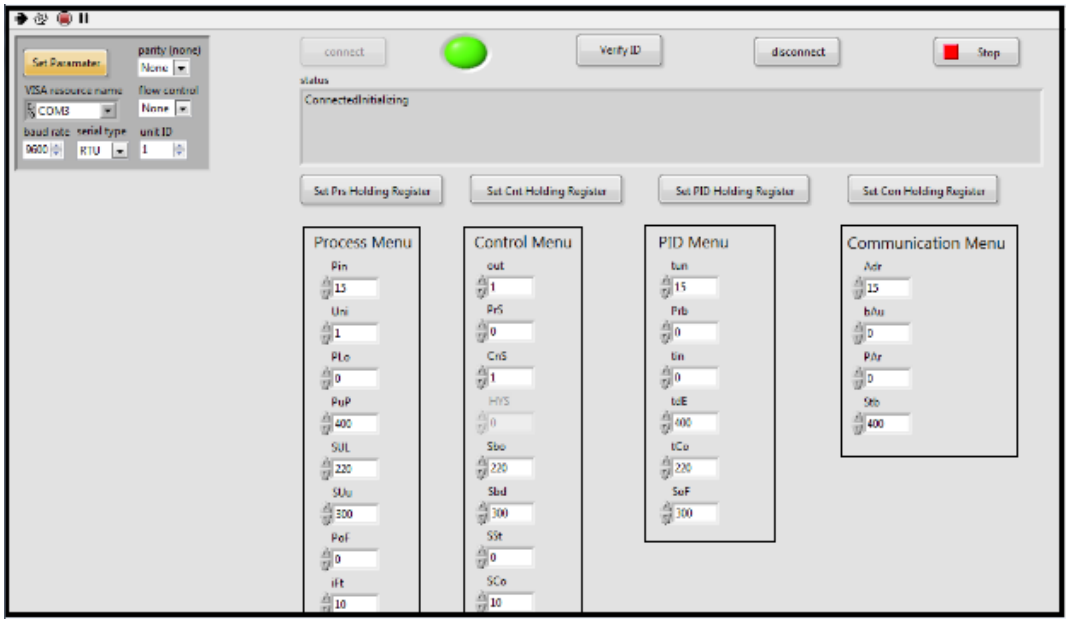


Figure (2d.2) les résultats de l'écriture des Registres sur la face avant du logiciel LabVIEW.