الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Département : Génie Electrique et Informatique Industrielle

Mémoire de projet de fin d'études Pour l'obtention du diplôme

D'ingénieur d'état en

Automatique

Spécialité
Automatique et informatique industrielle

Développement d'une passerelle de communication industrielle modbus-RTU / modbus TCP

Réalisé par

TRIKI Mohammed Hichem **ATBA Bilal**

Compositiondu Jury:

Président	M. A. ZAOUI A	Professeur	ENST
Promoteur	M. CHERIFI Tarek	M.C.B	ENST
Examinateur	Mme. OUSAADI Zahira	M.A.A	ENST
Examinateur	Mme.BENDOUDA Djamila	M.C.B	ENST

Année universitaire 2021-2022

REMERCIEMENT

En premier lieu, A Dieu le Misércordieux qui a guidé nos pas vers un avenir Incha'allah prometteur, où le travail, la persévérance et la quête du savoir seront notre devise, et qui nous a donné la force pour accomplir ce travail.

Nous adressons nos remerciements à nos chers parents, sœurs et frères qu'on aime tant et sans lesquels on ne serait jamais arrivé là où on est

Nous tenons à notifier un remerciement spécial

Pour notre encadrant Mr. CHERIFI pour sa disponibilité, son aide et pour ces conseils.

Nous exprimons nos profonds remerciements aux membres de jury pour leurs participations dans le jugement de notre travail.

Nous présentons nos chaleureux remerciements à tous nos amis spécialement Hadjab Chihab Eddine pour ces services on programmation

Mr. Fouzi bouhaouch, Mr. Nabil Laib et Mr. ramdane

Qui nous en a compagne dans le monde professionnel de l'ingénieur

Table des matières :

Remerciement ……i

Table des figures ……ii

Chapitre 01

I.1 Introduction 3

I.2 supervision industrielle 3

I.2.1 Définition 3

I.2.2 Techniques de la supervision 3

I.2.2.1 Acquisition de données 3

I.2.2.2 Surveillance 4

I.2.2.3 Diagnostic 5

I.2.2.4 Détection 5

I.2.3 La Supervision dans l'industrie(SCADA) 5

I.2.3.1 Architecture du SCADA 5

I.2.3.2 Interfaces graphiques du SCADA 6

I.2.3.3 Exemples d'utilisation industrielle de SCADA 7

I.3 Système de contrôle industriel distribué 7

I.4 Système Experion PKS 8

I.4.1 L'architecture du systèmeexpirion pks 9

I.4.2 Serveur PKS 10

I.4.3 La Station 11

I.4.4 Options de sécurité 11

I.4.5 FTE (Fault Tolerant Ethernet) 12

I.4.6 Concepts et fonctions 13

I.4.6.1 Propriété globale des données 13

I.4.6.2 Redondance 13

I.4.6.3 Prise en charge des équipements de télé conduite SCADA 13
I.4.6.4 Configuration Studio 14
I.4.7 Matériels de contrôle-commande série C 15
I.4.7.1 Description de la série C 15
I.4.7.2 Les modules d'entrées / sorties série C 16
I.4.7.3 Le pare-feu Control Firewall : 16
I.4.8 Système de sécurité (ESD) Safety Manager 17
I.4.8.1 Contrôleur 17
I.4.8.2 Processeur de Contrôle 18
I.4.8.3 les modules Entrés sorties 18
I.4.8.4 Bus d'entrées sorties 18
I.4.9 Intégration dans l'Experion PKS 18
I.4.9.1 Communication PCDI 19
I.5 Conclusion : 19

Chapitre 02

II.1 Introduction 22
II.2 Protocole de communication industrie 22
II.2.1 Définition d'un protocole de communication : 22
II.2.1.1 Couche physique 22
II.2.1.2 Couche de contrôle 22
II.2.1.3 Couche de supervision 23
II.2.1.4 Couche d'exécution et de planification 23
II.2.2 Fonctionnalités des protocoles de communication industrielle 23
II.2.2.1 Couche 1 (physique) 23
II.2.2.2 Couche 2 (liaison de données) 23

II.2.2.3 Couche 3 (réseau) 23

II.2.2.4 Couche 4 (Transport) 23

II.2.2.5 Couche 5 (session) 23

II.2.2.6 Couche 6 (Présentation) 23

II.2.2.7 Couche 7 (Application) 24

II.2.3 Les caractéristiques des protocoles de communication industrielle 24

II.2.4 Méthodes de communication 24

II.2.4.1 Maître/esclave 24

II.2.4.2 Anneau à jeton 25

II.2.4.3 Pair-à-pair (Peer-to-Peer : P2P) 26

II.3 Les types de protocoles de communication industriels 26

II.3.1 Les protocoles des communications industrielles ouvertes : 26

II.3.1.1 Protocole PROFINET 26

II.3.1.2 Protocole PROFIBUS 27

II.3.1.2.1.Les types de protocoles PROFIBUS 27

II.3.1.2.2.PROFIBUS DP (Périphériques décentralisés) 27

II.3.1.2.3.PROFIBUS PA (automatisation des processus) 27

II.3.1.3 Protocole AS-1 28

II.3.1.4 Protocole HART 28

II.3.1.6 Protocole du Réseau de contrôleurs (Controller Area Network (CAN)) 29

II.3.1.7 Protocole Modbus 30

II.4 Conclusion 40

Chapitre 03

III.1 Introduction 41
III.2 Approche proposée : 41
III.3 Présentation du matériel utilisé : 42

III.3.1 La Raspberry Pi : 42

III.3.2 Vnc viewer : 43

III.4 Présentation des langages de programmation utilisés 44

III.4.1 Python : 44

III.4.2 Tkinter : 44

III.5 Réalisation : 45

III.5.1 Réalisation d'un chat room : 46

III.5.2 Réalisation de passerelle 48

III.5.2.1 Premier phase : 48

III.5.2.2 Deuxième phase : 50

III.5.2.3 Troisième phase : 50

III.6 Résultat de simulation : 51

III.6.1 test de requêtes TCP: 52

III.6.2 Elément des trames: 52

III.6.3 réponse : 54

III.7 CONCLUSION : 54
Conclusion générale …….……..55

Liste des figures :

Figure I.1 : Différents type de surveillance..... 4

Figure I.2 : Architecture de la supervision dans un environnement SCADA..... 6

Figure I.3 : Interface graphique de supervision d'une installation de traitement d'eau..... 7

Figure I.4 : Architecture de DCS Experion PKS. 10

Figure I.5 : Intégration du serveur dans une architecture PKS. 11

Figure I.6 : Intégration du Fault Tolerant Ethernet dans une architecture PKS..... 12

Figure I.7 : FTE pour une connexion réseau double.. 13

Figure I.8 : Connexion SCADA en mode série avec le serveur Experion. 14

Figure I.9 : Contrôleur C300. 15

Figure I.10 : Série C modules d'entrées sorties redondants sur une IOTA..... 16

Figure I.11 : Module control firewall et topologie de série C..... 17

Figure I.12 : Les différentes connexions du Safety Manager 18

Figure I.13 : Connexion du SM avec l'architecture PKS via l'USI. 19

Figure II.1 : Méthode de communication Maître/esclave..... 23

Figure II.2 :Méthodes de communication Anneau à jeton..... 24

Figure II.3 :Méthodes de communication Pair-à-pair..... 25

Figure II.4 :système de moteur contrôlé par PLC qui n'utilise pas Modbus..... 30

Figure II.5 :système de moteur contrôlé par PLC qui utilise Modbus..... 31

Figure II.6: Trame de Modbus RTU/ASCII 32

Figure II.7 :liste des codes de fonction Modbus les plus utilisés..... 33

Figure II.8 :Les plages d'adresses absolues..... 33

Figure II.9 : Trame de requête et réponse de fonction 01 34

Figure II.10 :Trame de requête et réponse de fonction 02 35

Figure II.11 :Trame de requête et réponse de fonction 03 35

Figure II.12 :Trame de requête et réponse de fonction 04 36

Figure II.13 :Trame de requête et réponse de fonction 05 37

Figure II.14 :Trame de requête et réponse de fonction 06 37

Figure II.15 :Trame de requête et réponse de fonction 15 38

Figure II.16 :Trame de requête et réponse de fonction 16 38

Figure III.1: Schéma de conception générale de la passerelle. 41

Figure III.2 : Les composants de la carte Raspberry Pi. 43

Figure III.3 : connexion et affichage de la raspberry pi avec vnc viewer 43

Figure III.4 : mode de fonctionnement. 45

Figure III.5 : connexion de client au serveur. 47

Figure III.6 : interface graphique de client crée par TKinter. 47

Figure III.7 :chat room entre 3 clients. 48

Figure III.8 : sys sous Modbus TCP. 49

Figure III.9: Trame Modbus TCP 49

Figure III.10 : mode de fonctionnement de broadcast. 50

Figure III.11 : la différence entre la trame TCP et RTU 51

Figure III.12 :organigramme de l'étape 2 51

Figure III.13 :test 1 de conversion de TCP 52

Figure III.14 :description de mbap header 53

Figure III.15 :conversion de RTU. 54

Introduction générale

Avec le développement des automatismes et la complexité des processus, la nécessité de développer des systèmes de supervision fiable et efficace à augmenter afin de permettre aux opérateurs de savoirs l'état des processus et d'interagir rapidement en situations anormales.

La supervision va permettre de surveiller, en termes de **disponibilité** et de **performance de** l'ensemble de l'infrastructure et applications du système d'information d'une entreprise. Elle assure la surveillance du bon fonctionnement des du système.

Elle permet de surveiller, rapporter et alerter en cas de fonctionnement anormal d'un élément du parc informatique.

Face à ces besoins, l'utilisation de nouvelle technologie s'impose et cela afin de battre la complexité et d'adapter ces systèmes aux différentes exigences. Les systèmes de contrôles industriels qui commandent les infrastructures critiques les plus utilisées sont devenus obsolètes. Les pannes qui surviennent sur ces systèmes causent de fortes perturbations et leur maintenance est devenue très chères à cause du manque des pièces de rechange, ce qui engendre de graves pertes aux entreprises en termes d'argent et de temps

Les Technologies de l'information et de la communication (TIC) sont présentes dans trois principaux domaines : les secteurs qui produisent des biens, les secteurs des télécommunications et le secteur électroniqueCes technologies, en constante évolution, sont au cœur de l'activité industrielle.

Les réseaux industriels apportent une grande souplesse aux systèmes de contrôle / commande, ils diminuent les coûts de câblage. Ils offrent des possibilités nouvelles pour le contrôle et la supervision

Modbus est l'un des protocoles de communication de données les plus utilisé dans le secteur de la fabrication industrielle il contient plusieurs variantes, les plus connus sont le Modbus RTU (ancienne technologie) et TCP (nouvelle technologie).

Chapitre 01 :

Introduction aux systèmes de contrôle de l'industrie

Introduction

Le contrôle de processus industriel permet de surveiller et réguler les machines, les installations et les processus dans un grand nombre de secteur industriels Les processus industriels de traitement, de transformation et de fabrication sont aujourd'hui fortement automatisés afin de gérer les matières premières et l'énergie en toute sécurité, efficacement et sans gaspiller les ressources. Pour cela, les systèmes de supervision industrielle permettent de piloter en temps réel et à distance les procédés de production et assurer l'acheminement des informations sur l'état du système jusqu'au poste de supervision à chaque instant et sans interruption. Cela signifie que les organes de commande transmettent au poste de supervision les informations qu'ils recueillent des capteurs pour générer les commandes.

Dans notre cas, nous avons vudans l'industrie le système de contrôle DCS

Un DCS est une combinaison du concept d'une boucle simple de contrôle locale, les réseaux informatiques et de plusieurs types de systèmes de contrôle et d'instrumentation associée utilisés pour le contrôle des processus industriels.

Dans ce chapitre, nous allons d'abord faire le point sur la supervision en général, ces techniques ainsi que son utilisation dans l'industrie. Ensuite nous allons fait des points sur le système de contrôle ainsi que de leurs efficacités dans l'industrie. Enfin, nous allons détaillerlesystème de control « experion pks » ainsi que leurs éléments et ses variables, ses types de communication et ses utilisations.

I.1 Supervision industrielle

I.1.1 Définition

La supervision est un Ensemble d'outils et de méthodes utilisées pour faire fonctionner un processus industriel en situation normale, ainsi qu'en présence de défaillances elle consiste à conduire une installation industrielle aux moyens d'écrans de supervision placés aux postes de pilotage, rafraîchis à chaque instant par les informations provenant des automatismes et des capteurs intelligents.

La supervision est d'un niveau supérieur et qui superpose la boucle de commande, elle assure les conditions d'opérations pour lesquelles les algorithmes d'estimation et de commande ont été conçus. Parmi les taches principales de la supervision se trouvent l'acquisition de données, la surveillance, le diagnostic et la détection [01].

I.1.2 Techniques de la supervision

I.1.2.1 Acquisition de données

L'acquisition de données est la première étape de la supervision, telle qu'elle consiste à recueillir, à valider et à assurer l'acheminement des informations sur l'état du système jusqu'au poste de pilotage, cette tâche, est exécutée sans interruption et à chaque instant, ces opérations impliquent l'utilisation des capteurs permettant de mesurer les différentes variables du processus. Il peut permettre l'enregistrement automatique des données et fournir un emplacement centralisé pour leur stockage. En outre, il peut

transférer des données à d'autres systèmes Ces informations seront utilisées dans des relations de résidus pour accomplir l'étape de détection [02].

I.1.2.2 Surveillance

La surveillance utilise les données provenant du système pour représenter l'état de fonctionnement puis détecter les évolutions. Elle sert à filtrer les signaux et les événements issus du procédé et de la commande afin d'établir l'état du système. En fonctionnement normal, elle communique des comptes rendus filtrés à la commande. Elle permet également de détecter et diagnostiquer les fautes et les erreurs dans le système. En cas de dysfonctionnement, elle informe le module de maintenance et le module de supervision [03].

On distingue, deux types de surveillance :

- Surveillance de la commande : Basée sur la notion de filtre de commande, elle permet de vérifier que les ordres émis sont conformes à l'état de la partie opérative. Le concept d'Objet Commandable Elémentaire (OCE) a été développé pour la conception de ces filtres de commande
- Surveillance du système opérant : Elle a en charge la surveillance des défaillances du procédé qui, dans le cadre de la sûreté de fonctionnement, sont classées en deux catégories : Les défaillances cataleptiques et Les défaillances progressives

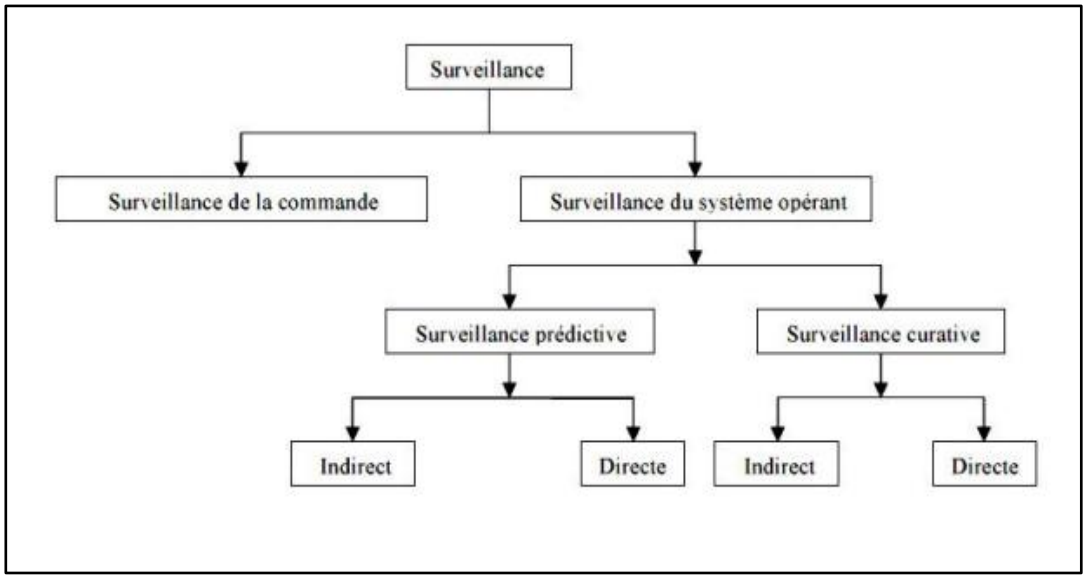


Figure I.1 :*Différents type de surveillance*

I.1.2.3 Diagnostic

Cette étape consiste à partir des défauts détectés, de localiser l'élément défaillant et d'identifier la cause qui a provoqué ce défaut. On appelle signature d'un défaut l'effet de celui-ci sur un ou plusieurs résidus. Si l'on dispose de la connaissance de la signature des défauts, il est possible, à partir de celle-ci, de remonter des effets aux causes, c.-à-d. à partir des résidus non nuls de remonter aux éléments défaillants. Cette étape nécessite donc un modèle de défauts du système permettant la résolution du problème inverse. En résumé, une procédure de diagnostic comprend deux étapes, une étape d'identification de défaut et une étape de localisation de défaut [03].

I.1.2.4 Détection

Cette étape est très importante car elle doit permettre de décider si le système se trouve ou non dans un état de fonctionnement normal en comparant les valeurs des résidus à des seuils qui sont fixés auparavant. En pratique, le signal mesuré est souvent entaché de bruit, et le modèle du système est souvent imparfait car le système réel est soumis à des perturbations non nécessairement mesurables, ce qui rend le résidu non nul même en l'absence de défauts, Par conséquent, cette étape fait le plus souvent appel aux tests statistiques .[2]

I.1.3 La Supervision dans l'industrie(SCADA)

SCADA est l'acronyme de « Supervisory Control And Data Acquisition » (système de contrôle et d'acquisition de données), un terme qui décrit les fonctions de base d'un système SCADA. Les entreprises utilisent des systèmes SCADA pour contrôler les équipements sur tous leurs sites, mais aussi pour collecter et enregistrer des données au sujet de leurs opérations. Un SCADA est généralement fourni sous forme de logiciel combiné à des éléments matériels, tels que des automates programmables industriels (API) et des unités terminales distantes (RTU). L'acquisition des données commence avec les API et RTU, qui communiquent avec l'équipement d'un centre de production, par exemple les machines et les capteurs d'une usine. Les données recueillies à partir de l'équipement sont ensuite envoyées au niveau supérieur, par exemple à une salle de contrôle, où des opérateurs peuvent superviser les contrôles des API et des RTU à l'aide d'interfaces homme-machine (IHM). Les IHM sont un composant essentiel des systèmes SCADA. Il s'agit des écrans que les opérateurs utilisent pour communiquer avec le système SCADA. Ils permettent un très haut niveau de sécurité, pour le personnels et pour l'installation et permettent aussi la réduction des coûts des opérations, les avantages qu'offre le SCADA sont obtenus avec la combinaison des outils softs et hard. [04].

I.1.3.1 Architecture du SCADA

SCADA entoure un transfert de données entre le Serveur (MTU, Master Terminal Units) et une ou plusieurs unités terminales distantes (Remote Terminal Units RTUs), et entre le Serveur et les terminaux des opérateurs, la figure ci-dessous représente un schéma sur l'architecture d'un réseau SCADA qui utilise des routeurs pour joindre le poste de pilotage par le billet de l'Internet [5]

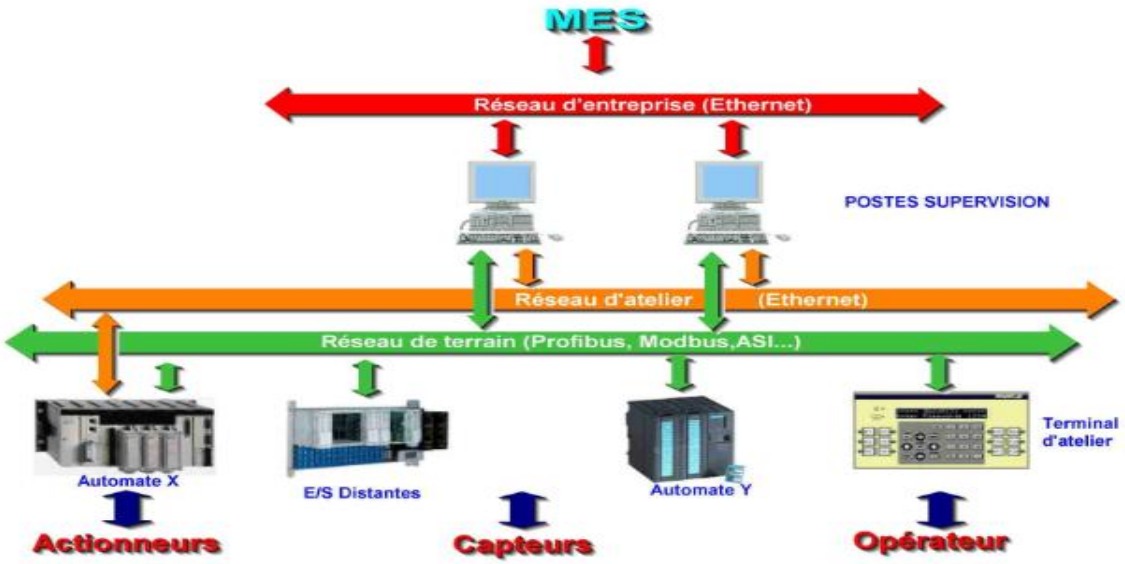


Figure I.2 : *Architecture de la supervision dans un environnement SCADA*

SCADA utilise Une architecture de système de contrôle similaire à Direct Digital Control (DDC) qui attribuant un seul microprocesseur chargé de gérer plusieurs fonctions de contrôle, avec communication numérique entre les unités à microprocesseur, est utilisé pour la gestion de systèmes qui sont par leur nature répartie sur de vastes régions géographiques.

C'est compose des plusieurs terminaux distants (RTU) connectés pour traiter les émetteurs et les éléments de contrôle finaux, en mettant en œuvre des fonctions de contrôle de base telles que démarrage/arrêt du moteur et contrôle de boucle PID. Ces appareils RTU communiquent numériquement avec un maître Dispositif d'unité terminale (MTU) à un emplacement central où les opérateurs humains peuvent surveiller le processus et émettre des commandes

I.1.3.2 Interfaces graphiques du SCADA

Les interfaces graphiques sont un outil très important pour le bon déroulement de la procédure d'aide à la décision, elles sont le seul point d'interaction entre l'opérateur et les algorithmes ainsi, elles aident l'opérateur dans sa tâche d'interprétation et de prise de décision, en lui offrant une très bonne visibilité sur l'état et l'évolution de l'installation, avec l'affichage en différentes des résidus, des alarmes et des propositions sur l'action à entreprendre. La figure Représente une interface graphique de la supervision d'une installation de traitement d'eau

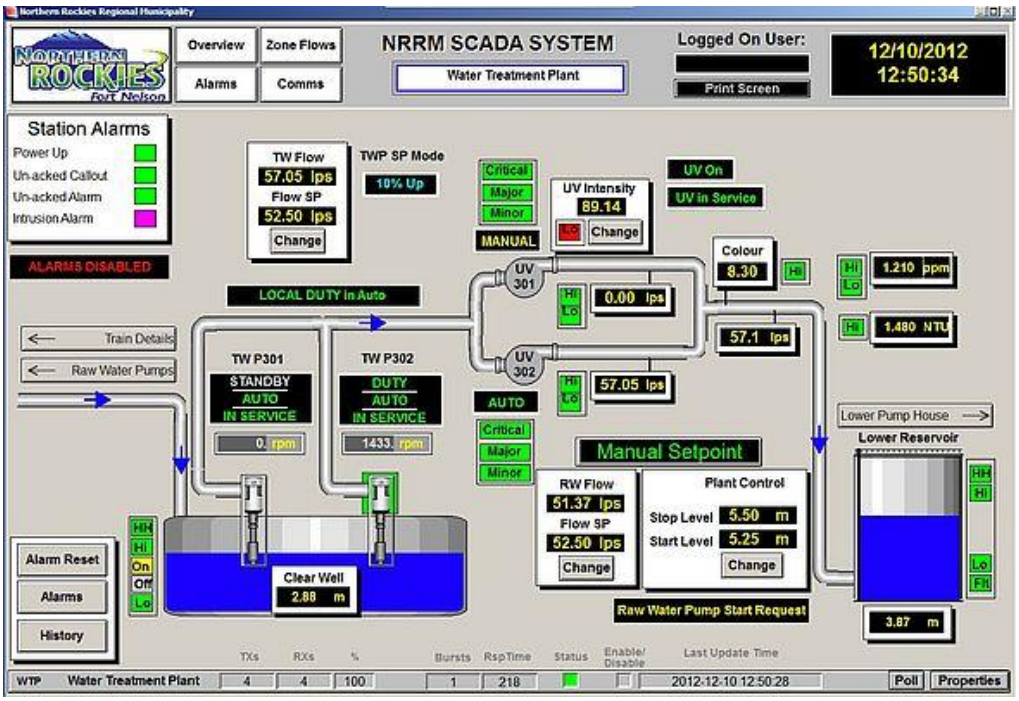


Figure I.3:*Interface graphique de supervision d'une installation de traitement d'eau*

I.1.3.3 Exemples d'utilisation industrielle de SCADA

Les grands et les petits systèmes peuvent être construits à l'aide du concept SCADA. Ces systèmes peuvent aller de quelques dizaines à des milliers de boucles de contrôle, selon l'application.

Les exemples de processus incluent les processus industriels, d'infrastructure et basés sur les installations, comme décrit ci-dessous :

- Les processus industriels comprennent la fabrication, le contrôle des processus, la production d'électricité, la fabrication et le raffinage, et peuvent fonctionner en mode continu, par lots, répétitif ou discret.
- Les processus d'infrastructure peuvent être publics ou privés et comprennent le traitement et la distribution de l'eau, la collecte et le traitement des eaux usées, les oléoducs et gazoducs, la transmission et la distribution d'électricité et les parcs éoliens.
- Processus d'installation, y compris les bâtiments, les aéroports, les navires et les stations spatiales. Ils surveillent et contrôlent les systèmes de chauffage, de ventilation et de climatisation (CVC), l'accès et la consommation d'énergie.

Cependant, les systèmes SCADA peuvent présenter des failles de sécurité. Les systèmes doivent donc être évalués pour identifier les risques et les solutions mises en œuvre pour atténuer ces risques. [6]

I.2 Système de contrôle industriel distribué

Le premier DCS, appelé "TDC 2000", Il a été conçu en 1975 par le Honeywell Dans la même année, il a introduit le système Yokogawa CENTUM et Bristol a introduit le

système « UCS 3000 ».En quelques années, ils ont été conçus autre DCS, jusqu'au début des années 90 plus de 50 DCS disponibles dans le commerce.

Les systèmes de contrôle distribués ont énormément amélioré la gestion des procèdes industriels avec plus de souplesse et de sécurité. Les systèmes de contrôle DCS permettent une conduite et une surveillance centralisée et un contrôle reparti, d'où le nom DCS (Distributed Control System). [7]

Un DCS fait référence à un système de contrôle d'un procédé, dans lequel les éléments régulateurs ne sont pas centralisés (comme le cerveau) mais distribués, avec chaque sous-système sous le contrôle d'un ou plusieurs régulateurs. Les éléments du système tout entier peuvent être connectés en réseau pour assurer les fonctions de communication, de conduite et de surveillance.

DCS est un terme assez large qui décrit des solutions dans une large variété d'industries telles que :

- Réseaux électriques et procédés de génération d'électricité.
- Systèmes de contrôle de l'environnement.
- Feux de circulation routière.
- Procédés de traitement des eaux.
- Raffineries et procédés pétrochimiques.
- Traitement de gaz et du pétrole.

L'architecture d'une solution DCS implique soit des connexions directes avec les équipements physiques tels que les contacts (switches), les pompes et les vannes, soit via un autre système secondaire tel que le système SCADA.

Dans notre cas en vas détaille le système existant dans l'industrie ou nous avons fait notre stage pratique

I.3 Système Experion PKS

L'architecture du système Experion PKS (process Knowledge system) repose sur un réseau informatique modulaire incluant des systèmes d'exploitation, réseaux et protocoles standards. Le système autorise la distribution des fonctions systèmes, telles que l'acquisition et le contrôle des données et une interface graphique utilisateur à tous les niveaux du réseau, afin d'assurer une souplesse et des performances optimales.

Le système Experion PKS a une architecture évolutive capable de supporter les extensions futures. Il permettra l'ajout de contrôleurs et d'unités d'interface opérateur supplémentaires, sans nécessité de prévoir d'autres serveurs de bases de données. L'extension de la taille de la base de données sera simple (par exemple, saisie d'un nouveau code d'autorisation ou licence) et ne nécessitera aucun serveur de base de données supplémentaire. De plus, le système permet de communiquer avec un large éventail d'équipements de contrôle utilisant des modules prêts à l'emploi. Dans une topologie de système Experion de base, le serveur et les automates de process C200/C300 partagent une base de données globale, de sorte que les données ne sont à saisir qu'une seule fois, afin d'éliminer les erreurs et de réduire considérablement le temps de configuration. [8]

Experion peut être divisé à des ensembles de plateforme de composant matériel :

- **La plateforme de supervision** :

Comprend des plateformes informatiques servant de serveur et de station qui peuvent servir d'interface d'étude ou d'exploitation, selon le logiciel chargé sur chaque nœud.

- **L'automate C200, l'automate C300 et/ou le contrôleur Safety Manager** :

Utilisant un petit facteur de forme et prenant en charge une architecture modulaire et extensible. Des automates tiers, le serveur peut fonctionner en liaison avec un certain nombre d'automate tiers, comme les PLC5 et SLC d'Allen Bradley et les automates de Modicon, de GE Fanuc et de Siemens…etc

- **Une plateforme de télécommunication**

Met en œuvre des normes de réseau ouvertes, comme :

Le réseau informatique d'installation basé sur la norme Ethernet ou sur la norme Fault Tolerant Ethernet (FTE) de Honeywell, reliant les serveurs et les clients pour des communications au niveau supervision, et utilise aussi pour les communications entre les automates C300 et le niveau supervision et pour les communications entre automates et E/S déportées.

Le réseau SafeNet assurant des communications sûres entre les automates Safety Manager sur un réseau séparé ou en utilisant le réseau Fault Tolerant Ethernet (FTE).

Un réseau ControlNet, Ethernet ou Fault Tolerant Ethernet (FTE) assurant les communications entre les automates C200 et le niveau supervision ainsi que les communications de pair à pair entre les automates.

I.3.1 L'architecture de system expirion pks

L'architecture prend en charge différents réseaux étendus, utilisant du matériel et des logiciels standards pour relier des nœuds en un système intégré unique. Le système prendra également en charge une interface graphique entièrement fonctionnelle permettant une configuration et une gestion à distance. [9]

L'Experion PKS est une application sophistiquée de régulation et de gestion qui :

- Montre les données système de façon aisément compréhensible.
- Permet le contrôle du système par l'envoi des commandes appropriées.
- Réalise automatiquement les tâches programmées.
- Signale les activités du système, y compris les alarmes et les événements.
- Génère des rapports.

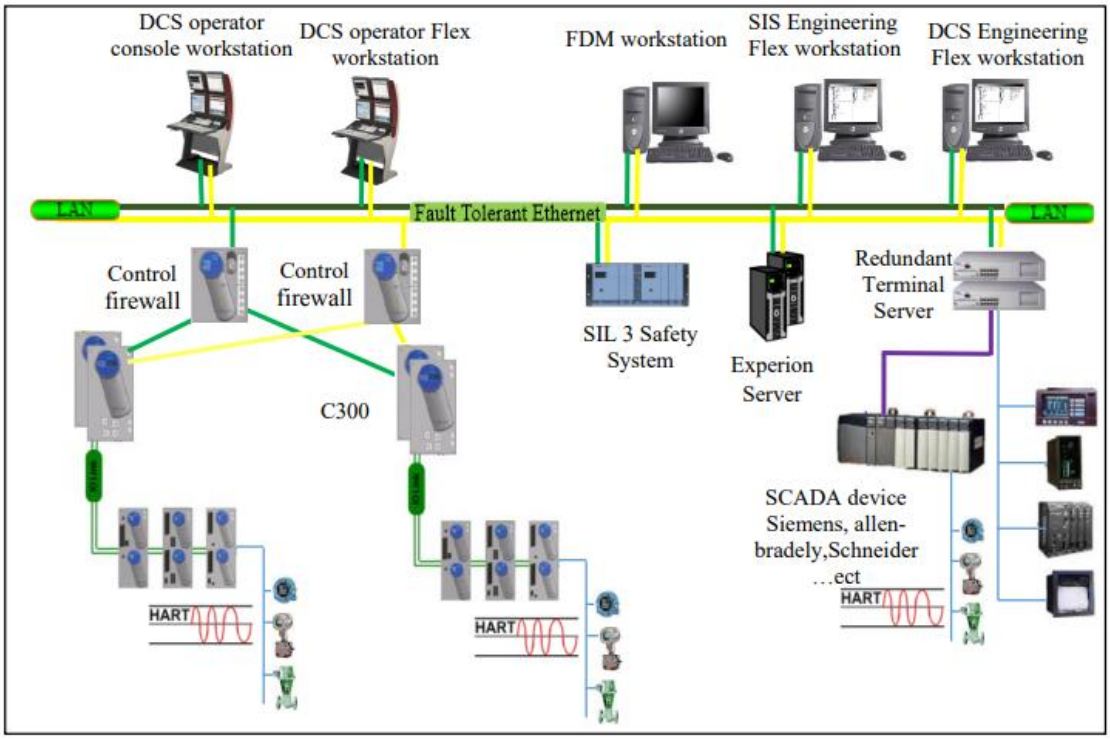


Figure I.4 *:Architecture de DCS Experion PKS.*

L'option Distributed System Architecture (DSA) permet à plusieurs systèmes de partager les données, les points, les alarmes, les messages et l'historique sans qu'il soit nécessaire de dupliquer la configuration sur un serveur quelconque. Elle permet également l'accès global aux données Experion sur tous les serveurs du système. Chaque serveur assure la mise en cache dynamique automatique des données distantes pour tous ses clients, de sorte que les clients accèdent à leur serveur local pour toutes les données. Ce mécanisme maximise l'efficacité tant au niveau des serveurs qu'au niveau du réseau.

I.3.2 Serveur PKS

Le système est fondé sur une véritable approche client-serveur. Un serveur de base de données unique dessert plusieurs clients, tels que des unités d'interface opérateur « station ». L'application Experion PKS fonctionne sur un serveur « l'ordinateur principale » qui réalise également plusieurs fonctions système, comme : L'acquisition et le traitement des données, La gestion des alarmes et des événements et La collecte, l'archivage et le suivi des historiques. [9]

Les environnements de serveur et de poste de travail informatique d'Experion fournissent des fonctions de niveau supervision, notamment :

- Le contrôle et la conduite des procédés configurés.
- Emission d'alarmes sur événements critiques.
- Consignation des événements et rapports événementiels déclenchés par les équipements, ou les opérateurs ou à la demande.
- Enregistrement de l'historique du process pour des intervalles spécifiés et affichage par échantillonnage, projection graphique, pour aider à discerner les tendances.

- Création d'écrans spécifiques.
- Configuration en ligne des canaux, des automates et des points, personnalisation des rapports et des suivis de tendance.
- Option serveur redondant pour augmenter la disponibilité avec lien actif pour la synchronisation.

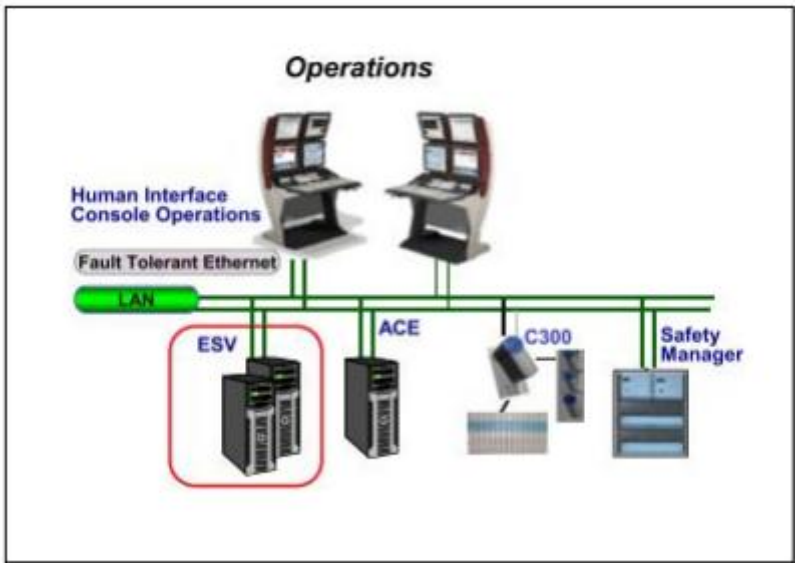


***Figure I.5** :Intégration du serveur dans une architecture PKS.*

I.3.3 La Station

La station Experion est une interface puissante de l'opérateur qui peut être utilisée dans différentes formes pour différentes fonctions autour d'une usine. Elle emploie la technologie principale HMIWeb pour fournir la plate-forme requise de différentes opérations industrielles d'aujourd'hui. Il y a plusieurs types de station Experion disponible à satisfaire une gamme générale de besoins, incluent :

- Experion Station – Flex (ES-F).
- Experion Station – Console (ES-C)
- Experion Station – Console Extension (ES-CE)

I.3.4 Options de sécurité

Il y a trois types de sécurité :

- **Sécurité opérateur :**

On attribue à chaque utilisateur un identifiant et un mot de passe à utiliser pour ouvrir une session et se connecter à la station.

- **Sécurité station :**

Une station donne à n'importe quel utilisateur le niveau de sécurité opérateur. Toutefois, les utilisateurs peuvent changer pour un niveau supérieur s'ils connaissent le mot de passe correspondant.

- **Sécurité intégrée :**

Avec la sécurité intégrée, chaque utilisateur a un compte intégré, lequel est la combinaison d'un compte utilisateur de Windows et d'un compte opérateur de la station. Les paramètres de sécurité du compte utilisateur de Windows servent à authentifier l'utilisateur, tandis que les paramètres de sécurité du compte opérateur de la station servent à régir les droits de l'utilisateur dans Experion.

I.3.5 FTE (Fault Tolerant Ethernet)

Le Fault Tolerant Ethernet est un réseau de supervision et de contrôle d'Experion PKS qui relie les éléments du site (capteurs et actionneurs) aux opérateurs à travers la base de données du serveur, pour leurs exploitations sur les interfaces hommes machines représentés par les pages graphiques au niveau des stations. Il est dédié à la mission de contrôle fournissant la tolérance aux pannes, les temps de réponse rapides, et la sécurité requise pour les entreprises industrielles. [10]

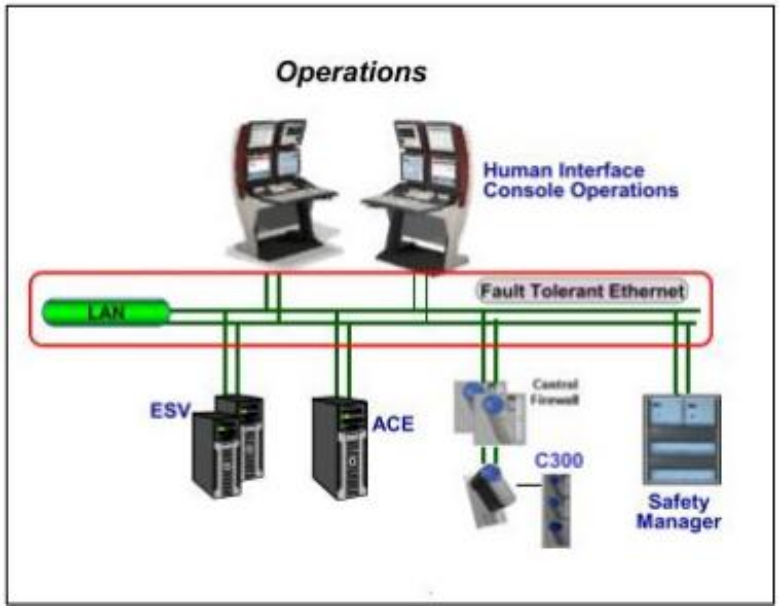


***Figure I.6** : Intégration du Fault Tolerant Ethernet dans une architecture PKS*

FTE est une topologie de réseau unique avec redondance, cette redondance est réalisée en utilisant le Honeywell driver FTE et les composants disponibles dans le marché. Le pilote et les composants compatibles FTE permettent une communication réseau sur une autre voie lorsque la route principale échoue. Chaque nœud FTE est connecté deux fois à un seul réseau local via la double carte d'interface réseau NIC (network interface card).

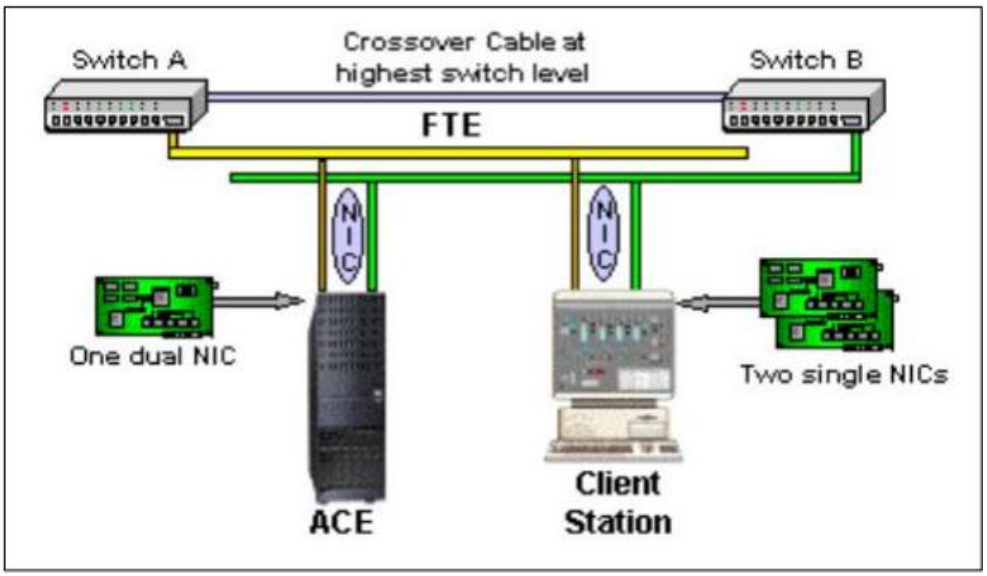


Figure I.7 : *FTE pour une connexion réseau double.*

I.3.6 Concepts et fonctions

I.3.6.1 Propriété globale des données

Dans le système Experion, l'automate est propriétaire de certaines données tandis que d'autres données sont détenues par une base de données sur serveur appelé « référentiel système ».Chaque donnée est la propriété d'une seule de ces entités, dans tout le système Experion, cette donnée à tout moment à la même valeur pour tous les utilisateurs, et chaque entité de données n'est créé qu'une seule fois dans Control Builder.

I.3.6.2 Redondance

Les composants essentiels du système sont dotés avec des composants secondaires pour basculer d'un équipement principal vers un équipement secondaire en cas de problème au niveau de l'équipement principal. Le système Experion a été conçu pour mettre en œuvre la protection par redondance au niveau des serveurs, des réseaux, des automates et de certaines E/S.

I.3.6.3 Prise en charge des équipements de télé conduite SCADA

Par définition, un système Experion comprend les automates C200, C300 et/ou ACE. Il peut aussi comprendre des équipements de télé conduite (SCADA), tels que des équipements à interface série (RS 232 ou RS 485), des connexions ControlNet et/ou des interfaces Ethernet (par exemple, MODBUS TCP/IP) ou diverses combinaisons de ces équipement.

De plus, des connexions et des points SCADA peuvent coexister avec des C200 et des TPS dans des limites données. L'application Quick Builder fournit les outils permettant de réaliser une interface de réseau SCADA.Un système purement SCADA ne comprend pas d'automates C200, C300 ou ACE.

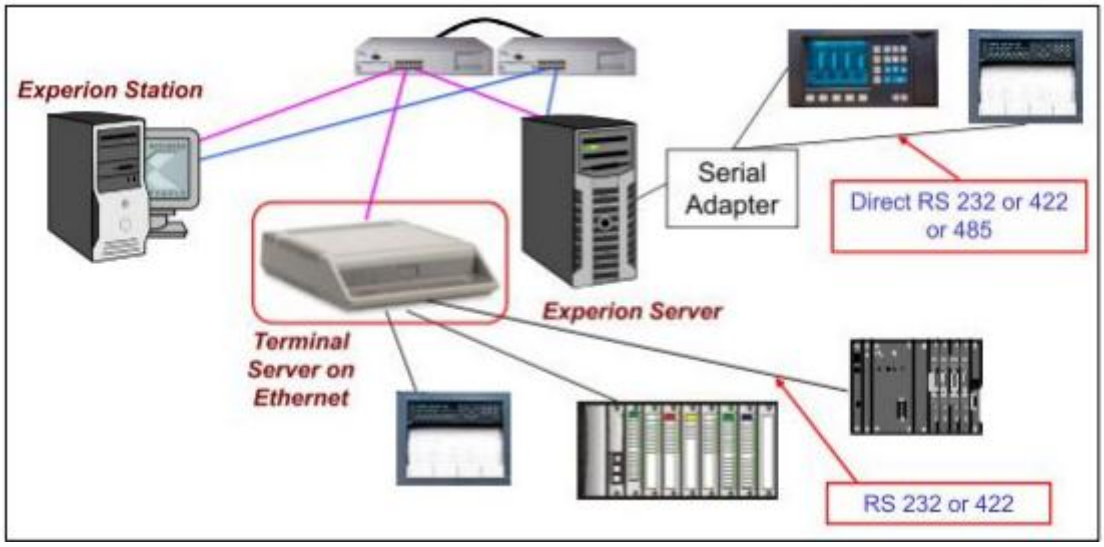


Figure I.8 *:Connexion SCADA en mode série avec le serveur Experion.*

I.3.6.4 Configuration Studio

Configuration Studio constitue un point central pour configurer le système Experion. Les outils individuels nécessaires pour configurer telle ou telle partie du système sont lancés à partir de Configuration Studio. Dans Configuration Studio, il y a une liste personnalisée des tâches à accomplir pour configurer le système. Lorsqu'on clique sur une tâche, l'outil approprié est lancé.

Ces outils comprennent :

- **Enterprise Model Builder** :

Outil graphique pour la création de modèle d'asset.

- **Quick Builder** :

Outil graphique pour la création d'objets matériels, tel que les Flex stations, les imprimantes, les automates/régulateurs/contrôleurs et les points standard pour la communication SCADA.

On utilise quick builder pour la configuration des contrôleurs et des points non Experion (non C200, C300), ces derniers seront reliés au système Experion PKS en utilisant une communication SCADA, il est aussi utiliser pour la création et la configuration d'objet matérielle tell que les stations et les imprimantes.

- **Écrans système :**

Ecrans utilisés pour configurer, les rapports, les écrans de groupe, les suivis de tendance, les paramètres de station et les consoles stations.

- **Control Builder :**

Outil graphique servant à créer une stratégie de conduite pour les automates de process.

Control Builder est un outil technique pour la création de stratégie de conduite. Sa conception graphique, réduit considérablement le travail de conception, de mise en œuvre des applications de contrôle-commande.

- **HMIWeb Display Builder :**

Outil graphique pour la création d'écrans spécifiques à l'aide de fonctionnalité de type internet. Les écrans sont enregistrés en format HTML,On utilise le HMIWeb Display Builder pour créer des pages de supervision spécifiques aux différentes unités industrielles. HMIWeb Display Builder est fourni avec un jeu de bibliothèque couvrant toute une variété d'industrie. [11]

- **Blocs fonctionnels :**

Un bloc fonctionnel est l'unité de base d'une fonctionnalité de contrôle-commande qui comprend des blocs de commande régulatrice, des blocs de commande d'appareil, des blocs logiques, des blocs séquentiels et des blocs auxiliaires. Avec Control Builder, les blocs fonctionnels sont choisis dans une bibliothèque de bloc fonctionnel. Ces blocs fonctionnels sont ensuite combinés pour mettre en œuvre la stratégie de conduite désirée.

I.3.7 Matériels de contrôle-commande série C

I.3.7.1 Description de la série C

La Série C est le terme utilisé pour décrire les nouveaux contrôleurs C300 et les modules associés, le matériel de la série C est conçus pour offrir plus d'espace utilisable, conçu verticalement avec une inclinaison de 18◦ pour mieux dissiper la chaleur, Combine le module d'E /S et les connexions de champ dans un seul ensemble ce qui permet un câblage plus efficace.

L'architecture de l'automate de process C300 est caractérisée par une construction verticale qui rend le montage et le câblage plus efficace. Il comprend des connexions intégrées pour FTE redondant et pour module d'E/S série C ou Process Manager.

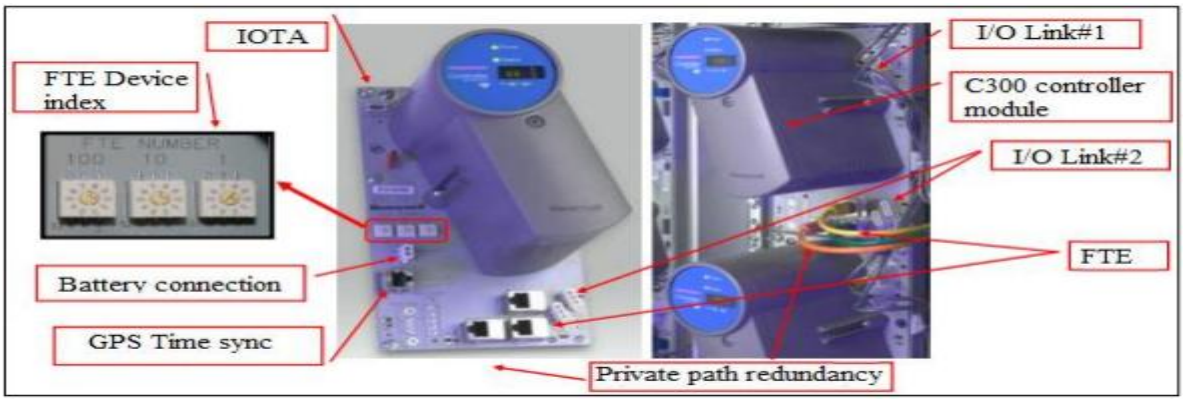


Figure I.9 *:Contrôleur C300.*

Le C300 autorise la mise en œuvre d'une configuration redondante, grâce à une connexion dédiée, et fournit l'environnement d'exécution CEE pour les applications de contrôle-commande, comme son prédécesseur l'automate C200. Le pare-feu Control Firewall assure l'intégrité des données et un sous-système d'alimentation intégré distribue efficacement l'énergie électrique dans une armoire. Les E/S série C TOR et analogiques peuvent être redondantes en option. [12]

I.3.7.2 Les modules d'entrées / sorties série C

Ils sont Conçues pour être utilisées avec l'automate C300, qui comportent des modules AI et AO compatibles avec HART, les modules AI bas niveaux, les modules d'entrées numériques qui supportent la haute tension AC et 24V DC, les modules de sorties numériques qui fournit 24 VDC. Les modules de protection (SP) qui prennent en charge l'algorithme de contrôle de vitesse et de logique de vote, Les modules Servo Valve Positioner (SVP) qui prennent en charge le PID. Un module d'E/S série C redondant est disponible en option.

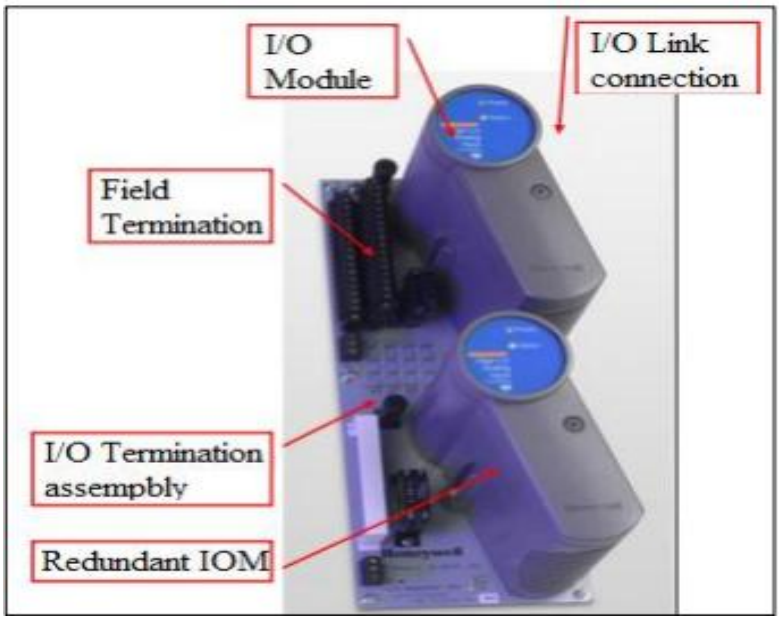


Figure I.10 :*Série C modules d'entrées sorties redondants sur une IOTA.*

I.3.7.3 Le pare-feu Control Firewall :

Fournit huit connexions FTE pour C300 Controller et série C FILMs dans une armoire de commande, et une liaison montante au réseau de communications FTE de supervision. Il rejette les messages Ethernet qui ne sont pas nécessaires au contrôle-commande. Si le trafic de contrôle-commande tend à ralentir, il limite le flux des messages les moins importance.

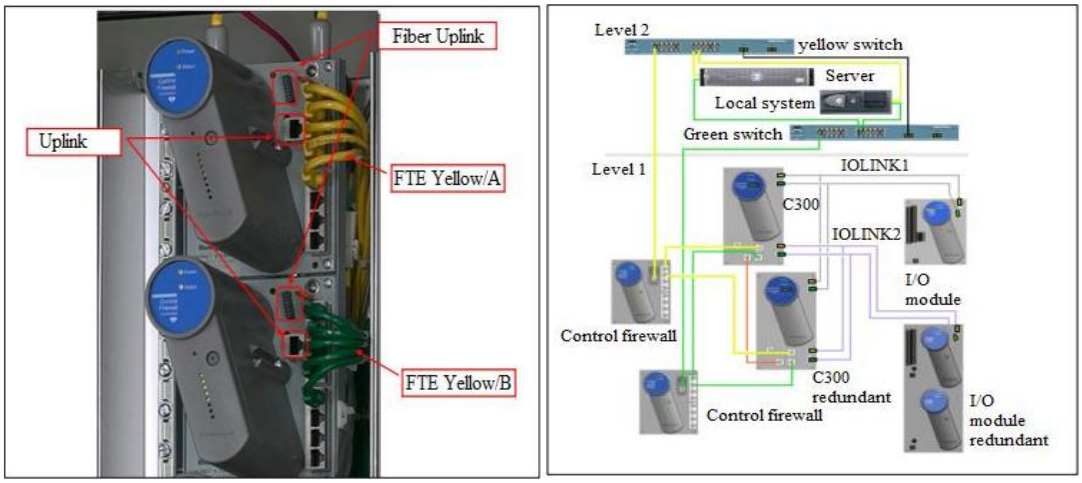


Figure I.11 :*Module control firewall et topologie de série C.*

I.3.8 Système de sécurité (ESD) Safety Manager

Safety Manager, la dernière plateforme de sécurité Honeywell, est une solution de sécurité efficace et fiable pour la conduite des procédés et d'équipements ainsi que la protection des procédés. La plateforme Safety Manager est issue de la plate-forme de sécurité Fail Safe Controller (FSC), qui a été adoptée et fait ses preuves dans le monde entier depuis plus de 15 ans. Elle intègre une technologie éprouvée grâce à une expérience de 20 ans en gestion de sécurité des procédés, et utilise des données de sécurité, des applications, des diagnostics systèmes et des stratégies de conduite critique. [12]

Le safety Manager est un système de sécurité programmable basé sur des microprocesseurs redondants doubles (DMR) ou redondants quadruples (QMR) pouvant exécuter une large gamme de contrôle de procédé et des fonctions de sécurité de haute intégrité, incluant :

- Systèmes de gestion des brûleurs / chaudières.
- Protection des processus et arrêt d'urgence.
- Contrôle et protection des turbines et des compresseurs.
- Systèmes de détection d'incendie et de gaz.
- Surveillance du pipeline.

En a fait l'identification des composants de Safety Manager comme ci-dessus

I.3.8.1 Contrôleur

Sa fonction est d'exécuter et de contrôler la logique d'application charger dans Safety Manager, de Communiquer avec d'autres systèmes, effectuer des diagnostics sur des fonctions critiques pour la sécurité, et répondre aux défauts détectés tout en maintenant la sécurité. Un contrôleur SM se compose d'un processeur de contrôle, d'un châssis de contrôleur, d'un module de batterie et de clé, et un fond de panier de processeur de contrôle, qui comprend un Bus système redondant rapide.

I.3.8.2 Processeur de Contrôle

Composant principal du contrôleur SM composé d'une Unité d'alimentation (PSU), module de processeur principale (Quadruple Processor Pack (QPP)) et 1 ou 2 modules de communication (USI).

I.3.8.3 les modules Entrés sorties

Leurs fonction est de connecter le contrôleur SM aux capteurs et actionneurs du site, la variété des modules E/S comprend des modules sûrs, analogiques et numériques, redondants et non redondants avec différentes tension

I.3.8.4 Bus d'entrées sorties

Le bus d'entrées sorties est l'interface pour transporter les données de terrain et les données de diagnostic entre les modules d'E/S et le contrôleur.

Les deux modules d'E/S sont placés dans le même châssis d'E/S et sont câblés en parallèle. En principe quand un default est détecté dans un canal d'une entrée, il sera désactivé à travers son processeur de contrôle correspondant, la valeur correcte est obtenue à partir du module E/S connecté à l'autre processeur de contrôle via le lien interne redondant, avant le début du cycle d'application. En principe, lorsqu'un défaut est détecté dans un canal de sortie, ce canal est désexcité par le SMOD. La valeur correcte est entraînée dans le champ par l'autre processeur de contrôle, mais pas après que les deux processeurs de contrôle ont accepté sa valeur via le lien interne de contrôle redondant.

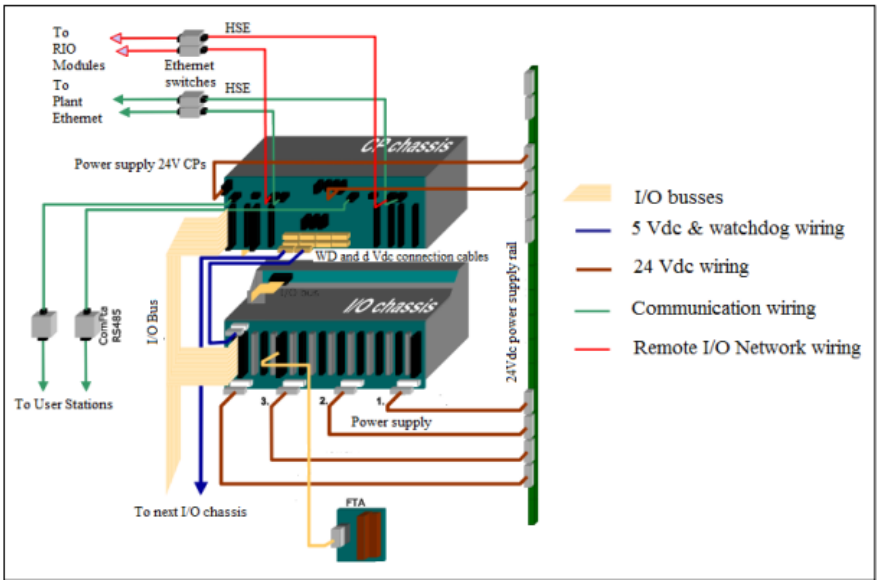


Figure I.12 : *Les différentes connexions du Safety Manager*

I.3.9 Intégration dans l'Experion PKS

Le safety manager support l'intégration dans l'Experion PKS, cela permet d'unifier et de réunir le contrôleur de sécurité de Honeywell avec la plateforme Experion PKS. L'intégration est réalisée à travers l'Universal Safety Interface (USI) via High Speed

Ethernet sur le Fault Tolerant Ethernet, qui est placé avec le contrôleur Safety Manager. Ce module USI fait de safety manager une partie intégrée de l'architecture d'Experion PKS.[12]

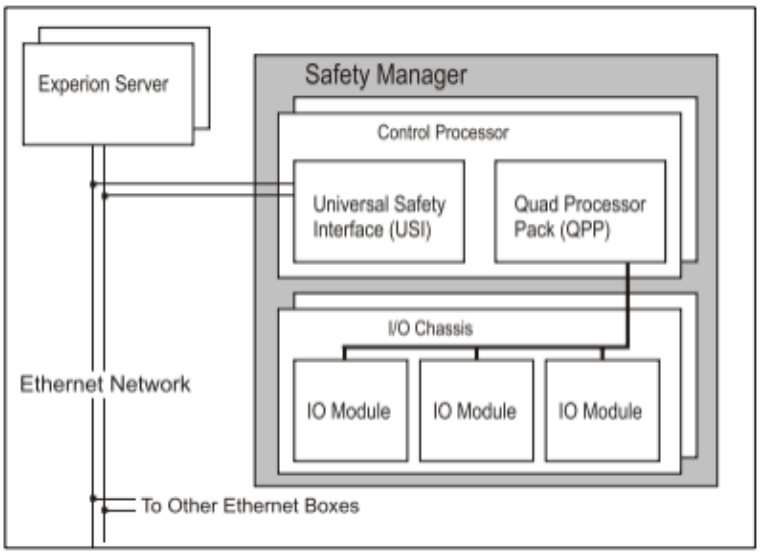


Figure I.13 : *Connexion du SM avec l'architecture PKS via l'USI.*

Cette intégration fournit des données ponctuelles, des diagnostics et des informations sur le système, des alarmes et des événements, des affichages pour opérateurs et des séquences d'informations sur les événements dans toute l'usine, accessible dans la station via le serveur d'Experion.

I.3.9.1 Communication PCDI

En outre, la communication entre pairs (peer to peer) via l'interface Peer Control Data Interface (PCDI) entre les contrôleurs Safety Manager et C300, peut facilement être établie, cela permet une communication entièrement redondante, robuste, rapide et rentable entre la sécurité des processus et contrôle de processus sans compromettre les exigences de séparation IEC 61508.

PCDI utilise le réseau FTE redondant existant, pour la communication entre le Safety Manager et le C300 sans le besoin d'équipement additionnel. L'utilisation de PCDI pour connecter safety Manager fait que les données ponctuelles du Safety Manager peuvent être utilisées dans n'importe quel contrôleur C300 sur la même communauté FTE.

I.4 Conclusion :

Dans ce chapitre, nous avons présenté les systèmes de contrôle industriel, la supervision en générale et l'utilité de système DCS pour le contrôle et la surveillance des processus industriels, ainsi que ces diffèrent fonctionnalité donc en conclu que La conversion de protocole est la clé pour connecter les appareils au réseau, les anciens systèmes de contrôle utilisent une version non routable de Modbus (Modbus-RTU), tandis que les systèmes modernes utilisent une version routable (Modbus-TCP), dans le chapitre suivant nous présenterons les different protocol existant et en base sur le protocole Modbus.

Chapitre 02 :

Les Protocoles de communication industrielle

II.1 Introduction

Les machines et systèmes industriels modernes sont censés être toujours plus intelligents et hautement efficaces. L'intelligence et l'efficacité sont déterminées par la facilité, la rapidité et la précision de l'accessibilité des données entre les systèmes. Pour qu'un réseau industriel devienne efficace, les communications horizontales et verticales entre les dispositifs et les systèmes de la pyramide ISA-95 doivent être transparentes. Pour ce faire, une connectivité robuste est essentielle.

L'écosystème d'une entreprise industrielle englobe un large éventail de dispositifs, de machines et d'applications, qui sont censés communiquer entre eux indépendamment de leur marque, de leur méthode de communication, etc. Il est donc nécessaire d'intégrer l'ensemble des protocoles de communication industrielle dans le système pour permettre cette communication.

Protocole Il s'agit d'une méthode de communication de données numériques entre deux ou plusieurs dispositifs situés à des endroits différents ou sur un réseau.

De nombreux protocoles sont utilisés dans le monde entier. Nous nous concentrerons sur les protocoles utilisés par les PLC, DCS, PAC et les dispositifs qui existent sur le même réseau que les contrôleurs industriels. Certains protocoles utilisés avec les contrôleurs servent uniquement à la communication de données, généralement par des capteurs et des transmetteurs. D'autres sont utilisés à la fois pour les communications de données et les applications de contrôle.

II.2 Les Protocole de communications industrielles

II.2.1 Définition d’un protocole de communication :

Les protocoles de communication industriels sont un ensemble de règles établies qui ont pour but de créer une communication transparente entre les appareils et les systèmes industriels. Au cours des dernières années, de nombreux protocoles de communication ont été développés, chacun étant dédié pour un objectif et un environnement spécifique.

Ces protocoles servent à doter les équipements de terrains tels que les capteurs sans fil, machines d’une connectivité dans et entre les réseaux industriels. De manière générale, les réseaux industriels peuvent être séparés en quatre couches différentes :

II.2.1.1 Couche physique

Cette couche est constituée de transducteurs, d’actionneurs ou de capteurs sans fil. Ces équipements collectent des informations de contrôle de processus ou d’environnement et les retransmettent aux systèmes de contrôle ou de surveillance. Ils exécutent également les commandes de terrain, telles que le déplacement physique du rotor.

II.2.1.2 Couche de contrôle

Cette couche se compose de contrôleurs logiques, tels que des automates ou des systèmes de contrôle distribués (DCS). Leur rôle principal est d’assurer le contrôle local des différents équipements d’un réseau industriel.

II.2.1.3 Couche de supervision

Cette couche comprend un SCADA/HMI pour un meilleur contrôle et une meilleure supervision de l'activité industrielle.

II.2.1.4 Couche d'exécution et de planification

Cette couche est un intermédiaire important entre les objectifs commerciaux et opérationnels. La planification des ressources d'entreprise est un bon exemple de fonction d'exécution dans cette couche.

II.2.2 Fonctionnalités des protocoles de communication industrielle

Les protocoles de communication industriels sont utilisés pour transporter des informations d'un endroit à un autre. Ils forment un empilement de couches du modèle OSI. Chaque couche communique avec celle au-dessus ou en dessous en transmettant des informations sous une forme appropriée à la couche suivante. L'avantage d'un tel système est que chaque couche fait abstraction de certaines fonctions techniques de la couche au-dessus. [13]

Le modèle OSI étant un modèle théorique de la façon dont les communications se produisent sur un réseau industriel est composé de 7 couches :

II.2.2.1 **Couche 1 (physique)**

Cette couche ne considère que les aspects physiques d'un réseau les câbles, convertisseurs, dispositifs d'interconnexion, etc

II.2.2.2 **Couche 2 (liaison de données)**

Cette couche s'intéresse à la façon dont les couches 1 et 3 fonctionnent ensemble.

II.2.2.3 Couche 3 (réseau)

Cette couche fournit un schéma d'adressage pour le routage des données et des messages

II.2.2.4 **Couche 4 (Transport)**

Cette couche s'assure que les messages arrivent à leur destination correcte.

II.2.2.5 **Couche 5 (session)**

Cette couche gère les connexions réelles entre les systèmes.

II.2.2.6 **Couche 6 (Présentation)**

Cette couche traite les différents systèmes qui représentent les données.

II.2.2.7 Couche 7 (Application)

Cette couche concerne principalement les applications logicielles utilisées sur un écran d'ordinateur.

L'utilisation de protocoles de communication industriels implique à la fois des Softwares et du Hardware, et il est difficile, mais parfois nécessaire, de faire la différence entre les fonctions exécutées par les périphériques Hardware et les fonctions exécutées par les Softwares. Il est donc utile de se référer parfois au modèle OSI.

II.2.3 Les caractéristiques des protocoles de communication industrielle

Chaque protocole de communication industriel a sa propre méthode de vérification qui couvre l'authentification, la détection, la signalisation et la correction d'erreurs. Cette vérification est utilisée dans les protocoles pour s'assurer que les données reçues à un nœud de destination sont identiques aux données transmises. Le bruit électrique, les problèmes de connexion ou une pièce défectueuse peuvent interférer avec les communications de données efficaces. Si le nœud expéditeur obtient une réponse du nœud récepteur indiquant que les données ont été reçues comme prévu, la tâche est terminée. Mais si une incompatibilité a été détectée ou qu'aucun accusé de réception des données n'est correctement reçu, les données seront renvoyées, un message d'information sera généré ou une autre action appropriée sera entreprise.

II.2.4 **Méthodes de communication**

Les protocoles de communication industriels peuvent communiquer avec les réseaux en utilisant 3 méthodes différentes :

II.2.4.1 Maître/esclave

Les communications maître-esclave sont assez simples. Il est également souvent utilisé dans les contrôleurs industriels. Dans cette technique, un dispositif maître envoie des commandes à un ou plusieurs esclaves, et la ou les esclaves répondent.

Par exemple, un automate qui est connecté à 3 racks externes (distants) peut être le maître. Toutes les 40 millisecondes, le processeur de l'automate vérifie les entrées reçues des racks externes puis il envoie une requête au 1er rack externe pour voir si des entrées ont changé ou si des pannes matérielles se sont produites. Toute modification est signalée au processeur. Le processus sera répété pour le 2ème rack externe une fois l'interrogation du 1er rack est terminée et le 3ème rack après le contrôle du 2ème rack.

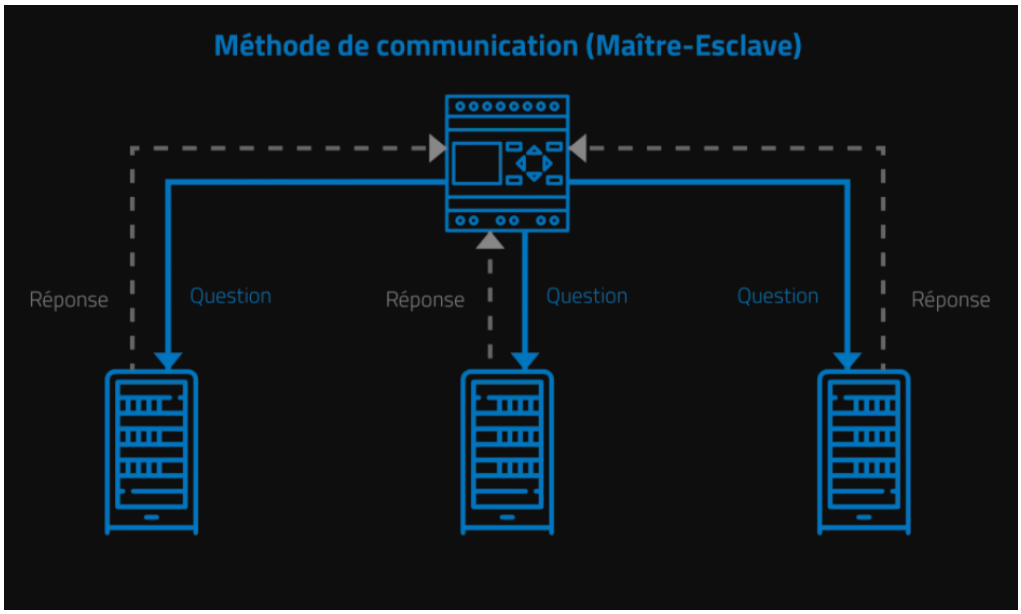


Figure II.1 : Méthode de communication Maître/esclave

II.2.4.2 Anneau à jeton

La méthode de communication en anneau à jeton repose sur le passage de jetons pour les communications de données. Comme son nom l'indique, les communications en anneau à jeton sont utilisées sur les réseaux en anneau. « Un message spécial, appelé jeton, est transmis d'une machine à une autre autour de l'anneau, et chaque machine ne peut transmettre ce message que quand elle détient le jeton ». La communication en anneau à jeton n'est jamais devenue répandue dans les entreprises et l'industrie. Cependant, le passage de jetons ne nécessite pas l'utilisation d'un réseau en anneau, il peut être utilisé sur des réseaux d'autres topologies.[14]

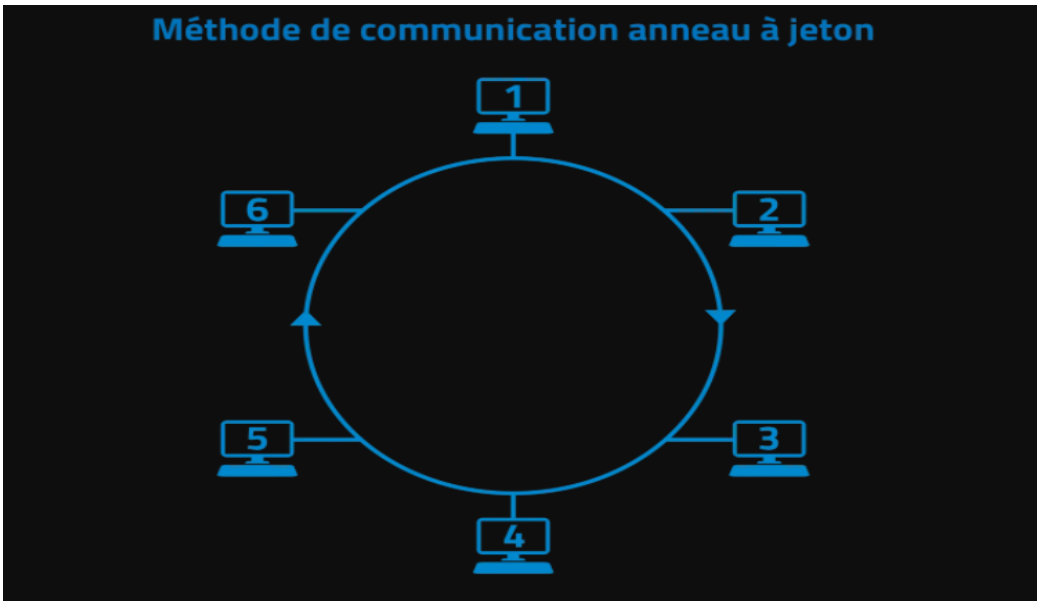


Figure II.2 :Méthodes de communication Anneau à jeton

II.2.4.3 Pair-à-pair (Peer-to-Peer : P2P)

Lorsqu'un protocole de communication peut transmettre des données à n'importe quel autre nœud du réseau, le réseau est appelé « Pair-à-pair ». La capacité de la méthode de communication pair-à-pair n'est possible que si le protocole de la couche application la prend en charge. Les réseaux pair-à-pair peuvent être implémentés sur des réseaux avec de nombreuses topologies différentes.

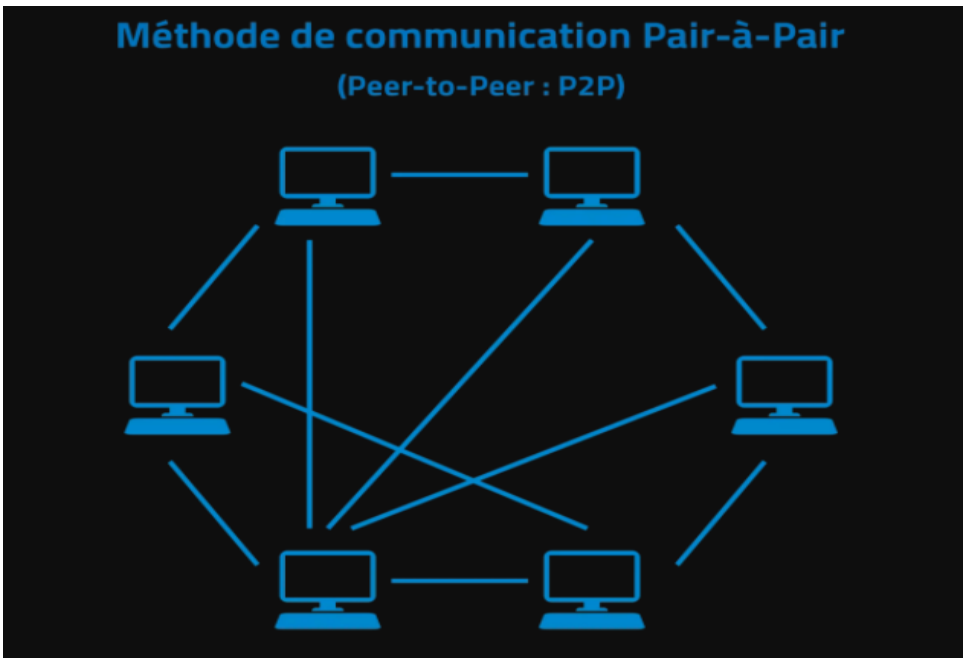


Figure II.3 :Méthodes de communication Pair-à-pair

II.3 Les types de protocoles de communication industriels

Les protocoles de communication industriels se subdivisent entre les types propriétaires (Cisco Discovery Protocol (CDP), Cisco IOS NetFlow…) et ouverts (Modbus, Profinet, Profibus…)

Les protocoles ouverts, quant à eux, sont développés par plus d'une organisation agissant en collaboration. La plupart des protocoles ouverts se sont développés à la suite de longs processus itératifs de propositions, de consultations et d'ajustements aux normes proposées avant la publication d'une norme.

II.3.1 Les protocoles des communications industrielles ouvertes

II.3.1.1 Protocole PROFINET

PROFINET est un protocole de communication industriel standard utilisé dans le monde entier, c'est une sorte d'Ethernet industriel, il est 100% compatible Ethernet comme défini dans le standard IEEE (IEEE 802.3).

PROFINET et PROFIBUS sont tous deux développés par l'organisation PROFIBUS INTERNATIONAL (PI). PROFINET utilise le modèle consommateur/fournisseur. C'est pourquoi il fournit une vitesse plus rapide de communication des données. Le nombre maximum d'appareils dans le réseau PROFINET peut être de 256. PROFINET fournit une vitesse de communication de 100 mbps ou 1Gbit/s. PROFINET est plus rapide avec plus de bande passante et fournit un plus grand espace de massage. PROFINET peut communiquer avec plusieurs appareils industriels.

II.3.1.1.1. Les avantages de PROFINET

- **Vitesse :** PROFINET fournit un échange de données à haute vitesse entre le contrôleur et les appareils, ce qui convient aux applications de contrôle de mouvement rapide.
- **Précision :** PROFINET est très précis car il fournit un cycle de précision < 1 µs. Cette précision en fait une technique de haute performance.
- **Structures de grandes quantités :** Avec PROFINET, 256 appareils peuvent être contrôlés.
- **Redondance sans faille** : PROFINET fournit la redondance sans faille pour les applications critiques.
- **Topologies flexibles :** PROFINET permet l'utilisation de topologies en étoile, en arbre et en anneau en plus de la topologie linéaire.
- **Standard ouvert.**
- **Extensibilité :** Le réseau PROFINET peut être étendu à volonté.

II.3.1.2 **Protocole PROFIBUS**

PROFIBUS (Process Field Bus) est un protocole de communication Fieldbus standard, utilisé dans l'automatisation industrielle, c'est un protocole de communication standard ouvert, utilisé pour interfacer des contrôleurs, des répéteurs, des capteurs, des IHM, des actionneurs, comme PROFINET, basé sur RS-485 et compatible avec la norme européenne EN-50170. Et communique par un câble torsadé et blindé à deux conducteurs.[15]

II.3.1.2.1. Les types de protocoles PROFIBUS

Il existe deux variantes de PROFIBUS (PROFIBUS DP et PROFIBUS AP) :

II.3.1.2.2. PROFIBUS DP (Périphériques décentralisés)

Est utilisé pour faire fonctionner des capteurs et des actionneurs par le biais d'un contrôleur décentralisé. La vitesse de transfert des données varie de 9,6Mbit/s à 12 Mbit/s. Une vitesse particulière peut être choisie pour un réseau donné afin de laisser suffisamment de temps à la communication pour tous les appareils. Il est possible de connecter des appareils sur PROFIBUS DP32.

II.3.1.2.3. PROFIBUS PA (automatisation des processus)

Est utilisé pour surveiller les équipements de mesure des processus par le biais du système de contrôle des processus. PROFIBUS PA est plus lent que PROFIBUS DP. Il fonctionne à une vitesse fixe de 32 kb/s. Les appareils connectés à PA sont limités. PA et DP fonctionnent tous deux avec le même protocole et utilisent un câble à deux conducteurs.

II.3.1.3 Protocole AS-l

Le protocole AS-I (Actuators Sensor - Interface) est l'un des protocoles de communication d'automatisation les plus en vogue. Il s'agit d'une solution très rentable, facile et efficace utilisée dans les industries de nos jours. Il est conçu pour connecter des dispositifs d'entrée/sortie sur un seul câble à deux conducteurs, remplaçant ainsi les systèmes de câblage discrets complexes.

Il est possible de connecter jusqu'à 31 dispositifs sur le système AS-l traditionnel et 62 dispositifs dans le système V-2.1 sur un seul câble à deux conducteurs, la longueur maximale du câble pouvant atteindre 100 mètres pour une seule boucle,est un système maître-esclave, le maître AS-l peut échanger les données entrée/sortie (commande, statut de retour, etc.) avec les dispositifs esclaves sur un seul câble à deux conducteurs, chaque esclave peut transférer jusqu'à 4 entrées et 4 sorties sur un cycle de bus, c'est un véritable protocole de communication standard ouvert,Les 31 noeuds peuvent être interrogés en 5 msec.

II.3.1.4 Protocole HART

Le protocole de communication HART (Highway Addressable Remote Transducer) est un protocole hybride pour l'envoi et la réception de données entre des appareils intelligents et un contrôleur. Il s'agit d'un protocole de communication bidirectionnel pour le contrôle et la surveillance d'appareils intelligents compatibles avec le protocole HART, qui permet une communication bidirectionnelle de 4 à 20 mA en mode semi-duplex,

HART est très fiable, facile à utiliser et très utile pour la mise en service et l'étalonnage des appareils SMART à l'aide d'un appareil portatif appelé communicateur HART.

HART est plus flexible que n'importe quelle autre technologie de communication.

HART utilise une modulation par déplacement de fréquence (FSK) de 1200 bauds basés sur la norme Bell 202 pour superposer des informations numériques au signal conventionnel de 4 à 20 mA.

HART préserve la stratégie de contrôle actuelle car il permet au signal traditionnel de 4 à 20 mA de coexister avec la communication numérique sur le système de boucle à 2 fils existant.

Il s'agit d'une norme technologique véritablement ouverte, soutenue par tous les fabricants d'appareils de mesure. **[16]**

II.3.1.4.1. FONCTIONNALITÉ HART

- Configuration ou reconfiguration du dispositif SMART
- Diagnostic du dispositif SMART.
- Dépannage des dispositifs SMART
- État de santé du dispositif
- Rééquilibrage du SPAN
- Changement en ligne de l'ID de l'appareil, du numéro d'étiquette.

- Informations sur le produit

II.3.1.4.2. RÉSEAU HART

Il existe deux types de réseaux HART

a. le mode point à point
b. Mode multi-dépôts

II.3.1.5 Protocole Fondation Fieldbus

Le système Fondation Fieldbus est un système de communication entièrement numérique, sériel, bidirectionnel et multidébit conçu pour interconnecter les dispositifs de bus de terrain intelligents (capteurs, actionneurs et autres E/S) avec le système de contrôle.

La Fondation Fieldbus remplace la technologie 4 à 20mA et le système de câblage complexe.

L'option de câblage à sécurité intrinsèque est disponible dans le système FF pour réduire le facteur environnemental dangereux.

Les appareils de terrain intelligents ne convertissent pas seulement les données des capteurs, mais ils peuvent également disposer d'un riche ensemble de blocs fonctionnels et d'algorithmes de contrôle.[17]

II.3.1.5.1. Caractéristiques de Protocole Fondation Fieldbus

- Disponibilité accrue des données - un dispositif peut être la source et/ou la destination de nombreuses variables de contrôle.
- L'horodatage des paramètres de contrôle est effectué dans les appareils de terrain et couplé aux données de contrôle pour optimiser les performances de fonctionnement.
- La technologie FF assure une communication de données à haut débit et fiable dans un environnement difficile.
- L'intégration de la fonction de contrôle et de surveillance est également possible dans les dispositifs de montage sur rail qui libèrent de l'espace dans le contrôleur central.

II.3.1.5.2. Implémentations du protocole Fondation Fieldbus

- Il existe deux types d'implémentations du protocole Fondation Fieldbus [18] :

- Foundation Fieldbus H1
- HSE (High-speed Ethernet)

II.3.1.6 Protocole du Réseau de contrôleurs (Controller Area Network (CAN))

Le bus CAN a été développé par BOSCH. CAN est un système de diffusion de messages multi-maître, et son taux de transfert de données maximum est de 1 mégabit par seconde (bps). Contrairement à un réseau Ethernet traditionnel, le CAN n'envoie pas de gros blocs de données point à point d'un nœud A à un nœud B.

Dans un réseau CAN, de nombreux messages courts comme la température ou le régime sont diffusés à l'ensemble du réseau, ce qui assure la cohérence des données dans chaque nœud du système.

Le protocole de bus CAN a été développé à l'origine pour les industries automobiles pour remplacer le câblage complexe harnais de câbles complexes par un bus à deux fils.

II.3.1.7 Protocole Modbus

Développé par la société Modicon (le fabricant original de l'automate programmable, ou PLC) en 1979 pour être utilisé dans ses produits de contrôle industriel, Modbus est un protocole conçu spécificalement pour l'échange de données de processus entre des dispositifs de contrôle industriel. La norme Modbus ne spécifie aucun détail de mise en réseau physique, et peut donc être déployée sur de nombreux différents types de réseaux physiques. En d'autres termes, Modbus relève principalement de la couche 7 du modèle de référence OSI (la couche dite "Application") et est donc compatible avec tous les protocoles de communication de niveau inférieur, notamment EIA/TIA-232, EIA/TIA-485, Ethernet (ce dernier via TCP/IP) et un réseau spécial à passage de jeton également développé par Modicon, appelé Modbus Plus. La norme Modbus défines principalement la signification des diverses commandes Modbus, le schéma d'adressage utilisé pour placer les données dans les appareils, et le formatage des données. Modbus consiste en un ensemble de codes numériques normalisés destinés à lire les données des appareils industriels et à y écrire des données. Un appareil industriel conforme à Modbus a été programmé pour comprendre ces codes et y répondre de manière appropriée lorsqu'il les reçoit. Les codes Modbus les plus simples lisent et écrivent des bits uniques de données dans la mémoire de l'appareil, par exemple le statut d'un canal d'entrée PLC, d'un canal de sortie PLC ou d'un bit de statut dans un programme PLC. D'autres codes Modbus fonctionnent sur des mots de données de 16 bits, utiles pour lire et écrire les valeurs accumulées des compteurs et des minuteries, les opérandes des instructions mathématiques, les signaux analogiques convertis, etc.[19]

Les premières implémentations de Modbus utilisaient EIA/TIA-485 comme couche physique du réseau, qui est strictement un protocole de couche 1. Cela signifie que Modbus devait spécifier un schéma d'arbitrage de canal afin de négocier les communications avec de multiples dispositifs sur un réseau. L'arbitrage choisi était maître/esclave, où un PLC fonctionnait comme le dispositif maître Modbus et tous les autres dispositifs fonctionnaient comme des esclaves Modbus. Il est intéressant de noter que ce vestige d'arbitrage maître/esclave survit jusqu'à ce jour, même lorsque les commandes Modbus sont communiquées via des réseaux ayant leurs propres méthodes d'arbitrage differing. Par exemple, les commandes Modbus communiquées sur Ethernet font toujours référence à des adresses "esclaves" même si le réseau Ethernet sur lequel ces messages sont envoyés utilise

l'arbitrage CSMA/CD. En d'autres termes, il y a un indice de la couche OSI 2 intégré dans les messages Modbus qui dicte toujours quels dispositifs Modbus peuvent émettre des commandes et lesquels doivent obéir aux commandes.

II.3.1.7.1. Aperçu général sur le Modbus

Nous pouvons commencer notre exploration de Modbus en considérant d'abord un exemple de système de moteur contrôlé par PLC qui n'utilise pas Modbus. Ici, l'API envoie des signaux de commande de marche avant, arrière et d'arrêt et de contrôle de vitesse câblés individuellement à un variateur de fréquence (variable-frequency drive (VFD)) qui envoie ensuite une puissance triphasée de fréquence variable à un moteur électrique pour effectuer une tâche utile [20]:

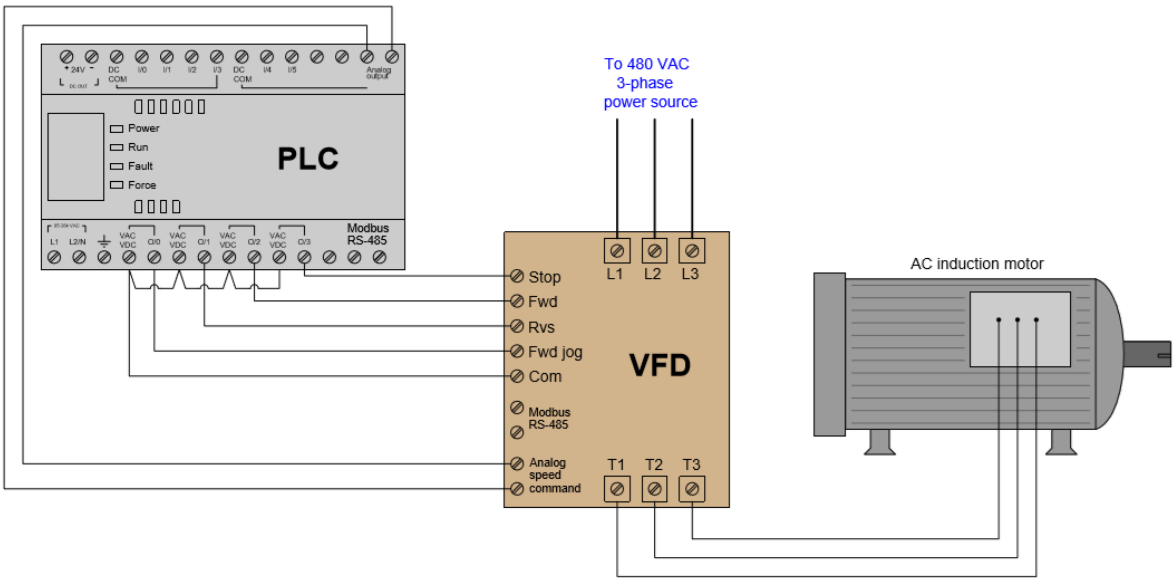


Figure II.4 :système de moteur contrôlé par PLC qui n'utilise pas Modbus

Les commandes discrètes (par exemple Stop, Forward, Reverse) ne sont rien de plus que des fermetures de contact on/off fournies par les canaux de sortie de l'automate aux bornes d'entrée du VFD. Lorsque le PLC commande au VFD de fonctionner en sens inverse, il active simplement le canal de sortie 0/1 qui ferme un contact de relais à l'intérieur du PLC pour connecter la borne "Rvs" du VFD à la borne "Com" du VFD. Le VFD détecte cette continuité électrique, et répond en faisant tourner le moteur dans le sens inverse. La vitesse du moteur est commandée par un signal de tension analogique (généralement de 0 à 10 volts CC) émis par l'automate, 0 volt représentant la vitesse zéro et 10 volts la vitesse maximale. Le VFD reçoit ce signal de tension analogique et y répond en fournissant au moteur à induction la fréquence appropriée du courant alternatif triphasé.

The VFD receives this analog voltage signal and responds to it by outputting the appropriate frequency of three-phase AC power to the induction motor.

Bien que ce système soit certainement fonctionnel, il peut être amélioré par l'utilisation de Modbus.

Considérons maintenant ce système de commande de moteur mis à jour, où les seuls fils de connexion entre l'automate et le VFD sont un seul câble à deux conducteurs entre les bornes Modbus/RS-485 des deux dispositifs. L'automate fonctionne comme un dispositif maître Modbus tandis que le VFD fonctionne comme un esclave Modbus :

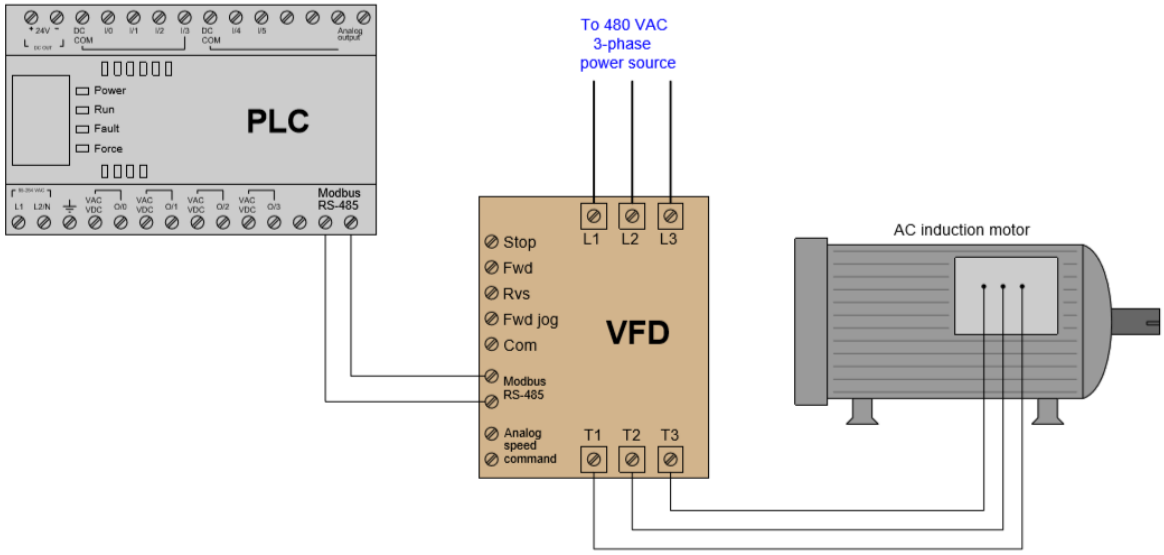


Figure II.5 :système de moteur contrôlé par PLC qui utilise Modbus

En utilisant les commandes Modbus appropriées transmises au VFD, le PLC est capable d'émettre toutes les mêmes commandes (par exemple, Stop, Forward, Reverse, speed control) comme auparavant, mais en utilisant beaucoup moins de fils. Par exemple, le code de commande Modbus 05 écrit un seul bit de données au dispositif récepteur, ce qui permet à l'automate d'envoyer des commandes à signaux discrets au VFD, une à la fois. Lorsque le PLC commande au VFD de fonctionner en sens inverse, il émet une commande 05 suivie d'un bit de données "1" adressé à l'emplacement mémoire approprié dans le VFD réservé au bit de commande "Inverse". Lorsque l'automate commande au VFD de changer la vitesse du moteur, il émet un code Modbus 06 ("write register") suivi d'un nombre de 16 bits représentant la vitesse désirée du moteur et l'adresse appropriée dans le VFD réservée à la commande de vitesse. Non seulement le PLC peut émettre toutes les mêmes commandes qu'auparavant, mais il peut également lire des données du VFD, ce qu'il ne pouvait pas faire auparavant. Par exemple, si le VFD fournit un emplacement de mémoire pour stocker les codes de défaut (par exemple, motor overcurrent, bus undervoltage, etc.), le PLC peut être programmé pour émettre un code Modbus 03 pour lire un registre unique (nombre binaire de 16 bits) à partir de cet emplacement de mémoire dans le VFD, et ainsi surveiller l'état du VFD pour alerter les techniciens humains des problèmes potentiels, et/ou pour modifier sa propre commande du moteur.Un autre avantage de la norme de communication Modbus est qu'elle est conçue pour s'adresser à plusieurs appareils sur le même réseau. Cela signifie que notre automate hypothétique n'est pas limité à la commande et à la surveillance d'un seul moteur, mais jusqu'à 247 dispositifs esclaves Modbus distincts sur le même câble de communication bifilaire.

II.3.1.7.2. Trames de données Modbus

La norme de communication Modbus défines un ensemble de commandes permettant de lire (recevoir) et d'écrire (transmettre) des données entre un appareil maître et un ou plusieurs appareils esclaves connectés au réseau. Chacune de ces commandes est référencée par un code numérique, les adresses des registres internes des appareils maîtres et esclaves (sources et destinations des données) étant spécifiées avec le code de fonction dans la trame Modbus. Deux formats différents sont spécifiés dans la norme Modbus : ASCII et RTU. La différence entre ces deux modes est la façon dont les adresses, les codes de fonction, les données et les bits de contrôle d'erreur sont représentés. En mode Modbus ASCII, toutes les adresses des dispositifs esclaves, les codes de fonction et les données sont représentés sous la forme de caractères ASCII (7 bits chacun), qui peuvent être lus directement par n'importe quel programme de terminal (par exemple minicom, Hyperterminal, kermit, etc.) interceptant le flux de données série. Cela facilite le dépannage : pouvoir visualiser directement les trames de données Modbus sous une forme lisible par l'homme. En mode Modbus RTU, toutes les adresses des dispositifs esclaves, les codes de fonction et les données sont exprimés sous forme binaire brute. Des techniques de contrôle d'erreur differentes sont utilisées pour les modes ASCII et RTU également. Le diagramme suivant compare les trames de données pour les deux modes Modbus [21] :

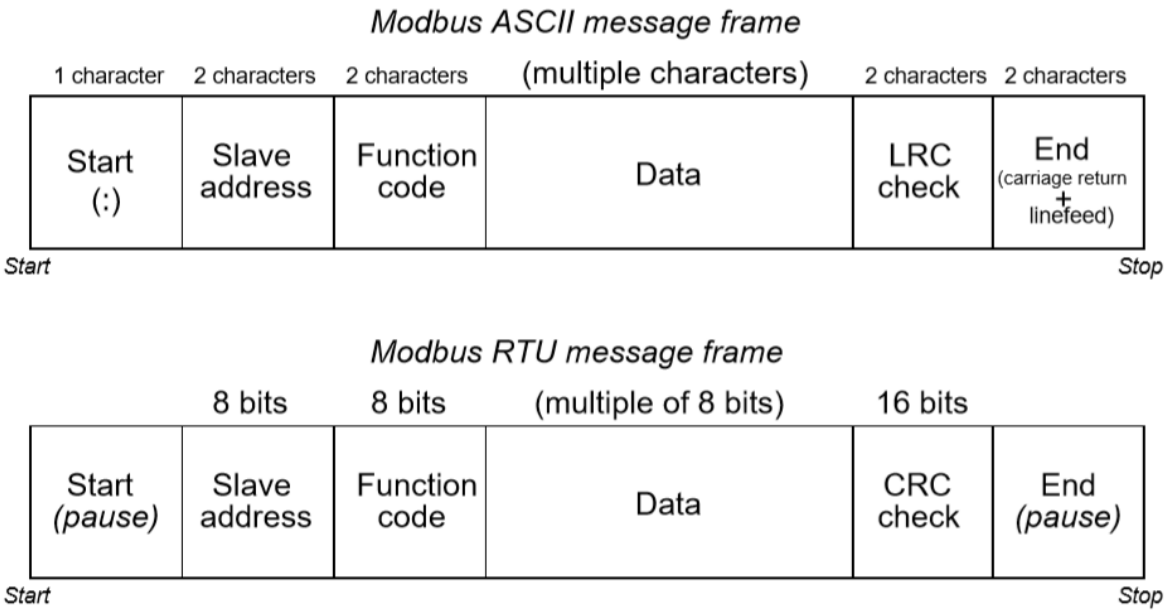


Figure II.6:Trame de Modbus RTU/ASCII

Comme vous pouvez le constater en comparant les deux trames, les trames ASCII nécessitent presque deux fois plus de bits que les trames RTU, ce qui rend Modbus ASCII plus lent que Modbus RTU pour tout débit de données donné (bits par seconde).

II.3.1.7.3. Codes de fonction et adresses Modbus

Une liste des codes de fonction Modbus les plus utilisés figure dans le tableau suivant :

Modbus code (decimal)	Function
01	Read one or more PLC output “coils” (1 bit each)
02	Read one or more PLC input “contacts” (1 bit each)
03	Read one or more PLC “holding” registers (16 bits each)
04	Read one or more PLC analog input registers (16 bits each)
05	Write (force) a single PLC output “coil” (1 bit)
06	Write (preset) a single PLC “holding” register (16 bits)
15	Write (force) multiple PLC output “coils” (1 bit each)
16	Write (preset) multiple PLC “holding” registers (16 bits each)

Figure II.7 :liste des codes de fonction Modbus les plus utilisés

Les données en direct à l'intérieur de tout dispositif numérique sont toujours situées à une certaine adresse dans la mémoire vive (RAM) de ce dispositif. La norme d'adressage Modbus "984" défines des ensembles d'adresses numériques fixées où différents types de données peuvent se trouver dans un automate ou un autre dispositif de contrôle. Les plages d'adresses absolues (selon le schéma Modbus 984) sont indiquées dans ce tableau, chaque adresse contenant 16 bits de données :

Modbus codes (decimal)	Address range (decimal)	Purpose
01, 05, 15	00001 to 09999	Discrete outputs (“coils”), *read/write*
02	10001 to 19999	Discrete inputs (“contacts”), *read-only*
04	30001 to 39999	Analog input registers, *read-only*
03, 06, 16	40001 to 49999	“Holding” registers, *read/write*

Figure II.8 :Les plages d'adresses absolues

Notez que toutes les plages d'adresses Modbus commencent par le chiffre un, et non par zéro comme c'est le cas pour de nombreux systèmes numériques. Par exemple, un PLC avec seize canaux d'entrée analogiques numérotés de 0 à 15 par le fabricant peut "mapper" ces registres d'entrée aux adresses Modbus 30001 à 30016, respectivement.

II.3.1.7.4. Formats de commande des fonctions Modbus

Chaque trame de données Modbus, que ce soit en mode ASCII ou RTU, comporte un field désigné pour les "données". Pour chaque fonction Modbus, le contenu de ce field "données" suit un format spécific. L'objectif de cette sous-section est de documenter les formats de données requis pour les fonctions Modbus courantes, à la fois le message "Query" transmis par le dispositif maître Modbus à un dispositif esclave, et le message "Response" correspondant renvoyé au dispositif maître par le dispositif esclave interrogé. Comme chaque trame de données Modbus est conditionnée en multiples de 8 bits (RTU), elles sont généralement représentées dans le texte comme des octets individuels (deux caractères hexadécimaux). Par exemple, un "mot" de 16 bits de données Modbus tel que

1100100101011011 sera typiquement documenté comme C9 5B avec une espace délibérée séparant les octets "haut" (C9) et "bas" (5B).

- **Code de fonction 01 - Read Coil(s)**

Cette fonction Modbus lit les états des sorties discrètes ("coils") du dispositif esclave, renvoyant ces états par blocs de huit (même si le "number of coils" spécifié dans la requête n'est pas un multiple de huit !). Les adresses Modbus pertinentes pour cette fonction vont de 00001 à 09999 (décimales), mais l'adresse de départ est un nombre hexadécimal représentant le (n-1)ème registre à partir du début de cette plage (par exemple, l'adresse décimale 00100 serait spécifiée comme hexadécimale 00 63).

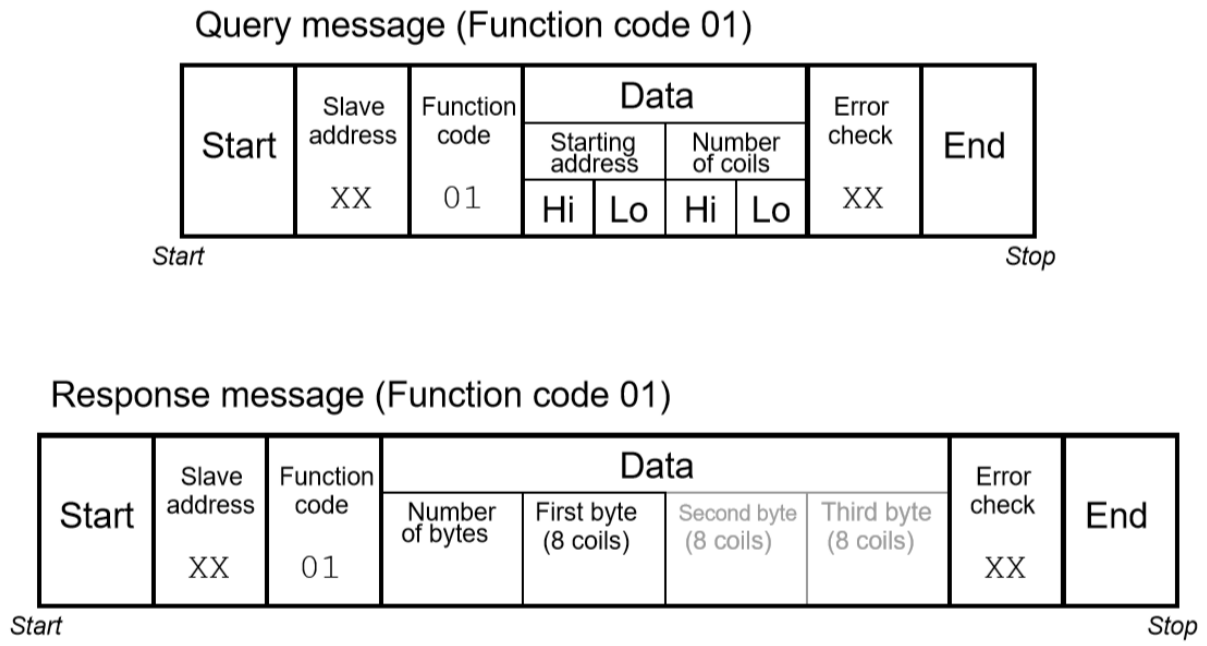


Figure II.9 :Trame de requête et réponse de fonction 01

Notez que les deuxième et troisième octets représentant l'état des coil sont représentés en gris, car leur existence suppose que plus d'un octet des coils a été demandé dans la query.

- **Code de fonction 02– Read Contact(s)**

Cette fonction Modbus lit les états des entrées discrètes (" contacts ") du dispositif esclave, en renvoyant ces états par blocs de huit (même si le " nombre de contacts " spécifié dans la requête n'est pas un multiple de huit !). Les adresses Modbus pertinentes pour cette fonction vont de 10001 à 19999 (décimal), mais l'adresse de départ est un nombre hexadécimal représentant le (n-1)ème registre à partir du début de cette plage (par exemple, l'adresse décimale 10256 serait spécifiée comme hexadécimal 00 FF).

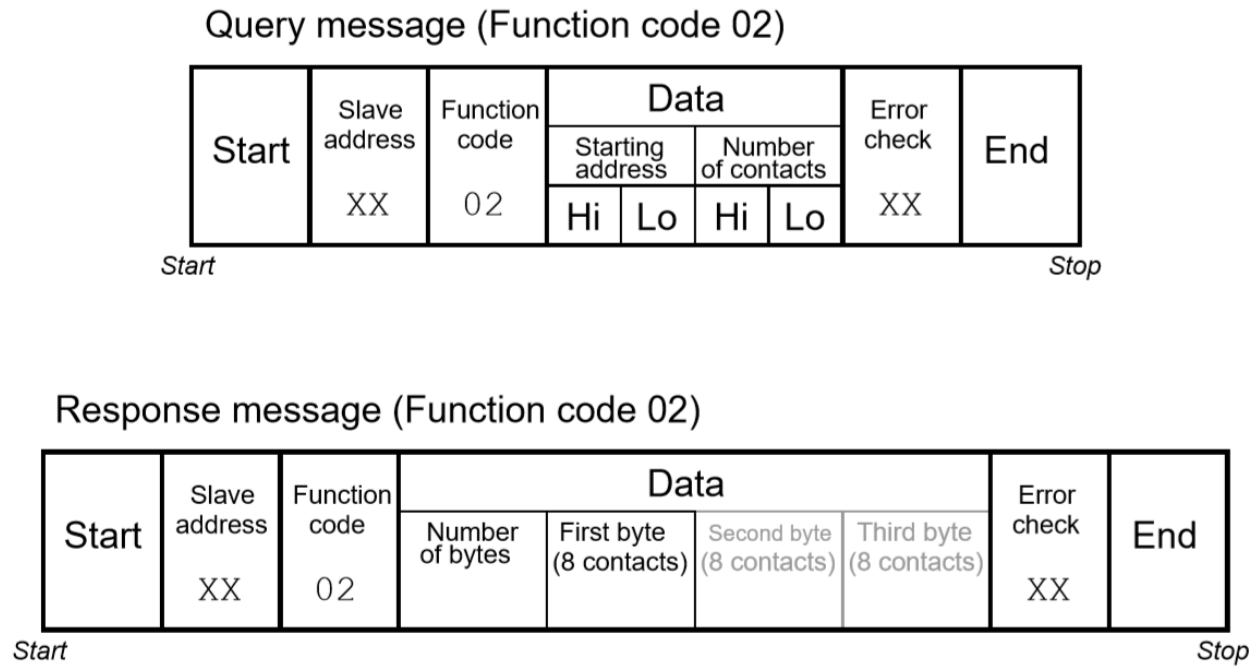


Figure II.10 :Trame de requête et réponse de fonction 02

- **Code de function 03 – Read Holding Register(s)**

Cette fonction Modbus lit les états des registres de " holding" dans le dispositif esclave, la taille de chaque registre étant supposée être de deux octets (16 bits). Les adresses Modbus pertinentes pour cette fonction vont de 40001 à 49999 (décimal), mais l'adresse de départ est un nombre hexadécimal représentant le (n-1)ème registre à partir du début de cette plage (par exemple, l'adresse décimale 40980 serait spécifiée comme hexadécimal 03 D3).

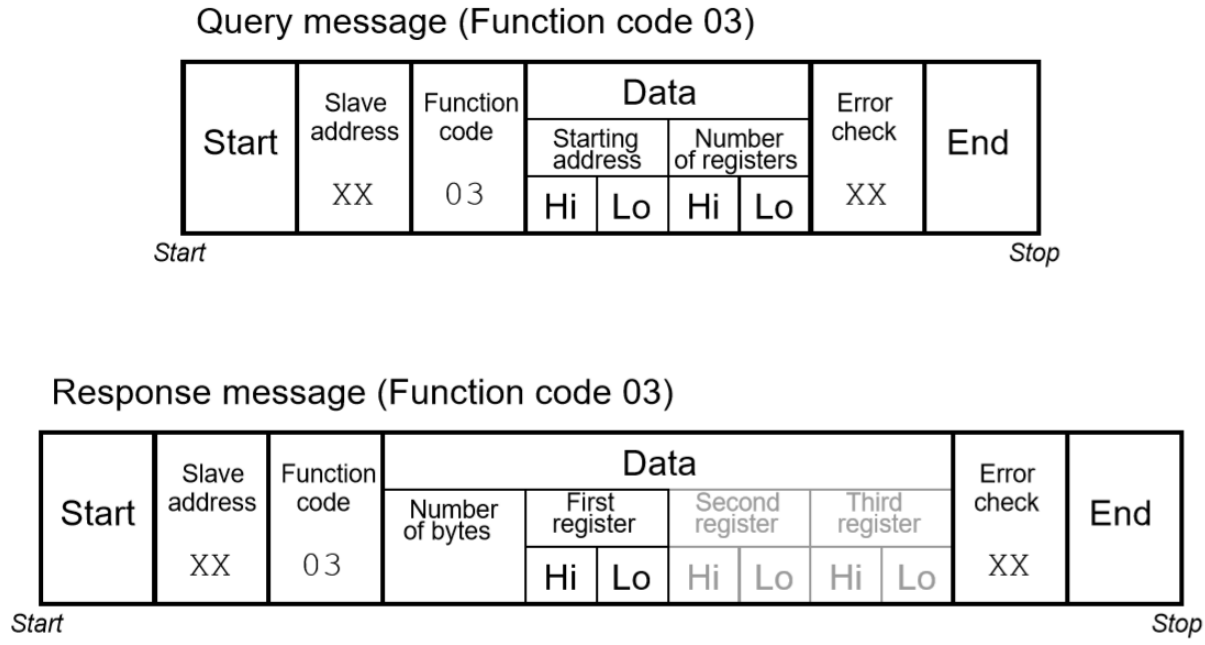


Figure II.11 :Trame de requête et réponse de fonction 03

Notez que puisque le message de query spécifie le nombre de registres (chaque registre ayant une taille de deux octets), et que le message de réponse répond avec le nombre d'octets, le field "nombre d'octets" du message de réponse aura une valeur double de celle du field "nombre de registres" du message de requête. Notez également que le nombre maximal de registres qui peuvent être demandés dans le message de query (65536) avec des valeurs d'octets "high" et "low" dépasse largement le nombre d'octets que le message de réponse peut indiquer (255) avec sa valeur d'un seul octet.

- **Code de fonction 04 – Read Analog Input Register(s)**

Cette fonction Modbus est pratiquement identique à 03 (Read Holding Registers), sauf qu'elle lit plutôt les registres " d'entrée " : les adresses 30001 à 39999 (décimales). Comme pour toutes les adresses relatives Modbus, l'adresse de départ spécifiée dans les deux messages est un nombre hexadécimal représentant le (n-1)ème registre à partir du début de cette plage (par exemple, l'adresse décimale 32893 serait spécifiée comme hexadécimale 0B 4C).

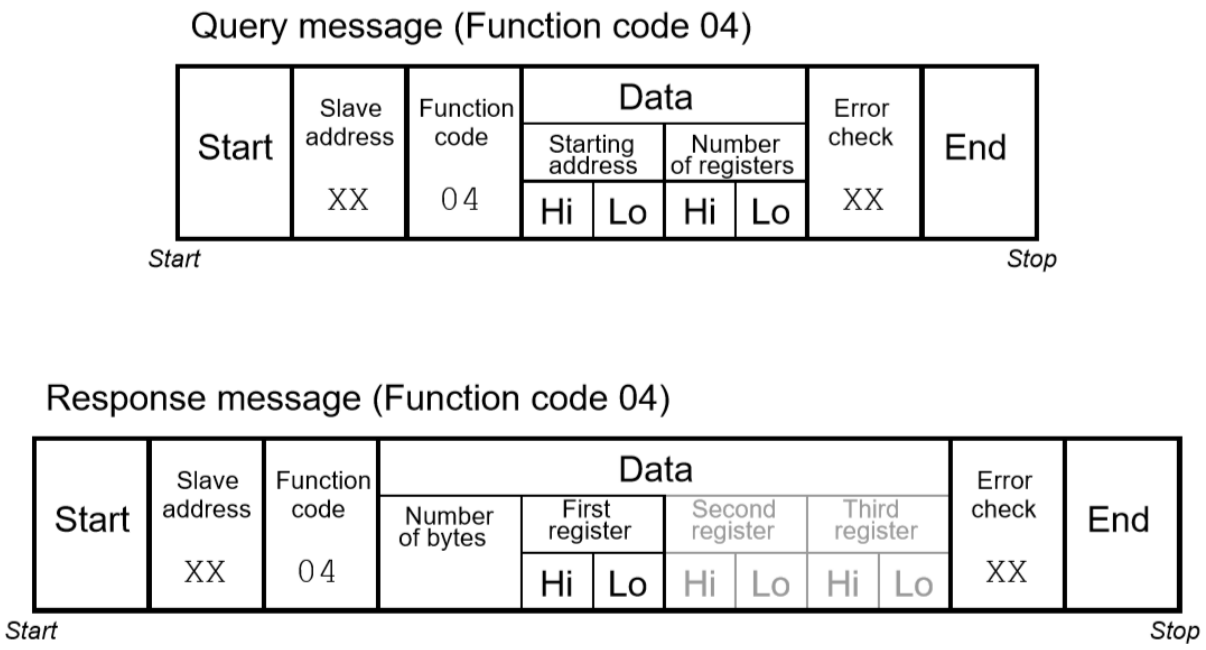


Figure II.12 :Trame de requête et réponse de fonction 04

Notez que puisque le message de query spécifie le nombre de registres (chaque registre ayant une taille de deux octets), et que le message de réponse répond avec le nombre d'octets, le field "nombre d'octets" du message de réponse aura une valeur double de celle du field "nombre de registres" du message de requête. Notez également que le nombre maximal de registres qui peuvent être demandés dans le message de query (65536) avec des valeurs d'octets "high" et "low" dépasse largement le nombre d'octets que le message de réponse peut indiquer (255) avec sa valeur d'un seul octet.

➢ **Code de fonction 05 – Write (Force) Single Coil**

Cette fonction Modbus écrit un seul bit de données sur une sortie discrète ("coil") dans le dispositif esclave. Les adresses Modbus pertinentes pour cette fonction vont de 00001 à 09999 (décimal) mais l'adresse de départ est un nombre hexadécimal représentant le (n-1)ème registre à partir du début de cette plage (par exemple, l'adresse décimale 07200 serait spécifiée comme hexadécimale 1C 1F).

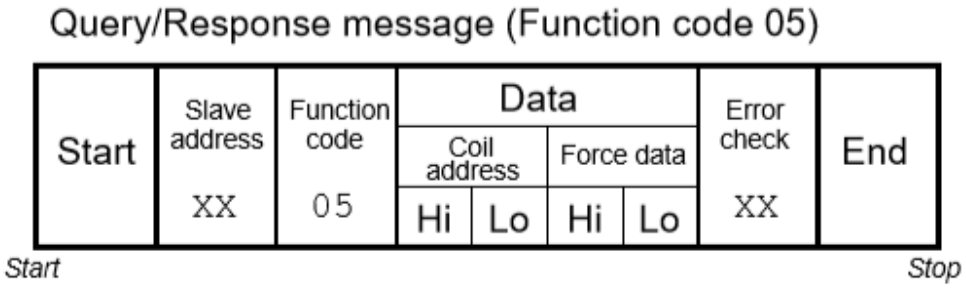


Figure II.13 :Trame de requête et réponse de fonction 05

Les "force data" pour une coil unique consistent en 00 00 (force coil off) ou FF 00 (force coil on). Aucune autre valeur de données ne suffice - toute valeur autre que 00 00 ou FF 00 sera ignorée par le dispositif esclave. Un message de réponse normal sera un simple écho (répétition mot à mot) du message de query.

➢ **Code de fonction 06 – Write (Preset) Single Holding Register**

Cette fonction Modbus écrit des données dans un seul registre "holding" dans le dispositif esclave. Les adresses Modbus pertinentes pour cette fonction vont de 40001 à 49999 (décimales) mais l'adresse de départ est un nombre hexadécimal représentant le (n-1)ème registre à partir du début de cette plage (par exemple, l'adresse décimale 40034 serait spécifiée comme hexadécimale 00 21).

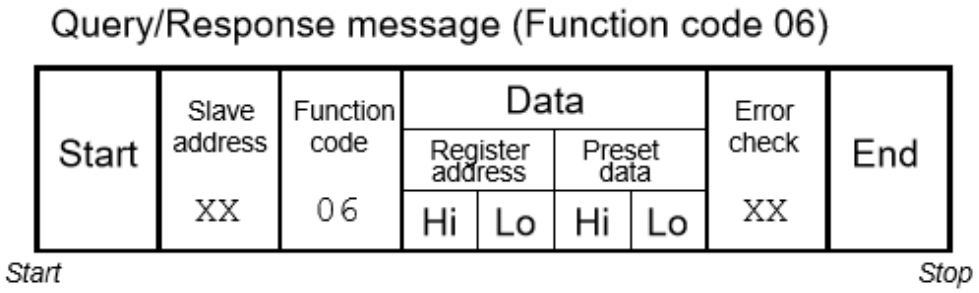


Figure II.14 :Trame de requête et réponse de fonction 06

Un message de réponse normal sera un simple écho (répétition mot à mot) du message de query.

➢ **Code de fonction 15 – Write (Force) Multiple Coils**

Cette fonction Modbus écrit plusieurs bits de données sur un ensemble de sorties discrètes ("coils") dans le dispositif esclave. Les adresses Modbus pertinentes pour cette

fonction vont de 00001 à 09999 (décimales) mais l'adresse de départ est un nombre hexadécimal représentant le (n-1)ème registre à partir du début de cette plage (par exemple, l'adresse décimale 03207 serait spécifiée comme hexadécimale 0C 86).

Query message (Function code 15)

Start	Slave address	Function code	Data								Error check	End
			Starting address		Number of coils		Number of bytes	Force data first word		Force data second word		
			Hi	Lo	Hi	Lo		Hi	Lo	Hi	Lo	
	XX	0F									XX	

Start — Stop

Response message (Function code 15)

Start	Slave address	Function code	Data				Error check	End
			Starting address		Number of coils			
			Hi	Lo	Hi	Lo		
	XX	0F					XX	

Start — Stop

Figure II.15 :Trame de requête et réponse de fonction 15

Notez que le message de query spécifies à la fois le nombre des coils(bits) et le nombre d'octets.

- **Code de fonction 16– Write (Preset) Multiple Holding Register**

Cette fonction Modbus écrit plusieurs mots de données dans un ensemble de registres de "holding" à l'intérieur du dispositif esclave. Les adresses Modbus pertinentes pour cette fonction vont de 40001 à 49999 (décimal) mais l'adresse de départ est un nombre hexadécimal représentant le (n-1)ème registre à partir du début de cette plage (par exemple, l'adresse décimale 47441 serait spécifiée comme hexadécimal 1D 10).

Query message (Function code 16)

Start	Slave address	Function code	Data								Error check	End
			Starting address		Number of registers		Number of bytes	Preset data first register		Preset data second register		
			Hi	Lo	Hi	Lo		Hi	Lo	Hi	Lo	
	XX	10									XX	

Start — Stop

Response message (Function code 16)

Start	Slave address	Function code	Data				Error check	End
			Starting address		Number of registers			
			Hi	Lo	Hi	Lo		
	XX	10					XX	

Start — Stop

Figure II.16 :Trame de requête et réponse de fonction 16

Notez que le message de query spécifie à la fois le nombre de registres (mots de 16 bits) et le nombre d'octets, ce qui est redondant (le nombre d'octets doit toujours être le double du nombre de registres, étant donné que chaque registre a une taille de deux octets). Notez également que le nombre maximal de registres pouvant être demandés dans le message de requête (65536) avec des valeurs d'octets "high" et "low" dépasse largement le nombre d'octets que le message de réponse peut rapporter (255) avec sa valeur d'un seul octet.

II.4 Conclusion

Les applications de communication industrielle disposent de multiples périphériques de communication précurseurs pour connecter les systèmes industriels avec le monde extérieur. Cela a été possible grâce aux nouveaux microprocesseurs qui sont spécialement conçus pour ce type d'applications. Pour répondre aux besoins des utilisateurs, les fournisseurs utilisent les processeurs pour améliorer les fonctions. Ils passent de protocoles de communication propriétaires à des protocoles de normes ouvertes où la technologie de mise en réseau d'autres industries est empruntée. Ce changement est dû à la nécessité d'accroître les communications entre les dispositifs industriels et le monde qui les entoure. Les protocoles de communication tels que TCP/IP, Controller Area Network (CAN) et Universal Serial Bus (USB) sont utilisés par ces entreprises. Ces protocoles s'avèrent également très nécessaires pour la conception industrielle et sont les mieux adaptés à une utilisation spécifique. Les interfaces de communication les plus simples pour les équipements industriels sont les interfaces série comme RS-232, RS-422 et RS-485, qui sont utilisées par les serveurs, les contrôleurs, les actionneurs, les capteurs simples, les valves et pour la communication.

Chapitre 03 :

Implémentation et test

III.1 Introduction

Dans ce chapitre, nous allons présenter les différent matériels utilise, langage de programmation ainsi que les différents tests et implémentations qui a nous aidons pour réaliser notre solution qui est représenté dans la conception d'une passerelle de conversion qui assurer la communication entre des équipements de contrôle industriels utilisant le protocole non routable Modbus-RTU avec ceux qui utilisant le protocole routable Modbus TCP.

III.2 Approche proposée :

Aujourd'hui, le protocole Modbus TCP/IP est largement utilisé par la plupart des Équipement industriel. Il s'agit d'une variante modbus étendue de l'ancienne version Protocole Modbus RTU. Contrairement à ce dernier, le Modbus TCP permet à un serveur (esclave) de communiquer avec plusieurs clients (maîtres) ce qui veut dire que il est possible de créer un type client-serveur d'un réseaux Ethernet industriel suffisamment grand et facile à mettre en œuvre et offre une vitesse très rapide avec protection au cours d'échange des données. Par la faveur de ses avantages qui est plus efficace et pour garder l'interopérabilité entre les anciens et les nouveaux équipements Il doit y avoir une passerelle de communication industrielle qui est utilisées pour faire communiquer des périphériques séries Modbus RTU avec des équipements Ethernet Modbus TCP. Cela permet entre autre d'augmenter les distances de connexion grâce à l'Ethernet et de connecter des périphériques séries vers une infrastructure réseau déjà existante

Bien que certains produits fassent bien à cet objectif comme le moxa M-gate qu'on adéjà vu dans l'industrie,notre thème sert à fournir un Solution évolutive et à moindre coût que nous avons proposons de l'implémenter sur une carte Microcontrôleurs industriels, peu coûteux et compatibles avec une variété des systèmes d'exploitation et peut être développé à savoir la carte Raspberry.

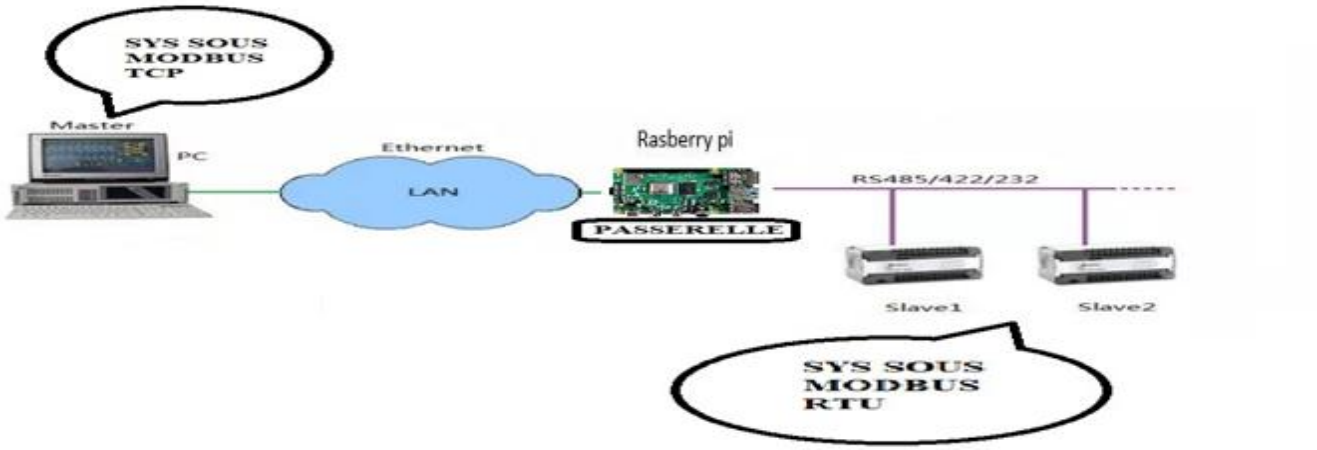


Figure III.1: Schéma de conception générale de la passerelle.

Notre travail comprend le développement et la programmation de la carte raspberry de telle manière que on avoir une passerelle qui assure la Communication entre deux systèmes de contrôle industriels fonctionnant avec deux versions différentes du protocole Modbus le premier system est sous Modbus TCPfonctionné sur le mode client-serveur c'est la version modbus routable de la nouvelle génération le deuxième system est sous modbus RTU la version ancienne de modbus fonctionné sur le mode maître-esclave.

III.3 Présentation du matériel utilisé :

III.3.1 La Raspberry Pi :

Nous choisissons comme microcontrôleur la carte raspberry pi qui peut nous fournir les bons outils de développement pour aboutir à la réalisation de notre passerelle car il permet l'exécution de plusieurs variantes du système d'exploitation libre et représente un véritable pc industriel avec des sorties industrielles.

Le Raspberry Pi fait partie de la famille des « Nano-Ordinateurs », des machines ayant en gros les dimensions d'une carte de crédit voire moins, et il est dit « monocarte » ou machine de type « System On a Chip », ou tous les éléments sont présents sur une seule carte de support.

Cet ordinateur, qui a la taille d'une carte de crédit, est conçu pour favoriser l'apprentissage de la programmation informatique, il permet l'exécution de plusieurs variantes du système d'exploitation libre ?GNU/Linux, notamment Debian, mais il fonctionne aussi avec une version Spécifique au système d'exploitation Microsoft Windows (le tout sur carte Micro). Sa conception permet d'ajouter très facilement les éléments de base pour le faire fonctionner : clavier et souris, écran avec connexion HDMI, chargeur de téléphone via Micro USB et carte SD pour le système d'exploitation. Ce design minimaliste permet de le produire à très faible coût, permettant au plus grand nombre d'utilisateurs de s'équiper. Il peut aussi être utilisé pour la programmation et le développement L'un des systèmes d'exploitation principal est Raspian, une version de Linux Debian spécifique pour cette plateforme aussi comme un Serveur personnel, bloqueur de pub par DNS interne et même pour la Gestion d'éléments divers dans une maison : chauffage, éclairage, relevé de température et humidité, contrôle des ouvertures.

Comme illustré par la Figure 2, le Raspberry Pi 4 est constitué des composants suivants :

1. Processeur intégré Broadcom BCM2711, quad-core Cortex-A72 (ARM v8) 64-bit 1.5GHz.
2. Lecteur de cartes Micro SD: pour le chargement du système d'exploitation et stockage de données.
3. 2 × micro HDMI (High Definition Multimedia Interface) ports: supporte jusqu'à 4kp60 permet de relier le RaspberryPI à un dispositif compatible : écran LCD ou un vidéoprojecteur.
4. 4 Ports USB (Universal Serial Bus): 2 ports USB 3.0 2 ports USB 2.0 est un port série qui sert à connecter le Raspberry aux autres périphériques.
5. port réseau fast Ethernet
6. Jack 3.5 mm, vidéo composite RCA (via jack 3.5mm).
7. 40 broches GPIO.

8. Alimentation : tension 5V, courant minimal 750mA.
9. Bus caméra CSI.
10. Bus afficheur LCD.



Figure III.2 : Les composants de la carte Raspberry Pi.

III.3.2 Vnc viewer :

Nous avons utilisé pour l'affichage de la carte raspberry le virtuel vnc viewer qui peut connecte à la carte raspberry via ca adresse IP comme figure 3 nous illustrons tant que les deux sans connecte au même wifi et ça pour éviter n'importe quelle câblage inutile

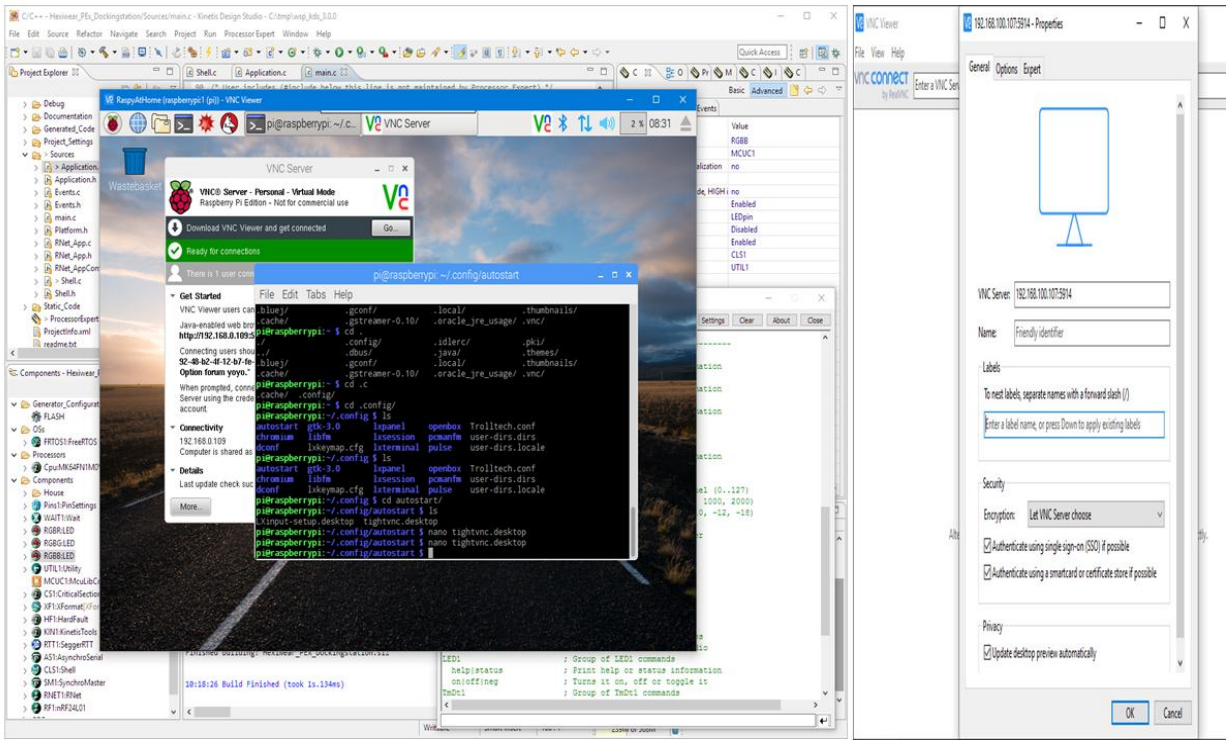


Figure III.3 :connexion et affichage de la raspberry pi avec vnc viewer

VNC (Virtual Network Computing) est un logiciel distribué gratuitement par les laboratoires de recherche d'AT&T. Il s'agit d'une application client/serveur permettant, à partir de sa machine ("client") de prendre aisément le contrôle total d'un ordinateur quel que soit son emplacement ("serveur") via un réseau local ou encore Internet (TCP/IP). Ce programme s'avère très pratique par exemple, pour le travail à domicile, l'administration de serveurs distants, ou encore en situation d'apprentissage dans une salle informatique. Les connexions sont protégées par mot de passe et peuvent être sécurisée grâce au protocole SSH (scure shell). Vnc offre un service d'une simplicité trompeuse, il vous permet de visualiser et de contrôler un système distant. L'application compacte 'VNC Server 4' s'exécute sur le système à contrôler. IL s'adapte automatiquement et dynamiquement à des conditions variables, y compris des contenus d'écran et des bandes passantes réseau différents. Le logiciel s'appelle désormais REALVNC.

RealVNC est l'un des principaux fournisseurs de solutions d'accès à distance, aidant certaines des plus grandes entreprises au monde à connecter des personnes via des appareils. En tant que société pionnière du protocole Internet VNC qui est maintenant la norme dans l'industrie, RealVNC est à l'avant-garde de son évolution, avec des logiciels dans plus de 1 milliard d'appareils dans le monde. Le logiciel d'accès à distance sécurisé de RealVNC, VNC Connect, est approuvé par les poids lourds de l'industrie, y compris des organisations de premier plan telles que Google, la NASA et Intel.

III.4 Présentation des langages de programmation utilisés

On a utilisé premièrement le langage python pour la programmation de code et même pour l'implémentation dans le raspberry pi car il est compatible avec ce dernier ensuite en a utilisé tkinter pour la réalisation des interfaces graphique.

III.4.1 Python :

Le langage de programmation Python a été créé en 1989 par Guido van Rossum, aux Pays-Bas. Le nom Python vient d'un hommage à la série télévisée Monty Python's Flying Circus dont G. van Rossum est fan. La première version publique de ce langage a été publiée en 1991.

Python est le langage de programmation open source le plus couramment utilisé par les informaticiens. Ce langage est propulsé au sommet dans les domaines de la gestion d'infrastructures, de l'analyse de données ou du développement de logiciels. Nous avons choisi ce langage de programmation car il possède de nombreuses fonctionnalités intéressantes qui nous permettent de décharger plus facilement les tâches parmi ces fonctionnalités : Il est multiplateforme C'est-à-dire qu'il fonctionne sur de nombreux systèmes d'exploitation :Windows, Linux, Android… ce qui nous a aider car on a travaillé dans deux diffèrent plateforme(raspberry system et Windows), Il est gratuit, Un script Python n'a pas besoin d'être compilé pour être exécuté contrairement à des langages comme le C ou le C++. Enfin, Il est orienté objet, C'est-à-dire qu'il est possible de concevoir en Python des entités qui fonctionne en temps réel avec un certain nombre de règles de fonctionnement et d'interactions ainsi que ce langage servira surtout pour le Scripting et l'automatisation.

III.4.2 Tkinter :

La bibliothèque Tkinter est dérivée du langage de programmation TCL. Ce langage a une extension appelée Tk qui permet la création d'interfaces graphiques. Tkinter est une binding des extensions Tk. Soit dit en passant, Tkinter signifie TK interface. Python n'est

pas le seul langage ayant réalisé un binding de Tk ; c'est aussi le cas des langages OCaml, Perl et Ruby.

Lorsque nous utilisons une interface graphique (par exemple, un traitement de texte), le programme attend que nous intervenions en écrivant quelque chose ou en cliquant sur un bouton. Les interventions auxquelles nous faisons réagir le programme sont les soi-disant événements. La surveillance des événements (parfois appelée écoute) est effectuée par un programme appelé boucle d'événements car il s'agit d'une boucle infinie qui recherche les événements et y réagit. Ce type de programmation est appelé programmation événementielle ou asynchrone.

Tkinter prend en charge une gamme de versions Tcl/Tk, construites avec ou sans prise en charge des threads. La version binaire officielle de Python regroupe les threads Tcl/Tk 8.6, le package tkinter ("interface Tk") est l'interface Python standard de la boîte à outils de l'interface graphique Tcl/Tk. Tk et tkinter sont disponibles sur la plupart des plates-formes Unix, y compris macOS, ainsi que sur les systèmes Windows.

III.5 Réalisation :

Dans notre travail, à partir de l'architecture serveur /client on a construire le modèle de fonctionnement de la communication modbus et nous implémenté les composant nécessaire. C'est-à-dire qu'un client Modbus TCP existe au niveau du réseau modbus TCP/IP (maitre) et RTU esclaves (slave) et Gateway (raspberry) permet la communication entre les deux modes. Comme était illustré dans la figure III.4

Le modèle de fonctionnement que nous suivons Comme indiqué ci-dessous

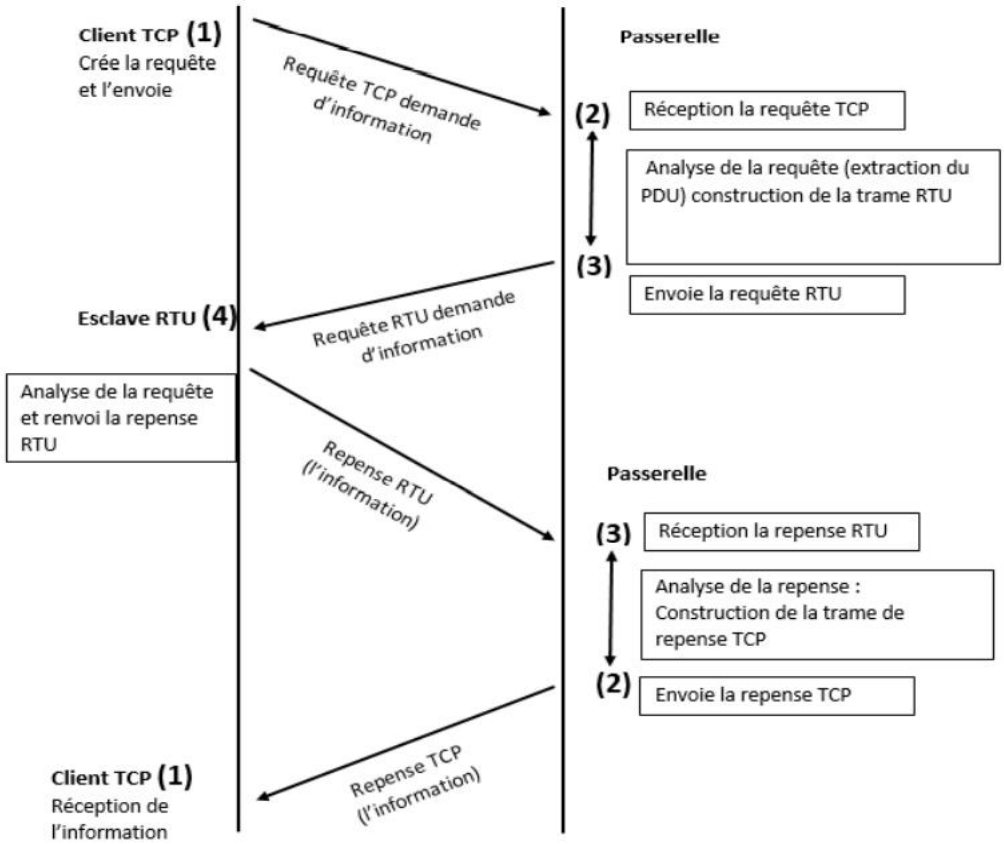


Figure III.4 : mode de fonctionnement.

III.5.1 Réalisation d'un chat room :

Pour faciliter la tâche de la réalisation de passerelles et de l'envoie et recevoir des données par un programme python on a proposé un code plus simple et efficace pour faire une base sur laquelle on va construire le reste de notre travail ce code est un chat room qui nous a programmé par python ce dernier fonctionne avec une architecture client-serveur, telle que l'ensemble des clients connecté au serveur qui permis la connections entre ses derniers.

- **Tache 1:**creation de socket

Un socket est une méthode de communication entre le client et le Serveur dans un réseau où il existe de nombreuses autres méthodes de communication entre le client et le serveur mais socket est jusqu'à présent la méthode de communication la plus populaire Maintenant, un socket est essentiellement une adresse d'une machine, par exemple, le serveur lie son socket avec une adresse pour que différents clients puissent l'identifier De toute évidence, vous avez besoin d'une adresse d'un endroit pour y'accéder Et maintenant, c'est pareil avec Socket. Il est composé de deux parties, **l'adresse IP** pour l'identification du machine et **le numéro de port** pour identifie dans quelle partie de la machine il doit y 'aller.

Tout d'abord, nous créons un serveur et un client en Python afin que vous ayez une idée génétique du fonctionnement d'un chat room. L'idée de base est que le serveur contiendra des informations, Il sera en cours d'exécution et continuera à écouter la demande du client, avant que le client puisse demander quelque chose, le serveur doit se trouver à l'adresse suivante afin qu'il puisse recevoir et traiter la demande du client. Le client enverra la demande désactivée, le serveur le reçoit, le traite et envoie les informations demandées au client.

Pour cela En a utilisé la commande serveur = socket.socket (socket.AF_INET, socket.SOCK_STREAM) qui fait la création d'une instance socket Les arguments passés à socket () sont des constantes utilisées pour spécifier la famille d'adresses et le type de socket. Le premier paramètre est **AF_INET**qui signifie la famille d'adresses Internet pour IPv4 le deuxième est **SOCK_STREAM** le type de socket pour TCP, le protocole qui sera utilisé pour transporter les messages sur le réseau.

Ensuite Pour faire la connections entre le serveur et le client on a travaillé avec la commandeserver.bind((IP,PORT))
IP=adresse IP de serveur ('127.0.0.1'), port= port de connexion ('9090')

La méthode bind () de la classe socket de Python attribue une adresse IP et un numéro de port à une instance de socket. La méthode bind () est utilisée lorsqu'un socket doit être créée comme un serveur. Et l'adresse IP à attribuer explicitement à un socket serveur.

- **Tache 2 :** Réalisation de serveur, client et la connexion entre eu

Pour la création de serveur on a utilisé de nombreux fonctions et commandes parmi eu

La commande Sock.listen () qui permit au serveur d'écouter la demande des clients

La fonction Accept_client_connections () qu'en utilisons à l'intérieur la commande sock.accept () qui permit l'acceptation de la connexion de client

Les règles de traduction d'une chaîne Unicode en une séquence d'octets sont appelées encoding de caractères ou simplement encoding,pour cela on a utilisé la fonction encode ('utf8')

Quand le client se connecte le serveur le attribué un port de communication et affiche un message que la connexion est établis avec (client_ip_adresse,port_numéro) et le stocke dans une liste des clients (clients={}) comme illustre dans la figure 5

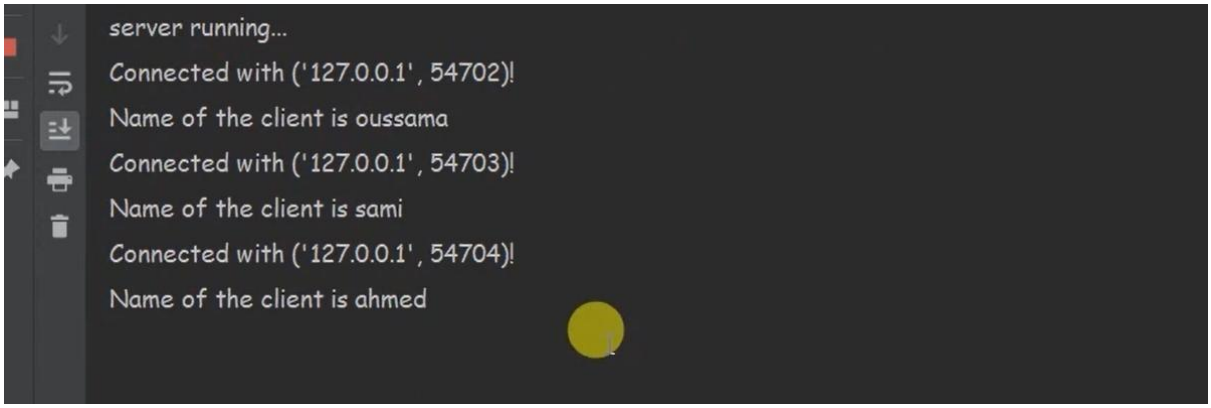


Figure III.5 :connexion de client au serveur.

Pour la création de client on a d'abord tout, utilise la librairie de TKinter import TKinter

Qui a nous permit de crée l'interface graphique comme présente dans la figure 6.

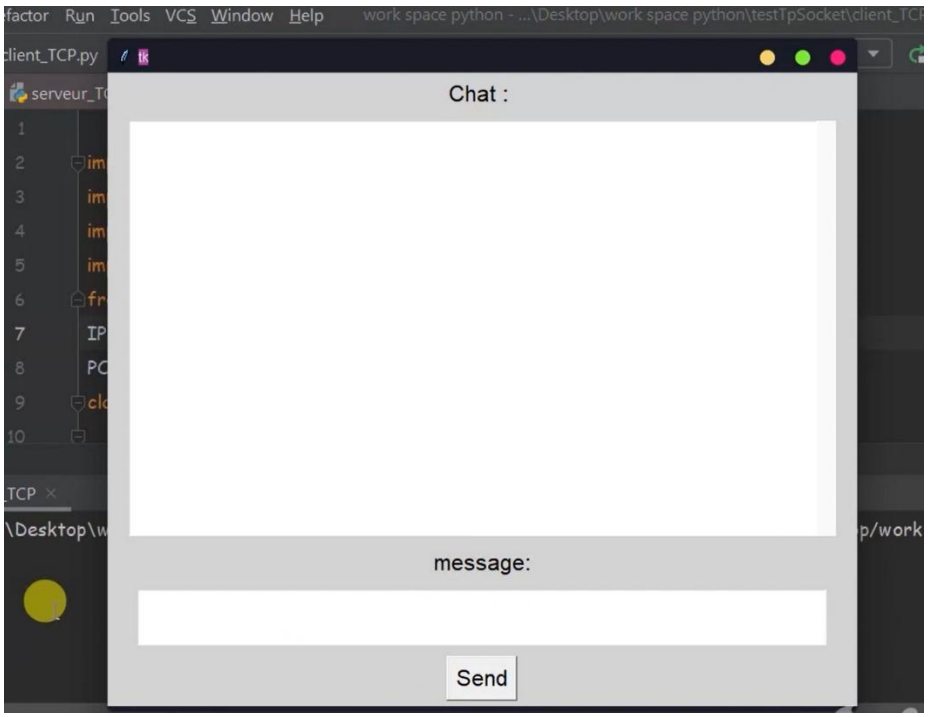


Figure III.6 : interface graphique de client crée par TKinter.

Ensuite on a définie deux fonction send () and receive () qui permit l'envoi et recevoir des messages entre les clients ou avec le serveur.

On a utilisé dans La fonction receive () la commande de la class socket s.receive (1024).decode ("utf8") telle que 1024 signifieNombre d'octets à recevoir. Cette fonction Renvoie les données reçues en tant qu'objet bytes c'est pour cela on a utilisé la fonction decode ("utf8") pour le décodage en caractère.

Et dans La fonction send () on a travaillé encore avec la commande s.send (bytes (msg,"utf8") de class socket qui va envoyer au serveur le message en forme de bytes et ce dernier va le distribuer aux autres clients.

- **Tache 3 :** La distribution des données

Dans cette tache l'architecture de serveur client nous a obligés à programmer le serveur pour distribuer les messages au client pour cet effet on a définir la fonction broadcast () dans laquelle on a utilisé un boucle qui nous permis de parcourir les éléments de la liste clients en utilisant une boucle for client in clients et la fonction client.send(message) qui faire l'envoi des messages enfin en fait l'appel à chaque fois dans le code de serveur à la fonction broadcast()

La figure 7 nous montrant un chat room final entre 3 clients

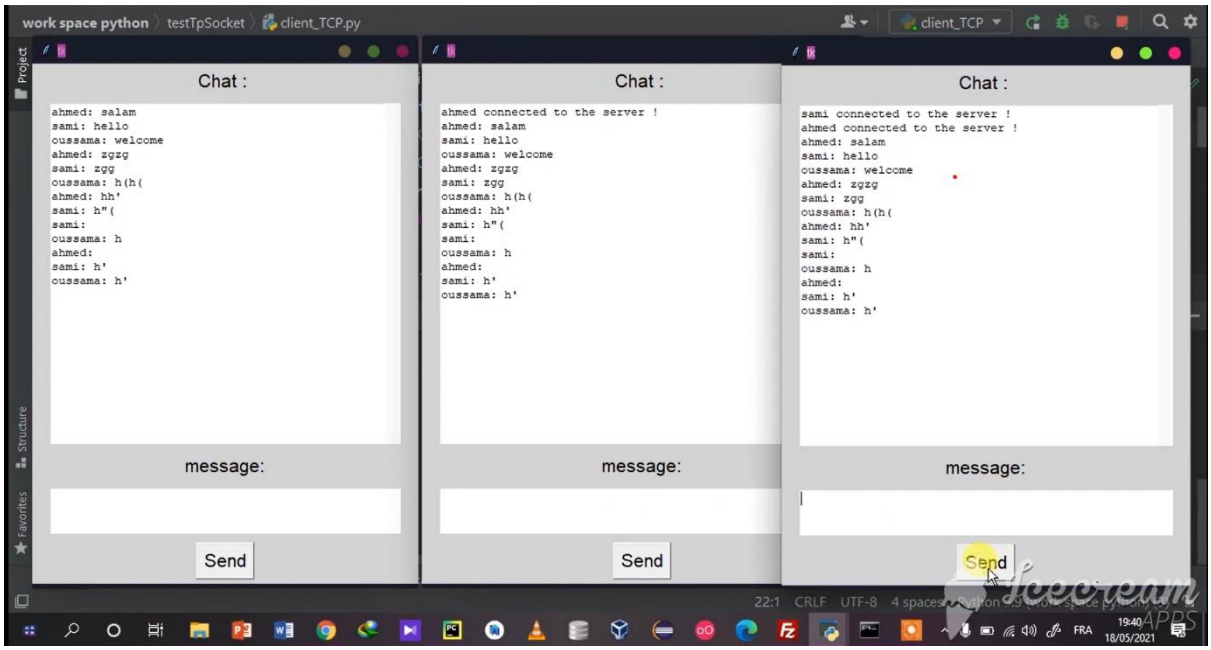


Figure III.7: chat room entre 3 clients.

III.5.2 Réalisation de passerelle

La réalisation de la passerelle comme il montre le schéma de conception dans la figure 1 est fait dans 3 phases

III.5.2.1 Premier phase :

Dans cette phase on va créer la communication de system sous Modbus Tcp avec notre passerelle

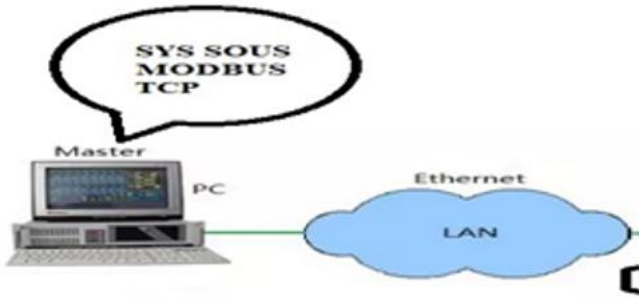


Figure III.8 :*sys sous Modbus TCP*.

Pour cet effet on va baser sur le travail de notre code de chat room car pour le protocole TCP elle est presque le même donc on a fait :

- Création de socket :

On a utilisé la commande socket.socket (socket.AF_INET, socket.SOCK_STREAM) déjà définie pour crée le socket, Mais maintenant en va déterminer les paramètres nécessaire pour la connections pour cela on à affecter à la commande sock.connect ('192.168.58.1',502) l'adresse de client TCP et le port 502 Par défaut, le protocole utilise ce port comme port local dans le serveur Modbus.

- Création de la trame modbus tcp :

La trame de requête se compose des champs PDU et en-tête TCP (MBAP header)

Le protocole d'application MODBUS définit une simple unité de données de protocole (PDU) indépendante des couches de communication sous-jacentes. Le mappage du protocole MODBUS sur un bus ou un réseau spécifique introduit des champs supplémentaires sur l'unité de données de protocole. Le client qui initie une transaction MODBUS construit la PDU MODBUS, puis ajoute des champs afin de construire la transaction appropriée.

Pour la génération de PDU on a utilisé la fonction qui est l'implémentation de librairie umodbus de la fonction modbus 15(0x10) WriteMultipleRegisters.request_pdu Ce code de fonction est utilisé pour écrire un bloc de registres contigus (1 à 123 registres) dans un équipement distant. Les valeurs écrites demandées sont spécifiées dans le champ de données de requête. Les données sont compressées sous forme de deux octets par registre. La réponse normale renvoie nous retourner le PDU spécifié du code fonction après ça la création de mbap header ce définit par le même code fonction est on l'ajouter au pdu pour construire la trame TCP

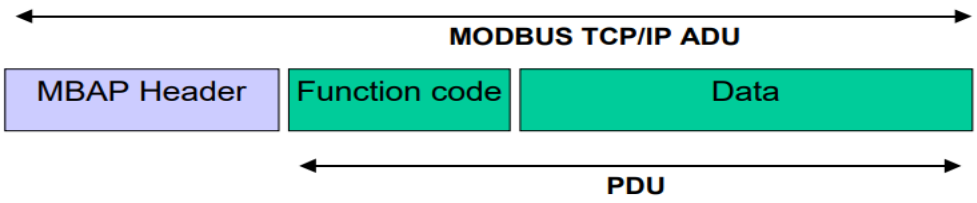


Figure III.9: Trame Modbus TCP

- L’envoi de la trame :

On a le modbus tcp/ip ADU est prêt pour être envoyer au serveur (slave) on a utilisé dans ce cas le même fonctionnement de la fonction broadcast() déjà définie dans laquelle on a travaillé avec sock.send() pour l’envoi au serveur et sock.recieve() pour le recevoir.

Remarque : la réponse depend de code fonction modbus

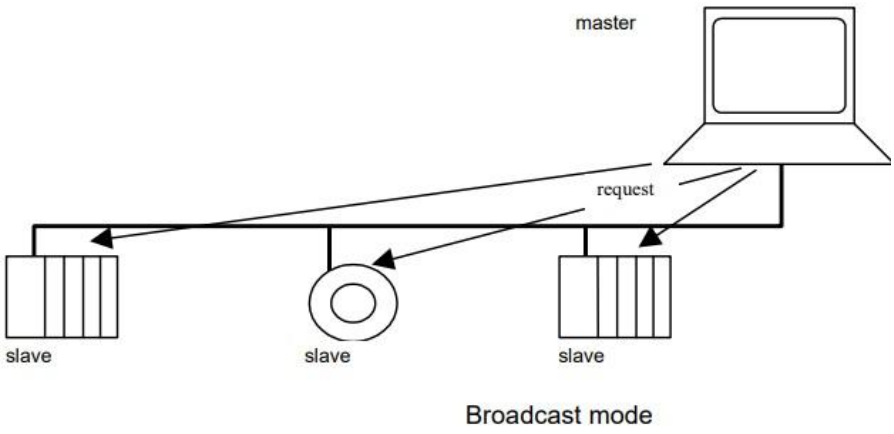


Figure III.10 :mode de fonctionnement de broadcast.

III.5.2.2 Deuxième phase :

Dans cette phase en va créer l’esclave(ou l’ensemble) MODBUS RTU

Pour l’atténuation a cette tache on va utiliser la fonction s=Serial ("/dev/ttyS1", baudrate=115200, timeout=3.0) pour connecte au port série et get_server (RTUserver,s) pour connecté le port série avec le serveur RTU, qui est l'implémentation de la fonction modbus rtu dans la bibliothèque uMODBUS.

III.5.2.3 Troisième phase :

La Réalisation de la passerellefait en deux étapes

- **Etape 1 :**

Comme la figure 11 nous a montré la différence entre la trame modbus TCP et modbus RTU en a eu l’idée de fonctionnement de passerelle donc pour convertir la trame TCP en

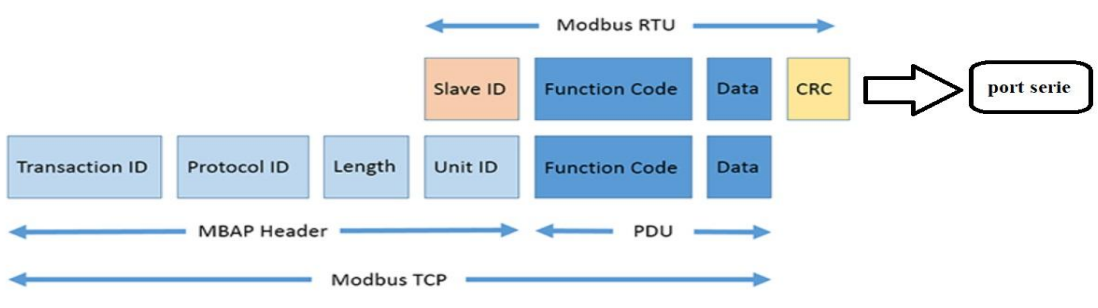


Figure III.11 : la différence entre la trame TCP et RTU

.

RTU en doit prend le MBAP header pour calculez le CRC et ajoutez-le à la fin du trame et enlever l'entête TCP carrément de cette manier en avoir une trame RTU,Enfin envoyer cette trame sur le port série.

- **Etape 2 :**

C'est exactement l'opposé de l'étape 1 maintenant l'esclave RTU nous a transmis une requête de réponse cette trame de la requête inclut un CRC la passerelle fait la conversion en TCP en supprimer le CRC et ajouter le mbap header au début de tram puis l'envoyer encore au client TCP le même mode de fonctionnement illustre dans la figure 4 au-dessus.

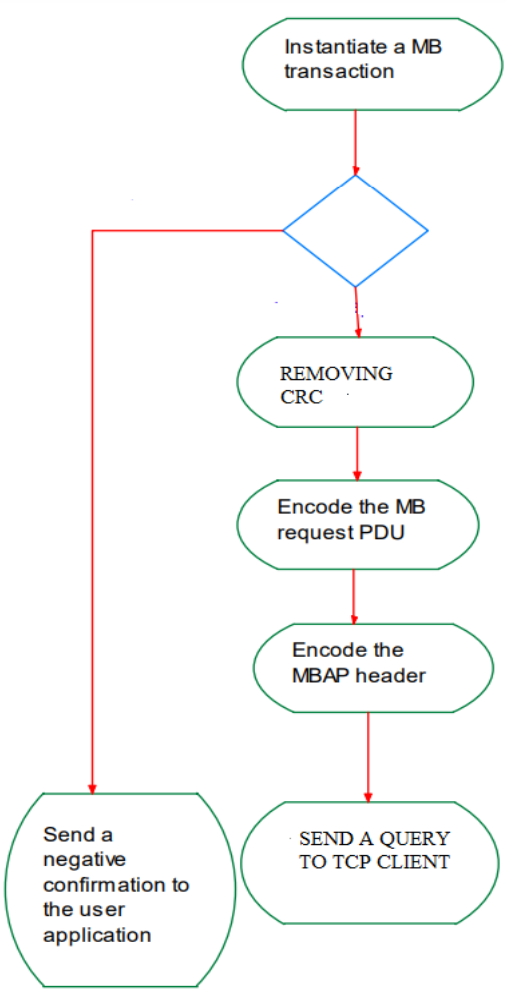


Figure III.12 :organigramme de l'étape 2.

III.6 Résultat de simulation :

On a fait l'implémentation de code de passerelle au raspberry pi et utiliser des Virtual pour simuler les trames de TCP et RTU

III.6.1 test de requêtes TCP:

On a construire une interface graphique qui a nous permis d'entre la valeur de code fonction en parallèle avec les donnée qu'on va l'envoyer donc dans le client tcp dans ce test on a entré le code fonction 15 et des donnée numéros quelle quand [0 7 1 5 3 19 6 5] Donc la trame TCP va être crée à base de le code fonction insérer, la passerelle va recevez les donnée en TCP faire le même fonctionnement de l'étape 1 enlever le mbap header et calculer le CRC puis l'ajouter a la fin pour faire la conversion et envoyer la réponse RTU comme la figure 13 nous a illustré.

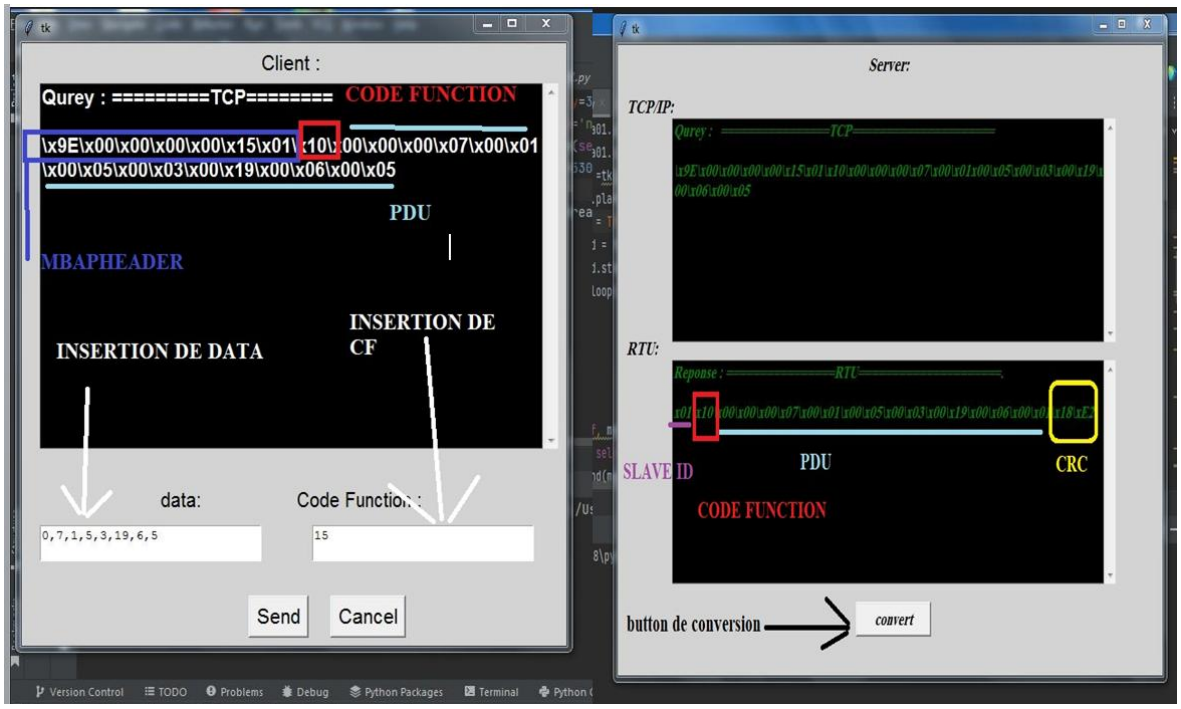


Figure III.13 :test 1 de conversion de TCP.

III.6.2 Elément des trames:

- Mbap header :

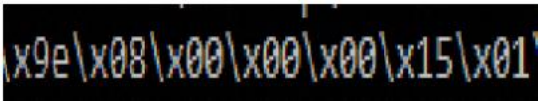
C'est Un header dédié est utilisé sur TCP/IP pour identifier l'unité de données d'application MODBUS (MODBUS Application Protocol header). Ce header présente certaines différences par rapport au RTU MODBUS.

Toutes les requêtes et réponses MODBUS sont conçues de manière à ce que le destinataire puisse vérifier qu'un message est terminé Pour les codes de fonction où le PDU MODBUS a une longueur fixe, le code de fonction seul est suffisant. Pour les codes de fonction contenant une quantité variable de données dans la requête ou la réponse, le champ de données (data field)

Fields	Length	Description -	Client	Server
Transaction Identifier	2 Bytes	Identification of a MODBUS Request / Response transaction.	Initialized by the client	Recopied by the server from the received request
Protocol Identifier	2 Bytes	0 = MODBUS protocol	Initialized by the client	Recopied by the server from the received request
Length	2 Bytes	Number of following bytes	Initialized by the client (request)	Initialized by the server (Response)
Unit Identifier	1 Byte	Identification of a remote slave connected on a serial line or on other buses.	Initialized by the client	Recopied by the server from the received request

Figure III.14 :description de mbap header.

Donc on a eu d'après le test les valeurs suivant qu'on va l'identifier



- Transaction Identifier : 9E O8 Il est utilisé pour l'appariement des transactions.
- Protocol Identifier : 00 00 ilest utilisé pour le multiplexage intra-système (identification de protocol).
- Length : 00 15 Le champ de longueur est un nombre d'octets des champs suivants, y compris l'identificateur d'unité et les champs de données.
- Unit identifier : 01 Ce champ est défini par le client MODBUS dans la requête et doit être renvoyé avec la même valeur dans la réponse par le serveur.

➢ PDU : [0 7 1 5 3 19 6 5]

Quand on a déjà définit alors il représente l'ensemble des donnée dans ce test tant que Le mappage du protocole MODBUS sur des bus ou des réseaux spécifiques peut introduire des champs supplémentaires sur l'unité de données d'application (ADU).

➢ Code function : 15(x10)

Le code fonction indique à l'esclave le type d'action à réaliser. Dans notre cas : lecture et écriture de registre, code de fonction 15. Le champ de données est codé sur n octets.

➢ CRC : x18 xe2

Le champ contrôle d'erreur CRC (Cyclical Redundancy Check) sert à s'assurer de l'intégrité des données reçues. Le CRC contient une valeur codée sur 16 bits.

III.6.3 réponse :

Ensuite la passerelle va envoyer les requêtes de réponses donc il va recevoir une trame RTU avec le même CRC calcule dans le premier sens et enlever le crc puis ajouter le mbapheader au premier de tram pour avoir la réponse TCP qui va l'envoyer au client comme la figure 15 nous a montré.

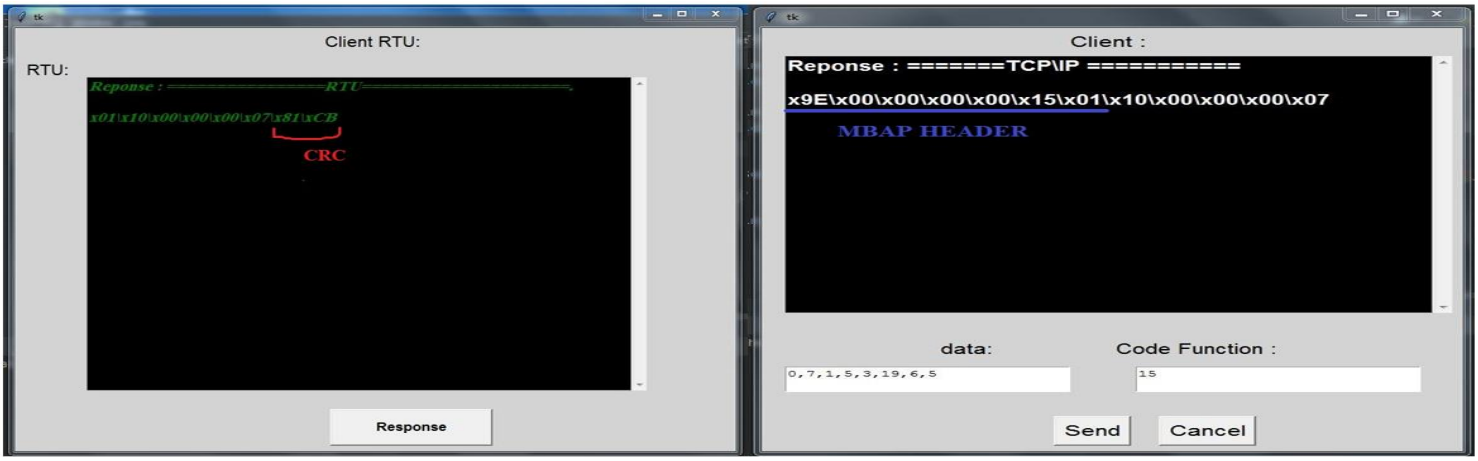


Figure III.15 :conversion de RTU.

La réponse RTU contient :

PDU : qui inclus l'adresse de l'esclave x01,le code fonction 15 x10 et la donnée envoyer qui est 07

CRC : x81 xcb

Tant que la réponse TCP qui a été créé contient :

MBAP header : inclut Transaction Identifier 9E 08, Protocol Identifier : 00 00(modbus TCP), Length : 00 15 =15octet et l'adresse de l'esclave x01

III.7 CONCLUSION :

Dans ce chapitre, nous décrivons les aspects pratiques de notre projet. Tout d'abord, nous commençons par poser le problème, puis proposons une solution, qui consiste à installer une passerelle qui fournit une communication RTU/TCP. Avant de commencer la mise en œuvre, nous décrivons brièvement le microcontrôleur et le logiciel utilisé ainsi que le langage de programmation. Nous avons divisé la mise en œuvre du projet premièrement en deux tâche la premier tache nous a aidée à être familiariser par les différents fonctions et commandes et avec le mode de fonctionnement la deuxième tâche est divise en trois phases. Nous expliquons d'abord les étapes de programmation du client TCP. Nous avons ensuite procédé à la programmation du RTU esclave et enfin nous avons mis en place une passerelle pour assurer la communication entre les deux premières parties.

CONCLUSION GENERALE

Dans le premier chapitre, les difficultés de liaison entre MODBUS, RTU et TCP sont définies Les principaux avantages de l'établissement de ce lien sont :

D’une part permettre d'augmenter la durée de vie du système utilisant le protocole RTU, de réduire les problèmes techniques de câblage et faire une liaison au industrie par le domaine de développement et de la programmation.

Dans le cadre de ce projet, nous avons développé une passerelle de communication Modbus TCP/RTU, que nous proposons d'implémenter sur une carte à microcontrôleur low-cost : Raspberry PI 4.

Ce projet nous a permis d'enrichir nos connaissances, d'affiner nos acquis et d'acquérir de l'expérience, tant au niveau de la théorie de l'apprentissage de nouveaux concepts dans des domaines auxquels nous n'avions jamais été exposés auparavant, qu'au niveau de la programmation que nous peut gérer. Apprenez le langage python en quelques mois, et le maitriser à travers la pratique.

On a familiarisé avec la carte microcontrôleur qui est clairement la future de domaine électronique et les domaines de développement.

Les perspectives de développement de notre projet seront orientées autour des axes suivants :

- surpasser les problèmes confrontés et intégrer plus de fonctionnalités en exploitant la puissance de la carte à microcontrôleur.

- Appliqué la méthode sur des différent protocoles industrielle comme : CAN, Profinet et Profibus.

Chapitre 1

Page référence

[1]: Daniel RACOCEANU « Contribution à la surveillance des Systèmes de Production en utilisant les Techniques de l'Intelligence Artificielle », Thèse d'habilitation, Université de FRANCHE-COMTÉ de Besançon, 2006, 114 p.

[2] : Gilles Zwingeistein, « Diagnostic des défaillances, théorie et pratique pour les systèmes industriels» édition Hermès, 1995, pp. 55-13.

[3] : Claudia VICTORIA ISAZA NARVAEZ, « Diagnostic par techniques d'apprentissage floues : conception D'une méthode de validation et d'optimisation des partitions », thèse doctorat, Institut National des Sciences Appliquées de Toulouse, 2007, 178 p.

[4]: Daniels, W.Salter, "WHAT IS SCADA?" International Conference on Accelerator and Large Experimental Physics Control Systems, 1999, Italy, pp. 5.

[5]: David Bailey, Edwin Wright, « Practical SCADA for Industry » Elsevier,pages47-55,2003.

[6]: *Boyer, Stuart A. (2010). Contrôle de supervision SCADA et acquisition de données. États-Unis : ISA - Société internationale d'automatisation. p. 179. ISBN 978-1-936007-09-7.*

[7]: *E. A. Parr, Programmable controllers: an engineer's guide (Elsevier Ltd, Oxford, 2003)*

[8]: *Rieger C, Ray I, Zhu Q, Haney M. Industrial Control Systems Security and Resiliency. Springer, 2019. DOI:10.1007/978-3-030-18214-4.*

[9]: *Experion PKS®CEE-based ControllerSpecifications and Technical DataEP03-300-200Release 200Revision Date: December 2003Version 1.0*

[10]:*J. Baranski, M. Crocker, and G. Lazarou, "Fault-tolerant Reconfigurable Ethernet-based IP Network Proxy", proceeding of Communications, Internet, and Information Technology, Nov. 17–19, 2003 Scottsdale, AZ, USA*

[11]:*Experion LX HMIWeb Display Building Guide EXDOC-XX54-en-110A February 2014 Release 110*

[12]: *Dobroskokin A.F., Mezentseva S.A., Haybullayev S.E., 2015. Experion PKS – high technology platform for industrial automation.Automation in industry, №7.*

Chapitre 2

[13] *"422 and 485 Standards Overview and System Configurations" Application Report SLLA070C, Texas Instruments Incorporated, Dallas, TX, 2002.*

[14] *"B&B Converters for the Industrial Bus World" Technical Article 13, B&B Electronics Manufacturing Company, Ottawa, IL, 2000.*

[15] *Floyd, Thomas L., Digital Fundamentals, 6th edition, Prentice-Hall, Inc., Upper Saddle River, NJ, 1997.*

[16] *Graham, Frank D., Audels New Electric Library, Volume IX, Theo. Audel & Co., New York, NY, 1942.HART Communications, Technical Information L452 EN; SAMSON AG, 1999.*

[17] *"FOUNDATION Fieldbus System Engineering Guidelines" (AG 181) Revision 2.0, The Fieldbus Foundation, 2004.*

[18] *"FOUNDATION Specification System Architecture" (FF 581) Revision FS 1.1, The Fieldbus Foundation, 2000.*

[19] *"Modbus Application Protocol Specification", version 1.1b, Modbus-IDA, Modbus Organization, Inc., 2006*

[20] *"Modicon Modbus Protocol Reference Guide", (PI-MBUS-300) revision J, Modicon, Inc. Industrial Automation Systems, North Andover, MA, 1996.*

[21] http://simplymodbus.ca/FC01.htm

Résume :

L’évolution dans les processus industriels et les moyens de production résulte d’un changement dans les protocoles de communication entre des équipements de contrôle. Dans ce contexte, la mise en place des passerelles de communications a introduit un gain important, du fait que ces moyens assurent l’interopérabilité entre les anciens et les nouveaux équipements. Dans le cadre de ce travail, nous proposons de développer une passerelle de communication évolutive. Son rôle est de permettre le transfert des données entres des équipements fonctionnant selon le protocole Modbus TCP et ceux mettant en œuvre l'ancienne version de ce protocole : le Modbus RTU.

Mot clé : Protocoles de communication, Modbus, trame, TCP/IP

Abstract:

The evolution in industrial processes and means of production results from a change in the communication protocols between control equipment. In this context, the establishment of communications gateways has introduced a significant gain, because these means ensure interoperability between old and new equipment. As part of this work, we propose to develop a scalable communication gateway. Its role is to enable the transfer of data between equipment operating according to the Modbus TCP protocol and those implementing the old version of this protocol: Modbus RTU.

Keyword: Communication protocols, Modbus, frame, TCP / IP

الملخص:

التطور في العمليات الصناعية ووسائل الإنتاج ناتج عن تغيير في بروتوكولات الاتصال بين معدات التحكم. في هذا السياق، أدى إنشاء بوابات الاتصالات إلى تحقيق مكاسب كبيرة، لأن هذه الوسائل تضمن إمكانية التشغيل البيني بين المعدات القديمة والجديدة. كجزء من هذا العمل، نقترح تطوير ملف بوابة اتصال قابلة للتطوير. دوره هو السماح بنقل البيانات بين الأجهزة التي تعمل وفقًا للبروتوكول مودباس تي سي بي الحديث وتلك التي تعمل وفق الإصدار القديم من هذا البروتوكول وهو مودباس ار تي او.

كلمات دلالية: Modbus، IP / TCP،بروتوكولات،حزمة بيانات