REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE YAHIA FARES DE MEDEA

Faculté de **T**echnologie

Département du **G**énie **E**lectrique

Mémoire de fin d'études de Master

Filière : **E**lectronique

Spécialité : Electronique des Systèmes Embarqués/ Instrumentation

Remote Supervision and Control of Industrial Programmable Logic Controller using LabVIEW and NI OPC Servers

Présentés par :

Ayache Sidali

Benkiar Wafaa

Proposés et Dirigés par :

Dr.A.ABBADI

Dr.F.Bettache

Promotion 2023

Acknowledgements

First and foremost, we thank Almighty God for granting us good health and the willpower to undertake this humble work. We also express our gratitude to our parents and family for their assistance and patience throughout this journey.

We would like to extend our sincere thanks to our supervisor, Dr. abbadi amel and Dr.Bettache farouk for thier invaluable support and guidance during our study and the completion of this work. they shared thier expertise with us and placed full trust in us.

We would also like to thank all individuals who contributed to the realization of this project, particularly Berkani Rafik you my dear friend stode by my side and didn't abandon me in my time of need and for that you have my eternal gratitude , Abaddi Amin,Boukhelkhal Mouloud ,Boutrig Nacer and a long list of others. We apologize for not mentioning everyone.

Furthermore, we would like to thank the depatment for all the support and Help

Dedication

I dedicate this work to those who are dear to me.

To my coleuge Wafaa Benkiar for always trying to adding her unique touch to the work.

To those who have always believed in me, to those who have always encouraged me, and to my dear mother.

Devotion cannot be eloquent enough to express what you deserve for all the sacrifices you have continuously made for me since my birth, even into my adulthood.

I dedicate this work to you as a testament to my deep love.

May Almighty God grant you success and reward you with goodness.

Long life and happiness.

To my dear father: This humble work is the result of many sacrifices for our education.

You have made great physical and moral efforts for our appreciation.

Thank you, my dear brothers and sisters.

To all the books that showed me that there is always light at the end of the tunnel

to thoes that always reminded me that

You do not yeild, you do not falter

To Byce Quinlan . *LIGHT IT UP !*

I am very grateful to my professors from the Department of Electrical Engineering

To my classmates of 2023,

Ayache Sidali.

Dedication

I dedicate this work to those who are dear to me.

To my coleuge Sidali Ayache for always ensuring this work is the most perfect.

To those who have always taken care of me, to those who have always encouraged me, and to my dear mother.

You are my world and my whole univers revolve around you you are the senter of my gravity and words cannot express how much I care for you

I dedicate this work to you as a testament to my deep love.

May Almighty God grant you success and reward you with goodness.

Long life and happiness.

To my dear father: you are my right hand This humble work is the result of many sacrifices for your dedication.

You have made great physical and moral efforts for our appreciation.

Thank you, my dear brother and sisters.

For all my frineds and dear ones that always stode by me

I am very grateful to my professors from the Department of Electrical Engineering

To my classmates of 2023,

Benkiar Wafaa.

ملخص

في هذا العمل ، سنقوم بمحاكاة ودراسة التحكم عن بعد لمحرك غير متزامن باستخدام محول تردد يعتمد على شبكة ناقل إيثرنت . هدفنا هو إدارة تشغيل / إيقاف المحرك بكفاءة ;مع التحكم في الاتجاه والسرعة .يتم محاكاة وحدة التحكم المنطقية القابلة للبرمجة (PLC)من خلال برنامج TIA Portal على كمبيوتر مضيف ، بينما يتم تلقي إشارات التحكم من أداة LabVIEW الافتراضية (VI)على الكمبيوتر الرئيسي .يتم دعم الإرسال بواسطة برنامج NI OPC Servers.

الكلمات الرئيسية: الأتوماتية الصناعية ، برنامج TIA Portal ، محرك غير متزامن ، برنامج LabVIEW ، برنامج NI OPC Servers**.**

Abstract

In this work, we will simulate and study the remote control of an asynchronous motor using a frequency converter based on an Ethernet bus network. Our goal is to efficiently manage motor start/stop; with controlling direction and speed. The programmable logic controller (PLC) is emulated through TIA Portal software on a host PC, while control signals are received from a LabVIEW virtual instrument (VI) on the main PC. Transmission is supported by NI OPC Servers software.

Keywords: Industrial automation, TIA Portal software, asynchronous motor, LabVIEW software, NI OPC Servers Software.

Résumée

Dans ce travail, nous allons simuler et étudier le contrôle à distance d'un moteur asynchrone à l'aide d'un convertisseur de fréquence en se basant sur un réseau en bus Ethernet. Notre objectif est de gérer efficacement le démarrage/l'arrêt, avec contrôle de la direction et de la vitesse. Le contrôleur logique programmable (PLC) est émulé via le logiciel TIA Portal sur un PC hôte, tandis que les signaux de commande sont reçus d'un instrument virtuel LabVIEW (VI) sur le PC principal. La transmission est assistée par le logiciel NI OPC Servers.

Mots-clés : Automatisme industriel, logiciel TIA Portal, moteur asynchrone, logiciel LabVIEW, logiciel NI OPC Servers.

TABLE OF CONTENTS

INTRODUCTION 1

CHAPTER I

State of The Art 3

I.1 Introduction 4

I.2 Overview of Industrial Automation 4

I.2.1 Definition of industrial automation 4

I.2.2 Components of an automated system 5

I.2.3 Overview of industrial control systems 6

I.2.4 Types of Automation (Fixed and Flexible Automation) 7

I.3 Synoptics in Industrial Automation 7

I.3.1 Definition of Synoptic in Industrial Automation 7

I.3.2 Importance of Synoptics in Industrial Automation 8

I.3.3 Types of Synoptic and their uses 8

I.3.4 Examples of Synoptic Displays in Industrial Automation 9

I.4 PID Control 10

I.4.1 Definition of PID control 10

I.4.2 Importance of PID Control in Industrial Automation 11

I.4.3 Components of a PID controller 12

I.4.4 Types of PID Controller 13

I.4.4.1 Propotional Action Controller (P) 13

I.4.4.2 Propotional Integral (PI) Controller 14

I.4.4.3 Proportional Derivatives Controller (PD) 16

I.4.4.4 Propotional-Integral-Derivative PID Controller 17

I.4.5 PID Controller Structures 17

I.4.5.1 Parallel Structure 18
I.4.5.2 Serial Structure 18
I.4.5.3 Mixed Structure 19
I.4.6 The use of PID control in industrial automation 20
I.5 Hardware 21
I.5.1 Asynchronous Machine 21
I.5.2 Asynchronous Motor 21
I.5.2.1 Definition 21
I.5.2.2 Constitution of the asynchronous motor 21
I.5.2.3 Mechanical components 23
I.5.2.4 Operating principle of an asynchronous motor 24
I.5.2.5 Mechanical characteristics of asynchronous motors 25
I.5.2.6 Winding and connection of an asynchronous motor 26
I.5.2.7 Power Balance and Formulas 28
I.5.2.8 Speed control of an asynchronous motor 31
I.5.2.9 Advantages and Disadvantages of an Asynchronous Motor 32
I.5.3 Variable Frequency Drive 33
I.5.3.1 Definition of a Variable Frequency Drive 33
I.5.3.2 Constitution of a Variable Speed Drive 33
I.5.3.3 The Control Circuit 34
I.5.3.4 U/f control (Scalar) 34
I.5.3.5 The advantages of U/f control 35
I.5.3.6 The disadvantages of U/f control 35
I.5.3.7 Vector control of voltage (or flux) 35
I.5.3.8 Advantages of VVC control 36
I.5.4 Industrial Programmable Automation (API) 36
I.5.4.1 Automated System Functionalities 36
I.5.4.2 The automaton's architecture 36
I.5.4.3 Cyclic operation of APIs 37

I.5.4.4 Automate API S7-300 37
I.5.4.5 Characteristics of the S7 300 Programmable Logic Controller 38
I.5.4.6 Components of an S7-300 39
I.6 Conclusion 41

CHAPTER II

Industrial automation software 41

II.1 Introduction 43
II.2 Software Programs 43
II.2.1 Importance of Software Programs 43
II.2.2 Main Software Programs used in industrial automation 44
II.3 Tia Portal (Totally Integrated Automation Portal) 45
II.3.1 Software Description 45
II.3.1.1 Simatic Step 7 45
II.3.1.2 Simatic WinCC 46
II.3.2 Views of Tia Portal 46
II.3.3 Tia Portal Project View 47
II.3.4 Addressing of I/O's 48
II.3.5 API variables 48
II.3.6 API variable table 48
II.3.7 The Using of Tia Portal with the API S7-300 49
II.3.8 The Advantages of Tia Portal Software 53
II.4 LabVIEW 54
II.4.1 Software Description 54
II.4.2 Understanding Virtual Instruments (VI) 55
II.4.3 Origins of LabVIEW 55
II.4.4 The concept of a virtual instrument 56
II.4.5 Getting Started 56
II.4.6 LabVIEW Environment 57

II.4.6.1 Front Panel 58
II.4.6.2 Block Diagram 58
II.4.6.3 Tools 59
II.4.6.4 The Context Help Window 61
II.4.6.5 Data types used in LabVIEW 61
II.4.7 The Advantages of LabVIEW 61
II.4.8 Integration of LabView with the API S7-300 62
II.4.9 Examples of Industrial Automation Applications Using LabVIEW 63
II.4.10 Integration of LabVIEW with Tia Portal 64
II.5 NI OPC Server 65
II.5.1 Description 65
II.5.2 The Advantages of NI OPC Server 65
II.5.3 Integration of NI OPC Server with API S7-300 66
II.5.4 Integration of NI OPC Server with LabVIEW 67
II.5.4.1 NI OPC Server 68
II.5.4.2 NI OPC in LabVIEW 70
II.5.4.3 NI OPC and LabVIEW Simultaneously 73
II.5.5 Integration of NI OPC Server with Tia Portal 73
II.5.6 Net to PLC Sim configuration 74
II.5.7 WIFI 76
II.6 Conclusion 77

CHAPTER III

Simulation and Results Discussions 78

III.1 Introduction 79
III.2 Programming 79
III.2.1 Specifications 79
III.2.2 Tia Portal 79
III.2.2.1 Program Creation 80

III.2.2.2 Program in Contact language 80
III.2.3 LabVIEW program 82
III.2.3.1 Wiring and bound contact 82
III.3 Program Simulation 83
III.3.1 TIA Portal 83
III.3.2 Net to PLC Sim 85
III.3.3 NI OPC Servers 86
III.3.4 LabVIEW 87
III.4 Results 88
III.5 Conclusion 94
CONCLUSION 96
Bibiograph 98

LIST OF FIGURES

Chapter I:

State of the art

Figure I-1	Industrial Automation	5
Figure I-2	Les systèmes automatisés Components of industrial automation system	6
Figure I-3	Industrial control systems on the rise VPN overview	6
Figure I-4	Types of automation	7
Figure I-5	Synoptic screen of the SCADA package	10
Figure I-6	PID Controller	11
Figure I-7	PID controller components	13
Figure I-8	Block Diagram of P controller	14
Figure I-9	Electronic Diagram of P controller	14
Figure I-10	PI controller Block Diagram	15
Figure I-11	PI Controller index response	15
Figure I-12	Electronic Diagram of PI Controller	15
Figure I-13	PD controller Block Diagram	16
Figure I-14	Index response of PD controller	16
Figure I-15	Electronic Diagram of PD Controller	17
Figure I-16	Parallel Structure Controller	18
Figure I-17	serial structure controller	18
Figure I-18	Mixed Structure Controller	19
Figure I-19	2nd Mixed Structure Controller	19
Figure I-20	PID controller, control valves	20
Figure I-21	PID controller performing automatic control of system	21

Figure I-22	Stator	22
Figure I-23	Rotor	22
Figure I-24	Operation of a motor	24
Figure I-25	Operating quadrants	25
Figure I-26	The connection of the terminals to the coils	26
Figure I-27	Star configuration	27
Figure I-28	Triangle configuration	27
Figure I-29	Star-triangle Connection Diagram	28
Figure I-30	Power balance of the asynchronous motor	28
Figure I-31	Variable speed drive (ATV12h018m2)	33
Figure I-32	VFD Wiring Diagram (ATV12H018M2) [11]	34
Figure I-33	constant torque operation	35
Figure I-34	Cycle for an API	37
Figure I-35	Programmable Logic Control S7-300	38
Figure I-36	Components of an S7-300	39
Figure I-37	Power supply of the S7 300 PLC	39
Figure I-38	CPU connection	40
Figure I-39	Address bus	41

Chapter II

Industrial automation Software

Figure II-1	Illustration of the construction of Tia portal	45
Figure II-2	Tia Portal Views	46
Figure II-3	Portal View	46
Figure II-4	Project View	47
Figure II-5	Addressing I/Os	48
Figure II-6	Create a Project	49
Figure II-7	Configure device	50

Figure II-8	1st Step to configure device	50
Figure II-9	3rd Step to configure device	51
Figure II-10	4th Step to configure device	51
Figure II-11	5th Step to configure device	52
Figure II-12	Tia Portal Interface	53
Figure II-13	LabVIEW	55
Figure II-14	Virtual Instrument	56
Figure II-15	Startup Screen of LabVIEW	57
Figure II-16	LabVIEW Parts	57
Figure II-17	Front Panel	58
Figure II-18	Block Diagram	59
Figure II-19	Front Panel, Block Diagram Toolbars	59
Figure II-20	Control Palette	60
Figure II-21	Help Window	61
Figure II-22	Data Types	61
Figure II-23	LabVIEW with API S7-300 via Ethernet (TCP)	63
Figure II-24	Industrial IOT Solution using LabVIEW	64
Figure II-25	NI OPC Server	65
Figure II-26	NI OPC Server with API S7-300	67
Figure II-27	2 Applications LabVIEW with NI OPC Server	68
Figure II-28	Open NI OPC Server	68
Figure II-29	NI OPC Server Configuration [27]	69
Figure II-30	NI OPC Server Create New Tag [27]	69
Figure II-31	NI OPC in LabVIEW [27]	70
Figure II-32	NI OPC write in LabVIEW [27]	70
Figure II-33	Writing in NI OPC server [27]	71
Figure II-34	Connect to NI OPC server [27]	71
Figure II-35	NI OPC read in LabVIEW	72
Figure II-36	Read from NI OPC [27]	72
Figure II-37	Read from NI OPC	72

Figure II-38	Reading and writing from NI OPC and LabVIEW in the same time	73
Figure II-39	Running write client and read client simultaneously [27]	73
Figure II-40	Getting Port 102	75
Figure II-41	inserting the network and Plc sim addresses	75
Figure II-42	obtaining the IPv4 Address	76

Chapter III

Simulation and Results Discussions

Figure III-1	Network 1	80
Figure III-2	Network 2	81
Figure III-3	Network 3	81
Figure III-4	Network 4	81
Figure III-5	Network 5	81
Figure III-6	The Input Section	82
Figure III-7	The Output Section	83
Figure III-8	Tia portal simulation bar	83
Figure III-9	Program loading	84
Figure III-10	Downloading to device	84
Figure III-11	Starting the simulation and monitoring	85
Figure III-12	Starting the PLC SIM	85
Figure III-13	Configuration of Net to PLC Sim	86
Figure III-14	Launching the NI OPC Servers Quick Client	86
Figure III-15	NI OPC Quick Client	87
Figure III-16	LabVIEW simulation bar	87
Figure III-17	Launching the LabVIEW	87
Figure III-18	Actual network configuration	88
Figure III-19	First action	89
Figure III-20	First action: count update	89

Figure III-21 Second action 89
Figure III-22 Second action: update count 90
Figure III-23 Move function in the second action 90
Figure III-24 Unscale function in the second action 90
Figure III-25 Third action 91
Figure III-26 Network 3 91
Figure III-27 Third action update count 91
Figure III-28 Forth action 92
Figure III-29 Network 1 92
Figure III-30 Quick Client Update Count 92
Figure III-31 Fifth action 93
Figure III-32 Network 2 93
Figure III-33 Quick client update count 93
Figure III-34 Sixth action 94
Figure III-35 Quick client update count 94

LIST OF TALBES

Chapter I

Table I-1 Number of poles and synchronous speed 26
Table I-2 Technical Specifications of S7 300 CPU 313 38
Table I-3 Inputs/outputs and address 40

Chapter III

Table III-1 variable table 80

ABBREVIATIONS, ACRONYMS, AND INITIALISMS

PLC	Programmable Logic Controllers
TIA PORTAL	Totally Integrated Automation Portal
VI	Virtual Instrument
NI OPC	National instruments ,OLE for Process Control
PC	Personal computer
PID	Proportional-Integral-Derivative
API	Application Programming Interface
VCC	Voltage Common Collector
I/Os	Inputs / Outputs
VFD	Variable Frequency Drive
CPU	Central Processing Unit
HMI	Human-machine interface
SCADA	Supervisory Control and Data Acquisition
DCS	Distributed Control System
ICS	industrial control systems
CNC	computer numerical control
IOT	Internet of things
TCP	Transmission Control Protocol

Used NOTATIONS

Ki	integral constant gain
Kp	proportional constant gain
Kd	dirivation constant gain
e	error signal
Ut	controller control
Ep	in proportional
Up	out proportional
T	torque
G	gain
Cp	controller transfer function
U(t)	T in u equation
N_0	synchronous speed rpm
ε(t)	error
n	motor speed rotation speed rpm
Cp	Redendance
Ti	integral action
Td	deritive action
α	alpha
U	Voltage
I	Current
Ps	consumed power
Ptr	rotor transmitted power
π	pi constant
Pu	out put power
Pem	electro magnitic power
S	apparent power
Ω	speed
Rpm	Round Per Minute
g	gravity

Introduction

PLCs have become a crucial part of industrial automation due to their versatility and flexibility. With their ability to store and execute complex instructions, PLCs can control a wide range of devices and processes, making them indispensable in modern manufacturing and industrial processes. PLCs have several benefits over traditional electrical controls. They allow for easier programming and maintenance, offer faster and more accurate monitoring and control, and can reduce the amount of physical wiring required, thus saving space and reducing installation costs. Furthermore, PLCs can be customized to suit specific applications, making them ideal for a wide range of industries. They can be programmed to perform complex operations such as motion control, process control, and data acquisition, and can communicate with other devices and systems in real-time. Also, the PLCs can simulate and optimize system functionality before deployment, resulting in reduced downtime and improved production efficiency.[1]

PLC project stands for Programmable Logic Controller project. PLC is an industrial computer control system that continuously monitors the state of input devices and makes decisions based on custom programs to control the state of output devices. In the context of industrial automation, a PLC project is often used to control, monitor and automate complex industrial processes and machinery. In this particular scenario, the PLC project is being used to establish seamless communication between two PCs using the TIA portal and LabVIEW. TIA Portal is a software package used for programming and configuring Siemens automation systems, including PLCs. LabVIEW, on the other hand, is a graphical programming environment used to develop measurement and automation applications. The combination of these two powerful software tools allows for the development of sophisticated industrial control and automation solutions. NI OPC servers, on the other hand, are software applications that enable communication between different software systems and hardware devices in an industrial automation environment. By integrating NI OPC servers into the overall solution, it becomes possible to establish seamless communication and data exchange between the PLC system, TIA portal, and LabVIEW applications, enabling efficient control, monitoring, and automation of industrial processes.

One of the challenges faced by designers in the field is maintaining a strong connecting link between software and hardware components. This is essential to ensure that the production process runs smoothly and seamlessly with no interruptions or downtime. Our objective in this work is to ensure the communication of the different components and automation devices and to ensure that they are compatible with each other.

To provide in-depth coverage and analysis of the topic, we will lay out three separate chapters. The first chapter will examine the state of the art in industrial automation systems. It will cover a variety of topics, including a general overview of industrial automation, the significance of synoptics, and PID control. Without omitting the hardware section, which will include the asynchronous machine, variable frequency drive, and industrial programmable automation (PLC), in particular, the S7-300 PLC. Our goal will be to provide a thorough grasp of the most recent advancements in automation, emphasizing their impact on production, efficiency, and control systems.

For the second chapter, we will conduct extensive study on the industrial automation softwares used in our case. Namely the TIA portal, and the LabVIEW. We will also explore the crucial role of NI OPC servers in connecting and synchronizing these software programs,

enabling reliable and optimized data transmission. This research aims to explore the importance of those softwares in improving the efficiency and productivity of industrial processes.

The third chapter will deal with the simulation and the analysis of the results. First, we will start with the programming of the different software, with an emphasis on the development and implementation of the code. Second, we will exploit simulation techniques to demonstrate how virtual environments replicate real-world scenarios. Finally, we will discuss the results obtained, and we will analyse and interpret the data generated by the simulation process.

We shall conclude our work with a broad conclusion.

Chapter I
State of The Art

I.1-Introduction

This chapter entitled « State of The Art », presents a general overview of industrial automation, including its importance, key components, industrial control systems, and the different types of automation fixed and flexible automation. On the other hand, we also introduce synoptic, their importance in industrial automation, and the use of PID control in industrial processes. Furthermore, this chapter delves into the concept of PID control, its definition, its importance in industrial automation, and the components of a PID controller and its use in the industry. Additionally, we presented the hardware components of industrial automation such as the asynchronous machine, asynchronous motor and their equations and diagrams. We also discussed the speed variator, industrial programmable automation and the API S7-300.

I.2-Overview of Industrial Automation

I.2.1-Definition of industrial automation

Industrial automation refers to using advanced technology, control systems, computers, and robotics to manage different processes and machines in an industrial environment. The aim is to automate repetitive or complex tasks that were previously carried out by humans, thus improving efficiency, reducing labor costs, and ensuring safety in the workplace [2].

Industrial automation is essential in modern manufacturing because it offers many benefits, including:

- Increased efficiency.
- Increased productivity.
- Improved quality.
- Reduced labor costs.
- Enhanced security.
- Greater flexibility [2].

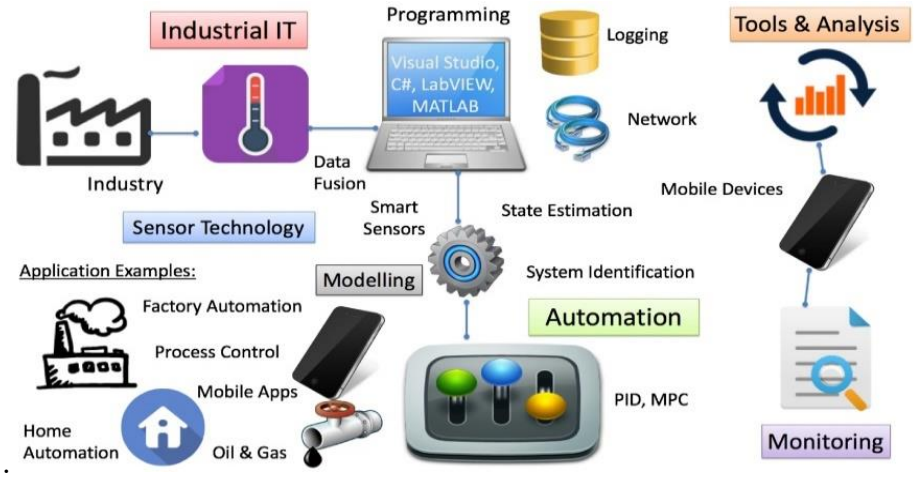


Figure I-1: Industrial Automation

I.2.2-Components of an automated system

- **Sensors and actuators:** these are used to gather information about the system and to control its operation. Sensors measure physical parameters such as temperature, pressure, and flow rate, while actuators are used to control valves, motors…
- **Programmable Logic Controllers (PLCs):** These are specialized computers that are used to control the operation of the automated system. They can be programmed to respond to sensor inputs and to control the operation of actuators.
- **Human-machine interface (HMI):** This is the interface through which operators interact with the system. It may include displays, buttons, and other input/output devices that allow operators to monitor the system and make adjustments as needed.
- **Communication network:** This is used to allow different components of the system to communicate with each other. It may include wired or wireless connections, depending on the specific application.
- **Software:** This includes the programming that is used to control the system. It may include specialized software for specific applications, as well as general-purpose programming languages such as C++ or Python.
- **Power Supply:** This is needed to provide power to the various components of the system. It may include transformers, power supplies, and other components [3].

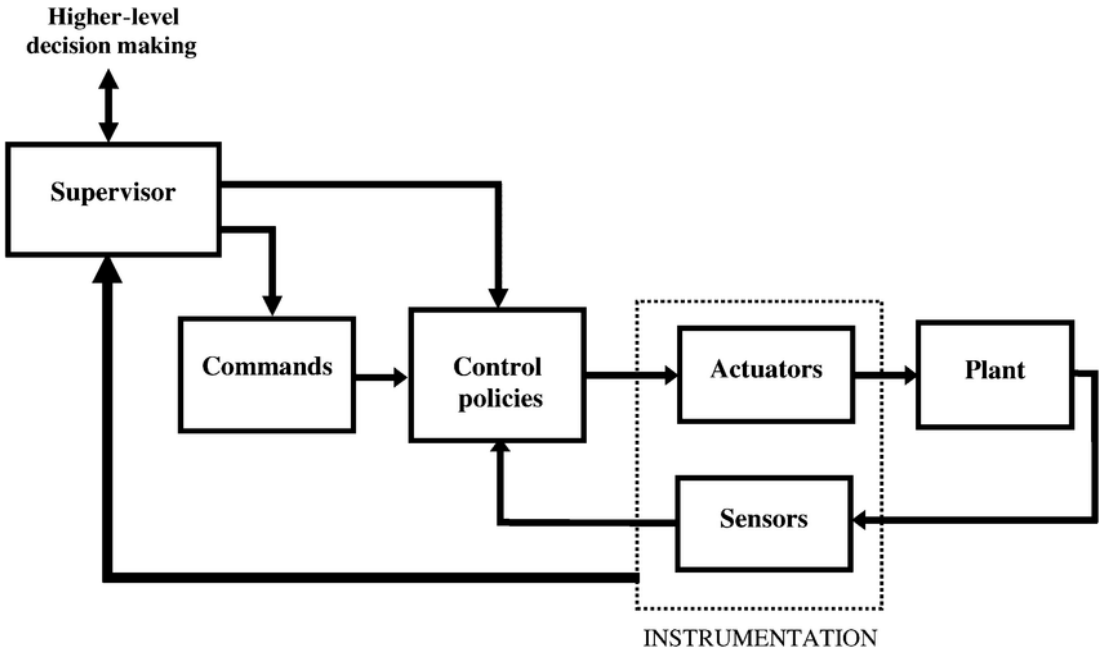


Figure I-2: Components of industrial automation system

I.2.3-Overview of industrial control systems

Industrial control systems (ICS) are control systems used in various industrial sectors and critical infrastructures. ICS are used to manage and control industrial processes and equipment. It includes different types of control systems such as SCADA systems, DCS systems, and PLCs. The ICS has two main parts, the process, and the control. The process is to produce the output, while the control part specifies the desired output or performance. ICS can operate in open loop, closed loop, or manual mode. These systems consist of various components that work together to monitor and control industrial processes [4].



Figure I-3: Industrial control systems on the rise VPN overview

I.2.4-Types of Automation (Fixed and Flexible Automation)

- **Fixed Automation:** Fixed automation, also known as hard automation, is a system that is designed to perform a specific task repeatedly, without any significant change in the product or process being produced. The system is often custom-built for a specific manufacturing process and is typically expensive to design, set up and reconfigure. This type of manufacturing automation uses specialized hardware to automate assembly or fixed sequential processing operations [3].
- **Flexible Automation:** Flexible automation, also known as soft automation, is a type of automation that uses a centralized computer that is human-controlled and responsible for controlling and updating the production process. Additionally, this type incorporates a material handling system that groups the machines. One of the key features of this type is its ability to be easily reconfigured or programmed to perform different tasks or to adapt to changes in the product or process being produced. This system uses a combination of sensors, controllers, and actuators to detect and adapt to changes in the production process. This type of system is more versatile than fixed automation and is typically used in environments where the production process changes frequently or where customization is required. Examples of this type include robots and computer numerical control (CNC) machines [3].

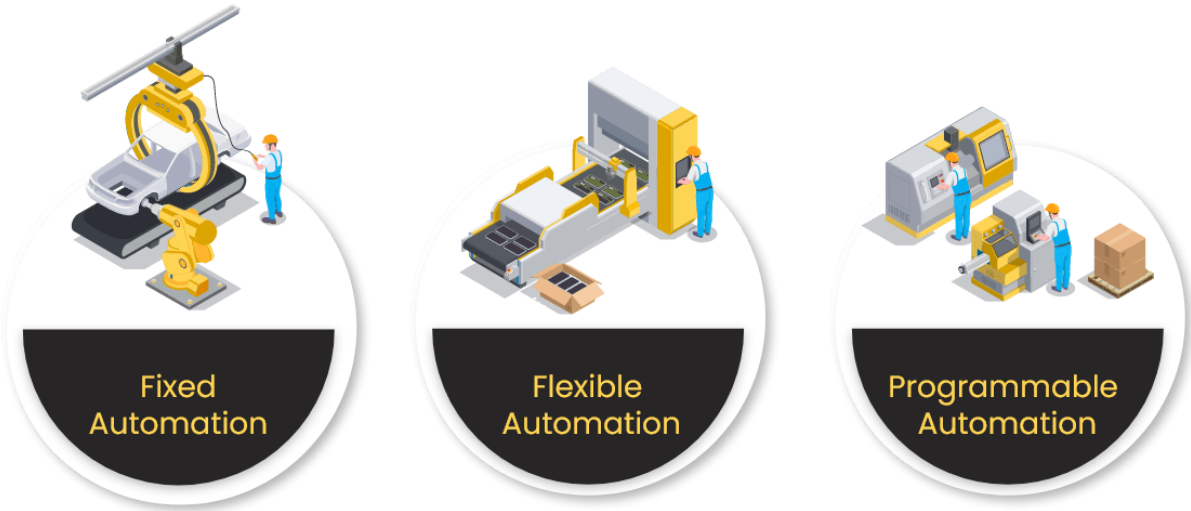


Figure I-4: Types of automation

I.3-Synoptics in Industrial Automation

I.3.1-Definition of Synoptic in Industrial Automation

Synoptic typically refers to the visual representation of processes or systems that display real-time information in a simplified and intuitive manner. Synoptic are commonly used in

SCADA (Supervisory Control and Data Acquisition) systems to provide operators with an overview of the process or system they are monitoring. They typically include graphical representations of problems or abnormal conditions. Synoptic are designed to provide operators with a clear and concise understanding of a process or system, enabling them to quickly identify and respond to any problems that may arise [5].

I.3.2-Importance of Synoptics in Industrial Automation

Synoptics are important in industrial automation for the following reasons::

- **Visualization:** Synoptic provides a graphical representation of industrial processes making it easier for operators to understand and monitor processes.
- **Control:** Synoptic allow operators to control industrial processes by providing an interface to adjust set points, turn equipment on and off, and respond to alarms or abnormal conditions.
- **Troubleshooting:** When a problem occurs in an industrial process, weather maps can be used to determine the root cause of the problem and take corrective action. Operators can quickly locate affected equipment or process areas and take appropriate action to resolve the problem.
- **Efficiency:** by providing a clear and concise overview of industrial processes, weather maps can help operators optimize processes, reduce waste and increase efficiency [6].

I.3.3- Types of Synoptic and their uses

Synoptics can be classified into different types based on their functions and applications in industrial automation. Some common types of synoptic and their uses are:

- **Process Synoptic**: This type of synoptic provides a visual representation of the entire industrial process, including the status of equipment materials and other resources. It helps operators monitor and control processes in real-time, quickly identify and diagnose problems, and make informed decisions.
- **Trend Synoptic:** Trend synoptic displays the historical trend of process variables, such as temperature, pressure, and flow over time. It helps operators identify patterns, identify anomalies and optimize process performance.
- **Maintenance Synoptic:** Maintenance synoptic provides an overview of the maintenance activity, including equipment status, planned and unplanned maintenance, and maintenance history. It helps maintenance personnel plan and prioritizes their tasks, reducing downtime and increasing equipment reliability.

- **Energy Synoptic:** Energy synoptic displays the energy consumption and production data, including electricity, gas, and water usage, as well as the status of energy-saving measures. It helps operators identify energy-wasting areas, take energy-saving measures and reduce energy costs.
- **Process Flow Diagrams:** These diagrams show the flow of materials or fluids through a system, often using arrows or lines to indicate the direction of flow. They are useful for monitoring the overall performance of a process and identifying areas where flow may be impeded or slowed down.
- **Piping and instrumentation diagrams (P&ID):** These synoptic provide a detailed view of a system's components and connections, including valves, pumps, and other equipment. They are commonly used in process control applications and can help operators identify and diagnose flow, pressure, and temperature problems.
- **Electrical Schematics:** These diagrams show the wiring and electrical connections between system components and are used in electrical control applications. They can help operators with problems related to electrical breakdowns or malfunctions.
- **Human Machine Interface (HMI):** HMI is a type of synoptic display that provides a graphical interface for the operator to interact with the control system. They typically include buttons, sliders, and other graphical elements and can be customized to meet the specific needs of a particular application.
- **Alarm Event displays:** These synoptic show real-time alarms and events occurring in the system and can be used to alert operators to critical issues that need attention. They can also provide historical data on past alarms and events, allowing operators to identify patterns and trends. It helps operators respond quickly to critical alerts, minimizing downtime and keeping plants and people safe [7].

I.3.4-Examples of Synoptic Displays in Industrial Automation

Here are some more examples of synoptic displays in industrial automation:

- **Power Plant Control Room Synoptic display:** This type of display shows the status of various plant components such as generators, pumps, and turbines. It allows operators to monitor plant performance and identify issues that need to be addressed.
- **Conveyor Control Indicator:** This type of indicator shows the status of the conveyor system, including belt position and speed, as well as the status of various sensors and actuators. It allows operators to monitor system performance and quickly identify any problems that may arise.

- **Building Automation Synoptic display:** This type of synoptic display shows the status of various building systems such as HVAC (**H**eating, **V**entilation, and **A**ir **C**onditioning), lighting, and security. It allows building operators to monitor and control these systems from a central location, improving energy efficiency and security.
- **Industrial Process Control Synoptic display:** This type of synoptic display shows the status of various industrial process components such as tanks, valves, and pumps. It allows operators to monitor and control processes, ensuring efficient and safe operation [8].

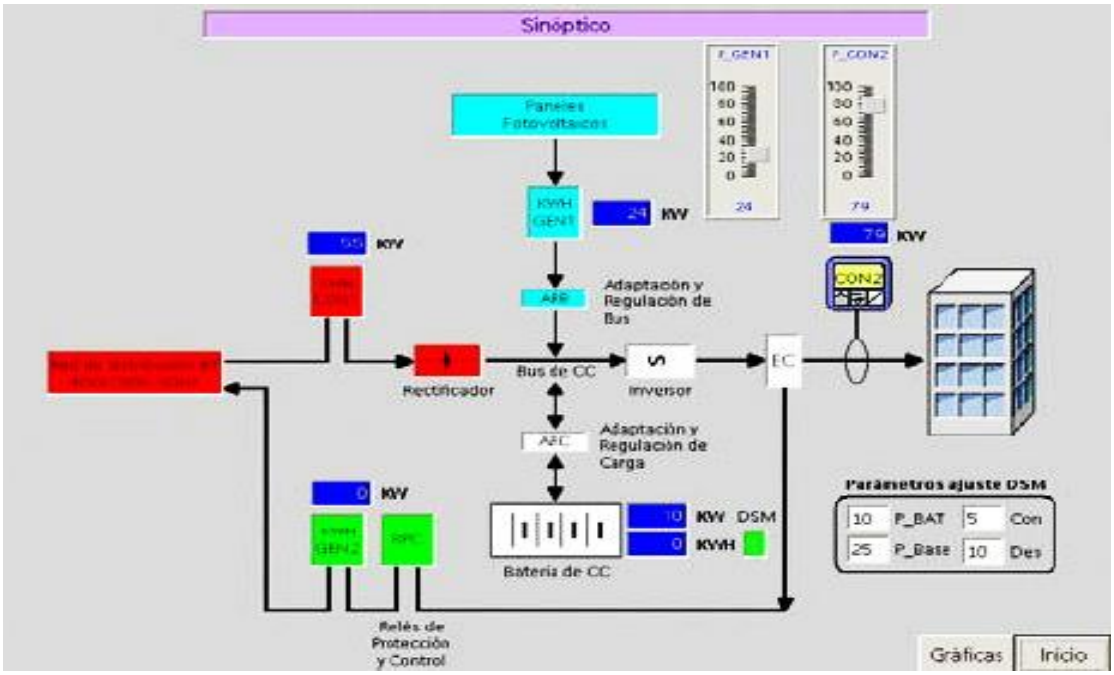


Figure I-5: Synoptic screen of the SCADA package

I.4-PID Control

I.4.1-Definition of PID control

PID (Proportional Integral Derivative) control, is a feedback control algorithm widely used in engineering and process control. It is a closed-loop control system that continuously calculates an error signal by comparing the desired set point with the current process variable and uses this error signal to adjust the control action.

The three terms of PID control represent different aspects of control action. The term proportionally adjusts the output based on the strength of the error signal, the integral term adjusts the output based on the cumulative error over time, and the derivative term adjusts the output based on the rate of change of the signal. Error signal.

PID control is commonly used in factory automation, robotics, and manufacturing process to regulate temperature, pressure, flow, and other process variables. It is also used in many applications, including engine control, the autopilot of aircraft, and even in heating and cooling systems in homes [9].

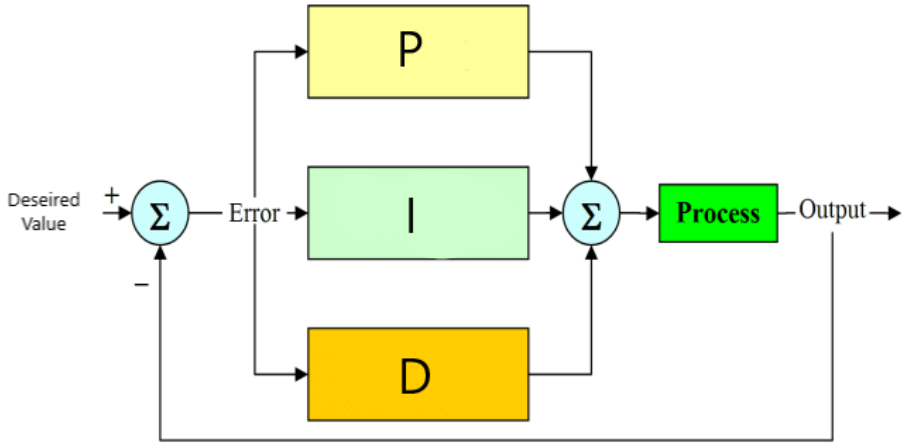


Figure I-6: PID Controller

I.4.2-Importance of PID Control in Industrial Automation

- **Precise control:** PID control allows precise control of industrial processes. With the ability to maintain the set point despite disturbances, it ensures the process is operating within the desired range.
- **Stability:** PID control ensures the stability of industrial processes with a minimum of fluctuations. This is especially important in processes where safety is very important, such as chemical plants.
- **Energy efficiency:** By providing precise control of industrial processes, PID control can help optimize energy consumption. This leads to cost savings and reduced carbon emissions.
- **Consistency:** PID control guarantees a constant quality of the final product. This is especially important in industries such as food and beverage, where consistency in product quality is critical to maintaining customer satisfaction.
- **Reduced downtime:** PID control can help reduce downtime in industrial processes. Providing precise and stable control, it minimizes the risk of equipment damage and failure, resulting in increased productivity and efficiency.
- **Improved control performance:** PID control is essential in industrial automation because it improves system control performance. The PID controller can accurately maintain the desired process variable even under changing conditions, thus improving

the quality of the final product. **Increase Efficiency:** PID control enables industrial processes to operate more efficiently by reducing process variability and reducing wasted energy. This translates into lower production costs and higher profits for the industry.

- **Reduce Downtime:** Using PID control in industrial automation can help reduce downtime and maintenance costs. Continuously monitoring and adjusting process variables, the system can prevent critical errors and ensure machines operate at peak efficiency.
- **Improves Product Quality:** PID control is important in industries that require precise control of variables, such as temperature and pressure, to produce high-quality products. Using PID control can help ensure that process variables stay within the desired range, resulting in consistent, reliable product quality.
- **Facilitates Automation:** PID control is an essential part of industrial automation as it enables automation systems to operate more efficiently and reliably. With PID control, automated systems can adapt to changing conditions and maintain peak performance without manual intervention.
- **Offers Flexibility:** PID control offers a high degree of flexibility in industrial automation systems, allowing engineers to adjust control parameters as needed to optimize performance. This versatility makes PID control an ideal solution for many industrial processes, from the simplest to the most complex [9].

I.4.3-Components of a PID controller

- **Proportional Component (P):** The proportional component of a PID controller provides an output signal proportional to the error signal between the setpoint and the measured values. It determines the corrective action based on the magnitude of the error signal.
- **Integral Component (I):** The integral component of a PID controller provides an output signal proportional to the integral of the error signal over time. It corrects any steady-state errors that persist after applying the scale component.
- **Derivative Component (D):** The derivative component of a PID controller provides an output signal proportional to the rate of change of the error signal. It predicts the future behavior of the error signal and works to prevent overshoot and oscillation of the output.
- **Set point:** The set point is the desired value for the process variable. This is the value that the PID controller strives to achieve.

- **Process Variable:** The process variable is the actual value of the variable controlled by the PID controller. It is measured by sensors and fed back to the controller.
- **Controller Output:** The controller output is a signal that is sent to a control element (such as a valve or motor) to regulate the process variable. It is the sum of the P, I, and D components, each of which is multiplied by a gain factor to adjust the overall response of the controller.
- **Gain:** The gain determines the relative weight of each component in the output of the controller. They are used to tailor the controller response to the characteristics of the controlled process.
- **Control Element:** The control element is a device that adjusts the process variable based on the output of the controller. It can be a valve, motor, or any other device that can change the process variable [9].

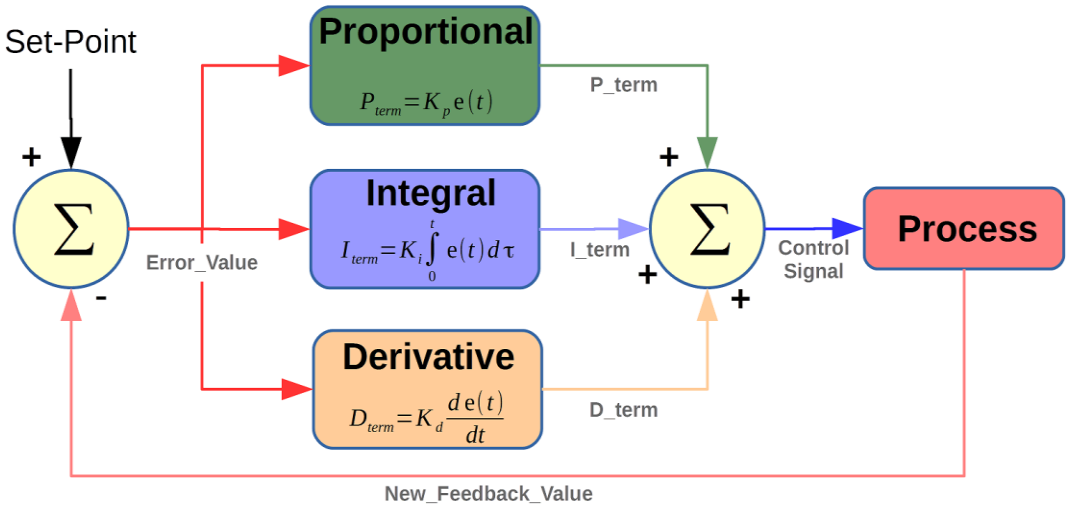


Figure I-7: PID controller components

I.4.4-Types of PID Controller

I.4.4.1-Propotional Action Controller (P)

The output of the proportional controller $u(t)$ is proportional to the error signal $e(t)$, which is the difference between the set point and the process variable[10].

- **Controller control Law:**

$$u(t) = k_p e(t) \tag{I.1}$$

- **P Controller Block Diagram**:

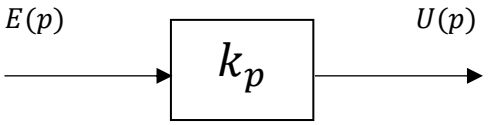


Figure I-8: Block Diagram of P controller

- **Practical Implementation of the P controller**:

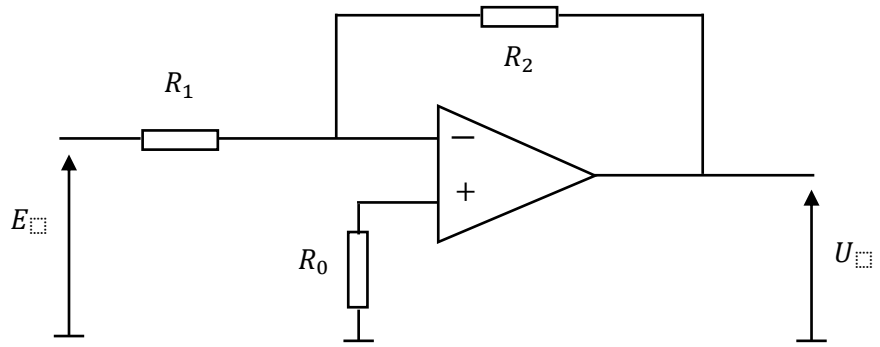


Figure I-9: Electronic Diagram of P controller

- **Transfer function of the P controller**:

$$G_c(p) = \frac{U(p)}{E(p)} = \frac{R_2}{R_1} = k_p \qquad \text{I.2}$$

I.4.4.2-Propotional Integral (PI) Controller

- **PI Controller Transfer Function**:

$$C(p) = K_p \frac{1 + T_i P}{T_i P} \qquad \text{I.3}$$

- **Time Equation**:

$$u(t) = K_p \varepsilon(t) \int_0^t \varepsilon(t) dt \qquad \text{I.4}$$

- **PI Controller Block Diagram**:

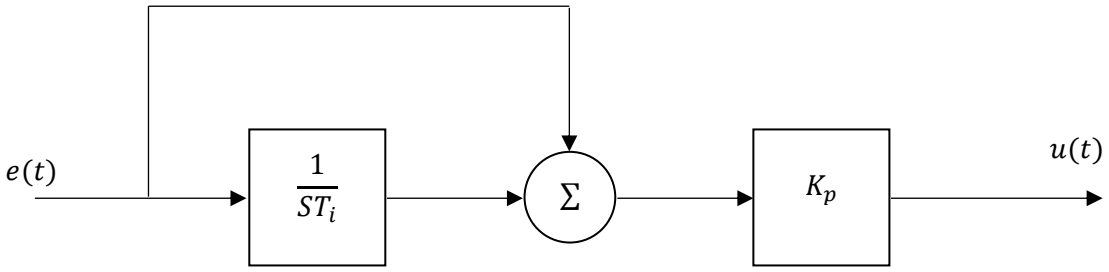


Figure I-10: PI controller Block Diagram

- **PI Controller index response**:

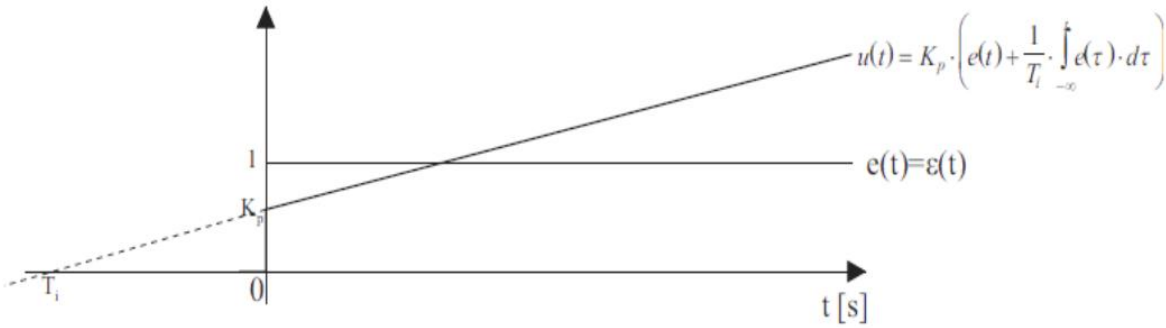


Figure I-11: PI Controller index response

- **Practical Implementation:**

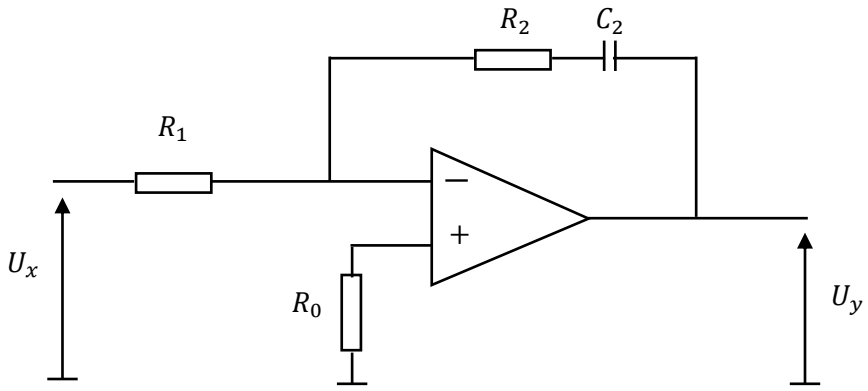


Figure I-12: Electronic Diagram of PI Controller

It has a counter-reaction circuit formed by a C_2 capacitor serial with resistor R_2 [10].

Finally, the transfer function will be:

$$G_R(p) = \frac{U_y(P)}{U_x(P)} = \frac{R_2}{R_1}\,\frac{1 + PR_2C_2}{PR_2C_2} \qquad \text{I.5}$$

I.4.4.3- Proportional Derivatives Controller (PD)

- **Transfer Function of the PD controller**:

$$C(p) = \frac{U(p)}{E(p)} = K_p(1 + T_d p) \qquad \text{I.6}$$

- **Time Equation:**

$$u(t) = K_p\left(e(t) + T_d\frac{de(t)}{dt}\right) \qquad \text{I.7}$$

- **PD Controller Block Diagram**:

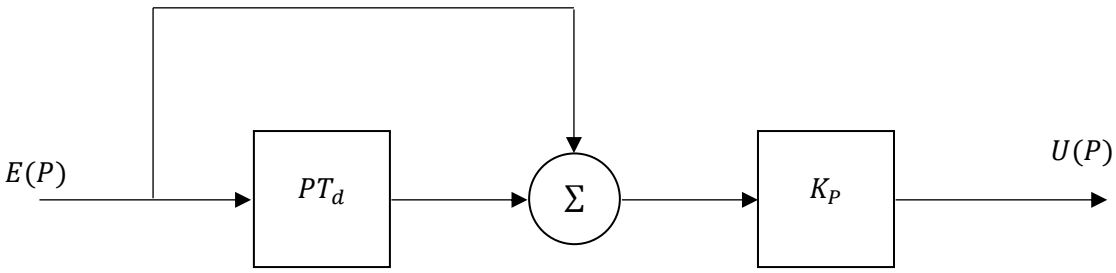


Figure I-13: PD controller Block Diagram

- **PD Controller index response**:

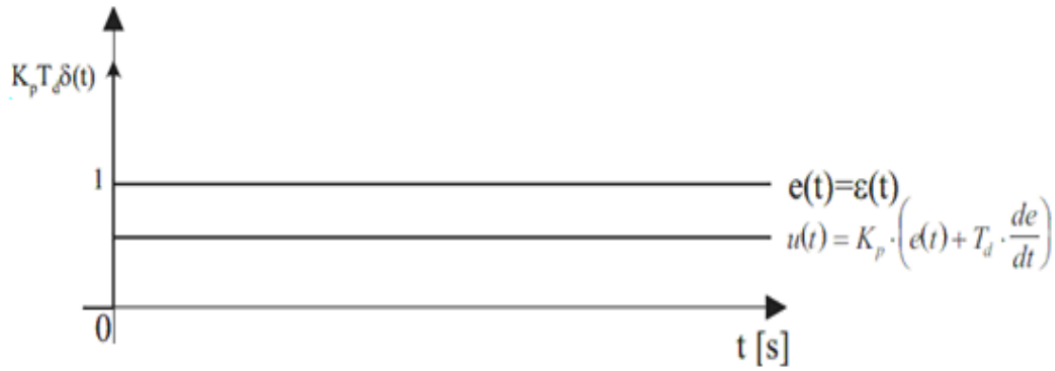


Figure I-14: Index response of PD controller

Practical Implementation:

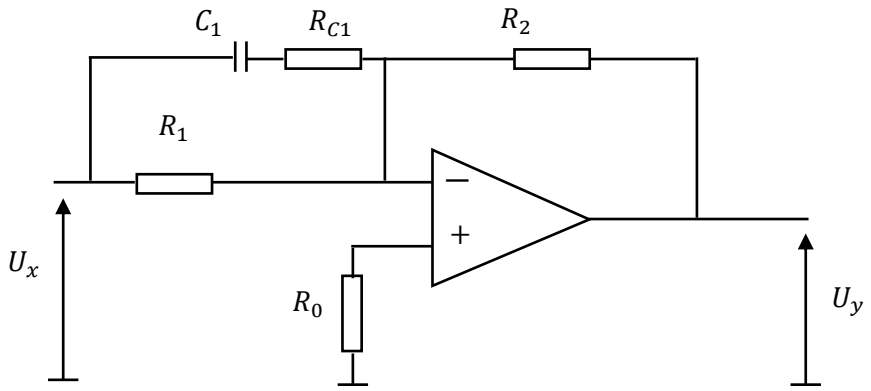


Figure I-15: Electronic Diagram of PD Controller

Finally the transfer function will be [10]:

$$G_R(s) = \frac{U_y(P)}{U_x(P)} = \frac{R_2(1 + P(R_{C1} + R_1)C_1)}{R_1(1 + PR_{C1}C_1)} \quad \text{I.8}$$

I.4.4.4-Propotional-Integral-Derivative PID Controller

The PID (Proportional Integral Derivative) controller is the combination of three core actions P, I, and D. Through the terms P and I, they allow the cancellation of a static error while allowing, through to the D action, faster performances than those of a PI controller. The PID controller is a control device for closed-loop regulation of an industrial system. It's the most used controller in the industry, and it controls a lot of processes [10].

The PID corrector acts in 3 ways:

- **Proportional action P:** It ensures speed. The error is multiplied by a gain K_p.
- **Integral action I:** It cancels the static error. The error is integrated and divided by a gain T_i .
- **Derivative action D**: It improves stability. The error is derived and multiplied by a gain T_d [10].

I.4.5-PID Controller Structures

The elementary actions of a controller can be associated in serval ways. We are talking about the algorithm structure of the controller. The PID parameter values do not give the same process behavior depending on whether the structure is parallel or mixed. This is why it is essential that the operator knows the existing structure of the controller. The structure is imposed by the controller manufacturer [10].

I.4.5.1-Parallel Structure

- **Time Equation**:

$$u(t) = K_p\varepsilon(t) + \frac{1}{T_i}\int_0^t \varepsilon(t)dt + T_d\frac{d\varepsilon(t)}{dt} + u_0 \quad \text{I.9}$$

- **Transfer function Equation**:

$$C(p) = \frac{y(p)}{\varepsilon(t)} = K_p + \frac{1}{T_i p} + T_d p \quad \text{I.10}$$

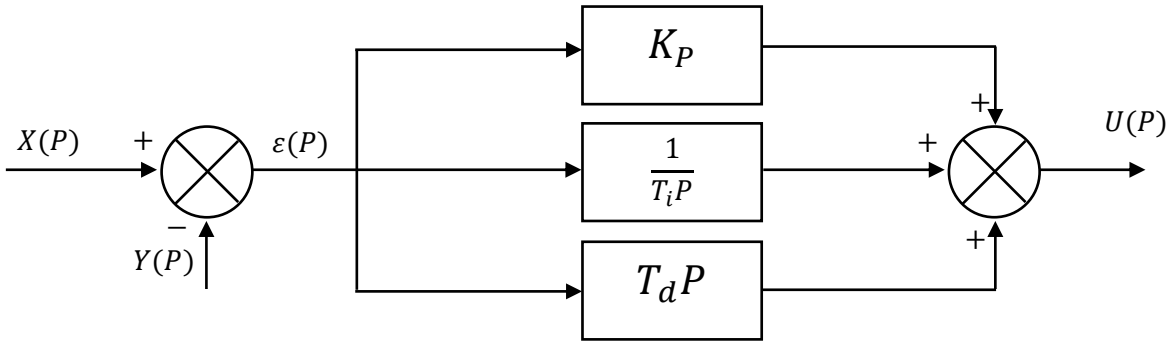


Figure I-16: Parallel Structure Controller

I.4.5.2-Serial Structure

- **Time Equation:**

$$u(t) = \alpha K_p\varepsilon(t) + \frac{K_p}{T_i}\int_0^t \varepsilon(t)dt + K_p T_d\frac{d\varepsilon(t)}{dt} + u_0 \quad \text{I.11}$$

Where:

$$\alpha = \frac{T_i + T_d}{T_i}$$

Is the theorical coefficient of interaction between integral and derived action [10].

Transfer function equation:

$$C(p) = \frac{y(p)}{\varepsilon(t)} = K_p\left(1 + \frac{1}{T_i p}\right)(1 + T_d p) \quad \text{I.12}$$

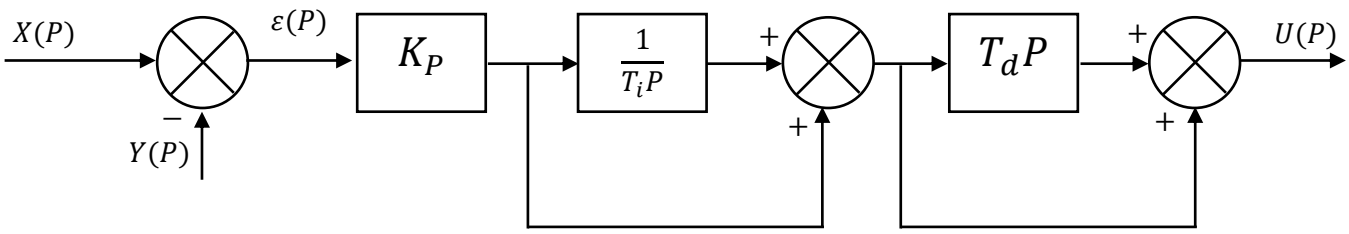


Figure I-17: Serial Structure Controller

I.4.5.3-Mixed Structure

This structure is currently the most used by builders [10].

- **Time Equation**:

$$u(t) = K_p\varepsilon(t) + \frac{K_p}{T_i}\int_0^t \varepsilon(t)dt + K_pT_d\frac{d\varepsilon(t)}{dt} + u_0 \quad \text{I.13}$$

- **Transfer function Equation**:

$$C(p) = \frac{y(p)}{\varepsilon(t)} = K_p\left(1 + \frac{1}{T_ip} + T_dp\right) \quad \text{I.14}$$

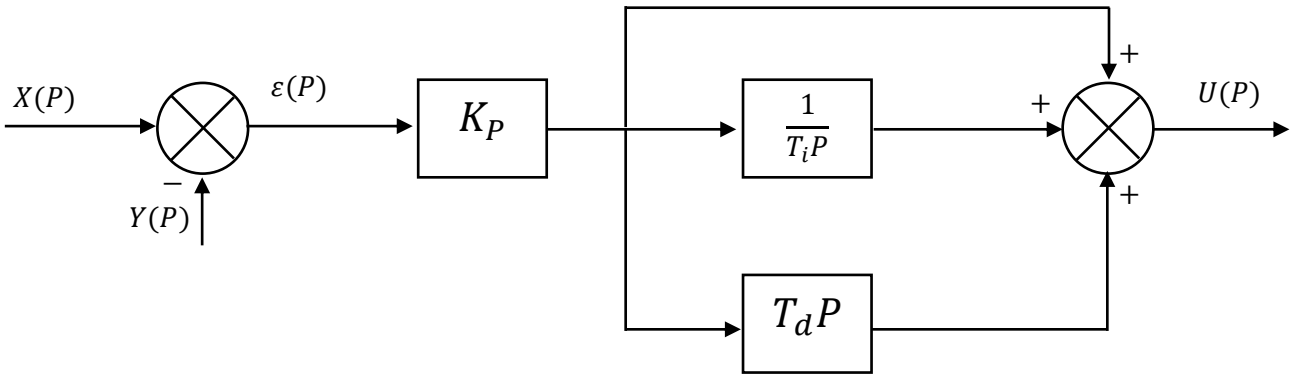


Figure I-18: Mixed Structure Controller

Other mixed structure also exist as [10]:

- **2nd Time Equation**:

$$u(t) = K_p\varepsilon(t) + \frac{K_p}{T_i}\int_0^t \varepsilon(t)dt + K_pT_d\frac{d\varepsilon(t)}{dt} + u_0 \quad \text{I.15}$$

- **2nd Transfer function Equation**:

$$C(p) = \frac{y(p)}{\varepsilon(t)} = K_p\left(1 + \frac{1}{T_ip}\right) + T_dp \quad \text{I.16}$$

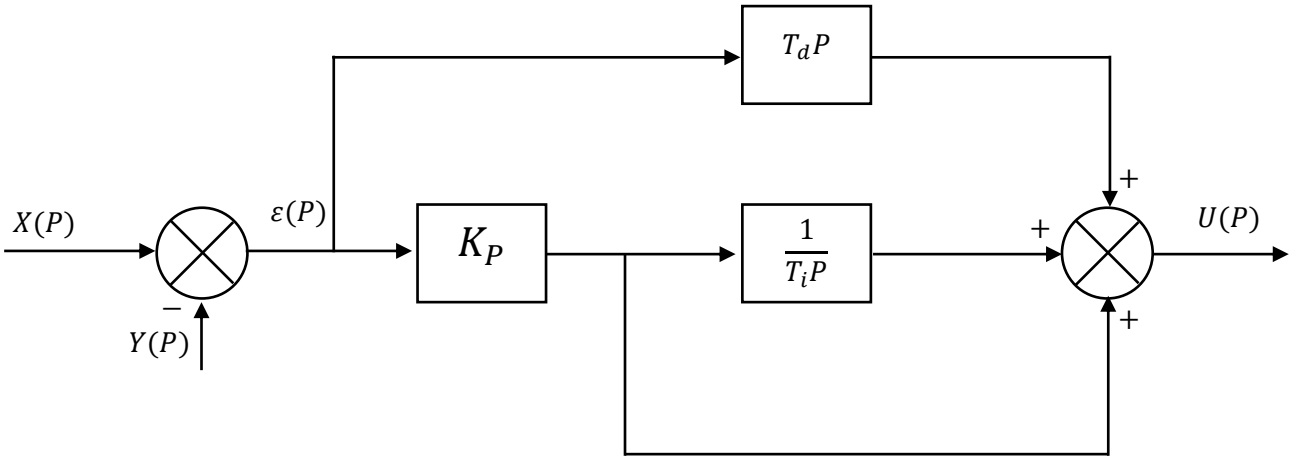


Figure I-19: 2nd Mixed Structure Controller

I.4.6- The use of PID control in industrial automation

PID control is widely used in industrial automation due to its ability to regulate and stabilize processes. Here are some additional ways to use PID control in industrial automation:

- **Temperature control**: PID controllers are commonly used to maintain a constant temperature in industrial processes, such as chemical reactions, heat treatment, and food processing.
- **Pressure control**: PID control is also used to regulate and stabilize the pressure in industrial processes, such as plastic manufacturing systems and pneumatic control.
- **Flow control:** A PID controller can be used to regulate the flow of liquids or gases in industrial processes, such as chemical manufacturing and water treatment.
- **Motion control:** The PID controller is used to control the speed, position, and direction of motors in industrial automation, such as in robotics and conveyor systems.
- **PH control:** The PID controller is commonly used in chemical and water treatment to maintain a constant pH level.
- **Level control:** A PID controller is used to control the level of a liquid or material in a tank or vessel, such as in oil and gas production and chemical production [9].

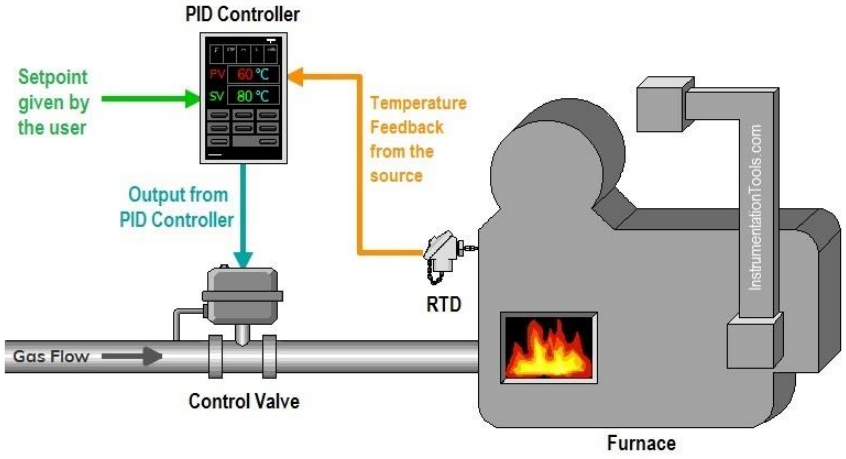


Figure I-20: PID controller, control valves

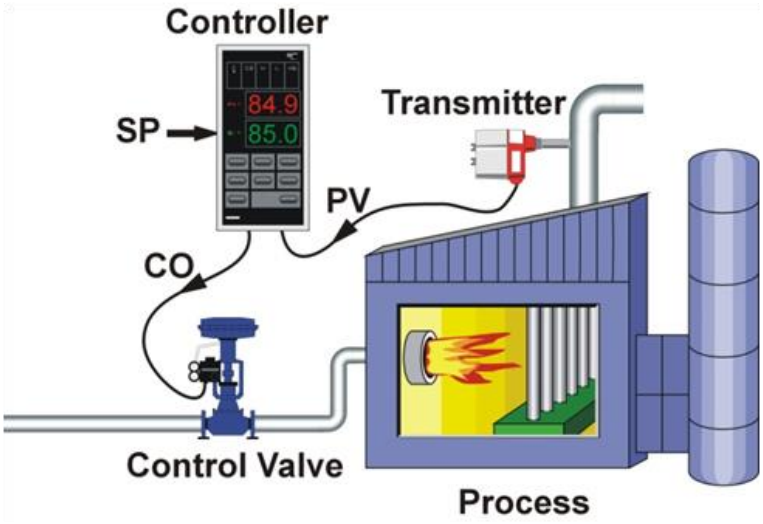


Figure I-21: PID controller performing automatic control of system

I.5-Hardware

I.5.1-Asynchronous Machine

The use of three-phase induction machines in industrial applications is determined by: simplicity of design, low price, high operational safety, and high technical performance (starting torque, high efficiency, etc.). The asynchronous machines are frequently employed as motors.

I.5.2-Asynchronous Motor

I.5.2.1-Definition

The asynchronous motor is the most widely used machine in the field of high power (greater than kilowatts) due to its numerous advantages such as robustness, power-to-weight ratio, low cost, ease of implementation, etc. This chapter aims to present the asynchronous motor. Thus, we provide an overview of the construction of an asynchronous motor and explain its operating principle, as well as discussing its coupling and winding. Finally, we present the advantages and disadvantages of this motor.[12]

I.5.2.2. Constitution of the asynchronous motor:

The asynchronous motor consists of two distinct parts:

A. The stator

A stator consisting of 3 coils positioned 120 degrees apart from each other and powered by a three-phase system, creating a rotating radial magnetic field in the air gap at the synchronous frequency.

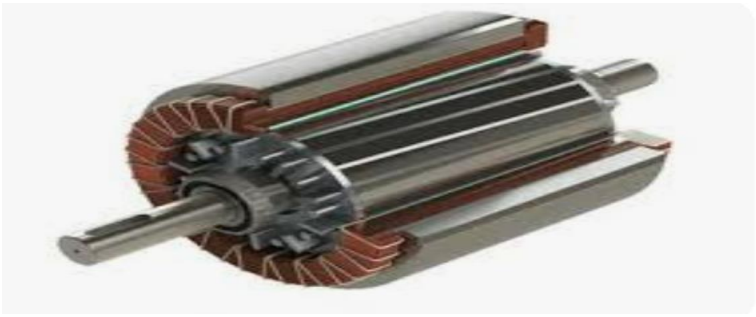


Figure I-22: Stator

B. The rotor

The rotor has the appearance of a squirrel cage. It is made up of identical metal bars, parallel or not to the axis of the rotor. These bars are connected to each other by two metal end rings, creating a short circuit. The interior is lined with thin metal sheets separated by an insulating varnish to limit eddy currents.

The rotor cannot rotate at the same speed as the magnetic field; otherwise, the cage would no longer be swept by the rotating field, resulting in the disappearance of induced currents and consequently the elimination of the Laplace forces and motor torque.

The two rotation frequencies cannot be synchronous, hence the name asynchronous motor.



Figure I-23: Rotor

It is the moving part of the motor. It is placed inside the stator and consists of a stack of steel laminations forming a cylinder keyed to the motor shaft. Among the most commonly used types, we distinguish:

a. Cage rotor (short-circuited rotor)

- ✓ **Single cage rotor:**

Conductors are placed in holes or slots arranged outwardly on the cylinder and parallel to its axis. At each end, these conductors are connected to a metal end ring. The assembly resembles a squirrel cage.

- ✓ **Double cage rotor**:

It consists of two sets of bars or conductors arranged in concentric circles.This design allows for better starting torque, higher efficiency, and improved resistance to rotor cage overheating.

b. Wound rotor (slip ring rotor):

In slots made on the laminations constituting the rotor, identical windings to those of the stator are housed. Generally, the rotor is three-phase. One end of each winding is connected to a common point (star connection). The free ends can be connected to a centrifugal coupling or to three insulated copper rings attached to the rotor. Graphite brushes connected to the starting device come into contact with these rings.

I.5.2.3. Mechanical components

The motor consists of a rotor and stator, as well as other mechanical components such as:

- Bearings
- Terminal plate
- Identification plate (or nameplate)
- End shields
- Rotor shaft
- Motor housing
- Fan
- Bearings

The motor frame serves as a support and provides protection against the external environment. The shaft is a transmission component, consisting of a central part that supports the rotor body and an end of the shaft on which a half coupling is attached. It is typically made of cast or forged steel. Its design depends on factors such as bending forces (centrifugal force, radial magnetic attraction, etc.), radial and tangential forces due to centrifugal forces, and torsional forces (electromagnetic torque transmitted in steady-state and transient conditions). The shaft is supported by one or more bearings, which support the rotor and ensure free rotation. The second bearing is free to accommodate thermal expansions of the shaft. One of the bearings is electrically insulated to eliminate currents in the shaft due to asymmetries in the magnetic

circuit reluctances. Bearings are typically of the rolling type for small and medium power machines.

I.5.2.4. Operating principle of an asynchronous motor

The operating principle of an asynchronous motor is based on two main aspects:

- On the one hand, an induced electric current is created in a conductor placed in a rotating magnetic field.
- On the other hand, the generation of a driving force on the considered conductor (through which a current pass and is placed in a rotating or variable magnetic field) whose direction is determined by the right-hand rule.

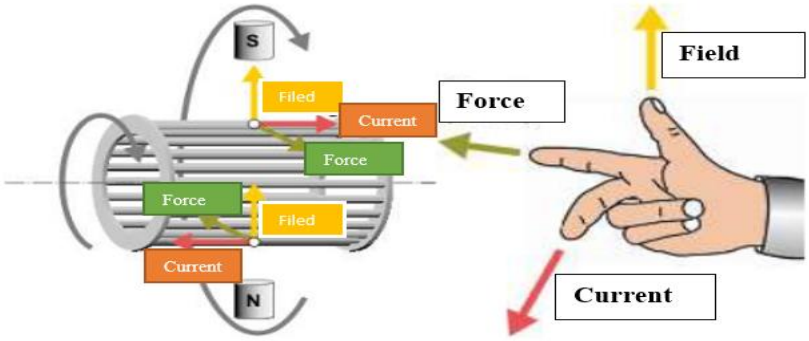


Figure I-24: Operation of a motor

As shown in the diagram above, the rotating field, at a given moment, is oriented upwards. When considering two diametrically opposite conductors, it can be observed that the induced currents in these two conductors are in opposite directions and, in association with the magnetic field, create opposing driving forces. Since the rotor is free to rotate about its axis, these two forces combine to impart torque to the conductors, enabling the rotation of the squirrel cage.

To maintain the motor's rotation, it is necessary to vary either the current in the conductors of the cage or the magnetic field. In an asynchronous motor, it is the magnetic field that changes in the form of a rotating field created in the stator.

Therefore, a rotor of an asynchronous motor never rotates at the synchronous speed (50 Hz). For a motor with one pair of poles (at 50 Hz, the rotating field's rotational speed is 3,000 [rpm]), the rotor's rotational speed may be, for example, 2,950 [rpm]; this is where the concept of slip comes into play.

- **Rotation direction**

When the motor is supplied with direct power, it rotates clockwise when facing the end of the shaft. By interchanging the supply of two phases, and only two phases, the rotation direction will be reversed.

I.5.2.5. Mechanical characteristics of asynchronous motors

A. No-load operation

Under no-load conditions, the motor does not drive any load, resulting in zero slip, and the motor operates at synchronous speed (g=0, ns =n).

The power factor is very low, and the absorbed current remains high. This is referred to as reactive or magnetizing current (which is used to create the magnetic field).

B. Loaded operation

The motor now delivers active power as a resisting torque is applied. As the resisting torque increases, the mechanical power required from the motor also increases.

- The current drawn increases.
- Cosφ increases.
- The rotational frequency decreases slightly.

C. Operating quadrants

Asynchronous motors can operate in four different ways, known as quadrants, defined as follows:

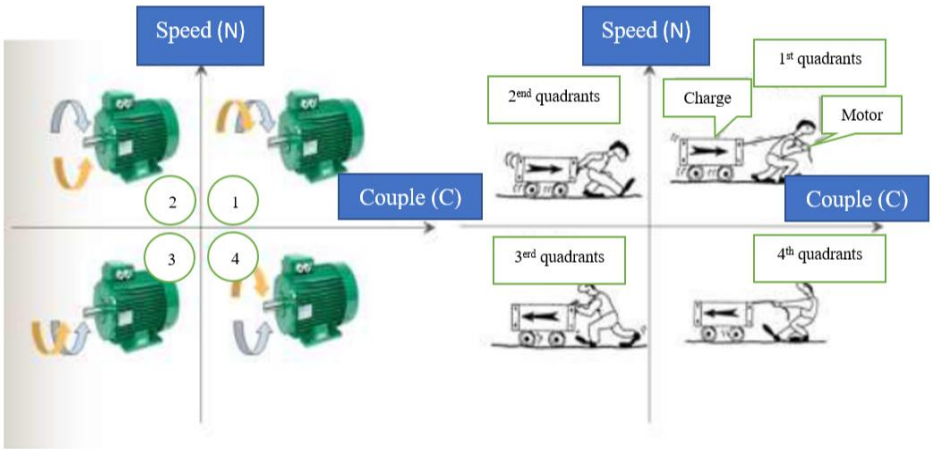


Figure I-25: Operating quadrants

- Quadrants 1 and 3: Motor operation. The rotating machine delivers mechanical power.
- Quadrants 2 and 4: Braking operation. The rotating machine absorbs mechanical power, and the motor becomes a generator.

I.5.2.6. Winding and connection of an asynchronous motor

A. Winding

The coils are housed in the slots of the stator. If there is one pair of magnetic poles for each of the three phases, the synchronous frequency is then 3000 rpm. By increasing the number of pole pairs, it is possible to obtain motors with different rotational frequencies.

Number of poles	2	4	6	8	10	12
Pole pairs	1	2	3	4	5	6
n_s (Tr/Min) under 50 Hz	3000	1500	1000	750	600	500
n_s (Tr/Min) under 60 Hz	360	1800	1200	900	720	600

Table I-1: Number of poles and synchronous speed.

The connection of the coils to the power supply is done at the terminal plate located on the top of the motor. There are 6 connections available, one for each end of the three coils. The terminals are connected to the coils according to the following diagram:

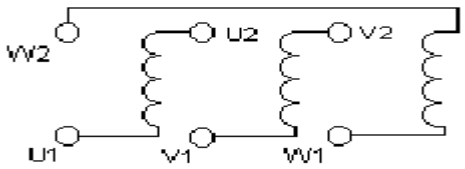


Figure I-26: The connection of the terminals to the coils.

B. Connection

The connection of an asynchronous motor can be done in a delta (triangle), wye (star), or star-triangle configuration.

a. Star Connection

The star connection is used when a 230V motor needs to be connected to a 400V power supply or when starting a motor with reduced power for a load with high mechanical inertia. This connection is made as shown in the figure below.

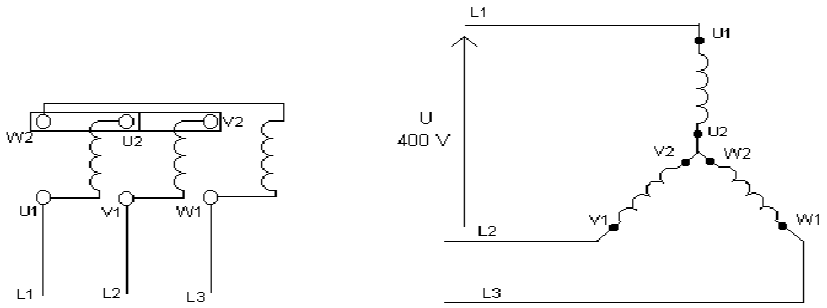


Figure I-27: Star configuration

b. Triangle (delta) Connection

In the triangle configuration, each coil is supplied with the nominal voltage (400V) as shown in the figure below.

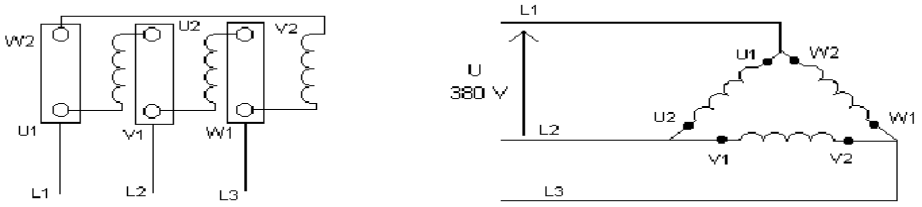


Figure I-28: Triangle configuration

c. Star-triangle Connection

The star-triangle starting method is widely used in electrical engineering for the startup of three-phase asynchronous electric motors. This system is used to reduce the risks associated with direct starting. In fact, the current intensity at startup is very high compared to the motor's nominal current (approximately 5 to 7 times the rated current). In the case of large motors, these high currents lead to overload on the power lines and the protection and control devices, resulting in wear and potentially premature failure of the starter components.

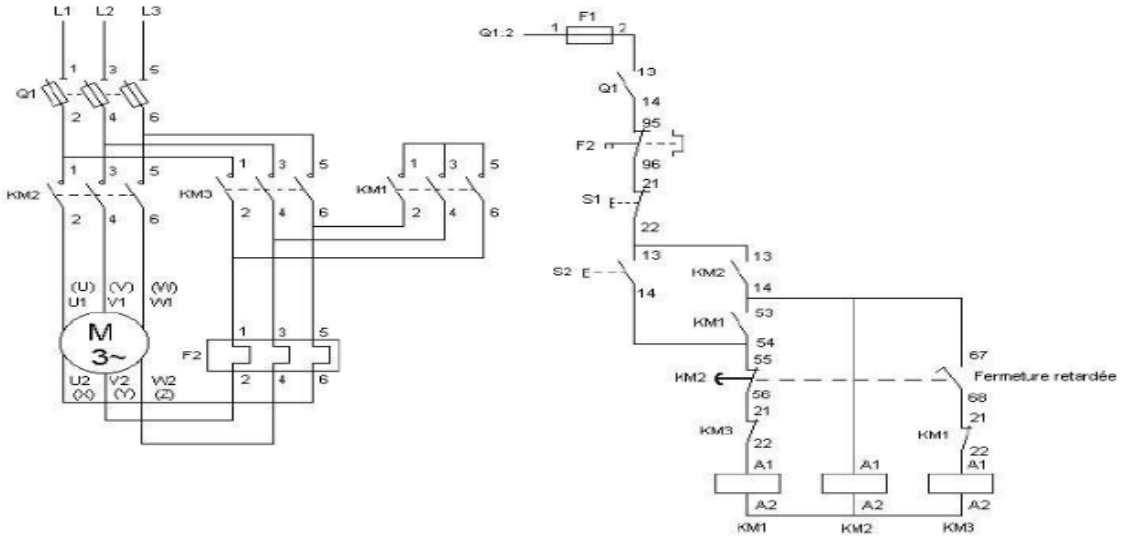


Figure I-29: Star-triangle Connection Diagram.

I.5.2.7. Power Balance and Formulas

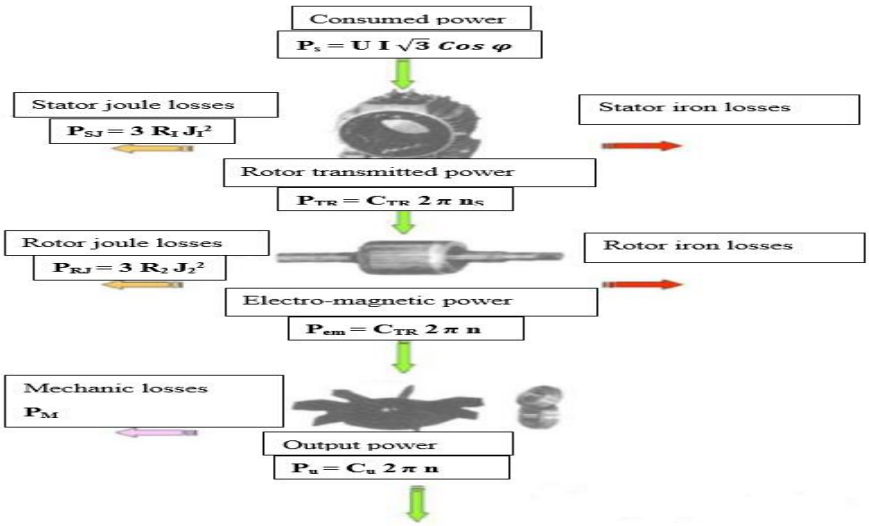


Figure I-30: Power balance of the asynchronous motor

A. Power:

a. Expression of the absorbed power

This is the absorbed power or electrical power supplied to the machine

$$Pa = U.\ I.\ \sqrt{3}.\ \text{Cos}\ (\varphi) = 3V.\ I\ \text{Cos}\ (\varphi) \qquad \text{I.17}$$

b. Reactive power

Reactive power represents the power generated by reactive elements in the circuit (capacitive or inductive reactance). It is measured in volt-amperes reactive (VARs).

$$Q = U.\ I\sqrt{3}\ Sin\ (\varphi) \qquad \text{I.18}$$

c. Apparent power

Apparent power is the power supplied to the load. It is measured in volt-amperes (VA). It is calculated as follows:

$$S = U.\ I\sqrt{3} \qquad \text{I.19}$$

Where: $$U = V\sqrt{3} \qquad \text{I.20}$$

d. Transmitted power

This is the power received by the rotor:

$$Ptr = Pa - Pjs - Pfs \qquad \text{I.20}$$

e. Total mechanical power

The electromagnetic torque Tem drives the rotor at speed Ω, thus transmitting the total mechanical power PM.

$$PM = Tem\ \Omega \qquad \text{I.21}$$

Or: $$PM = Tem\ \Omega = Ptr\ \frac{\Omega}{\Omega s} = Ptr(1 - g) \qquad \text{I.22}$$

This power includes useful power and mechanical losses.

f. Output power

It is the mechanical power minus the mechanical losses. The power indicated on the nameplate of an asynchronous motor is the nominal useful power. It is calculated as follows:

$$Pu = PM - Pm \qquad \text{I.23}$$

The output torque is then calculated using the following equation:

$$Tu = \frac{\text{Pu}}{\Omega} \qquad \text{I.24}$$

B. Losses:

a. Stator copper losses

The stator copper losses are due to the flow of currents in the stator windings.

$$Pjs = \frac{3}{2}\ \text{RI}^2 \qquad \text{I.25}$$

b. Stator iron losses

Noted as Pjs, they include losses due to hysteresis and eddy currents in the metal masses of the stator.

c. Rotor copper losses and iron losses

These losses represent the difference between Ptr and PM. They are due to induced currents. They are not directly measurable as the rotor is short-circuited, so they need to be calculated.

$$Pjr + Pfr = Ptr - PM = Ptr - Ptr\,(1 - g) = gPtr \qquad \text{I.26}$$

$$Pjr \approx gPtr$$

Rotor iron losses are negligible.

d. Mechanical losses

These losses correspond to friction in the bearings, ventilation, etc. They are often neglected in most cases.

e. Stray losses

These losses depend only on U, f, and n. Since these quantities are usually constant, the stator iron losses and mechanical losses are also constant.

$$Pc = Pfs + Pm \qquad \text{I.27}$$

The loss torque is defined as:

$$Tp = \frac{Pc}{\Omega s} \qquad \text{I.28}$$

C. Electromagnetic torque

The forces acting on the rotor conductors rotate at the speed Ωs, while they slip on the rotor, which rotates at the speed Ω.

The action of all electromagnetic forces reduces to an electromagnetic torque resulting from the moment Tem.

$$T\text{em} = \frac{Ptr}{\Omega s} \qquad \text{I.29}$$

D. Efficiency

Efficiency is equal to the ratio of useful power to absorbed power. It is always less than 1.

$$\eta = \frac{Pu}{PA} < 1 \qquad \text{I.30}$$

E. Speeds

Synchronous speed: It is the rotational speed of the rotating field:

$$ns = \frac{60.f}{p} \quad \text{I.31}$$

Motor rotation speed:

$$n = ns\ (1 - g) \quad \text{I.32}$$

Angular speed:

$$\Omega = 2\ \pi\ n \quad \text{I.33}$$

F. Slip

Slip is the relative difference between the machine's rotation speed and synchronous speed.

$$g = \frac{ns-n}{ns} \quad \text{I.34}$$

Note:

- At startup: g = 100%
- During operation: g < 1 as long as the load does not drive the motor.
- Under nominal conditions: gn = 3 to 10%

I.5.2.8. Speed control of an asynchronous motor

Controlling the rotational speed of an asynchronous motor is crucial for many applications.

The following relationship helps identify the parameters that can influence the rotational speed.

We have:

$$g = (n_0 - n) / n_0 \quad \text{I.35}$$

Where:

- g = slip [%].
- $n0$ = synchronous speed [rpm].
- n = motor shaft speed [rpm].

Or:

$$n=\frac{(1-s)(fx60)}{p} \quad \text{I.36}$$

With:

- f x 60: network frequency [Hz] multiplied by the number of seconds per minute.
- p = the number of pole pairs.
- Therefore, we can control the rotational speed by intervening on:
- The number of pole pairs (e.g., a two-speed motor),
- The slip of the motor (wound-rotor motor),
- The network frequency.

A. Control by Modifying the Number of Poles

Older systems still operate with two-speed motors. Most of the time, these are motors with a rotor composed of two different numbers of pole pairs. The windings are arranged in the stator slots in a particular way that adds to their complexity. The different pole pair couplings allow for different speeds.

B. Frequency Regulation

Currently, the speed control of asynchronous motors is achieved electronically through variable frequency drives. Therefore, we will only discuss frequency control here, as it is by far the most common method.

I.5.2.9. Advantages and Disadvantages of an Asynchronous Motor

A. Advantages

- The asynchronous motor is easy to implement.
- It is robust and has low maintenance costs.
- Good standardization among manufacturers.
- Direct starting on the network (high starting torque).
- Its efficiency is close to 99%.

B. Disadvantages

- For high-power motors, the starting time is long.
- If the load torque exceeds a certain limit, the motor stalls.
- An excitation is required, meaning a second source of energy.

Without power loss, the motor speed can be controlled by varying the frequency because the speed of the rotating field in the stator changes. To maintain the motor torque, the motor voltage must change with the frequency in a constant ratio.

I.5.3. Variable Frequency Drive:

I.5.3.1. Definition of a Variable Frequency Drive

A variable frequency drive (VFD) is a device that allows for the adjustment of the speed of a motor, which is necessary for many industrial processes. In fact, most motors operate at a constant speed. To modulate the speed of process equipment, various mechanical devices were traditionally used. The industry now demands precise speed control, which has led to the use of electronically controlled direct current (DC) motors with semiconductor-based VFDs. This technique involves varying the speed proportionally to the voltage.

In addition, variable frequency drives are also used with alternating current (AC) motors. These VFDs utilize pulse width modulation (PWM) and insulated gate bipolar transistors [14].



Figure I-31: Variable speed drive (ATV12h018m2)

I.5.3.2. Constitution of a Variable Speed Drive:

The frequency converter, supplied with a fixed voltage and frequency from the power grid, provides the motor with alternating current at variable voltage and frequency according to speed requirements.

To properly supply power to an asynchronous motor with constant torque regardless of the speed, it is necessary to maintain a constant flux. This requires the voltage and frequency to vary simultaneously and in the same proportions.

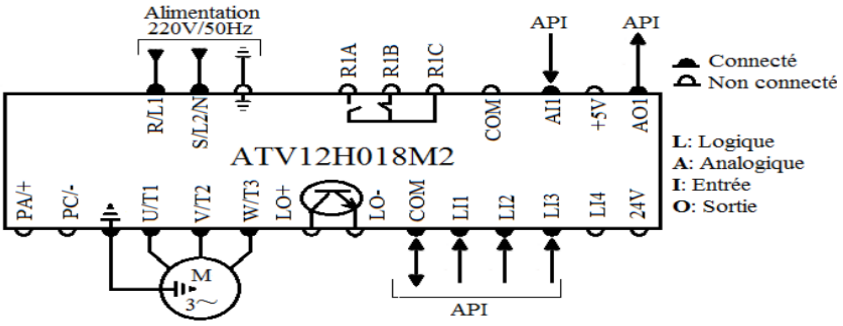


Figure I-32: VFD Wiring Diagram (ATV12H018M2) [11]

I.5.3.3. The Control Circuit

This circuit must ensure four essential functions:

- Control of the semiconductor devices of the variable speed drive.
- Exchange of control, regulation, and analysis information with peripherals.
- Fault control (interpretation and display).
- Protection of the variable speed drive and the motor.

The advent of microprocessors has allowed for increased speed in executing control and regulation information within the control circuit compared to other circuits (intermediate circuit, inverter, etc.).

The speed regulation of three-phase AC motors evolves according to two different control principles:

- U/f control (Scalar).
- Vector control of flux (VVC).

These principles determine the programming of control algorithms and speed control of variable speed drives. Both methods have advantages depending on specific performance requirements (torque, speed, etc.) and drive precision.

I.5.3.4. U/f control (Scalar)

U/f control is based on the measurement of scalar quantities (amplitude values in voltage and frequency). It is the basic control system of standard frequency converters.

In order to maintain a constant flux in the motor and thus a constant torque speed variation, the voltage and frequency vary proportionally up to the motor's nominal frequency (50 Hz). When the nominal voltage is reached and the voltage can no longer increase, it is still possible to increase the frequency. In this case, the variation occurs at constant power, and the torque

decreases with speed. This operating mode is interesting for loads with constant torque, such as motors. Indeed, the motor torque "matches" better with the profile of the resisting torque, resulting in lower energy consumption.

The figure below shows the profiles of torque curves as a function of speed for different U/f ratios:

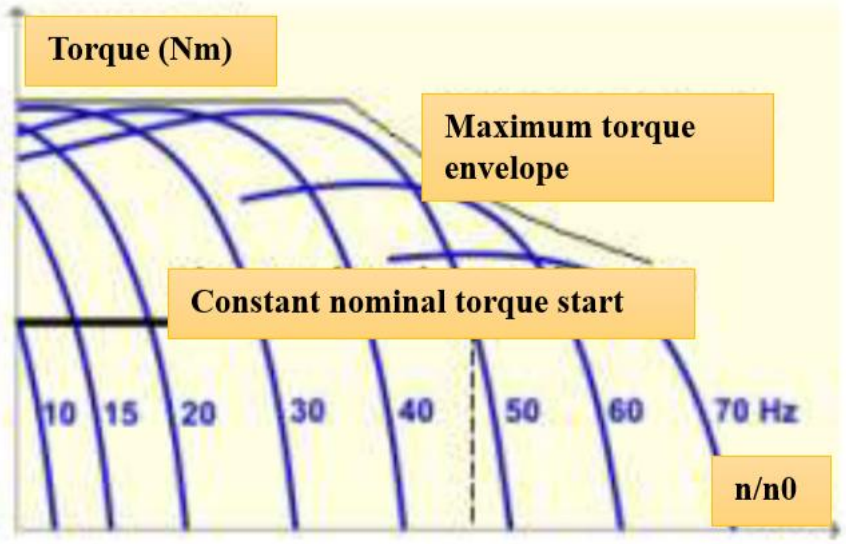


Figure I-33: constant torque operation

I.5.3.5. The advantages of U/f control are:

- Easy adaptation of the speed controller to the motor.
- Easily handles load variations across the entire speed range.
- Motor torque remains more or less constant depending on the speed.

I.5.3.6. The disadvantages of U/f control are:

- Speed regulation range is limited to 1/20.
- At low speeds, no compensation for slip and load management.

I.5.3.7. Vector control of voltage (or flux)

For this type of control, it is necessary to provide precise indications of the motor parameters (nameplate encoding). Vector control of voltage (VVC) operates based on the principle of calculating the optimal magnetization of the motor at different loads using compensation parameters to control the slip and motor load. As the name suggests, vector control of voltage works with the voltage vectors at no-load and compensation vectors in relation to load variation. Field-oriented vector control operates with the values of active currents, magnetization (flux),

and torque. Through an appropriate mathematical model, it is possible to determine the required torque for the motor based on the stator flux and rotor current vectors, in order to optimize and regulate the magnetic field and motor speed according to the load.

I.5.3.8. Advantages of VVC control:

- Good response to load variations.
- Full torque at zero speed and precise speed regulation.
- Performance is similar to DC drives.
- Quick response to speed variations and wide speed range (1/100).
- Better dynamic response to changes in the rotation direction.
- Only one control strategy is needed for the entire speed range.

I.5.4-Industrial Programmable Automation (API)

Industrial programmable controllers or APIs are electronic machines, programmable by non-technical staff, intended to control in an industrial environment and in real-time, sequential logical processes.

I.5.4.1-Automated System Functionalities

- Gather data using detectors and sensors.
- Process the data using an ordering party such as an ATM, etc.
- Issuing orders via pre-actuators such as distributors, contactors, and dimmers.
- Act on the working material with the operating portion (cylinders, motor).

Other system functions that can be automated include:

- Managing.
- Coordinating.
- Dialoguing.
- Communicating.
- Monitoring.

I.5.4.2-The automaton's architecture

The automation systems are made up of two primary sections that are linked together:

- **Control Part:** the operator enters the program data received from the sensors to the computer PC, it sends orders executed by the operative part.
- **Operative Part:** it sends information back and gives a signal to the PC emits by actuators or sensors requiring the consumption of electrical energy. the operator appreciated the smooth running of his work according to these signs [16].

I.5.4.3-Cyclic operation of APIs

Most APIs have cyclic operation according to the synchronous status of inputs and outputs:

- Internal data processing: ordering and updating the system.
- Support for inputs: which are frozen in memory.
- Program executions: the processor performs the action.

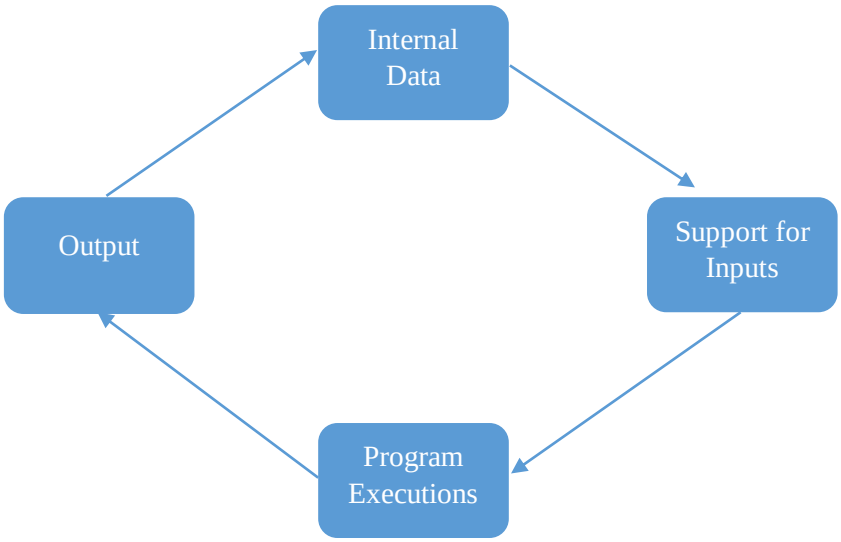


Figure I-34: Cycle for an API

I.5.4.4-Automate API S7-300

To suit our automation demands, the S7-300 controller provides the flexibility and capability to operate a wide range of devices. Its small size, adaptability, and huge number of instructions make it a perfect choice for managing a wide range of applications.

We have chosen the S7-300 CPU313C PLC. The selected programmable logic controller is characterized by a 64Kb memory, 24 inputs/16 outputs for discrete inputs/outputs, 5 inputs/2 outputs for analog signals, and a Profibus-DP (Decentralized Peripherals) communication [15].



Figure I-35: Programmable Logic Control S7-300

I.5.4.5. Characteristics of the S7 300 Programmable Logic Controller

Table I.2 contains some technical characteristics of the Siemens S7-300-CPU 313C.

- Integrated working memory	64 k octet
- Number of inputs and outputs	Inputs TOR:24 , output TOR:16 Inputs analog : 4(Voltage or current measurement) and 1 (Resistance or temperature measurement) output analog :2
- MPI communication port	yes
- Number of timers	256
- PID control	yes
- Profibus communication	Yes

Table I-2: Technical Specifications of S7 300 CPU 313.

I.5.4.6. Components of an S7-300:

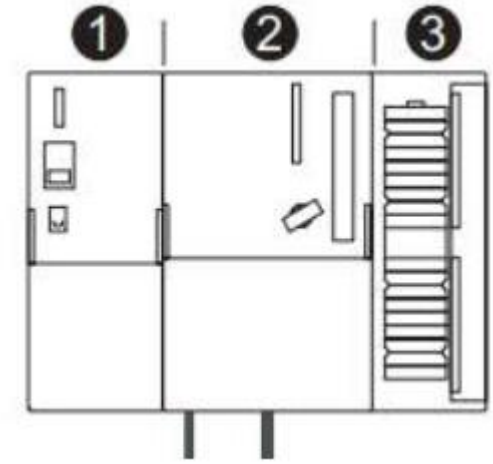


Figure I-36: Components of an S7-300

- ✓ A - Power Supply
- ✓ B - CPU Modules
- ✓ C - Input/output Modules

A. Power Supply:

It converts the mains voltage (AC 120/230 V) into a 24 V DC supply voltage and provides power to the S7-300 as well as external power supply for 24 V DC load circuits.

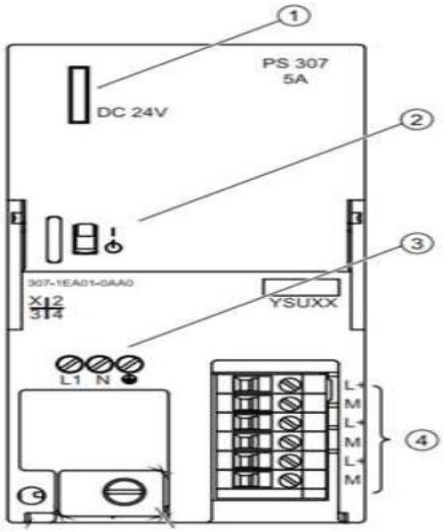


Figure I-37: Power supply of the S7 300 PLC.

- ✓ 1 - Power On LED display
- ✓ 2 - ON/OFF switch
- ✓ 3 - 24V power input
- ✓ 4 - 24V output

B. CPU:

Its main role is to execute the user program, power the internal bus of the S7-300, and communicate with other partners on an MPI network via the MPI interface.

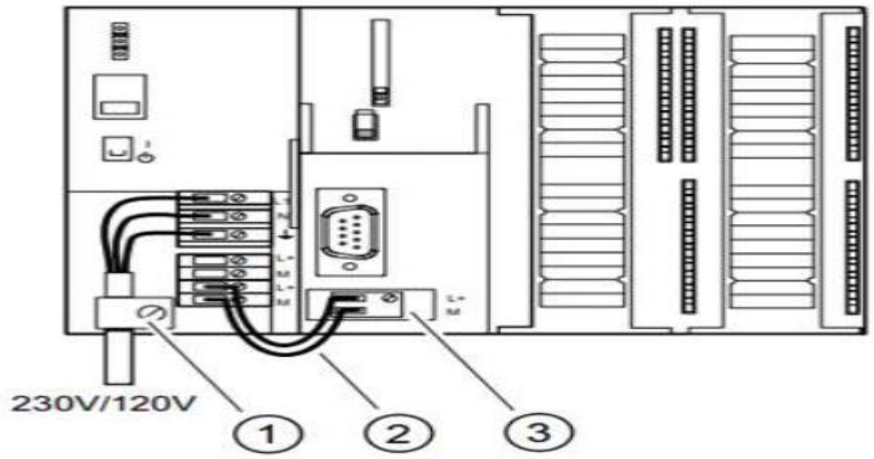


Figure I-38: CPU connection

- ✓ 1- Power pull-out stop collar
- ✓ 2- Connection cables between power supply (PS) and CPU
- ✓ 3- Plug-in connection for power supply
- ✓ 4- MPI input
- ✓ 5- ON/OFF switch
- ✓ 6- Input for memory card

C. Input/output modules

They adapt the different signal levels of the process signals to the S7-300. The built-in inputs and outputs of this CPU have the following addresses in Table I-3.

Inputs / outputs	Default address.
24 input TOR	124.0 to 126.7 Including 12 inputs for technological functions. 124.0 to 125.0 125.4 to 125.6
16 output TOR	124.0 to 125.7 Including 3 outputs for technological functions. 124.0 to 124.2
4+1 inputs analog	752 to 761
2 output analog	752 to 755

Table I-3: Inputs/outputs and address

The address of an input or output of a discrete I/O module consists of the byte address and the bit address.

Example: The example consists of the following elements:

- Input E
- Byte address 1
- Bit address 2

The byte address is based on the initial address of the modules. The bit address is indicated on the module. If the first discrete I/O module is placed in slot 4, it has a default starting address of 0. The starting address of each subsequent discrete I/O module increases by 4 per slot. The following figure illustrates the scheme by which addresses are created for the different channels of the discrete I/O module.

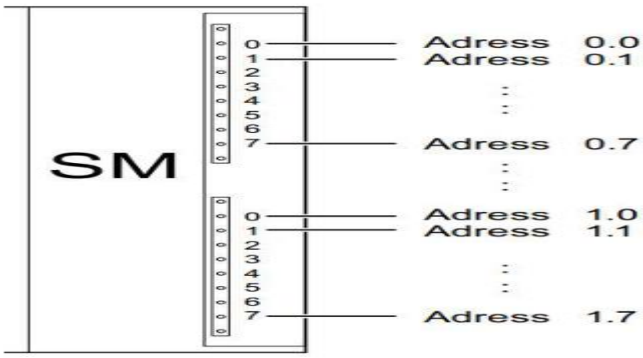


Figure I-39: Address bus.

I.6-Conclusion

In this chapter, we have given an overview of industrial automation, including its importance. We also presented the key components of an automated system with various industrial control systems. Additionally, we explored the different types of fixed and flexible automation. On the other hand, we have introduced the definition of synoptic, their importance, and their different types. Finally, we described the PID controller and its components and how it will be used in industrial automation. We have also described the hardware part comprising the asynchronous machine and more specifically the asynchronous motor. We also discussed the variable speed drive and its role in controlling the speed of the machine. Additionally, we have introduced Industrial Programmable Automation and S7-300 PLC, which are crucial components of modern industrial control systems.

Chapter II

Industrial automation Software

II.1-Introduction

Program software is very essential in modern industrial automation systems. They allow operators to control and monitor various processes and equipment. This chapter, entitled "Industrial automation software", discusses the importance of industrial automation software. It also gives an overview of the main software used in this field. We will explore the TIA Portal software and its features, benefits, and integration with the S7-300 API. Additionally, we will cover the LabView software, its capabilities, benefits, and integration with: the S7-300 API and the Tia Portal. Additionally, we will discuss NI OPC Server software, its benefits, and its integration with: S7-300 API, LabView, and Tia Portal. At the end of this chapter, we will present the main software that stimulates innovation and their role in industrial automation.

II.2-Software Programs

Software programs, often known as computer software or just software, are a set of instructions or programs that guide a computer on how to do certain tasks. These programs are intended to carry out certain jobs or activities, ranging from simple functions like word processing and data management to more complicated duties like industrial automation.

Software programs are an important component of modern industrial automation systems because they give the tools needed to design, control, and monitor industrial processes. These applications are used to design and manage the control logic for various automation systems, as well as to provide a graphical interface for monitoring system performance and making adjustments as needed [16].

II.2.1-Importance of Software Programs

Industrial automation relies heavily on software applications. They are critical instruments for managing and monitoring industrial processes, as well as assuring their peak performance. Using software programs in industrial automation has various advantages, including:

- **Increased Productivity:** Software applications aid in the automation of repetitive operations, lowering the time and effort necessary to accomplish them. This speeds up and improves the efficiency of industrial operations, resulting in greater production and cost savings.
- **Improved Accuracy:** Because software programs automate operations, they reduce human mistakes and inconsistencies that might arise in manual procedures. This

increases the accuracy and precision of industrial operations, resulting in higher product quality and less waste.

- **Remote Access:** Software packages enable remote access to industrial systems, allowing monitoring and control from anywhere in the globe. This is especially beneficial for businesses with many sites or those who operate in remote places.
- **Real-time Monitoring:** Software programs monitor industrial processes in real time, allowing for the rapid discovery of issues and fast intervention to prevent downtime and production losses.
- **Data Management:** Software applications may gather and analyze data from industrial processes, offering significant insights into performance, efficiency, and prospective development areas. This information can help to improve procedures and make data-driven decisions [16].

II.2.2-Main Software Programs used in industrial automation

There are a variety of software packages used in industrial automation, each with its unique set of features and capabilities. Here's a rundown of some of the most important software applications used in industrial automation:

- **TIA Portal:** Siemens created the TIA Portal engineering software. It is intended to simplify automation procedures and shorten engineering time. The TIA Portal is a unified platform for programming and customizing Siemens automation systems such as PLCs, HMIs, and drives.
- **LabVIEW:** National Instruments created LabVIEW, a graphical programming language. In industrial automation, it is commonly used for data collecting, instrument control, and automated testing. Users may utilize LabVIEW to create bespoke software programs for a variety of industrial automation tasks.
- **NI OPC Server:** National Instruments' NI OPC Server is a software package. It communicates with OPC servers, which are widely used in industrial automation to offer a standard interface for communication between various devices and software applications.
- **CODESYS:** CODESYS is a software suite for programming PLCs, HMIs, and other automation devices. It provides a versatile and scalable framework for creating industrial automation applications and supports a broad range of programming languages.
- **FactoryTalk View:** Rockwell Automation created the FactoryTalk View software package. It is employed in the design and management of HMI applications for

industrial automation. FactoryTalk View has a number of functions such as data logging, trending, and alarms and is intended to be used in combination with Rockwell Automation's PLCs and other automation devices [17].

II.3-Tia Portal (Totally Integrated Automation Portal)

II.3.1-Software Description

Siemens' TIA Portal is a software platform for industrial automation. It is used to program and configure Siemens PLCs, human-machine interfaces (HMIs), and other automation components. From design and planning through commissioning and maintenance, TIA Portal provides a full range of tools for the whole automation project. The simple user interface of TIA Portal is one of its main features, allowing engineers and operators to simply build and manage their automation projects. It supports a variety of programming languages, including ladder logic, function block diagram (FBD), structured text (ST), and sequential function chart (SFC), making it an extremely versatile tool for automation programming. Integration of TIA Portal with other Siemens automation technologies, such as SIMATIC controllers and SIMATIC WinCC for HMI development, is another advantage. This integration enables component-to-component communication and aids quick programming and debugging [18].

TIA Portal is a strong industrial automation solution that provides a complete set of functions and a simple user interface to make automation programming more efficient and effective.

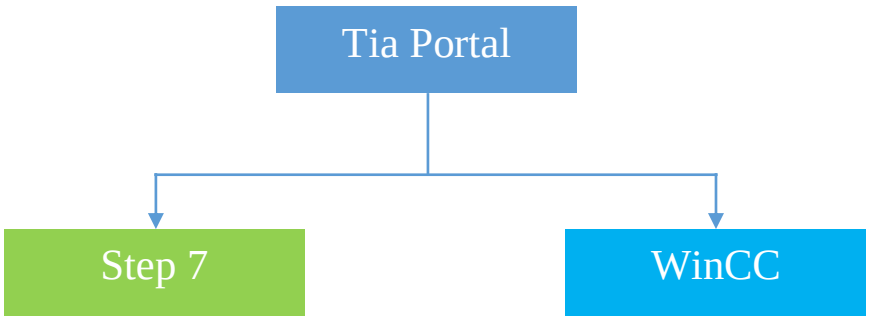


Figure II-1: Illustration of the construction of Tia portal

II.3.1.1-Simatic Step 7

SIMATIC STEP 7 is the world's best known and most widely used engineering software for industrial automation. SIMATIC STEP 7 Version 13 is an engineering system running under the Totally Integrated Automation Portal (TIA Portal) software. With SIMATIC STEP 7 V13, the user can configure, program, test and diagnose all SIMATIC controllers [19].

II.3.1.2-Simatic WinCC

WinCC (TIA portal) is an engineering software for the configuration of SIMATIC desks, SIMATIC industrial PCs and standard PCs by the visualization software. The SIMATIC WinCC in the TIA portal is part of a new integrated engineering concept that offers a homogeneous engineering environment for programming and configuration of control solution, training visualization, it is the software for all HMI applications ranging from simple control solutions with basic panels to SCADA applications for multi-postal PC-based systems [19].

II.3.2-Views of Tia Portal

When TIA Portal is launched, the work environment consists of two types of view:

- **View of the portal:** It is task-oriented and very quick to get started.
- **Project View:** It includes a tree with the different elements of the project, the required editors open according to the tasks to be performed. Data, settings and editors can be viewed in one view [20].

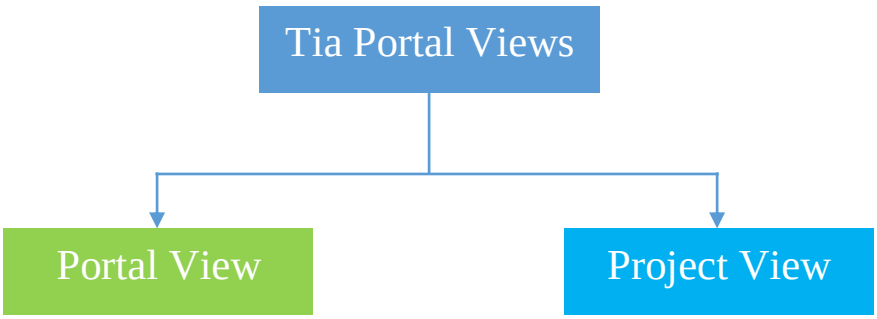


Figure II-2: Tia Portal Views

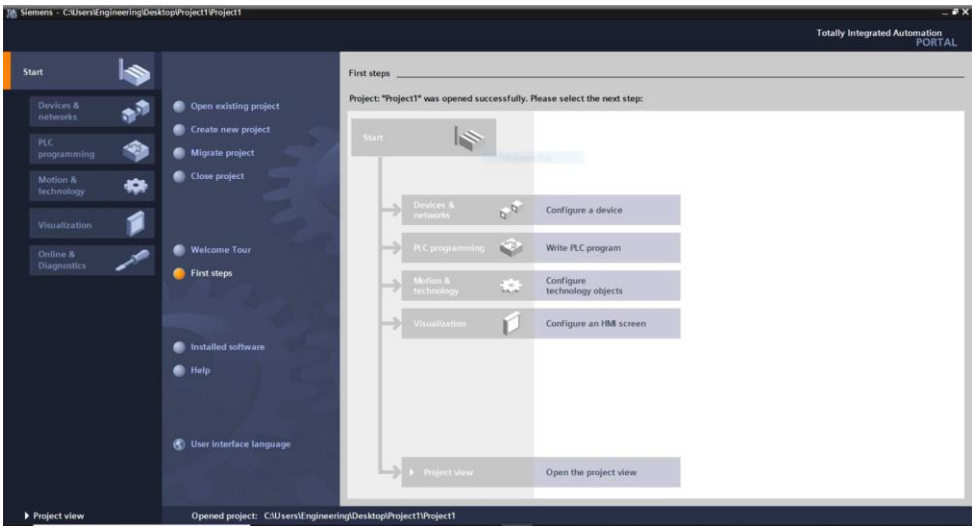


Figure II-3: Portal View

II.3.3-Tia Portal Project View

The "Project" element contains all the necessary elements and data to implement the desired automation solution.

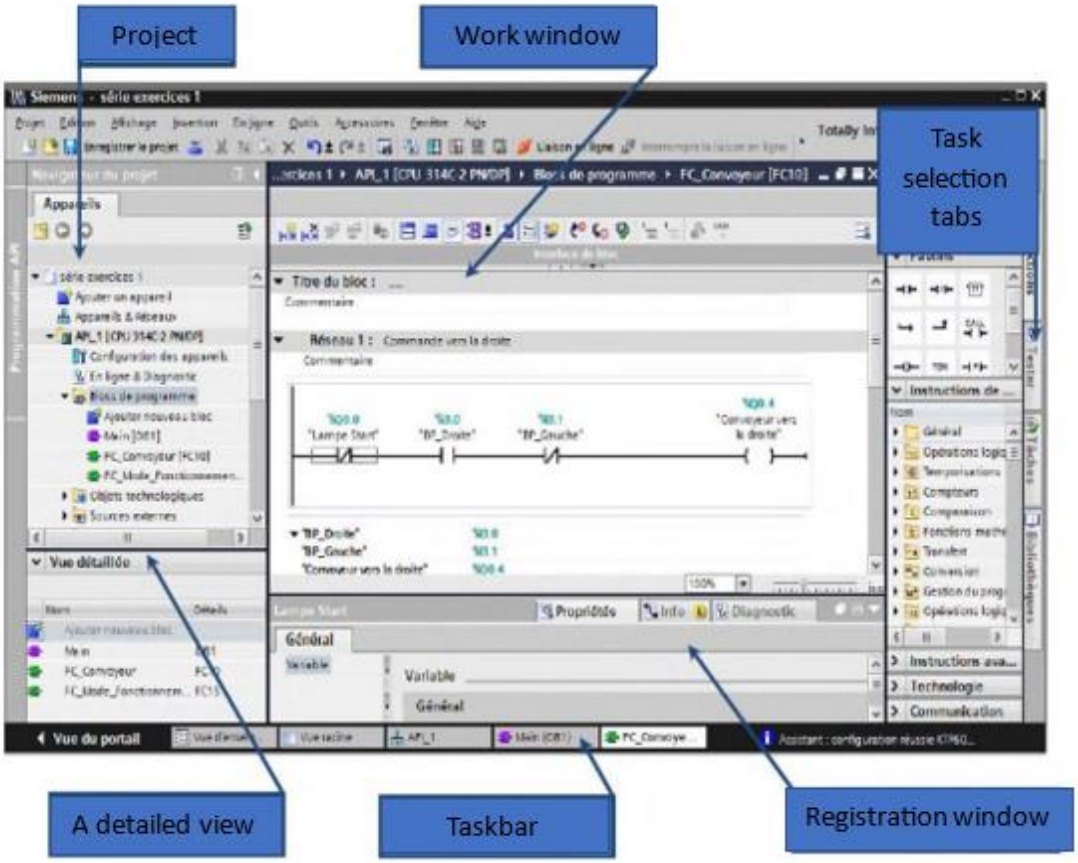


Figure II-4: Project View

- **The work window:** Allows you to view the selected objects in the project to be processed. These can be hardware components, program blocks, variable tables, HMIs, etc.
- **The inspection window**: Allows you to view additional information about a selected object or ongoing actions (hardware properties selected, error messages during program block compilation, etc.).
- **The task selection tabs**: Are content that varies depending on the selected object (hardware configuration, component libraries, program blocks, programming instructions).

This working environment contains a lot of data. It is possible to hide or minimize certain windows when not in use. It is also possible to resize, rearrange, and undock the different windows [20].

II.3.4-Addressing of I/O's

To know the addressing of inputs and outputs present in the hardware configuration, you need to go to "Devices and Networks" in the project browser. In the work window, you must make sure to be in the "Device view" tab and select the desired device [20].

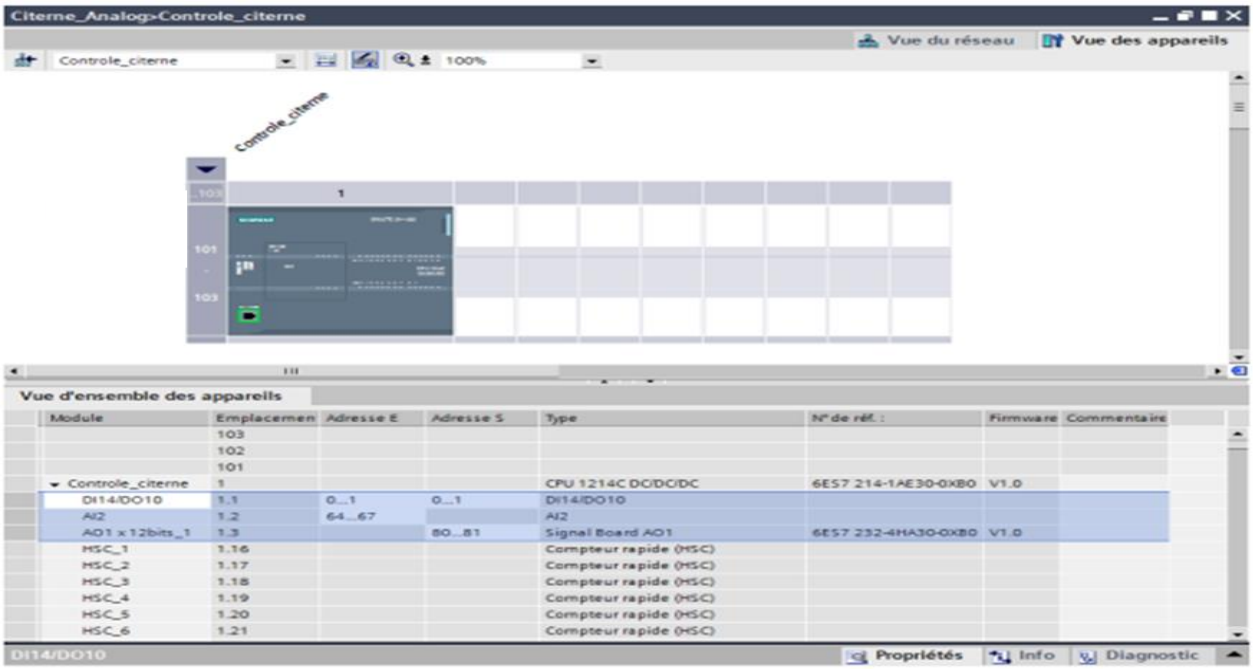


Figure II-5: Addressing I/Os

II.3.5-API variables

- Symbolic and absolute addressing In TIA Portal, all global variables (inputs, outputs, markers, etc.) have both a symbolic address and an absolute address.
- The absolute address represents the operand identifier (I, Q, M, etc.) and its address and bit number.
- The symbolic address corresponds to the name that the user gave to the variable (e.g. Start Button).
- The link between symbolic and absolute addresses is made in the API variable table.
- During programming, one can choose to display absolute addresses, symbolic addresses, or both simultaneously [20].

II.3.6-API variable table

It is in the API variable table that one can declare all variables and constants used in the program. When defining an API variable, one must define:

- A name: this is the symbolic addressing of the variable.
- The data type: BOOL, INT, etc.

- The absolute address: for example, Q1.5
- One can also insert a comment that provides information about this variable.
- The comment can be visible in each network using this variable [20].

II.3.7-The Using of Tia Portal with the API S7-300

TIA Portal is a powerful engineering software for industrial automation that supports various devices, including PLCs, HMIs, and drives. The software enables users to configure, program, test, and diagnose automation systems efficiently. TIA Portal also integrates with other Siemens software, such as SIMATIC WinCC for visualization and SCADA, and SIMOTION SCOUT for motion control applications.

- Create a project: Open TIA Portal and start a new project by selecting the kind of device to be used (in this example, API S7-300).

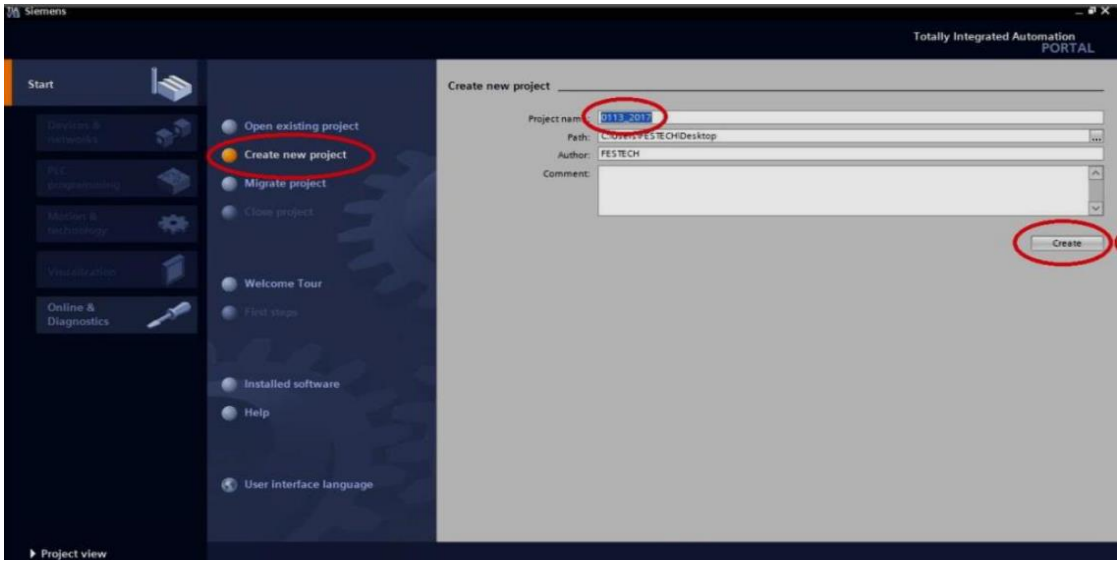


Figure II-6: Create a Project.

- **Configure the device:** After you have created a new project, you'll need to configure the API S7-300 by entering hardware attributes like IP address, name, and communication settings. You may accomplish this in TIA Portal by utilizing the "Device Configuration" option.

  We will first add the CPU, if you do not know the reference you can click on « PLC S7-300 CPU 314C-2 PN/DP », then you can enter it later or the hardware configuration will update when connecting with the said CPU.

  We will add a CPU 314C-2 PN/DP 6ES7 314-6EH04-0AB0

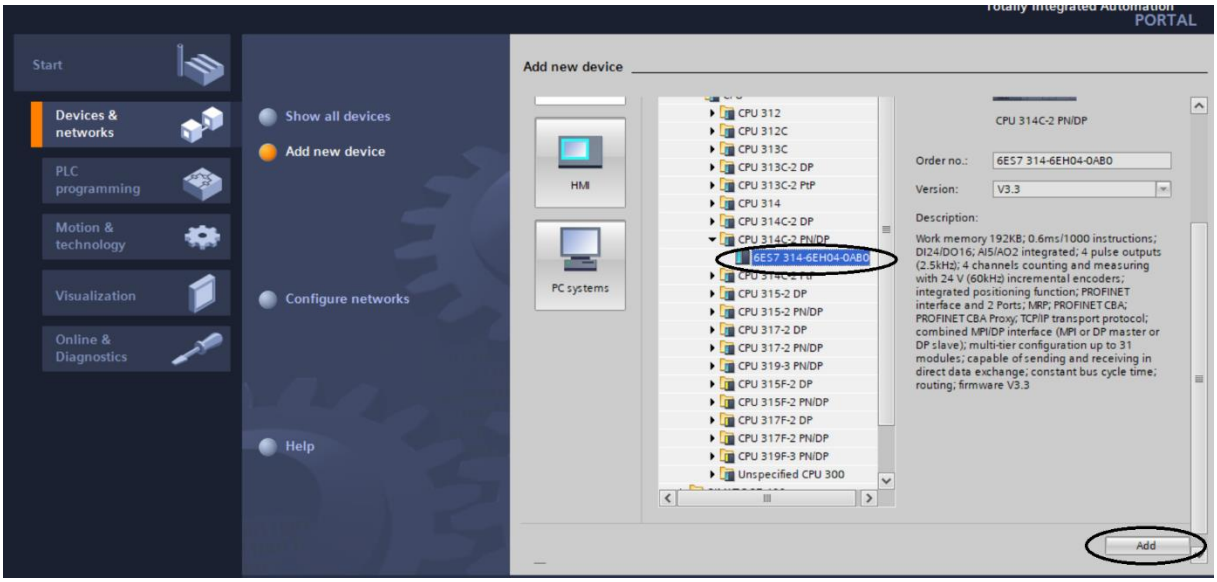


Figure II-7: Configure device

By clicking on your API, you will see other tabs in the properties window. On the "PROFINET Interface" tab and change the address: 192.168.0.1 to 192.168.0.10 and leave the mask on: 255.255.255.0.

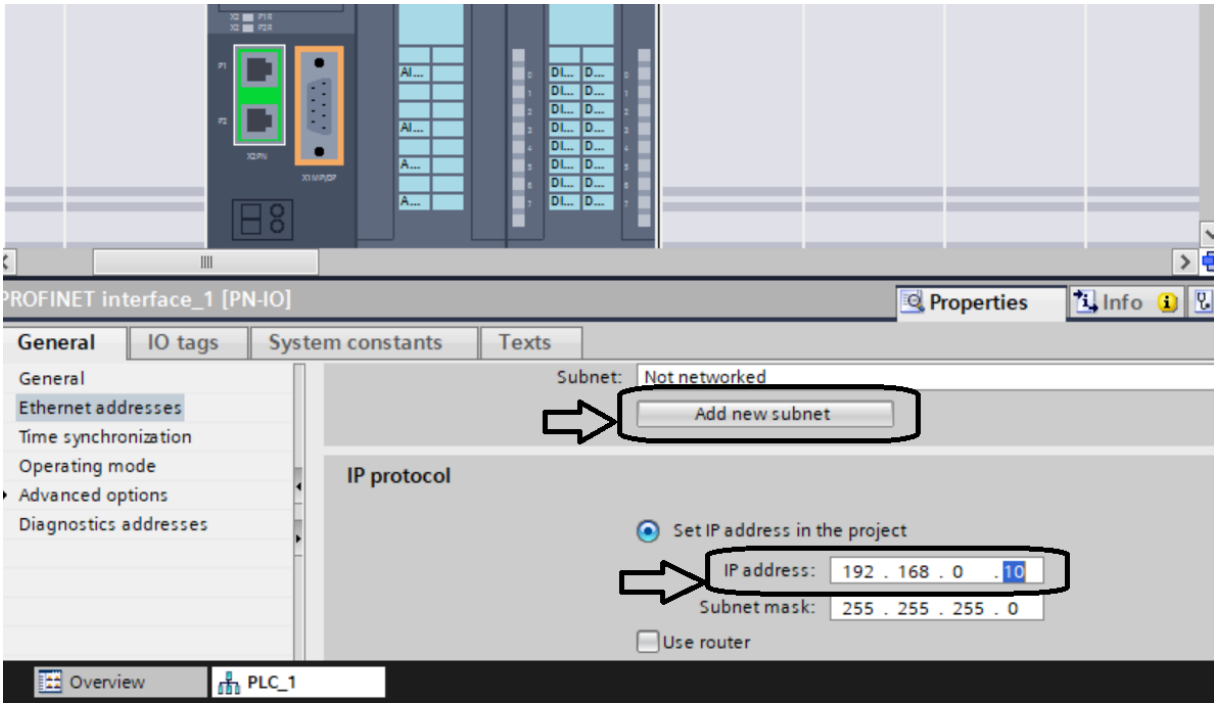


Figure II-8: 1st Step to configure device

Since nowadays we program with variables rather than absolute addresses, we must specify the global variables of the API. With a double-click, open the API variables table and enter, as shown below, the names of the inputs and outputs.

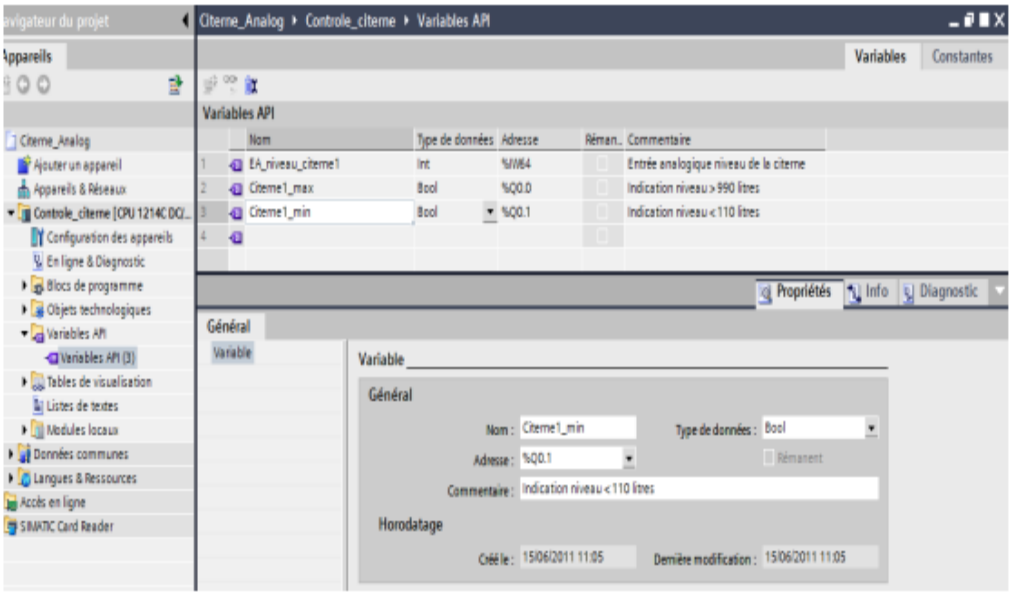


Figure II-9: 3rd Step to configure device

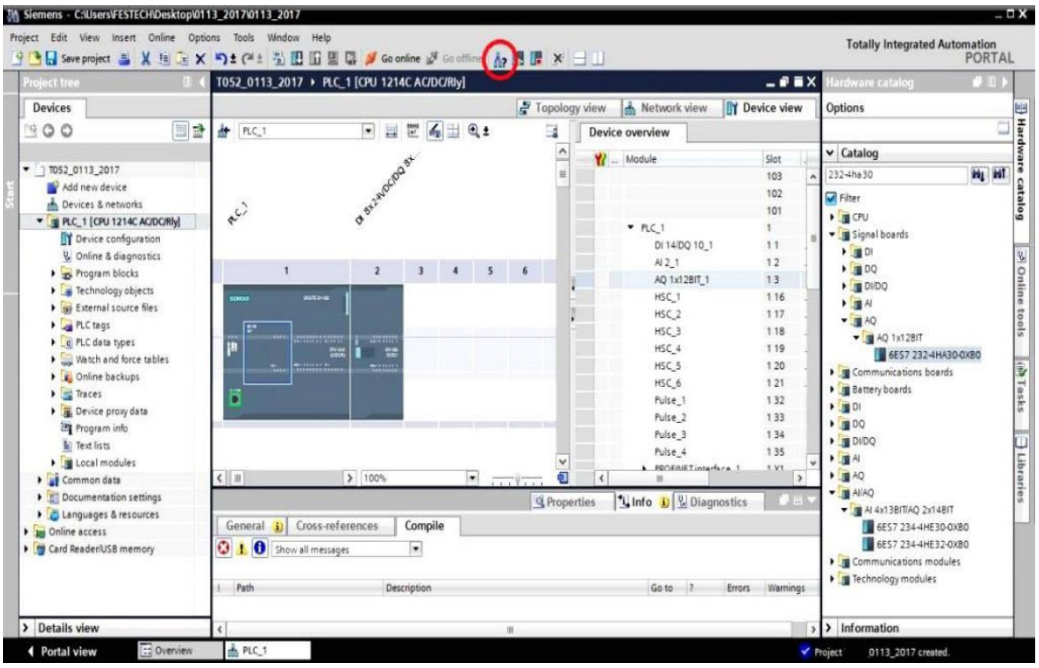


Figure II-10: 4th Step to configure device

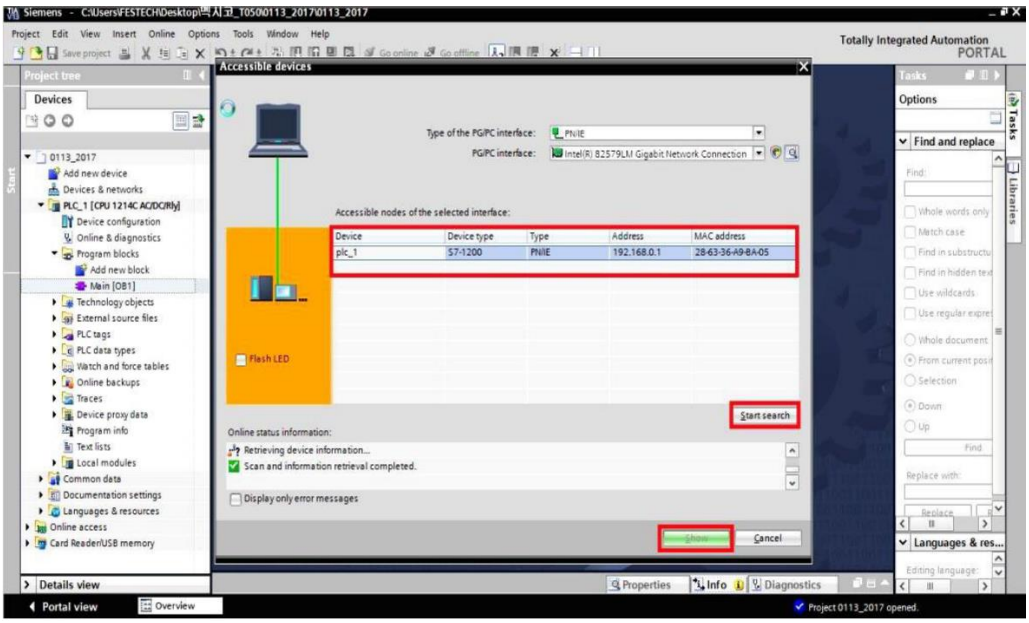


Figure II-11: 5th Step to configure device

- **Program the device:** Once the device is up-to-date, you can program it using the integrated programming environment in TIA Portal. This environment provides an easy-to-use interface for developing, revising, and testing PLC programs.
  - **Programming blocks:** In SIEMENS API programming, the program structure is organized into 4 different types of blocks:
    1. Organization blocks.
    2. Function.
    3. Functional block.
    4. Data block.
- **Test and simulate:** The TIA Portal provides a simulation tool that allows you to test your software before deploying it to a real device. This function is very helpful for testing complicated systems and reasoning.
- **Download and Deploy:** After testing and simulation, you may use the "Download" option in TIA Portal to download the application to the API S7-300. This capability lets you transmit the application from TIA Portal to the device through a network connection [21].

TIA Portal is an excellent software program for industrial automation that provides users with a comprehensive set of tools and features for configuring, programming, and testing automation systems. Its integration with other Siemens software and its user-friendly interface make it an essential tool for engineers and technicians in the automation [21].

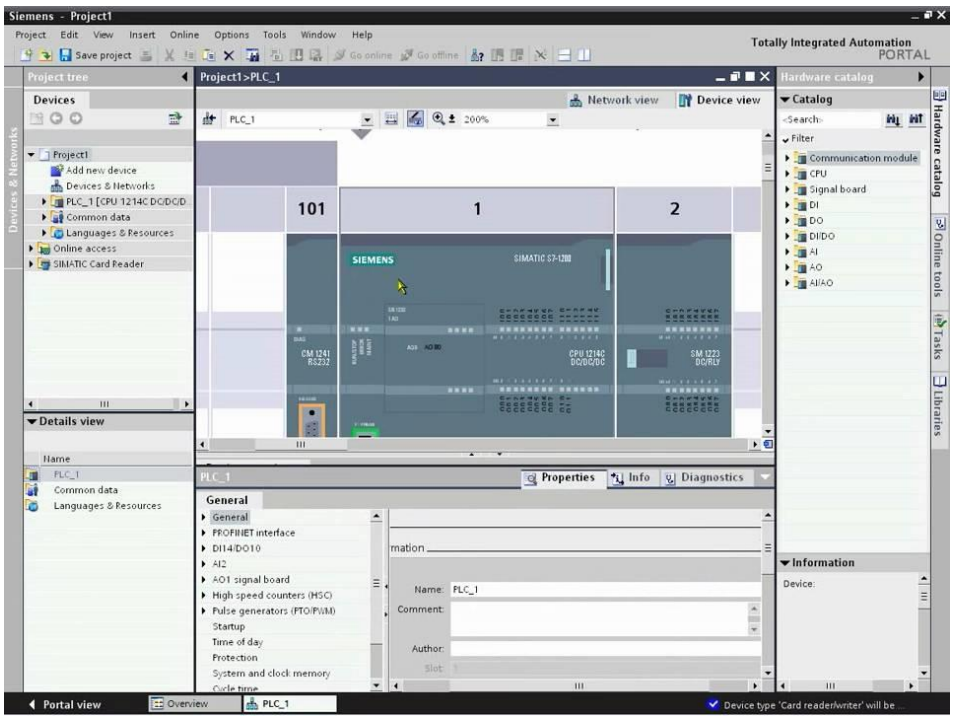


Figure II-12: Tia Portal Interface

II.3.8-The Advantages of Tia Portal Software

The TIA Portal software provides several benefits for industrial automation. Among these benefits are the following:

- **Efficiency:** Because of the effective use of technical resources and a uniform user interface across all components, TIA Portal provides for speedier development and commissioning of automation systems.
- **Flexibility:** Flexibility is provided by the software by allowing function customization and access to a range of programming languages, including ladder logic, structured text, and function block diagram.
- **Remote access:** TIA Portal enables for remote access to the automation system, allowing for remote monitoring and troubleshooting.
- **Engineering Simplified:** TIA Portal delivers a complete and integrated engineering environment that simplifies automation activities by allowing users to easily configure, program, and commission automation systems. The program provides a variety of libraries, predefined templates, and wizards that simplify the setup and maintenance of automation systems.
- **Scalability:** The TIA Portal software is designed to handle automation systems of varying sizes and complexity, making it suitable for usage in a variety of sectors such

as automotive, food and beverage, and pharmaceuticals. It is adaptable and may be scaled up or down to meet the requirements of the application.

- **Integration:** TIA Portal is designed to function in tandem with other Siemens automation solutions, making it simple to integrate with other hardware and applications. This connection allows customers to build a full solution that fulfills the specific requirements of their application.
- **User-Friendly Interface:** TIA Portal offers an easy-to-navigate and use interface that is suitable for both expert and novice users. The program has a drag-and-drop interface, real-time diagnostics, and graphical representations of automation systems to help users see and diagnose problems.
- **Saving time and effort:** The TIA Portal minimizes the time and effort necessary to configure, program, and commission automation systems. The software's user-friendly interface, pre-configured templates, and wizards accelerate the engineering process, lowering time to market and improving productivity.
- **Safety:** The TIA Portal provides several safety measures that aid in the safe running of automation systems. The program complies with international safety standards and allows users to rapidly and simply build and execute safety functionalities.
- **Maintenance:** The TIA Portal has comprehensive diagnostic capabilities that allow users to easily discover and rectify automation system faults. The program gives real-time feedback and thorough information on system performance, assisting in the reduction of downtime and maintenance costs [20].

II.4-LabVIEW

II.4.1-Software Description

LabVIEW is a graphical programming language-based development environment. Since graphical representations are a more natural design notation for engineers and scientists than textual coding, this method of application development significantly reduces the learning curve. Tools and functionality are available through interactive palettes, dialog boxes, menus, and hundreds of VIs (virtual instruments). These VIs can be dragged and dropped onto a block diagram to specify the behavior of the application being built.

This point-and-click method drastically reduces the time it takes to go from basic configuration to the final result. The LabVIEW graphical language provides scientists and engineers with a simple approach to creating measurement and control applications. The language is not only easy to learn and use but also provides the performance required for complex applications [22].

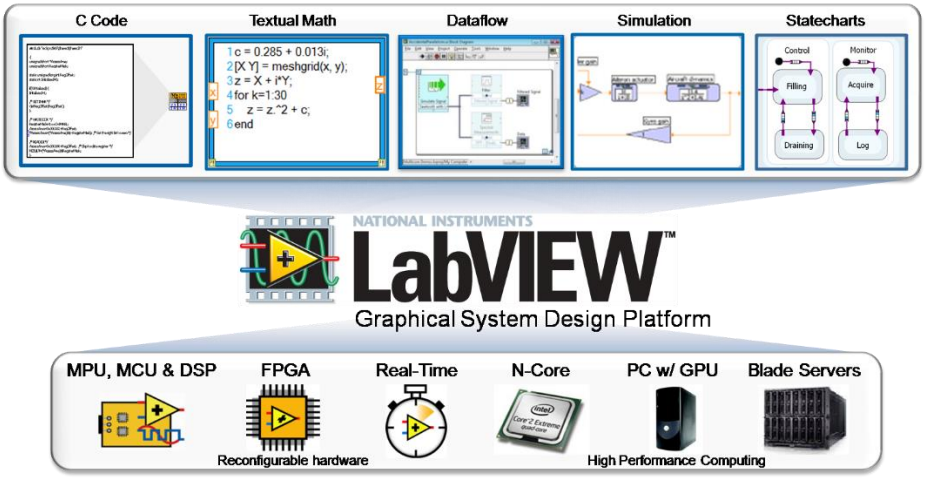


Figure II-13: LabVIEW

II.4.2-Understanding Virtual Instruments (VI)

LabVIEW programs are called Virtual Instruments (VIs). This is because their appearance and mode of operation are similar to that of physical instruments such as oscilloscopes or multi-meters, for example. Each VI uses functions that manipulate user-facing inputs or other sources and displays the results of processing or record them in result files or on other computers. A VI contains the following three elements:

- **Front panel:** Serves as the user interface.
- **Block diagram:** Contains the source code in a graphical form that defines the VI's functionality.
- **Icon and connector panel:** Identifies the VI so that you can use the VI in another VI. A VI inside another VI is called a sub VI. A sub VI corresponds to a subroutine (or function) in a text-based programming language such as C.

II.4.3-Origins of LabVIEW

LabVIEW, short for Laboratory Virtual Instrumentation Engineering Workbench, is a G-language development environment. Note that it is written as LabVIEW, not LabVIEW or LabVIEW, etc. It was first created in 1986, initially for Apple Macintosh, which was one of the only computers at the time offering a native graphical user interface. The history of LabVIEW explains its specific vocabulary and certain actions. Originally, LabVIEW ran on black and white screens, then on screens with 16 colors, 256, etc. LabVIEW is a proprietary development environment (as opposed to an open environment, allowing multiple people to develop compatible compilers, such as C/C++, Ada, Java, FORTRAN, etc.) developed and sold by the company National Instruments (NI). NI's primary business is to manufacture acquisition hardware (notably on the GPIB protocol in the early 1980s) quickly for the microcomputer

market (IBM PC, Apple Macintosh). Thus, the first version of LabVIEW aimed to provide a development environment whose role is to allow the user to simply create virtual instruments (VI) using NI acquisition hardware to reproduce on a microcomputer the behavior of a customized and customizable instrument at will [22].

II.4.4-The concept of a virtual instrument

The basic idea is to use an acquisition card or acquisition device, whose role is to acquire an electrical signal from the outside, usually a sensor (thermocouple, flow meter, voltmeter, etc.) or a set of sensors, perform processing, recording in a file or database, restitution on the screen via a graphical interface, and possibly perform a set of actions on the outside world using actuators (solenoid valve, motor, etc.). The possible variations are infinite depending on the needs and their evolutions. The advantage of the virtual instrument over the real instrument is undeniable since it is up to the virtual instrument programmer (therefore... you...) to evolve the virtual instrument according.

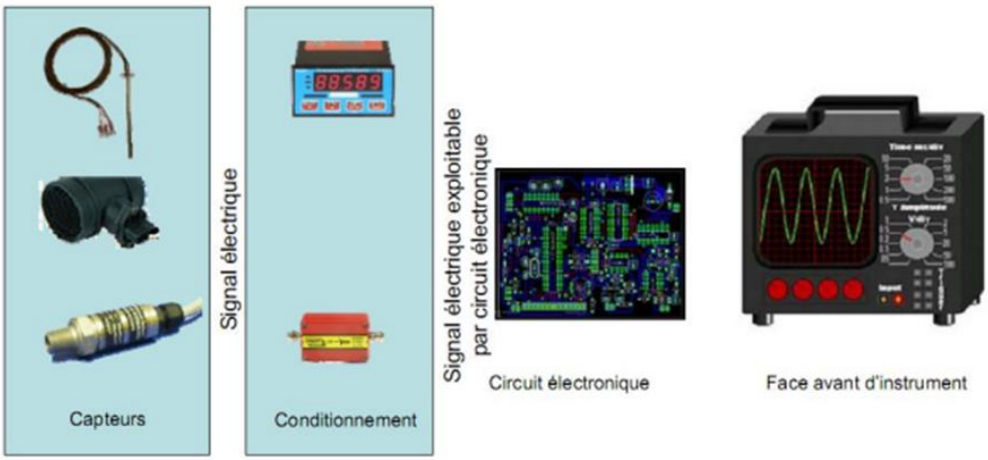


Figure II-14: Virtual Instrument

II.4.5-Getting Started

When launching LabVIEW, the startup screen allows you to start all the operations offered by this development environment to create a new program, select "Empty VI".



Figure II-15: Startup Screen of LabVIEW

LabVIEW uses the name VI (Virtual Instrument), pronounced in English as "vee-eye", for programs and sub-programs.

Programs or sub-programs will have the ".vi" extension and will therefore be named VI.

We will see that there is little difference between a program and a sub-program, so we may sometimes specifically use the term "sub-VI" for sub-program, but this will not always be the case. On the startup screen, the left part allows you to create or open a VI or project

II.4.6-LabVIEW Environment

LabVIEW is centered on the concept of a virtual instrument (VI). It is divided into two parts:

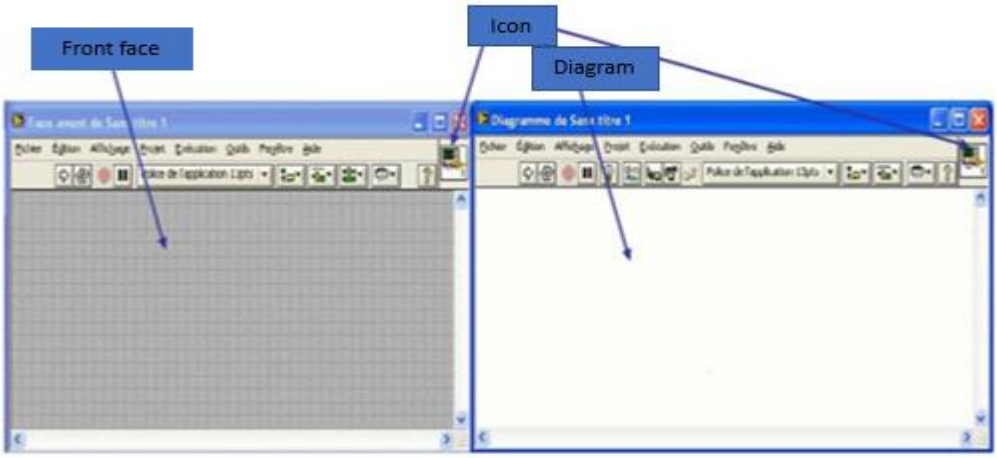


Figure II-16: LabVIEW Parts

II.4.6.1-Front Panel

The front panel is the user interface and consists of:

- Controls.
- Indicators.
- Controls: Simulate the inputs of virtual instruments and provide data to the block diagram.
- Indicators: Simulate the response of instruments and display data acquired or generated by VI's.

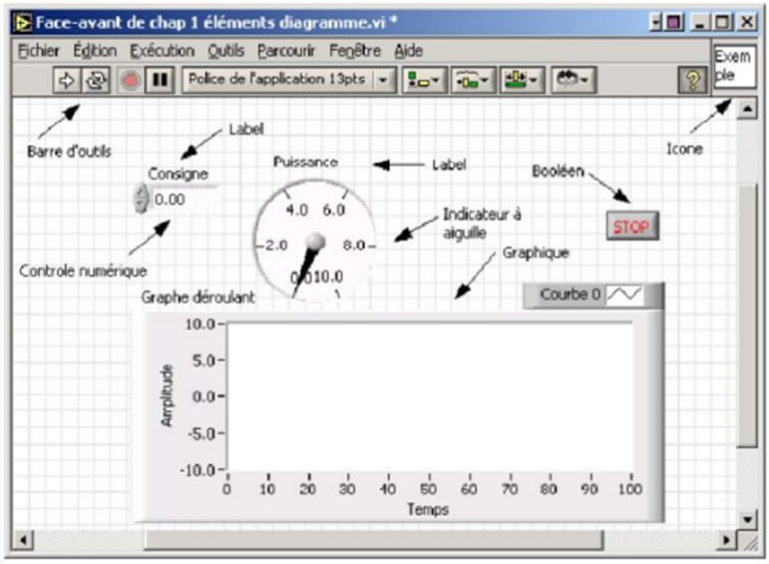


Figure II-17: Front Panel

II.4.6.2-Block Diagram

The block diagram is responsible for connecting inputs and outputs as well as the various functional elements of the program. It consists of:

- **Nodes:** Objects on the block diagram that have inputs and/or outputs and perform specific tasks when VI runs. They are equivalent to functions in text-based languages.
- **Terminals:** Objects on the front panel that appear as terminals on the block diagram. Terminals are communication ports between the front panel and the block diagram that reflects the data type of the control or indicator. They are equivalent to parameters and constants in text-based languages.
- **Wires:** Terminals and nodes are connected by wires to transfer data within the diagram. Each wire comes from a single source but can be distributed to multiple VIs or functions intended to process them. Wires are analogous to variables in text-based languages.

- **Structures:** More commonly known as LOOPS. Like many languages, a program can only be efficient with iteration mechanisms. In the case of LabVIEW, the most commonly used structures are loops.
- **Loops:** For and While.
- **Conditional structures.**
- **Sequence structures:** These are specific to LabVIEW and determine the order of execution of certain parts of the code.

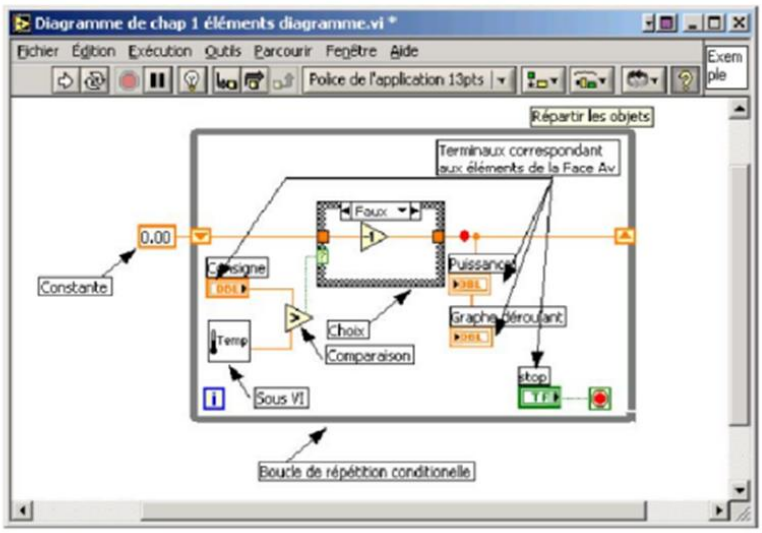


Figure II-18: Block Diagram

II.4.6.3-Tools

- **Front Panel and Block Diagram Toolbars:** The toolbars provide access to tools for running and presenting the program.



Figure II-19: Front Panel, Block Diagram Toolbars

- **Palettes:** LabVIEW palettes provide tools for creating and editing the front panel or block diagram. There are three palettes:
  - **Tool palette:** The tool palette exists on the block diagram and front panel. It allows for modifying values, and colors, as well as wiring inputs and outputs of icons together, setting breakpoints, probes, etc.

- **Control palette:** The Control palette is only accessible from the front panel. It allows for the creation of all types of controls and indicators. Each icon in the palette corresponds to a category that opens a secondary window containing all the elements related to that category that can be used on the front panel, such as:
  - Numbers.
  - Booleans.
  - Strings.
  - Arrays.
  - Clusters.
  - Graphs.
  - Special types.

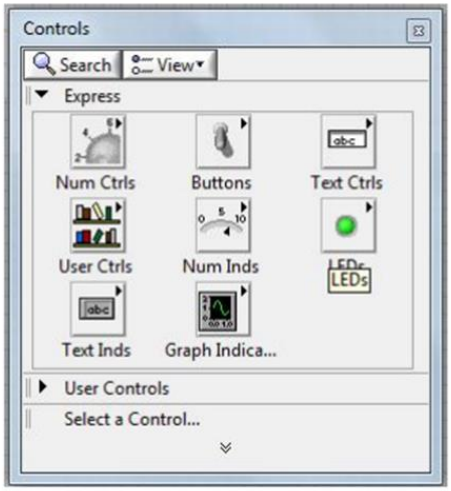


Figure II-20: Control Palette

- **Function palette:** The Function palette is only accessible on the diagram. It allows you to create all of LabVIEW's functions. Each icon on the palette corresponds to a section that provides access to a secondary window containing all of the elements related to that section that can be used on the diagram, for example:
  - Structures.
  - Numeric operations.
  - Logical operators.
  - Comparisons.
  - Time functions.
  - Input/Output.
  - Acquisition.
  - Signal processing.

II.4.6.4-The Context Help Window

Provides information about the object located under the mouse cursor, which can be information about the nature of a connection, the type of data to be provided to an indicator, and the operation of a sub-VI or a function.

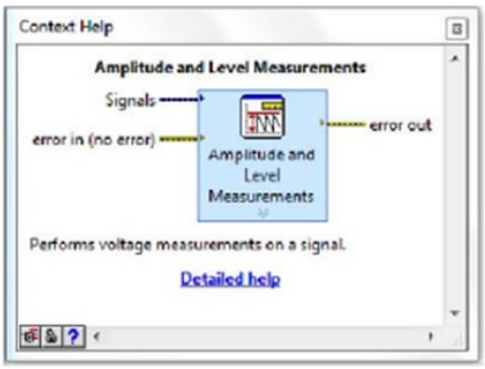


Figure II-21: Help Window

II.4.6.5-Data types used in LabVIEW

Like the vast majority of programming languages, LabVIEW has a wide range of variables (signed integers, unsigned integers, floating-point numbers, Booleans, strings, etc.). To identify them more easily in the diagram, each family of variables has a specific color.

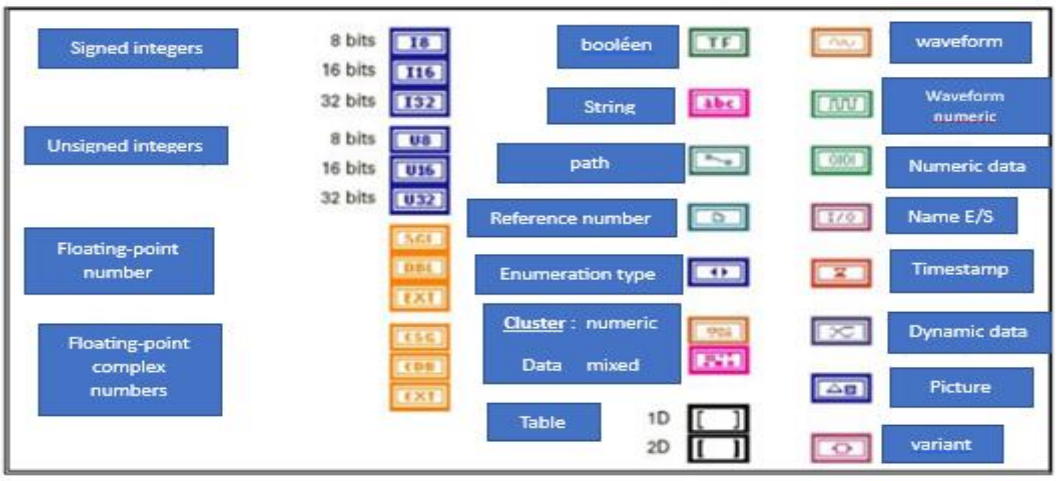


Figure II-22: Data Types

II.4.7-The Advantages of LabVIEW

- **Graphical Programming:** LabVIEW is a graphical programming language that allows users to program using a graphical user interface instead of traditional text-based programming. This feature makes it easier for non-programmers to use and understand.

- **Modular Programming:** LabVIEW allows users to create modular programs that can be easily reused in different projects, saving time and effort.
- **Data Acquisition:** LabVIEW has powerful tools for data acquisition, including support for multiple data types, real-time data processing, and the ability to interface with a variety of hardware devices.
- **Analysis and Visualization:** LabVIEW provides powerful tools for data analysis and visualization, including advanced graphing and charting capabilities, statistical analysis tools, and the ability to create custom reports.
- **Integration with other Software:** LabVIEW can easily integrate with other software tools, such as MATLAB and Microsoft Excel, allowing users to import and export data between programs.
- **Community Support:** LabVIEW has a large community of users and developers who provide support and share their expertise through online forums and resources.
- **Easy to Learn and Use:** LabVIEW has a graphical programming language that is intuitive and easy to learn, even for those with little or no programming experience.
- **Compatibility:** LabVIEW is compatible with a wide range of hardware devices, making it a versatile option for many different applications.
- **Real-Time Applications :** LabVIEW is well-suited for real-time applications, as it allows for data acquisition, signal processing, and control to be performed in real-time [23].

LabVIEW is a powerful tool for industrial automation that provides users with the ability to quickly develop complex programs, analyze and visualize data, and integrate with other software tools [23].

II.4.8-Integration of LabView with the API S7-300

LabVIEW can be integrated with the API S7-300 to monitor and control industrial processes. The integration can be achieved using the LabVIEW Real-Time Module and the LabVIEW Data logging and Supervisory Control (DSC) Module.

The LabVIEW Real-Time Module provides a development environment for creating real-time applications that can run on a real-time operating system. This module includes a set of VIs (Virtual Instruments) that allow users to communicate with the S7-300 PLC over Ethernet or serial communication protocols.

The LabVIEW DSC Module provides a set of VIs for data acquisition, logging, and supervisory control. It also includes a built-in OPC server that allows LabVIEW to communicate with third-party devices, such as the S7-300 PLC, using the OPC protocol.

By integrating LabVIEW with the API S7-300, users can create custom HMI (Human-Machine Interface) applications that allow operators to monitor and control industrial processes in real-time. They can also perform data logging and analysis, create custom reports, and implement advanced control algorithms to improve process performance and efficiency [24].

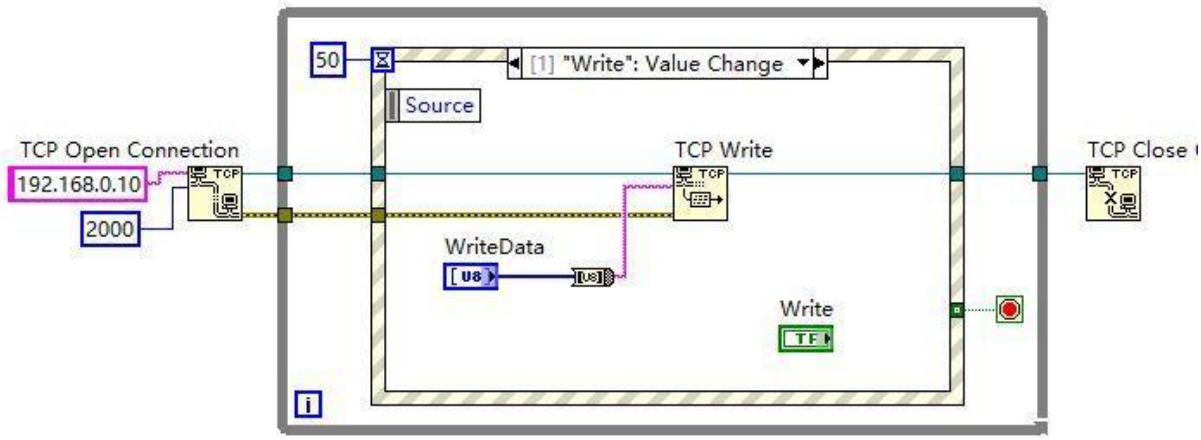


Figure II-23: LabVIEW with API S7-300 via Ethernet (TCP)

II.4.9-Examples of Industrial Automation Applications Using LabVIEW

- **Manufacturing:** LabVIEW can be used to monitor and control production lines in manufacturing plants. It can also be used for quality control and to perform data analysis to improve the efficiency of the production process.
- **Aerospace:** LabVIEW can be used in the aerospace industry to control the various systems in aircraft and to monitor their performance. It can also be used for testing and simulation of aircraft systems.
- **Automotive:** LabVIEW can be used to control the various systems in automobiles, such as engine control, emissions control, and safety systems. It can also be used for testing and validation of automotive components and systems.
- **Energy:** LabVIEW can be used in the energy industry for monitoring and control of power generation systems, such as wind turbines and solar panels. It can also be used for analysis of energy usage data to identify opportunities for energy savings.
- **Medical devices:** LabVIEW can be used to control and monitor medical devices, such as MRI machines, CT scanners, and patient monitors. It can also be used for data analysis to improve the performance of medical devices.

- **Robotics:** LabVIEW can be used to control industrial robots, such as those used in manufacturing and assembly lines. It can also be used for testing and simulation of robotic systems [24].

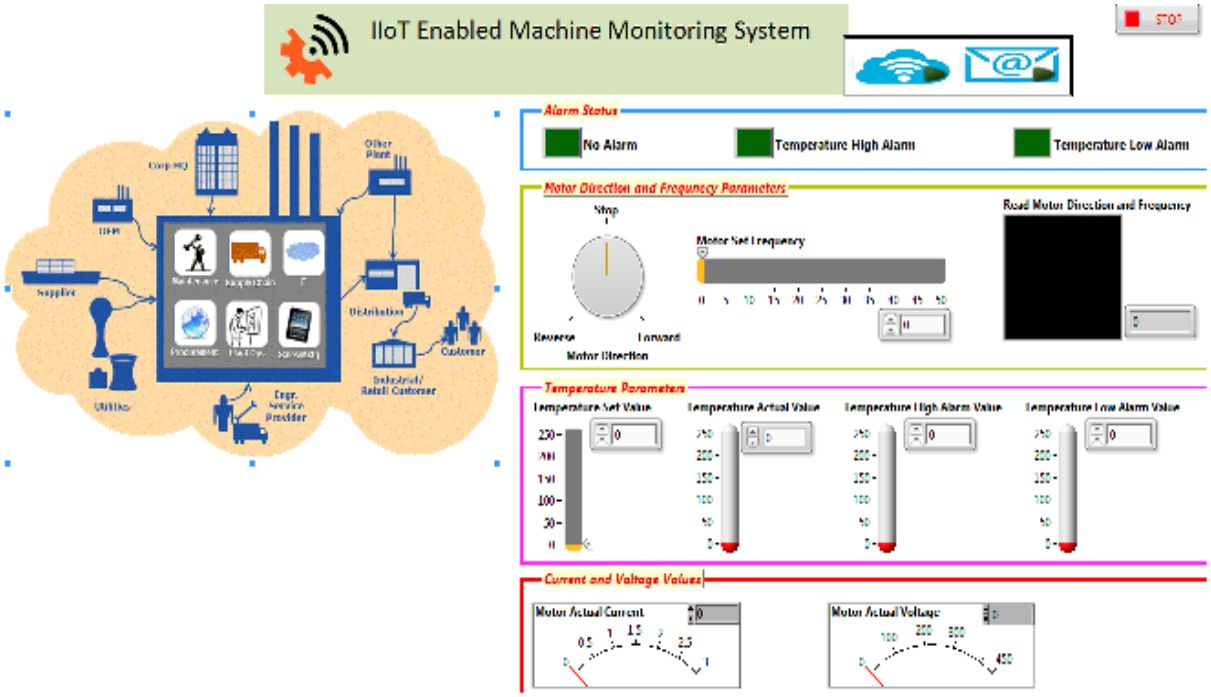


Figure II-24: Industrial IOT Solution using LabVIEW

II.4.10-Integration of LabVIEW with Tia Portal

- The integration of LabVIEW with the TIA Portal can be achieved through the use of the TIA Openness interface. TIA Openness is a set of software tools that allow external applications to access and modify TIA Portal projects and data.
- With the TIA Openness interface, LabVIEW can access and control the API S7-300 PLC in a TIA Portal project. This allows LabVIEW to perform tasks such as reading and writing data to the PLC, controlling the PLC's operation, and responding to alarms and events.
- To use the TIA Openness interface in LabVIEW, the LabVIEW OPC Toolkit must be installed. This toolkit provides the necessary tools and functions to communicate with OPC servers, including the NI OPC Server.
- Once the OPC toolkit is installed, LabVIEW can communicate with the NI OPC Server to access the data from the API S7-300 PLC in the TIA Portal project. This data can then be processed and displayed in LabVIEW's graphical user interface.
- The integration of LabVIEW with the TIA Portal provides a powerful and flexible solution for industrial automation applications. LabVIEW's graphical programming environment allows for rapid development and customization of control and

monitoring applications, while TIA Portal provides a comprehensive suite of tools for PLC programming and configuration.

II.5-NI OPC Server

II.5.1-Description

NI OPC Server is a software program that allows data exchange between different industrial devices and software applications. OPC (OLE for Process Control) is a standard protocol that enables communication between industrial automation devices and software systems, regardless of the manufacturer. NI OPC Server is an implementation of this protocol and provides a reliable and secure way to transfer data between OPC clients and servers.

NI OPC Server can be used with the API S7-300 by configuring the server to communicate with the S7-300 PLC using the OPC protocol. This allows for seamless data exchange between the PLC and other OPC-compliant devices and software applications.

NI OPC Server is an essential tool for integrating different devices and software applications in industrial automation and streamlining data exchange between them [25].

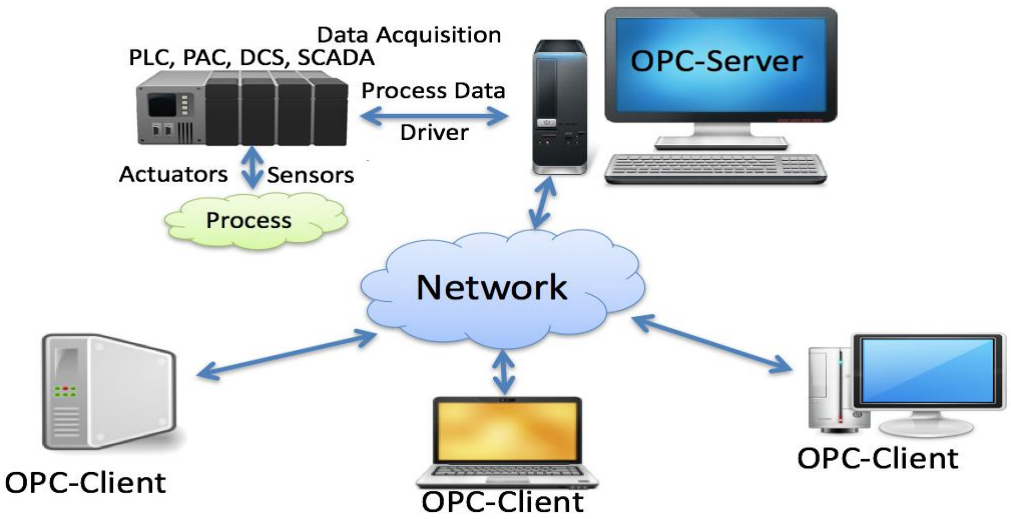


Figure II-25: NI OPC Server

II.5.2-The Advantages of NI OPC Server

NI OPC Server is a software program that enables the communication between various devices and software applications in an industrial automation system. Some of the advantages of using the NI OPC Server include:

- **Interoperability:** The NI OPC Server allows devices and software applications that utilize multiple protocols to communicate and exchange data. This interoperability allows various devices and software packages to operate together effortlessly.

- **Real-time data access:** The NI OPC Server allows for real-time data access to various devices and software applications, allowing operators to monitor and operate the system in real time.
- **Scalability:** As the automation system expands, the NI OPC Server is extremely scalable and can be quickly upgraded to handle more devices and software applications.
- **Easy configuration:** The NI OPC Server provides an intuitive user interface that makes it simple to configure and set up connections between devices and software applications.
- **Security:** To secure the integrity and confidentiality of data exchanged between devices and software applications, NI OPC Server includes sophisticated security features such as encryption, authentication, and access control.
- **Fault-tolerance:** The NI OPC Server has fault-tolerance techniques that enable ongoing operation even if a device or program fails.

NI OPC Server is a strong industrial automation technology that enables continuous connection and data sharing between various devices and software applications [25].

II.5.3-Integration of NI OPC Server with API S7-300

NI OPC Server can be integrated with the API S7-300 through a process known as OPC UA (Unified Architecture). OPC UA is a communication protocol that allows for interoperability between different industrial devices and systems.

To integrate the API S7-300 with NI OPC Server, the following steps can be taken:

- Install and configure the NI OPC Server software on the computer that will be acting as the OPC server.
- Connect the API S7-300 to the same network as the OPC server computer.
- Configure the API S7-300 to enable OPC UA communication. This can be done through the TIA Portal software by configuring the OPC UA server settings.
- Configure the NI OPC Server to connect to the API S7-300. This involves specifying the IP address and other relevant parameters in the server configuration.
- Once the connection is established, data can be exchanged between the API S7-300 and other devices or systems through the NI OPC Server using OPC UA.

The integration of the NI OPC Server with the API S7-300 allows for seamless communication and data exchange between different industrial devices and systems. This can greatly enhance the efficiency and effectiveness of industrial automation processes [26].

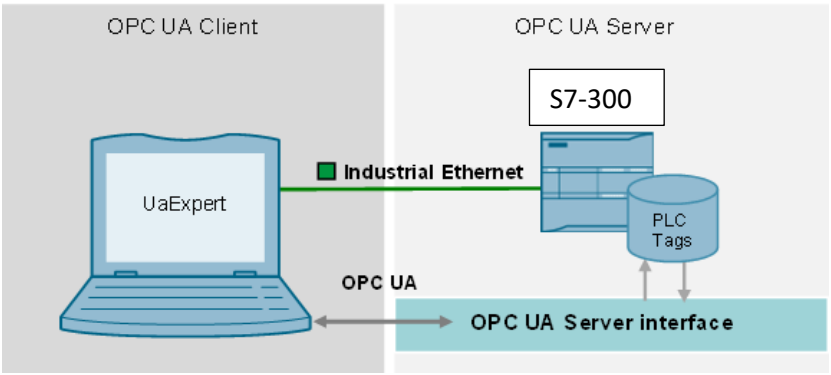


Figure II-26: NI OPC Server with API S7-300

II.5.4-Integration of NI OPC Server with LabVIEW

LabVIEW can interface with many industrial automation devices, including the API S7-300, thanks to the inclusion of NI OPC Server. This connection allows LabVIEW to read and write data from and to the S7-300, as well as manage and monitor the process. To use the NI OPC Server with LabVIEW, first, install the NI OPC Server on the computer. The NI OPC Server must then be set up to connect with the S7-300. The LabVIEW OPC client must then be installed, and the OPC server must be attached to the client. Once setup is completed, LabVIEW will be able to access data from the S7-300 via the OPC server.

The integration of the NI OPC Server with LabVIEW provides various benefits, including a simple and easy-to-use interface for LabVIEW users to retrieve data from the S7-300, real-time data collecting and monitoring, and data analysis and visualization. Furthermore, this connectivity provides greater versatility and scalability by allowing users to integrate numerous automation devices with LabVIEW, allowing them to operate and monitor several devices at the same time [26].

In the Next Example, LabVIEW Application 1 and LabVIEW Application 2 are on the same computer. Normally they are located on 2 different computers in a Network.

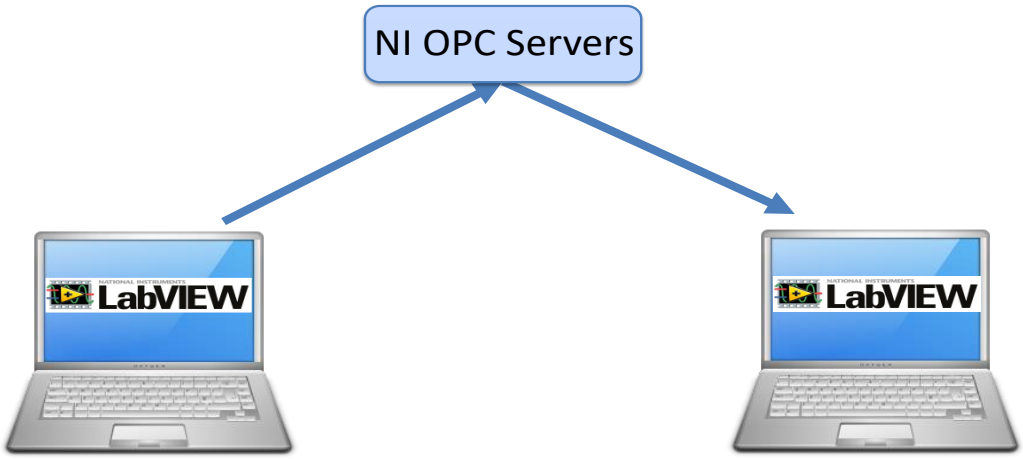


Figure II-27: 2 Applications LabVIEW with NI OPC Server

II.5.4.1-NI OPC Server

A Demo version should be included with LabVIEW DSC Module or LabVIEW Real-Time Module.

Use

- Search in Windows.
- Search for «OPC Servers».
- Select «OPC Servers Configuration» [27].



Figure II-28: Open NI OPC Server

- **Configuration:**

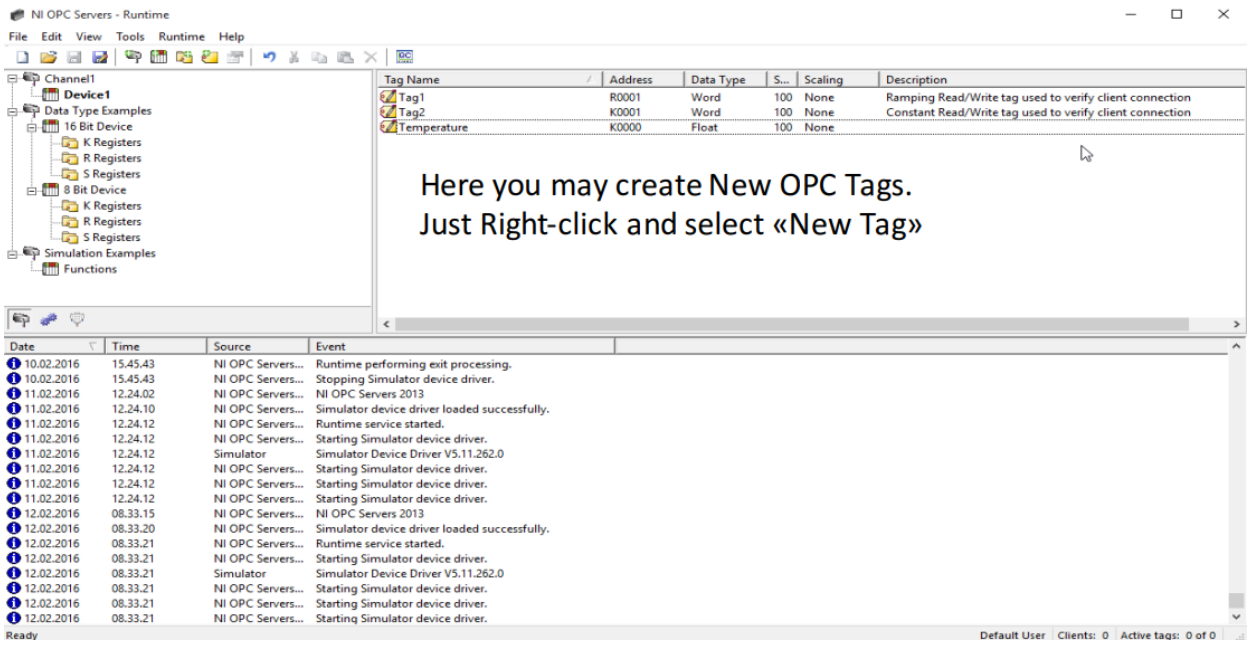


Figure II-29: NI OPC Server Configuration [27]

- **Configuration Create New Tag:**

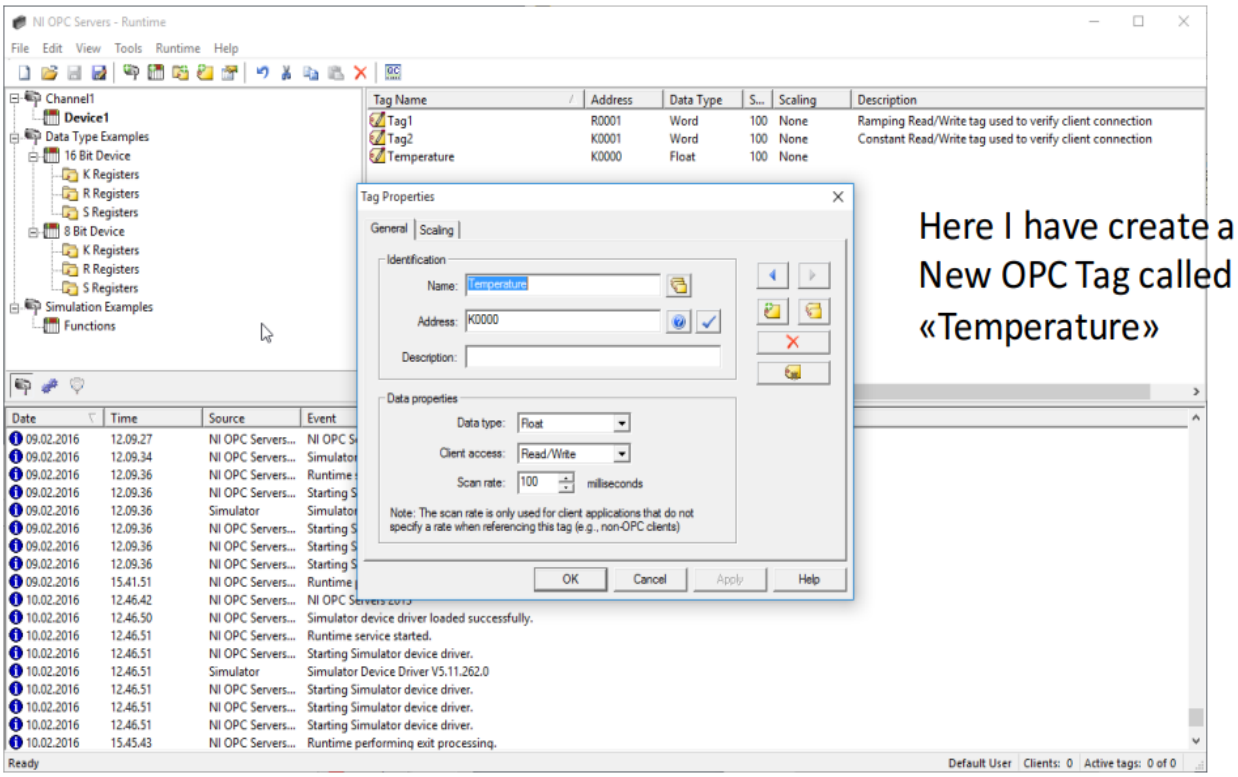


Figure II-30: NI OPC Server Create New Tag [27]

II.5.4.2-NI OPC in LabVIEW

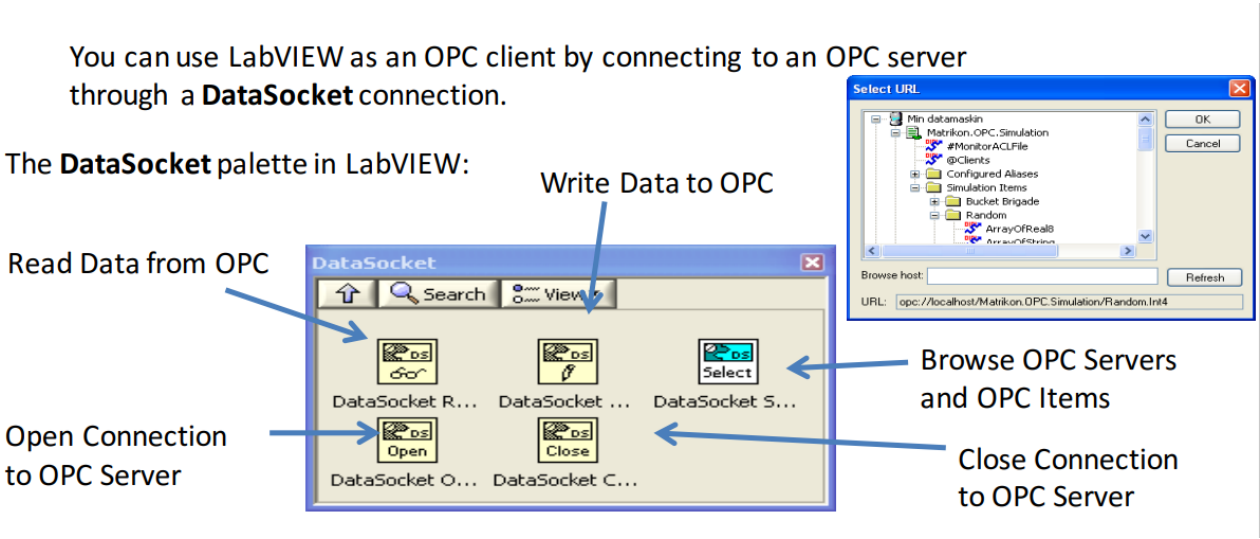


Figure II-31: NI OPC in LabVIEW [27]

- **NI OPC Write in LabVIEW:** in this example LabVIEW Application 1 and LabVIEW Application 2 are on the same computer. Normally they are located on 2 different computers in a network [27].

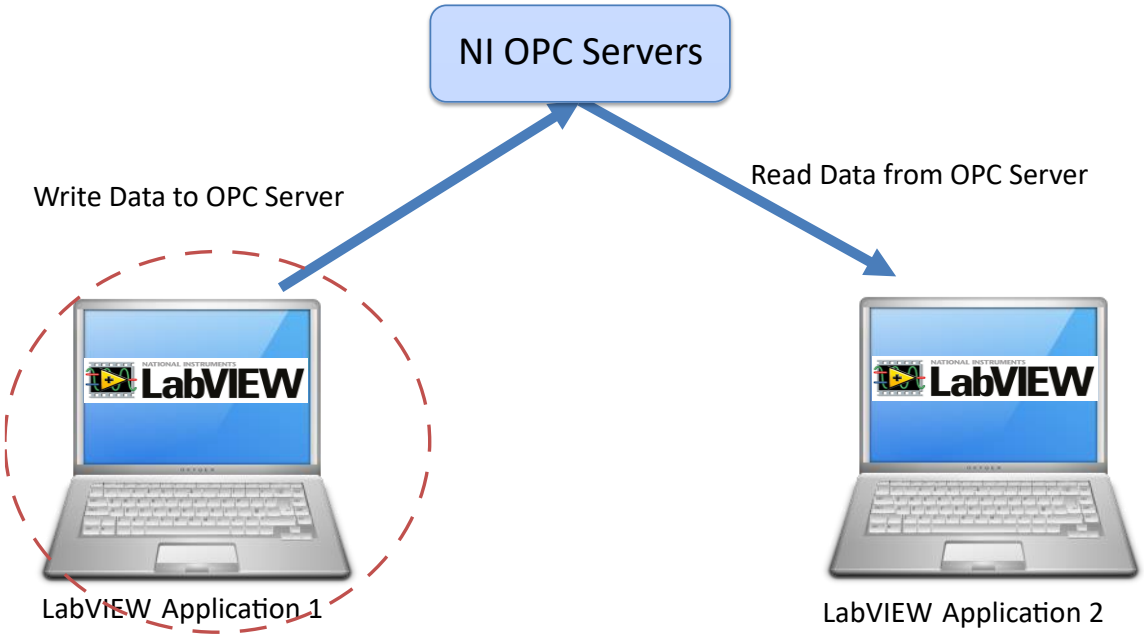


Figure II-32: NI OPC write in LabVIEW [27]

- **Write to OPC server using LabVIEW:**

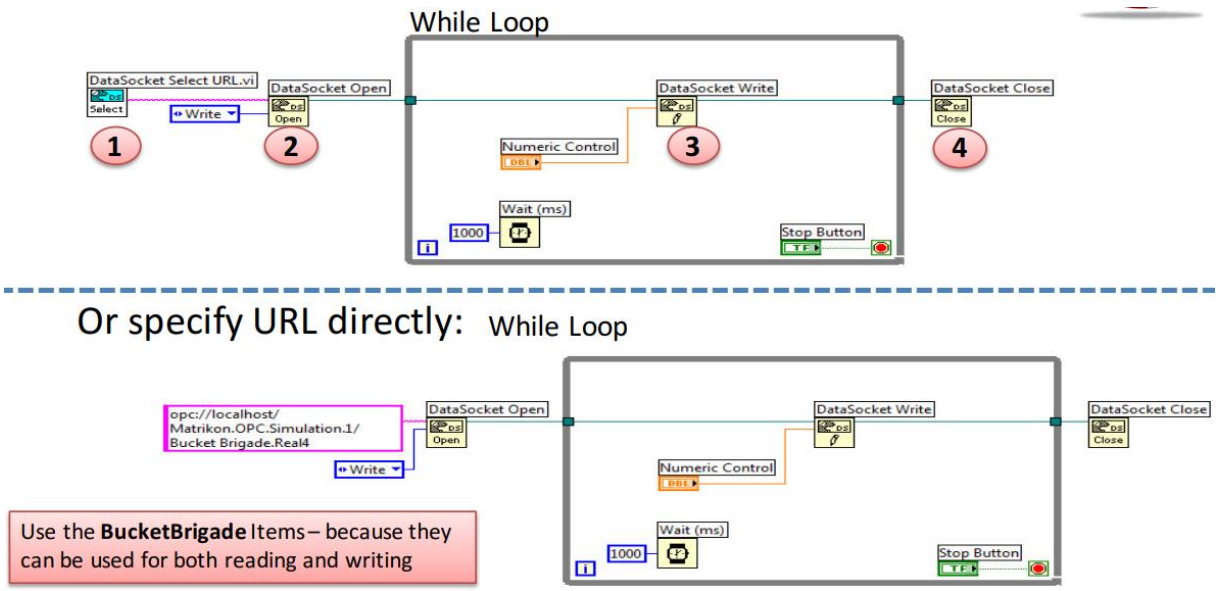


Figure II-33: Writing in NI OPC server [27]

- **Connect to NI OPC Server:**

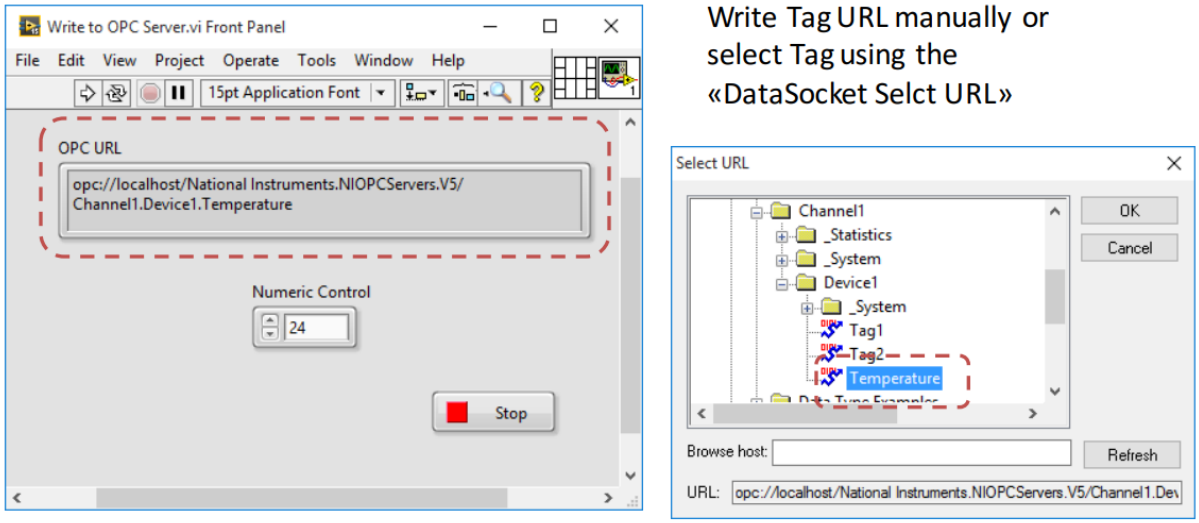


Figure II-34: Connect to NI OPC server [27]

- **NI OPC Server read in LabVIEW:** in this example, LabVIEW Application 1 and LabVIEW Application 2 are on the same computer. Normally they are located on 2 different computers in a network [27].

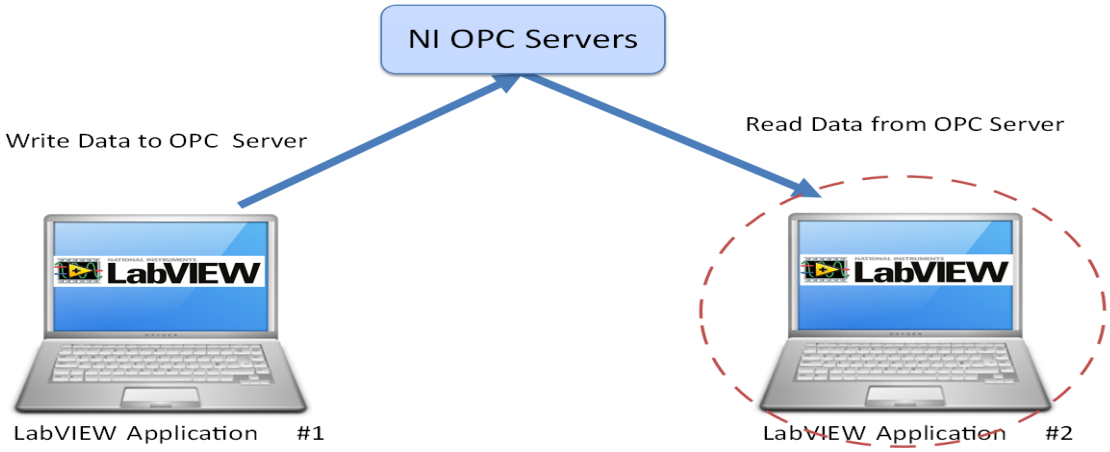


Figure II-35: NI OPC read in LabVIEW

- **Read From NI OPC Server with LabVIEW:**

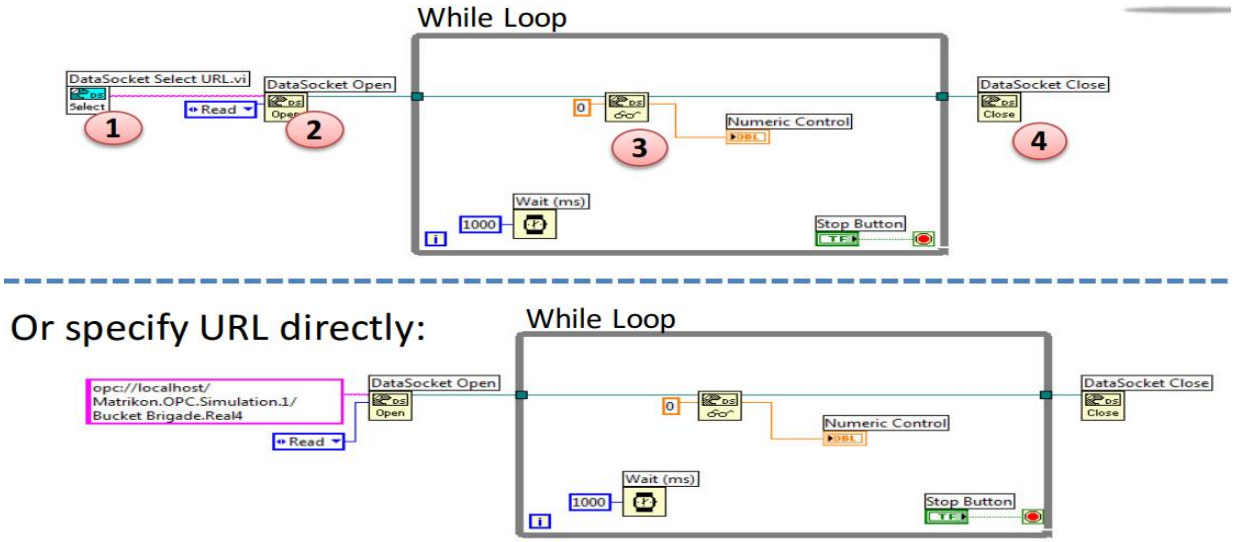


Figure II-36: Read from NI OPC [27]

- **Connect to NI OPC Server:**

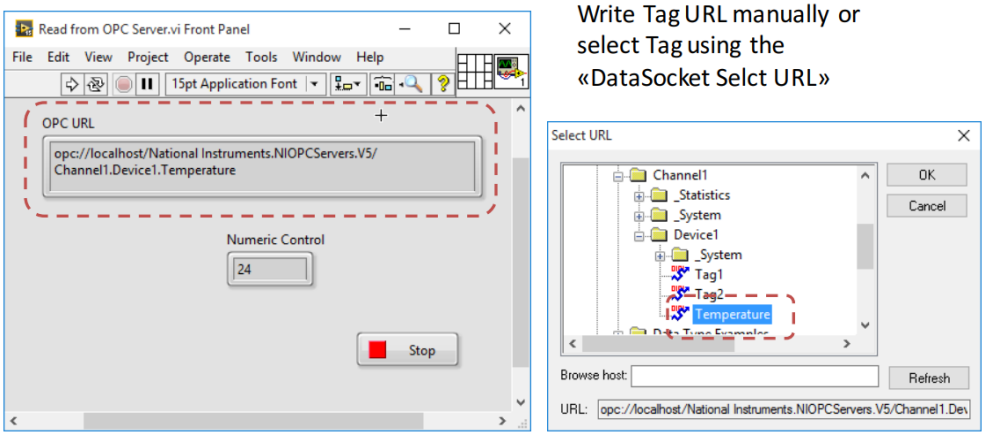


Figure II-37: Read from NI OPC

II.5.4.3-NI OPC and LabVIEW Simultaneously

In this example, LabVIEW Application 1 and LabVIEW Application 2 are on the same computer. Normally they are located on 2 different computers in a network [27].

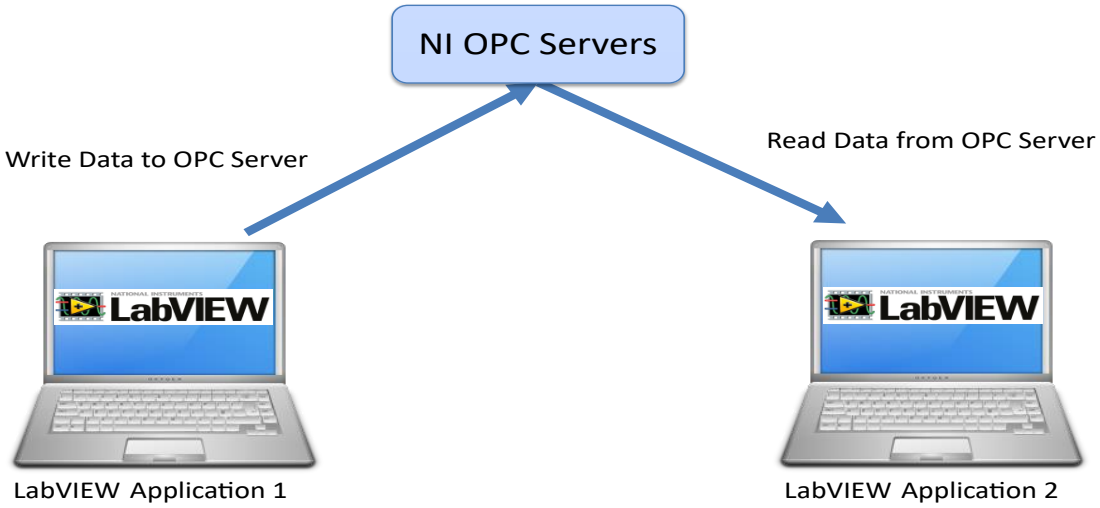


Figure II-38: Reading and writing from NI OPC and LabVIEW in the same time

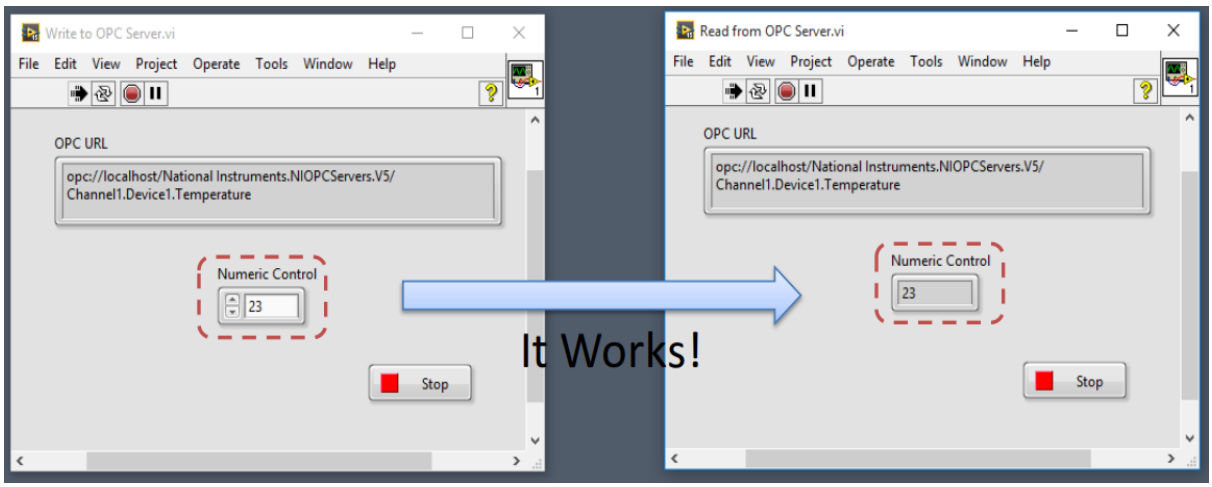


Figure II-39: Running write client and read client simultaneously [27]

II.5.5-Integration of NI OPC Server with Tia Portal

NI OPC Server can be integrated with TIA Portal using the following steps:

- Install the NI OPC Server on the computer that will act as the OPC server.
- Open the TIA Portal and create a new project.
- Add a new device to the project by selecting the "Devices" tab in the project view and clicking "Add new device".

- Select the "SIMATIC S7-300" option and configure the device settings, such as the IP address and communication parameters.
- Add a new data block by right-clicking on the device in the project view and selecting "Add new data block".
- Configure the data blocks settings and variables, such as data types and addresses.
- Open the NI OPC Server configuration utility and add a new OPC server connection by clicking "Add new connection" and entering the connection parameters, such as the server's name and IP address.
- Add a new OPC tag by selecting the data block and variables from the TIA Portal project view and dragging them into the OPC server configuration utility.
- Save the configuration changes and start the OPC server.
- Finally, open the LabVIEW application and use the NI OPC Server API to connect to the OPC server and access the data from the TIA Portal project.

With this integration, LabVIEW can communicate with the TIA Portal project via the NI OPC Server, allowing for real-time data acquisition and control of the industrial process [26].

II.5.6-Net to PLC Sim configuration:

Net To PLC Sim is a software tool connecting simulation software to external programs/devices. The main advantages of this software are:

- Acts as a bridge for data exchange between simulation software and external applications.
- Establishes virtual connections with third-party software/devices.
- Supports communication protocols like OPC and TCP/IP.
- Allows interaction with virtual PLC for realistic scenarios without physical hardware.
- Enables testing and debugging of PLC programs without physical PLC or system disruption.

In order to configure this software, the main PC must allow the usage of port 102 for this software to work properly

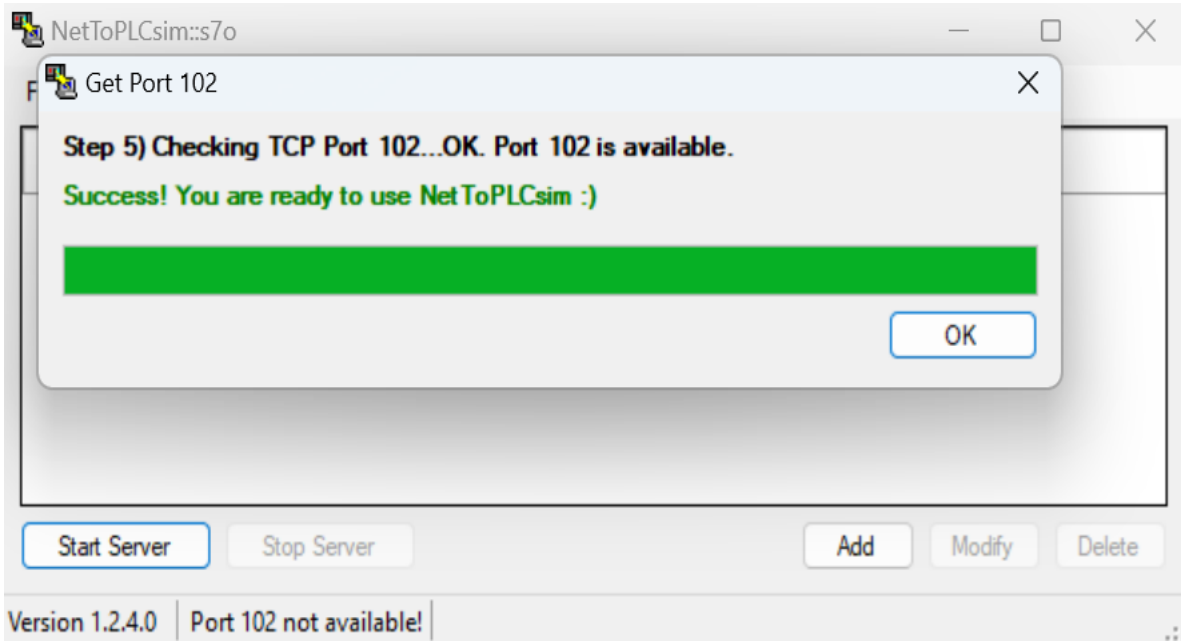


Figure II-40: Getting Port 102.

After getting port 102 in check, we add a station and select a name of choice afterward, we insert the network IPv4 main PC address and the PLC sim IP address (from the Tia portal).

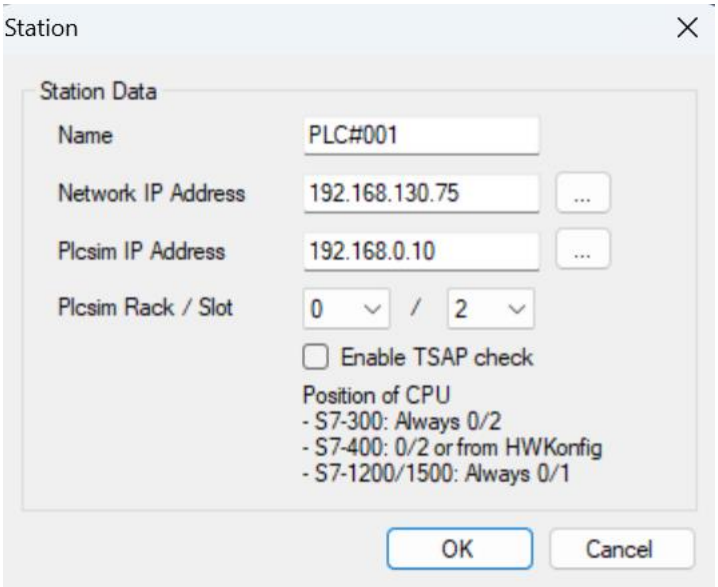


Figure II-41: inserting the network and Plc sim addresses

II.5.7-WIFI:

To share a Wi-Fi network using a phone hotspot, we employed a simple yet effective method. First, we accessed the settings of a phone and enabled the personal hotspot feature, transforming it into a mobile Wi-Fi hotspot. This allowed the phone to broadcast its own Wi-Fi network, enabling other devices to connect to it. Once the hotspot activates, we obtained the IP address associated with the phone's network. By navigating to the Wi-Fi settings on the device of choice, we selected the network corresponding to the phone's hotspot and entered the provided IP address. This allowed us to establish a successful connection and gain access to the internet through the phone's network. This convenient approach allowed us to share a Wi-Fi network seamlessly, providing connectivity to other devices without the need for a traditional router.

Afterward, we obtain the main PC`s IPv4 address for the next part which allows us to create a connection between two PCs (the main PC and the host PC), the figure below shows us obtaining the IPv4 address:

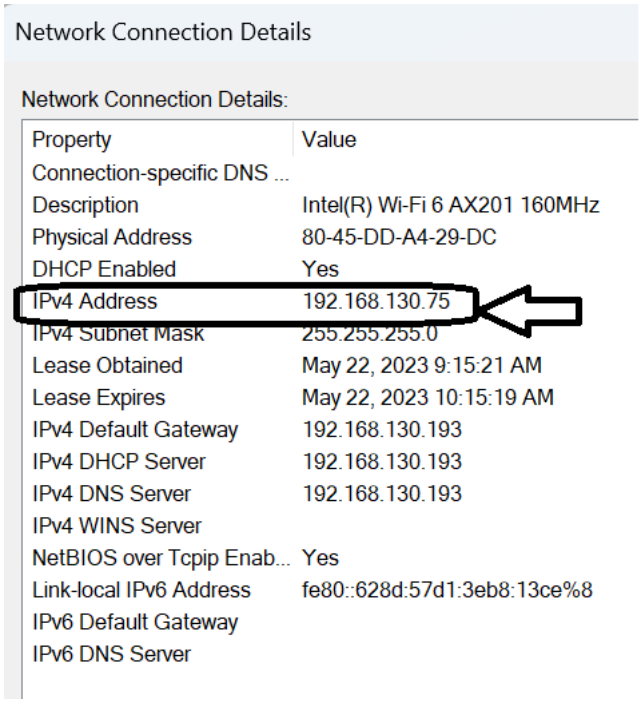


Figure II-42: obtaining the IPv4 Address

II.6-Conclusion

In conclusion, this chapter provided a comprehensive overview of the software programs used in our industrial automation system. We have described the main software programs used, including Tia Portal, LabVIEW, and NI OPC Server. We described also their features, advantages, and capabilities, as well as their integration with the API S7-300. Additionally, we presented examples of industrial automation applications that use LabVIEW. We discussed the integration of the NI OPC Server with LabVIEW and the TIA Portal. It is evident that software programs are essential in industrial automation as they provide efficient and effective ways to design, monitor, and control industrial processes. The software programs discussed in this chapter have shown great benefits in increasing productivity, reducing downtime, and improving quality control. In the next chapter, we will discuss the hardware components of industrial automation and their roles in industrial processes.

Chapter III

Simulation and Results Discussions

III.1-Introduction

In this chapter, we explore controlling an induction motor using LabVIEW from a main PC and TIA Portal from a host PC, seamlessly connected through the NI OPC server. LabVIEW serves as the main control interface on the main PC, allowing us to set precise control parameters and operational parameters for the motor. Tia Portal, running on the second PC, completes the system with advanced control features. The NI OPC Server acts as a bridge, facilitating efficient communication and data exchange between TIA PORTAL and LabVIEW, enabling synchronized control and monitoring of the engine system. Through simulation and discussion of the results, we aim to better understand motor performance and behavior, using the integrated control setup for industrial applications.

III.2-Programming

III.2.1-Specifications

The motor is turned off in the program's initial state, rendering all elements inactive. The program serves three primary purposes, outlined as follows:

- **Motor Control**: The motor is regulated through a variable speed drive, utilizing three buttons: forward, reverse, and ON/OFF.
- **Button Functions**: The forward, reverse, and ON/OFF buttons are responsible for commanding the motor's movement, and determining its direction.
- **Speed Adjustment**: The variable speed drive offers the capability to finely tune the motor's rotational velocity, enabling precise control over its speed.

The proposed configuration ensures the control of the motor in a precise and efficient way (movement and variation of the speed of the motor).

III.2.2-Tia Portal

In this scenario, TIA Portal ladder programming is leveraged to control a motor via a variable speed drive and a trio of buttons. The forward, reverse, and ON/OFF buttons assume the role of controlling the movement of the motor. Leveraging intuitive ladder logic, the program orchestrates seamless switching between forward and reverse directions, while the variable speed drive adjusts the rotational speed of the motor. Incorporating these elements into the TIA Portal framework allows us to effortlessly control and manipulate motor behavior, enabling precise and efficient motor control operations.

III.2.2.1-Program Creation

In any program, it is necessary to define the list of variables that will be used during programming. For this purpose, a variable table is created to store system variables.

Table III-1: variable table

	Name	Tag table	Data type	Address
1	INDICATOR_F	Table de variables standard	Bool	%Q1.0
2	REVERSE	Table de variables standard	Bool	%M1.1
3	INDICATOR_R	Table de variables standard	Bool	%Q1.1
4	SPEED	Table de variables standard	Real	%MD14
5	LIMIT_SP	Table de variables standard	Real	%MD20
6	ON/OFF	Table de variables standard	Bool	%M1.2
7	Tag_5	Table de variables standard	Word	%MW2
8	VOLTAGE	Table de variables standard	Integer	%QW200
9	STOP	Table de variables standard	Bool	%Q1.2
10	FORWARD	Table de variables standard	Bool	%M1.0
11	INDICATOR_ON-OFF	Table de variables standard	Bool	%Q1.3

III.2.2.2-Program in Contact language

- The system control program is written in distributed contact language across six networks, with each network representing a sequence. The program consists of the following networks:
- **Network 1:** Forward Control

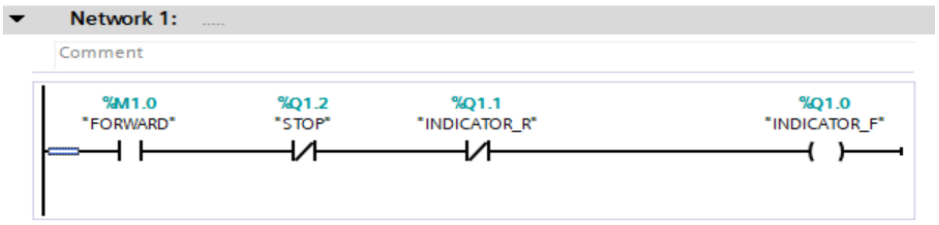


Figure III-1: Network 1

- **Network 2:** Reverse Control

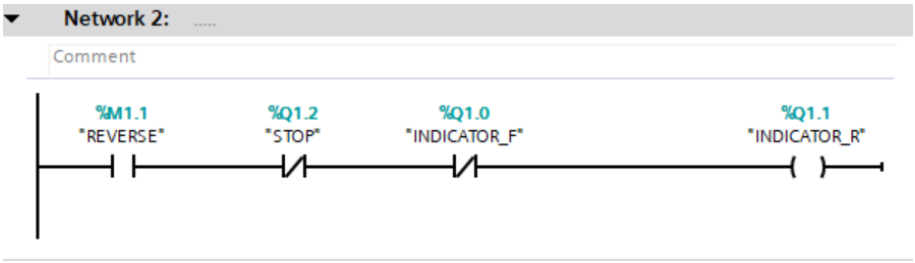


Figure III-2: Network 2

- **Network 3:** ON/OFF Mechanism

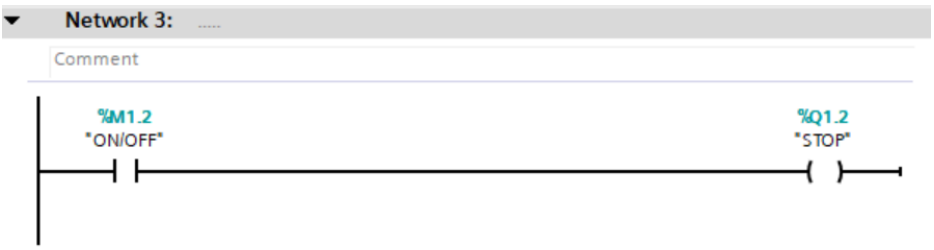


Figure III-3: Network 3

- **Network 4:** Variable Speed Drive Mechanism Move Function

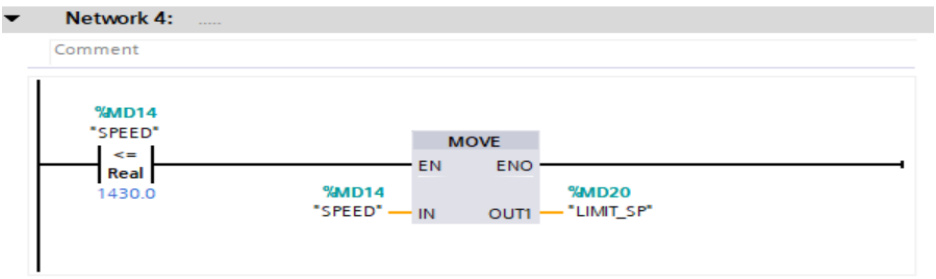


Figure III-4: Network 4

- **Network 5:** Variable Speed Drive Mechanism Unscale Function

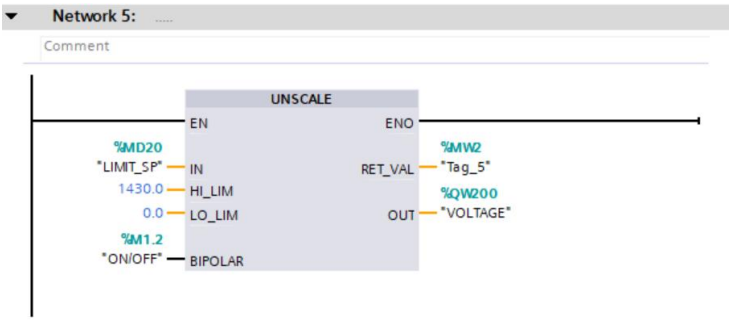


Figure III-5: Network 5

III.2.3-LabVIEW program

For efficient motor control, the TIA Portal is linked through the NI OPC server, to the LABVIEW. A set of essential components is required. These include LabVIEW buttons, lights, knobs, and bound variables. LabVIEW buttons serve as intuitive interfaces for controlling motor motion, providing options such as forward, reverse, and start/stop functionality. LEDs provide visual feedback on motor status, providing real-time monitoring. Knobs allow fine adjustment of motor parameters, such as speed and/or frequency. Bound variables facilitate transparent communication between LabVIEW and the TIA Portal, ensuring synchronized motor control operations. With this setup, we can harness the power of LabVIEW and TIA Portal, leveraging NI OPC servers for seamless motor control and monitoring.

III.2.3.1-Wiring and bound contact

The wiring and bonding program is divided into two sections the Inputs and the Outputs, with each section representing a sequence.

The sections are:

- **Section one:** Inputs (Push Buttons, Knob)

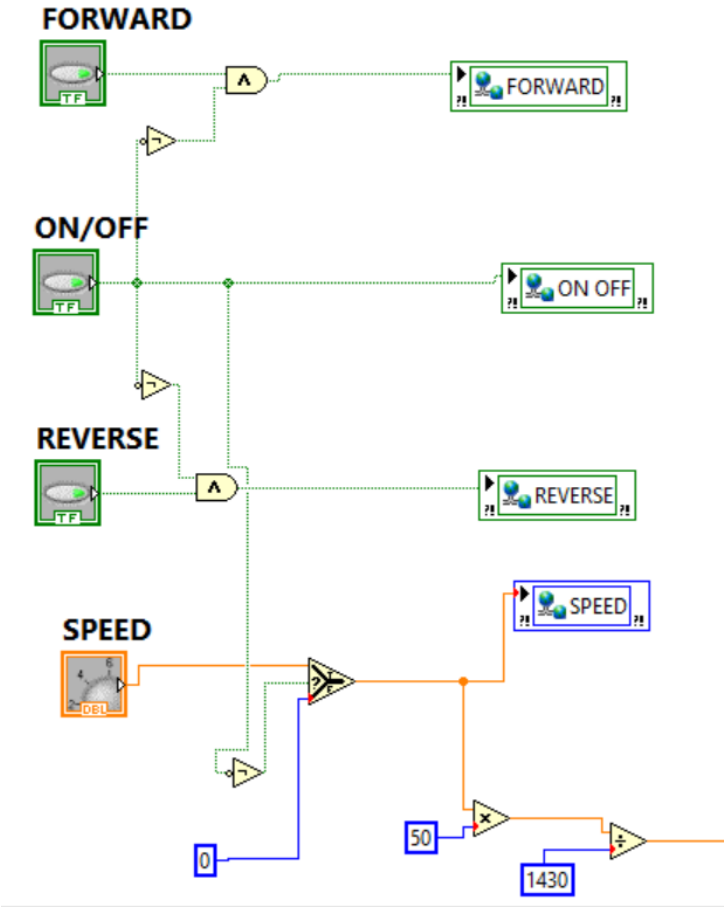


Figure III-6: The Input Section

- **Section two:** Outputs (LED's ,Display)

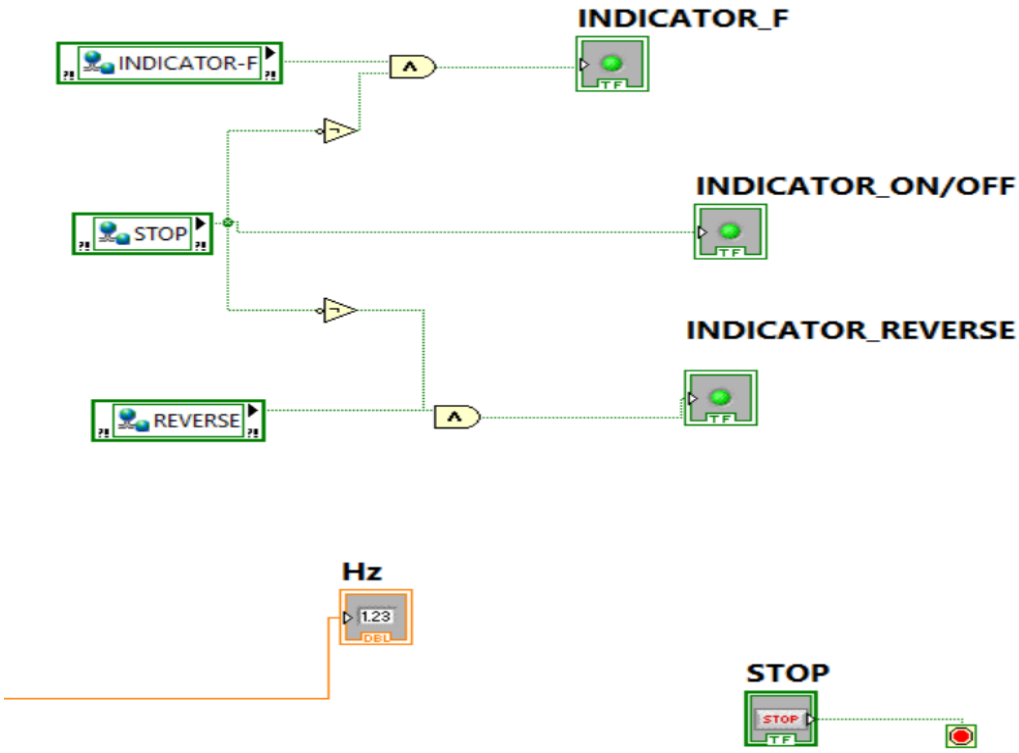


Figure III-7: The Output Section

III.3-Program Simulation

III.3.1-TIA Portal

Once the programs are developed, TIA Portal enables their simulation through its PLC SIM extension. The simulation process involves compiling the programs and loading them into the simulated controller using the simulation toolbar located at the top of the window. This allows us to test and validate the functionality of the programs within a simulated environment, ensuring their proper execution before actual implementation. TIA Portal's simulation capabilities provide a valuable tool for troubleshooting and optimizing program performance.

The following actions are launched on the Host PC:

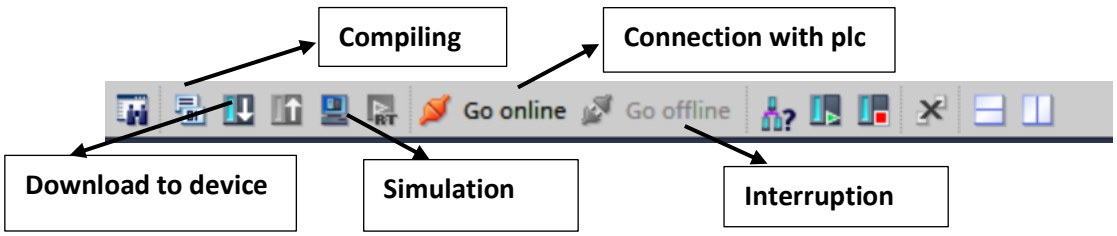


Figure III-8: Tia portal simulation bar

Before simulating the program, we load the program to the device.

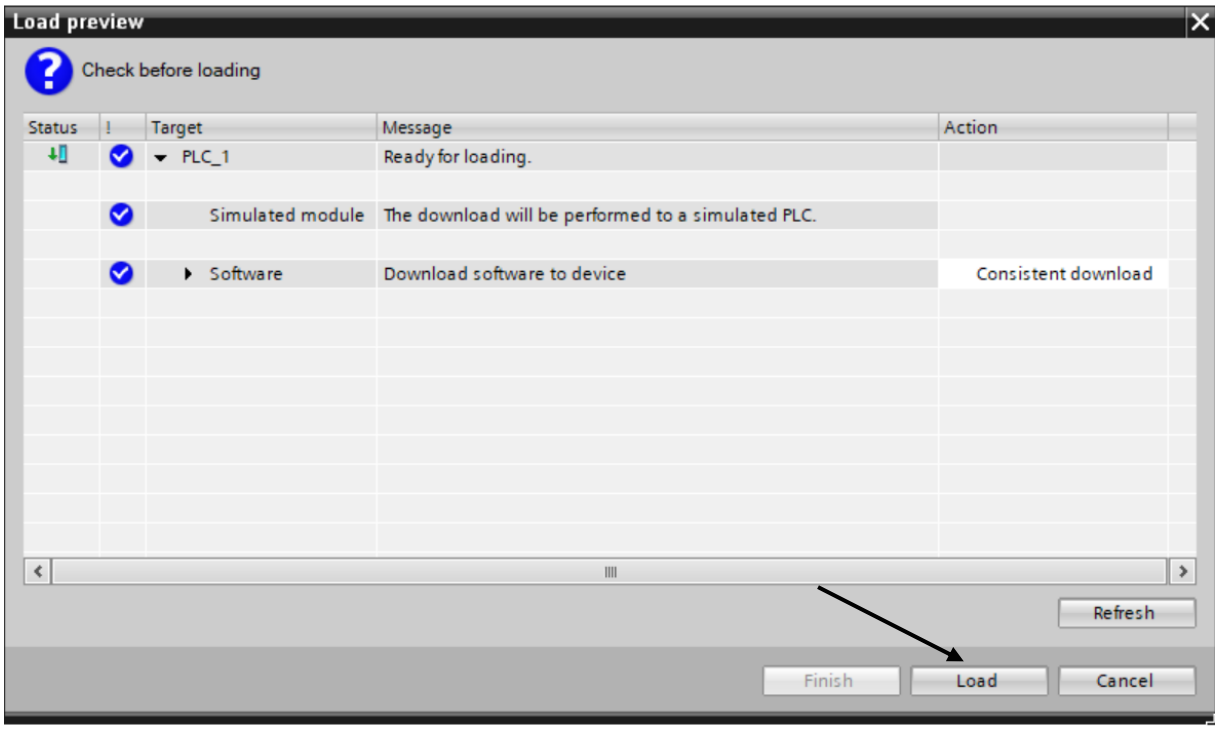


Figure III-9: Program loading

The next part is downloading the program to the device

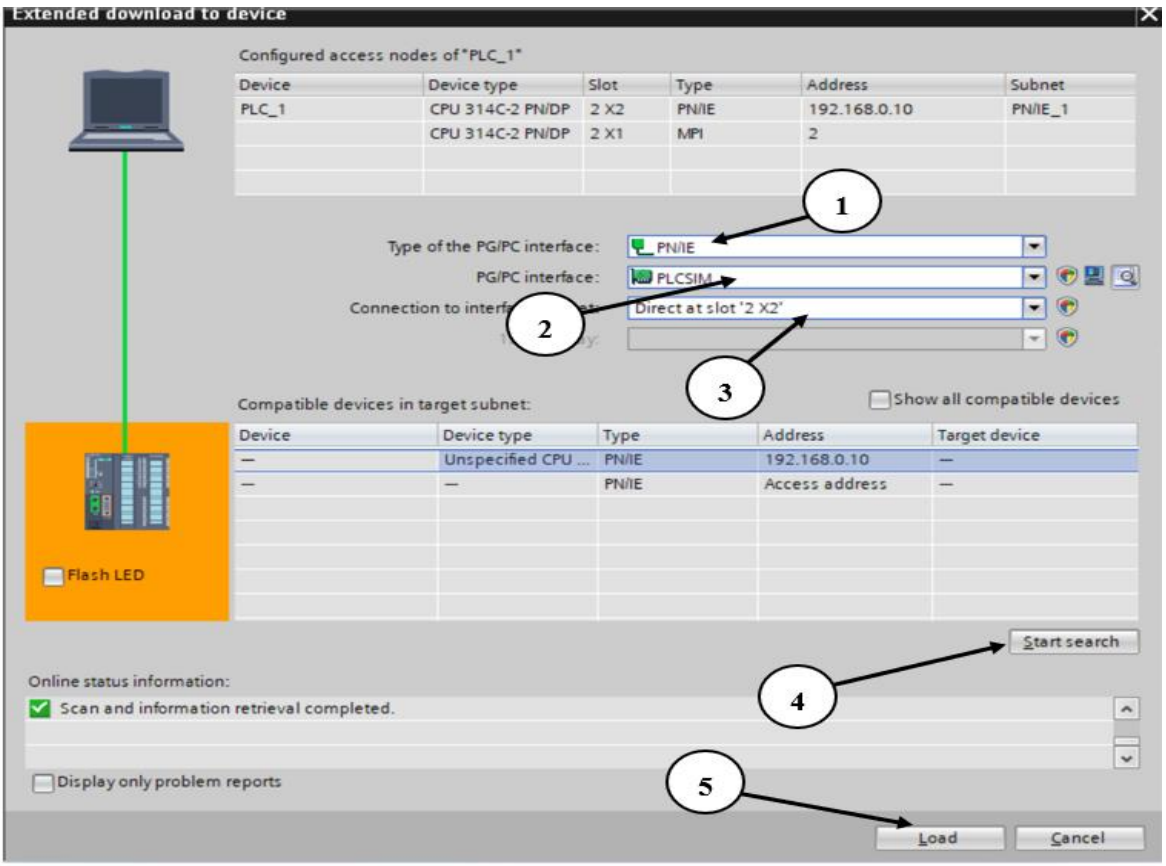


Figure III-10: downloading to device

Next, we start the simulation and launch the PLC Sim

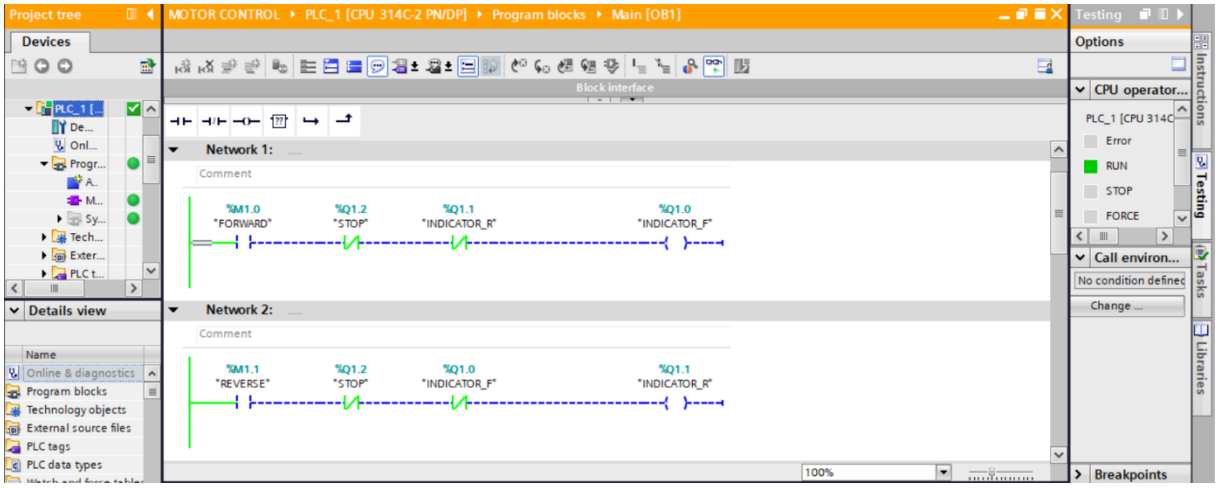


Figure III-11: Starting the simulation and monitoring

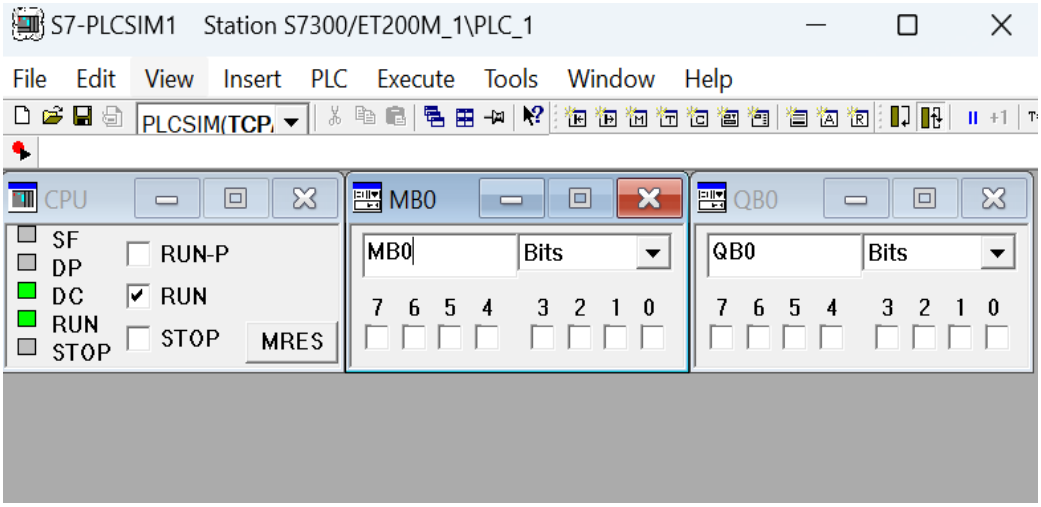


Figure III-12: Starting the PLC SIM

III.3.2-Net to PLC Sim

Next, we create our virtual link between the TIA Portal and the NI OPC Servers using the Net to Plc Sim software. In this part, we use the Host PC's IPv4 address and the PLC sim IP address that we obtain from the Tia Portal this process is launched in both: the main PC and the Host PC.

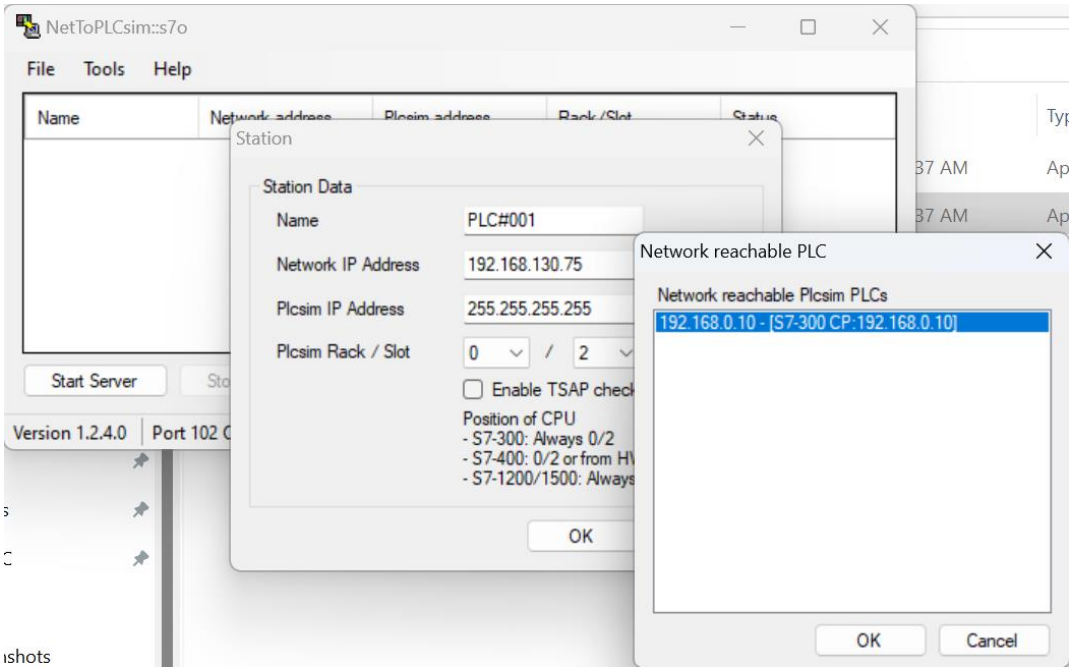


Figure III-13: Configuration of Net to PLC Sim

III.3.3-NI OPC Servers:

Afterward, we launch our NI OPC Server Quick Client on both the main PC and the Host PC. This part has to be maintained in order to keep the link going between the two PCs.

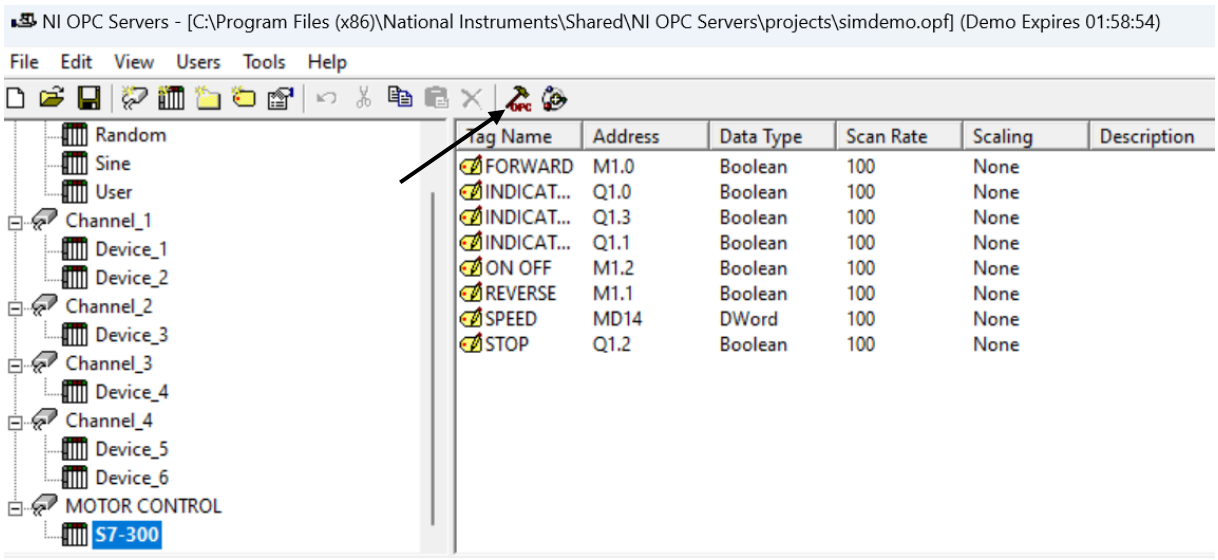


Figure III-14: Launching the NI OPC Servers Quick Client

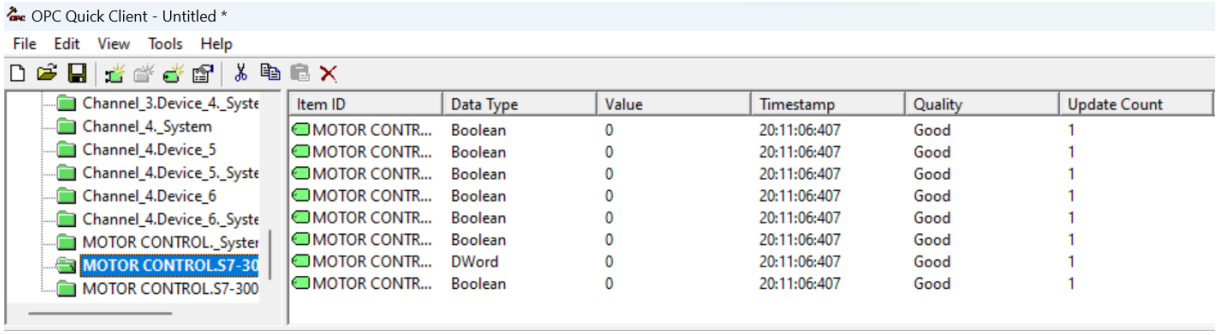


Figure III-15: NI OPC Quick Client

If the Quality of the server is 'good' as shown in the previous figure in the NI OPC Client then the connection between TIA Portal and the NI OPC server is working properly.

III.3.4-LabView

After creating the connection between the Tia Portal and NI OPC servers, we complete the link by launching the LabVIEW Simulation, this action is done on the Main PC Only.

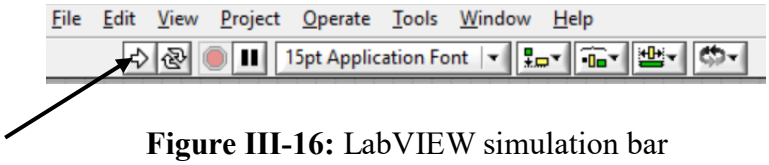


Figure III-16: LabVIEW simulation bar

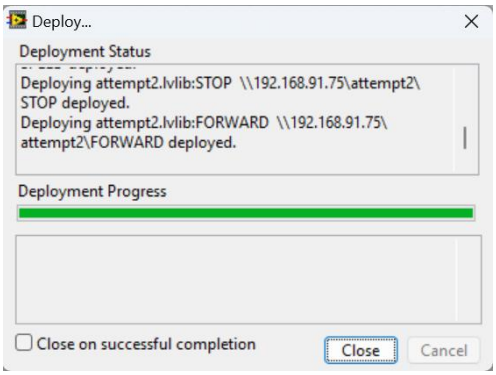


Figure III-17: Launching the LabVIEW

III.4-Results

After creating the Link between the Main Pc and the Host PC we can now View and maintain the results.

The PLCSIM application and the LabVIEW have allowed us to simulate the operation of an S7-300 programmable logic controller. We can test our control programs using LabVIEW without the need to connect the S7-300 to the hardware. PLC SIM provides a graphical user

interface that allows us to visualize and modify variables in the control program, execute the simulation CPU in single-cycle or continuous-cycle mode, and modify the operational state of the simulation API.

Then we have the LabVIEW that can actually show us the results of our several actions that are listed below.

Please note that the Host PC is equipped with the TIA Portal and has the virtual link activated (NI OPC Servers plus the Net to PLC sim), while the Main PC is running LabVIEW and has the virtual link enabled.

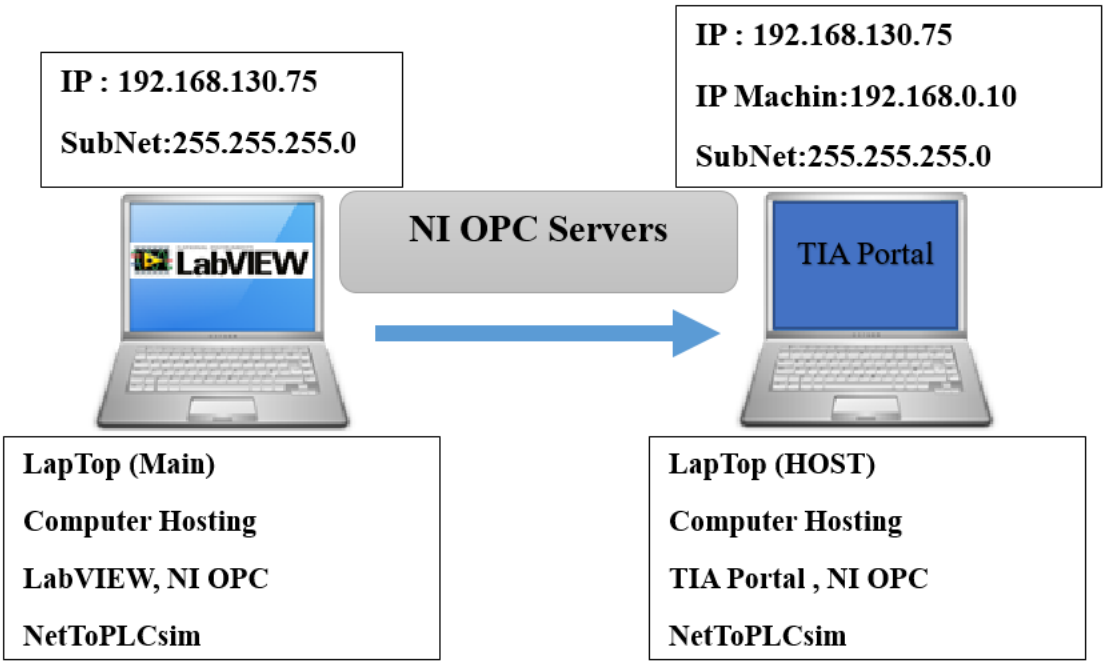


Figure III-18: Actual Network Configuration

- **First action:** The device is currently in an active state of "ON" without any further actions being performed.

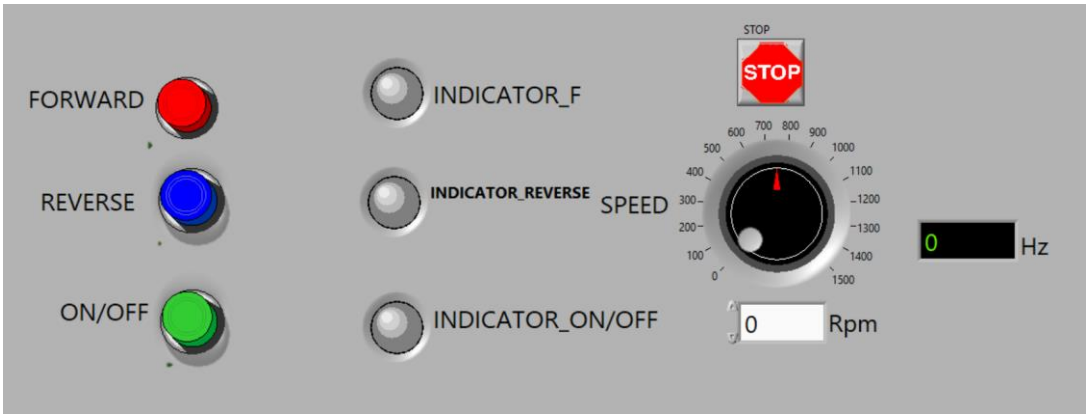


Figure III-19: First action

Upon the initial deployment of the program, in the primary stage, a marginal adjustment in the update count becomes apparent on the NI OPC Quick Client.

Item ID	Data Type	Value	Timestamp	Quality	Update Count
MOTOR CONTROL.S7-300.FORWARD	Boolean	0	21:00:17:039	Good	3
MOTOR CONTROL.S7-300.INDICATOR-F	Boolean	0	21:00:17:039	Good	3
MOTOR CONTROL.S7-300.INDICATOR-ON-OFF	Boolean	0	20:11:06:407	Good	1
MOTOR CONTROL.S7-300.INDICATOR-R	Boolean	0	20:11:06:407	Good	1
MOTOR CONTROL.S7-300.ON OFF	Boolean	0	21:20:16:482	Good	3
MOTOR CONTROL.S7-300.REVERSE	Boolean	0	20:11:06:407	Good	1
MOTOR CONTROL.S7-300.SPEED	DWord	0	21:20:18:796	Good	5
MOTOR CONTROL.S7-300.STOP	Boolean	0	21:20:16:482	Good	3

Figure III-20: First action: count update

We have observed a significant increase in the update count for the SPEED variable primarily due to the device operating at its lowest speed setting.

- **Second action:** In this scenario, we adjust the variable speed device knob to a higher setting (750 Rpm) in order to discern the distinction between the lowest position and an elevated position.

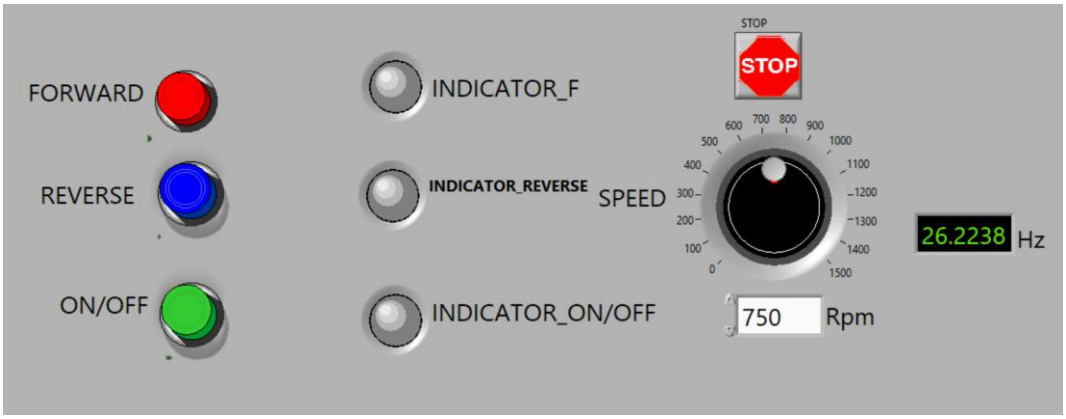


Figure III-21: Second action

We notice a change in the Hz display (26.2238).

Item ID	Data Type	Value	Timestamp	Quality	Update Count
MOTOR CONTROL.S7-300.FORWARD	Boolean	0	21:00:17:039	Good	3
MOTOR CONTROL.S7-300.INDICATOR-F	Boolean	0	21:00:17:039	Good	3
MOTOR CONTROL.S7-300.INDICATOR-ON-OFF	Boolean	0	20:11:06:407	Good	1
MOTOR CONTROL.S7-300.INDICATOR-R	Boolean	0	20:11:06:407	Good	1
MOTOR CONTROL.S7-300.ON OFF	Boolean	0	21:20:16:482	Good	3
MOTOR CONTROL.S7-300.REVERSE	Boolean	0	20:11:06:407	Good	1
MOTOR CONTROL.S7-300.SPEED	DWord	750	21:43:10:821	Good	25
MOTOR CONTROL.S7-300.STOP	Boolean	0	21:20:16:482	Good	3

Figure III-22: Second action: update count

In the second action, we can see that the value has changed in Quick Client and so has the Update count.

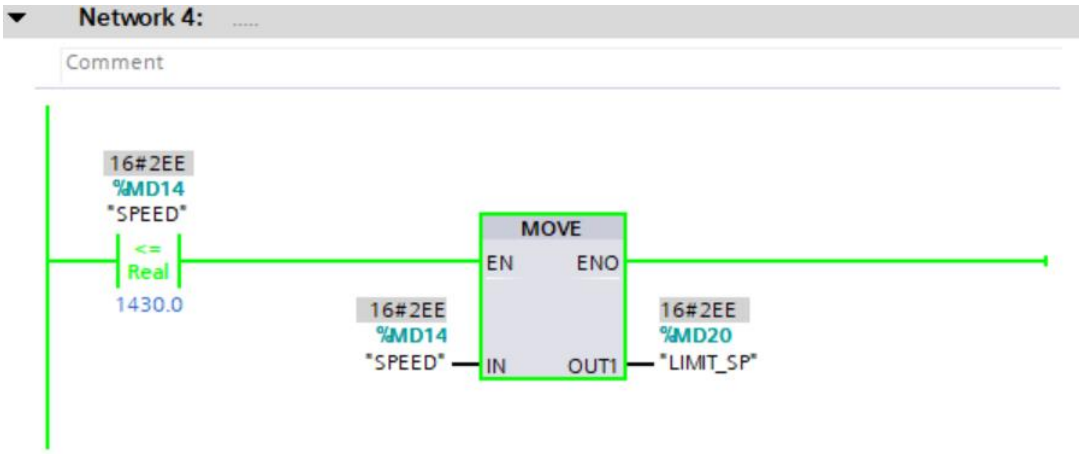


Figure III-23: Move function in the second action

The value of 750 Rpm is observed in the speed input of the Move function "#2EE," which will now be transmitted to the Unscale Function for conversion into its corresponding value of 26.2238 Hz. This value will then be displayed.

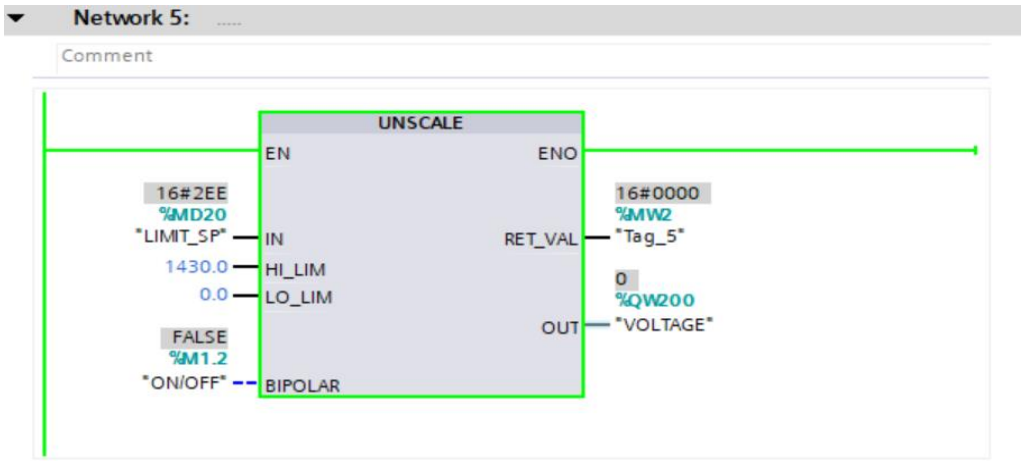


Figure III-24: Unscale function in the second action

Third action: In this particular scenario, our objective is to deactivate the system by engaging the ON/OFF button while maintaining the knob at its previous setting of 750 Rpm.

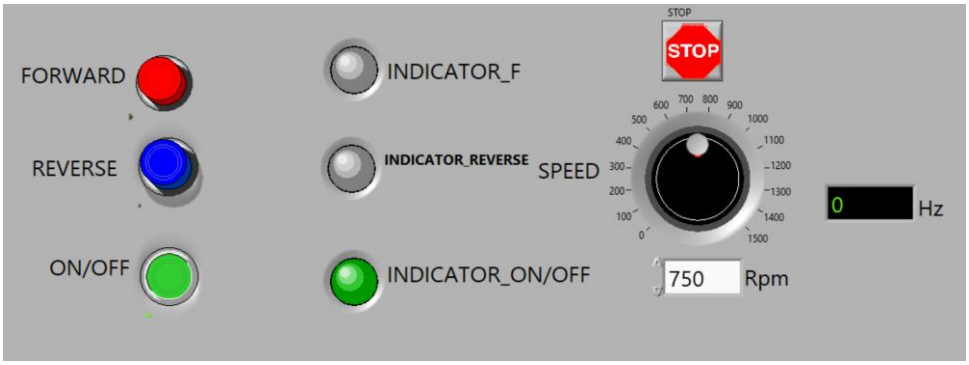


Figure III-25: Third action

We have observed that the indicator_ON/OFF is actively enabled in both LabVIEW and network 3, while no activity is detected on networks 4 and 5, and the Hz display does not show any value.



Figure III-26: Network 3

In the update count, all values are in a "0" state, except for the "ON/OFF" and "Stop" tags.

Item ID	Data Type	Value	Timestamp	Quality	Update Count
MOTOR CONTROL.S7-300.FORWARD	Boolean	0	21:00:17:039	Good	3
MOTOR CONTROL.S7-300.INDICATOR-F	Boolean	0	21:00:17:039	Good	3
MOTOR CONTROL.S7-300.INDICATOR-ON-OFF	Boolean	0	20:11:06:407	Good	1
MOTOR CONTROL.S7-300.INDICATOR-R	Boolean	0	20:11:06:407	Good	1
MOTOR CONTROL.S7-300.ON OFF	Boolean	1	22:01:47:751	Good	4
MOTOR CONTROL.S7-300.REVERSE	Boolean	0	20:11:06:407	Good	1
MOTOR CONTROL.S7-300.SPEED	DWord	0	22:01:47:751	Good	26
MOTOR CONTROL.S7-300.STOP	Boolean	1	22:01:47:751	Good	4

Figure III-27: Third action update count

- **Forth action:** In this procedure, we deactivate the ON/OFF switch while simultaneously engaging the forward button and adjusting the variable speed drive knob to an elevated setting (1090 Rpm).

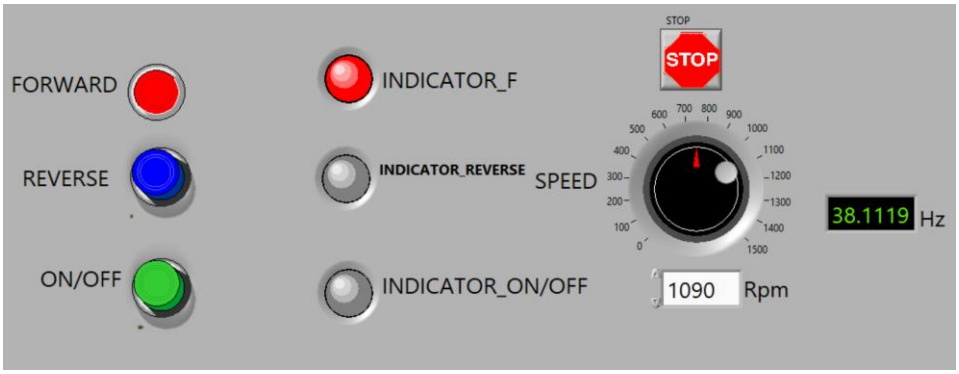


Figure III-28: Forth action

It is evident that INDICATOR_F is in an active state, as observed in network 1, with the Hz display indicating a new value of 38.1119 (Figure III.27).

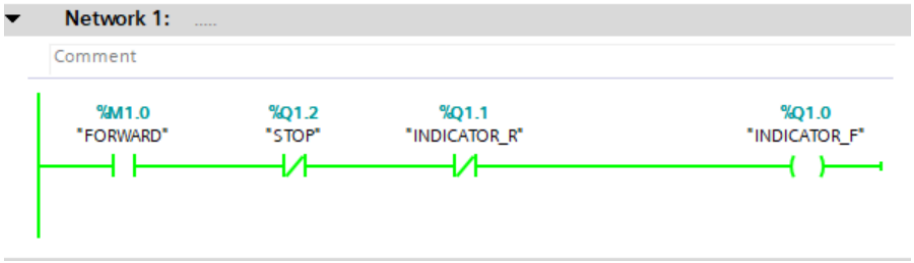


Figure III-29: Network 1

We also notice that there have been changes in both the number of updates and the value of the fast client.

Item ID	Data Type	Value	Timestamp	Quality	Update Count
MOTOR CONTROL.S7-300.FORWARD	Boolean	1	22:24:17:176	Good	5
MOTOR CONTROL.S7-300.INDICATOR-F	Boolean	1	22:24:17:176	Good	5
MOTOR CONTROL.S7-300.INDICATOR-ON-OFF	Boolean	0	22:23:41:803	Good	1
MOTOR CONTROL.S7-300.INDICATOR-R	Boolean	0	22:23:41:803	Good	1
MOTOR CONTROL.S7-300.ON OFF	Boolean	0	22:23:41:803	Good	1
MOTOR CONTROL.S7-300.REVERSE	Boolean	0	22:23:41:803	Good	1
MOTOR CONTROL.S7-300.SPEED	DWord	1090	22:23:41:803	Good	1
MOTOR CONTROL.S7-300.STOP	Boolean	0	22:23:41:803	Good	1

Figure III-30: Quick Client Update Count

- **Fifth action:** We activate the Reverse function and adjust the variable speed drive to a higher setting of 1500 RPM.

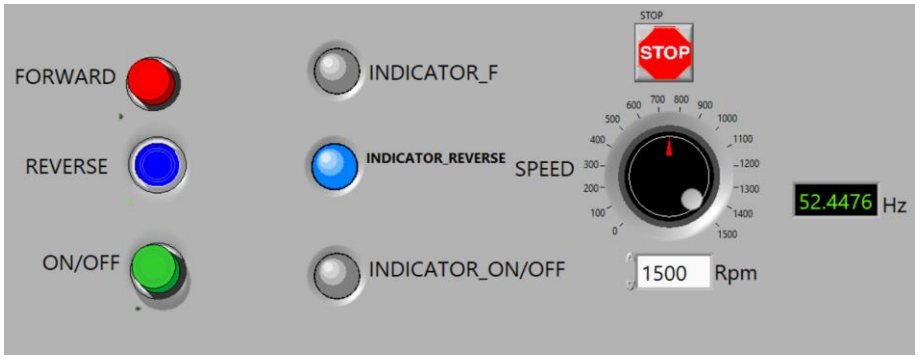


Figure III-31: Fifth action

In the current state, it is observed that the INDICATOR_REVERSE is in an active state. This alteration has also been observed in network 2, accompanied by a corresponding modification in the display value of the variable speed drive, which now reads 52.4476 Hz.

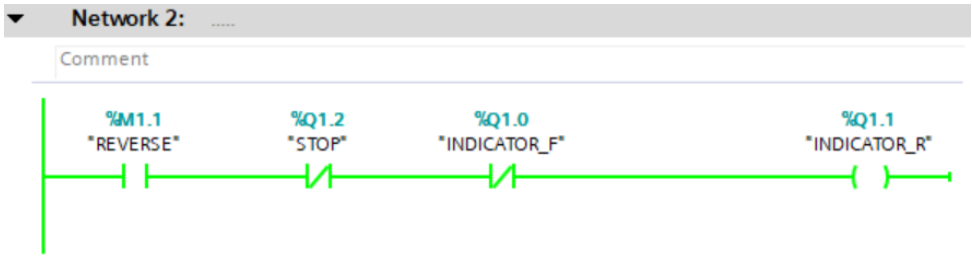


Figure III-32: Network 2

We also notice a change in the Quick Client value

Item ID	Data Type	Value	Timestamp	Quality	Update Count
MOTOR CONTROL.S7-300.FORWARD	Boolean	0	22:30:03:224	Good	6
MOTOR CONTROL.S7-300.INDICATOR-F	Boolean	0	22:30:03:224	Good	6
MOTOR CONTROL.S7-300.INDICATOR-ON-OFF	Boolean	0	22:23:41:803	Good	1
MOTOR CONTROL.S7-300.INDICATOR-R	Boolean	1	22:30:04:364	Good	2
MOTOR CONTROL.S7-300.ON OFF	Boolean	0	22:23:41:803	Good	1
MOTOR CONTROL.S7-300.REVERSE	Boolean	1	22:30:04:364	Good	2
MOTOR CONTROL.S7-300.SPEED	DWord	1500	22:30:08:207	Good	3
MOTOR CONTROL.S7-300.STOP	Boolean	0	22:23:41:803	Good	1

Figure III-33: Quick client update count

- **Sixth action:** In this part, we engage the STOP Button while maintaining the variable speed drive knob at its maximum setting of 1500rpm.

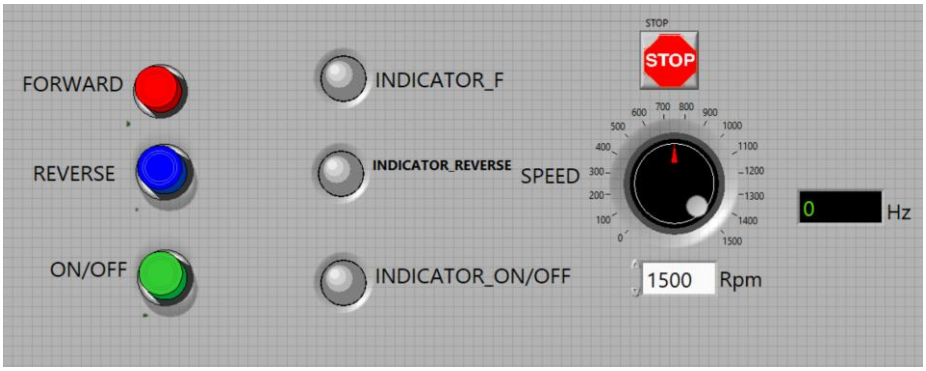


Figure III-34: Sixth action

We have observed a complete shutdown of the LabVIEW program, accompanied by the absence of any displayed values on the Hz display.

Item ID	Data Type	Value	Timestamp	Quality	Update Count
MOTOR CONTROL.S7-300.FORWARD	Boolean	0	22:30:03:224	Good	6
MOTOR CONTROL.S7-300.INDICATOR-F	Boolean	0	22:30:03:224	Good	6
MOTOR CONTROL.S7-300.INDICATOR-ON-OFF	Boolean	0	22:23:41:803	Good	1
MOTOR CONTROL.S7-300.INDICATOR-R	Boolean	0	22:43:21:012	Good	3
MOTOR CONTROL.S7-300.ON OFF	Boolean	0	22:23:41:803	Good	1
MOTOR CONTROL.S7-300.REVERSE	Boolean	0	22:43:21:012	Good	3
MOTOR CONTROL.S7-300.SPEED	DWord	0	22:45:43:983	Good	14
MOTOR CONTROL.S7-300.STOP	Boolean	0	22:23:41:803	Good	1

Figure III-35: Quick client update count

It is evident that all values have reached their minimum, indicating a complete absence of connection.

III.5- Conclusion:

In this chapter, we have simulated the control of the motor with its various buttons with the possibility of varying the speed, using LabVIEW from a main PC and TIA PORTAL from a second PC, connected via the NI server OPC. Through the utilization of TIA PORTAL and the NI OPC Quick Client, we were able to observe and analyze the outcomes of each action, providing valuable insights into the motor's behavior and performance. The integration of LabVIEW and TIA PORTAL, enabled by the NI OPC server, allowed for synchronized control and monitoring of the motor system from separate PCs. We have realized six different situations (action). The results obtained testify to the versatility and adaptability of this control configuration, showing its potential for application in various industrial applications.

Conclusion

Controlling the direction and speed of an asynchronous motor in industrial applications is usually done using variable frequency drives (VFDs). VFDs control the frequency of the electrical power supplied to the motor, which then regulates the speed. They also control the direction of the motor by changing the frequency and phase order of the power supply. The VFD can be controlled remotely via a programmable logic controller (PLC), LabVIEW (used as a human-machine interface) as in our case, and NI OPC servers, which communicate with LabVIEW via Ethernet for data exchange purposes and automation. This allows precise and efficient control of industrial processes that require variable motor speeds and directions.

The aim of our work is to remotely control the direction and speed of an asynchronous motor using a simulator. This will allow us to learn and practice the skills needed to operate and control the motor in real-world applications, an essential skill in many industrial settings.

To achieve this goal, we have divided our work into three chapters. The first chapter gave an overview of the state of the art of industrial automation systems. He highlighted the importance of mimics and PID control and critical hardware components used in industrial automation such as the asynchronous machine, variable frequency drive (VFD), and programmable logic controller (PLC).

In Chapter Two, in-depth research of the various features and capabilities of TIA Portal and LabVIEW, such as their programming languages, data processing tools, and graphical user interfaces was developed. The TIA Portal is widely used in industries for automation control and visualization, while LabVIEW is a popular software environment for data acquisition and analysis. The study further discussed the benefits of integrating these software tools with NI OPC servers, which provide a common platform for data exchange between various industrial devices and software programs.

The third chapter concentrates on the programming of various software and emphasizes the development and implementation of the code. Simulation techniques are then used to demonstrate how virtual environments reproduce real-world scenarios, and the results obtained are discussed, analysed, and interpreted. Indeed, the programming aspects of the TIA Portal and LabView are also presented in detail, with the objective of controlling a motor via LabView from a PC using a set of six distinct scenarios. These scenarios were designed to respond to three buttons (forward, reverse, and on/off) and a speed controller knob to showcase the versatility of the LabView platform in the motion control of the engine.

Throughout our project, we encountered several challenges that forced us to adapt our approach. First of all, we were faced with the problem of the unavailability of the S7-300 PLC with an Ethernet bus. Second, we encountered the obstacle of installing the 2016 version of the NI OPC servers. These obstacles forced us to modify our course of action, seeking alternative solutions to effectively overcome these limitations.

Finally, we would like our project to be completed by a practical realization, which can only be accomplished if we can obtain the necessary materials and resources for the effective completion of this project.

Bibliographic

[1]. F. Bettache ,S. Bentouati , A. Abbadi and F.Hamidia Simulated Ethernet Control AC Motor via PLC Using LabVIEW ,2022

[2]. Mendes, M.J.G.C. and L. Martins. An internet remote laboratory to teach Industrial Automation. in 2014 7th International Conference on Human System Interactions (HSI). 2014.

[3]. Groover, M.P., Automation, production systems, and computer-integrated manufacturing2016: Pearson Education India.

[4]. Stouffer, K., J. Falco, and K. Scarfone, Guide to industrial control systems (ICS) security. NIST special publication, 2011. **800**(82): p. 16-16.

[5]. Patterson, D.A. and J.L. Hennessy, Computer organization and design: the hardware/software interface 5th ed, 2014, Elsevier.

[6]. Laugier, C., Synoptic of the Research Activity (Updated version, Oct. 2007), 2007, INRIA Grenoble-Rhone-Alpes; LIG (Laboratoire informatique de Grenoble).

[7]. Lamb, F., Industrial Automation: Hands On2013: McGraw Hill LLC.

[8]. Fiset, J.-Y., Human-machine interface design for process control applications2009: ISA.

[9]. Ang, K.H., G. Chong, and Y. Li, PID control system analysis, design, and technology. IEEE transactions on control systems technology, 2005. **13**(4): p. 559-576.

[10]. Correvon, M., Systèmes électroniques, Chapitre 7 (Les Régulateurs Standards): Haute Ecole Spécialisée de Suisse Occidentale.

[11]. BETTACHE, F., M. BOUHEDDA, and A. ABBADI, IMPLÉMENTATION D'UN RÉGULATEUR PI ADAPTATIF FLOU SUR UN AUTOMATE PROGRAMMABLE INDUSTRIEL S7-300 EN VUE DE LA COMMANDE EN VITESSE D'UNE MACHINE ASYNCHRONE. REVUE ROUMAINE DES SCIENCES TECHNIQUES—SÉRIE ÉLECTROTECHNIQUE ET ÉNERGÉTIQUE, 2022. **67**(3): p. 253-258.

[12]. Gdaim, S., Commande directe de couple d'un moteur asynchrone à base de techniques intelligentes, 2013, Ecole Nationale d'Ingénieurs de Monastir. Tunisia.

[13]. Yahia, K., Estimation en ligne de l'état et des paramètres du moteur asynchrone triphasé, 2005, Université Mohamed Khider Biskra.

[14]. BAYLE, B., Technologie des asservissements variateurs de vitesse. Ecole Nationale Supérieure de Physique de Strasbourg, année, 2010. **2011**.

[15]. NEGUEZ, S., Association variateur de vitesse et moteur asynchrone avec communication Profinet par automate programmable siemens S7-300, 2022, university of M'sila.

[16]. V. Vyatkin, "Software engineering in industrial automation: State-of-the-art review," IEEE Transactions on Industrial Informatics, vol. 9, no. 3, pp. 1234-1249, 2013.

[17]. L. Wang and K. C. Tan, Modern Industrial Automation Software Design. John Wiley & Sons, 2006.

[18]. L. Hartmann and S. Wendzel, "Anomaly Detection for Industrial Control Systems Through Totally Integrated Automation Portal Project History."

[19]. H. Berger, Automating with STEP 7 in STL and SCL: SIMATIC S7-300/400 programmable controllers. John Wiley & Sons, 2012.

[20]. H. Berger, Automating with SIMATIC: controllers, software, programming, data. John Wiley & Sons, 2012.

[21]. H. Salih, H. Abdelwahab, and A. Abdallah, "Automation design for a syrup production line using Siemens PLC S7-300 and TIA Portal software," in 2017 International Conference on Communication, Control, Computing and Electronics Engineering (ICCCCEE), 2017: IEEE, pp. 1-5.

[22]. P. Tiernan, "Enhancing the learning experience of undergraduate technology students with LabVIEW™ software," Computers & Education, vol. 55, no. 4, pp. 1579-1588, 2010/12/01/ 2010, doi: https://doi.org/10.1016/j.compedu.2010.07.001.

[23]. J. Conway and S. Watts, A software engineering approach to LabVIEW. Prentice Hall Professional, 2003.

[24]. L. J. Barreiro Chevez and J. M. Molina Peñaranda, "Diseño e implementación de un control de planta de nivel Gunt RT450 del laboratorio de instrumentación industrial utilizando un PLC Simatic S7 300 con automatización total integrada (TIA Portal) e implementación de una interfaz para usuario diseñada en Labview con comunicación mediante OPC," Espol, 2017.

[25]. I. D. Pranowo and D. Artanto, "Improved control and monitor two different PLC using LabVIEW and NI-OPC server," International Journal of Electrical & Computer Engineering (2088-8708), vol. 11, no. 4, 2021.

[26]. I. González, A. J. Calderón, A. J. Barragán, and J. M. Andújar, "Integration of Sensors, Controllers and Instruments Using a Novel OPC Architecture," Sensors, vol. 17, no. 7, p. 1512, 2017. [Online]. Available: https://www.mdpi.com/1424-8220/17/7/1512.

[27]. M. S. Hans-Petter Halvorsen. "LabView OPC with NI OPC Servers." http://www.usn.no (accessed 02/05/2023).