NATIONAL UNIVERSITY OF ENGINEERING

FACULTY OF MECHANICAL ENGINEERING

SCHOOL OF MECHATRONIC ENGINEERING

PROJECT'S NAME:

VIRTUALIZATION OF MODULES OF TRAINING ORIENTED TO THE TRAINING OF STUDENTS IN THE AREA PROCESS CONTROL

PRODUCED BY:

CARLOS ALFREDO HUAMAN ORMEÑO

TEACHER:

ING. JOSÉ LUIS ALBITES SANABRIA

MT 818 - B

LIMA PERU

December 2020

Table of Contents

I. PROBLEMATIC SITUATION AND PROBLEM DEFINITION

I.1. Problematic situation

The benefits of automation in all industries are many: reduced production cycles, increased productivity, improvements in quality and reliability, reduced waste, better use of space, reduced costs, reduced accidents, etc. For these reasons, in these competitive times it is vital for every company to automate its operations, systems and machinery.

Peru is just beginning on this path of automation. But our country is still very far if we compare it with other countries in South America. According to the IFR Statistical Department, the number of robots in Central and South America in 2013 was 9,600 units, of which Peru only had about 20, which were installed using foreign technological resources, since our country was not prepared to provide adequate technical personnel for implementation and maintenance tasks.

In recent years, there has been a real interest in automating the productive sector in Peru. Many companies have started to introduce automation and industrial robots in their production processes, for example, manufacturing companies in the food sector, metal structures (welding), cement, the fishing industry and in the handling and packaging in all production lines.

Due to globalization, Peruvian companies have understood that they compete with others around the world and that the only way to do it is by automating, that is, improving their productivity, quality and reliability.

Although it is true that automation in Peru will displace many people who performed these manual tasks, at the same time many job opportunities will be created for personnel with a higher level of technological training and professionals in the field must be trained to attend this demand.

Industrial automation has had a rapid global advance thanks to the impulse and development of science and technology, particularly information technologies, making it possible in recent years to integrate the different functions of the company. This multidisciplinary integration requires the interaction between the various fields of engineering and the intensive use of computational tools.

I.2. Definition of the problem

I.2.1. General Problem

Training modules in the area of process control are not accessible for the reinforcement of the theoretical teaching of the student and the training centers.

I.2.2. Engineering Problem

What technological and engineering considerations of mechatronic design and software design must be taken into account, in order to develop a training module in the process control area that satisfies the requirements of the student and the training centers at the time to develop classes oriented to topics related to that area?

II. STATE OF ART

II.1. Existing products and solutions

DL 314 Process Control Trainer [1]

Manufacturer: De Lorenzo

The coach is composed of:

- A didactic panel, with a pressurized tank and a set of level, pressure, temperature and flow sensors and actuators.
- A control module, which contains the interface circuits for the sensors and actuators and ON / OFF, proportional, integral and derivative (PID) control circuits.

Advantage:

- The student recognizes a real physical control system and its parts.

Disadvantage:

- You need a storage space.
- It needs physical maintenance, both as a module and its individual parts.
- Limited access for manipulation by students when the amount is greater than the available modules.

- High acquisition costs that increase when the system is more complex.

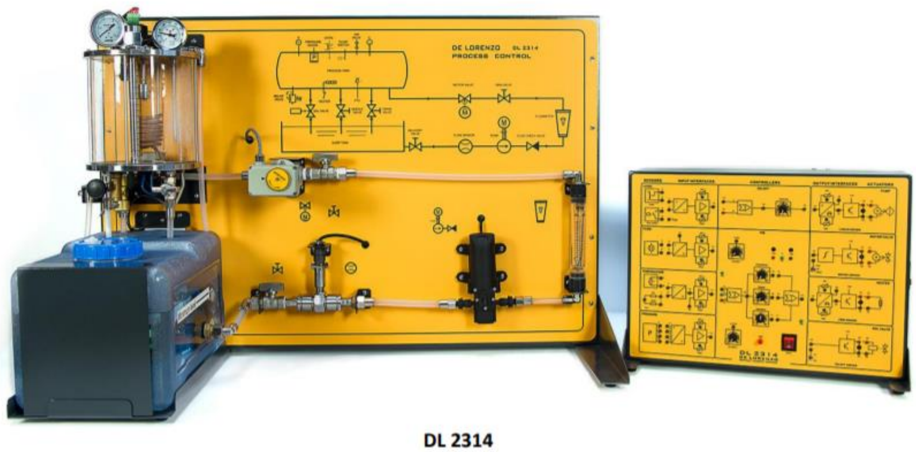


Figure 1: DL 314 Process Control Trainer

Building automation simulator: heating and air conditioning systems

Manufacturer: GUNT Hamburg [2]

The educational module is oriented to the operation and programming of a DDC (Direct Digital Control) or a PLC for a heating and air conditioning system. The process schematics for the two simulated systems are displayed on the front panel.

The simulated heating system has two boilers connected in parallel. The temperature of the feed stream is controlled via the outside temperature by 3-

way mixing valves. There are two heating lines with simulated heaters and different actuators: 3-way mixing valves and pumps. An additional 3-way mixing valve is the actuator for water heating.

Advantage:

- Compact system, very complex systems and simplification of connection systems can be simulated.
- Actuators and sensors do not require much maintenance.

Disadvantage:

- You need a storage space.
- When simulated it does not represent the real behavior of an air conditioning system.
- Strange behavior of the system depends on the nature of the simulated elements and not on the system.
- Limited access for manipulation by students when the amount is greater than the available modules.
- High acquisition costs.

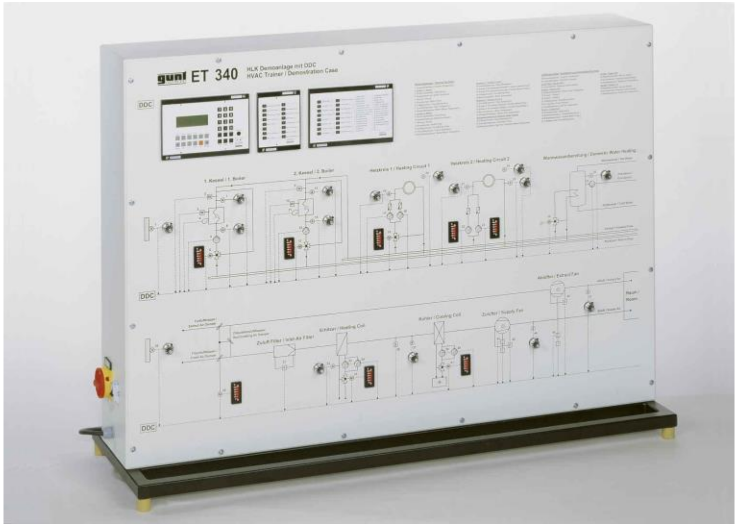


Figure 2: Building Automation Module in heating and air conditioning systems

II.2. Scientific / Engineering Publications

Cruz and Vera (2020) [3] in the publication entitled "Convolutional neural networks for the Hass avocado classification using LabVIEW in an agro-industrial plant" published in 2020 IEEE XXVII International Conference on Electronics, Electrical Engineering and Computing (INTERCON) presents the proposal of a plant, the hardware part consists of Aca2500 Basler camera, HR 2mm / F1 lens, and conveyor belt 1200, the software part of the PLC STEP S7: TIA PORTAL (OPC), a sequential algorithm, and convolutional neural network decision in which include the avocado size and color selection parameters. The classification process has three main stages: image acquisition, image processing and image recognition. Convolutional neural networks were used for image processing, obtaining an average classification precision of 60% in real time. From the results obtained, he concludes that the classification can be improved.

Duque, López and Navas (2017) [4] in the publication entitled "Auto-tuning of a PID controller implemented in a PLC using swarm intelligence" published in the Prospectiva magazine of the Faculty of Engineering of the Autonomous University of the Caribbean presents the proposal implementation in an Allen Bradley PLC of a swarm intelligence technique whose function is to determine the mathematical model of a system or process. With the system model found, the parameters of a PID controller are calculated that guarantee a desired behavior of the system. The swarm intelligence technique used to model the process is known as particle swarm optimization.

To test the operation of the algorithm implemented in the PLC, simulations of dynamic systems were carried out on a desktop computer with Matlab software, from this software a connection was established with the PLC using the OPC (OLE for Process Control) communication standard. In this way, Matlab sends information about the status of the process, and the PLC estimates the mathematical model of the system, tunes the PID controller and then sends the PC an appropriate control action. The results showed that when comparing the performance of a fixed PID with the self-tuned PID, their performances are similar, which is a good starting point for future improvements to the presented scheme.

Ospino-Pinedo, Esteban and Mantalla (2016) [5] in the publication entitled “Modeling and control of processes using the EMSO simulator and SIMATIC PLC as OPC interface” published in the Universidad de Antioquia Faculty of Engineering Magazine presents the modeling procedure and level and flow process control using the EMSO simulator coupled to the process with a SIMATIC S7-200 PLC as the OPC communication interface between the process instrumentation and the controller's computer system. For the supervision of the process variables, a SIMULINK model connected to the process through OPC communication The obtained model was used in the design of PID and PBSM controllers, which were tested against set point jumps and an imposed set point trajectory.

Among the results obtained, the PBSM indicated for the process module was in strong agreement both qualitatively and quantitatively with the experimental data obtained, and produced low deviation errors. Regarding the control systems implemented, in general the performance observed for the controllers was good. The PID controller worked fine for set point pitch disturbances, but failed for an imposed set point path. On the other hand, the PBSM-based controller did a great job in both cases, despite some small and continuous oscillations around the Set-Point. Additionally, time delays were observed when coupling the EMSO simulator, EMSO-OPC interface, and the actual process module, due to problems in the synchronization due to the delays generated by the sampling time of the EMSO-OPC interface and the calculation time. used by the simulator. As a final note for future work, it was observed that the variation of the sampling time parameter can improve the performance of the control system.

III. Justification

- The proposed solution solves the problematic situation posed because it will allow the development of a virtualized system that is capable of replacing and coexisting with existing solutions.

- The development of the proposed solution implies solving different problems in the training of the student, it will develop functions which the existing solutions do not have.

- It is estimated that the product to be generated could be cheaper than the current solutions, since the necessary resources for its implementation are available at the National University of Engineering.

- The proposed product follows the guidelines proposed for the courses implemented at the National Engineering University.

IV. Objectives

a. General objective

- Implement virtual training modules aimed at training students in the area of process control.

b. Specific objectives

- Select the technologies that will allow the implementation of virtual educational modules.
- Identify model plant to virtualize.
- Identify which are the physical variables necessary to carry out the correct modeling of the model plant.
- Model the individual behavior of each instrument involved in the model plant.
- Select the real instruments to virtualize.

V. Description of the proposed solution

a. Description

a.1. Pictorial block diagram:

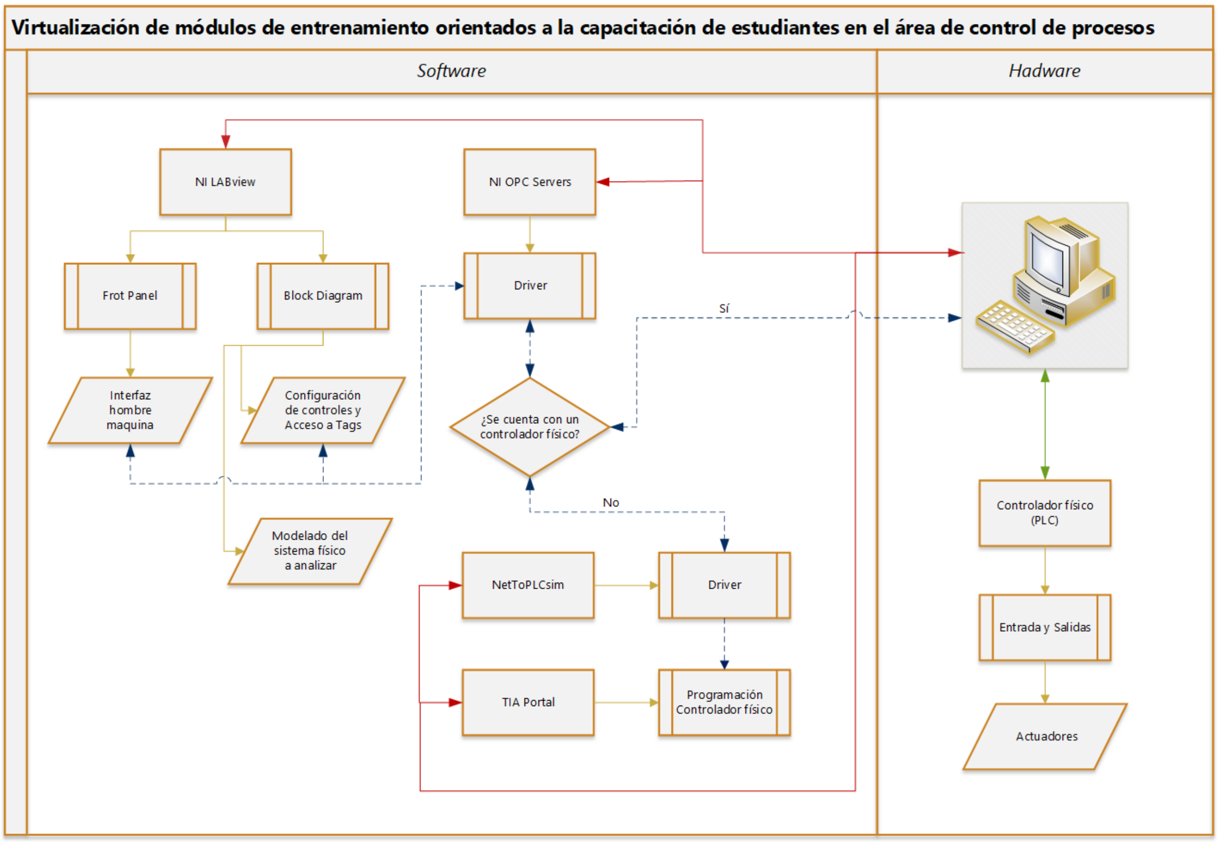


Figure 3: Pictorial block diagram.

a.2. Functioning:

The software developed will present a man-machine interface, with the necessary controls for the total description of the pilot plant, where the user can modify the plant according to the teacher's instructions.

The internal programming will have two parts, one dedicated to the configuration of the interface (controllers and alarms) connected to the physical controller (partial virtualization) or virtual controller (full virtualization) through the necessary drivers on demand (OPC Server for partial virtualization and adding NeToPLCSim for full virtualization), the other dedicated to the implementation of the simulation, which after its evaluation in other software (MATLAB), will be subsequently implemented in the LabVIEW environment.

a.3. Devices and components to use:

Devices and Components	**Main Features**
Laptop	OS: Windows 10 RAM: 12 GB Processor: I7 - 6700HQ
NI Labview	Version: 2015 sp1
NI OPC Servers	Version: 2013
MATLAB	Version: R2018b
TIA Portal	Version: 16 Module: PLC Sim
NetToPLCSim	Version: 1.2.4.0
PLC	S-300: CPU 314C-2 PN / DP (Virtualization)

a.4. Limitations of the solution:

- The virtualized modules will be as complex as the case to be modeled.
- The full virtualization process (Software and Hardware) will be limited to the use of the TIA Portal software, NetToPLCSim, and the simulation capability of the PLCSim (Hardware Virtualization) module.
- The partial virtualization process (Software only) will be limited by the chosen OPC Server and the availability of the driver found in it.

a.5 Expected results:

- The results displayed by the Male Interface Machine I know will compare with the data obtained in other previously verified Process Control simulators to check its correct operation and accuracy of the model to be simulated.

b. Design and implementation methodology

b.1. Description

Stage 1: Recognition of the software to be used.

- Once the problem is identified, all the programs available in the market that are offered as part of the solution are recognized.
- It is discriminated according to program benefits: choice of the environment controller programming (physical and virtual), Human

Interface programming environment Machine, Simulator and communication drivers.

- The recognition of the necessary benefits for the pilot plant is developed.
- After the choice, the individual programs are connected.

Stage 2: Information gathering and characterization of the model plant.

- Identification of the plant, survey of the PI&D plant, recognition and selection of alternative commercial sensors and actuators.
- Mathematical modeling of the pilot plant: modeling of parts and as a whole, simulation in MATLAB program.

Stage 3: Implementation of Human Machine Interface

- Implementation of the mathematical model in the LabVIEW interface through the available LabVIEW modules.
- Implementation of the Male Interface Machine in the LabVIEW interface, creating custom drivers.

Stage 4: Implementation of the training module.

- Selection of the physical controller necessary for the programming of the plant.
- Sensor selection and alarms necessary for the plant.
- Construction of training module.

b.2. Blocks diagram

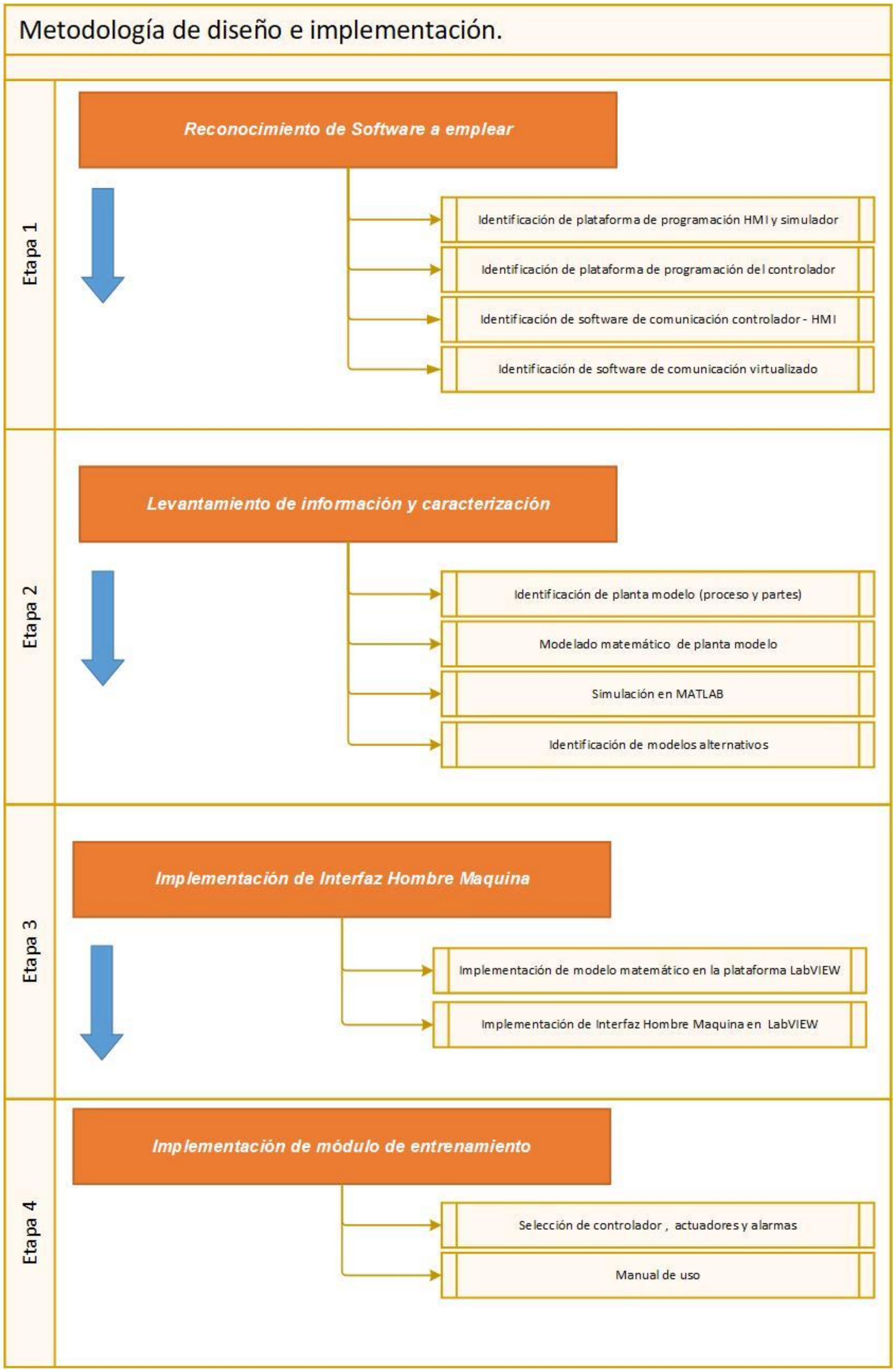


Figure 4: Design and implementation methodology.

VI. Applications and potential users of the product

The project is aimed at all those educational institutions and companies dedicated to training that have as part of their curricular plan the teaching of courses related to automation, instrumentation and process control.

VII. Viability

a. Technical viability

The development of the project is supported by the knowledge obtained during the training received at the university (Industrial Networks, Process Control, Sensors and signal conditioning) and in turn is oriented towards the implementation in the areas of practical reinforcement of the university.

b. Economic feasibility

The project is oriented towards virtualization, so its implementation will require software, which is used in training centers so they do not represent additional costs.

c. Social viability

The project, being oriented towards virtualization, reaches a greater number of students, which promotes the democratization of education.

d. Operational feasibility

The project is oriented to be implemented in training centers that have the necessary requirements for training in subjects oriented to process control, so that the compatibility of software and hardware is assured.

e. Alternatives

As indicated in the section and solve existing solutions, there are already products on the market that meet the requirements of training centers, but this product is not necessarily exclusive and replacement.

VIII. Weekly project development schedule

Type of activity	Activity	Week
Documentation	• MT818 regulation study	1
Documentation	• Establishment of project proposal • Development of chapters: I, II, Va, VII, IX. • Review of background and bibliographic sources.	2
Documentation	• Chapter development: V • Development of software connection documentation.	3
Technical	• Recognition of software and hardware to use.	
Documentation	• Development of digital signal test documentation.	4
Technical	• Software connection development. • Digital signal testing.	
Documentation	• Development of chapters III and IV • Development of documentation of controls associated with analog signals. • Review of regulations regarding HMI systems.	5
Technical	• Analog signal tests. • Recognition of controls associated with analog signals.	
Documentation	• Development of chapters: report. • Presentation 1 • Report Delivery 1 • Custom control development documentation.	6

Technical	• Development of custom controls. • Test custom controls with OPC connection • Documentation of the floor plan to simulate.	
Documentation	• Development of report chapters. • Survey of the floor plan to simulate.	7
Technical	• Mathematical modeling of the plant to simulate in MATLAB.	
Documentation	• Development of report chapters. • Analysis and modeling of alternatives to the plant to simulate.	8
Technical	• Analysis and modeling of alternatives to the plant to simulate.	
Documentation	• Development of report chapters. • Developing HMI Implementation Documentation: Part 1	9
Technical	• HMI Implementation: Part 1	
Documentation	• Development of report chapters. • Developing HMI Implementation Documentation: Part 2	10
Technical	• HMI Implementation: Part 2	
Documentation	• Development of report chapters. • Developing HMI Implementation Documentation: Part 3	11
Technical	• HMI Implementation: Part 3	
Documentation	• Chapter development: • Presentation 2 • Report Delivery 2 • Video Delivery 1	12

Documentation	• Chapter development: • Development of PLC module implementation documentation: Part 1	13
Technical	• PLC module implementation: Part 1	
Documentation	• Chapter development: • Development of PLC module implementation documentation: Part 2	14
Technical	• PLC module implementation: Part 2	
Documentation	• Chapter development: • Development of PLC module implementation documentation: Part 3	15
Technical	• PLC module implementation: Part 3	
Documentation	• Final presentation. • Delivery of Final Report. • Delivery • Video Delivery 2	16

IX. Project progress

IX.1. Programming environment configuration

IX.1.1. Objectives

- Recognize the TIA Portal V16 programming environment.
- Describe the PLC configuration process.
- Describe the compilation and simulation process.

IX.1.2. Theoretical support

IX.1.2.1 SIEMENS S7-300 PLC

The S7-300 CPU is a stand-alone, compact and robust device that is made up of:

- A central processing unit CPU,
- A power source.
- Digital inputs and outputs.
- Analog inputs and outputs.

The CPU executes the program and stores the data regarding the process to be automated. The system controls via digital inputs or outputs (I / O). The inputs monitor the signals from field devices (eg sensors, switches, signal transmitters), while the outputs control actuators (pumps, motors or other process devices). S7-300 CPUs have a built-in power supply capable of supplying the CPU, expansion modules, and other loads that require DC 24 V.

The S7-300 CPU supplies the 5 V direct current required for the expansion modules in the system. The LEDs indicate the operating mode of the CPU (RUN or STOP), the state of the physical inputs and outputs, as well as any detected system faults (SF).

Using expansion modules, additional digital or analog inputs and outputs (I / O) can be added to the CPU.

Communication modules improve communication performance.

An EEPROM plug-in cartridge (optional) is used to store the entire CPU program, as well as user data areas and configuration data and transfer it from one CPU to another, or send it by post to another part of the world.

The S7-300 also supports an optional battery cartridge that extends the time that RAM can be backed up after a power failure. The battery cartridge is activated only when the high-performance capacitor is discharged. Some CPUs have a built-in real time clock, while others may have an (optional) real time clock cartridge. Its purpose is to count hours of operation, label messages with date and time (accurate to the second, even in leap years), among other applications. [6]

IX.1.2.2 Programming languages for PLC according to IEC.61131-3

The IEC 61131-3 standard defines five programming languages. Although the functionality and structure of these languages is very different, they are treated as a single language family by IEC 61131-3, with overlapping structure elements (declaration of variables, organizational parts such as functions and function blocks, etc.) and configuration items.

Languages can be mixed in any way within a PLC project. The unification and standardization of these five languages represents a compromise of historical, regional and specific requirements of each sector. Future expansion (such as the function block principle or the Structured Text language) is planned; In addition, the necessary information on technological details (type of data, etc.) has been incorporated.

IX.1.2.2.1 Language of ladder diagram (Ladder Diagram) - LD

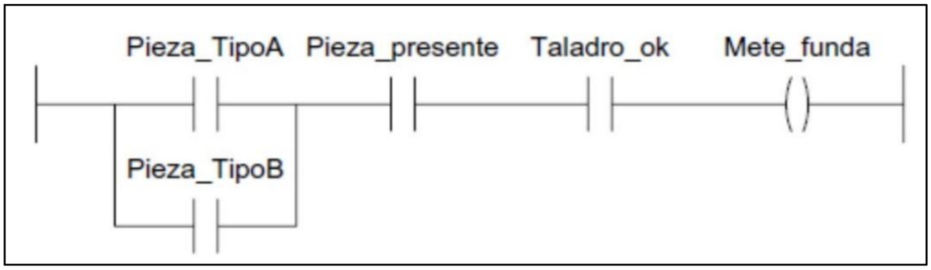


Figure 5: Example of LD language

Ladder Diagram is a graphical programming language derived from directly wired relay control circuit diagrams. The ladder diagram contains power lines to the right and left of the diagram; The terminals are connected to these lines, which are made up of contacts (normally open and normally closed) and coil elements, see example in Figure 5.

IX.1.2.2.2 Function block diagram language - FBD

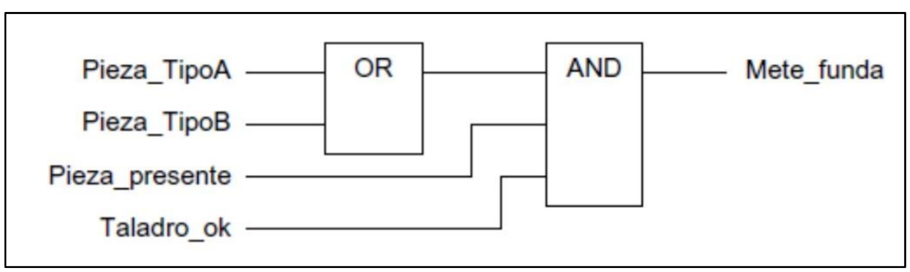


Figure 6: Example of FBD language

In the function block diagram, functions and function blocks are graphically represented and interconnected in networks. The function block diagram has its origin in the logic diagram that is used in the design of electronic circuits, example in Figure 6.

IX.1.2.2.3 Language of instruction lists (Stament List or Instruction List) - IL

The instruction list is an Assembler-type textual language, characterized by a simple machine model (processor with only one register). The list of instructions is formulated from control instructions consisting of an operator and an operand, as an example we have Figure 7.

```
LD   Pieza_TipoA
OR   Pieza_TipoB
AND  Pieza_presente
AND  Taladro_ok
ST   Mete_funda
```

Figure 7: IL language example

In terms of language philosophy, the ladder diagram, function block diagram and instruction list have been defined as they are used in current PLC technology. However, they are limited to the basic functions as far as their elements are concerned. This essentially sets them apart from the dialects that companies use today. The competitiveness of these languages is maintained due to the use of blocks and function blocks.

IX.1.2.2.4 Structured text language (Structure Text) - ST

Structured text is a high-level language based on Pascal, consisting of expressions and instructions. The instructions can be defined mainly as: Selection instructions, such as IF ... THEN ... ELSE, etc., Repeating instructions such as FOR, WHILE etc. and function block calls.

```
Mete_funda := (Pieza_TipoA OR Pieza_TipoB) AND Pieza_presente AND Taladro_OK;
```

Figure 8: ST language example

The structured text shown in Figure 8 allows the formulation of numerous applications, beyond pure function technology, such as algorithms (higher level regulation algorithms, etc.) and data management (data analysis, data processing complex data structures, etc.)

IX.1.2.3 Sequential Function Chart - SFC

The sequential function chart (almost identical to the French GRAFCET) (see Figure 9) is a language resource for structuring sequence-oriented control programs.

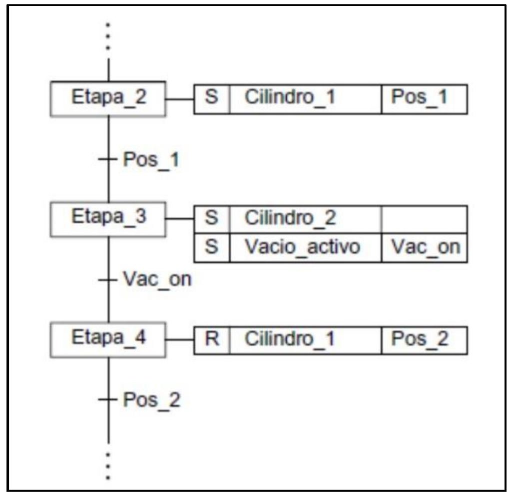


Figure 9: SF language example

The elements of the sequential function chart are alternate and parallel stages, transitions, and leads. Each stage represents a process state of a control program, which is active or inactive. One stage consists of actions that, like transitions, are formulated in the IEC 1131-3 languages. The actions themselves may again contain sequential structures. This feature allows the hierarchical structure of a control program. Therefore, the sequential function chart is an excellent tool for designing and structuring control programs.

IX.1.3. Progress analysis

The following describes the configuration process for the S7-300.

- **IP Address Configuration**

➢ ***Step 1:*** Left click on the RJ45 port.

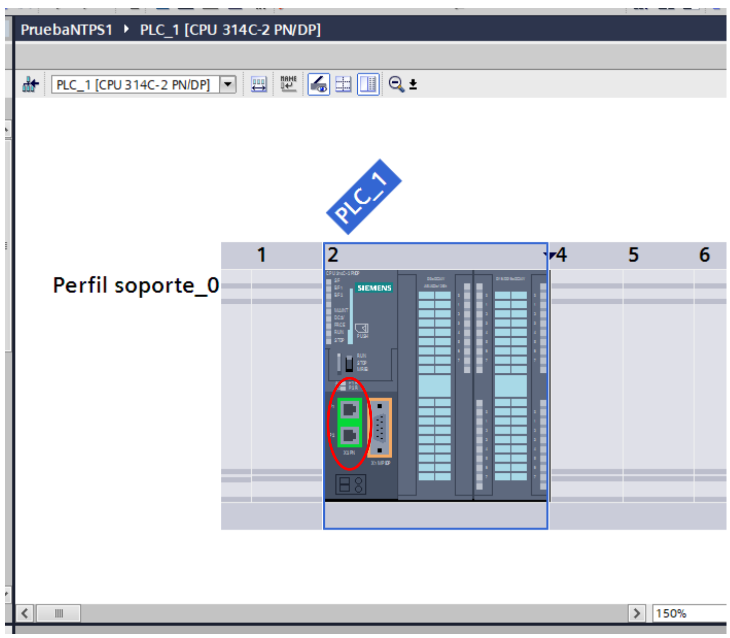


Fig. 10: **Step 1**

➢ ***Step 2:*** General: Internet addresses: IP Protocol: Configure IP Address at your convenience.

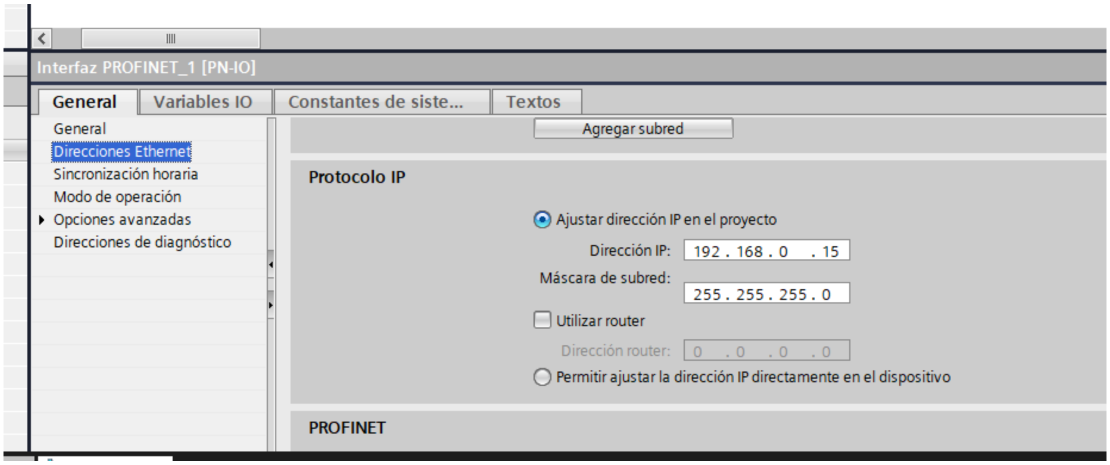


Fig. 11: **Step 2**

- **Programming, Compilation and simulation.**

➢ ***Step 1:*** Write program in Main

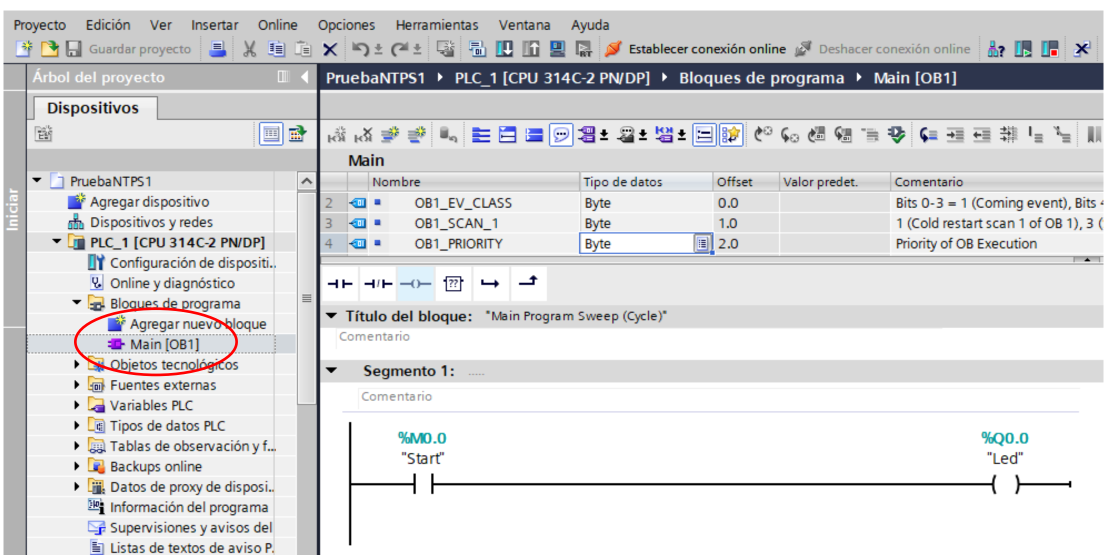


Fig. 12: Step 1

- ➢ ***Step 2:*** Proceed to compile (1) and Simulate (2).

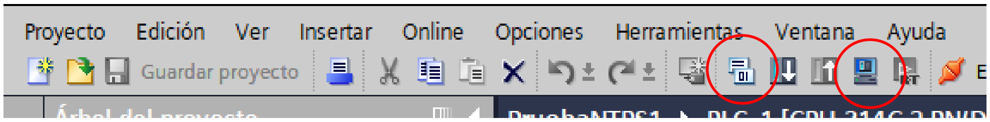


Fig. 13: Step 2

- ➢ ***Step 3:*** Activate RUN-P mode: Activate Marks (1): Activate outputs (2).

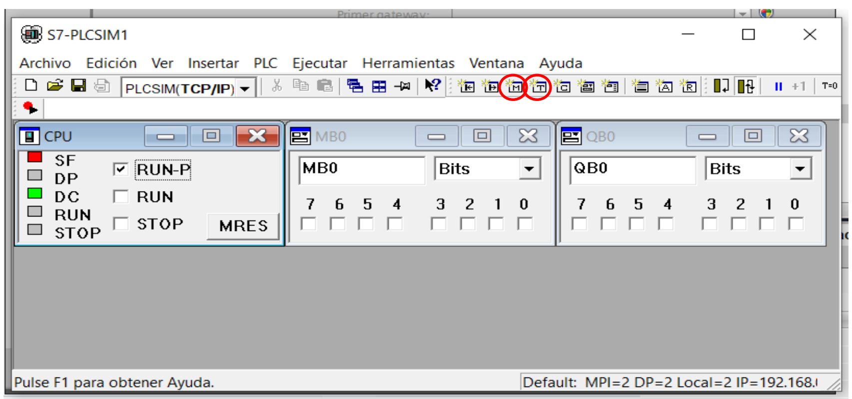


Fig. 14: Step 3

- ➢ ***Step 4:*** Select, load (hardware and software) and check simulation

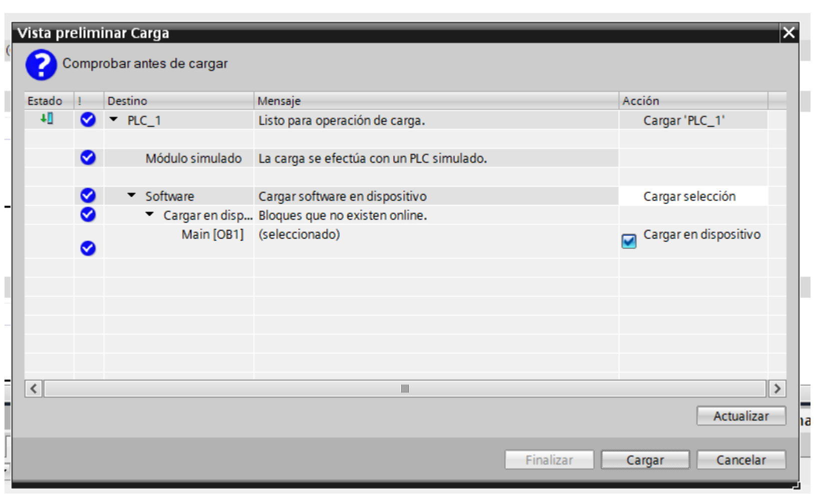


Fig. 15: Step 4

IX.1.4. Results, measurements and validation.

The figure shows the correct operation of the simulation of the test program developed.

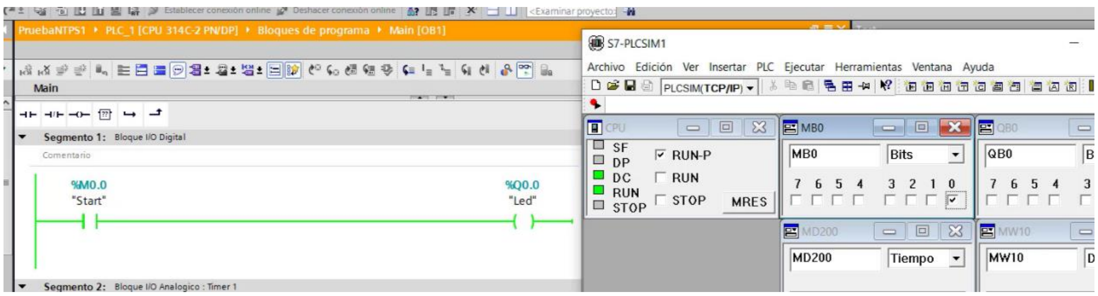


Fig. 16: Validation

IX.1.5. Problems

- There were no problems when making progress progress.

IX.1.6. Comments

- It should be taken into account that not all the models available in the TIA platform are capable of being simulated, it is a factor to take into account if a comparison is to be made between a simulated plant and a real plant.

IX.2. NetToPLCsim configuration (v 14.2.4.0)

IX.2.1. Objectives

- Recognize the NetToPLCsim environment.
- Describe the NetToPLCsim configuration process.

IX.2.2. Theoretical support

IX.2.2.1 NetToPLCsim

NetToPLCsim allows you to access the PLC simulator, S7-Plcsim from your network through TCP / IP communication (Iso-On-TCP), using the network interface of the PC on which the simulation is running. It is useful for testing a client application (SCADA, HMI, etc.) together with S7-Plcsim, without a real PLC.

NetToPLCsim is free software, you can redistribute and / or modify it under the terms of the GNU Lesser General Public License published by the Free Software Foundation, either version 3 of the License or (at your option) any later version. [7]

IX.2.2.2 TCP / IP

Under the acronym TCP / IP (Transfer Control Protocol / Internet Protocol) a package of data communication protocols is grouped. The package takes this name from two of the protocols that make it up, TCP, or Transfer Control Protocol, and IP, or Internet Protocol, two of the most important protocols that we can find in said package. Taking this into account, from now on we will refer to said packet as the TCP / IP protocols, in the plural. [9]

IX.2.2.2.1 Features

TCP / IP protocols have the following characteristics:

- They are free and open protocol standards. Its development and modifications are made by consensus, not at the will of a specific manufacturer. Anyone can develop products that meet your specifications.
- Independence at the software and hardware level Their wide use makes them especially suitable for interconnecting equipment from different manufacturers, not only to the Internet but also to form local networks. The independence of the hardware allows us to integrate several types of networks (Ethernet, Token Ring, X.25 ...) into a single one.

- They provide a common addressing scheme that enables one TCP / IP device to locate any other device anywhere on the network.
- They are high-level standardized protocols that support user services and are widely available and consistent.

IX.2.2.3 TCP / IP ports

The IP address is used to uniquely identify a computer on the network while the port number specifies the application to which the data is directed. Thus, when the equipment receives information that is directed to a port, the data is sent to the related application. If it is a request sent to the application, the application is called a server application. If it is a response, then we are talking about a client application. There are thousands of ports (encoded in 16 bits, which means that there are 65536 possibilities). That is why the IANA (Internet Assigned Numbers Authority) developed a standard application to help with network configurations.

Ports 0 to 1023 are the "known ports" or reserved. Generally speaking, they are reserved for system processes or programs run by privileged users. However, a network administrator can connect services with ports of his choice.

- Ports 1024 through 49151 are the "registered ports".
- Ports 49152 through 65535 are the "dynamic and / or private ports".

Some of the most commonly used known ports are listed in Table 4 below:

port	Service or application
twenty-one	FTP (File Transfer Protocol)
2. 3	Telnet (TELecommunication NETwork Terminal Client)
25	SMTP (Simple Mail Transfer Protocol Protocol Simple of Mail Transfer)
53	DNS (Domain Name System Domain Name System)
80	HTTP (HyperText Transfer Protocol) hypertext)
110	POP3 (Post Office Protocol Mail Protocol version 3)

Table 1: Designation of TCP / IP ports reserved for applications

IX 2.3. Progress analysis

Next, the NetToPLCSim configuration process is described.

- ***Step 1:*** Left click on the icon: Run as administrator: Warning: Yes

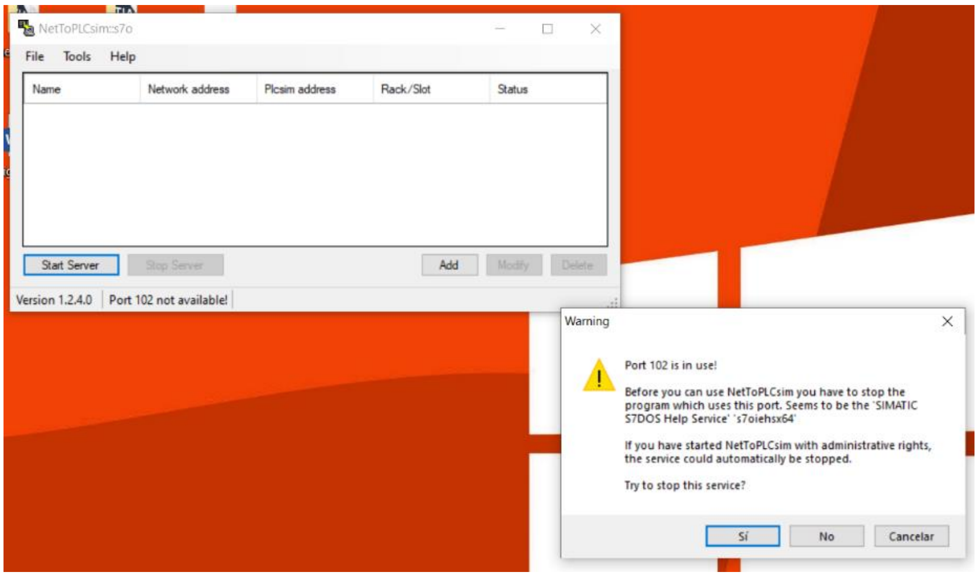


Fig. 17: Step 1

- ***Step 2:*** Wait until step 5 and OK.

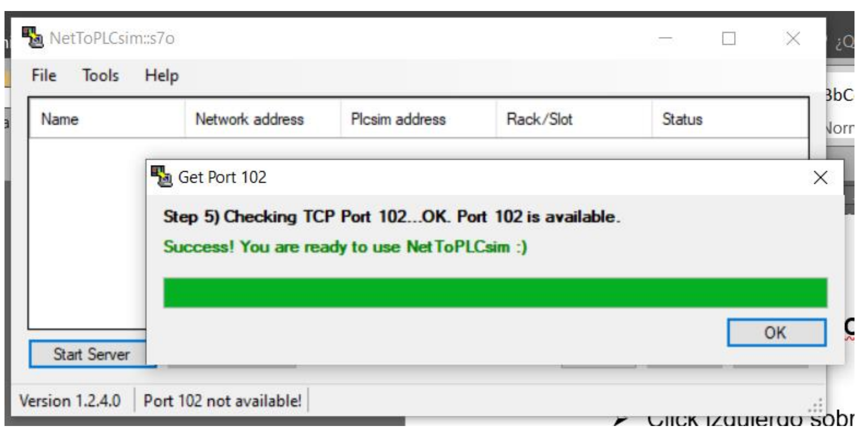


Fig. 18: Step 2

- ***Step 3:*** Press Add to create server.

Fig. 19: Step 3

- ***Step 4:*** Network IP addresses: xxx.xxx.xxx.xxx [Siemens PLCSIM Virtual Ethernet Adapter].

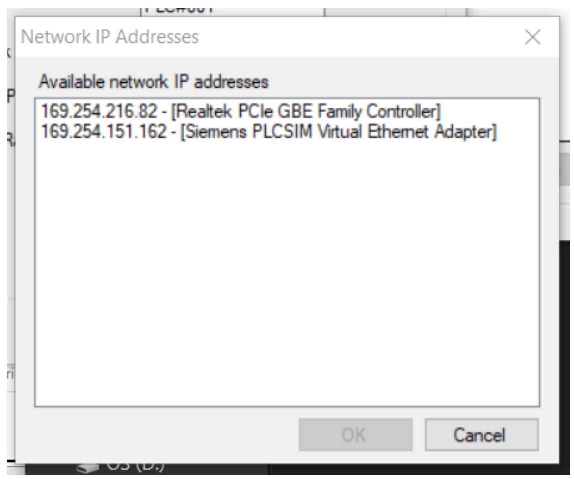


Fig. 20: Step 4

- ***Step 5:*** Network reachable Plcsim PLCs: xxx.xxx.x.xx [S7-300 CP: The previously configured ip]

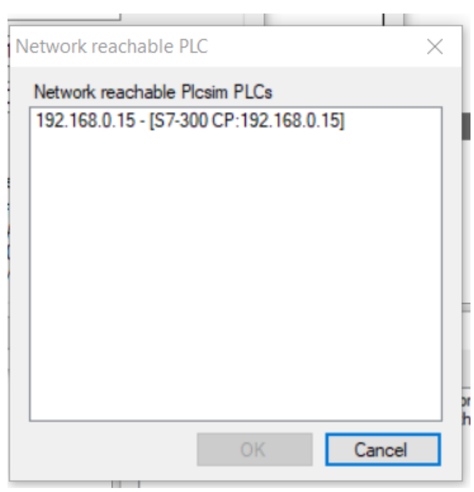


Fig. 21: Step 5

➢ ***Step 6:*** Plcsim Rack / Slot: 0/2 [For S7-300], Ok.

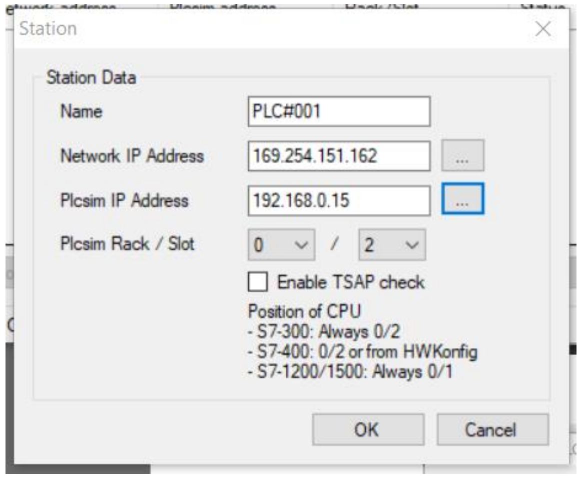


Fig. 22: Step 6

➢ ***Step 7:*** Start Server, and the server will be operating.

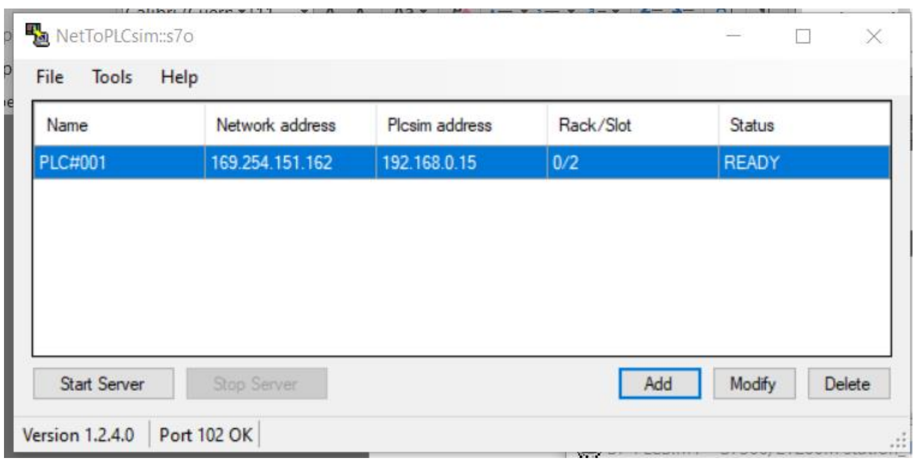


Fig. 23: Step 7

IX.2.4. Results, measurements and validation.

Figure 18 shows the correct operating status, in the following advances the objective of the connection and the verification of the connection between the softwares to be used is proposed.

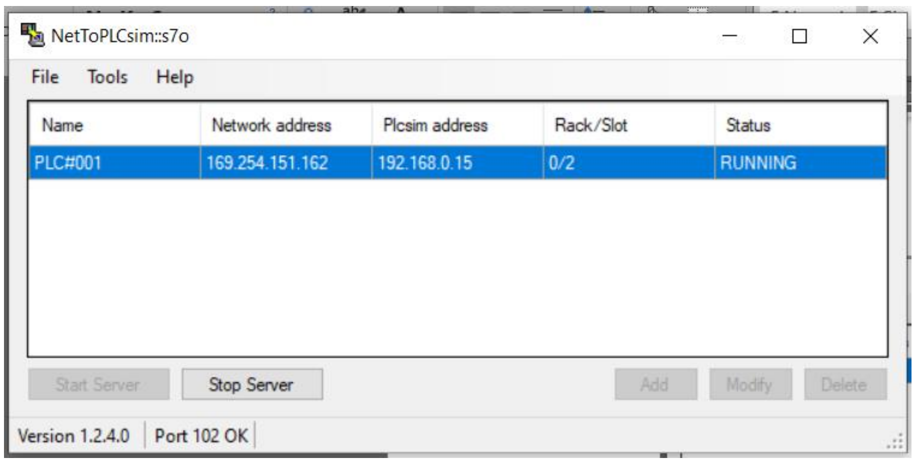


Fig. 24: Correct operating status.

IX.2.5. Problems

- There were no problems when making progress progress.

IX.2.6. Comments

- Being free software, and not belonging to SIEMENS products, it is susceptible to unforeseen errors when simulating certain devices, there were no problems in the updated version, but it is recommended to review the comment forum of the installer housing.

IX.3. NI OPC Servers 2016 configuration.

IX.3.1. Objectives

- Recognize the NI-OPC-Server 2016 configuration environment.
- Describe the NI-OPC-Server 2016 configuration process.

IX.3.2. Theoretical support

IX.3.2.1 OPC (OLE for Process Control - Object Linking and Embedding)

IX.3.2.1.1. Definition

An OPC server is a software application (driver) that conforms to one or more specifications defined by the OPC Foundation. The OPC Server acts as an interface communicating on the one hand with one or more data sources using their native protocols (typically PLCs, DCSs, scales, I / O modules, controllers, etc.) and on the other hand with OPC Clients (typically SCADAs , HMIs, report generators, graph generators, calculation applications, etc.). In an OPC Client / OPC Server architecture, the OPC Server is the slave while the OPC Client is the master. Communications between the OPC Client and the OPC Server are bi-directional, which means that the Clients can read and write to the devices through the OPC Server. [9]

IX.3.2.1.2 Types

There are four types of OPC servers defined by the OPC Foundation, and they are as follows:

- **OPC DA server:** Based on Spezifikationsbasis: OPC Data Access - specially designed for real-time data transmission.
- **OPC HDA Server:** Based on the Historic Data Access specification that provides the OPC HDA Client with historical data.
- **OPC A&E Server:** Based on the Alarms and Events specification - transfers Alarms and Events from the device to the OPC A&E Client.
- **OPC UA server:** Based on the Unified Architecture specification - based on the newest and most advanced set from the OPC Foundation, it allows OPC Servers to work with any type of data.

Collectively, the first three types of OPC Servers are known as "Classic" OPC Servers to distinguish them from OPC UA which will become the basis for future OPC architectures.

IX.3.2.1.3 Structure

1. OPC Client / OPC Server Communications (OPC DA Server, OPC HDA Server, OPC A&E Server)

Classic OPC Servers use the Microsoft Windows COM / DCOM infrastructure for data exchange. Which means that those OPC Servers must be installed under the Microsoft Windows Operating System. An OPC Server can support communication with multiple OPC Clients simultaneously.

2. OPC Server - Data Translation / Mapping

The main function of an OPC Server is to translate native data from the data source into an OPC format that is compatible with one or more OPC specifications mentioned above (example: OPC DA for real-time data). The OPC Foundation specifications only define the OPC portion of the OPC Server communications, so the efficiency and quality of translation from native to OPC and from OPC to native protocol depends entirely on the developer's implementation of the OPC Server.

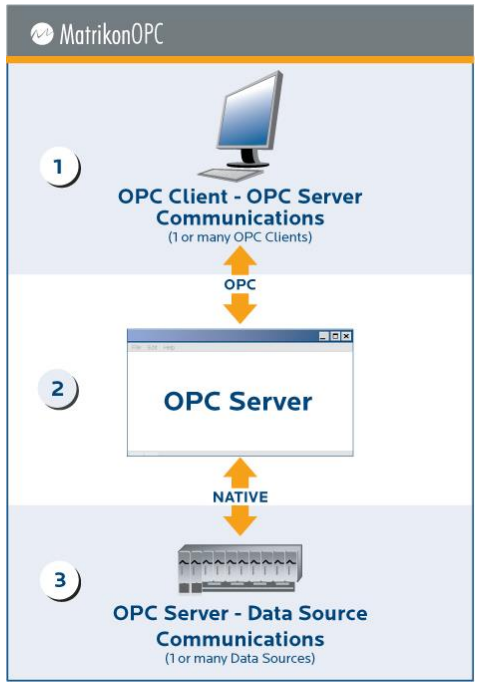


Figure 25. OPC structure

3. OPC Server - Data Source Communication

OPC Servers communicate natively with data sources, for example: devices, drivers and applications. The OPC Foundation specifications do not specify how the OPC Server should communicate with the data source because there are a wide variety of data sources available on the market. Every PLC, DCS, controller, etc. It has its own communication protocol or API that in turn allows the use of any number of physical connections (serial RS485 / 232, Ethernet, Wireless, proprietary networks, etc.).

Two common examples of how OPC Servers communicate with the Data Source are:

- Through an application programming interface (API) to a custom driver written specifically for the Data Source.

- Through a protocol that may or may not be proprietary, or based on an open standard (for example, using the Modbus protocol).

IX.3.3. Progress analysis

➢ ***Step 1:*** Click to add a channel: New Channel

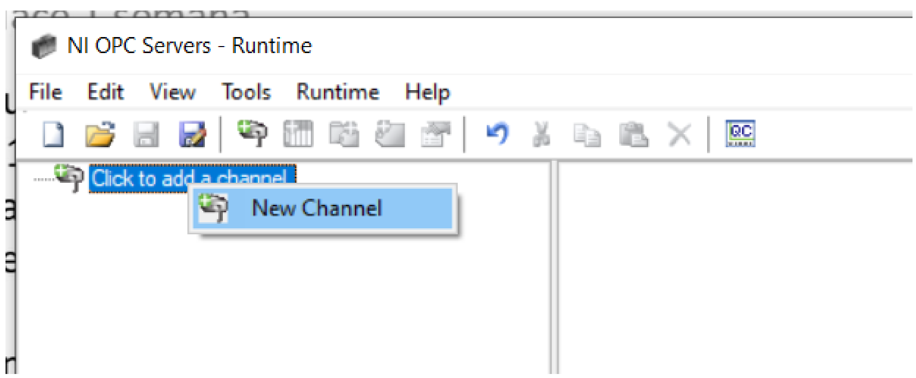


Figure 26. Step 1

- ***Step 2:*** Channel name: Write name to identify our channel> Next.

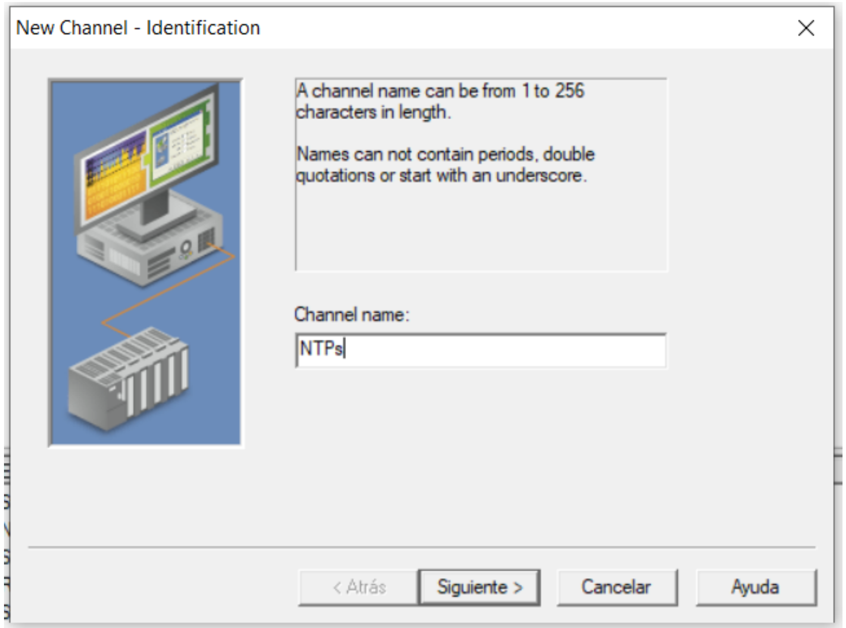


Figure 27: Step 2

- ***Step 3:*** Device driver: Siemens TCP / IP Ethernet> Next.

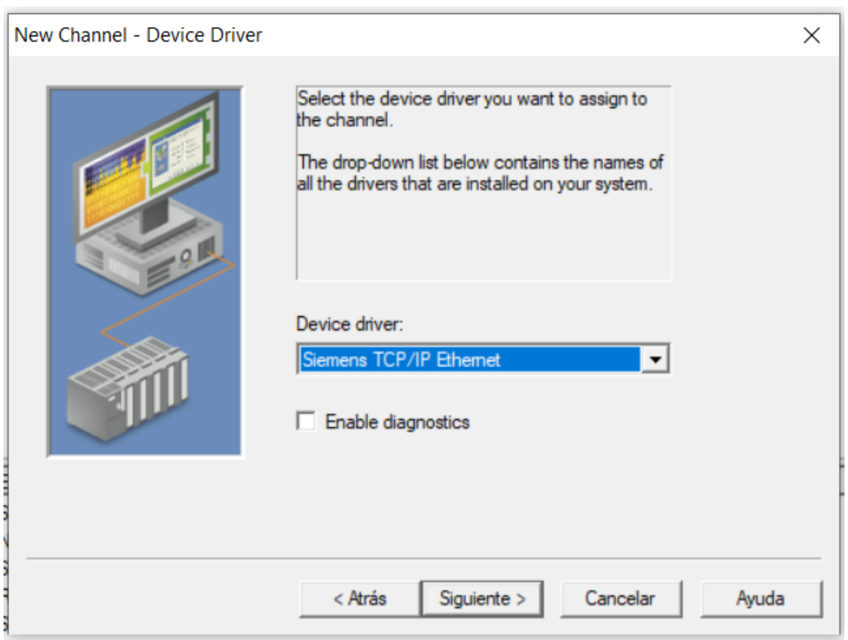


Figure 28: Step 3

➢ ***Step 4:*** Network Adapter: Default> Next to Finish.

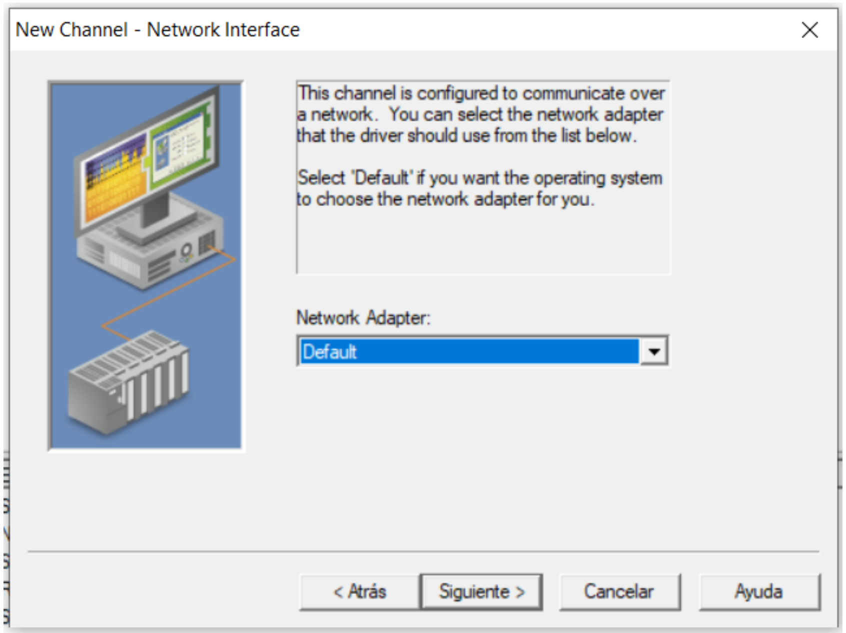


Figure 29: Step 4

➢ ***Step 5:*** Click to add a device.

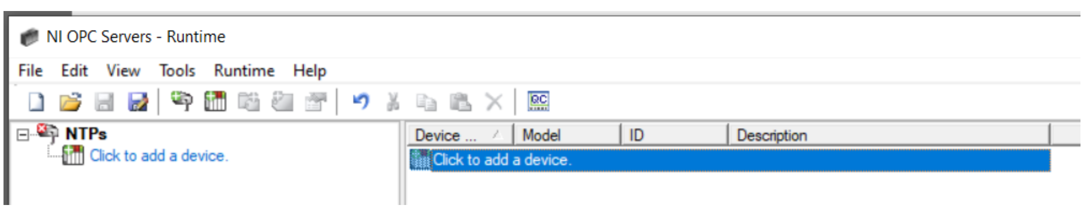


Figure 30: Step 5

- ***Step 6:*** Device name: S7-300 (Choose name to identify)> Next

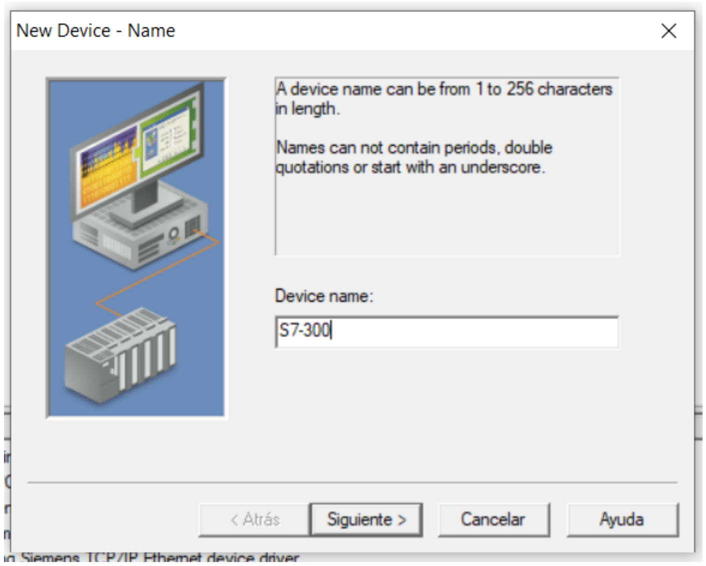
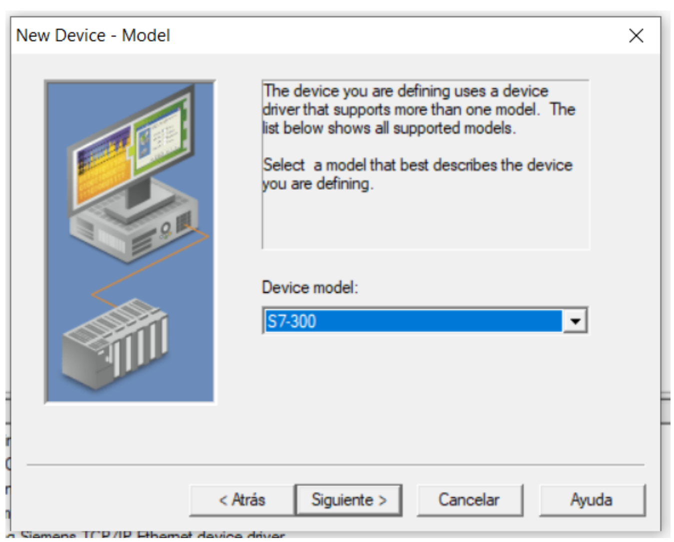


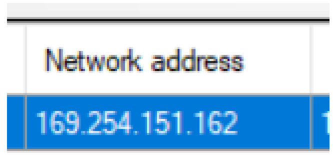
Figure 31: Step 6

- ***Step 7:*** S7-300 (Choose the model we are simulating)> Next

Figure 32: Step 7



➢ ***Step 8:*** xxx.xxx.xxx.xxx (Write NetToPLCsim Network addres)> Next, until finished.



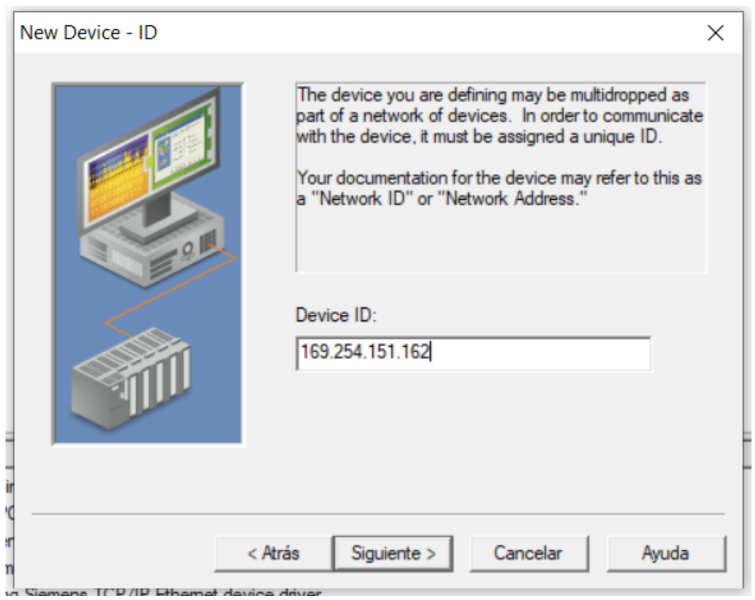


Figure 33: Step 8

➢ ***Step 9:*** Select <Device Name> (1).
➢ ***Step 10:*** Clock to add a static tag.

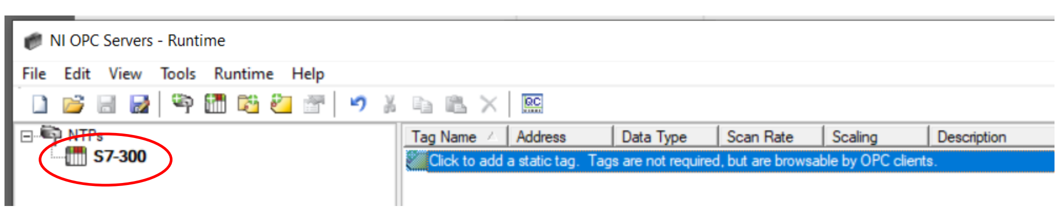


Figure 34: Step 9 and Step 10

➢ ***Step 11:*** Configure variable, according to characteristics> OK.

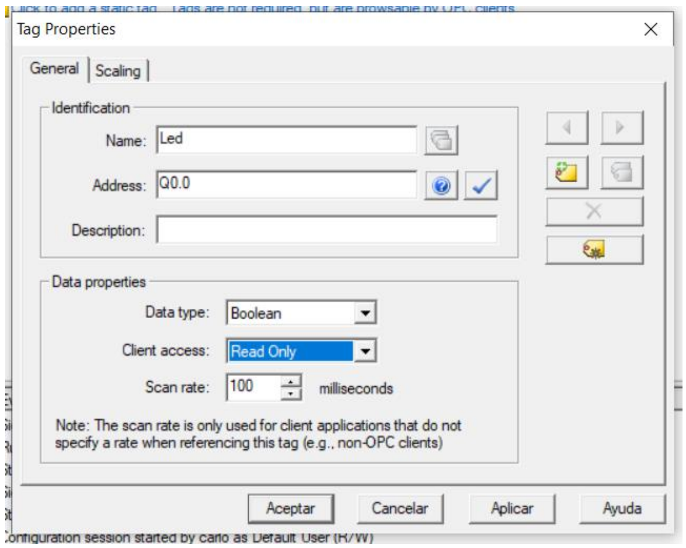


Figure 35: Step 11

- ***Step 12:*** Check the creation of the tag.
- ***Step 13:*** Click on OPC Quick Client (1).

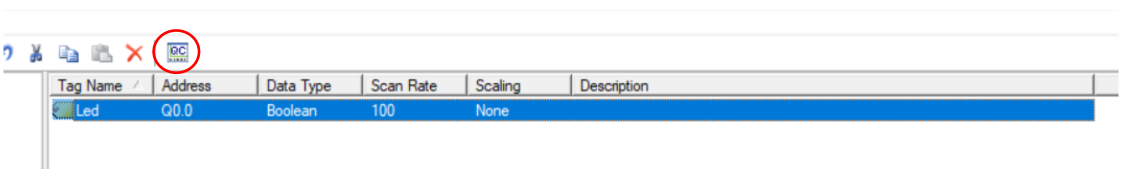


Figure 36: Step 11 and Step 12

IX.3.4. Results, measurements and validation.

- Figure 18 shows the correct operating status, it is checked with the Good status in the Quality item and the changes in the values in Valué, according to the changes made in the simulator.

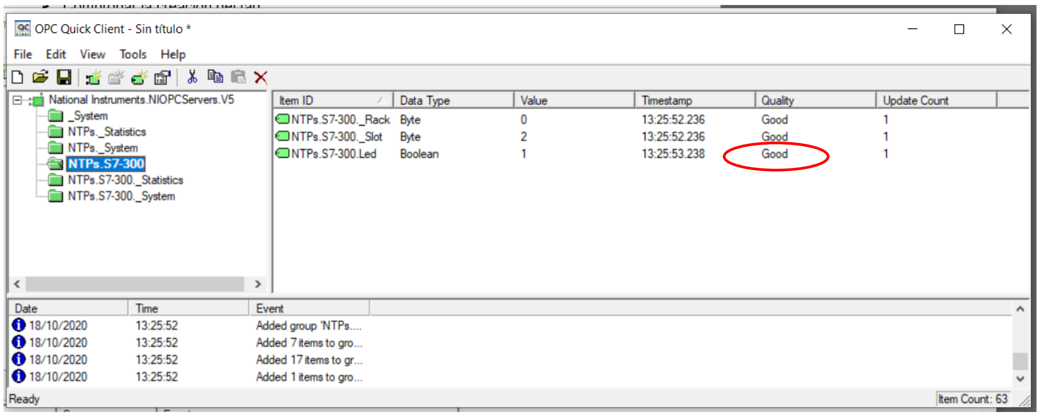


Figure 37: Validation

IX.3.5. Problems

- In the case of the NI OPC Server 2013, you will only have drivers for drivers manufactured before 2013. If you want drivers for more recent drivers, for example S7-1500, you will need to update to version 2016 or an updated OPC from another brand. [10]

IX.3.6. Comments

- For the test on the S7-1500 controller, the NI OPC Server 2016 version was used, in both cases the configuration is the same.

IX.4. NI LabVIEW 2015

IX.4.1. Objectives

- Recognize the LabVIEW programming environment.
- Describe the LabVIEW setup process.

IX.4.2. Theoretical support

IX.4.2.1. Definition

LabVIEW is a general-purpose application development program that National Instruments (NI) has created to facilitate the programming of virtual instruments (VI's). LabVIEW is in charge of managing computer resources through a simple, fast and efficient environment.

In this way, development times are greatly reduced when making programs. The programming language is graphical.

This program was created by NI 1976 to run on MAC machines, it was first released in 1986. It is now available for Windows, UNIX, and MAC platforms.

The programs made with LabVIEW are called VI (Virtual Instrument), which gives a perspective of their use in origin: instrument control. Among its objectives are to reduce the development time of applications of all kinds (not only in Test, Control and Design areas) and to allow non-expert programmers to enter computing.

In addition to distributing only software, National Instruments also manufactures hardware to associate with its software, develops hardware such as data acquisition cards, PACs, Vision (Cameras, Artificial Intelligence), and couples hardware from other companies.

IX.4.2.2. Characteristics

Its main feature is ease of use, valid for professional programmers and people with little programming knowledge can make relatively complex programs, impossible with traditional languages.

For lovers of the complex, with LabView you can create programs of thousands of VI's (equivalent to millions of pages of text code) for complex applications, automation programs of tens of thousands of I / O points, etc. There are even good scheduling practices to optimize scheduling performance and quality.

The main features of LabVIEW are as follows:

- Graphic development environment; the text code that we are used to using disappears. This results in a more intuitive way of programming.
- Design of the graphical interface of the virtual instrument, using predesigned elements (numerical controls, graphs, etc.).
- Automatic management in the creation of execution threads.
- Conventional tools for debugging programs (VI's): step-by-step execution, breakpoints, data flow, etc.
- Modular programming.
- Visualization and handling of graphs with dynamic data.
- Acquisition and treatment of images.
- Motion control (combined even with all of the above).
- Real time strictly speaking.
- FPGA's programming for control or validation.
- Synchronization between devices.
- Communications interfaces:
    - Serial port (RS232, RS422, RS485)
    - Parallel port
    - GPIB protocol
    - PXI
    - VXI
    - TCP / IP, UDP, Data Socket
    - Irda (Infrared Port)
    - Bluetooth
    - USB

    - OPC
- Ability to interact with other languages and applications:
    - DLL: Dynamic function libraries
    - NET
    - ActiveX
    - MultiSim
    - Matlab (Math Script)
    - AutoCad, SolidWorks

Signal acquisition is the process of obtaining or generating information in an automated way from analog and digital measurement resources such as sensors and devices under test.

Data acquisition basically consists of capturing a physical signal and taking it to a computer, this means taking a set of measurable variables in physical form and converting them into electrical voltages, in such a way that they can be used or can be read on the PC.

It is necessary for the physical signal to go through a series of stages that allow the computer to be able to interpret the signal sent. Once the electrical signals have been transformed into digital signals within the memory of the PC, they can be processed with an application program suitable for the use that the user wishes.

In the same way that an electrical signal is taken and transformed into a digital one within the computer, a digital or binary signal can be taken and converted into an electrical one, in this way the PC can send signals to actuating devices.

IX.4.2.2. Stages of data acquisition

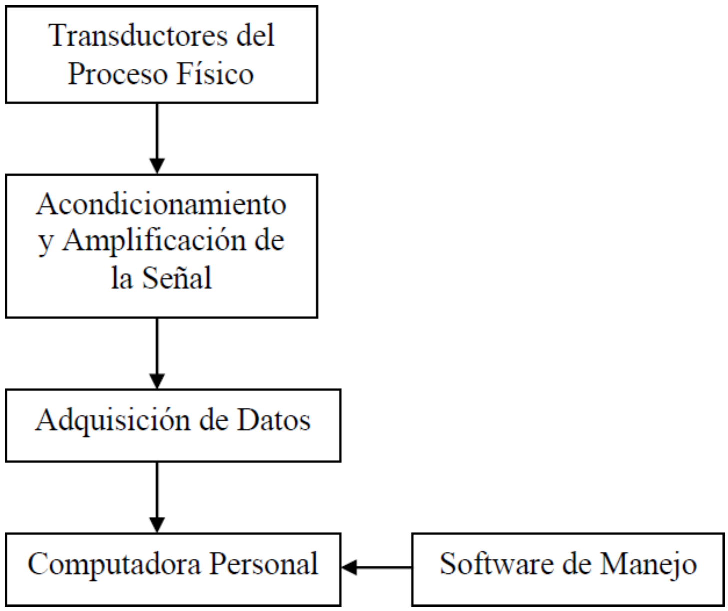


Figure 38. Stages of data acquisition

The physical signal goes through a series of stages in order to be read by the computer, these are:

Transducer stage: Transducers are devices that convert a physical signal (such as pressure, temperature, light) into electrical signals of voltage or current.

Transmission stage: It allows sending the output signals from one stage to another located in a remote location. For distances that are not excessive, it is common to use a 4-20 mA current loop for the transmission of signals or in other cases using either 0-5 V or 0-10 V voltage.

Conditioning stage: It contains electronic circuits responsible for transforming the sensing signals into new electrical variables, so that they are easier to deal with by the rest of the system stages, it involves noise filtering, stepping, adjustment to the A / D converter interval, etc.

Acquisition stage: It performs the transformation of analog information into a digital format, which makes possible further processing and storage through the use of a computer.

Processing stage: It takes place inside the computer, it consists of carrying out operations on the digital information obtained: decisions for the control of a system, detection of alarm situations, correction of measurements, storage and information reports.

IX.4.2.3. Signal classification

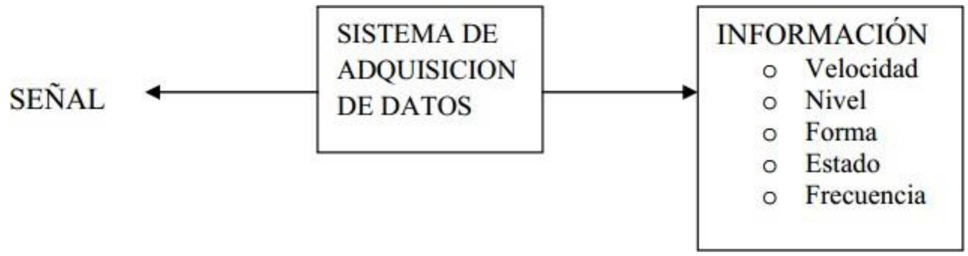


Figure 39: Information carried by a signal

A signal is a physical quantity that contains information both in magnitude and in time. These signals can be of different nature and therefore their physical units can be diverse. For the signal conditioner, the signal to be measured must be converted into an electrical signal, such as voltage or current using a transducer.

All analog signals are variable with time. However, to discuss signal measurement methods, a classification of the signals is made. A signal is classified as ANALOG or DIGITAL signal, according to the way the information is transported. A digital or binary signal has only two possible discrete levels, one high and the other low. An analog signal contains information on the continuous variation of the signal with respect to time.

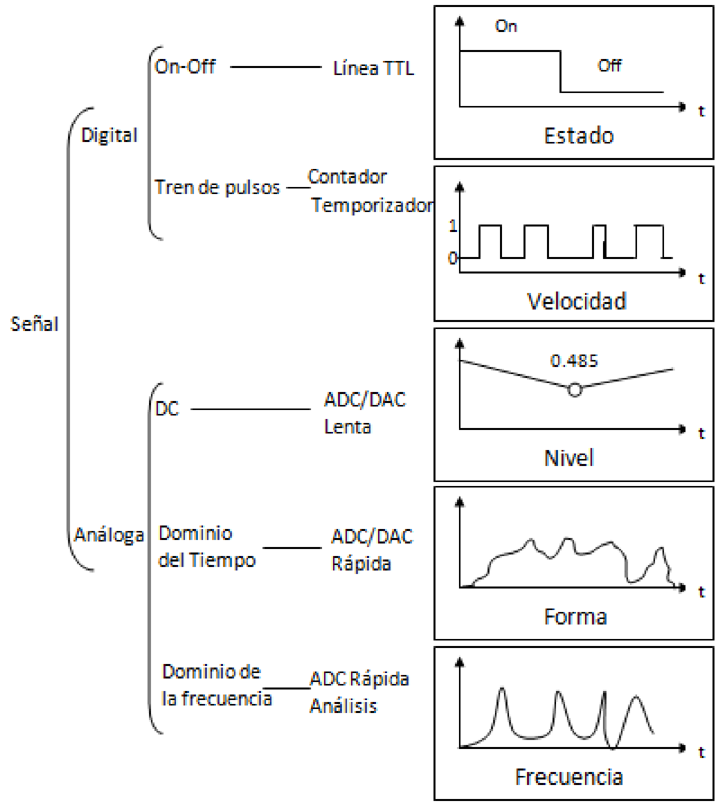


Figure. 40: Information carried by a signal

IX.4.2.3.1. ON-OFF signals

This signal carries the information in the immediate digital state of the signal. A digital status detector is used to measure that type of signal. The output of a switch or the output of a device with TTL logic is an example of the digital ON-OFF signal.

IX.4.2.3.2. Pulse train digital signal

This sign consists of a series of transitory states. The information is contained in the number of transient states that occurred, the speed at which the transient occurred, or the time between one or more transient states. The output signal of an electronic optocoupler mounted on the shaft of a motor is an example of the pulse train signal.

IX.4.2.3.3. Analog DC signal

Analog DC signals are static or have a slow variation. The most important characteristic of DC signals is the level or the amplitude, the precision in the level measurement is what matters more than the time or the speed at which the measurement was taken, that is why the instrument or Plug- in DAQ that measures the analog signal DC operates an analog digital converter ADC, changing the analog signal to a digital value, so that the computer can interpret it.

IX.4.2.4. Process control and monitoring

With this tool you can easily create user interfaces for virtual instrumentation without the need to develop programming code.

To specify the functions, it is only necessary to construct block diagrams. You have access to a controls palette from which you can choose numerical displays, meters, thermometers, tanks, graphs, etc., and include them in any of the control projects that are being designed.

It is based on a data flow programming model called G, which frees programmers from the rigidity of text-based architectures. It is also, according to NI, the only graphical programming system that has a compiler that generates optimized code, whose execution speed is comparable to the C language. The developments built are fully compatible with the VISA, GPIB, VXI and alliance standards. of VXI Plug & Play systems. To further facilitate the operation of this product, it has the inclusion of an assistant tool capable of automatically detecting any instrument connected to the computer, installing the appropriate drivers and facilitating communication with the instrument instantly.

Although it was originally created to build virtual instrumentation - oscilloscopes, function generators, voltmeters, etc.–, thanks to the wide availability of data acquisition cards and the ease of building applications in a graphical environment, the latest versions have been used widely to develop applications in process control. In addition, NI introduced from the version of LabView 6i, the combination of the traditional functions of the product combined with some tools for the Internet environment. This is the case of the LabView Player, an add-on that makes it easy to run applications over the network without having the full LabView product.

IX.4.2.5. Graphic interface

IX.4.2.5.1. VI front panel

The interactive user interface of a VI is called the Front Panel because it simulates the dashboard of a physical instrument.

The front panel can contain knobs, buttons, graphics and other controls and indicators, that is, it can be built according to the user's need. The controls simulate equipment input instruments and supply data to the VI block diagram - these can be buttons, push buttons, markers, and other input components.

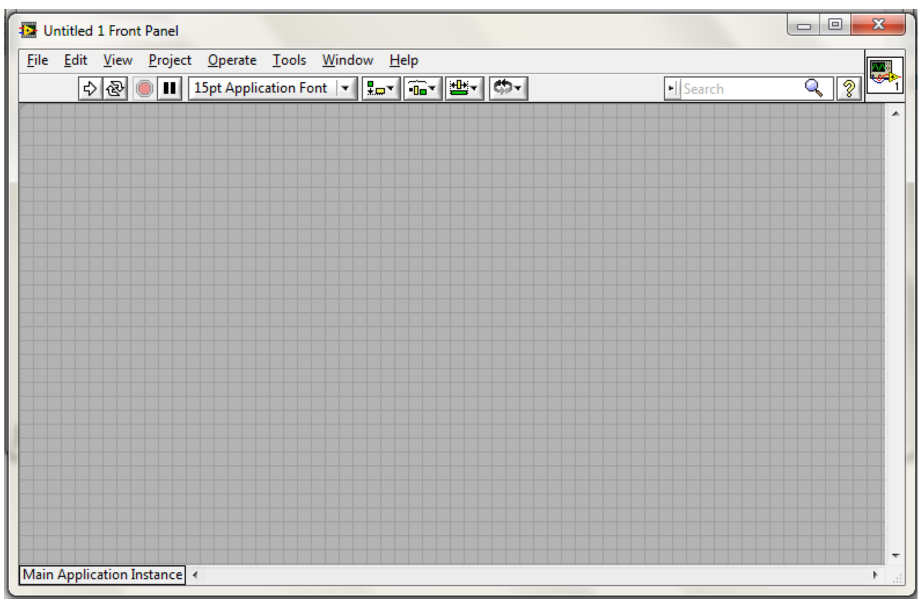


Figure 41: VI front panel

The indicators simulate instrument outputs and provide data that the block diagram acquires or generates, among others, these can be graphs, lights and other output devices, these interact with the VI terminals.

IX.4.2.5.2.Control palette

The control palette in LabVIEW provides the tools required to interface the VI with the user, on the front panel only.

To display the controls palette, you need to right-click on the front panel space in the same way to hide it.

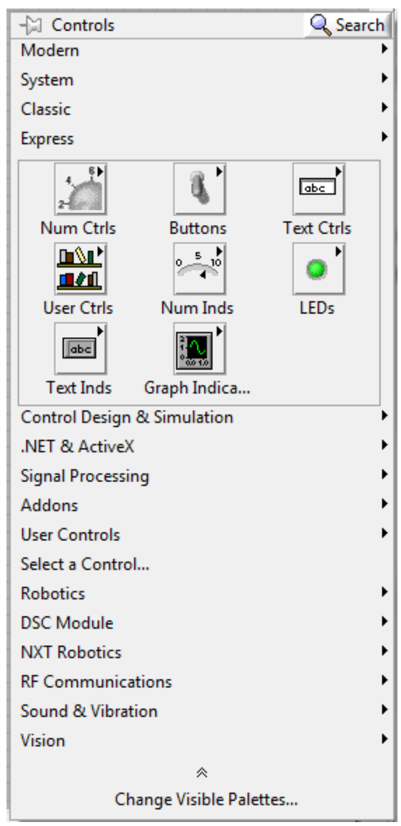


Figure 42: Control palette

IX.4.2.5.3. VI Block diagram

VIs receive instructions from a block diagram that unfolds in G. The block diagram is a graphical solution to a programming problem and is also the source code for VIs.

In the block diagram, the front panel objects appear as terminals, in addition the block diagram contains functions and structures incorporated in the LabVIEW libraries, the cables connect each of the nodes in the block diagram, including controls and indicators of terminal, functions and structures.

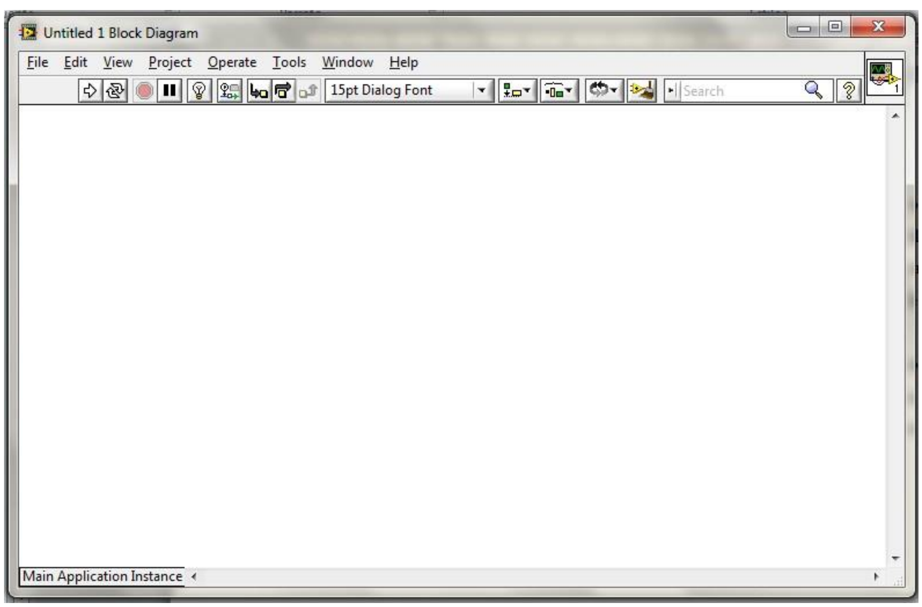


Figure 43: VI Block Diagram

VIs are hierarchical and modular. They can be used as main programs, or as subprograms within other programs. A VI within another VI is called a subVI. The VI icon and connector works as a graphical parameter list so that other VIs can pass data to a subVI.

IX.4.2.5.4.Features palette

The functions palette is used in the design of the block diagram, it also contains all the objects used in the implementation of the VI program, whether they are arithmetic functions, input / output signals, input / output data to file, acquisition of signals, timing of program execution.

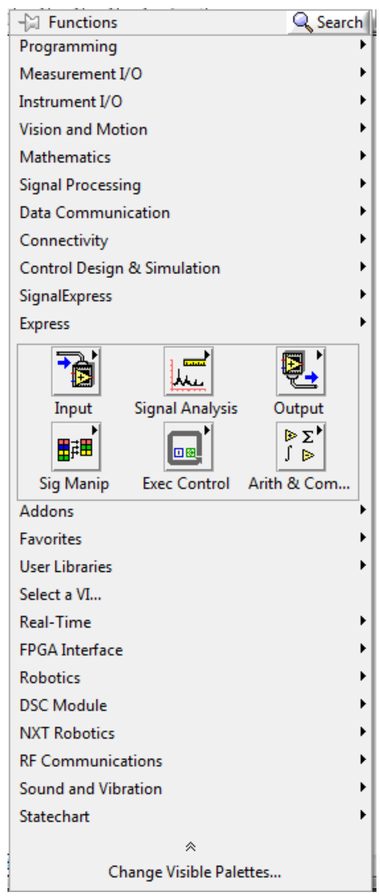


Figure 44: Features palette

IX.4.2.6. LabVIEW Control Design and Simulation Module

With the LabVIEW Control Design and Simulation Module, you can simulate dynamic systems, design sophisticated controllers, and implement your hardware control systems in real time. You can use both classical and state-space approaches to design controllers and calculators.

IX.4.2.7. LabVIEW Datalogging and Supervisory Control Module

The LabVIEW Datalogging and Supervisory Control (DSC) Module extends the benefits of graphical programming to developing Supervisory Control and Data Acquisition (SCADA) or multi-channel data logging applications. Use the tools to communicate conventional programmable logic controllers (PLCs) and programmable automation controllers (PACs), log data to databases, manage alarms and events, and create human-machine interfaces (HMIs).

IX.4.2.8. LabVIEW Real-Time Module

The NI LabVIEW Real-Time Module is a complete solution for creating autonomous and reliable embedded systems with a graphical programming approach. As a complement to the LabVIEW development environment, the module helps you develop and debug graphical applications that you can download and run on embedded hardware devices such as NI CompactRIO, CompactDAQ, PXI, vision systems, or third-party PCs.

IX.4.3. Progress analysis

After recognition of the software, we will proceed to create a project that will serve as the basis for creating the final project.

- ***Step 1:*** Create Project> Blank Project.

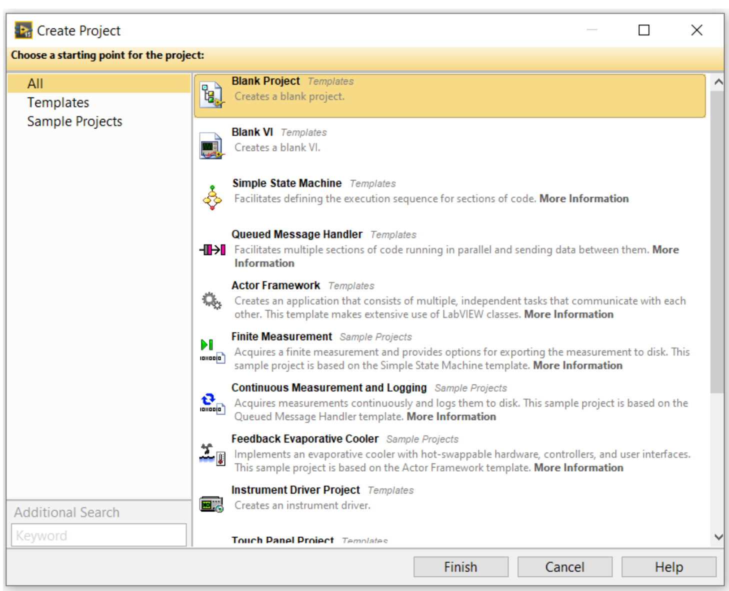


Figure 45: Step 1

➢ ***Step 2:*** My Computer> New> I / O Server.

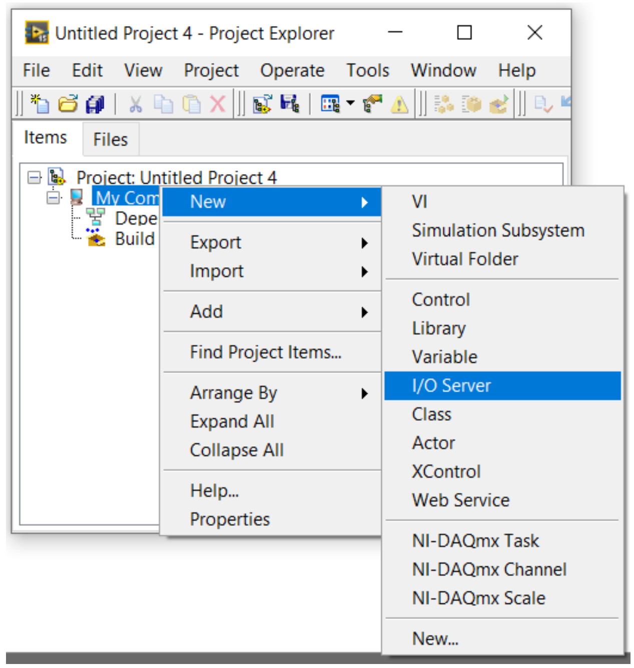


Figure 46: Step 2

➢ ***Step 3:*** OPC Client> Continue…

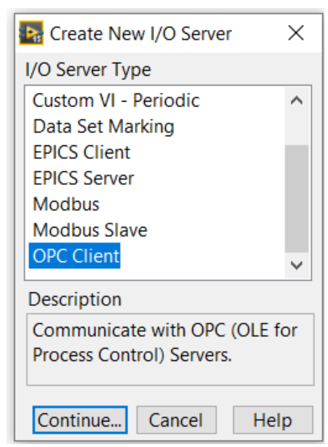


Figure 47: Step 3

- **_Step 4:_** Configure Update rate: 100 ms (according to project).
- **_Step 5:_**Registered OPC servers: National Instruments.NIOPCServers.V5> OK.

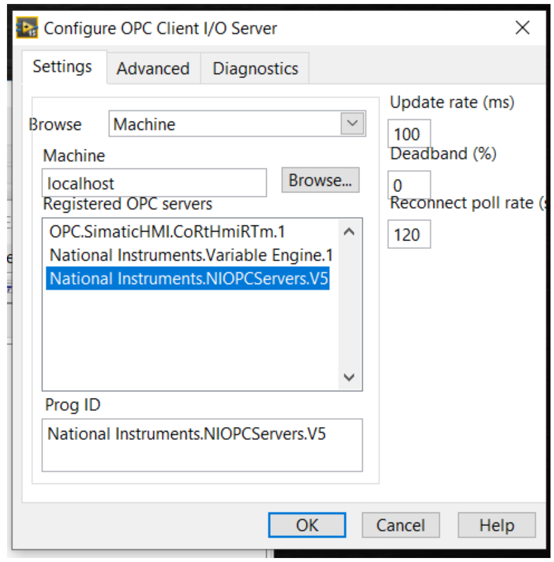


Figure 48: Step 4 and Step 5

- **_Step 6:_** DCOM Configuration Recommendation> OK.

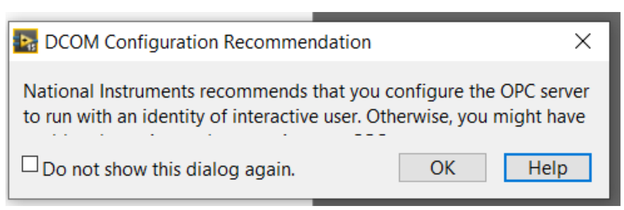


Figure 49: Step 6

➢ ***Step 7:***Project> My Computer> Untitled Library> OPC1> Create Bound Variables

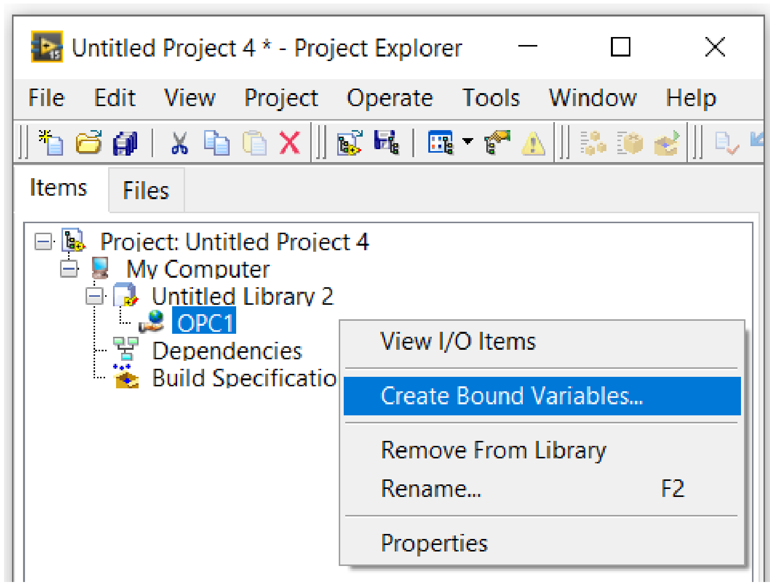


Figure 50: Step 7

➢ ***Step 8:***Project> My Computer> Untitled Library> OPC1> NTPs (Channel Name)> S7-300 (Device Name)> Select the variables.

➢ ***Step 9:***Add >>> OK.

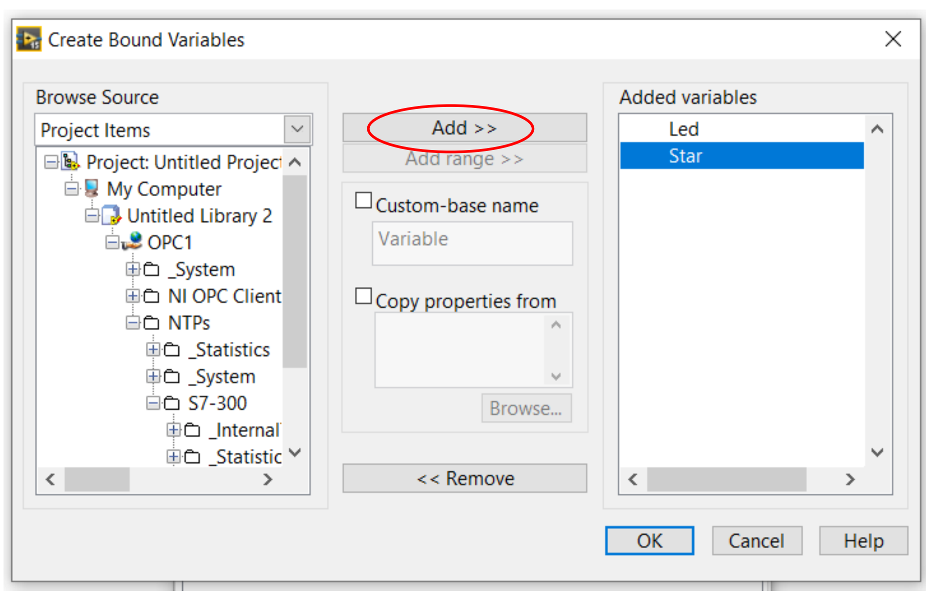


Figure 51: Step 7 and Step 8

- ***Step 10:***Multiple Variable Editor> Done.

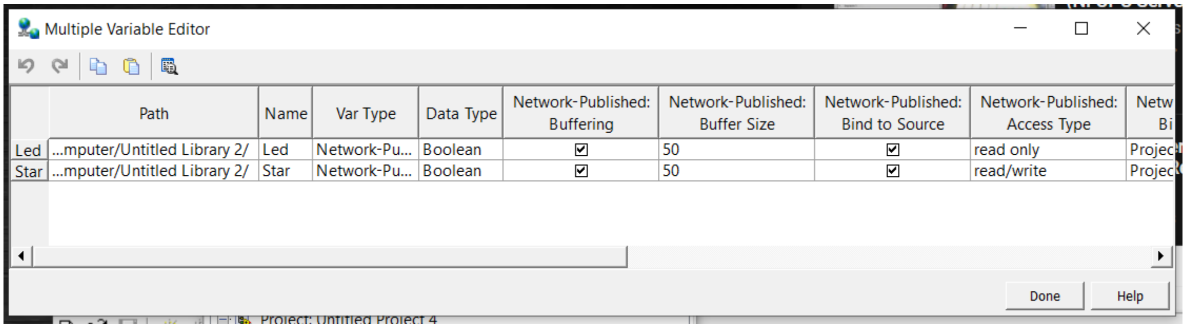


Figure 52: Step 10

- ***Step 11:***Project> My Computer> My Computer> New> VI.

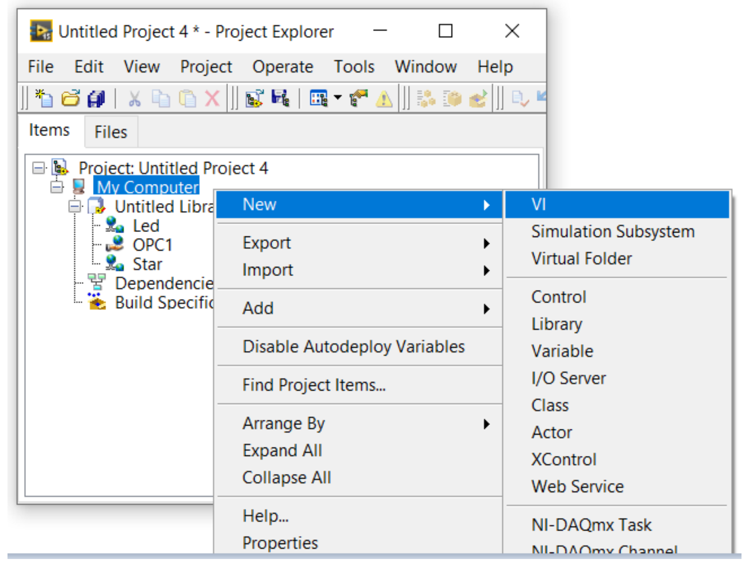


Figure 53: Step 11

- ***Step 12:***Select the variable to Block Diagram

Figure 54: Step 12

➢ ***Step 13:*** Control> Left Click> Access Mode> Read / Write

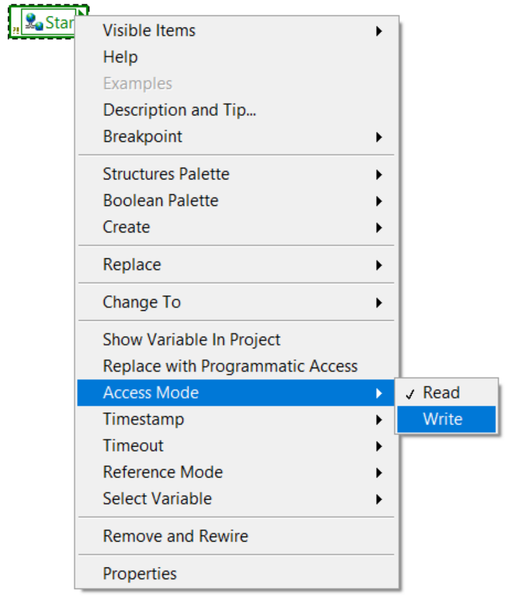


Figure 55: Step 13

➢ ***Step 14:*** Connect the variables.

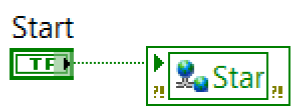


Figure 56: Step 14

IX.4.4. Results, measurements and validation.

After the connection of the first element, it is ready to do the same for all the different types of control in the project and the respective connection is made with the advances made previously, testing the correct connection and its operation.

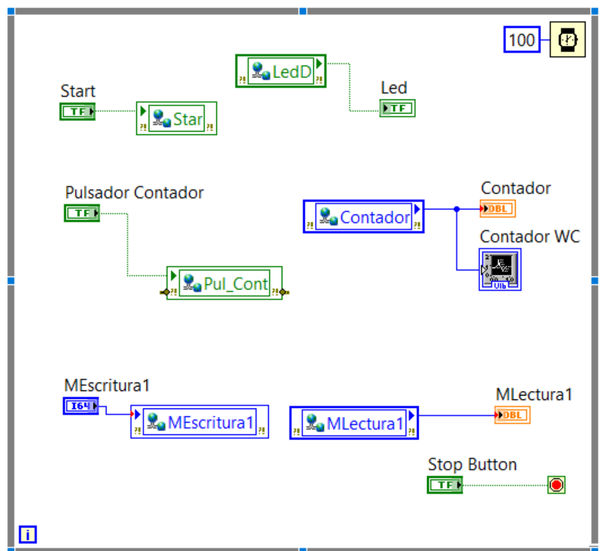


Figure 57: VI diagram block

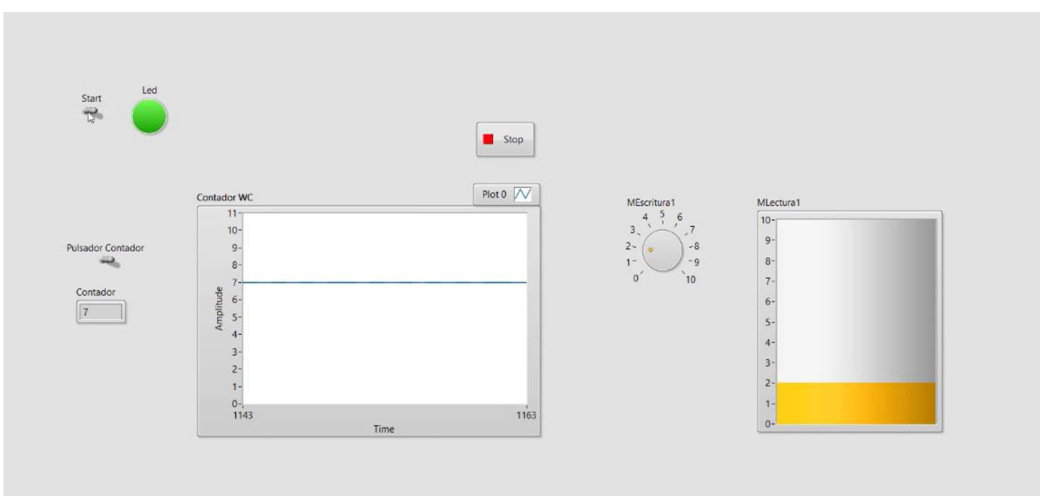


Figure 58: VI front panel with correct operation

IX.4.5. Problems

- No issues were found during the development of the trailer.

IX.4.6. Comments

- The compatibility between the main software and its modules must be taken into account, preferably all should be the same version of the year in which it was released. Otherwise, correct operation between modules and software not belonging to the National Instruments brand is not guaranteed.

IX.5. PID control

IX.5.1. . Objectives

- Recognize the LabVIEW programming environment.
- Describe the block configuration processPID_Compact

IX.5.2. Theoretical support

IX.5.2.1 Basic principles of regulation.

IX.5.2.1.1 Regulation loop

A simple example of a regulation loop is the regulation of the room temperature by means of a heater. The ambient temperature is measured with a sensor and transmitted to a regulator. This compares the current room temperature with a setpoint and calculates an output value (manipulated variable) to control the heating.

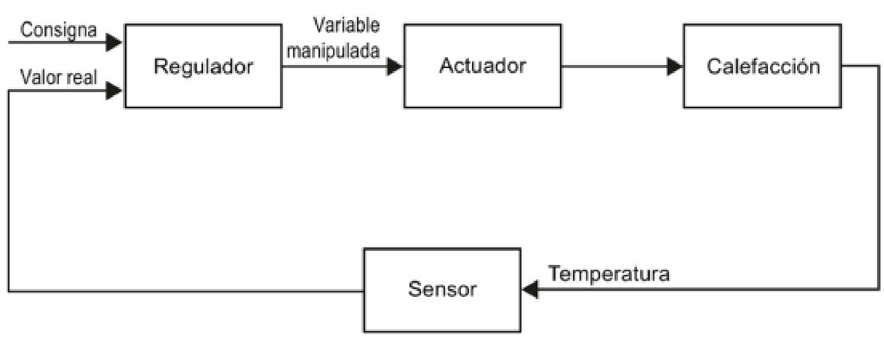


Figure 59: Closed control loop

A properly adjusted PID regulator reaches the setpoint as quickly as possible and then keeps it constant. After a change in the output value, the actual value usually changes slowly. This behavior must be compensated with the regulator.

IX.5.2.1.1 Final elements of regulation

The final control element is a component of the regulation loop and is influenced by the regulator. In this way a mass or energy flow is modified. The following table shows the fields of application of the final control elements.

Aplicación para	Elemento final de control
Flujo másico de líquidos y gases	Válvula, compuerta, corredera
Flujo másico sólido p. j. carga a granel	Válvula de descarga, cinta transportadora, canal vibratorio
Flujo eléctrico	Conmutador, contactor, relé, tiristor
	Resistor variable, transformador de regulación, transistor

Figure 60: Final control elements

IX.5.2.1.1 Regulated systems.

The properties of a regulated system are defined by technological and mechanical aspects that leave a very small margin of influence. For the regulation to produce good results, it is essential to choose an appropriate type of regulator, which can be optimally adapted to the speed of response of the regulated system. Therefore, it is essential to know in depth the characteristic data and the type of the regulated system in order to be able to configure the P, I and D action of the regulator.

- **Types of regulated systems**

Regulated systems are classified according to their speed of response to sudden changes in the output value.

The following regulated systems are distinguished:

- Regulated systems with compensation.

  - Regulated systems type P

  - Regulated systems type PT1

  - Regulated systems type PT2

- Regulated systems without compensation.
- Regulated systems with and without dead time.

- **Regulated systems with compensation.**

  - *Regulated systems type P*

    In proportional systems the real value follows the output value almost instantly. The relationship between the actual value and the output value is expressed by the proportional gain Gain of the regulated system.

    Examples:
    - Slide valve in a pipe system
    - Voltage divider
    - Reduction in hydraulic systems

  - *Regulated systems type PT1*

    In a PT1-type regulated system, the actual value initially varies in proportion to the change in the output value. As time elapses, the actual value changes more slowly (with a delay) until it reaches a final value.

Examples:

- Spring damping systems.
- Loading of RC components.
- Steam heated water tank.

The same time constants often apply for heating and cooling processes or charging and discharging curves. If the time constants are divergent, regulation becomes much more complicated.

- *Regulated systems type PT2*

In a PT2-type regulated system, the real value does not change at first with a step of the output value, but then increases in an upward ramp until finally approaching the setpoint with a downward ramp. The regulated system exhibits a proportional transfer behavior with 2nd degree delay.

Examples:

- Pressure regulation.
- Flow regulation.
- Temperature regulation.

- **Regulated systems without compensation.**

Regulated systems without compensation show comprehensive behavior. The real value pursues an infinitely large value

IX.5.2.1 PID_Compact

The technology object PID_Compact offers a continuous PID controller with integrated optimization. Furthermore, it is also possible to configure a pulse regulator. You can choose between manual and automatic mode. PID-Compact continuously records the actual measured value within a control loop and compares it with the desired setpoint. From the resulting regulation error, the PID_Compact instruction calculates an output value, with which the actual value is equated with the setpoint with maximum speed and stability. In PID controllers, the output value is made up of three actions:

- *Action P*
  The action P of the output value increases proportionally to the regulation error.

- *Action I*
  The I action of the output value increases until the regulation error is compensated.

- Action D
  Action D increases with increasing rate of variation of the regulation error. The real value equals the setpoint as quickly as possible. If the speed of change of the regulation error decreases again, so does action D.

  The PID_Compact instruction calculates the P, I, and D parameters for your regulated system autonomously during initial optimization. Parameters can be further optimized through fine tuning. It is not necessary to determine the parameters manually.

  PID_Compact is a PIDT1 controller with Anti-Windup and weighting of the P and D actions.

IX.5.2.1.1 Algorithm

The PID algorithm works according to the following formula:

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Símbolo	**Descripción**
y	Valor de salida del algoritmo PID
K_P	Ganancia proporcional
s	Operador laplaciano
b	Ponderación de la acción P
w	Consigna
x	Valor real
T_I	Tiempo de integración
T_D	Tiempo derivativo
a	Coeficiente para el retardo de la acción derivada (retardo de la acción derivada $T1 = a \times T_D$)
c	Ponderación de la acción D

IX.5.3. Progress analysis

The base program for a PID control will be developed.

- ***Step 1:***Creation of a segment dedicated to the scaling of the Set Point in the main cycle.

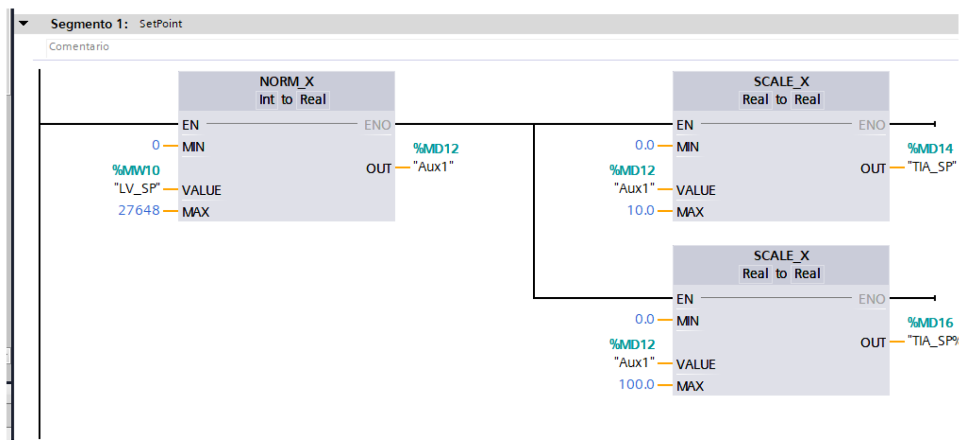


Figure 61: Step 1

- ***Step 2:***Creation of segment dedicated to scaling the sensor signal in the main cycle.

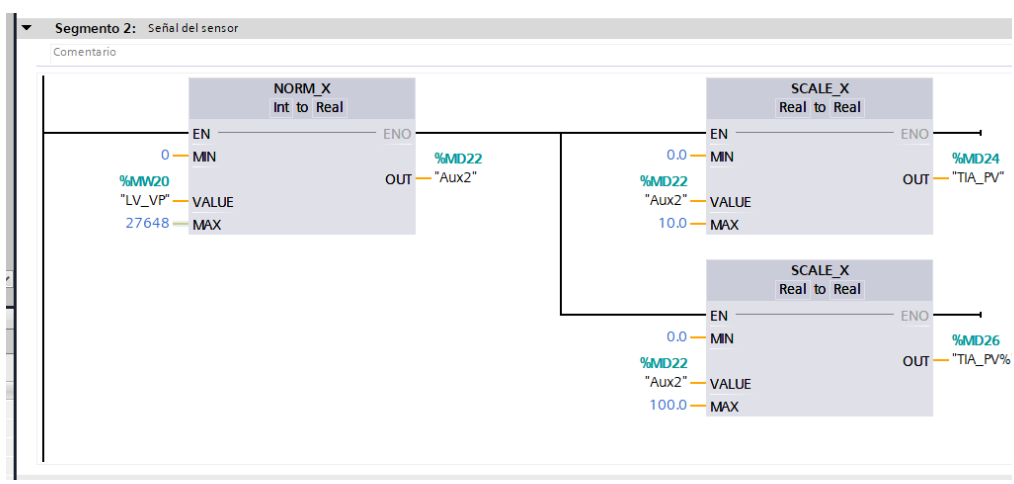


Figure 62: Step 2

- ***Step 3:***Creation of a segment dedicated to the scaling of the Set Point in the main cycle.

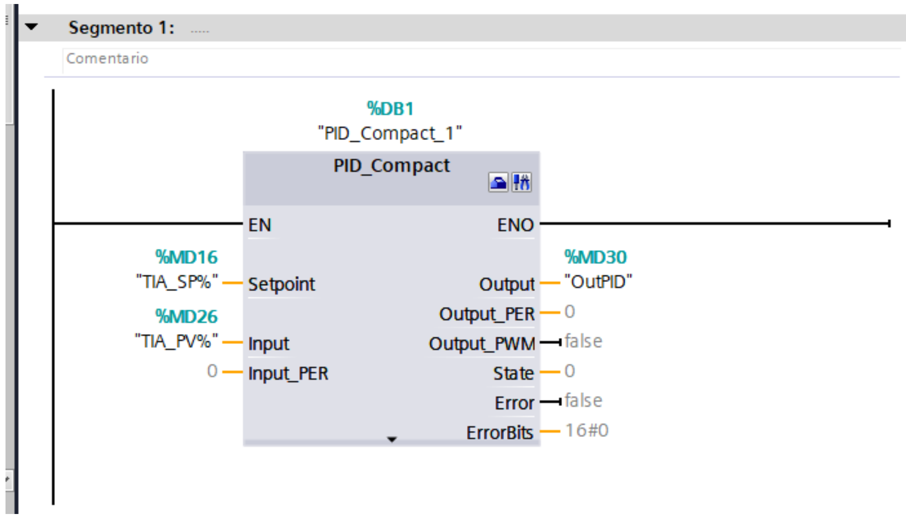


Figure 63: Step 3

- ***Step 4:***Open configuration window> Basic settings> Configure according to Figure XX.

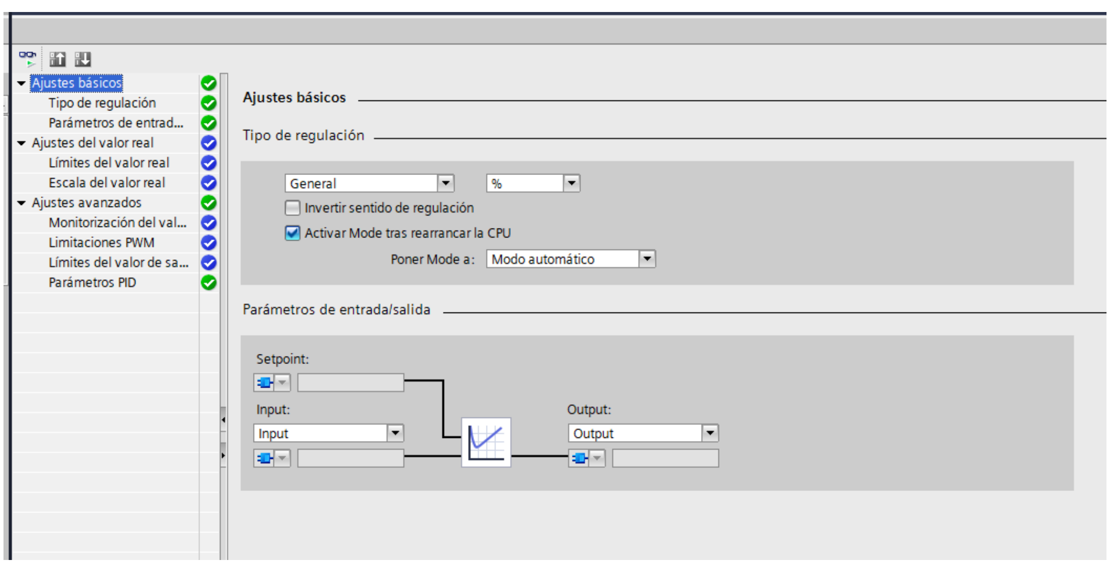


Figure 64: Step 4

➢ ***Step 5:***PID parameters> Activate manual input, disconnected to activate PID tuner.

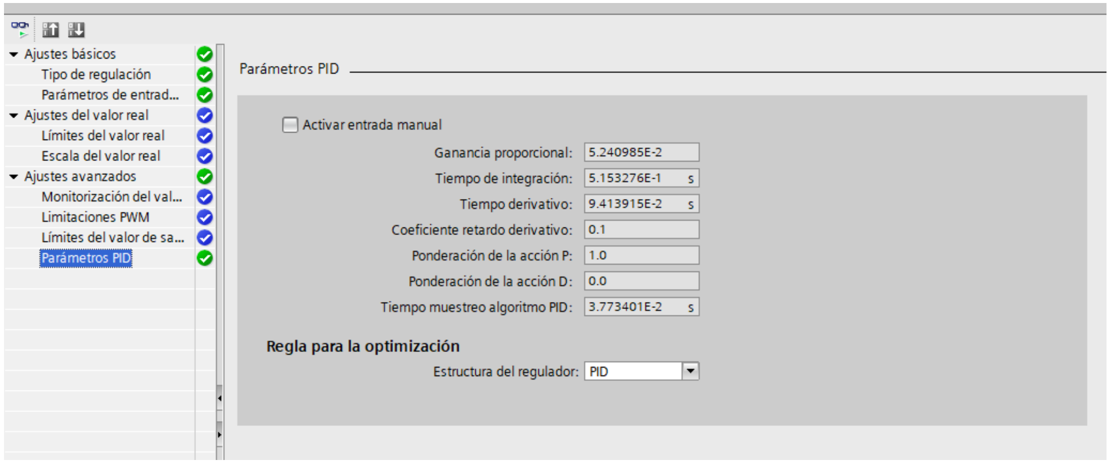


Figure 65: Step 5

➢ ***Step 6:***Perform modeling in LabVIEW.

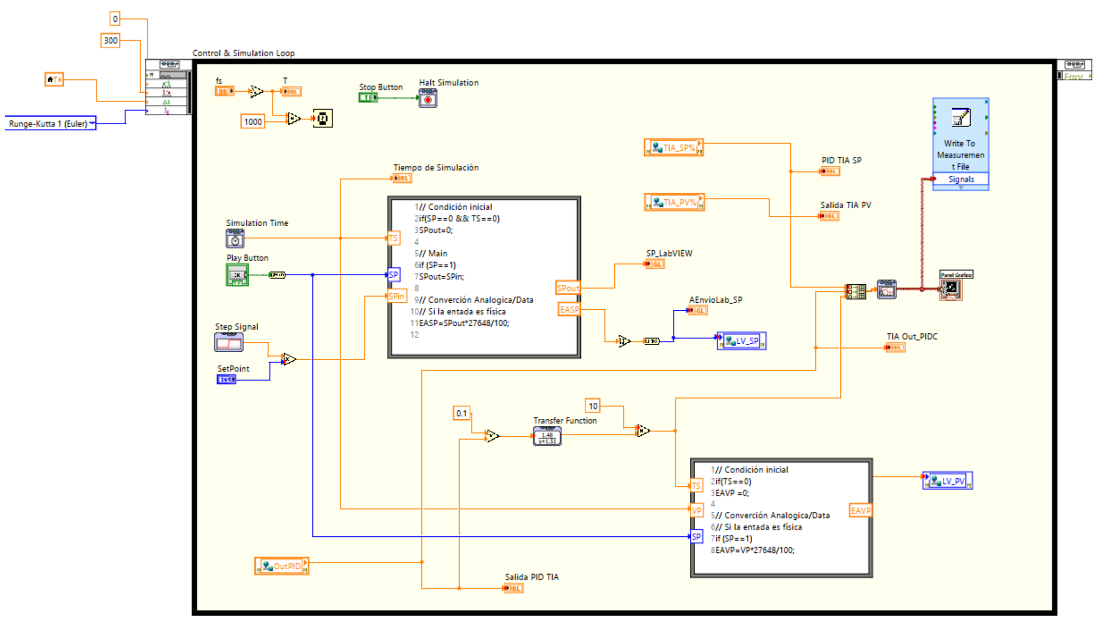


Figure 66: VI diagram block

IX.5.4. Results, measurements and validation.

Starting the simulation, the operation of the auto tuning is checked in the start-up window.

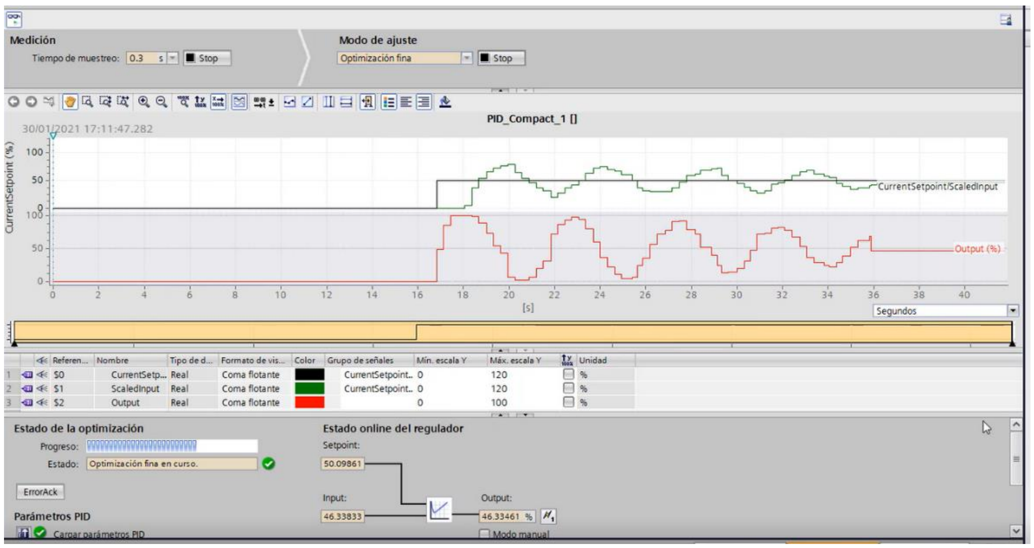


Figure 67: Start-up window.

The autotune operation is checked on the front panel of the VI.

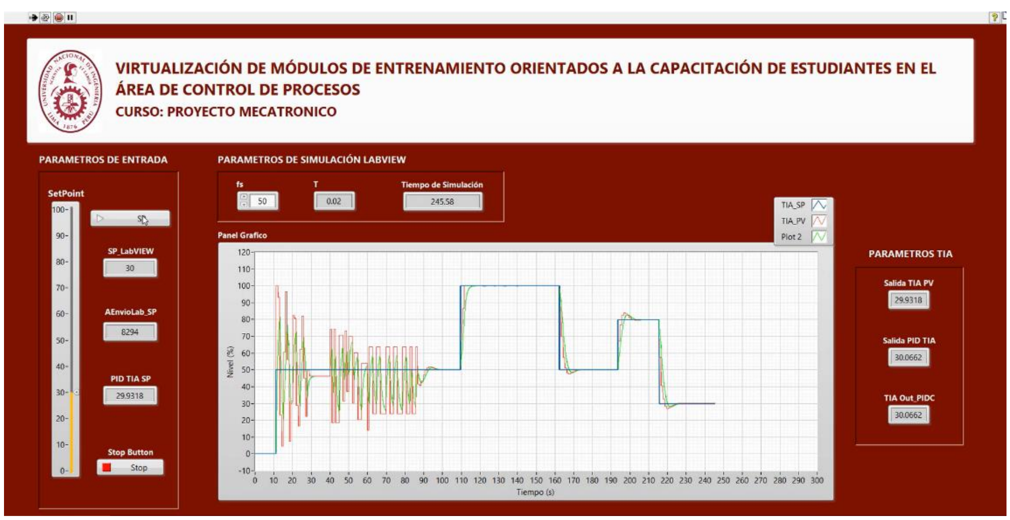


Figure 68: Auto-Tuning in LabVIEW

With the data obtained we can obtain a better tuning depending on the needs of our process.

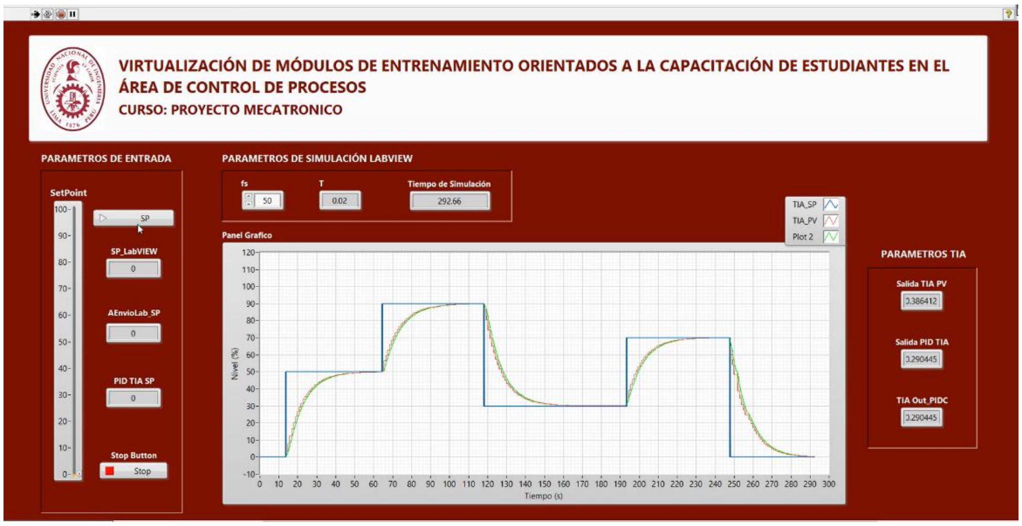


Figure 69: Regularized PID

IX.5.5. Problems

- During the development of the preview, the PID Compact block of S7-1200 modules could not be simulated, whereas with S7-1500 and S7-300 modules.

IX.5.6. Comments

- It should be noted that not all models available on the TIA platform are capable of simulating its PID block.

X. Financial and economic report.

Name	Quantity	Unit cost	Total cost	Comments
PC (i5 Processor, 16GB RAM, 500GB HDD)	1	0	3000	Available at university
PLC training module (s7-300)	1	0	5000	Available at university
NI LabVIEW 2015	1	0	10,000	Available at university
NI LabVIEW module 2015: Datalogging and Supervisory Control Module	1	0	10,000	Available at university
NI LabVIEW 2015: Real Time Deveploment	1	0	10,000	Available at university
NetToPLCSim (v14)	1	0	0	Free software
TIA Portal V16	1	0	0	Available in bundle with purchase of PLC
S7- PLCSim	1	0	0	Available in bundle with purchase of PLC
TOTAL (Soles):			395000	

XI. Conclusions and recommendations

- The project advance opens the possibility of implementing the project not only in the process control course, but also in control courses where it is required to implement other more complex control methods that can be implemented in LabVIEW, such as the digital control course. , non-linear control, etc., under the direction of the teachers in charge.

- If you want to implement the simulators in the courses, it is necessary to create an introductory guide on the LabVIEW environment for better handling of the virtualized modules.

XII. Bibliographic references

[1] From Lorenzo Group

Consultation: November 11, 2020

<https://www.delorenzoglobal.com/documenti/prodotti/210318_DL_2314_SPA.pdf>

[two] GUNT Hamburg

Consultation: November 23, 2020

<http://www.usdidactic.com/teaching-lab-equipment/p3237.htm#>

[3] Cruz De La Cruz, JE, & Vera Ramirez, OJ (2020). Convolutional neural networks for the Hass avocado classification using LabVIEW in an agro-industrial plant. Proceedings of the 2020 IEEE 27th International Conference on Electronics, Electrical Engineering and Computing, INTERCON 2020, 20–23.https://doi.org/10.1109/INTERCON50315.2020.9220246

[4] Ospino-Pinedo, J., Sánchez, ME, & Matallana-Pérez, LG (2018). Modeling and control of processes using the EMSO simulator and SIMATIC PLC as OPC inter-face. Magazine Faculty of Engineering, 87, 58–69.https://doi.org/10.17533/udea.redin.n87a08

[5] Duque Marín, A., Lopez, JA, & Navas, AF (2017). Auto-tuning of a PID controller implemented in a PLC using swarm intelligence. 15 (1), 35-41.http://dx.doi.org/10.15665/rp.v15i1.679

[6] Berger, H. (2014). Automating with SIMATIC S7-300 inside TIA Portal. Configuring, Programming and Testing with STEP 7 Professional. Publicis Erlangen.

[7] Thomas Wiens (2018). NetToPLCsim. Network extension for Plcsim.

Consultation: November 01, 2020.

<https://usermanual.wiki/Document/NetToPLCsimManualen.1468207178/view>

[8] Rodriguez Penin, A. (2008). Industrial Communications. Practical Guide. Editorial Marcombo.

[9] Matrikon. What is an OPC Server?

Consultation: November 01, 2020.

<https://www.matrikonopc.es/opc-servidor/index.aspx>

[10] *Siemens TCP / IP Ethernet Driver*

Consultation: November 02, 2020

<https://www.kepware.com/getattachment/b636f30a-8672-4169-8c9d-a7211a78990c/siemens-tcp-ip-ethernet-manual.pdf>

[11] Standard IEC 61131-3: 2003: Programmable controllers - Part 1: General Information.

Consultation: November 28, 2020

< https://webstore.iec.ch/publication/4550>

[12] Standard IEC 61131-2: 2017: Programmable controllers - Part 2: Equipment requirements and tests.

Consultation: November 28, 2020

< https://webstore.iec.ch/publication/31007>

[13] Standard IEC 61131-3: 2013: Programmable controllers - Part 3: Programming languages.

Consultation: November 28, 2020

< https://webstore.iec.ch/publication/4552>

[14] Standard IEC 61131-4: 2004: Programmable controllers - Part 4: User guidelines.

Consultation: November 28, 2020

< https://webstore.iec.ch/publication/4553>

[15] Standard IEC 61131-5: 2000: Programmable controllers - Part 5: Communications.

Consultation: November 28, 2020

< https://webstore.iec.ch/publication/4554>

[16] Standard IEC 61131-8: 2017: Programmable controllers - Part 8: Guidelines for the application and implementation of programming languages.

Consultation: November 28, 2020

< https://webstore.iec.ch/publication/33021>

XIII. Annexes

XIII.1 List of standards used at work.

1. **IEC 61131-1: 2003 standard: Programmable controllers - Part 1: General Information.**

IEC 61131-1: 2003 applies to programmable controllers (PLCs) and their associated peripherals, such as programming and debugging tools (PADT), human-machine interfaces (HMI), etc., intended for control and command . of machines and industrial processes. Provides definitions of terms used in this standard. Identify the main functional characteristics of programmable controller systems. This second edition cancels and replaces the first edition published in 1992 and constitutes a technical revision.

2. **IEC 61131-2: 2017 standard: Programmable controllers - Part 2: Equipment requirements and tests.**

IEC 61131-1: 2017 specifies the functional and electromagnetic compatibility requirements and related verification tests for any product where the primary purpose is to perform the function of industrial control equipment, including PLC and / or PAC, and / or their associated peripherals that are intended to use the control and command of machines, automated industrial and manufacturing processes, eg. ex. discrete, batch and continuous control.

This fourth edition cancels and replaces the third edition published in 2007. This edition constitutes a technical revision.

3. **IEC 61131-3: 2013 standard: Programmable controllers - Part 3: Programming languages.**

IEC 61131-3: 2013 specifies the syntax and semantics of a unified set of programming languages for programmable controllers (PCs). This suite consists of two textual languages, Instruction List (IL) and Structured Text (ST), and two graphical languages, Ladder Diagram (LD) and Function Block Diagram (FBD). This third edition cancels and replaces the second, published in 2003 and constitutes a technical revision.

4. **Standard IEC 61131-4: 2004: Programmable controllers - Part 4: User guidelines.**

Introduces end users of programmable controller (PLC) to the IEC 61131 series and assists end users in the selection and specification of their PLC equipment in accordance with the IEC 61131 series. This user guide is primarily intended for users end of PLC.

5. **IEC 61131-5: 2000 standard: Programmable controllers - Part 5: Communications.**

IEC 61131-5: 2000 specifies communication aspects of a programmable controller. It specifies from the point of view of a PC how any device can communicate with a PC as a server and how a PC can communicate with any device. In particular, it specifies the behavior of the PC as it provides services on behalf of other devices and the services that the PC application program can request from other devices. This bilingual version (2012-08) corresponds to the monolingual version in English, published in 2000-11.

6. **IEC 61131-8: 2017 standard: Programmable controllers - Part 8: Guidelines for the application and implementation of programming languages.**

The IEC TR 61131-8: 2017 (E) standard applies to the programming of programmable controller systems that use the programming languages defined in the IEC 61131-3 standard. The scope of the IEC 61131-3 standard is applicable to this part.

This document provides:

a) Guidelines for the application of the IEC 61131-3 standard,

b) Guidelines for the application of the languages of the IEC 61131-3 standard for programmable controller systems

c) Recommendations on programming and debugging tools (PADT).

This third edition cancels and replaces the second edition published in 2003.

XIII.2 Limitations and / or restrictions of the project.

- The main limitation in the development of the project has been the impossibility of buying the simulation with a real plant, due to the restriction of admission to the university due to the pandemic.