UNIVERSITY OF LJUBLJANA | FS Faculty of Mechanical Engineering

Vid Jerant

Implementation of a Servo System in TiaPortal (Siemens)

Project work for the subject Interdisciplinary Project

Ljubljana, 2024

Contents

List of Figures

List of Tables

Introduction

Some buildings have a shaft in the basement that serves to drain rising groundwater. The problem with these shafts is that when the groundwater level rises significantly, the shaft can no longer drain the water. This is solved by installing a pump. To avoid the need for a person to turn it on, it is automated.

In this project, I simulated the operation of the pump so that as the water level rises, the pumping speed also increases. As the position of the float, to which a potentiometer is attached, rises, the motor speed increases.

1 Task Description

The task instructions were to implement an actuator system with a servo motor using the Tia Portal software.

To make it easier for me, I came up with a problem involving pumping groundwater from a basement. I got the idea from last year's experience of pumping groundwater from my hometown school, where there is a shaft in the basement that starts collecting groundwater before it begins flooding the basement rooms.

1.1 Problem Solution

Water flooding would be prevented by automatically pumping water from the groundwater shaft. A float would be installed in the shaft to monitor the water level. When the water reached a certain level, the pump would start. If the water level continued to rise, the pump's flow would increase by speeding up the turbine in the pump.

2 Controller

[4] The controller used was the Siemens Simatic S7-1500. This controller is intended for industrial process automation and is used in various industries such as manufacturing, energy, building automation, and other industrial sectors.
Its key features are:

-High performance: The S7-1500 provides fast and reliable process control due to its high-performance processor.

-Safety: The controller supports built-in safety features, allowing the implementation of hardware safety measures such as emergency stop, safety speed, and other safety functions.

-Connectivity: The S7-1500 is equipped with various interfaces, including Profinet, one of the Ethernet protocols, which enables easy communication with other devices such as sensors, actuators, drives, etc.

-Scalability: The controller supports additional modules and expansions, allowing the system to be adapted to the specific requirements of the user.

-Ease of programming: Siemens S7-1500 allows programming in various languages such as LAD (ladder diagram), FBD (functional block diagram), STL (statement list), SCL (structured control language), and GRAPH (block-based step programming).

-Diagnostics and maintenance: The controller is equipped with advanced diagnostic tools, enabling quick identification and troubleshooting, as well as optimizing system performance.

-Flexibility: The S7-1500 supports various configurations, allowing the system to be tailored to specific production needs.

In the automation industry, the Siemens S7-1500 is known for its reliability, performance, and advanced features, making it one of the leading controllers on the market.

2.1 Siemens Simatic S7-1512C-1 PN

[3] The controller I used had a central processing unit (CPU) labeled Simatic S7-1512C-1 PN. It offers 5 analog and 32 digital inputs, with 2 analog and 32 digital outputs. It also has two PROFINET IO IRT connectors, which I used for programming.

The CPU has a very short program cycle time, processing one binary unit in just 60 ns. Besides speed, it has a large memory capacity, with 250 KB for programs and 1 MB for data. It also allows the addition of an SD memory card, which increases memory capacity. The CPU can connect to up to 31 additional modules, making it possible to program very complex industrial systems.

Figure 1: Controller S7-1512C-1 PN

2.2 Tia Portal

[5] The controller I used is programmed using Siemens' TIA Portal (Totally Integrated Automation Portal) software platform, developed for programming and configuring their industrial automation

products. This platform allows programming, system configuration, process visualization, and diagnostics and maintenance, all through a simple and integrated interface.

Here are some key features and modules of TIA Portal:

-Integration of various systems: TIA Portal integrates various components into a single interface, including PLCs (Programmable Logic Controllers), HMIs (Human Machine Interfaces), frequency converters, industrial communication networks, and management systems.

-Unified interface: All tools within TIA Portal share the same interface, simplifying the work for engineers and reducing the need to learn different software.

-PLC programming: TIA Portal allows PLC programming for controlling industrial processes. Various programming languages, such as STEP 7 (STL, LAD, FBD, SCL), are supported, including PLC simulation.

-Visualization (HMI): TIA Portal includes tools for creating user interfaces (HMIs) for system control and management. This includes designing screens, setting up alarm systems, and reporting.

-Diagnostics and maintenance: The software provides engineers with tools for system diagnostics and maintenance through advanced analysis and monitoring tools.

-Simulation: TIA Portal supports simulation, allowing engineers to test and verify their programs before implementing them in real conditions.

-Automated workflows: TIA Portal enables the automation of workflows from planning to implementation and maintenance, reducing the possibility of human errors and increasing efficiency.

TIA Portal has become a standard tool in the automation industry due to its integration, ease of use, and support for various aspects of system programming and configuration, enabling engineers to quickly develop, test, and maintain complex industrial systems more efficiently than ever before.

3 Siemens Simatic S210

[2] Simatic S210 is a servo drive system that enables advanced control of servomotors. This allows us to control high-performance dynamic applications. It is designed to control Simotics servo motors, which typically have a power range from 50 to 750 W.

The motor is connected to the drive system via the Motion-Connect OCC, which combines all the necessary conductors for motor operation. Motion-Connect provides power, encoder data, and a brake activation signal.

Simatic S210 simplifies programming significantly, as it has a built-in PID controller for motor speed, allowing the user to simply input the desired shaft speed, which the system then maintains with minimal stationary-state deviations and oscillations. Additionally, it allows customizable acceleration and deceleration settings.

Figure 2: Sinamics S210 servo drive


Figure 3: Simatic servo motor

I connected a 400 W Simotics 1FK2103-4AG00-1MA0 motor (figure 3) to the drive, and its basic specifications are shown in the following table [1]:

U_{IN}	110 V AC
M_0	1.27 Nm
I_0	2.4 A
M_n	1.27 Nm
I_n	2.4 A
n_n	3000 rev/min
$n_{\max}$	7300 rev/min

Table 1: Motor specifications

4 System Operation

4.1 Operating Principle

The system operates by turning on the motor that drives the pump when a certain water level is reached in the shaft. As the water level rises, the rotational speed of the motor increases as well. The control loop is shown in the following diagram.

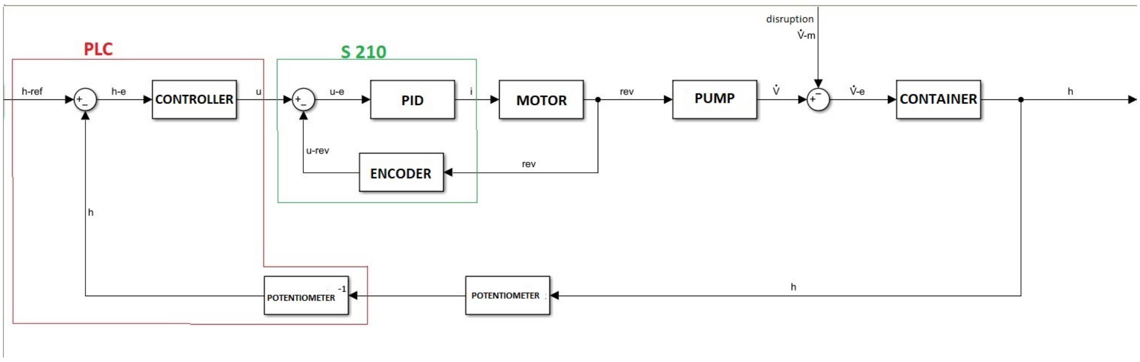

Figure 4: Control loop diagram

As seen in the figure, the desired level is input into the controller, which in our case is the minimum level the pump can still handle, while also monitoring the actual water level in the shaft. The controller sends a voltage signal to the drive, which has a built-in PID controller for motor speed. The drive continuously reads values from the motor's encoder to determine the actual motor speed. It compares this value with the setpoint provided by the controller and adjusts the speed accordingly by regulating the current sent to the motor.

4.2 Communication

The drive and the controller communicate via a Profibus cable, meaning they are connected to the same network and each has its own IP address. This simplifies the connection, especially if the system includes multiple modules like an HMI screen, servo drives, etc.

4.2.1 Profinet

Profinet is a high-performance industrial Ethernet protocol used for automation and industrial applications. It is an advancement from Profibus but is based on Ethernet technology, allowing for faster data transfer and greater performance compared to traditional Profibus.

Profinet enables high-speed communication between various automation devices such as controllers, sensors, actuators, and other equipment in industrial environments. Its Ethernet base allows easy integration with existing networks and infrastructure.

The advantages of Profinet include high data transfer speed, low latency, high communication reliability, easy integration with Ethernet networks, and support for various network topologies, including star, ring, and star-ring networks.

5 Program

5.1 Opening a New File and Selecting Devices

Before launching TiaPortal, I verified that it was the correct version and compatible with all the devices being used.

When TiaPortal is launched, we select 'New project' and then choose the devices we are using by double-clicking on 'Add new device' in the tree on the left side (Figure 5). Under the CPU section, I selected the controller I used, and under the 'Drive' section, I chose the drive interface for the motor. When selecting devices, both the controller and the drive, it is crucial to ensure the correct type is chosen, as specified on the actual device.

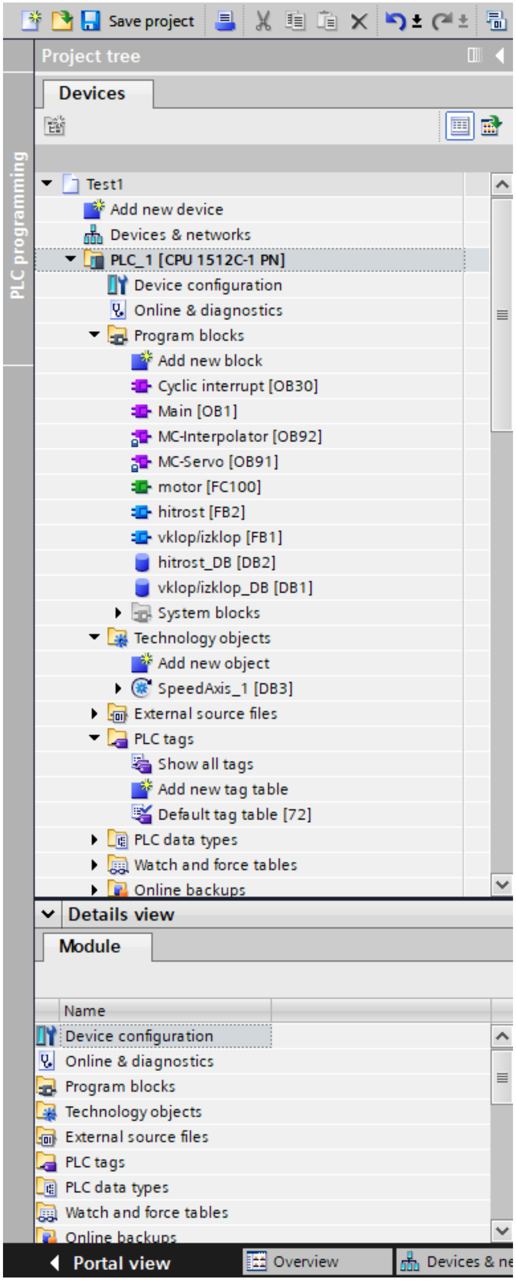

Figure 5: Tree structure

Once the drive is selected, we need to select the motor as well. To do this, we open 'Device view' (Figure 5), select the device (Figure 6), and click on the circle labeled 'MOT' on the drive schematic. This opens a tab where we can choose the motor model we have.

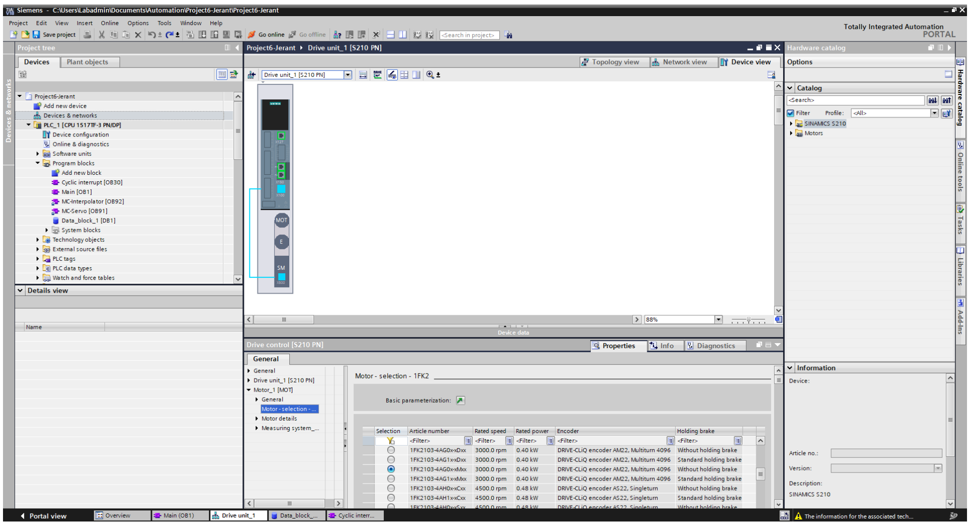

Figure 6: Device view S210

5.2 Writing the Program

-Defining connections: In the Network view (figure 7) and Topology view (figure 8), I specified how the devices are connected. In this case, the connection is almost identical.

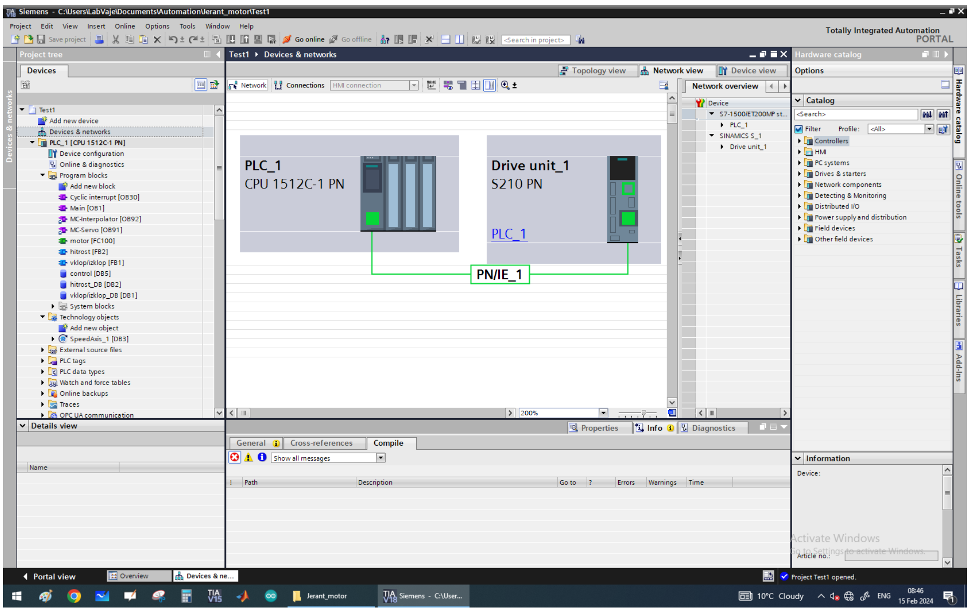

Figure 7: Network view

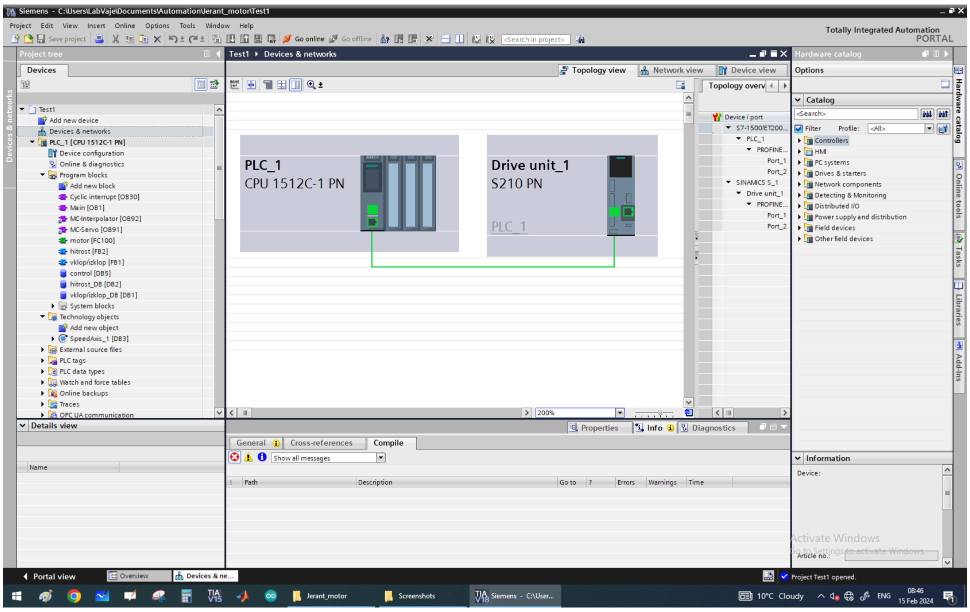

Figure 8: Topology view

-Defining variables linked to actual inputs and outputs: In the 'PLC tags' tab on the left, select 'Add new tag table', where I defined variables linked to the actual inputs and outputs of the controller.

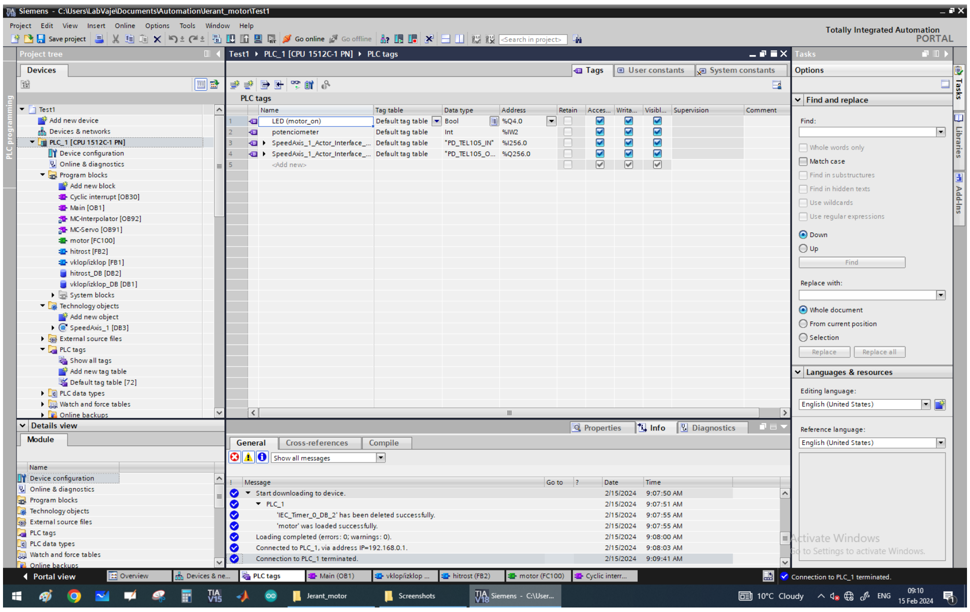

Figure 9: External variables

-Main program: In the main program, I added blocks, each with its own function, which I defined through subprograms. The program is written in Ladder Diagram (LBD) programming language. When the switch is activated, the motor and an LED, which indicates the motor is running, are turned on. When the switch is off, the motor shuts down. The switch functionality is defined in the 'On/Off' subprogram. A timer is added in the 5th line of the program, which resets the motor after 15 seconds by cutting off its power in the motor subprogram.

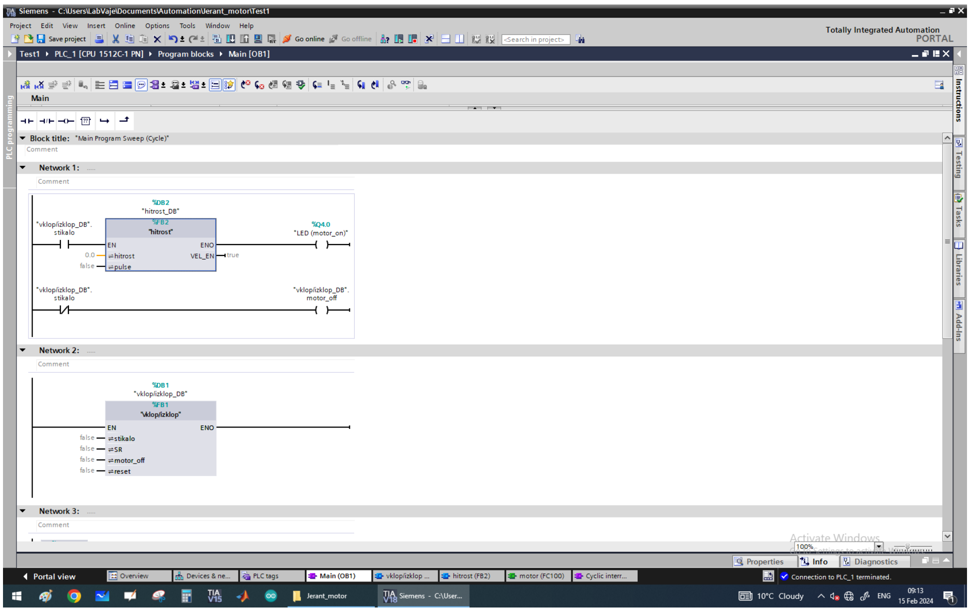

Figure 10: Main program (1/2)

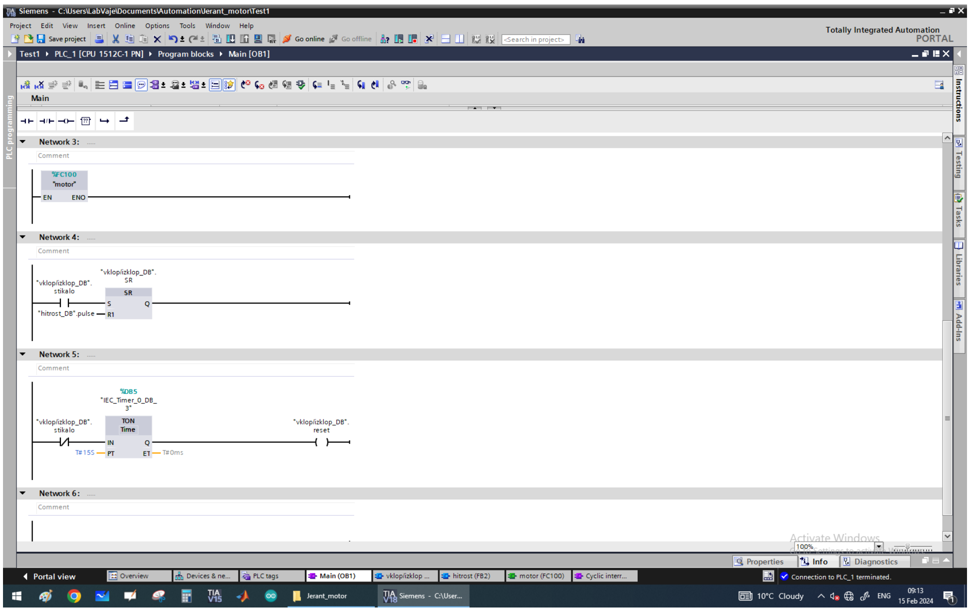

Figure 11: Main program (2/2)

-On/Off subprogram: Here, I defined when the 'switch' variable is activated. Specifically, when the signal from the potentiometer is greater than or equal to 8000, the 'switch' is set to logical 1. To define this behavior, I first had to assign the necessary variables. These are set under the 'On/Off_DB' tab. Some variables are not tied to this subprogram, as they can be used in all subprograms.

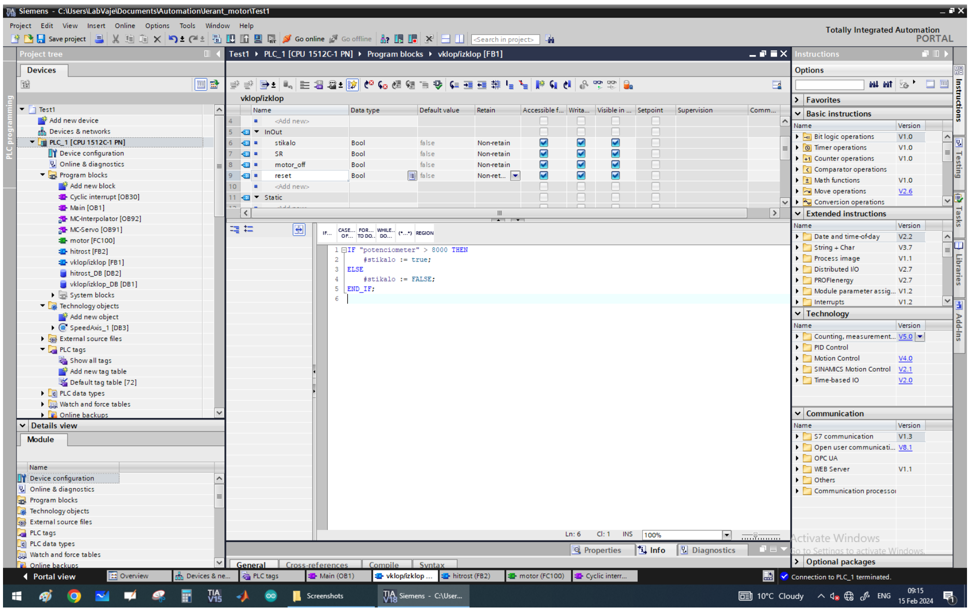

Figure 12: On/Off subprogram

-Motor Subprogram: For the subprogram that controls the motor, I first had to select what I would control on the motor under the 'Technology objects' tab. Since my application required speed control, I selected 'SpeedAxis'. Once this was selected, I created a new function (Add new block -¿ function) and named it 'motor'. Here, from the group of commands on the right-hand side, in the folder 'Technology' -¿ 'Motion control', I inserted the following blocks:

-*MC_Power:* This block ensures the motor is powered (figure 13).

-*MC_Reset:* Activating the execute signal resets the motor, meaning the power is cut off (figure 13).

-*MC_Halt:* This block stops the motor when the execute signal is activated, but it remains powered. I set the deceleration value under 'Deceleration' (figure 14).

-*MC_Movevelocity:* This block controls the motor's rotational speed, specifically under 'Velocity', where I linked the 'speed' variable from the corresponding subprogram. Acceleration and deceleration are also set here. I linked the 'pulse' variable to the Execute input, which I defined with the 'Cycle interrupt' block in the main program, as the MC_Movevelocity block only reads values on the rising edge of the pulse (figure 14).

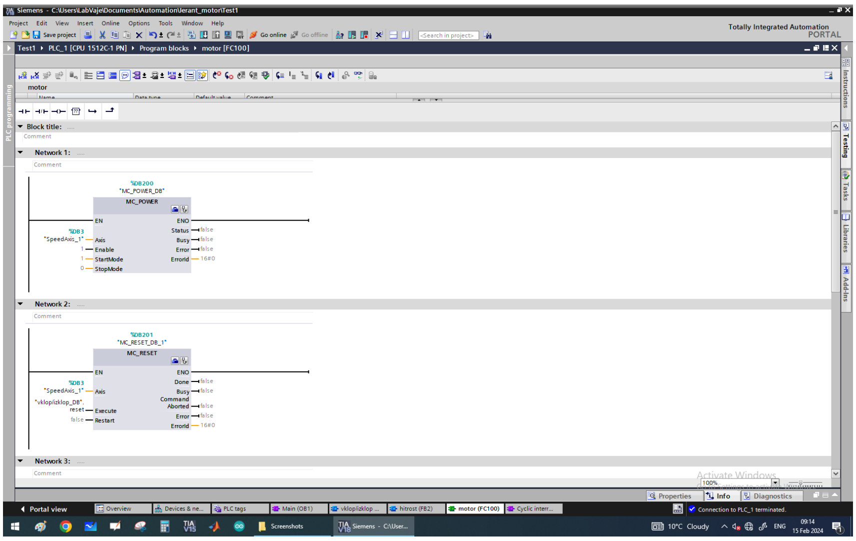

Figure 13: Motor subprogram (MC_Power and MC_Reset blocks)

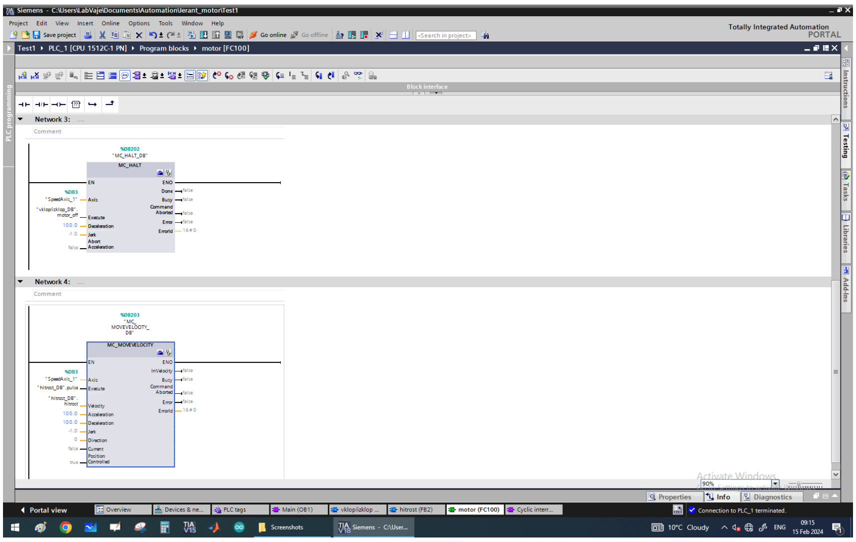

Figure 14: Motor subprogram (MC_Halt and MC_Movevelocity blocks)

-Speed Subprogram: Here, I defined the connection between the potentiometer values and the speed, which I determined through experimentation. I found that I had to divide the values obtained from the potentiometer by 6 for the relationship between the potentiometer and the motor's rotational speed to be linear. Since the data from the potentiometer is received as integers (whole numbers), I had to convert them to real numbers using the 'INT TO LREAL' command (figure 15).

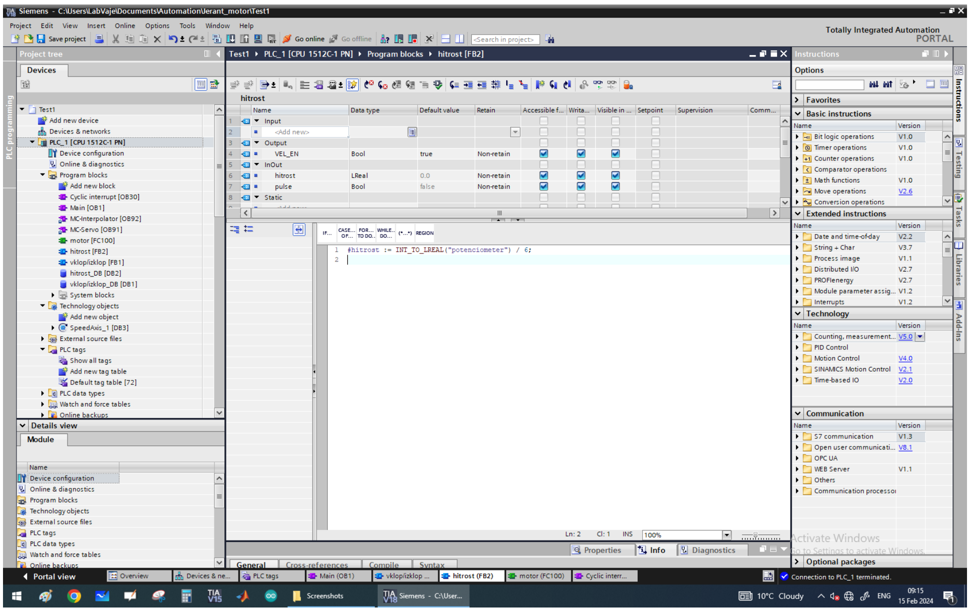

Figure 15: Speed subprogram

-Cycle Interrupt: In the cycle interrupt, I added a time block that allows the signal to pass every 100 ms. The block starts counting time when the SR block in the main program is activated (figure 16). The function of the Cycle Interrupt subprogram is that it is only active for a certain time, after which it turns off and restarts. I set the interruption time to 5 ms (figure 17).

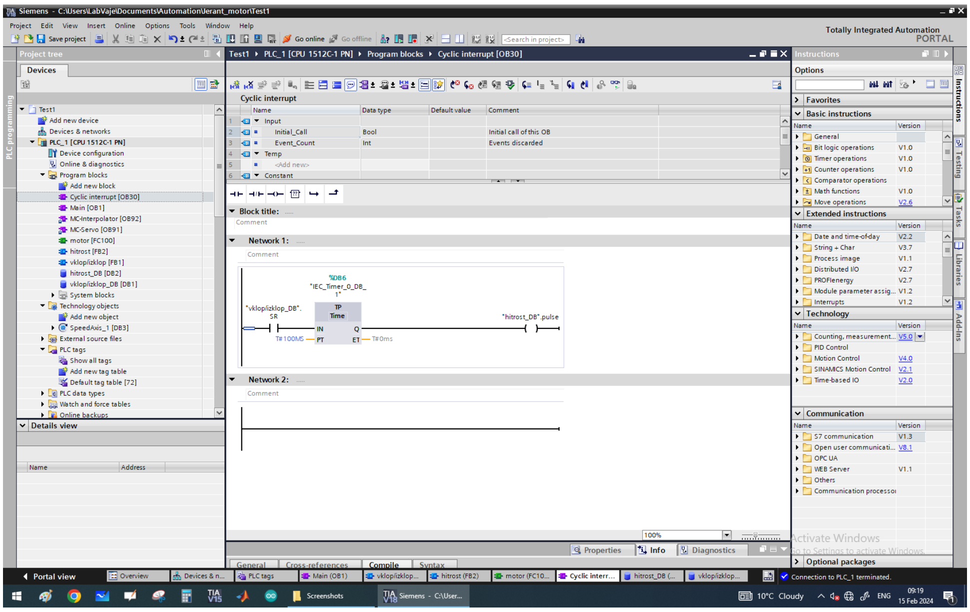

Figure 16: Cycle interrupt

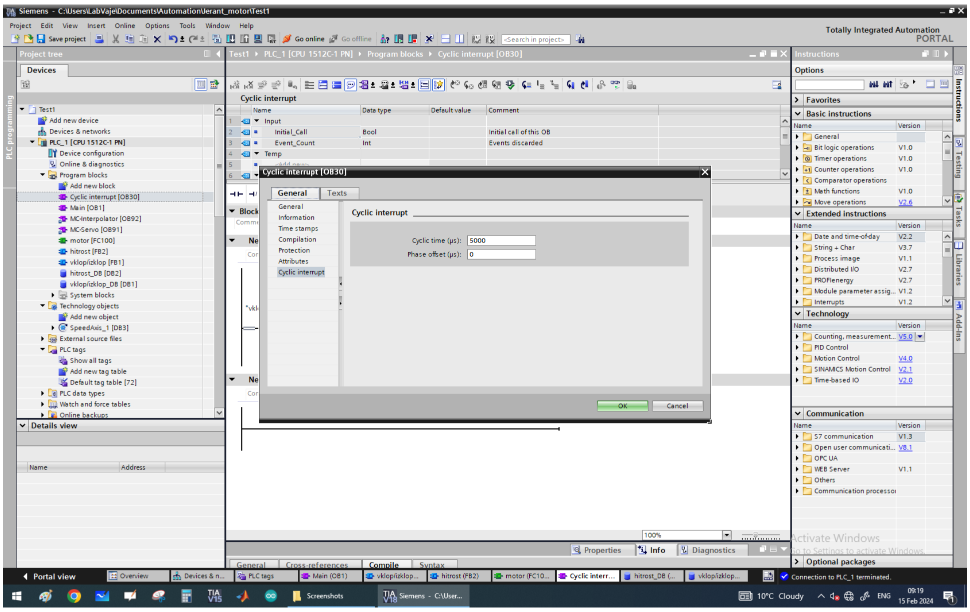

Figure 17: Cycle interrupt operation time

6 Conclusion

This document presents the basic control of a motor using a Siemens controller. With TIA Portal, it is also possible to control spindle movements, angles of rotation, etc. All of these, like the speed control of the spindle, already have pre-prepared programs for these applications under the 'Technology objects' tab.

Initially, I intended to display values on the HMI screen, but this was not possible because the TIA Portal version was not compatible with the screen's software version.

References

[1] Siemens. 1fk2103-4ag00-1ma0. https://mall.industry.siemens.com/mall/en/se/Catalog/Product/1FK2103-4AG00-1MA0, 2024. (6. 11. 2024).

[2] Siemens. Siemens simatic s210. https://www.siemens.com/global/en/products/drives/sinamics/low-voltage-converters/servo-converter/sinamics-s210.html, 2024. (6. 11. 2024).

[3] Siemens. Siemens simatic s7-1512c-1 pn. https://mall.industry.siemens.com/mall/en/WW/Catalog/Products/10268310?activeTab=ProductInformation, 2024. (6. 11. 2024).

[4] Siemens. Simatic s7-1500. https://www.siemens.com/global/en/products/automation/systems/industrial/plc/simatic-s7-1500.html, 2024. (6. 11. 2024).

[5] Siemens. Totally integrated automation portal – always ready for tomorrow. https://www.siemens.com/global/en/products/automation/industry-software/automation-software/tia-portal.html, 2024. (6. 11. 2024).