UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE CUENCA

CARRERA DE ELECTRÓNICA Y AUTOMATIZACIÓN

SISTEMA DE COMUNICACIÓN INALÁMBRICO Y HMI BASADO EN PYTHON PARA LA OPERACIÓN Y MONITOREO REMOTO DE UN MOTOR TRIFÁSICO DE INDUCCIÓN

Trabajo de titulación previo a la obtención del título de Ingeniero en Electrónica

AUTORES: JAE HYUN HWANG CÁRDENAS
JAIME ANDRÉS IÑIGUEZ AVILA

TUTOR: ING. JULIO CESAR ZAMBRANO ABAD, PhD.

Cuenca – Ecuador

2023

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Nosotros, Jae Hyun Hwang Cárdenas con documento de identificación N° 0104633920 y Jaime Andrés Iñiguez Avila con documento de identificación N° 0106045883; manifestamos que:

Somos los autores y responsables del presente trabajo; y, autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Cuenca, 14 de julio del 2023

Atentamente,

Jae Hyun Hwang Cárdenas	Jaime Andrés Iñiguez Avila
0104633920	0106045883

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Nosotros, Jae Hyun Hwang Cárdenas con documento de identificación N° 0104633920 y Jaime Andrés Iñiguez Avila con documento de identificación N° 0106045883, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del Proyecto Técnico: "Sistema de comunicación inalámbrico y HMI basado en PYTHON para la operación y monitoreo remoto de un motor trifásico de inducción" el cual ha sido desarrollado para optar por el título de: Ingeniero en Electrónica, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

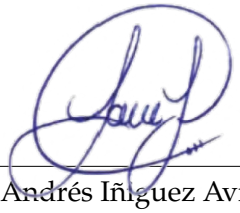
En concordancia con lo manifestado, suscribimos este documento en el momento que hacemos la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Cuenca, 14 de julio del 2023

Atentamente,

Jae Hyun Hwang Cárdenas

0104633920



Jaime Andrés Iñiguez Avila

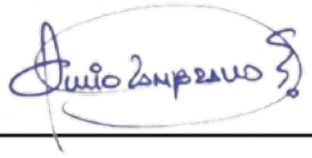
0106045883

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Julio Cesar Zambrano Abad con documento de identificación N° 0301489696, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: SISTEMA DE COMUNICACIÓN INALÁMBRICO Y HMI BASADO EN PYTHON PARA LA OPERACIÓN Y MONITOREO REMOTO DE UN MOTOR TRIFÁSICO DE INDUCCIÓN, realizado por Jae Hyun Hwang Cárdenas con documento de identificación N° 0104633920 y Jaime Andrés Iñiguez Avila con documento de identificación N° 0106045883, obteniendo como resultado final el trabajo de titulación bajo la opción Proyecto Técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Cuenca, 14 de julio del 2023

Atentamente,



Julio Cesar Zambrano Abad
0301489696

AGRADECIMIENTOS

Quiero agradecer a mis padres, Cumandá y Tae Sik, quienes nunca han dejado de confiar en mí y siempre me han guiado y amado. Es gracias a su ejemplo, su incansable esfuerzo y sus ganas de superación lo que me ha motivado y ha sido la base de todo lo que he logrado. Me enorgullece poder decir que soy su hijo y siempre los amaré.

A mis hermanas, Sophie y Anne, quienes han alegrado mis días y han sabido estar ahí siempre que lo he necesitado. Agradezco profundamente su presencia en mi vida y espero poder ser el hermano que merecen.

A mi querida esposa, Leslie, quien ha estado conmigo en todo momento. He logrado llegar hasta aquí gracias a ti. Tu apoyo incondicional, paciencia y capacidad para motivarme incluso en los momentos más difíciles, han sido esenciales para alcanzar este objetivo. Gracias por llegar a mi vida, te amo y siempre lo haré.

A mi hijo Marc, mi mayor tesoro y alegría, tú eres quien me da más fuerza y motivación. Todo lo que he hecho y todo lo que haré, siempre lo haré pensando en ti. Eres mi razón de ser, y no importa el sacrificio, tú me recuerdas que cada esfuerzo vale la pena para poder construir un futuro mejor para ti.

Quiero agradecer también a mi familia, a mis tíos, primos, abuelos y mis suegros. Han sido una fuente inagotable de comprensión y sin ellos en mi vida, no sería la persona que soy hoy.

A mi mejor amigo y compañero, Jaime. Gracias por estar conmigo a lo largo de este camino, es gracias a ti que hemos superado muchas dificultades. Eres el amigo más leal y confiable, y nuestra amistad es algo que siempre atesoraré.

Agradezco también a mi tutor, Ing. Julio Zambrano, uno de los mejores profesores

que me he encontrado en esta larga travesía académica. Gracias por su experiencia, conocimientos, paciencia y devoción a su trabajo.

Finalmente, quiero expresar mi gratitud por la confianza, el amor y el apoyo que me han brindado todos mis amigos. Espero poder ser capaz de devolver aunque sea un poco de todo lo que he recibido.

Jae Hyun Hwang Cárdenas

AGRADECIMIENTOS

En primer lugar, quiero expresar mi profundo agradecimiento a Dios, quien ha sido mi guía y ha brindado la fuerza necesaria para afrontar los desafíos que se presentaron a lo largo de mi vida universitaria.

A mis queridos padres, Vicente y Rocío, les agradezco enormemente por su apoyo, cariño y comprensión incondicional en cada etapa de mi vida. A mis hermanos, Carolina y Joaquín, mi mayor motivación y el motivo por el cual siempre sigo adelante. Los quiero mucho y siempre estaré para ustedes.

A mis abuelos, Polivio y Margarita, y a mis tíos, Fabián, Fernando y Henry, les doy las gracias por su cariño y sabios consejos. Siempre han estado presentes para brindarme su apoyo y experiencia, lo cual ha sido de gran ayuda durante este proceso.

A mi amigo, compañero y colega, Jae Hyun, quiero expresar mi profundo agradecimiento por la dedicación, empeño y paciencia que ha puesto en este trabajo y en toda nuestra carrera universitaria. Eres el mejor compañero y amigo que pude haber tenido. También quiero agradecer a mi tutor, Ing. Julio Cesar Zambrano, por sus conocimientos y acompañamiento durante el desarrollo de este trabajo.

Por último, quiero agradecer a mis primos, Sebastián, Josue, Luis y Maite, y todos mis amigos, quienes han estado a mi lado en las buenas y en las malas. Gracias por ser mi apoyo incondicional, por comprender mis ausencias y por alentarme en cada paso hacia la meta final. Su amistad ha sido un regalo preciado que valoro profundamente.

A cada uno de ustedes, mi gratitud es infinita por formar parte de este logro. Su presencia en mi vida ha sido fundamental para cumplir esta meta.

Jaime Andrés Iñiguez Avila

DEDICATORIAS

Dedicatoria de Jae Hyun

Este trabajo va dedicado a mi hijo, mi tesoro, quien día a día me da fuerzas para seguir adelante, nunca rendirme y ser un buen ejemplo para él. A mi esposa, cuyo apoyo y amor incondicional motiva cada paso que doy, todos mis logros así como mi vida son tuyos. A mis padres, quienes con amor y empeño formaron las bases de la persona que soy ahora. A mis hermanas, que con su compañía han sabido iluminar mis días. A mi familia, cuya su preocupación y afecto agradezco profundamente. Y, finalmente, a todos mis amigos y personas que de una u otra forma han influenciado mi vida.

Dedicatoria de Jaime Andrés

Dedico este trabajo a mi familia, con un especial reconocimiento a mis padres, hermanos, abuelos y tíos. Con profunda gratitud, quiero expresar mi más sincero agradecimiento a cada uno de ustedes, quienes han sido mi motivación y mi soporte en cada paso del camino. Sus palabras de aliento, su infinito cariño y su incondicional compañía han sido el motor que me impulsó a superar cada desafío y alcanzar esta meta. Sin su respaldo, este logro no habría sido posible. Gracias por ser el pilar fundamental de mi vida, por estar siempre presentes y por creer en mí.

Índice general

Índice de figuras

Índice de tablas

Resumen

El presente trabajo se centra en la implementación de un sistema de monitoreo y control remoto de un motor de inducción trifásico comandado por un accionamiento SINAMICS G120, a través de una conexión inalámbrica entre un computador y un PLC por medio de la tecnología SCALANCE. Se ha desarrollado una interfaz de usuario (HMI) en Python que permite al usuario monitorizar información del motor en tiempo real, tal como la velocidad actual, la corriente, la potencia y el torque generado.

La investigación se basa en el diseño y desarrollo de una arquitectura inalámbrica eficiente y confiable, que garantice una conexión estable y segura entre uno o más usuarios remotos y el PLC. Además, se ha dado especial atención al diseño de la interfaz de usuario, buscando una experiencia intuitiva y amigable para el usuario final. El sistema HMI permite visualizar de manera clara y precisa los datos relevantes del motor, y en caso de errores o fallas, proporciona notificaciones visuales y auditivas para una pronta intervención y solución de problemas.

Palabras clave: HMI; Monitoreo; Motor Trifásico; Python; Scalance

Abstract

The present work focuses on the implementation of a remote monitoring and control system for a three-phase induction motor commanded by a SINAMICS G120 drive, through a wireless connection between a computer and a PLC using SCALANCE technology. A user interface (HMI) has been developed in Python that allows the user to monitor motor information in real time, such as current speed, current, power and torque generated.

The research is based on the design and development of an efficient and reliable wireless architecture, which guarantees a stable and secure connection between one or more remote users and the PLC. In addition, special attention has been given to the design of the user interface, looking for an intuitive and friendly experience for the end user. The HMI system allows for clear and accurate visualization of relevant motor data, and in case of errors or failures, provides visual and audible notifications for prompt intervention and troubleshooting.

Keywords: HMI; Monitoring; Python; Scalance; Trifasic motor

Antecedentes

El Controlador Lógico Programable (PLC, del inglés Programmable Logic Controller) fue creado en la década de 1960, revolucionando la industria de la automatización y experimentando desde entonces numerosas mejoras que han impulsado un progreso exponencial en el ámbito industrial. En la actualidad, los PLCs son indispensables en la industria y se consideran el núcleo central de cualquier proceso de automatización. Estos dispositivos permiten la comunicación e interacción entre sí, así como con otros elementos industriales, lo que proporciona una mayor interoperabilidad, flexibilidad y robustez en los sistemas de automatización.

El monitoreo en tiempo real de la maquinaria industrial surge como una necesidad para el seguimiento y la evaluación del sistema de automatización. Gracias al monitoreo se puede analizar las variables de un proceso. Todos los datos de interés pueden ser guardados en una base de datos. En complemento, la información también puede ser exportada y utilizada para diversos propósitos [1]. La recolección de los datos requeridos y el control de procesos se puede dar desde elementos específicos dentro de una red industrial como periferias descentralizadas y accionamientos para motores o servomotores [2]; hasta datos finales del proceso de producción que nos permita la evaluación de la eficiencia del mismo.

Actualmente, existen diversas tecnologías de comunicación para la industria, como por ejemplo, Profibus, Profinet, Modbus, entre otras. La mayoría de los sistemas de comunicación de esta naturaleza se implementan sobre medios físicos de cobre. No obstante, a nivel industrial, las redes cableadas no siempre son factibles, debido a diversos factores. Hoy en día, se pueden encontrar diversas tecnologías para

realizar comunicaciones inalámbricas a nivel industrial. Por ejemplo, para la marca SIEMENS existe la tecnología SCALACE W774/734, la cual permite comunicar inalámbricamente dispositivos industriales bajo una arquitectura cliente/servidor.

Estas tecnologías de comunicación aún no están siendo explotadas en toda su capacidad, sobre todo en nuestro medio, donde tenemos la necesidad urgente de innovar. Con este trabajo de titulación se pretende realizar un sistema de comunicación inalámbrico basado en la tecnología SCALACE W774/734, de manera que se pueda brindar una alternativa de comunicación para la industria. En adición y para ofrecer una solución de bajo costo, el sistema de comunicación se complementa con un sistema HMI desarrollado en Python para poder gestionar las variables y la operación de un motor trifásico de inducción el cual opera sobre una red industrial Profinet, a través de un accionamiento SINAMICS G120.

Para resaltar la importancia de este proyecto, se debe mencionar que las interfaces HMI se configuran mayormente mediante software propietario [3], lo cual limita la escalabilidad de los sistemas desarrollados. Por otra parte la inversión en licencias y plataforma de hardware es elevada. Por ello surge como alternativa el desarrollar dichas interfaces mediante código libre, evitando así adquirir licencias y equipos específicos que sean compatibles con el controlador implementado.

Justificación

El uso de PLCs en la industria ha experimentado un crecimiento considerable debido a su utilidad y eficacia en la automatización. Sin embargo, a medida que los procesos se vuelven más complejos y diversos, se requiere un mayor control, lo que implica la necesidad de interfaces HMI que permitan una mayor interacción entre los usuarios y los PLCs.

En el mercado, podemos encontrar una extensa gama de dispositivos de control, cada uno con sus propias características distintivas, lo que los hace óptimos según las necesidades del usuario. También existen diversos complementos que sirven de apoyo y mejoran la interacción humano-máquina. Por lo tanto, al automatizar procesos, es importante tener en cuenta no solo la marca del controlador, sino también el tipo y los complementos a utilizar.

Es importante destacar que la configuración tanto de los dispositivos de control como de los complementos, como el propio HMI, requiere el uso de un programa propietario compatible con dichos dispositivos. Esto implica que las empresas no solo deben adquirir la parte física, sino también el software necesario para estos equipos. Cabe mencionar que el valor de estos programas suele ser significativamente elevado, sin mencionar la necesidad de asegurar la compatibilidad entre dispositivos e incluso marcas para lograr una comunicación efectiva y sin problemas.

Este estudio tiene como objetivo abordar tres problemáticas existentes. En primer lugar, se propone el desarrollo de un HMI basado en código libre, lo que elimina la necesidad de adquirir un HMI comercial y su paquete de software para su configuración. En su lugar, se utilizará un programa basado en Python capaz

de comunicarse con el PLC a través de la computadora. En segundo lugar, dadas las condiciones industriales, en muchas ocasiones la comunicación entre el PLC y el HMI requiere que este último esté a cierta distancia para evitar riesgos tanto para los usuarios como para el propio HMI. Mediante la comunicación inalámbrica, se pueden evitar problemas de conectividad, interferencias o peligros inherentes al proceso industrial. Por último, al ser un código libre y capaz de funcionar en cualquier ordenador, brinda flexibilidad al sistema al no depender de marcas o equipos específicos. A través del protocolo de comunicación, el HMI puede funcionar independientemente de estas características.

Objetivos

Objetivo General

- Implementar un sistema de comunicación inalámbrico y HMI basado en PYTHON para la operación y monitoreo remoto de un motoro trifásico de inducción.

Objetivos específicos:

- Implementar el sistema de control para el motor trifásico de inducción empleando el variador de frecuencia SINAMICS PM240-2 bajo comunicación PROFINET.
- Implementar el sistema de comunicación inalámbrico entre el PLC SIMATIC S7-1500 y la estación de monitoreo remoto, utilizando la tecnología SCALANCE W774.
- Desarrollar un sistema HMI basado en PYTHON para la operación y monitoreo de un motor trifásico de inducción.

Introducción

En los procesos industriales actuales, el monitoreo y control eficiente de los actuadores es fundamental para garantizar un funcionamiento óptimo de los sistemas. Uno de los dispositivos más ampliamente utilizados en la industria son los motores de inducción trifásicos. La capacidad de supervisar y regular de forma remota el rendimiento de estos motores brinda beneficios significativos en términos de seguridad y calidad de los procesos en los que se emplean. Además, el control y monitoreo remotos brindan flexibilidad y robustez al permitir su supervisión en múltiples estaciones y la capacidad de ajustar los parámetros en caso de ser necesario.

El presente trabajo se centra en abordar el desafío de desarrollar una solución de monitoreo y control de motores a través de una conexión inalámbrica entre un computador y un Controlador Lógico Programable (PLC). Se ha desarrollado una interfaz en Python que no requiere licencias, lo que permite a las empresas y usuarios un acceso libre y un ahorro en programas y componentes propietarios, como las pantallas HMI y sus respectivos programas.

La tecnología SCALANCE W700 se utiliza como base para establecer la conexión inalámbrica, proporcionando una solución confiable y segura para la transmisión de datos entre el computador y el PLC. La interfaz desarrollada permite al usuario controlar aspectos clave del motor, como la velocidad, el sentido de giro y el estado. Además, ofrece información detallada en tiempo real sobre la corriente, potencia, velocidad actual y torque del motor. En caso de que surja algún error en el motor o el controlador, se activará una alarma correspondiente para informar a los usuarios y realizar las acciones necesarias para corregirlo.

El objetivo principal de este trabajo es contribuir al campo de la automatización industrial, ofreciendo una solución innovadora, práctica y de bajo costo para el

monitoreo y control de motores a través de una conexión inalámbrica. Los resultados obtenidos en esta investigación proporcionarán a las empresas e industrias una herramienta valiosa para mejorar la eficiencia operativa, reducir costos y minimizar los tiempos de inactividad.

En los siguientes capítulos, se establecerá el marco teórico necesario para el desarrollo de la investigación. El Capítulo 1 aborda la información de la tecnología SCALANCE y su configuración, mientras que en el Capítulo 2 se explora la comunicación mediante Python y se describe el diseño de la interfaz. Posteriormente, en el Capítulo 3, se lleva a cabo la implementación del sistema de control y comunicación inalámbrica, basándose en los puntos anteriores, y se obtienen los resultados correspondientes. El capítulo 4 cuenta con el diseño HMI y las pruebas de funcionamiento. Finalmente, se presentan las conclusiones derivadas de los resultados obtenidos, resumiendo los logros alcanzados, discutiendo las limitaciones encontradas y proponiendo posibles direcciones para futuras investigaciones.

Capítulo 1

La tecnología SCALANCE W700

La implementación de una comunicación inalámbrica confiable y de alto rendimiento es de vital importancia en los entornos industriales actuales, ya que garantiza que las plantas y aplicaciones estén listas para enfrentar los desafíos futuros y se encuentren en una posición óptima para aprovechar plenamente las ventajas de las innovadoras tecnologías incorporadas en el campo de la automatización.

Las conexiones LAN inalámbricas permiten que las máquinas y dispositivos se muevan libremente, al tiempo que garantizan la seguridad de las personas. Esto resulta rentable desde la fase de planificación de plantas y procesos y proporciona beneficios en las operaciones de todas las industrias. Sin embargo, las necesidades industriales de las redes de comunicación inalámbrica van más allá de las capacidades de las WLAN comerciales estándar. Por lo tanto, es necesario integrar protocolos industriales especiales como PROFINET, PROFIsafe y EtherNet/IP para permitir la comunicación entre dispositivos de campo [4]. La LAN inalámbrica industrial (IWLAN) ha sido concebida como la solución idónea para cumplir con las exigentes demandas del entorno industrial. Se enfoca en brindar comunicación en tiempo real, alta redundancia y transmisión de datos de alta disponibilidad. Los equipos SCALANCE W700 de Siemens son ampliamente utilizados dentro de estas redes y han sido especialmente diseñados para adaptarse a las necesidades de los entornos tanto interiores como exteriores de las redes IWLAN. Estos dispositivos son altamente confiables y eficientes, permitiendo una conectividad sólida y estable en entornos industriales exigentes.

Esta solución ofrece un amplio soporte y proporciona soluciones específicas en diversos sectores de la industria. Comparado con las tecnologías cableadas para interconectar máquinas o dispositivos, el enfoque inalámbrico destaca por su mayor movilidad, flexibilidad y un menor costo tanto de instalación como de mantenimiento [5].

En este capítulo se analizan algunos aspectos fundamentales de la tecnología SCALANCE W774/W734. En primer lugar, se describen las características generales de estos dispositivos. A continuación, se presentan las herramientas utilizadas para la configuración de los mismos. Por último, se aborda el tema de la comunicación Cliente/Servidor empleada en las redes inalámbricas IWLAN.

1.1. Características generales de los dispositivos

Los equipos SCALANCE son dispositivos de red diseñados específicamente para el sector industrial. En particular, la tecnología SCALANCE W700 ha desarrollado equipos especializados para la comunicación inalámbrica enfocada en el ámbito de la automatización industrial.

Es importante destacar que los equipos SCALANCE W700 están fabricados para satisfacer las exigentes demandas del entorno industrial, donde la robustez y confiabilidad son requisitos fundamentales. Según [6], los principales beneficios que nos otorga la implementación de estos dispositivos son:

- Precisión y fiabilidad: Los dispositivos SCALANCE W700 permiten una transmisión rápida y confiable de datos en tiempo real mediante protocolos como PROFINET y EtherNet/IP. Además, ofrecen comunicación redundante para evitar pérdida de datos y cuentan con herramientas de diagnóstico integradas que permiten la identificación y corrección de errores de manera eficiente.

- Comunicación segura: La seguridad y privacidad de los datos son prioritarias dentro de las comunicaciones inalámbricas. Los dispositivos SCALANCE W700

utilizan protocolos estándar y encriptación para garantizar la seguridad de los datos transmitidos.

- Robustos y flexibles: Los dispositivos SCALANCE W700 con protección IP65 son ideales para su instalación en entornos industriales desafiantes, como naves de producción o ambientes al aire libre. Su clasificación de protección IP65 asegura que puedan resistir condiciones ambientales adversas y garantiza su fiabilidad y adaptabilidad en entornos industriales exigentes.

- Configuración sencilla: El sistema de gestión de redes SINEC NMS ofrece una solución eficiente para el diseño y configuración centralizada de redes inalámbricas. Permite una fácil ampliación de la red al agregar componentes adicionales y ofrece supervisión continua para prevenir errores.

1.1.1. SCALANCE W774

El SCALANCE W774 (figura 1.1) es un dispositivo de comunicación inalámbrica que actúa como un AP (Access Point). Su función principal es establecer redes inalámbricas entre equipos que no cuentan con esta capacidad de comunicación incorporada. Al actuar como un AP, el SCALANCE W774 crea una red WLAN (Wireless Local Area Network), lo que proporciona un punto de acceso para la comunicación con dispositivos en redes LAN (Local Area Network) [6].

Además de su función como AP, el SCALANCE W774 implementa tecnologías avanzadas para garantizar una comunicación fiable en entornos industriales. Estas tecnologías permiten el desarrollo de redes IWLAN (Industria Wireless Local Area Network), que se adaptan a las necesidades específicas de comunicación en entornos industriales [7].

Es importante destacar que un dispositivo SCALANCE W774 puede funcionar tanto como punto de acceso como cliente, dependiendo de los requisitos de la aplicación. Esta flexibilidad permite su adaptación a diferentes escenarios de red, brindando soluciones versátiles y eficientes en entornos industriales.

Figura 1.1: Módulo SCALANCE W774-1 RJ45 [8]

1.1.2. SCALANCE W734

El SCALANCE W734 (figura 1.2) es un módulo cliente que se utiliza para establecer conexiones en redes inalámbricas. A diferencia del SCALANCE W774, el SCALANCE W734 busca un suscriptor o punto de acceso al que pueda conectarse utilizando la configuración de SSID (Servicie Set Identifier) o nombre de red correspondiente [6], [9].

El SCALANCE W734 es compatible con el estándar IEEE 802.11 a/b/g/n, lo que le permite trabajar con diferentes tecnologías inalámbricas para garantizar una conectividad robusta y confiable en entornos industriales exigentes.

Figura 1.2: Módulo SCALANCE W734-1 RJ45 [8]

1.1.3. Estructura y funcionalidad de los dispositivos

A pesar de tener diferentes aplicaciones dentro de una red IWLAN, el SCALANCE W774 y el SCALANCE 734 comparten características estructurales similares, las cuales pueden visualizarse en la figura 1.3. Según [10] ambos dispositivos cuentan con:

- Dos conectores hembra R-SMA que permiten la conexión de antenas externas, proporcionando flexibilidad y opciones de mejora de la señal inalámbrica.
- Dos puertos RJ45 que admiten velocidades de transferencia de datos de 10/100 Mbits/s. Una de estas conexiones es compatible con Power over Ethernet (PoE).
- Dos conexiones de 24 VDC para una alimentación redundante.
- Un slot (KEY-PLUG/C-PLUG) para la inserción de un PLUG para funciones adicionales, como el respaldo de datos de configuración.
- LEDs de función que proporcionan una señalización óptica de fallos y estados de funcionamiento. Estos LEDs permiten una rápida visualización y diagnóstico de la condición del dispositivo.

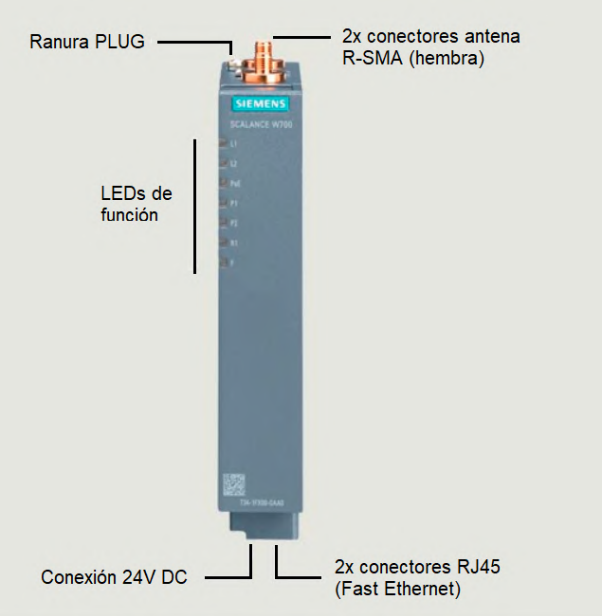


Figura 1.3: Estructura de los módulos SCALANCE W774 y SCALANCE W734

Los puertos RJ45 de los dispositivos SCALANCE W774/W734 son compatibles con los estándares IEEE 802.11a/b/g/h/n, lo que permite una amplia compatibilidad con dispositivos y redes inalámbricas existentes. El dispositivo opera en frecuencias de 2,4 y 5 GHz, lo que proporciona flexibilidad en la elección de la banda de frecuencia adecuada para la comunicación inalámbrica [11].

En cuanto a la velocidad de transferencia de datos, el SCALANCE W774 puede alcanzar una velocidad bruta de hasta 300 Mbits/s, lo que permite una comunicación rápida y eficiente en la red inalámbrica. Mientras que los puertos RJ45 trabajan con una velocidad máxima de 100 Mbits/s, lo que proporciona la opción de una conexión cableada adicional si es necesario [7].

El dispositivo también es compatible con Power over Ethernet (PoE), una tecnología que permite suministrar corriente eléctrica a dispositivos a través del mismo cableado de red Ethernet utilizado para la transmisión de datos. Esto elimina la necesidad de utilizar fuentes de alimentación externas y tomas de corriente adicionales para alimentar los dispositivos [12]. Además, la carcasa con grado de protección IP30 garantiza la protección contra el polvo y objetos pequeños, lo que lo hace adecuado para entornos industriales con condiciones adversas [9].

En términos de temperatura de funcionamiento, el SCALANCE W774 puede operar en un rango amplio, el cual va desde -20 hasta 60 °C. Esto permite su despliegue en entornos industriales en donde se pueden experimentar condiciones ambientales extremas.

1.1.4. Antena ANT795-4MA

Las antenas omnidireccionales ANT795-4MA (figura 1.4) son antenas diseñadas específicamente para su uso en redes inalámbricas industriales (IWLAN). Estas antenas ofrecen una solución de alta calidad para optimizar la cobertura y la calidad de la señal inalámbrica en entornos industriales desafiantes [11].

La ANT795-4MA proporciona una ganancia de 3/5 dBi, lo que contribuye a una mayor recepción y transmisión de señales inalámbricas. Está especialmente diseñada para operar en las frecuencias de 2.4 GHz y 5 GHz, lo que habilita su funcionamiento en redes WLAN de doble banda. [13].

La ANT795-4MA se puede montar directamente en dispositivos SCALANCE W774 Y SCALANCE W734 utilizando el sistema de conexión RSMA especificada en la figura 1.3.

Figura 1.4: Antena Omnidireccional ANT795-4MA [13]

1.2. Herramientas de configuración

Los módulos SCALANCE W774/W734, al igual que cualquier dispositivo de red inalámbrica (WLAN), necesitan ser configurados antes de poder utilizarse en un entorno industrial. La configuración es esencial para establecer los parámetros de red, como la dirección IP, la seguridad y otros ajustes específicos.

Para llevar a cabo esta configuración, se dispone de diversas alternativas de software que permiten realizar tanto configuraciones iniciales como avanzadas según los requisitos de los módulos. A continuación, se presentan algunas opciones de software disponibles.

1.2.1. SINEC PNI

SINEC PNI (Primary Network Initialization) es un software empleado para la detección e inicialización de dispositivos PROFINET, así como para la configuración básica de dispositivos SCALANCE, RUGGEDCOM y RTLS [14].

Además de proporcionar información sobre direcciones IP, SINEC PNI permite configurar parámetros específicos de PROFINET y PROFIBUS, así como las

credenciales de los dispositivos.

SINEC PNI ofrece diversas funciones para la configuración y gestión de dispositivos PROFINET y RUGGEDCOM. Mediante esta herramienta, es posible escanear la red y detectar todos los dispositivos PROFINET y RUGGEDCOM presentes. Además, permite realizar la inicialización de los dispositivos ajustando parámetros como la dirección IP, subred, pasarela y contraseña inicial. También es posible configurar información específica del dispositivo, como el nombre PROFINET y los datos IM de identificación. Adicionalmente, el programa facilita la activación del cliente DHCP, el restablecimiento a los valores de fábrica y a los valores estándar PROFINET, así como realizar pruebas de conectividad mediante el uso de ping. Adicionalmente, se puede controlar el parpadeo de los LEDs y acceder al "Web Based Management"de los dispositivos para una configuración adicional. Estas funciones permiten una gestión eficiente y centralizada de los dispositivos en la red [15].

En la figura 1.5 se muestra la interfaz principal de SINEC PNI, donde se pueden observar todos los elementos que la componen. Por otro lado, la Tabla 1.1 proporciona una descripción detallada de cada sección presente en la interfaz.

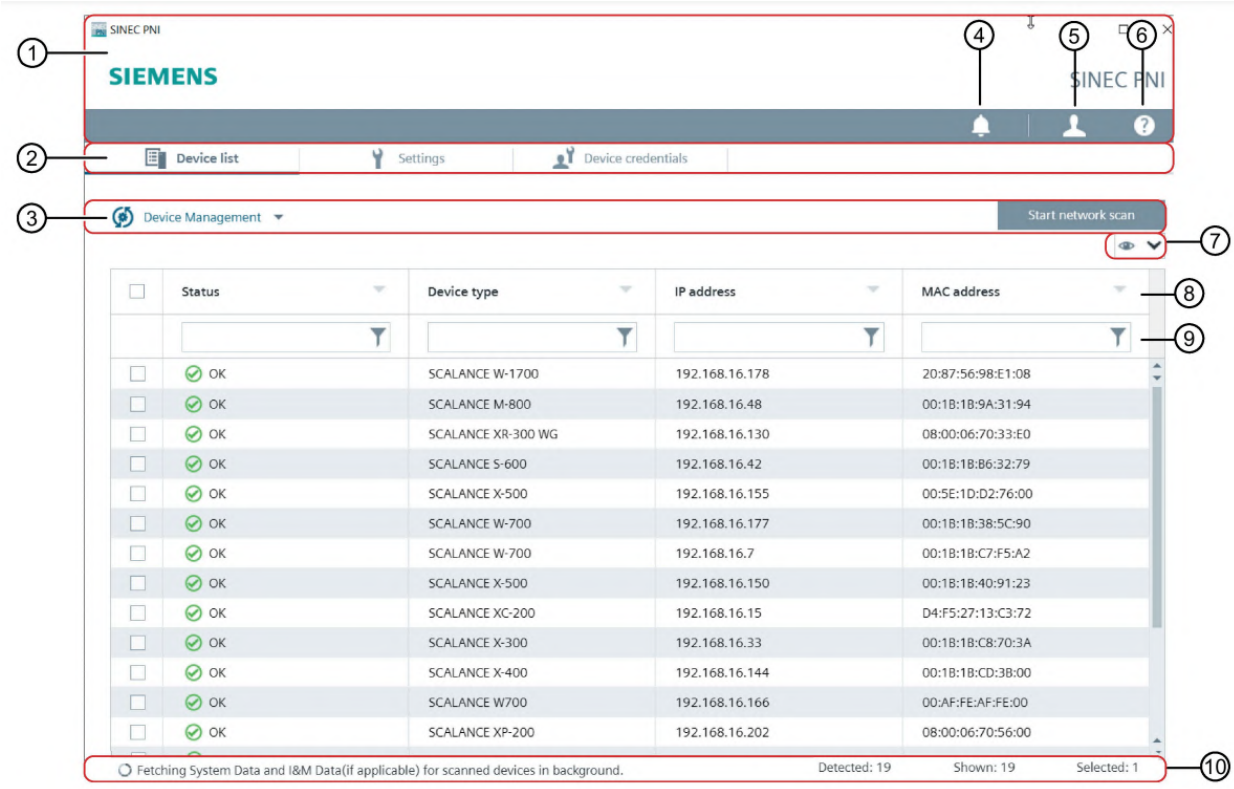


Figura 1.5: Interfaz software Sinec PNI [14]

Tabla 1.1: Descripción de interfaz software Sinec PNI [14].

No.	Descripción
1	Encabezamiento
2	Navegación
3	Menú de acciones y botones de acceso rápido
4	Menú de notificaciones
5	Menú de usuario
6	Menú de ayuda e información de software
7	Lista desplegable para selección de columnas
8	Iconos de clasificación
9	Cuadro de entrada para filtrar selección
10	Barra de estado e información del sistema

SINEC PNI requiere de programas complementarios para su funcionamiento adecuado. Estos programas adicionales están relacionados con aspectos específicos de sus funciones, como la administración del tráfico de red y la ejecución del programa. Por lo tanto, es necesario contar con estos programas complementarios para garantizar el correcto funcionamiento y aprovechamiento de todas las capacidades de SINEC PNI.

PCAP DRIVER

WinPcap es una biblioteca y conjunto de controladores utilizados en sistemas operativos Windows para capturar y analizar paquetes de red en tiempo real. Proporciona acceso de bajo nivel a la red y facilita el acceso a las capas de red subyacentes. Es ampliamente utilizado por aplicaciones que requieren el monitoreo y análisis del tráfico de red [16]. Esta herramienta permite a los desarrolladores crear aplicaciones de red capaces de interactuar directamente con la capa de enlace de datos de la pila de red de Windows. Esto incluye la captura de paquetes en redes Ethernet, la extracción de datos de los paquetes capturados y la implementación de funcionalidades avanzadas de análisis de tráfico.

Visual C++

Visual C++ es un entorno de desarrollo integrado (IDE) y un conjunto de herramientas empleado para programar en el lenguaje de programación C++. Forma parte de la suite de productos de Microsoft Visual Studio y proporciona una interfaz

de usuario gráfica y funcionalidades para desarrollar aplicaciones en C++ de forma eficiente. La instalación de los paquetes de Visual C++ garantiza la presencia del entorno de tiempo de ejecución necesario en el sistema, lo que posibilita la ejecución de aplicaciones desarrolladas sin la necesidad de tener instalado Visual Studio en sí. Esta facilidad simplifica la distribución y ejecución de aplicaciones desarrolladas con Visual C++ en diferentes dispositivos [17], lo que permite la correcta ejecución de SINEC PNI en el ordenador.

1.2.2. Web Based Management

El Web Based Management (WBM) es una interfaz basada en hipertexto que se encuentra integrada en dispositivos como concentradores, switches, enrutadores o puntos de acceso inalámbricos. Esta interfaz permite a los usuarios administrar y configurar el dispositivo desde cualquier ubicación en la red utilizando un navegador web estándar [18].

La ventaja del WBM radica en su capacidad para ofrecer una interfaz amigable e intuitiva que permite a los usuarios realizar diversas tareas de administración, como configurar parámetros de red, administrar la seguridad, monitorear el rendimiento y realizar diagnósticos. Al utilizar el protocolo HTTP, el navegador web establece una comunicación directa con el dispositivo, lo que facilita la administración remota sin la necesidad de instalar software adicional en el equipo del usuario [19].

Antes de acceder a la configuración de los dispositivos SCALANCE W774/W734, es imprescindible realizar la configuración inicial utilizando SINEC PNI. Estos equipos vienen de fábrica con la opción de DHCP y DCP habilitada, lo que significa que es necesario asignar una dirección IP para acceder a su interfaz de configuración [20]. Al realizar la configuración inicial con SINEC PNI, se podrá establecer una dirección IP adecuada y personalizar otros ajustes según los requerimientos de la red mediante WBM.

En la figura 1.6 podemos visualizar la interfaz de WBM del módulo SCALANCE W774 empleado para el presente proyecto.

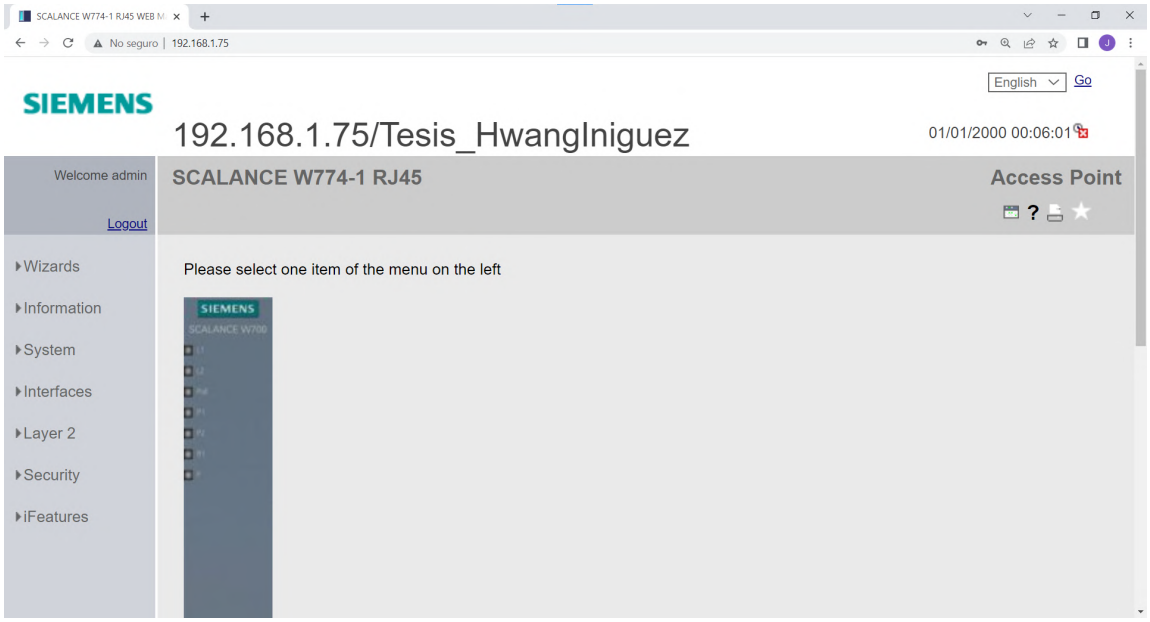


Figura 1.6: Interfaz WBM Siemens

1.3. Comunicación Cliente/Servidor

La comunicación Cliente/Servidor es una tecnología que ofrece soluciones a diversos problemas de gestión de datos. Este enfoque se basa en la división de tareas entre dos entidades autónomas e independientes: el Cliente y el Servidor [21]; cuya arquitectura se presenta en la figura 1.7

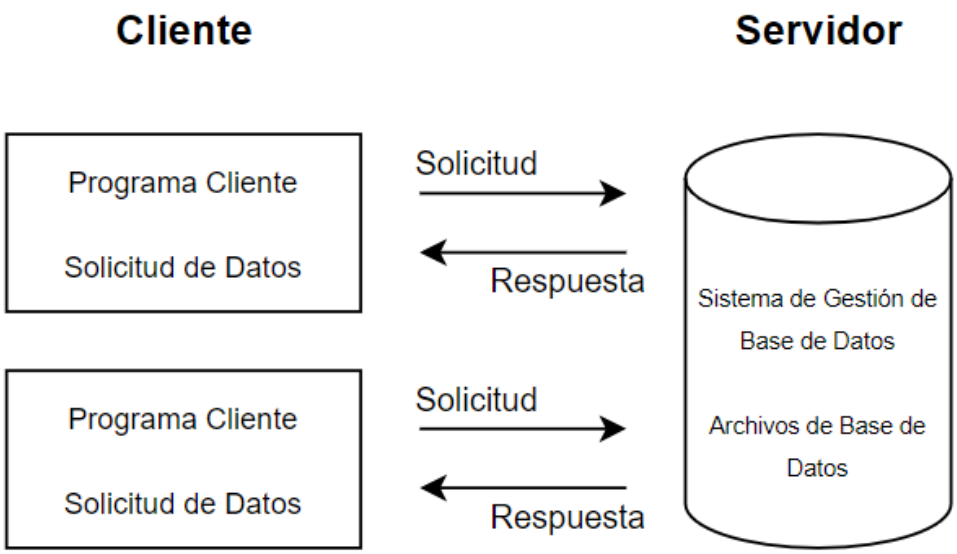


Figura 1.7: Arquitectura Cliente-Servidor [22].

El Cliente hace referencia a un dispositivo o proceso que solicita y consume servicios o recursos proporcionados por el Servidor. Puede ser una aplicación, un sistema embebido o cualquier otro componente que inicie la comunicación y envíe solicitudes al Servidor. El Cliente es responsable de enviar peticiones al Servidor y recibir las respuestas correspondientes [22]

Por otro lado, el Servidor se trata de un dispositivo o proceso que responde a las solicitudes enviadas por los Clientes y ofrece los servicios o recursos requeridos. Puede tratarse de un servidor web, un servidor de bases de datos o cualquier otro sistema diseñado para recibir y procesar solicitudes de los Clientes. El Servidor es responsable de recibir las peticiones, realizar las acciones necesarias y enviar las respuestas de vuelta al Cliente [23].

Esta arquitectura es ampliamente empleada en entornos industriales para administrar y controlar procesos, sistemas de automatización, monitoreo de equipos y recopilación de datos.

En estos entornos, los Clientes generalmente son dispositivos o sistemas de control como PLC, HMI o estaciones de monitoreo, quienes envían requerimientos a los Servidores para obtener datos o realizar acciones específicas. Los Servidores son dispositivos que almacenan y gestionan la información solicitada por los Clientes, y pueden ser controladores de procesos, bases de datos, sistemas de almacenamiento u otros equipos que brinden los servicios requeridos.

La comunicación entre Clientes y Servidores en el ámbito industrial se lleva a cabo mediante protocolos y tecnologías de red como Ethernet o TCP/IP. Estos protocolos permiten la transmisión de datos y la interacción entre los Clientes y los Servidores, lo que facilita el control y el monitoreo de los procesos industriales.

El modelo Cliente/Servidor en el ámbito industrial ofrece flexibilidad, escalabilidad y eficiencia en la gestión de sistemas y procesos. Permite la descentralización de la información y la distribución de tareas entre los dispositivos, lo que facilita el control y el monitoreo de los procesos industriales de manera más eficiente y segura.

Capítulo 2

Comunicación con PYTHON

En la actualidad, Python se ha consolidado como un lenguaje de programación versátil y eficaz en varios ámbitos, destacándose especialmente en la comunicación en redes industriales y en la creación de interfaces hombre-máquina (HMI) destinadas al control de procesos en entornos industriales. Gracias a la amplia bibliografía, foros y documentación de librerías y del propio lenguaje, es posible obtener una comprensión profunda de las herramientas y técnicas necesarias para establecer una comunicación efectiva entre dispositivos industriales y aplicaciones basadas en Python. Estos recursos proporcionan una visión integral de la utilización de bibliotecas especializadas, como matplotlib y NumPy, entre otras, que permiten manipular y analizar datos obtenidos en procesos industriales. Esto resulta fundamental para el desarrollo de interfaces de control intuitivas y eficientes. Además, la disponibilidad de documentación exhaustiva, ejemplos prácticos y otros elementos didácticos facilita la aplicabilidad de dicho lenguaje, lo que permite a los usuarios aplicar y desarrollar programas para la implementación de sistemas de control y supervisión en tiempo real de manera más sencilla. De este modo, se aprovechan al máximo las características que Python ofrece en el ámbito industrial. A continuación, se especificará con mayor detalle el uso de Python para la comunicación en redes industriales y la creación de HMIs, brindando una solución potente y versátil para el control de procesos en entornos industriales.

2.1. Introducción al lenguaje de programación

Python es un lenguaje de programación de alto nivel, interpretado y de propósito general que se distingue por su simplicidad, legibilidad y flexibilidad. Surgió a finales de la década de 1980 con el propósito de ser un lenguaje accesible y fácilmente aplicable, manteniendo su capacidad para abordar problemas complejos. Sin embargo, no fue hasta 1991 cuando Python resaltaría como uno de los lenguajes interpretados más populares. Desde entonces, su relevancia ha crecido aún más, especialmente a partir de 2005, cuando comenzó a emplearse ampliamente para la construcción de sitios web utilizando los numerosos frameworks disponibles para dicha tarea [24].

Python se caracteriza por su enfoque como "lenguaje de scripting", lo que significa que su formato principal de desarrollo se basa en la creación de scripts, programas generados en líneas de código implementables en consola, cuya finalidad es automatizar tareas y actividades al ejecutarse. No obstante, a medida que ha evolucionado, gracias a la comunidad activa que desarrolla nuevas librerías y aplicaciones, Python ha alcanzado un destacado reconocimiento y se ha convertido en uno de los lenguajes más ampliamente utilizados en el desarrollo de software. En el contexto académico, Python ha demostrado su utilidad en diversas disciplinas debido a la flexibilidad y amplia documentación. Su sintaxis clara y legible facilita la comprensión de los algoritmos y programas desarrollados, mientras que su amplia biblioteca estándar y la disponibilidad de numerosas librerías adicionales proporcionan una variedad de herramientas para abordar problemas específicos.

Python se ha vuelto relevante y ampliamente empleado en la actualidad debido a sus características y beneficios distintivos en comparación con otros lenguajes de programación similares. Según [25], las principales ventajas de Python son las siguientes:

- Orientación a objetos: Python soporta la programación orientada a objetos, lo que permite desarrollar código más eficiente y práctico. Esta característica facilita la reutilización de código previamente escrito y, en conjunto con su compatibilidad con la programación orientada a objetos, convierte a Python en

una herramienta extremadamente útil.

- Gratuito: Al igual que muchos otros lenguajes, Python es de libre acceso, lo que significa que cualquier persona puede utilizarlo sin restricciones. Además, debido a la distribución libre de código desarrollado en Python y a la activa comunidad de programadores, existe una amplia cantidad de foros y documentación que facilita el aprendizaje, la aplicación y la corrección de errores.

- Portabilidad: Python es un lenguaje altamente flexible y está ampliamente disponible en diversas plataformas virtuales como Windows, Linux, Mac y Android, entre otras.

- Potencia: Python combina la simplicidad y facilidad de uso propias de los lenguajes de programación con herramientas avanzadas de ingeniería de software, que generalmente se encuentran en lenguajes compilados.

- Integración con otros lenguajes: Python se puede integrar fácilmente con programas desarrollados en otros lenguajes, lo que amplía la variedad de aplicaciones disponibles y permite ahorrar tiempo al evitar la necesidad de replicar o traducir programas a Python.

- Facilidad de uso: Python cuenta con una sintaxis clara y comprensible, incluso si el programa fue desarrollado por otra persona. Además, no requiere compiladores ni terceros para ejecutar el programa, lo que ahorra tiempo al depurar o modificar el código, ya que se evitan los tiempos de carga y se ejecuta directamente el script desarrollado.

- Facilidad de aprendizaje: Gracias a la abundante documentación disponible, la curva de aprendizaje de Python no es pronunciada. Esto permite a los nuevos usuarios desarrollar programas significativos en cuestión de horas o, como máximo, días. Si bien Python puede llegar a ser un lenguaje complejo debido a su variedad de librerías y herramientas avanzadas, su núcleo básico destaca precisamente por su simplicidad y facilidad de aprendizaje.

2.2. Herramientas para la creación de sistemas HMI

Para la creación de sistemas HMI basados en Python, existen varias herramientas que permiten implementar elementos gráficos, como ventanas, botones y cuadros de texto, que luego se traducen a código o se importan desde el código principal de Python.

A través del código principal, se configuran los elementos gráficos mencionados anteriormente y se les enlaza con las acciones y funciones necesarias para el correcto funcionamiento del programa. Es decir, para que los botones u otros elementos gráficos colocados en las herramientas de diseño cumplan con su funcionalidad, es necesario implementar el código que haga uso de los eventos y parámetros de los elementos configurados [26].

Entre las herramientas destacadas para la creación de interfaces gráficas en Python se encuentran Tkinter, PyQt y PySide, siendo las más conocidas, aunque existen muchas otras. Cada una de estas herramientas tiene puntos fuertes y características que permiten satisfacer diferentes necesidades según los requisitos del usuario.

Tkinter es la más sencilla de todas, con un aprendizaje fácil e implementación sencilla. Debido a su nivel de dificultad y alta portabilidad al ser compatible con la mayoría de los sistemas operativos, Tkinter es ideal para aplicaciones medianas y pequeñas, donde la simplicidad y la eficacia son importantes. Otra ventaja de Tkinter es su manejo de la biblioteca y su compatibilidad con Python, lo que implica una instalación y configuración simple sin importar la versión o el sistema operativo utilizado [27].

Sin embargo, a medida que las necesidades y características de la interfaz se vuelven más complejas, la eficacia de Tkinter se ve limitada por su enfoque en la facilidad de uso y aprendizaje. Por esta razón, han surgido otras alternativas con funcionalidades más avanzadas, como PyQt y PySide.

Ambas herramientas comparten el framework base QT Designer, donde se colocan los elementos y se configuran sus características para luego generar un archivo .ui, que puede traducirse a Python utilizando herramientas como pyuics.

El framework de ambas herramientas se basa en la programación con C++,

por lo que ambas ofrecen una API similar. Sin embargo, las principales diferencias radican en la licencia. PyQt está disponible en las versiones PyQt4 y PyQt5, ambas con una licencia GPL. Esto significa que son gratuitas siempre y cuando las aplicaciones se distribuyan bajo la licencia GPL. Sin embargo, PyQt5 también ofrece una licencia gratuita LGPL para versiones comerciales o de pago. Por otro lado, PySide es de código abierto y utiliza una licencia LGPL, lo que permite su uso gratuito tanto para aplicaciones GPL como comerciales [28].

Debido a su compatibilidad y a la amplia cantidad de librerías desarrolladas por la comunidad, tanto PySide como PyQt5 destacan como herramientas que permiten crear interfaces con apariencia y funciones avanzadas. Sin embargo, para este proyecto, se prefiere PyQt5 debido a su compatibilidad, facilidad de uso y documentación existente sobre dichas librerías.

Para este proyecto en particular, las librerías externas que permiten visualizar elementos compatibles con PyQt5 son AnalogGaugeWidget y Matplotlib. Se utilizarán para crear gráficas circulares y plots de los valores del motor respectivamente. A continuación, se presentan ejemplos de uso de estas librerías en las figuras 2.1 y 2.2.

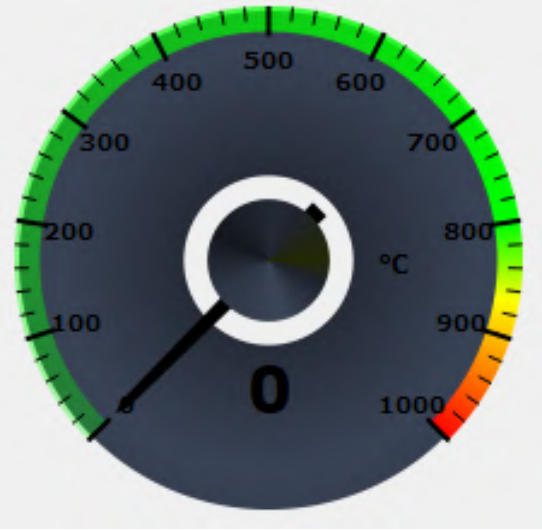


Figura 2.1: Ejemplo de uso de la librería Analoggaugewidget

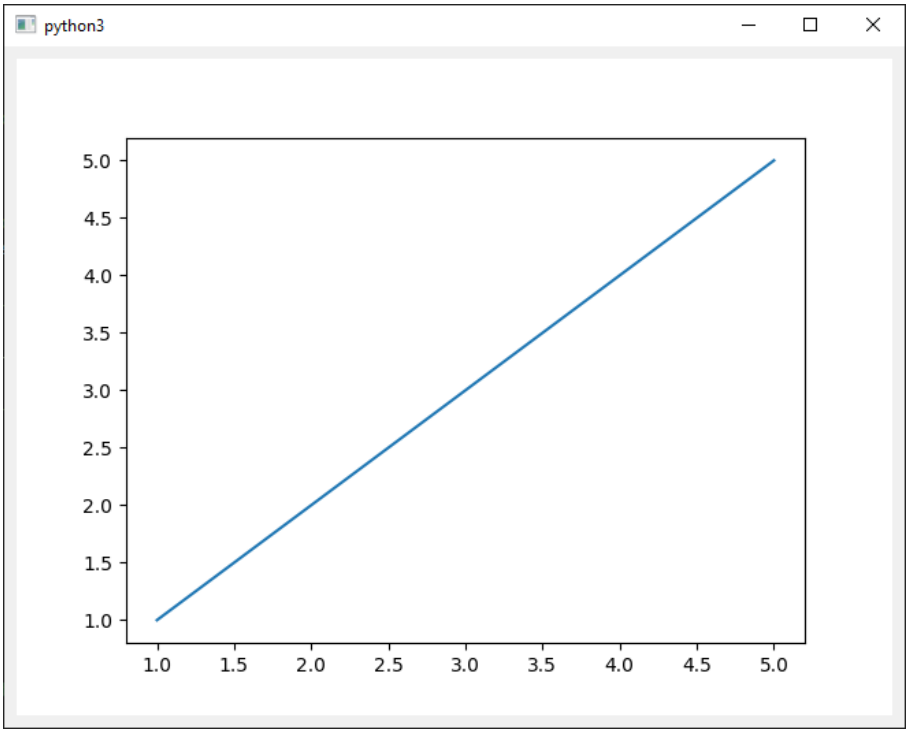


Figura 2.2: Ejemplo de uso de la librería MatPlotLib

2.3. Comunicación con dispositivos industriales

Como se trató en el capitulo anterior, La conexión entre el PLC y el ordenador se realiza mediante la tecnología SCALANCE W774, la cual actúa como el servidor, mientras que el cliente es el propio ordenador.

Una vez establecida la conexión entre el PLC y el ordenador, es posible intercambiar información como si estuvieran conectados a través de Profinet. Sin embargo, para lograr esto, el programa cargado en el PLC debe tener habilitada la lectura y escritura de datos, y debe contar con un DB (Data Block), que es una estructura de datos utilizada para el intercambio de datos entre diferentes partes del programa de control. El DB contiene variables y datos que son accesibles y compartidos por múltiples bloques de función (FB) o bloques de organización (OB) en un proyecto de TIA Portal.

En TIA Portal, se pueden crear y configurar los DB utilizando el entorno de programación STEP 7. Esto implica definir las variables y los tipos de datos que se almacenarán en el DB, así como establecer las conexiones y referencias necesarias para el intercambio de datos con otros bloques. La creación de un DB permite una mejor

gestión y control de las variables utilizadas, lo que contribuye a una estructura de programación ordenada y comprensible. En este caso, el DB se utiliza principalmente para acceder de manera remota a las variables a través de las funciones de la librería Snap7 [29].

La librería Snap7 es de código abierto y fue desarrollada con el fin de permitir la comunicación con los controladores de la serie SIMATIC S7 de Siemens. Proporciona funciones y métodos para acceder y controlar los datos de los controladores S7 de Siemens desde diferentes plataformas y lenguajes de programación.

Para el desarrollo de la interfaz gráfica en Python, es necesario utilizar una versión de la librería Snap7 compatible con ese lenguaje. Como se mencionó anteriormente, esta librería permite la lectura de datos de los PLC de Siemens. Sin embargo, para una mejor organización y acceso a la información, se utiliza las funciones "db_readz "db_write". Estas funciones, como su nombre indica, permiten la lectura y escritura directa de datos en los bloques de datos del PLC. Al utilizar estas funciones, se mantiene aislado el acceso a la memoria interna del PLC, lo que fortalece la seguridad y evita la sobreescritura y los fallos de memoria [30].

Para establecer la conexión mediante Snap7, se requieren ciertos parámetros: la dirección IP asignada al PLC, el rack donde se encuentra dicho componente y el slot donde se encuentra la CPU. Estos parámetros se proporcionan a un objeto de tipo "snap7.client.Client().[en] el parámetro çonnect()", como se muestra en la figura 2.3.

Es relevante considerar que el objeto de tipo "snap7.client.Client()"se crea para representar la conexión además de proporciona métodos y funciones para interactuar con el PLC. Como se puede apreciar en la figura 2.3, se enlaza a un PLC ubicado en el rack 0, slot 1, de dirección IP: "192.168.1.1".

```
PLC_Ip='192.168.1.1'
PLC_Rack=0
PLC_Slot=1
plc = snap7.client.Client()
plc.connect(PLC_Ip,PLC_Rack,PLC_Slot)
```

Figura 2.3: Ejemplo de conexión con el PLC desde python con la librería Snap7

Una vez establecida la conexión, se pueden utilizar varias funciones para extraer información, como el tipo de PLC o el estado de la conexión. Es posible acceder a la memoria interna del programador a través de diferentes áreas, lo que permite leer diferentes tipos de información, como DB, MK, CT, entre otros. Sin embargo, en este caso particular, solo es de interés acceder a los Data Blocks (DB) para mantener la integridad de la información almacenada en el programa del PLC y precautelarla de posibles fallos.

Para acceder a los Data Blocks, se utilizan los comandos "db_readz "db_write". Estos comandos requieren proporcionar el número del DB al que se desea acceder, el byte de inicio para la lectura-escritura, y la cantidad de bytes que se desean leer o escribir. Por ejemplo, en la figura 2.4, se muestra un ejemplo de lectura de datos donde se accede a un valor presumiblemente entero, debido a la cantidad de bytes leídos, ya que los valores enteros o las palabras "word"necesitan 2 bytes. En este caso, se accede al valor ubicado desde el byte 0 hasta el byte 2.

```
db_number=1
start_byte=0
num_bytes=2
r_int=plc.db_read(db_number, start_byte, num_bytes)
```

Figura 2.4: Ejemplo de lectura de datos en un DB con la librería Snap7

En cuanto a la escritura en un bloque de datos, es necesario obtener un bytearray con la información que se va a escribir. Para facilitar este proceso, Snap7 proporciona los comandos "set_bool", "set_int", "set_realz "set_string". Estos comandos permiten establecer nuevos valores para las variables dentro del bloque de datos.

Para utilizar los comandos mencionados, es necesario proporcionar un dato previamente leído utilizando el comando "db_read", la posición de la variable dentro del bloque de datos y el nuevo valor que se desea escribir. Una vez establecido el nuevo valor, se utiliza el comando "db_write"para escribir el dato en el bloque de datos correspondiente. Los parámetros necesarios para el comando "db_write"son el número del bloque de datos (DB), el byte de inicio y el bytearray por escribir.

En la figura 2.5, se muestra un ejemplo en el que se establece un nuevo valor de 15 para una variable previamente leída utilizando el comando"db_read", y luego

se escribe este nuevo valor en el bloque de datos utilizando el comando "db_write".

```
db_number=1
start_byte=0
num_bytes=2
r_int=plc.db_read(db_number, start_byte, num_bytes)
set_int(r_int,0,15)
plc.db_write(db_number,start_byte,r_int)
```

Figura 2.5: Ejemplo de escritura de datos en un DB con la librería Snap7

Es importante tener en cuenta que al utilizar los comandos de lectura y escritura en el bloque de datos, se deben considerar las especificaciones del programa del PLC y asegurarse de que se está escribiendo en la posición y formato correctos, respetando la posición así como también la longitud según el tipo de dato que se busca leer. Además, es fundamental garantizar que los datos proporcionados sean coherentes con los tipos de datos esperados en el bloque de datos. Finalmente, también se debe tener en cuenta que es necesario un parámetro extra tanto en la lectura como en el "set_bool", ya que no solo se especifica el byte de inicio, sino que aparte se debe especificar el bit que representa al booleano, ya que se pueden almacenar hasta 8 booleanos por byte.

Capítulo 3

Implementación del sistema de control y comunicación inalámbrica

En este capítulo se detalla la implementación del sistema propuesto en el proyecto, comenzando con una descripción de la arquitectura basada en la comunicación Cliente/Servidor. Se presentan los componentes de cada parte y su interconexión, proporcionando una visión clara de la estructura del sistema.

Luego, se aborda la configuración del sistema de control para el motor trifásico de inducción en sus diferentes etapas. Se incluye la configuración detallada del variador de frecuencia SINAMICS G120, la configuración de la red Profinet para asegurar la comunicación entre los dispositivos, la configuración del telegrama de comunicación estándar 20 para obtener los parámetros de operación del motor, y finalmente, se muestra la implementación de un bloque de datos para facilitar el intercambio de información entre el sistema de control y el HMI.

Posteriormente, se procede con la configuración del módulo SCALANCE W774, que actuará como Punto de Acceso (AP) para posibilitar la comunicación entre la subred Profinet y la estación de monitoreo remota. Se describen en detalle los pasos y parámetros requeridos para su correcta configuración.

Con esta implementación, se logra establecer una comunicación efectiva y segura entre los distintos componentes del sistema, permitiendo un monitoreo y control óptimo del motor trifásico de inducción.

3.1. Arquitectura del sistema

Como se definió en la Sección 1.3, la comunicación cliente/servidor es un modelo de interacción en el cual un dispositivo (cliente) envía solicitudes a otro dispositivo (servidor) para obtener información, recursos o servicios.

Dentro del presente proyecto se establece una comunicación cliente/servidor entre la estación de monitoreo remota y la subred Profinet, como se visualiza en la figura 3.1. El módulo SCALANCE W774 actúa como punto de acceso (AP) que permite a los clientes establecer una conexión con los dispositivos de la subred Profinet y de esta manera poder operar y monitorear el motor trifásico de inducción.

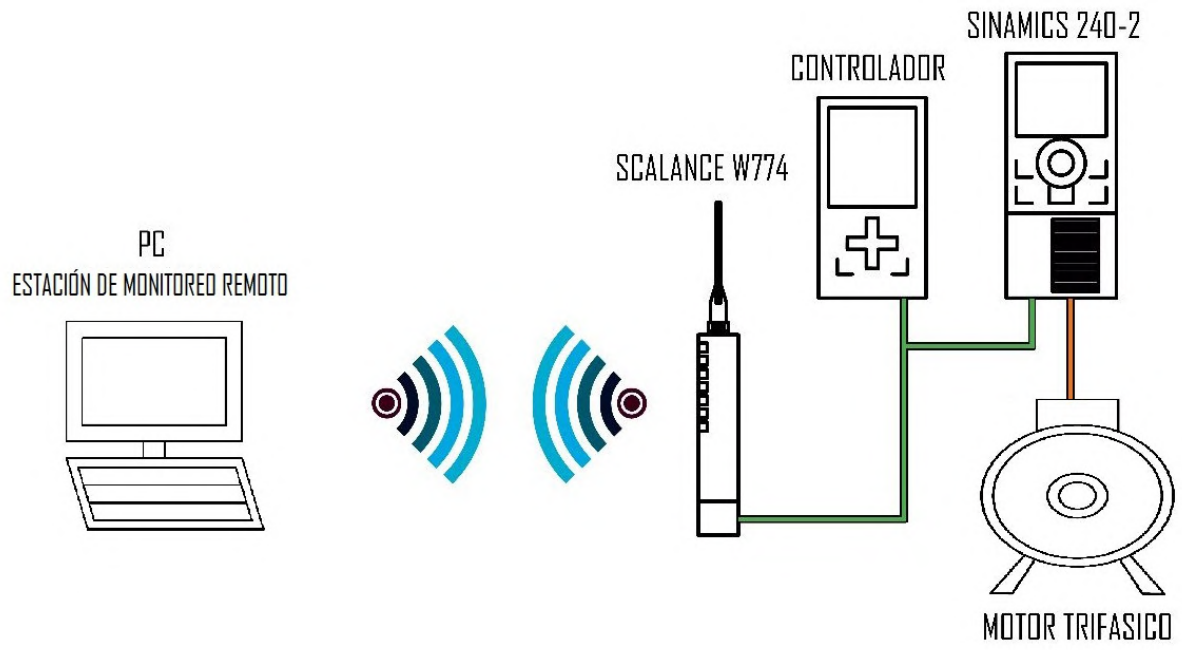


Figura 3.1: Arquitectura Cliente/Servidor del proyecto.

La estación de monitoreo remota actúa como cliente, enviando solicitudes al servidor representado por los dispositivos de la subred Profinet. Estas solicitudes pueden incluir comandos de control, como iniciar o detener el motor, o solicitar información sobre el estado del motor.

Por su parte, los dispositivos de la subred Profinet funcionan como servidores, respondiendo a las solicitudes del cliente proporcionando los datos solicitados o llevando a cabo las acciones requeridas en el motor. Esto implica el monitoreo de

variables clave, la realización de ajustes de parámetros y la generación de alarmas en caso de condiciones anormales.

A través de esta comunicación cliente/servidor, se logra un control efectivo y una operación eficiente del motor trifásico de inducción, permitiendo su supervisión en tiempo real, el ajuste de parámetros y el control de manera inalámbrica.

3.1.1. Cliente

La estación de monitoreo es un componente clave, ya que en ella se contará con el HMI desarrollado, el cual, permite supervisar y controlar el motor trifásico de inducción desde una ubicación remota. Para cumplir con sus funciones, la estación debe contar con las siguientes características:

- Interfaz de usuario amigable: La estación de monitoreo debe proporcionar una interfaz intuitiva y fácil de usar que permita a los operadores interactuar con el sistema. Esto puede incluir paneles de control, gráficos en tiempo real, pantallas de estado y botones de control para iniciar, detener o ajustar consignas del motor.
- Visualización de datos: La estación de monitoreo debe ser capaz de mostrar datos relevantes sobre el motor, como la velocidad, el torque, la corriente y la potencia. Estos datos se presentarán de manera clara y comprensible.
- Alertas y alarmas: La estación de monitoreo debe tener la capacidad de generar alertas y alarmas en caso de condiciones anormales o situaciones de riesgo, como sobrecargas o fallos en el motor. Estas notificaciones deben ser visibles y audibles para que los operadores puedan tomar medidas inmediatas.
- Registro de datos: Para un análisis posterior y un seguimiento detallado del rendimiento del motor, la estación de monitoreo debe ser capaz de registrar y almacenar datos relevantes de las variables de interés. Esto permitirá realizar análisis posteriores y tomar decisiones informadas sobre el mantenimiento y la optimización del motor.

3.1.2. Servidor

En el proyecto, el término "servidor"hace referencia al sistema de control del motor trifásico de inducción, que consta de varios componentes interconectados para su funcionamiento. Estos componentes son:

- SCALANCE W774: Es el punto de acceso inalámbrico (AP) que permite establecer la red IWLAN y conectar de forma inalámbrica el PLC y la interfaz de usuario.
- PLC SIMATIC S7-1500 1516-3 PN/DP: El PLC se encarga de recibir las señales y comandos de la interfaz de usuario, procesarlos y enviar las señales de control adecuadas hacia el variador de frecuencia.
- Variador de frecuencia SINAMICS G120: Es un equipo empleado para regular la velocidad y el funcionamiento del motor de inducción. El variador de frecuencia permite ajustar la velocidad del motor y controlar su arranque, parada y dirección de giro; así como obtener los datos de velocidad, torque, corriente y potencia del mismo.

Estos componentes se integran para habilitar el control y monitoreo del motor trifásico de inducción, estableciendo una conexión a través del estándar de comunicación Profinet, la cual se visualiza en la figura 3.2.

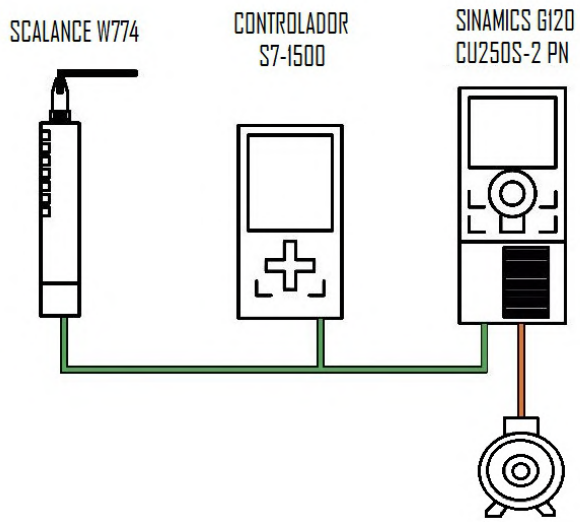


Figura 3.2: Topología de la subred PROFINET.

3.1.3. SINAMICS G120

Los convertidores de frecuencia SINAMICS G120 son dispositivos altamente eficientes que se utilizan para controlar con precisión el par y la velocidad de los motores trifásicos. Estos convertidores son especialmente adecuados para aplicaciones industriales y ofrecen una amplia gama de funciones y características avanzadas [31].

Gracias a su tecnología avanzada, los convertidores SINAMICS G120 permiten un control preciso y suave de los motores, lo que se traduce en un funcionamiento más eficiente y una mayor vida útil de los equipos. Estos convertidores son altamente flexibles y se pueden adaptar a una amplia variedad de aplicaciones, desde sistemas de bombeo y ventilación hasta maquinaria de procesamiento y producción.

Según [31], consta principalmente de dos unidades funcionales esenciales, las cuales se pueden ver en la figura 3.3:

- **Power Module (PM):** es responsable de la conversión de energía eléctrica para alimentar el motor. Este módulo utiliza una tecnología de estado sólido para convertir la corriente de entrada en una forma adecuada para el motor, permitiendo un control preciso de la velocidad y el par.
- **Control Unit (CU):** es responsable de controlar y supervisar el funcionamiento del Power Module y el motor. Esta unidad ofrece una variedad de modos de regulación que se pueden seleccionar, lo que permite ajustar el control del motor según las necesidades específicas de la aplicación.

(a) CU250S-2 (b) PM240-2

Figura 3.3: Módulos CU y PM del accionamiento SINAMICS G120 [32].

Además de los módulos antes mencionados, se pueden implementar otros módulos que permiten mejorar las prestaciones y la interacción con el driver. Uno de ellos es el IOP-2 (Intelligent Operator Panel 2) (figura 3.4), el cual, proporciona una interfaz intuitiva y fácil de usar que permite llevar a cabo varias funciones relacionadas con la puesta en marcha, el diagnóstico de fallos y la configuración de los parámetros de funcionamiento del dispositivo [33].

Con el IOP-2, se puede realizar una puesta en marcha rápida del variador, lo que agiliza el proceso de configuración inicial, sin necesidad de utilizar ordenadores o software adicional.

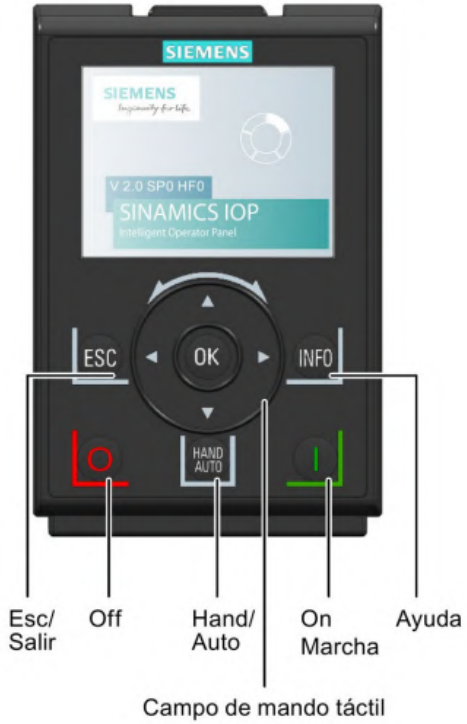


Figura 3.4: Vista frontal del IOP-2 [33].

3.1.4. Telegrama de comunicación 20 PZD 2/6

Los telegramas de comunicación son esenciales en sistemas de automatización industrial, ya que permiten el intercambio de datos entre dispositivos, facilitando la integración y la interoperabilidad. Esto garantiza una comunicación efectiva y confiable entre los componentes del sistema.

Los telegramas se encuentran estructurados en dos partes principales: las palabras de recepción y las palabras de transmisión. Las palabras de recepción son utilizadas por el PLC para recibir datos desde el variador de frecuencia, mientras que las palabras de transmisión son utilizadas para enviar datos desde el PLC al variador. En el caso del telegrama 20 PZD 2/6 empleado en el proyecto, se utilizan dos palabras para recepción de datos y seis palabras para transmisión de datos [34]. La estructura del telegrama de comunicación 20 PZD 2/6 se presenta en la Tabla 3.1.

Tabla 3.1: Estructura Telegrama de comunicación 20 PZD 2/6 [34].

	PZD1	**PZD2**	**PZD3**	**PZD4**	**PZD5**	**PZD6**
Palabra de recepción	STW1	NSOLL_A				
Palabra de transmisión	ZSW1	NIST_A_ GLATT	IAIST_ GLATT	MIST_ GLATT	P_IST_ GLATT	MELD_ NAMUR

Cada palabra de datos en el telegrama contiene información relevante para la operación y monitoreo del variador de frecuencia, como la velocidad del motor, la dirección de giro, la corriente, la potencia, entre otros parámetros. Además, permite la detección de errores dentro de la red según la definición de VIK_NAMUR.

Bloque SINA_SPEED_TLG20

La libreria "Library SINAMICS Extended"(LSINAExt) contiene una serie de bloques que permiten el control sencillo y cíclico de actuadores SINAMICS. Estos bloques de función están diseñados para facilitar el control de los actuadores, simplificando los parámetros de entrada al mínimo necesario. Esto permite un control fácil y directo sin la necesidad de conocer la compleja estructura del telegrama de comunicación [35].

El bloque SINA_SPEED_TLG20 facilita una comunicación efectiva entre dispositivos mediante el uso del telegrama estándar 20. Su principal función es controlar la operación de un motor trifásico de inducción, ajustando su velocidad según una consigna establecida.

Este bloque cuenta con parámetros de entrada que se utilizan para configurar y controlar el variador. Estos parámetros son transmitidos al equipo, permitiendo ajustar su funcionamiento de acuerdo a las necesidades específicas del sistema.

Por otro lado, los parámetros de salida proporcionan información del estado del variador y los datos en tiempo real de velocidad, corriente, potencia y torque. Además, el bloque SINA_SPEED_TLG20 tiene la capacidad de detectar y emitir información sobre estados de error y advertencias de fallo en caso de que ocurran, lo que contribuye a mantener un funcionamiento seguro y confiable del sistema en todo momento. En la Figura 3.5 se visualiza la estructura del bloque SINA_SPEED_TLG20.

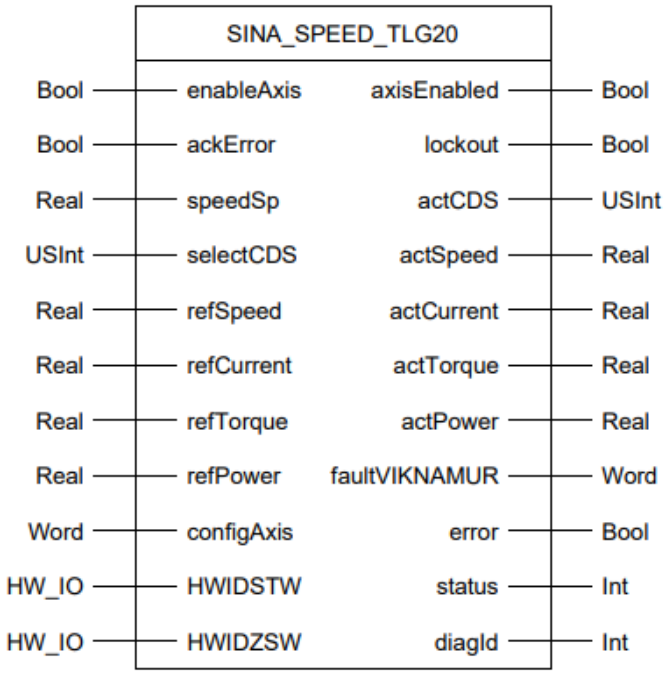


Figura 3.5: Bloque SINA_SPEED_TLG20 [35].

En las Tablas 3.2 y 3.3 se visualiza una descripción general de los parámetros de entrada y salida que constituyen el bloque SINA_SPEED_TLG20. Mientras que en la tabla 3.4 se presentan todos los avisos de fallo según la definición VIK-NAMUR que se pueden detectar gracias a la implementación del telegrama estándar 20.

Tabla 3.2: Parámetros de entrada SINA_SPEED_TLG20 [34].

Nombre	Tipo de Dato	Representación
enableAxis	Bool	Encendido de la unidad
ackError	Bool	Reconocimiento de fallos
speedSp	Real	Consigna de velocidad [1/min]
selectCDS	USInt	Selección del conjunto de datos de comando
refSpeed	Real	Velocidad de referencia [1/min]
refCurrent	Real	Corriente de referencia [Aeff]
refTorque	Real	Par de referencia [Nm]
refPower	Real	Potencia de referencia [kW]
configAxis	Word	Parámetro de entrada con código binario para controlar todos los bits de la palabra de control que no están disponibles como parámetros de entrada
HWIDSTW	HW_IO	ID de hardware de la ranura de consigna
HWIDZSW	HW_IO	ID de hardware de la ranura de valor real

Tabla 3.3: Parámetros de salida SINA_SPEED_TLG20 [34].

Nombre	Tipo de Dato	Representacion
axisEnabled	Bool	1 = Accionamiento en funcionamiento
lockout	Bool	1 = Conmutación inhibida activa
actCDS	Real	Conjunto de datos de comando activo
actSpeed	USInt	Velocidad actual [1/min]
actCurrent	Real	Valor actual de la corriente [Aeff]
actTorque	Real	Valor real del par [Nm]
actPower	Real	Valor real de potencia activa [kW]
faultVIKNAMUR	Real	Palabra de fallo según definición VIK-NAMUR
error	Word	1 = Error en el bloque de función / accionamiento
status	HW_IO	Salida de estado
diagId	HW_IO	Código de error de las funciones del sistema DPWR_DAT/DPRD_DAT

Tabla 3.4: Palabra de fallo según la definición VIK-NAMUR [34].

Bit	Significado
0	1 = La unidad de control informa de un fallo
1	1 = Error de red: Fallo de fase o tensión inadmisible
2	1 = Sobretensión del enlace CC
3	1 = Avería del módulo de potencia, por ejemplo, sobrecorriente o sobretemperatura
4	1 = Sobretemperatura del inversor
5	1 = Fallo a tierra / fallo de fase en el cable del motor o en el motor
6	1 = Sobrecarga del motor
7	1 = Comunicación al sistema de control superpuesto perturbada
8	1 = Error en un canal de monitorización seguro
10	1 = Fallo en la comunicación interna del inversor
11	1 = Fallo en la red
15	1 = Otros fallos

3.1.5. Diagrama de conexión

El proyecto se realiza en los bancos de trabajo las instalaciones del laboratorio de ”Redes Industriales” de la Universidad Politécnica Salesiana, Sede Cuenca.

En la Figura 3.6 se observa el banco de trabajo empleado. Por otra parte, en la Figura 3.7 se muestra el diagrama de conexiones realizadas para poner en marcha los equipos.

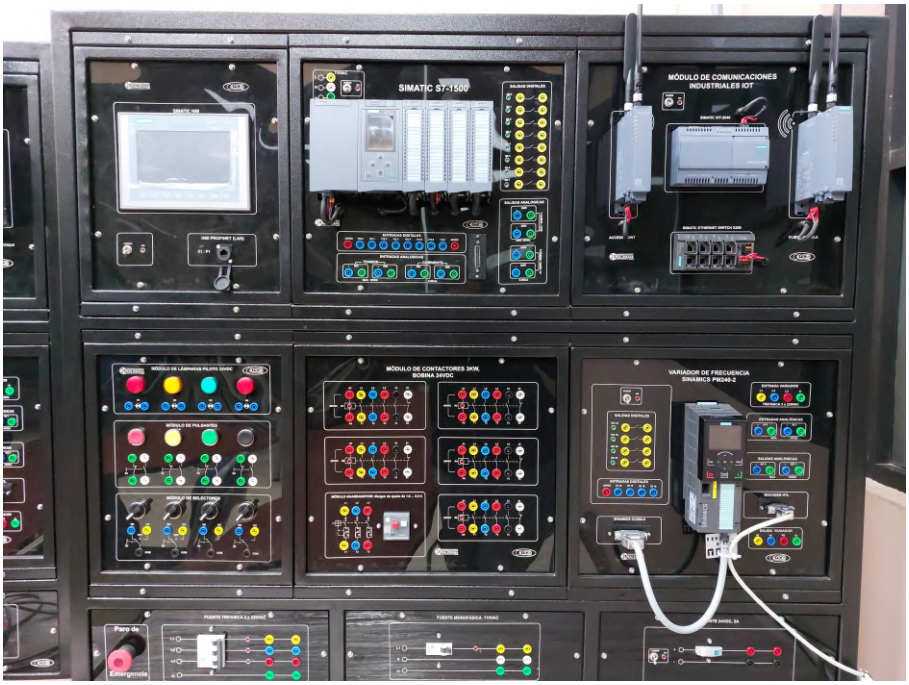


Figura 3.6: Banco de trabajo, Laboratorio ”Redes Industriales”, UPS Sede Cuenca.

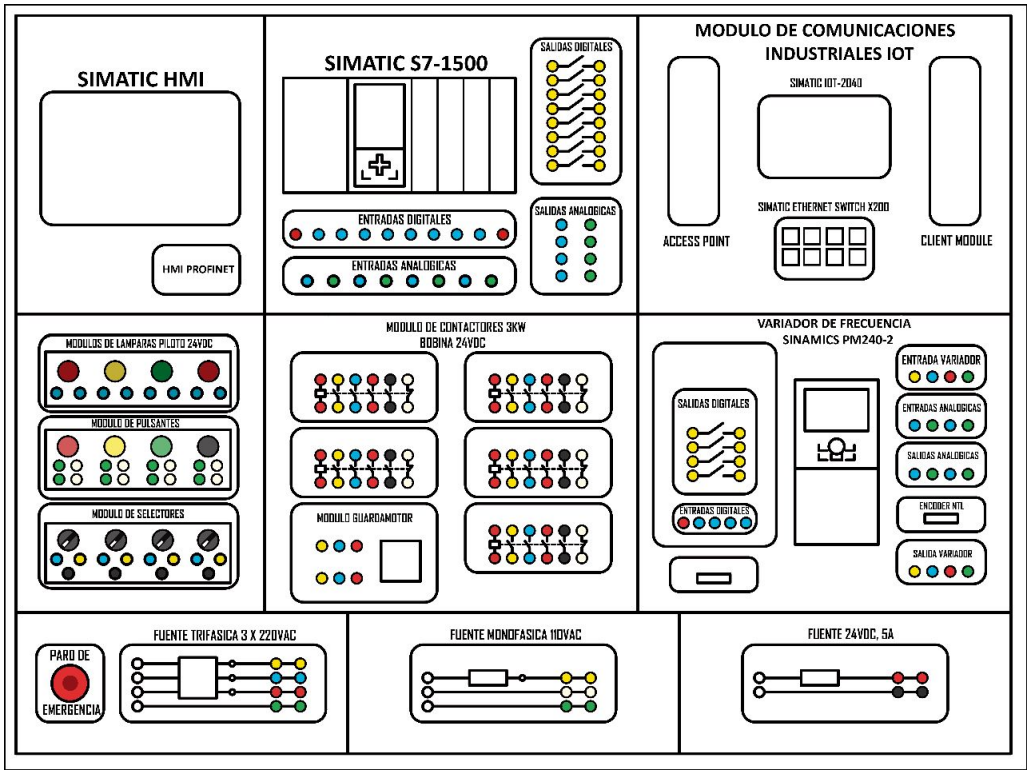


Figura 3.7: Diagrama de conexiones.

El actuador final se trata de un motor trifásico de inducción, el cual trabaja en conexión triángulo, tal como se muestra en la Figura 3.8.

Figura 3.8: Motor trifásico de inducción.

Finalmente en las Figuras 3.9 y 3.10 se presentan los diagramas de conexión de la etapa de potencia y de la subred Profinet, respectivamente.

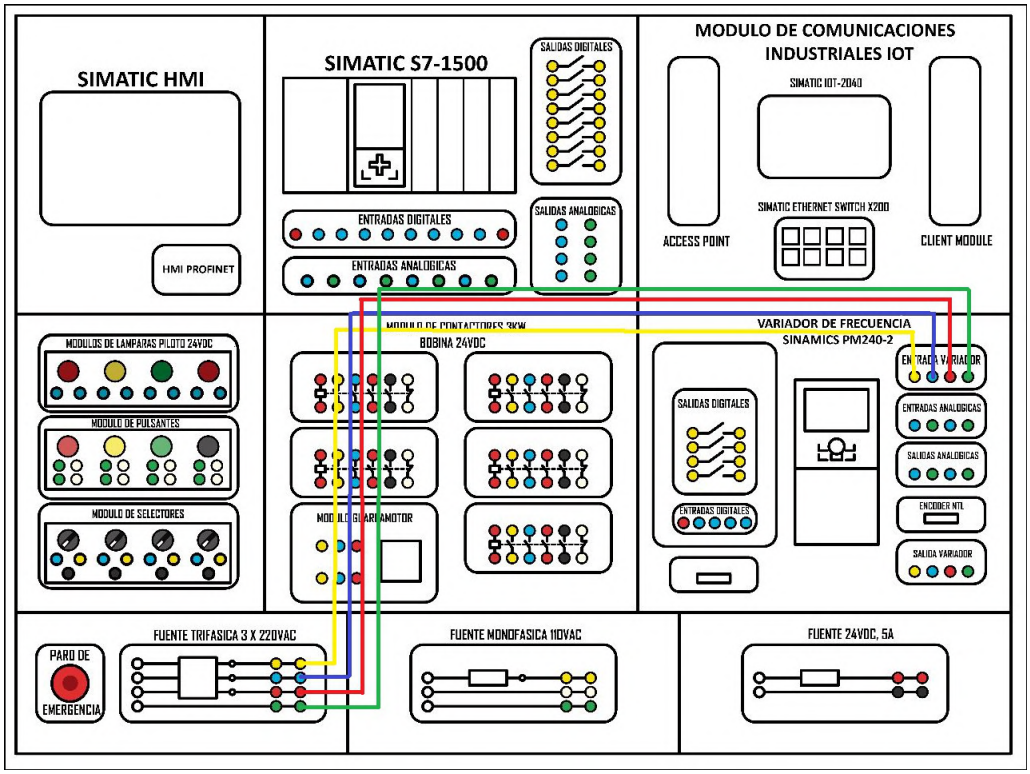


Figura 3.9: Diagrama de conexión de potencia.

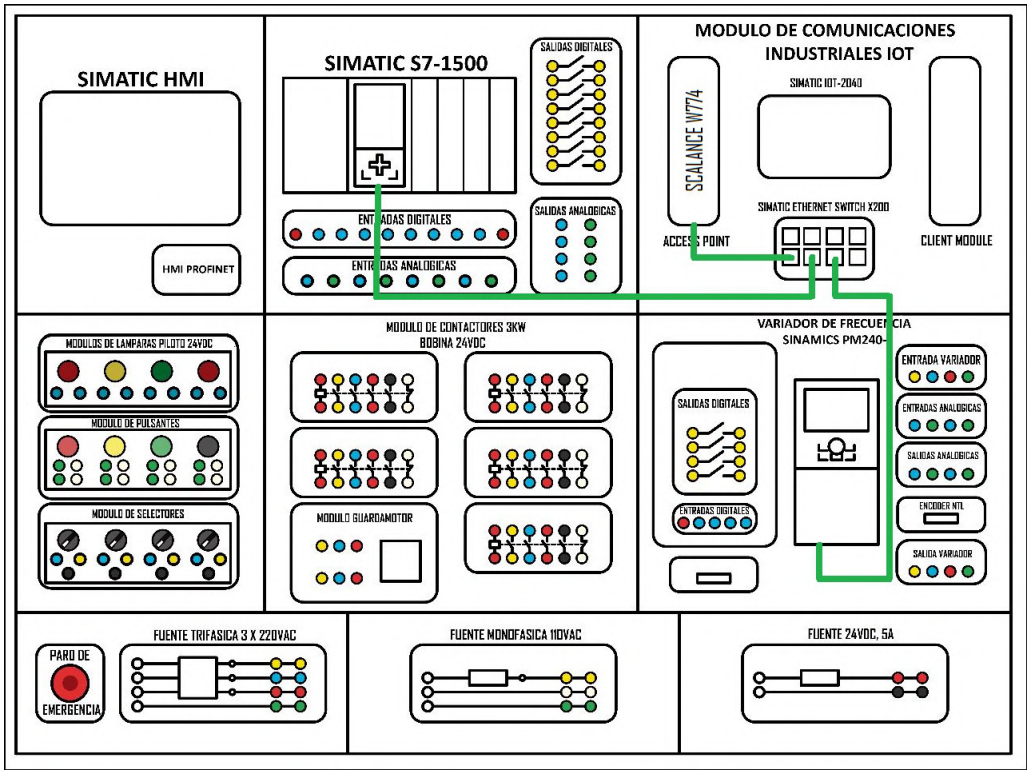


Figura 3.10: Diagrama de conexión Profinet.

3.2. Configuración del sistema de control

La configuración del sistema para el actuador se lleva a cabo en varias etapas. Primero, se realiza la configuración del SINAMICS G120, seguido de la configuración de la red Profinet. Finalmente, se establece el telegrama de comunicación apropiado para los requerimientos del sistema.

3.2.1. Configuración del accionamiento SINAMICS G120, CU250S-2 PN

La configuración de los ajustes del accionamiento SINAMICS G120 se puede realizar mediante la interfaz que ofrece el IOP-2, directamente desde el variador.

Configuración de parámetros del motor

En la pantalla principal del IOP-2, se debe elegir el menú de **"Instalación"** (véase la figura 3.11).

Figura 3.11: Pantalla principal del IOP-2.

Posteriormente, en la nueva pantalla se selecciona la opción **"Puesta en marcha rápida"** (véase la figura 3.12).

Figura 3.12: Pantalla de puesta en marcha.

En la ventana que se despliega se opta por la opción **"Si, restablecer ajustes de fábrica"** (véase la figura 3.13).

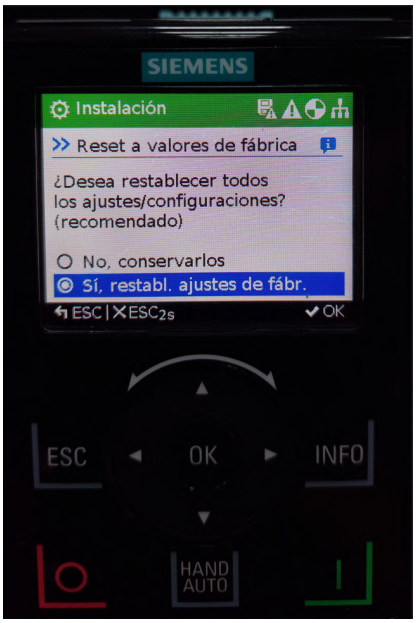


Figura 3.13: Pantalla de restablecimiento a valores predeterminados de fábrica.

Una vez restablecidos los ajustes de fábrica, se procede a llenar los parámetros correspondientes en la siguiente ventana (véase la figura 3.14), tomando en cuenta las características del motor empleado. Los parámetros empleados se visualizan en la tabla 3.5.

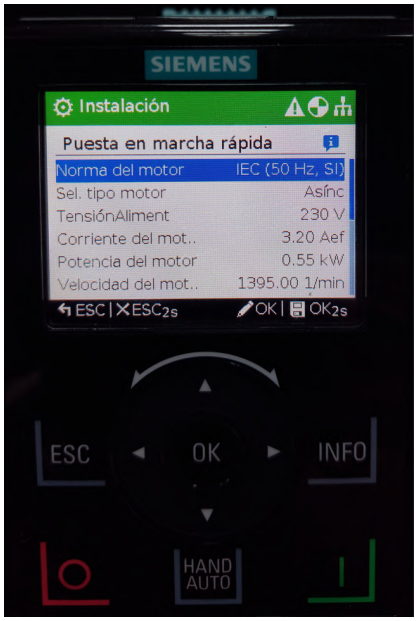


Figura 3.14: Pantalla de configuración de parámetros del motor.

Tabla 3.5: Parámetros del motor trifásico de inducción.

Parámetro	**Configuración**
Norma del Motor	Nema(60 Hz, SI)
Sel. tipo de motor	Asíncrono
Tensión de alimentación	220 V
Corriente del motor	2.20 Aef
Potencia del motor	0.55 kW
Velocidad del motor	1440.0 rpm
Tensión del motor	220 V
Frecuencia del motor	60 Hz
Velocidad mínima	0.0 rpm
Velocidad máxima	1440.0 rpm
Tiempo de aceleración	5 s
Tiempo de deceleración	5 s
Configuración de E/S	(7) Regulación PN/PD

Configuración del telegrama de comunicación

Para configurar el telegrama de comunicación, se lleva a cabo el proceso que se detalla a continuación.

En la pantalla principal, se elige el icono de **"Parámetros"**, de acuerdo a lo ilustrado en la figura 3.15.

Figura 3.15: Pantalla principal del IOP-2 - selección de parámetros.

A continuación, se selecciona la opción **"Búsqueda por número"**, de acuerdo con lo expuesto en la figura 3.16.

Figura 3.16: Pantalla de selección para búsqueda de parámetro por número.

Posteriormente se despliega una nueva ventana en donde se puede seleccionar el número de parámetro a buscar. Para los fines del proyecto, se ingresa el número **922** (véase la figura 3.17).

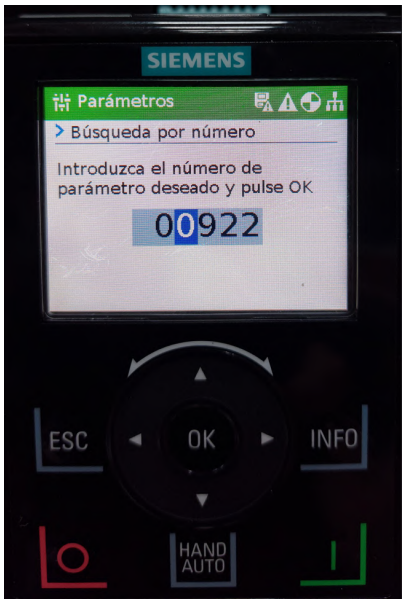


Figura 3.17: Pantalla de selección por número de parámetro.

En la ventana desplegada, se elige la opción de parámetro **"p922 PZD Selec_telegr"** para poder definir el telegrama de comunicación que se empleará entre el variador y el controlador (véase la figura 3.18).

Figura 3.18: Pantalla de selección de parámetro.

Dentro de la ventana seleccionada, se debe elegir la opción **"Telegrama estándar 20"** (véase la figura 3.19). Posteriormente, se debe regresar a la ventana principal presionando la tecla **"ESC"**.

Figura 3.19: Pantalla para selección de Telegrama.

Identificación del motor

Para finalizar la configuración del variador es necesario realizar el reconocimiento del motor. Para ello, se presiona la tecla **"HAND/AUTO"** para habilitar el modo de operación manual del dispositivo (véase la figura 3.19).

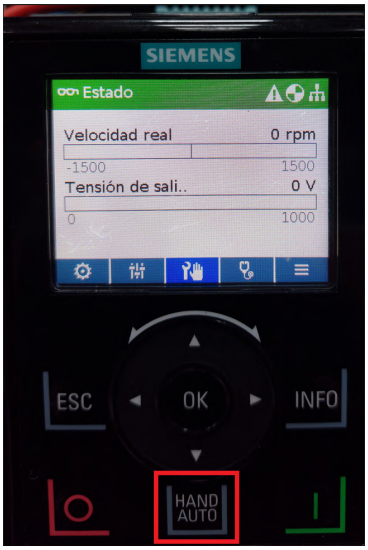


Figura 3.20: Pantalla principal del IOP-2 - selección: operación manual.

Dentro de la ventana de operación manual, el inicio del reconocimiento del motor se realiza presionando la tecla **"ON"** (véase la figura 3.21).

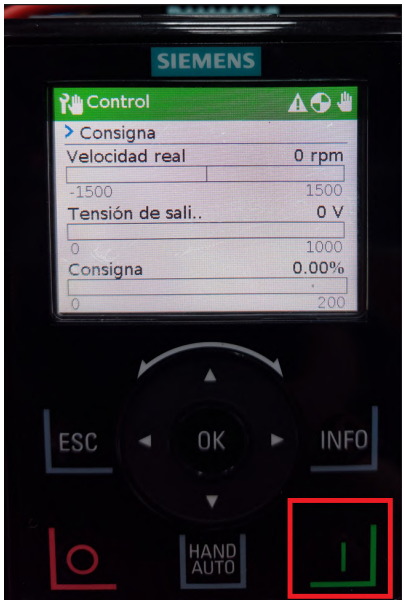


Figura 3.21: Pantalla de identificación del motor.

En el IOP-2 se podrá visualizar el porcentaje del proceso, una vez que el reconocimiento se complete se debe presionar la tecla **"OFF"** para regresar a la ventana de operación manual (véase la figura 3.22).

Figura 3.22: Identificación del motor.

Una vez reconocido el motor, se debe salir del modo manual para que el

variador pueda ser operado de forma automática a través de la subred Profinet. Para ello, se debe presionar nuevamente la tecla **"HAND/AUTO"** (véase la figura 3.23).

Figura 3.23: Pantalla: operación manual.

Finalmente, una vez ubicados en la pantalla principal del variador, se comprueba que se encuentre en modo automático verificando el icono ubicado en la equina superior derecha del IOP-2 (véase la figura 3.24).

Figura 3.24: Pantalla principal - modo automático.

3.2.2. Configuración de la comunicación Profinet

La puesta a punto de la comunicación Profinet y de los equipos que la conforman se lleva a cabo empleando el software TIA PORTAL V15.1, en un ordenador que debe estar integrado a la red Profinet mientras se realiza el proceso. Para ello, se debe crear un nuevo proyecto en el software y agregar el controlador y los módulos con los que se trabajará.

Se agrega el controlador CPU 1516-3 PN/DP versión 2.6, perteneciente a la familia **SIMATIC S7-1500**.

Mientras que los módulos con lo que se cuenta dentro del banco de trabajo son los siguientes:

- PM 190W 120/230VAC
- AI 8xU/I/RTD/TC ST
- AQ 4xU/I ST
- DI 32x24VDC HF
- DQ 32x24VDC/0.5A HF

En la Figura 3.25 se visualiza el controlador y los módulos agregados dentro del software.

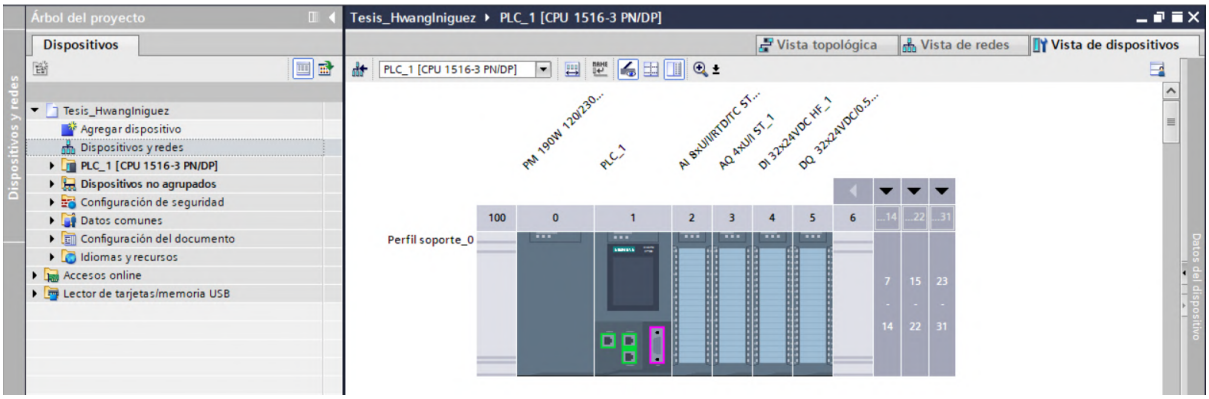


Figura 3.25: Controlador y módulos en TIA Portal.

Asignación de la dirección IP física para el accionamiento

Para lograr la comunicación entre el controlador y el accionamiento mediante Profinet, se requiere asignar una dirección IP a cada dispositivo en una red común.

Para llevar a cabo la configuración de la dirección IP del variador, es necesario acceder al **"Árbol del proyecto"** y hacer clic en **"Accesos online"**. De las opciones disponibles, se debe seleccionar la tarjeta de red con la que se trabaja. Para el presente proyecto la tarjeta utilizada será: **"Realtek USB FE Family Controller"**. A continuación se debe hacer clic en **"Actualizar dispositivos accesibles"** (véase la figura 3.26).

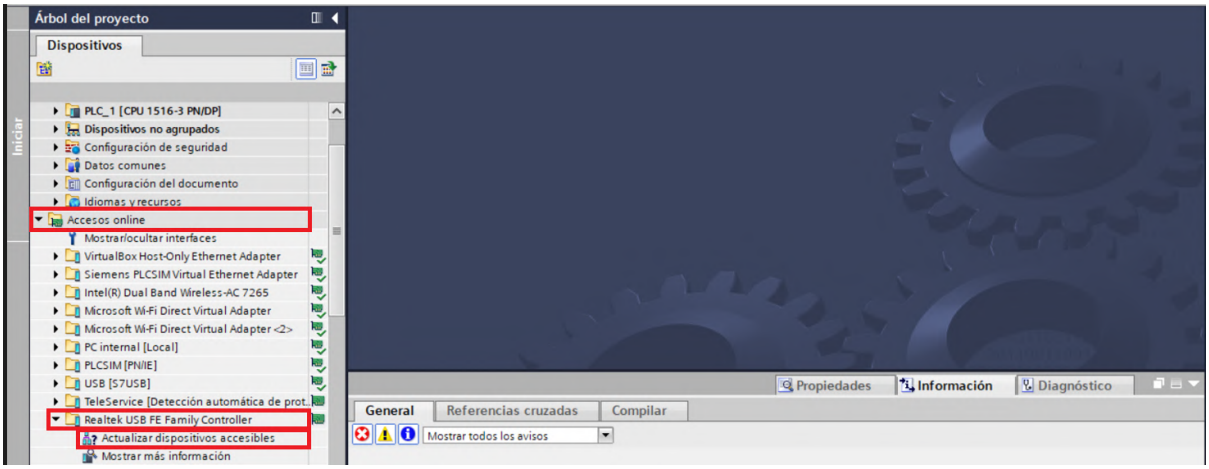


Figura 3.26: Accesos online.

Posteriormente, se debe seleccionar el dispositivo **"sinamics-g120sv-pn"** y se debe dar clic en **"Online y diagnóstico"** (véase la figura 3.27).

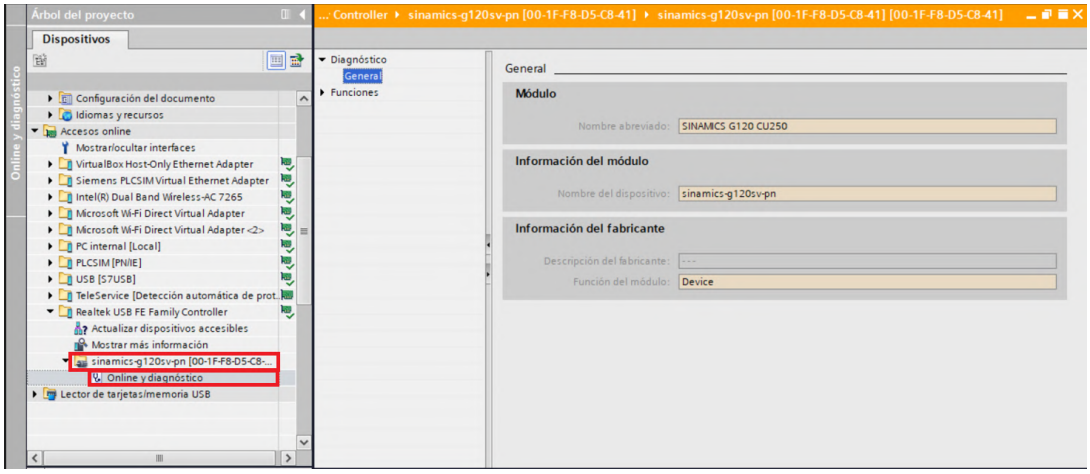


Figura 3.27: Ventana online y diagnóstico

A continuación, se selecciona la opción **"Funciones"** y **"Asignar dirección IP"**. En la ventana que se despliega, se debe escribir la Dirección IP y la Máscara de subred con la que se va a trabajar. Finalmente se debe dar clic en el botón **"Asignar dirección IP"** (véase la figura 3.28).

Para el presente proyecto se trabaja con las siguientes especificaciones:

- Dirección IP: 192.168.1.3
- Mascara de subred: 255.255.255.0

Figura 3.28: Asignación de la dirección IP para el accionamiento

Incorporación de dispositivos en la subred Profinet

En la sección **"Dispositivos y redes"** del software, se procede a añadir los equipos que conforman la subred Profinet para su configuración.

En la pestaña **"Vista de redes"**, se selecciona el menú **"Catálogo de hardware"**, se ubica el variador de frecuencia **"SINAMICS G120 CU250S-2 PN VECTOR Vector 4.7"** y se agrega al proyecto (véase la figura 3.29).

Se repite el proceso, pero ahora se agrega el dispositivo SCALANCE W774, para ello se ubica en el catálogo el equipo **"SCALANCE W774-1 6GK5 774-1FX00-0AA0"** y se agrega al proyecto (véase la figura 3.30).

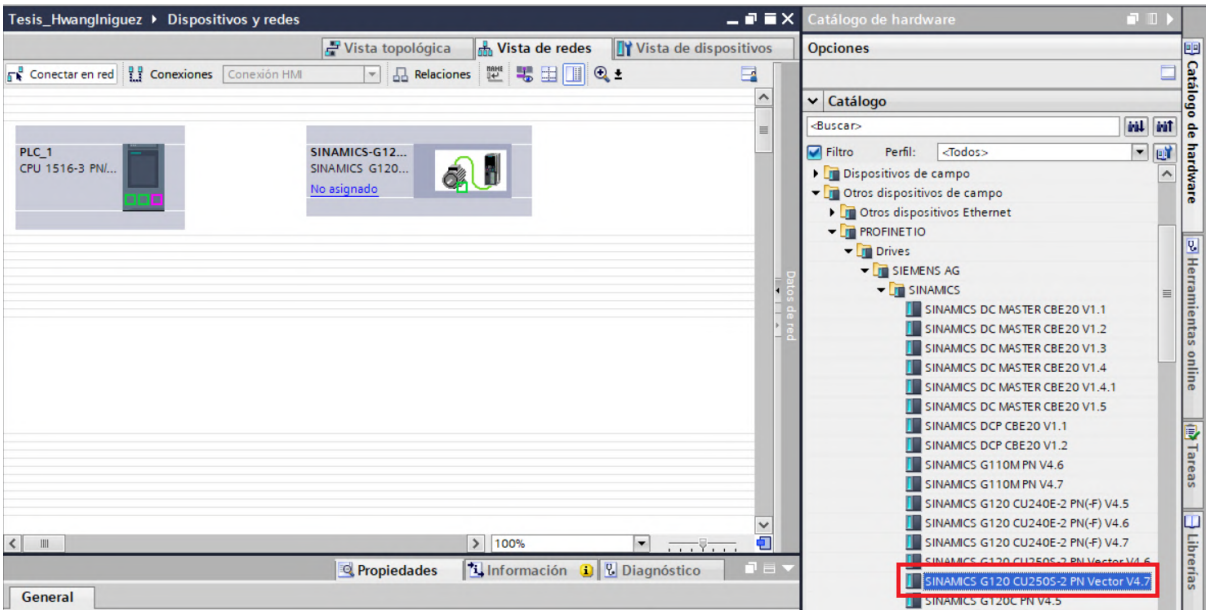


Figura 3.29: Incorporación del accionamiento SINAMICS G120 - CU250S-2 PN

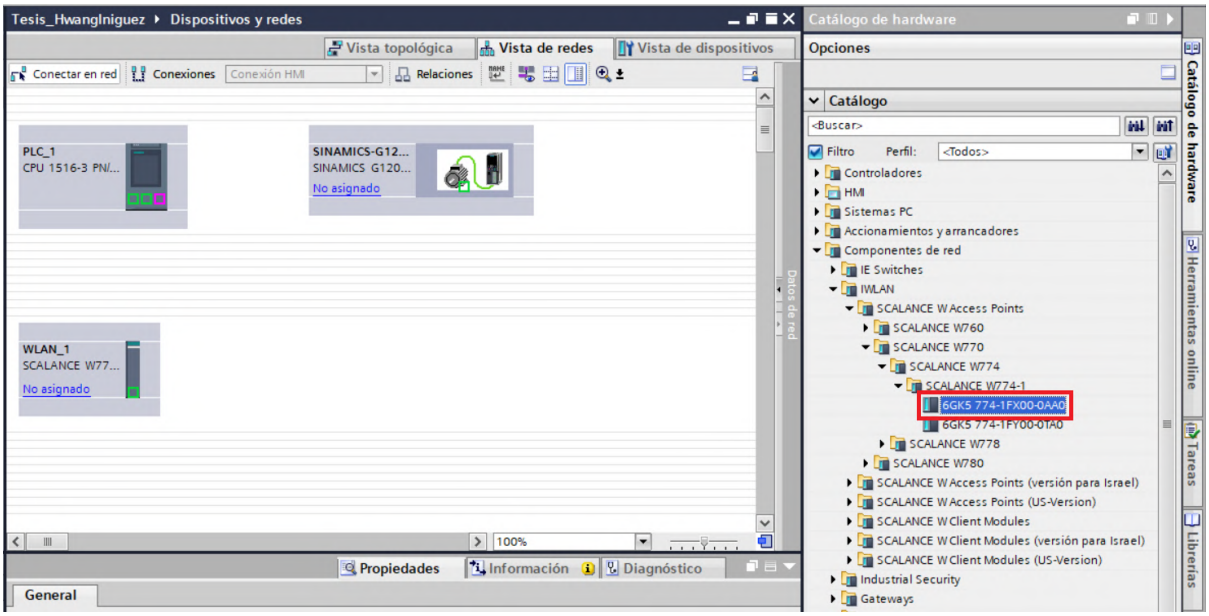


Figura 3.30: Incorporación del equipo SCALANCE W774

Asignación del telegrama de comunicación 20 PZD 2/6

Una vez agregados los dispositivos que conforman la subred, se procede a asignar el telegrama de comunicación que empleará el variador de frecuencia.

Para ello, se ingresa a la configuración del equipo y en la sección **"Submódulos"** del **"Catálogo de hardware"** se ubica la opción **"Telegrama estádar 20, PZD-2/6"** y se agrega al dispositivo (véase la figura 3.31).

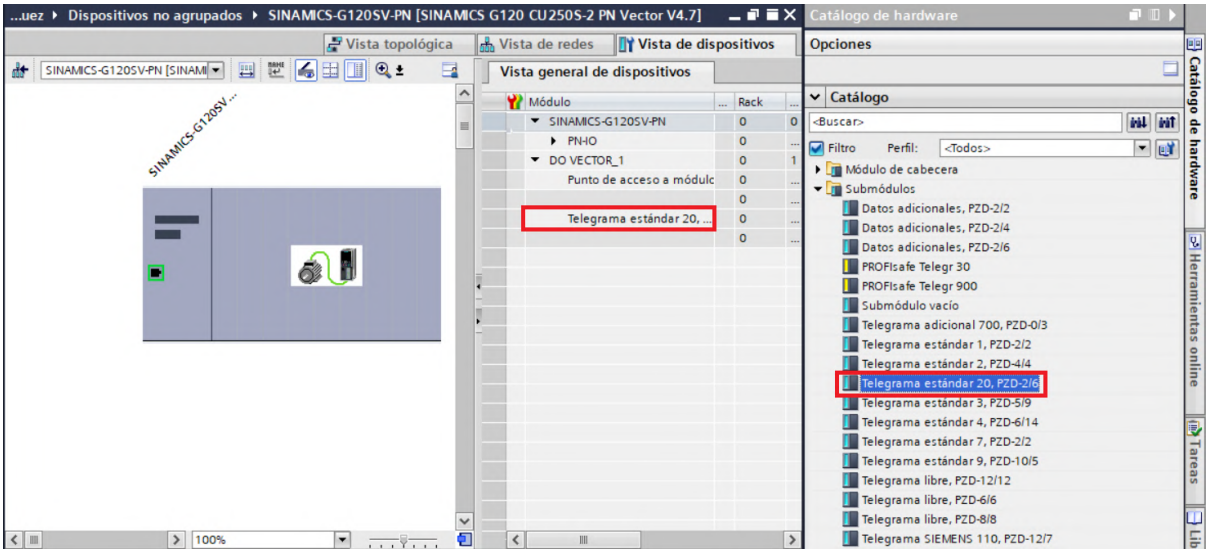


Figura 3.31: Asignación del telegrama de comunicación.

Asignación de direcciones IP para la subred Profinet

Para que la comunicación mediante la subred Profinet pueda darse, es esencial asignar una dirección IP única a cada dispositivo, coincidiendo con las configuraciones físicas de los mismos. La asignación de la dirección IP se logra haciendo clic en el puerto Profinet correspondiente de cada dispositivo y en el menú de **"Propiedades"** se selecciona la opción **"Direcciones Ethernet"** en donde es posible definir la dirección IP de los dispositivos (véase la figura 3.32).

En la Tabla 3.6 se puede visualizar las direcciones IP utilizadas en este proyecto.

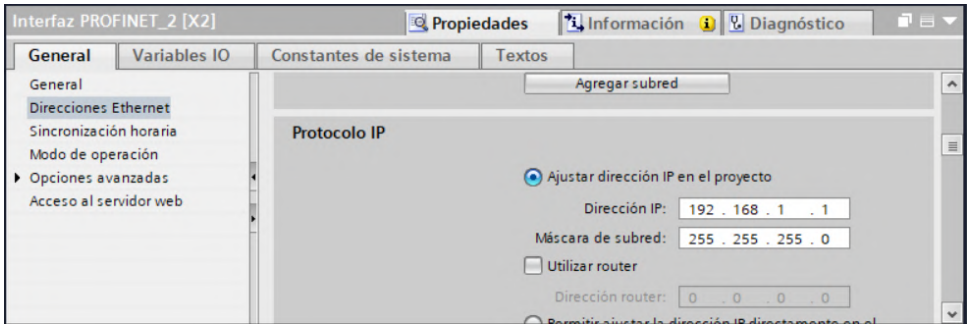


Figura 3.32: Ventana para la asignación de dirección IP para los dispositivos Profinet.

Tabla 3.6: Direcciones IP Dispositivos Profinet.

Dispositivo	**Dirección IP**
Controlador	192.168.1.1
SINAMICS G120	192.168.1.3
SCALANCE W774	192.168.1.75

Una vez configuradas las direcciones IP, se puede visualizar en **"Vista de Redes"** y comprobar la correcta asignación de las mismas, como se puede ver en la Figura 3.33.

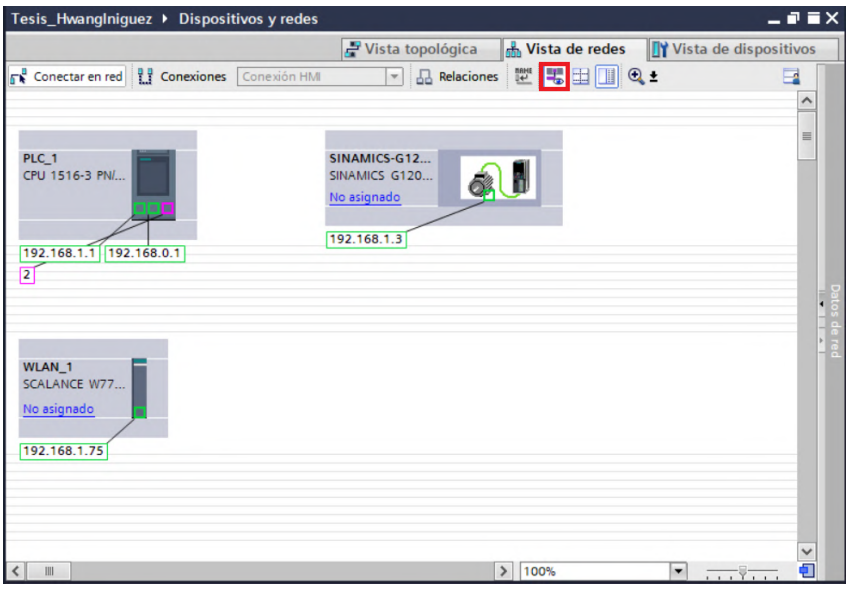


Figura 3.33: Direcciones IP asignadas.

Establecimiento de la subred Profinet

Para establecer la subred Profinet, simplemente se debe seleccionar el puerto Profinet del controlador y arrastrarlo hacia los puertos Profinet de los otros dispositivos, tal como se muestra en la Figura 3.34.

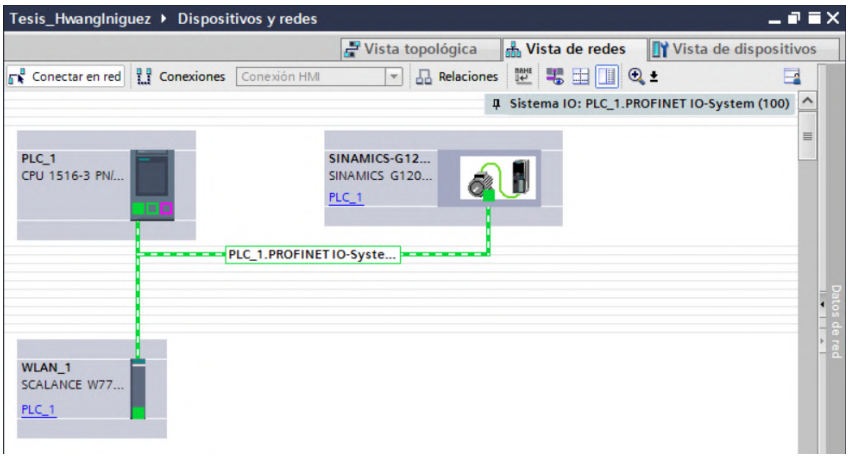


Figura 3.34: Establecimiento subred Profinet.

Identificación del accionamiento SINAMICS G120

Finalmente, para culminar con la configuración de la subred Profinet, se debe realizar la identificación del variador para poder asignar un nombre al dispositivo. Para ello, se debe dar clic derecho en la conexión establecida y seleccionar la opción **"Asignar nombre de dispositivo"**, de acuerdo a lo ilustrado en la Figura 3.35.

En la ventana que se despliega se deben elegir los parámetros correctos para la identificación del equipo.

En **"Nombre del dispositivo PROFINET:"** seleccionamos la opción **"sinamics-g120-sv-pn"**. En **"Filtros de Dispositivos"** se marca la opción **"Mostrar solo dispositivos del mismo tipo"**. Una vez establecidos los parámetros, se procede a **"Actualizar lista"** como se ve en la Figura 3.36.

Una vez finalizada la búsqueda de los dispositivos, se selecciona el equipo reconocido y se presiona el botón **"Asignar nombre"** como muestra la Figura 3.37.

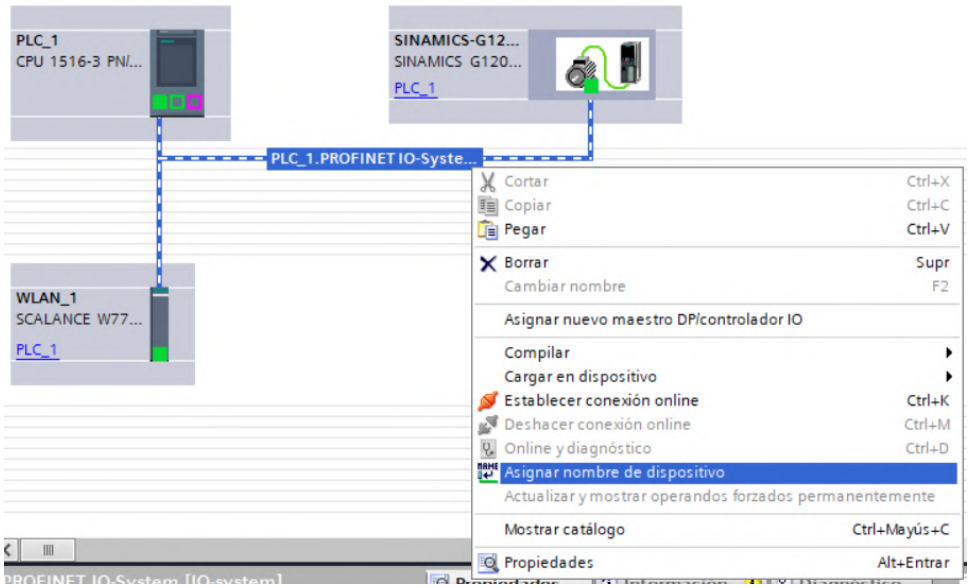


Figura 3.35: Identificación del dispositivo Profinet.

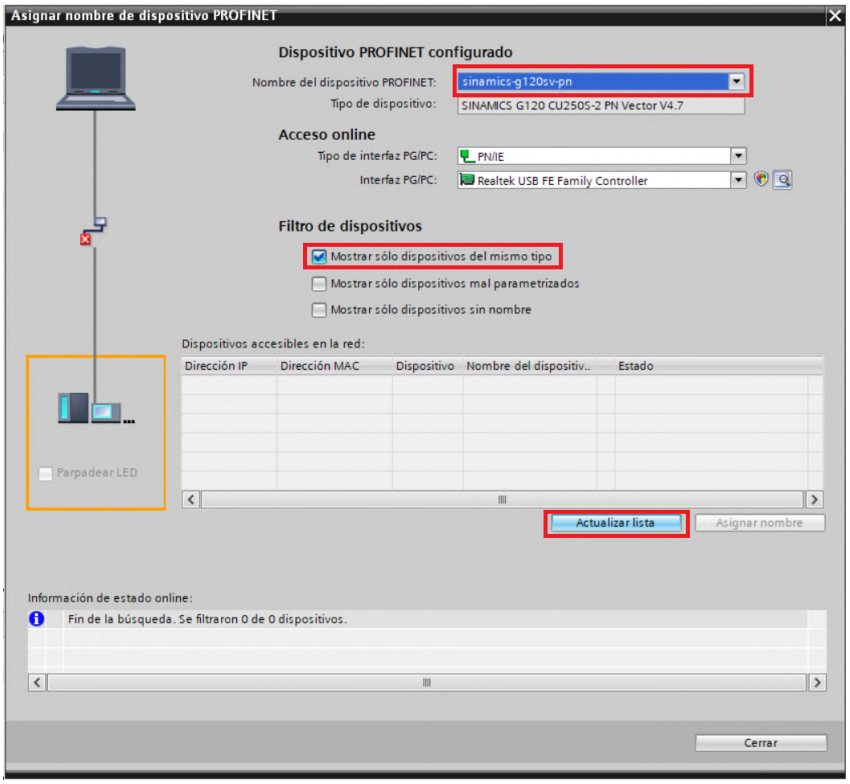


Figura 3.36: Asignación de nombre para el dispositivo PROFINET (1).

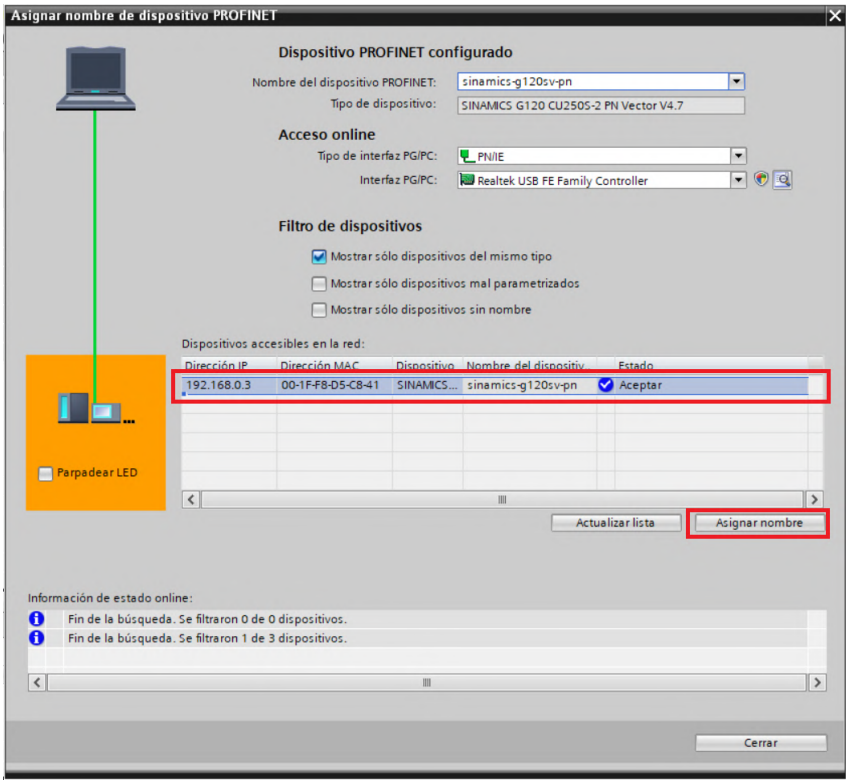


Figura 3.37: Asignación de nombre para el dispositivo PROFINET (2).

Al terminar con este proceso la subred Profinet se encuentra completamente configurada y lista para la comunicación entre los dispositivos. La subred se puede visualizar en la Figura 3.38.

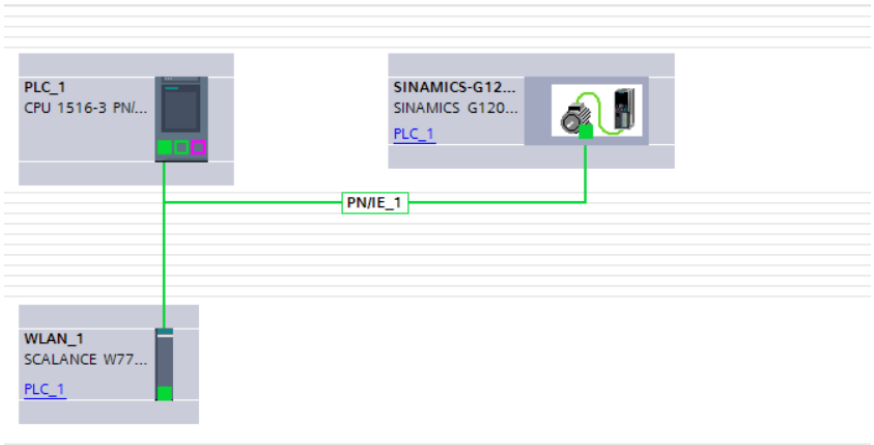


Figura 3.38: Subred PROFINET.

3.2.3. Configuración del Telegrama estándar 20 PZD 2/6

La configuración de los parámetros del telegrama estándar 20 se lleva a cabo empleando la librería LSINAext. Esta librería, mediante el bloque SINA_SPEED_TLG20, simplifica la comunicación entre el PLC y el SINAMICS G120. De esta manera, se logra establecer una conexión eficiente y efectiva entre ambos dispositivos para el intercambio de datos y el control del motor trifásico de inducción.

Instalación de la librería LSINAExt_v15.1

En primera instancia, para poder trabajar con el bloque SINA_SPEED_TLG20 es necesaria la instalación de la librería LSINAExt_v15.1, la cual se encuentra disponible en la página oficial de Siemens y que se debe tener previamente descargada en el ordenador.

Para ello, dentro del proyecto se selecciona la pestaña de **"Librerias"**. Dentro de la sección de **"Librerias locales"** se presiona la opción **"Abrir librería global"**, como se indica en la Figura 3.39.

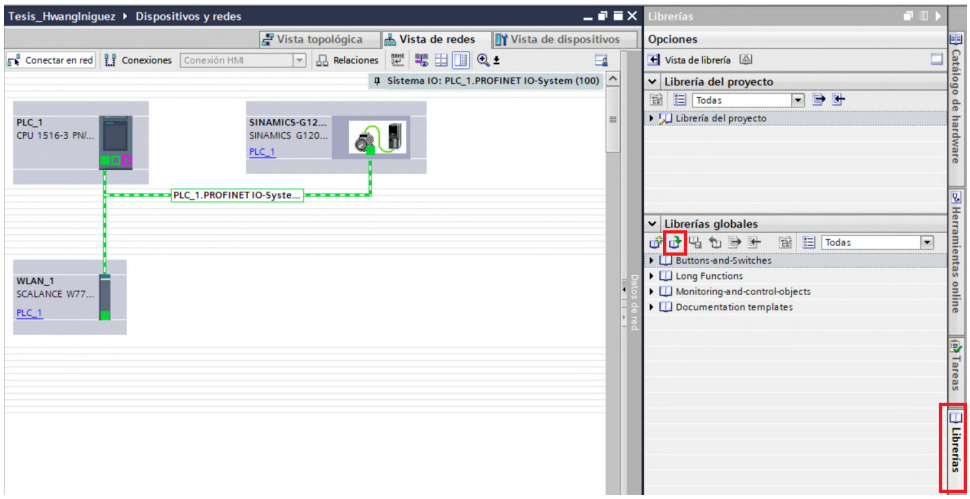


Figura 3.39: Librería global.

En la ventana que se despliega se debe ubicar la librería que se desea instalar y se debe presionar **"Abrir"**, como se muestra en la Figura 3.40.

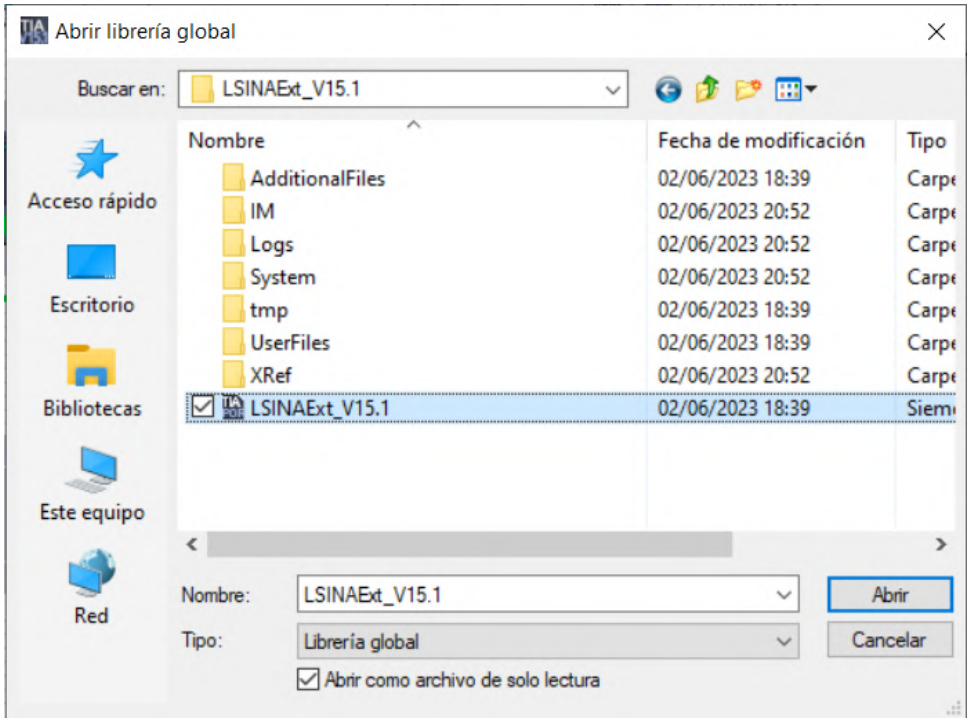


Figura 3.40: Abrir librería global.

Configuración del Bloque SINA_SPEED_TLG20

Una vez añadida la librería, se procede con la configuración del bloque SINA_SPEED_TLG20.

Para agregar el bloque se sitúa nuevamente sobre **"Librerías globales"**, en donde ya se visualiza la librería **"LSINAExt_v15.1"**. En **"Plantillas Maestras"** se selecciona el bloque **"SINA_SPEED_TLG20"**, como se visualiza en la Figura 3.41.

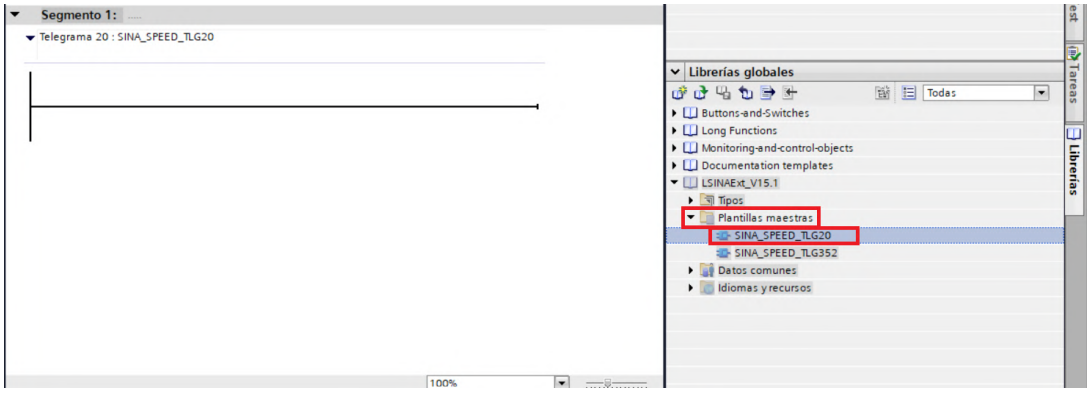


Figura 3.41: Ventana para añadir el bloque SINA_SPEED_TLG20.

En la ventana emergente se crea un DB (Bloque de Datos) propio para los

parámetros del motor. Se selecciona el número deseado y se oprime **"Aceptar"**, conforme se muestra en la Figura 3.42.

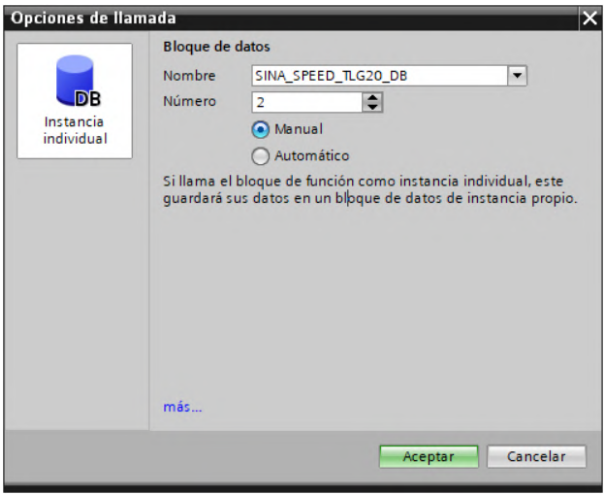


Figura 3.42: bloque DB para el SINA_SPEED_TLG20.

De esta manera se agrega el bloque SINA_SPEED_TLG20 en el bloque MAIN [OB1], y además un Bloque de Datos **SINA_SPEED_TLG20_DB[2]** y un Bloque de Función **SINA_SPEED_TLG20[FB38003]**, los mismos que se visualizan en la Figura 3.43.

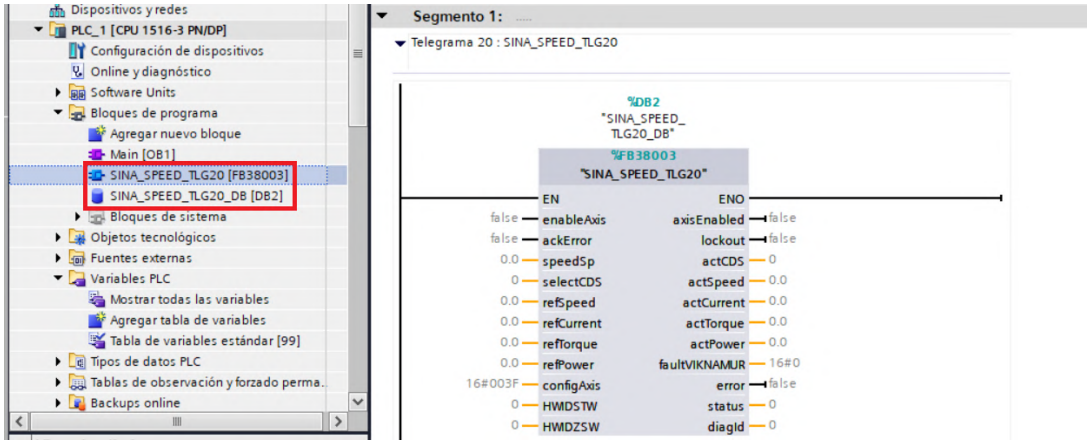


Figura 3.43: Bloque SINA_SPEED_TLG20.

Debido a la versión de firmware presente en el variador SINAMICS G120, es necesario realizar una corrección en el código de programación del Bloque de Función **SINA_SPEED_TLG20[FB38003]** para evitar que el parámetro "error" se active automáticamente. Para ello se accede al bloque y se corrige la línea de código 196 reemplazando el valor **16#0000** por **16#0001**, como se presenta en la Figura 3.44.

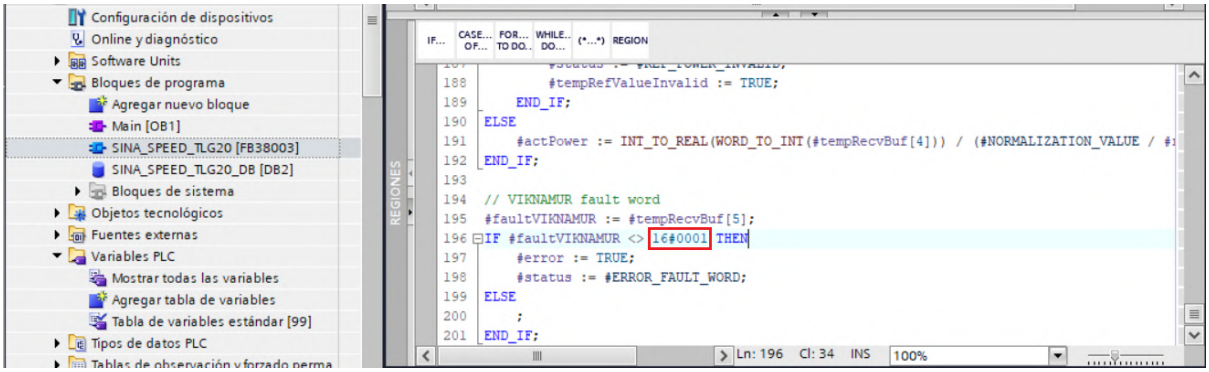


Figura 3.44: Configuración del bloque de función SINA_SPEED_TLG20[FB38003].

Una vez completada la configuración inicial del Bloque SINA_SPEED_TLG20, es necesario ajustar los parámetros de entrada y salida según los requisitos específicos del proyecto. Esto implica adaptar y personalizar los valores de entrada y salida para que el sistema funcione de manera óptima y cumpla con los objetivos y funcionalidades deseadas.

Para lograrlo, en primera instancia, es necesario definir y crear variables que permitan ingresar los datos en los parámetros de entrada y obtener los datos de los parámetros de salida. Estas variables actuarán como contenedores para la información que será enviada o recibida a través del bloque SINA_SPEED_TLG2020.

En el caso de los parámetros de entrada, las variables se utilizarán para establecer los valores que se enviarán al variador SINAMICS G120 para configurar su funcionamiento según las necesidades del sistema.

Por otro lado, las variables asociadas a los parámetros de salida almacenarán la información recibida del variador, como datos de velocidad actual, corriente consumida, torque generado, potencia utilizada, así como el estado de error y posibles advertencias que puedan surgir durante el funcionamiento del motor trifásico de inducción.

Al personalizar y ajustar adecuadamente estos parámetros de entrada y salida, se logrará una configuración óptima del Bloque SINA_SPEED_TLG20, permitiendo el control y monitoreo eficiente del motor y garantizando el correcto funcionamiento del sistema en el entorno industrial.

En la Figura 3.45 se presenta la tabla de variables estándar del proyecto, con las variables necesarias para la parametrización del bloque SINA_SPEED_TLG20.

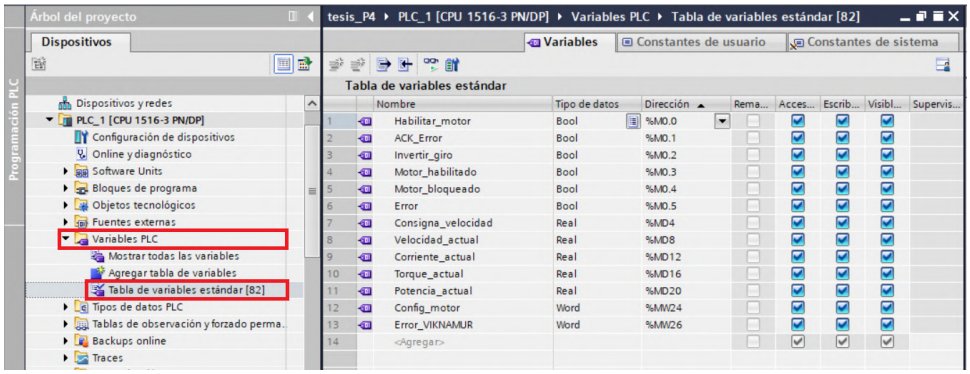


Figura 3.45: Tabla de variables estándar.

Con las variables creadas y los parámetros específicos del motor utilizado en el proyecto, se realiza la vinculación entre estas variables y los parámetros del bloque SINA_SPEED_TLG20 para su correcto funcionamiento. Esta vinculación se realiza mediante la asignación de los valores correspondientes a cada uno de los parámetros del bloque, como se puede visualizar en la Figura 3.46.

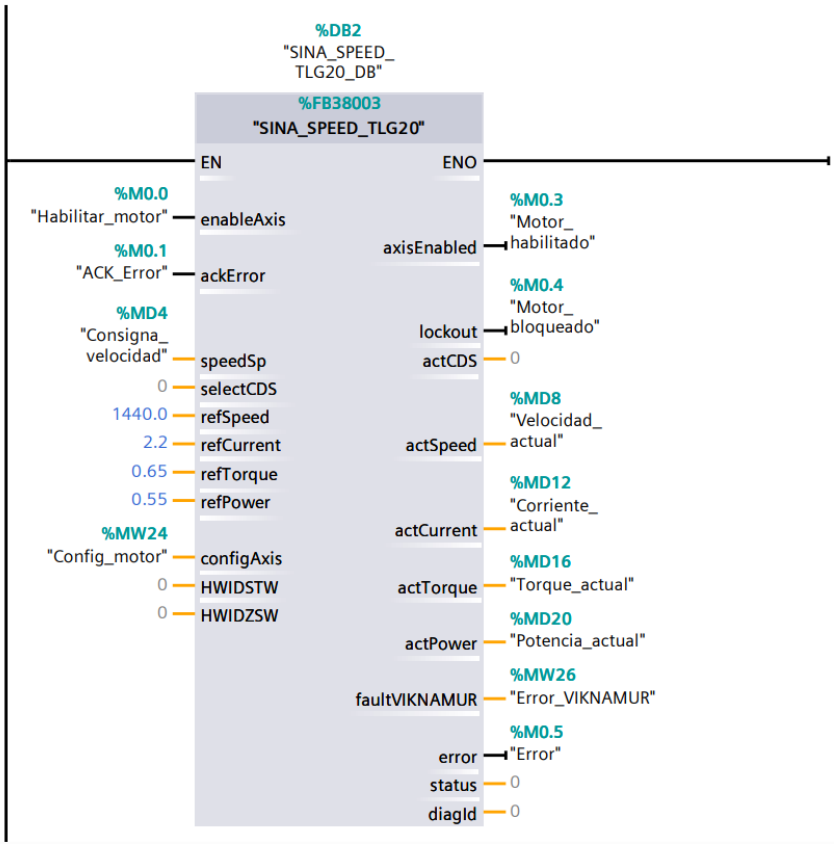


Figura 3.46: Parámetros iniciales del bloque SINA_SPEED_TLG20.

Los parámteros **HWIDSTW** y **HWIDZSW**, como se estableció en la Tabla 3.2, hacen referencia a un identificador de hardware de la ranura de consigna y un identificador de hardware de la ranura de valor real, respectivamente; y su configuración se realiza según se presenta en las Figuras 3.47 y 3.48.

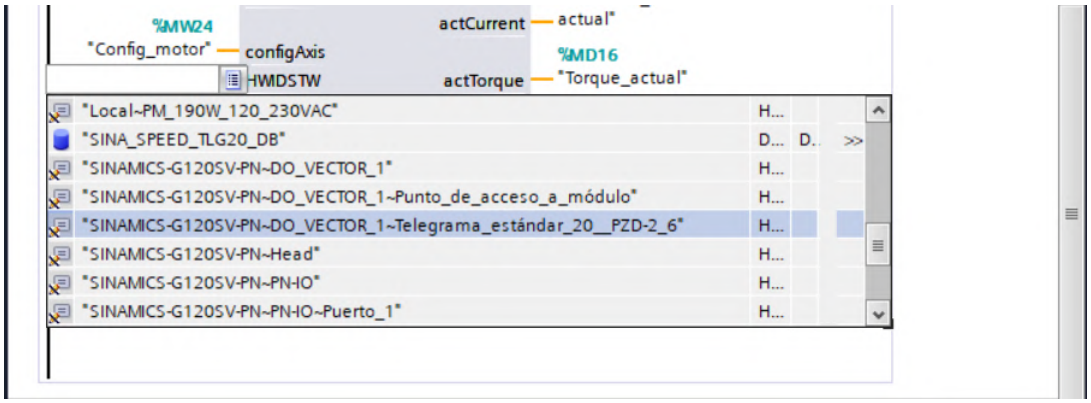


Figura 3.47: Configuración del parámetro HWIDSTW.

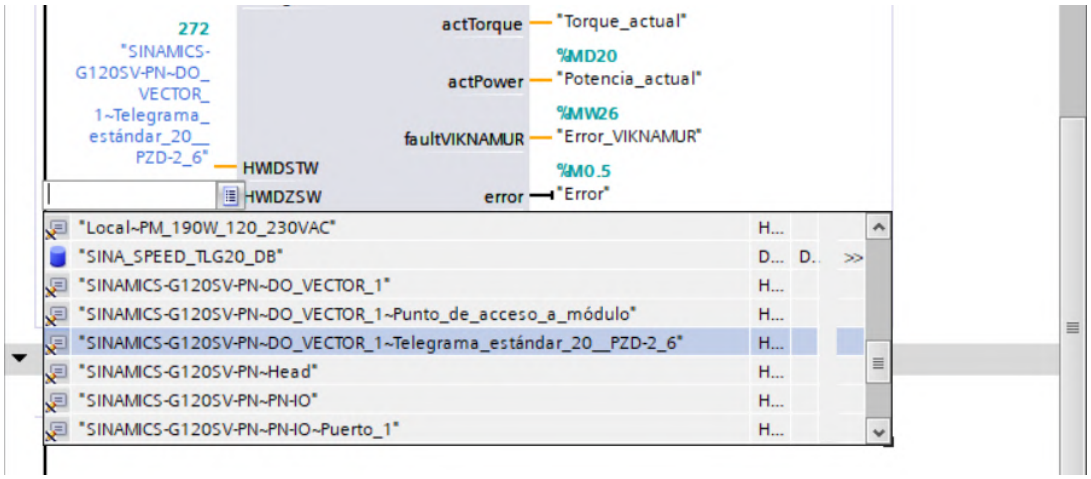


Figura 3.48: Configuración del parámetro HWIDZSW.

De esta manera, una vez que se han realizado todas las configuraciones y se han vinculado las variables y parámetros de acuerdo con las necesidades específicas del proyecto, el bloque SINA_SPEED_TLG20 queda completamente configurado y listo para su uso en el desarrollo del proyecto.

En la Figura 3.49 se visualiza el bloque con todos sus parámetros establecidos.

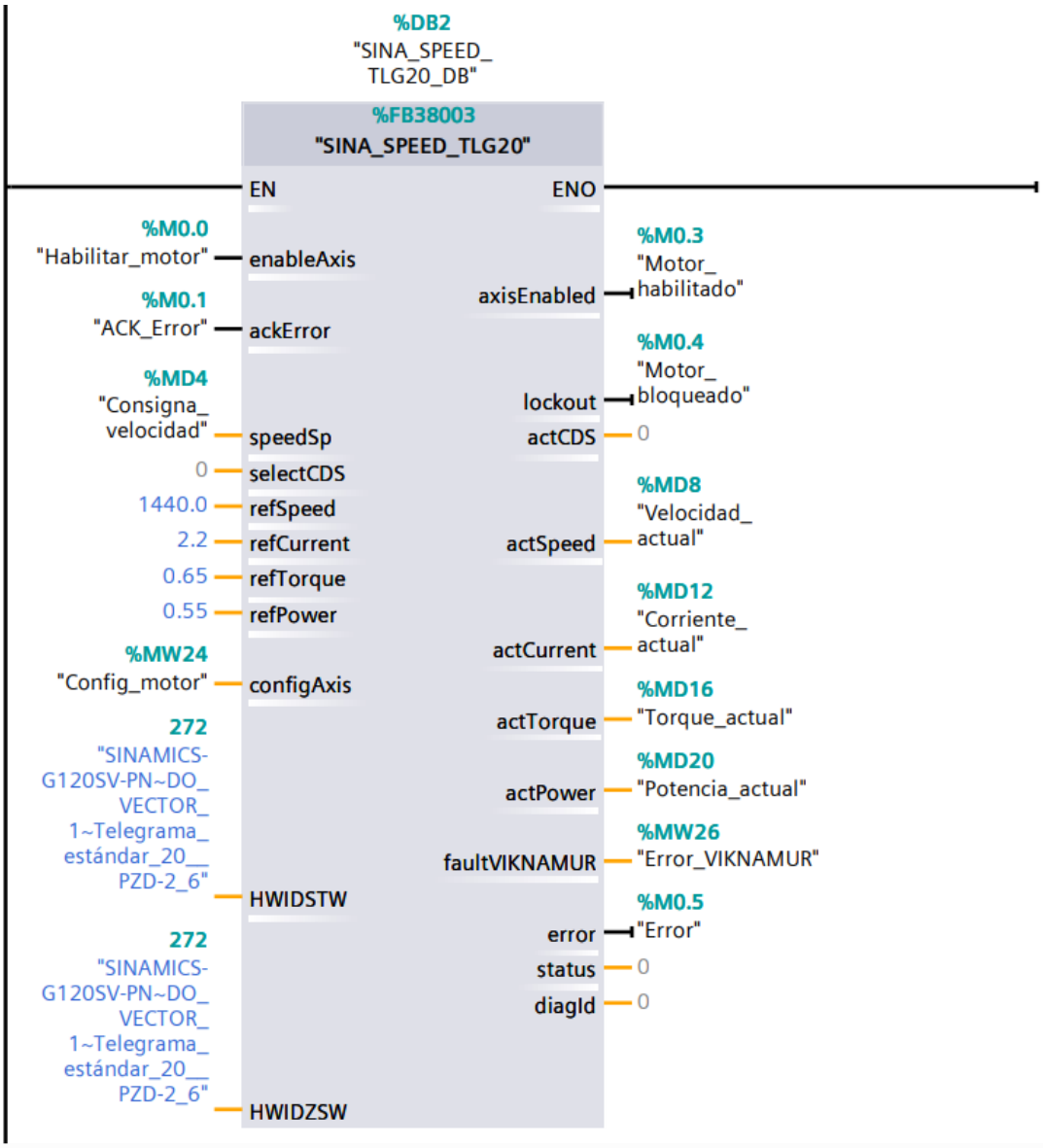


Figura 3.49: Bloque SINA_SPEED_TLG20 configurado.

3.2.4. Implementación de un Bloque de Datos para la comunicación con el HMI

Una vez configurado el telegrama de comunicación es necesario la creación de un **DB (Bloque de Datos)**, que permita el intercambio de datos entre los parámetros del bloque SINA_SPEED_TLG20 y la HMI.

La creación del DB es un paso esencial para lograr el intercambio de datos entre los parámetros del bloque SINA_SPEED_TLG20 y el HMI. El DB actúa como un medio para compartir información entre estos dos componentes del sistema industrial. En el DB, se definen las variables y estructuras de datos necesarias para almacenar y transmitir los valores de velocidad, corriente, torque, potencia, estado de error y otros datos relevantes obtenidos del variador SINAMICS G120 a la HMI. Este bloque garantiza que la información se mantenga organizada y estandarizada, lo que facilita la comunicación y el procesamiento de datos entre los diferentes dispositivos.

Incorporación del Bloque de Datos

En primer lugar, se debe agregar un DB al proyecto. Para ello, dentro de TIA Portal, en el **"Arbol del proyecto"** se busca y selecciona la carpeta **"Bloques de programa"** donde se desea agregar el nuevo bloque. Luego, haciendo clic sobre la carpeta seleccionada, se despliega un menú y se elige la opción **"Agregar nuevo bloque"**, de acuerdo a lo ilustrado en la Figura 3.50.

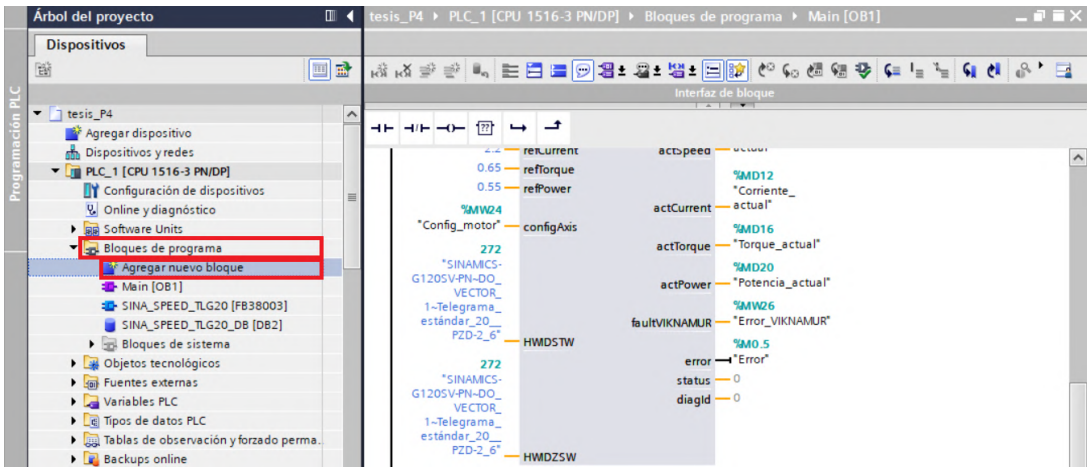


Figura 3.50: Procedimiento para agregar un bloque de programa.

De esta manera se abre una ventana emergente que permite elegir el tipo de bloque a crear. En este caso, se selecciona **"DB" (Bloque de Datos)** como tipo de bloque. Luego, se asigna un nombre al DB y se presiona **"Aceptar"** para crear el bloque (véase la figura 3.51).

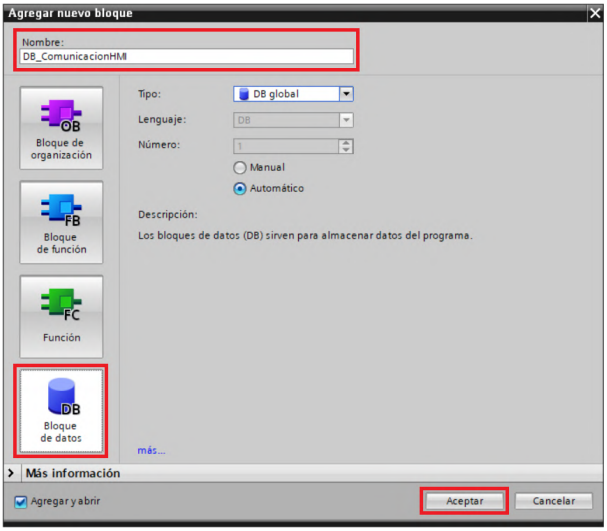


Figura 3.51: Ventana para agregar un nuevo bloque.

En la Figura 3.52, se muestra el Bloque de Datos (DB) que ha sido agregado exitosamente al proyecto. Este DB actúa como un área de memoria en la cual se almacenarán y compartirán los datos relevantes para la comunicación entre el bloque SINA_SPEED_TLG20 y el HMI.

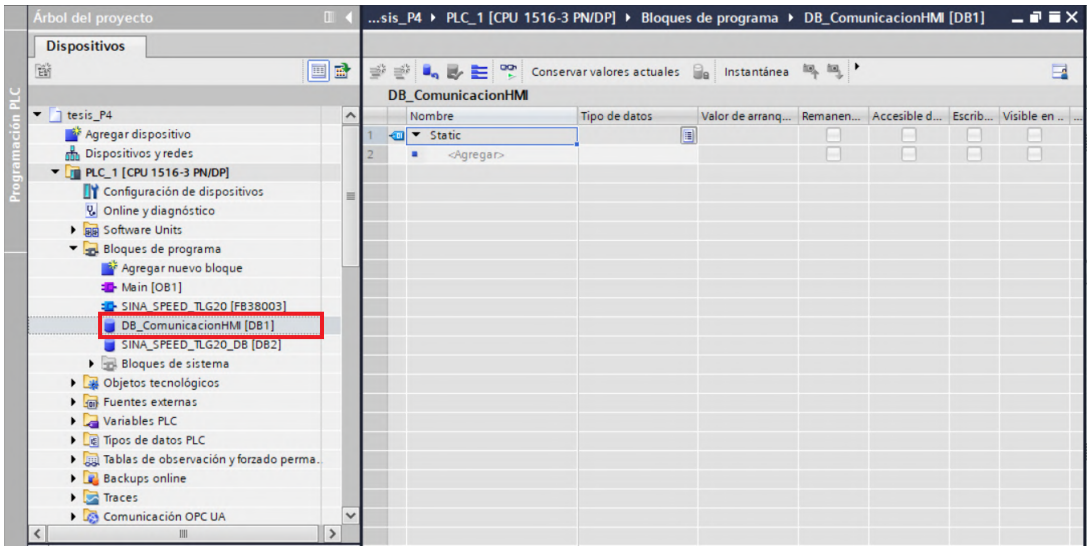


Figura 3.52: Bloque de Datos agregado en el proyecto.

Configuración de variables

Una vez creado el Bloque de Datos, se procede a configurar las variables que actúan como contenedores de información que serán actualizadas y compartidas entre el bloque SINA_SPEED_TLG20 y el HMI. Estas variables permitirán almacenar y transmitir los datos relevantes del variador SINAMICS G120, así como las consignas y comandos provenientes del HMI.

El Bloque de Datos con las variables establecidas se presenta en la Figura 3.53.

	Nombre	Tipo de datos	Offset	Valor de arranq...	Instantánea	Remanen...	Accesible d...	Esc...
1	Static							
2	Velocidad_Act	Real	0.0	0.0	0.0			
3	Corriente_Act	Real	4.0	0.0	0.001074219			
4	Torque_Act	Real	8.0	0.0	0.0			
5	Potencia_Act	Real	12.0	0.0	0.0			
6	Consigna_Vel	Real	16.0	0.0	0.0			
7	Habilitar_Motor	Bool	20.0	false	FALSE			
8	Inv_Giro	Bool	20.1	false	FALSE			
9	Conexion_Estado	Bool	20.2	false	--			
10	Error_Boolean	Bool	20.3	false	--			
11	Error_Acusar	Bool	20.4	false	--			
12	Error_Vik	Word	22.0	16#0	--			
13	IP	String	24.0	'0.0.0.0'	--			

Figura 3.53: Variables del Bloque de Datos.

Intercambio de datos

Una vez creadas las variables dentro del Bloque de Datos, es necesario establecer el intercambio de datos entre estas variables y las que están vinculadas a los parámetros del bloque SINA_SPEED_TLG20. Este intercambio de datos permitirá que la información relevante se transmita correctamente entre ambos componentes del sistema de automatización.

Para ello, en TIA Portal se realiza la programación de la lógica necesaria para el intercambio de datos en ambos bloques.

En la Figura 3.54, se presenta la programación para habilitar el motor. Para lograrlo, se obtiene el valor de la variable del Bloque de Datos y se transmite a la variable de habilitación del bloque SINA_SPEED_TLG20. De esta manera, se establece

la comunicación entre ambos bloques, permitiendo que el motor pueda ser activado o desactivado según el valor recibido. Esta configuración posibilita el control del motor de forma remota desde el HMI.

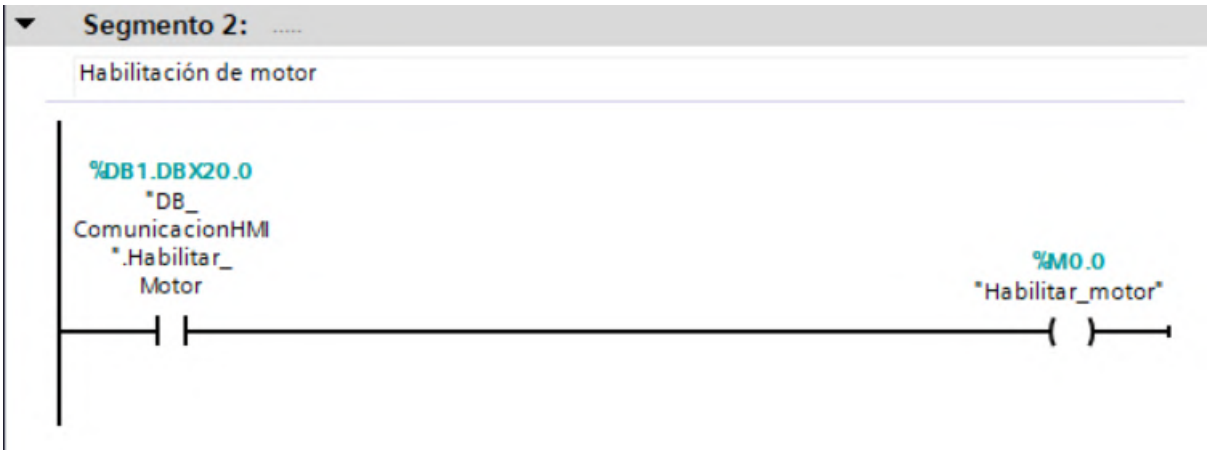


Figura 3.54: Habilitación del motor mediante el HMI.

En la Figura 3.55 se puede observar la programación para configurar el sentido de giro del motor. En este caso, se obtiene el valor de la variable relacionada con la inversión de giro del Bloque de Datos. Dependiendo de su estado, se envía la palabra de control correspondiente al parámetro de configuración del bloque SINA_SPEED_TLG20.

Las palabras definidas en la imagen permiten determinar el sentido de giro del motor de la siguiente manera:

- **16#003F:** Giro en sentido directo
- **16#007F:** Giro en sentido inverso

En la Figura 3.56 se muestra la programación para obtener el valor de la consigna de velocidad para la operación del motor. Para lograrlo, se obtiene el valor de la variable del Bloque de Datos y se transmite a la variable de consigna de velocidad del bloque SINA_SPEED_TLG20 mediante el comando MOVE. Esta configuración permite controlar y ajustar la velocidad del motor de forma remota a través del HMI.

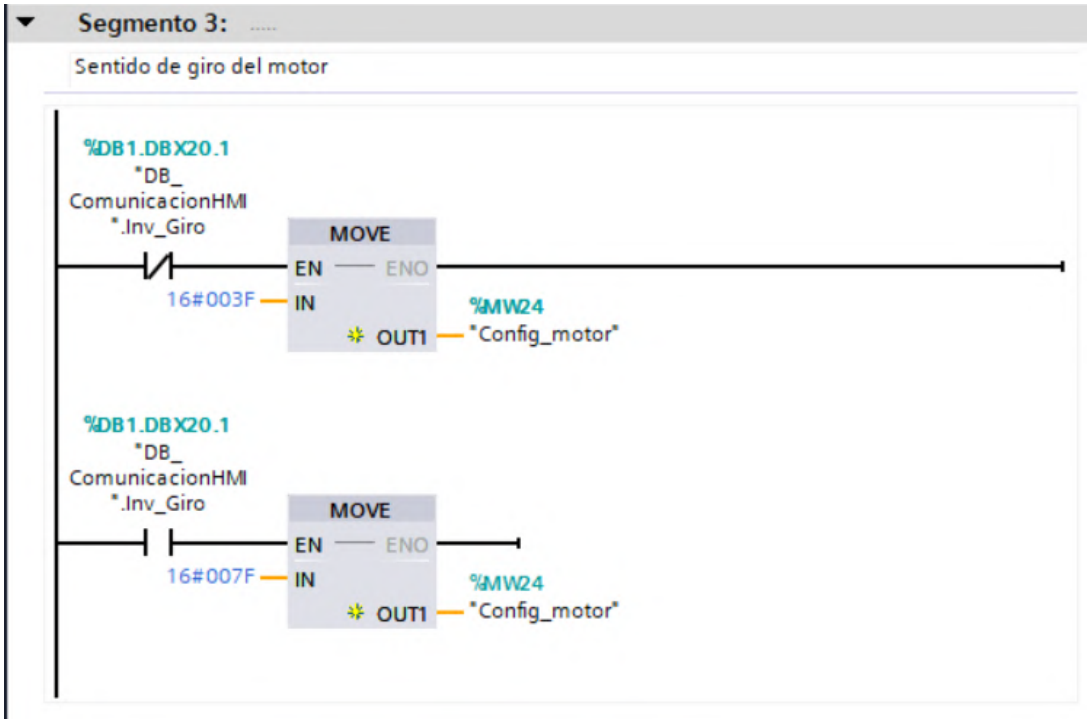


Figura 3.55: Configuración del sentido de giro mediante el HMI.

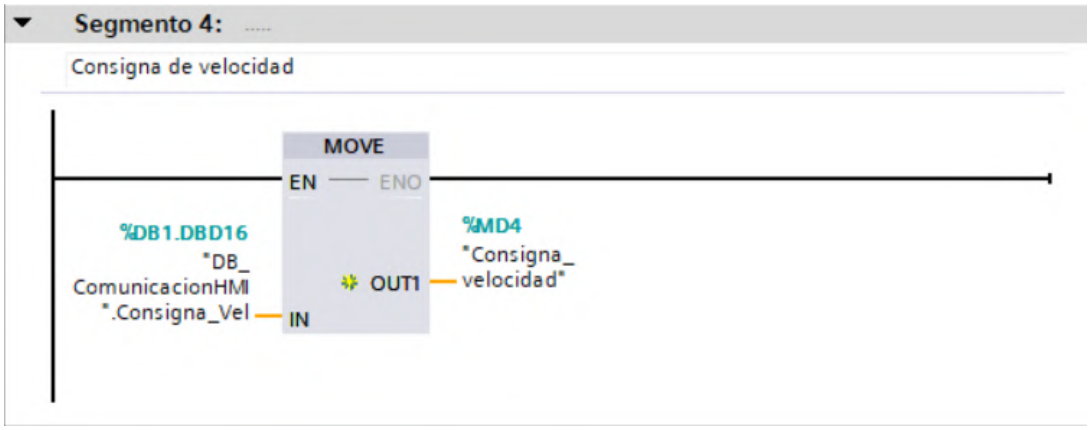


Figura 3.56: Control de la consigna de velocidad mediante el HMI.

En la Figura 3.57 se presenta la programación para obtener los datos de velocidad, corriente, torque y potencia del motor a través del bloque SINA_SPEED_TLG20 y actualizarlos dentro de las variables correspondientes en el Bloque de Datos. Esta configuración permite adquirir y almacenar de manera periódica los datos del motor, lo que facilita su posterior monitoreo y visualización en el HMI.

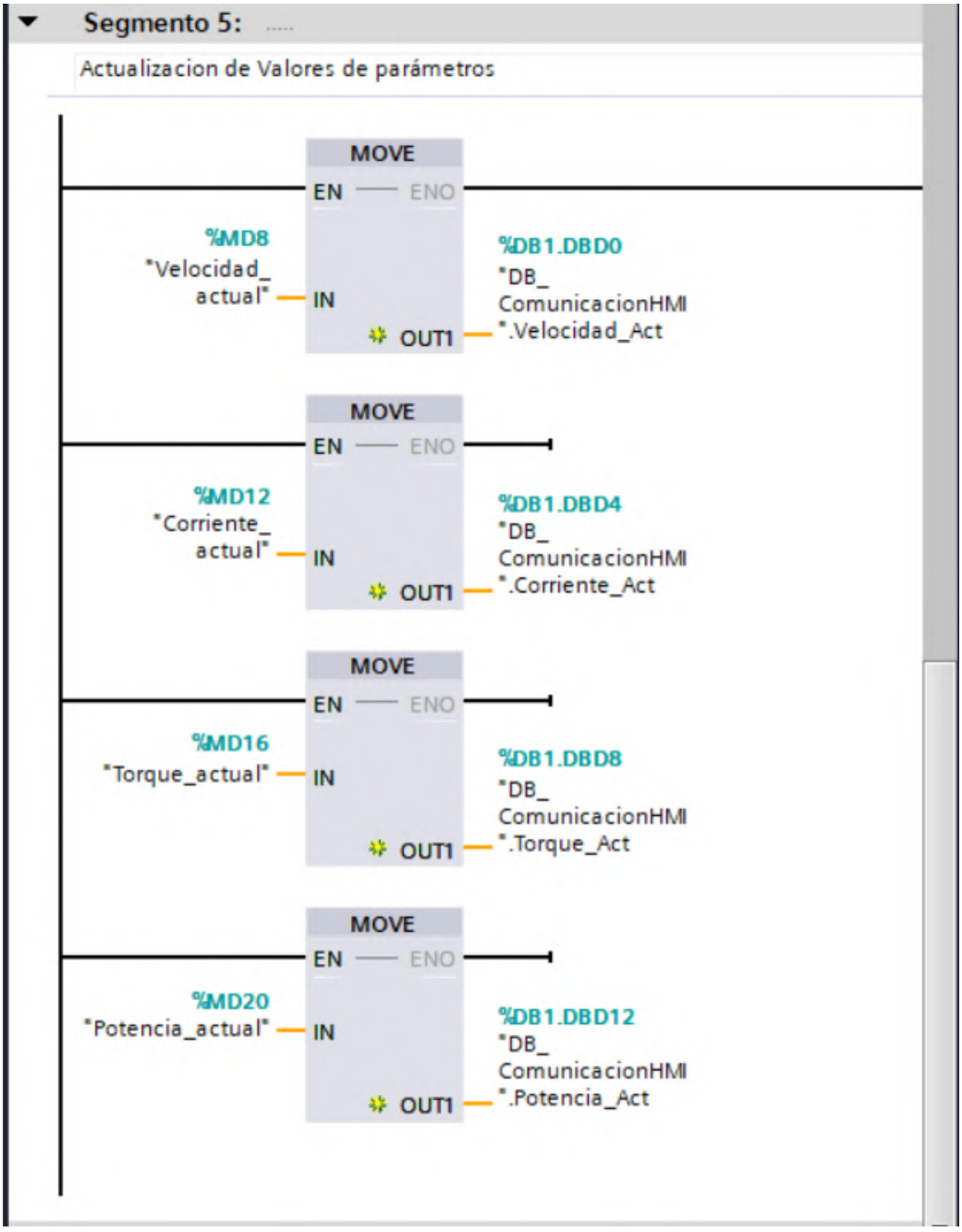


Figura 3.57: Obtención de los parámetros del motor para el HMI.

En la Figura 3.58, se muestra la programación para obtener la información de error del motor. Para lograrlo, se obtiene el valor booleano del estado de error y la palabra de error del bloque SINA_SPEED_TLG20, y luego se actualizan dentro del Bloque de Datos para tener esta información disponible en el HMI. Esta configuración permite monitorear y registrar posibles errores o fallas que puedan ocurrir en el motor. De esta manera, el HMI puede mostrar mensajes de advertencia o alarmas que alerten a los operadores sobre problemas en el motor o en el sistema de control.

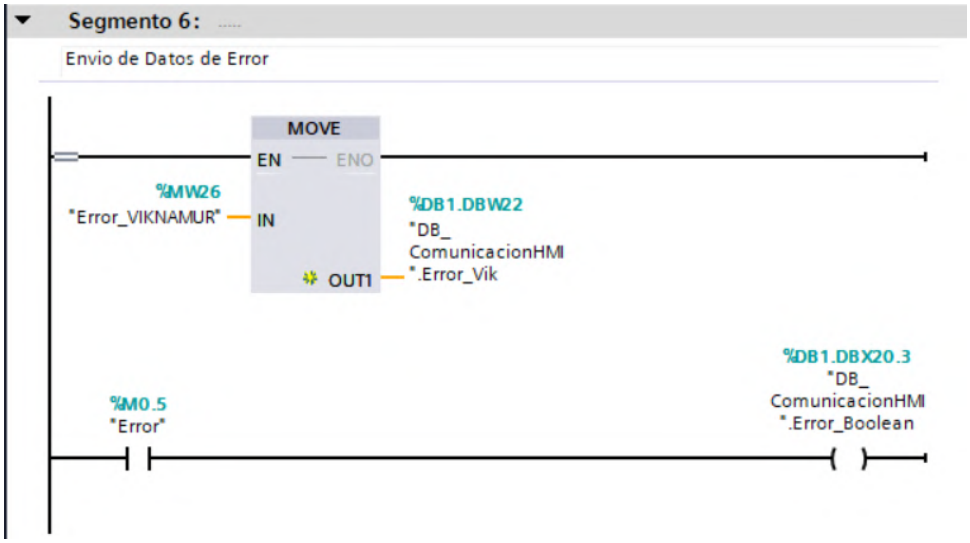


Figura 3.58: Obtención de información del estado de error para el HMI.

Finalmente, en la Figura 3.59, se muestra la programación que permite el acuso de fallos a través del HMI. Para lograrlo, se obtiene el estado de la variable relacionada con el acuso de fallos del Bloque de Datos y se transmite a la variable correspondiente del bloque SINA_SPEED_TLG20.

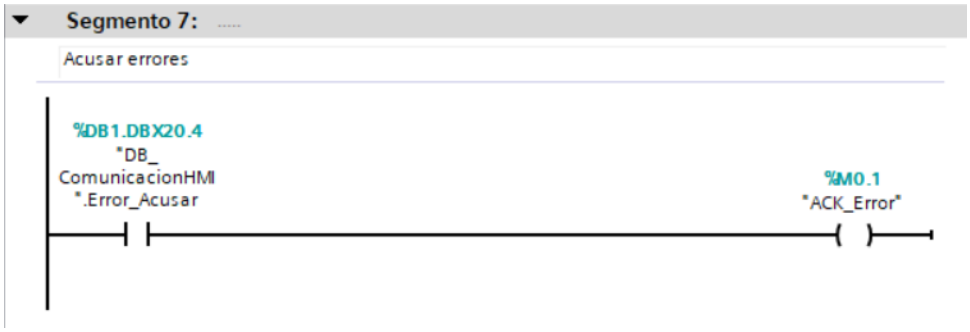


Figura 3.59: Reconocimiento de fallos mediante el HMI.

3.3. Configuración del módulo SCALANCE W774

Al concluir la configuración del sistema de control del motor trifásico de inducción, se procede a configurar el módulo SCALANCE W774 para habilitar la comunicación inalámbrica del sistema con el HMI.

Esta configuración del módulo SCALANCE W774 se lleva a cabo en dos etapas. En primera instancia, se realiza una configuración inicial mediante el software SINEC PNI. Este proceso permite establecer los parámetros básicos necesarios para la comunicación inalámbrica y asegurar la conectividad entre los dispositivos.

Una vez completada la configuración inicial con SINEC PNI, se procede a una segunda etapa de configuración más avanzada utilizando el Web Based Management (WBM). A través de esta interfaz, se pueden realizar ajustes más detallados y personalizados para garantizar un rendimiento óptimo de la red inalámbrica. Esto incluye la configuración de seguridad, intensidad de la señal, canales permitidos y otros parámetros que son esenciales para asegurar una comunicación inalámbrica estable y segura.

Con la configuración del módulo SCALANCE W774 completada, el sistema de control estará listo para establecer una comunicación efectiva y confiable con el HMI. Esto permitirá visualizar y monitorear el funcionamiento del motor trifásico de inducción de manera remota y realizar las operaciones de control necesarias desde la estación de monitoreo.

3.3.1. Configuración inicial mediante SINEC PNI

Una vez que el programa esté abierto, se procede a seleccionar la pestaña **"Settings" (Configuración)**, como se visualiza en la Figura 3.60. Aquí se pueden ajustar y seleccionar los parámetros para el reconocimiento de dispositivos Profinet. En esta sección, se podrán realizar las configuraciones necesarias para establecer la comunicación y reconocimiento adecuado de los dispositivos conectados a través de la red Profinet.

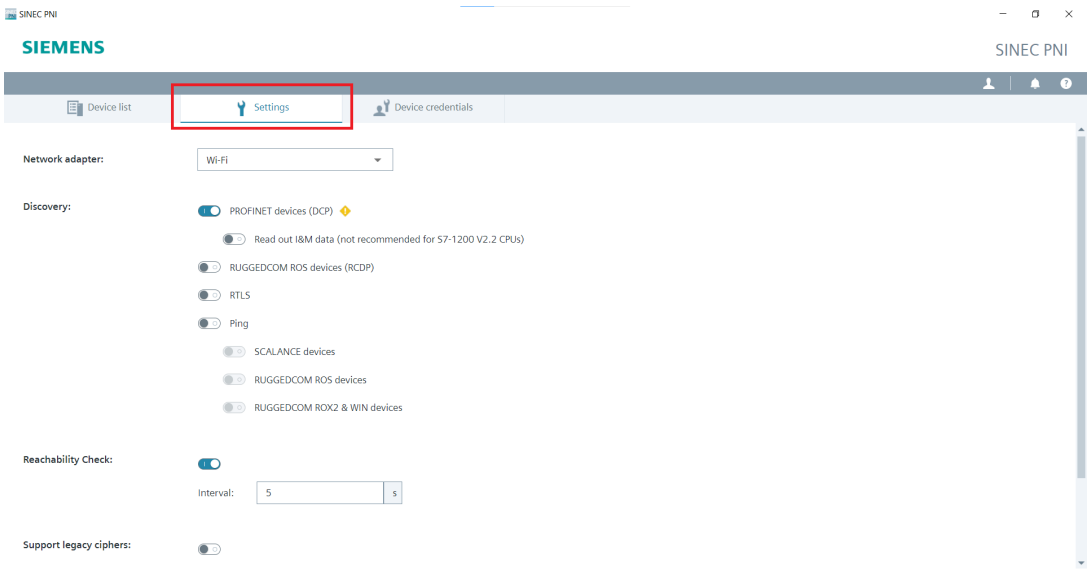


Figura 3.60: Interfaz SINEC PNI, Settings.

En la opción **"Network adapter" (Adaptador de red)**, se procede a seleccionar el adaptador de red que se utilizará para reconocer los dispositivos. En el caso específico del proyecto, se elige la opción **"Ethernet"** (véase la figura 3.61) como el adaptador de red preferido, debido a la necesidad de integrar el ordenador utilizado en la configuración a través de este medio.

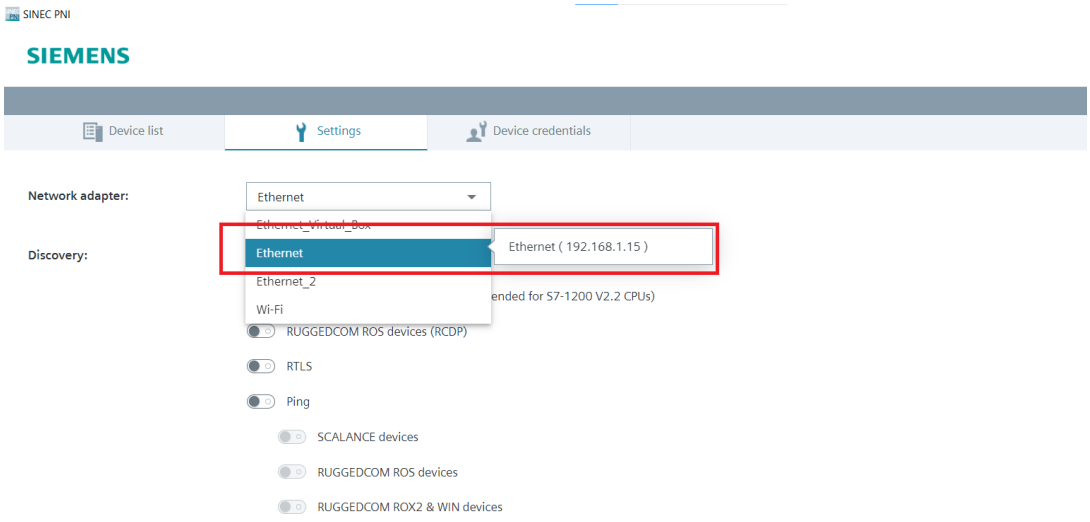


Figura 3.61: Selección del adaptador de red.

En la sección **"Discovery" (Descubrimiento)** , procedemos a habilitar la opción **"PROFINET devices (DCP)"** (véase la figura 3.62). Al activar esta opción, permitimos que el módulo SCALANCE W774 realice un proceso de descubrimiento

de dispositivos Profinet en la red.

El protocolo DCP (Discovery and Basic Configuration Protocol) es utilizado en Profinet para detectar automáticamente los dispositivos conectados en la red y obtener información básica sobre ellos, como sus direcciones IP y nombres.

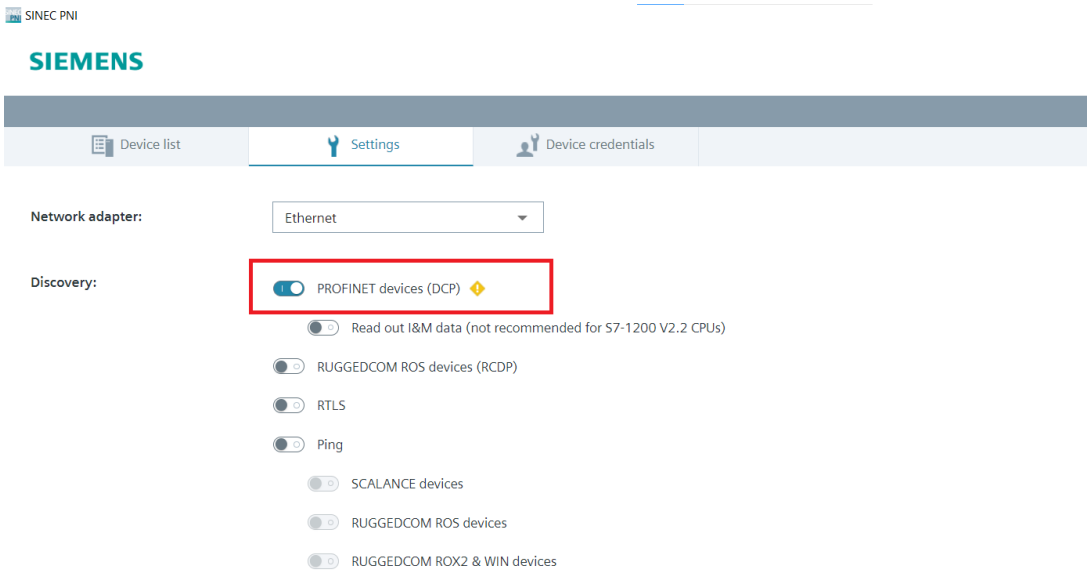


Figura 3.62: Selección del método de reconocimiento de dispositivos.

Los demás parámetros que no han sido mencionados previamente, se mantienen en sus valores por defecto para asegurar una configuración adecuada y compatible con el sistema, y se guardan los cambios realizados presionando el botón **"Save"** (véase la figura 3.63).

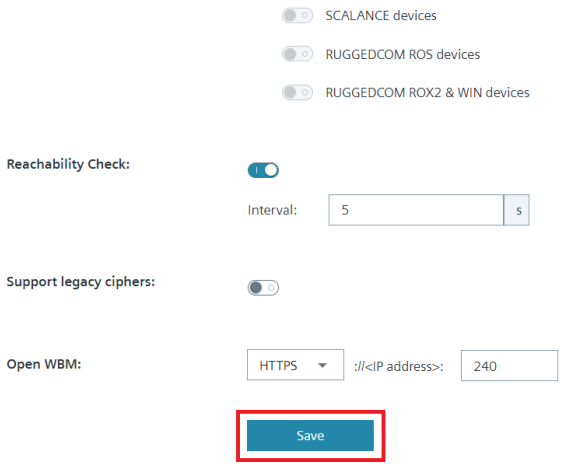


Figura 3.63: Guardar configuración de búsqueda de dispositivos.

Una vez que los parámetros de búsqueda de dispositivos han sido configurados, es necesario realizar un escaneo de la red para detectar los dispositivos disponibles. Para ello, en la pestaña **"Device List" (Lista de dispositivos)**, se debe hacer clic en el botón **"Start Network Scan"(Iniciar escaneo de red)** (véase la figura 3.64).

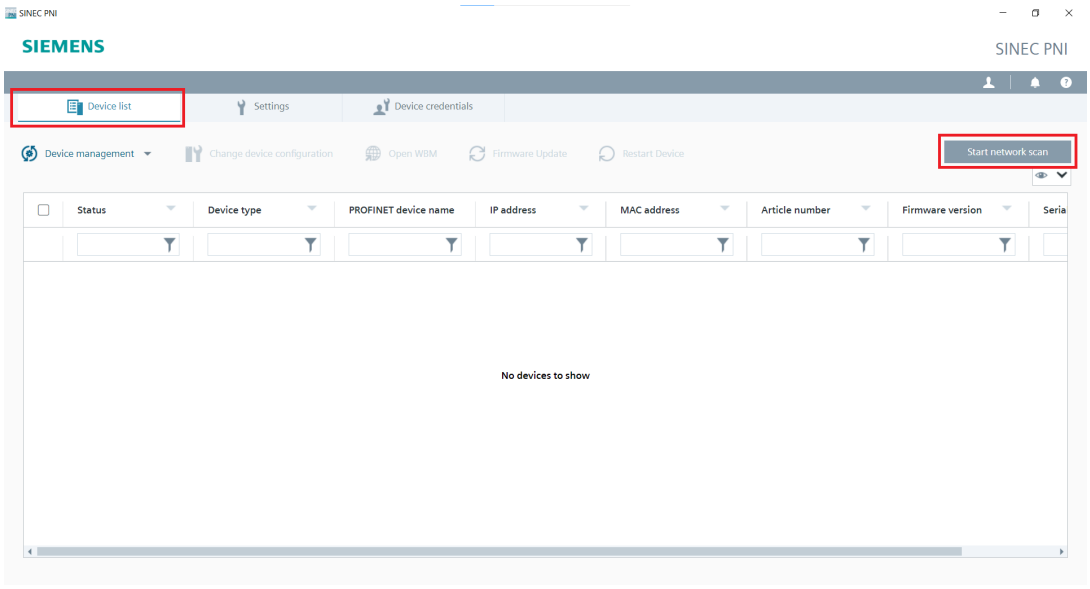


Figura 3.64: Escaneo de dispositivos Profinet.

Este proceso de escaneo de red permite buscar y detectar los dispositivos conectados mediante Profinet. Durante el escaneo, el software envía mensajes de descubrimiento a los dispositivos presentes en la red para identificarlos y obtener información sobre ellos, como su dirección IP y otros detalles relevantes.

El resultado del escaneo se muestra en la lista de dispositivos, donde se enumeran todos los dispositivos encontrados y sus respectivas direcciones IP. De esta manera, se podrá visualizar qué dispositivos están activos y disponibles en la red.

En la Figura 3.65 se puede observar el resultado del escaneo de la red aplicado en el proyecto para detectar el módulo SCALANCE W774. En la lista de dispositivos encontrados, se muestra el tipo de dispositivo, la dirección IP asignada y la dirección MAC del dispositivo. Con este procedimiento se confirma su presencia y disponibilidad dentro de la red.

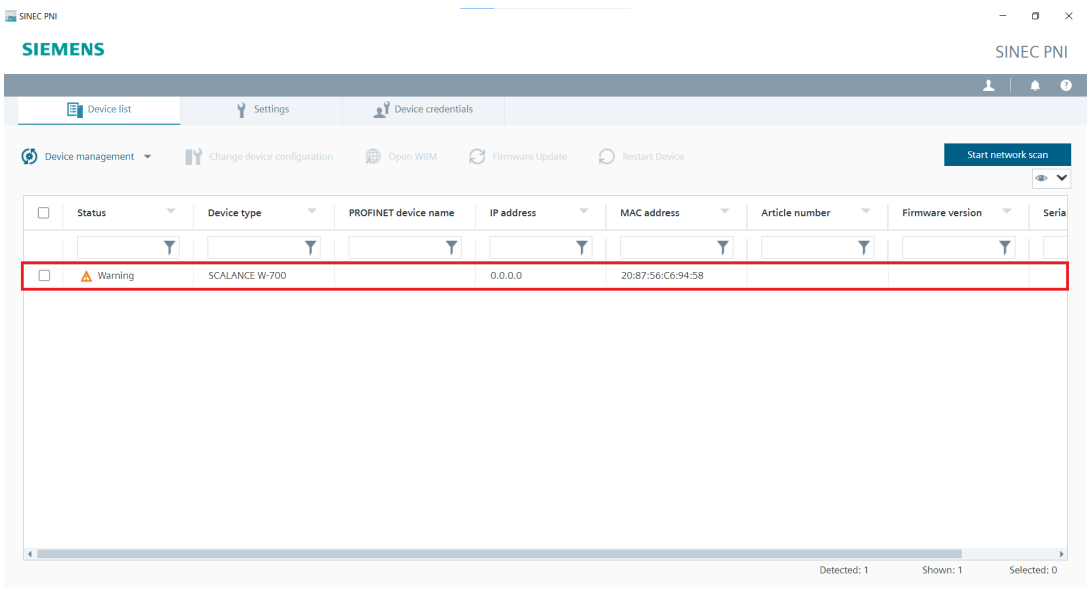


Figura 3.65: Dispositivos de red detectados.

Realizado el escaneo y reconocimiento del módulo conectado, es posible configurarlo según las necesidades del proyecto. Para ello, se procede a seleccionar el módulo de la lista de dispositivos detectados y se presiona la opción **"Device Management" (Gestión de Dispositivos)** en el software (véase la figura 3.66).

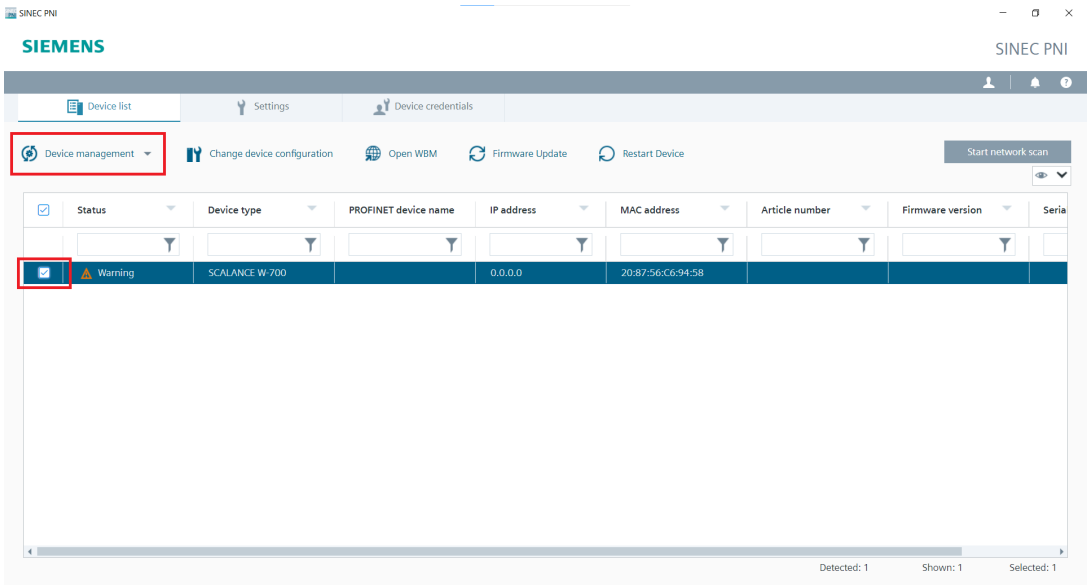


Figura 3.66: Interfaz SINEC PNI, Device List.

De entre la opciones que se despliegan, se debe seleccionar **"Change device configuration" (Cambiar Configuración del Dispositivo)**, que permite modificar las configuraciones específicas del módulo SCALANCE W774 (véase la figura 3.67).

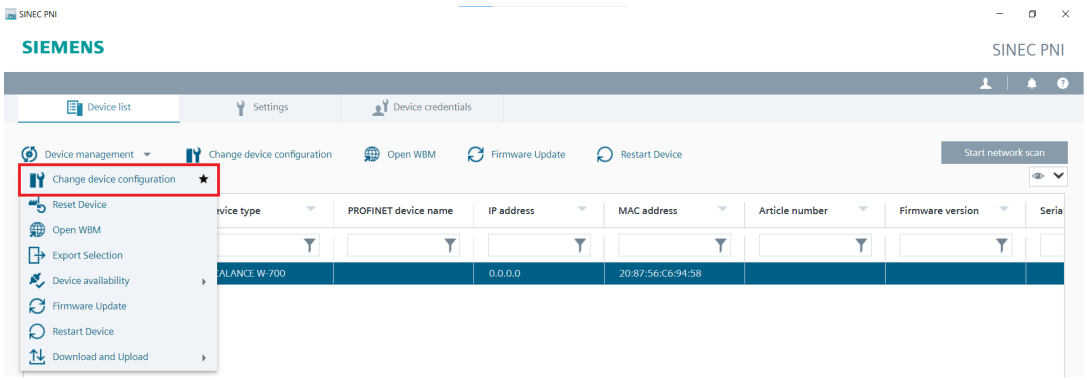


Figura 3.67: Device Management.

Dentro de la ventana **"Device Configuration"** que se muestra en la Figura 3.68, es necesario dirigirse a la pestaña **"IP Configuration" (Configuración de IP)**. En esta sección, se pueden definir la dirección IP y la máscara de subred del módulo SCALANCE W774. Para ello, se ingresa la dirección IP previamente definida en la Tabla 3.6 y se asegura de que la opción **"DHCP"** esté deshabilitada, ya que la dirección IP se está configurando manualmente.

Una vez que se han ingresado los parámetros deseados, se procede a presionar el botón **"LOAD"** para que la nueva configuración de la dirección IP y la máscara de subred se apliquen al módulo. De esta manera, el dispositivo estará listo para operar en la red inalámbrica con la dirección IP definida y la máscara de subred configurada según los requisitos del proyecto.

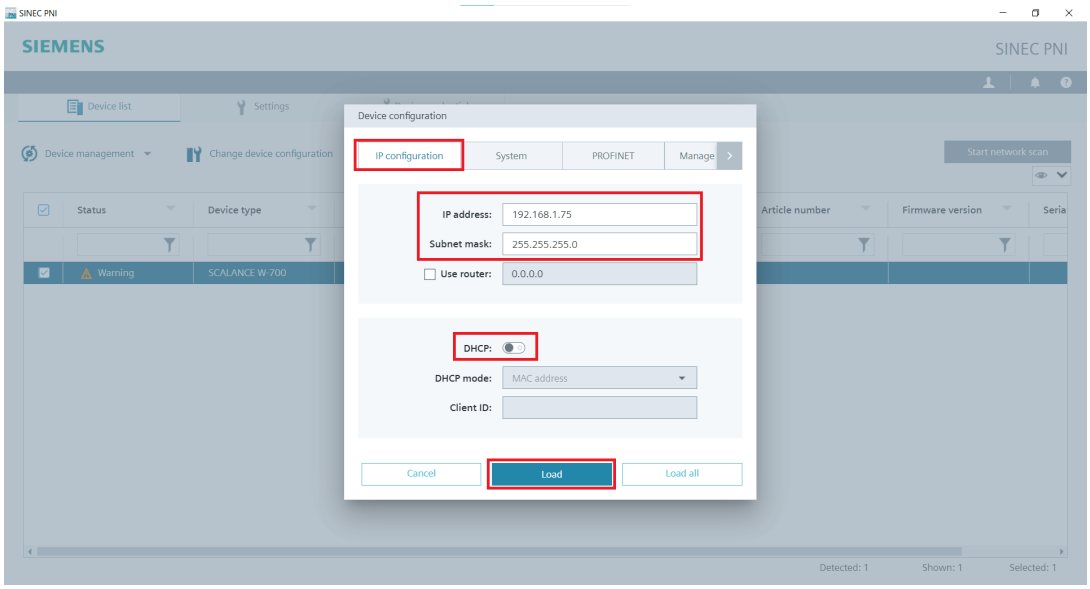


Figura 3.68: Device Configuration.

Una vez finalizada la configuración del módulo SCALANCE W774, los cambios efectuados en sus parámetros se reflejan en la lista de dispositivos, tal como se muestra en la Figura 3.69. En esta lista, se puede comprobar que el estado del módulo aparece como **OK**, lo que indica que la configuración se ha realizado correctamente y el módulo se encuentra en funcionamiento adecuado. Además, se verifica que la dirección IP ha sido actualizada según la configuración realizada previamente.

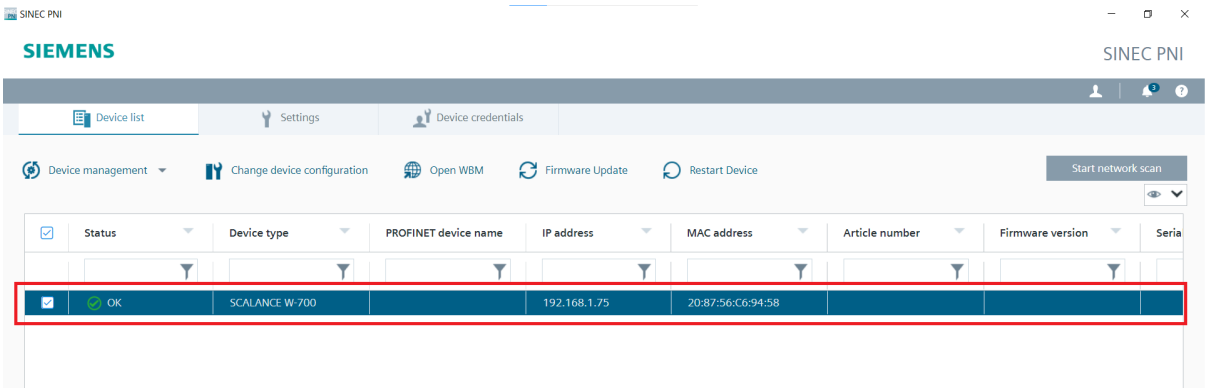


Figura 3.69: Verificación de la configuración del SCALANCE W774.

3.3.2. Configuración avanzada mediante WBM

Para una configuración más avanzada del módulo SCALANCE W774, se utiliza el Web Based Management (WBM) que es una interfaz basada en web proporcionada por el módulo. A través de WBM, se pueden realizar diversas tareas y ajustes adicionales para optimizar el rendimiento y la funcionalidad del módulo, ya que permite ajustar parámetros de red, seguridad y servicios adicionales.

Para iniciar la configuración a través de WBM, es necesario ajustar la dirección IP del ordenador asignando una IP dentro de la misma red configurada para el módulo SCALANCE W774.

Para ello, se debe acceder a las propiedades del adaptador de red Wi-Fi del ordenador y asginar una IP y una máscara de subred. En este caso, se trabaja con la dirección IP **192.168.1.50** y la máscara de subred **255.255.255.0**, como se ve en la Figura 3.70. Esta configuración permitirá la comunicación inalámbrica mediante TCP/IP entre el ordenador y el módulo SCALANCE W774, facilitando así la configuración y personalización avanzada del módulo para el proyecto.

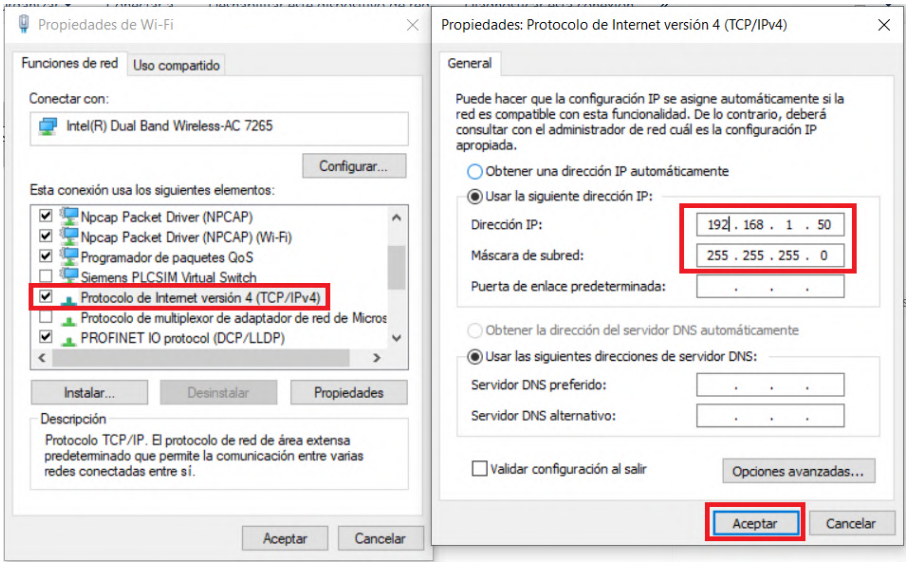


Figura 3.70: Configuración del adaptador de red Wi-Fi.

Para verificar la comunicación entre los dispositivos, se puede realizar una prueba utilizando el terminal de Windows. Para ello, se puede enviar un comando ping a la dirección IP asignada al módulo SCALANCE W774 y verificar si se reciben los datos correctamente. Este procedimiento se ilustra en la Figura 3.71.

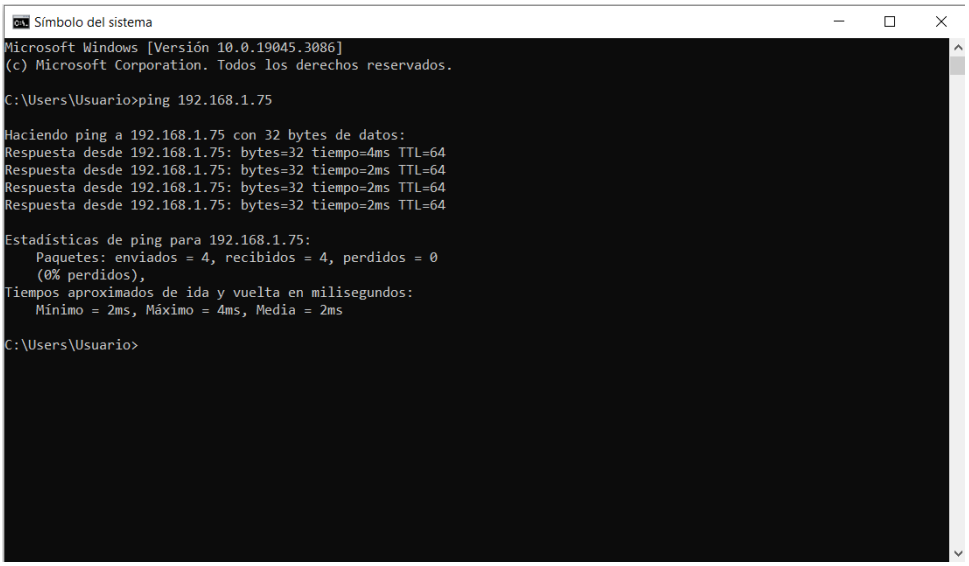


Figura 3.71: Aplicación del comando ping para verificar la conexión entre el PC y el módulo SCALANCE W774.

Comprobada la comunicación entre el ordenador y el módulo, se procede con la configuración avanzada a través del WBM.

Para acceder a esta interfaz, se debe ingresar la dirección IP del módulo en el navegador web. Al hacerlo, se abrirá la página de inicio de autenticación del SCALANCE W774, conforme se evidencia en la Figura 3.72.

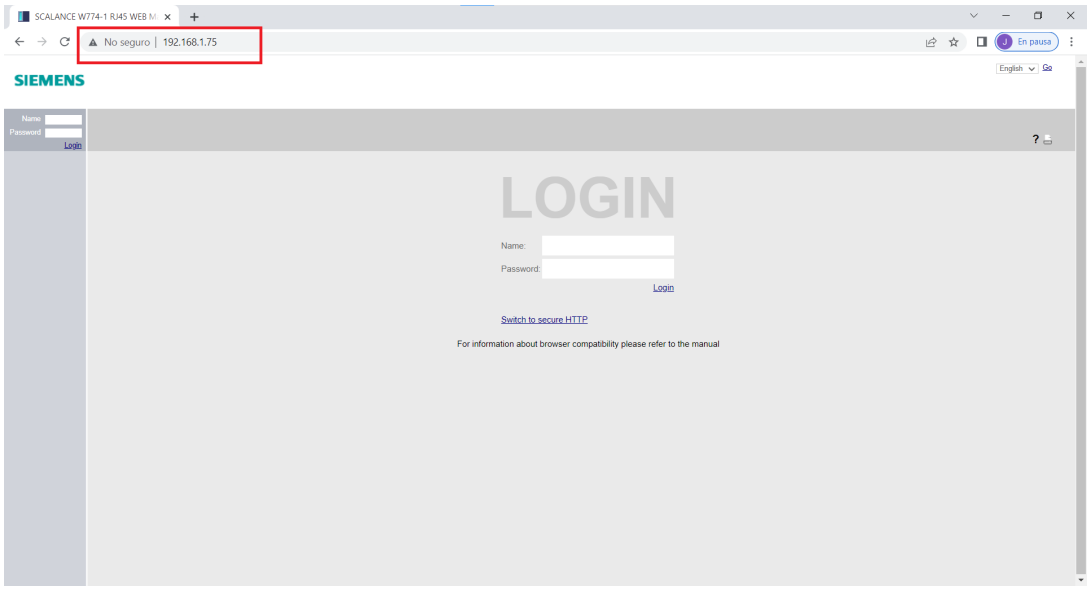


Figura 3.72: WBM, Ventana de Autenticación.

Las credenciales por defecto para acceder a la configuración de los dispositivos SCALANCE W700 son las siguientes:

- **Name:** admin
- **Password:** admin

Una vez ingresadas las credenciales, el software del SCALANCE W700 solicitará cambiar la contraseña por motivos de seguridad, conforme se observa en la Figura 3.73.

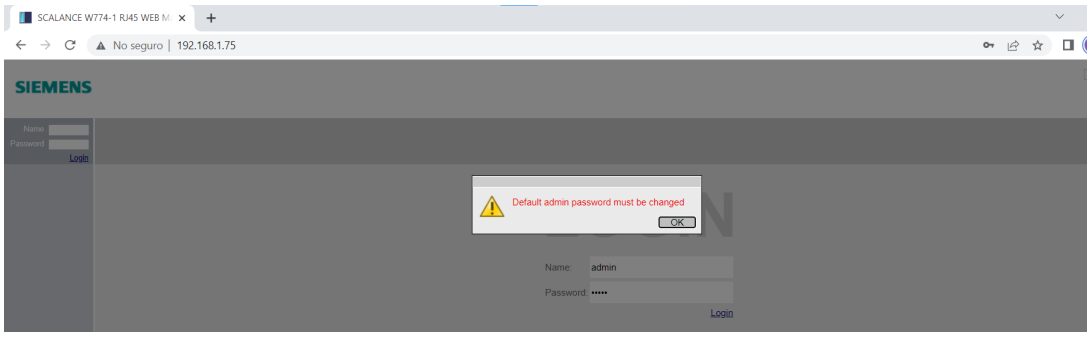


Figura 3.73: WBM, Solicitud de cambio de contraseña.

En la Figura 3.74 se puede apreciar la ventana de cambio de contraseña. Para facilitar la gestión y con fines prácticos en el presente proyecto, se ha establecido la nueva contraseña como:

- **New Password:** adminUPS

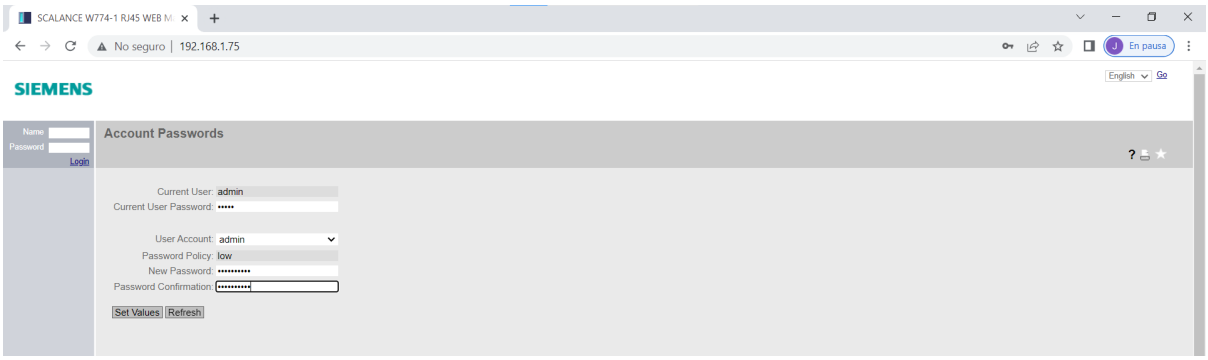


Figura 3.74: WBM, Cambio de contraseña.

Una vez realizada la modificación de la contraseña, al tratarse de la primera vez que se accede a la configuración del módulo SCALANCE W774, se habilitará el **"Basic Wizard" o "Asistente Básico de Configuraciones"**, lo cual permite configurar el equipo de forma rápida y sencilla.

El "Wizard Básico de Configuraciones"facilita la puesta en marcha del módulo SCALANCE W774, asegurando que las configuraciones iniciales sean realizadas adecuadamente y de manera intuitiva. Con este asistente, se podrán definir aspectos fundamentales como la dirección IP, máscara de subred, puerta de enlace y otras configuraciones básicas para establecer la comunicación del módulo en la red.

En la pestaña **"System Settings" (Ajustes del Sistema)** se encuentran las opciones para seleccionar el modo de operación del módulo SCALANCE W774. Este dispositivo puede trabajar en dos modos distintos: "AP" (Access Point) o "Cliente".

Para configurar el módulo como AP, se debe seleccionar la opción **"AP"** en el apartado **"Device Mode"** y luego se presiona el botón **"Next"** para continuar con la configuración (véase la figura 3.75).

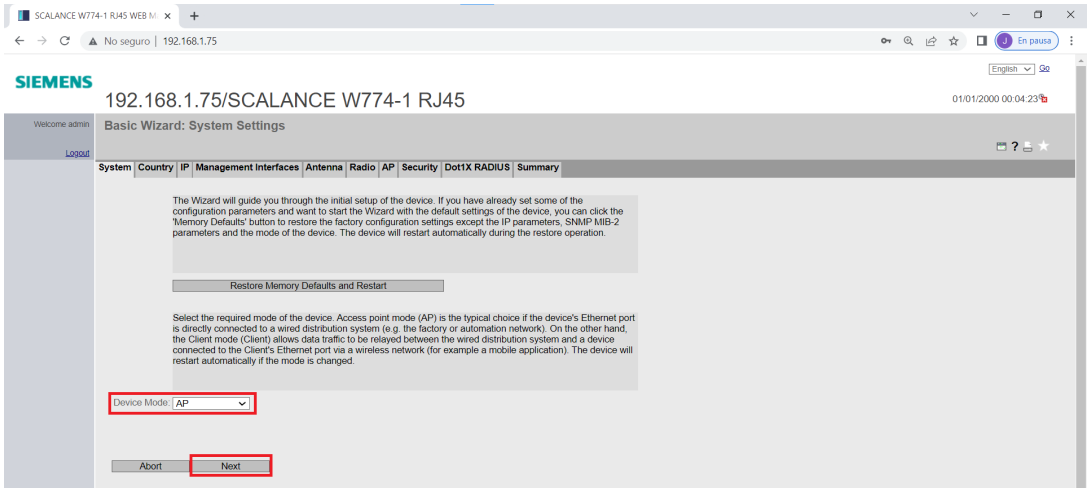


Figura 3.75: WBM, Ajustes del Sistema.

Posteriormente, en la pestaña **"Country Settings" (Ajustes del País)** se configura el país y el nombre del sistema. En el apartado **"Country Code" (Código del País)** se busca y selecciona la opción **"Ecuador"**, mientras que en la sección **"System Name" (Nombre del Sistema)** se agrega un nombre de identificación para el AP. En este caso se ha colocado el nombre **"Tesis_HwangIniguez"** (véase la figura 3.76).

Figura 3.76: WBM, Ajustes del País.

En la pestaña **"IP Address Settings" (Ajustes de la dirección IP)** (véase la figura 3.77), se realiza la configuración de la dirección IP y la máscara de subred del módulo SCALANCE W774. Estos valores deben coincidir con los que fueron previamente definidos en la Tabla 3.6 y asignados mediante el software SINEC IP durante la configuración inicial.

Figura 3.77: WBM, Ajustes de la dirección IP.

La pestaña de **"Management Interfaces" (Interfaces de Gestión)** permite habilitar las distintas interfaces desde donde es posible realizar la configuración y administración del dispositivo de manera remota y desde diferentes plataformas.

Para los objetivos del proyecto, se configura esta sección como se indica en la Figura 3.78.

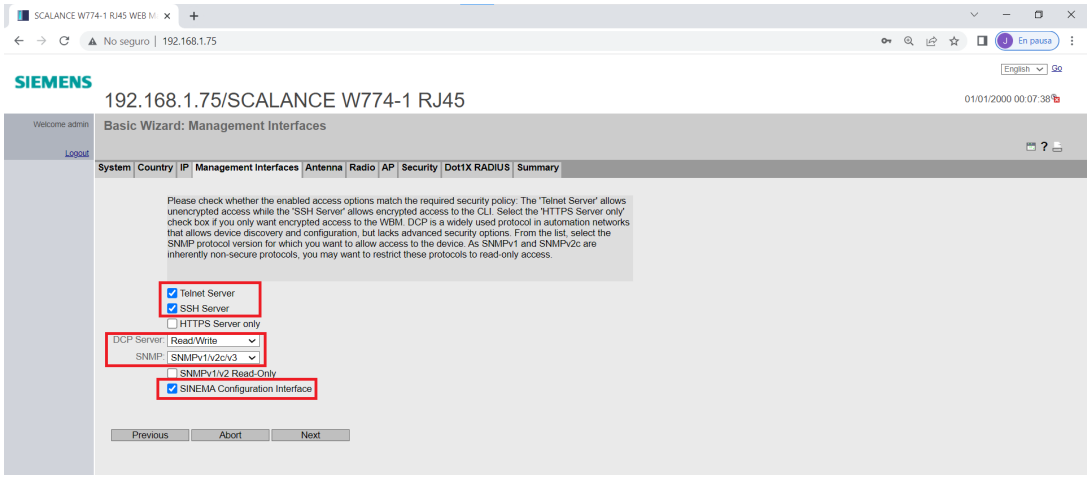


Figura 3.78: WBM, Interfaces de Gestión.

La siguiente pestaña **"Antenna Settings" (Ajustes de Antena)** permite seleccionar las antenas que se encuentran conectadas al módulo a través de sus dos puertos R-SMA. Por ende, se debe buscar y seleccionar para cada uno de los conectores disponibles la antena **Omni-Direct-Mount: ANT795-4MA**, cuyas características se vieron en la sección 1.1.4. Este procedimiento se ilustra en la Figura 3.79.

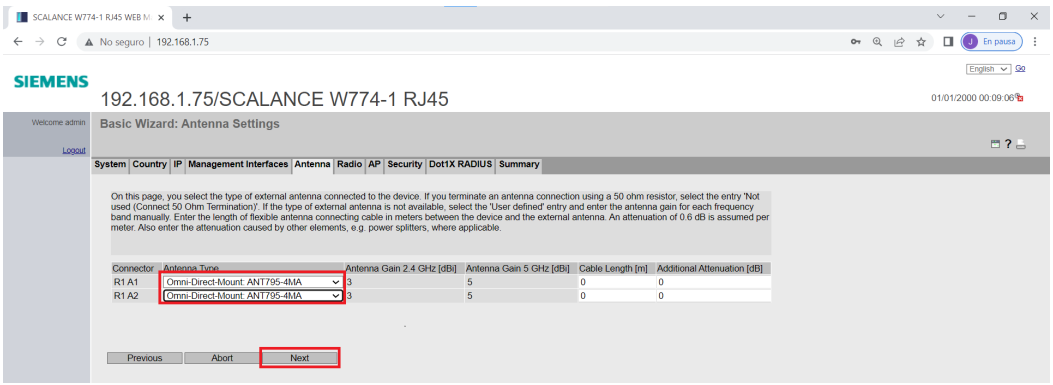


Figura 3.79: WBM, Ajustes de Antena.

En la pestaña de **"Radio Settings" (Ajustes de Radio)** se lleva a cabo la configuración de la red WLAN. Después de habilitar la red marcando la casilla "Enabled", se procede a definir la banda de frecuencia para la transmisión de la señal, el estándar de comunicación y la potencia de transmisión "TX Power", como se visualiza en la figura 3.80.

En la sección de **"Frequency Band" (Banda de Frecuencia)**, se elige la banda en la que operará la red inalámbrica, en este caso se trabaja con una banda de **2.4 GHz**.

En el apartado de **"Standard" (Estándar)**, se selecciona el protocolo de comunicación inalámbrica **802.11b/g/n**.

El parámetro **"TX Power" (Potencia de Transmisión)** hace referencia a la potencia con la que el dispositivo transmitirá la señal inalámbrica, lo cual afecta el alcance y la calidad de la conexión inalámbrica. En el proyecto se maneja una potencia de transmisión de **20 dBm**.

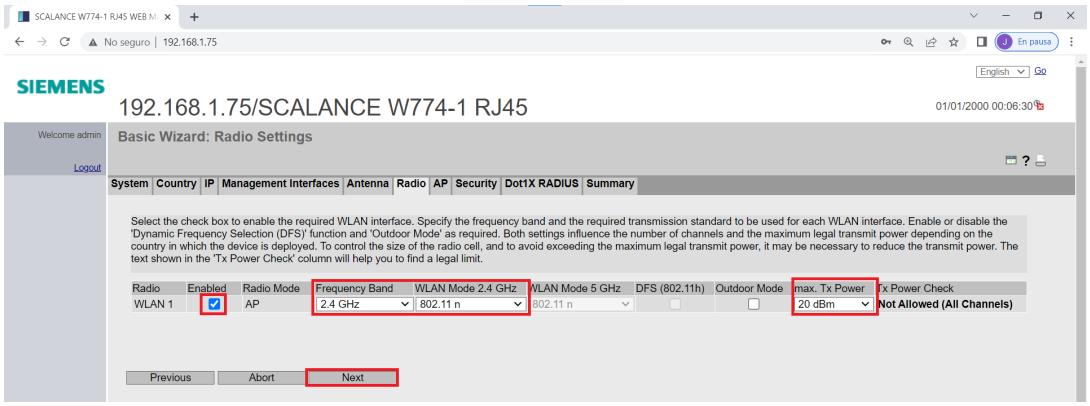


Figura 3.80: WBM, Ajustes de Radio.

La pestaña de **"Access Point Settings" (Ajustes del Punto de Acceso)** permite establecer el canal de transmisión de la red WLAN y asignar un nombre SSID a dicha red, como se muestra en la Figura 3.81.

La sección **"Channel" (Canal)** permite seleccionar el canal de transmisión para la red WLAN. Al configurarlo en modo **"Auto" (Automático)**, el dispositivo negociará la transmisión empleando el canal que tenga menor tráfico de datos, lo que garantiza una comunicación óptima y adecuada. Mientras que **"SSID" (Service Set Identifier)**, permite definir un nombre único para la red WLAN. Este nombre es el que aparecerá en la lista de redes inalámbricas disponibles cuando otros dispositivos intenten conectarse a la red. Para el proyecto se asigna el nombre **"Tesis_HwangIniguez"**.

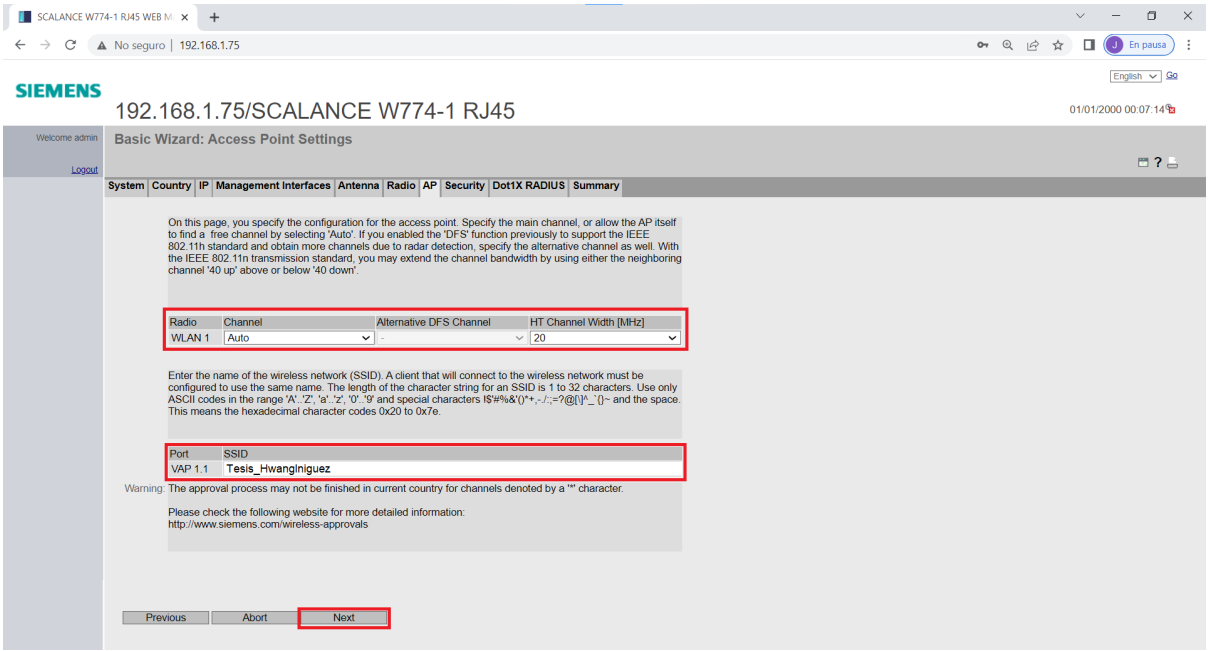


Figura 3.81: WBM, Ajustes del Punto de Acceso.

La siguiente pestaña **"Security Settings" (Ajustes de Seguridad)** permite configurar la seguridad que se aplicará al módulo para las conexiones inalámbricas. En la Figura 3.82 se muestran las diferentes opciones de seguridad que ofrece el módulo.

La primera opción es **"Open System"**, la cual no proporciona autenticación ni encriptación, permitiendo el acceso de cualquier dispositivo a la red WLAN sin restricciones. La segunda opción es **"WPA2 (RADIUS)"**, que emplea un servidor RADIUS para la autenticación de los dispositivos que deseen conectarse a la red

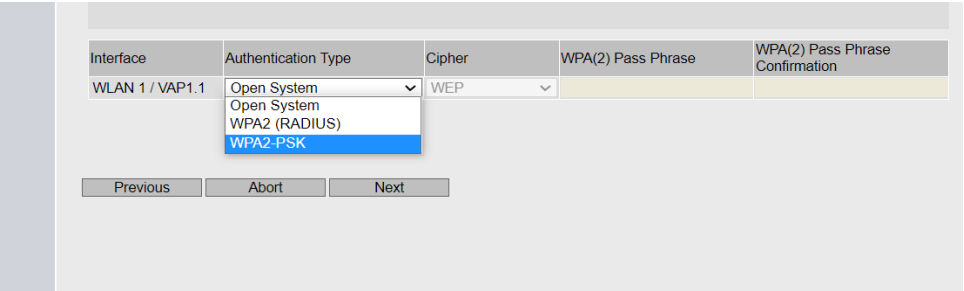


Figura 3.82: WBM, Opciones de seguridad.

WLAN, ofreciendo un mayor nivel de seguridad, pero requiere infraestructura de red adicional. La tercera opción es **"WPA2-PSK"**, que utiliza una clave para la autenticación de los dispositivos, lo que requiere que los dispositivos ingresen esta clave antes de poder acceder a la red WLAN.

Para el proyecto en cuestión, se ha decidido emplear la opción de seguridad **'WPA2-PSK'**, ya que esta alternativa proporciona una capa adicional de seguridad en el sistema sin requerir la implementación de hardware adicional, garantizando que solo los usuarios que tengan conocimiento de la clave de seguridad puedan acceder a la red WLAN, lo que ayuda a proteger la integridad de la comunicación inalámbrica de manera efectiva y sin complicaciones adicionales en la infraestructura.

En la Figura 3.83 se visualiza la selección de **"Authentication Type" (Tipo de Autenticación)**, además de definir la **"WPA(2) Pass Phrase" (Contraseña WPA(2))**

- **WPA(2) Pass Phrase:** Hwang_Iniguez

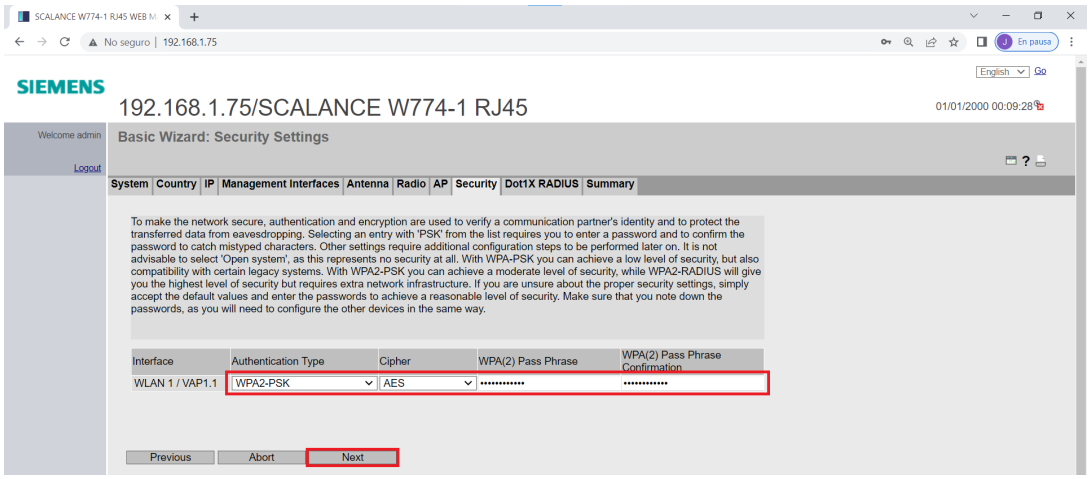


Figura 3.83: WBM, Ajustes de seguridad.

La última pestaña de configuración es **"Summary of Settings" (Resumen de Ajustes)**, mostrada en la Figura 3.84. En esta sección, se presenta un resumen de todas las configuraciones realizadas previamente en las distintas pestañas. Una vez confirmados los ajustes y comprobados los valores establecidos, se procede a guardar los cambios presionando el botón **"Set Values"**. De esta manera, se aplicarán las configuraciones en el módulo SCALANCE W774, asegurando que la red WLAN esté correctamente establecida y que todas las opciones de seguridad y ajustes necesarios estén activos para el funcionamiento adecuado del sistema de control.

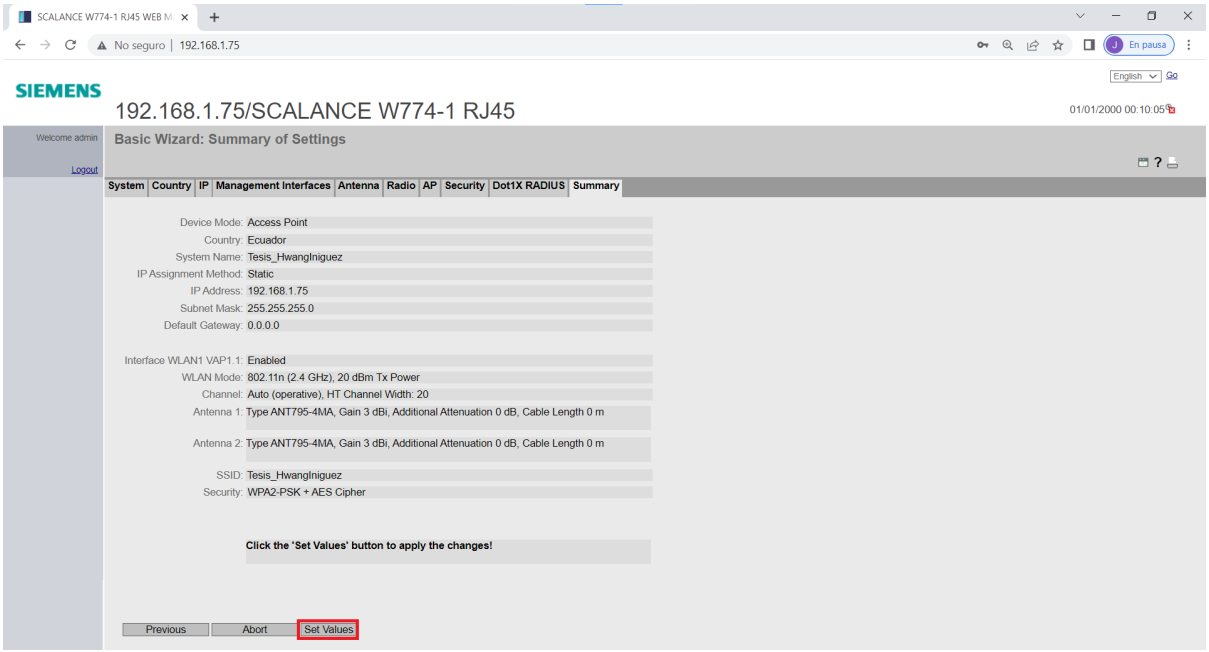


Figura 3.84: WBM, Resumen de Ajustes.

Capítulo 4

Diseño e implementación del sistema HMI

En este capítulo se presenta el diseño e implementación del sistema HMI (Interfaz Hombre-Máquina) basado en Python. Se describe la estructura del HMI y las funcionalidades implementadas, brindando una visión general de su operación y cómo interactúa con el resto del sistema.

Finalmente, se llevan a cabo las pruebas de funcionamiento del sistema en su conjunto, verificando el correcto intercambio de datos y el cumplimiento de los objetivos planteados en el proyecto. Se documentan los resultados obtenidos y se realizan las evaluaciones correspondientes para asegurar la operatividad y eficiencia del sistema implementado.

4.1. Diseño y funcionamiento del sistema HMI

El desarrollo de un sistema HMI (Interfaz Humano-Máquina) basado en Python requiere la integración de múltiples herramientas que permitan integrar la parte gráfica con las funciones programadas en dicho lenguaje.

En este contexto, se optó por utilizar la aplicación Qt Designer, la cual es compatible con diversos lenguajes de programación como C, C++, Python, entre otros. Qt Designer, como se explicó previamente en el capítulo correspondiente, necesita un framework adecuado para su correcto funcionamiento. A través de este framework,

se realiza la traducción de los elementos gráficos, propiedades y posiciones definidas en el programa, convirtiéndolos en un lenguaje que puede ser compilado por Python.

Para este proyecto específico, se seleccionó el framework PyQt5 debido a su alta compatibilidad con los elementos y librerías utilizadas en el desarrollo. Así pues, el primer paso consiste en instalar este complemento en Python, proceso que se simplifica mediante el uso del instalador pip. Es fundamental asegurarse de instalar tanto PyQt5 como PyQt5-tools para un funcionamiento adecuado.

Una vez completada la instalación, se procede a implementar la interfaz gráfica en Qt Designer, haciendo uso de una variedad de elementos, tales como botones (QPushButton) para establecer enlaces con las funciones, etiquetas (QLabels) para mostrar información relevante, contenedores (QFrame) para una organización eficiente de los elementos, listas (QListView) para la visualización de errores, páginas (QStackedWidget) que permitirán cambiar la información en función del menú seleccionado y finalmente, QVBoxLayout para organizar los objetos de manera ordenada, que para este caso, son usados para añadir elementos de librerías adicionales o códigos de programación que no están presentes en el panel de elementos de Qt por defecto.

Con los elementos colocados en su lugar, se modifican las propiedades de cada uno para establecer tamaños mínimos, máximos, así como también se modifican sus styleSheet, que a grandes rasgos, permite la personalización y agrega animaciones o eventos a los diferentes elementos.

En el contexto del desarrollo de interfaces gráficas, es importante destacar el papel significativo que juega el conjunto de propiedades utilizadas en combinación con Layouts y frames para organizar los elementos. Estos elementos ejercen una influencia positiva en el aspecto general de las interfaces y, al mismo tiempo, contribuyen a su carácter intuitivo y eficaz. Los resultados obtenidos de esta estrategia de diseño se pueden apreciar en las figuras 4.2, 4.3 y 4.4.

Después de completar el diseño de la interfaz gráfica, es crucial realizar la traducción del código generado en Qt Designer a Python. Para este propósito, se emplea el complemento Pyuics, el cual se instala mediante el comando pip. Pyuics es capaz de tomar el archivo .ui generado por Qt Designer y el archivo .qrc que contiene

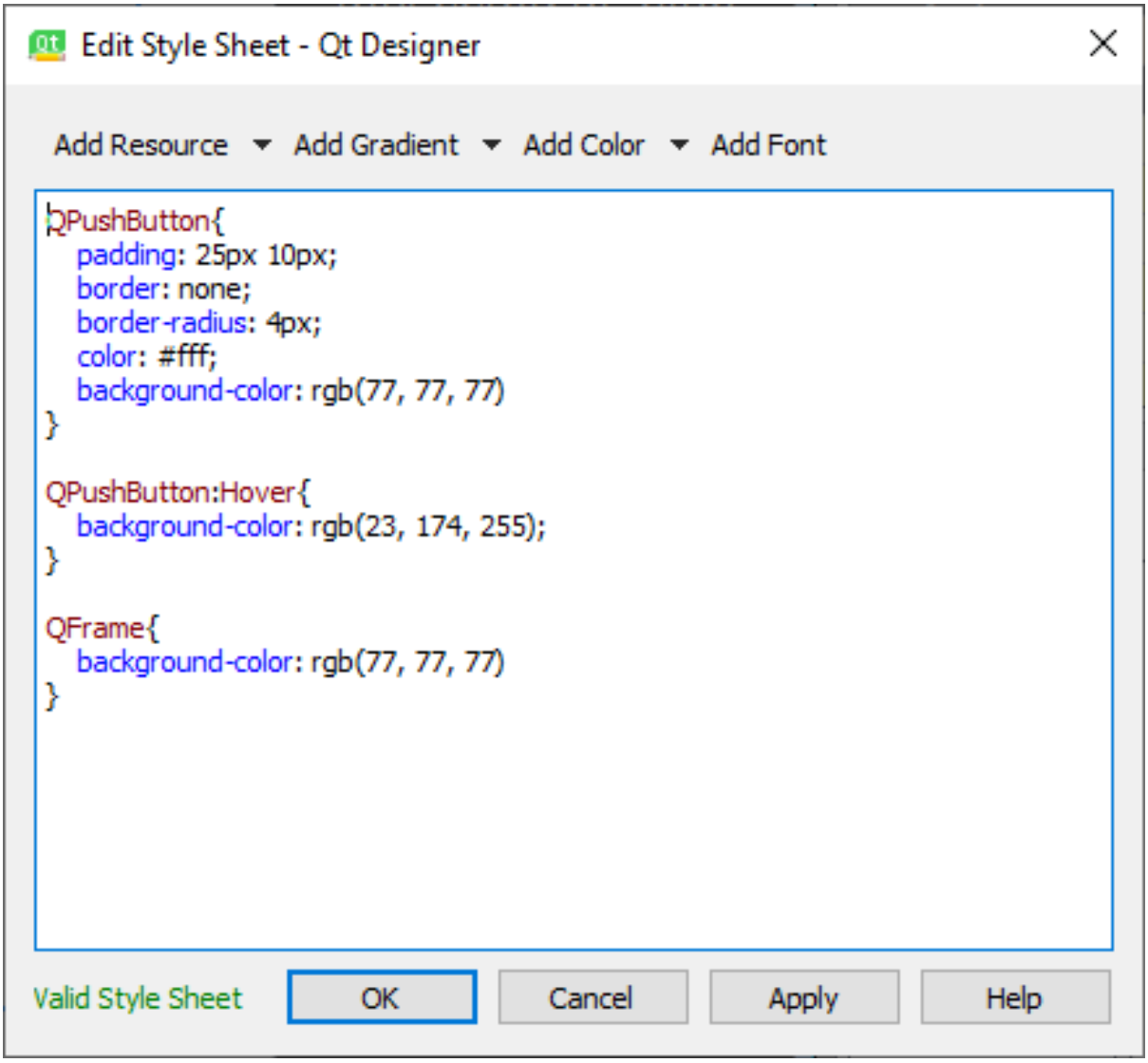


Figura 4.1: Modificación del styleSheet

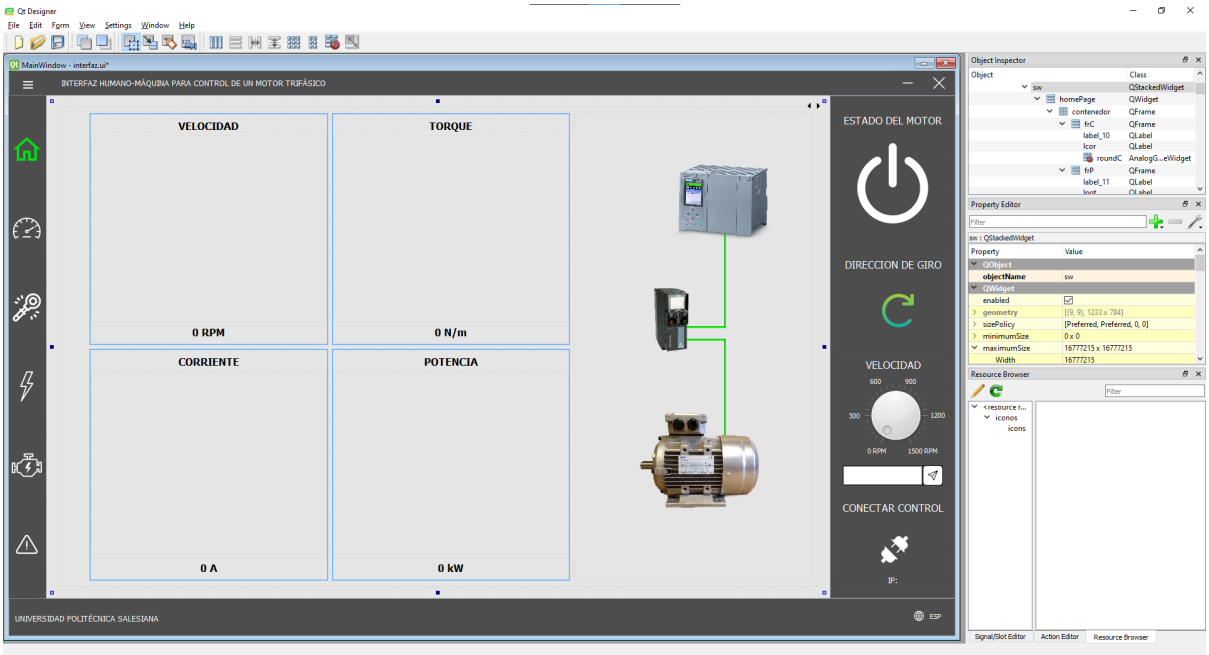


Figura 4.2: Interfaz gráfica diseñada en la página 1 correspondiente a la página de inicio

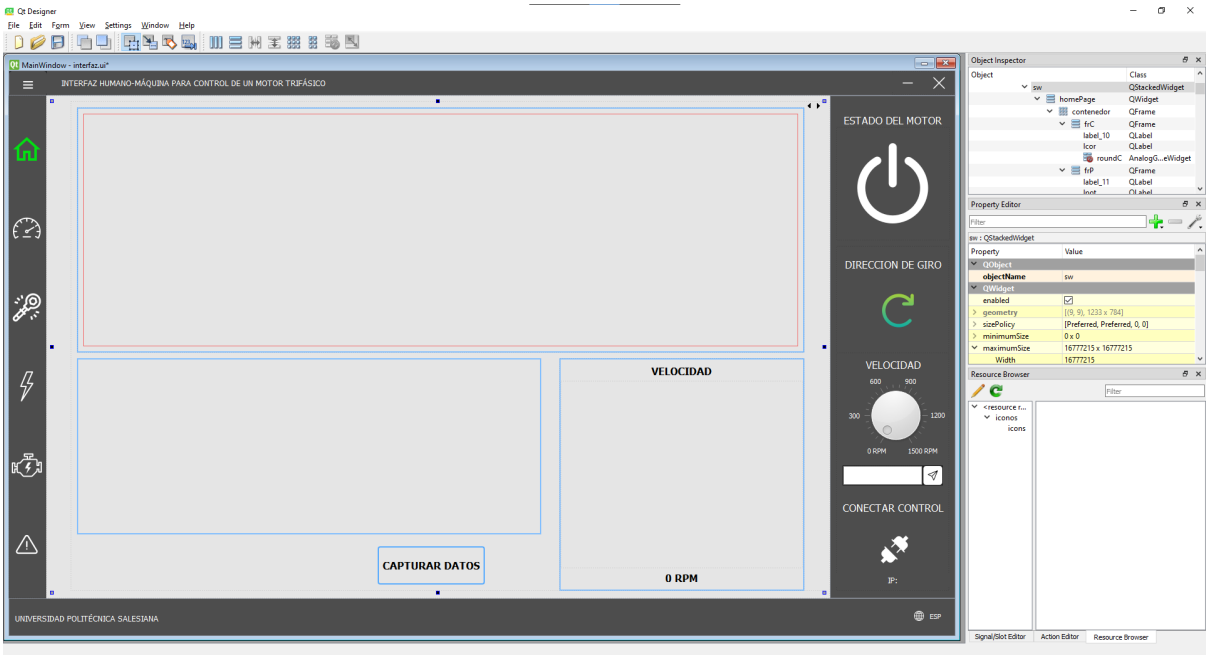


Figura 4.3: Interfaz gráfica diseñada en la página 2 correspondiente a visualización de parámetros específicos

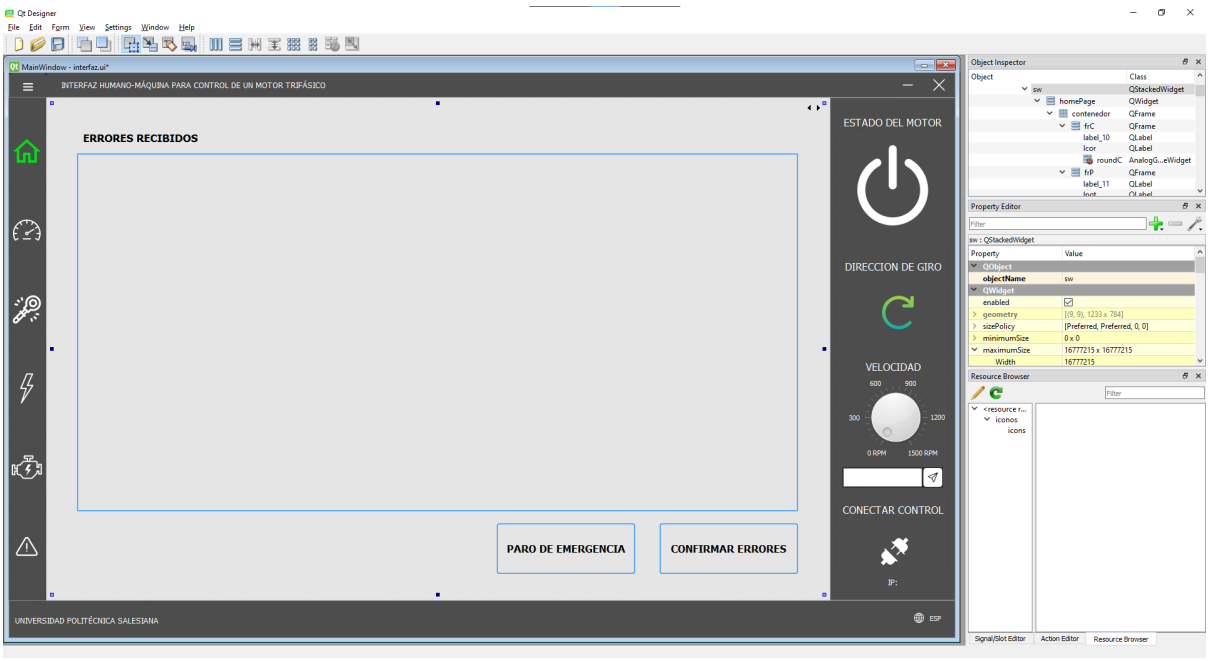


Figura 4.4: Interfaz gráfica diseñada en la página 3 correspondiente a los errores

los recursos e imágenes utilizadas en el proyecto, y los fusiona en un solo programa escrito en lenguaje Python. Esta fusión se realiza con el propósito de permitir la modificación del documento resultante o su importación en el código principal del proyecto para su posterior utilización.

Cabe recalcar que las actualizaciones que se realicen en la interfaz, tanto en propiedades, posición de elementos, o cualquier otro cambio, se puede ver reflejado en el archivo .ui, mas no en el código de python, por lo que es necesario realizar la traducción con Pyiucs siempre que se realice una modificación en Qt.

Una vez, completada la fase de diseño de la interfaz se pasa a la implementación en sí del código que permita el correcto funcionamiento de la misma, por lo que es necesario en este caso crear un nuevo documento en Python y cargar las librerías necesarias, así como también el archivo traducido de la interfaz. En esta etapa de implementación se escribe el código para la conexión con el PLC, la lectura y escritura de datos en el mismo, la implementación de la librería AnalogGaugeWidget, la creación e integración del Plot con datos obtenidos del PLC, la integración de los botones con sus respectivas funciones, modificación de propiedades de elementos para el menú plegable y la actualización visual de botones y labels, y, finalmente, la visualización de los errores en la lista, incluyendo un apagado del motor, forzando

a la página de errores y soltando una alarma sonora que indique fallos para salvaguardar la integridad del motor. A continuación se desglosa a grandes rasgos las funcionalidades de los bloques de programación.

El primer paso consiste en importar las librerías necesarias para el desarrollo del proyecto. Entre las principales librerías utilizadas se encuentran PyQt5, snap7 para la lectura y escritura de datos del PLC, las librerías enfocadas en gráficas del plot, AnalogGaugeWidget, pygame para las alertas sonoras, y, las librerías de threading ya que se emplea para evitar que la interfaz se congele durante la lectura y actualización de los datos. La utilización de hilos permite ejecutar estas tareas de manera simultánea, mejorando significativamente el rendimiento y la experiencia del usuario.

Posterior a ello es necesario definir variables globales ya que como el programa esta dividido en funciones, se usan estas variables como banderas para mantener información y emplearla en la toma de decisiones según los diferentes casos que se puedan presentar.

Empezando ya con el código, el programa en sí depende de los datos que se obtienen del PLC, por lo que, lo primero que se realiza para que el programa funcione es conectar el ordenador con el PLC mediante la librería de snap7:

Código de conexión con el PLC mediante Snap7

```
PLC_Ip='192.168.1.1'
PLC_Rack=0
PLC_Slot=1
plc = snap7.client.Client()
plc.connect(PLC_Ip,PLC_Rack,PLC_Slot)
```

Ya establecida la conexión, se puede crear las funciones globales que se utilizan para la lectura y escritura de datos del PLC. Si bien la librería de Snap7 cuenta con funciones propias para ello, usamos las mismas dentro de un lock, cuya función es simplemente esperar que el hilo este disponible para realizar la tarea dentro del

código:

Código para la lectura y escritura de datos en el PLC con bloqueo de hilos

```
def plc_db_read(db_number, start_byte, num_bytes):
    """Función para leer datos desde el PLC"""
    with plc_lock:
        data = plc.db_read(db_number, start_byte, num_bytes)
        return list(data)

def plc_db_write(db_number, start_byte, data):
    """Función para escribir datos en el PLC"""
    with plc_lock:
        plc.db_write(db_number, start_byte, data)
```

Dado que se coloca un lock en las funciones, es necesario entonces definir un objeto que pueda ser inicializado dentro de un hilo, por lo que se crea un QObject llamado dataReader:

Código definición del objeto clase DataReader

```
class DataReader(QObject):
    data_read = pyqtSignal(float, float, float, float, list,
    bool, bool, bool, bool)

    def __init__(self):
        super(DataReader, self).__init__()
        self._running = threading.Event()
```

Este objeto dataReader cumple con la función de leer datos del PLC mediante las funciones creadas anteriormente, por lo que debe permanecer en un bucle infinito mientras la aplicación este funcionando, así se garantiza una lectura constante de

valores desde el PLC. Este objeto primero lee los datos desde el PLC y lo guarda como un *bytearray*, luego traduce dichos valores a los formatos requeridos, ya sean de tipo bool o real y junto con la traducción prepara los datos para que puedan ser usados en el código. Con los datos ya escalados para las gráficas, emite dichos valores de modo que puedan ser leídos desde el main del código. En sí, la funcionalidad del código se muestra a continuación:

Código con la función del Objeto DataReader

```
def run(self):
    global FLAG
    FLAG=True
    self._running.set()
    while self._running.is_set():
        reading_v = plc_db_read(1, 0, 4)
        reading_c = plc_db_read(1, 4, 4)
        reading_t = plc_db_read(1, 8, 4)
        reading_p = plc_db_read(1, 12, 4)
        reading_e = plc_db_read(1, 22, 2)
        reading_bools = plc_db_read(1, 20, 1)
        velR1 = snap7.util.get_real(reading_v, 0)
        velR = abs(round(velR1, 3))
        corR1 = snap7.util.get_real(reading_c, 0)
        corR = 100*abs(round(corR1, 3))
        torR1 = snap7.util.get_real(reading_t, 0)
        torR = 1000*abs(round(torR1, 3))
        potR1 = snap7.util.get_real(reading_p, 0)
        potR = 1000*abs(round(potR1, 3))
        errR1 = snap7.util.get_word(reading_e, 0)
        errR = []
        acER = snap7.util.get_bool(reading_bools, 0, 3)
        if (acER==True):
            for i in range(16):
                bit_value = (errR1 >> i) & 1
                errR.append(bit_value)
        onR = snap7.util.get_bool(reading_bools, 0, 0)
        giroR = snap7.util.get_bool(reading_bools, 0, 1)
        conR = snap7.util.get_bool(reading_bools, 0, 2)
        self.data_read.emit(velR, corR, torR, potR, errR,
        acER, onR, giroR, conR)
        time.sleep(0.5)
```

Posteriormente, con un objeto creado, se necesita crear igualmente un hilo que contenga el objeto y permita que este sea iniciado desde el código principal, por lo que es necesario crear una clase QThread:

Código del hilo que utiliza el objeto DataReader

```
class DataReaderThread(QThread):
    def __init__(self):
        super(DataReaderThread, self).__init__()
        self.data_reader = DataReader()

    def run(self):
        self.data_reader.run()

    def stop(self):
        self.data_reader.stop()
```

Siguiendo en el código se encuentra la clase MplCanvas, la cuál es usada para la creación de un objeto tipo Canvas que permite a su vez la creación de una gráfica plot para mostrar las curvas de velocidad, torque, potencia o corriente del motor:

Código de implementación de clase MplCanvas,

```
class MplCanvas(FigureCanvas):
    def __init__(self, parent=None, width=5, height=4, dpi=100):
        fig = Figure(figsize=(width, height), dpi=dpi)
        self.ax = fig.add_subplot(111)
        super(MplCanvas, self).__init__(fig)
```

Una vez completadas las funciones, banderas y clases preliminares, se procede con la inicialización de la ventana de la interfaz gráfica, así como los parámetros iniciales de la misma:

Código de inicialización de la ventana de la interfaz gráfica

```
class MainWindow(QtWidgets.QMainWindow):


    def __init__(self, *args, **kwargs):
        super(MainWindow, self).__init__(*args, **kwargs)
        global LOCAL


        self.setWindowFlag(Qt.FramelessWindowHint)
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)
        self.setWindowTitle("HMI Motor")
        self.setWindowIcon(QtGui.QIcon(":/iconos/icons/iconoApp.png"))
```

Una vez iniciada la ventana, se procede a realizar la configuración inicial y obtener los parámetros iniciales del proyecto. Dado que el objetivo del proyecto es lograr un sistema multiestación, es fundamental establecer una conexión previa. Además, para evitar conflictos con las direcciones IP, se recomienda detectar automáticamente la dirección IP del ordenador y almacenar este valor para futuras referencias:

Código de obtención de dirección IP de la estación

```
if platform.system() == 'Windows':
    LOCAL = subprocess.getoutput("""for /f "tokens=2 delims=[]" %a
    in ('ping -n 1 -4 "%computername%"') do @echo %a""")
    else:
        LOCAL = subprocess.getoutput("ifconfig | grep 'inet '
        | grep -Fv 127.0.0.1 | awk '{print $2}'")

    l_ip='IP: '+str(LOCAL)
    self.ui.lip.setText(l_ip)
```

Dentro de los parámetros iniciales, se deben configurar la lectura de datos, el tratamiento de la información, los eventos y el enlace entre los botones y las funciones correspondientes. A continuación, se pueden apreciar los enlaces establecidos entre algunos botones y sus respectivas funciones.

Código de enlace entre botones y sus respectivas funciones.

```
self.ui.bclose.clicked.connect(lambda: self.close())
self.ui.bmenu.clicked.connect(lambda: self.slideLeftMenu())
self.ui.onoff.clicked.connect(lambda: self.setOnOff())
self.ui.enviar.clicked.connect(lambda: self.setVelTxt())
self.ui.chDir.clicked.connect(lambda: self.setGiro())
```

Asimismo, se procede a la configuración preliminar de diversos elementos, destacando especialmente el dial, el cual se configura para imprimir su valor dentro de un cuadro de texto editable. De esta manera, se cuentan con dos métodos para controlar la consigna de velocidad que será enviada al motor. En el siguiente cuadro de texto se muestra cómo se configuran los valores admitidos por el dial, así como su enlace con el cuadro de texto y el botón para el envío de la velocidad:

Código de definición de consigna de velocidad

```
self.ui.dialV.setMinimum(0)
self.ui.dialV.setMaximum(1440)
self.ui.dialV.setValue(0)
texto = float(self.ui.dialV.value())
self.ui.txtV.setText(str(texto))
self.ui.dialV.valueChanged.connect(self.setVelDial)
self.ui.txtV.returnPressed.connect(self.setVelTxt)
```

Posteriormente, se configuran parámetros iniciales para la presentación de datos por el AnalogGaugeWidget, por lo que se configura la parte gráfica, los colores, configuración del manejo de información tales como las unidades, el formato de texto, los valores máximos y mínimos:

Código de configuraciones del AnalogGaugeWidget

```
self.ui.roundV.units = "RPM"
self.ui.roundC.units = "A"
self.ui.roundT.units = "Nm"
self.ui.roundP.units = "KW"

self.ui.roundC.minValue = 0
self.ui.roundV.minValue = 0
self.ui.roundT.minValue = 0
self.ui.roundP.minValue = 0

self.ui.roundV.maxValue = 1500
self.ui.roundC.maxValue = 220
self.ui.roundT.maxValue = 50
self.ui.roundP.maxValue = 50
```

Ya con todos los elementos pre-configurados para su inicio junto con la interfaz, se empieza a correr las funciones y los hilos para la captura de datos por parte del PLC y del objeto de clase MplCanvas, utilizado para añadir la gráfica con las curvas del comportamiento del motor:

Código de ejecución de funciones e hilos del sistema

```
self.data_reader_thread = DataReaderThread()
self.data_reader_thread.data_reader.data_read.connect
    (self.update_progress)
self.data_reader_thread.finished.connect
    (self.data_reader_finished)
self.data_reader_thread.start()

self.canvas = MplCanvas(self.ui.velPage)
self.ui.graphV.addWidget(self.canvas)

self.xdata = []
self.ydataT = []
self.ydataC = []
self.ydataP = []
self.ydataV = []
```

Finalmente, antes de empezar la declaración de las funciones con las cuales los botones se enlazan, se agrega la configuración para la visualización de la tabla que se aprecia en la página 2 de la interfaz (véase la figura 4.3), colocando la información estática, tales como nombres de las columnas y los valores que se agregan, para así con la función de actualización agregar los valores instantáneos leídos del PLC:

Código de configuración de visualización de parámetros de la tabla

```
self.ui.tabV.setRowCount(4)
self.ui.tabV.setColumnCount(3)
self.ui.tabV.setEditTriggers(QAbstractItemView.NoEditTriggers)
self.ui.tabV.horizontalHeader().setDefaultSectionSize(166)
self.ui.tabV.horizontalHeader().setStretchLastSection(True)
self.ui.tabV.verticalHeader().setDefaultSectionSize(52)
self.ui.tabV.verticalHeader().setStretchLastSection(True)

nombreCol = ('Parametro', 'Valor', 'Unidad')
self.ui.tabV.setHorizontalHeaderLabels(nombreCol)

self.ui.tabV.setItem(0,0,QTableWidgetItem("Velocidad"))
self.ui.tabV.setItem(1,0,QTableWidgetItem("Torque"))
self.ui.tabV.setItem(2,0,QTableWidgetItem("Corriente"))
self.ui.tabV.setItem(3,0,QTableWidgetItem("Potencia"))

self.ui.tabV.setItem(0,2,QTableWidgetItem("RPM"))
self.ui.tabV.setItem(1,2,QTableWidgetItem("N/m"))
self.ui.tabV.setItem(2,2,QTableWidgetItem("A"))
self.ui.tabV.setItem(3,2,QTableWidgetItem("kW"))
```

La funciones de escritura en sí son similares, solo varía el tipo de dato que se escribe en el PLC según las herramientas vistas en el capítulo anterior. Resumiendo lo anteriormente visto, se lee un *bytearray* del dato que se desea sobre escribir, se usa un comando para cambiar el dato leído y se escribe en la misma posición de memoria de donde se leyó:

Código de la función Paro de Emergencia

```
def paroEmer(self):
    global B_ON_OFF
    global PV, PC, PT, PP
    B_ON_OFF=0
    reading_bool = plc_db_read(1, 20, 1)
    set_bool(reading_bool,0,0,0)
    plc_db_write(1,20,bytearray(reading_bool))
```

Se añadió también la función de cambiar el idioma entre español e inglés, por lo que se agrega la funcionalidad con banderas que se activan al momento para reescribir la información de los labels y botones. Se guardan en variables globales las palabras tanto en inglés como español, y según la bandera activa, el idioma de los elementos cambia:

Código de selección de idioma en la interfaz HMI

```
def setIdioma(self):
    global VEL_ESP, COR_ESP, POT_ESP, UPS_ESP, EST_ESP
    global DIR_ESP, CON_ESP, TIT_ESP, CAP_ESP, ERR_ESP
    global CONF_ESP, PAR_ESP, CONT1
    global VEL_ENG, COR_ENG, POT_ENG, UPS_ENG, EST_ENG
    global DIR_ENG, CON_ENG, TIT_ENG, CAP_ENG, ERR_ENG
    global CONF_ENG, PAR_ENG
    CONT1=CONT1+1
    if CONT1==1:
        self.ui.vel.setText(VEL_ENG)
        self.ui.pot.setText(POT_ENG)
        self.ui.cor.setText(COR_ENG)
        self.ui.label_8.setText(VEL_ENG)
        self.ui.label_10.setText(COR_ENG)
        ...
```

Entre las funciones principales, la más importante es la función Update, la cual, permite actualizar los valores de la tabla, las gráficas, los analogGaugeWidget, labels, entre otros elementos, así como también activar las alarmas en caso de fallo, y todo a partir de los datos leídos en el hilo anteriormente comentado.

A continuación se puede observar la parte del código enfocado en la actualización de datos de la primera página, concretamente hablando de los labels con la velocidad, torque, potencia y corriente instantánea, así como también la actualización de los datos en la gráfica circular:

Código de actualización de datos de la primera ventana de interfaz HMI

```
def update_progress(self, velR, corR, torR, potR,
   errR, acER, onR, giroR, conR):
     global FLAG
     global B_ON_OFF, B_Giro
     global CONT1

     self.ui.tabV.setItem(0,1,QTableWidgetItem(str(velR)))
     self.ui.tabV.setItem(1,1,QTableWidgetItem(str(torR)))
     self.ui.tabV.setItem(2,1,QTableWidgetItem(str(corR)))
     self.ui.tabV.setItem(3,1,QTableWidgetItem(str(potR)))

     self.ui.roundV.updateValue(velR)
     self.ui.roundC.updateValue(corR)
     self.ui.roundT.updateValue(torR)
     self.ui.roundP.updateValue(potR)

     strV=str(velR)+' RPM'
     strC=str(corR/100)+' A'
     strT=str(torR/1000)+' N/m'
     strP=str(potR/1000)+' kW'

     self.ui.lvel.setText(strV)
     self.ui.lcor.setText(strC)
     self.ui.ltor.setText(strT)
     self.ui.lpot.setText(strP)
```

Dentro de la misma función de actualización, se encuentran parámetros de lectura que proyectan el estado del motor en la interfaz, de modo que, al ser una interfaz multiplataforma, los cambios realizados por un usuario se visualicen en el resto de interfaces:

Código de actualización de datos de primera ventana de interfaz HMI enfocada en la multiestación

```
if onR==True:
    self.ui.onoff.setStyleSheet("background-position: ...")
    self.ui.lmotor.setStyleSheet("border-image: ...")
    B_ON_OFF=1
else:
    self.ui.onoff.setStyleSheet("background-position: ...")
    self.ui.lmotor.setStyleSheet("border-image: ...")
    B_ON_OFF=0

if giroR==True:
    self.ui.chDir.setStyleSheet("background-position:  ...")
    B_Giro=1
else:
    self.ui.chDir.setStyleSheet("background-position: ...")
    B_Giro=0
```

En la parte 3 de la misma función, se agregan los valores instantáneos al final de un vector, el cual es utilizado para visualizar los últimos cien datos en el objeto canvas, así como también poder exportar con la función capturar los vectores a un documento creado en la misma carpeta.

Código de actualización de valores almacenados

```
self.xdata.append(len(self.xdata))
self.ydataV.append(velR)
self.ydataC.append(corR)
self.ydataT.append(torR)
self.ydataP.append(potR)
```

Con los vectores con la información ya creados y actualizándose en cada iteración, procedemos con la elaboración de la gráfica que será agregada en la página 2. Como solo se cuenta con una página para la visualización de las curvas, en el momento que se pulsa un botón de selección en el menú, se actualiza la gráfica para mostrar dicha curva, en lugar de estar graficando las 4 al mismo tiempo, ahorrando así recursos de memoria y evitando que el programa se ralentice:

Código de actualización de valores de la gráfica

```
if PV==True:
    self.canvas.ax.clear()
    self.canvas.ax.plot(self.xdata, self.ydataV)

    # Ajustar el rango de los ejes x e y
    max_x = max(self.xdata)
    min_x = max(0, max_x - 100)
    max_y = 1550
    min_y = 0
    self.canvas.ax.set_xlim(min_x, max_x)
    self.canvas.ax.set_ylim(min_y, max_y)
    self.ui.lvel2.setText(strV)
    self.ui.roundV2.updateValue(velR)
    self.ui.roundV2.maxValue = 1500
    if CONT1==0:
        self.canvas.ax.set_xlabel('Tiempo (s)')
        self.canvas.ax.set_ylabel('Velocidad (RPM)')
    elif CONT1==1:
        self.canvas.ax.set_xlabel('Time (s)')
        self.canvas.ax.set_ylabel('Speed (RPM)')
```

Otro de los parámetros que se debe actualizar a cada instante es el error, dado que un bit de error que avisa que hay un problema en el controlador del motor puede

cambiar en cualquier momento, por lo que es importante que siempre el programa este preguntando por el estado de los errores. Se usa una bandera para cuando exista algún error y entre los bits recibidos se tiene la indicación de cuál fue el error recibido:

Código de detección de errores

```
if (acER==True and FLAG==True):
    FLAG=False
    mixer.music.play()
    if errR[1]==1:
        self.model.insertRow(self.model.rowCount())
        index = self.model.index(self.model.rowCount() - 1)
        self.ui.err.setStyleSheet("background-position: ...")
        act=True
        if CONT1==0:
            self.model.setData(index, 'Error 1: ...')
        elif CONT1==1:
            self.model.setData(index, 'Error 1: ...')
        self.setErr(act)
```

Concluida la función de actualización de los valores, se necesita una función especial propia del programa, la cuál será usada en caso de que el programa se cierre. Esta característica es importante ya que al ser una aplicación multiestación en la cual solo un dispositivo puede modificar los datos a la vez, se tiene que generar una conexión con la IP, por lo que al cerrarse el programa se debe desconectar la estación del PLC. En otras palabras, esta función es importante porque permite el paso a otras estaciones cuando se cierre el programa, evitando que el PLC se bloquee y otros usuarios no puedan modificar los parámetros del motor:

Código para para finalizar la ejecución del programa y permitir el acceso a otra estación remota

```
    def closeEvent(self, event):
        self.data_reader_thread.stop()
        self.data_reader_thread.wait()
        reading_b = plc_db_read(1, 20, 1)
        reading_str = plc_db_read(1,24,255)
        conR = snap7.util.get_bool(reading_b, 0, 2)
        conS = snap7.util.get_string(reading_str, 0)
        print(LOCAL)
        print(conS)
        print(conR)
        if conR==True and conS==LOCAL:
            set_bool(reading_b,0,2,0)
            plc_db_write(1,20,bytearray(reading_b))
            set_string(reading_str, 0, '0.0.0.0')
            plc_db_write(1,24,bytearray(reading_str))
        super(MainWindow, self).closeEvent(event)
```

Como se trato anteriormente, se necesita de un permiso para modificar los datos, por lo que inicialmente los botones de inicio, cambio de giro, así como el dial están deshabilitados. Solo se habilitarán en caso de que no exista ningún equipo conectado con el PLC, es por esta razón que se necesita obtener la IP del equipo, ya que al momento de la conexión como se puede ver a continuación, se establece la IP en el PLC con el fin de reservar el derecho de modificación, disponible solo para un dispositivo a la vez:

Código para control de permiso y reserva de derecho de modificación

```
def conControl(self):
    reading_b = plc_db_read(1, 20, 1)
    reading_str = plc_db_read(1,24,255)
    conR = snap7.util.get_bool(reading_b, 0, 2)
    conS = snap7.util.get_string(reading_str, 0)

    if conR==False:
        global LOCAL
        reading_IP = plc_db_read(1, 24, 255)
        set_string(reading_IP,0,LOCAL)
        plc_db_write(1,24,bytearray(reading_IP))
        set_bool(reading_b, 0, 2, True)
        plc_db_write(1,20,bytearray(reading_b))
        self.ui.linkC.setStyleSheet("background-position: ...")
        self.ui.onoff.setEnabled(True)
        self.ui.chDir.setEnabled(True)
        self.ui.dialV.setEnabled(True)
        self.ui.enviar.setEnabled(True)
    if conR==True and conS==LOCAL:
        set_bool(reading_b,0,2,0)
        plc_db_write(1,20,bytearray(reading_b))
        set_string(reading_str, 0, '0.0.0.0')
        plc_db_write(1,24,bytearray(reading_str))
        self.ui.linkC.setStyleSheet("background-position: ...")
        self.ui.onoff.setEnabled(False)
        self.ui.chDir.setEnabled(False)
        self.ui.dialV.setEnabled(False)
        self.ui.enviar.setEnabled(False)
```

Para concluir la subsección, con las funciones de los botones definidas y el trabajo de actualización con los hilos realizada, se necesita iniciar la ventana creada,

para lo que se agrega las siguientes líneas de código:

Código de inicialización de interfaz

```
def main():
    app = QtWidgets.QApplication(sys.argv)
    main = MainWindow()
    main.show()
    sys.exit(app.exec_())

if __name__ == '__main__':
    main()
```

4.2. Pruebas de funcionamiento

Con los preparativos contemplados en el capítulo anterior, la configuración del variador SINAMICS G120, la configuración Profinet, el telegrama estandar 20 PXD y el módulo para comunicación inalámbrica SCALANCE W774, además del diseño e implementación de la interfaz HMI, se dispone a la integración de cada elemento antes mencionado para realizar las pruebas de funcionamiento.

En el código de la interfaz ya se estable la conexión con el PLC a través de la librería Snap7, sin embargo, para dicho programa funcione correctamente es necesario que el dispositivo se encuentre en la red generada por el dispositivo SCALANCE W774, y, cuente con una dirección IP en la misma subred que el PLC.

Con los dos parámetros en orden, se inicia el programa en Python tal y como se puede apreciar en la figura 4.5. Cabe recalcar que el dispositivo ya se encuentra enlazado con el PLC, sin embargo el control esta bloqueado hasta que se ingrese la IP al PLC en caso de que no haya ningún otro dispositivo controlando el motor.

Con el botón que se encuentra en la esquina inferior izquierda, se puede enlazar con el PLC. Abajo de dicho botón se puede observar también la dirección IP del dispositivo en el que la interfaz está funcionando. Si se establece la conexión, los

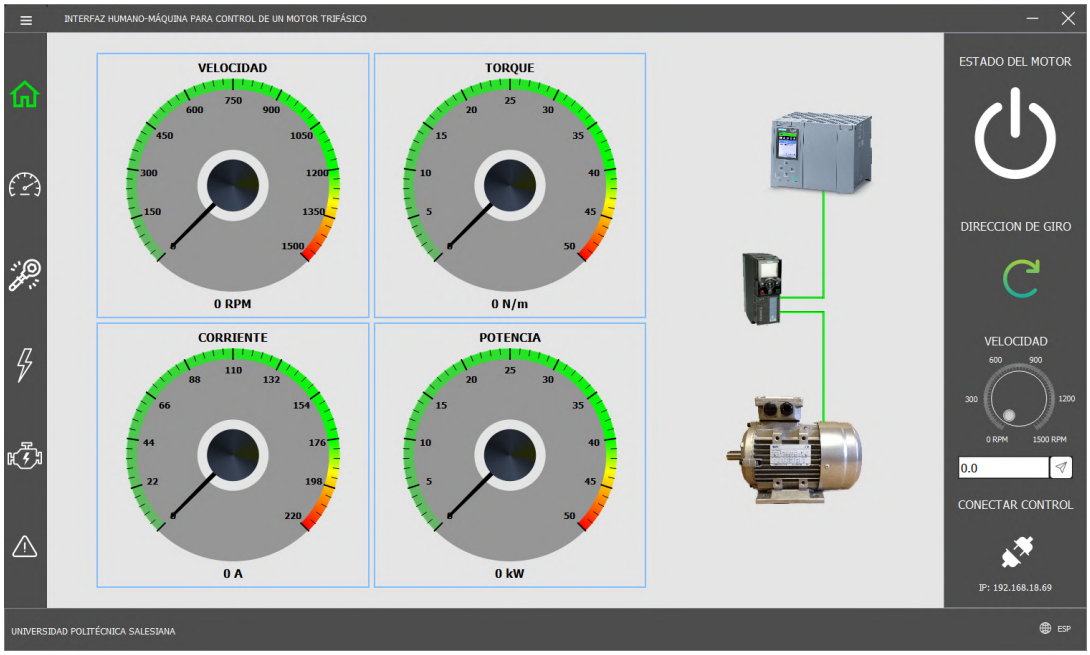


Figura 4.5: Inicio de la interfaz HMI

botones de control se habilitan y el icono de conexión del botón cambia como se puede observar en la figura 4.6

Antes de la conexión, los botones que no están bloqueados son los de monitoreo, es decir, los botones que llevan a la página 2 y 3, mostrando los errores y los parámetros. Esto se debe a que el propósito del sistema es que sea multiplataforma, por lo que, aunque no puedan controlar a la vez, si pueden monitorear. Otro botón que se encuentra funcionando es el de cambio de idioma, por lo que al presionarlo el idioma pasa de inglés a español o viceversa, sin necesidad de establecer conexión con el PLC.

A continuación, para comprobar que la conexión con el PLC se establece de forma correcta, se controlan los parámetros del motor, lo cuál debería verse reflejado en el HMI, ya que las variables del PLC son las que activan las animaciones y las gráficas del propio HMI. Primero se prueba el encendido del motor, lo cual, de resultar correcto es observable en el botón de encendido así como en las gráficas circulares de la primera página. Esto, junto con el cambio de idioma, se puede comprobar en la figura 4.7.

Para el resto de pruebas, se regresa al idioma español. En la siguiente prueba se utiliza el cuadro de texto para controlar la consigna de velocidad del motor, también

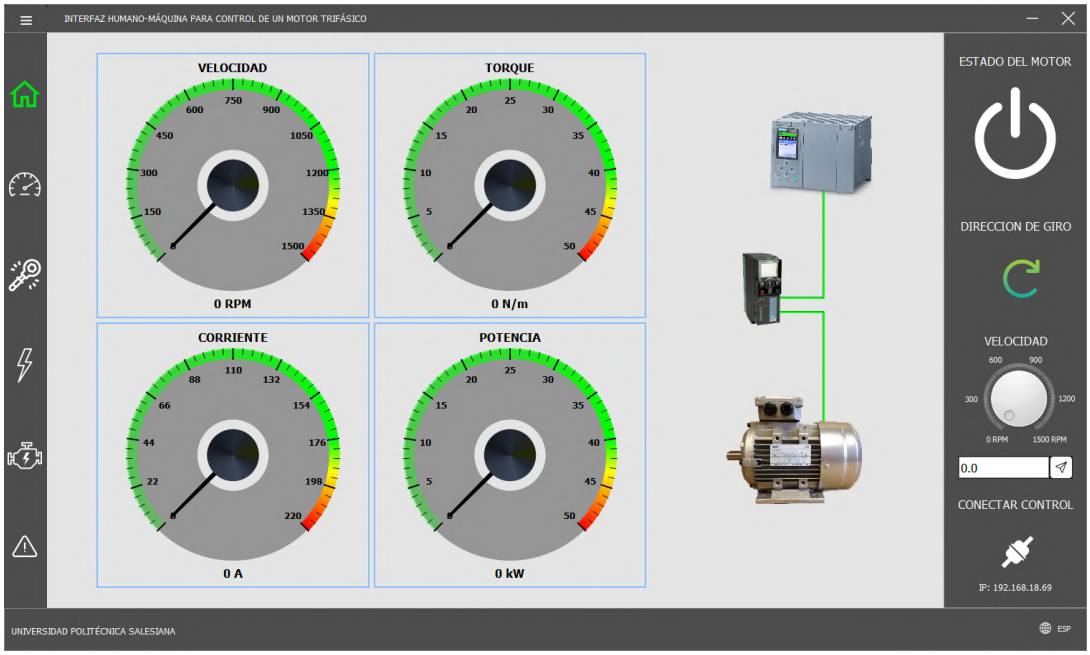


Figura 4.6: Interfaz HMI con conexión satisfactoria

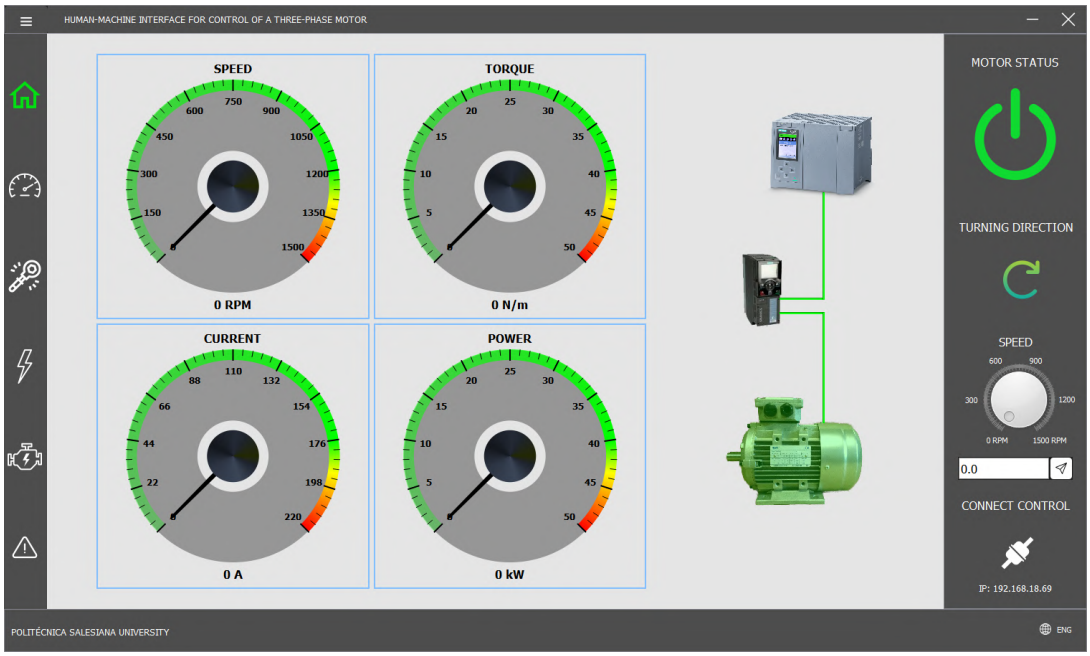


Figura 4.7: Interfaz HMI en inglés con el motor encendido

se usa el dial, que cumple la misma función. El motivo de la prueba es variar los parámetros del motor y ver reflejado dicho cambio no solo en las gráficas circulares de la página general, sino también en el plot de la segunda página. Desde la figura 4.8 hasta la figura 4.11 se pueden observar las curvas con los diferentes parámetros disponibles para el monitoreo en el HMI.

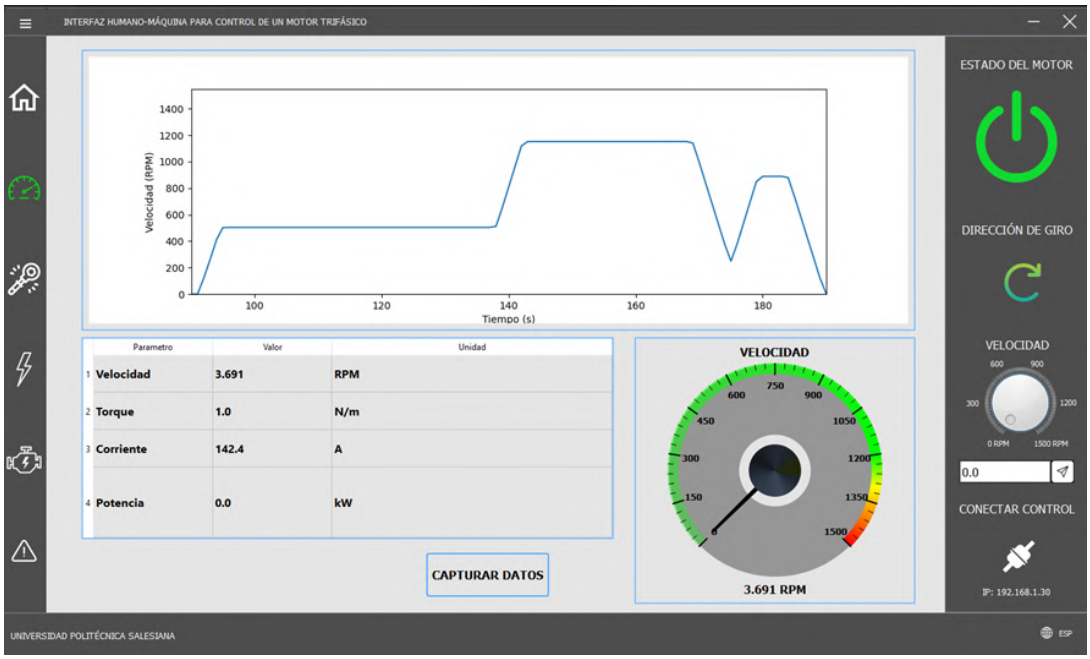


Figura 4.8: Interfaz HMI con la gráfica de velocidad

Como se pudo observar en las figuras anteriores, la curva en la gráfica muestra los últimos 100 puntos guardados. Además, se pueden observar los otros parámetros de forma instantánea desde la tabla de valores. Los valores de la gráfica se guardan en un vector que puede ser exportado a un documento aparte con el botón de capturar.

Finalmente, para completar las pruebas, forzamos un error quitando la alimentación del variador de frecuencia, con la finalidad de que salten las alarmas correspondientes y el motor se bloquee. Cuando la alarma salta, es imposible cambiar de página hasta que no se acuse dicho error, además, el motor se apaga y no se puede encender hasta que las alarmas no desaparezcan. Si se intenta acusar el error sin antes corregirlo se borran las alarmas, sin embargo, inmediatamente vuelven a saltar dado que el problema no fue corregido. En las siguientes figuras se aprecia la ventana de error con las alarmas por el error forzado, y posteriormente el estado del motor (apagado) luego de acusar los errores.

Figura 4.9: Interfaz HMI con la gráfica de corriente

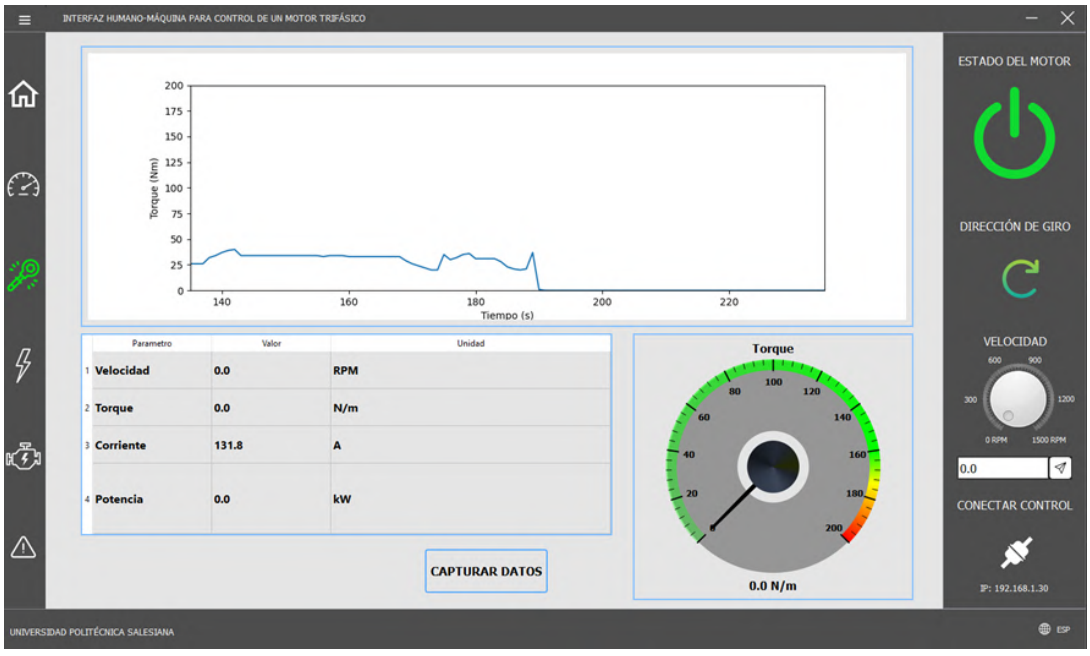


Figura 4.10: Interfaz HMI con la gráfica de torque

Figura 4.11: Interfaz HMI con la gráfica de potencia

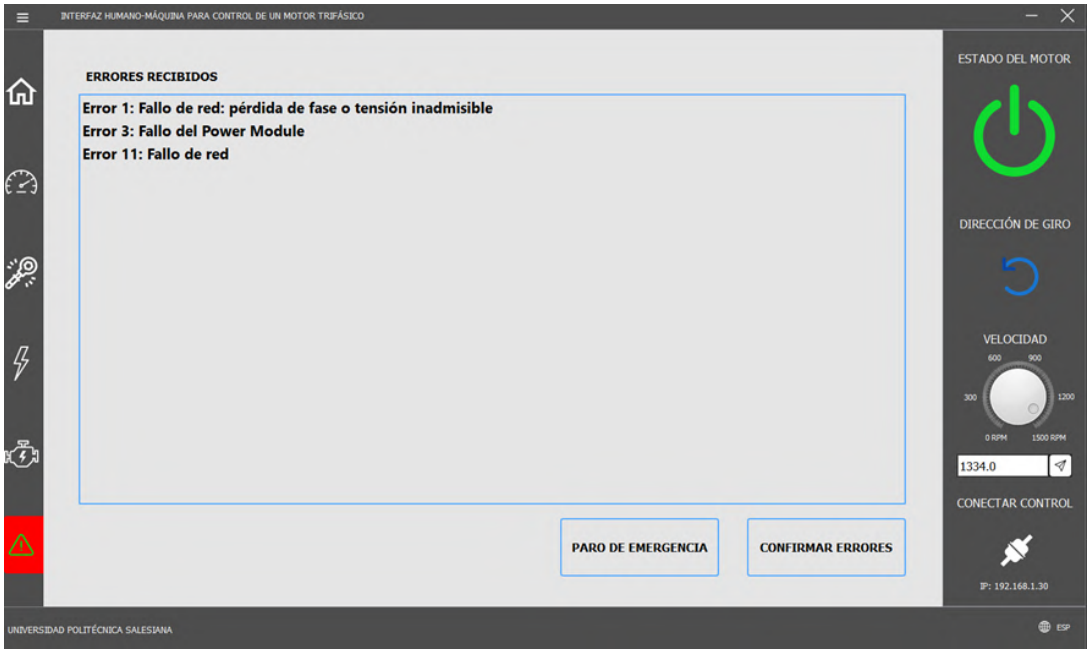


Figura 4.12: Interfaz HMI con alerta por errores en el motor

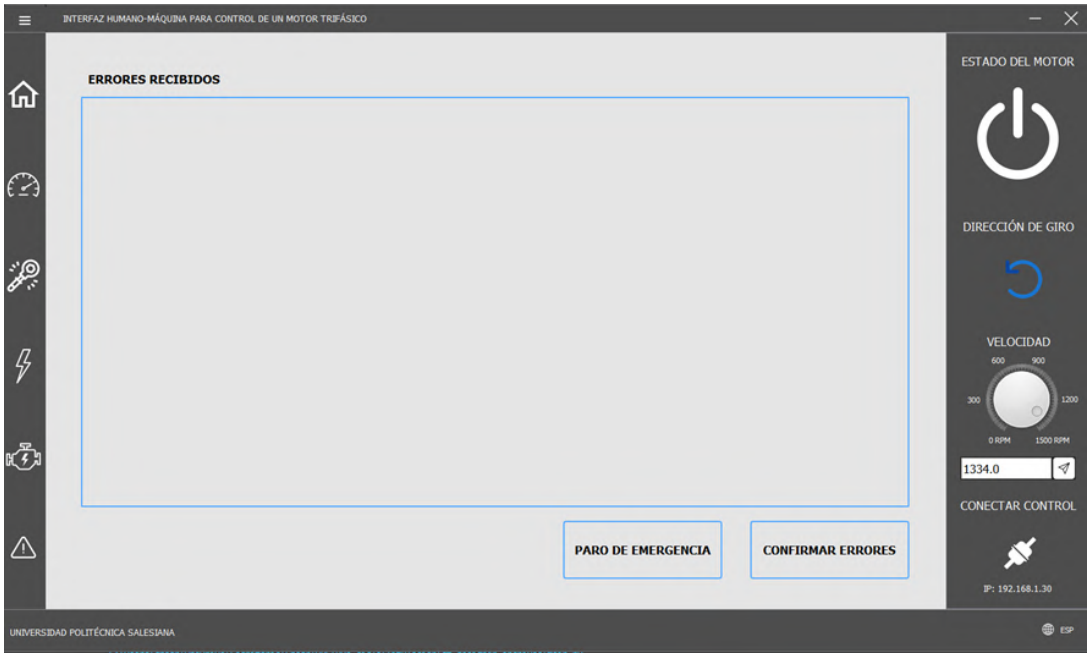


Figura 4.13: Interfaz HMI con los errores acusados

Capítulo 5

Conclusiones y Trabajos Futuros

El proyecto ha cumplido con los objetivos propuestos al implementar un sistema de control y comunicación inalámbrica para un motor trifásico de inducción.

La implementación del sistema de control fue exitosa, permitiendo la operación del motor trifásico mediante el variador SINAMICS G120. Se configuraron los parámetros necesarios para su correcto funcionamiento y se estableció la comunicación PROFINET para el intercambio de datos con los otros dispositivos pertenecientes a la subred. La utilización del telegrama de comunicación estándar 20, mediante la implementación del bloque SINA_SPEED_TLG2020, permitió una comunicación rápida, efectiva y segura. Además, la implementación del Bloque de Datos facilita el intercambio de información entre el sistema de control y el HMI, optimizando la comunicación de manera organizada y en tiempo real.

La configuración del módulo SCALANCE W774 fue efectiva, logrando establecer una conexión inalámbrica segura entre el PLC y la estación de monitoreo remoto. Esta configuración permitió la transmisión confiable de datos y la visualización en tiempo real desde cualquier ubicación con acceso a la red WLAN, asegurando la protección mediante la autenticación WPA2.

La interfaz HMI desarrollada en PYTHON brindó una experiencia amigable y accesible para el usuario, permitiendo la supervisión y control del motor desde la estación de monitoreo remoto. Se implementaron diversas funciones para visualizar los parámetros del motor, accionar el motor y recibir alertas en caso de errores o fallos, mejorando la operabilidad del sistema.

En cuanto al diseño del HMI en sí, se puede concluir que es necesaria la integración de múltiples herramientas. En este caso, la elección de Qt Designer como la aplicación para el diseño de la interfaz ofrece una compatibilidad amplia además de permitir crear interfaces gráficas de alta calidad e intuitivas. Usando el framework de PyQt5, compatible con Qt Designer facilita la traducción a Python, lo que a su vez aumenta la compatibilidad con los elementos y librerías usados para el desarrollo de todo el sistema.

Se recomienda el uso de Layouts y frames para organizar los elementos de la interfaz ha resultado en una apariencia atractiva y una experiencia de usuario más amigable. La utilización de estilos personalizados y animaciones ha enriquecido la interfaz visual y facilitado el aprendizaje y utilización de la aplicación.

La habilitación y deshabilitación controlada de los botones en función de la conexión con el PLC ha asegurado que solo un dispositivo pueda modificar los datos a la vez, garantizando un acceso seguro. De este modo se evitan conflictos por el manejo del mismo, salvaguardando así no solo la integridad del motor, sino que, ya para entornos industriales, esto garantiza la seguridad de los trabajadores y los procesos al evitarse cambios bruscos o errores en el proceso.

La característica de autenticación para el control de motor, junto con la detección inmediata de errores y el monitoreo constante del estado del motor han añadido una capa adicional de seguridad y diagnóstico, permitiendo una respuesta rápida y efectiva ante cualquier problema, siendo recomendable así para mantener un sistema seguro y confiable.

En conjunto, el sistema HMI desarrollado ha demostrado ser una solución valiosa para aplicaciones industriales y de control de maquinarias. La combinación de funcionalidades intuitivas, actualización de datos en tiempo real, control de acceso y seguridad, y la capacidad de adaptarse a diferentes idiomas, ofrece una experiencia de usuario completa, eficiente y segura en un entorno multiestación.

Como propuestas de trabajos futuros, se podría considerar la integración de sistemas de Inteligencia Artificial para mejorar el rendimiento del sistema. Esto incluiría el desarrollo de algoritmos de detección de fallos o mantenimiento

preventivo basados en los datos obtenidos del motor. La implementación de técnicas de aprendizaje automático podría permitir una optimización continua del sistema y una reducción de los tiempos de inactividad.

Además, se podría explorar la posibilidad de expandir la comunicación inalámbrica a otros dispositivos industriales, permitiendo una integración más amplia y completa de procesos de automatización en toda la planta.

Asimismo, se podrían plantear soluciones de ciberseguridad específicas para proteger la comunicación inalámbrica y los datos del sistema contra posibles amenazas y ataques externos

Glosario

AP Punto de Acceso - Access Point.

DB Bloque de Datos - Data Block.

HMI Interfaz Humano Máquina - Human Machine Interface.

IOP Panel de Operación Inteligente - Intelligent Operation Panel.

IWLAN Red de Área Local Inalámbrica Industrial - Industrial Wireless Local Area Network.

PLC Controlador Lógico Programable - Programmable Logic Controller.

WBM Gestión Basada en Web - Web Based Management.

Referencias

[1] B. Kwapisz, M. Doligalski, M. Ochowiak et al., «Monitoring of Measuring Devices Using a Programmable Logic Controller and a Dedicated Desktop Application,» *SENSORS*, vol. 22, n.º 23, dic. de 2022. DOI: 10.3390/s22239313.

[2] H. González-Acevedo y O. G. Villamizar-Galvis, «Implementación de un sistema de control para regular la velocidad y posición de motores industriales utilizando el protocolo de comunicación OPC,» *Revista UIS Ingenierías*, vol. 18, n.º 2, págs. 147-158, feb. de 2019. DOI: 10.18273/revuin.v18n2-2019014. dirección: https://revistas.uis.edu.co/index.php/revistauisingenierias/article/view/9046.

[3] F. Asadi, S. Phumpho y S. Pongswatd, «Remote monitoring and alert system of HV transformer based on FMEA,» *ENERGY REPORTS*, vol. 6, n.º 9, págs. 807-813, dic. de 2020, ISSN: 2352-4847. DOI: 10.1016/j.egyr.2020.11.128.

[4] Siemens AG, «Brochure IWLAN Industrial Wireless LAN EN,» 2022. dirección: https://cache.industry.siemens.com/dl/files/732/109807732/att_1131233/v1/IWLAN_EN.pdf.

[5] X. Gong, D. Plets, E. Tanghe, T. D. Pessemier, L. Martens y W. Joseph, «An efficient genetic algorithm for large-scale transmit power control of dense and robust wireless networks in harsh industrial environments,» *Applied Soft Computing*, vol. 65, págs. 243-259, abr. de 2018, ISSN: 1568-4946. DOI: 10.1016/J.ASOC.2018.01.016.

[6] Siemens AG, «IWLAN – Wireless LAN for industry - Industrial Communication - Global,» dirección: https://www.siemens.com/global/en/products/automation/industrial-communication/industrial-wireless-lan.html.

[7] Siemens AG, «6GK5774-1FX00-0AB0 - Industry Support Siemens,» jun. de 2023. dirección: https://support.industry.siemens.com/cs/pd/343257?pdti=td&dl=es&pnid=15866&lc=es-EC.

[8] Siemens AG, «Liberación para el suministro con restricciones del SCALANCE W774-1 RJ45 y del SCA... - ID: 85477515 - Industry Support Siemens,» ene. de 2014. dirección: https://support.industry.siemens.com/cs/document/85477515/liberaci%C3%B3n-para-el-suministro-con-restricciones-del-scalance-w774-1-rj45-y-del-scalance-w734-1-rj45?dti=0&lc=es-EC.

[9] Siemens AG, «SCALANCE W734 RJ45 para el armario eléctrico,» 2022. dirección: https://mall.industry.siemens.com/mall/es/es/Catalog/Products/10205204.

[10] Siemens AG, «SCALANCE W774 RJ45 para el armario eléctrico,» 2022. dirección: https://mall.industry.siemens.com/mall/es/WW/Catalog/Products/10257571#S.

[11] Siemens AG, «SIMATIC NET Industrial Wireless LAN SCALANCE W770 / W730 according to IEEE 802.11n Web Based Management V6.5 Configuration Manual 04/2022,» 2013.

[12] Cisco Systems Inc, «What Is Power over Ethernet (PoE)? - Cisco,» dirección: https://www.cisco.com/c/en/us/solutions/enterprise-networks/what-is-power-over-ethernet.html.

[13] Siemens AG, «Detalles del producto - Industry Mall - Siemens Spain,» 2022. dirección: https://mall.industry.siemens.com/mall/es/es/Catalog/Product/6GK57954MA000AA3.

[14] Siemens AG, «SIMATIC NET Network management SINEC PNI Operating Instructions 12/2021,» 2020.

[15] Siemens AG. «Descarga del software SINEC PNI Basic V1.0 + Service Pack 1 + Update 1 - ID: 109804190 - Industry Support Siemens.» (jun. de 2022), dirección: https://support.industry.siemens.com/cs/document/109804190/descarga-del-software-sinec-pni-basic-v1-0-service-pack-1-update-1?dti=0&lc=es-EC.

[16] WinPcap. «WinPcap - Home.» (2018), dirección: https://www.winpcap.org/.

[17] Microsoft. «Download Visual C++ Redistributable para Visual Studio 2015 from Official Microsoft Download Center.» (2023), dirección: https://www.microsoft.com/es-es/download/details.aspx?id=48145.

[18] D-Link. «What is Web-based Management? | D-Link UK.» (), dirección: https://eu.dlink.com/uk/en/support/faq/access-points-and-range-extenders/what-is-web-based-management.

[19] SRGWIN. «Sistema web - para la gestión y monitorización de redes.» (), dirección: https://www.sgrwin.com/es/web-based-network-management/.

[20] Siemens AG. «¿Cómo se establece una conexión inicial entre el punto de acceso y el módulo clie... - ID: 109798584 - Industry Support Siemens.» (ene. de 2022), dirección: https://support.industry.siemens.com/cs/document/109798584/%C2%BFc%C3%B3mo-se-establece-una-conexi%C3%B3n-inicial-entre-el-punto-de-acceso-y-el-m%C3%B3dulo-cliente-scalance-w700-mediante-web-based-management-(wbm)-?dti=0&lc=es-EC.

[21] S. Chandra, S. Kumar y S. kumar Singh, *An Introduction to Client Server Computing*. 2009.

[22] H. S. Oluwatosin, «Client-Server Model,» *IOSR Journal of Computer Engineering*, vol. 16, págs. 57-71, 1 2014, ISSN: 22788727. DOI: 10.9790/0661-16195771. dirección: http://www.iosrjournals.org/iosr-jce/papers/Vol16-issue1/Version-9/J016195771.pdf.

[23] M. A. C. PELAYO y M. A. 6. C. PELAYO, «Implementación y análisis de desempeño de un sistema de comunicación cliente-servidor para la difusión científica y tecnológica.,» *Exploraciones, intercambios y relaciones entre el diseño y la tecnología*, págs. 57-79, 2014. DOI: 10.16/CSS/JQUERY.DATATABLES.MIN.CSS. dirección: http://risisbi.uqroo.mx/handle/20.500.12249/1383.

[24] W. McKinney, *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*. O'Reilly Media, 2012, ISBN: 9781449323615. dirección: https://books.google.com.bo/books?id=BvoangEACAAJ.

[25] M. Lutz, *Learning Python: Powerful Object-Oriented Programming* (Safari Books Online). O'Reilly Media, 2013, ISBN: 9781449355692. dirección: https://books.google.com.ec/books?id=4pgQfXQvekcC.

[26] A. Mechtley y R. Trowbridge, «Chapter 8 - Advanced Graphical User Interfaces with Qt,» en *Maya Python for Games and Film*, A. Mechtley y R. Trowbridge, eds., Boston: Morgan Kaufmann, 2012, págs. 233-258, ISBN: 978-0-12-378578-7. DOI: https://doi.org/10.1016/B978-0-12-378578-7.00008-9. dirección: https://www.sciencedirect.com/science/article/pii/B9780123785787000089.

[27] G. Moruzzi, «Tkinter Graphics,» en *Essential Python for the Physicist*. Cham: Springer International Publishing, 2020, págs. 115-126, ISBN: 978-3-030-45027-4. DOI: 10.1007/978-3-030-45027-4_6. dirección: https://doi.org/10.1007/978-3-030-45027-4_6.

[28] D. Abbott, «Chapter 11 - Graphics programming with QT,» en *Linux for Embedded and Real-Time Applications (Fourth Edition)*, D. Abbott, ed., Fourth Edition, Newnes, 2018, págs. 173-185, ISBN: 978-0-12-811277-9. DOI: https://doi.org/10.1016/B978-0-12-811277-9.00011-0. dirección: https://www.sciencedirect.com/science/article/pii/B9780128112779000110.

[29] H. Wardak, S. Zhioua y A. Almulhem, «PLC access control: a security analysis,» en *2016 World Congress on Industrial Control Systems Security (WCICSS)*, 2016, págs. 1-6. DOI: 10.1109/WCICSS.2016.7882935.

[30] T. Raza, W. Lang y R. Jedermann, «Integration of Wireless Sensor Networks into Industrial Control Systems,» en *Dynamics in Logistics*, M. Freitag, H. Kotzab y J. Pannek, eds., Cham: Springer International Publishing, 2017, págs. 209-218, ISBN: 978-3-319-45117-6.

[31] Siemens AG, «Convertidores estándar SINAMICS G120,» 2023. dirección: https://mall.industry.siemens.com/mall/es/WW/Catalog/Products/10215579.

[32] Siemens AG, «Convertidores con las Control Units CU250S-2,» dirección: https://cache.industry.siemens.com/dl/files/554/94020554/att_56862/v1/G120_CU250S2_BA13_0414_esp.pdf.

[33] Siemens AG, «SINAMICS Intelligent Operating Panel 2 (IOP-2),» 2017. dirección: https://media.automation24.com/manual/es/iop1_op_instr_0417_es-ES.pdf.

[34] Siemens AG, «Library LSINAExt Control of a SINAMICS drive via function blocks,» 2019. dirección: http://www.siemens.com/industrialsecurity..

[35] Siemens AG. «SINAMICS S/G/V: Simple cyclic Funktions Blocks for Controlling a SINAMICS in TIA ... - ID: 109747655 - Industry Support Siemens.» (2022), dirección: https://support.industry.siemens.com/cs/document/109747655/sinamics-s-g-v-simple-cyclic-funktions-blocks-for-controlling-a-sinamics-in-tia-portal-?dti=0&lc=en-AR#!#.