UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE GUAYAQUIL

CARRERA DE INGENIERÍA ELECTRÓNICA

TRABAJO DE TITULACIÓN PREVIA A LA OBTENCIÓN DEL TÍTULO DE:
INGENIERO ELECTRÓNICO

PROYECTO TÉCNICO:
SISTEMA SCADA PARA VARIABLES ELÉCTRICAS DE MOTORES TRIFÁSICOS MEDIANTE BUSES DE CAMPO E IOT

AUTORES:
JENNY ROSARIO CEDILLO UVIDIA
VALERIA LISBETH SAMANIEGO BUENO

TUTOR:
ING. BYRON LIMA CEDILLO MSc.

GUAYAQUIL - ECUADOR
2021

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA

Nosotras, Jenny Rosario Cedillo Uvidia con cédula de identidad N°. 0953337011 y Valeria Lisbeth Samaniego Bueno con cédula de identidad N°.0923423750, declaramos que este trabajo de titulación “SISTEMA SCADA PARA VARIABLES ELÉCTRICAS DE MOTORES TRIFÁSICOS MEDIANTE BUSES DE CAMPO E IOT” ha sido implementado bajo los conceptos, análisis y conclusiones considerando los métodos de investigación, así como también el respeto a los derechos intelectuales a terceros, son de exclusiva responsabilidad de los autores; y la propiedad intelectual de la UNIVERSIDAD POLITÉCNICA SALESIANA.

Guayaquil, agosto del 2021

Jenny Rosario Cedillo Uvidia
CI: 0953337011

Valeria Lisbeth Samaniego Bueno
CI: 0923423750

CERTIFICADO DE CESIÓN DE DERECHO

Nosotras, Jenny Rosario Cedillo Uvidia con cédula de identidad N°.0953337011 y Valeria Lisbeth Samaniego Bueno con cédula de identidad N°.0923423750, manifestamos nuestra voluntad de ceder la titularidad sobre los derechos patrimoniales de este trabajo de titulación “SISTEMA SCADA PARA VARIABLES ELÉCTRICAS DE MOTORES TRIFÁSICOS MEDIANTE BUSES DE CAMPO E IOT” a la UNIVERSIDAD POLITÉCNICA SALESIANA según lo establecido por la ley de la propiedad intelectual y por la normativa institucional vigente.

Guayaquil, agosto del 2021

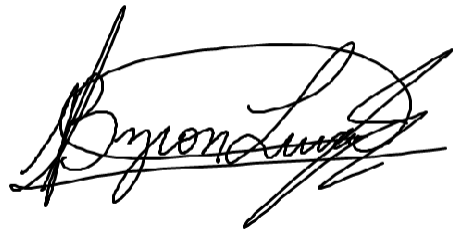
Jenny Rosario Cedillo Uvidia
CI: 0953337011

Valeria Lisbeth Samaniego Bueno
CI: 0923423750

CERTIFICADO DE DIRECCIÓN DE TRABAJO DE TITULACIÓN

Por medio de la presente declaro que bajo mi dirección y asesoría fue desarrollado este trabajo de titulación “SISTEMA SCADA PARA VARIABLES ELÉCTRICAS DE MOTORES TRIFÁSICOS MEDIANTE BUSES DE CAMPO E IOT” realizado por los estudiantes JENNY ROSARIO CEDILLO UVIDIA con cédula de identidad N°. 0953337011 y VALERIA LISBETH SAMANIEGO BUENO con cédula de identidad N°. 0923423750, el mismo que cumple con los objetivos del diseño de aprobación y todos los requisitos pertinentes.

Guayaquil, agosto del 2021



Ing. Byron Lima Cedillo, MSc.

Tutor de proyecto

DEDICATORIA

Dedico este trabajo principalmente a Dios quien estuvo presente en el caminar de mi vida y dándome fuerza para continuar con mis metas trazadas sin desfallecer y a mi familia por haber sido mi apoyo a lo largo de mi carrera universitaria.

Jenny Rosario Cedillo Uvidia

A Dios por concederme vida, salud y sabiduría para la toma de decisiones correctas durante el diario vivir como estudiante universitaria, mismas que han servido para poder culminar mis estudios de tercer nivel.

A mis padres Héctor Ramiro y Nila Justina a quienes amo con todo mi corazón, ellos representan la fuerza espiritual que tuve para superar barreras que le pone la vida y alcanzar el anhelo de ser profesional.

A mi hermano Alan Antonio que fue mi confidente en los momentos difíciles, siempre estuvo dispuesto a darme un consejo de cómo se debe actuar ante cualquier adversidad.

A toda mi familia que siempre estuvo pendiente de mí, brindándome su apoyo incondicional; a ellos mi eterna gratitud.

Valeria Lisbeth Samaniego Bueno

AGRADECIMIENTO

Quiero expresar un sincero agradecimiento a mis maestros quienes conforman la universidad politécnica salesiana por todas las pautas que me han dado para mi educación profesional y a mi tutor por tenernos paciencia y ser un buen guía en este trabajo.

Jenny Rosario Cedillo Uvidia

A la Universidad Politécnica Salesiana de Guayaquil, por permitirme ser parte de su historia, pues en sus aulas fue donde adquirí los conocimientos necesarios para culminar mis estudios con éxito.

A todos mis profesores que estuvieron conmigo durante los años de estudio, ellos a través de sus conocimientos forjaron el carácter necesario de cada uno de sus estudiantes y nos brindaron la oportunidad de ser personas de bien y ser profesionales en el área que nos gusta.

A mis amigos y compañeros con los que compartí las aulas de nuestra universidad, siempre los recordare.

Valeria Lisbeth Samaniego Bueno

ABSTRACT

YEAR	STUDENTS	PROJECT TUTOR	PROJECT THEME
2021	Jenny Rosario Cedillo Uvidia Valeria Lisbeth Samaniego Bueno	Eng. Byron Lima Cedillo MSc.	"SCADA system for electrical variables of three-phase motors using field buses and IOT".

This project shows the development of a system of monitoring, control and acquisition of data for remote observation of the electrical parameters of three-phase motors through the principles of the Internet of Things. With this application it is possible to observe the behavior of the engine and in turn compare with the nominal values and thus contribute to the decision-making of corrective and preventive maintenance.

Fieldbuses have been used to avoid parallel wiring limitations and obtain the most estimated data on frequency drives. In addition, a National Instrument OPC server is used as an intermediary between the PLC connected to the network actuators and the data to be sent to the cloud through Ubidots. SCADA is designed and implemented in LabVIEW software applying regulations for the development of HMI applications for industrial processes.

RESUMEN

AÑO	ALUMNOS	TUTOR DEL PROYECTO	TEMA DEL PROYECTO
2021	Jenny Rosario Cedillo Uvidia Valeria Lisbeth Samaniego Bueno	Ing. Byron Lima Cedillo MSc.	“Sistema SCADA para variables eléctricas de motores trifásicos mediante buses de campo E IOT”.

El presente proyecto muestra el desarrollo de un sistema de supervisión, control y adquisición de datos para la observación remota de los parámetros eléctricos de motores trifásicos mediante los principios del internet de las cosas. Con esta aplicación es posible observar el comportamiento del motor y a su vez comparar con los valores nominales y así contribuir a la toma de decisiones de mantenimiento correctivo y preventivo.

Se ha utilizado buses de campo para evitar las limitaciones del cableado paralelo y obtener la mayor cantidad de datos estimados en los variadores de frecuencia. Adicional a ello se hace uso de un servidor OPC de National Instrument como intermediario entre el PLC conectado a los actuadores en red y los datos a enviarse a la nube a través de Ubidots. El SCADA es diseñado e implementado en el software LabVIEW aplicando normativas para el desarrollo de aplicaciones HMI para procesos industriales.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Hoy en día, los sistemas de monitoreo para procesos industriales representan una gran herramienta para la toma de decisiones sobre posibles ampliaciones, modificaciones o priorización de actividades productivas. Esto ha sido posible debido las grandes ventajas que tienen estos desarrollos y la facilidad que existe en la aplicación de sistemas de inteligencia artificial orientados al análisis actual y predicción de futuras condiciones que podrían afrontarse a futuro. Es obvio pensar que al saber que habrá una posible falla, es posible prevenirla. En esencia es así, pero existe un factor importante a considerar, los datos. Siempre es requerido un sistema que permita la extracción de información importante y que sea mostrada de una forma amigable y remotamente.

Gracias al internet de las cosas, este monitoreo a distancia es posible y bastante utilizado por la mayoría de las empresas multinacionales en la actualidad. Debido a la relevancia de esta temática, el proyecto propuesto describe la aplicación de un método para monitoreo remoto de parámetros eléctricos de motores que se encuentran controlados a través de un bus de campo. Para ello se considerarán algunos elementos principales, tales como un PLC S7-1500, variadores de velocidad de diferentes fabricantes (Siemens y ABB) para motores de 0.5HP, red de comunicación Industrial Ethernet, entre otros.

En la presente memoria técnica se incluyen algunas secciones que ayudarán a la comprensión del proyecto propuesto. Primeramente, se encuentra la sección el problema, donde se aborda la motivación del proyecto y se definen los objetivos. En segundo lugar, se encuentra la sección marco teórico donde se muestra la fundamentación técnica de los elementos principales. Luego de ello se tiene el marco metodológico para indicar el proceso de realización de sistema de monitoreo. Finalmente se muestran los resultados obtenidos, el análisis de estos y las conclusiones respectivas.

1. EL PROBLEMA

1. Planteamiento del problema

El constante avance de tecnologías se ha derivado en aumentar e innovar los recursos necesarios para satisfacer a los nuevos estudiantes en el aprendizaje de tal forma que este sea óptimo y beneficiar a los mismos.

Una condición necesaria es incrementar los recursos en los respectivos laboratorios de la Carrera Ingeniería Electrónica para utilizar y generar nuevas experiencias concretas, vivenciales en el ámbito académico y profesional. Este avance obliga al estudiante al momento de especializarse en el campo del monitoreo remoto mediante la nube se vea en la necesidad de capacitarse a modo de formación continua para complementar lo impartido en el aula de clase.

Como parte integral en la formación de los profesionales de la industria, es necesario para los estudiantes conocer de estas nuevas tecnologías. Para aumentar las opciones en el diseño de soluciones integrales de automatización se plantea este proyecto de esta manera profundizar en los controladores lógicos programables de alta gama, S7-1500, en el diseño de interfaces humano-máquina y la interacción con motores trifásicos a través de variadores de frecuencia y buses de campo.

2. Antecedentes

Debido a los continuos cambios en la llamada cuarta revolución industrial, los centros educativos y universidades se ven en la obligación de incluir en su metodología la parte del control y la automatización de los procesos industriales.

Adicionalmente el creciente uso de sistemas de monitoreo remotos y la gestión de datos por medio de interfaces y protocolos de manera global permiten la conexión entre diferentes sistemas dando paso a la creación del presente proyecto.

Para conocer los sistemas industriales de automatización integrándose al IOT en nivel académico se plantea en este proyecto de titulación una guía didáctica.

3. Importancia y alcance

En la actualidad los avances tecnológicos en el área de automatización de sistemas han llevado a la integración de diversas áreas de ingeniería que inicialmente se manejaban de forma independiente. Por ejemplo, el control y monitoreo de variables eléctricas de un proceso automatizado se puede monitorear desde diferentes partes del mundo a través del uso de la red global. Por ende, resulta importante la combinación de conocimientos del área de automatización y telecomunicaciones para impulsar la implementación de nuevos proyectos para la supervisión y evaluación de sistemas.

4. Delimitación del problema

1.4.1. Delimitación espacial

El proyecto tiene como escenario el Laboratorio de Automatización II localizado en el tercer piso del bloque E de la Universidad Politécnica Salesiana sede Guayaquil. En este laboratorio se encuentran módulos para prácticas estudiantiles sobre controladores lógicos programables.

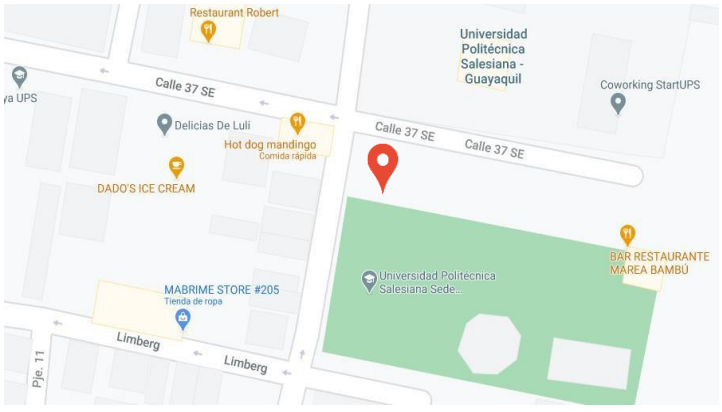


Figura01. Ubicación del laboratorio [1]

1.4.2. Delimitación temporal

El proyecto fue planificado para ser realizado en 6 meses.

1.4.3. Delimitación académica

Por medio de este proyecto se desea motivar a los estudiantes de Ingeniería Electrónica y carreras afines de la Universidad Politécnica Salesiana sede Guayaquil, a la implementación de sistemas de monitoreo en la nube utilizando los principios del Internet de las cosas - IOT.

Para la implementación de estos sistemas se ha utilizado controladores lógicos programables de última tecnología con muy buenas prestaciones para el ámbito industrial, precisamente hablando del modelo S7-1500. Adicional a ello se considerará la conexión a un servidor OPC con el software LabVIEW para la publicación de los valores de monitoreo por una comunicación http a una plataforma IOT alojada en la nube (Ubidots).

5. Objetivos

1.5.1. Objetivo general

Desarrollar un sistema de monitoreo remoto para variables eléctricas de motores trifásicos mediante buses de campo e internet mediante una plataforma IOT.

1.5.2. Objetivos específicos

- Desarrollar una guía para la creación de un sistema de monitoreo remoto para la conexión con una plataforma IOT.
- Utilizar los recursos del laboratorio de automatización de la Universidad Politécnica Salesiana para la creación de una red PROFIBUS y PROFINET teniendo un controlador PLC S71500 para el control de los motores.
- Desarrollar una conexión a una plataforma IOT mediante el software LabVIEW comunicándose con un servidor OPC y un PLC S71500 para el monitoreo de variables.
- Realizar un análisis de las ventajas de utilizar recursos de los tipos de redes PORFIBUS y el protocolo PROFINET.

2. MARCO TEÓRICO

1. Internet de las cosas (IOT)

El «Internet de las Cosas» (IOT) hace referencia, como se ha adelantado, a una tecnología basada en la conexión de objetos cotidianos a Internet que intercambian, agregan y procesan información sobre su entorno físico para proporcionar servicios de valor añadido a los usuarios finales. También reconoce eventos o cambios, y tales sistemas pueden reaccionar de forma autónoma y adecuada. Su finalidad es, por tanto, brindar una infraestructura que supere la barrera entre los objetos en el mundo físico y su representación en los sistemas de información. [2]

El término Internet de las Cosas (IOT) fue acuñado por primera vez por el pionero de la tecnología británica Kevin Ashton9 en una presentación que realizó en 1999 para la multinacional Procter & Gamble, donde describía un sistema en el cual los objetos en el mundo físico podrían conectarse a Internet a través de sensores para automatizar la recogida de datos, propugnando su aplicación en la cadena de suministro añadiéndoles etiquetas RFID (o identificación por radiofrecuencia, Radio Frequency Identification), que más adelante examinaremos con detalle. Concretamente, el padre del término destacaba cómo, hasta esa fecha, la información en la cadena de suministro era introducida de forma manual por las personas, con los consiguientes retrasos y posibles errores. Sin embargo, si la información provenía directamente de los objetos, entonces se podría hacer un seguimiento en tiempo real de su utilización, sus características, su vida útil, la necesidad de nuevos aprovisionamientos, su estado de funcionamiento, etc., lo cual se traduciría en aumentos de productividad y consiguiente reducción de costes. En la actualidad, además de los sistemas de etiquetas, tarjetas y transpondedores RFID, los datos también se recaban mediante los sensores Wireless, las cookies, así como otras tecnologías de seguimiento y captación de datos a las que haremos próxima referencia en el capítulo siguiente. Así, en su origen, el IOT nace como una forma de facilitar información, en la cadena de suministro, de bienes a las empresas. Pero, con posterioridad, se

extiende a todo tipo de objetos físicos y digitales, a los animales, a las personas y a los entornos o ambientes (cultivos, ecosistemas terrestres o marino). [2]

Figura02. Diversidad de equipos conectados vía IOT [3]

El IOT incluye todo, desde hogares inteligentes, dispositivos de salud móviles y juguetes conectados hasta el *Internet Industrial de las Cosas* (IIoT, por sus siglas en inglés) con agricultura inteligente, ciudades inteligentes, fábricas y las redes inteligentes. IIoT se puede caracterizar como una vasta cantidad de sistemas industriales conectados que se comunican y coordinan sus análisis de datos y sus acciones para mejorar el desempeño industrial y beneficiar a la sociedad en su totalidad. [4]

Los sistemas industriales que conectan el mundo digital con el físico a través de sensores y actuadores que resuelven problemas de control complejos se conocen comúnmente como sistemas físico-cibernéticos. Estos sistemas se están combinando con las soluciones de Big Analog Data para ganar una mirada más profunda de los datos y los análisis. Imagine sistemas industriales que se puedan ajustar a sus propios entornos o incluso a su propia salud. En lugar de continuar funcionando hasta fallar, las máquinas programan su propio mantenimiento o, aún mejor, ajustan sus algoritmos de control de forma

dinámica para compensar una parte desgastada y luego comunicar esos datos a otras máquinas y a las personas que dependen de ellas. [4]

Al hacer a las máquinas más inteligentes a través del procesamiento y la comunicación local, el IIoT puede resolver problemas de formas que antes eran inconcebibles. Pero, como dice el dicho: «Si fuera fácil, todos lo estarían haciendo». A medida que la innovación crece, también crece la complejidad. Esto hace que el IIoT sea un desafío inmenso que ninguna empresa puede afrontar por sí sola. [4]

2. Plataforma Ubidots

Ubidots es una plataforma que habilita la toma de decisiones a empresas de integración de sistemas a nivel global. Este producto permite enviar datos de sensores a la nube, configurar tableros y alertas, conectarse con otras plataformas, usar herramientas de analítica y arrojar mapas de datos en tiempo real. Es decir, hacer una gestión completa de los datos capturados para que las decisiones que toma la compañía sean informadas y precisas. [5]

Porque si algo tiene claro el equipo de Ubidots, es que "los datos son el nuevo petróleo, el nuevo paradigma". Para Agustín, "las empresas tienen muchos datos, pero no capturan más de la mitad porque no tienen Internet de las Cosas (IOT) para ayudarle a capturar los datos del entorno, además de los que tradicionales". Por esa necesidad y gracias a la “evangelización” que han hecho grandes compañías como Microsoft, Amazon, Google e IBM, Ubidots ha logrado posicionarse al punto de tener una comunidad de más de 40.000 desarrolladores y presencia en países como Argentina, Brasil, EE. UU., Inglaterra, Israel, Australia. [5]

La mayoría de sus clientes están concentrados en las áreas de salud, agricultura, monitoreo energético e Industria 4.0. En Israel, por ejemplo, usan Ubidots Industrial para ofrecer sistemas de riego inteligente y la forma en que le muestran a sus clientes que el riego está siendo más eficiente. En

Colombia, hay clientes que lo usan para monitorear su maquinaria y, a partir de los datos, saber cómo aumentar su productividad. [5]

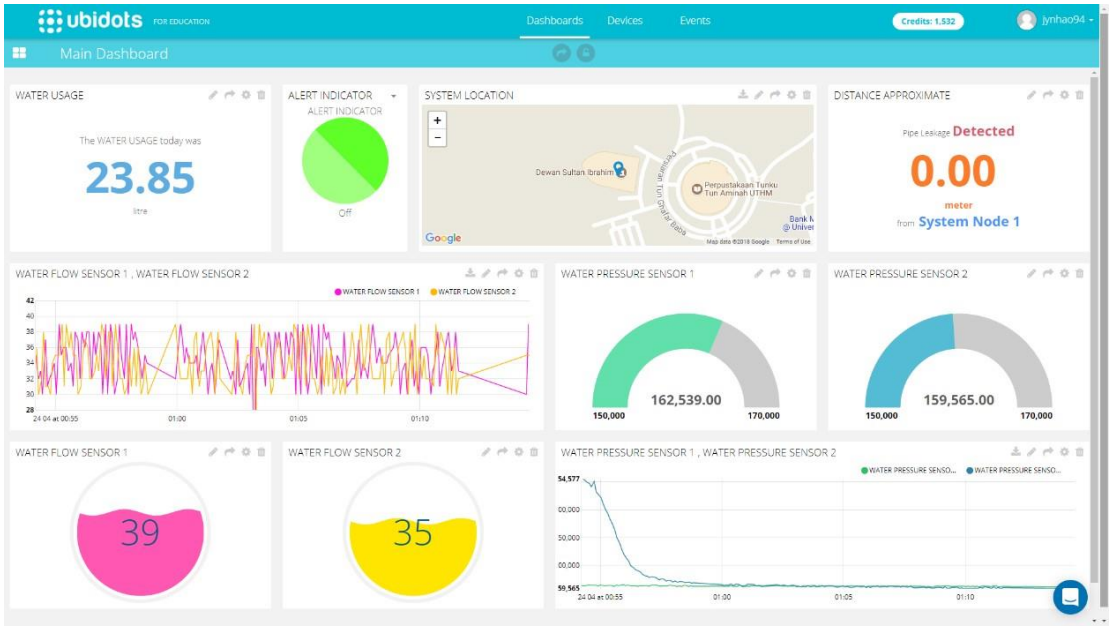


Figura03. Ejemplo de configuración de Dashboard en Ubidots. [6]

Así, Ubidots se ha convertido en una muestra de que “el ecosistema de acá funciona, y una empresa de Medellín puede acceder a fondos de capital de riesgo en todo el mundo”, según Agustín. Y de que Medellín tiene todo para desarrollar y atraer talento en temas de IOT y otras tecnologías habilitantes. [5]

2.2.1. Protocolo de comunicación MQTT

MQTT son las siglas MQ Telemetry Transport, aunque en primer lugar fue conocido como Message Queing Telemetry Transport. Es un protocolo de comunicación M2M (machine-to-machine) de tipo message queue. Está basado en la pila TCP/IP como base para la comunicación. En el caso de MQTT cada conexión se mantiene abierta y se "reutiliza" en cada comunicación. Es una diferencia, por ejemplo, a una petición HTTP 1.0 donde cada transmisión se realiza a través de conexión. [7]

MQTT fue creado por el Dr. Andy Stanford-Clark de IBM y Arlen Nipper de Arcom (ahora Eurotech) en 1999 como un mecanismo para conectar dispositivos empleados en la industria petrolera. Aunque inicialmente era un

formato propietario, en 2010 fue liberado y pasó a ser un estándar en 2014 según la OASIS (*Organization for the Advancement of Structured Information Standards*). [7]

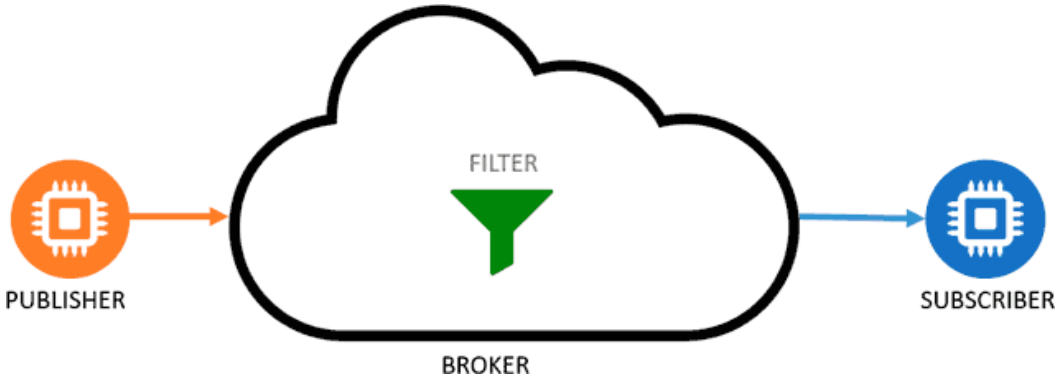


Figura04. Esquema de funcionamiento del protocolo MQTT. [7]

El funcionamiento del MQTT es un servicio de mensajería push con patrón publicador/suscriptor (pub-sub). Como vimos en la entrada anterior, en este tipo de infraestructuras los clientes se conectan con un servidor central denominado broker. [7]

Para filtrar los mensajes que son enviados a cada cliente los mensajes se disponen en topics organizados jerárquicamente. Un cliente puede publicar un mensaje en un determinado topic. Otros clientes pueden suscribirse a este topic, y el broker le hará llegar los mensajes suscritos. [7]

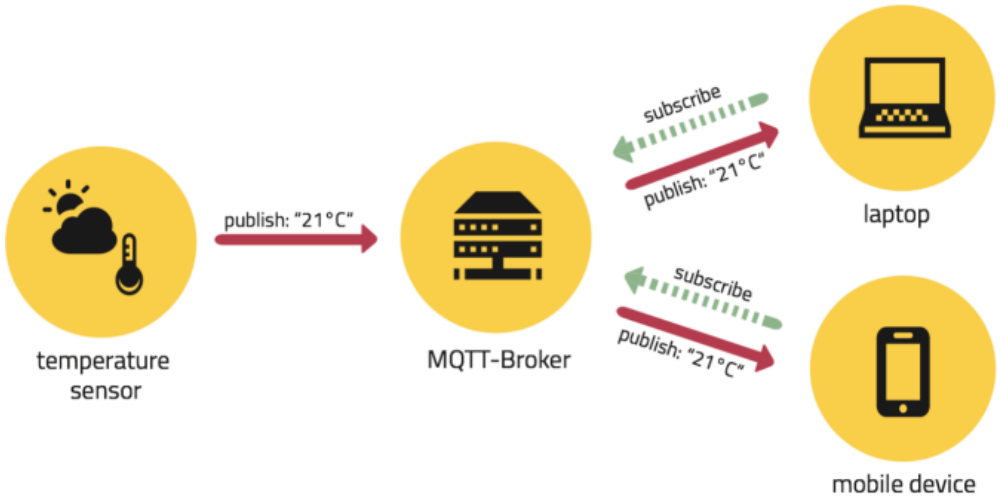


Figura05. Ejemplo de aplicación con protocolo MQTT. [8]

La seguridad siempre debe ser un factor importante por considerar en cualquier sistema de comunicación M2M. El protocolo MQTT dispone de distintas medidas de seguridad que podemos adoptar para proteger las comunicaciones. Esto incluye transporte SSL/TLS y autentificación por usuario y contraseña o mediante certificado. Sin embargo, hay que tener en cuenta que muchos de los dispositivos IOT disponen de escasa capacidad, por lo que el SLL/TLS puede suponer una carga de proceso importante. [7]

MQTT aporta una serie de características que le han hecho sobre salir sobre otros competidores. La principal, como hemos mencionado, es su sencillez y ligereza. Esto lo hace adecuado para aplicaciones IOT, donde frecuentemente se emplean dispositivos de escasa potencia. Además, esto menor necesidad de recursos se traduce en un menor consumo de energía, lo cual es interesante en dispositivos que funcionan 24/7 y muy especialmente en dispositivos alimentados por batería. [7]

Otra consecuencia de la ligereza del protocolo MQTT es que requiere un ancho de banda mínimo, lo cual es importante en redes inalámbricas, o conexiones con posibles problemas de calidad. Por último, MQTT dispone de medidas adicionales importantes, como la seguridad y calidad del servicio (QoS). Por último, es una solución largamente testada y consolidad, que aporta robustez y fiabilidad. [7]

2.2.2. Protocolo de comunicación HTTP

HTTP, de sus siglas en inglés: "Hypertext Transfer Protocol", es el nombre de un protocolo el cual nos permite realizar una petición de datos y recursos, como pueden ser documentos HTML. Es la base de cualquier intercambio de datos en la Web, y un protocolo de estructura cliente-servidor, esto quiere decir que una petición de datos es iniciada por el elemento que recibirá los datos (el cliente), normalmente un navegador Web. Así, una página web completa resulta de la unión de distintos subdocumentos recibidos, como, por ejemplo: un documento que especifique el estilo de maquetación de la página web (CSS), el texto, las imágenes, vídeos, scripts, etc. [9]

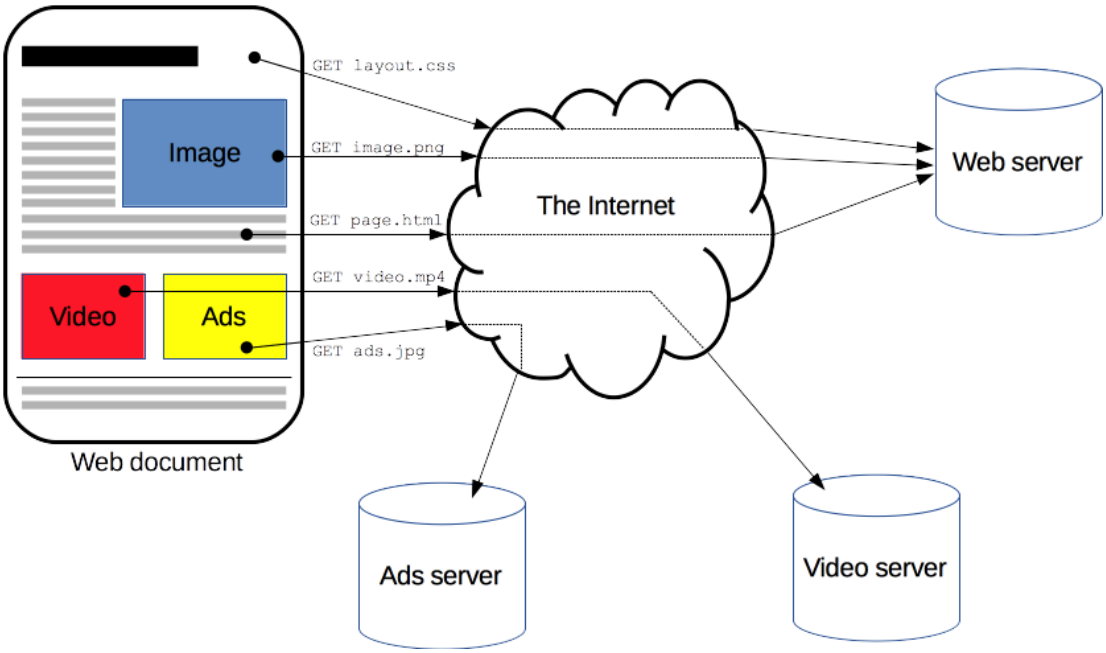


Figura06. Ejemplo de comunicación HTTP. [9]

Clientes y servidores se comunican intercambiando mensajes individuales (en contraposición a las comunicaciones que utilizan flujos continuos de datos). Los mensajes que envía el cliente, normalmente un navegador Web, se llaman peticiones, y los mensajes enviados por el servidor se llaman respuestas. [9]

La industria cuenta con algo de experiencia usando HTTP para la configuración de productos y dispositivos, pero no para el acceso a datos. De este modo, muchas plataformas TIC e IOT aceptan HTTP para proveer y recibir información, pero no así las plataformas industriales. Esto está cambiando a medida que cada vez más puertos y PLC agregan HTTP nativo. [10]

Use HTTP para enviar grandes cantidades de información, como lecturas de temperatura minuto a minuto cada hora. No use HTTP para información de video de alta velocidad. HTTP puede operar bajo el segundo, pero actualizaciones de cien milisegundos (100 ms) con HTTP son difíciles. Implica bastante sobrecarga por mensaje, así que enviar mensajes pequeños es ineficiente. Y siempre asegure la comunicación con HTTPS. La sobrecarga es mínima. [10]

2.2.3. Protocolo de comunicación TCP/UDP

El UDP (*User Datagram Protocol*) es un protocolo no orientado a la conexión, de manera que no proporciona ningún tipo de control de errores ni de flujo, aunque utiliza mecanismos de detección de errores. En caso de detectar un error, el UDP no entrega el data- grama a la aplicación, sino que lo descarta. Conviene recordar que, por debajo suyo, el UDP está utilizando IP, que también es un protocolo no orientado a la conexión. Por tanto, se pensó en definir un protocolo del nivel de transporte que permitiera que la aplicación explotara este tipo de características y que fuera simple y sencillo. [11]

La simplicidad del UDP hace que sea ideal para aplicaciones que requieren pocos retardos (por ejemplo, aplicaciones en tiempo real como pueden ser aplicaciones de voz y vídeo). UDP también es ideal para aquellos dispositivos que no pueden implementar un sistema tan complejo como el TCP. [11]

Figura07. Gráfica orientativa sobre diferencias entre TCP y UDP. [12]

Como hemos podido observar, UDP no garantiza la entrega de la información que le proporciona una aplicación. Tampoco reordena la información en caso de que llegue en un orden diferente de aquél en que se ha transmitido. Existen aplicaciones que no pueden tolerar dichas limitaciones. Para superarlas, el nivel de transporte proporciona un protocolo fiable extremo a extremo llamado TCP (*Transmission Control Protocol*). [11]

El TCP proporciona fiabilidad a la aplicación; es decir, garantiza la entrega de toda la información en el mismo orden en que ha sido transmitida por la aplicación de origen. Para conseguir esta fiabilidad, el TCP proporciona un servicio orientado a la conexión con un control de flujo y errores. [11]

A continuación, enumeramos las principales diferencias entre UDP y TCP:

- TCP está orientado a la conexión, mientras que UDP es un protocolo sin conexión. [12]
- TCP es altamente confiable para transferir datos útiles ya que toma el acuse de recibo de la información enviada. Y vuelve a enviar los paquetes perdidos si los hay. Mientras que, en el caso de UDP, si el paquete se pierde, no solicitará su retransmisión y el ordenador de destino recibirá un dato corrupto. Por lo tanto, UDP es un protocolo poco fiable. [12]
- TCP es más lento en comparación con UDP, ya que TCP establece la conexión antes de transmitir los datos y garantiza la entrega adecuada de los paquetes. Por otro lado, UDP no reconoce si los datos transmitidos son recibidos o no. [12]
- El tamaño de cabecera de UDP es de 8 bytes, y el de TCP es más del doble. El tamaño de la cabecera TCP es de 20 bytes desde entonces, y la cabecera TCP contiene opciones, relleno, suma de comprobación, banderas, desplazamiento de datos, número de confirmación, número de secuencia, puertos de origen y destino, etc. [12]
- Tanto TCP como UDP pueden comprobar si hay errores, pero sólo TCP puede corregir el error ya que tiene control de congestión y de flujo. [12]

3. Comunicación HTTP en LabVIEW

El software LabVIEW tiene disponible algunas herramientas de HTTP para crear un cliente web que interactúe con servidores, páginas web y servicios web. Entre las cuales se puede mencionar las librerías para agregar encabezados HTTP, almacenar cookies, proporcionar credenciales de

autenticación y enviar solicitudes web utilizando métodos HTTP como POST, GET, PUT, HEAD y DELETE. Estos VIs también pueden interactuar con los servicios Web de LabVIEW. A continuación, se describen los bloques utilizados en este proyecto. [13]

2.3.1. Bloque OpenHandle

Abre un identificador de cliente. Utilice identificadores de cliente para conectar varios VIs de cliente HTTP mientras conserva las credenciales de autenticación, los encabezados HTTP y las cookies. Puede especificar un nombre de usuario y una contraseña, si es necesario, para enviar solicitudes web a un servidor que requiere autenticación. También puede crear un archivo de cookies que almacene datos a través de múltiples solicitudes web. [14]

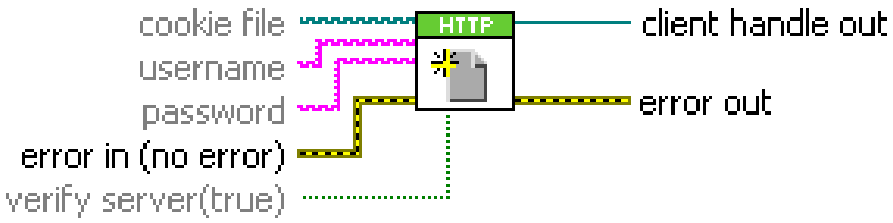


Figura08. Bloque OpenHandle, [14]

Los controles de cliente aumentan la eficiencia de la red al limitar la cantidad de puertos y sockets de red necesarios para realizar múltiples solicitudes web. Los identificadores de cliente no son necesarios cuando se realizan solicitudes web independientes sin datos persistentes, como encabezados o credenciales. [14]

La siguiente captura de pantalla muestra código que realiza lo siguiente:

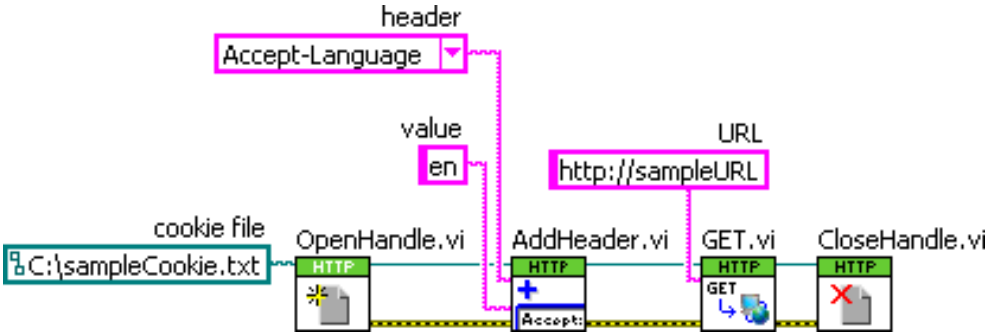


Figura09. Ejemplo de programación para cliente HTTP. [14]

- El OpenHandle VI abre un identificador de cliente y una cookie del lado del cliente que puede almacenar datos persistentes.
- El VI AddHeader agrega una nueva línea de campo de encabezado que establece el idioma preferido para las solicitudes web posteriores asociadas con el identificador del cliente.
- El GET VI realiza una solicitud web que incluye el encabezado Accept-Language.
- El CloseHandle VI cierra el identificador del cliente y elimina cualquier dato persistente, incluidos los encabezados [14]

2.3.2. Bloque Post (Buffer)

Envía una solicitud web que envía datos o un archivo a un servidor, página web o servicio web. Este VI usa el método POST HTTP. Puede asignar un identificador de cliente para agregar credenciales de autenticación, encabezados HTTP o una cookie a las solicitudes Web realizadas por el POST VI. [15]

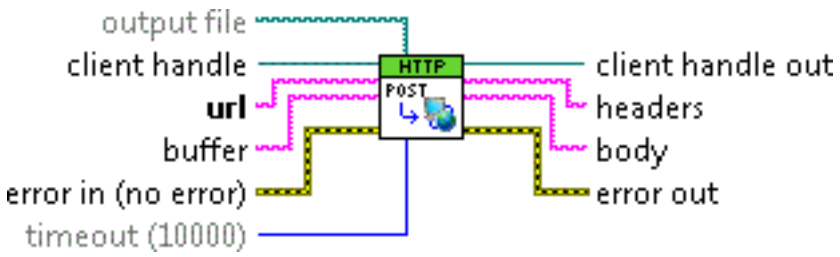


Figura10. Bloque POST (Buffer). [15]

2.3.3. Bloque Get

Envía una solicitud web que devuelve encabezados y datos del cuerpo de un servidor, página web o servicio web. Este VI usa el método GET HTTP y no envía ningún dato al servidor. También puede guardar los datos del cuerpo en un archivo de salida. [16]

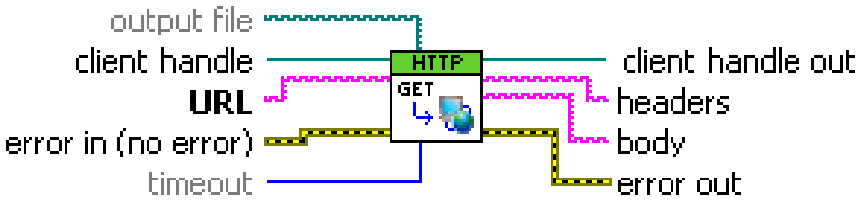


Figura11. Bloque GET. [16]

Se puede asignar un identificador de cliente para agregar credenciales de autenticación, encabezados HTTP o una cookie a las solicitudes Web realizadas por el GET VI. [16]

2.3.4. Bloque AddHeader

Agrega una línea de campo de encabezado a todas las solicitudes web asociadas con el identificador del cliente. Los encabezados definen los atributos de los datos intercambiados entre el cliente y el servidor. Si especifica un encabezado que ya existe, el valor especificado sobrescribe el valor del encabezado. [17]

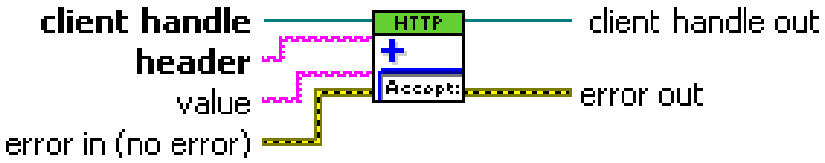


Figura12. Bloque AddHeader. [17]

4. Sistemas multibombas

Un sistema multibombas se refiere a un sistema en donde se cuenta con más de una bomba. Estos sistemas tienen un funcionamiento programado diario

igual de horas para cada bomba en funcionamiento como principal lo que prevé un desgaste igualitario de electrobombas. En la figura 13 se tiene un sistema multibombas, el principio de funcionamiento es que existe una bomba que funciona como principal la cual enciende cuando existe demanda en la red y si esta bomba no es capaz de abastecer una mayor demanda se enciende una segunda bomba y así consecutivamente hasta cubrir con la demanda del sistema. [18]

Para prever un funcionamiento equilibrado durante el día y dependiendo de la cantidad de bombas, cada una de ellas tiene un tiempo de funcionamiento como primaria, sin embargo, hay que tener en cuenta que esto no garantiza que el desgaste sea equitativo ya que esto varía de acuerdo a varios factores externos como horarios donde hay mayores demandas u otros. Para que el sistema sea capaz de hacer esta lógica se requiere de PLC y variadores de velocidad en los tableros controladores. [18]

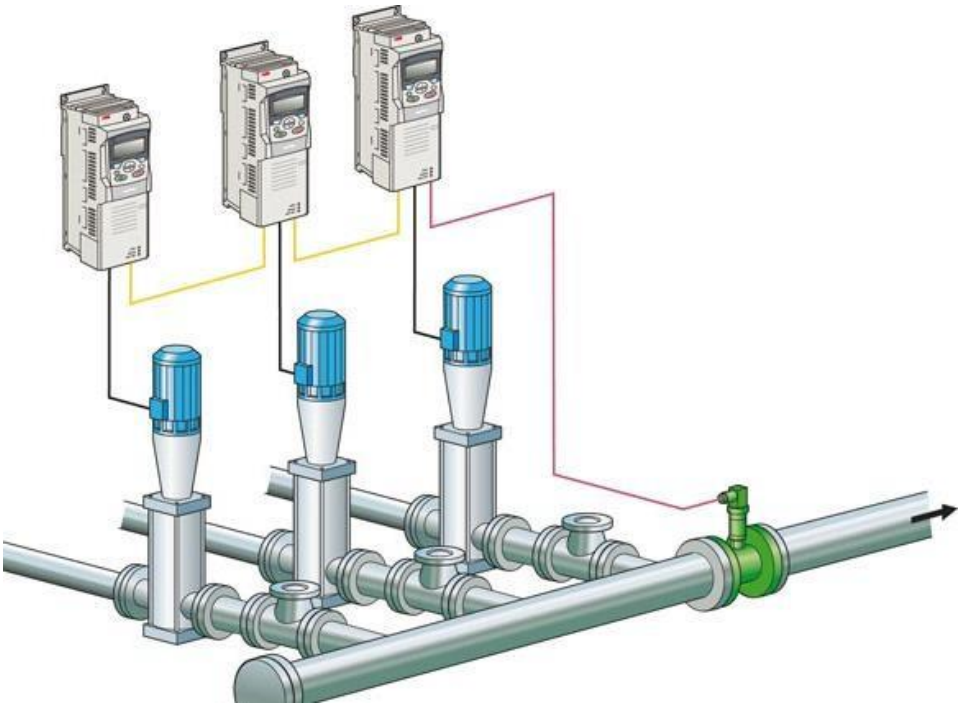


Figura13. Alternancia en sistemas de bombeo. [19]

Utilizar varias bombas en paralelo es útil cuando se exige una gran variación de caudal. La fiabilidad del servicio es otra de las ventajas. Es frecuente encontrar tres bombas en paralelo cada una con una capacidad del 50%. Así

se puede hacer trabajar una o dos bombas según el caudal requerido, y tener otra en previsión de averías y para mantenimiento. [20]

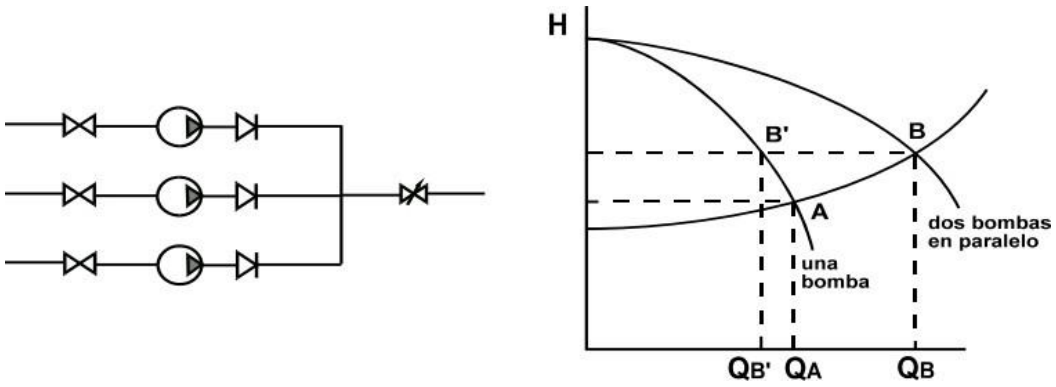


Figura14. Dos bombas en paralelo y una de seguridad. [20]

De esta forma se aumenta mucho la seguridad sin elevar demasiado los costes de instalación. (Otra opción es cuatro bombas, cada una con capacidad del 33%). [20]

5. Equipos e instrumentación utilizada.

En la realización de este proyecto se ha utilizado diversos equipos de automatización que se encuentran disponibles en el laboratorio de Automatización 2 en el bloque E de la Universidad Politécnica Salesiana sede Guayaquil. Tal como se muestra en la figura15, estos equipos se encuentran montados en láminas y forman parte de un tablero modular y reconfigurable para implementación de sistemas de automatización y redes industriales. A continuación, se mencionan algunos de los equipos utilizados en el proyecto:

- CPU S7-1500 + Módulos de entradas y salidas de tipo digital-analógico.
- Conmutador Industrial de 5 puertos.
- Variadores de frecuencia V20 para control analógico y G120 con comunicación Profinet.
- Variador de frecuencia ABB con modelo ACS355 con comunicación Profibus DP.
- Motores de inducción jaula de ardilla.

- Fuentes de corriente directa de 24V y de 10V para conexión de variables analógicas.
- Pulsadores, luces piloto y potenciómetros para simulación de sensor de flujo y sensores de nivel.

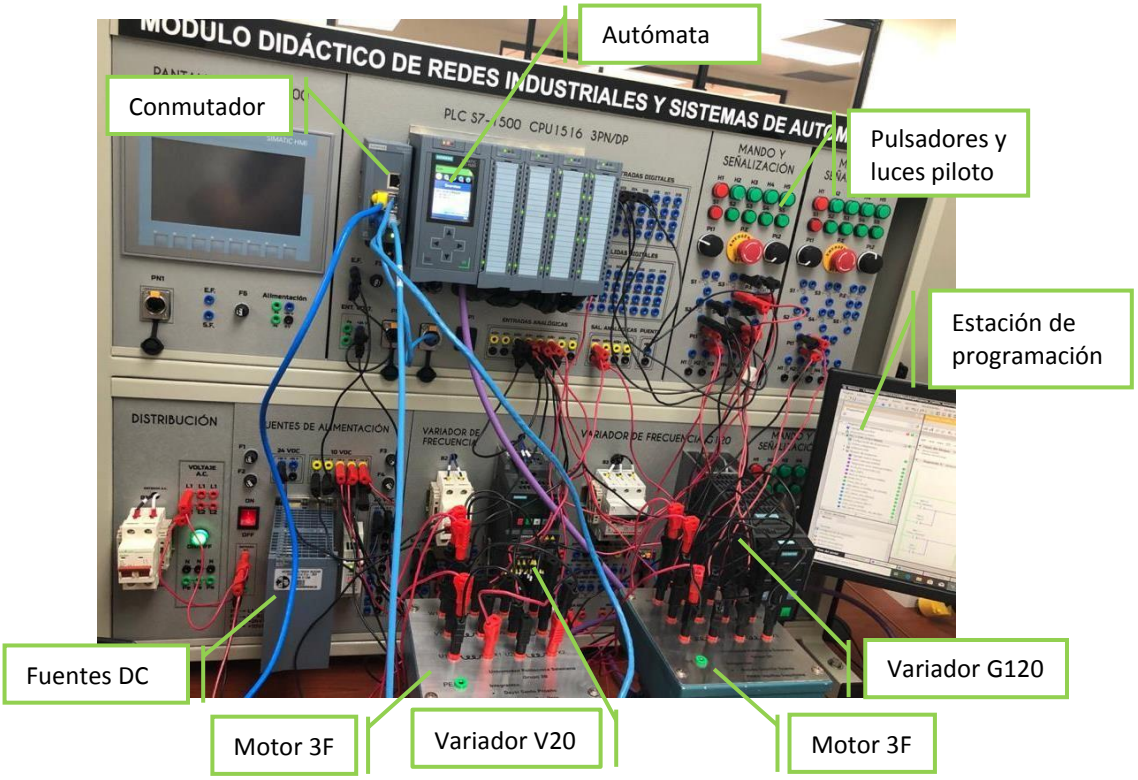


Figura15. Configuración de equipos utilizados.

3. MARCO METODOLÓGICO

1. Arquitectura de red

Este proyecto utiliza varios protocolos de comunicación para los diferentes dispositivos de automatización. Se ha considerado una red PROFIBUS DP entre el PLC S7-1500 y el variador de velocidad ACS355.También se cuenta con una red Ethernet Industrial entre el autómata y la interfaz humano-máquina ubicado en el PC utilizando el software LabVIEW. Finalmente existe una comunicación PROFINET IO entre el PLC y el variador de frecuencia SINAMICS G120. En la figura16 se puede visualizar la interconexión entre los equipos y dispositivos utilizados.

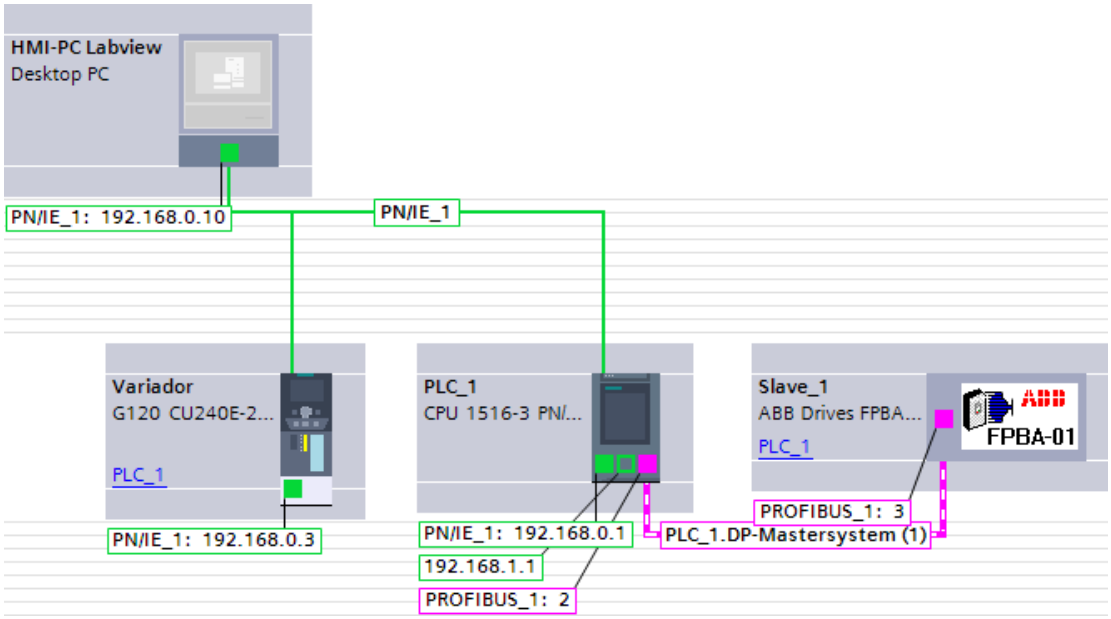


Figura16. Arquitectura de red

La topología de red Ethernet Industrial empleada es una estrella teniendo como nodo central el conmutador SCALANCE de 5 puertos, tal como se muestra en la figura 17. Adicional a ello se cuenta con acceso a la red general de la institución para interactuar con la plataforma Ubidots en Internet.

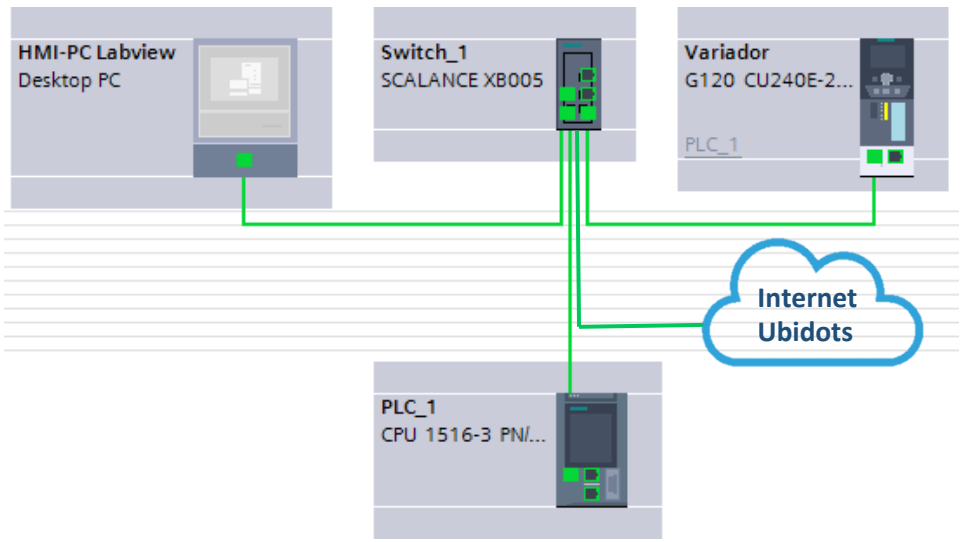


Figura17. Vista topológica de la red

2. Programación del controlador lógico programable

La programación del PLC ha sido realizada en lenguaje Ladder (KOP) teniendo en cuenta una distribución modular haciendo uso de bloques de función para control de variadores En la figura 18, se muestra un esquema de los bloques de programación utilizados.

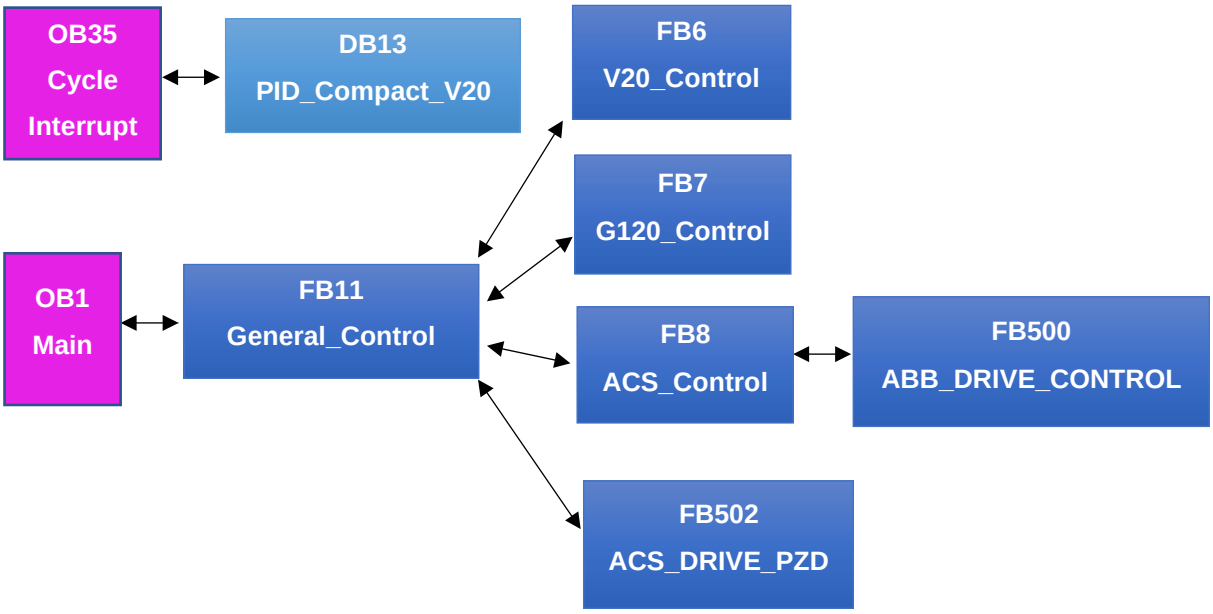


Figura18. Estructura de programación en la CPU

Como se puede apreciar en la figura previa, se hace uso de diferentes bloques que personalizados para control de variadores de velocidad. Cabe mencionar que se hace uso de bloques prediseñados para comunicación con el variador ABB, los cuales se encuentran disponibles en la página oficial de dicha marca. Dado que este variador de frecuencia hace uso del bus de campo PROFIBUS, se hace indispensable la adición de los archivos GSD, los cuales también se descargan de la página oficial de ABB [21]. En la figura 19 se puede apreciar los archivos instalados para la implementación del proyecto.

Figura19. Archivos GSD utilizados.

Para el motor G120 no fue necesario utilizar librerías adicionales ya que al tratarse de una configuración PROFINET IO se hizo uso de la aplicación StartDrive para la parametrización del mismo. En este punto resulta muy importante indicar la importancia de un dispositivo para establecer una comunicación correcta.

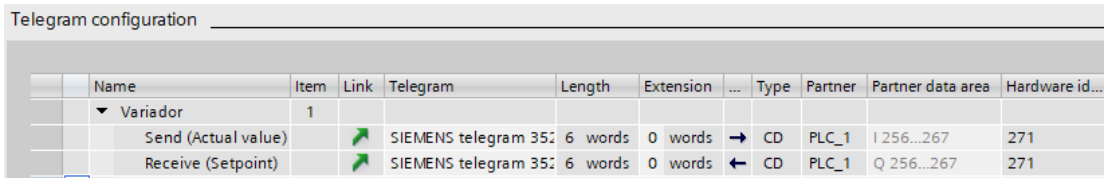


Figura20. Configuración de Telegrama para variador G120.

Para más información sobre la programación del PLC s7-1500, puede consultar al anexo2.

3. Configuración de servidor OPC

En esta sección se describe los pasos realizados para configuración del servidor OPC, utilizando el componente NI OPC Servers de *National Instruments*. En primer lugar, se agrega un nuevo canal como se indica en la figura 21.

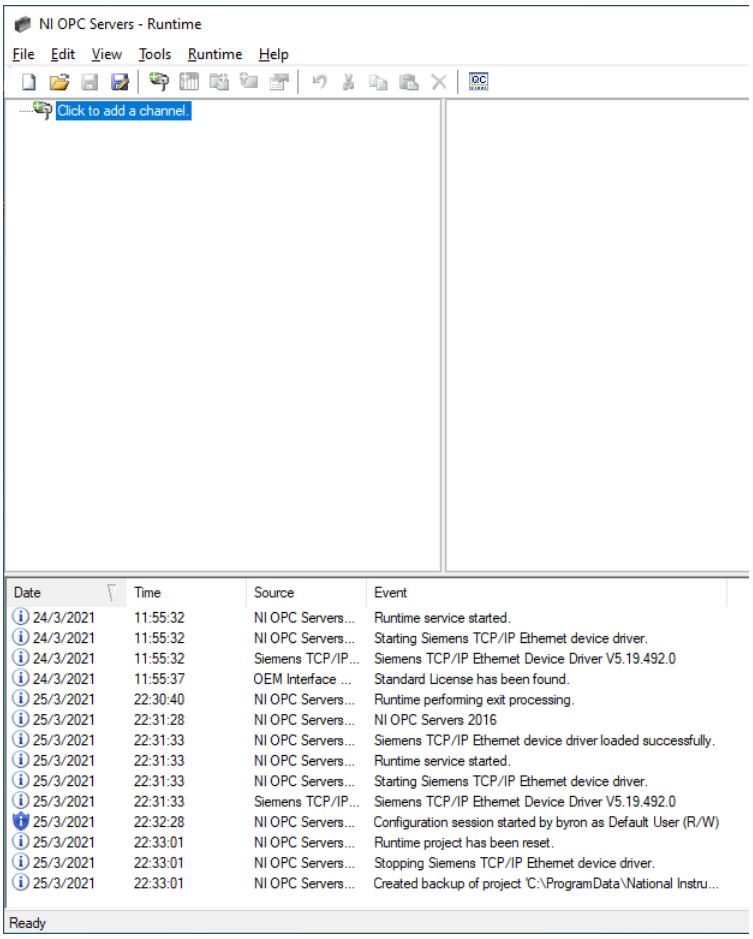


Figura21. Agregar nuevo canal en OPC Server

Como siguiente paso se asigna un nombre al canal, en este caso "*Proyecto_CedSam*" como se muestra en la figura 22. Adicional a ello se procede a especificar el controlador para conexión con el dispositivo a utilizar, en este caso se especifica "Siemens TCP/IP Ethernet". Esto resulta conveniente ya que el PLC utilizado cuenta con 2 puertos Ethernet independientes y eso facilita el monitoreo remoto.

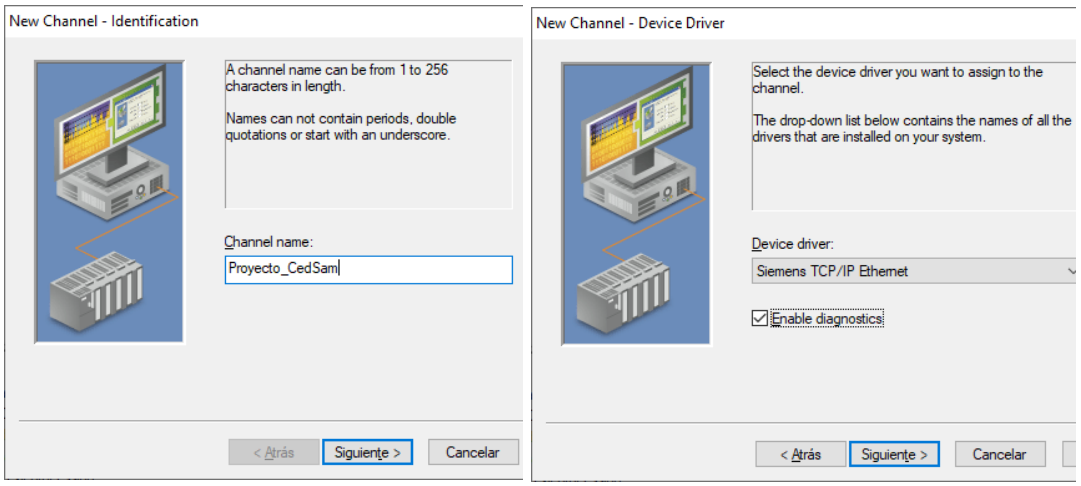


Figura22. Asignación de nombre y controlador para OPC.

En la figura23 podemos apreciar la configuración del hardware a utilizar en la comunicación con el autómata, para este caso se considera la tarjeta de red Ethernet disponible en el PC. Debe tener en cuenta la dirección IP para la lograr una conexión exitosa. Adicional a ello se configura el ciclo de trabajo para la optimización de escritura de variables.

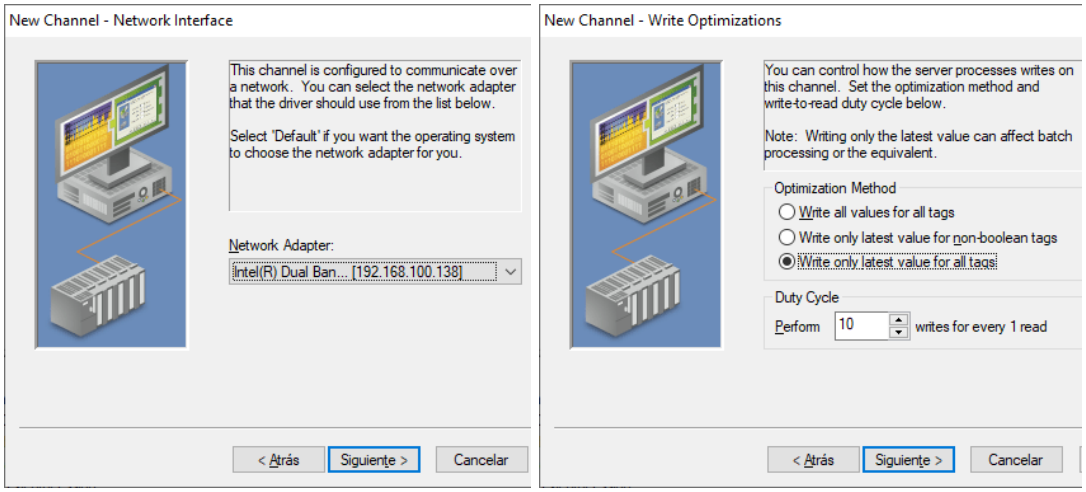


Figura23. Asignación de interfaz y opciones de escritura.

Finalmente se muestra un resumen de lo configurado en el canal para corroborar que los pasos previos han sido realizado correctamente, esto se puede evidenciar en las figuras 24 y 25.

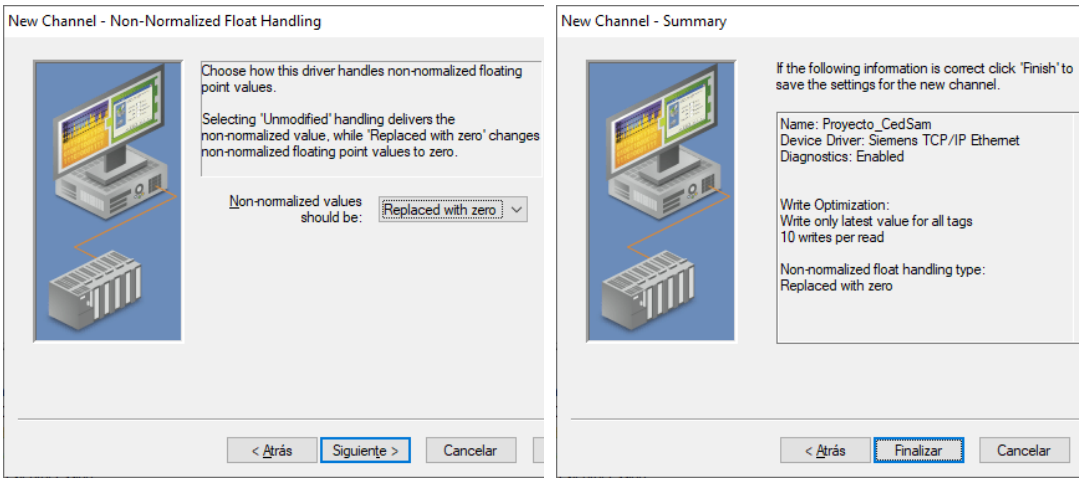


Figura24. Propiedades y resumen de parámetros servidor OPC.

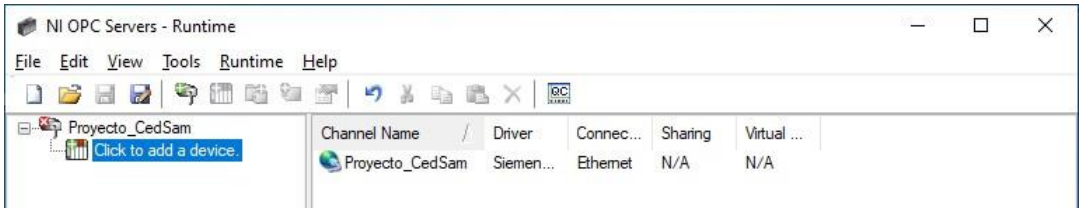


Figura25. Canal creado en el servidor OPC.

El siguiente paso consiste en definir y configurar un dispositivo dentro del canal creado previamente; en esta sección se puede revisar a detalle los parámetros de conexión entre el autómata y la estación de monitoreo (LabVIEW). Las figuras 26 y 27 muestra la adición de un controlador de tipo S7-1500 para esta aplicación, así como la configuración de dirección IP y modo lectura de variables.

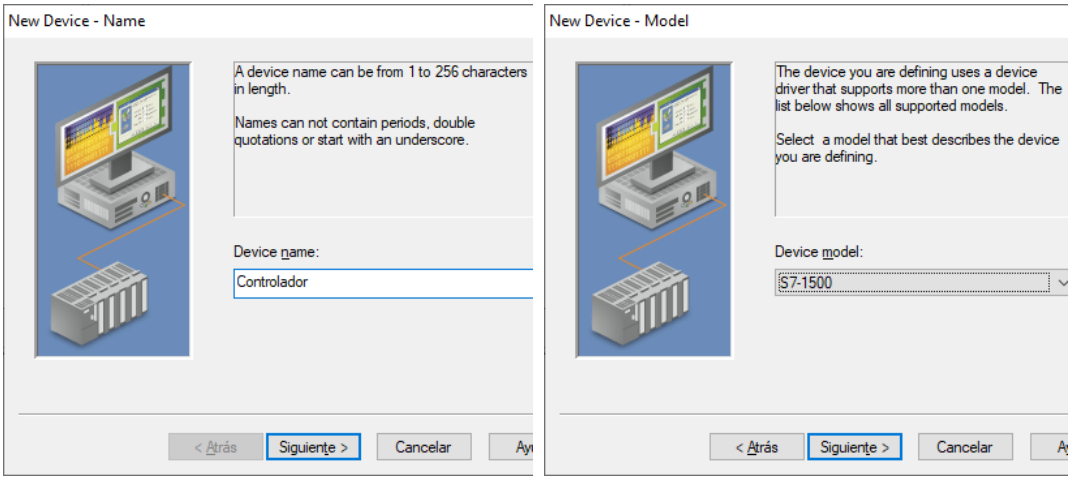


Figura26. Configuración de nuevo dispositivo.

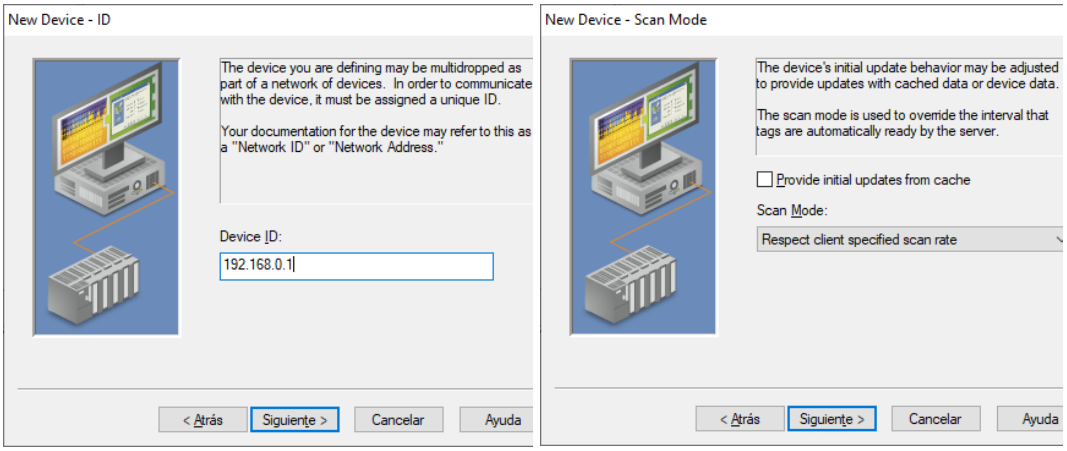


Figura27. Direccionamiento IP y configuración de modo para lectura.

En la figura28 se puede apreciar los parámetros de comunicación utilizados y la propiedad "*auto-demotion*" la cual consiste en poder paralizar temporalmente los requerimientos a un dispositivo que no responde a dichas peticiones, para así poder optimizar el tiempo de ciclo de peticiones de todo el canal.

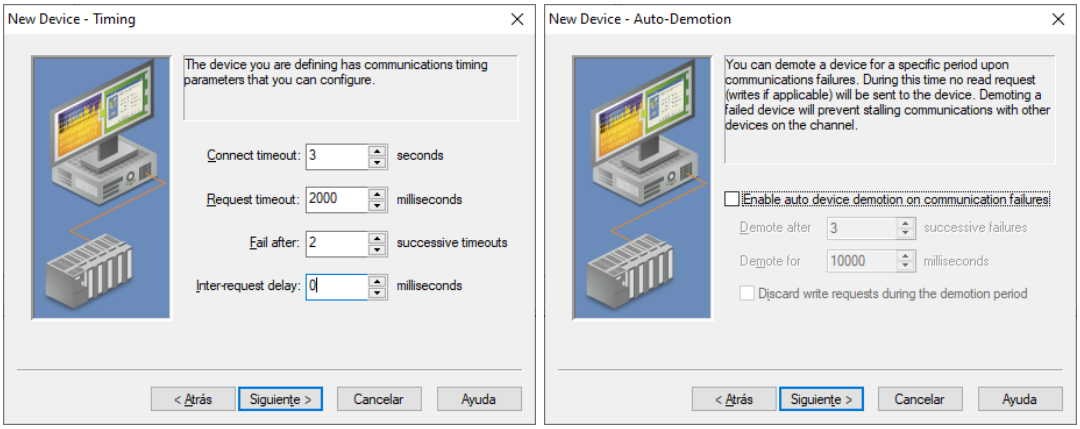


Figura28. Configuración de temporización y "*auto-demotion*".

A continuación, se configura el número de puerto TCP/IP del dispositivo a utilizar. Para ello se ha considerado el puerto 102 que viene definido por defecto, tal como se muestra en figura 29.

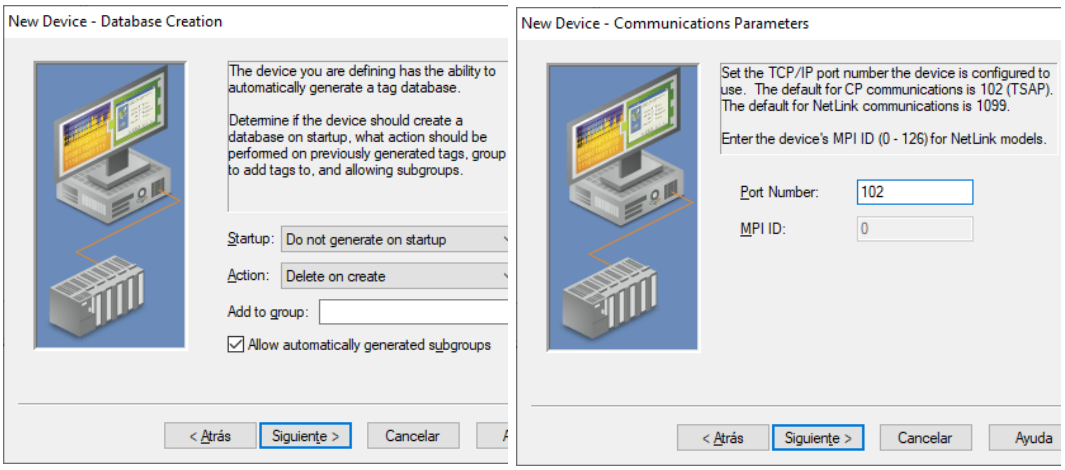


Figura29. Configuración de base de datos y puerto.

En la figura 30 se puede apreciar la configuración de los parámetros de comunicación S7 correspondiente a los diferentes controladores de Siemens. Adicional a ello se puede notar las opciones de direccionamiento disponibles, de las cuales se ha escogido Big Endian que se encuentra definida por defecto. Cabe mencionar que Big y Little Endian hacen referencia al orden de lectura de variables que tienen más de un byte. Por ejemplo, en una computadora big-endian, los dos bytes requeridos para el número hexadecimal 4F52 se almacenarían como 4F52 en el almacenamiento (si 4F se almacena en la dirección de almacenamiento 1000, por ejemplo, 52 estará en la dirección 1001). En un sistema little-endian, se almacenaría como 524F (52 en la dirección 1000, 4F en 1001). [21]

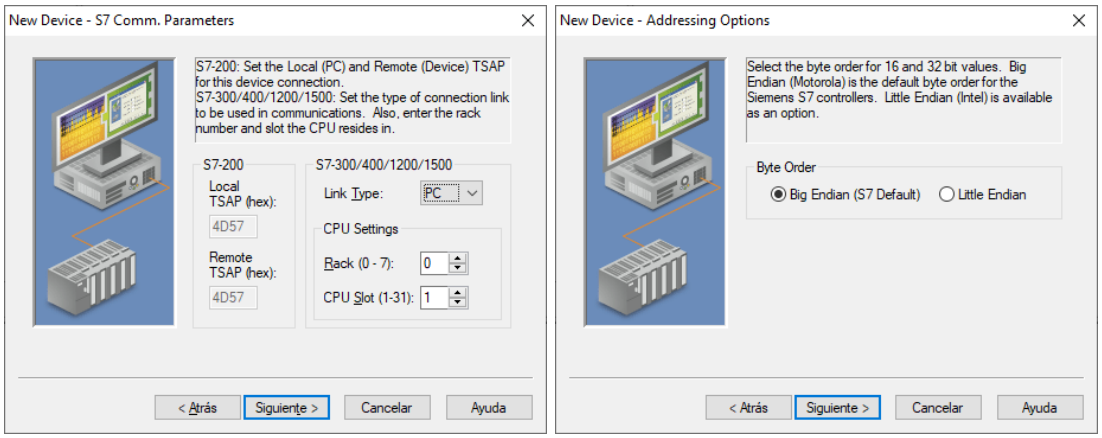


Figura30. Parámetros de comunicación S7

Tag Name	Address	Data Type	Scan Rate	Scaling	Description
ACS_ActualSpeed	MD104	Float	100	None	Velocidad actual de variador ACS355
ACS_CommOk	M1.7	Boolean	100	None	Indicador de comunicacion de variador ACS355
ACS_Error	M68.6	Boolean	100	None	Indicador de error de varidor ACS355
ACS_Intensity	MD124	Float	100	None	Consumo de corriente del variador ACS355
ACS_Running	M68.7	Boolean	100	None	Indicador de encendido del variador ACS355
ACS_SP	MD168	Float	100	None	Punto de referencia para variador ACS
ACS_Torque	MD112	Float	100	None	Torque actual de variador ACS355
ACS_VccBus	MD116	Float	100	None	Voltaje DC intermendio de variador ACS355
Elapsed_Time	DB14.DBD86	DWord	100	None	Tiempo transcurrido de la secuencia en ms
Flow_Sensor	I0.2	Boolean	100	None	Sensor indicador de flujo en tubería principal
G120_ActualSpeed	MD128	Float	100	None	Velocidad actual de variador G120
G120_CommOk	M68.4	Boolean	100	None	Indicador de comunicación de variador G120
G120_InOperation	M68.5	Boolean	100	None	Indicador de encendido de variador G120
G120_Intensity	MD176	Float	100	None	Consumo de corriente del variador G120
G120_SP	MD164	Float	100	None	Punto de referencia para variador G120
G120_Torque	MD172	Float	100	None	Torque actual de variador G120
Level1	IW70	Word	100	None	Sensor de nivel tanque1
Level2	IW72	Word	100	None	Sensor de nivel tanque2
Level3	IW74	Word	100	None	Sensor de nivel tanque3
On	DB14.DBX76.3	Boolean	100	None	Indicador de secuencia en proceso
P1_Time	MD80	Float	100	None	Tiempo de encendido de bomba1 en mililsegundos
P2_Time	MD84	Float	100	None	Tiempo de encendido de bomba2 en mililsegundos
P3_Time	MD88	Float	100	None	Tiempo de encendido de bomba3 en mililsegundos
Reset_Ubidots	M1.3	Boolean	100	None	Señal de reinicio de secuencia desde la nube
Sensor_Sim	M144.1	Boolean	100	None	Simulación de sensor indicador de flujo
Start_Ubidots	M1.0	Boolean	100	None	Señal de inicio de secuencia desde la nube
Stop_Ubidots	M1.1	Boolean	100	None	Señal de parada de secuencia desde la nube
V20_Control	DB27.DBD20	Float	100	None	Señal analógica de control para variador V20
V20_Frecuency	DB27.DBD16	Float	100	None	Frecuencia aplicada al variador V20
V20_On	Q0.0	Boolean	100	None	Indicador de encendido de variador V20
V20_SP	MD160	Float	100	None	Punto de referencia para variador V20

Figura31. Tabla de variables en servidor OPC.

En la figura 31 se encuentra la tabla de variables utilizada en el proyecto considerando diversas áreas de memoria tal vez como: entradas, salidas y marcas. Estas variables hacen referencia a diferentes parámetros del proceso, especialmente los valores a monitorear en cada variador de frecuencia como velocidad, intensidad de corriente, torque, entre otros.

4. Implementación de variables en LabVIEW

Luego de crear un proyecto nuevo en LabVIEW se procede a crear un servidor de entradas-salidas (IO Server) con la finalidad de vincular el mismo con el servidor OPC, pero considerando el modo Cliente. Los pasos a seguir se muestran en las figuras 32 y 33. Como se mencionó previamente, se ha utilizado el NI OPC Server V15.

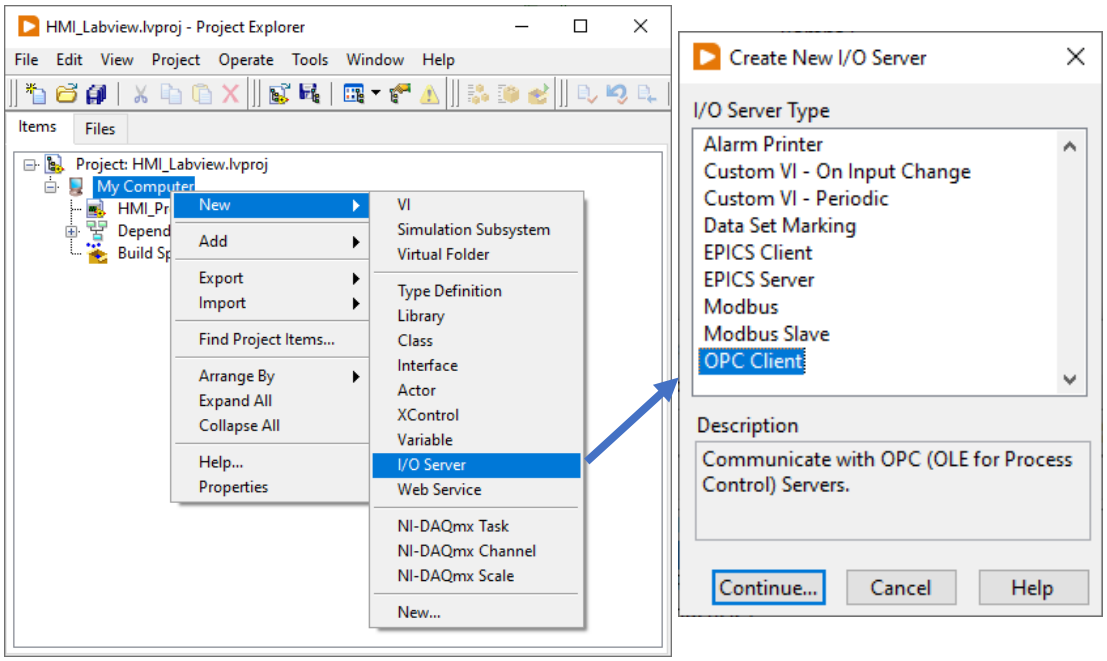


Figura32. Creación de I/O Server en LabVIEW.

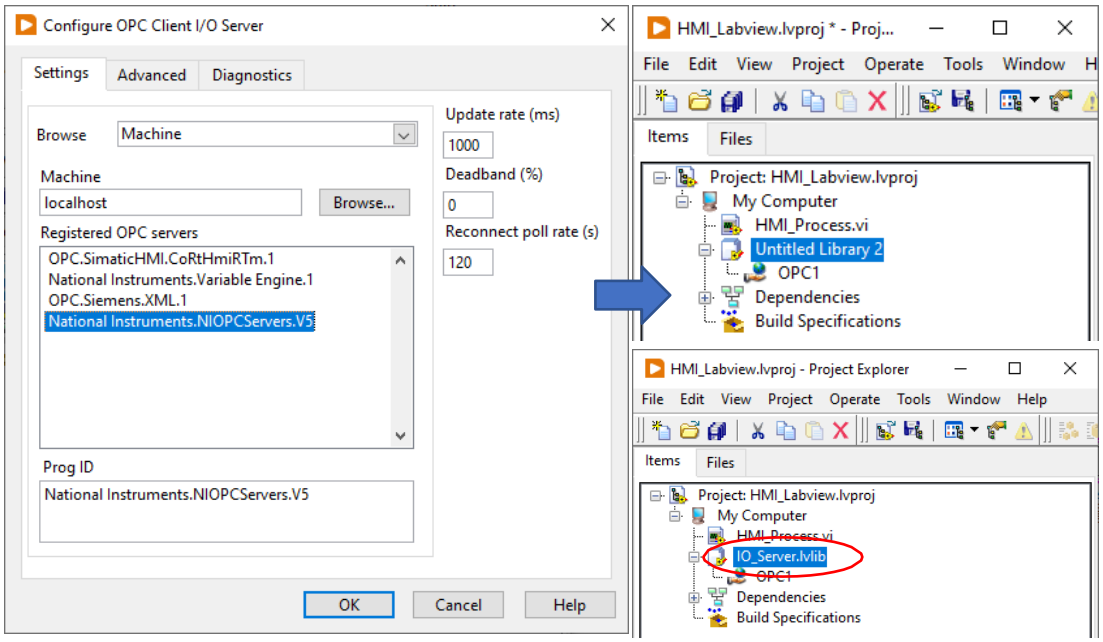


Figura33. Selección de servidor OPC para aplicación cliente.

Con el servidor IO, se procede a crear una librería que contendrá las variables vinculadas con el servidor OPC las cuales servirán para el monitoreo del proceso; en la figura 34 se muestra el proceso respectivo.

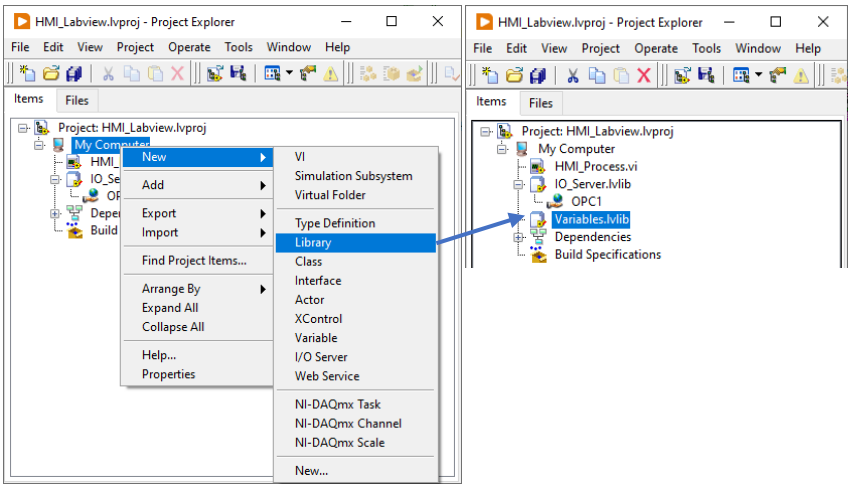


Figura34. Creación de librería para variables.

Como se puede apreciar en las figuras 35 y 36, se procede a la definición de variables vinculadas con el NI OPC.

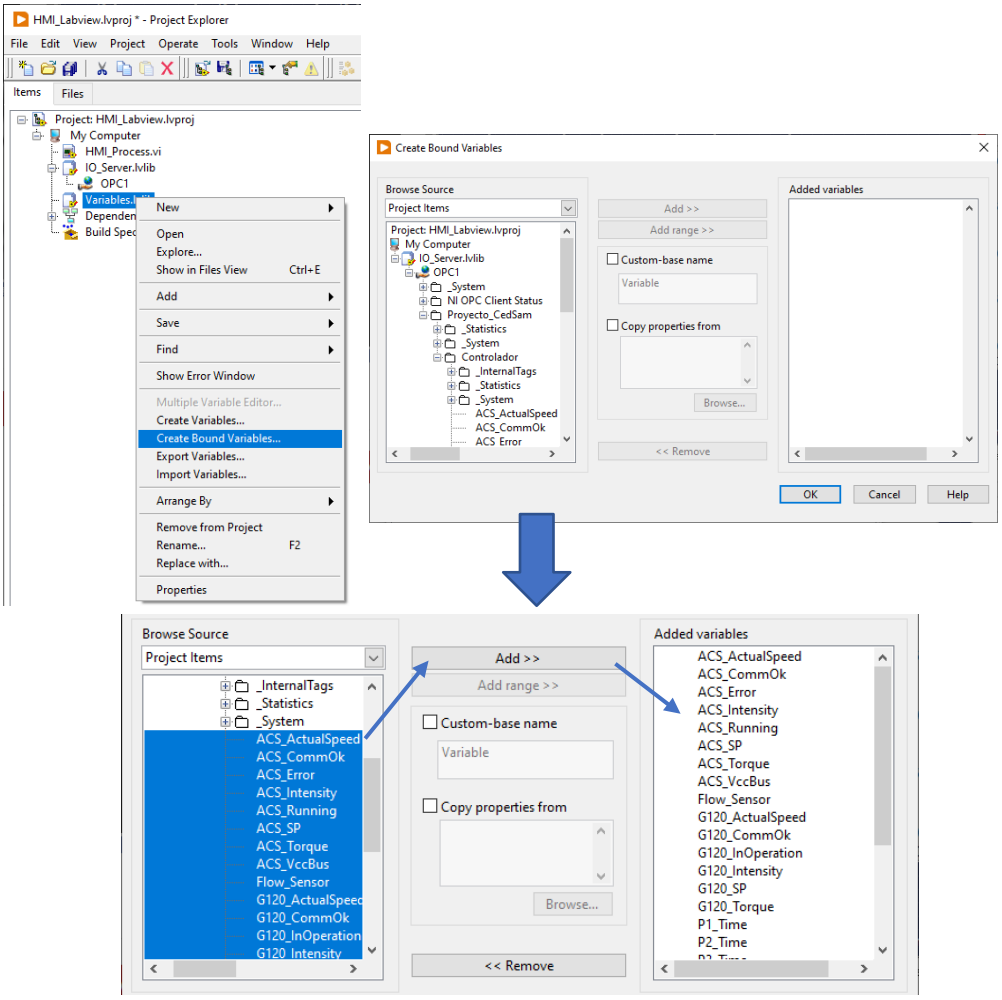


Figura35. Selección de variables compartidas OPC.

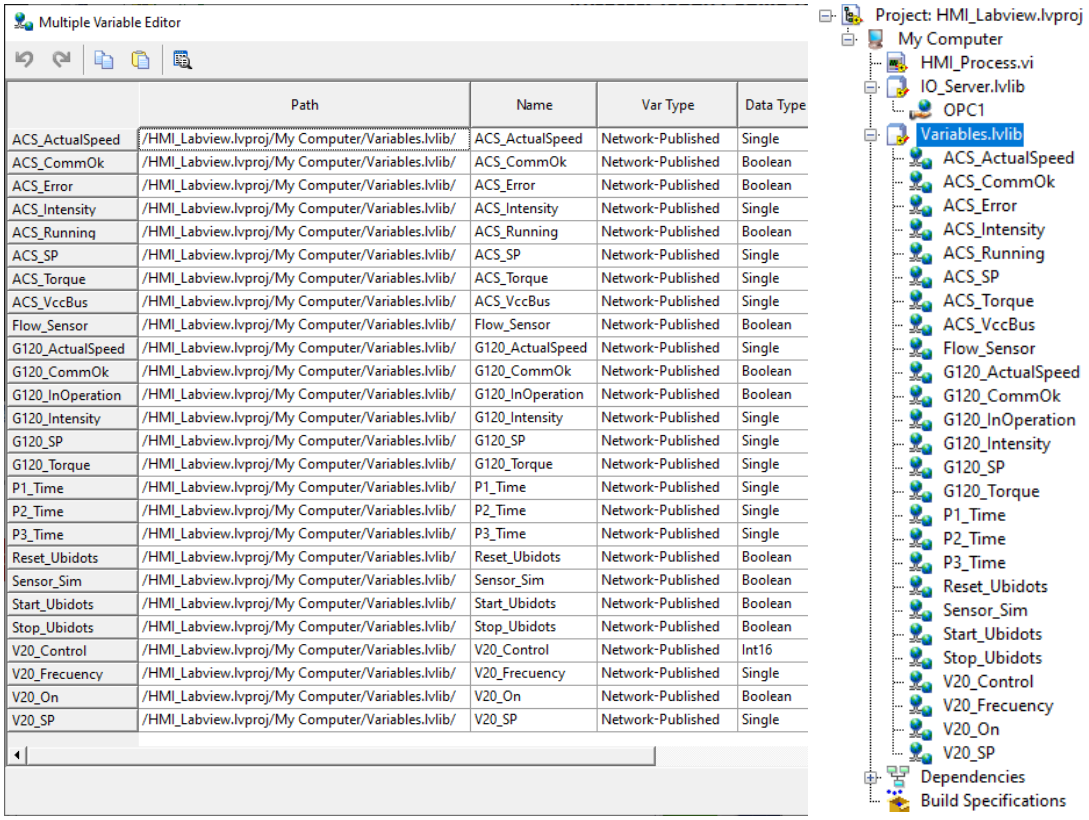


Figura36. Resumen de variables compartidas OPC.

5. Configuración de variables en Ubidots

Para la configuración de variables en Ubidots, se debe acceder a la página oficial de la plataforma: https://ubidots.com/ y hacer el registro respectivo como se muestra en la figura 37.

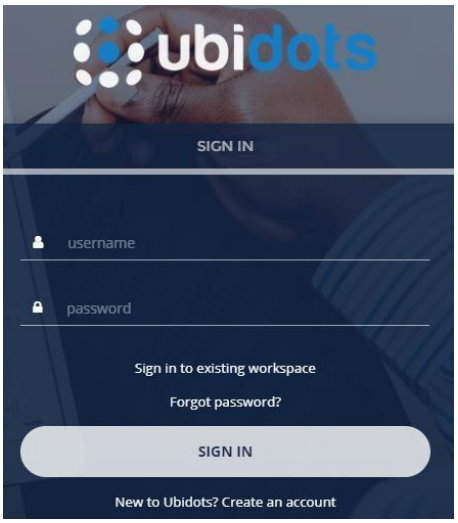


Figura37. Registro de usuario en Ubidots.

Posterior al registro de la cuenta, se agrega un nuevo dispositivo para comunicación con el software LabVIEW. Como se muestra en la figura 38, en este caso se escoge la opción "*Blank Device*" ya que en este proyecto no existe la necesidad de conectarse con algún dispositivo puntual. En caso de requerir dicha conexión, existen algunas librerías disponibles en la página oficial.

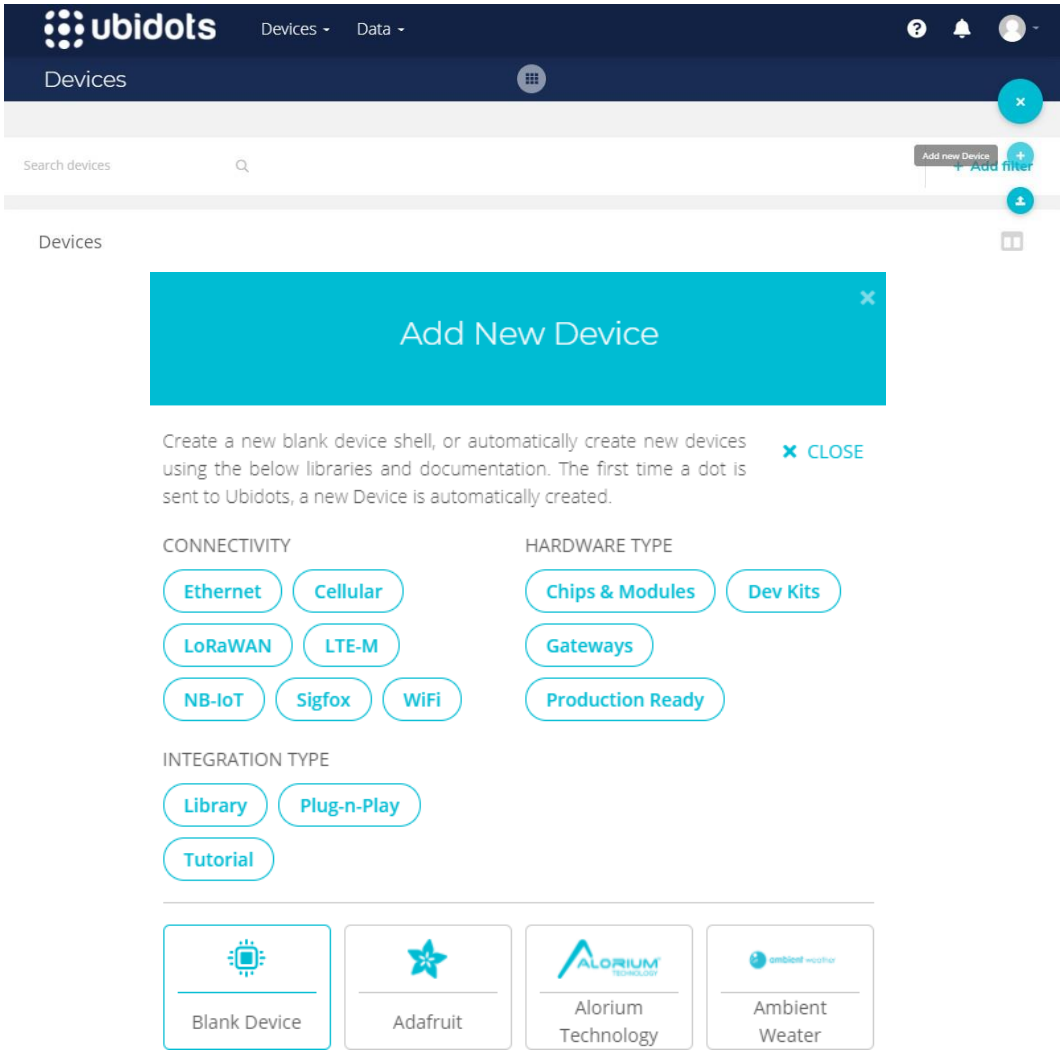


Figura38. Agregar nuevo dispositivo en Ubidots.

Se procede con la asignación del nombre y etiqueta del dispositivo a utilizar, tal como se muestra en la figura 39. El siguiente paso realizado fue agregar las variables para monitoreo e interacción con el proceso, especialmente las que corresponde a los variadores de velocidad. En la figura 40 se muestra la pantalla previa a la adición de variables.

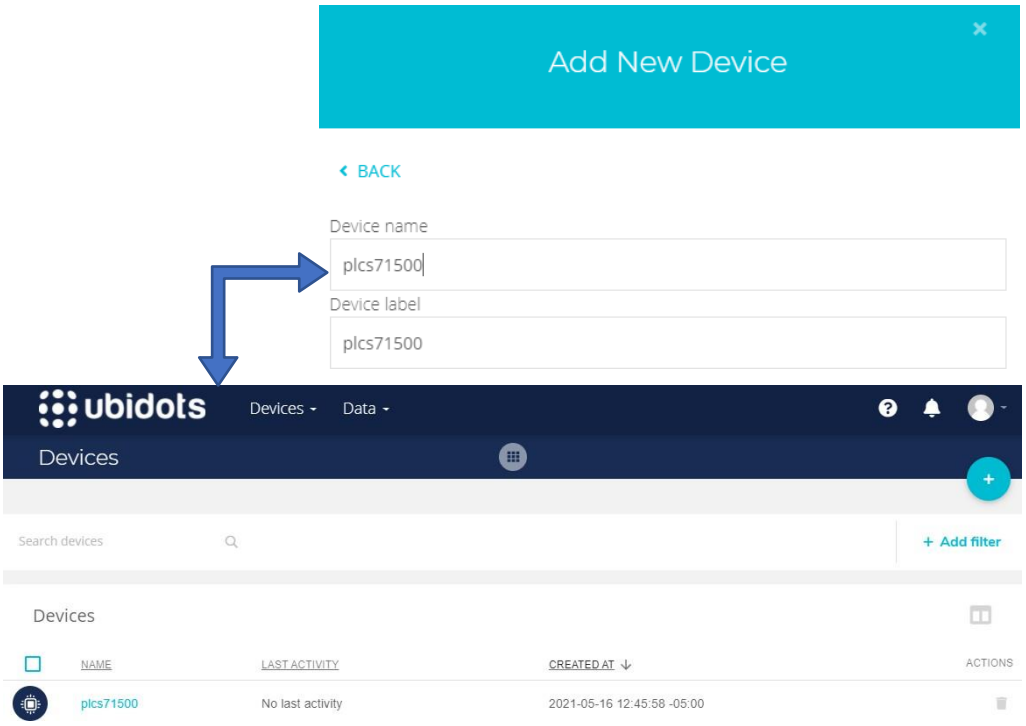


Figura39. Configuración de dispositivo para conexión.

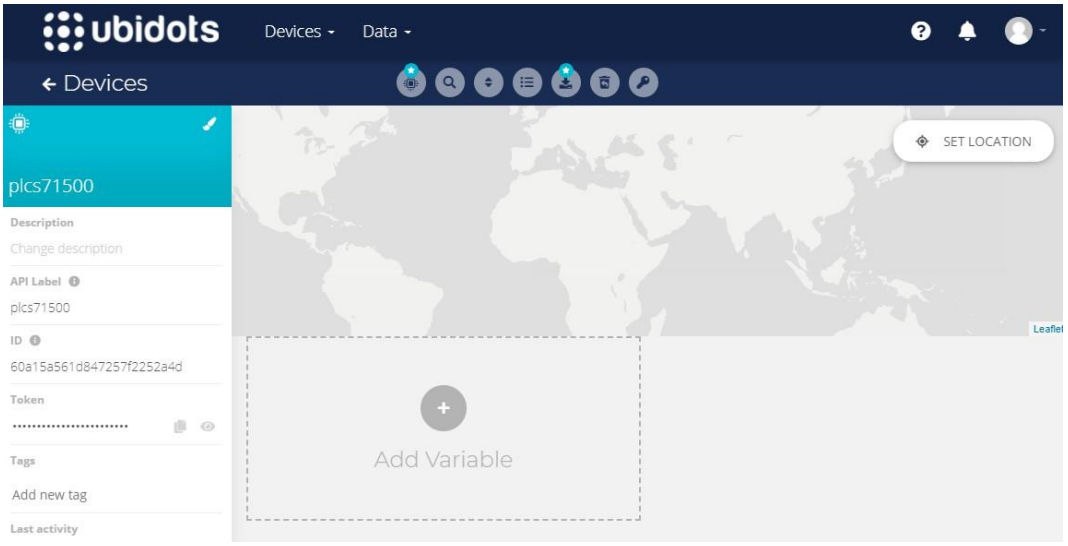


Figura40. Pantalla de adición de variables Ubidots.

Las variables utilizadas en este proyecto se muestran en las figuras 41, 42, 43 y 44, las cuales se encuentran agrupadas según el dispositivo a utilizar, en este caso se encuentran asociadas mayormente por cada variador de frecuencia: V20, G120 y ACS355.

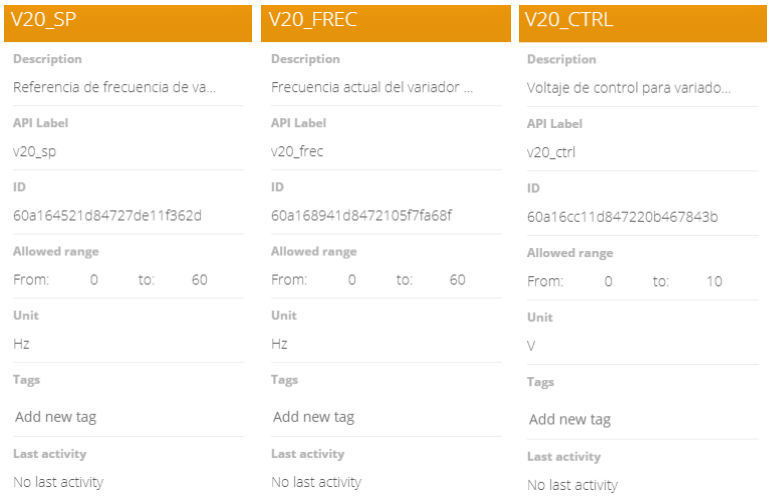


Figura41. Variables de controlador de motor V20 en Ubidots

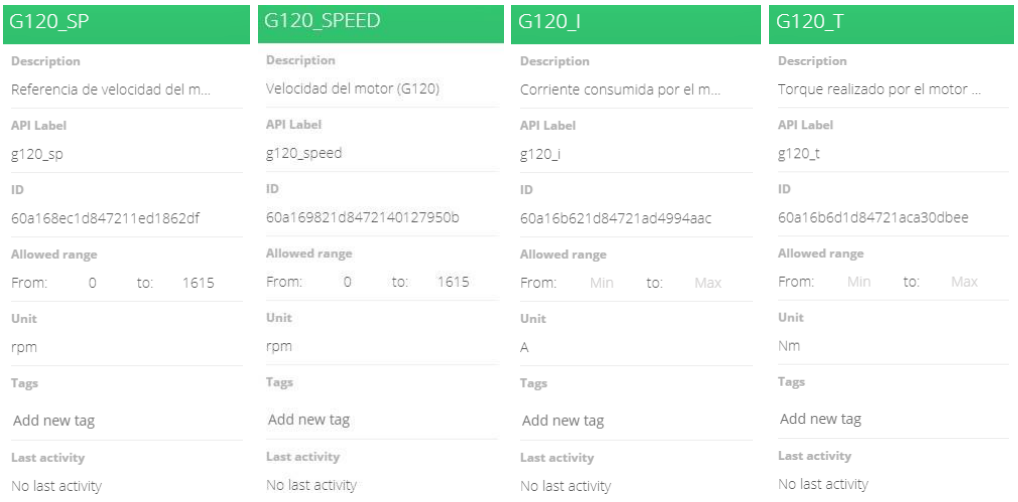


Figura42. Variables de controlador de motor G120 en Ubidots

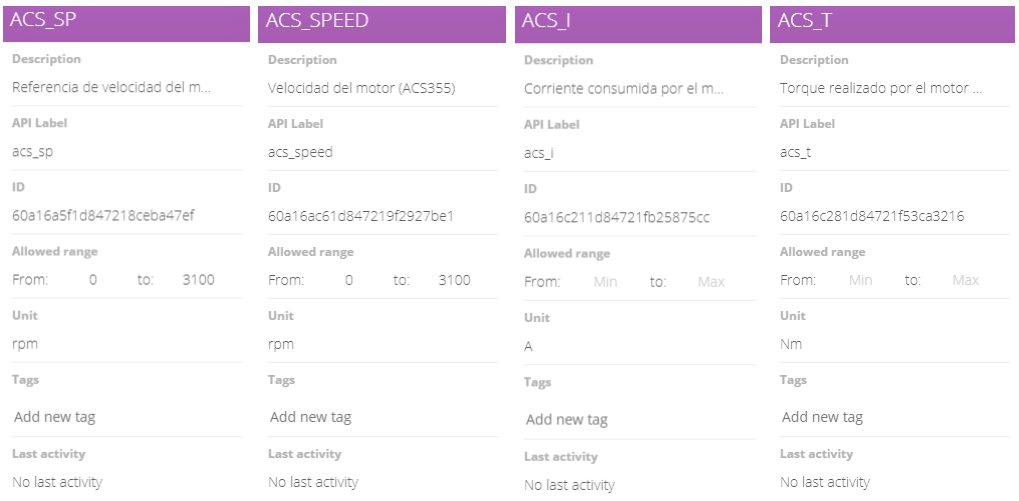


Figura43. Variables de controlador de motor ACS en Ubidots

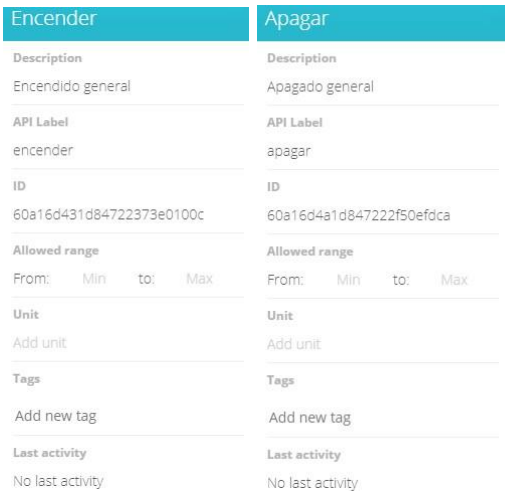


Figura44. Variables para control general en Ubidots

Hay que tener en cuenta la importancia del parámetro API label, ya que el mismo es utilizado en el software Labview para hacer la lectura y escritura de variables desde Ubidots. Adicional a ello, debe considerarse el rango de trabajo de las varaibles ya que se puede ajustar dicho parámetro en la plataforma.

6. Diseño de Dashboard en Ubidots

Hoy en día un dashborad o un panel de control resulta una herramienta muy potente para el monitoreo de procesos, más aún cuando no encontramos en el boom de la industria 4.0. En la figura 45 se puede apreciar el primer paso para iniciar con la configuración de un panel de control en la plataforma Ubidots. Mientras que en la figura 46 se muestra las propiedades configuradas para el dashboard; en esta sección hay detalles importantes como el tiempo para la visualización lo cual depende de la aplicación.

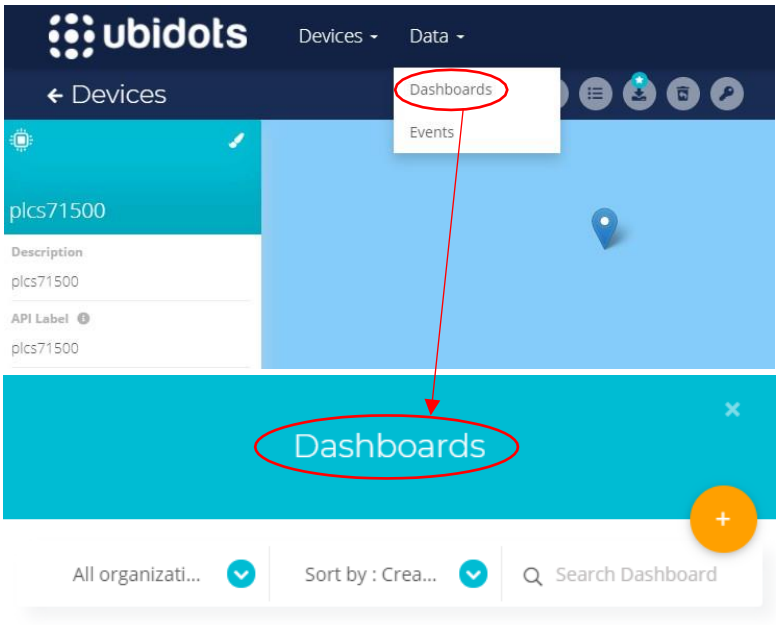


Figura45. Creación de nuevo *Dashboard*

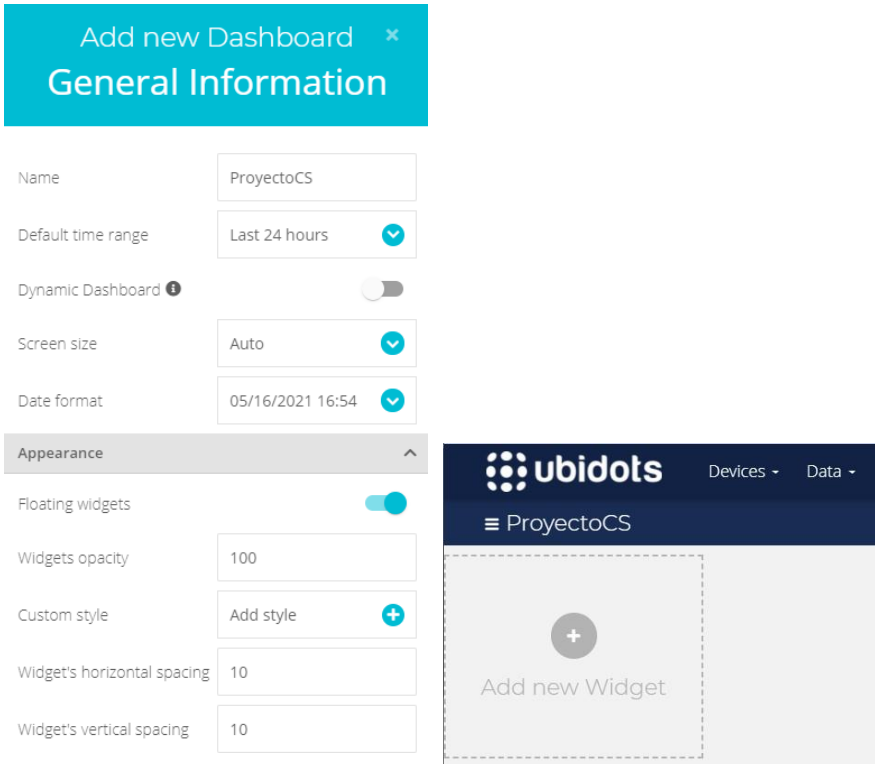


Figura46. Propiedades del *Dashboard*

Luego de agregar el panel de control el siguiente paso consiste en agregar los diferentes elementos para monitoreo e interacción con la plataforma (widget), lo cual se evidencia en la figura 47.

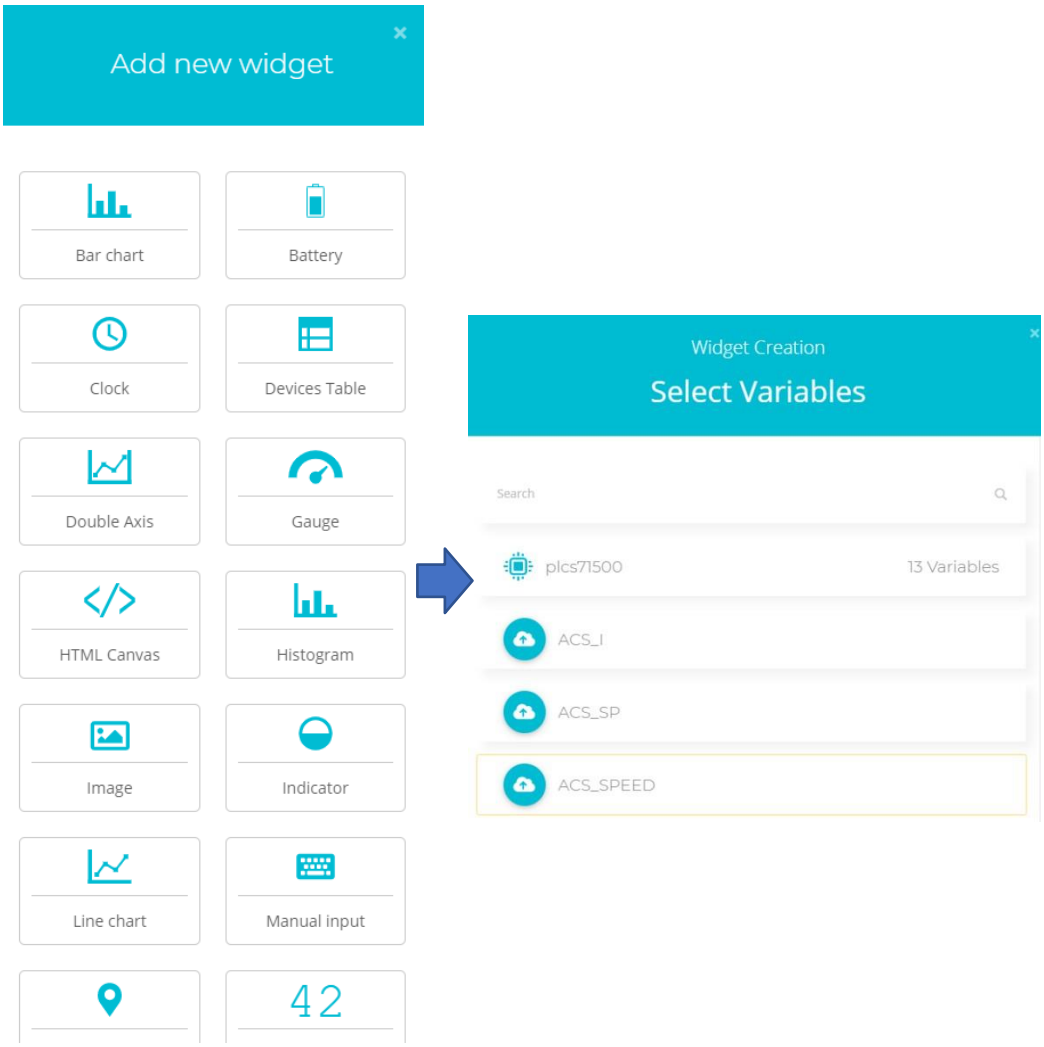


Figura47. Proceso de adición de widget

A continuación se muestran las propiedades los diferentes visualizadores utilizados. En las figuras de la 48 a la 52 se puede observar los parámetros utilizados en el graficador line chart, este elemento tiene como objetivo verificar el cambio de velocidades de los motorescontrolador por los drivers G120 y ACS355. Cabe mencionar que se incluyen las señales de referencia para saber la consigna de cada motor. Adicional a ello se consideran señales de corriente y torque para motores que tienen comunicación via Profibus y Profinet.

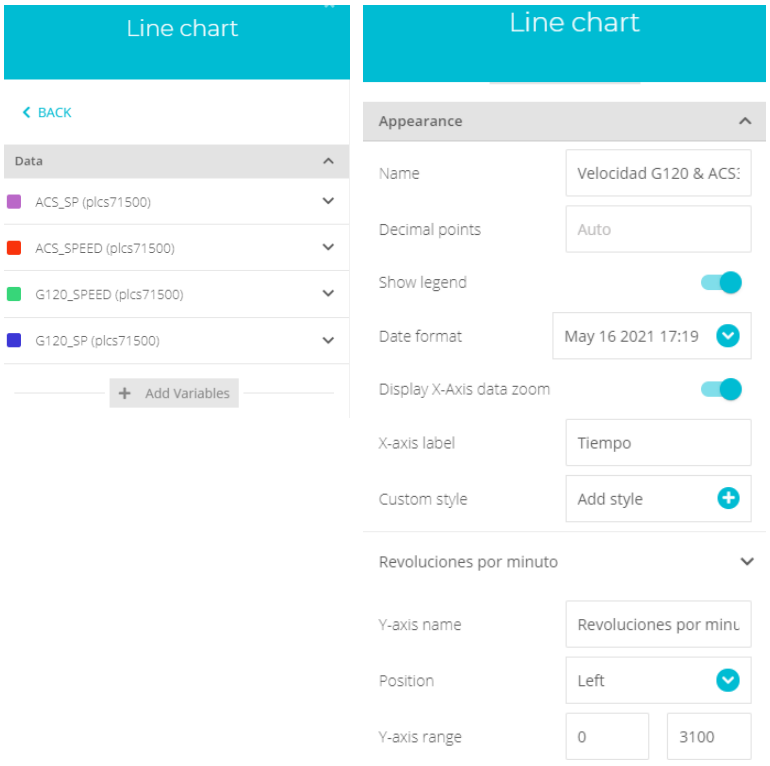


Figura48. Configuración de line chart1

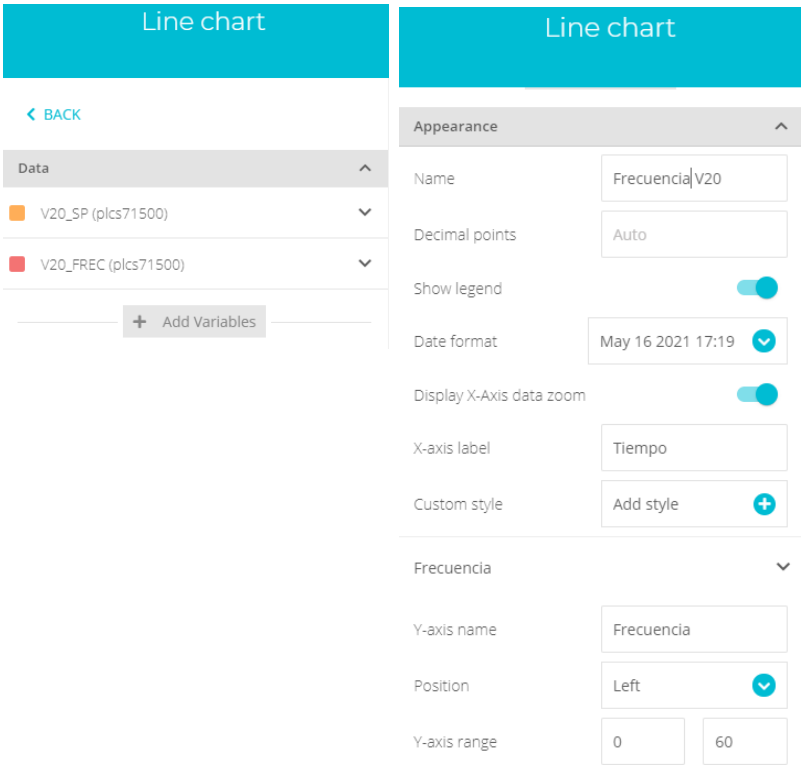


Figura49. Configuración de line chart2

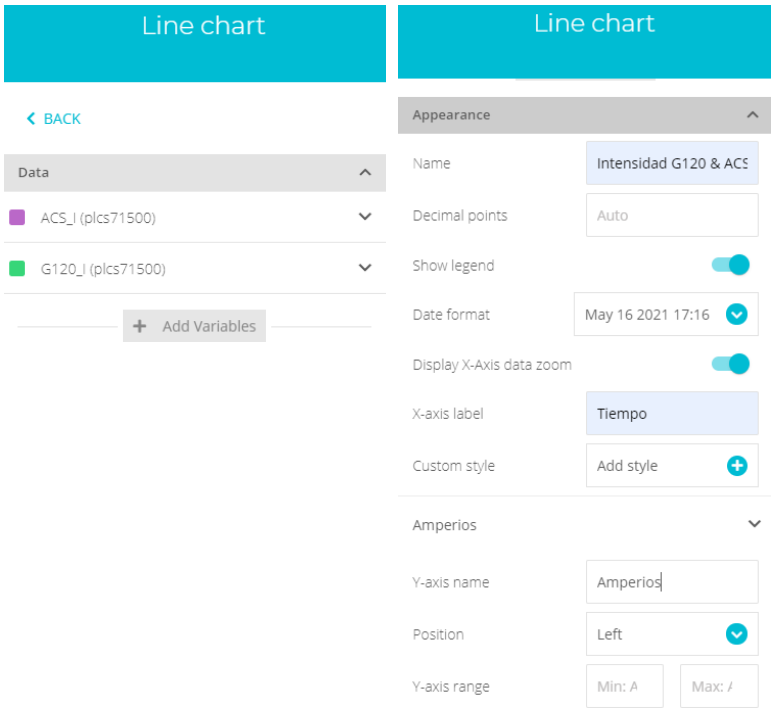


Figura50. Configuración de line chart3

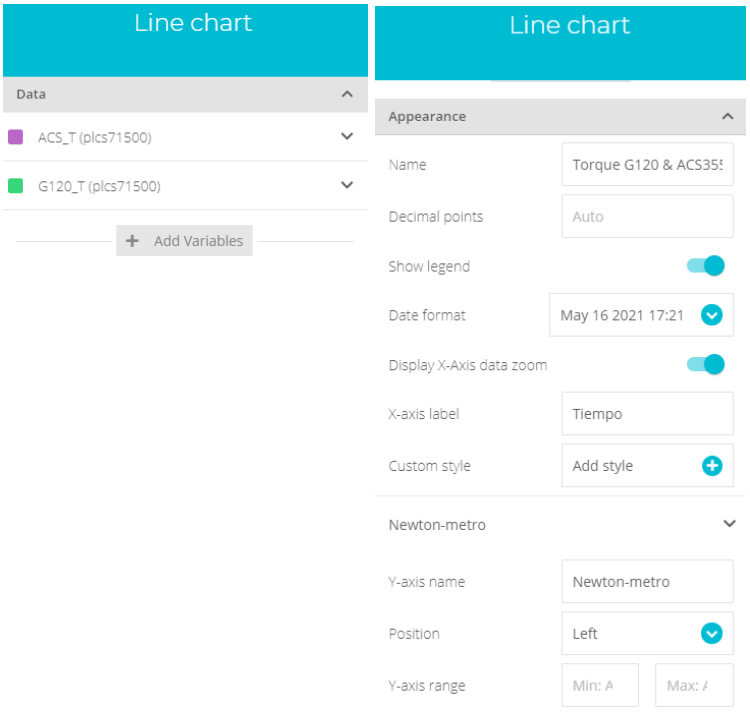


Figura51. Configuración de line chart4

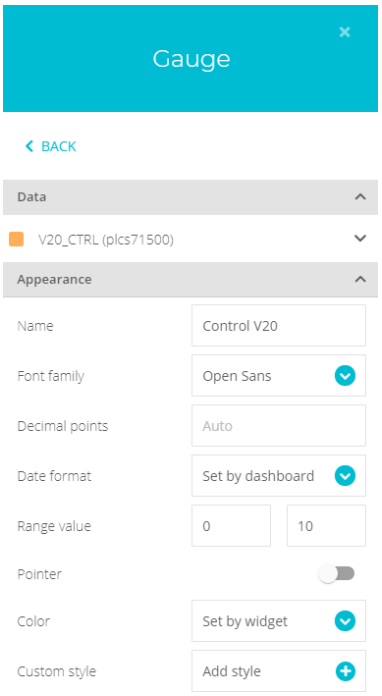


Figura52. Configuración de line chart5

7. Programación en LabVIEW

Para la programación en el PC, se ha realizado un proyecto en Labview en conjunto con algunos subVIs personalizados a fin de mejorar la presentación del programa principal y modularizar el esquema. En la figura 53 se puede apreciar el esquema propuesto para este proyecto.

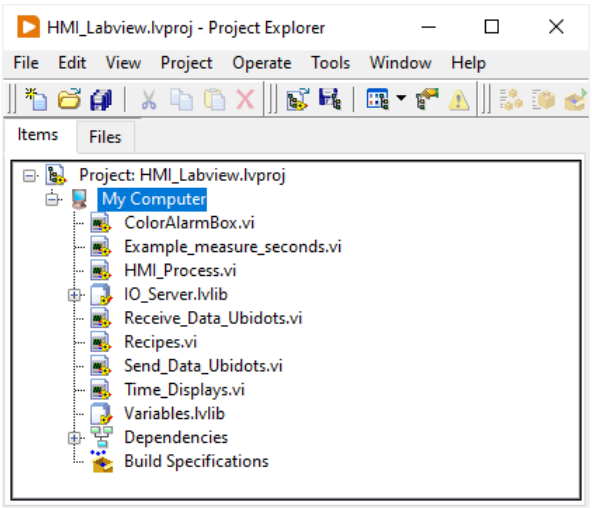


Figura53. Esquema de proyecto en LabVIEW

3.7.1. Diseño de panel frontal

El panel frontal consta de varias secciones para interacción con el usuario las cuales se detallan en la figura 54. En la imagen se pueden visualizar la existencia de múltiples pestañas para monitoreo de los parámetros eléctricos de cada motor etiquetadas con el nombre de cada variador de frecuencia.

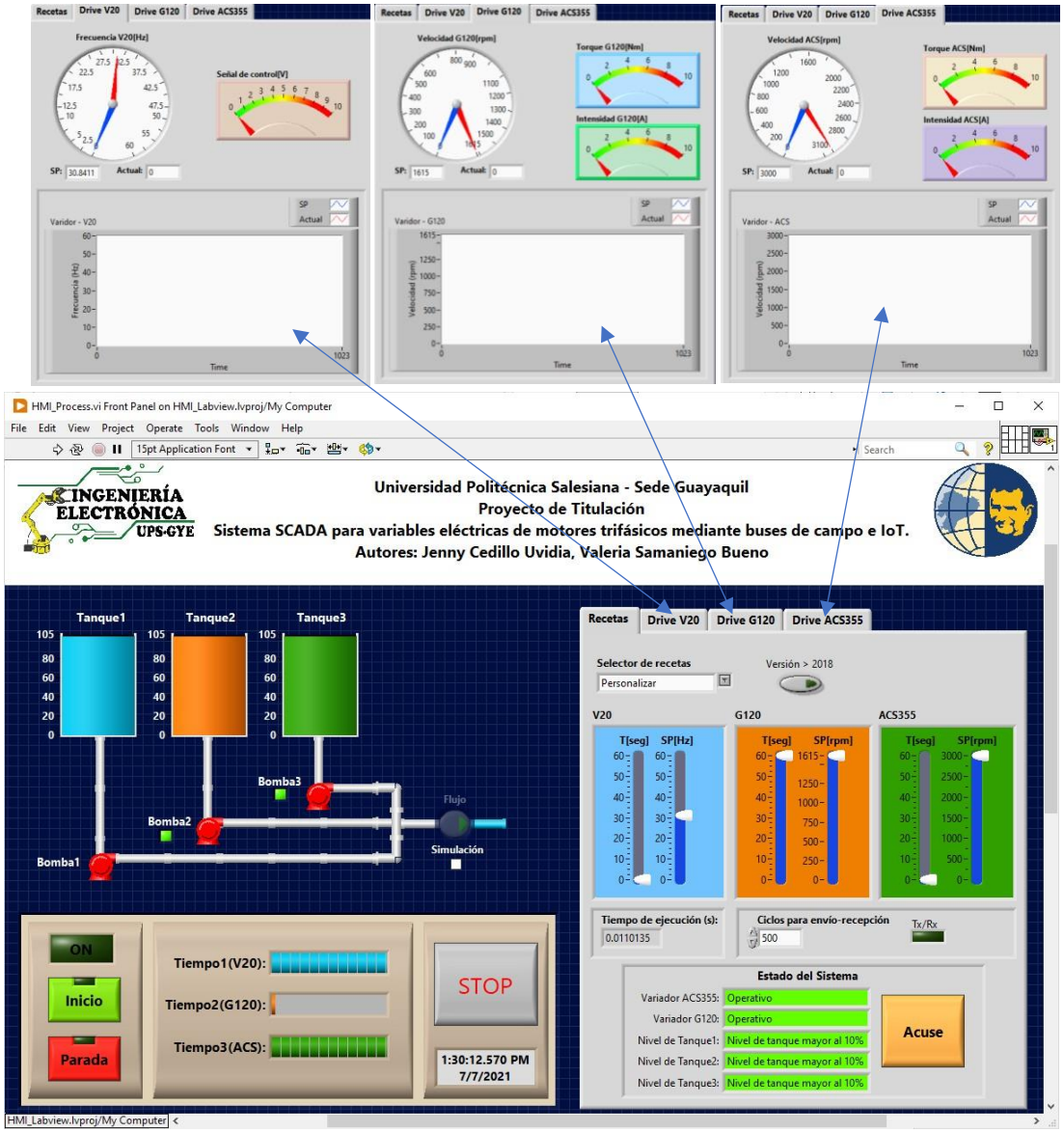


Figura54. Diseño de panel frontal

En el diseño propuesto se ha considerado una representación del proceso de bombeo con las respectivas animaciones: cambio de niveles en tanques, cambio de colores en tuberías según el circuito activo, indicadores de comunicación con variadores y simulación de sensor de flujo desde el PC. Cabe mencionar que esta

interfaz también cuenta con un panel de control para encendido, parada y monitoreo del tiempo de trabajo de cada bomba. En la pestaña Recetas, se puede configurar varias secuencias predefinidas para el trabajo de cada bomba, también se cuenta con una opción para personalizar la secuencia en la cual se puede variar el tiempo de trabajo y el valor de referencia de cada bomba. En la misma pestaña se puede monitorear algunas alarmas por posibles eventualidades contempladas para la simulación del proyecto, con el respectivo botón de acuse para reiniciar la misma.

3.7.2. Diagrama de bloques

3.7.3. SubVI Receive_Data_Ubidots

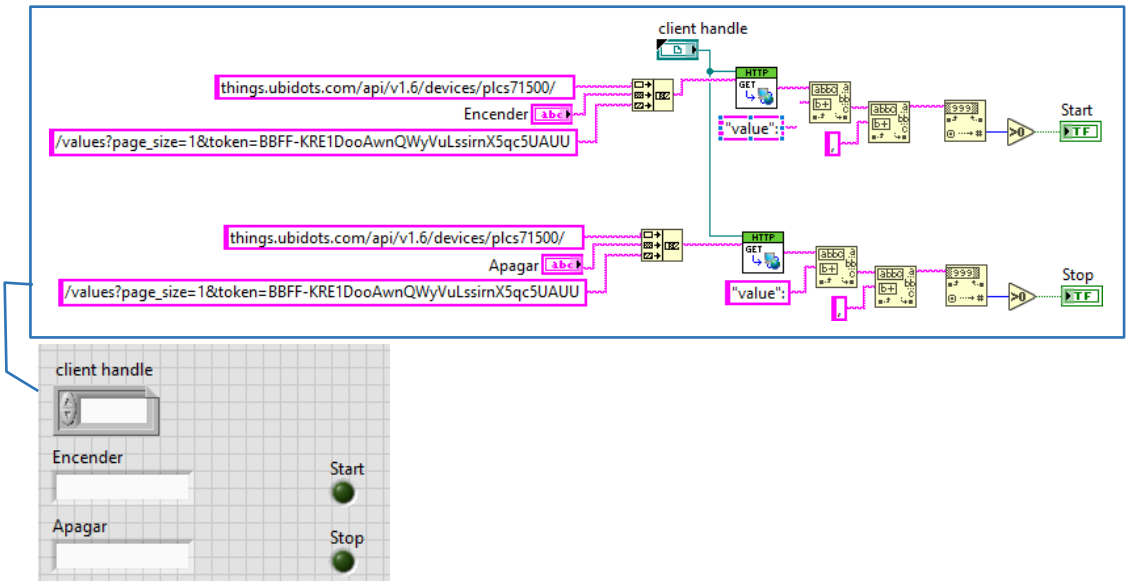


Figura55. SubVI para recepción de datos desde Ubidots

El subVI mostrado en la figura 55 muestra la conexión de bloques para la correcta recepción de datos desde la plataforma IOT (Ubidots). En este caso se debe tener en cuenta la sintaxis utilizada en la cual se incluye la dirección web de la página oficial Ubidots, el dispositivo configurado en la plataforma (plcs71500) y las etiquetas de las variables configuradas previamente.

3.7.4. SubVI Send_Data_Ubidots

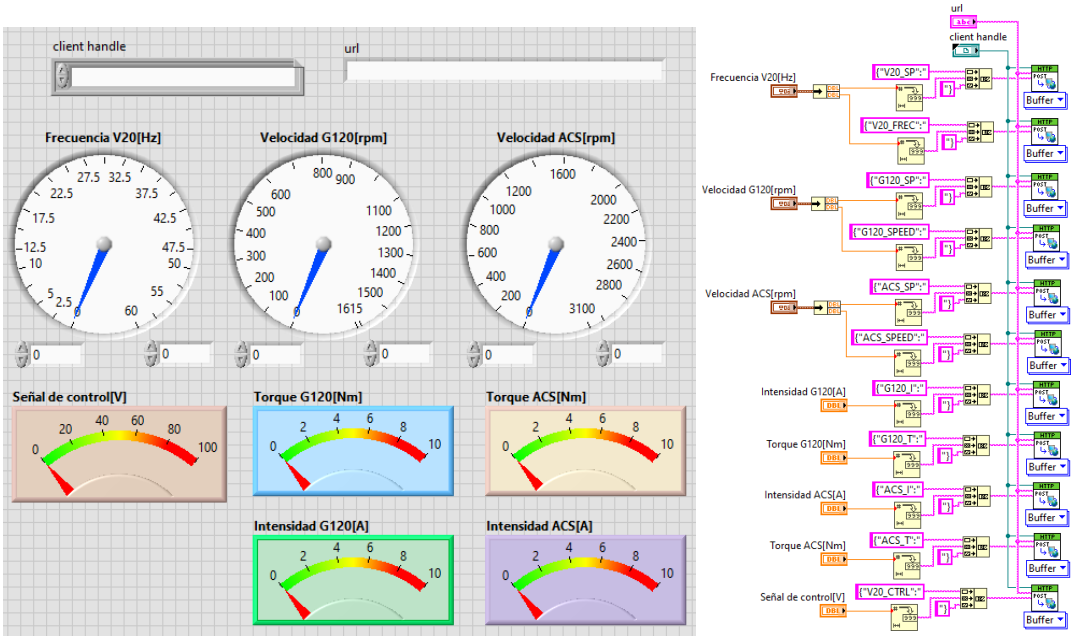


Figura56. SubVI para envío de datos hacia Ubidots

En la figura 56 se puede evidenciar la sintaxis utilizada para el envío de datos vía HTTP, nótese que se hace referencia a los nombres de las variables configuradas previamente como parte de las entradas al bloque PUT. Debe tener en cuenta que, al tratarse de nombres de variables, debe ser exactamente el mismo que el definido en la plataforma, incluyendo caracteres especiales.

3.7.5. SubVI Time_Displays

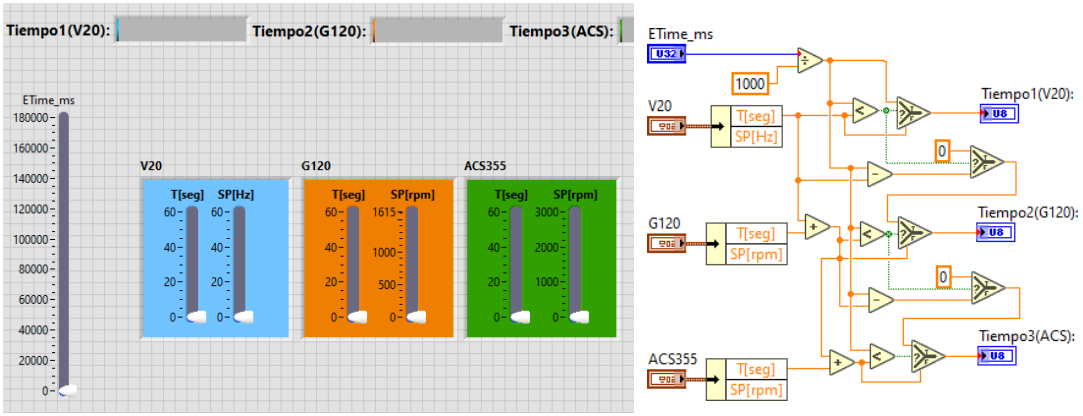


Figura57. SubVI para monitoreo de tiempos

El código mostrado en la figura 57 se utiliza para el monitoreo correcto para los tiempos de trabajo de cada bomba según la receta seleccionada previamente. Para tal efecto se toman los datos provenientes del PLC S71500 y a través de comparadores y operadores matemáticos se cumple objetivo.

3.7.6. SubVI ColorAlarmBox

El bloque mostrado en la figura 58 permite seleccionar un color para el fondo de los indicadores de texto en el monitoreo de alarmas. Dado que la selección depende de una señal booleana, se tienen dos estados para controlar el color y el fondo del indicador de texto, estado1: texto de color negro y fondo verde y estado2: testo de color amarillo y fondo rojo.

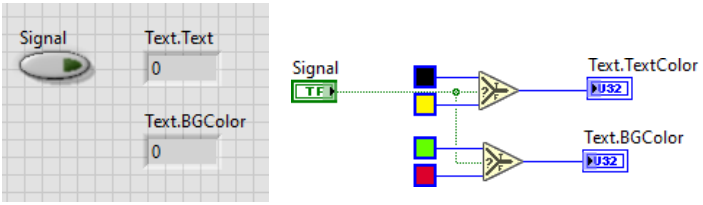


Figura58. SubVI para cambio de color

3.7.7. SubVI Recipes

El objetivo del subVI Recipes es ayudar con la selección de tiempos y valores de referencia para velocidad o frecuencia en cada uno de los motores a controlar. Se debe tener en cuenta que se han programado 4 secuencias, de las cuales 3 de estas son fijas y una de ellas puede personalizarse. Cuando se escoge la secuencia fija, se deshabilita la manipulación de los deslizadores para controlar la frecuencia y tiempo, lo cual no ocurre en la secuencia personalizada. Esta última opción se la ha incluido ya que en todo sistema pueda darse el caso de requerir cierto comportamiento para pruebas o mantenimientos.

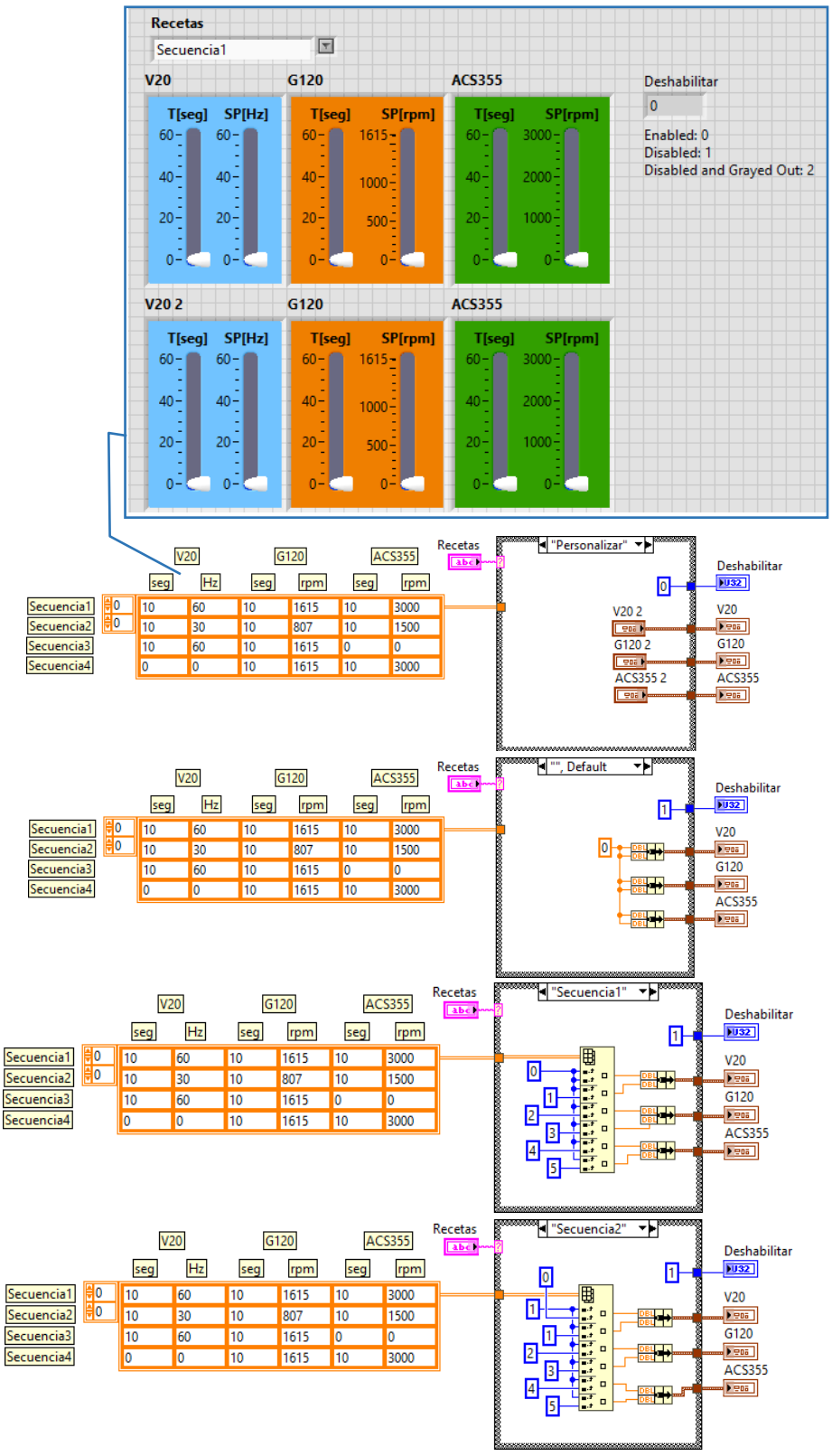


Figura59. SubVI para selector de recetas

3.7.8. Resumen de Guía

1. Desarrollar una arquitectura de red y programar el control lógico programable (PLC S71500). #Pag_33
2. Configurar las variables en el servidor OPC.#Pag_35
3. Vincular las variables del servidor OPC para implementarlas en LABVIEW. #Pag_36
4. Diseñar y configurar el Dashboard de las variables en Ubidots. #Pag_49
5. Ejecutar la programación en el software LABVIEW. #Pag_54
6. Realizar el diseño del HMI en el panel frontal. #Pag_55
7. Utilizar los SubVI para una mayor organización al momento de programar. #Pag_56
8. Realizar las conexiones de los motores con los variadores y el controlador lógico programable en el módulo. #Pag_70
9. Finalmente verificar el funcionamiento y los resultados en la plataforma Ubidots. #Pag_61

4. RESULTADOS

1. ANÁLISIS DE RESULTADOS

4.1.1. Comunicación entre variadores de frecuencia y autómata.

La comunicación entre variadores de velocidad fue realizada de manera exitosa al implementar las redes profinet IO y profibus DP. Como se aprecia en la figura 60, se hace uso de diferentes colores de cable para las redes utilizados; para la red profibus DP se utiliza el cable color morado el cual es par trenzado y apantallado cumpliendo con las normativas existentes. Adicional a ello se ha considerado la tecnología *"fast connect"* para la red profibus. En este ítem se puede evidenciar las ventajas del uso de buses de campo en comparación al cableado paralelo para los variadores de frecuencia.

Figura60. Implementación de red entre accionamientos y PLC

4.1.2. Desarrollo de aplicación HMI en la plataforma Ubidots.

Se han realizado diversas pruebas con la plataforma Ubidots para verificar el monitoreo de todas las variables configuradas tal como se muestra en la figura 61.

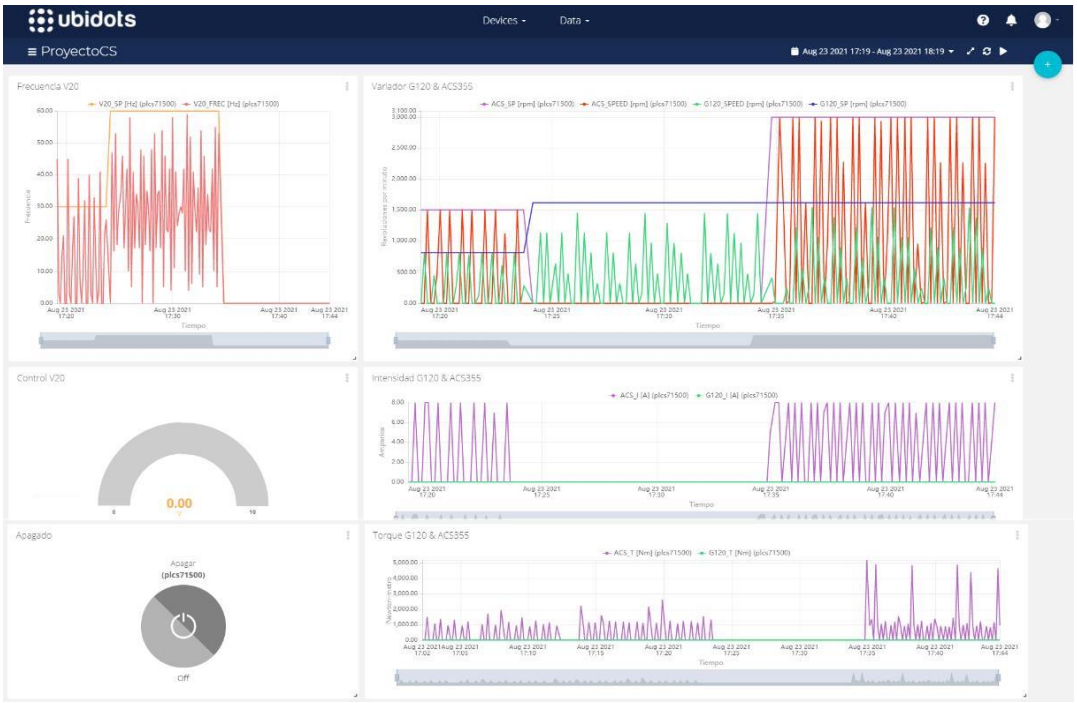


Figura61. Pruebas de Dashboard diseñado en Ubidots

En las figuras 62, 63 y 64 se muestra el resultado de la ejecución de las 4 secuencias predefinidas durante un intervalo de tiempo de aproximadamente 45 minutos. A continuación, se detallan las secuencias que se han probado:

- *Secuencia1*: V20 (t=10s, F=60Hz),
  G120 (t=10s, V=1615rpm),
  ACS355 (t=10s, V=3000rpm)
- *Secuencia2*: V20 (t=10s, F=30Hz),
  G120 (t=10s, V=807rpm),
  ACS355 (t=10s, V=1500rpm)
- *Secuencia3*: V20 (t=10s, F=60Hz),
  G120 (t=10s, V=1615rpm),
  ACS355 (t=0s, V=0rpm)
- *Secuencia4*: V20 (t=0s, F=0Hz),
  G120 (t=10s, V=1615rpm),
  ACS355 (t=10s, V=3000rpm)

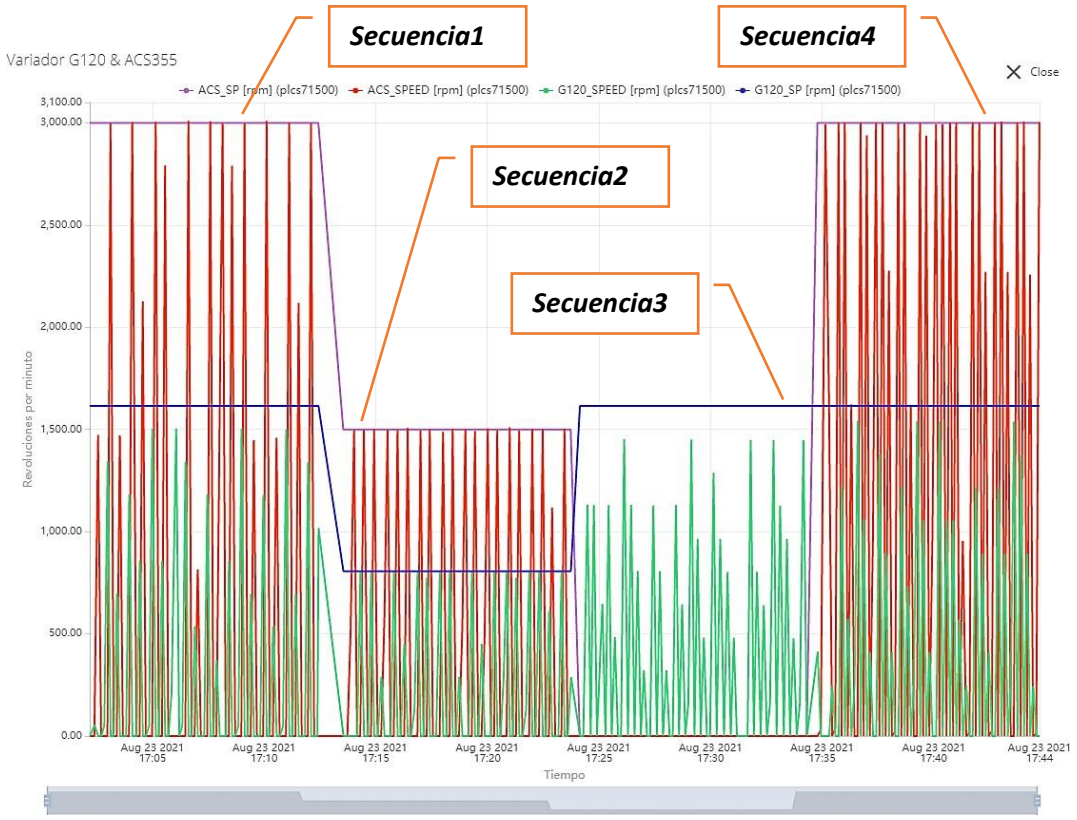


Figura62. Monitoreo de velocidad para variadores G120 y ACS355

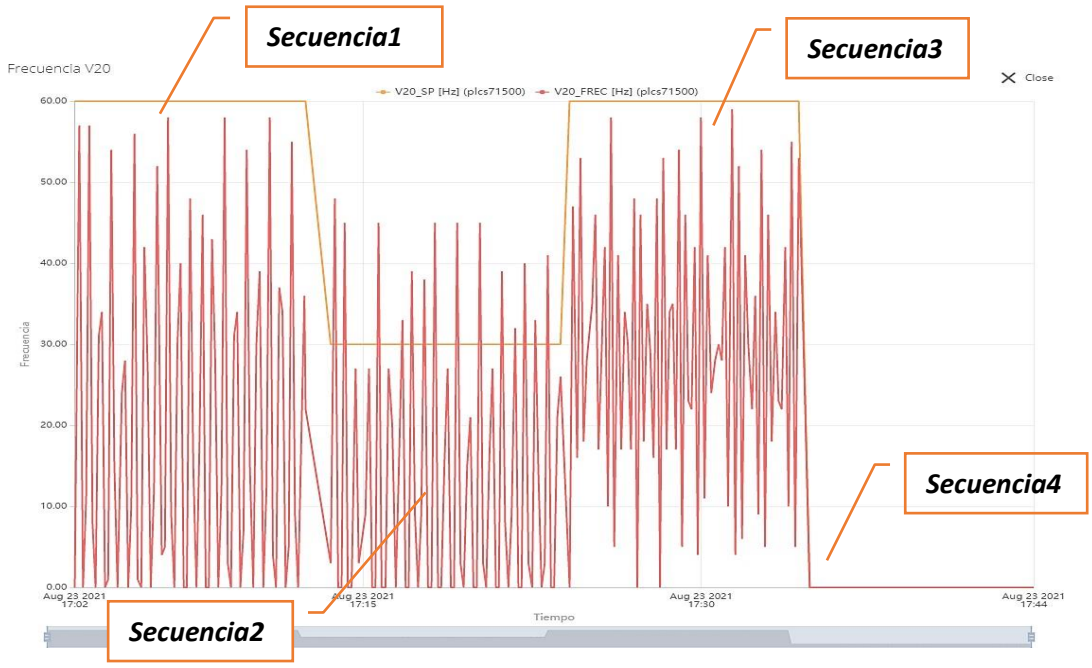


Figura63. Monitoreo de frecuencia para variador V20

Dado que los variadores de frecuencia G120 y ACS355 tiene comunicación por bus de campo PROFINET y PROFIBUS respectivamente, se ha realizado

el monitoreo de los valores de intensidad de corriente en amperios, los cuales se pueden apreciar en la figura 66.

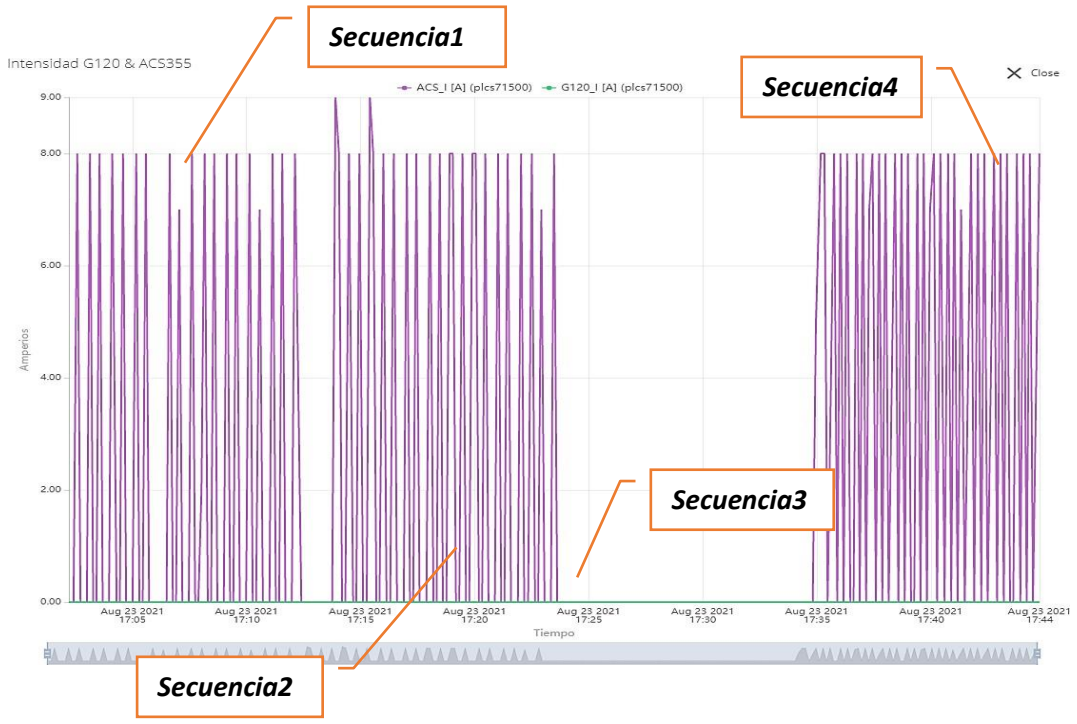


Figura64. Monitoreo de intensidad para variadores G120 y ACS355

CONCLUSIONES

- Al término de este proyecto se tiene una guía práctica y con detalles sobre como implementar una interfaz humano-máquina para monitoreo de variables de proceso a través de la plataforma Ubidots.
- Muchas plataformas para monitoreo de datos en la nube disponibles en la red hoy en día permiten la implementación de proyectos de gama baja sin costo considerando límites en cuanto a la cantidad de puntos utilizados de forma periódica. En el caso de este proyecto se tuvo en cuenta este detalle y se consideró el número de variables y el tiempo de muestreo adecuado para no llegar al límite diario y obtener el mayor beneficio.
- El uso de software LabVIEW para envío y recepción de datos con Ubidots resulta bastante intuitivo para los estudiantes con un conocimiento básico en el manejo de plataformas IOT, lo cual hace que este proyecto sea de mucha ayuda para el aprendizaje de los estudiantes de Electrónica y Automatización.
- El uso de equipos del laboratorio de Automatización permitió la obtención de datos de variadores de velocidad a través de buses de campo como Profinet IO y Profibus DP.
- Para el desarrollo de este proyecto, el uso de buses de campo ha sido una herramienta potente para el manejo de información de forma bidireccional, control y supervisión de accionamientos, lo cual hubiese sido muy trabajo con cableado paralelo.

RECOMENDACIONES

- Se recomienda el mantenimiento periódico de las láminas que contienen equipos de automatización, ya que al pasar el tiempo y con el uso de los mismos, se pueden aflojar tuercas y cableado.
- A la hora de diseño de aplicaciones con plataformas IOT se recomienda buscar un equilibrio entre la cantidad de información a monitorear y la frecuencia de lectura de estas para no rebasar los límites.
- Se recomienda el uso de otras plataformas de monitoreo en la nube para comparar las ventajas y desventajas que tiene una sobre la otro, así como observar la facilidad de configuración de variables.
- Como trabajo a futuro se podría considerar el uso de más equipos de Automatización para generar una red más extensa y así verificar si existen retardos en la supervisión de variables.
- Se recomienda verificar el buen estado de conectores para el cableado de red, ya que este fallo es uno de los más comunes en la implementación de sistemas de comunicación industrial.

REFERENCIAS BIBLIOGRÁFICAS

[1] «Google Maps,» 08 Octubre 2020. [En línea]. Available: https://www.google.es/maps/place/Universidad+Polit%C3%A9cnica+Salesiana+-+Guayaquil/@-2.2205959,-79.8875385,19z/data=!4m5!3m4!1s0x902d6e4fced73235:0xb76f5008ec6c4345!8m2!3d-2.2201497!4d-79.8866846.

[2] A. Moisés Barrio, Internet de las cosas, Madrid: Reus, 2020.

[3] Registratuimei, [En línea]. Available: https://www.registratuimei.cl/en/blog/cuales-son-los-dispositivos-iot/.

[4] M. Navas Sanchez, «El Internet Industrial de las Cosas,» 2015. [En línea]. Available: https://www.redeweb.com/ficheros/articulos/ni_1628283324.pdf.

[5] M. Camila Bernal, 13 Junio 2018. [En línea]. Available: https://www.rutanmedellin.org//es/tendencias/item/ubidots-la-startup-local-que-conecta-datos-de-todo-el-mundo.

[6] Ubidots Community, Abril 2018. [En línea]. Available: https://ubidots.com/community/t/solved-ubidots-dashboards-require-refresh-to-see-the-latest-and-event-not-function/1550.

[7] L. Llamas, «¿Qué es MQTT? Su importancia como protocolo IOT,» 17 Abril 2019. [En línea]. Available: https://www.luisllamas.es/que-es-mqtt-su-importancia-como-protocolo-iot/.

[8] Á. Hita Albarracín, «MQTT vs HTTP: ¿qué protocolo es mejor para IoT?,» Noviembre 2020. [En línea]. Available: https://borrowbits.com/2020/04/mqtt-vs-http-que-protocolo-es-mejor-para-iot/.

[9] MDN Web Docs, «Generalidades del protocolo HTTP,» Agosto 2021. [En línea]. Available: https://developer.mozilla.org/es/docs/Web/HTTP/Overview.

[10] A. Semle, «Protocolos IIoT para considerar,» Octubre 2016. [En línea]. Available: https://editores-srl.com.ar/sites/default/files/aa2_semle_protocolos_ilot.pdf.

[11] J. M. Barceló Ordinas, Protocolos y aplicaciones Internet, Barcelona: Editorial UOC, 2008.

[12] PC-solucion, «Diferencias entre UDP y TCP,» 4 Abril 2018. [En línea]. Available: https://pc-solucion.es/2018/04/04/diferencias-entre-udp-y-tcp/.

[13] National Instruments, «HTTP Client VIs,» Marzo 2018. [En línea]. Available: https://zone.ni.com/reference/en-XX/help/371361R-01/lvcomm/http_client/.

[14] National Instruments, «OpenHandle VI,» Marzo 2018. [En línea]. Available: https://zone.ni.com/reference/en-XX/help/371361R-01/lvcomm/http_client_openhandle/.

[15] National Instruments, «Post VI,» Marzo 2018. [En línea]. Available: https://zone.ni.com/reference/en-XX/help/371361R-01/lvcomm/http_client_post/#instance1.

[16] National Instruments, «Get VI,» Marzo 2018. [En línea]. Available: https://zone.ni.com/reference/en-XX/help/371361R-01/lvcomm/http_client_get/.

[17] National Instruments, «AddHeader VI,» Marzo 2018. [En línea]. Available: https://zone.ni.com/reference/en-XX/help/371361R-01/lvcomm/http_client_addheader/.

[18] F. C. Pahuara Rojas, «Diseño e Implementación de Sistema Automatizado en Cuarto de Bombas para el Suministro de Agua Potable en Edificio Torres Paz,» Lima, 2020.

[19] E. Sánchez, «Eficiencia energética en sistemas de bombeo y compresores,» [En línea]. Available:

https://www.mundohvacr.com.mx/2014/10/eficiencia-energetica-en-sistemas-de-bombeo-y-compresores/.

[20] E. Blanco Marigorta, S. Velarde Suárez y J. Fernández Francos, «Sistema de Bombeo,» 1994. [En línea]. Available: https://agasca.net/wp-content/uploads/2018/08/PDF_SistemasdeBombeo2.pdf.

[21] ABB, «Módulo adaptador PROFIBUS DP FPBA-01,» 2021. [En línea]. Available: https://new.abb.com/drives/es/conectividad/fieldbus/profibus-dp/profibus-dp-fpba-01.

[22] C. Bedell, «big-endian and little-endian,» June 2019. [En línea]. Available: https://searchnetworking.techtarget.com/definition/big-endian-and-little-endian.

ANEXOS

ANEXO1: Diagrama de conexiones

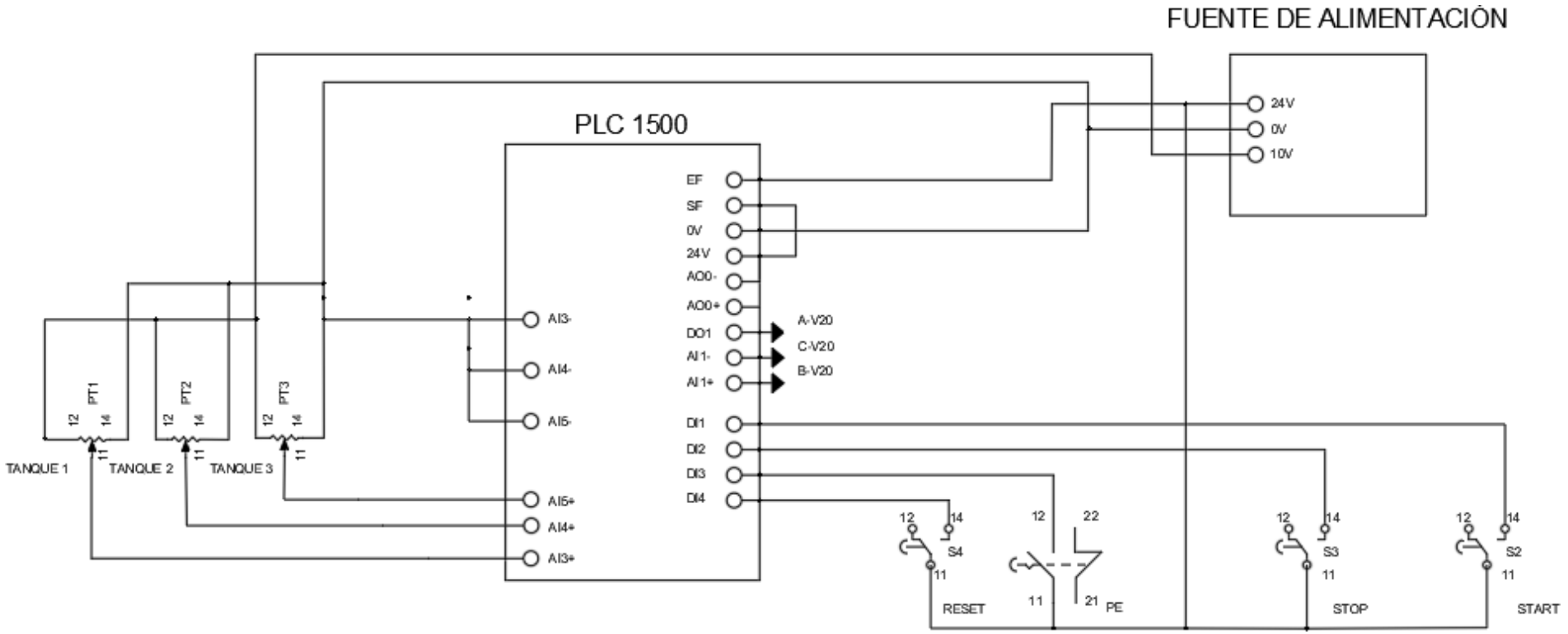


Diagrama de conexiones de control

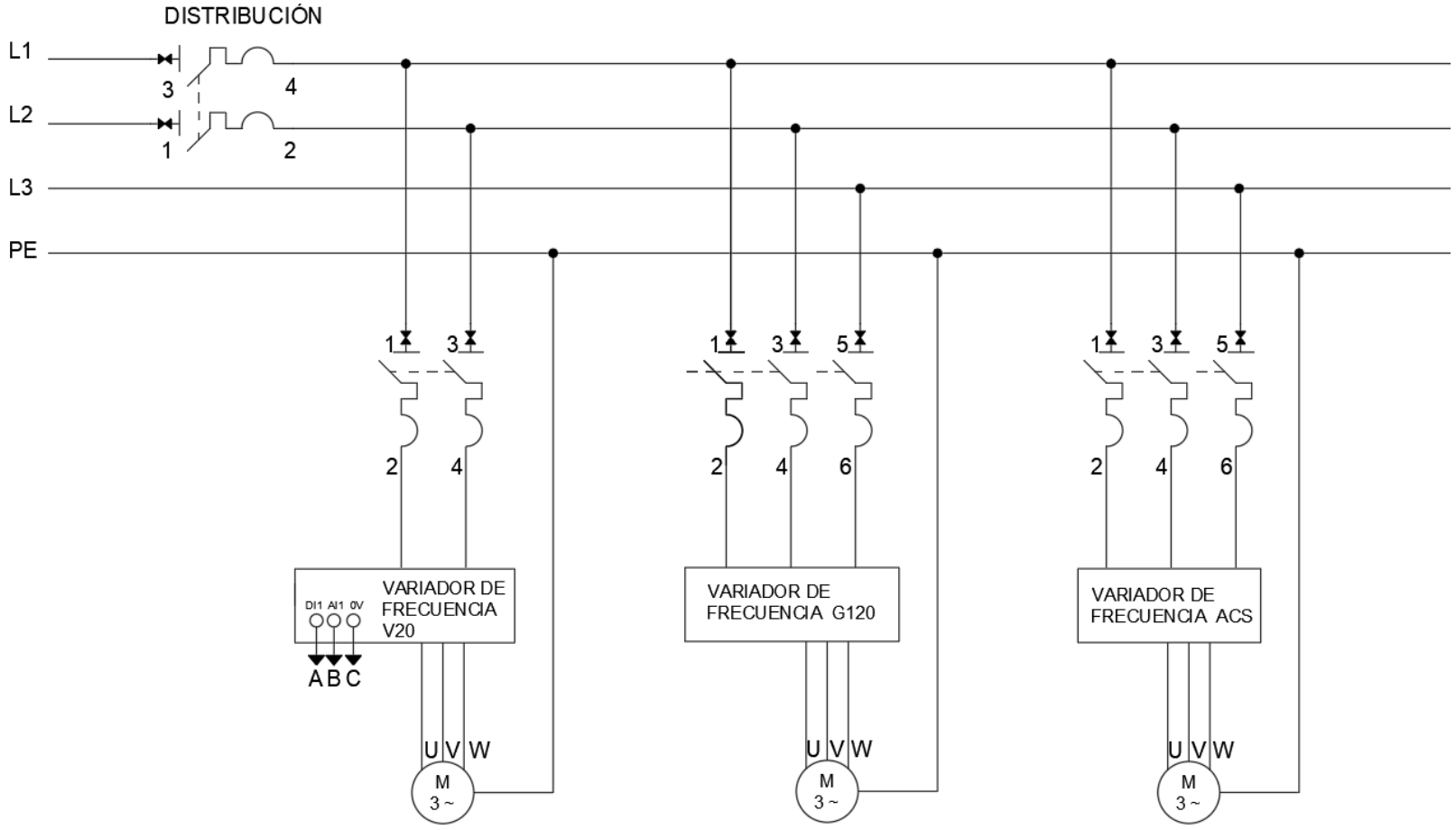
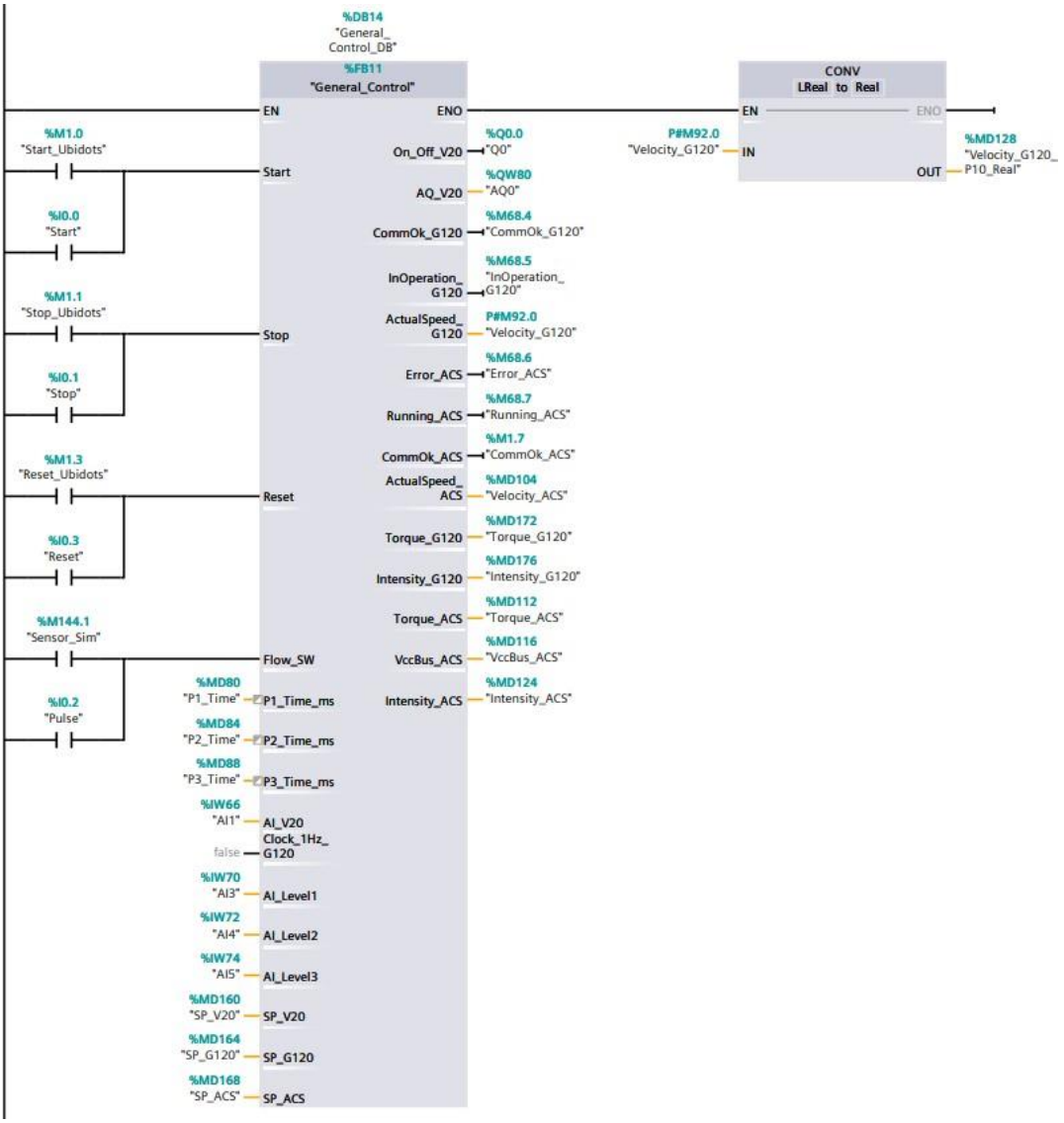


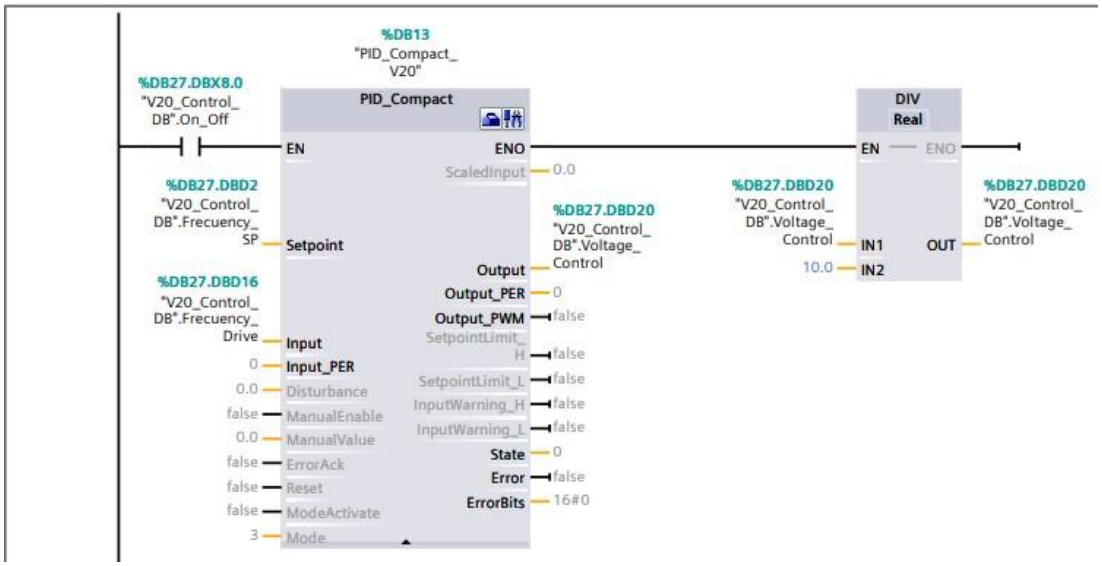
Diagrama de conexiones de fuerza

ANEXO2: Programación del PLC – Main (OB1)

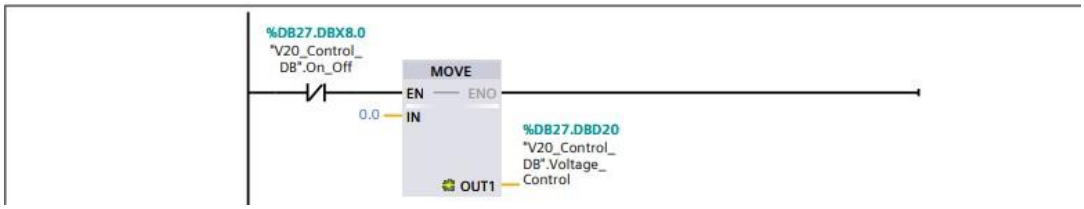


ANEXO3: Programación del PLC – Cycle Interrupt (OB30)

Network 1: Control PID para control de variador V20 - Ts=10ms



Network 2:



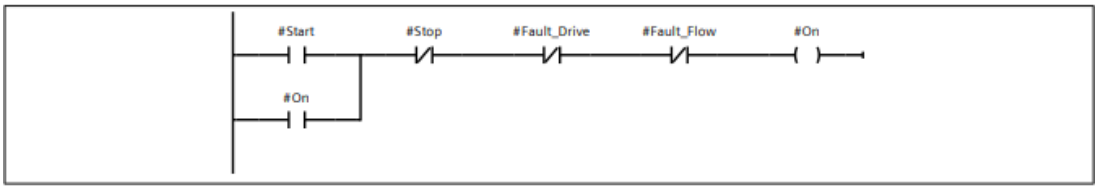
ANEXO4: Programación del PLC – General_Control (FB11)

Name	Data type	Offset	Default value	Acces-sible from HMI/OPC UA	Wri ta-ble fro m HM I/O PC UA	Visible in HMI engi-neer-ing	Set-point	Super-vision	Comment
▼ Input									
Start	Bool	0.0	false	True	Tru e	True	False		
Stop	Bool	0.1	false	True	Tru e	True	False		
Reset	Bool	0.2	false	True	Tru e	True	False		
Flow_SW	Bool	0.3	false	True	Tru e	True	False		
P1_Time_ms	UDInt	2.0	0	True	Tru e	True	False		
P2_Time_ms	UDInt	6.0	0	True	Tru e	True	False		
P3_Time_ms	UDInt	10.0	0	True	Tru e	True	False		
AI_V20	UInt	14.0	0	True	Tru e	True	False		
Clock_1Hz_G120	Bool	16.0	false	True	Tru e	True	False		
AI_Level1	UInt	18.0	0	True	Tru e	True	False		
AI_Level2	UInt	20.0	0	True	Tru e	True	False		
AI_Level3	UInt	22.0	0	True	Tru e	True	False		
SP_V20	Real	24.0	0.0	True	Tru e	True	False		Hz
SP_G120	Real	28.0	0.0	True	Tru e	True	False		rpm
SP_ACS	Real	32.0	0.0	True	Tru e	True	False		rpm
▼ Output									
On_Off_V20	Bool	36.0	false	True	Tru e	True	False		
AQ_V20	Int	38.0	0	True	Tru e	True	False		

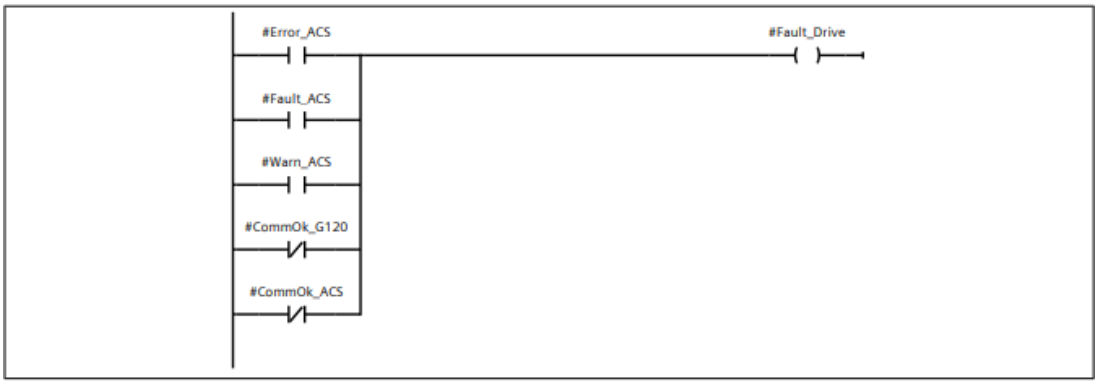
CommOk_G120	Bool	40.0	false	True	True	True	False		
InOperation_G120	Bool	40.1	false	True	True	True	False		
ActualSpeed_G120	LReal	42.0	0.0	True	True	True	False		
Error_ACS	Bool	50.0	false	True	True	True	False		
Running_ACS	Bool	50.1	false	True	True	True	False		
CommOk_ACS	Bool	50.2	false	True	True	True	False		
ActualSpeed_ACS	Real	52.0	0.0	True	True	True	False		
Torque_G120	Real	56.0	0.0	True	True	True	False		
Intensity_G120	Real	60.0	0.0	True	True	True	False		
Torque_ACS	Real	64.0	0.0	True	True	True	False		
VccBus_ACS	Real	68.0	0.0	True	True	True	False		
Intensity_ACS	Real	72.0	0.0	True	True	True	False		
InOut									
▼ Static									
P1	Bool	76.0	false	True	True	True	False		
P2	Bool	76.1	false	True	True	True	False		
P3	Bool	76.2	false	True	True	True	False		
On	Bool	76.3	false	True	True	True	False		
Fault_ACS	Bool	76.4	false	True	True	True	False		
Warn_ACS	Bool	76.5	false	True	True	True	False		
Fault_Drive	Bool	76.6	false	True	True	True	False		
Fault_Tank	Bool	76.7	false	True	True	True	False		
P1_P2_Time_ms	UDInt	78.0	0	True	True	True	False		
Total_ms	DWord	82.0	16#0	True	True	True	False		
ET_ms	DWord	86.0	16#0	True	True	True	False		
SP_Speed_norm_ACS	Real	90.0	0.0	True	True	True	False		
Speed_ref_ACS	Int	94.0	0	True	True	True	False		

Velocity_ACS	Int	96.0	0	True	True	True	False		
Velocity_norm_ACS	Real	98.0	0.0	True	True	True	False		
Fault_Flow	Bool	102.0	false	True	True	True	False		
Temp									
▼ Constant									
SP_Speed_ACS	Real		3000.0						

Network 1: Control de encendido de la secuencia

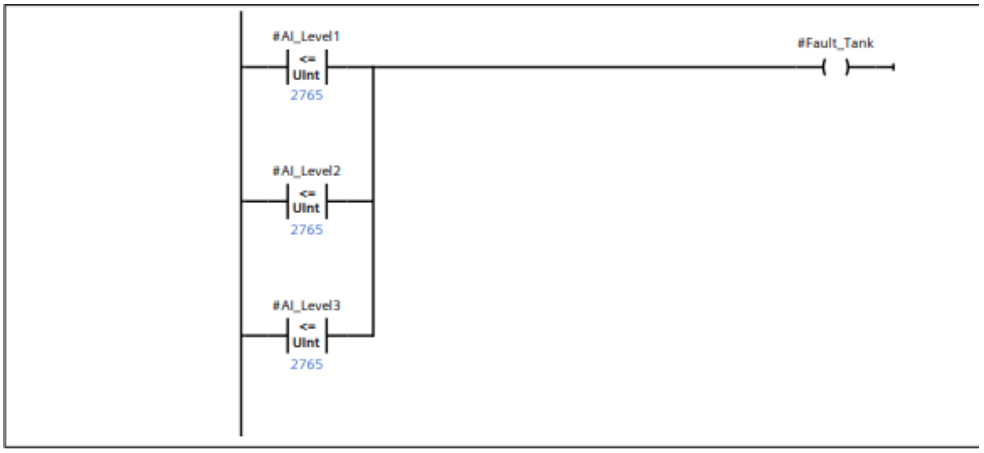


Network 2: Parada de la secuencia a causa de alguna eventualidad en los motores

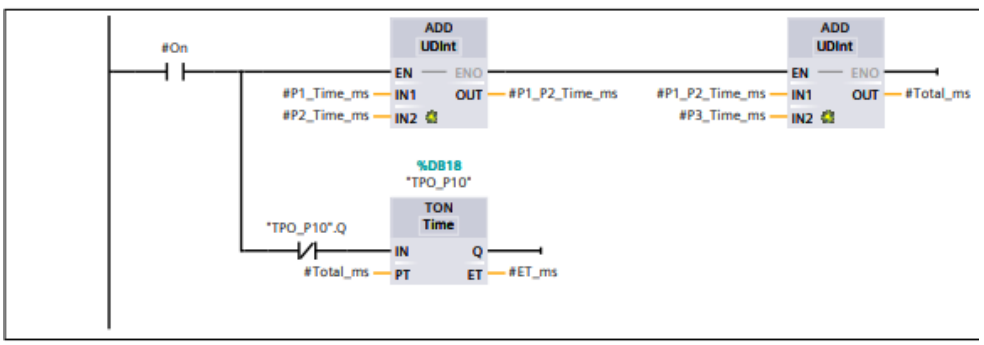


Network 3: Parada de la secuencia a causa de bajo nivel en alguno de los reservorios

Si el nivel de alguno de los tanques es menor al 10% (2764.8 aprox... 2765), se apagará la secuencia

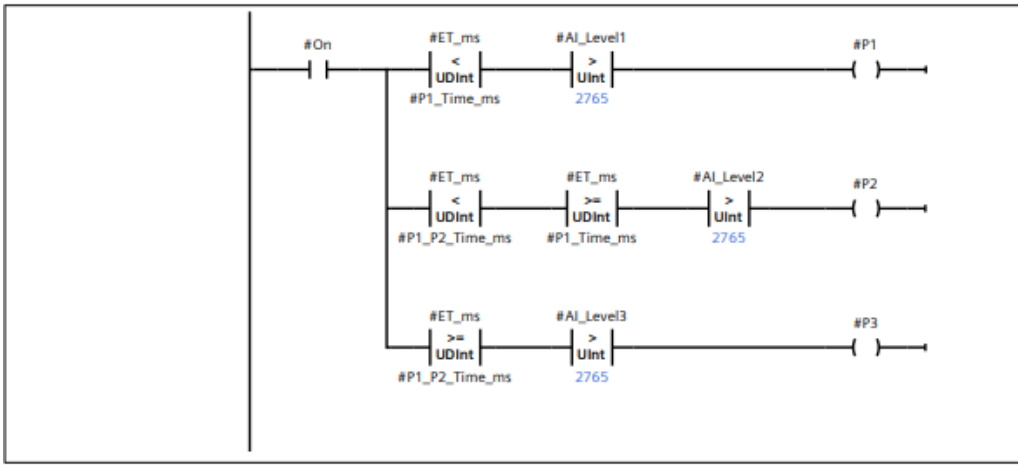


Network 4: Cálculo de tiempos de trabajo y temporizador general

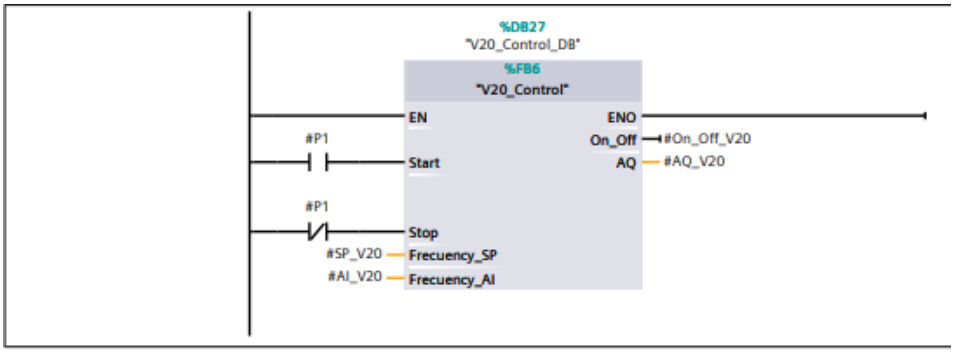


Network 5: Control de encendido de bombas según receta seleccionada

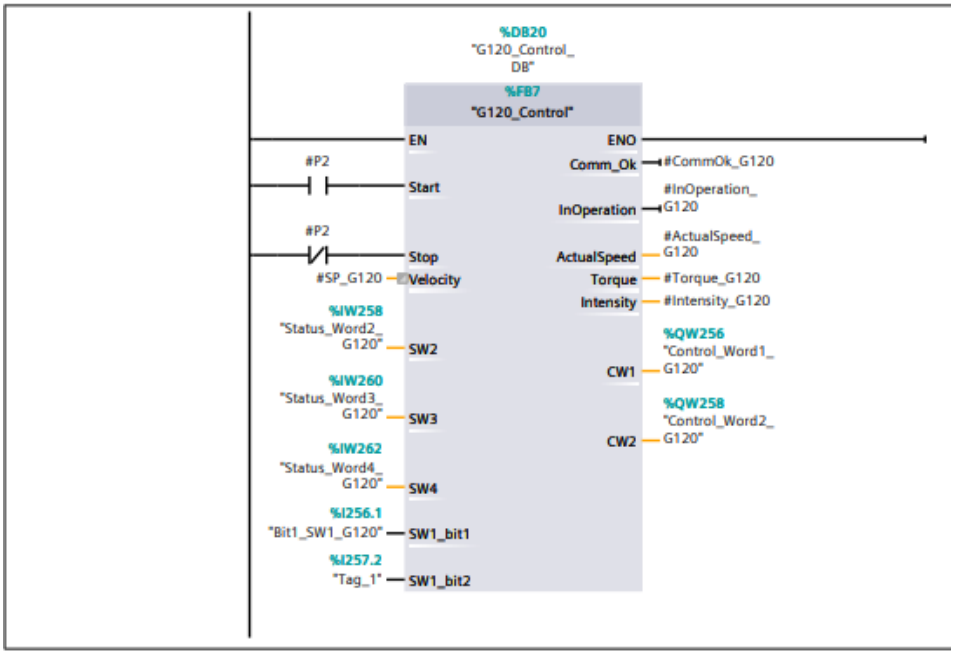
Orden de encendido: P1, P2 y P3



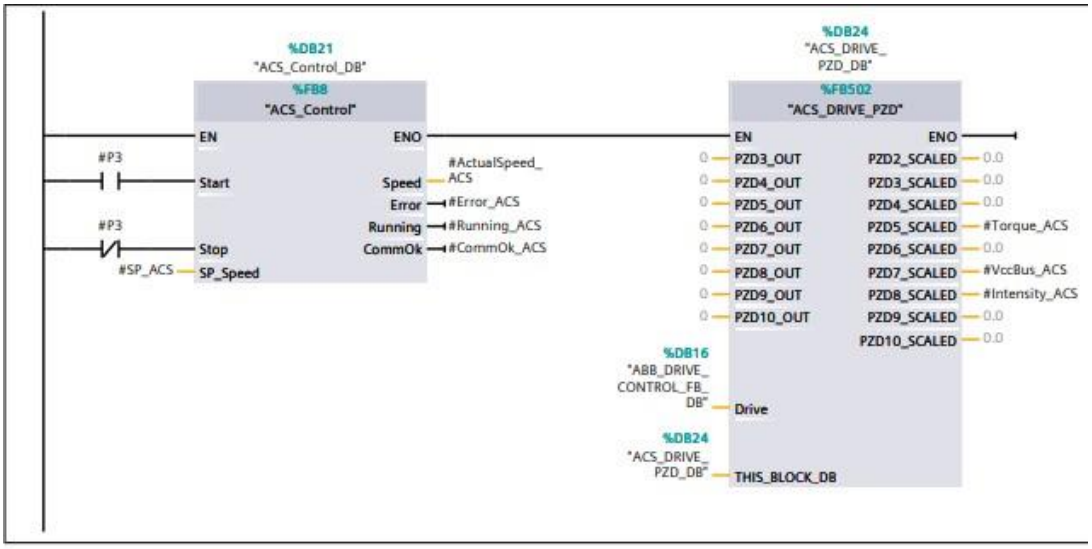
Network 6: Control de encendido de motor1 - Sinamic V20



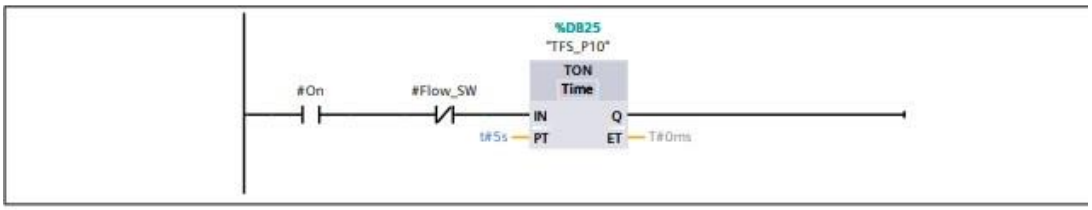
Network 7: Control de encendido de motor2 - Sinamic G120



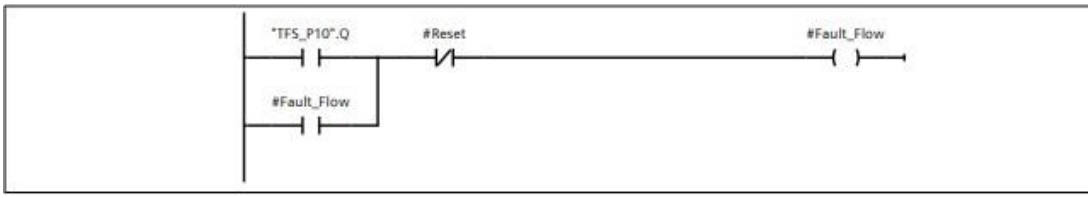
Network 8: Control de encendido de motor3 - ACS355



Network 9: Medición de tiempo para detección de flujo



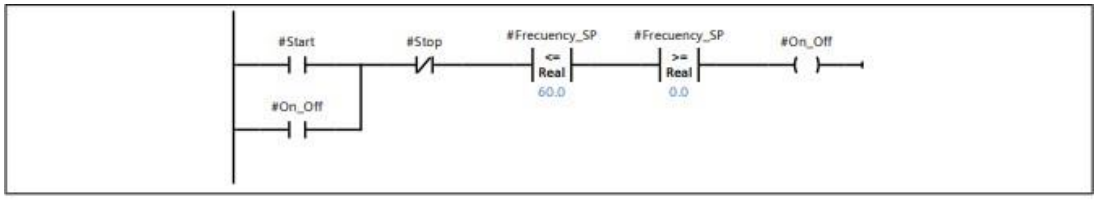
Network 10: Indicador de falla en el flujo



ANEXO5: Programación del PLC – V20_Control (FB6)

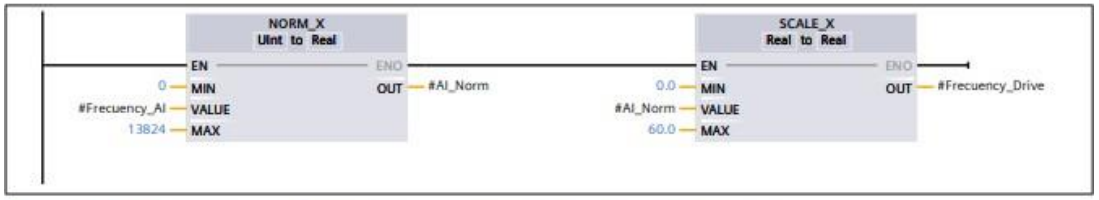
Name	Data type	Offset	Default value	Acces-sible from HMI/OPC UA	Wri ta-ble fro m HM I/O PC UA	Visible in HMI engi-neer-ing	Set-point	Super-vision	Comment
▼ Input									
Start	Bool	0.0	false	True	Tru e	True	False		
Stop	Bool	0.1	false	True	Tru e	True	False		
Frecuency_SP	Real	2.0	0.0	True	Tru e	True	False		
Frecuency_AI	UInt	6.0	0	True	Tru e	True	False		
▼ Output									
On_Off	Bool	8.0	false	True	Tru e	True	False		
AQ	Int	10.0	0	True	Tru e	True	False		
InOut									
▼ Static									
AQ_Norm	Real	12.0	0.0	True	Tru e	True	False		
Frecuency_Drive	Real	16.0	0.0	True	Tru e	True	False		
Voltage_Control	Real	20.0	0.0	True	Tru e	True	False		
AI_Norm	Real	24.0	0.0	True	Tru e	True	False		
Temp									
Constant									

Network 1: Control de encendido y apagado del controlador



Network 2: Acondicioanmienro de señal de entrada para estimación de frecuencia

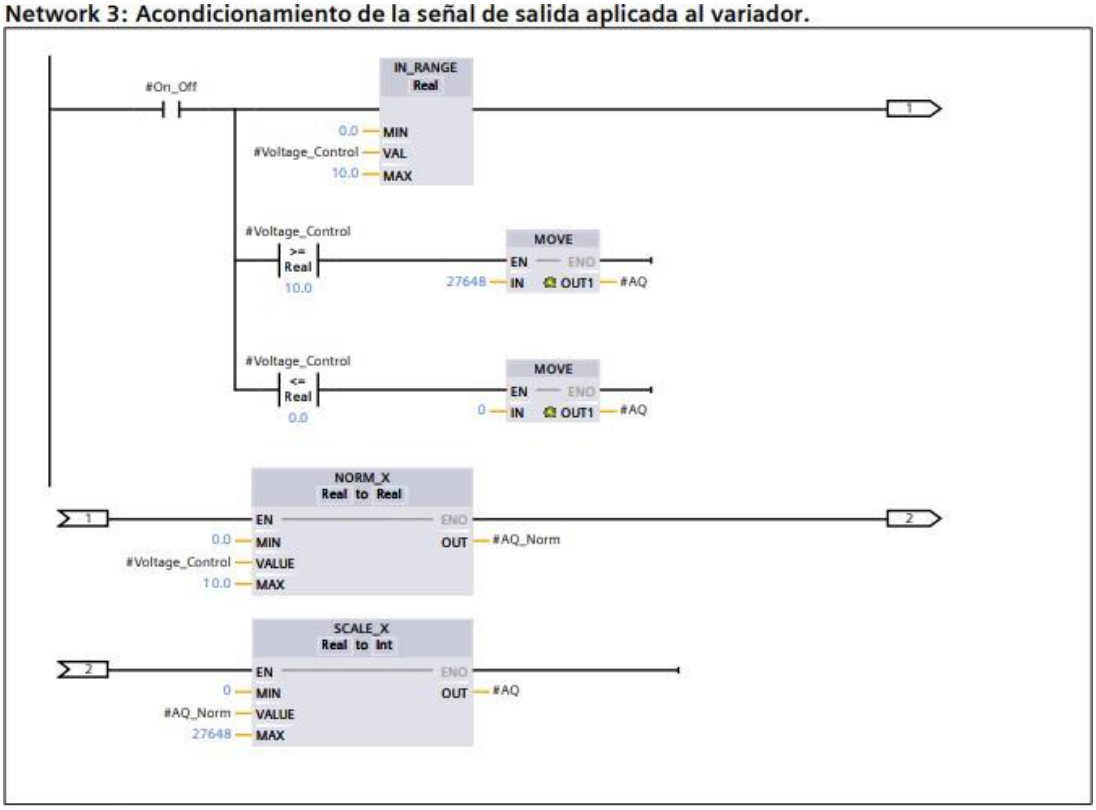
27648/13824



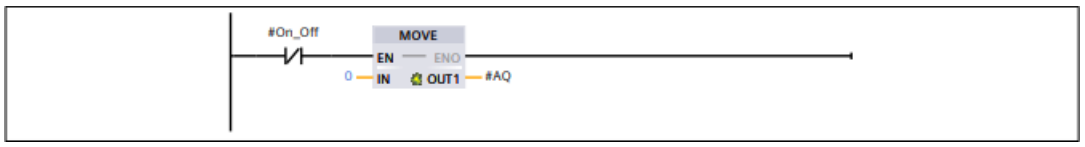
Network 3: Acondicionamiento de la señal de salida aplicada al variador.

La variable #Voltage_Control viene de la salida del bloque PID_Compact que se encuentra en el Cyclic Interrupt OB30.

Network 3: Acondicionamiento de la señal de salida aplicada al variador.



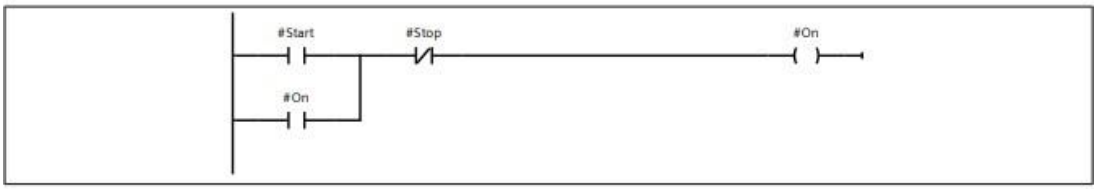
Network 4: Acciones de parada



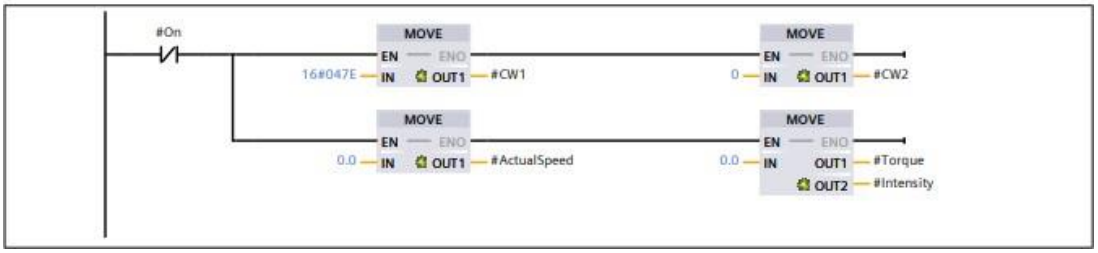
ANEXO6: Programación del PLC – G120_Control (FB7)

Name	Data type	Default value	Retain	Acces-sible from HMI/OPC UA	Wri ta-ble fro m HM I/O PC UA	Visible in HMI engi-neer-ing	Set-point	Super-vision	Comment
▼ Input									
Start	Bool	false	Set in IDB	True	Tru e	True	False		
Stop	Bool	false	Set in IDB	True	Tru e	True	False		
Velocity	LReal	0.0	Non-retain	True	Tru e	True	False		0...1615rpm
SW2	Int	0	Non-retain	True	Tru e	True	False		
SW3	Int	0	Non-retain	True	Tru e	True	False		
SW4	Int	0	Non-retain	True	Tru e	True	False		
SW1_bit1	Bool	false	Non-retain	True	Tru e	True	False		
SW1_bit2	Bool	false	Non-retain	True	Tru e	True	False		
▼ Output									
Comm_Ok	Bool	false	Non-retain	True	Tru e	True	False		
InOperation	Bool	false	Non-retain	True	Tru e	True	False		
ActualSpeed	LReal	0.0	Non-retain	True	Tru e	True	False		
Torque	Real	0.0	Non-retain	True	Tru e	True	False		
Intensity	Real	0.0	Non-retain	True	Tru e	True	False		
CW1	UInt	0	Non-retain	True	Tru e	True	False		Control word1
CW2	Int	0	Non-retain	True	Tru e	True	False		Control word2
InOut									
▼ Static									
On	Bool	false	Non-retain	True	Tru e	True	False		
Temp	Real	0.0	Non-retain	True	Tru e	True	False		
Temp1	Real	0.0	Non-retain	True	Tru e	True	False		
Temp2	Real	0.0	Non-retain	True	Tru e	True	False		
Temp3	Real	0.0	Non-retain	True	Tru e	True	False		
Temp									
Constant									

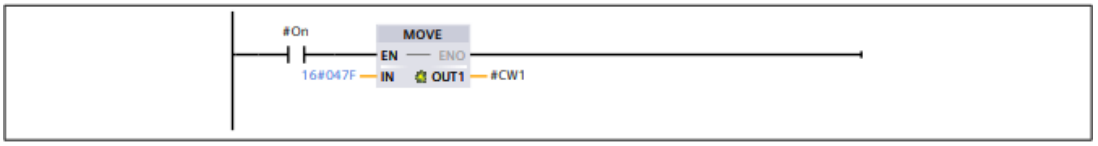
Network 1: Control de encendido del motor



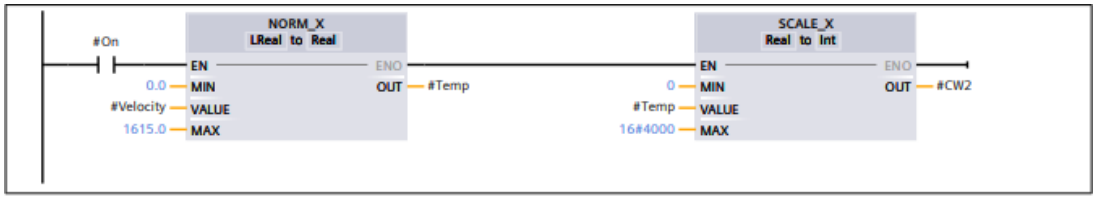
Network 2: Alistar el variador para la puesta en marcha



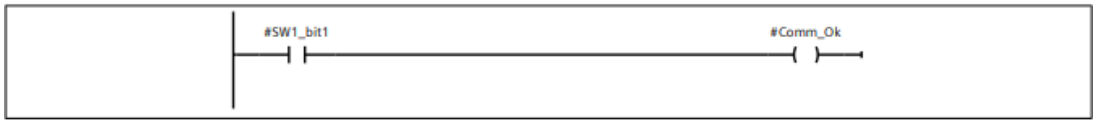
Network 3: Encendido del variador



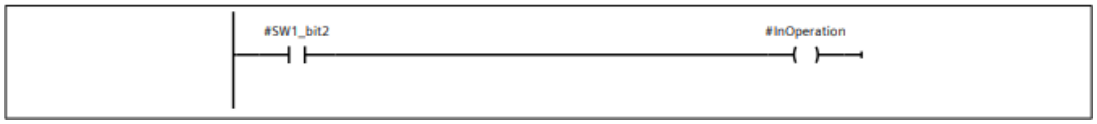
Network 4: Escalamiento de referencia



Network 5: Indicador de estado de comunicación



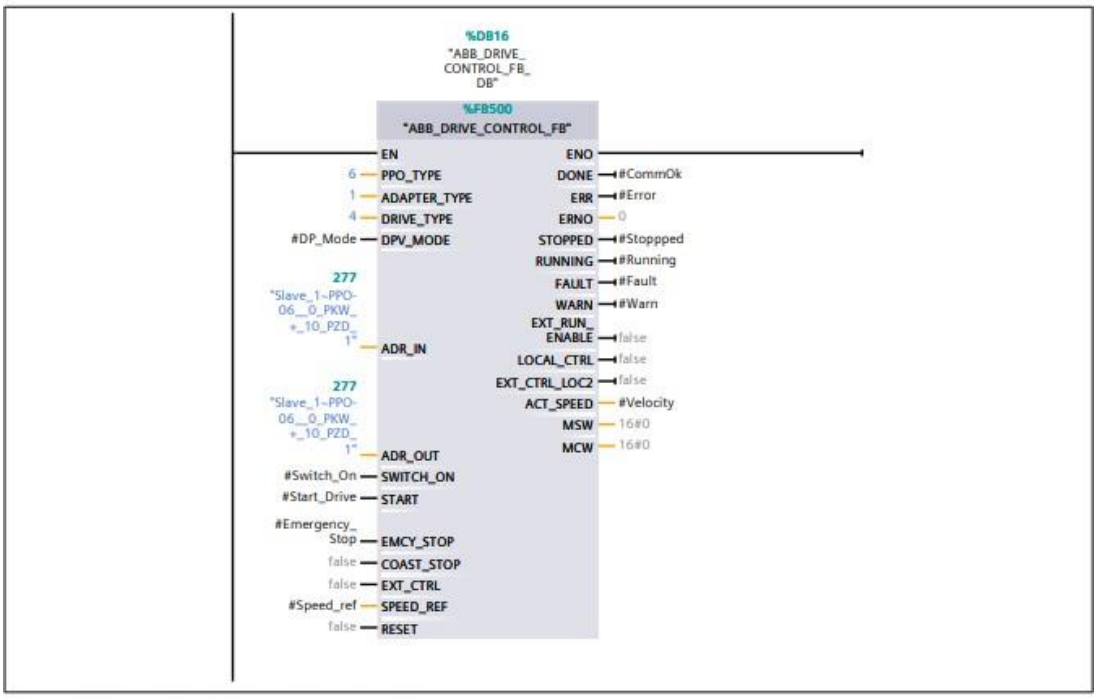
Network 6: Indicador de motor en funcionamiento



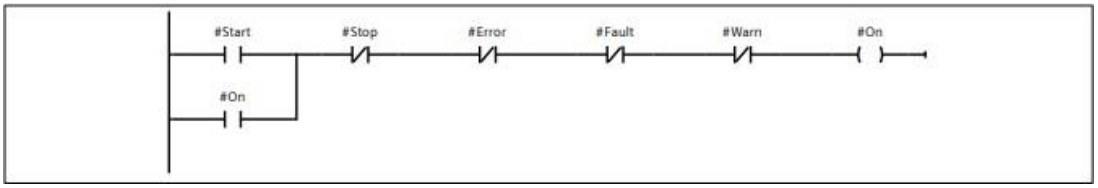
ANEXO7: Programación del PLC – ACS_Control (FB8)

Name	Data type	Default value	Retain	Acces-sible from HMI/OPC UA	Wri ta-ble fro m HM I/O PC UA	Visible in HMI engi-neer-ing	Set-point	Super-vision	Comment
▼ Input									
Start	Bool	false	Set in IDB	True	Tru e	True	False		
Stop	Bool	false	Set in IDB	True	Tru e	True	False		
SP_Speed	Real	0.0	Non-retain	True	Tru e	True	False		
▼ Output									
Speed	Real	0.0	Non-retain	True	Tru e	True	False		
Error	Bool	false	Non-retain	True	Tru e	True	False		
Running	Bool	false	Non-retain	True	Tru e	True	False		
CommOk	Bool	false	Non-retain	True	Tru e	True	False		
InOut									
▼ Static									
On	Bool	false	Non-retain	True	Tru e	True	False		
SP_Speed_norm	Real	0.0	Non-retain	True	Tru e	True	False		
Velocity	Int	0	Non-retain	True	Tru e	True	False		
Speed_ref	Int	0	Non-retain	True	Tru e	True	False		
Velocity_norm	Real	0.0	Non-retain	True	Tru e	True	False		
DP_Mode	Bool	false	Non-retain	True	Tru e	True	False		
Start_Drive	Bool	false	Non-retain	True	Tru e	True	False		
Switch_On	Bool	false	Non-retain	True	Tru e	True	False		
Emergency_Stop	Bool	false	Non-retain	True	Tru e	True	False		
Stoppped	Bool	false	Non-retain	True	Tru e	True	False		
Fault	Bool	false	Non-retain	True	Tru e	True	False		
Warn	Bool	false	Non-retain	True	Tru e	True	False		
Temp									
Constant									

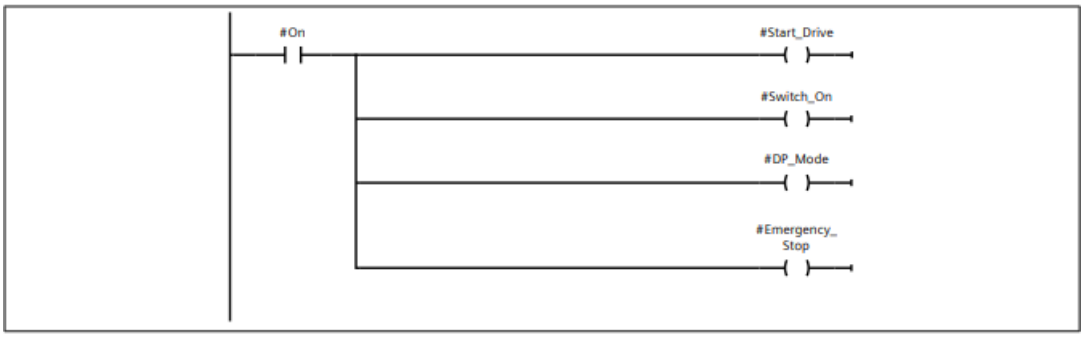
Network 1: Bloque de control para regulación de velocidad en rpm



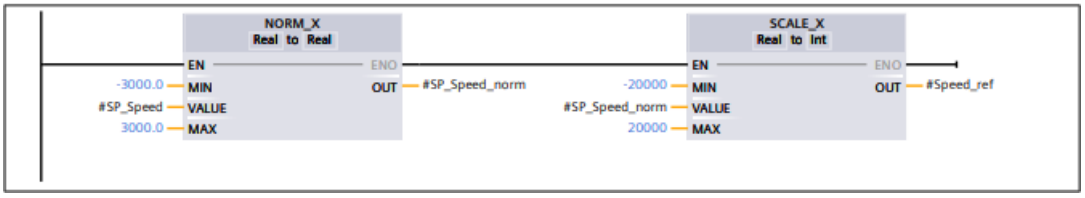
Network 2: Control del encendido del motor



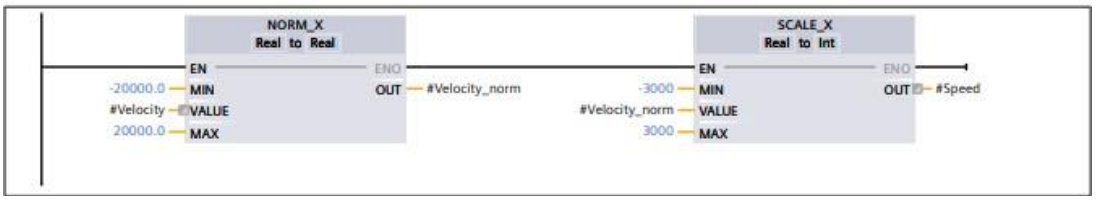
Network 3: Activación de variables de control para el drive



Network 4: Acondicionamiento de la velocidad deseada antes de aplicar al bloque de control



Network 5: Escalamiento de la velocidad actual estimada por el variador de velocidad



ANEXO8: Tabla de variables del autómata

	Variable	Type	Address
1	Start	Bool	%I0.0
2	Stop	Bool	%I0.1
3	Pulse	Bool	%I0.2
4	Reset	Bool	%I0.3
5	AI0	UInt	%IW64
6	AI1	UInt	%IW66
7	AI3	UInt	%IW70
8	AI4	UInt	%IW72
9	AI5	UInt	%IW74
10	Status_Word1_G120	UInt	%IW256
11	Bit1_SW1_G120	Bool	%I256.1
12	Bit2_SW1_G120	Bool	%I256.2
13	Tag_1	Bool	%I257.2
14	Status_Word2_G120	Int	%IW258
15	Status_Word3_G120	Int	%IW260
16	Status_Word4_G120	Int	%IW262
17	Q0	Bool	%Q0.0
18	Q1	Bool	%Q0.1
19	AQ0	Int	%QW80
20	AQ1	Int	%QW82
21	AQ2	Int	%QW84
22	AQ3	Int	%QW86
23	Control_Word1_G120	UInt	%QW256
24	Control_Word2_G120	Int	%QW258
25	N_Practice	Byte	%MB0
26	Start_Ubidots	Bool	%M1.0
27	Stop_Ubidots	Bool	%M1.1
28	Sequence_P5	Bool	%M1.2
29	Reset_Ubidots	Bool	%M1.3
30	P1_Manual	Bool	%M1.4
31	P2_Manual	Bool	%M1.5
32	P3_Manual	Bool	%M1.6
33	CommOk_ACS	Bool	%M1.7
34	Vin0	Real	%MD4
35	Vin1	Real	%MD8
36	Vin2	Real	%MD12

	Variable	Type	Address
37	Vin3	Real	%MD16
38	Vin4	Real	%MD20
39	Vin5	Real	%MD24
40	Vin6	Real	%MD28
41	Vin7	Real	%MD32
42	Frecuency	Real	%MD36
43	SP_Frecuency_P6	Real	%MD44
44	SP_Velocity_P7	LReal	%M52.0
45	Velocity_P7	LReal	%M60.0
46	CommOk_P7	Bool	%M68.0
47	InOperation_P7	Bool	%M68.1
48	Error_P8	Bool	%M68.2
49	Running_P8	Bool	%M68.3
50	CommOk_G120	Bool	%M68.4
51	InOperation_G120	Bool	%M68.5
52	Error_ACS	Bool	%M68.6
53	Running_ACS	Bool	%M68.7
54	Velocity_P8	Real	%MD72
55	SP_Velocity_P8	Real	%MD76
56	P1_Time	Real	%MD80
57	P2_Time	Real	%MD84
58	P3_Time	Real	%MD88
59	Velocity_G120	LReal	%M92.0
60	Clock_Byte	Byte	%MB100
61	Clock_10Hz	Bool	%M100.0
62	Clock_5Hz	Bool	%M100.1
63	Clock_2.5Hz	Bool	%M100.2
64	Clock_2Hz	Bool	%M100.3
65	Clock_1.25Hz	Bool	%M100.4
66	Clock_1Hz	Bool	%M100.5
67	Clock_0.625Hz	Bool	%M100.6
68	Clock_0.5Hz	Bool	%M100.7
69	Velocity_ACS	Real	%MD104
70	Torque_ACS	Real	%MD112
71	VccBus_ACS	Real	%MD116
72	Intensity_ACS	Real	%MD124

	Variable	Type	Address
73	Velocity_G120_P10_Real	Real	%MD128
74	ManAuto_Selector	Bool	%M144.0
75	Sensor_Sim	Bool	%M144.1
76	SP_V20	Real	%MD160
77	SP_G120	Real	%MD164
78	SP_ACS	Real	%MD168
79	Torque_G120	Real	%MD172
80	Intensity_G120	Real	%MD176