UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES
INSTITUTO DE POSTGRADO

TÍTULO

Automatización de una máquina de helado soft: integración de control local, monitoreo en la nube y gestión remota de la producción usando Arduino Cloud

AUTOR

Cachiguango Antamba, Luis Ivan

TRABAJO DE TITULACIÓN

Previo a la obtención del grado académico en
MAGÍSTER EN ELECTRÓNICA Y AUTOMATIZACIÓN

TUTOR

Chávez García, Geovanny Danilo

Santa Elena, Ecuador

Año 2025

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES
INSTITUTO DE POSTGRADO

TRIBUNAL DE SUSTENTACIÓN

Ing. Alicia Andrade Vera, Mgtr.
COORDINADORA DEL PROGRAMA

Ing. Geovanny Chávez García, Ph.D.
TUTOR

Ing. Fernando Chamba Macas, Mgtr.
DOCENTE ESPECIALISTA

Ing. Luis Chuquimarca Jiménez, Mgtr.
DOCENTE ESPECIALISTA

Abg. María Rivera González, MSc.
SECRETARIA GENERAL
UPSE

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES
INSTITUTO DE POSTGRADO

CERTIFICACIÓN

Certifico que luego de haber dirigido científica y técnicamente el desarrollo y estructura final del trabajo, este cumple y se ajusta a los estándares académicos, razón por el cual apruebo en todas sus partes el presente trabajo de titulación que fue realizado en su totalidad por LUIS IVAN CACHIGUANGO ANTAMBA, como requerimiento para la obtención del título de Magíster en Electrónica y Automatización.

TUTOR

Firmado electrónicamente por:
GEOVANNY DANILO CHAVEZ GARCIA
Validar únicamente con FirmaEC

Ing. Geovanny Chávez García, Ph.D.

Santa Elena, 14 de abril de 2025

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES
INSTITUTO DE POSTGRADO

DECLARACIÓN DE RESPONSABILIDAD

Yo, **Luis Ivan Cachiguango Antamba**

DECLARO QUE:

El trabajo de Titulación, AUTOMATIZACIÓN DE UNA MÁQUINA DE HELADO SOFT: INTEGRACIÓN DE CONTROL LOCAL, MONITOREO EN LA NUBE Y GESTIÓN REMOTA DE LA PRODUCCIÓN USANDO ARDUINO CLOUD previo a la obtención del título en Magíster en Electrónica y Automatización, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del Trabajo de Titulación referido.

Santa Elena, 14 de abril de 2025

EL AUTOR

Luis Ivan Cachiguango Antamba

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE CIENCIAS DE LA INGENIERÍA
INSTITUTO DE POSTGRADO

CERTIFICACIÓN DE ANTIPLAGIO

Certifico que después de revisar el documento final del trabajo de titulación denominado AUTOMATIZACIÓN DE UNA MÁQUINA DE HELADO SOFT: INTEGRACIÓN DE CONTROL LOCAL, MONITOREO EN LA NUBE Y GESTIÓN REMOTA DE LA PRODUCCIÓN USANDO ARDUINO CLOUD, presentado por el estudiante, Luis Ivan Cachiguango Antamba, fue enviado al Sistema Antiplagio COMPILATIO, presentando un porcentaje de similitud correspondiente al 7%, por lo que se aprueba el trabajo para que continúe con el proceso de titulación.

CERTIFICADO DE ANÁLISIS
magister

TESIS_AUTOMATIZACIÓN DE UNA MÁQUINA DE HELADO SOFT_FINAL

7% Textos sospechosos

3% Similitudes
0% similitudes entre comillas
< 1% entre las fuentes mencionadas
0% Idiomas no reconocidos
3% Textos potencialmente generados por la IA

Nombre del documento: TESIS_AUTOMATIZACIÓN DE UNA MÁQUINA DE HELADO SOFT_FINAL.pdf
ID del documento: fa5b8f72fb09437c37c8453612a7b9a2aeb54cc3
Tamaño del documento original: 8,35 MB

Depositante: GEOVANNY DANILO CHAVEZ GARCIA
Fecha de depósito: 14/4/2025
Tipo de carga: interface
fecha de fin de análisis: 14/4/2025

Número de palabras: 18.947
Número de caracteres: 132.822

TUTOR

Firmado electrónicamente por:
GEOVANNY DANILO CHAVEZ GARCIA
Validar únicamente con FirmaEC

Ing. Geovanny Chávez García, Ph.D.

UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA
FACULTAD DE SISTEMAS Y TELECOMUNICACIONES
INSTITUTO DE POSTGRADO

AUTORIZACIÓN

Yo, **Luis Ivan Cachiguango Antamba**

Autorizo a la Universidad Estatal Península de Santa Elena, para que haga de este trabajo de titulación o parte de él, un documento disponible para su lectura consulta y procesos de investigación, según las normas de la Institución.

Cedo los derechos en línea patrimoniales de mi Proyecto de titulación con componentes de investigación aplicada y/o de desarrollo con fines de difusión pública, además apruebo la reproducción de este Proyecto de titulación con componentes de investigación aplicada y/o de desarrollo dentro de las regulaciones de la Universidad, siempre y cuando esta reproducción no suponga una ganancia económica y se realice respetando mis derechos de autor.

Santa Elena, 14 de abril de 2025

EL AUTOR

Firmado electrónicamente por:
LUIS IVAN
CACHIGUANGO ANTAMBA
Validar únicamente con FirmaEC

Luis Ivan Cachiguango Antamba

AGRADECIMIENTO

Quiero agradecer un primer lugar a Dios, siempre me ha fortalecido en los buenos y especialmente malos momentos dándome las ganas para seguir adelante cada día. A mis padres, especialmente a mi madre que siempre me ha brindado su amor y cariño incondicional, siendo para mí, la inspiración que, pese a las limitaciones, siempre ha salido adelante.
A la luz de mi vida, mi hija, que, aunque sea muy pequeña al momento de desarrollar este trabajo siempre en sus grandes ojos veo esa energía que me transmite y me da fuerza para continuar. A mi compañera de vida, mi esposa que siempre me ha dado ánimos para continuar con esta maestría, gracias por su amor y paciencia.

Luis Ivan, Cachiguango Antamba

DEDICATORIA

A mi querida hija, mi inspiración absoluta, para ser mejor padre y persona en la vida.

A mis padres, su apoyo incondicional siempre ha estado presente en mi vida.

A mi esposa, su paciencia, amor y ganas de salir adelante han sido mi motivación para continuar estudiando.

Y también a todas las personas, que de una u otra manera se han hecho presentes ya sea con su apoyo o con sus palabras de motivación para poder realizar este trabajo con éxito.

Luis Ivan, Cachiguango Antamba

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

Este trabajo presenta la implementación de un sistema de automatización para una máquina de helado soft, integrando control local, monitoreo en la nube y gestión remota mediante Arduino Cloud. El objetivo es modernizar el equipo, mejorando su eficiencia operativa, calidad del producto y supervisión remota. Se utilizó una metodología experimental basada en la integración del microPLC LOGO! Siemens, sensores de temperatura y corriente, y comunicación Modbus TCP con un Arduino Nano ESP32. El sistema incorpora distintos modos de funcionamiento (servir, conservación, limpieza, etc.), permitiendo un control automatizado y seguro del proceso. Los resultados muestran una mejora significativa en el control de temperatura, eficiencia energética y reducción de errores operativos. En conclusión, la solución propuesta optimiza la operación de la máquina, disminuye costos, alarga su vida útil y sienta las bases para replicarse en otras aplicaciones industriales similares.

Palabras claves: Automatización, IoT, Arduino Cloud

ABSTRACT

This work presents the implementation of an automation system for a soft serve ice cream machine, integrating local control, cloud monitoring, and remote management through Arduino Cloud. The objective is to modernize the equipment by improving operational efficiency, product quality, and remote supervision. An experimental methodology was applied, based on the integration of a LOGO! Siemens microPLC, temperature and current sensors, and Modbus TCP communication with an Arduino Nano ESP32. The system includes various operating modes (serving, preservation, cleaning, etc.), allowing for automated and safe process control. The results show significant improvements in temperature control, energy efficiency, and reduction of operational errors. In conclusion, the proposed solution optimizes machine operation, reduces costs, extends its lifespan, and lays the foundation for replication in similar industrial applications.

Keywords: Automation, IoT, Arduino Cloud

INTRODUCCIÓN

Las máquinas de helado soft son una indispensable herramienta de productividad para pequeñas y medianas empresas (PYMES) cuyo propósito principal es la producción, venta y distribución de helado. Pero estas herramientas en su mayoría suelen funcionar con sistemas de control caducos, y eso perjudica considerablemente el monitoreo de los procesos mediante conectividad remota. Además no existe el modo de conservación de producto necesario mantener el producto fresco en periodos de no producción en las tolvas de almacenamiento, también se evidencian nulas o muy pocas protecciones eléctricas mediante guardamotores y/o relés térmicos y peor aún sistemas de medición de la corriente en los motores en tiempo real, todos estos problemas hacen que se afecte la calidad del producto, elevando los costos operativos y disminuyen el ciclo de vida útil del producto y la maquinaria (Zafar et al., 2018; Menéndez et al., 2023).

A estos problemas se unen los constantes cortes de electricidad que ha padecido el país, en gran parte durante la estación de verano, algunos de los cuales producen daños en las tarjetas de control, que son sin duda el corazón de la máquina. Este problema pone de manifiesto que se necesita modernizar estas máquinas utilizando soluciones que no solamente enriquezcan la operación del sistema de manera local, sino también de manera remota (Ahmed et al., 2021).

Con base en estos antecedentes, se propone el diseño e implementación de un sistema automatizado basado en el Internet de las Cosas (IoT) y tecnologías de computación en la nube, utilizando hardware como Arduino Nano ESP32, sensores de corriente, sensores de temperatura y componentes de protección eléctrica como guardamotores. Este sistema permitirá la supervisión en tiempo real, la gestión remota y la optimización de recursos, integrando un control local mediante una interfaz hombre-máquina (HMI) intuitiva y un monitoreo remoto a través de la plataforma Arduino Cloud (Dasari et al., 2019).

El proyecto incluye la implementación de modos específicos de funcionamiento de la máquina de helado soft como:

- Modo Servir Helado: Control en modo local que regula la operación del agitador, compresor y válvulas para sacar el helado para su venta

- Modo Hacer Helado: Monitoreo y regulación de temperatura, verificar consistencia del producto mediante consumo de corriente del agitador y controlar la temperatura en rangos de -11°C a 3 °C.
- Modo Conservación Automática: Activación programada para mantener la temperatura de las tolvas por debajo de 8°C.
- Modo Monitoreo Remoto: Registro de helados servidos, monitoreo de alarmas, monitoreo de variables, cambios de modo desde la nube.
- Modo Limpieza: Operación del agitador para facilitar el mantenimiento (Menéndez et al., 2023; Ahmed et al., 2021).

Los beneficios de incorporar estas funcionalidades se verán reflejados en optimizar en costos operativos y el que ahorro en energía eléctrica, además de obtener una mayor calidad del producto y el aumento del período de vida útil de las máquinas. También, este proyecto crea las bases para un sistema que es aplicable también a otras áreas de la economía alimenticia, fomentando la adopción de tecnologías accesibles y escalables en PYMES.

El presente documento está conformado de cinco capítulos, los cuales describen los aspectos más relevantes que se encuentran en esta investigación: El primer capítulo expone el marco teórico que da sustento a la investigación y describe los conceptos de IoT, HMI y comunicación modbus TCP.

En el segundo capítulo, se explica la metodología aplicada, que va desde la elección de la tecnología hasta al diseño investigatorio. En el tercer capítulo es de la implementación del proyecto, seguido por el cuarto capítulo donde se ocupa el análisis de resultados. Por último, el quinto capítulo se presenta las conclusiones y las recomendaciones recogidas de esta investigación

Planteamiento de la investigación (Fundamentación de la investigación)

1. La modernización de las máquinas de helado soft es fundamental para preservar su competitividad bajo los cambios industriales constantes que existen. Por lo que este proyecto se propone es solucionar problemas concretos como no tener conexión a la nube, no hay modo de conservación del producto, y poder controlar en tiempo real variables importantes, entre ellas la temperatura del producto y el

consumo eléctrico en motores principales. La implementación de tecnologías IoT habilita oportunidades para monitorear parámetros, optimizar el funcionamiento de la máquina y disminuir los costes operativos.

Este trabajo está dado por la necesidad de:

1. Establecer la conservación del producto en situaciones de inactividad o cortes de energía, anticipándose con una correcta configuración de temperatura.
2. Facilitar la supervisión remota para dueños y operadores de las máquinas.
3. Establecer un modelo eficiente, accesible y replicable para otras máquinas de la industria alimentaria, apuntando a las PYMES del país

Desde un ámbito profesional y científico, este proyecto promueve el uso de tecnologías actuales como IoT y la computación en la nube, fortaleciendo las competencias en áreas tecnológicas críticas y ofreciendo soluciones innovadoras que impactan positivamente en la industria alimentaria

Formulación del problema de investigación

¿Cómo puede la implementación de un sistema automatizado basado en IoT y Arduino Cloud mejorar la eficiencia operativa, la sostenibilidad y la competitividad de las máquinas de helado soft en pequeñas y medianas empresas?

Objetivo General:

Implementar un sistema automatizado de control moderno y la conectividad a internet de una máquina de helado soft, programando un control local, monitoreo a través de la nube y gestión remota de la producción usando ArduinoCloud, con el objetivo de mejorar su eficiencia, producción y que sea adaptable las demandas actuales del mercado.

Objetivos Específicos:

- Implementar el micro PLC LOGO! Siemens con Arduino Nano ESP32 para supervisar variables clave en Arduino Cloud que permita la supervisión remota de variables críticas y la gestión a distancia de la producción de la máquina de helado soft. Los dos equipos se comunicarán por Modbus TCP.

- Integrar funcionalidades avanzadas como modo conservación de producto, medición de corriente de motores del agitador y compresor para crear algoritmos de control de proceso, medición de consistencia,etc.

- Programar una interfaz HMI fácil de usar con el usuario u operador, mediante TD LOGO!, que facilite la operación y el control de la máquina de manera local.

- Instalar protecciones eléctricas para los dos motores: agitador y compresor, mejorando la seguridad y el monitoreo del equipo, con una gestión de alarmas eficiente y amigable con el usuario. Con el objetivo de alargar la vida útil de la maquinaria.

- Crear diagramas eléctricos normalizados de todo el sistema eléctrico de la máquina para disponer de una documentación clara que sirva para el diagnóstico y solución de fallas, además de facilitar la localización de componentes, rutas de cables, conexiones y puntos de testeo.

- Crear un manual de usuario de funcionamiento del nuevo sistema para su uso por parte del personal operativo y técnico.

Planteamiento hipotético

La implementación de un sistema de control automatizado con IoT y computación en la nube aumenta notablemente la eficiencia de la operación, la sostenibilidad y la competitividad en las máquinas de helado soft, aprovechando mejor los recursos, minimiza los problemas operativos y establece la conectividad, lo que da como resultado una mejor supervisión y gestión remota de la producción.

CAPÍTULO 1. MARCO TEÓRICO REFERENCIAL

1.1. Revisión de literatura

La modernización de procesos industriales mediante la incorporación de tecnologías IoT y soluciones basadas en microcontroladores como Arduino y PLCs ha demostrado ser una estrategia eficiente en diversos sectores. A continuación, se presentan estudios relevantes que fundamentan esta investigación.

Sistemas de Monitoreo Ambiental con IoT

Zafar et al. (2016), en “An IoT Based Real-Time Environmental Monitoring System Using Arduino and Cloud Service ", formuló en ese artículo una fácil opción para monitorear los parámetros ambientales como la temperatura y la humedad utilizando un sensor DHT11, una tarjeta Arduino UNO y un módulo WiFi ESP8266. Se administró el sistema mediante ThingSpeak en la nube, para que los datos fueran analizados en tiempo real y podían ser tratados por las aplicaciones móviles.

En este informe se revelan la importancia de comunicar dispositivos IoT con la nube, con el fin de mejorar la supervisión y poder aplicar estos modelos en una amplia gama de situaciones, destacando su innovación y potencial expansión e integración con otro tipo de sistemas de control de procesos industriales.

Experiencia de Usuario en Interfaces Hombre-Máquina

Aranburu et al. En su estudio "Evaluating the Human Machine Interface Experience in Industrial Workplaces", analizaron la relación que existe entre el diseño de interfaces y la experiencia del usuario (UX) en trabajos industriales. Se usó metodologías de análisis emocional y funcional empleadas para crear HMIs mejorando la interacción humano-máquina en todo aspecto. Concluyeron que un diseño centrado para el usuario aumenta la productividad de la operación y la comodidad del operador, condiciones fundamentales a ser implementadas en las máquinas de helado soft actualizadas a la Industria 4.0.

Robótica y Automatización con IoT

Ahmed et al. (2021), en su artículo *"Design and Implement of Robotic Arm and Control of Moving via IoT with Arduino ESP32"*, implementaron un sistema de control remoto para un brazo robótico, utilizando Arduino ESP32 con sensores ultrasónicos. Los datos fueron procesados mediante comandos web y simulaciones con el software Proteus, demostrando la viabilidad de IoT en entornos industriales para reducir costos y mejorar la precisión operativa. Este estudio establece que hay gran potencial de las tecnologías IoT para optimizar sistemas en aplicaciones de manipulación robótica.

Automatización de la Producción de Helados

Aldana-Fernández et al. (2023), en la tesis “Productivity Improvement by Means of Method Engineering Tools and Automation in Ice Cream Production at Bonanza Company”, analizaron cómo aumentaron la producción de helados mediante el análisis de operaciones y la automatización. A través de analizar tiempos y movimientos, adoptaron tecnologías IoT para supervisar procesos y tuvieron una reducción considerable de tempos muertos y une la mejora de calidad del producto. Este modelo se puede utilizar para modernizar máquinas de helado soft para hacerlas más competitivas.

Automatización de Hogares Inteligentes

En "Smart Homes using Internet of Things", Aravindan y James (2017) estudiaron la ejecución de IoT como controlador remoto de sistemas básicos en las casas, con dispositivos tales como el Raspberry Pi y medios de comunicación mediante MQTT. Sus resultados muestran cómo esas tecnologías pueden estar utilizadas como herramientas para supervisar y monitorear equipos industriales para aumentar la eficiencia de energía.

Desarrollo de Negocios Basados en Helados Soft

Sáenz y Villamarín (2013), en su plan de negocios llamado “Plan de Negocios para la Creación de una Heladería Self-Service Especializada en Helado de Yogurt Soft”, analizaron la posibilidad técnica y económica de utilizar un modelo de negocio automatizado. Se analizó la elección de equipos tecnológicos y actuar sobre la eficiencia en los procesos para obtener productos innovadores. Su conclusión recalca que es vital estar implicados en fomentar el uso de tecnologías avanzadas, para tener una mayor competitividad en la industria alimentaria.

Aplicaciones de IoT en Seguridad y Control

Dasari et al. (2019), en su artículo *"Implementation of Low Cost Home Monitoring, Controlling and Security System using IoT"*, diseñaron un sistema IoT para controlar dispositivos domésticos utilizando NodeMCU ESP8266 y sensores de movimiento. Este sistema, aunque enfocado en aplicaciones domésticas, destaca por su adaptabilidad a entornos industriales para monitorear y gestionar equipos, maximizando la eficiencia operativa y reduciendo costos

1.2. Desarrollo teórico y conceptual

1.2.1. Fundamentos de Sistemas de Refrigeración para Helados Soft

Los sistemas de refrigeración son vitales en la industria alimentaria para mantener las propiedades sensoriales y la calidad de productos sensibles a la temperatura, como los helados soft. Estos sistemas permiten el control de la transferencia de calor durante los procesos de fabricación, garantizando condiciones correctas de textura y conservación. En la Figura 1 se puede observar los componentes principales de un sistema de refrigeración básico.

Figura 1

Esquema de un sistema de refrigeración simple

Nota: Esta figura muestra un sistema de refrigeración simple (Autor desconocido, s.f., p. 1), extraída de Scribd.

1.2.1.1. Composición del Helado Soft y Relevancia del Enfriamiento

Cuando se hace un helado soft, el objetivo es que tenga excelente textura y sabor. Su composición correcta por lo general es que contenga entre 4% y 7% de grasa proveniente en gran medida de la leche entera que se usa, hay también de 13% a 15% de azúcares. También se compone de estabilizantes y emulsificantes en pequeñas cantidades (0.2–0.3% y 0.1–0.2%, respectivamente). Lo que realmente le da su característica suavidad es el aire, representa alrededor del 60% del volumen total. Gracias al aire y a un buen control de la temperatura durante el enfriamiento, se forman cristales de hielo muy pequeños, de entre 10 y 20 micras, que son los que le dan esa consistencia cremosa tan necesaria para su venta (Ballesteros & Quiroga, 2016).

1.2.1.2. Principios de Operación de los Sistemas de Refrigeración

La máquina que se usa para hacer helado soft funciona a través de un ciclo de refrigeración que incluye etapas: compresión, condensación, expansión y evaporación. De esta manera, elementos tales como el compresor, el condensador, la válvula de expansión y los evaporadores contribuyen directamente al movimiento del calor hacia el cilindro.

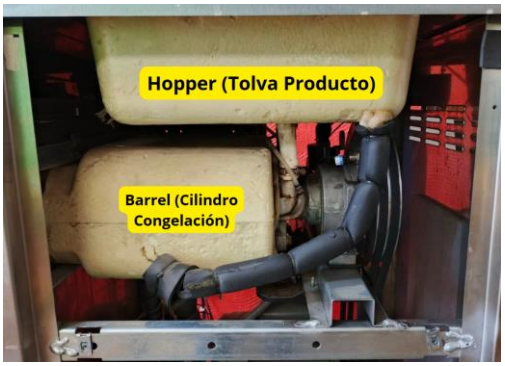
En el caso específico de nuestra máquina de helado soft, los evaporadores se presentan en dos variantes: hopper y barrel. El hopper es una tolva donde se almacena el producto y debe ser refrigerada para que el producto se mantenga en temperaturas de 8 °C hasta aproximadamente 4 °C o menos. Por otro lado, el barrel es una cámara cilíndrica equipada con un intercambiador de calor que permite reducir la temperatura desde los 10 °C hasta -11 °C, lo que contribuye a generar la textura suave y homogénea que se desea en el producto (Carvajalino & Celis, 2021).

La Figura 2 muestra la disposición de las tolvas y los cilindros de nuestra máquina de helado soft, las tolvas tienen un volumen de aproximadamente de 4 litros en cada tolva, ahí es donde se almacena y conserva el producto antes de ingresar a los cilindros.

En los cilindros es donde se congela el producto y mediante el agitador el helado toma una consistencia cremosa y dura, para luego de determinada temperatura el producto queda listo para su servicio al cliente final.

Figura 2

Cilindro y Tolva de una máquina de helado soft



Nota: Esta figura muestra el Hopper y Barrel de la máquina de Helado Soft. Fuente propia.

1.2.1.3. Automatización en Sistemas de Refrigeración

La automatización es un principal factor de la modernización de los sistemas de refrigeración. Lo que hace especial esta tecnología es que es posible monitorizar con seguridad variables de operación como son temperatura, presión, corriente eléctrica, maximizando el uso de los componentes compresores y condensadores. Por ejemplo, en empresas industriales han beneficiado del PLC y HMI el control de sistemas complejos que usan sistemas de refrigeración, reduciendo el error humano y teniendo condiciones adecuados de servicio (Guachi et al., 2016).

1.2.2. Evolución de los Sistemas de Control en la Industria Alimentaria

En la industria alimentaria, los cambios han sido enormes desde que la automatización empezó a formar parte de los procesos. Ya no se trata solo de hacer más rápido, sino de mantener calidad, reducir riesgos y adaptarse a una demanda cada vez más exigente. La presión viene también de fuera: mercados globales, normativas sanitarias, competencia fuerte.

Lo que se busca, básicamente, es reducir la intervención humana directa. “No para quitar gente”, sino para evitar errores repetitivos y garantizar que todo funcione igual cada vez. En esta industria, eso significa usar sistemas de control que están activos desde que entran las materias primas hasta que el producto final se empaca. Se usan PLCs, sensores de

temperatura, de presión, de corriente, actuadores, SCADA, incluso cámaras con visión artificial para detectar defectos. Todo esto no apareció de un día para otro.

La historia arranca con mecanismos simples: relés, interruptores mecánicos, secuencias fijas. Con el tiempo llegaron los autómatas programables y los sistemas SCADA, que dieron un giro enorme al control y la supervisión industrial.

1.2.2.1. Antecedentes Históricos de la Automatización:

Primeros Autómatas: La automatización tiene su historia ancestral. Los egipcios y los griegos inventaron complicados mecanismos imitatorios de los segmentos corporales, para controlar estatuas. Pero en el siglo XVII y el siglo XVIII se diseñaron los primeros autómatas pensados en la diversión. Jacques de Vauncansos fabricó músicos de dimensiones humanas, mientras que Henri Maillardet creó una muñeca mecánica que podía hacer dibujos. (Agudelo, N., Tano, G., & Vargas, C. A. s.f.). En la Figura 3 se presenta "El escritor" creado por Pierre Jaquet-Droz, podía escribir frases programadas (instrucciones mecánicas) en un papel usando una pluma, podía mover la cabeza y ojos, simulando el movimiento humano.

Figura 3

Fotografía de un autómata escribano

Agudelo, N., Tano, G., & Vargas, C. A. (s.f.). *Historia de la automatización* [Figura 1: Muñeco construido en 1805]. Universidad ECCI

La Revolución Industrial y la División del Trabajo: La revolución industrial que tuvo lugar alrededor del siglo XVIII, por que significó un punto de inflexión. El trabajo, analizado por Adam Smith en "La Riqueza de las Naciones", permitió la especialización de trabajadores y de elaborar máquinas que imitaban sus funciones. La mecanización de las fábricas, impulsada por la tecnología de transferencia de energía, incrementó la productividad y sentó las bases para la producción en masa.

La Línea de Montaje y la Producción en Masa: A principios del siglo XX, la industria automotriz revolucionó los conceptos de producción con la línea de montaje, un sistema que integró varias máquinas especializadas en un flujo de trabajo continuo para reducir costos. La línea de montaje se convirtió en el símbolo de la automatización y la producción en masa (Agudelo, N., Tano, G., & Vargas, C. A. s.f.)

1.2.2.2. Sistemas de Control en la Industria Alimentaria

Primeros Sistemas de Control: Los sistemas de control en la industria alimentaria se basaban en elementos mecánicos y electromecánicos: motores, relés, temporizadores y contadores. Estos sistemas, aunque rudimentarios, permitieron automatizar tareas simples y controlar algunas variables del proceso. (Agudelo, N., Tano, G., & Vargas, C. A. s.f.)

La Aparición del PLC: La década de 1960 vio nacer el controlador lógico programable (PLC), una tecnología flexible, robusta y fácil de programar. El PLC permitía controlar procesos secuenciales y gestionar múltiples entradas y salidas, ideal para la industria alimentaria. (Agudelo, N., Tano, G., & Vargas, C. A. s.f.). La Figura 4 muestra lo que es considerado el primer PLC, su inventor fue Dick Morley conocido como el "padre" del autómata programable. Este equipo ya disponía de una arquitectura modular y la posibilidad de programar en un lenguaje muy similar al Ladder moderno. Fue comercializado con el nombre de MODICON (Modular Digital Controller).

Figura 4

Uno de los primeros controladores lógicos programables (PLC)

Nota: Esta figura muestra el PLC MODICON 084. Tomada de (Unicrom, s.f.)

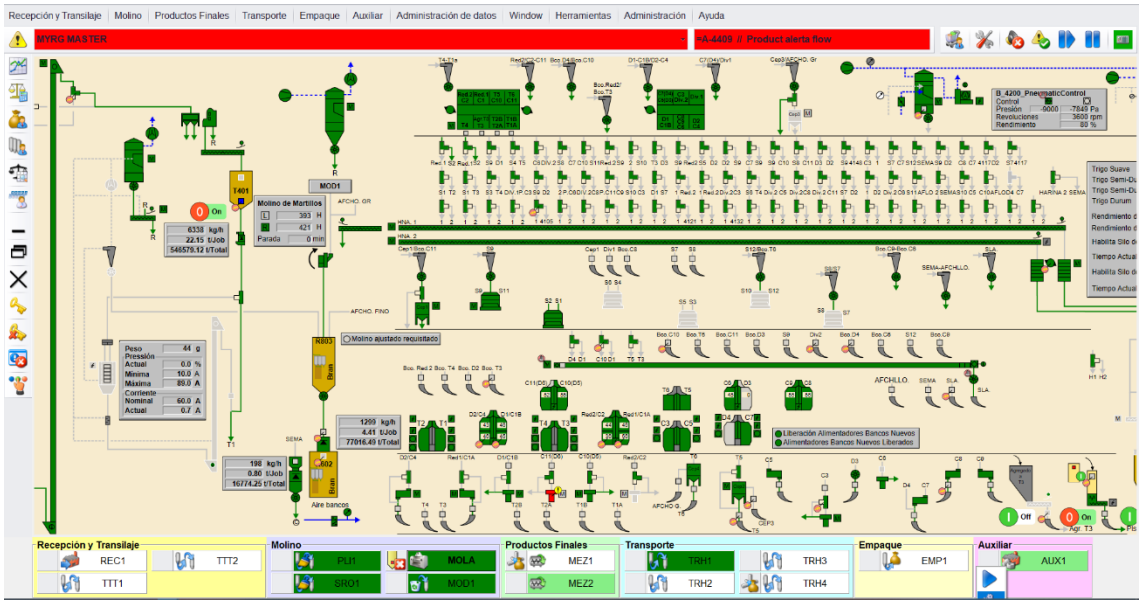
El Auge de los Microprocesadores y las Comunicaciones: La incorporación de microprocesadores a los PLCs en la década de 1970 supuso un gran avance. Los recientes creados PLCs eran más robustos y dinámicos para hacer cálculos matemáticos, dominar bucles de control y establecer comunicación con los demás equipos. Dando lugar a las comunicaciones industriales como Profibus o Ethernet industrial condujo al establecimiento de sistemas autómatas y a la integración de diversos sistemas control. (Agudelo, N., Tano, G., & Vargas, C. A. s.f.)

Sistemas SCADA y Supervisión de Procesos: Los sistemas de control supervisado para adquisición de información (SCADA) que llegaron a existir a finales de la década de 1980 permitieron que los procesos industriales sean controlados y monitorizados desde un solo lugar centralizado. El SCADA, con su interfaz gráfica y capacidades de análisis de datos, se convirtió en una herramienta esencial para la gestión de la producción en la industria alimentaria. (Tafur Morey, 2022)

En la Figura 5 se aprecia una captura de pantalla de un sistema SCADA de un proceso completo de molienda de trigo, donde se controla desde la admisión de trigo, preparación, molienda, y transferencias de producto final a las diferentes líneas de producción de una importante empresa alimenticia.

Figura 5

SCADA de un proceso industrial



Nota: Esta figura muestra el SCADA de un proceso industrial de molienda de trigo. Fuente propia.

1.2.2.3. Ventajas y Desventajas de la Automatización en la Industria alimenticia

Ventajas:

• **Mejor productividad:** La automatización permite reducir el tiempo de producción, así como la mano de obra y aumentar la productividad de los diferentes procesos industriales. (Tafur Morey, 2022)

• **Mejora de la calidad:** Los sistemas de control automatizados aseguran a los productos su uniformidad a veces afectada por errores humanos. (Tafur Morey, 2022)

• **Aumento de la seguridad:** La automatización minimiza los peligros de trabajo a los operarios con realizar labores peligrosas o repetitivas. (Tafur Morey, 2022)

• **Trazabilidad:** Los sistemas automatizados permiten registrar y guardar todos los parámetros de los productos que van fabricándose, por lo que lo que la trazabilidad está garantizada en su mayoría (Tafur Morey, 2022)

Desventajas:

• **Inversión inicial:** Implementar los sistemas que nos permitan automatizar implica pagar una gran inversión inicial. (Tafur Morey, 2022)

• **Dependencia a la Automatización:** Se requiere gente capacitada que siempre está allí para mantenerla. (Tafur Morey, 2022)

• **Reducción de Empleos:** La automatización puede resultar en la pérdida de fuentes de trabajo particularmente en funciones repetitivas. (Tafur Morey, 2022)

1.2.2.4. Tendencias Actuales en los Sistemas de Control:

• **Industria 4.0**: La cuarta revolución industrial está haciendo un cambio en la industria alimenticia con las tecnologías como el Internet de las Cosas (IOT), el Big Data, la inteligibilidad artificial, y la robótica.

• **Manufactura Aditiva:** La impresión 3D está provocando nuevas ventajas en la industria alimentaria y ha facilitado el establecimiento de productos con alta personalización y la producción dependiendo por demanda.

• **Realidad Aumentada:** La realidad aumentada y virtual se está centrando en entrenar a operarios, para construir fábricas y a controlar procesos.

1.2.3. Internet de las Cosas (IoT) en Procesos de Producción Alimentaria

El Internet de las Cosas (por sus siglas en inglés IoT) está revolucionando diversos sectores industriales, y la alimentaria no es una excepción. IoT permite la interconexión de dispositivos especialmente hacia Internet para obtener y analizar datos en tiempo real, dispone de mucho potencial para optimizar los procesos de producción, mejorando la eficiencia, la seguridad y la calidad de los productos (Aguilar Zavaleta, 2020).

1.2.3.1. Definición del Internet de las Cosas (IoT)

El Internet de las Cosas se define como una red global que conecta objetos identificables con capacidades de detección, actuación, procesamiento y comunicación con otros dispositivos, centro de datos o nubes computacionales empleando tecnologías de comunicación e información, con el propósito de obtener aplicaciones avanzadas e información productiva (Aguilar Zavaleta, 2020). En la Tabla 1 se resume las principales

características del IoT, orientado a la detección y accionamiento, las comunicaciones que intervienen, procesamiento de la información, entre otras.

Tabla 1

Principales características del IoT

Concepto	Descripción
Detección y actuación	Todo sistema IoT está conformado con sensores y actuadores para interactuar con el entorno.
Comunicación	Los equipos tecnológicos se comunican entre sí y con la nube mediante tecnologías como Wi-Fi, Bluetooth, modbus TCP, redes móviles, Lora WAN, etc.
Procesamiento	Los datos que se obtengan por los sensores son analizados para la toma de decisiones y activar los elementos actuadores en caso de requerir.
Aplicación	La información se presenta a los usuarios finales mediante interfaces gráficas en aplicaciones de software para dispositivos fijos o móviles.
Visión orientada a las cosas	Se centra en el rastreo de objetos mediante tecnologías como RFID.
Visión orientada al internet	Trata de la interacción de los equipos físicos a través de internet.
Visión orientada semánticamente	Resalta la importancia del procesamiento y análisis de la gran cantidad de datos generados por IoT.

Nota: La tabla presenta los principales conceptos de IoT, que consta de la detección, comunicación, procesamiento y aplicación de datos obtenidos de un proceso.

1.2.3.2. Antecedentes del IoT en la Producción Alimentaria

Diversos estudios e investigaciones previas han demostrado el potencial del IoT para transformar la industria alimentaria. A continuación, se presentan algunos ejemplos de la aplicación de esta tecnología en diferentes etapas de la cadena de producción:

- **Producción:** En la agricultura, el IoT se utiliza para monitorear las condiciones del suelo, el clima y el estado de los cultivos. Un ejemplo es el proyecto FOODIE, que utiliza una plataforma en la nube para brindar a los agricultores información actualizada y herramientas de gestión (Martínez & Mesías, 2021).
- **Transformación:** La denominada "fábrica inteligente" se beneficia del IoT para mejorar la eficiencia de los procesos de transformación de alimentos. El proyecto IoF2020 utiliza sensores y espectrómetros en bodegas para controlar la fermentación y la temperatura del vino (Prada Moreno, 2021).
- **Almacenamiento:** Los empaques inteligentes equipados con sensores permiten darnos información del estado de los alimentos y tomar una decisión antes de ser comidos. Un caso es el del uso de sensores de colores que reaccionan la fruta mientras pasa por una maduración (Prada Moreno, 2021).
- **Distribución:** El IoT permite el seguimiento de los productos durante su transporte, garantizando que se mantengan las condiciones óptimas de calidad y seguridad. El proyecto Big Wine Optimisation emplea sensores para monitorear de manera remota la temperatura, humedad e incluso detectar los golpes durante el transporte de vino (Prada Moreno, 2021).
- **Consumo:** El IoT facilita la interacción de los consumidores con los productos alimentarios. Los supermercados sin personal, donde los clientes hacen su compra a través de una aplicación móvil usando su celular, son un ejemplo de la aplicación del IoT en esta etapa (Prada Moreno, 2021).

1.2.3.3. Aplicaciones del IoT en la Producción Alimentaria

El IoT tiene muchas utilidades dentro de la industria agroalimentaria, para que se haga mucho más productivo y confiable. Algunas de estas aplicaciones incluyen:

- **Potenciar cultivos**: Sensores localizados en el campo captura información sobre la humedad del suelo, de temperatura, de radiación solar y de los nutrientes, de

modo que los agricultores se pueden tomar decisiones bien informadas sobre el riego, la fertilización y el tratamiento de plagas (Aguilar Zavaleta, 2020).

- **Control del clima en invernaderos:** Con IoT se automatiza el control de clima en los invernaderos, viendo el estado de la temperatura, la humedad e iluminación para un buen crecimiento de la planta (Martínez & Mesías, 2021).
- **Crianza de ganado:** Los sensores que se han colocado en los animales suelen registrar su lugar, estado de salud, además de facilitar la detección temprana de enfermedades y la mejora de la dieta (Martínez & Mesías, 2021).
- **Cadena de frío:** Con IoT, es posible efectuar la monitorización de la temperatura de alimentos mientras están en transporte y son refrigerados, de modo que los alimentos están en las óptimas condiciones para que se mantengas en buen estado (Martínez & Mesías, 2021).
- **Logística:** El IoT permite estar al día en los inventarios, organizar las rutas de los transportistas y hacer un monitoreo del proceso de entregas, para optimizar costos logísticos y disminuir perdidas.
- **Información al consumidor:** Los envases inteligentes, que disponen de sensores y códigos QR, pueden proporcionar a los consumidores finales información detallada sobre el origen, los ingredientes y la fecha de caducidad de los productos (Martínez & Mesías, 2021).

1.2.3.4. Desafíos y Oportunidades del IoT en la Producción Alimentaria

Toda aplicación de IoT en la industria alimentaria conlleva beneficios, como la mejora de la eficiencia de los procesos, aumentar la seguridad alimentaria, personalizar productos, etc., pero así mismo conlleva retos que se tienen que superar, como el costo que supone de implementar un proyecto IoT, salvaguardar los datos mediante

ciberseguridad, crear o modificar estándares que se adapten a IoT. Estos tópicos se resumen en la Tabla 2.

Tabla 2

Desafíos y Oportunidades que tiene IoT en la industria alimentaria

Desafíos del IoT	Oportunidades del IoT en la industria alimentaria
Costos: La instalación de sensores, comprobar las redes de comunicaciones y adquirir software pueden ser una importante inversión de dinero para las empresas.	Eficiencia y producción: La automatización de los diferentes procesos proveen un mejor uso de los recursos, por lo que se obtiene mayor productividad y eficiencia durante la producción alimenticia.
Seguridad de los datos: La gran demanda de datos de un sistema de IoT requiere fortalecidos sistemas de ciberseguridad con el objetivo de proteger la información valiosa de la empresa y del usuario.	Aumento de la seguridad alimentaria: Esto se hace vigilando de manera constante las condiciones de producción y distribución, cuyo objetivo es mejorar la seguridad alimentaria, limitando los riesgos de contaminación.
Carencia de estándares: La interoperabilidad entre los distintos elementos y plataformas IoT puede estar en contra de la unificación de los sistemas.	Mejores productos personalizados: El IoT ha permitido captar datos de las inclinaciones en preferencia de los consumidores, lo que permite a las compañías llegar a ofrecer unos productos más atractivos debido a su personalización.

Nota: IoT tiene sus desafíos, pero su aplicación en la industria alimentaria ofrece muchos beneficios en eficiencia, seguridad y el desarrollo de productos personalizados.

1.2.4. Plataforma Arduino Cloud y sus Aplicaciones en Control Industrial

La revolución del Internet de las Cosas (IoT) ha revolucionado la forma como comunicamos con el mundo. Y en el ámbito industrial no es la excepción. En este contexto, Arduino Cloud aparece como un servicio integrado para manejar y controlar el diseño industrial desde la conexión de los elementos hasta la visualización y analítica digital. A continuación, veremos información destacada sobre las potencialidades de conectividad y seguridad de Arduino Cloud. Pero así mismo sus desventajas y posibles limitaciones en la industria.

1.2.4.1. Arquitectura de Arduino Cloud

Arduino Cloud utiliza la arquitectura en la nube, de modo que es óptimos para tratar sistemas de IoT". Facilita la Conexión de dispositivos, visualización de variables en tiempo real de los datos y seguimiento de proyectos prácticamente en todo el mundo (Arduino, 2025). Esta arquitectura está constituida por los siguientes elementos fundamentales:

- Dispositivos: Son cualquier tipo de dispositivos físicos, como los sensores, actuadores, controladores, necesarios para su conectarse a la plataforma Arduino Cloud.
- Conectividad: A través de los protocolos de red como Wi-Fi, Bluetooth, MQTT, etc. Mediante los cuales se transmite los datos entre los equipos IoT con la nube.
- Plataforma alojada en la nube: Representa la base de Arduino Cloud, donde existen recursos en cuanto a almacenamiento de datos, procesamiento, análisis y visualización.
- Interfaz de usuario: Presenta una interfaz gráfica o de aplicación en el móvil que permita el acceso al registro, variables y administrar los dispositivos de manera remota (Arduino, 2025).

1.2.4.2. Capacidades de Conectividad y Comunicación

Arduino Cloud se integra con una amplia gama de dispositivos y protocolos de comunicación, brindando flexibilidad y adaptabilidad a diferentes entornos industriales. Algunas de las características de comunicación relevantes son:

- Opera con varios protocolos: Arduino Cloud usa protocolos comunes de la industria IoT, incluidos MQTT, HTTP y WebSockets, etc.
- Conexión Wi-Fi y Ethernet: Integrando los dispositivos a la nube, estos podemos enlazarnos a una red estándar sin problema.
- Enlace nativo a su nube: Arduino Cloud mantiene una conexión directa a dispositivos Arduino, lo que facilita la configuración y la realización de proyectos de diferente índole (Arduino, 2025).

1.2.4.3. Sistemas de Seguridad y Encriptación

Es muy importante la seguridad de los datos de cualquier plataforma IoT, sobre todo si se aplica al contexto industrial donde la sensibilidad de información y la integridad de los sistemas son importantes, Esta plataforma para uso industrial incluye:

- **Sistema alojado en la nube:** Arduino Cloud reposa sobre proveedores de servicio de red en la nube de uso conocido, como Amazon Web Services (AWS) y Google Compute Platform (GCP). Estos productos brindan una infraestructura fuerte que asegura que los Datos almacenados y tratados estén bien protegidos (Arduino Docs, 2024).
- **Segura Comunicación (TLS):** La conexión entre dispositivos y Arduino Cloud se transmite mediante TLS (Transport Layer Security), la cual garantiza la confidencialidad y la integridad de los datos transmitidos.
- **Elemento Seguro de Hardware:** Algunas placas de Arduino, como las de la familia MKR y Nano 33 IoT, incorporan un elemento seguro de hardware (por ejemplo, Microchip ATECC508A o ATECC608A). Este componente realiza funciones criptográficas esenciales, como:
    - ✓ Generación y almacenamiento seguro de claves privadas.
    - ✓ Autenticación de dispositivos mediante certificados únicos.

- ✓ Aceleración de operaciones criptográficas, mejorando el rendimiento y la seguridad.

- **Métrica de Aprovisionamiento:** En la configuración inicial, cada dispositivo crea una clave encriptada que queda guardada en la parte del hardware. Para después crear una Solicitud de Firma de Certificado (CSR) que se envían a Arduino para que se obtenga un certificado firmado. El mismo que hace que el dispositivo funcione con la nube estableciendo una conexión segura y que no se necesite credenciales de tipo tradicional, como usuario y contraseña (Arduino Blog ,2020).
- **Protección de Datos Sensibles en los Sketches:** Al compartir sketches (código de programación), es importante proteger la información delicada como credenciales Wi-Fi o claves API por lo que Arduino proporciona guías para almacenar de forma segura estos datos. (Arduino Docs, 2024).

1.2.4.4. Ventajas de Arduino Cloud en Aplicaciones Industriales

En la actualidad las plataformas IoT en la nube han cobrado relevancia por su relativa facilidad de desarrollo, capacidad de monitoreo remoto, entre otras. Es el caso de Arduino Cloud que tiene ventajas técnicas y económicas para su implementación en usos industriales. Se caracteriza por su facilidad de uso, escalabilidad y bajo costo. En la Tabla 3 se detallan estas características en detalle.

Tabla 3

Principales ventajas de Arduino Cloud en aplicaciones de control industrial

Característica	Descripción
Simplicidad y facilidad de uso	Arduino Cloud ofrece un interfaz fácil de usar incluyendo herramientas de desarrollo que simplifican creación, implementación y control de proyectos IoT, por lo que hasta para personas sin mucho conocimiento sobre la programación pueden usarlo.

Escalabilidad	La infraestructura en la nube permite extender los proyectos industriales de control desde nivel de prototipo, a aplicarlos a gran escala en producción industrial propiamente dicha.
Visualización y análisis de datos	Arduino Cloud tiene herramientas para observar datos en tiempo real, dashboards personalizados e históricos, por lo que se recibe información clave para la toma de decisiones y mejorar al proceso industrial
Bajo costo	La plataforma ofrece planes de suscripción flexibles y accesibles, lo que la convierte en una solución rentable para proyectos de control industrial.

Nota: Por costo y facilidad de uso Arduino Cloud es una buena opción para nuestro proyecto

1.2.4.5. Limitaciones de Arduino Cloud en Aplicaciones Industriales

Aunque Arduino Cloud tiene muchas prestaciones destacadas, también hay que analizar las cosas que se deben tener en cuenta al plantearla en aplicaciones industriales:

Dependencia exclusiva con Internet: Las funciones de Arduino Cloud se establecen con una conexión estable a Internet. La inexistencia de una cobertura de conexión puede debilitar funcionalmente el sistema de control, que por lo general es un factor crítico para su correcto funcionamiento.

Personalización limitada: Aunque Arduino Cloud cuenta con una plataforma fácil de usar, en caso de requerir aplicaciones más personalizadas puede ser una limitante. Por lo que se podría incluir la implementación de herramientas de desarrollo extra o la participación sobre un sistema parecido con otras herramientas de desarrollo.

1.2.4.6. Antecedentes y Aplicaciones en Control Industrial

Numerosos estudios y proyectos han validado la eficacia de IoT en aplicaciones de control industrial, en la Tabla 4 se describen dos ejemplos de proyectos donde se ha usado para el monitoreo de temperatura y humedad, así como para adquisición de datos.

Tabla 4

Ejemplos relevantes de aplicaciones IoT

Proyecto	Plataforma	Sensores	Comunicación	Plataforma en la nube	Aplicación
Sistema de Monitoreo Ambiental en Tiempo Real	Arduino UNO	DHT11 (Temperatura y Humedad)	Wi-Fi (ESP8266)	ThingSpeak	Monitoreo ambiental en tiempo real
Sistema de Adquisición de Datos y Control Industrial	Arduino Due	Sensores diversos	RS485, RS232, CAN	No especificado	Toma de datos y control para aplicaciones industriales

Nota: IoT ya trabaja en la industria con diversos tipos de comunicación

1.2.5. Uso del Controlador Lógico Programable (PLC) en la Industria Alimenticia

La industria de alimentos ha evolucionado mucho con los años, pero con ello conlleva una creciente necesidad de productos de mayor calidad, por ende, las industrias deben ir innovando constantemente especialmente en tecnología, por ello han incluido la automatización en sus procesos. Dentro de esto, está el Controlador Lógico Programable (PLC) que cuenta con múltiples funcionalidades como: estabilidad, robustez y capacidad de gobernar procesos complejos industriales.

1.2.5.1. Características Principales de los PLC:

Programabilidad: El PLC puede programarse en lenguajes simples, tales como el diagrama de escalera (Ladder), fácil de entender por técnicos que saben control eléctrico, esto permite aplicarlo a todo tipo de usos.

Robustez: Son equipos comprobados para entornos industriales, los PLC llegan a soportar ambientes hostiles, entre las que figuran vibraciones, temperaturas peligrosas y ruido eléctrico (Olortegui, s.f.).

Confiabilidad: Al ser equipos robustos los PLC brindan un nivel superior de confiabilidad y estabilidad, lo que ayuda a garantizar la continuidad de los procesos (Durán & Pinto, s.f.).

Versatilidad: Esto permite que con los PLC controlen todo tipo de dispositivos, especialmente motores, válvulas, sensores y actuadores, por lo que son más adaptativos a muchos procesos.

Comunicación: Los PLC, dependiendo su gama, pueden conectarse con diferentes redes industriales: es decir, supervisar y manejar en remoto los procesos (Olortegui, s.f.).

1.2.5.2. La implementación de PLC en la industria alimentaria ofrece una serie de ventajas:

Mejora de la calidad del producto: Los PLC permiten controlar con exactitud la temperatura, tiempo y presión del proceso, y la cual se traduce en una elevada consistencia y calidad del producto listo.

Aumento de la eficiencia: Se aplican en procesos donde haya tareas repetitivas y complejas, lo que permite a los operadores concentrarse en tareas más valiosas, contribuyendo mejor a la productividad y a la eficiencia en total (Olortegui, s.f.)

Mayor seguridad: Los PLC también pueden establecerse para introducir sistemas de seguridad que protejan a los operadores y al equipo cuando suceden defectos o situaciones peligrosas (Del Castillo Santana, 2022).

Reducción de costos: El uso del PLC en procesos se lo puede aplicar en sistemas que ahorren energía, materia prima y tiempo de producción, de modo que obtendrá importantes reducciones de costos operativos (Olortegui, s.f.).

Trazabilidad y registro de datos: En el PLC pueden registrarse exactamente las variables del proceso y eso facilita la trazabilidad y el análisis de datos para la toma de decisiones de mejoras de acción en dicho proceso (Durán & Pinto, s.f.).

1.2.5.3. Aplicaciones de los PLC en la Industria Alimentaria

Los PLC son usados en muchas aplicaciones dentro de la industria alimentaria, desde la manipulación de materias primas hasta el hasta el empaque del producto final. En la Tabla 5 se presenta otras aplicaciones donde su usan los autómatas programables.

Tabla 5

Principales aplicaciones de los PLC en la industria

Aplicación	Descripción
Control de procesos de pasteurización	Aplicaciones de regulación de temperatura, tiempo, y variables críticas para seguridad alimentaria y calidad del producto final.
Sistemas de envasado	Control de máquinas de llenado, de etiquetado y empacado para precisión y eficiencia, donde se usa con más frecuencia el control de movimiento
Control de temperatura en hornos y cámaras frigoríficas	Mantenimiento de temperatura adecuada los diferentes procesos industriales.
Control de líneas de producción	Organización y control de etapas de producción para optimizar el flujo de trabajo y tener eficiencia.
Sistemas de limpieza y desinfección (CIP)	Limpieza y desinfección mediante el uso de autómatas para garantizar la higiene y seguridad alimentaria.

Nota: El PLC lidera el control en la mayoría de industrias en el mundo

1.2.5.4. Tendencias Futuras en el Uso de PLC en la Industria Alimentaria

Integración con la Industria 4.0: Todos los PLC estarán conectados a su interacción con tecnologías como Internet Industrial de las Cosas (IIoT) e inteligencia artificial (IA) y el análisis de Big Data para ampliar aún más la eficiencia, la calidad y la seguridad de la producción de alimentos.

Interfaces hombre-máquina (HMI) más avanzadas: Las HMI serán más sencillas y rápidas de manipular y sus operadores tendrán un mayor control y de mejor vista sobre los procesos que se estén monitoreando.

Sistemas de control distribuido (DCS): Los sistemas DCS, que emplean diversos PLC en colaboración, serán más comunes en la industria alimentaria porque van a hacer posible un control más preciso y flexible de procesos complejos.

1.2.6. Protocolo de Comunicación Modbus TCP

Modbus TCP es una extensión al protocolo Modbus que funciona en redes Ethernet y ofrece comunicación entre equipos industriales como PLCs, sensores, módulos de entrada/salida y software SCADA. Este protocolo fue desarrollado por Schneider Automation para ampliar al clásico modbus serial, pero a través de TCP/IP que es usado en prácticamente la mayoría de las redes industriales modernas.

El protocolo Modbus TCP funciona con una arquitectura cliente/servidor. Dónde se establecen las peticiones del cliente y este recibirá respuestas del servidor. Lo que vendría a ser transacciones de comunicación. Cada transacción trabaja de manera independiente, lo que garantiza confiabilidad frente a interrupciones o errores.

Una de sus principales ventajas es que se puede trabajar con diversas conexiones en simultáneo, lo que facilita la integración de varios clientes y servidores en una misma red. A través el protocolo TCP se garantiza que la transmisión y la integridad de los mensajes sean confiables, esto en ambientes industriales es un tópico crítico.

El modelo de datos Modbus TCP se basa en cuatro tablas principales:

- **Entradas discretas (Input Discrete):** bits de solo lectura.
- **Salidas discretas (Coils):** bits de lectura y escritura.
- **Registros de entrada (Input Registers):** datos de 16 bits de solo lectura.

- **Registros de retención (Holding Registers):** datos de 16 bits de lectura y escritura.

El protocolo utiliza codificación big-endian, que trata de que el byte más significativo se transmite primero. Además, todas las solicitudes se envían a través del puerto TCP 502 y utilizan una estructura de mensaje estandarizada compuesta por un encabezado y el cuerpo de datos.

Entre las funciones más utilizadas en Modbus TCP se encuentran:

- **FC01 - Read Coils** (lectura de salidas digitales)
- **FC03 - Read Holding Registers** (lectura de registros analógicos)
- **FC05 - Write Single Coil** (escritura de una salida digital)
- **FC06 - Write Single Register** (escritura de un valor analógico)

1.2.7. Interfaces Hombre-Máquina (HMI) en la Industria Alimentaria: Principios de Diseño

En la industria de alimentos, donde priman la exactitud, la confiabilidad y la autonomía, las Interfaces Hombre-Máquina (HMI) marcan la diferencia entre las tareas de control y la gestión operativa de la industria. EL diseñó de estas interfaces ha de seguir principios básicos para que garantice la fluida interacción entre un operador y los sistemas automatizados.

1.2.7.1. Principios Ergonómicos y de Usabilidad

La ergonomía analiza la relación entre humanos y partes de un sistema para idear HMI para la industria de alimentos. Como se indica en la Tabla 6, la meta es diseñar interfaces dinámicas, fáciles de manejar y muy cognitivo para el personal que interviene en un proceso industrial (Udax, 2024).

Tabla 6

Criterios de ergonomía en las interfaces HMI

Criterio	Descripción
Facilidad de Uso	Las HMI deben ser fáciles de aprender, usar y recordar, presentando la información de manera clara y concisa.
Eficiencia	El diseño debe ser optimizado para un acceso rápido a funciones e información importante, con una navegación intuitiva.
Prevención de Errores	Se minimiza los errores mediante la validación de entradas, mensajes claros y teniendo mecanismos de seguridad.
Adaptabilidad y Personalización	Se puede establecer opciones personalizadas como el idioma, tamaño de fuente y contraste, brillo, etc. para comodidad y eficiencia.

Nota: La facilidad de uso es crítico para los operadores de HMI en una maquinaria

1.2.7.2. Consideraciones de Seguridad

La seguridad es primordial en el ámbito industrial, y la alimentaria no es la excepción, y el diseño de la HMI debe reflejar esta importancia.

Control de Acceso a los datos: Las interfaces hombre-máquina deben contar niveles de acceso a sus datos, para garantizar que solo el personal autorizado pueda acceder y ver la información del proceso.

Alarmas y Notificaciones: Las HMI deben proporcionar alertas claras y oportunas en caso de eventos críticos, como fallos del sistema, desviaciones en los parámetros del proceso o situaciones de emergencia, (Ajjingenieria, 2024).

1.2.7.3. Estándares Aplicables

Existen diversos estándares y normativas que rigen el diseño de HMI en la industria alimentaria. En la Tabla 7 se puede ver los más importantes.

Tabla 7

Estándares más relevantes que aplican a las interfaces HMI

Norma	Descripción
IEC 61508	Norma internacional para la seguridad funcional de los sistemas eléctricos, electrónicos y electrónicos programables relacionados con la seguridad.
ISO 13849	Norma internacional para la seguridad de la maquinaria relacionada con los sistemas de control.
FDA 21 CFR Parte 11	Reglamento de la FDA que establece los requisitos para los registros y firmas electrónicos en la industria alimentaria.

1.2.8. Protecciones eléctricas para motores y componentes

La industria demanda sistemas confiables para su optima producción, un elemento clave es el motor eléctrico, ya que por este equipo se realiza el movimiento de los productos, materia prima, etc. a fabricarse a lo largo de una cadena de producción. Por ello los motores eléctricos requieren sistemas de protección adecuados para garantizar su funcionamiento seguro y eficiente, evitando daños causados por sobrecargas, cortocircuitos y condiciones operativas adversas.

1.2.8.1. Tipos de protecciones eléctricas

Los motores eléctricos son equipos fundamentales en todo proceso industrial, por lo tanto, existe la necesidad de que cuenten con las protecciones eléctricas necesarias para que soporten condiciones anómalas internas o externas. En la Tabla 8 se detallan este tipo de protecciones.

Tabla 8

Tipos de Protecciones eléctricas para motores eléctricos

Tipo de Protección	Descripción	Ejemplos
Protecciones inherentes	Dispositivos integrados en el motor que protegen contra condiciones anómalas, como sobrecalentamiento.	- Detectores de temperatura por resistencia (RTD): Miden la temperatura del devanado. - Klixon: Dispositivo bimetálico que desconecta el motor ante temperaturas excesivas.
Protecciones no inherentes	Dispositivos externos al motor que protegen contra sobre corrientes y cortocircuitos.	- Relés térmicos bimetálicos: Desconectan el circuito ante sobre corrientes. - Fusibles: Se funden para interrumpir el flujo de corriente en cortocircuitos. - Disyuntores (breakers): Protegen contra sobre corrientes con apertura automática.

Nota: Mientras más protecciones eléctricas disponga una maquinaria es menos susceptible a fallos.

Consideraciones en la selección de protecciones

- Debe considerarse la potencia y tipo de motor.
- Los dispositivos deben cumplir con normativas internacionales como IEC, NEMA, y NEC.

- Factores de servicio, condiciones ambientales y métodos de arranque (Torres Carpio, 2013)

1.2.8.2. Sistemas de monitoreo de corriente

El monitoreo de corriente es esencial para detectar anomalías en el funcionamiento de los motores eléctricos y prevenir fallos prematuros.

Métodos de monitoreo de corriente

Monitoreo en tiempo real:

- Uso de sensores de corriente (transformadores de corriente - CT).
- Dispositivos de medición conectados a sistemas SCADA o PLCs.

Monitoreo de las magnitudes eléctricas:

- Medición de la corriente RMS para monitorear la carga del motor.
- Detección de armónicos y picos de corriente anormales.

Tendencias de mantenimiento predictivo

- Análisis de datos históricos para prevenir fallas.
- Implementación de estrategias de mantenimiento basadas en la condición.

Normas y estándares aplicables

- **NFPA 70 (NEC)**: Normativa eléctrica para instalaciones industriales.
- **ISO 50001**: Gestión de energía en sistemas eléctricos.
- **IEEE 112**: Pruebas de eficiencia en motores eléctricos (Torres Carpio, 2013)

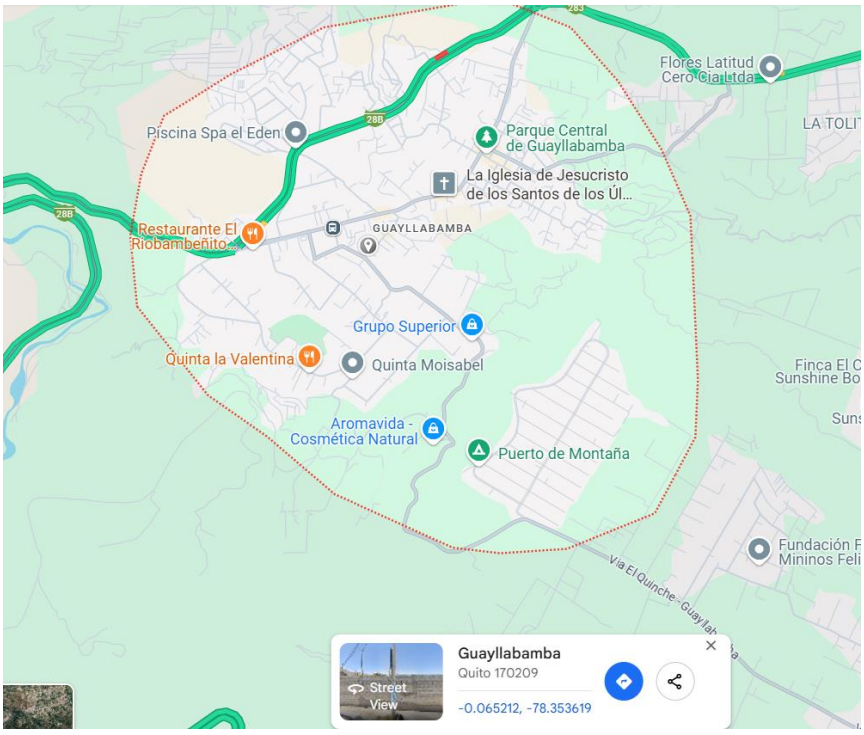
CAPÍTULO 2. METODOLOGÍA

2.1. Contexto de la investigación

La investigación se llevó a cabo en la heladería "Frosty Gamer", ubicada en la parroquia de Guayllabamba, en el Distrito Metropolitano de Quito, Ecuador. Guayllabamba es una parroquia rural situada al noreste de Quito como se indica en la Figura 6, reconocida por su clima cálido y su producción frutal. La heladería se encuentra en el centro de la parroquia, en una zona de alto tráfico peatonal y vehicular, lo que la convierte en un lugar ideal para implementar y evaluar nuevas tecnologías en la producción de helados.

Figura 6

Ubicación de la parroquia de Guayllabamba, DM de Quito



Nota: Esta figura muestra la ubicación geográfica de la parroquia de Guayllabamba. Tomado de Google Maps.

2.2. Diseño y alcance de la investigación

La investigación es experimental, esto significa que se implementará un sistema de automatización de sistema IoT y Arduino Cloud para adecuar la máquina de helado soft. Y los efectos de esta implementación se analizarán con pruebas de funcionamiento, estimación de eficiencia y estudios de datos recibidos en la nube.

Los campos de investigación incluyen el desarrollo del sistema de control, la integración con sistemas de análisis en remoto, la validación de los resultados y el entrenamiento del personal operativo. Se espera que esos hallazgos de este estudio sean repetibles en otras heladerías en condiciones similares.

2.3. Tipo y métodos de investigación

El presente estudio es de tipo cuantitativo, ya que se basa en la recolección y análisis de datos que se puedan medir para determinar el rendimiento del sistema implementado. Se obtendrán datos operativos, mediciones de eficiencia y desempeño con el fin de obtener resultados objetivos y replicables.

Los métodos de investigación empleados en este estudio son:

Deductivo: Se aplicarán teorías generales sobre automatización industrial e IoT aplicado a la modernización de la máquina de helado.

Analítico: Se estudiarán los equipos individuales del sistema para determinar su impacto en el rendimiento global.

Sintético: Se integrarán los hallazgos obtenidos en un modelo de solución que permita la replicabilidad en otros contextos.

2.4. Población y muestra

Debido a que el personal operativo es un grupo pequeño y es fundamental contar con información detallada sobre cómo interactúan con el sistema automatizado, se optó por realizar un censo para recoger sus opiniones y experiencias de manera precisa.

En relación con la máquina de helado, se llevarán a cabo mediciones periódicas para observar algunos aspectos importantes del funcionamiento, como los tiempos reales de

producción, cuánta energía consume y la temperatura final del producto. De esta forma, tendremos información concreta que permita realizar un análisis más objetivo.

2.5. Técnicas e instrumentos de recolección de datos

Para la recolección de datos se emplearán de manera exclusiva técnicas cuantitativas, con el objetivo de recibir información precisa y medible sobre el desempeño del sistema.

Las técnicas de recolección incluyen:

- **Mediciones operativas:** Se recopilarán datos de rendimiento de la máquina de helado, como tiempos de producción, temperatura, consumo energético y ciclos de operación.
- **Registros automáticos:** Se utilizarán datos obtenidos de la plataforma Arduino Cloud para un seguimiento continuo de variables clave como temperatura de las tolvas y cilindro, así como la corriente de consumo de los motores: agitador y compresor

Los instrumentos empleados serán:

- **Sensores y registros electrónicos**, como son los transductores de corriente a voltaje DC, y las sondas pt100 que proporcionarán mediciones precisas de variables operativas relevantes para la automatización.

2.6. Procesamiento de la evaluación: Validez y confiabilidad de los instrumentos aplicados para el levantamiento de información.

Para la automatización de la máquina de helado soft fueron necesarios realizar modificaciones al circuito de control y fuerza de dicha máquina, por lo que se hizo un análisis de cambio o adición de elementos a los circuitos.

2.6.1. Equipos para el circuito de Fuerza

La alimentación eléctrica para energizar la máquina es de 220VAC, monofásico con tierra, está dispuesto de una caja principal con un breaker bipolar de 32A, de igual manera se dispone de un regulador de voltaje marca BK 69BAR5000 de 5000 VA de potencia como se puede observar en la Figura 7, este equipo garantiza que el voltaje suministrado

a la máquina sea el óptimo (220VAC), ya que en medidas reales el voltaje que suministra la empresa eléctrica es de 195VAC.

Figura 7

Estabilizador de voltaje usado en la máquina de helado soft

Nota: Esta figura muestra el regulador de BK de 5000VA. Fuente propia.

Para la marcha de la máquina de helado soft intervienen dos principales equipos: el compresor y el agitador.

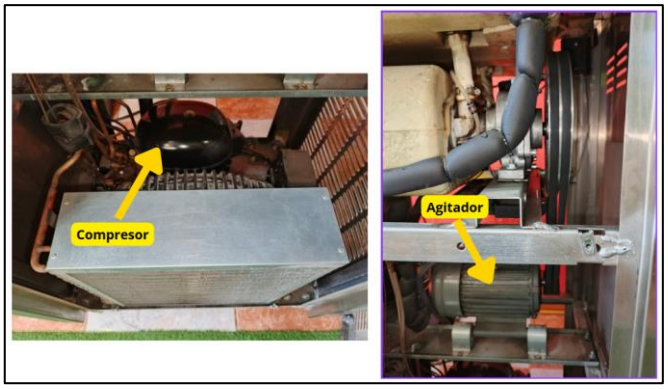
El compresor ejecuta el proceso de endurecer la base de helado por ciclo de refrigeración; así evapora el calor del cilindro apoyado de un ventilador eléctrico, mejorando así el rendimiento térmico.

Por otro lado, el agitador tiene doble sentido de trabajo: homogeniza la mezcla, y la airea, lo que es esencial para evitar la cristalización y obtener una textura cremosa

En la Figura 8 se puede ver la disposición real de los elementos como son el compresor con su ventilador que trabajan en conjunto, y también del motor agitador que pasa el movimiento mediante bandas de transmisión a las aspas colocadas dentro del cilindro.

Figura 8

Compresor y motor agitador de la máquina



Nota: Esta figura muestra el compresor y agitador de la máquina de Helado Soft. Fuente propia

2.6.2. Funcionamiento de la Máquina de Helado Soft: MODO LOCAL

El sistema automatizado de la máquina de helado soft como se muestra en la Figura 9, está diseñado para garantizar un funcionamiento eficaz y seguro mediante el control y monitoreo **local**, utilizando un controlador LOGO! Siemens. Este sistema gestiona las variables de entradas y salidas de la máquina, con ello se logra la correcta operación en los diferentes modos de funcionamiento.

Figura 9

Máquina de helado soft

Nota: Esta figura muestra la máquina de helado soft. Fuente propia

2.6.2.1 Entradas y Salidas del Sistema

Entradas digitales:

- Micro switch de palanca (activación del modo servir).
- Señales de contactos auxiliares de los guardamotores para protección de los motores (agitador y compresor).

Entradas analógicas:

- Sensor PT100 RTD para el cilindro del agitador.
- Sensor PT100 RTD para las tolvas de conservación.
- Dos transductores de corriente con salida de 0-10VDC, uno para el compresor y otro para el agitador.

Salidas digitales:

- Agitador.
- Compresor.
- Electroválvula de refrigeración del cilindro.
- Electroválvula de pre-refrigeración.
- Electroválvula del modo conservación.

Figura 10

Electroválvulas de paso de gas refrigerante

Nota: En la figura se muestra las 3 electroválvulas de refrigerante. Fuente propia

En la Figura 10 se puede observar la disposición de las electroválvulas que trabajan como actuadores de paso de gas de refrigeración para el cilindro, la pre-refrigeración en las tolvas y la conservación del producto en estas.

2.6.2.2. Modos de Operación

La máquina funcionará en distintos modos de operación dependiendo la aplicación que se requiera:

1. Modo "Servir Helado"

Este modo se activa mediante un micro switch de palanca, enviando una señal digital al controlador LOGO! Siemens. La secuencia de operación es la siguiente:

1. Al activarse el micro switch, se inicia el agitador, encargado de mantener la mezcla homogénea (cremosa y dura).
2. Al transcurrir 5 segundos, se enciende la electroválvula de gas principal, permitiendo la refrigeración del cilindro donde se almacena el producto.
3. Después de otros 5 segundos, se activan las dos electroválvulas adicionales, una para el pre-enfriamiento y otra para la conservación del producto en las tolvas.

Este proceso garantiza la correcta consistencia del helado antes de su dispensado, así como su correcto funcionamiento en las tolvas de almacenamiento.

2. Modo "Mantenimiento de Producto para Producción"

En este modo, el producto debe mantenerse a una temperatura de 3 a -11 grados Celsius. El sistema controla de manera automática la activación del motor del agitador, el compresor y la secuencia de electroválvulas para mantener el producto en condiciones óptimas de temperatura.

Se utilizan sensores PT100 RTD para monitorear y ajustar la temperatura del cilindro, activando o desactivando los elementos necesarios según las condiciones. Además, se verifica la corriente consumida por el agitador a fin de verificar la consistencia del helado, esto se mide con el transductor de corriente que va a una entrada analógica plc LOGO!.

3. Modo "Conservación de Producto" o “Keep Fresh”

Este modo permite conservar el producto en las tolvas, manteniéndolo entre **0 y 8 grados Celsius**, asegurando su calidad por un periodo prolongado. Este modo solo funciona cuando la máquina ya NO esté en el modo "Mantenimiento de Producto para Producción".

- Un **sensor PT100 RTD** en las tolvas mide la temperatura.
- El sistema se activa **cada 2 horas** durante aproximadamente **20 minutos**, o hasta alcanzar la temperatura establecida.
- Se activan el compresor y las electroválvulas necesarias para garantizar la refrigeración en los cilindros.

4. Modo "Limpieza"

Para garantizar la higiene del equipo, este modo permite la activación solo del motor del agitador, facilitando el revolver el agua con productos de limpieza dentro del cilindro, para así sacar todo residuo de base de helado y garantizar un cilindro inocuo para poder ingresar base de helado para la producción.

- El agitador opera durante 10 minutos, asegurando una limpieza eficiente.
- Si se requiere más tiempo de limpieza, el usuario puede volver a activar el proceso.

2.6.2.3. Sistema de Protección y Seguridad

El compresor y el agitador de la maquina están dotados de motores guardamotores, los cuales bloquean el funcionamiento en caso de sobrecarga. Además, se usan dos transductores de medición de corriente de consumo, con salida entre 0 10VDC, un transductor para el motor del compresor y otro transductor para el motor del agitador, lo que lo hace capaz de monitorizar con exactitud el consumo eléctrico.

2.6.3. Hardware a usar para el modo LOCAL

El microPLC LOGO! de Siemens en su versión 8.4 controlará todo el sistema de manera local, este equipo desde 1996, en sus primeras versiones, año en el que apareció en sus primeras versiones ha sido muy usado en todo tipo de sectores e industrias, demostrando su gran robustez y confiabilidad para una gran variades de aplicaciones. En la Tabla 9 se resume las características del LOGO! usado en el presente proyecto.

2.6.3.1. Características del MicroPLC LOGO! Siemens V8 8.4 12/24RC

Tabla 9

Logo Siemens 12/24 RC características

Especificación	Descripción
Rango de temperatura	Ampliado de -20 a 55°C sin condensación
Comunicación	Comunicación Ethernet integrada con soporte para Modbus TCP/IP, LOGO! Access Tool, y web server mejorado
Sincronización de fecha y hora	Sincronización automática de fecha y hora mediante NTP
Herramienta de acceso	Mejora en el uso de LOGO! Access Tool y funcionalidad web server mejorada con acceso desde navegadores móviles o PC
Software	LOGO! Soft Comfort V8.4 con soporte para Modbus TCP/IP, NTP, conversión de valores, y compatibilidad con múltiples sistemas operativos
Compatibilidad de hardware	Compatible con módulos de expansión FS:04, FS:03 y FS:02. Se mantiene retrocompatibilidad con versiones anteriores.
Capacidad de entradas y salidas	8 entradas digitales (de las cuales 4 pueden configurarse como analógicas de 0-10V) y 4 salidas de relé

Nota: De las principales características la que nos interesa para la comunicación REMOTA que va a tener nuestro sistema que se verá más adelante es Modbus TCP/IP.

2.6.3.2. Especificaciones del módulo de expansión LOGO! DM8 12/24R

La limitante del micro PLC LOGO! es que solo dispone de 4 salidas digitales a relé, por lo que ha sido necesario instalar el módulo de expansión DM8, de 4 entradas y 4 salidas digitales. En la Tabla 10 se adjunta en detalle sus características.

Tabla 10

Módulo expansión DM8 12/24RC características

Especificación	**Detalle**
Referencia	6ED1055-1MB00-0BA2
Alimentación	12/24 V DC
Entradas	4 entradas digitales
Salidas	4 salidas digitales tipo relé
Montaje	Montaje en carril DIN de 35 mm
Compatibilidad	Compatible con los módulos lógicos LOGO! 8 de Siemens

Este módulo permite ampliar las capacidades de entradas y salidas de los del LOGO! 8, facilitando la integración en sistemas que requieren mayor número de señales digitales.

2.6.3.3. Siemens LOGO! AM2 RTD

Para medir la temperatura en el cilindro y las tolvas de producto es recomendable usar el módulo de expansión para ese propósito, ya que no será necesario instalar equipo adicional como transmisores, este módulo soporta directamente dos entradas de sondas RTD. En la Tabla 11 se detallan sus características principales.

Tabla 11

Módulo de expansión analógico usado para la medición de temperatura

Especificación	Detalle
Referencia Siemens	6ED1055-1MD00-0BA2
Descripción	Módulo de expansión LOGO! AM2 RTD para LOGO! 8
Alimentación	12/24 VDC
Entradas Analógicas	2 entradas para sensores RTD (Pt100 o Pt1000)
Compatibilidad	Compatible con unidades LOGO! 8

Nota: Se usarán las dos entradas usando las direcciones AI3 y AI4.

2.6.3.4. Especificaciones del Siemens LOGO! TD

Para poder controlar de manera local y modificar los parámetros de la máquina de helado, la pantalla HMI LOGO! TD es una gran opción, ya que amplía la pantalla de LOGO! y dispone de 4 teclas de función que se las puede programar como entradas digitales. En la Tabla 12 se detallan sus especificaciones.

Tabla 12

Especificaciones de la interfaz de HMI LOGO! TD Siemens

Especificación	Detalle
Descripción	Visualizador de texto LOGO! TD de 6 líneas con 3 colores de fondo
Alimentación	24 VDC
Pantalla	6 líneas de texto con retroiluminación en 3 colores
Puertos Ethernet	2 puertos Ethernet para comunicación y programación

Especificación	Detalle
Teclas de función	4 teclas de función programables para interacción con el usuario
Compatibilidad	Diseñado para integrarse con los módulos lógicos LOGO! 8 de Siemens

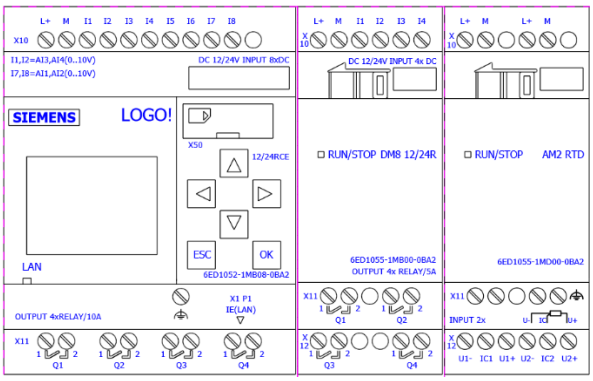
Este dispositivo permite una interacción sencilla y eficiente con los controladores lógicos programables LOGO! 8, facilitando la visualización y manipulación de datos en aplicaciones industriales.

2.6.3.5. Disposición de LOGO! con sus dos módulos.

Como se indica en la Figura 11 el micro PLC LOGO! se amplía con dos módulos adicionales mediante el bus lateral derecho. Estos y el PLC se alimentan de 24VDC conectando L+ y M. Las salidas de relé controlan el agitador, compresor y electroválvulas, mientras que las entradas digitales reciben señales del microswitch y dos guardamotores. El módulo analógico AM2 RTD mide la temperatura con sensores PT100 en el cilindro y el agitador, asegurando condiciones óptimas del producto.

Figura 11

Disposición de LOGO! Siemens con sus módulos



Nota: Esta figura muestra el montaje de LOGO! con sus módulos. Fuente propia.

2.6.3.6. Asignación de entradas y salidas del sistema.

Entradas Digitales

- **I1:** Microswitch de la palanca de servir helado
- **I4:** Guardamotor activado del compresor
- **I5:** Guardamotor activado del agitador

Entradas Analógicas

- **I7 (AI1):** Sensor de corriente del agitador (0-10VDC)
- **I8 (AI2):** Sensor de corriente del compresor (0-10VDC)
- **AI3 (Módulo RTD):** Sonda de Temperatura Cilindro
- **AI4 (Módulo RTD):** Sonda de Temperatura Tolvas

Salidas Digitales

- **Q1:** Agitador
- **Q2:** Compresor
- **Q5:** Electroválvula de refrigeración del cilindro
- **Q6:** Electroválvula de pre-refrigeración
- **Q7:** Electroválvula de conservación de producto

2.6.3.5. Cálculo de la corriente de fuente de alimentación de 24VDC.

En base a la a fórmula de la Ley de Watt y que dado que la mayoría de los fabricantes de PLCs y accesorios dan como dato la potencia de consumo de sus equipos:

Potencia(W)=Voltaje(V)×Corriente(A)

Corriente(A)= Potencia(W)/ Voltaje(V)

En la Tabla 13 se coloca el cálculo de cada equipo usado en la máquina de helado

Tabla 13

Cálculo de la corriente por medio de la ley de Watt

Equipo	Voltaje (V)	Potencia (W)	Corriente (A)
LOGO! 8	24V	3.5W	0.146A
DM8 12/24R	24V	1.2W	0.05A
AM2 RTD	24V	0.3W	0.0125A
Pantalla LOGO! TD	24V	3.5W	0.146A
Bobinas de dos contactores en DC	24V	20W	0.832A
Total	24V	**28.5W**	**1.18A**

Nota: Los valores de consumo de cada equipo fueron obtenidos de Siemens (2021) y Schneider Electric (2020).

En base a este consumo y por futuras conexiones de otros equipos en DC, selecciona la fuente LOGO! Power de 2.5A.

Como se puede ver en la Figura 12, esta fuente de voltaje DC se la puede colocar en riel din y soporta voltajes de entrada de 100VAC a 240VAC.

Figura 12

Fuente de alimentación estabilizada

Nota: Característica de voltaje de entrada: AC 100-240 V, salida: DC 24 V / 2,5 A. Fuente propia

2.6.4. Funcionamiento de la Máquina de Helado Soft: MODO REMOTO

Una vez establecido el funcionamiento correcto en modo LOCAL en donde se compruebe que la máquina funciona correctamente en sus diferentes modos de operación es necesario localizar que variables quiero publicar en Arduino Cloud. El "puente" entre LOGO! Siemens y Arduino Cloud será el Arduino NANO ESP32, esta placa está basada en el microcontrolador ESP32-S3, lo que lo hace una opción ideal para proyectos IoT. En la Tabla 14 se puede ver sus especificaciones más relevantes.

Tabla 14

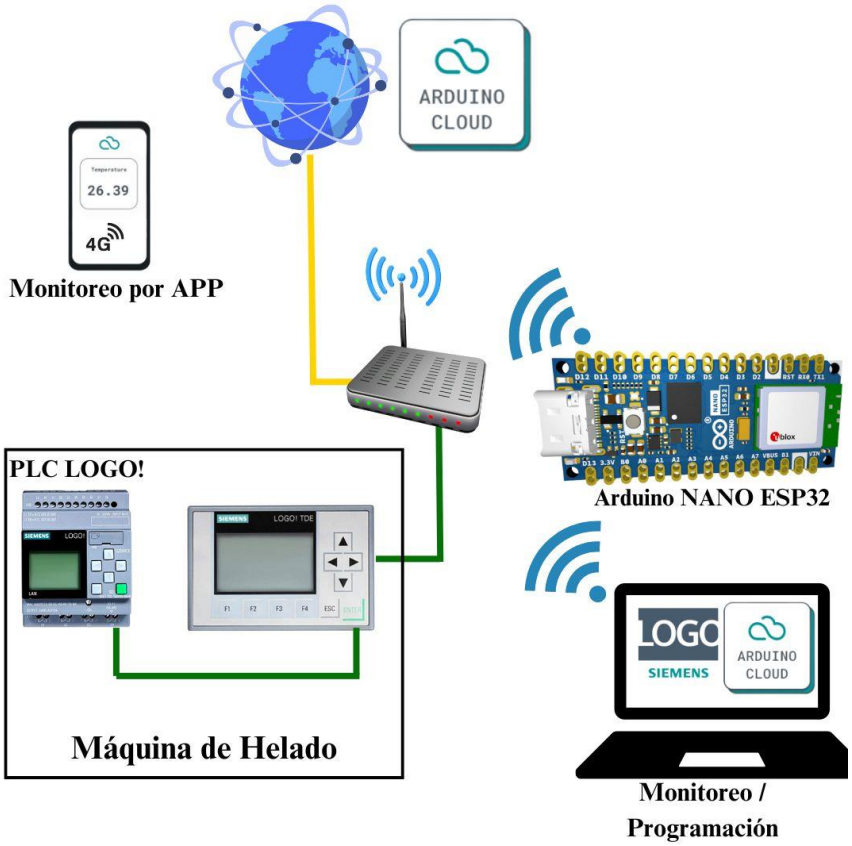
Especificaciones clave del Arduino Nano ESP32

Característica	**Especificación**
Microcontrolador	ESP32-S3 (dual-core, 240 MHz)
Memoria	16 MB Flash, 8 MB PSRAM
Conectividad	Wi-Fi 2.4 GHz, Bluetooth LE 5.0
GPIOs	23 pines (I/O digitales/analógicas)
Entradas analógicas	8 canales ADC de 12 bits
Interfaces	UART, SPI, I2C, I2S, CAN, USB
Alimentación	5V (USB-C), 3.3V lógica
Tamaño	45 x 18 mm
Programación	Arduino IDE, MicroPython, ESP-IDF

2.6.4.1. Monitoreo y Control Remoto de Máquina de Helado con PLC LOGO! y Arduino Nano ESP32 a través de Arduino Cloud

Figura 13

Esquema de conexión de equipos del sistema de monitoreo remoto



La Figura 13 muestra el sistema de monitoreo y control remoto para la máquina de helado, utilizando el PLC LOGO! Siemens y un Arduino Nano ESP32, conectados a través de Modbus TCP/IP mediante un router Wi-Fi.

- El PLC LOGO! controla la máquina de helado y envía datos al ESP32.
- El **ESP32** transmite estos datos a **Arduino Cloud**, permitiendo el monitoreo remoto desde una aplicación móvil y desde la computadora.
- Desde una computadora se puede programar y supervisar el sistema.

2.6.4.2. Software y pasos necesarios para realizar el enlace entre LOGO! y Arduino NANO y monitoreo para Arduino Cloud.

1. Configuración del LOGO! como cliente

- Definir el Arduino Nano ESP32 como servidor con una dirección IP estática en la red local, por ejemplo, 192.168.0.100.
- Crear bloques de función en **LOGO! Soft Comfort** para enviar solicitudes Modbus a la IP del ESP32.
- Establecer parámetros como:
    - Puerto: 502 (predeterminado de Modbus TCP).
    - Dirección de registros para lectura/escritura.
    - Tiempo de consulta y reintentos en caso de fallo.

2. Configuración del Arduino Nano ESP32 como servidor Modbus TCP

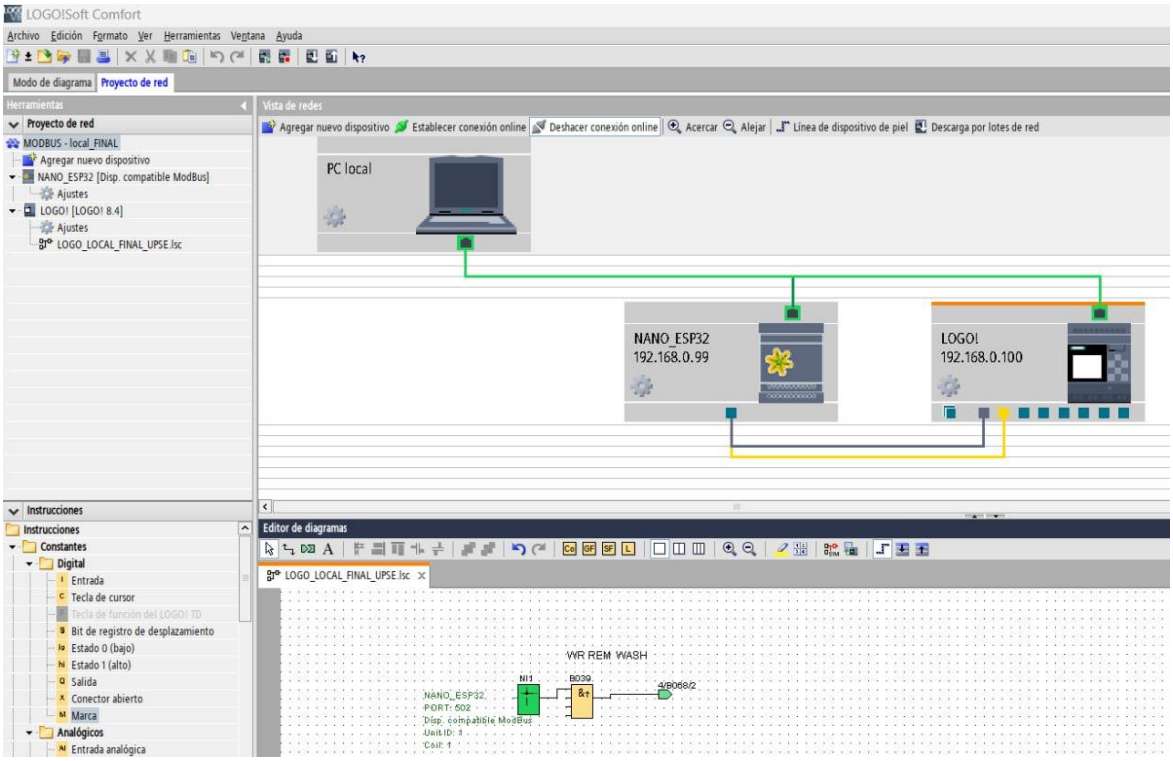
- Programar el ESP32 usando **Arduino IDE** con la biblioteca ArduinoModbus para manejar las solicitudes Modbus, así como debe estar en la misma subred, ejemplo ip a NANO ESP32: 192.168.0.99.
- Configurar el Arduino NANO ESP32 para comunicarse con LOGO!, proporcionando valores de sensores e interactuando con comandos de control
- Asignar registros Modbus en el ESP32 para representar variables como:
    - Entradas digitales (coils).
    - Entradas analógicas.
    - Registros de almacenamiento (holding registers) para datos de temperatura, corriente, etc.

La configuración en el software de programación de LOGO! Siemens debe darse en primer lugar en la creación del proyecto de red, como se muestra en la Figura 14, donde se asigna la dirección IP del LOGO!, así como el ingreso a la red del equipo con el que va a interactuar, en este caso el Arduino nano ESP32, que previamente se debe asignar su IP en el software de la placa. Estas direcciones IP deben estar en la misma subred. Se une mediante las líneas verdes, que representa la capa física Ethernet desde la computadora,

la placa Arduino y LOGO!. El medio de conexión será vía Wifi ya que soporta de forma nativa la placa ESP32 de Arduino.

Figura 14

Diseño de la red en LOGO! Soft Comfort



Nota: El medio de comunicación es vía Wifi mediante la red local.

El LOGO! enviará peticiones periódicas al ESP32 para consultar valores de sensores o enviar comandos de control.

Desde Arduino Cloud se puede visualizar el estado en tiempo real a través de dashboards y aplicaciones móviles.

CAPÍTULO 3. RESULTADOS Y DISCUSIÓN

En relación con el objetivo de implementar un sistema de automatización para la máquina de helado soft, integrando tecnologías de control local, monitoreo en la nube y gestión remota de la producción mediante Arduino Cloud, con el propósito de mejorar su eficiencia, funcionalidad y adaptabilidad a las demandas actuales del mercado, este capítulo presenta y analiza los resultados obtenidos.

La máquina de helado de soft funcionaba bajo un sistema obsoleto, ya que había limitaciones significativas, tales como la ausencia de conectividad a la nube mediante internet, condiciones de conservación limitadas del producto en las tolvas, nulas protecciones eléctricas para sus motores y tampoco no disponía de un manejo remoto y supervisión de la producción. Estos problemas tenían implicaciones en la productividad operativa, la calidad y consistencia del producto y la permanencia del sistema. La solución implementada, usa técnicas de Internet de las Cosas (IoT) y un controlador lógico programable (LOGO! Siemens), busca mejorar la operación de la máquina introduciendo control local y monitoreo remoto, para eso utiliza Arduino Cloud como su plataforma en la nube.

En concordancia con los objetivos específicos, los resultados que se presentan incluyen el desempeño de los diferentes modos operativos de la máquina como son: modo "Servir Helado", "Mantenimiento de Producto", "Conservación de Producto" y "Limpieza", así como el análisis de las variables operativas monitoreadas a través de la plataforma remota mediante un dashboard intuitivo para computadora y para el celular.

Además, se evalúa la eficiencia energética del sistema, la mejora en la calidad del producto final y el cómo impacta la automatización en la experiencia del operador.

El análisis consiste en una comparación de los indicadores de desempeño entre el antes y después de la implementación para demostrar cómo la modernización puede llevar a ahorrar costos operativos, calidad ambiental y competitividad en el ámbito industrial También se analizan los resultados respecto de los estudios comentados en el marco teórico, evidenciando la contribución de este proyecto a la aplicación de tecnologías IoT en la industria alimentaria.

Por último, esta parte presenta también la propuesta de mejora en base a las conclusiones, a mejorar la funcionalidad del sistema y buscar nuevos usos en situaciones similares en la industria. Por lo tanto, los resultados y el debate no solo demuestran la hipótesis desarrollada, sino que también establecen las bases para investigaciones y desarrollos posteriores en esta disciplina.

3.1. Resultados de la Implementación

El sistema automatizado implementado en la máquina de helado soft se apoya en el controlador lógico programable **LOGO! de Siemens 8 V8.4**, el cual gestiona sensores y actuadores esenciales para su funcionamiento. La programación se realizó mediante el software **LOGO! Soft Comfort**, utilizando diagramas de bloques funcionales que permiten una configuración sencilla, escalable y de fácil mantenimiento.

Modos de Operación

3.1.1. Modo "Servir Helado"

En la Figura 15 se observa la pantalla que muestra TD LOGO! en “MODO HELADO”, en este modo el equipo tiene esta secuencia de funcionamiento:

- Activación por micro switch de palanca.
- Operación del agitador, compresor y la activación secuencial de electroválvulas para refrigeración del cilindro
- Se mantiene así por 3 minutos, se apaga automáticamente en secuencia inversa, es decir, se apagan primero las electroválvulas, luego el compresor y finalmente el agitador

3.1.2. Modo "Mantenimiento de Producto listo para servir"

- Mantiene el producto entre +3 y -11°C con encendido automático del compresor y agitador.
- Sensor de corriente analógico para medir consistencia y protección de los motores compresor y agitador.

La diferencia entre estos dos modos es que en el de “servir helado”, se activa el sistema de refrigeración incluido el agitador, la palanca lo que hace es expulsar el helado para la venta, en cambio en “**"Mantenimiento de Producto listo para servir"** el sistema hace

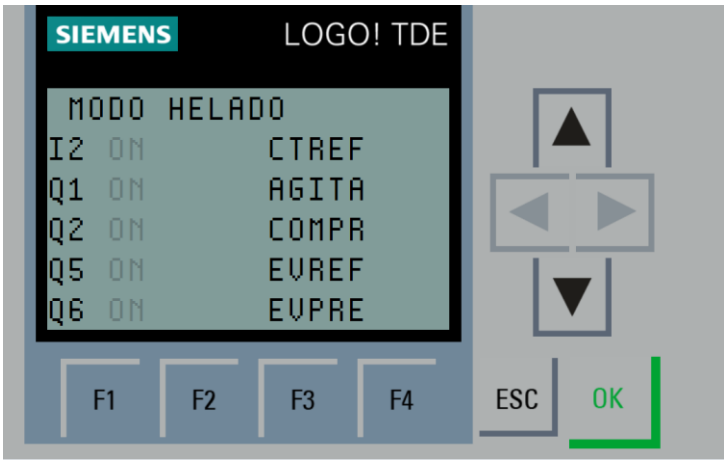
que el producto se mantenga entre +4 a -11 grados centígrados, encendiéndose a los 4 grados y apagándose a los -11 grados.

Por ende el sistema de refrigeración y agitación trabajan en estos dos modos (servir y mantener el producto)

Las temperaturas pueden ser modificadas a voluntad según la necesidad de producción o tipo de producto a servir. Se puede ver a más detalle en los ANEXOS, en la parte del Manual de Usuario.

Figura 15

Vista desde LOGO! Web Server en modo para servicio de "helado listo"



Nota: Se muestra el estado de las entradas y salidas en "helado listo". Fuente propia.

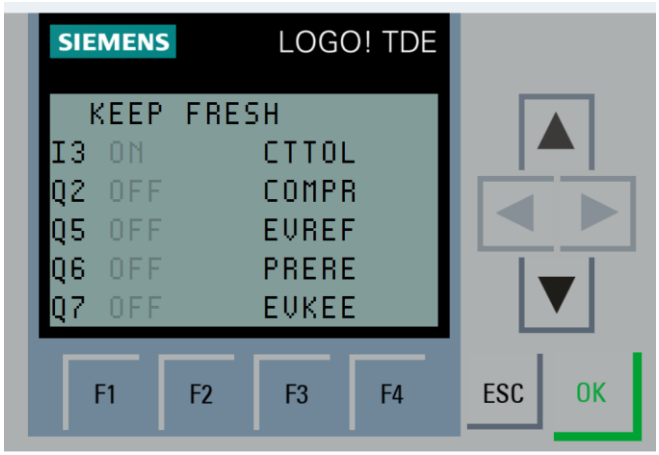
3.1.3. Modo "Conservación de Producto" o "KEEP FRESH"

- Mantiene la mezcla en tolvas entre 0 y 8°C.
- Se activa cada 2 hora durante 10 minutos o hasta alcanzar la temperatura deseada.

En la Figura 16 se muestra la pantalla en modo de conservación, mostrando además el estatus de las entradas y salidas que intervienen en modo keep fresh.

Figura 16

Vista desde LOGO! Web Server en modo para servicio de "keep fresh"



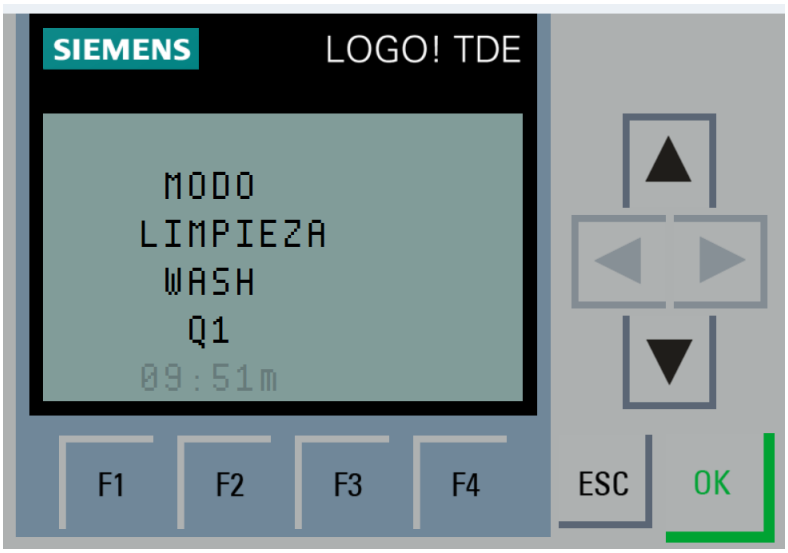
Nota: Se muestra el estado de las entradas y salidas en modo "keep fresh". Fuente propia.

3.1.4. Modo "Limpieza"

- Activa el agitador durante 10 minutos para la circulación de agua y productos de limpieza. En la Figura 17 se puede ver lo que muestra la pantalla TD en este modo.

Figura 17

Vista desde LOGO! Web Server en modo para servicio de "limpieza"



Nota: Se muestra el estado de la salida en modo "limpieza" o "wash". Fuente propia.

3.1.5. Configuración de Modos mediante LOGO! Soft Comfort

El sistema fue programado para permitir la selección de los modos de operación a través del panel **LOGO! TD**, utilizando las teclas funcionales:

- **F1 - Wash** (Modo Limpieza)
- **F2 - Modo Helado** (Modo Servir Helado)
- **F3 - Keep Fresh** (Modo Conservación de Producto)
- **F4 – Set Temperaturas** (Configurar temperaturas del cilindro y tolvas)

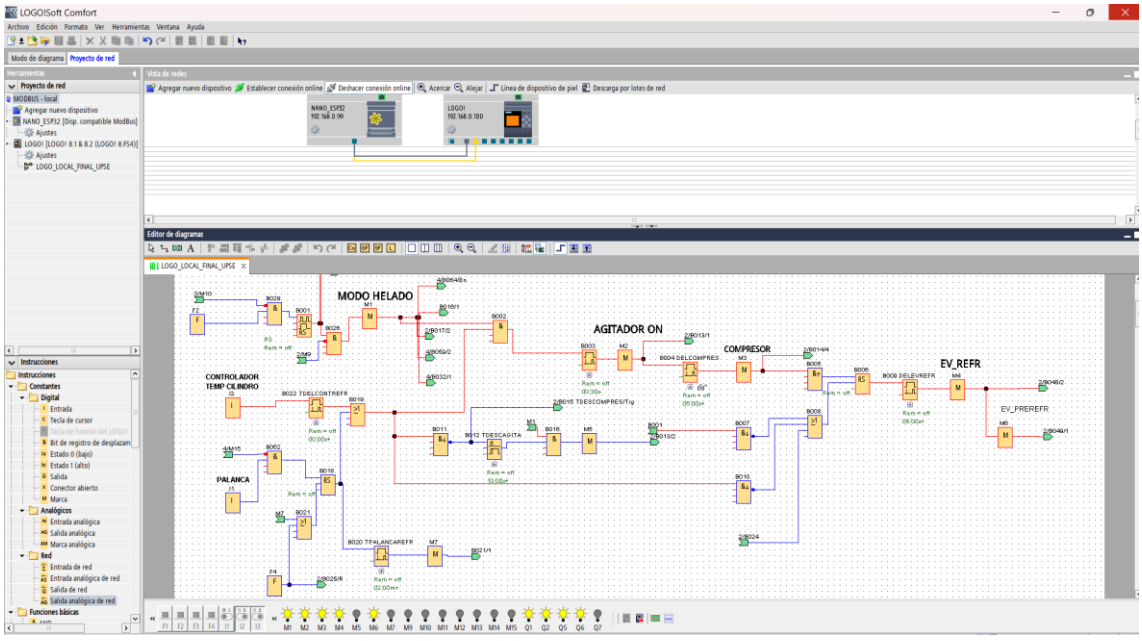
Estos modos se pueden ir verificando en el software en modo ONLINE en LOGO! Soft Comfort como se aprecia en la Figura 18.

3.1.6. Seguridad para el compresor

Si el compresor permanece encendido más de 20 minutos sin interrupción, el sistema activará la alarma y procederá a desactivar todos los componentes para evitar sobrecargas y daños en el sistema.

Figura 18

Vista de LOGO! Soft Comfort en modo ONLINE



Nota: Se muestra en el software LOGO! Soft Comfort con el funcionamiento de la máquina. Fuente propia.

3.2. Integración con Arduino Cloud

El **LOGO! Siemens** se comunica con un **Arduino Nano ESP32**, permitiendo:

- Monitoreo remoto de temperatura, consumo de corriente y estado del sistema.
- Cambio de modos de trabajo de manera remota, trends y alarmas.
- Control remoto mediante dispositivos móviles y PC.

3.2.1. Pasos para conectar una variable de LOGO! y verla en Arduino Cloud

Se realiza la prueba de ver el dato VW0 de LOGO! en Arduino Cloud usando el Arduino Nano ESP32 como intermediario de conexión.

3.2.1.1. Colocar los equipos en la misma subred:

Se asigna las IP a los equipos:

IP fija para LOGO!: 192.168.0.100

IP fija para Arduino Nano ESP32: 192.168.0.99

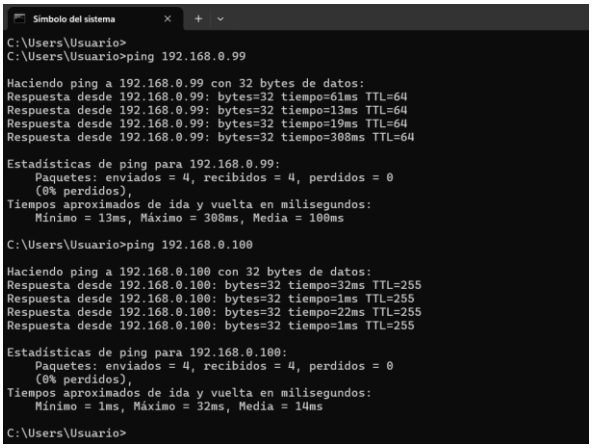
La IP de LOGO! se asigna a través de su teclado, en el menú red, y la IP del Arduino NANO ESP32 desde Arduino IDE.

3.2.1.2. Pruebas de RED

Haciendo ping se tiene conexión optima según la Figura 19, con ello se valida que la conexión física está establecida, y que todos los equipos se comunican entre sí.

Figura 19

Ping a LOGO! y Arduino NANO ESP32



Nota: Se muestra la IP de Arduino NANO asignada en Arduino IDE. Fuente propia.

3.2.1.3. Creación de la red modbus TCP

En LOGO! Soft se habilita el modbus como cliente y el NANO por consiguiente queda como servidor como se muestra en la figura 20.

Figura 20

Conexión Modbus TCP en LOGO! Soft Comfort

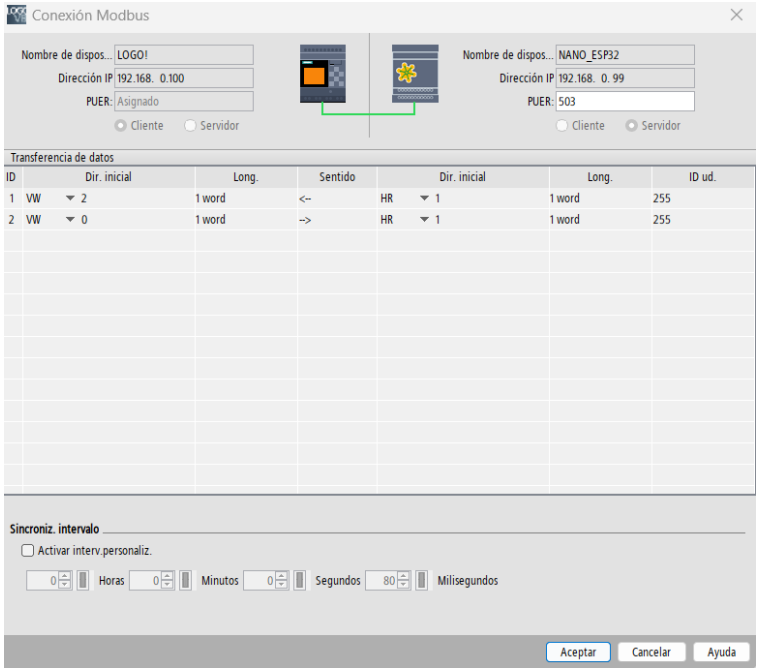
Nota: Se muestra la Conexión Modbus para el intercambio de datos. Fuente propia.

3.2.1.4. Configuración de la conexión Modbus

Se realiza la prueba de comunicación en los dos sentidos con de dos “word”, una por sentido de transmisión, como se indica en la Figura 21.

Figura 21

Parametrización para la conexión modbus



Transferencia de Datos (Modbus Holding Registers - HR). Fuente propia

a) LOGO! lee el valor de **HR 1** del Arduino y lo almacena en **VW2**.
b) LOGO! envía el valor de **VW 0** al Arduino y lo escribe en **HR 1**.

Vamos a ver b: LOGO! envía el valor de VW0 al Arduino y lo escribe en HR

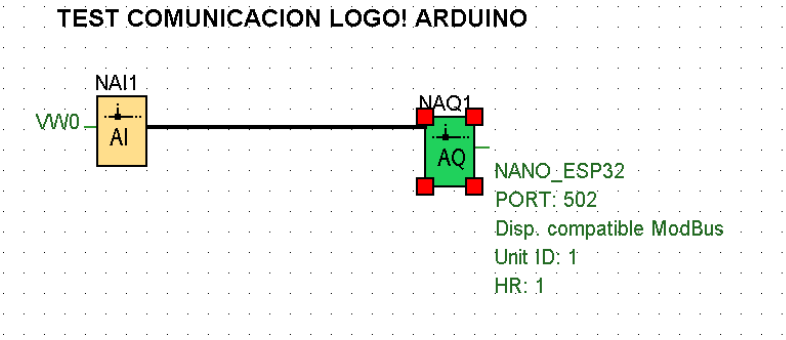
La asignación de las demás variables se hacen conforme la necesidad de monitoreo en Arduino Cloud.

Se realiza la conexión del bloque NAI1 (entrada de red analógica 1) en LOGO! VW0 y la vamos a escribir a la NAQ1 (salida de red analógica 1), al ser holding registers en MODBUS se le asignará la dirección 40001, (Figura 22).

Al ser valores analógicos que se pueden leer y escribir, lo que los hace útiles para enviar y recibir datos entre el LOGO! y el ESP32.

Figura 22

Conexión de bloques para la transferencia de datos



Nota: Bloque de pruebas para el intercambio de datos por modbus TCP. Fuente propia.

Cargamos a LOGO! el programa de la máquina con esta configuración.

3.2.1.5. Configuración de Arduino NANO ESP32

Registro de Arduino Nano ESP32 en Arduino Cloud

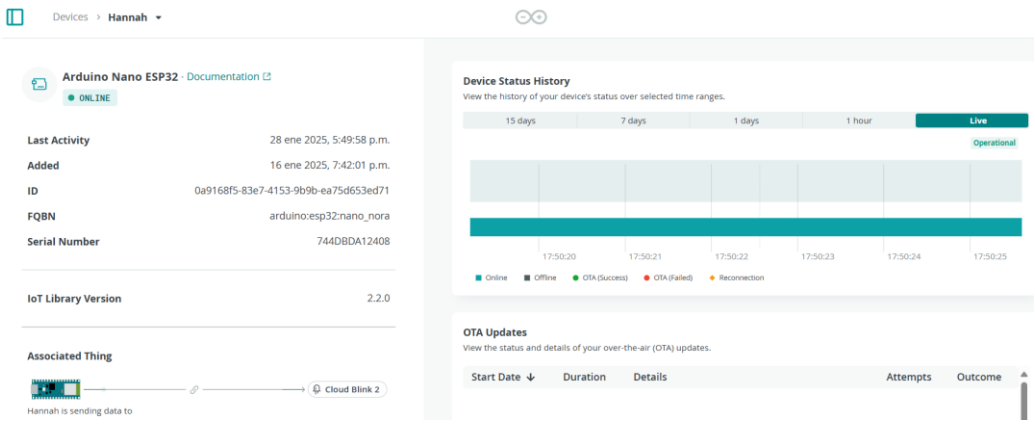
1. Crear Cuenta y Preparar Software
    - Accede a Arduino Cloud e inicia sesión.
    - Se Instala Arduino Create Agent en la computadora para detectar la placa mediante el puerto USB

2. Agregar la Tarjeta

   - Conectar el Nano ESP32 por USB.
   - Ira a Devices > Add Device y selecciona Arduino Nano ESP32. Aparecerá en modo ONLINE como en la Figura 23.

Figura 23

Registrar Tarjeta Arduino Nano ESP32 en Arduino Cloud

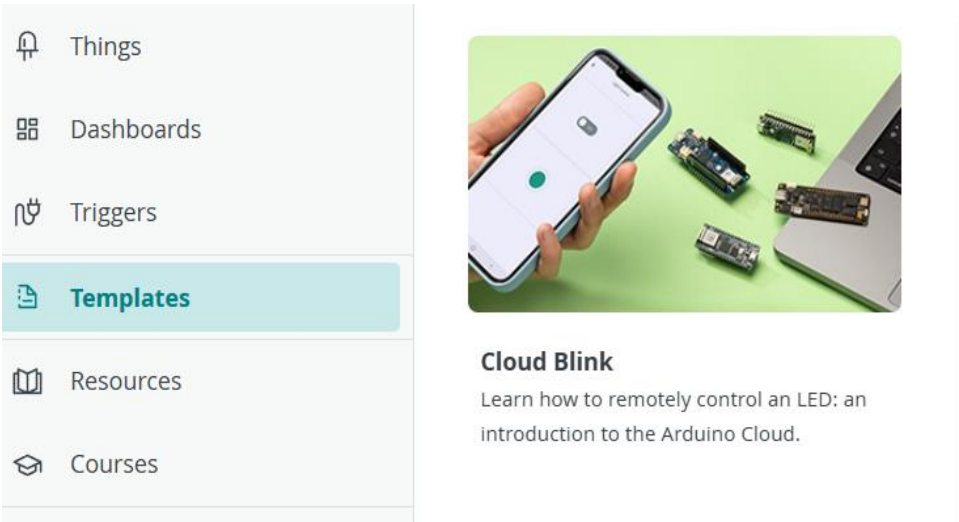


Nota: En la figura se muestra online la tarjeta ESP32. Fuente propia.

3. Usar el TEMPLATE Cloud Blink (Figura 24), y cargar estando conectada la tarjeta ESP32

Figura 24

Menú Template "cloud blink"

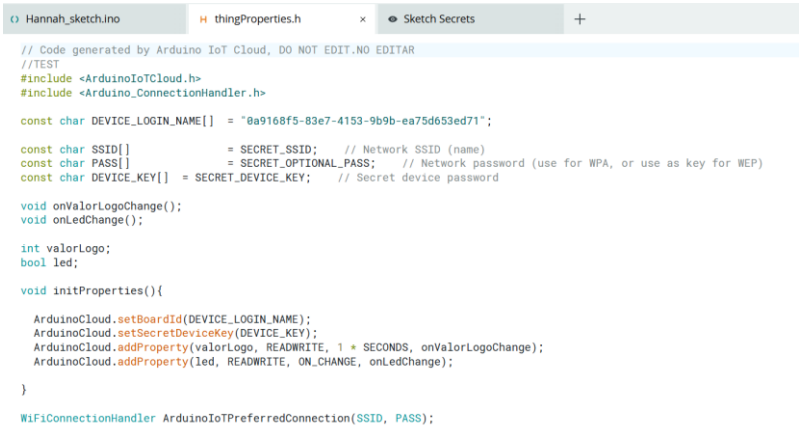


Nota: En la figura se muestra el template o plantilla Cloud Blink. Fuente propia.

El propósito de hacer esto es que se crea el código “thingProperties.h” (Figura 25), para bajarlo desde el sketch de Cloud a la computadora, ya que por cuestiones de compatibilidad de librearías, resulta mucho mejor la programación desde la PC.

Figura 25

Código “thingProperties.h”

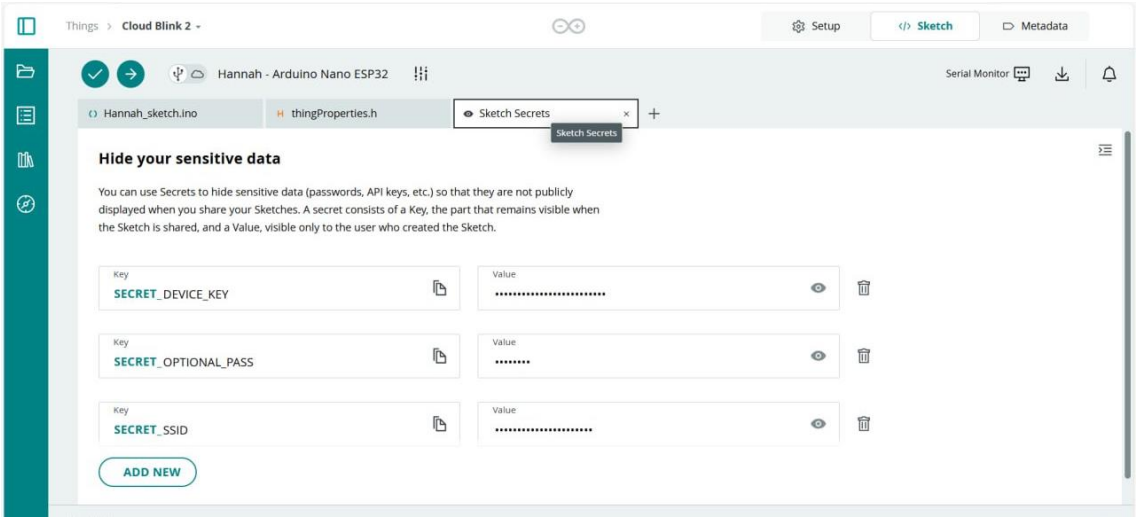


Nota: Pestaña de código necesaria para programar desde la computadora. Fuente propia.

También se crea la pestaña Sketch Secrets (Figura 26) que son las credenciales para conectar al Wifi y la “secret device key”.

Figura 26

Credenciales para la conexión a internet de Arduino Nano ESP32

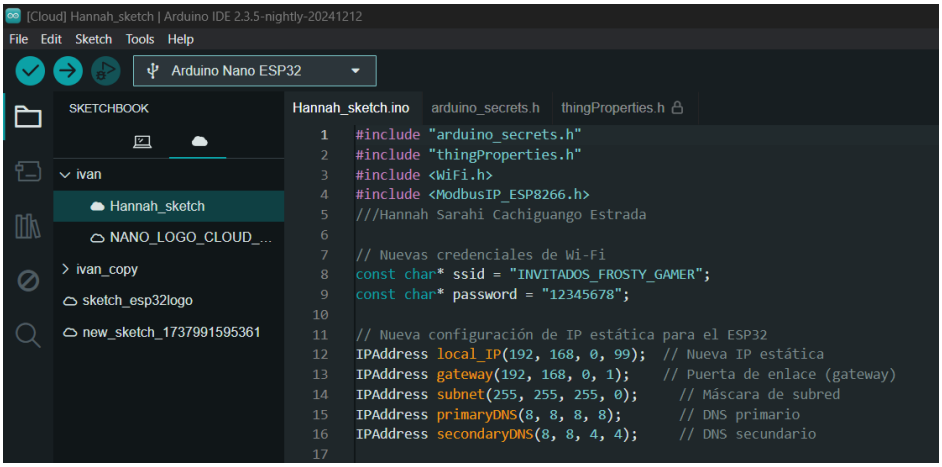


Nota: Pestaña de credenciales necesarias para la conexión. Fuente propia.

4. Una vez hecho esto ya podemos descargar el archivo .zip que se genera al pulsar el botón download que está en la parte superior derecha del editor en la nube. Abrimos y ya podemos sincronizar el código que hagamos en la PC (Figura 27), como se indicó anteriormente, todas las compilaciones se harán en la PC

Figura 27

Sketch descargado de cloud a la computadora y sincronizado nuevamente con la nube.

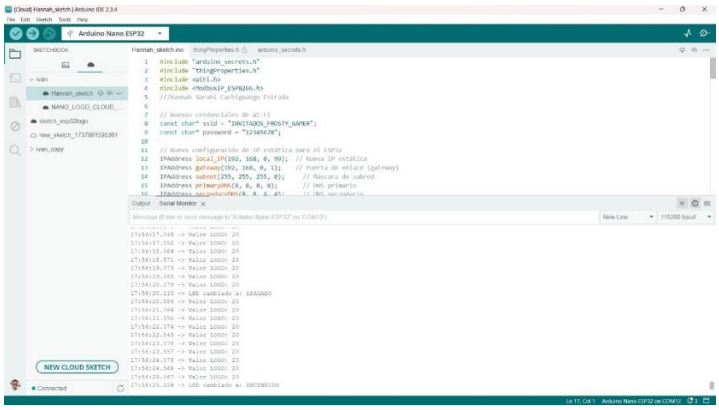


Nota: Código sincronizado con la nube Arduino Cloud.

5. Compilamos y cargamos y verificamos el puerto serial para verificar si hay algún dato que el Arduino esté recibiendo. Y si lo hace porque se recibe el en entero 20 (Figura 28).

Figura 28

Arduino Nano recibiendo el entero "20"

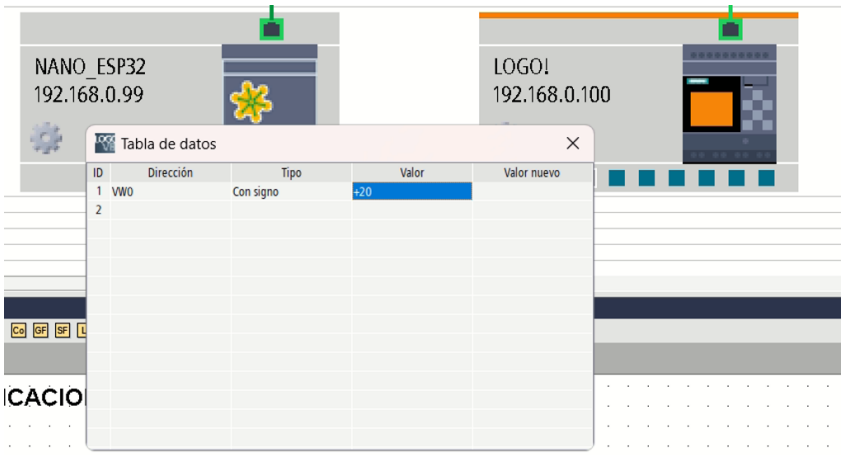


Nota: Arduino recibiendo el dato desde LOGO! Siemens. Fuente propia

El dato entero “20” que le manda LOGO! se escribe desde la tabla de datos (Figura 29).

Figura 29

LOGO! enviando el entero “20”



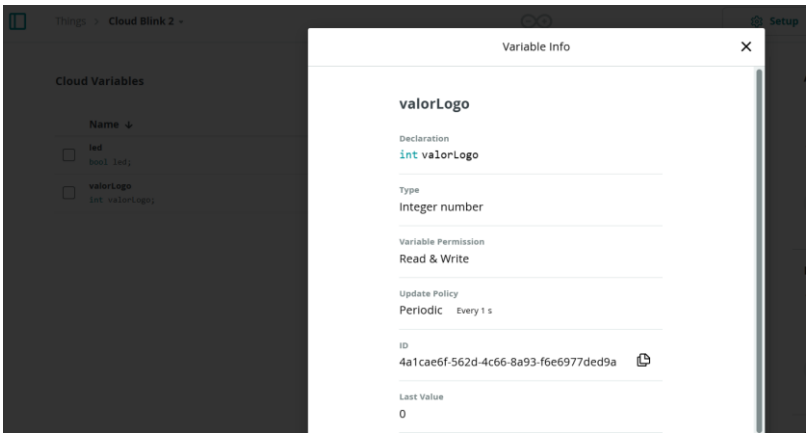
Nota: LOGO! enviando desde la dirección VW0 el entero “20”. Fuente propia

3.2.1.6. Asignar variables para el Dashboard en Arduino Cloud.

En el menú Things se asigna el TAG “valorLogo” como entero, de lectura y escritura y actualización cada segundo (Figura 30).

Figura 30

Asignación de valor entero en THINGS



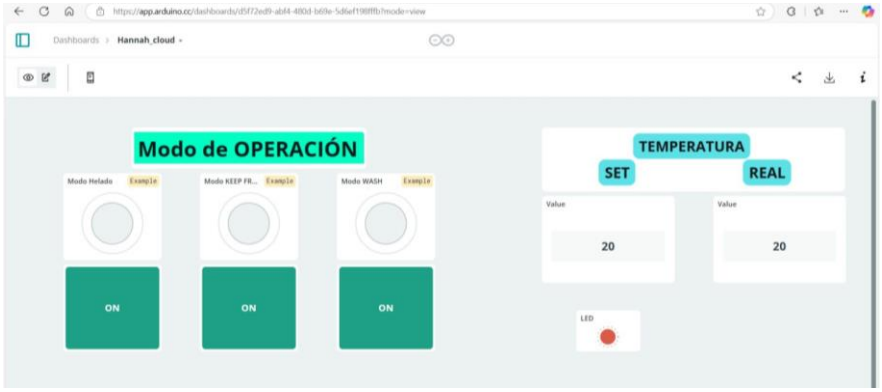
Nota: Creando la variable a publicarse en el dashboard (similar a HMI)

Una vez que se añade la variable vamos armando nuestra HMI, asignado la variable a un bloque “value”, de igual manera se arma el HMI para modo vista de computador y vista de móvil para verificar desde el celular.

Se corre el dashboard y se ve el numero 20 con lo que es exitosa la comunicación de un Arduino NANO ESP32 con el PLC LOGO! (Figura 31).

Figura 31

Visualización del entero "20" en el dashboard creado en DASHBOARDS en Arduino Cloud



Nota: Creando la variable a publicarse en el dashboard (HMI). Fuente propia.

Se comprueba también desde la aplicación móvil se puede ver el entero "20" (Figura 32).

Figura 32

Verificación de valor entero "20" en la aplicación IoT Remote de Arduino

Nota: Valor entero visto desde la aplicación de Arduino Cloud. Fuente propia

3.3. Enlace del programa de LOGO!, monitoreo remoto, control de la producción, alarmas con Arduino Cloud

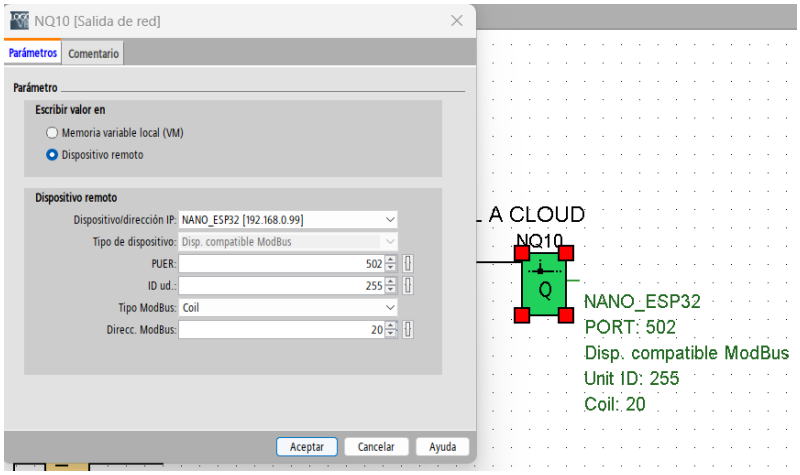
Una vez verificada la conexión entre LOGO! Siemens y Arduino Cloud y programados los modos de trabajo (Modo Helado, Wash, Keep Fresh) se proceden a enlazar las variables digitales y analógicas de cada función necesaria para el monitoreo y control remoto

3.3.1. Ejemplo de enlace de una variable digital de LOGO! a Arduino Cloud

Se asigna en LOGO! la variable salida digital de red NQ10 y se parametriza como la Figura 33.

Figura 33

Parametrización salida digital de red de LOGO!



Nota: Salida de red NQ10 a Arduino Cloud. Fuente propia

Al ser una salida digital un tipo Modbus: Coil, dirección 20.

NOTA IMPORTANTE: Para que se establezca la comunicación correcta con el Arduino NANO es necesario establecer un OFSET de -1, es decir en el código del Arduino es necesario declarar en 19. Ejemplo del código de declaración:

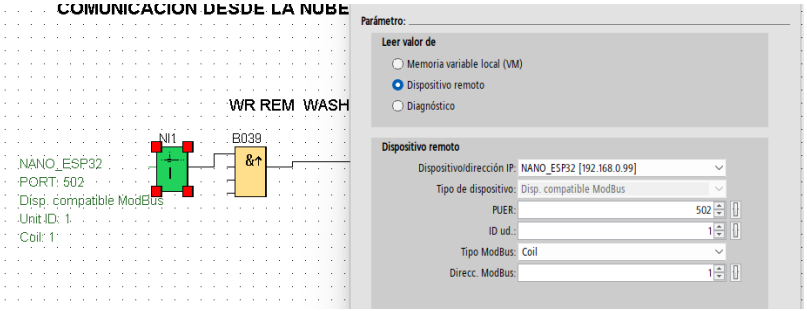
const int coilNQ10 = 19; //FALLA GENERAL A CLOUD

3.3.2. Ejemplo de enlace de una variable digital de Arduino Cloud a LOGO!

Se asigna en LOGO! la variable de entrada digital de red NI1 y se parametriza como la Figura 34.

Figura 34

Parametrización entrada digital de red de LOGO!



Nota: Entrada de red NI1 desde Arduino Cloud. Fuente propia

Al ser una entrada digital un tipo Modbus: Coil, dirección 1.

NOTA IMPORTANTE: Para que se establezca la comunicación correcta con el Arduino NANO es necesario establecer un OFSET de -1, es decir en el código del Arduino es necesario declarar en 0. Ejemplo del código de declaración:

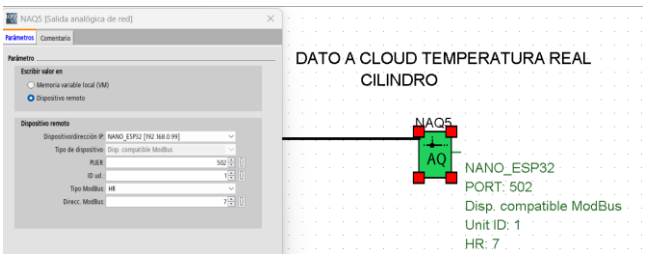
const int coilNI1 = 0; //Modo wash desde Cloud

3.3.3. Ejemplo de enlace de una variable analógica Logo! a Arduino Cloud

Se asigna en LOGO! la variable de salida analógica de red NAQ5 y se parametriza como la Figura 35.

Figura 35

Parametrización de salida analógica de red de LOGO! a Arduino Cloud



Nota: Salida de red hacia Arduino Cloud. Fuente propia

Al ser una salida analógica un tipo Modbus: HR, dirección 7.

NOTA IMPORTANTE: Para que se establezca la comunicación correcta con el Arduino NANO es necesario establecer un OFSET de -1, es decir en el código del Arduino es necesario declarar en 6. Ejemplo del código de declaración:

const int regAnalogico6 = 6; // LEE VALOR DE TEMPERATURA REAL CILINDROS B079 HR: 7 EN LOGO!

3.3.4. Monitoreo y Control Remoto

Se diseña el dashboard en Arduino Cloud que contiene las siguientes opciones:

Seleccionar modo de Trabajo: Se selecciona conforme la operación de la máquina: Wash, Helado, Keep Fresh (Figura 36).

Figura 36

Modo de operación desde dashboard de Arduino cloud.

Nota: Tres modos de operación desde Arduino Cloud. Fuente propia

Configurar temperatura: Como se puede ver en la Figura 37, se puede modificar desde la computadora o desde la aplicación, se configura el encendido y apagado del compresor con “SET ON” y “SET OFF”, esto es para el cilindro y las tolvas

Figura 37

Configurar temperaturas desde Arduino Cloud

Nota: Visualización y configuración de temperaturas. Fuente propia

Monitorear el consumo de corriente de los motores: Desde la aplicación se puede monitorear desde cualquier lugar el consumo de corriente a fin de verificar el correcto funcionamiento de estos (Figura 38).

Figura 38

Monitoreo de corrientes desde Arduino Cloud

Nota: Visualización del consumo de corriente de cada motor. Fuente propia

Control de producción: Para este requerimiento se programaron contadores para llevar el registro diario de venta de helados, estos valores se comparan con los consumibles (conos, vasos) que deben coincidir para llevar la contabilidad de productos vendidos diarios. En Arduino Cloud se ve como en la Figura 39.

Figura 39

Monitoreo del control de producción desde Arduino Cloud

Nota: Visualización del conteo para cada día de la producción. Fuente propia

Monitoreo de fallas: Con el funcionamiento continuo de la máquina se ha podido establecer posibles problemas que puedan suceder en el sistema, esto se ve reflejado en las alarmas. Para mayor detalle se puede consultar en los Anexos, que trata del manual de

usuario del sistema en su apartado de **análisis de fallas.** En la Figura 40 se puede ver el listado de fallas que pueden aparecer en el sistema.

Figura 40

Monitoreo de fallas del sistema en Arduino Cloud

Nota: Visualización del estado de fallas del sistema. Fuente propia

3.4. Pruebas de Desempeño

Para evaluar la efectividad de la automatización implementada, se realizaron pruebas de desempeño considerando tres aspectos clave: tiempos de producción, eficiencia energética y consistencia del producto.

Comparación de los tiempos de producción

. Con la implementación del sistema automatizado se ha ganado mejoras considerables en la producción lo que conllevan a una mejor producción de helado soft, tales como:

- La regulación de temperatura antes se hacía solo por tiempo y consistencia, ahora se hace por temperatura y consistencia.
- La coordinación entre el agitador, el compresor y las electroválvulas optimiza la entrega del helado, verificando si es necesario que electroválvulas activar dependiendo del modo de trabajo de la máquina.
- Ahora es posible batir el producto almacenado y preparar el helado antes de la apertura del local, el control y monitoreo está al alcance de las aplicaciones control

remoto mediante Arduino Cloud, lo que permite que, al iniciar el turno de venta, el helado ya esté listo para su comercialización, optimizando tiempos y mejorando la experiencia del cliente.

- Si hay algún corte de energía, por ejemplo, en la noche, antes de esta implementación, al estar en modo "conservación" ese modo se apagaba y al retornar la energía el producto se malograba, con el sistema implementado al volver la energía eléctrica automáticamente se pone en modo conservación, independiente si la máquina está conectada a la nube.

Análisis de la eficiencia energética

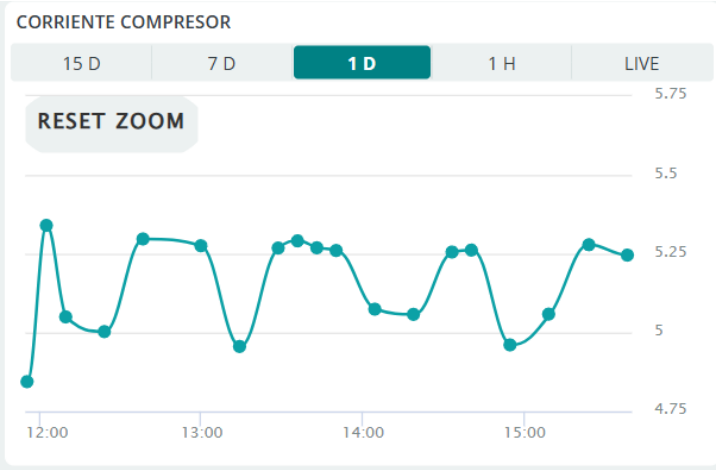
La integración de protecciones eléctricas y el programa de control ha permitido un uso más eficiente de la energía:

- **Optimización del encendido del compresor**: Se activa únicamente cuando la temperatura lo requiere, reduciendo su tiempo de funcionamiento innecesario, se activará en base al requerimiento de helado por parte de la clientela.

Usando el TREND de la Figura 41 tomado de Arduino Cloud de la corriente de compresor se verificó lo siguiente:

Figura 41

Trend de Corriente Compresor



Nota: Trend de corriente en 4 horas de producción. Fuente propia

La corriente no supera los 5.4A, siendo la nominal de 5.8A, lo que resulta que el compresor está trabajando en condiciones óptimas, lo que implica que su vida útil se alargue.

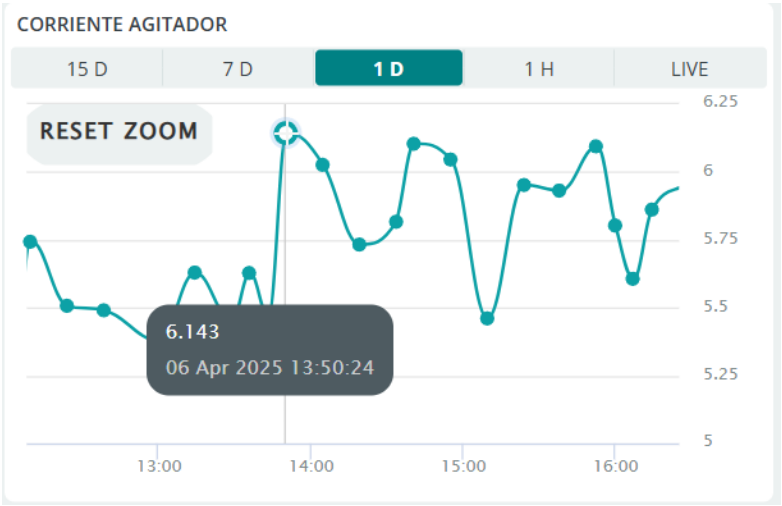
- **Reducción del consumo eléctrico del motor agitador**: Antes de la implementación del nuevo sistema, en el modo de conservación, el agitador y el compresor funcionaban en conjunto. Ahora, funciona solo el compresor con las electroválvulas, priorizando la refrigeración en las tolvas. En modo de producción, la prioridad es inversa, asegurando que la temperatura en el cilindro se mantenga correcta.

  Del mismo modo como se analizó el consumo de la corriente del compresor, usando el TREND del agitador en la Figura 42 podemos ver realmente que la configuración de los valores de temperatura es la adecuada del cilindro ya que no hacen que se eleve mucho la temperatura y el producto sea demasiado “duro” y esto afecte al motor agitador.

  Entonces podemos establecer que se configura la temperatura en un punto que la corriente del agitador no sobrepase su corriente nominal (7.2A), pero que la textura del helado sea la óptima, en la imagen podemos ver en análisis de 4 horas que la corriente máxima es de 6.1 A.

Figura 42

Trend de Corriente Agitador



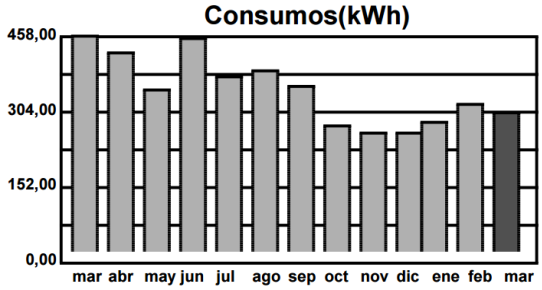
Nota: Trend de corriente del agitador en 4 horas de producción. Fuente propia

- **Reducción de consumo reflejado en la planilla de consumo eléctrico**

Se toma los datos de la planilla Nro.: 001-999-111456215 adjunta en Anexos, donde se puede analizar el consumo mensual (Figura 43), en el cual se puede ver el historial de consumo desde marzo del 2024 a marzo 2025, en base a ella se toma los consumos aproximados de cada mes.

Figura 43

Consumo eléctrico mensual de 1 año



Nota: Valores de consumo eléctrico en KW/h. Fuente propia

En la Tabla 15, se comparan los consumos promedio mensuales antes y después de la implementación del control con LOGO! realizada en octubre de 2024. Para garantizar la validez del análisis, se han excluido los periodos afectados por cortes forzados de suministro eléctrico en el país en el 2024, (15–30 abr 2024 y 23 sep–20 dic 2024).

Tabla 15

Análisis de los consumos aproximados de cada mes en base a la Figura 43

Periodo	Meses considerados	Consumo promedio (kWh/mes)	Reducción vs pre-intervención
Pre-intervención	Mar-24 (458), Jun-24 (458), Jul-24 (380), Ago-24 (395)	(458+458+380+395)/4 = **423**	–
Post-intervención	Ene-25 (285), Feb-25 (320), Mar-25 (305)	(285+320+305)/3 = **303**	(423−303)/423 ≈ **28,4 %**

Se puede apreciar un ahorro aproximado del 28%, esto es por la optimización de los arranques del compresor y agitador con el control de temperatura y solo el uso del compresor en el modo “KEEP FRESH”, o modo de mantenimiento de producto fresco en los periodos que no se produce, como por ejemplo en las noches.

- **Minimización de pérdidas térmicas**: La programación del PLC asegura la activación coordinada de las electroválvulas, manteniendo la refrigeración solo cuando es necesario.
- **Verificar el consumo de corriente de los motores**: Con el fin de monitorear su estado operativo, o caso contrario, si la corriente está por encima de la nominal, establecer un plan de mantenimiento preventivo

Consistencia del helado producido bajo diferentes condiciones operativas

Se realizaron pruebas para garantizar la uniformidad del producto en distintos escenarios:

- **Variaciones en la demanda**: La automatización permitió mantener una textura homogénea del helado sin importar el número de servicios consecutivos.

Las pruebas de funcionamiento que realizaron confirmaron que la automatización ha mejorado significativamente la eficiencia operativa, optimizado el consumo energético y estandarizado la calidad del producto, ámbitos esenciales para que los dueños de negocio puedan optar por migrar sus máquinas a un sistema más moderno.

3.5. Discusión de Resultados

El proyecto de automatización para la máquina de helado soft, logró integrar con éxito un PLC LOGO! de Siemens con un Arduino Nano ESP32. Para que se comunicaran entre sí, se usó el protocolo Modbus TCP, lo que se permitió enviar todos los datos obtenidos del PLC de la máquina a la plataforma de Arduino Cloud y poder verlos desde cualquier lugar.

Por lo tanto, se pudo monitorear las variables más importantes: la temperatura de las tolvas y los cilindros, y la corriente que consumían los motores. La comunicación fue muy estable, aunque se notó que a veces los datos tardaban hasta 5 segundos en actualizarse en el dashboard remoto. Pese a este pequeño retraso no afectó el funcionamiento de la máquina.

Todos los modos de operación que se programó como: "Servir Helado", "Mantenimiento", "Conservación" y "Limpieza" funcionaron tal como se esperaba. El sistema respetó siempre los tiempos y las temperaturas que se configuró, lo que asegura que el helado soft se produzca con buena calidad y de forma consistente.

Para la seguridad del sistema, se instaló guardamotores y sensores de corriente. Cuando el sistema detectaba una sobrecarga, se activaba una alarma tanto en la pantalla local (la HMI del LOGO!) como en la nube de Arduino. Esto es clave para proteger los motores y alargar su vida útil.

Un resultado muy positivo fue el ahorro de energía. Al controlar bien la temperatura y apagar el agitador cuando no es necesario (en el modo de conservación), logramos bajar de forma significativa el consumo en la factura eléctrica mensual.

Con lo descrito en lo anterior, este proyecto demuestra que se puede modernizar maquinaria y conectarla a la nube de forma accesible. Ya que se usó componentes relativamente comunes y tecnología abierta, este mismo sistema podría ser adaptado fácilmente por otras PYMEs, no solo en el sector de alimentos, sino en la industria local en general.

CONCLUSIONES

El presente trabajo de titulación tuvo como objetivo principal la automatización de una máquina de helado soft mediante la integración de tecnologías de control local, monitoreo en la nube y gestión remota de la producción utilizando Arduino Cloud. A continuación, se presentan las conclusiones en función de los objetivos generales y específicos planteados:

- **Implementación del sistema de control moderno basado en IoT y Arduino Cloud**

Se logró implementar con éxito un sistema de control moderno utilizando el **micro PLC LOGO! de Siemens** para el control local y el **Arduino Nano ESP32** como enlace entre el LOGO! y la plataforma **Arduino Cloud** mediante **Modbus TCP**. Este sistema permitió la supervisión remota de variables críticas como la temperatura, el consumo de corriente de los motores y el estado de la máquina, así como la gestión a distancia de la producción. La integración de tecnologías IoT y computación en la nube demostró ser una solución eficiente y accesible para mejorar la funcionalidad de la máquina de helado soft.

- **Integración de funcionalidades avanzadas**

Se implemento funcionalidades avanzadas, tales como la modalidad de conservación de producto, el medidor de corriente de los motores (agitador y compresor) y el desarrollo de algoritmos de control de proceso. Estas funcionalidades simplifican y mejoran el funcionamiento de la máquina para mantener y producir alta calidad de helado y abaratar los costos operativos. Se pudo mantener la temperatura en depósito de las tolvas inferior a 5°C en su **modo de "conservación"** que garantizó la calidad del helado durante horas de reposo, ósea fuera de horario de venta.

- **Implementación de una interfaz hombre-máquina (HMI) amigable**

Se ha implementado una interfaz hombre-máquina (HMI) sencilla y fácil de usar con el panel LOGO! TD, el usuario solo elige el modo de producción con sus teclas integradas y puede empezar a trabajar, de igual manera se complementa con Arduino Cloud mediante su dashboard interactivo. Donde se puede ver más variables y su control de

manera remota. Todo esto se traduce para una óptima experiencia usuario al usar la máquina, tanto para operadores como dueños de negocio.

- **Integración de protecciones eléctricas con monitoreo**

Se integraron protecciones eléctricas para los motores del agitador y el compresor, mejorando la seguridad y el monitoreo del equipo. Además, se implementó un sistema de gestión de fallas, que se muestran en la pantalla HMI y en el dashboard de Arduino Cloud, alertando sobre posibles problemas o condiciones anómalas, como sobrecargas o temperaturas fuera de los rangos establecidos. Esto contribuye a prolongar la vida útil de los motores y a reducir los riesgos operativos.

- **Mejora de la eficiencia operativa y sostenibilidad**

La implementación del sistema hizo que se mejoraran los tiempos de producción y se haga uso más eficiente de la energía, esto se logró mediante la optimización del encendido del motor agitador en modo de "conservación". Otra razón es que se logró mantener constante la consistencia del helado si aplicamos distintas condiciones operativas, haciendo más confiable la producción final. Estos resultados apoyan la teoría de que la automatización incorporando IoT y Arduino Cloud aumenta la eficiencia, la estabilidad y la competitividad de las máquinas de helado soft.

- **Replicabilidad del modelo en otras industrias**

Se demostró que nuestro modelo de implementación puede ser un modelo replicable para otros sectores alimenticios, sobre todo en pequeñas y medianas empresas (PYMEs). La adaptabilidad y flexibilidad de las tecnologías empleadas (LOGO! Siemens, Arduino NANO ESP32 IoT y Arduino Cloud) ofrecen la posibilidad de adaptarlos a diversos tipos de máquinas que manejen procesos industriales, y no necesariamente pueden ser de carácter alimenticio.

RECOMENDACIONES

- Se recomienda desarrollar algoritmos predictivos basados en Machine Learning, investigando acerca modelos de aprendizaje automático tales como regresión, redes neuronales, árboles de decisión, etc. para anticipar fallos en sensores o variaciones críticas del proceso, como temperatura, y corriente en los motores, y activar automáticamente modos de parada o mantenimiento antes de la ocurrencia de la falla.

- Se puede colocar una sonda de pH industrial en las tolvas, calibrada periódicamente, conectada al LOGO!, programando rangos de alarma predefinidos y visualización en HMI y Arduino Cloud, de modo que ante desviaciones críticas de acidez se detenga automáticamente la producción. Con esto se garantiza que el producto que este se mantenga inocuo.

- Se recomienda el desarrollo e implementación de una aplicación de realidad aumentada (AR) para tabletas o gafas, enfocada en optimizar el mantenimiento de la máquina. Esta herramienta debería reconocer equipos y superponer información técnica (diagramas, flujogramas, estado) en tiempo real, ofreciendo instrucciones de mantenimiento guiadas. Además, la funcionalidad de colaboración remota con expertos sería crucial para el soporte en campo. Se espera que esta solución reduzca errores, agilice intervenciones y elimine la necesidad de manuales impresos, mejorando significativamente la eficiencia operativa.

REFERENCIAS

Silva Proaño, C. S. (2017). *Tragbare Feinstaub Messstation mit Datenübermittlung an eine SQL Datenbank*. Tesis de maestría. Technische Universität Hamburg.

Ballesteros Martínez, M. Á., & Quiroga Villarraga, I. V. (s.f.). *Modelamiento y simulación en CFD de la transferencia de momento y calor en los dos sistemas de refrigeración de una máquina de soft serve ice cream*. Informe académico. Universidad de los Andes.

Carvajalino Illera, A. G., & Celis Pérez, D. L. (2021). *Simulación de sistema de intercambio de calor para una máquina de helados tipo soft serve a través de dinámica de fluidos computacional (CFD)*. Trabajo de grado. Universidad Industrial de Santander.

Guachi, D., Rojas, J., & Sotomayor, N. (2016). *Automatización y monitoreo del sistema de refrigeración de los cuartos fríos de Fabrilacteos Cía. Ltda. – Helados Jotaerre*. Ponencia. En *XXVI Jornadas en Ingeniería Eléctrica y Electrónica - EPN*.

Lojan Bermeo, E. F. (2015). *Automatización de la línea de ensamble de congeladores e ingreso de producto terminado en la empresa Induglob S.A*. Tesis de maestría. Universidad Politécnica Salesiana.

Zafar, S., Miraj, G., Baloch, R., Murtaza, D., & Arshad, K. (2018). *An IoT based real-time environmental monitoring system using Arduino and cloud service*. Artículo científico. *Engineering, Technology & Applied Science Research, 8*(4), 3238-3242. https://www.etasr.com/index.php/ETASR/article/view/1503

Palafox Rodríguez, E. D., & Ávila Camacho, F. J. (2023). *Utilización de Arduino en dispositivos electrónicos*. Artículo científico. *RIDT, Vol. X No. 00*, 32-35. https://doi.org/10.12345/ridt.ccaiXXiYY.ZZZZ

Tafur Morey, E. L. (2022). *Automatización de procesos en la industria alimentaria*. Examen de suficiencia profesional. Universidad Nacional de la Amazonía Peruana.

Agudelo, N., Tano, G., & Vargas, C. A. (s.f.). *Historia de la automatización*. Ensayo académico. Universidad ECCI.

Aranburu, E., Lasa, G., & Gerrikagoitia, J. K. (2018). *Evaluating the human machine interface experience in industrial workplaces*. Ponencia. En *British HCI 2018*. http://dx.doi.org/10.14236/ewic/HCI2018.93

Martínez, M., & Mesías, M. (2021). *Aplicación de las tecnologías de la información y la comunicación (TICs) en la cadena alimentaria*. *RITI Journal*, *9*(19), 47–48. https://doi.org/10.36825/RITI.09.19.004

Prada Moreno, P. D. (s.f.). *El internet de las cosas en la industria de alimentos*. Artículo académico. Fundación Universitaria de Popayán.

Yepez Teran, R. A. (2024). *Revisión sistemática de literatura sobre el internet de las cosas enfocada a la agricultura de precisión*. Trabajo de titulación. Universidad Politécnica Salesiana.

Aguilar Zavaleta, S. (2020). *Diseño de una solución basada en el internet de las cosas (IoT) empleando LoRaWAN para el monitoreo de cultivos agrícolas en Perú*. Tesis profesional. Universidad Nacional de Ingeniería.

Amaya Fariño, L. M., Tumbaco Reyes, A., Roca Quirumbay, E., Villón González, T., Mendoza Morán, B., & Reyes Quimís, Á. (2020). *El IoT aplicado a la Domótica*. *Revista Científica y Tecnológica UPSE, 7*(1), 21–28. https://doi.org/10.26423/rctu.v7i1.507

Alarcón Monteza, D. D. (2021). *Implementación de un sistema de monitoreo y control en tiempo real para el análisis predictivo de un sistema eléctrico de potencia con generación distribuida*. Tesis profesional. Universidad de Ingeniería y Tecnología.

Real, G. E., Jáuregui, F., & Vitali, A. O. (s.f.). *Sistema de adquisición de datos y control industrial basado en Arduino Due con hardware y software libre*. Proyecto académico. Universidad Nacional de General Sarmiento.

Ahmed, A. S., Marzog, H. A., & Abdul-Rahaim, L. A. (2021). *Design and implement of robotic arm and control of moving via IoT with Arduino ESP32*. *International Journal of Electrical and Computer Engineering, 11*(5), 3924–3933. https://doi.org/10.11591/ijece.v11i5.pp3924-3933

Durán, C. M., & Pinto, H. L. (s.f.). *Implementación de un controlador PID por medio de un PLC TWIDO aplicado al control de un proceso de pasteurizado de leche*. Universidad de Pamplona.

Del Castillo Santana, K. (2022). *Teoría de control y aplicaciones a la industria de alimentos. Neumática. Controladores lógicos programables (PLC)*. Examen de suficiencia profesional. Universidad Nacional de la Amazonía Peruana.

Herrera Paria, P. A. (2019). *Diseño de un sistema automático de control y registro de temperatura para el proceso de pasteurización en la industria alimentaria*. Tesis profesional. Universidad Peruana de Ciencias Aplicadas.

Flores de Válgaz Casanova, C. M. (2013). *Diseño y construcción de un sistema de automatización mediante PLC de una pasteurizadora del laboratorio de alimentos*. Tesis de grado. Universidad Laica Eloy Alfaro de Manabí.

Domínguez, R. (2020). *Control de motores eléctricos: sistemas de automatismos eléctricos con lógica cableada*. Manual teórico-práctico. Faradayos.

Dasari, P. K., Harsha, D. S., & Chandrasekhar, E. (s.f.). *Implementation of low cost home monitoring, controlling and security system using IoT*. *International Journal of Engineering Research & Technology (IJERT)*.

Menendez Elguera, E. W., Aldana Fernández, S. R., Torres Sanchez, S. D., & Arzapalo Bello, R. D. (2023). *Productivity improvement by means of method engineering tools and automation in ice cream production at Bonanza Company*. Tesis profesional. Universidad Continental.

Aravindan, V., & James, D. (2017). *Smart homes using Internet of Things*. *International Research Journal of Engineering and Technology (IRJET)*, *4*(4), 1725–1728.

Galicia Moreno, L. E., García Roa, E. A., & Reyes Chavarría, R. (2013). *Coordinación de protecciones para un sistema eléctrico industrial*. Tesis profesional. Instituto Politécnico Nacional, ESIME.

Sáenz Proaño, P. M., & Villamarín Granda, D. F. (2013). *Plan de negocios para la creación de una heladería self-service especializada en helado de yogurt soft a base de fruta congelada y toppings*. Trabajo de titulación. Universidad de las Américas (UDLA).

ANEXOS

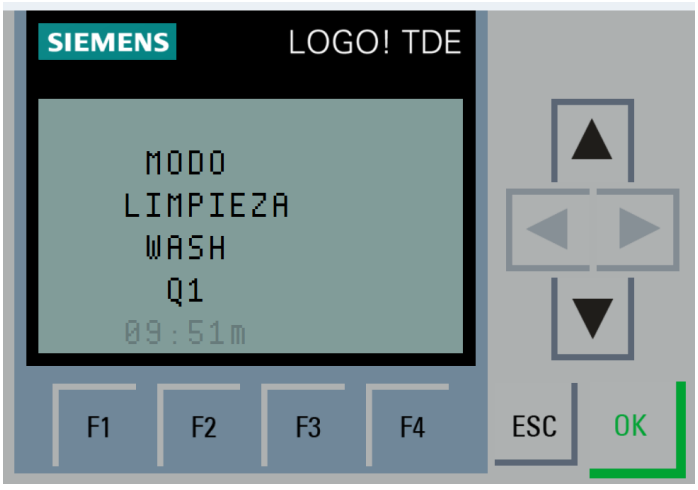
MANUAL DE USUARIO DE LA MAQUINA DE HELADO SOFT

La máquina de helado soft tiene 3 estados de trabajo:

1. Modo de Limpieza (WASH)
2. Modo de Hacer Helado
3. Modo de conservación (KEEP FRESH)

Cualquier modo se activa y desactiva presionando las teclas F1, F2, F3 o desde la aplicación IoT Arduino.

- **MODO LIMPIEZA (WASH)**



Es decir, si presionamos la tecla F1 se activa el modo de limpieza, entonces empieza a funcionar solo el motor agitador, este modo se usa al inicio de producción diaria, cuando

el helado ha estado en la noche en modo de conservación, para batirlo y que la mezcla se homogenice. Pero donde más vamos a usar este modo es cuando es necesario realizar la limpieza, en un periodo de cada 5 días.

PROCESO DE LIMPIEZA:

- Se saca todo el producto del interior y se refrigera
- Se apaga todo el sistema eléctrico
- Se coloca agua con jabón para biberones
- Se energiza la máquina y se deja en dos ciclos de WASH (20minutos) y luego de eso se saca el agua con jabón y residuos.
- Se coloca dos ciclos de WASH SOLO CON AGUA (20 min)
- Se quita la energía, se desarma la máquina conforme el manual del fabricante y se lavan sus piezas, se secan y se vuelven a armar.

- **MODO DE HACER HELADO**

Este modo como su nombre lo indica es para preparar la mezcla de helado dentro del cilindro, que, por el proceso de refrigeración, esta mezcla se irá congelando hasta obtener cremosidad y textura a fin de servir al cliente final.

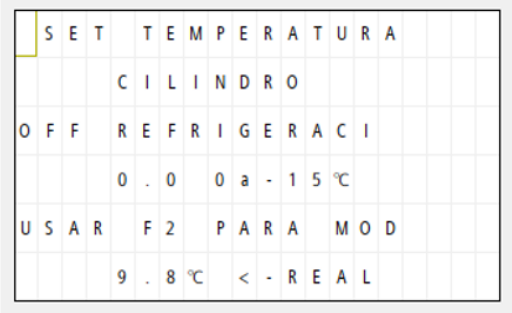
PROCESO DE "HACER HELADO" AL INICIO DE TURNO

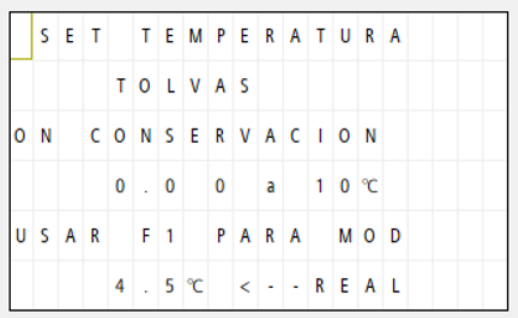
Modo LOCAL

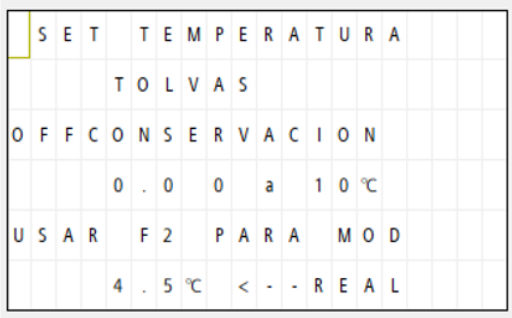
1. Ingresar la base de cada sabor (2 sabores) preparadas a las tolvas y verificar que se llene el cilindro y en las tolvas por lo menos hasta la mitad. Este paso es solo si el producto base estuvo en refrigeración y la maquina estaba vacía, por ejemplo, después de lavarla. Si ya hubo producto y la máquina estuvo en modo "MANTENER PRODUCTO" o KEEP FRESH pasar al paso 2.
2. Colocar modo WASH para para que la mezcla se homogenice durante un ciclo (10 min).

3. Luego de terminado el ciclo vamos a ver esta pantalla.

4. Verificar las temperaturas en el cilindro y las tolvas por si hubo algún cambio mediante la tecla F4, con esta tecla se va navegando entre estas pantallas:







Temperaturas que se sugieren:

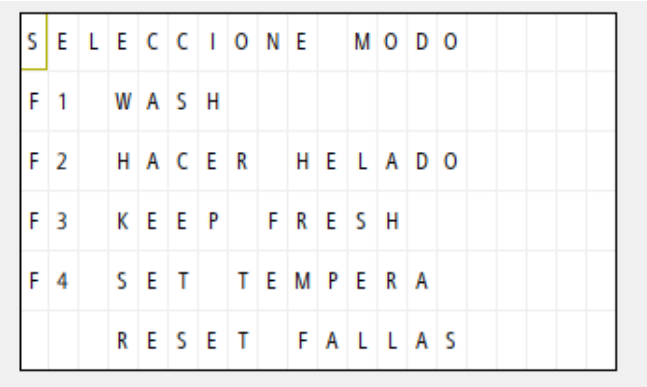
SET TEMP ON CILINDRO: 4°C

SET TEMP OFF CILINDRO: -11°C

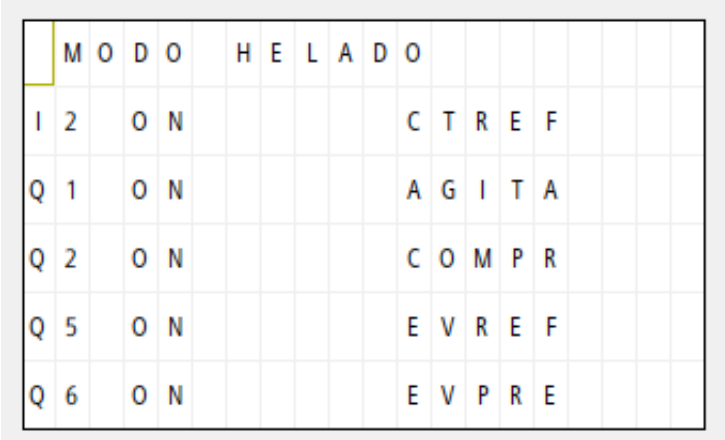
SET TEMP. ON TOLVAS: 6°C

SET TEMP. OFF TOLVAS: 0°C

5. Una vez que se verifican estas temperaturas con F4 se volverá nuevamente a la pantalla principal y presionamos F2:



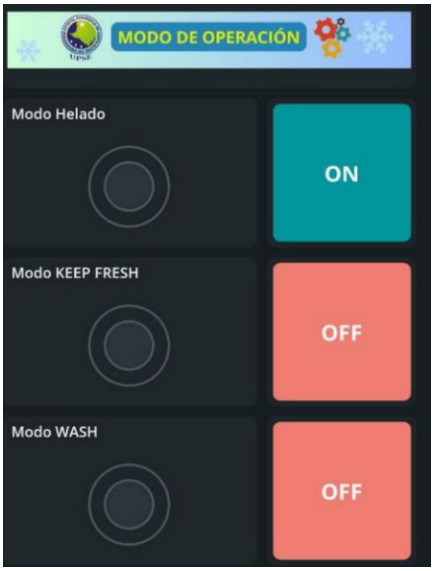
6. Se activa el modo de hacer helado e inicia automáticamente la secuencia de encender el agitador, luego el compresor y por ultimo las electroválvulas.



El sistema estará en funcionamiento hasta el que la temperatura descienda a -11°C y se apagará automáticamente.

Nuevamente se iniciará el encendido en dos casos, cuando la temperatura baje a +4°C y necesite otra vez refrigeración hasta -11°C, o cuando se active la palanca de SERVIR HELADO, cuando sucede lo segundo el sistema funciona durante 2 minutos independientemente de la temperatura, es decir no toma en cuenta durante ese tiempo ni

la temperatura de encendido ni la de apagado. Y así el sistema mantiene la temperatura adecuada de manera que sea eficiente con el consumo eléctrico.



Modo REMOTO

1. Resulta muy conveniente si el producto ha estado en KEEP FRESH y queremos hacer helado controlando la maquina ya sea desde el celular o la computadora mediante Arduino Cloud. Validamos así mismo las temperaturas desde la aplicación IoT Remote: Solo

desactivamos el modo KEEP FRESH y activamos MODO HELADO y el sistema empieza con la ejecución del helado.

El sistema inicia y se puede monitorear las temperaturas, corrientes en tiempo real:

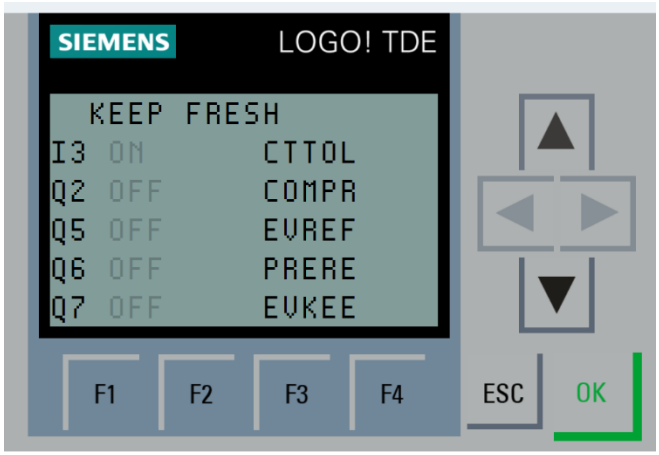
- **MODO KEEP FRESH.**

Este modo es para mantener la temperatura adecuada en las tolvas y en el cilindro el helado y este no se pueda llegar a dañar. Se ha comprobado por pruebas de funcionamiento que la temperatura no debe superar los 8 °C. Por ende, el sistema debe tratar de mantener esa temperatura activando únicamente el compresor y las electroválvulas.

Nota: Este modo máximo se puede usar 5 días seguidos, ya que a ese periodo se debe lavar la máquina.

El modo KEEP FRESH se puede activar en el fin de turno y el sistema en base a la temperatura que tiene configurado encenderá solo el compresor y las electroválvulas, esto da una eficiencia muy buena de consumo de energía eléctrica.

Se activa en modo local presionando la tecla F3 y aparece esta pantalla:



En modo remoto:

- **Análisis de TRENDS**

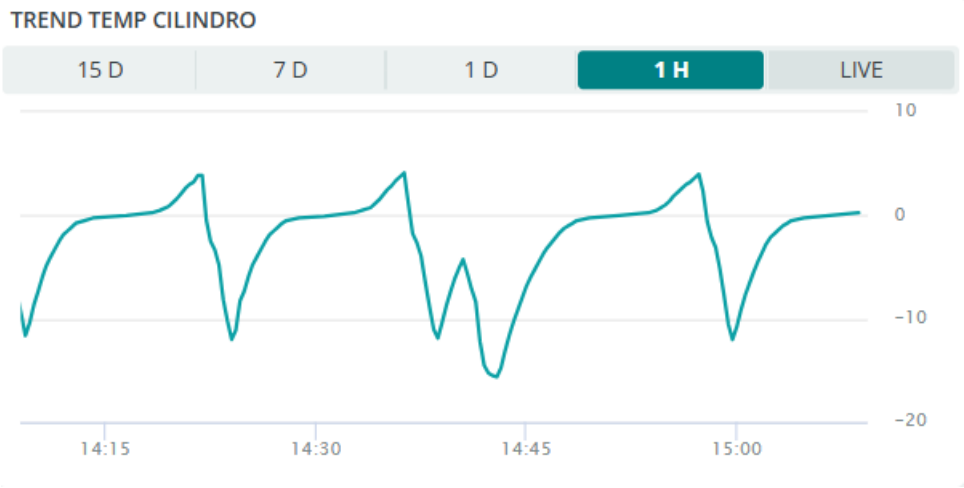
Sirven para ver como en el tiempo las variables como temperatura y las corrientes de los motores se van comportando, para ver si se hace correcciones pertinentes al sistema.

Hay trends para:

1. Temperatura cilindro.

2. Temperatura tolvas
3. Corriente compresor
4. Corriente agitador.

EJEMPLO: Trend del cilindro.



- **ANÁLISIS DE FALLAS**

Conforme el funcionamiento de la máquina se ha implementado un sistema de evidencias de fallas del sistema en modo local y remoto a fin de precautelar la vida útil de los elementos, especialmente del compresor y agitador.

Tenemos las siguientes fallas:

1. **Falla por mucho tiempo encendido compresor.**

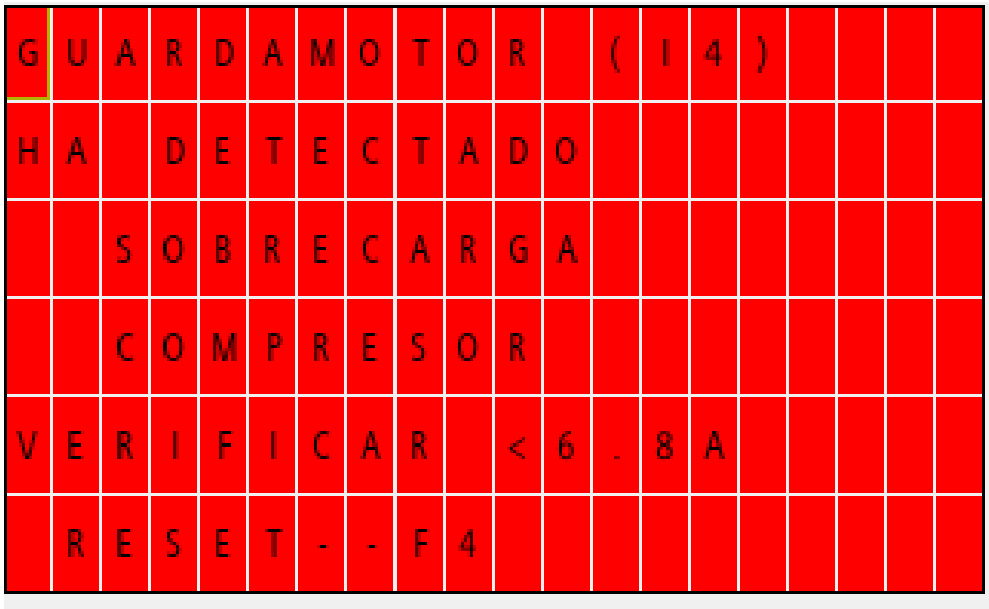
   Con esto se monitorea cuanto tiempo el compresor se demora en enfriar la mezcla, si por algún motivo, por ejemplo, perdida de gas refrigerante o mucha temperatura exterior, fallo de sensor, hacen que el encendido del motor compresor se pueda prolongar, por ende se activaría esa alarma, este tiempo de vigilancia está en 18 minutos.

2. **Sensor de sobre corriente compresor**: Cuando el sistema detecte más de 6.8A (nominal 5.8A), el sistema detiene, y se tiene que verificar que

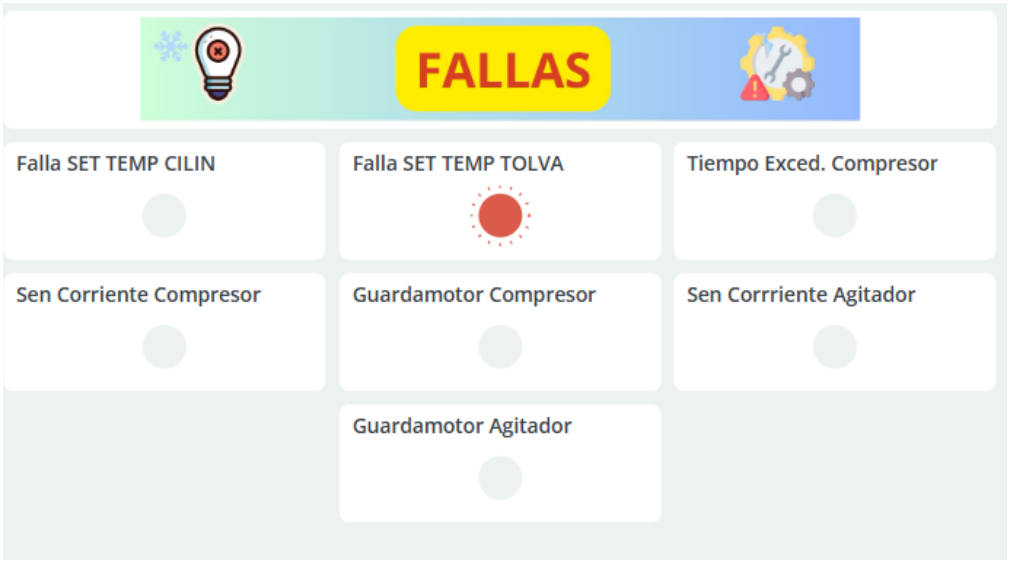
problema tiene, por lo general es necesaria una limpieza a la condensadora, donde se encuentra el ventilador.

3. **Guardamotor compresor:** Cumple la misma función que el sensor, pero este trabaja de manera que el guardamotor corta la alimentación al motor cuando este se encuentra en sobrecarga, de igual manera verificar la causa.
4. **Sensor sobre corriente agitador**: Cuando el sistema detecte una corriente encima de 7.9A (7.2A nominal) el sistema detiene al motor y se evidencia la alarma. Verificar la causa de sobrecarga, verificar banda, poleas, caja reductora si tiene aceite, máquina bien armada, producto no congelado en el cilindro, etc.
5. **Guardamotor agitador**, cumple la misma función, pero desconectando la parte de potencia del motor y parándolo de inmediato. Las causas pueden ser las mismas que el punto 4.
6. **Alarma por configuración incorrecta de temperatura del cilindro:** Cuando la diferencia de temperatura entre ON y OFF es menor que 13 unidades el sistema dará una alarma para verificar, con esto se consigue un producto final óptimo para venta.
7. **Alarma por configuración incorrecta de temperatura de las tolvas:** Cuando la diferencia de temperatura entre ON y OFF es menor que 6 unidades el sistema dará una alarma para verificar, con esto se consigue un producto bien mantenido en las tolvas para luego ser ingresado al cilindro.

EJEMPLOS DE ALARMAS EN LOGO TD (LOCAL)



EJEMPLO DE FALLA EN IOT REMOTE (Arduino Cloud)



CÓDIGO ARDUINO. Ino + thingPropiertes.h

```
#include "thingProperties.h"
#include <WiFi.h>
#include <ModbusIP_ESP8266.h>

// Datos de conexión Wi-Fi
const char* ssid = SECRET_SSID;
const char* password = SECRET_OPTIONAL_PASS;

// Configuración de IP estática
IPAddress local_IP(192, 168, 0, 99);
IPAddress gateway(192, 168, 0, 1);
IPAddress subnet(255, 255, 255, 0);
IPAddress primaryDNS(8, 8, 8, 8);
IPAddress secondaryDNS(8, 8, 4, 4);

// Variables de control del LED
unsigned long tiempoAnteriorLED = 0;
const long intervaloLED = 1000;
bool estadoLED = false;

// Verificación de fallos de seteo/////////////////////////////////////
bool falloSeteoCilindro = false;
unsigned long tiempoFalloCilindroAnterior = 0;
bool estadoFalloCilindro = false;

bool falloSeteoTolvas = false;
unsigned long tiempoFalloTolvasAnterior = 0;
bool estadoFalloTolvas = false;

const long intervaloFallo = 1000; // 1 segundo para intermitente


//////////////////////////////////////////////////////////////////////////////
// Modbus
ModbusIP mb;
const int regAnalogico = 0;
const int regAnalogico1 = 1;
const int regAnalogico2 = 2; // LEE VALOR DE LOGO! SET ON B091 CILINDRO HR: 3 EN LOGO!
const int regAnalogico3 = 3; // LEE VALOR DE LOGO! SET OFF B092 CILINDRO HR: 4 EN LOGO!
const int regAnalogico4 = 4; // LEE VALOR DE  LOGO! SET ON B097 TOLVAS   HR: 5 EN LOGO!
const int regAnalogico5 = 5; // LEE VALOR DE LOGO! SET OFF B098 TOLVAS   HR: 6 EN LOGO!
const int regAnalogico6 = 6; // LEE VALOR DE TEMPERATURA REAL  CILINDROS B079   HR: 7 EN LOGO!
const int regAnalogico7 = 7; // LEE VALOR DE TEMPERARURA REAL TOLVAS  B086   HR: 8 EN LOGO!
const int regAnalogico8 = 8; // LEE VALOR REAL DE CORRIENTE COMPRESOR  B0107   HR: 9 EN LOGO!
const int regAnalogico9 = 9; // LEE VALOR REAL DE CORRIENTE AGITADOR  B0108   HR: 10 EN LOGO!
const int regAnalogico10 = 10; //cont Lunes   AM3 HR:11 en LOGO!
const int regAnalogico11 = 11; //cont Martes  AM4 HR:12 en LOGO!
const int regAnalogico12 = 12; //cont Miercoles AM5 HR:13 en LOGO!
const int regAnalogico13 = 13; //cont Jueves AM6 HR:14 en LOGO!
const int regAnalogico14 = 14; //cont Viernes AM7 HR:15 en LOGO!
const int regAnalogico15 = 15; //cont Sabado AM8 HR:16 en LOGO!
const int regAnalogico16 = 16; //cont Domingo AM9 HR:17 en LOGO!
```

```
// Bobinas para NI (entradas digitales de LOGO!)
const int coilNI1 = 0;  //Modo wash desde Cloud
const int coilNI2 = 1;  //Modo Helado desde Cloud
const int coilNI3 = 2;  //Modo Keep Fresh desde Cloud
const int coilNI4 = 7;  // Modificar ON temperatura cilndro  desde Cloud  Coil=8 en LOGO!
const int coilNI5 = 8;  // Modificar OFF temperatura cilndro  desde Cloud Coil=9 en LOGO!
const int coilNI6 = 9;  // Modificar ON temperatura tolvas desde Cloud Coil=10 en LOGO!
const int coilNI7 = 10; // Modificar OFF temperatura tolvas desde Cloud Coil=11 en LOGO!
const int coilNI8 = 11;  // RESET desde Cloud

const int coilNI9 = 12;   // Entrada digital para LOGO! desde Cloud FALLA SET CILINDRO
const int coilNI10 = 13;  // Entrada digital para LOGO! desde Cloud  FALLA SET TOLVA


// Bobinas para NQ (salidas o marcas de LOGO!)
const int coilNQ1 = 3;  //
const int coilNQ2 = 4;
const int coilNQ3 = 5;
const int coilNQ4 = 6;  // Nueva salida NQ4
const int coilNQ5 = 14; //FALLA TIEMPO COMPRESOR EXCEDIDO A CLOUD
const int coilNQ6 = 15; //SENSOR SOBRECORRIENTE COMPRESOR a CLOUD
const int coilNQ7 = 16; //GUARDAMOTOR  SOBRECORRIENTE COMPRESOR A CLOUD
const int coilNQ8 = 17; //SENSOR SOBRECORRIENTE AGITADOR A CLOUD
const int coilNQ9 = 18; // GUARDAMOTOR  SOBRECORRIENTE AGITADOR A CLOUD
const int coilNQ10 = 19; //FALLA GENERAL A CLOUD

unsigned long tiempoAnteriorModbus = 0;
const long intervaloModbus = 500;
```

```
// Configuración de WiFi con IP estática
void setupWiFi() {
    if (!WiFi.config(local_IP, gateway, subnet, primaryDNS, secondaryDNS)) {
        Serial.println("Error al configurar IP estática");
    } else {
        Serial.println("IP estática configurada correctamente.");
    }

    WiFi.begin(ssid, password);
    Serial.print("Conectando a Wi-Fi");
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("\nConectado a Wi-Fi");
    Serial.print("IP asignada: ");
    Serial.println(WiFi.localIP());
}

// Verifica la conexión WiFi y reconecta si es necesario
void checkWiFiConnection() {
    if (WiFi.status() != WL_CONNECTED) {
        Serial.println("WiFi desconectado, intentando reconectar...");
        WiFi.disconnect();
        delay(1000);
        WiFi.begin(ssid, password);
    }
}

void setup() {
    Serial.begin(115200);
    delay(1500);

    setupWiFi();
```

```
    // Inicializar Arduino Cloud
    initProperties();
    ArduinoCloud.begin(ArduinoIoTPreferredConnection);
    setDebugMessageLevel(2);
    ArduinoCloud.printDebugInfo();

    // Configuración del LED
    pinMode(LED_BUILTIN, OUTPUT);
```

```
    // Configurar Modbus como servidor
    mb.server();
    mb.addHreg(regAnalogico, 0);
    mb.addHreg(regAnalogico1, 0);
    mb.addHreg(regAnalogico2, 0); //DATO SET  TEMP ON CILINDRO A CLOUD Agregar nuevo registro
    mb.addHreg(regAnalogico3, 0); //DATO SET TEMP OFF CILINDRO A CLOUD
    mb.addHreg(regAnalogico4, 0); //DATO SET TEMP ON TOLVAS A CLOUD
    mb.addHreg(regAnalogico5, 0); //DATO SET  TEMP OFF TOLVAS A CLOUD
    mb.addHreg(regAnalogico6, 0); // DATO TEMP REAL CILINDRO
    mb.addHreg(regAnalogico7, 0);// DATO TEMP REAL AGITADOR
    mb.addHreg(regAnalogico8, 0); //DATO REAL CORRIENTE COMPRESOR
    mb.addHreg(regAnalogico9, 0); // DATO REAL CORRIENTE AGITADOR
    mb.addHreg(regAnalogico10, 0); //
    mb.addHreg(regAnalogico11, 0); //
    mb.addHreg(regAnalogico12, 0); //
    mb.addHreg(regAnalogico13, 0); //
    mb.addHreg(regAnalogico14, 0); //
    mb.addHreg(regAnalogico15, 0); //
    mb.addHreg(regAnalogico16, 0); //

    mb.addCoil(coilNI1, false);
    mb.addCoil(coilNI2, false);
    mb.addCoil(coilNI3, false);
    mb.addCoil(coilNI4, false);
    mb.addCoil(coilNI5, false);
    mb.addCoil(coilNI6, false);
    mb.addCoil(coilNI7, false);
    mb.addCoil(coilNI8, false);
    mb.addCoil(coilNI9, false);
    mb.addCoil(coilNI10, false);
    mb.addCoil(coilNQ1, false);
    mb.addCoil(coilNQ2, false);
    mb.addCoil(coilNQ3, false);
    mb.addCoil(coilNQ4, false);
    mb.addCoil(coilNQ5, false);
    mb.addCoil(coilNQ6, false);
    mb.addCoil(coilNQ7, false);
    mb.addCoil(coilNQ8, false);
    mb.addCoil(coilNQ9, false);
    mb.addCoil(coilNQ10, false);


    Serial.println("🚀 Sistema Iniciado: ESP32 - LOGO! Modbus TCP y Arduino IoT Cloud.");
}

void loop() {
    checkWiFiConnection();
    ArduinoCloud.update();
    mb.task();

    unsigned long tiempoActual = millis();

    // Control del LED parpadeante (cada 1 segundo)
    if (tiempoActual - tiempoAnteriorLED >= intervaloLED) {
        tiempoAnteriorLED = tiempoActual;
        estadoLED = !estadoLED;
        digitalWrite(LED_BUILTIN, estadoLED);
        led = estadoLED;
    }

    // Leer valores analógicos del LOGO! cada 500ms
    if (tiempoActual - tiempoAnteriorModbus >= intervaloModbus) {
        tiempoAnteriorModbus = tiempoActual;

        int16_t rawValue0 = (int16_t)mb.Hreg(regAnalogico);
        int16_t rawValue1 = (int16_t)mb.Hreg(regAnalogico1);
        int16_t rawValue2 = (int16_t)mb.Hreg(regAnalogico2); // Nuevo valor leído
        int16_t rawValue3 = (int16_t)mb.Hreg(regAnalogico3); // Nuevo valor leído
        int16_t rawValue4 = (int16_t)mb.Hreg(regAnalogico4); // Nuevo valor leído
        int16_t rawValue5 = (int16_t)mb.Hreg(regAnalogico5); // Nuevo valor leído
        int16_t rawValue6 = (int16_t)mb.Hreg(regAnalogico6); // DATO TEMP REAL CILINDRO
        int16_t rawValue7 = (int16_t)mb.Hreg(regAnalogico7); //DATO TEMP REAL AGITADOR
        int16_t rawValue8 = (int16_t)mb.Hreg(regAnalogico8);
        int16_t rawValue9 = (int16_t)mb.Hreg(regAnalogico9);
        int16_t rawValue10 = (int16_t)mb.Hreg(regAnalogico10);
        int16_t rawValue11 = (int16_t)mb.Hreg(regAnalogico11);
        int16_t rawValue12 = (int16_t)mb.Hreg(regAnalogico12);
        int16_t rawValue13 = (int16_t)mb.Hreg(regAnalogico13);
        int16_t rawValue14 = (int16_t)mb.Hreg(regAnalogico14);
        int16_t rawValue15 = (int16_t)mb.Hreg(regAnalogico15);
        int16_t rawValue16 = (int16_t)mb.Hreg(regAnalogico16);
```

```
212
213           valorLogo = rawValue0;
214           valorLogo2 = rawValue2 / 10; // Escalar para mostrar en la nube dividido entre 10
215           valorLogo3 = rawValue3 / 10; // Escalar para mostrar en la nube dividido entre 10
216           valorLogo4 = rawValue4 / 10; // Escalar para mostrar en la nube dividido entre 10
217           valorLogo5 = rawValue5 / 10; // Escalar para mostrar en la nube dividido entre 10
218           valorLogo6 = (rawValue6 * 2.5 - 500.0) / 10.0; ///escalar para mostrar float
219           valorLogo7 = (rawValue7 * 2.5 - 500.0) / 10.0;
220           valorLogo8 = rawValue8 / 100.0;
221           valorLogo9 = rawValue9 / 100.0;
222           valorLogo10 = rawValue10;
223           valorLogo11 = rawValue11;
224           valorLogo12 = rawValue12;
225           valorLogo13 = rawValue13;
226           valorLogo14 = rawValue14;
227           valorLogo15 = rawValue15;
228           valorLogo16 = rawValue16;
229
230       // Verificar diferencia mínima entre SET ON y SET OFF - CILINDRO
231       int diferenciaCilindro = valorLogo2 - valorLogo3;
232       if (diferenciaCilindro < 13) {
233       falloSeteoCilindro = true;
234       } else {
235       falloSeteoCilindro = false;
236       mb.Coil(coilNI9, false); // Apagar intermitente si no hay fallo
237       }
238
239       if (diferenciaCilindro < 13) {
240       falloSeteoCilindro = true;
241       Serial.print("❌ FALLO SETEO CILINDRO: Diferencia insuficiente (");
242       Serial.print(diferenciaCilindro);
243       Serial.println("°C)");
244       } else {
245       falloSeteoCilindro = false;
246       Serial.print("✅ SETEO CILINDRO CORRECTO: Diferencia = ");
247       Serial.print(diferenciaCilindro);
248       Serial.println("°C");
249       mb.Coil(coilNI9, false);
250       }

252   // Verificar diferencia mínima entre SET ON y SET OFF - TOLVAS
253       int diferenciaTolvas = valorLogo4 - valorLogo5;
254       if (diferenciaTolvas < 6) {
255       falloSeteoTolvas = true;
256       } else {
257       falloSeteoTolvas = false;
258       mb.Coil(coilNI10, false); // Apagar intermitente si no hay fallo
259       }
260
261       if (diferenciaTolvas < 6) {
262       falloSeteoTolvas = true;
263       Serial.print("❌ FALLO SETEO TOLVAS: Diferencia insuficiente (");
264       Serial.print(diferenciaTolvas);
265       Serial.println("°C)");
266   } else {
267       falloSeteoTolvas = false;
268       Serial.print("✅ SETEO TOLVAS CORRECTO: Diferencia = ");
269       Serial.print(diferenciaTolvas);
270       Serial.println("°C");
271       mb.Coil(coilNI10, false);
272   }
273
274
275
276
277       // 🚨 Intermitente si hay fallo en CILINDRO
278       if (falloSeteoCilindro) {
279       unsigned long tiempoActual = millis();
280       if (tiempoActual - tiempoFalloCilindroAnterior >= intervaloFallo) {
281           tiempoFalloCilindroAnterior = tiempoActual;
282           estadoFalloCilindro = !estadoFalloCilindro;
283           mb.Coil(coilNI9, estadoFalloCilindro); // Salida hacia LOGO!
284       }
285       }
286
287
```

```
// 🚨 Intermitente si hay fallo en TOLVAS
    if (falloSeteoTolvas) {
    unsigned long tiempoActual = millis();
    if (tiempoActual - tiempoFalloTolvasAnterior >= intervaloFallo) {
        tiempoFalloTolvasAnterior = tiempoActual;
        estadoFalloTolvas = !estadoFalloTolvas;
        mb.Coil(coilNI10, estadoFalloTolvas); // Salida hacia LOGO!
    }
    }


        Serial.print("📝 Valor actual en LOGO! (regAnalogico0): ");
        Serial.println(valorLogo);
        Serial.print("📝 Valor actual en LOGO! (regAnalogico1 - con signo): ");
        Serial.println(rawValue1);
        Serial.print("📝 Valor actual en LOGO! (regAnalogico2): ");
        Serial.println(rawValue2); // Mostrar en consola
        Serial.print("📝 Valor actual en LOGO! (regAnalogico3): ");
        Serial.println(rawValue3); // Mostrar en consola
        Serial.print("📝 Valor actual en LOGO! (regAnalogico4): ");
        Serial.println(rawValue4); // Mostrar en consola
        Serial.print("📝 Valor actual en LOGO! (regAnalogico5): ");
        Serial.println(rawValue5); // Mostrar en consola
        Serial.print("📝 Valor actual en LOGO! (regAnalogico6): ");
        Serial.println(valorLogo6, 2); // Mostrar en consola
        Serial.print("📝 Valor actual en LOGO! (regAnalogico7): ");
        Serial.println(valorLogo7, 2); // Mostrar en consola
        Serial.print("📝 Valor actual en LOGO! (regAnalogico8): ");
        Serial.println(valorLogo8 , 2); // Mostrar en consola
        Serial.print("📝 Valor actual en LOGO! (regAnalogico9): ");
        Serial.println(valorLogo9 , 2); // Mostrar en consola
        Serial.print("📝 Valor actual en LOGO! (regAnalogico10): ");
        Serial.println(rawValue10); // Mostrar en consola
        Serial.print("📝 Valor actual en LOGO! (regAnalogico11): ");
        Serial.println(rawValue11); // Mostrar en consola
        Serial.print("📝 Valor actual en LOGO! (regAnalogico12): ");
        Serial.println(rawValue12); // Mostrar en consola
        Serial.print("📝 Valor actual en LOGO! (regAnalogico13): ");
        Serial.println(rawValue13); // Mostrar en consola
        Serial.print("📝 Valor actual en LOGO! (regAnalogico14): ");
        Serial.println(rawValue14); // Mostrar en consola
        Serial.print("📝 Valor actual en LOGO! (regAnalogico15): ");
        Serial.println(rawValue15); // Mostrar en consola
        Serial.print("📝 Valor actual en LOGO! (regAnalogico16): ");
        Serial.println(rawValue16); // Mostrar en consola
    }

    // 🔄 Verificar cambios en NQ1, NQ2, NQ3 y NQ4 , ETC, ETC  actualizar en la nube
    bool nuevoNQ1 = mb.Coil(coilNQ1);
    bool nuevoNQ2 = mb.Coil(coilNQ2);
    bool nuevoNQ3 = mb.Coil(coilNQ3);
    bool nuevoNQ4 = mb.Coil(coilNQ4);
    bool nuevoNQ5 = mb.Coil(coilNQ5);
    bool nuevoNQ6 = mb.Coil(coilNQ6);
    bool nuevoNQ7 = mb.Coil(coilNQ7);
    bool nuevoNQ8 = mb.Coil(coilNQ8);
    bool nuevoNQ9 = mb.Coil(coilNQ9);
    bool nuevoNQ10 = mb.Coil(coilNQ10);
    bool nuevoNI9 = mb.Coil(coilNI9);
    bool nuevoNI10 = mb.Coil(coilNI10);

    if (nuevoNQ1 != activarNQ1) {
        activarNQ1 = nuevoNQ1;
        Serial.print("🔄 Cambio detectado en NQ1: ");
        Serial.println(activarNQ1 ? "ACTIVADO" : "DESACTIVADO");
        ArduinoCloud.update();
    }

    if (nuevoNQ2 != activarNQ2) {
        activarNQ2 = nuevoNQ2;
        Serial.print("🔄 Cambio detectado en NQ2: ");
        Serial.println(activarNQ2 ? "ACTIVADO" : "DESACTIVADO");
        ArduinoCloud.update();
    }

    if (nuevoNQ3 != activarNQ3) {
        activarNQ3 = nuevoNQ3;
        Serial.print("🔄 Cambio detectado en NQ3: ");
        Serial.println(activarNQ3 ? "ACTIVADO" : "DESACTIVADO");
        ArduinoCloud.update();
    }

    if (nuevoNQ4 != activarNQ4) {
        activarNQ4 = nuevoNQ4;
        Serial.print("🔄 Cambio detectado en NQ4: ");
        Serial.println(activarNQ4 ? "ACTIVADO" : "DESACTIVADO");
        ArduinoCloud.update();
    }
```

```
        .

        if (nuevoNQ5 != activarNQ5) {
            activarNQ5 = nuevoNQ5;
            Serial.print("🔄 Cambio detectado en NQ5: ");
            Serial.println(activarNQ5 ? "ACTIVADO" : "DESACTIVADO");
            ArduinoCloud.update();
        }

        if (nuevoNQ6 != activarNQ6) {
            activarNQ6 = nuevoNQ6;
            Serial.print("🔄 Cambio detectado en NQ6: ");
            Serial.println(activarNQ6 ? "ACTIVADO" : "DESACTIVADO");
            ArduinoCloud.update();
        }
        if (nuevoNQ7 != activarNQ7) {
            activarNQ7 = nuevoNQ7;
            Serial.print("🔄 Cambio detectado en NQ7: ");
            Serial.println(activarNQ7 ? "ACTIVADO" : "DESACTIVADO");
            ArduinoCloud.update();
        }
        if (nuevoNQ8 != activarNQ8) {
            activarNQ8 = nuevoNQ8;
            Serial.print("🔄 Cambio detectado en NQ8: ");
            Serial.println(activarNQ8 ? "ACTIVADO" : "DESACTIVADO");
            ArduinoCloud.update();
        }
        if (nuevoNQ9 != activarNQ9) {
            activarNQ9 = nuevoNQ9;
            Serial.print("🔄 Cambio detectado en NQ9: ");
            Serial.println(activarNQ9 ? "ACTIVADO" : "DESACTIVADO");
            ArduinoCloud.update();
        }
        if (nuevoNQ10 != activarNQ10) {
            activarNQ10 = nuevoNQ10;
            Serial.print("🔄 Cambio detectado en NQ10: ");
            Serial.println(activarNQ10 ? "ACTIVADO" : "DESACTIVADO");
            ArduinoCloud.update();
        }
        if (nuevoNI9 != activarNI9) {
        activarNI9 = nuevoNI9;
        Serial.print("🔁 Estado NI9 actualizado en Cloud: ");
        Serial.println(activarNI9 ? "ACTIVO" : "INACTIVO");
        ArduinoCloud.update();
        }

        .

        if (nuevoNI10 != activarNI10) {
        activarNI10 = nuevoNI10;
        Serial.print("🔁 Estado NI10 actualizado en Cloud: ");
        Serial.println(activarNI10 ? "ACTIVO" : "INACTIVO");
        ArduinoCloud.update();
        }

    }
    // Callback: Escritura de la nube a LOGO!
    void onValorLogo1Change() {
        Serial.print("🌐 Nuevo valor recibido desde la nube en valorLogo1: ");
        Serial.println(valorLogo1);

        int16_t valorAEscribir = (int16_t)valorLogo1;
        int16_t valorActual = (int16_t)mb.Hreg(regAnalogico1);

        if (valorAEscribir != valorActual) {
            bool resultado = mb.Hreg(regAnalogico1, *(uint16_t*)&valorAEscribir);

            if (resultado) {
                Serial.println("✅ Modbus: Escrito en LOGO! correctamente.");
            } else {
                Serial.println("❌ Modbus: Error al escribir en LOGO!");
            }

            delay(50);

            int16_t valorEscrito = (int16_t)mb.Hreg(regAnalogico1);
            Serial.print("🔄 Verificación: Valor en regAnalogico1 después de escribir: ");
            Serial.println(valorEscrito);
        } else {
            Serial.println("⚠️ Valor en LOGO! ya es el mismo, no se escribe nuevamente.");
        }
    }

    // Callbacks para la nube (Entradas NI y Salidas NQ)
    void onActivarNI1Change() {
        Serial.print("⭕ Cambio de estado NI1 desde la nube: ");
        Serial.println(activarNI1);
        mb.Coil(coilNI1, activarNI1);
    }
```

```
463
464   void onActivarNI2Change() {
465       Serial.print("⭕ Cambio de estado NI2 desde la nube: ");
466       Serial.println(activarNI2);
467       mb.Coil(coilNI2, activarNI2);
468   }
469
470   void onActivarNI3Change() {
471       Serial.print("⭕ Cambio de estado NI3 desde la nube: ");
472       Serial.println(activarNI3);
473       mb.Coil(coilNI3, activarNI3);
474   }
475
476   void onActivarNI4Change() {
477       Serial.print("⭕ Cambio de estado NI4 desde la nube: ");
478       Serial.println(activarNI4);
479       mb.Coil(coilNI4, activarNI4);
480   }
481
482   void onActivarNI5Change() {
483       Serial.print("⭕ Cambio de estado NI5 desde la nube: ");
484       Serial.println(activarNI5);
485       mb.Coil(coilNI5, activarNI5);
486   }
487   void onActivarNI6Change() {
488       Serial.print("⭕ Cambio de estado NI6 desde la nube: ");
489       Serial.println(activarNI6);
490       mb.Coil(coilNI6, activarNI6);
491   }
492
493   void onActivarNI7Change() {
494       Serial.print("⭕ Cambio de estado NI7 desde la nube: ");
495       Serial.println(activarNI7);
496       mb.Coil(coilNI7, activarNI7);
497   }
498
499   void onActivarNI8Change() {
500       Serial.print("⭕ Cambio de estado NI8 desde la nube: ");
501       Serial.println(activarNI8);
502       mb.Coil(coilNI8, activarNI8);
503   }
504
505
506
507   void onActivarNI9Change() {
508       Serial.print("⭕ Cambio de estado NI9 desde la nube: ");
509       Serial.println(activarNI9);
510       mb.Coil(coilNI9, activarNI9);
511   }
512
513   void onActivarNI10Change() {
514       Serial.print("⭕ Cambio de estado NI10 desde la nube: ");
515       Serial.println(activarNI10);
516       mb.Coil(coilNI10, activarNI10);
517   }
518
519
520
521   void onActivarNQ1Change() {
522       Serial.print("⚡ Cambio de estado NQ1 desde la nube: ");
523       Serial.println(activarNQ1);
524       mb.Coil(coilNQ1, activarNQ1);
525   }
526
527   void onActivarNQ2Change() {
528       Serial.print("⚡ Cambio de estado NQ2 desde la nube: ");
529       Serial.println(activarNQ2);
530       mb.Coil(coilNQ2, activarNQ2);
531   }
532
533   void onActivarNQ3Change() {
534       Serial.print("⚡ Cambio de estado NQ3 desde la nube: ");
535       Serial.println(activarNQ3);
536       mb.Coil(coilNQ3, activarNQ3);
537   }
538
539   void onActivarNQ4Change() {
540       Serial.print("⚡ Cambio de estado NQ4 desde la nube: ");
541       Serial.println(activarNQ4);
542       mb.Coil(coilNQ4, activarNQ4);
543   }
544
545   void onActivarNQ5Change() {
546       Serial.print("⚡ Cambio de estado NQ5 desde la nube: ");
547       Serial.println(activarNQ5);
548       mb.Coil(coilNQ5, activarNQ5);
549   }
550
```

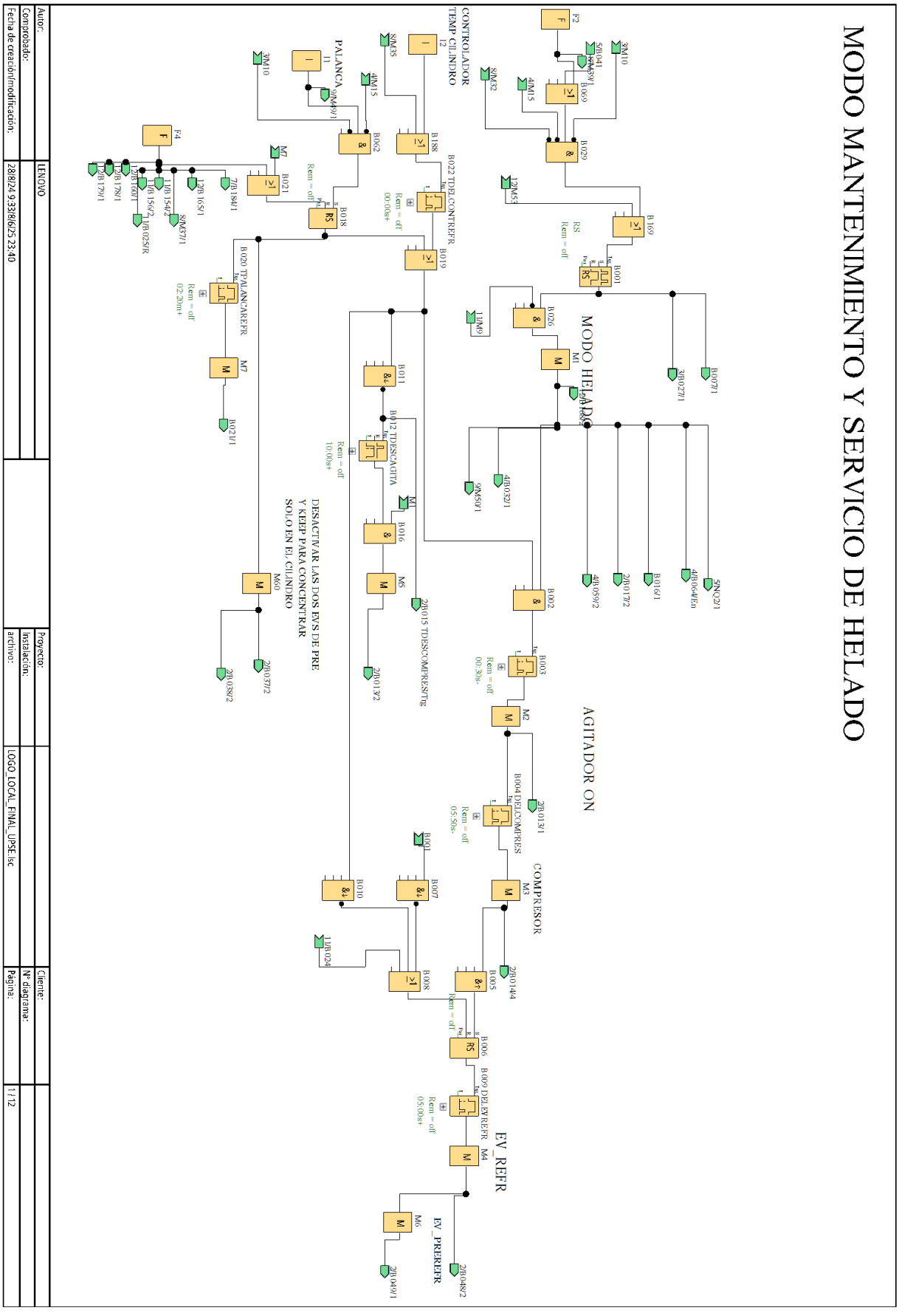
```
void onActivarNQ6Change() {
    Serial.print("⚡ Cambio de estado NQ6 desde la nube: ");
    Serial.println(activarNQ6);
    mb.Coil(coilNQ6, activarNQ6);
}

void onActivarNQ7Change() {
    Serial.print("⚡ Cambio de estado NQ7 desde la nube: ");
    Serial.println(activarNQ7);
    mb.Coil(coilNQ7, activarNQ7);
}

void onActivarNQ8Change() {
    Serial.print("⚡ Cambio de estado NQ8 desde la nube: ");
    Serial.println(activarNQ8);
    mb.Coil(coilNQ8, activarNQ8);
}

void onActivarNQ9Change() {
    Serial.print("⚡ Cambio de estado NQ9 desde la nube: ");
    Serial.println(activarNQ9);
    mb.Coil(coilNQ9, activarNQ9);
}

void onActivarNQ10Change() {
    Serial.print("⚡ Cambio de estado NQ10 desde la nube: ");
    Serial.println(activarNQ10);
    mb.Coil(coilNQ10, activarNQ10);
}

// Callback para el LED
void onLedChange() {
    Serial.print("💡 Led status changed: ");
    Serial.println(led);
    digitalWrite(LED_BUILTIN, led);
}
```

```
/*
  Since ValorLogo is READ_WRITE variable, onValorLogoChange() is
  executed every time a new value is received from IoT Cloud.
*/
void onValorLogoChange()  {
  // Add your code here to act upon ValorLogo change
}

/*
  Since ValorLogo2 is READ_WRITE variable, onValorLogo2Change() is
  executed every time a new value is received from IoT Cloud.
*/
void onValorLogo2Change()  {
  // Add your code here to act upon ValorLogo2 change
}
/*
  Since ValorLogo7 is READ_WRITE variable, onValorLogo7Change() is
  executed every time a new value is received from IoT Cloud.
*/
void onValorLogo7Change()  {
  // Add your code here to act upon ValorLogo7 change
}
/*
  Since ValorLogo8 is READ_WRITE variable, onValorLogo8Change() is
  executed every time a new value is received from IoT Cloud.
*/
void onValorLogo8Change()  {
  // Add your code here to act upon ValorLogo8 change
}
/*
  Since ValorLogo9 is READ_WRITE variable, onValorLogo9Change() is
  executed every time a new value is received from IoT Cloud.
*/
void onValorLogo9Change()  {
  // Add your code here to act upon ValorLogo9 change
}
```

thingPropiertes.h

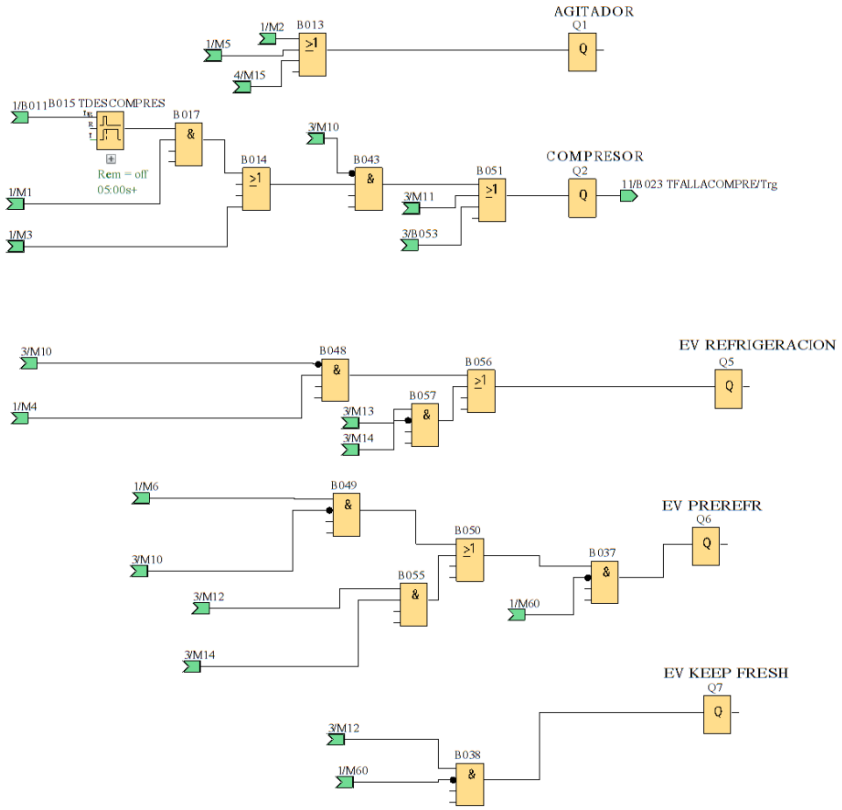
```
    ArduinoCloud.addProperty(valorLogo4, READ, ON_CHANGE, NULL);
    ArduinoCloud.addProperty(valorLogo5, READ, ON_CHANGE, NULL);
    ArduinoCloud.addProperty(activarNI1, READWRITE, ON_CHANGE, onActivarNI1Change);
    ArduinoCloud.addProperty(activarNI10, READWRITE, ON_CHANGE, onActivarNI10Change);
    ArduinoCloud.addProperty(activarNI2, READWRITE, ON_CHANGE, onActivarNI2Change);
    ArduinoCloud.addProperty(activarNI3, READWRITE, ON_CHANGE, onActivarNI3Change);
    ArduinoCloud.addProperty(activarNI4, READWRITE, ON_CHANGE, onActivarNI4Change);
    ArduinoCloud.addProperty(activarNI5, READWRITE, ON_CHANGE, onActivarNI5Change);
    ArduinoCloud.addProperty(activarNI6, READWRITE, ON_CHANGE, onActivarNI6Change);
    ArduinoCloud.addProperty(activarNI7, READWRITE, ON_CHANGE, onActivarNI7Change);
    ArduinoCloud.addProperty(activarNI8, READWRITE, ON_CHANGE, onActivarNI8Change);
    ArduinoCloud.addProperty(activarNI9, READWRITE, ON_CHANGE, onActivarNI9Change);
    ArduinoCloud.addProperty(activarNQ1, READ, ON_CHANGE, NULL);
    ArduinoCloud.addProperty(activarNQ10, READ, ON_CHANGE, NULL);
    ArduinoCloud.addProperty(activarNQ2, READ, ON_CHANGE, NULL);
    ArduinoCloud.addProperty(activarNQ3, READ, ON_CHANGE, NULL);
    ArduinoCloud.addProperty(activarNQ4, READWRITE, ON_CHANGE, onActivarNQ4Change);
    ArduinoCloud.addProperty(activarNQ5, READWRITE, ON_CHANGE, onActivarNQ5Change);
    ArduinoCloud.addProperty(activarNQ6, READ, ON_CHANGE, NULL);
    ArduinoCloud.addProperty(activarNQ7, READ, ON_CHANGE, NULL);
    ArduinoCloud.addProperty(activarNQ8, READ, ON_CHANGE, NULL);
    ArduinoCloud.addProperty(activarNQ9, READ, ON_CHANGE, NULL);
    ArduinoCloud.addProperty(led, READWRITE, ON_CHANGE, onLedChange);

  }

  WiFiConnectionHandler ArduinoIoTPreferredConnection(SSID, PASS);
void onActivarNQ5Change();
void onLedChange();

float valorLogo6;
float valorLogo7;
float valorLogo8;
float valorLogo9;
int valorLogo;
int valorLogo1;
int valorLogo10;
int valorLogo11;
int valorLogo12;
int valorLogo13;
int valorLogo14;
int valorLogo15;
int valorLogo16;
int valorLogo2;
int valorLogo3;
int valorLogo4;
int valorLogo5;
bool activarNI1;
bool activarNI10;
bool activarNI2;
bool activarNI3;
bool activarNI4;
bool activarNI5;
bool activarNI6;
bool activarNI7;
bool activarNI8;
bool activarNI9;
bool activarNQ1;
bool activarNQ10;
bool activarNQ2;
bool activarNQ3;
bool activarNQ4;
bool activarNQ5;
bool activarNQ6;
bool activarNQ7;
bool activarNQ8;
bool activarNQ9;
bool led;

void initProperties(){

  ArduinoCloud.setBoardId(DEVICE_LOGIN_NAME);
  ArduinoCloud.setSecretDeviceKey(DEVICE_KEY);
  ArduinoCloud.addProperty(valorLogo6, READ, ON_CHANGE, NULL);
  ArduinoCloud.addProperty(valorLogo7, READWRITE, ON_CHANGE, onValorLogo7Change);
  ArduinoCloud.addProperty(valorLogo8, READ, ON_CHANGE, NULL);
  ArduinoCloud.addProperty(valorLogo9, READ, ON_CHANGE, NULL);
  ArduinoCloud.addProperty(valorLogo, READWRITE, 1 * SECONDS, onValorLogoChange);
  ArduinoCloud.addProperty(valorLogo1, READWRITE, ON_CHANGE, onValorLogo1Change);
  ArduinoCloud.addProperty(valorLogo10, READ, ON_CHANGE, NULL);
  ArduinoCloud.addProperty(valorLogo11, READ, ON_CHANGE, NULL);
  ArduinoCloud.addProperty(valorLogo12, READ, ON_CHANGE, NULL);
  ArduinoCloud.addProperty(valorLogo13, READ, ON_CHANGE, NULL);
  ArduinoCloud.addProperty(valorLogo14, READ, ON_CHANGE, NULL);
  ArduinoCloud.addProperty(valorLogo15, READ, ON_CHANGE, NULL);
  ArduinoCloud.addProperty(valorLogo16, READ, ON_CHANGE, NULL);
  ArduinoCloud.addProperty(valorLogo2, READWRITE, ON_CHANGE, onValorLogo2Change);
  ArduinoCloud.addProperty(valorLogo3, READ, ON_CHANGE, NULL);
```

PROGRAMA de LOGO! SOFT COMFORT



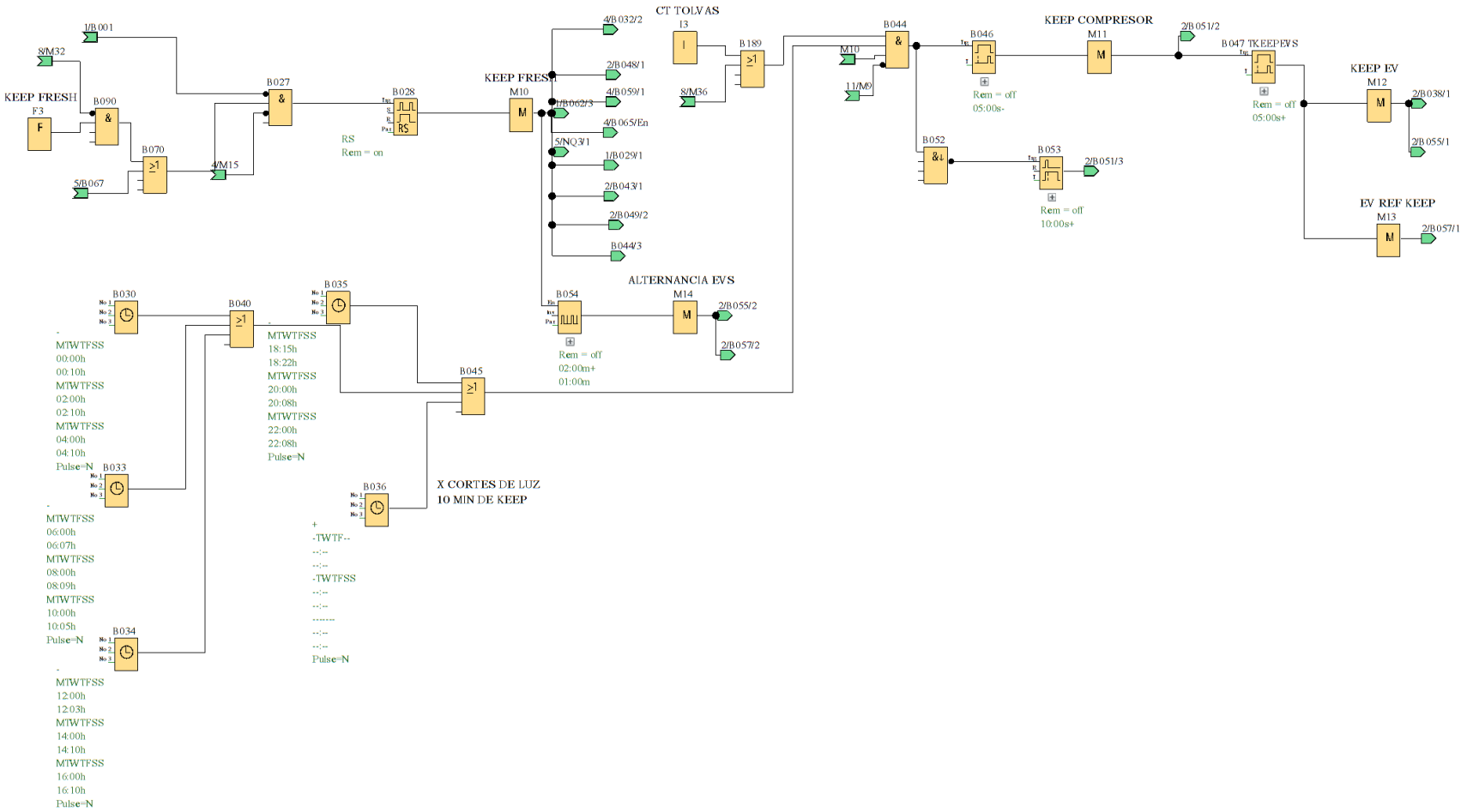
Autor:	LENOVO	Proyecto		Cliente:
Comprobado:		Instalación:		Nº diagrama:
Fecha de creación/modificación:	28/8/24 9:33/8/6/25 23:40	archivo:	LOGO_LOCAL_FINAL_UPSE.lsc	Página: 1/12

ACTIVACION DE ACTUADORES



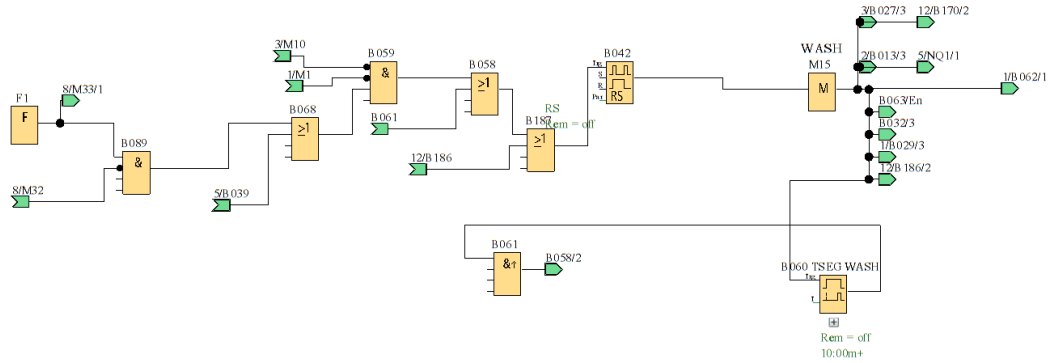
Autor:	LENOVO		Proyecto:		Cliente:	
Comprobado:			Instalación:		N° diagrama:	
Fecha de creación/modificación:	28/8/24 9:33/8/6/25 23:40		archivo:	LOGO_LOCAL_FINAL_UPSE.lsc	Página:	2 / 12

MODO KEEP FRESH (Mantener Producto Fresco)

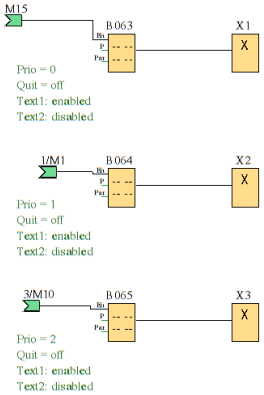


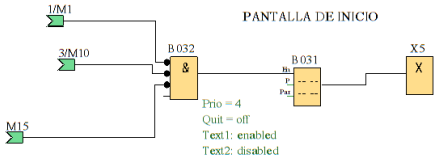
Autor:	LENOVO		Proyecto:		Cliente:	
Comprobado:			Instalación:		N° diagrama:	
Fecha de creación/modificación:	28/8/24 9:33/8/6/25 23:40		archivo:	LOGO_LOCAL_FINAL_UPSE.lsc	Página:	3 / 12

MODO WASH (Solo funciona el Agitador)



Mensajes de Modos de Operación

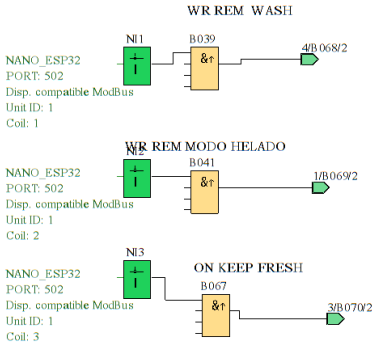




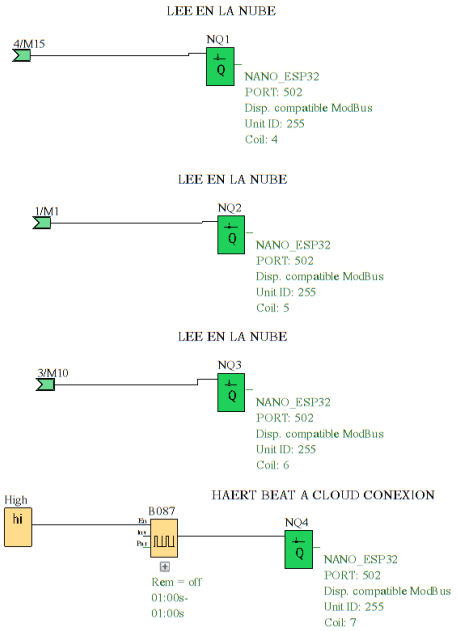
Autor:	LENOVO		Proyecto:		Cliente:		
Comprobado:			Instalación:		N° diagrama:		
Fecha de creación/modificación:	28/8/24 9:33/8/6/25 23:40		archivo:	LOGO_LOCAL_FINAL_UPSE.lsc	Página:	4 / 12	

Intercambio de Variables Nube-LOGO!; LOGO! - Nube)

MANDOS MODOS DE OPERACION Y
COMUNICACION DESDE LA NUBE CON ARDUINO

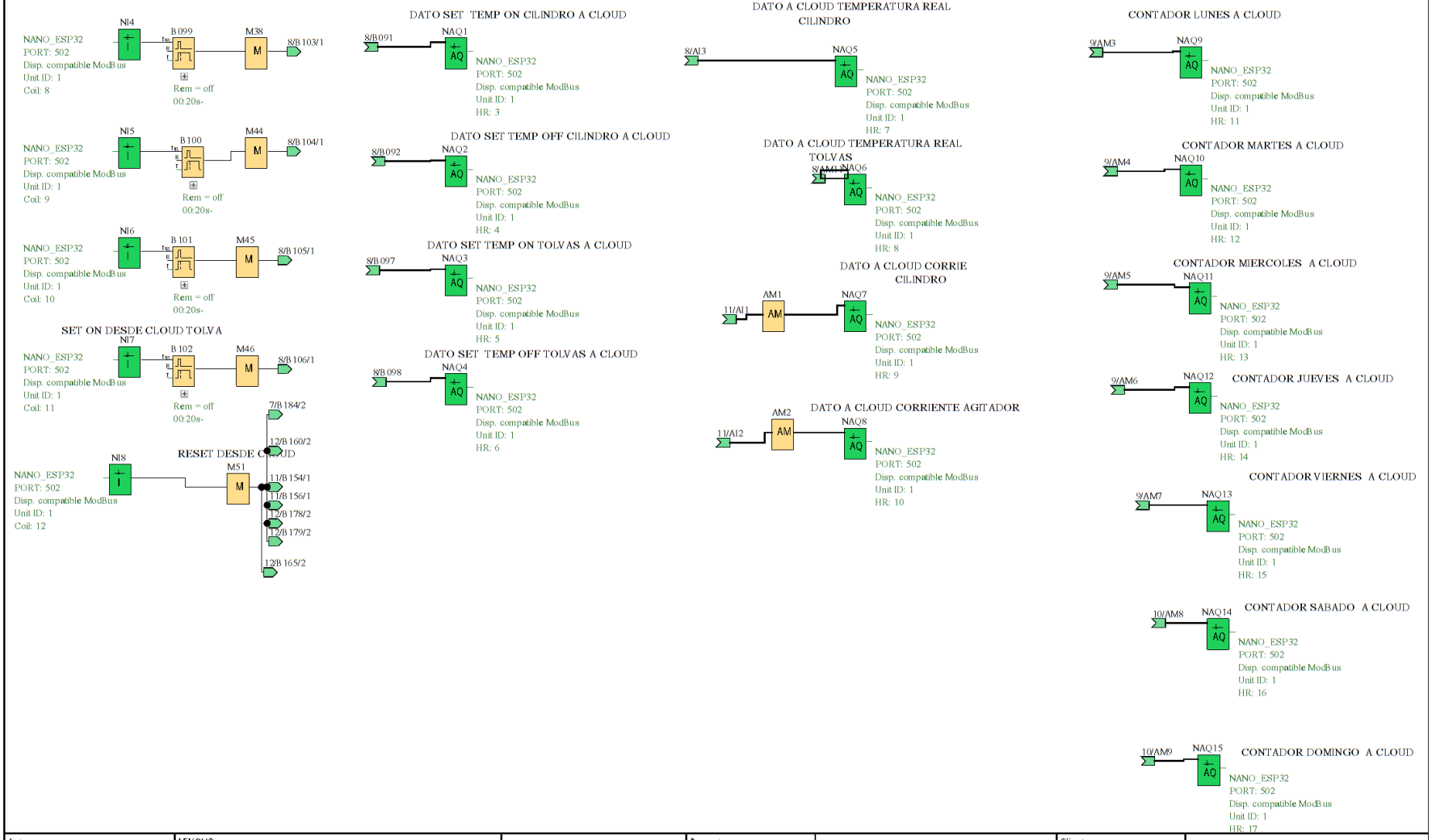


CONFIRMAR A CLOUD EL ESTADO DE MODO DE
OPERACION



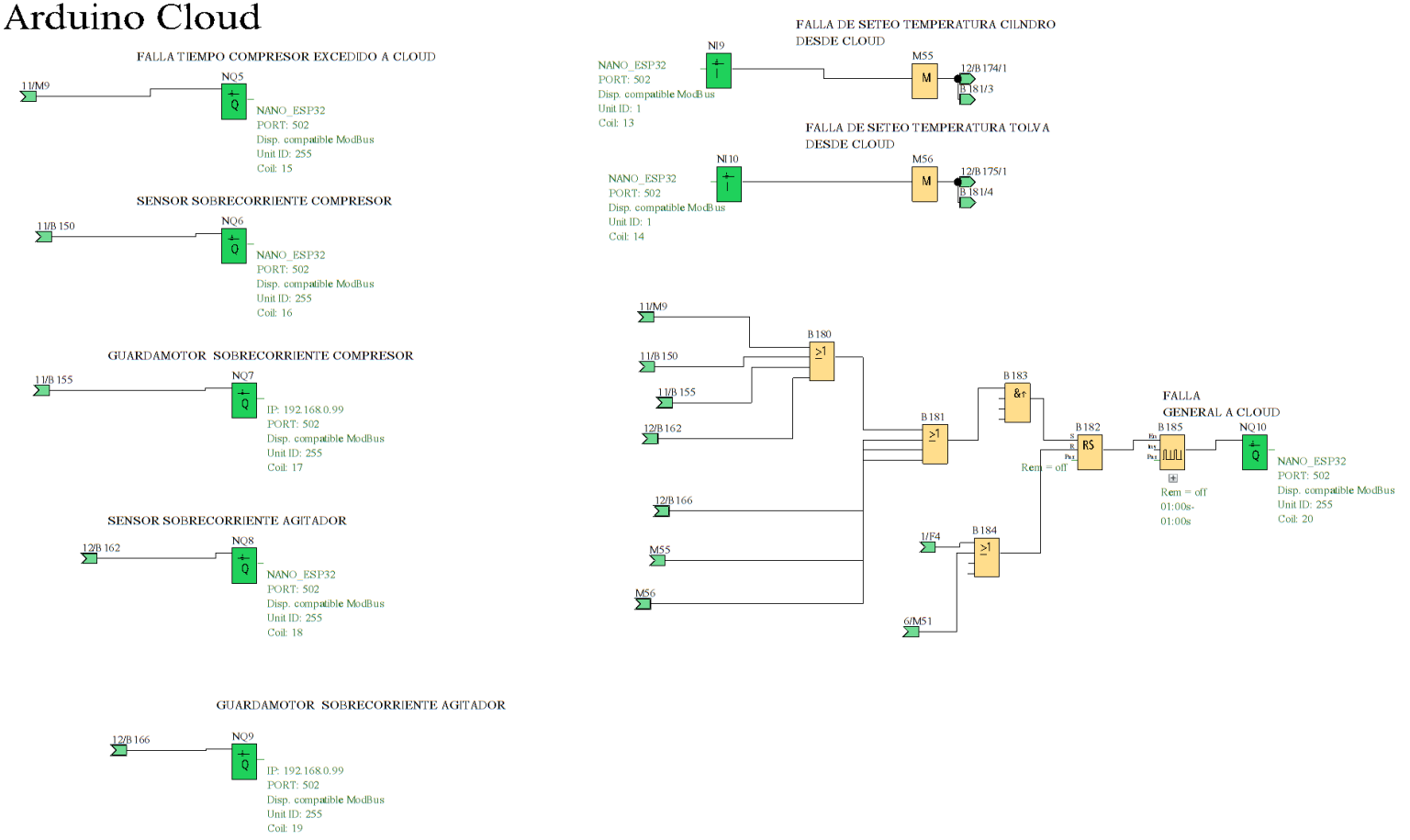
Autor:	LENOVO		Proyecto:		Cliente:		
Comprobado:			Instalación:		N° diagrama:		
Fecha de creación/modificación:	28/8/24 9:33/8/6/25 23:40		archivo:	LOGO_LOCAL_FINAL_UPSE.lsc	Página:	5 / 12	

Intercambio de Variables Nube-LOGO!; LOGO! - Nube)



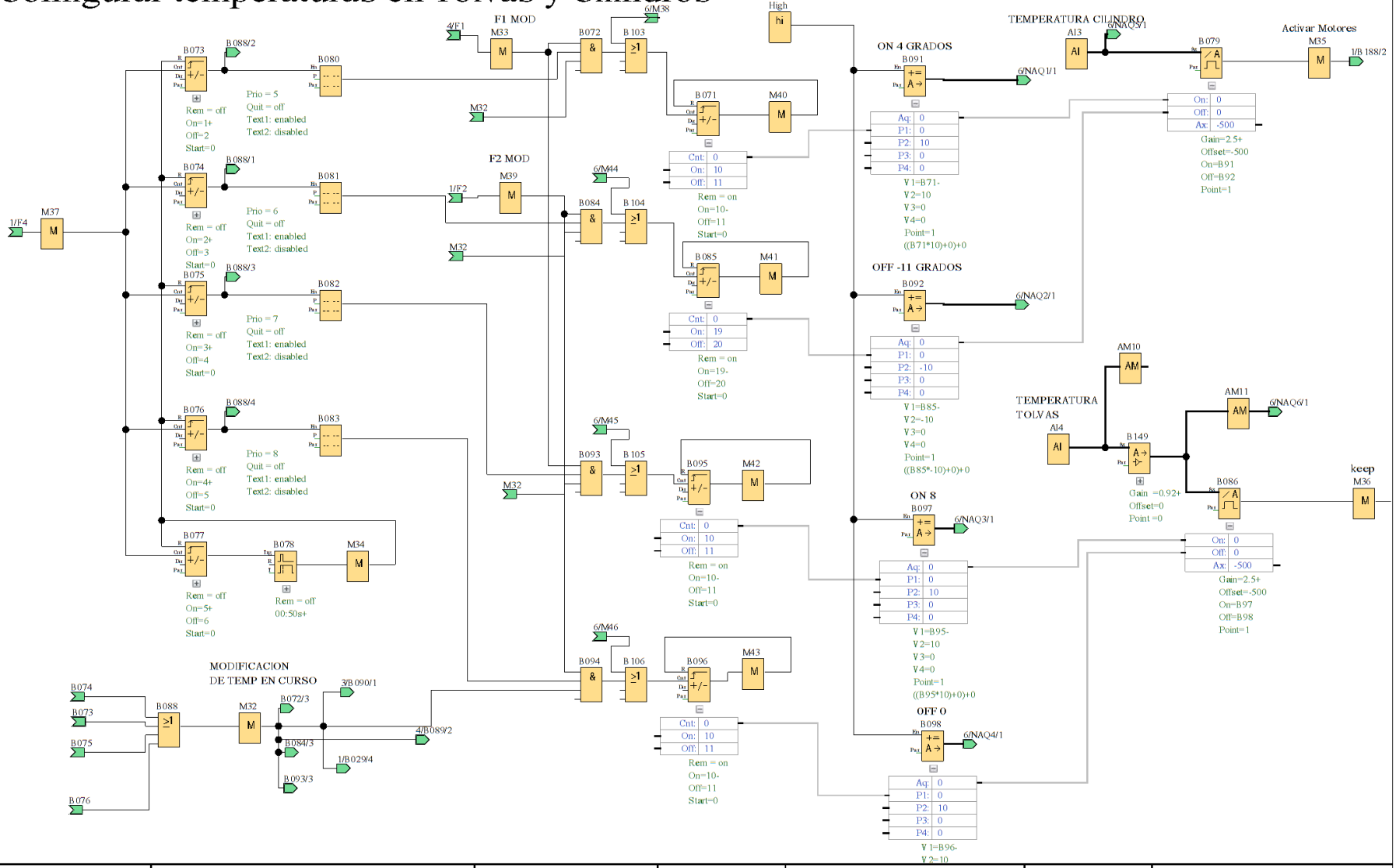
Autor:	LENOVO		Proyecto:		Cliente:	
Comprobado:			Instalación:		N° diagrama:	
Fecha de creación/modificación:	28/8/24 9:33/8/6/25 23:40		archivo:	LOGO_LOCAL_FINAL_UPSE.lsc	Página:	6 / 12

Fallas a Arduino Cloud



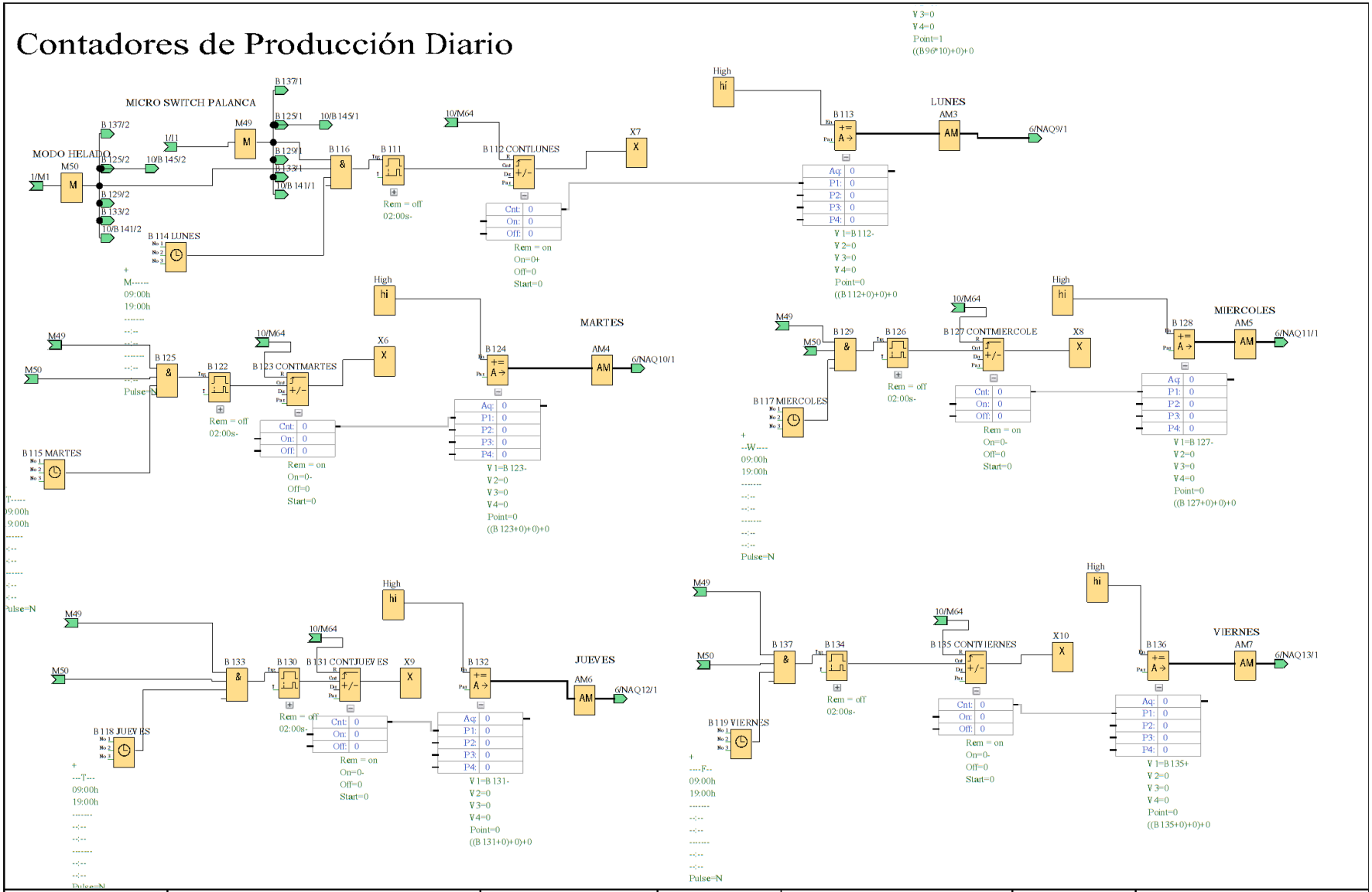
Autor:	LENOVO		Proyecto:		Cliente:	
Comprobado:			Instalación:		N° diagrama:	
Fecha de creación/modificación:	28/8/24 9:33/8/6/25 23:40		archivo:	LOGO_LOCAL_FINAL_UPSE.lsc	Página:	7 / 12

Configurar temperaturas en Tolvas y Cilindros



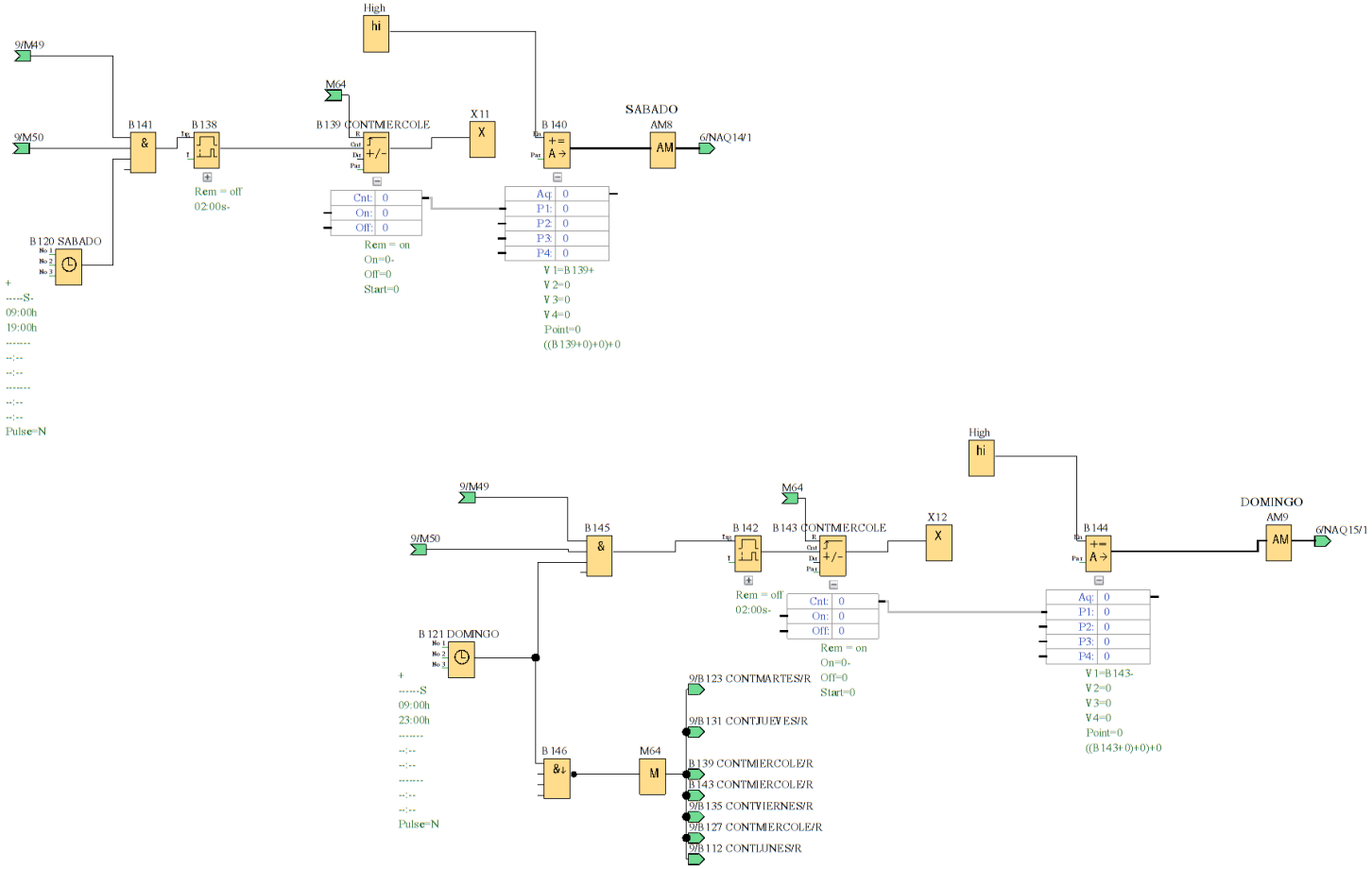
Autor:	LENOVO		Proyecto:		Cliente:		
Comprobado:			Instalación:		Nº diagrama:		
Fecha de creación/modificación:	28/8/24 9:33/8/6/25 23:40		archivo:	LOGO_LOCAL_FINAL_UPSE.lsc	Página:	8 / 12	

Contadores de Producción Diario



Autor:	LENOVO		Proyecto:		Cliente:		
Comprobado:			Instalación:		N° diagrama:		
Fecha de creación/modificación:	28/8/24 9:33/8/6/25 23:40		archivo:	LOGO_LOCAL_FINAL_UPSE.lsc	Página:	9 / 12	

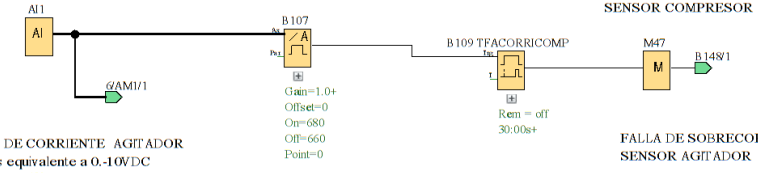
Contadores de Producción Diario



Autor:	LENOVO		Proyecto:		Cliente:		
Comprobado:			Instalación:		N° diagrama:		
Fecha de creación/modificación:	28/8/24 9:33/8/6/25 23:40		archivo:	LOGO_LOCAL_FINAL_UPSE.lsc	Página:	10 / 12	

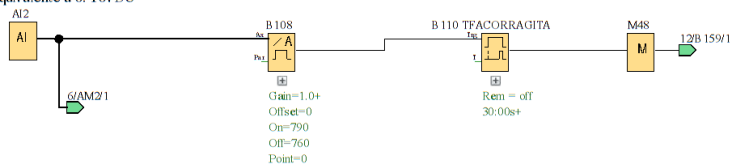
Alarmas

TRANSDUCTOR DE CORRIENTE COMPRESOR
0-10A medida es equivalente a 0.-10VDC



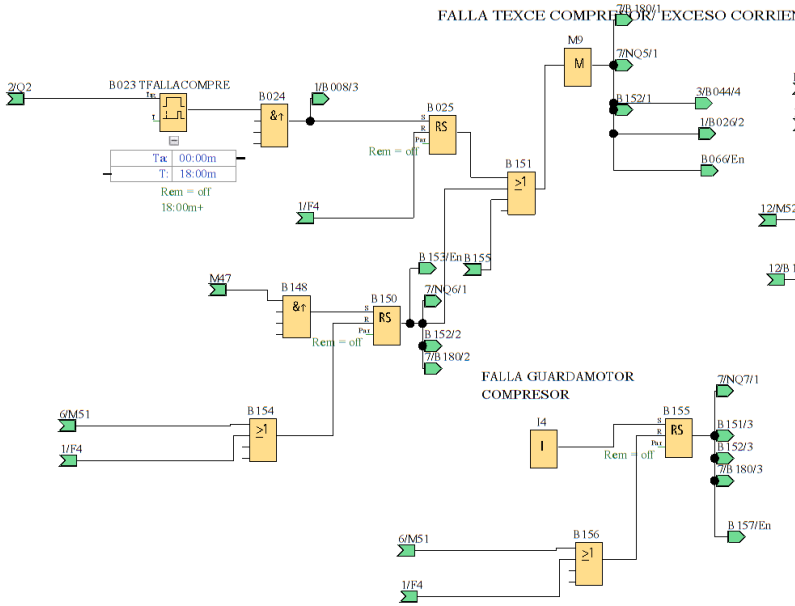
FALLA DE SOBRECORRIENTE POR SENSOR COMPRESOR

TRANSDUCTOR DE CORRIENTE AGITADOR
0-10A medida es equivalente a 0.-10VDC



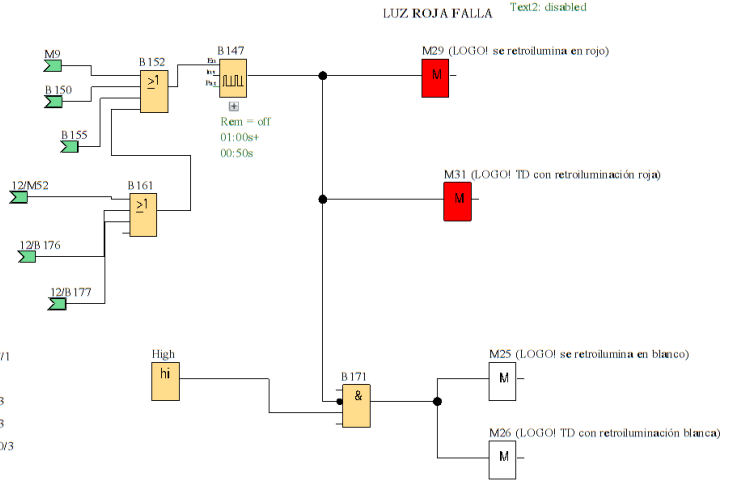
FALLA DE SOBRECORRIENTE POR SENSOR AGITADOR

FALLA TEXCE COMPRESOR/ EXCESO CORRIENTE

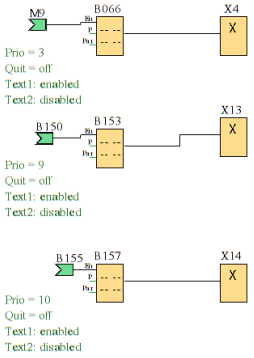


FALLA GUARDAMOTOR COMPRESOR

LUZ ROJA FALLA

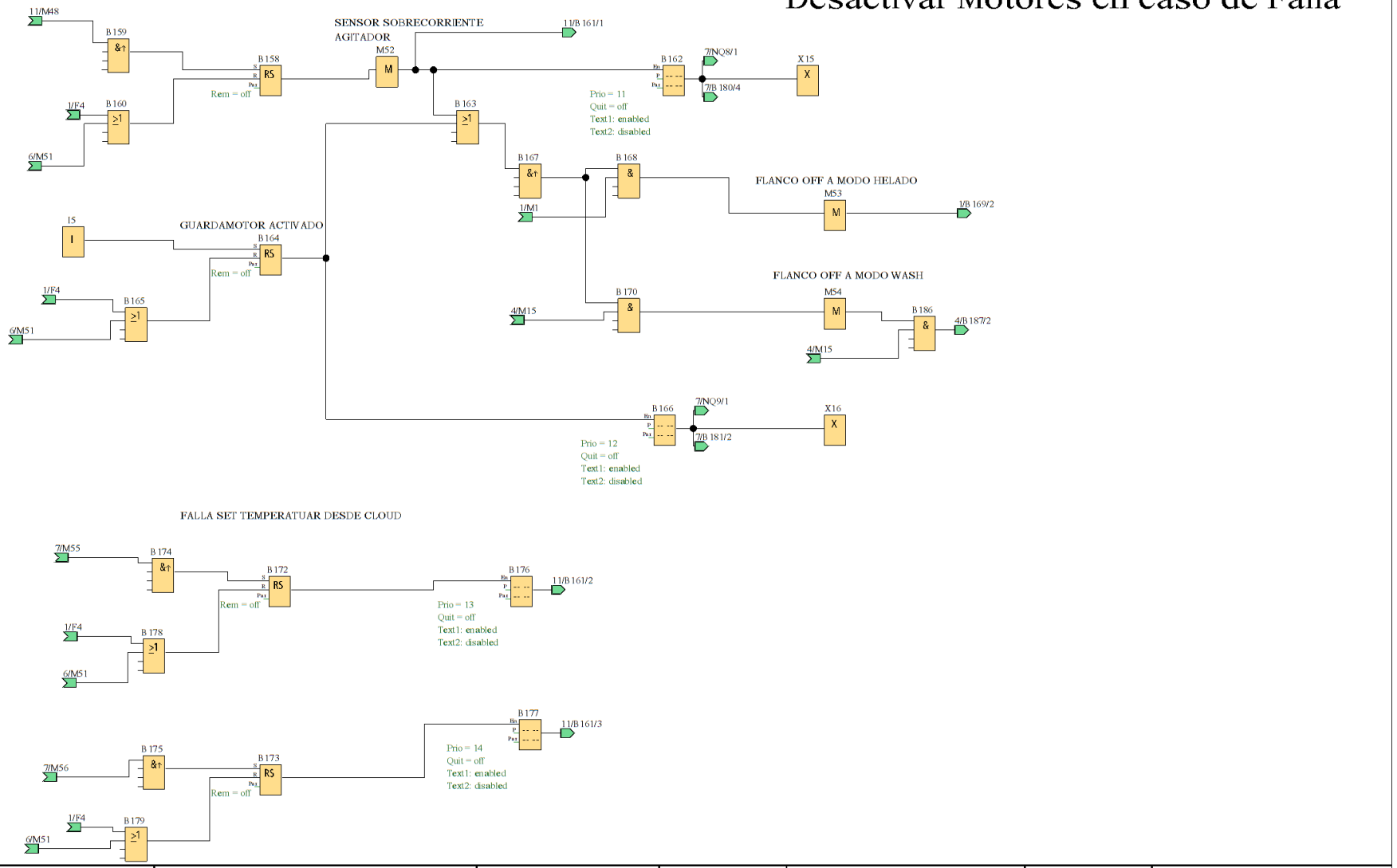


Mensajes de Alarma



Autor:	LENOVO		Proyecto:		Cliente:		
Comprobado:			Instalación:		N° diagrama:		
Fecha de creación/modificación:	28/8/24 9:33/8/6/25 23:40		archivo:	LOGO_LOCAL_FINAL_UPSE.lsc	Página:	11 / 12	

Desactivar Motores en caso de Falla



Autor:	LENOVO		Proyecto:		Cliente:		
Comprobado:			Instalación:		N° diagrama:		
Fecha de creación/modificación:	28/8/24 9:33/8/6/25 23:40		archivo:	LOGO_LOCAL_FINAL_UPSE.lsc	Página:	12 / 12	

DIAGRAMAS ELÉCTRICOS

F26_001-upse

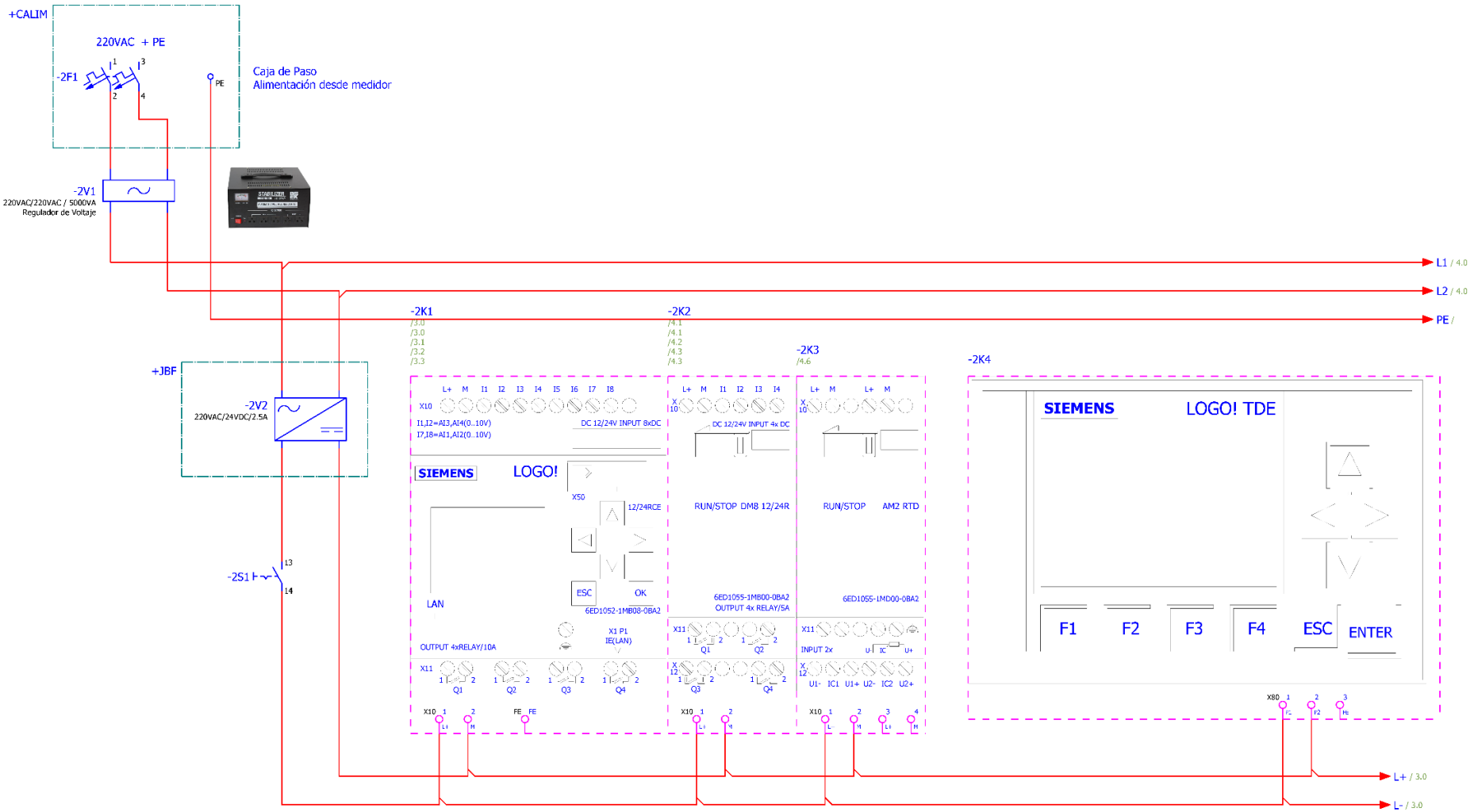
Universidad Estatal Península de Santa Elena

Empresa/cliente	FROSTY GAMER
Descripción de proyecto	AUTOMATIZACIÓN DE UNA MÁQUINA DE HELADO SOFT
Responsable del proyecto	LUIS IVAN CACHIGUANGO A
Lugar de instalación	QUITO, GUAYLLABAMBA

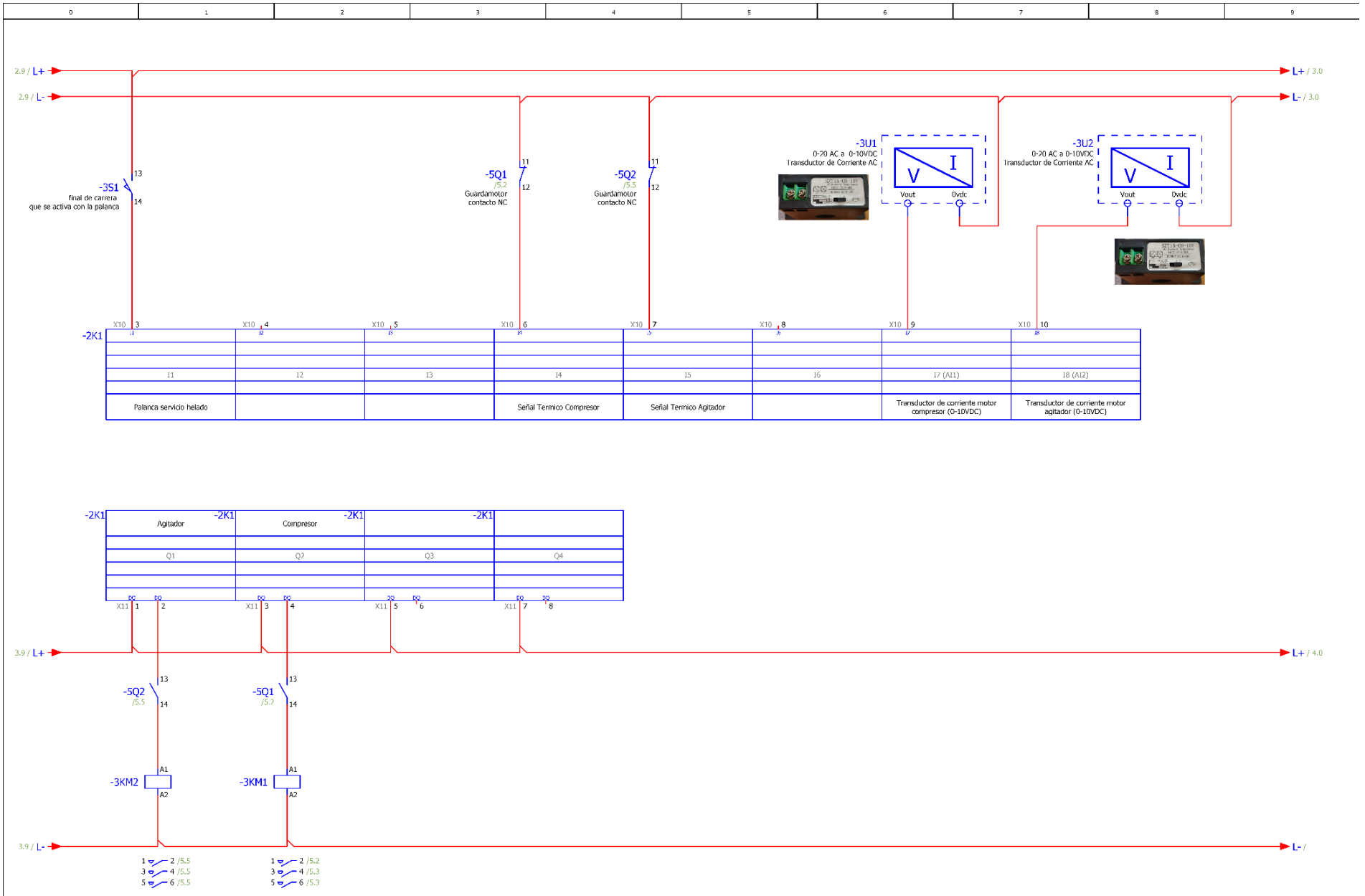
Creado	24/01/2025		
Modificado	08/04/2025	de (abreviatura) USUARIO	Número de páginas 6

2

			Fecha	08/04/2025	EPLAN	Frosty Gamer Heladería	Hoja de título		= AGEN
			Resp.	USUARIO					+ MSOFT
			Probado		AUTOMATIZACIÓN DE UNA MÁQUINA DE HELADO SOFT			UPSE_001	Hoja 1
Cambio	Fecha	Nombre	Original		Sustitución por	Sustituido por			Página 1 / 6

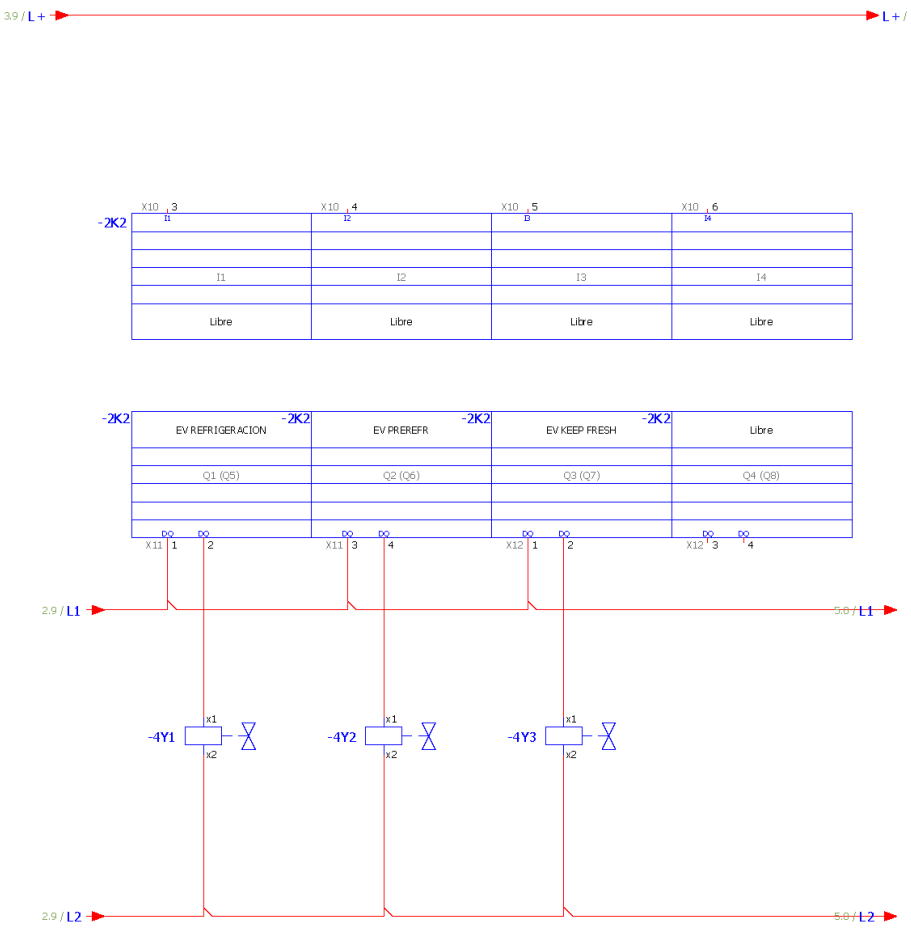


			Fecha	08/04/2025	EPLAN	Frosty Gamer Heladería	Alimentación de DC/AC	= AGEN	
			Resp.	USUARIO				+ MSOFT	
			Probado		AUTOMATIZACIÓN DE UNA MÁQUINA DE HELADO SOFT			UPSE 001	Hoja 2
Cambio	Fecha	Nombre	Original		Sustitución por	Sustituido por			Página 2 / 6

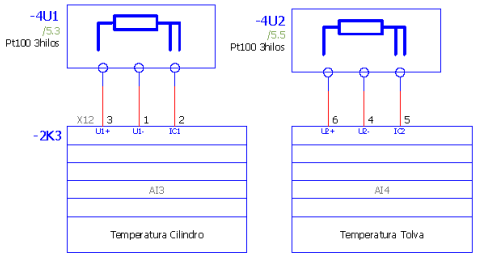


			Fecha	08/04/2025	EPLAN	Frosty Gamer Heladería	I/O CPU LOGO!		= AGEN
			Resp.	USUARIO					+ MSOFT
			Probado		AUTOMATIZACIÓN DE UNA MÁQUINA DE HELADO SOFT			UPSE_001	Hoja 3
Cambio	Fecha	Nombre	Original		Sustitución por	Sustituido por			Página 3 / 6

MODULO DM8

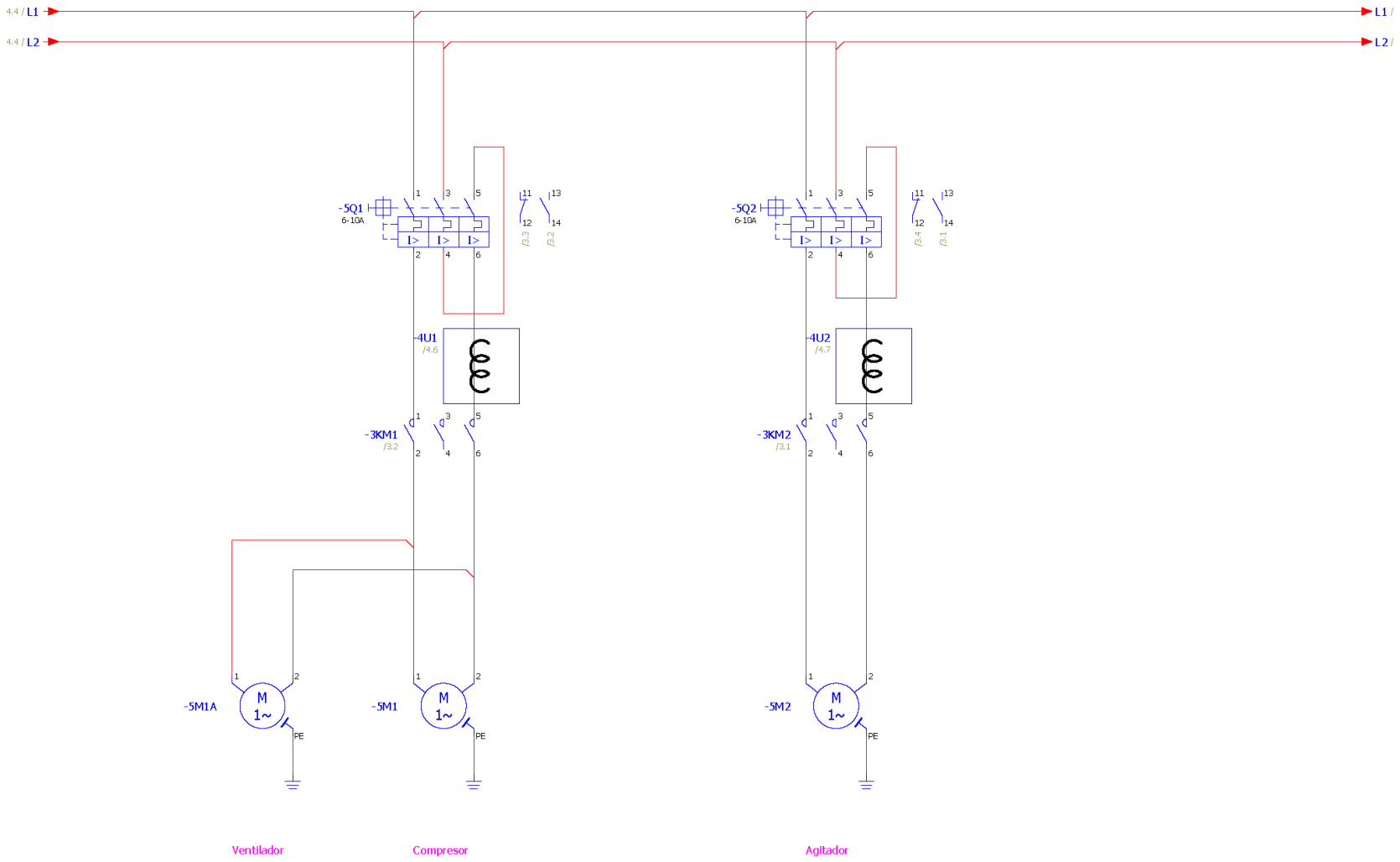


MODULO RTDs



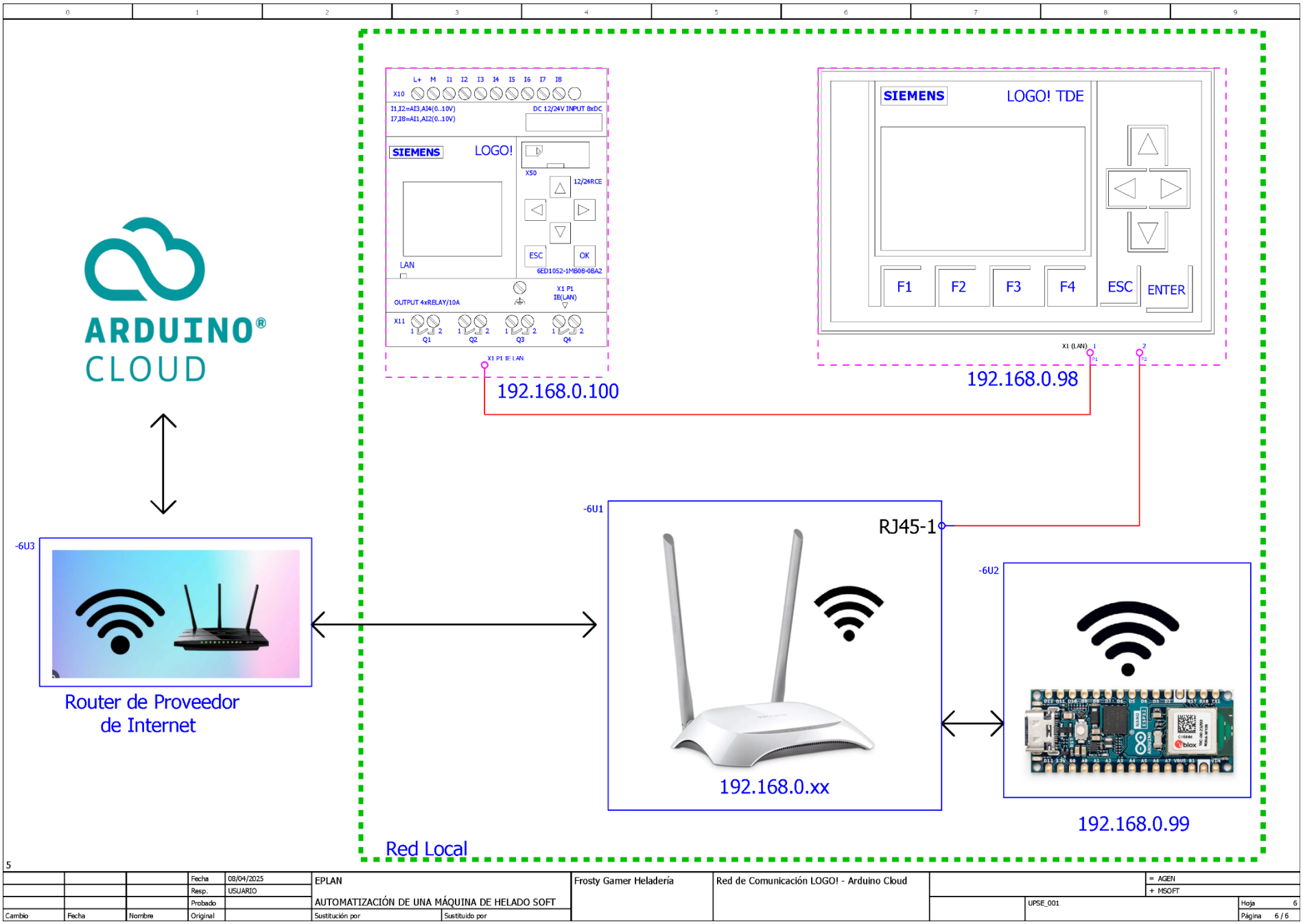
3 5

			Fecha	08/04/2025	EPLAN	Frosty Gamer Heladería	I/O DM8 y RTD MÓDULOS	= AGEN	
			Resp.	USUARIO				+ MSOFT	
			Probado		AUTOMATIZACIÓN DE UNA MÁQUINA DE HELADO SOFT			UPSE_001	Hoja 4
Cambio	Fecha	Nombre	Original		Sustitución por	Sustituido por			Página 4 / 6



4 6

			Fecha	08/04/2025	EPLAN	Frosty Gamer Heladería	Circuito de Fuerza	= AGEN
			Resp.	USUARIO				+ MSOFT
			Probado		AUTOMATIZACIÓN DE UNA MÁQUINA DE HELADO SOFT		UPSE_001	Hoja 5
Cambio	Fecha	Nombre	Original		Sustitución por / Sustituido por			Página 5 / 6


ARDUINO®
CLOUD
L+ M I1 I2 I3 I4 I5 I6 I7 I8
X10
I1,I2=AI3,AI4(0..10V)
I7,I8=AI1,AI2(0..10V)
DC 12/24V INPUT 8xDC
SIEMENS
LOGO!
X50
12/24RCE
ESC
OK
LAN
6ED1052-1MB08-0BA2
OUTPUT 4xRELAY/10A
X1 P1
IE(LAN)
X11
Q1
Q2
Q3
Q4
X1 P1 IE LAN
192.168.0.100
SIEMENS
LOGO! TDE
F1
F2
F3
F4
ESC
ENTER
X1 (LAN)
192.168.0.98
-6U3
Router de Proveedor
de Internet
-6U1
RJ45-1
192.168.0.xx
-6U2
192.168.0.99
Red Local
5
Fecha
08/04/2025
Resp.
USUARIO
Probado
Cambio
Fecha
Nombre
Original
EPLAN
AUTOMATIZACIÓN DE UNA MÁQUINA DE HELADO SOFT
Sustitución por
Sustituido por
Frosty Gamer Heladería
Red de Comunicación LOGO! - Arduino Cloud
= AGEN
+ MSOFT
UPSE_001
Hoja
6
Página
6 / 6

PLANILLA ELECTRICA

EMPRESA ELÉCTRICA QUITO S.A.

Empresa Eléctrica Quito S.A.E.E.Q.

Bartolome de las Casas E1-24 y Av. 10 de
Ruc: 1790053881001
Contribuyente especial, resolución No. 5368
OBLIGADO A LLEVAR CONTABILIDAD

Nro. Factura 001-999-111456215
Nro. doc. interno: 2412526579
Fecha emisión: 14-03-2025
Fecha de vencimiento: 13-04-2025
Número de autorización:
1403202501179005388100120019991114562150028437413

K200009543873

Fecha vencimiento	Valor Total
13 ABR 2025	38,44

INFORMACION DEL CONSUMIDOR

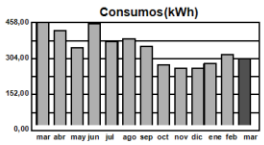
Cuenta contrato: 200009543873 **CUEN:** 1401211179
Nombre cliente: HARO LOPEZ ELVIA LEONOR (HARO LOPEZ ELVIA LEONOR)
Cédula: 0600099444 **Tarifa (ARCERNNR):** BTCGSD01 - BT Comercial
Geocodigo:
Unidad de Lectura 1408U503
Dirección AV SIMON BOLIVAR 489 PASAJE COMUNAL / GUAYLLABAMBA LA MERC PB 1

1. FACTURACION SERVICIO ELECTRICO Y ALUMBRADO PUBLICO

Medidor: 1002320979
Desde: 13-02-2025 **Hasta:** 13-03-2025 **Días:** 29
Tipo consumo: leído

DESCRIPCION	LECTURA ACTUAL	LECTURA ANTERIOR	CONSUMO MENSUAL	UNID.	VALORES
Energía activa total	7029,00	6747,00	282,00	kWh	22,84

HISTORIAL DE CONSUMOS



1.1 SERVICIO ELÉCTRICO (SE) Y SAPG

Valor Consumo	22,84
Comercialización	1,41
Subtotal Servicio Eléctrico (SE)	**24,25**
Servicio Alumbrado Público General	1,82
Subtotal Alumbrado Público (APG)	**1,82**
Base I.V.A. 0%	26,07
I.V.A. 0%	0,00
TOTAL SE Y AP (1)	**26,07**

SUBSIDIOS DEL GOBIERNO

Subsidio Tarifa Eléctrica	21,55-
TOTAL	21,55-

"La presente factura no constituye titulo traslaticio de dominio, sino únicamente la constancia de recibir un servicio público."

2. VALORES PENDIENTES

Concepto	Valores USD
VALORES PENDIENTES (2)	0,00

3. RECAUDACIÓN TERCEROS - PLANES DE FINANCIAMIENTO

Estos valores no forman parte de los ingresos de la Empresa Eléctrica

RECAUDACIÓN TERCEROS SECTOR ELÉCTRICO (3)	0,00

TOTAL A PAGAR

Servicio Eléctrico y Alumbrado Público (1)	26,07
Valores Pendientes (2)	0,00
Recaudación Terceros SE (3)	0,00
TOTAL SECTOR ELECTRICO (1) + (2) + (3): (A)	26,07

MENSAJES AL CONSUMIDOR

ORDEN DE COBRO POR CUENTA DE TERCEROS

LEY DEFENSA CONTRA INCENDIOS
C. BOMBEROS DISTRITO M QUITO
1768097950001

RUBROS/SUSTENTO	VALORES
Contribución Bomberos	7,05
Total ley de def. contra incendios (4)	**7,05**

Cuenta Contrato: 200009543873
Nombre: HARO LOPEZ ELVIA LEONOR
Cédula: 0600099444
Dirección: AV SIMON BOLIVAR 489 PASAJE
Fecha Emisión: 14-03-2025

ORDEN DE COBRO POR CUENTA DE TERCEROS

ORDENANZA MUNICIPAL
EMP METROPOLITANA DE ASEO
1768155310001

RUBROS/SUSTENTO	VALOR
Tasa de Recolección Basura	5,32
Total ordenanza municipal (5)	**5,32**

Cuenta Contrato: 200009543873
Nombre: HARO LOPEZ ELVIA LEONOR
Cédula: 0600099444
Dirección AV SIMON BOLIVAR 489 PASAJE
Fecha de Emisión: 14-03-2025

RESUMEN DE VALORES

Total Sector Eléctrico (Hoja 1) (A)	26,07
Total Recaudación por Cuenta Terceros (4)	12,37
VALOR TOTAL (A) + (4)	38,44

Formas de Pago

SIN UTILIZACION DEL SISTEMA FINANCIERO	26,07