Diseño de un sistema IoT para la monitorización de una microrred eléctrica experimental

Bryan Jesús López González y Oscar Díaz Sánchez

Trabajo de grado para optar al título de Ingenieros Electrónicos

Director

Juan Manuel Rey López

Doctor en Ingeniería Electrónica

Codirectora

Maria Alejandra Mantilla Villalobos

Doctora en Ingeniería

Codirector

Jhonathan Stiven Gómez Zuluaga

Ingeniero Electrónico

Universidad Industrial de Santander

Facultad de Ingenierías Fisicomecánicas

Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones

Ingeniería Electrónica

Bucaramanga

2023

Dedicatoria

A mi familia, con profundo agradecimiento por su amor, apoyo y sacrificio constante. Sin su guía, dedicación y paciencia, esta tesis no habría sido posible. Este logro es también el suyo. Muchas gracias por ser mi roca y mi faro en los momentos difíciles. Les quiero con todo mi corazón.

Oscar Díaz Sánchez

Dedicado a mis padres: Betsaida y David, quienes siempre creyeron en mí, incluso en los momentos en los que yo mismo tenía dificultades para hacerlo.

Bryan Jesús López González

Agradecimientos

A mis amigos (electropanas), les agradezco de corazón por su apoyo incondicional, su compañía y conocimiento en este camino. Han sido un gran apoyo y me han ayudado a mantenerme enfocado en mi objetivo.

A mi familia, les debo todo mi agradecimiento, su amor y apoyo ha sido fundamental para llevar a cabo esta tesis, no podría haberlo hecho sin ellos.

A Juan Manuel y Jonathan, quiero expresar mi más sincero agradecimiento por su orientación, paciencia y apoyo durante todo el proceso de investigación, su experiencia y conocimiento en el área fueron fundamentales para lograr completar esta tesis.

Oscar Díaz Sánchez

A mi familia y amigos, por brindarme un ambiente propicio para alcanzar este logro, por su apoyo constante durante todo el proceso y por hacer que estos cinco años fueran más amenos. Sin su amor y confianza, no habría sido posible llegar hasta aquí.

A mi compañero Oscar por su dedicación y compromiso en todas las etapas del proyecto; su colaboración ha sido fundamental para el éxito de este trabajo. A mi codirector Jonathan, por su mentoría y apoyo invaluable, sin los cuales habría sido imposible llevar a cabo este proyecto. Finalmente, a mi director Juan Manuel, por sus enseñanzas y experiencia valiosa.

Les agradezco a todos por su apoyo incondicional, y espero poder retribuirles en el futuro.

Bryan Jesús López González

Tabla de Contenido

Lista de Figuras

Lista de Tablas

Resumen

Titulo: Diseño de un sistema IoT para la monitorización de una microrred eléctrica experimental. [*]

Autores: Oscar Díaz Sánchez, Bryan Jesús López González. [**]

Director: Juan Manuel Rey López. Dr. Ingeniería Electrónica.

Co-Directores: Maria Alejandra Mantilla Villalobos, Dra. en Ingeniería. Jonathan Stiven Gómez Zuluaga, Ingeniero Electrónico.

Palabras Clave: Microrredes, IoT, IIoT, Industria 4.0.

Descripción: En el Laboratorio de Integración Energética (LIE) ubicado en el edificio de Investigaciones del Parque Tecnológico de Guatiguará se encuentra en desarrollo la construcción de una microrred eléctrica experimental, para la cual se solicita diseñar un sistema IoT que permita supervisar el comportamiento de variables eléctricas claves para los experimentos que se realizarán. Para diseñar este sistema, se llevará a cabo la selección adecuada de instrumentos de sensado y control, para elaborar una arquitectura de sistema IoT con el fin de comunicar algunos de los dispositivos que operan en la microrred, teniendo en cuenta las limitaciones de la misma y los requerimientos de investigación identificados en el LIE. Uno de ellos es la compatibilidad del sistema diseñado con el software Ignition 8.1 para la integración con distintos sistemas de la microrred.

---

[*] Trabajo de grado

[**] Facultad de Ingenierías Fisicomecánicas, Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones. Director Juan Manuel Rey Lopez. Codirectora María Alejandra Mantilla Villalobos. Codirector Jonathan Stiven Gómez Zuluaga.

Abstract

Title: Design of an IoT system for monitoring an experimental microgrid. [*]

Authors: Oscar Diaz Sanchez, Bryan Jesus Lopez Gonzalez. [**]

Director: Juan Manuel Rey Lopez, Ph.D in Electrical Engineering.

Co-directors: Maria Alejandra Mantilla Villalobos, Ph.D in Engineering. Jonathan Stiven Gómez Zuluaga, Electronic Engineer.

Keywords: Microgrids, IoT, IIoT, Industry 4.0.

Description: The Energy Integration Laboratory (LIE) located in the Research Building of the Guatiguará Technology Park is currently working on the development of an experimental microgrid. The goal of this degree final project is to design an IoT system that can monitor key electrical variables for the experiments that will be conducted. To design this system, the appropriate selection of sensing and control instruments will be made, and an IoT system architecture will be developed in order to connect some of the devices operating in the microgrid. The design will take into account the limitations of the microgrid and the research requirements identified by the LIE. One of the requirements is compatibility with the Ignition 8.1 software for integration with various systems within the microgrid.

---

[*] Bachelor Thesis

[**] Faculty of Physical and Mechanical Engineering, School of Electrical, Electronic and Telecommunications Engineering. Director Juan Manuel Rey Lopez. Codirector María Alejandra Mantilla Villalobos. Codirector Jonathan Stiven Gómez Zuluaga.

Introducción

Colombia, país ubicado en el norte de Sudamérica, caracterizado por su extensa y diversa cultura, la calidez de su gente y sus riquezas naturales, es un país que se encuentra en vías de desarrollo y que tiene un gigantesco potencial de crecimiento que puede ser alcanzado si se utilizan todos sus recursos de manera adecuada. Cuando se habla de recursos es inmediato pensar en aspectos como el dinero y aunque este, en efecto, es de suma importancia y puede determinar en gran medida las aspiraciones que se pueden tener en el proyecto de país, de nada sirve si no existe capital humano de calidad. Todos y cada uno de los habitantes de este país son el mayor activo presente en este mismo, son el agente que determina hacia donde se dirige la nación, son los que incurren día a día en actividades económicas y sociales y afectan directamente en los escenarios políticos encargados de representarles. El único elemento que de verdad puede hacer una diferencia en la calidad de todos estos factores mencionados que influyen en la vida de los colombianos es la educación y en qué proporción está presente la población colombiana en escenarios formativos que les permitan mejorar sus capacidades, sus competencias y contribuir en la construcción de conocimiento.

El mundo está cambiando de manera abrupta, el desarrollo tecnológico ha permitido la democratización de la información y esto ha acortado brechas educativas. Nunca en la historia de la humanidad había sido tan fácil acceder a la información y al conocimiento como lo es hoy en día. Sin embargo, para una gran parte de la población esto no es verdad, pues hay ciertos requerimientos mínimos que deben ser satisfechos para que los recursos de aprendizaje estén disponibles para

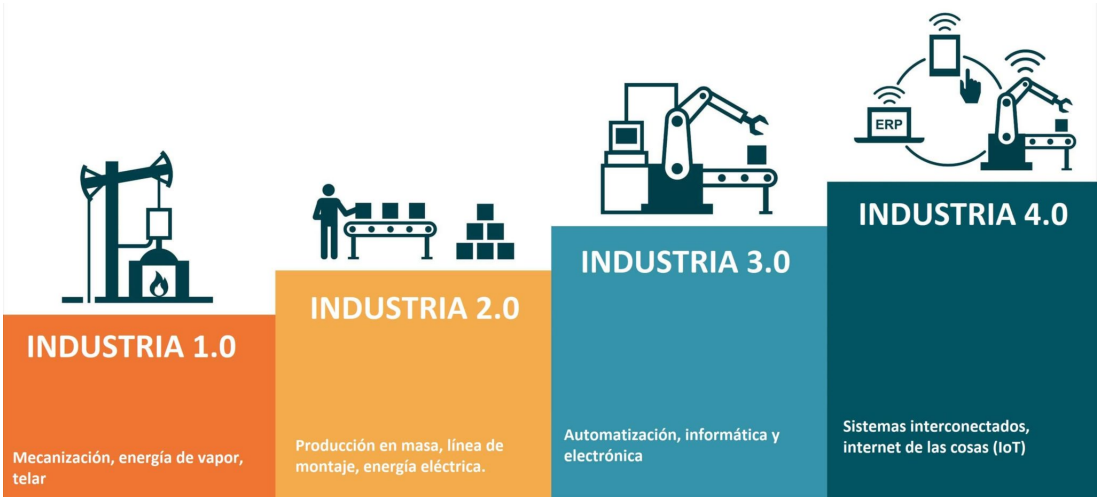
la gente, algunos de estos son: la disponibilidad de hardware que permita acceder a los sistemas de información (computadores o celulares), conexión a internet y un mínimo de ingresos económicos que permitan sostener un estilo de vida en donde las personas no están luchando por sobrevivir diariamente. En este punto, la relación entre estos requerimientos presentados y el bienestar económico de la sociedad es bastante evidente y una posible manera, no solo de satisfacerlos mínimamente, sino potencializarlos a sus máximos niveles, es a través de la inversión en tecnología. El grado en el que una sociedad acepta y fomenta el crecimiento tecnológico repercute directamente en la prosperidad de esta misma, la tecnología es un motor de crecimiento económico muy influyente en las naciones (Haussman and Domínguez, 2023). Colombia debe apuntar a esta dirección y tener la visión de volverse un país que no solamente sea cuna de emprendimientos de base tecnológica, sino que se caracterice por formar profesionales competentes en este contexto a través de la inversión en su capital humano.

En torno a esta temática se desarrolla la convocatoria 890 del Ministerio de Ciencias (Minciencias) cuyo propósito es: “Fortalecer las capacidades científicas, tecnológicas y de innovación en Instituciones de Educación Superior (IES) públicas a través de la conformación de un banco de proyectos elegibles cuyos resultados generen productos de nuevo conocimiento, desarrollo tecnológico, innovación y apropiación social del conocimiento que, a su vez, promuevan las competencias y habilidades en I+D+i de los estudiantes vinculados a los proyectos” (Ministerio de Ciencia, Tecnología e Innovación, 2021). La finalidad de este trabajo de grado presentado en modalidad de investigación, consiste en diseñar un sistema de monitoreo basado en IoT (Internet of Things o Internet de las cosas) para la microrred eléctrica experimental que se encuentra en cons-

trucción en el laboratorio de integración energética (LIE) del Parque Tecnológico de Guatiguará de la Universidad Industrial de Santander. Este sistema es de vital importancia, pues una vez esté terminado, servirá de base para la creación de una plataforma de aprendizaje en la que se entrenarán y capacitarán ingenieros colombianos en conceptos relativos a industria 4.0 no solo a nivel teórico, sino enfocado en escenarios realistas y en donde la tecnología puede agregar valor, es por esta razón que la microrred eléctrica fue escogida para ser parte de esta plataforma. La realización de este proyecto es un aporte hecho desde la ingeniería hacia ese proyecto de país en donde la formación profesional y la educación son prioridad en la sociedad colombiana.

Figura 1

Esquemático de las revoluciones industriales a lo largo de la historia. Una imagen para contextualizar este proyecto



Nota. Imagen extraída de (Lis Data Solutions, 2022).

1. Definiciones y conceptos básicos

El propósito de esta sección es presentar una serie de definiciones, en su mayoría asociadas a las tecnologías y elementos presentes en este proyecto, con la finalidad de que ayude a contextualizar y entender a cabalidad el diseño planteado.

1.1. Internet of Things (IoT)

El IoT (Internet of Things), es un concepto el cual se basa en la interconexión de dispositivos para que puedan intercambiar información mutuamente (Chaudhary et al., 2019). Es un campo con proyecciones de crecimiento y oportunidades a futuro en las áreas de ingeniería, ciencia y tecnología. Permite que los dispositivos se puedan conectar entre sí y que puedan reaccionar autónomamente a eventos del mundo real, tiene efectos no solamente en las maneras en las que se procesa la información, sino en los negocios y en aspectos sociales (Sorri et al., 2022) (Vermesan et al., 2011).

1.2. Microrred Eléctrica

Las microrredes eléctricas son sistemas de generación con capacidad de operar de manera autónoma o independiente a la red eléctrica. Las microrredes suelen integrar formas de generación distribuida de energía, especialmente de las renovables como paneles solares, aerogeneradores, entre otras. En (Alessia Cagnano, 2020) definen una microrred como: “una pequeña porción de un sistema de distribución energética con generación distribuida, dispositivos de almacenamiento energético y cargas controladas, lo cual puede dar lugar a un sistema energético autosuficiente”. Según esta referencia, la mayoría de las microrredes eléctricas funcionan de dos posibles maneras:

aisladamente o conectados a la red eléctrica. Las primeras pueden ser encontradas en áreas remotas donde por razones técnicas o económicas el tendido eléctrico no llega, las segundas son aquellas que funcionan en complemento al tendido eléctrico. Implementar estos sistemas implica ciertos desafíos a nivel técnico y operacional, muchos de ellos relativos al hecho de que su aplicación práctica se encuentra en etapas muy tempranas y es por ello por lo que hay una carencia de mano obra especializada en ellas. Uno de sus objetivos con su trabajo de investigación es dar a conocer distintas arquitecturas para el diseño de microrredes eléctricas, con el fin de acortar esta brecha de conocimiento existente y acelerar de esta manera el proceso de implementación práctica de estas mismas, por ser sistemas emergentes con futuro y beneficios tanto económicos como ambientales. Cabe destacar que en el contexto nacional, la matriz energética colombiana evidencia que más del 60 % de la capacidad instalada del país está basada en generadores hidroeléctricos, de hecho, es la sexta matriz energética más limpia en el mundo (Acolgen, 2023) por lo que, en el contexto nacional las microrredes son importantes más como sistemas secundarios o de auxilio para en casos en los que la red eléctrica principal falle. Esto no significa que no tengan otra posible aplicación en el escenario colombiano, debido a que, como ya fue presentado, las microrredes tienen la posibilidad de funcionar de manera aislada al sistema eléctrico, generando energía de manera independiente; es por esta característica que en las ZNI (Zona No Interconectada) las microrredes eléctricas tienen mucho potencial de mejorar la seguridad energética ((Consejo Internacional de Grandes Redes Eléctricas, 2020))

1.3. Modbus

Es un protocolo de comunicaciones desarrollado en 1979 por la empresa Modicon. Modbus específica como se debe transmitir la información entre los dispositivos actores los cuales son: Clientes y Servidores (también pueden denominarse Maestros y Esclavos). En Modbus un cliente (maestro) solicita información de un servidor (esclavo) y este responde con la información solicitada. Este protocolo tiene la ventaja de ser abierto, por lo cual los fabricantes pueden utilizarlo libremente y de forma gratuita, razón por la que es muy usado en la industria para conectar dispositivos a sistemas de control o sistemas de adquisición de datos. Modbus tiene diferentes versiones según el tipo especificación física con la que se implemente, en el contexto de este proyecto, al tratarse de comunicaciones seriales, se trabajó con Modbus RTU (Schneider Electric, 2022) (Modbus Organization, 2023).

Una trama de Modbus se compone de la dirección del servidor destino (1 byte), la función a ejecutar (1 byte), los datos (n bytes, dependiendo de la cantidad de datos) y el CRC (2 bytes). En donde la dirección es un valor numérico que representa al dispositivo en la red. La función es otro valor numérico que específica la acción esperada (en la tabla 1 se pueden ver algunas de las funciones que se pueden utilizar en Modbus). En Datos es donde se almacena la información (Rf Wireless World, 2012) y el CRC (de sus siglas en inglés: Cyclic Redundancy Check) es un algoritmo cuyo resultado son dos valores numéricos de 8 bits cada uno, computados a partir de la trama de información enviada/recibida, cuyo fin es detectar errores y/o corrupción de la información (IONOS, 2020).

Tabla 1
Algunas funciones de Modbus. Tabla obtenida de (General Electric, 2023)

Código de Función	Descripción	Máximo número de bobinas/registros
1	Leer estatus de bobina	2000
2	Leer estatus de entrada	2000
3	Leer registros de retención	125
4	Leer registros de entrada	125
5	Forzar bobina individual	1
6	Forzar registro individual	1
15(0x0F)	Forzar múltiples bobinas	800
16(0x10)	Preconfigurar múltiples registros	100

En Modbus una bobina se refiere a un bit, mientras que los registros son ubicaciones de memoria de 16 bits (2 bytes). Las funciones de Modbus utilizadas fueron: La 3 “Leer registwros de retención” y la 6 ‘’Forzar registro individual” las cuales fueron usadas para leer datos y escribir datos respectivamente.

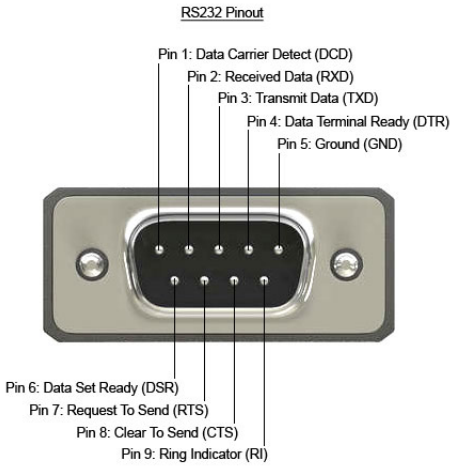
1.4. Comunicaciones Seriales (RS232)

Son una manera mediante la cual se puede intercambiar información en un medio físico (cableado). Existen varias especificaciones de comunicaciones seriales como UART, RS232 o RS485. Para este proyecto se utilizó la RS232. Una trama de esta especificación se compone de un bit que indica el inicio (start bit), luego siete u ocho dígitos de datos (el carácter como tal), después un bit opcional de paridad y finalmente uno o más bits de parada (stop bits). Los bits se transmiten del menos significativo al más significativo (Ibrahim, 2014). En la figura 2 se pueden ver las señales utilizadas en la especificación RS232, además de los datos que viajan en TXD y RXD, tiene otras

señales que soportan su funcionamiento. Las dSPACEs DS1104 cuentan con dos puertos seriales, uno para RS232 y el otro para RS422/RS485, el primero de estos fue utilizado en cada dSPACE para conectarse al EPIC con un concentrador USB.

Figura 2

Pines y señales utilizadas por RS232 en un conector DB9



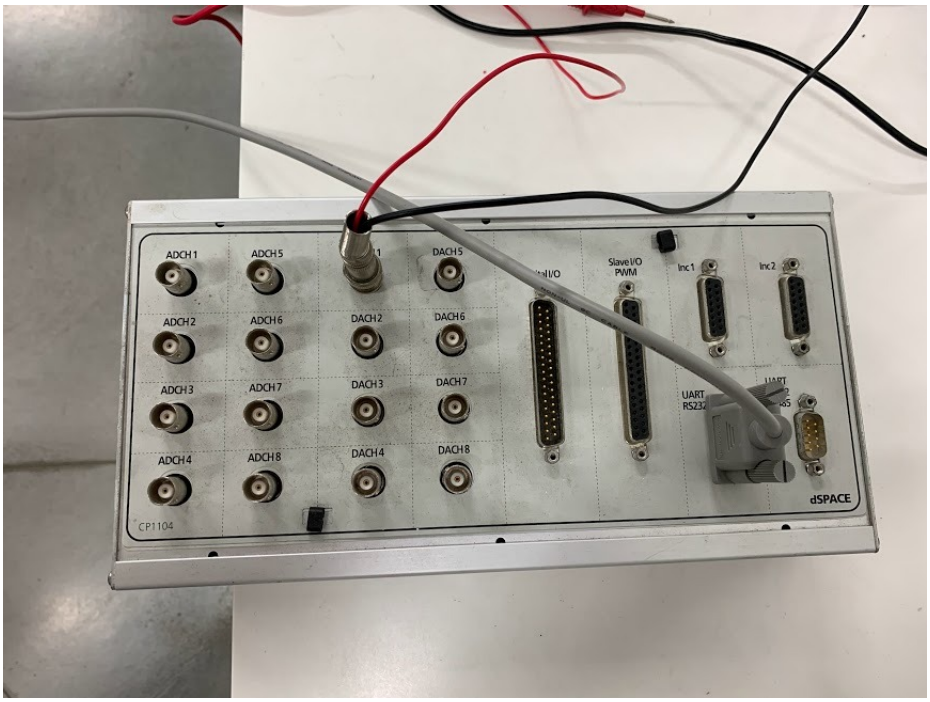
Nota. Imagen extraída de (Punto Flotante S.A., 2021).

1.5. dSPACE

Estas tarjetas son sistemas embebidos que cuentan con ADCs, DACs, salidas PWM, entradas y salidas digitales (dSPACE, 2022). Las dSPACE se encuentran instaladas en los computadores del LIE y son los dispositivos de los cuales se monitorea las lecturas de los ADCs y se modifican las tensiones de los DACs desde Ignition. Nótese que en la figura 3 lo que se presenta es el módulo de conexiones de la dSPACE, la tarjeta controladora se encuentra embebida en la tarjeta madre de los computadores del LIE.

Figura 3

Foto de un módulo de conexión para dSPACE presente en el LIE



1.6. Matlab y Simulink

Son productos distribuidos por la empresa Mathworks. Matlab es una plataforma de programación cuyo núcleo es su lenguaje de programación (MathWorks, 2023a). Simulink es un entorno de desarrollo utilizado para diseñar sistemas con diagramas de bloques (MathWorks, 2023b). La relación entre estas dos plataformas, aplicada a este desarrollo, es que en Simulink se describe el comportamiento deseado de las dSPACE usando los diagramas de bloques en donde, algunos de estos pueden contener código desarrollado en Matlab. En este proyecto se utilizó una implementación de Modbus en Simulink (Starynets, 2016) para usarse como protocolo de comunicaciones entre el EPIC y las dSPACE.

1.7. Controldesk

Es el software de las dSPACE el cual permite explotar diversas funcionalidades de las tarjetas (dSPACE, 2023). En el contexto de este proyecto, cuando se describe el comportamiento de las dSPACE en Simulink, este modelo debe construirse en lenguaje C, el resultado de esta construcción se carga a ControlDesk y de aquí el código se carga a las dSPACE.

1.8. Opto 22 EPIC (GRV-EPIC-PR2)

Específicamente modelo EPIC (**E**dge **P**rogrammable **I**ndustrial **C**ontroller) de la marca Opto 22 es un computador industrial (Opto 22, 2023), con distintos periféricos y módulos, el cual es utilizado como la unidad central de procesamiento del proyecto. Es el dispositivo que recoge los datos seriales de las dSPACE, procesa los datos de estas mismas en Node-RED y los envía a Ignition en el formato requerido por esta plataforma. En la figura 4 se puede ver el EPIC presente en el tablero IoT del LIE.

1.9. Node-RED

Es una herramienta de programación visual basada en NodeJs (Javascript) la cual es utilizada para interconectar dispositivos, servicios web y APIs (Node-RED, 2023). Node-RED está basado en web por lo que se utiliza dentro de cualquier navegador; esta herramienta permite, a través de bloques de código, realizar procesamiento de los datos y especificar el destino de estos mismos. En el contexto de este proyecto fue utilizado en el EPIC para recibir los datos seriales de las dSPACEs y enviarlos a Ignition.

Figura 4

Foto del computador industrial EPIC presente en el LIE

1.10. Ignition 8.1

Es una plataforma distribuida por la empresa Inductive Automation la cual puede implementarse como HMI (Human Machine Interface), SCADA (Supervisory Control and Data Acquisition) o MES (Manufactuting Execution System). Ignition es una plataforma que permite conectar, integrar y escalar hardware en sistemas complejos de monitoreo, control y visualización de datos. Ignition es muy flexible y potente, en el contexto de este proyecto fue utilizado para la realización de parte del backend y el frontend de la aplicación final en donde se pueden visualizar y modificar los datos de los ADCs y los DACs de cada dSPACE.

2. Objetivos

2.1. Objetivo General

Diseñar un sistema IoT para la monitorización de variables eléctricas en la microrred experimental del Laboratorio de Integración Energética.

2.2. Objetivos Específicos

- Determinar los requerimientos solicitados del LIE para el monitoreo y control de las variables eléctricas de la microrred.
- Identificar y definir los elementos, arquitecturas y dispositivos necesarios en el sistema de monitorización IoT.
- Verificar el funcionamiento del sistema de monitoreo y control IoT para futuros experimentos del grupo de investigación en la microrred.

3. Requerimientos del LIE

Obtener los requerimientos es un proceso que requiere tanto de entender las necesidades de los interesados como de una caracterización del caso de estudio previa a la ejecución del trabajo. En particular, luego de sostener reuniones con el propósito de contextualizar el proyecto y sus necesidades, se identificaron tres interesados:

- GISEL (Grupo de Investigación en Sistemas de Energía Eléctrica).
- Dautom (Diseño y Automatización Industrial SAS)
- LIE-UIS (Laboratorio de Integración Energética)

Cada uno de estos con necesidades específicas, las cuales se detallan en la siguiente sección:

3.1. Necesidades de los interesados

3.1.1. GISEL. Siendo el grupo de investigación en el que se está desarrollando el proyecto, requiere del diseño de un sistema de monitoreo basado en IoT para la microrred eléctrica experimental. Esto enmarcado, como ya fue mencionado, en la convocatoria 890 de Minciencias (Ministerio de Ciencia, Tecnología e Innovación, 2021). Dicho sistema debe ser diseñado pensando en la implementación futura de una plataforma de aprendizaje para enseñar sobre industria 4.0.

3.1.2. DAUTOM. Siendo la empresa que asesora y colabora con el grupo de investigación en el desarrollo del proyecto, requieren que el diseño planteado sea compatible con la plataforma Ignition de Inductive Automation. Dautom es una empresa santandereana con experien-

cia en hacer despliegues de Internet of Things en el contexto industrial (IIoT) y tienen experiencia haciendo sus desarrollos con este software.

3.1.3. LIE-UIS. Siendo el sitio en donde se está realizando este proyecto y en donde GISEL tiene más proyectos en curso como la microrred en sí misma, requiere que el diseño planteado sea compatible con los controladores de los inversores que se encuentran en el laboratorio (dSPACE). Además de la compatibilidad, se requiere que el desarrollo a hacer no interfiera con los proyectos de posgrado que para la fecha de elaboración de este proyecto se encuentran en ejecución (sección 3.2.6).

3.2. Recursos necesarios para la realización del proyecto

3.2.1. Conocimientos. Es necesario someterse a una curva de aprendizaje para poder plantear el diseño del sistema IoT, para esto, se hace necesario, hasta cierto punto, comprender la microrred eléctrica y su funcionamiento. Analizar el flujo de información entre la microrred y el sistema a diseñar, especificando los elementos necesarios para este proceso. Adicionalmente, se requieren conocimientos en protocolos de comunicación usados en la industria para dispositivos de control y/o sensado como modbus.

3.2.2. Personal. El proyecto requiere personal que recopile las necesidades del sistema (este proceso en sí mismo) y posteriormente, mediante un conocimiento adquirido, presente y ejecute una solución acorde a los requerimientos. Teniendo en cuenta las actividades por realizar, el tiempo de ejecución y demás procesos inherentes a al proyecto, un equipo de trabajo integrado por mínimo 2 y máximo 3 personas es adecuado para la realización.

3.2.3. Dispositivos. La adquisición de dispositivos (hardware) es necesaria en el proyecto. La selección de estos depende de parámetros como los requerimientos impuestos por los interesados del proyecto, la viabilidad económica y temporal y el hecho de que la microrred eléctrica no esté terminada. Siendo este uno de los puntos clave para poder empezar a desarrollar el sistema, la selección del hardware fue de las primeras cosas que se atacó desde el criterio técnico y que se adquirió con el presupuesto de Minciencias posterior a sostener reuniones con GISEL y Dautom con el propósito de definir el hardware adecuado para este caso de uso.

Dentro de las opciones más fuertes que se discutieron se consideraron: el Groov RIO EMU y el Groov EPIC, ambos de la empresa Opto22. El Groov RIO EMU (Opto22, 2022b) De sus siglas en inglés traduce Unidad de Monitoreo de Energía, es un hardware pensado específicamente para la medición de variables energéticas. El EPIC, cómo ya fue definido en 1.8, es un computador industrial o controlador que, por si solo, no tiene la posibilidad de medir variables energéticas, sin embargo, Opto22 desarrolla una serie de módulos que sirven distintos propósitos y que se pueden conectar al EPIC para dotarlo de características adicionales. El módulo GRV-R7-I1VAPM-3 es el mismo Groov RIO EMU, pero modificado para servir como módulo del Groov EPIC, con base en esto, la decisión final fue adquirir un Groov EPIC con un módulo GRV-R7-I1VAPM-3. Esta configuración escogida gana potencia computacional en relación con haber adquirido solamente el Groov RIO EMU; cuenta con la capacidad de hacer mediciones energéticas y adicionalmente otorga la flexibilidad en cuanto a la programación y configuración de funcionamiento que se le puede hacer al EPIC y la naturaleza modular de este mismo (Opto22, 2022a). Este controlador

permite no solamente monitorear las variables de la microrred, también que se pueda ejercer control sobre estas, lo cual es una característica deseada por el grupo de investigación para futuros desarrollos usando el sistema, una tabla comparativa de los criterios más importantes tenidos en cuenta para la selección del hardware se puede ver en la figura 2, en donde se puede apreciar que el criterio que, en efecto, fue el decisivo, fue el hecho de poder controlar o modificar variables de la microrred. Además del módulo de medición energética, se adquirió un módulo para conectar dispositivos seriales; no se usó para conectar las dSPACE al EPIC, pues estas se conectaron a través de un concentrador USB, por lo que este módulo queda disponible en el controlador para conectar más dispositivos con los que se quiera experimentar a posteridad. Una foto del tablero que contiene estos dispositivos se puede ver en la figura 5, dicho tablero fue realizado por la empresa Dautom posterior a las reuniones en las que se definieron los dispositivos a usar; los planos asociados a las conexiones de este se pueden encontrar en los anexos 10.

3.2.4. Software. La adquisición o creación de software es un recurso necesario en el proyecto. Como ya fue presentado en la introducción 1.6, las dSPACEs requieren de simulink para programarse, por lo que una licencia de Matlab es necesaria. Además, se requiere una licencia de Ignition para realizar el despliegue definitivo. De esos dos softwares mencionados, los computadores del LIE cuentan con licencias de Matlab y la licencia de Ignition no es necesaria para desarrollar en este software, sin embargo, como ya fue mencionado, sí es necesaria en el despliegue definitivo porque de manera gratuita se debe reiniciar el software cada dos horas (restricciones de la versión de prueba).

Figura 5
Tablero IIoT hecho por Dautom

3.2.5. Presupuesto. Complementando el punto expuesto en la sección 3.2.3 Las compras que requiere implementar el sistema contemplan: un controlador industrial programable, módulos de monitorización de energía y sus accesorios. Un presupuesto estimado de estos equipos está alrededor de $10.000 USD, dinero obtenido a partir de la convocatoria 890 de Minciencias. El reporte de la compra realizada para el proyecto se presenta en la tabla 3.

Tabla 2

Tabla comparativa, criterios a tener en cuenta para la selección.

Comparación	Groov EPIC PR2 + GRV-R7-I1VAPM-3	Groov RIO EMU
Posibilidad de hacer control	Sí	No
Memoria RAM	2 GB	1 GB
Almacenamiento	22 GB	4 GB
Medición de variables energéticas	Sí	Sí
Protocolos de comunicaciones	Sí	Sí
Puertos USB y Ethernet	Sí	Sí
Modularidad	Sí	No

Tabla 3

Resumen de la compra relacionada al computador industrial/controlador

Computador Industrial Microrred	Precio
Envolvente: Armario compacto IP65 de 400x300x210 (mm). Comm: Incluye Anybus Wireless on RJ45 y PoE embebidos, patchpanel Panduit para dos unidades de jacks y patchcord RJ45. Fuente: Allen Bradley entrada 120/240VAC monofásico, voltaje salida 24VDC 10A. Controlador: Groov Epic PR2, con convertidor 22-50 VDC, con módulos sobre chassis de 8 ranuras: GRV-IVAPM-3, GRV-CSERI-4, GRV-OACI-12, GRV-ODCI-12.	$47.690.226

3.2.6. Descripción del escenario de implementación. En la figura 6 se presenta una imagen del Laboratorio de Integración Energética que se encuentra en la sede Parque Tecnológico Guatiguará de la Universidad Industrial de Santander. En este laboratorio se encuentran los elementos que formarán la microrred eléctrica (Figura 7), sobre la cual será planteado el diseño del sistema IoT. Como ya fue descrito, se sostuvieron diversas reuniones (figura 8) con el equipo responsable de la construcción de la microrred eléctrica con el fin de comprender el espacio de trabajo, sus necesidades y limitaciones: La microrred eléctrica en construcción se presenta conceptualmente en la figura 9, en este esquema los rectángulos son lineas de distribución emuladas; los círculos son transformadores eléctricos de aislamiento, su propósito es separar etapas del sistema y las cargas locales y globales son de naturaleza resistiva. Dentro de sus especificaciones de diseño, contempla una capacidad de 12 kVA, estará integrada por fuentes AC y DC que permitirán la simulación de la red de distribución. Contará con inversores trifásicos controlados mediante tarjetas de control dSPACE DS1104 para la emulación de diversos generadores distribuidos, entre otros elementos necesarios para pruebas e investigación. En paralelo al proyecto de la construcción de la microrred eléctrica se están desarrollando cinco proyectos de posgrado (entre maestrías y doctorados, como se puede evidenciar en la tabla 4).

Para la realización de estos proyectos no es necesario que la micrrored esté montada al 100 %. Ejemplificado en el proyecto de doctorado, este se puede llevar a cabo usando una sola rama, gracias a que la red posee tres puntos de generación que están conectados la red principal mediante una fuente Chroma (figura 9).

Tabla 4

Tabla con los proyectos en curso en estos momentos que están relacionados a la microrred

Título	Programa	Autor	Directores
Control of Three Phase Inverters Under Voltage Sags	Tesis Doctoral	MSc. David Javier Rincón Adarme.	Director: PhD. María Mantilla Co-Directores: PhD. Juan Rey PhD. Miguel Garnica
Control de Inversores fotovoltaicos conectados a la red ante hundimientos de tensión considerando distorsión armónica	Maestría en Ingeniería Eléctrica	Ing. Wilmar Alejandro Sotelo Rueda.	Director: MSc. David Rincón Co-Directores: PhD. Maria Mantilla PhD. Juan Rey
Diseño e implementación de un sistema IIoT para la monitorización y control de una microrred eléctrica experimental	Maestría en Ingeniería de Telecomunicaciones	Ing. Jonathan Stiven Gomez Zuluaga	Director: PhD. Juan Rey Co-Director: PhD. Javier Solano
Diseño de una estrategia de mitigación de hundimientos de tensión en redes de distribución mediante sistemas fotovoltaicos	Maestría en Ingeniería Eléctrica	Ing. Astrid Clarissa Esparza Aponte.	Director: PhD. María Mantilla Co-Directores: D. Sc. Mario Arrieta PhD. Jairo Blanco
Control flexible de corriente para sistemas fotovoltaicos con capacidad de operación ante hundimientos de tensión	Maestría en Ingeniería Eléctrica	Ing. Ingrid Johanna Moreno Celis.	Director: PhD. Maria Mantilla Co-Directores: PhD. Juan Rey MSc. David Rincón

Figura 6

Laboratorio de Integración Energética (LIE)

Figura 7

Algunos elementos presentes en el lab, a la izquierda unos inversores y a la derecha una fuente AC marca Chroma

Figura 8

Una de las reuniones sostenidas con el profesor Juan Manuel Rey, director de este proyecto.

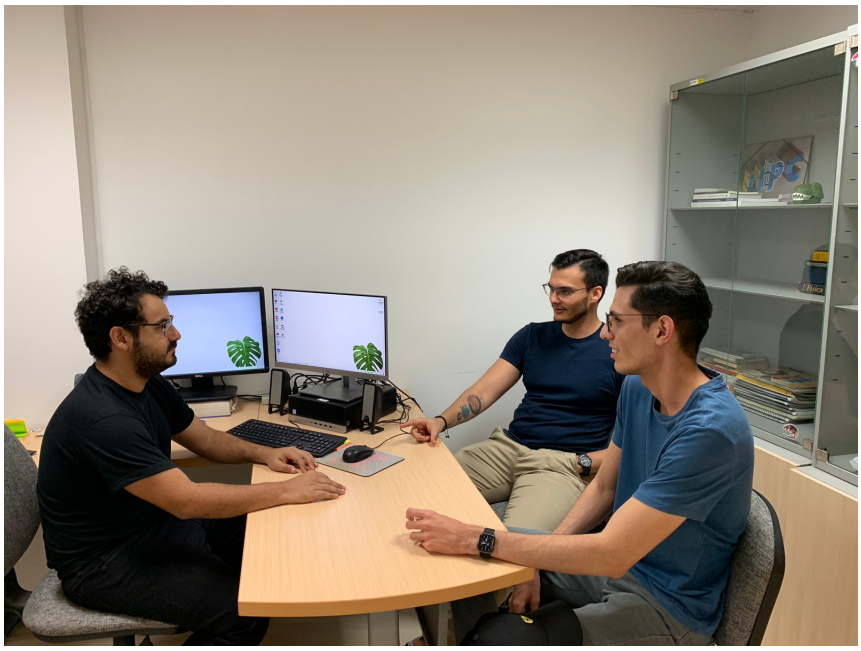


Figura 9

Esquema de la microrred eléctrica experimental en construcción

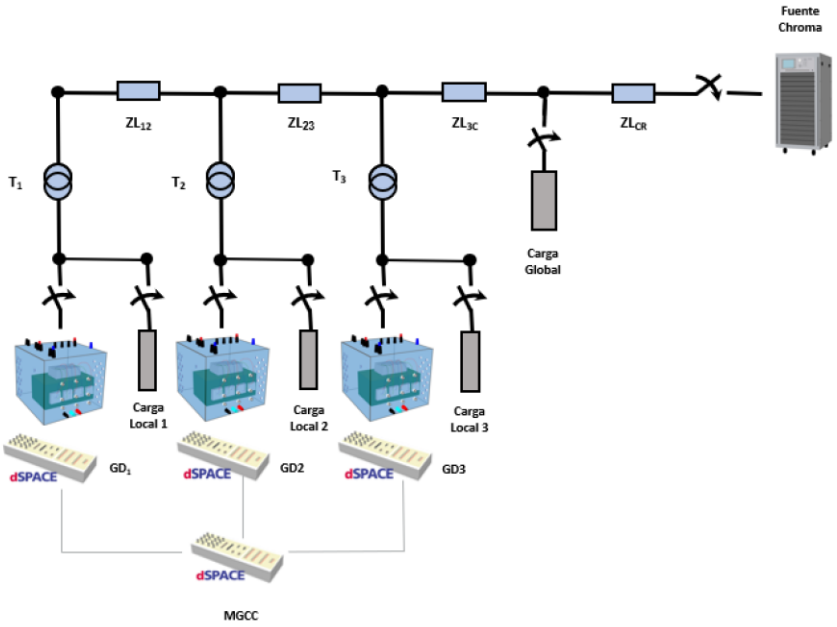
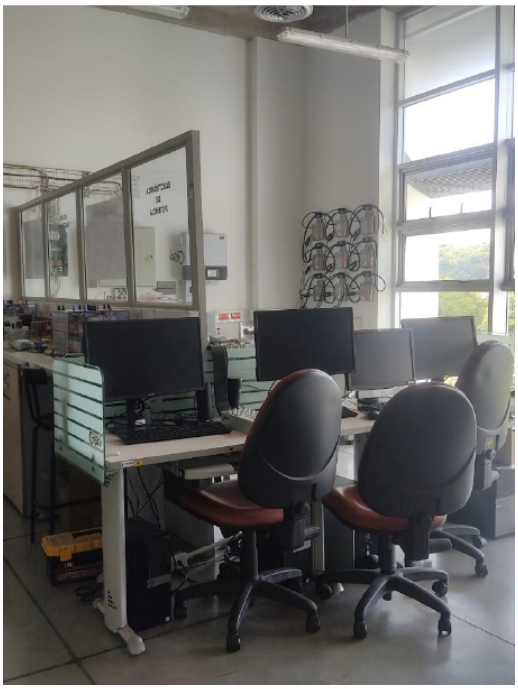


Figura 10

Computadores disponibles en el laboratorio que cuentan con tarjetas dSPACE ds1104



Dentro del Laboratorio de Integración Energética se cuenta con tres computadores (figura 10) que poseen instaladas tarjetas dSPACE DS1104. Estas tarjetas fueron adquiridas por el equipo de la microrred eléctrica y su principal propósito es controlar los inversores de la misma. En la figura 11 se presenta una torre CPU de una de los computadores que muestra adhesivos referenciados con el número de identificación de la tarjeta dSPACE DS1104 en su interior. Este número es de vital importancia en el momento de usar la tarjeta debido a que se requiere conectar un dispositivo usb con la licencia de la tarjeta (llave) al interior de la torre que permita su funcionamiento.

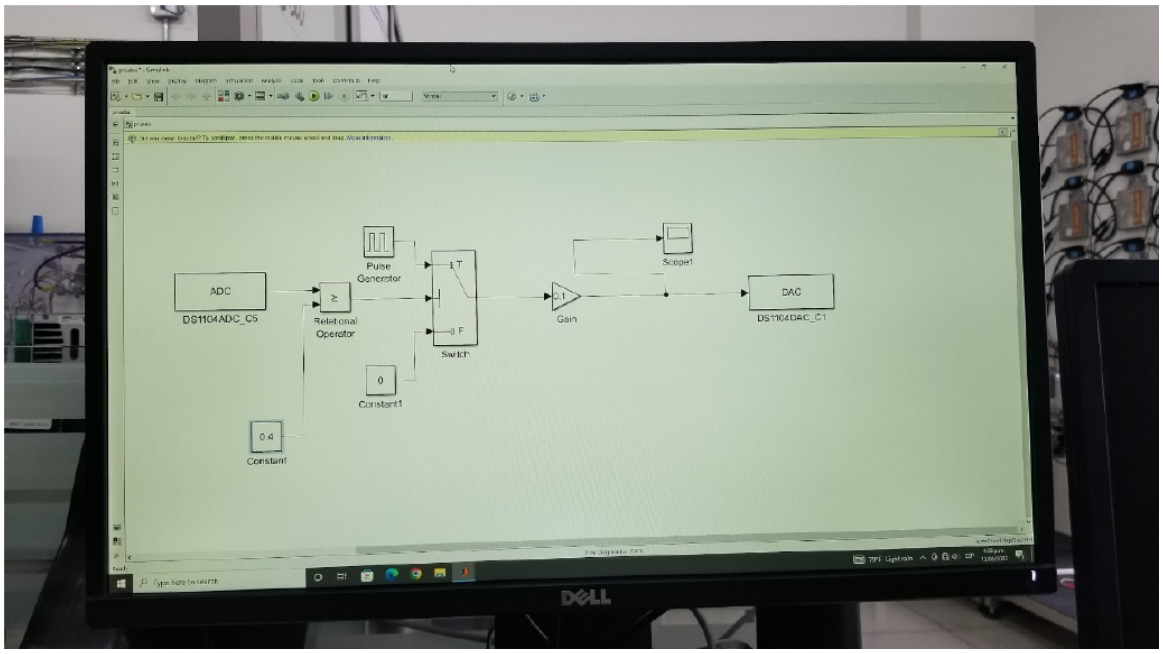
Las tarjetas de control dSPACE DS1104 son una tecnología que se integra en la tarjeta madre de un computador y se conecta con los elementos a controlar a través de un módulo de conexiones como el ya previamente presentado en la figura 3. La rutina de adquisición y procesamiento de datos

Figura 11

Torre CPU con identificación de tarjeta dSPACE ds1104

Figura 12

Diagrama de bloques en Simulink utilizado para programar la dSPACE



realizada por la dSPACE se carga a través del software ControlDesk y la descripción funcional de esta rutina se realiza en Simulink de MathWorks como se demuestra en la figura 12.

Con base en los requerimientos recolectados, el sistema IoT a diseñar tendrá un punto de monitoreo de la red. Este punto tiene gran importancia, puesto que, al utilizarse un controlador industrial programable Opto 22 GRV-EPIC-PR2 en conjunto con un módulo para realizar mediciones de variables eléctricas GRV-IVA-PM-3, se pueden realizar mediciones de voltajes y corrientes trifásicas y a partir de estos valores, computar otros datos de importancia como potencia activa y reactiva, factor de potencia y frecuencia. Por otra parte, el sistema a diseñar tendrá que contemplar una forma de comunicar entre sí las tarjetas de control dSPACE DS1104 para permitir un intercambio de información para su uso en el proceso de monitoreo y control de la microrred. Todos estos datos medidos deberán entregarse en un cliente de la plataforma Ignition 8.1 donde se deberá elaborar una interfaz que permita visualizar valores de la microrred y modificarlos. Los inversores requieren un proceso de control para su correcto funcionamiento, pero este será tenido en cuenta en el diseño de este sistema IoT, debido a su interés en otros procesos investigativos que se encuentran en ejecución (como ya fue presentado en la tabla 4).

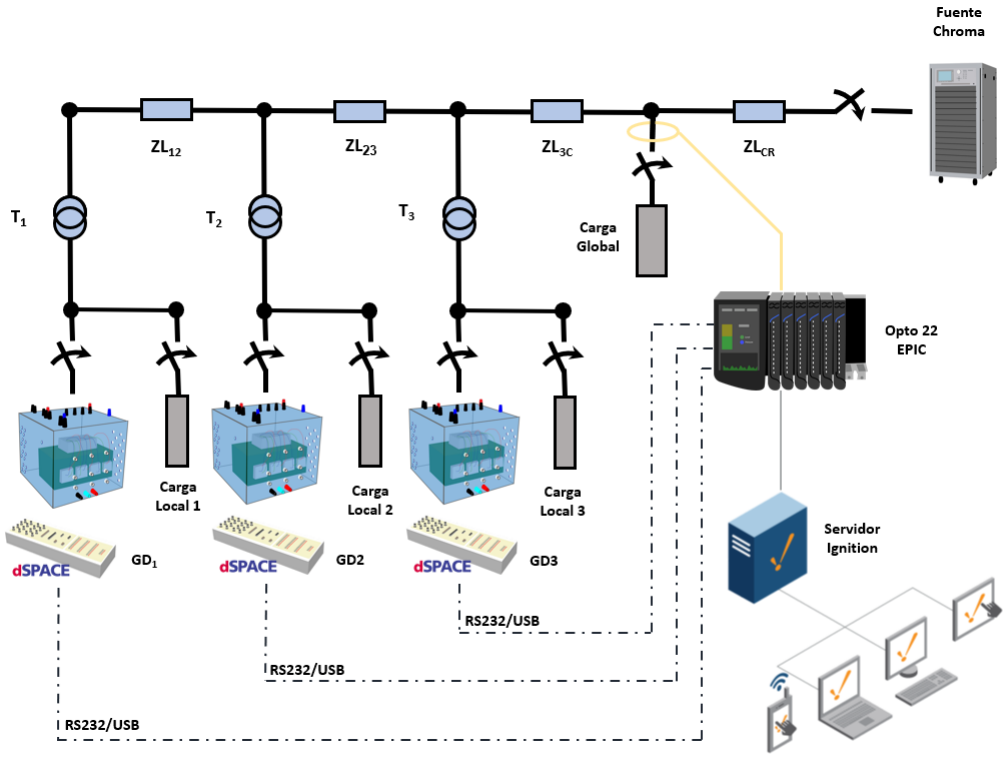
4. Diseño del Sistema IoT

4.1. Esquema General

Un esquema global de la microrred, ya contemplando el sistema IoT, se muestra en la figura 13.

Figura 13

Esquema general de la microrred.

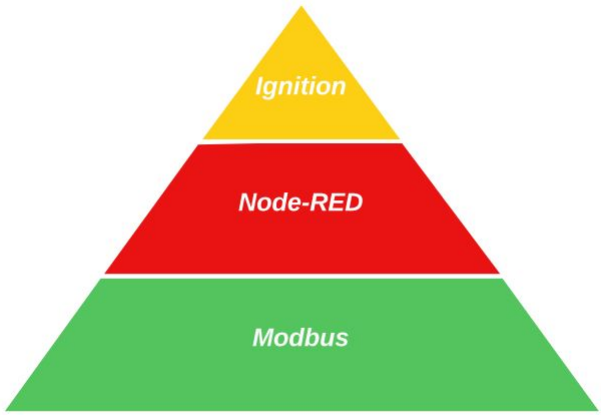


El sistema IoT a trabajar de la microrred se compone de tres elementos principales: las tarjetas de control **dSPACE**, el controlador programable industrial **Opto 22 EPIC** y el servidor **Ignition**.

El esquema de comunicaciones, control y visualización propuesto se muestra en la figura

14.

Figura 14
Esquema de comunicaciones de la microrred.



El sistema está soportado por el protocolo de comunicaciones Modbus, con Node-RED actuando como maestro principal, enviando y recibiendo información del software Ignition.

4.2. Tarjetas de control dSPACE

El Laboratorio de Integración Energética cuenta con 3 tarjetas dSPACE, encargadas de controlar los futuros inversores que conforman la microrred. Los modulos de conexión a estas tarjetas se muestran en la figura 15.

Cada una de estas dSPACE cuenta con una identificación distinta y su respectiva licencia, que debe ser insertada en el computador para realizar las simulaciones y validaciones, la llave se muestra en la figura 16. Estas licencias, su serial y nombre se muestran en la tabla 5.

Es importante resaltar nuevamente que lo que se muestra es solo una caja de conexiones (Connector Pannel), debido a que realmente la dSPACE está conectada al puerto PCI o PCIe de

Figura 15
Los 3 módulos de conexión de las dSPACE de la microrred.

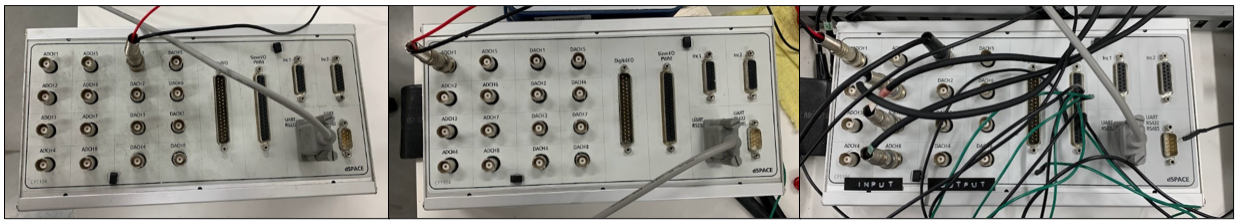


Figura 16
Licencia de la dSPACE

Tabla 5
Licencias asociadas a la dSPACE

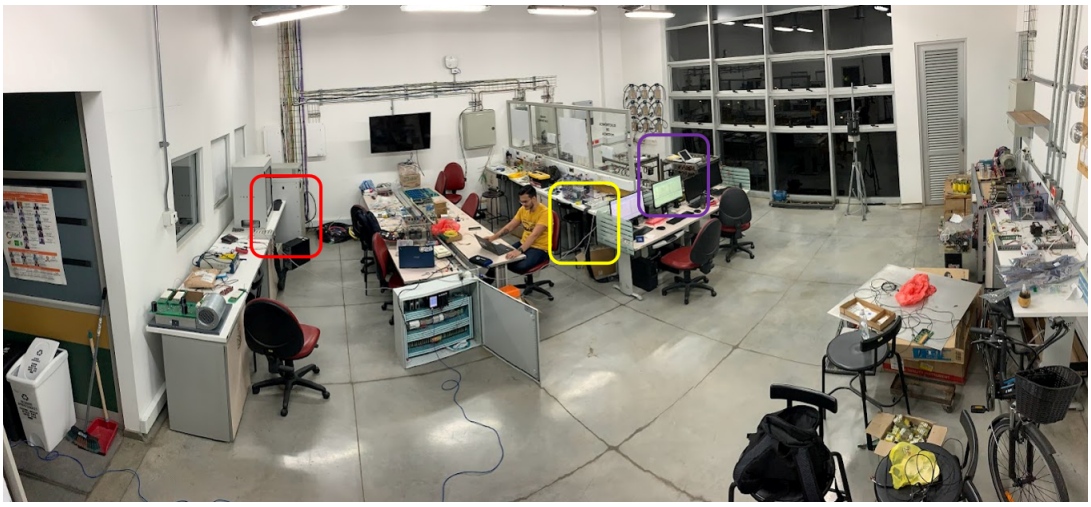
Nombre dSPACE	Licencia	Modelo
Chroma	32770	DS1104
Izquierda	32771	DS1104
Central	32772	DS1104

los computadores. De ahí, es donde nos permite tener los ADC, DAC, PWM, puertos seriales, etc. para poder realizar los experimentos de validación.

La ubicación espacial de las tarjetas en el laboratorio se muestran en la figura 17. En color rojo, la dSPACE *Chroma*, se encuentra a un costado de la fuente programable con el mismo nombre, en color Amarillo la dSPACE *Izquierda* y en color morado la dSPACE *Central*, estas últimas llamadas así debido a su posición en el laboratorio.

La tarjeta cuenta con 8 convertidores analógicos - digital (ADC), usados para lectura de

Figura 17
Ubicación de las dSPACE



variables eléctricas. Además, cuenta con 8 canales digital - análogico (DAC) para escribir señales de salida. En la tabla 6 se muestran las especificaciones de los canales ADC. En la tabla 7 se exponen las especificaciones de los canales DAC. (dSPACE GmbH, 2016)

Tabla 6
Especificaciones canales ADC

ADCH1 ... ADCH4	ADCH5 ... ADCH8
Multiplexed 4:1	Single Channel
16-bit resolution	12-bit resolution
±10 V input voltage range	
± 5 mV offset error	
± 0.25 % gain error	± 0.5 % gain error
>80 dB (at 10 kHz) signal-to-noise ratio	>70 dB signal-to-noise ratio

Ya conociendo las tarjetas a trabajar, el siguiente paso era buscar un protocolo de comunicación compatible, que permitiera usar los puertos seriales de las tarjetas para la comunicación, y el escogido fue Modbus RTU.

Tabla 7
Especificaciones canales DAC

DACH1 … DACH8
8 parallel DAC channels
16-bit resolution
±10 V output voltage range
± 1 mV offset error, 10 V/K offset drift
± 0.1 % gain error, 25 ppm/K gain drift
>80 dB (at 10 kHz) signal-to-noise ratio
± Transparent and latched mode

4.3. Modbus RTU

Se usa este protocolo de comunicaciones gracias a su capacidad de transmisión de información a través de líneas seriales. En el proyecto, los dos intereses principales eran: leer los datos que estaban leyendo los ADC y segundo, poder escribir en los DAC para enviar señales de control a los experimentos que se realizarían. Para el primer interés se usó la **Función 3** y para el segundo, la **Función 6**.

4.3.1. Función 3 - Leer registros de retención. La función permite leer varios registros al mismo tiempo, estos son los canales ADC que puede leer la dSPACE, en total son **8 canales** con rango de entrada de ± 10 [V]. El mensaje de request de la función 3 es la siguiente (Figura 18):

Figura 18
Función 3

- **Dirección (1 byte HEX)**: Corresponde a la dirección del esclavo, en este caso se le asigna las direcciones 10, 20, 30 a cada dSPACE (Figura 8).

Tabla 8
ID Modbus de las dSPACE

Nombre dSPACE	Slave Address
Izquierda	10
Central	20
Chroma	30

- **Función (1 byte HEX)**: Es el código de la función, en este caso la función 3 (Leer registros de retención).

- **Dirección 1.er Registro (2 bytes HEX)**: Se especifica desde cuál registro se quiere empezar la lectura, en este caso, deseamos leer desde el 1.er registro.

- **Cantidad de Registros (2 bytes HEX)**: Es el número de registros que se desea leer, en este caso, queremos leer 10 registros, 8 correspondientes a los canales ADC y otros dos extra que se pueden usar para otra aplicación.

- **CRC (2 bytes HEX)**: Es la verificación de redundancia cíclica, para evitar errores en la transmisión de los datos, estos corresponden a la codificación de los 6 bytes anteriores.

Así se completan los 8 bytes Hexadecimales de la función mensaje Modbus que se envía para solicitarle al dispositivo la transmisión de los datos almacenados en los registros.

La respuesta dada por el sistema después de recibir el mensaje es de la siguiente forma (Figura 19):

Figura 19
Función 3 Respuesta

- **Dirección (1 byte HEX)**: Corresponde a la dirección del esclavo (Figura 8).
- **Función (1 byte HEX)**: Es el código de la función, en este caso la función 3 (Leer registros de retención).
- **Número Bytes (1 byte HEX)**: Es el número de bytes que regresa la respuesta Modbus. En este caso muestra el número 14 hexadecimal, que en decimal sería 20, puesto que se leen 10 registros, y cada uno de estos tiene 2 bytes hexadecimales.
- **Contenido de los Registros (2 bytes HEX cada uno)**: Son los datos almacenados en los registros del REG1 al REG10, es la trama más larga, compuesta por 20 bytes.
- **CRC (2 bytes HEX)**: Es la verificación de redundancia cíclica, para evitar errores en la transmisión de los datos, estos corresponden a la codificación de los 23 bytes anteriores.

Del REG1 al REG8 corresponden a los 8 ADC que tiene la dSPACE en orden, los registros REG9 y REG10 ahora mismo se encuentran vacíos, se pueden llenar con alguna información extra que a futuro se desee transmitir.

En el ejemplo se hace la lectura de los 10 registros de la dSPACE **Izquierda**, que posee la dirección 10.

4.3.2. Función 6 - Forzar registro individual. Esta función permite escribir contenido en un registro. Estos registros corresponden a los canales DAC que tiene la dSPACE, en total son **8 canales** con rango de salida de ± 10 [V]. El mensaje de request de la función 6 es la siguiente (Figura 20):

Figura 20
Función 6

- **Dirección (1 byte HEX)**: Corresponde a la dirección del esclavo (Figura 8).
- **Función (1 byte HEX)**: Es el código de la función, en este caso la función 6 (Forzar registro individual).
- **Dirección registro (2 bytes HEX)**: Es la dirección del registro a la cual voy a escribir el valor deseado. Las direcciones de los DAC configurados se muestran en la siguiente tabla 9:
- **Valor a escribir (2 bytes HEX)**: Es el valor númerico que deseo configurar en el DAC. El valor máximo a escribir es FFFF, es decir, 65.535.
- **CRC (2 bytes HEX)**: Es la verificación de redundancia cíclica, para evitar errores en la transmisión de los datos, estos corresponden a la codificación de los 6 bytes anteriores.

Al ser Modbus un protocolo que siempre espera una respuesta del esclavo que escribió, este debe retornar un mensaje. Al tratarse de la función 6, la palabra a retornar es una copia exacta del

Tabla 9
Direcciones registros DAC

DAC	Dirección
DACH1	101
DACH2	201
DACH3	301
DACH4	401
DACH5	501
DACH6	601
DACH7	701
DACH8	801

mensaje que obtuvo inicialmente, que debe ser devuelta una vez se haya escrito el contenido del registro.

En el ejemplo, lo que se espera es que se modifique el DAC encontrado en la dSPACE **izquierda (10)**, con dirección 01F5 hexadecimal **(501 decimal)** correspondiente al **DACH5**, con valor 0BB8 hexadecimal **(3000 decimal)**. Esto hará que el DAC5 de la dSPACE izquierda tome un valor de 3 [V].

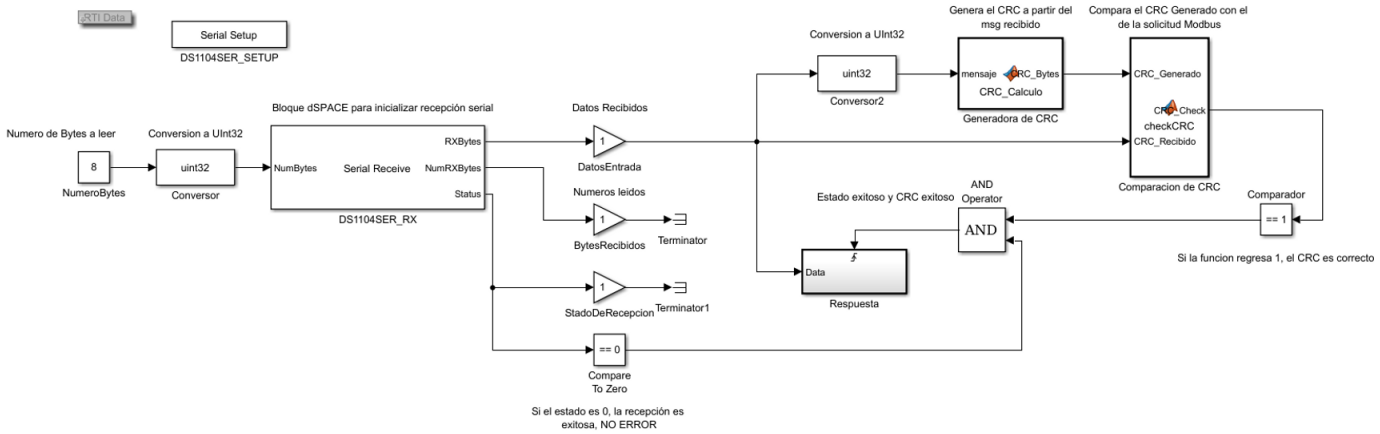
4.4. MATLAB - Simulink

Los bloques de programación usados para la configuración de la interfaz serial, el ecosistema Modbus, lectura ADC y escritura DAC se muestran en la figura 21.

A nivel general, este modelo de Simulink toma el mensaje del Modbus Master (Node-Red) y se encarga de armar el mensaje de respuesta con sus acciones de control. (Starynets, 2016)

- **DS1104SER_SETUP**: Este bloque inicializa la interfaz serial de la tarjeta. Se dan los parámetros iniciales UART (Figura 22).

Figura 21
Bloques de Simulink



Se configura el serial RS232 a 9600 baudios (bits/seg) con 8 bits de datos (por lo que se transmite en bytes), y con 1 bit de stop. Es importante resaltar el último parámetro (Copy data), este configura el envío de la información una vez haya llenado 8 bytes de información.

- **DS1104SER_RX**: El bloque recibe la trama Modbus del maestro. Se le especifica que recibirá un mensaje de 8 bytes, y se convierte a uint32 por la configuración del bloque. (Figura 23).

  El bloque regresa 3 valores, los datos que recibe *(RXBytes)*, el número de bytes recibidos *(NumRXBytes)* y el estado *(Status)*. Este último determina si la recepción fue exitosa, marcando como exitosa (0) o fallida (1). El número de bytes recibidos no es un valor que interese, puesto que la trama modbus tendrá siempre 8 bytes.

- **Función Generadora de CRC**: De los 8 bytes recibidos, los primeros 6 bytes son de información, y los dos últimos son de CRC recibidos. La función realiza el CRC de los primeros

Figura 22
Configuración Serial dSPACE

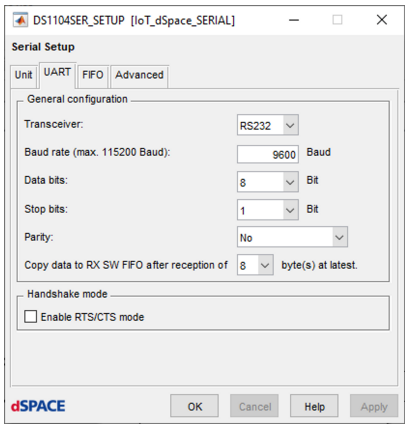
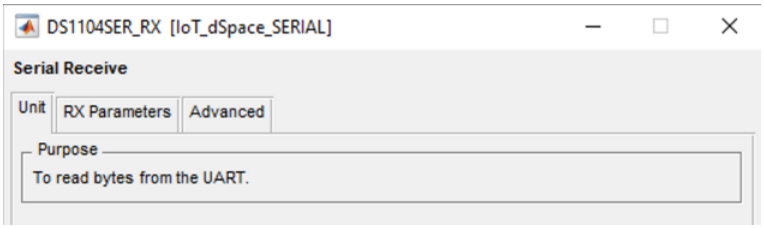


Figura 23
Recepción serial dSPACE



6 bytes, todo esto para evitar errores en recepción. El código se encuentra en el anexo 1 (Starynets, 2016). Su funcionamiento se ejemplifica en la figura 24.

La función retorna los bytes de CRC generados, para ser comparados con el CRC del mensaje modbus en la siguiente función.

- **Función Comparadora de CRC**: La función se encarga de comparar los bytes de CRC que llegan de la trama Modbus con los que generó la función anterior. Para una transmisión sin errores, estos códigos deben ser exactamente iguales. El código se encuentra en el anexo 2. Su funcionamiento se ejemplifica en la figura 25.

Figura 24
Explicación Función Generadora CRC

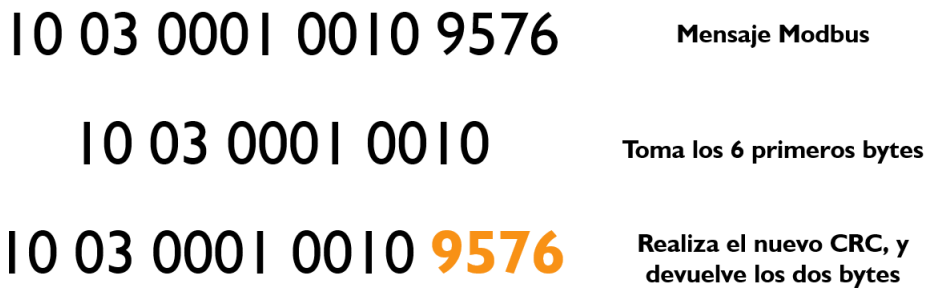
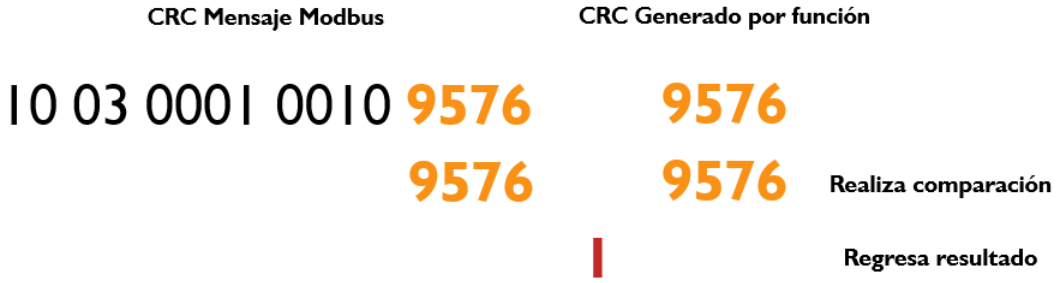


Figura 25
Explicación Función Comparadora CRC



La función regresa un valor booleano (0 o 1) según la comparación, que entra al siguiente bloque.

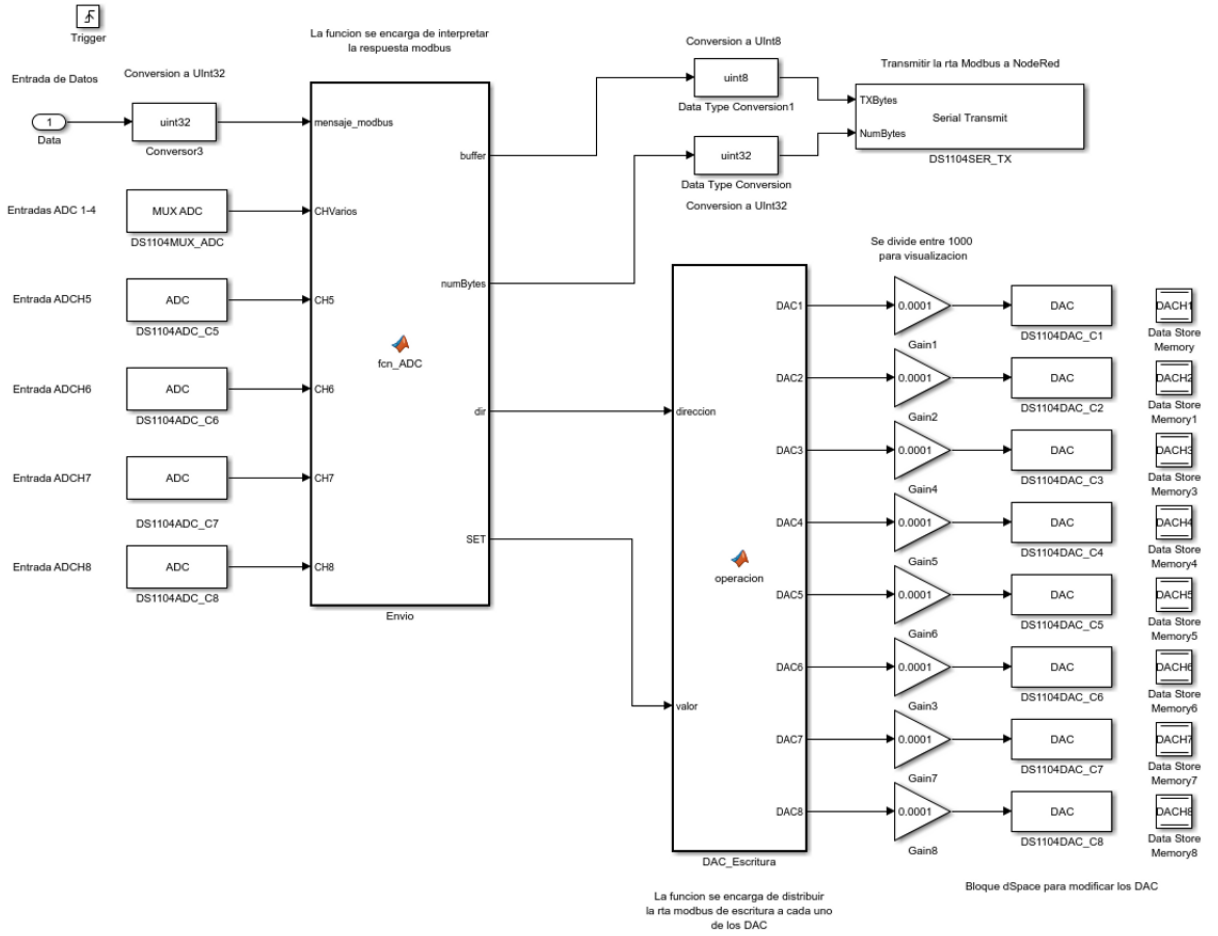
- **AND Operator**: El bloque valida que el estado de recepción sea exitoso y que la comparación del CRC sea correcta. Este bloque funciona como el trigger de la función respuesta, interpretando el mensaje modbus obtenido.

4.4.1. Triggered Subsystem - Interpretación. Con las validaciones concluidas, se parte al nuevo sistema de bloques, activado de la secuencia anterior. (Figura 26).

- **Data**: Son los datos recibidos del Modbus master, se convierten a uint32.

Figura 26

Triggered Subsystem - Interpretación



- **DS1104MUX_ADC**: Los canales ADCH1 ... ADCH4 son multiplexados 4:1 (Tabla 6), por lo que este bloque contiene la información de los 4 canales. (Figura 27)

  El bloque organiza los cuatro canales en un vector de 4 posiciones.

- **DS1104ADC_CX**: Los canales ADCH5 ... ADCH8 son de canal sencillo (Tabla 6). Cada bloque debe ser configurado para que lea la información del canal seleccionado. (Figura 28)

  En la figura 28 se está configurando el canal 5, correspondiente al bloque DS1104ADC_C5.

Figura 27
Mux ADC

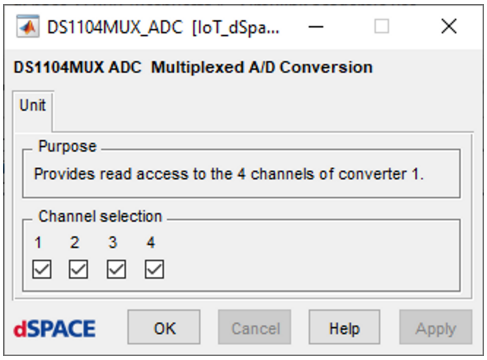
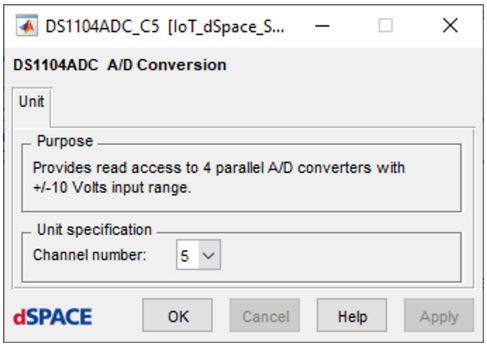


Figura 28
Canales ADC Sencillo

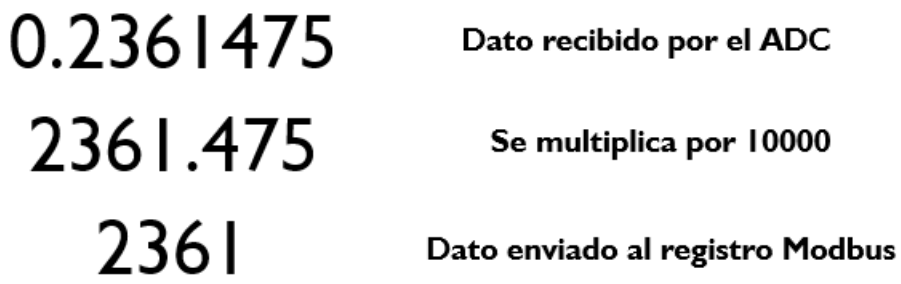


- **Función Envío**: Es la función cerebro de la comunicación, se encarga de obtener los datos de los ADC, asignar la dirección del esclavo, realizar el CRC respuesta, crear la trama completa Modbus de respuesta y de enviarle la dirección y valor a los bloques DAC. El código se encuentra en el anexo 3.

  Es importante resaltar que la tarjeta de control envía valores entre [-1,1], es decir, en sus ADCs un valor de 1 [V] corresponde a 0.1 y un valor de 10 [V] corresponde a 1. Debido a esto, se realiza una función auxiliar llamada preparar_envio.m, toma el valor de los ADC y lo multiplica por 10000, para así obtener tres decimales y enviar en el registro

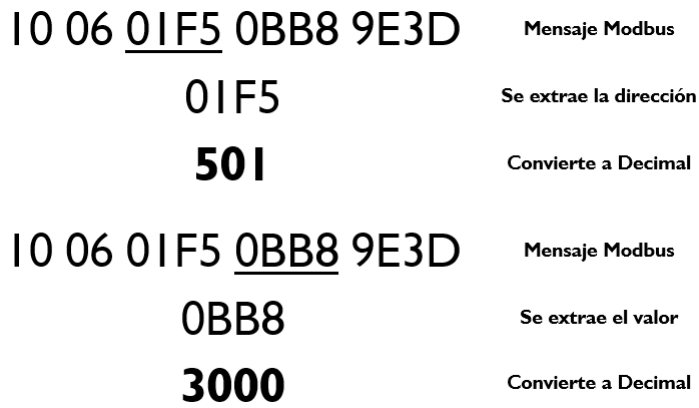
Modbus un número entero. Se explica con un ejemplo en la figura 29. El código se encuentra en el anexo 4.

Figura 29
Función Preparar Envío



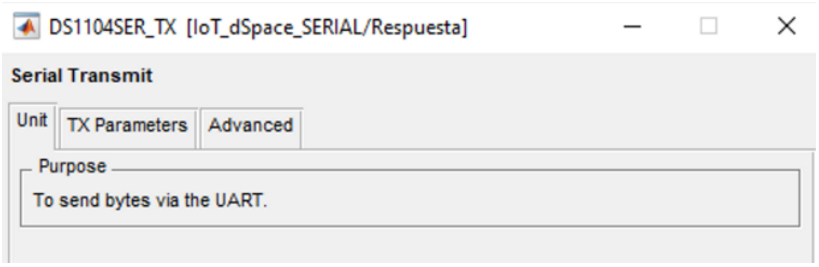
Cuando la solicitud Modbus corresponde a la función 6, la dirección y el valor se encuentra en 2 bytes hexadecimales, deben ser convertidos en decimal para enviarlos a la dSPACE. Debido a esto, se crea otra función auxiliar llamada preparar_recepcion.m . Esta se encarga de transformar estos 2 bytes a un valor decimal. Se explica con un ejemplo en la figura 30. El código se encuentra en el anexo 5.

Figura 30
Función Preparar Recepción



- **DS1104SER_TX**: El bloque se encarga de transmitir el mensaje de respuesta Modbus de vuelta al maestro. Es un mensaje de 25 bytes. Su descripción se muestra en la figura 31.

Figura 31
Transmisión serial dSPACE



  - Envío Función 3: El mensaje llena todos los 25 bytes, compuestos de la siguiente forma: 1 byte de dirección, 1 byte de función, 1 byte de número de bytes, 20 bytes de información de los registros (Lectura ADC)y 2 bytes de CRC.
  - Envío Función 6: El mensaje llena solo los primeros 8 bytes de la respuesta de 25 bytes. Se compone de: 1 byte de dirección, 1 byte de función, 2 bytes de dirección de registro, 2 bytes de valor a escribir y 2 bytes de CRC:

- **Función DAC Escritura**: La función se encarga de relacionar las direcciones y el valor obtenido desde el mensaje Modbus hacia los DAC de la tarjeta. Se explica con un ejemplo en la figura 32. El código se encuentra en el anexo 6.

  El código hace uso de variables globales para evitar que se sobreescriban en cada iteración de entrada de mensajes Modbus. Simulink obliga a agregar un bloque Data Store Memory en el diagrama, para así asociar la variable en el entorno gráfico. Su configuración se muestra en la figura 33. Allí se configura la variable DACH1.

Figura 32
Función Escritura DAC

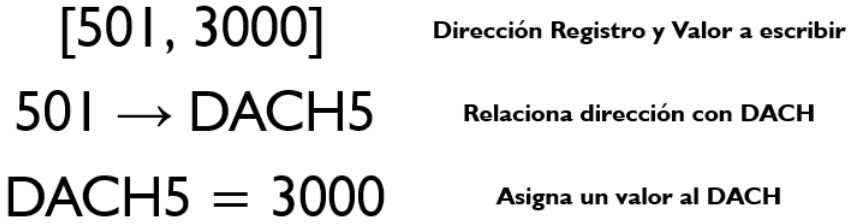
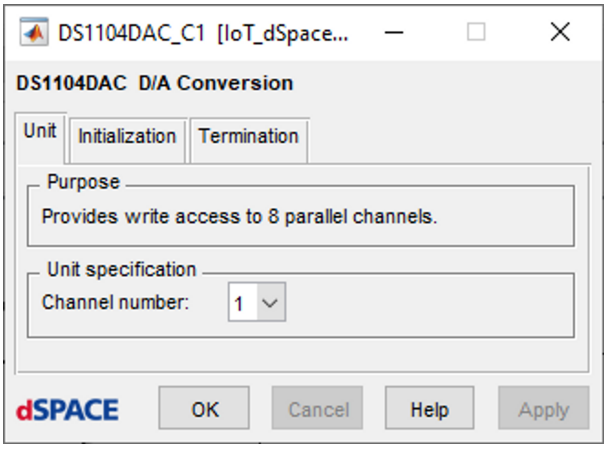


Figura 33
Bloque Data Store Memory

- **DS1104DAC_CX**: Es el bloque usado para configurar las salidas de los DAC de la tarjeta. Cada bloque debe ser configurado en el canal que desea ser escrito. (Figura 34). **El bloque de ganancia inmediatamente anterior divide entre 10000, debido a que la dSPACE solo admite valores entre [-1,1] y se trabaja con exactitud de 3 decimales.** En la figura 34 se configura el canal 1.

Figura 34
Canales DAC



4.5. groov Manage - Configuración groov EPIC

Es el entorno de configuración e información del controlador programable industrial **Opto 22 EPIC**. A partir de ahí, se obtiene el acceso a plataformas como **Node-RED** e **Ignition 8.1**, corriendo directamente desde el dispositivo. El menú principal se observa en la figura 35.

Figura 35
Menú del Opto 22 EPIC

- **Acceso**: Una vez se enciende el Opto 22 EPIC o se accede a través de un dispositivo externo,

se piden las credenciales de acceso. El usuario y la contraseña se encuentra en un sticker dentro del tablero por temas de seguridad. (Figura 36) (Opto22, 2021)

Figura 36
Inicio Opto 22 EPIC

Figura 37
Configuración de Red Opto 22 EPIC

- **Configuración de Red - Network**: El EPIC se conecta mediante ethernet a la red local. Viene configurado automáticamente con la dirección DNS: **opto-05-21-5f**. Su dirección IP es estática, adquiriendo **10.1.16.157**. La información se expone en la figura 37.

- **Dispositivos Seriales**: El dispositivo permite conectar adaptadores USB a seriales, pero solo modelos específicos (Opto22, 2022c), esos son los mostrados en la Figura 38.

El adquirido fue el **GM-FTD1-A12**, al ser compatible con RS232. Se observa en la figura 39. Está conectado a un hub USB, de la marca Anker.

Se conectan 3 adaptadores iguales, conectados a cada dSPACE con una dirección única (Tabla 10). Con la conexión exitosa, el EPIC muestra los dispositivos encontrados (Figura 40).

Tabla 10
Direcciones dSPACE conectadas al EPIC

dSPACE	Dirección serial EPIC
Central	/dev/ttySer1.1
Izquierda	/dev/ttySer1.2
Chroma	/dev/ttySer1.3

Figura 38
Dispositivos seriales Opto 22 EPIC

Serial communication

The following USB-to-serial adapters have been tested and proven to be compatible with *groov* EPIC:

Device Manufacturer	Model
B&B Electronics	USOPTL4 (isolated RS-485) USPTL4 (non-isolated RS-485) USO9ML2 (isolated RS-232)
Gearmo	GM-482422 (non-isolated RS-485/RS-422) GM-FTD1-A12 (non-isolated RS-232) SERIAL-B (non-isolated RS-232)

Figura 39
Adaptador Serial - USB

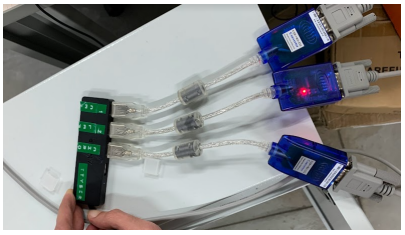
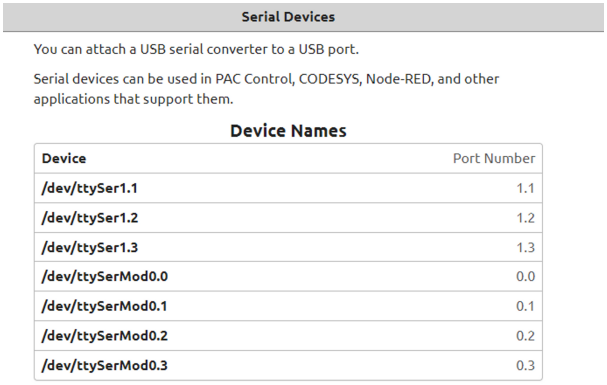
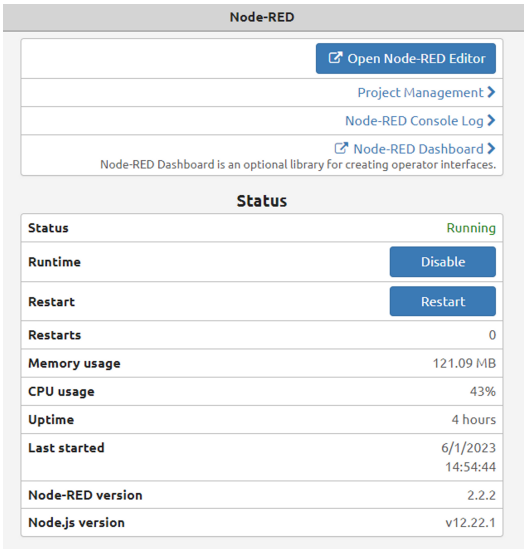


Figura 40
Dispositivos Seriales conectados



- **Node-RED Opto 22 EPIC**: El entorno de programación **Node-RED** se ejecuta directamente desde el dispositivo. Su configuración se encuentra en la sección 4.6. En el EPIC se observa su información general en la figura 41.

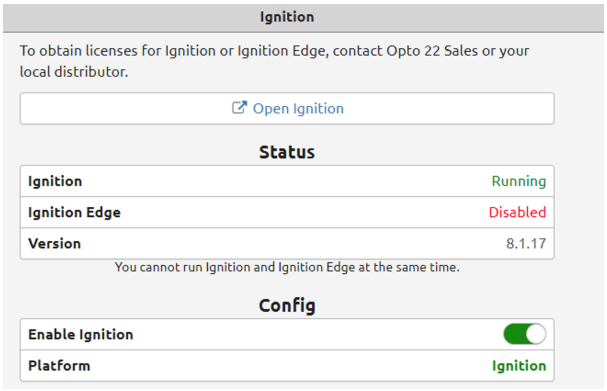
Figura 41
Node-RED en el EPIC



- **Ignition 8.1 Opto 22 EPIC**: El software de visualización y control **Ignition 8.1** es controla-

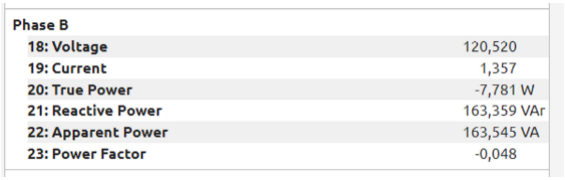
do desde el EPIC. Su configuración se encuentra en la sección 4.7. La información general se muestra en la figura 42.

Figura 42
Ignition 8.1 en el EPIC



- **Módulo de Energía GRV-IVAPM-3**: Es el módulo que permite visualizar variables eléctricas. Está conectado a la carga global de la microrred (Figura 13) y en el EPIC en el rack # 2 (Figura 4). Se validó su funcionamiento conectándola directamente a la red eléctrica y enlazándola a la medición de la fase B del módulo (Figura 43).

Figura 43
Mediciones del módulo de energía



4.6. Node-RED

Es el software encargado de manejar las direcciones de comunicación, crear las tramas Modbus y de enviar los datos a la plataforma Ignition. Permite observar los datos en tiempo real, crear distintos flujos de trabajo y ejercer control de las variables usando el lenguaje JavaScript.

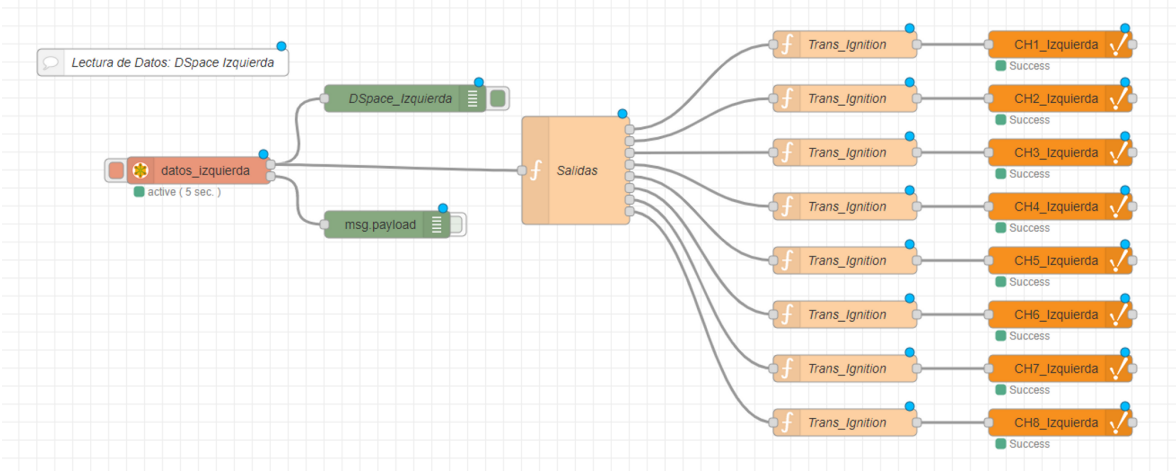
4.6.1. Configuración conexión dSPACE - EPIC. También llamada *Server*, aquí se configura la dirección física de la conexión entre ambos dispositivos. La configuración de la dSPACE *izquierda* Se muestra en la figura 44.

Figura 44
Servidores Modbus

Se configura como *Serial Expert*, con puerto serial según la tabla 10, con tasa de 9600 baud, 8 bits de datos al tratarse de 1 byte, 1 bit de stop, sin paridad y el delay de conexión se especifica a 100 [ms]. La configuración de las demás dSPACE es similar, se deben cambiar los parámetros **Name** y **Serial Port**.

4.6.2. Lectura de datos dSPACE con Node-RED. El flujo de trabajo para la lectura de los ADC se muestra en la figura 45.

Figura 45
Flujo de trabajo para lectura de datos



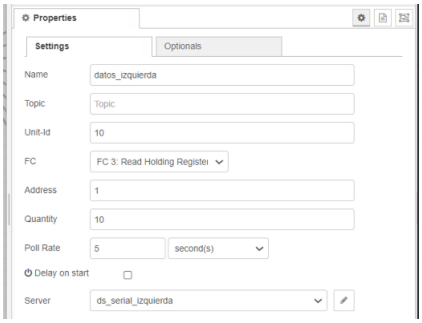
- **Modbus Read**: (Figura 46) Es el nodo encargado de configurar los parámetros Modbus del dispositivo que se va a leer.

Figura 46
Bloque Modbus Read

El bloque **datos_izquierda** envía el mensaje Modbus a la dSPACE izquierda, su configuración se muestra en la figura 47.

1. **Name**: Permite asignarle un nombre al bloque, en este caso, *datos_izquierda*.
2. **Unit_ID**: Corresponde a la dirección del esclavo (Tabla 8). En este caso es la dirección 10, al tratarse de la dSPACE izquierda.

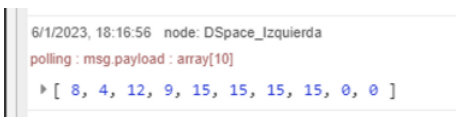
Figura 47
Configuración Modbus Read



3. **FC**: Es la función Modbus a implementar. En lectura, la función 3.
4. **Address**: Especifica el número del registro desde el cual quiere empezar la lectura. Interesa leer desde el primer registro.
5. **Quantity**: Configura la cantidad de registros a leer. En este escenario, se leen 10 registros.
6. **Poll Rate**: Corresponde al tiempo entre solicitudes Modbus.
7. **Server**: Es la dirección física de la conexión entre el dispositivo maestro (EPIC) y la dSPACE (Figura 44).

- **Debug Node**: Permite mostrar la trama Modbus y el estado de la conexión en la ventana *Debug*, es el bloque en color verde. Su funcionamiento se observa en la figura 48.

Figura 48
Nodo Debug

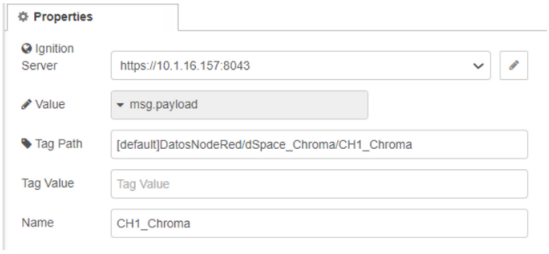


- **Función *Salidas***: Es la encargada de segmentar el vector de la respuesta Modbus. Tiene 8 salidas, cada una corresponde a un canal ADC. El código se encuentra en el anexo 7.

- **Función *Trans_Ignition***: Se encarga de convertir el dato obtenido en un objeto, para transferirlo a Ignition en el siguiente bloque. El código se encuentra en el anexo 8.

- **Ignition Tag Write Node**: Es usado para escribir tags en el servidor Ignition. Se le especifica la dirección del servidor y la raíz del tag que se desea escribir. Es explicado en la sección 4.7.3. La configuración se observa en la figura 49.

Figura 49
Bloque Ignition escribir tag



4.6.3. Lectura de datos módulo de energía con Node-RED. La librería Opto22 (Figura 50) permite la lectura y escritura de módulos instalados en el EPIC. Es usado para enviar los datos al servidor Ignition. El flujo se muestra en la figura 51.

Figura 50
Librería Opto22

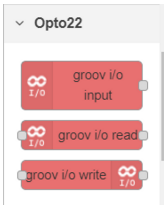
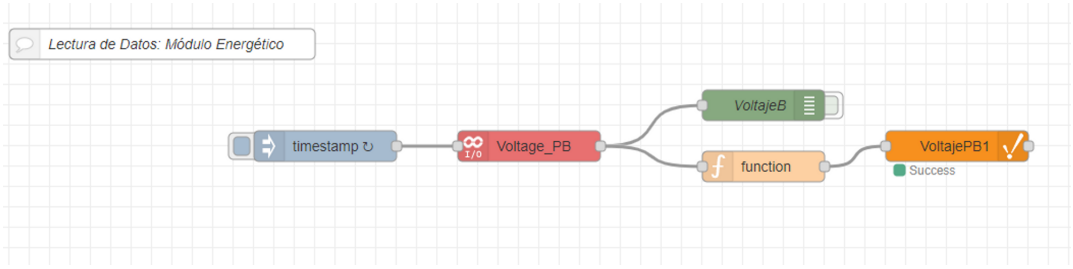
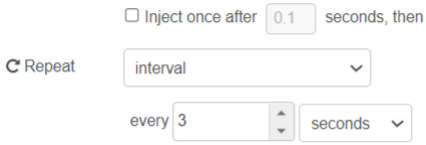


Figura 51
Flujo mediciones de energía



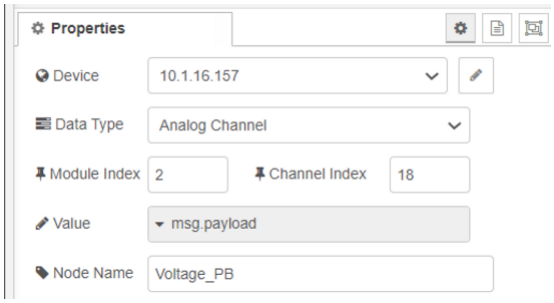
- **Timestamp**: El bloque configura el tiempo entre cada solicitud. **El tiempo mínimo es de 1 [s]**, se configura a 3[s] por limitaciones de hardware (Figura 52).

Figura 52
Bloque Timestamp



- **groov i/o read**: En el flujo como *Voltaje_PB*, es la configuración del canal que se desea leer. Se ajusta como un canal análogo, con el **Módulo 2, perteneciente al módulo de energía**, y envía el voltaje de la fase B, correspondiente al canal #18, como se mostró en la figura 43. Se observa la configuración en la figura 53.

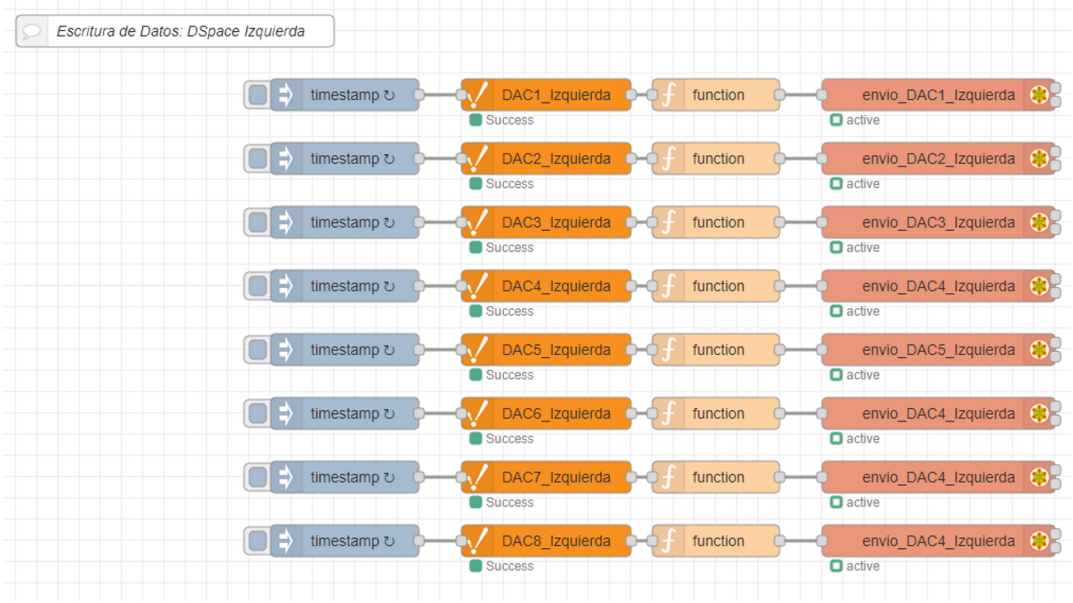
Figura 53
groov i/o read desde Node-RED



- **Función y Bloque Ignition**: Los bloques son idénticos a los descritos en la sección 4.6.2, solo debe cambiar la dirección del tag Ignition.

4.6.4. Escritura de datos dSPACE con Node-RED a través de Ignition. Desde Node-RED se leen los tags modificables en la herramienta de Ignition, su funcionamiento se explica en la sección 4.7. El flujo de trabajo para realizar esta acción se muestra en la figura 54.

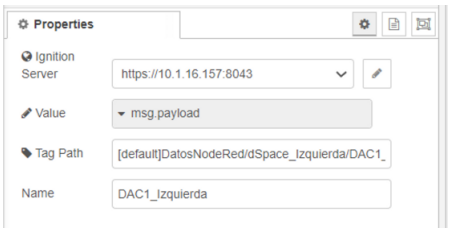
Figura 54
Escritura de datos dSPACE



- **Timestamp**: El bloque es idéntico al de la sección 4.6.3.

- **Ignition Tag Read Node**: Es usado para leer tags alojadas en el servidor Ignition. Se le especifica la dirección del servidor y la raíz del tag que se desea leer (Sección 4.7.3). Se configura como en la figura 55.

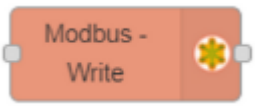
Figura 55
Bloque Ignition leer tag



- **Función transformar Ignition a valor**: Se encarga de convertir el dato obtenido desde Ignition a un valor para ser enviado a través de Modbus en el siguiente bloque directo hacia la dSPACE. El código se encuentra en el anexo 9.

- **Modbus Write**: (Figura 56) Es el nodo encargado de configurar los parámetros Modbus del dispositivo que se va a escribir.

Figura 56
Bloque Modbus Write



Los bloques **envío_DACX_Izquierda** envían el mensaje Modbus de escritura hacia la dSPACE izquierda, su configuración se muestra en la figura 57.

Figura 57
Configuración Modbus Read

1. **Name**: Permite asignarle un nombre al bloque, en este caso, se escribe hacia el DAC1 de la dSPACE izquierda, por lo que se le asigna *envío_DAC1_Izquierda*.

2. **Unit_ID**: Corresponde a la dirección del esclavo (Tabla 8). En este caso es la dirección 10, al tratarse de la dSPACE izquierda.

3. **FC**: Es la función Modbus a implementar. En escritura, la función 6.

4. **Address**: Especifica la dirección del registro al que se va a escribir (Tabla 9). En este caso, se escribe a la dirección 101, para modificar el DACH1.

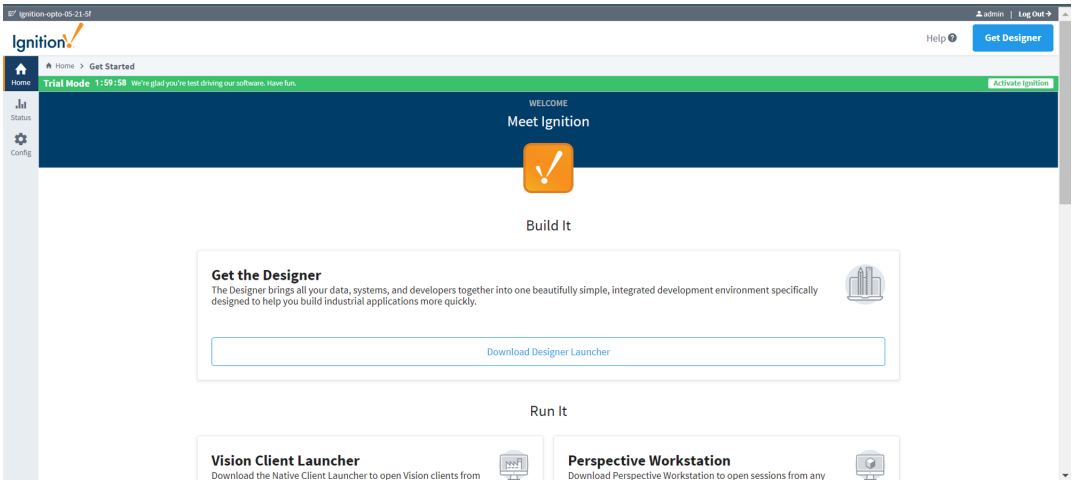
5. **Server**: Es la dirección física de la conexión entre el dispositivo maestro (EPIC) y la dSPACE (Figura 44).

4.7. Ignition 8.1

Es la interfaz visual de control y monitoreo de los ADCs y DACs de las tres dSPACES conectadas al sistema IoT. Desde allí se muestran en forma de tags para ser configuradas en el entorno visual.

4.7.1. Ignition Gateway. Se encuentra alojado en el puerto **8043** del EPIC, se puede acceder desde el groov Manage (Figura 42), o desde la dirección IP ***10.1.16.157:8043***. Se le pedirá al usuario las credenciales de acceso, son las mismas a las presentadas en la sección 4.5. El Ignition Gateway permite al usuario cambiar los ajustes del servidor, agregar módulos, configurar alarmas, etc. Su página principal se muestra en la figura 58.

Figura 58
Ignition Gateway



4.7.2. Ignition Designer. A través de un computador se accede al software de Ignition Designer, donde se configuran los tags, su forma de visualización, y las acciones que se deseen realizar. **Es importante instalar el certificado de red del EPIC a través del groov Manage**. La página principal del designer se observa en la figura 59.

Una vez abierto el proyecto, se configuran las tags que se van a usar, estas son **Memory Tags** con valor inicial **0**. Se guardan en subcarpetas, una para cada dSPACE, y una adicional para el módulo medidor de energía. (Figura 60)

Figura 59
Ignition Designer

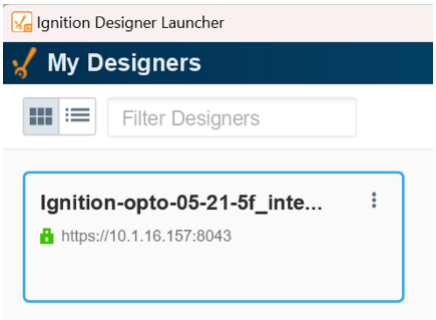
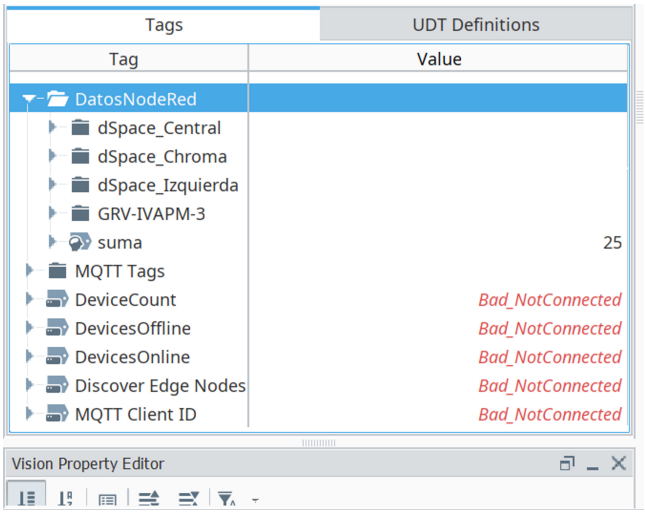


Figura 60
Carpetas en Ignition



De cada dSPACE se muestran los tags de los 8 ADCs y los 8 DACs (Figura 61). El designer debe estar configurado en *read/write mode*, para poder observar y escribir los tags.

Por último, se muestra el tag configurado para la lectura del voltaje de la fase B, desde el monitor de energía. (Figura 62.

Figura 61
Tags de cada dSPACE

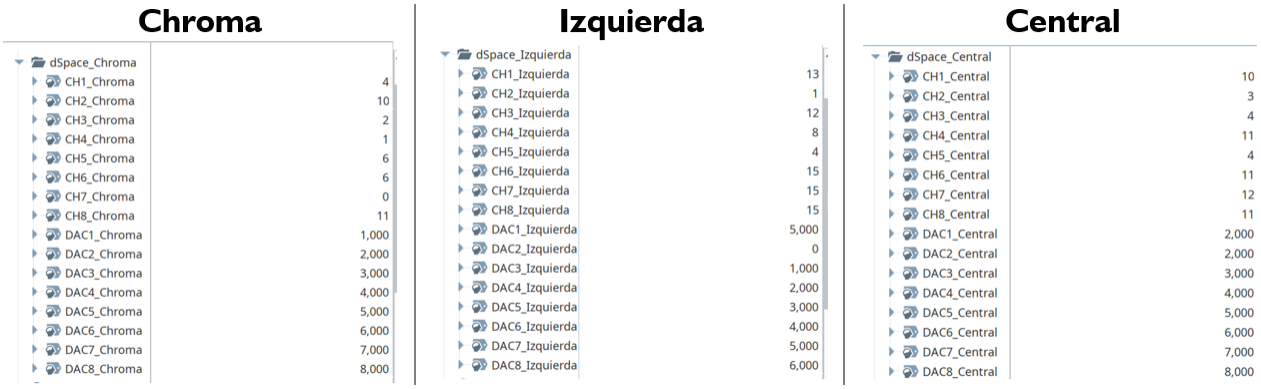


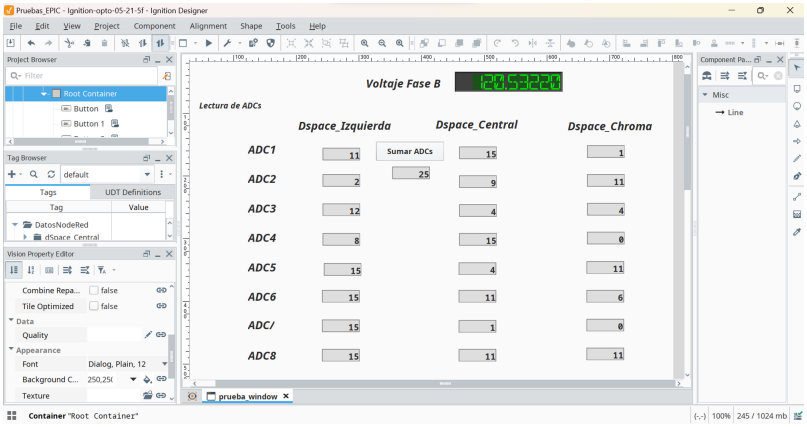
Figura 62
Tag del monitor de energía

4.7.3. Tag Path - Dirección de los tags. La raíz principal pertenece a la carpeta *DatosNodeRed*, de allí se desprenden las nuevas carpetas *dSpace_Central*, *dSpace_Chroma* y *dSpace_Izquierda* (Figura 60). Dentro de estas subcarpetas se encuentran los tags guardados (Figura 61). Debido a esto, si se desea escribir en el tag *CH1_Chroma* como en la figura 49, la dirección será *[default]_DatosNodeRed/dSpace_Chroma/CH1_Chroma*. De la misma forma, si se quiere leer el tag como en la figura 55 dentro del tag *DAC1_Izquierda*, la dirección corresponde a *[default]_DatosNodeRed/dSpace_Izquierda/DAC1_Izquierda*.

4.7.4. Diseño de la herramienta de visualización y control. El esquema general del diseño se muestra en la figura 63.

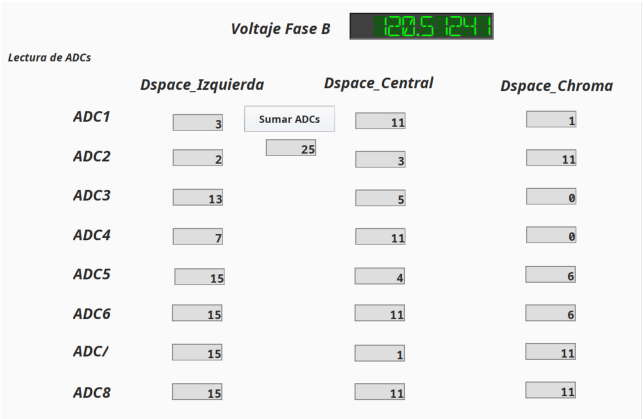
Se compone de dos partes principales:

Figura 63
Interfaz de diseño Ignition



1. **Visualización de ADC**: Se muestra al usuario los ocho ADCs de las dSPACEs, mediante lecturas de voltaje. (Figura 64)

Figura 64
Lectura de los ADCs



2. **Control de DACs**: Le permite al usuario modificar los valores de los DAC, con valores entre *[-10000,10000]*, donde *10000* corresponde a 10 [V].(Figura 65)

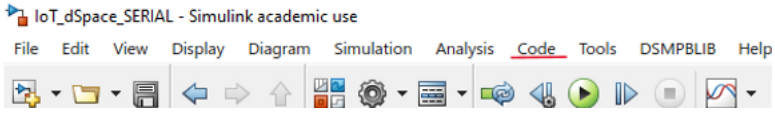
   Una vez el usuario haya colocado los valores, presiona el botón **Modificar Tags**.

Figura 65
Control de los DACs

4.8. ControlDESK

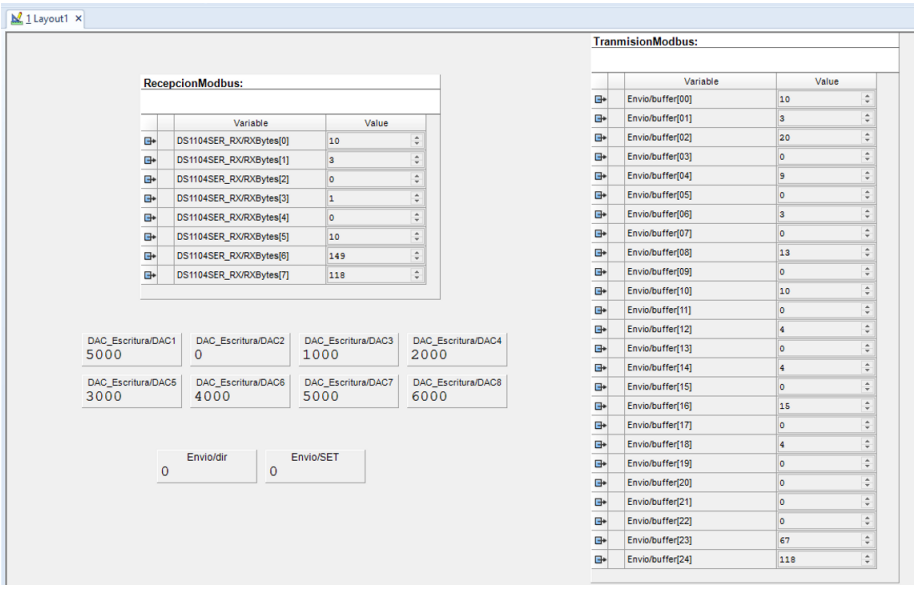
Es la herramienta que permite cargar los flujos de Simulink a la dSPACE. Una vez terminado el diagrama de bloques, se debe construir el código para la interpretación de la tarjeta de control (Figura 66). Este genera un archivo y se carga en la herramienta de ControlDESK.

Figura 66
Construcción del código



Se le agregan las variables del entorno de Simulink que se quieren observar, se sube el programa con el botón **Go Online**, y se monitorizan los datos Modbus que envía y recibe la dSPACE (Figura 67).

Figura 67
Visualización en ControlDESK



4.8.1. Recepción y transmisión en Función 3 Modbus. Desde ControlDESK, se puede observar el mensaje Modbus recibido por la dSPACE a través de Node-RED y el posterior envío de los registros de vuelta a Node-RED. Se muestra en la figura 68.

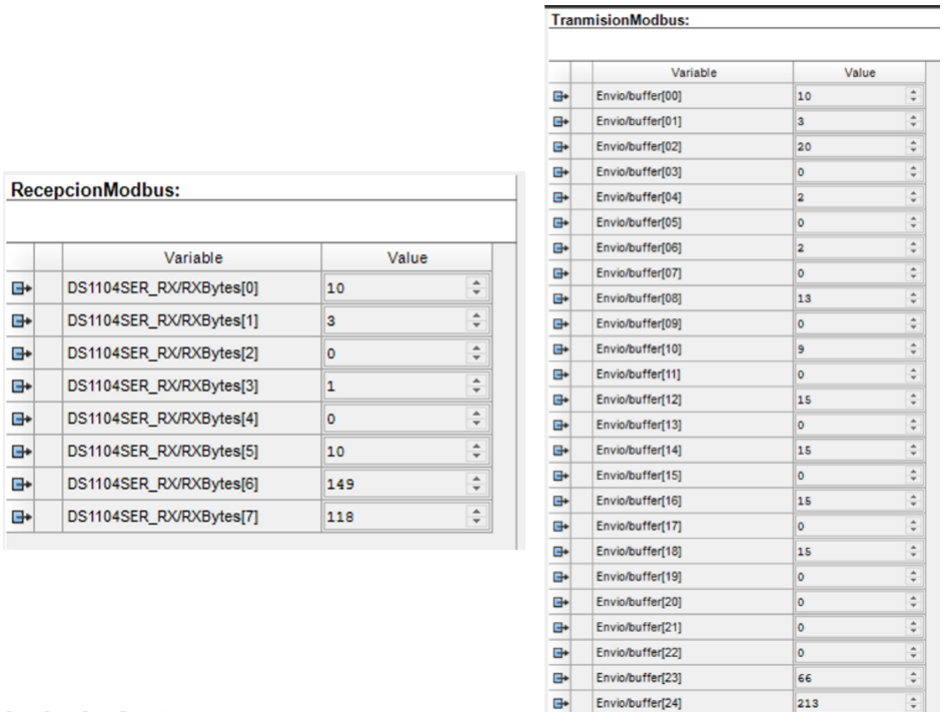
Se hace la solicitud a la dirección 10, con función 3, desde el 1er registro, 10 registros a leer.

4.8.2. Recepción y transmisión en Función 6 Modbus. La función 6 recibe y envía de vuelta la misma trama Modbus desde y hacia la dSPACE. (Figura 69)

Se hace la solicitud a la dirección 10, con función 6, dirección con valores [1, 245], en hexadecimal serían [01,F5], si se concatenan mostraría 01F5, en decimal dirección 501. Para escribir [11, 184], en hexadecimal [0B,B8], concatenados 0BB8, en decimal escribir 3000.

Figura 68

Recepción y envío en Función 3



RecepcionModbus:

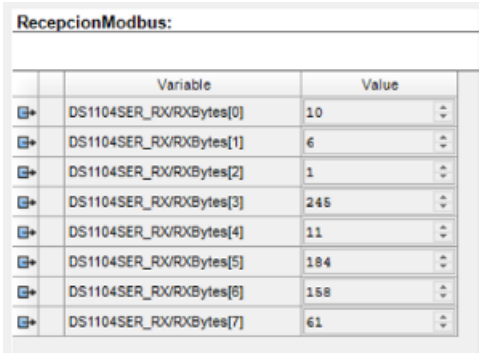
Variable	Value
DS1104SER_RX/RXBytes[0]	10
DS1104SER_RX/RXBytes[1]	3
DS1104SER_RX/RXBytes[2]	0
DS1104SER_RX/RXBytes[3]	1
DS1104SER_RX/RXBytes[4]	0
DS1104SER_RX/RXBytes[5]	10
DS1104SER_RX/RXBytes[6]	149
DS1104SER_RX/RXBytes[7]	118

TranmisionModbus:

Variable	Value
Envio/buffer[00]	10
Envio/buffer[01]	3
Envio/buffer[02]	20
Envio/buffer[03]	0
Envio/buffer[04]	2
Envio/buffer[05]	0
Envio/buffer[06]	2
Envio/buffer[07]	0
Envio/buffer[08]	13
Envio/buffer[09]	0
Envio/buffer[10]	9
Envio/buffer[11]	0
Envio/buffer[12]	15
Envio/buffer[13]	0
Envio/buffer[14]	15
Envio/buffer[15]	0
Envio/buffer[16]	15
Envio/buffer[17]	0
Envio/buffer[18]	15
Envio/buffer[19]	0
Envio/buffer[20]	0
Envio/buffer[21]	0
Envio/buffer[22]	0
Envio/buffer[23]	66
Envio/buffer[24]	213

Figura 69

Recepción y envío en Función 6



RecepcionModbus:

Variable	Value
DS1104SER_RX/RXBytes[0]	10
DS1104SER_RX/RXBytes[1]	6
DS1104SER_RX/RXBytes[2]	1
DS1104SER_RX/RXBytes[3]	245
DS1104SER_RX/RXBytes[4]	11
DS1104SER_RX/RXBytes[5]	184
DS1104SER_RX/RXBytes[6]	158
DS1104SER_RX/RXBytes[7]	61

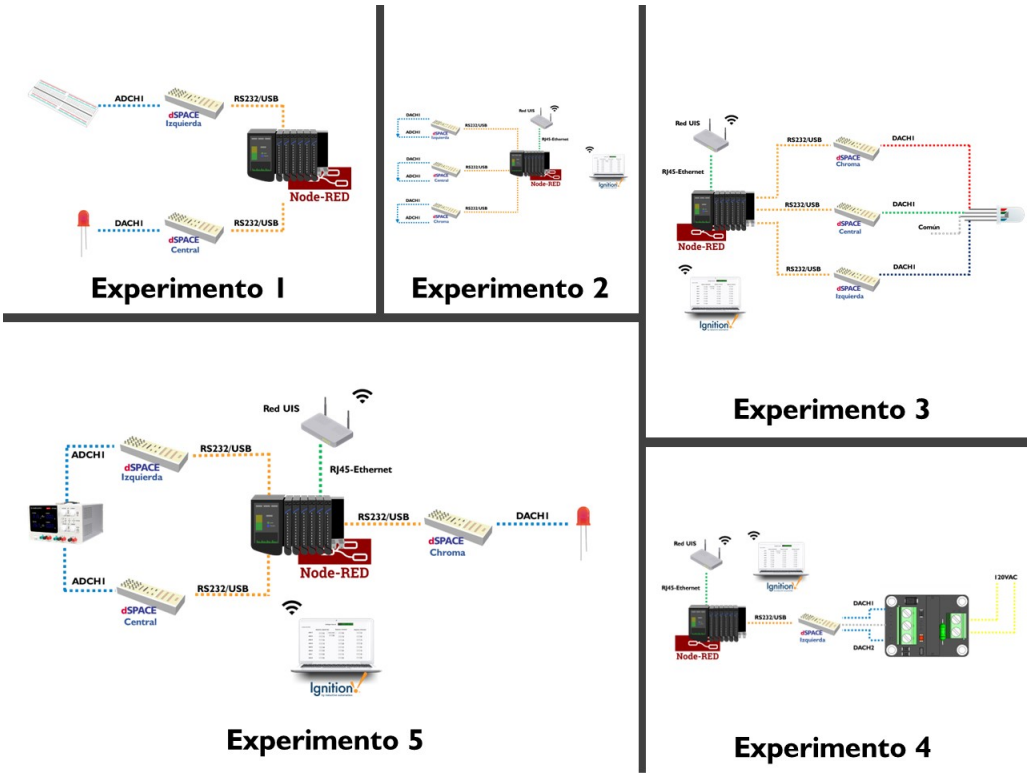
5. Verificación del funcionamiento del sistema diseñado

El objetivo de este capítulo es mostrar una serie de experimentos realizados con el fin de validar que el sistema diseñado cumpliese con los requerimientos y las expectativas de todos los interesados. El sistema diseñado tiene en mente las necesidades de:

- Monitorear variables eléctricas.
- Modificar valores de las dSPACE.

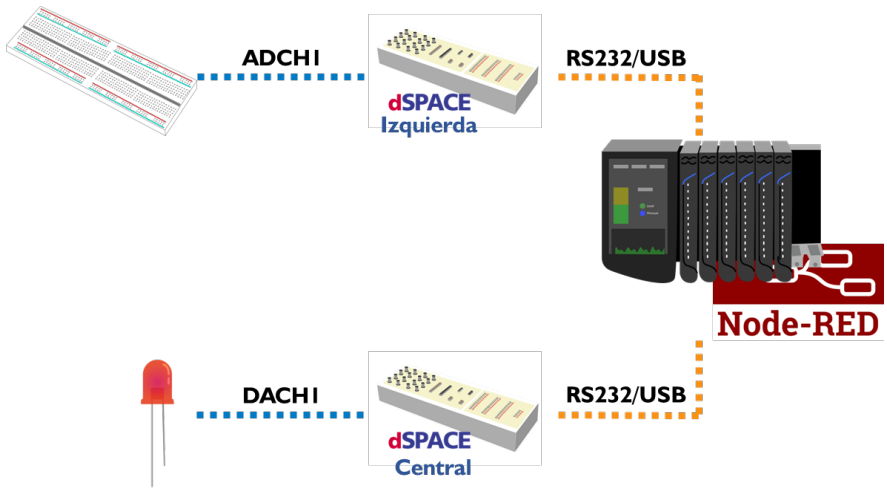
En la figura 70 se muestran esquemáticos asociados a los cinco experimentos realizados:

Figura 70
Experimentos realizados



Experimento 1: Lectura de potenciómetro y control de LED.

Descripción: Este experimento consiste en encender un LED usando como criterio la lectura de ADC de otra dSPACE.



Materiales:

1. Dos dSPACE y sus módulos de conexión.
2. Un Opto22 EPIC.
3. Dos conversores USB a Serial.
4. Dos cables extensores de RS232
5. Un concentrador (hub) USB
6. Un potenciómetro.
7. Un LED.
8. Fuente DC.
9. Cables y multímetro.

Paso a paso:

1. Conectar los módulos de conexión de las dSPACE al EPIC a través del concentrador USB, usando los conversores USB a serial y las extensiones de RS232.

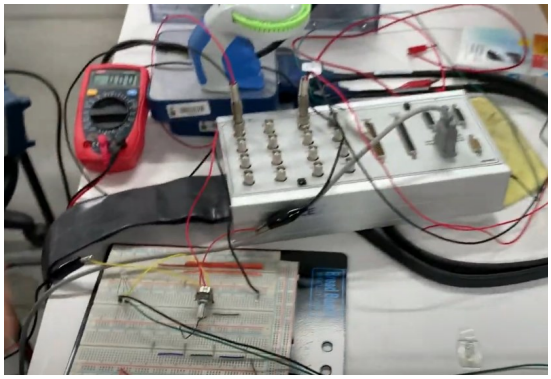
2. Realizar un divisor resistivo con un potenciómetro y una fuente DC.

3. Conectar el divisor resistivo al ADCH1 de la primera dSPACE.

4. Conectar el LED en el DACH1 de la segunda dSPACE.

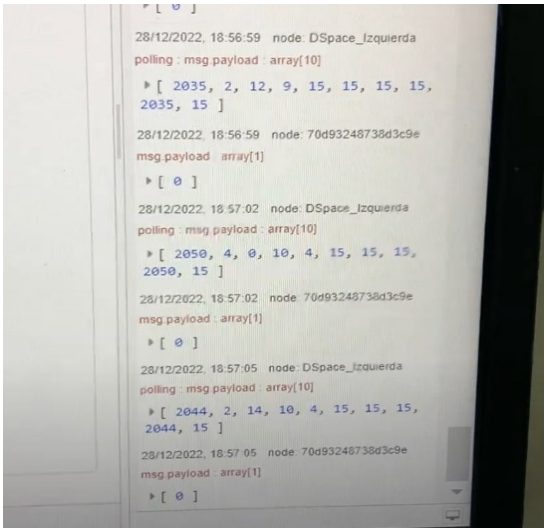
5. Realizar un flujo en Node-RED que lea los datos del ADCH1 de la primera dSPACE, compare con el umbral definido y en base a ese valor, emita una tensión de salida al DACH1 de la segunda dSPACE.

Experimento 1: Resultados

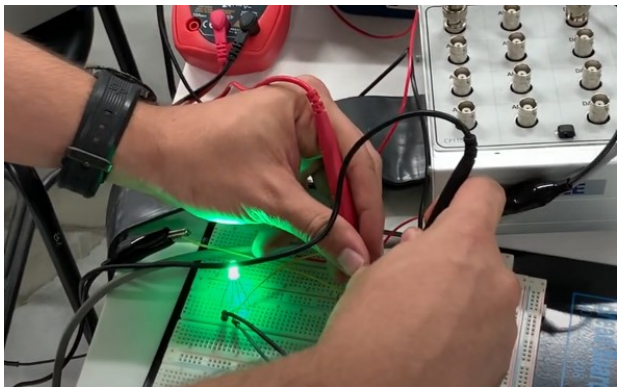


Configuración inicial

Lectura inicial del ADC



Datos seriales llegando a Node-RED



Encendido del LED posterior a la modificación de tensión en el divisor resistivo

5.1. Experimento 1: Lectura de potenciómetro y control de LED

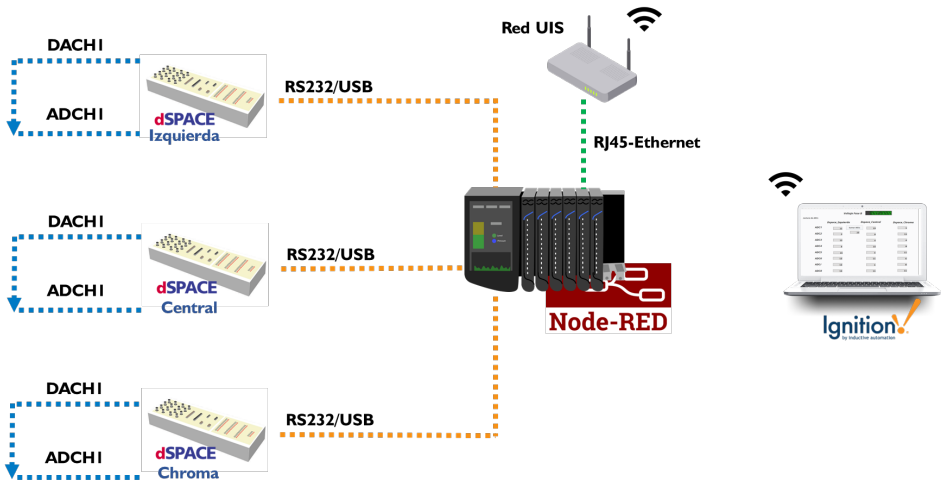
El primer experimento consiste en una prueba de comunicación entre dSPACEs básica. Se utilizaron dos dSPACE. Una de las tarjetas se encargó de medir el voltaje a través de un divisor resistivo y enviar ese valor medido a la otra tarjeta. La segunda tarjeta comparó el valor recibido con un umbral previamente establecido en 2500 (2.5 V). Si el valor medido superaba este umbral, la tarjeta modificaba uno de sus convertidores Digitales-Analógicos (DAC) para emitir 3 V, encendiendo así un LED conectado a este DAC. Este experimento demostró la capacidad que tiene el sistema diseñado de comunicar a las tarjetas dSPACE entre sí e intercambiar información.

Este hizo uso de Modbus y Rs232 para la llegada de comunicaciones seriales al EPIC, específicamente a Node-RED, ya estando en esta capa de la arquitectura, se definieron los bloques con el funcionamiento deseado: Un bloque de lectura de Modbus cuya salida pasa por otro bloque en donde compara contra el umbral (2500) y si cumplía la condición entonces retornaba el valor de 3000 al bloque que emitía la salida al DAC; si no cumplía la condición, en vez de retornar 3000, se retornaba 0. Además de validar la posibilidad de comunicación entre tarjetas, este experimento demuestra el poder de node–RED para definir el comportamiento del sistema posterior a Modbus y de las comunicaciones seriales, ya que este flujo no es el backend principal del proyecto que toma los datos y los envía a Ignition, es un backend diseñado para cumplir propósitos muy específicos.

Video: `https://tinyurl.com/primerexpiot` - `https://youtu.be/eUxwKWDFmqs`

Experimento 2: Modificación de valores de los DAC a través de Ignition.

Descripción: Este experimento consiste en validar completamente la arquitectura de comunicaciones, mostrando la lectura de ADCs y modificaciones de los DACs en Ignition.



Materiales:

1. Tres DSPACE y sus módulos de conexión.
2. Un Opto22 EPIC.
3. Tres conversores USB a Serial.
4. Tres cables extensores de RS232.
5. Un concentrador (hub) USB.

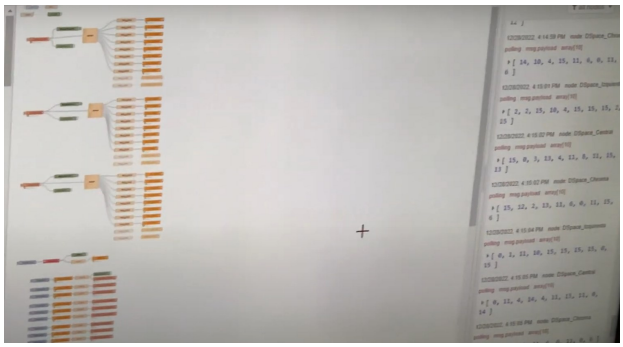
Paso a paso:

1. Conectar los módulos de conexión de las dSPACE al EPIC a través del concentrador USB, usando los conversores USB a serial y las extensiones de RS232.
2. Conectar, en cada dSPACE, el ADCH1 con el DACH1.
3. Monitorear desde Ignition las lecturas de los ADC.
4. Modificar los DACs y ver en Ignition como estos valores se pintan en los ADC.

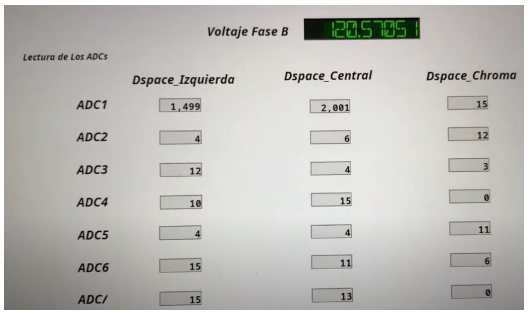
Experimento 2: Resultados

Escenario de experimentación, computadores mostrando ControDesk y módulos de conexión de las dSPACE.

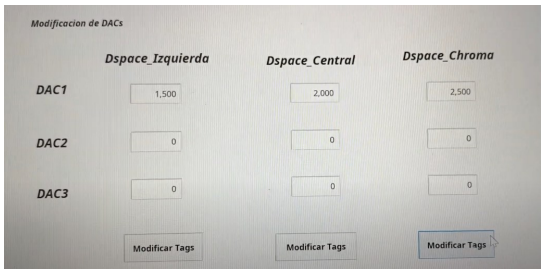
Conexión de dSPACEs al concentrador USB



Visión del Backend principal



Interfaz en Ignition: Valores de ADCs y fase B



Interfaz en Ignition: Modificación de DACs

5.2. Experimento 2: Modificación valores DAC a través de Ignition

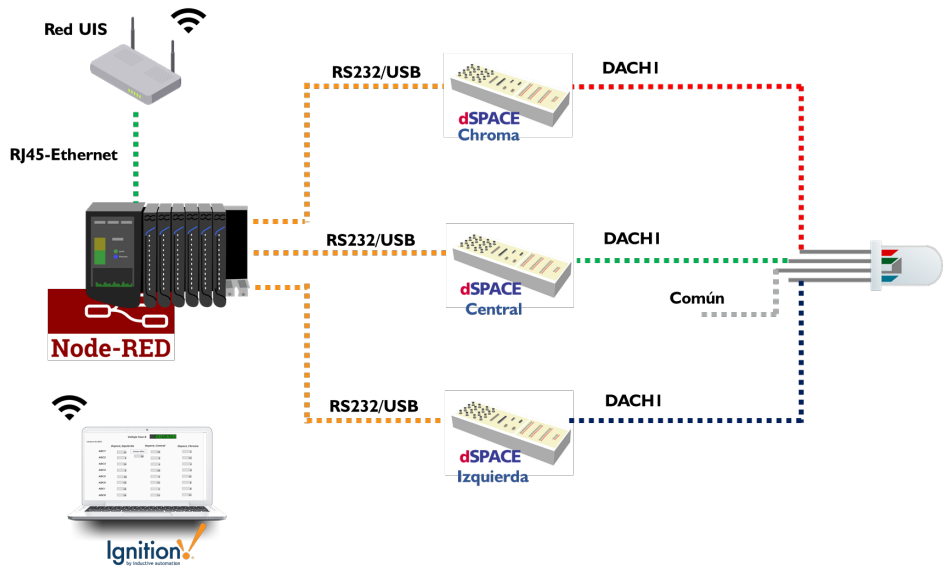
Este segundo experimento fue realizado teniendo en mente realizar una validación completa de toda la arquitectura de comunicaciones, desde el punto más bajo en las comunicaciones seriales, hasta la capa de usuario en Ignition. Se muestran las tres tarjetas (Izquierda, Central y Chroma) conectadas por RS232 al concentrador USB que a su vez se encuentra conectado al EPIC, luego se muestra en NodeRED los datos llegando a la consola y posteriormente se muestran estos mismos valores pero en Ignition, en conjunto con la lectura del voltaje de la fase B del laboratorio, la cual estaba siendo medida por el módulo energético embebido en el EPIC. De esta manera, ya se valida la lectura y monitoreo de los ADCs de las dSPACEs por lo que, para validar la modificación de los DACs, lo que se hizo fue conectar el primer DAC de cada dSPACE a su respectivo primer canal de ADC y de esta forma, cuando se modificara un valor de DAC desde Ignition, tendría que ser exactamente el mismo valor que se lee en el apartado de lectura de ADCs. Este experimento con resultados satisfactorios, además de validar la arquitectura de comunicaciones, valida la capa de usuario, desde Ignition se pueden plantear diseños de la interfaz, mostrar ciertos valores al usuario y darle una sección en donde puede modificar otros valores.

Video Funcionamiento parte 1: `https://tinyurl.com/segundoexpiot` - `https://youtu.be/-RAsCF8lubU`

Video Funcionamiento parte 2: `https://tinyurl.com/segundoexpiot2` - `https://youtu.be/RedYJvqpzbs`

Experimento 3: LED RGB.

Descripción: Este experimento consiste en controlar el color de un LED RGB usando las tres DSPACE.



Materiales:

1. Tres DSPACE y sus módulos de conexión.
2. Un Opto22 EPIC.
3. Tres conversores USB a Serial.
4. Tres cables extensores de RS232.
5. Un concentrador (hub) USB.
6. Un LED RGB.

Paso a paso:

1. Conectar los módulos de conexión de las dSPACE al EPIC a través del concentrador USB, usando los conversores USB a serial y las extensiones de RS232.

2. Conectar el primer DAC de la dSPACE “izquierda” al terminal azul del LED.

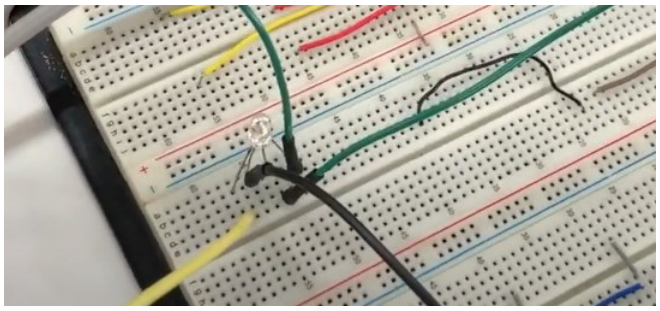
3. Conectar el primer DAC de la dSPACE “central” al terminal verde del LED.

4. Conectar el primer DAC de la dSPACE “chroma” al terminal rojo del LED.

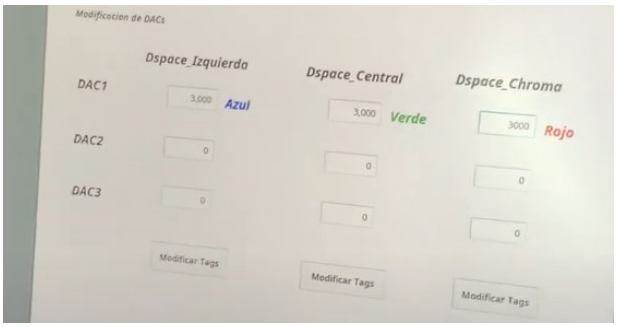
5. Conectar el ánodo del LED a la tierra de cualquier dSPACE

6. Modificar los valores de los DACs con 3000 (3 V) dependiendo del color que se quiera ver.

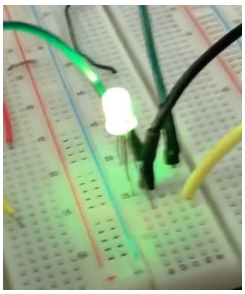
Experimento 3: Resultados



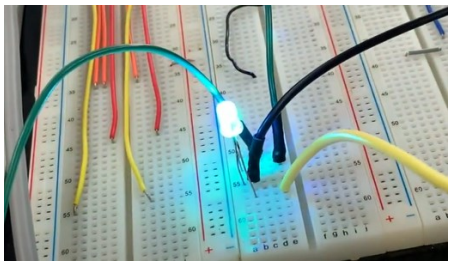
Montaje con el LED RGB



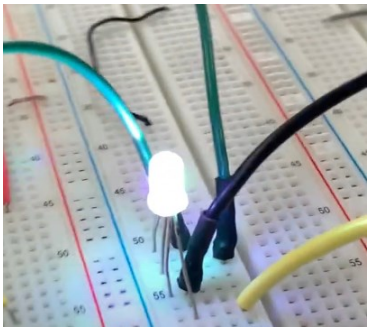
Interfaz en Ignition para modificar los DACs



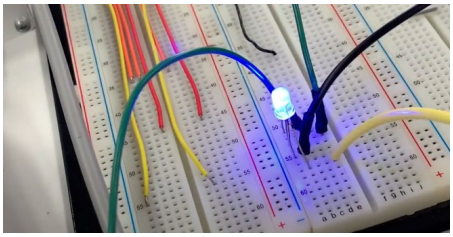
dSPACEs “Chroma” y “Central”



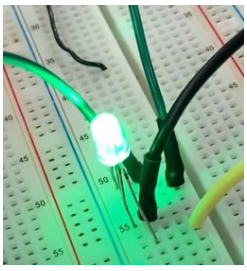
dSPACEs “Izquierda” y “Central”



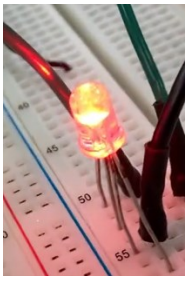
dSPACEs “Izquierda” , “Central” y “Chroma”



dSPACE “Izquierda”



dSPACE “Central”



dSPACE “Chroma”

5.3. Experimento 3: LED RGB

A pesar de que la modificación de los valores de los DACs ya fue validada en el punto anterior, este experimento fue hecho teniendo en mente el realizar múltiples modificaciones de estos valores en un periodo relativamente corto entre sí y analizar el comportamiento del sistema. Para este fin se utilizó un LED RGB, el cual se caracteriza por tener tres cátodos y un ánodo común, cada cátodo representa un color (Red: Rojo, Green: Verde, Blue: Azul). El DAC1 de la dSPACE Izquierda se encontraba conectado al cátodo azul, el DAC1 de la dSPACE Central se encontraba conectado al cátodo verde y el DAC1 de la dSPACE Chroma se encontraba conectado al cátodo rojo. Desde ignition se modificaron las tensiones de salidas de cada uno de los DACs y se probaron las múltiples combinaciones de colores que se podían obtener del LED desde que se introduce un valor en Ignition, luego este pasa a NodeRED y finalmente viaja por RS232 hasta la dSPACE. **Video Funcionamiento:** `https://tinyurl.com/tercerexperimentoiot` - `https://youtu.be/qaTIZMqaRC8`

Video Explicación: `https://tinyurl.com/tercerexperimentoiot2` - `https://youtu.be/lvHDQJZ3v0A`

Experimento 4: Relé.

Descripción: Este experimento consiste en controlar el estado de un relé desde Ignition.

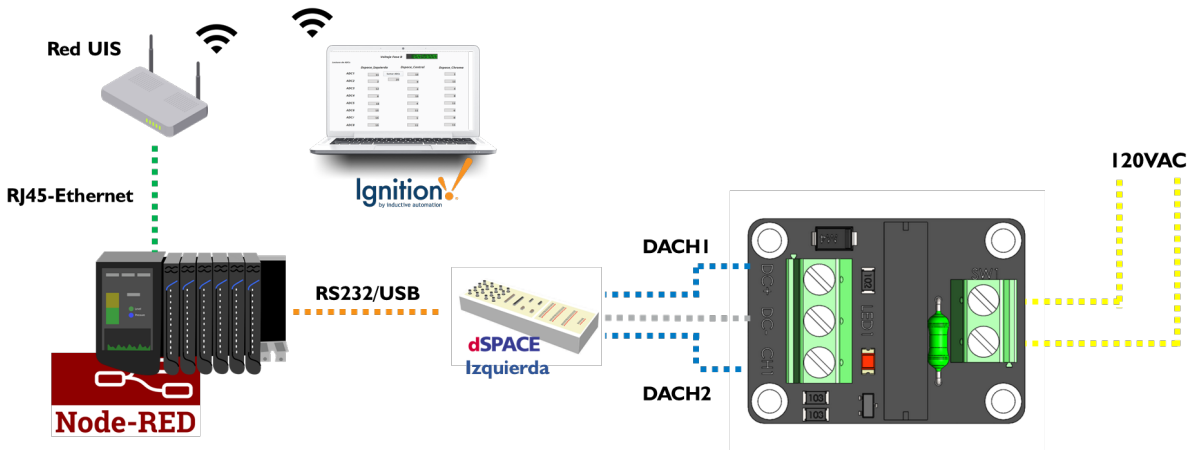


Figura relé de: grabcad.com

Materiales:

1. Una dSPACE y sus módulos de conexión.
2. Un Opto22 EPIC.
3. Un conversor USB a Serial.
4. Un cable extensor de RS232.
5. Un concentrador (hub) USB.
6. Un relé, en particular se probó con uno de estado sólido.

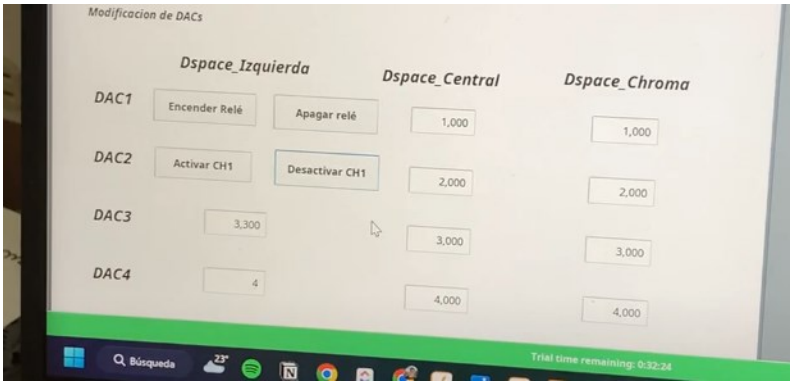
Paso a paso:

1. Conectar el módulos de conexión de la dSPACE al EPIC a través del concentrador USB, usando el conversor USB a serial y la extensión de RS232.

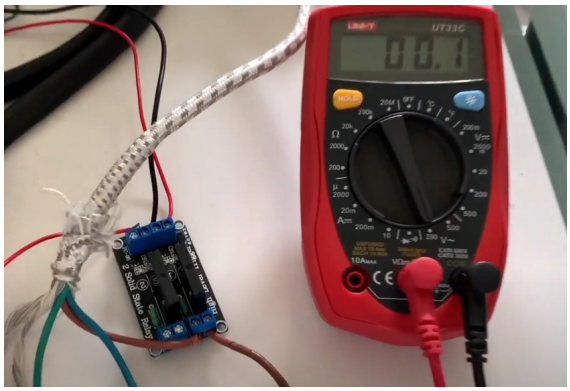
2. Conectar el primer DAC de la dSPACE “izquierda” a la alimentación del relé, positivo con 5VDC y negativo con tierra del relé.

3. Conectar el segundo DAC de la dSPACE “Izquierda” a los terminales de control del relé.

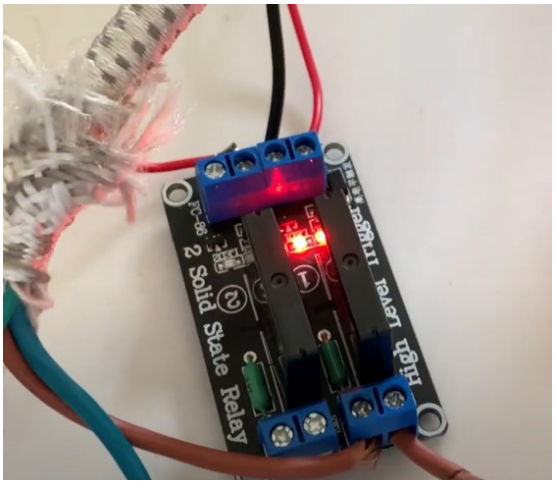
Experimento 4: Resultados



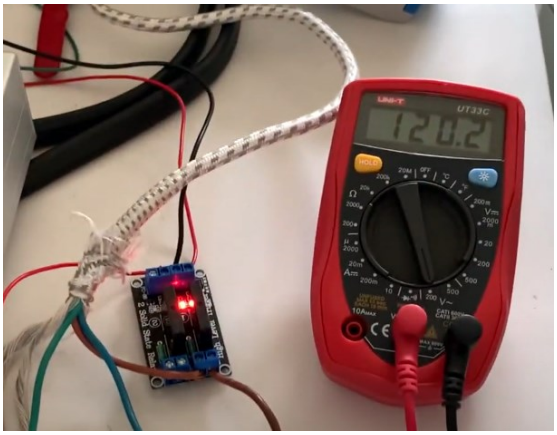
Interfaz de Ignition con los botones para controlar el relé



Relé apagado



Relé encendido



Relé encendido y activo

5.4. Experimento 4: Relé de estado sólido

Un relé es un elemento interruptor que, a partir de recibir tensión en uno de sus terminales de control, cierra sus terminales de potencia. El relé de estado sólido se caracteriza por no contar con elementos mecánicos, ya que es fabricado con electrónica de estado sólido. El objetivo de este experimento fue demostrar como desde Ignition se puede cerrar o abrir un relé, lo cual puede ser una aplicación interesante en el contexto de la microrred y pensado para la futura plataforma de aprendizaje. Para este experimento se hizo uso de una sola dSPACE y dos de sus DACs, uno para controlar si el relé está encendido o apagado y otro para controlar si el relé se activa (cierra los terminales de potencia) o se desactiva (abre los terminales de potencia). Desde un punto de vista funcional, este experimento no válida nada nuevo en relación con la modificación de valores de los DACs desde Ignition, sin embargo, sí valida la realización de backend desde esta plataforma.

Durante toda la ejecución del proyecto, node-RED fue utilizado como la herramienta principal para realizar todo el backend del proyecto, sin embargo, en este experimento (y el siguiente) se explotó la capacidad que tiene Ignition de hacer scripting y desencadenar eventos determinados a partir de las interacciones del usuario. Cuando el usuario presiona el botón “Encender relé” en Ignition se desencadena un evento que escribe 5 V en el tag relacionado al DAC1, sucede de la misma manera cuando el usuario presiona el botón “Activar relé”. El punto es que la configuración de ese valor (5 V) no se hace en node-RED sino desde el mismo Ignition, por lo que para realizar este experimento, que tiene un contexto específico como lo es activar o desactivar un relé, no se tuvo que modificar el backend principal del proyecto en Node-RED, sino que se sacó provecho de

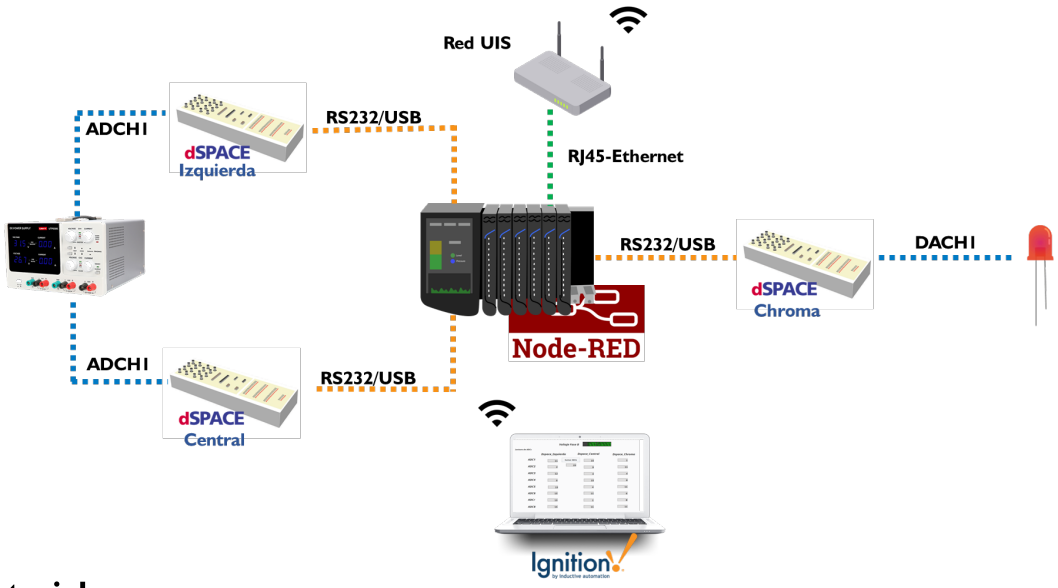
las funcionalidades de Ignition.

Video Funcionamiento: https://tinyurl.com/cuartoexperimentoiot - https://youtu.be/1PwuS1H_JDM

Video Explicación: https://tinyurl.com/cuartoexperimentoiot2 - https://youtu.be/bFtLU58bbyM

Experimento 5: Suma de variables eléctricas y encendido de un LED.

Descripción: Este experimento consiste en operar dos valores obtenidos de los ADC de las dSPACE y en base a esto, determinar si se enciende o se apaga un LED.



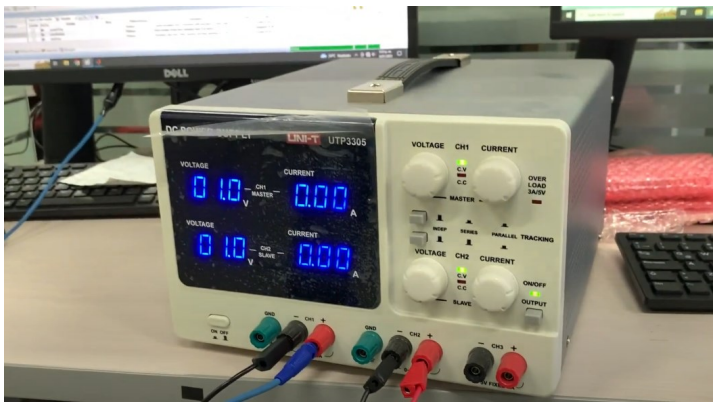
Materiales:

1. Tres dSPACE y sus módulos de conexión.
2. Un Opto22 EPIC.
3. Tres conversores USB a Serial.
4. Tres cables extensores de RS232.
5. Un concentrador (hub) USB.
6. Fuente DC dual.
7. Un LED.

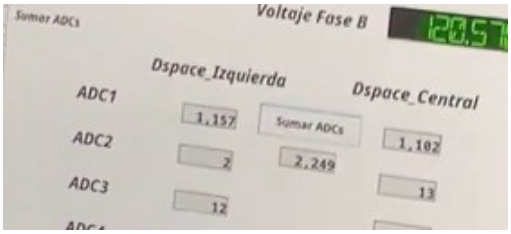
Paso a paso:

1. Conectar los módulos de conexión de las dSPACE al EPIC a través del concentrador USB, usando los conversores USB a serial y las extensiones de RS232.
2. Conectar el primer canal de la fuente dual al ADC1 de la dSPACE “Izquierda”.
3. Conectar el segundo canal de la fuente dual al ADC1 de la dSPACE “Central”.
4. Conectar un LED al primer DAC de la dSPACE “Chroma”.

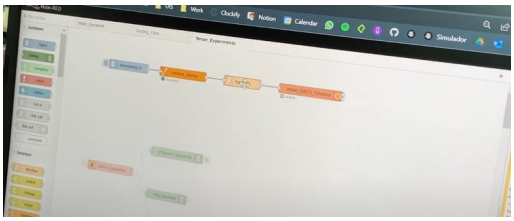
Experimento 5: Resultados



Fuente DC dual

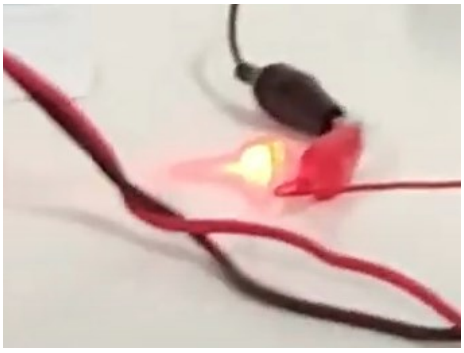


Interfaz de Ignition que permite sumar los ADCs



Flujo en Node-RED para este experimento

Función en Node-RED que verifica el umbral



Encendido del LED

5.5. Experimento 5: Suma de variables eléctricas y encendido de un LED

Este experimento complementa una de las ideas del experimento anterior, pues realiza lógica de funcionamiento tanto en Ignition como en Node-RED. Se hace uso de las dSPACE central e izquierda y un ADC de cada una. Estos ADCs se encuentran leyendo la tensión en DC que se le suministra con una fuente dual presente en el LIE, el objetivo es realizar operaciones con estos valores, en particular se definió que si la suma de los ADCs era superior a 5000 (5V), entonces la tercera dSPACE (dSPACE Chroma) emitiera 3 V en uno de sus DACs para encender un LED.

El usuario desde Ignition tiene lectura de los ADCs y en un instante determinado puede presionar el botón ”Sumar ADCs”. Al ejecutar esta acción se desencadena un evento en Ignition en el que se toman los dos valores de los ADCs, se suman y luego se escriben en un tag para que posteriormente node-RED haga lectura de este valor que ya contiene la suma, compare contra el umbral definido (5000) y finalmente emita 3 V en la salida si se cumple que la suma es mayor al umbral, en caso de no serlo, el DAC emite 0V.

Este experimento hace uso de todo el trabajo realizado en el proyecto, valida toda la arquitectura de comunicaciones y demuestra un punto muy importante y es que la lógica de funcionamiento no está remitida exclusivamente a ser descrita en Node-RED, también se pueden definir ciertas etapas de la lógica de funcionamiento en Ignition, como en este caso se hizo sumando los valores de los ADCs dentro de Ignition. Otra opción era tomar la lectura de cada ADC, mandarla a node-RED y dentro de node-RED realizar la suma. Este experimento, por lo tanto, valida que se tiene una arquitectura flexible que puede tener diversas combinaciones de funcionamiento en

la que es el desarrollador quien, en posesión del conocimiento de las herramientas, decide como describir el comportamiento del sistema en las capas superiores con el fin de dar solución a un problema o experimento que se quiera realizar, soportándose de que en las capas inferiores el sistema funciona con Modbus y con comunicaciones seriales para interactuar con las dSPACE.

Video Funcionamiento: `https://tinyurl.com/quintoexperimentoiot` - `https://youtu.be/Y3XM5ZX-q7I`

Video Explicación: `https://tinyurl.com/quintoexperimentoiot2` - `https://youtu.be/yReKLWT3OUk`

6. Conclusiones

En las primeras etapas del proyecto, se analizaron los requerimientos y limitaciones de los interesados, así como la disponibilidad de recursos. A partir de esta evaluación, se obtuvieron varias conclusiones importantes para la adquisición de equipos, la elección del software y el desarrollo del proyecto. Una de estas conclusiones de gran importancia es que la microrred eléctrica, como ya fue mencionado, al no encontrarse totalmente terminada, necesitaba de que el diseño planteado tuviese cierta modularidad. La microrred es un proyecto del grupo de investigación diferente al de IoT, por tanto, tiene sus propios tiempos de ejecución que son ajenos a este proyecto. Debido a esto, el sistema a diseñar debía ser modular, capaz de adaptarse a la microrred cuando ya se encuentre finalizada.

Otro aspecto importante a tener en cuenta es el crecimiento a futuro en los posibles escenarios de investigación que se pueden abordar en la microrred eléctrica. Un ejemplo de estos es la capacidad a largo plazo de realizar pruebas que involucren control sobre diversos actuadores u otro tipo de elementos.

En conclusión, este trabajo de tesis ha demostrado que es posible diseñar e implementar una arquitectura flexible para el procesamiento de datos en sistemas IoT. La implementación de los desarrollos realizados en Ignition y Node-RED ha permitido una mayor flexibilidad en la monitorización y control de una microrred eléctrica experimental. Los resultados obtenidos en los experimentos realizados han demostrado la eficacia de la arquitectura propuesta, siendo capaz de satisfacer las necesidades de investigación del Laboratorio de Integración Energética.

En este proyecto, se ha utilizado como protocolo de comunicación Modbus para conectar dispositivos seriales mediante RS232. También se ha aprendido a utilizar el software Ignition y Node-RED para procesar los datos recolectados de estos dispositivos de manera eficiente. Estas habilidades y conocimientos adquiridos son valiosos para futuros proyectos relacionados con IoT y Industria 4.0.

En general este trabajo de tesis ha sido una excelente oportunidad para aplicar los conocimientos adquiridos durante la carrera en un proyecto real y proporciona una base sólida para futuras investigaciones y desarrollos en el campo de las microrredes eléctricas y IoT. Además, el uso de dispositivos seriales conectados mediante RS232 y el protocolo de comunicación Modbus ha permitido una mayor flexibilidad y escalabilidad en la monitorización y control de la microrred eléctrica experimental.

Referencias Bibliográficas

Acolgen (2023). https://acolgen.org.co/, consultado el 19 de enero de 2023.

Alessia Cagnano, Enrico De Tuglie, P. M. (2020). Microgrids: Overview and guidelines for practical implementations and operation. *Applied Energy*, 258.

Chaudhary, S., Johari, R., Bhatia, R., Gupta, K., Bhatnagar, A., and Knuth, D. E. (2019). Craiot: Concept, review and application(s) of iot. *2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU)*, pages 1–4.

Consejo Internacional de Grandes Redes Eléctricas (2020). http://www.cigrecolombia.org/documents/memorias/workshop-c6-29-01-2020/8-%20colombia%20inteligente%20-%20juan%20david%20molina.pdf, consultado el 24 de enero de 2023.

dSPACE (2022). https://www.dspace.com/en/pub/home/products/hw/singbord/ds1104.cfm, consultado 16 de enero de 2023.

dSPACE (2023). https://www.dspace.com/en/inc/home/products/sw/experimentandvisualization/controldesk.cfm, consultado 17 de enero de 2023.

dSPACE GmbH (2016). *HelpDesk 2016-B*. dSPACE.

General Electric (2023). https://www.ge.com/digital/documentation/cimplicity/version11/oxy_ex-2/device_communications/topics/g_cimplicity_device_

communications_function_codes_supportedby_modbus_rtu_slave.html, consultado 16 de enero de 2023.

Haussman, R. and Domínguez, J. (2023). *Knowledge, Technology and Complexity in Economic Growth*. Real Colegio Complutense.

Ibrahim, D. (2014). Intermediate pic18 projects. *PIC Microcontroller Projects in C*.

IONOS (2020). https://www.ionos.com/digitalguide/server/know-how/crc-error/, consultado 16 de enero de 2023.

Lis Data Solutions (2022). https://www.lisdatasolutions.com/es/blog/que-es-la-industria-4-0/, consultado 17 de enero de 2023.

MathWorks (2023a). https://www.mathworks.com/discovery/what-is-matlab.html, consultado 17 de enero de 2023.

MathWorks (2023b). https://www.mathworks.com/products/simulink.html, consultado 17 de enero de 2023.

Ministerio de Ciencia, Tecnología e Innovación (2021). https://minciencias.gov.co/convocatorias/programa-y-proyectos-ctei/convocatoria-para-el-fortalecimiento-ctei-en-instituciones, consultado 17 de enero de 2023.

Modbus Organization (2023). https://modbus.org/, consultado 16 de enero 2023.

Node-RED (2023). https://nodered.org/, consultado 17 de enero de 2023.

Opto 22 (2023). https://www.opto22.com/products/groov-epic-system, consultado 17 de enero de 2023.

Opto22 (2021). https://blog.opto22.com/optoblog/new-groov-epic-firmware-3.3.1-is-here, consultado el 19 de enero de 2023.

Opto22 (2022a). *groov EPIC PROCESSOR*. Opto22.

Opto22 (2022b). *groov RIO ENERGY MONITORING UNIT*. Opto22.

Opto22 (2022c). https://documents.opto22.com/2245_groov_epic_processor_data_sheet.pdf, consultado el 20 de enero de 2023.

Punto Flotante S.A. (2021). https://www.puntoflotante.net/rs485.htm, consultado 16 de enero de 2023.

Rf Wireless World (2012). https://www.rfwireless-world.com/terminology/modbus-message-frame.html, consultado 16 de enero de 2023.

Schneider Electric (2022). https://www.se.com/us/en/faqs/fa168406/, consultado 16 de enero de 2023.

Sorri, K., Mustafee, N., and Seppänena, M. (2022). Revisiting iot definitions: A framework towards comprehensive use. *Technological Forecasting and Social Change*, 179.

Starynets, O. (2016). Communication in microgrids and virtual power plants. Master's thesis, The Artic University of Norway.

Vermesan, D. O., Friess, D. P., Guillemin, P., Gusmeroli, S., Sundmaeker, H., Bassi, D. A., Jubert, I. S., Mazura, D. M., Harrison, D. M., Eisenhauer, D. M., and Doody, D. P. (2011). Internet of things strategic roadmap. *Internet of things - global technological and societal trends*, 1(2011):9–52.

Apéndices

Apéndice A. Función Generadora CRC

```
function CRC_Bytes = CRC_Calculo(mensaje)
%Los ultimos 2 bytes de Modbus son los generadores de CRC, lo que
%hace la funcion es obtener el mensaje que llega, y a partir de este,
%generar los dos bytes de CRC, para ser comparados en el siguiente bloque.
% Inputs:
% mensaje,    El array de datos arrojados por la solicitud modbus
% Outputs:
% CRC_Byes,   Array de dos posiciones con los bytes de CRC
        N = length(mensaje)-2;
        %Se quitan los ultimos 2 valores del array mensaje, debido a que
        %estos son los generados por la solicitud Modbus, la funcion
        %calcula el CRC a partir del mensaje obtenido.
        %Inicio Funcion generadora de CRC:
  crc = uint32(hex2dec('ffff'));
  polynomial = hex2dec('a001');
  for i = 1:N
   crc = bitxor(crc,mensaje(i));
   for j = 1:8
```

```
    if bitand(crc,1)
     crc = bitshift(crc,-1);
     crc = bitxor(crc,polynomial);
    else
     crc = bitshift(crc,-1);
    end
   end
  end
  lowByte = bitand(crc,hex2dec('ff'));
  highByte = bitshift(bitand(crc,hex2dec('ff00')),-8);
        %Fin funcion generadora CRC
        CRC_Bytes = [lowByte, highByte]; %Los bytes de CRC son guardados.
end
```

Apéndice B. Función Comparadora CRC

```
function CRC_Check = checkCRC(CRC_Generado, CRC_Recibido)
%La funcion se encarga de comparar el CRC Generado con el CRC Generado con
%modbus, estos deben ser iguales para que la comunicacion sea exitosa.

% Inputs:
% CRC_Generado,   Es el CRC hecho en el bloque anterior
% CRC_Recibido,   Es el CRC recibido a traves de la solicitud Modbus
%
% Outputs:
% CRC_Check,  Regresa 0 o 1 segun si los CRC generado y recibido son iguales


CRC_Check = (CRC_Generado(1) == CRC_Recibido(length(CRC_Recibido)-1)) && (
    CRC_Generado(2) == CRC_Recibido(length(CRC_Recibido)));
% Compara el primer byte de CRC generado con el primer byte del CRC modbus
% en la penultima posicion y Compara el segundo byte de CRC generado con
% el segundo byte del CRC modbus en la ultima posicion
end
```

Apéndice C. Función Envío

```
function [buffer,numBytes,dir,SET] = fcn_ADC(mensaje_modbus,CHVarios,CH5,CH6
   ,CH7,CH8)
%La funcion prepara la respuesta modbus a partir de la lectura de sus
%canales ADC, realiza el CRC respectivo, y envia los datos de vuelta.
% Inputs:
% mensaje_modbus, es el mensaje de solicitud realizado y recibido
% CHVarios,       es la lectura de los canales 1—4, es un MuxADC
% CH5,            es la lectura del canal 5
% CH6,            es la lectura del canal 6
% CH7,            es la lectura del canal 7
% CH8,            es la lectura del canal 8
%Outputs:
%buffer,         es el array respuesta, con el CRC incluido
%numBytes,       es el numero de bytes a transmitir
%dir,            es la direccion a la que quiero escribir
%SET,            es el valor que quiero escribir en la direccion especif.
%Parameters
 slave_ID = 10; %Es el ID del esclavo, de la dSPACE.
%Separa mux
    CH1 = CHVarios(1); %Lectura canal 1
```

```
20      CH2 = CHVarios(2); %Lectura canal 2
21      CH3 = CHVarios(3); %Lectura canal 3
22      CH4 = CHVarios(4); %Lectura canal 4
23  %Data registers
24   data = zeros(20,1); %Se inicializa la variable
25      %La funcion preparar envio se hace para completar la palabra
26      %hexadecimal #0000, a partir de los datos ingresados.
27      [ADC1,ADC2] = Preparar_envio(CH1);
28      [ADC3,ADC4] = Preparar_envio(CH2);
29      [ADC5,ADC6] = Preparar_envio(CH3);
30      [ADC7,ADC8] = Preparar_envio(CH4);
31      [ADC9,ADC10] = Preparar_envio(CH5);
32      [ADC11,ADC12] = Preparar_envio(CH6);
33      [ADC13,ADC14] = Preparar_envio(CH7);
34      [ADC15,ADC16] = Preparar_envio(CH8);
35      %Los registros del 17 al 20, pueden ser utilizados a futuro si se
36      %quiere enviar otro valor, en este experimento se decide enviar 0,
37      %aunque se deja un ejemplo si se quiere enviar el maximo o el minimo.
38      %maximo = max([CH1 CH2 CH3 CH4 CH5 CH6 CH7 CH8]);
39      %minimo = min([CH1 CH2 CH3 CH4 CH5 CH6 CH7 CH8]);
```

```
40      %[ADC17,ADC18] = Preparar_envio(maximo);
41      %[ADC19,ADC20] = Preparar_envio(minimo);
42      [ADC17,ADC18] = Preparar_envio(0);
43      [ADC19,ADC20] = Preparar_envio(0);
44      %Se llena el vector de datos
45  data(1:20) = [ADC1 ADC2 ADC3 ADC4 ADC5 ADC6 ADC7 ADC8 ADC9 ADC10 ADC11
        ADC12 ADC13 ADC14 ADC15 ADC16 ADC17 ADC18 ADC19 ADC20];
46      %Esta porcion del codigo es la encargada de realizar la respuesta
47      %modbus a enviar desde la dspace a node red.
48  buffer = uint32(zeros(25,1)); %Inicializa el buffer
49      dir = 0; %Inicializa la direccion de escritura
50      SET = 0; %Inicializa el valor de escritura
51      %Se extraen los valores principales de modbus
52  ID = mensaje_modbus(1); %ID, direccion del esclavo con el que se quiere
        comunicar
53  fcn = mensaje_modbus(2); %Codigo de funcion que se solicita hacer
54  addr = mensaje_modbus(4); %Registro desde el que debe leer
55  addr_range = mensaje_modbus(6)*2; %Cantidad de registros a leer
56      %Generacion completa del mensaje modbus
57  if ID == slave_ID %Se verifica la direccion de la dspace
```

```
58        buffer(1) = ID; %Posicion 1 de la rta es el id
59  buffer(2) = fcn; %Posicion 2 de la rta es la funcion
60        if fcn == 6 %Si se quiere setear un valor en el registro (DAC)
61            dir = Preparar_recepcion(mensaje_modbus(3),mensaje_modbus(4));
62            %Posicion 3 y 4 de modbus es la direccion que se quiere escibir
63            [dir1,dir2] = Preparar_envio(dir);
64            SET = Preparar_recepcion(mensaje_modbus(5),mensaje_modbus(6));
65            %Posicion 3 y 4 de modbus es el numero que se quiere escibir
66            [SETA,SETB] = Preparar_envio(SET);
67            buffer(3) = dir1; %Posicion 3 de la rta es la direccion1
68            buffer(4) = dir2; %Posicion 4 de la rta es la direccion1
69            buffer(5) = SETA; %Posicion 5 de la rta es el valor1
70            buffer(6) = SETB; %Posicion 6 de la rta es el valor2
71            %Se a ade el CRC a la rta modbus
72            N = 6; %Realiza el CRC desde la posicion 1 hasta la 6
73            crc = uint32(hex2dec('ffff'));
74            polynomial = hex2dec('a001');
75            for i = 1:N
76                crc = bitxor(crc,buffer(i));
77                for j = 1:8
```

```
78                  if bitand(crc,1)
79                      crc = bitshift(crc,-1);
80                      crc = bitxor(crc,polynomial);
81                  else
82                      crc = bitshift(crc,-1);
83                  end
84              end
85          end
86          lowByte = bitand(crc,hex2dec('ff'));
87          highByte = bitshift(bitand(crc,hex2dec('ff00')),-8);
88          %Se a ade el CRC al buffer
89          buffer(7) = lowByte; %Posicion 7 de la rta es el CRC1
90          buffer(8) = highByte; %Posicion 8 de la rta es el CRC1
91          numBytes = 25; %Se van a enviar un vector de 25 posiciones,
92          %donde de la posicion 9 a la 25 se llena de ceros.
93      elseif fcn == 3 %Si se quieren leer los registros (ADC)
94          buffer(3) = addr_range; %Posicion 3 de la rta es la cantidad de
                registros
95          %El siguiente for llena desde la 4ta posicion hasta la 23 de
96          %los datos que esta leyendo de los ADC.
```

```
97      for j=1:addr_range
98       buffer(3+j) = data(addr+j-1);
99            end
100           %Se a ade el CRC a la rta modbus
101           N = addr_range+3; %Realiza el CRC hasta la posicion 23
102           crc = uint32(hex2dec('ffff'));
103           polynomial = hex2dec('a001');
104           for i = 1:N
105               crc = bitxor(crc,buffer(i));
106               for j = 1:8
107                   if bitand(crc,1)
108                       crc = bitshift(crc,-1);
109                       crc = bitxor(crc,polynomial);
110                   else
111                       crc = bitshift(crc,-1);
112                   end
113               end
114           end
115           lowByte = bitand(crc,hex2dec('ff'));
116           highByte = bitshift(bitand(crc,hex2dec('ff00')),-8);
```

```
117             %Se a ade el CRC al buffer
118             buffer(addr_range+4) = lowByte;
119             buffer(addr_range+5) = highByte;
120             numBytes = double(addr_range)+5; %Se envia un vector de 25 pos.
121         else
122             numBytes = 0; %Si se llama otro codigo de funcion,
123             %que no haga envio.
124         end
125 else
126  numBytes = 0; %Si la direccion del esclavo no es la que le
127         %corresponde a la dspace.
128 end
```

Apéndice D. Preparar Envío

```
function [ADC1,ADC2] = Preparar_envio(ADC)
    %La funcion se encarga de armar la rta en hexadecimal, para finalmente
    %ser enviada en decimal.
    %Inputes:
    % ADC,      es el valor en decimal leido en los canales de la dspace
    %Outputs:
    % ADC1,     son los primeros 2 digitos hexadecimales en decimal.
    % ADC2,     son los ultimos 2 digitos hexadecimales en decimal.
    ADC_total = ADC*10000; %Se multiplica para obtener 3 decimales
    ADC_hex= dec2hex(floor(ADC_total));% la funcion floor se anade para
    %evitar problemas con notacion cientifica en algunos numeros
    %Los if en cascada se encarga de concatenar ceros a la palabra
       hexadecimal.
    if length(ADC_hex)<2
        ADC_hex = [dec2hex(0),ADC_hex];
    end
    if length(ADC_hex)<3
        ADC_hex = [dec2hex(0),ADC_hex];
    end
    if length(ADC_hex)<4
```

```
20          ADC_hex = [dec2hex(0),ADC_hex];
21      end
22      %Se arman los numeros decimales
23      ADC1 = hex2dec(ADC_hex(1:2));
24      ADC2 = hex2dec(ADC_hex(3:4));
```

Apéndice E. Preparar Recepción

```
function [ADC] = Preparar_recepcion(ADC1,ADC2)
    %La funcion se encarga de concatenar el valor obtenido desde la
    %solicitud modbus.
    %Inputs:
    % ADC1,     es el valor en decimal a concatenar
    % ADC2,     es el valor en decimal a concatenar
    %Outputs:
    % ADC,      es el numero hexadecimal convertido en decimal
    %max = 65536;% Valor maximo con 2 bits hexadecimales
    a = dec2hex(ADC1);%Convierte a hexadecimal el primer byte
    b = dec2hex(ADC2);%Convierte a hexadecimal el segundo byte
    if length(b)<2
    b = [dec2hex(0),b];
    end
    c = [a,b];%Forma el numero hexadecimal de 2 bytes
    ADC = hex2dec(c);%Convierte a decimal el valor
```

Apéndice F. DAC Escritura

```
function [DAC1,DAC2,DAC3,DAC4,DAC5,DAC6,DAC7,DAC8]=operacion(direccion,
   valor)
    %La funcion se encarga de relacionar las direcciones con el valor
    %recibido de modbus.
    %Inputs:
    % direccion,    Es la direccion donde se va a asignar el valor
    % valor,        Es el valor asignado a una direccion determinada
    %Outputs:
    % DAC1,         La salida hacia el canal 1 DAC
    % DAC2,         La salida hacia el canal 2 DAC
    % DAC3,         La salida hacia el canal 3 DAC
    % DAC4,         La salida hacia el canal 4 DAC
    % DAC5,         La salida hacia el canal 5 DAC
    % DAC6,         La salida hacia el canal 6 DAC
    % DAC7,         La salida hacia el canal 7 DAC
    % DAC8,         La salida hacia el canal 8 DAC
    %Se crean 8 variables globales, para guardar los valores previos.
    global DACH1;
    global DACH2;
    global DACH3;
```

```
    global DACH4;
    global DACH5;
    global DACH6;
    global DACH7;
    global DACH8;
    %Los if se encargan de comparar el valor recibido con la direccion.
    % 101—DAC1, 102—DAC2, 103—DAC3, 104—DAC4, 105—DAC5, 106—DAC6,
    % 107—DAC7, 108—DAC8.
    if (direccion == 101)
        DACH1 = valor;
    elseif (direccion == 201)
        DACH2 = valor;
    elseif (direccion == 301)
        DACH3 = valor;
    elseif (direccion == 401)
        DACH4 = valor;
    elseif (direccion == 501)
        DACH5 = valor;
    elseif (direccion == 601)
        DACH6 = valor;
```

```
    elseif (direccion == 701)
        DACH7 = valor;
    elseif (direccion == 801)
        DACH8 = valor;
    end
    %Las variables globales son asignadas a las variables de salida
    DAC1 = DACH1;
    DAC2 = DACH2;
    DAC3 = DACH3;
    DAC4 = DACH4;
    DAC5 = DACH5;
    DAC6 = DACH6;
    DAC7 = DACH7;
    DAC8 = DACH8;
end
```

Apéndice G. Función Salidas

```
var msg1;
var msg2;
var msg3;
var msg4;
var msg5;
var msg6;
var msg7;
var msg8;
var msg9;
var msg10;
msg1 = { payload: msg.payload[0]};
msg2 = { payload: msg.payload[1]};
msg3 = { payload: msg.payload[2]};
msg4 = { payload: msg.payload[3]};
msg5 = { payload: msg.payload[4]};
msg6 = { payload: msg.payload[5]};
msg7 = { payload: msg.payload[6]};
msg8 = { payload: msg.payload[7]};
// msg9 = { payload: msg.payload[8]};
// msg10 = { payload: msg.payload[9]};

return [msg1, msg2, msg3, msg4, msg5, msg6, msg7, msg8] // msg9, msg10]
```

Apéndice H. Función Trans_Ignition

```
newObject={
    tagValue: msg.payload
};

return {payload:newObject};
```

Apéndice I. Función Ignition a Valor

```
var msg1;

msg1 = {payload:msg.payload.ignitionResult.value}

return msg1;
```

Apéndice J. Tablero IoT

Figura 71

Plano del tablero IIoT. Hoja de título

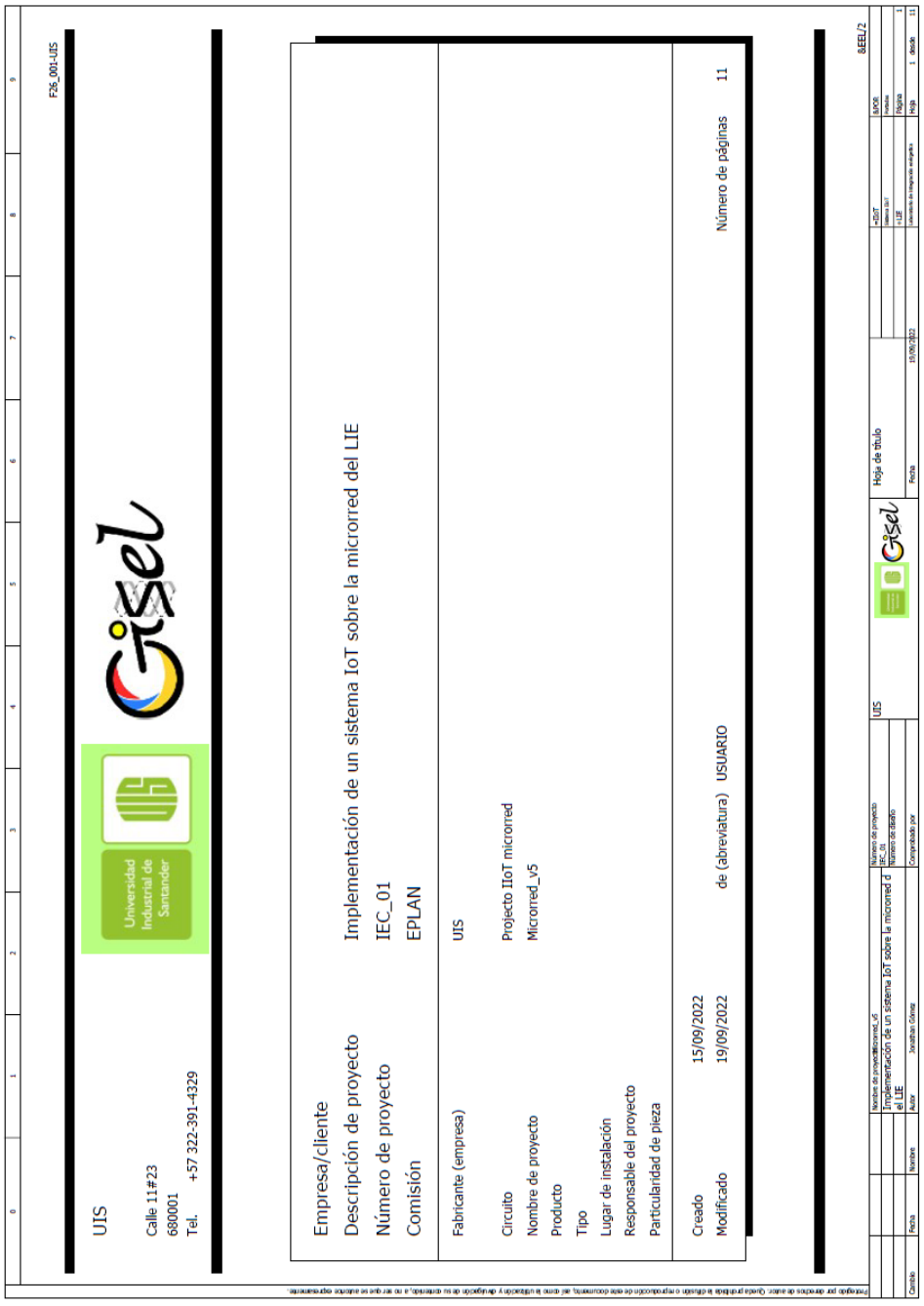


Figura 72

Plano del tablero IIoT. Alimentación 110V

Figura 73

Plano del tablero IIoT. Protección alimentación

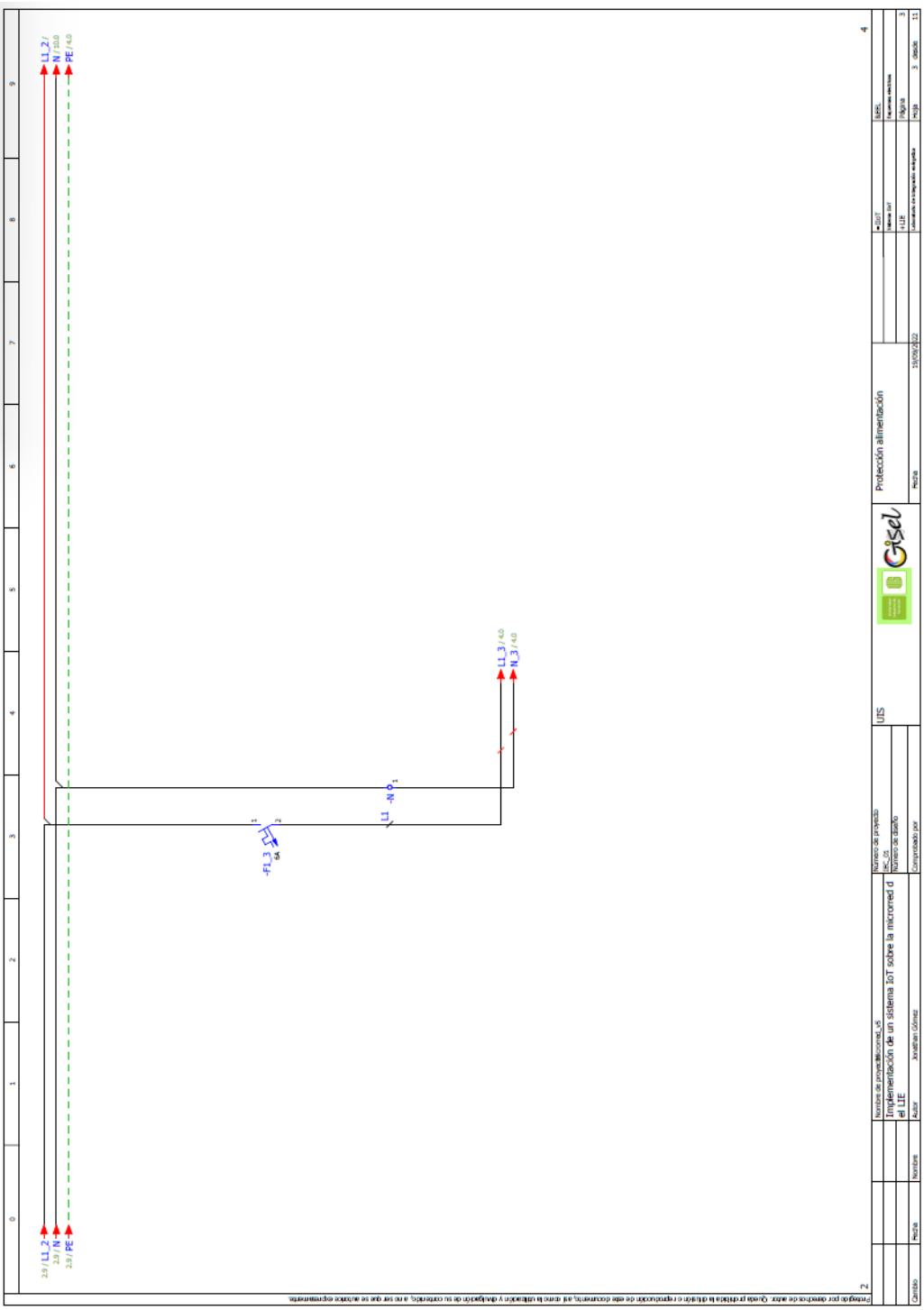


Figura 74

Plano del tablero IIoT. Distribución 110

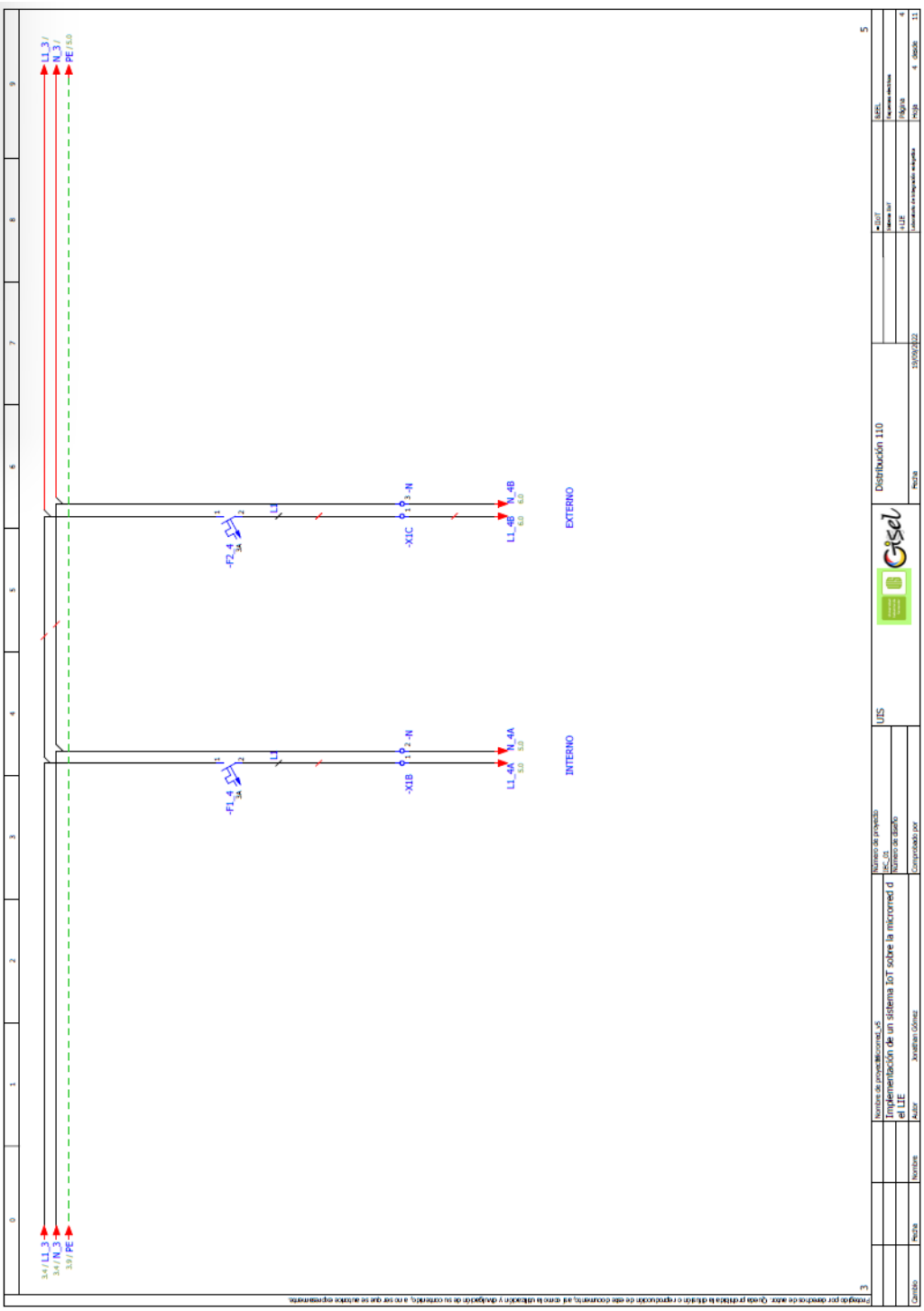


Figura 75

Plano del tablero IIoT. Distribución 110 internos

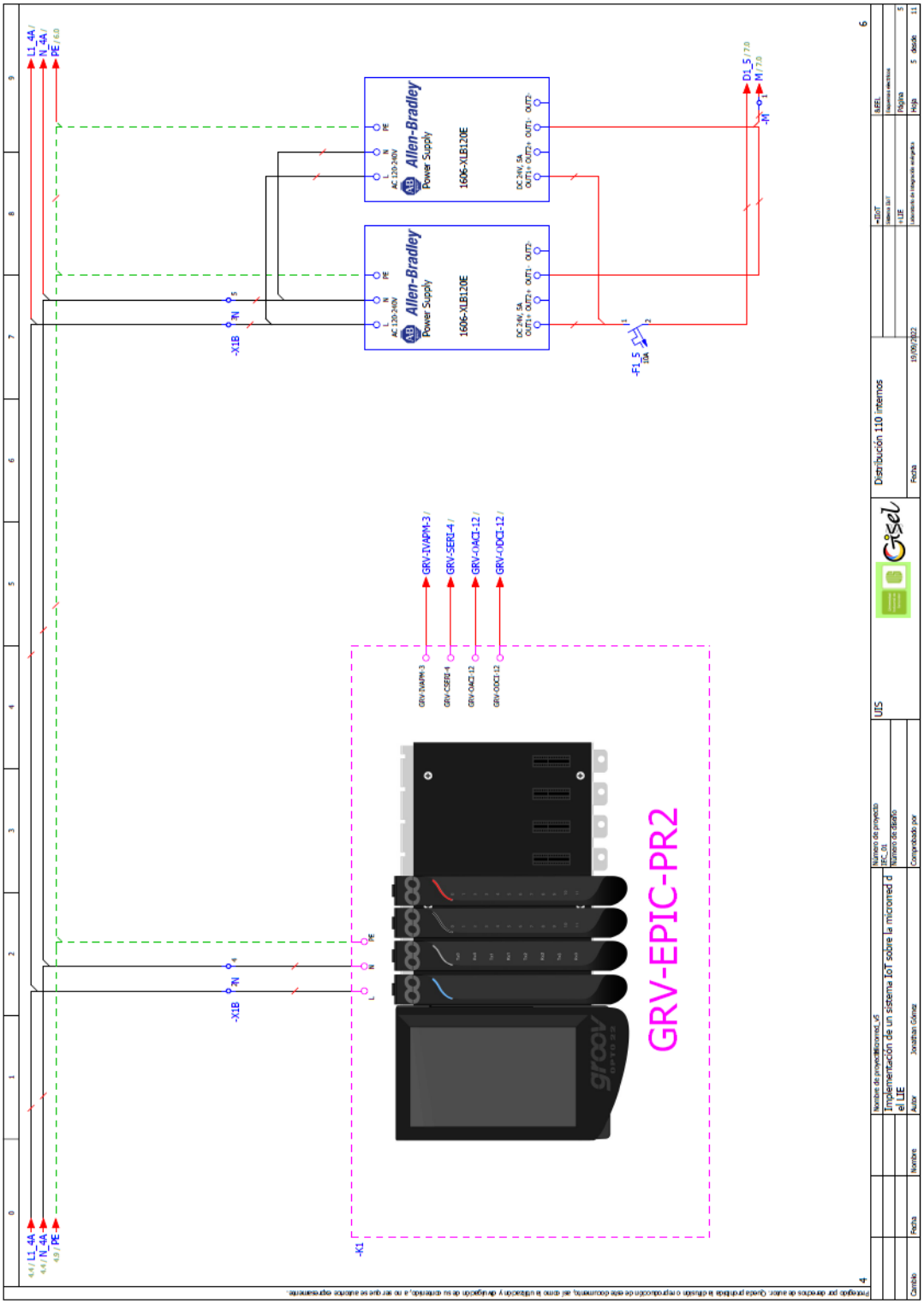


Figura 76

Plano del tablero IIoT. Distribución 110 externos

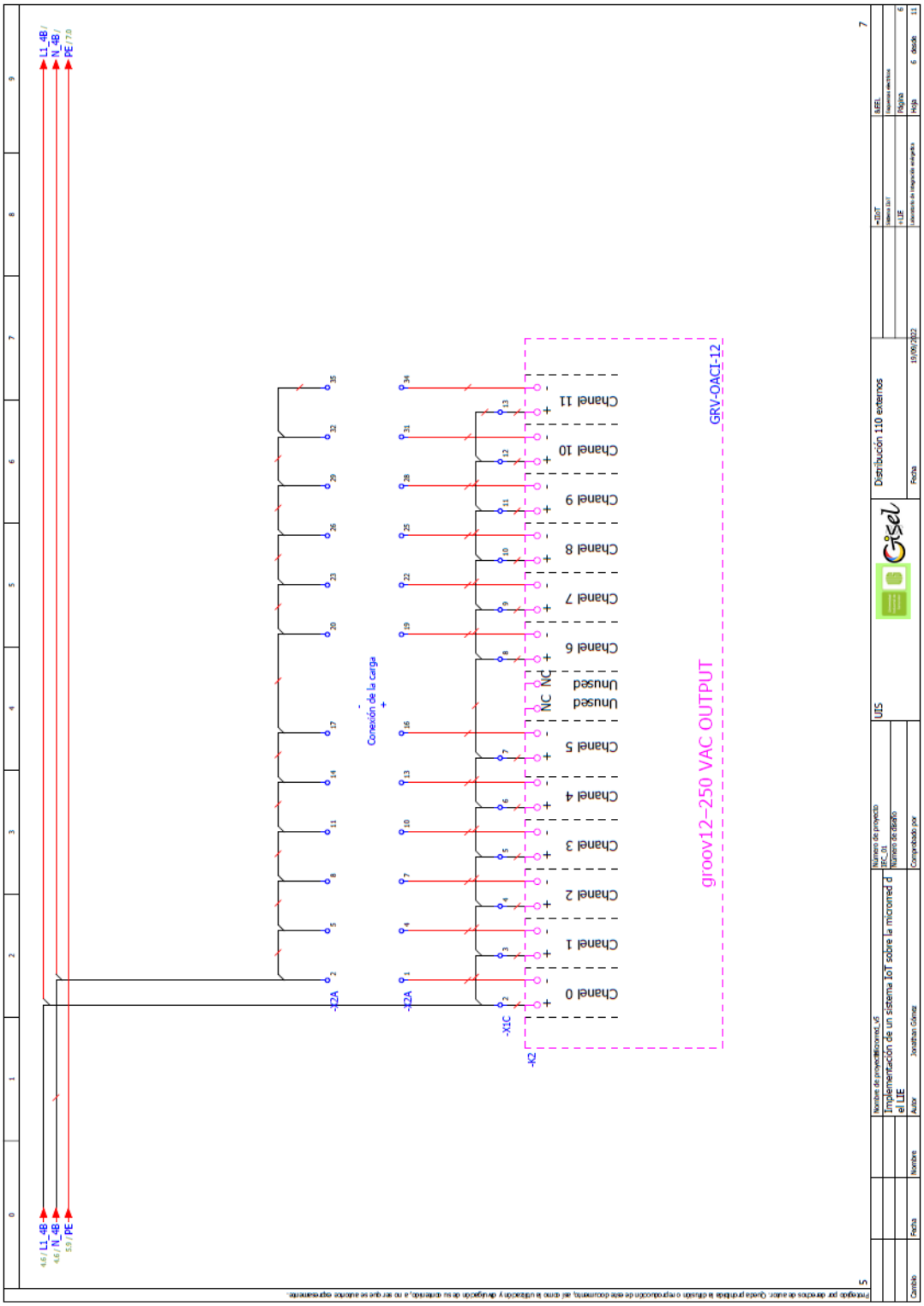


Figura 77

Plano del tablero IIoT. Distribución DC

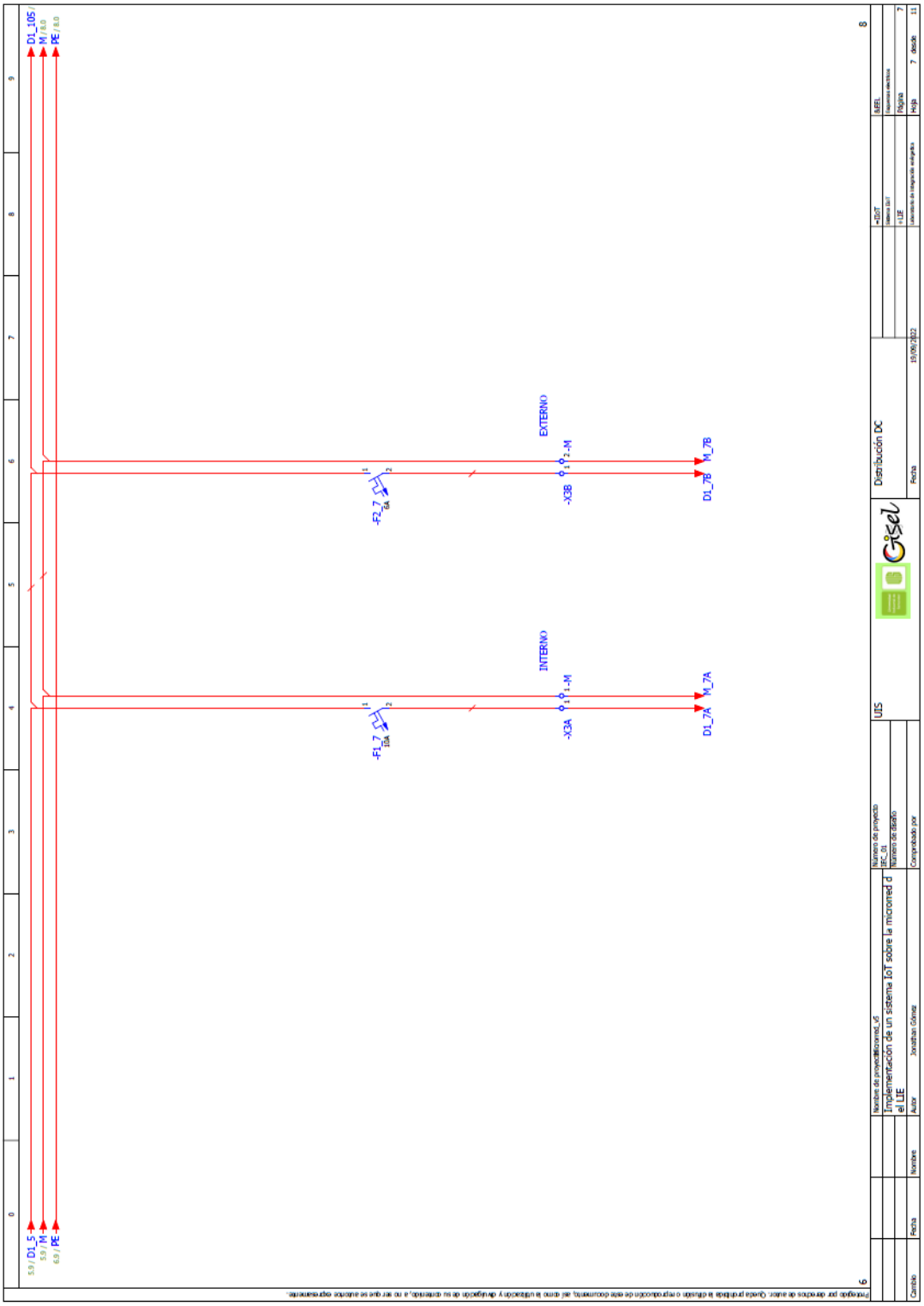


Figura 78

Plano del tablero IIoT. Distribución DC internos

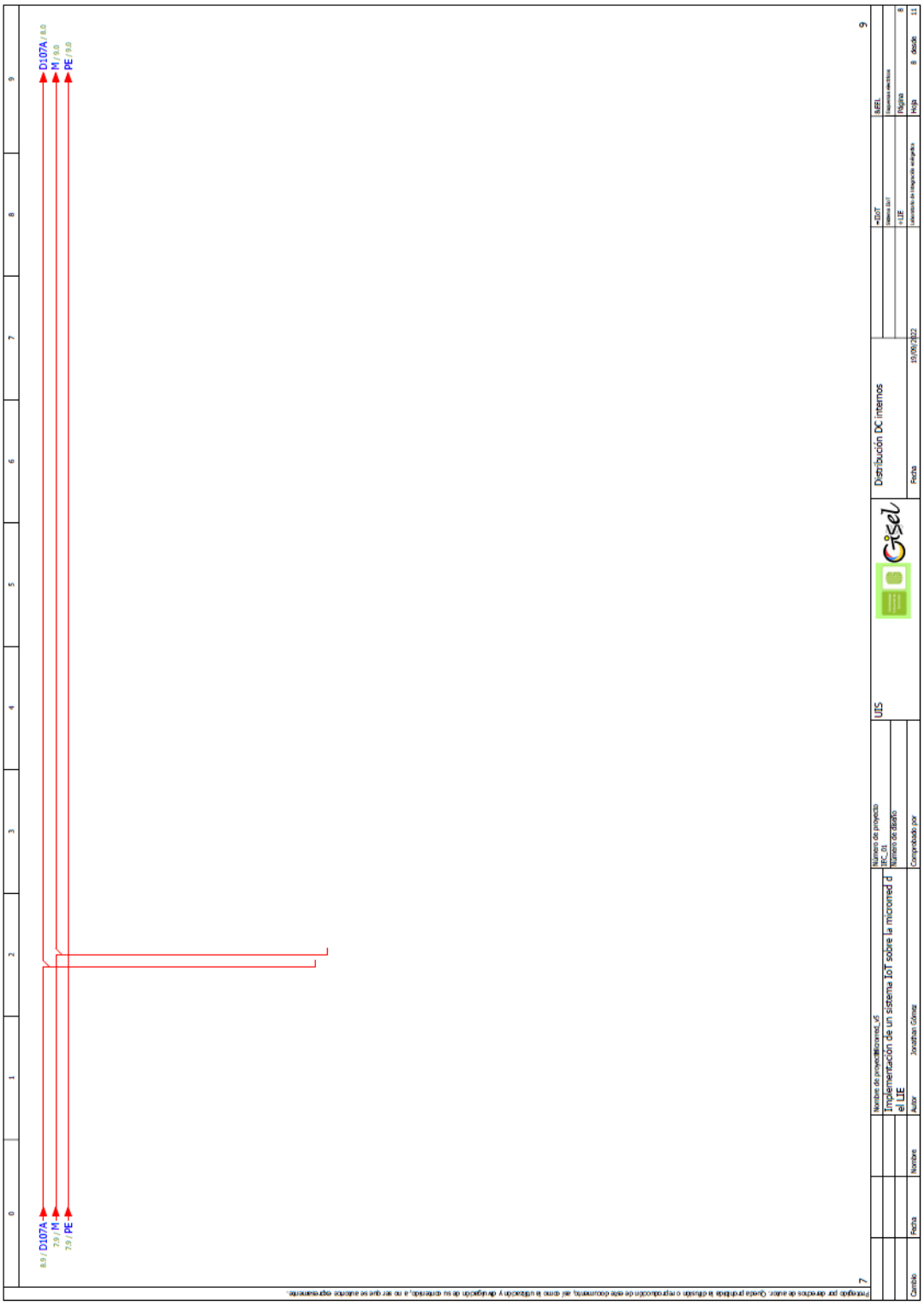


Figura 79

Plano del tablero IIoT. Distribución DC externos

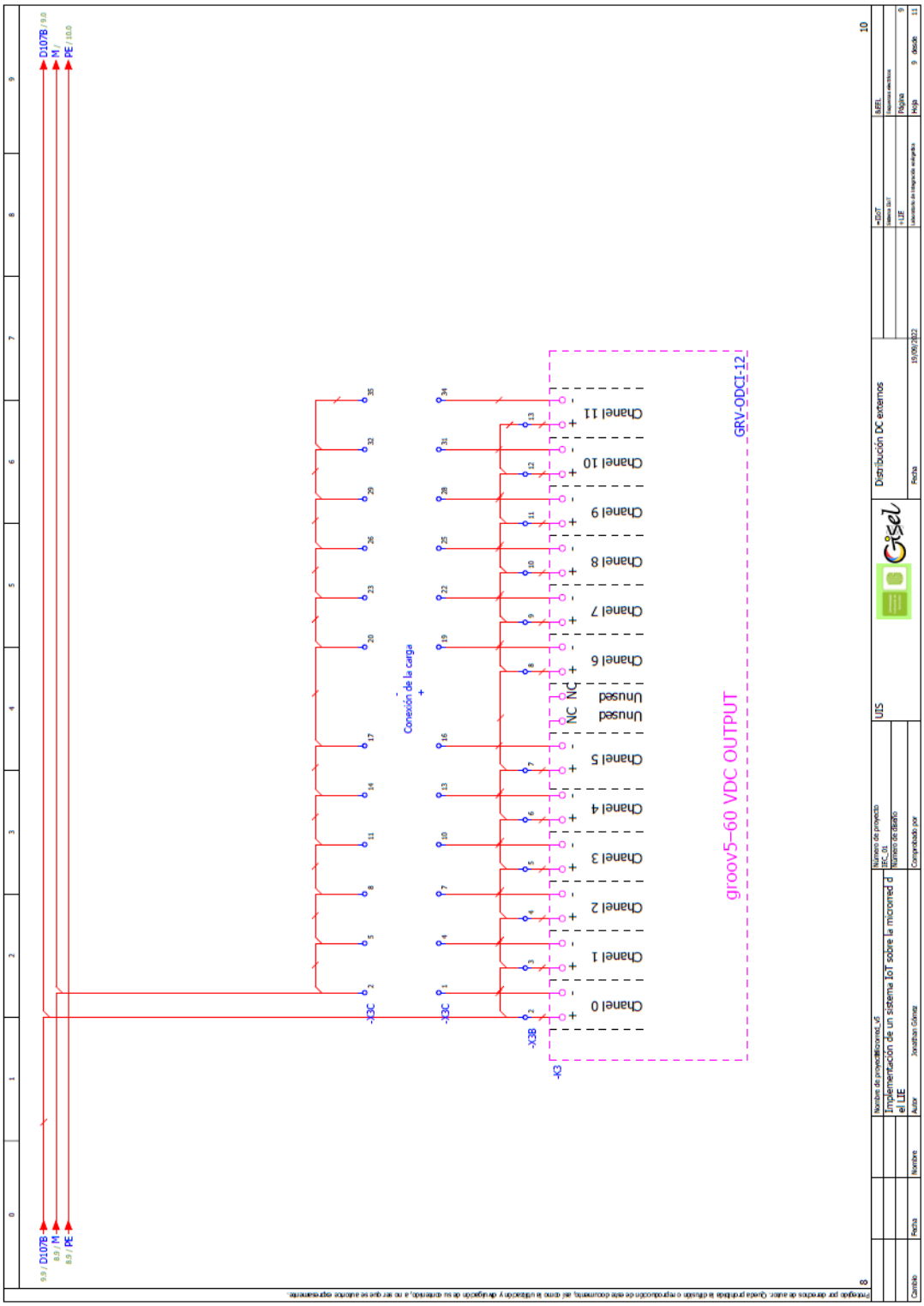


Figura 80

Plano del tablero IIoT. Módulo de energía

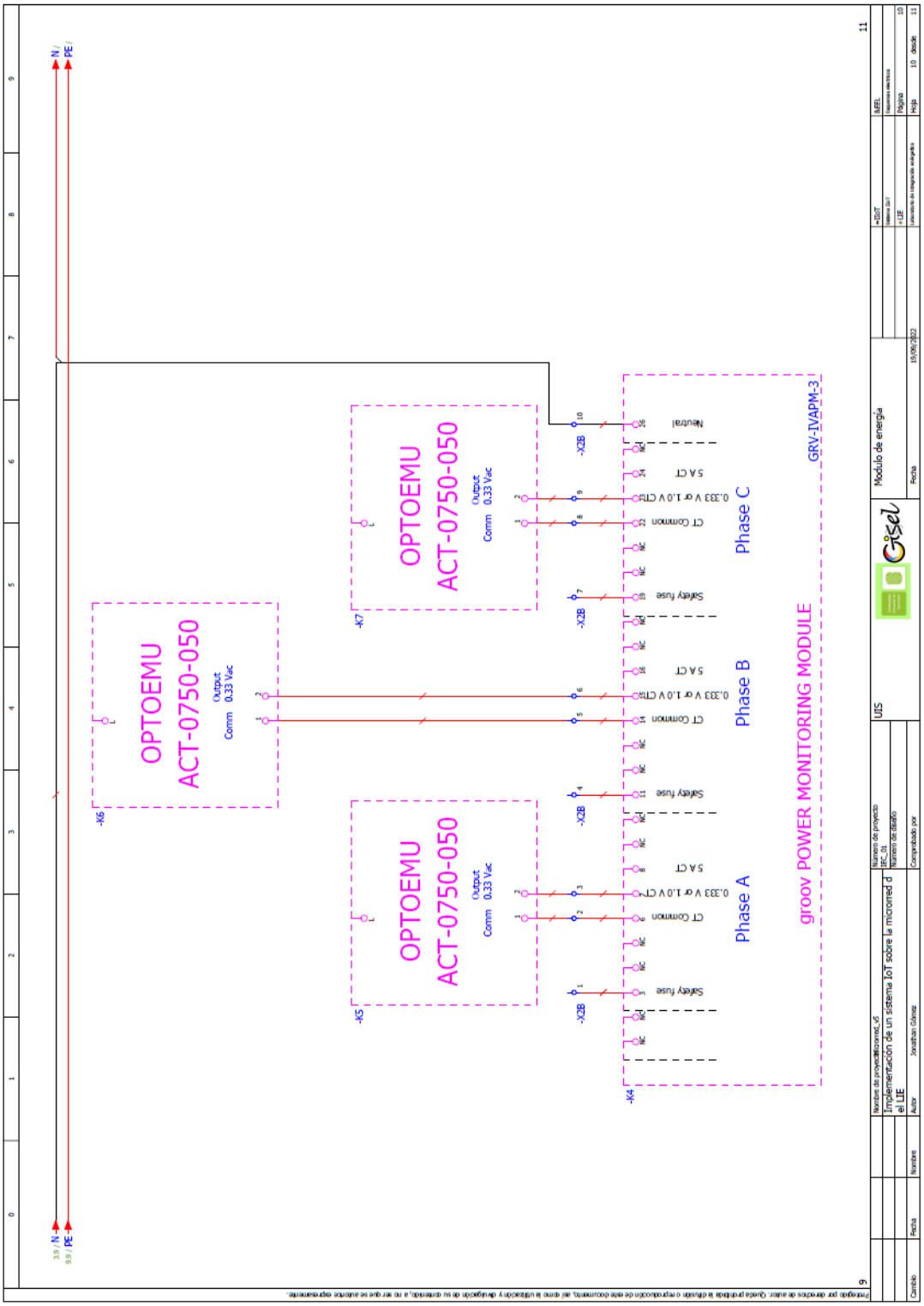


Figura 81

Plano del tablero IIoT. Módulo serial