



**Automatización y Monitoreo en Tiempo Real de Equipos de Laboratorio en GIMEL  
Usando SCPI y PyVISA**

Omar David Vargas Bonett

Trabajo de grado presentado para optar al título de Ingeniero Electrónico

Modalidad de Práctica

Trabajo de Grado

Seleccione tipo de orientador(es)

José Robinson Ortiz Castrillón, Doctor (PhD) Electrónica y Computación.

Universidad de Antioquia

Facultad de Ingeniería

Ingeniería Electrónica

Medellín, Antioquia, Colombia

2025

Cita	Omar David Vargas Bonett [1]
<b>Referencia</b>	[1] Omar David Vargas Bonett, “Automatización y Monitoreo en Tiempo Real de Equipos de Laboratorio en GIMEL Usando SCPI y PyVISA Trabajo de grado profesional, Ingeniería Electrónica, Universidad de Antioquia Medellín, Antioquia, Colombia, 2025.
Estilo IEEE (2020)	



Grupo de Investigación Manejo Eficiente de la Energía (GIMEL).

Sede de Investigación Universitaria (SIU).



Centro de Documentación Ingeniería (CENDOI)

**Repositorio Institucional:** <http://bibliotecadigital.udea.edu.co>

Universidad de Antioquia - [www.udea.edu.co](http://www.udea.edu.co)

El contenido de esta obra corresponde al derecho de expresión de los autores y no compromete el pensamiento institucional de la Universidad de Antioquia ni desata su responsabilidad frente a terceros. Los autores asumen la responsabilidad por los derechos de autor y conexos.

## **Dedicatoria**

A mi hermano menor,  
que durante mi práctica académica partió de este mundo, pero nunca de mi corazón.  
Tu compañía, apoyo incondicional y fe en mis sueños fueron mi fuerza para alcanzar cada meta.  
Sé que, allá donde estás, sigues guiando mis pasos con amor y orgullo.

A mis amigos,  
Que apoyaron a un muchacho de pueblo en una ciudad desconocida a sobrevivir y me  
acompañaron en los momentos más difíciles.

## **Agradecimientos**

Quiero expresar mi más sincero agradecimiento a todas las personas que contribuyeron a la  
realización de esta práctica académica y al desarrollo de este trabajo.  
Al profesor Robinson, por su dedicación y compromiso al acompañarme durante la realización de  
esta práctica, incluso revisando el trabajo durante sus vacaciones. Su orientación fue fundamental  
para alcanzar los objetivos planteados.  
Al profesor Nicolás, por proporcionarme las herramientas necesarias y compartir ideas valiosas  
que impulsaron el desarrollo de esta práctica. Su apoyo constante marcó una gran diferencia en  
mi aprendizaje.  
Al grupo GIMEL, por brindarme un espacio de trabajo propicio y colaborativo. Su disposición y  
apoyo me permitieron llevar a cabo este proyecto en un entorno enriquecedor.

## TABLA DE CONTENIDO

RESUMEN .....	8
ABSTRACT .....	9
I. INTRODUCCIÓN .....	10
II. OBJETIVOS.....	12
A. Objetivo general .....	12
B. Objetivos específico .....	12
III. MARCO TEÓRICO .....	13
Protocolo VISA .....	13
El estándar SCPI.....	13
La librería Pyvisa .....	14
Fuentes de alimentación programables .....	15
Simulador de red Itech IT7900.....	16
Fuente de alimentación DC programable bidireccional Itech IT6000C .....	16
Cargas Electrónicas programables .....	17
Carga electrónica programable Itech IT8600 .....	18
IV. METODOLOGÍA .....	19
Etapa 1: Diseño del Sistema de Comunicación.....	19
Etapa 2: Desarrollo de Scripts de Automatización.....	20
Etapa 3: Implementación del sistema de monitoreo.....	22
Etapa 4: Validación del Sistema.....	23
Etapa 5: Documentación .....	24
V. ANÁLISIS DE RESULTADOS .....	25
Diseño del Sistema de Comunicación.....	25
Desarrollo de la Librería de Automatización .....	27

Implementación del Sistema de Monitoreo.....	28
Validación del Sistema.....	29
Documentación.....	31
VI. CONCLUSIONES .....	32
VII. TRABAJOS FUTUROS .....	33
REFERENCIAS .....	35
ANEXOS.....	37
1. Requisitos Previos .....	37
Hardware:.....	37
Software: .....	37
2. Instalación del Sistema.....	37
3. Uso del Sistema.....	37

## LISTA DE FIGURAS

Fig. 1. Emulador de red Itech IT7900.	17
Fig. 2. Fuente de alimentación DC programable bidireccional Itech IT6000C.	18
Fig. 3. Carga electrónica programable Itech IT8600.	19
Fig. 4. Topología implementada en la red LAN del laboratorio de GIMEL.	20
Fig. 5. Diagrama de clases implementado en la librería para controlar los instrumentos ITECH.	22
Fig. 6. Emuladores conectados a la red local en el laboratorio GIMEL	27
Fig. 7. Pruebas de comunicación entre el controlador y los instrumentos.	27
Fig. 8. Código de ejemplo de una implementación de la clase Itech7900.	28
Fig. 9. Estructura del proyecto de automatización con SCPI y Pyvisa .	29
Fig.10. Comandos enviados durante una de las pruebas funcionales.	30
Fig.11. Interfaz gráfica mostrando los resultados de una configuración para una red 220V 60Hz.	31
Fig.12. Emulador de red Itech IT7900 funcionando con los parámetros para una red 220V 60Hz.	31
Fig.13. Recursos consumidos por el software durante las pruebas y alertas de desconexión.	32

## SIGLAS, ACRÓNIMOS Y ABREVIATURAS

<b>GIMEL</b>	Grupo de Manejo Eficiente de la Energía
<b>SCPI</b>	Standard Commands for Programmable Instruments
<b>SIU</b>	Sede de Investigación Universitaria
<b>LAN</b>	Local Area Network
<b>DC</b>	Direct current
<b>AC</b>	Alternating current
<b>VISA</b>	Virtual Instrument Software Architecture
<b>TCP</b>	Transmission Control Protocol
<b>USB</b>	Universal Serial Bus
<b>ASCII</b>	American Standard Code for Information Interchange
<b>UdeA</b>	Universidad de Antioquia

---

## RESUMEN

En este trabajo se diseñó e implementó un sistema de control y monitoreo en tiempo real para equipos de laboratorio en el grupo GIMEL de la Universidad de Antioquia. A través de comandos SCPI y la librería PyVISA, se desarrolló una herramienta que permite la configuración, operación y supervisión de cinco dispositivos de laboratorio a través de una red Ethernet. La metodología empleada incluyó el diseño de la red de comunicación, el desarrollo de una librería modular en Python, la creación de una interfaz gráfica de usuario y la validación exhaustiva del sistema mediante pruebas funcionales. El sistema implementado ofrece comunicación estable entre los dispositivos conectados, facilitando una visualización dinámica de datos críticos como voltajes y corrientes, junto con alertas visuales en tiempo real. Los resultados obtenidos demostraron su precisión, escalabilidad y robustez, validando su aplicabilidad para entornos de laboratorio complejos. Adicionalmente, la documentación técnica desarrollada facilita su replicación y futuras expansiones. Este proyecto representa un aporte significativo para la automatización y monitoreo en tiempo real de equipos de laboratorio y para la comunidad de Python, aportando una librería para el manejo de fuentes y cargas programables, sentando las bases para la integración de nuevas funcionalidades y dispositivos.

***Palabras clave* — Automatización de laboratorio, monitoreo en tiempo real, SCPI, PyVISA, interfaz gráfica, Ethernet, Fuentes y Cargas programables.**

---

**ABSTRACT**

In this work, a real-time control and monitoring system for laboratory equipment was designed and implemented for the GIMEL group at the University of Antioquia. Using SCPI commands and the PyVISA library, a tool was developed to enable the configuration, operation, and supervision of five laboratory devices through an Ethernet network. The methodology employed included designing the communication network, developing a modular library in Python, creating a graphical user interface, and thoroughly validating the system through functional testing. The implemented system provides stable communication between the connected devices, enabling dynamic visualization of critical data such as voltages and currents, along with real-time visual alerts. The results obtained demonstrated its accuracy, scalability, and robustness, validating its applicability in complex laboratory environments. Additionally, the technical documentation developed facilitates its replication and future expansions. This project represents a significant contribution to the automation and real-time monitoring of laboratory equipment and to the Python community, offering a library for managing programmable power supplies and loads, laying the groundwork for the integration of new functionalities and devices.

***Keywords* — Laboratory automation, real-time monitoring, SCPI, PyVISA, graphical interface, Ethernet, Programmable power supplies and loads.**

---

## I. INTRODUCCIÓN

La configuración manual y el monitoreo constante de equipos de laboratorio son tareas tediosas y propensas a errores, lo que limita la eficiencia y precisión de los experimentos. Estas limitaciones se agravan en entornos de investigación donde múltiples dispositivos deben ser controlados de manera simultánea, ya que la interacción manual con cada equipo consume tiempo valioso y puede introducir inconsistencias en los resultados debido a errores humanos. Además, la falta de integración entre los equipos dificulta el análisis automatizado de los datos, afectando la capacidad de los investigadores para identificar patrones y tomar decisiones basadas en evidencia. Para abordar esta problemática, el presente trabajo propone un sistema de automatización basado en comandos SCPI y la librería PyVISA, este sistema tiene como objetivo controlar de manera remota y eficiente cinco equipos de laboratorio pertenecientes al grupo de investigación GIMEL de la Universidad de Antioquia. Esta solución, implementada en el laboratorio de investigación 228 de la SIU, permite configurar, operar y monitorear en tiempo real emuladores de carga y fuentes de energía, optimizando así el proceso de pruebas y facilitando el análisis de datos.

El estándar SCPI es ampliamente utilizado en instrumentos de medición y control, ya que ofrece una interfaz común para la comunicación con dispositivos electrónicos programables. Este enfoque estandarizado facilita la interoperabilidad entre diferentes marcas y modelos de equipos, lo que resulta esencial en un laboratorio que emplea dispositivos de fabricantes variados, como emuladores de carga IT8615L, una fuente emuladora de energía DC IT6000C y un emulador AC IT7900. Por su parte, la librería PyVISA proporciona una herramienta flexible para interactuar con estos dispositivos a través de interfaces como Ethernet, USB o GPIB, simplificando el desarrollo de aplicaciones de control automatizado.

La importancia de automatizar equipos de laboratorio usando SCPI ha sido reconocida en la literatura científica [1]. Sin embargo, muchas implementaciones existentes presentan limitaciones significativas, como interfaces de usuario poco intuitivas o la ausencia de capacidades de monitoreo en tiempo real. Estas deficiencias restringen su aplicabilidad en escenarios donde es crucial obtener información inmediata sobre el estado de los dispositivos y los parámetros medidos.

---

El presente trabajo busco llenar este vacío al desarrollar un sistema modular y escalable, que no solo permitió controlar los equipos de manera efectiva, sino que también ofreció una visualización gráfica de los datos en tiempo real. Este enfoque mejora la experiencia del usuario al proporcionar una interfaz amigable que integra funciones avanzadas de monitoreo y análisis. Además, el sistema está diseñado para adaptarse a otros dispositivos que utilicen el estándar SCPI, lo que amplía su aplicabilidad en diferentes contextos de investigación y desarrollo.

Implementar esta solución en el laboratorio de investigación 228 de la SIU represento un avance significativo para el grupo GIMEL, ya que permitió centralizar el control de los dispositivos y facilito la realización de pruebas complejas que requieren la coordinación de múltiples equipos. Además, el desarrollo de esta herramienta automatizada tiene implicaciones más allá del laboratorio en cuestión. Al demostrar la viabilidad de una solución basada en tecnologías de código abierto como PyVISA, se abre la posibilidad de que otros laboratorios implementen sistemas similares, reduciendo los costos asociados con software propietario y promoviendo la colaboración entre instituciones académicas y centros de investigación [2], este trabajo permitió aportar una librería para la comunidad de desarrollo de Python, que podrá ser utilizada, retroalimentada y mejorada con equipos de trabajo o personas que trabajen en áreas similares o afrontes problemas en los que requieran automatizar sus instrumentos de forma remota.

En términos de resultados, el sistema desarrollado ha mostrado ser robusto y confiable, permitiendo a los investigadores interactuar con los dispositivos de manera efectiva y acceder a datos en tiempo real a través de una interfaz gráfica intuitiva. Esta capacidad no solo mejora la experiencia del usuario, sino que también facilita el análisis y la interpretación de los datos experimentales, contribuyendo a una toma de decisiones más informada.

## II. OBJETIVOS

### *A. Objetivo general*

Implementar la plataforma para el control y monitoreo en tiempo real de cinco equipos de laboratorio (tres emuladores de carga IT8615L, una fuente emuladora de energía DC IT6000C y un emulador AC IT7900), utilizando una interfaz con comandos SCPI y la librería PyVISA, garantizando una comunicación eficiente a través de una red Ethernet para mejorar las pruebas y la adquisición de datos en el entorno de laboratorio.

### *B. Objetivos específico*

- Comunicar los equipos del laboratorio y el sistema de control central mediante una red LAN Ethernet, garantizando una conexión estable y eficiente que permita la automatización y el monitoreo en tiempo real de los equipos.
- Automatizar la configuración y operación de los equipos mediante Scripts en Python, reduciendo el tiempo de configuración manual y mejorando las pruebas en el laboratorio.
- Monitorear variables y datos en tiempo real de los emuladores de carga, la fuente DC y el emulador AC, a través de la red Ethernet y visualizarlos en una interfaz de monitoreo asegurando el acceso inmediato de los datos críticos.
- Verificar las funcionalidades del sistema ejecutando secuencias de pruebas automáticas y comparando los resultados con las expectativas predefinidas, confirmando que el sistema opere de manera efectiva y precisa.

---

### III. MARCO TEÓRICO

En sistemas de automatización y monitoreo en tiempo real, como los que se implementan en laboratorios, Ethernet TCP/IP es particularmente útil debido a su confiabilidad, velocidad y facilidad de integración con software de monitoreo como SCADA o sistemas basados en SCPI (Standard Commands for Programmable Instruments) [3]. Estas soluciones permiten a los operadores monitorear y controlar múltiples dispositivos desde un punto central, facilitando la gestión y análisis de grandes cantidades de datos en tiempo real.

Una de las herramientas clave para lograr este tipo de automatización es el uso de VISA (Virtual Instrument Software Architecture) [4], un estándar que facilita la comunicación entre el software de control y los instrumentos de laboratorio, independientemente del protocolo físico. VISA permite que diferentes dispositivos se conecten y sean controlados a través de una única interfaz de software, simplificando la programación y el monitoreo en tiempo real.

#### ***Protocolo VISA***

El estándar VISA (Virtual Instrument Software Architecture) es uno de los protocolos de comunicación más utilizados para el control de instrumentación mediante computadoras. Es empleado por grandes fabricantes del sector, como *Keysight Technologies*, *National Instruments*, *ITECH*. Este estándar actúa como un puente entre los programas de control de procesos de medición, desarrollados en lenguajes de alto nivel como *C++* o *Python*, y los instrumentos de medición. Su función principal es adaptar el formato lógico-digital de la transmisión al estándar específico de conexión utilizado entre la computadora y el instrumento.

La amplia adopción de VISA se debe, en gran medida, a su compatibilidad con una amplia variedad de interfaces físicas de comunicación. Estas incluyen interfaces específicas para instrumentación, como GPIB, VXI o PXI, así como interfaces de uso general, como USB o TCP/IP. Esta versatilidad ha convertido a VISA en un estándar clave en el control y automatización de equipos de laboratorio [5].

#### ***El estándar SCPI***

---

El estándar SCPI (Standard Commands for Programmable Instruments) establece un conjunto de comandos estandarizados para el control remoto de instrumentos de medición y prueba. Incluido en el IEEE 488.2-1987[6], SCPI (*American Standard Code for Information Interchange*) define la sintaxis y un subconjunto básico de comandos que deben interpretar los instrumentos, permitiendo así su interoperabilidad independientemente del fabricante.

Inicialmente, SCPI fue diseñado para ser utilizado con el protocolo GPIB (General Purpose Interface Bus). Sin embargo, con el tiempo, se adaptó para soportar otros buses de comunicación, como USB y TCP/IP, haciéndolo compatible con tecnologías modernas. Los comandos SCPI son cadenas de texto codificadas en ASCII, lo que facilita su interpretación por parte de los usuarios. Además, el estándar define cómo deben formatearse los datos devueltos por los instrumentos, permitiendo su representación en formatos binarios (compactos) o ASCII (más legibles).

Así, mientras que VISA gestiona la comunicación entre el computador y el instrumento, SCPI estandariza la forma en que se emiten y reciben los comandos, proporcionando un lenguaje común para el control de equipos de diferentes fabricantes.

### ***La librería Pyvisa***

PyVISA es una librería de Python diseñada para interactuar con instrumentos de medición y prueba que cumplen con el estándar VISA (Virtual Instrument Software Architecture). Esta librería actúa como un intermediario que permite a los usuarios enviar comandos y recibir respuestas de dispositivos conectados mediante protocolos como GPIB, USB, Ethernet (TCP/IP) y RS-232.

- Funciona en sistemas operativos Windows, macOS y Linux, lo que la hace versátil para diversos entornos de desarrollo.
- Admite interfaces físicas como USB, TCP/IP, GPIB y RS-232, lo que facilita su integración con una amplia variedad de instrumentos.
- Proporciona una API intuitiva que permite enviar comandos SCPI con facilidad, así como manejar respuestas en formatos binarios o ASCII.
- PyVISA aprovecha las bibliotecas VISA subyacentes, como NI-VISA (National Instruments) o Keysight VISA, para gestionar la comunicación a nivel físico y lógico.

### ***Fuentes de alimentación programables***

Una fuente de alimentación programable es un dispositivo electrónico diseñado para suministrar energía eléctrica a equipos o sistemas bajo condiciones controladas y configurables. Estas fuentes se usan en aplicaciones de investigación, desarrollo, pruebas y producción por su capacidad para controlar los parámetros de salida, como el voltaje, la corriente y la potencia. Las fuentes programables permiten ajustar los parámetros de salida a través de interfaces locales, como paneles táctiles o botones, o remotas, mediante protocolos de comunicación como SCPI, USB, Ethernet o GPIB. También destacan por ofrecer una alta precisión en los valores de salida y una estabilidad superior frente a variaciones en la carga o en la red eléctrica [7].

Estas fuentes pueden realizar secuencias programadas de variación de parámetros, simulaciones de condiciones reales de operación y pruebas de dispositivos bajo diferentes escenarios. Incorporan mecanismos de seguridad como protección contra sobrecarga, sobrecalentamiento, cortocircuito y polaridad inversa. El principio básico de una fuente de alimentación programable radica en la conversión de la energía eléctrica de la red, generalmente en corriente alterna, a una salida de corriente continua o corriente alterna ajustable. Esto se logra mediante etapas de rectificación, filtrado, regulación y control [8]. La rectificación convierte la corriente alterna en corriente continua, el filtrado elimina las fluctuaciones y ruidos en la señal, la regulación ajusta los valores de voltaje y corriente según lo configurado por el usuario, y el control monitorea y ajusta los parámetros de salida garantizando la estabilidad y el cumplimiento de las especificaciones.

Estas fuentes de alimentación se utilizan en investigación y desarrollo para probar componentes electrónicos como transistores, LEDs y sensores, y en producción, en pruebas de calidad y verificación de dispositivos antes de su distribución. También son esenciales en sistemas embebidos para la alimentación de circuitos durante su diseño y depuración, así como en la simulación de condiciones reales de operación, replicando variaciones en el suministro eléctrico para evaluar el rendimiento de los equipos. Entre sus ventajas destacan la flexibilidad y personalización, la alta precisión y confiabilidad, y la posibilidad de automatización y control remoto.

### ***Simulador de red Itech IT7900***

La serie IT7900 es un simulador de red programable de cuatro cuadrantes. También es un amplificador de potencia de cuatro cuadrantes, que se puede utilizar para probar varios equipos conectados a la red. Por ejemplo, PCS, sistema de almacenamiento de energía, microrredes, BOBC (V2X), PHiL, etc. Con tecnología avanzada de SiC, una sola unidad de IT7900 puede realizar la prueba de protección anti-isla a través del modo isla (RLC configurable). Además, la densidad de potencia de la serie IT7900 es muy alta, 6 kVA en 1U, 15 kVA en 3U. Después de la conexión en paralelo, la potencia se puede extender a 960 kVA como máximo. Los ricos modos de funcionamiento cumplen con varios requisitos de prueba de monofásica, trifásica, fase inversa y multicanal. En modo inverso, el voltaje se puede extender al 200% del voltaje nominal. La potente función de edición de forma de onda arbitraria puede simular varias formas de onda de perturbación de la red eléctrica, lo que lo convierte en una opción ideal para laboratorios de prueba e I+D. [9]



Fig. 1. Emulador de red Itech IT7900 [9].

### ***Fuente de alimentación DC programable bidireccional Itech IT6000C***

La serie IT6000C es una fuente de alimentación de CC programable bidireccional que adopta la tecnología basada en SiC de tercera generación. Integra la función de fuente y sumidero en una unidad. Basado en estas dos funciones, IT6000C ofrece la funcionalidad de operación de dos cuadrantes. La capacidad permite que la energía consumida se devuelva a la red de manera limpia, ahorrando costos de consumo de energía y refrigeración, sin interferir con la red. La serie

IT6000C proporciona un voltaje de salida máximo. 2250 V, admite paralelo maestro-esclavo con distribución de corriente promedio, potencia de salida máxima de hasta 2 MW [10].



Fig. 2. Fuente de alimentación DC programable bidireccional Itech IT6000C [10].

### *Cargas Electrónicas programables*

Una carga programable es un dispositivo electrónico diseñado para simular y controlar de manera precisa el comportamiento de una carga eléctrica bajo diversas condiciones. Estas cargas se usan en pruebas y validación de fuentes de energía, baterías, convertidores de potencia y otros dispositivos electrónicos. Su principal función es absorber energía en forma controlada, permitiendo ajustar parámetros como corriente, voltaje, resistencia y potencia, dependiendo del modo de operación seleccionado.

El funcionamiento de una carga programable se basa en la capacidad de controlar y ajustar de manera precisa los parámetros de carga mediante un circuito interno. Estos dispositivos cuentan con diferentes modos de operación, como modo de corriente constante (CC), voltaje constante (CV), resistencia constante (CR) y potencia constante (CP). En el modo CC, la carga absorbe una corriente fija independientemente del voltaje de entrada. En el modo CV, la carga mantiene un voltaje constante mientras varía la corriente. En el modo CR, la resistencia se mantiene constante, y en el modo CP, la potencia absorbida permanece fija. Estas funcionalidades hacen que las cargas programables sean herramientas esenciales para la simulación de condiciones reales en equipos electrónicos y de potencia.

### ***Carga electrónica programable Itech IT8600***

IT8600 es la última serie de cargas electrónicas de CA/CC de ITECH con un rango de potencia de 0 a 14,4 kVA y una frecuencia ajustable de 45 Hz a 450 Hz. La exclusiva función de visualización de la forma de onda del osciloscopio de IT8600 puede mostrar la tensión y la corriente de entrada como forma de onda. Está equipado con modos de medición para diferentes parámetros, como la corriente de entrada, el valor pico, el valor efectivo y el factor de potencia (PF). La capacidad de medición de armónicos de tensión es de hasta el 50%. Las interfaces de comunicación LAN y USB integradas son para un control rápido y fiable [11].



Fig. 3. Carga electrónica programable Itech IT8600 [11].

#### IV. METODOLOGÍA

El desarrollo de este proyecto se llevó a cabo utilizando una metodología en cascada, estructurada en cinco etapas principales: diseño del sistema de comunicación, desarrollo de scripts de automatización, implementación del sistema de monitoreo, validación del sistema y documentación. A continuación, se describen las etapas y actividades realizadas:

##### *Etapa 1: Diseño del Sistema de Comunicación*

En esta etapa se estableció la comunicación entre los equipos de laboratorio (IT8615L, IT6000C, IT7900) y el sistema de control central mediante la red Ethernet. El diseño y configuración de un sistema de comunicación eficiente es esencial para garantizar la conectividad y funcionalidad en entornos de red modernos. En este proyecto, se implementó una red LAN basada en una topología de estrella utilizando el conmutador Alcatel OmniSwitch 6250-P24, tal como muestra la figura 1.

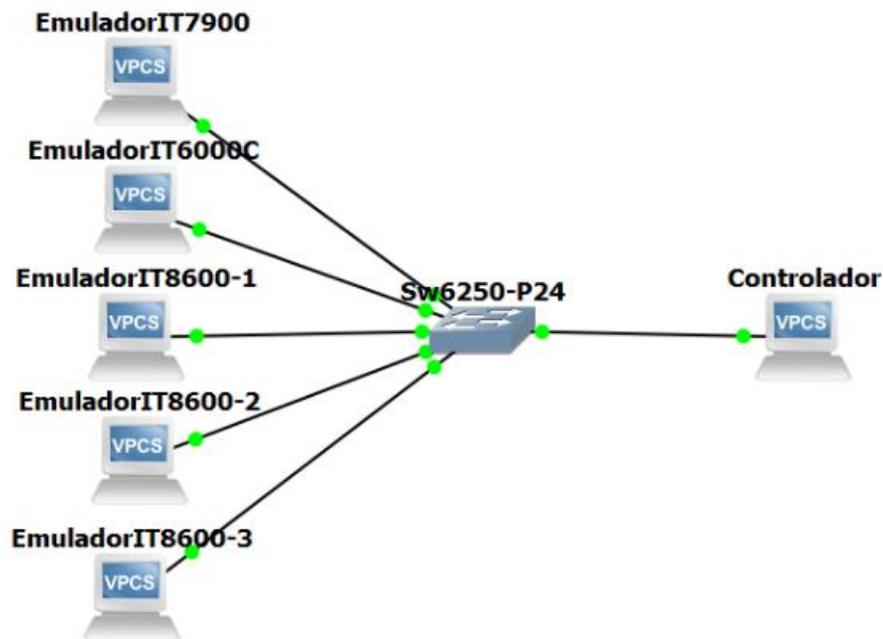


Fig. 4. Topología implementada en la red LAN del laboratorio de GIMEL.

La red fue configurada siguiendo los estándares del IEEE 802.3ab para Gigabit Ethernet, que garantiza velocidades de transferencia de hasta 1 Gbps sobre cables UTP categoría 5 [12]. Este tipo de cable, según el estándar ANSI/TIA-568A, proporciona características superiores de reducción de interferencias y capacidad para distancias de hasta 100 metros sin pérdida significativa de señal [13].

El conmutador Alcatel OmniSwitch 6250-P24 fue configurado para gestionar el tráfico de datos de forma eficiente mediante la creación de VLAN, asignación de direcciones IP estáticas. Estos ajustes se realizaron mediante el acceso al CLI (Command Line Interface) del switch, utilizando comandos específicos descritos en su manual técnico oficial [14].

### **Actividades realizadas:**

1. Montaje de la red LAN:
  - a. Se seleccionaron cables UTP categoría 6 debido a sus propiedades óptimas de transmisión y compatibilidad con velocidades Gigabit Ethernet.
  - b. La topología de estrella permitió centralizar las conexiones en el conmutador, facilitando la organización y expansión futura de la red.
2. Configuración del conmutador:
  - a. Acceso inicial: Se utilizó un cable consola conectado al puerto de consola del switch. En la PC, se abrió la terminal PuTTY, configurada en el puerto COM13 a una velocidad de 9600 baudios.
  - b. Inicio de sesión: Se accedió al sistema con las credenciales predeterminadas del dispositivo.
  - c. Asignación de nombre: Se configuró el nombre del switch como gimel-instruments.
  - d. Configuración de VLAN: Se creó la VLAN 10 para la segmentación de tráfico, permitiendo conectar los equipos de instrumentos y el controlador.

### ***Etapa 2: Desarrollo de Scripts de Automatización***

En esta etapa se desarrolló una librería en Python para facilitar la comunicación y automatización de instrumentos de laboratorio conectados a través de la red LAN. Para esto, se

utilizó la biblioteca PyVISA, que proporciona una interfaz estándar para la comunicación con dispositivos compatibles con el protocolo SCPI.

La librería fue diseñada con una arquitectura modular, implementando una clase padre que encapsula los métodos comunes necesarios para la conexión y ejecución de comandos generales, así como clases hijas que extienden la funcionalidad para cada tipo de instrumento, adaptándose a sus características y funciones específicas, la figura 2, muestra el diagrama de clases implementados en la librería y la dependencia con Pyvisa.

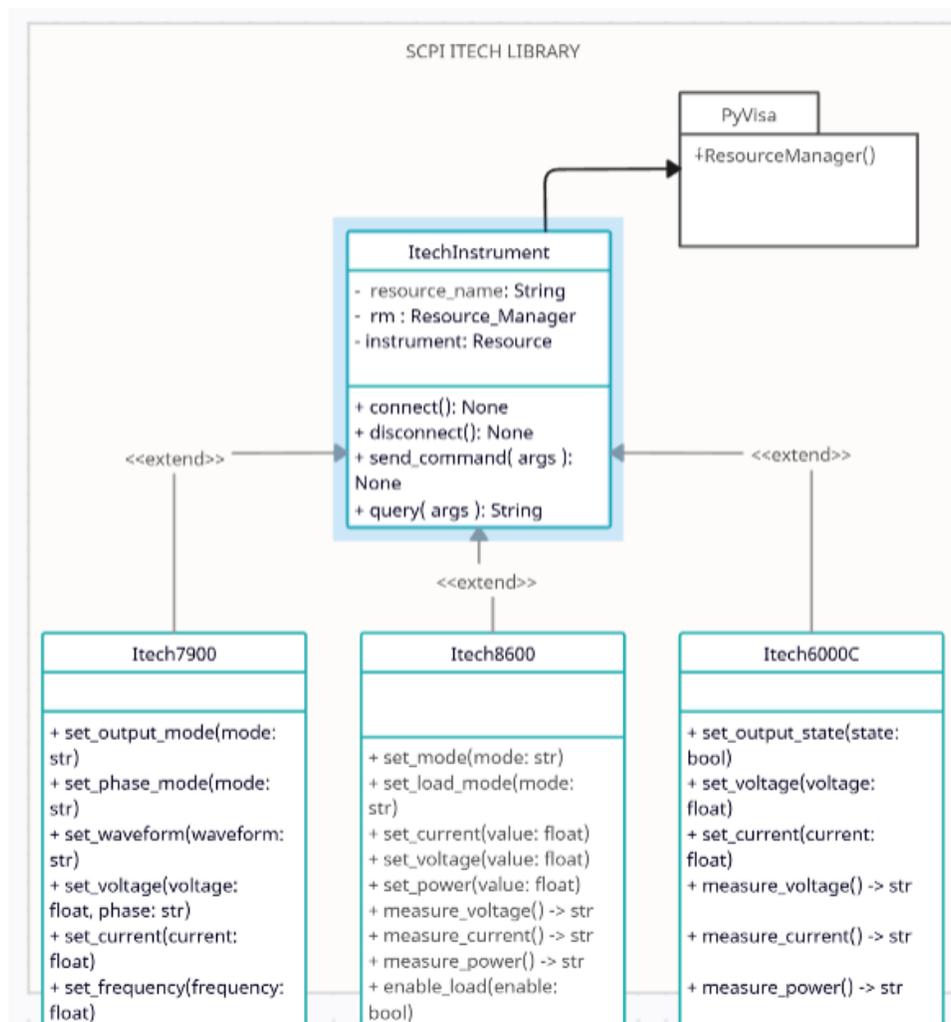


Fig. 5. Diagrama de clases implementado en la librería para controlar los instrumentos ITECH.

### Actividades realizadas:

1. Diseño de la arquitectura de la librería:
  - a. Se definió una estructura de clases orientada a objetos.
  - b. Se definió una clase padre que contiene métodos comunes tales como establecer conexión con el instrumento, verificar el estado del dispositivo mediante comandos SCPI estándar como **\*IDN?** y **\*RST**.
  - c. Se definió clases hijas para cada tipo de instrumento, las cuales implementan métodos específicos para cada tipo de instrumento, tales como configuración de voltaje y corriente para fuentes de poder, manejo de cargas electrónicas.
2. Implementación del módulo controlador:
  - a. Se construyeron métodos en la clase padre para establecer una conexión robusta con los dispositivos mediante PyVISA. Se definieron las siguientes funciones en la clase base: `connect()`, para establecer la conexión con un instrumento especificando la dirección de recurso VISA. `send_command()` para enviar comandos SCPI al instrumento. `query_command()` que envía un comando SCPI y retorna la respuesta. `disconnect()`: Cierra la conexión de manera segura. Extensión
3. para instrumentos específicos:
  - a. Se desarrollaron clases para cada tipo de equipo (IT8615L, IT6000C, IT7909-350-90), incluyendo: Métodos para configuraciones específicas.
  - b. Operaciones de medición y control.

### ***Etapa 3: Implementación del sistema de monitoreo***

La implementación de un sistema de monitoreo en tiempo real es esencial para supervisar las operaciones de los equipos de laboratorio y garantizar que las condiciones críticas se controlen eficazmente. En esta etapa, se desarrolló una interfaz gráfica interactiva que permite visualizar datos en tiempo real, incluyendo gráficos y alertas de parámetros clave como voltajes, corrientes y otros indicadores relevantes.

#### **Actividades realizadas:**

1. Diseño de la interfaz gráfica (GUI):
  - a. Se utilizó Tkinter como marco principal para la creación de la GUI, dada su simplicidad y soporte nativo en Python.

- b. Se integraron gráficos en tiempo real mediante Matplotlib, permitiendo la visualización dinámica de los datos.
  - c. La interfaz incluye: Gráficos de líneas para parámetros críticos (voltajes y corrientes). Indicadores de estado para alertas críticas. Controles básicos para configurar parámetros.
2. Integración con la librería de comunicación:
  - a. Se conectó la interfaz gráfica con la librería desarrollada en la Etapa 2, utilizando las clases específicas para cada instrumento.
  - b. Los datos se obtienen mediante los comandos SCPI implementados en la librería y se actualizan dinámicamente en la GUI.

#### ***Etapa 4: Validación del Sistema***

La validación del sistema es una etapa crucial en el desarrollo de cualquier solución tecnológica, ya que asegura que el sistema cumple con las expectativas predefinidas y opera de manera confiable bajo diferentes condiciones. En este caso, el enfoque principal fue garantizar que tanto los equipos de laboratorio como la interfaz de monitoreo respondan adecuadamente a los comandos y presenten información precisa y en tiempo real. Este proceso incluyó la ejecución de pruebas funcionales detalladas, la comparación de resultados obtenidos frente a valores esperados, y la implementación de ajustes necesarios para optimizar el rendimiento.

Además, se llevaron a cabo pruebas exhaustivas del sistema de monitoreo en tiempo real, verificando su capacidad para gestionar actualizaciones constantes de datos sin interrupciones. También se simularon escenarios de fallos comunes, como desconexiones de dispositivos o datos fuera de rango, para validar la robustez y resiliencia del sistema. Este enfoque permitió identificar y resolver posibles problemas, asegurando que el sistema pudiera operar de forma confiable y confiable.

#### **Actividades realizadas**

1. Ejecución de pruebas funcionales:
  - a. Se validó que los equipos respondan correctamente a los comandos enviados a través de la librería y la interfaz gráfica.

- 
- b. Método: Se definieron casos de prueba específicos para cada dispositivo: Verificación de comandos SCPI estándar como \*IDN? y MEAS:VOLT?. Configuración de parámetros operativos (voltaje, corriente, modos de carga, etc.).
  - c. Resultados esperados: Respuestas precisas de los equipos en los tiempos definidos.
2. Validación de la interfaz de monitoreo en tiempo real:
- a. Frecuencia de actualización: Se verificó que la interfaz pudiera manejar actualizaciones de datos con una frecuencia de 0.5 milisegundo sin interrupciones ni retrasos. Se evaluó el rendimiento del sistema para garantizar que el uso de recursos (CPU y memoria) estuvieran dentro de límites aceptables.
  - b. Pruebas de escenarios de fallo: Desconexión de equipos: Simulación de pérdida de conexión para verificar que el sistema notificará al usuario de manera oportuna. Datos fuera de rango: Generación de valores extremos para asegurar que las alertas visuales y sonoras se activen correctamente.

### ***Etapa 5: Documentación***

La etapa de documentación es esencial para garantizar la correcta implementación, operación y mantenimiento del sistema desarrollado. Se elaboró un manual técnico integral que no solo detalla los procedimientos necesarios para manejar y configurar el sistema, sino que también proporciona una guía clara y estructurada para futuras mejoras y ampliaciones. Este manual abarca todos los aspectos técnicos clave del proyecto, desde la configuración de los dispositivos hasta el uso de las API y la interfaz gráfica.

El enfoque principal fue ofrecer una documentación accesible tanto para técnicos con experiencia como para nuevos usuarios, asegurando la replicabilidad del sistema y facilitando su adopción en diferentes entornos.

---

**Actividades realizadas**

1. Documentación de la configuración de las interfaces LAN de los instrumentos:
  - a. Descripción de las direcciones IP asignadas a cada dispositivo.
  - b. Detalle de los pasos necesarios para configurar las interfaces de red en los equipos IT8615L, IT6000C e IT7909-350-90.
2. Documentación de la configuración del switch:
  - a. Instrucciones detalladas para acceder al switch mediante un cable de consola y la terminal PuTTY.
  - b. Comandos necesarios para configurar el nombre del switch, las VLAN y las direcciones IP.
3. Documentación técnica de las API:
  - a. Estructura de la librería desarrollada, explicando la función de cada clase y método.
4. Documentación de la UI del sistema:
  - a. Explicación detallada de las funcionalidades disponibles en la interfaz gráfica.
  - b. Guía para interpretar gráficos en tiempo real y gestionar alertas críticas.
  - c. Instrucciones para personalizar los parámetros de monitoreo y configuración.
  - d. Uso de la API de controladores
  - e. Descripción del flujo de trabajo para el control de los instrumentos.

## V. ANÁLISIS DE RESULTADOS

Siguiendo la metodología estructurada en etapas, se procedió a analizar los resultados de cada fase del proyecto de manera independiente, verificando y evaluando la funcionalidad de los componentes desarrollados. Este enfoque permitió identificar fortalezas, validar el cumplimiento de los objetivos propuestos y analizar posibles causas de error para optimizar el sistema.

### *Diseño del Sistema de Comunicación*

Se implementó una red LAN en topología estrella utilizando un switch Alcatel OmniSwitch 6250-P24. Se creó una VLAN 10 (VLAN nativa) y se asignaron estáticamente los puertos 1/1-6 a esta VLAN para conectar 5 instrumentos y un ordenador de control. La VLAN 10 utiliza el protocolo Ethernet y se configuró con el rango de direcciones IP 192.168.0.xx/24, asignando las direcciones de forma estática. Esta elección permite una segmentación lógica de la red y facilita la gestión.



Fig. 6. Emuladores conectados a la red local en el laboratorio GIMEL.

Para garantizar una comunicación eficiente, se verificó que todos los dispositivos tuvieran configuradas correctamente sus interfaces de red, incluyendo la dirección IP, máscara de subred y puerta de enlace. Además, se habilitó el puerto 8080 en los instrumentos para permitir conexiones TCP/IP mediante sockets. Las pruebas realizadas demostraron una latencia promedio de 1 ms entre el controlador y los instrumentos, tal como muestra la figura 7, lo que es fundamental para garantizar una respuesta en tiempo real. La topología en estrella, combinada con la VLAN 10, proporciona una alta confiabilidad y escalabilidad, permitiendo la conexión de hasta 23 emuladores adicionales en el futuro.

```

PS C:\Users\Informativa> ping 192.168.0.100
Haciendo ping a 192.168.0.100 con 32 bytes de datos:
Respuesta desde 192.168.0.100: bytes=32 tiempo=1ms TTL=64
Estadísticas de ping para 192.168.0.100:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 1ms, Máximo = 1ms, Media = 1ms
PS C:\Users\Informativa> ping 192.168.0.101
Haciendo ping a 192.168.0.101 con 32 bytes de datos:
Respuesta desde 192.168.0.101: bytes=32 tiempo=1ms TTL=64
Estadísticas de ping para 192.168.0.101:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 1ms, Máximo = 1ms, Media = 1ms
PS C:\Users\Informativa> ping 192.168.0.102
Haciendo ping a 192.168.0.102 con 32 bytes de datos:
Respuesta desde 192.168.0.102: bytes=32 tiempo=2ms TTL=64
Respuesta desde 192.168.0.102: bytes=32 tiempo=1ms TTL=64
Respuesta desde 192.168.0.102: bytes=32 tiempo=2ms TTL=64
Estadísticas de ping para 192.168.0.102:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 1ms, Máximo = 2ms, Media = 1ms
PS C:\Users\Informativa> ping 192.168.0.103
Haciendo ping a 192.168.0.103 con 32 bytes de datos:
Respuesta desde 192.168.0.103: bytes=32 tiempo=1ms TTL=64
Respuesta desde 192.168.0.103: bytes=32 tiempo=1ms TTL=64
Respuesta desde 192.168.0.103: bytes=32 tiempo=2ms TTL=64
Respuesta desde 192.168.0.103: bytes=32 tiempo=1ms TTL=64
Estadísticas de ping para 192.168.0.103:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 1ms, Máximo = 2ms, Media = 1ms
PS C:\Users\Informativa> ping 192.168.0.104
Haciendo ping a 192.168.0.104 con 32 bytes de datos:
Respuesta desde 192.168.0.104: bytes=32 tiempo=1ms TTL=64
Respuesta desde 192.168.0.104: bytes=32 tiempo=1ms TTL=64
Respuesta desde 192.168.0.104: bytes=32 tiempo=1ms TTL=64
Respuesta desde 192.168.0.104: bytes=32 tiempo=2ms TTL=64
Estadísticas de ping para 192.168.0.104:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 1ms, Máximo = 2ms, Media = 1ms

```

Fig. 7. Pruebas de comunicación entre el controlador y los instrumentos.

---

*Desarrollo de la Librería de Automatización*

Se desarrolló una librería de automatización en Python para facilitar la interacción con instrumentos de medición. La librería se basa en las bibliotecas PyVISA y SCPI y utiliza una arquitectura modular que incluye una clase base (`ItechInstrument`) y clases específicas para cada tipo de instrumento (por ejemplo, `Itech7900`, `Itech8600`, `Itech6000C`).

La clase base define métodos comunes como `connect()`, `disconnect()`, `send_command()` y `query()`, encapsulando las operaciones básicas de comunicación con los instrumentos. Las clases específicas heredan de la clase base e implementan métodos específicos para cada modelo de instrumento, como configuración de parámetros, realización de mediciones y control de funciones especiales. Se ha implementado un manejo robusto de excepciones para detectar y gestionar posibles errores durante la comunicación con los instrumentos. Además, se realizaron pruebas unitarias y de integración para verificar el correcto funcionamiento de la librería y garantizar su estabilidad.

La librería se ha documentado mediante docstrings, facilitando su uso y comprensión. Se han desarrollado ejemplos de uso para ilustrar cómo utilizar la librería en diferentes escenarios, como la automatización de mediciones de potencia, la configuración de fuentes de alimentación y el control de analizadores de espectro.

```
ip = "192.168.0.101"
controller = Itech7900(ip)

controller.connect()
controller.set_output_mode("AC")
controller.set_phase_mode("ONE")
controller.set_waveform("SINE")
controller.set_voltage(220)
controller.set_frequency(50)
controller.enable_output(True)

print(f"Voltaje RMS: {controller.measure_voltage()} V")
print(f"Corriente RMS: {controller.measure_current()} A")
print(f"Potencia Real: {controller.measure_power()} W")

controller.enable_output(False)
controller.disconnect()
```

Fig. 8. Código de ejemplo de una implementación de la clase `Itech7900`.

### *Implementación del Sistema de Monitoreo*

Se implementó un sistema de monitoreo en tiempo real para supervisar y controlar los instrumentos. La interfaz gráfica, desarrollada utilizando Python con Tkinter, proporcionó una visualización intuitiva de datos críticos como voltajes, corrientes y potencia con una frecuencia de actualización de 0.5 milisegundo, garantizando una respuesta rápida ante cualquier anomalía. Para asegurar la detección oportuna de eventos críticos, se implementó un sistema de alertas visuales ante condiciones fuera de los límites establecidos. Estas alertas se generaron en tiempo real gracias a la constante comparación de los datos medidos con los umbrales predefinidos.

La arquitectura del sistema se basó en el patrón Modelo-Vista-Controlador (MVC), lo que permitió una clara separación entre la lógica de negocio, la presentación de datos y la gestión de las interacciones del usuario tal como muestra la figura 9. La implementación del controlador se llevó a cabo utilizando la librería SCPI Itecth, desarrollada en la etapa 2 del proyecto. Esta librería facilitó la comunicación con los instrumentos de medida a través del protocolo SCPI, permitiendo una configuración flexible y una adquisición de datos eficiente. El sistema demostró un excelente rendimiento, manteniendo una tasa de actualización constante incluso con múltiples dispositivos conectados. Además, la arquitectura MVC y el uso de la librería SCPI contribuyeron a una mayor modularidad y mantenibilidad del código.

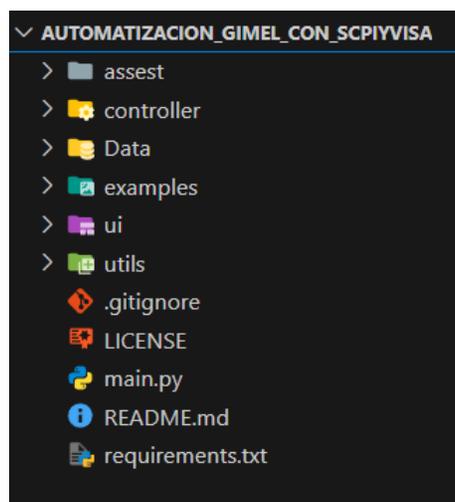


Fig. 9. Estructura del proyecto de automatización con SCPI y Pyvisa .

### Validación del Sistema

Buscando garantizar el funcionamiento del sistema de monitoreo, se realizaron pruebas exhaustivas. Estas pruebas incluyeron tanto la evaluación de la funcionalidad básica del sistema como su capacidad para operar bajo condiciones de conexión y monitoreo de todos los instrumentos, asegurando su confiabilidad y robustez.

```
PS C:\Users\omard\Documents\Gimel\Automatizacion_GIMEL
Conectado a TCP/IP::192.168.0.100::8080::SOCKET
Comando enviado: SYST:FUNC ONE
Comando enviado: FUNC AC
Comando enviado: VOLT 220.0
Comando enviado: CURR 10.0
Comando enviado: FREQ 60.0
Comando enviado: OUTP ON
Comando enviado: SYST:FUNC ONE
Comando enviado: FUNC AC
Comando enviado: VOLT 200.0
Comando enviado: CURR 10.0
Comando enviado: FREQ 60.0
Comando enviado: OUTP ON
Comando enviado: SYST:FUNC ONE
Comando enviado: FUNC AC
Comando enviado: VOLT 200.0
Comando enviado: CURR 10.0
Comando enviado: FREQ 50.0
Comando enviado: OUTP ON
Comando enviado: SCOP:RUN
Comando enviado: SCOP:STOP
```

Fig. 10. Comandos enviados durante una de las pruebas funcionales.

En las pruebas funcionales, se realizó una configuración detallada de los instrumentos de medida, ajustando parámetros como voltaje, corriente, potencia, frecuencias y protecciones para sobrecorriente o sobrevoltaje. Se verificó visualmente a través de la interfaz de los instrumentos que las configuraciones se aplicaron correctamente. Además, se validaron los datos tomados en tiempo real de los valores de salida de los instrumentos, comparándolos con los valores esperados según las configuraciones establecidas. Los resultados obtenidos, ilustrados en la Figura 11, confirmaron el correcto funcionamiento y la precisión de los datos recibidos por los instrumentos. Asimismo, se evaluó la respuesta de los instrumentos al enviar una variedad de comandos SCPI, verificando la correcta implementación de la librería utilizada. Los comandos fueron ejecutados con éxito, obteniendo las respuestas esperadas.



Fig. 11. Interfaz gráfica mostrando los resultados de una configuración para una red 220V 60Hz.

En cuanto a las pruebas de estrés, se simularon escenarios de alta carga conectando simultáneamente todos los instrumentos al sistema, tal como se observa en la figura 6. Se observó el consumo de recursos por parte del controlador, este demostró mantener un gasto constante de los recursos del sistema. también se evaluó la capacidad del sistema para manejar desconexiones inesperadas de los instrumentos, probando su habilidad para seguir funcionando y generando la alerta. En todos los casos, el sistema logró reanudar la adquisición de datos sin interrupciones significativas.



Fig. 12. Emulador de red Itech IT7900 funcionando con los parámetros para una red 220V 60Hz.

Nombre	22% CPU	62% Memoria	1% Disco	Consumo de energía
Python	17,8%	78,8 MB	0,1 MB/s	Muy alta
GIMEL CONTROL EMULADORES				

The screenshot shows the GIMEL software interface. On the left, there is a sidebar with a logo and three emulators: 'Emulador AC IT7900', 'Emulador DC IT7900', and 'Emulador Carga IT8600'. The main window has tabs for 'Conexión', 'Configuración de Salida', 'Protecciones', and 'Osciloscopio y Gráficos'. The 'Configuración de Salida' tab is active, showing settings for 'Modo de Operación' (AC), 'Voltaje (V)' (220), 'Corriente (A)' (1), 'Frecuencia (Hz)' (60), and 'Potencia (W)' (220). An 'Aplicar Configuración' button is visible. A warning dialog box titled 'Advertencia' is open, displaying a yellow warning icon and the text 'Conéctese al dispositivo primero.' with an 'Aceptar' button.

Fig. 13. Recursos consumidos por el software durante las pruebas y alertas de desconexión.

### Documentación

Se desarrolló un manual técnico integral para garantizar la replicabilidad y mantenimiento del sistema. Este documento detalla la arquitectura del sistema, incluyendo la configuración de la red, las interfaces de los instrumentos, la estructura de la API y la interfaz gráfica de usuario (GUI). Se incluyen diagramas, capturas de pantalla y ejemplos de código para facilitar la comprensión. Además, se proporcionan instrucciones detalladas para la instalación, configuración y solución de problemas. La documentación se generó utilizando herramienta de Markdown asegurando un formato profesional y fácil de navegar. Este invaluable recurso facilita la transferencia de conocimiento, agiliza el soporte técnico y garantiza la longevidad del proyecto.

---

## VI. CONCLUSIONES

Este trabajo se enfocó en el diseño e implementación de un sistema de control y monitoreo en tiempo real para equipos de laboratorio, utilizando una red Ethernet basada en una topología de estrella, comandos SCPI y la biblioteca PyVISA. El objetivo principal fue desarrollar una solución robusta que permita la automatización de procesos, la supervisión dinámica de parámetros críticos y la mejora de la eficiencia operativa en entornos de laboratorio. A través de un enfoque metodológico basado en etapas, se logró construir una infraestructura tecnológica eficiente y escalable, cumpliendo con los objetivos planteados.

El sistema desarrollado incluyó la configuración de una red LAN confiable mediante el uso del conmutador Alcatel OmniSwitch 6250-P24, logrando tiempos de respuesta óptimos y una conectividad estable entre los equipos e instrumentos del laboratorio. Además, se implementó una librería modular en Python, capaz de integrar de manera efectiva múltiples dispositivos utilizando PyVISA y SCPI, lo que simplificó la automatización de tareas repetitivas y garantizó una comunicación eficiente entre los componentes del sistema y se presenta como un recurso accesible y valioso para la comunidad de Python que permitirá expandir la librería y recibir retroalimentación.

En cuanto al monitoreo en tiempo real, se diseñó una interfaz gráfica que permite visualizar de manera dinámica parámetros clave como voltajes y corrientes. La inclusión de alertas visuales añadió un nivel de supervisión que asegura la detección temprana de condiciones críticas, clave para mantener la seguridad y operatividad en el laboratorio. Las pruebas realizadas validaron el correcto funcionamiento del sistema, confirmando su precisión, estabilidad y capacidad para manejar escenarios adversos, como desconexiones o datos fuera de rango.

La documentación técnica desarrollada representa un aporte significativo al proyecto, proporcionando una guía clara y detallada para la implementación, operación y mantenimiento del sistema. Esto no solo facilita su uso en el presente, sino que también asegura la replicabilidad y futuras expansiones en entornos similares.

## VII. TRABAJOS FUTUROS

En cuanto a trabajos futuros, se propone la ampliación del sistema para integrar nuevos equipos y dispositivos, adaptando la librería existente para soportar más funcionalidades específicas. También se recomienda explorar el desarrollo de interfaces más avanzadas, utilizando tecnologías modernas como PyQt o Dash, que mejoren la experiencia del usuario y permitan una mayor personalización. La implementación de un sistema de monitoreo remoto, complementado con almacenamiento en la nube para registrar y analizar datos históricos, podría agregar un valor significativo al sistema, permitiendo su uso desde cualquier ubicación y brindando herramientas para el análisis predictivo.

Finalmente, se sugiere la incorporación de algoritmos de inteligencia artificial para identificar patrones de operación, detectar anomalías y anticipar fallos, mejorando la proactividad del sistema. Estos desarrollos futuros no solo ampliarían las capacidades del sistema, sino que también consolidarían su utilidad como herramienta avanzada para la gestión de equipos de laboratorio en tiempo real.

Departamento de Ingeniería Electrónica y de Telecomunicaciones

# Automatización y Monitoreo en Tiempo Real de Equipos de Laboratorio en GIMEL Usando SCPI y PyVISA



## UNIVERSIDAD DE ANTIOQUIA

Facultad de Ingeniería

PRACTICANTE: Omar David Vargas Bonett

ASESORES: José Robinson Ortiz Castrillón

PROGRAMA: Ingeniería Electrónica

Semestre de la práctica: 2024-2

En el grupo GIMEL se desarrolló un sistema de control y monitoreo para equipos de laboratorio, el cual permite configurar, operar y supervisar de manera remota cinco dispositivos a través de una red Ethernet. Utilizando Python, SCPI y una interfaz gráfica, se logró una herramienta capaz de visualizar datos en tiempo real, generar alertas y garantizar una comunicación. Este sistema, además de ser escalable, ofrece una librería Python de libre acceso para la comunidad científica, facilitando la integración de nuevos equipos y funcionalidades.



Este sistema, además de ser preciso y escalable, ofrece una librería Python de libre acceso para la comunidad científica, facilitando la integración de nuevos equipos y funcionalidades. La implementación de este proyecto representa un avance significativo en la automatización de procesos de laboratorio.



## Introducción

La configuración manual y el monitoreo constante de equipos de laboratorio son tareas laboriosas y propensas a errores. Para solucionar esto, se desarrolló un sistema automatizado que permite controlar de manera remota y eficiente cinco equipos de laboratorio en el grupo GIMEL de la Universidad de Antioquia. Este sistema, basado en comandos SCPI y Python, facilita la configuración, operación y monitoreo en tiempo real de los equipos, mejorando la eficiencia, reduciendo errores humanos y permitiendo un análisis más preciso de los datos.



## Metodología

El desarrollo de este proyecto se llevó a cabo utilizando una metodología en cascada, estructurada en cinco etapas principales: diseño del sistema de comunicación, desarrollo de scripts de automatización, implementación del sistema de monitoreo, validación del sistema y documentación.



## Resultados

El desarrollo de este sistema ha demostrado la viabilidad de utilizar una red Ethernet y la programación en Python para construir una solución de control y monitoreo robusta y escalable. La implementación de una librería modular en Python, junto con el uso de comandos SCPI y PyVISA, ha facilitado la integración de diversos equipos de laboratorio y la comunicación eficiente entre ellos.

**Más información sobre el proyecto**



## Objetivos

- ✓ Comunicar los equipos del laboratorio y el sistema de control central mediante una red LAN Ethernet, garantizando una conexión estable y eficiente que permita la automatización y el monitoreo en tiempo real de los equipos.
- ✓ Automatizar la configuración y operación de los equipos mediante Scripts en Python, reduciendo el tiempo de configuración manual y mejorando las pruebas en el laboratorio.
- ✓ Monitorear variables y datos en tiempo real de los emuladores de carga, la fuente DC y el emulador AC, a través de la red Ethernet y visualizarlos en una interfaz de monitoreo asegurando el acceso inmediato de los datos críticos.
- ✓ Verificar las funcionalidades del sistema

## Conclusiones

- ✓ La solución se basa en una arquitectura robusta y confiable, utilizando componentes de alta calidad y software modular. La interfaz gráfica facilita la interacción con el sistema y la visualización de datos.
- ✓ La interfaz gráfica intuitiva permite a los usuarios visualizar en tiempo real los parámetros clave de los equipos, facilitando la detección temprana de anomalías y la toma de decisiones informadas.
- ✓ Al emplear tecnologías como PyVISA y SCPI, se ha logrado una comunicación eficiente y precisa con una amplia variedad de instrumentos. La arquitectura de red basada en una topología de estrella, combinada con el conmutador Alcatel OmniSwitch, garantiza una conectividad robusta y de bajo latencia.

DATOS DE CONTACTO DEL AUTOR:



+57 3122505311



odavid.vargas@udea.edu.co



david\_bonett

<https://www.linkedin.com/in/davidbonett19/>

## REFERENCIAS

1. D. Homan, "Acelerar el control y la automatización de instrumentos", en IEEE Instrumentation & Measurement Magazine, vol. 16, núm. 4, págs. 40-44, agosto de 2013, doi: 10.1109/MIM.2013.6572953
2. W. R. Rodríguez Dueñas, "Software libre para educación e investigación en ingeniería," Revista Educación en Ingeniería, vol. 9, no. 18, pp. 12–22, 2014. [Online]. Available: <https://doi.org/10.26507/rei.v9n18.383>
3. J. Pieper, Automatic Measurement Control: A Tutorial on SCPI and IEEE 488.2. Rohde & Schwarz, 2007.
4. L. Radoslav, Ed., Virtual Instrument Software Architecture, 1st ed. Pon Press, 2012.
5. IVI Foundation, "IVI Foundation," [Online]. Available: <https://www.ivifoundation.org/>. [Accessed: Sep. 16, 2024].
6. IEEE, IEEE Standard Codes, Formats, Protocols, and Common Commands for Use With ANSI/IEEE Std 488.1-1987 IEEE Standard Digital Interface for Programmable Instrumentation, IEEE Std 488.2-1987, 1987.
7. Keysight Technologies, "Programmable Power Supply," [Online]. Available: <https://www.keysight.com/blogs/en/tech/educ/2023/programmable-power-supply>. [Accessed: Dec. 21, 2024]
8. J. López Crespo, Módulo 4. Fundamentos de electrónica. Ediciones Paraninfo, S.A., 2015. ISBN: 9788428336628.
9. ITECH Electronics, "AC Power Supply - IT7900," [Online]. Available: <https://www.itechate.com/en/product/ac-power-supply/IT7900.html>. [Accessed: Sep. 16, 2024]
10. ITECH Electronics, "DC Power Supply - IT6000C," [Online]. Available: <https://www.itechate.com/en/product/dc-power-supply/IT6000C.html>. [Accessed: Sep. 16, 2024]
11. ITECH Electronics, "AC Electronic Load - IT8600," [Online]. Available: <https://www.itechate.com/en/product/ac-electronic-load/IT8600.html>. [Accessed: Sep. 16, 2024]

- 
12. IEEE Standard for Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements—Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications—Physical Layer Parameters and Specifications for 1000 Mb/s Operation over 4-pair of Category 5 Balanced Twisted Pair Cable, Type 1000BASE-T," IEEE Std 802.3ab-1999, pp. 1–144, Jul. 26, 1999, doi: 10.1109/IEEESTD.1999.90568.
  13. Telecommunications Industry Association, "TR-42 Telecommunications Cabling Systems," TIA Online, [Online]. Available: <https://www.tiaonline.org/>. [Accessed: Sept. 1, 2024].
  14. Alcatel-Lucent, "OmniSwitch OS6250 AOS 6.6.3 R01 Hardware User's Guide," [Online]. Available: [https://support.alcadis.nl/Support\\_files/Alcatel-Lucent/OmniSwitch//End%20of%20Sale%20products/OS6250%20-%20EOS/Manuals/OS6250%20AOS%206.6.3%20R01/OS6250%20AOS%206.6.3%20R01%20Hardware%20Users%20Guide.pdf](https://support.alcadis.nl/Support_files/Alcatel-Lucent/OmniSwitch//End%20of%20Sale%20products/OS6250%20-%20EOS/Manuals/OS6250%20AOS%206.6.3%20R01/OS6250%20AOS%206.6.3%20R01%20Hardware%20Users%20Guide.pdf). [Accessed: jul. 31, 2024]

---

## ANEXOS

Esta sección detalla cómo utilizar y configurar el sistema desarrollado para el control y monitoreo de equipos de laboratorio.

### 1. Requisitos Previos

Antes de utilizar el sistema, asegúrese de cumplir con los siguientes requisitos:

*Hardware:*

Emuladores IT8615L, IT6000C e IT7900.

Conmutador de red configurado para conectar todos los dispositivos.

Computadora con Python 3.6 o superior instalado.

*Software:*

Bibliotecas Python necesarias: PyVISA, Matplotlib, y Tkinter.

Controladores VISA instalados (NI-VISA o Keysight VISA).

### 2. Instalación del Sistema

Descargue el código fuente del sistema desde el repositorio [https://github.com/davito19/Automatizacion\\_GIMEL\\_con\\_SCPIyVISA](https://github.com/davito19/Automatizacion_GIMEL_con_SCPIyVISA)

". Instale las dependencias ejecutando: `pip install pyvisa matplotlib`

Conecte todos los dispositivos a la red utilizando el conmutador y asegúrese de que cada equipo tenga una dirección IP asignada en el rango 192.168.0.x/24.

Ejecute el programa principal: `python main.py`

### 3. Uso del Sistema

#### Establecer Conexión

Ingresa a la pestaña "Conexión" en la interfaz gráfica.

Especifique la dirección IP del dispositivo que desea controlar.

Haga clic en el botón "Conectar" para establecer la comunicación con el emulador. Si la conexión es exitosa, aparecerá un mensaje de confirmación.

### Configuración de Salida

Navegue a la pestaña "Configuración de Salida".

Seleccione el modo de operación según se especifique el instrumento utilizando el menú desplegable.

Ingrese los valores parámetros de configuración tales como:

Voltaje (V).

Corriente (A).

Frecuencia (Hz).

Potencia (W).

Haga clic en "Aplicar Configuración" para enviar los parámetros al emulador.

### Monitoreo en Tiempo Real

Diríjase a la pestaña "Osciloscopio y Gráficos".

Para visualizar los datos de voltaje instantáneo, clic en "Iniciar Gráfica".

Para detener el monitoreo, haga clic en "Detener Gráfica".

### Manual técnico en markdown

```
# Manual Técnico del Sistema de Automatización y Monitoreo de Equipos de Laboratorio
GIMEL

---

## **1. Configuración de la Red LAN**

Se configuró una red utilizando el estándar IEEE 802.3ab para Gigabit Ethernet,
logrando velocidades de hasta 1 Gbps sobre cables UTP categoría 5, según ANSI/TIA-
568A, con capacidad para distancias de hasta 100 metros sin pérdida significativa
```

de señal. El switch Alcatel OmniSwitch 6250-P24 fue ajustado mediante CLI para gestionar eficientemente el tráfico de datos, implementando VLAN y asignando direcciones IP estáticas según su manual técnico.

### ### **\*\*1.1. Descripción de las Direcciones IP Asignadas\*\***

Cada instrumento se encuentra configurado con una dirección IP única dentro del segmento de red **\*\*192.168.0.x/24\*\***, para garantizar una comunicación adecuada:

- **\*\*IT7909-350-90 (Fuente AC):\*\*** 192.168.0.100
- **\*\*IT6000C (Fuente DC):\*\*** 192.168.0.101
- **\*\*IT8615L (Emuladores de Carga):\*\***
  - Dispositivo 1: 192.168.0.102
  - Dispositivo 2: 192.168.0.103
  - Dispositivo 3: 192.168.0.104

### ### **\*\*1.2. Configuración de las Interfaces de Red y puerto\*\***

---

#### ### **\*\*1.2.1 Configuración de Conexión LAN y Puerto para Socket en el IT6000C\*\***

##### #### **Requisitos Previos**

- Asegúrate de que el equipo esté conectado a una red Ethernet funcional.
- Conecta el dispositivo a la red a través del puerto LAN de la parte trasera.
- Verifica que el dispositivo esté apagado antes de realizar cualquier conexión.

##### #### **1. Acceder al Menú de Configuración de Red**

1. Enciende el dispositivo.
2. Presiona la combinación de teclas ``[Shift]` + `[P-set]`` para entrar al menú de configuración del sistema.

##### #### **2. Configurar la Interfaz LAN**

1. Navega en el menú hasta la opción **\*\*LAN\*\*** y presiona ``[Enter]``.
2. Selecciona **\*\*IP-Conf\*\*** para configurar los parámetros de red:
  - **\*\*IP Mode\*\***:
    - ``Auto``: Para asignación automática mediante DHCP.
    - ``Manual``: Para configuración manual.
  - **\*\*IP\*\***: Introduce la dirección IP deseada (por ejemplo, ``192.168.0.100``).
  - **\*\*Mask\*\***: Introduce la máscara de subred (por ejemplo, ``255.255.255.0``).
  - **\*\*Gateway\*\***: Introduce la puerta de enlace (por ejemplo, ``192.168.0.1``).
3. Configura valores adicionales si es necesario:

```
- **DNS1**: Dirección del servidor DNS preferido.
- **DNS2**: Dirección del servidor DNS alternativo.
4. Configura el **Socket Port**: Introduce el puerto de comunicación deseado (por defecto, `30000`).

#### 3. Guardar los Cambios
1. Presiona `[Enter]` para guardar cada configuración.
2. Usa `[Esc]` para regresar al menú principal.

#### 4. Probar la Conexión
1. Conecta un dispositivo (como una computadora) en la misma red.
2. Realiza un **ping** al dispositivo utilizando la dirección IP configurada para verificar la conexión.
3. Si configuraste un puerto socket, asegúrate de que el software cliente pueda comunicarse a través del puerto especificado.

---

### **1.2.2 Configuración de Conexión LAN y Puerto para Socket en el IT7900**

#### Requisitos previos
- Conecta el equipo IT7900 a una red mediante el puerto LAN en el panel trasero.
- Asegúrate de que el dispositivo esté apagado antes de realizar cualquier conexión.

#### 1. Acceder al menú de configuración de comunicación
1. Enciende el dispositivo.
2. Presiona `[Shift]` + `[System]` en el panel frontal para acceder al menú del sistema.
3. Usa los botones de navegación o la pantalla táctil para seleccionar la opción **Set the communication interface**.

#### 2. Configurar la interfaz LAN
1. En el menú **LAN Interface**, configura los parámetros necesarios:
    - **IP Mode**:
      - `DHCP`: Asignación automática de dirección IP.
      - `Manual`: Configuración manual de la IP.
    - **IP Address**: Introduce la dirección IP deseada, por ejemplo, `192.168.0.100`.
    - **Subnet Mask**: Especifica la máscara de subred, por ejemplo, `255.255.255.0`.
    - **Gateway**: Configura la puerta de enlace, por ejemplo, `192.168.0.1`.
    - **Socket Port**: Define el puerto de comunicación, por ejemplo, `8080`.
```

#### #### 3. Activar y configurar servicios LAN

1. En el mismo menú, habilita los servicios de comunicación necesarios:
  - **RAW Socket**: Habilita la comunicación directa mediante sockets.
  - **Ping**: Activa la función de ping para probar la conectividad.
  - **Telnet-SCPI**: Permite comandos remotos a través de Telnet.

#### #### 4. Guardar los cambios

1. Una vez configurados los parámetros, presiona `[Enter]` para guardar los cambios.
2. Usa `[Esc]` para regresar al menú principal.

#### #### 5. Probar la conexión

1. Desde una computadora en la misma red, realiza un **ping** a la dirección IP configurada para confirmar la conectividad.
2. Si configuraste un puerto socket, utiliza software cliente (como PuTTY o Python) para conectarte al puerto y verificar la comunicación.

---

### ### **1.2.3 Configuración de Conexión LAN y Puerto para Socket en el IT8600**

#### #### Requisitos previos

- Conecta el equipo IT7900 a una red mediante el puerto LAN en el panel trasero.
- Asegúrate de que el dispositivo esté apagado antes de realizar cualquier conexión.

#### #### 1. Acceder al menú de configuración

1. Enciende el dispositivo.
2. Presiona el botón `[Menu]` en el panel frontal para acceder al menú principal.

#### #### 2. Configuración de la interfaz LAN

1. En el menú, selecciona **COMM CONFIG** (Configuración de comunicación) y presiona `[Enter]`.
2. Navega a la opción **LAN** y presiona `[Enter]`.
3. Configura los siguientes parámetros:
  - **IP Mode**:
    - `DHCP`: Para asignación automática de dirección IP.
    - `MANUAL`: Para configuración manual de la IP.
  - **IP Address**: Introduce la dirección IP deseada (por ejemplo, `192.168.0.100`).

```
- **Subnet Mask**: Introduce la máscara de subred (por ejemplo,
`255.255.255.0`).
- **Gateway**: Configura la puerta de enlace (por ejemplo, `192.168.0.1`).
- **Socket Port**: Define el puerto de comunicación (por ejemplo, `8080`).

#### 3. Guardar los cambios
1. Presiona `[Enter]` para guardar cada configuración.
2. Usa `[Esc]` para regresar al menú principal.

#### 4. Verificar la conectividad
1. Conecta un dispositivo en la misma red y realiza un **ping** a la dirección IP
configurada.
2. Si configuraste un puerto socket, utiliza un cliente de pruebas como PuTTY o
una herramienta de programación (por ejemplo, Python) para verificar la conexión
al puerto.

---

### **1.3 Configuración del Switch**

#### Configuración de Red en Topología Estrella con VLAN en Alcatel OS6250/6450

Este segmento describe el proceso paso a paso para configurar una red en topología
estrella utilizando VLAN en un switch Alcatel OS6250/6450.

#### Requisitos Previos

1. Un switch Alcatel OS6250/6450.
2. Software de terminal como PuTTY o Tera Term.
3. Credenciales de acceso al switch.
4. Conexión física entre los dispositivos y el switch.

### Pasos para la Configuración

#### 1. Conexión y Acceso al Switch

1. Conecta un cable de consola entre tu computadora y el puerto de consola del
switch.
2. Configura la conexión en el software de terminal:
- Velocidad: **9600 baudios**.
- Datos: **8 bits**.
- Paridad: **ninguna**.
- Bit de parada: **1 bit**.
```

3. Accede al CLI del switch con las credenciales correspondientes.

#### #### 2. Configuración de los Puertos

1. Identifica los puertos a utilizar para los dispositivos finales y el puerto troncal.

2. Habilita los puertos:

```
---
```

```
interface ethernet <puerto>
```

```
no shutdown
```

```
---
```

3. Verifica el estado de los puertos:

```
---
```

```
show interfaces status
```

```
---
```

#### #### 3. Creación de VLANs

1. Define las VLANs necesarias:

```
---
```

```
vlan <ID_VLAN>
```

```
name <nombre_VLAN>
```

```
exit
```

```
---
```

2. Repite este paso para cada VLAN requerida.

#### #### 4. Asignación de Puertos a las VLANs

- **\*\*Modo acceso\*\*** (dispositivos finales):

```
---
```

```
interface ethernet <puerto>
```

```
vlan pvid <ID_VLAN>
```

```
exit
```

```
---
```

- **\*\*Modo troncal\*\*** (hacia otro switch o router):

```
---
```

```
interface ethernet <puerto_troncal>
```

```
vlan trunk enable
```

```
vlan trunk allowed <ID_VLAN_1>,<ID_VLAN_2>,...
```

```
exit
```

```
---
```

#### #### 5. Habilitación de Etiquetado 802.1Q

```
Asegúrate de que el etiquetado de VLAN esté habilitado en los puertos troncales:  
```
```

```
interface ethernet <puerto_troncal>  
  vlan tagging enable  
  exit  
```
```

#### #### 6. Configuración de VLAN de Administración (Opcional)

1. Crea una VLAN para administración:

```
```  
vlan <ID_VLAN_ADMIN>  
  name ADMIN  
  exit  
```
```

2. Asigna una dirección IP al switch dentro de esta VLAN:

```
```  
interface vlan <ID_VLAN_ADMIN>  
  ip address <IP_ADMIN> <MASCARA>  
  exit  
```
```

#### #### 7. Verificación de la Configuración

1. Confirma las VLAN creadas:

```
```  
show vlan  
```
```

2. Verifica las interfaces asignadas:

```
```  
show interfaces vlan  
```
```

3. Revisa la configuración en ejecución:

```
```  
show running-config  
```
```

#### #### 8. Guardar la Configuración

Para evitar la pérdida de configuración tras un reinicio, guarda los cambios:

```
```  
write memory  
```
```

**#### 9. Pruebas Finales**

1. Conecta los dispositivos finales y verifica la conectividad dentro de cada VLAN.
2. Prueba la conectividad entre las VLAN y el gateway configurado.

**#### Notas**

- Consulta la documentación oficial del switch para obtener más detalles sobre comandos avanzados.

---

