Diseño e implementación de un sistema embebido para el monitoreo remoto y local de parámetros eléctricos, temperatura y humedad, en las UPS EATON serie DX, orientado a la reducción de costos de la empresa UPSISTEMAS S.A.S.

Alexander Sánchez Calvo
Diego Andrés Landinez Parra

Universidad Antonio Nariño
Facultad de Ingeniería Electrónica, Biomédica y Mecatrónica
Programa de Ingeniería Electronica
Cartagena de Indias, Colombia
2020

Diseño e implementación de un sistema embebido para el monitoreo remoto y local de parámetros eléctricos, temperatura y humedad, en las UPS EATON serie DX, orientado a la reducción de costos de la empresa UPSISTEMAS S.A.S.

Alexander Sánchez Calvo
Diego Andrés Landinez Parra

Trabajo de grado presentado como requisito para optar al título de:
Ingeniero Electrónico

Director:
Ing. Leonardo Torres Londoño

Universidad Antonio Nariño
Facultad de Ingeniería Electrónica, Biomédica y Mecatrónica
Programa de Ingeniería Electrónica
Cartagena de Indias, Colombia
2020

Agradecimientos

Principalmente a Dios por la oportunidad que día a día nos brindo para sacar este proyecto adelante. A nuestros padres que por su apoyo incondicional a pesar de la distancia, fueron clave para el impulso de superación. A nuestros docentes académicos y compañeros que en este camino trascendental fueron apoyo para cumplir este propósito. Un agradecimiento especial a nuestro director de tesis Ing. Leonardo Torres Londoño por compartir su gran experiencia, a nuestro compañero y amigo Carlos Aponte por seguir de cerca este proyecto.

Resumen

Los sectores de la salud, telecomunicaciones y áreas industriales entre otros, manejan procesos que requieren de un suministro energético sin interrupciones. Para garantizar el fluido continuo de energía, algunas empresas contratan o alquilan UPS por parte de terceros, ahorrando así en costos de implementación y mantenimiento. Las empresas contratadas deben realizar un monitoreo continuo de diferentes parámetros, lo cual implica inversiones adicionales en tarjetas de red suministradas por los fabricantes de las UPS, que a menudo poseen un elevado costo. En este documento se presenta como alternativa a las soluciones de monitoreo presentadas por el fabricante, el diseño e implementación de un dispositivo embebido que recopila e integra los parámetros suministrados por la UPS junto con mediciones externas de las variables eléctricas y ambientales. En las pruebas realizadas, el prototipo permitió la visualización de las variables eléctricas y estados de operación registrados internamente por el UPS. Por su parte, el sistema de metrología permitió mediante el uso de 6 sensores, la medición de 4 variables eléctricas y 2 variables del entorno (temperatura y humedad). Adicionalmente se pudo comprobar la integración del monitoreo remota o IoT, implementando las herramientas Grafana y thing speak. Se realizaron pruebas comparativas con herramientas certificadas para corroborar la veracidad de las medidas y posteriormente el prototipo se dejó en funcionamiento en un lapso de 3 días continuos con monitoreo de 24 horas permitiendo analizar los datos obtenidos. Comparado con las alternativas de monitoreo comerciales, incluidas las tarjetas de red sugeridas por el fabricante de las UPS, el dispositivo embebido implementado, redujo los costos de monitoreo.

Palabras Clave

Sistemas de notificaciones, monitoreo, dispositivo embebido, UPS EATON serie DX.

Abstract

The health, telecommunications and industrial sectors, among others, handle processes that require uninterrupted energy supply. To ensure continuous energy flow, some companies hire or rent UPSs from third parties, saving on implementation and maintenance costs. Contracted companies must continuously monitor different parameters, which involves additional investments in network cards supplied by UPS manufacturers, which often have a high cost. This document presents as an alternative to the monitoring solutions presented by the manufacturer, the design and implementation of an embedded device that collects and integrates the parameters supplied by the UPS along with external measurements of electrical and environmental variables. In the tests carried out, the prototype allowed the visualization of electrical variables and operating states recorded internally by the UPS. For its part, the metrology system allowed through the use of 6 sensors, the measurement of 4 electrical variables and 2 environment variables (temperature and humidity). Additionally, the integration of remote monitoring or IoT could be verified, implementing the Grafana and thing speak tools. Comparative tests were carried out with certified tools to verify the veracity of the measurements and subsequently the prototype was left in operation within 3 continuous days with 24-hour monitoring allowing the analysis of the data obtained. Compared to commercial monitoring alternatives, including network cards suggested by the UPS manufacturer, the embedded device implemented reduced monitoring costs.

Keywords

Notification systems, monitoring, embedded device, UPS EATON DX series.

Tabla de contenido

Lista de Tablas

Lista de Figuras

1. Introducción

La empresa UPSISTEMAS S.A.S. lleva más de 25 años en el mercado colombiano como integrador de infraestructura tecnológica, Una de sus ramas más fuertes es la venta, instalación, mantenimiento y reparación de UPS. Una UPS es un artefacto electrónico cuya función principal es dar soporte eléctrico a aquellos sistemas donde el fluido de energía debe ser continuo, por ejemplo: hospitales, Datacenter, Callcenter, procesos industriales etc.

Las UPS tienen como misión principal, suministrar energía de forma constante sin interrupciones. También se encarga de garantizar que ese fluido eléctrico sea regulado.

De manera general las UPS se pueden clasificar de acuerdo a su topología, entre las más reconocidas están:

- **UPS tipo Offline:** Se caracteriza por no realizar ningún tipo de regulación, su operación se basa en suministrar energía solo durante un corte del fluido eléctrico [1].
- **UPS interactiva:** Presenta regulación y soporte en baterías ante fluido eléctrico [2].
- **UPS tipo Online:** Se caracteriza por una tensión constante debido a su capacidad de realizar un proceso de rectificación donde luego se vuelve a convertir el voltaje DC en suministro alterno, tiene como característica brindar una doble conversión en línea donde el inversor, esto garantiza un trabajo continuo del inversor permite obtener una tensión constante y regulada en la salida del UPS.

En la actualidad existe una amplia cantidad de fabricantes de UPS, en este documento se implementó una UPS del fabricante EATON, modelo DX de

1.5kva. En todo sistema crítico soportado por UPS un factor importante es el monitoreo, este tiene por objetivo facilitar la identificación de fallos y causas de los mismos. Los fabricantes proporcionan algunas soluciones relacionadas con este aspecto, entre las más relevantes están:

- **Monitoreo local utilizando una conexión punto a punto:** Consiste en una conexión directa entre el UPS y un PC, en el PC se instala un software, generalmente propietario, este software registrar los niveles de voltaje y frecuencia y almacena el histórico de eventos del UPS.
- **Monitoreo implementando una tarjeta de red snmp II:** Permite visualizar de forma remota y local el estado de operación de la UPS, registrar los niveles de voltaje y frecuencia, y almacenar el histórico de eventos del equipo. También permite el envío de notificaciones por correo electrónico [3].
- **Monitoreo con sensor ambiental para NMC de EATON:** Se ofrece como un accesorio que se conecta a la tarjeta de red snmp, su funciona principal es la de monitorear los niveles de temperatura y humedad, permite configurar alarmas y enviar notificaciones. Este sensor no trabaja solo, trabaja en conjunto con la tarjeta snmp [4].
- **Monitoreo con tarjeta relay card MS:** La función principal de esta tarjeta consiste en activar o desactivar relevos en función de los estados o modos de operación, estos relevos pueden utilizarse de distintas formas, ya sea con indicadores luminosos, señales audibles o el arranque de plantas eléctricas [5].
- **Panel o dispositivo de monitoreo local:** Consiste en la instalación de un panel con indicadores de estados de operación del UPS.

Si bien las tarjetas mencionadas son una buena alternativa de monitoreo, no dejan de tener sus limitaciones o problemas, en términos generales se puede decir que se adquieren como accesorios de alto costo, lo cual disminuye el

margen de ganancia de los proveedores, y en su mayoría carecen de mediciones del entorno de operación de la UPS (exceptuando el sensor ambiental para NMC de EATON que también se vende como accesorio). Adicional a esto las alternativas mencionadas no brindan un sistema de monitoreo unificado, es decir que se requiere adquirir diversos elementos adicionales si se desea realizar un monitoreo completo.

En este documento se presenta el diseño e implementación de un dispositivo embebido que recopila e integra los parámetros suministrados por la UPS, junto con mediciones externas de las variables eléctricas y ambientales.

En el capítulo uno, se encuentra información correspondiente a investigaciones realizadas por otros autores, enfocadas a diseñar e implementar sistemas de monitoreo más económicos o eficientes. Adicionalmente se muestra la problemática presente en las tarjetas de red y sistemas de monitoreo que ofrece el fabricante, incluyendo en el mismo capítulo los objetivos propuesto destinados a corregir o solucionar este tipo de fallos.

En el capítulo dos, se plasman todos los conceptos, técnicas y tecnologías utilizadas en el diseño e implementación del prototipo, se dan detalles de algunos temas en específico y de cómo estos se relacionaron con el contenido y desarrollo del proyecto.

Por su parte, en el capítulo tres, se presenta el proceso de selección del sistema embebido, se muestran los criterios para escoger la placa sbc raspberry pi 3B, y el proceso de instalación, configuración e implementación del software NUT (Networks ups Tools). Este software permitió la visualización de los parámetros eléctricos de la UPS a través del uso de la consola. Adicionalmente se indica en el mismo capítulo el proceso de la creación de la base de datos utilizando la herramienta InfluxDB, para el almacenamiento de los datos obtenidos de la UPS.

En el capítulo cuatro, se muestra la implementación del sistema de instrumentación , el cual consistió en la integración de la placa arduino nano en conjunto con 6 sensores que se encargaron de la medición de los parámetros

eléctricos y del entorno, estos datos fueron trasmitidos vía puerto serie a la Raspberry pi.

En el capítulo cinco, se podrá observar la implementación del sistema de monitoreo y notificaciones, se hace mención de los servicios ofrecidos por el software NUT entre los cuales se menciona; NUT Monitor, el cual se utilizó en el entorno de escritorio del sistema operativo de la Raspberry PI, NUT CGI que permitió monitoreo Web utilizando el navegador de internet, y Adicionalmente el servicio de notificaciones de NUT que permitió conocer el estado de operación o alarmas presentes en el ups.

Si bien el software NUT permitió varios servicios de monitoreo, el análisis de los resultados muestra su limitación en otras funciones destinadas al almacenamiento o registro de los parámetros, su visualización gráfica en líneas de tiempo y el monitoreo remoto a través de IoT. Lograr añadir estas funciones implico un proceso de implementación y desarrollo en el que se pudo integrar todas las características que el prototipo requería. Se hizo necesario en una forma básica la utilización de scripts en lenguajes de programación como Python y php, y adicionalmente fue necesaria la creación de una base de datos en INflux DB y utilizar otras herramientas de monitoreo como los software Munin, Grafana, Thingspeak, y hasta la función de acceso remoto del programa VNC.

En el capítulo seis se presentan las conclusiones donde el desarrollo del proyecto logró evidenciar el diseño e implementación de un sistema integrado de monitoreo, que no solo posee los beneficios de los diferentes tipos de monitoreo mencionados, sino que también permitió solucionar algunos problemas que presentan las tarjetas de red suministradas por el fabricante a un costo de adquisición mucho más bajo.

1.1. Estado del arte

En el trabajo realizado por Lidong Fu y Bin Zhang(2011), se presenta un modelo de control y monitoreo de una UPS compatible con el protocolo megatec, su prototipo tenia como objetivo principal lograr la gestión y el control de dispositivos remotos a través de Internet. Para lograr dicho objetivo les fue necesario establecer un proceso de comunicación entre el navegador, el servidor web, y el servidor del dispositivo. Diseñaron un servidor web e incorporaron un software de monitoreo de UPS. El sistema se baso en un microcontrolador ATmega 161, una memoria flash AT45DB041B un controlador de Ethernet y un chip de reloj HT1380. Una vez integrado el sistema permitió la supervisión remota del UPS mediante un navegador web, y a su vez la recepción de comandos reconocibles por la UPS, permitiendo así gestionar procesos de encendido o apagado de forma remota [6].

Del anterior proyecto se puede concluir que es posible monitorear a través de Internet una UPS que carezca de forma nativa de esta conexión, siempre y cuando disponga de algún protocolo de comunicación local, para ello se puede implementar una tarjeta de red junto a un microcontrolador y otros accesorios, habilitando la gestión del equipo remotamente con resultados favorables.

Por su parte Y. Sannan Y W. Shaoxu(2011) describen la medición de parámetros como temperatura, flujo luminoso, corriente alterna y continua y voltaje alterno y continuo utilizando el procesador ARM920 S3C2410x de Samsung y el sistema operativo Linux. Es posible decir que su metodología se dividió en tres partes, la primera parte se centró en el diseño general del hardware, el cual no solo debía medir los parámetros mencionados si no almacenarlos y visualizarlos remotamente a través de la red. Una segunda parte de la metodología se basó en la medición de los parámetros, el proceso de registro de las señales entregadas por cada sensor y su conexion con los puertos IO del microprocesador. Por ultimo se encuentra el diseño de software el cual se baso en el sistema operativo Linux. El resultado principal de este proyecto, según lo descrito por los autores, es que los usuarios pueden acceder al sistema de forma local y remota, pudiendo a su vez configurar

limites de tolerancia en los parámetros seleccionados[7].

De este proyecto es posible resaltar las bondades que brinda un sistema embebido al utilizarse para el monitoreo remoto de variables físicas, ya que en su mayoría poseen amplia cantidad de puertos y periféricos para registrar señales digitales y analógicas, y a su vez facilitan la conexión a la red, convirtiéndose en sistemas integrados multifuncionales con funciones configurables y un buen rendimiento.

P. Alqinsi, I. J. Matheus Edward, N. Ismail y W. Darmalaksana, diseñaron un sistema de monitoreo de UPS basado en Internet de las cosas usando el protocolo MTTQ. En su implementación utilizaron una diversidad de sensores, una placa Arduino y una Raspberry Pi , los sensores fueron conestados al modulo Arduino, el Arduino se conectó a un shield de Ethernet y la Rasberry Pi funcionó como servidor web, MQTT Broker y servidor de base de datos. El sistema enviaba datos continuamente desde el Arduino hacia el MQTT Broker que se instalo en la Raspberry, y estos podían ser vistos desde una página web [8].

Del anterior proyecto se puede concluir que el protocolo MTTQ es una herramienta liviana y a la vez potente, se puede utilizar para monitorear varios parámetros en distintos dispositivos y posee la facilidad de ser instalado en algunos dispositivos embebidos tales como una Raspberri Pi, abriendo un campo a distintas aplicaciones de monitoreo basadas en iot.

De igual manera T. Adiono, M. Y. Fathany, S. Fuada, I. G. Purwanda and S. F. Anindya realizaron una investigación orientada al diseño, implementación y evaluación de un sistema de monitoreo de condiciones ambientales mediante un un sensor de humedad y temperatura. Para el prototipo utilizaron el sensor de humedad y temperatura DHT11, un microcontrolador STM32L100 un modulo de comunicación Zigbee, una batería recargable BL-5c y un circuito cargador, adicional a esto utilizaron una Raspberry Pi como host. El sistema funciona conectando el sensor al microcontrolador, los datos se trasmiten por el modulo zigbee a la Rasberry Pi, y la Raspberry Pi se conecta vía bluetooth a una aplicación android de monitoreo.

Realizaron pruebas de conectividad, medición de uso de energía, y duración de la batería, todas con resultados satisfactorios[9].

De este proyecto y el anterior, se puede concluir que la combinación de microcontroladores y Raspberry Pi, es una alternativa adecuada cuando se desea registrar señales y monitorearlas a través de diversas plataformas, bien sea via web o mediante aplicaciones android, adicionalmente utiliza poca energía lo cual lo hace ideal para aplicaciones IoT de bajo consumo.

Por último S. Balamurugan Y D. Saravanakamalam diseñaron un sistema de gestión y seguimiento energético, mediante una plataforma Arduino un modulo Wi-Fi, reles y LCD, el objetivo de su proyecto era lograr la medición del consumo de energía en los hogares y almacenar los datos en la nube, para que de esta forma los usuarios pudiesen monitorear sus consumos de energía en tiempo real y realizar ahorro apagando sus cargas [10].

De este proyecto se resalta el poder de los sistemas embebidos no solo en actividades de monitoreo si no también en control y/o automatización, permitiendo operar cargas remotamente y contribuir a un uso mas eficiente de la energía.

1.2. Planteamiento del Problema

Actualmente la empresa UPSISTEMAS SAS, requiere el monitoreo de las UPS Eaton serie DX, y aunque el fabricante de las UPS ha presentado soluciones enfocadas al monitoreo, estas soluciones no dejan de presentar problemas al momento de su implementación.

Si bien los fabricantes de UPS han trabajado y suministrado tarjetas para el monitoreo de parámetros de forma remota y/o local, sus alterativas no dejan de tener inconvenientes, limitaciones o desventajas, a continuación se describen de manera general las mas importantes:

El primer inconveniente es que todas estas tarjetas se adquieren como accesorios, lo cual aumenta los costos y disminuye el margen de ganancias del proveedor.

El segundo inconveniente que se encuentra presente en todas las tarjetas de monitoreo (exceptuando el sensor ambiental), radica en que la lectura de los parámetros eléctricos se realiza sobre los datos obtenidos por los sensores internos que posee la UPS, por lo cual no existe una medición directa fuera de la UPS en tiempo real que garantice que esos parámetros son los correctos y coincidan con los entregados al sistema.

En tercer lugar no es posible encontrar una alterativa unificada que integre los beneficios que brinda cada tarjeta de monitoreo, por lo cual adquirir estas tarjetas sigue siendo una inversión que soluciona problemas parcialmente pero no en su totalidad.

Para aclarar los diversos aspectos, y centrando el problema en las UPS Eaton serie DX, a continuación se presenta de manera especifica las desventajas que presenta cada tipo de monitoreo.

- **Monitoreo local utilizando una conexión punto a punto:** La desventaja principal de este sistema es que requiere de un computador dedicado a esta función, otro inconveniente radica en que no hay forma

de que el usuario se entere en tiempo real de los cambios en el estado de operación que posea el UPS sin que se encuentre cerca al computador donde se tiene instalado el software de monitoreo, es decir carece de sistema de notificaciones remotas.

- **Monitoreo implementando una tarjeta de red snmp II:** Si bien esta tarjeta es una buena opción de monitoreo no deja de tener sus desventajas siendo la principal el alto costo de la misma si la comparamos con el precio de la UPS. La segunda desventaja que poseen estas tarjetas, se presenta a nivel de hardware, ya que posee poca capacidad de almacenamiento local, lo cual se ha visto evidenciado en equipos que al perder la conexión a internet se pierde parte de la información registrada, ya que el almacenamiento local es tan poco que en ocasiones no permite identificar los eventos que ocurrieron durante la falla de conexión. La tercera desventaja se presenta en la forma de visualización de los registros de los parámetros eléctricos, estos se presentan en lineas de texto tipo txt y no se pueden visualizar en forma gráfica.

- **Monitoreo con sensor ambiental para NMC de EATON:** La primera desventaja de este monitoreo, radica en que obligatoriamente se necesita adquirir una tarjeta de red, ya que el sistema no funciona de forma autónoma si no como un accesorio que se adapta a la tarjeta snmp. La segunda desventaja es que solo incorpora dos entradas para monitorear el estado de las puertas o ventanas abiertas, siendo a veces necesario mas puntos de monitoreo. La tercera desventaja radica en que al igual que ocurre con la tarjeta de red, los parámetros registrados no se pueden visualizar de forma gráfica. La cuarta limitación se evidencia en el momento en el que ocurre un fallo en la conexión de internet, ya que automáticamente se pierde el monitoreo remoto y el sistema carece de un panel de aviso o notificaciones de forma local.

- **Monitoreo con tarjeta Relay Card MS:** La desventaja de este sistema radica en que ocupa el único slot que posee la UPS y no se puede usar en conjunto con una tarjeta de red o el sensor de temperatura, lo cual implica no poder monitorear las condiciones climáticas

del sitio y no poder enviar notificaciones por correo ó implementar monitoreo remoto. Tampoco permite monitorear las variables eléctricas presentes en el sitio.

- **Panel o dispositivo de monitoreo remoto:** Su desventaja radica en que funciona como un accesorio de la tarjetas Relay Card MS por lo cual es necesario adquirir la tarjeta y el panel simultáneamente, eso implica mayores costos, ademas se sigue teniendo el inconveniente de no poder monitorear las condiciones climáticas del sitio y no poder enviar notificaciones por correo o implementar monitoreo remoto.

Con base en los problemas descritos anteriormente , y tomando en cuenta los conocimientos adquiridos en el estudio y campo de la ingeniería electrónica, se plantea la siguiente pregunta ¿Es posible implementar un dispositivo embebido para reemplazar los sistemas de monitoreo actuales, que integre sus beneficios y reduzca los costos económicos para la empresa UPSISTEMAS SAS?

1.3. Justificación

El sistema embebido planteado en el presente proyecto integra las mejores cualidades de los diferentes sistemas de monitoreo descritos previamente y busca dar solución a cada uno de los problemas conocidos.

Al diseñar el prototipo se tuvo como objetivo construir un dispositivo que permitiese recopilar los parámetros nativos suministrados por la UPS, e integrar estos con la información exógena proveniente de diversos sensores.

El dispositivo debía permitir la visualización de información de forma gráfica para facilitar su uso por parte de personal no calificado, y adicionalmente debía ser económico en comparación con los sistemas tradicionales para permitir la reducción de costos de monitoreo en la empresa UPSISTEMAS SAS.

El uso de la Raspberry Pi en combinación con un Arduino Nano, diversos sensores, y la plataforma Grafana, permitió conseguir los resultados propuestos, demostrando así la utilidad de los sistemas embebidos en la solución de problemáticas reales de la industria colombiana.

1.4. Objetivos

1.4.1. Objetivo General

Diseñar e implementar un sistema de monitoreo remoto y local de los parámetros eléctricos, temperatura y humedad, en las UPS EATON serie DX, mediante un dispositivo embebido.

1.4.2. Objetivos específicos

- Seleccionar una topología o arquitectura de sistema embebido que permita el monitoreo de las UPS de la marca Eaton serie Dx.
- Implementar, mediante el dispositivo embebido seleccionado, un sistema de registro de los parámetros suministrados por la UPS.
- Diseñar e implementar un dispositivo electrónico para el registro de las variables eléctricas (voltaje y corriente) y del entorno (temperatura, humedad) presentes en el área de operación del UPS.
- Implementar, mediante el sistema embebido seleccionado, un sistema de notificación y monitoreo, remoto y local, de los diferentes parámetros y variables registradas.

2. Marco Teórico

Este trabajo se centró en el diseño e implementación de un sistema de monitoreo basado en un dispositivo embebido, a continuación vamos a definir todas las técnicas conceptos y tecnologías utilizadas en la ejecución de este proyecto.

2.1. UPS o SAI

Un SAI (Sistema de Alimentación Ininterrumpida), o UPS por sus siglas en ingles (Uninterruptible Power Supply) es un dispositivo conformado por un banco de baterías las cuales son las encargadas de proporcionar la energía eléctrica necesaria para el funcionamiento de un equipo, siguiendo dentro de la UPS tenemos un controlador de carga que cumple la tarea de supervisar y gestionar la carga de las baterías que componen el dispositivo, y por ultimo quien se encarga de realizar la conversión de voltaje DC para la energía eléctrica tradicional en AC, esta labor la realiza un inversor.

Muchos de estos dispositivos son coincidencialmente, equipos que estrictamente requieren de un voltaje regulado para la correcta operación de sus componentes. Una de las tareas de una UPS, es que proporciona mediante sus sistema, una alimentación regulada y continua, sea operando con la red eléctrica externa o con su banco de baterías, permitiendo el funcionamiento correcto de los equipos que estén conectados a ella [11].

2.2. Sistemas embebidos

Cuando hablamos de sistemas embebidos nos referimos a dispositivos con aplicación práctica de la electrónica digital, estos, en su mayoría, se ca-

racterizan por realizar procesos de computo en determinados periodos de tiempos o de una forma continua una vez se inicia un proceso. También realizan específicas, dependiendo de su construcción poseerán funciones limitadas, estos ejecutan algoritmos, procesos informáticos, procesamiento digital de señales etc. pero no se puede esperar de ellos el mismo poder computacional de un PC de escritorio [12].

Sin embargo con el paso de los años, han ido evolucionando, mejorando en su diseño, añadiendo mayor desempeño de hardware e implementación de software, lo cual le ha permitido interactuar cada vez más con un mayor número de accesorios, es común que requiera en su implementación hardware analógico, ya sean periféricos como por ejemplos pantallas o teclados (utilizados en las placas sbc), o la utilización de sensores para luego convertir señales eléctricas en señales digitales. Por otro lado, se debe mencionar la ampliación que han venido sufriendo referente a sus velocidades de procesamiento.

Existen diseños en los que el sistemas embebidos ya no solo ejecutan un código de instrucciones como lo hacían los antiguos microcontroladores, estos embebidos operan con un sistema operativo portable, el cual le añade mejoras, nuevas tareas y funcionalidades adicionales, sin necesidad de reemplazar el hardware del equipo [13].

Adicionalmente, estos sistemas son energéticamente eficientes , lo cual los ha hecho útiles en distintas áreas de trabajo o uso que requieren que un dispositivo sea portable o trabaje con baterías, los anteriores campos de aplicación han disparado su uso masivamente, Este amplio uso ha puesto a prueba, de manera empírica; su capacidad tanto tangible como intangible [14].

2.2.1. Raspberry Pi

Este es un dispositivo electrónico que pertenece a la familia de ordenadores de placa reducida o SBC (Single Board Computer) aunque también se le atribuye en algunas literaturas como un sistema llamado SOC (System On

Chip). Al ser prácticamente un computador de dimensiones reducidas, este requiere de un sistema operativo para funcionar, por lo general este sistema operativo está basado en Linux, portado a arquitectura ARM, pero también nos podemos encontrar con un sistema operativo de Microsoft llamado Windows 10 IoT core.

Esta placa fue construida inicialmente con fines académicos, con el propósito de enseñar el uso de las herramientas informáticas en países emergentes, sin embargo dada la funcionalidad y los proyectos que se podían realizar con este dispositivo, tuvo un impacto favorable entre las comunidades de programadores y ambientes académicos, siendo una de las placas más utilizadas para el desarrollo de infinidades de proyectos.

La Raspberry Pi posee varios modelos de placa, y en cuanto a características con respecto al hardware de la raspberry pi específicamente la 3b, podemos decir que incorpora puertos USB que permiten conectarse con diversos dispositivos actualmente consolidados en el mercado, dado que el puerto USB ha sido estandarizado como puerto de comunicación e interconexión de diferentes dispositivos, posee también conector RJ45, puerto HDMI, salida de audio, pines GPIO, conexiones inalámbricas por WiFi y bluetooth.

Una de sus características más interesante es su precio , también se añaden las cualidades de su bajo consumo energético, si bien esta placa posee varias ventajas, no podemos pretender obtener de ella el mismo poder computacional que un PC de escritorio, sumado a esto se encuentra el hecho de no poseer disco duro por lo cual en su reemplazo se utiliza tarjeta micro SD en la que se carga el sistema operativo, si bien esto parece una desventaja, también resulta siendo una ventaja útil ya que permite la copia de la imagen ISO, la cual se puede instalar en otras Raspberry pi sin necesidad de volver a repetir todo el proceso de instalación y configuración [15][16][17].

2.2.2. Arduino nano

Es una de las versiones de arduino que contiene el controlador Atmega 328, en la cual podemos desarrollar cualquier tipo de proyecto electrónico, don-

de sea una exigencia el bajo consumo y el tamaño, pero en nuestro caso, viene siendo el costo del dispositivo en relación con las tareas que este va a realizar y la gran ventaja de ser Open-source.

Este dispositivo se programa de igual manera que las demás placas en cuanto al entorno de desarrollo, pero esta basado en la placa de Arduino UNO, pero con algunas características técnicas importantes.

Dentro de las características importantes están las de poder tomar las señales analógicas y digitales, realizar mediante librerías y código fuente, la interpretación de estas variables [18].

2.3. Sistemas Operativos (OS)

Podemos definir los sistemas operativos como un conjunto de herramientas con tareas específicas o aleatorias, algunas dependientes, otras autónomas, distribuidas jerárquicamente, cuya función principal es la de ejecutar la funcionalidad práctica de la unidad central de procesos (CPU). Un sistema operativo se encarga de la ejecución del sistema de archivos, atiende todos los requerimientos de las aplicaciones y vigila la correcta ejecución de los programas , verifica que estos trabajen coordinadamente y que interactúen entre si de una forma organizada, gestiona el uso eficiente de los recursos y que estos respondan según sea su necesidad o funcionalidad. En su gestión de la CPU, el sistema operativo se encarga de la interacción de todos los componentes del hardware tanto internos como externos, y se encarga también del reconocimiento o uso de los driver y sus periféricos.

Un sistema operativo siempre busca un aprovechamiento óptimo de la memoria, este uso de la memoria por lo general va destinado al almacenamiento de datos, y al uso de los recursos del sistema, proporcionando un camino óptimo que garantice una comunicación acertada y sin interrupciones.

Dentro de los aspectos importantes de un sistema operativo se encuentra su nivel de interacción con el usuario final, teniendo en cuenta que el SO

se encargara de manejar los datos del usuario, debe brindar la protección de los datos utilizando los mecanismos destinados para dicho fin, como por ejemplo el uso de usuarios o contraseñas.

Todo sistema operativo en su esencia, se encontrara compuesto o estructurado por un núcleo o Kernel quien no solo se encarga de ejecutar todos los procesos de computo sino de vigilar que la ejecución de estos sean soportados por el sistema [19].

2.4. Sistemas Operativos Embebidos

Un sistema embebido no solo se compone de un hardware físico, también requiere de un conjunto de instrucciones escritas en determinado lenguaje de programación, el cual hará posible la utilidad del hardware, específicamente estamos hablando de un sistema operativo. En el sistema operativo se instalara un software de soporte que permitirá la gestión de determinado equipo por ejemplo en el caso de este proyecto, el software NUT.

Adicionalmente va arraigado una aplicación que interpreta las funciones del dispositivo que posteriormente serán interpretadas por el software, podría ser un driver por ejemplo.

Actualmente existen muchos aplicativos de los instrumentos los cual se encargan de interpretar las funciones del hardware, para que sean leídas o reconocidas por el sistema operativo, por lo general estos aplicativos son específicos [20].

2.4.1. Raspberry Pi OS

El sistema operativo portable llamado Raspberry pi os (anteriormente Raspbian) el cual es una distribución basada en el famoso sistema operativo de Linux llamada Debian. Este sistema operativo tiene la facultad de realizar múltiples funciones y esta disponible en una versión litte, también existe

una versión completa con interfaz gráfica y con una cantidad de programas predeterminados instalados [21].

2.5. Monitoreo de Red

En términos generales cuando hablamos de monitoreo de red nos referimos al uso de sistemas que supervisan constantemente redes computacionales. La función principal del monitoreo de red es la de informar y notificar cualquier falla, anomalía o comportamiento anormal que no este incluido dentro de la parametrización establecida en su diseño, y que pueda representar una falla en cualquier componente a nivel de hardware o software[22].

Para realizar este monitoreo, es necesario utilizar herramientas como software, programas, script etc, que han sido diseñados para esta función.

2.5.1. Herramienta de monitoreo NUT

Es un software de código abierto y libre cuya función principal es brindar soporte de monitoreo a distintas topologías de equipos energéticos como UPS, PDU, STS, et.c[23] de distintas marcas. Sobre las referencias de NUT podemos encontrar un artículo de la IEEEE donde se le hace mención exponiendo su utilidad[24].

Muchas marcas y modelos son compatibles y expuestos a través de un protocolo de red y una interfaz estandarizada.

- **Controlador:** Cuando hablamos de controlador o driver nos referimos a la herramienta informática que se encarga de tomar las señales o datos enviadas por determinado hardware, interpretarlas, decodificarlas y trasmitirlas a un lenguaje que pueda ser entendido por un sistema operativo, arquitectura o programa; Gracias al anterior proceso el driver realiza una labor de reconocimiento del hardware, logra establecer una comunicación e interpretación de datos y permite enviar señales o demás acciones según su funcionalidad o necesidad[25].

- **Controlador para equipo UPS:** El controlador o driver es aquel que permite a la herramienta de monitoreo comunicarse con la UPS y solicitar datos o enviar comandos[26].

2.5.2. Winpower

Es un software de monitoreo recomendado por el fabricante EATON [27], el cual permite el monitoreo de un UPS implementando una interfaz gráfica minimalista pero fácil de interpretar, la cual integra una función de apagado programado ante un corte del fluido eléctrico, a su vez permite a cualquier usuario dentro de su red de área local monitorear la UPS.

2.5.3. Grafana

Grafana es un software de licencia libre u open-source, se encuentra disponible para los sistemas operativos principales, de su funcionamiento se puede destacar que permite almacenar o registra los datos provenientes de otras fuentes para luego poder visualización mediantes distintos gráficos configurables, además permite configurar alarmas y notificaciones[28].

2.6. Variables eléctricas y del entorno

Las variables eléctricas se definen como valores o magnitudes que cambian con el tiempo, los cuales a su vez están relacionados con los fenómenos de la electricidad, tales como el voltaje, intensidad eléctrica y potencia. En las variables o magnitudes físicas del entorno encontramos la temperatura y humedad relativa.

2.6.1. Sensores

Los sensores son instrumentos que pueden obtener las magnitudes físicas o químicas del entorno y procesarlas o transformarlas en señales eléctricas, de manera que puedan se interpretadas por los diferentes elementos electrónicos como son los embebidos. Los sensores poseen un rango de medición y

precisión de la medición que realizan, por tanto tienen diferentes aplicaciones de acuerdo a sus especificaciones. Entre ellos encontramos gran variedad, pero centrándonos en los que utilizaremos en este proyecto, se mencionan los siguientes sensores:

- **Sensor de Voltaje AC:** Hace referencia a un modulo que realiza la lectura del voltaje en alterna y entrega una medición que es posible leerla por el embebido que se utilice en gestión.
- **Sensor de intensidad eléctrica** Este sensor permite la medición de la intensidad de corriente eléctrica que fluye por un conductor, lo que es posibilita saber con exactitud cual es el consumo en intensidad y mediante ecuaciones saber su consumo en potencia. Existen actualmente en el mercado dos tipos de sensores que posibilitan la medición de esta magnitud:
  - **Sensor de intensidad invasivo:** Este sensor permite la medición de la intensidad que fluye por un circuito o conductor. Para la implementación de este sensor, es necesario abrir el circuito para realizar una conexión en serie y de esta manera su medición.
  - **Sensor de intensidad no invasivo:** Este sensor permite la medición de la intensidad que fluye por un circuito o conductor pero, a diferencia de su antecesor, permite la medición de la magnitud sin la necesidad de abrir el circuito, utilizando el principio de inducción electromagnética.
- **Sensor de humedad relativa:** Este sensor permite la medición de la humedad en el ambiente, es muy utilizado en las automatizaciones donde se emplean sistemas de riego, en la agroindustria y estaciones meteorológicas[29].
- **Sensor de temperatura:** Este sensor detecta las variaciones de la temperatura en el entorno o en un determinado objeto y las transforma en señales eléctricas que son interpretadas por los embebidos o controladores que utilizan esta señal para determinada tarea[30].

2.7. IoT: Internet de las cosas

Representa a la tecnología que incorpora en su sistema de conexión a internet, objetos, información personal y no personal, datos, comportamiento de usuarios finales, datos de sensores etc. con los cuales de acuerdo a la analítica de esta información, es posible realizar predicciones y acciones tangibles en el mundo real [31].

3. Selección e Implementación de embebido

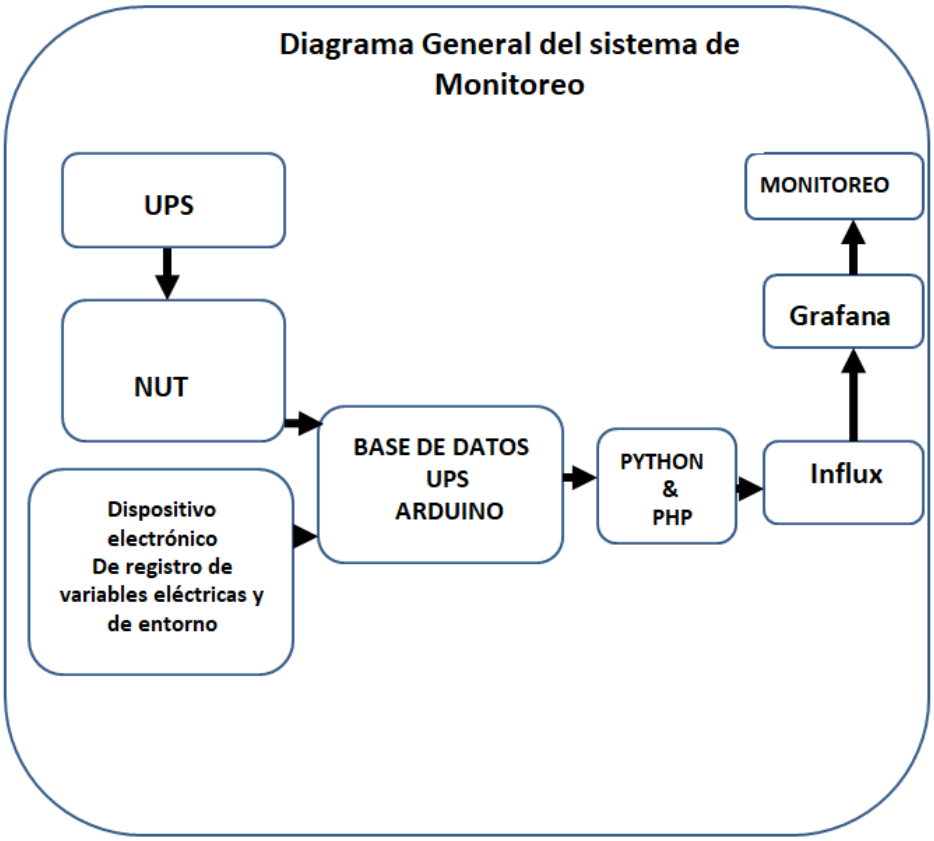

Figura 3-1.: Diagrama general de sistema de monitoreo

3.1. Selección de embebido

De acuerdo a los diversos dispositivos embebidos que se encuentran en el mercado, se hace un estudio previo con el cual se permite dar a conocer, cuáles son las características de los principales dispositivos embebidos, que pueden hacer parte del presente trabajo, dando por conclusión la selección de uno de ellos para culminar con el desarrollo del mismo.

A continuación, se describen a grandes rasgos mediante una tabla de comparación, las más importantes placas de dispositivos embebidos actualmente en el mercado.

Embebido	CPU	Velocidad Procesamiento	Memoria RAM	USB	Video	Ethernet	Precio (US)	Incluye
Intel Galileo	Quark SoC X1000	400Mhz	2GB	USB 2.0	N/A	Ethernet	$ 186	Solo embebido
Coral by Google	NXP i.MX 8M SOC	1,5Ghz	1GB	USB Tipo-C	HDMI	Ethernet	$ 150	Solo embebido
BeagleBone	Cortex-A8	1Ghz	512MB	USB 2.0 miniUSB 2.0	micro HDMI	N/A	$ 85	Solo embebido
ASUS Tinker	Quadcore ARM	1.8Ghz	2GB	USB 2.0	HDMI	Gbit LAN	$ 78	Incluye case, cargador,
pcDuino	Cortex-A7 Dualcore	1Ghz	1GB	USB 2.0	HDMI	Ethernet	$71.68	Solo embebido
RaspBerry pi 3	Broadcom 2837 - Cortex A-53	1.2Ghz	1GB	USB 2.0 miniUSB 2.0	HDMI	Ethernet	$49.99	Incluye case, cargador.
Arduino Nano	ATmega328	16MHz	32Kb	microUSB	N/A	N/A	$4	Embebido y cable USB

Figura 3-2.: Tabla de comparación de los diferentes embebidos

Para la selección del dispositivo embebido a utilizar en el proyecto, se toman las siguientes características que pueden direccionar al que se ajuste a las condiciones de trabajo. Una de las condiciones es que debe soportar el montaje de un sistema operativo de código abierto basado en Linux, otra de las características en las que se enfoca este documento, es la alta probabilidad que un cliente acceda al servicio por un bajo costo, en comparación con las tarjetas de red nativas. Iniciando el manejo de la herramienta Grafana para el monitoreo de las variables obtenidas de la UPS, la cual es una herramienta que no solicita muchos requisitos de sistema para funcionar, se toma en selección a la RaspBerry Pi para el desarrollo del proyecto en gestión. Además de su costo, con respecto a los demás dispositivos embebidos, reúne las características capaces de colocar en marcha el funcionamiento del mismo, soporta una plataforma basada en Linux, como la común distro de Raspbian, que la misma página de la raspberry la recomienda. Teniendo este sistema operativo, podemos desde ahí, montar Grafana y NUT para realizar el desarrollo del proyecto para la adquisición, almacenamiento y monitoreo de datos, generados por la UPS y la instrumentación implementada.

Otra característica que orienta a la selección de la Raspberry pi, es el bajo costo de adquisición, comparado con los demás dispositivos embebidos, que tiene muchas ventajas en cuanto potencia de hardware, en los cuales es posible hasta trabajar con IA, pero esto implica que tengan un alto costo, y no es lo que que se pretende.

En conclusión, se utilizará el dispositivo embebido RaspBerry Pi para el desarrollo de este proyecto.

3.2. Sistema de registro de parámetros del UPS

En este proyecto se seleccionó la raspberry pi como el dispositivo embebido encargado de la comunicación con el UPS y de la visualización y registro de sus parámetros, esta segunda fase de desarrollo implico una metodología

que se dividió en tres etapas, las cuales se unifican en un proceso de implementación y optimización logrando así el segundo objetivo del proyecto. El anterior proceso lo conceptualizamos en el siguiente diagrama de bloques:

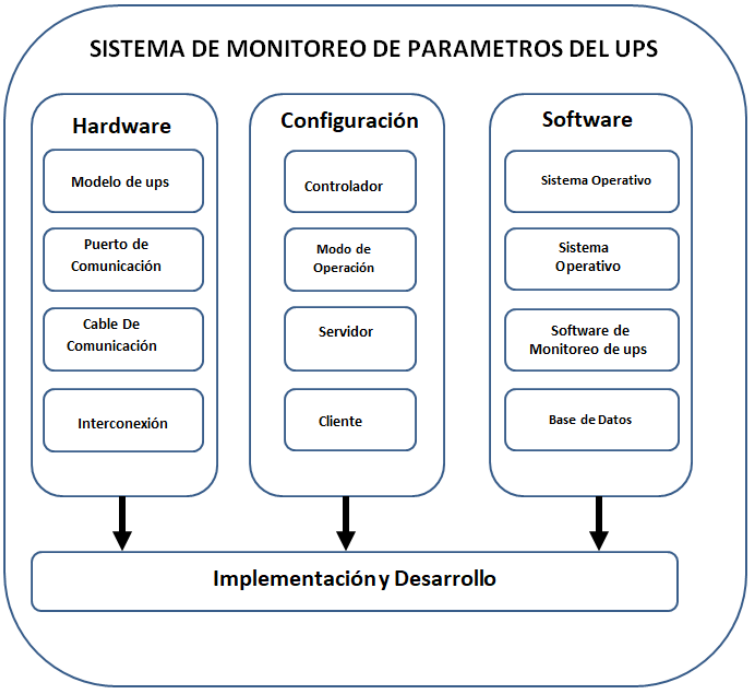

Figura 3-3.: Diagrama Sistema de registro de parámetros

3.3. Identificación de modelo de UPS

El software implementado en este prototipo brinda soporte a más de 1000 modelos de UPS, de más de 140 fabricantes, fue importante determinar si el modelo que utilizamos se encontraba dentro de la lista de equipos soportados, ya que de esto depende otras variables que explicaremos más adelante Y sin las cuales no hubiese sido posible la ejecución del proyecto. El modelo de ups que se utilizó corresponde a la marca EATON serie DX de 1.5kva con soporte del software del monitoreo[32].

Figura 3-4.: UPS

3.4. Puertos de comunicación

Para poder lograr la extracción de los datos entregador por el ups, fue necesario identificar los puertos de comunicación que posee el equipo. Este proceso de identificación fue importante y gracias a él, determinamos el protocolo de comunicación utilizado. La siguiente foto pertenece a la parte posterior del ups, en ella se pueden visualizar los dos puertos de comunicación que posee el equipo, uno de los puertos corresponde a una conexión RS-232 y la otra a un slot para insertar las tarjetas suministradas por el fabricante identificado como Intelligent- Slot.

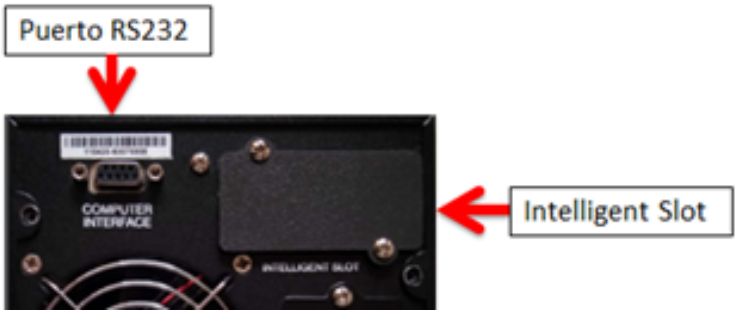

Figura 3-5.: Puertos de comunicación de UPS

3.5. Cable de comunicación

Una UPS puede tener un puerto similar, pero esto no significa que ambas utilicen el mismo tipo de cable, por esto es importante como indicamos anteriormente conocer el modelo de UPS y el fabricante, al igual que el protocolo de comunicación, apoyándonos en estos datos y en la guía de NUT implementamos un cable de conexión serial estándar[33].

Como la conexión la realizamos en al Raspberry pi, la cual no posee conector físico RS 232, se hizo necesario adquirir un cable conversor de puerto serie RS232 a puerto USB.

3.6. Tipo de interconexión

De una manera estándar, el tipo de interconexión o Modo, hace referencia a la conexión física que existe entre el ups y el servidor o equipo que la monitorea, esta interconexión presenta varias variantes, a continuación explicaremos cómo es cada tipo de interconexión, detallando al final cual se implementó en el proyecto.

3.6.1. Interconexión simple

Es la configuración más común, consiste en unos ups y una computadora, la ups es monitoreada por la computadora y la computadora recibe suministro eléctrico de la ups. El sistema se puede configurar para que, en el momento de ocurrir un corte en el suministro eléctrico, el computador cierre todos los programas y se apague automáticamente, esto evita apagados forzosos y perdida de información, por otro lado incrementa la vida útil de las baterías.

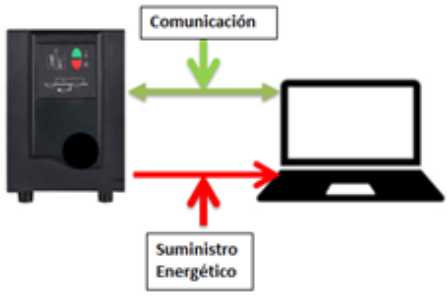

Figura 3-6.: Interconexión simple

3.6.2. Interconexión avanzada

Consiste en una ups que alimenta enérgicamente a varias computadoras, de esas computadoras solo una establece comunicación serial con el ups, esta computadora se llama maestro, los otros computadores no tienen comunicación serial con el ups, a estos se les llama esclavos, pero si se interconectan entre ellas y el computador maestro a través de la red LAN.

Una vez establecida la conexión física se configuran los archivos para determinar quién es el usuario maestro, y quien es el esclavo, al momento de ocurrir un fallo de energía, la configuración estándar cierra los programas y apaga los computadores en un orden programado, siendo el usuario maestro el ultimo.

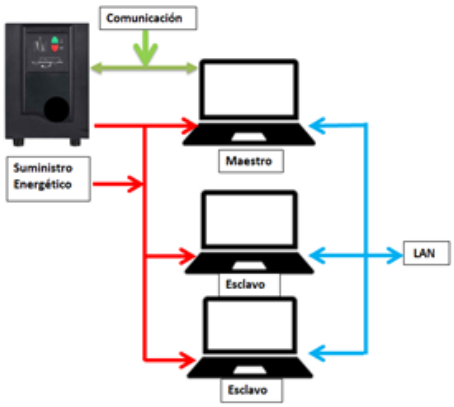

Figura 3-7.: Interconexión Avanzada

3.6.3. Interconexión "Big Box

Esta interconexión es común en servidores que se alimentan de doble fuente, donde dos ups alimentan enérgicamente al servidor y ambas UPS se comunican con el mediante el puerto serie. El programa ejecuta dos drivers (uno por cada UPS), pero maneja un solo proceso de apagado.

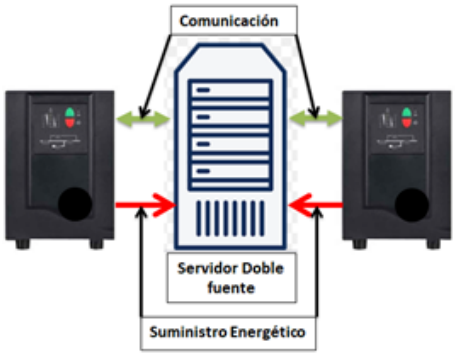

Figura 3-8.: Interconexión Big Box

3.6.4. Interconexión Bizarre

Podríamos decir que es un sistema mixto, Esta configuración tiene la particularidad de que pueden haber un UPS comunicándose con un computador al que no está alimentando, esto se debe posiblemente a que ambos UPS se encuentren en la misma locación y la facilidad de conexión se da sobre un único computador, en este caso el computador donde se conectan ambos UPS es un maestro para el UPS 1, pero respecto al UPS 2 solo se monitorea, los computadores conectados al UPS 2 solo son esclavos del UPS 2.

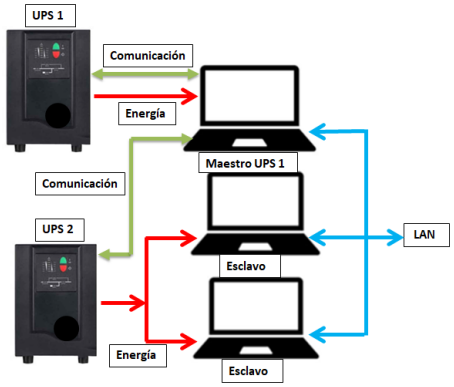

Figura 3-9.: Interconexión Bizarre

3.7. Interconexión implementada

Anteriormente ya habíamos visto los distintos tipos de interconexión de hardware que se pueden realizar para ejecutar el software NUT. Si bien es claro que el propósito general de este software es detectar el fallo en el suministro energético y realizar un apagado programado de los servidores y o computadores donde se encuentre instalado, hacemos la claridad de que en este proyecto se utilizó el software solo con el fin de establecer comunicación entre la raspberry y el UPS y poder extraer sus datos para monitoreo. La interconexión que más se acomodó a nuestro proyecto es la configuración simple, se reemplazó el computador o servidor por la placa SBC raspberry

pi y se ejecutó el software nut sobre esta, sin contemplar un apagado programado, adicional se implementó suministro eléctrico de la UPS de una fuente externa y no de la salida del UPS.

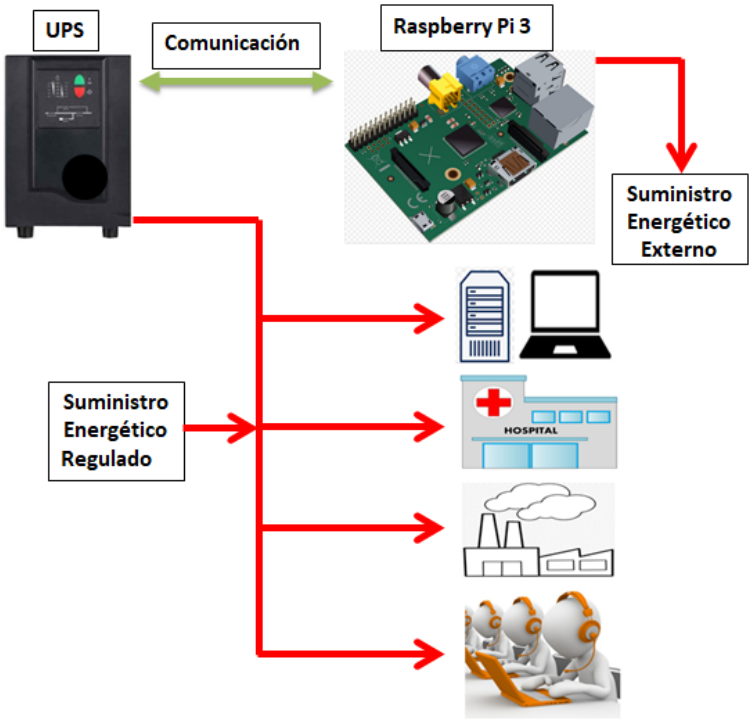

Figura 3-10.: Interconexión Implementada

3.8. Selección de sistema operativo

Dado que el hardware implementado en este prototipo fue la placa SBC Raspberry pi 3, se hizo necesaria la instalación de un sistema operativo sin el cual la placa no tendría ninguna utilidad. En la raspberry pi se instaló el sistema operativo Raspberry pi os. (Anteriormente llamado Rasbian). Raspberry pi os no es el único sistema operativo disponible para ser instalado, existen muchos otros sistemas operativos portables los cuales se pueden

encontrar en la página oficial del fabricante[21], también existen distribuciones disponibles en otras páginas para fines específicos. Nuestra razón para escoger este sistema operativo radico en que es el recomendado por el mismo fabricante, lo cual gracias a su amplia comunidad de desarrollo permite actualizaciones, mejoras de estabilidad y mejoras en seguridad, pese al anterior argumento se hicieron pruebas instalando otros sistemas operativos como ubuntu core o linux mint portable, si bien la experiencia en cuanto a interfaz gráfica fue muchísimo mejor, si se notó un leve aumento de temperatura en el hardware del dispositivo y ralentización al momento de ejecutar determinadas funciones o procesos(pruebas de esfuerzo) que si bien no están ligadas a nuestro proyecto, servían de marco de referencia para comparar que sistema operativo nos era más conveniente utilizar. La experiencia anterior fundamento el uso del sistema operativo recomendado por el fabricante. Existen otras versiones de hardware de raspberri pi con más capacidad de memoria y procesamiento que posiblemente ejecutaran los otros sistemas operativos de una forma más estable.

3.8.1. Instalación del sistema operativo

El proceso de instalación del sistema operativo se realizó utilizando un pc externo donde se instaló un quemador de imágenes ISO, la imagen ISO fue descargada desde la página oficial del fabricante y posteriormente se instaló en una tarjeta micro SD, se configuro una conexión ssh y una conexión wi-fi, se deja un documento anexo(instructivo) de guía donde se detalla todo el proceso de instalación.

3.9. Selección del software de monitoreo

Para lograr la extracción de los datos suministrados por el ups sin la necesidad de recurrir a las tarjetas suministradas por el fabricante, se hizo necesaria la implementación de un software de monitoreo que estuviera disponible, sin licencia de pago y de libre uso, las únicas alternativas funcionales que encontramos fueron el software Winpower y el software NUT(Networks UPS Tools). Se realizaron pruebas con ambos software, y al final se ter-

minó escogiendo el software NUT. Las pruebas que se realizaron con cada software se detallan a continuación, al igual que las conclusiones que nos motivaron a escoger a NUT como la mejor alternativa.

3.9.1. Winpower

Este software se encuentra disponible para descarga gratuita en la página del fabricante. Adicional hay otras páginas de internet de donde se puede descargar. Winpower fue probado en un pc con Windows mostrando resultados de conexión favorables y logrando obtener una interfaz gráfica amigable,posee un registro de log de eventos y registro de variables eléctricas, También fue instalado sobre una máquina virtual corriendo Linux, donde se obtuvieron resultados de conexión favorables con las mismas ventajas que se obtuvieron en windows. Se intentó instalar winpower en la Raspberry pi con tres sistemas operativos distintos; Raspberry Pi OS, Linux MINT y ubunto Core, en ninguno se pudo hacer funcionar el software.

3.9.2. NUT (Network ups tools)

NUT, a diferencia de winpower no posee un instalador oficial para Windows, la documentación sugiere un cliente llamado winnut sobre el cual hay poca información. El mayor uso de NUT está destinado para plataformas basadas en Linux, a diferencia de winpower, NUT no es un software que se instala y funciona la primera vez, su implementación y uso requiere de un proceso de configuración que será bastante complejo para quienes no están familiarizados con el sistema operativo linux, una vez comprendidos los comandos de la consola , la secuencia y la forma en cómo se deben configurar los scripts del programa, el proceso de configuración se hace más amigable. Si el software winpower se hubiese podida instalar en la raspberry, , igual la elección del software de monitoreo se encaminaba a escoger NUT y las razones son las siguientes:

1. En primera medida winpower esta encapsulado a monitorear un solo modelo de ups, es posible que presente compatibilidad con otros

modelos de ups, pero sería un proceso que tocaría llevar a experimentación ya que en la página de descarga no entregan muchos detalles y solo se especifica para el ups EATON DX, en cambio NUT, según declara en su página oficial brinda soporte a más de 140 fabricantes y más de 1000 modelos de ups, lo cual lo hace bastante llamativo ya que permite escalar este proyectos a monitorear no solo otro tipo de ups si no otros dispositivos de suministro de energía.

2. Por segunda ventaja tenemos que NUT es un software de código abierto, y de descarga gratuita, se puede utilizar sin pagar una licencia o sin miedo a que caduque en determinado tiempo, al ser de código abierto permite una comunidad de desarrollo encaminada a realizarle mejoras.

3. Como tercera ventaja podemos añadir que admite varios tipos de monitoreo como monitoreo por puerto serial, monitoreo por puerto usb, monitoreo SNMP y monitoreo de red , entre otros. Admite integración a varias plataformas y permite ser portado en dispositivos embebidos o placas SBC además de manejar una arquitectura cliente servidor.

Teniendo en cuenta las anteriores ventajas y sumando la posibilidad de escalar el proyecto a implementarlo para monitorear otros dispositivos de energía, fue que se escogió NUT como el software de monitoreo.

3.10. Instalación del software de monitoreo NUT

Antes de la instalación del software de monitoreo se hizo necesario actualizar el sistema operativo de la Raspberry pi, esto permite no tener problemas de seguridad y también brinda las últimas versiones del programa para garantizar la estabilidad del sistema, luego de este proceso se reinicia la raspberry pi, finalizado el reinicio se procede a ejecutar la instalación del software NUT. La siguiente imagen es tomada de la consola y nos muestra un proceso de instalación normal.

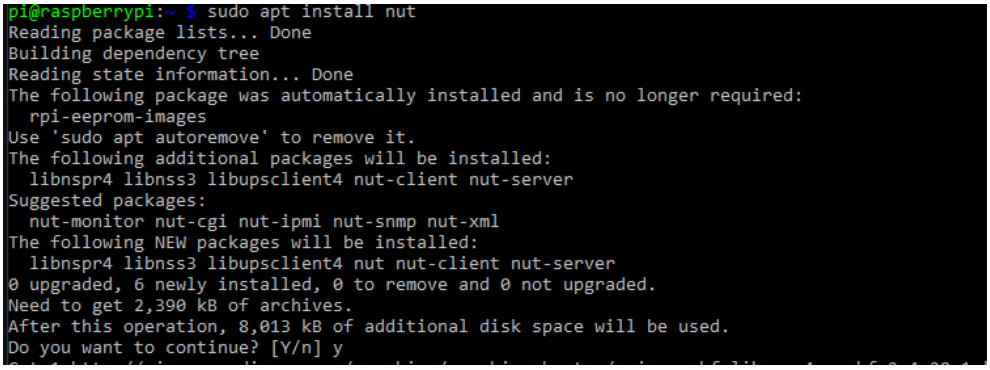

Figura 3-11.: Instalación de software NUT en Raspberry Pi

3.11. Identificación del puerto serie

El siguiente paso fue necesario en la ejecución del proyecto, consistió en la identificación del puerto serie por donde la ups se comunica con la Raspberry, para lograr esta identificación, tuvimos que realizar la conexión física entre el ups y la Raspberry, mediante el conversor RS-232 a usb. Hecho esto nos apoyamos de dos comandos, que nos ayudaron en la identificación, el primero nos mostró una lista de todos los dispositivos conectados a los puertos usb (lsusb), el segundo nos mostró el puerto serie asignado para el conversor(dmesg — grep tty) , en nuestra conexión el conversor se encuentro conectado y asignado en el puerto llamado ttyUSB0., al momento de identificarlo lo debemos llamar; /dev/ttyUSB0. La siguiente imagen nos muestra el comando y la identificación del puerto:

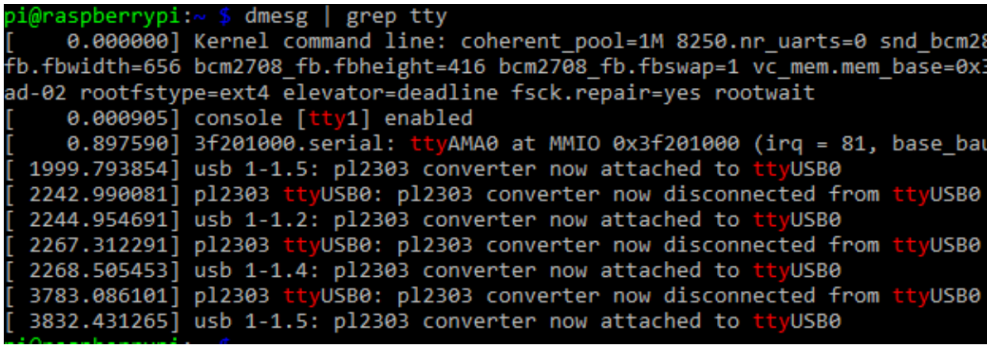

Figura 3-12.: Identificación del puerto

3.12. Configuración Básica

Para que el software pudiera reconocer el UPS, se hizo necesaria la modificación de unos scrip o archivos de configuración, en ellos se suministró el nombre del UPS, el tipo de Driver o controlador implementado para la comunicación, y el puerto de comunicación utilizado, la identificación de cada uno de estos ítems se realizó y quedo detallada en procesos anteriormente ya descritos. Respecto al prototipo utilizado se implementó una configuración básica la cual solo se encarga de establecer comunicación con el ups para lograr capturar sus datos, no se realizó ninguna configuración automática para apagado secuencial o apagado por batería baja, que es lo que normalmente ocurre en la forma como normalmente se utiliza el software NUT. La configuración básica va acorde al tipo de interconexión es decir una configuración básica es consecuente con una interconexión simple, pero no es conveniente realizar una configuración básica con una interconexión avanzada donde se requiere un apagado secuencial de varios servidores.

3.12.1. Identificación del controlador

Identificar el driver adecuado para monitorear el ups pudo haber sido una labor difícil, afortunadamente para el proyecto NUT ofrece en su página oficial una guía para determinar cuál es el controlador adecuado según la

marca y modelo del equipo a monitorear, en la siguiente imagen tomada de la página se puede identificar el driver especifico de acuerdo a la marca y modelo de ups que se utilizó en este proyecto (señalado con flecha roja).

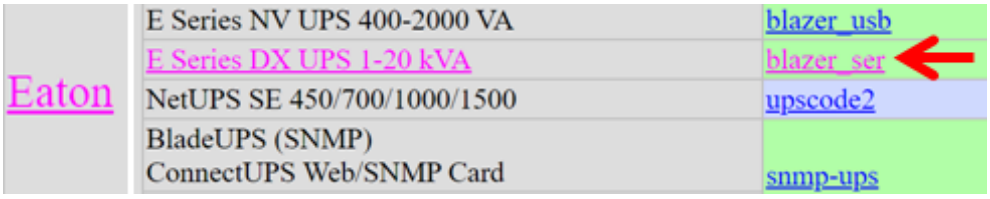

Figura 3-13.: Driver para instalación del NUT

3.12.2. Configuración del driver o controlador

Para este proceso fue necesario modificar el scrip o archivo de configuración identificado como: **ups.conf**. En sistemas operativos basados en Linux, se hace necesario la utilización de un editor de textos para lograr la edición o configuración de los scrips, en este prototipo se utilizo el editor de textos llamada "nano" , pero de igual forma pudo haberse utilizado uno distinto. Una vez se identificó el archivo de configuración y se tenía conocimiento del editor de texto ,se realizó la modificación del scrip con el siguiente comando:

sudo nano /etc/nut/ups.conf

Dentro del archivo de configuración, nos desplazamos con las teclas del cursor e identificamos las líneas a editar, hay que tener en cuenta que la edición requiere de comentar líneas o "des comentar" líneas lo cual no es mas que agregar o quitar el signo "". la siguiente imagen es un recorte del archivo donde se ve un dispositivo sin ningún tipo de configuración.

```
# The general form is:
#
# [upsname]
#       driver = <drivername>
#         port = <portname>
#       < any other directives here >
#
```

Figura 3-14.: Ambiente de configuración

Las líneas identificadas como “[upsname]” y “¡any other directivas here¿” son de libre edición , generalmente se coloca en ellas el nombre que le queremos dar al UPS o el modelo , lo que si no es de libre edición son las líneas de “driver” y “port” en ellas toco colocar obligatoriamente el nombre del controlador adecuado para el UPS y el puerto donde se encuentra conectado el conversor serie-usb, si estos dos últimos datos no son correctos será imposible la comunicación con el equipo, la siguiente imagen nos muestra la configuración que se utilizó para el driver de la UPS implementada en este proyecto.

```
# The general form is:
#
 [eatondx]
          driver = blazer_ser
          port = /dev/ttyUSB0
#       < any other directives here >
#
```

Figura 3-15.: Configuración de la UPS a utilizar

Sobre el mismo archivo nos ubicamos en la ultima línea , donde se encuentra el texto que dice; “maxretry = 3” y procedemos a comentarla, al comentar esta línea se deshabilita la función de maxretry, la cual es especificar el numero de intento en que se inicia el controlador en caso de haber una falla, en las pruebas que se realizaron salía un mensaje diciendo que no

era un nombre de variable valido para el driver, por este motivo se dejó deshabilitada para evitar conflictos.

3.13. Configuración del modo de operación

Esta configuración se realizó de acuerdo al tipo de interconexión que se implementó físicamente entre la raspberry y la ups, la cual según su topología corresponde a una interconexión simple, esta interconexión debe ser especificada dentro del archivo de configuración, donde la podemos encontrar definida como "standalone", o único ,definir este parámetro era importante , ya que determina que procesos ejecutara el programa al momento del arranque . la ruta de comando para dicha configuración es la siguiente:

sudo nano /etc/nut/nut.conf

3.14. Ejecución de permisos

Se implementa mediante dos comandos de consola los cuales cambian de propietario a un archivo y le asignan un grupo, en este caso un usuario root, de modo que la carpeta de NUT solo puede ser manipulada por el usuario root y arrancara automáticamente con el sistema.

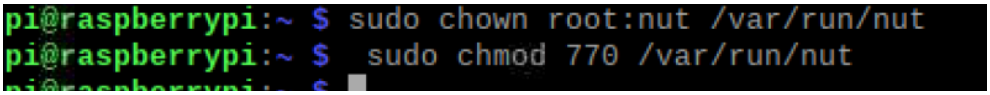

Figura 3-16.: Ejecución de permisos

3.15. Comprobación de driver y comunicación

Para garantizar el inicio del controlador o driver del ups, utilizamos un componente de nut llamado : **upsdrvctl**, este componente se encarga de iniciar el proceso de arranque del driver y comprobar la comunicación con

el ups, se ejecuta como comando de consola, a continuación se muestra una imagen de la ejecución de este componente.

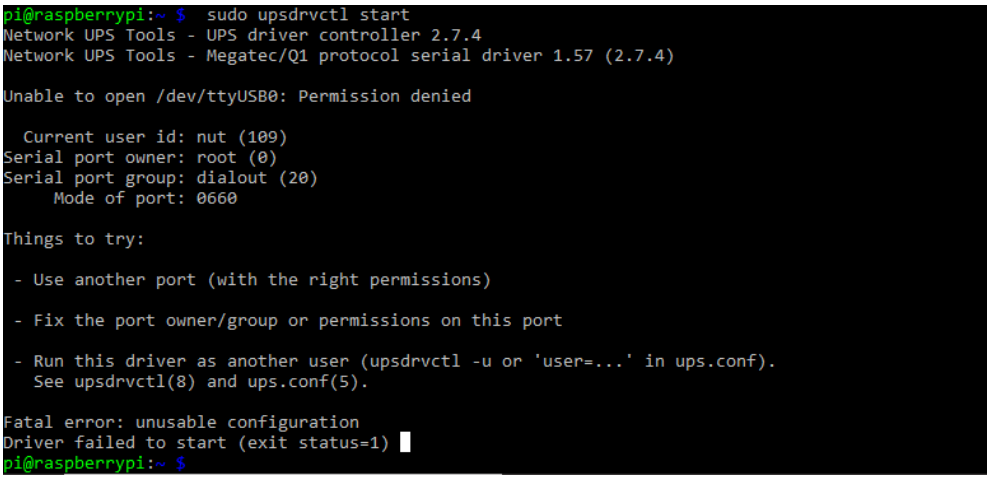

Figura 3-17.: Comprobación de comunicación en ventana de consola

Podemos ver en la figura anterior como se inicio el software NUT, y se identifico el controlador o driver, sin embargo se puede notar que hay un mensaje que indica que el permiso hacia el puerto **/dev/ttyUSB0** se encuentra denegado, si no hay permisos de acceso al puerto, no se puede establecer comunicación con el ups, por lo cual se puede leer en la última línea el mensaje de que el driver fallo al arrancar.

3.15.1. Permisos de acceso

Para solucionar el problema de permiso de acceso al puerto **/dev/ttyUSB0** el cual no permite la comunicación con el ups, se hizo necesario ejecutar el siguiente comando por consola:

sudo chmod 666 /dev/ttyUSB0

Luego de ejecutar el comando, se volvió a utilizar el componente **upsdrvctl** para darle inicio del proceso de arranque del controlador de nut, la siguiente

imagen nos muestra un proceso de conexión satisfactorio.

```
pi@raspberrypi:~ $ sudo upsdrvctl start
Network UPS Tools - UPS driver controller 2.7.4
Network UPS Tools - Megatec/Q1 protocol serial driver 1.57 (2.7.4)
Supported UPS detected with megatec protocol
Vendor information unavailable
No values provided for battery high/low voltages in ups.conf

Using 'guestimation' (low: 41.600000, high: 52.000000)!
Battery runtime will not be calculated (runtimecal not set)
```

Figura 3-18.: Comprobación de comunicación satisfactoria

3.16. Verificación de datos del UPS

Una vez finalizada la configuración de los scrip de nut, procedimos a verificar los datos suministrados por el UPS, mediante su visualización por la consola de la raspberry , para esto utilizamos el cliente de control **upsc** seguido del nombre que le colocamos a la ups en el archivo de configuración. La sentencia completa queda:

upsc eatondx

la siguiente imagen nos muestra la implementación del comando y la visualide los datos por consola :

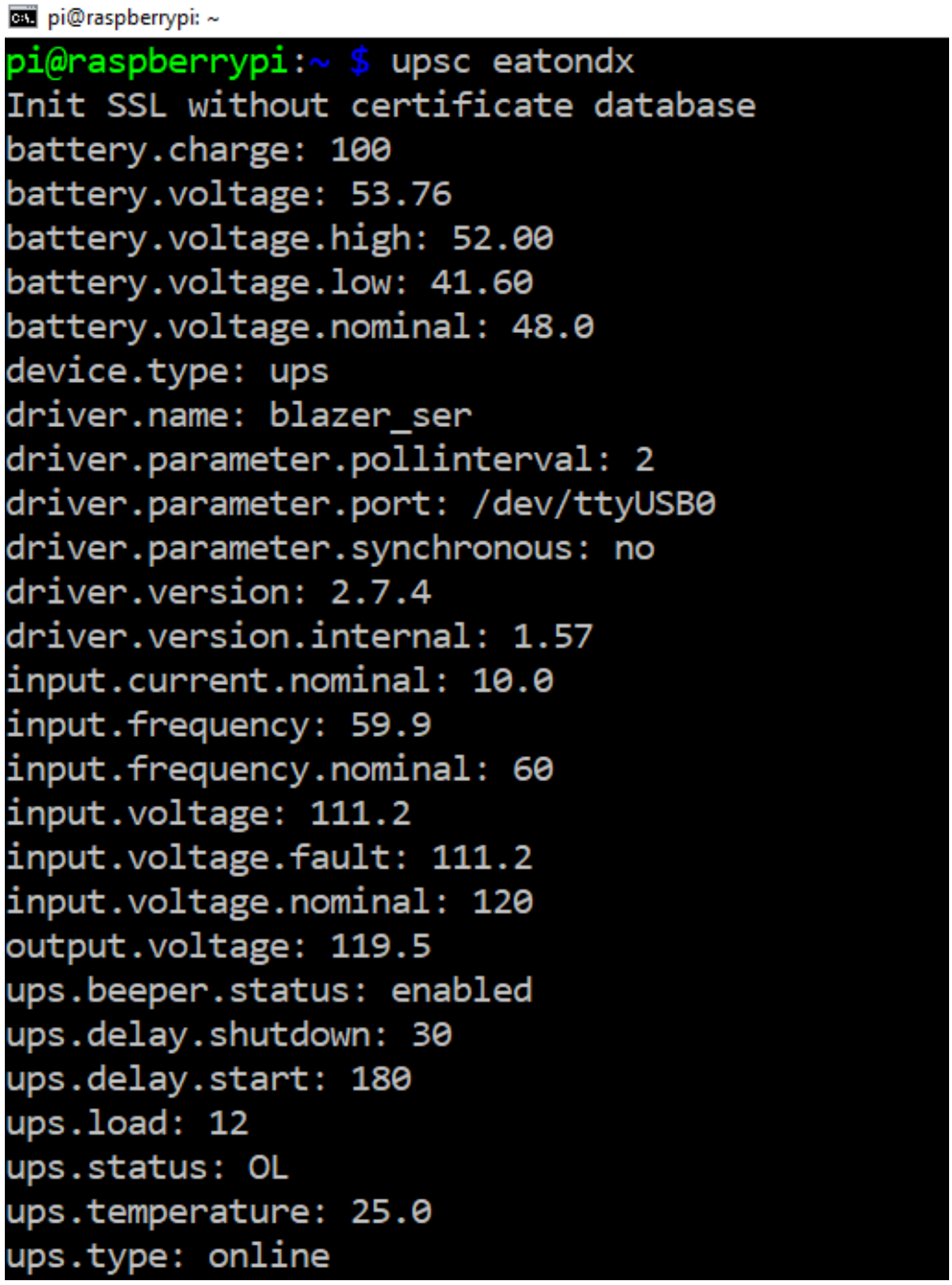

Figura 3-19.: Visualización de datos por consola

3.17. Automatización de arranque y servicio

En la ejecución de los pasos anteriores se logro establecer comunicación con el ups y se pudieron visualizar sus datos o parámetros eléctricos a través de la consola, se realizaron pruebas de reinicio o de apagado de la raspberry, y al estar nuevamente operativo se verifico si la comunicación con el ups se había restablecido, las pruebas de reinicio nos mostraron que el proceso de comunicación no se restablecía automáticamente, la anterior situación resulto siendo un inconveniente ya que implicaba la realización de un proceso manual para que el driver volviera a entablar comunicación con el ups. Para solucionar el problema anterior recurrimos al diseño de un scrip el cual otorgaba el permiso de acceso al puerto, e inicializaba el proceso de arranque del controlador de nut. Luego diseñamos un servicio o DAEMON quien se encarga de ejecutar el proceso en segundo plano sin interacción de ningún usuario, es decir de forma automática. Seguido de esto se ejecutaron unos comandos por consola para dar permisos de ejecución al Daemon y que logre arrancar al iniciar el sistema.

Se volvieron a realizar las pruebas de reinicio o apagado a la raspberry y se pudo comprobar que tanto el arranque del driver como la comunicación con el ups se realizaban de forma automática.

Se realiza el siguiente scrip para el arranque automático de los servicios:

/etc/init.d/service-nut.sh

```
#!/bin/bash
sudo chmod 666 /dev/ttyUSB0
sudo upsdrvctl start
```

Con el siguiente scrip se inician los servicios de NUT:

/etc/systemd/system/service-nut.service

Unit
Description=Arranque servicio NUT

Service
Type=simple
User=pi
ExecStart= sh /etc/init.d/service-nut.sh

Install
WantedBy=multi-user.target

3.18. Base de datos

En los procesos anteriormente explicados se logró la comunicación con el ups, y la visualización de sus parámetros mediante un comando de consola, también se logró acondicionar el sistema de monitoreo para que inicie automáticamente sin embargo hasta este punto del proceso estos datos solo se extraían en tiempo real y no se tiene ningún registro de ellos, por lo cual se hizo necesario la creación de una base de datos, para esto nos apoyamos de la herramienta INFLUXBD.

3.18.1. Instalación de INFLUXBD

Influx es una base de datos que posea alta capacidad de procesamiento y almacenamiento, puede realizar registros métricos de una infinidad de componentes, desde sensores básicos para aplicaciones de internet de las cosas, hasta datos de aplicaciones complejas con altos índices de registros, el programa trabaja almacenando los datos en series de tiempo. Para poder almacenar los datos obtenidos primero agregábamos la configuración de repositorios de INFLUXDB.

3.18.2. Inicialización y habilitación de servicios

Este proceso se realizó para que el servicio se inicie automáticamente, se logró mediante dos líneas de comandos, la siguiente imagen nos muestra el inicio del programa desde la consola de la raspberry pi.

Figura 3-20.: Inicialización por consola

3.18.3. Creación de la base de datos

Influx posee una Shell propia la cual es una interfaz de línea de comandos para registrar datos, permite consultar los datos de una forma dinámica y permite visualizar los registros en distintos formatos. Primero se ejecutó la Shell de nuestra base de datos con influxDB, luego se utilizo una línea de comandos para crear la base de datos, posteriormente se ingresó para verificar que la base de datos estuviera creada, en la siguiente imagen podemos ver nuestra base de datos creada.

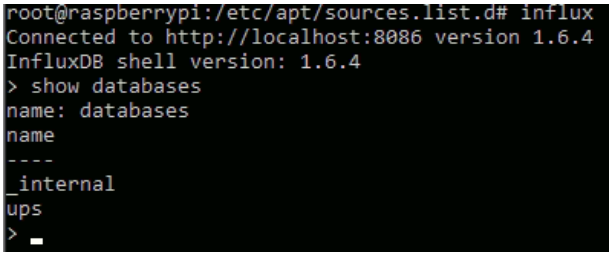

Figura 3-21.: Creación por consola

3.18.4. Creación del scrip de lectura de datos

En este proceso nos dirigimos al directorio **/opt/** donde se creó un scrip llamado **code.php** este scrip fue el encargado de leer los datos de la ups e insertarlos en la base de datos de ups en influx. Luego se procede a la instalación de las dependencias de php y automatizar el scrip para el scraping. La siguiente es una imagen de referencia del scrip:

```
#!/usr/bin/php
<?php

$command = "upsc";
$args = "ups";
$tagsArray = array(
"battery.charge",
"device.model",
"ups.realpower",
"ups.load",
"battery.runtime",
"output.voltage"
);

//do system call

$call = $command." ".$args;
$output = shell_exec($call);

//parse output for tag and value
```

Figura 3-22.: Creación de scrip por consola de lectura de datos

3.19. Prueba y verificación del sistema

Para validar si el sistema de registros de parámetros del UPS funcionaba adecuadamente, se procedió a realizar pruebas de operación sobre el UPS, estas pruebas consistieron en comandos de encendido y apagado, y una simulación del corte de energía. Estas pruebas forzaron al UPS a trabajar en todos sus modos de operación, se procedió a tomar un registro de las variables eléctricas en cada modo, utilizando un multímetro de la marca Fluke.

3.19.1. Mediciones eléctricas con la UPS en modo Standby

La siguiente imagen nos muestra el funcionamiento del UPS en el modo standby, y las mediciones registradas por el Multímetro.

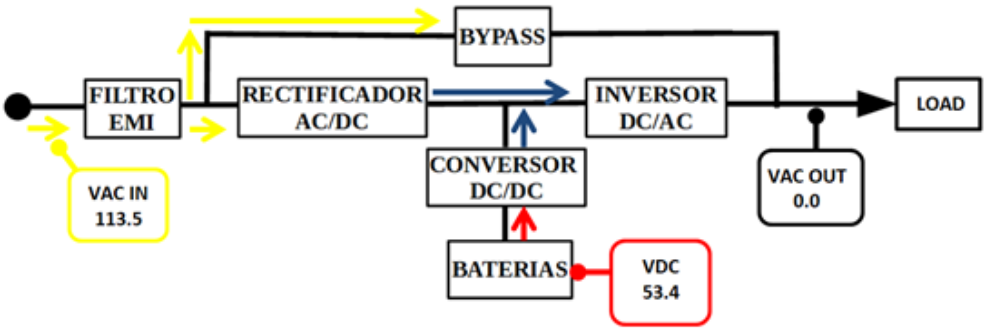

Figura 3-23.: Mediciones de voltaje AC y DC registradas por el multímetro con el UPS en el modo de operación standby

3.19.2. Funcionamiento del modo de operación Standby y registro de parámetros

En este estado de operación, la UPS recibe suministro de la red eléctrica comercial, las mediciones de voltaje de entrada ("VAC IN"), registraron un valor de 113.5 voltios AC, de igual manera se encuentra presente el voltaje de baterías ("VDC"), el cual nos mostró un valor de 53.4 Voltios DC. Al realizar la medición de voltaje en las toma corrientes de salida del UPS, el valor de medida fue cero voltios, esto se debe a que en el modo STANDBY el inversor se encuentra apagado y a la espera de ser encendido, se procede a registras los valores tal y como se observa en la siguiente tabla.

Estado del UPS	Mediciones con multímetro Fluke		
	VAC IN	VAC OUT	VDC
STANDBY	113.5	0.0	53.4

Tabla 3-1.: Registro de mediciones eléctricas del multímetro en el modo de operación STANDBY

3.19.3. Mediciones eléctricas con el UPS en modo BYPASS

La siguiente imagen nos muestra el funcionamiento del UPS en el modo BYPASS y las mediciones registradas por el multímetro.

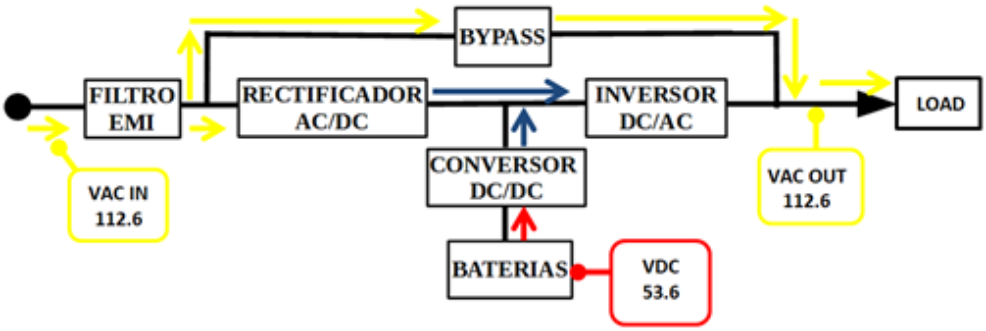

Figura 3-24.: Mediciones de voltaje AC y DC registradas por el multímetro con el UPS en el modo de operación BYPASS

En este estado de operación, la UPS recibe suministro de la red eléctrica comercial, las mediciones de voltaje de entrada ("VAC IN"), registraron un valor de 112.6 voltios AC, de igual manera se encuentra presente el voltaje de baterías ("VDC"), el cual nos mostró un valor de 53.6 Voltios DC. Al realizar la medición de tensión de salida("VAC out"), en las toma corrientes de suministro del UPS, el valor de medida fue de 112.6 voltios, esto se debe a que en el modo bypass el inversor de la UPS, no trabaja, pero el bypass interno del equipo se activa por lo cual se realiza una conexión directa entre la entrada y la salida del UPS, esto trae como consecuencia que la tensión de

salida sea similar a la tensión de entrada, se procede a registras los valores en la misma tabla donde se habían hecho los registros del modo Standby y como se observa en la siguiente tabla.

Estado del UPS	Mediciones con multímetro Fluke		
	VAC IN	VAC OUT	VDC
STANDBY	113.5	0.0	53.4
BYPASS	112.6	112.6	53.6

Tabla 3-2.: Se adiciono registro de mediciones eléctricas del multímetro en el modo de operación BYPASS

3.19.4. Mediciones eléctricas con el UPS en modo Normal

La siguiente imagen nos muestra el funcionamiento del UPS en el modo NORMAL y las mediciones registradas por el multímetro.

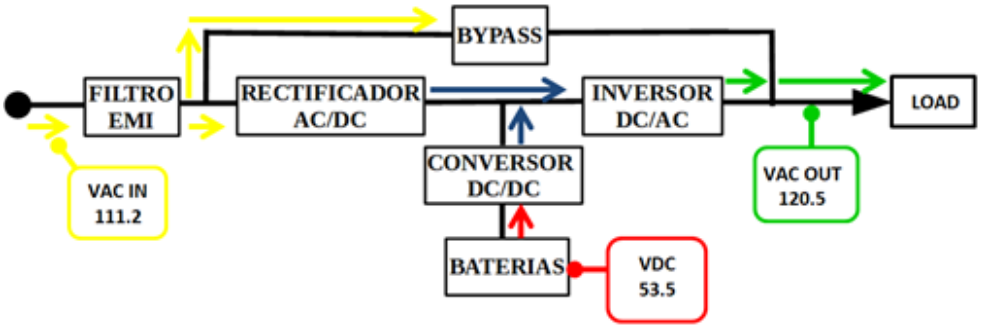

Figura 3-25.: Mediciones de voltaje AC y DC registradas por el multímetro con el UPS en el modo de operación NORMAL

En este estado de operación, la UPS recibe suministro de la red eléctrica comercial, las mediciones de voltaje de entrada ("VAC IN"), registraron un valor de 111.2 voltios AC, de igual manera se encuentra presente el voltaje de baterías ("VDC"), el cual nos mostró un valor de 53.5 Voltios DC. Al realizar

la medición de tensión de salida ("VAC UOT"), en las toma corrientes de suministro del UPS, el valor de medida fue de 120.5 voltios, esta variación de voltaje se debe a que en el modo NORMAL el inversor de la UPS se encuentra activo, en este modo de operación la entrada de voltaje del UPS puede sufrir variaciones en el orden de varios voltios, pero el voltaje de salida se mantiene en un nivel de tensión constante, es decir, un suministro regulado. Se procede a registras los valores en la misma tabla donde se habían hecho los registros del modo STANDBY y del modo BYPASS como se observa en la siguiente tabla.

Estado del UPS	Mediciones con multímetro Fluke		
	VAC IN	VAC OUT	VDC
STANDBY	113.5	0.0	53.4
BYPASS	112.6	112.6	53.6
NORMAL	111.2	120.5	53.5

Tabla 3-3.: Se adicionó registro de mediciones eléctricas del multímetro en el modo de operación NORMAL

3.19.5. Mediciones eléctricas con el UPS en modo Baterías

La siguiente imagen nos muestra el funcionamiento del UPS en el modo BATERÍAS y las mediciones registradas por el multímetro.

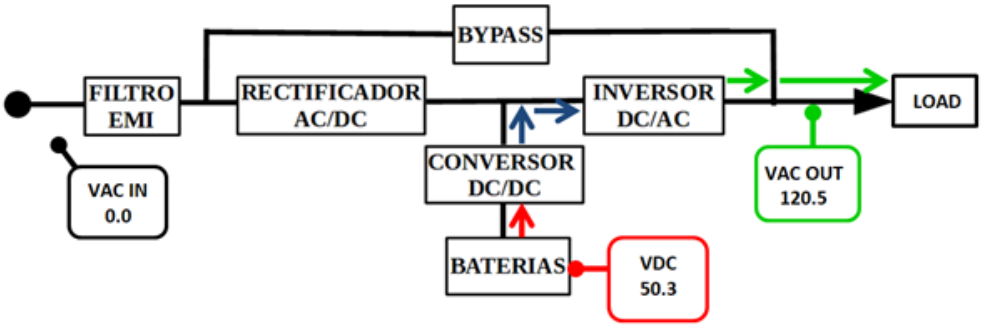

Figura 3-26.: Mediciones de voltaje AC y DC registradas por el multímetro con el UPS en el modo de operación BATERÍAS

En este modo de operación, se interrumpe el suministro eléctrico que recibe la entrada del UPS, la medición de voltaje de entrada ("VAC IN"), es de cero voltios AC, el UPS al detector el fallo de red eléctrica, suministra la energía almacenada en las baterías, por lo cual, hay una tensión DC presente, pero que irá disminuyendo en función del tiempo y de la carga, el voltaje de baterías ("VDC"), nos mostró un valor de 50.3 Voltios DC. Al realizar la medición de tensión de salida ("VAC out"), en las toma corrientes de suministro del UPS, el valor de medida fue de 120.5 voltios ya que el inversor de la UPS se encuentra activo, y es alimentado por las baterías del UPS. Se procede a registras los valores en la misma tabla donde se habían hecho los registros del modo STANDBY, del modo BYPASS, del modo NORMAL y ahora el modo baterías, como se observa en la siguiente tabla.

Estado del UPS	Mediciones con multímetro Fluke		
	VAC IN	VAC OUT	VDC
STANDBY	113.5	0.0	53.4
BYPASS	112.6	112.6	53.6
NORMAL	111.2	120.5	53.5
BATERÍAS	0.0	120.5	50.3

Tabla 3-4.: Se adicionó registro de mediciones eléctricas del multímetro en el modo de operación Baterías

3.20. Análisis de los resultados obtenidos

Las pruebas realizadas nos demostraron que dependiendo del modo de operación del UPS, las variables eléctricas experimentaron cambios asociados con el estado, por ejemplo en el modo STANDBY tuvimos una tensión de salida nula, en el modo BYPASS la tensión de salida fue similar a la tensión de entrada, en el modo normal la tensión de salida se mantuvo en un valor constante sin importar las variaciones de la tensión de entrada y por último en el modo baterías la tensión de entrada fue cero pero la salida del UPS siguió manteniendo su mismo valor constante, todas estas evidencias sirvieron de referencia para replicar los modos de operación y ver el registro realiza por el software NUT, de esta forma se pudo saber si el software respondió a los cambios registrados por el UPS en cada transición de estados, a continuación se muestra la toma de registros hechas por el software NUT.

Estado del UPS	NUT		
	VAC IN	VAC OUT	VDC
STANDBY	111.2	0.0	53.76
BYPASS	112.3	112.3	53.72
NORMAL	112.5	119.5	53.67
BATERÍAS	0.0	119.5	50.1

Tabla 3-5.: Medidas visualizadas por software NUT

3.20.1. Comparación de Tablas de resultados

Estado UPS	Mediciones Fluke			Mediciones NUT		
	VACIN	VACOUT	VDC	VACIN	VACOUT	VDC
STANDBY	113.5	0.0	53.4	111.2	0.0	53.76
BYPASS	112.6	112.6	53.6	112.3	112.3	53.72
NORMAL	111.2	120.5	53.5	112.5	119.5	53.67
BATERÍAS	0.0	120.5	50.3	0.0	119.5	50.1

Tabla 3-6.: Comparación de los registros del multímetro y software NUT

Al realizar una comparación de tablas, referenciadas sobre el mismo estado de operación del UPS, se pudo notar que en el modo STANDBY, ambas mediciones de salida (VAC OUT) fueron de cero voltios, en el modo bypass, los registros hechos por el multímetro fueron similares, tanto para la tensión de entrada de voltaje (VAC IN) como para la salida VAC OUT, el mismo comportamiento ocurrió con el software NUT.En el modo normal como en el modo baterías, tanto la medición del multímetro como la medición de NUT mostraron la tensión de salida (VAC OUT) constante lo cual es lógico ya que en ambos estados el UPS debe mantener una salida regulada e ininterrumpida. Y por último podemos observar que en el modo baterías tanto el multímetro como NUT mostraron una medición de cero voltios, lo cual es consecuente en ambos casos ya que la tensión de entrada fue interrumpida manualmente.

Dadas las pruebas anteriores se pudo concluir que las mediciones del software NUT registraron el mismo comportamiento que las mediciones realizadas por el multímetro en los distintos modos de operación, por lo cual en referencia a las medidas de voltaje de entrada, voltaje de salida y voltaje de baterías, la medición realizada por el sistema de registro de parámetros fue confiable. Los otros parámetros suministrados por NUT fueron analizados en los siguientes capítulos.

4. Implementación de Instrumentación

Para el proceso de medición de las magnitudes eléctricas de la UPS como sistema redundante, se realiza la selección de un embebido que sea de bajo costo y cumpla con las tareas de realizar la medición de voltaje alterno, voltaje directo, temperatura, humedad relativa e intensidad de consumo.

Para la medición de las anteriores variables, se realiza la viabilidad de utilizar un arduino nano, dado que utilizando el arduino UNO, el cual era la primera opción para la adquisición de datos, pero al hacer la comparativa, el costo esta por encima del arduino nano, lo cual es hacia donde esta orientado el proyecto. Por tanto en el arduino nano, encontramos la posibilidad de leer todos los sensores y en segunda instancia pero no menos importante, la de transmitir a la raspberry pi, quien es la encargada de almacenar esta información en la base de datos.

Realizando la transmisión de datos hacia el embebido principal por medio de puerto serial. Cuando los datos enviados por el arduino llegan a la raspberry, serán almacenados para posibles verificaciones y comparaciones con las mediciones entregadas por la UPS, siendo este una manera externa de comprobar las mediciones realizadas por la UPS.

En las pruebas se realizaba la visualización de estas variables en la pantalla del PC, pero el objetivo final fue realizar la entrega de estas variables a la raspberry pi.

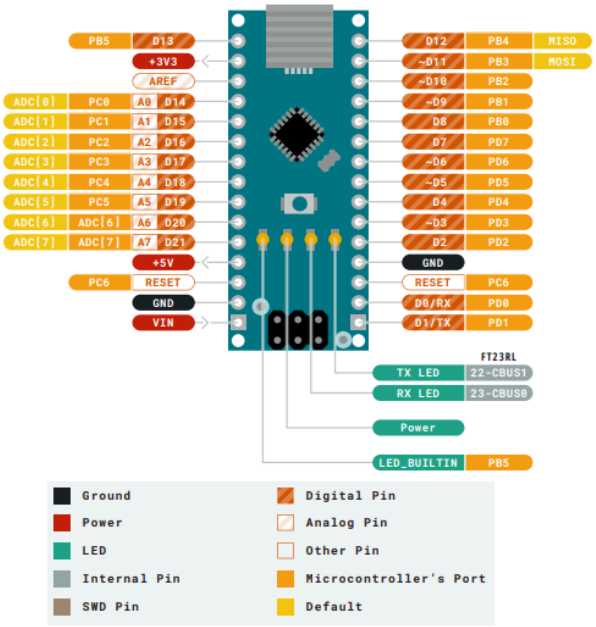

Figura 4-1.: Descripción de entradas y salidas de Arduino Nano

Una de las razones por las cuales también fue tenido en cuenta este dispositivo, es que cuenta con un shield de borneras que permite ser utilizado en ambientes robustos como lo son los entornos industriales.

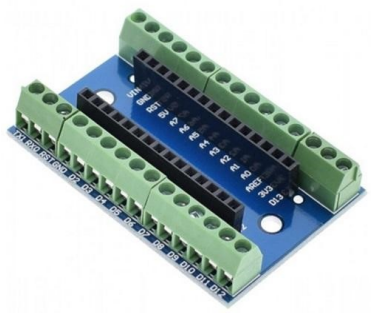

Figura 4-2.: Shield de borneras para arduino nano

Una vez seleccionado el arduino nano, se realiza el montaje de los sensores para la previa extracción de variables.

4.1. Sensores de adquisición de señales

A continuación se realiza la descripción de los sensores y se expone el diagrama de conexiones de los sensores utilizados este proyecto.

4.1.1. Sensor de medición de intensidad eléctrica

Iniciamos con el sensor que realizara la medición de la intensidad de consumida por la carga, que sera entregada por la UPS. Como en el inicio se puso en conocimiento la existencia de dos tipos de sensores para la medición de esta variable eléctrica, una que es invasiva, que requiere abrir el circuito para la medición de la variable, y la otra manera que es no invasiva, que solo requiere la separación de los cables para poder realizar la instalación de este y proceder a realizar la medición.

Como se había mencionado anteriormente, este sensor SCT-013 [34] trabaja con el principio físico de la inducción electromagnética, por otra parte en cuanto a lo que se refiere a software, este sensor es compatible para la utilización en embebidos, requiere una librería para la adquisición correcta de datos.

Figura 4-3.: Sensor de intensidad no invasivo

4.1.2. Sensor de medición de voltaje alterno

En esta sección se encuentra el sensor de medición de voltaje alterno, que posibilita la medición a la entrada de la UPS y comprobar que a la salida, esta nos entregue un voltaje regulado, para este fin utilizaremos dos sensores. También nos otorga la información de saber en que momento la UPS esta trabajando con la batería o cuando esta con la red eléctrica.

El sensor se consigue en el mercado como ZMPT101B [35], permite mediciones desde 50 hasta 220 Vac, el cual entrega un voltaje de medida de 0 a 5 Vdc.

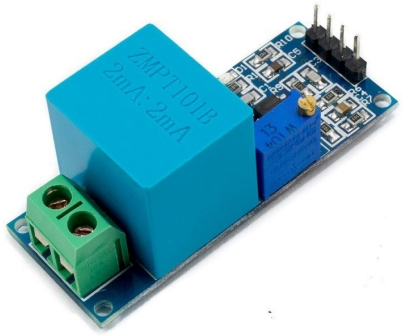

Figura 4-4.: Sensor de voltaje alterno

4.1.3. Sensor de medición de voltaje directo

Para esta medición, realizamos una escalización mediante un divisor de voltaje, para poder realizar la lectura con un rango de voltaje de 0 - 5Vdc. Cuando se realiza la verificación del sensor en el mercado, se comprueba que lo que este contiene es un divisor de voltaje. La medición de las baterías de la UPS oscilan entre 40 y 60 Vdc. Se realiza el calculo para la escalización de un voltaje de 0 hasta 70 Vdc. Mediante juego de resistencias con alta impedancia, se realiza el circuito con los siguientes valores:

$$Vout = Vin * \frac{R2}{R2 + R1}$$

$$Vout = 70 * \frac{7{,}700}{7700 + 100000}$$

$$Vout = 5Vdc$$

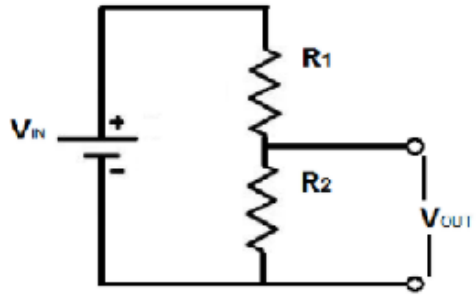

Figura 4-5.: Cálculos para medir voltaje directo

4.1.4. Sensor de medición de humedad relativa

Este es un sensor muy popular, se decidió realizar el trabajo con este sensor por que es un dispositivo de muy bajo costo que se utiliza para realizar este tipo de mediciones, integrando un sensor capacitivo que mide la humedad relativa y un termistor empleado para la medición de la temperatura. Los rangos de medición de este sensor, en temperatura son de 0 – 50 °C con una precisión de ±2.0 °C, en cuanto al rango de medición de la humedad relativa, comprende entre 20 % y 90 % humedad relativa. El DHT11 [36] es un sensor digital que utiliza 3 pines de conexiones, los cuales están distribuidos de la siguiente manera: 2 pines de alimentación (positivo y negativo) y un pin de datos por donde circulara toda la información adquirida del medio circundante. Realiza el envío de datos obtenidos del entorno mediante el protocolo de comunicación 1-wire (un hilo), hacia el embebido que estamos utilizando, en nuestro caso la Raspberry Pi. En cuanto a la interfaz, por preferencia y para evitar perturbación en campo, se recomienda que este sea cable apantallado y máximo hasta 20m de distancia, de igual manera, se recomienda que entre los pines de Vcc y Datos, se utilice una resistencia a modo de Pull-up.

Utilizaremos este sensor en primera medida para la medición de la humedad relativa, pero también aprovecharemos la medición de temperatura que este nos brinda, utilizándolo como unidad de medida redundante con respecto a la temperatura principal.

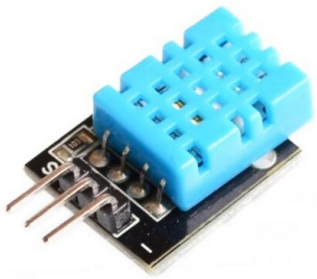

Figura 4-6.: Sensor de temperatura y humedad relativa DHT11

4.1.5. Sensor de medición de temperatura

Dentro de los sensores de temperatura para la realización de este proyecto, se tuvieron en cuenta el sensor RTD, que es un sensor de temperatura resistivo; estos sensores suelen tener una larga vida útil y ser confiables pero manejan una señal análoga. Otra alternativa fue la termopar, que tiene una variación de milivoltios de acuerdo a la variación de la temperatura, pero al igual que el sensor anterior, manejan una señal análoga.

Cabe resaltar que esta es una variable crítica, por las cuales el fabricante sugiere condiciones ambientales selectas, mas detalladamente con respecto a la temperatura de trabajo, que es la variable que prolonga la vida útil de las baterías, quienes son las protagonistas de estos sistemas. Para la medición de esta variable emplearemos un sensor que nos proporcione una señal digital, utilizaremos el DS18B20 [37].

Empleamos este sensor por que tiene como ventaja la precisión y que no es perturbado por el ruido, lo cual nos proporciona una señal confiable.

Figura 4-7.: Sensor digital de temperatura DS18B20 tipo sonda

4.2. Diseño y construcción de un banco de pruebas para el sistema de monitoreo

Si bien no estaba dentro de los objetivos del proyecto, se decidió realizar el diseño y construcción de un banco de pruebas para verificar la fiabilidad del proyecto.

4.2.1. Diseño del diagrama unifilar

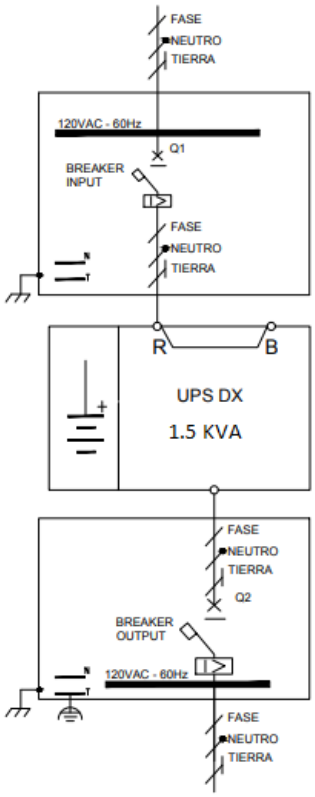

Figura 4-8.: Diagrama unifilar

El anterior es el diagrama unifilar donde se detalla las conexiones y protecciones eléctricas de la UPS.

En las protecciones eléctricas se utilizaron breakers de 16 amperios con conductores en calibre 12, adicional se colocó un banco resistivo a base de bombillos de 100 vatios, y dos toma corrientes, una con suministro regulado, y la otra con suministro normal, el sistema permite realizar pruebas de operación para probar el prototipo, en la siguiente imagen se puede ver el

diseño final.

Figura 4-9.: Banco de prueba para Sistema de monitoreo

4.3. Montaje de Instrumentación

Se realiza el montaje del siguiente diagrama, integrando de uno a uno los sensores mencionados anteriormente.

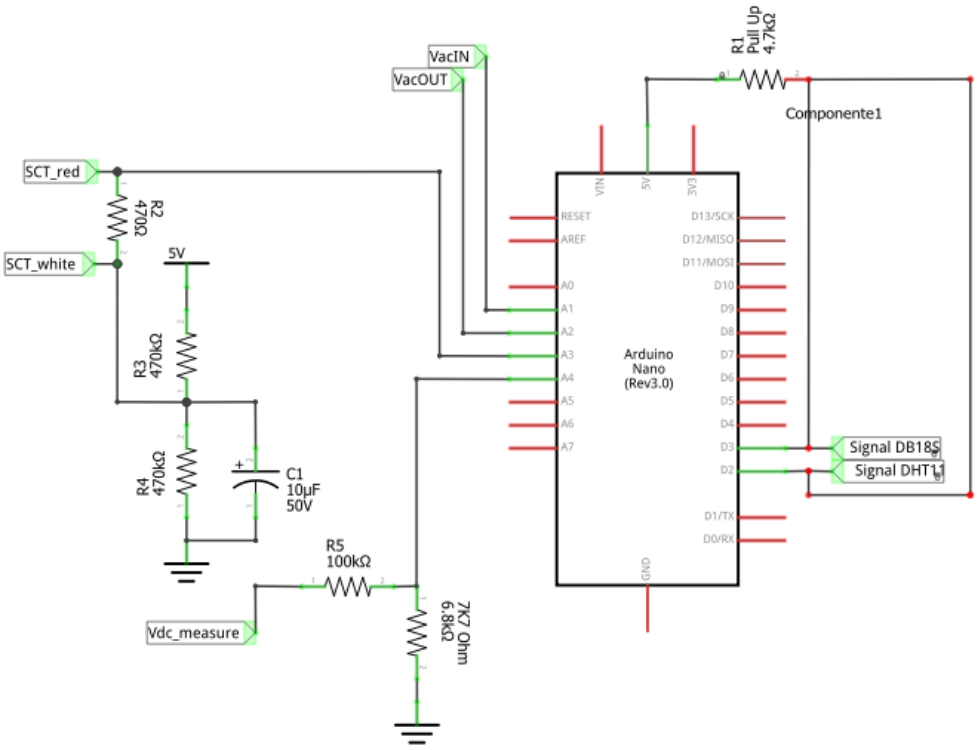

Figura 4-10.: Diagrama de instrumentación

Donde el arduino nano recibe las siguientes señales en sus determinados pines:

A1: Recibe la señal con etiqueta de **VacIN**, que indica la medición de voltaje alterno de entrada, dicho esto de otra manera, el voltaje de la red.
A2: Recibe la señal con etiqueta de **VacOUT**, la cual indica la medición del voltaje de salida de la UPS.
A3: Este pin recibe la señal de la intensidad eléctrica definida en el código como **Irms**, e identifica el consumo de la carga del sistema.
A4: Este pin recibe la señal escalizada de la medición de las baterías en voltaje directo.
D2: Este pin, nos recibe una señal digital con dos variables, una temperatura y la humedad relativa. En la impresión de las señales, esta temperatura esta etiquetada como **Temp 2** y **Humedad** en el código que contiene el arduino nano.
D3: Recibe la señal de temperatura proveniente del sensor instalado en el interior de la UPS el cual esta etiquetado con **Temp 1**.

El código contenido en el arduino nano, esta descrito en el anexo 1.

El montaje se realizo como prototipo de prueba en baquela perforada. Los resultados fueron correctos y se realizan pruebas de los diferentes estados que pueda tomar el equipo y su funcionamiento.

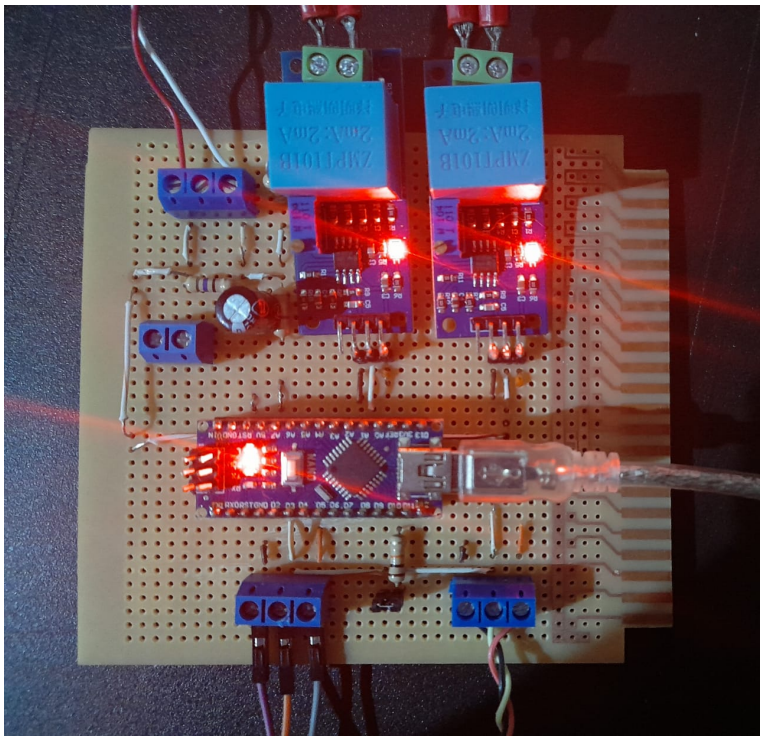

Figura 4-11.: Montaje físico del sistema de instrumentación

Realizado el montaje, se aprecia por medio de una ventana de monitoreo de puerto serial del computador, que los datos están siendo enviados desde el arduino nano hacia el monitor del computador y se aprecian las mediciones realizadas por la instrumentación implementada.

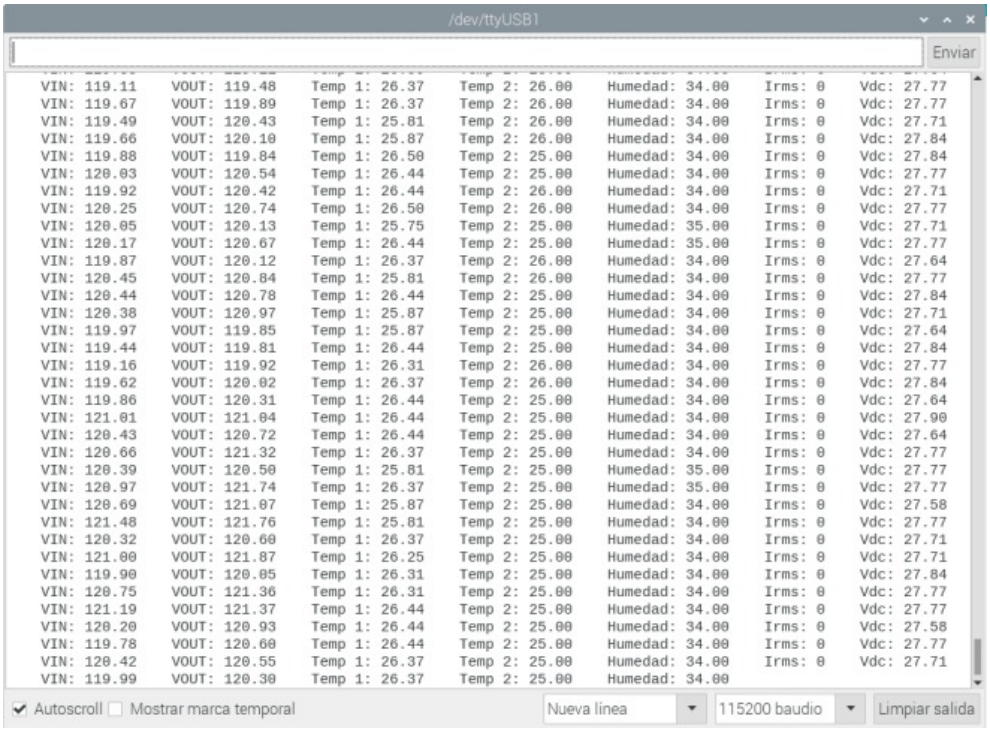

Figura 4-12.: Mediciones de la instrumentación visualizadas en monitor serial

4.4. Verificación Sistema de Instrumentación

Una vez finalizado el sistema de instrumentación se procedió a comprobar la fiabilidad de las mediciones arrojadas, se pudo observar que los datos suministrados eran erróneos ya que se conectaron ambos sensores a la salida regulada del UPS, la cual posee una tensión constante de muy baja variación, la imagen nos muestra el valor erróneo registrado por grafana:

Figura 4-13.: Voltaje en Grafana de UPS

Para descartar que la medida fuera un error del software de visualización grafana se procedió a verificar los datos por el monitor serie. La siguiente imagen nos muestra una variación de voltaje casi similar a la de grafana.

```
VIN: 131.29    VOUT: 131.19
VIN: 131.59    VOUT: 131.08
```

Figura 4-14.: Voltaje leído a través del puerto serial

Se procede a ajustar el sistema y se realizan 2 pruebas paralelas, la primera colocando el multimetro en la escala de mínimos y máximos por tiempo de una hora, con el fin de obtener la tensión mínima registrada y la tensión máxima registrada, el resultado se puede visualizar en la siguiente imagen:

Figura 4-15.: Voltaje medido a través de instrumento Fluke

De igual forma en la misma hora se tomaron 4 registros uno cada 15 minutos, los valores se observan en la siguiente tabla:

Tiempo	Valor medido por sensor de entrada de voltaje	Valor medido por sensor de salida de voltaje
15 minutos	122.12 VAC	122.17 VAC
30 minutos	121.90 VAC	122.01 VAC
45 minutos	122.49 VAC	122.35 VAC
60 minutos	121.34 VAC	122.81 VAC

Tabla 4-1.: mediciones tomadas cada 15 minutos, 4 muestreos totales

Dado el resultado anterior y descartando los decimales se pudo notar que la tensión de salida se mantuvo casi exacta en 122 vac en ambos sensores, el software grafana registro un valor similar.

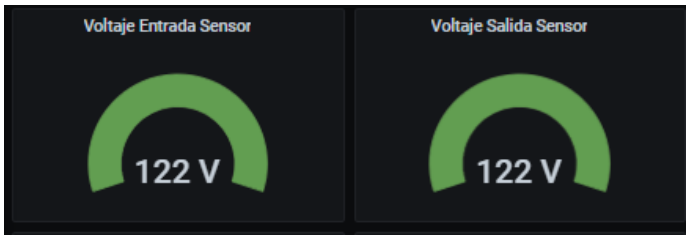

Figura 4-16.: Medidas registradas por grafana después del ajuste

4.4.1. Ajuste de medición de intensidad eléctrica

Para probar la medida del sensor de corriente se procedió a utilizar un banco de prueba en el que se utilizo una carga resistiva de conformado por 4 bombillos se midió la corriente con el multimetro obteniendo el siguiente valor:

Figura 4-17.: Corriente medida por el multimetro en la escala de 10 Amperios

Se procede a tomar 4 registros de corrientes en un intervalo de tiempo de 5 minutos, los valores obtenidos se registraron en la siguiente tabla:

Tiempo	Valor medido por sensor de corriente (Amp)
5 minutos	3.15
10 minutos	3.17
15 minutos	3.14
20 minutos	3.16

Tabla 4-2.: Valores de corrientes registrados por sensor de corriente

4.4.2. Análisis de resultados

Dada las observaciones de cada tabla y las medidas arrojadas por el multimetro se considera que los valores registrados por los sensores después de la calibración son bastantes similares a los registrados por el multimetro con variaciones solo en la parte decimal, por lo cual se concluye que el sistema arroja medidas confiables.

5. Sistema de monitoreo y notificación

En esta etapa de desarrollo del prototipo ya se había seleccionado la Raspberry pi ,se había implementado en ella el sistema de registro de parámetros del UPS y se había diseñado el dispositivo electrónico para el registro de las variables eléctricas y del entorno.

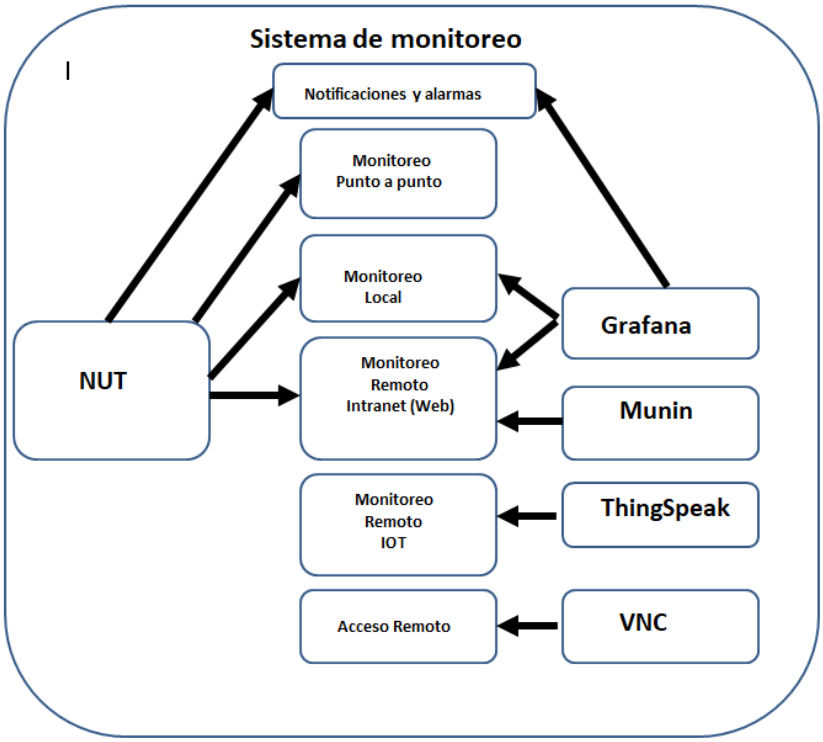

Figura 5-1.: Esquema general de monitoreo

Se procedió a la integración de todo el sistema y se inició el proceso de implementación y desarrollo encaminado a la elaboración de un sistema de notificaciones y monitoreo remoto y local que permitiera la reducción de los costos de monitoreo generados por los dispositivos suministrados por el fabricante.

A continuación se describen los procesos realizados con los que se logró el diseño e implementación del este sistema de monitoreo.

5.1. Instalación de herramientas de monitoreo

Esta etapa consistió en la instalación y configuración de los software , programas o agentes que se instalaron en la raspberry pi después de haber hecho pruebas con el software NUT , la instalación de estos software se hizo necesaria debido a las limitaciones que de por si presento el software NUT de no poseer una base de registro de los parámetros del UPS, y de no permitir la integración del dispositivo electrónico para el registro de las variables eléctricas y del entorno que se había diseñado.

A continuación presentamos las herramientas instaladas e implementadas con sus resultados.

5.1.1. Herramientas de monitoreo integradas de NUT

Estas herramientas han sido desarrolladas por el mismo fabricante de NUT. Si bien fueron dos las herramientas instaladas después de la instalación del NUT, su instalación añadió características adicionales al software. La implementación de estas herramientas se realizó con el fin de buscar nuevas características que permitieran integrar nuevas funcionalidades al sistema de monitoreo, los resultados del proceso de ejecución y los procedimientos realizados se detallan a continuación.

5.2. Monitoreo con Nut monitor

Su instalación se realizó por comandos desde la consola, este programa solo se pudo visualizar utilizando el entorno o escritorio gráfico del sistema operativo de la Raspberry, en su forma más básica, NUT monitor nos permitió, la visualización de una ventana con información básica del UPS, la siguiente imagen nos muestra los datos visualizados.

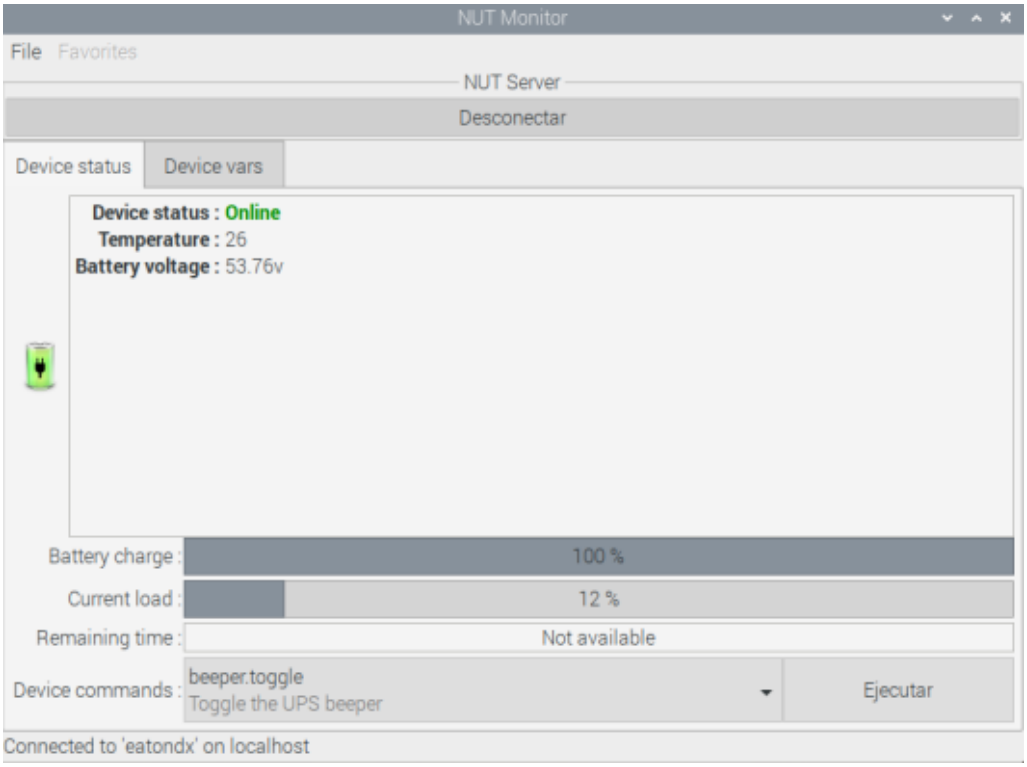

Figura 5-2.: Software NUT-monitor en ejecución

Se pudo visualizar en la ejecución del programa, que posee dos pestañas adicionales, la llamada "Device vars", visualizo todas las variables eléctricas del dispositivo monitoreado, la otra pestaña identificada como "Device Comand" desplegó un menú de comando de ejecución para controlar el UPS, lo cual es una ventaja porque permite encender o pagar el UPS desde un

acceso remoto a la raspberry pero también podría resultar siendo una desventaja en el sentido de que cualquier usuario con acceso a la Raspberry podría apagar el UPS, siendo esto un aspecto negativo si se trata de un sistema de misión crítica, a continuación se muestra las imágenes de las figuras desplegadas

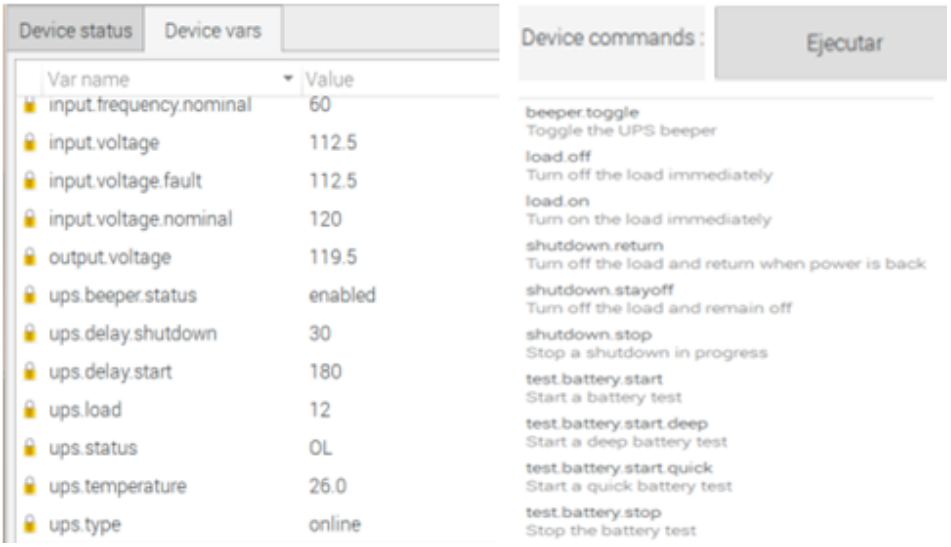

Figura 5-3.: a la izquierda visualización de las variables del dispositivo, a la derecha lista de comandos del dispositivo

Para verificar la funcionalidad de los comandos se procedió a ejecutar las ordenes pero ninguna funcionaba, se nos mostró en el proceso un mensaje de acceso restringido, lo cual significa que el software posee un usuario y contraseñas configurables para dicha ejecución, para verificar su funcionalidad se tuvo que realizar una configuración cliente-servidor, se realizó un proceso de edición en los scrip llamados "**/etc/nut/upsd.conf:** " y "**/etc/nut/upsd.users**"

Sobre el primero se habilitó la dirección IP y el puerto TCP que ya vienen establecidas por defecto, sobre esta dirección y puerto, el servidor escuchara a los clientes.

Sobre el segundo scrip se asigno el usuario y contraseña mediante los cuales nos podíamos conectar a ese servidor ejecutándolo sobre la misma computadora, se ejecutaron nuevas pruebas después de la edición de los scrip, y se realizó la conexión usando usuario contraseña y puerto local , se enviaron comandos al UPS pero solo respondieron 2, el primero era un apagado con retorno y el segundo era un encendido rápido, adicionalmente se comprobó que por la consola de la Raspberry se podía visualizar una lista de comandos disponibles por el driver de monitoreo y una sentencia para ejecutarlos, se adjunta evidencia a continuación:

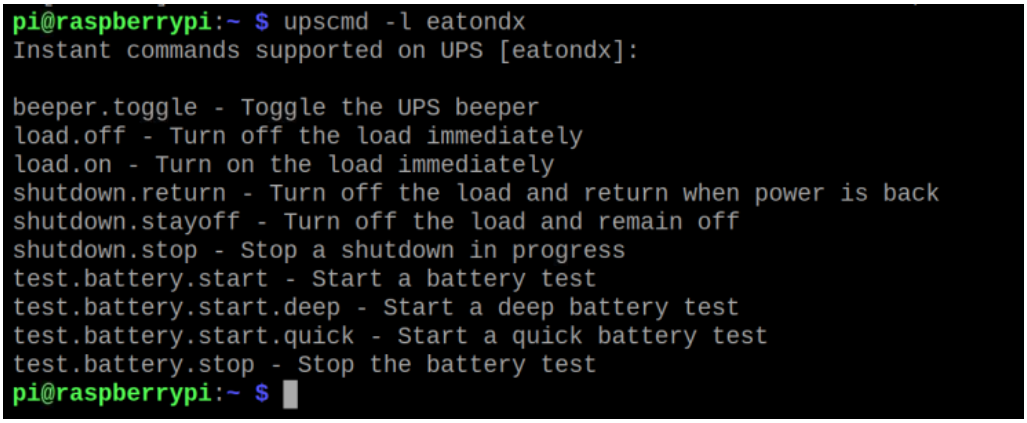

Figura 5-4.: Comandos soportados por UPS EATON

Posteriormente se realizaron las pruebas por consola y los resultados fueron los mismos, solo se pudieron ejecutar sobre la UPS dos comandos, el esto indico “OK” en la prueba pero no realizaban ninguna función, a continuación adjuntamos la imagen de los comandos ejecutados por consola:

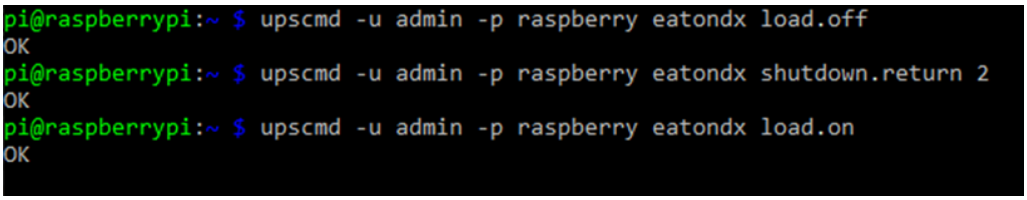

Figura 5-5.: Comandos ejecutados por consola para encendido y apagado del UPS

Hasta este punto habíamos finalizado todas las pruebas de NUT-monitor, si bien el software se ejecuta como un cliente de monitoreo con una interfaz gráfica amigable y unas pestañas para visualización de parámetros y comandos del dispositivo, el software seguía sin permitir almacenar datos, sin graficar y sin permitir la integración de la etapa de medición de variables de nuestro proyecto, sumando a eso la desventaja de que para su uso se requiere conexión directa a la raspberry con el uso de periféricos(teclado, mouse y pantalla), o en su defecto acceso a la Raspberry desde un cliente de acceso remoto al sistema operativo como VNC por ejemplo.

5.3. Monitoreo con NUT-CGI

NUT-CGI es un cliente de monitoreo web, que a diferencia de nut–monitor nos permitió monitorear la UPS desde la web, sin necesidad acceder al sistema operativo de la Raspberry ya sea directamente o mediante el uso de programas con acceso remoto, para poder utilizar este cliente de monitoreo nos fue necesario realizar dos procesos, el primero fue la instalación de un servidor apache 2 y la segunda instalar el cliente NUT-CGI, luego de unas configuraciones básicas procedimos a realizar las pruebas desde un PC con Windows conectados a la misma red LAN para poder acceder al monitoreo web se colocó la siguiente sentencia desde el navegador de internet:

$http : // < nuestra\_ip > /cgi - bin/nut/upsstats.cgi$

Lo cual despliega la siguiente ventana:

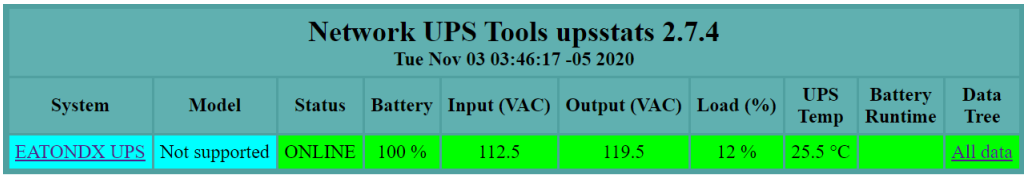

Figura 5-6.: Interface de acceso web de nut-cgi

La pestaña identificada como "**system**", visualizo una interfaz gráfica representada por barras visualmente llamativa, la podemos ver a continuación.

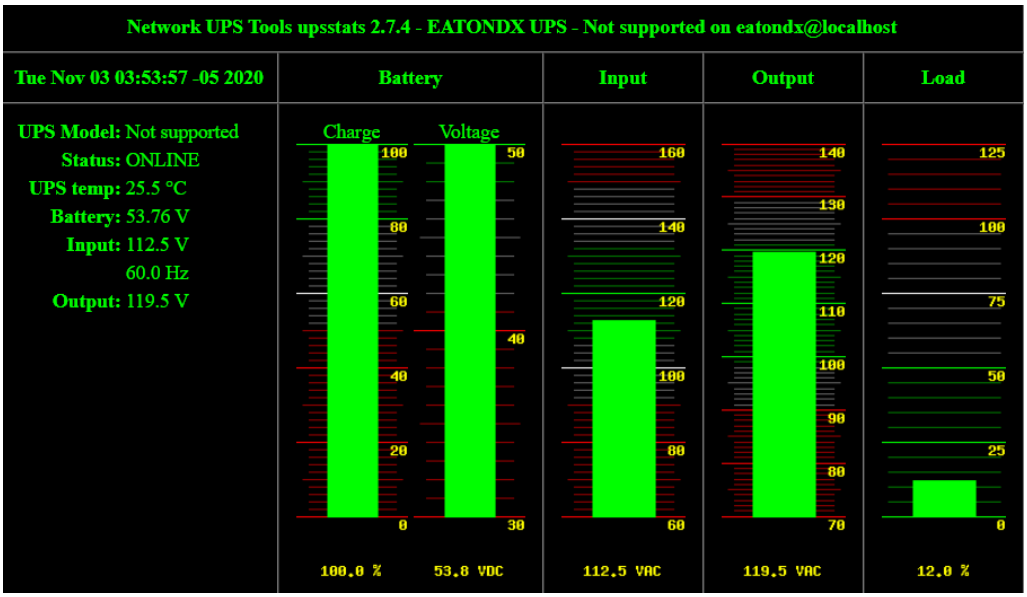

Figura 5-7.: Visualización de interfaz gráfica de nut-cgi a través de la web

De igual manera en la pestaña identificada como "**Data tree**", se logró visualizar los parámetros eléctricos registrados por la UPS, los datos los podemos ver en la siguiente imagen

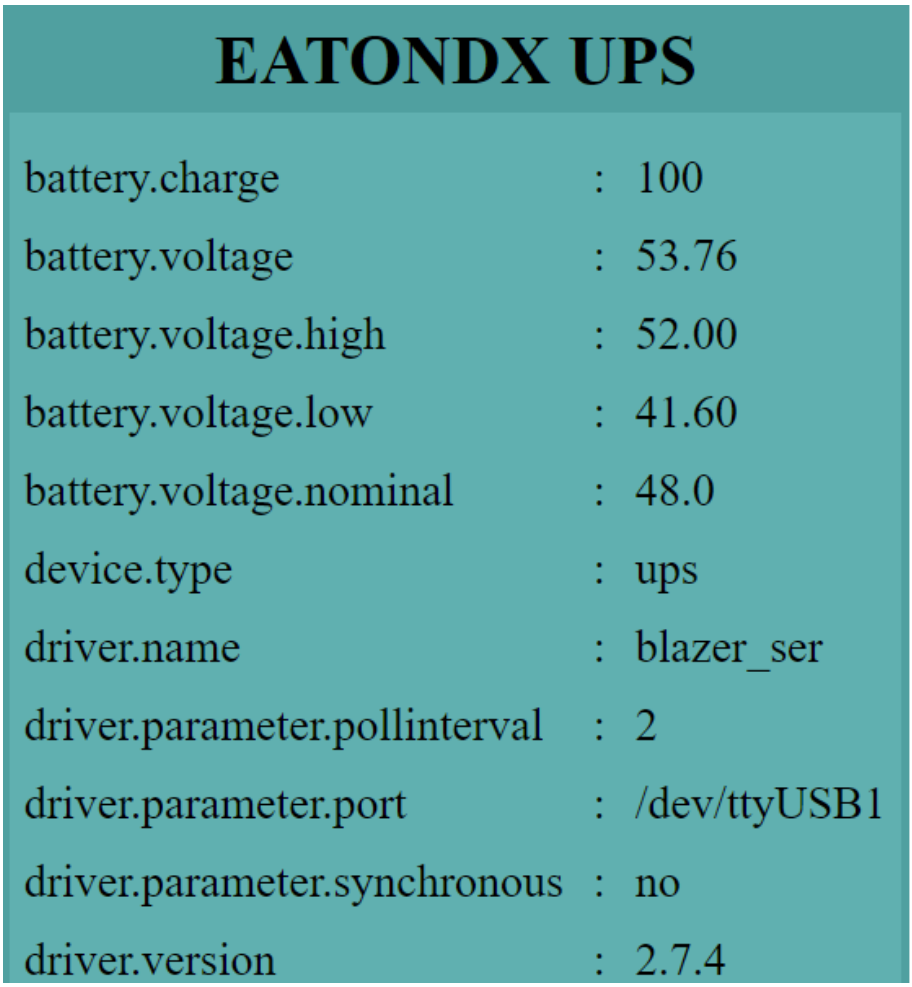

Figura 5-8.: visualización de parámetros eléctricos por nut-cgi

5.4. Software de Acceso remoto VNC

VNC es un software cliente servidor, que se habilito en la configuración inicial de la Raspberry Pi. VNC permitió el acceso remoto para la ejecución de las pruebas de operación y configuración de NUT monitor y las pruebas de conexión del dispositivo electrónico para el sistema de registro de

parámetros eléctricos y del entorno. Esto evitó el uso de periféricos como pantallas teclado y mouse en el embebido. Si bien no la podemos considerar como una herramienta de monitoreo, se hace su mención para indicar que permitió la conexión remota sobre el sistema operativo, facilitando el uso de las herramientas o formas de monitoreo locales.

Después de las pruebas realizadas con NUT Monitor y NUT-CGI, se pudo concluir que si bien estas herramientas aportaron características nuevas al sistema de monitoreo, se seguía presentando la deficiencia de una base de datos donde almacenar los registros tomados por el software, tampoco el programa poseía un registro en tiempo real de los eventos ocurridos, adicional a esto se sumó el no poder integrar el sistema de registros de parámetro eléctricos y del entorno, dadas estas condiciones, se procede a realizar la implementación de herramientas externas acondicionadas a cumplir las metas del prototipo.

5.5. Monitoreo con software Munin

Esta herramienta nos permitió el monitoreo y registro de los parámetros de la UPS y su visualización a través de la web, para su utilización era necesaria la instalación del servidor apache el cual ya habíamos instalado y un agente CGI, adicional a esto, la implementación de munin implicaba el uso de complementos o plugins, la utilización de munin implico la instalación de un servidor llamado Munin Server y un nodo llamado Munin Nodo, donde quedaron los registros de monitoreo guardados. Los resultados con Munin fueron favorables para visualizar los datos entregados por la UPS, la herramienta no permitió la integración de los datos suministrados por el dispositivo electrónico de medida de las variables externas que se diseñó, las gráficas de Munin permiten poca modificación y el periodo de muestreo era de cada 5 minutos, a continuación se muestra el registro gráfico que realizamos con munin mediante el cual realizamos un monitoreo web, se puede apreciar en ella las medidas de voltaje de entrada y salida alternos, voltaje de baterías y carga de la UPS.

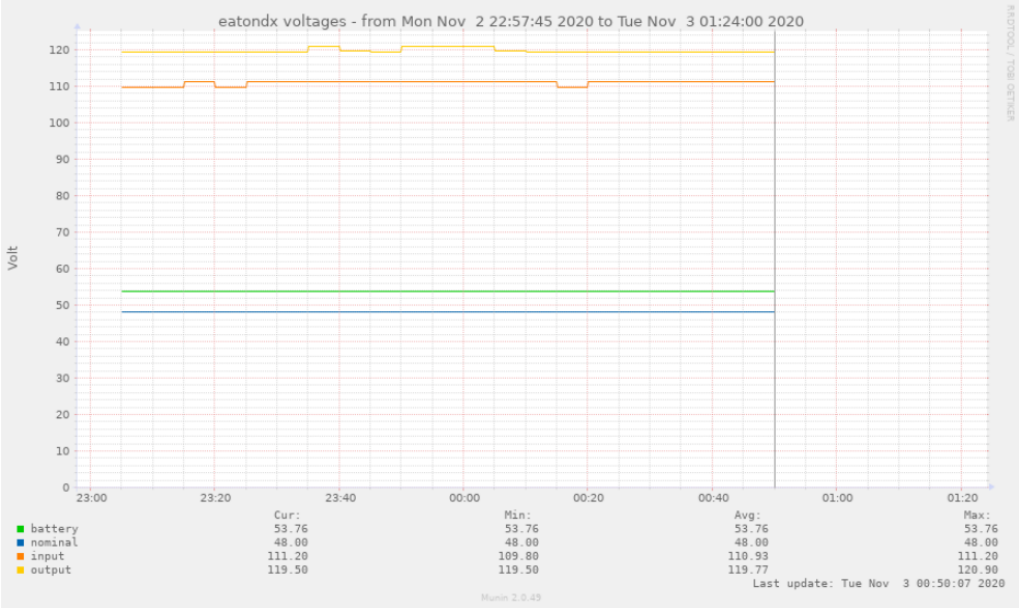

Figura 5-9.: Registro gráfico de los parámetros eléctricos realizado por Munin

5.6. Monitoreo con Grafana

Grafana termino siendo la herramienta en la etapa final que permitió la integración de todo el sistema , para comprender el funcionamiento de Grafana tenemos que tener en claro que esta herramienta trabaja de la mano con la base de datos influxDB ya que esta recibe la información enviada por la UPS, y por el dispositivo electrónico de monitoreo, no sin antes haber pasado por un proceso de transformación y acondicionamiento realizado por un scrip en PHP y un sistema de recepción de datos series de Python, a continuación detallamos como se realizó toda la ejecución de este proceso.

5.6.1. Instalación de Grafana

Inicialmente se añadieron los sources o fuentes de datos antes de proceder con la descarga de Grafana, posteriormente se creó un archivo llamado Grafana.list y se ingresó en una dirección url, y luego se ejecutó un comando la secuencia de pasos anteriormente realiza se evidencia en la siguiente gráfica.

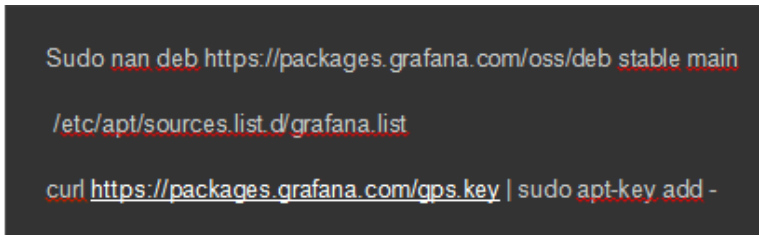

Figura 5-10.: secuencia de pasos previos a la instalación de Grafana

Luego se procedió al proceso de instalación de Grafana, se ejecutó la siguiente línea de comandos:

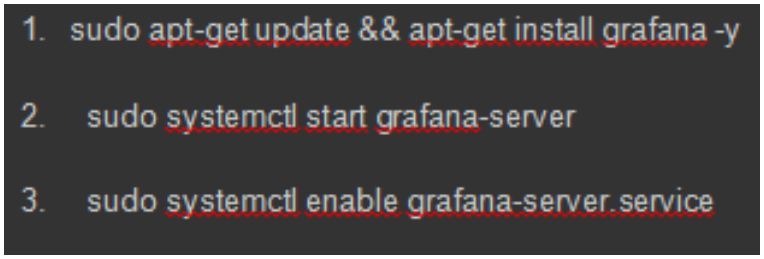

Figura 5-11.: Instalación de Grafana

5.7. Sistema de Captura de Datos

En el capítulo 3 se habló del diseño del dispositivo electrónico para el registro de las variables eléctricas y del entorno, este dispositivo permitió recibir toda la información proveniente de los sensores y trasmitir los datos vía puerto serie, sin embargo, su función más allá de ese proceso era limitada. El dispositivo por sí solo no realizaba ningún tipo de registro ni adquisición

de datos, dado lo anterior se procedió a integrar el dispositivo a la Raspberry pi con el fin de realizar una base de datos que guardará los registros tomados por los sensores.

Sin embargo, la base de datos, necesito de un mecanismo de captura de datos para su posterior almacenamiento, este mecanismo se realizó con la ejecución de 2 Scrips, uno de Python y uno en PHP. inicialmente lo que hace el scrip de Python es conectarse al puerto serie del arduino nano, luego ejecuta un único comando el cual imprime los datos enviados por el arduino, pero lo hace una única vez a continuación vemos el scrip de Python:

```
 GNU nano 3.2                                    prueba-puerto-serie.py

#!/usr/bin/php
import serial
import time
import requests
puertoSerial = serial.Serial('/dev/ttyUSB0', 115200)
datos = puertoSerial.readline()
print(datos)
```

Figura 5-12.: Scrip de Python para capturar datos una única vez

Luego el scrip de PHP ejecuta el scrip de python de forma continua, pero capturando los datos para enviarlos a la base de datos que previamente creamos en influxDB, por lo cual tendríamos en influx dos bases de datos, el proceso de creación de la nueva base de datos es similar al explicado en el capítulo 2 solo que posee un nombre distinto. A continuación mostramos parte del scrip realizado en PHP.

```
 GNU nano 3.2

#!/usr/bin/php
<?php
$command = "sudo python";
$args = "/opt/prueba-puerto-serie.py";
$tagsArray = array("VIN","VOUT", "Temp1", "Temp2","Humedad", "Irms", "Vdc");
//do system call
$call = $command." ".$args;
$output = shell_exec($call);
//parse output for tag and value
foreach ($tagsArray as $tag) {
preg_match_all("/".$tag."\s*:\s([\d|\.]+)/si",$output, $match);
//send measurement, tag and value to influx
//print_r($match[1]);
//print( $tag[1]);
sendDB($match[1], $tag);
//print_r($match[1]);
//print_r($output);
}
//end system call

function sendDB($val, $tagname) {
```

Figura 5-13.: Scrip de PHP para capturar datos

5.8. Automatización de scrip PHP

El siguiente proceso es automatizar el scrip de PHP para que se ejecute automáticamente desde que inicie el sistema operativo, eso lo realizamos con el comando crontab – esta aplicación se usa para ejecutar automáticamente comandos y scrips a determinada fecha y hora, en el proyecto se utilizó para automatizar los scrip que realizan las capturas de datos de la información enviada por NUT y el arduino, en la siguiente imagen se especifica la configuración que realizamos.

```
#
# m h  dom mon dow   command
* * * * * root /usr/bin/php /opt/code.php; /opt/example.php >/dev/null 2>&1
#* * * * * root /usr/bin/php /opt/example.php >/dev/null 2>&1
```

Figura 5-14.: Scrip de PHP para inicio automático

5.9. Recepción de datos y gráficas

Luego de que se ejecutó el archivo PHP de forma automática, y los datos fueron almacenados por influxDB, estos datos fueron enviados a Grafana. En Grafana se configuro InfluxDB para la recepción de estos. En la pestaña de configuración de Grafana se pudo observar las dos fuentes de datos de influxDB, en la siguiente imagen se visualiza:

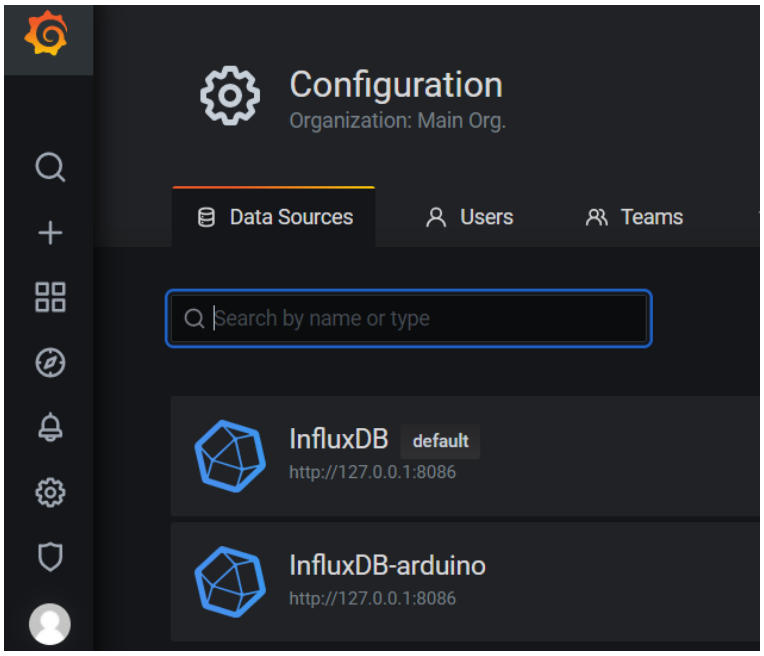

Figura 5-15.: Configuración de Grafana, se observan las dos fuentes de datos de influxDB

Esta fuente de datos se configuro para escuchar todos los datos provenientes de la dirección previamente asignada y de las bases de datos que creamos, en este caso llamadas UPS, en la imagen podemos ver:

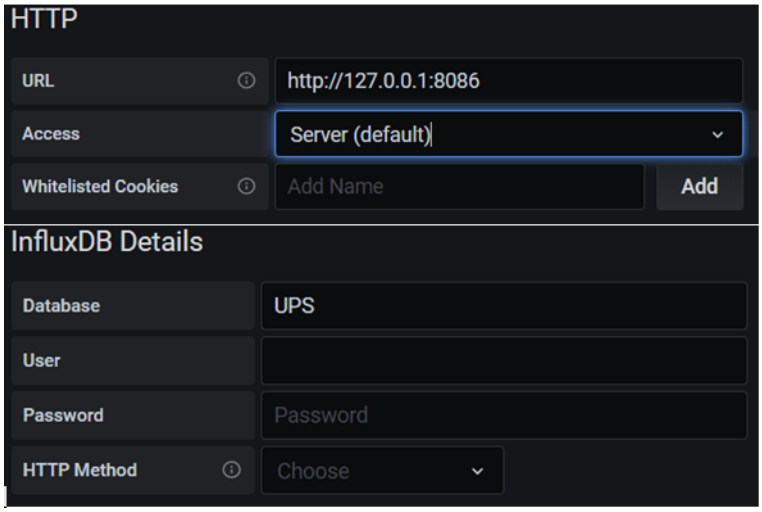

Figura 5-16.: configuración de dirección IP y fuente de datos

Luego de haber configurado el dashboard de Grafana y asignado los iconos a cada variable se obtuvieron los resultados de visualización los cuales podemos apreciar en las siguiente gráfica:

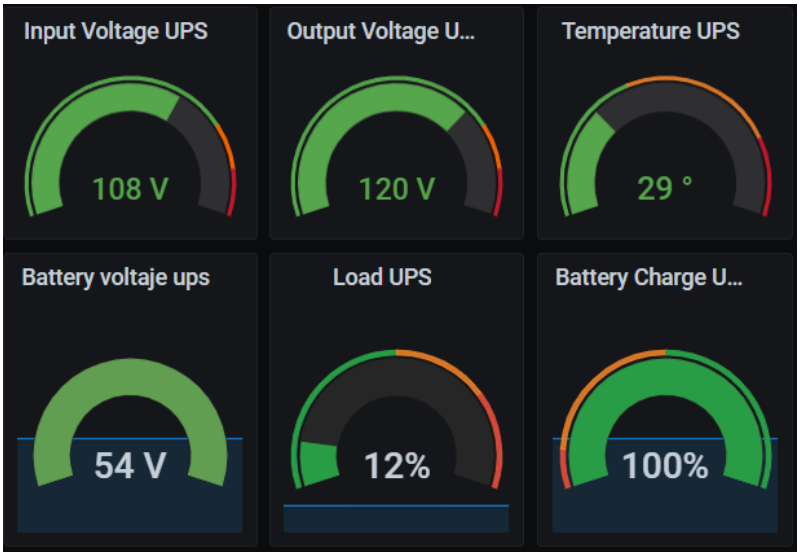

Figura 5-17.: Datos métricos del UPS visualizados

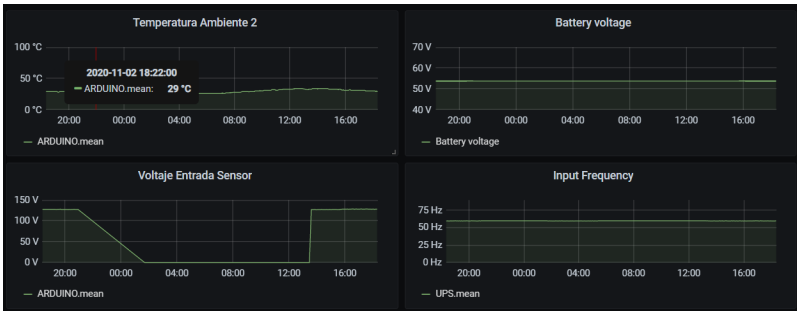

Figura 5-18.: Datos representados en gráficas de tiempo

Figura 5-19.: Interfaz integrada monitoreando datos del UPS y datos del Arduino Nano

5.10. Internet de las cosas

Se hicieron pruebas del prototipo graficando los valores en la aplicación thing speak de Matlab, sin embargo, esta aplicación posee la limitante de solo permitir 8 variables provenientes de 8 sensores distintos, en muestro caso se implementó la medición de 13 sensores por lo cual el sistema para monitorear a través de thing speak no permitía una integración de todas las variables, las pruebas se realizaron midiendo solo los datos de dos sensores acoplados a la Raspberry Pi.

5.11. Notificaciones

Las notificaciones corresponden a las alertas o alarmas o estados de operación que se generan y son enviadas por algún método de comunicación como por ejemplo el correo electrónico, a continuación, presentamos las notificaciones disponibles del prototipo implementado:

5.11.1. Notificaciones usando Grafana

La herramienta Grafana incorpora un sistema de notificaciones mediante los cuales se configura unos parámetros mínimos o máximos, cualquier valor fuera de este umbral se notifica por correo electrónico, la siguiente es el panel de alarmas y notificaciones:

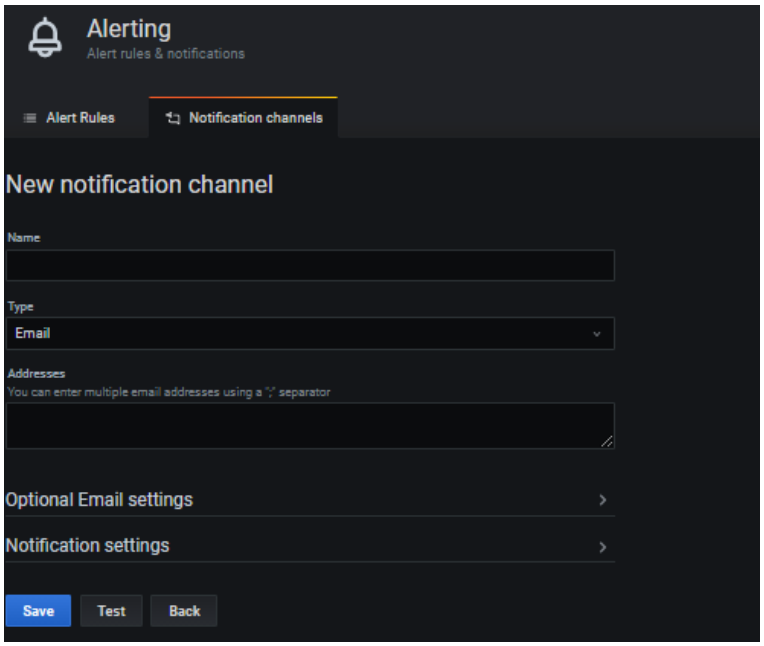

Figura 5-20.: Panel de notificaciones de Grafana

5.11.2. Notificaciones implementando NUT

El software NUT provee un sistema de notificaciones propios, donde indica el estado de operación del UPS, a continuación se muestra la interfaz o el archivo de configuración ubicado en la ruta **/etc/nut/upsmon.conf**

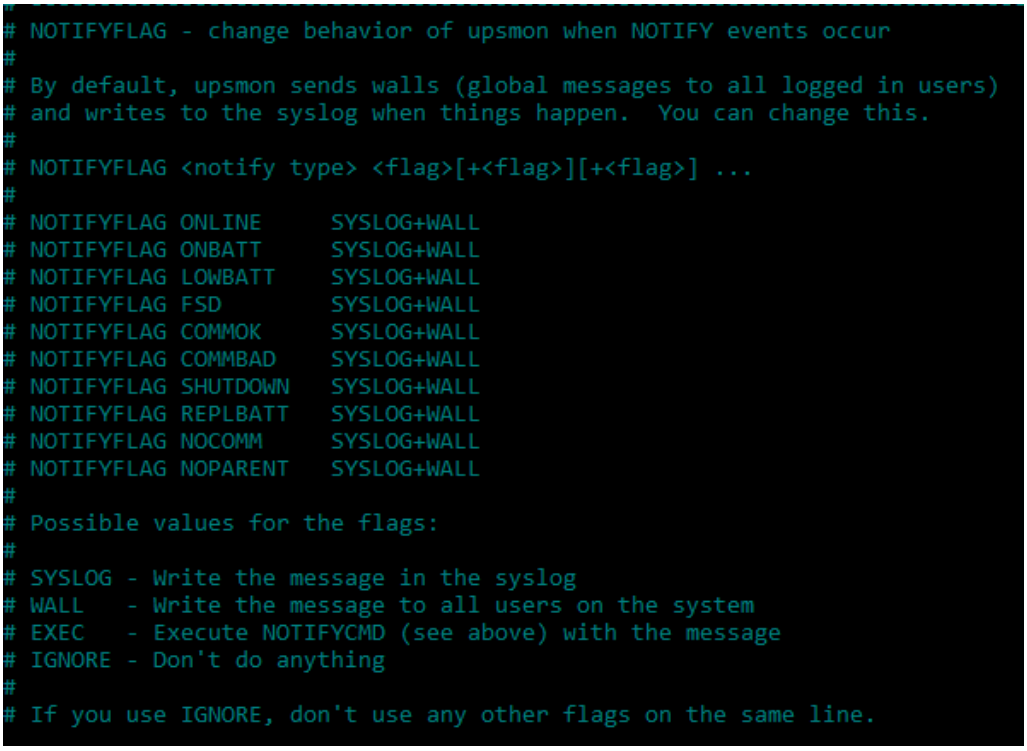

Figura 5-21.: Scrip de configuración de notificaciones de NUT

6. Análisis y Comparación de Costos

6.1. Comparación costos de adquisición del sistema de monitoreo

Las siguientes tablas nos muestra los costos comerciales de las tarjetas de monitoreo, y del prototipo implementado, hay que tener en cuenta que los valores expuestos de las tarjetas de red del fabricante corresponden a una cotización del día 6 de agosto de 2020, por lo cual el cálculo en pesos colombianos se hace con el valor del dólar en esa fecha que fue de:

1 USD = 3,775.95 COP

Descripción	Valor c/u (US)	IVA (US)	Valor total (US)
SNMP Web Card	387	73.53	460.53
Sensor temperatura	138	26.22	164.22
Tarjeta relay	434	82.46	516.46
Total monitoreo de fabricante			1,141.21

Tabla 6-1.: Valores comerciales de las tarjetas del sistema de monitoreo del fabricante

El costo total el pesos colombianos para la implementación del sistema de monitoreo de UPS por parte del fabricante tiene un costo de $4,310.000 COP.

Costo total del proyecto:

Descripción	Valor c/u (COP)
Raspberry Pi	192.000
Instrumentación	155.300
Costo Total Sistema de Monitoreo	347.300
Banco de pruebas	120.000
Costo total del proyecto	467.000

Tabla 6-2.: Valores comerciales para la implementación del proyecto

6.1.1. Análisis comparativo de Costos

Antes de realizar el análisis comparativo hay que tener en cuenta que el banco de pruebas no es un gasto necesario para la implementación del prototipo, este se hizo con fines didácticos y para demostrar la viabilidad del proyecto. Como se puede ver en ambas tablas, el sensor más económico vendido por el fabricante es el sensor de temperatura, pero este no se puede utilizar solo, se requiere de la tarjeta de red, si sumamos el costo de ambas tarjetas tendríamos:

Descripción	Valor c/u (US)	IVA (US)	Valor total (US)
SNMP Web Card	387	73.53	460.53
Sensor de temperatura	138	26.22	164.22
Total en COP			2,359,024.75

Tabla 6-3.: Valores comerciales de las tarjetas de red y el sensor de temperatura

Al realizar la comparación con el precio del sistema de monitoreo y al hacer una diferencia tendríamos:

Descripción	Valor c/u (COP)
SNMP Web Card + Sensor de temperatura	2,359,024.75
Sistemas de monitoreo	347,300
Valor sistema de monitoreo	347.300
Diferencia total en los sistemas de monitoreo	2,011,724.75

Tabla 6-4.: Comparativa de precios de ambos sistemas de monitoreo

Podemos ver claramente que al implementar el sistema de monitoreo diseñado existe un ahorro de 2,011,724.75 pesos o lo que seria un ahorro equivalente del 85 %. Por lo cual el sistema si reduce los costos de monitoreo, además de brindar opciones adicionales como monitoreo IoT, medición de parámetros físicos , y capacidad para nuevas funciones.

7. Conclusiones

Las pruebas realizadas en la UPS en las que se sometieron el UPS a trabajar en distintos modos de operación permitieron demostrar que el software NUT es una alternativa confiable de monitoreo, dando credibilidad al sistema de monitoreo implementado.

El sistema de instrumentación por su parte demostró ser una alternativa de monitoreo confiable ya que pudo registrar los parámetros eléctricos y del entorno los cuales coincidieron con las mediciones del software NUT.

El sistema implementado permitió monitoreo remoto y monitoreo local con opciones de almacenamiento de datos tanto localmente como en la nube.

El Análisis final de costos pudo demostrar que el sistema permitió un ahorro hasta del 80 % en comparación con las soluciones propuestas por el fabricante, por lo cual el sistema de monitoreo si es una alternativa de monitoreo económica orientada a reducir costos.

8. Trabajos futuros

A continuación, se proponen los trabajos futuros que se pueden realizar implementando el sistema de monitoreo anteriormente descrito

1. **Diseño de un panel de notificaciones y monitoreo local:** Consiste en dotar al prototipo actual, de una pantalla incorporada donde se puedan visualizar todos los parámetros eléctricos y del entorno, adicional el sistema debe poseer un buzzer o zumbador o alarma audible que se active indicando que ocurre un fallo de operación o cuando un parámetro crÍtico este fuera de rango.

2. **Diseño e implementación de un sistema autónomo de arranque de generador eléctrico:** Debido a que el prototipo quedo con todo su puertos GPIO libre, se pueden configurar estos para que cuando el sistema detecte un fallo de energía, envíe la señal de encendido a un generador eléctrico, adicional de poseer un sistema que mida el nivel de combustible del tanque de reserva.

3. **Desarrollo de un sistema embebido destinado a la medición de la calidad de la energía:** Consiste en ampliar la capacidad de medida del equipo, dotarlo de mejores sensores y configurarlo para realizar un dispositivo que este en la capacidad de medir la calidad de energía del sitio donde este sea implementado.

Anexos

A. Anexo I: General

Para facilitar la consulta, análisis y reproducción, de todas las actividades descritas en el presente proyecto, la base de datos y los programas desarrollados se encuentran disponibles a través del siguiente enlace:

https://drive.google.com/drive/folders/1gcCAazuv3Qi-2N1eTdEZUQuPFPtKHhsE?usp=sharing

B. Anexo II: Glosario Marco Teórico

- Monitoreo Remoto: Si bien el monitoreo remoto se define como la capacidad de visualizar datos o cualquier información desde cualquier lugar, se aclara que en este proyecto la palabra monitoreo remoto abarca la noción de tener acceso a visualizar esos datos usando un navegador o una aplicación, dentro de una misma red LAN o fuera de ella.
- Python: Lenguaje de programación de alto nivel que incorpora múltiples tipos de programación de fácil legibilidad.
- PHP: Es un lenguaje de programación ampliamente utilizado en entornos web, tiene como ventaja ser de fácil uso, y se puede ejecutar como scrip para correr determinados procesos.
- Influxdb: Base de datos que incorpora almacenamiento de series temporales , posee alta capacidad de almacenamiento , es de código abierto y permite integrar funciones.
- CGI: es una tecnología web que permite transferir datos solicitados por un cliente y un programa que este alojado en un servidor.
- VNC: herramienta de acceso remoto, posee una topología cliente servidor y es compatible con distintos sistemas operativos.
- Apache 2: Utilizado comúnmente en servidor de servicios de internet en plataformas de código abierto o en las que utilizan el protocolo http, posee base de datos y permite fácilmente su configuración.

Bibliografía

[1] Neil Rasmussen - Diferentes tipos de sistemas UPS - 2003 American Power Conversión - Pag 3

[2] Neil Rasmussen - Diferentes tipos de sistemas UPS - 2003 American Power Conversión - Pag 4

[3] Abad Domingo, A. (2013). Redes locales. McGraw-Hill España. https://ezproxy.uan.edu.co:2830/es/ereader/bibliouan/50228?page=40

[4] Corona Ramírez, L. G. Abarca Jiménez, G. S. y Mares Carreño, J. (2016). Sensores y actuadores: aplicaciones con Arduino. Grupo Editorial Patria. https://ezproxy.uan.edu.co:2830/es/ereader/bibliouan/39464?page=184

[5] Escalona, I. (2007). Transductores y sensores en la automatización industrial. El Cid Editor. https://ezproxy.uan.edu.co:2830/es/ereader/bibliouan/34463?page=14

[6] Lidong Fu y Bin Zhang, .El diseno e implementación de un sistema de monitor y control de UPS", Actas de la Conferencia Internacional de 2011 sobre Ciencias de la Computación y Tecnologia de Redes , Harbin, 2011, pp. 1322-1325, doi: 10.1109 / ICCSNT.2011.6182204.

[7] Y. Sannan and W. Shaoxu, "Parameter Measurement through Network Based on Embedded System," 2011 International Conference on Computer Distributed Control and Intelligent Environmental Monitoring, Changsha, 2011, pp. 107-110, doi: 10.1109/CDCIEM.2011.463.

[8] P. Alqinsi, I. J. Matheus Edward, N. Ismail and W. Darmalaksana, IoT-Based UPS Monitoring System Using MQTT Protocols," ¨ 2018 4th International Conference on Wireless and Telematics (ICWT), Nusa Dua, 2018, pp. 1-5, doi: 10.1109/ICWT.2018.8527815.

[9] T. Adiono, M. Y. Fathany, S. Fuada, I. G. Purwanda and S. F. Anindya, .A portable node of humidity and temperature sensor for indoor environment monitoring," 2018 3rd International Conference on Intelligent Green Building and Smart Grid (IGBSG), Yi-Lan, 2018, pp. 1-5, doi: 10.1109/IGBSG.2018.8393575.

[10] S. Balamurugan and D. Saravanakamalam, .Energy monitoring and management using internet of things", 2017 International Conference on Power and Embedded Drive Control (ICPEDC), Chennai, 2017, pp. 208- 212, doi: 10.1109/ICPEDC.2017.8081088.

[11] Roa Buendía, J. F. (2013). Seguridad informática. McGraw-Hill España. https://ezproxy.uan.edu.co:2830/es/ereader/bibliouan/50243?page=69

[12] Salas Arriarán, S. (2015). Todo sobre sistemas embebidos: arquitectura, programación y diseño de aplicaciones prácticas con el PIC18F. Universidad Peruana de Ciencias Aplicadas (UPC). https://ezproxy.uan.edu.co:2830/es/ereader/bibliouan/41261?page=38.

[13] Ali M. Sadegh, Ph.D.; William M. Worek, Ph.D. Marks' Standard Handbook for Mechanical Engineers, 12th Edition. DIGITAL CONTROL SYSTEMS, Chapter (McGraw-Hill Education: New York, Chicago, San Francisco, Athens, London, Madrid, Mexico City, Milan, New Delhi, Singapore, Sydney, Toronto, 2018). https://ezproxy.uan.edu.co:2107/content/book/9781259588501/toc-chapter/chapter10/section/section84

[14] Daniel R. Tomal, Ph.D.; Aram S. Agajanian, Ph.D. Electronic Troubleshooting, Fourth Edition. Troubleshooting Embedded Microprocessor Systems, Chapter (McGraw-Hill Education: New

York, Chicago, San Francisco, Athens, London, Madrid, Mexico City, Milan, New Delhi, Singapore, Sydney, Toronto, 2014). https://ezproxy.uan.edu.co:2107/content/book/9780071819909/chapter/chapter12

[15] Dr. Simon Monk. Programación de la Raspberry Pi: Introducción a Python, segunda edición. Introducción, capítulo (McGraw-Hill Education: Nueva York, Chicago, San Francisco, Atenas, Londres, Madrid, Ciudad de México, Milán, Nueva Delhi, Singapur, Sydney, Toronto, 2016). https://ezproxy.uan.edu.co:2107/content/book/9781259587405/chapter/chapter1

[16] Robert Chin. Proyectos de sensores Arduino y Raspberry Pi para Evil Genius. ¿Qué es Raspberry Pi ?, capítulo (McGraw-Hill Education: Nueva York, Chicago, San Francisco, Atenas, Londres, Madrid, Ciudad de México, Milán, Nueva Delhi, Singapur, Sydney, Toronto, 2018). https://ezproxy.uan.edu.co:2107/content/book/9781260010893/toc-chapter/chapter1/section/section27

[17] Daniel R. Tomal, Ph.D .; Aram S. Agajanian, Ph.D. Solución de problemas electrónicos, cuarta edición. Resolución de problemas de sistemas de microprocesadores integrados, capítulo (McGraw-Hill Education: Nueva York, Chicago, San Francisco, Atenas, Londres, Madrid, Ciudad de México, Milán, Nueva Delhi, Singapur, Sydney, Toronto, 2014). https://ezproxy.uan.edu.co:2107/content/book/9780071819909/chapter/chapter12

[18] López Aldea, E. (2015). Arduino: guía práctica de fundamentos y simulación. RA-MA Editorial. https://ezproxy.uan.edu.co:2830/es/ereader/bibliouan/106492?page=64

[19] Sol Llaven, D. (2016). Sistemas operativos: panorama para la ingeniería en computación e informática. Grupo Editorial Patria. https://ezproxy.uan.edu.co:2830/es/ereader/bibliouan/40429?page=19-21.

[20] Manual de instrumentos electrónicos de Clyde F. Coombs Jr., tercera edición. Computadoras integradas en instrumentos electrónicos, capítulo (McGraw-Hill, 2000, 1995, 1972). https://ezproxy.uan.edu.co:2107/content/book/9780070126183/chapter/chapter10

[21] B. Balon and M. Simić, Üsing Raspberry Pi Computers in Education,"2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 2019, pp. 671-676, doi: 10.23919/MIPRO.2019.8756967.

[22] Castaño Ribes, R. J. (2013). Redes locales. Macmillan Iberia, S.A. https://ezproxy.uan.edu.co:2830/es/ereader/bibliouan/43257?page=300

[23] I. Schieferdecker, "Trustworthiness of Open Source, Open Data, Open Systems and Open Standards", 2012 IEEE 36th Annual Computer Software and Applications Conference, Izmir, 2012, págs. 82-82, doi: 10.1109/COMPSAC.2012.104.

[24] Gupta and R. Rani, .A rule based advisory system for repair and maintenance of UPS," 2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT), Coimbatore, 2017, pp. 1-6, doi: 10.1109/ICECCT.2017.8118042.

[25] Raya Cabrera, J. L. y Raya González, L. (2015). Sistemas informáticos. RA-MA Editorial. https://ezproxy.uan.edu.co:2830/es/ereader/bibliouan/62481?page=32

[26] https://networkupstools.org/documentation.html

[27] https://www.eaton.com/Eaton/ESeriesUPS/WinpowerSoftwareDownloads/index.htm

[28] https://grafana.com/

[29] Corona Ramírez, L. G. y Abarca Jiménez, G. S. (2019). Sensores y actuadores: aplicaciones con Arduino (2a. ed.). Grupo Editorial Patria. https://ezproxy.uan.edu.co:2830/es/ereader/bibliouan/121284?page=214

[30] Corona Ramírez, L. G. Abarca Jiménez, G. S. y Mares Carreño, J. (2016). Sensores y actuadores: aplicaciones con Arduino. Grupo Editorial Patria. https://ezproxy.uan.edu.co:2830/es/ereader/bibliouan/39464?page=188

[31] Llaneza González, P. (2018). Seguridad y responsabilidad en la internet de las cosas (IoT). Wolters Kluwer España. https://ezproxy.uan.edu.co:2830/es/ereader/bibliouan/58379?page=21 9788429020380_internetdelascosas.pdf (pag. 17 - 19)

[32] https://networkupstools.org/ddl/Eaton/

[33] Dawoud Shenouda Dawoud; Peter Dawoud, "Serial Communication Protocols and Standards RS232/485, UART/USART, SPI, USB, INSTEON, Wi-Fi y WiMAX", en Serial Communication Protocols and Standards RS232/485, UART/USART, SPI, USB, INSTEON, Wi-Fi y WiMAX, River Publishers, 2020, pp.i-xl.

[34] Bharat Heavy Electricals Limited. Handbook of Switchgears. CURRENT TRANSFORMERS AND VOLTAGE TRANSFORMERS, Chapter (McGraw-Hill, 2007). https://ezproxy.uan.edu.co:2107/content/book/9780071476966/chapter/chapter8

[35] Bharat Heavy Electricals Limited. Handbook of Switchgears. CURRENT TRANSFORMERS AND VOLTAGE TRANSFORMERS, Chapter (McGraw-Hill, 2007). https://ezproxy.uan.edu.co:2107/content/book/9780071476966/chapter/chapter8

[36] Deeksha Srivastava, Awanish Kesarwani, Shivani Dubey - Measurement of Temperature and Humidity by using Arduino Tool and DHT11

- 2018 - https://d1wqtxts1xzle7.cloudfront.net/58144752/IRJET-V5I12167.pdf

[37] H. Shen, J. Fu and Z. Chen, .[Em]bedded system of temperature testing based on DS18B20,"2006 International Technology and Innovation Conference (ITIC 2006), Hangzhou, 2006, pp. 2223-2226.