

**Tecnológico Nacional de México**

**Instituto Tecnológico de Chetumal**

Departamento de Sistemas computacionales

**Soluciones Energéticas para Redes de Sensores: Eficiencia y Sustentabilidad**

Ingeniería en Sistemas Computacionales

Presentan:

Rosales Huey Francisco Jareth

Tox Dzul Omar

Docente:

Zarina Maryela Basulto Alvarez

Asesor

Esquivel Pat Agustín

Fecha

25 de Mayo de 2025

# Agradecimientos

Agradecimiento de Francisco

Agradecimiento de Omar

# Resumen

Este proyecto aborda los desafíos energéticos en las redes de sensores inalámbricos (WSN), esenciales para aplicaciones como monitoreo ambiental, agricultura de precisión y automatización industrial. Las WSN enfrentan problemas de sostenibilidad debido al uso de baterías tradicionales con limitada duración, lo que incrementa los costos operativos y limita su escalabilidad en entornos remotos.

El objetivo principal es optimizar el consumo de energía de estas redes mediante la implementación de algoritmos eficientes, protocolos de bajo consumo y el uso de fuentes renovables como energía solar. Se busca prolongar la vida útil de los nodos sensores, reducir costos energéticos, disminuir la huella ambiental y mejorar la sostenibilidad operativa.

La metodología incluye un enfoque experimental para diseñar y probar soluciones en entornos controlados, combinando hardware optimizado, fuentes de energía renovables y técnicas de ahorro energético. Además, se analizan protocolos de comunicación eficientes para maximizar la funcionalidad de las WSN.

El proyecto tiene un impacto significativo en la sostenibilidad ambiental y en la expansión de las aplicaciones de las WSN, contribuyendo al desarrollo de tecnologías más autónomas, sostenibles y adaptables en sectores críticos como la agricultura, la salud y el monitoreo ambiental.

# Abstract

This project addresses the energy challenges in Wireless Sensor Networks (WSN), which are essential for applications such as environmental monitoring, precision agriculture, and industrial automation. WSNs face sustainability issues due to the use of traditional batteries with limited lifespan, leading to increased operational costs and restricted scalability in remote environments.

The primary goal is to optimize the energy consumption of these networks through the implementation of efficient algorithms, low-power protocols, and the use of renewable energy sources like solar. The aim is to extend the lifespan of sensor nodes, reduce energy costs, minimize environmental impact, and enhance operational sustainability.

The methodology involves an experimental approach to design and test solutions in controlled environments, combining optimized hardware, renewable energy sources, and energy-saving techniques. Additionally, efficient communication protocols are analyzed to maximize the functionality of WSNs.

This project has a significant impact on environmental sustainability and the expansion of WSN applications, contributing to the development of more autonomous, sustainable, and adaptable technologies in critical sectors such as agriculture, healthcare, and environmental monitoring.

# Índice

Contenido

[Agradecimientos 2](#_Toc210392116)

[Resumen 3](#_Toc210392117)

[Abstract 4](#_Toc210392118)

[Índice 5](#_Toc210392119)

[Índice de figuras 7](#_Toc210392120)

[Índice de tablas 9](#_Toc210392121)

[Introducción 10](#_Toc210392122)

[Antecedentes 12](#_Toc210392123)

[Planteamiento del problema 17](#_Toc210392124)

[Objetivo general de la investigación: 18](#_Toc210392125)

[Objetivos específicos 18](#_Toc210392126)

[Justificación 19](#_Toc210392127)

[Factibilidad 21](#_Toc210392128)

[Factibilidad técnica 21](#_Toc210392129)

[Factibilidad económica 21](#_Toc210392130)

[Factibilidad operativa 22](#_Toc210392131)

[Factibilidad ambiental 23](#_Toc210392132)

[Marco teórico 24](#_Toc210392133)

[Marco conceptual 24](#_Toc210392134)

[Redes de sensores inalámbricas (WSN en inglés) 24](#_Toc210392135)

[Nodo sensor 27](#_Toc210392136)

[Nodo central 29](#_Toc210392137)

[Gestión de Energía en Redes de Sensores Inalámbricos (WSN) 31](#_Toc210392138)

[Energía Renovable 33](#_Toc210392139)

[Protocolos de Comunicación 35](#_Toc210392140)

[LoRa 36](#_Toc210392141)

[XBee 37](#_Toc210392142)

[MQTT (Message Queuing Telemetry Transport) 38](#_Toc210392143)

[Protocolo de Control de Acceso al Medio (MAC) 40](#_Toc210392144)

[Estado del Arte 41](#_Toc210392145)

[Protocolos Mac de bajo consumo 41](#_Toc210392146)

[Estrategias de recolección de energía en WSN 41](#_Toc210392147)

[Algoritmos de ahorro de energía en WSN 47](#_Toc210392148)

[Modelado del Consumo Energético en Redes de Sensores Inalámbricas (WSN) 51](#_Toc210392149)

[Metodologías de Modelado Energético en Redes de Sensores Inalámbricas 55](#_Toc210392150)

[Metodología 58](#_Toc210392151)

[1. Enfoque de la Investigación 58](#_Toc210392152)

[Enfoque Cuantitativo: 58](#_Toc210392153)

[Enfoque Cualitativo: 58](#_Toc210392154)

[Pruebas de Programación y Comunicación: 59](#_Toc210392155)

[Pruebas de Hardware: 59](#_Toc210392156)

[2. Selección de la Muestra 59](#_Toc210392157)

[3. Instrumentos y Técnicas de Recolección de Datos 60](#_Toc210392158)

[4. Procedimiento o Desarrollo de la Investigación 60](#_Toc210392159)

[5. Análisis de Datos 61](#_Toc210392160)

[6. Validez y Confiabilidad 62](#_Toc210392161)

[7. Consideraciones Éticas 62](#_Toc210392162)

[Descripción de las actividades realizadas 63](#_Toc210392163)

[Adquisición de conocimiento y Desarrollo de prototipos básicos 63](#_Toc210392164)

[Generación de un prototipo base para mejorar 66](#_Toc210392165)

[Análisis de mejoras a realizar 69](#_Toc210392166)

[Mejoras al hardware 71](#_Toc210392167)

[Actualización de Microcontroladores: Arduino Nano para el Sensor y ESP32 para el Coordinador 71](#_Toc210392168)

[Actualización de la Fuente de Alimentación 73](#_Toc210392169)

[Mejoras al Software 76](#_Toc210392170)

[Desarrollo de la Librería CodecWSN 76](#_Toc210392171)

[Desarrollo de la librería AdaptiveTXWSN 78](#_Toc210392172)

[Mejoras de la Sinergia Hardware-Software 80](#_Toc210392173)

# Índice de figuras

[Figura 1. Módulo de carga de las baterías 66](#_Toc210731438)

[Figura 2. Nodo Sensor Figura 3. Nodo coordinador 70](#_Toc210731439)

-

# Índice de tablas

[Tabla 1. Pines actualizados del nodo sensor con Arduino Nano 73](#_Toc210377121)

[Tabla 2. Pines actualizados del nodo coordinador con Esp32 73](#_Toc210377122)

[Tabla 3. Prototipo con regulador MP1584 y jumpers wires 75](#_Toc210377123)

# Introducción

Las redes de sensores inalámbricos (WSN) han emergido como una tecnología clave para la recopilación de datos en tiempo real en una variedad de aplicaciones, que incluyen el monitoreo ambiental, la automatización industrial y la agricultura de precisión. Estas redes, compuestas por nodos sensores interconectados, son esenciales para recopilar y transmitir información relevante, contribuyendo a la toma de decisiones basadas en datos y a la optimización de procesos en diversos sectores.

A pesar de los avances en la implementación de WSN, uno de los desafíos más importantes que enfrentan es la gestión eficiente de la energía. Los nodos sensores, por lo general, están ubicados en zonas de difícil acceso, lo que limita la posibilidad de reemplazar o recargar las baterías. Cuando un nodo agota su fuente de energía, se convierte en un "nodo muerto", generando puntos de falla en la red que pueden interrumpir el flujo de datos y afectar el rendimiento general del sistema. Esta problemática subraya la necesidad de investigar métodos para extender la vida útil de los nodos y mantener la sostenibilidad operativa de las redes.

Numerosos estudios, como los de Guo & Healy (2014) y Orlando & Asqui (2023), han abordado el desafío energético en las WSN, analizando modelos matemáticos de descarga de batería, recolección de energía y la optimización de topologías de red. Asimismo, investigaciones como las de Rivera (2020) y Gordillo (2021) han explorado sistemas auto configurables y basados en LPWAN que maximizan la eficiencia mediante el uso de fuentes de energía alternativas y algoritmos de machine learning. Estos antecedentes resaltan la importancia de desarrollar estrategias de ahorro energético que combinen hardware optimizado, software eficiente y fuentes de energía renovables.

El presente proyecto de investigación se centra en explorar y evaluar diversas soluciones para la gestión energética en WSN, desde la implementación de protocolos de comunicación de bajo consumo hasta la integración de sistemas de recolección de energía solar. Al combinar estos enfoques, buscamos proponer una solución integral que reduzca el consumo energético y potencie la sostenibilidad de las WSN, facilitando su despliegue comercial e industrial en entornos donde el acceso a fuentes de energía convencionales es limitado.

Este estudio no solo tiene el potencial de optimizar el uso de la energía en las redes de sensores, sino que también contribuye a sentar las bases para un futuro más sostenible en el ámbito de las WSN. La implementación de soluciones energéticas eficientes podría impulsar su adopción en campos tan críticos como el monitoreo ambiental y la gestión de recursos naturales, abriendo nuevas oportunidades para la innovación tecnológica y el desarrollo sustentable.

# Antecedentes

Las redes de sensores inalámbricos (WSN, por sus siglas en inglés) son sistemas interconectados de dispositivos sensores que recopilan, transmiten y procesan datos de manera autónoma. Estas redes tienen aplicaciones amplias en sectores como el monitoreo ambiental, la automatización industrial, agricultura de precisión, la gestión de ciudades inteligentes y la salud.

En el monitoreo ambiental, las WSN se utilizan para observar fenómenos naturales, evaluar condiciones meteorológicas y medir contaminantes en el aire o agua. Estas redes son frecuentemente desplegadas en áreas remotas, donde la reposición de energía se convierte en un obstáculo importante. Al mejorar la eficiencia energética, se extiende la vida útil de los sensores y se asegura la recolección continua de datos críticos para el análisis ambiental.

En aplicaciones industriales, las WSN desempeñan un rol en la automatización de procesos, el monitoreo de maquinaria y la seguridad. El ahorro energético en estos entornos reduce los costos de mantenimiento y reposición de baterías, haciendo que el sistema sea más rentable y menos propenso a interrupciones

La energía es un factor determinante para la sostenibilidad y eficiencia de las redes de sensores inalámbricos. En sectores como la agricultura de precisión, los sensores permiten monitorear el estado del suelo, la humedad, la temperatura y otros factores clave, mejorando la toma de decisiones y optimizando el uso de recursos. Sin embargo, las operaciones continuas de estos sensores dependen de una fuente de energía confiable, lo cual es difícil de mantener sin soluciones de recolección u optimización energética.

Uno de los retos más significativos que enfrentan las WSN es la gestión eficiente de la energía, dado que los nodos sensores suelen estar en ubicaciones remotas donde el acceso a fuentes de energía continua es limitado, debido a esto es que si un nodo se queda sin batería es considerado muerto ya que a pesar de que todos sus componentes electrónicos están en perfecto estado llega a ser costoso el recuperar los nodos y reemplazar su batería, esto desencadena puntos muertos en la red de monitoreo lo que dependiendo del campo de aplicación puede ser desde un pequeño retraso hasta un grave tope a futuro

Para abordar este desafío, es esencial estudiar diversas fuentes de energía, incluyendo fuentes renovables como la solar o la eólica para implementar en sistemas de recolección de energía, y optimizar el software y hardware de los sensores para tanto reducir el consumo energético innecesario como implementar baterías eficientes dependiendo del problema a tratar por cada WSN. Las estrategias de ahorro energético que incluyan baterías de bajo consumo, métodos de recolección de energía del entorno y protocolos eficientes de comunicación representan avances fundamentales en este campo

Guo & Healy (2014a) realizan un análisis de las limitaciones energéticas en las redes de sensores inalámbricos (WSN). Plantean que la energía es un recurso limitado en los nodos debido a su pequeño tamaño y que la batería, como fuente principal de energía, falla antes del tiempo proyectado. Proponen analizar modelos matemáticos para caracterizar la descarga de la batería y examinan tecnologías de recolección de energía, ste estudio es crucial porque establece la base del problema energético en las WSN, el cual intentamos mitigar en nuestra investigación al experimentar con fuentes de recolección de energía

Más recientemente, Orlando & Asqui (2023) abordan el tema de la eficiencia energética en las WSN en escenarios con despliegue aleatorio de nodos proponiendo estrategias para optimizar el consumo de energía, este trabajo ofrece aportes en la optimización energética mediante el uso de topologías estrella y de malla, aspectos que planteamos explorar al probar diferentes medios de transmisión y de implementación de topologías para el ahorro de energía, alineado a esto nos encontramos con el trabajo de Vera Pico et al.(2022) que, mediante una simulación, compara la eficiencia energética de diferentes configuraciones de red, este tema se alinea con nuestro objetivo de probar diferentes configuraciones de transmisión de datos

Otro trabajo a considerable en el campo del ahorro energético para las WSN es el de Mishra et al. (2022) que plantean una manera diferente para evitar consumir tanta energía ya que ellos implementan un paso antes del envío de los datos que es el comprimirlos antes de enviarlos ya que el envío de los datos es lo que más energía consume entonces al reducir el tamaño de los datos enviados se reduce el consumo energético, Mishra menciona que es importante considerar la naturaleza de los datos para saber si es óptimo utilizar éste método ya que es imprescindible que la compresión consuma menos energía que el ahorro que se hará, esto puede ser un punto de partida muy importante también para nosotros.

Otro estudio, Rivera (2020) implementó un prototipo de red de sensores auto configurable para la recolección de datos ambientales. Este sistema, compuesto por nodos secundarios y un nodo maestro, se comunica mediante radiofrecuencia para medir variables como temperatura, humedad y presión, transmitiéndolas a una aplicación de procesamiento. Este diseño incluye una red neuronal para predecir el estado de carga de la batería de cada nodo, con la capacidad de alternar entre una batería y un panel solar, lo cual maximiza la eficiencia energética. Asimismo, esta red ofrece soporte a un vehículo no tripulado para la adquisición de datos, mostrando así el potencial de las WSN en el monitoreo autónomo y sustentable del medio ambiente.

Gordillo (2021) desarrolló un sistema de monitoreo basado en redes LPWAN, con un Meta Sistema Operativo que permite recolectar y analizar datos de factores tales como rayos UV, CO₂, temperatura y humedad en invernaderos. Utilizando algoritmos de machine learning, el sistema identifica patrones de crecimiento y optimiza el riego, minimizando la dependencia de decisiones empíricas y maximizando la precisión en el cultivo. Las pruebas en un entorno controlado validaron la eficiencia del sistema, el cual recolecta y visualiza los datos mediante una interfaz gráfica, destacando la contribución de la tecnología WSN en la mejora de la agricultura de precisión.

Por último, cabe mencionar los avances en la optimización del consumo energético en redes de sensores inalámbricos implementados por Vizcaíno (2014) ya que gracias a ello se han llevado a estudiar arquitecturas y sistemas operativos especializados como TinyOS, MantisOS y Contiki. Estos estudios permitieron diseñar arquitecturas de energía que combinan recolección solar y supercondensadores, logrando una operación eficiente y autónoma. Asimismo, se analizó la viabilidad de recolección de energía de ondas de radio comerciales para alimentar nodos de bajo consumo, lo cual representa un recurso adicional de energía en condiciones adversas y asegura la funcionalidad de las redes de sensores.

Los antecedentes seleccionados para este proyecto ofrecen una base sólida para comprender los desafíos y las soluciones en la eficiencia energética de redes de sensores inalámbricos (WSN), tema central de nuestra investigación. La revisión de estudios previos revela cómo las WSN han sido aplicadas en sectores como el monitoreo ambiental, la automatización industrial y la agricultura de precisión, donde el ahorro y recolección de energía son esenciales debido a la operación en entornos de difícil acceso para la reposición de baterías. A partir de estas investigaciones, podemos explorar fuentes de energía renovables y configuraciones eficientes que potencien la sostenibilidad de las WSN. Los estudios de Rivera (2020) y Gordillo (2021) sobre redes auto configurables y basadas en LPWAN también destacan el uso de algoritmos avanzados y fuentes de energía alternativas, confirmando la relevancia de implementar tecnologías avanzadas para optimizar el consumo energético. Estos antecedentes son, por tanto, fundamentales para contextualizar y justificar las estrategias de ahorro y recolección de energía en nuestro proyecto de investigación.

# Planteamiento del problema

Las redes de sensores inalámbricos (WSN) son sistemas distribuidos que permiten la recolección y transmisión de datos en diversos entornos, incluyendo la agricultura, el monitoreo ambiental y la automatización industrial. Estas redes desempeñan un papel fundamental en el desarrollo del Internet de las Cosas (IoT), facilitando la toma de decisiones en tiempo real. Sin embargo, su operatividad se ve comprometida por un problema crítico: la eficiencia energética.

El problema principal de las WSN radica en la limitada capacidad de sus baterías y la dificultad de recargarlas en entornos remotos o de difícil acceso. Los nodos sensores dependen de estas baterías para su funcionamiento, y cuando se agotan, dejan de operar, generando interrupciones en la red y reduciendo su vida útil. A esto se suma que muchos protocolos de comunicación y software utilizados en las WSN no están optimizados, lo que provoca un consumo de energía mayor al necesario y afecta su sostenibilidad a largo plazo.

La falta de una gestión energética eficiente incrementa los costos operativos y limita la viabilidad de las WSN en aplicaciones a gran escala. Muchos proyectos quedan restringidos a la fase de prototipado debido a la imposibilidad de mantener una autonomía energética prolongada, lo que impide su implementación en escenarios reales. (Padilla, Quintero & Diaz, 2015)

Para abordar esta problemática, es necesario investigar estrategias que permitan optimizar el consumo energético de las WSN. Entre las soluciones viables se encuentran el uso de fuentes de energía renovable (como la solar) y la implementación de protocolos de comunicación de bajo consumo. Estas estrategias podrían extender la autonomía de las redes, reducir su impacto ambiental y potenciar su uso en sectores clave como la agricultura de precisión, el monitoreo ambiental y la industria como lo mencionan (Luis & Luis, 2019).

Dado el impacto de las WSN en múltiples áreas tecnológicas, abordar su problema energético es esencial para garantizar su operatividad a largo plazo. Esta investigación busca desarrollar soluciones integrales que permitan mejorar la sostenibilidad de estas redes y fomentar su adopción en el ámbito comercial e industrial.

## Objetivo general de la investigación:

Optimizar el consumo de energía en redes de sensores inalámbricas (WSN) mediante algoritmos de gestión energética, protocolos de bajo consumo y fuentes renovables, logrando una reducción del consumo en al menos 30% en un período de 6 meses de pruebas

## Objetivos específicos

* Evaluar en 3 meses el consumo energético en nodos sensores con baterías convencionales y fuentes renovables, comparando su autonomía mediante medición del tiempo de operación.
* Desarrollar e implementar en 4 meses un algoritmo de gestión energética que reduzca el consumo en al menos 20%, optimizando modos sleep y transmisión de datos
* Comparar en un periodo de 2 meses la eficiencia energética de diferentes estrategias de comunicación en WSN en distintos entornos.

## Justificación

Las redes de sensores inalámbricos (WSN) están compuestas por nodos autónomos alimentados por baterías, cuyo consumo de energía es un factor clave en su diseño. Esto ha impulsado la investigación en la optimización del consumo energético, explorando técnicas como la recolección de energía y la mejora en la eficiencia de las baterías. En esta investigación, se propone implementar algoritmos que permitan mejorar su rendimiento, además de comparar y analizar distintos protocolos de comunicación para identificar aquel que ofrezca el mejor ahorro de energía en función de factores como la distancia y las condiciones operativas. Asimismo, se implementa una fuente de alimentación que combina energía solar y supercondensadores, permitiendo una mayor autonomía para los dispositivos (Vizcaíno, 2014).

Este proyecto tiene un amplio impacto en el ámbito tecnológico especialmente en el campo del IoT ya que da paso a que las WSN tengan mayor facilidad para planificar su autonomía energética lo que da paso a que más proyectos puedan ser implementados de forma amplia, autónoma e incluso de forma comercial ya que estaría dejándose de lado una parte de los obstáculos y la preocupación referente a la fuente energética.

La manera en que ampliaría el campo comercial de las WSN es que habrá una mayor cantidad de procesos que se puedan realizar de forma autónoma con las redes de sensores a un menor costo.

En el ámbito social, este proyecto busca facilitar el monitoreo de las redes de sensores, permitiendo una gestión más eficiente y reduciendo la preocupación de los usuarios por su mantenimiento. Gracias a una mayor autonomía, se minimiza la necesidad de intervenciones frecuentes, lo que no solo optimiza el uso de recursos, sino que también mejora la experiencia de los usuarios al brindar un sistema más confiable y de fácil operación.

En el ámbito ambiental, este proyecto busca reducir el uso de baterías convencionales desarrollando soluciones que apliquen energía obtenida de fuentes renovables (eólica, solar, etc) además de optimizar el consumo de energía.

## Factibilidad

### Factibilidad técnica

El proyecto realizado presenta un enfoque sólido en la optimización y aplicación de pruebas de redes de sensores inalámbricos. Respaldado por los siguientes puntos clave:

1. Disponibilidad tecnológica:

Las herramientas necesarias están claramente definidas y accesibles, incluyendo:

* 1. Funciones de bajo consumo, como .sleep().
  2. Protocolos de comunicación eficientes en consumo energético (LoRa, XBee, MQTT).

1. Pruebas de hardware y software:  
   La implementación y optimización del código es viable con el conocimiento disponible. Sin embargo, el éxito del proyecto dependerá de:
   1. La integración correcta de los módulos de carga que permitan energías renovables.
   2. La optimización de algoritmos que permitan maximizar la eficiencia energética.
2. Complejidad Técnica:

Los principales riesgos identificados posterior al análisis incluyen:

* 1. Fallos en las pruebas de los nodos sensores.
  2. Dificultades en la integración de los protocolos de comunicación.

No obstante, estos riesgos pueden mitigarse mediante iteraciones y pruebas controladas en un entorno experimental.

En conclusión, la factibilidad técnica para la realización el proyecto es viable, siempre que se asegure la compatibilidad de los sensores con las estrategias propuestas.

### Factibilidad económica

El análisis económico del proyecto se fundamenta principalmente en los siguientes aspectos:

1. Costos iniciales:

En este apartado se contemplan los recursos necesarios para la fase de pruebas, tales como:

* 1. Sensores y hardware adicional (fuentes renovables, baterías, etc.).
  2. Kits de desarrollo.

1. Optimización energética a largo plazo:

Una de las metas principales del proyecto es reducir los costos operativos mediante:

* 1. La optimización del consumo energético, lo que refuerza la rentabilidad del proyecto.

1. Escalabilidad
2. Si bien la implementación de redes con 5-10 nodos resulta económica en la fase de prototipo, se debe considerar que los costos podrían incrementarse significativamente para despliegues industriales más grandes.

El proyecto es económicamente viable a nivel de prototipo. Sin embargo, será necesario desarrollar un presupuesto detallado para garantizar que los recursos sean suficientes durante las pruebas y evaluar la escalabilidad en etapas posteriores.

### Factibilidad operativa

El proyecto responde directamente a uno de los principales desafíos en redes de sensores: el consumo energético. A continuación, se desglosan los aspectos clave que refuerzan su viabilidad operativa:

1. Integración con operaciones existentes:  
   Las soluciones propuestas, como funciones de bajo consumo y protocolos de comunicación eficientes, están alineadas con aplicaciones prácticas en sectores clave como:
   1. Agricultura.
   2. Monitoreo ambiental.
   3. Automatización industrial.
2. Usuarios y entorno:  
   Los resultados esperados del proyecto serán de gran utilidad para:
   1. Desarrolladores de sistemas IoT.
   2. Operadores de redes de sensores, facilitando su aceptación e implementación.
3. Riesgos operativos:  
   Aunque el proyecto es prometedor, los siguientes riesgos podrían surgir:
   1. Fallos en la transmisión de datos.
   2. Limitaciones en la eficiencia en escenarios reales.  
      Estos riesgos requieren ajustes constantes y evaluaciones durante la fase de pruebas.

La factibilidad operativa es viable. Sin embargo, será esencial realizar pruebas exhaustivas en entornos reales para validar la aplicabilidad y efectividad de las soluciones propuestas.

### Factibilidad ambiental

El proyecto tiene un impacto positivo en el medio ambiente, destacándose por los siguientes aspectos:

1. **Uso de fuentes renovables**:  
   La integración de fuentes de energía renovable, como la solar, contribuye a:
   * Reducir el impacto ambiental.
   * Minimizar el uso de baterías desechables, lo que disminuye la dependencia de fuentes de energía no renovables.
2. **Reducción de desechos electrónicos**:  
   Al prolongar la vida útil de los nodos sensores, el proyecto ayuda a:
   * Disminuir la generación de residuos tecnológicos.
   * Reducir el ciclo de reemplazo de equipos electrónicos, lo que favorece la sostenibilidad a largo plazo.
3. **Aplicaciones ambientales directas**:  
   Este proyecto tiene un impacto directo en áreas clave como:
   * **Monitoreo ambiental**: Facilitando la vigilancia de ecosistemas y la predicción de desastres naturales.
   * **Agricultura de precisión**: Mejorando la toma de decisiones relacionadas con el uso eficiente de recursos naturales.

La factibilidad ambiental del proyecto es altamente positiva, ya que se encuentra alineado con los objetivos de sostenibilidad y tiene un impacto favorable en la reducción de residuos y el uso de energías renovables.

# Marco teórico

## Marco conceptual

### Redes de sensores inalámbricas (WSN en inglés)

Definición

Las Redes de Sensores Inalámbricos (WSN, por sus siglas en inglés) son sistemas compuestos por nodos sensores interconectados de manera inalámbrica, diseñados para monitorear y recopilar datos del entorno, como temperatura, humedad, presión, o niveles de luz (Novelo Sansores, 2020a) . Estos datos son procesados y transmitidos a través de la red hacia un nodo central o estación base, donde se pueden analizar y tomar decisiones en función de la información recolectada. Los nodos sensores están equipados con componentes como unidades de sensado, procesamiento, comunicación y alimentación, que les permiten operar de manera autónoma en entornos variados.

Importancia

Las WSN tienen un papel crucial en la evolución tecnológica y científica debido a su capacidad para operar en entornos remotos, hostiles o de difícil acceso. Su importancia radica en:

1. Automatización y monitoreo eficiente: Son esenciales para la recolección de datos en tiempo real en diversos campos como la agricultura, la salud, y la gestión ambiental.
2. Optimización de recursos: Permiten mejorar la eficiencia de procesos mediante el análisis de datos precisos y la toma de decisiones informadas.
3. Reducción de costos: Minimizan la intervención humana en tareas como el mantenimiento o monitoreo continuo.
4. Contribución al desarrollo sostenible: Su implementación facilita el uso eficiente de recursos energéticos y naturales.

Objetivos

1. Monitoreo y control en tiempo real: Proveer información confiable sobre las variables monitoreadas para la toma de decisiones.
2. Optimización de recursos energéticos: Implementar estrategias de bajo consumo para prolongar la vida útil de los nodos sensores.
3. Accesibilidad y autonomía: Operar en áreas remotas sin necesidad de infraestructura previa.
4. Adaptabilidad: Ser flexibles para diversas aplicaciones mediante configuraciones específicas y personalizadas.

Tipos

1. Basadas en topología (Philco Asqui, 2023):
   * Topología estrella: Cada nodo sensor se conecta directamente al nodo central. Es simple pero tiene baja tolerancia a fallos.
   * Topología de malla: Los nodos se conectan entre sí, permitiendo múltiples rutas para la transmisión de datos, lo que aumenta la tolerancia a fallos.
   * Topología jerárquica: Nodos organizados en niveles donde los de mayor jerarquía actúan como intermediarios para los de menor nivel.
2. Por aplicación:
   * Monitoreo ambiental: Para medir variables del medio ambiente.
   * Automatización industrial: Supervisar maquinaria o procesos.
   * Agricultura de precisión: Monitoreo de factores como humedad y calidad del suelo.
3. Por fuente de energía:
   * Alimentadas por baterías: Uso convencional, aunque con limitaciones de vida útil.
   * Con recolección de energía: Incorporan fuentes renovables como solar o eólica.

Alcance

El alcance de las redes de sensores inalámbricos incluye aplicaciones en múltiples sectores:

1. Agricultura: Optimización de riego, monitoreo de cultivos y prevención de plagas.
2. Salud: Monitoreo de pacientes, detección temprana de enfermedades y supervisión remota.
3. Gestión urbana: Desarrollo de ciudades inteligentes, incluyendo gestión de tráfico y calidad del aire.
4. Monitoreo ambiental: Detección de cambios en ecosistemas, predicción de desastres naturales y conservación de recursos naturales.
5. Defensa y seguridad: Vigilancia en áreas de alto riesgo y detección de intrusiones.

### Nodo sensor

#### **Definición**

Un nodo sensor es un dispositivo autónomo que forma parte de una red de sensores inalámbricos (WSN). Está diseñado para detectar y medir parámetros físicos o ambientales, como temperatura, humedad, presión, luz, vibraciones o movimiento (Philco Asqui, 2023). Además, tiene la capacidad de procesar los datos recolectados y transmitirlos de forma inalámbrica hacia otros nodos o hacia un nodo central, permitiendo la recopilación y análisis de información en tiempo real.

Los nodos sensores se componen generalmente de cuatro módulos principales:

1. **Unidad de sensado**: Incluye los sensores físicos encargados de medir las variables del entorno.
2. **Unidad de procesamiento**: Maneja los datos recogidos y ejecuta algoritmos para optimizar su uso.
3. **Unidad de comunicación**: Transmite los datos procesados hacia otros nodos o al nodo central utilizando protocolos inalámbricos.
4. **Unidad de alimentación**: Provee la energía necesaria para el funcionamiento del nodo, ya sea mediante baterías o fuentes de energía renovable.

#### **Importancia**

1. **Punto clave de las WSN**: Los nodos sensores son la base de las redes de sensores inalámbricos, ya que realizan la función primaria de sensado y transmisión de datos.
2. **Accesibilidad y monitoreo remoto**: Permiten medir variables en lugares de difícil acceso, lo que resulta esencial en aplicaciones como monitoreo ambiental y agricultura de precisión.
3. **Optimización de recursos**: Ayudan a reducir costos al automatizar procesos de recolección de datos.
4. **Escalabilidad**: Su diseño modular permite aumentar o reducir su número según las necesidades del proyecto.

#### **Objetivos**

1. **Monitorear el entorno**: Detectar y medir parámetros relevantes con precisión.
2. **Procesar y transmitir datos**: Asegurar que los datos recolectados sean enviados de manera eficiente hacia el nodo central.
3. **Asegurar bajo consumo energético**: Operar con mínima energía para prolongar su vida útil.

### **Nodo central**

#### **Definición**

El nodo central, también conocido como estación base, es el componente principal de una red de sensores inalámbricos (WSN) encargado de recopilar, procesar y almacenar los datos provenientes de los nodos sensores. Sirve como puente entre la red y el usuario final, transmitiendo la información a sistemas externos para su análisis o visualización (Rivera Lopez, 2020).

#### **Importancia**

1. **Coordinación de la red**: Actúa como el núcleo para administrar y sincronizar las actividades de los nodos sensores.
2. **Recolección de datos**: Centraliza la información obtenida en la red para su procesamiento y análisis.
3. **Conexión con sistemas externos**: Proporciona la interfaz necesaria para integrar la WSN con aplicaciones externas como bases de datos, sistemas en la nube o dashboards.
4. **Optimización del consumo energético**: Reduce el esfuerzo de procesamiento en nodos sensores al asumir tareas complejas.

#### **Objetivos**

1. **Concentrar datos**: Recopilar información de los nodos sensores para garantizar una visión integral del entorno monitoreado.
2. **Gestionar la red**: Supervisar el estado y funcionamiento de los nodos, incluyendo su conexión y consumo energético.
3. **Comunicar resultados**: Transmitir los datos procesados a sistemas externos para su análisis avanzado.

### **Gestión de Energía en Redes de Sensores Inalámbricos (WSN)**

#### **Definición**

La gestión de energía en redes de sensores inalámbricos (WSN) abarca las estrategias, técnicas y tecnologías implementadas para optimizar el uso de energía de los nodos sensores y otros componentes de la red. Su objetivo principal es prolongar la vida útil de los nodos y garantizar el funcionamiento continuo de la red, especialmente en entornos donde la reposición de energía es difícil o imposible.

#### **Importancia**

1. **Prolongación de la vida útil**: Reduce la frecuencia de mantenimiento y reposición de baterías, minimizando costos.
2. **Funcionamiento autónomo**: Permite que las redes operen de manera eficiente durante períodos prolongados, incluso en ubicaciones remotas.
3. **Sostenibilidad ambiental**: Reduce el impacto ambiental asociado al desecho de baterías y la sobrecarga energética.
4. **Fiabilidad de la red**: Minimiza interrupciones en la recolección y transmisión de datos.

#### **Objetivos**

1. **Optimizar el consumo energético**: Reducir el uso de energía mediante técnicas como modos de suspensión, compresión de datos y algoritmos eficientes.
2. **Implementar fuentes alternativas**: Incorporar tecnologías como recolección de energía solar, eólica o de vibraciones.
3. **Balancear el consumo**: Distribuir la carga energética entre los nodos para evitar puntos críticos.

#### **Tipos**

1. **Gestión pasiva**: Uso de hardware eficiente, como microcontroladores de bajo consumo o sensores optimizados.
2. **Gestión activa**: Implementación de estrategias de software, como protocolos de comunicación eficientes y optimización de rutas de transmisión (Romero Amondaray et al., 2002)
3. **Recolección de energía**: Uso de tecnologías como paneles solares, recolectores piezoeléctricos o de ondas de radio para recargar baterías.

### **Energía Renovable**

#### **Definición**

La energía renovable se refiere a aquellas fuentes de energía obtenidas de recursos naturales que son inagotables o se regeneran de forma continua, como la luz solar, el viento, el agua, la biomasa y el calor geotérmico. Estas fuentes representan una alternativa sostenible frente a los combustibles fósiles, ya que producen menos emisiones de gases de efecto invernadero y contribuyen a mitigar el cambio climático.

#### **~~Importancia~~**

1. **~~Sostenibilidad ambiental~~**~~: Reduce la dependencia de combustibles fósiles y disminuye las emisiones de carbono.~~
2. **~~Abundancia y disponibilidad~~**~~: Los recursos renovables están ampliamente distribuidos en todo el mundo.~~
3. **~~Independencia energética~~**~~: Reduce la dependencia de fuentes de energía importadas, fortaleciendo la autonomía energética.~~
4. **~~Reducción de costos a largo plazo~~**~~: Aunque la instalación inicial puede ser costosa, los costos de operación y mantenimiento suelen ser bajos.~~

#### **~~Objetivos~~**

1. **~~Mitigar el impacto ambiental~~**~~: Reducir la huella de carbono generada por actividades humanas.~~
2. **~~Fomentar la autosuficiencia energética~~**~~: Promover el uso de recursos locales para satisfacer necesidades energéticas.~~
3. **~~Impulsar la innovación tecnológica~~**~~: Incentivar el desarrollo de nuevas tecnologías y soluciones basadas en energías renovables.~~

#### **Ventajas**

1. **Impacto ambiental reducido**: Genera energía sin emisiones contaminantes significativas.
2. **Inagotabilidad**: Las fuentes renovables, como el sol y el viento, son virtualmente inagotables.
3. **Accesibilidad**: Puede implementarse en áreas remotas donde no llega la red eléctrica tradicional.
4. **Diversificación energética**: Reduce la dependencia de fuentes de energía no renovables.

### **Protocolos de Comunicación**

#### **Definición**

Un protocolo de comunicación es un conjunto de reglas y estándares que permiten la transferencia de datos entre dispositivos dentro de una red. Estos protocolos aseguran que los datos sean enviados, recibidos y procesados correctamente, definiendo aspectos como el formato, la sincronización y los métodos de transmisión. En redes de sensores inalámbricos (WSN), los protocolos juegan un papel crucial para garantizar la eficiencia energética, la robustez y la confiabilidad de la red (Romero Amondaray et al., 2002).

#### **Objetivos**

1. Estandarizar las reglas de comunicación para garantizar la compatibilidad entre dispositivos.
2. Optimizar los recursos disponibles, como el ancho de banda y la energía.
3. Proveer mecanismos de seguridad para proteger los datos transmitidos.
4. Garantizar la calidad del servicio (QoS) en aplicaciones críticas.

### **LoRa**

#### **Definición**

LoRa (Long Range) es una tecnología de modulación inalámbrica que utiliza espectro ensanchado por chirp para permitir la transmisión de datos a largas distancias con bajo consumo de energía. Es ampliamente utilizada en redes LPWAN (Low-Power Wide-Area Network) y es ideal para aplicaciones de IoT y sensores remotos según Guo & Healy.

#### **Importancia**

1. **Cobertura extensa**: Ofrece un rango de comunicación de hasta 15 km en condiciones óptimas.
2. **Eficiencia energética**: Diseñada para operar con dispositivos de bajo consumo, prolongando la vida útil de las baterías en redes de sensores.
3. **Flexibilidad**: Es compatible con múltiples dispositivos y aplicaciones en sectores como agricultura, monitoreo ambiental y ciudades inteligentes.

#### **Objetivos**

1. Proveer comunicación inalámbrica eficiente en términos de alcance y consumo energético.
2. Facilitar la conectividad en áreas remotas y entornos difíciles para redes convencionales.
3. Promover la integración de dispositivos IoT en redes de baja potencia.

### **XBee**

#### **Definición**

Nos dice Novelo Sansores (2020b) que XBee es una familia de módulos de radiofrecuencia que soportan estándares como Zigbee, 802.15.4, DigiMesh y Wi-Fi. Son ampliamente utilizados en redes de sensores y sistemas embebidos para la transmisión de datos a cortas y medias distancias.

#### **Importancia**

1. **Versatilidad**: Compatible con múltiples protocolos, adaptándose a distintas aplicaciones.
2. **Facilidad de uso**: Configuración y programación simplificadas, ideales para prototipos y proyectos educativos.
3. **Interoperabilidad**: Permite integrar redes heterogéneas con diversos dispositivos.

#### **Objetivos**

1. Proveer una solución de comunicación inalámbrica confiable y de bajo costo.
2. Facilitar la creación de redes malladas o punto a punto para aplicaciones de IoT.

### **MQTT (Message Queuing Telemetry Transport)**

#### **Definición**

MQTT es un protocolo de comunicación ligero basado en el modelo publicador-suscriptor (Tapias Baena (2021)). Fue diseñado originalmente por IBM en 1999 para transmisiones de datos de sensores en redes con ancho de banda limitado o inestables. Utiliza el protocolo TCP/IP para garantizar la entrega confiable de mensajes, lo que lo hace ideal para aplicaciones de Internet de las Cosas (IoT), donde el ahorro energético y la eficiencia de transmisión son críticos.

#### **Importancia**

1. **Eficiencia de recursos**: El bajo peso del protocolo lo hace adecuado para dispositivos con limitaciones de memoria, energía y procesamiento.
2. **Escalabilidad**: Su modelo publicador-suscriptor permite gestionar grandes volúmenes de dispositivos y mensajes sin sobrecargar la red.
3. **Confiabilidad**: Permite configurar niveles de calidad de servicio (QoS) para asegurar la entrega de mensajes en condiciones adversas.

#### **Características Clave**

1. **Modelo Publicador-Suscriptor**:
   * Los clientes publican mensajes en "tópicos" (canales de comunicación).
   * Otros clientes se suscriben a estos tópicos para recibir los mensajes publicados.
   * Desacopla al remitente del receptor, mejorando la flexibilidad.
2. **Persistencia**:
   * Los mensajes "retained" permiten que nuevos suscriptores reciban el último mensaje publicado en un tópico.
3. **Bajo Consumo Energético**:
   * Diseñado para reducir el intercambio de datos y prolongar la duración de baterías en dispositivos IoT.

#### **Ventajas**

1. **Ligereza**: Consume pocos recursos en comparación con protocolos como HTTP.
2. **Versatilidad**: Funciona en redes con conectividad intermitente o de baja calidad.
3. **Confiabilidad**: La capacidad de ajustar el QoS garantiza la entrega adecuada de datos en aplicaciones críticas.

### Protocolo de Control de Acceso al Medio (MAC)

Definición

El Protocolo de Control de Acceso al Medio (MAC, por sus siglas en inglés: Medium Access Control) es un conjunto de reglas que regulan cómo los dispositivos en una red comparten un medio de transmisión. Su principal función es gestionar la comunicación entre los nodos sensores, evitando colisiones y optimizando el uso del canal inalámbrico. En redes de sensores inalámbricas (WSN), los protocolos MAC están diseñados para minimizar el consumo energético mientras garantizan una transmisión eficiente de datos

Importancia

El protocolo MAC es esencial en las WSN, ya que define cuándo y cómo los nodos pueden transmitir datos, evitando interferencias y asegurando un uso eficiente del canal de comunicación. Dado que la transmisión de datos es una de las principales fuentes de consumo energético en los sensores, el diseño de un MAC eficiente puede prolongar la autonomía de la red, reducir la latencia y mejorar la fiabilidad en la entrega de paquetes.

Clasificación

Los protocolos MAC en redes de sensores se dividen en dos grandes categorías:

* Protocolos MAC síncronos: Como TDMA (Time Division Multiple Access), donde el tiempo se divide en ranuras asignadas a cada nodo para evitar colisiones.
* Protocolos MAC asíncronos: Como CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance), donde los nodos detectan si el canal está libre antes de transmitir.

## Estado del Arte

### Protocolos Mac de bajo consumo

En redes de sensores inalámbricas (WSN), el consumo energético se ve altamente influenciado por el protocolo de control de acceso al medio (MAC). Investigaciones previas han desarrollado protocolos como Sensor-MAC (S-MAC) y Timeout-MAC (T-MAC), que implementan ciclos de sueño para reducir el consumo de energía en los nodos sensores. Wei Ye et al. (2002) propusieron S-MAC, el cual sincroniza los periodos de actividad y reposo entre los nodos para minimizar el tiempo de escucha innecesario. Por otro lado, van Dam & Langendoen (2003) introdujeron T-MAC, que ajusta dinámicamente los tiempos de transmisión y suspensión con base en la carga de tráfico en la red. Además, se han desarrollado protocolos híbridos que combinan CSMA/CA y TDMA para mejorar la eficiencia energética. Al-Janabi & Al-Raweshidy (2019) propusieron un protocolo híbrido que utiliza una tabla de programación dinámica para la activación y suspensión de los nodos, logrando una reducción significativa en el consumo de energía. Y, según Sadeq et al. (2022), a pesar de que los protocolos MAC juegan un papel crucial en la eficiencia energética y el rendimiento de las WSN, a medida que la escala de la red aumenta, se presentan desafíos significativos en términos de latencia, escalabilidad y eficiencia energética que deben ser abordados a futuro

### Estrategias de recolección de energía en WSN

El consumo energético es uno de los principales desafíos en el diseño y operación de redes de sensores inalámbricas (WSN), especialmente cuando los nodos sensores se implementan en entornos donde el mantenimiento y la sustitución de baterías no es viable. Para abordar esta limitación, se han desarrollado diversas estrategias de recolección de energía que permiten extender la vida útil de los nodos, disminuyendo la dependencia de fuentes de alimentación tradicionales​

A. Fuentes de Recolección de Energía en WSN

Diferentes estudios han explorado diversas fuentes de energía renovable aplicadas a redes de sensores, destacando las siguientes:

1. Energía Solar

La recolección de energía solar es una de las técnicas más utilizadas debido a la alta disponibilidad y eficiencia de conversión de la radiación solar en electricidad. Según Lajara José (2014) , la implementación de celdas solares con almacenamiento híbrido en baterías y supercondensadores puede extender significativamente la autonomía de los nodos sensores, permitiendo que operen exclusivamente con energía solar en condiciones climáticas favorables

* Ventajas: Alta eficiencia en condiciones de buena iluminación, tecnología madura y relativamente económica.
* Desafíos: Variabilidad en la captación de energía debido a factores climáticos y cambios estacionales.​

2. Energía Piezoeléctrica

Otra alternativa es el uso de dispositivos piezoeléctricos para capturar energía mecánica del ambiente, como vibraciones y movimientos estructurales. Estudios recientes han demostrado que estos sistemas pueden generar energía suficiente para alimentar sensores de bajo consumo, aunque presentan limitaciones en la cantidad de energía capturada​Redes de sensores.

* Ventajas: Funcionamiento continuo en entornos con movimiento constante.
* Desafíos: Baja producción energética y necesidad de integración con otras fuentes.

3. Recolección de Energía de Radiofrecuencia (RF Energy Harvesting)

El aprovechamiento de ondas de radio como fuente de energía ha sido un área de creciente interés. Se han desarrollado circuitos capaces de capturar señales de radiofrecuencia y convertirlas en electricidad para alimentar nodos sensores. Zungeru et al. (2012) explican que la captación de energía de RF permite la recarga constante de los sensores, siempre que haya una fuente emisora lo suficientemente fuerte en las cercanías, la eficiencia de este tipo de sistemas depende en gran medida de la distancia entre el sensor y la fuente de emisión, así como de la densidad de señales en el entorno.

* Ventajas: Disponibilidad continua en entornos urbanos con múltiples fuentes de emisión.
* Desafíos: Baja eficiencia en la conversión de energía y dependencia de fuentes externas de transmisión de radio.

4. Termoeléctrica

El uso de diferencias de temperatura entre dos superficies para generar electricidad mediante el efecto Seebeck ha sido estudiado como una solución viable para WSN en entornos con variaciones térmicas, como en industrias o regiones geotérmicas. Wei Chen et al. (2013) desarrollaron un modelo experimental en el que pequeños generadores termoeléctricos (TEG) fueron capaces de alimentar sensores en tuberías industriales, aprovechando el calor residual.

* Ventajas: Genera energía pasiva en entornos con fluctuaciones térmicas, como instalaciones industriales y equipos de alta temperatura.
* Desafíos: Bajo nivel de generación eléctrica en condiciones de gradientes térmicos bajos, lo que requiere sistemas eficientes de almacenamiento.

5.Eólica

La energía eólica es otra fuente de energía alternativa común en nuestra vida diaria, sin embargo, no lo es para las WSN, esto se debe a varios factores como la infraestructura más compleja que manejan los aerogeneradores y las limitaciones en los espacios en los que las WSN se ven desplegadas, a pesar de ello, hay investigaciones recientes que exploran el uso de microturbinas eólicas para alimentar nodos sensores. Por ejemplo, el estudio de Yu et al. presenta un generador eléctrico impulsado por el viento basado en una unión Schottky dinámica, capaz de producir una corriente continua promedio de 4.4 mA, con picos de hasta 8.4 mA, durante 360 segundos. Este avance sugiere la viabilidad de utilizar el viento como fuente de energía para sensores distribuidos en entornos con flujo de aire constante

* Ventajas: Generación de energía en entornos con flujo de aire constante, posibilidad de complementar otras fuentes de energía en sistemas híbridos.
* Desafíos: Variabilidad en la velocidad del viento, necesidad de diseños que minimicen el desgaste mecánico y maximicen la eficiencia en condiciones de viento variable.

B. Estrategias de almacenamiento de Energía

Además de la captación de energía, es fundamental desarrollar estrategias de almacenamiento y gestión energética para optimizar el uso de los recursos recolectados. Entre las soluciones más estudiadas se encuentran:

1. Uso de supercondensadores

Los supercondensadores se han investigado como una alternativa eficiente a las baterías convencionales debido a su alta capacidad de carga y descarga rápida. En estudios recientes, se ha demostrado que el uso de supercondensadores en combinación con baterías recargables permite reducir la degradación de estas últimas, mejorando la autonomía de los nodos. Lajara José propone un sistema híbrido donde los supercondensadores priorizan la alimentación del nodo y solo en ausencia de carga solar se recurre a la batería.

2. Algoritmos de gestión de energía

Para mejorar la eficiencia del sistema, se han desarrollado algoritmos adaptativos que regulan el consumo energético en función de la energía recolectada. Calvo-Fullana et al. (2016) propusieron un enfoque basado en asignación dinámica de potencia, donde los sensores adaptan su consumo energético dependiendo de la cantidad de energía almacenada y disponible. Estos modelos mejoran la sostenibilidad reduciendo los períodos de inactividad debido a la falta de energía, esto nos da una maximización del aprovechamiento del uso de la energía recolectada, sin embargo, requiere procesamiento adicional lo que puede llegar a afectar nuestras mediciones

### Algoritmos de ahorro de energía en WSN

La eficiencia energética en las WSN no solo es un factor crítico para la vida útil de la red, sino que también influye en la precisión y continuidad de la recolección de datos, es por esto que para abordar este problema, investigadores han desarrollado diversos algoritmos de ahorro energético que optimizan el consumo en diferentes niveles, desde la administración del tiempo de actividad de los nodos hasta la optimización en la transmisión y procesamiento de datos; en la siguiente sección, se describirán los principales tipos de algoritmos de ahorro de energía, desde los más sencillos hasta los más avanzados, detallando su implementación, resultados y futuras mejoras propuestas por los autores

Algoritmos de Duty Cycling

Los algoritmos de Duty Cycling regulan el tiempo que un nodo permanece activo o en modo de bajo consumo para minimizar el gasto energético. Se basan en la premisa de que los sensores no necesitan estar activos todo el tiempo, sino solo en momentos clave​

Sedighimanesh et al. (2016) propusieron un esquema en el cual los nodos reducen su tiempo de actividad al mínimo y solo despiertan en intervalos específicos para transmitir datos, bajo este esquema corrieron simulaciones distintas bajo diferentes parámetros y condiciones, en estos estudios de simulación, este enfoque redujo el consumo de energía hasta un 40%, incrementando la autonomía de la red sin afectar la calidad de los datos recolectados​, los investigadores han planteado que una mejora futura en Duty Cycling sería la incorporación de algoritmos adaptativos, que ajusten dinámicamente el tiempo de actividad en función de la carga de tráfico y la cantidad de energía disponible en el nodo. Esto permitiría una optimización más eficiente de los recursos y evitaría la activación innecesaria de los nodos, prolongando aún más su vida útil​

Algoritmos de Balanceo de Carga Energética

Los algoritmos de balanceo de carga energética tienen como objetivo distribuir el consumo de energía de manera equitativa entre los nodos sensores, evitando que algunos se descarguen más rápido que otros y garantizando así una mayor duración de la red.

Samara et al. (2022) desarrollaron un algoritmo de balanceo energético basado en la identificación de nodos óptimos para actuar como líderes de clúster, seleccionándolos en función de su ubicación y su nivel de energía disponible. En sus experimentos de simulación, demostraron que este enfoque prolonga la vida de la red en un 31% en comparación con LEACH, uno de los protocolos estándar más utilizados en WSN​. El principal desafío de estos algoritmos radica en la dinámica de las redes de sensores, donde algunos nodos pueden consumir energía más rápido dependiendo de su función en la red. Por ello, los investigadores han sugerido la integración de estos algoritmos con sistemas de recolección de energía, permitiendo que los nodos que capturan más energía del entorno asuman una mayor carga de trabajo, mejorando así la eficiencia global de la red​

Algoritmos de Enrutamiento Energéticamente Eficientes

El enrutamiento de datos en WSN es una de las actividades que más consumo energético genera, ya que cada transmisión implica un gasto considerable de energía en los nodos intermedios. Para mitigar este problema, se han desarrollado algoritmos de enrutamiento optimizados, que buscan minimizar el número de transmisiones necesarias para enviar los datos al destino final

Barker & Swany (2020) propusieron un algoritmo de enrutamiento basado en reducción de datos, en el que los nodos procesan y filtran la información antes de enviarla, evitando la transmisión de datos redundantes o innecesarios. Sus experimentos demostraron que este enfoque reduce el uso de ancho de banda, disminuye el consumo energético y mejora la velocidad de transmisión en redes con tráfico elevado, los investigadores han identificado oportunidades de mejora en estos algoritmos mediante la integración de inteligencia artificial y aprendizaje automático, lo que permitiría ajustar dinámicamente las rutas de transmisión y reducir aún más el desperdicio energético. Se plantea que en el futuro estos algoritmos puedan incorporar sistemas de predicción de tráfico de datos, de manera que los nodos puedan seleccionar rutas óptimas en función del estado actual y futuro de la red

Algoritmos de Compresión de Datos para Ahorro de Energía

Una estrategia alternativa para reducir el consumo energético en WSN es la compresión de datos, la cual busca disminuir el volumen de información transmitida, reduciendo así el uso de energía en cada transmisión ya que la relación entre la energía gastada para enviar y el tamaño de los datos enviados es proporcional

Wei Chen et al. (2013) desarrollaron un esquema de compresión distribuida, en el cual los nodos sensores identifican y eliminan información redundante antes de enviarla. En sus experimentos, observaron que este enfoque redujo el consumo energético en un 50%, al disminuir significativamente la cantidad de paquetes de datos transmitidos, a pesar de los beneficios de la compresión de datos, existen desafíos en su implementación, especialmente en sensores con recursos de procesamiento limitados. Por esta razón, se han propuesto modelos que combinan compresión con técnicas de procesamiento en el borde (Edge Computing), permitiendo que la reducción de datos se realice en nodos con mayor capacidad de cómputo antes de enviarlos al servidor central, sin embargo, esta propuesta propondría nuevos retos al necesitar definir las rutas de envío y decidir cuales nodos tienen menor carga por lo cual abre espacio a nuevas investigaciones y pruebas en laboratorio

Los algoritmos de ahorro de energía en WSN son una herramienta fundamental para extender la vida útil de las redes y mejorar su eficiencia. Desde estrategias simples como Duty Cycling, hasta enfoques más avanzados como balanceo de carga, enrutamiento optimizado y compresión de datos, cada técnica aporta soluciones a distintos niveles del consumo energético​, en el futuro, se espera que estos algoritmos se combinen con fuentes de recolección de energía del entorno, logrando redes de sensores inalámbricas verdaderamente sostenibles y con autonomía prácticamente ilimitada

### Modelado del Consumo Energético en Redes de Sensores Inalámbricas (WSN)

El modelado del consumo energético es una técnica utilizada para describir, predecir y analizar el gasto de energía en un sistema determinado. En el contexto de las redes de sensores inalámbricas (WSN), se emplea para estimar cuánta energía consumen los nodos sensores bajo diferentes condiciones de operación.

Dado que estos sistemas funcionan en entornos con restricciones energéticas, como sensores desplegados en ubicaciones remotas o de difícil acceso, entender cómo se distribuye el consumo de energía es fundamental para optimizar su desempeño y prolongar su autonomía.

El modelado del consumo energético en WSN permite:

* Cuantificar el gasto energético en función de los procesos que realiza un nodo sensor (transmisión de datos, procesamiento, estado de reposo, etc.).
* Predecir la vida útil de los nodos y la duración de la red en distintas condiciones de uso.
* Optimizar el diseño de hardware y software de los nodos para reducir el desperdicio energético.
* Evaluar la efectividad de algoritmos de ahorro de energía antes de su implementación.

Este tipo de modelado se puede realizar a distintos niveles, desde el componente individual (por ejemplo, el consumo de un transceptor inalámbrico) hasta la red completa, analizando el impacto de diferentes protocolos y estrategias de gestión de energía.

A continuación, se presentan los principales enfoques utilizados para modelar el consumo energético en WSN

#### A. Modelado a Nivel de Transceptor

Este enfoque se centra en el consumo de energía asociado a la comunicación inalámbrica, uno de los aspectos más críticos en WSN. Se analiza el impacto de la transmisión y recepción de datos, considerando parámetros como la potencia de transmisión, el tiempo de envío y la modulación utilizada.

Ejemplo de Implementación

Lajara José (2014) estudió modelos teóricos del consumo de energía en redes de sensores enfocándose en el transceptor, considerando factores como el tipo de modulación y la distancia entre nodos. En este modelo, se estima la energía consumida en función de la ecuación de propagación de la señal, lo que permite optimizar la configuración del transceptor para minimizar el gasto energético.

Resultados y Futuras Mejoras

Este tipo de modelado ha permitido desarrollar protocolos MAC de bajo consumo, que ajustan dinámicamente la potencia de transmisión según la distancia y la calidad del canal. Investigaciones futuras buscan integrar aprendizaje automático para optimizar el uso del transceptor en función de las condiciones de la red.

#### B. Modelado a Nivel de Nodo Completo

En este enfoque, el modelado se extiende a todo el nodo sensor, considerando no solo el transceptor, sino también el microcontrolador, los sensores y los sistemas de almacenamiento de energía. Se busca evaluar el impacto del procesamiento de datos y el tiempo de actividad del nodo.

Ejemplo de Implementación

Rojas & Milton (2020) diseñaron un modelo de simulación basado en OPNET, donde analizaron el consumo energético de cada nodo incluyendo CPU, sensores y transceptor, lo que permitió obtener un panorama más completo del gasto de energía en distintas aplicaciones​Redes de sensores.

El estudio demostró que el procesamiento local de datos antes de transmitirlos puede reducir hasta un 30% el consumo energético, optimizando el uso del hardware del nodo.

Resultados y Futuras Mejoras

Gracias a estos modelos, se han desarrollado estrategias de duty cycling que desactivan partes del nodo cuando no están en uso, reduciendo significativamente el consumo.

Se están explorando técnicas de computación en el borde (Edge Computing) para descargar tareas pesadas a nodos con mayor capacidad de procesamiento.

#### C. Modelado a Nivel de Red

El modelado a nivel de red se enfoca en evaluar el impacto de la topología, los protocolos de comunicación y las estrategias de enrutamiento en el consumo energético global. Este enfoque es clave para diseñar redes optimizadas en términos de eficiencia energética y cobertura.

Ejemplo de Implementación

Rojas & Milton (2020) también realizaron una evaluación del desempeño energético a nivel de red, midiendo latencia, tráfico de datos y eficiencia del enrutamiento​

Se utilizaron simulaciones para probar diferentes protocolos de enrutamiento y su impacto en el consumo de la red en entornos urbanos e industriales.

Resultados y Futuras Mejoras

Este tipo de modelado ha permitido desarrollar protocolos de enrutamiento adaptativos, que seleccionan la mejor ruta de transmisión en función del estado energético de los nodos.

Se están integrando redes neuronales y optimización multiobjetivo para encontrar el equilibrio entre eficiencia energética y calidad de servicio.

### Metodologías de Modelado Energético en Redes de Sensores Inalámbricas

El modelado del consumo energético en redes de sensores inalámbricas (WSN) es un campo de investigación clave para optimizar la eficiencia de estos sistemas. Existen diversas metodologías que permiten estimar, predecir y mejorar el rendimiento energético de los nodos sensores, considerando factores como la topología de la red, los protocolos de comunicación y el hardware utilizado. A continuación, se presentan las principales metodologías empleadas en la literatura para modelar el consumo energético en WSN.

#### A. Modelado Masado en Modelos Analíticos

Los modelos analíticos utilizan ecuaciones matemáticas para describir el consumo energético de un nodo sensor en función de diferentes parámetros, como la potencia de transmisión, el tiempo de actividad y la cantidad de datos enviados.

Ejemplo de Implementación

* Lajara José (2014) propuso un modelo analítico que estima el consumo energético en función de la distancia entre nodos y la potencia de transmisión. Sus ecuaciones permitieron optimizar la selección de rutas en WSN para minimizar el gasto energético.
* Rojas & Milton (2020) desarrollaron un modelo que considera el impacto del protocolo MAC en el consumo energético, mostrando que la eficiencia puede aumentar hasta un 25% si se ajustan los tiempos de activación y reposo de los nodos.

Estos modelos han permitido diseñar protocolos energéticamente eficientes, como S-MAC y T-MAC.

#### B. Modelado Basado en Simulación

Los modelos de simulación permiten replicar el comportamiento de una red de sensores en un entorno virtual, evaluando el consumo energético bajo diferentes condiciones operativas. Se utilizan herramientas como NS-3, OMNeT++, MATLAB y Cooja para simular escenarios de comunicación y analizar el impacto de diferentes configuraciones de red.

Ejemplo de Implementación

* Rojas & Milton (2020) implementaron un modelo de simulación en OPNET, donde evaluaron el consumo energético de los nodos bajo distintos protocolos de comunicación y configuraciones de hardware. Su estudio mostró que el uso de enrutamiento adaptativo puede reducir el consumo hasta en un 30%.

#### C. Modelado Basado en Datos Empíricos

Los modelos empíricos se construyen a partir de mediciones reales de consumo energético en nodos sensores. Este enfoque proporciona resultados altamente precisos, aunque requiere experimentación en entornos reales

Ejemplo de Implementación

* Novelo Sansores (2020b) midió el consumo energético de sensores en un invernadero inteligente, identificando que el uso de protocolos eficientes de transmisión puede extender la vida útil de la batería en un 40%.
* Barker & Swany (2020) realizaron mediciones en nodos sensores con procesadores ARM, mostrando que la delegación de tareas de cómputo a nodos con mayor capacidad puede reducir el consumo energético global.

Las simulaciones basadas en este modelado nos han permitido validar las predicciones de los modelos analíticos y de simulación en entornos reales

Como hemos podido ver, las metodologías de modelado energético en WSN han evolucionado desde enfoques puramente analíticos hasta modelos híbridos que combinan simulaciones y mediciones experimentales. Cada metodología tiene sus ventajas y limitaciones, y su selección depende del nivel de precisión requerido y la disponibilidad de recursos computacionales

# Metodología

## 1. Enfoque de la Investigación

Este proyecto utiliza una metodología **mixta**, ya que combina enfoques **cuantitativos y cualitativos** para abordar el problema de optimización energética en redes de sensores inalámbricos (WSN).

### Enfoque Cuantitativo:

* Se mide el impacto energético mediante parámetros objetivos, como el consumo de energía eléctrica, la entrega de paquetes y la potencia de señal.
* Estas métricas son fundamentales para evaluar la eficiencia y efectividad de los algoritmos y protocolos implementados.

### Enfoque Cualitativo:

* Se analizan las implicaciones prácticas y contextuales de los resultados, como la viabilidad de las soluciones en escenarios reales y su potencial aplicación en el ámbito industrial y ambiental.
* Esto permite interpretar los datos cuantitativos en un marco más amplio, considerando las necesidades y limitaciones de los entornos donde se implementarán las WSN.

Este proyecto es **experimental**, ya que pone a prueba soluciones diseñadas específicamente para reducir el consumo energético en redes de sensores inalámbricos. Esto incluye:

### Pruebas de Programación y Comunicación:

* Se crearán y probarán algoritmos y configuraciones dentro de los protocolos de comunicación para evaluar su impacto en el ahorro de energía.

### Pruebas de Hardware:

* Se implementará una fuente de energía que use renovables como el sol y se probará su funcionamiento en diferentes condiciones además de evaluar su funcionalidad y fiabilidad en un entorno real

Al ser experimental, los resultados obtenidos se basan en pruebas reales hechas en un entorno controlado, lo que asegura que las conclusiones sean confiables.

## 2. Selección de la Muestra

Se implementarán **tres redes de nodos, separadas por tecnología de comunicación: LoRa, Xbee ZigBee. Además, se evaluará la eficiencia energética aplicando distintas estrategias dentro de cada tecnología en:**

* **Topología de estrella**
* **Topología de red mallada**

Cada configuración será sometida a mediciones energéticas y de desempeño para determinar la mejor combinación en términos de consumo y fiabilidad. No se modificará la capa MAC de las tecnologías evaluadas, sino que se analizarán las opciones de ahorro de energía dentro de sus respectivos protocolos.

## 3. Instrumentos y Técnicas de Recolección de Datos

Para la recolección de datos se utilizarán:

* **Mediciones de consumo energético**: Se emplearán multímetros digitales y sensores de consumo para registrar la corriente y voltaje de los nodos sensores.
* **Protocolos de transmisión de datos**: Se registrará el consumo asociado a diferentes tecnologías de comunicación (LoRa, LoRaWAN y ZigBee), además de analizar el impacto en cada tipo de topología y capa de acceso a datos.
* **Análisis de algoritmos de gestión energética**: Se comparará el impacto de distintos algoritmos de optimización (modos sleep, reducción de frecuencia de transmisión, compresión de datos).
* **Registro temporal**: Se medirá la autonomía de los nodos en diferentes escenarios de consumo.

## 4. Procedimiento o Desarrollo de la Investigación

El estudio se desarrollará en las siguientes fases:

* Fase 0: Revisión de Documentación y Preparación
  + Investigación sobre redes de sensores inalámbricos (WSN) y optimización energética.
  + Revisión de literatura científica y trabajos previos sobre tecnologías LoRa, ZigBee y estrategias de ahorro energético.
  + Selección y adquisición de herramientas y dispositivos necesarios para la implementación del proyecto.
  + Configuración de entornos de desarrollo y pruebas.
* Fase 1: Evaluación del Consumo Energético Actual
  + Implementación de nodos sensores con baterías convencionales y fuentes renovables en una red convencional.
  + Pruebas de recolección para fuentes renovables.
  + Registro de consumo energético en diferentes condiciones operativas.
  + Comparación de autonomía de los nodos.
* Fase 2: Desarrollo e Implementación de un Algoritmo de Optimización
  + Diseño de un algoritmo para la gestión eficiente de energía.
  + Aplicación de técnicas como modos sleep, reducción de frecuencia de transmisión y optimización del protocolo de comunicación.
  + Medición del impacto del algoritmo en la autonomía de los nodos a través de comparaciones de consumo energético antes y después de su implementación.
* Fase 3: Comparación de Estrategias de Comunicación
  + Pruebas con diferentes protocolos de transmisión en entornos variados.
  + Análisis comparativo de consumo entre LoRa, LoRaWAN y ZigBee considerando las distintas topologías y considerando sus respectivas opciones de optimización energética.
* Fase 4: Análisis y Conclusiones
  + Procesamiento y análisis de datos mediante Python (con pandas y NumPy).
  + Representación de resultados mediante tablas y gráficos estadísticos.
  + Aplicación de pruebas estadísticas como diferencia de medias y análisis de varianza (ANOVA) para evaluar mejoras en consumo.
  + Normalización de datos para garantizar comparaciones precisas entre diferentes estrategias.

## 5. Análisis de Datos

Los datos recolectados se analizarán mediante:

* **Análisis estadístico con Python**: Se utilizarán librerías como pandas, NumPy y Matplotlib para procesar y visualizar los datos.
* **Comparaciones de consumo**: Aplicación de pruebas estadísticas como **ANOVA** para medir diferencias en consumo entre diferentes configuraciones.
* **Visualización de resultados**: Generación de **gráficos comparativos** de consumo y autonomía de los nodos en distintos escenarios.

## 6. Validez y Confiabilidad

Para garantizar la validez de la investigación:

* Se realizarán pruebas controladas en entornos replicables.
* Se calibrarán los equipos de medición antes de cada prueba.
* Se ejecutarán experimentos repetidos para asegurar la consistencia de los resultados.
* Se realizarán mediciones en entornos donde tengamos permiso.

## 7. Consideraciones Éticas

* Se documentará el procedimiento de manera clara y transparente.
* Se garantizará el uso responsable de la energía en los experimentos, evitando el desperdicio innecesario.
* El despliegue de las redes de sensores se realizará en áreas públicas de uso libre y, en caso de ser necesario, en zonas privadas con permiso de los responsables asegurando el cumplimiento de normativas.

Con esta metodología, se espera proporcionar una solución eficiente para la gestión de energía en redes de sensores inalámbricas, optimizando su autonomía y funcionalidad en entornos reales.

# Descripción de las actividades realizadas

## Adquisición de conocimiento y Desarrollo de prototipos básicos

Antes de proceder con el diseño y la implementación de la red de sensores mejorada, se determinó que era fundamental llevar a cabo una fase exhaustiva de investigación y experimentación. Esta etapa preliminar tuvo como objetivo principal no solo la comprensión teórica de las tecnologías implicadas, sino también la adquisición de experiencia práctica en su manipulación y configuración. Para ello, el trabajo se estructuró en torno al desarrollo de una serie de prototipos funcionales, cada uno enfocado en un aspecto tecnológico clave del proyecto.

Prototipo I: Control y Modulación de Señales

El punto de partida fue la familiarización con la plataforma de desarrollo Arduino. El primer prototipo consistió en un circuito básico para el control de actuadores mediante la modulación por ancho de pulso (PWM). Se implementó un sistema que permitía variar la intensidad lumínica de un diodo LED de alta potencia y la velocidad de un pequeño motor de corriente continua. Este ejercicio práctico fue crucial para comprender cómo los microcontroladores pueden generar señales analógicas simuladas a partir de salidas digitales, un concepto esencial para la gestión de diversos tipos de sensores y actuadores en el sistema final. Los conocimientos adquiridos sentaron las bases para el manejo preciso de componentes que no operan con señales binarias de encendido/apagado.

Prototipo II: Comunicación Serial y Transmisión de Datos GPS

Una vez dominados los conceptos básicos, se procedió a explorar la comunicación entre microcontroladores. Para este fin, se diseñó un prototipo compuesto por dos placas Arduino. La primera, actuando como nodo emisor, fue equipada con un módulo GPS. Su función era adquirir las coordenadas geográficas (latitud y longitud) del sistema satelital y transmitirlas en tiempo real. La segunda placa, el nodo receptor, recibía esta información a través de una comunicación serial (UART) y la mostraba en un monitor serial.

Este desarrollo permitió profundizar en los protocolos de comunicación punto a punto, comprendiendo la importancia de la configuración de parámetros como la velocidad en baudios (baud rate) y la estructura de las tramas de datos para asegurar una transmisión fiable. Se enfrentó un desafío inicial relacionado con el parsing (análisis sintáctico) de las sentencias NMEA que entrega el módulo GPS, lo que obligó a desarrollar un pequeño algoritmo para aislar y convertir los datos de coordenadas a un formato numérico utilizable. Esta experiencia fue fundamental para el posterior manejo de flujos de datos provenientes de múltiples sensores.

Prototipo III: Almacenamiento Local y Gestión de Ficheros

Con el objetivo de asegurar la integridad de los datos ante posibles fallos de comunicación, se investigó el uso de sistemas de almacenamiento local. Se construyó un prototipo que integraba un Arduino con un adaptador para tarjetas micro SD. El objetivo fue implementar un sistema de registro de datos (datalogger) básico. Se desarrollaron funciones específicas para crear, escribir y leer archivos de texto (.txt) en la tarjeta de memoria.

Durante esta fase, se presentó un problema recurrente: la corrupción de archivos al interrumpir la alimentación del circuito mientras se realizaba una operación de escritura. Este inconveniente nos enseñó la importancia de implementar mecanismos de seguridad en el código, como el cierre explícito de los archivos después de cada escritura y la verificación de la correcta inicialización de la tarjeta SD al arrancar el sistema. Se logró así un entendimiento práctico sobre la gestión de sistemas de ficheros y la persistencia de datos, un requisito indispensable para el respaldo de información en los nodos de la red.

3.1.4 Prototipo IV: Intercomunicación entre Plataformas Heterogéneas (Arduino y ESP32)

Finalmente, para evaluar la interoperabilidad entre diferentes tecnologías, se abordó la comunicación entre una placa Arduino y un microcontrolador ESP32. Este último fue considerado por sus capacidades de conectividad Wi-Fi y Bluetooth. Se estableció una comunicación bidireccional utilizando el protocolo I2C, donde el ESP32 actuaba como maestro y el Arduino como esclavo.

El principal reto técnico en este prototipo fue la diferencia en los niveles lógicos de voltaje entre ambas placas (5V para el Arduino Uno y 3.3V para el ESP32). Una conexión directa podría haber dañado el ESP32, por lo que fue necesario investigar e implementar un convertidor de nivel lógico bidireccional para adaptar las señales de los buses I2C. La resolución de este problema proporcionó un aprendizaje invaluable sobre la importancia de la compatibilidad de hardware en sistemas embebidos y las soluciones de interfaz necesarias para integrar componentes con distintas especificaciones técnicas. Esta fase consolidó los conocimientos sobre protocolos de comunicación más complejos y preparó al equipo para la integración final de todos los subsistemas del proyecto.

## Generación de un prototipo base para mejorar

Para el desarrollo de este proyecto, se decidió no empezar completamente desde cero. En lugar de eso, tomamos como punto de partida un sistema de monitoreo y control de energía ya existente y funcional, el cual fue diseñado e implementado por Zamora (2017). La idea fue utilizar su trabajo, que ya había sido probado en un laboratorio, como la base sobre la cual nosotros construiríamos nuestras mejoras, un pequeño diseño de esto se puede ver en la Figura 1

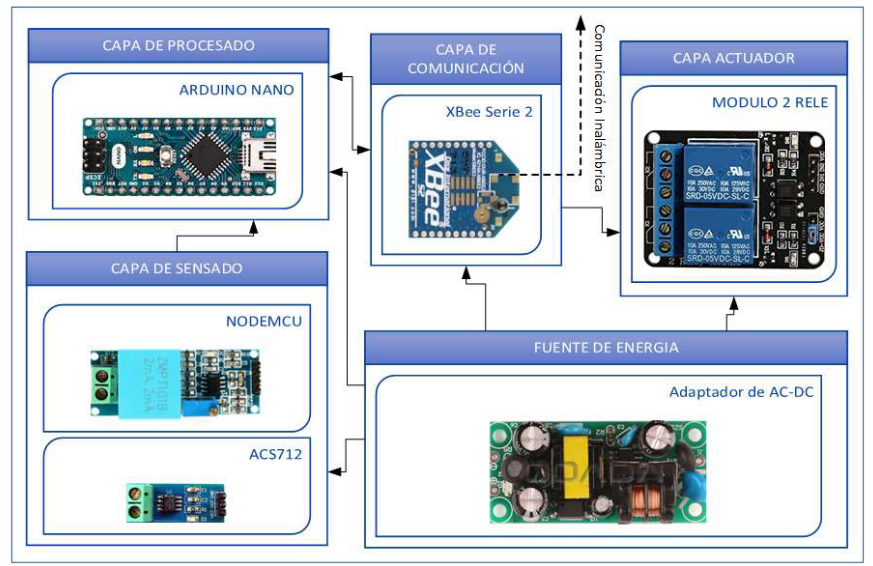


Figura 1. Diseño del nodo sensor. Reimpreso de Zamora(2017, p. 89)

El sistema original en el que nos basamos es una solución bastante completa para medir y controlar el consumo eléctrico. Su funcionamiento se puede resumir así:

Hay un nodo sensor, construido con un Arduino, que usa sensores específicos para medir en tiempo real el voltaje y la corriente de un circuito.

Toda la información que recoge este nodo se envía de forma inalámbrica a un nodo coordinador. Este segundo Arduino recibe los datos, los procesa y, si detecta alguna anomalía o un consumo fuera de lo normal, puede tomar una decisión.

Para la comunicación entre ellos, el sistema utiliza módulos XBee, que son una tecnología confiable y de bajo consumo.

Finalmente, el nodo coordinador puede controlar un relevador para cortar o permitir el paso de la corriente, protegiendo así el equipo conectado.

La razón principal para usar este trabajo como base fue muy práctica: ya se había demostrado que funcionaba. Al ser un sistema implementado en un entorno real, nos proporcionaba una plataforma validada y estable. Esto nos permitió saltarnos toda la fase inicial de diseño conceptual y las pruebas de viabilidad, dándonos la seguridad de que la idea central era sólida y real.

Por todo esto, nuestro enfoque no fue replicar lo que ellos ya hicieron, sino tomar ese sistema funcional como un cimiento robusto para atacar un problema fundamental: la gestión energética. Planeamos generar mejoras energéticas en cuanto a consumo de batería, autonomía de los nodos y técnicas de recolección de energía, no solo para este sistema, sino con la intención de que sean soluciones más genéricas. El objetivo final es ayudar a que las redes de sensores que otras personas desarrollen puedan ser más eficientes energéticamente y alcanzar una mayor autonomía.

Aunque el documento de Zamora fue una excelente guía que nos permitió entender a fondo el diseño y el funcionamiento de su hardware, no incluía el código fuente. Esto significó que, para poder replicar el sistema, tuvimos que desarrollar todo el software desde cero. Este desafío, si bien fue grande, nos obligó a comprender a un nivel mucho más profundo la lógica de programación tanto del nodo sensor como del nodo coordinador. Tuvimos que escribir las rutinas para la lectura de los sensores, el procesamiento de los datos, el empaquetado de la información y toda la lógica de control del relevador.

Paralelamente al desarrollo del software, nos enfrentamos al reto logístico de conseguir los componentes físicos. Armar nuestros propios nodos implicó un proceso de búsqueda y compra de cada una de las piezas. Tuvimos que navegar por diversas páginas y proveedores en línea para encontrar los modelos exactos de los sensores de voltaje y corriente, las placas Arduino, los relevadores y, afortunadamente, los módulos de comunicación XBee no tuvimos que adquirirlos por nuestra cuenta ya que nos fueron proporcionados por el Asesor Agustín Esquivel. Este proceso fue fundamental para familiarizarnos con el hardware que estaríamos utilizando.

Selección y Gestión de la Fuente de Alimentación Autónoma

Dado que el objetivo central del proyecto es la autonomía energética, un paso fundamental fue seleccionar una fuente de alimentación para los nodos que fuera extraíble y recargable. Decidimos utilizar baterías de ion de litio del tipo 18650 por varias razones estratégicas:

* Alta Densidad Energética: Ofrecen una excelente capacidad de almacenamiento de energía en relación con su tamaño y peso, ideal para mantener los nodos compactos.
* Larga Vida Útil: Son recargables y soportan cientos de ciclos de carga y descarga, lo que las hace una opción sostenible y económica a largo plazo.
* Voltaje Estable: Proporcionan un voltaje de salida relativamente constante durante la mayor parte de su ciclo de descarga, lo cual es beneficioso para la estabilidad de los microcontroladores y sensores.
* Estándar de la Industria: El formato 18650 es un estándar global, lo que garantiza su disponibilidad y una amplia gama de accesorios compatibles.

Trabajar con baterías de ion de litio requiere precaución, especialmente durante la carga. Para manejar esto de forma segura y eficiente, en lugar de integrar un circuito de carga en cada nodo sensor, optamos por construir una estación de carga externa. Como se aprecia en la Figura 1, esta estación utiliza varios módulos de carga TP4056 para recargar múltiples baterías 18650 de forma simultánea. Este enfoque nos permite tener siempre un juego de baterías listas para ser intercambiadas en los nodos desplegados, simplificando el mantenimiento en campo. Fue necesario estudiar el funcionamiento de los módulos TP4056 para comprender su perfil de carga y sus mecanismos de protección, garantizando así un manejo seguro y que prolongue la vida útil de las baterías.

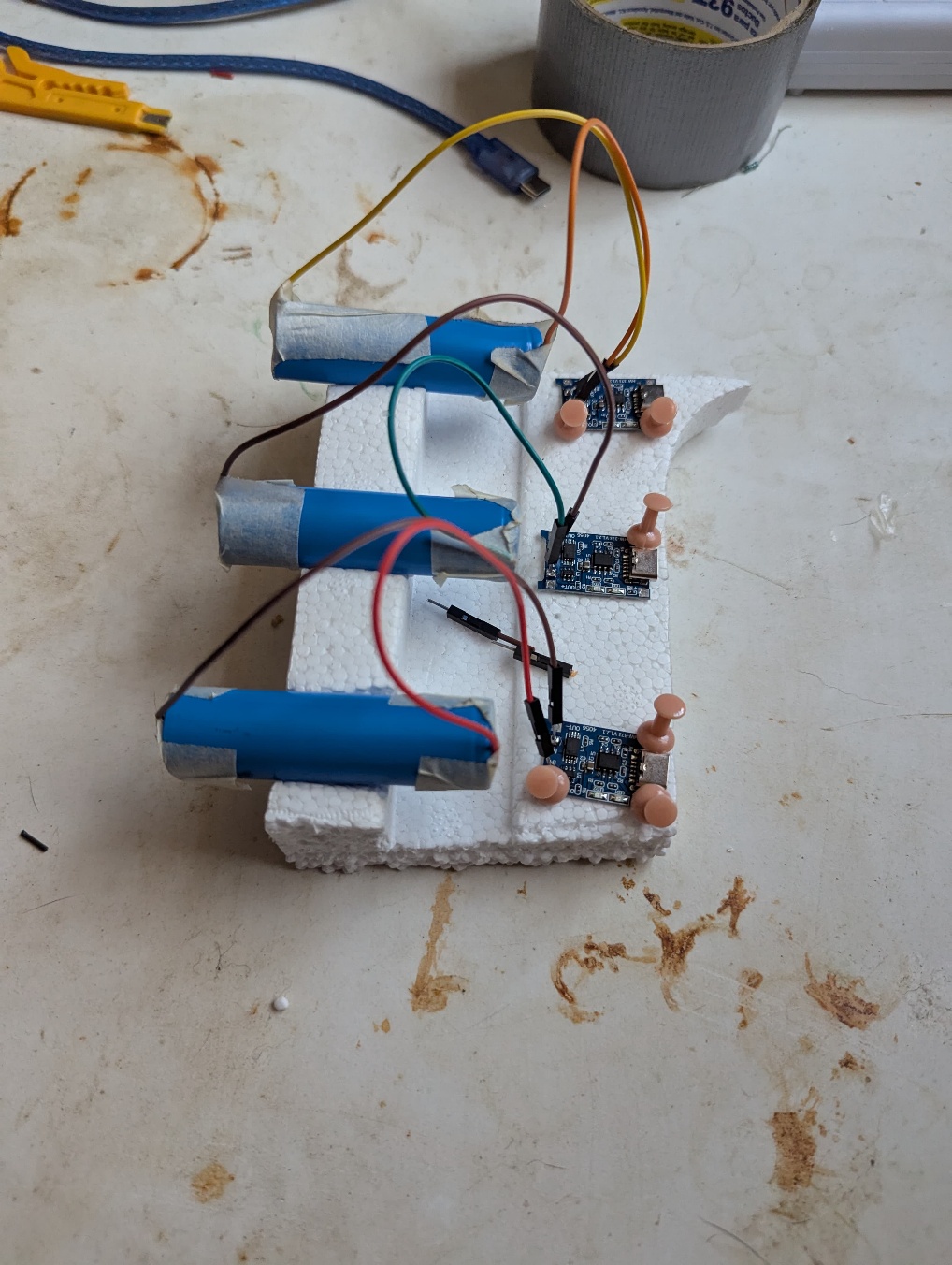


Figura 2. Módulo de carga de las baterías

Aprendizaje y Configuración de los Módulos XBee

Una vez que tuvimos los componentes, el primer gran obstáculo técnico fue la comunicación inalámbrica. Para empezar a trabajar, nos enfocamos en aprender a usar los módulos XBee, que son el corazón de la red. Esto requirió una inmersión profunda en su documentación, que es bastante extensa y compleja además de confusa ya que muchas versiones de firmware están descontinuadas, la documentación es algo ambigua y muchas veces lo que menciona no se relaciona al pie de la letra con la práctica.

El principal reto fue aprender a manejar XCTU, el software de configuración oficial de Digi. Al principio, peleamos mucho con la configuración; los módulos simplemente no se comunicaban entre sí. Pasamos bastante tiempo probando diferentes ajustes, entendiendo conceptos como el PAN ID (el identificador de la red), las direcciones de los módulos y los roles de "Coordinador" y "Router/End Device" (como se puede ver en la Figura 3) además de las topologías de red disponibles. Nos costó bastante lograr que establecfieran un enlace estable.

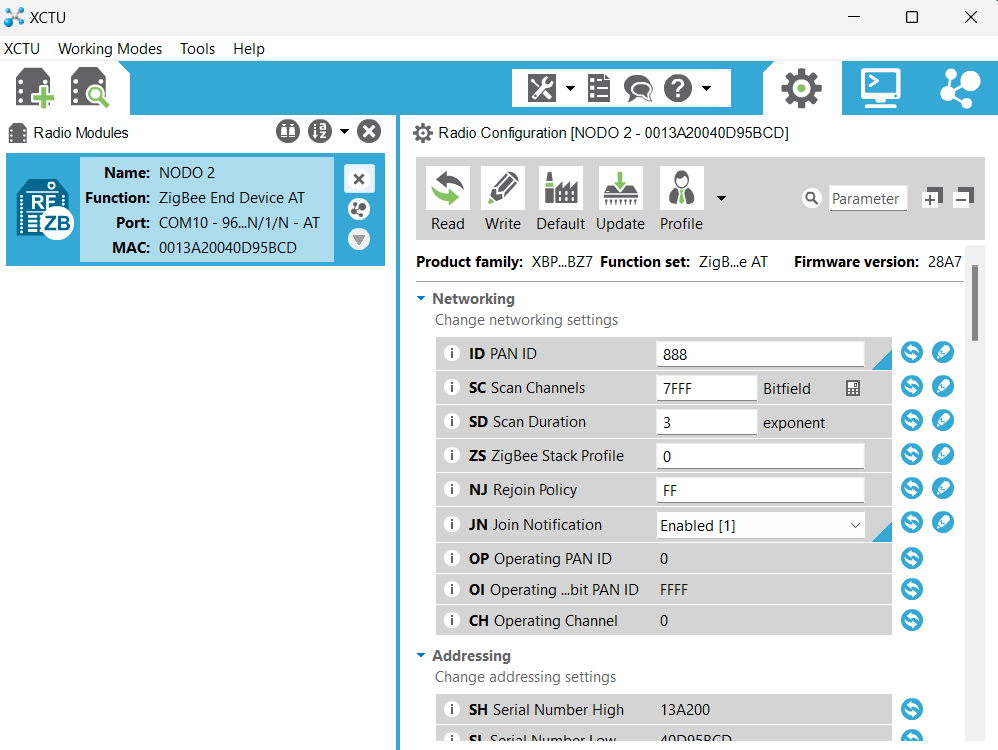


Figura 3. Configuración del End Device en XCTU

Finalmente, después de muchos intentos de prueba y error, logramos que se comunicaran. Las primeras pruebas exitosas las hicimos directamente con la consola integrada en el software XCTU., enviábamos un simple "hola" desde un módulo y veíamos cómo aparecía en la consola del otro, esto se puede apreciaren la . Ver ese mensaje llegar fue un gran avance, ya que nos confirmó que la configuración de la red era correcta y que estábamos listos para pasar a la siguiente fase: conectar los módulos a los Arduinos para empezar a enviar datos reales.

# Añadir imagen aqui

Con la comunicación inalámbrica ya resuelta, el siguiente y último paso de esta fase fue conectar todas las piezas. Integramos los módulos XBee con sus respectivas placas Arduino, conectamos los sensores de voltaje y corriente al nodo esclavo, y el módulo relevador al nodo coordinador. Cargamos el código que habíamos desarrollado desde cero en cada uno de los microcontroladores y procedimos a encender el sistema completo.

Realizamos pruebas simples para verificar que todo funcionara como un conjunto. Primero, programamos el nodo sensor para que leyera un valor y lo enviara al coordinador. Con el monitor serial abierto en el computador conectado al coordinador, esperamos y, efectivamente, vimos llegar el primer paquete de datos. Luego, enviamos un comando simple desde el coordinador al esclavo para activar y desactivar el relevador, y escuchamos el satisfactorio "clic" que confirmaba que la orden se había ejecutado correctamente.

El proceso de integración fue algo lento y metódico, ya que íbamos verificando cada conexión y cada parte del código paso a paso. Afortunadamente, no tuvimos demasiados problemas inesperados en esta etapa, en gran parte gracias a todo el trabajo previo de prototipado y configuración. Con el sistema base finalmente replicado y funcionando de manera estable, habíamos alcanzado nuestro primer gran objetivo. Ahora, con una plataforma operativa en nuestras manos, podíamos por fin pasar a la fase principal del proyecto: empezar a trabajar de verdad en las mejoras y optimizaciones que nos habíamos propuesto.

## Análisis de mejoras a realizar

Una vez que tuvimos una réplica funcional del sistema base, el siguiente paso fue realizar un análisis crítico para identificar sus debilidades y las oportunidades de mejora. Con un sistema estable en nuestras manos, podíamos experimentar y evaluar su rendimiento de manera objetiva. Este análisis se abordó desde tres perspectivas complementarias: el software, el hardware y la sinergia entre ambos. El objetivo era definir varias "rutas de ataque" claras para optimizar el sistema, enfocándonos principalmente en la eficiencia energética y la autonomía.

Análisis desde la Perspectiva del Software

El código que desarrollamos, aunque cumplía su función, fue una primera versión escrita con el objetivo de replicar la lógica del sistema original. Al analizarlo, identificamos un amplio margen para la optimización. El análisis se centró en la posibilidad de reescribir partes clave del código para hacerlo más eficiente. A grandes rasgos, se concluyó que podíamos reducir el consumo de recursos del microcontrolador mediante códigos más optimizados, por ejemplo, utilizando tipos de variables más eficientes, implementando rutinas de bajo consumo y minimizando las operaciones innecesarias para que el procesador estuviera activo el menor tiempo posible.

Análisis desde la Perspectiva del Hardware

Desde el punto de vista del hardware, el análisis se enfocó en evaluar los componentes del diseño original frente a las alternativas más modernas disponibles en el mercado. Si bien los componentes base como el Arduino UNO y los módulos XBee son confiables, la tecnología ha avanzado. Identificamos que la ruta de mejora más prometedora era la selección de hardware más eficiente. Esto implicaba buscar microcontroladores con un consumo energético mucho menor en estados activos y de reposo, así como sensores y módulos de comunicación diseñados específicamente para aplicaciones de bajo consumo (IoT) además de hardware auxiliar de menor consumo.

Análisis de la Sinergia Hardware-Software

Finalmente, entendimos que las mejoras más impactantes no vendrían de optimizar el software y el hardware por separado, sino de hacerlos trabajar en conjunto de una manera más inteligente. Este análisis definió la ruta de ataque principal del proyecto: aprovechar las características de un hardware de bajo consumo mediante software diseñado específicamente para ello. Por ejemplo, de nada serviría tener un microcontrolador con modos de sueño profundo (una característica de hardware) si el software no está programado para utilizarlos correctamente. De esta forma, definimos que la estrategia general sería seleccionar primero un hardware superior y luego desarrollar un software a medida que explotara al máximo sus capacidades de ahorro energético.

## Mejoras al hardware

Para mejorar la capacidad de diagnóstico y seguimiento del sistema, se incorporó un módulo de tarjeta micro SD en el nodo coordinador y también en el nodo sensor. Esta adición nos permite implementar un sistema de registro de eventos o "logs". Cada vez que el coordinador recibe datos, realiza una acción importante o detecta un error, puede escribir una entrada en un archivo de texto dentro de la tarjeta de memoria. De esta manera, si algo falla, tenemos un historial detallado para revisar qué salió mal y en qué momento, facilitando enormemente la depuración y el análisis del comportamiento del sistema a lo largo del tiempo.

### Actualización de Microcontroladores: Arduino Nano para el Sensor y ESP32 para el Coordinador

Para comenzar con la optimización del hardware, una de las decisiones más importantes fue reemplazar los microcontroladores Arduino UNO que usamos para replicar el sistema base. Aunque son excelentes para prototipar, no son la opción más eficiente en términos de tamaño y capacidades para un sistema final. Por ello, seleccionamos dos placas diferentes y más adecuadas para las tareas específicas de cada nodo.

Nodo Sensor: Adopción del Arduino Nano

Para el nodo encargado de tomar las mediciones, cambiamos el Arduino UNO por un Arduino Nano. La razón principal fue la reducción drástica de tamaño. El Nano utiliza el mismo microcontrolador que el UNO (el ATmega328P), por lo que tiene prácticamente la misma capacidad de procesamiento y el mismo número de pines que necesitábamos. Esto significó que podíamos migrar nuestro código sin hacer grandes cambios.

La ventaja es puramente física, pero muy relevante: al ser mucho más compacto, el Arduino Nano nos permite diseñar un nodo sensor final mucho más pequeño y discreto, más fácil de instalar en cualquier lugar. Para un dispositivo que funcionará con baterías, reducir el tamaño y el peso es un paso clave hacia un diseño más eficiente y práctico.

Nodo Coordinador: Potenciando con el ESP32

El cambio más significativo se realizó en el nodo coordinador. Reemplazamos el Arduino UNO por una placa de desarrollo ESP32. Esta decisión fue estratégica y abrió un abanico de nuevas posibilidades para el proyecto. Mientras que el Arduino Nano fue una mejora de forma, el ESP32 fue una mejora de capacidades y futuro.

Las ventajas de usar un ESP32 como coordinador son enormes:

Potencia de Procesamiento: El ESP32 es un microcontrolador de doble núcleo y mucho más rápido que el de un Arduino. Esto le da la capacidad de procesar la información de varios nodos sensores simultáneamente, realizar cálculos más complejos en tiempo real y gestionar múltiples tareas sin despeinarse.

Conectividad Integrada: Esta es la mejora clave. El ESP32 tiene Wi-Fi y Bluetooth incorporados de fábrica. Esto elimina la dependencia exclusiva de la red XBee para comunicarse con el mundo exterior. Ahora, nuestro nodo coordinador puede conectarse directamente a una red Wi-Fi y, por lo tanto, a Internet, gracias a la conexión a Internet, podemos implementar funcionalidades que eran impensables con el diseño original.

En resumen, al asignar un Arduino Nano al nodo sensor, optimizamos el tamaño y la eficiencia para una tarea simple. Al asignar un ESP32 al nodo coordinador, le dimos al sistema un cerebro mucho más potente y, lo más importante, una puerta de acceso al Internet de las Cosas (IoT), preparando el proyecto para futuras expansiones.

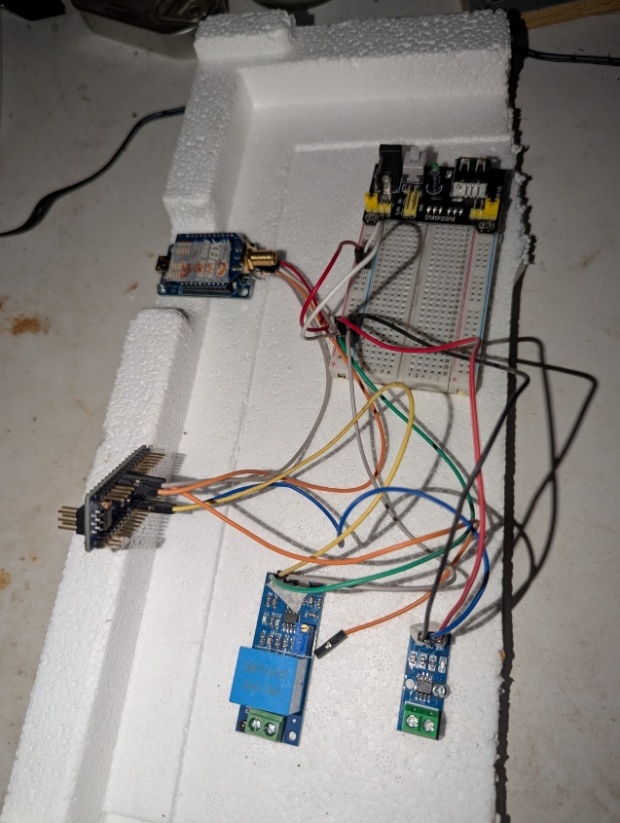


Figura 4. Nodo Sensor

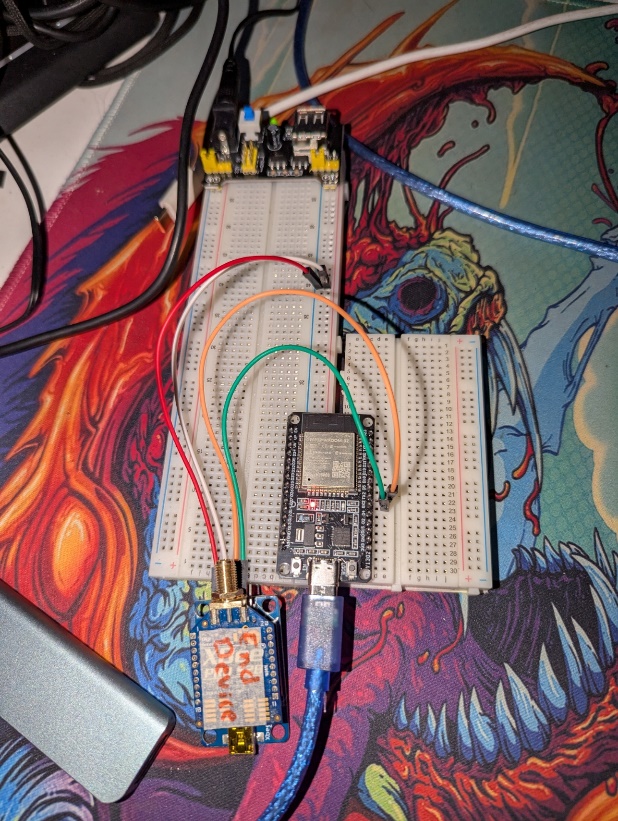


Figura 5. Nodo con Esp32

Después de seleccionar los nuevos componentes, el diseño final de nuestros dispositivos y sus conexiones quedó definido. En la Figura 2 se puede ver una foto del prototipo del nodo sensor ya armado, el cual montamos sobre una placa de unicel para mantener todo en su lugar durante las pruebas; todos sus componentes y conexiones se detallan en la Tabla 1. A su vez, el montaje final del nodo coordinador con el ESP32 como su pieza central se muestra en la Figura 3 , mientras que sus respectivas conexiones y componentes se pueden consultar en la Tabla 2

|  |  |  |
| --- | --- | --- |
| Pin lado A | Pin lado B | Notas |
| Adaptador Xbee TX (DOUT) | Arduino Nano Pin D2 |  |
| Adaptador Xbee RX (DIN) | Arduino Nano Pin D3 |  |
| Adaptador Xbee VCC | Fuente de alimentación 5V |  |
| Adaptador Xbee GND | GND de Fuente |  |
| ACS712 OUT | Arduino nano Pin A0 |  |
| ACS712 VCC | Fuente de alimentación 5V |  |
| ACS712 GND | GND de Fuente |  |
| ZMPT101B OUT | Arduino Nano Pin A1 |  |
| ZMPT101B VCC | Fuente de alimentación 5V |  |
| ZMPT101B GND | GND de Fuente |  |
| Relay IN | Arduino Nano Pin D4 |  |
| Relay VCC | Fuente de alimentación 5V |  |
| Relay GND | GND de Fuente |  |

Tabla 1. Pines actualizados del nodo sensor con Arduino Nano

|  |  |  |
| --- | --- | --- |
| Pin lado A | Pin lado B | Notas |
| Adaptador Xbee TX 2(DOUT) | ESP32-WROOM-32 GPIO16 (RX2) | Entrada de datos al ESP32 |
| Adaptador Xbee RX 3 (DIN) | ESP32-WROOM-32 GPIO17 (TX2) | Salida de órdenes desde el ESP32 |
| Adaptador Xbee VCC | Fuente de alimentación 5V | El adaptador aguanta 5v |
| Adaptador Xbee VCC | GND de Fuente |  |

Tabla 2. Pines actualizados del nodo Coordinador con Esp32

### Actualización de la Fuente de Alimentación

El siguiente punto de ataque para mejorar la eficiencia del sistema fue, lógicamente, la gestión de la energía. En cualquier dispositivo alimentado por baterías, el componente que se encarga de regular el voltaje es crítico, ya que un mal diseño en esta etapa puede desperdiciar una cantidad enorme de energía. Por ello, nuestra primera mejora estratégica fue reemplazar la fuente de alimentación original por una solución mucho más moderna y eficiente.

Del Regulador Lineal (MB102) al Regulador Conmutado (MP1584)

El Módulo Original: MB102

En prototipos iniciales y en muchos proyectos básicos, es común usar fuentes de alimentación como la MB102. Este módulo se basa en un regulador de voltaje lineal (generalmente el AMS1117). Para entender por qué es ineficiente, hay que saber cómo funciona: un regulador lineal toma un voltaje de entrada más alto (por ejemplo, 9V de una batería) y lo reduce al voltaje de salida deseado (por ejemplo, 5V para el Arduino). Para lograr esto, disipa la diferencia de voltaje (en este caso, 4V) en forma de calor, este se puede apreciar de mejor manera en la

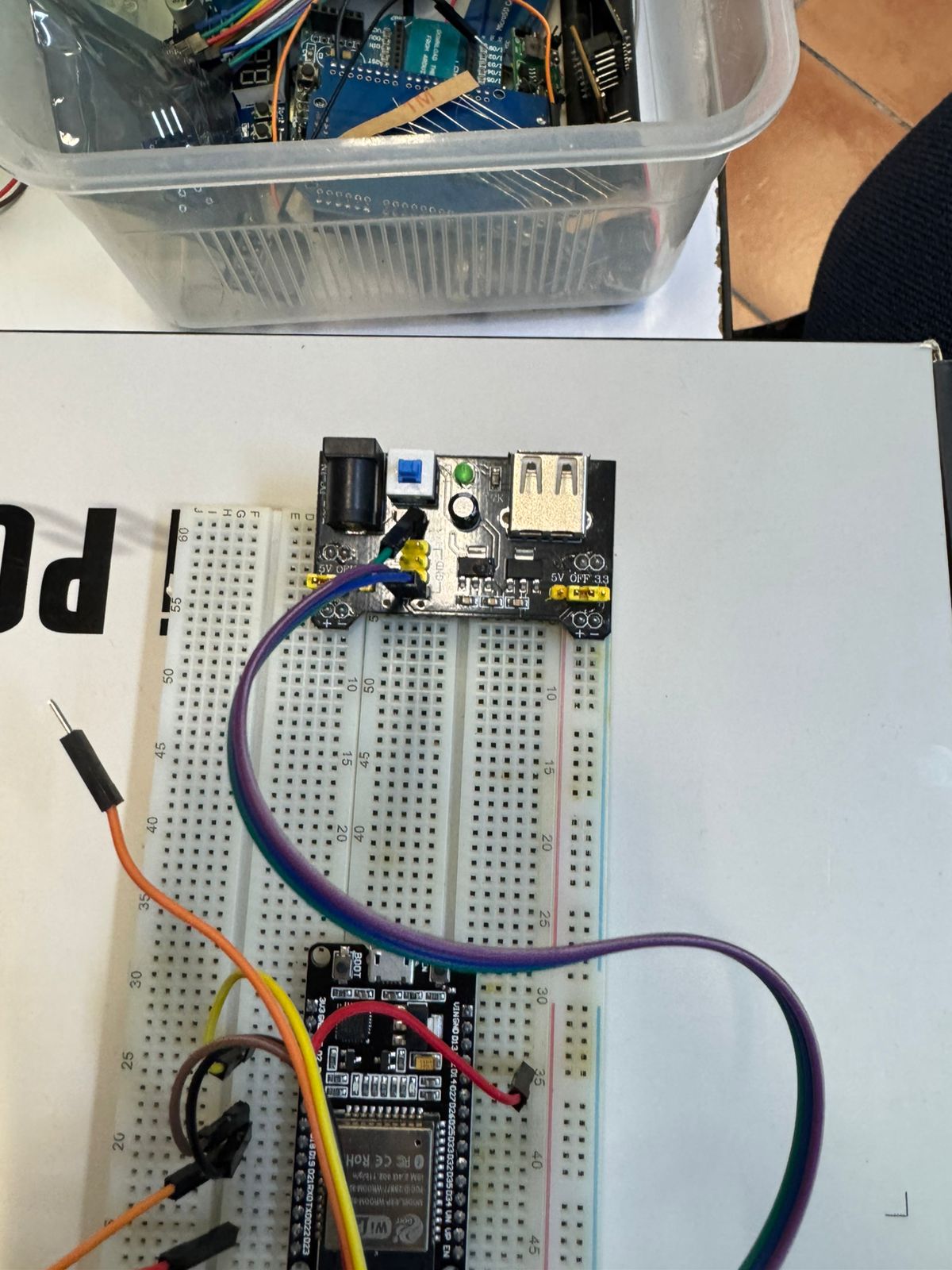


Figura 6. Regulador MB102

Este proceso es increíblemente derrochador. Es como si para llenar un vaso de agua tuvieras una manguera abierta al máximo y usaras una esponja para absorber todo el exceso que no cabe en el vaso. La esponja se calienta y el agua (la energía) se desperdicia. En nuestro sistema, esto significa que una parte considerable de la preciosa carga de la batería se convierte directamente en calor inútil, reduciendo drásticamente la autonomía del nodo antes de que la energía llegue siquiera al microcontrolador. Cuanto mayor es la diferencia entre el voltaje de la batería y el voltaje que necesita el dispositivo, más energía se desperdicia.

La Solución Mejorada: Módulo Regulador MP1584

Para solucionar este grave problema de ineficiencia, reemplazamos el MB102 por un módulo basado en el chip MP1584EN. Este componente es un regulador de voltaje conmutado, específicamente un convertidor "buck" o reductor. Su forma de trabajar es radicalmente diferente y mucho más inteligente.

En lugar de quemar el exceso de energía, un regulador conmutado funciona encendiendo y apagando un interruptor interno a una frecuencia muy alta (cientos de miles de veces por segundo). Al hacer esto, transfiere pequeños paquetes de energía desde la fuente a un circuito de almacenamiento (compuesto por un inductor y un capacitor), que suaviza estos pulsos para entregar un voltaje de salida constante y estable.

La gran ventaja de este método es su altísima eficiencia, que a menudo supera el 90%. Esto significa que casi toda la energía que se extrae de la batería se entrega efectivamente al dispositivo. Volviendo a la analogía, es como usar un grifo que se abre y se cierra rápidamente para dejar pasar solo la cantidad exacta de agua que necesitas. El desperdicio es mínimo. Para nuestro proyecto, este cambio es fundamental, ya que se traduce directamente en una mayor duración de la batería y, por lo tanto, en una mayor autonomía para nuestros nodos sensores.

Además, el módulo MP1584 es regulable. Viene con un pequeño potenciómetro que nos permite ajustar con precisión el voltaje de salida que necesitamos. Esta característica nos da una flexibilidad enorme, ya que podemos alimentar el módulo con una amplia gama de fuentes de energía (por ejemplo, un pack de baterías de 7.4V, una de 12V, o incluso un pequeño panel solar) y ajustar la salida para que sea exactamente de 5V para un Arduino o de 3.3V para un ESP32. Esta versatilidad hace que nuestro diseño no solo sea más eficiente, sino también mucho más adaptable a diferentes escenarios y tipos de baterías lo que nos es muy práctico debido a la naturaleza de hardware cambiante del proyecto

Este cambio de hardware no cambia en nada a nuestros pines más allá de cambiar la entrada de voltaje por lo tanto no fue necesario realizar la tabla de cableado nuevamente como se puede ver en la Figura 7.

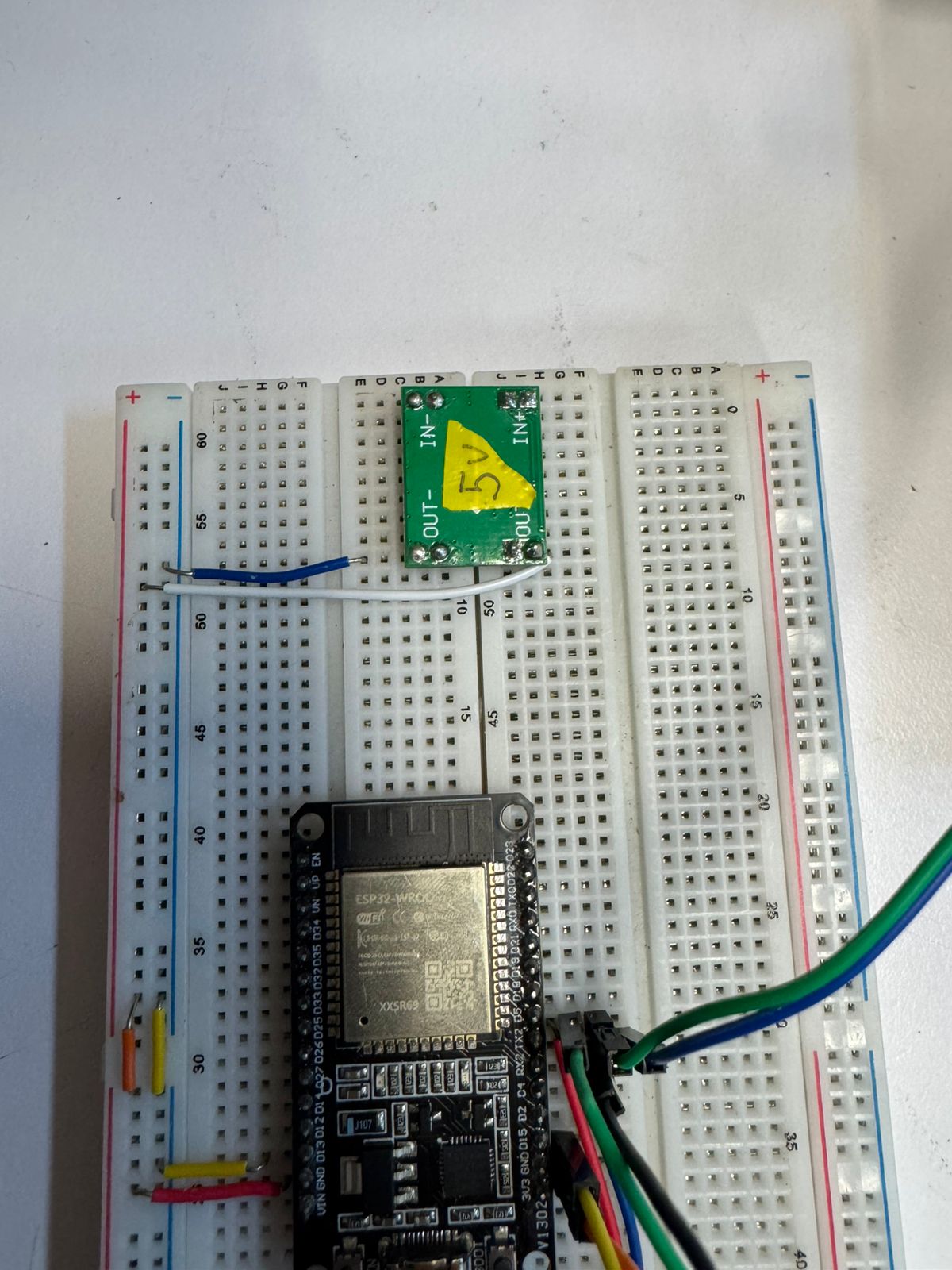


Figura 7. Módulo MP1584 implementado en nuestros nodos

Una mejora adicional, aunque parezca menor, tuvo un gran impacto en la fiabilidad de nuestros montajes de prueba. Decidimos reemplazar los cables Dupont flexibles tradicionales, que suelen colgar y crear un desorden en la protoboard, por jumper wires de alambre sólido. Estos cables, al ser rígidos, se insertan directamente y quedan pegados a la superficie de la protoboard. Este cambio nos facilitó enormemente el trabajo, ya que el circuito se volvió mucho más limpio y fácil de revisar. Lo más importante es que es mucho más difícil que se desconecten accidentalmente, eliminando falsos fallos durante las pruebas y haciendo todo el proceso de prototipado más rápido y seguro, apareciendo estos cambios en la Figura 4.

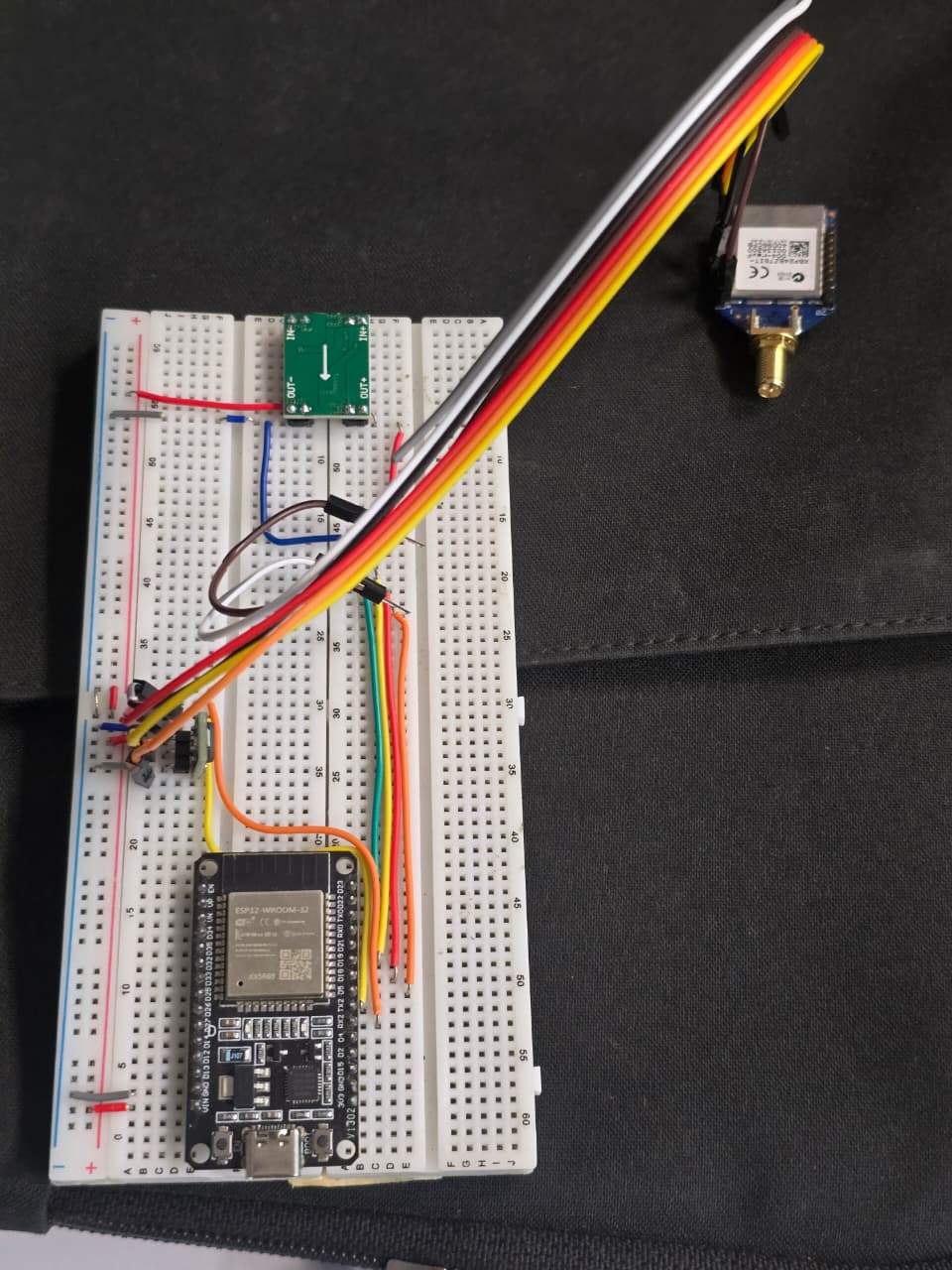


Figura 8.Prototipo con regulador MP1584 y jumpers wires

## Mejoras al Software

En cuanto al software, que era el área donde sentíamos que teníamos mayor capacidad para expandir el proyecto, comenzamos a trabajar directamente sobre nuestros códigos iniciales. Sin embargo, pronto nos dimos cuenta de que este enfoque no iba a funcionar. De nada sirve hacer una mejora si solo funciona para nuestro caso particular; la idea era crear algo que pudiera ser útil para otras personas y que se pudiera adaptar a otras redes de sensores. El objetivo es que el código sea reciclable, no que otros tengan que empezar desde cero. Por eso, decidimos cambiar de estrategia y optamos por desarrollar nuestras mejoras en formato de librería.

Para poder hacer esto, primero fue necesario aprender cómo funciona el ecosistema de Arduino a un nivel más profundo, específicamente cómo se manejan sus librerías, cómo se accede a ellas desde el código y cómo funciona la propia indexación del IDE. Una vez que dominamos esa parte técnica, pasamos a la fase de empezar a escribir el código de nuestra nueva librería.

### Desarrollo de la Librería CodecWSN

Basándonos en los principios de optimización en la transmisión de datos, como los analizados por Barker & Swany (2020) sobre la compresión de mensajes en comunicaciones, hemos desarrollado la librería “CodecWSN”. Si bien el artículo de Pedro nos brindó un marco teórico sobre la importancia de la eficiencia en la comunicación, la estrategia específica de compactación de datos para nuestra red de sensores inalámbricos (WSN) fue una iniciativa propia. El objetivo principal de “CodecWSN” es transformar la información de los sensores (ID, voltaje, corriente y voltaje de batería) a un formato binario compacto, minimizando el tamaño de los datos a transmitir y, por ende, optimizando todo el proceso de comunicación.

La estrategia central de nuestra librería es la serialización directa de los datos a un formato binario de tamaño fijo. En lugar de enviar la información como texto plano (ASCII), que es ineficiente y verboso, “CodecWSN” convierte las mediciones en una estructura de solo 8 bytes. Por ejemplo, un valor de voltaje como "12.34" ocuparía 5 bytes como texto. Nuestra librería lo representa en solo 2 bytes (un “int16\_t”), logrando una compresión significativa sin pérdida de precisión, ya que manejamos los valores como centésimas de voltio. Este enfoque se aplica a cada una de las variables, resultando en un paquete de datos extremadamente ligero y predecible, es sencillo de usar simplemente fue necesario realizar normalización por escala entera para enviar como enteros simples.

El ahorro en el espacio de transmisión es considerable. Si enviáramos los mismos datos en un formato de texto simple como, por ejemplo, “ID:1,V:23.50,C:150,VB:4.10”, estaríamos utilizando aproximadamente 30 bytes. Al emplear nuestra librería, los mismos datos se codifican en un paquete de solo 8 bytes. Esto representa una reducción de más del 70% en el tamaño del paquete. Esta optimización no solo acelera la transmisión, sino que también libera un ancho de banda crucial en la red, permitiendo que más sensores se comuniquen de manera fluida y sin congestión.

La consecuencia más importante de esta reducción de datos es el ahorro energético. En los nodos de sensores alimentados por baterías, el módulo de radio es uno de los componentes que más energía consume, y su consumo está directamente relacionado con el tiempo que permanece activo transmitiendo. Al enviar paquetes un 70% más pequeños, el tiempo de transmisión se reduce drásticamente, lo que se traduce en una menor demanda de energía y, por lo tanto, en una mayor autonomía y vida útil para las baterías de los dispositivos en campo. De esta manera, “CodecWSN” no solo optimiza la comunicación, sino que también mejora la sostenibilidad y viabilidad de toda la red de sensores.

Una vez que implementamos la transmisión binaria, nos topamos con un problema frustrante y difícil de diagnosticar. Durante las pruebas, observamos que, aunque la mayoría de los mensajes llegaban correctamente, algunos datos eran completamente incoherentes. Al principio, pensamos que había un error en nuestra lógica de codificación o decodificación. Para investigar, creamos una serie de "mensajes de prueba falsos", donde en lugar de enviar mediciones reales, transmitíamos patrones de bytes conocidos y predecibles. Esto nos permitió saber exactamente qué se enviaba y comparar con lo que se recibía. Fue entonces cuando confirmamos el fallo: el receptor a veces interpretaba los datos de manera incorrecta porque la trama de datos se "desfasaba", tal como se ilustra en la Figura 5 . Después de mucha experimentación, descubrimos que la causa era la pérdida o corrupción de un solo byte durante la transmisión inalámbrica. Si un byte se perdía, el receptor comenzaba a leer toda la secuencia de bytes subsiguiente de forma desalineada, perdiendo por completo la estructura del paquete.

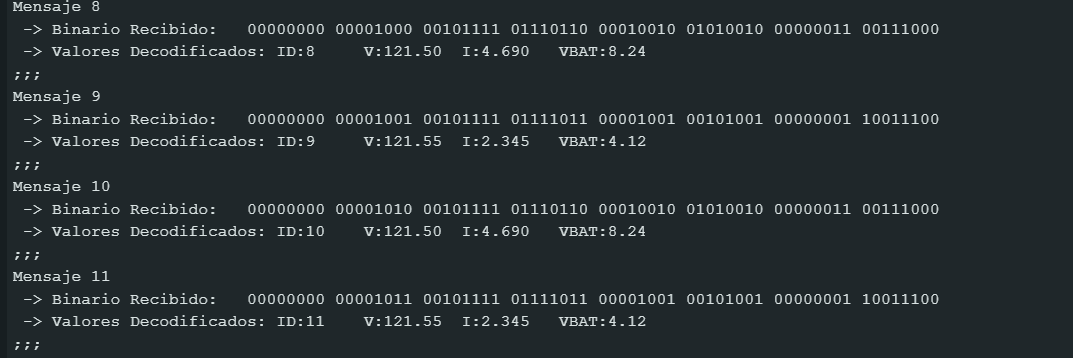


Figura 9. Desfase de bits en código inicial

Para solucionar este problema de desincronización y corrupción, investigamos estrategias de comunicación robustas y llegamos a una solución de dos capas. La primera fue implementar un mecanismo de entramado (framing). En lugar de enviar simplemente los 8 bytes de datos, decidimos envolverlos en una estructura más grande llamada "trama". Esta trama incluye una secuencia de inicio única, conocida como SOF (Start Of Frame), que en nuestro caso son los bytes 0xAA y 0x55. El receptor ahora ignora cualquier dato entrante hasta que detecta esta secuencia específica, lo que le permite resincronizarse automáticamente incluso si se han perdido bytes previamente.

Sin embargo, el framing por sí solo no resuelve la corrupción de datos dentro de la trama. Para ello, implementamos la segunda capa: un CRC (Cyclic Redundancy Check). Un CRC es, en esencia, una "huella digital" matemática altamente fiable que se calcula a partir de un bloque de datos. La lógica es la siguiente:

* 1. En el emisor: Antes de enviar la trama, aplicamos un algoritmo matemático (en nuestro caso, el CRC16-CCITT) al contenido del mensaje (versión, longitud y los 8 bytes de datos). El resultado es un número de 16 bits (2 bytes) único para ese contenido específico. Este valor de CRC se adjunta al final de la trama.
  2. En el receptor: Una vez que el receptor detecta una trama completa (usando el SOF y la longitud esperada), realiza exactamente el mismo cálculo de CRC sobre los datos que ha recibido.
  3. Verificación: Finalmente, compara el CRC que acaba de calcular con el CRC que venía al final de la trama recibida. Si los dos valores son idénticos, podemos estar casi un 100% seguros de que los datos no se corrompieron durante la transmisión. Si no coinciden, significa que al menos un bit cambió en el camino, por lo que el receptor descarta la trama completa por ser inválida y espera la siguiente.

Esta doble estrategia de sincronización con SOF y validación con CRC nos proporcionó un canal de comunicación extremadamente robusto. Ahora, el sistema no solo es eficiente en términos de energía y ancho de banda gracias a la compresión binaria, sino que también es altamente fiable, ya que puede recuperarse de pérdidas de sincronización y descartar activamente los datos corruptos.

### Desarrollo de la Librería AdaptiveTXWSN

En las redes de sensores inalámbricos (WSN), el recurso más preciado es, sin duda, la energía de la batería. Un nodo sensor puede tener el mejor hardware y el software más eficiente, pero su vida útil está intrínsecamente ligada a la duración de su fuente de poder. Partiendo de esta premisa, hemos desarrollado la librería AdaptiveTXWSN, un sistema de gestión de transmisión que no opera con una frecuencia fija, sino que adapta su comportamiento en tiempo real basándose en el estado energético del nodo.

El concepto central es la Transmisión Adaptativa. En lugar de transmitir datos a un intervalo constante (por ejemplo, cada 30 segundos), la librería ajusta dinámicamente este periodo. Cuando la batería está completamente cargada, el nodo puede permitirse enviar datos con alta frecuencia para proporcionar información actualizada. Sin embargo, a medida que el nivel de voltaje desciende, la librería reduce proactivamente la frecuencia de transmisión para conservar la energía restante, extendiendo así significativamente la vida operativa del dispositivo en el campo.

La inteligencia de AdaptiveTXWSN se basa en tres pilares fundamentales que trabajan en conjunto:

* 1. Monitoreo Preciso del Nivel de Energía: La librería mide continuamente el voltaje de la batería. Esto se logra a través de un pin ADC del microcontrolador, y es totalmente configurable para adaptarse a cualquier hardware mediante un divisor de voltaje. Para asegurar lecturas estables y evitar fluctuaciones por ruido eléctrico, promedia varias muestras consecutivas, una práctica que robustece la medición.
  2. Lógica de Decisión Basada en Umbrales: El sistema clasifica el estado de la batería en tres niveles operativos: ALTO, MEDIO y BAJO. Estos niveles no son fijos, sino que se definen mediante umbrales de voltaje que el usuario puede configurar. Por ejemplo, se puede establecer que por encima de 3.9V el nivel sea ALTO, entre 3.6V y 3.9V sea MEDIO, y por debajo de 3.6V sea BAJO. Además, incluye una función de seguridad crítica: un umbral de corte (corteVoltaje\_V). Si el voltaje cae por debajo de este punto, la librería detiene todas las transmisiones para proteger la batería de una descarga profunda, que podría dañarla permanentemente.
  3. Frecuencia de Transmisión Dinámica: Este es el núcleo de la adaptación. Cada uno de los niveles energéticos (ALTO, MEDIO, BAJO) tiene asociado un período de envío configurable. Siguiendo el ejemplo anterior, el nodo podría transmitir cada 5 segundos en nivel ALTO, cambiar a cada 15 segundos en nivel MEDIO, y finalmente reducir la frecuencia a cada 2 minutos en nivel BAJO, priorizando la supervivencia del nodo sobre la inmediatez de los datos.

Durante el desarrollo inicial, nos encontramos con un problema clásico en los sistemas de control: el "rebote" o "chattering". Si el voltaje de la batería fluctuaba justo en el borde de un umbral (por ejemplo, 3.60V), el ruido en la lectura o pequeños cambios de carga podían hacer que el sistema cambiara rápidamente y de forma errática entre el nivel MEDIO y BAJO. Esto provocaba que la frecuencia de transmisión saltara impredeciblemente entre 15 segundos y 2 minutos, un comportamiento ineficiente y no deseado.

La solución implementada fue la histéresis. En lugar de usar un único umbral para cambiar de estado en ambas direcciones, la librería utiliza dos umbrales separados por un pequeño margen:

* Para bajar de MEDIO a BAJO, el voltaje debe caer por debajo del umbral menos un diferencial (ej. 3.60V - 3%).
* Para subir de BAJO a MEDIO, el voltaje debe superar el umbral más un diferencial (ej. 3.60V + 3%).

Este mecanismo crea una "zona muerta" que absorbe las pequeñas fluctuaciones, asegurando que los cambios de estado sean deliberados y estables. Es el mismo principio que utiliza un termostato para evitar que el aire acondicionado se encienda y apague constantemente.

Una de las mayores ventajas de AdaptiveTXWSN es que es totalmente configurable. Prácticamente todos sus parámetros de funcionamiento pueden ser ajustados por el usuario, no solo en la configuración inicial sino también dinámicamente durante la ejecución. Esto permite:

* Adaptarse a cualquier tipo de batería: Baterías LiPo, Li-ion o LiFePO4 tienen curvas de descarga diferentes. Con esta librería, se pueden ajustar los umbrales para que coincidan con las características específicas de la batería utilizada.
* Ajustarse a los requisitos de la aplicación: Una aplicación de monitoreo ambiental puede permitirse intervalos largos, mientras que un sistema de alarma necesita reportes más frecuentes. Los períodos se pueden adaptar a cada necesidad.
* Realizar ajustes en campo: Es posible modificar los umbrales o los períodos de forma remota, permitiendo optimizar el comportamiento de la red sin necesidad de reprogramar físicamente cada nodo.

En conclusión, AdaptiveTXWSN transforma un simple temporizador de envío en un gestor de energía inteligente. Su capacidad para tomar decisiones en tiempo real basadas en el estado del recurso más vital —la batería— la convierte en una herramienta indispensable para crear redes de sensores robustas, autónomas y de larga duración.

## Mejoras de la Sinergia Hardware-Software: Librería EnergyWSN

El avance más significativo en nuestro proyecto no provino de una mejora aislada, sino del análisis profundo de cómo el hardware y el software interactúan. Entendimos que para lograr una autonomía energética radical, el nodo sensor no podía funcionar de manera monolítica, con todos sus componentes encendidos permanentemente. La solución fue diseñar un ciclo de trabajo por pulsos, donde el nodo pasa la mayor parte de su tiempo en un estado de letargo de consumo casi nulo y solo "despierta" durante breves y precisas ventanas de tiempo para cumplir sus tareas.

Esta estrategia requirió una simbiosis perfecta entre modificaciones físicas (hardware) y una lógica de control inteligente (software) que implementamos como una librería llamada EnergyWSN.

Para que el software pudiera gestionar la energía de forma granular, primero tuvimos que realizar cambios clave en el hardware del nodo:

1. Línea de Voltaje Independiente para Sensores: Aislamos la alimentación de todos los sensores (ZMPT, ACS712) en un riel de potencia separado. Esto es crucial porque muchos sensores consumen energía de forma pasiva incluso cuando no están tomando mediciones.
2. Control de Potencia por Software: Para gestionar esta nueva línea de voltaje, integramos un MOSFET conectado a un transistor NPN que actúa como un interruptor electrónico de alta eficiencia. El microcontrolador ahora puede, mediante un simple pulso digital, encender o apagar por completo todo el conjunto de sensores, eliminando cualquier consumo fantasma, y debido a la naturaleza del MOSFET que se controla por Voltaje por lo que para mantenerlo encendido su consumo es prácticamente cero
3. Gestión Avanzada del Módulo de Radio: Par|a controlar el módulo XBee, que opera a 3.3V, se añadió un par de pines conectados que permiten al Arduino (de 5V) "pedirle" al XBee que entre en su modo de hibernación de ultra bajo consumo, utilizando sus pines de control de energía específicos (SLEEP\_RQ).

Con el hardware preparado, el código se convierte en el director de orquesta que ejecuta una secuencia de operaciones meticulosamente cronometrada para minimizar el consumo en cada paso a partir de las funciones entregadas por nuestra librería. La lógica del programa es la siguiente:

1. El Despertar Controlado: El ciclo comienza cuando el microcontrolador sale de su estado de sueño profundo (deep sleep), activado por un temporizador interno.
2. Activación y Estabilización de Sensores: Lo primero que hace es enviar una señal al MOSFET para energizar los sensores. Inmediatamente después, el programa espera un breve instante (un delay de milisegundos). Este pequeño tiempo de espera es fundamental, ya que los componentes electrónicos necesitan un momento para que sus voltajes se estabilicen tras el encendido. Omitir este paso podría resultar en mediciones erróneas o inconsistentes.
3. Adquisición Rápida de Datos: Una vez estabilizados los sensores, el microcontrolador realiza las lecturas de voltaje y corriente (analogRead) de la forma más rápida posible.
4. Desactivación Inmediata: Tan pronto como se tienen las mediciones, el software corta la alimentación de los sensores a través del MOSFET. Con esto, nos aseguramos de que solo consuman energía durante la fracción de segundo estrictamente necesaria.
5. Almacenamiento Local: De forma siguiente, los datos se guardan en una tarjeta SD como respaldo local para evitar pérdidas.
6. Comunicación Eficiente: Con los sensores ya apagados, el programa "despierta" al módulo de radio XBee. Compone el paquete de datos y lo transmite. Inmediatamente después de confirmar el envío, le ordena al XBee que vuelva a dormir. El radio, que es el componente de mayor consumo, solo está activo durante el mínimo tiempo indispensable.
7. El Sueño Profundo: Una vez completadas todas las tareas, el microcontrolador ejecuta la instrucción final: entrar en un modo de hibernación (LowPower.sleepFor\_ms). En este estado, consume apenas unos microamperios. Permanece así durante el tiempo configurado (desde segundos hasta varios minutos) hasta que el ciclo deba repetirse.

Esta metodología tan precisa permite la mayor ventaja de ahorro energético que hemos avanzado hasta ahora y la más completa ya que nos da una solución robusta y reutilizable

Lograr esta orquestación precisa entre hardware y software no fue un camino directo. Fue un proceso iterativo lleno de investigación, retos inesperados y un aprendizaje práctico invaluable. La implementación de esta estrategia de ultra bajo consumo representó uno de los mayores desafíos técnicos del proyecto y por eso es que estamos tan orgullosos de EnergyWSN.

El primer paso fue adquirir los componentes necesarios. Conseguir el MOSFET adecuado, con las características de bajo voltaje de activación que necesitábamos, resultó ser sorprendentemente difícil. Cuando finalmente lo encontramos, cometimos un error de novato: lo compramos en un encapsulado SOT-23, un formato de montaje superficial. Estos componentes son increíblemente pequeños, diseñados para ser soldados por máquinas en producción en masa.

Para poder usarlos en nuestros prototipos, tuvimos que comprar adaptadores de SOT-23 a formato DIP/SIP, que es el estándar para protoboards. Aunque ya teníamos experiencia soldando, enfrentarnos a estos componentes fue nuestro mayor reto de soldadura hasta la fecha. Trabajar con piezas tan diminutas requirió un pulso de cirujano, herramientas de precisión y mucha paciencia, pero superarlo nos dio una nueva y valiosa habilidad

No bastaba con solo conectar los componentes. Tuvimos que sumergirnos en la teoría para usarlos de manera efectiva. Invertimos tiempo en aprender desde los fundamentos:

* MOSFETs: Entender cómo funcionan como interruptores controlados por voltaje, qué es el voltaje de umbral (gate threshold), y cómo elegir el correcto para que un microcontrolador de 5V pudiera activarlo sin problemas.
* Transistores NPN: Estudiar su funcionamiento como interruptores y, crucialmente, como "level shifters" para poder comunicar de forma segura la señal de encendido o apagado

Este conocimiento teórico fue el puente que nos permitió pasar de un simple cableado a un diseño electrónico robusto y fiable.

El afinamiento del software fue igualmente desafiante:

* Tiempos de Espera (delays): Determinar los retardos necesarios para la estabilización de los componentes fue un proceso de mucha prueba y error. Un delay muy corto resultaba en lecturas incorrectas; uno muy largo desperdiciaba milisegundos preciosos de batería en cada ciclo. Experimentamos hasta encontrar el punto óptimo entre fiabilidad y eficiencia.
* Documentación del Microcontrolador: Tuvimos que sumergirnos en las hojas de datos (datasheets) del microcontrolador ATmega328P. Allí descubrimos una limitación clave del modo de sueño profundo: el temporizador interno (Watchdog Timer) solo puede configurarse para un máximo de 8 segundos de sueño. Para lograr pausas más largas (como 1 o 2 minutos), tuvimos que diseñar un bucle en nuestro código que "despierta" brevemente cada 8 segundos solo para volver a dormirse, hasta completar el tiempo total deseado.
* Documentación del XBee: El modo de hibernación del XBee (Pin Sleep) requirió otra lectura intensiva de su documentación. Primero, tuvimos que reconfigurar los módulos como "End Device" usando el software XCTU para habilitar estas funciones de ahorro. Segundo, descubrimos que no se trataba de un solo pin, sino de un sistema de dos:
* SLEEP\_RQ (Sleep Request): Un pin de salida desde nuestro Arduino para solicitar al XBee que se duerma.
* ON/SLEEP (Status Indicator): Un pin de entrada hacia nuestro Arduino que el XBee usa para confirmar su estado actual (si está despierto o dormido).

Esta última revelación fue clave. Nuestra lógica final no solo le ordena al XBee que se duerma, sino que también corrobora que haya obedecido antes de proceder. Esta comunicación de dos vías (comando y confirmación) previene fallos críticos, como que el microcontrolador se duerma mientras el radio sigue encendido consumiendo energía. Fue esta atención al detalle, nacida de la necesidad y la investigación, lo que finalmente nos permitió construir un sistema verdaderamente eficiente

## Implementando múltiples tecnologías y avances

Sabíamos que para que nuestro proyecto fuera completo, no bastaba con probar y optimizar un sistema basado únicamente en la tecnología XBee. Para entender verdaderamente el impacto de nuestras mejoras, planteamos la necesidad de experimentar con diversos módulos de radio. El objetivo es realizar un análisis comparativo que nos permita conocer en profundidad las características de cada tecnología y el impacto de nuestras mejoras energéticas sobre ellos

Para ello implementaremos diferentes tecnologías de comunicación por radiofrecuencia y sobre estos prototipos aplicaremos nuestras mejoras energéticas diseñadas

### LoRa

Ahora pasamos a probar con otros módulos de radio de diferentes tecnologías para así experimentar y validar su alcance, consumo y eficiencia. Como mencionamos, es importante entender estas características para elegir el componente adecuado para cada aplicación. Decidimos que realizaríamos un programa de envío básico con nuestros sensores para establecer una comunicación funcional.

Para empezar, decidimos utilizar la tecnología LoRa. Esto requirió una fase de investigación inicial para comprender sus particularidades. Tuvimos que aprender sobre las bandas de frecuencia permitidas en nuestra región, y entender el significado de parámetros clave como el Factor de Propagación (Spreading Factor), el Ancho de Banda (Bandwidth) y la Tasa de Codificación (Coding Rate), ya que estos afectan directamente el alcance y la velocidad de transmisión.

Durante este proceso, nos encontramos con el primer reto de hardware: los módulos LoRa operan a un nivel lógico de 3.3 voltios, mientras que nuestro Arduino Nano trabaja a 5 voltios. Para asegurar una comunicación segura sin dañar el módulo de radio, fue necesario adquirir conversores de nivel lógico, conocidos como level shifters. Tuvimos que soldar estos pequeños componentes a los módulos, como se muestra en la FIGURA N. A nivel de software, descubrimos que la comunicación con el módulo se realizaba a través del protocolo SPI. Afortunadamente, nuestra experiencia previa configurando la tarjeta SD, que también usa SPI, nos ayudó a comprender la lógica de pines y a escribir rápidamente un sketch básico de comunicación.

Logramos establecer un enlace casi de inmediato; el nodo receptor nos confirmaba la llegada de paquetes y nos mostraba la intensidad de la señal (RSSI). Sin embargo, nos topamos con un problema frustrante: no podíamos leer el contenido del paquete, que aparecía en blanco. Después de mucha depuración, descubrimos que el error estaba en nuestra función de lectura. La librería de LoRa entregaba los datos como una secuencia de bytes, pero nuestra función inicial no añadía correctamente el carácter nulo de terminación al final del buffer antes de convertirlo a un String. Esto provocaba que al intentar imprimir el contenido, el programa no supiera dónde terminaba el mensaje. Una vez corregido esto, pudimos recibir los mensajes completos, como se ve en la FIGURA N2.

### Inyectando los radios

En teoría, el siguiente paso era adaptar nuestro sketch complejo con sensores y librerías para que funcionara con LoRa, y luego repetir todo el proceso que hicimos para XBee y que se planeaba para cualquier otro módulo futuro. En ese momento nos dimos cuenta de que este enfoque no era escalable. Crear un código completo y personalizado para cada tipo de radio sería un proceso muy largo, propenso a errores y difícil de mantener.

Decidimos poner en marcha una solución de software más elegante y eficiente. En lugar de tener múltiples códigos, crearíamos un único sketch "universal" y una forma de "inyectar" el módulo de radio necesario. Tras investigar patrones de diseño de software, y usando nuestro conocimiento de programación, decidimos crear una "interfaz". Esta interfaz actúa como un contrato o una plantilla. Define un conjunto de funciones genéricas que cualquier módulo de radio debe tener, como "iniciar", "enviar" y "leer".

Luego, creamos clases específicas para cada radio, como "LoraRadio" y "XBeeRadio". Cada una de estas clases hereda de la interfaz y contiene el código de bajo nivel y los comandos específicos para su hardware particular. Sin embargo, al llamarlas desde nuestro programa principal, usamos las funciones genéricas de la interfaz. De esta forma, el código principal no necesita saber si está hablando con un LoRa o un XBee; simplemente le dice a la interfaz "envía estos datos".

El cambio es evidente al comparar el código anterior, "Normal\_sensor.ino", con el nuevo "EmisorUniversal.ino". En el código antiguo, todo el programa estaba lleno de llamadas directas a "xbeeSerial", acoplando fuertemente la lógica a un solo tipo de hardware. En el emisor universal, el código del loop es completamente agnóstico al radio. Utiliza un puntero genérico llamado "radio" y funciones como "radio->enviar()".

La magia ocurre en la sección de configuración. Con una sola línea de código, podemos decidir qué radio usar. Al definir "USE\_LORA", el programa crea un objeto de la clase "LoraRadio", lo configura con todos los parámetros necesarios a través de una estructura "LoRaConfig", y lo asigna al puntero genérico "radio". Si quisiéramos cambiar a XBee, simplemente cambiaríamos la definición a "USE\_XBEE" y el programa crearía un objeto "XBeeRadio" en su lugar. Esta técnica, conocida como inyección de dependencias, nos permitió crear un sistema modular, limpio y fácilmente extensible para cualquier tecnología de radio que queramos probar en el futuro.

El resultado fue la creación de un conjunto de archivos de código limpios y modulares, tal como se muestra en la estructura de la IMAGENARCHIVOSINYECTABLES, donde cada radio tiene su propia clase y un sketch universal los controla.

El proceso de implementación fue notablemente rápido, ya que el trabajo más pesado ya estaba hecho. Contábamos con prototipos funcionales tanto para XBee como para LoRa, el hardware estaba montado y sabíamos que los componentes se comunicaban. Por lo tanto, la tarea se centró exclusivamente en una reestructuración del software.

Creamos los sketches EmisorUniversal.ino y ReceptorUniversal.ino. Durante el desarrollo, nos encontramos con un problema interesante de inicialización. Nuestro objetivo era que toda la configuración ocurriera dentro de la función iniciar() de cada clase de radio. Esto funcionó perfectamente para LoRa, pero descubrimos que para XBee, que depende de SoftwareSerial, era más estable y seguro iniciar el puerto serial en el setup() principal del sketch, antes de llamar a la función de la interfaz. La solución fue documentar este requisito y adaptar ligeramente el sketch universal, demostrando cómo a veces la arquitectura ideal debe ajustarse a las particularidades del hardware.

Una vez resuelto esto, las pruebas fueron un éxito rotundo y demostraron el poder de este nuevo enfoque:

* Prueba con XBee: En los sketches de emisor y receptor, activamos la línea de código para usar XBee. Al compilar y cargar el software en nuestros prototipos ya existentes, la comunicación se estableció de inmediato, funcionando a la perfección.
* Prueba con LoRa: Sin cambiar nada en el hardware del prototipo usando antes con LoRa, volvimos a los archivos de código y modificamos una sola línea en cada uno para activar Lora en lugar de XBee como se puede ver en la Figura 10. Tras compilar y cargar, los nodos LoRa también se comunicaron sin ningún problema como se puede apreciar en la Figura 11 .

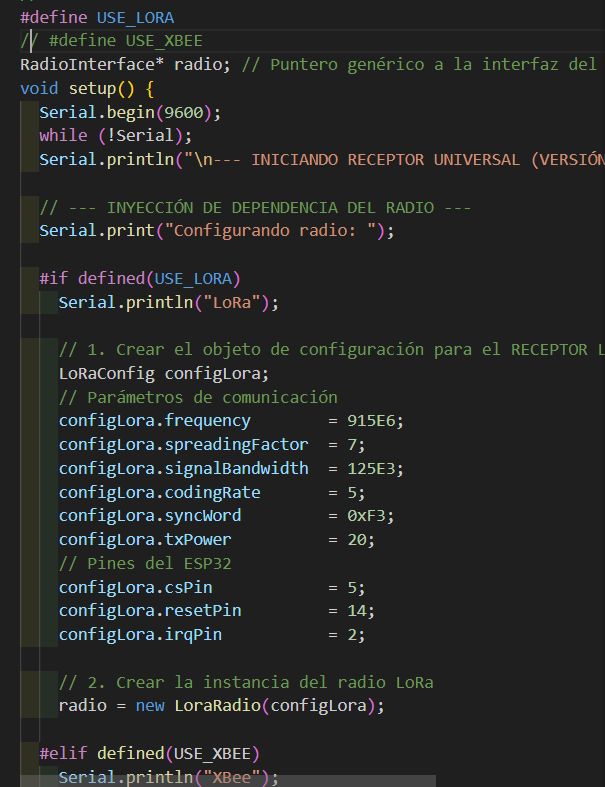


Figura 10. Cambio entre Xbee y Lora a través de una sola linea

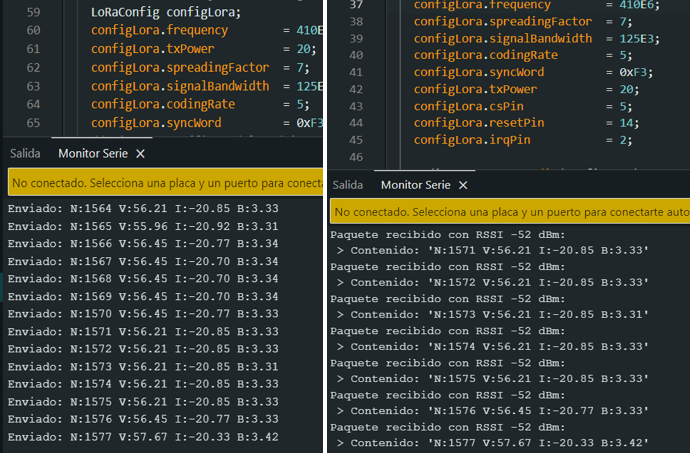


Figura 11. Comunicación de LoRa al inyectarse los módulos de radio

Ver que podíamos cambiar fundamentalmente la tecnología de comunicación inalámbrica con una modificación mínima en el código fue la confirmación de que nuestra estrategia había sido un éxito.

Esta arquitectura de software representa una mejora fundamental para nuestro proyecto por varias razones estratégicas:

* Eficiencia de Desarrollo: Ya no necesitamos escribir y mantener múltiples códigos complejos para cada tipo de radio. Ahora tenemos una única base de código para la aplicación (leer sensores, aplicar lógica) y solo tenemos que mantener las pequeñas clases de controlador para cada radio. Esto ahorra una cantidad enorme de tiempo y reduce la posibilidad de errores.
* Escalabilidad Futura: Como en el futuro queremos probar un tercer tipo de radio (como un nRF24L01), no necesitamos empezar de cero. Simplemente escribiremos una nueva clase que cumpla con el "contrato" de la RadioInterface y nuestro sistema universal será compatible con ella al instante.
* Código Desacoplado y Limpio: La lógica de la aplicación está ahora completamente separada de la capa de comunicación. Esto es un principio de ingeniería de software muy importante que hace que el código sea más fácil de leer, depurar y mejorar.
* Pruebas Aisladas: Nos permite probar cada parte del sistema de forma independiente. Podemos verificar que la lógica de los sensores funciona sin necesidad de un radio, o probar un nuevo radio con un simple mensaje de "hola mundo" sin necesidad de toda la lógica de los sensores.

En resumen, esta inversión en la arquitectura del software nos ha proporcionado un sistema robusto, flexible y preparado para el futuro, permitiéndonos centrarnos en lo que realmente importa: experimentar y comparar tecnologías en lugar de reescribir código repetitivo.

### Creacion de UniversalRadioWSN como librería

A medida que nuestra arquitectura de software se volvía más sofisticada, nos encontramos con una frustración persistente y limitante del propio entorno de desarrollo de Arduino. El IDE tenía problemas para reconocer nuestros archivos de cabecera (`.h`) cuando intentábamos organizarlos en subcarpetas o apuntar a direcciones relativas. Esto nos obligaba a mantener todos los archivos en una única carpeta junto al sketch principal, lo que contradecía nuestro objetivo de tener un código limpio, modular y organizado. A medida que añadíamos más clases, el directorio del proyecto se volvía cada vez más desordenado.

Decidimos que la única solución real era dar el siguiente paso y empaquetar todo nuestro sistema de radio universal en una librería de Arduino formal. Esto no solo resolvería los problemas de organización, sino que también haría nuestro código mucho más reutilizable y profesional.

El proceso implicó reestructurar por completo nuestros archivos. Aprendimos que una librería de Arduino sigue una estructura de carpetas estricta que el IDE sí sabe cómo interpretar. Creamos una carpeta principal para la librería y, dentro de ella, un subdirectorio llamado `src` (source) donde movimos todos nuestros archivos de bajo nivel: la interfaz `RadioInterface.h` y las clases concretas como `LoraRadio.h` y `XBeeRadio.h`.

Este cambio nos permitió simplemente incluir nuestra librería en cualquier sketch con una sola línea de código, como `#include <UniversalRadioWSN.h>`, y el IDE automáticamente encontraría y compilaría todos los archivos necesarios desde la carpeta `src` esto se puede apreciar más fácil en la Figura 12.

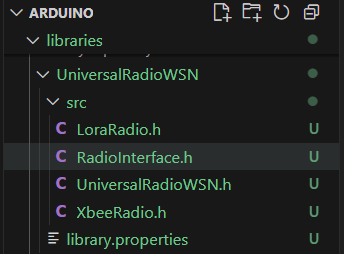


Figura 12. Organización de UniversalRadioWSN como librería

Esto resolvió de raíz el problema de la gestión de archivos, además, para que la librería fuera completa y fácil de usar en el futuro, movimos nuestros sketches de aplicación, como `EmisorUniversal.ino` y `ReceptorUniversal.ino`, a una carpeta llamada `examples`. De esta manera, cualquiera que instale la librería puede abrir estos ejemplos directamente desde el menú del IDE de Arduino para ver un caso de uso funcional y completo. Esta práctica no solo sirve como una excelente documentación, sino que también asegura que tengamos una base de código estable y probada para iniciar cualquier nuevo proyecto que utilice nuestro sistema de radio.

## Realización de pruebas

Para saber si las mejoras que implementamos realmente servían, fue necesario realizar mediciones objetivas sobre los nodos sensores. Nos centramos en dos pautas fundamentales que definen el rendimiento de cualquier red de sensores inalámbricos: la eficiencia energética y la fiabilidad de la comunicación.

### Análisis de las pruebas de Eficiencia Energética

Tras investigar las buenas prácticas para este tipo de análisis, llegamos a una conclusión clave: intentar medir la duración de la batería en tiempo real, simplemente esperando a que el nodo se apagara, era inviable. Debido a las optimizaciones de energía, una prueba así podría durar semanas, haciendo imposible realizar comparaciones rápidas y suficientes.

Por lo tanto, diseñamos una metodología de experimentación que nos permite obtener resultados precisos y rápidos: en lugar de una prueba de resistencia, adoptamos un enfoque de cálculo basado en mediciones precisas del ciclo de trabajo del nodo.

1. Medición de Corrientes Aisladas: Creamos dos sketches de prueba temporales para cada prototipo. El primero fuerza al nodo a operar continuamente en estado de trabajo (leyendo sensores, procesando y transmitiendo), lo que nos permite medir una corriente activa (I\_activa) estable. El segundo sketch fuerza al nodo a permanecer en estado de reposo (ya sea inactivo o en sueño profundo dependiendo el código que estemos evaluando), dándonos una lectura estable de la corriente de reposo (I\_reposo).
2. Medición de Tiempos: Con el código original, usamos la función millis() para medir con precisión la duración del estado de trabajo (T\_activo). El tiempo de reposo (T\_reposo) ya lo conocemos, pues es un valor que programamos en el código.
3. Cálculo de la Autonomía: Con estos cuatro valores, calculamos la corriente promedio que el nodo consume en cada ciclo completo: I\_promedio = ((I\_activa × T\_activo) + (I\_reposo × T\_reposo)) / (T\_activo + T\_reposo)

Finalmente, para estimar la vida útil del nodo, dividimos la capacidad total de nuestra batería (en miliamperios-hora, mAh) entre esta corriente promedio: Autonomía (horas) = Capacidad de la Batería (mAh) / I\_promedio (mA)

Este método nos permite predecir la autonomía de manera científica y, lo más importante, cuantificar rápidamente el impacto de cualquier cambio en el software o hardware.

### Análisis de las pruebas de Fiabilidad de la Comunicación

Para medir la confianza en la recepción de paquetes, implementamos un sistema de seguimiento simple pero robusto.

Contadores Sincronizados: En el software del nodo sensor, integramos un contador que se incrementa con cada paquete que se envía. Paralelamente, en el nodo coordinador, otro contador se incrementa con cada paquete válido que se recibe. Para asegurar que los datos no se pierdan en caso de un reinicio, estos contadores se guardan en la memoria no volátil (EEPROM) de cada Arduino.

La decisión de usar la EEPROM, sin embargo, trajo consigo su propia curva de aprendizaje. No se trataba simplemente de añadir una línea de código. Durante la investigación, descubrimos una característica crítica de esta memoria: tiene un número limitado de ciclos de escritura, alrededor de 100,000 por celda. Esta limitación nos obligó a ser muy cuidadosos. Para la validación de nuestros experimentos, decidimos actualizar el contador en la EEPROM con cada paquete para garantizar la máxima integridad de los datos y no perder la cuenta en caso de un reinicio, asumiendo el desgaste del chip como un factor controlado dentro de la prueba. Además, fue necesario aprender a gestionar el almacenamiento de datos de varios bytes, como nuestro contador, utilizando las funciones adecuadas para dividir y reconstruir la información correctamente en la memoria.

### AÑADIR QUE SE HIZO PARA TODOS LOS SKETCH

Cálculo de Paquetes Perdidos: Después de dejar el sistema funcionando durante un periodo de prueba determinado (por ejemplo, 12 horas), leemos el valor final de ambos contadores. La diferencia entre los paquetes enviados y los recibidos nos da el número exacto de paquetes que se perdieron en el camino.

Paquetes Perdidos = Contador de Enviados - Contador de Recibidos

Cálculo de la Tasa de Éxito: Con esos datos, calculamos el porcentaje de fiabilidad de la comunicación, una métrica clave para validar la calidad del enlace inalámbrico.

Tasa de Éxito (%) = (Paquetes Recibidos / Paquetes Enviados) × 100

Con esta doble estrategia de medición, podemos generar datos duros y objetivos que no solo validan nuestro trabajo, sino que también nos permiten entender profundamente las ventajas y desventajas de cada etapa de optimización.

### Pruebas de la Eficiencia energética

Para las pruebas de consumo energético, la herramienta principal fue el multímetro digital. Con el fin de asegurar la confiabilidad y precisión de los datos obtenidos, implementamos una práctica de validación cruzada: las mediciones clave se realizaron utilizando dos multímetros simultáneamente. Este método nos permitió corroborar que las lecturas de ambos instrumentos fueran consistentes, descartando así posibles errores debidos a la calibración o a un mal funcionamiento de uno de los equipos.

La técnica de medición consistió en configurar los multímetros para medir corriente, en la escala de amperios, y conectarlos en serie al circuito de alimentación del nodo. De esta manera, toda la corriente que consume el prototipo se ve obligada a pasar a través del instrumento, permitiéndonos registrar el consumo exacto en un momento dado. Este procedimiento requirió de nuestras habilidades y juicio técnico, ya que una conexión incorrecta, como colocar el amperímetro en paralelo, provocaría un cortocircuito que podría dañar el fusible del multímetro o el propio circuito. Fue fundamental verificar siempre la correcta configuración antes de energizar el sistema.

En la Figura 6 se puede apreciar una de las mediciones que realizamos, donde se registra el consumo del nodo durante la prueba de estado alto sostenido.

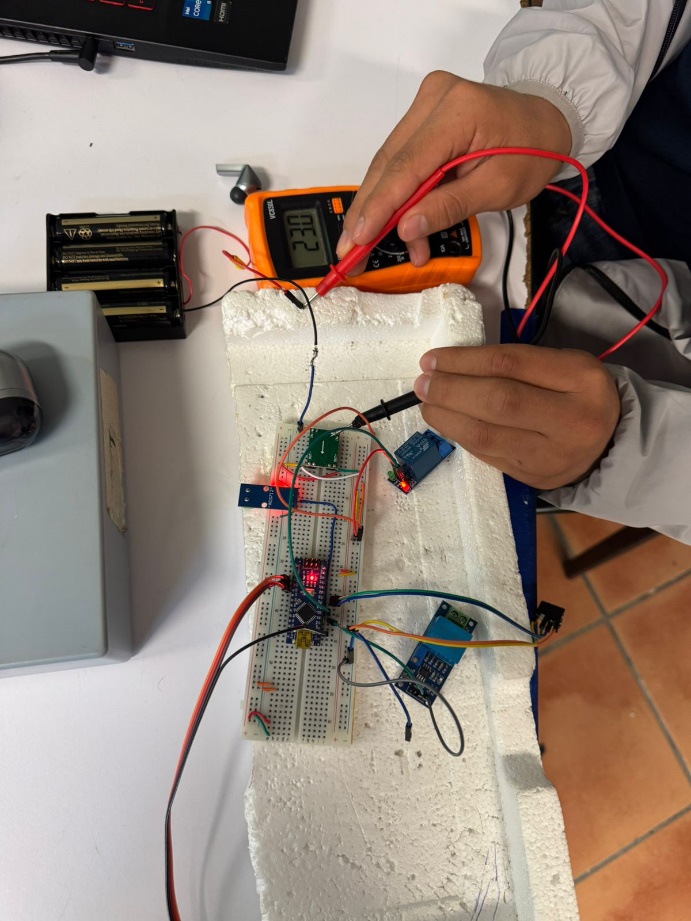


Figura 13. Medición de consumo en estado de trabajo en el nodo sensor sin modificar

A su vez, fue igualmente necesario medir el consumo del nodo en su estado de más bajo consumo. Para aislar esta medida, el procedimiento fue el mismo que para el estado activo: sin realizar ninguna modificación al hardware, simplemente cargamos en el microcontrolador un sketch temporal cuyo único propósito era poner el sistema en su estado de reposo inmediatamente después de arrancar.

Como se puede ver en la Figura 7, el resultado fue revelador y la diferencia en el consumo fue considerable. La corriente se desplomó desde lo que veíamos en el estado de trabajo, hasta casi la mitad. Estas mediciones preliminares, que realizamos como práctica para validar nuestra metodología, nos dieron una gran esperanza. Comprobar una caída tan significativa en el consumo del hardware base nos dio la confianza de que los prototipos optimizados, que combinan esta capacidad de reposo con una lógica de trabajo más eficiente, mostrarían una mejora aún mayor en la autonomía final.

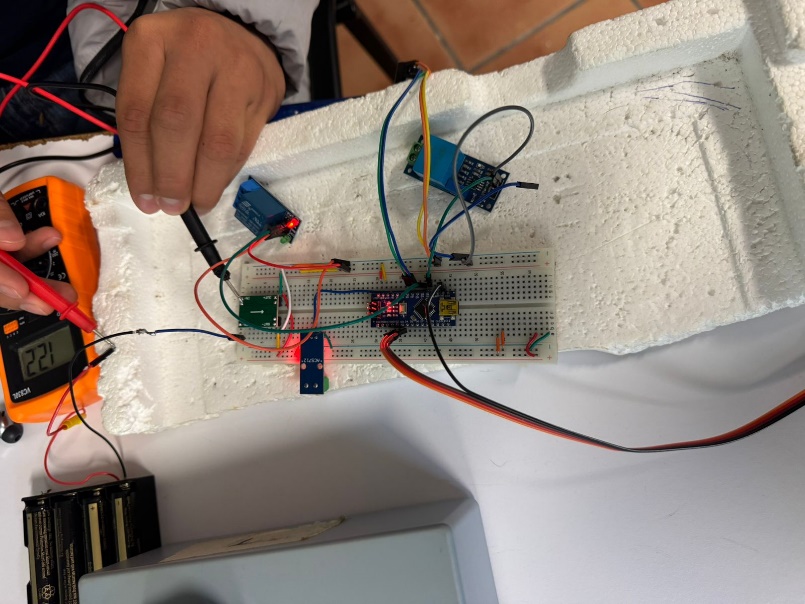


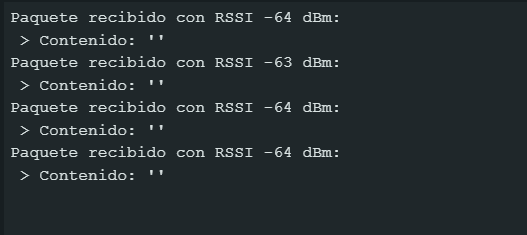
Figura 14. Medición de consumo en estado de delay en el nodo sensor

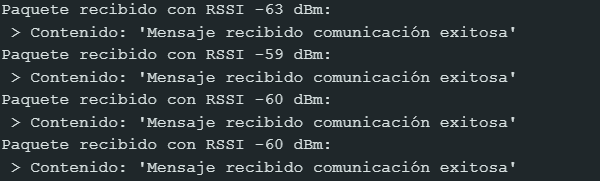
### PRUEBAS DE CONFAIZNA DE OMAR

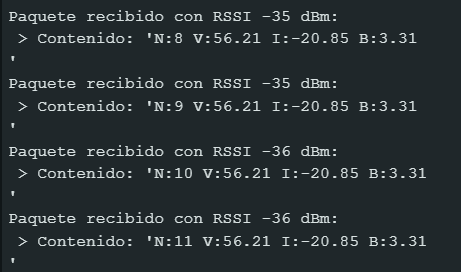
# Resto para añadir

Ahora pasamos a probar con otros módulos de radio de diferentes tecnologías para así experimentarlos probar su alcance, consumo, eficiencia, etc que como mencionamos es importante entender y conocer sus datos de esos que dijimos, lo que haremos es realizar igual un programa de envío básico de lo de nuestros sensores y hacer que se comuniquen , para empezar decidimos utilizr lora para ello investigamos (aquí añade q cosas sn necesarias saber) y entre eso supimos que se necesita un level shifter x los 3.3v y tuvimos que comrparlos soldarlos (como en la FIGURA N (ESTO DEJALO ASI PARA QUE YO RECUERDE AÑADIRLA)) además que tuvimos que aprender como funcionan y vimos que usa spi y el conocimiento de preubas previas con el sd nos ayudó a fácil hacer un sketch básico y logramos comunicarnos tal que recibíamos mensajes de confirmación de llegada pero no podíamos leer el paquete se veía en blanco (mensaje para la ia, se veía así Paquete recibido con RSSI -64 dBm:

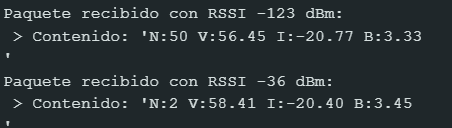
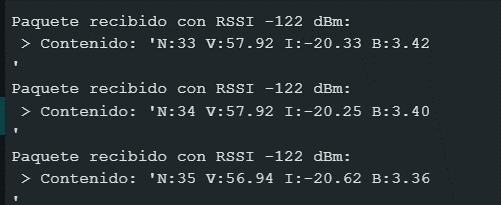
> Contenido: '')) pero hicios cambios y pudimos recibir el mensaje como sv e en la FIGURA N2 (aquí inventa tú algo de por qué no podkíamos leer el mensaje algo de lbirerías o algo);; ahora seguía en teoría pasar a implementar ocn nuestros sensores y luego empezar a poner librerías a cada sketch pero pensamos esto va a ser muy largo crear un sketch para cada cosa entonces decidimos poner en marcha otra forma que es crear una forma de inyectar el modulo del radio para que si necesitamos usar otro solo cambiamos nuestro objeto de configuraciones en lugar de crear otro código lo que es pues inyectar solo una línea de código para cambiar la radio que corresponde etnocnes nos pusimos manos a la obra a investigar como se podía y luego de muchas vueltas usamos nuestro conocimiento de porgramacion para crear una interface que consume clases especificas según el radio y que implementan sus propias funciones asi se llama solo a las mismas funciones y solo hay que decidir que radio es entonces en cada clase es que se implementan las funciones que van casdas con el radio pero al llamarlas se llamarán igual al llamar a la interface







CAMBIO EN LA FRECUENCIA DE 915 A 410 DEBIDO AL MODELO DEL MÓDULO DE RADIO

AUMENTÓ LA POTENCIA DE LA SEÑAL

Implementacion de paneles solares:

metodología experimental paso a paso:

### 1. El Montaje Experimental (El "Laboratorio")

Para cada experimento, necesitarás un montaje consistente que te permita medir las variables clave. La idea es registrar simultáneamente la energía que *entra* al sistema, la que *consume*, y el *estado* del almacenamiento.

**Componentes por Montaje de Prueba:**

* **Tu Nodo Sensor:** Con el Arduino Nano, los sensores y el módulo de radio.
* **Fuente de Energía:** Uno de tus paneles solares conectado al sistema de carga de la batería/supercondensador.
* **Almacenamiento:** La batería o supercondensador que planeas usar.
* **Instrumentos de Medición:**
  + **Energía Generada:** Sensores de voltaje y corriente (como los que ya usas, o un INA219) colocados *entre* el panel solar y el circuito de carga para medir la potencia de entrada.
  + **Consumo del Nodo:** Sensores de voltaje y corriente para medir la energía que consume tu nodo.
  + **Estado de la Batería:** Una simple lectura del voltaje de la batería es un buen indicador de su estado de carga.
  + **Condiciones Ambientales:** Esto es crucial. Necesitas medir la cantidad de sol disponible. La opción profesional es un **piranómetro**. 1 Una alternativa de bajo costo es usar un

**sensor de luz (LDR)** calibrado para dar una lectura aproximada de la irradiancia (por ejemplo, en Lux o una escala relativa). También mide la temperatura ambiente.

**Recolección de Datos:** La mejor manera de hacerlo es automatizar el registro. Conecta el Arduino que mide todo esto a un ordenador vía USB.

* **Software:** Puedes usar el macro de Excel **PLX-DAQ** o un script simple en Python para leer los datos del puerto serie y guardarlos en un archivo CSV con una marca de tiempo cada pocos segundos o minutos.
* **Datos a Registrar:** Por cada marca de tiempo, guarda: Timestamp, Voltaje\_Panel, Corriente\_Panel, Potencia\_Panel, Voltaje\_Bateria, Consumo\_Nodo, Irradiancia\_Solar, Temperatura.

### 2. Diseño de los Experimentos

Con tu montaje listo, puedes realizar una serie de pruebas enfocadas.

#### Experimento 1: Caracterización y Comparación de Paneles

**Objetivo:** Determinar cuál de tus dos tipos de panel es más eficiente en tus condiciones locales. **Procedimiento:**

1. Usa un nodo sensor con una carga de consumo constante y predecible (puedes programarlo para que no entre en modo de bajo consumo durante esta prueba).
2. Expón el **Panel Tipo A** a la luz solar directa y registra durante varias horas (idealmente un día completo) todos los parámetros.
3. Al día siguiente, en condiciones lo más similares posible, repite el experimento exacto pero con el **Panel Tipo B**.
4. Repite el proceso con los otros dos paneles para confirmar los resultados y asegurar que no haya un panel defectuoso.

**Resultado Esperado:** Gráficas que comparen la Potencia\_Panel generada por cada tipo de panel en función de la Irradiancia\_Solar. Esto te permitirá decir con datos: "El panel Tipo B genera, en promedio, un 15% más de energía que el Tipo A bajo las mismas condiciones de sol".

#### Experimento 2: Prueba de Autonomía y Autosuficiencia

**Objetivo:** Demostrar que el nodo puede operar de forma indefinida utilizando el mejor panel del experimento anterior. **Procedimiento:**

1. Carga completamente la batería de tu nodo sensor.
2. Activa el código completo con todas tus optimizaciones (AdaptiveTXWSN, el modo de sueño profundo, el control de energía de los sensores, etc.).
3. Deja el sistema funcionando durante varios días seguidos (una semana sería ideal).
4. Registra continuamente todos los parámetros (energía generada, consumo, voltaje de batería, irradiancia).

**Resultado Esperado:** El gráfico más importante de tu proyecto. Será una gráfica de series temporales de varios días.

* **Eje X:** Tiempo (días/horas).
* **Eje Y (izquierdo):** Voltaje de la Batería (V).
* **Eje Y (derecho):** Irradiancia Solar (W/m² o Lux).

Lo que buscas demostrar es que, a pesar de las fluctuaciones diarias (la batería se descarga de noche), la tendencia general del voltaje de la batería se mantiene estable o incluso aumenta ligeramente a lo largo de los días. Si la línea de voltaje de la batería no tiene una tendencia a la baja después de una semana, habrás demostrado la autosuficiencia.

#### Experimento 3: Pruebas de Estrés (Condiciones Variables)

**Objetivo:** Evaluar cómo se comporta el sistema en condiciones no ideales. **Procedimiento:**

1. Realiza pruebas de autonomía más cortas (de 1 o 2 días) bajo diferentes escenarios.
2. **Día Nublado:** Deja el sistema funcionando en un día con poca luz para ver cuánto tiempo sobrevive y cómo tu librería AdaptiveTXWSN reduce la frecuencia de transmisión para ahorrar energía.
3. **Sombra Parcial:** Simula una obstrucción (como una hoja o un edificio) que cubra parte del panel durante unas horas y observa cómo se recupera el sistema.

**Resultado Esperado:** Gráficas que muestren la resiliencia del sistema. Por ejemplo, una gráfica que muestre cómo el intervalo de transmisión aumenta (la frecuencia disminuye) a medida que el voltaje de la batería baja durante un día nublado.

### 3. Cómo Presentar los Resultados

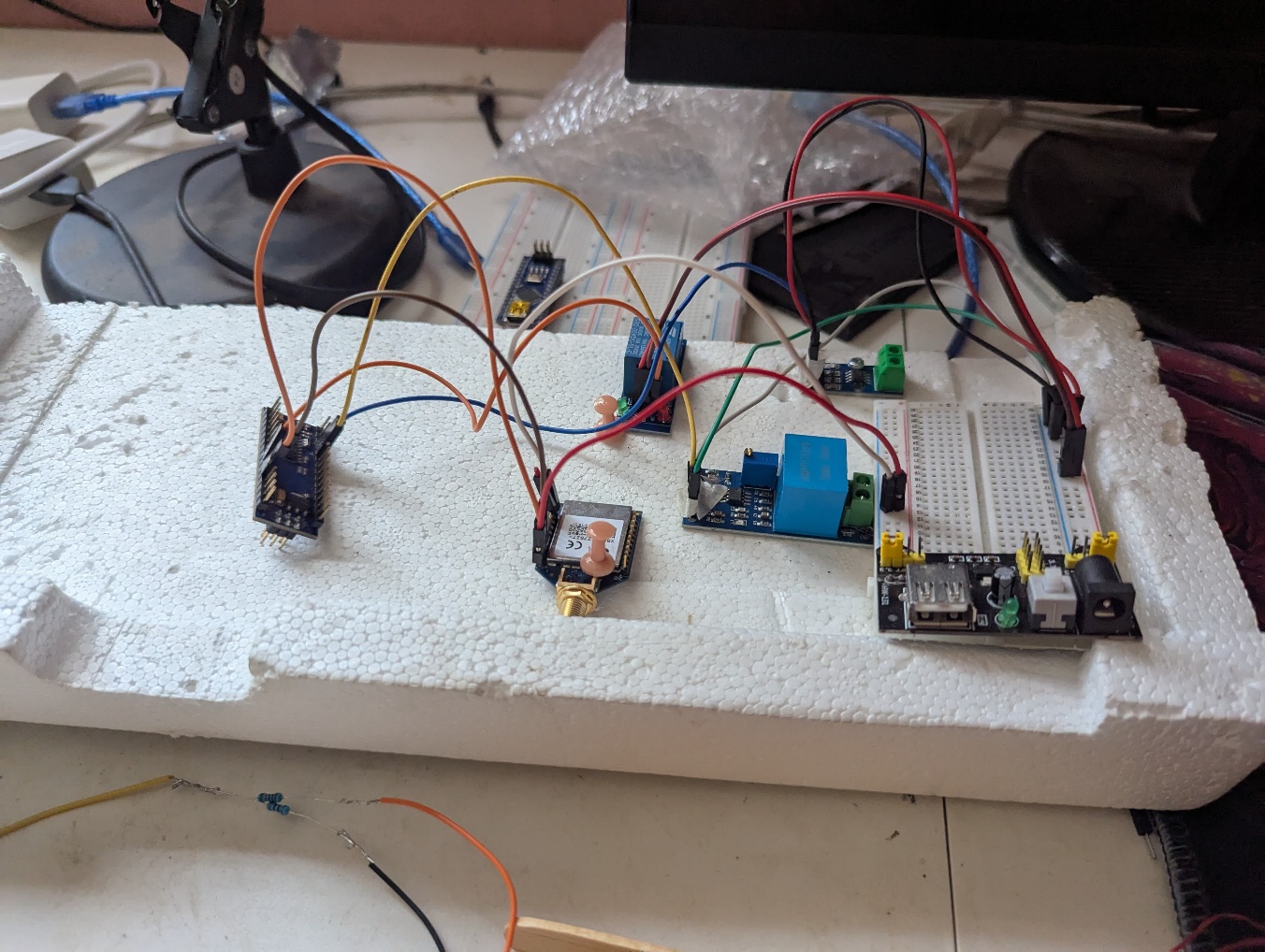
La clave es contar una historia con tus datos.

* **Tablas:** Usa tablas para resumir la configuración de tu hardware y los parámetros clave de tus librerías (ej. los umbrales de voltaje de AdaptiveTXWSN).
* **Gráficos Comparativos:** Para el Experimento 1, usa gráficos de barras o de líneas para comparar directamente la potencia generada por los paneles A y B.
* **Gráficos de Series Temporales:** Como se describió en el Experimento 2, este es tu gráfico principal. Asegúrate de que esté bien etiquetado y sea fácil de leer. La correlación visual entre la luz solar y el aumento del voltaje de la batería es muy poderosa.
* **Análisis de Consumo:** Crea gráficos de barras que comparen el consumo promedio del nodo en diferentes modos:
  + Modo "Normal" (sin ahorro de energía).
  + Modo "Optimizado" (con todas tus mejoras activadas).
  + Esto cuantificará el impacto de tu trabajo de software. Por ejemplo: "La implementación del ciclo de sueño profundo y el control de alimentación de los sensores redujo el consumo energético en un 85%".

**Un consejo adicional para darle más rigor científico:** Si tienes acceso a un multímetro de alta precisión, realiza una validación de tus sensores. Mide el consumo de corriente con el multímetro y compáralo con lo que reporta tu sensor INA219. Reportar el margen de error (ej. "nuestras mediciones a bordo tienen un error inferior al 5% comparado con un multímetro de laboratorio") le da una gran credibilidad a tus resultados.

En resumen: aprovecha tu limitación de paneles para hacer un análisis comparativo profundo, mide todo (entrada, salida y ambiente), y presenta tus resultados con gráficos claros que demuestren la relación causa-efecto entre el sol y la autonomía de tu nodo. ¡Tu proyecto está muy bien encaminado para producir resultados excelentes!

.



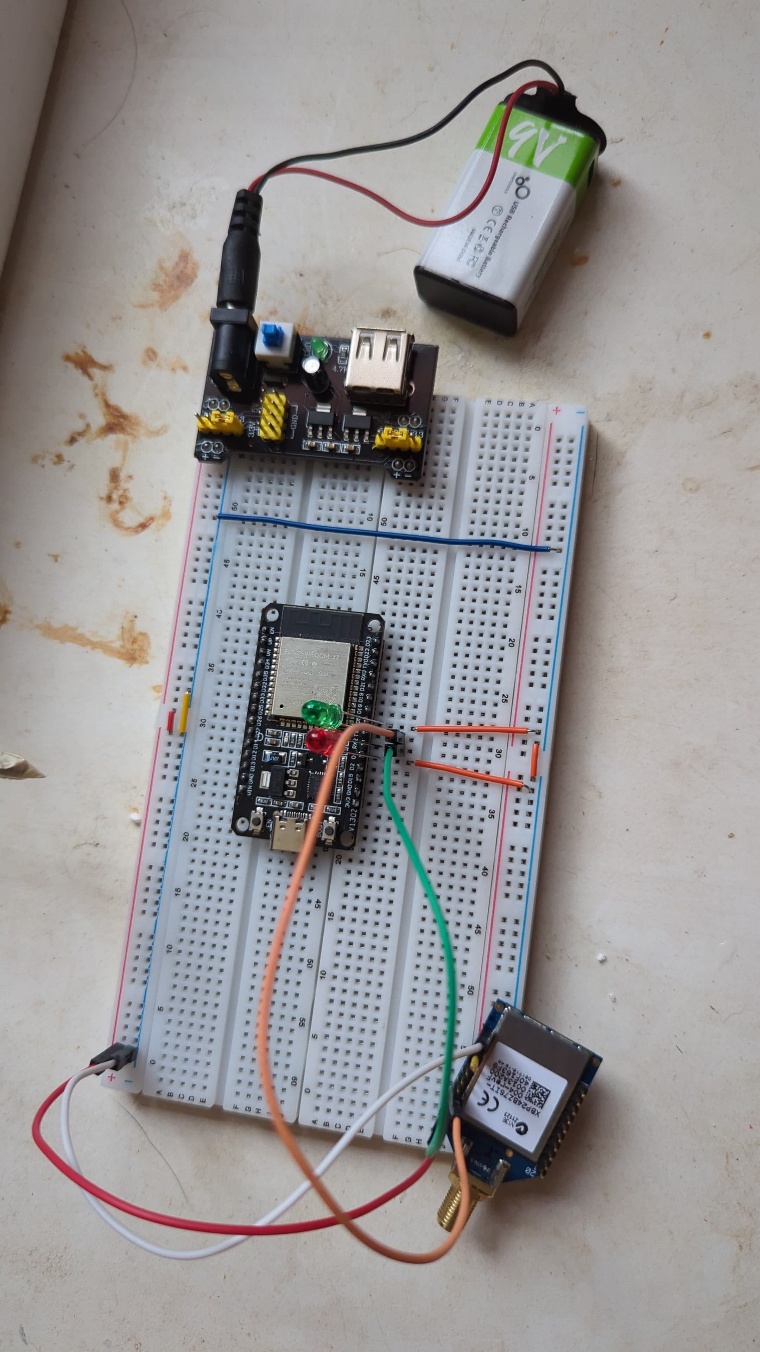


Imagen 2. Estructura Nodo Coordinador. Autor: Propio

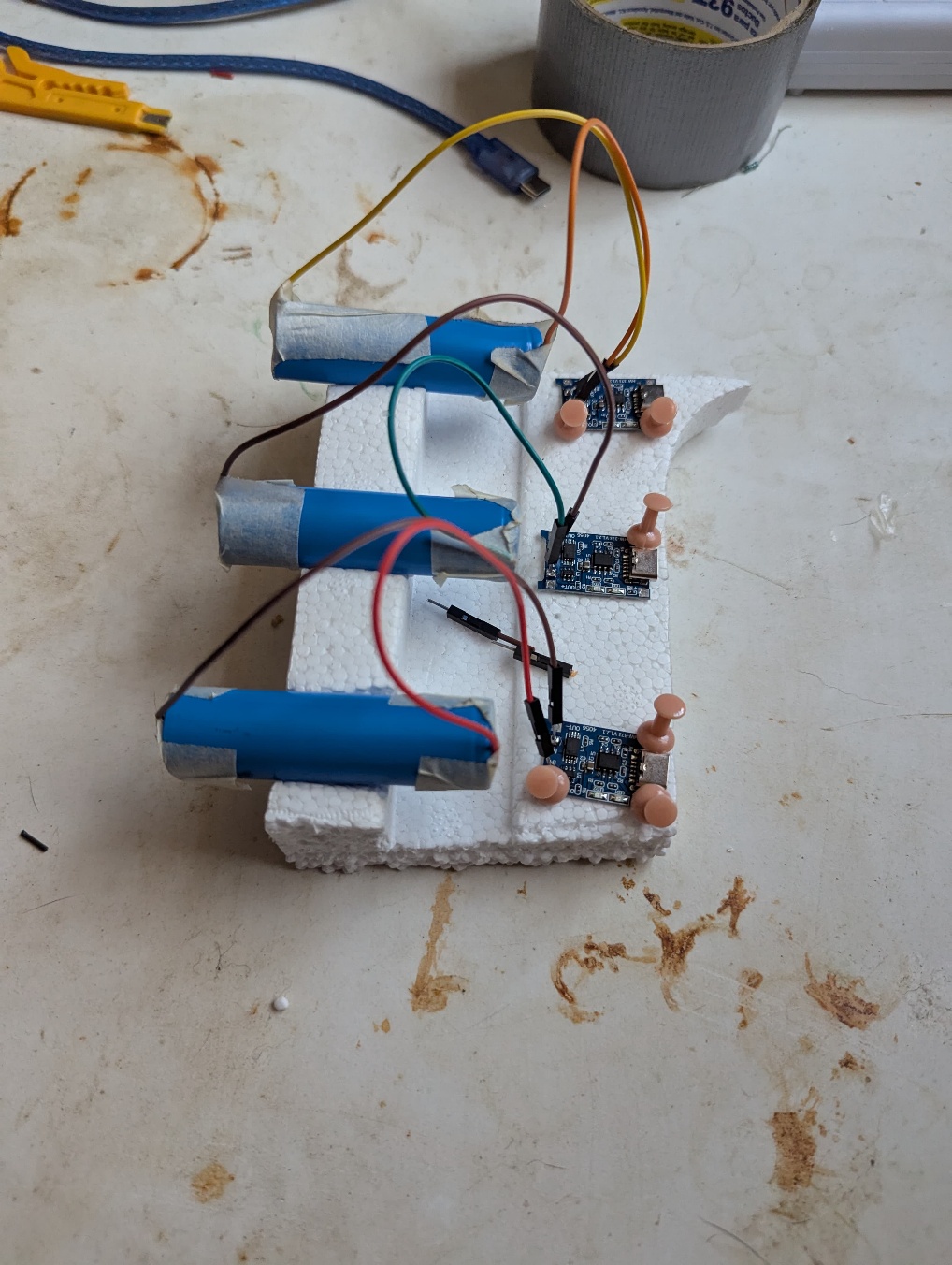
Imagen 3. Módulo de almacen de baterías. Autor: Propio 

Imagen 4. Modulo para cargar baterias. Autor: Propio