DEPARTAMENTO DE
INGENIERÍA ELÉCTRICA
UNIVERSIDAD DE CONCEPCIÓN

Desarrollo de un sistema de transmisión de datos a larga distancia utilizando tecnología LoRa para aplicaciones de internet de las cosas (IOT)

POR

Catalina Scarlett Barra Collinao

Memoria de Título presentada a la Facultad de Ingeniería de la Universidad de Concepción para optar al título profesional de Ingeniero Civil en Telecomunicaciones.

Profesor Guía
Sergio K. Sobarzo G., Ph.D.

Concepción,
5 de junio de 2025

Para mi querida familia y mi amado copito.

Resumen

El presente trabajo desarrolla un sistema de transmisión de datos a larga distancia basado en la tecnología LoRa, orientado a aplicaciones de Internet de las Cosas (IoT). LoRa es una tecnología de comunicación inalámbrica de bajo consumo energético y gran alcance, ideal para entornos donde las redes tradicionales no son viables debido a costos elevados o limitaciones de infraestructura.

El proyecto abarca desde la fundamentación teórica hasta la implementación practica del sistema, incluyendo el diseño del hardware y software, asi como la validación mediante pruebas de campo en distintos entornos. Se emplean placas LoRa ESP32 y una Raspberry Pi para establecer la comunicación entre dispositivos, optimizando la eficiencia energética y la estabilidad del enlace de datos.

El informe se estructura en cinco capítulos. En primer lugar, se introduce el contexto del proyecto, estableciendo los objetivos, alcances y limitaciones. Posteriormente, se presenta el marco teórico, donde se abordan los fundamentos de la tecnología LoRa y su comparación con otras soluciones de comunicación. Luego, en el capítulo de diseño e implementación, se detallan los criterios técnicos utilizados en la selección de componentes y el desarrollo del software. En el cuarto capítulo, se analizan los resultados obtenidos a partir de las pruebas de campo realizadas en diferentes escenarios. Finalmente, se presentan las conclusiones y recomendaciones, destacando los principales hallazgos y posibles mejoras para futuras implementaciones.

Este trabajo busca contribuir al desarrollo de soluciones de conectividad eficiente en el ámbito de IoT, demostrando el potencial de LoRa para la transmisión de datos en aplicaciones de monitoreo remoto, optimización de recursos y automatización de procesos en diversos sectores.

Abstract

This work develops a long-range data transmission system based on LoRa technology, aimed at Internet of Things (IoT) applications. LoRa is a low-power, long-range wireless communication technology, ideal for environments where traditional networks are not feasible due to high costs or infrastructure limitations.

The project spans from theoretical foundations to the practical implementation of the system, including hardware and software design, as well as validation through field testing in various environments. LoRa ESP32 boards and a Raspberry Pi are used to establish communication between devices, optimizing energy efficiency and data link stability.

The report is structured into five chapters. First, the project context is introduced, establishing the objectives, scope, and limitations. Next, the theoretical framework is presented, addressing the fundamentals of LoRa technology and its comparison with other communication solutions. Then, the design and implementation chapter details the technical criteria used in component selection and software development. In the fourth chapter, the results obtained from field tests conducted in different scenarios are analyzed. Finally, the conclusions and recommendations are presented, highlighting the main findings and possible improvements for future implementations.

This work aims to contribute to the development of efficient connectivity solutions in the IoT field, demonstrating the potential of LoRa for data transmission in remote monitoring, resource optimization, and process automation applications across various sectors.

Agradecimientos

Quiero agradecerle a mi familia por apoyarme durante todos estos años, en especial a mi madre Yanet, mi hermano Martin, mis abuelos María y Edgardo, alias Ilda y Lalito, y mis tíos. Les agradezco la confianza ciega que tuvieron en mi durante este difícil proceso y alentarme en cada momento complejo. Este logro no es solo mío, si no que de todos ellos.

Agradecerles a mis compañeros de carrera que se transformaron en mis buenos amigos, sin ellos la estadía en Concepción y en la Universidad no hubiese sido la misma, en especial a mi compañero, amigo y pareja, Dominic Darío, que estuvo a mi lado apoyándome y cuidándome en todo momento. Agradecerle a mi amiga Florencia que me escuchó mis llantos, mis alegrías y mis momentos de estrés. Me gustaría extender este agradecimiento a cada profesor de Telecomunicaciones que me alentó durante mi enseñanza universitaria y cada persona que fue parte de mi vida a lo largo de mi carrera.

Y, por último, pero no menos importante, agradecerle a mi copito, mi bebé perruno que madrugaba conmigo todas las noches de pandemia que pasaba estudiando, aunque fueran las 05:00 am se quedaba a mi lado, que cada vez que vuelvo a mi hogar se alegra como si fuera el día más feliz de su vida y espero que su corazoncito perruno sepa perdonar todas las veces que tuve que despedirme de el por viajar a Concepción.

Índice General

Índice de Figuras

Índice de Tablas

Siglas

ABP Activación por Personalización

ADR Tasa de Datos Adaptativa

AWGN Ruido Blanco Aditivo Gaussiano

bps bits por segundo

CDMA Acceso Múltiple por División de Código

CPU Unidad Central de Procesamiento

CRC Código de Redundancia Cíclica

CSMA Acceso Múltiple por Detección de Portadora

CSS Espectro Ensanchado por Chirp

dB decibelio

DoS Denegación de Servicio

DSSS Espectro Ensanchado de Secuencia Directa

ESP Extensión Serial Periférica

FDM Multiplexación por División de Frecuencia

FHSS Espectro Ensanchado por Salto de Frecuencia

Hz Hercios

IEEE Instituto de Ingenieros Eléctricos y Electrónicos

IP Protocolo de Internet

ISM Banda Industrial Científica y Médica

kbps kilobits por segundo

km kilómetro

LBT Detección de Ocupación de Canal

LLC Control de Enlace Lógico

LPWAN Red de Área Amplia de Baja Potencia

m metro

M2M Máquina a Máquina

MAC Control de Acceso al Medio

MFSK Modulación por Desplazamiento de Frecuencia Múltiple

ODS Objetivos de Desarrollo Sostenible

OLED Diodo Emisor de Luz Orgánico

ONU Organización de las Naciones Unidas

OSI Modelo de Interconexión de Sistemas Abiertos

OTAA Activación por el Aire

PPP Protocolo Punto a Punto

PRR Tasa de Recepción de Paquetes

QoS Calidad del Servicio

RSSI Indicador de Intensidad de Señal Recibida

SF Factor de Propagación

SNR Relación Señal-Ruido

SS Espectro Ensanchado

TMD Multiplexación por División de Tiempo

W Watts

WAN Red de Área Amplia

Capítulo 1

Introducción

En un mundo cada vez más conectado, la tecnología LoRa (Long Range) se presenta como una solución innovadora para la transmisión de datos a larga distancia en aplicaciones de internet de las cosas (IoT). No obstante, su implementación enfrenta desafíos importantes, especialmente en entornos donde la cobertura de red es limitada o la infraestructura de comunicación es costosa de mantener. Este problema es crítico para sectores como la agricultura, la monitorización ambiental y el transporte, donde los dispositivos IoT necesitan transmitir datos de manera eficiente y económica desde ubicaciones remotas.

El presente proyecto aborda este problema mediante el diseño y desarrollo de un sistema de transmisión de datos basado en tecnología LoRa, utilizando dos placas LoRa ESP32, y una Raspberry Pi. La propuesta busca demostrar que es posible mejorar la conectividad y reducir el consumo energético en aplicaciones IoT, superando las limitaciones actuales de cobertura y costo.

1.1. Trabajos previos

La revisión bibliográfica es una parte fundamental de este informe, ya que permite contextualizar y fundamentar el desarrollo del proyecto de habilitación profesional en base a los trabajos previos realizados en el área de redes LPWAN y específicamente en la tecnología LoRa. Esta revisión tiene como propósito identificar los avances, limitaciones

y desafíos actuales en el uso de LoRa para aplicaciones IoT, con el fin de establecer un marco teórico sólido que sustente la propuesta técnica del proyecto.

A continuación, se presenta una revisión cronológica de los principales trabajos relevantes en esta área, los demás están referenciados en la sección bibliografía del documento:

1. Primeras investigaciones sobre LPWAN y LoRa (2010-2014): La tecnología LoRa fue patentada por Semtech en 2014. Estudios iniciales, como el artículo "LoRaWAN - A Low Power WAN Protocol for Internet of Things: A Review and Opportunities"[1], destacaron la viabilidad de LoRa como tecnología base para redes IoT de bajo consumo y largo alcance. Estos trabajos definieron características fundamentales como el uso de bandas ISM y la capacidad de ajuste del factor de propagación (SF) para optimizar el alcance y la tasa de datos.

2. Expansión de LoRaWAN como protocolo estándar (2015-2017): Durante esta etapa, LoRaWAN comenzó a consolidarse como protocolo MAC para redes LPWAN. En el artículo .ª Survey of LoRaWAN for IoT: From Technology to Application"[2] se analizó la escalabilidad de LoRaWAN en aplicaciones urbanas, identificando desafíos como las colisiones de paquetes y la necesidad de mecanismos más eficientes para la gestión del espectro.

3. Aplicaciones IoT rurales y urbanas (2018-2020): Investigaciones como .ª Survey on LoRaWAN Technology: Recent Trends, Opportunities, Challenges and Solutions"[3] exploraron el uso de LoRa en escenarios rurales y urbanos. Estos estudios evidenciaron limitaciones de alcance en entornos urbanos densos debido a interferencias físicas, pero destacaron su viabilidad en aplicaciones agrícolas, como el monitoreo de humedad del suelo y el control de plagas. Además, se resaltaron las capacidades de LoRa para aplicaciones de ciudades inteligentes, como el monitoreo ambiental y la gestión de recursos urbanos.

4. Avances en eficiencia energética y seguridad (2021-2023): En años recientes, los esfuerzos se han centrado en la optimización del consumo energético y la seguridad en LoRaWAN. El artículo .ªnalysis of LoRa/LoRaWAN Challenges: A Review"[4] abordó mejorar en eficiencia energética mediante algoritmos de tasa de datos adaptativa (ADR), mientras que las actualizaciones del estándar LoRaWAN 1.1, publicadas por la LoRa Alliance, introdujeron medidas para mitigar ataques de

repetición y mejorar la gestión de claves de seguridad.

5. Documentos técnicos y estándares: Documento técnicos como "LoRaWAN® Specification v1.1"[5] y "LoRaWAN® 1.0.4 Specification Package"[6], ambos publicados por la LoRa Alliance, han sido fundamentales para establecer las bases del diseño y operación de redes LoRaWAN. Estas especificaciones incluyen lineamientos técnicos detallados para la implementación de dispositivos IoT, abordando aspectos como la seguridad, el manejo del espectro y la certificación de equipos.

1.1.1. Discusión

El estado del arte sobre LoRa y LoRaWAN muestra que estas tecnologías han logrado consolidarse como soluciones efectivas para redes IOT de bajo consumo energético y largo alcance. Entre las principales ventajas, se destacan su capacidad para operar en bandas no licenciadas, su flexibilidad para ajustar a diferentes conficiones mediante el uso de parámetros configurables como el factor de propagación, y su compatibilidad con aplicaciones en entornos rurales y urbanos. Estas características han permitido a LoRa abordar una amplia gama de aplicaciones, desde monitoreo ambiental en áreas remotas hasta sistemas de gestión en ciudades inteligentes.

Sin embargo, las investigaciones también han identificado limitaciones significativas. La tasa de datos es inherentemente baja, lo que restringe su aplicabilidad en escenarios que demandan transmisión de grandes volúmenes de información. En entornos urbanos densos, la interferencia de obstáculos físicos y la coexistencia con otras tecnologías en las bandas ISM limitan el alcance efectivo de las comunicaciones. Además, los estudios sobre redes densas evidencian problemas de escalabilidad, con pérdidas de paquetes que pueden alcanzar valores críticos cuando aumenta la cantidad de dispositivos conectados. En cuanto a la seguridad, aunque el estándar LoRaWAN 1.1 ha introducido mejoras importantes, como la mitigación de ataques de repetición y la gestión más robusta de claves, siguen existiendo riesgos relacionados con el uso de claves estáticas en dispositivos con configuraciones menos avanzadas.

Estos desafíos justifican la necesidad de continuar investigando en áreas clave como la optimización del consumo energético, la mejora de la seguridad y el diseño de soluciones

adaptadas a escenarios específicos.

1.2. Definición del problema

En la última década, el crecimiento exponencial de dispositivos conectados ha impulsado el desarrollo de nuevas soluciones para la transmisión de datos en aplicaciones de Internet de las Cosas (IoT). Sin embargo, muchas de estas aplicaciones se desarrollan en entornos donde el acceso a redes tradicionales, como Wi-Fi o redes móviles, resulta inviable por razones técnicas o económicas. Estas tecnologías, además de requerir infraestructura costosa, presentan un consumo energético elevado que limita su implementación en sistemas alimentados por baterías, como sensores remotos o estaciones de monitoreo autónomas.

Este escenario plantea un problema concreto, la falta de una solución de transmisión de datos eficiente, de bajo consumo y con cobertura adecuada para aplicaciones IoT en zonas con conectividad limitada. En particular, es necesario contar con sistemas que puedan operar en condiciones donde la infraestructura de red es escasa, la alimentación energética es restringida y se requiere una cobertura extensa.

La presente memoria aborda este problema mediante el diseño, implementación y validación de un sistema de transmisión de datos basado en tecnología LoRa, con el objetivo de demostrar su viabilidad como alternativa de bajo consumo energético y largo alcance para entornos urbanos y rurales con infraestructura limitada.

1.3. Objetivos

1.3.1. Objetivo general

Desarrollar e implementar un sistema de transmisión de datos a larga distancia utilizando tecnología LoRa, empleando dos placas LoRa ESP32, y una Raspberry Pi, para establecer una comunicación eficiente y de bajo consumo energético en aplicaciones de internet de las cosas (IoT), con el fin de mejorar la conectividad en entornos con cobertura de red limitada o costosa.

1.3.2. Objetivos específicos

En base al objetivo general dado anteriormente, los objetivos específicos serán:

1. Diseñar e implementar el hardware necesario para el sistema de comunicaciones LoRa, utilizando dos placas LoRa ESP32, y una Raspberry Pi.
2. Desarrollar el software de comunicación y protocolos de transmisión de datos para garantizar la eficiencia y confiabilidad en la comunicación entre los dispositivos LoRa.
3. Realizar pruebas de campo en diferentes entornos para evaluar la cobertura y eficiencia del sistema de transmisión de datos LoRa.
4. Optimizar el consumo energético del sistema mediante ajustes en la configuración de los parámetros de transmisión, para asegurar un equilibrio entre eficiencia energética y desempeño del sistema de transmisión.
5. Documentar los resultados obtenidos y elaborar recomendaciones para la implementación de sistemas de comunicaciones basados en LoRa en aplicaciones IoT.

1.4. Metodología

Este informe se compone de cinco capítulos, los cuales detallan el desarrollo e implementación de un sistema de transmisión de datos basado en tecnología LoRa para aplicaciones IoT.

El capítulo uno introduce el contexto y la motivación del proyecto, destacando la necesidad de soluciones de comunicación eficientes para IoT en entornos donde las redes tradicionales no son viables. Se plantea el problema a resolver, se establecen los objetivos generales y específicos del estudio y se delimitan los alcances y limitaciones del sistema propuesto.

El capítulo dos está dedicado al marco teórico, donde se explican los principios fundamentales de la tecnología LoRa y su aplicación en el ámbito del Internet de las Cosas. Se revisan conceptos clave como las redes LPWAN, los protocolos de

comunicación empleados y la eficiencia energética en la transmisión de datos, además de una comparación con otras tecnologías similares.

El capítulo tres aborda el diseño e implementación del sistema. Se detallan los criterios de selección de hardware, incluyendo las placas LoRa ESP32 y Raspberry Pi, así como el desarrollo del software de comunicación utilizando Python.

El capítulo cuatro presenta los resultados obtenidos a partir de las pruebas de campo realizadas en distintos entornos. Se analiza el alcance del sistema en escenarios urbanos y rurales.

Finalmente, el capítulo cinco expone la discusión y conclusiones del estudio. Se interpretan los resultados obtenidos, se identifican los desafíos enfrentados durante el desarrollo del proyecto y se proponen posibles mejoras para futuras implementaciones.

La metodología para llevar a cabo los objetivos es la siguiente

1. **Objetivo 1** Para alcanzar este objetivo, se seleccionarán y adquirirán las placas LoRa ESP32, y Raspberry Pi. Se diseñará un circuito básico de comunicación, asegurando la correcta integración entre los dispositivos mediante pruebas de conectividad en un entorno controlado.

2. **Objetivo 2** Se desarrollará un software personalizado utilizando lenguajes como C++ y Python para la comunicación entre los dispositivos LoRa. Se implementarán protocolos de transmisión de datos para maximizar la eficiencia y confiabilidad, basándose en estándares de comunicación IoT.

3. **Objetivo 3** Se llevará a cabo una serie de pruebas de campo en diferentes entornos (rurales y urbanos) para evaluar la cobertura y eficiencia del sistema LoRa. Las pruebas incluirán la medición de la calidad de la señal y el alcance efectivo de comunicación.

4. **Objetivo 4** Se optimizará el consumo energético mediante la implementación de técnicas de ahorro de energía, como la configuración de modos de bajo consumo en las placas LoRa y ajustes en el software de comunicación.

5. **Objetivo 5** Se documentarán todos los procesos y resultados, destacando las mejores prácticas y las posibles mejoras en la implementación de sistemas de comunicación LoRa para aplicaciones IoT.

1.5. Alcances y limitaciones

1.5.1. Alcances

Los principales aportes del trabajo son los siguientes:

- Diseño e implementación de Hardware y Software: Se desarrollará un sistema de comunicación basado en dos placas LoRa ESP32, y una Raspberry Pi, optimizado para garantizar una transmisión de datos eficientes y de bajo consumo energético en entornos con cobertura de red limitada.
- Pruebas de campo y validación: Se realizarán pruebas de campo en diferentes entornos (rurales y urbanos) para evaluar la cobertura, confiabilidad y eficiencia energética del sistema de transmisión de datos implementado, proporcionando resultados que permitan validar la viabilidad de LoRa para aplicaciones IoT.
- Optimización del consumo energético: Se explorarán diferentes configuraciones de hardware y software para optimizar el consumo energético del sistema, incluyendo la implementación de modos de bajo consumo en las placas LoRa y ajustes en los protocolos de comunicación.
- Generación de recomendación para aplicaciones futuras: A partir de los resultados obtenidos, se elaborarán recomendaciones para la implementación de sistemas de comunicación basados en LoRa en diversos contextos, aportando al conocimiento técnico y práctico en el ámbito de la ingeniería de telecomunicaciones.

1.5.2. Limitaciones

Este trabajo está sujeto a las siguientes limitacoines:

- Alcance geográfico limitado: Las pruebas de campo se realizarán exclusivamente en dos entornos específicos: uno rural y uno urbano, con un radio de cobertura máximo de 4 kilómetros. Esto implica que los resultados obtenidos pueden no ser representativos de otros contextos geográficos o condiciones ambientales.
- Recursos de Hardware Restringidos: El desarrollo del sistema se limitará al uso de dos placas LoRa ESP32, y una Raspberry Pi. No se contemplará la integración

de otros tipos de dispositivos o tecnologías adicionales que puedan mejorar la cobertura o eficiencia.

- Limitación en el análisis de datos: El análisis se centrará únicamente en la cobertura y eficiencia energética del sistema, excluyendo otros factores como la interferencia de señales de otros dispositivos o la variabilidad en las condiciones de transmisión debido a obstáculos físicos o cambios climáticos.
- Tiempo de desarrollo: El proyecto se llevará a cabo dentro de un marco de tiempo determinado, lo que puede limitar la profundidad de algunas pruebas o la posibilidad de realizar ajustes adicionales al sistema.

Capítulo 2

Marco Teórico

2.1. Introducción

En este capítulo se contextualizarán los puntos más importantes a considerar para comprender el tema tratado en este proyecto. Por ejemplo, es necesario conocer aspectos fundamentales de la comunicación entre máquinas, explicar algunos detalles sobre los enlaces inalámbricos, las redes de área amplia de baja potencia (LPWAN) y, en particular, la tecnología LoRa. Además, se dará énfasis a la importancia del *Signal to Noise Ratio* (SNR), una métrica que compara el nivel de una señal deseada con el nivel de ruido de fondo no deseado.

Para estudiar y evaluar la cobertura inalámbrica utilizando la tecnología LoRa, es fundamental comprender las capas inferiores del modelo de comunicación, es decir, la capa física y la capa de enlace, como se puede ver en la Fig. 2.1. La transmisión de los paquetes de datos, comúnmente llamados "paquetes" en informática, ocurre a través de los distintos enlaces de la red hasta que llegan al dispositivo terminal receptor.

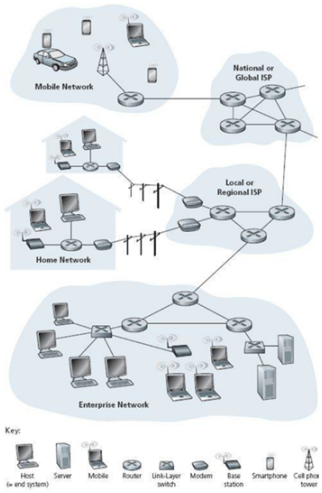

Fig. 2.1: Diagrama de la red de internet y sus enlaces.
[7]

2.2. Sistemas de comunicacion electrónicos

Los sistemas de comunicación electrónicos son aquellos que permiten la interconexión entre máquinas, teniendo como objetivo principal la transmisión, recepción y procesamiento de información a través de circuitos electrónicos. Para que la información, ya sea analógica o digital, pueda ser transmitida, debe ser transformada en fibra óptica o a través del aire, desde el transmisor hasta el receptor. [8] Estos sistemas se componen de tres elementos principales: un transmisor, un receptor y el medio de transmisión. En el contexto específico de este trabajo, el medio utilizado fue el aire, utilizando el espectro de radiofrecuencia de 915 MHz, (que es parte de la banda LoRa asignada a la región de américa del norte, Australia y algunos otros países.) para conectar el transmisor con el receptor.

En la Fig. 2.2 se pueden observar las distintas bandas del espectro radioeléctrico y cómo las tecnologías han evolucionado a lo largo del tiempo para aprovechar estas bandas.

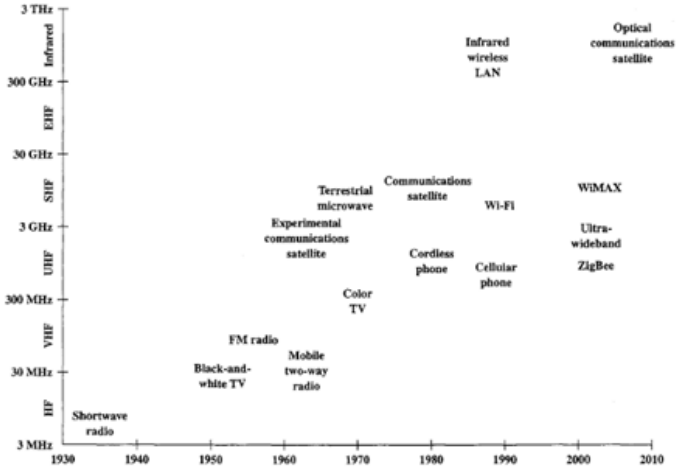

Fig. 2.2: Distintas bandas del espectro de radio y evolución del uso de tecnologías. [9]

Ya que las ondas electromagnéticas tienen un comportamiento complejo cuando se propagan a través de diferentes medios, y este fenómeno influye directamente en la calidad del enlace de comunicación. Al cambiar de medio, las ondas pueden sufrir reflexión, difracción o refrección. Estos efectos son especialmente relevantes en el aire, ya que la presencia de obstáculos como edificios, árboles o incluso condiciones meteorológicas puede afectar la calidad y el alcance de la señal.

Para evaluar adecuadamente el enlace de radio con LoRa, se analizarán los parámetros relacionados y se revisará el modelo de comunicación OSI, que se aplica en más de 130 países, enfocándose en sus capas iniciales.

2.2.1. Modelo de comunicacion OSI

El modelo de referencia Open System Interconnection (OSI) [8] es un conjunto de normas diseñado para facilitar la comunicación entre dispositivos como computadoras, terminales y redes. El modelo OSI organiza el proceso de comunicación en siete capas, tal como se muestra en la Fig. 2.3, donde se puede observar su estructura jerárquica dentro del protocolo. Cada capa del modelo añade encabezados a la información que se transmite, ocupando generalmente menos del 15 % del total del mensaje.

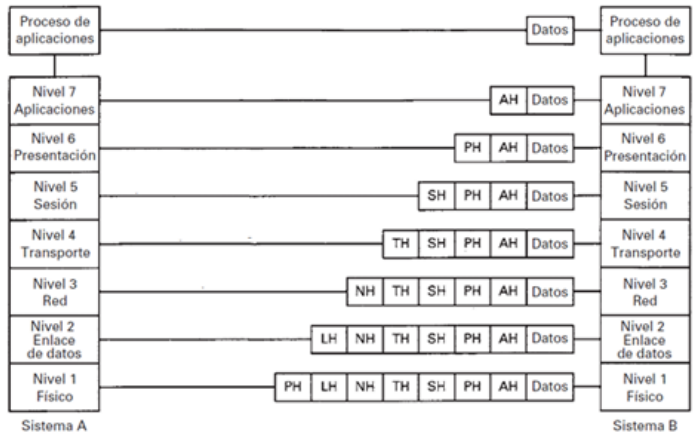

Fig. 2.3: Capas del modelo OSI.

[8]

Es importante resaltar que cada capa tiene su propio conjunto de protocolos de comunicación. Por ejemplo, en el estándar 802 del Institute of Electrical and Electronics Engineers (IEEE), que está basado en las referencias del modelo OSI, la sección 802.2 define el Protocolo de Control de Enlace Lógico (LLC), que forma parte de la capa de enlace de datos. La sección 802.11, conocida por ser la base del funcionamiento del Wi-Fi, se encarga de especificar los protocolos para redes inalámbricas. Las capas superiores (3 a 7) se ocupan de la comunicación directa entre computadoras, mientras que las capas inferiores (1 y 2) gestionan la transmisión física de los datos, a nivel de bits y tramas. En el caso de LoRa, los parámetros evaluados se relacionan con las dos primeras capas, que serán explicadas a continuación.

2.2.1.1. Modelo TCP/IP híbrido

Si bien el modelo OSI de 7 capas fue un referente teórico fundamental para entender la estructura de las redes de telecomunicaciones, en la práctica actual este modelo no se utiliza en su totalidad. La evolución de las tecnologías de redes ha dado paso al uso de un modelo más simple y adaptado a los estándares y protocolos reales, el modelo híbrido de 5 capas, también conocido como “modelo TCP/IP mejorado[10].

Este modelo es una simplificación del OSI que fusiona algunas de sus capas superiores, eliminando la capa de sesión (5) y la capa de presentación (6), que en la mayoría de las

implementaciones modernas están integradas dentro de las aplicaciones o protocolos de la capa de aplicación.

Las 5 capas del modelo híbrido son:

1. Capa física.

2. Capa de enlace de datos.

3. Capa de red.

4. Capa de transporte.

5. Capa de aplicación.

El modelo de 5 capas es más práctico y refleja mejor cómo funcionan realmente las redes actuales, simplificando las interacciones entre las capas. Las capas de sesión y presentación, que en el modelo OSI manejaban la sincronización y el formato de los datos, están incorporadas dentro de la capa de aplicación o en funciones específicas de los protocolos utilizados. Este modelo se adapta de forma natural al protocolo de Internet TCP7IP, que es la base de la mayoría de las redes modernas, incluidas las que utilizan tecnología LoRa para aplicaciones IoT, como el sistema desarrollado en este proyecto.

La adopción del modelo híbrido de 5 capas responde a la necesidad de simplificar las arquitecturas de red sin perder funcionalidad. Las redes modernas demandan eficiencia, flexibilidad y rapidez en la implementación de nuevos servicios, características que el modelo de 5 capas facilita al reducir la complejidad teórica del OSI. Además, este modelo se alinea completamente con los estándares de internet y las tecnologías emergentes que integran múltiples capas en soluciones optimizadas, como las redes de sensores inalámbricos y los sistemas IoT de bajo consumo energético.

2.2.2. Capa de enlace de datos

La capa de enlace de datos es la responsable de gestionar el envío de los paquetes a través de cada enlace de la red de manera individual, hasta que alcanzan su destino final. En esta capa, los datagramas que recibe de la capa de red se encapsulan en tramas y se aseguran de ser transmitidos de forma confiable entre los distintos segmentos de la

red.

Esta capa se compone de dos subcapas: la subcapa de control de enlace lógico (LLC), esta subcapa es responsable de la comunicación entre la capa de enlace de datos y la capa de red. Se encarga de identificar el protocolo de red que está en uso (por ejemplo, IPv4 o IPv6) y manejar la multiplexación de protocolos, lo que permite que varios protocolos de red coexistan en la misma conexión física. Y la subcapa de control de acceso al medio (MAC), esta subcapa gestiona el acceso al medio de transmisión y asegura que las tramas se envíen de manera ordenada. Aquí es donde se define el uso de direcciones MAC y se maneja la coordinación del acceso para evitar colisiones de datos.

Los enlaces que forman la red pueden variar significativamente, desde enlaces punto a punto hasta canales de difusión compartidos. En un enlace punto a punto, como el PPP (Point-to-Point Protocol), la coordinación de la comunicación es relativamente sencilla, ya que solo hay dos dispositivos involucrados, lo que facilita la prevención de colisiones de datos. Sin embargo, en redes inalámbricas, como una WAN o una red satelital, muchos dispositivos comparten el mismo canal de comunicación, lo que hace que la gestión y coordinación de las transmisiones sea mucho más compleja. En estos casos, la capa de enlace de datos utiliza protocolos de acceso al medio para organizar las transmisiones y evitar colisiones de tramas. Además, esta capa proporciona mecanismos para establecer, mantener y finalizar el enlace de datos, así como para detectar y corregir errores durante la transmisión.

La capa de enlace de datos, en conjunto con estos dispositivos y protocolos, desempeña un papel crucial en el funcionamiento fluido de las redes, asegurando que los datos viajen de manera segura y organizada a través de diversos medios y dispositivos.

Funciones principales de la capa de enlace de datos:

1. Encapsulación de tramas
2. Direccionamiento físico (MAC)
3. Detección y corrección de errores
4. Control de acceso al medio (MAC)
5. Fragmentación y reensamblaje

La capa de enlace de datos combina software y hardware para gestionar la transmisión de datos en la red, una parte de los protocolos que operan en esta capa es ejecutada directamente por la CPU, mientras que otra parte es manejada por el adaptador de red a nivel de hardware lo cual se puede apreciar en la Fig. 2.4. Esta capa actúa como un puente entre el código que procesa los datos y las señales físicas que se envían a través de la antena o el medio de transmisión.

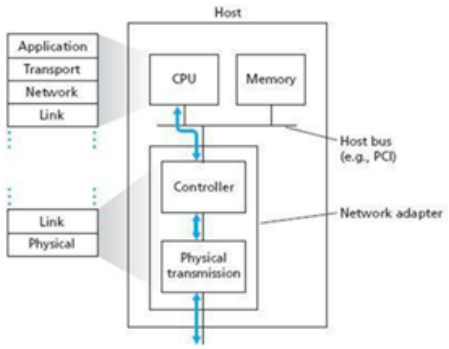

Fig. 2.4: Ejemplo de un adaptador de red y su conexión.
[7]

Protocolos de acceso múltiple

Estos protocolos permiten que varios dispositivos compartan el mismo canal de transmisión de forma organizada. A lo largo del tiempo, se han desarrollado numerosos protocolos de este tipo, los cuales se pueden clasificar en tres grandes categorías:

1.- Protocolos de particionamiento de canal: Están diseñados para dividir un canal de comunicación en varias partes, de manera que cada nodo tenga asignado su propio segmento para transmitir datos sin interferir con otros nodos.

a) Multiplexación por división de tiempo (TDM)

b) Multiplexación por división de frecuencia (FDM)

c) Acceso múltiple por división de código (CDMA)

2.- Protocolos de acceso aleatorio: En estos protocolos, cada nodo intenta transmitir cuando lo necesita y utilizando el máximo ancho de banda del canal disponible.

a) ALOHA con ranuras de tiempo

b) Acceso múltiple con sondeo de portadora (CSMA)

3.- Protocolos de toma de turno: Están diseñados para asegurar que cada nodo en la red tenga la oportunidad de transmitir de forma ordenada, mediante la asignación de turnos de transmisión.

a) Protocolo de sondeo (Polling)

b) Protocolo de paso de testigo (Token)

2.2.3. Capa física

En la capa física, la información se representa en su forma más básica: en bits. Mientras que la capa de enlace une estos bits en tramas o frames para su transmisión, la capa física se encarga de mover cada bit individualmente de un nodo al siguiente. Esta capa, que es la primera en el modelo de red, define los límites mínimos y máximos en valores como el voltaje, la frecuencia y la duración de los pulsos.

Una función esencial de la capa física es garantizar la sincronización y el control del ritmo de datos entre el emisor y el receptor. Esto asegura que ambos dispositivos interpreten los bits al mismo ritmo, evitando errores de comunicación debido a desincronizaciones. Sin una sincronización adecuada, los bits podrían ser interpretados incorrectamente, lo que afectaría la integridad de los datos transmitidos.

Además, la capa física debe abordar desafíos como la atenuación de la señal y el ruido electromagnético. La atenuación se refiere a la disminución de la potencia de la señal a medida que se propaga por el medio, lo cual puede provocar pérdidas de información si no se compensa adecuadamente. El ruido electromagnético son señales indeseadas que se superponen a la señal original, pudiendo distorsionarla. Para mitigar estos efectos, la capa física implementa técnicas como el uso de amplificadores, filtros y métodos de modulación robustos.

Los principales objetivos de esta capa son:

1. Definir el medio físico de transmisión de los datos.

2. Definir las características materiales y eléctricas de la comunicación.

3. Definir las características funcionales de la interfaz.

Cada uno de estos objetivos permite que el hardware cumpla con las funciones

principales de la capa física, como el envío y recepción de bits a través del medio seleccionado, la gestión de las señales eléctricas, y la conexión básica (sin garantía de fiabilidad), lo cual constituye la base de las comunicaciones IoT a larga distancia. [9]

2.2.4. Modulación

Un tipo de modulación similar a la que utiliza LoRa es el Multiple Frequency Shift Keying (MFSK), en el que se emplean diversas frecuencias (M) para identificar los distintos símbolos, es decir, secuencias específicas de bits transmitidas. [9]

$$MFSK : S_i(t) = A\cos(2\pi f_i t) \quad ; \quad 1 \leq i \leq M \tag{2.1}$$

Donde:

- $f_i = f_c + (2i - 1 - M)f_d$
- f_c: Frecuencia portadora
- f_d: Delta entre frecuencias i
- M: Número de símbolos (tramas de bits) distintos, $M = 2^L$
- L: Número de bits del símbolo

La modulación Multiple Frequency Shift Keying (MFSK) y la modulación de espectro ensanchado por chirp (CSS) de LoRa son técnicas de transmisión que, aunque comparten algunas similitudes, se adaptan a diferentes necesidades. MFSK transmite datos mediante la asignación de distintas frecuencias a cada símbolo, lo que lo hace resistente a interferencias y adecuado para entornos ruidosos.

2.2.5. Optimización de la sensibilidad mediante espectro ensanchado (SS)

En el ámbito de la teoría de la información, el teorema de Shannon-Hartley establece el límite máximo de velocidad a la que pueden transmitirse datos en un canal de comunicación con ruido blanco aditivo (AWGN) dentro de un ancho de banda específico.

Este límite, conocido como la *capacidad del canal de Shannon* [11], determina la tasa máxima de datos:

$$C = B \log_2(1 + \frac{S}{N}) \tag{2.2}$$

Donde los términos representan:

- C: Capacidad del canal en bits por segundo (bps)
- B: Ancho de banda del canal en hercios (Hz)
- S: Potencia promedio de la señal en vatios (W)
- N: Potencia promedio del ruido e interferencias en vatios (W)
- S/N: Relación señal a ruido (SNR)

Usando propiedades del logaritmo natural, esta fórmula puede adaptarse para ser representada de manera aproximada por:

$$\frac{C}{B} \approx 1{,}443 \cdot \frac{S}{N} \tag{2.3}$$

En aplicaciones que usan *espectro ensanchado* (SS), los valores de SNR son generalmente bajos, ya que la potencia de la señal puede estar incluso por debajo del nivel de ruido. Bajo condiciones en las que $\frac{S}{N} \ll 1$, la ecuación puede simplificarse aún más, resultando en:

$$\frac{C}{B} \approx \frac{S}{N} \quad \text{o bien} \quad \frac{N}{S} \approx \frac{B}{C} \tag{2.4}$$

Este resultado indica que, para mantener una transmisión de datos sin errores en un canal con SNR bajo, se puede aumentar la sensibilidad del receptor ampliando el ancho de banda de la señal (B). Esto permite que el sistema pueda captar señales débiles y mejorar su rendimiento, incluso en condiciones de ruido elevado.

Existen dos métodos principales de espectro ensanchado: *Direct Sequence Spread Spectrum* (DSSS) y *Frequency Hopping Spread Spectrum* (FHSS). En DSSS, cada bit de datos se multiplica por una secuencia de código a una tasa mucho más alta, lo que ensancha la señal en el dominio de frecuencia y reduce su susceptibilidad al ruido. Por

otro lado, en FHSS, la señal salta entre diferentes frecuencias de forma pseudoaleatoria durante la transmisión, lo que la hace menos propensa a interferencias en una frecuencia específica. Cada método tiene sus propias ventajas; DSSS ofrece una mayor resistencia al ruido continuo, mientras que FHSS es más resistente a interferencias en frecuencias específicas. Estas técnicas permiten que los sistemas de comunicación optimicen la transmisión en función de sus necesidades y entorno operativo.

En el contexto de LoRa, el espectro ensanchado permite alcanzar transmisiones a larga distancia sin comprometer la eficiencia energética, lo cual es esencial para aplicaciones IoT. LoRa utiliza una variante de DSSS, denominada *Chirp Spread Spectrum* (CSS).

Un tipo particular de *Espectro Ensanchado* (SS) es el *Direct Sequence Spread Spectrum* (DSSS), cuya operación es similar al proceso de modulación de LoRa. En la Fig. 2.5 se muestra un ejemplo donde la información se multiplexa con un código de expansión (secuencia de bits PN generada localmente) en el transmisor, y luego se recupera en el receptor aplicando el mismo proceso.

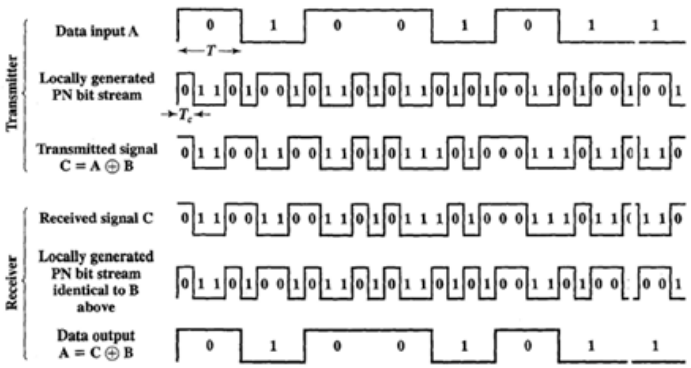

Fig. 2.5: Ejemplo de DSSS.
[9]

La frecuencia de la señal de información $(1/T)$ se incrementa cuatro veces al combinarse con el código $(1/T_c)$, como se puede ver en la Fig. 2.6.

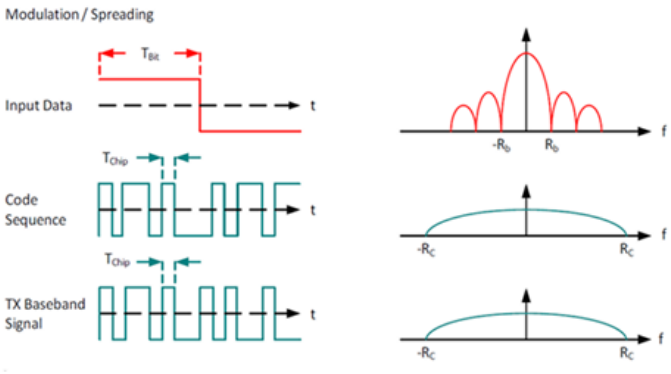

Fig. 2.6: Ejemplo de esparcimiento SS.
[11]

Una característica fundamental de los sistemas de espectro ensanchado es que la señal de información se envía a un codificador de canal que genera una señal analógica con un ancho de banda relativamente limitado. A continuación, esta señal es modulada utilizando una secuencia específica conocida como código de esparcimiento o secuencia de expansión (spreading code), lo cual expande significativamente el ancho de banda de la señal transmitida. En el receptor, se utiliza el mismo código de esparcimiento para demodular la señal, permitiendo su recuperación. El espectro ensanchado proporciona tres beneficios claves: [9]

1. Inmunidad a ruido e interferencias (Resiliencia a ruido y multipath).
2. Seguridad y privacidad mediante codificación de la señal.
3. Capacidad de múltiples usuarios en un canal (uso de CDMA).

2.3. Redes inalámbricas de área amplia y baja potencia (LPWAN)

Low Power Wide Area Networks (LPWAN) han emergido como una tecnología esencial en el ámbito de la Internet de las Cosas (IoT) y la comunicación de máquina a máquina (M2M). Estas redes ofrecen una solución eficaz para la conectividad de dispositivos en entornos donde es fundamental mantener un bajo consumo de energía y una cobertura extendida [12].

Las LPWAN se destacan como la opción ideal para aplicaciones que requieren transmisión de datos a largas distancias como una frecuencia baja, pero que no exigen tasas de transferencia de datos elevadas. A diferencia de las redes celulares convencionales, como 4G y 5G, que priorizan la velocidad y la baja latencia, las LPWAN están diseñadas para optimizar la duración de la batería y la cobertura en entornos de difícil acceso, como áreas rurales y zonas urbanas densas.

Las LPWAN se caracterizan por ofrecer una combinación única de bajo consumo de energía, amplia cobertura y capacidad para conectar un gran número de dispositivos. Una de sus principales ventajas es la posibilidad de operar con una cobertura de hasta 40km en zonas rurales y entre 1 y 5 km en áreas urbanas densas[13]. Esta cobertura extendida se logra utilizando frecuencias sub-GHz, que tienen mejores características de programación en comparación con bandas más altas, como la de 2.4 GHz, comúnmente utilizada en Wi-Fi y Bluetooth. Las frecuencias sub-GHz permiten una mayor penetración de señal a través de obstáculos y estructuras, lo que es crucial para aplicaciones de monitoreo en áreas como la agricultura de precisión, donde los sensores pueden estar ubicados en terrenos irregulares o entre cultivos densos.[14]

Dentro del ecosistema de LPWAN, existen varias tecnologías que se destacan por sus enfoques y características particulares, tales como LoRaWAN, Sigfox, Narrowband IoT (NB-IoT) y LTE-M [15]. Cada una de estas tecnologías tiene sus propias fortalezas y limitaciones, lo que las hace más o menos adecuadas para diferentes casos de uso.

La seguridad es otro desafío importante, debido a la simplicidad de los dispositivos y el riesgo de ciberataques en entornos IoT. Las redes LPWAN deben implementar mecanismos robustos de autenticación y cifrado para proteger los datos transmitidos, así como protocolos que aseguren la privacidad de los usuarios y la integridad de la red. Además, la capacidad de estas redes para manejar la interferencia en entornos densamente poblados es fundamental para mantener la calidad del servicio y evitar la pérdida de paquetes de datos.

Aunque las LPWAN están diseñadas para aplicaciones de baja velocidad de datos y larga duración de batería, se complementan con tecnologías móviles de alta velocidad como 5G. Las redes 5G son ideales para aplicaciones que requieren una baja latencia y un gran ancho de banda, como la realidad aumentada o la conducción autónomo. Sin embargo, no son adecuadas para conectar dispositivos de bajo consumo en áreas

remotas debido a su alto costo y consumo energético. [14]

Las LPWAN han demostrado ser una solución esencial para el crecimiento del Internet de las Cosas, permitiendo la conexión de dispositivos a largo plazo y con bajo consumo de energía en entornos donde las redes tradicionales no son viables. Su capacidad para proporcionar conectividad económica y eficiente las convierte en una herramienta clave para aplicaciones de monitoreo y control a gran escala, tanto en entornos urbanos como rurales. A medida que tecnologías como 5G continúan desarrollándose, el papel de las LPWAN como una tecnología complementaria seguirá siendo fundamental para el avance de IoT, proporcionando una base sólida para la conectividad masiva en el futuro.[16]

2.3.1. LoRaWAN

LoRaWAN combina un diseño de capa física eficiente, basado en la modulación chirp, con un protocolo MAC robusto que admite múltiples clases de dispositivos. Esto le permite satisfacer una amplia gama de necesidades, desde sistemas de monitoreo en tiempo real hasta aplicaciones de seguimiento y localización. Además, su capacidad para operar en bandas de frecuencia no licenciadas y su compatibilidad con redes privadas y públicas han hecho de LoRaWAN una tecnología esencial para el ecosistema IoT. En la Fig. 2.7 Se puede apreciar el diagrama de conexión de un sistema LoRaWAN.

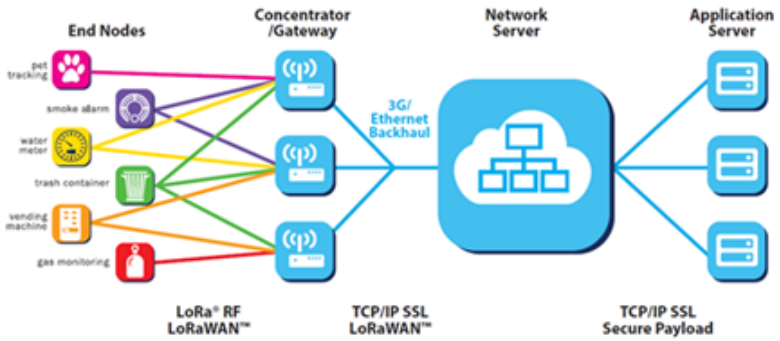

Fig. 2.7: Diagrama de conexión de un sistema LoRaWAN. [17]

Arquitectura y diseño técnico de LoRaWAN

La capa física de LoRaWAN está basada en el protocolo LoRa, una tecnología patentada

por Semtech en 2014. La característica distintiva de LoRa es su modulación de espectro ensanchado basada en chirps. Este enfoque genera señales que varían de forma controlada en frecuencia durante un intervalo definido, lo que mejora la resistencia de las comunicaciones frente a interferencias, ruido y pérdida de la señal. En donde más adelante se explicará más a fondo esta modulación y algunas especificaciones extras.[18]

En un experimento llevado a cabo en un entorno suburbano, se demostró que utilizando un SF12, un nodo LoRaWAN fue capaz de transmitir datos a una distancia de 2.8 km con una tasa de recepción de paquetes (PRR) del 80 %. Sin embargo, la reducir el SF a 7, la distancia efectiva disminuyó a 650 m, aunque la PRR mejoró al 95 %. Este experimento destaca el equilibrio que debe mantenerse entre el alcance y la tasa de datos en aplicaciones específicas.[19]

El protocolo MAC de LoRaWAN está diseñado para optimizar el acceso al medio, soportar comunicaciones bidireccionales y manejar dispositivos con capacidades energéticas limitadas [5]. Los dispositivos se dividen en tres clases principales: [20]

1. Clase A: Abre breves ventanas de recepción, solo para transmitir datos

2. Clase B: Introduce ventanas de recepción adicionales.

3. Clase C: Ofrece ventanas de recepción casi continuas.

La arquitectura de red de LoRaWAN sigue un modelo en estrella, donde los nodos finales transmiten datos directamente a los gateways. Estos gateways reenvían los datos al servidor de red central, el cual administra la autenticación, el enrutamiento y la entrega de datos a los servidores de aplicaciones.

Rendimiento técnico de LoRaWAN

LoRaWAN es capaz de cubrir grandes áreas geográficas con un solo gateway, lo que lo hace ideal para aplicaciones rurales o de baja densidad de infraestructura. En entornos rurales abiertos, la tecnología ha demostrado alcanzar distancias superiores a 15 km, mientras que, en áreas urbanas densas, el alcance suele limitarse a entre 1 y 2 km debido a la interferencia y los obstáculos físicos.

Seguridad en LoRaWAN

La seguridad en LoRaWAN es un aspecto crítico, dado que sus aplicaciones suelen incluir datos sensibles, como información médica o datos de infraestructura crítica.

El diseño de LoRaWAN incorpora dos niveles principales de cifrado:

- Cifrado de datos: Utiliza claves de sesión (AppSKey).
- Autenticación.

El estándar define dos métodos de activación:

- Over-The-Air Activation (OTAA): Consiste en un proceso dinámico en el que los dispositivos negocian nuevas claves de sesión en cada unión.
- Activation by Personalization (ABP): Usa claves estáticas configuradas previamente en los dispositivos.

A pesar de las medidas implementadas, existen vulnerabilidades en LoRaWAN que pueden comprometer su uso en aplicaciones críticas:

- Ataques de repetición.
- Eavesdropping.
- Bit-flipping.
- Ataques de canal lateral.

2.3.2. LoRa

LoRa se ha convertido en una tecnología fundamental dentro del ecosistema de las redes LPWAN debido a su capacidad para proporcionar conectividad a larga distancia con un bajo consumo energético. Su atractivo comercial radica en que un solo *gateway* permite generar una red de topología estrella y de largo alcance, capaz de sostener miles de nodos conectados transmitiendo con bajo *duty cycle*. A medida que la demanda de dispositivos IoT continúa creciendo, la necesidad de soluciones de conectividad eficientes y escalables ha impulsado el desarrollo de tecnologías como LoRa. Esta tecnología responde a la creciente necesidad de comunicación de sensores distribuidos en amplias áreas geográficas, como zonas rurales o entornos urbanos densos donde otras tecnologías no alcanzan o su costo es prohibitivo[17].

En LoRa predomina el tráfico de *uplink* sobre el de *downlink*, ya que su uso está pensado para que los dispositivos finales (*end-devices*) envíen información al *gateway*.

Dependiendo de la región y sus regulaciones, esta tecnología puede funcionar en las bandas ISM de 433 MHz, 868 MHz o 915 MHz.

La modulación de LoRa, basada en *Chirp Spread Spectrum* (CSS), aprovecha la resistencia de esta técnica a las interferencias, una característica derivada de su origen militar. Este enfoque ha permitido que LoRa sea una opción viable para aplicaciones donde la fiabilidad de la conexión es crucial, como el monitoreo de infraestructura crítica y la agricultura de precisión[21].

La modulación CSS de LoRa utiliza señales o pulsos *chirp* con una variación lineal de frecuencia en el tiempo para codificar la información[22]. Debido a la linealidad de los pulsos *chirp*, los desfases de frecuencia entre el receptor y el transmisor son equivalentes a los desfases en el tiempo, lo que hace que esta modulación tenga inmunidad al efecto Doppler, permitiendo que las transmisiones sean más resistentes al ruido y a las interferencias. En este proceso, el ancho de banda de la señal se expande, lo que facilita que los receptores puedan recuperar la señal incluso en niveles de ruido altos. Esto es particularmente útil en escenarios de IoT donde la calidad de la señal puede verse comprometida por obstáculos físicos o interferencias de otras fuentes de radiofrecuencia.

Un aspecto clave de la modulación CSS es la relación entre los factores de dispersión (*Spreading Factor*, SF) y la tasa de datos. Al aumentar el SF, se incrementa la sensibilidad del receptor, permitiendo que la señal sea detectada a mayores distancias, aunque a costa de una menor tasa de transmisión de datos. Esto implica un equilibrio que debe ajustarse según los requisitos de la aplicación, como en el caso de sensores que envían datos esporádicos, pero deben mantenerse conectados en ubicaciones remotas.

El *Chirp Spread Spectrum* (CSS) es una técnica de modulación que comparte características con el acceso múltiple por división de código (CDMA) y tiene similitudes con el *Direct Sequence Spread Spectrum* (DSSS). Se puede considerar una forma de modulación de espectro ensanchado, donde se permite la transmisión de múltiples señales utilizando diferentes códigos. Como se explicó anteriormente, su funcionamiento se basa en el uso de pulsos *chirp*, que son señales cuya frecuencia varía progresivamente, ya sea aumentando (*upchirp*) o disminuyendo (*downchirp*) durante un intervalo de tiempo determinado.

En la Fig. 2.8 se pueden observar los dos tipos de *chirps* mencionados, donde los *upchirps* elevan su frecuencia a lo largo del tiempo, mientras que los *downchirps* la reducen.

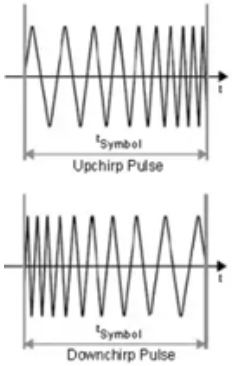

Fig. 2.8: Pulsos upchirps y downchirp.
[22]

CDMA (Code Division Multiple Access) es una técnica de comunicación inalámbrica que permite que múltiples usuarios compartan un mismo canal de frecuencia al utilizar códigos únicos para cada transmisión. Cada señal se modula con un código específico, lo que permite que los receptores separen las señales al reconocer el código correspondiente, incluso si se transmiten simultáneamente en la misma frecuencia. [22]

DSSS (Direct Sequence Spread Spectrum) es un método de modulación de espectro ensanchado donde la señal original se multiplica por una secuencia de código de mayor ancho de banda. Esto dispersa la energía de la señal en un rango de frecuencias más amplio, lo que hace que la transmisión sea más resistente a las interferencias y mejora la seguridad de la comunicación.

A medida que aumenta la adopción de LoRa, especialmente en ciudades inteligentes y grandes áreas agrícolas, la escalabilidad se convierte en un desafío importante. LoRaWAN fue diseñado para soportar una gran cantidad de dispositivos, pero la eficiencia del protocolo de acceso al medio (MAC layer) es limitada en redes muy densas debido a su enfoque basado en ALOHA puro, donde los dispositivos transmiten sin verificar si el canal está libre.

En el ámbito de la agricultura inteligente, LoRa ha demostrado ser una herramienta valiosa para la optimización del uso de agua y el control de plagas. Sensores de humedad del suelo y estaciones meteorológicas con conectividad LoRa pueden proporcionar datos en tiempo real a plataformas de gestión agrícola, lo que permite a los agricultores ajustar sus prácticas de riego y reducir el uso de insumos químicos.

Las ciudades inteligentes son otro campo donde LoRa ha encontrado una amplia aplicación. Ejemplos de ello incluyen el monitoreo de la calidad del aire, la gestión de estacionamientos y el control de alumbrado público. En estos casos, LoRa facilita la implementación de soluciones que requieren una baja latencia y una cobertura amplia, sin la necesidad de recurrir a infraestructura de red celular costosa.

En el sector de la salud, LoRa está siendo utilizado para el monitoreo remoto de pacientes, especialmente en áreas rurales donde la cobertura de redes móviles es limitada. Dispositivos equipados con LoRa pueden transmitir datos vitales como la frecuencia cardíaca y la saturación de oxígeno a centros de salud, permitiendo una atención más oportuna y mejorando la calidad de vida de los pacientes.

La simulación de redes LoRa es esencial para comprender su comportamiento en diferentes escenarios antes de realizar despliegues físicos. Herramientas como LoRa y LoRaSim permiten simular la comunicación entre dispositivos y gateways, evaluando parámetros como el consumo energético, la tasa de pérdida de paquetes y la interferencia en diferentes condiciones ambientales. Estas herramientas también ayudan a evaluar el impacto de la densidad de dispositivos y la elección de los factores de dispersión en la eficiencia de la red. Esto es especialmente útil en entornos urbanos, donde la presencia de múltiples gateways y la superposición de señales puede llevar a un alto nivel de interferencia. La simulación permite además probar diferentes configuraciones de la red para optimizar la ubicación de los gateways y mejorar la cobertura de la red. Esto reduce los costos asociados a la instalación de la infraestructura y asegura que la red funcione de manera eficiente desde el inicio. [4]

A medida que la demanda de soluciones IoT continúa creciendo, LoRa se posiciona como una de las tecnologías más prometedoras para ofrecer conectividad a bajo costo y con una gran cobertura. Las mejoras continuas en el estándar LoRaWAN, como la mayor integración con plataformas de computación en la nube y la capacidad de soportar actualizaciones de firmware remotas, facilitan su adopción en proyectos de IoT cada vez más complejos.

2.3.3. Ganancia de procesamiento en LoRa

La tecnología LoRa optimiza la sensibilidad del receptor mediante la *ganancia de procesamiento* (G_p), la cual es una característica propia de la modulación CSS. Esta ganancia está directamente relacionada con la tasa de ensanchamiento de la señal, la cual depende de la relación entre el número de chirps por bit transmitido. En términos matemáticos, G_p se expresa en decibeles (dB) y se determina mediante la siguiente ecuación:

$$G_p = 10 \log_{10} \left(\frac{R_c}{R_b} \right) \quad [\text{dB}] \tag{2.5}$$

Donde:

- R_c : representa la tasa de chirp (chirps/seg)
- R_b : es la tasa de datos (bits/seg)

Gracias a esta ganancia, LoRa puede recuperar señales incluso en condiciones de relación señal-ruido (SNR) negativas. Además, el ancho de banda de la señal transmitida coincide con el ancho de banda del chirp (BW), lo que implica que la tasa de chirp (R_c) es equivalente al ancho de banda de operación. En los módulos LoRa, los valores típicos de BW utilizados son 125 kHz, 250 kHz y 500 kHz.

2.3.4. Parámetros claves en la modulación LoRa

La configuración de LoRa puede ajustarse mediante tres parámetros fundamentales que influyen en su rendimiento:

- **Ancho de banda (BW)**: Define la cantidad de espectro utilizado en la transmisión.
- **Factor de expansión (SF)**: Determina la cantidad de chirps que conforman un símbolo, lo que influye en la duración de este y en la capacidad de transmitir datos.
- **Tasa de codificación (CR)**: Representa la cantidad de bits de información útiles dentro de un código de corrección de errores.

En LoRa, cada símbolo está compuesto por 2^{SF} chirps, lo que significa que cada símbolo puede codificar SF bits de información. El periodo de símbolo (T_s) y la tasa de símbolos (R_s) se relacionan mediante la siguiente ecuación:

$$R_s = \frac{BW}{2^{SF}} \quad \text{[símbolos/seg]} \tag{2.6}$$

A partir de esto, la tasa de chirp (R_c) puede expresarse como:

$$R_c = R_s \cdot 2^{SF} = BW \quad \text{[chirps/seg]} \tag{2.7}$$

Adicionalmente, el código de corrección de errores (CR) se define como:

$$CR = \frac{4}{4+n}, \quad n = 1, 2, 3, 4 \tag{2.8}$$

Finalmente, la tasa de datos (R_b) que puede alcanzarse bajo una configuración dada de LoRa se obtiene mediante la siguiente expresión:

$$R_b = SF \cdot CR \cdot \frac{BW}{2^{SF}} \quad \text{[bits/seg]} \tag{2.9}$$

2.3.5. Interdependencia entre parámetros y sensibilidad del receptor

Dado que la tasa de chirp depende únicamente del ancho de banda, esto tiene implicancias importantes en la modulación. Un incremento de una unidad en el SF implica que la frecuencia de cada chirp se divide por dos y que la duración del símbolo se duplica. Como resultado, la tasa de símbolos y la tasa de datos están directamente relacionadas con el ancho de banda, lo que significa que al duplicar el BW, también se duplica la tasa de transmisión de datos.

Sin embargo, estos parámetros también afectan la sensibilidad del receptor. Algunas observaciones claves incluyen:

- Aumentar el SF incrementa la duración del símbolo, lo que mejora la sensibilidad del receptor al permitir la detección de señales más débiles, pero también reduce la tasa de datos.
- Incrementar el ancho de banda (BW) mejora la tasa de transmisión, pero reduce la sensibilidad, ya que se necesita una señal más fuerte para ser detectada correctamente.
- Reducir la tasa de corrección de errores (CR) ayuda a disminuir la tasa de error en paquetes (*Packet Error Rate*, PER). Un CR de 4/8 ofrece mayor tolerancia a interferencias en comparación con CR de 4/5.

En la siguiente tabla 2.1 se muestran los valores de sensibilidad esperados para distintas combinaciones de SF y BW [14]:

Tabla 2.1: Sensibilidad del receptor LoRa para diferentes combinaciones de SF y BW.

SF	**BW = 125 kHz**	**BW = 250 kHz**	**BW = 500 kHz**
7	-123 dBm	-120 dBm	-116 dBm
8	-126 dBm	-123 dBm	-119 dBm
9	-129 dBm	-125 dBm	-122 dBm
10	-132 dBm	-128 dBm	-125 dBm
11	-133 dBm	-130 dBm	-128 dBm
12	-136 dBm	-133 dBm	-130 dBm

Se observa que a mayor SF, la sensibilidad del receptor mejora, permitiendo la recepción de señales más débiles. Sin embargo, esto implica un trade-off, ya que también reduce la velocidad de transmisión.

2.3.6. Frame format para el paquete de LoRa

El protocolo LoRa define una estructura específica para los paquetes transmitidos, la cual incluye secciones fijas y opcionales, como se puede ver en la Fig. 2.9. Entre los elementos obligatorios se encuentran el preámbulo y el payload, mientras que el header y el CRC del payload pueden o no estar presentes según la configuración elegida.

- Preambulo: El preámbulo cumple la función de establecer la sincronización entre el transmisor y el receptor antes de la transmisión de datos.
- Header: El encabezado es un componente opcional dentro del paquete del LoRa y está configurado con una tasa de corrección de errores (CR) de 4/8. Su función principal es proporcionar información clave sobre el payload,
- Payload: El payload es el bloque de datos que se desea transmitir, con un tamaño que puede oscilar entre 2 y 255 bytes.
- CRC del payload: En caso de estar presente, este campo implementa un mecanismo de corrección de errores hacia adelante (FEC, Forward Error Correction) basado en un CRC de 16 bits.

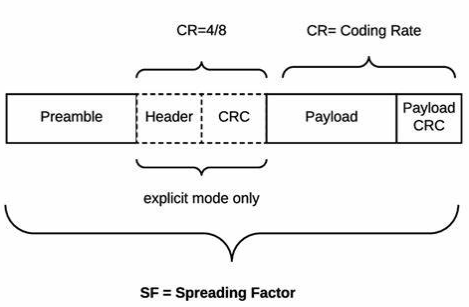

Fig. 2.9: Frame format.

2.3.7. Relación señal a ruido (SNR)

En un sistema de comunicación inalámbrico, la relación señal a ruido (*SNR*, por sus siglas en inglés *Signal-to-Noise Ratio*) representa el cociente entre la potencia de la señal transmitida que logra ser recibida y la potencia del ruido presente en el receptor. Este parámetro es fundamental para evaluar la calidad de la transmisión, ya que un SNR alto implica una mejor recepción de la señal con menor interferencia.

Matemáticamente, el SNR se expresa mediante la siguiente ecuación:

$$\text{SNR} = \frac{P_{SR}}{P_R} \tag{2.10}$$

donde P_{SR} representa la potencia de la señal recibida y P_R es la potencia del ruido en el canal.

En el caso de comunicaciones digitales, es común reformular esta ecuación en términos de la relación entre la energía por bit (E_b) y la densidad espectral del ruido (N_0). Al introducir estos conceptos, el SNR puede expresarse como:

$$\text{SNR} = \frac{R_b}{BW} \cdot \frac{E_b}{N_0} \tag{2.11}$$

donde:

- R_b es la tasa de transmisión de datos en bits por segundo (bps).
- BW corresponde al ancho de banda del canal en Hertz (Hz).

2.3.7.1. Impacto del SNR en la Tasa de Error de Bits (BER)

La BER (Bit Error Rate) mide la probabilidad de que un bit se reciba de forma errónea debido a interferencias o degradaciones en el canal. Aumentar la potencia de transmisión podría mejorar el SNR y reducir el BER, sin embargo, en tecnologías de baja potencia y largo alcance (LPWAN) como LoRa, esta estrategia no es eficiente, ya que incrementaría el consumo energético del sistema. En lugar de depender únicamente de una mayor potencia de transmisión, LoRa implementa mecanismos como el ADR (Adapttive Data Rate) para optimizar el rendimiento sin sacrificar eficiencia energética.

En el diseño de enlaces inalámbricos, es fundamental establecer un BER máximo aceptable, determinando así el umbral mínimo de calidad requerido para que la comunicación se considere confiable. Durante la implementación real del sistema, se espera que el desempeño del enlace supere este valor umbral para garantizar una transmisión efectiva.

2.3.7.2. Relación entre SNR y la modulación utilizada

La relación entre la razón señal a ruido (SNR) y la modulación es un aspecto crucial en la transmisión de datos, ya que define la robustez del enlace frente a interferencias

y ruido. En términos generales, modulaciones más complejas requieren un SNR mayor para garantizar una tasa de error de bits (BER) aceptable.

Por ejemplo, la modulación FSK, que codifica la información en cambios de frecuencia, necesita niveles de SNR relativamente altos para mantener una comunicación confiable, ya que su capacidad de corrección de errores es limitada. En contraste, la tecnología LoRa emplea CSS, una modulación basada en barridos de frecuencia (chirps) que ofrece una mayor resistencia al ruido y permite operar incluso con valores negativos de SNR. Esta diferencia es clave en redes LPWAN como LoRaWAN, ya que permite alcanzar largas distancias sin depender de una potencia de transmisión elevada, lo que se traduce en un menor consumo energético.

2.3.7.3. Optimización del SNR en redes LoRa

Dado que en aplicaciones IoT con LoRa el aumento de la potencia de transmisión no siempre es viable por restricciones energéticas, existen otras estrategias para optimizar el SNR y mejorar la calidad del enlace:

Ajuste del Spreading Factor (SF): En LoRa, el Spreading Factor define la cantidad de símbolos utilizados para representar un bit. Un SF alto incrementa la sensibilidad del recetor, permitiendo operar con un SNR más bajo, pero a costa de una menor tasa de transmisión. Por ejemplo, un SF de 7 permite mayores tasas de datos, mientras que un SF de 12 maximiza el alcance y la robustez.

Selección de canales menos congestionados: En entornos con múltiples redes LoRa operando en la misma banda de frecuencia, el uso de canales con menor interferencia mejora el SNR. Para ello, pueden emplearse algoritmos de optimización de frecuencia o simplemente ajustar manualmente las bandas de operación.

Uso de antenas con mayor ganancia: El empleo de antenas direccionales o de alta ganancia mejora la recepción de la señal sin necesidad de aumentar la potencia de transmisión, lo que ayuda a optimizar el balance de SNR en enlaces de larga distancia.

Adaptive Data Rate (ADR): LoRaWAN incorpora un mecanismo llamado ADR, que ajusta automáticamente la tasa de transmisión, la potencia de emisión y el SF según la calidad del enlace. Esto permite mantener un SNR óptimo sin desperdiciar recursos energéticos.

Capítulo 3

Desarrollo

3.1. Requerimientos

El presente proyecto tiene como objetivo principal desarrollar un sistema de transmisión de datos a larga distancia utilizando tecnología LoRa para aplicaciones del Internet de las Cosas (IoT). Para cumplir con este propósito, se definen los siguientes requerimientos técnicos y funcionales, directamente relacionados con los objetivos planteados en el capítulo 1:

- **Transmisión de datos a larga distancia**: Este punto está relacionado con lo mencionado en el objetivo general y los objetivos específicos 1 y 3. El sistema debe ser capaz de transmitir datos a varios kilómetros de distancia, especialmente en zonas rurales o de difícil acceso, donde no existe infraestructura de red tradicional. La elección de la tecnología LoRa se justifica por su capacidad de alcanzar largas distancias utilizando bandas de frecuencia no licenciadas, lo que permite establecer soluciones de conectividad sin depender de redes celulares o Wi-Fi, las cuales suelen presentar limitaciones técnicas y económicas en estos entornos [23].
- **Bajo consumo energético**: Este punto está vinculado con el objetivo general. En aplicaciones IoT, muchos dispositivos deben operar durante largos periodos de tiempo utilizando baterías, lo que vuelve imprescindible la eficiencia energética. LoRa permite configurar parámetros como el Spreading Factor (SF), el cual influye directamente en el alcance y el consumo energético. A mayor SF, se

incrementa el alcance, pero también el tiempo de transmisión, lo que repercute en el consumo. Esta posibilidad de ajustar el SF permite encontrar un equilibrio entre eficiencia energética y rendimiento del sistema, según el escenario de uso. Además, se contempla la realización de un experimento específico para evaluar el consumo energético del nodo transmisor en función de distintos parámetros LoRa, especialmente el SF, con el fin de identificar configuraciones óptimas para aplicaciones de bajo consumo.

- **Configurabilidad y adaptabilidad**: Este punto está directamente relacionado con los objetivos específicos 2 y 4. El sistema debe ser configurable y adaptable a distintos escenarios de uso, lo que implica la capacidad de modificar parámetros como el Spreading Factor (SF), el ancho de banda (Bandwidth, BW) y la tasa de codificación (Coding Rate, CR). Esta flexibilidad permite optimizar el rendimiento del sistema de acuerdo al entorno (urbano o rural), el nivel de interferencias, y los requisitos de alcance o eficiencia energética, lo que resulta esencial para la aplicabilidad del sistema en diferentes contextos IoT.
- **Integración con plataformas IoT y formatos de salida**: Esto está relacionado con el objetivo general y objetivo específico 2. El sistema debe generar datos en formatos estándar simulando un payload de aplicación real de IoT, y entregar esos datos como archivos, que faciliten su posterior análisis y su integración con plataformas de monitoreo o visualización. Esto contribuye a que la solución sea escalable y aplicable en entornos reales de IoT.
- **Capacidad de evaluación en terreno y documentación**: Este punto está vinculado con los objtivos específicos 3 y 5.El sistema debe estar preparado para realizar pruebas en terreno, tanto en entornos urbanos como rurales, y permitir la recolección de métricas clave como RSSI y la tasa de pérdida de paquetes. Además, debe contar con mecanismos que faciliten la documentación sistemática de los resultados, con el fin de evaluar el desempeño del sistema y generar recomendaciones para futuras implementaciones basadas en tecnología LoRa.

3.2. Diseño

El diseño del sistema se realizó con el objetivo de cumplir con los requerimientos técnicos y funcionales establecidos en la sección 3.1, respondiendo directamente a cada uno de ellos mediante decisiones específicas en la arquitectura, componentes y configuración del sistema experimental. A continuación, se detalla cómo cada requerimiento fue abordado en el diseño:

- **Transmisión de datos a larga distancia**: Para dar respuesta a este requerimiento, se optó por la selección de la tecnología LoRa como medio de comunicación entre los nodos. Se diseñó un sistema compuesto por una placa transmisora (Nodo TX) y una placa receptora (Nodo RX), que se comunican utilizando modulación LoRa en la banda de 915 MHz, empleando antenas integradas. La elección de LoRa como tecnología de comunicación se justifica principalmente por su capacidad para cubrir grandes distancias con bajo consumo energético, lo que la hace ideal para aplicaciones del Internet de las Cosas. A diferencia de otras tecnologías disponibles en las placas utilizadas como Wi-Fi y Bluetooth, LoRa permite mantener la transmisión incluso en zonas remotas, donde no hay cobertura de red o infraestructura disponible. Wi-Fi presenta un consumo energético elevado y un alcance limitado, mientras que Bluetooth está diseñado para distancias cortas, lo cual no cumple con los requerimientos del sistema.
- **Bajo consumo energético**: El diseño consideró el consumo energético como una prioridad, y para cumplir con este requerimiento durante el diseño, se incorporó la posibilidad de ajustar parámetros como *Spreading Factor* (SF), que afecta tanto el alcance como el tiempo de transmisión. La posibilidad de ajustar este parámetro permite evaluar el impacto energético y optimizar el rendimiento según el entorno, cumpliendo con el objetivo general del proyecto. A estos distintos escenarios se le analizará el consumo energético, lo cual se realizará mediante la implementación de un osciloscopio y una resistencia a la placa Tx con el propósito de analizar dicho comportamiento y orientar futuras decisiones de configuración.
- **Configurabilidad y adaptabilidad**: El sistema fue diseñado para permitir la modificación dinámica de parámetros como el SF, el *Bandwidth* (BW) y la *Coding*

Rate (CR). Esto se implementó a nivel de software en las placas LoRa ESP32, lo que permite adaptar el sistema a diferentes entornos (urbano o rural), evaluar el efecto de interferencias, y ajustar la configuración para lograr un balance entre eficiencia energética y desempeño. Los parámetros que se utilizarán se puede ver en la Tabla 3.1.

Tabla 3.1: Parámetros utilizados en mediciones.

	SF	BW (kHz)	CR
Medición 1	7	125	4/5
Medición 2	9	125	4/6
Medición 3	11	125	4/8

Los valores de SF, BW y CR fueron seleccionados para representar distintos compromisos entre tasa de transmisión, alcance y robustez del enlace. Un menor SF (como 7) permite una mayor velocidad de transmisión, pero con menor sensibilidad; mientras que un mayor SF (como 11) incrementa el alcance y la capacidad de resistir interferencias, a costa de un mayor tiempo de aire y consumo. El BW se mantuvo constante en 125 kHz por ser el valor típico para enlaces balanceados en LoRa, y se variaron las tasas de codificación (CR) para obtener confiabilidad del enlace.

- **Integración con plataformas IoT y formatos de salida**: El nodo transmisor fue programado para simular un *payload* de datos representativo de una aplicación IoT real, en formato JSON, mientras que la Raspberry Pi conectada al nodo receptor registra los datos recibidos y los almacena en formato CSV. Esto facilita su posterior análisis y su integración con plataformas de monitoreo o visualización, asegurando que el sistema pueda escalar y adaptarse a aplicaciones reales de IoT.
- **Capacidad de evaluación en terreno y documentación**: El diseño experimental incluye la conexión de la placa receptora a una Raspberry Pi, que actúa como estación de recepción y registro de datos. Esto permite realizar pruebas de campo en entornos urbanos y rurales, midiendo parámetros como RSSI y tasa de pérdida de paquetes. Además, se registró la distancia de cada prueba mediante una laptop conectada al nodo transmisor, permitiendo documentar y analizar los resultados de manera sistemática.

El esquema del setup experimental se muestra en la Fig.3.1. Este sistema está compuesto por una Laptop o PC que se conecta a una placa transmisora LoRa (Nodo TX) a través de un puerto USB para registrar la distancia en cada medición. Esta placa envía los datos mediante radiofrecuencia utilizando LoRa a otra placa receptora (Nodo RX), que a su vez está conectada a una Raspberry Pi, también por medio de USB. La Raspberry Pi actúa como estación de recepción y almacenamiento de datos.

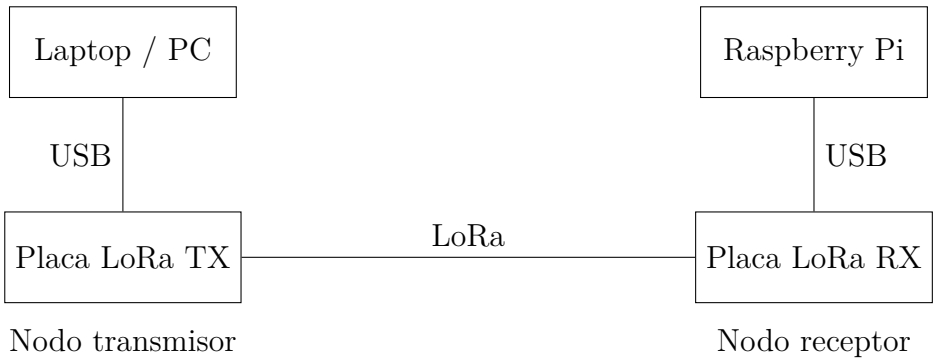

Fig. 3.1: Esquema del setup experimental para pruebas de conectividad con LoRa.

Si bien tecnologías como las redes celulares (3G, 4G, LTE-M) o la conectividad satelital pueden ofrecer cobertura en zonas de baja infraestructura, estas implican un costo operativo significativo [24]. El uso de tarjetas SIM o servicios satelitales conlleva un pago recurrente, lo que se vuelve económicamente inviable cuando se pretende implementar el sistema en cientos o miles de dispositivos, como suele ocurrir en soluciones IoT a gran escala. En cambio, LoRa opera en bandas de frecuencia no licenciadas, eliminando estos costos operativos y facilitando la escalabilidad del sistema.

3.3. Implementación

En base al diseño planteado en la sección anterior, se procedió a implementar el sistema utilizando una arquitectura compuesta por dos nodos: un nodo transmisor y un nodo receptor. El nodo transmisor está compuesto por una placa TTGO LoRa32 T3 ESP32 915 MHz, mientras que el nodo receptor se basa en una Raspberry Pi 4 de 8GB, la cual se comunica con una segunda placa LoRa idéntica conectada por USB.

La elección de la placa TTGO LoRa32 T3 ESP32 de la Fig. 3.2 se fundamenta principalmente en su integración de un microcontrolador ESP32 con conectividad Wi-

Fi, Bluetooth y un módulo LoRa de 915 MHz. Sin embargo, para este proyecto se optó por utilizar principalmente la tecnología LoRa, debido a sus ventajas en cuanto a alcance y eficiencia energética, las cuales son requisitos esenciales definidos en la sección 3.1, y utilizar la tecnología Wi-Fi de las placas para hacer un contraste de alcance y consumo energético. Tecnologías como Wi-Fi o Bluetooth, si bien están disponibles en la misma placa, presentan limitaciones importantes: Wi-Fi requiere infraestructura existente y tiene un alcance reducido, mientras que Bluetooth está diseñado para distancias cortas, generalmente inferiores a 100 metros [23]. En cambio, LoRa permite alcanzar distancias del orden de kilómetros con un consumo energético muy inferior, especialmente en aplicaciones de bajo ancho de banda como las del Internet de las Cosas (IoT).

La Raspberry Pi 4 de la Fi. 3.3 se escogió como nodo de recepción debido a su capacidad de procesamiento, su compatibilidad con interfaces USB para comunicación con la placa LoRa receptora, y su sistema operativo basado en Linux, que facilita la recopilación y análisis de datos.

Por otro lado, se decidió conectar la placa transmisora a un PC o laptop con el objetivo de registrar manualmente la distancia a la que se encontraba cada medición, evitando así la necesidad de incorporar un módulo GPS en el setup, lo que simplificó la implementación sin comprometer los objetivos del experimento.

En resumen, la implementación se diseñó para cumplir estrictamente con los requerimientos planteados: un sistema de bajo consumo energético, capaz de transmitir datos a larga distancia, adaptable a distintos escenarios y basado en tecnología económica y de libre uso. Alternativas como comunicación celular o satelital fueron descartadas debido a su elevado costo operacional, lo cual no es viable en sistemas IoT que pueden requerir el despliegue de cientos o miles de nodos[24]. La solución basada en LoRa representa entonces una alternativa eficaz, escalable y de bajo costo para entornos tanto rurales como urbanos con baja infraestructura de red.

Fig. 3.2: Placa LoRa.

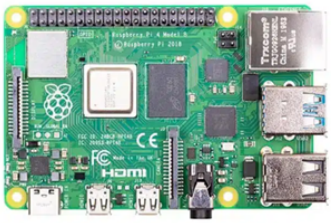

Fig. 3.3: Raspberry Pi.

3.3.1. Placa transmisora

La placa utilizada como nodo transmisor en este proyecto corresponde a una TTGO LoRa32 T3 ESP32 915 MHz, la cual integra dos componentes clave, el microcontrolador ESP32 y un módulo LoRa SX1276 como se puede ver en la Fig. 3.4 . Esta integración facilita el desarrollo de soluciones de comunicación de largo alcance y bajo consumo energético sin necesidad de incorporar múltiples módulos externos.

El ESP32 es un microcontrolador de doble núcleo con conectividad Wi-Fi y Bluetooth, ampliamente utilizado en aplicaciones de IoT por su bajo costo y versatilidad. Aunque ofrece conectividad Wi-Fi, en este proyecto se emplea únicamente para efectos comparativos, dado que la tecnología LoRa es más adecuada para alcanzar grandes distancias con bajo consumo energético, como se discutió en la sección 3.1.

El módulo LoRa SX1276, por otro lado, permite comunicaciones en la banda ISM de 915 MHz. Este chip emplea modulación de espectro ensanchado basada en chirp spread spectrum (CSS), lo que le confiere una alta sensibilidad y robustez frente a interferencias, y lo hace ideal para entornos con infraestructura limitada.

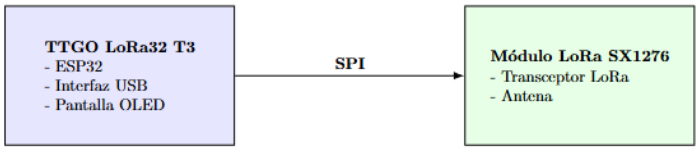

Fig. 3.4: Diagrama del hardware de placa tx.

El diagrama de la Fig. 3.5 muestra la conexión del microcontrolador ESP32 con

los componentes principales del sistema, destacando la interfaz SPI utilizada para comunicarse con el módulo LoRa SX1276. Los pines GPIO del ESP32 (SCK, MOSI, MISO y SS) están configurados para transmitir y recibir datos, mientras que los pines RST y DI0 gestionan el reinicio y las interrupciones del módulo LoRa, respectivamente. Además, el ESP32 se conecta a una pantalla OLED a través de la interfaz I2C (pines SCL y SDA) para mostrar información en tiempo real. El módulo LoRa está equipado con una antena externa para la transmisión y recepción de datos a través de tecnología LoRa. Finalmente, el ESP32 está conectado a un computador por USB, lo que permite su alimentación y comunicación en serie para la configuración y monitoreo del sistema.

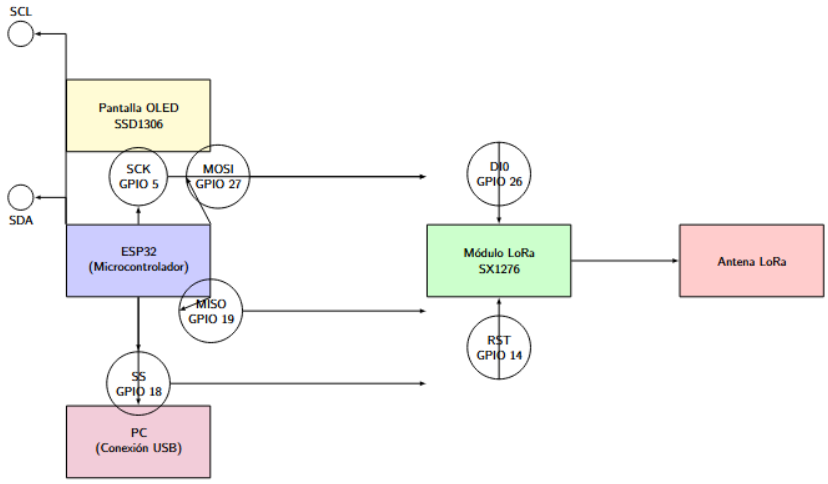

Fig. 3.5: Diagrama de interfaz entre ESP32 y módulo LoRa.

El código en el microcontrolador ESP32 se encarga de enviar paquetes de datos LoRa cada segundo. Estos paquetes simulan un payload típico de IoT, como si fueran datos sensados en terreno. En este caso, la distancia se ingresa manualmente a través de la consola serial USB, lo que permite emular mediciones sin necesidad de integrar un módulo GPS.

Cada paquete se transmite en formato JSON, lo que facilita su análisis posterior y la interoperabilidad con otros sistemas. Un ejemplo del formato enviado es:

```
21:30:30 -> Setting up LoRa...
21:30:30 -> LoRa pins set
21:30:30 -> LoRa initialized successfully!
21:30:30 -> LoRa configured OK!
21:30:30 -> Enter distance in meters and press Enter:
```

```
21:30:42 -> Setting distance to: 100.00 meters
21:30:42 -> Packet sent [1]: {"seq":0,"dist":10.00,"total":0,"timestamp":13537}
21:30:43 -> Packet sent [2]: {"seq":1,"dist":10.00,"total":1,"timestamp":14540}
```

El software permite modificar parámetros críticos de la transmisión LoRa, como el Spreading Factor (SF), el cual incide directamente en el alcance y el consumo energético del nodo. Durante las pruebas de campo, se modificará este parámetro para analizar su impacto en las métricas mencionadas, lo que permitirá tomar decisiones fundamentadas sobre la configuración óptima del sistema para diferentes escenarios (urbano o rural), en la Fig. 3.6 se puede ver como queda la placa tx con el software implementado.

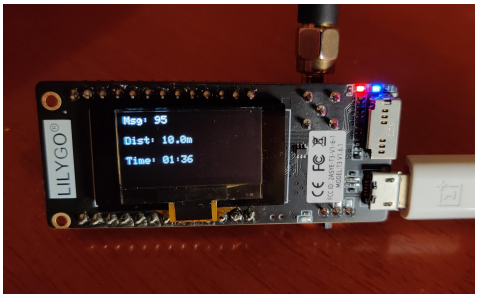

Fig. 3.6: Placa transmisora.

3.3.2. Placa receptora

El nodo receptor empleado en este proyecto es también una placa TTGO LoRa32 T3 ESP32 915 MHz, que integra un microcontrolador ESP32 y un módulo LoRa SX1276. Esta elección facilita la simetría en la implementación entre transmisor y receptor, y permite una integración eficiente de hardware y software en un mismo dispositivo.

Internamente, la placa cuenta con una interfaz de comunicación SPI (Serial Peripheral Interface) que conecta el microcontrolador ESP32 con el módulo LoRa SX1276, esta interfaz es la misma que en la placa transmisora como se puede ver en la Fig. 3.5.

El sistema también incluye una pantalla OLED conectada mediante la interfaz SDA y SCL que permite visualizar en tiempo real la información relevante de cada paquete recibido, como el identificador de secuencia, la distancia simulada, el RSSI, la tasa de pérdida de paquetes y el SNR. Esto es especialmente útil para realizar pruebas de campo sin necesidad de una conexión permanente a un computador.

A nivel de software, el receptor implementa una rutina de procesamiento de paquetes que extrae los campos del payload en formato JSON, tales como seq (número de secuencia), dist (distancia simulada), total (total de paquetes transmitidos), y timestamp, como se puede ver en la Fig. 3.7. Esta información se utiliza para calcular métricas clave como la pérdida de paquetes y para registrar el comportamiento de la comunicación en diferentes condiciones.

Esta arquitectura permite realizar un análisis detallado del comportamiento del enlace LoRa en distintos escenarios, midiendo indicadores como RSSI y tasa de pérdida de paquetes en función de la distancia y la configuración de los parámetros del enlace. La información recolectada por el receptor puede ser visualizada en tiempo real en la pantalla OLED y almacenada mediante la consola serial para su posterior análisis.

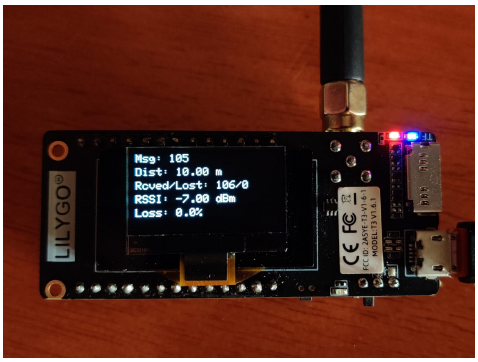

Fig. 3.7: Placa receptora.

3.3.3. Análisis de consumo energético del nodo transmisor

Con el objetivo de evaluar el impacto que tienen los parámetros de configuración LoRa especialmente el Spreading Factor (SF) sobre el consumo energético del nodo transmisor, se diseñó un setup experimental que permite medir la corriente consumida durante la transmisión de datos.

El esquema experimental utilizado se presenta en la Fig. 3.8. El sistema se alimenta mediante una batería portátil que entrega 5V a través de un cable USB. En serie con la alimentación se incorpora una resistencia de $0{,}5\,\Omega$, sobre la cual se mide la caída de voltaje utilizando un osciloscopio. Esta medición permite calcular la corriente mediante la ley de Ohm ($I = V/R$), y así estimar el comportamiento energético del sistema bajo distintas configuraciones.

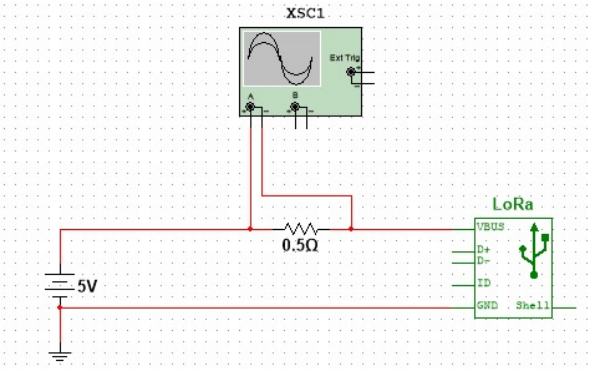

Fig. 3.8: Diagrama del setup para la medición del consumo energético.

Los resultados de estas mediciones se analizarán en el Capítulo 4, donde se comparará el consumo energético asociado a distintos valores de SF y a la utilización de tecnología Wi-Fi.

3.3.4. Raspberry Pi

La Raspberry Pi 4 utilizada en este proyecto cumple el rol de nodo receptor final, encargado de registrar y almacenar los datos recibidos desde el módulo LoRa. Estos datos son enviados por la placa transmisora, capturados por la placa receptora (ESP32 con módulo LoRa) y finalmente transferidos vía puerto serial a la Raspberry Pi para su almacenamiento.

El registro de datos se realiza en archivos .csv (valores separados por comas), los cuales permiten organizar la información de manera estructurada para su posterior análisis. Cada archivo CSV corresponde a una sesión de pruebas a una determinada distancia, y su nombre incluye tanto la distancia como una marca temporal, por ejemplo: `lora_data_10.00m_2025-05-01_12-30-15.csv`

A modo de ejemplo, la Tabla 3.2 muestra el formato y contenido típico del archivo CSV correspondiente a una sesión de pruebas a 10 metros de distancia.

Timestamp	Event	MessageNumber	Distance (m)	Received	Lost	RSSI	SNR	TotalPackets	LossRate
21:51:18	DATA	0	10	1	0	-63	9.75	1	0
21:51:19	DATA	1	10	2	0	-63	10.5	1	0
21:51:20	DATA	2	10	3	0	-63	10.0	2	0

Tabla 3.2: Datos de transmisión LoRa a 10 metros de distancia.

Capítulo 4

Resultados

4.1. Análisis de cobertura

Se realizaron campañas de medición en dos entornos contrastantes: una zona urbana y una zona rural. En ambos casos, se efectuaron pruebas a distintas distancias entre el nodo transmisor y el nodo receptor, con el objetivo de evaluar el desempeño del sistema en función de la propagación de la señal en diversos escenarios y analizar su alcance. Durante las mediciones, se configuró el sistema para transmitir paquetes de datos con una periodicidad de un segundo.

Asimismo, en cada ubicación se llevaron a cabo mediciones variando el parámetro de SF, CR y el parámetro BW se mantuvo constante, con el fin de analizar su impacto en el alcance, la fiabilidad de la comunicación y la calidad del enlace. Por cada entorno medido se mostrarán gráficos en donde se representa el RSSI de cada medición y el porcentaje de paquetes perdidos en cada una de las distancias.

4.1.1. Zona urbana

La zona urbana elegida para la medición fue la ciudad de Concepción, específicamente en la calle Camilo Henríquez, en la Fig. 4.1 se puede ver las calles en las que se tomó las mediciones. Se tomaron distintas mediciones a 10m, 50m, 100m, 250m, 500m y 1000m. Esta última medición fue la máxima distancia que se logró transmitir paquetes.

Fig. 4.1: Mediciones en zona urbana.

En cada medición se enviaron paquetes durante un minuto aproximadamente, por cada medición realizada se crea un archivo Excel en donde se registra la distancia medida, los paquetes enviados, los paquetes recibos, los paquetes perdidos, RSSI, SNR, y porcentaje de paquetes perdidos. En la Fig. 4.2 se puede ver un ejemplo del archivo Excel que se crea por cada medición.

Timestamp	Event	MessageNumber	Distance(m)	Received	Lost	RSSI	SNR	TotalPackets	LossRate
2025-03-24 21:51:18	DATA	0	10	1	0	-63	9,75	1	0
2025-03-24 21:51:19	DATA	1	10	2	0	-63	10,5	2	0
2025-03-24 21:51:20	DATA	2	10	3	0	-63	10	3	0
2025-03-24 21:51:21	DATA	3	10	4	0	-63	9,75	4	0
2025-03-24 21:51:22	DATA	4	10	5	0	-63	9,75	5	0
2025-03-24 21:51:23	DATA	5	10	6	0	-64	10,25	6	0
2025-03-24 21:51:24	DATA	6	10	7	0	-61	10	7	0
2025-03-24 21:51:25	DATA	7	10	8	0	-64	9,75	8	0

Fig. 4.2: Ejemplo de los datos de una medición.

Para realizar una comparación de los resultados con un modelo de pérdidas se seleccionó el modelo Okumura-Hata [25]. La selección de este modelo para el análisis de señales en este proyecto se fundamenta en su reconocida aplicabilidad para estimar la pérdida de propagación en sistemas de comunicación inalámbrica de largo alcance. Su capacidad para ofrecer estimaciones confiables tanto en entornos urbanos como rurales lo convierte en una herramienta adecuada para comparar las condiciones teóricas de propagación con las mediciones empíricas realizadas durante las pruebas de campo.

Si bien este modelo presenta una mayor precisión para rangos de distancia superiores a 1 km y situaciones donde la antena transmisora se ubica a alturas elevadas respecto de la receptora, se consideró igualmente pertinente su uso en este proyecto dada la

necesidad de contar con una referencia teórica sólida para contrastar los resultados experimentales.

Cabe señalar que las condiciones específicas del sistema, con ambas antenas situadas a una altura de aproximadamente 1.60 metros, y las distancias de medición que en algunos casos fueron menores a 1 km, no se ajustan de manera óptima al rango de aplicaciones típico del modelo Okumura-Hata. No obstante, ante la ausencia de parámetros precisos sobre la ganancia real de las antenas utilizadas y considerando que este modelo permite cierto grado de adaptación, se procedió a realizar ajustes en la curva teórica generada. En particular, se consideró una altura de transmisor de 30 metros en la parametrización del modelo ya que esta altura está dentro del rango en donde el modelo funciona de manera óptima y es una altura utilizada en aplicaciones reales[26], buscando con ello compensar parcialmente la diferencia de altura respecto a las condiciones ideales del modelo original.

Adicionalmente, se aplicó un desplazamiento (offset) a la curva del modelo de pérdida, con el propósito de centrarla en torno a las medias de las mediciones obtenidas experimentalmente. Esta corrección fue necesaria para alinear la tendencia teórica del modelo con los datos empíricos, debido a la incertidumbre existente en las características exactas del sistema de antenas empleado, como la mencionada anteriormente. De esta manera, si bien se reconoce que la aplicación del modelo Okumura-Hata no es perfectamente representativa de las condiciones del experimento, los ajustes introducidos permitieron obtener una referencia válida que facilitó el análisis comparativo de los resultados de campo y la validación general del comportamiento del enlace LoRa evaluado.

La fórmula utilizada para el modelo Okumura-Hata en zona urbana es la siguiente:

$$L_u = 69{,}55 + 26{,}16\log_{10}(f) - 13{,}82\log_{10}(h_1) - a(h_2) + (44{,}9 - 6{,}55\log_{10}(h_1))\log_{10}(d) \tag{4.1}$$

Donde:

- f: Frecuencia de operación, válida entre 150 MHz y 1500 MHz.
- d: Distancia entre el transmisor y receptor, entre 1 km y 20 km.

- h_1: Altura efectiva de la antena transmisora, entre 30 m y 200 m.
- h_2: Altura efectiva de la antena receptora, entre 1 m y 10 m.

En este caso, como fue una ciudad grande la elegida para hacer las mediciones en zona urbana, el factor de corrección para frecuencias mayores o iguales a 300 MHz fue:

$$a(h_2) = 3{,}2\left[\log_{10}(11{,}75h_2)\right]^2 - 4{,}97 \tag{4.2}$$

En la Fig. 4.3 se ven los valores de RSSI registrados en las distintas mediciones tomadas con un SF de 7. En la figura se observa que el RSSI decrece de manera irregular a medida que aumenta la distancia. Particularmente, entre 250m y 500m, la caída de señal es pronunciada, lo que refleja la presencia de obstáculos urbanos que generan sombras de señal. Este comportamiento evidencia la alta sensibilidad del entorno urbano a interferencias y pérdidas por multitrayectoria, afectando la estabilidad del enlace. La caída de señal sigue una tendencia logarítmica típica de entornos urbanos densos y como SF 7 es menos tolerante a la atenuación, limita el alcance efectivo por lo que no es recomendado para despliegues urbanos si se busca cobertura extendida, es más útil en nodos cercanos que priorizan la velocidad.

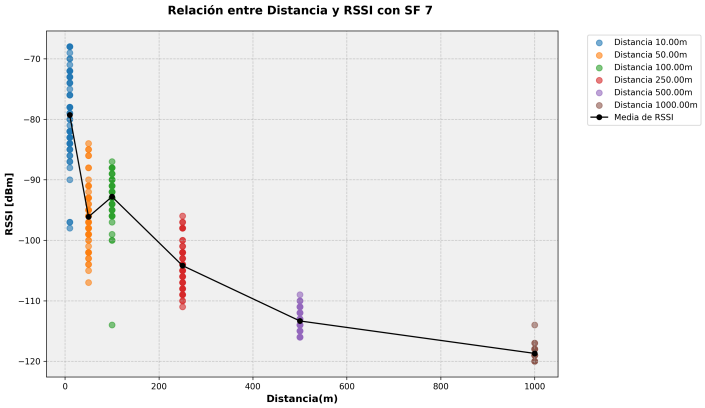

Fig. 4.3: Relación entre distancia y RSSI con SF 7.

Al analizar las mediciones reales de RSSI para SF 7 en el entorno urbano vs la curva teórica en la Fig. 4.4, la curva teórica muestra una atenuación progresiva conforme aumenta la distancia, ajustada con un offset para compensar la ganancia desconocida de las antenas. Los valores experimentales de RSSI presentan una dispersión moderada, especialmente en distancias más largas y en las distancias cortas una mayor dispersión

en donde el modelo tiende a subestimar la pérdida de señal. Esto se atribuye a las limitaciones del modelo en distancias cortas y a la interferencia urbana (edificios, obstáculos, etc.). A partir de 1 km, la alineación mejora, aunque persisten variaciones debido a la altura idéntica de las antenas, contrarias a las condiciones ideales del modelo. El SF 7, al priorizar velocidad sobre alcance, refleja una mayor sensibilidad a obstrucciones, acentuando las diferencias en entornos urbanos densos.

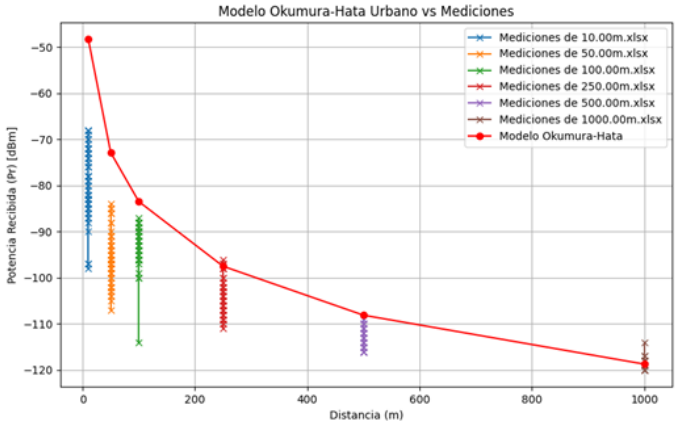

Fig. 4.4: Modelo Okumura-Hata vs mediciones.

En la Fig. 4.5 se ven los valores de RSSI registrados en las distintas mediciones tomadas con un SF de 9. Con un Spreading Factor de 9, la atenuación del RSSI sigue un patrón similar, pero muestra una leve mejora respecto al SF 7, especialmente entre los 500m y lo 1000m. Esto demuestra que el aumento del SF contribuye a mejorar la robustez de la comunicación en ambientes urbanos, aunque no elimina completamente las caídas abruptas ocasionadas por obstáculos densos. La mayor duración del símbolo de SF 9 permite mejorar la sensibilidad del receptor. Estos resultados muestran que con SF 9 se tiene un buen equilibrio para entornos urbanos con requerimientos de mayor cobertura sin necesidad de datos en tiempo real.

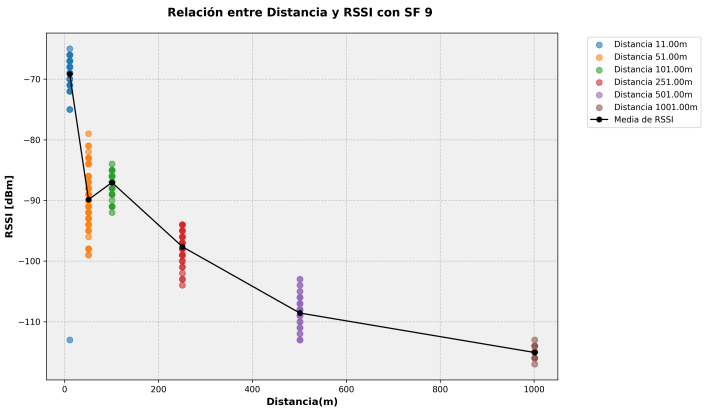

Fig. 4.5: Relación entre distancia y RSSI con SF 9.

Para la comparación de RSSI en SF 9 vs curva del modelo se puede apreciar la Fig. 4.6, en este caso, el modelo teórico se aproxima mejor a las mediciones, particularmente en el rango de 500 m y 1 km. El SF 9, al equilibrar alcance y robustez, reduce la variabilidad del RSSI en comparación con SF 7. Sin embargo, en distancias cortas (menores a 500 metros), los valores experimentales muestran una atenuación mas pronunciada que la predicha, posiblemente por efectos de multitrayectoria o absorción por estructuras cercanas. El offset aplicado logra una coincidencia aceptable en la tendencia general, aunque la falta de diferencia de diferencia en la altura de las antenas introduce un error sistemático.

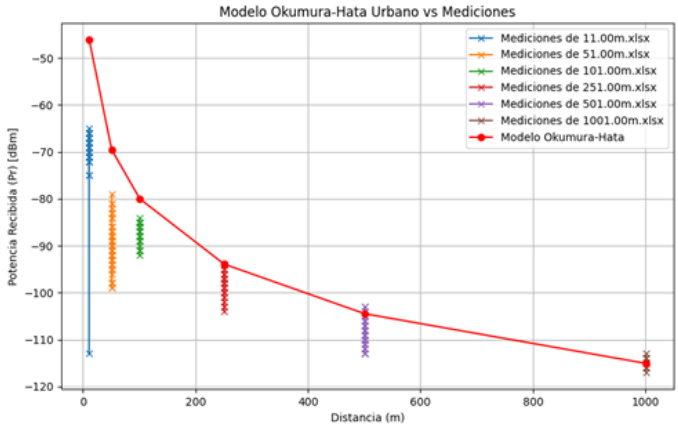

Fig. 4.6: Modelo Okumura-Hata vs mediciones.

En la Fig. 4.7 se ven los valores de RSSI registrados en las distintas mediciones tomadas con un SF de 11. El gráfico evidencia que el uso de SF 11 permite extender la cobertura hasta el máximo de 1000m medido, logrando una mejor recepción de señal

en comparación con SF menores. Sin embargo, se mantienen las oscilaciones de señal a distancias intermedios, lo que sugiere que, pese a la mejora, el entorno urbano sigue siendo desafiante para la propagación de la señal. De los tres parámetros utilizados este es el que muestra una mejor robustez ya que el incremento del tiempo en el aire de cada paquete provoca que incremente la inmunidad a ruido, pero reduce la tasa de transmisión. Lo que lo hace ideal para sensores que requieran bajo consumo y gran alcance, como monitoreo ambiental o infraestructura urbana crítica.

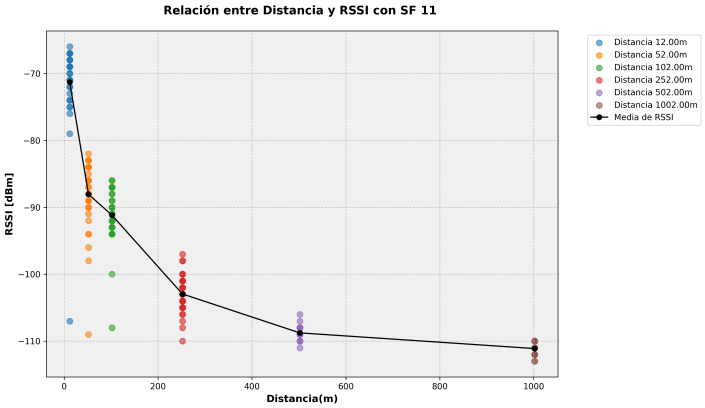

Fig. 4.7: Relación entre distancia y RSSI con SF 11.

Finalmente, en SF 11 la comparación con la curva teórica y las mediciones reales se puede ver en la Fig. 4.8, este SF está diseñado para máxima resistencia y alcance, lo que resulta en una curva experimental con menor dispersión y una atenuación mas gradual. El modelo teórico coincide con la medición a 1km de manera muy fidedigna, validando su aplicabilidad en entornos urbanos para largas distancias, ya que este modelo funciona de forma representativa a distancias superiores a 1 km, en donde a mediciones menores a esta distancia los valores de RSSI experimentales son mas bajos que el previsto, evidenciando que el modelo no captura pérdidas adicionales por obstáculos a corto alcance.

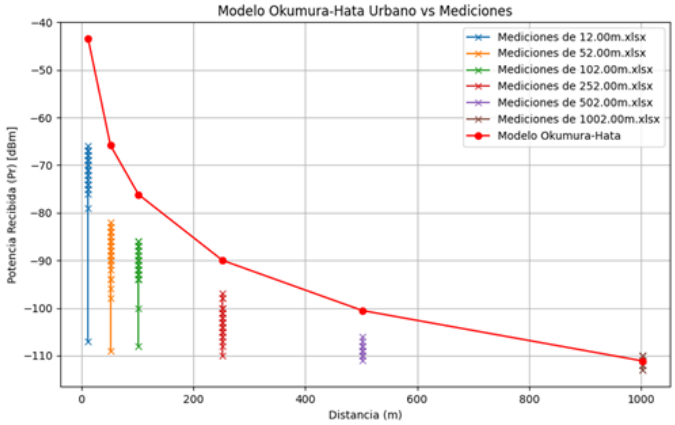

Fig. 4.8: Modelo Okumura-Hata vs mediciones.

Para las distintas mediciones realizadas en zona urbana para los 3 distintos SF, se registraron los porcentajes de paquetes perdidos que se pueden ver en la Fig. 4.9. Se aprecia un incremento progresivo de la pérdida de paquetes conforme aumenta la distancia, con picos pronunciados a partir de los 500m. Este comportamiento se correlaciona directamente con la caída de RSSI, y refleja la complejidad de la propagación en áreas urbanas. Confirma que RSSI bajo -120 dBm incrementa las probabilidades de error en la capa física. En despliegues urbanos se debe considerar la densidad de nodos para mantener tasas de éxito aceptables.

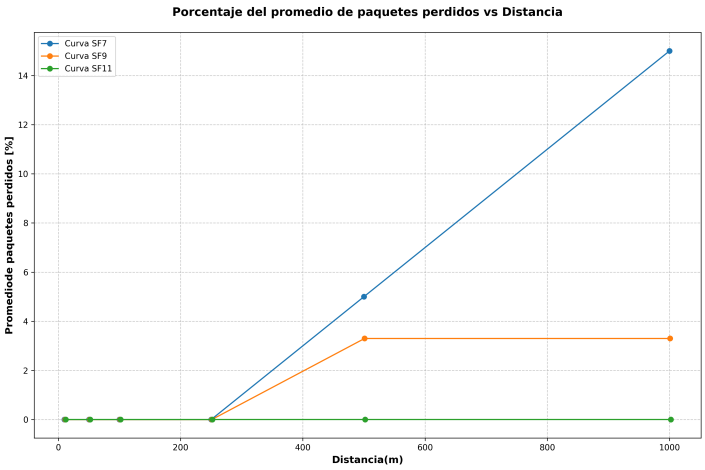

Fig. 4.9: Relación entre el porcentaje del promedio de paquetes perdidos y distancia.

4.1.2. Zona rural

La zona rural elegida para llevar a cabo las mediciones fue en un camino rural en las cercanías de Purén, novena región. En la Fig. 4.10 se puede ver el sector elegido, el recorrido se extiende en un entorno abiertos, minimizando obstáculos. Se tomaron mediciones en las siguientes distancias, 10m, 50m, 100m, 250m, 500m, 1000m, 1500m, 2000m, 2500m, 3000m, 3500m y 4000m.

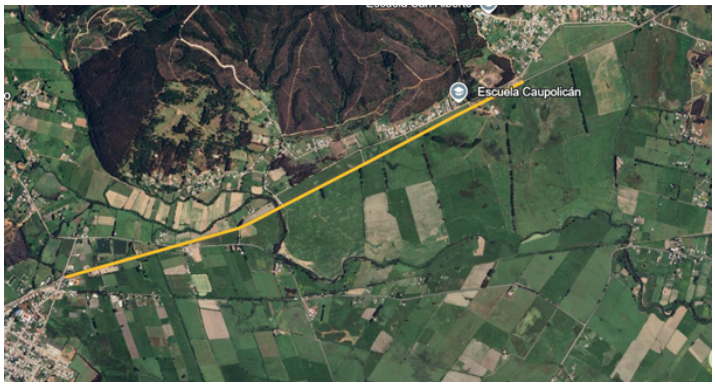

Fig. 4.10: Mediciones en zona rural.

Para la zona rural, la ecuación del modelo Okumura-Hata que se utilizó fue la siguiente:

$$L_r = L_u - 4{,}78(\log_{10} f)^2 + 18{,}33 \log_{10} f - 40{,}94 \tag{4.3}$$

Al igual que en la zona urbana, por cada medición se creó un archivo Excel en donde se registraban todos los parámetros importantes para cada medición. En la Fig. 4.11 se puede ver el gráfico con los distintos valores de RSSI de las mediciones con el parámetro SF 7, con este parámetro se logró enviar paquetes a una distancia máxima de 3000m. El gráfico muestra una degradación del RSSI mucho más suave y predecible en comparación con la zona urbana, aquí extiende su alcance útil hasta 2000m aproximadamente antes de aproximarse a un valor de -120dBm. La propagación directa reduce efectos de multitrayecto y atenuación severa, y se puede apreciar que SF 7 es viable en ambientes rurales para aplicaciones que requieran tasas de datos más altas, ya que, con menor densidad de obstáculos, la propagación de la señal es más eficiente.

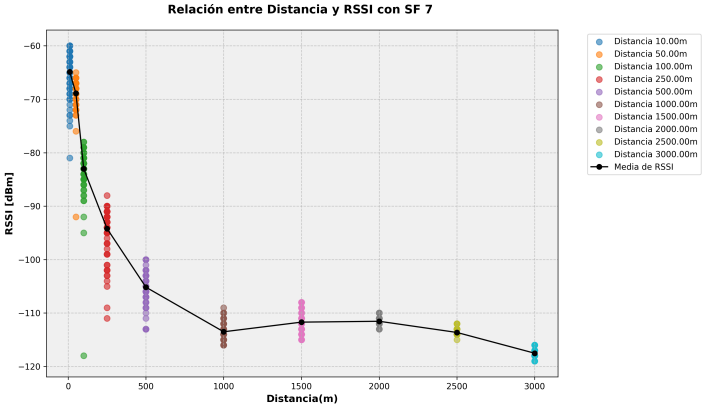

Fig. 4.11: Relación entre distancia y RSSI con SF 7.

En los gráficos de la zona rural, la comparación entre la curva teórica de pérdida y las mediciones reales de RSSI se puede ver en la Fig. 4.12, en el entorno rural, la curva teórica se alinea mejor con los datos experimentales, incluso en distancias cortas, debido a la menor presencia de obstáculos. El SF 7, aunque menos robusto, muestra una correlación aceptable hasta 1 km, donde el modelo comienza a subestimar ligeramente la atenuación. El offset aplicado compensa eficazmente la ganancia desconocida, logrando que la curva pase por el centro de las mediciones. Para distancias superiores a 1 km, la coincidencia mejora, reforzando la validez del Okumura-Hata en áreas rurales extensas.

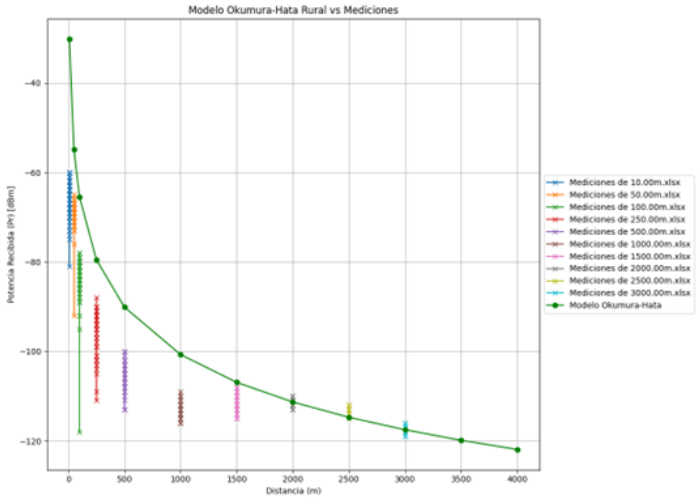

Fig. 4.12: Modelo Okumura-Hata vs mediciones.

En la Fig. 4.13 se puede ver los valores de RSSI para las mediciones con el parámetro de SF 9, con este parámetro la distancia máxima a la que se pudo enviar paquetes fue

3500m. La pendiente del descenso de RSSI es moderada y constante, validando que el incremento del SF tiene un impacto positivo en la cobertura en ambientes abiertes, sin los efectos abruptos observados en zona urbana. La mejora es significativa respecto a SF 7 del gráfico anterior, la sensibilidad extra de SF 9 permite extender la cobertura sin comprometer excesivamente la tasa de datos.

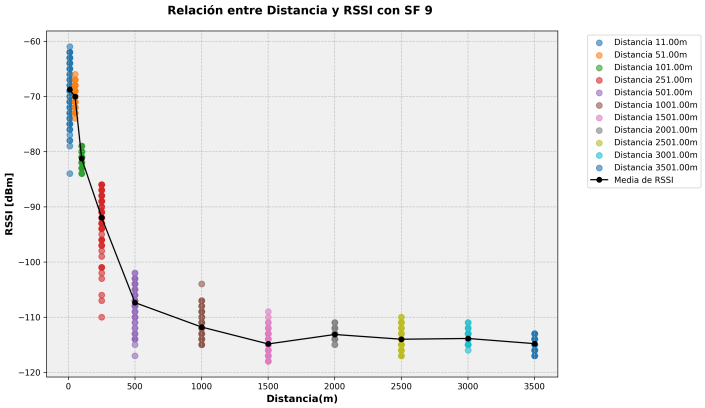

Fig. 4.13: Relación de distancia y RSSI para SF 9.

El SF 9 en zona rural exhibe una convergencia notable entre el modelo y los resultados reales, sobre todo en distancias más largas, como se puede ver en la Fig. 4.14. La atenuación teórica y experimental siguen patrones similares, con desviaciones mínimas atribuidas a los parámetros propios del modelo para predecir de manera efectiva y los efectos ambientales de la zona.

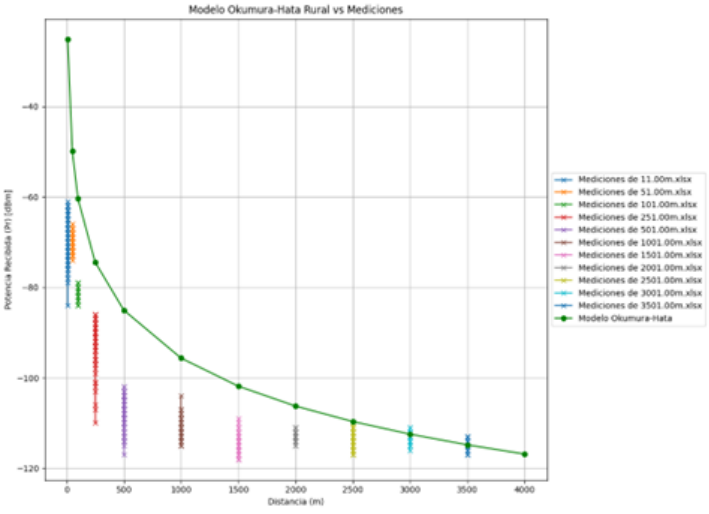

Fig. 4.14: Modelo Okumura-Hata vs mediciones.

En la Fig. 4.15 están representados los valores de RSSI que se obtuvieron en las mediciones con el parámetro SF 11, con este parámetro la máxima distancia que se logró fue de 4000m. Este resultado destaca la eficiencia de este parámetro en maximizar la cobertura rural, especialmente para aplicaciones IoT de monitoreo remoto que requieran gran alcance.

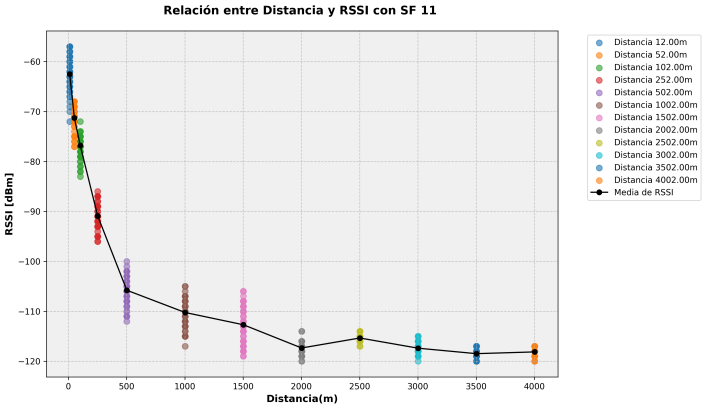

Fig. 4.15: Relación de distancia y RSSI para SF 11.

En el caso de SF 11, los valores de RSSI en comparación con la curva del modelo se pueden ver en la Fig. 4.16, en esta oportunidad se muestra la mejor correlación entre todos los casos analizados. El modelo ajustado se superpone casi perfectamente a las mediciones de distancias mas largas, validando una vez mas que el modelo funciona de forma representativa para distancias mayores a 1 km. A pesar de eso, en mediciones con distancias mas cortas sigue siendo una buena aproximación de la predicción teórica, demostrando que el modelo utilizado es una herramienta confiable para hacer predicciones de pérdida de potencia.

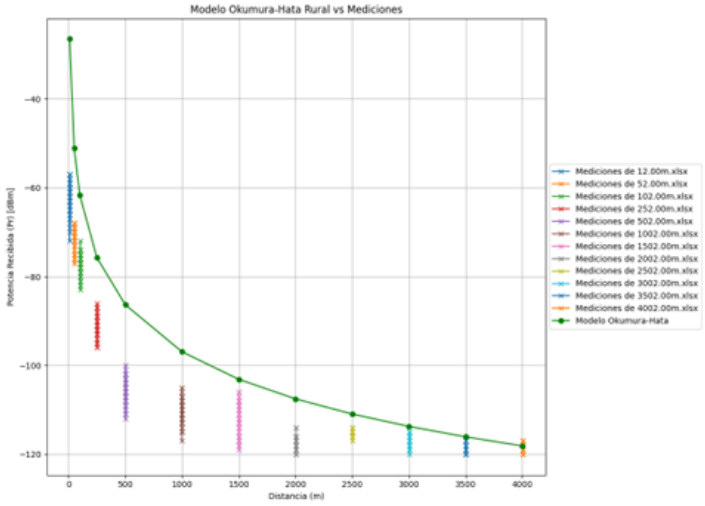

Fig. 4.16: Modelo Okumura-Hata vs mediciones.

Para las mediciones que se hicieron en los distintos parámetros se graficaron los porcentajes de los promedios de paquetes perdidos para cada medición, los cuales se pueden ver en la Fig. 4.17. El gráfico evidencia un comportamiento favorable en zona rural, con pérdidas de paquetes muy bajas hasta aproximadamente los 2000m a partir de este punto, se incrementa gradualmente, pero se mantiene mejor que en zona urbana ya que la menor cantidad de obstáculos favorece la integridad de los paquetes confirmando la viabilidad de despliegues rurales extendidos para aplicaciones IoT.

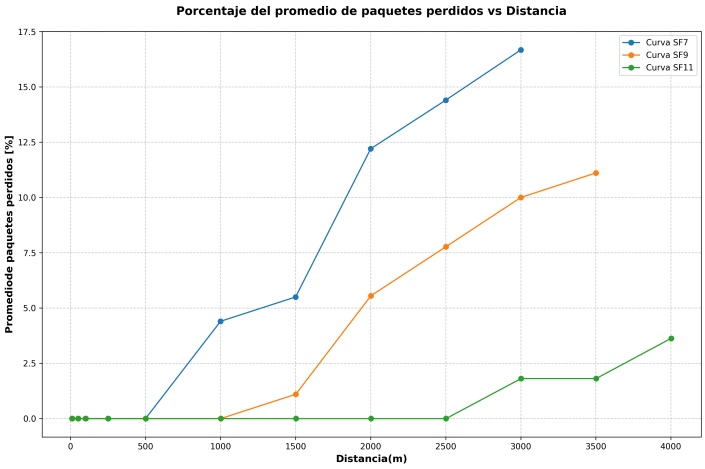

Fig. 4.17: Relación entre el porcentaje del promedio de paquetes perdidos y distancia.

4.1.3. Comparación con Wi-Fi

Para realizar una comparación del alcance de la tecnología LoRa, y comprobar lo que se mencionó en el capítulo anterior sobre alcances en la transmisión, se realizó una medición con la tecnología Wi-Fi la cual viene incorporada en las placas, para analizar cual era el máximo alcance que obtenia con esta tecnología, ya que según la teoría es inferior a 100m.

Para realizar esto se usaron los mismo códigos utilizados en LoRa, solo que ahora se utiliza el chip ESP32 para hacer uso del Wi-Fi, y realizando la medición se llegó a una distancia máxima de 124.7 m como se puede ver en la Fig. 4.18 y 4.19, después de esto las placas perdían conexión. Podemos notar que la ditancia que obtuve es mayor que la teórica sin embargo, sigue siendo drásticamente menor que la obtenida con tecnología LoRa, ya que, con LoRa aún en el peor escenario, que es en urbano con un SF 7, se logró transmitir datos a una distancia máxima de 1km. Mostrando y comprobando que la tecnología elegida es la adecuada para transmtir datos a larga distancia y supera otras tecnologías como Wi-Fi.

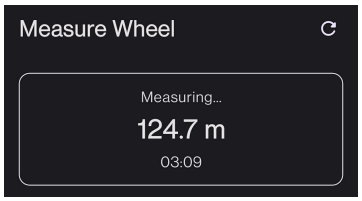

Fig. 4.18: Distancia máxima medida.

Fig. 4.19: Distanica medida con google Earth.

4.1.4. Resumen de resultados

A continuación, se presenta un resumen consolidado de los resultados obtenidos durante las pruebas de cobertura realizadas en distintas zonas geográficas, utilizando tecnología LoRa con factores de expansión (Spreading Factor, SF) variables, y una comparación con la cobertura alcanzada mediante tecnología WiFi.

En el entorno urbano, la cobertura registrada fue de 1 kilómetro para todos los valores de SF utilizados (SF 7, SF 9 y SF 11). Este comportamiento uniforme sugiere que,

en contextos con alta densidad de edificaciones, interferencias y obstáculos físicos, las ventajas teóricas de utilizar mayores valores de SF se ven limitadas por las condiciones del entorno, restringiendo el alcance efectivo de la señal.

En contraste, en la zona rural se evidenció un comportamiento más acorde con las propiedades esperadas del protocolo LoRa. A medida que se incrementó el SF, se observó una mejora progresiva en la distancia de cobertura: 3 kilómetros con SF 7, 3,5 kilómetros con SF 9 y hasta 4 kilómetros con SF 11. Este incremento se explica por el aumento en la sensibilidad de recepción que permite alcanzar mayores distancias.

Por otro lado, la tecnología WiFi demostró un alcance considerablemente más limitado, con una cobertura máxima medida de 124,7 metros. Esta diferencia resalta la ventaja comparativa de LoRa en aplicaciones que requieren comunicaciones de largo alcance, especialmente en entornos rurales o de baja infraestructura. Ya que con Wi-Fi se obtuvo menos del 80 % de cobertura que se obtuvo en urbano, en la Tabla 4.1 se puede ver claramente esta comparación.

Tabla 4.1: Distancia máxima alcanzada según tecnología y configuración de SF

Zona	**Tecnología**	**Spreading Factor (SF)**	**Distancia alcanzada (m)**
Urbana	LoRa	7	1000
		9	1000
		11	1000
Rural	LoRa	7	3000
		9	3500
		11	4000
Urbana	Wi-Fi	–	124.7

En resumen, los resultados experimentales corroboran la idoneidad de LoRa como tecnología de transmisión para aplicaciones del Internet de las Cosas (IoT) en escenarios donde la cobertura es una de las prioridades, particularmente en zonas rurales por sobre otras tecnologías más comunes como Wi-Fi.

4.2. Análisis de consumo energético

Con el objetivo de evaluar el consumo energético asociado a cada transmisión, se realizaron mediciones experimentales utilizando un osciloscopio digital y una resistencia de 0,5 ohm conectada en serie con la alimentación de la placa transmisora. Esta configuración permitió obtener el perfil de corriente consumida durante el envío de datos para distintos factores de expansión (SF 7, SF 9 y SF 11) en tecnología LoRa, así como para una transmisión equivalente utilizando tecnología Wi-Fi. Las señales capturadas permitieron determinar el valor de corriente durante cada pulso de transmisión, así como la duración de dichos pulsos, lo que facilitó el cálculo de la energía consumida por cada paquete enviado. Posteriormente, con base en estos datos, se realizó una estimación del rendimiento energético en un escenario hipotético que considera el uso de una batería de capacidad conocida, con el fin de comparar la autonomía potencial que ofrecería cada tecnología bajo condiciones similares. Este análisis resulta fundamental para aplicaciones IoT en las que la eficiencia energética y la duración de la batería son criterios críticos de diseño, particularmente en dispositivos desplegados en ubicaciones remotas o de difícil acceso.

Cabe mencionar que estas estimaciones consideran únicamente el consumo energético asociado al proceso de transmisión de datos, excluyendo el consumo basal de la placa en reposo o en otras etapas del ciclo de operación. Esta delimitación metodológica fue aplicada de forma consistente en todas las configuraciones evaluadas (SF 7, 9, 11 y Wi-Fi), lo que permite obtener una comparación relativa representativa entre las distintas tecnologías y parámetros analizados, manteniendo la validez del análisis energético realizado.

4.2.1. Consumo energético de LoRa

En las ondas cuadradas a continuación se puede apreciar la duración del pulso y el valor de voltaje en cada transmisión. En donde se puede apreciar que para la transmision con SF 7 se obtuvo un valor de Ancho Pos (duración del pulso de transmisión) de 106.8 ms Fig. 4.20, para un SF 9 se obtuvo 394.2 ms Fig. 4.21 y para SF 11 un tiempo de 1.708 s Fig. 4.22. Por esta razón, se optó por establecer un intervalo de transmisión de 1 segundo para los casos con SF 7 y SF 9, mientras que para SF 11, debido a su mayor

duración de pulso, se definió un intervalo de 2 segundos entre cada transmisión, con el fin de evitar solapamientos y asegurar una evaluación precisa del consumo energético por paquete enviado. Y en todas las mediciones el mismo valor de Vpico-pico de 74.0 mV.

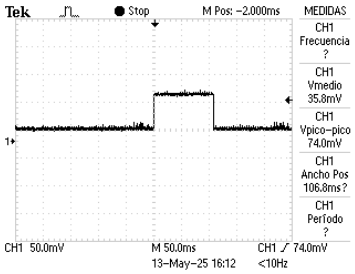

Fig. 4.20: SF 7.

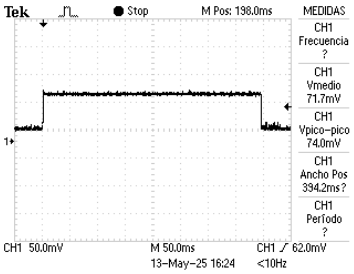

Fig. 4.21: SF 9.

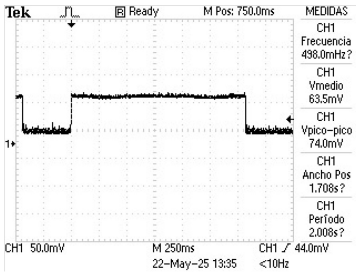

Fig. 4.22: SF 11.

A partir del voltaje medido en los pulsos de transmisión (señales cuadradas), se estimó la corriente usando la Ley de Ohm, esta corriente calculada coincide con el valor medido a traves de un multímetro:

$$I = \frac{V_{pp}}{R} = \frac{74{,}0\,\text{mV}}{0{,}5\,\Omega} = 148\,\text{mA} \tag{4.4}$$

Por lo tanto, se tienen los valores de la Tabla 4.2 para cada transmisión.

Tabla 4.2: Parámetros de transmisión para distintos valores de SF.

SF	$\boldsymbol{V_{pp}}$ **(mV)**	**Duración (s)**	**Corriente (A)**
7	74.0	0.1068	0.148
9	74.0	0.3942	0.148
11	74.0	1.708	0.148

Luego, con la corriente estimada y el tiempo de duración del pulso de transmisión (medido como Ancho Pos del pulso), se calculó la energía consumida por cada transmisión usando la fórmula:

$$E = V \cdot I \cdot t \tag{4.5}$$

Donde:

- $V = 5\,\text{V}$: Tensión de operación.
- $I = 0{,}148\,\text{A}$: Corriente durante la transmisión.
- t: Duración de la transmisión medida con el osciloscopio.

Por ende, reemplazando la fórmula 4.5 con los parámetros de cada caso se obtienen los siguientes valores de energía consumida:

$$\text{Para SF 7:} \qquad E = 5 \cdot 0{,}148 \cdot 0{,}1068 = 79{,}0\ \text{mJ} \tag{4.6}$$

$$\text{Para SF 9:} \qquad E = 5 \cdot 0{,}148 \cdot 0{,}3942 = 291{,}7\ \text{mJ} \tag{4.7}$$

$$\text{Para SF 11:} \qquad E = 5 \cdot 0{,}148 \cdot 1{,}708 = 1261{,}9\ \text{mJ} \tag{4.8}$$

Como se puede ver, el consumo energético por transmisión aumenta significativamente conforme se aumenta el SF. Este fenómeno está relacionado con la mayor cantidad de datos que deben ser enviados, lo que requiere una mayor duración de la transmisión. Un SF más alto significa que se envían más símbolos de manera más lenta, aumentando la cantidad de energía necesaria para completar la transmisión. Este aumento en el consumo energético es una consecuencia directa de la mayor robustez de la señal proporcionada por un SF más alto. LoRa permite un alcance mayor y una mayor resistencia a interferencias, pero a costa de un mayor consumo energético. Esta relación es crucial en aplicaciones IoT, donde el consumo de energía es un factor limitante, especialmente en dispositivos alimentados por baterías.

Este resultado destaca la importancia de seleccionar cuidadosamente el Spreading Factor, especialmente en aplicaciones energéticamente restringidas como sensores IoT alimentados por baterías.

Para tener una diferenica mas clara de como se ve la transmision a lo largo de un tiempo, se pueden apreciar las siguientes imagenes en donde el osciloscopio tiene ajustado el parámetro M a 1s. Se pude ver la clara diferencia del ancho de la onda cuadrada en donde cada una de estas representa al envío del paqquete, se refleja lo que se confirmó recientemente sobre el consumo energético, entre más alto el SF máyor será el tiempo que está en corriente alta, esto se va reflejando en como se va ensanchando la onda cuadrada, en SF 11 dura más tiempo en transmitir los paquetes por ende, transmite menos paquetes que en los demás valores de SF. En promedio el SF 11 transmite aproximadamente un 50 % menos cantidad de paquetes en comparación con SF 7 y

9, ya que como se aprecia en las imagenes con SF de 7 Fig. 4.23 y 9 Fig. 4.24 se transmitieron 10 paquetes en 10 segundos y con SF 11 Fig. 4.25 solo 5 paquetes en 10 segundos, ya que el tiempo de transmisión era el doble.

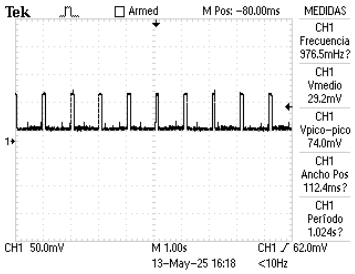

Fig. 4.23: SF 7.

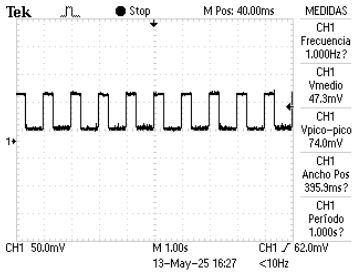

Fig. 4.24: SF 9.

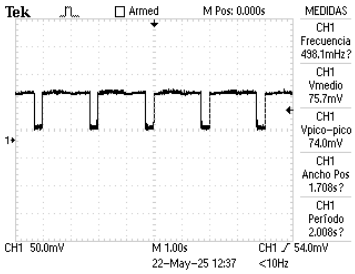

Fig. 4.25: SF 11.

Ahora para calcular la autonomia de una batería con tecnología LoRa vamos a suponer que utilizamos una batería de litio común de 3.7 V con una capacidad de 2000 mAh (una capacidad bastante común para dispositivos IoT [27]). Y se realizará el calculo suponiendo distintos intervalos de tiempo, especificamente a 1, 5 y 15 minutos. Cabe mencionar que, si bien las mediciones de consumo energético fueron realizadas con una fuente de 5V, el análisis hipotético se realizó considerando una batería de 3.7V y 2000 mAh (26,640 J). Se asume que el sistema mantiene un comportamiento similar en términos de consumo energético por transmisión, considerando que la placa incorpora un regulador interno para operar con dicha tensión

Para calcular la energía total disponible de la batería hay que considerar que la batería es de 3.7 V y 2000 mAh = 2 Ah, por lo que la formula de energía quedaría de la siguiente forma:

$$E_{\text{batería}} = V \cdot I \cdot t = 3{,}7\,\text{V} \cdot 2\,\text{Ah} = 7{,}4\,\text{Wh} \tag{4.9}$$

Sabemos que:

$$1\,\text{Wh} = 3600\,\text{J} \tag{4.10}$$

Por lo tanto:

$$E_{\text{batería}} = 7{,}4 \cdot 3600 = 26640\,\text{J} \tag{4.11}$$

Para el caso de tecnología LoRa con SF 7 se tienen los siguientes cálculos:

Energía por transmisión:

$$E_{\text{tx}} = 0{,}079\,\text{J} \tag{4.12}$$

Entonces, para calcular el número total de transmisiones posibles:

$$N_{\text{tx}} = \frac{E_{\text{batería}}}{E_{\text{tx}}} = \frac{26640J}{0{,}079J} \approx 337215 \text{ transmisiones} \tag{4.13}$$

Considerando ese número de transmisiones hacemos los calculos para los distintos casos de intervalos, partiendo con cuantas transmisiones se puede realizar cada 1 hora segun los distintos intervalos para finalmente hacer la estimacion de los años:

Intervalo: cada 1 minuto (transmisiones por hora: 60)

$$\text{Horas} = \frac{337215}{60} = 5620{,}25 \text{ horas} \tag{4.14}$$

$$\text{Días} = \frac{5620{,}25}{24} \approx 234{,}18 \text{ días} \tag{4.15}$$

Intervalo: cada 5 minutos (transmisiones por hora: 12)

$$\text{Horas} = \frac{337215}{12} \approx 28101{,}25 \text{ horas} \tag{4.16}$$

$$\text{Días} = \frac{28101{,}25}{24} \approx 1170{,}89 \text{ días} \tag{4.17}$$

Intervalo: cada 15 minutos (transmisiones por hora: 4)

$$\text{Horas} = \frac{337215}{4} = 84303{,}75 \text{ horas} \tag{4.18}$$

$$\text{Días} = \frac{84303{,}75}{24} \approx 3512{,}66 \text{ días} \tag{4.19}$$

Los cálculos realizados se pueden apreciar en la Tabla 4.3. Para los demás valores de SF, se llevará acabo el mismo procedimiento matemático con la diferencia del valor de la energia por transmisión, ya que para SF 9 el valor de energía por transmisión es de $E_{\text{tx}} = 0{,}2917\,\text{J}$ lo cual da un resultado de 91305,57 transmisiones y para SF 11 es de $E_{\text{tx}} = 1{,}2619\,\text{J}$ lo cual da un resultado de 21104,98 transmisiones, por ende solo se mostrará el resultado obtenido en las tablas 4.4 y 4.5 sin el procedimiento matemático ya que es el mismo que para el SF 7 ya mostrado.

Tabla 4.3: Duración de la batería para diferentes intervalos de transmisión usando LoRa SF 7.

Intervalo (min)	**Duración (horas)**	**Duración (días)**	**Duración (años)**
Cada 1	5620,25 h	234,18 días	0,64 años
Cada 5	28101,25 h	1170,89 días	3,21 años
Cada 15	84303,75 h	3512,66 días	9,62 años

Tabla 4.4: Duración de la batería para diferentes intervalos de transmisión usando LoRa SF 9.

Intervalo (min)	**Duración (horas)**	**Duración (días)**	**Duración (años)**
Cada 1	1521,71 h	63,40 días	0,17 años
Cada 5	7608,80 h	317,03 días	0,87 años
Cada 15	22826,39 h	951,10 días	2,61 años

Tabla 4.5: Duración de la batería para diferentes intervalos de transmisión usando LoRa SF 11.

Intervalo (min)	**Duración (horas)**	**Duración (días)**	**Duración (años)**
Cada 1	351,71 h	14,65 días	0,04 años
Cada 5	1758,75 h	73,28 días	0,20 años
Cada 15	5276,25 h	219,85 días	0,60 años

Esas son las estimaciones de un caso hipotetico si se usara la tecnología LoRa a distintos intervalos de tiempo, ahora se verá como sería con tecnología Wi-Fi. Con LoRa, el módulo solo consume energía significativamente cuando transmite un paquete. Entre transmisiones está mayormente en reposo.

4.2.2. Consumo energético de Wi-Fi

A diferencia de LoRa, donde el consumo energético presenta un único pulso claramente identificable durante el envío de cada paquete, en el caso de Wi-Fi se observaron varios picos de corriente por segundo (aproximadamente 8), incluso en ausencia de transmisión activa de datos. Estos pulsos, registrados en la medición experimental, corresponden a las múltiples operaciones de mantenimiento y gestión de la conexión que realiza el módulo Wi-Fi, como la sincronización con el punto de acceso y el manejo de beacons, lo que evidencia que la interfaz permanece activa de forma continua.

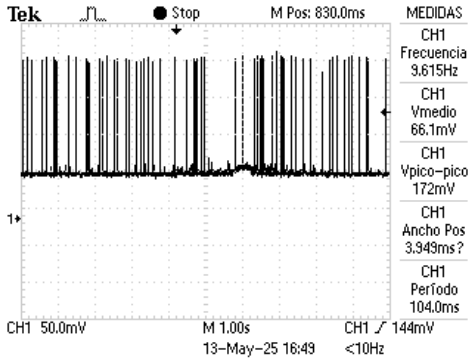

Fig. 4.26: Wi-Fi.

Este comportamiento es típico de Wi-Fi y es uno de los motivos por los cuales el consumo energético en Wi-Fi es significativamente mayor, incluso cuando la tasa de transmisión de datos es baja. La diferencia clave es que Wi-Fi no entra en modo reposo profundo entre paquetes a menos que se implemente gestión de energía específica, mientras que LoRa permanece en modo inactivo casi todo el tiempo, despertando únicamente para transmitir.

Los valores de Vpico-pico y Ancho Pos que se pueden apreciar en la Fig. 4.26 para Wi-Fi se calculó la corriente de la misma forma que con tecnología LoRa, a traves de la Ley de Ohm, resultando los siguientes parámetros de la Tabla 4.6:

Tabla 4.6: Valores medidos y calculados.

Parámetro	**Descripción**	**Valor**
V_{pp}	Voltaje pico a pico	172 mV
R	Resistencia	0,5 Ω
I	Corriente	344 mA
t	Tiempo	3,959 ms = 0,003959 s

Si embargo, no se puede seguir el procedimiento realizado con tecnología LoRa para el consumo energético ya que como se vió en las imagenes, con LoRa teniamos identificada cuando la onda rectangular hacía referencia a un paquete enviado, lo que no sucede con Wi-Fi ya que el hecho de que se vean múltiples pulsos por segundo no significa que se esten enviando múltiples paquetes. Estos pulsos corresponden a las operaciones internas del protocolo Wi-Fi (sin transmisión de datos).

Si bien el sistema Wi-Fi realiza múltiples pulsos por segundo, podemos pensar que la tasa de consumo energético debe estar asociada a la cantidad de tiempo que el sistema está activo realizando tareas como mantenimiento de la red. Los 8 pulsos por segundo indican que, aunque el paquete se envíe cada segundo, el dispositivo está haciendo un esfuerzo constante para mantener la conexión.

Dado que estamos observando pulsos continuos, no debemos asociar los pulsos con el envío de un paquete de datos específico. En lugar de eso, calculamos el consumo energético durante un segundo completo de actividad, considerando que la placa Wi-Fi está trabajando constantemente para mantener la conexión, manejar la red y enviar datos para lo cual se consideraron 5 segundos de trabajo exclusico de Wi-Fi como por ejemplo la asociasion Acces Point y obtencion de direccion IP, lo cual tiene que ser considerado en el calculo de consumo energético.

Por lo tanto los datos que tenemos para hacer el calculo de energía se pueden ver en la Tabla 4.7:

Tabla 4.7: Datos utilizados para el cálculo de consumo energético de Wi-Fi.

Parámetro	**Valor**
Corriente (I)	344 mA = 0,344 A
Voltaje (V)	5 V
Duración de un pulso (t_{pulso})	3,949 ms = 0,003949 s
Pulsos por segundo	8
Tiempo de encendido del Wi-Fi	5 s

Para hacer el calculo del tiempo total activo se hace lo siguiente:

$$t_{\text{total\_activo}} = 8\,\text{pulsos/s} \times 0{,}003949\,\text{s/pulso} \times 5\,\text{s} \qquad = 0{,}15796\,\text{s} \tag{4.20}$$

Por lo tanto la energía estimada para una transmisión es:

$$\begin{aligned} E &= V \cdot I \cdot t \\ &= 5\,\text{V} \cdot 0{,}344\,\text{A} \cdot 0{,}15796\,\text{s} \\ &= 0{,}2712\,\text{J} \end{aligned} \tag{4.21}$$

Ahora que tenemos calculada la energía consumida con Wi-Fi se puede realizar el cálculo de la autonomía de una batería con los mismas especificaciones de la batería e intervalos de tiempo utilizados anteriormente con tecnología LoRa

Sabemos que la batería tiene:

$$E_{\text{batería}} = 26640\,\text{J}$$

Entonces, la duración que puede sostener ese consumo es:

$$N_{\text{tx}} = \frac{E_{\text{batería}}}{E_{\text{tx}}} = \frac{26640\,\text{J}}{0{,}2712\,\text{J}} \approx 98249\,\text{transmisiones} \tag{4.22}$$

A partir del resultado obtenido de transmisiones, se procederá a realizar el calculo para los mismos intervalos de tiempo que anteriormente con tecnología LoRa (1, 5 y 15 minutos), como se sigue el mismo procedimiento ya mostrado para SF 7 de LoRa, solo se mostrarán los resultados obtenidos en la Tabla 4.8.

Tabla 4.8: Duración de la batería para diferentes intervalos de transmisión usando Wi-Fi.

Intervalo (min)	**Duración (horas)**	**Duración (días)**	**Duración (años)**
Cada 1	1637,60 h	68,23 días	0,19 años
Cada 5	8187,98 h	341,17 días	0,93 años
Cada 15	24563,95 h	1023,50 días	2,80 años

4.2.3. Resumen de resultados

Es importante recordar que el análisis presentado en esta sección tiene un carácter referencial y no representa una estimación completamente realista del consumo energético total del sistema. Esto se debe a que los cálculos se basan únicamente en la energía consumida durante la transmisión de datos, sin considerar el consumo en estado inactivo. Por lo tanto, los valores de energía y duración de batería expuestos deben interpretarse como valores mínimos teóricos, útiles para comparar de forma relativa el desempeño energético de cada tecnología y configuración, pero no como proyecciones exactas del funcionamiento en un entorno real.

La Tabla 4.9 presenta una comparación entre las tecnologías LoRa y Wi-Fi en términos de energía consumida por transmisión y el número total de transmisiones posibles con

una batería de 3.7 V y 2000 mAh.

Tabla 4.9: Energía consumida por transmisión y número de transmisiones para diferentes tecnologías.

Tecnología / SF	**Energía consumida (J)**	**Número de transmisiones**
LoRa SF 7	0,0790	337215
LoRa SF 9	0,2917	91305
LoRa SF 11	1,2619	21104
Wi-Fi	0,2712	98249

Se observa que la opción más eficiente en términos energéticos es LoRa con SF 7, con un consumo de 0.0790 J por transmisión, lo que lo convierte en la alternativa más adecuada para aplicaciones donde la duración de la batería es crítica y considerando que con este parámetro se obtuvo el mismo alcance que con SF 9 y 11 en urbano y en rural un alcance de 3km, es una opcion completamente viable para aplicaciones IoT de largo alcance con bajo consumo energético.

A medida que se incrementa el Spreading Factor, el consumo energético también aumenta considerablemente. En el caso de SF 9, la energía por transmisión asciende a 0,2917 J, mientras que en SF 11, alcanza los 1,2619 J, lo que representa un aumento de más de 15 veces respecto a SF 7. Este comportamiento es consistente con el hecho de que un mayor SF implica una mayor duración del chirp y, por ende, un mayor tiempo en que el dispositivo permanece activo consumiendo energía. Se observa que Wi-Fi consume menos energía por transmisión (0,2712 J) que LoRa con SF 9 y SF 11, sin embargo es fundamental considerar que Wi-Fi tiene un alcance significativamente limitado, de apenas 124 metros, frente a los 1-4 km de LoRa dependiendo del Spreading Factor (SF) y el entorno, por ende, hay que tener presente que Wi-Fi consume menos energía que SF 9 de LoRa por ejemplo, pero esta última tecnología tiene un alcance de hasta 28 veces más en zona rural y 8 veces más en zona urbana, y este proyecto está enfocado en largo alcance y bajo consumo energético.

En aplicaciones IoT donde el alcance es crítico, como en áreas rurales o urbanas con dispositivos distribuidos, la capacidad de LoRa para operar en rangos superiores justifica su mayor consumo energético. Por ejemplo, con SF 11, LoRa puede alcanzar hasta 4 km en áreas rurales y 1 km en zonas urbanas, lo que lo hace ideal para

aplicaciones en agricultura de precisión, monitoreo de infraestructura o entornos industriales extensos. En contraste, Wi-Fi, aunque energéticamente más eficiente en algunos casos, es adecuado solo para aplicaciones de corto alcance, como entornos domésticos o industriales donde los dispositivos se encuentran a distancias cercanas.

La Tabla 4.10 muestra la duración de la batería para diferentes tecnologías e intervalos de transmisión. A primera vista, Wi-Fi parece ofrecer una mayor duración de batería frente a LoRa con SF 9 y SF 11. Sin embargo, este análisis debe contextualizarse considerando los alcances y aplicaciones de cada tecnología.

Tabla 4.10: Duración de la batería (en años) según tecnología e intervalo de transmisión.

Tecnología	**Intervalo (min)**	**Duración (años)**
LoRa SF 7	Cada 1	0,64
	Cada 5	3,21
	Cada 15	9,62
LoRa SF 9	Cada 1	0,17
	Cada 5	0,87
	Cada 15	2,61
LoRa SF 11	Cada 1	0,04
	Cada 5	0,20
	Cada 15	0,60
Wi-Fi	Cada 1	0,19
	Cada 5	0,93
	Cada 15	2,80

En escenarios donde los dispositivos están separados por grandes distancias, LoRa es claramente superior. Por ejemplo, con un intervalo de transmisión de 15 minutos, LoRa con SF 7 permite una duración de batería de 9,62 años, con un alcance de hasta 3 km en zonas rurales y 1 km en áreas urbanas. En comparación, Wi-Fi, que tiene una duración de 2,80 años en el mismo intervalo, está limitado a un alcance de apenas 124 metros. Incluso con SF 9 y SF 11, que presentan duraciones de batería más cortas (2,61 y 0,60 años, respectivamente), LoRa sigue siendo una opción preferible para aplicaciones que exigen una cobertura amplia.

Además del alcance y el consumo energético, el tipo de aplicación también influye en la elección de la tecnología. LoRa se utiliza comúnmente en transmisiones de datos esporádicos y de baja velocidad, características propias de muchas aplicaciones IoT,

como sensores ambientales o medidores inteligentes. En este tipo de escenarios, las tasas de transmisión que ofrece LoRa son suficientes. Por ejemplo, utilizando un ancho de banda de 125 kHz, las tasas de transmisión pueden variar aproximadamente entre 0,3 kbps (con SF = 12) y 5,5 kbps (con SF = 7), dependiendo del Spreading Factor y del código de corrección de errores (CR) configurado.

La tasa de transmisión en LoRa puede estimarse utilizando la siguiente fórmula:

$$R_b = \frac{SF \cdot BW}{2^{SF}} \cdot \frac{4}{4 + CR} \tag{4.23}$$

donde:

- R_b es la tasa de bits (bps),
- SF es el Spreading Factor (de 7 a 12),
- BW es el ancho de banda (en Hz),
- CR es el denominador del código de corrección de errores, típicamente entre 1 y 4 (para CR = 4/5, CR = 1).

Por ejemplo, para $SF = 7$, $BW = 125000$ Hz y $CR = 1$ (es decir, CR = 4/5), se obtiene:

$$R_b = \frac{7 \cdot 125000}{2^7} \cdot \frac{4}{4 + 1} = 5468{,}75 \text{ bps} \approx 5{,}5 \text{ kbps}$$

Estas bajas tasas permiten alcanzar una gran cobertura y mantener un consumo energético reducido, lo que resulta ideal para dispositivos IoT que transmiten pequeños volúmenes de datos a intervalos esporádicos.

En contraste, Wi-Fi está diseñado para aplicaciones que requieren una alta velocidad de transmisión, como en entornos multimedia o de conectividad general. Por ejemplo, bajo el estándar IEEE 802.11ax (Wi-Fi 6), con canal de 20 MHz, es posible alcanzar velocidades de hasta 143,4 Mbps en condiciones ideales

Estas diferencias en la capacidad de transmisión de datos y el diseño del protocolo reflejan los objetivos distintos de cada tecnología, y explican por qué LoRa y Wi-Fi resultan más adecuados en contextos muy distintos dentro del ecosistema IoT.

Aunque Wi-Fi puede parecer más eficiente para intervalos cortos y aplicaciones de proximidad, LoRa ofrece una solución más equilibrada y sostenible para aplicaciones IoT que requieren un gran alcance y duraciones prolongadas de batería, a pesar de un mayor consumo energético. Esto subraya la importancia de seleccionar la tecnología adecuada según el caso de uso específico.

En resumen, este análisis energético realizado valida la problemática planteada en el proyecto. Además, refuerza que la elección del SF y del intervalo de transmisión son factores clave para maximizar la autonomía del dispositivo y cumplir con los objetivos definidos al inicio del trabajo.

Capítulo 5

Conclusiones

5.1. Sumario

El desarrollo e implementación de un sistema de transmisión de datos a larga distancia basado en la tecnología LoRa ha permitido explorar de forma práctica y técnica las capacidades reales de esta tecnología dentro del contexto de aplicaciones IoT. Durante el proceso de diseño, programación y prueba del sistema, se abordan múltiples desafíos relacionados con la integración de los componentes físicos, como los módulos TTGO LoRa32, así como la gestión eficiente de los datos transmitidos entre las placas emisoras y receptoras.

Los resultados obtenidos evidencian que, bajo condiciones controladas y con una configuración adecuada de parámetros como el Spreading Factor, la potencia de transmisión y la tasa de codificación, el sistema es capaz de establecer una comunicación robusta en distancias considerables, superando los 2 kilómetros en línea de vista sin pérdida de paquetes. Esta distancia, si bien es modesta en comparación con el máximo teórico de la tecnología LoRa, es significativa en entornos urbanos o semiurbanos donde la infraestructura de red es limitada o inexistente. Además, se validó la estabilidad del sistema en la transmisión de tramas simples, lo que refuerza su aplicabilidad para escenarios donde se requiera el envío periódico de datos de sensores, como temperatura, humedad, o niveles de gas.

Cabe destacar que durante el proceso de pruebas también se identificaron limitaciones

propias del hardware y el entorno de pruebas. Factores como la interferencia electromagnética, la ubicación física de las antenas y las condiciones topográficas influyeron directamente en la calidad del enlace. Estos hallazgos permiten comprender con mayor profundidad los aspectos críticos que deben considerarse al diseñar una red LoRa real en el mundo físico, especialmente en proyectos de mayor escala o en aplicaciones industriales.

Además, se llevó a cabo un análisis detallado del consumo energético del sistema durante la transmisión de datos, utilizando diferentes configuraciones del Spreading Factor (SF) y una comparación con tecnología Wi-Fi. A través de mediciones de corriente y cálculos de energía por paquete transmitido, se evidenció que el SF tiene un impacto directo en la duración de los pulsos de transmisión y, por ende, en el consumo energético total. Se estimó la autonomía del sistema bajo distintos intervalos de envío y se comprobó que, con configuraciones adecuadas y un intervalo de transmisión espaciado, es posible alcanzar una larga duración de batería. Estos resultados refuerzan la viabilidad de emplear LoRa en aplicaciones IoT que requieren operación prolongada y autónoma en entornos sin acceso continuo a energía eléctrica.

5.2. Conclusiones

El desarrollo e implementación de un sistema de transmisión de datos a larga distancia basado en tecnología LoRa permitió validar satisfactoriamente la problemática inicial de este proyecto. A través de la utilización de hardware accesible como las placas LoRa ESP32 y una Raspberry Pi, se logró establecer una comunicación efectiva y de bajo consumo energético entre dispositivos IoT, demostrando que es posible implementar soluciones viables en entornos donde las redes tradicionales presentan limitaciones de cobertura o altos costos operativos.

Las pruebas de campo realizadas en entornos urbanos y rurales permitieron identificar claramente las diferencias en el comportamiento del sistema frente a las condiciones propias de cada escenario. En áreas urbanas, se observó una degradación progresiva de la señal conforme aumentaba la distancia, producto de la presencia de interferencias electromagnéticas y obstáculos físicos como edificaciones y mobiliario urbano. A pesar de ello, la tasa de pérdida de paquetes se mantuvo dentro de márgenes tolerables,

validando la eficacia de la tecnología LoRa en estos contextos. En contraste, las pruebas en zonas rurales ofrecieron resultados más favorables, con una mayor estabilidad en la señal y una menor tasa de pérdida de paquetes, lo que resalta las ventajas de operar en entornos con menor densidad de interferencias y una topografía más abierta.

Desde el punto de vista energético, la implementación de modos de bajo consumo en las placas LoRa ESP32 fue determinante para mejorar la eficiencia general del sistema. Esta optimización contribuye significativamente a extender la vida útil de las baterías, aspecto crucial en aplicaciones IoT que requieren operaciones autónomas prolongadas, especialmente en ubicaciones remotas donde el acceso a la energía es limitado o inexistente, y su comparación energética con Wi-Fi evidencia el correcto uso de tecnología LoRa para esta clase de proyectos que estén orientados a largo alcance con bajo consumo energético.

La arquitectura modular del sistema demostró ser una ventaja estratégica, facilitando tanto el proceso de implementación como su posible escalabilidad. La combinación entre la Raspberry Pi y las placas LoRa ESP32 permitió un diseño flexible, capaz de adaptarse a futuras integraciones o ampliaciones de la red, lo que otorga un valor añadido a la solución desarrollada. Adicionalmente, este proyecto refuerza la pertinencia de LoRa como tecnología para aplicaciones de monitoreo remoto, gestión de recursos y automatización de procesos, mostrando que con una adecuada configuración de hardware y software es posible superar muchas de las limitaciones que tradicionalmente enfrenta la conectividad en zonas de difícil acceso.

En definitiva, este trabajo no solo valida técnicamente la capacidad de LoRa para establecer redes IoT eficientes y de largo alcance, sino que también demuestra que es una alternativa realista y práctica para diversos sectores que requieren soluciones de comunicación asequibles, sostenibles y con buena proyección de escalabilidad. Los aprendizajes obtenidos durante el desarrollo de este proyecto aportan un conocimiento valioso para futuros trabajos en la ingeniería de telecomunicaciones, abriendo nuevas oportunidades para la expansión de tecnologías IoT orientadas a mejorar la conectividad en diferentes entornos.

En conclusión, la eficiencia energética demostrada por LoRa, combinada con su capacidad de alcanzar distancias considerables, refuerza su idoneidad para aplicaciones IoT de largo alcance y operación prolongada, especialmente en contextos con

limitaciones energéticas.

5.3. Trabajo a Futuro

Si bien este proyecto logró cumplir con sus objetivos, existen múltiples caminos para continuar su desarrollo. Una extensión natural del sistema sería la implementación de una red LoRa basada en la topología star-of-stars, utilizando un Gateway LoRaWAN que permita conectar múltiples nodos emisores con un servidor central en la nube. Esto abriría la posibilidad de gestionar redes IoT más complejas, con recolección de datos centralizada, acceso remoto a los registros históricos, y visualización en tiempo real mediante interfaces web o móviles.

Asimismo, se recomienda incorporar sensores reales (como de temperatura, humedad, gases, etc.) al sistema, con el fin de generar información relevante que pueda ser utilizada en casos de uso específicos, como el monitoreo de condiciones ambientales o el control de variables en agricultura de precisión. La implementación de técnicas de compresión de datos o control de errores avanzados también puede optimizar aún mas el rendimiento del sistema, especialmente en entornos ruidosos o con interferencias.

Otro camino prometedor es el estudio de mecanismos de seguridad en la transmisión de datos, incluyendo cifrado de extremo a extremo y autenticación de nodos, para garantizar la integridad y privacidad de la información. Finalmente, sería valioso realizar pruebas a mayor escala en diversos entornos, urbano, rural y forestal. Para evaluar el comportamiento del sistema en condiciones reales, incluyendo la durabilidad de las placas, la estabilidad del enlace y la eficiencia energética en despliegues prolongados.

Bibliografía

[1] J. de Carvalho Silva, J. J. P. C. Rodrigues, A. M. Alberti, P. Solic, and A. L. L. Aquino, "LoRaWAN — A low power WAN protocol for Internet of Things: A review and opportunities." in *Proc. 2nd Int. Multidiscip. Conf. Comput. Energy Sci. (SpliTech)*, 2017, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8019271

[2] J. Haxhibeqiri, E. D. Poorter, I. Moerman, and J. Hoebeke, "A Survey of LoRaWAN for IoT: From Technology to Application." *IEEE Sens. J.*, vol. 18, no. 11, pp. 10–27, 2018.

[3] M. A. M. Almuhaya, W. A. Jabbar, N. Sulaiman, and S. Abdulmalek, "A Survey on LoRaWAN Technology: Recent Trends, Opportunities, Simulation Tools and Future Directions." *Electronics*, vol. 11, no. 1, pp. 3–25, 2022.

[4] L. Ntseane and B. Isong, "Analysis of LoRa/LoRaWAN Challenges: Review." in *Proc. Int. Multidiscip. Inf. Technol. Eng. Conf. (IMITEC)*, 2019, pp. 1–7.

[5] LoRa Alliance, "LoRaWAN Specification v1.1." 2017, [Online]. Available: https://hz1.37b.myftpupload.com/resource_hub/lorawan-specification-v1-1/.

[6] LoRa Alliance, "LoRaWAN 1.0.4 Specification Package." 2020, [Online]. Available: https://hz1.37b.myftpupload.com/resource_hub/lorawan-104-specification-package/.

[7] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach.*, 6th ed. Upper Saddle River, NJ, USA: Pearson Education, 2013.

[8] W. Tomasi, *Sistemas de Comunicaciones Electrónicas.*, 4th ed. Mexico City, Mexico: Pearson Educación, 2003.

[9] W. Stallings, *Wireless Communications and Networks.*, 2nd ed. Upper Saddle River, NJ, USA: Pearson Prentice Hall, 2005.

[10] W. Stallings, *Data and Computer Communications.*, 10th ed. Boston, MA, USA: Pearson, 2014.

[11] Semtech Corporation, "AN1200.22 LoRa Modulation Basics." pp. 1–26, 2015, [Online]. Available: https://docslib.org/doc/2371755/an1200-22-lora-modulation-basics.

[12] D. Newman, "Return On IoT: Dealing With The IoT Skills Gap." Forbes, 2019, [Online]. Available: https://www.forbes.com/sites/danielnewman/2019/07/30/return-on-iot-dealing-with-the-iot-skills-gap/.

[13] S. Shea, "What is LPWAN (Low-Power Wide-Area Network)?" Search IoT, TechTarget, 2017, [Online]. Available: https://www.techtarget.com/iotagenda/definition/LPWAN-low-power-wide-area-network.

[14] B. S. Chaudhari and M. Zennaro, Eds., *LPWAN Technologies for IoT and M2M Applications.* London, UK: Elsevier, 2020.

[15] WAVIoT, "NB-Fi Specification." pp. 1–29, 2016, [Online]. Available: https://waviot.com/technology/nb-fi-specification/.

[16] U. Raza, P. Kulkarni, and M. Sooriyabandara, "Low Power Wide Area Networks: An Overview." *IEEE Commun. Surveys Tuts.*, vol. 19, no. 2, pp. 855–873, 2017.

[17] LoRa Alliance Technical Marketing Workgroup, "A Technical Overview of LoRa and LoRaWAN." White Paper, LoRa Alliance, 2015, [Online]. Available: https://lora-alliance.org/resource_hub/technical-overview-of-lora-and-lorawan/.

[18] LoRa Alliance, "About LoRaWAN." 2025, [Online]. Available: https://lora-alliance.org/about-lorawan/.

[19] LoRa Alliance, "LoRaWAN Link Layer Specification v1.0.4." 2020, [Online]. Available: https://resources.lora-alliance.org/technical-specifications.

[20] LoRa Alliance, "LoRaWAN Specification v1.0." 2015, [Online]. Available: https://hz1.37b.myftpupload.com/resource_hub/lorawan-specification-v1-0/.

[21] A. Augustin, J. Yi, T. Clausen, and W. M. Townsley, "A Study of LoRa: Long Range & Low Power Networks for the Internet of Things." *Sensors (Basel)*, vol. 16, no. 9, pp. 2–16, 2016.

[22] Nanotron Technologies GmbH, "Technology CSS Location Running." 2018, [Online]. Available: https://nanotron.com/EN/co_techn-css-php/.

[23] C.-H. Chu, Y.-M. Chen, Y.-T. Huang, R. Carvalho, C.-C. Hsu, and L.-J. Chen, "Measurement of long-distance Wi-Fi connections: An empirical study." in *Proc. IEEE Int. Conf. Commun. (ICC)*. Sydney, NSW, Australia: IEEE, 2014, pp. 1–6.

[24] A. Anttonen and M. Höyhtyä, "Emerging 5G Satellite-Aided Networks for Mission-Critical Services." *IEEE Aerospace and Electronic Systems Magazine*, vol. 35, no. 11, pp. 30–37, 2020.

[25] D. Pinto, J. M. Torres, A. S. G. Bello, N. A. Pérez, and J. R. Uzcátegui, "Modelo para Estimación de Pérdidas de Propagación en Sistema de Televisión Digital Abierta." Technical Report, Universidad Central de Venezuela, 2023.

[26] J. Smith, L. Johnson, M. Lee, and A. Martinez, "LoRa Technology Propagation Models for IoT Network Planning in the Amazon Regions." *Sensors*, vol. 24, no. 5, pp. 1–20, 2024.

[27] A. Elmaghbub and B. Hamdaoui, "Release Note: Comprehensive LoRa RF Datasets for Device Fingerprinting Using Deep Learning." NIST National Institute of Standards and Technology, 2020, [Online]. Available: https://www.nist.gov/publications/release-note-comprehensive-lora-rf-datasets-device-fingerprinting-using-deep-learning.

Capítulo 6

ANEXO: Código

.1. Código de la Placa Transmisora

```
#include <SPI.h>
#include <LoRa.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET    -1
#define SCREEN_ADDRESS 0x3C

#define SCK     5
#define MISO    19
#define MOSI    27
#define SS      18
#define RST     14
#define DIO     26

#define FREQUENCY 915E6
#define TX_POWER  20

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire,
   OLED_RESET);
```

```
23 String inputString = "";
24 boolean stringComplete = false;
25 float distance = 0.0;
26 unsigned long packetCounter = 0;
27 unsigned long totalPacketsSent = 0;
28 unsigned long startTime = 0;
29 unsigned long lastPacketTime = 0;
30 unsigned long lastDisplayUpdateTime = 0;
31 unsigned long elapsedTime = 0;
32 bool transmissionActive = false;
33
34 void setup() {
35   Serial.begin(115200);
36   delay(1000);
37
38   Serial.println("LILYGO␣LoRa␣Distance␣Transmitter");
39
40   Wire.begin();
41   if(!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
42     Serial.println("SSD1306␣allocation␣failed");
43   } else {
44     display.clearDisplay();
45     display.setTextSize(1);
46     display.setTextColor(SSD1306_WHITE);
47     display.setCursor(0, 0);
48     display.println("LoRa␣Distance␣TX");
49     display.println("Initializing...");
50     display.display();
51   }
52
53   Serial.println("Setting␣up␣LoRa...");
54
55   SPI.begin(SCK, MISO, MOSI, SS);
56
57   LoRa.setPins(SS, RST, DIO);
58
59   Serial.println("LoRa␣pins␣set");
60
61   unsigned long startAttemptTime = millis();
62   bool loraInitialized = false;
63
```

```
64    while ((millis() - startAttemptTime) < 5000 && !loraInitialized) {
65      if (LoRa.begin(FREQUENCY)) {
66        loraInitialized = true;
67      } else {
68        Serial.println(".");
69        delay(500);
70      }
71    }
72
73    if (!loraInitialized) {
74      Serial.println("Starting␣LoRa␣failed!");
75      display.clearDisplay();
76      display.setCursor(0, 0);
77      display.println("LoRa␣init␣failed!");
78      display.display();
79
80    } else {
81      Serial.println("LoRa␣initialized␣successfully!");
82
83      LoRa.setTxPower(TX_POWER);
84
85      LoRa.setSpreadingFactor(7);
86      LoRa.setSignalBandwidth(125E3);
87      LoRa.setCodingRate4(5);
88      LoRa.setPreambleLength(12);
89      Serial.println("LoRa␣configured␣OK!");
90    }
91
92    display.clearDisplay();
93    display.setCursor(0, 0);
94    display.println("LoRa␣Ready!");
95    display.println("Enter␣distance␣in␣meters");
96    display.println("via␣serial␣console...");
97    display.display();
98
99    Serial.println("Enter␣distance␣in␣meters␣and␣press␣Enter:");
100 }
101
102 void loop() {
103
104   while (Serial.available() > 0) {
```

```
105     char inChar = (char)Serial.read();
106
107
108     if (inChar == '\n' || inChar == '\r') {
109       if (inputString.length() > 0) {
110         stringComplete = true;
111       }
112     } else {
113       inputString += inChar;
114     }
115   }
116
117
118   if (stringComplete) {
119     distance = inputString.toFloat();
120
121     if (distance > 0) {
122       Serial.print("Setting␣distance␣to:␣");
123       Serial.print(distance);
124       Serial.println("␣meters");
125
126       packetCounter = 0;
127       totalPacketsSent = 0;
128       startTime = millis();
129       lastPacketTime = 0;
130       transmissionActive = true;
131
132       display.clearDisplay();
133       display.setCursor(0, 0);
134       display.println("Starting␣transmission");
135       display.print("Distance:␣");
136       display.print(distance);
137       display.println("␣m");
138       display.display();
139     } else {
140       Serial.println("Invalid␣distance.␣Please␣enter␣a␣positive␣number
            .");
141
142       display.clearDisplay();
143       display.setCursor(0, 0);
144       display.println("ERROR:");
```

```
145       display.println("Invalid distance");
146       display.println("Enter a positive number");
147       display.display();
148     }
149
150     inputString = "";
151     stringComplete = false;
152   }
153
154   if (transmissionActive) {
155     unsigned long currentTime = millis();
156
157     if (currentTime - lastPacketTime >= 1000) {
158       sendPacket();
159       lastPacketTime = currentTime;
160     }
161
162     if (currentTime - lastDisplayUpdateTime >= 250) {
163       updateDisplay();
164       lastDisplayUpdateTime = currentTime;
165     }
166   }
167
168   delay(10);
169 }
170
171 void sendPacket() {
172   unsigned long currentTime = millis();
173   elapsedTime = currentTime - startTime;
174
175   LoRa.beginPacket();
176
177   String packet = "{";
178   packet += "\"seq\":" + String(packetCounter) + ",";
179   packet += "\"dist\":" + String(distance) + ",";
180   packet += "\"total\":" + String(totalPacketsSent) + ",";
181   packet += "\"timestamp\":" + String(currentTime);
182   packet += "}";
183
184   LoRa.print(packet);
185
```

```
186   LoRa.endPacket();
187
188   packetCounter++;
189   totalPacketsSent++;
190
191   Serial.print("Packet␣sent␣[");
192   Serial.print(packetCounter);
193   Serial.print("]:␣");
194   Serial.println(packet);
195
196   if (Serial.available() > 0) {
197     Serial.println("\nNew␣input␣detected.␣Stopping␣transmission.");
198     Serial.println("Enter␣distance␣in␣meters␣and␣press␣Enter:");
199
200     display.clearDisplay();
201     display.setCursor(0, 0);
202     display.println("Transmission␣stopped");
203     display.println("Enter␣new␣distance...");
204     display.display();
205
206     transmissionActive = false;
207   }
208 }
209
210 void updateDisplay() {
211   display.clearDisplay();
212
213   display.setTextSize(1);
214   display.setCursor(0, 0);
215   display.print("Msg:␣");
216   display.println(packetCounter);
217
218   display.setTextSize(1);
219   display.setCursor(0, 20);
220   display.print("Dist:␣");
221   display.print(distance, 1);
222   display.println("m");
223
224   display.setTextSize(1);
225   display.setCursor(0, 40);
226   display.print("Time:␣");
```

```
227
228   unsigned long seconds = elapsedTime / 1000;
229   unsigned long minutes = seconds / 60;
230
231   if (minutes < 10) display.print("0");
232   display.print(minutes % 60);
233   display.print(":");
234
235   if ((seconds % 60) < 10) display.print("0");
236   display.print(seconds % 60);
237
238   display.display();
239 }
```

Listing 1: Código en C++

.2. Código de la Placa Receptora

```
1  #include <SPI.h>
2  #include <LoRa.h>
3  #include <Wire.h>
4  #include <Adafruit_GFX.h>
5  #include <Adafruit_SSD1306.h>
6
7  #define SCREEN_WIDTH 128
8  #define SCREEN_HEIGHT 64
9  #define OLED_RESET    -1
10 #define SCREEN_ADDRESS 0x3C
11
12 #define SCK     5
13 #define MISO    19
14 #define MOSI    27
15 #define SS      18
16 #define RST     14
17 #define DIO     26
18
19 #define FREQUENCY 915E6
20
21 #define DEBUG_MODE true
```

```
22
23 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire,
       OLED_RESET);
24
25 unsigned long lostPackets = 0;
26 unsigned long expectedSequence = 0;
27 float distance = 0.0;
28 float lastDistance = -1.0;
29 float rssi = 0;
30 float snr = 0;
31 float packetLossRate = 0.0;
32 unsigned long totalPacketsSent = 0;
33 unsigned long lastDisplayUpdateTime = 0;
34 unsigned long currentPacketId = 0;
35 unsigned long cumulativeReceived = 0;
36
37 unsigned long lastPacketTime = 0;
38 bool receivedFirstPacket = false;
39
40 void setup() {
41   Serial.begin(115200);
42   delay(1000);
43
44   if (DEBUG_MODE) {
45     Serial.println("LILYGO␣LoRa␣Distance␣Receiver");
46   }
47   Wire.begin();
48   if (!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
49     if (DEBUG_MODE) Serial.println("SSD1306␣allocation␣failed");
50   } else {
51     display.clearDisplay();
52     display.setTextSize(1);
53     display.setTextColor(SSD1306_WHITE);
54     display.setCursor(0, 0);
55     display.println("LoRa␣Distance␣RX");
56     display.println("Initializing...");
57     display.display();
58   }
59
60   SPI.begin(SCK, MISO, MOSI, SS);
61
```

```
62    LoRa.setPins(SS, RST, DIO);
63
64    unsigned long startAttemptTime = millis();
65    bool loraInitialized = false;
66    while ((millis() - startAttemptTime) < 5000 && !loraInitialized) {
67      if (LoRa.begin(FREQUENCY)) {
68        loraInitialized = true;
69      } else {
70        if (DEBUG_MODE) Serial.print(".");
71        delay(500);
72      }
73    }
74
75    if (!loraInitialized) {
76      if (DEBUG_MODE) Serial.println("Starting␣LoRa␣failed!");
77      display.clearDisplay();
78      display.setCursor(0, 0);
79      display.println("LoRa␣init␣failed!");
80      display.display();
81    } else {
82      if (DEBUG_MODE) Serial.println("LoRa␣initialized␣successfully!");
83      LoRa.setSpreadingFactor(11);
84      LoRa.setSignalBandwidth(125E3);
85      LoRa.setCodingRate4(8);
86      LoRa.setPreambleLength(12);
87    }
88
89    display.clearDisplay();
90    display.setCursor(0, 0);
91    display.println("LoRa␣Ready!");
92    display.println("Waiting␣for␣packets...");
93    display.display();
94
95    lastDisplayUpdateTime = millis();
96  }
97
98  void loop() {
99    int packetSize = LoRa.parsePacket();
100   if (packetSize) {
101     processPacket(packetSize);
102   }
```

```
103
104   unsigned long currentTime = millis();
105   if (currentTime - lastDisplayUpdateTime >= 100) {
106     updateDisplay();
107     lastDisplayUpdateTime = currentTime;
108   }
109
110   delay(1);
111 }
112
113 void processPacket(int packetSize) {
114   String receivedData = "";
115   while (LoRa.available()) {
116     receivedData += (char)LoRa.read();
117   }
118
119   rssi = LoRa.packetRssi();
120   snr = LoRa.packetSnr();
121
122   unsigned long transmittedPacketId = extractValue(receivedData, "seq"
        ).toInt();
123   distance = extractValue(receivedData, "dist").toFloat();
124   int newTotal = extractValue(receivedData, "total").toInt();
125
126   if (lastDistance == -1.0 || distance > lastDistance) {
127     if (DEBUG_MODE) {
128       Serial.print("New␣session␣detected:␣resetting␣counters.␣(Old␣
            Distance:␣");
129       Serial.print(lastDistance);
130       Serial.print("␣->␣New␣Distance:␣");
131       Serial.print(distance);
132       Serial.println(")");
133     }
134     cumulativeReceived = 0;
135     lostPackets = 0;
136     totalPacketsSent = 0;
137     expectedSequence = transmittedPacketId + 1;
138     receivedFirstPacket = true;
139     lastDistance = distance;
140   }
141
```

```
142   if (transmittedPacketId == 0 && expectedSequence != 0) {
143     if (DEBUG_MODE) {
144       Serial.println("Received␣anomalous␣packet␣with␣sequence␣0;␣
              logging␣as␣new␣DATA␣and␣incrementing␣expectedSequence.");
145     }
146
147     currentPacketId = 0;
148     cumulativeReceived++;
149     totalPacketsSent = newTotal;
150     if (newTotal > 0) {
151       packetLossRate = 100.0 * lostPackets / newTotal;
152     }
153     outputLoggableData();
154     updateDisplay();
155     lastPacketTime = millis();
156
157     expectedSequence++;
158     return;
159   }
160
161   if (transmittedPacketId >= expectedSequence) {
162
163     if (transmittedPacketId > expectedSequence) {
164       unsigned long gap = transmittedPacketId - expectedSequence;
165       for (unsigned long missing = expectedSequence; missing <
              transmittedPacketId; missing++) {
166         Serial.print("LOST:␣Packet␣");
167         Serial.println(missing);
168       }
169       lostPackets += gap;
170     }
171     expectedSequence = transmittedPacketId + 1;
172   } else {
173
174     if (DEBUG_MODE) {
175       Serial.print("Out-of-order␣packet:␣received␣");
176       Serial.println(transmittedPacketId);
177     }
178   }
179
180   cumulativeReceived++;
```

```
181   currentPacketId = transmittedPacketId;
182   totalPacketsSent = newTotal;
183
184   if (newTotal > 0) {
185     packetLossRate = 100.0 * lostPackets / newTotal;
186   }
187
188   outputLoggableData();
189   updateDisplay();
190
191   lastPacketTime = millis();
192 }
193
194 String extractValue(String data, String key) {
195   String keyStr = "\"" + key + "\":";
196   int keyIndex = data.indexOf(keyStr);
197   if (keyIndex == -1) return "0";
198   int valueStart = keyIndex + keyStr.length();
199   int valueEnd = data.indexOf(",", valueStart);
200   if (valueEnd == -1) {
201     valueEnd = data.indexOf("}", valueStart);
202   }
203   if (valueEnd == -1) return "0";
204   return data.substring(valueStart, valueEnd);
205 }
206
207 void updateDisplay() {
208   display.clearDisplay();
209   display.setTextSize(1);
210   display.setTextColor(SSD1306_WHITE);
211   display.setCursor(0, 0);
212
213   display.print("Msg:␣");
214   display.println(currentPacketId);
215
216   display.setCursor(0, 12);
217   display.print("Dist:␣");
218   display.print(distance, 2);
219   display.println("␣m");
220
221   display.setCursor(0, 24);
```

```
222   display.print("Rcved/Lost: ");
223   display.print(cumulativeReceived);
224   display.print("/");
225   display.println(lostPackets);
226
227   display.setCursor(0, 36);
228   display.print("RSSI: ");
229   display.print(rssi);
230   display.println(" dBm");
231
232   display.setCursor(0, 48);
233   display.print("Loss: ");
234   display.print(packetLossRate, 1);
235   display.println("%");
236
237   display.display();
238 }
239
240 void outputLoggableData() {
241   Serial.print("DATA,");
242   Serial.print(millis());
243   Serial.print(",");
244   Serial.print(currentPacketId);
245   Serial.print(",");
246   Serial.print(distance);
247   Serial.print(",");
248   Serial.print(cumulativeReceived);
249   Serial.print(",");
250   Serial.print(lostPackets);
251   Serial.print(",");
252   Serial.print(rssi);
253   Serial.print(",");
254   Serial.print(snr);
255   Serial.print(",");
256   Serial.print(totalPacketsSent);
257   Serial.print(",");
258   Serial.println(packetLossRate, 1);
259 }
```

Listing 2: Código en C++

.3. Código de Raspberry Pi

```
import serial
import time
import csv
import os
import glob
from datetime import datetime
import argparse

CSV_HEADERS = ["Timestamp", "Event", "MessageNumber", "Distance(m)", "
    Received", "Lost", "RSSI", "SNR", "TotalPackets", "LossRate"]

def find_serial_port():
    acm_ports = glob.glob('/dev/ttyACM*')
    if acm_ports:
        print("Found ACM ports:", ", ".join(acm_ports))
        return acm_ports[0]
    usb_ports = glob.glob('/dev/ttyUSB*')
    if usb_ports:
        print("Found USB ports:", ", ".join(usb_ports))
        return usb_ports[0]
    print("No serial ports found. Retrying...")
    return None

def continuously_check_for_port(timeout=60):
    start_time = time.time()
    while time.time() - start_time < timeout:
        port = find_serial_port()
        if port:
            return port
        time.sleep(5)
    return None

def process_data_line(line):
    parts = line.split(",")
    if len(parts) < 10:
        print("Incomplete DATA line received:", line)
        return None
    try:
```

```
38          data = {
39              "Timestamp": datetime.now().strftime("%Y-%m-%d␣%H:%M:%S"),
40              "Event": "DATA",
41              "MessageNumber": parts[2],
42              "Distance(m)": parts[3],
43              "Received": parts[4],
44              "Lost": parts[5],
45              "RSSI": parts[6],
46              "SNR": parts[7],
47              "TotalPackets": parts[8],
48              "LossRate": parts[9].strip()
49          }
50
51          if data["TotalPackets"].strip() == "0":
52              data["TotalPackets"] = data["Received"]
53          return data
54      except Exception as e:
55          print("Error␣processing␣DATA␣line:", e)
56          return None
57
58  def process_lost_line(line):
59
60      try:
61          parts = line.split()
62          lost_packet = parts[-1]
63          data = {
64              "Timestamp": datetime.now().strftime("%Y-%m-%d␣%H:%M:%S"),
65              "Event": "LOST",
66              "MessageNumber": lost_packet,
67              "Distance(m)": "",
68              "Received": "",
69              "Lost": "",
70              "RSSI": "",
71              "SNR": "",
72              "TotalPackets": "",
73              "LossRate": ""
74          }
75          return data
76      except Exception as e:
77          print("Error␣processing␣LOST␣line:", e)
78          return None
```

```
79
80  def log_data(data, csv_log):
81      file_exists = os.path.isfile(csv_log)
82      try:
83          with open(csv_log, "a", newline="") as csvfile:
84              writer = csv.DictWriter(csvfile, fieldnames=CSV_HEADERS)
85              if not file_exists:
86                  writer.writeheader()
87              writer.writerow(data)
88          print("Logged data:", data)
89      except Exception as e:
90          print("Error writing to CSV:", e)
91
92  def main():
93      parser = argparse.ArgumentParser(description='LoRa Data Logger for
            DomH-giti')
94      parser.add_argument('--port', help='Serial port (e.g. /dev/ttyACM0
            )')
95      parser.add_argument('--baud', type=int, default=115200, help='Baud
            rate')
96      parser.add_argument('--log-dir', default=os.path.expanduser("~/
            lora_data_logs"), help='Directory for logs')
97      parser.add_argument('--wait-for-port', type=int, default=60, help=
            'Seconds to wait for serial port')
98      args = parser.parse_args()
99
100
101     log_dir = args.log_dir
102     os.makedirs(log_dir, exist_ok=True)
103
104     current_csv_log = None
105     last_distance = None
106     last_lost = None
107
108     print("Using log directory:", log_dir)
109     port = args.port
110     if not port:
111         print("Auto-detecting serial port...")
112         port = continuously_check_for_port(args.wait_for_port)
113     if not port:
114         print("ERROR: Could not find a serial port.")
```

```
115         return
116
117     print("Serial port:", port)
118
119     try:
120         ser = serial.Serial(port, args.baud, timeout=1)
121
122         ser.reset_input_buffer()
123     except Exception as e:
124         print("Failed to open serial port:", e)
125         return
126
127     print("Serial connection established on", port)
128     print("Waiting for data... (Press Ctrl+C to stop)")
129
130     while True:
131         try:
132             if ser.in_waiting:
133
134                 line = ser.readline().decode('utf-8', errors='replace'
                        ).strip()
135                 if line:
136                     print("RAW:", line)
137
138
139                     if line.startswith("Large gap detected"):
140                         print("Ignoring large gap message; not
                                updating Lost count.")
141                         continue
142
143
144                     if line.startswith("LOST:"):
145                         lost_data = process_lost_line(line)
146                         if lost_data and current_csv_log:
147                             log_data(lost_data, current_csv_log)
148                         else:
149                             print("No CSV log file created yet; LOST
                                    event not logged to CSV.")
150                         continue
151
152
```

```
153                     if line.startswith("DATA,"):
154                         data = process_data_line(line)
155                         if data is None:
156                             continue
157                         try:
158                             current_distance = float(data["Distance(m)
                                "])
159                         except Exception as e:
160                             print("Error converting distance to float:
                                ", e)
161                             continue
162 
163                         if last_distance is not None and
                            current_distance == 0.0 and last_distance
                            != 0.0:
164                             print(f"Transient 0.00 value detected,
                                using last valid Lost count: {last_lost
                                if last_lost is not None else 0}")
165                             data["Lost"] = str(last_lost if last_lost
                                is not None else 0)
166                         else:
167 
168                             try:
169                                 current_lost = int(float(data["Lost"])
                                    )
170                             except Exception:
171                                 current_lost = 0
172                             last_lost = current_lost
173 
174                         if last_distance is None:
175                             last_distance = current_distance
176                             timestamp = datetime.now().strftime("%Y-%m
                                -%d_%H-%M-%S")
177                             current_csv_log = os.path.join(log_dir, f"
                                lora_data_{current_distance:.2f}m_{
                                timestamp}.csv")
178                             print(f"New session detected: New distance
                                ({current_distance:.2f} m). Creating
                                log file: {current_csv_log}")
179                         elif current_distance <= last_distance and
                            current_distance != last_distance:
```

```
180                                 print(f"Transient␣or␣lower␣value␣detected␣
                                        (Current:␣{current_distance:.2f}␣m,␣
                                        Last:␣{last_distance:.2f}␣m).␣
                                        Continuing␣to␣log␣data␣in␣file␣for␣{
                                        last_distance:.2f}␣m")
181                             elif current_distance > last_distance:
182                                 last_distance = current_distance
183                                 timestamp = datetime.now().strftime("%Y-%m
                                        -%d_%H-%M-%S")
184                                 current_csv_log = os.path.join(log_dir, f"
                                        lora_data_{current_distance:.2f}m_{
                                        timestamp}.csv")
185                                 print(f"New␣distance␣detected␣({
                                        current_distance:.2f}␣m).␣Creating␣new␣
                                        log␣file:␣{current_csv_log}")
186
187                             if current_csv_log:
188                                 log_data(data, current_csv_log)
189                 else:
190                     time.sleep(0.1)
191             except KeyboardInterrupt:
192                 print("Exiting...")
193                 break
194             except Exception as e:
195                 print("Error␣reading␣from␣serial:", e)
196                 time.sleep(1)
197
198 if __name__ == "__main__":
199     main()
```

Listing 3: Código en Python